An effective strategy for churn prediction and customer profiling

Louis Geiler, Séverine Affeldt, Mohamed Nadif

An effective strategy for churn prediction and customer profiling

Louis Geiler[a,b], Séverine Affeldt[b,*], Mohamed Nadif[b]

[a] *Brigad, 34 Rue du Sentier, Paris, 75002, France.*
[b] *Centre Borelli UMR 9010, Université Paris Cité, 75006 Paris, France.*

Abstract

Customer churn prediction and profiling are two major economic concerns for many companies. Different learning approaches have been proposed, however the a priori choice of the most suitable model to perform both tasks remains non-trivial as it is highly dependent on the intrinsic characteristics of the churn data. Our study compares eight supervised machine learning methods combined with seven sampling approaches on thirteen public churn data sets. Our evaluations, reported in terms of area under the curve (AUC), explore the influence of rebalancing strategies and data properties on the performance of learning methods. We rely on the Nemenyi test and Correspondence Analysis as means of visualizing the association between models, rebalancing and data. This work identifies the most appropriate methods in an attrition context and proposes an effective pipeline based on an ensemble approach and deep autoencoders segmentation. Our strategy can enlighten marketing or human resources services on the behavioral patterns of customers and their attrition probability. The described experiments are fully reproducible and our proposal can be successfully applied to a wide range of *churn-like* datasets.

Keywords: Churn prediction, Customer profiling, Machine learning, Ensemble approach, Deep autoencoders

1. Introduction

Management and marketing services are trying to cope with the ever-rising competition in industry by focusing their efforts on a strong Customer Relationship Management (CRM). In particular, customer retention has attracted interest as it clearly appeared that retained customers can be of great help for the company by spreading positive word of mouth [1]. Such behavior can subsequently lower the marketing costs of new customers acquisition [2]. Besides, it has become clearer that the acquisition costs of a new customer can be much more higher than the retention costs of an existing one [3, 4, 5]. Hence, preventing customer *churn* or *attrition* can be vital for subscription-based service firms, that rely on fixed and regular membership fees, in numerous areas among which insurance [6], banking [7], online gambling [8], online video games [9], music streaming [10], online services [11] or telecommunication [12, 13, 14, 15]. Therefore, accurately predicting the customers who are prone to churn has become a priority in many industries.

Beyond the churn prediction, the study of the dynamic relationship between the customer satisfaction, the service quality and the customer behavior – loyalty or switching – is today a lively field of research. Indeed, a better understanding of customers experience offers valuable information for marketers. As an example, satisfied customers will be more tolerant to price increases which will in turn bring greater profits [16]. However, certain customer groups may have different perceptions of service providers [17]. For instance, many studies propose to describe the customer satisfaction as a composite of factors such as the corporate image, the internal organization, the physical environment, the staff service and the customer-personal interaction [18]. In the Banking industry, Laroche *et al.* [19] decompose the customer satisfaction into the speed service, the conveniency of the location, the staff competence and the bank friendliness [19]. The amalgamation of the multiplicity and divergence of customer expectations and perceptions naturally calls for customer base segmentation to optimize churn behavior management.

While the negative effects of customer churn can be easily observed – lack of revenues or supplementary costs of attracting new customers –, the churn causes are under continuous study, as these causes generally vary across economical fields and customer groups. For service industries, Cronin and Taylor [20] relied on the effects of time, money constrains, lack of credible alternatives, switching costs, habit, price, convenience and availability to explain customers switching. Similarly, Keaveney [21] identified eight main causal variables for churn, namely price, inconvenience, core service failures, service encounter failures, competitive issues, ethical problems and involuntary factors. Following on these proposals, Athanassopoulos [22] proposed, based on Confirmatory Factor Analysis, five dimensions to describe different customer satisfaction profiles in retail banking services. These dimensions are staff service, business profile, innovativeness, convenience and price. The author also validated the interest in dividing customers into segments market that correspond to their preferences regarding particular aspects of service. The motivation behind customers segmentation – which is one of the most significant methods used in marketing studies – is to select appropriate cus-

*Corresponding author
Email address: severine.affeldt@u-paris.fr (Séverine Affeldt)

tomers for a campaign. This typically increase customer profitability through adapted customer targeting [23, 24]. In fact, a large amount of segmentation methods are developed each year [25, 26], making hard any exhaustive comparison between them.

In this work, our first motivation is to evaluate several machine learning techniques on the churn prediction task. Concerned by the multidimensionality of attrition causes, we also study the performance of ensemble learning approaches to improve the attrition prediction. Finally, in order to explicitly take into account the underlying customer segmentation, we rely on a deep unsupervised clustering method before exploiting an ensemble machine learning approach. Hence, our global objective is to compare several variants of a processing chain for churn analysis.

This chain includes (Figure 1),

(i) a class rebalancing step or a clustering step,

(ii) a supervised or a meta ensemble learning phase,

(iii) a robust evaluation procedure.

As all the variants of the algorithms in the proposed pipeline can not be exhaustively studied in this article, we only consider the algorithms in their original version. Furthermore, the benchmark datasets for our experiments have a relatively important class imbalance between the minority class – unsubscribed individuals – and the majority class. This decreases the performance of standard classifiers [27] which can be aggravated by an overlap of the classes or a fragmentation of the minority class into subsets corresponding to different customer profiles. This motivate the idea to combine the model fitting step with class rebalancing approaches. Through our results, we formulate practical recommendations and propose a generic and novel *ensemble* approach that performs well on a wide range of attrition datasets. Beyond the good performance obtained with our ensemble proposal, we also make the customer segmentation explicit via a deep autoencoder-based clustering. This clustering reveals the features associated to each underlying customer group.

We first introduce the imbalance class distribution issue and describe seven widespread balancing techniques (Section 4). Then, we provide an overview of supervised, ensemble supervised machine learning techniques (Sections 5.1). We also discuss evaluation procedures (Section 5.2) and metrics (Section 5.3) before providing the first comparative experimental results of our pipeline variants (Section 6). These results reveal interesting complementary behaviors between machine learning techniques (Section 6.1) which are summarized with Nemenyi tests and Correspondence Analysis visualizations (Section 6.2). Then, we propose an advisable ensemble churn analysis pipeline which can be successfully applied to various churn-like datasets (Section 6.3).Ultimately, we enrich our ensemble proposal with a data segmentation that respect the underlying customer behavior patterns (Section 6.4). The corresponding prediction results are given in Section 6.5 and compared with the recent `LLM` [28] and `RF`-based [29] models. We discuss the benefits of our approach in terms of churn prediction in Section 6.5.1 and customer profiling in Section 6.5.2. The overall conclusion is given in Section 7.

2. Background

This section describes the churn prediction issue. It also provides summarized information on the publicly available datasets analyzed in this work.

2.1. Notation and problem definition

Throughout the paper, we use bold uppercase characters to denote vectors, uppercase characters to denote random variable and lowercase characters to denote variable values. Let $\mathbf{X} = (x_{ij})$ be a data matrix of $n \times d$ dimension. We assume that Y is the random variable indicating the class y_i of an observation $\mathbf{x}_i = [x_{i1}, \ldots, x_{id}]^\top$ which denotes the i^{th} instance of $\mathbf{X}$. The total number of observations is noted n, and G is the number of classes $C_1, \ldots, C_G$. In a binary or churn prediction context, $G = 2$ and we consider the two classes $+,-$ that correspond to the churn and non churn classes respectively. The churn prediction problem can be modeled as a standard binary classification task. Formally, it is an assignment task that amounts to estimate the conditional probability of $Y = y_i$ given $\mathbf{x}_i$, $P(Y = y_i|\mathbf{x}_i)$, so-called *class posterior*.

2.2. Public datasets

This work involves a comparative evaluation of multiple churn analysis techniques on publicly available datasets only. A churn dataset usually comprises features of different types – numerical and categorical variables – that reflect customers behavior. It also generally exhibits a strong class imbalance, as the proportion of churners is typically lower than the proportion of customers that remain with the company.

The Table 1 gives the public churn datasets that are considered in this work and provides their online access (see also Appendix 7 for details). These datasets have diverse number of instances, number of features, and percentage of churners and *dummified* features. Specifically, before fitting a model, categorical variables are converted to their numerical representation through a *dummification* process where each category becomes a binary variable. We also provide the number of continuous and categorical variables after *dummification*.

Although the general data characteristics given in Table 1 suggest similarities between several datasets, it is important to remind that multiple intrinsic data properties can impact the prediction in the churn context. This includes in particular the existence of small *disjuncts*, the overlap between classes, the noisy data or the borderline instances (see Section 3.3). To establish the extent to which the classes may be intertwined, we propose a *mixture score*, which is defined as follows,

$$mix.Score = (\mu_+ - \mu_-)^\top \left(\frac{\mathbf{\Sigma}_+ + \mathbf{\Sigma}_-}{2} \right)^{-1} (\mu_+ - \mu_-), \qquad (1)$$

where μ_i is the mean vector and $\mathbf{\Sigma}_i$ the covariance matrix of the cluster i respectively. Note that as we deal with mixed data

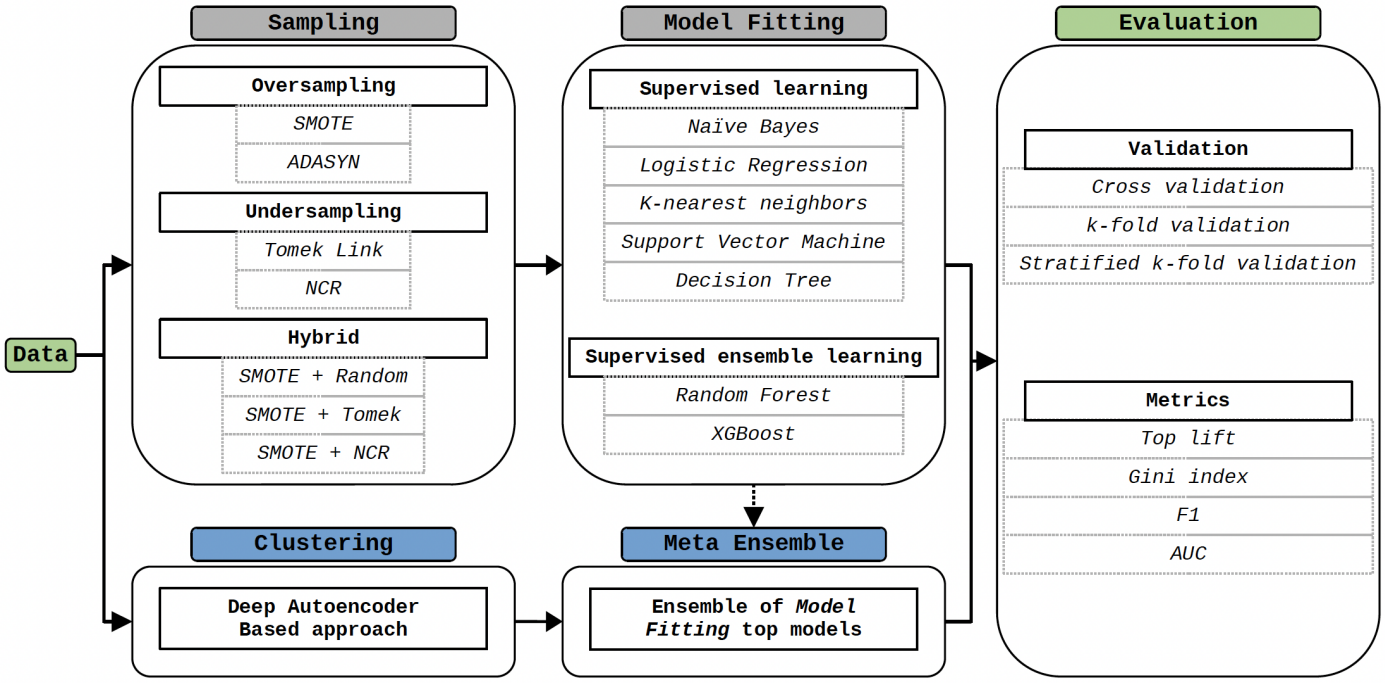


Figure 1: Machine learning pipeline for churn prediction and analysis.

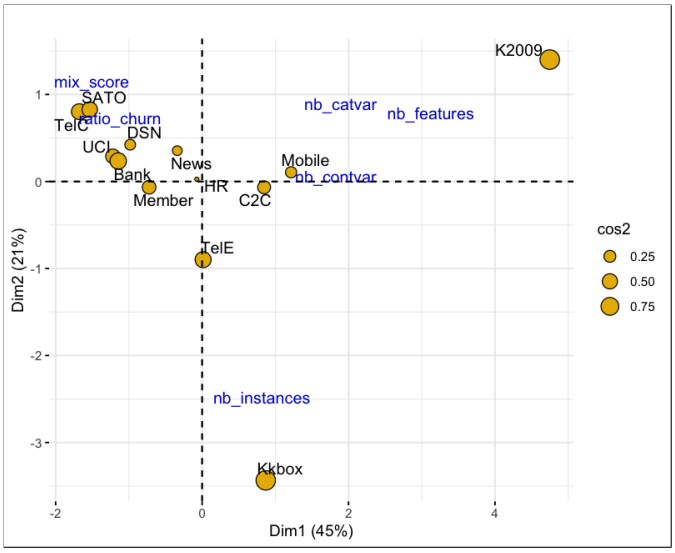
Table 1: Publicly available churn datasets with online access link

Dataset	#Instances	#Features	#Cont.Feat.	#Cat.Feat.[1]	mix.Score	$\frac{churn}{nonchurn}$
K2009	50,000	230	37	1,001	7.28×10^{-4}	0.08
KKbox	970,960	49	12	43	1.48×10^{-2}	0.10
UCI	5,000	20	0	20	2.66×10^{-1}	0.16
HR	1,470	34	14	71	1.03×10^{-1}	0.19
TelE	190,776	19	15	10	3.75×10^{-2}	0.19
News	15,855	18	2	304	1.56×10^{-1}	0.23
Bank	10,000	12	5	10	2.30×10^{-1}	0.25
Mobile	66,469	62	57	5	1.07×10^{-1}	0.27
TelC	7,043	20	3	30	3.69×10^{-1}	0.37
C2C	71,047	71	32	42	1.89×10^{-2}	0.41
Member	10,362	14	4	21	6.26×10^{-2}	0.43
SATO	2,000	13	9	19	1.72×10^{-1}	1
DSN	1,401	15	10	21	2.68×10^{-2}	1

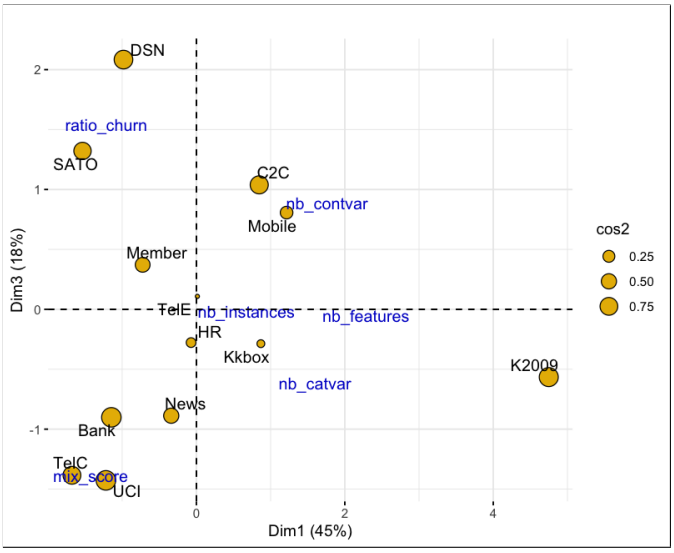
(1) Categorical variables with more than two levels are converted to their numerical representation by *dummification* where each category becomes a binary variable.

