Formalization and study of statistical problems in Credit Scoring

Adrien Ehrhardt

▶ **To cite this version:**

Adrien Ehrhardt. Formalization and study of statistical problems in Credit Scoring: Reject inference, discretization and pairwise interactions, logistic regression trees. Methodology [stat.ME]. Université de Lille, 2019. English. NNT : . tel-02302691

HAL Id: tel-02302691
https://hal.science/tel-02302691

Submitted on 1 Oct 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Université de Lille
Crédit Agricole Consumer Finance - Inria Lille-Nord Europe

École doctorale **Sciences pour l'Ingénieur**
Unité de recherche **Équipe-projet MΘDAL**

Thèse présentée par **Adrien Ehrhardt**

Soutenue le **30 septembre 2019**

En vue de l'obtention du grade de docteur de l'Université de Lille

Discipline **Mathématiques et leurs interactions**
Spécialité **Statistique**

Titre de la thèse

Formalisation et étude de problématiques de scoring en risque de crédit

Inférence de rejet, discrétisation de variables et interactions, arbres de régression logistique

Thèse dirigée par	Christophe Biernacki	directeur
	Philippe Heinrich	co-encadrant
	Vincent Vandewalle	co-encadrant

Composition du jury

Rapporteurs	François Husson	professeur à l'Agrocampus Ouest	
	Jean-Michel Loubes	professeur à l'Université de Toulouse	
Examinateur	Camelia Goga	professeure à l'Université de Bourgogne Franche-Comté	présidente du jury
Invités	Jérôme Beclin	Crédit Agricole Consumer Finance	
	Hakim Djemmane	Crédit Agricole Consumer Finance	
Directeurs de thèse	Christophe Biernacki	professeur à l'Université de Lille	
	Philippe Heinrich	MCF à l'Université de Lille	
	Vincent Vandewalle	MCF à l'Université de Lille	

Université de Lille
Crédit Agricole Consumer Finance - Inria Lille-Nord Europe

École doctorale **Sciences pour l'Ingénieur**
Unité de recherche **Équipe-projet MΘDAL**

Thèse présentée par **Adrien Ehrhardt**

Soutenue le **30 septembre 2019**

En vue de l'obtention du grade de docteur de l'Université de Lille

Discipline **Mathématiques et leurs interactions**
Spécialité **Statistique**

Titre de la thèse

Formalisation et étude de problématiques de scoring en risque de crédit

Inférence de rejet, discrétisation de variables et interactions, arbres de régression logistique

Thèse dirigée par	Christophe Biernacki	directeur
	Philippe Heinrich	co-encadrant
	Vincent Vandewalle	co-encadrant

Composition du jury

Rapporteurs	François Husson	professeur à l'Agrocampus Ouest	
	Jean-Michel Loubes	professeur à l'Université de Toulouse	
Examinateur	Camelia Goga	professeure à l'Université de Bourgogne Franche-Comté	présidente du jury
Invités	Jérôme Beclin	Crédit Agricole Consumer Finance	
	Hakim Djemmane	Crédit Agricole Consumer Finance	
Directeurs de thèse	Christophe Biernacki	professeur à l'Université de Lille	
	Philippe Heinrich	MCF à l'Université de Lille	
	Vincent Vandewalle	MCF à l'Université de Lille	

Université de Lille
Crédit Agricole Consumer Finance - Inria Lille-Nord Europe

Doctoral School **Sciences pour l'Ingénieur**
University Department **Équipe-projet MΘDAL**

Thesis defended by **Adrien Ehrhardt**

Defended on **30th September, 2019**

In order to become Doctor from Université de Lille

Academic Field **Applied Mathematics**
Speciality **Statistics**

Thesis Title

Formalization and study of statistical problems in Credit Scoring

Reject inference, discretization and pairwise interactions, logistic regression trees

Thesis supervised by

Christophe Biernacki	Supervisor
Philippe Heinrich	Co-Advisor
Vincent Vandewalle	Co-Advisor

Committee members

Referees	François Husson	Professor at Agrocampus Ouest	
	Jean-Michel Loubes	Professor at Université de Toulouse	
Examiner	Camelia Goga	Professor at Université de Bourgogne Franche-Comté	Committee President
Guests	Jérôme Beclin	Crédit Agricole Consumer Finance	
	Hakim Djemmane	Crédit Agricole Consumer Finance	
Supervisors	Christophe Biernacki	Professor at Université de Lille	
	Philippe Heinrich	Associate Professor at Université de Lille	
	Vincent Vandewalle	Associate Professor at Université de Lille	

L'Université de Lille n'entend donner aucune approbation ni improbation aux opinions émises dans les thèses : ces opinions devront être considérées comme propres à leurs auteurs.

Mots clés : scoring, risque, crédit, prédiction, discrétisation, segmentation
Keywords: scoring, credit, risk, prediction, discretization, clustering

Cette thèse a été préparée dans les laboratoires suivants.

Équipe-projet MΘDAL
Inria Lille Nord-Europe
40 Avenue Halley
59650 Villeneuve-d'Ascq

☎ +33 (0)3 59 57 78 00
+33 (0)3 59 57 78 50
contact-lille@inria.fr
Site https://www.inria.fr/centre/lille

Laboratoire Paul Painlevé
CNRS U.M.R. 8524
59655 Villeneuve d'Ascq Cedex
France

☎ (+33) 03 20 43 48 50
Site https://math.univ-lille1.fr/

À ma famille,

À mes amis,

À mes lecteurs.

The task of the human brain remains what it has always been ; that of discovering new data to be analyzed, and of devising new concepts to be tested.

Isaac Asimov, *I, Robot*

J'respecte R.

Damso

Formalisation et étude de problématiques de scoring en risque de crédit
Inférence de rejet, discrétisation de variables et interactions, arbres de régression logistique

Résumé

Cette thèse se place dans le cadre des modèles d'apprentissage automatique de classification binaire. Le cas d'application est le scoring de risque de crédit. En particulier, les méthodes proposées ainsi que les approches existantes sont illustrées par des données réelles de Crédit Agricole Consumer Finance, acteur majeur en Europe du crédit à la consommation, à l'origine de cette thèse grâce à un financement CIFRE.
Premièrement, on s'intéresse à la problématique dite de "réintégration des refusés". L'objectif est de tirer parti des informations collectées sur les clients refusés, donc par définition sans étiquette connue, quant à leur remboursement de crédit. L'enjeu a été de reformuler cette problématique industrielle classique dans un cadre rigoureux, celui de la modélisation pour données manquantes. Cette approche a permis de donner tout d'abord un nouvel éclairage aux méthodes standards de réintégration, et ensuite de conclure qu'aucune d'entre elles n'était réellement à recommander tant que leur modélisation, lacunaire en l'état, interdisait l'emploi de méthodes de choix de modèles statistiques.
Une autre problématique industrielle classique correspond à la discrétisation des variables continues et le regroupement des modalités de variables catégorielles avant toute étape de modélisation. La motivation sous-jacente correspond à des raisons à la fois pratiques (interprétabilité) et théoriques (performance de prédiction). Pour effectuer ces quantifications, des heuristiques, souvent manuelles et chronophages, sont cependant utilisées. Nous avons alors reformulé cette pratique courante de perte d'information comme un problème de modélisation à variables latentes, revenant ainsi à une sélection de modèle. Par ailleurs, la combinatoire associé à cet espace de modèles nous a conduit à proposer des stratégies d'exploration, soit basées sur un réseau de neurone avec un gradient stochastique, soit basées sur un algorithme de type EM stochastique.
Comme extension du problème précédent, il est également courant d'introduire des interactions entre variables afin, comme toujours, d'améliorer la performance prédictive des modèles. La pratique classiquement répandue est de nouveau manuelle et chronophage, avec des risques accrus étant donnée la surcouche combinatoire que cela engendre. Nous avons alors proposé un algorithme de Metropolis-Hastings permettant de rechercher les meilleures interactions de façon quasi-automatique tout en garantissant de bonnes performances grâce à ses propriétés de convergence standards.
La dernière problématique abordée vise de nouveau à formaliser une pratique répandue, consistant à définir le système d'acceptation non pas comme un unique score mais plutôt comme un arbre de scores. Chaque branche de l'arbre est alors relatif à un segment de population particulier. Pour lever la sous-optimalité des méthodes classiques utilisées dans les entreprises, nous proposons une approche globale optimisant le système d'acceptation dans son ensemble. Les résultats empiriques qui en découlent sont particulièrement prometteurs, illustrant ainsi la flexibilité d'un mélange de modélisation paramétrique et non paramétrique.
Enfin, nous anticipons sur les futurs verrous qui vont apparaître en Credit Scoring et qui sont pour beaucoup liés la grande dimension (en termes de prédicteurs). En effet, l'industrie financière investit actuellement dans le stockage de données massives et non structurées, dont la prochaine utilisation dans les règles de prédiction devra s'appuyer sur un minimum de garanties théoriques pour espérer atteindre les espoirs de performance prédictive qui ont présidé à cette collecte.

Mots clés : scoring, risque, crédit, prédiction, discrétisation, segmentation

Équipe-projet MΘDAL
Inria Lille Nord-Europe – 40 Avenue Halley – 59650 Villeneuve-d'Ascq

Formalization and study of statistical problems in Credit Scoring
Reject inference, discretization and pairwise interactions, logistic regression trees

Abstract

This manuscript deals with model-based statistical learning in the binary classification setting. As an application, credit scoring is widely examined with a special attention on its specificities. Proposed and existing approaches are illustrated on real data from Crédit Agricole Consumer Finance, a financial institute specialized in consumer loans which financed this PhD through a CIFRE funding.

First, we consider the so-called reject inference problem, which aims at taking advantage of the information collected on rejected credit applicants for which no repayment performance can be observed (*i.e.* unlabelled observations). This industrial problem led to a research one by reinterpreting unlabelled observations as an information loss that can be compensated by modelling missing data. This interpretation sheds light on existing reject inference methods and allows to conclude that none of them should be recommended since they lack proper modelling assumptions that make them suitable for classical statistical model selection tools.

Next, yet another industrial problem, corresponding to the discretization of continuous features or grouping of levels of categorical features before any modelling step, was tackled. This is motivated by practical (interpretability) and theoretical reasons (predictive power). To perform these quantizations, *ad hoc* heuristics are often used, which are empirical and time-consuming for practitioners. They are seen here as a latent variable problem, setting us back to a model selection problem. The high combinatorics of this model space necessitated a new cost-effective and automatic exploration strategy which involves either a particular neural network architecture or a Stochastic-EM algorithm and gives precise statistical guarantees.

Third, as an extension to the preceding problem, interactions of covariates may be introduced in the problem in order to improve the predictive performance. This task, up to now again manually processed by practitioners and highly combinatorial, presents an accrued risk of misselecting a "good" model. It is performed here with a Metropolis-Hastings sampling procedure which finds the best interactions in an automatic fashion while ensuring its standard convergence properties, thus good predictive performance is guaranteed.

Finally, contrary to the preceding problems which tackled a particular scorecard, we look at the scoring system as a whole. It generally consists of a tree-like structure composed of many scorecards (each relative to a particular population segment), which is often not optimized but rather imposed by the company's culture and / or history. Again, *ad hoc* industrial procedures are used, which lead to suboptimal performance. We propose some lines of approach to optimize this logistic regression tree which result in good empirical performance and new research directions illustrating the predictive strength and interpretability of a mix of parametric and non-parametric models.

This manuscript is concluded by a discussion on potential scientific obstacles, among which the high dimensionality (in the number of features). The financial industry is indeed investing massively in unstructured data storage, which remains to this day largely unused for *Credit Scoring* applications. Doing so will need statistical guarantees to achieve the additional predictive performance that was hoped for.

Keywords: scoring, credit, risk, prediction, discretization, clustering

Remerciements

Aboutissement d'un travail personnel, cette thèse n'en est pas moins une réussite collective et la contribution de nombreuses personnes, injustement absente de la page de couverture de ce manuscrit, doit ici être extensivement mentionnée.

Tout d'abord, je suis persuadé que le principal facteur de succès d'une thèse CIFRE est l'implication de l'entreprise d'accueil, de la conception du sujet à l'usage des fruits du travail de recherche. A ce titre, je remercie Crédit Agricole Consumer Finance de m'avoir permis de réaliser cette thèse dans de très bonnes conditions. En particulier, j'ai eu la chance d'interagir avec des managers réceptifs à la démarche de recherche et qui m'ont fait confiance : un grand merci à Jérôme Beclin et Nicolas Borde. Je me dois également de saluer la probité intellectuelle de Sébastien Beben ; nos riches échanges de début de thèse constituent sans doute le carburant de ce doctorat.

Haut-lieu de la recherche publique française, Inria m'a permis, en acceptant d'être le laboratoire d'accueil de cette CIFRE, de compléter ma formation d'ingénieur généraliste centralien en tentant de combler le vide technique ressenti en fin de cursus, ce qui m'avait motivé à poursuivre en thèse. Je vous laisse le soin, chers lecteurs, d'apprécier l'éventuelle réussite de cet objectif initial. Je remercie chaleureusement le centre de Lille et plus particulièrement l'équipe-projet MΘDAL pour m'avoir permis de (re)connaître la beauté des mathématiques. Contributeurs directs et véritables artisans de ce travail de recherche, mes trois co-directeurs de thèse ont constitué le moteur de cette thèse ; merci à Christophe Biernacki dont j'espère garder la rigueur scientifique ; merci à Philippe Heinrich, pour m'avoir montré qu'un problème bien posé est déjà à moitié résolu ; merci à Vincent Vandewalle, dont les éclairages passionés, à grands coups de feutre virevoltant sur le tableau ou scripts R envoyés au milieu de la nuit, ont pour la plupart donné la vitesse initiale à chaque partie de ce manuscrit.

Enfin, il convient de saluer la part de responsabilité de mes proches dans ce travail et les quelques mots qui suivront sont bien peu de choses en comparaison de tout ce qu'ont pu apporter ma famille et mes amis dans ma formation intellectuelle au sens large. Un immense merci revient tout d'abord à mes parents, ils m'ont tout donné. Merci également à tous mes amis, de nombreux rires restent à partager. Merci bien sûr à ma femme Valentine qui m'a toujours soutenu ; je suis très fier de tout ce que nous avons accompli. Puisse-t-on traverser la vie comme ces dix dernières années.

Cher lecteur, merci de parcourir le manuscrit que tu tiens entre les mains ; sans toi, il n'existerait pas. Bonne lecture.

Sommaire

Liste des tableaux

Table des figures

Glossaire

C | L | S

C

crédit affecté Le crédit affecté est accordé par un établissement de crédit ou une banque. Il est utilisé pour un achat déterminé : un bien mobilier (crédit automobile par exemple) ou une prestation. Il est souvent contracté directement sur le lieu de vente. Généralement, le défaut du crédit entraîne la récupération du bien sous-jacent par un huissier. 4

crédit classique Les conditions du prêt sont fixées à l'avance, lors de la signature du contrat. Le taux, la durée, et les mensualités du prêt sont fixes. Le coût total du financement est ainsi connu dès le début du prêt. 3

crédit renouvelable Le crédit renouvelable, encore appelé crédit permanent, crédit revolving ou crédit reconstituable, consiste à mettre à la disposition d'un emprunteur une réserve d'argent qu'il pourra utiliser et reconstituer selon son gré. Ce crédit est proposé par un établissement financier ou une enseigne commerciale. Il peut être couplé avec une carte de crédit et peut être couvert par une assurance. 4

cut Le cut d'un score est un seuil qui représente la note (ou, de façon équivalente, la probabilité) à partir de laquelle un client est accepté ; en-dessous de celui-ci, le client est jugé trop risqué et il est refusé. 11, 12, 34

L

location La location est elle-même commercialisée sous deux formes : la location avec option d'achat (L.O.A.), pour laquelle le client peut décider d'acquérir le bien loué en fin d'échéancier pour un montant d'option d'achat fixé à l'avance et la location longue-durée (L.L.D.) pour laquelle c'est le magasin / concessionnaire qui dispose d'une option d'achat. 4

S

score Le score désigne la "fonction" qui attribue une note, témoignant de la propension à rembourser le crédit, d'un demandeur en fonction de ses caractéristiques. 8, 10–13, 23, 27, 29, 31–33

Acronymes

C | E | F | L | M | P | S

C

CACF Crédit Agricole Consumer Finance. 1, 3–5, 7, 20, 23, 32–37, 40, 46, 57, 61, 62, 66, 74, 77, 80–82, 84, 103, 106, 110–112, 115, 119, 120
CEM Classification Expectation Maximization. 30

E

EM Expectation Maximization. 29, 30, 33, 55, 56, 94–97

F

FAMD Factor Analysis of Mixed Data. 80, 83, 84, 86, 99, 102, 109

L

LMT Logistic Model Trees. 88, 89, 91, 95, 99, 101, 102, 127, 128
LOTUS Logistic Tree with Unbiased Selection. 87, 88, 90, 91

M

MAR missing at random. 27, 29–32, 35, 36, 62, 118, 119
MCA Multiple Correspondence Analysis. 80–83, 86, 87, 102
MCAR missing completely at random. 27, 35
MLE Maximum Likelihood Estimation. 31, 55, 56, 67, 74, 94, 95
MNAR missing not at random. 25, 27, 32, 35, 36, 62, 118
MOB Model-Based Recursive Partitioning. 89–91, 99, 101, 102, 128

P

PCA Principal Component Analysis. 80–84, 86, 100, 102
pdf densité de probabilité. 13, 18, 69, 83, 95, 108
PLS Partial Least Squares. 86, 87, 99, 100, 102, 109, 127

S

SEM Stochastic Expectation Maximization. 37, 39, 52, 53, 55–60, 63, 68, 69, 73, 83, 95–99, 102, 103, 111, 125, 139
SPC Supervised Principal Components. 87, 109, 128

Notations

Espaces

Terme	Description	Pages
$\mathbb{R}$	nombres réels	5, 10, 13, 20, 44, 47, 49, 50, 90, 91
$\mathbb{N}$	entiers naturels	44, 90, 91
$\mathbb{N}_{l_j}$	entiers naturels de 1 à l_j	5, 10, 13, 20, 43, 90
$\mathcal{X}$	espace du vecteur de caractéristiques (on se limite au produit de l'ensemble des réelles -variables continues- et des entiers naturels -variables catégorielles)	5, 10, 13, 15, 109
$\boldsymbol{Q}$	espace des quantifications multivariées	44, 45, 47–49, 55, 67
$\mathcal{Q}_{\boldsymbol{m}}$	espace des quantifications multivariées à $\boldsymbol{m}$ modalités	44
$\mathcal{Q}_j$	espace des quantifications univariées de la variable j	45
$\mathcal{Q}_{j,m_j}$	espace des quantifications univariées de la variable j à m_j modalités	44
F	ensemble des indices de clients financés	25–27, 31, 114
NF	ensemble des indices de clients non financés	25, 26, 29, 30

Variables aléatoires

Terme	Description	Pages
X	variable aléatoire	13, 18
$\boldsymbol{X}$	vecteur aléatoire de caractéristiques à d composantes continues ou catégorielles	5, 10, 11, 13–15, 17, 31, 80
X_j	$j^{\text{ème}}$ variable aléatoire de $\boldsymbol{X}$	5, 10, 11, 13, 45
Y	variable aléatoire binaire dissociant "bons" et "mauvais" clients	8, 10, 11, 13, 15, 17, 31, 35, 80
Z	variable aléatoire binaire dissociant clients "financés" et "non financés"	13
C	variable aléatoire latente du segment auquel appartient un client	90

Scalaires

Terme	Description	Pages
x	réalisation de X	5, 13, 23, 39, 42–45, 47, 49–59, 65–67, 71, 73, 74, 81, 87, 91, 98–100, 120, 121
$\boldsymbol{x}$	réalisation du vecteur de caractéristiques d'un client	5, 10, 11, 13–18, 20, 23–35, 42, 43, 45–47, 49, 50, 52, 53, 55, 65, 66, 71, 73, 80–83, 86–88, 90–97, 113–115, 117, 118, 125, 127, 128
l_j	nombre de modalités associés à la variable qualitative X_j	5, 10, 11, 13, 43, 45, 49–51, 54, 56, 88, 90
$x_{i,j}$	réalisation de X_j pour le $i^{\text{ème}}$ client (en colonne)	5, 11, 44, 51, 53–55, 97
$\mathbf{x}$	matrice de design de caractéristiques (en colonnes) d'un ensemble de clients (en lignes)	5, 7, 10, 13, 15–19, 25, 30, 44, 45, 47, 51–53, 55–57, 69–75, 79, 81, 82, 86, 88, 90, 92–98, 113–118, 120–128
y	observation du caractère "bon" (1) ou "mauvais" (0) d'un client	13, 17, 23–35, 39, 46, 47, 50, 55, 57, 58, 67, 71, 74, 94, 96, 97, 113
$\mathbf{y}$	colonne de réponses associées à la matrice de design $\mathbf{x}$	10, 13, 15–17, 19, 20, 25, 26, 28–31, 33, 35, 36, 47, 53, 55, 56, 69–74, 79, 86, 88, 90, 92, 94–98, 102, 114–118, 121, 123–125, 127, 128
z	observation du caractère "financé" (f) ou "non financé" (nf) d'un client	13, 24–29, 31, 32, 34, 114, 117
$\mathbf{z}$	colonne de décisions d'acceptation / rejet associées à la matrice de design $\mathbf{x}$	13, 25, 28, 30
$\mathcal{T}$	ensemble d'observations d'apprentissage	13, 14, 17, 25–31, 33
$\mathcal{T}_{\text{f}}$	ensemble d'observations de clients financés	25, 27–29, 31–33, 44, 45, 47, 49–51, 53, 67, 88, 90, 113
$\mathcal{T}_{\text{nf}}$	ensemble d'observations de clients non financés	25, 28, 29
$\boldsymbol{m}$	vecteur des nombres de modalités associés à chaque quantification	44, 51–55, 57, 58, 73, 120, 121
m_j	nombre de modalités associé à la quantification de la $j^{\text{ème}}$ composante	43–45, 47–52, 54, 56, 58, 66, 73, 120, 121
$\boldsymbol{e}_h^{m_j}$	$h^{\text{ième}}$ vecteur de base de $\mathbb{R}^{m_j}$: $(0,\ldots,0,1,0,\ldots,0)$	49, 50, 53, 54, 56, 57, 74

Terme	Description	Pages
δ	matrice d'interactions dont l'entrée $\delta_{k,\ell}$ encode l'interaction (1) ou l'absence d'interaction (0) entre les variables k et ℓ	65–75, 80, 90, 97, 98
K	nombre de segments de clients	90, 92, 94, 95, 97–99, 102
c	observation du segment d'un client	90–95, 97

Fonctions

Terme	Description	Pages
q	fonction de quantification multivariée	43–52, 54–59, 65–67, 69–75, 80, 90, 96, 97, 120–122, 124, 126
q_j	fonction de quantification univariée de la $j^{\text{ème}}$ composante	43–45, 47, 48, 65
KL	divergence de Kullback-Leibler	14, 15, 19, 31, 113, 115

Paramètres

Terme	Description	Pages
θ	vecteur de paramètres, généralement pour la régression logistique, du modèle prédictif de Y conditionnellement aux caractéristiques $\boldsymbol{X}$	10, 11, 14–20, 25–27, 29–33, 39, 45–51, 53–57, 65–69, 71–76, 80, 81, 90, 92, 94–98, 113–117, 124, 125
$\theta^\star$	vrai (bon modèle) ou "meilleur" (mauvais modèle) vecteur de paramètres au sens de la divergence de Kullback-Leibler	14, 15, 17, 18, 29, 35, 53
ϕ	paramètre du mécanisme d'acceptation / rejet (Z) des demandeurs de crédit conditionnellement aux données $\boldsymbol{X}$ et Y	25–27, 30, 32, 34, 36, 114–117
α	paramètre de relaxation du problème de quantification	48–51, 54–58, 73, 74, 96–98, 124, 125
β	paramètre de relaxation du problème de segmentation	93–98

Liste des Algorithmes

Avant-propos

> Anyone who has ever struggled with poverty knows how extremely expensive it is to be poor.
>
> James A. Baldwin

Les cas d'application des travaux de ce manuscrit portent sur plusieurs problèmes connexes au *Credit Scoring*.

Pour un particulier, le recours au crédit, c'est-à-dire à l'emprunt d'argent en échange d'une promesse de remboursement étalé dans le temps et assorti d'un intérêt, est possible depuis très longtemps, les plus anciennes traces "modernes" de crédits bancaires se situant au XII^{ème} siècle en Italie [4]. De nos jours, l'emprunt immobilier ou automobile, c'est-à-dire pour financer un lieu de résidence ou l'achat d'un véhicule, est répandu [1]. Par opposition au crédit immobilier, on parle souvent de crédit à la consommation pour désigner le financement de biens et de services : automobile, électroménager, travaux, *etc*. De manière plus formelle, le crédit à la consommation est définie dans la loi №2010-737 du 1er juillet 2010 [2] comme une :

> Opération ou contrat de crédit, une opération ou un contrat par lequel un prêteur consent ou s'engage à consentir à l'emprunteur un crédit sous la forme d'un délai de paiement, d'un prêt, y compris sous forme de découvert ou de toute autre facilité de paiement similaire, à l'exception des contrats conclus en vue de la fourniture d'une prestation continue ou à exécution successive de services ou de biens de même nature et aux termes desquels l'emprunteur en règle le coût par paiements échelonnés pendant toute la durée de la fourniture.

De nombreux acteurs bancaires proposent des crédits à la consommation, si bien qu'en 2017 environ 27,2 % des ménages ont un crédit à la consommation [3]. Crédit Agricole Consumer Finance (CACF), à l'origine de cette thèse CIFRE, est un acteur majeur du crédit à la consommation, à travers une marque spécialisée en France, Sofinco, et des partenaires distributeurs de crédit conso.

Parmi l'ensemble des demandeurs de crédit à la consommation, il est souhaitable, à plusieurs égards, de ne pas financer tous les crédits. Premièrement, si tant est que l'on puisse prêter un rôle sociétal à une entité bancaire, il paraît responsable de ne pas détériorer voire mettre en danger la santé financière de l'emprunteur. Pour ce faire, des contrôles automatiques permettent de refuser la clientèle dite fragile : taux d'endettement trop élevé, fichage bancaire pour incidents de paiements, ... Par ailleurs, d'un point de vue économique cette fois, un client se trouvant dans l'incapacité de rembourser le crédit souscrit sera vraisemblablement peu ou pas profitable pour l'institution financière du fait des coûts de traitements et de personnels de relance et procédure(s) judiciaire(s) qui peuvent aboutir à une annulation totale ou partielle de la dette du client engendrant une perte sèche pour l'organisme prêteur.

Dans ce cadre, le score vise à évaluer la propension d'un client à être "bon" ou "mauvais", selon des critères à définir ultérieurement, pour ainsi prendre une décision de financement ou de rejet de façon quantitative et objective. On donnera dans le chapitre 1 quelques éléments de contexte supplémentaires nécessaires à la bonne compréhension des cas d'application de cette thèse, un état de l'art de la pratique industrielle ainsi qu'un état de l'art académique des techniques d'apprentissage transposables au *Credit Scoring*.

Le chapitre 2 est consacré à l'étude du problème de "Réintégration des refusés" (ou *Reject Inference*) qui peut être réinterprété comme un problème de biais d'échantillon comme on peut en trouver dans les sondages par exemple, sur la variable à prédire. En effet, le système d'acceptation en place ayant déjà pour but de refuser la clientèle risquée, la clientèle en portefeuille servant d'échantillon au statisticien pour dériver de nouvelles règles de classement est bien moins risquée que la population totale des demandeurs ; c'est en ce sens qu'elle est biaisée.

Ce problème d'échantillonnage résolu, il paraît naturel au statisticien de s'atteler à la modélisation : quelle relation existe-t-il entre les caractéristiques de l'emprunteur et la quantité de risque qu'il présente ? Le chapitre 1 aura mis en avant la nature des caractéristiques disponibles ainsi que certaines faiblesses statistiques de la procédure actuelle : les chapitres 3 et 4 présentent une nouvelle méthode de recherche et de sélection du meilleur modèle dans la famille imposée par le cas d'application.

Le chapitre 5 prend du recul sur les chapitres précédents et remet le problème du *Credit Scoring* au niveau du système d'acceptation dans sa globalité. En effet, on verra au chapitre 1 que plusieurs scores sont en place sur des segments de clientèle différents ; un score spécifique peut être dédié aux crédits automobiles par exemple. Chacun de ces scores est optimal localement, sur son segment. En revanche, les techniques de construction des segments reposant sur des heuristiques relativement éloignés de l'objectif de prédiction, le système global est *a priori* sous-optimal. On formalisera cette architecture de "mélange d'experts" pour proposer des solutions adaptées de la littérature statistique à l'optimisation globale du système d'acceptation.

Enfin, l'émergence récente du *Big Data* n'échappe pas au monde du crédit à la consommation. Le chapitre 1 aura mis en évidence que les pratiques industrielles sont souvent peu formalisées, ce qui justifie la présente thèse, et ne seront pas adaptées à la grande dimension en termes de prédicteurs. En conséquence, pour répondre au double objectif d'utilisation à court-terme de données dites *web* (*e.g.* cookies, clics, logs) et d'éviter de se résoudre à des procédures *ad hoc* qui nécessiteraient une formalisation ultérieure, des premières directions d'étude pour l'utilisation raisonnée de telles données dans le cadre du *Credit Scoring* sont données dans la Conclusion and prospects.

Références de l'avant-propos

[1] *Les Français recourent toujours largement au crédit pour acheter leur voiture*. Oct. 2010. URL : https://www.latribune.fr/vos-finances/banques-credit/credit-auto-moto/20101007trib000556639/les-francais-recourent-toujours-largement-au-credit-pour-acheter-leur-voiture.html.

[2] *LOI n° 2010-737 du 1er juillet 2010 portant réforme du crédit à la consommation (1)*. Juil. 2010. URL : https://www.legifrance.gouv.fr/affichTexte.do?cidTexte=JORFTEXT000022419094&categorieLien=id.

[3] Michel Mouillart. *Observatoire des crédits aux ménages*. Rapp. tech. Fédération Bancaire Française, jan. 2018.

[4] H. Thomas. *The Wards of London : Comprising a Historical and Topographical Description of Every Object of Importance Within the Boundaries of the City. With an Account of All the Companies, Institutions, Buildings, Ancient Remains ... and Biographical Sketches of All Eminent Persons Connected Therewith*. The Wards of London : Comprising a Historical and Topographical Description of Every Object of Importance Within the Boundaries of the City. With an Account of All the Companies, Institutions, Buildings, Ancient Remains ... and Biographical Sketches of All Eminent Persons Connected Therewith vol. 1 à 2. J. Gifford, 1828. URL : https://books.google.fr/books?id=PDMQAAAAYAAJ.

Chapitre 1

Apprendre des demandes de crédit à la consommation

Les hommes mentent mais pas les chiffres.

Kaaris

Sommaire

Ce chapitre est destiné à poser les bases de l'apprentissage statistique dans le cadre des crédits à la consommation. On introduira dans une première partie la terminologie consacrée du crédit à la consommation avant de s'attarder plus en détails, dans une seconde partie, sur l'état de l'art industriel du *Credit Scoring* à travers une étude bibliographique et la pratique de CACF. On clotûrera le chapitre par une troisième partie, la plus traditionnelle pour débuter un manuscrit de thèse, à savoir l'état de l'art académique de l'apprentissage statistique, en nous limitant bien entendu aux cas d'usage spécifiques aux crédits à la consommation mis en exergue dans les deux premières parties de ce chapitre.

1.1 Le marché du crédit à la consommation : quels enjeux ?

S'agissant d'une thèse CIFRE, il apparaît comme nécessaire de planter le décor industriel de la problématique. Dans cette première partie, on verra succintement le coeur du métier de CACF, les produits que l'entreprise propose et l'environnement dans lequel elle s'insert.

1.1.1 Qu'est-ce qu'un crédit à la consommation ?

La définition légale en a été donnée en avant-propos. En pratique, on peut distinguer trois produits de crédit à la consommation.

Le premier d'entre eux, le crédit classique, est le produit historique. De la même manière qu'un crédit immobilier, le client emprunte une somme fixe qui lui est attribuée au moment du financement et qu'il rembourse selon un échéancier défini à l'avance (taux et nombre de mensualités fixes). D'un point de vue statistique, le traitement est relativement simple : que ce soit à l'octroi, pour déterminer le risque du client, ou au cours de la vie du dossier, pour provisionner les pertes potentielles, tout est connu à l'avance. Il suffit en quelque sorte de vérifier le paiement de la mensualité à la date prévue. Il convient également de préciser que certains crédits classiques sont dits crédits affectés, c'est-à-dire qu'ils financent un bien précis et identifié, de sorte que le prêt transite directement de l'organisme prêteur au vendeur (un concessionnaire par exemple). Par ailleurs, la mise en défaut du crédit entraîne généralement une procédure de recouvrement de la dette qui peut se solder, dans le cas d'un crédit affecté, par la récupération du bien par un huissier. Là encore, d'un point de vue statistique, il paraît indispensable de consigner les caractéristiques du bien sous-jacent afin d'intégrer sa valeur résiduelle récupérable en cas de défaut.

Le second produit, développé à partir de 1965 en France et ayant connu une forte croissance depuis [7] mais néanmoins bien moins répandu en Europe qu'aux Etats-Unis par exemple [22], est le crédit renouvelable. Un capital dit accordé ou autorisé est attribué au demandeur qui peut utiliser tout ou partie de ce montant et le rembourse à un taux et par mensualités dépendants tous deux de la proportion du capital consommé. Au fur et à mesure du remboursement du capital emprunté, le capital "empruntable", c'est-à-dire la différence entre le capital accordé et le capital emprunté à date, se reconstitue et de nouvelles utilisations sont possibles, toujours dans la limite du capital accordé au départ. D'un point de vue statistique à nouveau, plusieurs problèmes se posent du fait du caractère intrinsèquement aléatoire de l'utilisation ou non de tout ou partie de la ligne de crédit accordée. Plus précisément, ce produit présente un risque important porté par deux facteurs : premièrement, le taux élevé attire des clients risqués, au taux de défaut plus élevé que pour un crédit classique par exemple ; deuxièmement, ces crédits portent un risque dit de hors-bilan très fort, puisqu'à tout moment, l'ensemble des crédits accordés mais non utilisés et donc non comptabilisés "au bilan", c'est-à-dire comme une dette du client envers l'établissement bancaire, peuvent être utilisés et faire défaut. La mauvaise quantification de ce risque est à présent reconnu comme un important catalyseur de la récente crise financière [14].

Enfin, la location a récemment connu un essor important [18]. D'abord concentrée sur le secteur automobile, elle se développe actuellement pour les produits électroniques (smartphones notamment) et même plus récemment pour des produits plus insolites comme les matelas [6]. Comme le crédit affecté, il est important de prendre en compte les données du bien loué afin d'évaluer le risque que porte ce produit, la difficulté supplémentaire reposant sur l'éventualité de l'exercice de l'hypothétique option d'achat.

De cette partie, deux considérations statistiques doivent retenir notre attention : d'abord, ces différents produits nécessitent des traitements différents dans la mesure où leur risque est intrinsèquement différent ; ensuite, les données disponibles pour chacun de ces produits

diffèrent : par exemple, les données du produit financé ne sont disponibles que pour les crédits affectés et les locations. Cette dernière notion de "blocs" de variables est au coeur du chapitre 5.

1.1.2 Crédit Agricole Consumer Finance

CACF opère dans de nombreux pays. En France, c'est principalement à travers la marque Sofinco que sont commercialisés les crédits à la consommation pour lesquels il existe une relation directe entre CACF et le client (dite B2C), par exemple lorsqu'un demandeur se rend directement sur le site internet sofinco.fr.

Par ailleurs, de nombreux crédits à la consommation sont distribués à travers un réseau de partenaires, qui jouent le rôle d'intermédiaires (on parle alors de B2B) : concessionnaires automobile, distributeurs d'électroménager, etc.

Enfin, CACF faisant partie du groupe Crédit Agricole, de nombreuses agences bancaires distribuent des crédits à la consommation à leur clientèle bancarisé, par l'intermédiaire des gestionnaires de compte.

Là encore, on constate que les spécificités des canaux de distribution des crédits impactent grandement la collecte des données et leur traitement statistique. En effet, les informations collectées sur le client, le produit et éventuellement l'apporteur d'affaires sont différentes selon le canal.

Dans la partie suivante, la méthodologie présentée est spécifique à CACF ; il pourra néanmoins être admis que, dans les grandes lignes, cette méthodologie est similaire à la concurrence d'une part, et à la pratique d'autres pays (européens du moins) puisque la législation sur la protection et le traitement des données est sensiblement similaire (du fait de l'entrée en vigueur récente de la GDPR) et le fait que les établissements bancaires possèdent généralement des filiales dans plusieurs pays d'Europe et y font appliquer la même méthodologie.

1.2 Le *Credit Scoring* : état de l'art de la pratique industrielle

Cette partie vise à présenter la pratique actuelle en matière de *Credit Scoring* et pose un certain nombre de questions statistiques dont certaines ont été traitées dans cette thèse, d'autres trouvent des réponses (parfois partielles) dans la littérature et dont certaines références sont données à titre informatif mais ne sont pas développées dans ce manuscrit ; enfin, certaines questions ne trouvent *a priori* pas de réponse immédiate dans la littérature et sont autant de matière à de futurs travaux dans le domaine !

1.2.1 Collecte des données

La partie précédente a mis en exergue la pluralité des sources de données, schématisées en figure 1.1 : Crédit Agricole, à travers sa filiale dédiée aux crédits à la consommation Crédit Agricole Consumer Finance, finance des crédits en France à travers sa marque Sofinco (B2C), ou en magasins / concessions chez des partenaires (B2B) où les données du demandeur de crédit sont collectées. La figure 1.2 présente par exemple le formulaire de souscription en vigueur pour un crédit automobile auprès de Sofinco *via* son site web. Dans cet exemple, des données socio-démographiques et du véhicule à financer sont demandées. Pour un client, elles sont notées $\boldsymbol{x} = (x_j)_1^d$ dans la suite (on reviendra de manière plus formelle sur l'ensemble des notations introduites pour les besoins du cas d'application en fin de chapitre). Ces informations sont de nature continue, c'est-à-dire $x_j \in \mathbb{R}$, ou catégorielle, c'est-à-dire que l'on se donne, à titre d'exemple, un encodage "Métier = technicien" $\rightarrow x_j = 1$, "Métier = ouvrier" $\rightarrow x_j = 2, \ldots$ de telle

FIGURE 1.1 – Schéma des interactions entre les entreprises / marques et la collecte de demandes de crédits.

TABLEAU 1.1 – Exemple simplifié de caractéristiques de demandeurs de crédit : présence de valeurs manquantes ou extrêmes.

Travail	Logement	Durée d'emploi	Enfants	Statut familial	Salaire
Ouvrier qualifié	Propriétaire	20	3	Veuf	30 000
Technicien	En location	Manquant	1	Concubinage	1700
Technicien spécialisé	Accédant	5	0	Divorcé	4000
Cadre	Par l'employeur	8	2	Célibataire	2700
Employé	En location	12	2	Marié	1400
Ouvrier	Par la famille	2	0	Célibataire	1200

sorte que l'on considère que $x_j \in \mathbb{N}_{l_j} = \{1, \ldots, l_j\}$, où l_j représente le nombre de modalités de x_j et sans notion d'ordre.

On considère que ces caractéristiques sont une réalisation du vecteur aléatoire de design $\boldsymbol{X} = (X_j)_1^d \in \mathcal{X}$ sur un espace probabilisé $(\Omega, \mathcal{A}, \mathbb{P})$, que l'on observe sur l'ensemble des n demandeurs de crédit à la consommation pour former, dans la littérature consacrée au *machine learning*, la matrice de design $\mathbf{x} = (x_{i,j})_{1 \leq i \leq n, 1 \leq j \leq d}$.

A ce stade, deux remarques importantes doivent être faites : d'abord, une partie de ces caractéristiques peut être absente. Par ailleurs, elles sont à ce stade déclaratives (des contrôles supplémentaires peuvent avoir lieu en fonction du montant demandé par exemple), et donc associées à un degré de certitude variable, la tentation étant grande, afin de s'assurer de l'attribution du crédit, de déformer la réalité de ses charges, ses revenus, etc. En synthèse, le tableau 1.1 présente un exemple simplifié de matrice de design en *Credit Scoring*. En pratique un tel tableau structuré est directement mis à disposition des statisticiens de CACF à travers le logiciel de traitement statistique SAS.

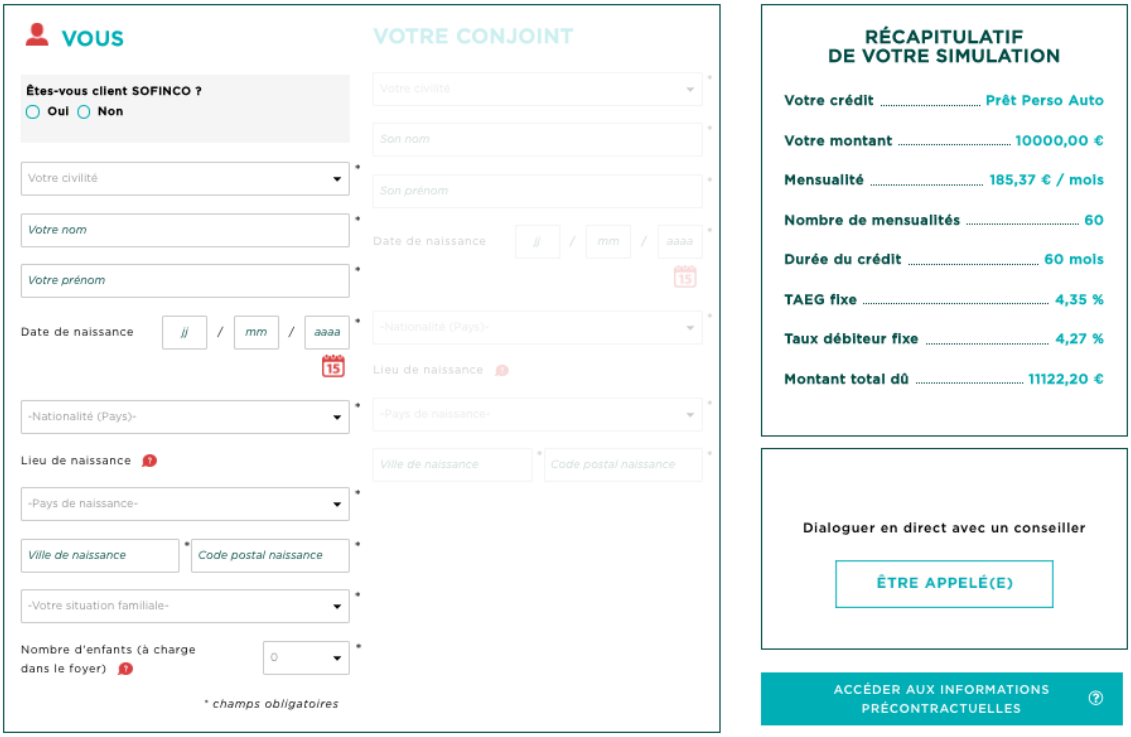


FIGURE 1.2 – Formulaire de souscription d'un crédit automobile Sofinco.

Tableau 1.2 – Exemple de variable continue discrétisée.

Âge du client	18	Manquant	47	25	35	61
Âge discrétisé	18-30 & Manquant	18-30 & Manquant	45-∞	18-30	30-45	45-∞

1.2.2 Préparation des données et segmentation

Le tableau 1.1 fait apparaître deux problèmes bien connus en statistique : la gestion des observations manquantes et celle des valeurs extrêmes (*outliers*).

Concernant les observations manquantes, deux stratégies différentes peuvent être employées. CACF réalise une "segmentation" de sa clientèle, de sorte que, à titre d'exemple, plusieurs modèles statistiques spécialisés à un sous-ensemble de la population totale peuvent être employés, chacun d'eux bénéficiant alors de données complètes. Le processus de choix des "segments", *i.e.* la partition des lignes de $\mathbf{x}$ sur lesquels développer des modèles séparés, est basé soit sur l'histoire de l'entreprise (par exemple, un modèle spécifique aux crédits automobiles a pu être développé au début de la commercialisation de ce produit), soit sur des heuristiques très simples. On détaillera ce problème en chapitre 5.

L'autre pré-traitement répandu dans le milieu du *Credit Scoring* pour faire face aux données manquantes et aux valeurs extrêmes est la discrétisation (pour les variables continues uniquement). Cela consiste à transformer une variable continue dont certaines observations sont manquantes en une variable catégorielle dont chaque modalité correspond à un intervalle de la variable continue d'origine et / ou au fait que l'observation d'origine était manquante. Un exemple de discrétisation de la variable "Âge du client" est visible en figure 1.2 ; ainsi, le fait que l'observation soit manquante est considérée comme une information à part entière et les valeurs extrêmes sont regroupées dans le dernier intervalle. Les mécanismes de données manquantes seront discutés en chapitre 2. Le processus de discrétisation est discuté en détail au chapitre 3.

À présent, on dispose de données rendues complètes sur l'ensemble des demandeurs de crédit et l'on souhaite prédire le niveau de risque présenté par un nouveau demandeur. Il convient donc dans un premier temps de quantifier le risque de chaque échantillon de la matrice de design $\mathbf{x}$.

1.2.3 Définir les "bons" et "mauvais" payeurs

L'institut financier emprunte de l'argent sur les marchés à un taux relativement faible et le redistribue aux demandeurs de crédit qu'il juge profitables, c'est-à-dire susceptible de rembourser cette dette. Il y a donc un système d'acceptation, reposant sur un ensemble de règles automatiques et potentiellement une étude humaine. On considère que le mécanisme qui conduit au financement *in fine* de la demande de crédit est aléatoire, noté Z et prenant les valeurs f (pour les clients dont la demande est financée) et nf (pour les non-financés).

Il convient de noter ici que les différents processus qui conduisent à un non financement du dossier sont très nombreux : interruption / rétractation du demandeur, refus automatique (endettement, score existant, ...) ou refus d'un conseiller clientèle. On y reviendra très brièvement au chapitre 2.

En essence, il est souhaitable de mesurer la profitabilité de chaque crédit, par exemple en actualisant les remboursements et les pertes générés par chaque client à la date de déblocage des fonds, et en déduisant l'ensemble des coûts (financement, traitement, recouvrement, ...). En pratique, peu d'institutions procèdent ainsi malgré quelques travaux récents [9]. Par ailleurs, les caractéristiques du client sont elles-mêmes évolutives : les informations collectées à $t = 0$ au moment de la demande peuvent avoir changé au moment du financement du bien à $t = \text{fin}$ (qui peut intervenir plusieurs mois après pour un véhicule sur commande par exemple), tout

Tableau 1.3 – Exemple d'évolution de dossiers à différents niveaux d'impayés.

Impayés consécutifs	Amélioration	Stabilité	Dégradation
0	0 %	95 %	5 %
1	60 %	10 %	30 %
2	10%	30 %	60 %
3	5%	25 %	70 %
4	5%	15 %	80 %
5	5%	5 %	90 %

comme les moments de vie ultérieurs éventuels comme les divorces, les pertes d'emploi, ... qui ne peuvent être collectées ultérieurement par les organismes financiers, comme schématisé sur la figure 1.3.

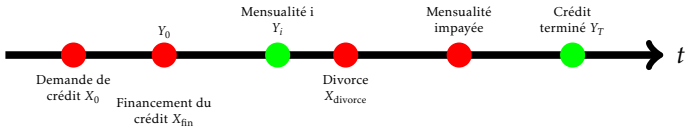


Figure 1.3 – Temporalité du crédit : évolution des covariables.

En conséquence, on sélectionne généralement 12 mois de dossiers de demandes de crédit pour s'affranchir de phénomènes de saisonnalité et on observe le mois suivant la date de financement de chaque dossier si la mensualité a été remboursée. On répète le processus jusqu'à un horizon de 12 à 24 mois selon la disponibilité des données. On dispose alors pour chaque client d'une série temporelle qui indique si le remboursement mensuel a été effectué ou non. On cherche ensuite à se ramener à une seule variable aléatoire cible $Y \in \{0,1\}$ qualifiant un client "bon" par $Y = 1$ ou "mauvais" par $Y = 0$. L'heuristique actuellement utilisée est la suivante :

— Pour un ensemble d'horizons $T \in \{6,12,18,24\}$ mois et d'impayés consécutifs $I \in \{1,\dots,4\}$,
  — Tracer le graphique d'"horizon du risque" : la proportion de clients ayant I impayés consécutifs T mois après leur financement, dont un exemple est donné en figure 1.4 pour $I = 2$.
    On cherche un point d'inflexion sur cette courbe, qui traduirait le fait qu'au-delà d'un certain horizon T, la proportion de dossiers "mauvais" n'évolue plus et l'on considère que tous les "mauvais" clients sont déjà identifiés.
  — Construire le tableau des *Roll Rates*, dont un exemple est donné en tableau 1.3 pour $T = 12$.
    On cherche le nombre d'impayés consécutifs I au-delà duquel la proportion de dossiers se dégradant (et donc fortement susceptibles de générer des pertes) est "importante", généralement au-delà de 50 %.
— Choisir le couple (T, I) qui répond au mieux aux critères ci-dessus et permet d'avoir un nombre significatif de dossiers "mauvais". Il faut garder à l'esprit que plus l'on choisit un horizon T faible et / ou un nombre élevé d'impayés consécutifs I, plus la proportion $\hat{\pi}_0$ (l'estimateur de la moyenne pour $\pi_i = p(Y = i)$) de dossiers "mauvais" par rapport aux dossiers "bons" devient faible. Or, on veut éviter au maximum les nombreux problèmes que génèrent des classes déséquilibrées en classification supervisée [24].

Pour des raisons pratiques et historiques, on choisit généralement $T = 12$ mois et $I = 2$ impayés consécutifs. On considère donc comme "mauvais" $(Y = 0)$ les dossiers financés ayant eu

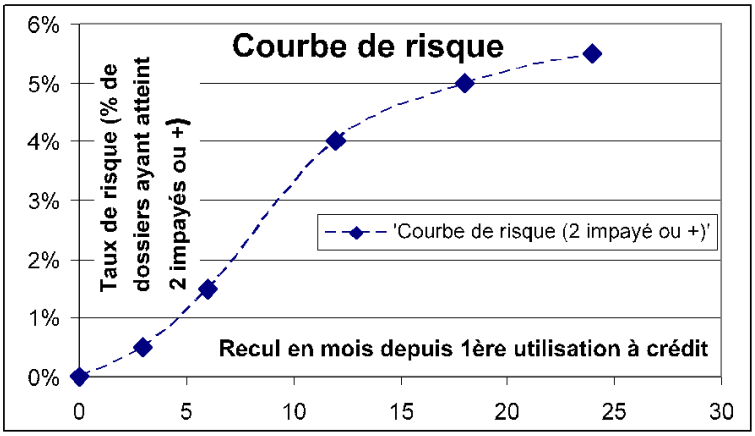


FIGURE 1.4 – Exemple de courbe d'horizon risque : proportion de "mauvais" clients (2 impayés consécutifs) en fonction du nombre de mensualités observées.

TABLEAU 1.4 – Exemple de vecteur **y** de qualification du risque des clients.

y
1
Manquant - Non-financé
0
Manquant - Indéterminé
0
1

au moins 2 mensualités impayées consécutives dans les 12 mois qui ont suivi leur financement, comme "bons" ($Y = 1$) les dossiers n'ayant pas eu d'impayés, comme "indéterminés" les dossiers ayant eu 1 impayé qui sont exclus de la modélisation, et on exclut également tous les dossiers non financés ($Z = \text{nf}$). On a alors le vecteur de réponses **y** dont un exemple est donné en tableau 1.4.

On en conclut que la performance de remboursement n'est observable que pour les clients financés non indéterminés, que l'on va assimiler dans la suite à ceux pour lesquels $Z = \text{f}$, problème qui fait l'objet du chapitre 2. Toujours est-il qu'à présent, on dispose de données $(\mathbf{x}, \mathbf{y})$ complètes grâce auxquelles on souhaite apprendre un score qualifiant la qualité des emprunteurs, et associé à un cutoff produisant une fonction de classification binaire discernant, parmi les futurs demandeurs de crédit, les "bons" des "mauvais" clients.

1.2.4 L'apprentissage d'un score

Malgré l'existence de nombreux modèles statistiques permettant de prédire Y connaissant les caractéristiques $\boldsymbol{X}$ d'un client et que nous discuterons en partie 1.3, la régression logistique est très largement utilisée en *Credit Scoring* [25]. Plusieurs travaux empiriques ont suggéré que du fait du faible nombre de covariables et de classes très mélangées (en particulier, absence de frontière de séparation linéaire entre "bons" et "mauvais" clients), aucun autre modèle de classification supervisée ne produit de résultats significativement supérieurs à la régression logistique sur les données à disposition de leurs auteurs respectifs (se référer par exemple à [13, 1, 3]).

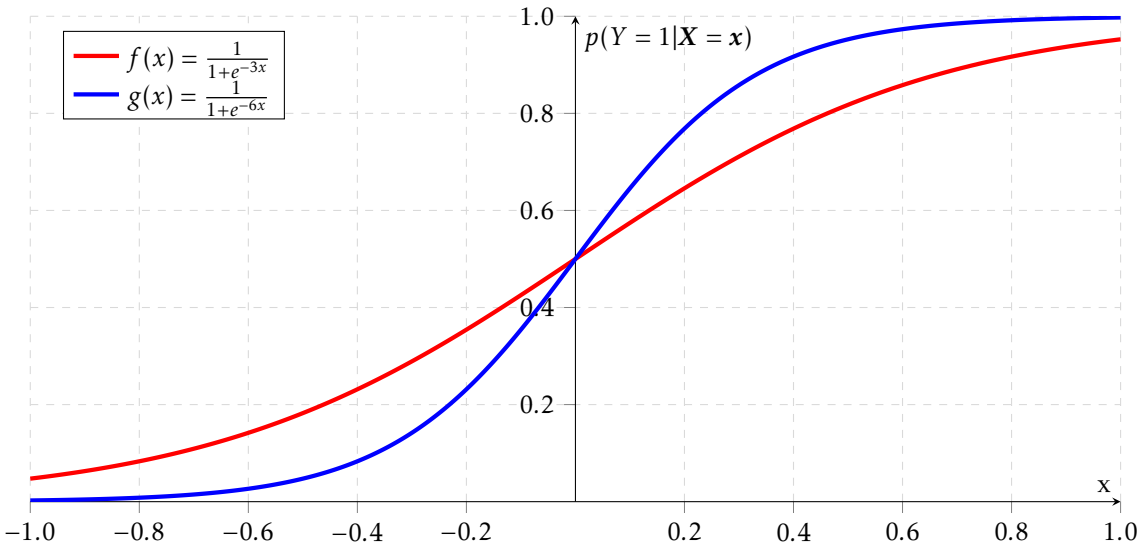


FIGURE 1.5 – Deux exemples de courbes logistiques à une variable explicative sans paramètre de biais.

Le modèle de régression logistique, contrairement à ce que son nom suggère, est un modèle de classification qui impose une structure particulière de loi de probabilité d'une variable aléatoire cible binaire Y conditionnellement à des covariables $\boldsymbol{X} \in \mathcal{X} = \mathbb{R}^d$ donnée par :

$$\text{logit}[p(Y = 1|\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{\theta})] = \ln \frac{p(Y = 1|\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{\theta})}{1 - p(Y = 1|\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{\theta})} = (1, \boldsymbol{x})'\boldsymbol{\theta}. \tag{1.1}$$

Le vecteur $\boldsymbol{\theta} = (\theta_0, \ldots, \theta_d) \in \Theta = \mathbb{R}^{d+1}$ est appelé paramètre. Le coefficient θ_0 définit le biais, c'est-à-dire $\text{logit}[p(Y = 1|\boldsymbol{X} = \boldsymbol{0}, \boldsymbol{\theta})]$. Cette relation est ensuite inversée afin d'obtenir la probabilité d'être "bon" sachant les caractéristiques d'un client et le paramètre $\boldsymbol{\theta}$:

$$p(Y = 1|\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1 + \exp(-(1, \boldsymbol{x})'\boldsymbol{\theta})},$$

et dont des exemples de courbe sont donnés en figure 1.5.

On peut facilement étendre ce modèle aux variables catégorielles $X_j \in \mathbb{N}_{l_j}$ en procédant à un encodage *one-hot*, c'est-à-dire en créant une matrice dite "disjonctive" à i lignes (correspondant toujours à chaque individu $1 \leq i \leq n$) et l_j colonnes binaires (correspondant respectivement à la présence ou l'absence de chaque modalité). À l'indice (i, k) de cette matrice, on trouve la valeur 1 si $x_{i,j} = k$, pour toute modalité $1 \leq k \leq l_j$, 0 sinon. Par exemple pour $l_j = 3$, un encodage possible est :

$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Cette pratique conduit cependant à une sur-paramétrisation : la somme des colonnes pour chaque ligne vaut 1 et la matrice de design, complétée d'une première colonne de 1 pour le terme d'intercept (cf équation (1.1)), n'est alors pas de plein-rang, ce qui pose un problème pour

l'estimation de $\boldsymbol{\theta}$ comme nous le verrons en partie 1.3 ; il faut donc "supprimer" une colonne en considérant une modalité dite de référence (*i.e.* pour laquelle le coefficient est nul). Cet encodage est implicite dans de nombreux logiciels statistiques, si bien que l'on notera les coefficients de régression logistique associés à chaque valeur d'une variable catégorielle X_j en exposant : $\theta_j^1, \dots, \theta_j^{l_j}$. On considérera la dernière modalité comme référence, d'où $\theta_j^{l_j} = 0$.

En fonction du risque que l'institut financier est prêt à prendre, on décide d'un cut, c'est-à-dire d'une probabilité de défaut au-delà de laquelle on refuse la demande de crédit. On désigne traditionnellement par score la fonction $S(\cdot, \boldsymbol{\theta}) : \boldsymbol{x} \mapsto (1, \boldsymbol{x})'\boldsymbol{\theta}$.

La question du support de $\boldsymbol{\theta}$, *i.e.* de ses composantes non nulles, est un problème plus connu sous le nom de "sélection de variables" en statistiques comme en *machine learning*. Un coefficient nul témoigne du fait que la variable associée X_j, conditionnellement aux autres variables que l'on notera $\boldsymbol{x}_{-\{j\}}$ dans la suite, ne permet pas d'expliquer Y. En industrie, il est courant de commencer par sélectionner les variables dont la corrélation avec la variable cible est jugée suffisante. Cette technique univariée ne permet pas de rendre compte de phénomènes multivariées comme la redondance d'information entre covariables ou, à l'inverse, la qualité prédictive d'une variable dont la corrélation avec la cible peut être faible mais qui apporterait une information conditionnellement aux autres variables explicatives. La communauté statistique a donc développé des outils spécifiques à cette question que l'on développera, avec les fondements théoriques des modèles paramétriques comme la régression logistique, en partie 1.3.

1.2.5 La métrique de performance

La métrique utilisée pour comparer la qualité de scores (le score ancien et un nouveau score proposé par exemple) est traditionnellement l'indice de Gini, qui est en fait directement lié à l'aire sous la courbe (AUC) ROC. Cette courbe représente la sensibilité d'un classificateur binaire (*i.e.* la proportion de "bons" clients classés comme "bons") en fonction de son antispécificité (1− la spécificité, *i.e.* la proportion de "mauvais" clients classés comme "bons"). L'AUC s'interprète de plusieurs manières, dont par exemple la probabilité qu'un "bon" (tiré aléatoirement parmi les "bons") ait un score plus élevé qu'un "mauvais" (tiré aléatoirement parmi les "mauvais"). Un exemple de courbe ROC est donné en figure 1.6.

Il faut remarquer à ce stade que ce critère est à la fois différent de celui optimisé par la régression logistique, que nous verrons en détails dans la partie suivante, et de l'objectif industriel de maximiser le profit, soit directement par l'usage de variables de nature financière [9], soit indirectement par le choix d'un cut approprié. Néanmoins, une étude empirique [8] montre que la maximisation de ces différents objectifs est *a priori* relativement équivalente, la qualité prédictive de différents modèles maximisant chacun de ces objectifs étant similaire sur le jeu de données considéré par l'auteur. On suppose cette équivalence dans la suite et sauf indication contraire, les résultats sur données réelles sont donnés en Gini, dont on donnera un intervalle de confiance selon la méthode développée dans [23].

