Vrije Universiteit Amsterdam
Faculty of Science

Master Thesis Business Analytics

Detecting and mitigating bias according to the European AI Act in the credit risk engine of a Dutch telecom company

Author: Davita van der Pol (2701002)

VU Supervisor: Rikkert Hindriks
Second reader: Mark Hoogendoorn

February 2026

1 Abstract

With the arrival of the European AI Act, many companies have had to reevaluate the use of their AI systems. If one is classified as high risk, it has to comply with various regulations. One of those regulations is that the model should not contain bias that can lead to discriminatory decisions. The Dutch telecom company X has a model to predict whether a potential customer will commit fraud. This prediction is then used in the decision of whether the customer gets accepted. This project investigates methods that can be used to detect and mitigate bias in this model. There exists literature on fairness in machine learning regarding fraud data, however, there is a gap where methods that specifically consider fraud data are applied on real world data. This research uses existing techniques to detect and mitigate bias on real world fraud data, which comes with its own set of challenges, but can be valuable from both a business and academic perspective. To test group fairness, the precision parity and recall parity are measured, and to test individual fairness, the counterfactual fairness technique is applied. The results indicated bias against several variables, likely attributable to the large imbalances in the data. Next, two methods were applied to mitigate bias. The first method combined Fair-SMOTE with threshold optimization. It performed worse than the original model on accounts of both accuracy and fairness, likely due to the excessive amount of synthetic data generated by Fair-SMOTE. The second method used the Area Under the Precision Recall curve instead of the Area Under the Receiver Operating Characteristic curve as an evaluation metric and combined it with threshold optimization. This method slightly improved fairness with only a small accuracy loss. This research succeeded in detecting bias according to the European AI Act and in reducing this bias. It also highlights the need to use more advanced mitigation techniques to fully mitigate bias.

2 Introduction

In recent years, the use of Artificial Intelligence (AI) has increased rapidly, leading to a growing concern about the fairness of AI (Garcia et al., 2024; Trinh & Zhang, 2024; Kozodoi et al., 2022). AI can offer significant benefits, particularly in tasks that can be performed more quickly and efficiently than by humans. For example, there has been a growing number of studies in healthcare, in which AI can help diagnose patients faster and more accurately (Alowais et al., 2023). However, AI systems can cause harm when not used appropriately. They can unintentionally perpetuate existing biases in society (Mehrabi et al., 2021). For example, an algorithm that predicts recidivism in criminals and is being used by the American government to help make decisions about parole was found to exhibit racial bias against Black individuals (Howard & Borenstein, 2018).

In 2021, the European Union proposed the European AI Act to regulate the use of AI (European Parliament, 2023), which has since been adopted. The AI

Act categorizes AI systems according to risk. Essentially, systems that have the potential to create harm are classified as high-risk and should comply with certain regulations. One of those regulations is the need for methods to detect, prevent, and mitigate possible biases that can lead to discrimination (European Union, n.d.-a). The Credit Risk Engine (CRE) used by the Dutch telecom company X predicts whether a potential customer will pay their subscription or commit fraud. The system is classified as high risk because it is used to evaluate a person's creditworthiness and can determine their access to telecommunications, which is considered an essential service (European Union, n.d.-b).

This project aims to answer the following research question: "*Does the credit risk engine at Dutch telecom company X contain bias according to the European AI Act, and if so, how can this bias be mitigated while keeping accuracy at an appropriate level?*"

Essentially, this question can be split into two parts. The first part is detecting bias and the second part is mitigating bias. There are numerous techniques to do both, so the background of the data should be considered when selecting the appropriate methods. For example, the data are highly imbalanced.

There have been studies that specifically considered the fairness of fraud detection systems. However, most of this research either did not use real-world data, which presents additional challenges, or focused only on one aspect of fairness in fraud data. For example, many studies on data balancing, such as the study by Gupta et al. (2023), considered class imbalance but not group imbalance, which can damage protected attributes further. Kusner et al. (2017) introduced counterfactual fairness to test whether a model can be considered fair on an individual level. However, it assumes that a correct causal graph of the data is already known, which is an unrealistic expectation to have of a real-world dataset. This research aims to fill this scientific gap by finding appropriate methods to detect and mitigate bias that can be used on real-world data and that consider the different aspects of fairness in fraud data. Since businesses are required to comply with the European AI Act, this research is also valuable from a business perspective.

The remainder of this paper is organized as follows. First, more context on the European AI Act is provided. Then, in section 3 there is an overview of the existing literature on the subject. In section 4, the data and the current model are explained. Section 5 explains the methods that were used to detect and mitigate bias in this project, and section 6 provides the results of these methods. Finally, the discussion and conclusion will be in sections 7 and 8, respectively.

The European AI Act. As mentioned, high-risk AI systems must comply with many regulations (European Union, n.d.-c). Examples include the requirement that the system should be regularly reviewed and updated, that the datasets are managed properly, that factors such as data preparation and biases are considered, that technical documentation should be kept, and that the system must be transparent, accurate, and secure.

This research focuses on detecting and mitigating potential biases. The AI Act requires appropriate measures to detect, prevent, and mitigate possible biases that are likely to lead to discrimination prohibited under Union law, especially where output influences the input for future operations (European Union, n.d.-a). This includes, but is not limited to, discrimination against race, gender, nationality, and age (Het College voor de rechten van de mens, n.d.).

The AI Act does not specify guidelines on how to implement its requirements. Therefore, appliedAI and its partners created an in-depth report sharing best practices to implement the requirements for high-risk AI systems (Machado et al., 2025). For bias detection, they recommend measuring class imbalance, detecting disparities by comparing model predictions across demographic groups, assessing fairness by slightly modifying demographic-related inputs to see if predictions change, and applying fairness metrics and checking if the output disproportionately favors one group. The report does not mention specific guidelines to mitigate bias.

3 Literature review

3.1 Credit risk models

There have been many studies on credit risk models. Traditional statistical models are widely used in this domain due to their interpretability and relatively low computational costs (X. Zhang & Yu, 2024; Gunnarsson et al., 2021). Recently, more complex methods have become more widely researched. This includes ensemble methods and even deep learning algorithms (Gunnarsson et al., 2021).

Dastile et al. (2020) conducted a comprehensive literature study on existing credit scoring techniques. They reviewed the most commonly used statistical and machine learning techniques to model credit scoring. In terms of Percentage Correctly Classified (PCC), they found that Convolutional Neural Networks (CNN) performed best, followed by ensemble classifiers such as Random Forests, Bagging, and Boosting. The survey also includes techniques to handle data imbalance, which is a significant obstacle in credit risk modeling. There are usually much more non-fraud cases (majority class) than fraud cases (minority class). They found that not many studies balance their data, and if they do, the most used technique is under-sampling the majority class. The data imbalance also influences how well certain evaluation metrics capture the accuracy of the model. This paper found that the G-mean, the recall, and the F1-score perform best for imbalanced data.

X. Zhang & Yu (2024) compared classification algorithms for credit risk assessment. Traditional single classifiers, such as Linear Discriminant Analysis (LDA), Logistic Regression (LogR), K-Nearest Neighbor (KNN), and Naive Bayes (NB), are widely used in credit scoring. They have less computation time and better interpretability than more complex methods. However, predictive performance is limited as they may not capture nonlinear relationships in

the data. Intelligent single classifiers, such as Artifical Neural Network (ANN), Decision Tree (DT), Support Vector Machine (SVM), and deep learning techniques, have a better prediction performance than traditional single classifiers. However, they can suffer from local optimization, overfitting, and parameter sensitivity. Ensemble multiple classifiers are formed by combining multiple different classifiers, examples are Random Forest (RF), XGBoost, and LightGBM. Although these algorithms generally have high accuracy, the interpretability is low.

Gunnarsson et al. (2021) researched the appropriateness of deep learning techniques for credit scoring. They found that deep learning techniques do not seem to be appropriate for credit scoring. Deep neural networks did not outperform their shallower counterparts and have a much greater computational cost. Of the methods they considered, XGBoost performed best for credit scoring. One drawback is that it is a "black box" model, meaning that the interpretability is low.

