Machine Learning with Applications 9 (2022) 100349

Contents lists available at ScienceDirect

Machine Learning with Applications

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

# Application of the VNS heuristic for feature selection in credit scoring problems

Victor Gomes Helder [*], Tiago Pascoal Filomena, Luciano Ferreira, Guilherme Kirch

*School of Administration, Federal University of Rio Grande do Sul, 855 Washington Luis Street, Porto Alegre, 90010-460, Rio Grande do Sul, Brazil*

ARTICLE INFO

*Keywords:*
Credit scoring
Machine learning
Feature selection

ABSTRACT

Credit scoring plays a major role for financial institutions when making credit-granting decisions. In this context, machine learning techniques have been used to develop credit scoring models, as they seek to recognize existing patterns in databases containing the credit history of borrowers to infer potential defaulters. However, these databases often contain a large number of variables, some of which can be noisy, leading to imprecise results and loss of performance/accuracy. In the present work, a feature selection technique is proposed based on a variable neighborhood concept, so-called VNS. The applicability of the method is assessed in conjunction with seven of the main techniques used to make default prediction in credit analysis problems. Its performance was compared to the feature selection obtained by the well-known PCA statistical method. The results indicate superior performance of the VNS in most of the applied tests, suggesting the robustness of the method.

## 1. Introduction

Granting credit plays an important role in today's economy. Often, individuals are only able to acquire more expensive goods through loans, and most businesses only start after some form of credit has been granted. Credit operations are, therefore, present in almost all types and scales of financial transactions.

Due to the large volume of capital associated with credit, it is of the utmost importance that credit is granted responsibly and pragmatically. As such, tools have been developed to help analyze the data of those who apply for loans. In this context, credit scoring has revealed itself as a mean to classify individuals into “good" or “bad" payers according to their propensity to repay their loans (Hand & Henley, 1997). There is a wide range of techniques that can be used to compose a credit scoring model. In general, these are based on the analysis and classification of a database containing various information about borrowers, so that, based on this analysis, it is possible to classify new data into the categories of interest.

In problems involving credit analysis and credit scoring, it is common to come across databases that have a large number of variables related to borrowers. However, in many cases, a significant number of variables can mean too much information, generating noise in the database. In this context, feature selection presents itself as a way to simplify a database by identifying key features, reducing computational cost and improving the prediction performance (Zhang, He, & Zhang, 2019). Several papers explore different feature selection approaches, some of them in the credit scoring context. For instance, Chen and Li (2010) proposed the comparison of four methods of feature selection, being them Linear Discriminant Analysis, Rough Sets Theory, Decision tree and F-score, using Support Vector Machine (SVM) as the base classifier. The authors also suggest that using a heuristic could be a good alternative to address the problem. Wang, Hedar, Wang, and Ma (2012) proposed the use of a feature selection method, called RSFS, based on rough sets and scatter search, which is an algorithm based on the recombination of samples to generate a group of solutions.

An interesting alternative to perform a feature selection are the metaheuristic search techniques, since they explore the solution space of problems (Talbi, 2009), which helps in reaching solutions close to the optimum. In this sense, Variable Neighborhood Search (VNS) is a metaheuristic proposed by Mladenović and Hansen (1997) based on the concept of exploring the neighborhoods of the incumbent solution. According to Talbi (2009), the basic idea of VNS is to explore a pre-defined set of neighborhoods to arrive at the best solution. The method can explore the set of neighborhoods randomly or systematically to reach different local optimums. As such, VNS takes advantage of the prerogative that if several neighborhoods are used, one of them may eventually contain the global optimum. The VNS can be used in different approaches, Marinaki and Marinakis (2015) developed a hybridized version of the clonal selection algorithm by combining it with the VNS and the iterated local search. García-Torres, Gómez-Vela, Melián-Batista, and Moreno-Vega (2016) proposes a strategy for

[*] Corresponding author.
*E-mail addresses:* victor.helder@ufrgs.br (V.G. Helder), tiago.filomena@ufrgs.br (T.P. Filomena), ferreira.luciano@ufrgs.br (L. Ferreira), gkirch@ufrgs.br (G. Kirch).

https://doi.org/10.1016/j.mlwa.2022.100349
Received 29 March 2022; Received in revised form 18 May 2022; Accepted 31 May 2022
Available online 8 June 2022

variable reduction in high-dimensional databases that consists of reducing the input space by grouping correlated features using the concept of Markov blankets, and then applying the VNS to perform the feature selection. Boughaci and Alkhawaldeh (2018) made a comparison of Local Search (LS), Stochastic Local Search (SLS) and VNS for the feature selection problem in credit scoring, utilizing Support Vector Machine (SVM) as the base classifier, and they conclude that the VNS had the better performance among the three methods analyzed.

Aiming to contribute with a little-explored alternative to select variables in credit scoring problems, in this paper we propose using the Variable Neighborhood Search (VNS) metaheuristic to improve default prediction performance. Seven machine learning techniques are tested in conjunction with the VNS in order to infer the most adequate to be used as the base classifier. To reduce the probability that the analyzed data may cause a bias in the results, the comparison will be performed using five different credit databases publicly available in digital repositories. We show that using VNS in the feature selection step can significantly improve credit scoring models performance and thus our proposed machine learning based algorithm can help firms in granting credit to customers in better terms and avoiding unnecessary credit risk at the same time.

The papers closest to ours are Wang et al. (2012) and Boughaci and Alkhawaldeh (2018). They also analyze the performance of feature selection methods combined with machine learning algorithms for credit scoring. Different from them, we use VNS for feature selection, a wide range of machine learning algorithms, and a broad set of credit databases to evaluate performance. We believe that these improvements allow us to reach more generalizable conclusions and, as we expand on these previous findings, broaden our understanding of the importance of feature selection for credit scoring models in particular and for machine learning algorithms in general.

Section 2 presents a brief review of the concepts of credit scoring, feature selection, and the classifiers used in this work. In Section 3, the methodology is described, comprising the framework of the feature selection method, the evaluation criteria adopted, and the databases used. Section 4 describes the results of the application of the proposed methodology.

## 2. Fundamental concepts

This section presents some concepts of credit scoring, feature selection and methods of machine learning used in this work.

### 2.1. Credit scoring

Credit scoring can be defined as a set of decision models that aid lenders in the granting of consumer credit (Thomas, Crook, & Edelman, 2017). Traditionally, credit scoring is used to estimate the default risk. According to Thomas (2009), default risk is usually considered to be the chance that an applicant will go 90 days overdue on their payments in the next 12 months.

The idea of using credit scoring is to make the decision process consistent and automatic (Thomas, 2009), which is especially useful in the presence of a large volume of applicants. Still according to the author, the philosophy behind credit scoring is pragmatic, as it seeks only to improve the predictive power, and not to explain the factors behind the predictions. Credit scoring models reduce credit analysis costs, contribute to granting decisions, and save time and effort (Ong, Huang, & Tzeng, 2005).

Currently, the use of credit scoring by financial institutions is widespread. West (2000) estimated that 97% of banks that approved credit card applications used some form of credit scoring model at the time of the article publication. In terms of academic research, it is also clear that there is great interest in the field; in this regard, Huang (2015) points to a significant increase in research on the subject as of 2009, i.e., right after the 2008 subprime crisis.

There are several techniques that can be applied to generate a credit score, but one vital point they all have in common is the analysis of a large sample containing information about previous customers and their credit history (Thomas et al., 2017). The techniques seek to connect the characteristics of individuals with their respective histories, usually separating them into ‘‘good" or ‘‘bad" payers; i.e., those who did not default and those who did in the period considered. The successful application of some of the first credit scoring techniques attracted the attention of academics and researchers to develop advanced statistical methods and machine learning techniques (Van Gestel & Baesens, 2008). In this context, the information learned from the databases serves as a reference for the creation of a model that predicts the default risk of future applicants. For this paper, seven well-known machine learning techniques were considered, and they are presented as follows.

