National College of Ireland

Improving Credit Default Prediction Using Explainable AI

MSc Research Project
Data Analytics

Ciaran Egan
Student ID: x19163568

School of Computing
National College of Ireland

Supervisor: Dr. Majid Latifi

Improving Credit Default Prediction Using Explainable AI

Ciaran Egan
x19163568

Abstract

Despite recent improvements in machine-learning prediction methods, the methods used by most lenders to predict credit defaults have not changed. This is because most of the high-performing methods are of a black-box nature. It is a requirement that credit default prediction models be explainable. This research creates credit default prediction models using tree-based ensemble methods. It is shown that model performance can be improved by using gradient boosting methods over traditional credit default predictions models. The top performing XGBoost model is then taken and made explainable. This research proposes a model-agnostic counterfactual extraction algorithm that explains the drivers behind a particular prediction. The algorithm focuses on extracting the counterfactuals that have the fewest contrasting features. This results in counterfactuals that are easily understood by humans and can be easily translated into insights that the lay user can understand. A definite standard of explainability is defined and the counterfactual extraction algorithm results in explanations that meet this standard. Given that the explanation method is model agnostic, it can be used on any prediction model and can be deployed for a wide range of applications.

1 Introduction

The most significant risk that banks and lenders are exposed to is a large number of borrowers not meeting their loan repayment obligations. This is known as credit default risk. A lenders income stream is dependent on borrowers repaying their loans. This means that a large increase in credit defaults may bring about expenses that the lender cannot absorb and the lender may become insolvent. Many banks meet the "too big to fail" criteria whereby the failure of a large bank can cause widespread economic adversity. In the previous global financial crisis banks faced increases in credit defaults so severe, that the governing bodies of many major economies had to implement expensive bailouts to avoid the aforementioned adverse economic outcomes that would be brought about by the failure of a large bank. Many banks and lenders are legally obligated to create models that predict credit defaults. These models advise banks capital requirements which ensure that the bank can absorb losses brought about by a significant increase in credit defaults. Given the potential economic adversity brought about by a banks inability to absorb such losses, these models are heavily regulated and scrutinised. Although this highlights the need for credit default prediction models to perform well, model performance alone is not the target of these models. Interpretability (or explainablilty) being a key requirement.

Transparency and explainability is a necessity when deploying credit default prediction models in practice. In other words, the reasons behind the predictions of these models must be adequately understood by a lay user. Along with predicting regulatory capital requirements, these models decide whether a potential borrower is given credit. For decades, the right to explanation has been legally guaranteed in most legal jurisdictions. This not only underscores the need for model explainability, but also that the explanations derived must be transparent and accessible. This interpretability requirement means that the model of choice for many bank's credit default models is Logistic Regression. Logistic Regression is a linear model that is high-performing and explainable. There have been advances in Machine Learning (ML) techniques such that models now exist, that typically outperform Logistic Regression. These models have yet to be deployed because of their opaque (or "Black-Box") nature.

This research asks the following question:

> *Can credit default prediction be improved using explainable AI and can the predictions of the models be adequately explained?*

This research takes an arbitrary dataset and builds a credit default prediction model using high-performance black-box methods. Once an optimal model is derived, the model's predictions are made transparent to the point where a defined minimum standard of explainability is met. The derived model is then compared to a traditional Logistic Regression-based credit default prediction model that is modelled on the same dataset. The objective of this research is for the ML method to outperform the traditional credit default prediction method while upholding a minimum standard of explainability.

The following document describes the process undertaken to build a credit default model and make it explainable. The literature review in section 2 reviews other attempts to improve credit default prediction and highlights why these models must be explainable. The literature review section also looks at other attempts to make credit default prediction models explainable and highlights why they do not meet the explainability standard required in an industrial setting. The methodology section 3 commences where the proposed model prediction and explainability methods are explored. The methodology section is followed by section 4 where a practical approach to building the models and explaining them is documented. The documentation of the model build, explanation process and the challenges faced in each process is discussed in section 5. This is followed by the evaluation section 6 where the performance and transparency of the finalized solution is explored. The final section 7 concludes by determining whether the research objectives are met and how this will affect future work on the topic.

2 A Review of the Related Work

The following literature review explores other attempts to improve credit default prediction and shows why these models must be explainable. Firstly, the research showing how credit default prediction can be improved is discussed. This is followed by looking at historical research demonstrating why such models must be explainable. This is then followed by looking at other attempts to make credit default models explainable.

Given that banking portfolios can be worth billions, even marginal improvements can be considered significant. This means that the pursuit to improve model performance in credit default prediction is a worthwhile one. Previous research has shown that credit default prediction can be implemented successfully using less interpretable machine learning

methods. Most of this research focuses on corporate bankruptcy prediction. For example, Moscatelli et al. (2020), Barboza et al. (2017) and Guégan and Hassani (2018) all show that tree-based methods such as Random Forest and Gradient boosted trees (XGBoost, LightGBM) outperform Logistic Regression as a corporate bankruptcy prediction tool. Barboza et al. (2017) and Guégan and Hassani (2018) attempts methods such as Support Vector Machines (SVM) and Artificial Neural Networks (ANN) but the performance gains seen from these compared with a Logistic Regression is either; less conclusive (Barboza et al.; 2017) or significantly worse (Guégan and Hassani; 2018). Fitzpatrick and Mues (2016) used machine learning methods for predicting mortgage defaults and found that tree-based methods outperform Logistic Regression. This shows that using less interpretable machine learning methods (tree-based methods in particular) can result in valuable performance improvements. As stated in Moscatelli et al. (2020) tree-based methods are better able to pick up non-linear relationships that a Logistic Regression typically does not. Qiu et al. (2019) found that the Random Forest model did not conclusively outperform the Logistic Regression but another tree-based ensemble model, the LightGBM did. There are certain cases where the model choice is not a factor at all in model performance such as in Chen et al. (2021) where none of the models conclusively outperform another. It is worth noting that Chen et al. (2021) used a very rigorous evaluation procedure where the out-of-time testing was performed, and the performance was scored based on a varying risk appetite.

Model performance is not the sole requirement for credit default prediction models. Credit default prediction models must be transparent and explainable. Credit default prediction models are often used to automate decisions on loan applications making them sensitive by nature. Onay and Öztür (2018) mentions the legislation in the US (Fair and Accurate Credit Transactions Act, US Fair Credit Reporting Act and Equal Credit Opportunity Act) that uphold the rights for customers to know why their loan applications are declined. In the EU, the General Data Protection Regulation (GDPR) grants a right to explanation (Wachter et al.; 2018). There exist ethical reasons why these credit default models must be explainable. Explainability can highlight any discriminatory bias the model may have. Munnell et al. (1996) found that racial discrimination in lending application success exists. This can have wider socioeconomic effects which might increase racial wealth inequality. This was found to be the case in Charles and Hurst (2002) where they propose that one of the factors influencing wealth inequality between Blacks and Whites in the US may be varying loan application success rates between the races.