(continuous and categorical variables) we perform the Factor Analysis for Mixed Data [30] on the original dataset, and derive μ_i and $\mathbf{\Sigma}_i$. Thereby, the higher the mixture score, the more separable the classes.

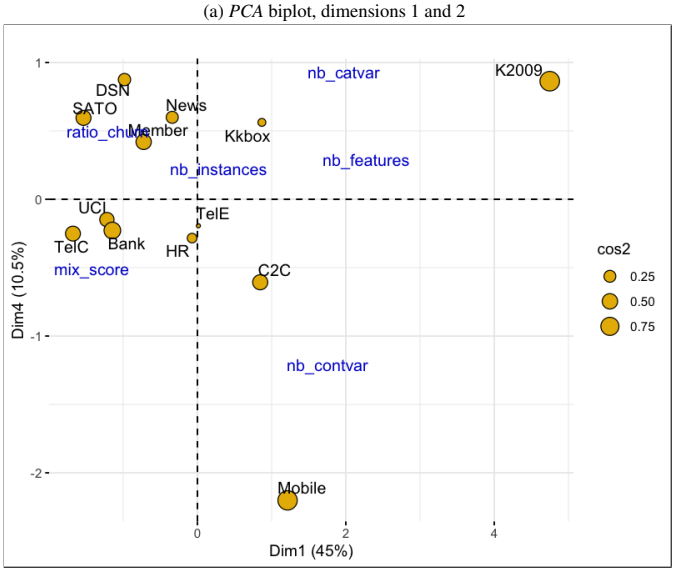
Directly drawing conclusions on the most suitable machine learning algorithm based on the general characteristics given in Table 1 remains challenging. To get a better overview of the multiple datasets facets, we provide in Figure 2, PCA (Principal Component Analysis) biplot representations of the datasets distribution over the characteristics identified in Table 1. As different dimensions can provide different information, we give biplots for the 4 first PCA components explaining 94.5% of the total variance. However, what is important is above all to observe the diversity of these data by the characteristics that describe them. To this end, we rely on the quality of representations of datasets depicted in Fig. 2 (e) and the correlation between the variables and the components depicted in Fig. 2 (f). Thus in Fig. 2 (a,e,f), we note the opposition between very balanced and mixed datasets, with many categorical variables (about 27 times of categorical variables than continuous) such as K2009 and more balanced and less mixed datasets, with fewer variables and only about twice categorical variables than continuous such as SATO, UCI and TeIC. In Fig. 2 (b,e,f), we observe that dimension 2 is mainly characterized by the KKbox dataset with a very high number of instances followed by TeIE the closest dataset. The 3^{rd} component in Fig. 2 (c,e,f) charac-



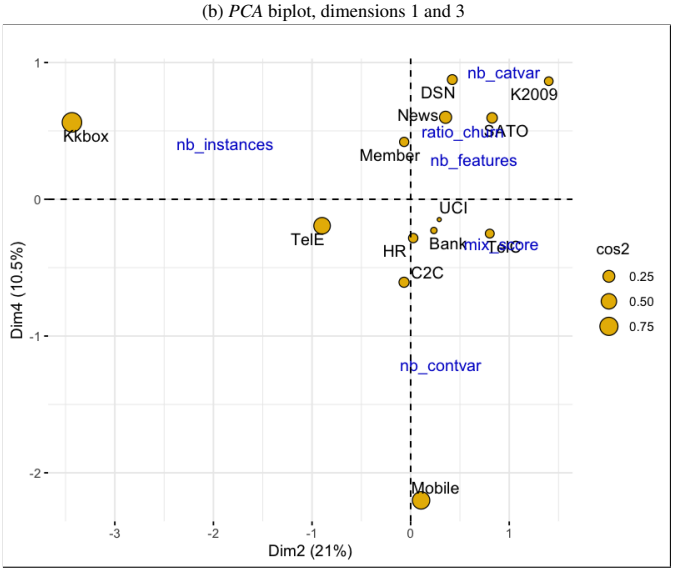
(a) *PCA* biplot, dimensions 1 and 2



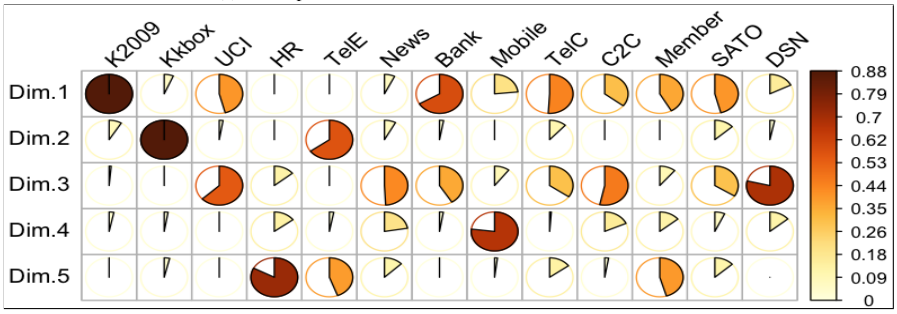
(b) *PCA* biplot, dimensions 1 and 3



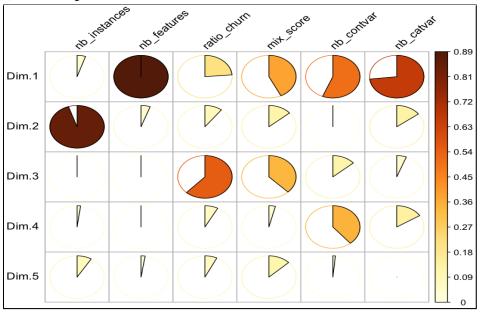
(c) *PCA* biplot, dimensions 1 and 4



(d) *PCA* biplot, dimensions 2 and 4



(e) Individuals representation quality



(f) Variables representation quality

Figure 2: (a, b, c & d) Biplots visualization for publicly available churn-like datasets (*individuals*) and their characteristics (*variables*) for different PCA components. (e & f) Quality of representations on the factor map.

terized mainly by the ratio-churn contrasts highly balanced and less well separated data such as DSN, SATO and less balanced and better separated datasets such as UCI and TeIC. Finally, the 4^{th} component makes it possible to show the opposition between datasets with a very high ratio of continuous variables, compared to the number of categorical variables, such as the Mobile dataset and the rest of the datasets with opposite characteristics. The other datasets not mentioned before share the same interpretations, according to their proximity with the other datasets cited, while taking into account their quality of representation.

To sum up, the diversity of datasets used in this paper will make it possible to highlight the strengths and weaknesses of the methods compared (Section 6.2).

3. Related work

Machine learning techniques are being increasingly used in the customer churn context. These techniques include supervised and semi-supervised approaches that can be devoted to churn prediction or profiling.

3.1. Machine learning for churn prediction

The *K*-nearest neighbors, Naive Bayes classifiers, Linear Regression, Logistic Regression, Linear Discriminant Analysis [31], Decision Tree learning [32, 33] and Support Vector Machine are among the widely used supervised algorithms in the context of churn prediction. Algorithmic modifications [34] and cost-sensitive learning variants [35, 36] of the aforementioned learning methods have also been proposed in the context of imbalanced classes, as encountered in churn datasets. Finally, several studies proposed to rely on ensemble approaches such as Random Forest, AdaBoost [31], Gradient Boosting [33, 37] or XGBoost [38] to tackle the churn prediction task. Successful semi-supervised methods have been proposed [39], as well as deep learning approaches [11, 38, 32, 33].

The strong interest in churn prediction led to various comparative studies related to machine learning in the fields of telecommunication industry, human resources, bank subscription or financial services. Sniegula *et al.* [40] compare three machine learning techniques on a single churn dataset in the context of telecommunication industry. Similarly, Saradhi *et al.* compare three machine learning techniques in the *employee churn* context [41]. They provide results on a private dataset using a cross-validation procedure. Keramati *et al.* [42] proposed a literature and comparative experimental study with four models on a private dataset.

Although interesting, these studies compare very few machine learning techniques in the churn context. Besides, their results usually involve private datasets, making the experiments not reproducible and extrapolation to novel datasets difficult. Finally, these works rarely raised the topic of evaluation procedures, that impacts the validity and robustness of the evaluations, and typically omit the techniques for classes rebalancing, which is an important issue for churn prediction.

While our study does not analyze the customers' churn decision through time, it is important to mention that multivariate times series data have triggered innovative techniques last years in the context of churn. Indeed, it is reasonable to hypothesis that the modifications of customers' behavior can be detected during the time leading to a churn decision. To deal with multivariate times series, several techniques were proposed that are based either on the featurization of the time series data to construct a tabular dataset or on dimension reduction combined with a binary classifier [43, 44]. More recently, Wang *et al.* [45] propose to use recurrent neural networks to tackle the time series data classification task. Finally, Óskarsdóttir *et al.* [46] designed extensions of the similarity forest method and successfully applied them for classifying multivariate time series data for churn prediction.

3.2. Machine learning for churn profiling

Recently, several studies focused on churn prediction models that can reach a good trade-off between the prediction performance and the results interpretability in terms of customers profile. As an example, De Caigny *et al.* [28] designed the Logit Leaf Model (`LLM`), which consists in two phases, namely a segmentation phase followed by a prediction step. For `LLM`, the segmentation is based on the partitioning obtained at the leaves of a decision tree that exploits the churn label from the input data. Then, for each data subset a logistic regression model is fitted which offers prediction and interpretability capabilities. `LLM` also include a random undersampling and a features selection phase. The authors provide experimental results on fourteen datasets ranging from the Financial Service to Telecommunication industry.

Following on `LLM` proposal, Ullah *et al.* [29] designed a churn prediction model using Random Forest which aims at providing both interpretability and prediction efficiency. The authors performed customers profiling using *k*-means and partition the data into three groups labeled as *Low*, *Medium* and *Risky* churners. As `LLM`, Ullah *et al.*'s `RF`-based model includes features selection. Customer churn data have usually a complex structure which reflects a strong class imbalance and also an intrinsic data segmentation due to the multiplicity of customer behavior patterns. Let us remember that the standard *k*-means algorithm considers the uniform spherical Gaussian mixture model with equal proportions. Hence, when the clusters are not easily separable, one should depart from the standard *k*-means assumptions by using novel representations that takes into a account the non linearity of the underlying data structure.

Successful clustering strategies have proposed to rely on Deep AutoEncoders [47, 48] (DAE) to handle data that require weak assumptions regarding the clusters shapes and filter out irrelevant features [49, 50]. Deep AutoEncoders can generate a more cluster-friendly representation of the data (or *encoding*) in an unsupervised manner while automatically learning important features. This type of *self-supervised* neural network is trained to replicate its input at output while optimizing a cost function. Several works have proposed to combine deep embeddings and clustering in a sequential way or within a joint optimization. Stacked DAEs were successfully used to learn the representation of an affinity graph before running *k*-means on the learned representations in order to identify clusters [51]. In [52] the authors incorporate a DAE into the Deep Embedded Clustering (DEC) framework [53] to jointly learn features and clustering. A novel ensemble method was introduced in [54] that uses *landmarks* and DAE to perform an efficient deep spectral clustering.

Customer data typically involved continuous and categorical features which should both be taken into account by the embeddings. In this work, we propose the use of a DAE loss function that jointly optimizes the novel representations based on categorical *and* continuous variables, which avoids the usual *dummification* pre-processing that can be damaging for the underlying data structure (Section 6.4).

3.3. Our contribution

In this work, we first evaluate multiple alternatives within a machine learning churn prediction pipeline composed of a sampling stage, a model fitting phase and a robust evaluation procedure (Figure 1; green and gray parts). While there exists deep machine learning variants that may sometimes competitive *traditional* approaches on specific datasets, we choose to focus solely on *traditional* machine learning that are less time-consuming and commonly used in the churn context. Hence, our comparative study involves the following models, for which we recall here the experimental studies or literature reviews that support their use in a churn context (see Section 3.3): Naïve Bayes [41], Logistic Regression [33, 55], *K*-nearest neighbors [42], Support Vector Machine [31, 41, 55], Decision Tree [32, 33, 31, 41], Random Forest [41, 55] and XGBoost [38, 37].

To tackle the imbalance issue [56, 57, 58], we associate each learning method with widespread sampling approaches to balance the classes distribution as it was shown to play a significant role in the performance of standard classifiers [27]. Based on theses experiments results, we can identify the most effective machine learning techniques and propose an ensemble method that can be successfully applied to a wide range of *churn-like* datasets.

Finally, following on recent developments in machine learning customer profiling [28, 29] and the promising results obtained with deep clustering approaches (Section 3.2), we demonstrate the effectiveness of our ensemble proposal on a segmented version of several churn benchmark datasets which makes it possible to directly draw conclusions on customer profile (Figure 1; green and blue parts).

All our experiments are performed with freely accessible Python packages (Appendix 7) and publicly available datasets exclusively (Table 1 & Appendix 7). Thus, our results are fully reproducible and the proposed procedure can be easily applied to novel datasets.

4. Data sampling

Attrition data typically requires the use of rebalancing techniques to change the distribution of classes. These methods usually consist in introducing instances into the minority class (*oversampling*), removing instances from the majority class (*undersampling*), or combining these two strategies (*hybrid*). Various rebalancing methods have been proposed [59] and several studies tend to show that undersampling performs better than oversampling [60].

