Economic Analysis and Policy 74 (2022) 175–187

Contents lists available at ScienceDirect

Economic Analysis and Policy

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

Modelling economic policy issues

Machine learning for liquidity risk modelling: A supervisory perspective

Pedro Guerra [*], Mauro Castelli, Nadine Côrte-Real

NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, 1070-312, Lisbon, Portugal

ARTICLE INFO

Article history:
Received 10 August 2021
Received in revised form 14 January 2022
Accepted 4 February 2022
Available online 12 February 2022

JEL classification:
C61
681
033
E58

Keywords:
Banking supervision
Risk assessment
Machine learning
EWS
Liquidity
Scenario analysis
ECB risk assessment system

ABSTRACT

The purpose of an effective liquidity risk assessment policy is to ensure that any given credit institution can meet its cash flow obligations, even factoring in the uncertainty caused by external factors. As part of the Supervisory Review and Evaluation Process (SREP), the European Central Bank (ECB) has determined this assessment should take into consideration both the institution's ability to meet its short-term obligations and its long-term funding strategy. Due to the fast pace of financial markets and more demanding regulations, there is a structural need for a precise and widely accepted risk assessment methodology. Furthermore, the ability to foresee alternative scenarios by stressing the involved key risk indicators is of the utmost importance. This work investigates whether machine learning techniques can successfully model liquidity risk, thus providing insights for stress-testing scenarios. We have applied the Risk Assessment System (RAS) methodology to classify credit institutions from the Portuguese banking sector according to their liquidity risk, using real supervisory data (from 2014 until March 2021). We then studied the ability to model this risk classification, by comparing a series of well-established machine learning algorithms to a traditional statistical model for benchmarking. The results show that extreme gradient boosting (XGBoost) outperforms other methods for this classification problem. The resulting model can be set up for a production environment and provide scenarios for stress-testing, or as an early warning system (EWS), thus supporting the overall SREP exercise.

1. Introduction

Ever since the 1990s, the financial sector has stimulated the development of decision support systems (Zopounidis et al., 1997). Classic statistical methods, like linear or logistic regressions, have been a pillar of those systems and financial analytical models in general. More recently, machine learning (ML) has been gaining thrust as the preferable tool, mainly due to the vast amount of data collected and the increasing computational power available. ML has been proven to unveil previously undetected complex data patterns, which are almost impossible to model. Furthermore, a recent study from the Bank of England (Hertig, 2021) emphasises machine learning as a growing technology for supervisory processes, mainly for detecting illegal market practices. These findings support expanding the use of ML for other supervisory tasks, namely, risk assessment processes. The findings in Guerra and Castelli (2021) also settle the intersection of this two knowledge areas as the way forward.

[*] Corresponding author.
E-mail addresses: pedroarteagaguerra@gmail.com (P. Guerra), mcastelli@novaims.unl.pt (M. Castelli), nreal@novaims.unl.pt (N. Côrte-Real).

https://doi.org/10.1016/j.eap.2022.02.001

1.1. Risk assessment measures

A universally accepted risk assessment methodology has always been a hot debate topic for researchers in this area. The methodologies used are increasing in sophistication as sup-tech is incorporated into day-to-day tasks. Additionally, this happens due to the necessity to accurately measure the risks banks are incurring in, from several perspectives. On the other hand, the need to comply with new regulatory requirements also triggers new approaches to risk analysis.

Several have proposed methods for bankruptcy probability assessment (Hillegeist et al., 2004; Ribeiro et al., 2012; Climent et al., 2019; Leo et al., 2019; Wang et al., 2021). The method presented by Hillegeist et al. (2004), Black–Scholes–Merton option-pricing model, outperforms two other well-known and reliable measurements: Z-score (Altman, 1968) and O-score (Ohlson, 1980). The authors stress the need for a standardised methodology to support the comparability between institutions.

Most of the current literature models risk classification according to a binary classification, for instance, "failure" or "no failure" of a bank. This target variable is derived from a set of financial ratios, most often from public or proxy datasets.

For this study, we consider the classification method presented in a well-established and widely approved methodology for risk measurement – the Supervisory Review and Evaluation Process (SREP) (European Central Bank, 2022) – defined by the ECB in cooperation with the National Competent Authorities (NCAs). This is the process through which supervisors periodically assess and measure the risk for each bank from five perspectives: liquidity, credit, market, operational, and profitability. The authors support our risk classification on the automatic Risk Assessment System (RAS), which is then reclassified according to expert judgement.

This methodology uses real supervisory data collected through the European Banking Authority (EBA) directive for Implementing Technical Standards (European Banking Authority, 2013), within the scope of the Single Supervisory Mechanism (SSM) (European Commission, 2015). Data is used to classify each institution in terms of its risk level, according to the automatic risk assessment system from the SREP process. These observations range from 2014 until March 2021. The data used in this research is extensively validated, thus ensuring a positive correlation with liquidity risk assessment capabilities (Ng, 2011).

1.2. Machine learning for risk assessment

Risk assessment is a predominantly quantitative exercise, often adjusted through expert judgement. The use of machine learning methods from a central bank perspective is a recent topic of interest, not only from NCAs and other agencies' perspective, but also from the academic point of view.

Since the early 2000s, risk assessment has been identified as a top priority for the efficient use of financial resources (Galindo and Tamayo, 2000). Early in that decade, the same authors established that tree-based models are more adequate in prediction tasks when compared to artificial neural networks (ANN), using structured data. This result is reinforced by other publications, throughout the years. Kolari et al. (2019) specifically address stress testing, suggesting it is an assessment of a bank's ability to deal with the risk it is exposed to, rather than the bank's actual resilience.

Recent technological evolution has been supporting the development of more sophisticated models (Strydom and Buckley, 2019), like deep learning (DL) models, as well as new ensemble methods like extreme gradient boosting (XGBoost) (Abellán and Castellano, 2017), due to their capability to capture the complexity of this type of phenomenon. DL first reappeared in 2012 with ImageNet (Krizhevsky et al., 2012). However, DL was applied to financial risk assessment only in 2016. Dastile et al. (2020) confirm DL as a promising tool in risk assessment, in particular for credit risk. They hypothesise extrapolating this approach to other risk perspectives, although the lack of interpretability of DL is seen by these authors as the main barrier for adopting this approach.

At the same time, several studies showcase the level of precision with which deep learning models adapt to structured data. Petropoulos et al. (2018) expand on the use of advanced ML techniques from a supervisory perspective. These authors developed an Early Warning System (EWS) for credit risk prediction, using data from Greek banks' corporate loans (Bank of Greece; 2005–2015). Although XGBoost emerged as the best model, DNNs also presented promising results. Similarly to what Iturriaga and Sanz (2015) have demonstrated, modelling a timeline evolution is where neural networks excel (in this case, deep neural networks — DNN's).