This all stresses the need for model explainability and transparency. For the purposes of this research, a minimum standard of model explainability must be defined. A report on big data produced by the European Banking Authority (EBA) (European Banking Authority; 2020) describes how a model is explainable if the following conditions are met:

1. How the result is reached is understandable by humans.

2. Justifications for the main factors that led to the output must be provided and these must be understood by humans.

This defines a clear minimum standard of explainability required by the models in order to be considered explainable and is the minimum standard required by the models derived by this research.

The previous research mentioned focus almost exclusively on model performance. Explainability if addressed at all, was not a primary concern. This means the models

created by the research would not be deployed in an industrial setting. There are cases where model interpretability is mentioned such as in Fitzpatrick and Mues (2016) who acknowledge the Logistic Regression's widespread use in industry due to its interpretable nature. Moscatelli et al. (2020) and Fitzpatrick and Mues (2016) derive feature importance measures which fail to the meet the standards set in the previous paragraph as they do not adequately justify the model prediction. Chen et al. (2021) state that the goal of ML models is purely prediction and that explanation of the relationships between the dependent and independent variables is the goal of statistical models such as Logistic Regression. It may be true that ML methods have little or no obligation to be explainable, but this does not mean that the performance gains from these models cannot be utilized while simultaneously upholding a defined standard of explainability.

There have been other attempts to make "black-box" credit default prediction models explainable. The research is quite sparse with most papers being written within the past two years. Arguably, the most popular method of making such black-box models explainable is using SHapley Additive exPlanations (SHAP) (Lundberg and Lee; 2017). SHAP gives feature importance measures that aid the user in understanding the features that influence the model's prediction. SHAP is a model agnostic method which means that it is implemented in parallel with the prediction algorithm. Using model agnostic methods gives the researcher more freedom when it comes to model choice and for this reason model agnostic methods are used in this research. Bussmann et al. (2020) used SHAP values as a tool to explain a black-box credit default prediction model. Bussmann et al. (2020) compares the model performance of an XGBoost model to a Logistic Regression. Similar to the research discussed in first paragraph of this literature review, the less interpretable XGBoost model outperforms the Logistic Regression. Although the methods used in Bussmann et al. (2020) make the reasons behind the model predictions clearer, it does not meet the standard set by European Banking Authority (2020). Insufficient justification is given behind the motivation behind the models prediction. The feature importance measures are ambiguous. If the model is used to decide the outcome of loan applications is used, the customer would not have sufficient information to improve their creditworthiness.

Example rule-based extraction is another method used to justify black-box model predictions. Islam et al. (2021) state that these rules can be easily understood by humans. Keane and Smyth (2020) state that counterfactuals in particular, satisfy GDPR's right to explanation. An example of using rule extraction to explain a black-box model (Random Forest) is in Prentzas et al. (2019). They use a case-based reasoning rule extraction method in parallel to the Random Forest model. This method is model agnostic and, although their model is not used for credit default prediction but the insights are directly applicable. Fernandez et al. (2020) compares the counterfactual approach to the feature importance methods. They highlight the limitations in using feature importance to explain credit default predictions such as how the most important features may not be the ones that make the crucial decision. They conclude by stating that feature importance methods can be ambiguous and often misleading, stating that using counterfactuals is a superior approach as a result.

All the academic papers reviewed can be summarized in Table 1:

Table 1: Summary of literature review

Paper	**Methodology and Domain**	**Insights**	**Shortcomings**
Moscatelli et al. (2020)	Optimize model performance for corporate bankruptcy prediction.	Tree-based ensemble models outperform Logistic Regression.	No attention paid to model explainability. Although Feature importance measures are derived, they are not explored in detail.
Barboza et al. (2017)	Optimize model performance for corporate bankruptcy prediction.	Tree-based ensemble models outperform Logistic Regression. Performance gains from neural networks and SVM compared with Logistic Regression are less conclusive	No attention was paid to model explainability.
Guégan and Hassani (2018)	Apply machine learning methods to a credit scoring problem.	Tree-based ensemble models outperform Logistic Regression. SVM and neural network models significantly underperform Logistic Regression.	Paper ignores explainability.
Fitzpatrick and Mues (2016)	Mortgage default prediction using ML methods.	Tree-based ensemble models outperform Logistic Regression.	No attempt to make models explainable but does acknowledge the Logistic Regression's widespread use due to its interpretable nature.
Qiu et al. (2019)	Using ML methods to predict credit defaults on the Kaggle Home Credit dataset.	LightGBM outperforms Logistic Regression but Random Forest does not.	No attempt to make model explainable.
Chen et al. (2021)	Predicting mortgage delinquency using ML methods	Model choice does not play a part in model performance. Very rigourous evaluation methods that are worth considering.	Does not attempt to make models explainable. States that ML methods have no obligation to be explainable.
Onay and Öztür (2018)	A literature review to address the challenges presented by big data in credit scoring.	Addresses the regulatory challenges big data presents in credit default prediction.	Paper is a pure literature review, and no data analysis or modelling is performed.
Wachter et al. (2018)	Using counterfactuals to satisfy model explainability challenges in credit default prediction	Highlights the US regulation that requires credit default prediction models to be explainable.	Lack of empirical examples on real-world data.

Munnell et al. (1996)	Highlights racial inequality in lending in Boston in the mid-1900s	Shows the ethical issues surrounding loan application decisions	Given publication date, insights although relevant, may be out of date.
Charles and Hurst (2002)	Investigates the influence of unequal lending decisions on the wealth gap between blacks and whites in the US	Highlights the societal importance of transparent credit decisions	Given publication date, insights although relevant, may be out of date.
European Banking Authority (2020)	Investigates the challenges faced by banks in implementing big-data analytical techniques.	Defines a clear standard of model explainabilty.	Lack of empirical examples.
Lundberg and Lee (2017)	Provides a method to interpret black-box machine learning models.	Method is useful and hence widely used	Empirical examples shown deal with deep learning and are not directly applicable to credit default prediction
Bussmann et al. (2020)	Tries to make a credit default prediction model explainable using SHAP.	Provides an effective method to explain a black-box credit default prediction model. Compares the model to a traditional Logistic Regression model.	Arguably fails to meet the minimum standard of explainability described in European Banking Authority (2020).
Islam et al. (2021)	Surveys a wide range possible approaches to make a credit default prediction model explainable	Each approach is explored in detail.	Lack of empiricism. Paper concludes with stating that traditional methods are more effective.
Keane and Smyth (2020)	Uses counterfactuals to explain a black-box ML model.	Describes how to find good counterfactuals. States that good counterfactuals satisfy GDPR's right to explanation	Concludes by saying that good counterfactuals are sparse.
Prentzas et al. (2019)	Uses rule extraction to explain a Random Forest model	Explains the Random Forest model very effectively	Research focused on application different to credit default prediction.
Fernandez et al. (2020)	Compares using counterfactuals to feature importance measures as a method to explain model predictions	Shows why counterfactuals are superior to feature importance measures for explaining model predictions	None relevant to this research

The literature review done sets out the objectives for this research clearly. As shown, credit default prediction model performance can be improved by using black-box ML prediction methods, with tree-based ensemble models being the best performing on aggregate. A definite explainability standard was defined (European Banking Authority; 2020). This sets a clear definition of what makes a model explainable. The literature shows that counterfactuals are the best way to make a black-box model explainable.

