Expert Systems With Applications xxx (xxxx) xxx

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Lessons learned from data stream classification applied to credit scoring

Jean Paul Barddal [a,*], Lucas Loezer [a], Fabrício Enembreck [a], Riccardo Lanzuolo [b]

[a] PPGIa, Pontifícia Universidade Católica do Paraná, Curitiba, Brazil
[b] 4KST, Curitiba, Brazil

ARTICLE INFO

Keywords:
Credit scoring
Machine learning datasets
Data stream classification

ABSTRACT

The financial credibility of a person is a factor used to determine whether a loan should be approved or not, and this is quantified by a 'credit score,' which is calculated using a variety of factors, including past performance on debt obligations, profiling, amongst others. Machine learning has been widely applied to automate the development of effective credit scoring models over the years. Yet, studies show that the development of robust credit scoring models may take longer than a year, and thus, if the behavior of customers changes over time, the model will be outdated even before its deployment. In this paper, we made 3 anonymized real-world credit scoring datasets available alongside the results obtained. In each of these datasets, we verify whether the credit scoring task should be thought as an ephemeral scenario since many of the variables may drift over time, and thus, data stream mining techniques should be used since they were tailored for incremental learning and to detect and adapt to changes in the data distribution. Therefore, we compare both traditional batch machine learning algorithms with data stream algorithms in different validation schemes using both Kolmogorov–Smirnov and Population Stability Index metrics. Furthermore, we also provide insights on the importance of features according to their Information Value, Mean Decrease Impurity, and Mean Positional Gain metrics, such that the last depicts changes in the importance of features over time. For 2 of the 3 tested datasets, the results obtained by data stream learners are comparable to predictive models currently in use, thus showing the efficiency of data stream classification for the credit scoring task.

1. Introduction

The financial credibility of a person is a factor used to determine whether a loan should be approved or not. Nowadays, the financial credibility is quantified by a 'credit score', which is calculated using factors that include a person's information on past performance on debt obligations, profiling, main household, income, occupation, demographics, possessions (e.g., cars, and other residences, if any), and census information. Naturally, the development and management of effective and reliable risk assessment credit scoring models are time-consuming, and over decades, multiple automatic credit scoring models have been created using machine learning techniques to mitigate this issue and possible biases that may be introduced into models by credit risk managers. Even with the help of machine learning, application papers such as (Crook et al., 1992; Hand & Adams, 2014) show that the development of a robust credit scoring model can range from 3 to 18 months. As a result, it is not rare for financial institutions and credit scoring operators to use the same credit scoring model for years without changes. As brought up by authors in Žliobaitė et al. (2016), if a model is built on top of 2 or more years of historical data, while shifted 3 years away from the point they will be used, an 5-year shift is often exceeded. If we assume that the world is nearly stationary, then the average accuracy of the credit scoring model would remain unchanged, yet, this condition is often untrue.

In this paper, our claim is that data stream classifiers are useful in credit scoring tasks. More specifically, our motivation is to verify whether the credit scoring task should be thought as an ephemeral scenario since many of the variables as mentioned above may drift over time. Consequently, incremental and adaptive models are expected to help in the process of mitigating the unnecessary time of re-creating credit scoring models over time. Therefore, we make 3 real-world datasets obtained from Brazilian financial institutions and credit scoring operators publicly available after anonymization. Using these datasets, our main contribution is to compare and analyze traditional batch machine learning algorithms with data stream algorithms in both holdout and monthly test-then-train validation schemes using industry and academic accepted evaluation metrics. We also provide insights about feature importance according to Information Value (IV) (Mays & Mays, 2004), Mean Decrease Impurity (MDI) (Louppe et al., 2013),

* Corresponding author.
E-mail addresses: jean.barddal@ppgia.pucpr.br (J.P. Barddal), loezerl@ppgia.pucpr.br (L. Loezer), fabricio@ppgia.pucpr.br (F. Enembreck), riccardo@4kst.com (R. Lanzuolo).

https://doi.org/10.1016/j.eswa.2020.113899
Received 11 March 2020; Received in revised form 26 July 2020; Accepted 16 August 2020
Available online 20 August 2020

and Mean Positional Gain (MPG) (Karax et al., 2019) metrics, such that the last allows the identification of drifting features over time.

This paper is divided as follows. We introduce the credit scoring task from a machine learning point of view in Section 2. Next, we discuss related works that also propose techniques for automatic credit scoring using machine learning in Section 3. We then bring forward in Section 4 the experimental setup used in our analysis, including learning algorithms, datasets used, evaluation metrics, and validation schemes. The results are reported and assessed in Section 5, whereas Section 6 concludes this paper.

2. Credit scoring and machine learning

The scoring process is an important part of the credit risk management system used in financial institutions to predict the risk of loan applications. This process often relies on statistical models that take into account information from the application and also from the customer and estimates the defaulting probability. Over time, standard approaches such as *scorecards* were replaced or combined with automated approaches using machine learning models since both the dimension and size of historical data are ever-growing (Jung et al., 2015; Kennedy et al., 2013).

The development of an assertive scoring model based on machine learning depends on several factors, including (i) the gathering of veracious historical data, (ii) the identification of key attributes from the customer and his/hers past loans, and the (iii) proper construction and validation of the predictive model. In each of the steps mentioned above, the interaction between machine learning engineers, data analysts, and risk analysts is necessary as each contributes to the development and assessment of credit scoring models. In practice, each of these stakeholders may impose constraints on which kind of data can be used, often to enable audits and prevent issues, e.g., discriminatory behavior.

In this paper, we denote $(\vec{x}, y)$ to be a loan request, where $\vec{x}$ a vector of characteristics (features) detailing the loan request and the customer requesting it, and $y \in \{0, 1\}$ the target feature, such that 1 represents that the customer paid this specific loan in full, and 0 that the customer defaulted. The classification task targets the creation of a predictive model $h : \vec{x} \rightarrow y$ that accurately maps features and their values into classes. Naturally, different types of classification systems exist, including, for instance, decision trees, linear regression models, and ensembles; and these are discussed in Section 4.1.

Instead of providing a boolean answer determining whether a customer loan request is expected to be fully paid or not, financial institutions and credit operators often work with scores. In Brazil, credit scores are bounded in the $[0; 1,000]$ interval, where 0 is the value that represents a customer that is undoubtedly going to default, while 1000 represents a customer that will surely pay his debts in full. In practice, these extreme values are nearly inexistent in real-world applications, so institutions should decide which threshold to use and discern between customers who should be granted a loan or not. To obtain this type of scores, we work with classifiers that instead of only providing their classification outputs $h(\vec{x})$, also provide probabilities $P[h(\vec{x}) = 1]$ and $P[h(\vec{x}) = 0]$, which can be directly re-scaled to the $[0; 1,000]$ interval. Luckily, most of the classifiers allow the extraction of such probabilities. In Section 4.1 we discuss the classifiers used in our analysis and mention how probabilities are calculated from each one of them.

3. Related works

The use of machine learning models and artificial intelligence techniques for credit scoring have been drawing the effort of both researchers and practitioners over the last decade. In Li and Liao (2011), authors performed an exhaustive comparison of different types of classifiers, i.e., decision tree, neural networks, logistic regression, and regression trees; both with and without principal component analysis, and verified that decision trees were the most appropriate models w.r.t. Kolmogorov–Smirnov values (see Section 4) in a private Indian dataset. In contrast, feature-weighted Support Vector Machines have been applied in Chen and Shi (2013) to achieve superior F1 results in the Australian and German statlog datasets.[1] The work of Chen et al. (2017) proposed an iterative approach to train Support Vector Machines and Naive Bayes classifiers in a Bulgarian credit scoring dataset. The authors in Nalić and Svraka (2018) applied data mining techniques from the Oracle Data Miner and a feature selection to learn from a real-world dataset provided by a Micro-finance Institution from Bosnia. The credit scoring model built was a Logistic Regression model yielded by the Oracle platform. In Okesola et al. (2017) authors proposed the use of demographic variables and wealthy indicators to model the behavior of creditworthy and non-creditworthy creditors. In this dataset, a Naive Bayes model was built, tuned, and evaluated until it achieved a good prediction rate, which surpassed the Classification Tree (CTree) and K-Nearest Neighbor (KNN) in terms of global accuracy. The comparative study reported in Singh (2017) evaluated 25 classifiers from different categories in the Taiwan Credit dataset (Yeh & Lien, 2009): decision trees, lazy learners, Bayesian learners, simple logistics, multi-layer perceptron (MLP), and rule-based learners. Among them, the MLP showed the best AUC performance and the best models were Random Forest, Bagging and Logistic based on predictive accuracy. In Guo and Dong (2017), authors have proposed the use of Multi-Objective Particle Swarm Optimization (MOPSO) against traditional machine learning models on top of the German and United Kingdom datasets, such that the latter is private. The problem of credit scoring is tackled as a time series prediction issue in Saia and Carta (2017), where authors use Fourier transforms to use a different data representation. In this work, the German dataset was again used in conjunction with the Default of Credit Clients dataset.[2] Similarly, the same authors proposed the use of wavelets to create new features from the Australian, German, and Japanese Credit Screening[3] datasets, before a Random Forest was used to obtain the final scores. Assuming the same datasets described in the works above, a recent benchmark was given in Soares de Melo Junior et al. (2019), where authors conducted a grid search on 8 different types of classification algorithms, and showed that Random Forests (Breiman, 2001) achieve superior recognition rates in a training phase smaller than 5 min.

