A survey on machine learning methods for churn prediction

Louis Geiler, Séverine Affeldt, Mohamed Nadif

A survey on machine learning methods for churn prediction

Louis Geiler[1,2], Séverine Affeldt[1*] and Mohamed Nadif[1]

[1*]Université de Paris, Centre Borelli UMR 9010, 75006 Paris, France.
[2]Brigad, 34 Rue du Sentier, Paris, 75002, France.

*Corresponding author(s). E-mail(s): severine.affeldt@u-paris.fr;
Contributing authors: louis.geiler@u-paris.fr; mohamed.nadif@u-paris.fr;

Abstract

The diversity and specificities of today's businesses have leveraged a wide range of prediction techniques. In particular, churn prediction is a major economic concern for many companies. The purpose of this study is to draw general guidelines from a benchmark of supervised machine learning techniques in association with widely used data sampling approaches on publicly available datasets in the context of churn prediction. Choosing a priori the most appropriate sampling method as well as the most suitable classification model is not trivial, as it strongly depends on the data intrinsic characteristics. In this paper we study the behavior of eleven supervised and semi-supervised learning methods and seven sampling approaches on sixteen diverse and publicly available *churn-like* datasets. Our evaluations, reported in terms of the Area Under the Curve (AUC) metric, explore the influence of sampling approaches and data characteristics on the performance of the studied learning methods. Besides, we propose Nemenyi test and Correspondence Analysis as means of comparison and visualization of the association between classification algorithms, sampling methods and datasets. Most importantly, our experiments lead to a practical recommendation for a prediction pipeline based on an ensemble approach. Our proposal can be successfully applied to a wide range of *churn-like* datasets.

Keywords: churn prediction, machine learning, ensemble technique

1 Introduction

Building a strong Customer Relationship Management (CRM) has become a crucial topic for many companies in recent years. In particular, management and marketing services are focusing their attention on the customer retention, as it clearly appeared that the acquisition costs of a new customer can be much more higher than the retention costs of an existing one [109, 119, 148]. Besides, retained customers can be of great help for the company by spreading positive word of mouth [108], which would subsequently lower the marketing costs of new customers acquisition [17]. The ever-rising competition in industry has therefore pushed forward companies to carefully control the switch of customers or subscribers to another company, also known as customer *churn*, customer *attrition* or customer *defection*. The customer churn can be particularly damaging for subscription-based service firms, such as insurance [59], banking [82], online gambling [38], online video games [77], music streaming [34], online services [123] or telecommunication [1, 50, 72, 74]. As such companies are expecting fixed and regular membership fees, customer switching behavior should be tempered to ensure sustainable profit. Therefore, accurately predicting the

customers who are prone to churn has become a priority in industry.

In addition to the systematic prediction of customers with switching intentions, firms also seek to determine the causes of churn behavior. Knowing the reasons for customers defection would both provide support for the profiling of defection-prone customers and help fostering efficient pro-active campaigns for customers retention [85]. The customer data generally contains service usage (e.g. frequency, duration), billing information (e.g. regularity of payments, contract term) and support service usage and satisfaction. Among the most probable antecedents of customer churn, several prior studies have reported the satisfaction and the service quality [7, 152]. Finding the most significant churn behavior causes (or features) also bring a valuable technical advantage for the prediction model formulation. Indeed, the number of features in churn datasets is usually large and dimentionality reduction helps reducing overfitting and improving the generalization of the prediction models.

Marketing and financial industry services preferentially focused on statistic modeling methods to tackle the churn analysis and prediction task. A well-known approach is the *survival analysis* that proposes to model the occurrence and timing of events [14, 16, 107]. In the context of customer attrition, the time to failure corresponds to the churn behavior. The potential churner behavior has also been analyzed using *structural equation modeling* [54, 101, 134]. Such approach can be of great interest for managerial decisions, as it evaluates the effect of suspected influential features on a specific customer decision, such as churn. The *analysis of variance* was also widely used in marketing and business areas to uncover customer behavior [91, 94, 152]. Financial and retail services also rely on *T-test* and *Chi square* statistics to forecast customer behavior and perceptions [70, 93, 106].

The proposed survey is not exploring these traditional approaches and rather focuses on machine learning techniques that are being increasingly encountered in the customer churn context. These techniques include supervised and semi-supervised approaches. K-nearest neighbors, Naive Bayes classifiers, Linear Regression, Logistic Regression, Linear Discriminant Analysis [146], Decision Tree learning [63, 95] and Support Vector Machine are among the widely used supervised algorithms in the context of churn prediction. Algorithmic modifications [150] and cost-sensitive learning variants [47, 151] 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 [146], Gradient Boosting [84, 95] or XGBoost [58] to tackle the churn prediction task. Successful semi-supervised methods have been proposed [86], as well as deep learning approaches that offer promising results [58, 63, 95, 123].

The churn prediction problem relates to the broader issue of class imbalance from which the anomaly or outlier detection is an extreme case [81]. Efficient anomaly detection systems provide valuable information in a wide range of diverse domains, such as medical diagnostic systems [27], fraud detection [76] or industrial fault detectors [145]. Many approaches have been proposed to tackle the outlier detection task [5, 30, 103, 122]. In particular, semi-supervised approaches regularly provide state-of-the-art results [5, 135]. Among the well-known semi-supervised techniques for anomaly detection, one could cite Local Outlier Factor (LOF) [23], One-Class SVM (ocSVM) [116], Isolation Forest (iForest) [88] and Support Vector Data Description (SVDD) [126] methods. The deep learning research field enabled also the emergence of a large number of deep anomaly detection methods [105]. 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 [97]. DevNet (Deviation Network) also demonstrates efficiency and competing results for anomaly detection [104].

1.1 Related works

In recent years, churn prediction triggered novel strategies for which machine learning approaches were used and adapted. The strong interest in churn prediction led to various surveys related to machine learning in the fields of telecommunication industry, human resources, bank subscription or financial services. Saradhi *et al.*

reviewed three machine learning techniques in the *employee churn* context [115], a problem similar to customer churn prediction. They provide comparative results on a private dataset using a cross-validation procedure. Similarly, Sniegula *et al.* compare three machine learning techniques on a single churn dataset in the context of telecommunication industry [120]. Keramati *et al.* proposed a literature and comparative experimental study with four models on a private dataset. Other comparative studies based on ensemble machine learning approaches were also proposed [84, 111, 138]. Umayaparvathi *et al.* literature survey [129], which focuses on customer churn prediction in telecommunication, provides a list of regularly encountered models in churn analysis. The authors indicate four publicly available churn datasets and briefly discuss the possible metrics. A more thorough literature review was proposed by Gracìa *et al.* [55]. Several steps of the churn prediction analysis are discussed by the authors, among which the data gathering, the features selection, the model implementation and the possible evaluation procedures and metrics. Their survey concludes with recommendations based on literature. Several deep learning approaches have been investigated for churn prediction. In [118], Seymen *et al.* proposed a novel deep learning model which is compared to logistic regression and artificial neural network models. Their study encloses a detailed literature review of deep learning methods in churn prediction. Beyond this domain, several reviews dedicated to anomaly detection, which can be seen as an extreme case of churn prediction, have been proposed. In [112], the authors highlight connections between classic *shallow* and novel deep approaches applied to anomaly detection. A thorough *deep anomaly detection* review, recently proposed by Pang *et al.* [105], provides a comprehensive taxonomy of deep learning techniques for anomaly detection and discusses the associated challenges and perspectives.

Although interesting, these surveys compare very few machine learning techniques in the churn context and hardly include any experimental study. Furthermore, comparative results usually involve private datasets, making the experiments not reproducible and extrapolation to novel datasets difficult. Beyond discussion on the models themselves, these reviews typically omit the techniques for classes rebalancing, which is an important issue for churn prediction. Finally, churn prediction surveys rarely raised the topic of evaluation procedures that impact the validity and robustness of the evaluations.

1.2 Our contribution

In this survey, our primary goal is to compare multiple alternatives within a machine learning churn analysis pipeline that involves *(i)* a sampling stage, *(ii)* a model fitting phase and *(iii)* a robust evaluation procedure (Fig. 2). An exhaustive analysis of all existing algorithmic variants and cost-sensitive approaches within this pipeline would not be reasonably feasible. Hence, we rather focus on base learning algorithms in combination with widespread sampling approaches to finally propose a pipeline that is successful on a wide range of churn-like datasets. In the churn context, several data issues have been pointed out in relation with classes imbalance [15, 90, 121], among which the existence of small *disjuncts* [71, 140, 141], the overlap between classes [44, 56], the noisy data [117] or the borderline instances [98]. For this study, we do not try to correct for these specific issues and rather focus on the balancing of the classes distribution as it was shown to play a significant role in the performance of standard classifiers [57]. Several deep learning approaches were proposed to tackle the churn prediction problem [28, 46, 130, 147]. We propose to compare traditional machine learning approaches to a simple feed-forward neural network and also to more recent and sophisticated deep learning methods which have been shown to be particularly efficient for imbalanced data or in the context of outliers detection [97, 104]. Reviewing in depth the wide range of features selection techniques would require another survey and is out of the scope of this review. We invite the reader to refer to the literature which is abundant with thorough surveys and comparative studies [61, 132, 136].

Hence, we compare in this paper a range of machine learning techniques in the context of churn prediction and give practical recommendations. We first provide an overview of publicly available churn datasets (Section 2). Then, we introduce the imbalance class distribution issue

Table 1: Publicly available churn and *churn-like (*)* datasets with link

Link to Data	#Instances	#Features	#Dum.Feat.	$\#churn$	$\#non-churn$	$\%churn$	$\frac{\#churn}{\#non-churn}$
*Fraud**	284,807	29	29	492	284,315	0.0017	0.0017
K2009	50,000	230	1,039	3,672	46,327	0.07	0.08
*Thyroid**	7,200	21	21	534	6,666	0.07	0.08
KKbox	970,960	49	56	87,330	883,630	0.09	0.10
UCI	5,000	20	21	707	4,293	0.14	0.16
*Campaign**	41,188	17	63	4,640	36,548	0.12	0.13
HR	1,470	34	86	37	1,233	0.16	0.19
TelE	190,776	19	26	29,884	160,892	0.16	0.19
News	15,855	18	307	3,037	12,818	0.19	0.23
Bank	10,000	12	16	2,037	7,963	0.20	0.25
Mobile	66,469	65	65	13,907	52,562	0.21	0.27
TelC	7,043	20	34	1,869	5,174	0.27	0.37
C2C	71,047	71	75	20,609	50,438	0.29	0.41
Member	10,362	14	26	3,143	7,219	0.30	0.43
SATO	2,000	13	29	1,000	1,000	0.50	1
DSN	1,401	15	32	700	700	0.50	1

and describe seven widespread balancing techniques (Section 3). The description of supervised, ensemble supervised, semi-supervised and deep learning techniques are given in Section 4. We also discuss three evaluation procedures (Section 5) and four metrics (Section 5.2) before providing the exhaustive experimental results of our pipeline variants (Section 6). Our experiments are performed on sixteen publicly available *churn-like* datasets that range from human resources, to telecommunication, internet subscription and music streaming industry. Our results reveal interesting complementary behaviors between machine learning techniques (Section 6.2.1) and ultimately indicate an advisable churn analysis pipeline which can be successfully applied to various churn-like datasets (Section 6.2.3). We summarized our experimental findings with Nemenyi tests and Correspondence Analysis visualizations (Section 6.2.2). The overall conclusion is given in Section 8.

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

2 Background

This section formalizes the churn prediction problem. It also introduces publicly available churn datasets and discusses appropriate evaluation metrics. Besides, this section introduces a machine learning churn prediction pipeline and the associated variants that we review in this survey.

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[1] is the number of classes $C_1, \ldots, C_G$. 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*.

[1] 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.

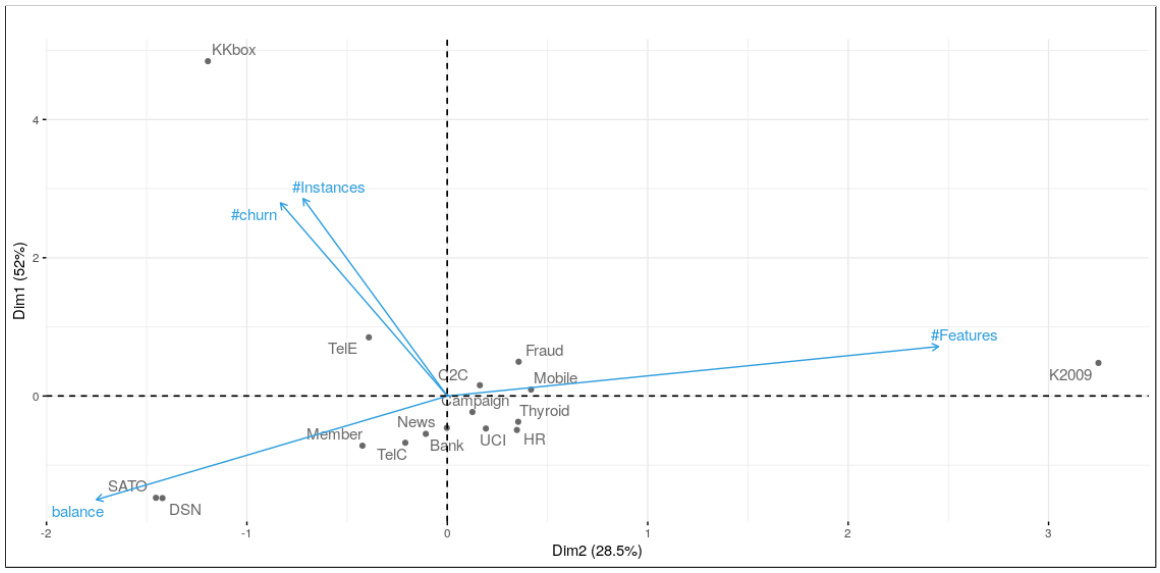


Fig. 1: Datasets distribution on the two first PCA components of Table 1

2.2 Public datasets

Several studies have evaluated machine learning approaches for churn modeling on various datasets. However, these studies typically include private datasets that prevent from reproducibility and extrapolation to novel datasets. In this survey, we perform a comparative evaluation of multiple churn analysis techniques on publicly available datasets only. A churn dataset usually comprises features of different types 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. Our benchmark datasets are also enriched with three datasets that are usually found in anomaly detection contexts, namely *Fraud*, *Thyroid* and *Campaign*.