Counterfactual examples satisfy GDPR's right to explanation (Keane and Smyth; 2020). The counterfactuals show what features need to be changed to achieve a compatible result thus highlighting the drivers behind the model prediction and hence justifying the model prediction. This satisfies the explainability standard set by European Banking Authority (2020). It is for these reasons that counterfactuals will be used as the novel approach for this research.

3 Methodology

3.1 The Underlying Data

The data required must satisfy a number of requirements. Predicting credit defaults is a binary classification problem. The target variable will represent a binary outcome that indicates the presence or absence of a loan default. Most banks look at other financial data linked to the customers such as their current account spending, credit card data and/or payment profiles with respect to other loans they might have. Finding this data can be difficult given that financial data is sensitive and confidential.

The dataset chosen for this project is the Kaggle Home Credit dataset [1]. This was offered as a Kaggle competition in May 2018. Home Credit is an international non-bank financial institution that specializes in lending to people with little or no credit history. The task set by the Kaggle competition was to correctly classify a binary outcome indicating the presence or absence of default. Because the dataset was offered as part of a Kaggle competition, the data has been widely explored and many high-performing models have been created. The sole goal of the Kaggle competition is to optimize model performance, with the AUC being the target performance metric. Model transparency was not a requirement for the competition and, little or no attention was paid to model transparency. The expectation of this research was not to outperform the leading Kaggle submission but to build on the work done by the competitors by addressing model transparency.

The data satisfies the requirements for this project. The target variable is binary indicating the presence or absence of a loan default. There is surrogate data that can be linked to each loan such as the customers credit history, instalment data on other loans and external data from the Credit Bureau. This surrogate data matches what a bank would typically have when building a credit default prediction model. Given that the source of this data is Kaggle, this data is open and publicly available.

3.2 Data Exploration

3.2.1 Data Structure

The dataset comes as a set of CSVs[2]. Given that the data contains different tables linked by different keys, an ideal way to store this was by using a Postgres relational database[3]. The database diagram is found on the Kaggle website[4]. The use of a Postgres SQL database allows the easy querying and manipulation of data. An Analytical Base Table (ABT) was created using this database and the SQL interface in Section 5.2.

[1] `https://www.kaggle.com/c/home-credit-default-risk`

[2] `https://www.kaggle.com/c/home-credit-default-risk/data`

[3] Data Pipeline: `https://github.com/i-am-yohan/home_credit_2_postgres`

[4] Data Information: `https://www.kaggle.com/c/home-credit-default-risk/data`

The application train/test dataset is dimensioned using `SK_ID_CURR`. Other unique identifiers include `SK_ID_PREV` and `SK_ID_BUREAU`. The `SK_ID_CURR` variable has a one-to-many relationship with the other unique identifiers. To create meaningful features, the features engineered from tables at different levels to `SK_ID_CURR` are aggregated to `SK_ID_CURR`. SQL makes this straightforward using the `Group By` command. The feature engineering is further discussed in Section 5.2.

3.2.2 Train and Test Splitting

The Kaggle Home Credit data comes with both and train and test set. The test dataset does not contain target labels and the performance score is obtained by submitting the predictions to the Kaggle website. The only performance score output from submitting a prediction is the AUC. Although this will be used to evaluate model performance, the AUC alone is not sufficient to determine if the model meets the required performance criteria. Measures such as accuracy, precision and recall are also required (Section 3.6.1). The training set was partitioned into a training and hold-out (test) sample at an 80:20 ratio respectively. This allows the calculation of a wider range of model evaluation scores which will be used in tandem with the Kaggle submission score(s).

3.2.3 Class Imbalance

The class imbalance in the target variable is significant. Defaulted cases make up approximately 8% of the overall population. This results in the trained model underclassifying the minority cases. Class imbalance is typically dealt with using sampling techniques and/or class weightings. Optimizing performance with respect to sampling methods is out of scope for this research and as such the decision was taken to use class weightings to deal with class imbalance. Class weightings penalize the model's loss function disproportionately to balance and hence unbias the predictions. This is convenient as no sampling was required and does not typically cause the model to underperform.

3.3 Model Explanation Approach

As previously discussed, the model must be explainable for it considered for industrial use. Trivially the traditional credit scorecard model is explainable by nature. There is no need to modify the model to make it explainable. The machine learning models are not explainable and extra steps were taken to make these models explainable (discussed further in section 3.4). As discussed in the literature review section 2, extracting counterfactual predictions meet these criteria. Taking the hypothetical counterfactual: *"Customer A could achieve a credit score of Y if they increase their Loan-to-Value to X_1 and reduce their proportion of late credit card payments to X_2 like that of customer B"*. This explains why the customer received the credit score they got and it also shows the features that caused the model to make the prediction it made. Counterfactuals must be simple for them to be understood easily by lay humans. Therefore, the target counterfactual will be the one that requires the adjustment of the smallest possible number of features. This will require finding the counterfactual that has a very similar profile to the case that requires explanation. This is known as the nearest-unlike-neighbour (NUN) as discussed in Keane and Smyth (2020). The approach to finding the adequate NUN is documented in Section 4.2.

Finding the counterfactual with the smallest distance is not sufficient to explain the model outcome. The counterfactual values should be applied to the case that requires explanation and this should result in the desired prediction or better. Using the hypothetical example in the last paragraph, if by changing the Loan-to-value and number of late credit card payments for customer A to that of customer B does not result in the desired credit score or better, then the counterfactual is invalid. This was assessed when searching for the counterfactuals.

The criteria for finding adequate counterfactuals is summarized in the below:

1. Must be the smallest possible.

2. The adjustments to the parameters because of the change in feature set must result in the desired predicted outcome or better.

This is the stopping criteria for the algorithm discussed in Section 4.3.

3.4 Overall Research and Model Build Approach

The research and model build approach is highlighted in Figure 1.

