LUND UNIVERSITY

SCHOOL OF ECONOMICS AND MANAGEMENT

LUND UNIVERSITY

BACHELOR'S THESIS IN ECONOMICS

Nonstandard Errors in Bank Default Prediction Using Machine Learning

Author:

Emil Svalfors

Supervisor:

Anders Vilhelmsson

May, 2024

Abstract

This thesis analyses the risk of nonstandard errors affecting bank prediction using machine learning. Nonstandard errors are defined as the type of errors that occur during the Evidence Generating Process (EGP), meaning that these occur as a consequence of decision-making by researchers, rather than from sampling. The aim is to analyze how different choices of methods for pre-processing and data engineering create variation in the prediction performance of a classifier, hence signifying the existence of nonstandard error. This is done by creating 20 different pre-processing and data engineering pipelines consisting of different choices of methods. The variation in the performance of the different pipelines then gives an estimate of the nonstandard errors. By using recall, precision and ROC-AUC as scoring metrics, this thesis finds that the size of the nonstandard errors are smaller than the standard errors for recall and ROC-AUC. For precision, the nonstandard errors are larger. Overall, the size of the errors across the metrics were of similar magnitude. This thesis concludes that nonstandard errors are likely to affect predictions of bank defaults. The implication of this is that researchers always need to be aware of nonstandard errors and compare as many parts of the machine learning pipeline as possible.

Keywords— Nonstandard Errors, Machine Learning, Outlier Detection, Resampling, Feature Selection

Contents

1 Introduction

Predicting banking defaults is a critical area of research due the to the effects such events have on markets and the economy at large. Being able to correctly predict bankruptcies leads to saving various stakeholders from the costs of experiencing sudden bankruptcies (Faris et al., 2020). Furthermore, bank defaults are also known to cause general volatility in the financial markets, leading to a chain reaction of increased costs throughout society (Fiordelisi & Marques-Ibanez, 2013). To improve the accuracy of these predictions, machine learning has become a popular method of analysis. Because machine learning handles vast amounts of data efficiently it has become a good fit for bank default prediction.

Machine learning research is a broad subject as it applies to many different subject areas One way to describe the work where machine learning is applied in empirical science to produce new knowledge is as an evidence-generating process, EGP. This fits into the framework used by Menkveld et al. (2024) where the source of error in empirical studies stems from both the data-generating process, DGP and the evidence-generating process. Here, the DGP is the process where the scientist gathers samples from a population, leading to an error can be modeled as standard error. The EGP, on the other hand, is described by the authors as the process where knowledge is produced by processing the samples. This introduces another kind of error, called nonstandard error. The nonstandard error is described to exist because there is not one clear path to take when generating evidence in empirical science.

When working with samples, multiple decisions must be made on how to generate evidence. For each step in the EGP there are multiple choices to be made, which means that the process can be viewed as a forking path in a "garden of forking paths" (Menkveld et al., 2024). All forking paths then make a tree of paths that theoretically could have been chosen when evidence was generated. This reasoning can be applied to machine learning where there are multiple forking paths available. An example of this is when data cleaning where choices are made on which data seem fit for use. Other such steps are pre-processing and feature engineering such as choice of algorithms for resampling, outlier detection and feature selection/dimensionality reduction.

As machine learning is applicable to bank default prediction, this also means that the discussion of nonstandard errors can be applied for bank default prediction as well.

To analyze the extent to which the EGP affects the prediction error of bank defaults is then to discern how different choices regarding the machine learning process affect the error. To, in practice, analyze the nonstandard error in machine learning one first needs to find the available forking paths. Common forking paths include pre-processing and data engineering methods such as outlier outlier detection, resampling and feature selection.

Outlier detection involves the practice of erasing samples consisting of feature values that makes them rare. This implies that, within the feature space, the sample would not be found in close proximity to many other samples. Exactly what defines an outlier depends on what method is chosen for outlier detection. There exists a large variety of methods containing different interpretations of what an outlier would be, for example, statistical methods, but also non-parametric methods where distance or clustering is used (Smiti, 2020).

Another common pre-processing step is dealing with the data set imbalance regarding the number of non-defaults to defaults. As a default is a rare occurrence for a bank this leads to there being not many data points where a default has occurred. The issue when training a machine learning model is that the majority class, non-defaults, has a larger effect on training parameters for the model (Faris et al., 2020). This leads to the model not being able to predict bankruptcies of banks as the imbalance has led to the model not being tuned to find bankruptcies.

Within finance, there exists a large variety of different data points, leading to the problem of the curse of dimensionality. This is a problem within machine learning where the number of features (the dimension) is too large in comparison to the number of samples (i.e. the data becomes more sparse in the feature space). The issue is that this leads to overfitting as each sample is overinterpreted by the classifier, meaning that noise is interpreted as a significant relationship (Faris et al., 2020). This is seen when the classifier performs significantly worse on a test set than on a training set. The solution to this problem is then to use feature selection methods to reduce the number of features and therefore the dimension of the data (Faris et al., 2020). This forking path is then often included in machine learning as it solves the curse of dimensionality, which is often present in finance.

Described above are different parts of the machine learning process where non-

standard errors are likely to occur for empirical research within default prediction for banks. This study will analyze the risk of the nonstandard error affecting predictions by implementing several machine learning pipelines consisting of different choices regarding pre-processing and data engineering steps described above. The variation of predictions of all pipelines can then be used to calculate the nonstandard error, as in Menkveld et al. (2024), with the difference that the experiment is varying the methods, instead of research teams. The error is also calculated by analyzing the variance of different performance metrics. The quantification then opens for comparison to the normal standard error size-wise, which shows the risk of a potential nonstandard error being present.

1.1 Definition of Problem

The aim of this study is to analyze the risk of nonstandard errors affecting results for bank default prediction using machine learning. To achieve this goal various pipelines are constructed, all consisting of various combinations of pre-processing and data engineering methods. The nonstandard error is then calculated on three different scoring metrics, with focus on recall, to see both the size of the error and also if it varies among metrics. Finally this is compared to the standard error of calculating the three metrics.

The rest of the thesis is structured as follows: in section 2 previous research on the subject is reviewed. In section 3 the theory used for constructing the different pipelines and comparing them is presented, which is followed by section 4, where the method used to do this is described. In section 5 the data that was used is presented. The results are presented in section 6 and are further discussed in section 7.

2 Literature Review

This study will focus on bank defaults to analyze the extent to which nonstandard errors can affect machine learning predictions. Furthermore, the focus will be on the potential forking paths of various pre-processing and data engineering steps to see how the choice of path affects predictions. Even though the focus of machine learning research within finance tends to be on comparing different machine learning models, there have also been several studies looking at pre-processing methods.

Nonstandard errors within finance is a new area of research where Menkveld et al. (2024) has performed one of the first experiments on the subject. The experiment was motivated by the desire to quantify the nonstandard errors so that they could be compared to standard errors. They achieved this by having 164 independent research teams test six different hypotheses using the same dataset. As all research teams were given the same samples this led to the variation in predictions to be due to the nonstandard error. Menkveld et al. (2024) concluded that the nonstandard errors are likely to be of the same size as the standard errors.

When it comes to resampling, a comparative study researching resampling methods was conducted by Zhou (2013) when researching corporate default predictions. The researcher compared the oversampling methods of SMOTE and Random Oversampling With Replication (ROWR) with the undersampling method of Random Undersampling (RU), Undersampling Based on Clustering from Gaussian Mixture Distribution (UBOCFGMD) and Undersampling Based on Clustering from Nearest Neighbor (UBOCFNN). What was established was that undersampling tends to perform better on data where the number of bankruptcy observations is not too small while oversampling performs better when there are fewer observations of bankruptcies available.