Brown & Mues (2012) compare classification algorithms for credit scoring based on real-life imbalanced datasets. They found that random forests and gradient boosting classifiers were able to handle large class imbalances well. They also saw that the most commonly used credit scoring techniques, LDA and LogR, were quite competitive with the more complex techniques. However, performance was measured with the Area Under the Receiver Operating Characteristic Curve (AUC), which does not always capture all information when data are highly imbalanced (Hancock et al., 2023).

In this research, the XGBoost model is used. It was found to perform well compared to traditional single classifiers (X. Zhang & Yu, 2024) and deep learning techniques (Gunnarsson et al., 2021). Additionally, it can perform well on imbalanced data (Brown & Mues, 2012). One drawback is the low interpretability.

3.1.1 Evaluation metrics

Due to the imbalanced nature of the data, not all evaluation metrics are appropriate in the credit scoring context. Several studies have investigated which metrics capture the full picture and which metrics can give a distorted view of the performance when data are imbalanced.

Hancock et al. (2023) state that when the data are imbalanced, an alternative to the AUC is the Area Under the Precision Recall Curve (AUPRC). The AUPRC plots precision and recall scores, while the AUC plots the true positive rate (TPR) and the false positive rate (FPR). Recall and TPR are the same, so the distinction lies in the difference between precision and FPR. The FPR involves true negatives, and the precision does not. The negative class (no fraud) is the majority class in credit risk modeling, so in the FPR, false positives are drowned out due to the large true negative class. This makes the precision more appropriate than the FPR in this context.

de la Cruz Huayanay et al. (2024) investigated the performance of evaluation metrics for classification tasks with imbalanced data. They found that the

Matthews Correlation Coefficient (MCC), G-mean, and Cohen's Kappa perform best, and that AUC and Accuracy metrics perform poorly in the studied scenarios. They also found that if the negative class is the majority class, the following metrics perform well: TPR, Critical Success Index (CSI), Standard Gilbert skill score (SGS), Success Synchronization Index (SSI), FAITH, MCC, G-mean, F1 score, and Cohen's Kappa.

Chicco & Jurman (2020) compared the performance of the MCC, the F1-score, and the accuracy score on imbalanced data. They found that on data where the majority class contains negative samples, the accuracy score gives an overly optimistic view of the performance, whereas the MCC and F1-score are much more realistic.

Due to the high data imbalance, this research includes the AUPRC, G-mean, F1-score, and MCC.

3.1.2 Data imbalance

Data imbalance can significantly influence the performance and fairness of credit risk models. Various studies have investigated the best technique for handling imbalanced data, with varying results. Studies that also considered fairness (Chakraborty et al., 2021; Sonoda, 2023) generally found that fair oversampling techniques performed best.

Gupta et al. (2023) compared balancing techniques to deal with imbalanced fraud data. The techniques were aimed at improving precision, recall, F1, and accuracy scores. They found that the Random Oversampling (ROS) technique suited the imbalanced data best and that XGBoost was the best performing model.

Xiao et al. (2021) tried combinations of different resampling methods and classification models on imbalanced credit scoring problems. They found that Random Undersampling (RUS) and Synthetic Minority Oversampling Technique (SMOTE) combined with Wilson's edited nearest neighbor performed best as resampling methods. As for classification models, LogR and Adaptive Boosting performed best. They found the optimal combination to be RUS with Random Subspace (an ensemble classification model).

Chakraborty et al. (2021) not only focused on class imbalance, but also included group imbalance in their research. Class imbalance means that the target variable is imbalanced and group imbalance means that groups within variables are imbalanced. Group imbalance can lead the model to be biased towards the minority group within a sensitive attribute. They compared how class balancing techniques affect fairness and found that traditional techniques such as RUS, ROS, SMOTE, and KMeans-SMOTE usually improve performance but damage fairness. This is because those techniques randomly generate (oversampling) or discard (undersampling) samples to equalize two classes, which can further damage the protected attribute. To combat this, they introduce a fair oversampling technique called Fair-SMOTE, which considers class imbalance and group imbalance. This technique balances data such that the privileged and unprivileged groups end up with an equal amount of positive and negative samples.

Sonoda (2023) also considered class and group imbalance, and proposed a fair oversampling technique using heterogeneous clusters. It is similar to Fair-SMOTE, the difference being that Fair-SMOTE uses homogeneous clusters. With homogeneous clusters, they are intersections of a single group with a single class. If those clusters are small, it may lead to overfitting. Using heterogeneous clusters, the proposed technique should be robust to overfitting. They compared the performance of their technique to SMOTE and several fair oversampling techniques. They tested it with different models and on different datasets, one of which was a German credit dataset. On this dataset, their algorithm generally achieved the best fairness, followed by other fair oversampling techniques. They measured the accuracy of the model with Balanced Accuracy (BAcc), which is used often in imbalanced classification problems. The performance of each technique depended on the model, but generally, the fair oversampling techniques performed best.

Dablain et al. (2024) considered the fairness accuracy trade-off. They linked the trade-off to data imbalance, because this can cause one group to be underrepresented with respect to another group. They introduced a technique that they call Fair Oversampling (FOS), which numerically balances classes and debiases protected features through feature blurring. The number of training examples are equalized between classes and it mixes samples between members of the protected group, so that the classifier becomes confused about the protected feature. They found that FOS improved both accuracy and fairness over baselines.

Oversampling techniques were found to work well for fraud data (Gupta et al., 2023; Xiao et al., 2021). Since the data in this project suffer from group imbalance as well as class imbalance, the focus is on fair oversampling techniques.

3.1.3 Bias and fairness

Data imbalance can lead to a bias in favor of the majority class (Dastile et al., 2020) and the majority groups (Sonoda, 2023). Additionally, credit risk models generally suffer from sampling bias (Kozodoi et al., 2025), which comes from the fact that there are only data on previously accepted customers.

Kozodoi et al. (2025) address this sampling bias by developing methodologies that mitigate it. The first methodology introduces bias-aware self-labeling (BASL), which addresses sampling bias during training. Their second methodology consists of a Bayesian evaluation (BE) framework that leverages unlabeled data of rejected clients and estimates accuracy from a joint sample of accepted and rejected clients.

More generally, Garcia et al. (2024) conducted a systematic literature review on algorithmic discrimination in the credit domain. They found that most research considers algorithmic discrimination related to race and gender. The results revealed the presence of discrimination in many countries against women and people of certain races. Proxies are also researched, which are attributes that correlate with the sensitive attribute. Simply removing the sensitive attribute was found to be ineffective in removing bias due to the existence of these

proxies.

Trinh & Zhang (2024) introduced several bias mitigation techniques and include the fairness-accuracy tradeoff corresponding to these techniques. For example, synthetic data generation, which creates balanced training datasets by generating additional instances for underrepresented groups, has a relatively low impact on accuracy.

Kozodoi et al. (2022) considered the potential trade-off between fairness and profit. They found that in-processing techniques, which introduce fairness constraints during model training, perform best in finding a profit-fairness trade-off. Achieving perfect fairness is costly, however, it is possible to reduce discrimination to a reasonable extent without sacrificing too much profit.

3.2 Detecting bias

Detecting bias (in the context of discrimination) can be done by measuring fairness. Often, methods to detect bias are split into those that measure group fairness and those that measure individual fairness. Individual fairness considers whether the algorithm gives similar predictions to similar individuals, while group fairness considers whether whole groups are treated equally (Mehrabi et al., 2021). Methods that detect bias in credit risk models should account for the imbalanced nature of fraud data.