Table 1 lists the public churn datasets that are considered in this work and provides their online access (see also Appendix A). These datasets have diverse number of instances, number of features and *dummified* features [2], and percentage of churners. The Figure 1 gives the distribution of these datasets in the 2D space obtained with the two first PCA (Principal Component Analysis) components based on the Table 1. Although the Figure 1 suggests similarities between several datasets, it is important to remind that multiple intrinsic data properties might impact the prediction in the churn context, such as the existence of small *disjuncts*, the overlap between classes, the noisy data or the borderline instances (see Section 1.2). Hence, directly drawing conclusions on the most suitable machine learning based on the general characteristics given in Table 1 remains challenging.

[2]Before fitting a model, categorical variables are converted to their numerical representation through a *dummification* process where each category becomes a binary variable.

3 Data sampling

3.1 Churn prediction pipeline

This survey provides a comparative study that follows the analysis pipeline depicted in Figure 2. This pipeline unfolds in three parts, namely *(i)* Sampling, *(ii)* Model fitting and *(iii)* Evaluation, through which we sequentially combine several techniques. Our prediction pipeline uses only freely available Python packages (see Appendix B). For the sampling, we explore seven different approaches that either correspond to *oversampling*, *undersampling* or *hybrid* (Section 3). The sampling objective is to transform the original churn dataset into a similar dataset with a better class balance, either by reducing the majority class, expanding the minority class or both. For the model fitting, we consider eleven supervised and semi-supervised techniques,

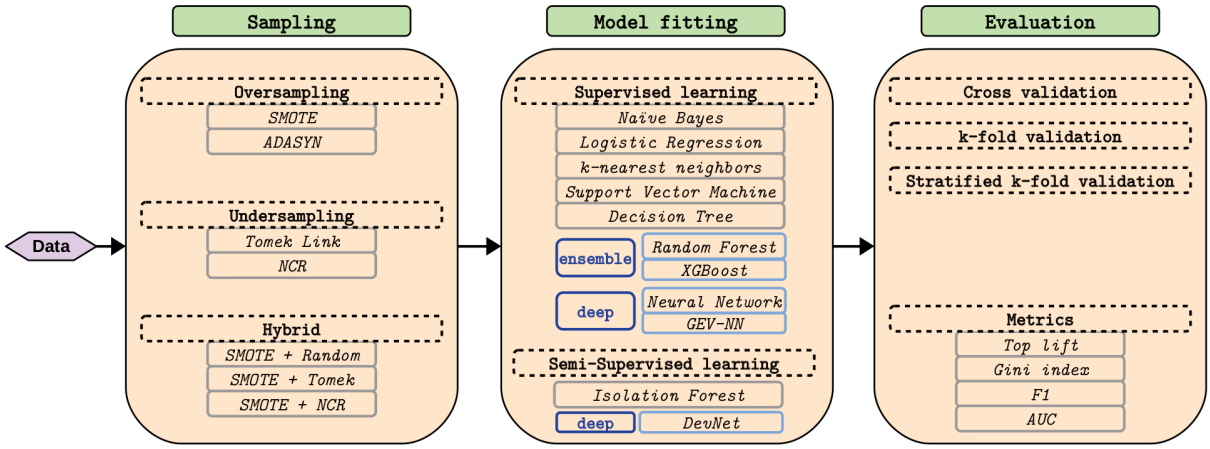


Fig. 2: Machine learning pipeline for churn prediction and analysis

some of which are *ensemble* approaches. Finally, we discuss in the evaluation step three different procedures and four evaluation metrics.

Customer defection is an *infrequent* event that is inevitably associated with a class imbalance hassle that impedes the quality of customer churn prediction. This is particularly true when the classes are highly overlapping and when the minority class is divided into sub-clusters. The class rarity issue is widespread throughout a broad range of contexts beyond churn prediction such as fraudulent credit card usage, telecommunication equipment failure or patient survival prediction. In such contexts, instances of the minority or *positive* class induce a great cost when they are not well classified.

Churn datasets call for the use of various sampling methods [9, 10] to change the class distribution. These methods consist in either introducing data points within the minority class (*oversampling*), removing datapoints from the majority class (*undersampling*) or applying both sampling strategies (*hybrid*). Basic and advanced sampling methods have been proposed [31, 45], and several studies showed that undersampling tends to overtake oversampling [32, 48].

3.2 Oversampling

The oversampling methods generally consist in duplicating instances in the minority class or synthesizing new examples from the available instances. A straightforward oversampling approach is the *random oversampling* that randomly selects the instances to be replicated [87]. However, random replication can impede the decision boundary performance by for instance repeating outliers. We describe in the following two more sophisticated and widely used oversampling approaches, namely the *Synthetic Minority Oversampling Technique* (SMOTE) [31, 51] and the *Adaptative Synthetic Sampling* (ADASYN) [69].

3.2.1 Synthetic Minority Oversampling Technique

The SMOTE technique consists in oversampling the minority class by generating *synthetic* instances along the line segments created by a k-nearest neighbors approach. Specifically, a sample $\mathbf{x}$ is taken at random from the minority class. Then, its k-nearest neighbors $\{\mathbf{x}_i\}_{i\in\{1\dots n\}}$ are considered and used to generate a new synthetic instance following the formula,

$$\mathbf{x}_i^{new} = \mathbf{x} + \mathcal{U}([0,1]) \times (\mathbf{x}_i - \mathbf{x}).$$

While the simple duplication of random instances won't bring any information, new SMOTE instances are plausible observations, similar to original instances from the minority class. However, while SMOTE helps avoiding the overfitting problem, its synthetic instances might be ambiguous in case of strongly overlapping classes.

To address this issue, three extensions have been proposed, namely *Borderline SMOTE* [65], *Borderline Oversampling SVM* [100] and

ADASYN [69]. The *Borderline SMOTE* focuses on generating instances based on observations that are difficult to classify, according to a k-nearest neighbors classifier while *Borderline Oversampling SVM* uses a SVM classifier to generate new instances. In the following, we focus on the third SMOTE extension, ADASYN.

3.2.2 Adaptive Synthetic Method

ADASYN, which is based on SMOTE, adaptively generates minority data instances according to their distributions. Specifically, more synthetic instances are generated in the features space regions where the observations density is low, and conversely, fewer synthetic instances are generated from the high density regions. Hence, ADASYN focuses on the class separation boundary region. As for *Borderline SMOTE* and *Borderline Oversampling SVM*, it would be advisable to remove outliers before applying ADASYN.

3.3 Undersampling

Undersampling techniques delete instances from the majority class or select a subset of examples. A straighforward approach is to randomly delete instances. However, this can be hazardous and make the classification task more complex as it could lead to the removal of important observations. *Tomek Links* [128] and *Neighborhood Cleaning rule* (NCR) [83] are more advanced undersampling strategies.

3.3.1 Neighborhood Cleaning rule

The NCR technique combines two methods that remove from the majority class the instances that are *(i)* redundant and *(ii)* noisy or ambiguous. The first technique is the *Condensed Nearest Neighbor (CNN) Rule* [67], that selects a *minimal consistent set* which is a subset of observations from the majority class that cannot be correctly classified. These samples are considered more relevant for learning. The second approach is the *Edited Nearest Neighbors (ENN) Rule* [143]. It finds and removes noisy and ambiguous instances using a k-`nearest neighbors` approach. With ENN, if a majority class instance is misclassified by its neighbors, it is removed from the dataset. Besides, if a minority class instance is misclassified by its majority class neighbors, the majority class neighbors are also deleted. As shown in [83], NCR is useful to learn a model upon difficult small classes.

3.3.2 Tomek links

This technique builds on the *Condensed Nearest Neighbor (CNN) Rule* [67] and proposes to identify all *cross-class* pairs of datapoints, i.e. pairs that have a sample from the majority and the minority class that are closest neighbors. Hence, majority samples that belong to *Tomek links* are either boundary instances or noisy instances and should be removed. It is also common to combine CNN and Tomek links, as the former will remove redundant samples, while the later deletes noisy/borderline instances.

3.4 Hybrid

Over problems beyond the class distribution skewness are usually encountered with churn-like datasets, such as classes overlapping where majority class examples invade the minority class space and conversely. To create a better class separation while balancing the data, various combinations of upsampling and undersampling methods have been proposed. A straightforward hybrid method is to combine SMOTE and Random Undersampling approaches. Chawla *et al.* shown that this combination performs better than plain undersampling [31]. A more sophisticated combination, proposed by Batista *et al.* [8], combines SMOTE with Tomek Links. It has been successfully applied on an imbalanced genomic dataset.

3.4.1 SMOTE and Random Undersampling

As detailed in Section 3.2.1, SMOTE selects instances that are similar in the features space and synthesizes new instances in between. This technique increases the size of the minority class. A random deletion of instances from the majority class, in combination with this approach, helps to improve the data balancing and the class clusters separation. However, an obvious limitation with the random undersampling stage is that information-rich samples might be deleted from the majority class.

3.4.2 SMOTE and Tomek Links

This combination has been proposed in [8]. It first uses SMOTE to oversample the minority class by creating synthetic samples. However, as class clusters are generally not well defined, synthetic minority class examples can invade the majority class leading to overfitting. Applying Tomek links undersampling procedure on the over-sampled dataset by removing the *cross-class pairs* finally produces a balanced dataset with well defined class clusters.

3.4.3 SMOTE and NCR

For this technical survey, we also propose to combine SMOTE with NCR. Our experimental results (Section 6) show that these two sampling approaches tend to improve some machine learning techniques. NCR has a positive effect on non ensemble approaches. SMOTE preferentially improves LR. By combining SMOTE and NCR, we expect an improvement of several machine learning techniques compared in this survey.

4 Machine learning techniques

We detail in this section the most widespread data mining techniques that have been proposed to tackle the customer churn prediction task. In the following, we mainly focus on *base* machine learning approaches that do not embed any weight correction for the imbalance nature of churn datasets. For our experiments, we rather choose to alleviate the class imbalance using sampling approaches. We invite the reader to refer to the literature which is abundant on the variants of machine learning methods in the context of imbalanced data [47, 64, 89, 150, 151]. We also introduce several machine learning techniques which are suitable for strongly imbalanced data and usually applied in anomaly detection. Hence, Section 4 reports several supervised and semi-supervised learning algorithms and supervised ensemble methods. It also briefly covers some aspects of semi-supervised techniques.

4.1 Supervised learning

4.1.1 k-nearest neighbors

The k-nearest neighbors (k-NN) is a non parametric *memory-based* algorithm. It assigns to an instance $\mathbf{x}_i$ the label that corresponds to the majority label among its k closest training samples Ω_k. Formally,

$$p(C_i = g \mid \mathbf{x}_i) = \frac{1}{K} \sum_{j \in \Omega_k} \mathbb{1}\{\mathbf{x}_j\}$$

where the indicator function $\mathbb{1}$ is defined as being equal to one when $\mathbf{x}_i \in +$, zero otherwise. k-NN depends on two main parameters, namely *(i)* the number of neighbors k and *(ii)* a pairwise metric distance function. For continuous data, the following distance is commonly used $dist(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||$ with $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ ($||.||$ denotes the Frobenius norm).

The simplicity and efficiency of k-NN have made this algorithm very attractive in the field of machine learning. Yet, it has several significant drawbacks when used on churn-like data, as shown in [49, 124].

4.1.2 Naive Bayes Classifier

The Gaussian Naive Bayes (Gnb) classifier [66, 75] is appropriate in a high feature space context, when the density estimation is difficult. The term *naive* results from a simplifying assumption that posits the conditional independence of the d features $\mathbf{x}^j$ given the class value k. This leads to

$$f_k(\mathbf{x}_i) = \prod_{j=1}^{d} f_{kj}(x_{ij}|k). \quad (1)$$

Note that from Eq. 1, we can formally write the Gnb classifier function as a *generalized additive model*. The Gnb classifier is simple, scalable and often outperforms more complex approaches. Although, it appears to be sensitive to the class imbalance issue [13, 35, 110] - in particular due to the strong bias in the prior estimation -, good results can also be achieved for the churn prediction problem [73].

4.1.3 Logistic Regression

The *logistic regression* (LR) models the posterior probability of the classes via a linear function in $\mathbf{x}$. In a binary context, such as churn prediction, the posterior probability of the positive class simply amounts to,

$$P(C = +|\mathbf{x}) = \frac{\exp(\beta_{+0} + \beta_{+}\mathbf{x})}{1 + \exp(\beta_{+0} + \beta_{+}\mathbf{x})}$$

and sum to 1 with $P(C = -|\mathbf{x})$. This model is usually fitted by the maximization of the likelihood $L(\theta)$. The maximization can be made with the Newton-Raphson algorithm, which requires the second derivative of $L(\theta)$. Hence, fitting the LR model amounts to solve,

$$\frac{\partial L(\beta)}{\partial \beta} = \mathbf{X}^{\top}(Y - \mathbf{p}) \text{ and } \frac{\partial^2 L(\beta)}{\partial \beta \partial \beta^{\top}} = -\mathbf{X}^{\top}\mathbf{W}\mathbf{X}$$

where $\mathbf{p}$ is the vector of fitted probabilities, $p_i = P(C_i = +|\mathbf{x}_i)$, and $\mathbf{W}$ is a $n \times n$ diagonal matrix with $w_{ii} = p_i(1 - p_i)$. These equations can get solved repeatedly, following the IRLS algorithm (*iteratively reweighted least squares*) [26].

In the context of unbalanced datasets, it has been shown that the bias of the regression vector intercept tends to be stronger with the unbalanced ratio [102, 114]. This issue can be overcome with a *prior* correction that takes into account the minority class or with a penalized likelihood where the maximum likelihood formula is weighted by the fraction of ones in the target variable [79]. The good performance of LR was previously pointed out in [24].