Another comparison of different pre-processing methods was performed by Faris et al. (2020). In this comparison, the main steps compared were different feature selection algorithms, resampling and classifiers. For feature selection five different algorithms were tested: Correlation Feature Selection (CFS), Relief-F, Information Gain, Classifier Attribute Evaluator (CAE), and Correlation Attribute Evaluator (CorrAE). For resampling, SMOTE, Borderline-SMOTE and ADASYN were compared. The comparison was performed using a dataset on bankruptcies of Spanish financial and non-financial firms. The conclusion was that SMOTE performs best with the ADAboost

ensemble method paired with either CFS or Information Gain feature selectors.

Garcia (2022) has performed another study comparing different synthetic resampling methods using several different classifiers applied to US firm bankruptcy data. The study focused on six common synthetic resampling techniques: SMOTE, ADASYN, Borderline SMOTE, DB SMOTE, Safe-Level SMOTE, and SMOTE with CB undersampling. These were compared using 11 different classification models, with logistic regression, random forest, and KNN being examples of classifiers used. The performance of the different resamplers and classifiers was analyzed using six different metrics, with recall being prioritized as the problem was classifying bankruptcies. The performance of the different resamplers varied depending on what score was prioritized and which classifier was used. Garcia (2022) concludes that, in general, using SMOTE, ADASYN, DB SMOTE and SMOTE-CBU all lead to significant improvements in recall in general, in comparison to not using any resampler. When looking at the results, Borderline SMOTE can be seen to get high recall when applied to the logistic regression. It also had higher precision scores than the other resamplers.

Regarding resampling, another study that was performed by Kotb and Ming (2021) focused on resampler comparison. This study aimed to find what resampler and classifier would perform best in classifying defaults in premium payments. Here, nine different resamplers were compared, all belonging to what the researchers name the "SMOTE-Family", where all are different variations of the oversampling technique used in SMOTE. Used here were also eleven different classifiers. The researchers find that SMOTE-Tomek outperforms all other resamplers when it comes to both recall and ROC-AUC. They further concluded that sing SVM as a classifier together with SMOTE-Tomek gave the best overall results regarding all the different scoring metrics used.

When it comes to outlier detection, one study focusing on this is performed by Liu et al. (2008), where they propose a new outlier detector called isolation forest. The isolation forest uses a random forest architecture to find outliers. The outliers are isolated early in a decision tree as they by definition should be easily partitioned in the feature space. In the paper, the authors compared the performance of the isolation forest to other outlier detectors, namely ORCA (Outlier detection and Robust Clustering for Attributed graphs), LOF (Local Outlier Factor) and RF (Random Forest) by using

AUC score. The results demonstrated that the isolation forest outperformed ORCA on almost all datasets using AUC score. It also outperformed LOF and RF, though the two were not run on every dataset as computing complexity and memory size were a limitation for these algorithms. This was also connected to the other aspect measured, execution time, where isolation forest outperformed all methods when handling datasets with more than 1000 data points.

Another study focusing on outlier detection methods was conducted by Smiti (2020) where multiple outlier detectors were presented. The study was in the form of a literature review categorizing and reviewing outlier detection methods. The authors categorized the different methods into four categories: statistical methods, distance-based methods, density-based methods and clustering methods. Here several algorithms were presented for each category, examples being regression and histogram-based methods as statistical methods, local outlier factor for density-based methods and DBSCAN for clustering methods. The different categories of methods come with their respective advantages and disadvantages. The authors conclude that, while statistical methods can work well when the underlying distribution is known, they cannot be applied if it is unknown. Both distance and density-based methods solve this by being nonparametric and calculating distances in the feature space. The drawback of these two methods has instead to do with computing time where they can be inefficient for high-dimensional data. Cluster-based methods, such as DBSCAN are more efficient, but require tuning of multiple hyperparameters for high performance.

The third pre-processing and data engineering step discussed in this thesis is feature selection. A study done on feature selection has been performed by Battiti (1994) where a model based on mutual information that reduces the number of features used for classification was proposed. The mutual information metric is based on information theory and contains information on how much two features can be said to affect each other. By choosing the smallest available subset that also maximizes the mutual information to the target class, the model aims to optimize feature selection. Here the feature selector was compared to using Principal Component Analysis (PCA) and random dimensionality reduction using several different data sets. They found that the mutual information feature selection outperformed the other two techniques in terms of accuracy. Chen et al. 2020 have also conducted an experiment that touches upon

feature selection. In this experiment the aim was to construct a new method for optimizing prediction on protein-to-protein interaction. This was done by constructing a pipeline consisting first of biological encoders, then XGBoost for feature selection, followed by classification using an ensemble forest. To evaluate the effectiveness of using XGBoost for feature selection it was compared to other models, such as kernel principal component analysis, random forest and LightGBM. It was then shown that XGBoost outperformed the other methods for all scoring metrics, among them recall, AUC and precision. The study and the experiment presented here are two examples of comparisons of feature selection methods for machine learning.

3 Theoretical Framework

This study includes a comparison of the prediction performance of several different pipelines. The pipelines consist of a combination of different resamplers, outlier detectors and feature selectors. In total five resamplers, two outlier detectors and two feature selectors were used and combined into 20 different pipelines. All nine methods are presented in this section. After this, three different classifiers are also presented.

3.1 Resampling

Resampling is a method used for solving the issue of imbalanced datasets, which occurs when there, for a classification problem, exists a majority class that is unproportionally large in comparison to the other classes of the set (Faris et al., 2020). In this study, the dataset has a binary class of bankruptcy/not bankruptcy where the majority class is the samples consisting of not bankruptcy.

There exist several different solutions to handle the issue of imbalanced data (Faris et al., 2020). For example, one can separate them into data-oriented ones and algorithmic ones. Data-oriented solutions consist of algorithms that in different ways adjust the data so that it is not imbalanced when applied to the classifier (Werner de Vargas et al., 2023). These methods can be further divided into undersampling, oversampling and mixed methods. Undersampling methods reduce the number of samples of the majority class, while oversampling methods increase the number of samples of the minority class and the mixed ones use both methods (Werner de Vargas et al., 2023). This study will focus on various iterations of the SMOTE oversampling method and one mixed method called SMOTE-Tomek.

3.1.1 Synthetic Minority Oversampling Technique, SMOTE

The SMOTE technique, developed by Chawla et al. (2002), works by oversampling the minority class. This is done by creating new synthetic samples of the minority class by, for each sample, finding the k-nearest neighbors and then selecting a random point between the sample and all/some of the neighbors in the feature space (Chawla et al., 2002). This means that the whole feature space of the minority samples has been generalized as new points has been created in between already existing points in the

space.

3.1.2 Borderline SMOTE

Borderline-SMOTE is a method that complements SMOTE by only creating new synthetic samples along the borderline between the minority class and the majority class (Han et al., 2005). This is done by first, for all minority class samples, finding the K-nearest neighbors, where unlike for standard SMOTE all samples belonging to either class are considered. After this, the minority samples are filtered so that only samples for which:

$$\frac{K}{2} \leq N^i_{majority} < K \tag{1}$$

where K is the number of neighbors counted and $N^i_{majority}$ is the number of neighbors to sample i belonging to the majority class (Han et al., 2005). These samples belong to the borderline class, for which the synthetic samples are created.

