A novel approach to define the local region of dynamic selection techniques in imbalanced credit scoring problems

Leopoldo Melo Junior[a,*], Franco Maria Nardini[b], Chiara Renso[b], Roberto Trani[b], Jose Antonio Macedo[a]

[a] *Federal University of Ceará, Fortaleza, Brazil*
[b] *ISTI-CNR, Pisa, Italy*

Abstract

Lenders, such as banks and credit card companies, use credit scoring models to evaluate the potential risk posed by lending money to customers, and therefore to mitigate losses due to bad credit. The profitability of the banks thus highly depends on the models used to decide on the customer's loans. State-of-the-art credit scoring models are based on machine learning and statistical methods. One of the major problems of this field is that lenders often deal with imbalanced datasets that usually contain many paid loans but very few not paid ones (called *defaults*). Recently, dynamic selection methods combined with ensemble methods and preprocessing techniques have been evaluated to improve classification models in imbalanced datasets presenting advantages over the static machine learning methods. In a dynamic selection technique, samples in the neighborhood of each query sample are used to compute the local competence of each base classifier. Then, the technique selects only competent classifiers to predict the query sample. In this paper, we evaluate the suitability of dynamic selection techniques for credit scoring problem, and we present Reduced Minority k-Nearest Neighbors (RMkNN), an approach that enhances state of the art in defining the local region of dynamic selection techniques for imbalanced credit scoring datasets. This proposed technique has a superior prediction performance in imbalanced credit scoring datasets compared to state of the art. Furthermore, RMkNN does not need any preprocessing or sampling method to generate the dynamic selection dataset (called DSEL). Additionally, we observe an equivalence between dynamic selection and static selection classification. We conduct a comprehensive evaluation of the proposed technique against state-of-the-art competitors on six real-world public datasets and one private one. Experiments show that RMkNN improves the classification performance of the evaluated datasets regarding AUC, balanced accuracy, H-measure, G-mean, F-measure, and Recall.

Keywords:
Credit scoring, Imbalanced learning, Dynamic Selection Classification

1. Introduction

Credit offer is a key activity for banks that aim at improving their profitability and competitiveness. Small improvements in the default prediction imply significant profits to the financial institutions Hand & Henley (1997). However, the decision to grant a loan to a customer is complex and risky because it requires an accurate default prediction to protect banks from financial losses, especially during the financial crises. Thomas et al. (2017) pointed out several aspects affecting the default rate over time, such as the cost of the money (interest rate), the supply and demand for credit, the state of the economy, and the cyclical variations of credit over time. Besides these aspects, data availability, accuracy, and reliability make the default prediction much harder than other domain-specific classification problems. Therefore, new methods and techniques, called credit scoring models, are required to cope with these problems while guaranteeing a low percentage of defaults.

Basel accords regulate credit scoring. Basel Capital Accord II defines the validation and verification of three estimates: the probability of default (PD), loss given default (LGD), and exposure at default (EAD) Thomas et al. (2005). In this paper, as in Feng et al. (2018), we look for improvements in the estimate of the probability of default (PD) in existing historical loan data.

Available historical loan data creates an excellent opportunity to take advantage of trending machine learning methods for building accurate credit scoring models. However, in the real world, credit scoring datasets are imbalanced and skewed data is a challenge for machine learning methods since classifiers tend to predict only the majority class.

In the past few decades, researchers have attempted to optimize the predictive performance in imbalanced data. According to Haixiang et al. (2017), the two most used approaches are *Resampling*, a kind of preprocessing technique that changes the class distribution of the training set, and *Ensemble methods*, which could combine several base classifiers, resampling, and cost-sensitive approaches. Ensemble methods, also known as Multiple Classifier Systems (MCS), are justified by several theoretical Kuncheva (2002); Dietterich (2000) and empirical

[*] Corresponding author
Email addresses: leopoldosmj@lia.ufc.br (Leopoldo Melo Junior), francomaria.nardini@isti.cnr.it (Franco Maria Nardini), chiara.renso@isti.cnr.it (Chiara Renso), roberto.trani@isti.cnr.it (Roberto Trani), jose.macedo@insightlab.ufc.br (Jose Antonio Macedo)

Fernández-Delgado et al. (2014); Opitz & Maclin (1999) studies which demonstrate the superiority of ensembles over individual classifier models. They are widely used to solve many real-world problems, including credit scoring Lessmann et al. (2015); Xiao et al. (2016), and class imbalance Galar et al. (2012).

An ensemble is typically composed of three phases: (1) pool generation, (2) selection of base classifiers, and (3) integration of predictions Britto Jr et al. (2014). The main target of the pool generator phase is generating diverse classifiers. The selection phase is responsible for selecting the most competent classifiers to perform the prediction, while the integration phase is responsible for the fusion of the results of all models in the ensemble prediction.

The selection phase can be static or dynamic. The static selection consists of selecting the base models once and use the resulting ensemble to predict all the test samples. In the dynamic selection, the base classifier's competence in the neighborhood of the query sample is used to select which base models are used to predict each sample. Recently, dynamic selection has received attention from the academic community.

It is worth noticing that, as highlighted by Lessmann et al. (2015), dynamic selection classification techniques might violate regulatory requirements in credit scoring because they use different scorecards for different customers. The motivation for this regulation constraint is to avoid customer discrimination. We believe that this regulatory requirement can change if some work confirms that the use of dynamic classification does not include any customer discrimination. However, the demonstration of the lack of correlation between dynamic classification and customer discrimination is not the aim of this paper. Additionally, the existence of other papers evaluating dynamic classification to credit scoring problem Feng et al. (2018); Ala'raj & Abbod (2016a,b) encourages us to explore this topic.

Analyzing recent works about credit scoring problem, we find several papers that evaluate the prediction performance of classification approaches for credit scoring datasets, such as García et al. (2019); Feng et al. (2018); He et al. (2018); Sun et al. (2018); Xia et al. (2018); Abellán & Castellano (2017); Xia et al. (2017); Ala'raj & Abbod (2016b); Xiao et al. (2016); Ala'raj & Abbod (2016a); Lessmann et al. (2015). However, to the best of our knowledge, a combination of preprocessing approaches, dynamic selection techniques, and pool generators ensembles is presented only in Melo Jr et al. (2019b). Nonetheless, this previous work only compares the combination of techniques.

Beyond this gap in credit scoring papers, we do not find scientific papers evaluating the suitability of dynamic selection techniques to credit scoring problem. However, Britto Jr et al. (2014) concluded that dynamic selection is appropriate for complex datasets. This motivates us to evaluate the complexity of credit scoring datasets in comparison to datasets of other domains. If credit data is more complex than the average, this can suggest that dynamic selection may be appropriate for this domain.

This work is partially motivated by the outstanding results achieved recently by the dynamic selection techniques Roy et al. (2018); Britto Jr et al. (2014). Roy et al. (2018) proposes the use of a bagging pool generator combined with oversampling techniques to reduce the effects of the skewed data. Furthermore, although the literature recommends the use of different data in the dynamic selection dataset (DSEL) and training data, they also use oversampling techniques over the training data to generate the DSEL. They decided to use this approach to avoid the lack of minority samples in the training data and the DSEL. The authors rely on the diversity ability of oversampling techniques to avoid bias results.

However, in this paper, instead of using oversampling techniques to guarantee the diversity between the training dataset and DSEL and to overcome the skewed data, we adopt different strategies for each issue. Next, we explain the problems of the previous approach and our strategies.

The main problem of use oversampling techniques to balance the DSEL is the inclusion of noise in the dataset. Oversampling techniques have been shown to be useful for building imbalanced prediction models in the last fifteen years Fernández et al. (2018). However, the DSEL function needs to determine the competence of the base models, and this means that a noise sample in the DSEL can produce miscalculated competence of the base classifiers. That is the reason why we decided to evaluate new approaches.

To address the diversity between training and DSEL, we use bootstrapping: the use of random sampling with replacement. We evaluate this method since it always uses a subset of available samples to train each base classifier. In this way, each classifier does not know the entire training data. We believe that this characteristic is sufficient to guarantee diversity when using the same dataset to train the base models and as the DSEL.

To address the skewed of the data in the DSEL, we develop a modification in the k-NN algorithm, named Reduced Minority k-NN (RMkNN), used by the dynamic selection techniques to define the local region of a query sample. Dynamic selection techniques use k-NN to select the samples in the DSEL that define the competence level of each base classifier. However, in an imbalanced dataset, the k-NN algorithm selects mainly samples of the majority class, producing a poor base classifiers competency evaluation.

To evaluate the performance of RMkNN, we extend Melo Jr et al. (2019b) comparison, including other combinations of pool generators and preprocessing techniques and testing them on seven datasets. We evaluate several combinations of dynamic selection techniques, sampling approaches, and pool generators to assess the effectiveness of our proposal. More specifically, we aim to answer the following research questions related to the credit scoring problem:

- **RQ1)** Are dynamic selection techniques appropriate for imbalanced credit scoring problems?
- **RQ2)** Does the RMkNN improve the prediction performance of kNN?
- **RQ3)** Does the use of the RMkNN technique - that defines a novel competence region of dynamic selection

techniques - improve the classification performance of imbalanced credit scoring datasets?

To answer the questions above, we present and discuss novel contributions that include:

- a novel local competence definition for imbalance dynamic selection classification.
- an evaluation of the classification complexity of credit scoring datasets in comparison to the classification complexity of datasets from other domains.
- an evaluation of the prediction performance of RMkNN in comparison with the regular kNN.
- a static selection representation of dynamic selection techniques.

Paper Melo Jr et al. (2019b) evaluated the combination of pool generators, re-sampling approaches, and dynamic selection techniques. That paper shows a simple comparison of the influence of different re-sampling and dynamic selection techniques on the performance of pool generators in imbalanced credit scoring datasets, in contrast to the present paper where we propose three new research questions and introduce a new method to identify a new local region of the query sample to define the competence of each base classifier in a dynamic selection approach.

The organization of this paper is as follows. Section 2 reviews the literature about credit scoring, imbalance learning approaches, and dynamic selection techniques. Section 3 includes a brief description of the classification approaches used in this paper. Section 4 presents the first contribution of this paper that is an evaluation of the suitability of dynamic selection for credit scoring problems. This section also presents the main contribution of this paper that is the Reduced Minority k-NN (RMkNN) technique. Section 5 presents the experimental setup used. Section 6 shows the experimental results. Finally, the last section is dedicated to the conclusion and future work. The online appendix[1] provides details of the results.

2. Background and related work

This study involves four main elements: credit scoring, imbalanced learning, pool generators, and dynamic selection classification. Next, we present the credit scoring related works and the background of pool generators, imbalanced learning, and dynamic selection classification.

2.1. Credit scoring related works

Several works have been published in last years using ensembles focusing on default loan prediction. Table 1 shows a summary of studies in the literature on classifier ensembles used for credit scoring from 2015 to 2019. The comparison contains the number of datasets used, the percentage of datasets with imbalance ratio (IR), the cardinality of the majority class divided by the cardinality of the minority class, under 3, sampling approaches used, column *Sampling*, whether the developed classifier ensembles are homogeneous or heterogeneous, column *Kind*, the type of selection, static (SS), or dynamic (DS), and the pool generators adopted. As can be seen, only Melo Jr et al. (2019b) evaluate the combination of dynamic selection, preprocessing techniques, and different pool generators for credit scoring datasets.

Another important aspect of this comparative analysis is the percentage of low imbalanced datasets of each study, with $IR \leq 3$. Except for Melo Jr et al. (2019b), the percentage of low imbalanced datasets of all reviewed papers is at least 50%. We believe that a high number of low imbalanced datasets can produce a bias evaluation of an imbalanced learning approach.

Besides these ensemble-based solutions, Serrano-Cinca & Gutiérrez-Nieto (2016) evaluate an alternative approach to traditional credit scoring. In this paper, the authors use the internal rate of return (IRR) to train models, instead of the binary concept of default loan. It is an interesting strategy, once the main target of most financial institutions is the profit.

Bastani et al. (2019) also evaluate the profitability of a credit scoring problem instead of evaluating only the probability of default. The authors define a two-stage approach to evaluate the probability of default combined with the profitability using a wide and deep learning strategy.

All techniques cited above, even the dynamic selection classification techniques, are based on a static learning setting. It means that a static dataset is used to build the prediction model. Different from static learning, dynamic models update the model periodically with new data available. Sousa et al. (2016) define a modeling framework for credit risk assessment and observe that dynamic modeling outperforms the static models. Despite that, static models are widely more used than dynamic ones in the credit scoring field.

2.2. Background

2.2.1. Imbalanced learning approaches

As mentioned in Section 1, the prediction task in credit scoring datasets suffers from the lack of sufficient samples of the minority class, the defaulters. Haixiang et al. (2017) defined four categories of techniques for handling class imbalance: (1) modify the data distribution, called *preprocessing solutions*; (2) apply different costs to misclassification of positive and negative samples, the *cost-sensitive solutions*. (1) and (2) are "basic strategies" for imbalanced learning. (3) and (4) are "classification algorithms": (3) adapts a classifier to deal with the class imbalance, the *algorithm level solutions*; and (4), *ensemble-based solutions*, combines the previous solutions using an ensemble. We describe the two most common imbalanced approaches briefly in the following paragraphs, preprocessing and ensemble-based.