As in bankruptcy prediction, using machine learning to model a risk assessment usually sums up to a classification task where the developed model assigns a binary result to a certain observation of context: "fail" or "no fail". This means that for a set of independent variables/indicators, that represent a bank's context in a certain period, the model will first learn, then predict, whether that bank will go bankrupt or not, with a particular degree of certainty.

On the business side, it is crucial to understand how banks, national competent authorities and other agencies are adapting to this evolution. In particular, we are interested in how central banks use innovative technologies to leverage their analytical capabilities, namely for risk assessment.

According to what Stock and Watson (2001) formulate that macroeconometricians at policy institutions do, NCAs are responsible for:

1. Summarising and analysing data;
2. Forecasting the key macroeconomic variables;

3. Conducting risk analysis and balance of uncertainties;
4. Performing structural/causal analysis, as well as scenario analysis;
5. Making decisions, communicating them and justifying these decisions vis-a-vis the public.

A study conducted by Broeders and Prenio (2018) showcases the experience of early users of innovative technology in supervision (sup-tech). This work presents a new definition of sup-tech and shows how it is used for data collection and analytics. Chakraborty and Joseph (2017) published a similar study where the authors compile, present and compare the approaches adopted by NCAs and other agencies. As noted before, the amount of available data emerges as an important vector for the development of decision support systems based on ML.

Massaro et al. (2020) present a production-ready solution using ML to support a NCA's everyday tasks. Although this work is not a risk assessment tool, it proves how these NCAs can leverage on sup-tech.

We found only one paper addressing risk assessment using ML, from a supervisory perspective (Filippopoulou et al., 2020). The EWS developed by these authors is of great relevance for central banks. It addresses risk assessment, but most importantly, it uses real data gathered in the aftermath of the 2008 economic collapse (European Central Bank Macroprudential Database). Pompella and Dicanio (2017) also propose an EWS to alert for banks' distress signs. The authors propose a credit risk model to help adjusting rating assignments by the responsible agencies. Along with Filippopoulou et al. (2020), these findings suggest EWS as reliable instruments supporting supervisory processes.

The paucity of studies such as the one just mentioned, is a gap we propose to address. To the best of our knowledge, there are no papers addressing liquidity risk assessment from a supervisory perspective. Additionally, this work uses real-world data collected at a central bank in the context of supervisory directives. The fact that this type of datasets are privileged and therefore confidential further justifies the nonexistence of similar studies.

The few studies addressing risk assessment with ML techniques use public or proxy datasets. These early works set the tone for the particular use case of central banks. In the supervisory context, data is confidential and the processes are supported by European-wide legislation, thus making these papers more likely to stem from joint works with NCAs. Additionally, we do not use a sample dataset but rather the entire population: the Portuguese banking sector. Also supporting the novelty of this work is the risk assessment methodology used: the quantitative pillar of SREP, the Risk Assessment System (RAS). We model the risk assessment task through a classification problem. As opposed to the papers cited above, we propose expanding the usual binary classification into multiple classes, according to banks' risk level and as established in the RAS methodology:

1. low risk;
2. medium-low risk;
3. medium risk;
4. high risk.

This approach ensures that we can look through the same lenses at all banks in the Euro-area, making these assessments comparable, replicable and transparent.

In this work, we decide to consider solely liquidity risk due to its high importance to a bank's financial health (Vento and Ganga, 2009). A liquidity crisis can lead a bank to bankruptcy in less than a week (Shah et al., 2018). Therefore, it is of the utmost importance to deliver innovative tools that increase the current analytical capabilities of central banks. We aim to provide a solid base for a scenario analysis tool.

2. Methodology

The fundamental purpose of machine learning (ML) is extracting predictions from underlying data (or Big Data). Generally, Machine Learning algorithms are applied to data to get insights from it. In this case we are using Cross Sectional Data, that can be captured at any point in time. Using information from previously observed circumstances (cross sectional data), ML algorithms can predict values pertaining to events that have yet to occur.

Fig. 1 displays the steps performed in the experimental phase of this study. We started by retrieving the data from the *Banco de Portugal* production database for supervisory data. This dataset includes all the available features, as well as the pre-computed target — the RAS score for liquidity risk. Data transformation comprises data cleaning, implementing a strategy to deal with missing values, and the feature selection process. In the experiment phase, we compare three different approaches to evaluate the ML algorithms for this task: the classic train-test split, the more accurate cross-validation, and the TPOT AutoML framework (Olson et al., 2016). We then use the f1-score and the confusion matrices to compare the results and finally, select the best model. In future use, this model can be deployed as an Early Warning System making predictions for the liquidity risk level.

In this section we will describe the methods used in this research, from data gathering to model performance evaluation.

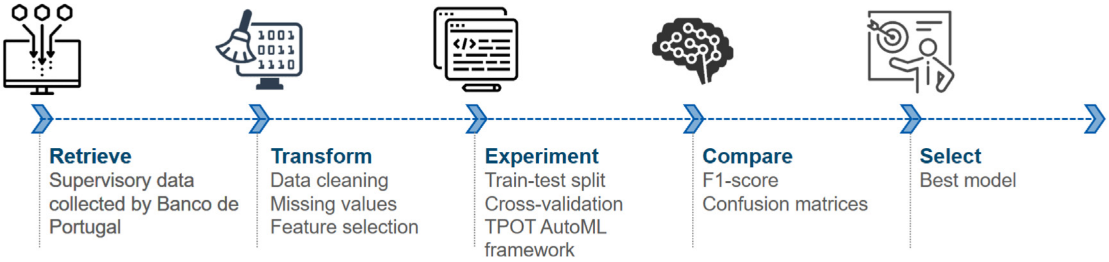


Fig. 1. Methodology process overview.

2.1. The data

This study relies on supervisory data collected by *Banco de Portugal* (Portuguese Central Bank — BdP) within the Capital Requirements Regulation (CRR) and Capital Requirements Directive IV (CRD IV) European Parliament (2013). The data ranges from March 2014 until March 2021. Depending on its nature, some data is gathered monthly while in other cases it is gathered quarterly (European Banking Authority, 2013). Due to confidentiality issues, the dataset used in this study cannot be made available for public consult.

Data is extracted via SQL query from BdP's production database into a *comma-separated-values* (csv) file to be imported using the Python programming language. An extraction routine was implemented to assure consistency and automation in data gathering. No filter is applied regarding reference date, institutions or level of consolidation. The extraction is structured in two steps:

1. First, the features are selected from the reported data. These belong to the 4 main reporting frameworks for banking supervision: Financial Reporting, Common Reporting, Asset Encumbrance and Funding Plans. This set encompasses all possible predictors.
2. The target variables are selected. These are computed through a corporate calculation process but all intermediate variables are discarded, in order to avoid any possible mathematical relation between features and target.