4.1.4 Support Vector Machine

The *Support Vector Machine* (SVM) was introduced by Vapnik [133] as a kernel based machine learning model for classification and regression task. A recent survey is available in [29]. The SVM classifier aims to construct an optimal separating hyperplane between two linearly separable classes, and can be extended to the non-separable case. The hyperplane can be defined as,

$$\{\mathbf{x}_i | \sum_{j=1}^{d} x_{ij}\beta_j + \beta_0 = \mathbf{x}_i^{\top}\boldsymbol{\beta} + \beta_0 = 0\}$$

where the coefficients β_j are defined up to a multiplicative factor. Thereby the SVM classification problem can be formally written as,

$$min_{\boldsymbol{\beta},\beta_0} ||\boldsymbol{\beta}||^2 \text{ subject to } y_i(\mathbf{x}_i^{\top}\boldsymbol{\beta} + \beta_0) \geq 1, i \in \{1 \dots n\}.$$

In the case of overlapping classes, the SVM classifier can be optimized by allowing for some points to be on the wrong side of the margin, with a *cost* of $\xi = (\xi_1, \dots, \xi_n)$. Hence, bounding the $\sum_i \xi_i$ by a constant $\mathcal{C}$ leads to bounding the total number of misclassifications, and the standard SVM classifier problem can finally be expressed as,

$$min_{\boldsymbol{\beta},\beta_0} ||\boldsymbol{\beta}||^2 \textit{ subject to } \begin{cases} y_i(\mathbf{x}_i^{\top}\boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i \ \forall i \\ \xi_i \geq 0, \sum_i \xi_i \leq \mathcal{C}. \end{cases} \tag{2}$$

The SVM as described above, uncovers linear boundaries in the input feature space. Based on a quadratic programming solution using Lagrange multipliers, we can re-express the SVM classifier problem of Eq. 2 as the following Lagrangian dual objective function,

$$L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,i'=1}^{n} \alpha_i \alpha_{i'} y_i y_{i'} \mathbf{x}_i^{\top} \mathbf{x}_{i'}. \tag{3}$$

We then maximize L_D subject to $0 \leq \alpha_i \leq \mathcal{C}$, $\sum_{i=1}^{n} \alpha_i y_i = 0$ and the Karush-Kuhn-Tucker conditions to find the solution for β.

Note that we can easily enlarge the feature space by using basis expansions h to identify nonlinear boundaries in the original space. This only requires the use of a kernel function, $K(\mathbf{x}, \mathbf{x}') = \langle h(\mathbf{x}), h(\mathbf{x}') \rangle$ at the inner product position of Eq. 3. Three widespread kernel functions are regularly encountered in the SVM literature, namely *Radial basis*, *Neural network* and d^{th}*-Degree polynomial* functions. Since SVM only takes into account the *support vectors*, i.e. the points that are closed to the boundary, it is an interesting candidate for moderately imbalanced datasets [4, 39], although it performs poorly when the class distribution is too skewed [127].

4.1.5 Decision Tree

The *Decision Tree* (DT) method iteratively partitions the feature space into a set of *rectangles*, for which split-points achieve the best fit, until a stopping rule is reached. Within each partition,

or *region* R_m, the target variable Y can be modeled as a constant c_m [22, 52]. A major advantage of tree-based methods is that the recursive binary partition is highly interpretable, and somehow mimics a logical human thinking. For classification purpose, the best split point s is obtained with an impurity measure Q_m that is based on the proportion $\hat{p}_{mk}$ of class k in the region R_m with N_m observations,

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \tag{4}$$

where I is an indicator function. Hence, at node m, observations are classified at the class $k(m)$ that maximizes the proportion in Eq. 4. Three impurity measures are usually encountered in `DT` classification, namely *Misclassification error*, *Gini index* and *Cross-entropy*, the two last measures being generally preferred as they are differentiable and more sensitive to changes in the node probabilities. In a binary classification problem, such as churn, the Gini index and the Cross-entropy measures simple amount to $2p(1-p)$ and $-p\log(p) - (1-p)\log(1-p)$ respectively, weighted by the number of observations in the obtained regions at split. In the context of imbalance datasets authors argue that decision trees are not viable [18, 139], while others propose an insensitive splitting strategies based, for instance, on the Hellinger distance [18, 149].

4.1.6 Deep neural networks

Deep neural techniques have led to state-of-the-art results in various application domains. While generally efficient on datasets with balanced class distribution, deep neural networks performance can be severely impede by imbalanced classes [137, 154]. To overcome this issue, some authors focused on specific loss function [137] or cost-sensitive learning [155] on neural networks.

Recently, Munkhdalai *et al.* [97] proposed an end-to-end deep neural network architecture using the Gumbel distribution as an activation function to tackle the class imbalance issue. Their proposal, so-called GEV-NN (Generalized Extreme Value distribution), outperforms the state-of-the-art baselines while giving a beneficial advantage to interpret variable importance. GEV-NN framework decomposes in three components: *(i)* a feed-forward weighting neural network which provides variable scores to adaptively control input variables [96], *(ii)* an auto-encoder to generate encoded representation and extract efficient features for the minority class [157] and *(iii)* a prediction network that receives a concatenation of scored input variables, encoded representation and features.

A key element of the GEV-NN approach is the Gumbel distribution which is used as an activation function [37]. Also known as Generalized Extreme Value distribution, it is widely used to model the distribution of extreme samples and has been extensively applied to characterize, for instance, age at death or risk assessment in financial context. Its cumulative distribution function is given by $F(x) = e^{-e^{-x}}$. The Gumbel function asymmetry naturally provides a different misclassification penalization on both classes.

4.2 Ensemble Supervised Learning

Ensemble methods are meta-algorithms that combine several models into one predictive model in order to decrease variance (bagging) or bias (boosting).

4.2.1 Bagging and Random Forest

Bagging, which stands for bootstrap aggregation, is an ensemble method for improving unstable estimation or classification schemes. In [19] the author motivated bagging as a variance reduction technique for a given base classifier, such as decision tree. This approach stands out from basic ensemble algorithms by fitting a new model to a bootstrap resample of size less than n. As M models are trained, the final decision $\hat{f}_{bag}$ averages the M decision rules $\hat{f}_m(\mathbf{X})$ obtained from the boostrapped training sets.

The *Random Forest* approach applies bagging to decision trees while sampling the variables [20, 52]. Specifically, the `DT` algorithm creates subpartitions by choosing a variable among the available features and splitting following an *impurity* criterion such as *Gini*. With `RF`, the choice of the variable is done within a random subset of features. This ensemble strategy produces more accurate predictions than `DT`. The easily interpretable decision rules are not available anymore, by contrast with `DT`, however `RF` can provide a measure of feature importance for the model accuracy. Previous

studies highlighted the good performance of `RF` on imbalanced datasets (see for instance [32]).

4.2.2 eXtreme Gradient Boosting

The *boosting* method is similar to bagging in that it combines the results of several classifiers, which are commonly decision trees. Yet, in the boosting strategy, each model tries to minimize the errors of the previous model, by contrast with bagging. The well-known variants of boosting are *Adaboost*, *gradient boosting* and *stochastic gradient boosting* which is the most general and widely used boosting technique.

The key ingredient of *Adaboost* is the observation weights w_i, that are larger for misclassified instances. Hence, the approach forces the model $\hat{f}_m$ to train harder on the data for which it performs poorly and iteratively updates the weights. Each model seeks to minimize the weighted error e_m, which corresponds to the sum of the weights for the misclassified observations. Finally, the boosted estimate is given by $\hat{F} = \sum_{m=1}^{M} \alpha_i \hat{f}_i$ where the $\alpha_i = \frac{\log(1-e_m)}{e_m}$ ensure that the models with less errors have a larger weight in the final decision. Instead of adjusting weights, the *gradient boosting* variant optimize a cost function, while the *stochastic gradient boosting* strategy adds observations and variables sampling at each iteration. The most widely used implementation for boosting is `XGBoost`, a computationally efficient implementation of stochastic gradient boosting [33]. It is interesting to note that with certain parameters setting, the boosting algorithm can emulate `RF`.

When dealing with imbalanced dataset, `XGBoost` has been shown to outperform other types of methods [153]. Yet, some studies are less optimistic and suggest that `XGBoost` should be combined with other ensemble methods to achieve state-of-the-art performance [113].

4.3 Semi-supervised learning

Although very few churn prediction and analysis studies focus on semi-supervised techniques, we briefly address this type of approaches as they could be of great interest for future innovative developments in the field. Semi-supervised techniques have been widely studied in the context of anomaly detection, an extreme case of churn prediction. These approaches combine unsupervised learning - which does not require labeled data - and supervised learning - which learns from labeled data. Semi-supervised techniques can be either generative, discriminative or a combination of both. Generative models attempt to model the joint probabilities of examples and their labels. Once this joint probability is modeled, one can generate new examples for a particular class, as well as determine the most likely class for a given example. Discriminative models restrict themselves to determining the most likely class for a given example by estimating the probability of each class given the data example. Discriminative models do not model the classes, so generation of new class examples is difficult. An example of semi-supervised learning in the context of churn for telecommunication area can be found in [11]. More recently, in [144] the authors propose to combine a semi-supervised approach with *Metacost*, a cost-sensitive model, in an ensemble strategy.

In the context of anomaly detection, One-Class Support Vector Machine (ocSVM) [116] and Isolation Forest (iForest) [88] are among the most widely used semi-supervised anomaly detection algorithms. ocSVM identifies the smallest hypersphere containing the majority class datapoints [126]. As for SVM (Section 4.1.4), ocSVM supports the introduction of a kernel function to allow for more flexibility. Although interesting, this approach does not perform well on large databases [135]. Indeed, ocSVM introduces significant memory requirements and is computationally expensive when the number of instances increases. By contrast, iForest [88] has a low linear time complexity and a small memory requirement. This approach posits that outlier datapoints can be isolated more easily than normal datapoints. iForest is based on a recursive 2D partitioning that can be represented by a tree structure (Section 4.1.5), so-called *Isolation Tree*. Anomalies or outliers correspond to leaf node with the smaller path length in the tree. This approach has been shown to perform well on imbalanced datasets in several studies [104, 135].

Recently, Pang *et al.* [104] proposed a semi-supervised *deep anomaly detection* framework, so-called `DevNet`, which outperforms state-of-the-art methods. DevNet relies on neural deviation learning, requires few labeled anomalies and uses a

prior probability that enforces statistically significant deviations of the anomaly scores. Specifically, DevNet decomposes as follows: *(i)* assigning an anomaly score to each training data object, *(ii)* providing a reference score based on the mean of the anomaly scores of normal data objects based on a prior probability and *(iii)* defining a loss function (*deviation loss*) to enforce statistically significant deviations of the anomaly scores as compared to normal data objects. A strength of `DevNet` framework is that it can naturally accommodate anomalies with different anomalous behaviors.

5 Model validation

5.1 Validation strategies

Model validation aims at estimating how effective is the model for the predictions of *unseen* instances. A straightforward validation principle is the *holdout set*, where some data subset that was not used for the training is used for evaluating the predictions of the trained model. We describe and discuss in the following subsections two validation approaches that build on and improve the *holdout set* idea.

5.1.1 Cross-validation

A clear disadvantage of the *holdout set* strategy is that a portion of the data is *lost* for the model training. This especially becomes an issue when the dataset is small. The *cross-validation* addresses this issue by defining a training set and a validation set, and then switching the sets before combining the two validation scores.

5.1.2 K-fold validation

The aforementioned cross-validation idea can be expanded to more subsets or *folds*, which is of great interest when data are scarce. The dataset is split in K subsets of equivalent sizes and the model is fitted on $K-1$ folds. The prediction error of the fitted model is then calculated on the k^{th} *unseen* subset. This strategy is repeated K times while taking another subset as validation set. Finally, the K estimates are combined. This is known as *K-fold cross-validation*. A typical value for K is 5 or 10 [21, 25, 80].

The K-fold cross validation is not appropriate as is for evaluating models on churn-like datasets which are typically imbalanced [68]. Indeed, as the data is split into K-fold with a uniform probability distribution, it is likely that one or more folds will have few or no examples from the minority class, which in turn severely impedes the model training.

5.1.3 Stratified K-fold validation

The dataset imbalance issue can be addressed with a *stratified* sampling, where the target variable y is used to control the sampling process. Hence, for a K-fold cross validation procedure, each fold will roughly contain the same distribution of class labels as the whole dataset.

The stratified K-fold validation is the validation strategy retained for our experiments, as it is the validation procedure that would be applicable in both balance and imbalance class contexts.

5.2 Evaluation metrics

The assessment procedure of a predictive model can rely on different metrics. Several metrics have been proposed in marketing and machine learning areas. We present in the following the most common metrics and emphasize their strengths and drawbacks when tackling churn-like data.

5.2.1 Metrics based on probability

Top decile-lift The top decile-lift is one of the oldest evaluation metric among marketers to evaluate and compare predictive models. It is also a widespread measure in the churn literature [24, 84]. The lift measure considers the observations/-customers in order of their predicted probability of being churners. Specifically, when focusing on the 10% riskiest customers, the top decile-lift gives the ratio between the proportion of churners in the risky segment, $\pi_{10\%}$, and the whole proportion of churners in the validation set, π, $lift_{10\%} = \hat{\pi_{10\%}}/\hat{\pi}$. Hence, this measure evaluates if churners predicted as risky are actually at risk. The top decile-lift is directly related to the profitability or *gain* [99] which is formally defined as,

$$GAIN = n\alpha\hat{\pi}(\Delta lift_{10\%})\left[\gamma LVC - \delta(\gamma - \psi)\right]$$

where n is the number of customers, α is the number of customers under study (here, 10%),

$\Delta lift_{10\%}$ is the top decile-lift increase, γ is the success rate of the incentive among the churners, LVC is the lifetime value of a customer [60], δ is the incentive cost among customers and ψ is the success rate of the incentive among the non-churners.

Gini coefficient While the top decile-lift measure focuses on the 10% riskiest customer, the Gini coefficient takes also into account the less risky customers. This coefficient is formally defined as follows,

$$Gini = \frac{2}{M}\sum_{\ell=1}^{M}(\pi_\ell^c - \pi_\ell)$$

where M is the size of the validation set, π_ℓ^c is the fraction of actual churners above the threshold $\hat{f}(\mathbf{x}_i)$, π_ℓ the fraction of customers above the same threshold $\hat{f}(\mathbf{x}_i)$ and $\hat{f}(\mathbf{x}_i)$ corresponds to a predicted churn probability. In the same way as for the top decile-lift, the Gini coefficient takes advantage of the predicted churn probabilities. It is also a complementary measure as it considers the ability to predict less risky customers.