This method generalizes the minority samples the same way as in ordinary SMOTE, but also aims to define the border more clearly in the feature space between the majority and the minority samples. The thought behind this is that the borderline samples are the ones that are most important for classification as these are the ones that are most probable to be misclassified (Han et al., 2005). Creating synthetic samples all belonging to the minority class on the border then creates a more defined border, leading to fewer misclassifications.

3.1.3 Adaptive Synthetic method, ADASYN

The adaptive synthetic (ADASYN) method, developed by He et al. (2008), is similar to borderline-SMOTE. K-nearest neighbors is used to create synthetic samples of the minority class. As with Borderline SMOTE the idea is to focus on borderline samples to reduce the number of misclassifications. The difference is that, instead of filtering those samples within a threshold of the number of neighbors in the majority class, a weight is introduced for each minority sample. This weight increases with the number of its K-nearest neighbors belonging to the majority class and is then normalized so that it becomes a distribution density (sum of all weights equalling one) (He et al.,

2008). The distribution density can then be used get the number of synthetic samples created from each sample.

3.1.4 SMOTE-Nominal Continuous, SMOTE-NC

SMOTE-NC is an oversampling method that was developed by Chawla et al. (2002) to handle a mix of numerical and categorical data. The other resamplers described in this thesis use the whole feature space for creating synthetic variables, leading to the dummy variables being interpreted as continuous. This leads to the synthetic samples not having binary values for the created dummy variables. To handle this, SMOTE-NC first calculates the median of all standard deviations over all continuous feature values for all samples in the minority class. The median can then be used as a penalty in the way that it is added to the Euclidean distance between two samples if the nominal feature is differing, giving the total distance between two samples. The synthetic sample is created the same way as in SMOTE for the continuous features while the nominal features are given the same value as the majority of features have in the K-nearest neighbors (Chawla et al., 2002). This would then lead to all categorical features being assigned binary values.

3.1.5 SMOTE-Tomek

The SMOTE-Tomek resampler, proposed by Batista et al. (2004), is a hybrid approach consisting of both an oversampler and an undersampler. The oversampler is the ordinary SMOTE while the undersampling is performed by a Tomek links undersampler. The Tomek link is by the authors described as a relation that can exist between two samples belonging to different classes in the feature space. There exists a Tomek link between two samples if there are no other samples belonging to any class having a shorter distance to any of the two samples in question (Batista et al., 2004). The Tomek links undersampler is then developed to find majority class samples belonging to a Tomek link and erase them, which leads to the undersampling erasing majority samples that are close to the minority class in the feature spaces. This could then lead to easier classification for the classifier as there are less ambiguous points in the feature space.

3.2 Outlier Detection

Outlier detection is an important part of the pre-processing pipeline because, depending on the methods used, an outlier can have a large effect on pre-processing and classification (Smiti, 2020). When, for example, implementing a resampler used to balance the dataset, an outlier could lead to resampling inflicting a bias on the minority class. This is because a vector-based resampler working in the feature space would potentially connect the outlier to the rest of the minority class data points, leading to new synthetic samples being located in the majority class in the feature space (Smiti, 2020). The assumption made in the example of resampling outlier data is that the outlier is noise to the dataset, meaning that its location in the feature space is due to reasons not being modeled for. As the variation in noise is unmodeled, the outlier causing it should be erased so that it does not affect the classification or resampling. Note that outliers are not treated as erroneous values in this thesis. The assumption is that they simply are not captured by the model due to the limitations of the variables used or the complexity of the classifier. Hence the outliers are removed, not because they are incorrect, but because they are potentially worsening the training of the classifier.

In this study two different outlier detectors are used: DBSCAN and Isolation Forest.

3.2.1 Density-Based Spatial Clustering of Applications with Noise, DBSCAN

DBSCAN, developed by Ester et al. (1996), is a clustering algorithm that works as an outlier detector by dividing the data into clusters where data left out are assumed to be outsiders which can be removed. As described by the authors, the algorithm operates by first finding all points for which there are other points within a pre-set maximum distance (neighbors) in the feature space and then counting them. They mean that all points for which there are more neighbors than a pre-defined threshold are core points, which then form a cluster with other core point neighbors. Then, other points that have a core point as neighbor are also added to the cluster and points that have no core points as neighbors are defined as outliers. As both the maximum distance and minimum number of neighbors are predefined, these two parameters need to be tuned manually.

3.2.2 Isolation Forest

Isolation forest is another method that can be used for outlier detection and removal (Liu et al., 2008). It works differently from many other outlier detectors by aiming at finding anomalies directly, in contrast to algorithms such as DBSCAN where normal values are found and grouped in order to find what data points are not included (Ester et al., 1996). Isolation forests instead partition the feature space randomly for each tree. If a data point is an anomaly it should on average be completely partitioned off faster than a normal value. For a decision tree, this means a shorter distance to the root of the tree. After partitioning, each data point is given an anomaly score, aggregated for all trees in the forest (Liu et al., 2008).

3.3 Feature Selection

Feature selection is an important step in the machine learning process when it comes to finance, the reason for this is the need to tackle the issue of overfitting (Faris et al., 2020). Another method, other than feature selection, of dealing with overfitting is dimensionality reduction, where principal component analysis is well-used. One key part of dimensionality reduction is through the replacement of features for new ones, leading to loss of interpretability, which is important in bankruptcy prediction (Faris et al., 2020).

There exist three main feature selection methods: Filter-, wrapper- and embedded methods. Filter methods tweak the dataset before it is applied to a classifier using statistical methods. Wrapper methods work by iteratively improving classification performance by tweaking the feature selection. Embedded methods instead have the feature selection built into the classifier learning process. An example of this is the LASSO method which works as a linear regression with a term included which potentially reduces the impact of certain features.

In this study one filter method and one wrapper method are used, namely mutual information for filtering and XGBoost for wrapping.

3.3.1 Mutual Information

Mutual information is an information-theoretic measure where the mutual information is a measure of the amount of information two variables holds of each other (Battiti, 1994). It is calculated as:

$$MI(X,Y) = \sum\sum p(x,y)log(\frac{p(x,y)}{p(x)p(y)}) \tag{2}$$

where $p(x)$ and $p(y)$ are the probability density function of the respective variables and $p(x,y)$ is the joint probability density function of the two variables. This measure can then be applied to feature selection by identifying the subset of features that are most informative about the class (default/no default). This is done by finding the smallest subset that also maximizes mutual information (Battiti, 1994).

This method can be compared to the correlation metric, where the relationship between two variables also is measured. One main difference though, lies in the linearity of the correlation metric, which is not present for mutual information, rendering it more generalizable (Battiti, 1994).

3.3.2 XGBoost

XGBoost is a gradient boosting algorithm used for classification problems that was developed by Chen and Guestrin (2016). This method is an ensemble method containing pre-processing and feature engineering. Gradient boosting uses iterative classification to optimize hyperparameters for improving scoring. This is done by using a base learner, here a decision tree is used, which then is optimized iteratively using a scoring function. Even though XGBoost can be used for classification in itself, it can also be used for feature selection by its use of feature importance, as in Chen et al. 2020. Feature importance is built into the method where certain features are selected for their effect on performance. As XGBoost works iteratively and is optimized by performance, this means that the feature selection in this instance works as a wrapper method.

3.4 Classification

For machine learning classification there are multiple models available, from logistic regression, multi-layered perceptron networks to deep learning algorithms. Presented in this section are some of the most used ones in research.