The data resides in a relational database where each row represents a reported value. This means that in the data source, several rows represent a single observation. During extraction, data is anonymised using MD5 algorithm within a hash function. This step assures the same identifier for every row in the same observation. The base dataset has the following topology:

1. **ID** - a hash code representing each observation's identifier;
2. **variable** - a code with business meaning that represents each reported value;
3. **val** - the actual numeric value of the variable.

2.2. Transformations

A python routine imports the CSV file, preparing the data for machine learning algorithms. The first step is pivoting the data set so that each of the resulting lines corresponds to an observation. Subsequently, we go through the data cleaning process that starts by discarding the target columns that fall out of the liquidity context. By this stage, each row corresponds to a single observation, and the last column represents our target variable (the RAS liquidity risk score). The other columns portray all the features available in our dataset.

The next steps delineate under which circumstances a row or column is discarded from our dataset:

1. Rows for which the target variable is null.
2. Rows that have a target variable 0. This value represents a non-applicable observation.
3. Rows where all features/columns are null.
4. Null columns: every column/feature has at least one reported value. After completing the previous steps, we must confirm that every feature still has values.

Finally, we deal with missing values for each feature. As pointed out by Madley-Dowd et al. (2019), multiple imputations can attain unbiased results up until 90% of missing data. Since in our dataset we have at most 20% of missing values, we do not discard observations based on this criteria. Instead, we use the median to fill out the missing values, which is the most adequate strategy for numeric datasets where the features present different distributions (Acuna and Rodriguez, 2004). If within the same feature/column we have similar mean and median it is indifferent which strategy to use. The use of the median gives a more appropriate idea of data distribution. After undergoing this process, the final sample included 5299 observations.

2.3. Feature selection

The selection of the most relevant predictors is an important step, not only for reducing computational time, but also to compare and contrast with the business perspective, the ECB Risk Assessment Methodology. After cleaning the data and dropping some non-representative features we are still dealing with the total universe of available data.

For the feature selection process we used Random Forest Classifier with an 85% threshold for the feature importance. This method was chosen due to its ability to rank the purity of each node (gini impurity): greatest impurity decrease occurs at the top of the tree (near root level) whereas smaller impurity decrease are is observed at the end (near leaf nodes). When this algorithm prunes below a particular node, it creates a subset of the most important features.

Through this strategy, we are able to technically assess the relevance of each feature regarding the variable we want to predict and select the ones that explain 85% (the importance threshold defined in the algorithm) of our target variable. The final dataset has a total of 3409 features selected from a universe of 82 559 predictors, and 5299 observations.

Afterwards, we compare the similarity of the obtained features with the ones the methodology highlights. This, per se, is a useful analysis since it gives hints to the analysts on which indicators to monitor more closely.

For the purpose of reducing computational time we have also considered, at first, the Principal Component Analysis (PCA). Although this method is associated with dimensionality reduction, its use compromises model explainability. At the same time, PCA loses track of the features that better represent our target variable, by projecting the feature space into a lower dimensional space.

At the end of this process we compute the correlation matrix of the dataset to assure there is not a high correlation between features and target. This would suggest that a certain feature represents the same phenomena as the target. The correlation indices range between a positive 26% and a negative 32%.

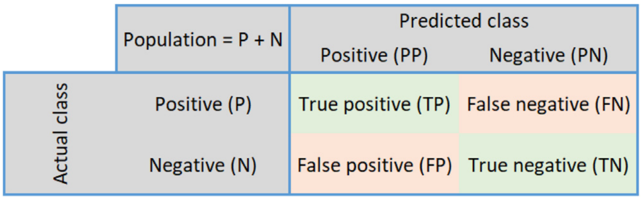
2.4. Experiments

The experiments carried out to assess and compare the performance of each model were organised in three separate phases, each of which is explained in the following subsections. First, we adopted the most straight-forward approach of splitting the data into two sets, the train and test sets. Afterwards, we use cross validation to measure the average performance of each model, considering every observation for either training or testing. Finally, we use an auto-ml library, TPOT (Olson et al., 2016), to have another evaluation perspective.

For each of the three approaches we calculate a measure of performance/scoring for both train and test sets. Furthermore, we compute the confusion matrix for a precise picture of each model's prediction.

We have selected a list of some of the most common machine learning algorithms used for classification problems. For the purpose of these experiments we have selected scikit-learn implementation of the following models:

1. Logistic Regression (LG) by Cox (1958), or Multinomial Logistic Regression, is an extension of the Bivariate Logistic Regression proposed by McCullagh and Nelder in 1989 (Glonek and McCullagh, 1995) for problems with more than two discrete outcomes. The original approach was designed for binary problems, and the target variable was modelled through a binomial probability distribution function. In its multiclass form, the probability is distributed by the number of classes of the problem at hand. In this paper, we used the scikit-learn implementation of the Logistic Regression for multi-classes (Pedregosa et al., 2011).
2. Support Vector Machine Classifier (SVC) – or Multi-class Support Vector Machine – is a generalisation proposed by Weston and Watkins (1998) of the binary classification Support Vector. Instead of computing the probability of an observation corresponding to a certain class (like the Logistic Regression), this method represents all datapoints in an n-dimensional space, and aims at creating a boundary, called a hyperplane, that separates the datapoints into classes. The algorithm tries to maximise the distance between the boundary and the nearest datapoints. Real-world data is seldom linearly separable, so it becomes computationally expensive to project all data into a higher dimensional space for calculating the distances to the optimal boundary. To overcome this computational hurdle, SVM uses the kernel trick, a method that uses a kernel function that takes two vectors/datapoints in the original space and computes their dot product in the feature space. Since the vectors are normalised the result is related to the Euclidean distance of both vectors — the distance we wanted to compute. In other words, this method shortcuts the computation of the distances from the datapoints to the possible hyperplanes, by performing them in the original n-dimensional space, thus reducing wall time (Adankon and Cheriet, 2009). We have used scikit-learn implementation of SVM based on libsvm library.
3. Naive Bayes Classifier (NBC) is a supervised learning method based on Bayes theorem, based upon the statistical independence of features. This simplified approach to learning shows it is up to par with more sophisticated classifiers, namely when dealing with high dimensionality and complex classification problems (Rish, 2001). Naive Bayes algorithms are thus very efficient to train and require little data to converge. This derives from the fact that they only require to compute the probability of each class, the conditional probabilities of each input value given a certain class, and the mean and standard deviation values of each attribute for each class. In this paper, we use the Gaussian Naive Bayes implementation from scikit-learn which presupposes a Gaussian distribution of the features.



	Population = P + N	Predicted class Positive (PP)	 Negative (PN)
Actual class	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

Fig. 2. Example of a confusion matrix for a binary classification problem.