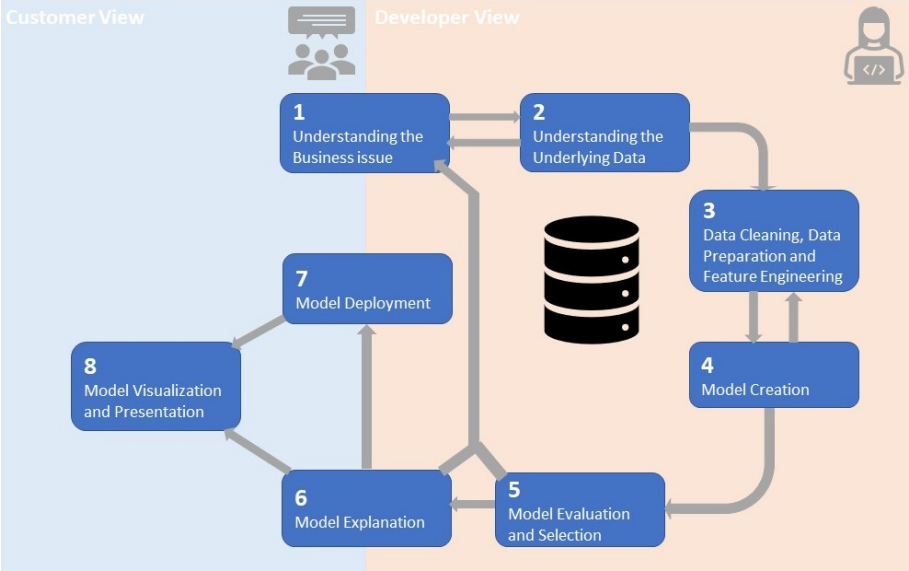

Figure 1: Research Approach

The approach taken is based off CRISP-DM[5] with extra steps added for the model explanation component of this project. Given that the counterfactual explanation approach is model agnostic, it will only need to be implemented on one model. It is because of this that step 5 also contains a model selection component where the best candidate is chosen to move to the next stage. The stage after evaluation is the explanation phase where the counterfactuals will be extracted. Simultaneously with model deployment, the predictions of each model must be visualized so that they can be digestible to potential borrower and other non-technical audiences. This is the research and model build approach taken for this research project.

[5]CRISP-DM: `https://www.datascience-pm.com/crisp-dm-2/`

3.5 Prediction Models Used

These are the learning algorithms built in the model creation phase. The models created are as follows:

- Traditional Credit Default Risk Prediction Model
- Random Forest
- Gradient Boosted Trees - Light Gradient Boosting Model (LightGBM)
- Gradient Boosted Trees - Extreme Gradient Boosting Model (XGBoost)

As discussed in Section 2, tree-based models tend to be the highest performing models when predicting credit defaults. These are the model prediction algorithms used in this research. As previously discussed, the Traditional Credit Default Risk Prediction Model will act as the control group.

3.5.1 Traditional Credit Default Risk Prediction Model

Historically Logistic Regression is the main tool of choice for credit default prediction[6]. Typical credit default models features $x_n = (x_{i,1}, \ldots, x_{i,j})$ are binned. The Weights of Evidence (WoE) for each of those bins are calculated as follows:

$$WoE = ln\left(\frac{\%Goods}{\%Bads}\right) \tag{1}$$

The WoE gives the predictive power of the independent variable in relation to the dependent variable. This gives the developer a method of handling outliers, non-linearity, feature scaling and missing values. The predictive power of each bucket or feature is the Information Value (IV). It is calculated as follows:

$$IV = ln\left(\%Goods - \%Bads\right) * WoE \tag{2}$$

A Logistic Regression is trained on the best found subset of features binned by the WoE. Chen et al. (2020) states how the IV is the main risk measurement tool for the risk rating model and they define the risk rating model as the international mainstream risk model.

This research created a credit default prediction model using the methods mentioned in the previous paragraphs. To calculate WoE and IV values, the R package `scorecard` [7] was utilised. The `scorecard` implements optimal binning, WoE calculation and IV calculation as documented in both Siddiqi (2005) and Refaat (2011). This model will act as a control group and the ML-based model must outperform this while upholding the minimum transparency criteria.

3.5.2 Random Forest

Random Forest is an ensemble method that is commonly used for classification. The model is an ensemble of Decision Trees. A Decision Tree is a set of *if-then*-like logical statements used to split into specified classification buckets. Training the Decision Tree

[6] `https://www.accenture.com/nl-en/blogs/insights/the-future-of-default-prediction-a-comparison-of-machine-learning-model-performance`

[7] Scorecard R package `https://CRAN.R-project.org/package=scorecard`

involves deriving the optimal decision rules that best classify the example. Decision Trees are explainable and transparent and are sometimes considered suitable for deployment in industry. The main disadvantage is that Decision Trees suffer from over-fitting and bias. In general, Decision Trees, because of their simplistic nature, are not considered competitive with Logistic Regression and were excluded from the current research because of this. Random Forests are proposed as a method to remedy this disadvantage Breiman (2001). Random forests are an ensemble of Decision Trees where each Decision Tree is grown independently, and each individual Decision Tree has an equal vote as to what the outcome is. This is done at the expense of interpretability, but as shown in Section 2 they can be a powerful classification algorithm.

3.5.3 Gradient Boosted Trees

Like the Random Forest algorithm, Gradient Boosted Trees are an ensemble algorithm consisting of (typically) Decision Trees. The main difference is how the trees are built and how the overall prediction is made. Boosted trees are built sequentially where each tree tries to correct the errors of the previous tree. Unlike the Random Forest where each tree has an equal vote, the classification is a weighted vote of each tree's prediction. The gradient boosting algorithms chosen for this research project are XGBoost (Chen and Guestrin; 2016) and LightGBM (Ke et al.; 2017). As shown in Section 2 they can typically outperform the Logistic Regression. The main difference between the two algorithms is how each individual tree is grown. In XGBoost the tree is grown *level-wise* while in LightGBM each tree is grown *leaf-wise*.

3.6 Model Evaluation Approach

Optimizing model performance while upholding a clearly defined standard of explainability is the primary goal of this research project. The models are evaluated under the headings of *Model Performance* and *Model Explainability*. The following subsections will detail the criteria that the derived model must meet for it to be suitable for deployment.

3.6.1 Evaluation of Model Performance

In the context of this research, model performance is defined as the model's ability to correctly predict potential defaults. This does not encapsulate the model's transparency and the model performance measures discussed in this section do not account for the model's transparency. To assess the performance of the model the following performance measures were derived:

- Accuracy - The proportion of correct classifications in the evaluation data.
- Precision - The proportion of true positives among the predicted positives.
- Recall - The proportion of positives correctly predicted.
- Area Under receiver operating Curve (AUC) - The Receiver Operating Curve (ROC) measures the model's classification ability subject to varying decision boundary thresholds. The ROC plots the true-positive rate to the false-positive rate. The area under the curve (AUC) aggregates the performance measures given by the ROC curve.

The class imbalance (discussed in Section 3.2.3) is considered when considering what performance measures must be favoured. Given that the proportion of defaults tends to outweigh non-defaults, accuracy alone is not a sufficient measure to assess model performance. This is related to the misclassification cost imbalance seen with loan defaults. A false negative is significantly more costly than a false positive. Therefore, Recall and AUC measured are favoured for this research project. The precision and accuracy measures are only considered in the model selection phase when the Recall and AUC measures do not yield a conclusive optimal model.

To identify bias and/or over-fitting, K-fold cross validation is implemented. If the performance metrics are stable for each fold, then for the purposes of this research, the model was considered free of bias and overfitting.

The evaluation metrics are calculated for both the traditional credit default prediction model and the candidate ML-models. As part of the goal of this research, it is a requirement for at least one of the ML models to outperform the traditional credit default prediction model. This is because the candidate ML models are of a Black-box nature while the traditional credit scorecard model is by its nature explainable. Black-box models for credit default prediction must offer performance gains over traditional methods to justify their use and/or the resources used to make them explainable.