5.2.2 Metrics from confusion matrix

Let TP be the True Positive, the number of customers predicted as churners who actually churned, and FP (False Positive) the number of customers predicted as churners who did not churn. Similarly, we can define TN (True Negative), the number of customers predicted as non churners who did not resign, and FN (False Negative), the number of customers predicted as non churner who actually churned. Hence, the number of correct predictions would be (TP+TN). By dividing with the total number of predictions (TP+TN+FP+FN), we obtain the *accuracy* that can summarize the classification performance of a model. However, using accuracy for churn predictive model evaluation is not appropriate as the data is strongly imbalanced [139]. We present below two metrics that are advisable in the churn context.

$\mathbf{F_1}$ score This score summarizes the *Precision* and *Recall* metrics. The *Precision* estimates the ability of the model to obtain TP among its positive predictions, i.e. $Precision = \frac{TP}{TP+FP}$. It is a complementary measure to the *Recall*, that evaluates the ability of the model to recover TP, i.e. $Recall = \frac{TP}{TP+FN}$. The F_1 score proposes an harmonic mean of these two metrics, $F_1 = 2 \times \frac{Precision \cdot Recall}{Precision+Recall}$.

Area Under the Curve (AUC) The AUC measure first requires to express the performance of the model with a *Receiver Operating Characteristic* (ROC) curve. This curve gives the True Positive Rate ($TPR = \frac{TP}{TP+FN}$) as a function of the False Positive Rate ($FPR = \frac{FP}{FP+TN}$) for a series of decision thresholds. The AUC corresponds to the *Area Under the Curve*. Hence, it provides an aggregated performance measure for all possible ranking thresholds. This measure can be interpreted as the probability that the model correctly classifies an instance as positive as compared to a negative instance.

6 Experiments

This section presents the churn prediction evaluations for several variants of our pipeline (Fig. 2). The retained datasets cover a range of domains where churn is regarded as a core issue (Table 1). We first summarize the experiments settings and necessary preprocessing steps. We then detail the machine learning performance on these datasets when associated to a sampling approach or not.

6.1 Experimental settings

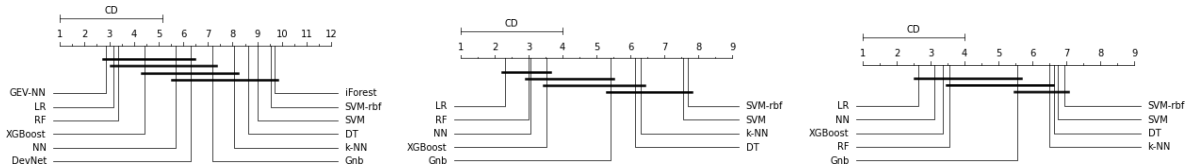
We consider nine popular supervised algorithms - namely K-Nearest Neighbors (k-`NN`), Gaussian Naive Bayes (`Gnb`), Logistic Regression (`LR`), Support Vector Machine with Radial Basis Function kernel (`SVM-rbf`) and without kernel (`SVM`) [3], Decision Tree (`DT`), Random Forest (`RF`), `XGBoost`, a feed-forward neural network (`NN`) and `GEV-NN` - in association with different undersampling, oversampling and hybrid sampling strategies. Two semi-supervised techniques are also considered, namely `iForest` and `DevNet` [4]. All the implementations are freely available from python packages. We mainly kept default parameters (Appendix B). In this survey, we focus on the association

[3] In our experiments, we consider both the linear `SVM` and the `SVM-rbf`, which is a kernel `SVM` using the *Radial basis* function, following Amnueypornsakul *et al.* results [6]

[4] `GEV-NN`, `iForest` and `DevNet` being specifically designed for imbalance binary classification or anomaly detection, these approaches are only evaluated without sampling.

Table 2: AUC Classification results *(No Sampling approach)*.

Dataset	k-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	GEV-NN	iForest	DevNet
Fraud	0.8990	0.9217	**0.9766**	0.9465	0.9441	0.8660	0.9466	0.9456	0.9573	0.9707	0.9459	0.9621
K2009	0.5004	0.5002	**0.5135**	0.5052	0.4989	0.4993	0.5114	0.5112	0.4999	0.5058	0.4975	0.4997
Thyroid	0.7598	0.5876	0.8645	0.9821	0.9786	0.9834	**0.9996**	0.9994	0.6223	0.9941	0.7551	0.7924
KKBox	0.5835	0.6468	0.6763	0.5022	0.4983	0.5302	0.6442	0.6800	0.6994	**0.7054**	0.5757	0.6184
UCI	0.7731	0.8477	0.8244	0.5963	0.7528	0.8447	**0.9182**	0.9174	0.8033	0.9137	0.6711	0.8139
Campaign	0.7596	0.8271	0.9331	0.5971	0.6451	0.7290	**0.9395**	0.9322	0.9134	0.9362	0.7338	0.7687
HR	0.6575	0.7442	**0.8596**	0.8091	0.4984	0.6053	0.7867	0.7993	0.6310	0.8558	0.6243	0.7677
TelE	0.8226	0.7505	0.7584	0.5335	0.6098	0.8514	0.9380	**0.9411**	0.8924	0.9320	0.5883	0.6769
News	0.7484	0.5655	0.8369	0.5958	0.6227	0.6754	**0.8615**	0.8323	0.8266	0.8525	0.5364	0.7003
Bank	0.7768	0.7166	0.8322	0.6645	0.7248	0.6908	0.8506	0.8216	0.8295	**0.8583**	0.6969	0.7686
Mobile	0.7567	0.7201	0.9030	0.4605	0.5463	0.6660	0.8095	0.7816	**0.9118**	0.8916	0.7963	0.8576
TelC	0.7822	0.8245	**0.8458**	0.6498	0.6548	0.6555	0.8210	0.7983	0.8357	0.8404	0.4542	0.7897
C2C	0.4387	0.5181	**0.5222**	0.4578	0.4656	0.4440	0.3518	0.3862	0.4541	0.3698	0.4985	0.4878
Member	0.5827	0.5914	0.6146	0.4874	0.5088	0.5462	0.6130	0.5987	0.6084	0.6243	0.5606	**0.6283**
SATO	0.6900	0.7272	0.7594	0.7116	0.7153	0.6365	**0.7882**	0.7396	0.7367	0.7600	0.6321	0.7030
DSN	0.6576	0.6671	0.7319	0.6868	0.6293	0.7350	**0.8590**	0.8516	0.6537	0.7493	0.6282	0.6941
$\widetilde{AUC}$	0.7526	0.7184	0.8283	0.5967	0.6260	0.6707	0.8358	0.8104	0.7700	**0.8542**	0.6262	0.7353
$\overline{Rank}$	8.06	7.19	3.19	9.00	9.56	8.62	3.38	4.44	5.69	**2.88**	9.69	6.31



(a) No sampling (with AD methods) (b) SMOTE (c) ADASYN

Fig. 3: Approaches similarities based on Critical Difference diagrams *(Oversampling)*

between base machine learning techniques, sampling strategies and datasets in a churn prediction context. Hence, we do not resort to hyperparameters tuning. We adjusted the sampling so as to obtain a balance distribution as suggested by the AUC results presented in [142], where the authors show that the best class distribution for learning tends to be near the balanced class distribution. Our evaluations follow a stratified K-fold cross validation procedure where $K = 5$ ($K \in [5, 10]$ is typically advised in the literature [21, 25, 80]).

Several preprocessing steps where performed on all datasets. First, we exclude features that take a unique value for each observation (e.g. customer ID, phone number, address). Besides, only observations with less than 20% missing feature values are retained. All numeric variables are standardized. The missing values are replaced by the feature mean for numeric variables and the majority category for a categorical variable (see Appendix A for details).

6.2 Experimental results

6.2.1 Learning with sampling

We evaluate the churn prediction for all the pipeline alternative as given in Figure 2. The evaluation procedure follows a stratified 5-fold cross-validation. Results are given in AUC without sampling (Table 2), and with various oversampling (Table 3), undersampling (Table 4) and hybrid sampling approaches (Tables 5 & 6). The mean rank and the median AUC ($\widetilde{AUC}$) for each algorithm are given in the last two columns of each table.

The median AUC ($\widetilde{AUC}$) given in Tables 2 to 6 indicate only small $\widetilde{AUC}$ variations over sampling strategies. We can notice that the sampling methods generally degrade $\widetilde{AUC}$ for `RF` as compared to results obtained without sampling (from $\widetilde{AUC} = 0.8358$ to $\widetilde{AUC} = 0.8020$). Only SMOTE combined with NCR strongly increases `RF` $\widetilde{AUC}$ (0.8404). On average, `XGBoost` performance is

Table 3: Oversampling methods: AUC Classification results *(top, SMOTE; bottom, ADASYN).*

SMOTE	k-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	Max-Min
Fraud	0.9054	0.9238	**0.9751**	0.7062	0.3136	0.8408	0.9693	0.9462	0.9648	0.6615
K2009	0.5001	0.4991	**0.5135**	0.4965	0.4993	0.5022	0.5023	0.4991	0.5054	0.0170
Thyroid	0.8006	0.5644	0.9039	0.8394	0.7128	0.9846	**0.9995**	0.9992	0.8624	0.4351
KKBox	0.5918	0.6430	0.6763	0.5590	0.4370	0.5272	0.6129	0.6414	**0.6851**	0.2481
UCI	0.7871	0.8273	0.8278	0.5327	0.7729	0.8490	0.9130	**0.9154**	0.8701	0.3827
Campaign	0.7657	0.7712	0.9311	0.6063	0.5761	0.7521	**0.9406**	0.9318	0.9258	0.3645
HR	0.6631	0.7168	**0.8501**	0.7066	0.5040	0.6309	0.7304	0.7905	0.7412	0.3461
TelE	0.8277	0.7497	0.7626	0.5470	0.5692	0.8482	0.9373	**0.9421**	0.9094	0.3951
News	0.7452	0.5664	0.8336	0.5651	0.6337	0.6881	0.8136	0.8333	**0.8428**	0.2777
Bank	0.7744	0.7861	0.8325	0.5830	0.7204	0.6940	0.8255	0.8234	**0.8422**	0.2592
Mobile	0.6479	0.6993	0.8942	0.6185	0.4404	0.6570	0.8138	0.7835	**0.9124**	0.4720
TelC	0.7650	0.8224	**0.8451**	0.5098	0.6881	0.6656	0.8007	0.7941	0.8439	0.3353
C2C	0.4375	0.5033	**0.5160**	0.4965	0.4751	0.4415	0.3944	0.3878	0.4348	0.1282
Member	0.5865	0.5936	**0.6213**	0.5176	0.5187	0.5489	0.6122	0.5959	0.6203	0.1037
SATO	0.6900	0.7272	0.7594	0.7116	0.7152	0.6385	**0.7601**	0.7396	0.7393	0.1216
DSN	0.6576	0.6671	0.7319	0.6868	0.6298	0.7314	0.8166	**0.8516**	0.6584	0.2218
$\widetilde{AUC}$	0.7176	0.7081	0.8302	0.5740	0.5726	0.6768	0.8137	0.8088	**0.8425**	
$\overline{Rank}$	6.31	5.38	**2.31**	7.56	7.69	6.12	<u>3.00</u>	3.56	3.06	

ADASYN	k-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	Max-Min
Fraud	0.8990	0.9217	**0.9766**	0.9466	0.9428	0.8621	0.9514	0.9456	0.9635	0.1145
K2009	0.5007	0.4987	**0.5137**	0.5032	0.5053	0.4985	0.4945	0.5013	0.5013	0.0192
Thyroid	0.7598	0.5876	0.8645	0.9821	0.9786	0.9806	**0.9995**	0.9994	0.6381	0.4119
KKBox	0.5899	0.6421	0.6777	0.5491	0.5239	0.5268	0.6107	0.6468	**0.6923**	0.1684
UCI	0.7791	0.8293	0.8276	0.5512	0.7601	0.8483	0.9112	**0.9156**	0.8712	0.3644
Campaign	0.7596	0.8271	0.9331	0.5971	0.6505	0.7269	**0.9398**	0.9322	0.9156	0.3427
HR	0.6612	0.7241	**0.8476**	0.6768	0.5026	0.5814	0.7597	0.7978	0.7566	0.3450
TelE	0.8248	0.7551	0.7634	0.4678	0.5559	0.8382	0.9364	**0.9418**	0.9097	0.4740
News	0.7377	0.5661	0.8309	0.5467	0.6419	0.6876	0.8107	0.8328	**0.8384**	0.2917
Bank	0.7647	0.7865	0.8315	0.6403	0.7123	0.6865	0.8197	0.8225	**0.8408**	0.2005
Mobile	0.6203	0.6814	0.8848	0.1398	0.4864	0.6644	0.7970	0.7937	**0.9100**	0.7702
TelC	0.7515	0.8311	**0.8444**	0.4093	0.6822	0.6546	0.8003	0.7968	0.8429	0.4351
C2C	0.4408	0.5031	0.5171	**0.5271**	0.4734	0.4401	0.3971	0.3905	0.4606	0.1366
Member	0.5791	0.5958	**0.6266**	0.5015	0.5304	0.5479	0.6092	0.5973	0.6153	0.1251
SATO	0.6900	0.7272	0.7594	0.7116	0.7153	0.6375	0.7494	0.7396	**0.7613**	0.1238
DSN	0.6576	0.6671	0.7319	0.6869	0.6297	0.7336	0.8038	**0.8516**	0.6602	0.2219
$\widetilde{AUC}$	0.7138	0.7028	**0.8292**	0.5502	0.6358	0.6754	0.8020	0.8101	0.7998	
$\overline{Rank}$	6.50	5.56	**2.62**	6.75	6.94	6.62	3.56	3.31	<u>3.12</u>	

slightly improved when using NCR and SMOTE combined with NCR (+0.0188 and +0.0186). The approach that benefits the most from the sampling strategies is `NN`, with a maximum $\widetilde{AUC}$ increase of 0.0728 when using SMOTE combined with Tomek Links. The top approaches over all datasets and sampling strategies are `LR`, `RF`, `XGBoost` and `NN`, with a mean rank of 2.61, 3.21, 3.33 and 3.66 respectively. When considering particular methods and datasets, greater improvement can be

Table 4: Undersampling methods: AUC Classification results (top, NCR; bottom, Tomek).