4. Random Forest Classifier (RFC) is a learning method that combines tree predictors working together to minimise the error (Breiman, 2001). As thoroughly explained by Fawagreh et al. (2014), each decision tree in the forest is a base classifier using a sample of the instances in-bag, hence the bagging technique. The trees are combined through a voting system – one vote per tree – where the forest chooses the class with most votes. Another aspect that improved the randomness of the trees was the use of the Gini index — features with the highest index are used to split the inner node of the tree. This algorithm presents great results when dealing with data noise and avoiding overfit, and handles large datasets with high dimensionality. Here again, we are using its scikit-learn implementation.
5. Extreme Gradient Boosting (XGBC) Classifier proposed by Chen and Guestrin (2016) is a machine learning algorithm used for tree boosting that uses data compression and sharding (a data partition technique) to scale to large amounts of data. Due to its capability to avoid overfitting and its efficient use of large amounts of data, it has become one of the most popular ML methods in the last few years (Sahin, 2020). Having Friedman (2001) gradient boosting technique as its pillar, XGBoost uses a differentiable loss function and optimises it with gradient descent algorithm, in order to build an ensemble of classification trees. For this algorithm, we have used the authors' implementation package (Chen and Guestrin, 2016).

The TPOT auto-ml library automatically selects the best model and we use that result to compare with the others.

In order to have all features in a similar scale we have applied a scaling method when preprocessing the data. MinMaxScaler was the best choice since it preserves the shape of the original distribution. It does not significantly change the information embedded in the original data. Note that MinMaxScaler does not reduce the importance of outliers. The default range for the feature returned by MinMaxScaler is 0 to 1.

Here we present a list of the main characteristics of the experiments' environment:

1. Lenovo ThinkPad P50 with an Intel Xeon processor (2.8 GHz), 32 GB of RAM, 1 TB SSD;
2. Windows 10 64-bits;
3. Python 3.9.1 64-bits;
4. Pandas 1.2.0;
5. scikit-learn 0.24.0;
6. TPOT 0.11.7.

2.4.1. Performance measures

We used two different tools for comparison purposes: the confusion matrix and the f1-score. The confusion matrix is the most detailed view of how a particular machine learning model is performing in a classification problem (Tharwat, 2018). Through this tool, we are able to assess each of our model's predictions and compare them with the correct value.

Fig. 2 shows a generic matrix for a binary classification problem where we can observe each possible classification:

- True positive (TP) corresponds to the model correct hits.
- False negative (FN) represents every missed case, where the model underestimated.
- False positive (FP) represents false alarms, where the model overestimated.
- True negative (TN) corresponds to the correct rejections made by the model.

For our specific problem where we have four classes representing the risk levels for liquidity, we will have a *4X4* matrix for each model, which is simply a generalisation of the one just presented.

There are several metrics that one can extract from these statistics. However we will focus on the f1-score and two others derived from it, precision — or positive predictive values, that is the number of positive results that are true positives – and recall — also known as sensitivity or true positive rate, which measures the number of positive hits among all the positives:

- f1-score represents the harmonic mean of precision and recall. It is most suited for uneven class distributions, as is the case of our dataset . It is calculated as

$$f1 = 2 * \frac{precision * recall}{precision + recall} \tag{1}$$

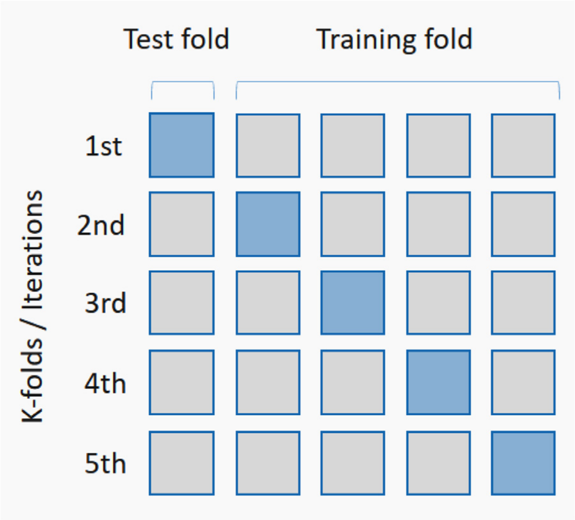


Fig. 3. Example of a 5-fold cross-validation process.

where

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

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

2.4.2. *Train-test split*

Our first approach to evaluating the performance of each model is through a train-test split of the available sample data. As a general principle, we used 80% of the data for training and 20% for testing.

The assessment is organised as follows:

1. Use the MinMaxScaler, as specified above;
2. Iterate through all machine learning models;
3. Fit the model to the data;
4. Assess train and test scores;
5. Compute the confusion matrix;
6. Store the results.

2.4.3. *Cross-validation*

When we are dealing with small to medium datasets a simple train-test split will most likely misrepresent our real-world problem by missing some classes. This is the main indication for using cross-validation, where every single observation is eligible for the train and test sets. The technique consists of splitting the dataset into a specific number of folds, or partitions, and iterating through the partitions.

In Fig. 3 we picture how a 5-fold cross-validation example would process. First, the dataset is split into 5 folds. Then, in each of the five iterations, one of the folds assumes the role of *test fold* and the other four as *training fold*. In each iteration, the machine learning algorithms are trained on the *training fold*, and their performance is assessed on the *test fold*. By the end of the five iterations, the average of the performance obtained on each iteration is the value considered for comparison. Cross-validation is the preferred method for assessing model performance because it gives models the opportunity to train on multiple train-test splits. This will better indicate how well a model will perform on unseen data. Conversely, a simple train-test split is dependent on just a single data split which can overestimate the overall performance.

In this experiment we used StratifiedKFold (a form of cross-validation) to preserve the percentage of samples among classes. The purpose of this specific form is for the test to be as close as possible to the whole dataset. The stratification ensures class frequencies of the partitions are equal to the complete dataset. This is particularly advantageous in an imbalanced dataset scenario, where this method ensures every class is represented.

The use of cross-validation can also raise some issues. Since we are assessing performance of a model on several splits, situations may arise where data leaks from one iteration to another. In other words, data leakage can happen when we are learning from both the testing and training set. If we do any pre-processing outside the cross-validation algorithm, we will bias our results and most likely overfit our model. To avoid this common problem we feed our cross-validation cycle the entire dataset and perform every transformation within each iteration. Although the authors concede that this repetition takes its toll on performance, the extra step assures no data is leaking from each of the splits or iterations.

F1-score is used as a performance measure since it keeps a balance between precision and recall. Furthermore, since we observe uneven class distribution in the dataset, F1-score is more appropriate than AUC (F1 gives a score for a specific threshold, whereas AUC averages over all possible thresholds).

Confusion matrix was selected as the best tool for describing performance on a classification model. This is an NxN matrix where *N* is the number of classes in our classification problem (as mentioned earlier, classes 1, 2, 3, and 4 representing the risk tiers for any given financial institution).