1.2.6 Suivi temporel de la performance du score

Les changements de contexte économique, agissant à la fois sur le vecteur de variables explicatives $\boldsymbol{X} = (X_1, \dots, X_d)$ défini en section 1.2.4 et représentant les caractéristiques du client (l'inflation ou le passage à l'euro impacte l'échelle des salaires par exemple) et la variable cible (la récession entraîne l'augmentation des impayés), la performance du score, selon la métrique précédemment décrite, évolue au cours du temps. Naturellement, cette évolution est la plupart du temps à la baisse puisque la fonction de score apprise s'éloigne de la vérité. Par ailleurs, comme vu en partie 1.2.3, l'apprentissage du score nécessite environ 30 mois de recul, auxquels

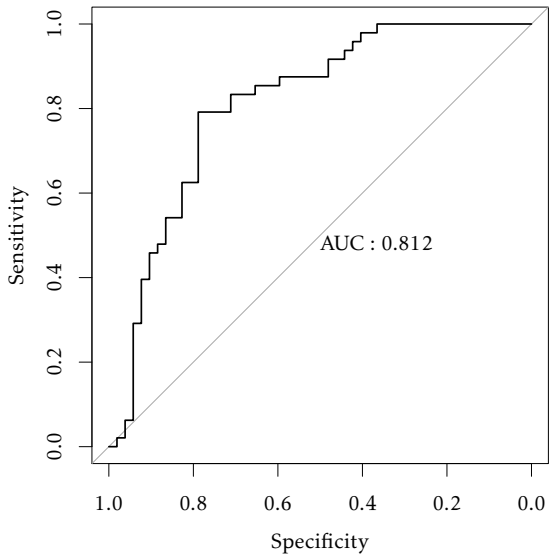


FIGURE 1.6 – Exemple de courbe ROC sur un petit jeu de données simulées et valeur de l'AUC correspondante.

peuvent s'ajouter un délai de mise en production. Dès lors, le statisticien voit émerger deux questions : premièrement, quels sont les "signes" indiquant qu'une refonte, c'est-à-dire la mise en place d'un nouveau modèle prédictif, est nécessaire ? Deuxièmement, est-il possible de construire un modèle prédictif "robuste" à ce problème, communément désigné par *population drift* dans la littérature [13] ?

En pratique, seules la baisse de performance d'un score et / ou son ancienneté importante (5 à 10 ans) conduisent à sa refonte et l'aspect temporel n'est pas pris en compte dans la construction ou l'utilisation des scores.

En conclusion, le *Credit Scoring* repose sur des bases statistiques qui soulèvent de nombreuses questions, dont certaines trouvent dans le milieu industriel une réponse *ad hoc*, très empirique, qu'il convient de formaliser. La partie suivante plonge l'apprentissage du score dans le contexte de l'apprentissage statistique.

1.3 Apprentissage statistique : fondements théoriques du *Credit Scoring*

Après cette mise en situation industrielle qui aura mis en avant les approximations statistiques et autres heuristiques actuellement utilisées dans le milieu bancaire, il convient de formaliser les concepts introduits en partie 1.2. Cette partie s'inspire librement d'introductions de plusieurs ouvrages, dont le bien connu *The Elements of Statistical Learning* [10].

1.3.1 Mécanisme de génération des données

On rappelle brièvement les notations introduites dans la partie précédente : les clients ont d caractéristiques indicées par $j = 1, \dots, d$ dans la suite du manuscrit. Une caractéristique X_j est une variable aléatoire dont on notera la réalisation x_j. L'aggrégation de toutes ces caractéristiques sous la forme d'un vecteur aléatoire est distinguée, comme les autres vecteurs du manuscrit, par une police grasse, en l'occurence $\boldsymbol{X}$. Ce vecteur appartient à l'espace $\mathcal{X}$ qui est un produit de $\mathbb{R}$ (variables continues) ou $\mathbb{N}_{l_j}$ (variables catégorielles à l_j modalités). La variable aléatoire binaire à prédire, le caractère bon / mauvais d'un client, et sa réalisation sont notées respectivement $Y \in \{0, 1\}$ et y. Le même raisonnement s'applique à la variable aléatoire binaire de financement / non financement et sa réalisation, notées respectivement $Z \in \{\mathrm{f}, \mathrm{nf}\}$ et z. Enfin, on dispose d'un n-échantillon $\mathcal{T} = (\mathbf{x}, \mathbf{y}, \mathbf{z})$, où, $\mathbf{x} = (\boldsymbol{x}_i)_1^n$, $\mathbf{y} = (y_i)_1^n$ et $\mathbf{z} = (z_i)_1^n$.

On note p la densité de probabilité (pdf) de $(\boldsymbol{X}, Y)$ et $p(\cdot|\boldsymbol{x})$ la loi de probabilité de Y sachant $\boldsymbol{x}$, qui s'obtient à partir de p et de la relation de Bayes :

$$p(y|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, y)}{p(\boldsymbol{x})},$$

que l'on désignera par "oracle" dans la suite. On aimerait "retrouver" cette loi par calcul, or elle est inconnue (si elle était connue, le problème serait résolu !), et on a uniquement accès au n-échantillon $\mathcal{T}$.

Imaginons un instant que $p(\cdot|\boldsymbol{x})$ soit connu. Une première approche consiste en quelque sorte à exprimer notre connaissance de cette loi en la forçant à appartenir à un modèle (ou à une famille de modèles). Autrement dit, on suppose que $p(\cdot|\boldsymbol{x})$ appartient à un ensemble (très) restreint des lois possibles. Comme énoncé plus haut, dans le cadre du *Credit Scoring*, on s'intéresse au modèle de régression logistique (1.1) noté $p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})$ dans la suite. Dès lors, une formulation simple du problème consiste à se donner une notion de distance entre $p(\cdot|\boldsymbol{x})$ et $p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})$ afin d'estimer le "meilleur" paramètre $\boldsymbol{\theta}^\star$ au sens de cette "distance". Un bon candidat est la divergence de Kullback-Leibler [15] :

$$\mathrm{KL}(p(\cdot|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})) = \sum_{y \in \{0,1\}} p(y|\boldsymbol{x}) \ln\left(\frac{p(y|\boldsymbol{x})}{p_{\boldsymbol{\theta}}(y|\boldsymbol{x})}\right). \tag{1.2}$$

Cette divergence est donnée pour une valeur particulière $\boldsymbol{x}$ de $\boldsymbol{X}$. Or, l'institut financier voudrait que le modèle $p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})$ soit similaire à $p(\cdot|\boldsymbol{x})$ en moyenne pour tous ses clients, ce qui conduit au paramètre

$$\boldsymbol{\theta}^\star = \operatorname*{argmin}_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{X}}[\mathrm{KL}(p(\cdot|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x}))].$$

Comme $\mathrm{KL}(p(\cdot|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})) \geq 0$, on peut voir cette opération comme une projection de la loi $p(\cdot|\boldsymbol{x})$ dans l'espace du modèle (ou de la famille de modèles), illustrée sur la figure 1.7. Cette interprétation géométrique permet d'affirmer que si $\min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{X}}[\mathrm{KL}(p(\cdot|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x}))] = 0$, alors on a pour tout $\boldsymbol{x}$, $p(\cdot|\boldsymbol{x}) = p_{\boldsymbol{\theta}^\star}(\cdot|\boldsymbol{x})$. Dans ce cas, on parlera dans la suite de "vrai modèle" ; dans le cas contraire, de "modèle mal spécifié" (anglicisme de *misspecified model*).

N'ayant accès à $p(\cdot|\boldsymbol{x})$ qu'à travers un échantillon, il nous faut développer un critère empirique à partir du critère théorique (souvent de nature asymptotique) donné ici.

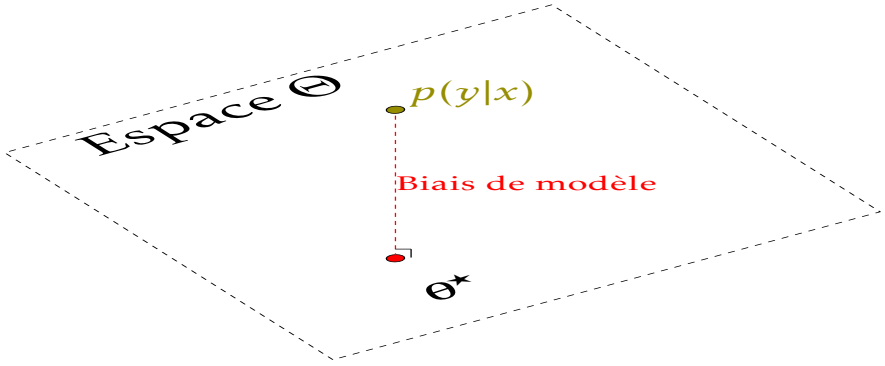


FIGURE 1.7 – Vision géométrique du biais de modèle.

1.3.2 Minimisation du risque empirique et maximum de vraisemblance

On peut réécrire $\mathrm{KL}(p(\cdot|\boldsymbol{x})||p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x}))$ pour faire apparaître une quantité indépendante de $p_{\boldsymbol{\theta}}$:

$$\mathrm{KL}(p(\cdot|\boldsymbol{x})||p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})) = \sum_{y\in\{0,1\}} p(y|\boldsymbol{x})\ln[p(y|\boldsymbol{x})] - \underbrace{\sum_{y\in\{0,1\}} p(y|\boldsymbol{x})\ln[p_{\boldsymbol{\theta}}(y|\boldsymbol{x})]}_{\mathbb{E}_{Y|\boldsymbol{X}=\boldsymbol{x}}[\ln[p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})]]}.$$

On va donc naturellement se concentrer sur la maximisation du second terme pour l'ensemble des clients en moyenne, c'est-à-dire

$$\boldsymbol{\theta}^\star = \operatorname*{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{X}}[\mathbb{E}_{Y|\boldsymbol{X}}[\ln[p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{X})]]] = \operatorname*{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{X},Y)\sim p}[\ln[p_{\boldsymbol{\theta}}(Y|\boldsymbol{X})]].$$

On se place dans le cadre d'un n-échantillon i.i.d. ce qui est toujours le cas en *Credit Scoring* sous réserve que les crédits observés soient issus de clients différents (ce que l'on supposera dans la suite). L'hypothèse d'indépendance nous permet aussi d'approximer l'espérance sur $\mathcal{X}\times\mathcal{Y}$ par l'espérance sur l'échantillon et on obtient le critère $\ell(\theta;\mathbf{x},\mathbf{y})$:

$$\ell(\boldsymbol{\theta};\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} \ln[p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)]. \tag{1.3}$$

Ce critère correspond en fait au maximum de vraisemblance : la probabilité d'observer les données $\mathbf{y}$ sachant les covariables $\mathbf{x}$ et le paramètre $\boldsymbol{\theta}$. L'hypothèse d'indépendance nous permet d'écrire la vraisemblance sous la forme d'un produit :

$$\mathcal{L}(\boldsymbol{\theta};\mathbf{x},\mathbf{y}) = p_{\boldsymbol{\theta}}(y_1,\dots,y_n|\boldsymbol{x}_1,\dots\boldsymbol{x}_n) = \prod_{i=1}^{n} p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i).$$

En passant cette expression au logarithme, fonction strictement croissante, on retrouve bien la formulation de $\ell(\boldsymbol{\theta};\mathbf{x},\mathbf{y})$.

Dans la littérature *machine learning*, où l'on minimise plutôt un risque empirique, sous-entendu de "mauvais classement" au sens d'une fonction de coût à définir, le maximum de vraisemblance est équivalent au minimum de la "log loss". Dans la suite, on préférera la notion

de vraisemblance.

Dans le cas de la régression logistique (1.1), la log-vraisemblance prend la forme suivante :

$$\ell(\boldsymbol{\theta};\mathbf{x},\mathbf{y}) = \underbrace{\sum_{i=1}^{n} y_i(\boldsymbol{\theta}' \times (1,\boldsymbol{x}))}_{\text{fonction affine de } \boldsymbol{\theta}} - \underbrace{\ln(1+\exp(\boldsymbol{\theta}' \times (1,\boldsymbol{x}))}_{\text{log-sum-exp d'une fonction affine de } \boldsymbol{\theta}}.$$

Cette fonction est concave et tout maximum local est donc global.

Passage à la dérivée du critère de log-vraisemblance

Le "réflexe" pour obtenir un maximum local conduit à dériver la fonction de vraisemblance et trouver $\hat{\boldsymbol{\theta}}$ pour lequel cette dérivée est nulle :

$$\frac{\partial \ell}{\partial \theta_j}(\hat{\boldsymbol{\theta}};\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} (y_i - p_{\hat{\boldsymbol{\theta}}}(1|\boldsymbol{x}_i))x_{i,j} = 0.$$

Cependant, contrairement à la régression linéaire où l'on dispose d'une formule explicite pour l'estimateur du maximum de vraisemblance $\hat{\boldsymbol{\theta}}$, il n'existe rien de tel pour la régression logistique puisque cette équation n'est pas linéaire en $\boldsymbol{\theta}$ et l'on doit recourir à des algorithmes itératifs, dont le plus connu est la descente de gradient.

Algorithmes itératifs de descente de gradient

On désigne le gradient de la log-vraisemblance par rapport à $\boldsymbol{\theta}$ par $\nabla_{\boldsymbol{\theta}}\ell = \left(\frac{\partial \ell}{\partial \theta_j}\right)_0^d$. L'algorithme de descente de gradient consiste à mettre à jour à l'étape (s) le paramètre $\boldsymbol{\theta}^{(s)}$ dans la direction qui améliore le critère $\ell(\boldsymbol{\theta};\mathbf{x},\mathbf{y})$:

$$\boldsymbol{\theta}^{(s+1)} = \boldsymbol{\theta}^{(s)} + \epsilon \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}).$$

Une immense littérature est dédiée au choix de ϵ, appelé *learning rate* en *machine learning* et à d'autres astuces destinées à accélérer la convergence éventuelle vers $\hat{\boldsymbol{\theta}}$. Cette littérature s'est particulièrement développée dans le cadre des réseaux de neurones, pour lesquels la méthode de Newton, bien adaptée à la régression logistique et que l'on développera ci-après, n'est pas adaptée.

Méthode de Newton-Raphson

On note la matrice hessienne de ℓ en $\boldsymbol{\theta}$ par $\mathbf{H}_{\boldsymbol{\theta}} = \left(\frac{\partial^2 \ell}{\partial \theta_j \partial \theta_k}\right)_{0 \le j,k \le d}$. Le développement de Taylor, qui revient à considérer que la log-vraisemblance est localement quadratique, donne à l'étape (s) :

$$\ell(\boldsymbol{\theta}^{(s+1)};\mathbf{x},\mathbf{y}) = \ell(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}) + \nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y})'(\boldsymbol{\theta}^{(s+1)} - \boldsymbol{\theta}^{(s)}) + \frac{1}{2}(\boldsymbol{\theta}^{(s+1)} - \boldsymbol{\theta}^{(s)})'\mathbf{H}_{\boldsymbol{\theta}}(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y})(\boldsymbol{\theta}^{(s+1)} - \boldsymbol{\theta}^{(s)}).$$

En dérivant cette expression par rapport à $\boldsymbol{\theta}^{(s+1)}$ et en remarquant que l'on souhaiterait arriver au maximum de ℓ à l'étape $(s+1)$, autrement dit en posant $\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}^{(s+1)};\mathbf{x},\mathbf{y}) = 0$, on obtient :

$$0 = \nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}) + (\boldsymbol{\theta}^{(s+1)} - \boldsymbol{\theta}^{(s)})\mathbf{H}_{\boldsymbol{\theta}^{(s)}}(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}).$$

En réarrangeant cette expression, on obtient la valeur mise à jour du paramètre :

$$\boldsymbol{\theta}^{(s+1)} = \boldsymbol{\theta}^{(s)} - \mathbf{H}_{\boldsymbol{\theta}}(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y})^{-1}\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}),$$

où $\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}) = (\mathbf{1},\mathbf{x})'(\mathbf{y}-\Pi)$ et $\mathbf{H}_{\boldsymbol{\theta}}(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}) = (\mathbf{1},\mathbf{x})\mathbf{W}(\mathbf{1},\mathbf{x})'$ avec $\Pi = (p_{\boldsymbol{\theta}^{(s)}}(1|\boldsymbol{x}_1),\dots,p_{\boldsymbol{\theta}^{(s)}}(1|\boldsymbol{x}_n))$ et $\mathbf{W} = \mathrm{diag}(\Pi\odot(\mathbf{1}-\Pi))$ où $\odot$ désigne le produit d'Hadamard (*i.e.* élément par élément). Plusieurs points importants transparaissent de cette dernière équation. D'abord, si à une étape (s), le point fixe est trouvé, *i.e.* $\boldsymbol{\theta}^{(s)} = \hat{\boldsymbol{\theta}}$, alors $\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}^{(s)};\mathbf{x},\mathbf{y}) = 0$ et on ne bouge plus : $\forall s' \geq s$, $\boldsymbol{\theta}^{(s')} = \boldsymbol{\theta}^{(s)}$. En pratique, cela conduit la majorité des bibliothèques logicielles implémentant la méthode de Newton à laisser à leur utilisateur le soin de calibrer deux paramètres : la précision au-delà de laquelle l'algorithme s'arrête, c'est-à-dire η tel que s'il existe s tel que $\|\boldsymbol{\theta}^{(s+1)} - \boldsymbol{\theta}^{(s)}\|_\infty \leq \eta$, où $\|\boldsymbol{x}\|_\infty = \max_j |x_j|$, alors $\hat{\boldsymbol{\theta}} \approx \boldsymbol{\theta}^{(s+1)}$ et le nombre de pas maximum $s_{\max}$ à effectuer (la condition précédente n'étant potentiellement jamais remplie, l'algorithme pourrait ne pas se terminer). Une revue des principales méthodes d'optimisation utilisables dans le cadre de la régression logistique, suivie de leur étude empirique [17] montre que l'algorithme de Newton et la méthode BFGS [5], de complexité respective $O(nd^2)$ et $O(d^2+nd)$ présentent un bon compromis précision / coût de calcul lorsque comparées à d'autres méthodes de descente de gradient et sous différents scénarios de génération des données. Tous les paramètres de régression logistique de ce manuscrit sont par conséquent estimées par l'algorithme de Newton, car à l'exception des remarques sur la grande dimension données en conclusion, le nombre de covariables d est faible (10-100) relativement à n (10^5-10^6). Enfin, l'algorithme requiert une initialisation $\boldsymbol{\theta}^{(0)}$ qui peut en influencer la vitesse de convergence. Les bibliothèques utilisent généralement $\boldsymbol{\theta}^{(0)} = 0$.

Compromis biais-variance

En conclusion, là où le probabiliste, en figure 1.7 n'avait qu'un problème de biais de modèle, le statisticien qui souhaite estimer ce modèle à partir de données est préoccupé par deux problèmes supplémentaires. Le premier est l'erreur d'estimation, c'est-à-dire la différence entre le meilleur modèle de paramètre $\boldsymbol{\theta}^\star$ et le modèle estimé de paramètre $\hat{\boldsymbol{\theta}}$:

$$\begin{aligned}
&\mathbb{E}_{\mathcal{T}}\mathbb{E}_{\boldsymbol{X}}[p_{\hat{\boldsymbol{\theta}}}(y|\boldsymbol{x}) - p(y|\boldsymbol{x})]^2 \\
=&\mathbb{E}_{\boldsymbol{X}}[\underbrace{[p(y|\boldsymbol{x}) - \mathbb{E}_{\mathcal{T}}[p_{\hat{\boldsymbol{\theta}}}(y|\boldsymbol{x})]]^2}_{\text{biais de modèle}} + \underbrace{\mathbb{E}_{\mathcal{T}}[[p_{\hat{\boldsymbol{\theta}}}(y|\boldsymbol{x}) - \mathbb{E}_{\mathcal{T}}[p_{\hat{\boldsymbol{\theta}}}(y|\boldsymbol{x})]]^2]}_{\text{variance}}] \qquad (1.4)\\
\approx&\mathbb{E}_{\boldsymbol{X}}[\underbrace{[p(y|\boldsymbol{x}) - p_{\boldsymbol{\theta}^\star}(y|\boldsymbol{x})]^2}_{\text{biais de modèle}} + \underbrace{\mathbb{E}_{\mathcal{T}}[[p_{\hat{\boldsymbol{\theta}}}(y|\boldsymbol{x}) - p_{\boldsymbol{\theta}^\star}(y|\boldsymbol{x})]^2]}_{\text{erreur d'estimation}}]. \qquad (1.5)
\end{aligned}$$

Pour la dérivation rigoureuse de ce résultat, se référer à [20] (p. 308–314). Le passage de 1.4 à 1.5 est garanti par le caractère asymptotiquement sans biais de l'estimateur du maximum de vraisemblance, même dans le cas du modèle mal spécifié [27]. Autrement dit, pour n assez grand, on a $\sqrt{n}(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^\star) \sim \mathcal{N}(\mathbf{0}, \mathcal{I}(\boldsymbol{\theta}^\star)^{-1})$, où $\mathcal{I}(\boldsymbol{\theta}) = -\mathbb{E}_{(\boldsymbol{X},Y)}[(\frac{\partial^2 \ln p_{\boldsymbol{\theta}}(y|\boldsymbol{x})}{\partial\theta_j\partial\theta_k}|_{\boldsymbol{\theta}})_{0\leq j,k,\leq d}]$ est la matrice d'information de Fisher. On a alors la consistance asymptotique en probabilité de l'estimateur du maximum de vraisemblance $\hat{\boldsymbol{\theta}}$ vers $\boldsymbol{\theta}^\star$. Le dernier terme de variance a été introduit en quelque

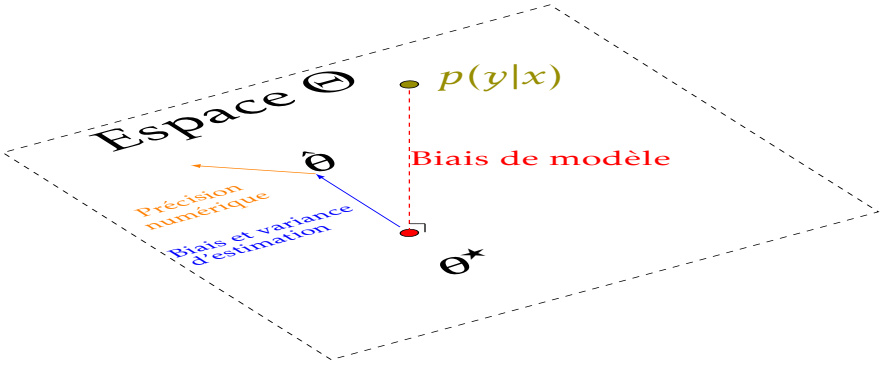


FIGURE 1.8 – Vision géométrique du biais de modèle, biais et variance d'estimation.

sorte par le passage du critère KL asymptotique (1.2) au critère empirique de vraisemblance (1.3). Ce terme est matérialisé en bleu sur la figure 1.8 Le deuxième problème est numérique et généralement négligé : il s'agit de l'erreur de précision développée au paragraphe précédent et matérialisée en orange sur la figure 1.8.

1.3.3 Sélection de modèle en *Credit Scoring*

Dans la partie précédente, on a réduit le problème à la seule estimation de $\boldsymbol{\theta}$, et on a implicitement utilisé l'ensemble des d variables dans $\boldsymbol{X}$. En théorie, se faisant, les variables indépendantes de Y conditionnellement aux autres variables devraient avoir un coefficient θ_j nul. C'est le cas lorsqu'une variable est totalement indépendante de la cible, par exemple la météo du jour de la demande du prêt, ou lorsqu'une variable est redondante avec une autre variable, par exemple les revenus annuels et mensuels qui sont égaux à un facteur multiplicatif près.

En pratique, tous les coefficients de $\boldsymbol{\theta}$ seront différents de 0 du fait des deux phénomènes illustrés sur le graphique 1.8 : l'(im)précision numérique abordée dans la partie précédente et le design $\mathbf{x}$ fixe introduisant un biais et une variance d'estimation. C'est pourquoi il est nécessaire de sélectionner les "bonnes" variables prédictives parmi $X_1,\dots,X_d$ au sens d'un critère que l'on développe ci-après, afin de réduire l'erreur d'estimation.

Par ailleurs et toujours dans le but de trouver un compromis entre biais de modèle et erreur d'estimation, sous certaines conditions que l'on développera dans les chapitres 3 et 4, il peut s'avérer nécessaire d'ajouter des variables par calcul ou combinaison des variables $X_1,\dots,X_d$. On s'intéressera plus précisément aux processus de discrétisation de variables continues, de regroupement de modalités de variables catégorielles et d'introduction d'interactions, c'est-à-dire de produits de variables pré-existantes.

Sélection de variables

Le premier réflexe du statisticien face à un problème de classification est la sélection de variables. A l'extrême, lorsque $d > n$, le problème est mal défini (la matrice hessienne n'est pas inversible) ; dans une moindre mesure, lorsque $n > d$ mais que certaines variables n'ont pas de pouvoir prédictif conditionnellement à celles déjà dans le modèle, c'est-à-dire par exemple $p(y|\boldsymbol{x}) = p(y|x_2,\dots,x_d)$, alors le coefficient $\hat{\theta}_1$ ajoute une dimension "inutile" à l'espace $\boldsymbol{\Theta}$ (on parle de la capacité d'un modèle en *machine learning*) qui augmente la variance du modèle $p_{\boldsymbol{\theta}}$ (on parle

d'*overfitting* en *machine learning*) en essayant en quelque sorte de prédire le bruit, c'est-à-dire les résidus du modèle. Dans les chapitres suivants, on utilisera abusivement la notation p pour toute pdf lorsque les variables dont elle dépend sont explicites.

Dans le cas particulier du *Credit Scoring*, une thèse CIFRE récente a même été consacrée au sujet de la sélection de variables [26] et recommande l'utilisation de la procédure LASSO, de la "famille" des méthodes de pénalisation : une contrainte est ajoutée à la vraisemblance pour l'optimisation des paramètres. Le critère devient :

$$\begin{aligned}\hat{\boldsymbol{\theta}}^{\text{Lasso}} &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \ell(\boldsymbol{\theta};\mathbf{x},\mathbf{y}) \text{ avec } \sum_{j=1}^{d} |\theta_j| \leq t \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \ell(\boldsymbol{\theta};\mathbf{x},\mathbf{y}) + \lambda \sum_{j=1}^{d} |\theta_j|\end{aligned}$$

où t et λ sont mutuellement dépendants et règlent la sévérité de la régularisation. De manière générale, la régularisation présente plusieurs avantages, et la motivation première est le contrôle du compromis biais-variance. Néanmoins, par l'utilisation d'une pénalisation de type L^1 comme le LASSO, un effet de bord désirable est la sélection de variables, c'est-à-dire la capacité à "forcer" des coefficients estimés exactement à 0. Plusieurs variantes ou raffinements du LASSO existent aujourd'hui et possèdent des propriétés asymptotiques différentes ou meilleures : nous y reviendrons brièvement en Conclusion and prospects.

Critère de sélection de modèle

Une approche de résolution indirecte du problème de sélection de variables est le choix de modèle : considérons M modèles $\Theta^{(1)},\ldots,\Theta^{(M)}$ de régression logistique différents, c'est-à-dire pour lesquels les variables incluses ne sont pas les mêmes. On peut d'ailleurs voir le problème de sélection de variables comme un choix entre tous les 2^d modèles possibles. Dans ce cadre, de nombreux critères de choix de modèle, voire d'aggrégation de modèles, c'est-à-dire de sélection de tout ou partie de ces modèles en pondérant leur contribution globale, ont été proposés. La justification de ces critères sort largement du cadre de ce manuscrit ; aussi nous nous limiterons, dans le cadre de la sélection de modèle, au critère BIC (proposé dans [21]). Outre sa consistance asymptotique, autrement dit la capacité de sélectionner, sous certaines conditions sur la famille notamment, le "quasi-vrai" modèle (le modèle de plus faible divergence KL et de complexité ν minimale - cf ci-après) avec une probabilité tendant vers 1 lorsque la taille d'échantillon n augmente, ce critère possède une propriété au coeur du chapitre 4 qui le lie à la probabilité *a posteriori* d'un modèle conditionnellement aux données. On donne les grandes lignes de la dérivation de ce résultat au chapitre 4 et on reviendra plusieurs fois sur la question du comptage du nombre de paramètres estimés ν ci-dessous qui constitue le terme de pénalisation du critère BIC. Le lecteur désireux d'en apprendre plus sur les propriétés et les fondements théoriques du critère BIC peuvent consulter [16] et en particulier une de ses références [4].

Le critère BIC s'écrit de la manière suivante et doit être minimisé :

$$\text{BIC}(\hat{\boldsymbol{\theta}}) = -2\ell(\hat{\boldsymbol{\theta}};\mathbf{x},\mathbf{y}) + \nu \ln(n), \tag{1.6}$$

où $\hat{\boldsymbol{\theta}}$ est l'estimateur du maximum de vraisemblance et $\nu = \dim(\Theta)$.

1.3.4 Autres modèles prédictifs

L'objectif de cette partie est de donner un éclairage à d'autres familles de modèles prédictifs qui pourraient être utilisés en lieu et place de la régression logistique traditionnellement utilisée en *Credit Scoring* pour les nombreuses raisons pratiques et statistiques précédemment évoquées. Ces modèles sont utilisés dans le chapitre 2 de ce manuscrit en comparaison de la régression logistique, ainsi que, pour les réseaux de neurones, dans le chapitre 3.

Arbres de décision

Principe Toutes les observations entrent au sommet de l'arbre qui dispose d'un seul noeud. Ce noeud contient une règle de classement parmi les noeuds fils de type `si ... alors ...`. Chacun de ces noeuds fils dispose alors d'un sous-ensemble des observations de départ, et la procédure se répète récursivement jusqu'aux feuilles de l'arbre, c'est-à-dire les noeuds dépourvus de fils, dont les observations sont affectées, dans le cadre de l'apprentissage supervisée, à une classe bon / mauvais payeur. Cette structure est utilisée en *Credit Scoring* dans le cadre de la segmentation (section 1.2.2), pratique que l'on revisite au chapitre 5 et où un exemple d'arbre est visible en figure 5.1.

Algorithmes Ainsi posé, l'arbre de décision semble à la fois simple dans sa formulation, et complexe dans la mise en oeuvre de son apprentissage : comment choisir les règles de chaque noeud, le nombre de noeuds fils à chaque noeud, le critère d'arrêt, etc. En pratique, de nombreux algorithmes ont été proposés. Dans les expériences du chapitre 2, on utilise l'algorithme C4.5 [19] qui repose sur la divergence de Kullback-Leibler pour choisir une variable x_j à chaque noeud et un ensemble C_j tel que les observations vérifiant $x_j \in C_j$ (resp. $x_j \notin C_j$) soient orientées vers le noeud fils gauche (resp. droit), où $C_j =]-\infty; c_j]$, $c_j \in \mathbb{R}$ pour les variables continues et $C_j \subset \mathbb{N}_{l_j}$ pour les variables catégorielles. L'algorithme s'arrête lorsque les feuilles ne contiennent qu'une seule classe, et des techniques d'"élagage" permettent ensuite de réduire la complexité de l'arbre résultant pour garantir un bon compromis biais-variance.

Faiblesses Les arbres de décision souffrent souvent de large variance [11]. C'est pourquoi, les Forêts Aléatoires [2] et / ou algorithmes dits de "Boosting" [28] sont plébiscités : plusieurs arbres de décision sont appris, sur des sous-échantillons et / ou en pondérant les observations d'apprentissage, dont les décisions sont ensuite combinées. Pour les données de *Credit Scoring*, il a été constaté en interne à CACF que ces modèles permettent d'obtenir de bonnes performances, en perdant cependant l'interprétation aisée des arbres de décision ou de la régression logistique.

Réseaux de neurones

Principe Chaque variable d'entrée, c'est-à-dire une covariable $\boldsymbol{x}_j$, est vue comme un neurone, tout comme la variable de sortie, c'est-à-dire la variable dépendante à prédire y. Les neurones intermédiaires, formant la (les) couche(s) cachée(s) réalisent un calcul à partir de leur(s) neurone(s) parent(s) (phase de propagation dite *feedforward*) consistant typiquement en une addition et une transformation non-linéaire (comme la fonction sigmoïde - l'application réciproque du logit - qui sert en régression logistique). Les résultats prédits $\hat{\mathbf{y}}$ sont comparés aux exemples d'apprentissage $\mathbf{y}$ et l'erreur est rétropropagée (phase dite *backpropagation*) : comme en régression logistique, les couches cachées disposent de coefficients $\boldsymbol{\theta}$ qui sont ajustés par descente de gradient. La comparaison biologique est cependant bien plus limitée que ce que leur nom laisse supposer : les neurones représentent simplement un état résultant d'un calcul, et les synapses sont les arêtes du graphe de calcul (qui déterminent le(s) neurone(s) parent(s) / enfant(s) de chaque neurone).

Une architecture particulière de ce modèle est utilisée dans le chapitre 3 comme graph de calcul pour résoudre le problème de discrétisation évoqué en section 1.2.2.

Limites et développements récents Les inconvénients de ce type de modèle vont de paire avec leur avantage de flexibilité : le grand nombre de paramètres et hyperparamètres rendent leur interprétation et leur apprentissage compliqués. L'interprétation aisée du modèle, *e.g.* de l'effet de chaque variable (et de la significativité de cet effet), de la forme de la frontière de décision, est primordial dans de nombreux contextes applicatifs comme le *Credit Scoring* : le management, dont l'exposition aux statistiques est faible ou nulle, doit pouvoir comprendre le processus de décision de même que le client pouvant se voir refuser l'accès au crédit. C'est pourquoi les régulateurs bancaires sont attentifs à ce que les décisions soient explicables au client, ce qui est généralement garanti par l'usage massif de la régression logistique, modèle développé dans les parties précédentes, mais qui est moins immédiat dans le cas présent des réseaux de neurones du fait de l'introduction de nombreuses non-linéarités et combinaisons de plusieurs variables (toutes les variables dans le cas des réseaux dits densément connectés). Par ailleurs, ces modèles reposent sur des techniques de descente de gradient, brièvement évoqués en partie 1.3.2, qui demandent des connaissances *ad hoc* et / ou spécifiques au domaine d'application pour la calibration des nombreux hyperparamètres entre autres liés au pas de gradient.

Le lecteur désireux d'approfondir sa connaissance sur ce type de modèle, devenu une discipline de recherche à part entière, peut se référer à l'ouvrage *Deep Learning* [12].

Ce chapitre a permis de présenter les méthodes statistiques industrielles du *Credit Scoring*, qui soulèvent des questions théoriques dont certaines sont traitées dans ce manuscrit. Le chapitre suivant rend compte de travaux menés en première année de thèse ; on s'intéresse au problème de la réintégration des refusés, abordé en partie 1.2.3, qui est un bon exemple de la nécessaire formalisation mathématique de pratiques historiques du domaine et dont les hypothèses sous-jacentes sont mal maîtrisées dans l'industrie.

Références du chapitre 1

[1] Bart Baesens et al. « Benchmarking state-of-the-art classification algorithms for credit scoring ». In : *Journal of the operational research society* 54.6 (2003), p. 627-635.

[2] Leo Breiman. « Random forests ». In : *Machine learning* 45.1 (2001), p. 5-32.

[3] Iain Brown et Christophe Mues. « An experimental comparison of classification algorithms for imbalanced credit scoring data sets ». In : *Expert Systems with Applications* 39.3 (2012), p. 3446-3453.

[4] Kenneth P Burnham et David R Anderson. *Model selection and multimodel inference : a practical information-theoretic approach*. Springer Science & Business Media, 2003.

[5] Richard H Byrd et al. « A limited memory algorithm for bound constrained optimization ». In : *SIAM Journal on Scientific Computing* 16.5 (1995), p. 1190-1208.

[6] Elsa Dicharry. *Maison de la literie lance la location avec option d'achat*. Oct. 2017. url : https://www.lesechos.fr/26/10/2017/lesechos.fr/030786701376_maison-de-la-literie-lance-la-location-avec-option-d-achat.htm.

[7] Hélène Ducourant. « Le crédit revolving, un succès populaire ». In : *Sociétés contemporaines* 4 (2009), p. 41-65.

[8] Steven Finlay. « Are we modelling the right thing? The impact of incorrect problem specification in credit scoring ». In : *Expert Systems with Applications* 36.5 (2009), p. 9065-9071.

[9] Steven Finlay. « Credit scoring for profitability objectives ». In : *European Journal of Operational Research* 202.2 (2010), p. 528-537.

[10] Jerome Friedman, Trevor Hastie et Robert Tibshirani. *The Elements of Statistical Learning*. T. 1. 10. Springer series in statistics New York, NY, USA : 2001.

[11] Pierre Geurts et Louis Wehenkel. « Investigation and reduction of discretization variance in decision tree induction ». In : *European Conference on Machine Learning*. Springer. 2000, p. 162-170.

[12] Ian Goodfellow et al. *Deep Learning*. T. 1. MIT press Cambridge, 2016.

[13] David J Hand et William E Henley. « Statistical classification methods in consumer credit scoring : a review ». In : *Journal of the Royal Statistical Society : Series A (Statistics in Society)* 160.3 (1997), p. 523-541.

[14] Dilruba Karim et al. « Off-balance sheet exposures and banking crises in OECD countries ». In : *Journal of Financial Stability* 9.4 (2013), p. 673-681.

[15] Solomon Kullback et Richard A Leibler. « On information and sufficiency ». In : *The annals of mathematical statistics* 22.1 (1951), p. 79-86.

[16] Emilie Lebarbier et Tristan Mary-Huard. « Le critère BIC : fondements théoriques et interprétation ». In : RR-5315 (2004), p. 17. url : https://hal.inria.fr/inria-00070685.

[17] Thomas P Minka. « A comparison of numerical optimizers for logistic regression ». In : *Unpublished draft* (2003), p. 1-18.

[18] Jean-Philippe Peden. *Vente de voitures : la part des formules de location a décollé en 2017*. Jan. 2018. url : https://news.autoplus.fr/Location-LLD-LOA-Vente-Marques-premium-1523494.html.

[19] J Ross Quinlan. *C4. 5 : programs for machine learning*. Elsevier, 2014.

[20] Hinrich Schütze, Christopher D Manning et Prabhakar Raghavan. *Introduction to information retrieval*. T. 39. Cambridge University Press, 2008. url : https://nlp.stanford.edu/IR-book/pdf/irbookprint.pdf.

[21] Gideon Schwarz. « Estimating the Dimension of a Model ». In : *The Annals of Statistics* 6.2 (1978), p. 461-464. issn : 00905364. url : http://www.jstor.org/stable/2958889.

[22] Statista. *Credit cards per household by country in 2016*. 2016. url : https://www.statista.com/statistics/650858/credit-cards-per-household-by-country/.

[23] Xu Sun et Weichao Xu. « Fast Implementation of DeLong's Algorithm for Comparing the Areas Under Correlated Receiver Oerating Characteristic Curves ». In : *IEEE Signal Processing Letters* 21.11 (2014), p. 1389-1393.

[24] Yanmin Sun, Andrew KC Wong et Mohamed S Kamel. « Classification of imbalanced data : A review ». In : *International Journal of Pattern Recognition and Artificial Intelligence* 23.04 (2009), p. 687-719.

[25] Lyn C Thomas. « A survey of credit and behavioural scoring : forecasting financial risk of lending to consumers ». In : *International journal of forecasting* 16.2 (2000), p. 149-172.

[26] Clément Vital. « Scoring pour le risque de crédit : variable réponse polytomique, sélection de variables, réduction de la dimension, applications ». 2016REN1S111. Thèse de doct. 2016. url : http://www.theses.fr/2016REN1S111.

[27] Halbert White. « Maximum likelihood estimation of misspecified models ». In : *Econometrica : Journal of the Econometric Society* (1982), p. 1-25.

[28] Zhi-Hua Zhou. *Ensemble methods : foundations and algorithms*. Chapman et Hall/CRC, 2012.

Chapter 2

Reject Inference: a rational review

Sounds good, doesn't work.

Donald J. Trump

Sommaire

The granting process of all credit institutions is based on the probability that the applicant will refund his loan given his characteristics. This probability also called score is learnt based on a dataset in which rejected applicants are *de facto* excluded (see Section 1.2.3). This implies

that the population on which the score is used will be different from the learning population. Thus, this biased learning can have consequences on the scorecard's relevance. Many methods dubbed "reject inference" have been developed in order to try to exploit the data available from the rejected applicants to build the score. However most of these methods are considered from an empirical point of view, and there is some lack of formalization of the assumptions that are really made, and of the theoretical properties that can be expected. We propose a formalisation of these usually hidden assumptions for some of the most common reject inference methods, and we discus the improvement that can be expected. These conclusions are illustrated on simulated data and on real data from CACF.

2.1 Introduction

In consumer loans, the acceptance process was described in Chapter 1 and can be formalized as follows. For a new applicant's profile and credit's characteristics, the lender aims at estimating the repayment probability. To this end, the *credit modeler* fits a predictive model, often a logistic regression, between already financed clients' characteristics $\boldsymbol{x} = (x_1, \ldots, x_d)$ and their repayment status, a binary variable $y \in \{0, 1\}$ (where 1 corresponds to good clients and 0 to bad clients). The model is then applied to the new applicant and yields an estimate of its repayment probability, called score after an increasing transformation (see Section 1.2.4). Over some cut-off value of the score, the applicant is accepter, except if further expert rules come into play as can be seen from Figure 2.1.

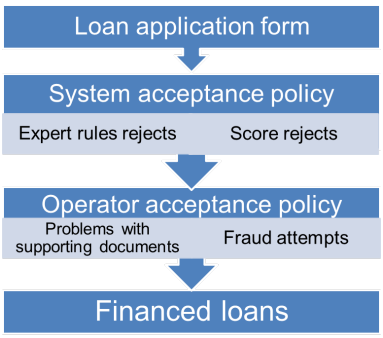


Figure 2.1 – Simplified Acceptance mechanism in Crédit Agricole Consumer Finance

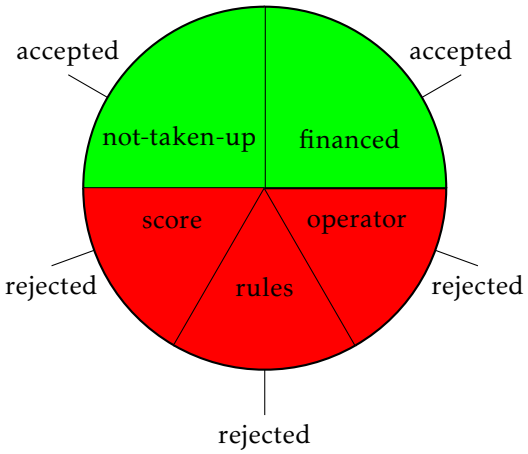


Figure 2.2 – Simplified Acceptance status in Crédit Agricole Consumer Finance - scale relations not respected

The through-the-door population (all applicants) can be classified into two categories thanks to a binary variable z taking values in $\{\mathrm{f}, \mathrm{nf}\}$ where f stands for financed applicants (in green on Figure 2.2) and nf for not financed ones (in red on Figure 2.2). As the repayment variable y is unobserved for not financed applicants, credit scorecards are only constructed on financed clients' data but then applied to the whole through-the-door population. The relevance of this process is a natural question which is dealt in the field of reject inference. The idea is to use the characteristics of not financed clients in the scorecard building process to avoid a population bias, and thus to improve the prediction on the whole through-the-door population. Such methods

have been described in [15, 7, 1, 12], and have also notably been investigated in [6] who first saw reject inference as a missing data problem. In [8], the misspecified model case on real data is studied specifically and is also developed here.

In fact, it can be considered as a part of the semi-supervised learning setting, which consists in learning from both labelled and unlabelled data. However, in the semi-supervised setting [3] it is generally assumed that labelled data and unlabelled data come from the same distribution, which is rarely the case in *Credit Scoring*. Moreover, the main use case of semi-supervised learning is when the number of unlabelled data is far larger than the number of labelled data, which is not the case in *Credit Scoring* since the number of rejected clients and accepted clients is often balanced and depends heavily on the financial institution, the portfolio considered, *etc.*

The purpose of the present chapter is twofold: a clarification of which mathematical hypotheses, if any, underlie those reject inference methods and a clear conclusion on their relevance. In Section 2.2, we present a criterion to assess a method's performance and discuss missingness mechanisms that characterize the relationship of z with respect to $\boldsymbol{x}$ and y. In Section 2.3, we go through some of the most common reject inference methods and exhibit their mathematical properties. To confirm our theoretical findings, we test each method on real data from Crédit Agricole Consumer Finance in Section 2.4. Finally, some guidelines are given to practitioners in Section 2.5.

2.2 *Credit Scoring* modelling

2.2.1 Data

The decision process of financial institutions to accept a credit application is usually embedded in the probabilistic framework. The latter offers rigorous tools for taking into account both the variability of applicants and the uncertainty on their ability to pay back the loan. In this context, the important term is $p(y|\boldsymbol{x})$, designing the probability that a new applicant (described by his characteristics $\boldsymbol{x}$) will pay back his loan ($y = 1$) or not ($y = 0$). Estimation of $p(y|\boldsymbol{x})$ is thus an essential task of any *Credit Scoring* process.

To perform estimation, a specific $n+n'$-sample $\mathcal{T}$ is available, decomposed into two disjoint and meaningful subsets, denoted by $\mathcal{T}_{\mathrm{f}}$ and $\mathcal{T}_{\mathrm{nf}}$ ($\mathcal{T} = \mathcal{T}_{\mathrm{f}} \cup \mathcal{T}_{\mathrm{nf}}$, $\mathcal{T}_{\mathrm{f}} \cap \mathcal{T}_{\mathrm{nf}} = \emptyset$). The first subset ($\mathcal{T}_{\mathrm{f}}$) corresponds to n applicants with features $\boldsymbol{x}_i$ who have been financed ($z_i = \mathrm{f}$) and, consequently, for who the repayment status y_i is known, with their respective matrix notation $\mathbf{x}_{\mathrm{f}}$, $\mathbf{z}_{\mathrm{f}}$ and $\mathbf{y}_{\mathrm{f}}$. Thus, $\mathcal{T}_{\mathrm{f}} = (\boldsymbol{x}_i, y_i, z_i)_{i\in \mathrm{F}} = (\mathbf{x}_{\mathrm{f}}, \mathbf{y}_{\mathrm{f}}, \mathbf{z}_{\mathrm{f}})$ where $\mathrm{F} = \{i : z_i = \mathrm{f}\}$ denotes the corresponding subset of indexes. The second subset ($\mathcal{T}_{\mathrm{nf}}$) corresponds to n' applicants with features $\boldsymbol{x}_i$ who have *not* been financed ($z_i = \mathrm{nf}$) and, consequently, for who the repayment status y_i is *unknown*, with their respective matrix notation $\mathbf{x}_{\mathrm{nf}}$ and $\mathbf{z}_{\mathrm{nf}}$. Thus, $\mathcal{T}_{\mathrm{nf}} = (\boldsymbol{x}_i, z_i)_{i\in \mathrm{NF}} = (\mathbf{x}_{\mathrm{nf}}, \mathbf{z}_{\mathrm{nf}})$ where $\mathrm{NF} = \{i : z_i = \mathrm{nf}\}$ denotes the corresponding subset of indexes. We notice that y_i values are excluded from the observed sample $\mathcal{T}_{\mathrm{nf}}$, since they are missing. These data can be represented schematically as:

$$\mathcal{T} = \begin{array}{c} \mathcal{T}_{\mathrm{f}} = \\ \cup \\ \mathcal{T}_{\mathrm{nf}} = \end{array} \begin{array}{c} \\ \\ \end{array} \begin{pmatrix} \mathbf{x}_{\mathrm{f}} \\ \mathbf{x}_{\mathrm{nf}} \end{pmatrix} \begin{pmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \vdots & \vdots \\ x_{n,1} & \cdots & x_{n,d} \\ x_{n+1,1} & \cdots & x_{n+1,d} \\ \vdots & \vdots & \vdots \\ x_{n+n',1} & \cdots & x_{n+n',d} \end{pmatrix}, \quad \begin{pmatrix} \mathbf{y}_{\mathrm{f}} \\ \mathbf{y}_{\mathrm{nf}} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \\ \mathrm{NA} \\ \vdots \\ \mathrm{NA} \end{pmatrix}, \quad \begin{pmatrix} \mathbf{z}_{\mathrm{f}} \\ \mathbf{z}_{\mathrm{nf}} \end{pmatrix} \begin{pmatrix} \mathrm{f} \\ \vdots \\ \mathrm{f} \\ \mathrm{nf} \\ \vdots \\ \mathrm{nf} \end{pmatrix}.$$

2.2.2 General parametric model

Estimation of $p(y|\boldsymbol{x})$ has to rely on modelling since the true probability distribution is unknown. Firstly, it is both convenient and realistic to assume that triplets in $\mathcal{T}_c = (\boldsymbol{x}_i, y_i, z_i)_{1\leq i\leq n+n'}$ are all independent and identically distributed (i.i.d.), including the unknown values of y_i when $i \in \text{NF}$. Secondly, it is usual and convenient to assume that the unknown distribution $p(y|\boldsymbol{x})$ belongs to a given parametric family $\{p_{\boldsymbol{\theta}}(y|\boldsymbol{x})\}_{\boldsymbol{\theta}\in\Theta}$, where Θ is the parameter space as was discussed in Chapter 1. For instance, logistic regression is often considered in practice, even if we will be more general in this section. However, logistic regression will be important for other sections since some standard reject inference methods are specific to this family (Section 2.3) and numerical experiments (Section 2.4, Appendices A.1.6, and A.1.7) will implement them.

As in any missing data situation (here z indicates if y is observed or not), the relative modelling process, namely $p(z|\boldsymbol{x}, y)$, has also to be clarified. For convenience, we can also consider a parametric family $\{p_{\boldsymbol{\phi}}(z|\boldsymbol{x}, y)\}_{\boldsymbol{\phi}\in\Phi}$, where $\boldsymbol{\phi}$ denotes the parameter and Φ the associated parameter space of the financing mechanism. Note we consider here the most general missing data situation, namely a missing not at random (MNAR) mechanism [10]. It means that z can be stochastically dependent on some missing data y, namely that $p(z|\boldsymbol{x}, y) \neq p(z|\boldsymbol{x})$. We will discuss this fact in Section 2.2.4.

Finally, combining both previous distributions $p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$ and $p_{\boldsymbol{\phi}}(z|\boldsymbol{x}, y)$ leads to express the joint distribution of (y, z) conditionally to $\boldsymbol{x}$ as:

$$p_\gamma(y, z|\boldsymbol{x}) = p_{\boldsymbol{\phi}(\gamma)}(z|y, \boldsymbol{x}) p_{\boldsymbol{\theta}(\gamma)}(y|\boldsymbol{x}) \tag{2.1}$$

where $\{p_\gamma(y, z|\boldsymbol{x})\}_{\gamma\in\Gamma}$ denotes a distribution family indexed by a parameter γ evolving in a space Γ. Here it is clearly expressed that both parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ can depend on γ, even if in the following we will note shortly $\boldsymbol{\phi} = \boldsymbol{\phi}(\gamma)$ and $\boldsymbol{\theta} = \boldsymbol{\theta}(\gamma)$. In this very general missing data situation, the missing process is said to be *non-ignorable*, meaning that parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ can be functionally dependent (thus $\gamma \neq (\boldsymbol{\phi}, \boldsymbol{\theta})$). We also discuss this fact in Section 2.2.4.

2.2.3 Maximum likelihood estimation

Mixing previous model and data, the maximum likelihood principle can be invoked for estimating the whole parameter γ, thus yielding as a by-product an estimate of the parameter $\boldsymbol{\theta}$. Indeed, $\boldsymbol{\theta}$ is of particular interest, the goal of the financial institutions being solely to obtain an estimate of $p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$. The observed log-likelihood can be written as:

$$\ell(\gamma; \mathcal{T}) = \sum_{i\in\text{F}} \ln p_\gamma(y_i, \text{f}|\boldsymbol{x}_i) + \sum_{i'\in\text{NF}} \ln\left[\sum_{y\in\{0,1\}} p_\gamma(y, \text{nf}|\boldsymbol{x}_{i'})\right]. \tag{2.2}$$

Within this missing data paradigm, the EM algorithm (see [4]) can be used: it aims at maximizing the expectation of the complete likelihood $\ell_c(\gamma; \mathcal{T}_c)$ (defined hereafter) where $\mathcal{T}_c = \mathcal{T} \cup \mathbf{y}_{\text{nf}}$ over the missing labels. Starting from an initial value $\gamma^{(0)}$, iteration (s) of the algorithm is decomposed into the following two classical steps:

E-step compute the conditional probabilities of missing y_i values:

$$t_{iy}^{(s)} = p_{\boldsymbol{\theta}(\gamma^{(s-1)})}(y|\boldsymbol{x}_i, \text{nf}) = \frac{p_{\gamma^{(s-1)}}(y, \text{nf}|\boldsymbol{x}_i)}{\sum_{y'=0}^{1} p_{\gamma^{(s-1)}}(y', \text{nf}|\boldsymbol{x}_i)}; \tag{2.3}$$

M-step maximize the conditional expectation of the complete log-likelihood:

$$\ell_c(\gamma;\mathcal{T}_c) = \sum_{i=1}^{n+n'} \ln p_\gamma(y_i, z_i|\boldsymbol{x}_i) = \sum_{i=1}^{n} \ln p_\gamma(y_i, \mathrm{f}|\boldsymbol{x}_i) + \sum_{i=n+1}^{n+n'} \ln p_\gamma(y_i, \mathrm{nf}|\boldsymbol{x}_i); \tag{2.4}$$

leading to:

$$\begin{aligned}\gamma^{(s)} &= \operatorname*{argmax}_{\gamma\in\Gamma} \mathbb{E}_{\mathbf{y}_{\mathrm{nf}}}[\ell_c(\gamma;\mathcal{T}_c)|\mathcal{T}, \gamma^{(s-1)}] \\ &= \operatorname*{argmax}_{\gamma\in\Gamma} \sum_{i\in\mathrm{F}} \ln p_\gamma(y_i, \mathrm{f}|\boldsymbol{x}_i) + \sum_{i'\in\mathrm{NF}} \sum_{y=0}^{1} t_{i'y}^{(s)} \ln p_\gamma(y, \mathrm{nf}|\boldsymbol{x}_{i'}).\end{aligned}$$

Usually, stopping rules rely either on a predefined number of iterations, or on a predefined stability criterion of the observed log-likelihood.

2.2.4 Some current restrictive missingness mechanisms

The latter parametric family is very general since it considers both that the missingness mechanism is missing not at random (MNAR) and non-ignorable. But in practice, it is common to consider ignorable models for the sake of simplicity, meaning that $\gamma = (\boldsymbol{\phi}, \boldsymbol{\theta})$. There exists also some restrictions to the MNAR mechanism.

The first restriction to MNAR is the missing completely at random (MCAR) setting, meaning that $p(z|\boldsymbol{x}, y) = p(z)$. In that case, applicants should be accepted or rejected without taking into account their descriptors $\boldsymbol{x}$. Such a process is not realistic at all for representing the actual process followed by financial institutions. Consequently it is always discarded in *Credit Scoring*.

The second restriction to MNAR is the missing at random (MAR) setting, meaning that $p(z|\boldsymbol{x}, y) = p(z|\boldsymbol{x})$. The missing at random (MAR) missingness mechanism seems realistic for *Credit Scoring* applications, for example when financing is based solely on a function of $\boldsymbol{x}$, *e.g.* in the case of a score associated to a cut-off, provided all clients' characteristics of this existing score are included in $\boldsymbol{x}$. It is a usual assumption in *Credit Scoring* even if, in practice, the financing mechanism may depend also on unobserved features (thus not present in $\boldsymbol{x}$), which is particularly true when an operator (see Figure 2.1) adds a subjective, often intangible, expertise. In the MAR situation the log-likelihood (2.2) can be reduced to:

$$\ell(\gamma;\mathcal{T}) = \ell(\boldsymbol{\theta};\mathcal{T}_{\mathrm{f}}) + \sum_{i=1}^{n+n'} \ln p_{\boldsymbol{\phi}}(z_i|\boldsymbol{x}_i), \tag{2.5}$$

with $\ell(\boldsymbol{\theta};\mathcal{T}_{\mathrm{f}}) = \sum_{i\in\mathrm{F}} \ln p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)$. Combining it with the ignorable assumption, estimation of $\boldsymbol{\theta}$ relies only on the first part $\ell(\boldsymbol{\theta};\mathcal{T}_{\mathrm{f}})$, since the value $\boldsymbol{\phi}$ has no influence on $\boldsymbol{\theta}$. In that case, invoking an EM algorithm due to missing data y is no longer required as will be made explicit in Section 2.3.2. In practice, if there are other features $\tilde{\boldsymbol{x}}$ that are either present in the preceding scorecard or that are available to manual operators that influence the outcome y (see Figures 2.1 and 2.3), a MNAR missingness mechanism has to be assumed. However, as is generally assumed in Credit Scoring, the MAR mechanism can be safely hypothesized since features used in the scorecards do not differ much (all available features are usually retained) and lots of portfolios (*e.g.* for low amounts) have almost-automatic acceptance rules (such that manual operators follow the decision of the scorecards and only control the adequacy between declarative features $\boldsymbol{x}$ and supporting documents).

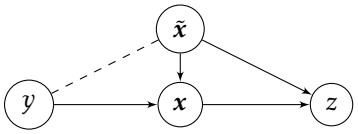


Figure 2.3 – Dependencies between random variables y, $\tilde{\boldsymbol{x}}$, $\boldsymbol{x}$ and z

2.2.5 Model selection

At this step, several kinds of parametric model (2.1) have been assumed. It concerns obviously the parametric family $\{p_{\boldsymbol{\theta}}(y|\boldsymbol{x})\}_{\boldsymbol{\theta}\in\Theta}$, and also the missingness mechanism MAR or MNAR. However, it has to be noticed that MAR versus MNAR cannot be tested since we do not have access to y for not financed clients [11]. However, model selection is possible by modelling also the whole financing mechanism, namely the family $\{p_{\boldsymbol{\phi}}(z|\boldsymbol{x},y)\}_{\boldsymbol{\phi}\in\Phi}$.

Scoring for credit application can be recast as a semi-supervised classification problem [3]. In this case, classical model selection criteria can be divided into two categories [14]: either scoring performance criteria as *e.g.* error rate on a test set $\mathcal{T}^{\text{test}}$, or information criteria like *e.g.* BIC as was introduced in Section 1.3.3.

In the category of error rate criteria, the typical error rate is expressed as follows:

$$\text{Error}(\mathcal{T}^{\text{test}}) = \frac{1}{|\mathcal{T}^{\text{test}}|} \sum_{i\in\mathcal{T}^{\text{test}}} \mathbb{I}(\hat{y}_i \neq y_i), \tag{2.6}$$

where $\mathcal{T}^{\text{test}}$ is an i.i.d. test sample from $p(y|\boldsymbol{x})$ and where $\hat{y}_i$ is the estimated value of the related y_i value involved by the estimated model at hand. The model leading to the lowest error value is then retained. However, in the *Credit Scoring* context this criterion family is not available since no sample $\mathcal{T}^{\text{test}}$ is itself available. This problem can be exhibited through the following straightforward expression

$$p(y|\boldsymbol{x}) = \sum_{z\in\{\text{f},\text{nf}\}} p(y|\boldsymbol{x},z)p(z|\boldsymbol{x}) \tag{2.7}$$

where $p(y|\boldsymbol{x},z)$ is unknown and $p(z|\boldsymbol{x})$ is known since this latter is defined by the financial institution itself. We notice that obtaining a sample from $p(y|\boldsymbol{x})$ would require that the financial institution draws $\mathbf{z}^{\text{test}}$ i.i.d. from $p(z|\boldsymbol{x})$ before to observe the results $\mathbf{y}^{\text{test}}$ i.i.d. from $p(y|\boldsymbol{x},z)$. But in practice it is obviously not the case, a threshold being applied to the distribution $p(z|\boldsymbol{x})$ for retaining only a set of fundable applicants, the non-fundable applicants being definitively discarded, preventing us from getting a test sample $\mathcal{T}^{\text{test}}$ from $p(y|\boldsymbol{x})$. As a matter of fact, only a sample $\mathcal{T}_{\text{f}}^{\text{test}}$ of $p(y|\boldsymbol{x},\text{f})$ is available, irrevocably prohibiting the calculus of (2.6) as a model selection criterion.

In the category of information criteria, the BIC criterion (presented in Section 1.3.3) is expressed as the following penalization of the maximum log-likelihood:

$$\text{BIC} = -2\ell(\hat{\gamma};\mathcal{T}) + \dim(\Gamma)\ln n, \tag{2.8}$$

where $\hat{\gamma}$ is the maximum likelihood estimate of γ and $\dim(\Gamma)$ is the number of parameters to be estimated in the model at hand. The model leading to the lowest BIC value is then retained. Many other BIC-like criteria exist [14] but the underlined idea is unchanged. Contrary to the error rate criteria like (2.6), it is thus possible to compare models without funding "non-fundable applicants" since just the available sample $\mathcal{T}$ is required. However, computing (2.8) requires to

precisely express the model families $\{p_\gamma(y,z|\boldsymbol{x})\}_{\gamma\in\Gamma}$ which compete.