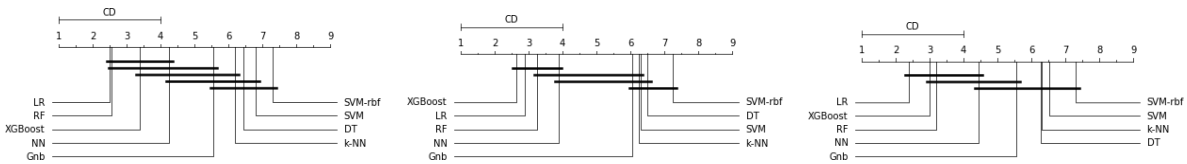
NCR	k-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	Max-Min
Fraud	0.9000	0.9226	**0.9762**	0.9472	0.9423	0.8803	0.9496	0.9405	0.9664	0.0959
K2009	0.5061	0.5004	0.5146	0.5017	0.5033	0.5027	0.5105	**0.5149**	0.5065	0.0145
Thyroid	0.7650	0.5887	0.8574	0.9726	0.9548	0.9824	**0.9993**	0.9992	0.6557	0.4106
KKBox	0.6099	0.6483	0.6762	0.5353	0.4797	0.5488	0.6397	0.6824	**0.7002**	0.2205
UCI	0.8052	0.8512	0.8234	0.6309	0.6288	0.8500	0.9145	**0.9200**	0.8118	0.2912
Campaign	0.7789	0.8150	0.9287	0.6751	0.6828	0.7934	**0.9374**	0.9353	0.9017	0.2623
HR	0.6761	0.7350	**0.8580**	0.8332	0.4984	0.6194	0.7430	0.7918	0.6803	0.3596
TelE	0.8295	0.7468	0.7615	0.4438	0.6260	0.8583	0.9394	**0.9417**	0.8922	0.4979
News	0.7804	0.5672	0.8371	0.6727	0.6745	0.7306	0.8298	**0.8399**	0.8189	0.2727
Bank	0.7994	0.7460	0.8313	0.6647	0.7938	0.7327	0.8361	**0.8369**	0.8335	0.1722
Mobile	0.7274	0.7255	0.8867	0.4912	0.6077	0.6710	0.7862	0.7745	**0.8883**	0.3971
TelC	0.8028	0.8205	**0.8438**	0.8007	0.7920	0.7136	0.8201	0.8216	0.8380	0.1302
C2C	0.4069	0.4890	0.4985	**0.5659**	0.4533	0.4146	0.3527	0.3668	0.4360	0.2132
Member	0.5915	0.5886	0.6209	0.4915	0.5512	0.5693	0.6129	0.6104	**0.6218**	0.1303
SATO	0.7028	0.7348	0.7645	**0.7741**	0.7089	0.6615	0.7631	0.7685	0.7198	0.1126
DSN	0.6634	0.6328	0.7311	0.7186	0.6308	0.7214	0.8173	**0.8672**	0.6952	0.2364
$\widetilde{AUC}$	0.7462	0.7302	0.8274	0.6687	0.6298	0.7175	0.8187	**0.8292**	0.7658	
$\overline{Rank}$	6.25	6.06	<u>2.88</u>	6.31	7.25	6.50	3.25	**2.62**	3.88	

Tomek	k-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	Max-Min
Fraud	0.8990	0.9217	**0.9766**	0.9457	0.9445	0.8793	0.9477	0.9446	0.9629	0.0973
K2009	0.4999	0.5002	**0.5138**	0.5007	0.4961	0.5044	0.5106	0.5017	0.4944	0.0194
Thyroid	0.7607	0.5879	0.8638	0.9825	0.9769	0.9825	**0.9996**	0.9994	0.6779	0.4117
KKBox	0.5873	0.6470	0.6761	0.5335	0.4762	0.5337	0.6189	0.6805	**0.6994**	0.2232
UCI	0.7773	0.8487	0.8252	0.6336	0.7540	0.8431	0.9134	**0.9150**	0.8241	0.2814
Campaign	0.7628	0.8252	0.9324	0.5985	0.6502	0.7449	**0.9391**	0.9341	0.9141	0.3406
HR	0.6671	0.7426	**0.8585**	0.8260	0.4990	0.6152	0.7481	0.7997	0.6281	0.3595
TelE	0.8236	0.7501	0.7589	0.5695	0.6031	0.8543	0.9379	**0.9412**	0.8906	0.3717
News	0.7533	0.5653	**0.8376**	0.6010	0.6395	0.6909	0.8132	0.8365	0.8263	0.2723
Bank	0.7797	0.7196	**0.8321**	0.5793	0.7500	0.6963	0.8243	0.8253	0.8314	0.2528
Mobile	0.7514	0.7182	0.8991	0.3813	0.5211	0.6619	0.7880	0.7868	**0.9061**	0.5248
TelC	0.7882	0.8240	**0.8459**	0.7019	0.7055	0.6683	0.8001	0.8017	0.8375	0.1776
C2C	0.4359	0.5164	**0.5208**	0.4803	0.4567	0.4427	0.3863	0.3855	0.4488	0.1353
Member	0.5890	0.5924	**0.6170**	0.4801	0.5162	0.5474	0.6036	0.6033	0.5960	0.1369
SATO	0.6891	0.7247	**0.7573**	0.7253	0.7029	0.6415	0.7483	0.7514	0.7034	0.1158
DSN	0.6535	0.6632	0.7286	0.7000	0.6241	0.7293	0.8294	**0.8655**	0.6518	0.2414
$\widetilde{AUC}$	0.7524	0.7189	**0.8286**	0.5998	0.6318	0.6796	0.8067	0.8135	0.7638	
$\overline{Rank}$	6.31	5.56	**2.38**	6.50	7.31	6.31	3.19	<u>3.00</u>	4.44	

observed. For instance, combining `SVM` with NCR increases AUC of 0.1081 on *C2C*. The performance of `XGBoost` is also increased when using the hybrid sampling SMOTE & Tomek Links (from 0.8516 to 0.8694) on *DSN*. We notice an AUC increase of 0.0124 when using SMOTE in combination with NCR on *Member* with `LR`. Hence, while a global improvement of *all* the machine learning

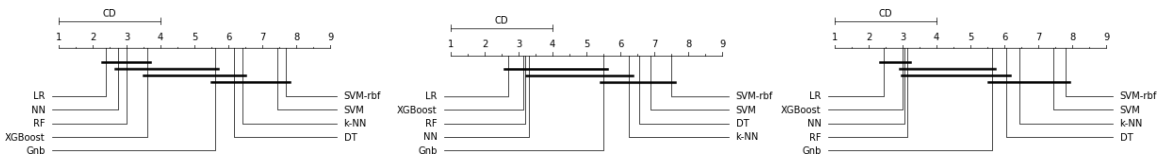
Table 5: Hybrid methods: *AUC* Classification results

	k-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	Max-Min
Dataset	SMOTE + Random undersampling									
Fraud	0.9054	0.9238	**0.9751**	0.7758	0.3237	0.8357	0.9694	0.9462	0.9746	0.6514
K2009	0.5001	0.4991	**0.5135**	0.4967	0.5012	0.5023	0.5055	0.4991	0.5038	0.0168
Thyroid	0.8006	0.5644	0.9039	0.8394	0.7224	0.9835	**0.9995**	0.9992	0.8548	0.4351
KKBox	0.5918	0.6430	0.6763	0.5654	0.4628	0.5277	0.6199	0.6480	**0.6997**	0.2369
UCI	0.7871	0.8273	0.8278	0.5326	0.7727	0.8499	**0.9168**	0.9154	0.8715	0.3842
Campaign	0.7657	0.7712	0.9311	0.6063	0.5761	0.7500	**0.9403**	0.9318	0.9279	0.3642
HR	0.6631	0.7168	**0.8501**	0.7065	0.5031	0.6295	0.7560	0.7905	0.7601	0.3470
TelE	0.8275	0.7497	0.7626	0.5756	0.5677	0.8486	0.9373	**0.9421**	0.9084	0.3744
News	0.7454	0.5664	0.8337	0.5652	0.6337	0.6871	0.8117	0.8333	**0.8415**	0.2763
Bank	0.7744	0.7861	0.8325	0.5830	0.7204	0.6936	0.8240	0.8234	**0.8430**	0.2600
Mobile	0.6586	0.6993	0.8942	0.5304	0.5588	0.6586	0.7953	0.7835	**0.9080**	0.3776
TelC	0.7650	0.8224	**0.8451**	0.5785	0.6881	0.6675	0.7947	0.7941	0.8419	0.2666
C2C	0.4375	0.5033	**0.5160**	0.5097	0.4783	0.4429	0.3964	0.3878	0.4557	0.1282
Member	0.5866	0.5936	0.6213	0.5179	0.5169	0.5426	0.5985	0.5959	**0.6235**	0.1066
SATO	0.6900	0.7272	**0.7594**	0.7117	0.7152	0.6375	0.7491	0.7396	0.7405	0.1219
DSN	0.6576	0.6671	0.7319	0.6868	0.6293	0.7343	0.8156	**0.8516**	0.6677	0.2223
$\widetilde{AUC}$	0.7177	0.7081	0.8302	0.5771	0.5719	0.6773	0.8035	0.8088	**0.8417**	
$\overline{Rank}$	6.38	5.56	**2.38**	7.44	7.69	6.19	3.00	3.62	2.75	



(a) No sampling (b) Neighborhood Cleaning rule (c) Tomek

Fig. 4: Approaches similarities based on Critical Difference diagrams *(Undersampling)*



(a) SMOTE + Random undersampling (b) SMOTE + NCR (c) SMOTE + Tomek

Fig. 5: Approaches similarities based on Critical Difference diagrams *(Hybrid sampling)*

approaches cannot be observed, *local* improvements can be observed for given methods and samplings, depending on the datasets.

It is important to highlight the almost systematic complementary behaviors of `LR`, `RF`, `XGBoost` and `NN` overall datasets. As can be seen from Table 3 to Table 6, whenever `LR` is not the best approach, `XGBoost`, `RF` or `NN` outperforms the other machine learning techniques, and conversely (see for instance bold values of Table 4, Tomek

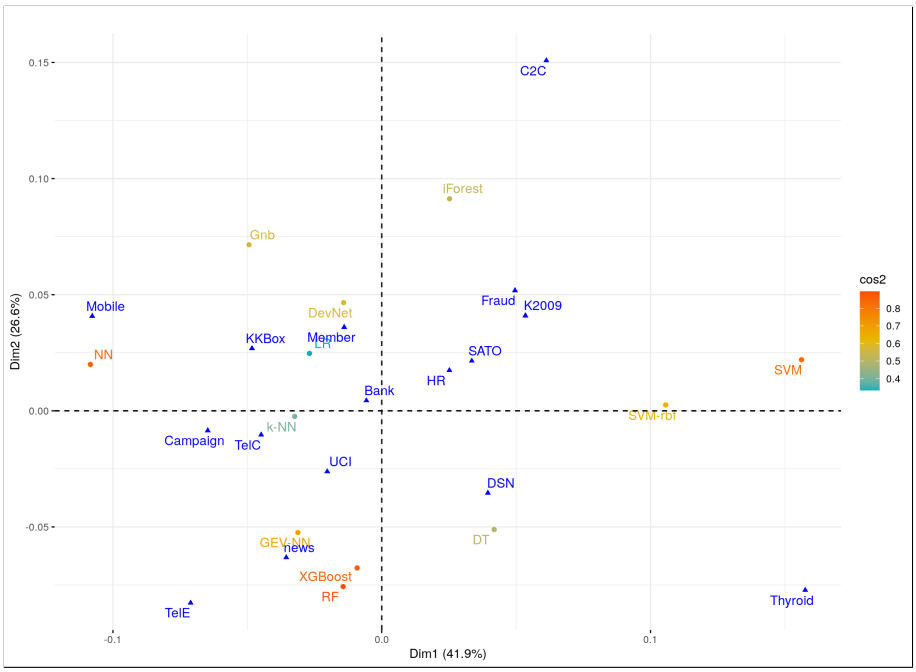
Table 6: Hybrid methods: *AUC* Classification results *(top, SMOTE-Tomek; bottom, SMOTE-NCR)*

ST-T.L.	*k*-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	Max-Min
Fraud	0.9054	0.9238	**0.9751**	0.7893	0.3207	0.8377	0.9679	0.9462	0.9698	0.6544
K2009	0.5001	0.4991	**0.5135**	0.4999	0.4985	0.5050	0.5088	0.5084	0.5047	0.0150
Thyroid	0.8006	0.5656	0.9035	0.8683	0.7173	0.9826	**0.9995**	0.9993	0.8685	0.4339
KKBox	0.5926	0.6432	0.6764	0.5098	0.4378	0.5291	0.6142	0.6494	**0.7017**	0.2639
UCI	0.7871	0.8273	0.8278	0.5685	0.7700	0.8457	**0.9189**	0.9150	0.8750	0.3504
Campaign	0.7633	0.7708	0.9304	0.5914	0.5887	0.7491	**0.9399**	0.9335	0.9294	0.3512
HR	0.6631	0.7168	**0.8501**	0.7065	0.5018	0.6298	0.7533	0.7905	0.7378	0.3483
TelE	0.8270	0.7496	0.7628	0.5042	0.5492	0.8482	0.9359	**0.9402**	0.9098	0.4360
News	0.7450	0.5690	0.8335	0.5414	0.6363	0.6882	0.8124	0.8273	**0.8435**	0.3021
Bank	0.7746	0.7860	0.8325	0.5952	0.7295	0.6958	0.8232	0.8273	**0.8420**	0.2468
Mobile	0.6351	0.6995	0.8941	0.2132	0.5761	0.6639	0.7951	0.7939	**0.9073**	0.6941
TelC	0.7708	0.8223	**0.8449**	0.5011	0.7051	0.6717	0.7980	0.7960	0.8447	0.3438
C2C	0.4370	0.5034	**0.5158**	0.4691	0.4705	0.4419	0.3894	0.3846	0.4574	0.1312
Member	0.5852	0.5925	0.6201	0.4627	0.5118	0.5470	0.6007	0.6029	**0.6206**	0.1579
SATO	0.6986	0.7219	0.7581	0.7438	0.7122	0.6375	0.7565	**0.7602**	0.7388	0.1227
DSN	0.6531	0.6644	0.7304	0.7125	0.6257	0.7314	0.8066	**0.8694**	0.6691	0.2437
$\widetilde{AUC}$	0.7218	0.7082	0.8302	0.5550	0.5824	0.6800	0.8023	0.8116	**0.8428**	
$\overline{Rank}$	6.44	5.62	**2.44**	7.44	7.81	6.06	3.12	3.00	3.06	