Similarly to the train-test split approach, we cycled through each model as follows:

1. Define a pipeline to streamline scaling, using the MinMaxScaler, and training;
2. Define a 10-fold cross-validation process using StratifiedKFold;
3. Use cross-val-predict to obtain predictions for each element (we want to compare and contrast predictions obtained from different models);
4. Compute the F1-score as well as the confusion matrix;
5. Store the results.

2.4.4. TPOT - An AutoML approach

As noted by Zöller and Huber (2019), AutoML has been around since the early 90's with the automated selection of algorithms via grid search - an exhaustive search method that automates the process of trying all possible combinations of a given set of hyper-parameters. However, only in 2018 did the first commercial full-pipeline solutions come to light. In order to keep up with these developments and to provide another perspective on how to approach this research, we used an autoML tool, TPOT, that optimises machine learning pipelines through genetic programming.

The use of this framework is straightforward and the parameters used were:

1. **generations** is an evolutionary computation concept that determines the number of iterations in the optimisation process. It gives the tool more time to find an optimal solution. This parameter was set to 5;
2. **population_size** is also an evolutionary computation concept that represents the number of possible solutions (as a subset of the total population, or number of solutions) to be considered in each generation. This parameter was set to 100;
3. **cv** specifies the number of cross-validation folds in a StratifiedKFold. This parameter was set to 10;
4. **verbosity** is a definition of how much information TPOT will provide during run-time. This parameter was set to 5;
5. **random_state** is a random number generator to assure TPOT will provide the same results given the same inputs. This parameter was set to 42.

This framework takes the full dataset and saves a portion (in our case, predefined as 20%) of randomly selected observations for validating the best model. Both the model's score and its confusion matrix are assessed based on the unseen data.

3. Results and discussion

In this section we present how the selected models compare when solving the liquidity risk problem, organised by approach: train-test split, cross-validation and autoML tool (TPOT).

Supervisors were closely engaged in this process, namely validating the underlying methodology and contributing to the feature selection process. One of the innovative aspects of this work is the risk assessment methodology used to classify each bank – the quantitative pillar of ECB's Risk Assessment Methodology – that assigns a score of one to four to each bank according to its liquidity risk level: low, medium-low, medium, or high risk.

RAS is used for all banks in the euro-area, and it is developed and maintained through a joint task-force from the ECB and the National Central Banks (NCBs). To the best of our knowledge, there are no studies using this methodology, nor considering this multi-class risk score.

The processing and evaluation times for each of the approaches were:

1. Train-test split: 1 min and 18 s;
2. Cross-validation: 8 min and 50 s;
3. TPOT framework: 21 h, 34 min and 51 s.

The execution times were measured using *%%time* python statement within jupyter notebook. This statement returns the wall time for the cell under evaluation. In this case, each approach is in a cell of its own.

Fig. 4 pictures the train-test split approach. This graph compares each of the selected models, contrasting their train and test scores as well as the models amongst themselves.

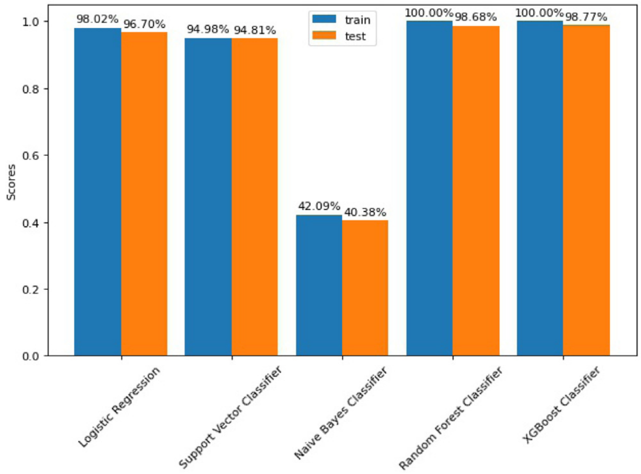
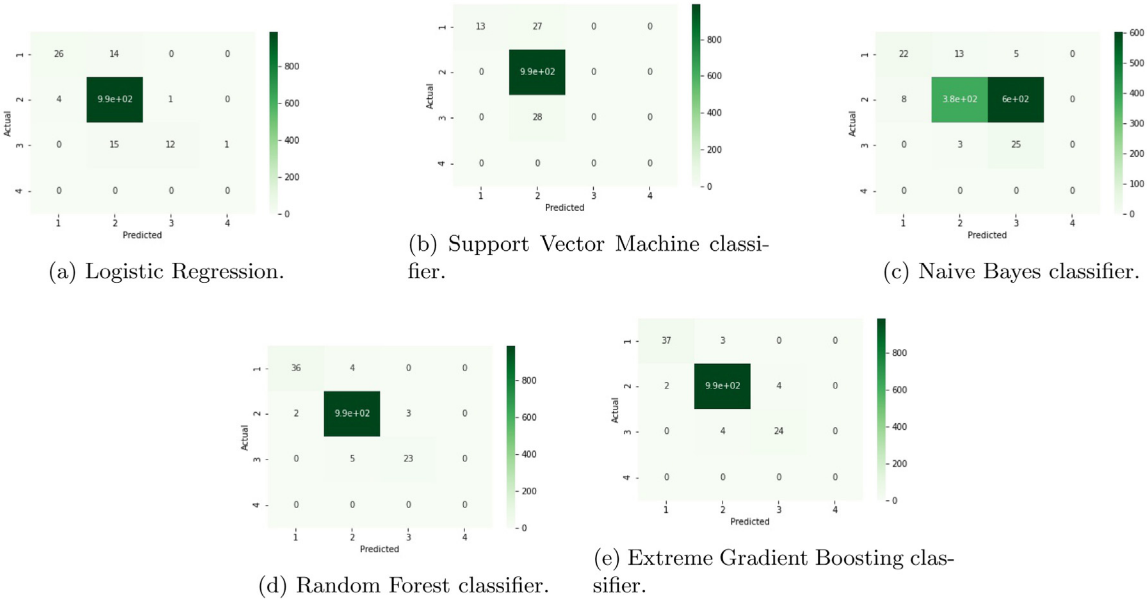


Fig. 4. Precision scores of each model, using train-test split approach.



(a) Logistic Regression.

(b) Support Vector Machine classifier.

(c) Naive Bayes classifier.

(d) Random Forest classifier.

(e) Extreme Gradient Boosting classifier.

Fig. 5. Confusion matrices generated when evaluating the above mentioned models, using train-test split approach.

The overall picture stresses that Naive Bayes classifier is inadequate for this problem, at least without further tuning. In the authors' opinion, the hypothetical gains do not compensate for the processing time required for the tuning process.

Fig. 5 gives a more exact notion of each model's precision through its confusion matrix. The Logistic Regression 5(a) presents a solid score. Nonetheless, its confusion matrix clearly demonstrates a difficulty detecting a risk score of 3 or 4. The Support Vector classifier 5(b) fares even worse, not detecting any scores of classes 3 or 4, and predicting very few class 1 scores.