3.2.1 Group fairness

Kamalaruban et al. (2024) discussed fairness metrics that take the imbalanced nature of fraud data into account. They gave several group fairness metrics that measure the difference in certain rates between groups and normalized them. They found that without this normalization, many biases can go unnoticed because of the data imbalance. For example, the FPR parity metric measures the absolute difference in FPR between two groups. Let $G \in \{0, 1\}$ be a sensitive attribute with groups 0 and 1. Then the FPR parity can be calculated as follows:

$$\Delta_{FPR} = |FPR_{G=0} - FPR_{G=1}|$$

and the normalized FPR parity can be computed with:

$$\frac{|FPR_{G=0} - FPR_{G=1}|}{\max\{FPR_{G=0}, FPR_{G=1}\}}.$$

Other commonly used group fairness metrics are (Mehrabi et al., 2021; Pessach & Shmueli, 2022; Kamalaruban et al., 2024):

Demographic parity: Also known as statistical parity, this ensures that the positive predictions are equal across groups. This measure is computed as:

$$P(\hat{Y} = 1|G = 0) = P(\hat{Y} = 1|G = 1).$$

Equalized odds: This measure is different from demographic parity because it does not necessarily equalize positive predictions. Rather, it computes the

differences between the FPR of the two groups and the TPR of the groups. It is computed as follows:

$$P(\hat{Y} = 1|G = 0, Y = y) = P(\hat{Y} = 1|G = 1, Y = y), \text{ for } y \in \{0, 1\}.$$

Equal opportunity: This is similar to the equalized odds metric, but focuses only on the TPR. It is formulated as:

$$P(\hat{Y} = 1|G = 0, Y = 1) = P(\hat{Y} = 1|G = 1, Y = 1).$$

Choosing which metric to use depends on the problem at hand. Often, not all metrics can be satisfied simultaneously. For example, when the groups have different proportions of positive outcomes, equalized odds and demographic parity become incompatible. This can also be explained by considering the three fairness criteria: independence, separation, and sufficiency (Kozodoi et al., 2022).

Independence: This criterion requires the model predictions to be independent of someone's group membership. Measures that focus on independence ignore actual outcomes. Demographic parity is an example of such a measure. Often, this constraint is infeasible for real-world applications (like credit scoring), since the resulting loss in model performance will have a negative effect on a business's performance.

Separation: The equalized odds measure is related to the separation criterion. The equalized odds difference can be defined as (Mariscal et al., 2024):

$$EOD = \frac{|FPR_{G=1} - FPR_{G=0}| + |TPR_{G=1} - TPR_{G=0}|}{2}.$$

For credit scoring, this criterion might be the most fitting. Instead of requiring equal positive predictions, it requires similar error rates between groups, making it closer to reality than the independence criterion.

Sufficiency: Here, the predicted outcome is required to be independent of group membership. Given that the Negative Prediction Value (NPV) (Kamalaruban et al., 2024) is measured by $NPV = \frac{TN}{TN+FN}$, the sufficiency criterion can be measured as follows:

$$SF = |NPV_{G=0} - NPV_{G=1}|.$$

Kozodoi et al. (2022) argue that this criterion is less suitable for credit scoring, because even if it is satisfied, the separation criterion can still be violated. Therefore, in this research, metrics related to the separation criterion are used. It is more relevant to obtain similar error rates than similar amounts of fraud predictions between groups.

3.2.2 Individual fairness

Individual fairness is satisfied if the algorithm gives similar predictions to similar individuals. A way to test this is with counterfactual fairness, where the model

output with observed data as input can be compared to the model output with counterfactual data as input. Counterfactual data are data where the sensitive attribute in the original data is flipped. One drawback of simply changing the sensitive attribute is that it ignores causal relationships. In reality, a change in one attribute would likely result in a change in various other attributes (Krafft et al., 2024).

Kusner et al. (2017) introduced counterfactual fairness that considers causal dependencies. Essentially, a causal model is defined based on a causal graph that represents causal dependencies within the data. The causal model is then used to generate counterfactual data. If the predictions remain the same for the observed data and counterfactual data, the model can be considered fair because the predictions do not depend on the sensitive attribute.

Dwork et al. (2012) tested individual fairness in a different manner. They used a task-specific similarity metric that describes the extent to which pairs of individuals should be considered similar for the classification task. The similarity metric is formalized as a Lipschitz condition on the classifier. This condition requires that the distributions of the results observed by variables x and y are indistinguishable up to their distance $d(x, y)$. They developed an optimization problem that minimizes the expected loss and is subject to the Lipschitz condition. However, this research will focus on counterfactual fairness, since it is researched more broadly.

3.3 Mitigating bias

Techniques for removing bias generally fall into three categories: pre-processing methods, in-processing methods, and post-processing methods (Mehrabi et al., 2021). Pre-processing techniques modify the dataset before training the classifier to reduce the influence of protected attributes on the model output. In-processing methods integrate bias mitigation into the classifier training process. Generally, it includes adjusting the objective function to balance accuracy with fairness constraints (Trinh & Zhang, 2024). Post-processing methods aim to make predictions more fair after the classifier has made them (Kisten & Khosa, 2024). In-processing techniques are often the most effective. However, since those methods are mostly problem specific, their generality is limited. Post-processing techniques are more general, however, adjusting the prediction according to certain fairness criteria generally results in lower accuracy (Kozodoi et al., 2022).

3.3.1 Pre-processing

Fairness through unawareness simply removes the sensitive attribute from the dataset. However, multiple studies found this method to be ineffective due to proxies, which are variables that strongly correlate with the sensitive attribute (Genovesi et al., 2024; Garcia et al., 2024; Kamalaruban et al., 2024).

Calders et al. (2009) proposed reweighing as a mitigation technique. This technique assigns weights based on the probabilities of group-class combina-

tions, which are the possible combinations that can be made from the variable groups and the classes (e.g., men who commit fraud, men who do not commit fraud, women who commit fraud, and women who do not commit fraud). The data is then sampled with replacement according to those weights to create a balanced dataset. Kisten & Khosa (2024) and Trinh & Zhang (2024) found that reweighing can reduce disparities with a limited accuracy loss. Kozodoi et al. (2022) considered how bias mitigation techniques influence profit and found that while reweighing improved fairness, there was a significant loss in profit.

Feldman et al. (2015) introduced the disparate impact remover, which minimizes disparate impact by modifying the value of the sensitive attribute. Disparate impact is a measurement that considers the ratio of favorable predictions for the unprivileged group against the privileged group. Kisten & Khosa (2024) and Trinh & Zhang (2024) both found that the disparate impact remover can reduce bias with a minimal loss in accuracy. Kozodoi et al. (2022), who investigated the trade-off between fairness and profit, found that the disparate impact remover retains profit but offers only small fairness improvements.

Synthetic data generation is a pre-processing technique that creates synthetic data. Examples are data balancing methods where new data is generated. Trinh & Zhang (2024) stated that this can significantly improve fairness without sacrificing too much accuracy. Therefore, this technique is focused on.

3.3.2 In-processing

Hort et al. (2024) conducted a survey on bias mitigation techniques in machine learning. They specified several in-processing techniques, such as regularization and constraints. They both change the algorithm's loss function. Regularization adds a term to the loss function that penalizes discrimination, and constraints set limits on bias levels that cannot be exceeded during training.

An example of a regularization technique is the prejudice remover, introduced by Kamishima et al. (2012). It is based on the prejudice index (PI), which measures the amount of mutual information between the sensitive attribute (x_a) and the class (y) (Kozodoi et al., 2022):

$$PI = \sum_{y, x_a \in D} \mathbf{P}(y, x_a) \ln \left(\frac{\mathbf{P}(y, x_a)}{\mathbf{P}(x_a)\mathbf{P}(y)} \right),$$

where $\mathbf{P}(y, x_a), \mathbf{P}(x_a),$ and $\mathbf{P}(y)$ are empirical distributions of y and x_a in the sample D. The prejudice remover adds ηPI to the loss function of the classifier, where η controls the importance of PI. Kozodoi et al. (2022) found that the prejudice remover has a limited loss in accuracy and profit, but a relatively small improvement in fairness.

An example of a mitigation technique that uses constraints is the meta fair algorithm (Celis et al., 2019). This algorithm takes a fairness constraint as input. It takes a group-wise fairness metric, where similar values across groups indicate fairness. The constraint requires that the difference between the metric values between groups cannot exceed a certain threshold. Kozodoi et al. (2022)

found that the meta fair algorithm performed poorly in terms of accuracy, profit, and fairness.