3.4.1 Logistic Regression

The logistic regression is a machine learning binary classification method where the data is fitted to a logistic curve:

$$P(x) = \frac{1}{1 + e^{-\beta_0 - \sum_{i=1}^{n} \beta_i x}} \tag{3}$$

where $x = (x_1, x_2, \ldots, x_n)$ are all features and β_i are the coefficients that are optimized for best classification. The optimization is performed using the cross entropy function, where the loss for sample k with class y^k is:

$$l_k = -y^k k ln(P(x^k)) - (1 - y^k) ln(P(1 - x^k)). \tag{4}$$

The optimization is then done by minimizing the cross entropy over all samples, with a L_2 regularization term added L:

$$L = \sum_{k=1}^{n} l_k. + \lambda \sum_{i=1}^{n} \beta_i^2. \tag{5}$$

Here λ is a regularization term that, when minimizing L, reduces overfitting and therefore reduces volatility of the weight values when L is iteratively minimized using numerical optimization.

3.4.2 Random Forest

The random forest classifier is an aggregated decision tree model developed by Breiman (2001). He describes the decision tree as a tree-based algorithm that seeks to partition the data into different classes. The algorithm works by dividing the feature space into smaller subsets, which is equivalent to dividing the samples into leaves in a tree. The tree separates samples into smaller subsets until there either only exists samples belonging to one class in a node, making it a leaf, or until the predefined maximum depth has been reached. The forest aggregates several different trees by randomly sampling the data and the features to use for each tree. The classification is made by a voting system where, for binary classification, a majority vote decides the class of a sample.

3.4.3 Perceptron Network

Popescu et al. (2009) describes the perceptron network as a machine learning model that uses the perceptron as a model to classify data. Here, the perceptron consists of an

activation function, a weight per input signal and a bias and the perceptron operates by applying the activation function to the weighted input signals and bias. The network is structured by connecting perceptrons via the inputs and outputs as all perceptrons in one layer have as input the output of all perceptrons in the layer above. The first layer of perceptrons has the system input signals as input and the last layer of perceptrons output is aggregated into one or more perceptrons (depending on the dimension of the classification problem) whose output signal is the classification made. The network is trained by back-propagating the error to adjust each weight using some optimization technique.

4 Methodology

Throughout the study described in this thesis python was used to create all pipelines, to compare them and to create all tables. For efficient coding, the Github Copilot was used, which is an AI-powered code generator developed by Gitbhub and OpenAI (GitHub, Inc., n.d.). It was used for fast auto-completion and error handling.

On a general level, the goal of this study was to identify the possible size of the nonstandard error of the machine learning process when applied to bank defaults with a focus on the pre-processing and data engineering steps. Therefore several different machine learning pipelines were constructed, all consisting of every possible combination of the different pre-processing and data engineering methods described above. This led to every pipeline being able to be viewed as a unique path in the "garden of forking paths" as there has been a unique combination of choices made on every fork on the path. Thereafter the predictive performance of the different pipelines was measured by applying different scoring metrics. Lastly, the variance of the scores for all pipelines was used to estimate the standard error and the nonstandard error.

4.1 Pipeline Construction and Data Cleaning

The first part of the study consisted of data cleaning, which can be described as the pre-processing that could be done for all data, without risking data leakage between the training and test sets. After having looked at the dataset, missing data were noticed for almost all variables for certain samples. These samples were erased. Certain categorical variables, such as state operated in and regulator were encoded into dummy variables.

In the next step of the study 20 different pipelines were constructed. Each pipeline, visualized in figure 1 consisted of five different steps. The same data standardization and imputation methods were used in all pipelines. The choice of using standardization instead of normalization was made due to the logistic regressor being used, which is sensitive to different feature scales. Standardization is also more robust to outliers, which have not been erased yet. The numerical features were standardized using the mean and standard deviation. The mean was used for imputing the numerical features, while there were no categorical features missing. All three next steps of the pipeline consisted of a unique combination of the resamplers, outlier detectors and feature

selectors described above and presented in table 1.

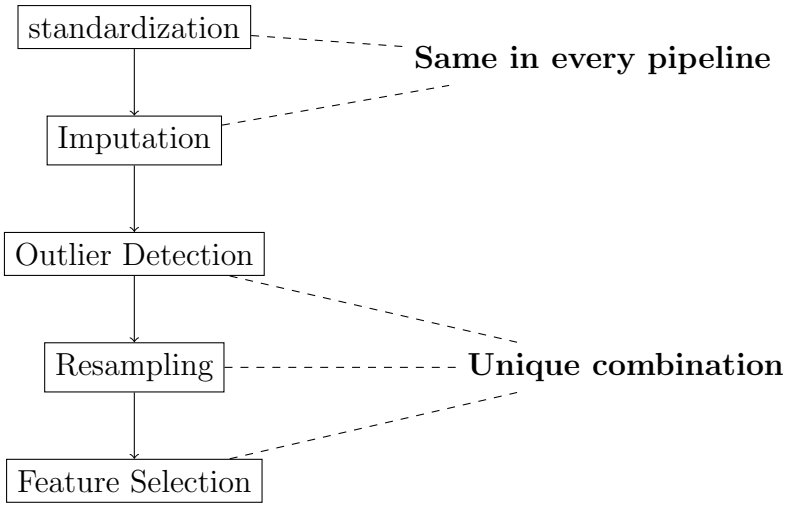

Figure 1: A general visualization of the machine learning pipelines used in this study.

Table 1: Resampling, Outlier Detection, and Feature Selection Techniques used when constructing pipelines.

Resampling	**Outlier Detection**	**Feature Selection**
SMOTE	DBSCAN	Mutual Information
Borderline SMOTE	Isolation Forest	XGBoost
ADASYN		
SMOTE-NC		
SMOTE-Tomek		

The various methods presented in table 1 in each step were chosen as most have been used and favored in previous research. To note is that DBSCAN is mentioned to be efficient if hyperparameters are tuned correctly (Smiti, 2020), which is one of the reasons for this study using five-fold cross-validation for hyperparameter tuning. Regarding SMOTE-NC it does not appear in previous research as it was not part of any experiments found by this thesis. Two arguments can be used for including it. Firstly, the method is built on SMOTE, which is favored in many papers, for example in Faris et al. (2020) and Garcia (2022). Secondly, with the addition that it handles categorical variables, it could potentially add more information to the classifier, leading to improved performance.

4.2 Hyperparameter Tuning and Testing

Before using all pipelines to classify the data and measure performance, a classifier was chosen. Here, multiple aspects were of importance. The first aspect was that the objective was not to optimize prediction performance, but to find the variation in performance depending on which pipeline was being used. This meant that it would be unnecessary to use a computing-heavy model, such as random forest, to classify the data. The second aspect was the fact that in finance and bankruptcy prediction, interpretability can be necessary. This is because an institution then can analyze the reasons for a predicted default. Therefore, a supervised machine model is necessary. These two aspects led to the logistic regression to be chosen as the model to use in this study. The logistic regression is, in comparison, quick and the coefficients are interpretable in the sense that they can be ordered by importance.