Another relevant work for machine learning-based credit scoring proposed the use of ensembles (Lawi et al., 2017), in which logistic regression models were combined with the gradient boosting method, and the results showed that this combination surpassed the traditional Bagging (Breiman, 1996) and the single logistic regression model in terms of accuracy rates. Also regarding ensembles, the work of Abellán and Castellano (2017) is relevant as it compares multiple types ensembles of classification models for the credit scoring problem, thus showing that these overcome single models. Ensembles were also combined with SMOTE oversampling technique (Chawla et al., 2002) in the work of Melo et al. (2019), where authors performed dynamic classifier selection in static credit scoring systems with reasonable success in mildly imbalanced scenarios.

Finally, we highlight that even though data stream mining is cited in surveys (Gomes et al., 2019; Žliobaitė et al., 2016) as an interesting technique for the creation of adaptive models for credit scoring, we found a small number of papers in which data stream mining has been applied to credit scoring problems. Therefore, this paper brings forward a new analysis of data stream classifiers in three different credit scoring datasets, thus showing their applicability and behavior.

[1] Both datasets are available at the UCI repository at http://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval) and https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data), respectively.

[2] Default of Credit Clients dataset is available in the UCI repository at https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients.

[3] The Japanese Credit Screening is available at https://archive.ics.uci.edu/ml/datasets/Japanese+Credit+Screening.

4. Experimental setup

In this section we report the experimental setup adopted during our analysis. First, we discuss the learning algorithms used, including both traditional batch learners and learners that are tailored for data streams, i.e., they allow updates over time as new training data becomes available. Later, we discuss the evaluation metrics used, followed by the datasets used and their main characteristics. Finally, we describe the validation schemes used to compute the evaluation metrics used in the comparison of different learners with different characteristics.

4.1. Learning algorithms

In this section we describe the learning algorithms used in our analysis. These algorithms were chosen due to their availability in data mining frameworks, i.e. *scikit-learn* (Pedregosa et al., 2011), and Massive Online Analysis (MOA) (Bifet, Holmes, Kirkby et al., 2010); by achieving state-of-the-art results (Bifet, Holmes and Pfahringer, 2010; Gomes et al., 2017) or because they are currently used by the dataset donors. Below, we describe these learners and divide them into batch and stream learners, where the former are trained over a static amount of training data, while the latter can be incremented over time.

4.1.1. Batch learners

Logistic Regression. Logistic regression (LR) is a linear model where the target variable (class) is categorical (Cox, 1958). For the binary task, it creates a linear model based on a sigmoid function that is used to estimate the probability of a binary response (in our case, if a customer requesting a loan will pay his debts in full or not) based on the input features. This is, by far, the most widely used approach for credit scoring and all 3 of our partners adopt it in their current models. As mentioned in Section 2, the reason for this choice is that Logistic Regression models are human-readable, which means that each feature and its coefficient can be analyzed individually, thus determining how important it is to discern between creditworthy and non-creditworthy customers. To extract probabilities for a loan request, a logit function is applied to the coefficients of our regression model.

J48. A decision tree (DT) uses a hierarchical representation for classification where nodes represent tests over the attributes and leaves represent the classes. Decision trees are learned by recursion, where each leave is replaced by test nodes, starting at the root, if a goodness-of-fit criterion is met. In particular, the J48 tree is an implementation of the C4.5 algorithm (Quinlan, 1993) available in WEKA (Hall et al., 2009) used in this study partitions the data and branches the tree according to the Information Gain function, and performs tree pruning to avoid overfitting by testing if the data distribution at a test node is statistically different to the overall training data. To extract probabilities from a J48 tree, each loan application $\vec{x}$ is traversed along the tree until a leaf is reached, and the probability is calculated as the ratio between past fully paid and defaulted loan requests observed at this node.

Naive Bayes. The Naive Bayes (NB) is a probabilistic classifier that works under the assumption that all features are independent from one another, with the exception of the target. In bayesian models, classifications are drawn using conditional probabilities of features and the target.

Random Forest. The Random Forest (RF) classifier (Breiman, 2001) is an ensemble of unpruned classification trees, which are induced from bootstraps of the training data (Breiman, 1996). In contrast to conventional decision trees, during the branching process, only a randomly selected subset of features is evaluated (Ho, 1998). The final predictions scores are obtained by averaging the output probabilities given by each of the trees.

4.1.2. Stream learners

Hoeffding Tree. Conventional decision trees assume that the entire dataset is available for training, yet, this assumption does not hold in streaming scenarios since data becomes available over time. To relax this constraint, authors in Domingos and Hulten (2000) proposed a swift method to incrementally learn decision trees called Hoeffding Tree (HT). In practice, the definition of whether a leaf should split and which feature to split on are estimated using a small data sample calculated with the *Hoeffding bound* (Hoeffding, 1963). The rationale behind Hoeffding trees is that, with high probability, the data distribution observed in a sample with size n adheres to the population distribution, which is expected to be infinite in streaming scenarios. Similarly to the J48 tree, probabilities for a loan application $\vec{x}$ are obtained by traversing the decision tree until a leaf is reached, yet, a bayesian classifier is used instead of the class proportion observed at the terminal node.

Hoeffding Adaptive Tree. The Hoeffding Adaptive Tree (Bifet & Gavaldà, 2009) is an extension to the original Hoeffding Tree that constantly keeps track of the error rates of split nodes as the stream progresses. These error rates are used to feed an ADWIN concept drift detector (Bifet, 2009) which flags changes whenever significant increases in the error rates occur, and as a result, a split node is replaced by a leaf. This allows the Hoeffding Adaptive Tree to dynamically adjust its predictive model over time as new data becomes available. Finally, the output probabilities for a test instance are obtained via a bayesian classifier observed at the terminal leaf node.

Leveraging Bagging. Leveraging Bagging (LEVBAG) (Bifet, Holmes and Pfahringer, 2010) is an extension to Online Bagging (Oza, 2005) where the weights of training instances are randomized according to a Poisson distribution with a mean $\lambda = 6$, the ADWIN drift detector (Bifet, 2009) is used to flag drifts and reset classifiers accordingly, and random output codes are used to improve the accuracy of the whole ensemble. As in the original paper, we have conducted our analysis by creating a leveraging Bagging ensemble of Hoeffding trees with the default values available in the MOA framework (Bifet, Holmes, Kirkby et al., 2010), i.e. an ensemble size of 10 trees. Finally, probabilities are obtained by averaging the probabilities obtained in each of the subtrees available in the ensemble.

Adaptive Random Forest. The Adaptive Random Forest algorithm (ARF) was introduced in Gomes et al. (2017) with the goal of allowing adaptive learning from data streams by extending the original Random Forests of Breiman (2001). ARF combines drift detection as in Leveraging Bagging, ensemble adjustments, limited tree sizes, and background learning to improve accuracy rates over concept drifting data streams. As in the other ensembles, the final output probability scores are obtained by averaging the outputs obtained from each subtree.

4.2. Evaluation metrics

The assessment of machine learning tailored for credit scoring requires specific metrics that highlight how well creditworthy and non-creditworthy customers are discerned. In this paper, we follow the assessment procedure provided in guidelines of the area, more specifically the works of Kritzinger (2017) and Rezac and Rezac (2011a). To perform the assessment of the ability of machine learning methods to discriminate customers that will pay their debts in full or not, we follow the Kolmogorov–Smirnov (KS) metric (Rezac & Rezac, 2011b). Furthermore, it is also relevant to determine whether the population scores are stable or fluctuate over time. To measure how volatile the population scores are, we adopted the Population Stability Index (PSI) (Karakoulas, 2004). If the PSI rates fluctuate over time, this may be an indication that the predictive model has deteriorated or that the population itself is changing and that a new model should be built.

KS. The Kolmogorov–Smirnov (KS) statistic indicates the maximum distance between the cumulative probability distribution function (*cdfs*) obtained by customers that pay their debts in full and those who

default (Rezac & Rezac, 2011b). Assuming that we are scoring $(n+m)$ customers, we denote that the ith customer will default as $D_i = 1$, and $D_i = 0$ otherwise. Also, the empirical cumulative distribution function (*cdf*) of good and bad customers are given by Eqs. (1) and (2), respectively, where n is the total number of good customers, m is the number of bad customers, $L = \min s_i, 1 \leq i \leq (n+m)$ is the lower bounds of all the scores available, and $H = \max s_i, 1 \leq i \leq (n+m)$ is the upper bound.

$$F_{\text{good}}(a) = \frac{1}{n}\sum_{i=1}^{n}\begin{cases}1, \text{if } s_i \leq a \wedge D_i = 1\\ 0, \text{otherwise}\end{cases}, \text{with } a \in [L, H] \tag{1}$$

$$F_{\text{bad}}(a) = \frac{1}{m}\sum_{i=1}^{m}\begin{cases}1, \text{if } s_i \leq a \wedge D_i = 0\\ 0, \text{otherwise}\end{cases}, \text{with } a \in [L, H] \tag{2}$$

The KS metric is then given by Eq. (3), which is the maximum difference between the *cdfs* that describe the good and bad customers.

$$KS = \max_{a \in [L,H]} |F_{\text{bad}}(a) - F_{\text{good}(a)}| \tag{3}$$

A zero value for the KS statistic means that the two credit-score distributions are the same and indicates that the credit score fails to differentiate between defaulters and nondefaulters; a value equal to 100 indicates that the credit score perfectly differentiates defaulters from nondefaulters. As a rule of thumb, a KS score greater than 35% depicts a reasonable discriminative power of the predictive model to discern between the different types of customers.