Preprocessing comes before the learning phase. Resampling, the most common preprocessing technique, is used to balance the sample space for an imbalanced dataset to reduce

[1] Supplementary material associated with this article can be found in the online version, at ...

Table 1: Approaches tracking credit scoring in literature

Ref.	Year	# datasets	% datasets w/ $IR \leq 3$	Sampling	Ensemble		
					Kind	Selection	Pool generators
Melo Jr et al. (2019b)	2019	4	0.25	RUS, SMOTE, RAMO	Homog.	SS, DS	(a)
García et al. (2019)	2019	14	0.50	-	Homog.	-	(b)
Feng et al. (2018)	2018	10	0.60	-	Heterog.	DS	Bagging and different parameters
He et al. (2018)	2018	6	0.50	Based on RUS	Heterog.	SS	Based on bagging
Sun et al. (2018)	2018	1	1.00	SMOTE	Homog	SS	Bagging
Xia et al. (2018)	2018	4	0.75	-	Heterog.	-	Based on bagging
Abellán & Castellano (2017)	2017	6	0.67	-	Homog.	-	(c)
Xia et al. (2017)	2017	5	0.80	-	Homog.	-	Based on boosting
Xiao et al. (2016)	2016	2	1.00	-	Heterog.	DS	Bagging based on clustering
Ala'raj & Abbod (2016b)	2016	7	0.71	-	Heterog.	DS	Feature selection based
Ala'raj & Abbod (2016a)	2016	5	0.80	-	Heterog.	DS	Bagging
Lessmann et al. (2015)	2015	8	0.75	-	Both	SS, DS	(d)

(a) Bagging, Boosting, Random Forest, Rotation Forest, Hybrid
(b) Bagging, Boosting, Random subspace, Random Forest, Rotation Forest, DECORATE
(c) Bagging, Boosting, Random Subspace, DECORATE, Rotation Forest
(d) Bagging, Boosting, Random Forest, Rotation Forest

the skewed class distribution in the learning process. There are three possible methods to do it, oversampling, undersampling, and hybrid. The first one is over-sampling, which consists of creating new minority class samples synthetically. We test the widely used method, Synthetic Minority Over-sampling Technique (SMOTE), Chawla et al. (2002). The second one is under-sampling, which consists of removing samples from the majority class. We test the most used method, Random Under-sampling (RUS) Barandela et al. (2003). The hybrid methods combine the two previous ones.

Ensemble approaches to imbalanced learning consist of combining preprocessing, cost-sensitive, and classifier algorithm modifications. They combine the power of an ensemble with the ability of other imbalanced techniques to overcome the imbalance issue.

2.2.2. *Pool generators*

A typical ensemble, also known as multiple classifier systems (MCS), has the following phases: the pool generation, the selection, and the integration. The main challenge of the pool generation phase is to generate a pool of accurate and diverse classifiers. Homogeneous or heterogeneous base classifiers can achieve this diversification. Regarding the homogeneous pools, the diversity comes from different subsets of training data (Bagging, Boosting, or Hybrid), or using different features subspaces (Random Subspace Selection), or based on feature extraction (Rotation Forest).

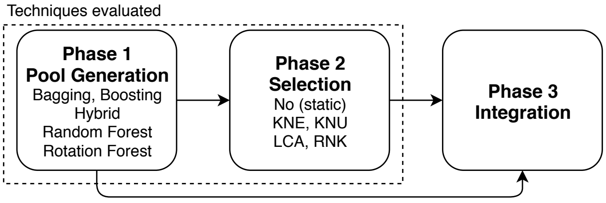

Figure 1: The three MCS phases and the techniques evaluated in this work.

2.2.3. *Dynamic selection*

The second phase of an MCS is the selection of base classifiers to the prediction procedure, as shown in Figure 1. The main concepts of this phase are related to the type of selection and the notion of classifier competence (ability to predict correctly). The type of selection may be static, where the decision about the competence of the base learners occurs at the fitting time; or dynamic, when the decision occurs at prediction time. The intuition behind the preference for dynamic over static selection is to select the most locally-accurate classifiers for each unknown sample. A dynamic selection approach defines competence measures, mostly related to the classifier accuracy in some part of the feature space, and a procedure to select the best estimators.

The dynamic selection approaches are classified by the selection methodology. According to this classification, there are two kinds of strategies: dynamic classifier selection (DCS) and dynamic ensemble selection (DES). The difference between them is the number of classifiers selected to predict each sample. The DCS selects only the most competent base classifier, and the DES selects a set of competent classifiers.

Roy et al. (2018) is the most recent work that evaluated dynamic selection techniques to solve imbalance classification problems. They test DS strategies based on different notions of competence measure as previous papers that evaluated dynamic selection in the context of imbalanced learning Xiao et al. (2012). For example, Local Class Accuracy (LCA) considers the local class accuracy separately. The Modified Classifier Rank (RNK) ranks the classifiers. These two techniques are DCS. They also test two versions of K-Nearest Oracles (KNORA), which are Dynamic Ensemble Selection (DES) techniques. Next, we briefly describe the four DS strategies obtained from Cruz et al. (2018) and adopted in this paper.

- The Local Class Accuracy (LCA) Woods et al. (1997); Britto Jr et al. (2014) gets the prediction of the test sample of each base classifier and, according to the predicted class, compute the class accuracy regarding only the predicted class. The classifier with the higher class accuracy is used to predict the test sample.

- The Modified Classifier Rank (RNK) Sabourin et al. (1993); Britto Jr et al. (2014) method ranks the accuracy of the base classifiers in the neighborhood for each test instance. The classifier with the highest accuracy is used to predict the test instance.

- The K-Nearest Oracles (KNORA) Ko et al. (2008) techniques are inspired by the Oracle Kuncheva (2002) concept. Among them, the most promising are KNORA-Eliminate (KNE) and KNORA-Union (KNU). The KNE selects only the base classifiers with the perfect accuracy in the neighborhood of the test instance. On the other hand, in the KNU technique, the level of competence of a base classifier is measured by the number of correctly classified instances in the defined region of competence. In this case, every classifier that correctly classified at least one instance can vote.

The dynamic selection approaches require a dynamic selection dataset (DSEL) to define the local regions of the feature space. This data is used to measure the competence of the base classifiers on each part of the feature space. The main challenge in the DSEL generation is to use a reasonable part of the training data obtaining good performance of the DS approach and keep the other part for the training the base classifiers. The separation between the training data and the DSEL is essential to avoid overfitting. In an imbalanced dataset, this task is even more difficult due to the lack of samples in the minority class. Roy et al. (2018) work around this problem applying oversampling techniques to generate the DSEL.

The integration is the last step of an MCS, and it consists of applying the selected classifiers to recognize a given testing pattern. In cases where all classifiers are used (without selection) or when a subset is selected, a fusion strategy is necessary. Majority voting is the most common fusion approach used by ensembles. Section 4 evaluates the classification complexity and performance measures for credit scoring problems. Next, we briefly describe the classification approaches used in this paper.

3. Overview of classification techniques

This study aims to evaluate the performance of a novel dynamic selection approach for imbalanced credit scoring datasets over a wide range of classification techniques. For the propose of this study, two sampling approaches, four credit scoring benchmarks, and eight imbalanced ensembles have been selected based on previous credit scoring papers Brown & Mues (2012); Melo Jr et al. (2019a).

3.1. Credit Scoring Benchmarks

Our credit scoring benchmarks list starts with Logistic regression. This binary classifier is a trendy statistical model in commercial credit scoring. It models the relationship between independent variables and the response variable using a logistic function.

The support vector machine method constructs a hyperplane to split the two classes of borrowers. In this paper, we use the linear version of SVM, the linear support vector machine (LSVM), and also non-linear kernels of SVM, such as *poly* and rbf.

The next classification approach is multilayer perceptron artificial neural networks (ANN). It employs sigmoidal functions to determine the model parameters by minimizing some loss-function. We consider ANNs with logistic activation function in the hidden and output layer.

We also include the Random Forest ensemble Liaw et al. (2002) in our credit scoring benchmarks list. This ensemble is an extension of bagging that uses decision trees as base classifiers and samples the features used by each decision tree.

The last classifier in the credit scoring benchmarks class is extreme gradient boosting Chen & Guestrin (2016). This ensemble is a novel implementation method for the gradient boosting machine, and it aims to prevent over-fitting.

3.2. Imbalanced ensembles

An imbalanced ensemble is an ensemble designed that uses some sampling technique to balance the data before the base learners training step. We use the imbalanced ensembles available on Lemaître et al. (2017) and implement others.

We start with the bagging ensemble Breiman (1996). Bagging, also known as Bootstrap aggregating, constructs bootstrap samples from the training data to produce T base models. Bagging uses a majority voting to fusion the T predictions. The first imbalanced ensemble we use is Balanced Bagging (BBAG). It includes an additional step to balance the training set using random undersampling (RUS). We also use Bagging SMOTE (BGSM), the bagging ensemble, with a SMOTE step to balance the training set.

We also use two imbalanced ensembles using Random Forest. We use Balanced Random Forest (BRND) Chen et al. (2004), the combination of this ensemble with RUS, and the Random Forest SMOTE, the combination of Random Forest with SMOTE oversampling technique.

Next, we test an imbalanced ensemble that uses the rotation forest. This ensemble applies principal component analysis on bootstrap samples to rotate the training data. Based on BRND, we develop a balanced rotation forest (BROT), the combination of rotation forest and RUS, to execute the experiments of this work.

The next three imbalanced ensembles evaluated are derived from adaptive boosting, also known as AdaBoost. Two ensembles are the combination of AdaBoost and the RUS and SMOTE preprocessing techniques. They are RUSBoost (RUSB), and SMOTEBoost (SMTB), respectively. They use the preprocessing technique to balance the data in each step of the boosting algorithm. The last imbalance ensemble, easy ensemble (EASY) Liu et al. (2009), is a bagging ensemble that uses AdaBoost ensemble as base classifiers. It also uses RUS to balance the data before the training step.

Table 2 shows the list of evaluated combinations: **(I)** contains our proposal of modification of k-Nearest Neighbors to select balanced samples of the DSEL; **(II)** contains preprocessing techniques (SMOTE, and RUS) to balance the DSEL; **(III)** contains the imbalanced ensembles strategies; **(IV)** lists the dynamic selection techniques evaluated; and **(V)** lists the credit scoring benchmarks evaluated.

The combination of the pool generators and preprocessing approaches occurs as follows. In all imbalanced ensembles that use RUS, each base classifier receives a subset of the dataset with the same number of samples in each class. In the boosting ensembles combined with SMOTE, we double the number of samples of the minority class in each boost iteration. For Bagging and Random Forest ensembles combined with SMOTE, we make the equal size of both classes for each iteration, as done in Roy et al. (2018).

In this paper, we test these classification approaches. Next, we present the Reduced Minority kNN (RMkNN) algorithm and its application in imbalanced dynamic selection classification.

4. Reduced Minority kNN for dynamic selection techniques

The main purpose of the dynamic selection dataset (DSEL) in a dynamic selection technique is to introduce the measurement of the competency level of each classifier in each part of the feature space. These parts are called the *local regions*. The neighbors of a query sample define a local region, and k-Nearest Neighbors(kNN) is used to find them. These samples are used to evaluate the competence of each base classifier of the ensemble. Finally, the prediction procedure uses only the most competent classifiers.

This approach works fine in a balanced DSEL. However, the use of k-NN in an imbalanced DSEL returns almost always the samples of the majority class. This behavior is not desirable because the measure of competence of the base classifiers considers mainly the majority class.

Nevertheless, instead of using sampling techniques to generate the DSEL, this paper evaluates a modification in the k-NN procedure to try to balance the set of neighbors used to measure the competence of the base classifiers. The main idea is to reduce the distance of the minority samples from the predicted instance.

4.1. Suitability of dynamic selection for credit scoring

Before evaluating the improvements of dynamic selection techniques to credit scoring datasets, we analyze whether the dynamic selection classification is appropriate to credit scoring datasets. As pointed out by Britto Jr et al. (2014), the performance of dynamic selection techniques is related to the classification complexity of the datasets. Considering this, we decide to evaluate the complexity of credit scoring datasets.

To perform the study, we evaluate the twelve complexity measures presented by Ho (1995). However, some complexity measures to binary classification have bias results for imbalanced datasets. For instance, the measure of Error Rate for 1NN Classifier (N3) tends to be low in high imbalanced datasets. Finally, we choose two less influenced by the imbalanced ratio of the dataset, Maximum Fisher's Discriminant Ratio (F1), and Ratio of average intra/inter-class NN distance (N2). Next, we briefly describe the F1 and N2 measures.