Before using the pipelines a training/test set split was performed so that no data leakage would occur. The split was performed so that the ratio of bank default to non-defaults was kept. Then, using only the training data, five-fold cross-validation was used for hyperparameter tuning on each method using some different pipelines. For all five splits, the training data (fours of the folds) was then processed as described in figure 1 where the validation set (one of the folds) was run in parallel. This led to the validation set being standardized and imputed using the same metrics as calculated for each training fold. The validation set was also affected by the feature selection where the features not selected for the training set were erased also from the validation set. The output of each pipeline was all training sets and validation sets, in a processed form. For each split the processed training set was then fitted to the logistic regression classifier, which then was used to predict the validation set. The predictions were then scored in comparison to the real default category of the validation set using recall, precision and ROC-AUC. These scores were aggregated for all five set-pairs using mean. This whole process was then used to tune all hyperparameters in all steps in the pipelines. To note is that recall was the parameter used for actual tuning, as different scores can lead to different conclusions one had to be chosen. Here recall was chosen as it is a standard metric to use for default prediction, it is for example used by Faris et al. (2020), Zhou (2013) and Garcia (2022).

The hyperparameters tuned and their settings are presented in table 2. For the re-

samplers SMOTE, Borderline SMOTE and ADASYN the hyperparameter tuned was how many neighbors to include in k-nearest neighbors (called n-neighbors in ADASYN) for creating synthetic samples. For SMOTE-NC a mask of the categorical variables indices was used, though it was not tuned, only compared to not marking the categorical features at all. For SMOTE-Tomek different SMOTE versions were tried in tuning. For DBSCAN the tuning consisted of finding optimal epsilon, which is the distance used when finding neighbors. Also the neighbor threshold for being a core point, *min_samples* was tuned. For Isolation Forest the number of base estimators, *n_estimators*, was tuned. When it comes to mutual information what was tuned was how many features to include when SelectKBest, a Scikit program (Pedregosa et al., 2011), fetched the best features according to the mutual information metric. At last, for XGBoost the base learners were tuned in both max depth and maximum amount of children. The learning rate, eta, was also tuned. Also, similar to in mutual information, a package, here called SelectFromModel (Pedregosa et al., 2011) was used to get the features that performed best using a threshold.

Table 2: Settings of all hyperparameters that were tuned in the study if not included then the hyperparameter used was standard.

Category	**Method**	**Hyperparameters**
Resampling	SMOTE	$k\_neighbors = 3$
	Borderline SMOTE	$k\_neighbors = 4$
	ADASYN	$n\_neighbors = 5$
	SMOTE-NC	$categorical\_features = categorical\_mask$
	SMOTE-Tomek	$smote = SMOTE()$
Outlier Detection	DBSCAN	$eps = 2.75$, $min\_samples = 5$
	Isolation Forest	$n\_estimators = 100$
Feature Selection	Mutual Information	SelectKBest: $score\_func = mutual\_info\_classif\ k = 7$
	XGBoost	XGB: $eta = 0.1$, $max\_depth = 3$, $min\_child\_weight = 5$ SelectFromModel: $threshold =' mean'$

When hyperparameter tuning had been performed on all pipelines, they were tested

using the test set. The testing was performed similarly to the validation described above. The training data, now consisting of 80% of all data points, were together with the test set, consisting of 20 % of the data, copied 20 times. After this, each copied pair of both sets were processed by one pipeline, where the test set, in parallel, was modified in the same way as for the validation test. The testing was performed by training the classifier on the processed training set and then having it predict the default class using the test set. The predictions were scored by comparing the predictions with the correct classes for all samples in the test set.

4.3 Scoring Metrics

One important decision made in this study was the choice of scoring metrics for comparing the different pipelines in performance so that the nonstandard error could be measured. A standard way of scoring the performance of a binary classifier is to look at a confusion matrix as in table 3.

		Predicted	
		Positive	**Negative**
Actual	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 3: Confusion Matrix describing all four possible outcomes of binary classification.

Here a standard metric is accuracy:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}, \tag{6}$$

in other words, the ratio of correct predictions. Depending on what is studied though, this metric might miss important information (Faris et al., 2020). For example, in bankruptcy prediction where bankruptcy is a low-probability event, there is a need to look at type 1 and type 2 errors. Especially type 2 errors, meaning the occurrence of False negatives, is important as missed bankruptcies are costly, and the goal is often to find the high-risk banks before bankruptcies occur. Therefore, two other

metrics, recall and precision, were used:

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

$$Precision = \frac{TP}{TP + FP}. \tag{8}$$

Here, recall is the ratio of correctly predicted bankruptcies to the total amount of bankruptcies while precision is the ratio of correctly predicted bankruptcies to total predicted bankruptcies. Recall can be defined as the accuracy if only defaults are analyzed and precision the efficiency of finding defaults. A normal scoring prioritization could then be described as achieving as large a recall as possible without the precision being too small. This is equivalent to wanting to be able to find all bankruptcy-prone banks without there being too many banks being falsely flagged as high risk.

In this study, the goal was not to find the best-performing pipeline, but to find how the scoring varies among pipelines. Therefore the choice of scoring metrics was mostly based on what is standard in the literature. One more metric was included and looked at: the receiver operating characteristic curve where the metric was Area Under the Curve, in short: ROC-AUC. The ROC curve plots recall against the false positive rate. The false positive rate is defined as: $FPR = \frac{FP}{FP+TN}$ meaning the ratio of false default predictions to all non-defaults. The plotting is performed by step-wise changing the decision threshold of the classifier, from all samples classified as non-defaults to all being classified as defaults. A perfect classifier would then mean that the AUC would equal one. The AUC is the area under the curve constructed. Since the ROC-AUC assesses the classification on all thresholds, this means that it is a more suitable metric for imbalanced datasets (Zhou, 2013). The AUC can be interpreted as the probability that the classifier will rank a random default sample higher than a random nondefault sample, i.e. giving a higher probability of the sample being a default one. All three scoring metrics described: recall, precision and AUC were used in calculating the standard and nonstandard errors whereas recall was used in hyperparameter tuning.

4.4 Estimating Errors

To quantify the risk of nonstandard errors occurring this study used bootstrapping to estimate the standard error stemming from the SGP while calculating the interquartile range to estimate the nonstandard error stemming from the EGP.

4.4.1 Standard Error

The standard error was calculated using bootstrapping, which is the practice of random sampling using replacement. By doing bootstrapping multiple times estimating the standard error was made possible by calculating the standard deviation using the variance of the scoring metrics over all reruns. In practice, this meant that one of the pipelines was chosen and thereafter the bootstrapping was run 30 times, giving 30 samples of each scoring metric for which standard deviation was calculated.

4.4.2 Nonstandard Error

In the study performed by Menkveld et al. (2024) the model used for measuring the nonstandard error was the Interquartile Range, IQR:

$$IQR = Q_{0.75} - Q_{0.25}, \tag{9}$$

where $Q_{0.75}$ and $Q_{0.25}$ were the two quartiles. One reason for using IQR is that the measure is nonparametric, meaning that there is no assumption made of the underlying distribution of the samples. This fits the study presented in this thesis as it used both continuous and binary variables and whereas some, for example age, were constructed manually. The IQR was calculated for each scoring metric and then compared to the standard error.

5 Data

The data used in this study consists of yearly data of US banks consisting of bank-specific variables and also whether a bank has failed or not according to the Federal Deposit Insurance Corporation (FDIC) between the years 1992-2018. Most of the data is numeric while some of it is categorical. The data used is presented in table 4. The categorical data consist of state location, what regulator is assigned to the bank, sector operated in and type of bank. The dataset consists of 243282 samples where 662 of which belong to the default class and the rest to the non-default class.