2.3 Rational reinterpretation of reject inference methods

2.3.1 The reject inference challenge

As discussed in the previous section, a regular way to use the whole observed sample $\mathcal{T}$ in the estimation process implies some challenging modelling and assumption steps. A method using the whole sample $\mathcal{T}$ is traditionally called a reject inference method since it uses not only financed applicants (sample $\mathcal{T}_\mathrm{f}$) but also not financed, or rejected, applicants (sample $\mathcal{T}_\mathrm{nf}$). Since modelling the financing mechanism $p(z|\boldsymbol{x},y)$ is sometimes a too heavy task, such methods propose alternatively to use the whole sample $\mathcal{T}$ in a more empirical manner. However, this is somehow a risky strategy since we have also seen in the previous section that validating methods with error rate like criteria is not possible through the standard *Credit Scoring* process. As a result, some strategies are proposed to perform a "good" score function estimation without possibility to access their real performance.

However, most of the proposed reject inference strategies may make some hidden assumptions on the modelling process. Our challenge is to reveal as far as possible such hidden assumptions to then discuss their realism, failing to be able to compare them by the model selection principle.

2.3.2 Strategy 1: ignoring not financed clients

Definition

The simplest reject inference strategy is to ignore not financed clients for estimating $\boldsymbol{\theta}$. Thus it consists to estimate $\boldsymbol{\theta}$ by maximizing the log-likelihood $\ell(\boldsymbol{\theta};\mathcal{T}_\mathrm{f})$.

Missing data reformulation

In fact, this strategy is equivalent to using the whole sample $\mathcal{T}$ (financed and not financed applicants) under both the MAR and ignorable assumptions. See the related explanation in Section 2.2.4 and [17]. Consequently, this strategy is truly a particular "reject inference" strategy although it does not seem to be.

Estimate property

By noting $\hat{\boldsymbol{\theta}}_\mathrm{f}$ and $\hat{\boldsymbol{\theta}}$ respectively the maximum likelihood estimates of $\ell(\boldsymbol{\theta};\mathcal{T}_\mathrm{f})$ and $\ell(\boldsymbol{\theta};\mathcal{T}_\mathrm{c})$ provided we know y_i for $i\in\mathrm{NF}$, classical maximum likelihood properties [16, 17] yield under a well-specified model hypothesis (there exists $\boldsymbol{\theta}^\star$ s.t. $p(y|\boldsymbol{x})=p_{\boldsymbol{\theta}^\star}(y|\boldsymbol{x})$ for all $(\boldsymbol{x},y)$) and a MAR missingness mechanism that $\hat{\boldsymbol{\theta}}\approx\hat{\boldsymbol{\theta}}_\mathrm{f}$ for large-enough samples $\mathcal{T}_\mathrm{f}$ and $\mathcal{T}_\mathrm{nf}$.

2.3.3 Strategy 2: fuzzy augmentation

Definition

This strategy can be found in [12] and is developed in depth in Appendix A.1.1. It corresponds to an algorithm which is starting with $\hat{\boldsymbol{\theta}}^{(0)}=\hat{\boldsymbol{\theta}}_\mathrm{f}$ (see previous section). Then, all $(y_i)^{(1)}_{i\in\mathrm{NF}}$ are imputed by their expected value given by: $\hat{y}_i^{(1)}=p_{\hat{\boldsymbol{\theta}}^{(0)}}(1|\boldsymbol{x}_i)$ (notice that these imputed values

are not in $\{0,1\}$ but in $]0,1[$. The completed log-likelihood $\ell_c(\boldsymbol{\theta};\mathcal{T}_c^{(1)})$ given in (2.4) with $\mathcal{T}_c^{(1)} = \mathcal{T}_f \cup (y_i)_{i\in NF}^{(1)}$ is maximized and yields parameter estimate $\hat{\boldsymbol{\theta}}^{(1)}$.

Missing data reformulation

Following the notations introduced in Section 2.2.3, and recalling that this method does not take into account the financing mechanism $p(z|\boldsymbol{x},y)$, this method is an Expectation Maximization (EM)-algorithm yielding $\hat{\boldsymbol{\theta}}^{(1)} = \text{argmax}_{\boldsymbol{\theta}}\, \mathbb{E}_{\mathbf{y}_{nf}}[\ell_c(\boldsymbol{\theta};\mathcal{T}_c^{(1)})|\mathcal{T},\hat{\boldsymbol{\theta}}^{(0)}]$. The complete data $\mathcal{T}_c$ can be schematically expressed as:

$$\mathcal{T}_c^{(1)} = \begin{pmatrix} \mathbf{x}_f \\ \mathbf{x}_{nf} \end{pmatrix} \begin{pmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \vdots & \vdots \\ x_{n,1} & \cdots & x_{n,d} \\ x_{n+1,1} & \cdots & x_{n+1,d} \\ \vdots & \vdots & \vdots \\ x_{n+n',1} & \cdots & x_{n+n',d} \end{pmatrix}, \quad \begin{pmatrix} \mathbf{y}_f \\ \mathbf{y}_{nf} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \\ \hat{y}_{n+1}^{(1)} \\ \vdots \\ \hat{y}_{n+n'}^{(1)} \end{pmatrix}, \quad \begin{pmatrix} \mathbf{z}_f \\ \mathbf{z}_{nf} \end{pmatrix} \begin{pmatrix} \text{f} \\ \vdots \\ \text{f} \\ \text{nf} \\ \vdots \\ \text{nf} \end{pmatrix}.$$

Estimate property

It can be easily shown (Appendix A.1.1) that $\text{argmax}_{\boldsymbol{\theta}}\, \ell_c(\boldsymbol{\theta};\mathcal{T}_c^{(1)}) = \hat{\boldsymbol{\theta}}_f$ so that this method is similar to the scorecard learnt on the financed clients.

2.3.4 Strategy 3: reclassification

Definition

This strategy corresponds to an algorithm which is starting with $\hat{\boldsymbol{\theta}}^{(0)} = \hat{\boldsymbol{\theta}}_f$ (see Section 2.3.2). Then, all $(y_i)_{i\in NF}^{(1)}$ are imputed by the *maximum a posteriori* (MAP) principle given by: $\hat{y}_i^{(1)} = \text{argmax}_{y\in\{0,1\}}\, p_{\hat{\boldsymbol{\theta}}^{(0)}}(y|\boldsymbol{x}_i)$. The completed log-likelihood $\ell_c(\boldsymbol{\theta};\mathcal{T}_c^{(1)})$ given in (2.4) with $\mathcal{T}_c^{(1)} = \mathcal{T} \cup (y_i)_{i\in NF}^{(1)}$ is maximized and yields parameter estimate $\hat{\boldsymbol{\theta}}^{(1)}$.

Its first variant stops at this value $\hat{\boldsymbol{\theta}}^{(1)}$. Its second variant iterates until potential convergence of $(\hat{\boldsymbol{\theta}}^{(s)})$, s designing the iteration number. In practice, this method can be found in [7] under the name "iterative reclassification", in [15] under the name "reclassification" or under the name "extrapolation" in [1]. It is developed in depth in Appendix A.1.2.

Missing data reformulation

This algorithm is equivalent to the so-called Classification Expectation Maximization (CEM) algorithm where a Classification (or MAP) step is inserted between the Expectation and Maximization steps of an EM algorithm (described in Section 2.2.3). CEM aims at maximizing the completed log-likelihood $\ell_c(\boldsymbol{\theta};\mathcal{T}_c)$ over both $\boldsymbol{\theta}$ and $(y_i)_{i\in NF}$. Since $\boldsymbol{\phi}$ is not involved in this process, we first deduce from Section 2.2.4 that, again, MAR and ignorable assumptions are present. Then, standard properties of the estimate maximizing the completed likelihood indicate that it is not a consistent estimate of $\boldsymbol{\theta}$ [2], contrary to the traditional maximum likelihood one.

Estimate property

The CEM algorithm is known for "sharpening" the decision boundary: predicted probabilities are closer to 0 and 1 than their true values as can be seen from simulated data from a MAR mechanism on Figure 2.4. The scorecard $\hat{\theta}_{\mathrm{f}}$ on financed clients (in green) is asymptotically consistent as was emphasized in Section 2.3.2 while the reclassified scorecard (in red) is biased even asymptotically.

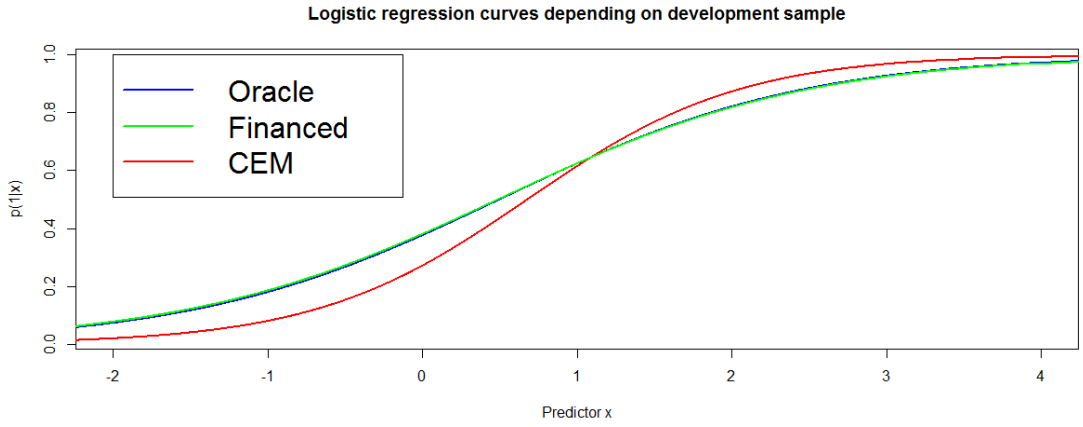


Figure 2.4 – In the context of a probabilistic classifier, it is known that the CEM algorithm employed implicitly by the Reclassification method amounts to a bigger bias in terms of logistic regression parameters, but a "sharper" decision boundary.

2.3.5 Strategy 4: augmentation

Definition

Augmentation can be found in [15]. It is also documented as a "Re-Weighting method" in [7, 1, 12] and is described in Appendix A.1.3. This technique is directly influenced by the Importance Sampling [17] literature because intuitively, as for all selection mechanism such as survey respondents, observations should be weighted according to their probability of being in the sample w.r.t. the whole population, i.e. by $p(z|\boldsymbol{x}, y)$. By assuming implicitly a MAR missingness mechanism, as emphasized in Section 2.2.4, we get $p(z|\boldsymbol{x}, y) = p(z|\boldsymbol{x})$.

For *Credit Scoring* practitioner, the estimate of interest is the Maximum Likelihood Estimation (MLE) of $\ell(\boldsymbol{\theta}; \mathcal{T} \cup \mathbf{y}_{\mathrm{nf}})$, which cannot be computed since we don't know $\mathbf{y}_{\mathrm{nf}}$. However, in Section 1.3.2, we derived the likelihood from the KL divergence by focusing on $\mathbb{E}_{\boldsymbol{X},Y}[\ln[p_{\boldsymbol{\theta}}(Y|\boldsymbol{X})]]$. By noticing that $p(\boldsymbol{x}) = \dfrac{p(\boldsymbol{x}|\mathrm{f})}{p(\mathrm{f}|\boldsymbol{x})} p(\mathrm{f})$ and by assuming a MAR missingness mechanism, we get:

$$\mathbb{E}_{\boldsymbol{X},Y}[\ln[p_{\boldsymbol{\theta}}(Y|\boldsymbol{X})]] = p(\mathrm{f}) \sum_{y=0}^{1} \int_{\mathcal{X}} \frac{\ln p_{\boldsymbol{\theta}}(y|\boldsymbol{x})}{p(\mathrm{f}|\boldsymbol{x})} p(y|\boldsymbol{x}) p(\boldsymbol{x}|\mathrm{f}) d\boldsymbol{x} \approx_{n\to\infty} \frac{p(\mathrm{f})}{n} \sum_{i \in \mathrm{F}} \frac{1}{p(\mathrm{f}|\boldsymbol{x}_i)} \ln p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i).$$

Consequently, had we access to $p(\mathrm{f}|\boldsymbol{x})$, the parameter maximizing the above mentioned likelihood would asymptotically be equal to the one on the through-the-door population, had we access to

$\mathbf{y}_{\text{nf}}$. However, $p(\text{f}|\boldsymbol{x})$ must be estimated by the practitioner's method of choice, which will come with its bias and variance.

This method proposes to bin observations in $\mathcal{T}$ in, say, 10 *equal-length* intervals of the score given by $p_{\hat{\theta}_\text{f}}(1|\boldsymbol{x})$ and estimate $p(z|\boldsymbol{x})$ as the proportion of financed clients in each of these bins. The inverse of this estimate is then used to weight financed clients in $\mathcal{T}_\text{f}$ and retrain the model.

Missing data reformulation

The method aims at correcting for the selection procedure yielding the training data $\mathcal{T}_\text{f}$ in the MAR case. As was argued in Section 2.3.2, if the model is well-specified, such a procedure is superfluous as the estimated parameter $\hat{\boldsymbol{\theta}}_\text{f}$ is consistent. In the misspecified case however, it is theoretically justified as will be developed in the next paragraph. However, it is unclear if this apparent benefit is not offset by the added estimation procedure (which comes with its bias / variance trade-off).

Estimate property

The Importance Sampling paradigm requires $p(\text{f}|\boldsymbol{x}) > 0$ for all $\boldsymbol{x}$ which is clearly not the case here: for example, jobless people are never financed.

2.3.6 Strategy 5: twins

Definition

This reject inference method is documented internally at CACF and in Appendix A.1.4; it consists in combining two scorecards: one predicting y learnt on financed clients (denoted by $\hat{\boldsymbol{\theta}}_\text{f}$ as previously), the other predicting z learnt on all applicants, before learning the final scorecard using the predictions made by both scorecards on financed clients.

Missing data reformulation

The method aims at re-injecting information about the financing mechanism in the MAR missingness mechanism by estimating $\hat{\boldsymbol{\phi}}$ as a logistic regression on all applicants, calculating scores $(1,\boldsymbol{x})'\hat{\boldsymbol{\theta}}_\text{f}$ and $(1,\boldsymbol{x})'\hat{\boldsymbol{\phi}}$ and use these as two continuous features in a third logistic regression predicting again the repayment feature y.

Estimate property

It can be easily shown (Appendix A.1.4) that this method is similar to the scorecard learnt on the financed clients.

2.3.7 Strategy 6: parcelling

Definition

The parcelling method can be found in [7, 1, 15]. It is also described in Appendix A.1.5. This method aims to correct the log-likelihood estimation in the MNAR case by making further assumptions on $p(y|\boldsymbol{x},z)$. It is a little deviation from the fuzzy augmentation method in a MNAR setting: we start with $\hat{\boldsymbol{\theta}}^{(0)} = \hat{\boldsymbol{\theta}}_\text{f}$ and the practitioner arbitrarily decides to discretize the subsequent range of scores $(p_{\hat{\boldsymbol{\theta}}^{(0)}}(y_i|\boldsymbol{x}_i))_1^{n+n'}$ into, say, K scorebands $B_1,\dots,B_K$ and "prudence factors" $\boldsymbol{\epsilon} = (\epsilon_1,\dots,\epsilon_K)$ generally such that $1 < \epsilon_1 < \cdots < \epsilon_K$ (non-financed low refunding probability

clients are considered way riskier, all other things equal, than their financed counterparts). The method is thereafter strictly equivalent to fuzzy reclassification: a new logistic regression parameter is deduced from maximizing the expected complete log-likelihood as follows:

$$\hat{\boldsymbol{\theta}}^{(1)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathbb{E}_{\mathbf{y}_{\mathrm{nf}}}[\ell_c(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{c}}) | \mathcal{T}, \hat{\boldsymbol{\theta}}^{(0)}, \boldsymbol{\epsilon}] = \ell(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{f}}) + \sum_{i'=n+1}^{n+n'} \epsilon_i p_{\hat{\boldsymbol{\theta}}^{(0)}}(y_i | \boldsymbol{x}_i) \ln p_{\boldsymbol{\theta}}(y_i | \boldsymbol{x}_i),$$

where $\epsilon_i = \sum_{k=1}^{K} \epsilon_k \mathbb{1}(p_{\hat{\boldsymbol{\theta}}^{(0)}}(y_i | \boldsymbol{x}_i) \in B_k)$ is simply the prudence factor of each individual, depending on their scoreband, decided by the practitioner.

Missing data reformulation

By considering not-financed clients as riskier than financed clients with the same level of score, *i.e.* $p(y|\boldsymbol{x}, \mathrm{nf}) > p(y|\boldsymbol{x}, \mathrm{f})$, it is implicitly assumed that manual operators (see Figure 2.1) have access to additional information, say $\tilde{\boldsymbol{x}}$ such as supporting documents, that influence the outcome y even when $\boldsymbol{x}$ is accounted for. In this setting, rejected and accepted clients with the same score differ only by $\tilde{\boldsymbol{x}}$, to which we do not have access and is accounted for "quantitatively" in a user-defined prudence factor $\boldsymbol{\epsilon}$ stating that rejected clients would have been riskier than accepted ones.

Estimate property

The prudence factor encompasses the practitioner's *belief* about the effectiveness of the operators' rejections. It cannot be estimated from the data nor tested and is consequently a matter of unverifiable expert knowledge.

2.4 Numerical experiments

Appendix A.1.6 shows all reject inference methods developed here applied to simulated data from a multivariate Gaussian distribution for each class. A logistic regression is learnt on all drawn observations, yielding $\hat{\boldsymbol{\theta}} = \operatorname{argmax}_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{c}})$ where $\mathbf{y}_{\mathrm{nf}}$ is known. We simulate not-financed clients by progressively "rejecting" observations such that $p_{\hat{\boldsymbol{\theta}}}(1|\boldsymbol{x}_i) < \epsilon$ with a varying threshold ϵ. This corresponds to a well-specified logistic regression and a MAR missingness mechanism. It comes as no surprise that no reject inference method performs better than standard logistic regression on financed clients (strategy 1). Since the data generating mechanism is Gaussian homoscedastic, we also resorted to semi-supervised linear discriminant analysis (LDA). This is straightforward with the Rmixmod R package [9]: the membership of unlabeled observations is assessed by the EM algorithm. Although the decision boundaries of these two models is asymptotically equivalent, by making further assumptions, LDA suffers from less variance even asymptotically [5]. With less financed clients, its advantage becomes clearer since it benefits from information on non-financed clients to evaluate $p(\boldsymbol{x})$. This model was tested on real data but since the normality assumption does not hold, it shows poor results.

Here we focus on reject inference methods based on logistic regression applied to various CACF datasets: Electronics loans, Sports goods and Standard loans. They contain $n = 180{,}000$, $n = 35{,}000$ and $n = 28{,}000$ respectively and $d = 5$, $d = 8$ and $d = 6$ categorical features with 3 to 10 levels per feature. The Electronics dataset consists in one year of financed clients through a partner of CACF that mainly sells electronics goods. The Sports dataset consists in one year of financed clients through a partner of CACF that sells all kinds of sports goods. The Standard

consists in one year of financed clients stemming directly from sofinco.fr. The acceptance / rejection mechanism $p_\phi(z|\boldsymbol{x})$ is the existing scorecard and we simulate rejected applicants by progressively increasing the cut (the preceding threshold ϵ) of the existing scorecard.

The results in terms of Gini index are reported in Figure 2.5, 2.6 and 2.7 respectively. All methods perform relatively similarly and suffer from a big performance drop once the acceptance rate is below 50 % (at which point there are very few "bad borrower" events - $y = 0$). If we were to report 95 % confidence intervals around the Gini indices, we would get insignificant predictive performances, which confirms our theoretical findings.

Other predictive models are compared to logistic regression on the real datasets presented here in Appendix A.1.7. These experiments confirm that "global" methods in the sense of [17] (which explicitly or implicitly estimate $p(\boldsymbol{x})$ to access $p(y|\boldsymbol{x})$ like LDA without unlabelled observations) degrade rapidly when the proportion of financed clients decreases.

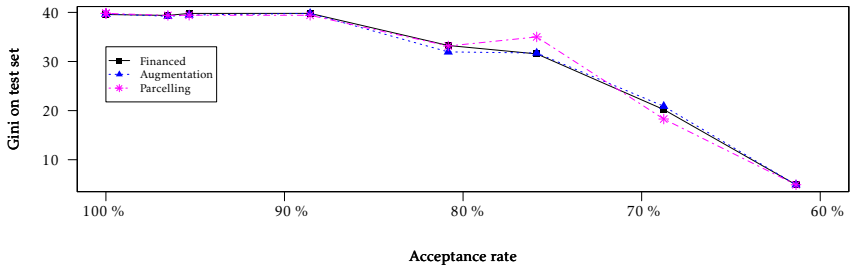


Figure 2.5 – Performance resulting from the use of reject inference methods in terms of Gini on an Electronics loans dataset from CACF.

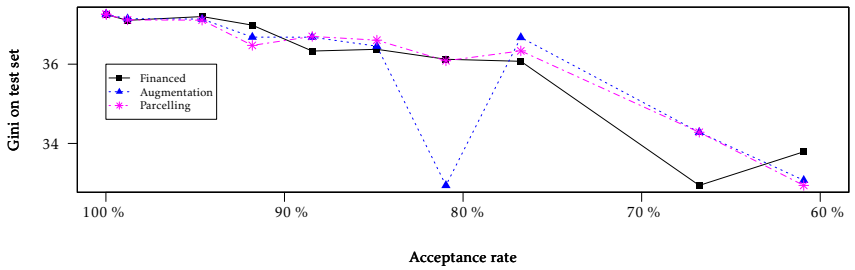


Figure 2.6 – Performance resulting from the use of reject inference methods in terms of Gini on a Sports goods loans dataset from CACF.

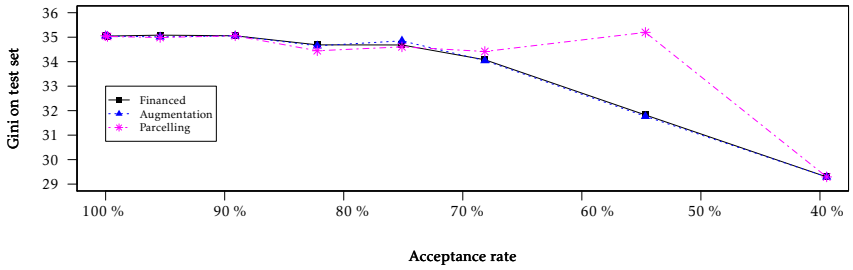


Figure 2.7 – Performance resulting from the use of Reject Inference methods in terms of Gini on a Standard loans dataset from CACF.

2.5 Discussion: choosing the right model

2.5.1 Sticking with the financed clients model

Constructing scorecards by using a logistic regression on financed clients is a trade-off: on the one hand, it is implicitly assumed that it is well-specified, and that the missingness mechanism governing the observation of Y is MAR. In other words, we suppose $p(y|\boldsymbol{x}) = p_{\theta^\star}(y|\boldsymbol{x}, \mathrm{f})$. On the other hand, these assumptions, which seem strong at first hand, cannot really be relaxed: first, the use of logistic regression is a requirement from the financial institution. Second, the comparison of models cannot be performed using standard techniques since $\mathbf{y}_{\mathrm{nf}}$ is missing (section 2.2.5). Third, strategies 4 and 6 that tackle the misspecified model and MNAR settings respectively require additional estimation procedures that, supplemental to their estimation bias and variance (section 1.3.2), take time from the practitioner's perspective and are rather subjective (see sections 2.3.5 and 2.3.7), which is not ideal in the banking industry since there are auditing processes and model validation teams that might question these practices.

2.5.2 MCAR through a Control Group

Another simple solution often stated in the literature would be to keep a small portion of the population where applicants are not filtered: everyone gets accepted. This so-called *Control Group* would constitute the learning and test sets for all scorecard developments.

Although theoretically perfect, this solution faces a major drawback: it is costly, as many more loans will default. To construct the scorecard, a lot of data is required, so the minimum size of the *Control Group* is equivalent to a much bigger loss than the amount a bank would accept to lose to get a few more Gini points.

2.5.3 Keep several models in production: "champion challengers"

Several scorecards could also be developed, e.g. one using each reject inference technique. Each application is randomly scored by one of these scorecards. As time goes by, we would be able to put more weight on the most performing scorecard(s) and progressively less on the least performing one(s): this is the field of Reinforcement Learning [13].

The major drawback of this method, although its cost is very limited unlike the *Control Group*, is that it is very time-consuming for the credit modeller who has to develop several scorecards,

for the IT who has to put them all into production, for the auditing process and for the regulatory institutions.

For years, the necessity of reject inference at CACF and other institutions (as it seems from the large literature coverage this research area has had) has been a question of personal belief. Moreover, there even exists contradictory findings in this area.

By formalizing the reject inference problem in Section 2.2, we were able to pinpoint in which cases the current scorecard construction methodology, using only financed clients' data, could be unsatisfactory: under a MNAR missingness mechanism and/or a misspecified model. We also defined criteria to reinterpret existing reject inference methods and assess their performance in Section 2.2.5. We concluded that no current reject inference method could enhance the current scorecard construction methodology: only the augmentation method (strategy 4) and the parcelling method (strategy 6) had theoretical justifications but introduce other estimation procedures. Additionally, they cannot be compared through classical model selection tools (Section 2.2.5).

We confirmed numerically these findings in the Appendices: given a true model and the MAR assumption, no logistic regression-based reject inference method performed best than the current method. In the misspecified model case, the augmentation method seemed promising but it introduces a model that also comes with its bias and variance resulting in very close performances compared with the current method. With real data provided by CACF, we showed that all methods gave very similar results: the "best" method (by the Gini index) was highly dependent on the data and/or the proportion of unlabelled observations. Last but not least, in practice such a benchmark would not be tractable as $\mathbf{y}_{\text{nf}}$ is missing. In light of those limitations, adding to the fact that implementing those methods is a non-negligible time-consuming task, we recommend credit modellers to work only with financed loans' data unless there is significant information available on either rejected applicants ($\mathbf{y}_{\text{nf}}$ - credit bureau information for example, which does not apply to France) or on the acceptance mechanism ϕ in the MNAR setting. On a side note, it must be emphasized that this work only applies to logistic regression and can be extended to all "local" models per the terminology introduced in [17]. For "global" models, *e.g.* decision trees, it can be shown that they are biased even in the MAR and well-specified settings, thus requiring *ad hoc* reject inference techniques such as an adaptation of the augmentation method (strategy 4 - see [17]).

References of Chapter 2

[1] John Banasik and Jonathan Crook. « Reject inference, augmentation, and sample selection ». In: *European Journal of Operational Research* 183.3 (2007), pp. 1582–1594. URL: http://www.sciencedirect.com/science/article/pii/S0377221706011969 (visited on 08/25/2016).

[2] Gilles Celeux and Gérard Govaert. « A classification EM algorithm for clustering and two stochastic versions ». In: *Computational statistics & Data analysis* 14.3 (1992), pp. 315–332.

[3] Olivier Chapelle, Bernhard Schlkopf, and Alexander Zien. *Semi-Supervised Learning*. 1st. The MIT Press, 2010. ISBN: 0262514125, 9780262514125.

[4] Arthur P Dempster, Nan M Laird, and Donald B Rubin. « Maximum likelihood from incomplete data via the EM algorithm ». In: *Journal of the royal statistical society. Series B (methodological)* (1977), pp. 1–38.

[5] Bradley Efron. « The efficiency of logistic regression compared to normal discriminant analysis ». In: *Journal of the American Statistical Association* 70.352 (1975), pp. 892–898.

[6] A. Feelders. « Credit scoring and reject inference with mixture models ». In: *International Journal of Intelligent Systems in Accounting, Finance & Management* 9.1 (2000), pp. 1–8. URL: http://www.ingentaconnect.com/content/jws/isaf/2000/00000009/00000001/art00177 (visited on 08/25/2016).

[7] Asma Guizani et al. « Une comparaison de quatre techniques d'inférence des refusés dans le processus d'octroi de crédit ». In: *45 emes Journées de statistique*. 2013. URL: http://cedric.cnam.fr/fichiers/art_2753.pdf (visited on 08/25/2016).

[8] Nicholas M. Kiefer and C. Erik Larson. « Specification and informational issues in credit scoring ». In: *Available at SSRN 956628* (2006). URL: http://papers.ssrn.com/sol3/papers.cfm?abstract_id=956628 (visited on 08/25/2016).

[9] Rémi Lebret et al. « Rmixmod: the R package of the model-based unsupervised, supervised and semi-supervised classification mixmod library ». In: *Journal of Statistical Software* 67.6 (2014), pp. 241–270.

[10] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[11] Geert Molenberghs et al. « Every missingness not at random model has a missingness at random counterpart with equal fit ». In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 7.2 (2008), pp. 371–388.

[12] Ha Thu Nguyen. *Reject inference in application scorecards: evidence from France*. Tech. rep. University of Paris West-Nanterre la Défense, EconomiX, 2016. URL: http://economix.fr/pdf/dt/2016/WP_EcoX_2016-10.pdf (visited on 08/25/2016).

[13] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018. URL: http://incompleteideas.net/book/the-book-2nd.html.

[14] Vincent Vandewalle. « Estimation et sélection en classification semi-supervisée ». Theses. Université des Sciences et Technologie de Lille - Lille I, Dec. 2009. URL: https://tel.archives-ouvertes.fr/tel-00447141.

[15] Emmanuel Viennet, Françoise Fogelman Soulié, and Benoît Rognier. « Evaluation de Techniques de Traitement des Refusés pour l'Octroi de Crédit ». In: *arXiv preprint cs/0607048* (2006). URL: http://arxiv.org/abs/cs/0607048 (visited on 08/25/2016).

[16] Halbert White. « Maximum Likelihood Estimation of Misspecified Models ». In: *Econometrica* 50.1 (1982), pp. 1–25. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1912526.

[17] Bianca Zadrozny. « Learning and evaluating classifiers under sample selection bias ». In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 114.

Chapter 3

Supervised multivariate quantization

All models are wrong, but some are useful.

Georges Box, "Empirical Model-Building and Response Surfaces", 1978.

Sommaire

Table 3.1 – Example of a final scorecard on quantized data.

Feature	Level	Points
Age	18-25	10
	25-45	20
	45-+∞	30
Wages	−∞-1000	15
	1000-2000	25
	2000-+∞	35
...	...	...

To improve prediction accuracy and interpretability of logistic regression-based scorecards, a preprocessing step quantizing both continuous and categorical data is usually performed: continuous features are discretized by assigning factor levels to intervals and, if numerous, levels of categorical features are grouped. However, a better predictive accuracy can be reached by embedding this quantization estimation step directly into the predictive estimation step itself. By doing so, the predictive loss has to be optimized on a huge and intractable discontinuous quantization set. To overcome this difficulty, we introduced a specific two-step optimization strategy: first, the optimization problem is relaxed by approximating discontinuous quantization functions by smooth functions; second, the resulting relaxed optimization problem is solved either *via* a particular neural network and a stochastic gradient descent or an SEM algorithm. These strategies give then access to good candidates for the original optimization problem after a straightforward *maximum a posteriori* procedure to obtain cutpoints. The good performances of these approaches, which we call *glmdisc*-NN and *glmdisc*-SEM respectively, are illustrated on simulated and real data from the UCI library and CACF. The results show that practitioners finally have an automatic all-in-one tool that answers their recurring needs of quantization for predictive tasks.

3.1 Motivation

As stated in [20] and illustrated in this manuscript, in many application contexts (*Credit Scoring*, biostatistics, *etc.*), logistic regression is widely used for its simplicity, decent performance and interpretability in predicting a binary outcome given predictors of different types (categorical, continuous). However, to achieve higher interpretability, continuous predictors are sometimes discretized so as to produce a "scorecard", *i.e.* a table assigning a grade to an applicant in *Credit Scoring* (or a patient in biostatistics, *etc.*) depending on its predictors being in a given interval, as exemplified in Table 3.1.

Discretization is also an opportunity for reducing the (possibly large) modeling bias which can appear in logistic regression as a result of the linearity assumption on the continuous predictors in the model which was discussed in Section 1.3 of Chapter 1. Indeed, this restriction can be overcome by approximating the true predictive mapping with a step function where the tuning of the steps and their sizes allow more flexibility. However, the resulting increase of the number of parameters can lead to an increase in variance (overfitting) as shown in [43]. Thus, a precise tuning of the discretization procedure is required. Likewise when dealing with categorical features which take numerous levels, their respective regression coefficients suffer from high variance. A straightforward solution formalized by [26] is to merge their factor levels

which leads to less coefficients and therefore less variance. We showcase this phenomenon on simple simulated data in the next section. On *Credit Scoring* data, a typical example is the number of children (although not continuous strictly speaking). The log-odd ratio of clients' creditworthiness w.r.t. their number of children is often visually "quadratic", *i.e.* the risk is lower for clients having 1 to 3 children, a bit higher for 0 child, and then it grows steadily with the number of children above 4. This can be fitted with a parabola, see Figure 3.1a. As using a spline is not very interpretable, this is not done in practice. Without quantizing the number of children, a linear relationship is assumed as displayed on Figure 3.1b. When quantizing this feature, a piecewise constant relationship is assumed, see Figure 3.1c. In this example, it is visually unclear which is best, such that there is a need to formalize the problem.

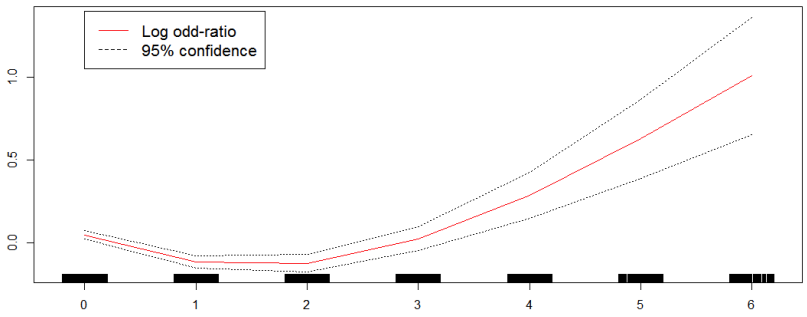
Another potential motivation for quantization is optimal data compression: as will be shown rigorously in subsequent sections, quantization aims at "squeezing" as much predictive information in the original features about the class as possible. Taking an informatics point of view, quantization of a continuous feature is equivalent to discarding a float column (taking *e.g.* 32 bits per observation) by overwriting it with its quantized version (which would either be one column of unsigned 8 bits integers - "interval" encoding without order - or several 1 bit columns - one-hot / dummy encoding). The same thought process is applicable to quantizations of categorical features. In the end, the "raw" data can be compressed by a factor of $32/8 = 4$ without losing its predictive power, which, in an era of Big Data, is useful both in terms of data storage and of computational power to process these data since by 2040, the energy needs for calculations will exceed the global energy production (see [41] p. 123).

From now on, the generic term quantization will stand for both discretization of continuous features and level grouping of categorical ones. Its aim is to improve the prediction accuracy. Such a quantization can be seen as a special case of *representation learning* [3], but suffers from a highly combinatorial optimization problem whatever the predictive criterion used to select the best quantization. The present work proposes a strategy to overcome these combinatorial issues by invoking a relaxed alternative of the initial quantization problem leading to a simpler estimation problem since it can be easily optimized by either a specific neural network or an SEM algorithm. These relaxed versions serve as a plausible quantization provider related to the initial criterion after a classical thresholding (*maximum a posteriori*) procedure.

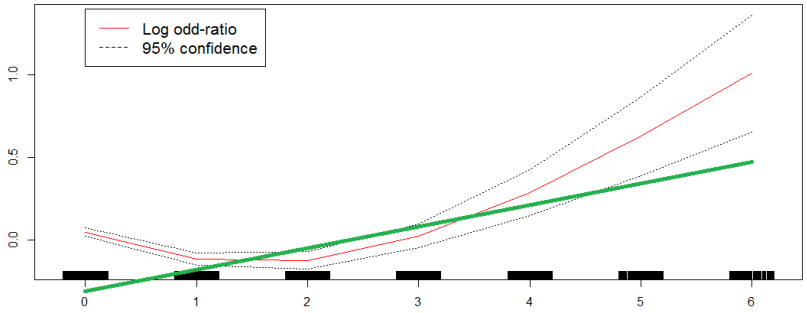
The outline of this chapter is the following. After some introductory examples, we illustrate cases where quantization is either beneficial or detrimental depending on the data generating mechanism. In the subsequent section, we formalize both continuous and categorical quantization. Selecting the best quantization in a predictive setting is reformulated as a model selection problem on a huge discrete space which size is precisely derived. In Section 3.4, a particular neural network architecture is used to optimize a relaxed version of this criterion and propose good quantization candidates. In Section 3.5, an SEM procedure is proposed to solve the quantization problem. Section 3.6 is dedicated to numerical experiments on both simulated and real data from the field of Credit Scoring, high-lightening the good results offered by the use of the two new methods without any human intervention. A final section concludes the chapter by stating also new challenges.

3.2 Illustration of the bias-variance quantization trade-off

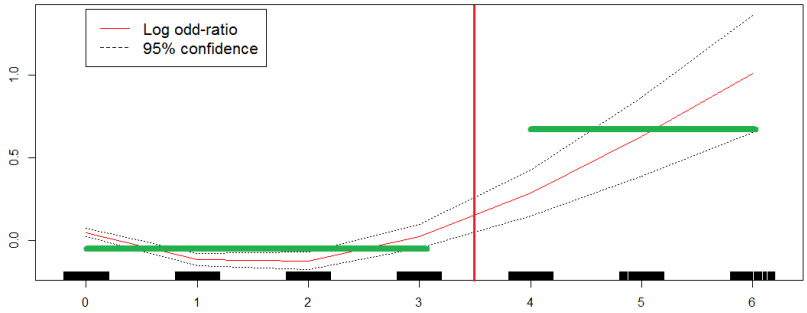
The previous section motivated the use of quantization on a practical level. On a theoretical level, at least in terms of probability theory, quantization is equivalent to throwing away information: for continuous features, it is only known that they belong to a certain interval and for categorical features, their granularity among the original levels is lost.



(a) Risk of CACF clients w.r.t. their number of children and output of a spline regression.



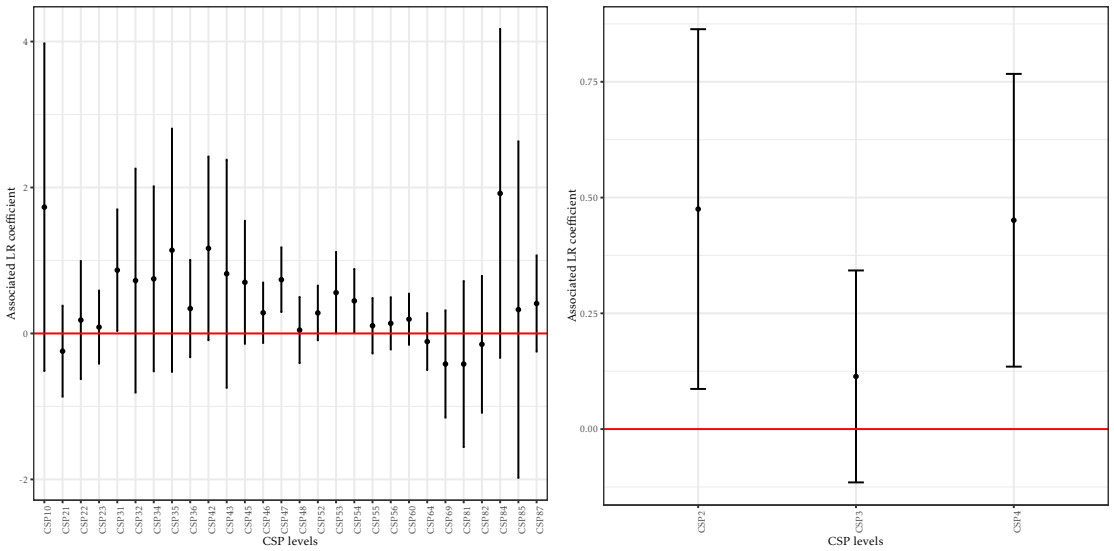
(b) When the logistic regression is used without quantization, it amounts to assuming the green linear relationship.



(c) When the logistic regression is used with quantization, *e.g.* more or less than 3 children, it amounts to assuming the risk is similar for all levels and equals the green steps.

Figure 3.1 – Relationship of the creditworthiness of a client w.r.t. his / her number of children, all else being equal.

However, two things must appear clearly: first, we are in a "statistical" setting, *i.e.* finite-dimensional setting, where variance of estimation can play a big role, as was developed in Section 1.3.2, which partly justifies the need to regroup categorical levels. Second, we are in a predictive setting, with an imposed classification model p_θ. We focus on logistic regression, for which continuous features get a single coefficient: their relationship with the logit transform of the probability of an event (bad borrower) is assumed to be linear which can yield model bias. Thus, having several coefficients per feature, which can be achieved with a variety of techniques (*e.g.* splines), can yield a lower model bias (when the true model is not linear, which is generally the case for *Credit Scoring* data) at the cost of increased variance of estimation.



(a) Having a lot of levels means having lots of coefficients, few of which are significant.

(b) By grouping levels, fewer coefficients are obtained, which variance is significantly smaller and are thus significant.

Figure 3.2 – logistic regression coefficients of the levels of the job of borrowers.

This phenomenon can be very simply captured by a small simulation: in the misspecified model setting, where the logit transform is assumed to stem from a sinusoidal transformation of x on $[0;1]$, it can clearly be seen from Figure 3.4a that a standard linear logistic regression performs poorly. Discretizing the feature x results, using a very simple unsupervised heuristic named *equal-length* (described in-depth in Appendix A.2.1), in good results (*i.e.* visually mild bias / low variance) so long as the number of intervals, and subsequently of logistic regression coefficients, is low (see the animation on Figure 3.3 or still on Figure 3.4b). When the number of intervals gets large, the bias gets low (the sinus is well approximated by the little step functions), but the variance gets bigger (see the animation on Figure 3.3 or still on Figure 3.4c).

As the number of intervals is directly linked to the number of coefficient, and to a notion of "complexity" of the resulting logistic regression model, the bias-variance trade-off (introduced in Section 1.3.2 of Chapter 1) plays a key role in choosing an appropriate step size, and, as will be seen in the next section which was not possible for the simple *equal-length* algorithm, appropriate step locations (hereafter called cutpoints). Again, this can be witnessed visually by looking at a model selection criterion, *e.g.* the BIC criterion (which was introduced in Section 1.3.3 of Chapter 1), for different values of the number of intervals on Figure 3.5. As expected, the continuous fit is poor, yielding a high BIC value. For a low number of bins, as described in the previous paragraph, the steps of Figure 3.3 are poor approximations of the true relationship between x and y resulting in a high BIC value. By discretizing in more intervals, the BIC value gets lower, and eventually starts to increase again when variance kicks in and overfitting occurs. As was visually concluded from Figure 3.3, somewhere around 10-15 intervals seem the most satisfactory since we clearly witness a low BIC value. Of course, as the model was misspecified, the flexibility brought by discretization was beneficial. The same phenomenon can be witnessed for categorical features on Figure 3.2 with real data from CACF. On Figure 3.2a, the logistic regression coefficients of the raw job types are displayed: none are significant and estimation

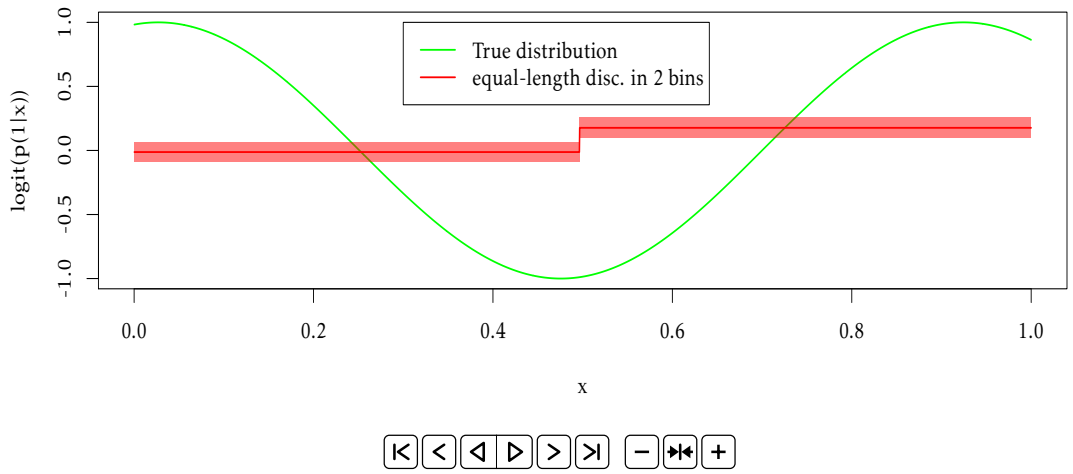


Figure 3.3 – Animation of logistic regression fits on data generated by a sinus with a number of discretization steps in the *equal-length* algorithm ranging from 2 to 100.

variance is large. Grouping these levels results in narrower confidence intervals and significant logistic regression parameters as can be seen in Figure 3.2b. We formalize these empirical findings in the next section.

3.3 Quantization as a combinatorial challenge

3.3.1 Quantization: definition

General principle The quantization procedure consists in turning a d-dimensional raw vector of continuous and/or categorical features $\boldsymbol{x} = (x_1, \ldots, x_d)$ into a d-dimensional categorical vector *via* a component-wise mapping $\boldsymbol{q} = (\boldsymbol{q}_j)_1^d$:

$$\boldsymbol{q}(\boldsymbol{x}) = (\boldsymbol{q}_1(x_1), \ldots, \boldsymbol{q}_d(x_d)).$$

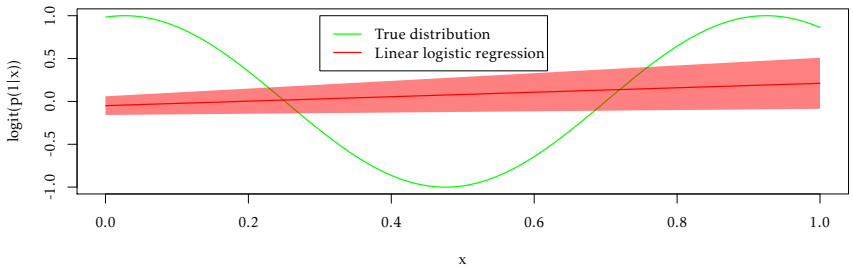
Each of the univariate quantizations $\boldsymbol{q}_j(x_j) = (q_{j,1}(x_j), \ldots, q_{j,m_j}(x_j))$ is a vector of m_j dummies:

$$q_{j,h}(x_j) = 1 \text{ if } x_j \in C_{j,h}, 0 \text{ otherwise}, 1 \leq h \leq m_j, \tag{3.1}$$

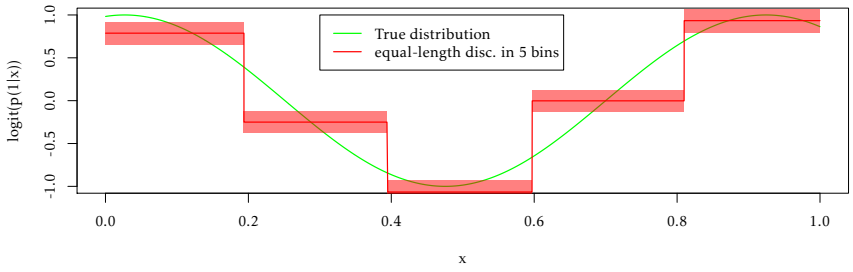
where m_j is an integer, denoting the number of intervals / groups to which x_j is mapped and the sets $C_{j,h}$ are defined with respect to each feature type as is described just below.

Raw continuous features If x_j is a continuous component of $\boldsymbol{x}$, quantization $\boldsymbol{q}_j$ has to perform a discretization of x_j and the $C_{j,h}$'s, $1 \leq h \leq m_j$, are contiguous intervals:

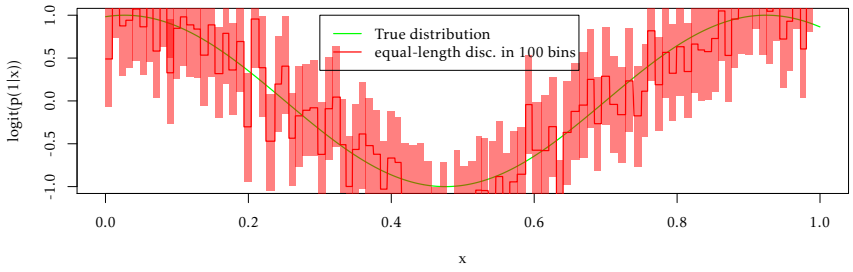
$$C_{j,h} = (c_{j,h-1}, c_{j,h}], \tag{3.2}$$



(a) Linear logistic regression (in red) fit (the shaded area represents the 95 % confidence interval) on data generated by a sinus (in green).



(b) Logistic regression fit (in red) on data generated by a sinus (in green) with 5 discretization steps in the *equal-length* algorithm (the shaded area represents the 95 % confidence interval).



(c) Logistic regression fit (in red) on data generated by a sinus (in green) with 100 discretization steps in the *equal-length* algorithm (the shaded area represents the 95 % confidence interval).

Figure 3.4 – Motivational example: achieving a good bias-variance trade-off.

where $c_{j,1}, \ldots, c_{j,m_j-1}$ are increasing numbers called cutpoints, $c_{j,0} = -\infty$, $c_{j,m_j} = \infty$. For example, the quantization of the unit segment in thirds would be defined as $m_j = 3$, $c_{j,1} = 1/3$, $c_{j,2} = 2/3$ and subsequently $\boldsymbol{q}_j(0.1) = (1, 0, 0)$. This is visually exemplified on Figure 3.6.

Figure 3.5 – BIC of the resulting logistic regression on quantized data in green with a varying number of bins in the *equal-length* algorithm and the linear logistic regression Gini in red (both misspecified).

Raw categorical features If x_j is a categorical component of $\boldsymbol{x}$, quantization $\boldsymbol{q}_j$ consists in grouping levels of x_j and the $C_{j,h}$s form a partition of the set $\mathbb{N}_{l_j} = \{1, \ldots, l_j\}$:

$$\bigcup_{h=1}^{m_j} C_{j,h} = \mathbb{N}_{l_j},$$
$$\forall h, h', C_{j,h} \cap C_{j,h'} = \varnothing.$$

For example, the grouping of levels encoded as "1" and "2" would yield $C_{j,1} = \{1, 2\}$ such that $\boldsymbol{q}_j(1) = \boldsymbol{q}_j(2) = (1, 0, \ldots, 0)$. Note that it is assumed that there are no empty buckets, *i.e.* $\nexists j, h$ s.t. $C_{j,h} = \varnothing$. This is visually exemplified on Figure 3.7.

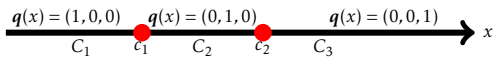


Figure 3.6 – Quantization (discretization) of a continuous feature.

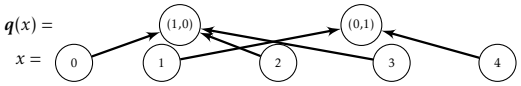


Figure 3.7 – Quantization (factor levels merging) of categorical feature.

3.3.2 Cardinality of the quantization family

Notations for the quantization family In both continuous and categorical cases, keep in mind that m_j is the dimension of $\boldsymbol{q}_j$. For notational convenience, the (global) order of the quantization $\boldsymbol{q}$ is set as

$$|\boldsymbol{q}| = \sum_{j=1}^{d} m_j.$$

The space where quantizations $\boldsymbol{q}$ live (resp. $\boldsymbol{q}_j$) will be denoted by $\mathcal{Q}_{\boldsymbol{m}}$ in the sequel (resp. $\mathcal{Q}_{j,m_j}$), when the number of levels $\boldsymbol{m} = (m_j)_1^d$ is fixed. Since it is not known, the full model space is $\boldsymbol{Q} = \cup_{\boldsymbol{m} \in \mathbb{N}_\star^d} \mathcal{Q}_{\boldsymbol{m}}$ where $\mathbb{N}_\star^d = (\mathbb{N} \setminus \{0\})^d$.

Equivalence of quantizations Let $\boldsymbol{q}^1$ and $\boldsymbol{q}^2$ in $\boldsymbol{Q}$ such that $\boldsymbol{q}^1 \mathcal{R}_{\mathcal{I}_\mathrm{f}} \boldsymbol{q}^2 \equiv \forall i, j\ \boldsymbol{q}_j^1(x_{i,j}) = \boldsymbol{q}_j^2(x_{i,j})$. See Figure 3.8 for an example.

Lemma Relation $\mathcal{R}_{\mathcal{I}_\mathrm{f}}$ defines an equivalence relation on $\boldsymbol{Q}$.

Proof. Relation $\mathcal{R}_{\mathcal{I}_\mathrm{f}}$ is trivially reflexive and symmetric because of the reflexive and symmetric nature of the equality relation in $\mathbb{R}$: $\forall i, j\ \boldsymbol{q}_j^1(x_{i,j}) = \boldsymbol{q}_j^1(x_{i,j})$ and $\forall i, j\ \boldsymbol{q}^1(x_{i,j}) = \boldsymbol{q}^2(x_{i,j})$. Similarly, let $\boldsymbol{q}^3 \in \boldsymbol{Q}$ such that $\boldsymbol{q}^1 \mathcal{R}_{\mathcal{I}_\mathrm{f}} \boldsymbol{q}^3 \equiv \forall i, j\ \boldsymbol{q}_j^1(x_{i,j}) = \boldsymbol{q}_j^3(x_{i,j})$. Again, we immediately get $\forall i, j\ \boldsymbol{q}_j^2(x_{i,j}) = \boldsymbol{q}_j^3(x_{i,j})$, *i.e.* $\boldsymbol{q}^2 \mathcal{R}_{\mathcal{I}_\mathrm{f}} \boldsymbol{q}^3$ which proves the transitivity of $\mathcal{R}_{\mathcal{I}_\mathrm{f}}$. □

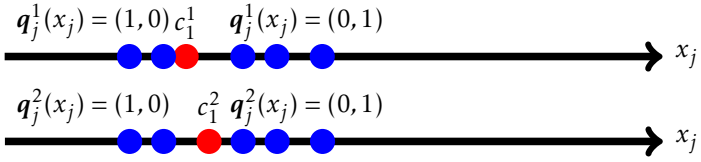


Figure 3.8 – On the sample $\mathbf{x}$ (blue points), the two discretization functions $\boldsymbol{q}^1$ and $\boldsymbol{q}^2$ (which respective unique cutpoint c_1^1 and c_1^2 are displayed in red) take the same value and are thus equivalent w.r.t. $\mathcal{R}_{\mathcal{I}_\mathrm{f}}$.

Cardinality of the quantization family in the continuous case
For a continuous feature x_j, let $\boldsymbol{q}_j \in \mathcal{Q}_{j,m_j}$ and cutpoints $\boldsymbol{c}_j$. Without any loss of generality, *i.e.* up to a relabelling on individuals i, it can be assumed that there are no empty intervals (all m_j levels are observed) and consequently $m_j + 1$ observations $x_{1,j}, \ldots, x_{m_j+1,j}$ s.t. $x_{1,j} < c_{j,1} < x_{2,j} < \cdots < c_{m_j-1,1} < x_{m_j+1,j}$. Indeed, if for example there exists $k < m_j - 1$ s.t. $c_{j,k} < \cdots < c_{j,m_j-1}$ and $\max_{1 \leq i \leq n} x_{i,j} < c_{j,k}$, then discretization $\boldsymbol{q}_j^{\mathrm{bis}} \in \boldsymbol{Q}_{j,k}$ with $k+1$ cutpoints $(-\infty, c_{j,1}, \ldots, c_{j,k-1}, +\infty)$ is equivalent w.r.t. $\mathcal{R}_{\mathcal{I}_\mathrm{f}}$ to $\boldsymbol{q}_j$: $\forall i,\ \boldsymbol{q}_j(x_{i,j}) = \boldsymbol{q}_j^{\mathrm{bis}}(x_{i,j})$. A similar proof can be conducted with cutpoints below the minimum of $\boldsymbol{x}_j$ or with several cutpoints in-between consecutive values of the observations. Subsequently, there are $\binom{n-1}{m_j-1}$ ways to construct $\boldsymbol{c}_j$, *i.e.* equivalence classes $[\boldsymbol{q}_j]$ for a fixed $m_j \leq n$. The number of intervals m_j can range from 2 (binarization) to n (each $x_{i,j}$ is in its own interval, thus $\boldsymbol{q}_j(x_{i,j}) \neq \boldsymbol{q}_j(x_{i',j})$ for $i \neq i'$;), so that the number of admissible discretizations of $\mathbf{x}_j$ is $|\mathcal{Q}_j| = \sum_{i=2}^{n} \binom{n-1}{i-1}$. Note that $|\mathcal{Q}_j|$ depends on the number of observations n; we shall go back to this property in the following section.

Cardinality of the quantization family in the categorical case For a continuous feature x_j, let $\boldsymbol{q}_j \in \mathcal{Q}_j$ with m_j groups. The number of re-arrangements of l_j labelled elements into m_j unlabelled groups is given by the Stirling number of the second kind $S(l_j, m_j) = \frac{1}{m_j!} \sum_{i=0}^{m_j} (-1)^{m_j - i} \binom{m_j}{i} i^{l_j}$. As m_j is unknown, it must be searched over the range $\{1, \ldots, l_j\}$. Thus for categorical features, model space $\mathcal{Q}_j$ is also discrete; subsequently, $\boldsymbol{Q}$ is discrete.

3.3.3 Literature review

The current practice of quantization is prior to any predictive task, thus ignoring its consequences on the final predictive ability. It consists in optimizing a heuristic criterion, often totally

unrelated (unsupervised methods) or at least explicitly (supervised methods) to prediction, and mostly univariate (each feature is quantized irrespective of other features' values). The cardinality of the quantization space $\boldsymbol{Q}$ was calculated explicitely w.r.t. d, $(m_j)_1^d$ and, for categorical features, l_j: it is huge, so that a greedy approach is intractable and such heuristics are needed, as will be detailed in the next section.

Many algorithms have thus been designed and a review of approximatively 200 discretization strategies, gathering both criteria and related algorithms, can be found in [30], preceded by other enlightening review articles such as [14, 25]. They classify discretization methods by distinguishing, among other criteria and as said previously, unsupervised and supervised methods (y is used to discretize $\boldsymbol{x}$), for which model-specific (assumptions on $p_{\boldsymbol{\theta}}$) or model-free approaches are distinguished, univariate and multivariate methods (features $\boldsymbol{x}_{-\{j\}} = (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d)$ may influence the quantization scheme of x_j) and other criteria as can be seen from Figure 3.9 reproduced from [30] with permission.

For factor levels grouping, we found no such taxonomy, but some discretization methods, *e.g.* χ^2 independence test-based methods can be naturally extended to this type of quantization, which is for example what the CHAID algorithm, proposed in [21] and applied to each categorical feature, relies on. A simple idea is also to use Group LASSO [28] which attempts to shrink to zero all coefficients of a categorical feature to avoid situations where a few levels enter the model, which is arguably less interpretable. Another idea would be to use Fused LASSO [38], which seeks to shrink the pairwise absolute difference of selected coefficients, and apply it to all pairs of levels: the levels for which the difference would be shrunk to zero would be grouped. A combination of both approaches would allow both selection and grouping [1].

For benchmarking purposes, and following results found in the taxonomy of [30], we used the MDLP [16] discretization method, described in-depth in Appendix A.2.2, which is a popular supervised univariate discretization method, and we implemented an extension of the discretization method ChiMerge [22] to categorical features, performing pairwise χ^2 independence tests rather than only pairs of contiguous intervals, which we called ChiCollapse and describe in-depth in Appendix A.3. Note that various refinements of ChiMerge have been proposed in the literature, Chi2 [24], ConMerge [42], ModifiedChi2 [37], and ExtendedChi2 [35], which seek to correct for multiple hypothesis testing [34] and automize the choice of the confidence parameter α in the χ^2 tests, but adapting them to categorical features for benchmarking purposes would have been too time-consuming. A similar measure, called Zeta, has been proposed in place of χ^2 in [19] and subsequent refinement [18]: it is the classification error achievable by using only two contiguous intervals; if it is low, the two intervals are dissimilar w.r.t. the prediction task, if not, they can be merged.

3.3.4 Quantization embedded in a predictive process

In what follows, focus is given to logistic regression since it is a requirement from CACF but it is applicable to any other supervised classification model.