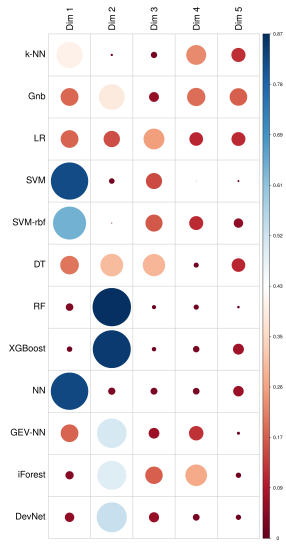
ST-NCR	*k*-NN	Gnb	LR	SVM	SVM-rbf	DT	RF	XGBoost	NN	Max-Min
Fraud	0.9054	0.9238	**0.9751**	0.8562	0.3237	0.8358	0.9681	0.9452	0.9642	0.6514
K2009	0.5003	0.4995	**0.5153**	0.4972	0.5044	0.4984	0.4944	0.4974	0.5063	0.0209
Thyroid	0.8004	0.5672	0.9032	0.8399	0.7201	0.9865	**0.9994**	0.9991	0.8587	0.4322
KKBox	0.6054	0.6485	0.6801	0.5243	0.4790	0.5479	0.6665	0.6705	**0.7004**	0.2214
UCI	0.7856	0.8341	0.8274	0.5683	0.7524	0.8537	0.9144	**0.9187**	0.8726	0.3504
Campaign	0.7536	0.7706	0.9284	0.6180	0.5952	0.7495	**0.9402**	0.9311	0.9223	0.3450
HR	0.6569	0.7080	**0.8274**	0.7500	0.4992	0.6620	0.7911	0.8031	0.7334	0.3282
TelE	0.8178	0.7465	0.7633	0.5954	0.5967	0.8524	0.9364	**0.9413**	0.9095	0.3459
News	0.7495	0.5936	0.8388	0.6342	0.7010	0.7323	**0.8537**	0.8477	0.8404	0.2601
Bank	0.7781	0.7827	0.8320	0.6542	0.7773	0.7232	**0.8495**	0.8423	0.8414	0.1953
Mobile	0.6260	0.6984	**0.8799**	0.6541	0.5329	0.5825	0.6210	0.6689	0.8747	0.3470
TelC	0.7754	0.8176	**0.8435**	0.6038	0.7778	0.7139	0.8312	0.8156	0.8425	0.2397
C2C	0.4225	0.4963	**0.5022**	0.4692	0.4468	0.4101	0.3153	0.3638	0.4563	0.1869
Member	0.5860	0.5791	0.6270	0.4485	0.5654	0.5590	0.6218	0.6125	**0.6354**	0.1869
SATO	0.7053	0.7387	0.7575	0.7556	0.7138	0.6850	**0.7811**	0.7671	0.7371	0.0961
DSN	0.6513	0.6515	0.7392	0.7334	0.6393	0.6986	0.8556	**0.8661**	0.6909	0.2268
$\widetilde{AUC}$	0.7274	0.7032	0.8274	0.6261	0.5960	0.7062	0.8404	0.8290	**0.8409**	
$\overline{Rank}$	6.25	5.50	**2.69**	6.88	7.50	6.56	3.19	3.12	3.31	

Links or Tables 5, SMOTE & Random Undersampling). This finding suggests the use of an *ensemble* method based on the top four approaches, `LR`, `XGBoost`, `RF` and `NN` (see Section 6.2.3).

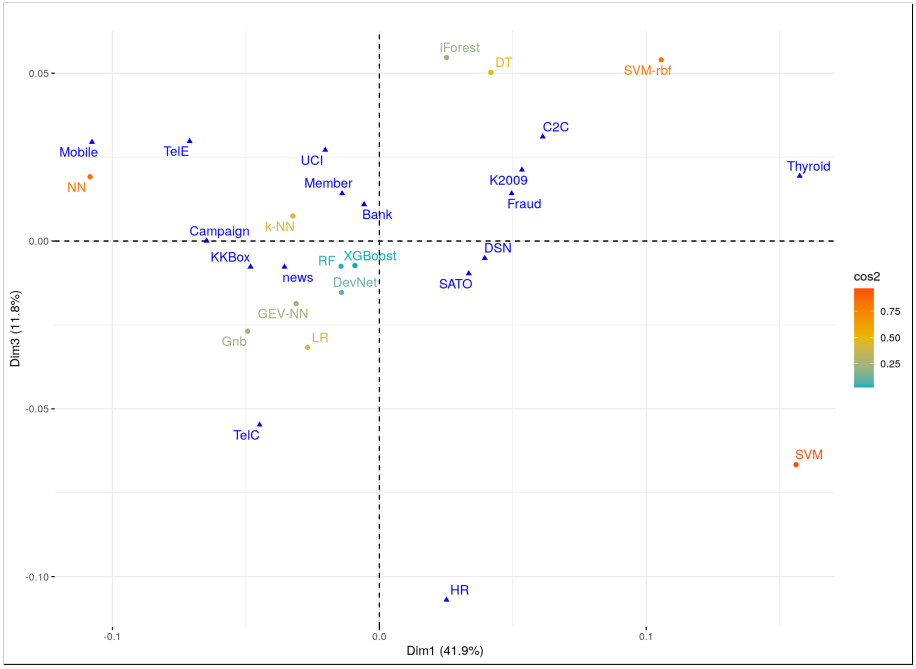
We propose to visualize the machine learning performance similarities and ranking with Critical Difference (CD) diagrams [43] based on statistical pairwise comparisons computed from the AUC



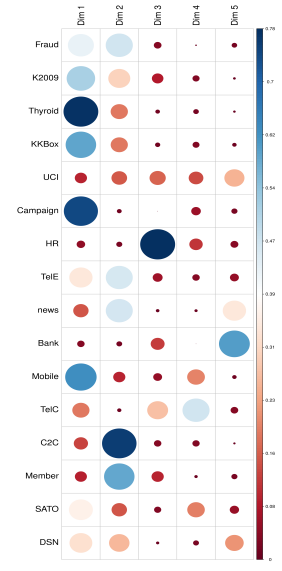
(a) *CA* biplot, dimensions 1 and 2, no sampling



(b) Representation Quality



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



(d) Representation Quality

Fig. 6: (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.

results (Table 2 to Table 6). For these comparisons, we consider the post-hoc Nemenyi test ($\alpha =$ 0.05) for which Figures 3, 4 and 5 provide the

CD diagrams [43] for each sampling strategy. Horizontal lines connect the approaches for which we cannot exclude the hypothesis that the average AUC rank is equal. As can be seen, the sampling strategies have a weak effect on the machine learning approaches ranking.

6.2.2 Models and datasets CA

To go beyond the analyses in Section 6.2.1, we propose to visualize the relationships between the machine learning techniques and the churn-like datasets in a two-dimensional plot based on the AUC results. To this end, we perform a Correspondence Analysis (`CA`) - a geometric approach that extends principal component analysis - on an AUC results table (Table 2). The Figure 6 provides a `CA` result overview that is useful for interpretation.

As can be seen from correlation plots in Figures 6(b) and 6(d), `SVM`, and `NN` are well represented by the first dimension, `RF` and `XGBoost` by the second dimension and `LR` by the third dimension. Similarly, not all datasets are well represented by the two first components and some of them are found on the third and the fourth dimensions. Hence, we provide in Figures 6(a) and 6(c) two CA biplots based either on the two first components, or on the first or third dimensions.

The Figure 6(a) suggests a similar behavior between `RF` and `XGBoost`. It also highlights the difference with these approaches and `SVM` and `SVM-rbf`. *News* appears associated with `RF`, `XGBoost` and `GEV-NN`, in agreement with the AUC Table 2. We also visualize the *Mobile* dataset in the vicinity of `NN` which is the most suitable technique without sampling. Similarly, *TelE* is found near `XGBoost`. The Figure 6(b) uses the third dimension instead of the second dimension, bringing a better representation of `LR`. We notice the positioning of *News* between `RF` and `GEV-NN`, as expected from the AUC table. Interestingly, *SATO* has shifted towards `RF`, `GEV-NN` and `LR`. This is in agreement with Table 2, as these machine learning techniques provide the best top three AUC results. Similarly, *KKBox* stands towards `LR` and `GEV-NN`.

6.2.3 Ensemble study and proposal

In this Section, we combine `LR`, `XGBoost`, `RF` and `NN` for the churn prediction. Specifically, we average predicted probabilities for each instance, over two, three or four methods among `LR`, `XGBoost` `RF` and `NN`. The Figure 7 shows, for each sampling strategy, and over all datasets, the AUC for `LR`, `XGBoost`, `RF` and `NN` (light gray), their pairwise ensembles (light orange), the combination of three methods (dark orange) and the combination of all four methods (dark blue). As can be seen from Figure 7, `LR|XGBoost|RF|NN` ensemble mostly outperforms the other methods, closely followed by `LR|XGBoost|RF` (Table 7). Overall, the best ensemble approach is obtained when combining the three approaches (`LR|XGBoost|RF`) and without sampling strategy (Table 7, $\widehat{AUC} = 0.8577$).

The Table 8 provides for each dataset, the pipeline that produces the highest AUC (*Best non ensemble pipeline AUC* & *Best non ensemble pipeline* columns). Our recommended ensemble pipeline (`LR|XGBoost|RF` and no sampling) provides an AUC that nearly reaches the best AUC result, for almost all datasets. The only exception is for *C2C*. All in all, in practice, we recommend the use of the ensemble `LR|XGBoost|RF` with no sampling for analyzing novel churn-like datasets.

7 Discussion

We compare in this study eleven well-established and popular supervised machine learning techniques used for churn prediction, imbalance dataset or anomaly detection. Our results provide information on the relationships between supervised machine learning methods, imbalanced datasets preprocessing and the datasets. We discuss in this section overall advisable strategies and improvement perspectives.

In this survey, we only consider the default parameters for each approach. However, the supervised context would also allow for boosting versions of some of these techniques. This could significantly improve their classification results, in particular for `SVM` [131]. The boosting strategy has been successfully applied to the prediction of customer churn in retail [36] and telecom companies [84]. Generally speaking, ensemble approach should be considered for the classification task in a churn-like context, as they repeatedly performed better than individual classifiers in the field of data mining. Ahmed *et al.* [3] even proposed nested ensemble learners models that outperform traditional ensemble when applied to churn prediction in telecom industry.

Fig. 7: AUC ensemble results on the three top machine learning approaches and all datasets

Table 7: $\widetilde{AUC}$ for *ensemble* and *non ensemble* approaches and all datasets.

Sampling	no	SMOTE	ADASYN	NCR	Tomek Links	SMOTE & R.U.	SMOTE & T.L.	SMOTE & NCR	$\overline{\widetilde{AUC}}$
LR	0.8283	0.8301	0.8293	0.8274	0.8287	0.8301	0.8302	0.8274	0.8294
XGBoost	0.8104	0.8087	0.8102	0.8292	0.8135	0.8087	0.8117	0.8290	0.8167
RF	0.8358	0.8137	0.8021	0.8187	0.8066	0.8035	0.8023	0.8403	0.8162
NN	0.7700	0.8425	0.7998	0.7658	0.7617	0.8417	0.8428	0.8409	0.8159
LR|XGBoost	0.8479	0.8464	0.8465	0.8457	0.8485	0.8464	0.8466	0.8395	0.8464
LR|RF	0.8516	0.8439	0.8460	0.8457	0.8476	0.8467	0.8466	0.8470	0.8472
LR|NN	0.8383	0.8442	0.8408	0.8378	0.8403	0.8446	0.8449	0.8424	0.8418
XGBoost|RF	0.8325	0.8256	0.8251	0.8374	0.8267	0.8240	0.8255	0.8405	0.8313
XGBoost|NN	0.8388	0.8461	0.8450	0.8412	0.8388	0.8484	0.8448	0.8352	0.8431
RF|NN	0.8533	0.8409	0.8365	0.8411	0.8358	0.8375	0.8395	0.8449	0.8423
LR|XGBoost|RF	**0.8577**	0.8526	0.8489	**0.8500**	0.8529	0.8521	0.8489	0.8466	0.8517
LR|XGBoost|NN	0.8498	0.8457	0.8477	0.8459	0.8452	0.8478	0.8465	0.8462	0.8473
LR|RF|NN	0.8523	0.8462	0.8484	0.8472	0.8491	0.8485	0.8470	0.8479	0.8483
XGBoost|NN|RF	0.8566	0.8512	0.8463	0.8486	0.8533	0.8510	0.8464	0.8464	0.8501
LR|XGBoost|RF|NN	0.8562	**0.8533**	**0.8506**	0.8491	**0.8546**	**0.8537**	**0.8492**	**0.8513**	**0.8526**

Table 8: Our ensemble proposal vs. best non ensemble approach for each dataset.