Table 4: Bank Data Variables

Variable	**Description**
Regulator	Regulatory authority overseeing the bank.
Sector	The industry sector in which the bank operates.
BadEvent	Indicator for default/non-default
Loss_Loan	Loss allowance to loan.
CFEA	Cost of funding with earning assets.
NIM	Net interest margin.
NOI_ASS	Net operating income to assets.
ROA	Return on assets.
ROE	Return on equity.
RE_EQ	Retained earning to average equity
EFF	Efficiency ratio.
LOSS_NPL	Loss allowance related to non-performing loans.
NCASS_ORE	Non-performing assets and other real estates to assets
NLOAN_CDEP	Net loans to core deposits.
LEV	Leverage ratio.
CAR	Capital adequacy ratio.
EQUITY	Total equity of the bank.
log_equity	Natural logarithm of the equity.
NPL	Non-performing loans.
Age	Years since established
stalp	state operated in
bkclass	type pf bank

The data set used was at first constructed by Hardarson and Vuono (2019) when replicating parts of a study performed by Portopoulos (2020), where the CAMELS framework was used when finding variables to analyze bank defaults. To note is that, as the goal of the study presented in this thesis was not to optimize performance, but to see the variation from using different pipelines, the data was not analyzed much further before data cleaning was performed. As most methods in the different parts of the pipeline were nonparametric, this meant that no assumption had to be made on the distribution and other underlying statistics of the data used.

6 Results

After all pipelines were constructed and tested on the test set three scoring metrics were used to measure their performance. The scores for each pipeline are presented in section 6.1. After this the nonstandard errors were calculated using the IQR on the results for each scoring metric. The nonstandard errors were compared to the standard errors, calculated by running one of the pipelines on bootstrapped samples. The comparison of the two errors for each scoring metric is presented in section 6.2.

6.1 Pipeline Scoring

In table 5 the recall test score is shown for each pipeline tested. The table is sorted on the scoring metric, meaning that the top pipeline is the one with the highest score. In the table, it can be seen that the best-performing pipelines when it comes to recall are the SMOTE Tomek-DBSCAN-XGBoost and ADASYN-DBSCAN-XGBoost pipelines with a score of 0.932. The scoring means that 93.2 % of all bankruptcies were correctly predicted as such according to equation (7). Other aspects are that using Borderline SMOTE in all pipeline variations led to the worst performance in recall and that DBSCAN, in general, outperforms Isolation Forest.

Table 5: Recall score for each pipeline when fitted to the test set. Rounded to three decimals

Resampler	**Outlier Detector**	**Feature Selector**	**Recall**
SMOTETomek	DBSCAN	XGBoost	0.932
ADASYN	DBSCAN	XGBoost	0.932
SMOTE	DBSCAN	XGBoost	0.924
SMOTENC	DBSCAN	Mutual Information	0.917
ADASYN	IsolationForest	XGBoost	0.917
SMOTE	DBSCAN	Mutual Information	0.917
ADASYN	DBSCAN	Mutual Information	0.917
SMOTENC	IsolationForest	XGBoost	0.917
SMOTETomek	DBSCAN	Mutual Information	0.917
SMOTE	IsolationForest	XGBoost	0.909
SMOTE	IsolationForest	Mutual Information	0.909
SMOTENC	IsolationForest	Mutual Information	0.909
ADASYN	IsolationForest	Mutual Information	0.909
SMOTETomek	IsolationForest	XGBoost	0.909
SMOTETomek	IsolationForest	Mutual Information	0.902
SMOTENC	DBSCAN	XGBoost	0.894
BorderlineSMOTE	IsolationForest	Mutual Information	0.894
BorderlineSMOTE	DBSCAN	Mutual Information	0.894
BorderlineSMOTE	IsolationForest	XGBoost	0.886
BorderlineSMOTE	DBSCAN	XGBoost	0.886

In table 6 all pipelines are listed and sorted by the precision test score. Here the best-performing pipeline is Borderline SMOTE-Isolation Forest-Mutual Information. The fact that this pipeline performs best regarding precision means that it has the highest ratio of true predicted defaults to all predicted defaults. The precision of 0.140 means that 14 % of all predicted defaults were true defaults. Generally, all pipelines using Borderline SMOTE scored the highest scores and Isolation Forest tends to outperform DBSCAN.

Table 6: Precision score for each pipeline when fitted to the test set. Rounded to three decimals.

Resampler	**Outlier Detector**	**Feature Selector**	**Precision**
BorderlineSMOTE	IsolationForest	Mutual Information	0.140
BorderlineSMOTE	IsolationForest	XGBoost	0.126
BorderlineSMOTE	DBSCAN	XGBoost	0.112
BorderlineSMOTE	DBSCAN	Mutual Information	0.107
SMOTENC	IsolationForest	Mutual Information	0.081
SMOTENC	IsolationForest	XGBoost	0.080
SMOTE	IsolationForest	Mutual Information	0.080
SMOTE	IsolationForest	XGBoost	0.079
SMOTETomek	IsolationForest	Mutual Information	0.078
SMOTETomek	IsolationForest	XGBoost	0.076
ADASYN	IsolationForest	Mutual Information	0.072
ADASYN	IsolationForest	XGBoost	0.071
SMOTENC	DBSCAN	XGBoost	0.052
SMOTENC	DBSCAN	Mutual Information	0.052
SMOTE	DBSCAN	Mutual Information	0.051
SMOTETomek	DBSCAN	Mutual Information	0.050
ADASYN	DBSCAN	Mutual Information	0.047
SMOTETomek	DBSCAN	XGBoost	0.045
SMOTE	DBSCAN	XGBoost	0.043
ADASYN	DBSCAN	XGBoost	0.043

Finally, in table 7 all pipelines are listed and sorted by ROC-AUC score. According to the table, the best performing pipeline in terms of AUC score is the SMOTENC-Isolation Forest-XGBoost pipeline. In general the Isolation Forest outperforms DBSCAN for outlier detection. To note is that the SMOTENC resampler is both part of the top performing pipeline and the worst performing pipeline.

Table 7: ROC-AUC score for each pipeline when fitted to the test set. Rounded to three decimals

Resampler	**Outlier Detector**	**Feature Selector**	**ROC-AUC**
SMOTENC	IsolationForest	XGBoost	0.944
ADASYN	IsolationForest	XGBoost	0.942
SMOTENC	IsolationForest	Mutual Information	0.941
SMOTE	IsolationForest	Mutual Information	0.940
SMOTE	IsolationForest	XGBoost	0.940
BorderlineSMOTE	IsolationForest	Mutual Information	0.940
SMOTETomek	IsolationForest	XGBoost	0.939
SMOTETomek	DBSCAN	XGBoost	0.939
ADASYN	IsolationForest	Mutual Information	0.938
ADASYN	DBSCAN	XGBoost	0.937
BorderlineSMOTE	DBSCAN	Mutual Information	0.937
SMOTETomek	IsolationForest	Mutual Information	0.936
SMOTENC	DBSCAN	Mutual Information	0.936
SMOTE	DBSCAN	Mutual Information	0.935
BorderlineSMOTE	IsolationForest	Mutual Information	0.935
SMOTETomek	DBSCAN	Mutual Information	0.935
SMOTE	DBSCAN	XGBoost	0.934
BorderlineSMOTE	DBSCAN	XGBoost	0.934
ADASYN	DBSCAN	Mutual Information	0.933
SMOTENC	DBSCAN	XGBoost	0.925

6.2 Standard and Nonstandard Error

For calculating the errors, the three different scoring metrics were used. First, the nonstandard errors were calculated by using the IQR as in equation 9. This was done by using all recall scores in table 5. The same was done using the precision scores in table 6 and the AUC-ROC scores in table 7.