1. *Fisher's Discriminant Ratio (F1)*: This is a class overlapping measure computed over every single feature as denoted in Eq. 1. In this Equation, f_a is the Fisher's Discriminant Ratio of feature a, and μ_{a1}, μ_{a2}, σ^2_{a1}, σ^2_{a2} are the means and the variances of the two classes, respectively. For a multidimensional problem, not necessarily all features have to contribute to class discrimination. As long as there exists one discriminating feature, the problem is easy. Therefore, F1 is the maximum of f_i over all the feature dimensions.

$$f_a = \frac{(\mu_{a1} - \mu_{a2})^2}{\sigma^2_{a1} + \sigma^2_{a2}} \tag{1}$$

2. *Ratio of Average Intra/Inter class NN distance (N2)*: This is a nonparametric separability of classes measure. It compares the intraclass dispersion with the interclass separability, as denoted in Eq. 2. In this equation, let $n_1^{intra}(s_i)$ and $n_1^{inter}(s_i)$ denote the intra and inter-class nearest neighbors of the sample s_i, while δ represents the the Euclidian distance. $N2$ calculates the ratio between the intra and inter-class dispersions. A small $N2$ value suggests high separability, and consequently, an easier classification problem.

$$N2 = \frac{\sum_i^N \delta(n_1^{intra}(s_i), s_i)}{\sum_i^N \delta(n_1^{inter}(s_i), s_i)} \tag{2}$$

Table 2: Techniques evaluated.

Label	Type	Acronym	Method
(I)	Reduced Minority k-NN	RMkNN	Modified kNN that reduce the distance of the minority class samples
(II)	Imbalance Preprocessing	SMTE	Synthetic Minority Over-sampling Technique
		RUS	Random under-sampling
(III)	Imbalanced Ensembles (Pool generator + sampling)	BBAG	Balanced Bagging (Bagging + RUS)
		BGSM	Bagging SMOTE (Bagging + SMOTE)
		BRND	Balanced Random Forest (Random Forest + RUS)
		RFSM	Random Forest SMOTE (Random Forest + SMOTE)
		BROT	Balanced Rotation Forest (Rotation Forest + RUS)
		RUSB	RUS Boost (AdaBoost + RUS)
		SMTB	SMOTE Boost (AdaBoost + SMOTE)
		EASY	Easy ensemble (Bagging of AdaBoost + RUS)
(IV)	Dynamic Selection	KNE	k-Nearest Oracles-Eliminate
		KNU	k-Nearest Oracles-Union
		LCA	Local Class Accuracy
		RNK	Modified Classifier Rank
(V)	Credit Scoring Benchmarks	LOGR	Logistic Regression
		XGB	eXtreme Gradient Boosting
		ANN	Airtificial Neural Networks
		LSVM	Linear Support Vector Machine
		SVM	Support Vector Machine
		RNDF	Random Forest

4.2. *The Reduced Minority k-NN algorithm*

To evaluate the dynamic selection approaches with imbalanced datasets using DSELs without sampling techniques, we develop a modification in the k-NN algorithm shown in Algorithm 1. The intuition is to reduce the distance of the minority class samples from the predicted sample in the k-NN computation. The first step of Algorithm 1 is to separate the samples of each class, lines 2 and 3. After, we compute the imbalance ratio of the dataset, line 4. Next, we compute the k nearest neighbors and their distances from sample query, s_q, for each class, lines 5 and 6. After, we reduce the distances of the minority class samples using the *distance_reduction_function*. The next step is to concatenate the indexes and distances of minority and majority samples, lines 8 and 9. On line 10, we compute the indexes of the k shortest distances of D and return on line 11 the distances and the indexes of the nearest neighbors.

We present now a simple example of the modified k-NN in Figure 2. This figure shows a DSEL where the majority samples are numbered circles, and the minority are numbered squares. kNN algorithm finds the k nearest neighbors of the predicted sample, the question mark diamond, to compute the confidence level of the base classifiers in the local region of the diamond. Figure 2(A) shows that the normal k-NN, using $n_{neighbors} = 7$, selects only one sample of the minority class, one square, and six samples of the majority class, circles. Figure 2(B) shows the use of Reduced Minority k-NN (RMkNN) for the same scenario. Different from normal k-NN, before selecting the nearest neighbors, the algorithm reduces the distance between the query sample and the minority class neighbors. Figure 2(B) shows the reduced distance of the minority class samples as the dotted squares. Because of this reduced distance, the samples 1 and 3 are also selected as nearest neighbors of the query sample. The presented approach introduces a balanced local region definition, in terms of the number of samples, in an imbalanced dataset.

Algorithm 1 Reduced Minority K Nearest Neighbour

```
1:  Inputs:
        dataset: $X$, labels: $y$, sample query: $s_q$, # neighbors: $k$, function: $distance\_reduction\_function$
2:  $majority\_class, minority\_class \leftarrow get\_classes(y)$
3:  $X_M \leftarrow X[y == majority\_class]$
4:  $X_m \leftarrow X[y == minority\_class]$
5:  $IR \leftarrow$ imbalance ratio of $[y]$
6:  $D_m, N_m \leftarrow$ k nearest neighbors of $s_q$ using $X_m$
7:  $D_M, N_M \leftarrow$ k nearest neighbors of $s_q$ using $X_M$
8:  $D_m \leftarrow distance\_reduction\_function(D_m, IR)$
9:  $N \leftarrow concatenate(N_m, N_M)$
10: $D \leftarrow concatenate(D_m, D_M)$
11: $I_k \leftarrow$ index of $k$ smallest distances of $D$
12: return $D[I_k], N[I_k]$
```

The use of RMkNN permits a fair evaluation of base classifiers without using sampling approaches to balance the DSEL. This reduces the noise produced by oversampling approaches. Also, it enables the use of the entire DSEL, impossible when undersampling approaches balances the DSEL. We believe that this brings a more efficient use of the DSEL to identify the most competent classifiers in the selection step.

However, the intensity of the reduction should be applied carefully. With a significant reduction, only minority class samples will be used to define the competence of the base classifiers. On the other hand, a small reduction does not balance the samples used. We define the reduction function for our problem based on the datasets evaluated in Section 5. Next, we explain

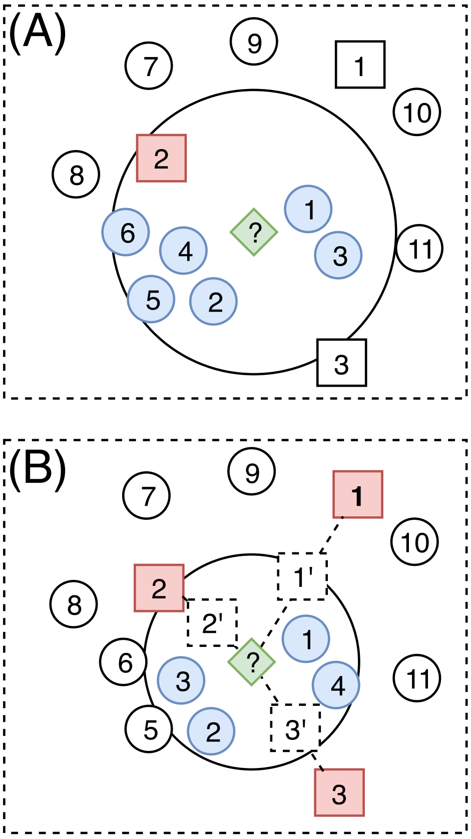

Figure 2: The normal k-NN (top) and the Modified k-NN (bottom).

the intuition behind RMkNN.

4.3. *"Why does RMkNN should work?"*

This subsection explains why RMkNN should improve the performance of dynamic selection techniques in imbalanced data sets. We analyze the role of kNN on a dynamic selection technique and the benefits of RMkNN for an imbalanced dataset.

In a dynamic selection technique, kNN is used to define the local region of a query sample. This local region is defined by selecting the nearest neighbors, usually 7, of a query sample. Once defined, a dynamic selection procedure computes the competence of the base classifiers in this local region and performs some selection based on the base classifiers' local competence.

In an imbalanced dataset, it is common to find local regions composed only by samples of the majority class. This is good when the local region contains only majority class samples, but this is not desirable in overlap regions, regions that can contain samples of both classes. The reduced number of minority class samples available can lead to kNN only find majority class samples in the neighborhood of a query sample. An overlapping local region composed only by samples of the majority class may reduce the influence of base classifiers that correctly recognize minority class samples in that local region.

The intuition behind Reduced Minority k-NN (RMkNN) is to increase the probability of overlap regions to contain samples from minority and majority classes. RMkNN do it by reducing the distance between the DSEL minority class samples and the query sample. This distance reduction should be enough to include minority samples in overlapping local regions, but it should not include minority samples in non-overlapping regions. That the reason why RMkNN uses the imbalanced ratio to define distance reduction.

Next, we describe the other possible kNN modifications evaluated to enhance the prediction performance of dynamic selection techniques.

4.4. *Other possible kNN approaches*

Beyond the RMkNN, we also evaluate two other possible k-NN modifications to handle the DSEL imbalance problem: Weighted k-NN, and the use of a fixed amount of samples of the same class in the feature space region definition. We comment in the next paragraphs the reasons why we do not use them.

Weighted k-NN consists of add weights to samples in k-NN computation. For instance, in the example of Figure 2(A), we can define that the weight for each square is 0.9, and the weight for each circle is 0.1. However, for all regions with only majority class samples, we are not able to evaluate the ability of base learners to classify the minority class. That is the reason we do not use weighted k-NN to define the competence region in the dynamic selection techniques evaluated.

Another possible approach evaluated in this work is to select a fixed amount of samples of each class in the k-NN procedure. For instance, in a dynamic selection technique that uses seven samples of DSEL to define the region of a query sample in the feature space, this approach consists in selecting four nearest neighbors of the majority class and three nearest neighbors of the minority one. However, this approach is not desirable in a region of the feature space that contains only samples of the majority class. In this kind of local region, the goal of the dynamic selection procedure is to identify the classifiers that correctly recognize the majority class. That is the reason we decided to evaluate an approach based on the reduction of the distance of the minority class.

Different from the approaches presented in this section, the novel RMkNN can define a balanced region of competence without forcing the use of far samples. Additionally, with this approach, the regions that contain only samples of one class be evaluated only by this class. The next section describes the methodology used in our experiments.

5. Experimental setup

This paper evaluates a novel approach to define the local region used to compute the competence of base classifiers in imbalanced datasets. We now present the experimental setup used to evaluate our proposal.

5.1. *Credit datasets and data preprocessing*

We perform the comparison by exploiting seven real-world credit scoring datasets. Two datasets, German and Default, are provided by UCI machine learning repository[2]. PPDai dataset comes from a Chinese internet finance enterprise named PaiPaiDai[3]. The Iranian dataset comes from paper Sabzevari et al. (2007). The private dataset comes from a financial institution in Brazil[4]. GiveMe[5] comes from a Kaggle competition. LC2015Q123 is the last one and contains loans of 36 months and a low-interest rate of the three first quarters in 2015 from Lending Club[6]. Table 3 shows the details of these datasets. We use the Imbalance Ratio (IR) measure, the cardinality of the majority class divided by the cardinality of the minority class, to sort the datasets from the less imbalanced to the most imbalanced. In the first one, German, the number of samples of the majority class is 2.33 times higher than the number of samples of the minority class. In the last one, LC2017Q1, the majority class has 80.92 times more samples than the minority class. Readers may notice that we only consider imbalanced datasets with IR greater than 2.

Table 3: Datasets description

Dataset	**#Samples**	**#Features**	**Imbalance Ratio (IR)**
German	1,000	24	2.33
Default	29,892	24	3.52
PPDai	55,596	29	6.74
Private	4,976	56	9.05
GiveMe	150,000	10	13.96
Iran	997	27	19.77
LC2015Q123	23,677	70	80.92

We perform the following data preprocessing steps. First, we use one-hot encoding to transform each categorical feature with N values in N binary features. We also fill the missing values with the mean/mode for numeric/nominal features. These are base procedures to train any machine learning model.

Additionally, we apply z-score standardization for numeric features. For instance, considering that a feature of the dataset contains the values $[40,18,18,18]$, after removing the mean and scale, we get the values $[1.732,-0.577,-0.577,-0.577]$. This is important because our solution uses the kNN algorithm, and different features lie within different ranges. Without feature standardization, large-scale features perform a bigger influence than small-scale ones. Next, we evaluate the approaches we use to measure the gains of our proposal, Reduced Minority k-NN (RMkNN).

5.2. Hyper-parameter optimization and experiment framework

We use a grid-search to find the best hyper-parameters of each ensemble using F-measure to choose the best model. We test three pool sizes for all ensembles: $[60,100,200]$. For Bagging and Random Forest-based ensembles, we test two values for the maximum number of samples: $[0.8,1]$. For Adaboost based ensembles, SMOTEBoost, RUSBoost, and Easy Ensemble, we test two values for learning rate: $[0.1,1]$. For Balanced Random Forest (BRDF), we test three values for the maximum number of features: [$\sqrt{\#features/2}$, $\sqrt{\#features}$, $\sqrt{2\times\#features}$]. From Balanced Rotation Forest (BRTF), we test two possibilities for the size of the feature group. $[3,9]$. These are the most common values adopted on the credit scoring papers of Table 1.