Another in-processing method is adversarial debiasing (Kozodoi et al., 2022). This method uses two neural networks with different objectives. The first network, called the predictor, tries to learn a function to predict y given X, while minimizing the success of the second network, the adversary. The adversary takes the output layer of the first network $\hat{y}$ and the observed values y and tries to predict the sensitive attribute. B. H. Zhang et al. (2018) found that adversarial debiasing can improve fairness with a limited loss of accuracy. Kozodoi et al. (2022) found that while adversarial debiasing generally improves fairness, prejudice remover improves it even more and results in higher accuracy and profit.

Z. Liu & Bondell (2019) found that classifiers that optimize the AUPRC can outperform classifiers that optimize the AUC. Adjusting the loss function can lead to fairness improvements with limited accuracy losses. However, determining exactly how to change it can be difficult and would require separate research.

3.3.3 Post-processing

An example of a post-processing method is threshold optimization. Threshold optimization techniques aim to equalize error or approval rates by applying different decision thresholds across demographic groups (Trinh & Zhang, 2024). Leevy et al. (2023) found that for balanced data, the default threshold generally yields good performance but that it can yield poor performance if the data is imbalanced. Therefore, threshold optimization can work well for this research.

Reject option classification is another post-processing method (Kamiran et al., 2012). It defines a critical region of high uncertainty and relabels instances in this region in a manner that reduces discrimination. Kozodoi et al. (2022) found that the reject option classification performs well on the fairness criteria, however, it has a relatively high loss of accuracy and profit. Since retaining an appropriate accuracy level is important in this research, this method will not be focused on.

4 Data and model

4.1 Data

The dataset contains consumer orders for sim only or a small handset loan (below €250) at a Dutch telecom company.

The training data used for this project range from 1 September 2022 to 31 March 2025. It contains 492,956 rows (91.9%) and the test set contains 43,287 rows (8.1%). There are 373 variables, however, many of them are irrelevant since they contain information about businesses and this model is purely for consumers. Those variables contain only NaN values in this dataset and are removed. Furthermore, a whitelist was created for the existing model. In this

project, only the variables on this whitelist are considered. An exception is the variable *gender*, which is not on the whitelist. It is not used in training models, but it is interesting to consider this variable regarding bias. Including *gender*, 88 variables are used in this project. 5 are binary (5.7%), 69 are continuous (78.4%), and 14 are categorical (15.9%). The categorical variables are one-hot-encoded before using the data to train the XGBoost model.

In the datatset, there were 492,341 unique customers (99.9%). There was one duplicate row, so the duplicate was removed. It was a non-fraud case, so it makes relatively no difference to the total (non-)fraud cases.

It should be noted that there is a difference between people who commit fraud on purpose and people who simply cannot pay their subscription. Presumably, people who intend to pay their subscription but end up being unable to do so fill in accurate information. People who intend to commit fraud are unlikely to provide accurate personal information. When applying for a subscription, most fields are empty, so when someone would fill in the fields randomly, it is not the case that one value is more likely than the other. Only the type of identification document is automatically filled in as drivers license and the country of the identification document is automatically set to the Netherlands. However, the document number is checked after the model has run, so customers with false numbers are eventually rejected. It should also be noted that since this model is used for customers who request a sim only or small handset loan, the cost of someone committing fraud is generally low.

4.1.1 Seasonality

The date and time of the request are known. The distribution of the months, weekdays, and hours can be found in Figures 1, 3, and 5, respectively. The normalized distributions of the months, weekdays, and hours, which display the fraud cases relative to the total samples for each value, can be found in Figures 2, 4, and 6, respectively.

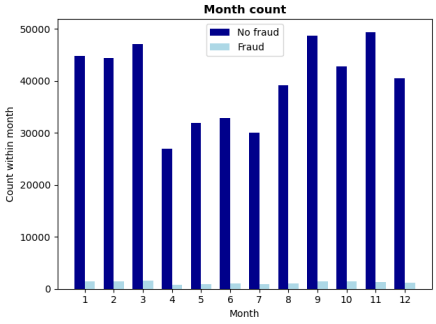

Figure 1: Absolute counts per month

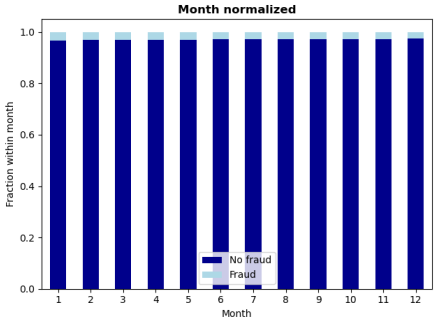

Figure 2: Normalized counts per month

It can be seen in Figure 1 that the number of subscription applications differs

per month. This is in part due to the fact that the data range from September 2022 to March 2025, so the months from April to August have a year less of data. However, Figure 2 demonstrates that the month makes relatively no difference in the probability of committing fraud.

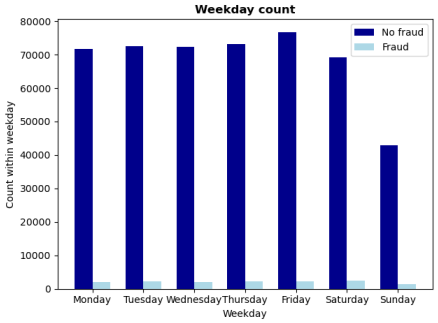

Figure 3: Absolute counts per weekday

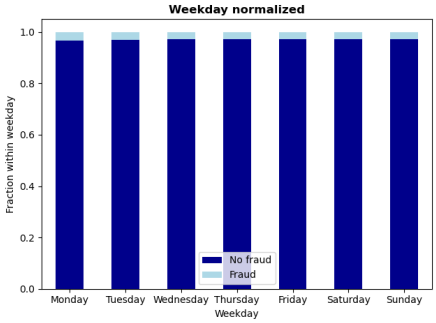

Figure 4: Normalized counts per weekday

In Figure 3 it can be seen that from Monday to Saturday there is a similar number of applications, with slightly more on Friday and slightly less on Saturday. There are relatively few applications for subscriptions on Sunday. Figure 4 indicates that the weekday of application makes relatively no difference in the probability of committing fraud.

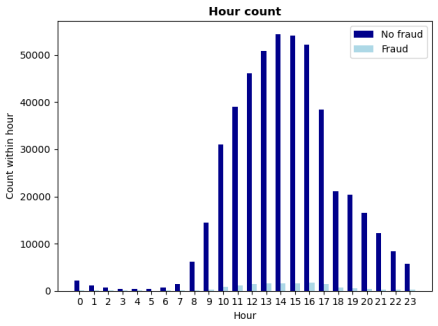

Figure 5: Absolute counts per hour

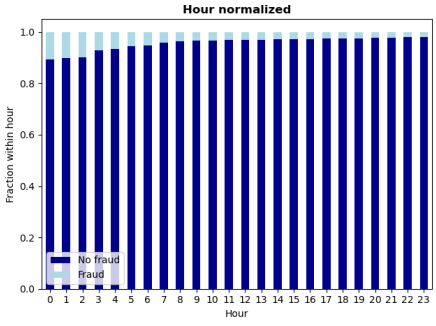

Figure 6: Normalized counts per hour

Figure 5 indicates that most applications are made during the day. Toward the end of the day, there are fewer applications, and there are almost none at night. Interestingly, it can be seen in Figure 6 that relatively many applications made at night are fraudulent. During the day, the specific hour makes relatively no difference in the probability of fraud.

4.1.2 Sensitive attributes

Sensitive or protected attributes are features that could be discriminatory towards an individual's race, gender, nationality, etc.. The features that are considered sensitive in this project are displayed in Table 1.