### 2.2. Predictive methods

Logistic Regression (LR) is one of the most popular methods when it comes to credit scoring (Lessmann, Baesens, Seow, & Thomas, 2015), and was developed specifically for use in situations where the output variables are binary (Cox, 1958). LR is based on the maximum likelihood method (Thomas et al., 2017), which determines which parameters have the highest probability of producing the observed data.

The k-Nearest Neighbors (k-NN) method is a nonparametric technique that classifies a data point by computing the votes of its $k$ nearest neighbors (Brown & Mues, 2012). The choice of the value of $k$ is highly dependent on the database, nevertheless, Lessmann et al. (2015) recommends that the value of $k$ be an odd number to avoid ties. Another parameter to be set is the distribution of the weights among neighbors, which can be uniform or proportional to their distance from the sample under analysis. Regarding the distance metric, Hu, Huang, Ke, and Tsai (2016) points out that the Euclidean distance is the most widely used with the k-NN method.

Bootstrap Aggregation or Bagging is an ensemble learning algorithm based on data diversification. In this sense, diversification of the training data occurs by generating a certain number (parameter to be set by the programmer) of subsets, each of which is randomly generated from the original database (Ala’raj & Abbod, 2015). At each iteration, Bagging redefines the training subset with replacement. After processing all the iterations, each trained subset will provide one vote, and the best strategy will therefore be chosen by counting all the votes.

Boosting is an ensemble method that aims to improve the performance of a learning algorithm (Freund, Schapire, et al., 1996). The best-known algorithm of the Boosting family is the AdaBoost (Marqués, García, & Sánchez, 2012), which has no random elements and generates a set of trees by successively rebalancing the weights of the training data, with the weights of each iteration depending on the history of previous iterations (Breiman, 2001), and the number of iterations being a parameter to be set.

Random Forest (RF) consists of a procedure in which several decision trees are generated, each contributing with one vote, and the final result is obtained by summing the votes (Breiman, 2001). According to Lorena et al. (2011), for each tree, a new training set is obtained by randomly rearranging the previous set with bootstrap sampling replacement. The Random Forest technique has two main parameters that require tuning, namely the number of trees generated and the number of attributes in the subsamples (Brown & Mues, 2012).

The Support Vector Machine (SVM) method consists of a classifier model in which binary classified data is separated by a hyperplane such that the margin between the hyperplane and the inputs is maximized (Bellotti & Crook, 2009). The performance of SVM is highly influenced by the values of kernel parameter and types of kernel function that are selected in the training task (Ab Hamid, Sallehuddin, Yunos, & Ali, 2021). According to Bellotti and Crook (2009), linear, polynomial and radial basis are among three of the most widely used kernel's functions. Moreover, another important aspect when implementing the SVM is the parameter C, which controls the margin of the hyperplane and represents a penalty used in misclassification cases.

An artificial neural network consists of a number of small primitive processing units interconnected by weighted direct connections. Each unit receives input signals via weighted input connections and responds by sending a signal to all units that have output connections (Masson & Wang, 1990). Multi-Layer Perceptron (MLP) is one of the most common neural network architectures, and it consists of an input layer, one or more hidden layers, and an output layer (Bao, Lianju, & Yue, 2019). Ala'raj and Abbod (2016) point out that the number of neurons in the hidden layer and the rate of readjustment of the weights (learning rate) are the main parameters that require tuning.

Beyond the classifiers selected for this work, there are a wide variety of other methods that can be explored in future studies, such as Multivariate Adaptive Regression Splines (MARS) which is based on a statistical spline model (Ju, Rosenberger, Chen, & Liu, 2022) and Classification and Regression Trees (CART), a tree-based classifier that uses the *Gini* coefficient to guide tree growing (Lessmann et al., 2015).

### 2.3. Feature selection

Feature selection is an NP-hard problem that involves finding an optimal subset from the original features that leads to the best predictive performance (Peng, Albuquerque, Kimura, & Saavedra, 2021). According to Wang et al. (2012), feature selection is one of the most fundamental problems in the field of machine learning. The authors further state that credit databases often have a large number of variables, which can lead to a model with excessive complexity but without remarkable accuracy. In this sense, it is advantageous to limit the number of input variables in a classifier in order to obtain a minimum set of attributes that retain the relevant information for effective prediction.

Variable selection algorithms can be classified into two categories: filter approach and wrapper approach (Chen & Li, 2010). The filter approach selects a set of variables independent of the classifier so that filtering is based on features observed in the training phase. The wrapper approach, on the other hand, generally uses an evaluation criterion associated with a predetermined learning algorithm to determine the sets to be selected. Chen and Li (2010) also state that wrapper methods usually obtain better results in finding sets with the most relevant variables, which is natural since wrapper methods work in conjunction with the classifier. It is also possible to combine both filter and wrapper approaches, as presented by Beuren and Anzanello (2019), in which a Mutual Information technique is used to remove less significant variables (filter phase), and then three non-parametric tests (Anderson–Darling, Kruskal–Wallis and Steel's Test) are used to rank the remaining variables (wrapper phase).

Many wrapper methods are based on heuristic search techniques. A heuristic can be defined as a technique that seeks to find a good solution (local optimum) at a reasonable computational cost, but without guaranteeing the achievement of an optimal solution (global optimum) (El-Sherbeny, 2010). The search for a solution or an optimal set generally involves the application of techniques that allow the algorithm to continue searching for alternatives even after reaching a local optimum point. In this context, metaheuristic techniques emerge as interesting solution options. According to El-Sherbeny (2010), a metaheuristic is an iterative strategy that guides and modifies the operations of subordinate heuristics by intelligently combining different concepts to explore the search space.

## 3. Methodology

In this section, some details about the databases are presented, subsequently, the procedure of the proposed algorithm for feature selection is described, as well as details about the implementation and adjustment of the parameters of the predictive methods, and then the adopted comparison criteria are discussed.

**Table 1**
Main information regarding the databases used.

| Database | Instances | Default/Not default | Variables |
|---|---|---|---|
| AC | 690 | 383/307 | 14 |
| GC (num.) | 1000 | 300/700 | 24 |
| JC | 689 | 383/306 | 15 |
| TC | 30000 | 6636/23364 | 23 |
| AER | 1319 | 296/1023 | 11 |

**Table 2**
Descriptive Statistics of the features that are common to at least two databases.

| Variable | GC | TC | AER |
|---|---|---|---|
| Amount of given credit[a] | 3271 (28223) | 167484 (129747) | NA |
| Credit card expenditure[a] | NA | 51223 (73636) | 185.06 (272.22) |
| Credit history[a] | 2.54 (1.08) | 0.36 (0.76) | 0.46 (1.34) |
| Age | 35.55 (11.38) | 35.48 (9.22) | 33.21 (10.14) |
| Gender (male/fem.) | 690/310 | 11888/18112 | NA |
| Married (Yes/No)[a] | 402/598 | 13659/16341 | NA |
| Home status (own/rent) | 713/287 | NA | 581/738 |
| Time at current address[a] | 2.84 (1.1) | NA | 55.27 (66.27) |

For interval or ratio variables we present the mean and standard deviation (in parenthesis). For nominal variables we present the frequency of each category, separated by a slash bar.

[a] The features are not exactly the same, are expressed in different currencies, or do not refer to the same period across different datasets, but their meaning are similar enough to be categorized as the same.

### 3.1. Databases

Credit databases basically consist of grouping certain information about individuals who have obtained loans. In this context, each instance (row) represents a person, while each column indicates some information or characteristic about that person. The information contained varies according to the database, but one indispensable data category of interest to this study is the default, which is a binary variable that indicates whether the individual has defaulted in a given period. A widely accepted value for default is a 90-day delay in payment.