	LR|XGBoost|RF & no sampling *AUC*	Best *non ensemble* pipeline AUC	Best *non ensemble* pipeline
Fraud	0.9794	0.9766	no sampling & LR
K2009	0.5197	0.5153	SMOTE-NCR & LR
Thyroid	0.9989	0.9996	no sampling & RF
KKBox	0.6890	0.7054	no sampling & GEV-NN
UCI	0.9215	0.9200	NCR & XGBoost
Campaign	0.9440	0.9402	SMOTE-NCR & RF
HR	0.8443	0.8596	no sampling & LR
TelE	0.9435	0.9421	SMOTE & XGBoost
News	0.8636	0.8615	no sampling & RF
Bank	0.8531	0.8583	no sampling & GEV-NN
Mobile	0.8761	0.9124	ADASYN & NN
TelC	0.8340	0.8459	Tomek Links & LR
C2C	0.3852	0.5659	NCR & SVM
Member	0.6201	0.6354	SMOTE-NCR & NN
SATO	0.7765	0.7882	no sampling & RF
DSN	0.8623	0.8694	SMOTE-T.Links & XGBoost
$\widetilde{AUC}$	0.8069	0.8240	

The finance industry has gradually adapted various machine learning techniques. In particular, detecting economic crimes (eg., accounting fraud, money laundering) triggered successful applications of machine learning. `LR`, `Gnb` and `SVM` are among the most classic methods exploited in this

area. The emergence of new kinds of fraud with the growth of electronic market has also popularized deep learning methods in finance. Ensemble strategies and boosting also remain a valuable option in this area. An enhanced hybrid ensemble approach, named *RS-MultiBoosting* [156] has been proposed; it incorporates *random subspace* and *MultiBoosting* to improve the accuracy of forecasting credit risk.

As already mentioned in this study the existence of small disjuncts within the minority class – corresponding in the churn context to the customer profile heterogeneity – can significantly impede the classifier performance. Hence, it would be advisable to segment the minority class upstream of or during the model training phase. The *Logit Leaf Model* [41] (LLM) is a successful example of this strategy; it is an hybrid classification algorithm that combines `DT` and `RF` over a dataset whose partitioning is in agreement with the heterogeneity between customers. Hence, LLM is an ensemble approach that takes into account specific group characteristics that remained unknown when a single classifier is trained over the whole dataset.

Most of the churn-like prediction frameworks consider traditional structured data. However, as a large proportion of big data consists of diverse unstructured data [53], 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 [12, 39, 125]. Recently, De Caigny *et al.* [42] proposed the incorporation of textual information based on Convolutional Neural Network.

If the advantage of supervised learning is that all input labels are typically meaningful and serve as basis for an explainable discriminative classifier, the need for labels collection is however by itself a strong limitation. First of all, when the volume of the data is too large, it becomes prohibitively expensive to collect all labels. Furthermore, when distinctive labels are hard to find, it implies noise or uncertainties in the supervision which can lead to inaccurate results [27, 122]. In addition, in an imbalanced or strongly imbalanced classes distribution context, accessing high quality labels for the minority class is generally challenging. Indeed, the existence of different instance profiles within the positive class strongly impedes the training phase [122].

Unsupervised or semi-supervised learning can be used to overcome these issues. While unsupervised learning requires no class label, semi-supervised learning only requires a small number of labeled samples. A key idea is to learn a model for the class associated with the normal behavior and then use this model to identify abnormal behaviors [30]. Hence, semi-supervised or unsupervised approaches can handle, during the test phase, abnormal behaviors that did not appear in the training dataset. This is a clear advantage as compared to supervised learning strategy.

Deep learning techniques can be of great help to learn efficient model with few or none abnormal instances label [105]. Indeed, deep learning provides novel representations of the data which in turn can be used to identify minority class samples. However, whenever the representation learning is independent from the classification task, deep representations might be suboptimal or even irrelevant. Recently, efforts have been made to incorporate the identification of the abnormal instances within the representation learning phase to improve their expressiveness.

8 Conclusion

This technical survey aims to review, evaluate and compare several popular machine learning approaches in the context of churn prediction. It also offers original analyses and visualizations, and ultimately provides a general recommendation on a churn prediction pipeline based on an ensemble approach.

In our proposal, we included a background of the churn analysis research, an introduction to widespread data sampling and classifier approaches and a presentation of advisable evaluation metrics and strategies. First, we described publicly available churn-like datasets covered in this study and provide links for an easy access. Then, we introduced data sampling approaches, which unfold in three categories, namely oversampling, undersampling and hybrid. We also detailed several machine learning classifiers encountered

in the churn research field and discussed their reported success in the literature. The validation strategies and metrics are then discussed. Finally, machine learning approaches are combined and evaluated on sixteen publicly available churn-like datasets. We summarized our results in terms of AUC score.

Ultimately, we proposed effective visualizations shading light on behavioral relationships between classifiers/sampling methods and their association with churn-like benchmark datasets. Most importantly, we presented a general churn analysis pipeline based on a straightforward ensemble technique that can be successfully used in practice. Hence, this technical survey provides a good reference to users interested in machine learning choices in the context of churn prediction.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

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 [62]. #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 [34]. #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* [131]. #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 [92]. #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 [24, 39, 40]. #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 [78]. #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 [2]. #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.

Fraud (*Credit Card Fraud Detection*) The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. It is an anomaly detection dataset.

Thyroid (*Thyroid Disease*) This data are from the Garavan Institute. The problem is to determine whether a patient referred to the clinic is hypothyroid. 92 percent of the patients are not hyperthyroid in this dataset which contains 7,200 instances. It is an anomaly detection dataset.

Campaign (*Bank Marketing*) The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be ('yes') or not ('no') subscribed. It is an anomaly detection dataset.

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 B1 summarizes information on packages, functions and parameters used for our experiments. It also provides links to the online description of each function.

Table B1: 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
	Feed Forward Neural Network	`NN`	*default*	–	Neural-Network-Churn-Prediction
	Generalize Extreme Value-NN	`GEV-NN`	*default*	–	GEV-NN
Semi-supervised	Isolation Forest	`IsolationForest`	*default*	0.23.2	sklearn.ensemble.IsolationForest
	Deep AD with Deviation Networks	`DevNet`	*default*	–	deviation-network
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

References

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

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

[3] Ahmed M, Siddiqi I, Afzal H, et al (2018) MCS: Multiple classifier system to predict the churners in the telecom industry. 2017 Intelligent Systems Conference, IntelliSys 2017 2018-January(September):678–683. https://doi.org/10.1109/IntelliSys.2017.8324367

[4] Akbani R, Kwek S, Japkowicz N (2004) Applying support vector machines to imbalanced datasets. In: European conference on machine learning, Springer, pp 39–50

[5] Alam S, Sonbhadra SK, Agarwal S, et al (2020) One-class support vector classifiers: A survey. Knowl Based Syst 196:105,754

[6] Amnueypornsakul B, Bhat S, Chinprutthiwong P (2015) Predicting Attrition Along the Way: The UIUC Model. Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs October:55–59. https://doi.org/10.3115/v1/w14-4110

[7] Anderson EW, Sullivan MW (1993) The antecedents and consequences of customer satisfaction for firms. Marketing science 12(2):125–143

[8] Batista GE, Bazzan AL, Monard MC, et al (2003) Balancing training data for automated annotation of keywords: a case study. In: WOB, pp 10–18

[9] Batista GE, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD explorations newsletter 6(1):20–29

[10] Batuwita R, Palade V (2010) Efficient resampling methods for training support vector machines with imbalanced datasets. In: The 2010 International Joint Conference on Neural Networks (IJCNN), IEEE, pp 1–8

[11] Benczúr AA, Csalogány K, Lukács L, et al (2007) Semi-supervised learning: A comparative study for web spam and telephone user churn. In: In Graph Labeling Workshop in conjunction with ECML/PKDD, Citeseer

[12] Benoit DF, Van den Poel D (2012) Improving customer retention in financial services using kinship network information. Expert Systems with Applications 39(13):11,435–11,442

[13] Bermejo P, Gámez JA, Puerta JM (2011) Improving the performance of naive bayes multinomial in e-mail foldering by introducing distribution-based balance of datasets. Expert Systems with Applications 38(3):2072–2080

[14] Bhattacharya C (1998) When customers are members: Customer retention in paid membership contexts. Journal of the academy of marketing science 26(1):31–44

[15] Błaszczyński J, Stefanowski J (2018) Local data characteristics in learning classifiers from imbalanced data. In: Advances in Data Analysis with Computational Intelligence Methods. Springer, p 51–85

[16] Bolton RN (1998) A dynamic model of the duration of the customer's relationship with a continuous service provider: The role of satisfaction. Marketing science 17(1):45–65

[17] Bolton RN, Bronkhorst TM (1995) The relationship between customer complaints to the firm and subsequent exit behavior. ACR North American Advances 22:94–100

[18] Branco P, Torgo L, Ribeiro RP (2016) A survey of predictive modeling on imbalanced

domains. ACM Computing Surveys 49(2):1–50. https://doi.org/10.1145/2907070

[19] Breiman L (1996) Bagging predictors. Machine learning 24(2):123–140

[20] Breiman L (2001) Random forests. Machine learning 45(1):5–32

[21] Breiman L, Spector P (1992) Submodel selection and evaluation in regression. the x-random case. International statistical review/revue internationale de Statistique 60(3):291–319

[22] Breiman L, Friedman JH, Olshen RA, et al (1984) Classification and regression trees, belmont, california: Wadsworth

[23] Breunig MM, Kriegel HP, Ng RT, et al (2000) Lof: Identifying density-based local outliers. SIGMOD Rec 29(2):93–104. https://doi.org/10.1145/335191.335388

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

[25] Burman P (1989) A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. Biometrika 76(3):503–514. URL http://www.jstor.org/stable/2336116

[26] Burrus CS, Barreto J, Selesnick IW (1994) Iterative reweighted least-squares design of fir filters. IEEE Transactions on Signal Processing 42(11):2926–2936

[27] Cabral GG, Oliveira A (2014) One-class classification for heart disease diagnosis. 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC) pp 2551–2556

[28] Castanedo F, Valverde G, Zaratiegui J, et al (2014) Using deep learning to predict customer churn in a mobile telecommunication network

[29] Cervantes J, Garcia-Lamont F, Rodríguez-Mazahua L, et al (2020) A comprehensive survey on support vector machine classification: Applications, challenges and trends. Neurocomputing 408:189–215

[30] Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: A survey. ACM Comput Surv 41(3). https://doi.org/10.1145/1541880.1541882

[31] Chawla NV, Bowyer KW, Hall LO, et al (2002) Smote: synthetic minority oversampling technique. Journal of artificial intelligence research 16:321–357

[32] Chen C, Liaw A, Breiman L, et al (2004) Using random forest to learn imbalanced data. University of California, Berkeley 110(1-12):24

[33] Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, ACM, pp 785–794

[34] Chen Y, Xie X, Lin SD, et al (2018) Wsdm cup 2018: Music recommendation and churn prediction. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, ACM, pp 8–9

[35] Chowdhury A, Alspector J (2003) Data duplication: an imbalance problem? In: ICML'2003 Workshop on Learning from Imbalanced Data Sets (II), Washington, DC

[36] Clemente M, Giner-Bosch V, San Matías S (2010) 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

[37] Cooray K (2010) Generalized gumbel distribution. Journal of Applied Statistics 37(1):171–179

[38] Coussement K, De Bock KW (2013) Customer churn prediction in the online gambling industry: The beneficial effect of

ensemble learning. Journal of Business Research 66(9):1629–1636

[39] Coussement K, Van den Poel D (2008) Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. Expert systems with applications 34(1):313–327

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

[41] De Caigny A, Coussement K, De Bock KW (2018) A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees. European Journal of Operational Research 269(2):760–772. https://doi.org/https://doi.org/10.1016/j.ejor.2018.02.009

[42] De Caigny A, Coussement K, De Bock KW, et al (2020) Incorporating textual information in customer churn prediction models based on a convolutional neural network. International Journal of Forecasting 36(4):1563–1578. https://doi.org/https://doi.org/10.1016/j.ijforecast.2019.03.029

[43] Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 7:1–30

[44] Denil M, Trappenberg T (2010) Overlap versus imbalance. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 6085 LNAI:220–231. https://doi.org/10.1007/978-3-642-13059-5_22

[45] Deville JC, Tillé Y (2004) Efficient balanced sampling: the cube method. Biometrika 91(4):893–912

[46] Dingli A, Marmara V, Fournier NS (2017) Comparison of deep learning algorithms to predict customer churn within a local retail industry. International journal of machine learning and computing 7(5):128–132

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

[48] Drummond C, Holte RC, et al (2003) C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: Workshop on learning from imbalanced datasets II, Citeseer, pp 1–8

[49] Dubey H, Pudi V (2013) Class based weighted K-Nearest neighbor over imbalance dataset. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 7819 LNAI(PART 2):305–316. https://doi.org/10.1007/978-3-642-37456-2_26

[50] Effendy V, Baizal ZA, et al (2014) 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, pp 325–330

[51] Fernández A, García S, Herrera F, et al (2018) SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. Journal of Artificial Intelligence Research 61:863–905. https://doi.org/10.1613/jair.1.11192

[52] Friedman J, Hastie T, Tibshirani R (2001) The elements of statistical learning, vol 1. Springer series in statistics New York

[53] Gandomi A, Haider M (2015) Beyond the hype: Big data concepts, methods, and analytics. International journal of information management 35(2):137–144

[54] Ganesan S (1994) Determinants of long-term orientation in buyer-seller relationships. Journal of marketing 58(2):1–19

[55] García DL, Nebot À, Vellido A (2017) Intelligent data analysis approaches to churn as a business problem: a survey. Knowledge and Information Systems 51(3):719–774

[56] García V, Mollineda RA, Sánchez JS (2008) On the k-nn performance in a challenging scenario of imbalance and overlapping. Pattern Analysis and Applications 11(3):269–280

[57] García V, Sánchez JS, Mollineda RA (2012) On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. Knowledge-Based Systems 25(1):13–21

[58] Gregory B (2018) Predicting customer churn: Extreme gradient boosting with temporal data. arXiv preprint arXiv:180203396

[59] Günther CC, Tvete IF, Aas K, et al (2014) Modelling and predicting customer churn from an insurance company. Scandinavian Actuarial Journal 2014(1):58–71

[60] Gupta S, Lehmann DR, Stuart JA (2004) Valuing customers. Journal of marketing research 41(1):7–18

[61] Guyon I, Gunn S, Nikravesh M, et al (2008) Feature extraction: foundations and applications, vol 207. Springer

[62] Guyon I, Lemaire V, Boullé M, et al (2009) 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, pp 1–22