Feature	Explanation
calculated_countrycode_nld	1 if the individual has a Dutch identification document, 0 otherwise
calculated_age	Age
gender	1 if male, 0 otherwise
focumcheck_statpc4index	Risk index on 4 position zipcode (e.g., 1234)
focumcheck_statpc5index	Risk index on 5 position zipcode (e.g., 1234A)
focumcheck_statpc6index	Risk index on 6 position zipcode (e.g., 1234AB)
focumcheck_debtresult_d02	1 if score on address > 0, 0 otherwise
focumcheck_debtresult_d03	1 if score on zipcode > 0, 0 otherwise
focumcheck_debtresult_d99	1 if score on person, address, and zipcode is 0, 0 otherwise

Table 1: Sensitive attributes and their explanations

The dataset has a high class imbalance, as only 2.9% are fraud cases. In addition, a variable can have a high group imbalance. This is the case if one group within a variable is much larger than another group within that variable. The distribution of each sensitive attribute in Table 1 will be discussed.

calculated_countrycode_nld. The variable *calculated_countrycode_nld* is a binary variable with a value of 1 when an individual has a Dutch identification document (ID) and 0 if they do not. For example, a Dutch visa also counts as a Dutch ID, so a value of 1 does not automatically imply a Dutch nationality. This variable has a high group imbalance, as 90% of the samples have value 1, and only 10% of the samples have value 0. Of the samples with a Dutch ID, 2.1% committed fraud, while 10.3% of the samples without a Dutch ID committed fraud. The distribution of absolute counts can be found in Figure 7. Here, the high group imbalance can be seen. The normalized distribution is visualized in Figure 8, where the counts are normalized with respect to the groups. It can be seen that, relative to the group sizes, group 0 contains much more fraud cases than group 1. This variable does not contain NaN values.

calculated_age. The variable *calculated_age* is a continuous variable that represents the age of an individual. The ages range from 17.1 to 123.8 and are calculated based on what someone entered as their birthdate, which could be incorrect. Figure 9 illustrates how the absolute counts of the binned ages are

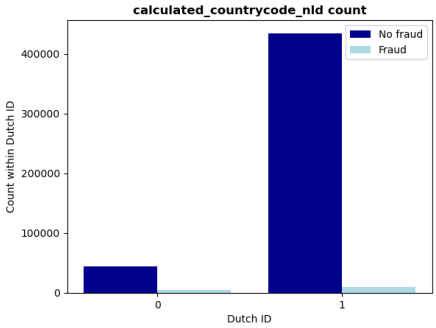

Figure 7: Absolute counts calculated_countrycode_nld

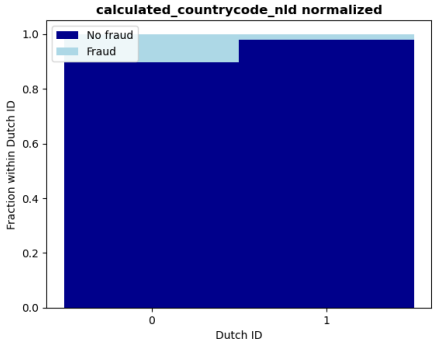

Figure 8: Normalized counts calculated_countrycode_nld

distributed. Figure 10 depicts the normalized counts with respect to the total samples in each age range. Here, it is clear that younger people commit relatively more fraud than older people up to approximately age 90. Of the people who are over 90 years of age, relatively many commit fraud, presumably due to them dying and being unable to pay anymore. Additionally, there are limited samples in that range, so a small amount of fraud cases can look more extreme than when there are many samples. This variable does not contain NaN values.

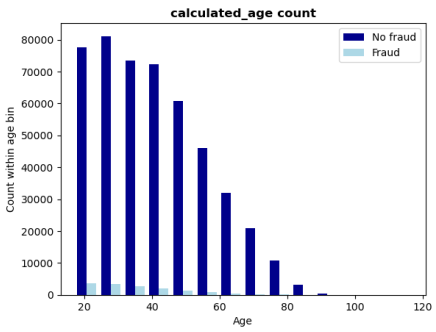

Figure 9: Absolute counts calculated_age

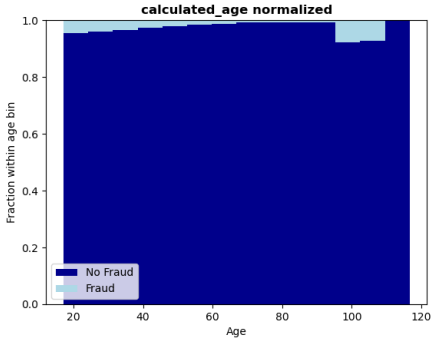

Figure 10: Normalized counts calculated_age

gender. The variable *gender* is not used during model training. It has three categories: man (M), woman (V), and unknown (O). There are no additional NaN values. In the training data, 57.2% are male, 42.8% are female, and 0.0% are unknown. In Figure 11 this group imbalance is visualized. Figure 12 presents the normalized counts for each group. It can be seen that in the unknown group there are few fraud cases, specifically, only 0.6% committed fraud. For men

and women, the fraud rates are closer together. 3.5% of men committed fraud and 2.1% of women. Since the unknown group is extremely small, and most techniques used in this project require the variable to be split into two groups, unknown samples and women are grouped together.

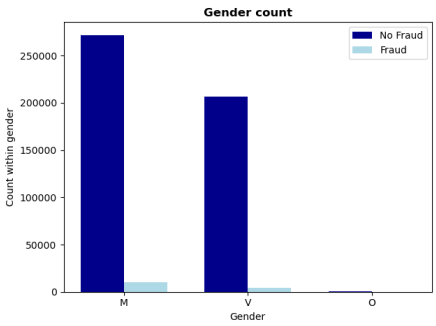

Figure 11: Absolute counts gender

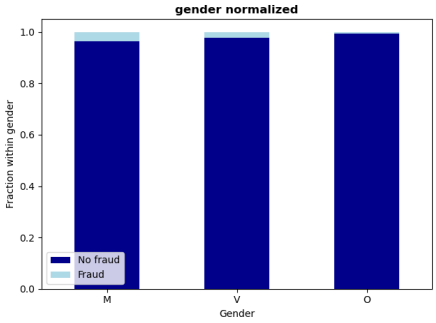

Figure 12: Normalized counts gender

focumcheck_statpc4index. The variable *focumcheck_statpc4index* is continuous and ranges from 0 to 463. It indicates a risk score based on the four position zipcode (e.g., 1234). In Figure 13 it can be seen that the majority of the samples fall around the 100 mark or below. Figure 14 depicts how many people committed fraud of the total samples in the corresponding bin. It can be seen that as the risk score increases, so does the fraud rate. However, the fraud rate starts to decrease again at approximately the 300 mark. Presumably, sampling bias plays a role in this. There are only data on previously accepted customers, and people with a high risk score were likely mostly accepted if their other attributes indicated that they were less likely to commit fraud. This variable contains 1255 (0.3%) NaN values, 48 (3.8%) of which were fraud cases and 1207 (96.2%) of which were not.

focumcheck_statpc5index. Continuous variable *focumcheck_statpc5index* indicates a risk score based on the five position zipcode (e.g., 1234A). The values range from 0 to 1157, most of which range from 0 to 600 (99.7%). This can also be seen in Figures 15 and 16. It follows a similar pattern to the variable *focumcheck_statpc4index*, where the fraud rate increases along with the score up to a certain value. At around 400, there start to be less fraud cases, likely for the same reasons mentioned before. There are almost no samples with scores above 700, represented by the gaps in Figure 16. Samples for which this is the case are considered outliers. This variable contains 1255 (0.3%) NaN values, the same as *focumcheck_statpc4index*.