Random Forest 5(d) and XGBoost 5(e) both show consistently good performance. Unsurprisingly, however, they both make similar mistakes, confusing predominantly the same risk classes. These results imply the need for a more robust approach.

Fig. 6 shows the comparison of the same models using the f1-scores to evaluate the cross-validation process. In relative terms, the picture shows the same distribution as the train-test split approach, although with cross-validation we have used the whole dataset. The reasoning behind this is that cross-validation already considers several train and tests splits (k splits for k-fold cross-validation), ensuring that the final score is not biased to a particular random split, and we are not excluding observations that can compromise the final score.

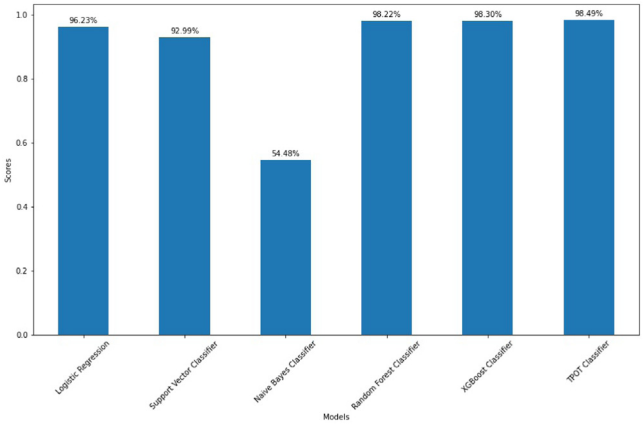
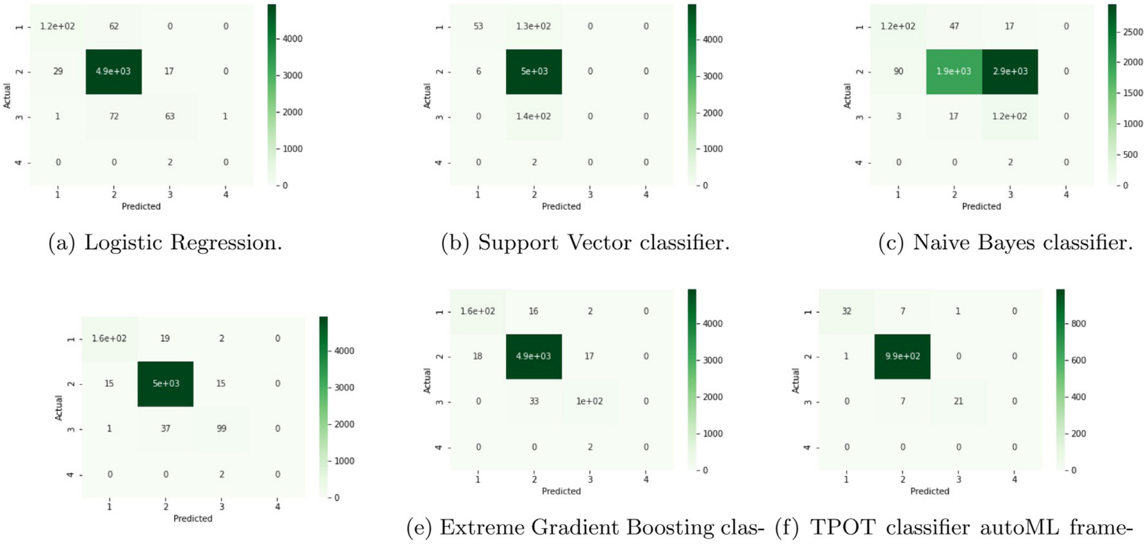


Fig. 6. Precision f1-scores of each model, using cross-validation approach.



(a) Logistic Regression.

(b) Support Vector classifier.

(c) Naive Bayes classifier.

(d) Random Forest classifier.

(e) Extreme Gradient Boosting classifier.

(f) TPOT classifier autoML framework.

Fig. 7. Confusion matrices generated when evaluating the above mentioned models, using cross-validation approach.

On the same premise as before, we use the confusion matrix to better assess each model's precision. Fig. 7 provides a comprehensive view of the classifications accomplished by each model. Although the Logistic Regression 7(a), used as a benchmark, shows a 96% f1-score with cross-validation, when we dive into each individual classification we realise that this is not the case. Class 2 shows 99% of correct classifications. However, classes 1 and 3 show 39.9% and 48.4% of misclassifications, respectively.

Support Vector Classifier 7(b) and Naive Bayes 7(c) models both show several misclassifications. The former shows good results detecting class 2, but not the other classes. The latter, although it got the worse f1-score, shows an average performance regarding classes 1 and 3. Class 2 presents 64% of misclassifications.

Random Forest classifier 7(d) and XGBoost 7(e) present very similar results. The misclassifications occur in the same classes and differ only by a couple of observations. This can be explained by the fact that both models are based on decision trees.

TPOT classifier 7(f) takes a step further by significantly increasing precision in classes 1, 2, and 3. Although this precision gain comes at the cost of 21 h of training, this approach proves itself worthwhile due the financial impacts that liquidity risk assessment might have.

Contrarily to cross-validation, where we used the full dataset to train and validate the best model (see Section 2.4.3), with TPOT, we set aside part of the dataset to validate the scores of the model as well as its confusion matrix. This explains the smaller number of observations in its confusion matrix when compared to others in Fig. 7.

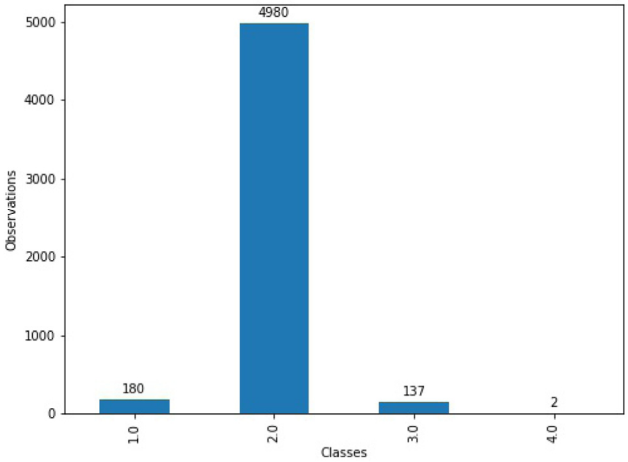


Fig. 8. Number of observations per target class in the dataset.

One aspect that is common to all models, regardless of the approach, is the fact that none correctly classifies class 4. This can be easily explained by Fig. 8 that shows how imbalanced our dataset is — by providing only two class 4 observations, we make it almost impossible for the models to establish a pattern. We considered using oversampling/undersampling techniques, however, we decided not to due to the nature of the problem and the fact that this represents the frequency of occurrence of each class in the real world.