4.1. Oversampling

This technique usually replicates the instances of the minority class or synthesizes new ones. Random oversampling is a straightforward approach that randomly selects the instances to be replicated. Yet, this can strongly degrade the quality of the decision boundary, by repeating for example outliers.

More advanced approaches have been proposed, such as Synthetic Minority Oversampling Technique (SMOTE) and Adaptive Synthetic Sampling (ADASYN, [61]). SMOTE oversamples the minority class by generating synthetic instances along the segments created by a *k*-NN approach. The new SMOTE instances are thus plausible observations that avoid overfitting. But these synthetic instances can be ambiguous in case of strong overlap between classes.

Extensions have been proposed to overcome this problem, including the widely used ADASYN, which generates *adaptively* minority instances based on their distribution. Thus, many instances are generated in regions of the feature space where the observation density is low, and vice versa.

4.2. Undersampling

The undersampling technique typically removes instances from the majority class or selects a subset. The Random Undersampling is a straightforward approach that randomly removes instances, which can lead to the removal of important instances.

More advanced strategies have been proposed, such as *Neighborhood Cleaning Rule* (NCR, [62]) and Tomek Links [63]. NCR combines two *rules* that remove redundant and ambiguous instances from the majority class. The first rule, *Condensed Nearest Neighbor* (CNN, [64]), selects a subset of instances from the majority class that cannot be classified correctly. These instances are considered relevant for learning. The second rule, *Edited Nearest Neighbors* (ENN, [65]), removes ambiguous instances via a $k-\texttt{NN}$ approach. A majority class instance misclassified by its neighbors is removed from the dataset. A minority class instance misclassified by its majority class neighbors implies the deletion of these same neighbors.

Tomek Links relies on CNN and identifies the *cross-class* pairs of instances. These are the pairs composed of an instance of the majority class and an instance of the minority class identified as its nearest neighbor. The majority instances that belong to the Tomek Links are noisy and should be deleted.

4.3. Hybrid strategies

Various combinations of oversampling and undersampling methods have been proposed to improve class separation by balancing the data. A simple hybrid method is to combine SMOTE with random undersampling. [59] have shown that this combination gives better results than the undersampling alone. A more complex combination, proposed by [66], associates SMOTE with Tomek Links. It has been successfully applied on an unbalanced genomic dataset.

5. Learning, Evaluation and Metrics

5.1. Supervised machine learning techniques

In this section, we provide an overview of the supervised machine learning techniques that were considered for our churn prediction and profiling pipeline. These approaches are of great interest in the attrition context. We only consider approaches that do not involve weights to compensate for class imbalance

and choose instead to mitigate the imbalance issue with resampling approaches.

Thus, for this study, we compare the performance of seven supervised machine learning techniques including two ensemble approaches. The considered models are the naive Bayesian classifier (`Gnb`, [67, 68]), the Logistic Regression (`LR`), the *k*-Nearest Neighbors (*k*-`NN`), the Support Vector Machine (`SVM`, with and without kernel, [69]), and the Decision Trees (`DT`, [70, 71]).

We also consider two ensemble methods, namely *Random Forest* (`RF`, [71]) and *eXtreme Gradient Boosting* (`XGBoost`, [72]), which generally perform well on attrition data [73, 74]. `RF` relies on *bagging* [75] and constructs C deep decision trees from C training sets obtained by *bootstrap*. Their predictions are combined by a majority vote. A known shortcoming of bagging is the tendency for classifiers to be correlated because they share the same set of properties. `RF` decorrelates the trees by forcing them to learn on a randomly chosen subset of the properties. As shown in [29], this approach also provides an interesting mean to identify the most important features and thus decompose several customer behavioral patterns.

`XGBoost` incorporates the *boosting* method which, like bagging, combines the results of several classifiers. However, in the boosting strategy, each model tries to minimize the errors of the previous model. Well-known variants of boosting are *Adaboost*, *gradient boosting* and *stochastic gradient boosting*. Instead of adjusting the weights like *Adaboost*, the *gradient boosting* variant optimizes a cost function, while the *stochastic gradient boosting* strategy adds observations and variable sampling at each iteration. `XGBoost` is the most widely used implementation for boosting.

5.2. Evaluation

A simple evaluation process is the "holdout set", where a subset of the data which is not used for training is used to evaluate the predictions of the trained model. A drawback is that some of the data is lost to training. The *cross-validation* strategy alleviates this problem by defining a training set and a validation set, then swapping these sets before combining the two evaluation scores. This idea can be extended to multiple subsets or *folds*. The data is divided into K *folds* of equivalent size and the model is trained on $K-1$ *folds*. The prediction error of the model is then calculated on the K^{th} subset. This strategy is repeated K times before combining the K estimates. This strategy is named the K-fold cross-validation (with typically $K = 5$ or 10).

However, this validation is not appropriate for attrition data because of the high imbalance. In this case, *stratified folds* should be produced to guarantee that each *fold* will respect the initial distribution of classes. The stratified K-fold cross-validation ($K = 5$) is the strategy chosen for our experiments.

5.3. Metrics

The *top decile-lift* and the *Gini coefficient* are the preferred evaluation measures used by marketing departments to evaluate predictive models. The *lift* considers instances in the order of their probability of being in the minority class. Focusing on the 10% of riskiest customers, the top decile lift gives the proportion of churned customers in the risky segment, $\pi_{10\%}$, divided by the total proportion of churners in the validation set, π, $lift_{10\%} = \hat{\pi}_{10\%}/\hat{\pi}$. This measure assesses whether the clients predicted to be risky are actually risky. The Gini coefficient takes into account the risky and less risky customers.

In machine learning, the score $\mathbf{F_1}$ and the Area Under the Curve (AUC) are two recommended metrics in the context of attrition. $\mathbf{F_1}$ is the harmonic mean of the *Precision* and the *Recall*. The *Precision* estimates the ability of the model to obtain positive *true* among its positive predictions. This measure is complementary to the *Recall* which estimates the ability of the model to recover positive *true* predictions. The AUC requires to express the performance of the model by a Receiver Operating Characteristic (ROC) curve which gives the true positive rate as a function of the false positive rate for a series of decision thresholds. It thus provides an aggregate performance measure for all possible classification thresholds. The AUC, suitable for unbalanced datasets, is the measure chosen for our experiments.

6. Experiments on public datasets

Our 13 public datasets have churn percentages ranging from 0.07% to 0.50% (Table 1, *%churn* column), and accessible online (Table 1, *Access* column). We have kept the default parameters, as provided in the Python packages scikit-learn (0.23.2) and `xgboost` (1.0.2), for the 8 learning techniques considered; Logistic Regression (`LR`), Linear Support Vector Machine (`SVM`) and with Radial Basis Function (`SVM-rbf`), Naive Bayesian Classifier (`Gnb`), Random Forest (`RF`), Decision Tree (`DT`), eXtreme Gradient Boosting (`XGBoost`) and K-nearest Neighbors (*k*-`NN`). These approaches are evaluated without and with rebalancing (oversampling/undersampling and hybrid strategies; (see Figure 1, *Sampling*).

6.1. Comparison of classification algorithms

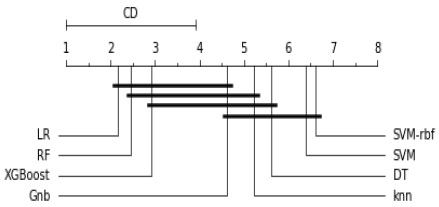
We evaluated the churn prediction for all combinations of the Figure 1 processing chain (gray and green parts). The evaluation follows a 5-fold stratified cross-validation. Results were provided in AUC without sampling, with different over-/under-sampling and with hybrid sampling approaches. The Tables 2 & 3 were extracted from the results obtained with the 8 resampling approaches. The average rank and median AUC ($\widetilde{AUC}$) of each algorithm are shown in the last two columns. Our experiments indicate that the $\widetilde{AUC}$ is overall little affected by the rebalancing mode. For `RF`, resampling globally degrades the $\widetilde{AUC}$ compared to the results obtained without sampling (from 0.8095 to 0.7862). On average, the performance of `XGBoost` is slightly improved with Tomek Links (+0.0014; Table 3) or SMOTE combined with NCR (+0.0048; not shown).

The approach that benefits the most from the resampling strategies is `LR`, with a maximum increase for the $\widetilde{AUC}$ of 0.0051 using NCR. Finally, the three best approaches - regardless of dataset and rebalancing strategy - are `LR`, `XGBoost`, and `RF`, with an average rank of 2.01, 2.74, and 2.94 respectively.

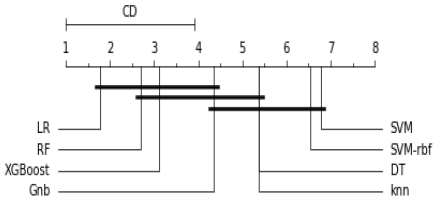
Table 2: AUC results and CD diagrams from our pipeline experiments (no sampling vs. oversampling).

Data	Bank	C2C	DSN	HR	K2009	KKBox	Member	Mobile	SATO	TelC	TelE	UCI	News	$\overline{Rang}$	$\widetilde{AUC}$
	No Sampling														
k-NN	0.7768	0.4387	0.6576	0.6575	0.5004	0.5835	0.5827	0.7567	0.6900	0.7822	0.8226	0.7731	0.7484	5.23	0.6900
Gnb	0.7166	0.5181	0.6671	0.7442	0.5002	0.6468	0.5914	0.7201	0.7272	0.8245	0.7505	0.8477	0.5655	4.62	0.7166
LR	0.8322	**0.5222**	0.7319	**0.8596**	**0.5135**	0.6763	**0.6146**	**0.9030**	0.7594	**0.8458**	0.7584	0.8244	0.8369	**2.15**	**0.7594**
SVM	0.6645	0.4578	0.6868	0.8091	0.5052	0.5022	0.4874	0.4605	0.7116	0.6498	0.5335	0.5963	0.5958	6.38	0.5958
SVM-rbf	0.7248	0.4656	0.6293	0.4984	0.4989	0.4983	0.5088	0.5463	0.7153	0.6548	0.6098	0.7528	0.6227	6.62	0.6098
DT	0.6908	0.4440	0.7350	0.6053	0.4993	0.5302	0.5462	0.6660	0.6365	0.6555	0.8514	0.8447	0.6754	5.62	0.6555
RF	**0.8506**	0.3518	**0.8590**	0.7867	0.5114	0.6442	0.6130	0.8095	**0.7882**	0.8210	0.9380	**0.9182**	**0.8615**	**2.46**	**0.8095**
XGBoost	0.8216	0.3862	0.8516	0.7993	0.5112	**0.6800**	0.5987	0.7816	0.7396	0.7983	**0.9411**	0.9174	0.8323	**2.92**	**0.7983**
Max-Min	0.1861	0.1704	0.2297	0.3612	0.0146	0.1817	0.1272	0.4425	0.1517	0.1960	**0.4076**	0.3219	0.2960		
	SMOTE *(oversampling)*														
k-NN	0.7744	0.4375	0.6576	0.6631	0.5001	0.5918	0.5865	0.6479	0.6900	0.7650	0.8277	0.7871	0.7452	5.38	0.6631
Gnb	0.7861	0.5033	0.6671	0.7168	0.4991	0.6430	0.5936	0.6993	0.7272	0.8224	0.7497	0.8273	0.5664	4.38	0.6993
LR	**0.8325**	**0.5160**	0.7319	**0.8501**	**0.5135**	**0.6763**	**0.6213**	**0.8942**	0.7594	**0.8451**	0.7626	0.8278	**0.8336**	**1.77**	**0.7626**
SVM	0.583	0.4965	0.6868	0.7066	0.4965	0.559	0.5176	0.6185	0.7116	0.5098	0.5470	0.5327	0.5651	6.77	0.5590
SVM-rbf	0.7204	0.4751	0.6298	0.5040	0.4993	0.4370	0.5187	0.4404	0.7152	0.6881	0.5692	0.7729	0.6337	6.62	0.5692
DT	0.6940	0.4415	0.7314	0.6309	0.5022	0.5272	0.5489	0.6570	0.6385	0.6656	0.8482	0.8490	0.6881	5.38	0.6570
RF	0.8255	0.3944	0.8166	0.7304	0.5023	0.6129	0.6122	0.8138	**0.7601**	0.8007	0.9373	0.9130	0.8136	**2.69**	**0.8007**
XGBoost	0.8234	0.3878	**0.8516**	0.7905	0.4991	0.6414	0.5959	0.7835	0.7396	0.7941	**0.9421**	**0.9154**	0.8333	**3.08**	**0.7905**
Max-Min	0.2495	0.1282	0.2218	0.3461	0.0170	0.2393	0.1037	0.4538	0.1216	0.3353	0.3951	0.3827	0.2685		
	ADASYN *(oversampling)*														
k-NN	0.7647	0.4408	0.6576	0.6612	0.5007	0.5899	0.5791	0.6203	0.6900	0.7515	0.8248	0.7791	0.7377	5.38	0.6612
Gnb	0.7865	0.5031	0.6671	0.7241	0.4987	0.6421	0.5958	0.6814	0.7272	0.8311	0.7551	0.8293	0.5661	4.38	0.6814
LR	**0.8315**	0.5171	0.7319	**0.8476**	**0.5137**	**0.6777**	**0.6266**	**0.8848**	**0.7594**	**0.8444**	0.7634	0.8276	0.8309	**2.00**	**0.7634**
SVM	0.6403	**0.5271**	0.6869	0.6768	0.5032	0.5491	0.5015	0.1398	0.7116	0.4093	0.4678	0.5512	0.5467	6.31	0.5467
SVM-rbf	0.7123	0.4734	0.6297	0.5026	0.5053	0.5239	0.5304	0.4864	0.7153	0.6822	0.5559	0.7601	0.6419	6.23	0.5559
DT	0.6865	0.4401	0.7336	0.5814	0.4985	0.5268	0.5479	0.6644	0.6375	0.6546	0.8382	0.8483	0.6876	5.69	0.6546
RF	0.8197	0.3971	0.8038	0.7597	0.4945	0.6107	0.6092	0.7970	0.7494	0.8003	0.9364	0.9112	0.8107	**3.31**	**0.7970**
XGBoost	0.8225	0.3905	**0.8516**	0.7978	0.5013	0.6468	0.5973	0.7937	0.7396	0.7968	**0.9418**	**0.9156**	**0.8328**	**2.69**	**0.7968**
Max-Min	0.1912	0.1366	0.2219	0.3450	0.0192	0.1538	0.1251	**0.7450**	0.1219	0.4351	0.4740	0.3644	0.2861		

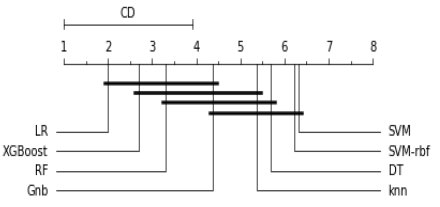
(a) No sampling



(b) SMOTE



(c) ADASYN



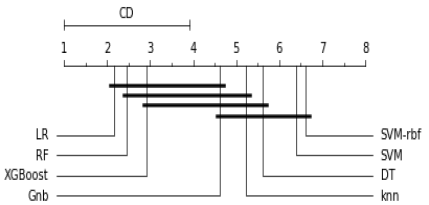
For some *dataset/technique* pairs, we can observe a more significant improvement. For example, the combination of `SVM` & NCR increases the *AUC* by 0.1081 on *C2C* (Table 3). The *AUC* of `XGBoost` also increases when using the hybrid sampling SMOTE & Tomek Links (from 0.8516 to 0.8694; not shown) on *DSN*. We find an increase in *AUC* of 0.0124 using SMOTE & NCR on *Member* with `LR`. Therefore, while an overall improvement in all learning approaches cannot be observed, there are *local* improvements, depending on the datasets.