The standard error was calculated by choosing one of the top performing pipelines for recall, ADASYN-DBSCAN-XGBoost and then using bootstrapping on the training

set to train the pipeline and classifier. Thereafter the classifier was fitted to the test set, which was constructed in the same bootstrap. All three scores were calculated for each of the 30 runs of bootstrapped data processed using the ADASYN-DBSCAN-XGBoost pipeline. Here the limited computing power led to not all pipelines being tested, therefore the top-performing pipeline in recall was chosen. The standard error was calculated by taking the standard deviation of the 30 instances for each score. All error estimates were presented in table 8.

Table 8: Standard and nonstandard error calculated for recall, precision and AUC

Metric	**Standard Error (Standard Deviation)**	**Nonstandard Error (IQR)**
Recall	0.036	0.017
Precision	0.007	0.029
AUC	0.018	0.004

Table 8 shows that the variation depends on which scoring metric was calculated. For both recall and ROC-AUC then standard deviation, i.e. the standard error, was larger than the estimated nonstandard error. For precision, the nonstandard error was estimated to be larger.

7 Discussion and Conclusion

In the discussion first the results in 6.2 and then the results in 6.1 are discussed in relation to the problem formulation. After this the existence of nonstandard errors will be discussed, followed by an analysis of limitations and also future research. This section ends with a conclusion of the findings.

7.1 Error analysis and Pipeline Comparison

The aim of this study was to analyze the risk of nonstandard errors affecting predictions for bank defaults. After having constructed pipelines, each consisting of a unique combination of pre-processing and data engineering methods, the variation in performance was measured using IQR. In table 8 the results are presented and compared to the standard error for the three different scoring metrics used. For both recall and AUC the nonstandard error was estimated to be lower, while for precision it was estimated to be higher.

The results in section 6.2 could mean that it is inconclusive as to the extent of the severity of the nonstandard error. To remember though, is that recall is the metric that is often used within bank default prediction, leading to the result pointing towards the nonstandard errors being smaller than standard errors. However as there are no confidence intervals for the error metrics themselves, one cannot say exactly how much they are likely to differ, as they are still closer than for the two other scoring metrics.

The three tables: 5, 6 and 7 show all pipelines sorted after performance in recall, precision and ROC-AUC. The recall scoring table shows that in general, DBSCAN outperforms Isolation Forest, while XGBoost is often better than using Mutual Information. Borderline SMOTE seems to bring the most decisive difference as all pipelines incorporating it perform the worst. However, the issue is that even though some methods in general outperform others in each category, there is no unambiguous choice to make in each step. This means that the best-performing method in a step depends on what methods have been chosen in the other steps. As an example: if SMOTENC would be chosen for resampling paired with the DBSCAN outlier detector, then Mutual Information gives best performance. If SMOTENC instead would be paired with Isolation Forest, then XGBoost gives best performance. Therefore, the best choice of

method in a step depends on what other methods have been chosen in the other steps.

In the precision scoring table (6) the pipelines are closer to being unambiguous in the performance where Borderline SMOTE again brings the most decisive difference as it always outperforms the other resamplers. This is followed by the choice of method in outlier detection. Disregarding feature selection, choosing Borderline SMOTE and Isolation Forest is always best, not depending on other choices. The other resamplers are also unambiguous if an outlier detector has been pre-selected. Though, here too the choice of feature selection is not unambiguous. For example, the Mutual Information selector is better for Borderline SMOTE and Isolation Forest, while not for Borderline SMOTE and DBSCAN.

The third scoring metric used was ROC-AUC and the pipeline performance is listed in table 7. In general Isolation Forest can be seen as outperforming DBSCAN for outlier detection. For resampling, the SMOTENC resampler scores both the highest and the lowest scores. As for the other scoring metrics, there is still ambiguity as to which method performs best, as it depends on what other methods has been chosen.

When aggregating the results discussed for all three scoring metrics, one first remark is that the choice of method in the feature selection step has less effect on performance than the choice made in the other steps. This is seen in all three tables as the feature selector methods never are all collected on one end of the sorted list. For all three scoring metrics, the most decisive metric is the outlier detector. For recall and precision if he resampler is Borderline SMOTE or not is also a decisive metric. A main occurrence in all tables is that all pipelines lack unambiguity in the ranking of methods. There exist no method that consistently outperforms the other methods if one were to disregard the other steps in the pipeline.

7.2 The Existence of Nonstandard Errors

One necessary discussion is the possible difference between the nonstandard error estimate in this study and the nonstandard errors discussed in Menkveld et al. (2024). In their study, the nonstandard errors are defined not as erroneous, but erratic as there is no one correct path to take within the EGP. For the analysis of pipeline variation in the study discussed in this thesis to be compatible with the definition of nonstandard errors, all pipelines need to be equally correct choices. Within machine learning stud-

ies, standard practice is to incorporate different methods to compare and then choose the top performing one, as done in Faris et al. (2020). This would then mean that the comparison done in this study is not viable as a forking paths problem, as the best one simply is chosen for each study. If the best one is always chosen, then the nonstandard error would be zero as there would be no variation in the EGP. However, there are several aspects in the study presented in this thesis pointing toward the existence of nonstandard errors.

One aspect that points to the fact that nonstandard errors are an issue within this study is the order in which the different pipelines are ranked. There exists no unambiguous order of methods for the pre-processing categories for either of the three scoring metrics. This means that within each category, there does not exist any alternative that always is better than the others. The top-performing choice depends on all the other choices. This demonstrates that there exists no top-to-bottom order of choosing methods, as each method must be evaluated with all other methods across all stages of pre-processing and data engineering.

Throughout the study, the methods chosen have been chosen because of previous studies pointing towards them being well-performing and also because they have been good choices in regard to the data used, examples being Faris et al. (2020) and Zhou (2013). Hence, even though the pipelines have not performed equally well, they have all been viable choices. This implies that modeling the variation as nonstandard errors and calculating it with the interquartile range would be possible.

Another main reason for the results pointing toward the possibility of nonstandard error is the fact that not many studies include multiple categories of pre-processing and data engineering in their analysis. Comparative studies, such as the one performed by Faris et al. (2020) includes a comparison of one instance of the pipeline, in their case the feature selection, and different classifiers. Had their study included outlier detector it could have led to other results. This discussion can be formalised using set theory:

Let the set of all possible pipelines for predicting bank defaults be Ω. With all possible pipelines, it is implied that all steps used here and more are regarded. Let ω^* be the optimal pipeline in terms of performance for a pre-set scoring metric, f, so that: $f(\omega^*) \geq f(\omega)$. Then if a study tries all pipelines, i.e. all possible combinations of methods, then ω^* would always be found and the nonstandard error would be zero. In

practice though, researchers only look at a subset of all pipelines: $\Omega^{'} \subset \Omega$, which then gives them the best pipeline, ω' for which $f(\omega^{'}) \leq f(\omega^{*})$. This leads to the fact that when several researchers do the study, they will use different $\Omega^{'}$, giving different $\omega^{'}$, leading to $f(\omega^{'})$ varying among researchers, which means that the nonstandard error now is non-zero.