3.6.2 Evaluation of Model Explainabilty

Explainability is subjective and cannot be quantified using a numeric measure. It is because of this that a definite standard of explainability was defined in Section 2. The transparency goal of the resulting solution will be to meet this standard. If the model cannot be adequately explained to this standard, then it will not be considered for deployment.

4 Design Specification

4.1 Model Building

In this section the algorithms employed to derive an optimal model are described. During the model build, the model is validated using a hold-out sample taken as a proportion of the training data. This is to test the performance variance brought about by hyperparameter changes, feature selection etc. Given that time and computational resources were limited, methods such as grid search and random search are not used to optimize the chosen feature set and hyperparameters. Once the models were built, each model was validated using K-fold cross validation. The model building process by model created is described in the following sections.

4.1.1 Traditional Credit Default Risk Prediction Model

Before any model training takes place, the variables must be binned and the WoE transformation for each feature must be calculated. This can be done automatically using the `scorecard` package in R (see Section 3.5.1). The WoE bins are adjusted if required. Once the variables are preprocessed, the model building process can begin.

Regularization is not commonly used when building credit scorecards in industry. Instead, overfitting is dealt with by using dimensional reduction. In this case, backward

stepwise selection is implemented. Before any model training is done, variables with poor IVs are dropped as they will likely have no predictive power. The Logistic Regression model was then trained on all remaining features using R's built-in `glm` function. The model is tested for multicollinearity using the Variance Inflation Factor (VIF) measure. The variable with the largest VIF is dropped and this process continues until all features are below the required VIF threshold. Once that is completed all insignificant variables are dropped as these add noise. The WoE binning calculation means that in isolation, every variable must have a positive relationship with loan default. Any variables with a negative relationship with the target variable are dropped so that when the scorecard is created, no variable will give negative scores. If overfitting and bias still exist, then variables will be removed iteratively by selecting the lowest IV. This is done until overfitting and bias does not exist. These are the steps undertaken to ensure a robust credit scorecard model is created.

4.1.2 Tree-Based Ensemble Models

All of the black-box models considered for this research project are tree-based ensemble models and therefore the build process for each model was largely the same. For the Random Forest model, the `RandomForestClassifier` function in the `scikit-learn` python package was used while the LightGBM and XGBoost models were built using their namesake python packages. The complexity of the trees was the main parameter in adjusting for over/underfitting. For the Random Forest model, the complexity of each tree was adjusted using the minimum impurity decrease. For LightGBM model, the max number of leaves was used to adjust the complexity of each tree while the XGBoost was adjusted using the max number of levels for each tree. This is intuitive because LightGBM and XGBoost build trees leaf-wise and level-wise respectively. The number of trees was adjusted iteratively. For the Random Forest model, extra trees are used to remedy overfitting but this can have diminishing returns as the number of trees get large. In this case, the number of trees was increased until the model performance converged. For XGBoost and LightGBM, additional trees have the potential to cause overfitting, so the number of trees was increased until the AUC in the training and validation dataset diverged. For each model, the feature importance measures are used for feature selection.

4.2 Finding the Nearest-Unlike-Neighbour Counterfactual

Given that the target is to find the counterfactuals with the smallest number of features to vary, the NUN will be found by finding the example with smallest Euclidian distance with 1 or more features removed. This is calculated as follows:

$$NUN(\{x_1, \ldots, x_k\}) = argmin_j \left(\sum_{\substack{i=0 \\ i \setminus \{x_1, \ldots, x_k\}}}^{n-k} |T_i - CF_{i,j}| \right) \tag{3}$$

where:

- T_i - Standarized feature i for the case of interest T
- $CF_{i,j}$ - Standarized feature i for potential counterfactual j
- $\{x_1, \ldots, x_k\}$ - The list of excluded features.

All potential counterfactuals $CF_{i,1}, \ldots, CF_{i,n}$ are selected due to user specified criteria. For example, if the user wants to see what features to adjust to achieve a score of Y or higher then $CF_{i,1}, \ldots, CF_{i,n}$ will be all cases with a predicted credit score of Y or higher.

The counterfactuals found must satisfy the criteria set in Section 3.3. If the changes to features $\{x_1, \ldots, x_k\}$ are applied to T_i as they are in $CF_{optimal}$, then this must yield a credit score that meets the specified criteria. If changing $\{x_1, \ldots, x_k\}$ in T_i to what they are in $CF_{optimal}$ does not result in the desired credit score or better, then the counterfactual is not valid. The feature set $\{x_1, \ldots, x_k\}$ must be small. The search for the optimal feature set(s) $\{x_1, \ldots, x_k\}$ is quite exhaustive given that a model with m features will result in $m!$ feature combinations. This means that in addition to the criteria set in Section 3.3, the algorithm must be scalable.

4.3 Finding the Desired Counterfactual Set

An algorithm to find the desired feature set(s) $\{x_1, \ldots, x_k\}$ to exclude when looking for the NUN counterfactual that satisfy the requirements set out in section 4.2 must be derived. The algorithm to find the optimal feature set(s) $\{x_1, \ldots, x_k\}$ operates using the following step-by-step procedure:

1. Take an observation that requires explanation.
2. Find the NUN for each input feature in isolation.
3. For each NUN counterfactual found, for each respective feature/feature set, replace the value in the case of interest with the value of that of the counterfactual and predict each outcome.
4. If at least one of the predictions meets the specified criteria, then output the respective counterfactuals and terminate the algorithm. If none of the predictions meet the specified criteria, then continue to the next step.
5. Take a sample of the predictions that are closest to the required criteria. The sample size is set by the user.
6. For each feature relating to each resulting counterfactual, add each other remaining features and calculate the NUN for each.
7. Return to step 3.

The pseudo-code in Algorithm 1 describes the algorithm in detail. Given that n is constant, the algorithm has complexity $\mathcal{O}(n)$. This means that it is scalable. The algorithm is an iterative process and terminates when a counterfactual that satisfies the required criteria is found. The number of features increases by 1 at each iteration and the algorithm terminates when the criteria is reached. It is because of this that the number of features that require change to achieve the desired outcome will likely be the smallest possible.

Algorithm 1 How to find an ideal set of Counterfactuals

Data: Every training and test example with their predicted outcome.

Result: An array of counterfactuals that explain the model's predictions for a particular case c.

Require: any X^* such that $y^* - \alpha < C(X^*) < y^* + \beta$ where α, β are arbitrary tolerances set by the user.