Five databases were used for this study, all of them public databases found in digital repositories. The Australian Credit Approval (AC) database is one of the most widespread in the credit literature, it has 14 input variables and 690 instances and concerns credit card applications. Another widely used database is the German Credit Data (GC), which classifies people described by a set of attributes as good or bad credit risks. This dataset has an alternative version, which was used in this study, containing only numerical data, resulting in 24 variables and 1000 instances. Another database considered was the Japanese Credit Screening Data Set (JC), which contains 689 instances and 15 variables. The fourth database is the Taiwan Default of Credit Card Clients Data Set (TC), which has 30000 instances and 23 variables. All the above databases were obtained free of charge from the UCI Machine Learning Repository. We also used a database obtained from the Kaggle platform, the Credit Card Data (AER) base, taken from the book "Econometric Analysis", which has 1319 instances and 11 input variables related to credit card applications. Table 1 summarizes the main information mentioned above.

In relation to the features that make up each database, it is stated on the UCI Machine Learning Repository website that, in order to protect the confidentiality of the data, both AC and JC have their attributes names and values changed to meaningless symbols. The other three databases (GC, TC, and AER) have variables with meaningful names, in this regard, each dataset has a unique set of features, although some of them are common to more than one base. Table 2 shows a list of features that exist in at least two of the datasets and some descriptive statistics.

### 3.2. VNS

As the name suggests, the Variable Neighborhood Search (VNS) algorithm searches for the best solution by exploring a certain number of $k$ neighborhoods (Hansen, Mladenović, Todosijević, & Hanafi, 2017). In our VNS approach, a candidate solution can be represented as a binary vector of $n$ positions, where $n$ indicates the number of variables or features to be considered. To represent a solution, we define the following binary notation: if a variable is present in the solution, the value 1 is assigned to the corresponding position, otherwise a value 0 is used. During the experiments, eight neighborhoods structures ($k = 8$) were defined, each one of them with an inner routine to select features. In the first structure ($N_1$), candidate solutions are obtained by modifying, sequentially, one bit of the current solution ($x_r$). For example, if $x_r = 11111$, then the neighbor solutions for $N_1$ are: $01111, 10111, 11011, 11101, 11110$.

The second and third neighborhoods structures are analog to the first, but they change two and three variables sequentially and simultaneously. The fourth structure changes one random variable. The fifth, sixth, seventh and eighth are analog to the fourth structure, but they change, respectively, two, three, four, and five variables randomly. Next, following the corresponding structure, the algorithm performs a search through the select neighborhood. After the search, the best solution found is compared with the incumbent solution. If the new solution is better, it assumes the position of the incumbent solution, and a counter is reset to zero, otherwise, the solution is not updated and one unit is added to the counter. The process is finished (stopping criteria) when the counter reaches a pre-determinate value, which was set at 100 iterations without improvement, so the set of variables that achieved the best accuracy is adopted.

The local search procedure starts with a random solution $x_r$ and tries to find a better solution $x_i$ in the current neighborhood. Similar to $N_1$, the neighboring solution $x_i$ of the solution $x_r$ is obtained by modifying one variable. Among the neighbors, we select for the next iteration the one with the best accuracy. The function $f(x)$ calculates the accuracy of $x$ regarding each classifier used in our experiments. As this procedure can be highly time-consuming, we save the computed values of $f(x)$ in a hash table for use in other iterations and runs, generating historical data to improve the performance of our algorithm. Algorithm 1 presents a synthesis of the VNS used in our paper.

**Algorithm 1** Feature Selection VNS.

```
Input : a set of neighborhood structures $N_k$ for $k = 1, 2, ..., k_{max}$, and a
        initial solution $x$.
  Repeat
        $k = 1$
        Repeat
              Shaking: choose a random solution $x_r$ from $N_k(x)$
              $x_i = localSearch(x_r)$
              If $f(x_i) > f(x)$ Then
                    $x = x_i$
                    $k = 1$
              Else $k = k + 1$
        Until $k = k_{max}$
  Until Stopping criteria
Ouput : Best found solution.
```

### 3.3. Implementation and parameter adjustment

The problem was implemented and developed in the Python programming language (version 3.6.8). The implementation of the prediction methods was done by using the Scikit-Learn library, which is an open-source machine learning library containing various classification and regression algorithms.

**Table 3**
Confusion matrix.

| | Predict as good | Predict as bad |
|---|---|---|
| Actually good | TP | FN |
| Actually bad | FP | TN |

Predictive methods can adapt differently depending on the database used, so it is useful that their parameters are set in order to achieve the best possible performance for a classifier in each database. The grid search technique, which is commonly found on literature (Bellotti & Crook, 2009; Brown & Mues, 2012), was used to define the parameters of the methods mentioned in Section 2.2, except for the Logistic Regression, which does not have any parameter to be set (Baesens, Van Gestel, Viaene, Stepanova, Suykens, & Vanthienen, 2003; Brown & Mues, 2012).

Before running any test, the data is split into train and test, being 75% of the data, which is an intermediate value to those found in the literature, used as the training set, and the remaining 25% used as the testing set. Methods that are based on distance comparisons, such as the k-NN, may lose efficiency if different variables have considerably different ranges between their values, therefore, a normalization process using the Min-Max scaler was applied to the features in all databases. When no feature selection is applied, each one of the classifiers is fitted to all the variables of the training set of a given dataset and generates a model, which is then used to predict the labels of the testing set. When the VNS is applied, the iterative process described in the previous section is carried out for each one of the classifiers. Fig. 1 shows the flowchart of the procedure applying feature selection. Summing up, each one of the seven classifiers is tested without features selection (all features) and with VNS, for each one of the five databases.

### 3.4. Evaluation criteria

To evaluate the performance of the classifiers with and without feature selection, each combination of classifier and database is repeated 20 times. In each case, the labels of the testing set predicted by the model are compared with the real values. Four mutually exclusive possibilities emerge from the comparison between the defaults predicted and the actual defaults, which are true positive (TP), true negative (TN), false positive (FP), and false negative (FN), which comprise the so-called confusion matrix, as shown in Table 3.

There are several metrics for comparing the performance of different predictive methods. One of the most elementary and widely used is accuracy, which can be represented by Eq. (1):

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Considering the previous formulation, one can see that accuracy indicates the percentage of correct predictions in relation to the total, but it is a somewhat incomplete measure in the sense that it does not indicate whether misclassifications have a bias; i.e., with accuracy alone, there is no perception if there is a predominance of false positives or false negatives.

Precision and Recall are two metrics complementary to accuracy in problems involving Machine Learning. According to Géron (2017), the former represents the percentage of correctly predicted non-defaults in relation to the total number of instances predicted as non-defaults, i.e., it is the accuracy of positive predictions. Still according to the author, Recall, or sensitivity, indicates the proportion of correctly predicted non-defaults in relation to the total of non-defaults, that is, the proportion of positive instances correctly detected by the classifier. Represented mathematically, we get Eqs. (2) and (3):

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

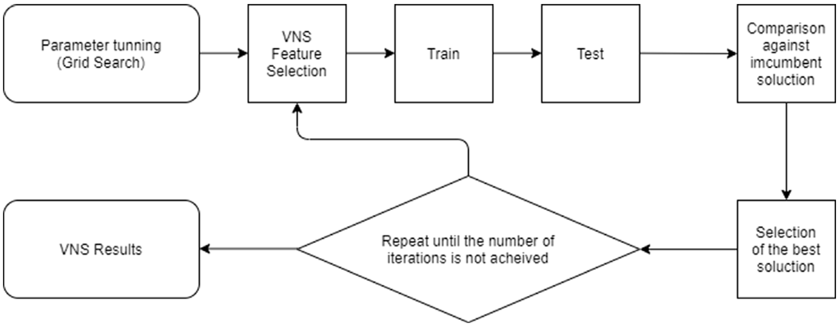
Parameter tunning (Grid Search)
VNS Feature Selection
Train
Test
Comparison against imcumbent soluction
Selection of the best soluction
Repeat until the number of iterations is not acheived
VNS Results