Logistic regression on quantized data Quantization is a widespread preprocessing step to perform a learning task consisting in predicting, say, a binary variable $y \in \{0, 1\}$, from a quantized predictor $\boldsymbol{q}(\boldsymbol{x})$, through, say, a parametric conditional distribution $p_{\boldsymbol{\theta}}(y|\boldsymbol{q}(\boldsymbol{x}))$ like logistic regression; the whole process can be visually represented as a dependence structure among $\boldsymbol{x}$, its quantization $\boldsymbol{q}(\boldsymbol{x})$ and the target y on Figure 3.10. Considering quantized data instead of raw data has a double benefit. First, the quantization order $|\boldsymbol{q}|$ acts as a tuning parameter

1. See https://stats.stackexchange.com/questions/60100/penalized-methods-for-categorical-data-combining-levels-in-a-factor

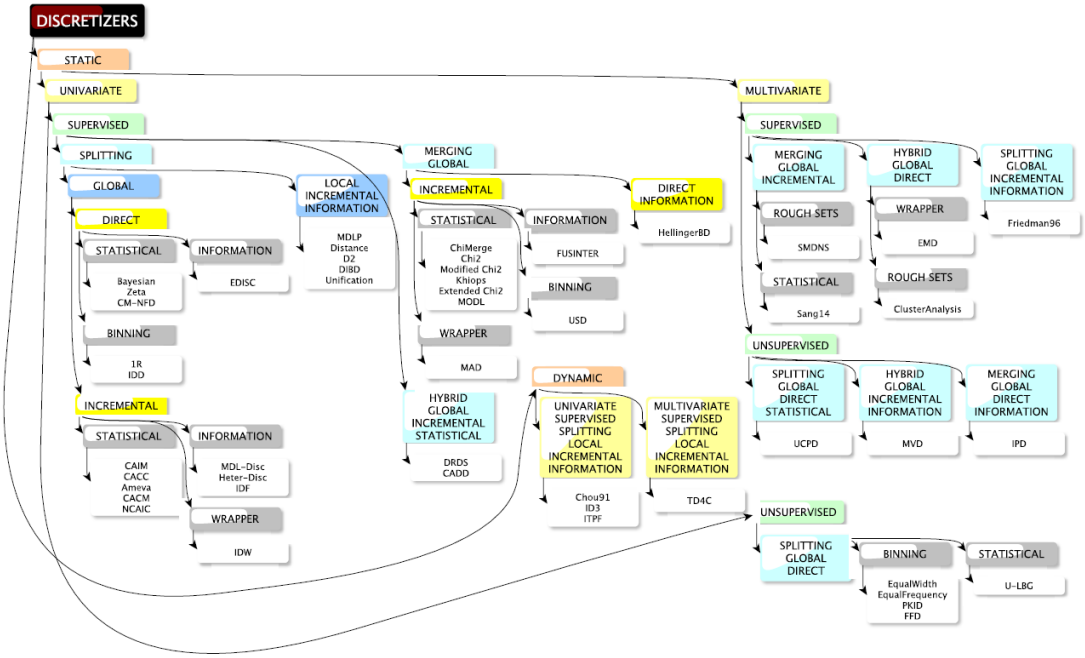


Figure 3.9 – Taxonomy of discretization methods.

for controlling the model's flexibility and thus the bias/variance trade-off of the estimate of the parameter $\boldsymbol{\theta}$ (or of its predictive accuracy) for a given dataset. This claim becomes clearer with the example of logistic regression we focus on, as a still very popular model for many practitioners. On quantized data, Equation (1.1) becomes:

$$\ln\left(\frac{p_{\boldsymbol{\theta}}(1|\boldsymbol{q}(\boldsymbol{x}))}{1-p_{\boldsymbol{\theta}}(1|\boldsymbol{q}(\boldsymbol{x}))}\right) = \theta_0 + \sum_{j=1}^{d} \boldsymbol{q}_j(x_j)'\boldsymbol{\theta}_j, \tag{3.3}$$

where $\boldsymbol{\theta} = (\theta_0, (\boldsymbol{\theta}_j)_1^d) \in \mathbb{R}^{|\boldsymbol{q}|+1}$ and $\boldsymbol{\theta}_j = (\theta_j^1, \ldots, \theta_j^{m_j})$ with $\theta_j^{m_j} = 0$, $j = 1 \ldots d$, for identifiability reasons (see Section 1.2.4). Second, at the practitioner level, the previous tuning of $|\boldsymbol{q}|$ through each feature's quantization order m_j, especially when it is quite low, allows an easier interpretation of the most important predictor values involved in the predictive process. Denoting the n-sample of financed clients as in the previous chapter by $(\mathbf{x}_\mathrm{f}, \mathbf{y}_\mathrm{f})$, with $\mathbf{x}_\mathrm{f} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ and $\mathbf{y}_\mathrm{f} = (y_1, \ldots, y_n)$, the log-likelihood

$$\ell_{\boldsymbol{q}}(\boldsymbol{\theta}; \mathcal{T}_\mathrm{f}) = \sum_{i=1}^{n} \ln p_{\boldsymbol{\theta}}(y_i|\boldsymbol{q}(\boldsymbol{x}_i)) \tag{3.4}$$

provides a maximum likelihood estimator $\hat{\boldsymbol{\theta}}_{\boldsymbol{q}}$ of $\boldsymbol{\theta}$ for a given quantization $\boldsymbol{q}$. For the rest of the chapter and consistently with the manuscript, the approach is exemplified with logistic regression as $p_{\boldsymbol{\theta}}$ but it can be applied to any other predictive model, as will be recalled in the concluding section.

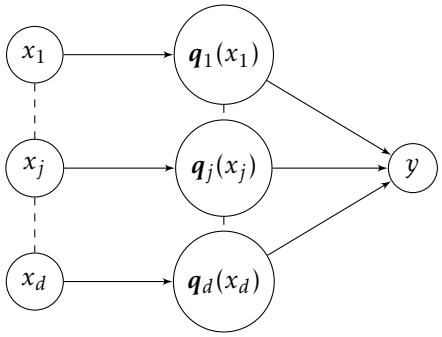


Figure 3.10 – Dependence structure between x_j, $\boldsymbol{q}_j$ and y.

Quantization as a model selection problem As dicussed in the previous section, and emphasized in the literature review, quantization is often a preprocessing step; however, quantization can be embedded directly in the predictive model. Continuing our logistic example, a standard information criteria such as the BIC (see Section 1.3.3) can be used to select the best quantization:

$$\hat{\boldsymbol{q}} = \underset{\boldsymbol{q} \in \boldsymbol{Q}}{\operatorname{argmin}} \operatorname{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q}}) \tag{3.5}$$

where the "complexity" parameter ν depends on $\boldsymbol{q}$ and is traditionally the number of continuous parameters to be estimated in the $\boldsymbol{\theta}$-parameter space. We shall insist here on the fact that choosing the BIC as our gold standard to compare quantizations is only a matter of consistency throughout the chapters. The practitioner can swap this criterion with any other penalized criterion on training data such as AIC [1] or, as *Credit Scoring* people like, the Gini index on a test set. Note however that, regardless of the used criterion, an exhaustive search of $\hat{\boldsymbol{q}} \in \boldsymbol{Q}$ is an intractable task due to its highly combinatorial nature as was explicitly formulated in the previous section. Anyway, the optimization (3.5) requires a new specific strategy that is described in the next section.

Remarks on model selection consistency In high-dimensional spaces and among models with a wildly varying number of parameters, classical model selection tools like BIC can have disappointing asymptotic properties, as emphasized in [10], where a modified BIC criterion, taking into account the number of models per parameter size, is proposed. Moreover in essence, as is apparent from the $\hat{\boldsymbol{\theta}}_{\boldsymbol{q}}$ symbol, and supplemental to the logistic regression coefficients *per se*, the inherent "parameters" $C_{j,h}$ of $\boldsymbol{q}$ (see Equation (3.2)) shall be accounted for in the penalization term ν: they are estimated indirectly in all quantization methods, and in particular in the one we propose in the subsequent section.

In addition, the BIC criterion relies on the Laplace approximation [23] which requires the likelihood to be twice differentiable in the parameters. However, as $\boldsymbol{q}$ consists in a collection of step functions of parameters $C_{j,h}$, this is not the case. For continuous features, since it is nevertheless almost everywhere differentiable, for the properties of the BIC criterion to hold, it suffices that there exists a neighbourhood $V_{j,h}$ around true parameters $c^{\star}_{j,h}$ where there is no observation: $\nexists i,\ x_{i,j} \in V_{j,h}$. For categorical features, the Laplace approximation [23] is no longer valid and there is no way, in general, to approximate the integral (*i.e.* the sum over the discrete parameter space) by "counting" the number of parameters as in the continuous case [40].

3.4 The proposed neural network based quantization

3.4.1 A relaxation of the optimization problem

In this section, we propose to relax the constraints on $\boldsymbol{q}_j$ to simplify the search of $\hat{\boldsymbol{q}}$. Indeed, the derivatives of $\boldsymbol{q}_j$ are zero almost everywhere and consequently a gradient descent cannot be directly applied to find an optimal quantization.

Smooth approximation of the quantization mapping A classical approach is to replace the binary functions $q_{j,h}$ (see Equation (3.1)) by smooth parametric ones with a simplex condition, namely with $\boldsymbol{\alpha}_j = (\boldsymbol{\alpha}_{j,1}, \ldots, \boldsymbol{\alpha}_{j,m_j})$:

$$\boldsymbol{q}_{\boldsymbol{\alpha}_j}(\cdot) = \left(q_{\boldsymbol{\alpha}_{j,h}}(\cdot)\right)_{h=1}^{m_j} \text{ with } \sum_{h=1}^{m_j} q_{\boldsymbol{\alpha}_{j,h}}(\cdot) = 1 \text{ and } 0 \leq q_{\boldsymbol{\alpha}_{j,h}}(\cdot) \leq 1,$$

where functions $q_{\boldsymbol{\alpha}_{j,h}}(\cdot)$, properly defined hereafter for both continuous and categorical features, represent a fuzzy quantization in that, here, each level h is weighted by $q_{\boldsymbol{\alpha}_{j,h}}(\cdot)$ instead of being selected once and for all as in Equation (3.1). The resulting fuzzy quantization for all components depends on the global parameter $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_d)$ and is denoted by $\boldsymbol{q}_{\boldsymbol{\alpha}}(\cdot) = \left(\boldsymbol{q}_{\boldsymbol{\alpha}_j}(\cdot)\right)_{j=1}^{d}$.

For continuous features, we set for $\boldsymbol{\alpha}_{j,h} = (\alpha_{j,h}^0, \alpha_{j,h}^1) \in \mathbb{R}^2$

$$q_{\boldsymbol{\alpha}_{j,h}}(\cdot) = \frac{\exp(\alpha_{j,h}^0 + \alpha_{j,h}^1 \cdot)}{\sum_{g=1}^{m_j} \exp(\alpha_{j,g}^0 + \alpha_{j,g}^1 \cdot)} \tag{3.6}$$

where $\boldsymbol{\alpha}_{j,m_j}$ is set to $(0,0)$ for identifiability reasons.

For categorical features, we set for $\boldsymbol{\alpha}_{j,h} = \left(\alpha_{j,h}(1), \ldots, \alpha_{j,h}(l_j)\right) \in \mathbb{R}^{l_j}$

$$q_{\boldsymbol{\alpha}_{j,h}}(\cdot) = \frac{\exp\left(\alpha_{j,h}(\cdot)\right)}{\sum_{h'=1}^{m_j} \exp\left(\alpha_{j,h'}(\cdot)\right)}$$

where l_j is the number of levels of the categorical feature x_j.

Parameter estimation With this new fuzzy quantization, the logistic regression for the predictive task is then expressed as

$$\ln\left(\frac{p_{\boldsymbol{\theta}}(1|\boldsymbol{q}_{\boldsymbol{\alpha}}(\boldsymbol{x}))}{1 - p_{\boldsymbol{\theta}}(1|\boldsymbol{q}_{\boldsymbol{\alpha}}(\boldsymbol{x}))}\right) = \theta_0 + \sum_{j=1}^{d} \boldsymbol{q}_{\boldsymbol{\alpha}_j}(x_j)' \boldsymbol{\theta}_j, \tag{3.7}$$

where $\boldsymbol{q}$ has been replaced by $\boldsymbol{q}_{\boldsymbol{\alpha}}$ from Equation (3.3). Note that as $\boldsymbol{q}_{\boldsymbol{\alpha}}$ is a sound approximation of $\boldsymbol{q}$ (see above), this logistic regression in $\boldsymbol{q}_{\boldsymbol{\alpha}}$ is consequently a good approximation of the logistic regression in $\boldsymbol{q}$ from Equation (3.3). The relevant log-likelihood is here

$$\ell_{\boldsymbol{q}_{\boldsymbol{\alpha}}}(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{f}}) = \sum_{i=1}^{n} \ln p_{\boldsymbol{\theta}}(y_i|\boldsymbol{q}_{\boldsymbol{\alpha}}(\boldsymbol{x}_i)) \tag{3.8}$$

and can be used as a tractable substitute for (3.4) to solve the original optimization problem (3.5), where now both $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ have to be estimated, which is discussed in the next section.

Deducing quantizations from the relaxed problem We wish to maximize the log-likelihood (3.7) which would yield parameters $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\theta}})$; these are consistent if the model is well-specified (*i.e.* there is a "true" quantization under classical regularity conditions). Denoting by A the space of $\boldsymbol{\alpha}$ and $\boldsymbol{Q}_A$ the space of $\boldsymbol{q}_{\boldsymbol{\alpha}}$, to "push" $\boldsymbol{Q}_A$ further into $\boldsymbol{Q}$, $\hat{\boldsymbol{q}}$ is deduced from a *maximum a posteriori* procedure applied to $\boldsymbol{q}_{\hat{\boldsymbol{\alpha}}}$:

$$\hat{q}_{j,h}(x_j) = 1 \equiv \hat{\boldsymbol{q}}_j(x_j) = \boldsymbol{e}_h^{m_j} \text{ if } h = \underset{1 \leq h' \leq m_j}{\operatorname{argmax}} q_{\hat{\boldsymbol{\alpha}}_{j,h'}}(x_j), 0 \text{ otherwise.} \tag{3.9}$$

If there are several levels h that satisfy (3.9), we simply take the level that corresponds to smaller values of x_j to be in accordance with the definition of $C_{j,h}$ in Equation (3.2). This *maximum a posteriori* principle will be exemplified in Figure 3.15 on simulated data. These approximations are justified by the following arguments.

Rationale From a deterministic point of view, we have $\boldsymbol{Q} \subset \boldsymbol{Q}_A$: First, the *maximum a posteriori* step (3.9) produces contiguous intervals (*i.e.* there exists $C_{j,h}$; $1 \leq j \leq d$, $1 \leq h \leq m_j$, s.t. $\hat{\boldsymbol{q}}$ can be written as in Equation (3.1)) [32]. Second, in the continuous case, the higher $\alpha^1_{j,h}$, the less smooth the transition from one quantization h to its "neighbor" [1] $h+1$, whereas $\dfrac{\alpha^0_{j,h}}{\alpha^1_{j,h}}$ controls the point in $\mathbb{R}$ where the transition occurs [9]. Concerning the categorical case, the rationale is even simpler as $q_{\lambda\boldsymbol{\alpha}_j}(x_j) \to \boldsymbol{e}_h^{m_j}$ if $h = \operatorname{argmax}_{h'} q_{\alpha_{j,h'}}(x_j)$ as $\lambda \to +\infty$ [31].

From a statistical point of view, provided the model is well-specified, *i.e.*:

$$\exists \boldsymbol{q}^\star, \boldsymbol{\theta}^\star, \forall \boldsymbol{x}, y,\ p(y|\boldsymbol{x}) = p_{\boldsymbol{\theta}^\star}(y|\boldsymbol{q}^\star(\boldsymbol{x})); \tag{3.10}$$

and under standard regularity conditions and with a suitable estimation procedure (see later for the proposed estimation procedure), the maximum likelihood framework would ensure the consistency of $(\boldsymbol{q}_{\hat{\boldsymbol{\alpha}}}, \hat{\boldsymbol{\theta}})$ towards $(\boldsymbol{q}^\star, \boldsymbol{\theta}^\star)$ if $\boldsymbol{\alpha}^\star$ s.t. $\boldsymbol{q}_{\boldsymbol{\alpha}^\star} = \boldsymbol{q}$ was an interior point of the parameter space A. However, as emphasized in the previous paragraph, "$\boldsymbol{\alpha}^\star = +\infty$" such that the maximum likelihood parameter is on the edge of the parameter space which hinders asymptotic properties (*e.g.* normality) in some settings [33], but not "convergence" on which we focus here. We did not investigate this issue further since numerical experiments showed consistency: from an empirical point of view, we will see in Section 3.6 and in particular in Figure 3.15, that the smooth approximation $\boldsymbol{q}_{\hat{\boldsymbol{\alpha}}}$ converges towards "hard" quantizations [1] $\boldsymbol{q}$.

However, and as is usual, the log-likelihood $\ell_{\boldsymbol{q}_{\boldsymbol{\alpha}}}(\boldsymbol{\theta}, \mathcal{T}_{\mathrm{f}})$ cannot be directly maximized w.r.t. $(\boldsymbol{\alpha}, \boldsymbol{\theta})$, so that we need an iterative procedure. To this end, the next section introduces a neural network of suitable architecture.

3.4.2 A neural network-based estimation strategy

Neural network architecture To estimate parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ in the model (3.7), a particular neural network architecture can be used. We shall insist that this network is only a way to use common deep learning frameworks, namely Tensorflow [27] through the high-level API

1. Up to a permutation on the labels $h = 1 \dots m_j$ to recover the ordering in $C_{j,h}$ (see Equation (3.2)).

Keras [11] instead of building a gradient descent algorithm from scratch to optimize (3.8). The most obvious part is the output layer that must produce $p_\theta(1|\boldsymbol{q}_{\boldsymbol{\alpha}}(\boldsymbol{x}))$ which is equivalent to a densely connected layer with a sigmoid activation (the reciprocal function of logit).

For a continuous feature x_j of $\boldsymbol{x}$, the combined use of m_j neurons including affine transformations and softmax activation obviously yields $\boldsymbol{q}_{\boldsymbol{\alpha}_j}(x_j)$. Similarly, an input categorical feature x_j with l_j levels is equivalent to l_j binary input neurons (presence or absence of the factor level). These l_j neurons are densely connected to m_j neurons without any bias term and a softmax activation. The softmax outputs are next aggregated via the summation in model (3.7), say Σ_θ for short, and then the sigmoid function σ gives the final output. All in all, the proposed model is straightforward to optimize with a simple neural network, as shown in Figure 3.11.

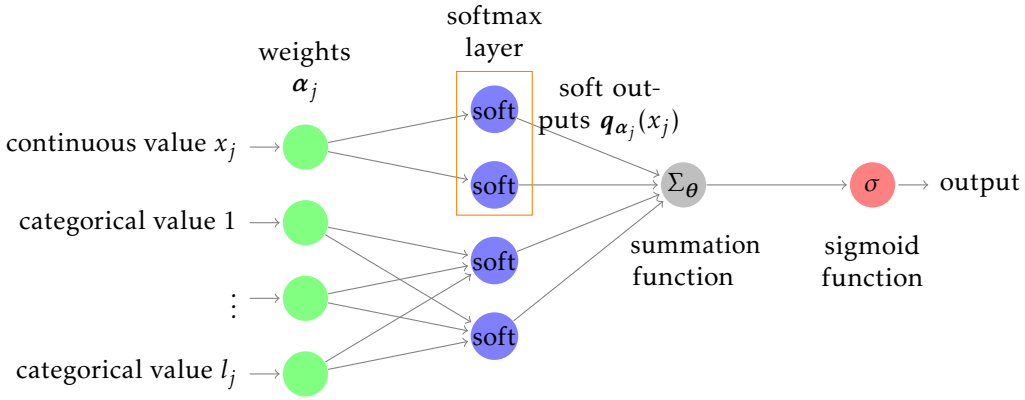


Figure 3.11 – Proposed shallow architecture to maximize (3.8).

Stochastic gradient descent as a quantization provider By relying on stochastic gradient descent, the smoothed likelihood (3.8) can be maximized over $(\boldsymbol{\alpha}, \boldsymbol{\theta})$. Due to its convergence properties [5], the results should be close to the maximizers of the original likelihood (3.4) if the model is well-specified, when there is a true underlying quantization. However, in the misspecified model case, there is no such guarantee. Therefore, to be more conservative, we evaluate at each training epoch (s) the quantization $\hat{\boldsymbol{q}}^{(s)}$ resulting from the *maximum a posteriori* procedure explicited in Equation (3.9), then classically estimate the logistic regression parameter *via* maximum likelihood, as done in Equation (3.4):

$$\hat{\boldsymbol{\theta}}^{(s)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell_{\hat{q}^{(s)}}(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{f}}) \tag{3.11}$$

and the resulting $\mathrm{BIC}(\hat{\boldsymbol{\theta}}^{(s)})$ as in (3.5). If S is a given maximum number of iterations of the stochastic gradient descent algorithm, the quantization retained at the end is then determined by the optimal epoch

$$s^\star = \underset{s\in\{1,\dots,S\}}{\operatorname{argmin}}\, \mathrm{BIC}(\hat{\boldsymbol{\theta}}^{(s)}). \tag{3.12}$$

You can think of S as a computational budget: contrary to classical early stopping rules (*e.g.* based on validation loss) used in neural network fitting, this network only acts as a stochastic quantization provider for (3.12) which will naturally prevent overfitting. We reiterate that, in (3.12), the BIC can be swapped for the user's favourite model choice criterion.

Lots of optimization algorithms for neural networks have been proposed, which all come with their hyperparameters. As, in the general case, $\ell_{q_{\boldsymbol{\alpha}}}(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{f}})$ of Equation (3.8) is not guaranteed to be convex, there might be several local maxima, such that all these optimization methods might diverge, converge to a different maximum, or at least converge in very different numbers of epochs, as can be examplified in the animation of Figure 3.12 [2]. We chose the RMSProp method, which showed good results and is one of the standard methods.

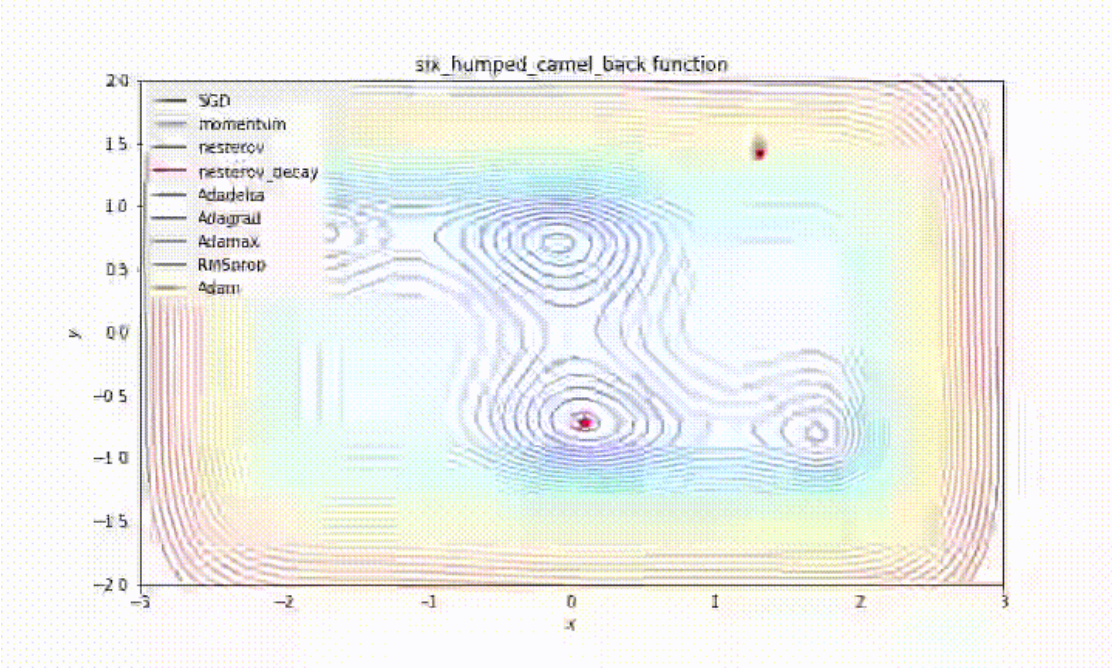


Figure 3.12 – Animation of several optimization methods (the ⋆ denotes the global maximum).

Choosing an appropriate number of levels Concerning now the number of intervals or factor levels $\boldsymbol{m} = (m_j)_1^d$, they have also to be estimated since in practice they are unknown. Looping over all candidates $\boldsymbol{m}$ is intractable. But in practice, by relying on the *maximum a posteriori* procedure developed in Equation (3.9), a lot of unseen factor level might be dropped. Indeed, for a given level h, all training observations $x_{i,j}$ in $\mathcal{T}_{\mathrm{f}}$ and all other levels h', if $q_{\boldsymbol{\alpha}_{j,h}}(x_{i,j}) < q_{\boldsymbol{\alpha}_{j,h'}}(x_{i,j})$, then the level h "vanishes".

This phenomenon can be witnessed in Figure 3.15a (algorithm and experiments detailed later) where $q_{\alpha_{0,2}}$ is "flat" across the support of $\mathbf{x}_0$ and only two intervals are produced. In practice, we recommend to start with a user-chosen $\boldsymbol{m} = \boldsymbol{m}_{\max}$ and we will see in the experiments of Section 3.6 that the proposed approach is able to explore small values of $\boldsymbol{m}$ and to select a value $\hat{\boldsymbol{m}}$ drastically smaller than $\boldsymbol{m}_{\max}$. This phenomenon, which reduces the computational burden of the quantization task, is also illustrated in Section 3.6.

The full algorithm is described in Appendix A.2.3.

2. Reproduced from `https://github.com/wassname/viz_torch_optim`

3.5 An alternative SEM approach

In what follows, the quantization $\boldsymbol{q}(\boldsymbol{x})$ is seen as a latent (unobserved) feature denoted by $\mathfrak{q}$, which is still the vector of quantizations $(\mathfrak{q}_1, \ldots, \mathfrak{q}_d)$ of features $(x_1, \ldots, x_d)$ (see Equation (3.1)). These component-wise quantizations are themselves binary-valued vectors $(\mathfrak{q}_{j,h})_{h=1}^{m_j}$ where $\mathfrak{q}_{j,h} \in \{0,1\}$ and $\sum_{h=1}^{m_j} \mathfrak{q}_{j,h} = 1$. We denote by $\mathfrak{Q}_{\boldsymbol{m}}$ the space of such latent features and the n-sample of latent quantizations corresponding to $\boldsymbol{q}(\mathbf{x}_{\mathrm{f}})$ by $\mathfrak{q}$, *i.e.* the matrix of all n quantizations.

In the following section, we translate earlier assumptions on the function $\boldsymbol{q}(\cdot)$ in probabilistic terms for the latent feature $\mathfrak{q}$. In the subsequent section, we make good use of these assumptions to provide a continuous relaxation of the quantization problem, as was empirically argumented in Section 3.4.1. The main reason to resort to this formulation as a latent feature problem will be made clear in Section 3.5.3, where we provide a stochastic estimation algorithm for the latent feature $\mathfrak{q}$.

3.5.1 Probabilistic assumptions regarding the quantization latent feature

Firstly, only the well-specified model case is considered. This hypothesis formalized in Equation (3.10) for the neural network approach, translates with this new latent feature as a probabilistic assumption:

$$\exists \boldsymbol{\theta}^\star, \mathfrak{q}^\star \text{ s.t. } Y \sim p_{\boldsymbol{\theta}^\star}(\cdot|\mathfrak{q}^\star). \tag{3.13}$$

Secondly, the result of the quantization is assumed to be "self-contained" w.r.t. the predictive information in $\boldsymbol{x}$, *i.e.* it is assumed that, independently from the logistic regression modelling, all available information about y in $\boldsymbol{x}$ has been "squeezed" by quantizing the data, as explained in Section 3.3.4 and Figure 3.11:

$$\forall \boldsymbol{x}, y,\ p(y|\boldsymbol{x}, \mathfrak{q}) = p(y|\mathfrak{q}). \tag{3.14}$$

Thirdly, the component-wise nature of the quantization can be stated as a conditional independence assumption as in Figure 3.11:

$$\forall \boldsymbol{x}, \mathfrak{q},\ p(\mathfrak{q}|\boldsymbol{x}) = \prod_{j=1}^{d} p(\mathfrak{q}_j|x_j). \tag{3.15}$$

3.5.2 Continuous relaxation of the quantization as seen as fuzzy assignment

If we consider the deterministic discretization scheme defined in Section 3.3, we have, analogous to Equation (3.1):

$$p(\mathfrak{q}_j = \boldsymbol{e}_h^{m_j} | x_j) = 1 \text{ if } x_j \in C_{j,h},$$

which is a step function. Rewriting $p(y|\boldsymbol{x})$ by integrating over these new latent features, we get:

$$\begin{aligned}
p(y|\boldsymbol{x}) &= \sum_{\mathfrak{q}\in\mathfrak{Q}_{\boldsymbol{m}}} p(y,\mathfrak{q}|\boldsymbol{x}) \\
&= \sum_{\mathfrak{q}\in\mathfrak{Q}_{\boldsymbol{m}}} p(y|\mathfrak{q},\boldsymbol{x})p(\mathfrak{q}|\boldsymbol{x}) \\
&= \sum_{\mathfrak{q}\in\mathfrak{Q}_{\boldsymbol{m}}} p(y|\mathfrak{q})p(\mathfrak{q}|\boldsymbol{x}) && \text{(using (3.14))} \\
&= \sum_{\mathfrak{q}\in\mathfrak{Q}_{\boldsymbol{m}}} p(y|\mathfrak{q})\prod_{j=1}^{d} p(\mathfrak{q}_j|x_j) && \text{(using (3.15)).}
\end{aligned}$$

This sum over $\mathfrak{Q}_{\boldsymbol{m}}$ is intractable. The well-specified model hypothesis (3.13) yields $p_{\theta^\star}(y|\mathfrak{q}^\star) = \sum_{\mathfrak{q}\in\mathfrak{Q}_{\boldsymbol{m}}} p(y|\mathfrak{q})\prod_{j=1}^{d} p(\mathfrak{q}_j|x_j)$; we claim that if the quantizations are "obvious" given the data, *i.e.* $p(\mathfrak{q}^\star|\boldsymbol{x}) \approx 1$, then the above sum reduces to $p_{\theta^\star}(y|\mathfrak{q}^\star)\prod_{j=1}^{d} p(\mathfrak{q}_j^\star|x_j)$. The same reasoning over all training observations in $\mathcal{T}_{\mathrm{f}}$ yields:

$$\begin{aligned}
p(\mathbf{y}_{\mathrm{f}}|\mathbf{x}_{\mathrm{f}}) &= \prod_{i=1}^{n} p(y_i|\boldsymbol{x}_i) \\
&= \sum_{\mathfrak{q}\in\mathfrak{Q}_{\boldsymbol{m}}} \prod_{i=1}^{n} p(y_i|\mathfrak{q}_i)\prod_{j=1}^{d} p(\mathfrak{q}_{i,j}|x_{i,j}) \\
&\approx \prod_{i=1}^{n} p_{\theta^\star}(y_i|\mathfrak{q}_i^\star)\prod_{j=1}^{d} p(\mathfrak{q}_{i,j}^\star|x_{i,j}).
\end{aligned}$$

Thus, we have :

$$\mathfrak{q}^\star \approx \underset{\theta\in\Theta,\mathfrak{q}\in\mathfrak{Q}_{\boldsymbol{m}}}{\operatorname{argmax}} \prod_{i=1}^{n} p_\theta(y_i|\mathfrak{q}_i)\prod_{j=1}^{d} p(\mathfrak{q}_{i,j}|x_{i,j}).$$

This new formulation of the best quantization is still intractable since it requires to evaluate all quantizations in $\mathfrak{Q}_{\boldsymbol{m}}$ (possibly for all $\boldsymbol{m}$ which we naturally denote by $\mathfrak{Q}$) just like in Equation (3.5). In the misspecfied model-case however, there is no such simplification but it can still be claimed that the best candidate $\mathfrak{q}^\star$ in terms of criterion (3.5) dominates the sum.

Our goal in the next section is to generate good candidates $\mathbf{q}$ as in Section 3.4.2. Among other things detailed later on, models for $p(y|\mathfrak{q})$ and $p(\mathfrak{q}_j|x_j)$ shall be proposed. A stochastic "quantization provider" is designed as in the previous section. Following arguments of the preceding paragraph, its empirical distribution of generated candidates shall be dominated by $\boldsymbol{q}^\star$, which, as in Section 3.4 with the neural network approach, can be selected with the BIC criterion (3.5). Using (3.13), it seems natural to use a logistic regression for $p(y|\mathfrak{q})$. Following Section 3.4 and as was empirically argumented in Section 3.4.1, the instrumental distribution $p(\mathfrak{q}_j|x_j)$ will take a similar form as $\boldsymbol{q}_{\boldsymbol{\alpha}}$. However, contrary to the neural network approach which iteratively optimizes $\boldsymbol{\theta}$ given the "fuzzy" quantization $\boldsymbol{q}_{\boldsymbol{\alpha}}$ (continuous values in $]0;1[$ for all its values), this approach iteratively draws candidates $\mathfrak{q}$ (where only one of its entries is equal to 1 for each feature, all others are equal to 0) which we call a "stochastic" quantization.

For a continuous feature, we resort to a polytomous logistic regression, similar to the softmax

function of Equation (3.6) without the over-parametrization (one level per feature j, say m_j, is considered reference):

$$p_{\boldsymbol{\alpha}_{j,h}}(\mathfrak{q}_j = \boldsymbol{e}_h^{m_j}|x_j) = \begin{cases} \frac{1}{\sum_{h'=1}^{m_j-1} \exp(\alpha_{j,h'}^0 + \alpha_{j,h'}^1 x_j)} & \text{if } h = m_j, \\ \frac{\alpha_{j,h}^0 + \alpha_{j,h}^1 x_j}{\sum_{h'=1}^{m_j-1} \exp(\alpha_{j,h'}^0 + \alpha_{j,h'}^1 x_j)} & \text{otherwise.} \end{cases}$$

For categorical features, simple contingency tables are employed:

$$p_{\boldsymbol{\alpha}_{j,h}}(\mathfrak{q}_j = \boldsymbol{e}_h^{m_j}|x_j = o) = \boldsymbol{\alpha}_{j,h}^o, 1 \leq o \leq l_j.$$

Similarly, $p_{\boldsymbol{\alpha}_j}(\mathfrak{q}_j|x_j)$ are no more step functions but smooth functions as in Figure 3.15.

Remark on polytomous logistic regressions Since the resulting latent categorical feature can be interpreted as an ordered categorical features (the *maximum a posterior* operation yields contiguous intervals as argued in Section 3.4.1), ordinal "parallel" logistic regression [29] could be used (provided levels h are reordered). This particular model is of the form:

$$\ln \frac{p(\mathfrak{q}_j = \boldsymbol{e}_{h+1}^{m_j}|x_j)}{p(\mathfrak{q}_j = \boldsymbol{e}_h^{m_j}|x_j)} = \alpha_{j,h,0} + \alpha_j x_j, 1 \leq h < m_j,$$

which restricts the number of parameters since all levels h share the same slope α_j. Its advantages lie in the fact that it might lead to sharper door functions quicker, and that it has fewer parameters to estimate, thus reducing *de facto* the estimation variance of each "soft" quantization $p_{\boldsymbol{\alpha}_j}$. However, it makes it harder for levels to "vanish" which would require to iterate over the number of levels per feature m_j which we wanted to avoid (see paragraph "Choosing an appropriate number of levels" in the next section). In practice, it yielded similar results to polytomous logistic regression such that they remain a parameter of the R package *glmdisc* (see Appendix B).

3.5.3 Stochastic search of the best quantization

We parametrized $p(y|\boldsymbol{x})$ as:

$$p(y|\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\alpha}) = \sum_{\mathfrak{q} \in Q} p_{\boldsymbol{\theta}}(y|\mathfrak{q}) \prod_{j=1}^{d} p_{\boldsymbol{\alpha}_j}(\mathfrak{q}_j|\boldsymbol{x}_j). \tag{3.16}$$

A straightforward way to maximize the likelihood of $p(y|\boldsymbol{x}; \boldsymbol{\theta}, \boldsymbol{\alpha})$ in $(\boldsymbol{\theta}, \boldsymbol{\alpha})$ (not to be mistaken with (3.8)), as was done in Section 3.4, to deduce $\hat{\boldsymbol{q}}$ from $\boldsymbol{\alpha}$ *via* the argmax operation (see Section 3.4.1 and Equation (3.9)), is to use an EM algorithm [12].

However, maximizing this likelihood directly is intractable as the Expectation step requires to sum over $\mathfrak{q} \in \mathfrak{Q}_{\boldsymbol{m}}$:

E-step:

$$t_{i,\mathfrak{q}}^{(s)} = \frac{p_{\boldsymbol{\theta}^{(s)}}(y_i|\mathfrak{q}) \prod_{j=1}^{d} p_{\boldsymbol{\alpha}_j^{(s)}}(\mathfrak{q}_j|x_{i,j})}{\sum_{\mathfrak{q}' \in \mathfrak{Q}_{\boldsymbol{m}}}} p_{\boldsymbol{\theta}^{(s)}}(y_i|\mathfrak{q}') \prod_{j=1}^{d} p_{\boldsymbol{\alpha}_j^{(s)}}(\mathfrak{q}_j'|x_{i,j})$$

Classically, the EM can be replaced by the Stochastic Expectation Maximization [7] algorithm: the expectation (the sum over $\mathfrak{q} \in \mathfrak{Q}_{\boldsymbol{m}}$) is approximated by the empirical distribution (up to a normalization constant) of draws $\mathfrak{q}^{(1)}, \ldots, \mathfrak{q}^{(S)}$ from $p_{\boldsymbol{\theta}}(y|\cdot) \prod_{j=1}^{d} p_{\boldsymbol{\alpha}_j}(\cdot|x_j)$.

SEM as a quantization provider

As the parameters $\boldsymbol{\alpha}$ of $\boldsymbol{q}_{\boldsymbol{\alpha}}$ were initialized randomly in the neural network approach, the latent features observations $\mathbf{q}^{(0)}$ are initialized uniformly (*i.e.* by sampling from an equiprobable multinouilli distribution). At step s, the SEM algorithm allows us to draw $\mathbf{q}^{(s)}$. As the logistic regression $p_{\boldsymbol{\theta}}(y|\mathfrak{q})$ is multivariate, it is hard to sample simultaneously all latent features. We have to resort to the Gibbs-sampler [6]: $\mathfrak{q}_j$ is sampled while holding latent features $\mathfrak{q}_{-\{j\}} = (\mathfrak{q}_1, \ldots, \mathfrak{q}_{j-1}, \mathfrak{q}_{j+1}, \ldots, \mathfrak{q}_d)$ fixed:

S-step:

$$\mathfrak{q}_j^{(s)} \sim p_{\hat{\boldsymbol{\theta}}^{(s-1)}}(y|\mathfrak{q}_{-\{j\}}^{(s-1)}, \cdot) p_{\hat{\boldsymbol{\alpha}}_j^{(s-1)}}(\cdot|x_j). \tag{3.17}$$

This process is repeated for all features $1 \leq j \leq d$.

Using these latent features, we can compute the MLE $\boldsymbol{\theta}^{(s)}$ (resp. $\boldsymbol{\alpha}^{(s)}$) of $\boldsymbol{\theta}$ (resp. $\boldsymbol{\alpha}$) given $\mathbf{q}^{(s)}$ by maximizing the following likelihoods (M-steps):

M-steps:

$$\begin{aligned}
\boldsymbol{\theta}^{(s)} &= \operatorname*{argmax}_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathbf{q}^{(s)}, \mathbf{y}_{\mathrm{f}}) \quad = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{n} \ln p_{\boldsymbol{\theta}}(y_i|\mathfrak{q}_i^{(s)}), \\
{\boldsymbol{\alpha}_j}^{(s)} &= \operatorname*{argmax}_{\boldsymbol{\alpha}_j} \ell(\boldsymbol{\alpha}_j; \mathbf{x}_{\mathrm{f},j}, \mathbf{q}_j^{(s)}) = \operatorname*{argmax}_{\boldsymbol{\alpha}_j} \sum_{i=1}^{n} \ln p_{\boldsymbol{\alpha}_j}(\mathfrak{q}_{i,j}^{(s)}|x_{i,j}) \text{ for } 1 \leq j \leq d.
\end{aligned} \tag{3.18}$$

Remark on their optimization algorithms The MLE of $\boldsymbol{\theta}$ is obtained, as in the preceding sections and as was thoroughly discussed in Chapter 1, *via* Newton-Raphson. For polytomous logistic regression, the same optimization procedure can be used. For categorical features, the MLE is simply the proportion of observations in each category:

$$\hat{\boldsymbol{\alpha}}_{j,h}^{o} = \frac{|\mathbf{q}_{j,h}|}{|\{x_j = o\}|} \text{ for } 1 \leq o \leq l_j.$$

This SEM provides parameters $\boldsymbol{\alpha}^{(1)}, \ldots, \boldsymbol{\alpha}^{(S)}$ which can be used to produce $\hat{\boldsymbol{q}}^{(1)}, \ldots, \hat{\boldsymbol{q}}^{(S)}$ following the *maximum a posteriori* scheme from Equation (3.9), adapted to this new formulation:

$$\hat{\boldsymbol{q}}_j^{(s)}(\cdot) = \operatorname*{argmax}_h p_{\boldsymbol{\alpha}_j^{(s)}}(\boldsymbol{e}_h^{m_j}|\cdot).$$

The logistic regression parameters $\hat{\boldsymbol{\theta}}^{(s)}$ on quantized data are obtained similarly as in (3.11). The best proposed quantization $\hat{\boldsymbol{q}}^{(s^\star)}$ is thus chosen among them *via e.g.* the BIC criterion as in Equation (3.12).

Validity of the approach

The pseudo-completed sample $(\mathbf{x}_f, \mathbf{q}^{(s)}, \mathbf{y}_f)$ allows to compute $(\boldsymbol{\theta}^{(s)}, \boldsymbol{\alpha}^{(s)})$ which does not converge to the MLE of $p(\mathbf{y}|\mathbf{x};\boldsymbol{\theta},\boldsymbol{\alpha})$, for the simple reason that, being random in essence, it does not converge pointwise. From its authors, the SEM is however expected to be directed by the EM dynamics [8] and its empirical distribution converges to the target distribution $p(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}^\star,\boldsymbol{\alpha}^\star)$ provided such a distribution exists and is unique. This existence is guaranteed by remarking that for all features j, $p(\mathfrak{q}_j|x_j,\mathfrak{q}^{(s)}_{-\{j\}},y;\boldsymbol{\theta},\boldsymbol{\alpha}) \propto p_{\boldsymbol{\theta}}(y|\mathfrak{q}^{(s)}_{-\{j\}},\mathfrak{q}_j)p_{\boldsymbol{\alpha}_j}(\mathfrak{q}_j|x_j) > 0$ by definition of the logistic regression and polytomous logistic regressions or the contingency tables respectively. The uniqueness is not guaranteed since levels can disappear and there is an absorbing state (the empty model): this point is detailed in the next section.

In its original purpose [8], the SEM was employed either to find good starting points for the EM (*e.g.* to avoid local maxima) or to propose an estimator of the MLE of the target distribution as the mean or the mode of the resulting empirical distribution, eventually after a burn-in phase. However, in our setting, we are not directly interested in the MLE but only to the best quantization in the sense of Equation (3.5). The best proposed quantization $\boldsymbol{q}^\star$ is thus chosen among them *via* the BIC criterion as in Equation (3.12).

Choosing an appropriate number of levels

Contrary to the neural network approach developed in Section 3.4, the SEM algorithm alternates between drawing $\mathbf{q}^{(s)}$ and fitting $\boldsymbol{\theta}^{(s)}$ and $\boldsymbol{\alpha}^{(s)}$ at each step s. Therefore, additionally to the phenomenon of "vanishing" levels caused by the *maximum a posteriori* procedure similar to the neural network approach, if a level h of $\mathfrak{q}$ is not drawn, following Equation (3.17), at step s, then at step $s+1$ when adjusting parameters $\boldsymbol{\alpha}_j$ by maximum likelihood from Equation (3.18), this level will have disappeared and cannot be drawn again. A Reversible-Jump MCMC approach would be needed [17] to "resuscitate" these levels, which is not needed in the neural network approach because its architecture is fixed in advance. As a consequence, with a design matrix of fixed size n, there is a non-zero probability that for any given feature, any of its levels collapses at each step such that $m_j^{(s+1)} = m_j^{(s)} - 1$.

The MCMC has thus an absorbing state for which all features are quantized into one level (the empty model with no features) which is reached in a finite number of steps (although very high if n is sufficiently large as is the case with *Credit Scoring* data). The SEM algorithm is an effective way to start from a high number of levels per feature $\boldsymbol{m}_{\max}$ and explore smaller values.

The full algorithm is described in Appendix A.2.3 and schematically in Figure 3.13.

3.6 Numerical experiments

This section is divided into three complementary parts to assess the validity of our proposal, that is called hereafter *glmdisc*-NN and *glmdisc*-SEM, designating respectively the approaches developed in Sections 3.4 and 3.5. First, simulated data are used to evaluate its ability to recover the true data generating mechanism. Second, the predictive quality of the new learned representation approach is illustrated on several classical benchmark datasets from the UCI library. Third, we use it on *Credit Scoring* datasets provided by CACF. The code of all experiments, excluding the confidential real data, can be retrieved following the guidelines in Appendix B.

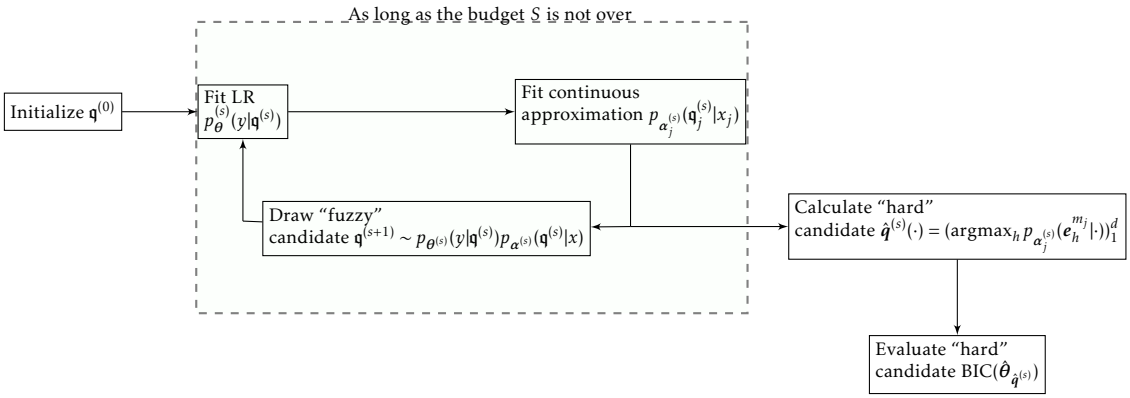


Figure 3.13 – Schema of the SEM quantization approach.

3.6.1 Simulated data: empirical consistency and robustness

Focus is here given on discretization of continuous features (similar experiments could be conducted on categorical ones). Two continuous features x_1 and x_2 are sampled from the uniform distribution on $[0,1]$ and discretized as exemplified on Figure 3.14 by using

$$\boldsymbol{q}_1(\cdot) = \boldsymbol{q}_2(\cdot) = (\mathbb{1}_{]-\infty,1/3]}(\cdot), \mathbb{1}_{]1/3,2/3]}(\cdot), \mathbb{1}_{]2/3,\infty[}(\cdot)).$$

Here, following (3.2), we have $d = 2$ and $m_1 = m_2 = 3$ and the cutpoints are $c_{j,1} = 1/3$ and $c_{j,2} = 2/3$ for $j = 1, 2$. Setting $\boldsymbol{\theta} = (0, -2, 2, 0, -2, 2, 0)$, the target feature y is then sampled from $p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{q}(\mathbf{x}))$ via the logistic model (3.3).

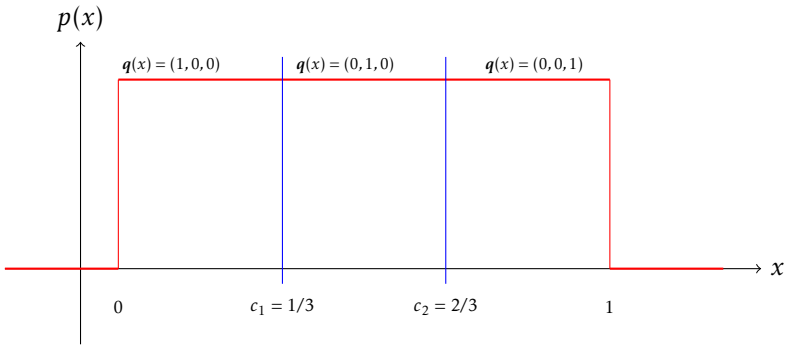


Figure 3.14 – Pdf of the simulated continuous data x and the true quantization $\boldsymbol{q}$.

From the *glmdisc* algorithm, we studied three cases:

(a) First, the quality of the cutoff estimator $\hat{c}_{j,2}$ of $c_{j,2} = 2/3$ is assessed when the starting maximum number of intervals per discretized continuous feature is set to its true value $m_1 = m_2 = 3$;

(b) Second, we estimated the number of intervals $\hat{m}_1$ of $m_1 = 3$ when the starting maximum number of intervals per discretized continuous feature is set to $m_{\max} = 10$;

(c) Last, we added a third feature x_3 also drawn uniformly on $[0,1]$ but uncorrelated to y and estimated the number $\hat{m}_3$ of discretization intervals selected for x_3. The reason is that a

non-predictive feature which is discretized or grouped into a single value is *de facto* excluded from the model, and this is a positive side effect.

From a statistical point of view, experiment (a) assesses the empirical consistency of the estimation of $C_{j,h}$, whereas experiments (b) and (c) focus on the consistency of the estimation of m_j. The results are summarized in Table 3.2 where 95% confidence intervals (CI [36]) are given, with a varying sample size. Note in particular that the slight underestimation in (b) is a classical consequence of the BIC criterion on small samples.

The neural network approach *glmdisc*-NN seems to outperform the SEM approach on experiments (b) and (c) where the true number of levels $\boldsymbol{m}$ has to be estimated. This is rather surprising since theoretically, the SEM approach can explore the model space easier than *glmdisc*-SEM thanks to the additional "disappearing" effect of the drawing procedure (the absorbing state of the MCMC: see paragraph "Choosing an appropriate number of levels" in Section 3.5.3). This inferior performance is somewhat confirmed with real data in the subsequent sections. A rough guess about this performance drop with an equivalent computational budget S is the "noise" introduced by drawing $\mathfrak{q}$ rather than maximizing directly the log-likelihood $\ell_{\boldsymbol{q},\boldsymbol{\alpha}}$ (Equation (3.8)) which can be achieved by *glmdisc*-NN through gradient descent. Therefore, *glmdisc*-SEM might need way more iterations than *glmdisc*-NN to converge, especially in a misspecified model setting.

Table 3.2 – For *glmdisc*-NN and *glmdisc*-SEM and different sample sizes n, (a) CI of $\hat{c}_{j,2}$ for $c_{j,2} = 2/3$. (b) Bar plot of $\hat{m} = 2, 3, 4$ (resp.) for $m_1 = 3$. (c) Bar plot of $\hat{m}_3 = 1, 2, 3$ (resp.) for $m_3 = 1$ with a computational budget $S = 500$ iterations.

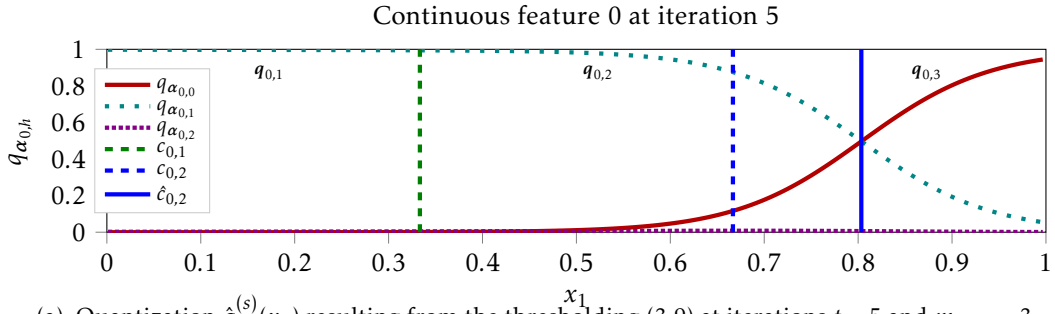
Algorithm	n	(a) $\hat{c}_{j,2}$	(b)	$\hat{m}_1$	(c)	$\hat{m}_3$
glmdisc-NN	1,000	$[0.656, 0.666]$	9 90 1		60 32 8	
glmdisc-SEM	1,000	$[0.664, 0.669]$	2 53 44		34 56 10	
glmdisc-NN	10,000	$[0.666, 0.666]$	0 100 0		88 12 0	
glmdisc-SEM	10,000	$[0.666, 0.666]$	2 69 30		30 48 22	

To complement these experiments on simulated data following a well-specified model, a similar study can be done for categorical features: 10 levels are drawn uniformly and 3 groups of levels, which share the same log-odd ratio, are created. The same phenomenon as in Table 3.2 is witnessed: the empirical distribution of the estimated number of groups of levels is peaked at its true value of 3.

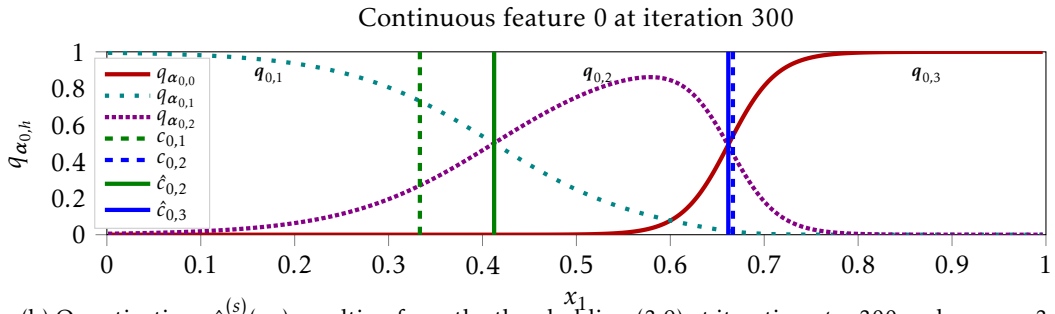
Finally, it was argued in Section 3.3 that by considering all features when quantizing the data, relying on a multivariate approach could yield better results than classical univariate techniques in presence of correlation. This claim is verified in Table 3.3 where multivariate heteroskedastic Gaussian data is simulated on which the log odd ratio of y depends linearly (misspecified model setting for the quantized logistic regression). The proposed SEM approach yields significantly better results then ChiMerge and MDLP.

Table 3.3 – Gini of the resulting misspecified logistic regression from quantized data using ChiMerge, MDLP and *glmdisc*-SEM: the multivariate approach is able to capture information about the correlation structure.

	ChiMerge	MDLP	*glmdisc*-SEM
Performance	50.1 (1.6)	77.1 (0.9)	**80.6** (0.6)



(a) Quantization $\hat{\boldsymbol{q}}_1^{(s)}(x_1)$ resulting from the thresholding (3.9) at iterations $t = 5$ and $m_{\max} = 3$.



(b) Quantizations $\hat{\boldsymbol{q}}_1^{(s)}(x_1)$ resulting from the thresholding (3.9) at iterations $t = 300$ and $m_{\max} = 3$.

Figure 3.15 – Quantizations $\hat{\boldsymbol{q}}_1^{(s)}(x_1)$ of experiment (a) resulting from the thresholding (3.9).

3.6.2 Benchmark data

To test further the effectiveness of *glmdisc* in a predictive setting, we gathered 6 datasets from the UCI library: the Adult dataset ($n = 48,842$, $d = 14$), the Australian dataset ($n = 690$, $d = 14$), the Bands dataset ($n = 512$, $d = 39$), the Credit-screening dataset ($n = 690$, $d = 15$), the German dataset ($n = 1,000$, $d = 20$) and the Heart dataset ($n = 270$, $d = 13$). Each of these datasets have mixed (continuous and categorical) features and a binary response to predict. To get more information about these datasets, their respective features, and the predictive task associated with them, readers may refer to the UCI website [3].

Now that the proposed approach was shown empirically consistent, *i.e.* it is able to find the true quantization in a well-specified setting, it is desirable to verify the previous claim that embedding the learning of a good quantization in the predictive task *via glmdisc* is better than other methods that rely on *ad hoc* criteria. As we were primarily interested in logistic regression, I will compare the proposed approach to a naïve linear logistic regression (hereafter ALLR), *i.e.* on non-quantized data, a logistic regression on continuous discretized data using the now standard MDLP algorithm from [16] and categorical grouped data using χ^2 tests of independence between

3. [13] : http://archive.ics.uci.edu/ml

Table 3.4 – Gini indices (the greater the value, the better the performance) of our proposed quantization algorithm *glmdisc* and two baselines: ALLR and MDLP / χ^2 tests obtained on several benchmark datasets from the UCI library with a single run and a computational budget $S = 500$ iterations.

Dataset	ALLR	*ad hoc* methods	Our proposal: *glmdisc*-NN	Our proposal: *glmdisc*-SEM
Adult	81.4 (1.0)	**85.3** (0.9)	80.4 (1.0)	81.5 (1.0)
Australian	72.1 (10.4)	84.1 (7.5)	92.5 (4.5)	**100** (0)
Bands	48.3 (17.8)	47.3 (17.6)	58.5 (12.0)	**58.7** (12.0)
Credit	81.3 (9.6)	88.7 (6.4)	**92.0** (4.7)	87.7 (6.4)
German	52.0 (11.3)	54.6 (11.2)	**69.2** (9.1)	54.5 (10)
Heart	80.3 (12.1)	78.7 (13.1)	**86.3** (10.6)	82.2 (11.2)

each pair of factor levels and the target in the same fashion as the ChiMerge discretization algorithm proposed by [22] (hereafter MDLP/χ^2). In this section and the next, Gini indices are reported on a random 30 % test set and CIs are given following a method found in [36]. Table 3.4 shows our approach yields significantly better results on these rather small datasets where the added flexibility of quantization might help the predictive task.

As argued in the preceding section, *glmdisc*-SEM yields slightly worse results than *glmdisc*-NN which, additionally to their inherent difference (the S-step of the SEM), might be due to the sensitivity of the SEM to its starting points (a single Markov Chain is run in this section and the next).

3.6.3 *Credit Scoring* data

Discretization, grouping and interaction screening are preprocessing steps relatively "manually" performed in the field of *Credit Scoring*, using χ^2 tests for each feature or so-called Weights of Evidence ([44]). This back and forth process takes a lot of time and effort and provides no particular statistical guarantee.

Table 3.5 shows Gini coefficients of several portfolios for which there are $n = 50,000$, $n = 30,000$, $n = 50,000$, $n = 100,000$, $n = 235,000$ and $n = 7,500$ clients respectively and $d = 25$, $d = 16$, $d = 15$, $d = 14$, $d = 14$ and $d = 16$ features respectively. Approximately half of these features were categorical, with a number of factor levels ranging from 2 to 100. All portfolios come from approximately one year of financed clients. The Automobile dataset is composed of car loans (and thus we have data about the cars, such as the brand, the cost, the motor, *etc* which generally boosts predictive performance), the Renovation, Mass retail and Electronics datasets are composed of standard loans through partners that respectively sell construction material (to private persons, not companies), retail products (*i.e.* supermarkets), and electronics goods (smartphones, TVs, *etc*). The Standard and Revolving datasets are clients coming directly to Sofinco (CACF's main brand) through the phone or the web.

We compare the rather manual, in-house approach that yields the current performance, the naïve linear logistic regression and *ad hoc* methods introduced in the previous section and finally our *glmdisc* proposal. Beside the classification performance, interpretability is maintained and unsurprisingly, the learned representation comes often close to the "manual" approach: for example, the complicated in-house coding of job types is roughly grouped by *glmdisc* into *e.g.* "worker", "technician", *etc.* Notice that even if the "naïve" logistic regression reaches some very decent predictive results, its poor interpretability skill (no quantization at all) excludes it from standard use in the company.

Table 3.5 – Gini indices (the greater the value, the better the performance) of our proposed quantization algorithm *glmdisc*, the two baselines of Table 3.4 and the current scorecard (manual / expert representation) obtained on several portfolios of Crédit Agricole Consumer Finance with a single run and a computational budget $S = 500$ iterations.

Portfolio	ALLR	Current performance	*ad hoc* methods	Our proposal: *glmdisc*-NN	Our proposal: *glmdisc*-SEM
Automobile	**59.3 (3.1)**	55.6 (3.4)	**59.3 (3.0)**	58.9 (2.6)	57.8 (2.9)
Renovation	52.3 (5.5)	50.9 (5.6)	54.0 (5.1)	**56.7 (4.8)**	55.5 (5.2)
Standard	39.7 (3.3)	37.1 (3.8)	**45.3 (3.1)**	43.8 (3.2)	36.7 (3.7)
Revolving	62.7 (2.8)	58.5 (3.2)	**63.2 (2.8)**	62.3 (2.8)	60.7 (2.8)
Mass retail	52.8 (5.3)	48.7 (6.0)	61.4 (4.7)	**61.8 (4.6)**	61.0 (4.7)
Electronics	52.9 (11.9)	55.8 (10.8)	56.3 (10.2)	**72.6 (7.4)**	62.0 (9.5)

Our approach shows approximately similar results than MDLP/χ^2, potentially due to the fact that contrary to the two previous experiments with simulated or UCI data, the classes are imbalanced (< 3% defaulting loans), which would require special treatment while back-propagating the gradients [2]. Note however that it is never significantly worse; for the Electronics dataset and as was the case for most UCI datasets, *glmdisc* is significantly superior, which in the *Credit Scoring* business might end up saving millions to the financial institution.