We propose to visualize the similarities and rankings of the learning techniques via Critical Difference diagrams (CD, [76]) based on pairwise statistical comparisons computed from all our *AUC* results (Tables 2 & 3; bottom Figures). For these comparisons, we consider the Nemenyi test ($\alpha = 0.05$). Horizontal lines connect the approaches for which we cannot exclude the hypothesis that the mean ranks are equal. The DC plots re-

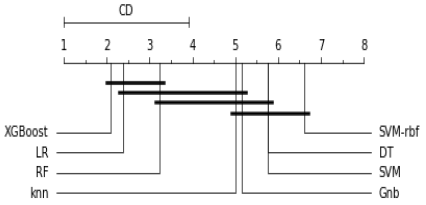
Table 3: AUC results from our pipeline experiments (no sampling vs. undersampling).

Data	Bank	C2C	DSN	HR	K2009	KKBox	Member	Mobile	SATO	TelC	TelE	UCI	News	$\overline{Rang}$	$\widetilde{AUC}$
	No Sampling														
k-NN	0.7768	0.4387	0.6576	0.6575	0.5004	0.5835	0.5827	0.7567	0.6900	0.7822	0.8226	0.7731	0.7484	5.23	0.6900
Gnb	0.7166	0.5181	0.6671	0.7442	0.5002	0.6468	0.5914	0.7201	0.7272	0.8245	0.7505	0.8477	0.5655	4.62	0.7166
LR	0.8322	**0.5222**	0.7319	**0.8596**	**0.5135**	0.6763	**0.6146**	**0.9030**	0.7594	**0.8458**	0.7584	0.8244	0.8369	**2.15**	**0.7594**
SVM	0.6645	0.4578	0.6868	0.8091	0.5052	0.5022	0.4874	0.4605	0.7116	0.6498	0.5335	0.5963	0.5958	6.38	0.5958
SVM-rbf	0.7248	0.4656	0.6293	0.4984	0.4989	0.4983	0.5088	0.5463	0.7153	0.6548	0.6098	0.7528	0.6227	6.62	0.6098
DT	0.6908	0.4440	0.7350	0.6053	0.4993	0.5302	0.5462	0.6660	0.6365	0.6555	0.8514	0.8447	0.6754	5.62	0.6555
RF	**0.8506**	0.3518	**0.8590**	0.7867	0.5114	0.6442	0.6130	0.8095	**0.7882**	0.8210	0.9380	**0.9182**	**0.8615**	**2.46**	**0.8095**
XGBoost	0.8216	0.3862	0.8516	0.7993	0.5112	**0.6800**	0.5987	0.7816	0.7396	0.7983	**0.9411**	0.9174	0.8323	**2.92**	**0.7983**
Max-Min	0.1861	0.1704	0.2297	0.3612	0.0146	0.1817	0.1272	0.4425	0.1517	0.1960	**0.4076**	0.3219	0.2960		
	Neighborhood Cleaning Rule *(undersampling)*														
k-NN	0.7994	0.4069	0.6634	0.6761	0.5061	0.6099	0.5915	0.7274	0.7028	0.8028	0.8295	0.8052	0.7804	5.00	0.7028
Gnb	0.7460	0.4890	0.6328	0.7350	0.5004	0.6483	0.5886	0.7255	0.7348	0.8205	0.7468	0.8512	0.5672	5.15	0.7255
LR	0.8313	0.4985	0.7311	**0.8580**	0.5146	**0.6762**	**0.6209**	**0.8867**	**0.7645**	**0.8438**	0.7615	0.8234	0.8371	**2.38**	**0.7645**
SVM	0.6647	**0.5659**	0.7186	0.8332	0.5017	0.5353	0.4915	0.4912	0.7741	0.8007	0.4438	0.6309	0.6727	5.77	0.6309
SVM-rbf	0.7938	0.4533	0.6308	0.4984	0.5033	0.4797	0.5512	0.6077	0.7089	0.7920	0.6260	0.6288	0.6745	6.62	0.6260
DT	0.7327	0.4146	0.7214	0.6194	0.5027	0.5488	0.5693	0.6710	0.6615	0.7136	0.8583	0.8500	0.7306	5.77	0.6710
RF	0.8361	0.3527	0.8173	0.7430	0.5105	0.6397	0.6129	0.7862	0.7631	0.8201	0.9394	0.9145	0.8298	**3.23**	**0.7862**
XGBoost	**0.8369**	0.3668	**0.8672**	0.7918	**0.5149**	0.6824	0.6104	0.7745	0.7685	0.8216	**0.9417**	**0.9200**	**0.8399**	**2.08**	**0.7918**
Max-Min	0.1722	0.2132	0.2364	0.3596	0.0145	0.2027	0.1294	**0.3955**	0.1126	0.1302	0.4979	0.2912	0.2727		
	Tomek Links *(undersampling)*														
k-NN	0.7797	0.4359	0.6535	0.6671	0.4999	0.5873	0.5890	0.7514	0.6891	0.7882	0.8236	0.7773	0.7533	5.38	0.6891
Gnb	0.7196	0.5164	0.6632	0.7426	0.5002	0.6470	0.5924	0.7182	0.7247	0.8240	0.7501	0.8487	0.5653	4.69	0.7182
LR	**0.8321**	**0.5208**	0.7286	**0.8585**	**0.5138**	0.6761	**0.6170**	**0.8991**	**0.7573**	**0.8459**	0.7589	0.8252	**0.8376**	**1.92**	**0.7589**
SVM	0.5793	0.4803	0.7000	0.8260	0.5007	0.5335	0.4801	0.3813	0.7253	0.7019	0.5695	0.6336	0.6010	6.31	0.5793
SVM-rbf	0.7500	0.4567	0.6241	0.4990	0.4961	0.4762	0.5162	0.5211	0.7029	0.7055	0.6031	0.7540	0.6395	6.69	0.6031
DT	0.6963	0.4427	0.7293	0.6152	0.5044	0.5337	0.5474	0.6619	0.6415	0.6683	0.8543	0.8431	0.6909	5.31	0.6619
RF	0.8243	0.3863	0.8294	0.7481	0.5106	0.6189	0.6036	0.788	0.7483	0.8001	0.9379	0.9134	0.8132	**3.08**	**0.7880**
XGBoost	0.8253	0.3855	**0.8655**	0.7997	0.5017	**0.6805**	0.6033	0.7868	0.7514	0.8017	**0.9412**	**0.9150**	0.8365	**2.77**	**0.7997**
Max-Min	0.2528	0.1353	0.2414	0.3595	0.0177	0.2043	0.1369	0.5178	0.1158	0.1776	0.3717	0.2814	0.2723		

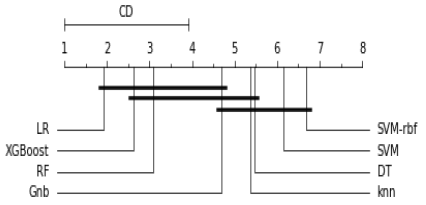
(a) No sampling



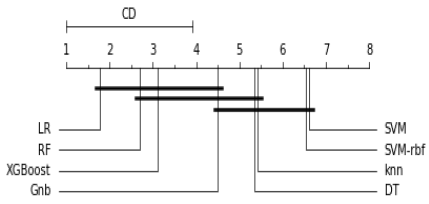
(b) Neighborhood Cleaning Rule



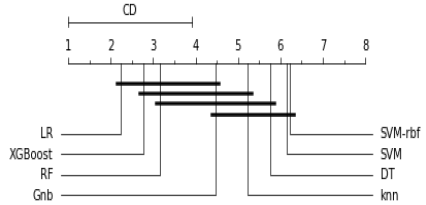
(c) Tomek



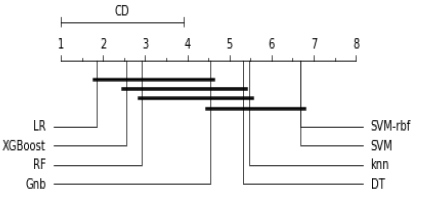
(d) SMOTE + Random Undersampling



(e) SMOTE + NCR



(f) SMOTE + Tomek



flect well the fact that sampling strategies have little effect on the ranking of learning approaches. We can also easily see that RF is one of the two best approaches if we use SMOTE alone, or in combination with random undersampling, or without any sampling.

6.2. Models and datasets CA

Based on AUC values, we propose to use Correspondence Analysis (CA) [77, 78] to visualize the relationships between learning techniques and datasets. CA can assist in deciphering the complex information contained in Table 3; it provides a way for suggesting hypotheses and recommendations to the user. Thereby Fig. 3 gives an overview of the results of this analysis and highlights about the similar or opposite behaviors of methods according to the characteristics of the data. Below we provide the salient interpretations from axes 1, 2 which bring 85% of the total variance.

Fig. 3(a) shows similar behavior of RF and XGBoost. First, recall that in Fig. 2(b), axis 3 is characterized by *nb_instances* and TelE is distinguished by a high *nb_instances*. In Fig. 3(a), by interpreting axis 1, we deduce that both methods (on the left) outperform the other methods for TelE that has higher number of instances, few variables and not complex. However, their performances decrease significantly with C2C (on the right) that has much less instances, more variables and more complex given the recorded results by all methods. Note that, neither method of both is superior to the other. Furthermore, as for K2009 and SATO which are well presented but not sufficiently eccentric to be preponderant in the interpretation, like TelE and C2C, means that both methods are indistinguishable from the others for the two datasets.

On the other hand, Fig. 3(a) also highlights the difference between SVM and SVM-rbf on axis 2; SVM-rbf appears significantly more effective than SVM when the number of categorical variables is preponderant (it is the case of the UCI dataset well represented on this axis), however, this superiority declines with the presence of continuous variables in favor of SVM as it is the case for example of the HR dataset.

On axis 3, we mainly observe an interesting opposition between Mobile and DSN. This opposition has been also observed on axis 4 of Fig. 3(d) which is essentially described by the number of continuous variables. Thus LR which is opposed to XGboost and RF is the only method mainly characterizing this axis, appears as a powerful method for data having a high rate of number of continuous variables compared to categorical variables which is the case of Mobile and not that of DSN (11.4 vs. 0.47).

Considering the different analyzes of our experiments, we have identified three complementary methods LR, XGBoost and RF taking into account the different characteristics discussed in section 2.2 and observed in Fig. 3. This reinforces our belief that an ensemble approach grouping the three methods could perform well on a large part of the churn datasets.

6.3. Comparative study of ensemble methods

The conclusions of the previous sections motivate the combination of LR, XGBoost and RF in an ensemble approach for churn prediction. More specifically, we compute for each observation the average of the probabilities predicted by two or three techniques taken from these three approaches following the *soft voting* [79] technique described below.

For instance, let us consider an ensemble of 3 models M^1, M^2 and M^3. Using the soft voting, the expected score $\hat{y}_{ens}$ is then expressed as a weighted sum of the individual scores,

$$\hat{y}_{ens} = \omega_1 \hat{y}_{M^1} + \omega_2 \hat{y}_{M^2} + \omega_3 \hat{y}_{M^3}, \tag{2}$$

where

$$\omega_1, \omega_2, \omega_3 = softmax(\widetilde{\omega_1}, \widetilde{\omega_2}, \widetilde{\omega_3}), \tag{3}$$

with

$$\widetilde{\omega_k} = \frac{1}{\rho(\hat{\mathbf{Y}}_{M^k}, \hat{\mathbf{Y}}_{M^\alpha}) + \rho(\hat{\mathbf{Y}}_{M^k}, \hat{\mathbf{Y}}_{M^\beta})}. \tag{4}$$

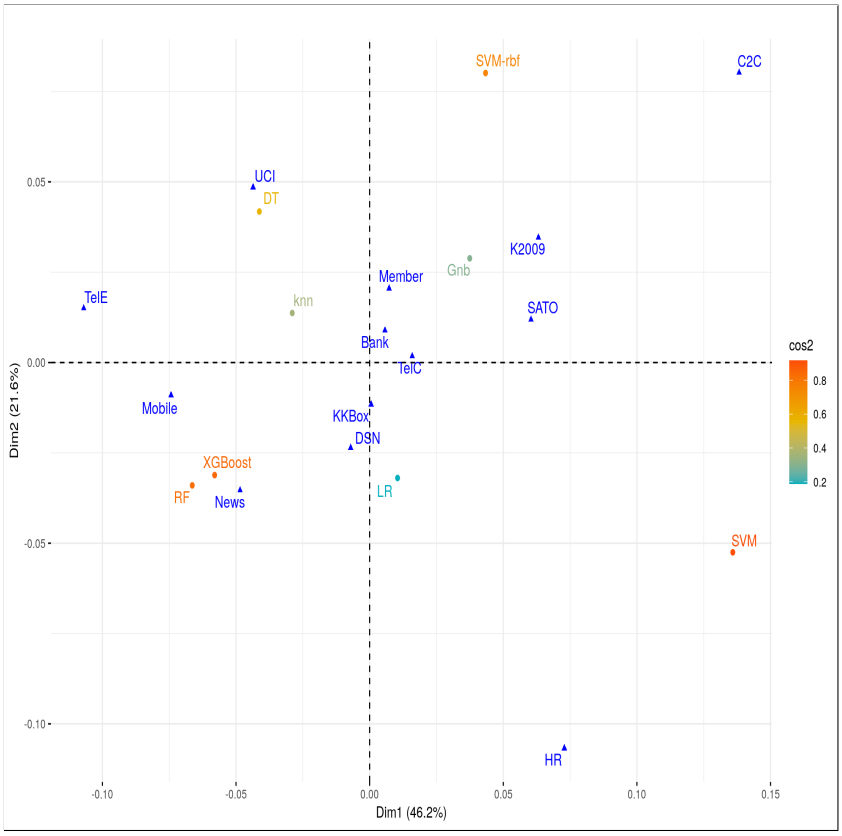
where ρ denotes the Pearson correlation, $\alpha \neq k$ and $\beta \neq k$.