**Fig. 1.** Flowchart of the proposed methodology applying the VNS.

**Table 4**
Australian Dataset: VNS and no variable selection.

| Classifier | Full set | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.8610 | 0.91 | 0.83 | 0.924 | 0.8746 | 0.92 | 0.84 | 0.923 | 0.083 |
| k-NN | 0.8538 | 0.91 | 0.82 | 0.909 | 0.8725 | 0.92 | 0.84 | 0.917 | 0.015 |
| Bagging | 0.7604 | 0.87 | 0.67 | 0.903 | 0.8809 | 0.91 | 0.87 | 0.929 | <0.001 |
| Boosting | 0.8529 | 0.89 | 0.84 | 0.924 | 0.8752 | 0.93 | 0.84 | 0.925 | 0.011 |
| RF | 0.8604 | 0.88 | 0.87 | 0.924 | 0.8873 | 0.91 | 0.89 | 0.928 | 0.001 |
| SVM | 0.8491 | 0.93 | 0.79 | 0.923 | 0.8491 | 0.93 | 0.79 | 0.923 | 1 |
| NN | 0.8549 | 0.88 | 0.85 | 0.925 | 0.8801 | 0.91 | 0.87 | 0.924 | 0.006 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the full set of variables. Columns six to nine present the same indicators, but applying the VNS. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

**Table 5**
German Dataset: VNS and no variable selection.

| Classifier | Full set | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.7636 | 0.79 | 0.89 | 0.796 | 0.7768 | 0.80 | 0.91 | 0.793 | 0.080 |
| k-NN | 0.7268 | 0.74 | 0.94 | 0.764 | 0.7674 | 0.78 | 0.93 | 0.775 | <0.001 |
| Bagging | 0.7476 | 0.79 | 0.87 | 0.774 | 0.7726 | 0.80 | 0.89 | 0.774 | 0.007 |
| Boosting | 0.7456 | 0.79 | 0.86 | 0.766 | 0.7694 | 0.81 | 0.88 | 0.772 | 0.003 |
| RF | 0.7514 | 0.78 | 0.89 | 0.781 | 0.7786 | 0.80 | 0.91 | 0.777 | 0.002 |
| SVM | 0.7608 | 0.80 | 0.87 | 0.792 | 0.7736 | 0.80 | 0.89 | 0.793 | 0.136 |
| NN | 0.7548 | 0.79 | 0.88 | 0.792 | 0.7754 | 0.80 | 0.90 | 0.789 | 0.003 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the full set of variables. Columns six to nine present the same indicators, but applying the VNS. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Another measure commonly found in the literature is the Receiver Operating Characteristics (ROC) Curve and its corresponding Area Under the Curve (AUC). The vertical axis of the ROC Curve is called True Positive Rate (TPR), which is obtained in the same way as Recall, while the horizontal axis is called False Positive Rate (FPR), and it can be calculated with Eq. (4):

$$FPR = \frac{FP}{FP + TN} \tag{4}$$

The area under the plotted curve has a dimension that varies between 0 and 1. A completely random model is expected to have an area of 0.5, and the better the results are predicted, the closer the area under the curve is to 1.

## 4. Results and discussion

In this section, a comparison of the predictions performed with and without the VNS are presented. In order to have a reference of another feature selection technique, a Principal Component Analysis (PCA) was implemented, and its results were compared with those obtained in conjunction with the VNS. Yao, Zhang, and Wang (2018) also used PCA for feature selection in their study of financial statement fraud detection models. More precisely, these authors compare PCA and Xgboost to do feature selection, combined with five machine learning algorithms. All our tests were performed using a computer with a 7th generation Intel Core i5 processor, 2.5 GHz and 8 GB of RAM.

### 4.1. Results without feature selection and with VNS

Initially, the idea was to compare the performance of the classifiers when associated with the VNS and utilizing all the variables. In both cases, each combination of classifier and database was submitted to twenty rounds of tests. Tables 4 to 8 show the average results obtained in the AC, GC, TC, JC and AER databases, respectively. The column *p-value* in each of these tables presents significance probability relative to accuracy comparison. In this regard, Bhattacharyya, Jha, Tharakunnel, and Westland (2011), in their study involving the comparison of Logistic Regression, SVM and Random Forest, considered a *p*-value less than 0.01 (p <0.01) as a significant difference.

When comparing the results of the predictions obtained by classifiers without any variable selection technique and with the presence of VNS, one can see that in almost all situations (except those using SVM) there was an improvement in terms of accuracy with the application of variable selection. However, in most of these cases, only marginal and

**Table 6**
Taiwan Dataset: VNS and no variable selection.

| Classifier | Full set | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.8022 | 0.81 | 0.97 | 0.714 | 0.8056 | 0.81 | 0.98 | 0.708 | 0.048 |
| k-NN | 0.7915 | 0.80 | 0.97 | 0.678 | 0.8111 | 0.83 | 0.95 | 0.708 | <0.001 |
| Bagging | 0.8117 | 0.84 | 0.94 | 0.741 | 0.8147 | 0.84 | 0.94 | 0.741 | 0.023 |
| Boosting | 0.8149 | 0.83 | 0.96 | 0.758 | 0.8203 | 0.84 | 0.96 | 0.751 | <0.001 |
| RF | 0.8147 | 0.84 | 0.94 | 0.751 | 0.8174 | 0.84 | 0.95 | 0.749 | 0.026 |
| SVM | 0.7776 | 0.78 | 1 | 0.695 | 0.7776 | 0.78 | 1 | 0.689 | 0.994 |
| NN | 0.8168 | 0.84 | 0.95 | 0.748 | 0.8217 | 0.84 | 0.95 | 0.748 | <0.001 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the full set of variables. Columns six to nine present the same indicators, but applying the VNS. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

**Table 7**
Japan Dataset: VNS and no variable selection.

| Classifier | Full set | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.8552 | 0.91 | 0.81 | 0.916 | 0.8659 | 0.92 | 0.82 | 0.915 | 0.259 |
| k-NN | 0.8549 | 0.90 | 0.83 | 0.912 | 0.8769 | 0.92 | 0.85 | 0.917 | 0.028 |
| Bagging | 0.8159 | 0.86 | 0.79 | 0.912 | 0.8856 | 0.90 | 0.89 | 0.926 | 0.001 |
| Boosting | 0.8459 | 0.89 | 0.82 | 0.927 | 0.8812 | 0.92 | 0.86 | 0.930 | 0.002 |
| RF | 0.8601 | 0.88 | 0.87 | 0.927 | 0.8957 | 0.92 | 0.89 | 0.931 | <0.001 |
| SVM | 0.8543 | 0.93 | 0.80 | 0.912 | 0.8543 | 0.93 | 0.80 | 0.913 | 1 |
| NN | 0.8416 | 0.90 | 0.80 | 0.917 | 0.8725 | 0.93 | 0.83 | 0.918 | 0.004 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the full set of variables. Columns six to nine present the same indicators, but applying the VNS. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

**Table 8**
AER Dataset: VNS and no variable selection.

| Classifier | Full set | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.8497 | 0.84 | 0.99 | 0.951 | 0.8580 | 0.85 | 1 | 0.951 | 0.249 |
| k-NN | 0.8470 | 0.86 | 0.96 | 0.869 | 0.9815 | 1 | 0.98 | 0.989 | <0.001 |
| Bagging | 0.9789 | 0.99 | 0.98 | 0.994 | 0.9880 | 1 | 0.98 | 0.994 | <0.001 |
| Boosting | 0.9815 | 0.99 | 0.98 | 0.996 | 0.9865 | 1 | 0.98 | 0.995 | 0.003 |
| RF | 0.9823 | 1 | 0.98 | 0.995 | 0.9871 | 1 | 0.98 | 0.995 | 0.016 |
| SVM | 0.9653 | 0.99 | 0.97 | 0.996 | 0.9750 | 1 | 0.98 | 0.995 | 0.008 |
| NN | 0.9633 | 0.98 | 0.97 | 0.992 | 0.9809 | 1 | 0.98 | 0.992 | <0.001 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the full set of variables. Columns six to nine present the same indicators, but applying the VNS. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

statistically not significant differences were found. Even so, in certain situations, variable selection proved to be highly relevant in improving results, particularly when using k-NN on the AER database, as shown in Table 8, where the average accuracy was increased by more than 13% and the AUC increased by 0.12. A considerable improvement can also be observed in the use of the Bagging method in the AC database, which can be seen in Table 4, where an average accuracy of about 12% higher was recorded with the use of VNS variable selection. This same method also showed a significant improvement with the use of VNS in the JC database, as seen in Table 7, with the average accuracy increasing by almost 7%.