Table 3.6 is somewhat similar but is an earlier work (*glmdisc*-NN was not implemented at that time): no CI is reported, only continuous features are considered so that pure discretization methods can be compared, namely MDLP and ChiMerge. Three portfolios are used with approx. 10 features and $n = 180{,}000$, $n = 30{,}000$, and $n = 100{,}000$ respectively. The Automobile 2 dataset is again a car loan dataset with information on the cars, the Young clients datasets features only clients less than 30 years old (that are difficult to address in the industry: poor credit history or stability), and the Basel II dataset is a small portion of known clients for which we'd like to provision the expected losses (regulation obligation per the Basel II requirements). The proposed algorithm *glmdisc*-SEM performs best, but is rather similar to the achieved performance of MDLP. ChiMerge does poorly since its parameter α (the rejection zone of the χ^2 tests) is not optimized which is blatant on Portfolio 3 where approx. 2,000 intervals are created, so that predictions are very "noisy".

The usefulness of discretization and grouping is clear on *Credit Scoring* data and although *glmdisc* does not always perform significantly better than the manual approach, it allows practitioners to focus on other tasks by saving a lot of time, as was already stressed out. As a rule of thumb, a month is generally allocated to data pre-processing for a single data scientist working on a single scorecard. On Google Collaboratory, and relying on Keras ([11]) and Tensorflow ([27]) as a backend, it took less than an hour to perform discretization and grouping for all datasets. As for the *glmdisc*-SEM method, quantization of datasets of approx. $n = 10{,}000$ observations and approx. $d = 10$ take about 2 hours on a laptop within a single CPU core. On such a small rig, $n = 100{,}000$ observations and trying to perform interaction screening becomes however prohibitive (approx. 3 days). However, using higher computing power aside, there is still room for improvement, *e.g.* parallel computing, replacing bottleneck functions with C++ code, etc. Moreover, the ChiMerge and MDLP methods implemented in the R package discretization are not much faster while showing inferior performance and being capable of only discretization on non-missing values.

Table 3.6 – Gini indices for three other portfolios of Crédit Agricole Consumer Finance involving only continuous features and following three methods: ChiMerge, MDLP and *glmdisc*-SEM compared to the current performance.

Portfolio	Current performance	ChiMerge	MDLP	Our proposal: *glmdisc*-SEM
Automobile 2	57.5	16.5	**58.0**	**58.0**
Young clients	27.0	26.7	29.2	**30.0**
Basel II	70.0	0	**71.3**	**71.3**

3.7 Concluding remarks

3.7.1 Handling missing data

For categorical features, handling missing data is straightforward: the level "missing" is simply considered as a separate level, that can eventually be merged in the proposed algorithm with any other level. If it is MNAR (*e.g.* co-borrower information missing because there is none) and such clients are significantly different from other clients in terms of creditworthiness, then such a treatment makes sense. If it is MAR and *e.g.* highly correlated with some of the feature's levels (for example, the feature "number of children" could be either 0 or missing to mean the borrower has no child), the proposed algorithm is highly likely to group these levels.

For continuous features, the same strategy can be employed: they can be encoded as "missing" and considered a separate level. However, this prevents this level to be merged with another one by having *e.g.* a level "[0; 200] or missing".

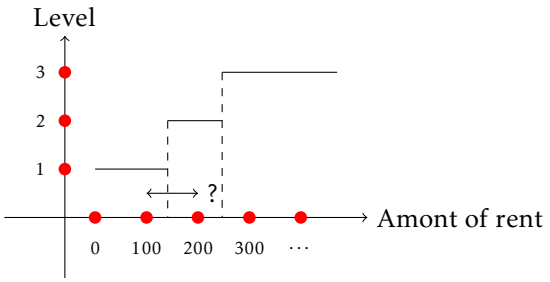
3.7.2 Integrating constraints on the cut-points

Another problem that CACF faces is to have interpretable cutpoints, i.e. having discretization intervals of the form [0; 200] and not [0.389; 211.2] which are arguably less interpretable. But it is also highly subjective, and it would require the addition of an hyperparameter, namely the set of admissible discretization and / or the rounding to perform for each feature j such that we did not pursue this problem. For the record, it is interesting to note that a straightforward rounding might not work: in the optimization community, it is well known that integer problems require special algorithmic treatment (dubbed integer programming). As an undergraduate, I applied some of these techniques to financial data in [15] where I give a counterexample. Additionally, forcing estimated cutpoints to fall into a constrained set might drastically change predictive performance if levels collapse as on Figure 3.16.

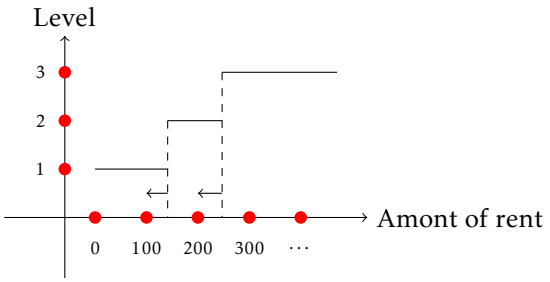
3.7.3 Wrapping up

Feature quantization (discretization for continuous features, grouping of factor levels for categorical ones) in a supervised multivariate classification setting is a recurring problem in many industrial contexts. This setting was formalized as a highly combinatorial representation learning problem and a new algorithmic approach, named *glmdisc*, has been proposed as a sensible approximation of a classical statistical information criterion.

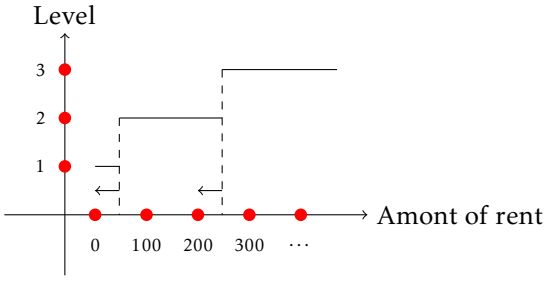
The first proposed implementation relies on the use of a neural network of particular architecture and specifically a softmax approximation of each discretized or grouped feature. The second proposed implementation relies on an SEM algorithm and a polytomic multiclass logistic regression approximation in the same flavor as the softmax. These proposals can alternatively



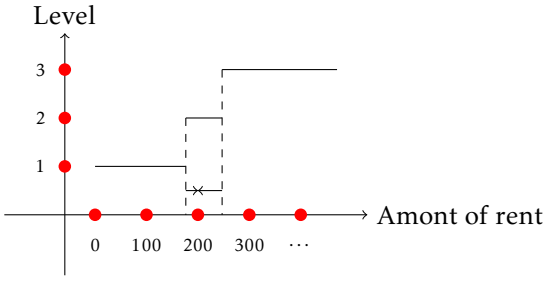
(a) If the admissible quantizations are those displayed in red, and the estimated cutpoints are the dashed lines, which "rounding" shall be chosen?



(b) In this setting, rounding has no side effect.



(c) The first level is "collapsed" and disappears, with possible consequences to the predictive task.



(d) The narrow middle level is collapsed, also with possible predictive consequences, since *e.g.* rent amounts are often very concentrated around their mean.

Figure 3.16 – Different settings of estimated quantizations and the consequences of constraints on the set of admissible cutpoints.

be replaced by any other univariate multiclass predictive model, which make them flexible and adaptable to other problems. Prediction of the target feature, given quantized features, was exemplified with logistic regression, although here as well, it can be swapped with any other supervised classification model.

Although both estimation methods are arguably much alike, since they rely on the same continuous approximation, results differed sensibly. Indeed, the SEM approach necessitated a sampling step, whereas the neural network , which has nevertheless the clear advantage of exploring . Additionally, the neural network approach relies on standard deep learning libraries, highly parallelizable and , which can make it way faster than the SEM approach that cannot be parallelized straightforwardly .

The experiments showed that, as was sensed empirically by statisticians in the field of *Credit Scoring*, discretization and grouping can indeed provide better models than standard logistic regression. This novel approach allows practitioners to have a fully automated and statistically well-grounded tool that achieves better performance than *ad hoc* industrial practices at the price of decent computing time but much less of the practitioner's valuable time.

As described in the introduction, logistic regression is additive in its inputs which does not allow to take into account conditional dependency, as stated by [4]. This problem is often dealt with by sparsely introducing "interactions", *i.e.* products of two features. This leads again to a model selection challenge on a highly combinatorial discrete space that could be solved with a

similar approach. In a broader context with no restriction on the predictive model, [39] already made use of neural networks to estimate the presence or absence of statistical interactions. We take another approach in the subsequent chapter where we tackle the parsimonious addition of pairwise interactions among quantized features, that might influence the quantization process introduced in this chapter.

References of Chapter 3

[1] Hirotugu Akaike. « Information theory and an extension of the maximum likelihood principle ». In: *2nd International Symposium on Information Theory, 1973*. Akademiai Kiado. 1973, pp. 267–281.

[2] Rangachari Anand et al. « An improved algorithm for neural network classification of imbalanced training sets ». In: *IEEE Transactions on Neural Networks* 4.6 (1993), pp. 962–969.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. « Representation learning: A review and new perspectives ». In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828.

[4] William D Berry, Jacqueline HR DeMeritt, and Justin Esarey. « Testing for interaction in binary logit and probit models: Is a product term essential? » In: *American Journal of Political Science* 54.1 (2010), pp. 248–266.

[5] Léon Bottou. « Large-scale machine learning with stochastic gradient descent ». In: *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[6] George Casella and Edward I George. « Explaining the Gibbs sampler ». In: *The American Statistician* 46.3 (1992), pp. 167–174.

[7] Gilles Celeux. « The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem ». In: *Computational statistics quarterly* 2 (1985), pp. 73–82.

[8] Gilles Celeux, Didier Chauveau, and Jean Diebolt. *On Stochastic Versions of the EM Algorithm*. Research Report RR-2514. INRIA, 1995. URL: https://hal.inria.fr/inria-00074164.

[9] Faicel Chamroukhi et al. « A regression model with a hidden logistic process for feature extraction from time series ». In: *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*. IEEE. 2009, pp. 489–496.

[10] Jiahua Chen and Zehua Chen. « Extended Bayesian information criteria for model selection with large model spaces ». In: *Biometrika* 95.3 (2008), pp. 759–771.

[11] François Chollet et al. *Keras*. https://keras.io. 2015.

[12] Arthur P Dempster, Nan M Laird, and Donald B Rubin. « Maximum likelihood from incomplete data via the EM algorithm ». In: *Journal of the royal statistical society. Series B (methodological)* (1977), pp. 1–38.

[13] Dua Dheeru and Efi Karra Taniskidou. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml.

[14] James Dougherty, Ron Kohavi, and Mehran Sahami. « Supervised and unsupervised discretization of continuous features ». In: *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 194–202.

[15] Adrien Ehrhardt. « Projet recherche : optimisation non-linéaire - application à la création d'indices boursiers ». MA thesis. École Centrale de Lille, 2014.

[16] Usama Fayyad and Keki Irani. « Multi-interval discretization of continuous-valued attributes for classification learning ». In: *13th International Joint Conference on Artificial Intelligence*. 1993, pp. 1022–1029.

[17] Peter J Green. « Reversible jump Markov chain Monte Carlo computation and Bayesian model determination ». In: *Biometrika* 82.4 (1995), pp. 711–732.

[18] KM Ho and Paul D Scott. « An efficient global discretization method ». In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 1998, pp. 383–384.

[19] KM Ho and PD Scott. « Zeta: a global method for discretization of cotitinuous variables ». In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*. 1997, pp. 191–194.

[20] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*. Vol. 398. John Wiley & Sons, 2013.

[21] Gordon V Kass. « An exploratory technique for investigating large quantities of categorical data ». In: *Applied statistics* (1980), pp. 119–127.

[22] Randy Kerber. « Chimerge: Discretization of numeric attributes ». In: *Proceedings of the tenth national conference on Artificial intelligence*. Aaai Press. 1992, pp. 123–128.

[23] Emilie Lebarbier and Tristan Mary-Huard. « Le critère BIC : fondements théoriques et interprétation ». In: RR-5315 (2004), p. 17. URL: https://hal.inria.fr/inria-00070685.

[24] Huan Liu and Rudy Setiono. « Chi2: Feature selection and discretization of numeric attributes ». In: *Tools with artificial intelligence, 1995. proceedings., seventh international conference on*. IEEE. 1995, pp. 388–391.

[25] Huan Liu et al. « Discretization: An enabling technique ». In: *Data mining and knowledge discovery* 6.4 (2002), pp. 393–423.

[26] Aleksandra Maj-Kańska, Piotr Pokarowski, Agnieszka Prochenka, et al. « Delete or merge regressors for linear model selection ». In: *Electronic Journal of Statistics* 9.2 (2015), pp. 1749–1778.

[27] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[28] Lukas Meier, Sara Van De Geer, and Peter Bühlmann. « The group lasso for logistic regression ». In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70.1 (2008), pp. 53–71.

[29] Ann A O'Connell. *Logistic regression models for ordinal response variables*. 146. Sage, 2006.

[30] Sergio Ramírez-Gallego et al. « Data discretization: taxonomy and big data challenge ». In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 6.1 (2016), pp. 5–21.

[31] Paul Reverdy and Naomi Ehrich Leonard. « Parameter estimation in softmax decision-making models with linear objective functions ». In: *IEEE Transactions on Automation Science and Engineering* 13.1 (2016), pp. 54–67.

[32] Allou Samé et al. « Model-based clustering and segmentation of time series with changes in regime ». In: *Advances in Data Analysis and Classification* 5.4 (2011), pp. 301–321.

[33] Steven G. Self and Kung-Yee Liang. « Asymptotic Properties of Maximum Likelihood Estimators and Likelihood Ratio Tests Under Nonstandard Conditions ». In: *Journal of the American Statistical Association* 82.398 (1987), pp. 605–610. ISSN: 01621459. URL: http://www.jstor.org/stable/2289471.

[34] Juliet Popper Shaffer. « Multiple hypothesis testing ». In: *Annual review of psychology* 46.1 (1995), pp. 561–584.

[35] Chao-Ton Su and Jyh-Hwa Hsu. « An extended chi2 algorithm for discretization of real value attributes ». In: *IEEE transactions on knowledge and data engineering* 17.3 (2005), pp. 437–441.

[36] Xu Sun and Weichao Xu. « Fast Implementation of DeLong's Algorithm for Comparing the Areas Under Correlated Receiver Oerating Characteristic Curves ». In: *IEEE Signal Processing Letters* 21.11 (2014), pp. 1389–1393.

[37] Francis EH Tay and Lixiang Shen. « A modified chi2 algorithm for discretization ». In: *IEEE Transactions on Knowledge & Data Engineering* 3 (2002), pp. 666–670.

[38] Robert Tibshirani et al. « Sparsity and smoothness via the fused lasso ». In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.1 (2005), pp. 91–108.

[39] Michael Tsang, Dehua Cheng, and Yan Liu. « Detecting Statistical Interactions from Neural Network Weights ». In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=ByOfBggRZ.

[40] Vincent Vandewalle. « How to Take into Account the Discrete Parameters in the BIC Criterion? » In: *COMPStat*. 2010.

[41] Cédric Villani et al. *Donner un sens à l'intelligence artificielle: pour une stratégie nationale et européenne*. Conseil national du numérique, 2018.

[42] Ke Wang and Bing Liu. « Concurrent discretization of multiple attributes ». In: *Pacific Rim International Conference on Artificial Intelligence*. Springer. 1998, pp. 250–259.

[43] Ying Yang and Geoffrey I Webb. « Discretization for naive-Bayes learning: managing discretization bias and variance ». In: *Machine learning* 74.1 (2009), pp. 39–74.

[44] Guoping Zeng. « A necessary condition for a good binning algorithm in credit scoring ». In: *Applied Mathematical Sciences* 8.65 (2014), pp. 3229–3242.

Chapter 4

Interaction discovery for logistic regression

> A little rudeness and disrespect can elevate a meaningless interaction to a battle of wills and add drama to an otherwise dull day.
>
> Bill Watterson, "The Complete Calvin and Hobbes".

Sommaire

Continuing our pursuit of interpretable representation learning algorithms for logistic regression, we tackle in this chapter a common problem in *Credit Scoring* and other application contexts relying either on logistic regression or additive models of the form $f(y) = \sum_{j=1}^{d} \boldsymbol{q}_j(x_j)'\boldsymbol{\theta}_j$. To further reduce the model bias discussed in Section 1.3 and thus obtain better predictive performance while maintaining interpretability, *Credit Scoring* practitioners are used to introducing pairwise interactions.

4.1 Motivation: XOR function

As described in the introduction, logistic regression is linear in its inputs which does not allow to take into account conditional dependency: the change of slope of a feature's log-odds given another (moderator) feature (see [1]). This problem is often dealt with by sparsely introducing "interactions", *i.e.* products of two features. Unfortunately, this leads again to a model selection challenge as the number of pairs of features is $\frac{d(d-1)}{2}$. We denote by $\boldsymbol{\delta}$ the triangular inferior matrix with $\delta_{k,\ell} = 1$ if $k < \ell$ and features k and ℓ "interact" in the logistic regression in the sense of [1]. The logistic regression with interactions $\boldsymbol{\delta}$ is thus:

$$\text{logit}[p_{\boldsymbol{\theta}}(1|\boldsymbol{q}(\boldsymbol{x}), \boldsymbol{\delta})] = \theta_0 + \sum_{j=1}^{d} \boldsymbol{q}_j(x_j)'\boldsymbol{\theta}_j + \sum_{1\leq k<\ell\leq d} \delta_{k,\ell}\boldsymbol{q}_k(x_k)'\boldsymbol{\theta}_{k,\ell}\boldsymbol{q}_\ell(x_\ell), \tag{4.1}$$

where $\boldsymbol{\theta}_j = (\theta_j^1, \ldots, \theta_j^{m_j})$ as in the previous chapter, $\boldsymbol{\theta}_{k,\ell} = (\theta_{k,\ell}^{r,t})_{1\leq r\leq m_k, 1\leq t\leq m_\ell}$ and for all features j, m_j is set as the "reference" value and consequently for all j, $\theta_j^{m_j} = 0$ and for all $1 \leq k < \ell \leq d$, $\theta_{k,\ell}^{m_k,m_\ell} = 0$. The resulting coefficient $\boldsymbol{\theta}$ is the vector of all main effects $(\boldsymbol{\theta}_j)_1^d$ (as in the previous chapter) and the interaction coefficients $(\boldsymbol{\theta}_{k,\ell})_{1\leq k,\ell\leq d}$.

Since, in presence of an interaction between k and ℓ, $\boldsymbol{\theta}_k$ already encodes the log-odd ratio of feature k conditionally to ℓ being at its reference value m_ℓ, $\theta_{k,\ell}^{1,m_\ell}, \ldots, \theta_{k,\ell}^{m_k-1,m_\ell}$ are redundant and thus set to 0. Note that we could have removed the "main effect" $\boldsymbol{\theta}_k$ altogether instead (which is the classical, in-house formulation of interactions at CACF) but since we will be adding/removing interactions back and forth, the present formulation seems more adequate.

This formulation seems rather complicated visually and in terms of parameter dimension: a single interaction between two quantized features (or more broadly speaking, categorical features) amounts to adding $(m_k - 1) \cdot (m_\ell - 1)$ coefficients. Since we advocated "interpretable", *i.e.* sparse, simple models to yield scorecards as in Table 3.1, and we witnessed a high variance when estimating numerous coefficients on Figure 3.4c, it does not seem like a good idea. As a side note, interpretation of the resulting parameters can be rather tricky. The monographs [8, 7] were really helpful and are sincerely recommended to the interested reader.

Nevertheless, as thoroughly explained in [1], there are situations where interactions terms are unavoidable. A simple (but quite extreme) example is the XOR (exclusive or) function $f(x_1, x_2) = (x_1 + x_2) \cdot (2 - x_1 - x_2)$ where $x_1, x_2 \in \{0, 1\}$. Such functions cannot be learnt by a standard logistic regression. For a more illustrative example, the broad field of medicine is often interested in knowing the factors of risks of a given disease and if these factors have additive or cumulative effects (see [15] for an example), *e.g.* risk of contracting disease A is doubled with factors B and C individually, but 6 times more when both factors are present. This is precisely what is observed in the *Credit Scoring* industry: we observe higher risk among workers than executives but when associated with the time spent in the current job position, workers with "stability" of employment may appear less risky than less "stable" executives in a non-additive way.

Moreover, the number of coefficients is nevertheless kept low by having few levels, as emphasized in Chapter 3, and few interactions, as emphasized by the $\boldsymbol{\delta}$ notation. Additionally and traditionally in *Credit Scoring*, so-called "main-effects", *i.e.* features $\boldsymbol{q}_k(x_k)$ and $\boldsymbol{q}_\ell(x_\ell)$ are removed when their interaction term is present ($\delta_{k,\ell} = 1$) because as stated earlier, on categorical features, it is strictly equivalent, which is not true in the general case (mixed data). In biostatistics for example, it is usually the contrary (interactions are only considered when main effects are present), as will be seen in the following section, where a literature review is given, alongside a

reformulation of the problem.

Besides, we assume the dependence structure of Figure 4.1 such that:

$$p(\delta|\boldsymbol{q}(\boldsymbol{x})) = p(\delta), \tag{4.2}$$
$$p(\boldsymbol{q}(\boldsymbol{x})|\delta) = p(\boldsymbol{q}(\boldsymbol{x})). \tag{4.3}$$

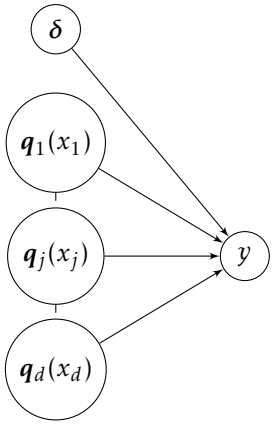


Figure 4.1 – Dependence structure between $\boldsymbol{q}_j$, $\boldsymbol{\delta}$ and y.

4.2 Pairwise interaction screening as a feature selection problem

Criterion (3.5) developed in the context of quantization can be adapted to take into account interactions:

$$(\boldsymbol{q}^\star, \boldsymbol{\delta}^\star) = \underset{\boldsymbol{q}\in\boldsymbol{Q},\delta\in\{0,1\}^{\frac{d(d-1)}{2}}}{\operatorname{argmin}} \operatorname{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q},\delta}), \tag{4.4}$$

where $\hat{\boldsymbol{\theta}}_{\boldsymbol{q},\delta}$ is the MLE of $\boldsymbol{\theta}$ given $\mathcal{T}_{\mathrm{f}}$, $\boldsymbol{q}$ and $\boldsymbol{\delta}$. Supplemental to the "Remarks on model selection consistency" of Section 3.3.4, $\delta_{k,\ell}$ is an estimated parameter of the model and an additional $\frac{d(d-1)}{2}$ term shall be added to the penalization term ν of the BIC criterion (3.5). The combinatorics involved in this problem are much higher than those of criterion (3.5), which already lead to an intractable greedy approach. For each feasible quantization scheme of Section 3.3.2's "Cardinality of the quantization family in the continuous case", there is now $2^{\frac{d(d-1)}{2}}$ models to test! In this section, we will first consider the discretization fixed and develop a stochastic approach similar to the one proposed for quantization.

With a fixed quantization scheme $\boldsymbol{q}$, criterion (4.4) amounts to $\boldsymbol{\delta}^\star = \operatorname{argmin}_{\boldsymbol{\delta}} \operatorname{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q},\boldsymbol{\delta}})$ which optimization through a greedy approach is intractable with more than a few features ($d > 10$). The first approach that seems straightforward in this setting is to simply see all $\frac{d(d-1)}{2}$ interactions as features to select from. It is worth mentioning that software to do so is available, for example in the R package *glmulti* [3] which is not restricted to interaction screening but aims at selecting features based on *e.g.* the BIC criterion. It can build all subset models of the hypothesis space or, optionnally, conduct a random search using a genetic algorithm. However, with interactions, the

search space is too vast to do such a brute force algorithm. In this potentially high-dimensional parameter space, the most computationnaly-effective approach is to resort to penalization. Various penalization approaches have been developed recently, among which LASSO [19] and its derivatives can effectively perform feature selection. A CIFRE PhD has even been dedicated to the subject with application to *Credit Scoring* [20] as was explained in Chapter 1.

The penalization approach has been applied to the interaction screening problem, very often in biostatistics-related problems, *e.g.* gene-gene interactions. Its first use [16] relies on L^2 regularization, for stability of estimation of the large number of coefficients, and forward stagewise selection of features to avoid a phenomenon where an interaction between a meaningful feature and a meaningless one would remain in the model: it would imply that the main effect coefficient of the meaningful feature could be lower, and thus preferable w.r.t. the L^2 penalty. The LASSO has been applied to this problem as well [23] by first selecting features with the LASSO based solely on main effects, then selecting pairwise interactions among the selected main effects, again with the LASSO, which requires all main effects of the considered interactions to be present in the model. Rather than conducting such a two-stage procedure, which might greatly influence the considered interactions and does not work under the "weak hierarchy" hypothesis (only one of the features of a pairwise interaction need to be present as a main effect), a set of convex constraints is added to the LASSO in [2] such that main effects and pairwise interactions are selected in a one-shot procedure with the hierarchy constraint. Other stepwise methods have been proposed, *e.g.* relying on χ^2 tests [22] or on a Reversible-Jump MCMC method [9, 17] which resembles what is proposed in this chapter but is not limited to pairwise interactions at the cost of added computational complexity. In a different setting, with much stricter hypotheses on the data, namely that they are Gaussian for each class, a statistical test involving the sample pairwise correlations is derived in [18] to characterize the probability of an interaction. Other works [5, 12] focus on this setting and propose a stepwise procedure and a dimensionality reduction technique respectively. Finally, a much simpler intuition found in [1], on which we rely in subsequent sections, is that testing for the presence of interaction in bivariate logistic regression (for all pairs of features) is roughly equivalent to the full multivariate logistic regression (in absence of correlation) but is much simpler since it amounts to construct $2^{d(d-1)/2}$ bivariate logistic regression and conduct a t-test for each interaction coefficient.

Model-free approaches have been proposed in the biostatistics literature ([24, 25, 11, 4, 26, 21] and references therein) where focus is given to large d, small n settings. Moreover, these approaches are not multivariate (*i.e.* other features than the ones involved in the pair that is tested are discarded) and are not directly applicable to a particular predictive model p_θ. Since interaction screening will be considered alongside quantization, we discard these methods from the present study.

4.3 A novel model selection approach

We take another approach here, which foremost benefit will appear in the subsequent section, and which closely resembles the strategy employed in the quantization setting of Chapter 3 and in particular the SEM approach developed in Section 3.5. The variable δ can be seen as an observation of a latent random matrix so that we will employ a stochastic approach to search for $\delta^\star$.

From the results of Chapter 3 which showed a slight advantage for *glmdisc*-NN over *glmdisc*-SEM, a natural question that arises from Table 3.5 is the adaptation of the interaction screening procedure developed in this chapter to the neural network quantization approach *glmdisc*-NN which is not straightforward: δ would be represented by nodes on the computation graph given

their dependence structure with θ (see *e.g.* Figure 3.11) which would have to be changed at each iteration. In standard deep learning frameworks, in particular TensorFlow [13], this graph is compiled and cannot be changed once gradient descent has begun.

4.3.1 Relation of the BIC criterion and the interaction probability

The BIC criterion has a desirable property, from a Bayesian perspective, relating it to the likelihood of the data given the model (in our case, a given interaction matrix δ) given the data (see [10]), where the parameter space Θ depends on the model δ:

$$\begin{aligned} p(\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f}|\delta) &= \int_\Theta p_\theta(\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f}|\delta) p(\theta|\delta) d\theta \\ &= \int_\Theta p_\theta(\mathbf{y}_\mathrm{f}|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \delta) p(\theta|\delta) p(\boldsymbol{q}(\mathbf{x}_\mathrm{f})|\delta)) d\theta \\ &= \int_\Theta p_\theta(\mathbf{y}_\mathrm{f}|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \delta) p(\theta|\delta) p(\boldsymbol{q}(\mathbf{x}_\mathrm{f})) d\theta \qquad \text{(using (4.3)).} \end{aligned}$$

Thus, we have:

$$\begin{aligned} \ln p(\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f}|\delta) &= \int_\Theta \ln p_\theta(\mathbf{y}_\mathrm{f}|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \delta) p(\theta|\delta) d\theta + \ln p(\boldsymbol{q}(\mathbf{x}_\mathrm{f})) \\ &= -\mathrm{BIC}(\hat{\theta}_\delta)/2 + \ln p(\boldsymbol{q}(\mathbf{x}_\mathrm{f})) + O(1). \end{aligned}$$

Rewriting the posterior probability of the model by introducing the preceding likelihood, we get:

$$\begin{aligned} p(\delta|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f}) &\propto p(\mathbf{y}_\mathrm{f}|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \delta) p(\delta) \\ &\propto p(\mathbf{y}_\mathrm{f}|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \delta) p(\delta|\boldsymbol{q}(\mathbf{x}_\mathrm{f})) \qquad \text{(using (4.2))} \\ &\approx \exp(-\mathrm{BIC}(\hat{\theta}_{\boldsymbol{q},\delta})/2) p(\delta). \end{aligned}$$

This last expression will be useful to design a stochastic algorithm proposing clever interaction matrices: the Metropolis-Hastings algorithm described hereafter.

4.3.2 Metropolis-Hastings sampling algorithm

This section is dedicated to describing the Metropolis-Hastings [6] sampling algorithm that will be used in the next section to sample from $p(\delta|\boldsymbol{q}(\mathbf{x}), \mathbf{y})$ that will be denoted, for simplicity, by $\pi(\delta)$ in this section.

The distribution $\pi(\delta)$ is not known explicitly but $f(\delta) = \exp(-\mathrm{BIC}(\hat{\theta}_{\boldsymbol{q},\delta})/2) p(\delta)$ is approximately proportional to $\pi(\delta)$. Consequently, given two matrices δ and δ', the pdf ratio is: $\frac{\pi(\delta)}{\pi(\delta')} \approx \frac{f(\delta)}{f(\delta')}$.

Now suppose we have at our disposal a transition kernel, that defines a probability distribution, of the form:

$$\begin{aligned} T : (\{0,1\}^{\frac{d(d-1)}{2}}, \{0,1\}^{\frac{d(d-1)}{2}}) &\to [0;1] \\ (\delta, \delta') &\mapsto T(\delta, \delta'). \end{aligned}$$

This instrumental conditional distribution will be used to design a Markov Chain which em-

pirical distribution of drawn matrices $\boldsymbol{\delta}^{(0)},\dots,\boldsymbol{\delta}^{(\text{iter})}$ approaches $\pi(\boldsymbol{\delta})$. The SEM algorithm in Section 3.5 followed the same strategy. The algorithm is the following:

Data : f, T, S
Result : $\boldsymbol{\delta}^{(0)},\dots,\boldsymbol{\delta}^{(S)}$
Initialization of $\boldsymbol{\delta}^{(0)} \sim p(\boldsymbol{\delta})$;
$s = 0$;
while $s < S$ **do**
  Draw $\boldsymbol{\delta}' \sim T(\cdot|\boldsymbol{\delta}^{(s)})$;
  Calculate the acceptance probability $A = \min\left(1, \frac{f(\boldsymbol{\delta}')}{f(\boldsymbol{\delta}^{(s)})} \frac{T(\boldsymbol{\delta}', \boldsymbol{\delta}^{(s)})}{T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}')}\right)$;
  Let $\boldsymbol{\delta}^{(s+1)} \leftarrow \begin{cases} \boldsymbol{\delta}' \text{ with probability } A, \\ \boldsymbol{\delta}^{(s)} \text{ with probability } 1-A. \end{cases}$;
end

Algorithm 1 : Metropolis-Hastings (the min function enforces $0 \leq A \leq 1$).

This algorithm reaches asymptotically the target distribution $\pi(\boldsymbol{\delta})$ if such a stationary distribution exists and is unique [14]. "Detailed balance" is a sufficient but not necessary condition of existence according to which each transition $\boldsymbol{\delta}^{(s)} \to \boldsymbol{\delta}^{(s+1)}$ is reversible. Uniqueness is guaranteed if the resulting Markov Chain is ergodic. This is satisfied if every interaction matrix $\boldsymbol{\delta}$ is aperiodic and positive recurrent (i.e. each matrix $\boldsymbol{\delta} \in \{0,1\}^{\frac{d(d-1)}{2}}$ is reachable in a finite number of iterations).

It is also important to notice that, apart from verifying the above assumptions, there are no guidelines about how to choose the proposal distribution or the number of iterations necessary for proper estimation. These are "hyperparameters" that may influence greatly the effectiveness of the method.

4.3.3 Designing a Markov Chain of good interactions

It follows from the preceding section that one can design a Metropolis-Hastings algorithm (1) which draws "good" interaction matrices $\boldsymbol{\delta}$ from the target posterior distribution $p(\boldsymbol{\delta}|\boldsymbol{q}(\mathbf{x}), \mathbf{y})$.

Transition probability

Metropolis-Hastings only requires a proposal of a transition probability between two matrices of the Markov chain that was denoted by T. This approach would require to compute $2^{d(d-1)}$ probabilities (*i.e.* one per unique couple of matrices $(\boldsymbol{\delta}, \boldsymbol{\delta}')$). It is thus desirable to reduce this combinatorics by making further assumptions. In what follows, we restrict possible transitions to matrices that are on a one unit $L^{1,1}$ distance (sum of all absolute entries) to the current interaction matrix, or equivalently which belong to the $\left(\frac{d(d-1)}{2} - 1\right)$-sphere of center $\boldsymbol{\delta}$ denoted by $S_{\boldsymbol{\delta}}^{\left(\frac{d(d-1)}{2}-1\right)}$:

$$\begin{aligned} T(\boldsymbol{\delta}, \boldsymbol{\delta}') = 0 \text{ if } & \sum_{k=1}^{d}\sum_{\ell=1}^{d} |\delta_{k,\ell} - \delta'_{k,\ell}| \neq 1 \\ & \equiv \|\boldsymbol{\delta} - \boldsymbol{\delta}'\|_{1,1} \neq 1 \\ & \equiv \boldsymbol{\delta}' \notin S_{\boldsymbol{\delta}}^{\left(\frac{d(d-1)}{2}-1\right)}. \end{aligned}$$

Only $\frac{d(d-1)}{2}$ coefficients are now needed, which can be reinterpreted as the probability to switch on (resp. off) an entry of $\boldsymbol{\delta}$ which is currently off (resp. on). We claim that a good intuition about whether two features interact is the relative gain (or loss) in BIC between their bivariate model *with* their interaction and this model *without* their interaction. The rationale behind such a procedure, relying again on the properties of BIC, is the following:

$$\begin{aligned}\forall\, 1 \leq k < \ell \leq d,\ p(\delta_{k,\ell}|\boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k}), \boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f},\ell}), \mathbf{y}_\mathrm{f}) &\propto p(\mathbf{y}_\mathrm{f}|\boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k}), \boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f},\ell}), \delta_{k,\ell}) p(\delta_{k,\ell}|\boldsymbol{q}_k(\mathbf{x}_k), \boldsymbol{q}_\ell(\mathbf{x}_\ell)) \\ &\propto p(\mathbf{y}_\mathrm{f}|\boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k}), \boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f},\ell}), \delta_{k,\ell}) p(\delta_{k,\ell}) \qquad \text{(using (4.2))} \\ &\approx \exp(-\mathrm{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q}_k, \boldsymbol{q}_\ell, \delta_{k,\ell}})/2) p(\delta_{k,\ell}).\end{aligned}$$

Setting a uniform prior which will simplify all subsequent calculations (see next section) $p(\delta_{k,\ell} = 1) = \begin{cases} 0 \text{ if } k \geq \ell \\ \frac{1}{2} \text{ otherwise} \end{cases}$ and denoting by $p_{k,\ell}$ the probability of an interaction given features $\boldsymbol{q}_k(x_k)$ and $\boldsymbol{q}_\ell(x_\ell)$:

$$p_{k,\ell} = p(\delta_{k,\ell} = 1|\boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k}), \boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f}\ell}), \mathbf{y}_\mathrm{f}) \propto\!\!\!\propto \exp\left(\frac{\mathrm{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q}_k, \boldsymbol{q}_\ell, \delta_{k,\ell}=0}) - \mathrm{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q}_k, \boldsymbol{q}_\ell, \delta_{k,\ell}=1})}{2}\right). \tag{4.5}$$

We normalize $p_{k,\ell}$ s.t. $\sum_{1 \leq k < \ell \leq d} p_{k,\ell} = 1$ and denote their triangular inferior matrix arrangement by P. Note that in this setting, we have nested models that were discussed in Paragraph Remarks on model selection consistency of Section 3.3.4 such that the approximation in (4.5) is consistent in n. We claim that if $p_{k,\ell}$ is close to 1 (resp. 0), then there is a strong chance that $\delta^\star_{k,\ell} = 1$ (resp. $\delta^\star_{k,\ell} = 0$) even in the full multivariate model, which amounts to:

$$p_{k,\ell} \approx p(\delta_{k,\ell} = 1|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f}).$$

This holds in particular if features $\boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k})$ and $\boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f},\ell})$ are independent to other features $\boldsymbol{q}_{\mathrm{f},-\{k,\ell\}}(\mathbf{x}_{\mathrm{f},-\{k,\ell\}})$: the presence or absence of their interaction in the logistic regression $p_{\boldsymbol{\theta}}(y|\boldsymbol{q}(), \boldsymbol{\delta})$ controlled by $\delta_{k,\ell}$ depends solely on $y, \boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k})$ and $\boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f},\ell})$, as formalized in the following lemma.

Lemma Our target distribution $p(\delta_{k,\ell} = 1|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f})$ is equal to our instrumental distribution $p_{k,\ell} = p(\delta_{k,\ell} = 1|\boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k}), \boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f},\ell}), \mathbf{y}_\mathrm{f})$ if quantized features $\boldsymbol{q}_k(\mathbf{x}_{\mathrm{f},k})$ and $\boldsymbol{q}_\ell(\mathbf{x}_{\mathrm{f},\ell})$ are independent to other features $\boldsymbol{q}_{-\{k,\ell\}}(\mathbf{x}_{\mathrm{f},-\{k,\ell\}})$.

Note also that in this setting and for $n \to +\infty$, the first step of the Metropolis-Hastings described hereafter suffices since the $p_{k,\ell}$'s converge to $\delta^\star_{k,\ell}$ thanks to the properties of the BIC criterion.

Consequently, if at step s of the Markov chain, $\delta^{(s)}_{k,\ell} = 1$ (resp. 0) and $p_{k,\ell}$ is close to 0 (resp. 1), a good candidate for $\boldsymbol{\delta}^{(s+1)}$ should be to change $\delta_{k,\ell}$ to $\delta^{(s+1)}_{k,\ell} = 0$ (resp. $\delta^{(s+1)}_{k,\ell} = 1$). Our proposal is thus to calculate the difference between the current interaction matrix and P which is denoted by $T^{(s)} = |\boldsymbol{\delta}^{(s)} - P|$ and normalized.

This defines a proper transition probability between two interaction matrices (recall that $\odot$ denotes the element-wise Hadamard product):

$$T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}') = \begin{cases} 0 \text{ if } \|\boldsymbol{\delta}^{(s)} - \boldsymbol{\delta}'\|_{1,1} \neq 1, \\ T^{(s)} \odot |\boldsymbol{\delta}^{(s)} - \boldsymbol{\delta}'| \text{ otherwise.} \end{cases}$$

Acceptance probability of the proposed transition

Following Algorithm (1), a Metropolis-Hastings step can now be conducted by drawing $\boldsymbol{\delta}' \sim T(\boldsymbol{\delta}^{(s)}, \cdot)$. The acceptance probability of this candidate is given by:

$$\begin{aligned} A &= \min\left(1, \frac{p(\boldsymbol{\delta}'|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f})}{p(\boldsymbol{\delta}^{(s)}|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f})} \frac{1 - T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}')}{T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}')}\right) \\ &\approx \min\left(1, \exp\left(\frac{\mathrm{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q},\delta_{k,\ell}=0}) - \mathrm{BIC}(\hat{\boldsymbol{\theta}}_{\boldsymbol{q},\delta_{k,\ell}=1})}{2}\right) \frac{1 - T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}')}{T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}')}\right). \end{aligned}$$

It must here be remarked that by construction of $T^{(s)}$ and the transition probability $T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}')$, we have $T(\boldsymbol{\delta}', \boldsymbol{\delta}^{(s)}) = 1 - T(\boldsymbol{\delta}^{(s)}, \boldsymbol{\delta}')$. Still following Algorithm (1), the candidate $\boldsymbol{\delta}'$ is accepted with probability A s.t. $\boldsymbol{\delta}^{(s+1)} = \begin{cases} \boldsymbol{\delta}' \text{ with probability } A, \\ \boldsymbol{\delta}^{(s)} \text{ with probability } 1 - A. \end{cases}$

Validity of the approach

The existence of the stationary distribution $p(\boldsymbol{\delta}|\boldsymbol{q}(\mathbf{x}_\mathrm{f}), \mathbf{y}_\mathrm{f})$ is guaranteed by construction of the Metropolis-Hastings algorithm as the generated Markov chain fulfills the detailed balance condition. The uniqueness of the stationary distribution is given by the ergodicity of the Markov chain: as $\forall\, 1 \leq k < \ell \leq d$, $T^{(s)}_{k,\ell} > 0$ and a transition changes only one entry $\delta_{k,\ell}$ of the interaction matrix, every state can be reached in at most $\frac{d(d-1)}{2}$ steps.

In practice with a fixed quantization scheme, this stochastic approach is probably outperformed in computing time by LASSO-based methods or correlation-based methods like [18], which might obtain a suboptimal model in a fixed computing time, contrary to our approach which might take lots of steps to converge in distribution. Its double benefit however lies in the ability of the practitioner to define before-hand how many steps shall be performed and the natural integration to the quantization algorithm proposed in the previous chapter, which we develop in the next section.

4.4 Interaction screening and quantization

We return to our original objective (4.4) and consider optimizing the BIC criterion both in terms of quantization and pairwise interactions, as varying the quantization $\boldsymbol{q}$ might influence the "best" interactions $\boldsymbol{\delta}^\star$ and vice versa.

We can mix the MCMC approach proposed in the previous section with the *glmdisc* algorithm proposed in the previous chapter. A brute force way of doing this is to conduct a full Metropolis-Hastings algorithm, as proposed in the previous section, for each proposed quantization $\hat{\boldsymbol{q}}^{(s)}$ from Chapter 3 (either with the SEM or NN approach). Of course, this is too computationally intensive: our proposed Metropolis-Hastings algorithm necessitates the calculation of 2 bivariate logistic regression per $p_{k,\ell}$ *i.e.* $d(d-1)$ logistic regressions and one logistic regression per step (s). These $O(d^2 + S)$ logistic regressions shall be estimated for each proposed quantization which itself requires the estimation of $O(d)$ softmax (see Section 3.4.1) or polytomous logistic regression and contingency tables (see Section 3.5.2). Focus is given to the *glmdisc*-SEM approach in what follows.

The foremost bottleneck lies in the aforementioned initialization of the Metropolis-Hastings algorithm. In essence, the proposed initialization is not required: the closer the proposal

distribution to the target distribution, the quicker the convergence, which was our aim with the formulation of $p_{k,\ell}$ in Equation (4.5). However, the asymptotic convergence in distribution is nevertheless maintained whatever proposal distribution is used, provided the resulting Markov Chain has a unique stationary distribution. We will consequently design a proposal distribution of $\boldsymbol{\delta}$ for all quantizations $\mathfrak{q} \in \mathfrak{Q}_{\boldsymbol{m}}$ ($\mathfrak{q}$ is the latent features representing the quantization $\boldsymbol{q}(\boldsymbol{x})$ of $\boldsymbol{x}$ and $\mathfrak{Q}_{\boldsymbol{m}}$ is the space of all latent quantizations with $\boldsymbol{m} = (m_j)_1^d$ levels - see Section 3.5). To still put "*prior*" information (to the Markov Chain, it is not *a priori* in the Bayesian sense) about potential interactions into it, we will rely, to construct the transition kernel T from bivariate logistic regression, on the "raw" features instead:

$$p_{k,\ell} = p(\delta_{k,\ell} = 1 | \mathbf{x}_{\mathrm{f},k}, \mathbf{x}_{\mathrm{f},\ell}, \mathbf{y}_{\mathrm{f}}) \propto \exp\left(\frac{\mathrm{BIC}(\hat{\boldsymbol{\theta}}_{x_k, x_\ell, \delta_{k,\ell}=0}) - \mathrm{BIC}(\hat{\boldsymbol{\theta}}_{x_k, x_\ell, \delta_{k,\ell}=1})}{2} \right). \tag{4.6}$$

The second bottleneck is the number of Metropolis-Hastings steps per SEM step. It must be remembered that these two stochastic algorithms are meant to be used for their asymptotic properties. The target distribution of $\mathfrak{q}$ in the SEM algorithm is thought to be extremely dominated by $\mathfrak{q}^\star$ as argumented in Section 3.5.2, such that after some step (s), proposed quantizations are very close to $\boldsymbol{q}^\star$ (and consequently very similar to each others) at which point the interaction screening algorithm while performing the quantization steps is somewhat similar to its "static" counterpart developed in the preceding section.

As a consequence, a single interaction matrix can be proposed at each step of the SEM-Gibbs algorithm proposed in Section 3.5. Apart from the initialization defined above, nothing changes in the *glmdisc*-SEM algorithm: the SEM-Gibbs sampler allowed us to "hold" latent features $\mathfrak{q}_{-\{j\}}^{(s)}$ while drawing $\mathfrak{q}_j^{(s+1)}$ for all features, resulting in the latent features of all features and observations $\mathbf{q}^{(s+1)}$. Here, the same applies with the interaction matrix $\boldsymbol{\delta}$ that is drawn holding $\mathbf{q}^{(s+1)}$. The formal algorithm is detailed in Appendix A.2.3 and is reproduced partially hereafter.

Initialization First, P is computed according to the formulation of $p_{k,\ell}$ given above in Equation (4.6). Second, related to the Metropolis-Hastings algorithm, we need to initialize the interaction matrix $\boldsymbol{\delta}^{(0)}$ following a uniform distribution, as hypothesized earlier. Related to the SEM algorithm, $\mathbf{q}^{(0)}$ is also initialized uniformly (from an equiprobable multinouilli distribution, see Section 3.5.3).

S-steps Then, the SEM algorithm begins: following Section 3.5, the SEM-Gibbs sampler is also still applicable and $\mathfrak{q}_j^{(s)}$ can be drawn according to:

$$\mathfrak{q}_j^{(s)} \sim p_{\hat{\boldsymbol{\theta}}^{(s-1)}}(y | \mathfrak{q}_{-\{j\}}^{(s-1)}, \cdot, \boldsymbol{\delta}^{(s-1)}) p_{\hat{\boldsymbol{\alpha}}^{(s-1)}}(\cdot | x_j).$$

At this point, $\boldsymbol{\delta}^{(s)}$ can be computed from the Metropolis-Hastings algorithm similarly to the preceding section.

M-steps Then the logistic regression parameters are estimated:

$$\begin{aligned}
\boldsymbol{\theta}^{(s)} &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathbf{q}^{(s)}, \mathbf{y}_{\mathrm{f}}, \boldsymbol{\delta}^{(s)}) \\
\boldsymbol{\alpha}_j^{(s)} &= \underset{\boldsymbol{\alpha}_j}{\operatorname{argmax}}\, \ell(\boldsymbol{\alpha}_j; \mathbf{x}_{\mathrm{f},j}, \mathbf{q}_j^{(s)}) \text{ for } 1 \leq j \leq d.
\end{aligned}$$

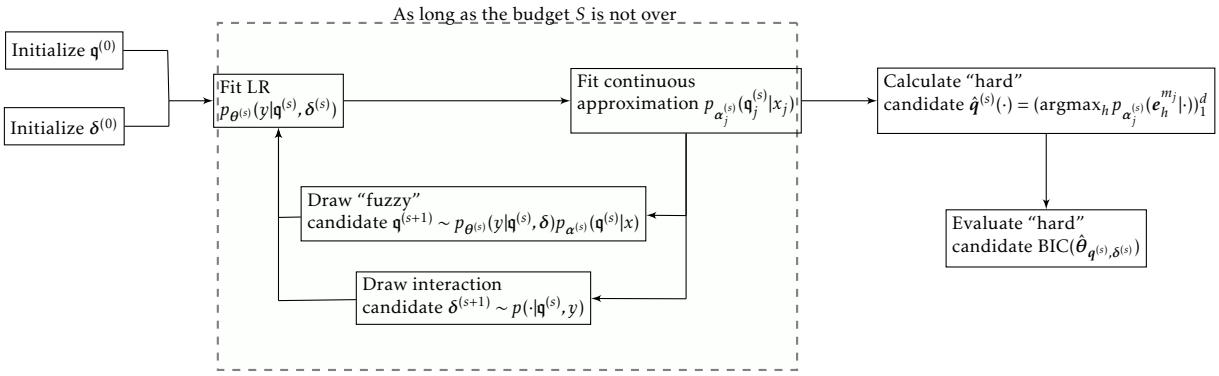


Figure 4.2 – Schema of the SEM quantization and Metropolis-Hastings interaction screening approach.

Note the presence of δ here, which is the only difference with what was proposed in Chapter 3.

Quantization and interaction provider We iterate this procedure for a user-defined number of steps S. Parallel to these steps, "hard" quantizations $\hat{\boldsymbol{q}}^{(s)}$ are derived from the MAP rule (3.9) and their associated logistic regression coefficients $\hat{\theta}^{(s)}$ follows from the MLE:

$$\hat{\theta}^{(s)} = \underset{\theta}{\text{argmax}}\, \ell(\theta; \hat{\boldsymbol{q}}^{(s)}(\mathbf{x}_f), \mathbf{y}_f, \delta^{(s)})).$$

The best quantization and interaction matrix can then be selected via BIC as in Equations (3.12) and (4.4), or any other model selection tool as emphasized earlier.

This procedure is much less costly than what could originally be thought of: the $d(d-1)/2$ logistic regression necessary to initialize the Metropolis-Hastings algorithm are still needed, but only once, and at each step, only one logistic regression, supplementary to the complexity of the original *glmdisc*-SEM approach, is needed.

The full algorithm is described in Appendix A.2.3 and schematically in Figure 4.2. We turn to numerical experiments in the next section on simulated, benchmark and real data.

4.5 Numerical experiments

In the same flavor as Chapter 3, the proposed algorithm for interaction screening is first tested on simulated data, to show empirically its consistency, then on benchmark datasets and on *Credit Scoring* data from CACF as in the previous chapter. The same scheme is applied for the *glmdisc* algorithm augmented with the interaction screening approach as described in the previous section. The code used for numerical experiments is available as packages, see Appendix B.

4.5.1 Simulated data

In this first part, focus is given on showing empirically the consistency of the approach. The same data generation process as in Chapter 3 is employed: two continuous features x_1 and x_2 are sampled from the uniform distribution on $[0,1]$ and discretized as exemplified on Figure 3.14 by

using

$$\boldsymbol{q}_1(\cdot) = \boldsymbol{q}_2(\cdot) = (\mathbb{1}_{]-\infty,1/3]}(\cdot), \mathbb{1}_{]1/3,2/3]}(\cdot), \mathbb{1}_{]2/3,\infty[}(\cdot)).$$

Here, following (3.2), we have $d = 2$ and $m_1 = m_2 = 3$ and the cutpoints are $c_{j,1} = 1/3$ and $c_{j,2} = 2/3$ for $j = 1, 2$. Setting $\boldsymbol{\theta}^1 = (0, -2, 2, 0, -2, 2, 0)$, $\boldsymbol{\delta}^1 = 0$, $\boldsymbol{\theta}^2 = (\boldsymbol{\theta}^1, \boldsymbol{\theta}_{1,2})$, where $\theta^{r,3}_{1,2} = \theta^{3,t}_{1,2} = 0$ for all $1 \leq r, t \leq 3$ as defined in Section 4.1, $\theta^{1,1}_{1,2} = \theta^{2,2}_{1,2} = 4$ and $\theta^{1,2}_{1,2} = \theta^{2,1}_{1,2} = -4$, $\boldsymbol{\delta}^2 = 1$; the target feature y is then sampled from $p_{\boldsymbol{\theta}^o}(\cdot|\boldsymbol{q}(\mathbf{x}), \boldsymbol{\delta}^o), o = \{1, 2\}$ via the logistic model (4.1). Two cases are studied:

(a) First, we assess that in the absence of a true interaction, no interaction is found by *glmdisc* while quantizing the data, so that we simulate $Y \sim p_{\boldsymbol{\theta}^1}$ and provide $\mathbf{x}$;

(b) Second, we assess that in the presence of a true interaction, it is discovered by *glmdisc* while quantizing the data, so that we simulate $Y \sim p_{\boldsymbol{\theta}^2}$ and provide $\mathbf{x}$.

Note that the interaction screening procedure is also applicable to continuous features, which is not tested here. These 2 experiments are run 100 times with $n = \{1{,}000, 10{,}000\}$ and histograms are given in Table 4.1. They show good performance and empirical consistency.

Table 4.1 – For *glmdisc* w.o. providing true quantization and different sample sizes n, (a) Bar plot of $\hat{\boldsymbol{\delta}} = 0, 1$ (resp.) for $\boldsymbol{\delta} = 0$. (b) Bar plot of $\hat{\boldsymbol{\delta}} = 0, 1$ (resp.) for $\boldsymbol{\delta} = 1$.

Algorithm	n	(a)	$\hat{\boldsymbol{\delta}}$	(b)	$\hat{\boldsymbol{\delta}}$
glmdisc w.o. provided quantization	1,000	39 61		60 40	
glmdisc w.o. provided quantization	10,000	6 94		85 15	

4.5.2 Benchmark datasets

We complement Table 3.4 (see Section 3.6.2 for details about the datasets) with the *glmdisc*-SEM approach with interactions in the last column of Table 4.2.

Half of the datasets do not benefit from the enriched model space but the performance of *glmdisc* without interactions was already high such that it is likely that the proposed decision boundaries were already close to the oracle and interaction coefficients are not significant. The other half shows a slightly superior performance.

In addition to these datasets, we used *glmdisc*-SEM on medicine-related datasets for a seminar talk in a biostatistics research team. These new benchmark datasets are Pima (available in R, $n = 768$, $d = 8$), Breast (available in R and also available on UCI under the name "Breast Cancer Wisconsin (Original)", $n = 699$, $d = 10$) and Birthwt (available in R, $n = 189$, $d = 16$). Table 4.3 shows the obtained Gini indices: for Pima, ALLR performs best such that the linearity assumption is probably "not as false" as in our motivational example in Section 3.2; for Breast, ALLR, *glmdisc*-SEM and *glmdisc*-SEM w. interactions show similar results, but at least for *Credit Scoring* practitioners, the resulting quantized scorecard as in Table 3.1 is more interpretable. For Birthwt, *glmdisc*-SEM w. interactions clearly outperforms all other approaches.

4.5.3 Real data from Crédit Agricole Consumer Finance

We complement Table 3.5 (see Section 3.6.3 for details about the datasets) with the *glmdisc*-SEM approach with interactions in the last column of Table 4.4.

Table 4.2 – Gini indices (the greater the value, the better the performance) of our proposed quantization algorithm *glmdisc* and two baselines: ALLR and MDLP / χ^2 tests obtained on several benchmark datasets from the UCI library.

Dataset	ALLR	*ad hoc* methods	Our proposal: *glmdisc*-NN	Our proposal: *glmdisc*-SEM	*glmdisc*-SEM w. interactions
Adult	81.4 (1.0)	**85.3** (0.9)	80.4 (1.0)	81.5 (1.0)	81.5 (1.0 - no interaction)
Australian	72.1 (10.4)	84.1 (7.5)	92.5 (4.5)	**100** (0)	**100** (0 - no interaction)
Bands	48.3 (17.8)	47.3 (17.6)	58.5 (12.0)	**58.7** (12.0)	**58.8** (13.0)
Credit	81.3 (9.6)	88.7 (6.4)	**92.0** (4.7)	87.7 (6.4)	87.7 (6.4 - no interaction)
German	52.0 (11.3)	54.6 (11.2)	**69.2** (9.1)	54.5 (10)	56.5 (9.0)
Heart	80.3 (12.1)	78.7 (13.1)	**86.3** (10.6)	82.2 (11.2)	84.5 (10.8)

Table 4.3 – Gini indices of our proposed quantization algorithm *glmdisc*-SEM and two baselines: ALLR and ALLR with all pairwise interactions on several medicine-related benchmark datasets.

	Pima	Breast	Birthwt
ALLR	**73.0**	94.0	34.0
ALLR LR w. interactions	60.0	51.0	15.0
glmdisc-SEM	57.0	93.0	18.0
glmdisc w. interactions	62.0	**95.0**	**54.0**

The enriched model space allows obviously for better predictive performance, provided this space can be effectively visited by the Metropolis-Hastings algorithm, itself very dependent of the propositional transition probability. It seems effective on real data since it yields the best performance on most datasets, significantly above all other methods, at the price of higher computational cost.

4.6 Conclusion

The essentially industrial problem of introducing pairwise interactions in a supervised multivariate classification setting was formalized and a new approach, relying on a Metropolis-Hastings algorithm has been proposed. This algorithm relies on the use of logistic regression, although other predictive models can be plugged in place of p_θ as argued in the preceding chapter.

The true underlying motivation was to perform interaction screening while quantizing data

Table 4.4 – Gini indices (the greater the value, the better the performance) of our proposed quantization algorithm *glmdisc*, the two baselines of Table 3.4 and the current scorecard (manual / expert representation) obtained on several portfolios of Crédit Agricole Consumer Finance.

Portfolio	ALLR	Current performance	*ad hoc* methods	Our proposal: *glmdisc*-NN	Our proposal: *glmdisc*-SEM	*glmdisc*-SEM w. interactions
Automobile	**59.3 (3.1)**	55.6 (3.4)	59.3 (3.0)	58.9 (2.6)	57.8 (2.9)	**64.8 (2.0)**
Renovation	52.3 (5.5)	50.9 (5.6)	54.0 (5.1)	**56.7 (4.8)**	55.5 (5.2)	55.5 (5.2 - no interaction)
Standard	39.7 (3.3)	37.1 (3.8)	45.3 (3.1)	43.8 (3.2)	36.7 (3.7)	**47.2 (2.8)**
Revolving	62.7 (2.8)	58.5 (3.2)	63.2 (2.8)	62.3 (2.8)	60.7 (2.8)	**67.2 (2.5)**
Mass retail	52.8 (5.3)	48.7 (6.0)	61.4 (4.7)	**61.8 (4.6)**	61.0 (4.7)	60.3 (4.8)
Electronics	52.9 (11.9)	55.8 (10.8)	56.3 (10.2)	**72.6 (7.4)**	62.0 (9.5)	63.7 (9.0)

using the approach developed in the preceding chapter: *glmdisc*. The experiments showed that, as was sensed empirically by statisticians in the field of *Credit Scoring*, interactions between quantized features can indeed provide better models than without interactions, or standard logistic regression. This novel approach allows practitioners to have a fully automized and statistically well-grounded tool that achieves better performance than both *ad hoc* industrial practices and academic quantization heuristics at the price of decent computing time but much less of the practitioner's valuable time.

Moreover, in Section 4.3.3, we set a uniform prior on the interaction matrix to simplify subsequent calculations and because it made sense for the *Credit Scoring* industry. However, without much modifications nor difficulty, it can be re-introduced which is of particular interest to *e.g.* biostatistics applications where a lot more features are considered and expert-knowledge is available to "guide" the Markov Chain by choosing an appropriate prior over the interaction matrix.

The previous chapters were about constructing one scorecard while a financial institution like CACF might have dozens of them, such that they are scarcely reviewed (one new / replacement scorecard should appear to the reader, at this point of the manuscript, relatively costly in terms of practitioners' time). The next chapter aims at proposing a strategy to build several in a one-shot fashion.

References of Chapter 4

[1] William D Berry, Jacqueline HR DeMeritt, and Justin Esarey. « Testing for interaction in binary logit and probit models: Is a product term essential? » In: *American Journal of Political Science* 54.1 (2010), pp. 248–266.

[2] Jacob Bien, Jonathan Taylor, and Robert Tibshirani. « A lasso for hierarchical interactions ». In: *Annals of statistics* 41.3 (2013), p. 1111.