As an overall consideration, we can add that the machine learning techniques clearly outperform the traditional Logistic Regression approach. Although we can be misled by a high score, an in-depth analysis of the confusion matrices clearly expose the classification advantages of ML techniques when compared to classic statistical methods.

These findings can bring a great advantage for regulators when considering risk assessment tools. The best performing model can be set up as a decision support system, either as stand-alone stress-testing tool, or as part of an EWS. Furthermore, this research will open a new path to address and support the overall risk assessment exercise, as part of the SREP process. If all other risk perspectives in this methodology present similar results, this work will foster a new approach to the SREP decisions, by combining all its components and establishing a baseline for each institutions' capital requirements.

4. Conclusion

A proper risk assessment methodology requires a thorough evaluation of a credit institution's practices as well as an integrated analysis of the uncertainty factors involved. In this study, we focused on liquidity risk assessment through the lenses of the SREP methodology established by the ECB. This step is the linchpin of this work since it establishes a widely accepted methodology for risk assessment.

4.1. Practical and theoretical implications

This paper intends to contribute to the growing body of knowledge regarding the use of ML techniques on sup-tech solutions. It specifically proposes a comprehensive and innovative approach, that considers all the knowledge expressed in historical data to support and envisage a critical European supervisory business process (ECB RAS).

The novelty of this work comprises two main vectors:

1. The use of a European-wide risk assessment methodology, the quantitative pillar of ECB's Risk Assessment System, guaranteeing a transparent approach to risk, as well as comparable results across the Euro-area. We classify the observed banks according to their risk level.
2. A dataset that represents the most up-to-date reality in the Portuguese supervisory context: we use real-world data retrieved within the scope of the SSM context from 2014 to March 2021.

Based on these two pillars, we compared several well-established machine learning algorithms to a traditional statistical method to evaluate which would best model this decision process.

Results showed that not only we can model this classification problem, but also the ML techniques used in this work clearly outperform the classic approach. Moreover, XGBoost stands out as the best quick approach to solve this classification problem. Conversely, the autoML framework TPOT takes 21 h to evaluate but with very few misclassifications.

Given that liquidity risk is an extremely sensitive aspect of a credit institution's health, we believe the precision gains compensate for the added processing time.

This paper intends to be the outset of new approach to risk assessment from several perspectives:

- For NCAs, an EWS based on this model can significantly increase the robustness of the RAS decision process.
- The findings of this paper can leverage the application of this methodology to other perspectives, like market, operational, rendibility and credit risks.
- This work also benefits banks and consultancy companies when implementing similar decision support systems. Although the underlying methodology is confidential, banks have the required data and the scores assigned by the NCA. This should enable them to better analyse their risk profile in terms of regulatory compliance.

4.2. Limitations and future work

Throughout this study we have identified several aspects that could be revisited in order to improve the results. The first issue is related to the dataset. The fact that the dataset is imbalanced hinders the detection of some classes. As shown in the previous section, class 4 contains just two observations which makes it difficult to model that particular decision process.

Another limitation is the fact that this dataset only reflects the Portuguese context. Further investigation would benefit from the use of all central banks' data, thus reflecting a broader picture of the supervisory landscape. Furthermore, the increase in the dataset size would strengthen the validation of the ML models.

The inclusion of quantitative data from non-supervisory frameworks is also extremely beneficial for improving the model's robustness. Data from financial markets, payment systems and macroeconomic indicators provide a context that supports the overall risk assessment.

We also find relevant to include expert judgement to reinforce to final risk assessment. To this end, qualitative data sources like internal notes and risk assessment reports, as well as risk scores reassigned by the supervisors should contribute to the model's learning phase.

Finally, we believe the other risk perspectives comprised in the SREP methodology should also be addressed using the same methodology. Ultimately, combining all risk perspectives could be a stepping stone for regulators as a support of the SREP exercise.

Disclaimer

The views, thoughts, and opinions expressed in the text belong solely to the authors.

References

Abellán, J., Castellano, J.G., 2017. A comparative study on base classifiers in ensemble methods for credit scoring. Expert Syst. Appl. 73, 1–10. http://dx.doi.org/10.1016/j.eswa.2016.12.020.

Acuna, E., Rodriguez, C., 2004. The treatment of missing values and its effect on classifier accuracy. In: Classification, Data Analysis, and Knowledge Organisation. pp. 639–647.

Adankon, M.M., Cheriet, M., 2009. Encyclopedia of Biometrics - Support Vector Machine. Springer US, pp. 1303–1308. http://dx.doi.org/10.1007/978-0-387-73003-5_299.

Altman, E., 1968. Financial ratios, discriminant analysis and the prediction of corpporate bankruptcy. J. Finance XXIII, 589–609.

Breiman, L., 2001. Random forests. Mach. Learn. 45, 5–32.

Broeders, D., Prenio, J., 2018. FSI insights innovative technology in financial supervision. FSI Insights Policy Implement. July 2018, 29, URL: https://www.bis.org/fsi/publ/insights9.pdf.

Chakraborty, C., Joseph, A., 2017. Machine learning at central banks. SSRN Electron. J. http://dx.doi.org/10.2139/ssrn.3031796.

Chen, T., Guestrin, C., 2016. XGBoost: A scalable tree boosting system. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. International Conference on Knowledge Discovery and Data Mining, pp. 785–794. http://dx.doi.org/10.1145/2939672.2939785.

Climent, F., Momparler, A., Carmona, P., 2019. Anticipating bank distress in the Eurozone: An Extreme Gradient Boosting approach. J. Bus. Res. 101, 885–896. http://dx.doi.org/10.1016/j.jbusres.2018.11.015.

Cox, D.R., 1958. The regression analysis of binary sequences. J. R. Stat. Soc. Ser. B Methodol. 20, http://dx.doi.org/10.1111/j.2517-6161.1958.tb00292.x.

Dastile, X., Celik, T., Potsane, M., 2020. Statistical and machine learning models in credit scoring: A systematic literature survey. Appl. Soft Comput. 91, 106263. http://dx.doi.org/10.1016/j.asoc.2020.106263.

European Banking Authority, 2013. EBA implementing technical standards (ITS). p. 24, URL: http://www.eba.europa.eu/documents/10180/532570/EBA-ITS-2013-12+(Final+draft+ITS+on+Hypothetical+Capital+of+a+CCP).pdf,

European Central Bank, 2022. Supervisory review and evaluation process. URL: https://www.bankingsupervision.europa.eu/about/ssmexplained/html/srep.en.html.

European Commission, 2015. Single supervisory mechanism. URL: https://ec.europa.eu/info/business-economy-euro/banking-and-finance/banking-union/single-supervisory-mechanism_en.

European Parliament, 2013. Directive 2013/36/EU. Off. J. Eur. Union 338–436, URL: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2013:176:0338:0436:En:PDF.