It should be noted that some methods are more or less sensitive to the application of a variable selection technique than others. In this regard, the SVM method stands out as the least affected by the use of feature selection techniques. It can be clearly observed in Tables 4, 6 and 7, that the accuracy values with the full set of variables, or with the VNS, were nearly the same, and the *p*-value was 1, or very close to 1, indicating that in those cases there was no noticeable effect with the use of VNS for the SVM classifier.

When analyzing only the performance of the classifiers without performing any variable selection, the Boosting, Random Forest and Neural Networks methods were the ones with the most consistent performance over the five databases, since the other four classifiers had a considerably inferior performance in at least one dataset in comparison to the methods mentioned above. These evidences corroborate the findings of Severino and Peng (2021), who showed that Random Forest, Gradient Boosting, and Deep Neural Networks had superior performance to predict property insurance fraud in a major Brazilian insurance company than other machine learning classifiers.

Regarding the results of comparable studies, Boughaci and Alkhawaldeh (2018) used two databases present in this work, the AC and GC bases, and they reach an average accuracy of 86,5% for the Australian dataset, and 77,46% for the German database. Wang et al. (2012) also used two databases present in this work, the Australian and Japanese datasets, and with the best combination of feature selection and classifier, they got a Precision of 0,893 and Recall of 0,889 for the AC base, and 0,909 Precision and 0,905 Recall for the JC base, indicating that our findings are consistent with related studies in the literature.

The computational time is highly dependent on the machine learning technique, database, and the number of iterations. The procedure described in Section 3.2 seeks to minimize the time cost of the classifier. Even though, it still influences the total time of the process, as can be seen in Table 9, which shows the average computational time for each combination of classifier and database, with the same number of iterations in every case.

**Table 9**
Average computational time (seconds) of the VNS for each combination of database and classifier.

| Classifier | AC | GC | TC | JC | AER |
|---|---|---|---|---|---|
| Logistic Reg | 0.138 | 1.33 | 1.457 | 0.125 | 0.126 |
| k-NN | 0.513 | 0.363 | 83.874 | 0.49 | 0.174 |
| Bagging | 6.870 | 3.397 | 312.995 | 1.387 | 2.348 |
| Boosting | 1.032 | 2.291 | 14.056 | 0.562 | 0.563 |
| Random Forest | 4.161 | 6.705 | 185.344 | 4.283 | 3.893 |
| SVM | 0.575 | 1.733 | 874.799 | 0.571 | 3.94 |
| Neural Network | 11.337 | 14.306 | 94.581 | 12.508 | 33.912 |

**Table 10**
Average number of variables obtained for each classifier/database combination with VNS.

| Classifier | AC | GC | TC | JC | AER |
|---|---|---|---|---|---|
| Logistic Reg | 71.43% | 75% | 56.52% | 80% | 72.73% |
| k-NN | 64.29% | 54.17% | 30.43% | 60% | 18.18% |
| Bagging | 71.43% | 70.83% | 86.96% | 73.33% | 63.64% |
| Boosting | 64.29% | 62.5% | 43.48% | 62.5% | 72.73% |
| Random Forest | 71.43% | 66.67% | 82.61% | 62.5% | 54.54% |
| SVM | 100% | 83.33% | 95.65% | 100% | 72.73% |
| Neural Network | 64.29% | 66.67% | 65.22% | 62.5% | 54.54% |

**Table 11**
Number of variables for each variance level considered.

| Cumulative variance | AC | GC | TC | JC | AER |
|---|---|---|---|---|---|
| 100% | 14 | 24 | 23 | 15 | 11 |
| 90% | 50% | 62.5% | 26.09% | 60% | 54.54% |
| 80% | 35.71% | 45.83% | 17.39% | 46.67% | 36.36% |
| 70% | 28.57% | 37.5% | 13.04% | 33.33% | 27.27% |

### *4.2. PCA and VNS results*

Aiming to compare the performance of the VNS with another feature selection method, a PCA was implemented. The PCA is a well-known statistical multivariate analysis method and can be used for variable selection problems (Matharaarachchi, Domaratzki, & Muthukumarana, 2021; Song, Guo, & Mei, 2010). According to Jolliffe (1986), the main idea of PCA is to reduce the dimensionality of databases containing correlated variables while retaining the maximum variability of the original data. This reduction is achieved by transforming the original data into a new set of orthogonal variables, the so-called Principal Components (PCs). The PCs are ordered so that the first will retain the largest portion of the variability of the original variables, while the second component will preserve the second largest portion of the original variability, and so on. Uğuz (2011) states that a simple and eligible way to settle the number of components is the cumulative percentage of variance criteria, in which the minimum percentage of the original variance to be preserved is set, then the first $n$ components that sum up the established percentage are selected. The percentage value to be chosen is strongly dependent on the database used, but the author notes that it is common to establish a variance of 70% to 90% as a minimum limit. In the present work, three variance levels were tested for each database, namely 70%, 80% and 90%.

Table 10 shows the average percentage of variables selected utilizing the VNS as a feature selector. For comparison purposes, Table 11 exhibits the relative number of components found for each database and variance level of the PCA. It can be seen that the PCA tends to reduce more the dimensional space than the VNS. Regarding the methods associated with the VNS, it can be inferred that the SVM tends to retain more variables during selection since that in the five analyzed bases, this was the method that concentrated more variables after the application of VNS. The opposite can also be inferred about k-NN since it was the method with the most variables discarded.

Besides PCA, another popular feature selection method was considered in this work, the Linear Discriminant Analysis (LDA). The approach followed by LDA is very much analogous to that of PCA. Apart from maximizing the variance of data, LDA also maximizes separation of multiple classes. The goal of LDA is to project a dimension space onto a lesser subspace without disturbing the class information (Reddy et al., 2020). Both PCA and LDA were tested with all seven classifiers and five databases, and from this comparison, PCA obtained better results in most of the tests, so it was kept as the referential to VNS. The results obtained using the LDA in the preliminary phase are presented in Appendix. The average results obtained with PCA and VNS for the AC, GC, TC, JC and AER databases are present on Tables 12 to 16, respectively.

When coupled with VNS, the classifiers consistently produced better results both in terms of accuracy and AUC, even if sometimes by a small margin of difference. This is exemplified by analyzing Table 14 for the Taiwan database. In this case, it is possible to see that the average difference in accuracy between the classifiers associated with VNS and PCA was between 1% and 2% (except for SVM, where this difference was close to zero), yet, a significant $p$-value was found. This situation can be explained by the fact that, even though those differences were small on average, they were constant throughout the twenty observations, resulting in a statistically significant difference. More distinct results between PCA and VNS can be found in Table 13, for the German database. In this case, the average difference regarding the accuracy was about 4% in favor of the VNS, for every classifier, including the SVM.

The ensemble methods (Bagging, Boosting and Random Forest) are the most sensitive about the use of PCA or VNS since they were the only ones to present statistically significant differences in accuracy across all databases. Furthermore, in three of the datasets, an ensemble method associated with VNS generated the highest values in accuracy and AUC, as can be seen in Tables 12, 15 and 16, which indicates that those are good alternatives to be used in conjunction with the proposed variable selector.