The SMOTE preprocessing technique also has parameters. We use the number of nearest neighbors equal 5. Finally, for all the dynamic selection methods, we use seven nearest neighbors to define the region of competence. We get these hyper-parameters from Roy et al. (2018).

We also test different hyper-parameters for the credit scoring benchmark approaches. For Logistic Regression (LOGR), we test five values for the regularization parameter C: [0.01, 0.03, 0.1, 0.3, 1]. We test two different class weights: [balanced, None], two solvers: [liblinear, saga], and two levels of tolerance for stopping criteria: [0.0001, 0.001]. For linear support vector machine (LSVM), we test five values for the regularization parameter C: [0.01, 0.03, 0.1, 0.3, 1]. We test two different class weights: [balanced, None], to levels of tolerance for stopping criteria: [0.0001, 0.001], and two maximum number of iterations: [1000, 2000]. For non-linear support vector machine (SVM), we test four values for the regularization parameter C: $[0.01,0.1,0.5,1]$. We test two different class weights: [balanced, None], to levels of tolerance for stopping criteria: [0.0001, 0.001], two maximum number of iterations: $[1000,2000]$, and two different kernels: $[rbf,poly]$. For eXtreme Gradient Boosting (XGB), as for the other ensembles, we test three ensemble sizes: [60, 100, 200]. We also test two values for control the balance of positive and negative class weights: [1, <imbalance ratio of the dataset>]. We also test to different learning rates: $[0.01,0.2]$, two max tree depth: $[3,6]$, three minimal child weight: $[1,3,5]$, two values for gamma: $[0.1,0.3]$, and two values for L1 and L2 regularization weights: $[1e-5,1e-2]$. We also test the Random Forest ensemble without any resampling step to balance the data (RNDF). For RNDF, we test all the hyperparameters combinations of imbalanced Random Forest ensembles described above, and we also test five different values for maximum tree depth: $[5,8,15,25,30]$, five values for the minimal samples split: $[2,5,10,15,100]$, four values for the minimal samples leaf:$[1,2,5,10]$, and two values for class weight:$[None,balanced]$. Finally, for artificial neural networks, we test two hidden layer sizes:[20, 40].

Figure 3 shows the experimental framework of this work. To evaluate each classification approach, we perform 5-fold cross-validation to get the mean and the standard deviation of each method. For each training fold of the 5-fold, we perform 3-fold grid search cross-validation to find the best hyper-parameters of each static classifier (steps *III* and *V* of boxes C and D of Figure 3). For boxes C, and D, we use the best static ensemble to predict the test part of the 5-fold cross-validation. For boxes A and B of Figure 3, we use the 3-fold training data

[2] https://archive.ics.uci.edu

[3] https://www.ppdai.com

[4] The citations, observations, analyzes, and conclusions related to any references to this Brazilian financial institution contained in this academic work, and their eventual implications, are the sole responsibility of the first author and do not necessarily represent the thinking or agreement of the institution or its administrators.

[5] https://www.kaggle.com/c/GiveMeSomeCredit

[6] https://www.lendingclub.com

as DSEL, box A, or to generate the DSEL using a preprocessing approach, box B. Then, we use the DSELs and the dynamic selection approaches on the imbalanced ensembles to find the best dynamic selection model, boxes A, and B of Figure 3.

5.3. Dynamic selection setup

As we mentioned above, the DSEL and the training dataset must be different to avoid overfitting. However, when a dataset has few samples of one class, splitting it increases the complexity of the training phase. The learning algorithm has to identify the patterns with even fewer samples of one class. Therefore, as adopted by Roy et al. (2018), we do not split the data used to train each approach in training and DSEL. Roy et al. used the diversity of the new samples generated by the oversampling preprocessing techniques to ensure the difference between the training data and the DSEL. However, instead of applying oversampling techniques, we propose a new approach to define the local region. This approach provides a balanced local region to evaluate the competence of base models.

5.4. Evaluation measures

A correct selection of evaluation measures is critical to avoid biased results. For instance, the percentage of correctly classified measure is widely used in classification but is not appropriate to an imbalanced dataset, since a naive classifier always predicting the majority class achieves a high score.

We evaluate six metrics to measure the predictive accuracy of the classifiers: Area under the ROC curve (AUC), H-measure, balanced accuracy (BAcc), G-mean, F-measure, and True Positive Rate (TPR). As in other work about imbalanced classification, we consider the minority class, namely the bad credit, as the positive class in order to avoid bias results in F-measure. In the next paragraphs, we present some essential measure elements and comment briefly on each performance measure evaluated in this paper.

Based on the elements of the confusion matrix, true positive (TP), false negative (FN), true negative (TN), and false positive (FP), we can define the precision, $Precision = \frac{TP}{TP+FP}$, the recall, or sensitivity or true positive rate (TPR), $Recall = \frac{TP}{TP+FN}$, the specificity or true negative rate (TNR), $Specificity = \frac{TN}{TN+FP}$, and the false positive rate (FPR), $FPR = 1 - TNR = \frac{FP}{TN+FP}$.

We now describe the performance metrics used in this paper. AUC is an extensively used evaluation measure obtained from the area under the ROC curve. The x-axis of the ROC curve represents the FPR, and the y-axis represents TPR (sensitivity). The balanced accuracy (BAcc) is the arithmetic mean of the positive class and negative class accuracy, as shown in Eq. (3). The F-measure is the weighted harmonic mean between precision and recall, as shown in Eq. (4). The β in the F-measure formula is a hyper-parameter for weighting differently the precision and recall. In this paper, we evaluate three values for β: $[1,5,35]$. The first, 1, gives equal importance for precision and recall. The second, 5, is based on the misclassification cost difference evaluated in West (2000). The last, 35, is based on Altman et al. (1977). Finally, Eq. (5) shows the G-mean, the geometric mean of sensitivity and specificity.

$$\text{Balanced Accuracy} = \frac{\frac{TP}{TP+FN} + \frac{TN}{TN+FP}}{2} \tag{3}$$

$$\text{F-measure} = \frac{(1+\beta^2) \times Precision \times Recall}{\beta^2 \times Precision + Recall} \tag{4}$$

$$\text{G-mean} = \sqrt{Sensitivity \times Specificity} \tag{5}$$

H-measure is a threshold-varying evaluation metric proposed by Hand (2009). This measure overcomes the AUC deficiency in the use of different misclassification costs distributions for different classifiers. H-measure gives a normalized classifier assessment based on expected minimum misclassification loss, ranging from zero to one for a random and perfect classifier, respectively.

5.5. The reduction function deduction

This section defines the distance function for RMkNN based on the credit scoring datasets. The distance reduction in the most imbalanced datasets should be more significant than in the less imbalanced ones. This means that the Imbalanced Ratio (IR) of the dataset should influence the reduction function. Analyzing the imbalanced ratio (IR) of real credit scoring datasets evaluated in the papers of Table 1, we see that the IRs varies from 1 to 78. For this reason, we analyze the reduction in this range, and we use the datasets we selected to perform our experiments.

To find boundaries of the distance reduction function, we use Reduced Minority k-NN to compute the percentage of minority samples selected when we get the seven nearest neighbors of each sample of a dataset with different functions. The goal of this experiment is to find a reduction function that produces a balanced percentage of minority and majority class samples selected. The Equation 6 shows the formula used to compute this percentage: S is a dataset, e is a sample of S, e_{mn} means the set of minority samples among the nearest neighbors of e using Reduced Minority k-NN, and k is the number of neighbors, in this experiment, seven.

$$Percentage\ of\ minority\ samples\ selected = \frac{\sum_{e \in S} |e_{mn}|}{k * |S|} \tag{6}$$

We perform this experiment with three functions: $f(D_m) = D_m$, $f(D_m) = 2D_m/3$, and $f(D_m) = D_m/2$, where D_m is the distance of the minority sample from the query sample. Figure 4 shows the percentage of minority samples for each dataset we test and for each boundary function. Since the ideal percentage is 50%, half of the minority and majority samples, we consider as the desired percentage, the range between 30% and 70%. Indeed, this percentage guarantees a minimal amount of samples of the minority and the majority class in the local region definition. We observe empirically that, on average, the reduction function should reduce the distance of the minority class samples by a factor between 1 and $2/3$.

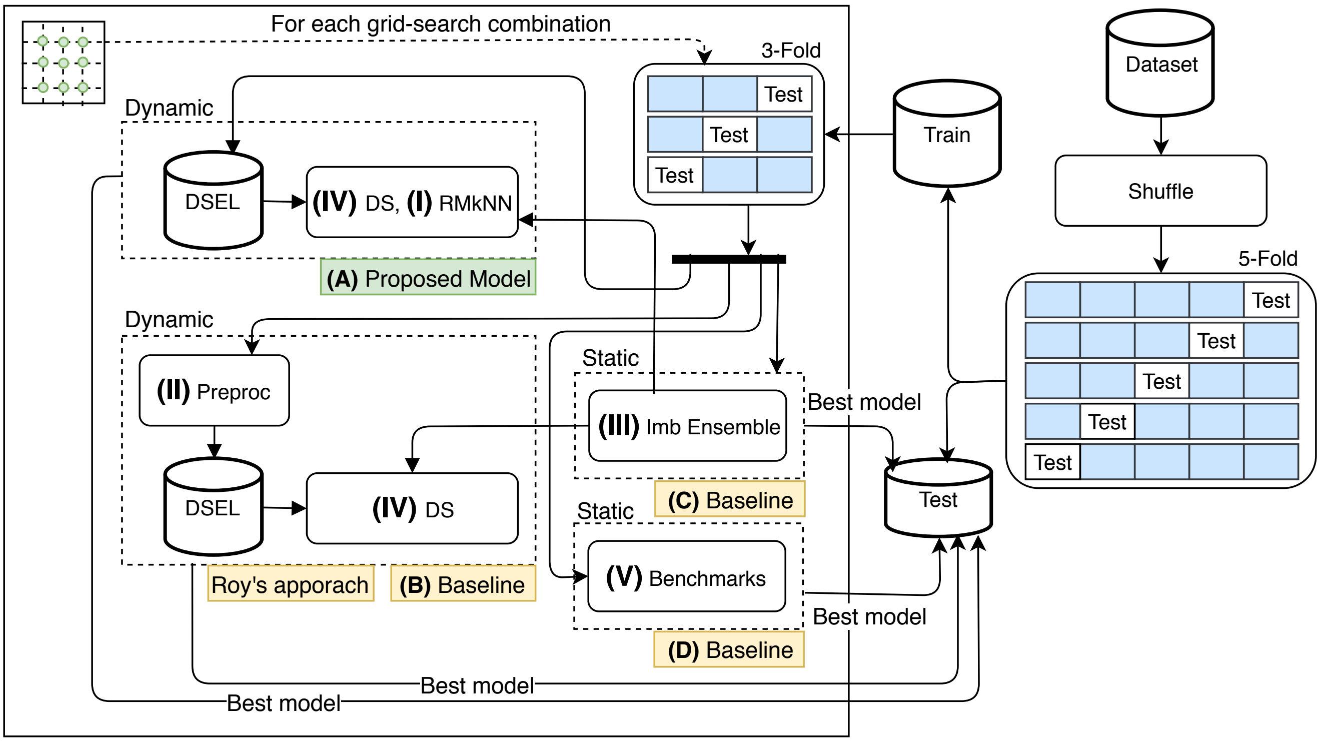

Figure 3: The proposed approach and the baselines (adapted from Roy et al. (2018)).

With these parameters in mind, we propose the function in the Eq. 7. First, we evaluate the behavior of a linear function. However, the distance reduction of the linear function is too low, mainly in less imbalanced datasets. Then, we decide to use a natural logarithmic function. Finally, we use factor 10 to adjust the result of the reduction function to generate a result near the range of 1 and 2/3. Figure 4 also shows the percentage of minority samples selected with this proposed function. As we can see, the proposed function produces a percentage of minority samples between 30% and 70% for five of seven datasets. Only the datasets GiveMe and Iran presented a percentage of minority neighbors selected below of our established threshold.

$$f(D_m, IR) = \frac{D_m}{(1 + \frac{log(IR)}{10})} \tag{7}$$

5.6. Statistical significance tests

As recommended by Demšar (2006) and followed by other credit scoring papers Xia et al. (2018); Lessmann et al. (2015), we employed nonparametric tests instead of parametric ones because the assumptions of parametric tests tend to be violated when comparing classification models. We employ the Friedman test Friedman (1940), which is a rank-based nonparametric test, to compare different models. Eq. 8 formalizes the statistic of the Friedman test.

$$X_F^2 = \frac{12D}{K(K+1)} \left[\sum_{j=1}^{K} AR_j^2 - \frac{K(K+1)^2}{4} \right], where\ AR_j = \frac{1}{D} \sum_{i=1}^{D} r_i^j. \tag{8}$$

In Eq (8), D denotes the number of datasets used in the study, K is the total number of classifiers and r_i^j is the rank of classifier j on dataset i. X_F^2 is distributed according to the Chi-square $(\tilde{\chi}^2)$ distribution with $K-1$ degrees of freedom. If the value of X_F^2 is large enough, then the null hypothesis that there is no difference between the techniques can be rejected. The Friedman statistic is well suited for this type of data analysis as it is less susceptible to outliers.