PSI. The population Stability Index (PSI) indicates changes in the population of loan applicants. It is important to note that this may or not be an indicative of deterioration of the predictive model to predict risk, yet, PSI depicts changes in the environment that need to be further investigated by the bank experts to determine whether any macroeconomic conditions or lending policies are affecting the model outcomes (Karakoulas, 2004). To compute the PSI score, the probability distribution function (*pdf*) of the defaulting customers in two different time periods is calculated. First, the *pdfs* of these distributions are computed using a specified number of ranges r so that each range has approximately the same number of defaulting customers. Here, we denote n_i and m_i to be the counters of defaulting customers in the two samples at the ith bin, and that $\sum n_i = N$ and $\sum m_i = M$. Given these counters, it is possible to compute the PSI, which is given by Eq. (4).

$$PSI = \sum_{i=1}^{r}\left[\left(\frac{n_i}{N} - \frac{m_i}{M}\right) \times \left(\ln\frac{n_i}{N} - \ln\frac{m_i}{M}\right)\right] \tag{4}$$

In our analysis, we report PSI rates obtained by comparing two subsequent months. According to our partners, PSI rates below 10% show that the population is reasonably stable and that the model is not unacceptably volatile.

4.3. Feature importance

In this paper we report the feature importance of the 10 best-ranked features according to three perspectives: Information Value (IV) (Mays & Mays, 2004), Mean Decrease Impurity (MDI) (Louppe et al., 2013), and Mean Positional Gain (MPG) (Barddal & Enembreck, 2019).

Information Value (IV). The Information Value (IV) measures the strength of the relationship between each variable x_i individually and the target y. IV ranks features based on the importance and amount of information they carry. As mentioned in Mays and Mays (2004), IV should be used only during the creation of Logistic Regression models, as it analyzes each feature individually. Assuming that a feature is split into j partitions, the IV for a feature x_i is given by:

$$\text{IV}(x_i) = \sum_j (\%\text{ of non-defaults in } j - \%\text{ of events in } j) \times \text{WOE}(x_i, j) \tag{5}$$

such that the Weight of Evidence (WOE) of the same feature x_i in its jth partition is given by Eq. (6).

$$\text{WOE}(x_i, j) = \ln\left(\frac{\#\text{ non-defaults events in } j}{\#\text{ of defaults in } j}\right) \tag{6}$$

Mean Decrease Impurity (MDI). MDI (Louppe et al., 2013) ranks features according to their position in decision tree ensembles. It is a weighted sum of the heuristic values obtained during the time of the split that accounts for the number of samples it splits. MDI is given by Eq. (7), where t_i is an arbitrary tree inside the ensemble of trees $T = \{t_1, t_2, \ldots, t_E\}$, N_t is the number of instances observed in a split node b, N is the total number of samples observed in the entire tree, $J(b)$ is the heuristic value (goodness-of-fit) computed during the split process of a node b, and $\Omega(b)$ is the function that returns the feature $X_i \in X$ selected in b.

$$\text{MDI}(x_i) = \frac{1}{|T|}\sum_{t_i}^{T}\sum_{b}^{t_i}\begin{cases}\frac{N_t}{N} \times J(b), & \text{if } \Omega(b) = x_i\\ 0, & \text{otherwise}\end{cases} \tag{7}$$

In the following analysis provided in Section 5, we apply MDI to rank features in the Random Forest (RF) classifier.

Mean Positional Gain (MPG). Using MDI in incremental models that use decision trees such as Hoeffding Trees (HTs) and the Adaptive Random Forest (ARF) is problematic as the N component continuously grows and N_t is static. In the HT and its variants N_t is the grace period parameter, which is static across all split nodes, while N continuously grows as new data becomes available. Consequently, $\frac{N_t}{N}$ penalize all features equally regardless of their position inside the tree structure. This behavior is different from the one observed in traditional decision trees, where the proportion $\frac{N_t}{N}$ is larger on the superior nodes and decreases as we traverse the tree structure and reach the leaves. Therefore, the Mean Positional Gain (MPG) (Karax et al., 2019) replaces the $\frac{N_t}{N}$ component by another ratio that takes into account the position of a node b inside the tree structure w.r.t. the entire tree depth. More formally, the MPG for a feature X_i is given by Eq. (8), where $\gamma(t_i)$ is the function that returns the number of split nodes in a tree $t_i \in T$, $h(t_i)$ is the tree height, $h(b, t_i)$ is the tree depth of a split node b in t_i, and R is the maximum value of the heuristic J.

$$\text{MPG}(x_i) = \frac{1}{|T|}\sum_{t_i}^{T}\frac{1}{\gamma(t_i)}\sum_{b}^{t_i}\begin{cases}\frac{h(t_i)-h(b,t_i)}{h(t_i)} \times \frac{J(b)}{R}, & \text{if } \Omega(b) = x_i\\ 0, & \text{otherwise}\end{cases} \tag{8}$$

In Section 5, we report the MPG for the Adaptive Random Forest (ARF) classifier.

4.4. Hyper-parameter tuning

Each of the learning algorithms described in Section 4.1 possess different hyper-parameters that may impact the performance of the predictive models to be learned. Per dataset, batch learners were optimized using a grid search with 10-fold cross-validation with the goal of maximizing the KS rates in the test data. Similarly, data stream learning algorithms were tuned so that the KS rates during the test were optimized. The hyper-parameters tweaked during this process for both batch and stream learning algorithms are given in Table 1. The only classifier omitted from Table 1 is Naive Bayes as no hyper-parameters are available for tuning.

4.5. Datasets

In this section we present the main characteristics of the datasets analyzed. We used 3 different datasets during this study, each obtained from a different financial institution or credit scoring operators, hereafter referred to as *Credit Scoring DataSet* (CSDS) 1, 2, and 3. We refrained from using traditional credit scoring datasets used in the literature as they are either (i) too small in terms of features and instances available, and/or (ii) unclear with regard to what each

Table 1
Hyper-parameters tested during the tuning process for both batch and stream learning classifiers.

Classifier	Hyper-parameter	Values
Logistic regression	Penalty	{l1, l2 elasticnet}
	Intercept	{True, False}
	Solver	{newton-cg, saga}
J48	Maximum tree depth	{3, 5, 10, 20, ∞}
Random forest	Number of trees	{10, 50, 100, 150}
	Maximum tree depth	{3, 5, 10, 20, ∞}
	Number of features to test during split	{3, 5, 10, $\sqrt{d}$}
Hoeffding tree	Grace period	{50, 200, 500}
	Pre-prune	{True, False}
Hoeffding adaptive tree	Grace period	{50, 200, 500}
	Pre-prune	{True, False}
Leveraging bagging	Ensemble size	{10, 50, 100, 150}
Adaptive random forest	Number of trees	{10, 50, 100, 150}
	Maximum tree depth	{3, 5, 10, 20, ∞}
	Number of features to test during split	{3, 5, 10, $\sqrt{d}$}

Table 2
Overview of the datasets used in this study.

Dataset	CSDS-1	CSDS-2	CSDS-3
# of months	15	25	16
# of Instances	315,539	50,401	97,226
Class ratio (%) (Fully paid - defaulted)	87–13	98–2	74-26
# of Features	178	37	152
Performance on past debts	✓	✓	✓
Profiling	✓		✓
Mainhousehold	✓		
Income	✓	✓	✓
Occupation	✓		
Demographics	✓		
Possessions	✓		✓
Census	✓		

feature stands for. An entirely anonymized version of the datasets can be retrieved from http://www.ppgia.pucpr.br/~jean.barddal/datasets/CSDS.zip, strictly for academic use. Within the provided *zip* file, both ARFF and CSV formats are made available. In each file, the features included are: `CROP`, which is the year and month of each datum, `TARGET` is the label, and the remainder of the features were anonymized and are labeled in the `Vi` format, where `i` is the index of the feature.

The actual names of the companies that provided the datasets were omitted, as well as the variables available in each dataset, due to contract limitations. However, in Table 2, we overview the datasets used and some of its main characteristics, including the number of months, the number of instances, the class ratio (i.e., the number of loans that were or not fully paid), number of features, and whether the dataset contains or not certain types of features, such as performance of customers on past debts, profiling, main household, income, occupation, demographics, possessions and census data. Also regarding the overview provided in Table 2, it is important to emphasize that each dataset contains different features for each of the feature types, e.g., the features available for CSDS-1 in terms of performance on past debts not necessarily are the same as those available for CSDS-2 and CSDS-3, and so forth. The target for all datasets represents whether customers paid their debt in full (0) or defaulted (1) in the month being analyzed.

CSDS-1. This dataset contains information about more than 315 thousand loans requests that were analyzed, approved, and conceded. The target here is to identify whether each customer paid his debts in full or defaulted in a timespan of 6 months. Overall, this data represents loan requests from June 2016 until August 2017. In this dataset, the goal was to construct predictive models with competitive KS rates when compared to two major credit scoring bureaus of Brazil while maintaining PSI rates below 10% between two subsequent months.

CSDS-2. This dataset contains information about 50 thousand loans provided for customers in a car financing company. We highlight that this dataset is highly imbalanced in the sense that nearly all customers paid their debts in full and only 2% defaulted in a 2-month period. In total, this dataset represents loan requests from January 2016 until January 2018. The goal in this dataset was to construct predictive models to evaluate whether the features available in this car financing company would result in interesting KS results and low PSI.

CSDS-3. Similarly to CSDS-1, this dataset contains information about 97 thousand loans that were analyzed, approved, and conceded by a large bank in Brazil. The goal is to identify whether a customer defaulted or not in a 3-month span. This dataset represents loan requests from October 2013 until January 2015. Similarly as before, the goal in this last dataset was to build predictive models with KS rates that were competitive against the predictive model already in use by the bank whilst reporting PSI rates below 10% across two subsequent months.