The fact that the PCA had a poor performance overall is not surprising. As PCA works as a filter method, it means that the reduction is performed independently from the classifier (contrary to a wrapper method, like the VNS), thus, the effect of this reduction is not necessarily positive in terms of accuracy. It is possible to observe that in several situations the PCA performed even worse than using all features. It probably means that some of the variation lost in the dimensionality reduction process was effectively benefiting the model's performance. Those results corroborate the fact that feature selection is not a trivial task, and simply keeping most of the information might not improve a classifier's performance.

### *4.3. Features analysis*

Analyzing the solutions obtained over the five databases, it is clear that they are highly influential in generating the results. In this sense, contrary to what happened in the other datasets, the tests applied to the AER base, which can be seen in Table 16, produced distinct results concerning the use of PCA or VNS, since all classifiers, except for Logistic Regression, had an average difference in terms of accuracy higher than 10% when associated with VNS, in addition to having a higher AUC. In this same database, the Bagging classifier combined with VNS registered the highest accuracy in this study, reaching 98.8% correct predictions on average. Still, concerning the AER base, it is interesting to observe that k-NN obtained a significant performance gain with only 18% of variables on average.

Since three of the datasets (GC, TC and AER) have some features in common, it is possible to check if there is any correspondence between them. Even though, a couple of considerations must be taken into account: first, the classifier act in conjunction with the VNS in the process of feature selection, which means that different classifiers can (and probably will) led to different sets of optimal features. Second, because the VNS has some random steps in its iterative process, two

**Table 12**
Australian Dataset: PCA and VNS.

| Classifier | PCA (0.8) | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.8575 | 0.92 | 0.81 | 0.921 | 0.8746 | 0.92 | 0.84 | 0.923 | 0.022 |
| k-NN | 0.8442 | 0.90 | 0.82 | 0.907 | 0.8725 | 0.92 | 0.84 | 0.917 | 0.001 |
| Bagging | 0.8451 | 0.87 | 0.84 | 0.897 | 0.8809 | 0.91 | 0.87 | 0.929 | <0.001 |
| Boosting | 0.8486 | 0.87 | 0.85 | 0.902 | 0.8752 | 0.93 | 0.84 | 0.925 | 0.002 |
| RF | 0.8436 | 0.87 | 0.84 | 0.899 | 0.8873 | 0.91 | 0.89 | 0.928 | <0.001 |
| SVM | 0.8497 | 0.93 | 0.79 | 0.920 | 0.8491 | 0.93 | 0.79 | 0.923 | 0.937 |
| NN | 0.8598 | 0.92 | 0.82 | 0.920 | 0.8801 | 0.91 | 0.87 | 0.924 | 0.009 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the PCA. Columns six to nine present the same indicators, but applying the VNS. The number between parentheses next to the PCA indicates the best variance level found. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

**Table 13**
German Dataset: PCA and VNS.

| Classifier | PCA (0.8) | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.7374 | 0.77 | 0.89 | 0.774 | 0.7768 | 0.80 | 0.91 | 0.793 | <0.001 |
| k-NN | 0.7262 | 0.75 | 0.91 | 0.757 | 0.7674 | 0.78 | 0.93 | 0.775 | <0.001 |
| Bagging | 0.7316 | 0.77 | 0.87 | 0.755 | 0.7726 | 0.80 | 0.89 | 0.774 | <0.001 |
| Boosting | 0.7194 | 0.77 | 0.85 | 0.724 | 0.7694 | 0.81 | 0.88 | 0.772 | <0.001 |
| RF | 0.7316 | 0.77 | 0.87 | 0.757 | 0.7786 | 0.80 | 0.91 | 0.777 | <0.001 |
| SVM | 0.7364 | 0.77 | 0.89 | 0.770 | 0.7736 | 0.80 | 0.89 | 0.793 | <0.001 |
| NN | 0.7382 | 0.77 | 0.89 | 0.774 | 0.7754 | 0.80 | 0.90 | 0.789 | <0.001 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the PCA. Columns six to nine present the same indicators, but applying the VNS. The number between parentheses next to the PCA indicates the best variance level found. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

**Table 14**
Taiwan Dataset: PCA and VNS.

| Classifier | PCA (0.9) | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.7957 | 0.80 | 0.98 | 0.701 | 0.8056 | 0.81 | 0.98 | 0.708 | <0.001 |
| k-NN | 0.7912 | 0.80 | 0.96 | 0.670 | 0.8111 | 0.83 | 0.95 | 0.708 | <0.001 |
| Bagging | 0.7964 | 0.82 | 0.94 | 0.703 | 0.8147 | 0.84 | 0.94 | 0.741 | <0.001 |
| Boosting | 0.7977 | 0.81 | 0.96 | 0.697 | 0.8203 | 0.84 | 0.96 | 0.751 | <0.001 |
| RF | 0.8000 | 0.82 | 0.94 | 0.709 | 0.8174 | 0.84 | 0.95 | 0.749 | <0.001 |
| SVM | 0.7776 | 0.78 | 1 | 0.667 | 0.7776 | 0.78 | 1 | 0.689 | 1 |
| NN | 0.8010 | 0.83 | 0.93 | 0.728 | 0.8217 | 0.84 | 0.95 | 0.748 | <0.001 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the PCA. Columns six to nine present the same indicators, but applying the VNS. The number between parentheses next to the PCA indicates the best variance level found. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

**Table 15**
Japan Dataset: PCA and VNS.

| Classifier | PCA (0.9) | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.8590 | 0.91 | 0.82 | 0.909 | 0.8659 | 0.92 | 0.82 | 0.915 | 0.445 |
| k-NN | 0.8546 | 0.90 | 0.83 | 0.912 | 0.8769 | 0.92 | 0.85 | 0.917 | 0.030 |
| Bagging | 0.8202 | 0.83 | 0.85 | 0.891 | 0.8856 | 0.90 | 0.89 | 0.926 | <0.001 |
| Boosting | 0.8419 | 0.86 | 0.86 | 0.898 | 0.8812 | 0.92 | 0.86 | 0.930 | <0.001 |
| RF | 0.8465 | 0.86 | 0.86 | 0.903 | 0.8957 | 0.92 | 0.89 | 0.931 | <0.001 |
| SVM | 0.8558 | 0.93 | 0.80 | 0.908 | 0.8543 | 0.93 | 0.80 | 0.913 | 0.870 |
| NN | 0.8584 | 0.91 | 0.83 | 0.908 | 0.8725 | 0.93 | 0.83 | 0.918 | 0.097 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the PCA. Columns six to nine present the same indicators, but applying the VNS. The number between parentheses next to the PCA indicates the best variance level found. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

runs of VNS, even with the same classifier, may led to different subset of variables.

Beginning with the AER dataset, the average monthly amount of credit card expenditure was the feature that appear most times across the tests with all classifiers. Besides that, number of bad reports, age, and other variables not present in any other database, such as income and ratio of expenditure over income were frequent features as well. The German Credit (GC) dataset usually held more features, as can be seen in Table 10. In this sense, credit history, amount of credit, and age were among the most common features selected, together with other exclusive variables, such as purpose of credit, time at the same job, status of checking account, duration of the credit in months, and if the

**Table 16**
AER Dataset: PCA and VNS.

| Classifier | PCA (0.9) | | | | VNS | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| LR | 0.8283 | 0.83 | 0.98 | 0.903 | 0.8580 | 0.85 | 1 | 0.951 | 0.001 |
| k-NN | 0.8306 | 0.88 | 0.91 | 0.868 | 0.9815 | 1 | 0.98 | 0.989 | <0.001 |
| Bagging | 0.8589 | 0.91 | 0.91 | 0.916 | 0.9880 | 1 | 0.98 | 0.994 | <0.001 |
| Boosting | 0.8367 | 0.89 | 0.90 | 0.893 | 0.9865 | 1 | 0.98 | 0.995 | <0.001 |
| RF | 0.8597 | 0.91 | 0.92 | 0.920 | 0.9871 | 1 | 0.98 | 0.995 | <0.001 |
| SVM | 0.8591 | 0.91 | 0.91 | 0.919 | 0.9750 | 1 | 0.98 | 0.995 | <0.001 |
| NN | 0.8553 | 0.91 | 0.90 | 0.920 | 0.9809 | 1 | 0.98 | 0.992 | <0.001 |

Left column indicates the base classifier. Columns two to five present, respectively, the Accuracy, Precision, Recall and Area Under ROC Curve obtained with the PCA. Columns six to nine present the same indicators, but applying the VNS. The number between parentheses next to the PCA indicates the best variance level found. Last column shows the significance probability (*p*-value) for the comparison of the Accuracy obtained with VNS and with the full set.