The post hoc Nemenyi test Nemenyi (1962) is applied to report any significant differences between individual classifiers. The Nemenyi post hoc test states that the performances of two or more classifiers are significantly different if their average ranks differ by at least the critical difference (CD), given by

$$CD = q_{\alpha,\infty,K} \sqrt{\frac{K(K+1)}{12D}}. \tag{9}$$

In this formula, the value $q_{\alpha,\infty,K}$ is based on the Studentized range statistic Nemenyi (1962). Finally, the results from Friedman's statistic and the Nemenyi post hoc tests are displayed using a modified version of significance diagrams Demšar (2006); Lessmann et al. (2008). These diagrams display the ranked performances of the classification techniques along with the critical difference to clearly show any techniques which are significantly different from the best-performing classifiers. Next, we discuss the results achieved in these tests.

6. Experimental results

We now present the results by answering each research question. First, we analyze if the dynamic selection techniques are

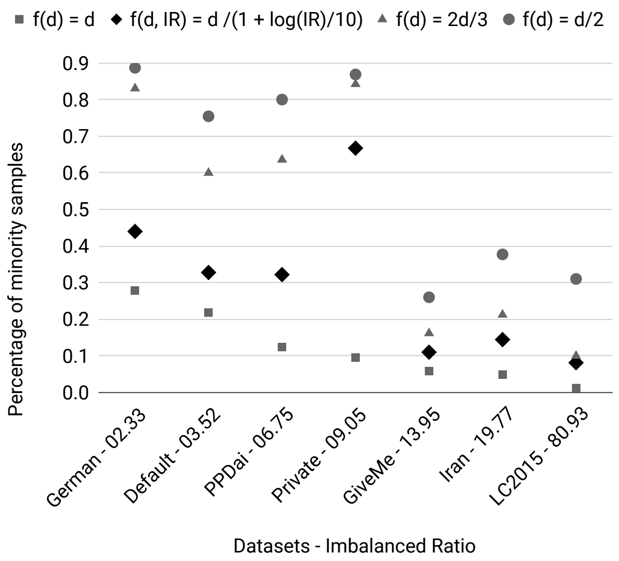

Figure 4: The percentage of minority samples selected when different reduction functions are used in seven datasets.

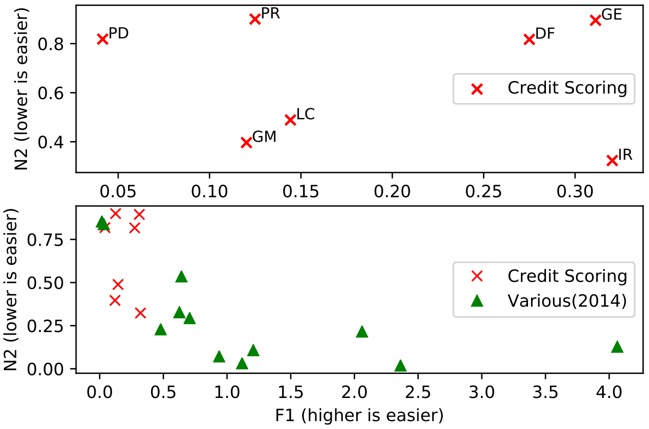

Figure 5: Credit scoring classification complexity measures (up). Credit scoring classification complexity measures compared with other datasets (bottom). Dataset abbreviations: DF: Default, GE: German, GM: GiveMe, IR: Iran, LC: LC2015Q123, PD: PPDai, PR: private.

appropriate to credit scoring datasets. Then, we analyze the differences between performance measures. Finally, we compare the proposed approach with dynamic ensemble approaches that use DSEL generated by preprocessing techniques and the static ensembles.

As in previous works Lessmann et al. (2015); Abellán & Castellano (2017), we use the average rank of the selected performance measures. For the F-measure, we adopted three values for β: $[1, 5, 35]$. $\beta = 1$ means to give the same weight for precision and recall in the Equation 4. The other two F-measures gives, respectively, 5 and 35 times more important to positive class misclassification than to negative class error. Henceforth, we refer to F-measures as F1, F5, and F35, when β is $[1, 5, 35]$, respectively. Next, subsections answer the research questions.

6.1. *Dynamic selection for Imbalanced credit scoring datasets*

To answer **RQ1) "Are dynamic selection techniques appropriate for imbalanced credit scoring problems?"**, we work in two lines. First, we investigate two classification complexity measures presented by Ho & Basu (2002). After, we reduce the dynamic selection techniques presented in Cruz et al. (2018) to static selection. The next subsections describe these two investigations.

6.1.1. *Credit scoring dataset complexity*

We evaluate the classification complexity based on the conclusion of Britto Jr et al. (2014) that dynamic selection techniques are appropriate to complex datasets. We apply *Fisher's Discriminant Ratio (F1)* and *Ratio of Average Intra/Inter class NN distance (N2)* as classification complexity measures to the datasets described in Table 3. The results are in the top part of Figure 5. Each axis of the graph is one of the complexity measures. Each red cross represents a credit scoring dataset.

The first conclusion we find is that, regarding N2, the datasets Iran (IR) and GiveMe (GM) have a lower ratio of Intra/Inter class NN distance. It means that these datasets have a higher separability of classes. This explains the different behavior of these datasets in the experiment performed in subsection 5.5. As there is a higher separability of the classes on these datasets, the impact of reducing the Euclidian distance of the minority class in k-NN is lower. Fortunately, for five of seven credit scoring datasets evaluated, the N2 measure is higher, suggesting that most of the datasets are complex.

The second finding is related to Fisher's Discriminant Ratio (F1). Regarding this measure, all seven credit scoring datasets evaluated have $F1 < 0.31$. This means that all credit scoring datasets are complex regarding F1.

We now compare the F1 and N2 measures with the datasets evaluated by Britto Jr et al. (2014) in the bottom part of Figure 5. In this figure, the datasets evaluated by Britto Jr et al. (2014) are indicated by the green triangles, while red crosses indicate the credit scoring datasets used in this paper. Only one among all datasets evaluated by Britto Jr et al. (2014) is more complex than the credit scoring datasets of Table 3 regarding F1. Regarding N2, the easiest credit scoring datasets, Iran (IR) and GiveMe (GM), are among the three harder datasets evaluated by Britto Jr et al. (2014).

Finally, based on the result of Britto Jr et al. (2014) stating that dynamic selection techniques are more appropriate to complex classification problems, we can conclude empirically that credit scoring datasets are complex and suitable for dynamic selection techniques. We use Fisher's Discriminant Ratio (F1) and Ratio of Average Intra/Inter class NN distance (N2) to measure and compare with datasets of other fields.

6.1.2. *Equivalence of dynamic and static selection techniques*

To handle the regulatory compliance of Basel accords, that requires the use of the same prediction model to all costumers, we find a static classifier equivalence to the dynamic selection techniques of Cruz et al. (2018). We begin this subsection describing the structure of the KNORA-Union (KNU) dynamic selection technique. After, we describe the static equivalence of KNU.

KNORA-U implementation starts by collecting all base classifiers' predictions of all DSEL samples. If all classifiers predict the same class for one query sample, there is no selection to be done. Otherwise, this prediction information is used to compute the local accuracy of each base classifier. The local accuracy is used to define the weight of each base classifier in the final prediction. To illustrate this behavior with an example, if the accuracy in some local region of classifiers A, B and C is 1, 0.7 and 0, respectively, the weights of the classifiers A and B in the final prediction are 1, and 0.7, respectively, while the classifier C does not influence the final prediction.

We observe that all information needed to compute the base models' accuracy in each part of the feature space is available at the fit time. With the base models and the DSEL samples, we can define statically in all local regions of the feature space which base classifiers participate and what is their contribution weight in the final prediction.

To illustrate the concept described previously, Figure shows a simple example of the definition of the local regions. Figure 6 (up) shows a bi-dimensional feature space with nine DSEL samples marked in green. Using only two neighbors to define a local region, Figure 6 (bottom) shows the local regions defined by these 9 DSEL samples in different colors. In each local region, different colors of Figure 6 (bottom), the local competence of the base classifiers is the same. It means that the influence of the base classifiers on the final prediction is the same on all local region.

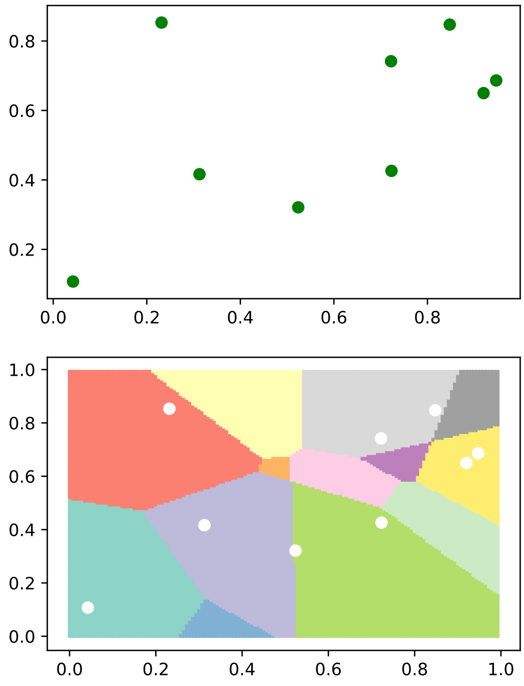

Figure 6: A bi-dimensional feature space with nine DSEL random samples in green (up). The 13 local regions defined by these nine samples. Each local region is defined by two DSEL nearest neighbors (bottom).

As these local regions define the influence of each base classifier on the final prediction, we can define a static tree where the root node has one child for each different local region, in the example of Figure 6 (bottom), 13 child nodes. In each node of this tree, we can have a static ensemble with the weights defined by the competence of the base models in the DSEL samples that define the local region.

Figure 7 shows the static tree equivalent to the dynamic approach. In Figure 7, the SE's represent the 13 different sub-ensembles. As in each local region of Figure 6 the local competence of the base classifiers are the same, each local region define a static sub ensemble of the original ensemble.

In this subsection, we observe that a dynamic classification technique has a similar static approach. This can be a starting point to allow the use of dynamic classification in credit scoring field.

6.2. RMkNN and kNN comparison

To assess **RQ2) "Does the RMkNN improve the prediction performance of kNN?"**, we use these techniques as classifiers and perform the static experiment flow of Figure 3 to compare them. Table 4 shows the results of two classifiers over the seven evaluated datasets. To simplify the results evaluation task, we sort the datasets by the imbalance level. About the performance measures, we start with the threshold-free measures, AUC, and H-measure. After, we include balanced accuracy and geometric mean (G-mean). We also include the F-measures measures at an increasing level of True Positive Rate (TPR) influence, F1, F5, and F35. Finally, we include TPR alone. For all threshold dependent measures, we consider 0.5 as the threshold.

First, we observe that RMkNN outperforms kNN regarding G-mean, F1-score, F5-score, F35-score, and TPR. Evaluating G-mean, we notice that kNN outperforms RMkNN only on the private dataset. However, the performance difference between RMkNN and kNN is only $0.01 = (0.51 - 0.5)$. Regarding the threshold-free measures, AUC and H-measure, we observe that RMkNN outperforms kNN in 4 of 7 datasets. Additionally, we observe that on the 3 cases where kNN achieves superior performance, RMkNN achieves similar results.

We can conclude the superiority of RMkNN in imbalanced credit scoring problems considering the following arguments. First, we remember that AUC and H-measure give the same weight for the misclassification error of both classes, and F5,

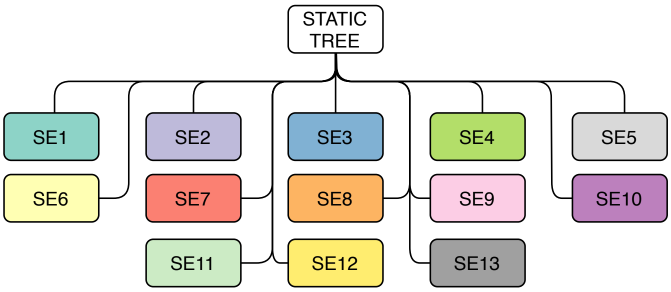

Figure 7: Static tree equivalent to a dynamic selection classification.