4.6. Validation schemes

When using batch machine learning, it is good practice to divide the data available into training and test sets, such that these samples are disjoint, a process called holdout validation. The holdout validation scheme is also the process that all of our partners perform as they sort the data over time and then often assume the first 60% or 70% of the data as training data and the remainder as test. One drawback of holdout validation is that it was not tailored for streaming scenarios, where data becomes available over time. Therefore, we also adopted a test-then-train validation process where the data of each month is used for training right after its evaluation. Both processes are detailed below.

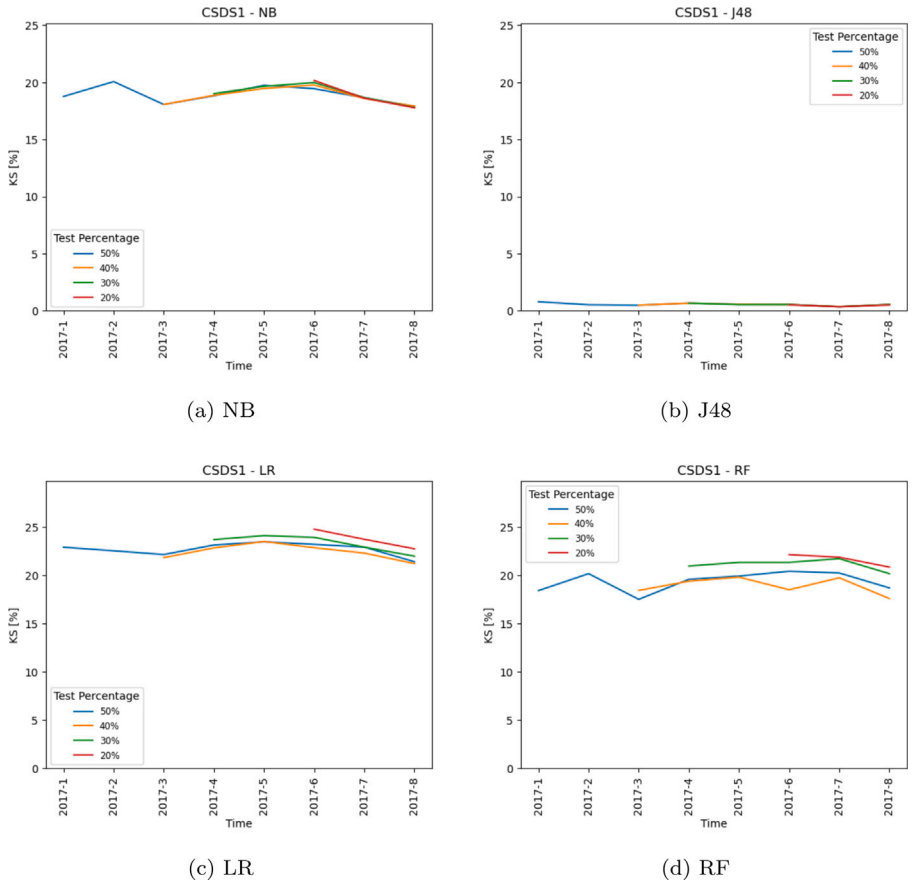
Holdout. In this validation scheme, we split the data available in a specific number of months for training and the remainder for testing. The idea is to verify whether after the creation of the predictive model, AUROC, KS and PSI behave well or not over time. For instance, if the data is indeed ephemeral, we would expect both AUROC and KS rates to drop over time after the model is learned, while PSI rates would increase. If one observes that AUROC, KS, and PSI rates stagnate, incremental learning would not be required. During our analysis, we will test different ratios of months for training and testing, including 50%–50%, 60%–40%, 70%–30% and 80%–20%. The reason for testing different proportions of the dataset here is to verify the impact of more or less training data and whether the behavior of loan applications change over time. For instance, if there is a drift that is affecting loans, and this is only observable in the test set, then a batch model would imprecisely score these applications, resulting in lower KS and AUROC rates and higher PSI.

Monthly Test-then-train. In this validation scheme, each month is used for training right after it is used for evaluation. The only exception is the data from the first month, which is used solely for training. The idea here is to compare the results obtained by a learner that is constantly being updated with new data against a model that was learned until a certain point and only evaluated after. By comparing these results, we will be able to identify whether continuously updating our predictive models significantly improves the AUROC and KS results without jeopardizing PSI rates.

5. Analysis

In this section we analyze both batch and data stream learning algorithms in the context of 3 real-world datasets previously presented in Section 4.5. The results obtained for each dataset will be discussed separately and in 3 steps:

1. First, we report and analyze the KS results obtained by batch learning algorithms when trained and tested using different proportions of the dataset using holdout validation. The goals with this analysis are: (i) to verify whether more training data directly translates to higher KS rates on test data, and (ii) to pick the best performing classifier for further comparisons against data stream approaches and bureaus (whenever possible);



(a) NB (b) J48

(c) LR (d) RF

Fig. 1. Results obtained different training and test set ratios.

2. Next, we compare different data stream mining algorithms in terms of KS and PSI rates. The goal is to identify whether continuously updating the predictive models using a monthly test-then-train validation process leads to higher prediction rates without damaging PSI rates;
3. The next step then compares the results obtained by the best performing batch classifier against data stream learners, thus highlighting whether there is significant difference between them in terms of KS and PSI or not;
4. For CSDS-1 and CSDS-3, the KS and PSI rates obtained for one or more existing credit scoring models are also available, and thus, the best results are compared to verify how well batch and streaming approaches perform; and finally
5. We analyze feature importance according to Information Value (IV), Mean Decrease Impurity (MDI), and Mean Positional Gain (MPG).

5.1. CSDS-1

In this section, we first report the results obtained by batch classifiers across different ratios of training and test data in Fig. 1. First, we highlight the difference of KS rates across the learners, as J48 underfits and is unable to discern between creditworthy and non-creditworthy customers, and thus, it is removed from the remainder of the analysis. In practice, despite the hyper-parameter tuning, J48 is unable to learn a decision tree and the model remains a single leaf. On the other hand, the remainder of methods show KS results that go from 11% up to 20% across the different training and test portions of the dataset. At this point, we must select which classifier is the most appropriate in terms of KS for comparisons against data stream learning algorithms. In this specific scenario, it was a request that the validation process was held to a holdout approach where approximately 70% of the data would be used for training and the remaining 30% for testing, and thus, only the months highlighted in gray (from 2017–04 to 2017–08) were used for this comparison. Given the aforementioned requirement, the average KS rates obtained by the Logistic Regression (LR) classifier rank it as the most suited, and thus, it was chosen for further analyses. It is also worthy to highlight that, as shown in Fig. 2(b), the PSI rates obtained for all streaming learners, except HT and HAT, are remarkably low and comparable to LR, thus showing that the models and data are stable. In the same figure, the results obtained by Logistic Regression and Random Forest across multiple portions of the dataset for training and test sets show that the average KS rates obtained with more training data (roughly 80%), yielded the best results, yet, the same did not hold for Naive Bayes, as using 70% of the data for training was the best option.

In Fig. 2(a) we continue our analysis with a comparison KS rates of stream learning algorithms, i.e., ARF, HT, HAT, and LEVBAG. The results for ARF, HT, and LEVBAG show that there is a positive trend as the KS rates grow over time. On the other hand, the results for the HAT classifier are reasonably lower when compared to others, thus showing that this classifier is not able to discern between creditworthy and non-creditworthy customers even with the arrival of more training data. In terms of average KS rates in the test data, also assumed to be the period between 2017–04 and 2017–08, the ARF classifier was the most accurate, and thus, it was used for comparison against LR. Following the Wilcoxon non-parametric paired test and assuming a confidence level of 95%, we found that there is no significant statistical difference between both methods, and that the PSI rates of both classifiers are low.

Next, as KS rates and PSI rates for two bureaus were available for the same dataset, it was also possible to compare both batch and streaming

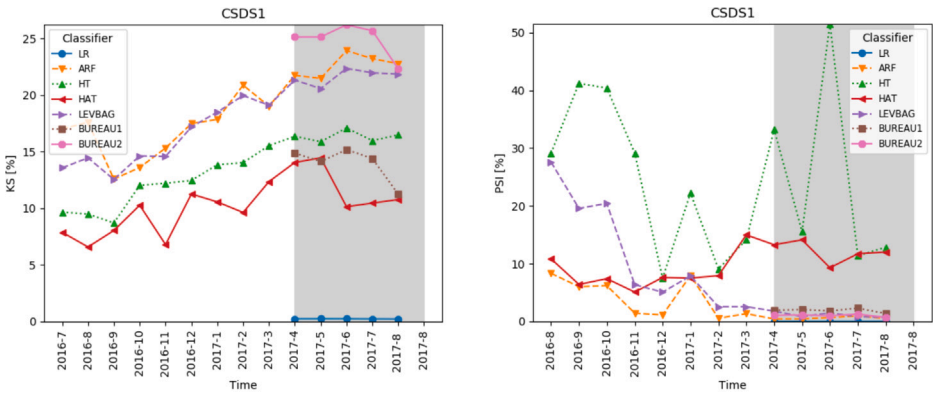


Fig. 2. KS (%) and PSI (%) rates obtained in CSDS-1 experiment. The gray area corresponds to test data.