**Table 17**
Taiwan Dataset: full set and features selected for another dataset.

| Classifier | Full set | | | | Set from another base | | | | *p*-value |
|---|---|---|---|---|---|---|---|---|---|
| | Acc. | Prec. | Recall | AUC | Acc. | Prec. | Recall | AUC | |
| k-NN | 0.7915 | 0.80 | 0.97 | 0.678 | 0.8058 | 0.82 | 0.95 | 0.717 | <0.001 |
| RF | 0.8147 | 0.84 | 0.94 | 0.751 | 0.8131 | 0.84 | 0.94 | 0.744 | 0.226 |

applicant has registered telephone. For the Taiwanese database (TC), credit history and credit expenditure were among the most selected variables, followed by amount of credit given.

Since the Taiwan Credit (TC) dataset is much larger than both AER and GC, it can be interesting to analyze how the TC behaves using the features selected for the two smaller datasets. For that, among the features cited above as the most important for the AER and German dataset, we consider just those that have a similar or parallel in the Taiwan database, which are amount of credit given, credit history, age and credit card expenditure, which are very similar to the actual set of features found for the TC database. For that analysis, we used as base classifiers the k-NN, because it was the one that usually kept fewer variables, and the Random Forest because it was the classifier with better results overall. Table 17 below shows a comparison of the results of prediction.

As shown in Table 17, with Random Forest as the base classifier, the results for the TC dataset using all its features, or just the set selected before, were quite similar, as demonstrated by the *p*-value, which indicates that this difference is not statistically significant. More interesting though were the results using k-NN as the base classifier, in this case, the selected set of features provided a model with higher accuracy (statistically significant) than using all features. This indicates that similar databases can benefit from the same feature selection process.

### *4.4. Main findings*

Based on the presented results, it is not possible to point to a single feature selection and classifier combination that is clearly superior. However, one can see that some methods underperformed on certain datasets, especially the SVM method on the TC dataset, as shown in Tables 6 and 14. In this case, the accuracy presented both in conjunction with PCA, VNS or without any feature selection was 77.76%, which places it slightly below the other methods, but observing that the recall measure was 1, and taking into account that the Taiwan database has about 22% default, it is clear that the classifier simply considered all data as non-default, regardless of being associated with any feature selection method or not.

In general, the use of VNS variable selection indicates to be advantageous in terms of default prediction, even if this was sometimes small. Furthermore, there were cases in which a classifier originally performed poorly without the presence of a feature selection technique, but its results were significantly improved when VNS was used, indicating high robustness of the method. The use of VNS also proved to be advantageous compared to PCA, since the average accuracy and AUC were higher in almost all cases analyzed. Regarding the classifiers, Boosting, Random Forest and Neural Networks showed a consistent performance in all tests performed, besides that, the Random Forest and VNS combination led the average accuracy in three of the five databases, thus representing the main classifier and variable selector indication of this study.

## 5. Conclusions

The main objective of this study was to increase the accuracy of predictions in credit scoring models by reducing the dimensionality of the variable space. To this end, a variable selection technique was proposed based on a Variable Neighborhood Search (VNS) concept. This technique consists in generating several sets of variables, and the one that provides the best results is selected. In order to test the applicability of VNS, it was associated with seven machine learning methods used to predict the probability of default.

To evaluate the effectiveness of the proposed method, a comparison was made with both the results generated from the reduction of variables obtained through the PCA statistical method and those generated with the total set of variables, that is, without the use of any selection method. The main comparison measure used was accuracy, which measures the ratio of correctly predicted outcomes over the total. The auxiliary measures to accuracy Precision and Recall were also adopted, in addition to the traditional Area Under the ROC Curve (AUC) measure.

Based on the results obtained, VNS has shown a better performance than PCA, in general, since for most cases a significant difference in accuracy was obtained, in addition to a constantly better AUC. Regarding the results without any feature selection technique, the differences in performance were generally more subtle, but they still favored VNS. However, some particular cases showed that VNS obtained a significant improvement, indicating that it is a robust technique.

No clearly superior variable selection and classifier combination was pointed out, but some inferences emerged from this study. Among all machine learning methods used as classifiers, it was clear that the performance of SVM is the least affected by variable selection. Combined with the fact that it performed particularly poorly on one of the databases analyzed, SVM becomes one of the least interesting methods to use in conjunction with a feature selection technique.

The Boosting, Random Forest and Neural Networks methods showed satisfactory results, even without being associated with any classifier. The k-NN and Bagging methods showed poor results in some cases, but

**Table A.18**
Australian Dataset: PCA and LDA.

| Classifier | PCA | LDA |
|---|---|---|
| | Accuracy | Accuracy |
| LR | 0.8503 | 0.8486 |
| k-NN | 0.8422 | 0.8520 |
| Bagging | 0.8347 | 0.8092 |
| Boosting | 0.8486 | 0.8485 |
| RF | 0.8393 | 0.8087 |
| SVM | 0.8445 | 0.8486 |
| NN | 0.8538 | 0.8503 |
| Average | 0.8448 | 0.8380 |

**Table A.19**
German Dataset: PCA and LDA.

| Classifier | PCA | LDA |
|---|---|---|
| | Accuracy | Accuracy |
| LR | 0.7348 | 0.7696 |
| k-NN | 0.7256 | 0.7400 |
| Bagging | 0.7304 | 0.6908 |
| Boosting | 0.7236 | 0.7540 |
| RF | 0.7316 | 0.6908 |
| SVM | 0.7372 | 0.7692 |
| NN | 0.7616 | 0.7380 |
| Average | 0.7316 | 0.7394 |

when coupled with VNS, the five methods mentioned in this paragraph had a solid performance in all databases, indicating that, among the analyzed techniques, these methods in combination with VNS are the most appropriate to be applied in credit scoring problems.

Future studies could consider classifiers that were not used in this study, as well as other wrapper methods to be compared with VNS. The adoption of a hybrid approach for the variable selection stage could also be recommended, consisting of a pre-selection method associated with a heuristic.

## CRediT authorship contribution statement

**Victor Gomes Helder:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Tiago Pascoal Filomena:** Conceptualization, Formal analysis, Methodology, Project administration, Resources, Supervision, Validation, Writing – review & editing. **Luciano Ferreira:** Methodology, Software, Supervision, Validation, Writing – review & editing. **Guilherme Kirch:** Supervision, Validation, Writing – review & editing.

## Declaration of competing interest

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

## Appendix. PCA and LDA comparison

A comparison between PCA and LDA was conducted to establish the more suitable rival to VNS. Tables A.18 to A.22 show the average accuracy of PCA and LDA, over 10 runs of each combination of classifier and database.

The results show that there was not a clear winner from this comparison. Yet, PCA achieved a better accuracy in 17 of the 35 combinations (against 16 of the LDA, and two ties), outperformed the LDA in 3 of the 5 datasets, and had a better average accuracy overall, so it was selected as a reference for feature selection.