Table 4: kNN and RMkNN comparison (each column contains the average and standard deviation of 5-fold execution)

Dataset	Classifier	Performance measures							
		AUC	H	BAcc	G-mean	F1	F5	F35	TPR
German	kNN	0.74 (0.047)	0.09 (0.05)	0.6 (0.036)	0.5 (0.063)	0.38 (0.077)	0.29 (0.066)	0.28 (0.065)	0.28 (0.065)
	RMkNN	**0.75 (0.024)**	**0.16 (0.043)**	**0.69 (0.026)**	**0.69 (0.025)**	**0.57 (0.029)**	**0.71 (0.042)**	**0.72 (0.043)**	**0.72 (0.043)**
Default	kNN	**0.73 (0.018)**	**0.16 (0.02)**	0.64 (0.011)	0.56 (0.019)	0.43 (0.022)	0.34 (0.024)	0.34 (0.024)	0.34 (0.024)
	RMkNN	0.72 (0.018)	0.16 (0.027)	**0.67 (0.016)**	**0.65 (0.019)**	**0.48 (0.023)**	**0.52 (0.029)**	**0.52 (0.029)**	**0.52 (0.029)**
PPDai	kNN	0.57 (0.017)	0.01 (0.005)	0.51 (0.004)	0.14 (0.036)	0.04 (0.018)	0.02 (0.01)	0.02 (0.01)	0.02 (0.01)
	RMkNN	**0.57 (0.007)**	**0.01 (0.004)**	**0.55 (0.007)**	**0.52 (0.017)**	**0.23 (0.007)**	**0.35 (0.056)**	**0.37 (0.067)**	**0.37 (0.067)**
Private	kNN	**0.57 (0.047)**	**0.01 (0.008)**	**0.51 (0.009)**	0.14 (0.036)	0.04 (0.021)	0.02 (0.011)	0.02 (0.011)	0.02 (0.011)
	RMkNN	0.51 (0.049)	0.0 (0.002)	0.5 (0.014)	**0.22 (0.036)**	**0.18 (0.005)**	**0.71 (0.019)**	**0.93 (0.028)**	**0.94 (0.028)**
GiveMe	kNN	0.72 (0.003)	0.07 (0.008)	0.55 (0.005)	0.32 (0.015)	0.17 (0.015)	0.11 (0.01)	0.1 (0.009)	0.1 (0.009)
	RMkNN	**0.77 (0.007)**	**0.22 (0.009)**	**0.65 (0.005)**	**0.58 (0.008)**	**0.37 (0.008)**	**0.35 (0.01)**	**0.35 (0.01)**	**0.35 (0.01)**
Iran	kNN	**0.77 (0.043)**	**0.1 (0.141)**	0.5 (0.002)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
	RMkNN	0.75 (0.061)	0.06 (0.073)	**0.56 (0.082)**	**0.38 (0.242)**	**0.13 (0.097)**	**0.2 (0.162)**	**0.21 (0.172)**	**0.21 (0.173)**
LC2015 Q123	kNN	0.6 (0.008)	0.0 (0.002)	0.51 (0.003)	0.17 (0.02)	0.05 (0.011)	0.03 (0.007)	0.03 (0.007)	0.03 (0.007)
	RMkNN	**0.61 (0.007)**	**0.02 (0.005)**	**0.57 (0.009)**	**0.55 (0.011)**	**0.28 (0.01)**	**0.42 (0.019)**	**0.44 (0.02)**	**0.44 (0.02)**

(a) **BAcc** stands for balanced accuracy.

F35, and TPR give a higher weight to the positive class misclassification. When we observe split results in AUC and H-measure and RMkNN superiority in F5, F35, and TRP, we can conclude that RMkNN is more appropriate than kNN to handle classification problems when the positive class misclassification is higher.

6.3. Reduced Minority k-NN on dynamic selection techniques

To answer **RQ3) Does the use of the RMkNN technique - that defines a novel competence region of dynamic selection techniques - improve the classification performance of imbalanced credit scoring datasets?**, we perform three experiments. First, we compute the overall average ranking of 110 classification approaches. After, we compare the best estimator of the previous test with the credit scoring benchmarks. Finally, we simulate a real scenario of a credit scoring problem. The following subsections describe each experiment.

6.3.1. Overall average ranking

In this experiment, we compare the combinations of pool generators, preprocessing approaches, and dynamic selection techniques of Table 2 with the static application of the imbalanced ensemble and with credit scoring benchmarks. We evaluate the average rank of all 110 combinations (8 imbalanced ensembles × 4 selection approaches × 3 strategies to handle DSEL + 8 static imbalanced ensembles + 6 credit scoring benchmarks) to start the investigation of the best approaches to imbalanced credit scoring datasets.

To get a first observation of the best results among the 110 approaches evaluated, we compute the average ranking of the eight performance measures evaluated, AUC, H-measure, balanced accuracy, geometric mean, F1-score, F5-score, F35-score, and recall (TPR). After, we compute the average of these averages to find a unique global rank. Table 5 shows these ranks and the overall average of the average ranks. In this table, the gray cells indicate the lowest average rank of each performance measure. As we can see, the balanced versions of Random Forest (BRDN) and Rotation Forest (BROT) achieve the best global average rankings.

Table 5 shows that the three imbalanced ensembles achieve the lowest average ranks of all performance measures evaluated. The 14 first places in the ranking are composed only by Balanced Random Forest (BRND), Balanced Rotation Forest (BROT), Easy Ensemble (EASY), and Balanced Bagging (BBAG). Extreme Gradient Boosting achieves only the 15th place in this rank.

Another important observation extracted from Table 5 is that the lowest average ranking of each imbalanced ensemble uses KNORA-Union and Reduced Minority kNN. We highlight in green the lines of Table 5 these combinations.

6.3.2. Comparison of the best average ranking with the credit scoring benchmarks

After this preliminary evaluation, we decide to evaluate the actual results of the balanced random forest combined with the dynamic selection technique KNORA-Union and Reduced Minority kNN (RMkNN), the lowest rank of Table 5, and the benchmark approaches for credit scoring: Logistic Regression (LOGR), Artificial Neural Networks (ANN), Linear Support Vector Machine (LSVM), Non-linear Support Vector Machine (SVM), Random Forest (RNDF) and eXtreme Gradient Boosting (XGB). We also include the static version of the balanced random forest to evaluate the improvement of the dynamic selection technique by each dataset.

Table 6 shows the these results. For each dataset evaluated, Table 6 shows the average and the standard deviation of 5-fold execution explained in Figure 3. Here, we also highlight the best result of each dataset and each performance measure in gray.

The investigation of German, Private, Iran and LC2015123 is quite straightforward. BRDN+KNU+RMkNN achieves the best results inat least 4 of 8 performance measures. In the Private dataset, BRDN+KNU+RMkNN achieves the best result in 6 of 8 performance measures.

Table 5: Average ranking of all 110 techniques

Appr.	Selection	Performance Measures								Avg
		AUC	H	BAcc	G-mean	F1	F5	F35	TPR	
BRND	KNU+RMkNN	12.8 (12.12)	15.4 (15.09)	**10.5 (14.08)**	11.6 (15.53)	18.7 (17.02)	15.6 (16.92)	18.4 (16.59)	19.2 (16.63)	15.3
BROT	KNU+RMkNN	**12.7 (8.5)**	14.8 (18.4)	11.9 (13.87)	**11.4 (13.33)**	**16.8 (14.07)**	17.1 (15.1)	20.1 (14.12)	20.7 (12.95)	15.7
BROT	KNU+SMTE	14.7 (8.39)	19.0 (14.91)	13.9 (10.28)	13.6 (11.36)	21.5 (12.68)	16.1 (12.65)	18.3 (12.61)	18.7 (12.58)	17.0
BRND	KNU+SMTE	14.4 (7.93)	20.5 (14.55)	13.2 (12.79)	13.7 (14.94)	23.8 (16.94)	15.8 (15.47)	17.9 (15.93)	18.5 (15.72)	17.2
BRND	STATIC	15.2 (16.55)	24.6 (19.79)	13.8 (15.36)	14.5 (16.39)	29.6 (25.1)	**14.6 (15.19)**	**15.5 (15.92)**	**15.0 (15.99)**	17.8
BROT	STATIC	14.4 (12.59)	23.6 (16.73)	14.8 (12.32)	14.7 (13.8)	28.9 (21.07)	15.7 (12.82)	16.5 (13.45)	16.1 (13.6)	18.1
BROT	KNU+RUS	15.8 (11.83)	23.0 (16.84)	14.3 (11.64)	14.1 (12.61)	27.8 (19.91)	16.1 (13.37)	17.3 (13.47)	17.7 (13.43)	18.3
BRND	KNU+RUS	15.5 (14.64)	24.9 (19.11)	15.4 (16.35)	15.5 (16.97)	30.0 (24.78)	16.2 (15.74)	17.2 (16.18)	17.4 (16.13)	19.0
EASY	KNU+RMkNN	29.6 (21.55)	21.5 (23.01)	22.2 (29.24)	22.9 (30.91)	31.1 (26.09)	21.3 (22.44)	22.3 (21.79)	22.2 (21.78)	24.1
EASY	KNU+SMTE	30.3 (21.07)	23.2 (23.04)	23.8 (28.78)	24.5 (31.48)	33.5 (26.66)	22.4 (22.89)	23.6 (22.31)	23.5 (22.19)	25.6
EASY	STATIC	24.4 (21.45)	26.7 (24.38)	26.0 (29.29)	26.0 (30.98)	37.4 (29.61)	22.6 (22.57)	22.6 (22.7)	21.6 (23.27)	25.9
EASY	KNU+RUS	32.1 (22.02)	26.1 (25.34)	25.1 (29.03)	25.5 (31.4)	36.7 (30.03)	22.3 (22.64)	22.5 (22.71)	22.0 (22.92)	26.5
EASY	KNE+RMkNN	28.2 (20.18)	25.3 (21.1)	31.4 (26.86)	32.6 (28.66)	28.5 (24.5)	32.7 (23.34)	34.5 (22.6)	35.1 (22.53)	31.0
BBAG	KNU+RMkNN	26.4 (25.38)	**14.7 (21.04)**	26.6 (26.35)	28.8 (21.93)	20.2 (27.92)	41.9 (20.51)	47.2 (18.65)	48.1 (17.45)	31.7
XGB	STATIC	18.5 (20.92)	25.4 (26.38)	26.1 (29.38)	28.6 (32.47)	23.9 (25.41)	41.6 (30.22)	44.7 (29.22)	45.2 (29.19)	31.8

It is a significant result, once we have one low imbalanced dataset, German, one moderate imbalanced one, Private, and two high imbalanced, Iran and LC2015Q123.

On the other hand, BRDN+KNU+RMkNN, our proposed combination, does not achieve any best result in any performance measure on the Default, PPDai, and GiveMe datasets. However, if we evaluate the difference between the best approaches of these datasets carefully, we see that the differences between BRDN+KNU+RMkNN and the highest scores are under 0.03. For instance, regarding AUC and H-measure, measures that give the same importance to the misclassification cost of both classes, the highest differences between BRDN+KNU+RMkNN and the best results are 0.023 (H-measure difference in Default dataset) and 0.021 (AUC difference in PPDai dataset). Additionally, evaluating the three different f-measures, the gap of BRDN+KNU+RMkNN to the best results is under 0.002, an acceptable result.

6.3.3. *Real credit scoring scenario*

Our last experiment to measure the ability of RMkNN to improve the prediction performance of credit scoring datasets is a practical application of credit scoring. We use the entire LC2015Q123 dataset to train the models using the experimental setup defined in Figure 3, and we evaluate the performance of all 110 models in the last quarter, LC2015Q4.

After collecting the performance measure of all classifier combinations, we compute the average rank of all performance measures to find, by each ensemble, the best combination. The best combinations and the credit scoring benchmarks results are in Table 7.

The first exciting outcome from Table 7 is the amount of best ensemble combinations with RMkNN. Four of the eight best ensemble combinations use RMkNN. They are: Balanced Random Forest (BRND), Random Forest SMOTE (RFSM), Bagging SMOTE (BGSM), and Easy ensemble (EASY). This result shows the superiority of RMkNN over the other imbalanced dynamic selection strategies evaluated.

Another interesting result of this experiment is the performance of BRND+KNU+RMkNN. As in the results shown in Table 6, BRND+KNU+RMkNN does not achieve the best results on F35 and TPR. However, the performance of this combination on these measures is not far from the best ones.

With these experiments, we infer that RMkNN combined with dynamic selection approaches improves the prediction performance of imbalanced ensembles. We also note that Balanced Random Forest combined with KNORA-Union and RMkNN outperforms classical credit scoring classifiers, such as eXtreme Gradient Boosting, Support Vector Machine, Artificial Neural Networks, and Logistic Regression.

Finally, we test RMkNN in a real credit scoring scenario, where we train the model with the available data on time to predict future loans. Again, the dynamic selection approaches combined with RMkNN outperform credit scoring benchmarks.

6.4. *Discussion*

We now investigate the best combination strategy among all evaluated. To achieve it, we compute a new average rank of the best results of each ensemble combination and the credit scoring benchmarks. Applying the Friedman test on the average ranking of these twelve classifiers, we get a Friedman test statistic $= 90.41$, and a $p - value < 0.005$. As this result is significant ($p < 0.005$), we can apply the post hoc Nemenyi test to the distribution.

Figure 8 shows the average ranks of these best combinations and the Critical Distance of Nemenyi test. This figure shows that balanced random forest (BRDF) and balanced Rotation Forest (BROT) combined with KNORA Union (KNU) and using RMkNN to generate the DSEL are the best approaches, the lowest ranks. However, these approaches are statistically better than Artificial Neural Networks and Support Vector Machine, as indicated by the critical distance bar.