Fig. 4 shows, for each resampling strategy, and for all datasets, the AUC for LR, XGBoost and RF (light gray), their pairwise ensembles (light orange), and the combination of the three methods (dark orange). It appears that the ensemble approach LR|XGBoost|RF performs best, followed by the pairwise ensemble approaches, LR|XGBoost and LR|RF. Table 4 shows the $\widetilde{AUC}$ on all datasets for both ensemble and non-ensemble approaches. While the results for XGBoost and RF outperform those for LR ($\widetilde{AUC}$ of 0.7956 and 0.7953 vs. 0.7622), combining the XGBoost and RF approaches does not significantly increase the $\widetilde{AUC}$ (0.8061). By contrast, the addition of LR in the ensemble approach (LR|XGBoost and LR|RF) significantly increases the $\widetilde{AUC}$ (0.8413 and 0.8365 respectively). Overall, the best approach together is LR|XGBoost|RF, combining the three techniques, with the oversampling strategy ADASYN ($\widetilde{AUC} = 0.8483$).

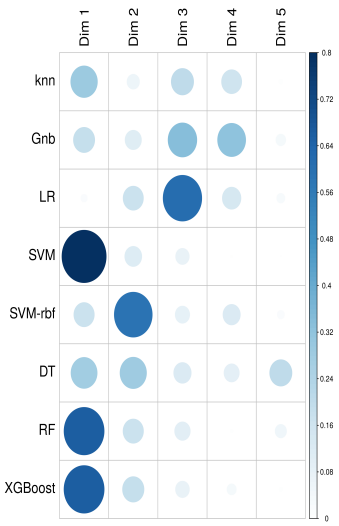
Table 5 provides the pipeline variant that produces the best AUC for each dataset (columns "Best Processing Chain" and "Best AUC"). The ensemble approach we recommend– LR|XGBoost|RF & ADASYN - provides an AUC score very close to that of the best approach. The only exception is for *C2C*, for which the set LR|Gnb without resampling is a better choice ($AUC = 0.5247$). Thus, in practice, we recommend using the ensemble approach LR|XGBoost|RF with ADASYN for exploring new attrition datasets.

It is worth noting that LR|XGBoost and LR|RF also provide high AUC results, without sampling. Their AUC almost reach the AUC of the ensemble proposal that lumps the three machine learning models (0.8422 and 0.8440 versus 0.8483). This is in line with the high similarity of the XGBoost and RF behavior, which is clearly shown by their positioning on the planes (1.2) and (1.3) of the CA biplots (Fig. 3).

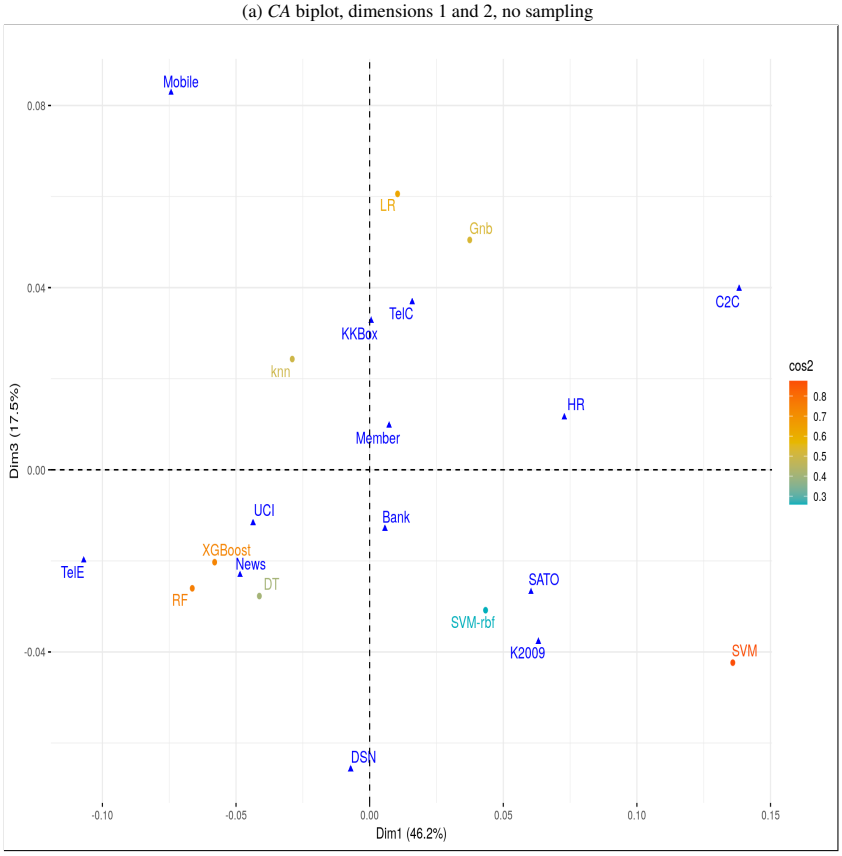
Next, we provide a detailed scheme that describes the embeddings learning process of numerical and categorical features while combining simultaneously embedding and clustering tasks. To do this we rely on entity embedding and deep clustering.



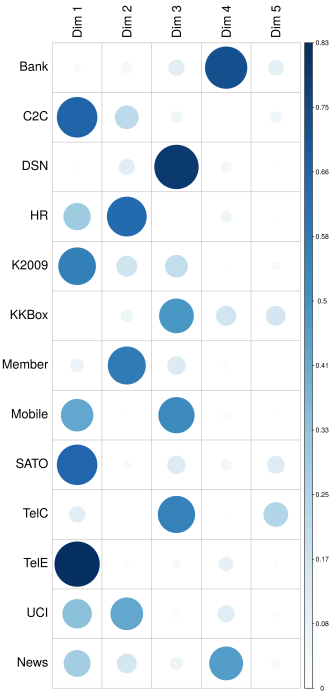
(a) *CA* biplot, dimensions 1 and 2, no sampling



(b) Representation Quality



(c) *CA* biplot, dimensions 1 and 3, no sampling



(d) Representation Quality

Figure 3: (a & c) Visualization of associations between machine learning approaches and churn-like datasets without sampling using Correspondance Analysis. (b & d) Quality of representations on the factor map.

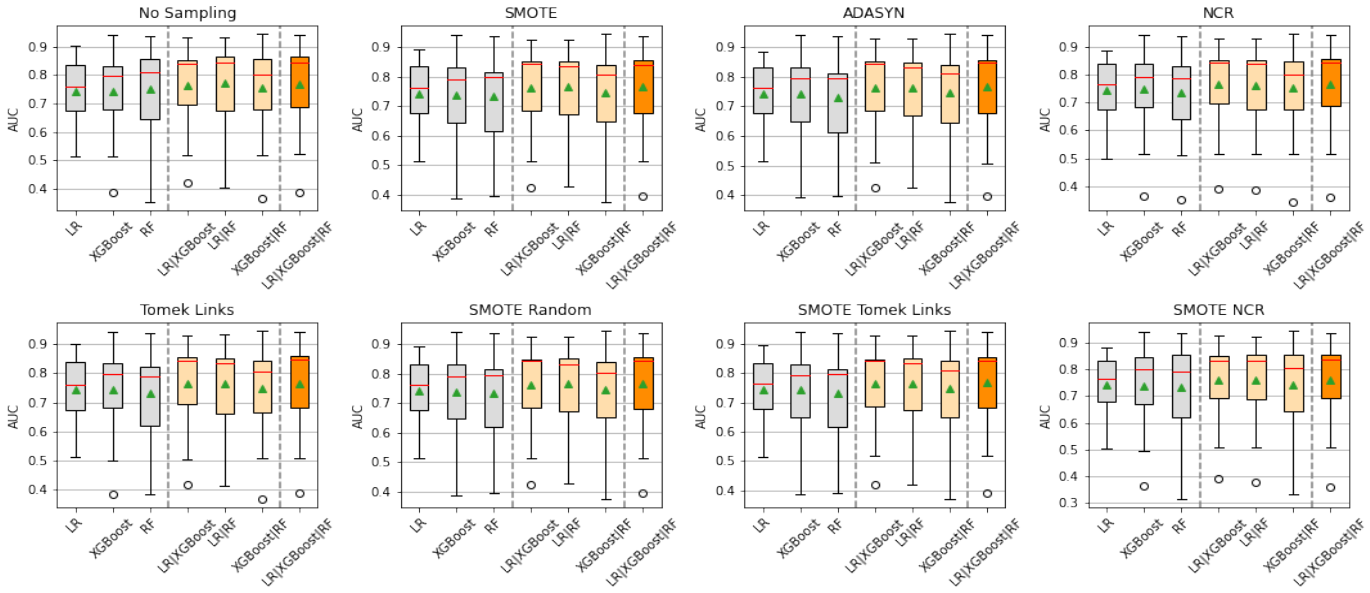


Figure 4: AUC ensemble results on the three top machine learning approaches and all datasets.

Table 4: $\widetilde{AUC}$ for *ensemble* and *non ensemble* approaches and all datasets; the best results are in bold and the second ones results are underlined.

Sampling	no sampling	SMOTE	ADASYN	NCR	Tomek Links	SMOTE & R.U.	SMOTE & Tomek	SMOTE & NCR	$\widetilde{AUC}$
LR	0.7594	0.7626	0.7634	0.7645	0.7589	0.7626	0.7628	0.7633	0.7622
XGBoost	0.7983	0.7905	0.7968	0.7918	0.7997	0.7905	0.7939	0.8031	0.7956
RF	0.8095	0.8007	0.797	0.7862	0.788	0.7947	0.7951	0.7911	0.7953
LR|XGBoost	0.8422	**0.8422**	0.8422	**0.8416**	0.8433	0.8422	0.8419	0.8346	0.8413
LR|RF	0.8440	0.8369	0.8339	0.8402	0.8358	0.8334	0.8349	0.8325	0.8365
XGBoost|RF	0.8028	0.8078	0.8115	0.8015	0.8055	0.8047	0.8094	0.8055	0.8061
LR|XGBoost|RF	**0.8443**	0.8401	**0.8483**	0.8413	**0.8468**	**0.8453**	**0.8434**	**0.8374**	**0.8434**

Table 5: Our ensemble proposal vs. best non ensemble approach for each dataset.

	LR|XGBoost|RF & ADASYN	Best AUC	Best Processing Chain
Bank	0.8492	0.8506	no sampling & RF
C2C	0.3962	0.5659	NCR & SVM
DSN	0.8486	0.8694	SMOTE-T.Links & XGBoost
HR	0.8483	0.8596	no sampling & LR
K2009	0.5070	0.5153	SMOTE-NCR & LR
KKBox	0.6778	0.6805	Tomek Links & XGBoost
Member	0.6209	0.6270	SMOTE-NCR & LR
Mobile	0.8788	0.9030	no sampling & LR
SATO	0.7703	0.7882	no sampling & RF
TelC	0.8302	0.8458	no sampling & LR
TelE	0.9408	0.9421	SMOTE & XGBoost
UCI	0.9214	0.9200	NCR & XGBoost
News	0.8574	0.8615	no sampling & RF
$\widetilde{AUC}$	0.8483	0.8506	

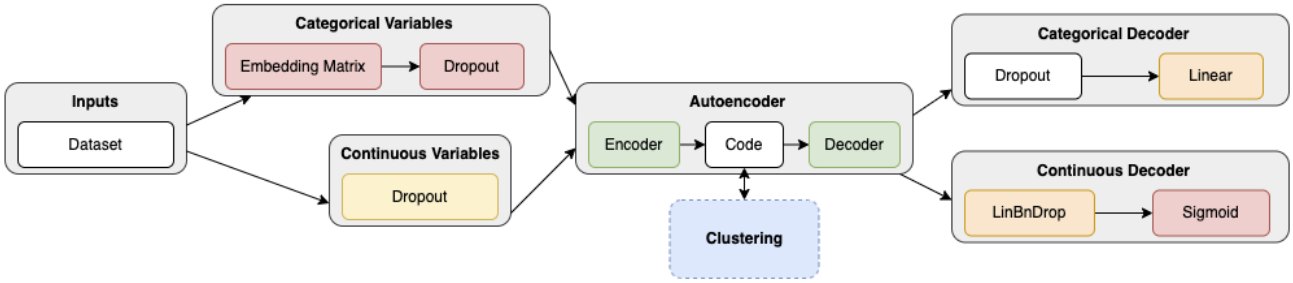


Figure 5: Deep network pipeline for the joint learning of instances embeddings and customer segmentation. *Adapted from the online course walkwithfastai.com*

6.4. Unsupervised machine learning techniques

An autoencoder is a neural network that is trained in an unsupervised or *self-supervised* manner. Its parameters are learned in such a way that the output values tend to replicate the input training samples. The internal hidden layer can be used as a low dimensional representation of the input which captures the more salient features. We can decompose an autoencoder in two parts, namely an *encoder* f_θ, followed by a *decoder* g_ψ. The first part provides the *encoding* of the input dataset by computing a feature vector $\mathbf{y}_i = f_\theta(\mathbf{x}_i)$ for each input training sample. Then, the encoding is transformed back to its original representation by the *decoder* part, following $\hat{\mathbf{x}}_i = g_\psi(\mathbf{y}_i)$. The sets of parameters for the encoder f_θ and the decoder g_ψ are learned simultaneously during the reconstruction task while minimizing the *loss*, referred to as $\mathcal{J}$ given by

$$\mathcal{J}_{AE}(\theta, \psi) = \sum_{i=1}^{n} \mathcal{L}(\mathbf{x}_i, g_\psi(f_\theta(\mathbf{x}_i))), \tag{5}$$

where $\mathcal{L}$ is a cost function for measuring the divergence between the input training sample and the reconstructed data. The encoder and decoder parts can have several shallow layers, yielding a deep autoencoder (DAE) that enables to learn higher order features. The network architecture of these two parts usually mirrors each other. Churn data typically contain numerical *and* categorical data. A straightforward manner for a neural network to process categorical input is by using the *one-hot encoding* strategy. However, as shown in [80], embeddings should be preferred to one-hot encoding vectors, as they reduce memory usage and speed up the neural network learning. Besides, embeddings can capture intrinsic properties of the categories and reveal relationship between them.