The study in this thesis includes more pre-processing and data engineering steps than many other studies, leading to a larger subset of pipelines. What this then leads to is that the variation in results shown in this study are indeed potential nonstandard errors. This is because multiple other studies have a smaller subset of pipelines, leading to a final error in prediction not being due to sampling, but because they had not included a comparison of all steps in the pre-processing and data engineering process. What this shows is that to reduce nonstandard errors, as many parts of the pipeline as possible need to compare multiple methods. This result aligns with the one in Menkveld et al. (2024) reasonably well, as the conclusion they make is that nonstandard errors are likely to affect empirical research within finance. One difference between this study and Menkveld et al. (2024) is the possibility of ranking the variation leading to nonstandard errors in the study presented in this thesis. When looking at the results in section 6.1 all pipelines are ranked. This then gives the possibility of analyzing potential nonstandard errors while also analyzing performance. One further difference lies in nonstandard error mitigation. One of their conclusions is that integrating more peer-reviewing stages reduces nonstandard errors, while in this thesis it is proposed to integrate more stages into the machine learning process for comparison before choosing a pipeline.

7.3 Limitations and Further Research

What this study has shown is that there is a possibility of nonstandard errors existing for bank default predictions using machine learning. The general size of the errors across different scoring metrics points towards this possibility. One limit that has been present throughout this study is the limited computing power available. Firstly this has led to not all methods being available, an example being Generative Adversarial Networks for resampling. The limited computing power has further limited the bootstrapped estimation of the standard error using standard deviation. Here only 30 iterations of one pipeline were used because of this. For more exact results all pipelines could have

been used to construct a confidence interval of this metric. Also more iterations could have been run for each pipeline. Therefore, for further research, more computing power could be used to both include more computing-heavy machine learning methods and to improve the standard error estimation.

Another reason for using more computing power is not only the possibility of including more methods for each of the three pre-processing and data engineering categories used in this study, but it could also be used to include more steps. As an example, there exists multiple data imputation and normalization/standardization techniques, both parametric and non-parametric ones. To connect this to the set theory discussion, this would then give a larger subset of all possible pipelines, Ω'', so that $\Omega' \subset \Omega'' \subset \Omega$. This could then expand the discussion by seeing if the probability of nonstandard errors existing increases or decreases with even more pre-processing and data engineering steps included. In connection to this is the fact that table 5 shows that there are many pipelines that gets an equal score. The reason for this is that the test set consisted of only 132 minority class (bankruptcy) samples, leading to an equal amount of correctly predicted default for many pipelines as all had high recall. Therefore, even more samples could be included by, for instance allowing larger time spans.

7.4 Conclusion

To conclude, this study had the aim to analyze the risk of nonstandard errors affecting bank default prediction results when utilizing machine learning. Quantitatively the nonstandard error was largest for the precision metric, where it was larger than the normal error. For recall and AUC it was found to be smaller than the normal error. It has been shown that, even though the nonstandard error estimation for both the recall and AUC metrics is lower than the estimated standard error, there still seem to be a risk of nonstandard errors existing. This conclusion stems from the size of the errors and an analysis of the ranking of the different pipelines for the three scoring metrics. The study has shown that for bank default prediction using machine learning, as many steps in the pre-processing and data engineering as possible need to compare methods to reduce the nonstandard errors as this would improve reproducibility and decrease the overall errors in empiric research.

References

Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, *vol. 6*(no. 1), pp. 20–29. https://doi.org/https://doi.org/10.1145/1007730.1007735

Battiti, R. (1994). Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on neural networks*, *vol. 5*(no. 4), pp. 537–550. https://doi.org/10.1109/72.298224

Breiman, L. (2001). Random forests. *Machine learning*, *vol. 45*, pp. 5–32. https://doi.org/https://doi.org/10.1023/A:1010933404324

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *vol. 16*, pp. 321–357. https://doi.org/https://doi.org/10.1613/jair.953

Chen, C., Zhang, Q., Yu, B., Yu, Z., Lawrence, P. J., Ma, Q., & Zhang, Y. (2020). Improving protein-protein interactions prediction accuracy using xgboost feature selection and stacked ensemble classifier. *Computers in biology and medicine*, *vol. 123*, pp. 103899. https://doi.org/https://doi.org/10.1016/j.compbiomed.2020.103899

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794. https://doi.org/10.1145/2939672

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231.

Faris, H., Abukhurma, R., Almanaseer, W., Saadeh, M., Mora, A. M., Castillo, P. A., & Aljarah, I. (2020). Improving financial bankruptcy prediction in a highly imbalanced class distribution using oversampling and ensemble learning: A case from the spanish market. *Progress in Artificial Intelligence*, *vol. 9*, pp. 31–53. https://doi.org/https://doi.org/10.1007/s13748-019-00197-9

Fiordelisi, F., & Marques-Ibanez, D. (2013). Is bank default risk systematic? *Journal of Banking & Finance*, *vol. 37*(no. 6), pp. 2000–2010. https://doi.org/https://doi.org/10.1016/j.jbankfin.2013.01.004

Garcia, J. (2022). Bankruptcy prediction using synthetic sampling. *Machine Learning with Applications*, *vol. 9*, pp. 100343. https://doi.org/https://doi.org/10.1016/j.mlwa.2022.100343

GitHub, Inc. (n.d.). GitHub Copilot [Accessed 2024-05-13].

Han, H., Wang, W.-Y., & Mao, B.-H. (2005). Borderline-smote: A new over-sampling method in imbalanced data sets learning. *International conference on intelligent computing*, pp. 878–887.

Hardarson, B., & Vuono, M. (2019). Predicting bank insolvency with random forest classification.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1322–1328. https://doi.org/10.1109/IJCNN.2008.4633969

Kotb, M. H., & Ming, R. (2021). Comparing smote family techniques in predicting insurance premium defaulting using machine learning models. *International Journal of Advanced Computer Science and Applications*, *vol. 12*(no. 9). https://doi.org/10.14569/IJACSA.2021.0120970

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. *2008 eighth ieee international conference on data mining*, pp. 413–422. https://doi.org/10.1109/ICDM.2008.17

Menkveld, A. J., Dreber, A., Holzmeister, F., Huber, J., Johannesson, M., Kirchler, M., Neusüss, S., Razen, M., Weitzel, U., Abad-Díaz, D., et al. (2024). Nonstandard errors. *The Journal of Finance*, *vol. 79*(no. 3), pp. 2339–2390. https://doi.org/https://doi.org/10.1111/jofi.13337

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

Petropoulos, A., Siakoulis, V., Stavroulakis, E., & Vlachogiannakis, N. E. (2020). Predicting bank insolvencies using machine learning techniques. *International Journal of Forecasting*, *vol. 36*(no. 3), pp. 1092–1113. https://doi.org/https://doi.org/10.1016/j.ijforecast.2019.11.005

Popescu, M.-C., Balas, V. E., Perescu-Popescu, L., & Mastorakis, N. (2009). Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, *vol. 8*(no. 7), pp. 579–588.

Smiti, A. (2020). A critical overview of outlier detection methods. *Computer Science Review*, *vol. 38*, pp. 100306. https://doi.org/10.1016/j.cosrev.2020.100306

Werner de Vargas, V., Schneider Aranda, J. A., dos Santos Costa, R., da Silva Pereira, P. R., & Victória Barbosa, J. L. (2023). Imbalanced data preprocessing techniques for machine learning: A systematic mapping study. *Knowledge and Information Systems*, *vol. 65*(no. 1), pp. 31–57. https://doi.org/https://doi.org/10.1007/s10115-022-01772-8

Zhou, L. (2013). Performance of corporate bankruptcy prediction models on imbalanced dataset: The effect of sampling methods. *Knowledge-Based Systems*, *vol. 41*, pp. 16–25. https://doi.org/https://doi.org/10.1016/j.knosys.2012.12.007