We also observe that RMkNN is present on four best combinations of eight ensembles. They are highlighted in green on Figure 8, and they are: Balanced Random Forest (BRND),

Table 6: Balanced Random Forest combined with KNORA-U and RMkNN compared with state-of-the-art classifiers in credit scoring problem

Dataset	Classif.	Selection	Performance Measures							
			AUC	H	BAcc	G-mean	F1	F5	F35	TPR
German	XGB	STATIC	0.788 (0.02)	0.232 (0.04)	0.720 (0.02)	0.717 (0.02)	0.607 (0.02)	0.668 (0.04)	0.673 (0.04)	0.673 (0.04)
	LOGR	STATIC	0.795 (0.03)	**0.264 (0.05)**	**0.738 (0.03)**	**0.736 (0.03)**	**0.628 (0.03)**	0.718 (0.08)	0.726 (0.08)	0.727 (0.08)
	ANN	STATIC	0.767 (0.02)	0.180 (0.04)	0.675 (0.02)	0.651 (0.03)	0.540 (0.03)	0.501 (0.05)	0.498 (0.05)	0.498 (0.05)
	LSVM	STATIC	0.721 (0.02)	0.232 (0.03)	0.721 (0.02)	0.717 (0.03)	0.606 (0.03)	0.685 (0.09)	0.693 (0.10)	0.693 (0.10)
	SVM	STATIC	0.796 (0.03)	0.188 (0.07)	0.672 (0.03)	0.637 (0.04)	0.532 (0.05)	0.465 (0.04)	0.460 (0.04)	0.460 (0.04)
	RNDF	STATIC	0.791 (0.03)	0.225 (0.04)	0.715 (0.02)	0.711 (0.02)	0.600 (0.03)	0.658 (0.07)	0.663 (0.07)	0.663 (0.07)
	BRDF	STATIC	0.800 (0.03)	0.240 (0.07)	0.729 (0.03)	0.727 (0.03)	0.617 (0.04)	0.750 (0.04)	0.764 (0.04)	0.764 (0.04)
	BRDF	KNU+RMkNN	**0.802 (0.03)**	0.256 (0.06)	0.737 (0.03)	0.736 (0.03)	0.627 (0.04)	**0.757 (0.04)**	**0.770 (0.04)**	**0.770 (0.04)**
Default	XGB	STATIC	0.783 (0.02)	0.229 (0.04)	0.714 (0.02)	**0.709 (0.02)**	0.540 (0.03)	0.617 (0.03)	0.624 (0.03)	0.624 (0.03)
	LOGR	STATIC	0.722 (0.02)	0.137 (0.03)	0.672 (0.02)	0.671 (0.02)	0.478 (0.02)	0.623 (0.03)	0.639 (0.03)	0.639 (0.03)
	ANN	STATIC	0.773 (0.02)	0.205 (0.04)	0.659 (0.02)	0.592 (0.03)	0.475 (0.04)	0.377 (0.03)	0.371 (0.03)	0.371 (0.03)
	LSVM	STATIC	0.673 (0.02)	0.140 (0.03)	0.673 (0.02)	0.671 (0.02)	0.479 (0.02)	0.619 (0.03)	0.635 (0.03)	0.635 (0.03)
	SVM	STATIC	0.633 (0.02)	0.001 (0.00)	0.501 (0.00)	0.022 (0.03)	0.003 (0.00)	0.001 (0.00)	0.001 (0.00)	0.001 (0.00)
	RNDF	STATIC	**0.784 (0.02)**	**0.239 (0.04)**	**0.714 (0.02)**	0.704 (0.02)	**0.546 (0.03)**	0.591 (0.03)	0.595 (0.03)	0.596 (0.03)
	BRDF	STATIC	0.780 (0.02)	0.211 (0.04)	0.709 (0.02)	0.706 (0.02)	0.529 (0.02)	**0.631 (0.03)**	**0.641 (0.03)**	**0.641 (0.03)**
	BRDF	KNU+RMkNN	0.779 (0.02)	0.216 (0.04)	0.711 (0.02)	0.707 (0.02)	0.533 (0.03)	0.627 (0.03)	0.637 (0.03)	0.637 (0.03)
PPDai	XGB	STATIC	**0.632 (0.05)**	**0.024 (0.02)**	**0.564 (0.04)**	**0.455 (0.26)**	**0.211 (0.12)**	0.357 (0.21)	0.379 (0.22)	0.380 (0.22)
	LOGR	STATIC	0.629 (0.03)	0.019 (0.04)	0.521 (0.04)	0.146 (0.20)	0.066 (0.13)	0.058 (0.12)	0.058 (0.12)	0.058 (0.12)
	ANN	STATIC	0.627 (0.03)	0.009 (0.00)	0.509 (0.01)	0.132 (0.08)	0.041 (0.03)	0.023 (0.02)	0.023 (0.02)	0.022 (0.02)
	LSVM	STATIC	0.519 (0.04)	0.018 (0.04)	0.519 (0.04)	0.115 (0.21)	0.061 (0.13)	0.055 (0.12)	0.054 (0.12)	0.054 (0.12)
	SVM	STATIC	0.477 (0.05)	0.000 (0.00)	0.500 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
	RNDF	STATIC	0.631 (0.04)	0.022 (0.02)	0.560 (0.04)	0.437 (0.25)	0.204 (0.12)	0.406 (0.27)	0.448 (0.31)	0.449 (0.31)
	BRDF	STATIC	0.615 (0.05)	0.018 (0.01)	0.554 (0.03)	0.428 (0.23)	0.200 (0.11)	**0.459 (0.30)**	**0.519 (0.35)**	**0.521 (0.35)**
	BRDF	KNU+RMkNN	0.611 (0.04)	0.018 (0.01)	0.554 (0.03)	0.432 (0.23)	0.200 (0.11)	0.452 (0.30)	0.509 (0.35)	0.511 (0.35)
Private	XGB	STATIC	0.682 (0.04)	0.067 (0.05)	0.603 (0.06)	0.541 (0.13)	0.242 (0.07)	0.369 (0.17)	0.387 (0.19)	0.388 (0.19)
	LOGR	STATIC	0.668 (0.05)	0.060 (0.03)	0.620 (0.03)	0.618 (0.03)	0.245 (0.02)	0.550 (0.06)	0.612 (0.08)	0.613 (0.08)
	ANN	STATIC	0.561 (0.07)	0.019 (0.01)	0.513 (0.06)	0.414 (0.05)	0.160 (0.04)	0.212 (0.05)	0.220 (0.05)	0.221 (0.05)
	LSVM	STATIC	0.546 (0.04)	0.017 (0.02)	0.546 (0.04)	0.403 (0.23)	0.144 (0.09)	0.308 (0.26)	0.347 (0.30)	0.347 (0.30)
	SVM	STATIC	0.644 (0.05)	0.015 (0.01)	0.539 (0.03)	0.355 (0.21)	0.149 (0.09)	0.184 (0.12)	0.188 (0.13)	0.188 (0.13)
	RNDF	STATIC	0.719 (0.02)	0.106 (0.06)	0.618 (0.05)	0.540 (0.12)	**0.283 (0.07)**	0.343 (0.14)	0.351 (0.15)	0.352 (0.15)
	BRDF	STATIC	**0.719 (0.03)**	0.105 (0.02)	0.662 (0.02)	0.661 (0.02)	0.279 (0.01)	0.603 (0.05)	0.666 (0.06)	0.668 (0.06)
	BRDF	KNU+RMkNN	0.718 (0.03)	**0.111 (0.03)**	**0.667 (0.03)**	**0.666 (0.02)**	0.281 (0.01)	**0.617 (0.06)**	**0.683 (0.07)**	**0.685 (0.07)**
GiveMe	XGB	STATIC	**0.865 (0.00)**	0.343 (0.01)	**0.786 (0.00)**	**0.786 (0.00)**	0.338 (0.00)	0.703 (0.01)	0.770 (0.01)	0.772 (0.01)
	LOGR	STATIC	0.806 (0.01)	0.252 (0.01)	0.732 (0.00)	0.726 (0.00)	0.311 (0.01)	0.593 (0.01)	0.641 (0.01)	0.642 (0.01)
	ANN	STATIC	0.833 (0.01)	0.125 (0.01)	0.582 (0.00)	0.414 (0.01)	0.266 (0.01)	0.178 (0.01)	0.173 (0.01)	0.173 (0.01)
	LSVM	STATIC	0.651 (0.00)	0.089 (0.00)	0.651 (0.00)	0.651 (0.00)	0.203 (0.00)	0.545 (0.01)	0.632 (0.01)	0.635 (0.01)
	SVM	STATIC	0.484 (0.03)	0.000 (0.00)	0.500 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
	RNDF	STATIC	0.862 (0.00)	**0.349 (0.02)**	0.779 (0.00)	0.777 (0.01)	**0.356 (0.02)**	0.674 (0.02)	0.728 (0.03)	0.729 (0.03)
	BRDF	STATIC	0.862 (0.00)	0.337 (0.01)	0.785 (0.00)	0.785 (0.00)	0.331 (0.00)	**0.708 (0.00)**	**0.780 (0.00)**	**0.782 (0.00)**
	BRDF	KNU+RMkNN	0.861 (0.00)	0.344 (0.01)	0.786 (0.00)	0.785 (0.00)	0.340 (0.00)	0.701 (0.00)	0.767 (0.01)	0.769 (0.01)
Iran	XGB	STATIC	0.760 (0.06)	0.146 (0.06)	0.608 (0.03)	0.489 (0.06)	**0.267 (0.06)**	0.251 (0.06)	0.251 (0.06)	0.251 (0.06)
	LOGR	STATIC	0.777 (0.06)	0.000 (0.00)	0.499 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
	ANN	STATIC	0.769 (0.03)	0.010 (0.02)	0.501 (0.01)	0.032 (0.07)	0.018 (0.04)	0.010 (0.02)	0.010 (0.02)	0.010 (0.02)
	LSVM	STATIC	0.567 (0.16)	0.093 (0.21)	0.567 (0.16)	0.169 (0.38)	0.062 (0.14)	0.157 (0.35)	0.179 (0.40)	0.180 (0.40)
	SVM	STATIC	0.760 (0.09)	0.000 (0.00)	0.500 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
	RNDF	STATIC	0.792 (0.04)	0.107 (0.04)	0.569 (0.02)	0.374 (0.06)	0.227 (0.06)	0.148 (0.05)	0.145 (0.05)	0.144 (0.05)
	BRDF	STATIC	0.770 (0.05)	0.176 (0.08)	0.708 (0.05)	0.707 (0.05)	0.192 (0.04)	**0.585 (0.07)**	**0.706 (0.08)**	**0.709 (0.08)**
	BRDF	KNU+RMkNN	**0.810 (0.07)**	**0.279 (0.12)**	**0.735 (0.07)**	**0.722 (0.08)**	0.266 (0.06)	0.566 (0.12)	0.625 (0.15)	0.627 (0.15)
LC2015 Q123	XGB	STATIC	**0.712 (0.04)**	0.081 (0.02)	0.636 (0.02)	0.622 (0.04)	**0.049 (0.00)**	0.295 (0.03)	0.507 (0.08)	0.515 (0.08)
	LOGR	STATIC	0.693 (0.02)	0.028 (0.04)	0.555 (0.08)	0.253 (0.35)	0.017 (0.02)	0.120 (0.16)	0.230 (0.32)	0.234 (0.32)
	ANN	STATIC	0.519 (0.01)	0.004 (0.00)	0.503 (0.01)	0.049 (0.08)	0.007 (0.01)	0.013 (0.03)	0.016 (0.03)	0.016 (0.03)
	LSVM	STATIC	0.502 (0.00)	0.000 (0.00)	0.502 (0.00)	0.057 (0.13)	0.005 (0.01)	0.015 (0.03)	0.017 (0.04)	0.017 (0.04)
	SVM	STATIC	0.550 (0.02)	0.000 (0.00)	0.500 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)	0.000 (0.00)
	RNDF	STATIC	0.709 (0.03)	0.030 (0.05)	0.539 (0.06)	0.202 (0.29)	0.024 (0.03)	0.090 (0.13)	0.120 (0.19)	0.121 (0.19)
	BRDF	STATIC	0.703 (0.03)	0.083 (0.01)	0.654 (0.01)	0.653 (0.01)	0.043 (0.00)	0.317 (0.01)	**0.661 (0.03)**	**0.676 (0.03)**
	BRDF	KNU+RMkNN	0.702 (0.03)	**0.092 (0.03)**	**0.656 (0.02)**	**0.655 (0.02)**	0.047 (0.00)	**0.319 (0.02)**	0.614 (0.04)	0.626 (0.04)

Table 7: Classification results of the 8 best ensemble combinations and the 4 credit scoring benchmark approaches over the last quarter of 2015 of LC2015 dataset.