Inspired by Guo *et al.* [80] proposal, we adapt the *entity embedding* in a unsupervised context to automatically learn the representation of categorical features in multi-dimensional spaces which puts the feature's values with similar effect close to each other. Such an approach reveals the inherent continuity of the categorical data. Practically, it consists in *transforming* categorical columns (vectors of size n) into an embedding matrix (of size $n_{instances} \times embedding_{dim}$) taken from a neural network trained with those categories (Fig. 5). In this study, we set $embedding_{dim}$ to be 2 when the categorical variables have only two unique values, and to be $ceil(n_{unique} \times compression)$, where $compression = \frac{1}{2}$. We provide in Table 6 a toy example of entity embeddings obtained for two categorical variables cat^a ($n_{unique} = 2$) and cat^b ($n_{unique} = 4$), as done in our experiments.

Table 6: Toy example of an entity embeddings for 2 categorical variables

instance	cat^a	cat^b	$\mathbf{x}^{cat}$ *(entity embeddings)*			
			cat^a_0	cat^a_1	cat^b_0	cat^b_1
$i = 0$	1	2	0.002598	−0.012928	0.036055	−0.003408
$i = 1$	1	1	−0.015642	0.016857	0.036055	−0.019931
...	...	...	...	...	...	...
$i = n$	2	4	−0.015642	0.016857	0.013035	−0.019931

Thus, to optimize the customer segmentation while learning a combination of numerical and categorical features within a unique embedding, we train the parameters of a DAE as given in Fig. 5[2]. Inspired by [52] we propose to combine embedding and clustering simultaneously as depicted in Fig. 5. This respects the idea of improving embedding taking into account local structure preservation. Thereby the loss function to be minimized amounts to the sum of a reconstruction loss noted $\mathcal{J}_{DAE}$ and a clustering loss noted $\mathcal{J}_{clust}$ given by

$$\mathcal{J}_{DAE}(\theta, \psi) = \sum_{i=1}^{n} ||y_i - g_\psi(f_\theta(\mathbf{x}_i^{cont}))||_2^2 - \sum_{i=1}^{n} y_i \log(g_\psi(f_\theta(\mathbf{x}_i^{cat}))), \tag{6}$$

and

$$\mathcal{J}_{clust}(\theta, \psi) = \sum_{i=1}^{n} \sum_{k=1}^{G} r_{ik} ||g_\psi(\mathbf{x}_i) - \mu_k||_2^2, \tag{7}$$

with n the number of samples, G the number of clusters, $r_{ik} = 1$ if sample i belongs to cluster k, and the concatenation of the vectors $\mathbf{x}_i^{cont}$ and $\mathbf{x}_i^{cat}$ gives $\mathbf{x}_i$. Ultimately, for our experiments, each customers' segments is then split in train and test embeddings subsets, before the machine learning models are fit on the train part (see Section 6.5).

6.5. Ensemble method for profiling and prediction

In this section, we propose to associate our churn prediction ensemble method (`LR|XGBoost|RF`; Section 6.3) to a deep customer data profiling. Data are segmented based on the approach

[2] **Dropout** refers to cutting the connection to a set of random neurons in order to reduce overfitting; **LinBnDrop** is a sequence of linear layer and batch normalization that aims at standardizing the input to improve training and dropout. [81].

described in Section 6.4 (see also Fig. 5) that jointly learns a k-means partitioning (G clusters) with DAE encodings and entity embeddings. Each cluster C_i is split into a C_i^{train} train set and a C_i^{test} test set, in a stratified manner (80%/20%). The aggregated score of the models $\{M^j\}_{j=1..m}$ is then used to predict churn behavior on each segment C_i, and the average of all the test sets AUC_i provides the overall AUC prediction result (Fig. 6). In a supervised churn prediction context, labels are already known for our benchmark datasets and can be used for the model evaluation. In practice, novel observations for which the company requires a label would correspond to our test subsets.

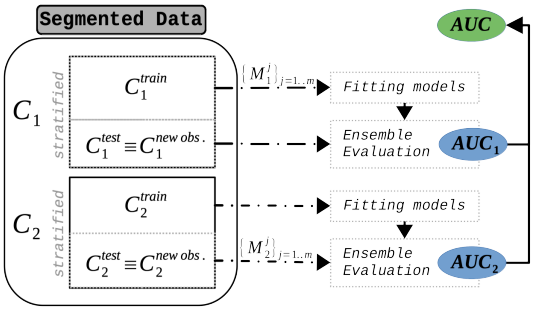


Figure 6: Evaluation of the ensemble profiling and prediction approach. Two customers subgroups are identified in an unsupervised manner ($G = 2$; see Section 6.4). In practice, test subsets would correspond to novel observations for which the company expects a label

6.5.1. *Quantitative evaluation of churn prediction*

We compare our approach to state-of-the-art methods in the context of churn, namely `LLM` and Ullah's `RF` model. For `LLM`, we used the implementation provided by the `LLM` R package (V1.1.0) [3]. Ullah's `RF`-based model was implemented following the author's description and based on *scikit-learn* Python package. We performs 50 runs on all benchmark datasets [4] for the compared approaches. Table 7 summarizes AUC results for different number of clusters (from $G = 2$ to $G = 6$). As can be seen, our ensemble proposal combined with the DAE data segmentation outperforms the competitive approaches, with $\widetilde{AUC}$ between 0.8516 and 0.8546, while `LLM` and the `RF`-based model reach 0.8450 and 0.8317 respectively. It should be highlighted that `LLM` encounters difficulty to handle several datasets, for which its execution could exceed 3 hours (vs. less than half an hour on average) and our experiments should be stopped before the convergence of the approach (Table 7, *overtime* labels).

The AUC value embeds two metrics which are the *Precision* and the *Recall*. While the Precision estimates the ability of the model to obtain actual churners among its predicted churners, the *Recall* estimates the ability of the model to recover actual churners. Usually, the cost of a false positive in the churn context is considered as less damaging than the non identification of actual churners. Indeed, contacting loyal customers to propose them with several advantages such as discounts generally reinforce their loyalty at a fixed cost.

Yet, missing an actual churner could represent a significant loss of profit. Hence, *Recall* is, along with AUC, an important metric to consider when building a churn prediction model. Table 8 summarizes the Recall of our ensemble approach combined with data segmentation. Our proposal outperforms the *Recall* of `LLM` and Ullah's `RF`-based model.

6.5.2. *Qualitative evaluation of churn profiling*

Beyond the performance in churn prediction for our ensemble approach, it is important to highlight the benefit of the data segmentation in terms of customers profiling. Indeed, the partitioning of the customer data puts forward the most important features on which the M^i models are fitted. These features can be further assigned to subgroups of churners and non churners within each cluster. Hence, proactive marketing campaigns could be designed to target a group of both churners and non churners – reinforcing the loyalty for the former while potentially retaining the latter – or focus only on several churners subgroups.

Table 9 provides the 3 most important features for three datasets; *Bank*, *Member* and *TelC*. The features are ranked based on their *importance score* which is computed from the mean impurity decrease of each split during class prediction (Section 6.5). This score is further multiplied by the average standardized mean value of each segment in each class. The top most important features are obtained on these final importance values. We present the features for 4 clusters, $\{C_i\}_{G=1..4}$. As can be seen, the top features are fairly different between the clusters of a given dataset, sharing at most one or 2 variables. Besides, the subgroups of churners/non churners customer also exhibit different top features.

Bank dataset As an example, the *tenure*[5] feature helps to discriminate churners and non churners in clusters C_2 to C_4 for *Bank*, while geographical aspects and credit type information are more important in cluster C_1. The *creditscore* variable also plays a discriminative role in clusters C_1 and C_3. A plausible interpretation would be that a customer with higher credit score would tend to remain with the same bank. Hence, it would be interesting for the company to conduct investigations along this line in order to build efficient proactive marketing campaigns.

Member dataset Another example is given by *Member*, where only the cluster C_3 is not concerned by the *annual_fees*[6] variable. It is rather impacted by the *member_gender* information. This is indicative of a particular customer subgroup. We also notice for this cluster the impact of the *membership_package* on the non churner subgroup. This variable indicates whether fees are customized for member's personal package, suggesting straightforward manner to improve member loyalty.

TelC dataset With *TelC*, we can notice that clusters C_1 to C_3 decomposes into churners subgroups that are concerned by different payment method (C_1, bank transfer; C_2, credit card;

[3] `https://cran.r-project.org/web/packages/LLM/LLM.pdf`

[4] for the largest datasets (*K2009*, *KKBox* and *C2C*), 20 runs were done

[5] *tenure* refers to the number of years that the customer has been a client of the bank

[6] *annual_fees* are paid in return for using the exclusive facilities offered by this club

Table 7: AUC results for our ensemble proposal vs. LLM [28] and RF-based [29]. The best results are in bold and the second ones results are underlined.

	LR|XGBoost|RF & DAE-based Segmentation			**LLM**	**RF-based**
	G=2	*G*=4	*G*=6	*(De Caigny et al. [28])*	*(Ullah et al. [29])*
Bank	**0.8620** ± 0.0088	0.8612 ± 0.0096	0.8605 ± 0.0086	0.8501 ± 0.0089	0.8422 ± 0.0119
C2C	**0.6719** ± 0.0062	0.6671 ± 0.0048	0.6708 ± 0.0051	*overtime*	0.6558 ± 0.0046
DSN	**0.8923** ± 0.0157	0.8867 ± 0.0185	0.8852 ± 0.0193	0.8589 ± 0.0278	0.8885 ± 0.0171
HR	0.8402 ± 0.0267	**0.8449** ± 0.0294	0.8335 ± 0.0367	*overtime*	0.7491 ± 0.0355
K2009	0.5063 ± 0.0083	0.5059 ± 0.0113	**0.5091** ± 0.0105	*overtime*	**0.5091** ± 0.0112
KKBox	0.8764 ± 0.0009	**0.8765** ± 0.0012	0.8756 ± 0.0014	*overtime*	0.8749 ± 0.0013
Member	**0.7048** ± 0.0094	0.7028 ± 0.0122	0.6985 ± 0.0106	0.6708 ± 0.0118	0.6858 ± 0.0121
Mobile	0.9071 ± 0.0026	0.9074 ± 0.0028	0.9069 ± 0.0036	*overtime*	0.8985 ± 0.0039
SATO	**0.8175** ± 0.0213	0.8142 ± 0.0199	0.8133 ± 0.0189	0.7835 ± 0.0188	0.8171 ± 0.0182
TelC	0.8465 ± 0.0097	0.8480 ± 0.0098	**0.8490** ± 0.0096	0.8399 ± 0.0114	0.8212 ± 0.0107
TelE	0.9360 ± 0.0023	0.9341 ± 0.0022	0.9319 ± 0.0024	*overtime*	**0.9409** ± 0.0016
UCI	0.9120 ± 0.0215	**0.9152** ± 0.0209	0.9084 ± 0.0197	0.8732 ± 0.0323	0.9095 ± 0.0213
News	**0.8639** ± 0.0076	0.8620 ± 0.0076	0.8541 ± 0.0071	*overtime*	0.8554 ± 0.0084
$\widetilde{AUC}$	**0.8620**	0.8612	0.8541	0.8450	0.8219
$\overline{AUC}$	**0.8182**	0.8174	0.8151	0.8127	0.7978

Table 8: Recall results for our ensemble proposal vs. LLM [28] and RF-based [29].

	LR|XGBoost|RF & DAE-based Segmentation			**LLM**	**RF-based**
	G=2	*G*=4	*G*=6	*(De Caigny et al. [28])*	*(Ullah et al. [29])*
Bank	**0.7551** ± 0.0329	0.7548 ± 0.0346	0.7490 ± 0.0410	0.7398 ± 0.0376	0.7162 ± 0.0396
C2C	0.6663 ± 0.0490	0.6578 ± 0.0658	**0.6750** ± 0.0466	overtime	0.6006 ± 0.0400
DSN	0.8356 ± 0.0459	0.8065 ± 0.0489	0.8027 ± 0.0408	0.8118 ± 0.0608	**0.8376** ± 0.0507
HR	**0.7488** ± 0.0591	0.7297 ± 0.0590	0.7311 ± 0.0789	*overtime*	0.6629 ± 0.0887
K2009	**0.5376** ± 0.2220	0.4416 ± 0.2668	0.4669 ± 0.2608	*overtime*	0.4403 ± 0.1780
KKBox	0.7466 ± 0.0147	0.7507 ± 0.0195	**0.7569** ± 0.0162	*overtime*	0.7440 ± 0.0144
Member	**0.7482** ± 0.0635	0.7426 ± 0.0751	0.7276 ± 0.0867	0.7140 ± 0.0921	0.6996 ± 0.0776
Mobile	0.8365 ± 0.0152	**0.8415** ± 0.0116	0.8388 ± 0.0111	*overtime*	0.8219 ± 0.0140
SATO	0.7371 ± 0.0797	0.7283 ± 0.0797	0.7106 ± 0.0712	0.7079 ± 0.0733	**0.7631** ± 0.0417
TelC	0.7985 ± 0.0528	**0.8026** ± 0.0443	0.7940 ± 0.0452	0.8060 ± 0.0450	0.7671 ± 0.0416
TelE	0.9313 ± 0.0092	0.9320 ± 0.0076	0.9322 ± 0.0091	*overtime*	**0.9361** ± 0.0067
UCI	0.8155 ± 0.0289	0.8227 ± 0.0335	0.8208 ± 0.0424	0.7727 ± 0.0470	**0.8257** ± 0.0385
News	0.7729 ± 0.0408	**0.7772** ± 0.0362	0.7675 ± 0.0389	*overtime*	0.6715 ± 0.0681
$\widetilde{AUC}$	0.7551	0.7548	**0.7569**	0.7563	0.7440
$\overline{AUC}$	**0.7638**	0.7529	0.7518	0.7587	0.7297

Table 9: Top 3 features for our ensemble proposal with DAE-based segmentation (G = 4).