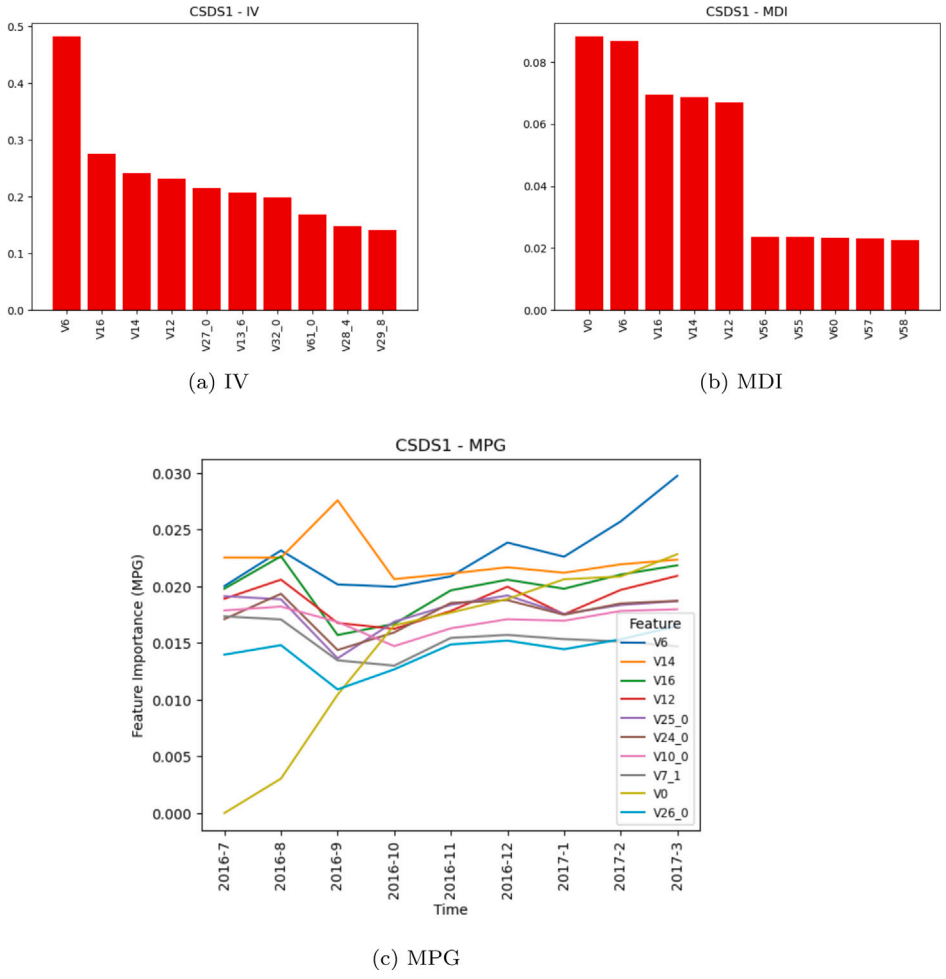
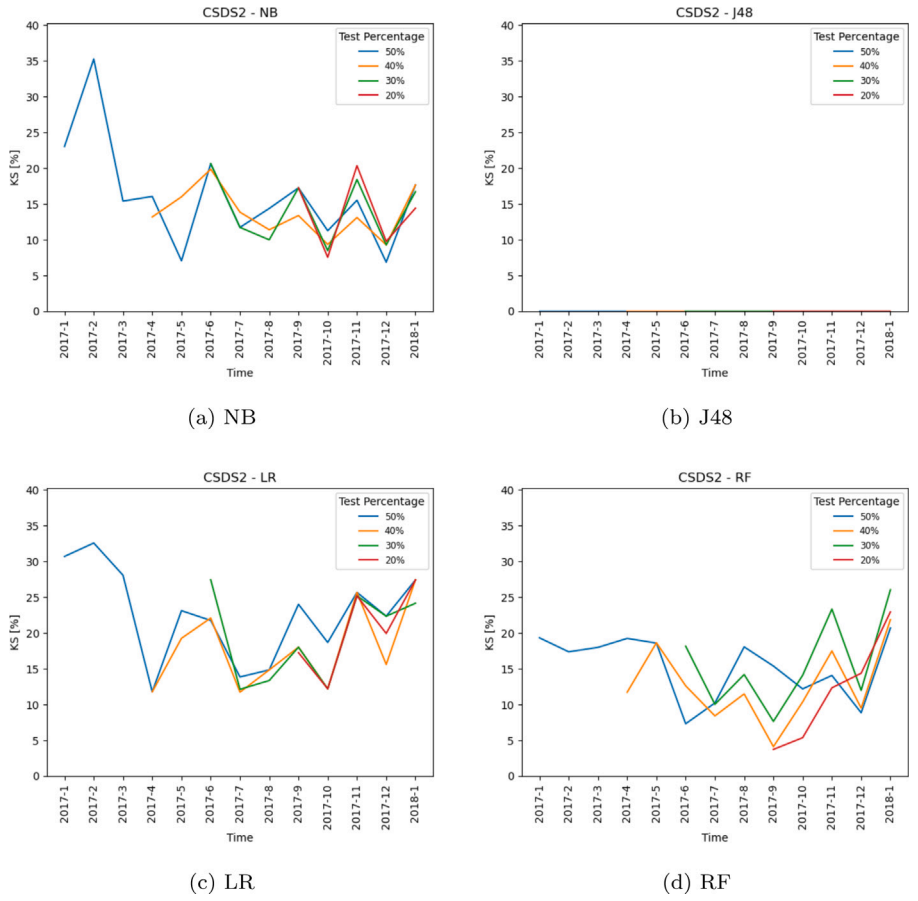


Fig. 3. Feature importance obtained using Information Value (IV), Mean Decrease Impurity (MDI), and Mean Positional Gain (MPG) in the CSDS1 experiment.

models against the results obtained by these major market players. The results obtained by Bureau #1 are clearly worse than the those obtained by LR and ARF, as the average KS is of 13.96%, while the KS for LR is of 23.35% and of 23.82% for ARF. On the other hand, the results for Bureau #2 are much more competitive, as it yields 24.89% of KS, which is statistically superior to both LR but not ARF, while also showing small PSI rates. The results obtained for Bureau #2 are expected since this market player is well known for gathering data about customers from the entire country and maintains both black- and white-lists of customers according to their credit requests and payments.

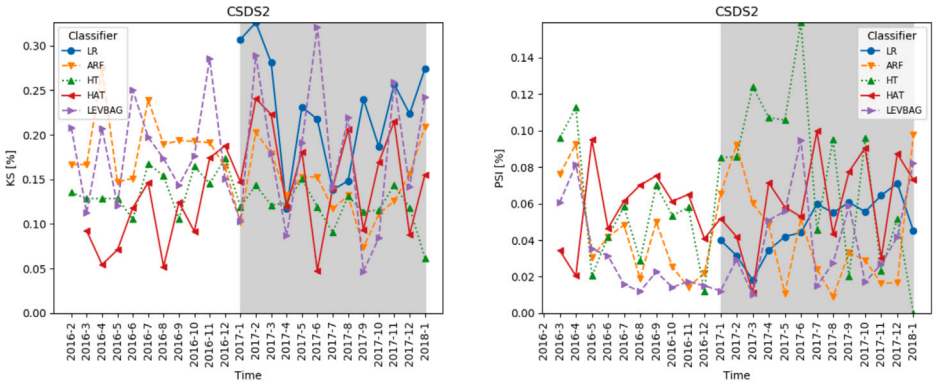
Finally, the feature importance values obtained using IV, MDI, and MPG are given in Fig. 3. The IV results depicted in Fig. 3(a) show that V6 is, by far, the most informative feature, followed by V16, V14, and V12. On the other hand, the MDI rates obtained using a batch Random Forest (RF) classifier show that V0 and V6 are the best-ranked features, followed by V16, V14, and V12. Comparing the ranking obtained by IV and MDI, we observe that V0 was ranked as the best feature in terms



(a) NB (b) J48

(c) LR (d) RF

Fig. 4. Results obtained different training and test set ratios.



(a) (b)

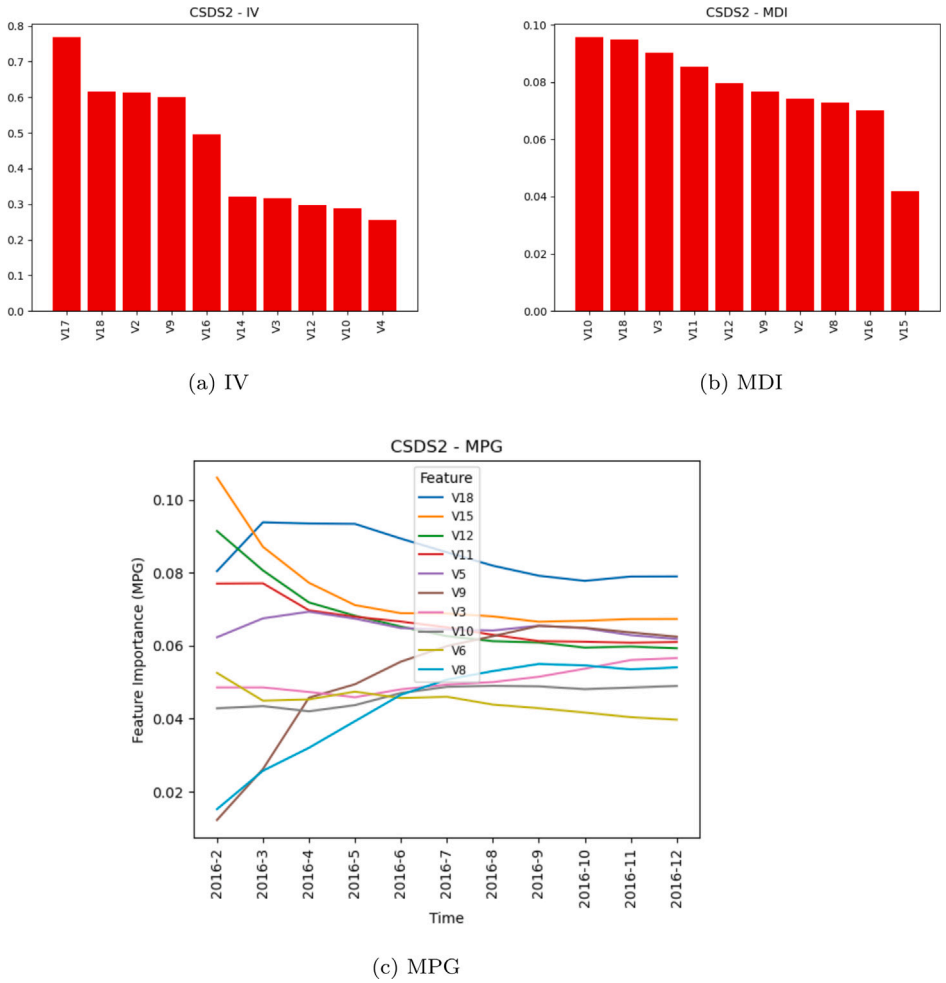
Fig. 5. KS (%) and PSI (%) rates obtained in CSDS-2 experiment. The gray area corresponds to test data.