$y = C(X)$: where y is the predicted outcome where X is the input feature set $\{x_i\}$ for all $i \in 1, \ldots, m$;
y^*: the desired predicted outcome;
$y_0 = C(X_0)$: as the initial prediction which requires explanation;
n: a number to sample at each iteration set be the user;
$k := 0$;
if $k := 0$ **then**
  find $NUN(x_i)$ for each $x_i \in \{x_1, \ldots, x_m\}$;
  take $\{\chi_{1,k}, \ldots, \chi_{m,k}\}$ as the feature sets resulting from $NUN(x_i)$ for each $x_i \in \{x_1, \ldots, x_m\}$
  for each feature set $\chi_{i,k}$ for each $NUN(x_i)$ replace the values in X_0 for the feature set $\chi_{i,k}$ with the values in $NUN(x_i)$ resulting in values $\{X'_{1,k}, \ldots, X'_{m,k}\}$;
  if $y^* + \alpha < C(X'_{i,k}) < y^* - \beta$ for any $X'_{i,k} \in \{X'_{1,k}, \ldots, X'_{m,k}\}$ **then**
    return $NUN(\chi_{i,k})$ for which $\chi_{i,k}$ that satisfies $y^* - \alpha < C(x'_{i,k}) < y^* + \beta$;
  end if
end if
while $\neg(y^* - \alpha < C(\chi_{i,j}) < y^* + \beta)$ for any found i, j **do**
  $k := k + 1$;
  find n features sets $\chi_{i,k-1}$ which $C(X'_{1,k-1})$ is closest to y^*;
  for each remaining $\chi_{i,k-1}$, add x_i for each $x_i \in \{x_1, \ldots, x_m\}$ resulting in feature sets $\chi_{i,k}$ for $i = 1, \ldots n * m$;
  find $NUN(\chi_{i,k})$ for all $\chi_{i,k}$ for $i = 1, \ldots, n * m$;
  Calculate $C(X'_{i,k})$ for $i = 1, \ldots n * m$;
  if $y^* - \alpha < C(X'_{i,k}) < y^* + \beta$ for any $X_{i,k}$ **then**
    return $NUN(\chi_{i,k})$ for which $\chi_{i,k}$ that satisfies $y^* - \alpha < C(X'_{i,k}) < y^* + \beta$;
  end if
end while

5 Implementation

This section describes the model build and explanation phases[8].

5.1 Configuration

The process was configured such that all redundant and duplicative data cleaning and preprocessing was minimized. For example, operations such as train/test splitting, outlier removal and feature engineering were done prior to any model train. The data resulting from these operations was loaded to the postgres DB and the models were built using the resulting data. This ensured that each model is built using the same ABT. There are cases where some operations can not be generalized for each model, and these operations

[8]Project Code: `https://github.com/i-am-yohan/explainable_credit_scoring`

were done during the respective model build. Operations such as feature selection and feature scaling were done in each respective model build phase as they vary from model to model.

5.2 Data Cleaning, Feature Engineering and Analytical Base Table Creation

5.2.1 Data Cleaning and Preprocessing

The data contains many missing values to be imputed. The imputation methods vary based on the feature. Features such as `AMT_CREDIT`, `AMT_INCOME_TOTAL`, `AMT_ANNUITY` and `AMT_GOODS_PRICE` are highly correlated so missing values were imputed using a linear regression. The external credit score variables (`EXT_SOURCE_X`) were imputed using their mean values. When joining other tables on to the main `application_train|test`, null values are created when a link does not exist between the base table and the joined table. In the cases where this occurs, the feature will be imputed with a 0 value.

5.2.2 Feature Engineering and Analytical Base Table Creation

Intuitively, financial distress and/or the possibility of incurred risk (colloquially known as "Skin in the game") is what drives loan default. The features created must capture this. The features can be created from the base table or from the surrogate tables or a combination of the surrogate tables. For example, the principal of the loan relative to the value of the underlying asset might be a good measure of the "skin in the game" the customer has in the underlying loan or the number of late payments on other loans might indicate financial distress. Combinations of other features were taken such as the `EXT_SOURCE_X` multiplied by the age of the applicant. Given that the Kaggle competition has finished, features created by other users were considered. An example of this is the `TARGET_NEIGHBORS_500_MEAN` feature created by the winning submission [9]. The feature engineering process resulted in 326 features.

5.3 Model Build

This section will describe the process and challenges faced when building each model.

5.3.1 Parameterization Optimization

Model performance was optimized via feature engineering. Feature engineering delivered the most performance gains relative to time and resources spent. In the early stages of development when the model training failed to result in adequate model performance, the feature engineering phase of development was revisited. New features were created, and this improved model performance. This is consistent with the development approach discussed in section 3.4.

For the credit scorecard model, there were no hyperparameters that were adjusted. Any overfitting was dealt with by removing features until overfitting is no longer observed. This is discussed further in Section 5.3.2. With respect to hyperparameter selection in the tree-based ensemble models, in the early stages of the research, methods such as grid search and random search were explored. These were found to be slow and did not yield

[9] `https://medium.com/thecyphy/home-credit-default-risk-part-2-84b58c1ab9d5`

many significant model performance improvements. Therefore, the hyperparameters were adjusted iteratively. This produced models that outperformed (with the exception of the Random Forest) the control model so further hyperparameter optimization was deemed redundant. For the tree-based ensemble models, feature selection showed a negligible effect on model performance. This is discussed further in Section 5.3.3. This was the approach taken to optimize parameterization.

The build process split by model is described in the following subsections:

5.3.2 Traditional Credit Default Risk Prediction Model

The WoE binning procedure was done using the `scorecard` package. All constant variables that could not be used in this analysis were dropped. Once the WoE values were created, variables with a total IV of less than 5% were dropped. 5% as this removed potentially poor model predictors while still leaving large number of features to tune the model adequately. The model was then trained using the remaining features. To identify multicollinearity, the Variance Inflation Factor (VIF) of each feature was calculated. The feature with the largest VIF was be removed iteratively until all features have a VIF of 5 or lower. The model then was retrained with the new feature set. All insignificant (p-value greater than 5%) features were removed. The model was then retrained and all features with negative relationships with the target variable were removed. The resulting model was adequate and passed the k-fold cross-validation test. This resulted in the candidate credit default model that will be considered for deployment. Given that this is the model typically used in industry, this will serve as the control model which the black-box methods proposed by this research will have to outperform.

5.3.3 Tree-Based Ensemble Models

The features were standardized for all the models. To avoid noise caused by highly correlated variables, when two variables were correlated with a value above 85%, one of the variables was removed. For all models reducing the number of features based on feature importance had a negligible effect on performance so the number of features was reduced until model performance began to decline. This removed redundant features without sacrificing model performance. Each tree model showed overfitting at first so the complexity of the individual trees were reduced. For the Random Forest model, the `min_impurity_decrease` hyperparameter was reduced until no overfitting occurred. For the LightGBM model the maximum number of leaves for each tree was optimised to a value of 4. For the XGBoost model maximum depth parameter was set to 2. All the tree-based ensemble models passed K-fold cross-validation and were brought to the next stage of analysis.

5.4 Model Explanation

5.4.1 Mapping Prediction to Credit Score

Typically, in credit scorecard development, the model prediction output from the model is mapped to a natural number. This is done using the typical Odds-to-Score mapping formulae [10]. For the deployed model, the target score was 600, the target odds was set to

[10]Convert odds to score formulae: `https://rstudio-pubs-static.s3.amazonaws.com/376828_032c59adbc984b0ab892ce0026370352.html`

1 and the points to double the odds was set to 50. Given that the decision boundary for each model prediction is 50%, the required score to be predicted as non-defaulted is 600 or higher. If the deployed model was used to decide the outcome of a loan application, then the required credit score for a successful application is 600 or higher.