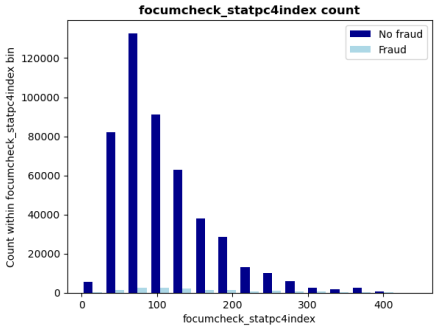

Figure 13: Absolute counts focumcheck_statpc4index

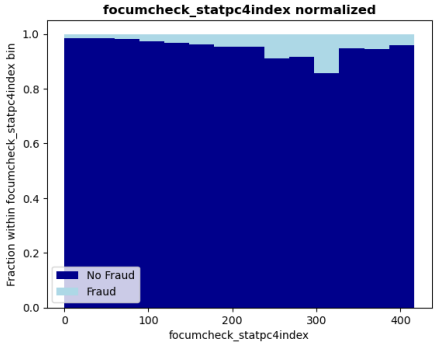

Figure 14: Normalized counts focumcheck_statpc4index

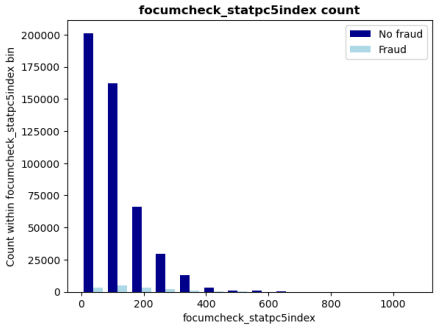

Figure 15: Absolute counts focumcheck_statpc5index

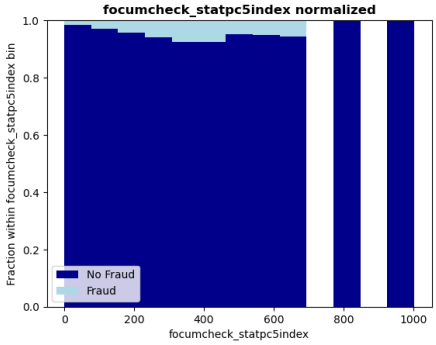

Figure 16: Normalized counts focumcheck_statpc5index

focumcheck_statpc6index. The variable *focumcheck_statpc6index* is a continuous variable that indicates a risk score based on the six position zipcode (e.g., 1234AB). It ranges from 0 to 2314, where 99.6% of the values range between 0 and 1000. In Figures 17 and 18 it can be seen that the distribution of this variable is similar to the other variables that indicate a risk score based on zipcode, namely *focumcheck_statpc4index* and *focumcheck_statpc5index*. Figure 18 indicates that the fraud rate is lower for lower risk scores and that the rate mainly increases as the score increases up to a certain point. For risk scores above 1500 the fraud rate starts fluctuating again, likely for the same reasons as the previous variables. Since there are limited samples with risk scores that high, the relative number of fraud cases can seem more dramatic than they actually are. The NaN values are the same as for *focumcheck_statpc4index* and *focumcheck_statpc5index*.

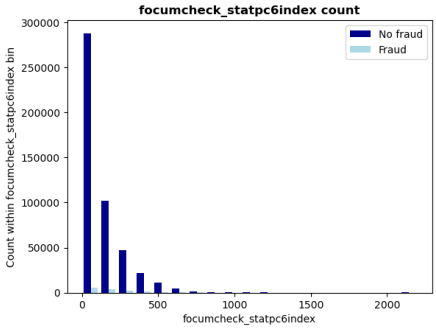

Figure 17: Absolute counts focumcheck_statpc6index

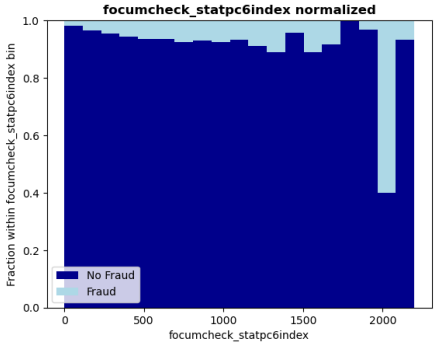

Figure 18: Normalized counts focumcheck_statpc6index

focumcheck_debtresult. The last three features in Table 1 are all one-hot-encoded from the categorical variable *focumcheck_debtresult*. The possible values of this variable are *d01*, *d02*, *d03*, *d99*, or *none*. *d01* means that the score on the person is higher than 0, *d02* means that the score on address is higher than 0, and *d03* means that the score on the zipcode is higher than 0. The value is *d99* if all three have a score of 0 and it is *none* if there is no score. Figure 19 demonstrates that most samples are in category *d99* or *d03*. Figure 20 depicts the fraud rates for each category. It can be seen that category *d99* has the smallest fraud rate, only 1.7% of this category committed fraud. Category *d03* also has a relatively low fraud rate, which is 3.9%. The categories *d01*, *d02*, and *none* have higher fraud rates, which are 8.8%, 8.5%, and 8% respectively. The variable contains 1168 (0.2%) NaN values, 41 (3.5%) of which were fraud cases.

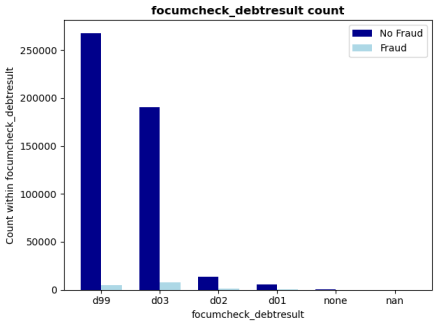

Figure 19: Absolute counts focumcheck_debtresult

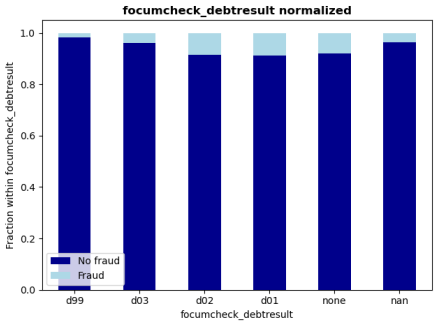

Figure 20: Normalized counts focumcheck_debtresult

4.2 Model

The credit risk engine (CRE) receives data about a potential customer as input and predicts whether they will pay their subscription or commit fraud. The CRE is an extreme gradient boosting (XGBoost) model. XGBoost is known for its processing speed and performance (Dastile et al., 2020). It works similarly to gradient boosting, but the decision trees are built in parallel instead of in a series. It minimizes the optimization function $\mathcal{L}^{(t)}$:

$$\mathcal{L}^{(t)} = \sum_{k=1}^{n} l(y_k, \hat{y}_k^{(t-1)} + \phi_t(x_k)) + \Omega(\phi_t),$$

where $l()$ is the loss function and $\Omega(\phi_t)$ is a regularization term that penalizes the complexity of the model. Here, ϕ_t approximates the function that denotes the prediction. The goal of XGBoost is to find the value of ϕ_t that minimizes the objective function $\mathcal{L}^{(t)}$. In this project, logistic regression for binary classification is used.

To build the model, the *xgboost* package in Python was used (Chen et al., 2026). Several of the model's hyperparameters were set to a certain value, and others were tuned with "*hyperopt*" (Bergstra et al., 2013). The other hyperparameters were set to their default values. Specifically, *n_estimators*, *objective*, *eval_metric*, and *n_jobs* were set to certain values which can be found in Table 17 in the appendix. The following parameters were tuned with *hyperopt*: *base_score*, *learning_rate*, *max_depth*, *min_child_weight*, *subsample*, *gamma*, *colsample_bytree*, and *scale_pos_weight*. The resulting values for these hyperparameters can also be found in Table 17 in the appendix.

Since XGBoost is a black-box model and, therefore, has low interpretability, SHAP values are reported to provide insight into how the model comes to the output (X. Zhang & Yu, 2024). Figure 22 in the appendix depicts the SHAP values of the twenty variables with the highest SHAP values for the model. It can be interpreted in the following manner. The color represents the value of the variable, so pink indicates a high value and blue indicates a low value. For example, take the variable *calculated_countrycode_nld*. It is a binary variable, so the value is either 0 (low) or 1 (high). In the SHAP plot (Figure 22) it can be seen that the left (negative) side is mainly pink and the right (positive) side is mainly blue. This means that a low value (blue) often leads to a positive prediction and a high value (pink) often leads to a negative prediction. In other words, if this variable has value 0 (low) it is more likely to lead to a fraud prediction (positive), and if the value is 1 (high) it is more likely to lead to a non-fraud prediction (negative).