of MDI, but is was unlisted in the IV ranking. The MPG rates observed during the training of the Adaptive Random Forest (ARF) classifier are given in Fig. 3(c), where the best-ranked features are V6, V0, and V16. One important aspect here regards feature V0, as it started with nearly 0 importance, and it grew over time, reaching the second spot in the feature ranking in March, 2013.

5.2. CSDS-2

As in the previous section, the KS results obtained for different batch learning algorithms across different training and test proportions are reported in Fig. 4. Similarly to CSDS-1, the J48 classifier was unable to learn a consistent predictive model to discern between creditworthy and non-creditworthy customers. In practice, no decision tree was learned at all, as a single tree node exists in each of the trees attempted and these classify all proposals as creditworthy since this is the majority class. Differently from the previous experiment, the KS results observed in Fig. 5(a) shows that there is no trend exhibiting that more training data yields higher KS rates. The reason behind this behavior is that the dataset is quite stable and that there is no statistical difference between the KS rates observed.

Also in contrast to the previous experiment, where the size of the holdout partitions were predefined, we did not have a specific training and test proportion to follow. Therefore, we conducted a



(a) IV (b) MDI

(c) MPG

Fig. 6. Feature importance obtained using Information Value (IV), Mean Decrease Impurity (MDI), and Mean Positional Gain (MPG) in the CSDS2 experiment.

statistical comparison to determine what classifier and partition sizes were the most suited when compared to one another. As a result of comparisons performed with the Mann–Whitney U-test and Bonferroni correction (Corder & Foreman, 2011) assuming a 95% confidence level, the Logistic Regression classifier was selected, yet, now using a 50%–50% proportion for training and testing.

The next step in our analysis was then to compare the best performing batch classifier against streaming approaches. The KS and PSI rates obtained and that allow this comparison are given in Fig. 5. We start the comparison by analyzing the KS and PSI rates obtained by ARF, HT, HAT, and LEVBAG. During the beginning of the experiment, we observe that ARF achieves the highest KS rates, while LEVBAG finds the best rates during the second part of the experiment. Also, exhibiting a behavior similar to the one observed in CSDS-1, all learners obtained PSI rates which are below 10% throughout the entire experiment.

Continuing the analysis, focusing on the KS rates located at the shaded area of Fig. 5(a), we then compared LEVBAG against HAT using the Wilcoxon paired test assuming a 95% confidence level, which showed that there is no statistical different amongst them. Given that, LEVBAG was chosen as the best performing streaming algorithm which was then compared to LR. Here, the difference in KS rates between LR and LEVBAG is marginal, as the former achieves an average KS of 22.67%, opposed to 22.48% achieved by the latter. In this case, the Wilcoxon paired test showed that there is statistical difference between LR and LEVBAG assuming a 95% confidence level.

Finally, Fig. 6 depicts the feature importance values obtained according to IV, MDI, and MPG metrics. First, in Fig. 6 we have IV that is related to Logistic Regression, which is the overall best performing model. Amongst the 10-best ranked features regarding IV, we see that V17 is the most relevant feature, followed by V18, V2, V9, and V16. A relevant drop is then observed, and features V14, V3, V12, V10, and V4 are listed. The MDI results obtained with a batch Random Forest (RF) are depicted in Fig. 6(b), where 9 out of the 10 best-ranked features show similar MDI values. Comparing the results for IV and MDI, we see that these are dissimilar, for instance, as V17 is the best feature in terms of IV, but it is not even in the top 10 list for MDI. Finally, the MPG rates obtained for features in the Adaptive Random Forest are given in Fig. 6(c), where V18 is listed as the most important feature during most of the training data, followed by V15 (which is not listed for IV), and V12.

5.3. CSDS-3

In this final section, we compare the results obtained during the CSDS-3 experiment. In Fig. 7, we report the KS rates obtained by the batch classifiers across different training and test ratios. In contrast to what has been observed in the previous experiments, the KS rates shown here are higher, thus highlighting the efficiency of the models to discern between creditworthy and non-creditworthy loan requests. The main reason behind the higher KS rates is that the entire dataset has been preprocessed in the sense that both feature engineering and selection have been conducted by a team of experts that have been working on this scenario for many years.

Similarly to the protocol followed in CSDS-1, it was requested that our analysis would follow a holdout process where approximately 70%

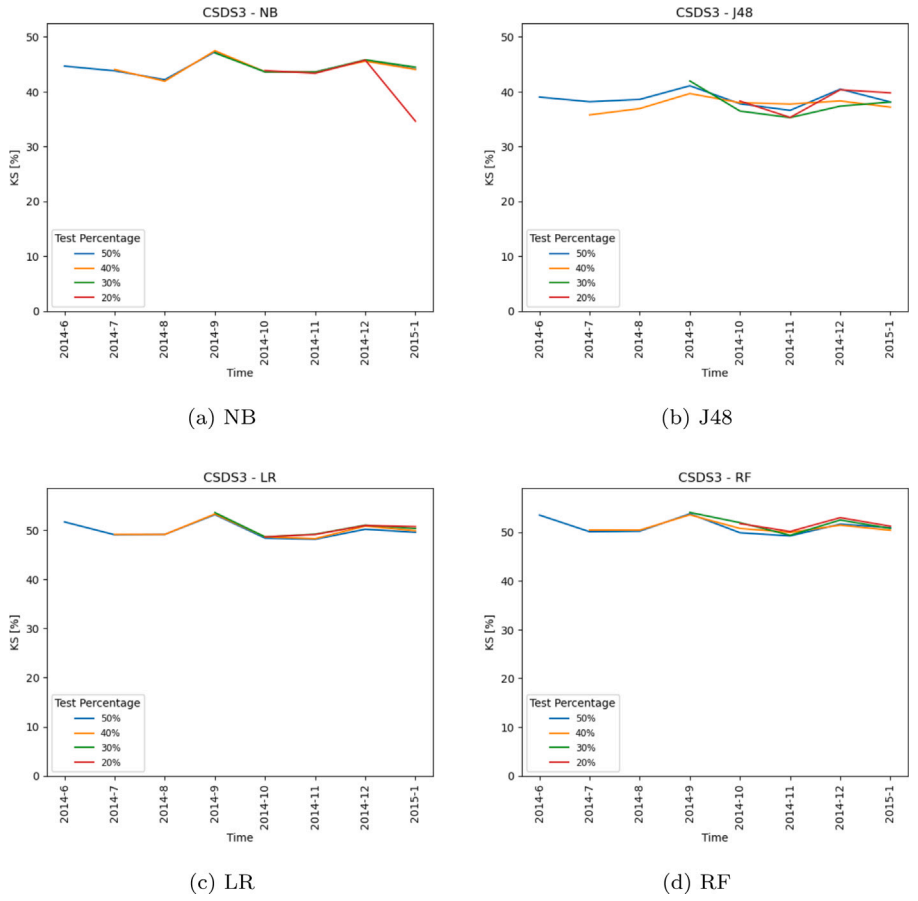


Fig. 7. Results obtained different training and test set ratios.