5.4.2 Optimal Counterfactual Search

The goal of this research was not to find the optimal counterfactual but rather to make a black-box model explainable. The goal was to find at least one sufficient counterfactual rather than finding the optimal counterfactual. It is because of this that the counterfactual algorithm discussed in Section 4.3 terminates a soon as one or more counterfactuals that meet the specified criteria discussed in Section 3.3 are found. The algorithm favours counterfactuals that involve adjusting the smallest number of features because trivially, these are easiest to explain. Finding the optimal counterfactual explanation by any metric was out of scope for this research.

5.4.3 Nearest-Unlike-Neighbour Search Results

As discussed in section 3.3, finding the NUN does not always result in the desired outcome. Take the example in Table 2:

Table 2: Counterfactual Example

	Counterfactual 0	**Counterfactual 1**
CF account no.	`321072`	`321072`
Feature to adjust	`amt_annuity`	`days_birth`
Case value	33309	-14311
CF value	25447.5	-8930
Score of case	587.64	587.64
Score of CF	643.71	643.71
New predicted score	595.28	629.88

Account number `100177` was predicted to have a score of 587.64. Using the deployed model in an application setting with a decision boundary of 600, this case would be refused credit. In Table 2 the NUN for exclusive of two features was found, `amt_annuity` and `days_birth`. These were found using the formula in section 4.2. This resulted in finding the same counterfactual (abbreviated as CF in Table 2) for both features. Assume the desired score is 600 or above. Changing the value of `amt_annuity` in account number `100177` to that of `321072` will result in a score of 595. This means that the counterfactual is invalid because adjusting the `amt_annuity` to that of the counterfactual will not result in a score of 600 or above. In contrast, changing the value of `days_birth` in account number `100177` to that of `321072` will result in a score of 630 which satisfies the required criteria. If deployed in the algorithm in section 4.3 the algorithm would terminate after the first iteration given that Counterfactual 1 satisfies the required criteria. Counterfactual 1 can be easily translated to the sentence: "Case `100177` has a low credit score because the applicant is too old". The ethical issues surrounding such an explanation are discussed in Section 6.2.

5.4.4 Deployment of Counterfactual Search Algorithm

The algorithm deployed delivers varying counterfactuals depending on the required criteria. Take the case of account `101999`, which has a predicted score of 548. If the target score is set to 600 or higher, the algorithm outputs a single counterfactual suggesting alterations to `target_neighbors_500_mean`. The `target_neighbors_500_mean` feature is a combination of external bureau credit scores and the credit to annuity ratio. This means that the customer has a poor credit rating with the bureau and the loan term must be increased to decrease the credit-annuity ratio. This adequately explains why the model predicted default for this customer. This can be easily translated to actionable insights that can be understood by the lay user. The underlying factors that led to the output are also very clear. Taking `101999` above, and setting a target score of 750, the algorithm outputs three counterfactuals each with three features, indicating three ways the customer can improve their credit score to a value of 750.

Case `131594` with a particularly poor credit rating (475) and is predicted as being highly likely to default. If the algorithm runs with the target score of 600 or higher, it outputs 4 counterfactuals each with three features for two different NUNs. Taking one of the counterfactuals which contains features `avg_bal_limit_ratio`, `ext_source_max` and `target_neighbors_500_mean`. Both `ext_source_max` and `target_neighbors_500_mean` indicate that the customer has a poor rating with the bureau. The counterfactual indicates `avg_bal_limit_ratio` that needs to be decreased indicating that the customer has a lot of credit card debt which could indicate financial distress. If the target score is 750 or higher the number of features that require adjustment increases to 5.

Case `116492` is unlikely to default with a predicted score of 848. Running the algorithm with a target score of 600 or lower results in two counterfactuals which show how `116492` has a much more favourable external bureau credit rating among other features to that of a case with a score of 600 or lower. This shows that the algorithm can be used to explain a wide range of scenarios.

Using a narrow counterfactual search range, results in simple counterfactuals also. Taking the case `131594` which has a predicted score of 475 with a counterfactual search range between 600 and 610, the algorithm outputs 1 counterfactual with 4 features to adjust. This indicates that the counterfactual extraction algorithm outputs simple and effective counterfactuals for a wide range of scenarios.

6 Evaluation

The model was evaluated subject to performance and transparency.

6.1 Model Performance

The model performance subject to the measures discussed in Section 3.6 are summarized in Table 3:

Each model has the same performance profile. The implication that business with varying risk appetites will choose different models is not an issue when choosing between the above models as each model delivers the same risk profile. This means evaluating the models subject to a varying misclassification cost as seen in Chen et al. (2021) is not necessary and likely will not yield any significant results.

Table 3: Comparison of Model Performance Between Algorithms

		Accuracy	**Precision**	**Recall**	**AUC**
Train	Credit Scorecard	68.97%	16.49%	70.43%	76.21%
	LightGBM	69.83%	17.24%	72.52%	78.30%
	Random Forest	68.42%	16.30%	70.85%	76.00%
	XGBoost	71.63%	18.34%	73.30%	80.00%
Test	Credit Scorecard	68.84%	16.37%	69.65%	75.66%
	LightGBM	69.60%	16.93%	70.80%	77.39%
	Random Forest	68.47%	16.14%	69.26%	75.15%
	XGBoost	71.40%	17.87%	70.69%	78.12%
Kaggle Submission (AUC)	Credit Scorecard	N/A	N/A	N/A	75.56%
	LightGBM	N/A	N/A	N/A	76.78%
	Random Forest	N/A	N/A	N/A	74.19%
	XGBoost	N/A	N/A	N/A	77.59%

The Random Forest fails to outperform the credit scorecard model. This makes it unsuitable for deployment. The Random Forest is a black box model and attempting to make the Random Forest explainable is a waste of resources given that the traditional credit scorecard model is by its nature, explainable. Both gradient boosting methods outperform the credit scorecard model in every performance metric. When comparing the performance of each model, the XGBoost model it the highest performing overall. Given that the counterfactual method of explanation is model agnostic, the decision was taken to deploy the counterfactual method of explanation on the XGBoost model only.

This shows that model performance can be improved using opaque tree-based ensemble models rather than Logistic Regression. In some cases, it might not be worth the resources to deploy a black-box model and then make it explainable. A banking loan portfolio can be worth billions and in that case any gain in performance can create value. Choosing to deploy the black-box model followed by making it explainable will ultimately depend on the business issue at hand and the resources the developers have. It could be the case that the black box model outperforms the explainable model to the extent that choosing not to deploy it would be costly. This makes the pursuit to come up with a method to make the models explainable worthwhile.