[3] Vincent Calcagno, Claire de Mazancours, et al. « glmulti: an R package for easy automated model selection with (generalized) linear models ». In: (2010).

[4] Changzheng Dong et al. « Exploration of gene–gene interaction effects using entropy-based methods ». In: *European Journal of Human Genetics* 16.2 (2008), p. 229.

[5] Yingying Fan et al. « Innovated interaction screening for high-dimensional nonlinear classification ». In: *The Annals of Statistics* 43.3 (2015), pp. 1243–1272.

[6] W Keith Hastings. « Monte Carlo sampling methods using Markov chains and their applications ». In: *Biometrika* 57.1 (1970), pp. 97–109.

[7] James Jaccard. *Interaction effects in logistic regression*. Vol. 135. SAGE Publications, Incorporated, 2001.

[8] James Jaccard, Jim Jaccard, and Robert Turrisi. *Interaction effects in multiple regression*. 72. Sage, 2003.

[9] Charles Kooperberg and Ingo Ruczinski. « Identifying interacting SNPs using Monte Carlo logic regression ». In: *Genetic epidemiology* 28.2 (2005), pp. 157–170.

[10] Emilie Lebarbier and Tristan Mary-Huard. « Le critère BIC : fondements théoriques et interprétation ». In: RR-5315 (2004), p. 17. URL: https://hal.inria.fr/inria-00070685.

[11] Jiahan Li et al. « A model-free approach for detecting interactions in genetic association studies ». In: *Briefings in bioinformatics* 15.6 (2013), pp. 1057–1068.

[12] Yang Li and Jun S Liu. « Robust variable and interaction selection for logistic regression and general index models ». In: *Journal of the American Statistical Association* (2018), pp. 1–16.

[13] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[14] Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

[15] Craig Morgan et al. « Adversity, cannabis use and psychotic experiences: evidence of cumulative and synergistic effects ». In: *The British Journal of Psychiatry* 204.5 (2014), pp. 346–353.

[16] Mee Young Park and Trevor Hastie. « Penalized logistic regression for detecting gene interactions ». In: *Biostatistics* 9.1 (2007), pp. 30–50.

[17] Holger Schwender and Katja Ickstadt. « Identification of SNP interactions using logic regression ». In: *Biostatistics* 9.1 (2007), pp. 187–198.

[18] Noah Simon and Robert Tibshirani. « A Permutation Approach to Testing Interactions for Binary Response by Comparing Correlations Between Classes ». In: *Journal of the American Statistical Association* 110.512 (2015).

[19] Robert Tibshirani. « Regression shrinkage and selection via the lasso ». In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), pp. 267–288.

[20] Clément Vital. « Scoring pour le risque de crédit : variable réponse polytomique, sélection de variables, réduction de la dimension, applications ». 2016REN1S111. PhD thesis. 2016. URL: http://www.theses.fr/2016REN1S111.

[21] Xiang Wan et al. « BOOST: A fast approach to detecting gene-gene interactions in genome-wide case-control studies ». In: *The American Journal of Human Genetics* 87.3 (2010), pp. 325–340.

[22] Haitian Wang et al. « Interaction-based feature selection and classification for high-dimensional biological data ». In: *Bioinformatics* 28.21 (2012), pp. 2834–2842.

[23] Tong Tong Wu et al. « Genome-wide association analysis by lasso penalized logistic regression ». In: *Bioinformatics* 25.6 (2009), pp. 714–721.

[24] Can Yang et al. « SNPHarvester: a filtering-based approach for detecting epistatic interactions in genome-wide association studies ». In: *Bioinformatics* 25.4 (2008), pp. 504–511.

[25] Xiang Zhang et al. « TEAM: efficient two-locus epistasis tests in human genome-wide association study ». In: *Bioinformatics* 26.12 (2010), pp. i217–i227.

[26] Yu Zhang and Jun S Liu. « Bayesian inference of epistatic interactions in case-control studies ». In: *Nature genetics* 39.9 (2007), p. 1167.

Chapter 5

Tree-structure segmentation for logistic regression

All religions, arts and sciences are branches of the same tree. All these aspirations are directed toward ennobling man's life, lifting it from the sphere of mere physical existence and leading the individual towards freedom.

Albert Einstein, "Moral Decay", 1937.

Sommaire

In Chapter 1, it was argued in Section 1.2.2 that, what is referred to as "segmentation" in the *Credit Scoring* industry, could be a straightforward solution to deal with missing values and outliers that are quite common problems in *Credit Scoring*. This means we learn "expert" logistic regression models on separate "segments" of clients arranged in a tree. Its more theoretical justification is similar to that of quantization, sketched in Section 3.2, which is to achieve a good bias-variance trade-off of the predictive task. This goal was embedded explicitly in the proposed quantization algorithm. Here again, the resulting segmentation and scorecards therein can be viewed as a single model for the whole population. In the next section, we give some industrial context to the problem which is followed in Section 5.2 by a literature review. Section 5.3 reinterprets this problem, as advertised, as a model selection problem for which a specific approach is designed in subsequent sections.

5.1 Introduction

5.1.1 Context

As was emphasized in all previous chapters, logistic regression is the building block of a scorecard predicting the creditworthiness of an applicant and partly automating the acceptance / rejection mechanism. However, estimating logistic regression coefficients means that training data $(\mathbf{x}, \mathbf{y})$ is available. This is not the case when a new product, *e.g.* smartphone leasing, is added to the acceptance system. On a practical note, some other previously learnt scorecard may not be applicable on this new market because the same information is not asked to applicants, *e.g.* marital status, because given the low amounts at stake, it was decided to collect the fewest data possible, to make the process as simple and quick as possible. On a more theoretical note, it is probable that applicants to smartphone leasing are not stemming from the same data generating mechanism $\boldsymbol{X}, Y \sim p$ as any other previous applicants (*i.e.* on other markets). Put it another way, the possibility of having several logistic regression scorecards on sub-populations of the total portfolio allows to have more flexibility, and thus it potentially reduces model bias discussed in Chapter 1.

For these reasons, several industries, among which *Credit Scoring*, rely on several "weak learners" such as logistic regression, arranged in a tree. Such decision process is illustrated on Figure 5.1. This tree structure and the vocabulary of "weak learners" would indicate a use-case of Mixtures of Experts [8] or aggregation / ensemble methods [12] respectively. However, these fuzzy methods imply that all applicants are scored by all scorecards, which is obviously neither desirable (for interpretation purposes) nor feasible (since available features differ).

The next section illustrates how such a structure is achieved using CACF's in-house practices.

5.1.2 In-house *ad hoc* practice

Credit Scoring practitioners are often asked by the management to "locally" study the decision process displayed on Figure 5.1 by *e.g.* merging branches (Standard loans and Leasing for Fiat) or conversely to separate sub-populations by splitting a leaf (Kawasaki into Standard loans and Leasing). To do so, they resort to simple unsupervised generative "clustering" techniques, such as PCA and its refinements on categorical or mixed data (MCA or FAMD resp.) which are described hereafter, used to represent the data on the two first principal axes, and derive "clusters" from separated point clouds.

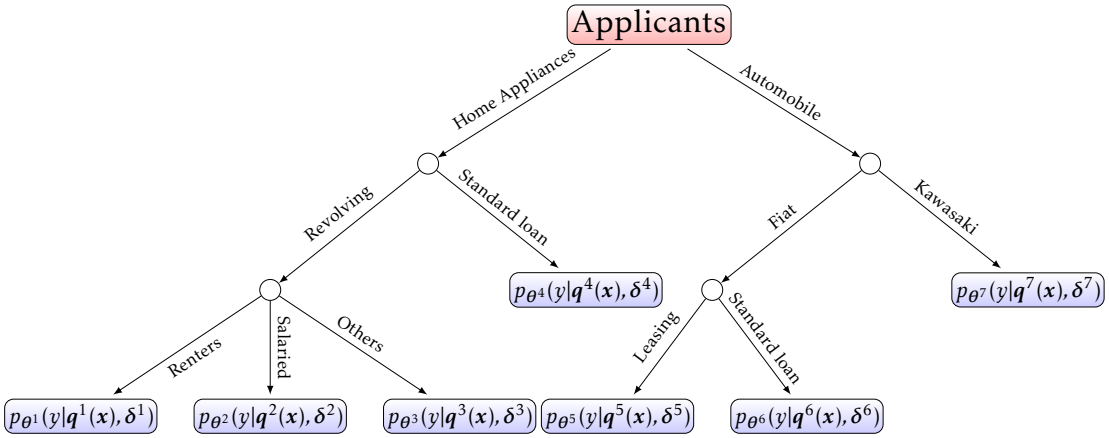
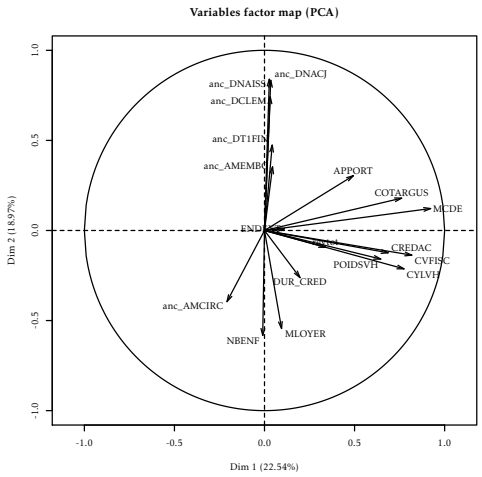


Figure 5.1 – Simplified cartography of the application scorecards.

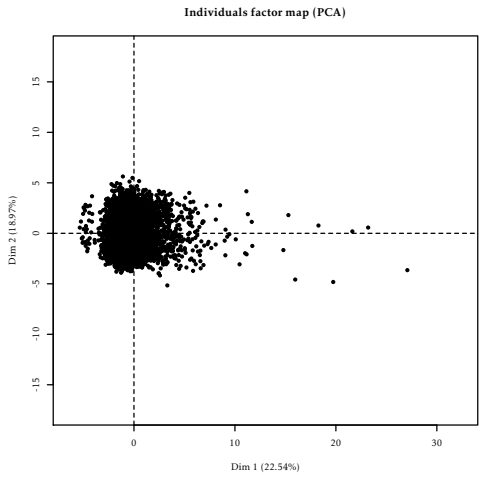
Principal Component Analysis (PCA) The goal of PCA [13] is to represent observations graphically in a way that exhibits most efficiently their similitude and differences by combining input features in so-called orthogonal "principal components" $\boldsymbol{u} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d)$ such that the inertia of each axis j (the variance of $\boldsymbol{x}'\boldsymbol{u}_j$) is maximized. It can be shown that it is equivalent to seeking the ordering of the eigenvalues $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_d)$ of the covariance matrix $\Sigma = \mathbf{x}'\mathbf{x}$. The explained variance of each axis j is given by $\frac{\lambda_j}{\sum_{j'=1}^{d} \lambda'_j}$. Classically, only the two first axes $(\boldsymbol{u}_1, \boldsymbol{u}_2)$ (after reordering from largest to lowest explained variance) are used. The composition of theses axes in the original features x_j is often represented first, to see if groups of features can be formed which would define the subsequent segments. PCA has been applied to the Automobile dataset from CACF (n = 50,000, d = 25 among which 18 continuous and 7 categorical features which number of levels go from 5 - family status - to 100 - brand of the vehicle and 200,000 missing entries) on Figure 5.2, where the aforementioned principal components' composition is displayed on Figure 5.2a (relying on the FactoMineR R package [10]): interestingly, the first axis is dominated by car and loan characteristics such as the vehicle's price ("APPORT", "MCDE", "CREDAC"), its fiscal and mechanical characteristics ("CVFISC", "POIDSVH", "CYLVH") while the second axis is composed of clients' characteristics such as their age ("anc_DNAISS", "anc_DNACJ"), their number of children ("NBENF"), their job stability ("anc_AMEMBC"). Note that a good portion of the total variance is explained by the first axes (22.54 % and 18.97 % resp.). The second classical representation is the observations themselves in this new space, which is displayed on Figure 5.2b: no clear group is distinguishable from the pack. With these two representations, the *Credit Scoring* practitioner decide if, visually, clusters are formed (*i.e.* clouds with little intra-class variance) which would be used to build separate scorecards ($p_{\theta^1}, p_{\theta^2}$).

However, the *Credit Scoring* data is of mixed type and *Credit Scoring* practitioners are used to quantizing the data, as explained thoroughly in Chapter 3, such that the MCA algorithm, specific to categorical features, becomes applicable to all features, by using *e.g. equal-freq* or χ^2 tests (see Appendix A).

Multiple Correspondence Analysis (MCA) In presence of only categorical features (or following quantization), the MCA algorithm is more appropriate: it extends the PCA approach to categorical features by using the disjunctive table (dummy / one-hot encoding of $\mathbf{x}$ described in Section 1.2.4). For a thorough introduction to MCA, see *e.g.* [11]. What is most interesting in this



(a) The first axis is dominated by features linked to the financed good and characteristics of the loan while the second axis is composed of characteristics of the client and durations.



(b) However, no clear segments appear on the new representation of data along these two axes.

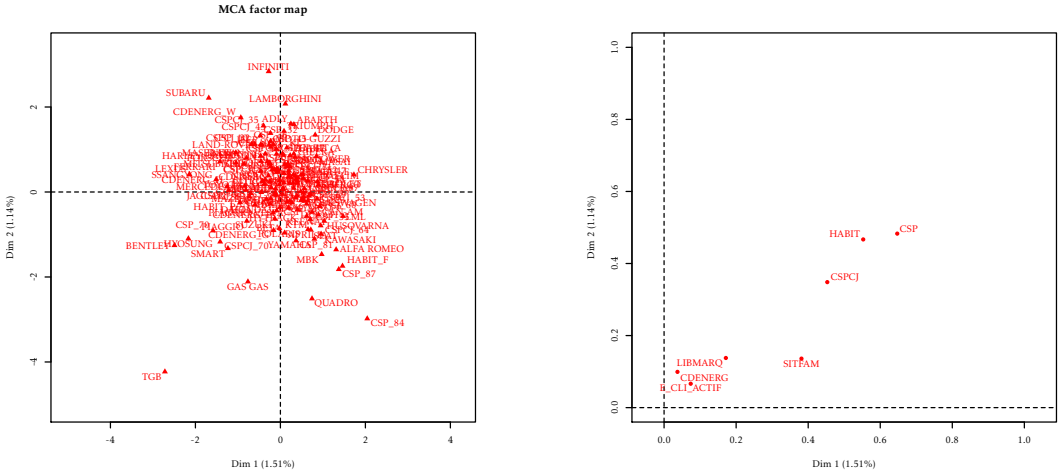
Figure 5.2 – The result of a PCA applied to continuous features of CACF data from the car loan market.

method is that both categorical features' levels and observations can be simultaneously displayed on the first principal components axes. This is of high practical interest in *Credit Scoring* because clouds of points are directly characterized by the categorical levels that are displayed nearest, contrary to PCA where groups correspond to surfaces of equation $\boldsymbol{x}'\boldsymbol{u}_1 + \boldsymbol{x}'\boldsymbol{u}_2 \geq \alpha$ where α encodes the separation boundary of resulting clusters, which would be the edges of our decision system, as on Figure 5.1, and make it arguably less interpretable.

When applied to the Automobile dataset, the MCA algorithm yields Figure 5.3 (relying again on the FactoMineR R package [10]). As categorical features' levels are dummy encoded, they are all represented separately as in Figure 5.3a. Unfortunately, as the vehicle's brand takes a lot of levels, this figure is not very informative. A useful trick, apart from grouping levels, is to plot the barycentre of a feature's levels (weighted by the number of observations in each level), as displayed on Figure 5.3b. Note that a low portion of the total variance is explained by the first axes (1.51 % and 1.14 % resp.) since the data, when one-hot encoded, is very high dimensional. As for PCA, no groups are formed when displaying the (uninformative) equivalent of Figure 5.2a and no factor level(s) is / are isolated from the others in Figure 5.3a. A practitioner would conclude the absence of segments on which to build several scorecards.

Nevertheless, a method applicable to mixed data exists as well and could directly be applied to "raw" features.

Factor Analysis of Mixed Data (FAMD) The FAMD algorithm [13] aims at performing both MCA on categorical features and PCA on continuous features in a simultaneous fashion. Resulting principal component axes depend on both data types, as can be seen from Figure 5.4 (relying again on the FactoMineR R package [10]). As on Figures 5.2a and 5.3a, categorical features' levels' and continuous features' contributions to the two first principal components can



(a) The factor map contains all levels of all categorical features, making it hard to read. Note that very little of the total variance is explained by the two first axes.

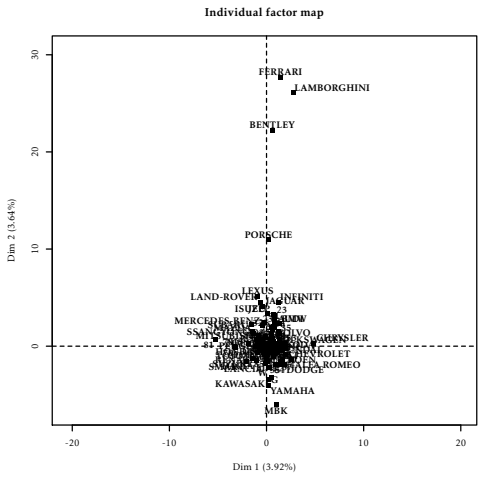
(b) The barycenter of a categorical feature is obtained by weighting the contribution of each level in Figure 5.4a.

Figure 5.3 – The result of an MCA applied to categorical data of CACF data from the car loan market.

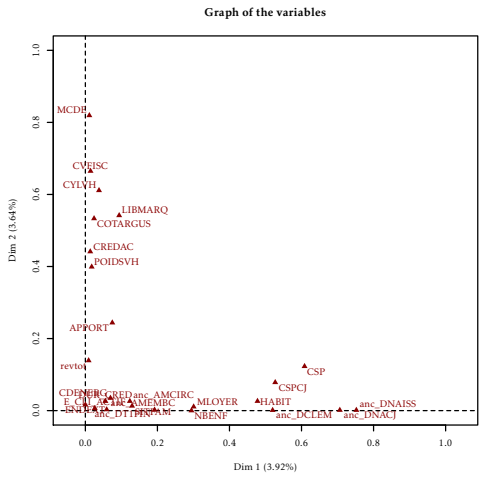
be displayed on Figure 5.4a, where vehicle brands make it rather hard to read. When switching to the categorical features' levels barycentre representation, as in Figure 5.3b, interpretation is easier and somewhat similar to the PCA method (up to a permutation on the first and second components): the first axis contains information about the client, represented by continuous (*e.g.* age) and categorical features (*e.g.* job category) while the second axis is about the loan and the vehicle (its brand, cost, etc.). This method has the advantage of not requiring the *Credit Scoring* practitioner to preprocess the "raw" data by quantizing it, which could have a huge impact on the results of the subsequent method employed, as argued in Chapter 3. Moreover, the practitioner would fine-tune the quantization of each sub-population which, if fed back to the MCA, would potentially yield completely different results! As for PCA and MCA, the equivalent of Figure 5.2a (not shown here) for FAMD does not display distinguishable groups of observations. Nevertheless, the luxury car brands are now well separated in Figure 5.4a from other continuous features and other categorical features' levels which would be interpreted by a *Credit Scoring* practitioner as the need to build a specific scorecard for this market. However, due to the low volumes of applicants and considering that all of them are probably good clients that are all accepted (the score has little relevance for such markets), no segmentation would be performed.

It appears clearly that all these methods do not directly optimize a predictive goal such as the one optimized by logistic regression. Moreover, the *ad hoc* preprocessing step of quantization might influence the structure of the retained segmentation.

For numerical experiments of Section 5.6, we will use, among others, the FAMD approach.



(a) Analogous of Figure 5.3a, categorical features' levels involved in the FAMD can be plotted but not much can be drawn from the graph.



(b) Categorical features can be represented as the barycenter of their levels as on Figure 5.3b alongside the contribution of continuous features. The interpretation of the axes is switched: the first one is composed of clients' characteristics, *e.g.* age and job, while the second relates to the financed good and the characteristics of the loan, *e.g.* its brand or the down payment.

Figure 5.4 – The result of a FAMD applied to categorical data of CACF data from the car loan market.

5.1.3 These practices can fail

Of course, like all *ad hoc* methods that rely on "two-stages" procedures (find segments using an MCA algorithm and learn separate logistic regression scorecards on them) which do not share a common objective, the aforementioned in-house practice can fail. *Credit Scoring* practitioners are probably aware that their methods are not bullet-proof, but like most industries, unless provided to them with easily usable software replacing these methods, these practices remain.

This chapter has no intent in filling that gap as was ambitioned in Chapters 3 and 4 but rather to give insights on more elaborate, readily usable methods that will be covered in Section 5.2 and to propose a few ideas for future research. That is why, in the present, we show two data generating mechanisms where current in-house methods fail. In Section 5.3, we will propose an SEM algorithm that shares similitude with the one proposed for quantization in Section 3.5 that performs well where current methods fail.

The first of these failing situations is when the pdf of covariates (suppose for simplicity that all of them are continuous) $p(\boldsymbol{x})$ is multi-modal as on Figure 5.5 where we distinguish the lower, middle and upper-classes of respective low, average and high wages and indebtedness. An unsupervised generative approach like PCA would urge the practitioner to construct 3 scorecards (one for each of the aforementioned classes). However, displaying y as red (resp. green) for bad borrowers (resp. good borrowers), we can see that perfect separation can be achieved: it depends solely on the indebtedness level (the ratio of wages over indebtedness). Thus, the resulting scorecards would be asymptotically the same, but they use three times more parameters! In a

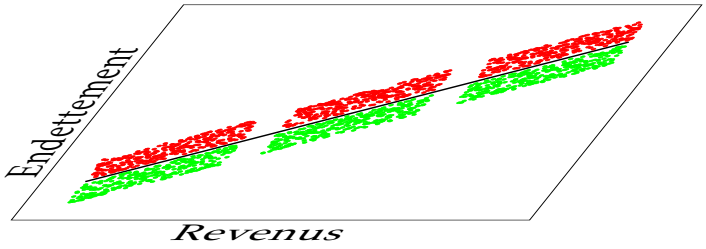


Figure 5.5 – Multi-modal wages and indebtedness data generating mechanism with $y = \{0, 1\}$ classes displayed in red and green respectively.

finite sample setting, and following remarks in Chapters 1, 3 and 4 on model bias and model selection consistency, it will imply lower performance since each of these coefficients have three times less samples to train on, which amounts to increasing the variance by the same factor. On a practical note, one could argue that it reduces interpretability by adding an avoidable complexity to the decision system. This particular data generating mechanism is revisited in the experiments of Section 5.6.1.

The second failing situation is the counterpart of the first tailored data generating mechanism. This time, suppose the covariates wages and indebtedness are uniformly sampled. Suppose there is a third categorical feature "wages source" which is drawn uniformly from three levels: renters, salaried workers and self-employed. One could argue that renters' risk level do not depend on their indebtedness, which is typically low (and a higher one is a major red flag), salaried workers' risk level is positively correlated with their indebtedness ratio as was the case for the first introductory example (see Figure 5.5) and self-employed people's risk level is negatively correlated with this indebtedness ratio (say, the higher their personal engagement, the higher the chances of success of their business). This example data generating mechanism is illustrated on Figure 5.6. In this situation, and contrary to the first example, an unsupervised generative "clustering" algorithm like the projection of the data on the two first PCA axes shown here would not partition the data and the *Credit Scoring* practitioner would construct only one scorecard. This scorecard would have high model bias since it is too simple to accommodate for the variety of the data generating mechanism and consequently perform poorly. This particular data generating mechanism is also revisited in the Experiments of Section 5.6.1.

5.2 Literature review

This section aims at providing an eluded literature review of some well-known supervised 'clustering approaches that could be transposed to the *Credit Scoring* industry.

5.2.1 Supervised generative clustering methods

In the preceding section, examples of classical unsupervised "clustering" methods were given: PCA (continuous data), MCA (categorical data) and ultimately FAMD (mixed data), completed with a projection of the data on their respective two first axes. In this section, focus is given to supervised generative methods. Indeed, a fully generative model $p(\boldsymbol{x}, y)$, if sufficiently flexible, could have easily spotted the bottlenecks of the failures of the PCA approach illustrated on Figures 5.5 and 5.6.

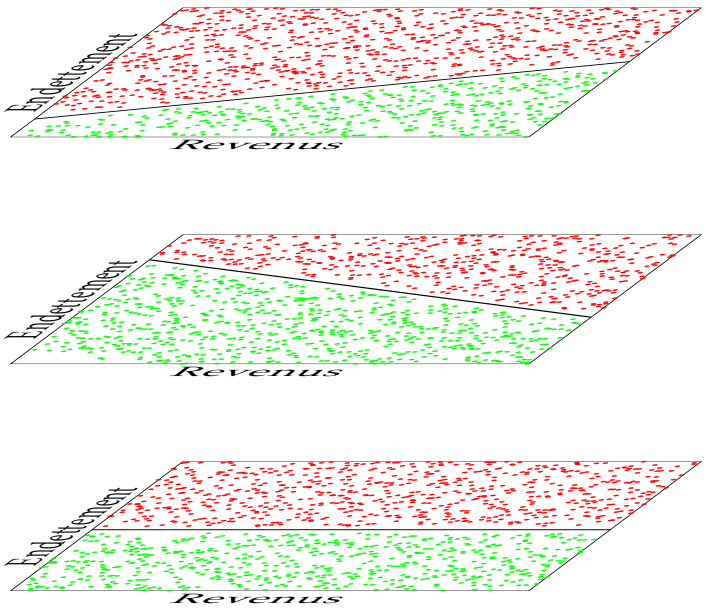


Figure 5.6 – Uni-modal wages and indebtedness data generating mechanism with $y = \{0, 1\}$ classes displayed in red and green respectively which depends on a third feature.

Partial Least Squares (PLS) The PLS [18] algorithm seeks to combine the strengths, in its original proposal, of PCA in explaining the variance of the features $\boldsymbol{x}$ and regression in predicting y with the resulting principal components. In a classification setting, it is termed PLS-DA where DA stands for discriminant analysis.

The main idea is to construct a first component from the sum of the univariate regressions of $\mathbf{x}_j$ on $\mathbf{y}$, then a second component from the sum of the univariate regressions of $\mathbf{x}_j$ subtracted by the first component on $\mathbf{y}$, and so on. In a sense, a trade-off between reconstruction quality of $\mathbf{x}$ and $\mathbf{y}$ with as few components as possible is achieved.

The PLS algorithm is given in Section A.4.2 of the Appendix, Algorithm 10. It was used in [16] in a classification setting which results in Figure 5.7 reproduced with permission [1]. It is striking how classes are better separated when using PLS. However, this does not guarantee that the resulting inferred segments' logistic regression will yield better predictive performance, considering that a *Credit Scoring* practitioner would effectively spot two groups in Figure 5.7 (right) and separate them on the first PLS axis being above or below a threshold of approximately 0.01. When applied to the Automobile dataset, it does not show such spectacular results (see Figure 5.8).

Supervised Principal Components (SPC) The SPC [1] algorithm is motivated by genomics applications where $d > n$, but is applicable to our current setting as well, and by the fact that, in a predictive setting, variance of the features $\boldsymbol{x}$ is only interesting if correlated with y. The inner-workings of the algorithm are relatively simple: the correlation between each feature x_j and y is computed. Only the features for which this correlation exceeds a user-defined threshold are retained, and the first few principal components of these features are calculated and used to predict y.

1. ©2009 IEEE

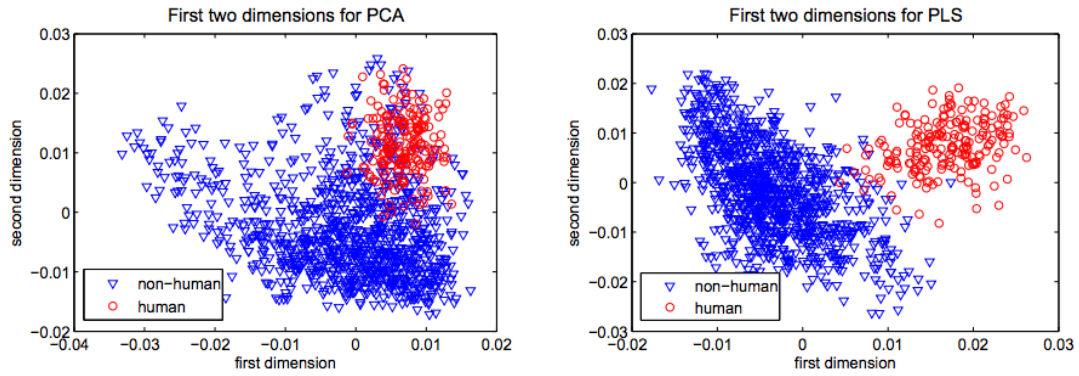


Figure 5.7 – Cloud points resulting from the application of PCA (left) and PLS (right) on a binary-labelled multivariate continuous dataset.

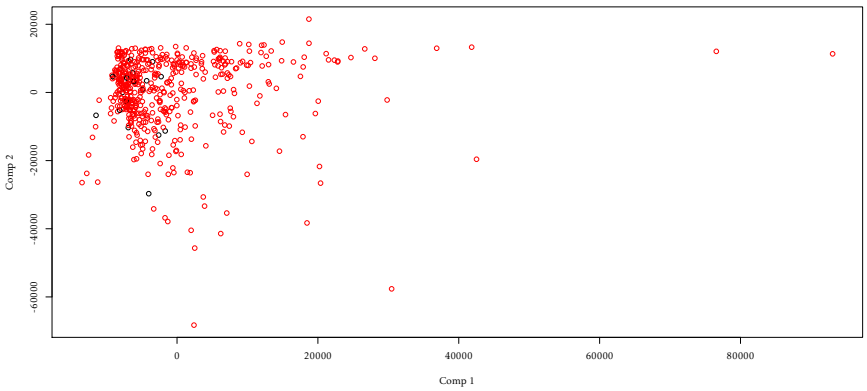


Figure 5.8 – Cloud points resulting from the PLS algorithm applied to the running example of the Automobile dataset with good and bad borrowers in red and black respectively.

There is a close link between PLS and SPC that is thoroughly explained in [5] Section 18.6.2 p. 680. For numerical experiments of Section 5.6, we will use, among others, the PLS approach.

Although these methods make good use of y in constructing sub-populations on which the practitioner would construct separate scorecards, the resulting segments would be, as described in the MCA Section, visually separated clouds of points on the graph of the two first principal components. This paradigm has two major drawbacks: first, as explained in the preceding section, the separation boundary is complex and multivariate (as the two first principal components will most likely involve all features). Second, to make a complete tree as on Figure 5.1, these procedures would have to be repeated "recursively" which yields the need for a stopping criterion and an objective splitting criterion in place of the rather subjective visual separation. Direct approaches of estimating such trees are reviewed in the next section.

5.2.2 Direct approaches: logistic regression trees

Logistic Tree with Unbiased Selection (LOTUS) The first research work focusing on a similar problem than the present one seems to be Logistic Tree with Unbiased Selection (LOTUS) [3], where logistic regression trees are constructed so as to select features to split the data on the tree's nodes which break the linearity assumption of logistic regression. The original article states an application case similar to this one, namely the insurance market.

Their motivation is that logistic regression has a fixed parameter space, defined by the number of input features, whereas trees adapt their flexibility (*i.e.* depth) to the sample size n; however, trees perform well for classification (*i.e.* their label estimates $\hat{y}$ can achieve low classification error) but poorly in assessing the probability of the event (*i.e.* the estimate $\hat{p}(y|\boldsymbol{x})$ is the proportion of the event y among observations $\boldsymbol{x}$ at each leaf which is arguably not very informative) as it is piecewise constant; if the true decision boundary separating the two classes of y given $\boldsymbol{x}$ is linear, they need an infinite depth to estimate it as well as logistic regression. Thus, they search for trees which leaves are logistic regression with a few continuous features and which intermediate nodes split the population based on categorical or continuous features which relationship to the log-odd ratio of y is not linear (*i.e.* features that would perform poorly in a logistic regression).

They propose a feature selection method for node splits that is claimed to be "bias-free": as seen in Chapters 3 and 4, the number of partitions of l_j labelled factor levels into m_j unlabelled categories (which would here be the tree split criterion and define its sub-populations) is huge which yields overfitting; thus, their approach relies on a χ^2 test which degrees of freedom is linked to the number of potential rearrangements of l_j levels into 2 bins to avoid wrongfully selecting categorical features that have lots of levels. Their optimized criterion is the sum of the log-likelihoods of the logistic regression on the tree's leaves. Of course, this leads also to overfitting which requires the tree to be pruned (as is classical for classification trees) using a method closely related to the one developed in the classical CART [2] algorithm. Lastly, their proposed method is not directly applicable to missing values: these observations are not used during training (in the *Credit Scoring* industry, there would most likely be at least one missing value for each observation) and during test, their missing values are imputed by the mean or median.

To sum up, although their motivation is similar to the present problem, LOTUS is not directly usable since only continuous features are used as predictive features in the logistic regression of the tree's leaves, it does not handle missing values gracefully, and there are currently no implementation available in R or Python.

Logistic Model Trees (LMT) The second approach very close to our industrial problem is named LMT [9]. As for LOTUS, the result is a tree of logistic regression at its leaves and the motivation is very similar. Their introductory example, reproduced here with permission on Figure 5.9 is enlightening: a quadratic bivariate boundary cannot be well approximated by trees or logistic regression alone, but a combination of both achieves good performance and interpretation.

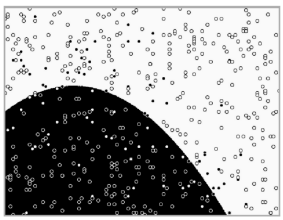
Their approach differs however drastically from LOTUS in that they rely on a particular boosting approach derived from the LogitBoost algorithm [6] to adjust the logistic regression, and an adaptation of the classical C4.5 [15] algorithm to grow the tree. The two central ideas behind their usage of the LogitBoost algorithm, reproduced in Algorithm 9 of Section A.4.1 of the appendices, are simple: it allows to perform feature selection *via* a stagewise-like process where one feature enters the model at each step and to recursively "refine" the logistic regression by boosting the logistic regression fitted at a node's parent. Indeed, a first logistic regression is fitted at the tree's root via LogitBoost using all observations in $\mathcal{T}_f$, which is further boosted separatly at its subsequent children nodes on sub-populations, say $((\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^2, \mathbf{y}^2))$ and so on. This most probably induces less parameter estimation variance in each leaf since they partly benefit from samples not in their leaf but used to fit the parents' logistic regression. One if its main advantages compared to other approaches is that it is fast. Here again, the resulting tree must be pruned and either a tactic similar to the classical tree algorithm CART, or cross-validation, or the AIC criterion (in a refinement of the method proposed in [17]) are used.

However, categorical features are dummy / one-hot encoded so that only a few factor levels might be selected at each leaf, which amounts to merging the not selected levels into a reference value. Conversely, when used as a split feature, each level yields a distinct branch. Moreover, missing values are imputed by the mean or mode.

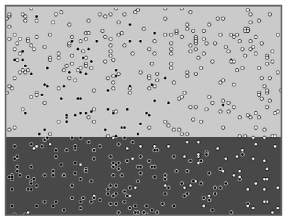
Its original implementation is in Java (Weka) but the R package RWeka provides interfaces and wrappers to it. When applied directly to the Automobile dataset, as LMT does not handle missing values, a first preprocessing step is to select only complete observations: there remains only approx. 4,000 observations (among 50,000) and no segmentation is performed. Due to the use of the LogitBoost algorithm, only a few features are selected: one continuous feature and three particular levels of three different categorical features, yielding a rather low performance of 44.3 Gini points (compared to the current performance of 55.7) which is nevertheless impressive given the few training observations and features used. To help the LMT algorithm further, features with the highest rate of missing values are deleted; we now have $d = 21$ and $n = 20{,}000$. Finally, in a third experiment, missing values of categorical features are encoded as a particular level and continuous features' missing values are imputed by their mean, all observations and features can now be used. In these two last experiments, the LogitBoost algorithm seems to fail since no segmentation is performed and only the age of the client is retained, yielding a low performance (30 Gini points).

Model-Based Recursive Partitioning (MOB) Lastly, a third approach closely related to our problem is MOB [20] which is an adaptation of a better-known paper [7] to parametric models in the leaves of a recursively partitioned dataset (hence the name).

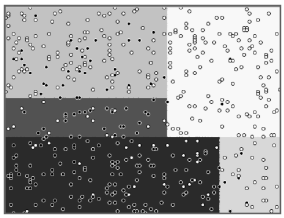
Their algorithm consists in fitting the chosen model (in our case, logistic regression) for all observations in $\mathcal{T}_f$ at the current node and decide to split these into subsets based on a correlation measure (several such measures are proposed) of the residuals of the current model and splitting features $\boldsymbol{V} = (V_1, \ldots, V_p)$ where $V_j \in \mathbb{R}$ or $\mathbb{N}_{l_j}$, $1 \leq j \leq p$, which are not necessarily included in $\mathbf{x}$ (they are specified by the user). The procedure is repeated until no significant "correlation" is detected. The C4.5 algorithm, in presence of a binary outcome, orders the levels of categorical features by their proportion of events y and split as if the feature were ordinal (it can be shown



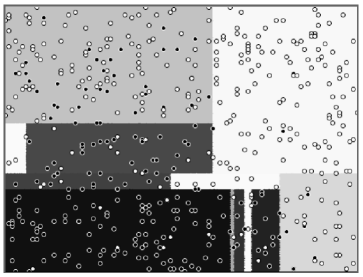
(a) Quadratic data generation process: the true boundary depends on the square of input features.



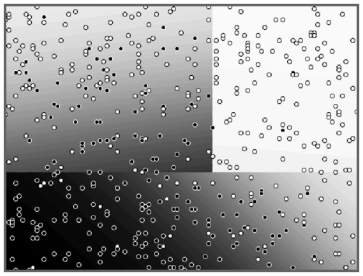
(b) The first split of a classification tree is way too simplistic.



(c) The second split is more helpful.



(d) The subsequent splits yield overfitting: these nodes shall be pruned.



(e) A logistic regression tree with only two leaves (and consequently two logistic regression) yields the best result.

Figure 5.9 – LMT motivational example.

that it is optimal, see [5] Section 9.2.4). Similarly to LOTUS and contrary to C4.5, MOB performs, for example for binary splits and when confronted to categorical features j having l_j levels, 2^{l_j} tests. Moreover, there is no mention of an eventual treatment of missing values. Finally, the number of segments per split is searched exhaustively.

Its implementation is available through the R packages party and partykit which will be used in numerical experiments of Section 5.6. It worked well on small toy data, however, on real data, even with complete cases ($n = 4{,}000$) and by arbitrarily selecting $d = 4$ features, computation took a very long time. With bigger datasets, I got "file size"-related errors.

To sum up, these direct approaches are far more promising than unsupervised and supervised generative approaches of Sections 5.1.2 and 5.2.1 respectively, in that they produce directly the sought tree-structure of Figure 5.1 (apart, of course, from quantization $\boldsymbol{q}^1,\ldots,\boldsymbol{q}^7$ and interactions $\boldsymbol{\delta}^1,\ldots,\boldsymbol{\delta}^7$). However, their treatment of missing values and categorical features are not satisfactory: classical *Credit Scoring* data would require preprocessing steps such as imputation or quantization (or at least merging numerous factor levels) which might greatly influence the resulting segmentation as emphasized in Section 5.1.1. Moreover, quantization has to be segment-specific: on a theoretical note, it participates in reducing model bias; on a practical note, it does not make much sense to use the same quantization of wages on segments of applicants to a leasing for a Ferrari or for a smartphone.

In the next section, we formalize the problem as a model selection problem, similarly to the three approaches presented here, with our own notations introduced in the previous section and

chapters.

5.3 Logistic regression trees as a combinatorial model selection problem

We assume there are $K^\star$ "clusters" which form the leaves of a tree similar to Figure 5.1 and which assigning latent random feature is denoted by $C^\star$ (lower-case for observations). The other notations employed inspire from the preceding chapters: the superscript notation is used to insist on the fact that available features $\boldsymbol{x}^{c^\star}$ differ potentially in each of the scorecards. For $c^\star \in \mathbb{N}_{K^\star}$, $(\mathbf{x}^{c^\star}, \mathbf{y}^{c^\star})$ denotes the subset of observations of $(\mathbf{x}_\mathrm{f}, \mathbf{y}_\mathrm{f})$ for which $C^\star = c^\star$, such that $\cup_{c^\star=1}^{K^\star}(\mathbf{x}^{c^\star}, \mathbf{y}^{c^\star}) = (\mathbf{x}_\mathrm{f}, \mathbf{y}_\mathrm{f})$ and for $c^\star, c^{'\star}$, $(\mathbf{x}^{c^\star}, \mathbf{y}^{c^\star}) \cap (\mathbf{x}^{c^{'\star}}, \mathbf{y}^{c^{'\star}}) = \varnothing$. It follows that quantizations $\boldsymbol{q}^{c^\star}$ and interactions $\boldsymbol{\delta}^{c^\star}$, discussed in Chapters 3 and 4 respectively are also different. Consequently, the logistic regression coefficients $\boldsymbol{\theta}^{\star,c^\star}$ are also obviously different. In this section, we drop the quantization and interactions requirement, such that:

$$\forall \boldsymbol{x}, y, \exists c^\star \in \mathbb{N}_{K^\star}, \boldsymbol{\theta}^{\star,c^\star} \in \Theta^{\star,c^\star}, p(y|\boldsymbol{x}) = p_{\boldsymbol{\theta}^{\star,c^\star}}(y|\boldsymbol{x}). \tag{5.1}$$

The membership of an observation $\boldsymbol{x}$ to a segment c is given by a tree. We restrict to binary trees for simplicity, such that a segment c with depth $\mathcal{D}(c)$ has $r = 1, \ldots, \mathcal{D}(c)$ parents successively denoted by $\mathcal{P}a^r(c)$. At these parent nodes, a binary rule is taken. This rule is univariate: it depends on only one feature $x_{\sigma(r,c)}$ where $\sigma(r,c)$ denotes the anti-rank of the feature used in rule r for segment c. Being a binary rule, the membership of $x_{\sigma(r,c)}$ is tested between $C_{\mathcal{P}a^r(c),1}$ and $C_{\mathcal{P}a^r(c),2}$ such that $C_{\mathcal{P}a^r(c),1} \cup C_{\mathcal{P}a^r(c),2} = \mathbb{R}$ for continuous features (half-spaces), or $\mathbb{N}_{l_{\sigma(r,c)}}$ for categorical features respectively. This membership is denoted by $\lambda(r,c)$ such that $x_{\sigma(r,c)} \in C_{\mathcal{P}a^r(c),\lambda(r,c)}$. With all these newly introduced notations, the probability of a segment c given covariates $\boldsymbol{x}$ can be expressed as:

$$p(c|\boldsymbol{x}) = \prod_{r=1}^{\mathcal{D}(c)} \mathbb{1}_{C_{\mathcal{P}a^r(c),\lambda(r,c)}}(x_{\sigma(r,c)}). \tag{5.2}$$

An example is given on Figure 5.10.

The above mentioned algorithms LOTUS, LMT and MOB optimized the sum of the segments' log-likelihoods, then needed pruning since it leads to obvious overfitting: infinite log-likelihood is achievable by putting each sample into its own segment, provided there is at least one continuous feature and no identical examples with different labels, or combinations of categorical features' levels that separate classes perfectly.

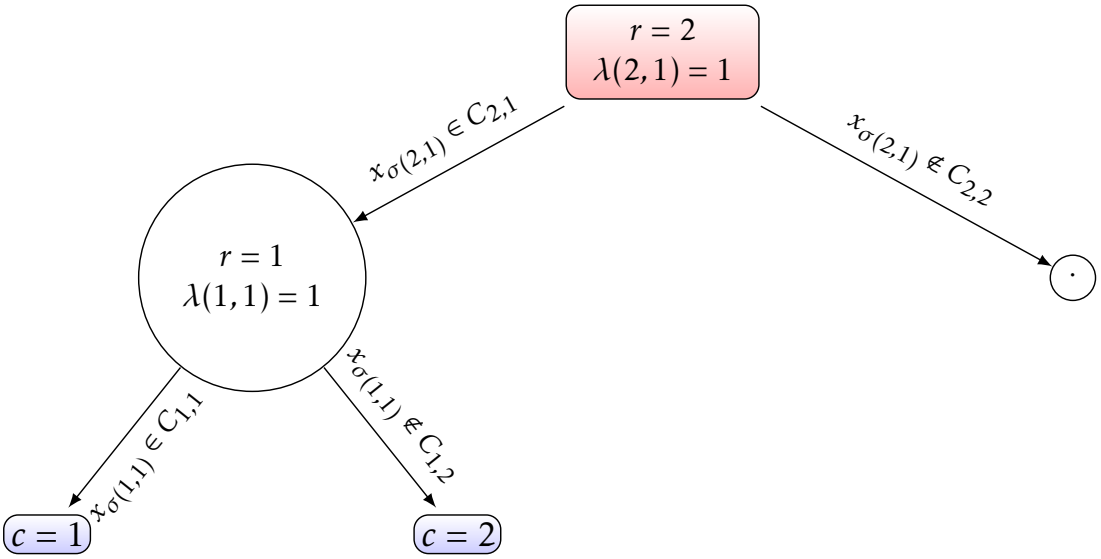


Figure 5.10 – Notations of a segmentation tree by an example.

Another approach can be taken by considering the segment c as a latent random feature:

$$\begin{aligned} p(\mathbf{x}_\mathrm{f}, \mathbf{y}_\mathrm{f}) &= \sum_{c=1}^{K^\star} p(\mathbf{y}_\mathrm{f}|\mathbf{x}_\mathrm{f}, c) p(c|\mathbf{x}_\mathrm{f}) p(\mathbf{x}_\mathrm{f}) && (p(c|\boldsymbol{x}) \text{ is non-zero only for } c = c^\star) \\ &= \prod_{c^\star=1}^{K^\star} p(\mathbf{y}^{c^\star}|\mathbf{x}^{c^\star}, c^\star) p(\mathbf{x}_\mathrm{f}) \\ &= \prod_{c^\star=1}^{K^\star} \int_{\Theta^{\star,c^\star}} p_{\boldsymbol{\theta}^{\star,c^\star}}(\mathbf{y}^{c^\star}|\mathbf{x}^{c^\star}) p(\boldsymbol{\theta}^{\star,c^\star}|c^\star) d\boldsymbol{\theta}^{\star,c^\star} p(\mathbf{x}_\mathrm{f}). \end{aligned} \tag{5.3}$$

Thus, we have:

$$\begin{aligned} \ln p(\mathbf{x}_\mathrm{f}, \mathbf{y}_\mathrm{f}) &= \sum_{c^\star=1}^{K^\star} \int_{\Theta^{\star,c^\star}} \ln p_{\boldsymbol{\theta}^{\star,c^\star}}(\mathbf{y}^{c^\star}|\mathbf{x}^{c^\star}) p(\boldsymbol{\theta}^{\star,c^\star}|c^\star) d\boldsymbol{\theta}^{\star,c^\star} + \ln p(\mathbf{x}_\mathrm{f}) \\ &\approx -\sum_{c^\star=1}^{K^\star} \mathrm{BIC}(\theta^{\star,c^\star})/2 + O(K^\star) + \ln p(\mathbf{x}_\mathrm{f}). \end{aligned}$$

Since in our application, the number of sample $n \approx 10^6$ is large and the number of desired segments $K^\star \approx 10$ is low, we use the following criterion to select a segmentation:

$$(K^\star, c^\star) = \operatorname*{argmin}_{K,c} \sum_{c=1}^{K} \mathrm{BIC}(\hat{\boldsymbol{\theta}}^c). \tag{5.4}$$

As was thoroughly explained for quantizations and interactions in Sections 3.3.4 and 4.2 respectively, it is unclear how many parameters should be accounted for in this BIC criterion since

the tree of Equation (5.2) has "parameters", in the sense that it selects a splitting feature and a splitting criterion, which have to be estimated (this is somewhat reflected in Equation (5.3) by the $p(\boldsymbol{\theta}^c|c)$ term); some are continuous (when the split is done on a continuous feature), some are discrete (when it concerns a categorical feature). As discussed in Section 3.3.4, discrete parameters are usually not counted, but here, following the C4.5 approach of considering the levels of categorical features as ordered (w.r.t. the proportion of events y associated to them - see Section 5.2.1, Paragraph Model-Based Recursive Partitioning (MOB)), a split on categorical features can count as one continuous parameter. However, when there are more than two classes c (typically, a financial institution of moderate to big size would have $K = 4$ to 30 scorecards), this "ordering" simplification about the search for discrete parameters does not apply. We still stick with criterion (5.4) as it will show good empirical properties in Section 5.6.

In the next section, we propose to relax the constraint of Equation (5.2), exactly as was done for quantizations in Chapter 3, by using a continuous approximation of this discrete problem (and thus highly difficult to optimize directly).

5.4 A mixture and latent feature-based relaxation

The difficulty in optimizing Criterion (5.4) directly lies in the discrete nature of c given $\boldsymbol{x}$, illustrated by the profusion of indicator functions in Equation (5.2), which is very similar to the problems of quantization (see Chapter 3 and in particular Section 3.4.1) and interaction screening (see Chapter 4 and in particular Section 4.3.3). In both cases, highly-combinatorial discrete problems were relaxed, by approaching door functions by softmax and relying on MCMC methods. A somewhat similar approach can be taken here to design a "smooth" approximation of $p(c|\boldsymbol{x})$ which will be denoted by $p_\beta(c|\boldsymbol{x})$: the classical probability estimate of decision trees consisting in the proportion of training examples in each class in all leaves.

5.4.1 The proposed relaxation: tree structure and piecewise constant membership probability

As emphasized in Section 5.2.1, classification trees aim at predicting c by making their leaves as "pure" as possible (hence the use of the term "impurity measure" to designate their optimized criterion), *i.e.* where one class strongly dominates the others by being the labels of most observations that fall into it. However, as for logistic regression, they can be viewed as probabilistic classifiers by substituting their classical majority vote by the proportion of each class in each leaf:

$$p_\beta(c|\boldsymbol{x}) = \frac{|\mathbf{c}^{\mathcal{L}(\boldsymbol{x}_\mathrm{f})}|}{|\mathbf{x}^{\mathcal{L}(\boldsymbol{x})}|}, \tag{5.5}$$

where $\mathcal{L}(\boldsymbol{x})$ denotes the leaf where $\boldsymbol{x}$ falls, $|\mathbf{c}^{\mathcal{L}(\boldsymbol{x})}|$ the number of training examples in $\mathbf{x}$ of class c, the number of training examples in $\mathbf{x}$ falling in leaf $\mathcal{L}(\boldsymbol{x}_\mathrm{f})$, and β is sloppily used to denote all parameters involved in classical classification tree methods such as CART and C4.5, as written in Equation (5.2). Indeed, in this soft assignment, $\mathcal{L}(\boldsymbol{x})$ and its segment c are not identifiable anymore. An example of such behaviour is given on Figure 5.11 where there are two classes: "survived" and "not survived" for Titanic passengers given their age, sex and passenger class. The proportion of each class in each leaf is given in parentheses.

This "soft" assignment will be useful to design an algorithm that does not greedily evaluate all possible segmentations of the form of Equation (5.2) and its subsequent logistic regression. A softmax could have been used similarly as in Chapter 3 but would have yielded a major drawback: the assignment decisions would have been multivariate, thus losing the interpretability of the

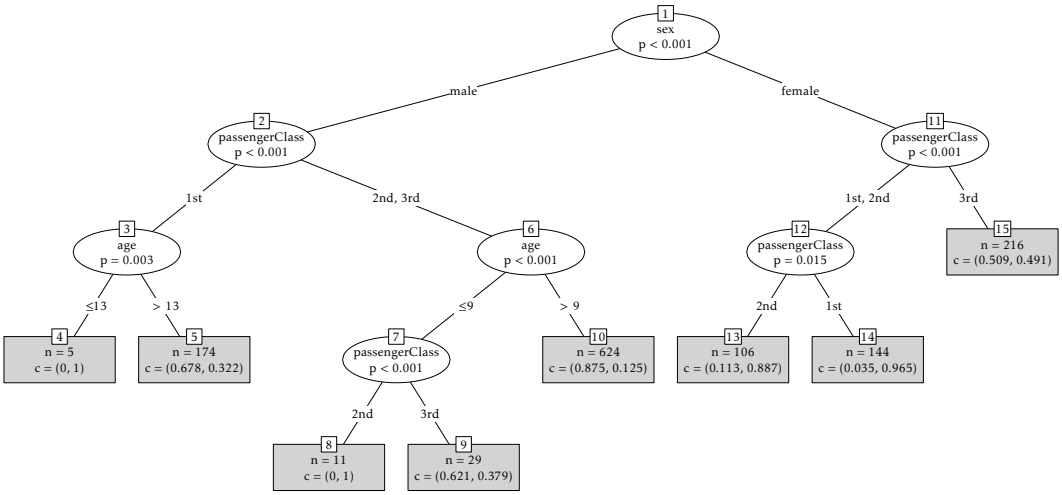


Figure 5.11 – A C4.5 decision tree applied to the famous Titanic dataset containing the fate of 1309 passengers alongside their class, age and sex.

tree structure. Using this new parametrization, we get a mixture model:

$$p(y|\boldsymbol{x}) = \sum_{c=1}^{K} p_{\boldsymbol{\theta}^c}(y|\boldsymbol{x}, c) p_{\boldsymbol{\beta}}(c|\boldsymbol{x}),$$

where feature c is latent and which makes immediately think of a straightforward estimation strategy: the EM algorithm. Indeed, it can be easily remarked that:

$$p(c|\boldsymbol{x}, y) \propto p_{\boldsymbol{\theta}^c}(y|\boldsymbol{x}, c) p_{\boldsymbol{\beta}}(c|\boldsymbol{x}),$$

which will be at the basis of the EM's fuzzy assignment among segments, detailed in the next section.

5.4.2 A classical EM estimation strategy

Following the preceding section, we would like to maximize the following likelihood, both in terms of the segments and their resulting logistic regressions:

$$\ell(\boldsymbol{\beta}, K, (\boldsymbol{\theta}^c)_1^K; \mathbf{x}_\mathrm{f}, \mathbf{y}_\mathrm{f}) = \sum_{c=1}^{K} \sum_{i=1}^{n} \ln p_{\boldsymbol{\theta}^c}(y_i|\boldsymbol{x}_i, c) p_{\boldsymbol{\beta}}(c|\boldsymbol{x}_i).$$

The EM algorithm [4] is an iterative method that can be used to estimate the *maximum a posteriori* of $p(c|\boldsymbol{x}, y)$, since c is latent, and alternates between the expectation (E-)step, which computes the relative membership of the observations into each segment, and a maximization (M-)step, which computes the MLE of the parameters of the log-likelihoods of each segment's logistic regression and the tree structure. These new logistic regression and tree estimates are then used to determine the distribution of the latent variables in the next E-step. Considering the number

of segments K fixed, the E- and M-steps of the EM can be derived as follows.

E-step At iteration $(s+1)$, the membership of an observation i to segment c can be computed as:

$$t_{i,c}^{(s+1)} = \frac{p_{\theta^{c(s)}}(y_i|\boldsymbol{x}_i)p_{\beta^{(s)}}(c|\boldsymbol{x}_i)}{\sum_{c'=1}^{K} p_{\theta^{c'(s)}}(y_i|\boldsymbol{x}_i)p_{\beta^{(s)}}(c'|\boldsymbol{x}_i)}.$$

For notational convenience, we denote the matrix of partial membership of all observations to all segments as $\mathbf{t} = (t_{i,c})_{1\leq i\leq n, 1\leq c\leq K}$.

M1-step The previous E-step allows to derive the new MLE of the logistic regression parameters of each segment c as:

$$\begin{aligned}\boldsymbol{\theta}^{c(s+1)} &= \operatorname*{argmax}_{\boldsymbol{\theta}^c} \mathbb{E}[\ell(\boldsymbol{\beta}, K, (\boldsymbol{\theta}^{c'})_1^K; \mathbf{x}_{\mathrm{f}}, \mathbf{y}_{\mathrm{f}}, \mathbf{t}^{(s+1)})|(\boldsymbol{\theta}^{(c(s)})_1^K, \boldsymbol{\beta}^{(s)}, K] \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{n} t_{i,c}^{(s+1)} \ln p_{\boldsymbol{\theta}^c}(y_i|\boldsymbol{x}_i).\end{aligned}$$

M2-step Similarly, a new tree structure can be derived by the new MLE of its parameter $\boldsymbol{\beta}$:

$$\begin{aligned}\boldsymbol{\beta}^{(s+1)} &= \operatorname*{argmax}_{\boldsymbol{\beta}} \mathbb{E}[\ell(\boldsymbol{\beta}, K, (\boldsymbol{\theta}^c)_1^K; \mathbf{x}_{\mathrm{f}}, \mathbf{y}_{\mathrm{f}}, \mathbf{t}^{(s+1)})|\boldsymbol{\theta}^{(c(s)}, \boldsymbol{\beta}^{(s)}, K] \\ &= \operatorname*{argmax}_{\boldsymbol{\beta}} \sum_{i=1}^{n}\sum_{c=1}^{K} t_{i,c}^{(s+1)} \ln p_{\boldsymbol{\beta}}(c|\boldsymbol{x}_i),\end{aligned}$$

where $p_{\boldsymbol{\beta}}(c|\boldsymbol{x}_i)$ is given by Equation (5.5) and estimated by relative frequency in each leaf, such that $p_{\boldsymbol{\beta}}(c|\boldsymbol{x}) = \frac{|\mathbf{c}^{\mathcal{L}(x)}|}{|\mathbf{x}^{\mathcal{L}(x)}|}$. Unfortunately, tree induction methods like CART or C4.5 do not follow a maximum likelihood approach, so that they rather try to minimize a so-called impurity measure, the Gini index or the entropy, respectively. However, since it is hoped that segments $c^\star$ are "peaks" of the distribution $p_{\boldsymbol{\beta}}(c|\boldsymbol{x})$, just as it was supposed that the best quantization $\mathfrak{q}^\star$ dominated its posterior pdf in the SEM algorithm proposed in Section 3.5.2, we assume the log-likelihood can be approximated by the entropy:

$$\boldsymbol{\beta}^{(s+1)} \approx \operatorname*{argmax}_{\boldsymbol{\beta}} \sum_{i=1}^{n}\sum_{c=1}^{K} t_{i,c}^{(s+1)} \underbrace{p_{\boldsymbol{\beta}}(c|\boldsymbol{x}_i)}_{\begin{cases}\approx 1 \text{ for } c = c^\star, \\ 0 \text{ otherwise.}\end{cases}} \ln p_{\boldsymbol{\beta}}(c|\boldsymbol{x}_i).$$

This last formulation allows to obtain $\boldsymbol{\beta}^{(s)}$ from a simple application of the C4.5 algorithm, with observations properly weighted by $t_{i,c}$.

However, this approach suffers from two main drawbacks: first, all observations are used in all logistic regression $p_{\boldsymbol{\theta}^c}$ which might be problematic with real data since there will be "blocks" of available features (*e.g.* vehicle information); second, all possible values of K must be iterated through since the EM algorithm does not allow for the disappearance of a segment c contrary to the SEM approach developed hereafter.

5.4.3 An SEM estimation strategy

In a similar fashion as the MCMC approaches developed in Chapters 3 and 4 where a "clever" quantization (resp. interaction matrix) was drawn and evaluated at each step, refining it for the subsequent steps, a straightforward way of building logistic regression trees is to propose a tree structure, fit logistic regression at its leaves, and evaluate the goodness-of-fit using Criterion 5.4 of the resulting logistic regression tree. This is somehow the way LMT works: a tree structure is proposed based on C4.5, logistic regression are fitted using the LogitBoost algorithm, and the tree is pruned back using a goodness-of-fit criterion.

Similarly to the quantization and the interaction screening problems, doing so for all possible tree structures is intractable, so that a way of generating "good" candidates can be designed by relying on an SEM algorithm, which we call *glmtree*. The E-step of the previous Section is thus replaced by a Stochastic (S-) step which has some consequences on the M-steps.