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

[64] Haixiang G, Yijing L, Shang J, et al (2017) Learning from class-imbalanced data: Review of methods and applications. Expert Systems with Applications 73:220–239. https://doi.org/10.1016/j.eswa.2016.12.035

[65] Han H, Wang WY, Mao BH (2005) Borderline-smote: a new over-sampling method in imbalanced data sets learning. In: International conference on intelligent computing, Springer, pp 878–887

[66] Hand DJ, Yu K (2001) Idiot's Bayes - Not so stupid after all? International Statistical Review 69(3):385–398. https://doi.org/10.1111/j.1751-5823.2001.tb00465.x

[67] Hart P (1968) The condensed nearest neighbor rule (corresp.). IEEE transactions on information theory 14(3):515–516

[68] He H, Ma Y (2013) Imbalanced learning: foundations, algorithms, and applications. John Wiley & Sons

[69] He, H., Bai, Y., Garcia, E., & Li S (2008) 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):1322– 1328

[70] Hitt LM, Frei FX (2002) Do better customers utilize electronic distribution channels? the case of pc banking. Management Science 48(6):732–748

[71] Holte RC, Acker L, Porter BW, et al (1989) Concept learning and the problem of small disjuncts. In: IJCAI, Citeseer, pp 813–818

[72] Hosein P, Sewdhan G, Jailal A (2021) Softchurn: 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, pp 1–6

[73] Huang B, Kechadi MT, Buckley B (2012) Customer churn prediction in telecommunications. Expert Systems with Applications 39(1):1414–1425. https://doi.org/10.1016/j.eswa.2011.08.024

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

[75] John GH, Langley P (1995) Estimating continuous distributions in bayesian classifiers. In: Proceedings of the Eleventh conference on Uncertainty in artificial intelligence, Morgan Kaufmann Publishers Inc., pp 338–345

[76] Kamaruddin S, Ravi V (2016) Credit card fraud detection using big data analytics: Use of psoaann based one-class classification. In: Proceedings of the International Conference on Informatics and Analytics. Association for Computing Machinery, New York, NY, USA, ICIA-16, https://doi.org/10.1145/2980258.2980319

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

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

[79] King G, Zeng L (2001) Logistic regression in rare events data. Political analysis 9(2):137–163

[80] Kohavi R, et al (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Ijcai, Montreal, Canada, pp 1137–1145

[81] Kong J, Kowalczyk W, Menzel S, et al (2020) Improving imbalanced classification by anomaly detection. In: Bäck T, Preuss M, Deutz A, et al (eds) Parallel Problem Solving from Nature – PPSN XVI. Springer International Publishing, Cham, pp 512–523

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

[83] Laurikkala J (2001) Improving identification of difficult small classes by balancing class distribution. In: Conference on Artificial Intelligence in Medicine in Europe, Springer, pp 63–66

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

[85] Leung CK, Pazdor AG, Souza J (2021) Explainable artificial intelligence for data science on customer churn. In: 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), IEEE, pp 1–10

[86] Li W, Gao M, Li H, et al (2016) Dropout prediction in MOOCs using behavior features and multi-view semi-supervised learning. Proceedings of the International Joint Conference on Neural Networks 2016-Octob:3130–3137. https://doi.org/10.1109/IJCNN.2016.7727598

[87] Ling CX, Li C (1998) Data mining for direct marketing: Problems and solutions. In: Kdd, pp 73–79

[88] Liu FT, Ting KM, Zhou ZH (2012) Isolation-based anomaly detection. ACM Trans Knowl Discov Data 6(1). https://doi.org/10.1145/2133360.2133363

[89] López V, Fernández A, Moreno-Torres JG, et al (2012) Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. Expert Systems with Applications 39(7):6585–6608

[90] López V, Fernández A, García S, et al (2013) An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Information sciences 250:113–141

[91] Maxham III JG (2001) Service recovery's influence on consumer satisfaction, positive word-of-mouth, and purchase intentions. Journal of business research 54(1):11–24

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

[93] Mittal B, Lassar WM (1998) Why do customers switch? the dynamics of satisfaction versus loyalty. Journal of services marketing 12(3):177–194

[94] Mittal V, Kamakura WA (2001) Satisfaction, repurchase intent, and repurchase behavior: Investigating the moderating effect of customer characteristics. Journal of marketing research 38(1):131–142

[95] Mozer MC, Wolniewicz R, Grimes DB, et al (2000) Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. IEEE Transactions on neural networks 11(3):690–696

[96] Munkhdalai L, Munkhdalai T, Park KH, et al (2019) An end-to-end adaptive input selection with dynamic weights for forecasting multivariate time series. IEEE Access 7:99,099–99,114

[97] Munkhdalai L, Munkhdalai T, Ryu KH (2020) Gev-nn: A deep neural network architecture for class imbalance problem in binary classification. Knowledge-Based Systems 194:105,534

[98] Napierała K, Stefanowski J, Wilk S (2010) Learning from imbalanced data in presence of noisy and borderline examples. In: International Conference on Rough Sets and Current Trends in Computing, Springer, pp 158–167

[99] Neslin SA, Gupta S, Kamakura W, et al (2006) Defection detection: Measuring and understanding the predictive accuracy of customer churn models. Journal of marketing research 43(2):204–211

[100] Nguyen HM, Cooper EW, Kamei K (2011) Borderline over-sampling for imbalanced data classification. International Journal of Knowledge Engineering and Soft Data Paradigms 3(1):4–21

[101] Nguyen N, LeBlanc G (1998) The mediating role of corporate image on customers' retention decisions: an investigation in financial services. International journal of bank marketing 16(2):52–65

[102] Owen AB (2007) Infinitely imbalanced logistic regression. Journal of Machine Learning Research 8(Apr):761–773

[103] Pang G, Xu H, Cao L, et al (2017) 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, pp 807–816

[104] Pang G, Shen C, van den Hengel A (2019) Deep anomaly detection with deviation networks. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp 353–362

[105] Pang G, Shen C, Cao L, et al (2021) Deep learning for anomaly detection: A review. ACM Comput Surv 54(2). https://doi.org/10.1145/3439950

[106] Paulin M, Perrien J, Ferguson RJ, et al (1998) Relational norms and client retention: external effectiveness of commercial banking in canada and mexico. International Journal of Bank Marketing 16(1):24–31

[107] Van den Poel D, Lariviere B (2004) Customer attrition analysis for financial services using proportional hazard models. European journal of operational research 157(1):196–217

[108] Reichheld FF, Sasser WE (1990) Zero defections: Quality comes to services. Harvard business review 68(5):105–111

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

[110] Rennie JD (2001) Improving multi-class text classification with naive bayes. Technical Report AITR 4

[111] Risselada H, Verhoef PC, Bijmolt TH (2010) Staying power of churn prediction models. Journal of Interactive Marketing 24(3):198–208

[112] Ruff L, Kauffmann JR, Vandermeulen RA, et al (2021) A unifying review of deep and shallow anomaly detection. Proceedings of the IEEE

[113] Ruisen L, Songyi D, Chen W, et al (2018) Bagging of xgboost classifiers with random under-sampling and tomek link for noisy label-imbalanced data. In: IOP Conference Series: Materials Science and Engineering, IOP Publishing, p 012004

[114] Salas-Eljatib C, Fuentes-Ramirez A, Gregoire TG, et al (2018) A study on the effects of unbalanced data when fitting logistic regression models in ecology. Ecological Indicators 85:502–508

[115] Saradhi VV, Palshikar GK (2011) Employee churn prediction. Expert Systems with Applications 38(3):1999–2006

[116] Schölkopf B, Williamson R, Smola A, et al (1999) Support vector method for novelty detection. MIT Press, Cambridge, MA, USA, NIPS'99, p 582–588

[117] Seiffert C, Khoshgoftaar TM, Van Hulse J, et al (2014) An empirical study of the classification performance of learners on imbalanced and noisy software quality data. Information Sciences 259:571–595

[118] Seymen OF, Dogan O, Hiziroglu A (2020) Customer churn prediction using deep learning. In: International Conference on Soft Computing and Pattern Recognition, Springer, pp 520–529

[119] Siber R (1997) 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):77–81

[120] Śniegula A, Poniszewska-Marańda A, Popović M (2019) 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, pp 640–644

[121] Stefanowski J (2016) Dealing with data difficulty factors while learning from imbalanced data. In: Challenges in computational statistics and data mining. Springer, p 333–363

[122] Taha A, Hadi AS (2019) Anomaly detection methods for categorical data: A review. ACM Comput Surv 52(2). https://doi.org/10.1145/3312739

[123] Tan F, Wei Z, He J, et al (2018) A Blended Deep Learning Approach for Predicting User Intended Actions. Proceedings - IEEE International Conference on Data Mining, ICDM 2018-Novem:487–496. https://doi.org/10.1109/ICDM.2018.00064

[124] Tan S (2005) Neighbor-weighted k-nearest neighbor for unbalanced text corpus. Expert Systems with Applications 28(4):667–671

[125] Tang L, Thomas L, Fletcher M, et al (2014) Assessing the impact of derived behavior information on customer attrition in the financial service industry. European Journal of Operational Research 236(2):624–633

[126] Tax DMJ, Duin RPW (1999) Support vector domain description. Pattern Recogn Lett 20(11–13):1191–1199. https://doi.org/10.1016/S0167-8655(99)00087-2

[127] Tian J, Gu H, Liu W (2011) Imbalanced classification using support vector machine ensemble. Neural computing and applications 20(2):203–209

[128] Tomek I (1976) Tomek Link: Two Modifications of CNN. IEEE Trans Systems, Man and Cybernetics SMC-6:769–772. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp={&}arnumber=4309452

[129] Umayaparvathi V, Iyakutti K (2016) A Survey on Customer Churn Prediction in Telecom Industry: Datasets, Methods and Metrics. International Research Journal of Engineering and Technology 3:2395–56

[130] Umayaparvathi V, Iyakutti K (2017) Automated feature selection and churn prediction using deep learning models. International Research Journal of Engineering and Technology (IRJET) 4(3):1846–1854

[131] Vafeiadis T, Diamantaras KI, Sarigiannidis G, et al (2015) A comparison of machine learning techniques for customer churn prediction. Simulation Modelling Practice and Theory 55:1–9

[132] Van Hulse J, Khoshgoftaar TM, Napolitano A, et al (2009) Feature selection with high-dimensional imbalanced data. In: 2009 IEEE International Conference on Data Mining Workshops, IEEE, pp 507–514

[133] Vapnik V (1998) Statistical learning theory wiley-interscience. New York

[134] Varki S, Colgate M (2001) The role of price perceptions in an integrated model of behavioral intentions. Journal of Service Research 3(3):232–240

[135] Villa-Pérez ME, Álvarez-Carmona MÁ, Loyola-González O, et al (2021) Semi-supervised anomaly detection algorithms: A comparative summary and future research directions. Knowledge-Based Systems p 106878. https://doi.org/https://doi.org/10.1016/j.knosys.2021.106878

[136] Wang S, Li D, Song X, et al (2011) A feature selection method based on improved fisher's discriminant ratio for text sentiment classification. Expert Systems with Applications 38(7):8696–8702

[137] Wang S, Liu W, Wu J, et al (2016) Training deep neural networks on imbalanced data sets. In: 2016 international joint conference on neural networks (IJCNN), IEEE, pp 4368–4374

[138] Wang W, Yu H, Miao C (2017) Deep model for dropout prediction in MOOCs. ACM International Conference Proceeding Series Part F1306:26–32. https://doi.org/10.1145/3126973.3126990

[139] Weiss GM (2004) Mining with rarity: a unifying framework. ACM Sigkdd Explorations Newsletter 6(1):7–19

[140] Weiss GM (2010) The impact of small disjuncts on classifier learning. In: Data Mining, Springer, pp 193–226

[141] Weiss GM, Hirsh H (2000) A quantitative study of small disjuncts. AAAI/IAAI 2000:665–670

[142] Weiss GM, Provost F (2003) Learning when training data are costly: The effect of class distribution on tree induction. Journal of artificial intelligence research 19:315–354

[143] Wilson DL (1972) Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man, and Cybernetics SMC-2(3):408–421

[144] Xiao J, Huang L, Xie L (2018) Cost-sensitive semi-supervised ensemble model for customer churn prediction. In: 2018 15th International Conference on Service Systems and Service Management (ICSSSM), IEEE, pp 1–6

[145] Xiao Y, Wang H, Xu W, et al (2016) Robust one-class svm for fault detection. Chemometrics and Intelligent Laboratory Systems 151:15–25. https://doi.org/https://doi.org/10.1016/j.chemolab.2015.11.010

[146] Xie Y, Li X (2008) Churn prediction with linear discriminant boosting algorithm. In: 2008 International Conference on Machine Learning and Cybernetics, IEEE, pp 228–233

[147] Yang C, Shi X, Jie L, et al (2018) I know you'll be back: Interpretable new user clustering and churn prediction on a mobile social application. In: Proceedings of the

24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 914–922

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

[149] Yin L, Ge Y, Xiao K, et al (2013) Feature selection for high-dimensional imbalanced data. Neurocomputing 105:3–11. https://doi.org/10.1016/j.neucom.2012.04.039

[150] Zadrozny B, Elkan C (2001) 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, pp 204–213

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

[152] Zeithaml VA, Berry LL, Parasuraman A (1996) The behavioral consequences of service quality. Journal of marketing 60(2):31–46

[153] Zhao Z, Peng H, Lan C, et al (2018) Imbalance learning for the prediction of n 6-methylation sites in mrnas. BMC genomics 19(1):574

[154] Zhou F, Yang S, Fujita H, et al (2020) Deep learning fault diagnosis method based on global optimization gan for unbalanced data. Knowledge-Based Systems 187:104,837

[155] Zhou ZH, Liu XY (2005) Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Transactions on knowledge and data engineering 18(1):63–77

[156] Zhu Y, Zhou L, Xie C, et al (2019) Forecasting smes' credit risk in supply chain finance with an enhanced hybrid ensemble machine learning approach. International Journal of Production Economics 211:22–33. https://doi.org/https://doi.org/10.1016/j.ijpe.2019.01.032

[157] Zong B, Song Q, Min MR, et al (2018) Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International conference on learning representations