	Bank		**Member**		**TelC**	
	churner	*non churner*	*churner*	*non churner*	*churner*	*non churner*
C_1	creditscore	geography_spain	annual_fees	membership_term_years	monthlycharges	totalcharges
	numproducts_2	numproducts_2	additional_member_3	member_annual_income	totalcharges	techsupport_no
	geography_germany	hascrcard	member_occupation_cd_2	annual_fees	paymentmethod_bank transfer	onlinebackup_no
C_2	estimatedsalary	age	annual_fees	member_age_at_issue	paymentmethod_credit card	gender
	gender	gender	member_age_at_issue	payment_mode_semi-annual	partner	monthlycharges
	tenure	numproducts_2	payment_mode_annual	member_occupation_cd_2	gender	paymentmethod_elec. check
C_3	age	creditscore	member_annual_income	membership_package	monthlycharges	monthlycharges
	balance	balance	membership_term_years	member_occupation_cd_1	tenure_group_tenure_24-48	totalcharges
	hascrcard	tenure	member_gender	payment_mode_annual	paymentmethod_elec. check	techsupport_yes
C_4	estimatedsalary	estimatedsalary	member_age_at_issue	member_age_at_issue	totalcharges	seniorcitizen
	age	balance	membership_term_years	additional_member_0	seniorcitizen	dependents
	tenure	gender	annual_fees	member_occupation_cd_2	paperlessbilling	streamingmovies_yes

C_3, electronic check). The C_4 *TelC* cluster stands out from the rest of the clusters in terms of most important features, both for churners and non churners. Interestingly, the C_3 cluster seems to indicate a non churner subgroup that is satisfied with the technical support.

All in all, these qualitative evaluations put forward the intrinsic multidimensionality and multiplicity of the customer behavior patterns.

7. Conclusion and perspectives

In this study, we propose to review, evaluate, and compare several widespread machine learning approaches in the context of churn prediction and profiling. We also provide insightful visualizations and general recommendations for the choice of a processing pipeline for churn prediction and profiling based on an *ensemble* approach.

Note that we only consider the default parameters for each approach while the supervised context would also allow for boosting versions of some of these techniques [82, 37]. This could significantly improve classification results, in particular for `SVM` [83]. Furthermore, churn prediction issue pertains to the broader class of imbalance data problem. It is therefore related to the extreme case of anomaly or *outlier* detection [84] for which many approaches have been proposed [85, 86, 87, 88]. In particular, semi-supervised approaches regularly provide state-of-the-art results [86, 89]. Among the well-known semi-supervised techniques for anomaly detection, one could cite Local Outlier Factor (`LOF`) [90], One-Class SVM (`ocSVM`) [91], Isolation Forest (`iForest`) [92] and Support Vector Data Description (`SVDD`) [93] methods. These type of techniques should be the object of our future works.

Another type of approaches for which a particular interest should be taken in the context of attrition are the deep learning methods. We can observe that the finance industry is gradually adapting various machine learning techniques. In particular, detecting economic crimes (eg., accounting fraud, money laundering) triggered successful applications of machine learning to this area, where `LR`, `Gnb` and `SVM` are among the most classic methods exploited. However, the occurence of new kinds of fraud, with the growth of electronic market, has popularized deep learning methods which enable the emergence of numerous and innovative deep anomaly detection methods [94]. In particular, `GEV-NN` (Generalized Extreme Value Neural Network) which proposes to use Gumbel distribution as an activation function, reaches state-of-the-art results in the context of imbalanced data [95].

It is also important to notice that most of the churn-like prediction frameworks typically consider only *structured data*. However, as a large proportion of big data consists of diverse *unstructured data* [96], it is important to find strategies that enable the incorporation of the information that they contain. Indeed, online communication means between customers and companies or banks are expanding rapidly. Previous studies demonstrate that textual data can improve the churn prediction performance. Examples can be found with the use of highly unstructured data coming from social networks [97, 98, 55]. Recently, De Caigny *et al.* [99] proposed the incorporation of textual information based on Convolutional Neural Network. Harnessing information from social network features – comments, friend sharing – [100] can also improve churn predictions by enabling causal information discovery [101]. Indeed, social influence is one of the key reasons for churn behavior [102]. These last considerations should be part of an interesting short-term study.

Appendix A: Datasets complementary information

K2009 (*KDD-Cup 2009 small*) This dataset was proposed in the context of the *KDD Cup 2009: Churn relationship prediction* and originates from the French telecommunication company *Orange* in order to predict the switch of provider [103]. #Dummified Features: 1039.

KKBox's (*WSDM CUP 2018*) This churn dataset was proposed for the 11^{th} ACM International Conference on Web Search and Data Mining (WSDM 2018) and originates from the KKbox Taiwanese music streaming company. The proposed challenge is to predict if a subscriber will churn as soon as the subscription expires [10]. #Dummified Features: 56.

UCI (*MLC Churn*) This dataset is similar to the *Telecom SingTel*, *CrowdAnalytix* and *UCI* datasets. *MLC Churn* is proposed in the **R** package *modeldata* [83]. #Dummified Features: 21.

HR (*IBM Employee Attrition*) This dataset originates from IBM HR and includes $1,470$ records of individuals who left the company or not. It is an artificial dataset created by IBM data scientists from Watson analytics, and has been proposed to uncover the factors that lead to employee attrition [104]. #Dummified Features: 86.

TelE (*Telco-Europa*) This dataset corresponds to the real data of a small telecommunications company in Oceania that has only 14 months of historical data. It is found in online churn prediction tutorials. #Dummified Features: 26.

News (*Newspaper*) This datasets contains information on Californian newspaper subscribers and an attrition variable. It is found in online churn prediction tutorials. Other newspaper private datasets were analyzed in previous studies; see [105, 55, 106]. #Dummified Features: 307.

Bank This data set contains details of a bank's customers and their departure. It is found in online churn prediction tutorials. #Dummified Features: 16.

TelC (*IBM Telco Churn*) This dataset is proposed by IBM and is used in an online tutorial to train a model that predicts if a customer is likely to leave the telecom provider. #Dummified

Features: 34.

C2C (*Cell2Cell*) The data sets is provided by the Teradata Center for CRM (Customer Relationship Management). Data were provided by the Cell2Cell company, which is one of the largest wireless company in the USA [107]. #Dummified Features: 75.

Member (*Membership Woes*) This dataset is cited in online tutorials. #Dummified Features: 26.

SATO (*South-asian*) This dataset is provided by a South Asian Telecom Operator, also called SATO. Data were collected between August 2015 and September 2015 [108]. #Dummified Features: 29.

DSN (*DSN-telecom 'Nigerian Telecom'*) This dataset has been proposed in the context of the *DSN Telecoms Churn Prediction 2018* challenge, which is one of the pre-qualification to the *2018 Data Science Nigeria hackathon*. #Dummified Features: 32.

Appendix B: Python package and functions

All experiments in this survey were performed on public datasets using freely available Python packages. Hence, results are entirely reproducible. Table 10 summarizes information on packages, functions and parameters used for our experiments. It also provides links to the online description of each function.

References

[1] F. F. Reichheld, W. E. Sasser, Zero defections: Quality comes to services, Harvard business review 68 (5) (1990) 105–111.

[2] R. N. Bolton, T. M. Bronkhorst, The relationship between customer complaints to the firm and subsequent exit behavior, ACR North American Advances 22 (1995) 94–100.

[3] W. J. Reinartz, V. Kumar, The impact of customer relationship characteristics on profitable lifetime duration, Journal of marketing 67 (1) (2003) 77–99.

[4] R. Siber, Combating the churn phenomenon-as the problem of customer defection increases, carriers are having to find new strategies for keeping subscribers happy., Telecommunications-International Edition 31 (10) (1997) 77–81.

[5] Z. Yang, R. T. Peterson, Customer perceived value, satisfaction, and loyalty: The role of switching costs, Psychology & Marketing 21 (10) (2004) 799–822.

[6] C.-C. Günther, I. F. Tvete, K. Aas, G. I. Sandnes, Ø. Borgan, Modelling and predicting customer churn from an insurance company, Scandinavian Actuarial Journal 2014 (1) (2014) 58–71.

[7] D. A. Kumar, V. Ravi, et al., Predicting credit card customer churn in banks using data mining, International Journal of Data Analysis Techniques and Strategies 1 (1) (2008) 4–28.

[8] K. Coussement, K. W. De Bock, Customer churn prediction in the online gambling industry: The beneficial effect of ensemble learning, Journal of Business Research 66 (9) (2013) 1629–1636.

[9] J. Kawale, A. Pal, J. Srivastava, Churn prediction in MMORPGs: A social influence based approach, in: 2009 International Conference on Computational Science and Engineering, Vol. 4, IEEE, 2009, pp. 423–428.

[10] Y. Chen, X. Xie, S.-D. Lin, A. Chiu, Wsdm cup 2018: Music recommendation and churn prediction, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, ACM, 2018, pp. 8–9.

[11] F. Tan, Z. Wei, J. He, X. Wu, B. Peng, H. Liu, Z. Yan, A Blended Deep Learning Approach for Predicting User Intended Actions, Proceedings - IEEE International Conference on Data Mining, ICDM 2018-Novem (2018) 487–496. doi:10.1109/ICDM.2018.00064.

[12] V. Effendy, Z. A. Baizal, et al., Handling imbalanced data in customer churn prediction using combined sampling and weighted random forest, in: 2014 2nd International Conference on Information and Communication Technology (ICoICT), IEEE, 2014, pp. 325–330.

[13] M. F. Abdillah, J. Nasri, A. Aditsania, Using deep learning to predict customer churn in a mobile telecomunication network, eProceedings of Engineering 3 (2) (2016).

[14] A. Hudaib, R. Dannoun, O. Harfoushi, R. Obiedat, H. Faris, Hybrid data mining models for predicting customer churn, International Journal of Communications, Network and System Sciences 8 (05) (2015) 91.

[15] P. Hosein, G. Sewdhan, A. Jailal, Soft-churn: Optimal switching between prepaid data subscriptions on e-sim support smartphones, in: 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), IEEE, 2021, pp. 1–6.

[16] D. A. Garvin, Managing quality: The strategic and competitive edge, Simon and Schuster, 1988.

[17] P. Gilmour, G. Borg, P. A. Duffy, N. D. Johnston, B. Limbek, M. R. Shaw, Customer service: differentiating by market segment, International Journal of Physical Distribution & Logistics Management 24 (4) (1994) 18–23.

[18] G. LeBlanc, N. Nguyen, Customers' perceptions of service quality in financial institutions, International Journal of Bank Marketing (1988).

[19] M. Laroche, J. A. Rosenblatt, T. Manning, Services used and factors considered important in selecting a bank: an investigation across diverse demographic segments, International Journal of bank marketing (1986).

[20] J. J. Cronin Jr, S. A. Taylor, Measuring service quality: a reexamination and extension, Journal of marketing 56 (3) (1992) 55–68.

[21] S. M. Keaveney, Customer switching behavior in service industries: An exploratory study, Journal of marketing 59 (2) (1995) 71–82.

[22] A. D. Athanassopoulos, Customer satisfaction cues to support market segmentation and explain switching behavior, Journal of business research 47 (3) (2000) 191–207.

[23] C.-Y. Tsai, C.-C. Chiu, A purchase-based market segmentation methodology, Expert Systems with Applications 27 (2) (2004) 265–276.

[24] A. Vellido, P. Lisboa, K. Meehan, Segmentation of the on-line shopping market using neural networks, Expert systems with applications 17 (4) (1999) 303–314.

[25] R. Kuo, Y. An, H. Wang, W. Chung, Integration of self-organizing feature maps neural network and genetic k-means algorithm for market segmentation, Expert systems with applications 30 (2) (2006) 313–324.

[26] C. C. H. Chan, Intelligent value-based customer segmentation method for campaign management: A case study of automobile retailer, Expert systems with applications 34 (4) (2008) 2754–2762.

[27] V. García, J. S. Sánchez, R. A. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowledge-Based Systems 25 (1) (2012) 13–21.

[28] A. De Caigny, K. Coussement, K. W. De Bock, A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees, European Journal of Operational Research 269 (2) (2018) 760–772.

[29] I. Ullah, B. Raza, A. K. Malik, M. Imran, S. U. Islam, S. W. Kim, A churn prediction model using random forest: analysis of machine learning techniques for churn prediction and factor identification in telecom sector, IEEE access 7 (2019) 60134–60149.

[30] M. Bécue-Bertaut, J. Pagès, Multiple factor analysis and clustering of a mixture of quantitative, categorical and frequency data, Computational Statistics & Data Analysis 52 (6) (2008) 3255–3268.

[31] Y. Xie, X. Li, Churn prediction with linear discriminant boosting algorithm, in: 2008 International Conference on Machine Learning and Cybernetics, Vol. 1, IEEE, 2008, pp. 228–233.

[32] J. Hadden, A. Tiwari, R. Roy, D. Ruta, Churn prediction: Does technology matter, International Journal of Intelligent Technology 1 (2) (2006) 104–110.