S-step The "soft" assignment of the EM algorithm of the previous Section is hereby replaced by a "hard" stochastic assignment such that:

$$c_i^{(s+1)} \sim p_{\boldsymbol{\theta}^{\cdot(s)}}(y_i|\boldsymbol{x}_i)p_{\boldsymbol{\beta}^{(s)}}(\cdot|\boldsymbol{x}_i).$$

M1-step Thanks to the previous step, the segments are now "hardly" assigned such that the logistic regression are estimated using only observations affected to their segment:

$$\begin{aligned}\boldsymbol{\theta}^{c(s+1)} &= \operatorname*{argmax}_{\boldsymbol{\theta}^c} \ell(\boldsymbol{\theta}; \mathbf{x}^{c(s+1)}, \mathbf{y}^{c(s+1)}) \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}^c} \sum_{i=1}^{n} \mathbb{1}_c(c_i^{s+1}) \ln p_{\boldsymbol{\theta}^c}(y_i|\boldsymbol{x}_i, c).\end{aligned}$$

M2-step Similarly, a new tree structure is given by:

$$\begin{aligned}\boldsymbol{\beta}^{(s)} &= \operatorname*{argmax}_{\boldsymbol{\beta}} \ln p_{\boldsymbol{\beta}}(c_i|\boldsymbol{x}_i) \\ &= \operatorname*{argmax}_{\boldsymbol{\beta}} \ell(\boldsymbol{\beta}; \mathbf{x}, \boldsymbol{c}).\end{aligned}$$

This last expression is again approximated by C4.5's impurity measure: the entropy. Without more theoretical and empirical work, it is unclear which of the EM and SEM approaches will perform best. However, as mentioned earlier, this SEM algorithm calls for an easy integration with the quantization and interaction screening methods proposed in the previous chapter.

5.4.4 Choosing an appropriate number of "hard" segments

Going back to "hard" segments The motivation of Section 5.4.1 was to propose a relaxation of Equation (5.2) so that an iterative estimation, be it an EM or an SEM algorithm, could be carried out. In Chapter 3, a similar relaxation was proposed for quantization, which lead us to propose a "soft" quantization $\boldsymbol{q}_{\boldsymbol{\alpha}}(\cdot)$ or $p_{\boldsymbol{\alpha}}(\mathfrak{q}_j|\cdot)$ for the neural network and the SEM approaches respectively. These relaxations allowed quantized features to be "partly" in all intervals or groups for continuous or categorical features respectively. Thus, to get back to the original quantization problem, a *maximum a posteriori* scheme was introduced in Section 3.4.1 to deduce "hard" quantizations from this relaxation. In our tree setting, a similar approach has to be taken:

this soft segmentation can be interpreted as a mixture of logistic regression which implies that all applicants are scored by all scorecards which is arguably not interpretable. An assignment of each applicant i to a single scorecard, *i.e.* to a leaf of the segmentation tree, is easily done again by a *maximum a posteriori* step such that:

$$\hat{c}_i^{(s)} = \underset{c}{\operatorname{argmax}}\, p_{\beta^{(s)}}(c|\boldsymbol{x}_i). \tag{5.6}$$

Segmentation candidates Similarly to the neural network architecture introduced in Section 3.4, the SEM algorithm which proposed quantization candidates introduced in Section 3.5 and the Metropolis-Hastings algorithm for pairwise interaction screening introduced in Section 4.3.3, the EM and SEM strategies introduced in the two previous sections for segmentation are merely "segments providers". Indeed, through the iterations 1 to S, as argued in the preceding paragraph, segmentations $\hat{\mathbf{c}}^{(1)},\dots\hat{\mathbf{c}}^{(S)}$ are proposed through a *maximum a posteriori* rule parallel to these algorithms. These candidates are then reintroduced to our original criterion (5.4) and the best performing segmentation is found according to:

$$s^\star = \underset{s}{\operatorname{argmin}}\, \mathrm{BIC}(\hat{\theta}^{c^{(s)}}), \tag{5.7}$$

which bears resemblance with Equation (3.12) for quantizations.

Exploring a fewer number of segments In the preceding sections, the number of segments K was assumed to be fixed. However, the *maximum a posteriori* scheme introduced in this section allows, similarly to the one used to go from "soft" ($\boldsymbol{q}_{\boldsymbol{\alpha}}(\cdot)$ or $p_{\boldsymbol{\alpha}}(\mathfrak{q}_j|\cdot)$) to "hard" ($\boldsymbol{q}(\cdot)$) quantizations, to explore a number of segments potentially way lower than K: for a fixed segment c, if there is no observation i such that $p_\beta(c|\boldsymbol{x}_i) > p_\beta(c'|\boldsymbol{x}_i)$ for $c' \neq c$, than the segment is empty, which is equivalent to producing a segmentation in $K-1$ segments. Supplemental to this thresholding effect, the use of an SEM algorithm makes it possible to enforce this phenomenon: as c is drawn in the S-step, and as was argued for quantizations with an SEM algorithm in Section 3.5, Paragraph Choosing an appropriate number of levels, there is a non-zero probability of not drawing a particular segment c at a given step (s). When run long enough, the chain will stop with $K = 1$, just like the *glmdisc*-SEM algorithm could be run until all features are quantized in one level. This can be seen as a strength since it does not require to loop on the number of segments K which would be required for an EM algorithm, which is why focus is given to the SEM algorithm in what follows.

5.5 Extension to quantization and interactions

The SEM estimation strategy proposed in the previous section has one clear advantage: it could easily be used in conjunction with the *glmdisc*-SEM algorithm proposed in Chapters 3 and 4 for quantization and interaction screening. Following the preceding sections, we would like to maximize the following likelihood, both in terms of segments, the quantizations in each segment and the resulting logistic regressions:

$$\ell((\boldsymbol{\alpha}^c)_1^K, \boldsymbol{\beta}, K, (\boldsymbol{\theta}^c)_1^K; \mathbf{x}_\mathrm{f}, \mathbf{y}_\mathrm{f}) = \sum_{c=1}^{K}\sum_{i=1}^{n} \ln p_{\theta^c}(y_i|\mathfrak{q}_i, c)\, p_\beta(c|\boldsymbol{x}_i)\, p_{\boldsymbol{\alpha}^c}(\mathfrak{q}_i|\boldsymbol{x}_i).$$

The following modifications to the two SEM algorithms (for the quantization and segmentation problems) previously proposed would have to be performed:

S1-step The segment is drawn, for an observation i such that $\boldsymbol{x}_i$ belongs to segment c, according to:

$$c_i^{(s+1)} \sim p_{\boldsymbol{\theta}^{\cdot(s)}}(y_i|\mathfrak{q}^{\cdot(s)}, \boldsymbol{\delta}^{\cdot(s)}) p_{\boldsymbol{\beta}^{(s)}}(\cdot|\boldsymbol{x}_i).$$

S2-step The *glmdisc*-SEM performs the subsequent S-steps. The quantization is drawn according to:

$$\mathfrak{q}_{i,j}^{c(s+1)} \sim p(y_i|\mathfrak{q}_{i,-\{j\}}, \cdot, \boldsymbol{\delta}^{c(s)}) p_{\boldsymbol{\alpha}_j^{c(s)}}(\cdot|x_{i,j}).$$

S3-step The interaction matrix is drawn following the Metropolis-Hastings approach developed in the preceding Chapter and denoted for simplicity as MH here:

$$\boldsymbol{\delta}^{c(s+1)} \sim \mathrm{MH}(\boldsymbol{\delta}^{c(s)}, \mathbf{q}^{c(s+1)}, \mathbf{y}^{c(s+1)}).$$

M1-step The logistic regression parameters are obtained in each segment by using the appropriate quantization, interaction matrix and observations:

$$\boldsymbol{\theta}^{c(s+1)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathbf{x}^{c(s+1)}, \mathbf{y}^{c(s+1)}, \boldsymbol{\delta}^{c(s+1)}).$$

M2-step In each segment and for each predictive feature in this particular segment, polytomous logistic links are fitted between the "soft" quantization and the raw feature:

$$\boldsymbol{\alpha}_j^{c(s+1)} = \underset{\boldsymbol{\alpha}_j}{\operatorname{argmax}}\, \ell(\boldsymbol{\alpha}_j; \mathbf{x}_j^{c(s+1)}, \mathbf{q}_j^{c(s+1)}).$$

M3-step The tree-structure is obtained again via the C4.5 algorithm as an approximation of:

$$\boldsymbol{\beta}^{(s+1)} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\beta}, \mathbf{x}, \boldsymbol{c}^{(s+1)}).$$

As proposed in Chapter 3, parallel to this SEM algorithm, "hard" quantizations are obtained by performing a *maximum a posteriori* operation on the quantization probability $p_{\boldsymbol{\alpha}}$ (see Section 3.4.1):

$$\hat{q}_{j,h}^{c(s)}(\cdot) = 1 \text{ if } h = \underset{1 \le h' \le m_j}{\operatorname{argmax}}\, p_{\boldsymbol{\alpha}_{j,h'}^{(c(s)}}(\boldsymbol{e}_{h'}^{m_j}|\cdot), 0 \text{ otherwise.}$$

As proposed in Section 5.4.4, "hard" segments are obtained *via* a *maximum a posteriori* operation on the segmentation probability $p_{\boldsymbol{\beta}}$ (see Equation (5.6)). The best logistic regression tree is thereafter chosen via the following BIC criterion (5.7) adapted from Equations (5.4) and (3.5).

Although this extension seems straightforward, it is relatively computationally expensive since at each step (s), K Metropolis-Hastings steps have to be performed and a tree, K logistic regression and $K \times d$ polytomous logistic regressions are fitted. With a relatively small number of segments, *i.e.* 4 to 30 as proposed earlier, it seems nevertheless feasible but it will require more work. In particular, since classification tree methods with more than 2 labels are computationally intensive when presented with categorical features with many levels (see Section 5.2.2), a straightforward workaround is to consider the quantized features as ordered.

Table 5.1 – Comparison of several clustering approaches w.r.t. the subsequent predictive performance in experiment (a).

	Oracle = ALLR	*glmtree*-SEM	FAMD	PLS	LMT	MOB
Gini	69.7	**69.7**	65.3	47.0	**69.7**	64.8

5.6 Numerical experiments

This chapter is based on more recent work which consequently limits the exhaustiveness of the numerical experiments. The next section aims at comparing the proposed approach to other methods on simulated data from the proposed model, and in particular the failing situations discussed in Section 5.1.3.

5.6.1 Empirical consistency on simulated data

As for the two preceding chapters, the first set of numerical experiments are dedicated to verifying empirically the consistency of the proposed approach. To do so, we simulate the failing situations presented in Section 5.1.3.

(a) Two covariates (x_1, x_2) are independently simulated from an equally probable mixture of $\mathcal{N}(3,1)$, $\mathcal{N}(6,1)$ and $\mathcal{N}(9,1)$ and the log odd ratio of y is given by $\theta_0 + \theta_1 x_1 + \theta_2 x_2$ where $\theta_0 = 3$, $\theta_1 = 0.5$ and $\theta_2 = -1$. This data generating mechanism is illustrated in Figure 5.12. Results of various clustering methods developed in this chapter are given in Table 5.1.

(b) Two covariates (x_1, x_2) are simulated from $\mathcal{U}(0,1)$ and a third categorical covariate x_3 with 6 uniformly drawn levels. For levels 1 and 2 of feature x_3, the log odd ratio of y is given by $\theta_1 x_1 + \theta_2 x_2$ where $\theta_1 = -1$ and $\theta_2 = 0.5$. For levels 3 and 4, we have $\theta_1 = -0.5$ and $\theta_2 = 1.5$ and finally for levels 5 and 6, we set $\theta_1 = 1$ and $\theta_2 = -0.5$. This data generating mechanism is illustrated in Figure 5.13. Results of various clustering methods developed in this chapter are given in Table 5.2.

For both experiments, the SEM algorithm is initialized randomly with $K = 5$ segments. In experiment (a), the proposed approach selects effectively no partitions. The *maximum a posteriori* scheme of Equation (5.6) is able, as argued in Section 5.4.4, to make segments "vanish" and explore segmentations with less than K segments. In experiment (b), the proposed approach is able to recover the tree structure. Consequently, the proposed algorithm yields the best performance in both settings. As for FAMD and PLS which resulting projections for experiment (a) are displayed on Figure 5.14 and 5.15 respectively, they form 3 clusters and consequently the 3 resulting logistic regression suffer from a higher estimation variance loosely reflected in their inferior performance in Table 5.1. LMT recovers the truth by producing a single logistic regression but not MOB (see Figure 5.17) which splits the data into 2 segments. On experiment (b), FAMD produces worse results than a single logistic regression and the benefit of using the target y is clear from the result of PLS (see Table 5.2). MOB also recovers the true structure (see Figure 5.18) but not LMT which first splitting node is a continuous feature not involved in the data generating mechanism of the segments as displayed on Figure 5.16. For both experiments, it would be useful to report confidence intervals and or bar plots as was done in Chapters 3 and 4 to derive an empirical consistency of the proposed approach.

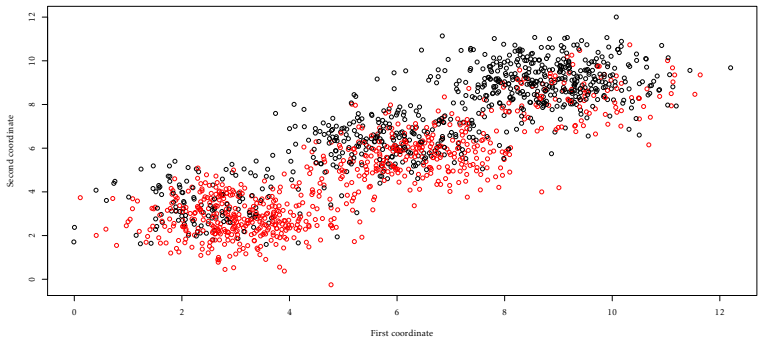
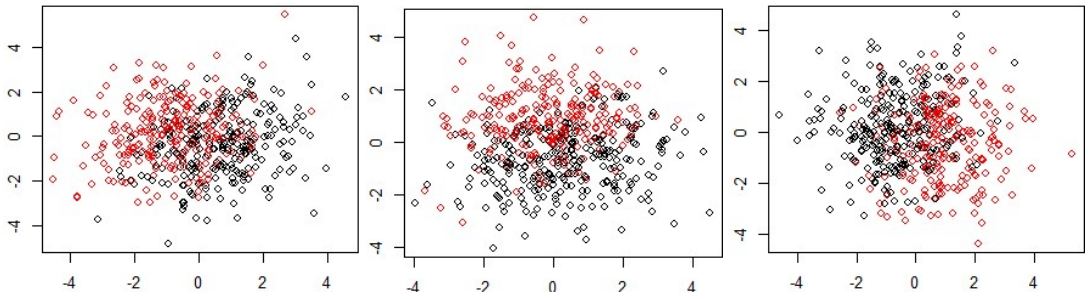


Figure 5.12 – Cloud points of simulated data from (a) with respective labels in red and black.



(a) Representation of y w.r.t. (x_1, x_2) for $x_3 \in \{1, 2\}$.

(b) Representation of y w.r.t. (x_1, x_2) for $x_3 \in \{3, 4\}$.

(c) Representation of y w.r.t. (x_1, x_2) for $x_3 \in \{5, 6\}$.

Figure 5.13 – Cloud points of simulated data from (b) with respective labels in red and black.

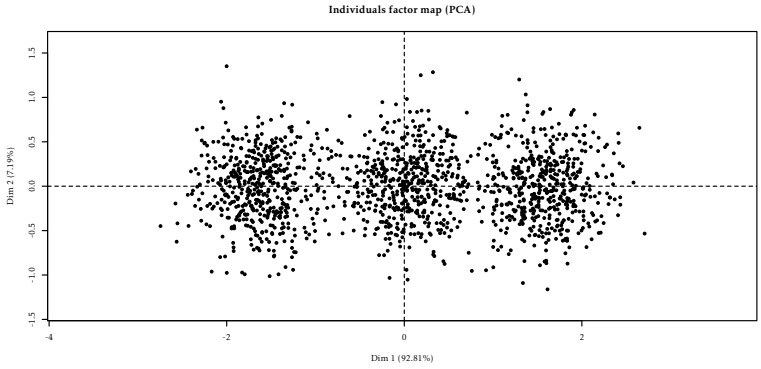


Figure 5.14 – Cloud points of simulated data from (a) after applying the PCA algorithm.

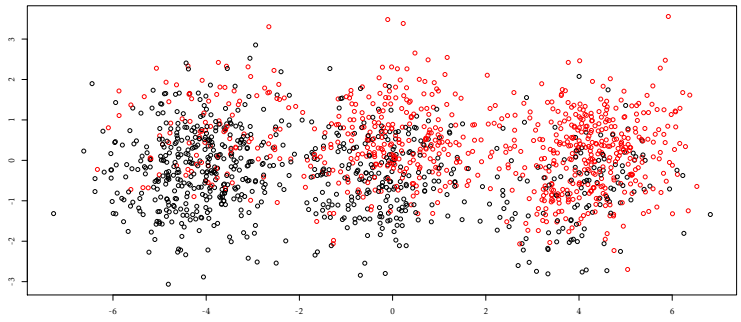


Figure 5.15 – Cloud points of simulated data from (a) with respective labels in red and black after applying the PLS algorithm.

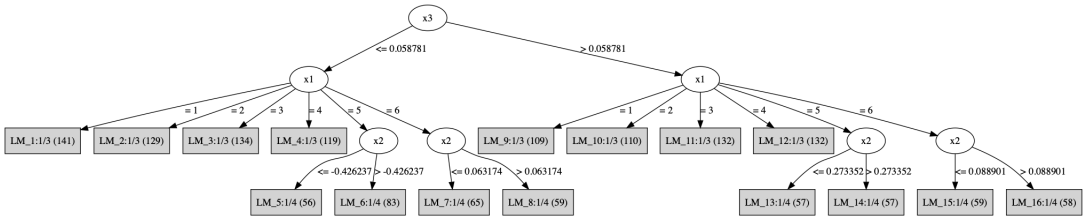


Figure 5.16 – LMT tree resulting from simulated data from (b).

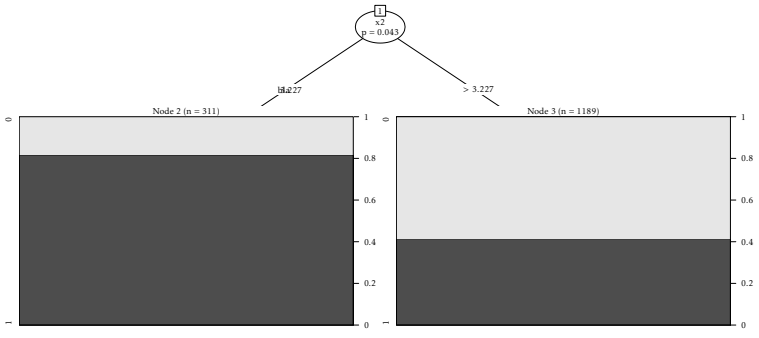


Figure 5.17 – MOB tree resulting from simulated data from (a).

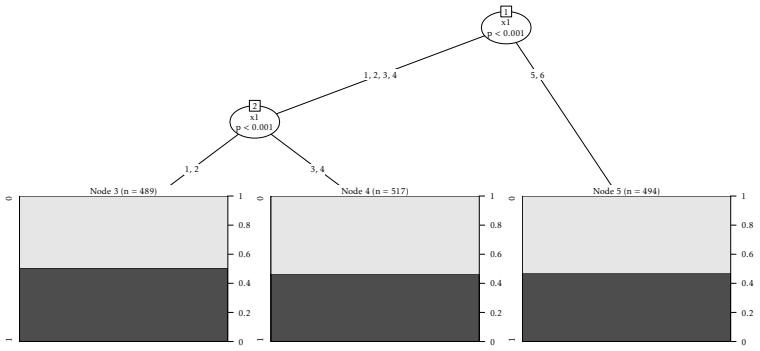


Figure 5.18 – MOB tree resulting from simulated data from (b).

Table 5.2 – Comparison of several clustering approaches w.r.t. the subsequent predictive performance in experiment (b).

	Oracle	ALLR	*glmtree*-SEM	FAMD	PLS	LMT	MOB
Gini	69.7	25.8	**69.7**	17.7	48.4	65.8	**69.7**

5.6.2 Benchmark on *Credit Scoring* data

The running example: the Automobile dataset

Recall from Sections 5.1.2 and 5.2.1 that PCA, MCA, FAMD and PLS revealed no segments on this dataset and from Section 5.2.2 that LMT produced disappointing results and MOB could not be tested. A logistic regression learnt on 70 % of the sample yields an overall performance of 57.9 Gini points.

By applying *glmtree*-SEM to the Automobile dataset, we get $\hat{K} = 2$ segments defined by the value of the car being over 10,000 Euros, yielding an overall performance of 58.7 Gini points. This difference might not seem significant, but in the *Credit Scoring* industry, and as was argued in Chapter 3 and 4 to motivate quantization and interaction screening, such small improvements might result in the selection of a few more good applicants (resp. a few less bad applicants) whose car loans are of high amount. It is thus a high stake for financial institutions.

One year of financed applications

A subset of all applications, representative of approx. 30 portfolios with different scorecards, has been extracted for the purpose of the present benchmark with $n = 900{,}000$ observations and $d = 18$ among which 12 continuous features and 6 categorical features with 6 to 100 levels (most features are similar to the Automobile dataset). The missing values have been preprocessed such that no continuous features have missing values and the categorical features have a separate and meaningful "missing" level. Also for simplification purposes, no quantization or interaction screening is performed so that the SEM algorithm is conducted as presented in Section 5.4.3.

Generative approaches (FAMD and PLS) are not used due to their subjectivity (visual separation) and the fact that they are used by practitioners to provide "local" segments (*e.g.* for the Automobile market). Hence for such a large dataset, they would have to be applied "recursively" (applying FAMD / PLS on each of the resulting visually separated segments). For computational

Table 5.3 – Comparison of the existing segmentation and the proposed approach *glmtree*-SEM.

Current segmentation $K = 30$		*glmtree*-SEM $K = 10$
Current performance *via* Platt scaling	ALLR	ALLR
54.6	52.0	50.2

reasons that became apparent in applying LMT and MOB to the Automobile dataset, these methods cannot cope either with this larger dataset.

Consequently, *glmtree*-SEM is only compared to the current performance. The combined scorecards have an overall performance of approximately 46 Gini points but they are not on the same "scale" since they were developed at different times. I rely on the Platt scaling method developed in [14] and [19] and used in common *machine learning* libraries such as Scikit-learn, to put all of them on the same scale by fitting a logistic regression between the observed labels $\mathbf{y}$ and the scores outputted by each scorecard. After this procedure, overall performance jumps to approximately 54.6 Gini points.

Another possible benchmark with the current segmentation is to learn additive linear logistic regression (hereafter ALLR as in Chapter 3 for each existing segment. This approach leads to an overall Gini fo 52. As our proposed algorithm *glmtree*-SEM will be applied without quantization or interaction screening, this result is a "fairer" baseline than the preceding one since they are both in the same model space: additive linear logistic regression trees.

The *glmtree*-SEM applied to this big dataset yields only $\hat{K} = 12$ segments for an overall performance of 50.2 Gini points. This is satisfactory in two aspects: the performance is relatively close to the existing segmentation while using 3 times less segments / logistic regressions. We can hope for even better interpretability and performance by performing quantization and interaction screening. Results are summarized in Table 5.3.

5.7 Conclusion

This chapter aimed at formalizing an old problem in *Credit Scoring*, providing a literature review as well as a research idea for future work. As is often the case, practitioners have had good intuitions to deal with practical and theoretical requirements, such as performing clustering techniques, choosing segments empirically from the resulting visualization and fitting logistic regression on these.

However, situations can easily be imagined where such practices can fail, which is why other existing methods, that take into account the predictive task, were exemplified. Nevertheless, as in the best case scenario, practitioners would like to have an all-in-one tool that works with missing values and eventually performs quantization and interaction screening while guaranteeing the best predictive performance by embedding the learning of a segmentation in the predictive task of learning its logistic regression, a new method is proposed, based on an SEM algorithm, that was adapted to be usable with the *glmdisc* method developed in the two preceding chapters.

On simulated data, it shows very promising results that aims at demonstrating the consistency of the approach. On real data from CACF, other methods yielded disappointing results while *glmtree*-SEM was able to compete with the current performance which required months of manual adjustments. By adding the quantization and interaction screening ability to this algorithm, as described in Section 5.5, we could easily imagine beating this *ad hoc* segmentation by a significant margin.

References of Chapter 5

[1] Eric Bair et al. « Prediction by supervised principal components ». In: *Journal of the American Statistical Association* 101.473 (2006), pp. 119–137.

[2] Leo Breiman et al. *Classification and Regression Trees*. Statistics/Probability Series. Belmont, California, U.S.A.: Wadsworth Publishing Company, 1984.

[3] Kin-Yee Chan and Wei-Yin Loh. « LOTUS: An algorithm for building accurate and comprehensible logistic regression trees ». In: *Journal of Computational and Graphical Statistics* 13.4 (2004), pp. 826–852.

[4] Arthur P Dempster, Nan M Laird, and Donald B Rubin. « Maximum likelihood from incomplete data via the EM algorithm ». In: *Journal of the royal statistical society. Series B (methodological)* (1977), pp. 1–38.

[5] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*. Vol. 1. 10. Springer series in statistics New York, NY, USA: 2001.

[6] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. « Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors) ». In: *The annals of statistics* 28.2 (2000), pp. 337–407.

[7] Torsten Hothorn, Kurt Hornik, and Achim Zeileis. « Unbiased recursive partitioning: A conditional inference framework ». In: *Journal of Computational and Graphical statistics* 15.3 (2006), pp. 651–674.

[8] Michael I Jordan and Robert A Jacobs. « Hierarchical mixtures of experts and the EM algorithm ». In: *Neural computation* 6.2 (1994), pp. 181–214.

[9] Niels Landwehr, Mark Hall, and Eibe Frank. « Logistic model trees ». In: *Machine learning* 59.1-2 (2005), pp. 161–205.

[10] Sébastien Lê, Julie Josse, and François Husson. « FactoMineR: An R Package for Multivariate Analysis ». In: *Journal of Statistical Software, Articles* 25.1 (2008), pp. 1–18. ISSN: 1548-7660. DOI: 10.18637/jss.v025.i01. URL: https://www.jstatsoft.org/v025/i01.

[11] Ludovic Lebart, Alain Morineau, and Marie Piron. *Statistique exploratoire multidimensionnelle*. Vol. 3. Dunod Paris, 1995.

[12] David Opitz and Richard Maclin. « Popular ensemble methods: An empirical study ». In: *Journal of artificial intelligence research* 11 (1999), pp. 169–198.

[13] Jérôme Pagès. *Multiple factor analysis by example using R*. Chapman and Hall/CRC, 2014.

[14] John Platt et al. « Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods ». In: *Advances in large margin classifiers* 10.3 (1999), pp. 61–74.

[15] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[16] William Robson Schwartz et al. « Human detection using partial least squares analysis ». In: *2009 IEEE 12th International Conference on Computer Vision (ICCV)*. IEEE. 2009, pp. 24–31.

[17] Marc Sumner, Eibe Frank, and Mark Hall. « Speeding up logistic model tree induction ». In: *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer. 2005, pp. 675–683.

[18] Svante Wold et al. « The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses ». In: *SIAM Journal on Scientific and Statistical Computing* 5.3 (1984), pp. 735–743.

[19] Bianca Zadrozny and Charles Elkan. « Transforming classifier scores into accurate multiclass probability estimates ». In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2002, pp. 694–699.

[20] Achim Zeileis, Torsten Hothorn, and Kurt Hornik. « Model-based recursive partitioning ». In: *Journal of Computational and Graphical Statistics* 17.2 (2008), pp. 492–514.

Conclusion and prospects

It's not complicated, it's just a lot of it.

Richard Feynman, interview for *The World from Another Point of View*, Yorkshire Television, 1972.

Sommaire

Various sources estimate the growth of created data to be exponential. However, the difficulty of processing these data has superseded the difficulty of storing them: "data is the new oil" is the catch-phrase often repeated in industry. While this oil has been extensively extracted and stored in a lot of application contexts, including *Credit Scoring*, there is not always a motor capable of burning it. *Scalability* refers to the problem of applying an existing method to increasingly more data. It turns out that, either by lack of computing power and / or by statistical properties or assumptions not met, not all methods are scalable. Consequently, the statistics and machine learning communities have already tackled lots of problems stemming specifically from large n and / or large d settings. These problems form a vast literature and are out of the scope of the present work. The aim of these concluding remarks is to give a concise context of high-dimensional data w.r.t. the *Credit Scoring* industry, what problems does it give rise to, and some simple existing solutions from an eluded literature review.

Motivation

This first section aims at presenting the data currently collected but remaining unused in the context of *Credit Scoring* and the two sub-problems that were identified and tackled in this chapter.

Industrial context

Technological advancements in big data storage and processing has sparked interest in exploiting these for *Credit Scoring*, although most hereafter presented data sources were available for quite some time...

Payment data Once a loan has been granted, monthly payments due by clients are most of the time debited from their main bank account. These debits might be accepted or refused by their bank depending on their balance and several tries might be performed before going into a recovery process. Such data are presented on Table 5.4.

Table 5.4 – Payment data.

Client	Date	Should pay	Has paid	Type	Outstanding	Status
1	05/01/2019:10:00:00	50	0	Automatic debit	5,000	Refused
1	08/01/2019:10:00:00	50	50	Automatic debit	4,950	Accepted
1	05/02/2019:10:00:00	50	0	Automatic debit	4,950	Refused
1	08/02/2019:10:00:00	50	0	Automatic debit	4,950	Refused

Recovery data In the case of Client 1 from the previous example in Table 5.4, once the second automatic debit is refused, it enters a recovery process that can be long and complex and is way out of the scope of the present manuscript. It creates however tremendous amounts of data, that could be used in the context of *Credit Scoring*, *e.g.* for better assessment of the class to predict (good / bad borrower) or as predictive features for a known client that applies for another loan.

Table 5.5 – Monthly per-client recovery data.

Client	Date	Should pay	Fees	Has paid	Outstanding	Status
1	09/02/2019:11:24:12	50	10	0	4,960	Manual recovery by phone
1	09/02/2019:11:26:09	60	0	60	4,900	Debit card payment

Credit card data Transactions from credit card holders are recorded and can easily be retrieved. They are well-structured but contain lots of text fields, as exemplified on Table 5.6.

Table 5.6 – Daily per-client credit card data.

Client	Date	Amount	Company	Location	Category	...
1	01/01/2019:09:05:18	10.9	Amazon	Online	Online retail	...
1	01/01/2019:12:50:25	14.5	Les 3 Brasseurs	22 Place de la Gare, 59800 LILLE	Restaurant	...
1	02/01/2019:19:10:20	78.9	Carrefour	1 Avenue Willy Brandt, 59000 LILLE	Retail consumer goods	...

Log data In the same fashion as social media users are targeted with personalized ads thanks to their visitation pattern [10], connexion logs can be used to personalize the loan offer in terms of amount, rate, ... An example is given in Table 5.7.

Marketing data Finally, clients often apply to loans after having been exposed to diverse forms of adverts, some of which can be properly recorded and affected to a client, *e.g.* mailing or e-mailing campaigns, Google AdWords, etc. An example of such data is visible on Table 5.8.

Table 5.7 – Log data.

Client	Platform	Device	Date	URL
1	Leboncoin	MAC OS	10/12/2018:22:33:50	/leboncoin/Nord/Electromenager/
1	Main site	MAC OS	10/12/2018:22:34:10	/sofinco/home
1	Main site	MAC OS	10/12/2018:22:34:30	/sofinco/perso/electromenager
1	Main site	MAC OS	10/12/2018:22:35:12	/sofinco/simulation

These data can be very informative of the class (good / bad borrower) of each client: a prospective client coming from AdWords in the middle of the night might be riskier than a targeted prospect via an email on a week-end afternoon for example.

Table 5.8 – Marketing data.

Client	Marketing lever	Date	Device	Opened	Visited	URL
1	email	11/12/2018:15:02:54	Android	Yes	No	/media/new_credit_ad/car_loan&id=1&…
1	mail	12/12/2018:10:00:00	NA	NA	NA	NA
1	Google Adword	13/12/2019:12:10:10	Windows	NA	Yes	/adword/personal_credit&id=1&…

All these kinds of data are not directly used by CACF in its *Credit Scoring* practices, although by simply looking at the exemplary tables, one is able to draw simple intuitions of signals of low / high risk of default. In the subsequent section, two problems pertaining the usage of these data, justifying in a sense why they were not used to this day, are identified and formalized.

Two identified sub-problems

A very simple way of dealing with all examples of additional data of the preceding Section is to add them as columns of our "traditional" data (displayed in Figure 1.1 of Chapter 1 for example). Taking Table 5.8 as an example, each marketing contact with the client can be reshaped so as to fit in separate columns relative to contact #1, contact #2, etc. This would yield Table 5.9 and we could easily imagine appending to it log, credit card, recovery and payment data in a similar way. As a consequence, we are artificially back to a traditional setting with a very high number of covariates d. This setting is the subject of the next section.

A probably clever way to use these data is to exploit their temporal structure, just as Recurrent Neural Networks have been able, on *e.g.* sentiment analysis problems by analysing raw text, to perform better than methods not making use of this structure and requiring manual pre-processing such as n-grams [8]. However, such methods are hardly interpretable [7], which forced practitioners to resort to manual, intractable feature engineering, such as counting the number of credit card transactions over various time periods, which serve as inputs to simple models such as logistic regression. To avoid this time-consuming task without harming the interpretability of the resulting model nor its performance, and similarly to the quantization approach developed in Chapter 3, suitable structured representations of these data have to be automatically extracted. Some techniques are provided in the subsequent section.

Table 5.9 – Long data.

Client	Job	Children	Marketing lever 1	Date 1	Device 1	Marketing lever 2	Date 2	Device 2	…
1	Skilled worker	1	email	02/03/2019:15:02:54	Android	Google Adword	04/03/2019:12:01:01	Windows	…
2	Technician	3	mail	02/04/2019:10:00:00	NA	NA	NA	…	
3	Executive	0	Google Adword	15/04/2019:12:10:10	Windows	mail	01/05/2019:10:00:00	NA	…

Longitudinal data in high dimension

This section is an eluded literature review of problems that arise in high dimension for "classical" data. It was first tackled by bio-statisticians working with omics data, such as DNA that can span over thousands of features for each patient, which yields a situation where more features than observations are available!

Remark on the $d > n$ setting

A lot of work has been dedicated to this setting (see [2] for a review) since a lot of classical statistical methods do not work out-of-the box, including logistic regression. Independently from their relative consistency properties on selecting the "best" features, penalization methods naturally select at most d features (whatever the amount of penalization) [11] such that we can assume in what follows $d \leq n$.

In the next section, we review the statistical properties associated with the "curse of dimensionality", a term attributed to Bellman:

> All [problems due to high dimension] may be subsumed under the heading "the curse of dimensionality". Since this is a curse, [...], there is no need to feel discouraged about the possibility of obtaining significant results despite it.
>
> R. Bellman, "Dynamic programming", 1957

The curse of dimensionality

The foundation of statistics is that by having enough observations of random variables, we can approximate well (possibly intractable) integrals of their (possibly unknown) pdf by an empirical average. Thus, a major problem in high dimensions is their relative emptiness, which makes the use of averages obsolete. Two classical examples are usually given: first, suppose we have data that live in $[0,1]^d$ and you want to cover a neighbourhood of the origin of volume $v < 1$, *e.g.* to perform nearest-neighbours classification. You want to know which fraction s of each dimension needs to be covered. This fraction is given by $s = v^{1/d}$, such that for example a volume $v = 0.5$ on a square is covered by $s = \sqrt{2}/2 \approx 0.71$, on a cube by $s = \sqrt[3]{0.5} \approx 0.79$, and so on. In high dimensional spaces such that $d >> 1$, this fraction s is approximately 1 for every $v > \epsilon$: hence, neighbourhoods are not local anymore. Second, and somewhat subsequently to this first remark, the expected squared euclidean distance between two independent variables drawn uniformly in $[0,1]^d$ is $d/6$ as illustrated on Figure 5.19. Consequently, it is often concluded that high dimension spaces are "empty" since points are all far away from each others.

The blessings of dimensionality

The relative "emptiness" of the feature space $\mathcal{X}$ in high dimension is not solely a curse: recall that we motivated the use of logistic regression, quantization and logistic regression trees for their interpretability. We would like simple, linear boundaries between good and bad borrowers. As should now be apparent from the few Gini figures given in this manuscript, such boundaries do not exist in *Credit Scoring* in small dimension which motivates the use of additional data. The higher the dimension d, the higher the likelihood of existence of a linear boundary between good and bad borrowers [5]. However, if some feature are just "noise" w.r.t. the class or contain redundant information, this linear boundary is highly likely to not generalize well, *i.e.* overfit [3].

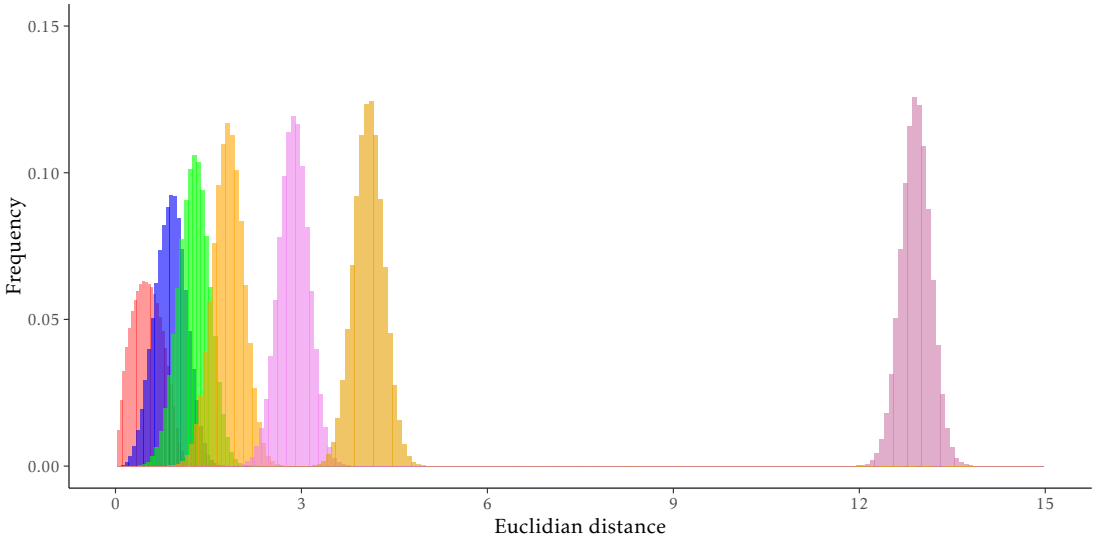


Figure 5.19 – Distribution of the Euclidean distance between two random points of $[0,1]^d$ w.r.t. the dimension of the space $d \in \{2, 5, 10, 20, 50, 100, 1000\}$.

To avoid this pitfall but still benefit from the vast amount of available data, a simple solution is to reduce the dimensionality d of the data to only "useful" dimensions (in the sense of the predictive task).

Dimension reduction

A straightforward way of avoiding the curse(s) of dimensionality for logistic regression is to get back to a small dimension d' relatively to n by pre-processing the d features. In Chapter 3, and particularly Section 3.1, it was argued that quantization could be thought of as a dimensionality reduction technique, because information was compressed in intervals and "meta"-groups for continuous and categorical features respectively without affecting predictive power (on the contrary!). Two way more classical ways of performing dimensionality reduction are presented here: combining original features in principal components, which was already discussed in Chapter 5 when building segments of clients, and feature selection, which are the subjects of the two subsequent sections respectively.

By combining input features

Various algorithms were designed to map features onto a new "representation" which can have interesting properties. In Chapter 5, we discussed at great length of FAMD which consists in constructing orthogonal principal components, ranked by their respective eigen value which corresponds to the proportion of the total explained variance. In the same fashion as *Credit Scoring* practitioners only care about the first two axes when looking for potential segments (see Chapter 5), one could discard many dimensions of its original data by considering only the principal components that explain at least a given proportion of the total variance.

However, as was sensed in Chapter 5 for segmentation, the goal remains to predict a class, not to account for most of the variance of the predictors! These two goals being possibly disjoint,

we introduced the PLS and SPC algorithms, which aim to take into account the target feature. Moreover, the SPC algorithm is an iterative procedure to select only principal components that have predictive power.

By selecting input features

Practitioners are often not convinced by the preceding approach since the new representation of the data is hard to grasp. Subsequently, feature selection approaches, which remain in the canonical space, are usually preferred. Such algorithms are out of the scope of the present manuscript, although the LASSO was mentioned earlier. This penalization method performs feature selection as a side effect and one of its refinement, the adaptive LASSO [12], has strong oracle properties (*e.g.* it finds the subset of the features that participate in the true model, if it exists) which are appealing to both academic and industrial practitioners.

New data types in a supervised classification setting

In essence, it is not solely the volume of data that has to be addressed, but the variety of its format, as is apparent from the motivational section. These unstructured data require specific modelling techniques, ideally to automatically extract their predictive information into inputs that can be processed by simple interpretable models like logistic regression. Internally at CACF, some simple features are extracted, typically from the credit card transactions, and seems to be the case in other financial institutions [9]. In this applied work, the author compares the "traditional" approach to a model exploiting only similarities between clients' credit card transactions, achieving a good overall performance, in particular in conjunction with traditional data. This is not the case at CACF where attempts to use only web logs showed poor performance. This empirical study motivated CACF to study ways to structure these data and extract only the most important predictive information.

Is this structuring work a result of some statistical procedure or shall it remain a manual feature engineering work done by field experts? This question has found a clear answer in favour of automatic statistical procedures in fields like Computer Vision and Speech Recognition where neural networks, which are motivated by their automatic feature engineering, have made tremendous improvements over previous approaches relying on manual feature engineering [4]. These models are not directly applicable to *Credit Scoring* since we require interpretability through simple models like logistic regression. Applying techniques from metric learning [1] and functional principal components [6] are future areas of research, since credit card transactions can be reduced to a similarity measure as in [9] and web visitation patterns can be seen as functional categorical data (where categories are web pages) that can be summarized by (functional) principal components.

Conclusion générale

Cette thèse a permis d'explorer cinq sujets directement inspirés de problématiques industrielles de *Credit Scoring*, sans doute graduellement du plus opérationnel, dont le questionnement était parfaitement posé, la "réintégration des refusés", au plus ouvert, l'utilisation de données non structurées en grande dimension, pour laquelle il ne semble pas y avoir d'approche universelle existante. On passe rapidement en revue ces problèmes en donnant les idées clés du problème, de sa résolution et des contributions de cette thèse.

Le chapitre 2 consacré à la "réintégration des refusés" a permis de poser un problème ancien de l'industrie du crédit à la consommation : l'ensemble d'apprentissage de la règle de classement bons / mauvais payeurs est un échantillon de la population ayant déjà été financée. Ce financement est fortement corrélé à plusieurs règles existantes destinées à ne financer que des clients supposés bons. Cela induit-il un biais dans l'estimation des modèles de classification supervisée, en particulier la régression logistique ? En réinterprétant la classe des clients non financés comme des données manquantes, et en distinguant les cas du vrai (*well-specified*) et du mauvais (*misspecified*) modèle, on a montré que le paramètre de la régression logistique peut en effet être biaisé. Néanmoins, en reformulant les techniques *ad hoc* d'utilisation des informations des clients non financés comme des tentatives de modélisation du mécanisme de financement, on a montré que la méthode actuelle consistant à n'utiliser que les clients financés pour lesquels $Z = \mathrm{f}$ était satisfaisante.

Rassuré sur la pertinence de l'échantillon d'apprentissage, le praticien poursuit ses travaux par certains pré-traitements, qui ont une justification pratique mais aussi théorique : apprendre une "meilleure" représentation des données au sens de l'interprétation du modèle mais aussi de sa qualité prédictive. La quantification (*quantization*) regroupe la discrétisation de prédicteurs continus (la transformation d'un âge en une tranche d'âge par exemple) et le regroupement de modalités de prédicteurs catégoriels (le regroupement de modèles de véhicule en segments comme les citadines, routières, *etc*.). Ce pré-traitement manuel est à faible valeur ajoutée pour le statisticien et lui prend un temps considérable, qui tend à augmenter (du fait de l'augmentation du nombre de prédicteurs) ; de plus, en reposant sur des méthodes *ad hoc* et univariées, la qualité prédictive du modèle résultant est diminuée. Il s'agissait alors de formaliser ce problème et de proposer une automatisation qui faisait néanmoins sens du point de vue statistique. Une nouvelle méthode, *glmdisc*, est proposée, ainsi que deux stratégies d'estimation différentes, dont les résultats sur données réelles sont meilleurs que les approches *ad hoc* susmentionnées et les méthodes d'état de l'art.

De manière similaire, pour des raisons pratiques et théoriques, il est courant d'étudier des croisements (*pairwise interactions*) de variables : on suppose que l'effet combiné de deux prédicteurs sur le risque du client est différent de la somme des effets de ces prédicteurs. Encore une fois, des techniques *ad hoc*, sous-optimales, étaient employées, nécessitant des données préalablement quantifiées. Or, la quantification et l'introduction d'interactions, en agissant sur l'espace des modèles considérés, doivent être effectuées simultanément. Une approche de type MCMC, utilisant l'algorithme de Metropolis-Hastings, a été proposée pour l'introduction d'interactions et dont l'intérêt principal est l'utilisation aisée en combinaison de l'algorithme *glmdisc* construit pour le problème de quantification. Il est alors possible d'obtenir une régression logistique performante et interprétable en quelques heures de temps machine, ce qui nécessitait un à deux mois de temps humain.

Nous avions ensuite pris du recul sur le quotidien du praticien en *Credit Scoring* qui se voit confier des scores et / ou des améliorations "locales" du système d'acceptation, c'est-à-dire ne concernant qu'une partie de la population totale des demandeurs de crédit. En effet, ledit système est bien souvent composé de nombreuses règles "métier" (écartées de cette étude) mais surtout de nombreux scores, c'est-à-dire des régressions logistiques utilisant des variables différentes, des quantifications et croisements différents, et utilisées sur des clientèles différentes. En ne remettant jamais en cause la structure d'arbre du système d'acceptation total, la qualité prédictive est nécessairement sous-optimale, ce qui nous a conduit à présenter les méthodes actuelles utilisées en industrie pour construire des segments sur lesquelles différentes régressions logistiques sont ensuite construites. Plusieurs méthodes alternatives de l'état de l'art, très simples à mettre en oeuvre et qui produisent de meilleurs résultats que l'approche actuelle

sur des données simulées, ainsi qu'une piste de résolution, sous la forme d'un algorithme SEM comparable à celui exploité dans *glmdisc* ont été passées en revue et comparées sur des données simulées et réelles et montrent de bons résultats préliminaires.

Enfin, tous ces travaux exploitaient des données dites "classiques" en *Credit Scoring*, c'est-à-dire majoritairement issues de formulaires remplis par le client ou par le vendeur (en magasin). CACF dispose par ailleurs d'autres données, dont l'intérêt, la capacité prédictive additionnelle en tête de liste, reste à démontrer, comme par exemple les données transactionnelles de cartes de crédit, les données de log de connexion au site internet, les données marketing, *etc*. La présente conclusion a été l'occasion de voir les problèmes classiques liés à l'augmentation de dimension : des espaces vides où la notion de "voisin" ne fait pas toujours sens, mais où il est plus facile de trouver de simples hyperplans séparant les classes bons et mauvais payeurs, pourvu que toutes ces nouvelles dimensions apportent de l'information. On s'est ensuite attardé sur les données non structurées à la disposition des banques dont la granularité fine (des centaines de transactions de carte bancaire pour un seul client) conduit le praticien, une fois de plus, à s'engouffrer dans des techniques *ad hoc*, manuelles et chronophages d'agrégation, sans garantie statistique. La littérature relative à ces nouvelles données a été synthétisée de manière à favoriser la diffusion de ces bonnes pratiques ; l'application de ces techniques sur les données réelles de CACF est un futur axe de travail.

Pour conclure, cette thèse a permis d'apporter des réponses théoriques à des problèmes récurrents connexes au *Credit Scoring* et nécessitant un tel travail de formalisation. Elle a également permis, s'agissant d'une thèse CIFRE, d'apporter une solution pratique aux problèmes de quantification et de croisements de variables sous la forme de deux solutions logicielles. Le chapitre 1 a permis d'introduire plusieurs problèmes ouverts liés au *Credit Scoring*, parmi lesquels la "segmentation" et l'utilisation de données massives non structurées. Ces deux sujets ont fait l'objet des derniers travaux et ont abouti à une bibliographie épurée ainsi qu'à des simulations donnant une base solide à de futurs travaux. Les perspectives de travaux de recherche applicables au *Credit Scoring* ne se tariront pas de si tôt, tant le contexte de disponibilité de nombreuses sources de données et les enjeux économiques importants sont catalyseurs des besoins de traitements statistiques rigoureux.

Références de la conclusion

[1] Aurélien Bellet, Amaury Habrard et Marc Sebban. *Metric Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2015. doi : 10.2200/S00626ED1V01Y201501AIM030. url : https://doi.org/10.2200/S00626ED1V01Y201501AIM030.

[2] Peter Bühlmann et Sara Van De Geer. *Statistics for high-dimensional data : methods, theory and applications*. Springer Science & Business Media, 2011.

[3] Bertrand Frédéric et al. *Model choice and model aggregation*. Editions Technip, 2017.

[4] Ian Goodfellow et al. *Deep Learning*. T. 1. MIT press Cambridge, 2016.

[5] Alexander N Gorban et Ivan Yu Tyukin. « Blessing of dimensionality : mathematical foundations of the statistical physics of data ». In : *Philosophical Transactions of the Royal Society A : Mathematical, Physical and Engineering Sciences* 376.2118 (2018), p. 20170237.

[6] Peter Hall et Mohammad Hosseini-Nasab. « On properties of functional principal components analysis ». In : *Journal of the Royal Statistical Society : Series B (Statistical Methodology)* 68.1 (2006), p. 109-126. doi : 10.1111/j.1467-9868.2005.00535.x. eprint : https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9868.2005.00535.x. url : https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2005.00535.x.

[7] Yin Lou, Rich Caruana et Johannes Gehrke. « Intelligible models for classification and regression ». In : *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2012, p. 150-158.

[8] Christopher D Manning, Christopher D Manning et Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[9] Joost Verkade. « Credit scoring for small medium enterprises using transaction data ». Mém. de mast. TU Delft, avr. 2018. url : https://repository.tudelft.nl/islandora/object/uuid:6ed89f2f-2c5f-4b85-859b-47a244da609b.

[10] Zhixian Yan et al. « You Are What Apps You Use : Transfer Learning for Personalized Content and Ad Recommendation ». In : *Proceedings of the Eleventh ACM Conference on Recommender Systems*. RecSys '17. Como, Italy : ACM, 2017, p. 350-350. isbn : 978-1-4503-4652-8. doi : 10.1145/3109859.3109923. url : http://doi.acm.org/10.1145/3109859.3109923.

[11] Ji Zhu et Trevor Hastie. « Classification of gene microarrays by penalized logistic regression ». In : *Biostatistics* 5.3 (2004), p. 427-443.

[12] Hui Zou. « The adaptive lasso and its oracle properties ». In : *Journal of the American statistical association* 101.476 (2006), p. 1418-1429.

Appendix A

Algorithms

Sommaire

A.1 Reject inference methods

A.1.1 Fuzzy augmentation

Fuzzy augmentation can be found in [12]; it is the following procedure:

1. Construct Scorecard "Known Good Bad" (KGB) $\hat{\theta}_{\text{f}}$ with financed clients' data (Figure A.1a);
2. Calculate $p_{\hat{\theta}_{\text{f}}}(1|\mathbf{x}_{\text{nf}})$ for rejects (Figure A.1b);

Table A.1 – Example of implementation of the Fuzzy Augmentation method on a small dataset

$\mathbf{y_f}$	$\mathbf{x_f}$
1	0.562
1	0.910
0	0.430

(a) Scorecard $\hat{\boldsymbol{\theta}}_{\mathrm{f}}$ on financed loans

Weight	$\hat{\mathbf{y}}_{\mathbf{nf}}$	$\mathbf{x_{nf}}$
0.68	1	0.347
0.10	1	0.140
0.35	1	0.295

(b) Inferred good not financed loans and their weights

Weight	$\hat{\mathbf{y}}_{\mathbf{nf}}$	$\mathbf{x_{nf}}$
0.32	0	0.347
0.90	0	0.140
0.65	0	0.295

(c) Inferred bad not financed loans and their weights

Weight	y	x
1	0	0.562
1	1	0.910
1	0	0.430
0.68	1	0.347
0.10	1	0.140
0.35	1	0.295
0.32	0	0.347
0.90	0	0.140
0.65	0	0.295

(d) Fuzzy augmented learning dataset

3. Infer rejected client i as good with weight $p_{\hat{\theta}_{\mathrm{f}}}(1|\mathbf{x}_{\mathrm{nf}})$ and as bad with weight $1 - p_{\hat{\theta}_{\mathrm{f}}}(1|\mathbf{x}_{\mathrm{nf}})$ (Figures A.1b and A.1c) ;
4. Calibrate a new scorecard with the "augmented" dataset (Figure A.1d).

Clearly:

$$\forall j = 1,\dots,d,\ \frac{\partial \sum_{i=n+1}^{n'+n} \sum_{y_i=0}^{1} p_{\hat{\theta}_{\mathrm{f}}}(y_i|\boldsymbol{x}_i) \ln(p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i))}{\partial \theta_j} = 0 \Leftrightarrow \theta = \hat{\boldsymbol{\theta}}_{\mathrm{f}},$$

such that:

$$\underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmax}} \sum_{i=n+1}^{n'+n} \sum_{y_i=0}^{1} p_{\hat{\theta}_{\mathrm{f}}}(y_i|\boldsymbol{x}_i) \ln(p_{\boldsymbol{\theta}}(y_i|\boldsymbol{x}_i)) = \hat{\boldsymbol{\theta}}_{\mathrm{f}},$$

and finally:

$$\underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathcal{T}_c) = \underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathcal{T}_{\mathrm{f}}) = \hat{\boldsymbol{\theta}}_{\mathrm{f}}.$$

To conclude, this method will not change the estimated parameters of any discriminant model, asymptotically and with a finite set of observations, regardless of any assumption on the missingness mechanism or the true model hypothesis. In other words, Fuzzy Augmentation has no effect on the KL divergence, making this method useless because it is no different than the financed clients model.

A.1.2 Reclassification

Reclassification can be found in [15], also sometimes referred to as extrapolation as in [2]; it is the following procedure:

1. Construct Scorecard "Known Good Bad" (KGB) $\hat{\boldsymbol{\theta}}_{\mathrm{f}}$ with financed clients' data (Figure A.2a);
2. Calculate $p_{\hat{\theta}_{\mathrm{f}}}(1|\boldsymbol{x})$ for rejects;
3. Infer default status of rejected client i if $p_{\hat{\theta}_{\mathrm{f}}}(1|\boldsymbol{x})$ > threshold; typically threshold = 0.5 (Figure A.2b);
4. Calibrate a new scorecard with the "augmented" dataset (Figure A.2c).

Table A.2 – Example of implementation of the Reclassification method on a small dataset

$\mathbf{y_f}$	$\mathbf{x_f}$
1	0.562
1	0.910
0	0.430

(a) Development of scorecard S^{f} on financed clients

$p_{\hat{\theta}_{\mathrm{f}}}(1\|\mathbf{x})$	$\hat{\mathbf{y}}_{\mathbf{nf}}$	$\mathbf{x_{nf}}$
0.68	1	0.347
0.10	0	0.140
0.35	0	0.295

(b) We force $\mathbf{y}_{\mathrm{nf}} = \mathbf{1}$ if $\mathrm{logit}(S^{\mathrm{f}}(x)) \geq 0.5$

y	**x**
0	0.562
1	0.910
0	0.430
1	0.347
0	0.140
0	0.295

(c) Reclassified learning dataset

Table A.3 – Example of implementation of the Augmentation method on a small dataset

y	**z**	**Score-band**
1	f	1
1	f	1
0	f	1
NA	nf	1
NA	nf	1
NA	nf	1
...	...	...

(a) Calculation of K score-bands on the ACRJ score

Score-band	**Weight**
1	2
...	...
K	1.1

(b) Aggregate the data to estimate the inverse of the probability of being accepted in each score band

Weight	**Score-band**	**y**	**x**
2	1	1	0.123
2	1	0	0.432
2	1	1	0.562
...	...	...	...
1.1	K	0	0.962
1.1	K	0	0.812

(c) Merge weights and data on financed clients to construct the new scorecard

A.1.3 Augmentation

Augmentation can be found in [15]. It is also documented as a "Re-Weighting method" in [8, 2, 12].

1. Construct Scorecard "Accept Reject" (ACRJ) $\hat{\phi}$ with financed clients' data on target variable Z (Figure A.3a);
2. Create K score bands $B_1, \ldots, B_K$ according to $p_{\hat{\phi}}(z|\boldsymbol{x})$;
3. Compute in each score band $\hat{p}(\mathrm{f}|p_{\hat{\phi}}(z|\boldsymbol{x}) \in B_k) = \dfrac{|B_k|}{|\mathrm{F}|}$ (Figure A.3b);
4. Construct a new scorecard on target variable Good/Bad with financed clients' data re-weighted (Figure A.3c).

A.1.4 Twins

The twins method is an internal method at CACF documented in [6] (confidential) where Figure A.1 is given; it consists in the following procedure:

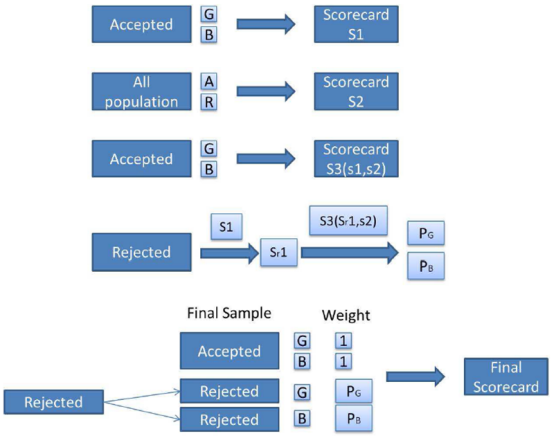


Figure A.1 – The accompanying Figure of the Twins method in the internal documentation.

Table A.4 – Example of implementation of the Twins method on a small dataset

y	x
1	0.562
1	0.910
0	0.430
NA	0.361
NA	0.402
NA	0.294

(a) Development of scorecard $\hat{\theta}_f$ on financed clients

z	x
f	0.562
f	0.910
f	0.430
nf	0.361
nf	0.402
nf	0.294

(b) Development of a scorecard $\hat{\phi}$ on all clients

y	$(1,\mathbf{x}_f)'\hat{\theta}_f$	$(1,\mathbf{x}_f)'\hat{\phi}$
1	1.3	2.5
1	3.1	4.5
0	-0.3	0.4
NA	-1.2	-0.5
NA	-0.4	0.3
NA	-2.0	-2.5

(c) Development of a new scorecard on financed clients

Weight	$\hat{y}_{nf}$	x_{nf}
1	1	0.562
1	1	0.910
1	0	0.430
0.64	0	0.361
0.73	0	0.402
0.44	0	0.294
0.36	1	0.361
0.27	1	0.402
0.37	1	0.294

(d) Inference for not financed clients

1. Develop KGB ("Known Good/Bad") scorecard $\hat{\theta}_f$ on financed clients' data predicting $\mathbf{y}_f$ given $\mathbf{x}_f$ (Figure A.4a);
2. Develop ACRJ (Accept/Reject) scorecard $\hat{\phi}$ on all applicants predicting z given $\mathbf{x}$; this gives us $\hat{\phi}$ (Figure A.4b);
3. Develop a scorecard on financed clients' data predicting $\mathbf{y}_f$ based solely on $(1,\mathbf{x}_f)'\hat{\theta}_f$ and $(1,\mathbf{x}_f)'\hat{\phi}$; this gives us $\hat{\theta}^{\text{twins}}$ (Figure A.4c);
4. Calculate $p_{\hat{\theta}^{\text{twins}}}(1|\boldsymbol{x})$ on rejected applicants and reintegrate them twice in the training dataset like we did with fuzzy augmentation in Section A.1.1 (Figure A.4d);
5. Develop a new scorecard on all applicants' data.

Following notations introduced in Chapter 2, we have:

$$\ell(\boldsymbol{\theta};(\mathbf{1},\mathbf{x}_f)'\hat{\phi},(\mathbf{1},\mathbf{x}_f)'\hat{\theta}_f,\mathbf{y}_f) = \sum_{i=1}^{n} \ln(p_{\boldsymbol{\theta}}(y_i|(1,\boldsymbol{x}_i)'\hat{\theta}_f,(1,\boldsymbol{x}_i)'\hat{\phi})).$$

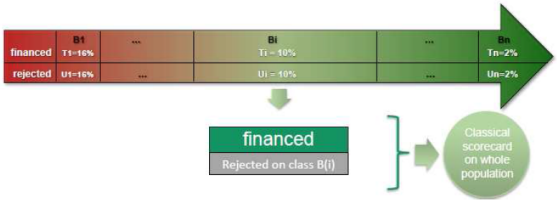


Figure A.2 – The accompanying Figure of the Parceling method in the internal documentation.