Dataset	Appr.	Selection	Performance Measures							
			AUC	H	BAcc	G-mean	F1	F5	F35	TPR
LC2015Q4	BRND	KNU+RMkNN	0.679	**0.065**	**0.631**	**0.629**	0.037	**0.272**	0.563	0.576
	BBAG	KNU+RUS	0.67	0.039	0.584	0.534	0.038	0.215	0.343	0.348
	RFSM	RNK+RMkNN	0.505	0.01	0.519	0.25	0.036	0.06	0.064	0.064
	BGSM	RNK+RMkNN	0.518	0.012	0.518	0.234	**0.038**	0.054	0.056	0.056
	RUSB	RNK+RUS	0.572	0.012	0.558	0.558	0.026	0.218	0.557	0.576
	EASY	KNU+RMkNN	0.65	0.051	0.62	0.619	0.034	0.262	**0.577**	**0.592**
	BROT	STATIC	0.662	0.04	0.605	0.604	0.032	0.25	0.557	0.572
	SMTB	KNU+RUS	0.589	0.012	0.54	0.434	0.031	0.149	0.218	0.22
	LOGR	STATIC	0.649	0.037	0.599	0.595	0.033	0.243	0.515	0.528
	ANN	STATIC	0.502	0.003	0.502	0.045	0.008	0.004	0.004	0.004
	LSVM	STATIC	0.5	0.0	0.5	0.0	0.0	0.0	0.0	0.0
	XGB	STATIC	**0.683**	0.04	0.597	0.581	0.035	0.237	0.451	0.46
	RNDF	STATIC	0.664	0.021	0.546	0.275	0.019	0.112	0.185	0.188
	SVM	STATIC	0.5	0.0	0.5	0.0	0.0	0.0	0.0	0.0

Balanced Rotation Forest(BROT), Easy Ensemble (EASY), and Balanced Bagging (BBAG). The next three best ranking positions are combinations that use Random Undersampling (RUS) to generate the dynamic selection dataset (DSEL). They are: SMOTEBoost (SMTB), RUSBoost (RUSB), and Random Forest SMOTE (RFSM). Only the last position, Bagging SMOTE (BGSM), uses SMOTE to generate the DSEL. Figure 8 highlights these last four combinations in yellow.

Another important finding is the performance of KNORA-Union (KNU). This dynamic ensemble selection technique is in the six best ranking combinations, BRND, BROT, EASY, BBAG, RUSB, and RFSM. The last two ranking positions are combinations that use KNORA-Elimination (KNE) and Local Class Accuracy (LCA). This result demonstrates that KNU is an excellent technique to combine with imbalanced pool generators to address imbalanced datasets.

With these experiments, we observe that RMkNN improves the local region definition in a dynamic selection technique. We also observe that KNORA-Union (KNU) is an excellent dynamic selection technique to combine with imbalanced ensembles. After, we observe that BRDF is the best pool generator to combine with KNU.

6.5. *Limitations of the study*

This study presents RMkNN, a new kNN algorithm used by dynamic selection techniques for imbalanced credit scoring datasets. The first apparent issue in this work is the performance of RMkNN. The proposed version of kNN runs two kNN internally. It is slower than the original kNN algorithm.

Another possible limitation is the reduction function proposed in Eq. 7. This reduction function uses only the imbalance ratio. Maybe a better result can be achieved with the inclusion of other variables, such as a complexity measure of the dataset.

7. Conclusions and future work

In this work, we present a study of the credit scoring problem. We assess the combination of Dynamic Selection (DS) methods, data preprocessing, and pool generation ensembles to deal with the imbalanced nature of the credit scoring data sets using a novel approach to define the local regions of a dynamic selection technique.

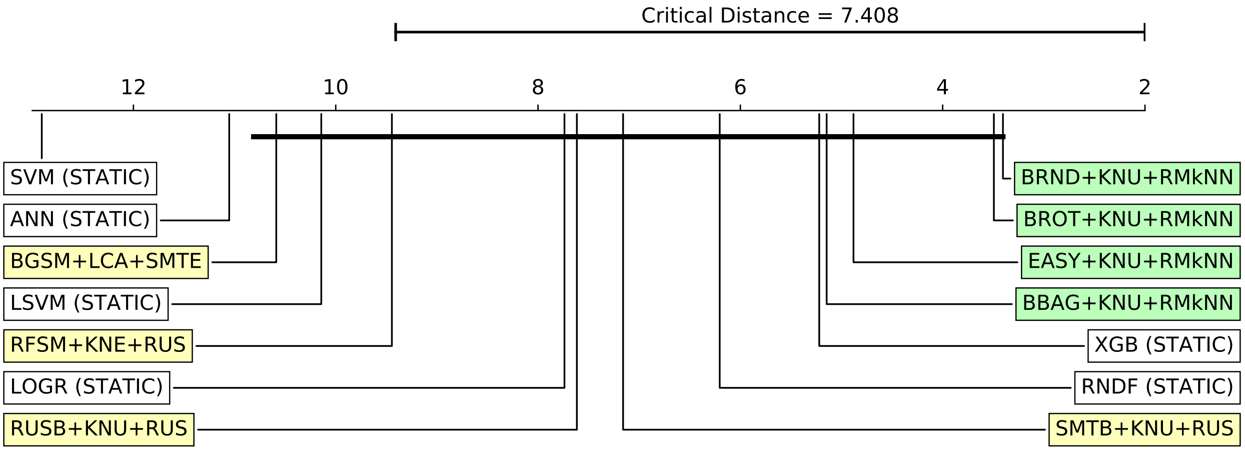

Figure 8: The average rank of the best combinations.

We propose RMkNN to perform a balanced selection of DSEL samples in a dynamic selection technique. To assess the performance of our technique, we compare our proposal with two preprocessing techniques, SMOTE and RUS.

Experiments conducted on seven datasets shown that the credit scoring problem is complex, and dynamic selection techniques are appropriate to this kind of problem. We also find that combining RMkNN with DS techniques enhances the prediction performance according to 8 performance measures. We also reduce a DS technique to a static selection approach. After, we empirically conclude that the KNORA-Union (KNU) is the best DS technique to use in these combinations. Finally, we evaluate our proposed technique in a real-life credit scoring problem to assess that RMkNN outperforms other techniques and classical credit scoring benchmark classifiers.

An interesting future work is to examine the inclusion of dataset complexity scores in the reduction function of RMkNN. There is also a possibility to increase the performance of dynamic selection techniques using heterogeneous ensembles to increase the diversity of base classifiers. We also consider the evaluation of two-step dynamic selection techniques for credit scoring datasets.

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.

Ala'raj, M., & Abbod, M. F. (2016a). Classifiers consensus system approach for credit scoring. *Knowledge-Based Systems*, *104*, 89–105.

Ala'raj, M., & Abbod, M. F. (2016b). A new hybrid ensemble credit scoring model based on classifiers consensus system approach. *Expert Systems with Applications*, *64*, 36–55.

Altman, E. I., Haldeman, R. G., & Narayanan, P. (1977). A new model to identify bankruptcy risk of corporations. *Journal of banking and finance*, *1*, 29–54.

Barandela, R., Valdovinos, R. M., & Sánchez, J. S. (2003). New applications of ensembles of classifiers. *Pattern Analysis & Applications*, *6*, 245–256.

Bastani, K., Asgari, E., & Namavari, H. (2019). Wide and deep learning for peer-to-peer lending. *Expert Systems with Applications*, *134*, 209–224.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*, 123–140.

Britto Jr, A. S., Sabourin, R., & Oliveira, L. E. (2014). Dynamic selection of classifiers—a comprehensive review. *Pattern Recognition*, *47*, 3665–3680.

Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications*, *39*, 3446–3453.

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*, 321–357.

Chen, C., Liaw, A., & Breiman, L. (2004). Using random forest to learn imbalanced data. *University of California, Berkeley*, *110*, 1–12.

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* (pp. 785–794). ACM.

Cruz, R. M. O., Hafemann, L. G., Sabourin, R., & Cavalcanti, G. D. C. (2018). DESlib: A Dynamic ensemble selection library in Python. *arXiv preprint arXiv:1802.04967*, .

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, *7*, 1–30.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International workshop on multiple classifier systems* (pp. 1–15). Springer.

Feng, X., Xiao, Z., Zhong, B., Qiu, J., & Dong, Y. (2018). Dynamic ensemble classification for credit scoring using soft probability. *Applied Soft Computing*, *65*, 139–151.

Fernández, A., Garcia, S., Herrera, F., & Chawla, N. V. (2018). Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, *61*, 863–905.

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, *15*, 3133–3181.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, *11*, 86–92.

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*, 463–484.

García, V., Marqués, A. I., & Sánchez, J. S. (2019). Exploring the synergetic effects of sample types on the performance of ensembles for credit risk and corporate bankruptcy prediction. *Information Fusion*, *47*, 88–101.

Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, *73*, 220–239.

Hand, D. J. (2009). Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine learning*, *77*, 103–123.

Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: a review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *160*.

He, H., Zhang, W., & Zhang, S. (2018). A novel ensemble method for credit scoring: Adaption of different imbalance ratios. *Expert Systems with Applications*, *98*, 105–117.

Ho, T. K. (1995). Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on* (pp. 278–282). IEEE volume 1.

Ho, T. K., & Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, *24*, 289–300.

Ko, A. H., Sabourin, R., & Britto Jr, A. S. (2008). From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, *41*, 1718–1731.

Kuncheva, L. I. (2002). A theoretical study on six classifier fusion strategies. *IEEE Transactions on pattern analysis and machine intelligence*, *24*, 281–286.

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, *18*, 1–5. URL: http://jmlr.org/papers/v18/16-365.html.

Lessmann, S., Baesens, B., Mues, C., & Pietsch, S. (2008). Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE Transactions on Software Engineering*, *34*, 485–496.

Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, *247*, 124–136.

Liaw, A., Wiener, M. et al. (2002). Classification and regression by randomforest. *R news*, *2*, 18–22.

Liu, X.-Y., Wu, J., & Zhou, Z.-H. (2009). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *39*.

Melo Jr, L., Macedo, J. F., Nardini, F. M., & Renso, C. (2019a). 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). IEEE.

Melo Jr, L., Nardini, F. M., Renso, C., & Macedo, J. A. (2019b). On combining dynamic selection, sampling, and pool generators for credit scoring. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*. ibai-publishing.

Nemenyi, P. (1962). Distribution-free multiple comparisons. In *Biometrics* (p. 263). International Biometric Soc 1441 I ST, NW, Suite 700, Washington, DC 20005-2210 volume 18.

Opitz, D., & Maclin, R. (1999). Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, *11*, 169–198.

Roy, A., Cruz, R. M., Sabourin, R., & Cavalcanti, G. D. (2018). A study on combining dynamic selection and data preprocessing for imbalance learning. *Neurocomputing*, *286*, 179–192.

Sabourin, M., Mitiche, A., Thomas, D., & Nagy, G. (1993). Classifier combination for hand-printed digit recognition. In *Document Analysis and Recognition, 1993., Proceedings of the Second International Conference on* (pp. 163–166). IEEE.

Sabzevari, H., Soleymani, M., & Noorbakhsh, E. (2007). A comparison between statistical and data mining methods for credit scoring in case of limited available data. In *Proceedings of the 3rd CRC Credit Scoring Conference* (pp. 1–5). Citeseer.

Serrano-Cinca, C., & Gutiérrez-Nieto, B. (2016). The use of profit scoring as an alternative to credit scoring systems in peer-to-peer (p2p) lending. *Decision Support Systems*, *89*, 113–122.

Sousa, M. R., Gama, J., & Brandão, E. (2016). A new dynamic modeling framework for credit risk assessment. *Expert Systems with Applications*, *45*, 341–351.

Sun, J., Lang, J., Fujita, H., & Li, H. (2018). Imbalanced enterprise credit evaluation with dte-sbd: Decision tree ensemble based on smote and bagging with differentiated sampling rates. *Information Sciences*, *425*, 76–91.

Thomas, L., Crook, J., & Edelman, D. (2017). *Credit scoring and its applications* volume 2. Siam.

Thomas, L., Oliver, R., & Hand, D. (2005). A survey of the issues in consumer credit modelling research. *Journal of the Operational Research Society*, *56*, 1006–1015.

West, D. (2000). Neural network credit scoring models. *Computers & Operations Research*, *27*, 1131–1152.

Woods, K., Kegelmeyer, W. P., & Bowyer, K. (1997). Combination of multiple classifiers using local accuracy estimates. *IEEE transactions on pattern analysis and machine intelligence*, *19*, 405–410.

Xia, Y., Liu, C., Da, B., & Xie, F. (2018). A novel heterogeneous ensemble credit scoring model based on bstacking approach. *Expert Systems with Applications*, *93*, 182–199.

Xia, Y., Liu, C., Li, Y., & Liu, N. (2017). A boosted decision tree approach using bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications*, *78*.

Xiao, H., Xiao, Z., & Wang, Y. (2016). Ensemble classification based on supervised clustering for credit scoring. *Applied Soft Computing*, *43*, 73–86.

Xiao, J., Xie, L., He, C., & Jiang, X. (2012). Dynamic classifier ensemble model for customer classification with imbalanced class distribution. *Expert Systems with Applications*, *39*.