[33] M. C. Mozer, R. Wolniewicz, D. B. Grimes, E. Johnson, H. Kaushansky, Predicting subscriber dissatisfaction and improving retention in the

Table 10: Packages, functions and parameters summary for the churn pipeline

	Approach	*Function*	*parameters*	*version*	*online details*
		Sampling			
over.	SMOTE	`SMOTE`	*default*	0.7.0	imblearn.over_sampling.SMOTE
	ADASYN	`ADASYN`	*'not minority'*	0.7.0	imblearn.over_sampling.ADASYN
under.	Tomek links	`TomekLinks`	*default*	0.7.0	imblearn.under_sampling.TomekLinks.html
	NCR	`NeighbourhoodCleaningRule`	*default*	0.7.0	imblearn.under_sampling. NeighbourhoodCleaningRule
hybrid	SMOTE+Random	`SMOTE` `RandomUnderSampler`	*default*	0.7.0	imblearn.over_sampling.SMOTE imblearn.under_sampling. RandomUnderSampler
	SMOTE+Tomek links	`SMOTETomek`	*default*	0.7.0	imblearn.combine.SMOTETomek
	SMOTE+NCR	`SMOTE` `NeighbourhoodCleaningRule`	SMOTE: *default* NCR: *'minority'*	0.7.0	imblearn.over_sampling.SMOTE imblearn.under_sampling. NeighbourhoodCleaningRule
		Model Fitting			
Supervised	*k*-nearest neighbors	`KNeighborsClassifiere`	*default*	0.23.2	neighbors.KNeighborsClassifier
	Naïves Bayes	`GaussianNB`	*default*	0.23.2	sklearn.naive_bayes.GaussianNB
	Logistic Regression	`LogisticRegression`	*default*	0.23.2	sklearn.linear_model.LogisticRegression
	Support Vector Machine	`SVC`	*default*	0.23.2	svm.SVC
	Decision Tree	`DecisionTreeClassifier`	*default*	0.23.2	sklearn.tree.DecisionTreeClassifier
Ensemble Supervised	Random Forest	`RandomForestClassifier`	*default*	0.23.2	sklearn.ensemble.RandomForestClassifier
	XGBoost	`XGBClassifier`	*default*	1.0.2	xgboost.readthedocs.io
		Evaluation			
Strategy	Cross Validation	`train_test_split`	*default*	0.23.2	sklearn.model_selection.train_test_split
	K-fold validation	`KFold`	*K=5*	0.23.2	sklearn.model_selection.KFold
	Stratified k-fold validation	`StratifiedKFold`	*K=5*	0.23.2	sklearn.model_selection.StratifiedKFold
Metric	Top-lift	`plot_lift_curve`	*default*	0.3.7	rasbt.github.io – lift_score
	F1-score	`f1_score`	*default*	0.23.2	sklearn.metrics.f1_score
	AUC	`roc_auc_score`	*default*	0.23.2	sklearn.metrics.roc_auc_score

wireless telecommunications industry, IEEE Transactions on neural networks 11 (3) (2000) 690–696.

[34] B. Zadrozny, C. Elkan, Learning and making decisions when costs and probabilities are both unknown, in: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, 2001, pp. 204–213.

[35] P. Domingos, Metacost: A general method for making classifiers cost-sensitive, in: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, 1999, pp. 155–164.

[36] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: Third IEEE international conference on data mining, IEEE, 2003, pp. 435–442.

[37] A. Lemmens, C. Croux, Bagging and boosting classification trees to predict churn, Journal of Marketing Research 43 (2) (2006) 276–286.

[38] B. Gregory, Predicting customer churn: Extreme gradient boosting with temporal data, arXiv preprint arXiv:1802.03396 (2018).

[39] W. Li, M. Gao, H. Li, Q. Xiong, J. Wen, Z. Wu, Dropout prediction in MOOCs using behavior features and multi-view semi-supervised learning, Proceedings of the International Joint Conference on Neural Networks 2016-Octob (2016) 3130–3137. doi:10.1109/IJCNN.2016.7727598.

[40] A. Śniegula, A. Poniszewska-Marańda, M. Popović, Study of machine learning methods for customer churn prediction in telecommunication company, in: Proceedings of the 21st International Conference on Information Integration and Web-based Applications & Services, 2019, pp. 640–644.

[41] V. V. Saradhi, G. K. Palshikar, Employee churn prediction, Expert Systems with Applications 38 (3) (2011) 1999–2006.

[42] A. Keramati, R. Jafari-Marandi, M. Aliannejadi, I. Ahmadian, M. Mozaffari, U. Abbasi, Improved churn prediction in telecommunication industry using data mining techniques, Applied Soft Computing 24 (2014) 994–1012.

[43] C. Orsenigo, C. Vercellis, Combining discrete svm and fixed cardinality warping distances for multivariate time series classification, Pattern Recognition 43 (11) (2010) 3787–3794.

[44] G. He, Y. Duan, G. Zhou, L. Wang, Early classification on multivariate time series with core features, in: International Conference on Database and Expert Systems Applications, Springer, 2014, pp. 410–422.

[45] L. Wang, Z. Wang, S. Liu, An effective multivariate time series classification approach using echo state network and adaptive differential evolution algorithm, Expert Systems with Applications 43 (2016) 237–249.

[46] M. Óskarsdóttir, T. Van Calster, B. Baesens, W. Lemahieu, J. Vanthienen, Time series for early churn detection: Using similarity based classification for dynamic networks, Expert Systems with Applications 106 (2018) 55–65.

[47] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, science 313 (5786) (2006) 504–507.

[48] Y. Bengio, L. Yao, G. Alain, P. Vincent, Generalized denoising auto-encoders as generative models, Advances in neural information processing systems 26 (2013).

[49] C. Song, F. Liu, Y. Huang, L. Wang, T. Tan, Auto-encoder based data clustering, in: Iberoamerican congress on pattern recognition, Springer, 2013, pp. 117–124.

[50] M. Alkhayrat, M. Aljnidi, K. Aljoumaa, A comparative dimensionality reduction study in telecom customer segmentation using deep learning and PCA, Journal of Big Data 7 (1) (2020) 1–23.

[51] F. Tian, B. Gao, Q. Cui, E. Chen, T.-Y. Liu, Learning deep representa-

tions for graph clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 28, 2014.
[52] X. Guo, L. Gao, X. Liu, J. Yin, Improved deep embedded clustering with local structure preservation., in: IJCAI, 2017, pp. 1753–1759.
[53] J. Xie, R. Girshick, A. Farhadi, Unsupervised deep embedding for clustering analysis, in: International conference on machine learning, PMLR, 2016, pp. 478–487.
[54] S. Affeldt, L. Labiod, M. Nadif, Spectral clustering via ensemble deep autoencoder learning (SC-EDAE), Pattern Recognition 108 (2020) 107522.
[55] K. Coussement, D. Van den Poel, Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques, Expert systems with applications 34 (1) (2008) 313–327.
[56] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics, Information sciences 250 (2013) 113–141.
[57] J. Błaszczyński, J. Stefanowski, Local data characteristics in learning classifiers from imbalanced data, in: Advances in Data Analysis with Computational Intelligence Methods, Springer, 2018, pp. 51–85.
[58] J. Stefanowski, Dealing with data difficulty factors while learning from imbalanced data, in: Challenges in computational statistics and data mining, Springer, 2016, pp. 333–363.
[59] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, Smote: synthetic minority over-sampling technique, Journal of artificial intelligence research 16 (2002) 321–357.
[60] C. Drummond, R. C. Holte, et al., C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling, in: Workshop on learning from imbalanced datasets II, Vol. 11, Citeseer, 2003, pp. 1–8.
[61] S. He, H., Bai, Y., Garcia, E., & Li, ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In IEEE International Joint Conference on Neural Networks, 2008, IJCNN 2008.(IEEE World Congress on Computational Intelligence) (pp. 1322– 1328) (3) (2008) 1322– 1328.
[62] J. Laurikkala, Improving identification of difficult small classes by balancing class distribution, in: Conference on Artificial Intelligence in Medicine in Europe, Springer, 2001, pp. 63–66.
[63] I. Tomek, Tomek Link: Two Modifications of CNN, IEEE Trans. Systems, Man and Cybernetics SMC-6 (1976) 769–772.
URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp={&}arnumber=4309452
[64] P. Hart, The condensed nearest neighbor rule (corresp.), IEEE transactions on information theory 14 (3) (1968) 515–516.
[65] D. L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man, and Cybernetics (3) (1972) 408–421.
[66] G. E. Batista, A. L. Bazzan, M. C. Monard, et al., Balancing training data for automated annotation of keywords: a case study., in: WOB, 2003, pp. 10–18.
[67] G. H. John, P. Langley, Estimating continuous distributions in bayesian classifiers, arXiv preprint arXiv:1302.4964 (2013).
[68] D. J. Hand, K. Yu, Idiot's bayes—not so stupid after all?, International statistical review 69 (3) (2001) 385–398.
[69] V. Vapnik, Statistical learning theory wiley-interscience, New York (1998).
[70] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification and regression trees, Routledge, 2017.
[71] T. Hastie, R. Tibshirani, J. Friedman, The elements of statistical learnin, Cited on (2009) 33.
[72] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.
[73] C. Chen, A. Liaw, L. Breiman, et al., Using random forest to learn imbalanced data, University of California, Berkeley 110 (1-12) (2004) 24.
[74] Z. Zhao, H. Peng, C. Lan, Y. Zheng, L. Fang, J. Li, Imbalance learning for the prediction of n 6-methylation sites in mrnas, BMC genomics 19 (1) (2018) 1–10.
[75] L. Breiman, Bagging predictors, Machine learning 24 (2) (1996) 123–140.
[76] J. Demšar, Statistical comparisons of classifiers over multiple data sets, The Journal of Machine Learning Research 7 (2006) 1–30.
[77] J.-P. Benzécri, et al., L'analyse des données, Vol. 2, Dunod Paris, 1973.
[78] M. Greenacre, Correspondence analysis in practice, chapman and hall/crc, 2017.
[79] M. Mohandes, M. Deriche, S. O. Aliyu, Classifiers combination techniques: A comprehensive review, IEEE Access 6 (2018) 19626–19639.
[80] C. Guo, F. Berkhahn, Entity embeddings of categorical variables, arXiv preprint arXiv:1604.06737 (2016).
[81] S. Loffe, C. Normalization, Accelerating deep network training by reducing internal covariate shift, arXiv (2014).
[82] M. Clemente, V. Giner-Bosch, S. San Matías, Assessing classification methods for churn prediction by composite indicators, Manuscript, Dept. of Applied Statistics, OR & Quality, UniversitatPolitècnica de València, Camino de Vera s/n 46022 (2010).
[83] T. Vafeiadis, K. I. Diamantaras, G. Sarigiannidis, K. C. Chatzisavvas, A comparison of machine learning techniques for customer churn prediction, Simulation Modelling Practice and Theory 55 (2015) 1–9.
[84] J. Kong, W. Kowalczyk, S. Menzel, T. Bäck, Improving imbalanced classification by anomaly detection, in: T. Bäck, M. Preuss, A. Deutz, H. Wang, C. Doerr, M. Emmerich, H. Trautmann (Eds.), Parallel Problem Solving from Nature – PPSN XVI, Springer International Publishing, Cham, 2020, pp. 512–523.
[85] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Comput. Surv. 41 (3) (jul 2009). doi:10.1145/1541880.1541882.
[86] S. Alam, S. K. Sonbhadra, S. Agarwal, P. N. Nagabhushan, One-class support vector classifiers: A survey, Knowl. Based Syst. 196 (2020) 105754.
[87] G. Pang, H. Xu, L. Cao, W. Zhao, Selective value coupling learning for detecting outliers in high-dimensional categorical data, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 807–816.
[88] A. Taha, A. S. Hadi, Anomaly detection methods for categorical data: A review, ACM Comput. Surv. 52 (2) (may 2019). doi:10.1145/3312739.
[89] M. E. Villa-Pérez, M. Á. Álvarez-Carmona, O. Loyola-González, M. A. Medina-Pérez, J. C. Velazco-Rossell, K.-K. R. Choo, Semi-supervised anomaly detection algorithms: A comparative summary and future research directions, Knowledge-Based Systems (2021) 106878doi:https://doi.org/10.1016/j.knosys.2021.106878.
[90] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, Lof: Identifying density-based local outliers, SIGMOD Rec. 29 (2) (2000) 93–104. doi:10.1145/335191.335388.
[91] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, J. Platt, Support vector method for novelty detection, NIPS'99, MIT Press, Cambridge, MA, USA, 1999, p. 582–588.
[92] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation-based anomaly detection, ACM Trans. Knowl. Discov. Data 6 (1) (mar 2012). doi:10.1145/2133360.2133363.
[93] D. M. J. Tax, R. P. W. Duin, Support vector domain description, Pattern Recogn. Lett. 20 (11–13) (1999) 1191–1199. doi:10.1016/S0167-8655(99)00087-2.
[94] G. Pang, C. Shen, L. Cao, A. V. D. Hengel, Deep learning for anomaly detection: A review, ACM Comput. Surv. 54 (2) (mar 2021). doi:10.1145/3439950.
[95] L. Munkhdalai, T. Munkhdalai, K. H. Ryu, Gev-nn: A deep neural network architecture for class imbalance problem in binary classification, Knowledge-Based Systems 194 (2020) 105534.
[96] A. Gandomi, M. Haider, Beyond the hype: Big data concepts, methods, and analytics, International journal of information management 35 (2) (2015) 137–144.
[97] L. Tang, L. Thomas, M. Fletcher, J. Pan, A. Marshall, Assessing the impact of derived behavior information on customer attrition in the financial service industry, European Journal of Operational Research 236 (2) (2014) 624–633.
[98] D. F. Benoit, D. Van den Poel, Improving customer retention in financial services using kinship network information, Expert Systems with Applications 39 (13) (2012) 11435–11442.
[99] A. De Caigny, K. Coussement, K. W. De Bock, S. Lessmann, Incorporating textual information in customer churn prediction models based on a convolutional neural network, International Journal of Forecast-

ing 36 (4) (2020) 1563–1578. doi:https://doi.org/10.1016/j.ijforecast.2019.03.029.

[100] A. Salah, M. Nadif, Social regularized von mises–fisher mixture model for item recommendation, Data Mining and Knowledge Discovery 31 (5) (2017) 1218–1241.

[101] G. Zhang, J. Zeng, Z. Zhao, D. Jin, Y. Li, A counterfactual modeling framework for churn prediction, in: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, pp. 1424–1432.

[102] I. Nitzan, B. Libai, Social effects on customer retention, Journal of Marketing 75 (6) (2011) 24–38.

[103] I. Guyon, V. Lemaire, M. Boullé, G. Dror, D. Vogel, Analysis of the kdd cup 2009: Fast scoring on a large orange customer database, in: Proceedings of the 2009 International Conference on KDD-Cup 2009-Volume 7, JMLR. org, 2009, pp. 1–22.

[104] I. McKinley Stacker, Ibm waston analytics. sample data: Hr employee attrition and performance [data file] (2015).

[105] J. Burez, D. Van den Poel, Handling class imbalance in customer churn prediction, Expert Systems with Applications 36 (3) (2009) 4626–4636.

[106] K. Coussement, D. F. Benoit, D. Van den Poel, Improved marketing decision making in a customer churn prediction context using generalized additive models, Expert Systems with Applications 37 (3) (2010) 2132–2143.

[107] Y. Kim, Toward a successful crm: variable selection, sampling, and ensemble, Decision Support Systems 41 (2) (2006) 542–553.

[108] M. Ahmed, H. Afzal, I. Siddiqi, M. F. Amjad, K. Khurshid, Exploring nested ensemble learners using overproduction and choose approach for churn prediction in telecom industry, Neural Computing and Applications 8 (2018). doi:10.1007/s00521-018-3678-8.