Fawagreh, K., Gaber, M.M., Elyan, E., 2014. Random forests: From early developments to recent advancements. Syst. Sci. Control Eng. 2, 602–609. http://dx.doi.org/10.1080/21642583.2014.956265.

Filippopoulou, C., Galariotis, E., Spyrou, S., 2020. An early warning system for predicting systemic banking crises in the Eurozone: A logit regression approach. J. Econ. Behav. Organ. 172, 344–363. http://dx.doi.org/10.1016/j.jebo.2019.12.023.

Friedman, J.H., 2001. Greedy function approximation: A gradient boosting machine. Ann. Statist. 29, 1189–1232.

Galindo, J., Tamayo, P., 2000. Credit risk assessment using statistical and machine learning: Basic methodology and risk modeling applications. Comput. Econ. 15, 107–143. http://dx.doi.org/10.1023/a:1008699112516.

Glonek, G.F., McCullagh, P., 1995. Multivariate logistic models. J. R. Statist. Soc. 57, 533–546.

Guerra, P., Castelli, M., 2021. Machine learning applied to banking supervision a literature review. Risks 9, 136. http://dx.doi.org/10.3390/risks9070136.

Hertig, G., 2021. Using artificial intelligence for financial supervision purposes. pp. 1–29, Bank of England.

Hillegeist, S.A., Keating, E.K., Cram, D.P., Lundstedt, K.G., 2004. Assessing the probability of bankruptcy. Rev. Account. Stud. 9, 5–34. http://dx.doi.org/10.1023/B:RAST.0000013627.90884.b7.

Iturriaga, F.J.L., Sanz, I.P., 2015. Bankruptcy visualization and prediction using neural networks: A study of U.S. commercial banks. Expert Syst. Appl. 42, 2857–2869. http://dx.doi.org/10.1016/j.eswa.2014.11.025.

Kolari, J.W., López-Iturriaga, F.J., Sanz, I.P., 2019. Predicting European bank stress tests: Survival of the fittest. Glob. Finance J. 39, 44–57. http://dx.doi.org/10.1016/j.gfj.2018.01.015.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. URL: http://code.google.com/p/cuda-convnet/.

Leo, M., Sharma, S., Maddulety, K., 2019. Machine learning in banking risk management: A literature review. Risks 7, http://dx.doi.org/10.3390/risks7010029.

Madley-Dowd, P., Hughes, R., Tilling, K., Heron, J., 2019. The proportion of missing data should not be used to guide decisions on multiple imputation. J. Clin. Epidemiol. 110, 63–73. http://dx.doi.org/10.1016/j.jclinepi.2019.02.016, URL: https://www.sciencedirect.com/science/article/pii/S0895435618308710.

Massaro, P., Vannini, I., Giudice, O., 2020. Institutional sector cassifier, a machine learning approach. SSRN Electron. J. 548, http://dx.doi.org/10.2139/ssrn.3612710.

Ng, J., 2011. The effect of information quality on liquidity risk. J. Account. Econ. 52, 126–143. http://dx.doi.org/10.1016/j.jacceco.2011.03.004.

Ohlson, J.A., 1980. Financial ratios and the probabilistic prediction of bankruptcy. J. Account. Res. 18, 109. http://dx.doi.org/10.2307/2490395.

Olson, R.S., Bartley, N., Urbanowicz, R.J., Moore, J.H., 2016. Evaluation of a tree-based pipeline optimization tool for automating data science. In: GECCO 2016 - Proceedings of the 2016 Genetic and Evolutionary Computation Conference. pp. 485–492. http://dx.doi.org/10.1145/2908812.2908918.

Pedregosa, F., Michel, V., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Vanderplas, J., Cournapeau, D., Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Dubourg, V., Passos, A., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830, URL: http://scikit-learn.sourceforge.net.

Petropoulos, A., Siakoulis, V., Stavroulakis, E., Klamargias, A., 2018. A robust machine learning approach for credit risk analysis of large loan level datasets using deep learning and extreme gradient boosting. Use Big Data Anal. Artif. Intell. Cent. Bank. 50, 30–31, URL: https://www.bis.org/ifc/publ/ifcb49_49.pdf.

Pompella, M., Dicanio, A., 2017. Ratings based Inference and Credit Risk: Detecting likely-to-fail Banks with the PC-Mahalanobis Method. Econ. Model. 67, 34–44. http://dx.doi.org/10.1016/j.econmod.2016.08.023.

Ribeiro, B., Silva, C., Chen, N., Vieira, A., Neves, J.C.D., 2012. Enhanced default risk models with svm+. Expert Syst. Appl. 39, 10140–10152. http://dx.doi.org/10.1016/j.eswa.2012.02.142.

Rish, I., 2001. An empirical study of the naive Bayes classifier.

Sahin, E.K., 2020. Assessing the predictive capability of ensemble tree methods for landslide susceptibility mapping using XGBoost, gradient boosting machine, and random forest. SN Appl. Sci. 2, http://dx.doi.org/10.1007/s42452-020-3060-1.

Shah, S.Q.A., Khan, I., Shah, S.S.A., Tahir, M., 2018. Factors affecting liquidity of banks: Empirical evidence from the banking sector of Pakistan. Colombo Bus. J. 9, 01. http://dx.doi.org/10.4038/cbj.v9i1.20.

Stock, J., Watson, M., 2001. Vector autoregressions. J. Econ. Perspect. 15, 101–115. http://dx.doi.org/10.1002/9780470996249.ch33.

Strydom, M., Buckley, S., 2019. AI and Big Data's Potential for Disruptive Innovation, first ed. IGI Global, pp. 1–405. http://dx.doi.org/10.4018/978-1-5225-9687-5.

Tharwat, A., 2018. Classification assessment methods. Appl. Comput. Inform. 17, 168–192. http://dx.doi.org/10.1016/j.aci.2018.08.003.

Vento, G.A., Ganga, P.L., 2009. Bank liquidity risk management and supervision : Which lessons from recent market turmoil ? J. Money Invest. Bank. 10, 79–126.

Wang, T., Zhao, S., Zhu, G., Zheng, H., 2021. A machine learning-based early warning system for systemic banking crises. Appl. Econ. 00, 1–19. http://dx.doi.org/10.1080/00036846.2020.1870657.

Weston, J., Watkins, C., 1998. Multi-class support vector machines.

Zöller, M.A., Huber, M.F., 2019. Benchmark and survey of automated machine learning frameworks. 70, pp. 411–474. http://dx.doi.org/10.1613/jair.1.11854, ArXiv.

Zopounidis, C., Doumpos, M., Matsatsinis, N.F., 1997. On the use of knowledge-based decision support systems in financial management: A survey. Decis. Support Syst. 20, 259–277. http://dx.doi.org/10.1016/S0167-9236(97)00002-X.