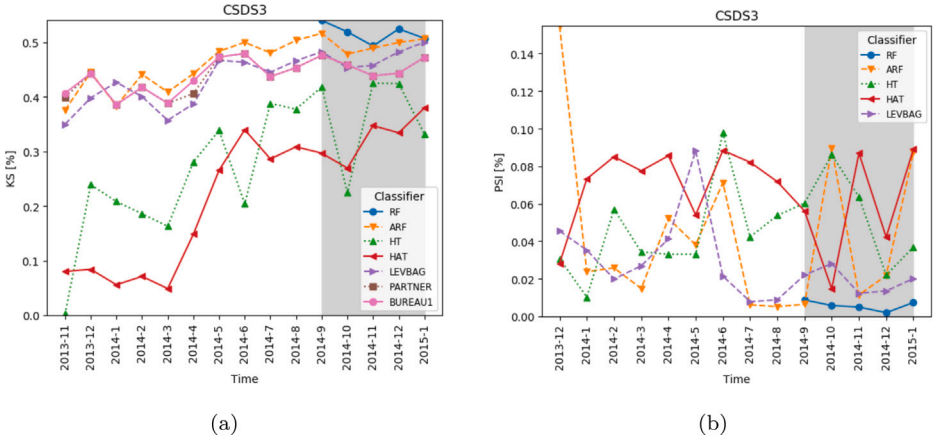
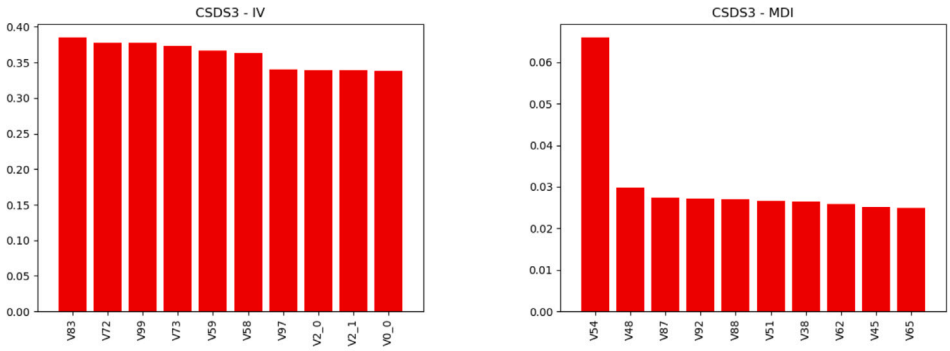


Fig. 8. KS (%) and PSI (%) rates obtained in CSDS-3 experiment. The gray area corresponds to test data.

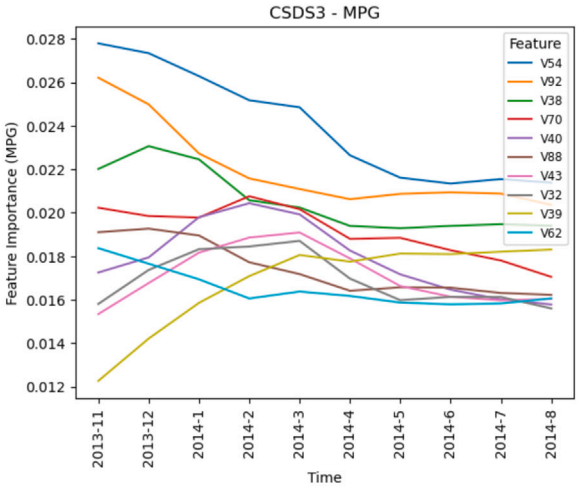
of the data would be used for training and the remaining 30% for testing. Given that, the results depicted in Fig. 8(a) show that the Random Forest (RF) achieves the highest KS rates (51.71%), followed by Logistic Regression (LR) with 50.59% and Naive Bayes (NB) with 44.87%. These classifiers were then compared against each other with the Wilcoxon test and Bonferroni correction, and as a result, the Random Forest was selected for posterior comparison as it has superior performance assuming a 95% confidence level.

Next, in Fig. 8(a) and (b), the KS and PSI rates obtained for data stream learners are given and compared to RF and a bureau model. Focusing on the results obtained by streaming algorithms, i.e., ARF, HT, HAT and LEVBAG; we observe that ARF dominates most part of the experiment in terms of KS, followed by LEVBAG. Comparing the KS rates obtained by all stream learners in the test portion of the experiment (highlighted in gray in Fig. 8(a)), we observe that ARF is statistically superior to HT, HAT, and LEVBAG assuming a 95% confidence level. Furthermore, following the trends observed in experiments CSDS-1 and CSDS-2, the PSI rates for streaming algorithms is stable, as the largest PSI value observed is of 08.91% for the HAT.

Next, still in Fig. 8(a), we compare the results obtained by the traditional Random Forest (RF) against its adaptive version, the Adaptive Random Forest (ARF). At this point, it becomes clear that RF outranks ARF in all the months evaluated during the test period, which is corroborated with the application of Wilcoxon's test. Furthermore,



(a) IV (b) MDI



(c) MPG

Fig. 9. Feature importance obtained using Information Value (IV), Mean Decrease Impurity (MDI), and Mean Positional Gain (MPG) in the CSDS3 experiment.

we highlight the fact that both ARF and RF overcome the KS rates observed for the bureau model, thus showing that the features available are relevant and would yield beneficial results when used to build predictive models for our partner. This would be beneficial as the partner in question would yield higher non-creditworthiness rates, thus leading to smaller default rates, while also reducing the costs with bureau scores, as these would not be required.

Finally, the IV, MDI, and MPG rates are given in Fig. 9. Amongst the 10 best-ranked features in terms of IV and Logistic Regression (Fig. 9(c)), we see that not much deviation in their importance values is observed, yet, V83, V72, and V99 are the best-ranked features. The behavior observed for MDI and the batch Random Forest is entirely different, as V54 is, by far, the most important feature, followed by V48, V87, and V92. Another behavior is seen in MPG and the Adaptive Random Forest in Fig. 9, as V54, V92, and V38 are the best-ranked features. These results show that MDI and MPG rankings are similar, yet, the KS observed for batch RF show that tree adaptation worsened ARF results and also made its PSI rates higher (Fig. 8(a) and (b), respectively). Such differences are really interesting, as the intersection between the best-ranked features in LR and RF models is roughly inexistent, yet, both achieve comparable KS rates (Fig. 8(c) and (d)).

6. Conclusion

In this paper, we bring forward an analysis of 3 different interactions with Brazilian financial institutions and credit scoring operators. In each of these interactions, we verified whether the credit scoring task should be addressed as a data stream classification problem, as many of the variables as mentioned above may drift over time, and thus, data stream mining techniques should be used since they are incremental and adaptive. We applied both batch and data stream learning techniques in different validation schemes to assess the Kolmogorov–Smirnov (KS) and Population Stability Index (PSI) of all models. Throughout our analysis, different aspects of the credit scoring application using machine learning can be highlighted. First, when creating credit scoring models using traditional batch learning classifiers, there is no guarantee that more training data will result in higher KS rates. Evidently, these results may be counter-intuitive, yet, more data may lead classifiers to overfit, and the data from different months and years may exhibit drifting characteristics. Next, our analysis targeted a comparison of data stream learning techniques against batch learning classifiers. As a result, data stream learning techniques yielded interesting discriminative rates in all the tested datasets, even surpassing the rates obtained by credit scoring bureaus that possess sensitive data and decades of experience in the credit scoring business (see CSDS-3, for example). This is of the utmost importance for financial institutions since they possess humongous data lakes that could be better taken advantage of. Furthermore, the data stream processing approach for credit scoring is also beneficial in the sense that there is no need to learn an entire model from scratch, as it can be simply updated over time as new training data becomes available. Given that, financial institutions should account for data stream learning and add these types of models to their plethora of approaches to be assessed when creating credit scoring models, as the working hours and financial resources spent on tailoring a new model could be greatly improved.

In future works, we plan to further address the credit scoring problem using data stream mining techniques as they present two important issues: class imbalance and feature selection. For instance, in all of the tested datasets, the ratio between creditworthy and non-creditworthy customers is imbalanced, which causes several classifiers to poorly separate these classes (Krawczyk, 2016). Therefore, we plan to develop class balancing techniques that are specific for the credit scoring problem and compare these to existing works in the area (Cano & Krawczyk, 2020; Ferreira et al., 2019; Loezer et al., 2020).

Another important gap regards feature selection, as the experiments conducted in this paper use all features available, yet, there is no guarantee that all the features are relevant. Furthermore, we plan to further analyze feature importance over time, more specifically in CSDS-2 and CSDS-3, as larger date spans are available and feature drifts are possible (Barddal et al., 2017).

Finally, we expect that the datasets and results reported in this paper are relevant to the credit scoring community, as similar analyses can be performed using data provided, as well as data from different countries, thus highlighting whether economically more stable countries exhibit similar behavior over time.

CRediT authorship contribution statement

Jean Paul Barddal: Conceptualization, Methodology, Software, Validation, Resources, Writing - original draft, Writing - review & editing, Visualization. **Lucas Loezer:** Methodology, Software, Validation, Data curation, Writing - original draft. **Fabrício Enembreck:** Conceptualization, Resources, Writing - review & editing. **Riccardo Lanzuolo:** Conceptualization, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abellán, J., & Castellano, J. G. (2017). A comparative study on base classifiers in ensemble methods for credit scoring. *Expert Systems with Applications*, *73*, 1–10. http://dx.doi.org/10.1016/j.eswa.2016.12.020, URL: http://www.sciencedirect.com/science/article/pii/S0957417416306947.

Barddal, J. P., & Enembreck, F. (2019). Learning regularized hoeffding trees from data streams. In *SAC '19, Proceedings of the 34th ACM/SIGAPP symposium on applied computing* (pp. 574–581). New York, NY, USA: ACM, http://dx.doi.org/10.1145/3297280.3297334, URL: http://doi.acm.org/10.1145/3297280.3297334.

Barddal, J. P., Gomes, H. M., Enembreck, F., & Pfahringer, B. (2017). A survey on feature drift adaptation: Definition, benchmark, challenges and future directions. *Journal of Systems and Software*, *127*, 278–294. http://dx.doi.org/10.1016/j.jss.2016.07.005.

Bifet, A. (2009). Adaptive learning and mining for data streams and frequent patterns. *SIGKDD Explorations Newsletter*, *11*(1), 55–56. http://dx.doi.org/10.1145/1656274.1656287, URL: http://doi.acm.org/10.1145/1656274.1656287.

Bifet, A., & Gavaldà, R. (2009). *Adaptive learning from evolving data streams* (pp. 249–260). Berlin, Heidelberg: Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-642-03915-7_22.

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). Moa: Massive online analysis. *Journal of Machine Learning Research*, *11*, 1601–1604, URL: http://dl.acm.org/citation.cfm?id=1756006.1859903.