6.2 Model Explainability

The goal of the transparency aspect of the model was so that it could meet the explainability standard set out in European Banking Authority (2020) as documented in Section 2. The counterfactuals extracted for each case can be understood by humans. The underlying factors that drive the model prediction are clear and easily understood by humans. However, knowledge of the underlying features is required to translate these explanations into insights that the lay user can understand. The explanations satisfy the criteria discussed in European Banking Authority (2020) and highlighted in Section 2. This explanation method is model agnostic. This gives banks and other lending institutions a much greater choice of what model to use when predicting loan defaults. This also can be used for other applications where model transparency is a necessity. Overall, the model explainability method proposed satisfies the required criteria making the overall research a success.

There could be a case where the explanation fails to meet ethical standards. Tak-

ing the example in Section 5.4.3, the counterfactual extracted can be interpreted as the clients older age as the reason for a poor credit score. This is unethical because this is discrimination based on the clients age which is banned by the US Equal Credit Opportunity Act. Although this is an issue for the model created in this paper it is not an issue for the proposed concept. This feature could be removed from this model or the algorithm in Section 4.3 could be set up to avoid such sensitive features when searching for counterfactuals. Such issues are beyond the scope of this research. This further highlights the effectiveness of the model explanation approach. The counterfactual-based explanation has highlighted this ethical issue to the developer where it might have been previously missed.

7 Conclusion and Future Work

This research shown that model performance can be improved using black-box predictions methods over the traditional credit scorecard-based models and that this can be done while upholding a defined minimum standard of explainability. The counterfactual search algorithm proposed in this research has been shown to extract rules that explain the model predictions and the drivers behind such predictions. These counterfactuals can be easily translated into insights that can be understood by the lay user. This has wide-reaching implications not only for credit default prediction but for any modelling approach that requires explainability. This extends model choice to include black-box models that are better able to identify the non-linear relationships between the target variable and the predictor variables. This means that banks and lenders have the potential to use models which could help them reduce expenses brought about by loan defaults and/or identify and avoid lending opportunities that are too risky.

Future work on this topic will involve improving the presentation of the explanations. It would be interesting to explore the possibility of improving the visualization outputs from the process to make the explanations clearer. Future work should explore whether the counterfactuals can be automatically output as natural language descriptions. This will allow a user to explain the model predictions with little or no knowledge of the underlying features. In addition to improving the explanations, it would be interesting to assess the generality of the proposed counterfactual extraction algorithm. Further research on the topic would involve assessing the model's effectiveness on different datasets and/or for different prediction applications. It would be interesting to find out if this type of explainable AI can be used for a wide range of applications because it would open up a lot of potential for ML models to be deployed in areas where they were previously deemed not appropriate.

References

Barboza, F., Kimura, H. and Altman, E. (2017). Machine learning models and bankruptcy prediction, *Expert Systems with Applications* **83**: 405–417.
URL: *https://www.sciencedirect.com/science/article/pii/S0957417417302415*

Breiman, L. (2001). Random forests, *Machine Learning* **45**(1): 5–32.
URL: *https://link.springer.com/article/10.1023/A:1010933404324*

Bussmann, N., Giudici, P., Marinelli, D. and Papenbrock, J. (2020). Explainable machine learning in credit risk management, *Computational Economics* .
URL: *https://doi.org/10.1007/s10614-020-10042-0*

Charles, K. K. and Hurst, E. (2002). The transition to home ownership and the black-white wealth gap, *The Review of Economics and Statistics* **84**(2): 281–297.
URL: *https://doi.org/10.1162/003465302317411532*

Chen, K., Zhu, K., Meng, Y., Yadav, A. and Khan, A. (2020). Mixed credit scoring model of logistic regression and evidence weight in the background of big data, *in* A. Abraham, A. K. Cherukuri, P. Melin and N. Gandhi (eds), *Intelligent Systems Design and Applications*, Springer International Publishing, Cham, pp. 435–443.

Chen, S., Guo, Z. and Zhao, X. (2021). Predicting mortgage early delinquency with machine learning methods, *European Journal of Operational Research* **290**(1): 358–372.
URL: *https://www.sciencedirect.com/science/article/pii/S0377221720306846*

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system, *CoRR* **abs/1603.02754**.
URL: *http://arxiv.org/abs/1603.02754*

European Banking Authority (2020). Eba report on big data and advanced analytics.
URL: *https://www.eba.europa.eu/file/609786/*

Fernandez, C., Provost, F. J. and Han, X. (2020). Explaining data-driven decisions made by AI systems: The counterfactual approach, *CoRR* **abs/2001.07417**.
URL: *https://arxiv.org/abs/2001.07417*

Fitzpatrick, T. and Mues, C. (2016). An empirical comparison of classification algorithms for mortgage default prediction: evidence from a distressed mortgage market, *European Journal of Operational Research* **249**(2): 427–439.
URL: *https://www.sciencedirect.com/science/article/pii/S0377221715008383*

Guégan, D. and Hassani, B. (2018). Regulatory learning: How to supervise machine learning models? an application to credit scoring, *The Journal of Finance and Data Science* **4**(3): 157–171.
URL: *https://www.sciencedirect.com/science/article/pii/S2405918817300648*

Islam, S. R., Eberle, W., Ghafoor, S. K. and Ahmed, M. (2021). Explainable artificial intelligence approaches: A survey.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, p. 3149–3157.

Keane, M. T. and Smyth, B. (2020). Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable ai (xai), pp. 163–178.

Lundberg, S. and Lee, S. (2017). A unified approach to interpreting model predictions, *CoRR* **abs/1705.07874**.
URL: *http://arxiv.org/abs/1705.07874*

Moscatelli, M., Parlapiano, F., Narizzano, S. and Viggiano, G. (2020). Corporate default forecasting with machine learning, *Expert Systems with Applications* **161**: 113567.
URL: *https://www.sciencedirect.com/science/article/pii/S0957417420303912*

Munnell, A. H., Tootell, G. M. B., Browne, L. E. and McEneaney, J. (1996). Mortgage lending in boston: Interpreting hmda data, *The American Economic Review* **86**(1): 25–53.
URL: *http://www.jstor.org/stable/2118254*

Onay, C. and Öztür, E. (2018). A review of credit scoring research in the age of big data, *Journal of Financial Regulationand Compliance* **26**: 382–405.
URL: *http://dx.doi.org/10.1108/JFRC-06-2017-0054*

Prentzas, N., Nicolaides, A., Kyriacou, E., Kakas, A. and Pattichis, C. (2019). Integrating machine learning with symbolic reasoning to build an explainable ai model for stroke prediction, pp. 817–821.

Qiu, Z., Li, Y., Ni, P. and Li, G. (2019). Credit risk scoring analysis based on machine learning models, *2019 6th International Conference on Information Science and Control Engineering (ICISCE)*, pp. 220–224.

Refaat, N. (2011). *Credit Risk Scorecard: Development and Implementation Using SAS*, John Wiley and Sons.

Siddiqi, N. (2005). *Credit Risk Scorecards: Developing And Implementing Intelligent Credit Scoring*, SAS Publishing.

Wachter, S., Mittelstadt, B. and Russell, C. (2018). Counterfactual explanations without opening the black box: Automated decisions and the gdpr, *Harvard journal of law and technology* **31**: 841–887.