5 Methods

5.1 Bias Detection

Choosing which method to use to detect bias is highly dependent on the context of the problem at hand. In this project, it is important that the selected methods consider fairness and the imbalanced nature of the data. Methods for detecting bias consider either group fairness or individual fairness. As the name indicates, group fairness compares groups. For example, it considers whether men and women are treated equally. Likewise, individual fairness compares individuals. It is based on the idea that the output of the model should be the same for similar individuals (Genovesi et al., 2024). The methods used answer the following questions: Are individuals in one group more often falsely predicted to commit fraud? Do individuals in one group get away with fraud more often? Is an individual more likely to receive a fraud prediction if they belong to a different demographic?

5.1.1 Group Fairness

Group fairness metrics are used to judge whether different groups are treated equally by the model. There are those that focus on the independence criterion, which requires the model output to be independent of group membership. Other metrics focus on the separation criterion, which takes actual outcomes into account (Pessach & Shmueli, 2022). For this problem, metrics of the latter category are selected. This is because the model should make fair and accurate predictions, but it should not predict that someone will or will not commit fraud solely to equalize the distributions between groups. Accepting people who are not able to pay can lead to further perpetuating existing unfairness instead of achieving fairness (Kozodoi et al., 2022). The precision parity and recall parity are calculated for each sensitive attribute. Both metrics measure differences between two groups. For binary variables, this works straightforward. For continuous variables, the decision was made to split the values into two groups and to calculate the metrics of each group. A threshold is chosen so that the variable is split into two groups of approximately equal sample size (Suárez Ferreira et al., 2025). For these metrices the confusion matrix is used. It is displayed in Table 2, 0 means *no fraud* and 1 means *fraud*. True positives (TP) are *fraud* cases that were predicted as *fraud*, false positives (FP) are *no fraud* cases that got a *fraud* prediction, true negatives (TN) are *no fraud* cases that got a *no fraud* prediction, and false negatives (FN) are *fraud* cases that got a *no fraud* prediction. As there are few actual fraud cases and predicted fraud cases compared to the total sample size, FP, FN, and TP are relatively small and TN will be relatively large.

Precision Parity. Precision is the ratio of true positives to predicted positives: $\frac{TP}{TP+FP}$ (Kamalaruban et al., 2024). Essentially, it is the ratio of correct fraud predictions relative to all fraud predictions. If it is higher for one group,

	Predicted	
Actual	0	1
0	TN	FP
1	FN	TP

Table 2: Confusion matrix

the model is more accurate in predicting fraud for that group than for the other group. A lower value indicates more false fraud predictions relative to correct fraud predictions for the group. The precision parity is the difference in precision between groups. A high precision parity indicates bias because one group is more often falsely accused of fraud than the other group. For a feature with groups $G \in \{0, 1\}$ the precision parity metric is the absolute difference between the precision of each group. For imbalanced data, it is important to normalize it by dividing it by the maximum precision (Kamalaruban et al., 2024):

$$\Delta_{\text{Precision}} = \frac{|\text{Precision}_{G=0} - \text{Precision}_{G=1}|}{\max\{\text{Precision}_{G=0}, \text{Precision}_{G=1}\}}.$$

Recall Parity. Recall is the ratio of true positives to total positives, also known as the true positive rate: $\frac{TP}{TP+FN}$ (Kamalaruban et al., 2024). In words, it is the ratio of how often a fraud case is identified as fraud relative to all fraud cases. If it is lower for a group, people who commit fraud in that group get away relatively more often than people who commit fraud from the group with the higher score. Recall parity is the difference of recall between groups, if this is high, it indicates bias as the groups are judged differently. For a feature with groups G, where $G \in \{0, 1\}$, the recall parity metric is the absolute difference between recalls divided by the maximum recall for normalization:

$$\Delta_{\text{Recall}} = \frac{|\text{Recall}_{G=0} - \text{Recall}_{G=1}|}{\max\{\text{Recall}_{G=0}, \text{Recall}_{G=1}\}}.$$

5.1.2 Individual Fairness

Counterfactual Fairness. The counterfactual fairness technique measures individual fairness by considering whether an individual would receive the same prediction if one of their sensitive attributes were different (Grari et al., 2023). A data point is replaced by a copy where the value of the sensitive attribute is changed. Then, by comparing the model output for the generated input and the original input, fairness can be tested (Krafft et al., 2024). One difficulty is that a change in the sensitive attribute would realistically also cause a change in other attributes, due to dependencies. For example, a change in gender could result in a change in income. Because of this, a causal graph is needed that captures the dependencies between variables. This is then used to create a causal model that determines the influence of changing sensitive attributes on other attributes. Finally, counterfactuals can be generated that have not only

a flipped sensitive attribute, but also adjusted dependent attributes.

The causal Graph can vary per situation. A general example is displayed in Figure 21 (Kusner et al., 2017). It is a directed acyclic graph (DAG) with sensitive attribute A, target variable Y, observed variables X, and unobserved variables U. For example, $A = gender$ could influence $X = income$, which in turn can influence Y, which is the target variable for whether someone commits fraud. U is a set of unknown variables that can influence known variables. The goal is to identify X so that it can be adjusted when generating counterfactuals.

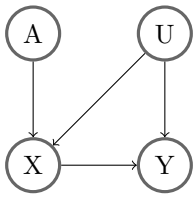

Figure 21: Directed acyclic causal graph

There are too many features to consider all of them when creating the causal graph, so a subset is selected that likely includes X. The graph is created by considering the dependencies between variables in the subset, so if this subset is too large, the computational time would become infeasible (Le et al., 2019). The subset includes the 10 features with the highest SHAP values (Lundberg & Lee, 2017) and the 10 features that correlate the most with the protected attribute, as well as the target variable. Features with high SHAP values are considered because they influence the target variable and might be children of the sensitive attribute. Features that are correlated with the sensitive attribute are selected because, as Liang & Yang (2021) state, correlation does not imply causation, but causation does imply correlation. This means that variables that have a causal relationship with the sensitive attribute are likely also correlated with it. To test correlations between variables, Spearman's rank test is used when both variables are binary or both are continuous, as it does not require the data to be distributed normally (Sedgwick, 2014). When one variable is continuous and the other is binary, the point-biserial correlation coefficient is calculated, as it is specifically suited for when one variable is binary and the other continuous (Kornbrot, 2014).

A causal discovery algorithm is selected to generate the causal graph based on the selected variables. Most of these algorithms work on a single data type. However, in this case, the variables can be continuous or binary and are not necessarily normally distributed. A fitting causal discovery algorithm for this situation is the so called latent-PC algorithm, introduced by Cai et al. (2022), using Fast Causal Inference (FCI) instead of Peter-Clark (PC).

The PC algorithm (Spirtes et al., 2000) is a constraint-based approach to causal discovery. It starts from a complete undirected graph, then removes edges recursively to obtain the skeleton, a partially connected undirected graph.

The PC algorithm does this by using conditional dependencies, inferred from partial correlation between two variables given a third variable with Fisher's z-transformation at the significance level α (Kalisch & Bühlmann, 2007):

$$Z(i, j|k) = \frac{1}{2} \log \left(\frac{1 + \hat{\rho}_{i,j|k}}{1 - \hat{\rho}_{i,j|k}} \right). \tag{1}$$

Conditional independence is concluded if the null hypothesis of a zero correlation coefficient is not rejected. Then, orientation rules are applied to direct as many edges as possible, resulting in a completed partially directed acyclic graph. The PC algorithm assumes the following (Glymour et al., 2019):

- **Causal Sufficiency**: There are no hidden confounders, meaning that there are no unobserved variables (U in Figure 21) influencing observed variables.
- **Causal Markov Condition**: Every variable is independent of its non-descendants given its parents, so if a variable's parents is known, its non-descendants add no additional information. Let V be the set of vertices in the causal graph. Then for every W in V, W is independent of $V \backslash (Descendants(W) \cup Parents(W))$ given $Parents(W)$. Parents of variable W are variables that cause W. Its descendents are variables that are caused by W, and non-descendents are other variables.
- **Faithfullness assumption**: The only conditional independencies in the data are those that are implied by d-separation in the true DAG. If two variables are d-separated, they are conditionally independent given a third variable. For example, if A and C are d-seperated, then $A \perp\!\!\!\perp C \mid B$. This is the case if $A \rightarrow B \rightarrow C$ or $A \leftarrow B \rightarrow C$, because if B is known, then A does not provide additional information on C. If $A \rightarrow B \leftarrow C$, A and C are not d-separated, because if B is known, A and C both become more likely, indicating dependence.