Table A.5 – Example of implementation of the Parcelling method on a small dataset

Weight	$\mathbf{y_f}$	$\mathbf{x_f}$
1	1	0.562
1	1	0.910
1	0	0.430

(a) Development of scorecard $p_{\hat{\theta}_f}(1|\boldsymbol{x})$ on financed clients

Score-band	T	U
1	0.5	0.8
...	...	...
K	0.01	0.04

(b) Calculation of $T(k)$ and $U(k)$

Weight	y	x
1	0	0.562
1	1	0.910
1	0	0.430
1	1	0.347
1	0	0.140
1	0	0.295

(c) Inference for not financed clients

We can rewrite $\ell(\boldsymbol{\theta};(\mathbf{1},\mathbf{x}_f)'\hat{\boldsymbol{\phi}},(\mathbf{1},\mathbf{x}_f)'\hat{\boldsymbol{\theta}}_f,\mathbf{y}_f)$ by remarking that the logit of $p_{\boldsymbol{\theta}}(y_i|(1,\boldsymbol{x}_i)'\hat{\boldsymbol{\theta}}_f,(1,\boldsymbol{x}_i)'\hat{\boldsymbol{\phi}})$ is a linear combination of $\boldsymbol{x}$. We know that $\hat{\boldsymbol{\theta}}_f \in \operatorname{argmax}_{\boldsymbol{\theta}\in\Theta}\ell(\boldsymbol{\theta};\mathbf{x}_f,\mathbf{y}_f)$ so that under the identifiability assumption, this method will give the same results as $\hat{\boldsymbol{\theta}}^f$. In terms of KL divergence and as for Fuzzy Augmentation, this method is useless because it is no different than the financed clients model.

A.1.5 Parcelling

Parcelling is a process of reweighing according to the probability of default by score-band that is adjusted by the credit modeler. It has been documented in [8, 2, 15], as well as in [6] where Figure A.2 is given.

1. Construct Scorecard "Known Good Bad" (KGB) $\hat{\boldsymbol{\theta}}_f$ with financed clients' data (Figure A.5a);
2. Create K score bands $B_1,\ldots,B_K$ according to $p_{\hat{\theta}_f}(1|\boldsymbol{x})$;
3. Compute the observed default rate for each band $T(k) = \dfrac{|\text{Bad financed in } B_k|}{|B_k|}$, $1 \le k \le K$;
4. Infer for each band the not financed default rate $U(k) = \epsilon_k T(k)$ where $1 < \epsilon_1 < \ldots < \epsilon_k < \ldots < \epsilon_K$ (Figure A.5b);
5. Reintegrate 2 times each rejected applicant from B_k with weight $U(k)$ as bad and weight $1 - U(k)$ as good, like the Fuzzy Augmentation method in Section A.1.1 (Figure A.5c);
6. Construct the final scorecard.

A.1.6 Simulation of reject inference methods applied to multivariate gaussian data

This early work aimed at comparing several reject inference methods in the well-specified model-case for 8 multivariate Gaussian features on Figure A.3 and 20 features on Figure A.4 in terms of error rate. The horizontal axis represents the cut-off value of a logistic regression that simulates the acceptance / rejection mechanism $p_\phi(z|\boldsymbol{x})$, such that it roughly corresponds to the fraction of missing labels $\mathbf{y}_{\text{nf}}$. In this setting, the semi-supervised generative approach obviously yields better results since the data lies in its restricted hypothesis space and it is able to use the predictors with missing labels $\mathbf{x}_{\text{nf}}$. As explained thoroughly in Chapter 2, standard logistic regression and Augmentation perform similarly (since they are both well-specified models and we are in a MAR setting). Parcelling does not work well since it is designed for a MNAR setting. In presence of well-separated data, Reclassification works well with 20 features since the true decision boundary is "sharp" (see the very small error rate) such that $\text{argmax}_y \hat{p}(y|\boldsymbol{x}) \approx \max_y \hat{p}(y|\boldsymbol{x})$, which is less apparent with 8 features. In both cases, the mean of all features is 0 (resp. 1) if $Y = 0$ (resp. $Y = 1$) and the respective variance matrices are random positive definite matrices (see the Github repository of the manuscript at `https://www.github.com/adimajo/manuscrit_these` to reproduce them).

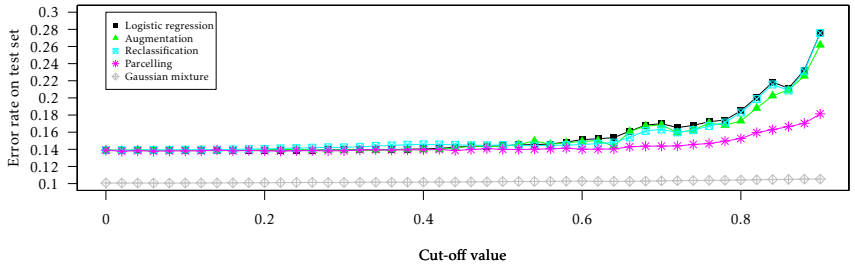


Figure A.3 – Simulation of multivariate 8-dimensional Gaussian features and performance of various reject inference methods including the proposed generative approach.

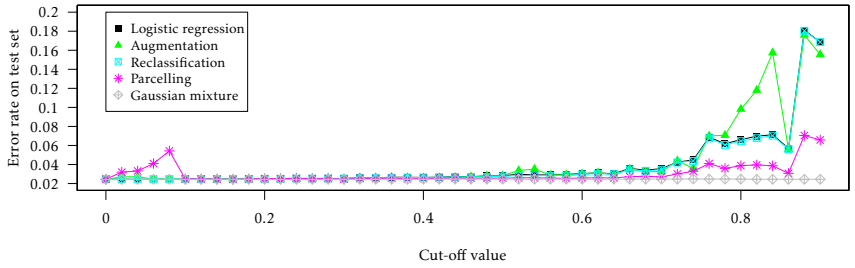


Figure A.4 – Simulation of multivariate 20-dimensional Gaussian features and performance of various reject inference methods including the proposed generative approach.

A.1.7 Performance of other predictive models w.r.t. the acceptance level

Also part of an earlier work, this section's aim was to compare the performance of other "machine learning" models (although we purposely restricted our focus in Chapter 2 to the logistic regression) to see if, when presented with the same data and in presence of a simulated acceptance / rejection mechanism as earlier showcased, it would not be of better interest to switch to a different model, although it was argued in Chapter 2 that these models would not perform good under a MAR setting. The same datasets as in the previous section are used: the proposed models all perform poorer than logistic regression but most importantly their performance drops significantly with the proportion of simulated accepted clients. Some studies of the reject inference problem have focused on these "machine learning" models, see *e.g.* [7].

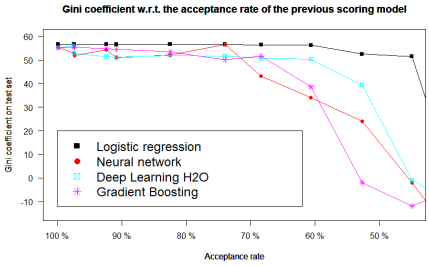


Figure A.5 – Performance resulting from the use of other predictive methods in terms of Gini on an Electronics loans dataset from CACF.

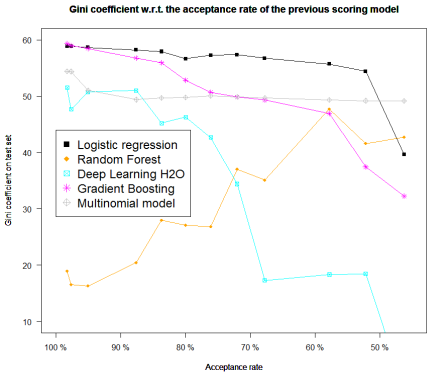


Figure A.6 – Performance resulting from the use of other predictive methods in terms of Gini on a Sports good loans dataset from CACF.

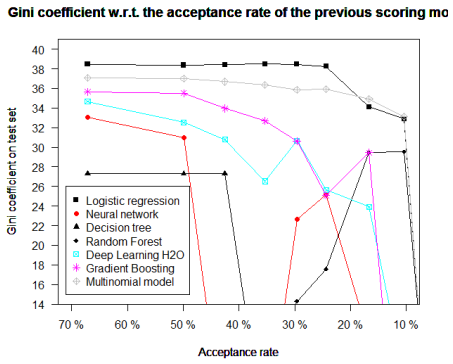


Figure A.7 – Performance resulting from the use of other predictive methods in terms of Gini on a Standard loans dataset from CACF.

A.2 Discretization methods

A.2.1 Unsupervised methods

The *equal-freq* algorithm

The *equal-freq* algorithm 2 is illustrated on Figure A.8.

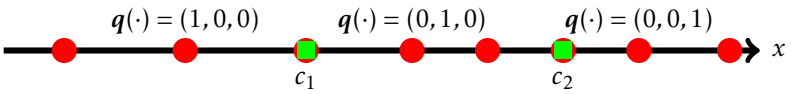


Figure A.8 – Original data in red is discretized using an *equal-freq* procedure resulting in $m = 3$ intervals using the two cutpoints in green.

Data : $n, \mathbf{x}, \boldsymbol{m} = (m_j)_1^d$
Result : $\hat{\boldsymbol{q}}$
for $j = 1$ *to* d **do**
  Sort $\mathbf{x}_j$ by ascending order;
  Let $c_0 = -\infty$, $c_{m_j} = +\infty$ and $c_{j,h} = x_{j,\left\lceil \frac{h \cdot n}{m_j} \right\rceil, j}$;
  Let $C_{j,h} =]c_{j,h-1}; c_{j,h}]$ and $\hat{\boldsymbol{q}}_j(\cdot) = (\hat{q}_{j,h}(\cdot))_1^{m_j}$;
  Set $\hat{q}_{j,h}(\cdot) = \mathbb{1}_{C_{j,h}}(\cdot)$.
end

Algorithm 2 : *equal-freq* discretization: an equal number of training observations are in each bin.

The *equal-length* algorithm

The *equal-length* algorithm 3 is illustrated on Figure A.9.

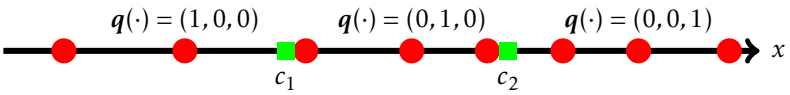


Figure A.9 – Original data in red is discretized using an *equal-length* procedure resulting in $m = 3$ intervals using the two cutpoints in green.

Data : $n, \mathbf{x}, \boldsymbol{m} = (m_j)_1^d$
Result : $\hat{\boldsymbol{q}}$
for $j = 1$ *to* d **do**
  Let $w_j = \max_i x_{i,j} - \min_i x_{i,j}$;
  Let $c_0 = -\infty$, $c_{m_j} = +\infty$ and $c_{j,h} = \frac{w_j \cdot h}{m_j} + \min_i x_{i,j}$;
  Let $C_{j,h} =]c_{j,h-1}; c_{j,h}]$ and $\hat{\boldsymbol{q}}_j(\cdot) = (\hat{q}_{j,h}(\cdot))_1^{m_j}$;
  Set $\hat{q}_{j,h}(\cdot) = \mathbb{1}_{C_{j,h}}(\cdot)$.
end

Algorithm 3 : *equal-length* discretization: each bin has the width of the training set's total support divided by the number of bins.

A.2.2 Supervised univariate methods

The *ChiMerge* algorithm

ChiMerge [9], given in Algorithm 4, is a supervised (it takes into account the labels $\mathbf{y}$) univariate (it does not take into account the other features $\mathbf{x}_{\{-j\}} = (\mathbf{x}_1, \ldots, \mathbf{x}_{j-1}, \mathbf{x}_{j+1}, \ldots, \mathbf{x}_d)$)). It is used indirectly in the benchmarks of Chapter 3 where we applied the same approach to categorical features by computing all the pairwise χ^2 indepedence tests. Here we showcase a rather naïve implementation as a pseudo-code in Algorithm 8, which in practice would perform poorly (because of exponential complexity in the number of categories l_j) but, from an iteration to another, a lot of χ^2 tests are unaffected, so they can be stored rather than computed again. The initial and final steps of *ChiMerge* are illustrated in Table A.6. Refinements of the method, taking into account multiple testing and / or adapting the tests' siognificance level α to each feature, were done in [11, 16, 14, 13].

Table A.6 – p-values of χ^2 tests between subsequent categories on the iris dataset.

Int	Class frequency			χ^2
4.3	16	0	0	4.1
4.9	4	1	1	2.4
5.0	25	5	0	8.6
5.5	2	5	0	2.9
5.6	0	5	1	1.7
5.7	2	5	1	1.8
5.8	1	3	3	2.2
5.9	0	12	7	4.8
6.3	0	6	15	4.1
6.6	0	2	0	3.2
6.7	0	5	10	1.5
7.0	0	1	0	3.6
7.1	0	0	12	

Int	Class frequency			χ^2
4.3	45	6	1	30.9
5.5	4	15	2	6.7
5.8	1	15	10	4.9
6.3	0	14	25	5.9
7.1	0	0	12	

Data : $n, \mathbf{x}, \alpha$
Result : $\hat{\boldsymbol{q}}$
for $j = 1$ *to* d **do**
  $\alpha_{\max} = 1$;
  Sort $\mathbf{x}_j$ in ascending order;
  Let $c_0 = -\infty$, $m_j = n$, $c_{m_j} = +\infty$ and $c_{j,h} = \frac{x_{i,j}+x_{i+1,j}}{2}$ for $1 \leq i \leq n-1$;
  while $\alpha_{max} > \alpha$ **do**
    Let $C_{j,h} =]c_{j,h-1}; c_{j,h}]$ and $\hat{\boldsymbol{q}}_j(\cdot) = (\hat{q}_{j,h}(\cdot))_1^{m_j}$;
    Set $\hat{q}_{j,h}(\cdot) = \mathbb{1}_{C_{j,h}}(\cdot)$;
    for $1 \leq h \leq m_j - 1$ **do**
      $\chi^2_h = \sum_{h'=h}^{h+1} \sum_{y=0}^{1} \frac{\left(\sum_{i=1}^n \mathbb{1}_y(y_i)\hat{q}_{j,h'}(x_{i,j}) - \frac{\sum_{i=1}^n \hat{q}_{j,h'}(x_{i,j}) \times \sum_{i=1}^n \mathbb{1}_y(y_i)}{n}\right)^2}{\frac{\sum_{i=1}^n \hat{q}_{j,h'}(x_{i,j}) \times \sum_{i=1}^n \mathbb{1}_y(y_i)}{n}}$;
    end
    Let $c_{j,\text{argmin}_h \chi^2_h} = \frac{c_{j,h}+c_{j,h+1}}{2}$ and $c_{j,h'} \leftarrow c_{j,h'+1}$ for $\text{argmin}_h \chi^2_h < h' < m_j$;
    Let $m_j \leftarrow m_j - 1$;
    Let $X \sim \chi^2$ and $\alpha_{\max} = \max_h p(X \geq \chi^2_h) = p(X \geq \min_h \chi^2_h)$.
  end
end

Algorithm 4 : The ChiMerge algorithm discretizes features by performing χ^2 tests recursively at a user-defined level α.

The *MDLP* algorithm

The *MDLP* algorithm [3] is an entropy-based discretization method. Contrary to ChiMerge, where at the beginning all distinct values are put into separate categories and thereafter merged (bottom-up method), MDLP recursively calculates the entropy produced by each candidate

cutpoint on their subsequent binary splits. It is reproduced in Algorithm 5.

Data : A continuous feature $\mathbf{x}_j$ which subscript j is dropped in what follows; targets $\mathbf{y}$.
Result : Cutpoints $\mathcal{C}_\star$
Initialize $\mathcal{C}_\star = \varnothing$;
Order $\mathbf{x}$;
Compute the set $\mathcal{I}$ of indices i such that $y_i \neq y_{i+1}$ (contiguous observations which are not of the same class);
Compute the set of candidate cutpoints $\mathcal{C}$ as the mean between these points, *i.e.* $\mathcal{C} = \{\frac{x_i + x_{i+1}}{2} | i \in \mathcal{I}\}$;
Compute the class entropy of each candidate cutpoint $c \in \mathcal{C}$ as:

$$\mathrm{E}(c) = \frac{|\mathbf{x} < c|}{|\mathbf{x}|}\mathrm{Ent}(\mathbf{x} < c) + \frac{|\mathbf{x} > c|}{|\mathbf{x}|}\mathrm{Ent}(\mathbf{x} > c),$$

where $\mathrm{Ent}(\mathbf{x} < c) = -\sum_{y=0}^{1} p(y, \mathbf{x} < c) \ln p(y, \mathbf{x} < c)$ and p is estimated *via* class proportions, *i.e.* $p(y, \mathbf{x} < c) = \frac{|\{i | y_i = y \,\&\, x_i < c\}|}{|\mathbf{x} < c|}$;
Select $c_\star$ which minimizes $\mathrm{E}(c)$ and append it to $\mathcal{C}_\star$;
Repeat steps for $\{\mathbf{x} < c\}$ and $\{\mathbf{x} > c\}$;
Stop when $\mathrm{Gain}(c, \mathbf{x}) = \mathrm{E}(c) - \mathrm{Ent}(\mathbf{x}) \leq \frac{\ln_2(|\mathbf{x}|-1)}{|\mathbf{x}|} + \frac{\Delta(c,\mathbf{x})}{|\mathbf{x}|}$ where $\Delta(c, \mathbf{x}) = \ln_2 7 - (k\mathrm{Ent}(\mathbf{x}) - k_{<c}\mathrm{Ent}(\mathbf{x} < c) - k_{>c}\mathrm{Ent}(\mathbf{x} < c))$ and k, $k_{<c}$, $k_{>c}$ represent the number of classes (1 or 2 in the binary setting) in their respective sample $\mathbf{x}$, $\mathbf{x} < c$ and $\mathbf{x} > c$;

Algorithm 5 : The MDLP algorithm recursively performs discretization with an information gain criterion.

A.2.3 Proposal: *glmdisc*

***glmdisc* with neural networks**

This section describes the *glmdisc*-NN algorithm developed in Chapter 3 and examplifies it on simulated data in Figure A.10.

Data : $((\mathbf{x}_j)_1^d, \mathbf{y}), S, \boldsymbol{m}_{\text{start}}$
Result : $\hat{\boldsymbol{q}}^{(s^\star)}$
Initialization of the network: for each feature feature j, add a softmax with $m_{j,\text{start}}$ outputs which are themselves combined in a sigmoid neuron;
Initialization of the network's weights at random;
$s = 0$;
while $s < S$ **do**
  Perform feed-forward pass of the data: calculate $p_{\boldsymbol{\theta}^{(s)}}(\mathbf{y}|\boldsymbol{q}_{\boldsymbol{\alpha}^{(s)}}(\mathbf{x}))$;
  Perform back-propagation of the error via Stochastic Gradient Descent which yields parameters $(\boldsymbol{\theta}^{(s+1)}, \boldsymbol{\alpha}^{(s+1)})$;
  Compute the *maximum a posteriori* of the hidden layer's representations $\hat{\boldsymbol{q}}^{(s)}(\mathbf{x})$ such that $\hat{q}_{j,h}(x_j) = 1$ if $h = \text{argmax}_{1 \leq h' \leq m_j} q_{\hat{\boldsymbol{\alpha}}_{j,h'}}$, 0 otherwise.;
  Compute the associated logistic regression parameters $\hat{\boldsymbol{\theta}}^{(s)} = \text{argmax}_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \hat{\boldsymbol{q}}_j^{(s)}, \mathbf{y})$;
  $s \leftarrow s + 1$;
end
Choose the best quantization $\hat{\boldsymbol{q}}^{(s^\star)}$, which associated logistic regression parameters yield the lowest BIC criterion;

Algorithm 6 : *glmdisc*-NN: supervised multivariate quantization for logistic regression with neural networks.

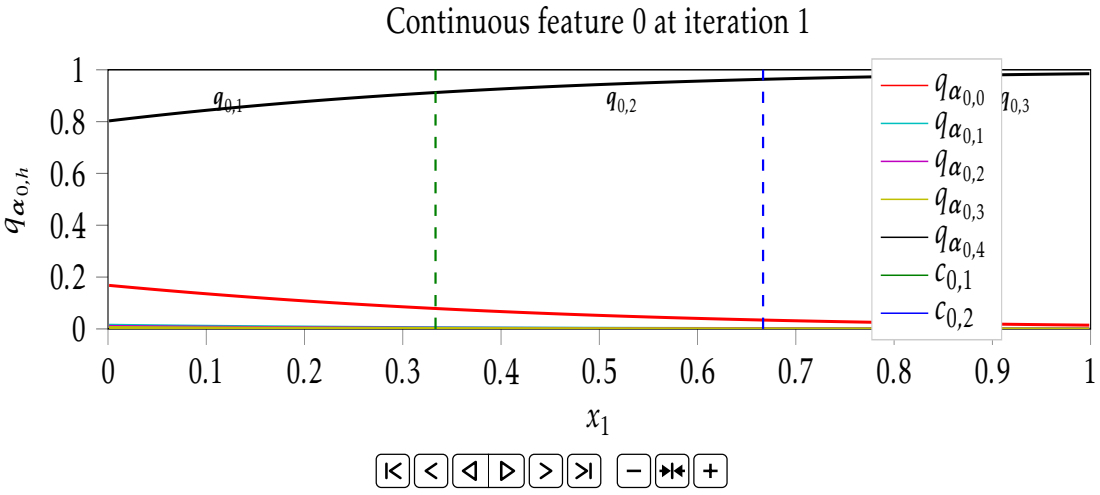


Figure A.10 – Animation of the softmax activation functions $\boldsymbol{q}_{\boldsymbol{\alpha}^{(s)}}$ over the epoch (s).

***glmdisc* with an SEM algorithm**

This section describes the *glmdisc*-SEM algorithm developed in Chapter 3 and examplifies it on simulated data in Figure A.11.

Data : $((\mathbf{x}_j)_1^d, \mathbf{y}), S, \boldsymbol{m}_{\text{start}}$
Result : $\hat{\mathbf{q}}^{(s^\star)}$
Initialization of $\mathbf{q}_j^{(0)}$ at random in $\{1, \dots, m_{j,\text{start}}\}$ and one-hot encode the resulting vector;
$s = 0$;
while $s < S$ **do**
  Adjust logistic regression $\boldsymbol{\theta}^{(s)} = \text{argmax}_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathbf{q}^{(s)}, \mathbf{y})$;
  for $j = 1$ **to** d **do**
    Adjust multinomial logistic regression or contingency tables $\boldsymbol{\alpha}_j^{(s)} = \text{argmax}_{\boldsymbol{\alpha}_j} \ell(\boldsymbol{\alpha}_j; \mathbf{x}_j, \mathbf{q}_j^{(s)})$;
    Draw new latent features $\mathfrak{q}_j^{(s+1)} \sim \text{Mult}\left(p_{\boldsymbol{\theta}^{(s)}}(y|\mathfrak{q}_{-\{j\}}^{(s)}, \cdot) p_{\boldsymbol{\alpha}_j^{(s)}}(\cdot|\boldsymbol{x}_j)\right)$ for each observation (the subscript i is voluntarily omitted);
    Compute the *maximum a posteriori* of these latent features for all observations $\hat{\mathbf{q}}_j^{(s)} = \text{argmax}_{\mathfrak{q}_j} p_{\boldsymbol{\alpha}_j^{(s)}}(\mathfrak{q}_j|\boldsymbol{x}_j)$;
    Compute the associated logistic regression parameters $\hat{\boldsymbol{\theta}}^{(s)} = \text{argmax}_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \hat{\mathbf{q}}_j^{(s)}, \mathbf{y})$;
  end
  $s \leftarrow s + 1$;
end
Choose the best quantization $\hat{\mathbf{q}}^{(s^\star)}$, which associated logistic regression parameters $\hat{\boldsymbol{\theta}}^{(s_\star)}$ yield the lowest BIC criterion.

Algorithm 7 : *glmdisc*-SEM: supervised multivariate quantization for logistic regression with an SEM algorithm.

A.3 Factor levels grouping method

As part of Chapter 3, we gave results for competing methods MDLP / χ^2 tests where the χ^2 tests can be explicited in Algorithm 8 which I called ChiCollapse. My rather naïve implementation (where, as in the pseudo-code, all pairwise χ^2 tests are recalculated at each step) is available as a gist on Github at `https://gist.github.com/adimajo/eb007492007d650091f6bd7cb2047493`. An example of the resulting usage of the grouped levels in a predictive setting is also given as a

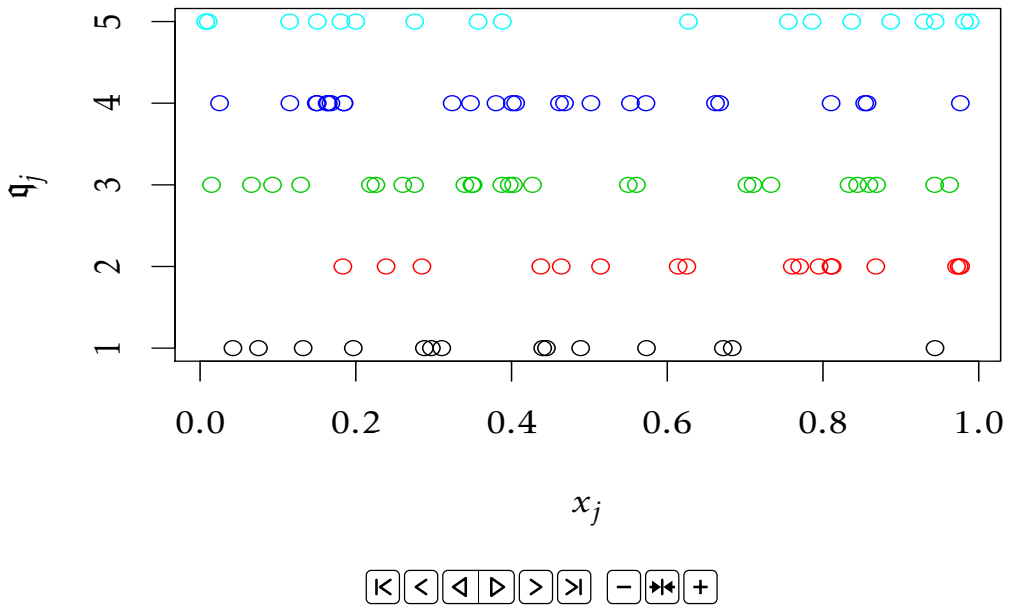


Figure A.11 – Animation of the $\mathfrak{q}_j$ of *glmdisc*-SEM through the iterations (s).

gist on Github at `https://gist.github.com/adimajo/8f8401b59ba838c65534673842b0f60d`.

Data : $n, \mathbf{x}, \alpha$
Result : $\hat{\boldsymbol{q}}$
for $j = 1$ *to* d **do**
  $\alpha_{\max} = 1$;
  Let $C_{j,h} = \{h\}$ for $1 \leq i \leq l_j$;
  while $\alpha_{max} > \alpha$ **do**
    Let $\hat{\boldsymbol{q}}_j(\cdot) = (\hat{q}_{j,h}(\cdot))_1^{m_j}$;
    Set $\hat{q}_{j,h}(\cdot) = \mathbb{1}_{C_{j,h}}(\cdot)$;
    for $1 \leq h_1 < h_2 \leq m_j$ **do**
      $\chi^2_{h_1,h_2} = \sum_{h'=h_1}^{h_2} \sum_{y=0}^{1} \frac{\left(\sum_{i=1}^n \mathbb{1}_y(y_i)\hat{q}_{j,h'}(x_{i,j}) - \frac{\sum_{i=1}^n \hat{q}_{j,h'}(x_{i,j}) \times \sum_{i=1}^n \mathbb{1}_y(y_i)}{n}\right)^2}{\frac{\sum_{i=1}^n \hat{q}_{j,h'}(x_{i,j}) \times \sum_{i=1}^n \mathbb{1}_y(y_i)}{n}}$;
    end
    Let $(h_1, h_2) = \text{argmin}_{h_1,h_2} \chi^2_{h_1,h_2}$, $C_{j,h_1} = C_{j,h_1} \cup C_{j,h_2}$ and $C_{j,h} \leftarrow C_{j,h+1}$ for $h_2 \leq h < m_j$;
    Let $m_j \leftarrow m_j - 1$;
    Let $X \sim \chi^2$ and $\alpha_{\max} = \max_{h_1,h_2} p(X \geq \chi^2_{h_1,h_2}) = p(X \geq \min_{h_1,h_2} \chi^2_{h_1,h_2})$.
  end
end

Algorithm 8 : ChiCollapse algorithm: adaptation of ChiMerge to categorical features.

A.4 Logistic regression-based trees

A.4.1 LogitBoost

The LogitBoost algorithm [5] for 2 classes is equivalent to the Iterative Reweighted Least Squares method [4]. However, in LMT, in order to perform feature selection, a slight modification is brought to the algorithm so as to fit univariate regressions and pick the best. It is given in Algorithm 9.

Data : $n, \mathbf{x}, \mathbf{y}, S$
Result : $F(\boldsymbol{x})$
Let weights $w_i = 1/n$, $F(x) = 0$, $p(1|\boldsymbol{x}) = \frac{\exp(F(\boldsymbol{x})}{\exp(F(\boldsymbol{x})+\exp(-F(\boldsymbol{x})}$;
for $s = 1$ *to* S **do**
  Compute weights $\mathbf{w} = p(1|\mathbf{x}) \odot (\mathbf{1} - p(1|\mathbf{x}))$;
  Compute response $z_i = \frac{y_i - p(1|\boldsymbol{x})}{w_i}$;
  for $j = 1$ *to* d **do**
    Fit the univariate reweighted regression coefficient $\hat{\theta}_{s,j} = \text{argmin}_\theta \sum_{i=1}^{n} w_i \cdot (z_i - \theta x_{i,j})^2$;
  end
  Retrieve the best univariate coefficient $j^\star = \text{argmin}_{1 \le j \le d} \sum_{i=1}^{n} w_i \cdot (z_i - \hat{\theta}_{s,j} x_{i,j})^2$;
  Update $F(\boldsymbol{x}) = F(\boldsymbol{x}) + \frac{1}{2}\hat{\theta}_{s,j^\star} x_j$;
end

Algorithm 9 : LogitBoost algorithm.

A.4.2 PLS

The PLS algorithm given in (10) has been proposed in [17]. This version is adapted from [4] (Algorithm 3.3 in Section 3.5.2 p. 81).

Data : $\mathbf{x}, \mathbf{y}$
Result : $\mathbf{z}$
Standardize each $\mathbf{x}_j$ to have mean zero and variance one (which implies that categorical features have been one-hot encoded prior to this step) ;
Set $\hat{\mathbf{y}}^{(0)} = \mathbf{1}'|\mathbf{y}|$;
Set $\mathbf{x}_j^{(0)} = \mathbf{x}_j$;
for $j = 1$ *to* d **do**
  Let $\mathbf{z}_j = \sum_{j'=1}^{d} \hat{\phi}_{j,j'} \mathbf{x}_{j'}^{(j-1)}$ where $\hat{\phi}_{j,j'} = \mathbf{x}_{j'}^{(j-1)\prime}\mathbf{y}$;
  Let $\theta_j = \frac{\mathbf{z}_j{}'\mathbf{y}}{\mathbf{z}_j{}'\mathbf{z}_j}$;
  Let $\hat{\mathbf{y}}^{(j)} = \hat{\mathbf{y}}^{(j-1)} + \theta_j \mathbf{z}_j$;
  Orthogonalize each $\mathbf{x}_{j'}^{(j-1)}$ w.r.t. $\mathbf{z}_j$: $\mathbf{x}_{j'}^{(j)} = \mathbf{x}_{j'}^{(j-1)} - \frac{\mathbf{z}_j{}'\mathbf{x}_{j'}^{(j-1)}}{\mathbf{z}_j{}'\mathbf{z}_j}\mathbf{z}_j$ for $1 \le j' \le d$;
end

Algorithm 10 : PLS algorithm (adapted from [4]).

A.4.3 SPC

The SPC algorithm given in (11) has been proposed in [1]. This version is adapted from [4] (Algorithm 18.1 in Section 18.6. p. 678).

Data : $\mathbf{x}, \mathbf{y}, \{T_1, \ldots, T_K\}, \{m_1, \ldots, m_L\}$
Result : $\mathbf{z}$
Standardize each $\mathbf{x}_j$ to have mean zero and variance one (which implies that categorical features have been one-hot encoded prior to this step) ;
Compute the univariate regression coefficients for the outcomes $\mathbf{y}$ as a function of each features $\mathbf{x}_j$;
for *Principal components* $m \in \{m_1, \ldots, m_L\}$ **do**
  for *Threshold* $T \in \{T_1, \ldots, T_K\}$ **do**
    Form a reduced data matrix consisting of only those features whose univariate coefficient exceeds T in absolute value, and compute the first m principal components of this matrix;
    Use these principal components in a logistic regression model to predict the outcomes $\mathbf{y}$;
  end
end
Pick $(T^\star, m^\star)$ by cross-validation.

Algorithm 11 : SPC algorithm (adapted from [4]).

A.4.4 LMT

The LMT algorithm given in (12) has been proposed in [10].

Data : $\mathbf{x}, \mathbf{y}$
Result : The logistic regression tree
Fit a classification tree using the C4.5 algorithm;
Fit a logistic regression at the root node by using LogitBoost (Algorithm 9 - the number of iterations is determined by cross-validation);
Fit a logistic regression at the children nodes by resuming LogitBoost (Algorithm 9) on the respective sub-populations;
Prune the tree based on the CART algorithm's pruning criterion (composed of misclassification error and complexity penalization);

Algorithm 12 : LMT algorithm (adapted from [10]).

A.4.5 MOB

The MOB algorithm given in (13) has been proposed in [18].

Data : $\mathbf{x}, \mathbf{y}$
Result : The logistic regression tree
Fit a logistic regression to all observations in current node;
Test for "parameters instability" for all features $x_j \in \boldsymbol{x}$;
If the minimum p-value of these tests is lower than a user-defined threshold, the process is recursively repeated on the children nodes defined by this feature;

Algorithm 13 : MOB algorithm (adapted from [18]).

References of Appendix A

[1] Eric Bair et al. « Prediction by supervised principal components ». In: *Journal of the American Statistical Association* 101.473 (2006), pp. 119–137.

[2] John Banasik and Jonathan Crook. « Reject inference, augmentation, and sample selection ». In: *European Journal of Operational Research* 183.3 (2007), pp. 1582–1594. URL: http://www.sciencedirect.com/science/article/pii/S0377221706011969 (visited on 08/25/2016).

[3] Usama Fayyad and Keki Irani. « Multi-interval discretization of continuous-valued attributes for classification learning ». In: *13th International Joint Conference on Artificial Intelligence*. 1993, pp. 1022–1029.

[4] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*. Vol. 1. 10. Springer series in statistics New York, NY, USA: 2001.

[5] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. « Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors) ». In: *The annals of statistics* 28.2 (2000), pp. 337–407.

[6] Crédit Groupe. « Scorecard Development Methodology Guidelines ». In: *Crédit Agricole Internal Guidelines* (2015).

[7] Asma Guizani. « Traitement des dossiers refusés dans le processus d'octroi de crédit aux particuliers. » 2014CNAM0941. PhD thesis. 2014. URL: http://www.theses.fr/2014CNAM0941/document.

[8] Asma Guizani et al. « Une comparaison de quatre techniques d'inférence des refusés dans le processus d'octroi de crédit ». In: *45 emes Journées de statistique*. 2013. URL: http://cedric.cnam.fr/fichiers/art_2753.pdf (visited on 08/25/2016).

[9] Randy Kerber. « Chimerge: Discretization of numeric attributes ». In: *Proceedings of the tenth national conference on Artificial intelligence*. Aaai Press. 1992, pp. 123–128.

[10] Niels Landwehr, Mark Hall, and Eibe Frank. « Logistic model trees ». In: *Machine learning* 59.1-2 (2005), pp. 161–205.

[11] Huan Liu and Rudy Setiono. « Chi2: Feature selection and discretization of numeric attributes ». In: *Tools with artificial intelligence, 1995. proceedings., seventh international conference on*. IEEE. 1995, pp. 388–391.

[12] Ha Thu Nguyen. *Reject inference in application scorecards: evidence from France*. Tech. rep. University of Paris West-Nanterre la Défense, EconomiX, 2016. URL: http://economix.fr/pdf/dt/2016/WP_EcoX_2016-10.pdf (visited on 08/25/2016).

[13] Chao-Ton Su and Jyh-Hwa Hsu. « An extended chi2 algorithm for discretization of real value attributes ». In: *IEEE transactions on knowledge and data engineering* 17.3 (2005), pp. 437–441.

[14] Francis EH Tay and Lixiang Shen. « A modified chi2 algorithm for discretization ». In: *IEEE Transactions on Knowledge & Data Engineering* 3 (2002), pp. 666–670.

[15] Emmanuel Viennet, Françoise Fogelman Soulié, and Benoît Rognier. « Evaluation de Techniques de Traitement des Refusés pour l'Octroi de Crédit ». In: *arXiv preprint cs/0607048* (2006). URL: http://arxiv.org/abs/cs/0607048 (visited on 08/25/2016).

[16] Ke Wang and Bing Liu. « Concurrent discretization of multiple attributes ». In: *Pacific Rim International Conference on Artificial Intelligence*. Springer. 1998, pp. 250–259.

[17] Svante Wold et al. « The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses ». In: *SIAM Journal on Scientific and Statistical Computing* 5.3 (1984), pp. 735–743.

[18] Achim Zeileis, Torsten Hothorn, and Kurt Hornik. « Model-based recursive partitioning ». In: *Journal of Computational and Graphical Statistics* 17.2 (2008), pp. 492–514.

Appendix B

Software

B.1 The R Statistical Software

The vast majority of the code used throughout this manuscript has been done in R. Most experiments can be rerun from the Github repository of the manuscirpt at `https://www.github.com/adimajo/manuscrit_these`.

More information about the R Statistical Software, RStudio, and git, which I used extensively during the PhD, can be found respectively in [1, 2, 3].

B.1.1 The glmdisc package

The glmdisc package can be found on CRAN at `https://cran.r-project.org/web/packages/glmdisc/index.html` and on Github at `https://www.github.com/adimajo/glmdisc`. It consists in the R implementation of the *glmdisc* algorithm for discretizing continuous attributes, merging factor levels and introducing sparse pairwise interactions proposed in Chapters 4 and 5.

Quick installation guide The package can be installed from CRAN using:

```
install.packages("glmdisc")
```

As the package is also hosted on Github, to get the latest development version, a simple installation procedure is to get the devtools package and run:

```
devtools::install_github("adimajo/glmdisc", build_vignette = TRUE)
```

The build_vignette argument ensures the package's vignette is installed as well.

Behind company proxies however, devtools::install_github might not work (contrary to install.packages if the proxy is well set up). A workaround is to get the httr package which allows to wrap the previous function call in with_config(use_proxy(YOUR_PROXY_SETTINGS),...).

Main functions Once installed, the R help and vignette detail the functioning of the package. Nevertheless, I should mention a few useful tips:

— The package's vignette can be obtained *e.g.* by vignette("glmdisc").

— The main function is glmdisc() (see help(glmdisc) for details).

— Its main arguments are predictors and labels where predictors are assumed to be of type num or factor.
— An option to seek for interactions (see Chapter 4) is given as interact = TRUE.
— For the moment, it is not possible to do Cross-Validation or to use a user-defined validation / test dataset(s).
— However, arguments validation, test and proportions lets the user choose if the quantization / interactions are optimized on the validation set, if a test performance has to be reported, and what proportion(s) of the original data has to be used for each of these datasets, *e.g.* proportions = c(0.2,0.2) splits predictors and labels into 60 % training, 20 % validation and 20 % test.
— To use a learnt quantization, and its associated logistic regression on a new set, the predict and discretize functions are provided.
— If a presumable bug is encountered, an issue may be raised on the Github page of the package.

B.1.2 Miscellaneous

Apart from the glmdisc package, I produced a package named scoring for the purpose of *Credit Scoring* practitioners, which contains the glmdisc package, the reject inference methods discussed thoroughly in Chapter 2 and detailed in Appendix A.1, enhances the discretization package containing, among others, the MDLP and χ^2 discretization methods to which *glmdisc* is compared in Chapter 3. The scoring package can be found at `https://www.github.com/adimajo/scoring`.

The figures that were generated by the combined used of R code and the tikzDevice package can be rerun and are located in the `R_CODE_FIGURES` folder of the repository.

B.2 The Python programming language

Some experiments were performed in Python, both to benefit from implementations not available in R and to learn this rapidly-growing multi-purpose language which machine learning packages have "catched up" on the exhaustivity of the R framework.

B.2.1 The glmdisc package

The *glmdisc*-SEM algorithm is available in Python, though in inferior state of development in comparison to the R implementation, at the following link: `https://www.github.com/adimajo/glmdisc_python`.

Quick installation guide As the package is hosted on Github, a simple installation procedure is to use pip.

```
pip install --upgrade https://github.com/adimajo/glmdisc_python/archive/master.tar.gz
```

Again, behind company proxies, it might be useful to add the --proxy=http://username:password@server:port option.

Main functions Once installed, the Python help can be browsed with the help function. I tried to use the coding style of scikit –learn as much as possible such that the quantization algorithm *glmdisc* is here provided as a class, which itself provides several methods among which fit is appropriate to train the quantization.

B.2.2 The glmdisc-NN notebooks

As mentioned in Chapter 3, the implementation of *glmdisc*-NN is straightforward in terms of neural network architecture. Therefore, all experiments involving *glmdisc*-NN were performed in Jupyter Notebooks. The Notebooks for experiments on simulated data can be found in the `PYTHON_NOTEBOOKS` folder of the repository.

Prior to this work, a quick proof of concept for discretizing continuous features was performed and the following snippet shows how simple it is with standard deep learning libraries (x_1, x_2 and y are drawn from the running example of Section 3.6):

```
from keras import *
from keras.layers import *

input1 = Input((1,))
hidden1 = Dense(3, activation = 'softmax')

input2 = Input((1,))
hidden2 = Dense(3, activation = 'softmax')

full_hidden = merge([hidden1(input1), hidden2(input2)], mode = '
    concat')
output = Dense(1,activation = 'sigmoid')(full_hidden)

model = Model([input1,input2],[output])
model.compile(loss='binary_crossentropy', optimizer='adam')
model.fit([x[:,0],x[:,1]],y, nb_epoch = 300)
```

The following snippet illustrates how straightforward the implementation of *glmdisc*-NN is, as seen as a computational graph on Figure 3.11.

```
# The data function which provides all inputs to create_model is not
    shown here for concision.

def create_model(x_quant,x_qual,x_qual_dummy,y,x_quant_test,x_qual_
    test,x_qual_dummy_test,y_test):
    """Creates and trains the proposed neural network architecture.
    Args:
        x_quant,x_qual,x_qual_dummy,y,x_quant_test,x_qual_test,x_qual_
            dummy_test,y_test - input data given by the data() function
             not shown here for concision.
    Returns:
        loss - the performance (here Gini on test sample) of the
            resulting best quantization
        model - the trained model
        predicted - the predicted probabilities on the test set using
            the best quantization (to compute confidence intervals)
```

```
    """

    def initialize_neural_net(m_quant,m_qual):
        """Initializes the neural network architecture for
           quantization.
            Args:
               m_quant - list of maximum number of categories per
                   continuous feature
               m_qual - list of maximum number of groups of levels
                   per categorical feature
            Returns:
            liste_inputs_quant, liste_layers_quant, liste_layers_
               quant_inputs - lists of inputs / layers for continuous
                features
            liste_inputs_qual, liste_layers_qual, liste_layers_qual_
               inputs - same for categorical features
            """

        liste_inputs_quant = [None] * d1
        liste_inputs_qual = [None] * d2

        liste_layers_quant = [None] * d1
        liste_layers_qual = [None] * d2

        liste_layers_quant_inputs = [None] * d1
        liste_layers_qual_inputs = [None] * d2

        for i in range(d1):
            liste_inputs_quant[i] = Input((1, ))
            liste_layers_quant[i] = Dense(m_quant[i], activation='
               softmax')
            liste_layers_quant_inputs[i] = liste_layers_quant[i](
                liste_inputs_quant[i])

        for i in range(d2):
            liste_inputs_qual[i] = Input((len(np.unique(x_qual[:, i])
               ), ))
            if (len(np.unique(x_qual[:, i])) > m_qual[i]):
                liste_layers_qual[i] = Dense(
                m_qual[i], activation='softmax', use_bias=False)
            else:
                liste_layers_qual[i] = Dense(
                len(np.unique(x_qual[:, i])), activation='softmax',
                    use_bias=False)

            liste_layers_qual_inputs[i] = liste_layers_qual[i](
                liste_inputs_qual[i])

        return ([
```

```
51              liste_inputs_quant, liste_layers_quant, liste_layers_
                    quant_inputs,
52              liste_inputs_qual, liste_layers_qual, liste_layers_qual_
                    inputs
53          ])
54
55
56      def from_layers_to_proba_training(d1,d2,liste_layers_quant,liste_
            layers_qual):
57          """Computes q_(alpha) for training samples.
58              Args:
59                  d1, d2 - number of continuous (resp. categorical)
                        features
60                  liste_layers_quant, liste_layers_qual - given by
                        initialize_neural_net(...)
61              Returns:
62                          results - list of matrices of q_(alpha,i,j,h)
                                on training samples
63              """
64
65          results = [None] * (d1 + d2)
66
67          for j in range(d1):
68              results[j] = K.function([liste_layers_quant[j].input],
69                                      [liste_layers_quant[j].output])(
70                                          [x_quant[:, j, np.newaxis]])
71
72          for j in range(d2):
73              results[j + d1] = K.function([liste_layers_qual[j].input
                    ],
74                                           [liste_layers_qual[j].output
                                                ])(
75                                            [liste_qual_arrays[j]])
76
77          return (results)
78
79
80
81
82      def from_weights_to_proba_test(d1,d2,m_quant,m_qual,history,x_
            quant_test,x_qual_test,n_test):
83          """Computes q_(alpha) for test samples.
84              Args:
85                  d1, d2 - number of continuous (resp. categorical)
                        features
86                  m_quant, m_qual -
87                  history -
88                  x_quant_test, x_qual_test, n_test -
89              Returns:
```

```
90                      results - list of matrices of q_(alpha,i,j,h)
                            on test samples
91              """
92
93          results = [None] * (d1 + d2)
94
95          for j in range(d1):
96              results[j] = np.zeros((n_test,m_quant[j]))
97              for i in range(m_quant[j]):
98                  results[j][:,i] = history.best_weights[j][1][i] +
                        history.best_weights[j][0][0][i]*x_quant_test[:,j]
99
100
101         for j in range(d2):
102             results[j+d1] = np.zeros((n_test,history.best_weights[j+
                    d1][0].shape[1]))
103             for i in range(history.best_weights[j+d1][0].shape[1]):
104                 for k in range(n_test):
105                     results[j+d1][k,i] = history.best_weights[j+d1
                            ][0][x_qual_test[k,j],i]
106
107         return(results)
108
109
110     def evaluate_disc(type,d1,d2,misc):
111         """Evaluates the quality of a quantization.
112             Args:
113                type - train or test
114                d1, d2 - number of continuous (resp. categorical)
                       features
115                misc - depends on type
116             Returns:
117                   performance - for type="train" BIC; for type="test"
                           Gini.
118                   predicted - the resulting quantization of either
                           train or test data depending on type.
119             """
120
121         if type=="train":
122             proba = from_layers_to_proba_training(d1,d2,misc[0],misc
                    [1])
123         else:
124             proba = from_weights_to_proba_test(d1,d2,misc[0],misc[1],
                    misc[2],misc[3],misc[4],misc[5])
125
126
127         results = [None] * (d1 + d2)
128
129         if type=="train":
```

```
130             X_transformed = np.ones((n, 1))
131         else:
132             X_transformed = np.ones((n_test, 1))
133 
134         for j in range(d1 + d2):
135             if type=="train":
136                 results[j] = np.argmax(proba[j][0], axis=1)
137             else:
138                 results[j] = np.argmax(proba[j], axis=1)
139             X_transformed = np.concatenate(
140                 (X_transformed, sk.preprocessing.OneHotEncoder(
                        categories='auto',sparse=False,handle_unknown="
                        ignore").fit_transform(
141                         X=results[j].reshape(-1, 1))),
142                 axis=1)
143 
144         proposed_logistic_regression = sk.linear_
                model.LogisticRegression(
145             fit_intercept=False, solver = "lbfgs", C=1e20, tol=1e-8,
                    max_iter=50)
146 
147 
148         if type=="train":
149             proposed_logistic_regression.fit(X=X_transformed, y=
                    y.reshape((n, )))
150             performance = 2 * sk.metrics.log_loss(
151               y,
152               proposed_logistic_regression.predict_proba(X=X_
                        transformed)[:, 1],
153               normalize=False
154           ) + proposed_logistic_regression.coef_.shape[1] * np.log(n)
155             predicted = proposed_logistic_regression.predict_proba(X_
                    transformed)[:,1]
156 
157         else:
158             proposed_logistic_regression.fit(X=X_transformed, y=y_
                    test.reshape((n_test, )))
159             performance = 2*sk.metrics.roc_auc_score(y_test,proposed_
                    logistic_regression.predict_proba(X_transformed)[:,1])
                    -1
160             predicted = proposed_logistic_regression.predict_proba(X_
                    transformed)[:,1]
161 
162         return (performance, predicted)
163 
164 
165 
166     class LossHistory(Callback):
167         """Callback for Keras. At each epoch, computes the
```

```
        performance of the proposed quantization."""
168
169         def on_train_begin(self, logs={}):
170             self.losses = []
171             self.best_criterion = float("inf")
172             self.best_outputs = []
173
174         def on_epoch_end(self, batch, logs={}):
175             self.losses.append(evaluate_disc("train",d1,d2,[liste_
                layers_quant,liste_layers_qual])[0])
176             if self.losses[-1] < self.best_criterion:
177                 self.best_weights = []
178                 self.best_outputs = []
179                 self.best_criterion = self.losses[-1]
180                 for j in range(d1):
181                     self.best_weights.append(liste_layers_quant[j]
                        .get_weights())
182                     self.best_outputs.append(
183                         K.function([liste_layers_quant[j].input],
184                                    [liste_layers_quant[j].output])(
185                                        [x_quant[:, j, np.newaxis]]))
186                 for j in range(d2):
187                     self.best_weights.append(liste_layers_qual[j].get
                        _weights())
188                     self.best_outputs.append(
189                         K.function([liste_layers_qual[j].input],
190                                    [liste_layers_qual[j].output])(
191                                        [liste_qual_arrays[j]]))
192
193
194         # quant is the number of maximum intervals per continuous
              feature
195         # it is the single user-defined parameter of our proposal
196
197     quant = 10
198     qual = 5
199
200     m_quant = [int(quant)] * d1
201     m_qual = [int(qual)] * d2
202
203         liste_inputs_quant, liste_layers_quant, liste_layers_quant_
            inputs, liste_inputs_qual, liste_layers_qual, liste_layers
            _qual_inputs = initialize_neural_net(m_quant,m_qual)
204
205         # full_hidden is the concatenation of all component-wise
              layers
206
207     full_hidden = concatenate(
208         list(
```

```
209             chain.from_iterable(
210                 [liste_layers_quant_inputs, liste_layers_qual_inputs]))
                    )
211     output = Dense(1, activation='sigmoid')(full_hidden)
212     model = Model(
213       inputs=list(chain.from_iterable([liste_inputs_quant, liste_
            inputs_qual])),
214       outputs=[output])
215
216     optim = optimizers.SGD(lr=10**-3)
217
218     model.compile(loss='binary_crossentropy', optimizer=optim,
            metrics=['accuracy'])
219
220     history = LossHistory()
221
222     model.fit(
223       list(chain.from_iterable([list(x_quant.T), liste_qual_arrays]))
            ,
224       y,
225       epochs=600,
226       batch_size=128,
227       callbacks=history)
228
229     n_test = x_quant_test.shape[0]
230     performance, predicted = evaluate_disc("test",d1,d2,misc=[m_quant
            ,m_qual,history,x_quant_test,x_qual_test,n_test])
231
232     return {'loss': -performance, 'model': model, 'predicted':
            predicted}
```

References of Appendix B

[1] Sébastien Déjean and Thibault Laurent. *Encore besoin d'R*. 2016. URL: https://www.math.univ-toulouse.fr/~sdejean/PDF/R-avance.pdf.

[2] Colin Gillespie and Robin Lovelace. *Efficient R programming*. Dec. 2016. URL: https://csgillespie.github.io/efficientR/.

[3] G. Gauthier Marc and Emily Reese. *Manage your code with Git and GitHub*. OpenClassrooms. 2019. URL: https://openclassrooms.com/fr/courses/3321726-manage-your-code-with-git-and-github.

Appendix C

Publications

C.1 Poster

Les travaux de discrétisation, regroupement et introduction d'interactions pour le modèle de régression logistique discutés aux chapitres 3 et 4 ont fait l'objet d'un poster :

Adrien Ehrhardt et al. « Model-based multivariate discretization for logistic regression ». In: Data Science Summer School. 2017. URL: http://2017.ds3-datascience-polytechnique.fr/wp-content/uploads/2017/08/DS3_posterID_049.pdf

C.2 Présentations à des conférences avec comité de relecture

Les travaux concernant la réintégration des refusés discutés au chapitre 2 ont fait l'objet de deux communications orales :

Adrien Ehrhardt et al. « Credit Scoring : biais d'échantillon ou réintégration des refusé ». In: Rencontres des jeunes statisticiens. 2017. URL: https://adimajo.github.io/assets/publications/EHRHARDT_RJS_REINTEGRATION.pdf

Adrien Ehrhardt et al. « Réintégration des refusés en Credit Scoring ». In: *49e Journées de Statistique*. Avignon, France, May 2017. URL: https://hal.archives-ouvertes.fr/hal-01653767

Les travaux de discrétisation, regroupement et introduction d'interactions pour le modèle de régression logistique discutés aux chapitres 3 et 4 ont fait l'objet d'une communication orale :

Adrien Ehrhardt et al. « Supervised multivariate discretization and levels merging for logistic regression ». In: *23rd International Conference on Computational Statistics*. Iasi, Romania, Aug. 2018. URL: https://hal.archives-ouvertes.fr/hal-01949128

C.3 Articles scientifiques

Les travaux concernant la réintégration des refusés discutés au chapitre 2 font l'objet d'un article scientifique en cours de rédaction.

Les travaux de quantification pour le modèle de régression logistique discutés aux chapitres 3 avec un mécanisme d'estimation basé sur les réseaux de neurones, appelé *glmdisc*-NN, ont fait l'objet d'un article scientifique (preprint soumis) :

Adrien Ehrhardt et al. « Feature quantization for parsimonious and interpretable predictive models ». In: *arXiv preprint arXiv:1903.08920* (2019)

Les travaux de quantification et d'introduction d'interactions pour le modèle de régression logistique discutés aux chapitres 3 et 4 avec un mécanisme d'estimation basé sur un algorithme SEM et un algorithme Metropolis-Hastings, appelé *glmdisc*-SEM, vont faire l'objet d'un article scientifique en cours de préparation.

Table des matières

Formalization and study of statistical problems in Credit Scoring
Reject inference, discretization and pairwise interactions, logistic regression trees

Abstract

Cette thèse se place dans le cadre des modèles d'apprentissage automatique de classification binaire. Le cas d'application est le scoring de risque de crédit. En particulier, les méthodes proposées ainsi que les approches existantes sont illustrées par des données réelles de Crédit Agricole Consumer Finance, acteur majeur en Europe du crédit à la consommation, à l'origine de cette thèse grâce à un financement CIFRE.
Premièrement, on s'intéresse à la problématique dite de "réintégration des refusés". L'objectif est de tirer parti des informations collectées sur les clients refusés, donc par définition sans étiquette connue, quant à leur remboursement de crédit. L'enjeu a été de reformuler cette problématique industrielle classique dans un cadre rigoureux, celui de la modélisation pour données manquantes. Cette approche a permis de donner tout d'abord un nouvel éclairage aux méthodes standards de réintégration, et ensuite de conclure qu'aucune d'entre elles n'était réellement à recommander tant que leur modélisation, lacunaire en l'état, interdisait l'emploi de méthodes de choix de modèles statistiques.
Une autre problématique industrielle classique correspond à la discrétisation des variables continues et le regroupement des modalités de variables catégorielles avant toute étape de modélisation. La motivation sous-jacente correspond à des raisons à la fois pratiques (interprétabilité) et théoriques (performance de prédiction). Pour effectuer ces quantifications, des heuristiques, souvent manuelles et chronophages, sont cependant utilisées. Nous avons alors reformulé cette pratique courante de perte d'information comme un problème de modélisation à variables latentes, revenant ainsi à une sélection de modèle. Par ailleurs, la combinatoire associé à cet espace de modèles nous a conduit à proposer des stratégies d'exploration, soit basées sur un réseau de neurone avec un gradient stochastique, soit basées sur un algorithme de type EM stochastique.
Comme extension du problème précédent, il est également courant d'introduire des interactions entre variables afin, comme toujours, d'améliorer la performance prédictive des modèles. La pratique classiquement répandue est de nouveau manuelle et chronophage, avec des risques accrus étant donnée la surcouche combinatoire que cela engendre. Nous avons alors proposé un algorithme de Metropolis-Hastings permettant de rechercher les meilleures interactions de façon quasi-automatique tout en garantissant de bonnes performances grâce à ses propriétés de convergence standards.
La dernière problématique abordée vise de nouveau à formaliser une pratique répandue, consistant à définir le système d'acceptation non pas comme un unique score mais plutôt comme un arbre de scores. Chaque branche de l'arbre est alors relatif à un segment de population particulier. Pour lever la sous-optimalité des méthodes classiques utilisées dans les entreprises, nous proposons une approche globale optimisant le système d'acceptation dans son ensemble. Les résultats empiriques qui en découlent sont particulièrement prometteurs, illustrant ainsi la flexibilité d'un mélange de modélisation paramétrique et non paramétrique.
Enfin, nous anticipons sur les futurs verrous qui vont apparaître en Credit Scoring et qui sont pour beaucoup liés la grande dimension (en termes de prédicteurs). En effet, l'industrie financière investit actuellement dans le stockage de données massives et non structurées, dont la prochaine utilisation dans les règles de prédiction devra s'appuyer sur un minimum de garanties théoriques pour espérer atteindre les espoirs de performance prédictive qui ont présidé à cette collecte.

Keywords: scoring, credit, risk, prediction, discretization, clustering

---

Équipe-projet MΘDAL
Inria Lille Nord-Europe – 40 Avenue Halley – 59650 Villeneuve-d'Ascq

Formalization and study of statistical problems in Credit Scoring
Reject inference, discretization and pairwise interactions, logistic regression trees

Abstract

This manuscript deals with model-based statistical learning in the binary classification setting. As an application, credit scoring is widely examined with a special attention on its specificities. Proposed and existing approaches are illustrated on real data from Crédit Agricole Consumer Finance, a financial institute specialized in consumer loans which financed this PhD through a CIFRE funding.

First, we consider the so-called reject inference problem, which aims at taking advantage of the information collected on rejected credit applicants for which no repayment performance can be observed (*i.e.* unlabelled observations). This industrial problem led to a research one by reinterpreting unlabelled observations as an information loss that can be compensated by modelling missing data. This interpretation sheds light on existing reject inference methods and allows to conclude that none of them should be recommended since they lack proper modelling assumptions that make them suitable for classical statistical model selection tools.

Next, yet another industrial problem, corresponding to the discretization of continuous features or grouping of levels of categorical features before any modelling step, was tackled. This is motivated by practical (interpretability) and theoretical reasons (predictive power). To perform these quantizations, *ad hoc* heuristics are often used, which are empirical and time-consuming for practitioners. They are seen here as a latent variable problem, setting us back to a model selection problem. The high combinatorics of this model space necessitated a new cost-effective and automatic exploration strategy which involves either a particular neural network architecture or a Stochastic-EM algorithm and gives precise statistical guarantees.

Third, as an extension to the preceding problem, interactions of covariates may be introduced in the problem in order to improve the predictive performance. This task, up to now again manually processed by practitioners and highly combinatorial, presents an accrued risk of misselecting a "good" model. It is performed here with a Metropolis-Hastings sampling procedure which finds the best interactions in an automatic fashion while ensuring its standard convergence properties, thus good predictive performance is guaranteed.

Finally, contrary to the preceding problems which tackled a particular scorecard, we look at the scoring system as a whole. It generally consists of a tree-like structure composed of many scorecards (each relative to a particular population segment), which is often not optimized but rather imposed by the company's culture and / or history. Again, *ad hoc* industrial procedures are used, which lead to suboptimal performance. We propose some lines of approach to optimize this logistic regression tree which result in good empirical performance and new research directions illustrating the predictive strength and interpretability of a mix of parametric and non-parametric models.

This manuscript is concluded by a discussion on potential scientific obstacles, among which the high dimensionality (in the number of features). The financial industry is indeed investing massively in unstructured data storage, which remains to this day largely unused for *Credit Scoring* applications. Doing so will need statistical guarantees to achieve the additional predictive performance that was hoped for.

Keywords: scoring, credit, risk, prediction, discretization, clustering