**Table A.20**
Taiwan Dataset: PCA and LDA.

| Classifier | PCA | LDA |
|---|---|---|
| | Accuracy | Accuracy |
| LR | 0.7966 | 0.8032 |
| k-NN | 0.7928 | 0.7662 |
| Bagging | 0.7970 | 0.7156 |
| Boosting | 0.7990 | 0.8099 |
| RF | 0.8013 | 0.7143 |
| SVM | 0.7792 | 0.7792 |
| NN | 0.8015 | 0.8125 |
| Average | 0.7953 | 0.7716 |

**Table A.21**
Japan Dataset: PCA and LDA.

| Classifier | PCA | LDA |
|---|---|---|
| | Accuracy | Accuracy |
| LR | 0.8659 | 0.8040 |
| k-NN | 0.8624 | 0.8104 |
| Bagging | 0.8191 | 0.7890 |
| Boosting | 0.8399 | 0.8098 |
| RF | 0.8434 | 0.7867 |
| SVM | 0.8561 | 0.8058 |
| NN | 0.8653 | 0.8046 |
| Average | 0.8503 | 0.8015 |

**Table A.22**
AER Dataset: PCA and LDA.

| Classifier | PCA | LDA |
|---|---|---|
| | Accuracy | Accuracy |
| LR | 0.8306 | 0.8879 |
| k-NN | 0.8318 | 0.8809 |
| Bagging | 0.8652 | 0.8652 |
| Boosting | 0.8488 | 0.8939 |
| RF | 0.8600 | 0.8642 |
| SVM | 0.8712 | 0.8839 |
| NN | 0.8664 | 0.8900 |
| Average | 0.8534 | 0.8809 |

## References

Ab Hamid, T. M. T., Sallehuddin, R., Yunos, Z. M., & Ali, A. (2021). Ensemble based filter feature selection with harmonize particle swarm optimization and support vector machine for optimal cancer classification. *Machine Learning with Applications*, Article 100054.

Ala'raj, M., & Abbod, M. (2015). A systematic credit scoring model based on heterogeneous classifier ensembles. In *2015 international symposium on innovations in intelligent systems and applications (INISTA)* (pp. 1–7). IEEE.

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

Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., & Vanthienen, J. (2003). Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society*, *54*(6), 627–635.

Bao, W., Lianju, N., & Yue, K. (2019). Integration of unsupervised and supervised machine learning algorithms for credit risk assessment. *Expert Systems with Applications*, *128*, 301–315.

Bellotti, T., & Crook, J. (2009). Support vector machines for credit scoring and discovery of significant features. *Expert Systems with Applications*, *36*(2), 3302–3308.

Beuren, G. M., & Anzanello, M. J. (2019). Variable selection using statistical non-parametric tests for classifying production batches into multiple classes. *Chemometrics and Intelligent Laboratory Systems*, *193*, Article 103830.

Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. *Decision Support Systems*, *50*(3), 602–613.

Boughaci, D., & Alkhawaldeh, A. A.-s. (2018). Three local search-based methods for feature selection in credit scoring. *Vietnam Journal of Computer Science*, *5*(2), 107–121.

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

Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications*, *39*(3), 3446–3453.

Chen, F.-L., & Li, F.-C. (2010). Combination of feature selection approaches with SVM in credit scoring. *Expert Systems with Applications*, *37*(7), 4902–4909.

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *20*(2), 215–232.

El-Sherbeny, N. A. (2010). Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. *Journal of King Saud University-Science*, *22*(3), 123–131.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. *Vol. 96*, In *Icml* (pp. 148–156). Citeseer.

García-Torres, M., Gómez-Vela, F., Melián-Batista, B., & Moreno-Vega, J. M. (2016). High-dimensional feature selection via feature grouping: A variable neighborhood search approach. *Information Sciences*, *326*, 102–118.

Géron, A. (2017). *Hands-on machine learning with scikit-learn and tensorflow: concepts, tools, and techniques to build intelligent systems*. ''O'Reilly Media, Inc.''.

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*(3), 523–541.

Hansen, P., Mladenović, N., Todosijević, R., & Hanafi, S. (2017). Variable neighborhood search: basics and variants. *EURO Journal on Computational Optimization*, *5*(3), 423–454.

Hu, L.-Y., Huang, M.-W., Ke, S.-W., & Tsai, C.-F. (2016). The distance function effect on k-nearest neighbor classification for medical datasets. *SpringerPlus*, *5*(1), 1304.

Huang, J. (2015). Feature selection in credit scoring-a quadratic programming approach solving with bisection method based on Tabu search. (Ph.D. thesis), Texas A&M International University.

Jolliffe, I. T. (1986). Principal components in regression analysis. In *Principal component analysis* (pp. 129–155). Springer.

Ju, X., Rosenberger, J. M., Chen, V. C., & Liu, F. (2022). Global optimization on non-convex two-way interaction truncated linear multivariate adaptive regression splines using mixed integer quadratic programming. *Information Sciences*, *597*, 38–52.

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*(1), 124–136.

Lorena, A. C., Jacintho, L. F., Siqueira, M. F., De Giovanni, R., Lohmann, L. G., De Carvalho, A. C., et al. (2011). Comparing machine learning classifiers in potential distribution modelling. *Expert Systems with Applications*, *38*(5), 5268–5275.

Marinaki, M., & Marinakis, Y. (2015). A hybridization of clonal selection algorithm with iterated local search and variable neighborhood search for the feature selection problem. *Memetic Computing*, *7*(3), 181–201.

Marqués, A., García, V., & Sánchez, J. S. (2012). Exploring the behaviour of base classifiers in credit scoring ensembles. *Expert Systems with Applications*, *39*(11), 10244–10250.

Masson, E., & Wang, Y.-J. (1990). Introduction to computation and learning in artificial neural networks. *European Journal of Operational Research*, *47*(1), 1–28.

Matharaarachchi, S., Domaratzki, M., & Muthukumarana, S. (2021). Assessing feature selection method performance with class imbalance data. *Machine Learning with Applications*, *6*, 1–18.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, *24*(11), 1097–1100.

Ong, C.-S., Huang, J.-J., & Tzeng, G.-H. (2005). Building credit scoring models using genetic programming. *Expert Systems with Applications*, *29*(1), 41–47.

Peng, Y., Albuquerque, P. H. M., Kimura, H., & Saavedra, C. A. P. B. (2021). Feature selection and deep neural networks for stock price direction forecasting using technical analysis indicators. *Machine Learning with Applications*, Article 100060.

Reddy, G. T., Reddy, M. P. K., Lakshmanna, K., Kaluri, R., Rajput, D. S., Srivastava, G., et al. (2020). Analysis of dimensionality reduction techniques on big data. *IEEE Access*, *8*, 54776–54788.

Severino, M. K., & Peng, Y. (2021). Machine learning algorithms for fraud prediction in property insurance: Empirical evidence using real-world microdata. *Machine Learning with Applications*, *5*, 1–14.

Song, F., Guo, Z., & Mei, D. (2010). Feature selection using principal component analysis. *Vol. 1*, In *2010 international conference on system science, engineering design and manufacturing informatization* (pp. 27–30). IEEE.

Talbi, E.-G. (2009). *Vol. 74*, *Metaheuristics: From design to implementation*. John Wiley & Sons.

Thomas, L. C. (2009). *Consumer credit models: Pricing, profit and portfolios: pricing, profit and portfolios*. OUP Oxford.

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

Uğuz, H. (2011). A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. *Knowledge-Based Systems*, *24*(7), 1024–1032.

Van Gestel, T., & Baesens, B. (2008). *Credit risk management: basic concepts: financial risk components, rating analysis, models, economic and regulatory capital*. OUP Oxford.

Wang, J., Hedar, A.-R., Wang, S., & Ma, J. (2012). Rough set and scatter search metaheuristic based feature selection for credit scoring. *Expert Systems with Applications*, *39*(6), 6123–6128.

West, D. (2000). Neural network credit scoring models. *Computers & Operations Research*, *27*(11–12), 1131–1152.

Yao, J., Zhang, J., & Wang, L. (2018). A financial statement fraud detection model based on hybrid data mining methods. In *2018 international conference on artificial intelligence and big data (ICAIBD)* (pp. 57–61). IEEE.

Zhang, W., He, H., & Zhang, S. (2019). A novel multi-stage hybrid model with enhanced multi-population niche genetic algorithm: An application in credit scoring. *Expert Systems with Applications*, *121*, 221–232.