Bifet, A., Holmes, G., & Pfahringer, B. (2010). Leveraging bagging for evolving data streams. In J. L. Balcázar, F. Bonchi, A. Gionis, & M. Sebag (Eds.), *Machine learning and knowledge discovery in databases* (pp. 135–150). Berlin, Heidelberg: Springer Berlin Heidelberg.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140. http://dx.doi.org/10.1023/A:1018054314350.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Cano, A., & Krawczyk, B. (2020). Kappa updated ensemble for drifting data stream mining. *Machine Learning*, *109*, 175–218. http://dx.doi.org/10.1007/s10994-019-05840-z.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*(1), 321–357.

Chen, M., Dautais, Y., Huang, L., & Ge, J. (2017). Data driven credit risk management process: A machine learning approach. In *ICSSP 2017, Proceedings of the 2017 international conference on software and system process* (pp. 109–113). New York, NY, USA: ACM, http://dx.doi.org/10.1145/3084100.3084113, URL: http://doi.acm.org/10.1145/3084100.3084113.

Chen, W., & Shi, L. (2013). Credit scoring with f-score based on support vector machine. In *Proceedings 2013 international conference on mechatronic sciences, electric engineering and computer (MEC)* (pp. 1512–1516). http://dx.doi.org/10.1109/MEC.2013.6885307.

Corder, G., & Foreman, D. (2011). *Nonparametric statistics for non-statisticians: A step-by-step approach*. Wiley, URL: http://books.google.com.br/books?id=T3qOqdpSz6YC.

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, 215–242.

Crook, J. N., Thomas, L. C., & Hamilton, R. (1992). The degradation of the scorecard over the business cycle. *IMA Journal of Management Mathematics*, *4*(1), 111–123. http://dx.doi.org/10.1093/imaman/4.1.111.

Domingos, P., & Hulten, G. (2000). Mining high-speed data streams. In *KDD '00, Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 71–80). New York, NY, USA: ACM, http://dx.doi.org/10.1145/347090.347107, URL: http://doi.acm.org/10.1145/347090.347107.

Ferreira, L. B., Heitor Murilo Gomes, A. B., & de Oliveira, L. S. (2019). Adaptive random forests with resampling for imbalanced data streams. (pp. 1–6). http://dx.doi.org/10.1109/IJCNN.2019.8852027.

Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G., & Abdessalem, T. (2017). Adaptive random forests for evolving data stream classification. *Machine Learning*, *106*(9), 1469–1495. http://dx.doi.org/10.1007/s10994-017-5642-8.

Gomes, H. M., Read, J., Bifet, A., Barddal, J. P., & Gama, J. a. (2019). Machine learning for streaming data: State of the art, challenges, and opportunities. *SIGKDD Explorations Newsletter*, *21*(2), 6–22. http://dx.doi.org/10.1145/3373464.3373470.

Guo, Y., & Dong, C. (2017). A novel intelligent credit scoring method using mopso. In *2017 29th Chinese control and decision conference (CCDC)* (pp. 6584–6588). http://dx.doi.org/10.1109/CCDC.2017.7978359.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explorations Newsletter*, *11*(1), 10–18. http://dx.doi.org/10.1145/1656274.1656278, URL: http://doi.acm.org/10.1145/1656274.1656278.

Hand, D. J., & Adams, N. M. (2014). Selection bias in credit scorecard evaluation. *The Journal of the Operational Research Society*, *65*(3), 408–415. http://dx.doi.org/10.1057/jors.2013.55.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(8), 832–844. http://dx.doi.org/10.1109/34.709601.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, *58*(301), 13–30, URL: http://www.jstor.org/stable/2282952?.

Jung, K. M., Thomas, L. C., & So, M. C. (2015). When to rebuild or when to adjust scorecards. *The Journal of the Operational Research Society*, *66*(10), 1656–1668. http://dx.doi.org/10.1057/jors.2015.43, arXiv:https://doi.org/10.1057/jors.2015.43.

Karakoulas, G. (2004). Empirical validation of retail credit-scoring models. *The RMA Journal*, *87*(1), 56–60.

Karax, J. A., Barddal, J. P., & Malucelli, A. (2019). Decision tree-based feature ranking in concept drifting data streams. In *SAC '19, Proceedings of the 33rd annual ACM symposium on applied computing, SAC 2019, Limassol, Cyprus, April 08-12, 2019*.

Kennedy, K., Namee, B. M., & Delany, S. J. (2013). Using semi-supervised classifiers for credit scoring. *The Journal of the Operational Research Society*, *64*(4), 513–529. http://dx.doi.org/10.1057/jors.2011.30, arXiv:https://doi.org/10.1057/jors.2011.30.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, *5*(4), 221–232. http://dx.doi.org/10.1007/s13748-016-0094-0, URL: https://doi.org/10.1007/s13748-016-0094-0.

Kritzinger, N. (2017). *Improving credit risk measurement and management: A new application of statistical techniques* (Ph.D. thesis), North-West University.

Lawi, A., Aziz, F., & Syarif, S. (2017). Ensemble gradientboost for increasing classification accuracy of credit scoring. In *2017 4th international conference on computer applications and information processing technology (CAIPT)* (pp. 1–4). http://dx.doi.org/10.1109/CAIPT.2017.8320700.

Li, W., & Liao, J. (2011). An empirical study on credit scoring model for credit card by using data mining technology. In *2011 seventh international conference on computational intelligence and security* (pp. 1279–1282). http://dx.doi.org/10.1109/CIS.2011.283.

Loezer, L., Fabricio Enembreck, J. P. B., & Britto, A. (2020). Cost-sensitive learning for imbalanced data streams. (pp. 498–504). http://dx.doi.org/10.1145/3341105.3373949.

Louppe, G., Wehenkel, L., Sutera, A., & Geurts, P. (2013). Understanding variable importances in forests of randomized trees. In *NIPS'13, Proceedings of the 26th international conference on neural information processing systems - volume 1* (pp. 431–439). Red Hook, NY, USA: Curran Associates Inc..

Mays, E., & Mays, F. (2004). *Credit scoring for risk managers: The handbook for lenders*. Thomson/South-Western, URL: https://books.google.com.br/books?id=AVl2QgAACAAJ.

Melo, L., Nardini, F. M., Renso, C., & Macedo, J. A. (2019). Knora-iu: Improving the dynamic selection prediction in imbalanced credit scoring problems. In *2019 IEEE 31st international conference on tools with artificial intelligence (ICTAI)* (pp. 424–431). http://dx.doi.org/10.1109/ICTAI.2019.00066.

Nalić, J., & Svraka, A. (2018). Using data mining approaches to build credit scoring model: Case study - implementation of credit scoring model in microfinance institution. In *2018 17th international symposium INFOTEH-JAHORINA (INFOTEH)* (pp. 1–5). http://dx.doi.org/10.1109/INFOTEH.2018.8345543.

Okesola, O. J., Okokpujie, K. O., Adewale, A. A., John, S. N., & Omoruyi, O. (2017). An improved bank credit scoring model: A naive bayesian approach. In *2017 international conference on computational science and computational intelligence (CSCI)* (pp. 228–233). http://dx.doi.org/10.1109/CSCI.2017.36.

Oza, N. C. (2005). Online bagging and boosting. In M. Jamshidi (Ed.), *International conference on systems, man, and cybernetics, special session on ensemble methods for extreme environments* (pp. 2340–2345). New Jersey: Institute for Electrical and Electronics Engineers.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research (JMLR)*, *12*, 2825–2830.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc..

Rezac, M., & Rezac, F. (2011a). How to measure the quality of credit scoring models. *Czech Journal of Economics and Finance (Finance a Uver)*, *61*(5), 486–507, URL: https://EconPapers.repec.org/RePEc:fau:fauart:v:61:y:2011:i:5:p:486-507.

Rezac, M., & Rezac, F. (2011b). How to measure the quality of credit scoring models. *Czech Journal of Economics and Finance (Finance a Uver)*, *61*(5), 486–507, URL: https://ideas.repec.org/a/fau/fauart/v61y2011i5p486-507.html.

Saia, R., & Carta, S. (2017). A fourier spectral pattern analysis to design credit scoring models. In *IML '17*, *Proceedings of the 1st international conference on internet of things and machine learning* (pp. 18:1–18:10). New York, NY, USA: ACM, http://dx.doi.org/10.1145/3109761.3109779, URL: http://doi.acm.org/10.1145/3109761.3109779.

Singh, P. (2017). Comparative study of individual and ensemble methods of classification for credit scoring. In *2017 international conference on inventive computing and informatics (ICICI)* (pp. 968–972). http://dx.doi.org/10.1109/ICICI.2017.8365282.

Soares de Melo Junior, L., Nardini, F. M., Renso, C., & Fernandes de Macêdo, J. A. (2019). An empirical comparison of classification algorithms for imbalanced credit scoring datasets. In *2019 18th IEEE international conference on machine learning and applications (ICMLA)* (pp. 747–754). http://dx.doi.org/10.1109/ICMLA.2019.00133.

Yeh, I.-C., & Lien, C.-h. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, *36*(2), 2473–2480. http://dx.doi.org/10.1016/j.eswa.2007.12.020, http://dx.doi.org/10.1016/j.eswa.2007.12.020.

Žliobaitė, I., Pechenizkiy, M., & Gama, J. (2016). An overview of concept drift applications. In N. Japkowicz, & J. Stefanowski (Eds.), *Big data analysis: New algorithms for a new society* (pp. 91–114). Cham: Springer International Publishing, http://dx.doi.org/10.1007/978-3-319-26989-4_4.