Fast Causal Inference (FCI) (Spirtes et al., 2000) works similar to the PC algorithm, however, it can deal with latent confounders by not assuming every edge is directed one way or another. In other words, FCI does not assume causal sufficiency. Using the PC algorithm while latent confounders are present can lead to arrows pointing in inaccurate directions in the causal graph. FCI and PC create the skeleton in the same manner, which means that FCI can be used instead of PC in the latent-PC algorithm. The possibility of using FCI instead of PC in this algorithm is also mentioned by its creators (Cai et al., 2022). Therefore, FCI is used in this project.

Conditional independencies are used in PC and FCI. They can be inferred from partial correlations in the Gaussian case. Assuming that the random vector X follows a multivariate normal distribution, $X^{(i)}$ and $X^{(j)}$ are conditionally independent given $X^{(k)}$, if $H_0 : \rho_{i,j|k} = 0$ is not rejected, where $\rho_{i,j|k}$ denotes the partial correlation between i and j given k and $i \neq j \in \{1, ..., p\}, k \subseteq$

$\{1, ..., p\} \backslash \{i, j\}$. This is an elementary property of the multivariate normal distribution (Kalisch & Bühlmann, 2007). The problem is that the data for this project do not necessarily follow a normal distribution. The latent-PC algorithm (Cai et al., 2022) is based on the mixed latent copula model, which relaxes this Gaussian assumption in the data and is applicable to continuous, binary, and ordinal data.

Before explaining the algorithm itself, Sklar's theorem should be introduced to gain a better understanding of how copula's work (Jaworski et al., 2010). The theorem states that a d-dimensional distribution function F can be expressed in terms of its univariate margins $F_1, F_2, ..., F_d$ and a copula C, which represents its dependency structure:

$$F(x_1, x_2, ..., x_d) = C(F_1(x_1), F_2(x_2), ..., F_d(x_d)).$$

The mixed latent copula model is built on the Gaussian copula, which assumes the joint normality of transformed variables. In essence, a Gaussian copula creates a joint distribution with a conditional dependence structure that resembles the conditional dependence structure of a multivariate normal distribution, while allowing the marginal distributions to remain arbitrary. The Gaussian copula with parameters μ and Σ is (H. Liu et al., 2009):

$$C(u_1, ..., u_d) = \Phi_{\mu,\Sigma}(\Phi^{-1}(u_1), ..., \Phi^{-1}(u_d)),$$

where $\Phi_{\mu,\Sigma}$ is the multivariate Gaussian cdf and Φ is the univariate standard Gaussian cdf. A continuous random vector $X = \{X_1, ..., X_d\}^T \in \mathbb{R}^d$ follows a Gaussian copula model if there exists a set of monotonically increasing transformations $g = \{g_j\}_{j=1}^d$, such that the transformed variables follow a multivariate normal distribution: $\{g_1(X_1), ..., g_d(X_d)\}^T \sim N_d(0, \Sigma)$ with correlation matrix Σ. Then X can be denoted as $X \sim GC(\Sigma, g)$. A continuous random variable can be transformed to the standard normal distribution using $g = \Phi^{-1}(F_j(x))$, where Φ^{-1} is the quantile function for the standard normal distribution and $F_j(x)$ is the distribution function of X_j. $F_j(x)$ can be estimated with the marginal empirical distribution function $\hat{F}_j(x)$ (H. Liu et al., 2009):

$$\hat{F}_j(t) = \frac{1}{n} \Sigma_{i=1}^n \mathbf{1}_{\{X_j^{(i)} \leq t\}}$$

In the binary case, it is assumed that there is a latent continuous Gaussian vector Z behind the observed binary vector X. The binary variable is obtained by dichotomizing the latent continuous vector Z at a vector of unknown constants C. A binary random vector $X \in \mathbb{R}^d$ follows a binary latent Gaussian copula model if there exists a d-dimensional latent vector $Z \sim GC(\Sigma, g)$ such that $X_j = I(Z_j > C_j), \forall j = 1, ..., d$ where $I(\cdot)$ is the indicator function. The binary latent Gaussian copula is denoted as $X \sim BCG(\Sigma, g, C)$ with latent vector $Z \sim GC(\Sigma, g)$. C_j can be inferred as follows:

$$p_j = \mathbb{P}[X_j = 1] = \mathbb{P}(Z_j > C_j) = 1 - \mathbb{P}(Z_j \leq C_j) = 1 - \Phi(C_j)$$

$$1 - p_j = \Phi(C_j)$$
$$C_j = \Phi^{-1}(1 - p_j)$$

Because $p_j = \mathbb{P}[X_j = 1]$, p_j can be estimated by taking the mean. Now that C_j is estimated, X_j can be replaced with latent continuous variable Z_j. Z_j is sampled from $\mathcal{N}(0, 1)$ truncated to $[-\infty, C_j]$ if $X_j = 0$ and from $\mathcal{N}(0, 1)$ truncated to $[C_j, \infty]$ if $X_j = 1$. Now that the original data are transformed to follow a standard normal distribution, it can be used to generate the causal graph with the FCI algorithm, using *causal-learn* in Python (Zheng et al., 2024). As mentioned, FCI uses conditional independencies, which can be inferred from partial correlations in the Gaussian case. Therefore, the algorithm requires normally distributed data, so that it can determine the conditional dependencies. If the data are not normal, the dependency tests become less reliable. Only edges from A to X to Y and edges from other parents of X to X are kept, as these are the only ones necessary for the causal model.

The Causal model is created with *DoWhy* in Python. An Invertible Structural Causal Model is chosen because it can be used to generate counterfactuals. It is a Structural Causal Model (SCM) that requires the underlying causal mechanisms to be invertible with respect to noise. This is because the noise is needed to generate counterfactuals, so it has to be possible to extract it. The SCM consists of a set of causal mechanisms that depend on the data type of each node:

- Root node: An empirical distribution. The distribution is represented by random sampling with replacement from the provided data.
- Non-root node and continuous data: Additive Noise Models (ANM) of the form $X_i = f(pa(X_i)) + u_i$, where $pa(X_i)$ are the parents of X_i and the unobserved noise u_i is assumed to be independent of $pa(X_i)$. It is invertible with respect to the noise, so $u_i = X_i - f_i(pa(X_i))$. The regression model with the smallest mean squared error is selected for f (Hoyer et al., 2008).
- Non-root node and discrete data: Discrete Additive Noise Models are defined the same as non-discrete ANMs, but with an added constraint to return discrete values (Peters et al., 2011).

Once the causal mechanisms are assigned, they are fitted and the counterfactual samples can be generated.

Generating counterfactuals first requires the sensitive attribute to be flipped. Then, with the causal model, the observed variables influenced by the sensitive attribute can be adjusted accordingly. For binary variables, the value is simply flipped and for continuous variables, the value is flipped to the opposite quantile. For example, if x is in the 10%-quantile, x' will be flipped to the $100 - 10 = 90$%-quantile. Then the causal model changes other variables accordingly, by using the casual mechanisms that were assigned. The fraud

prediction model then predicts for the generated counterfactual data as well as the original data whether the individuals will commit fraud. The predictions are then compared by looking at for how many individuals the outcome changed when the sensitive attribute was changed.

5.1.3 Significance testing

The threshold for bias significance is set to $\alpha = 0.05$ (Kamalaruban et al., 2024). For the parity metrics, the ideal value is 0. Fisher's exact test is used to measure the p-value of the parities. This test is chosen because it handles imbalanced data well (de Lima Cabral & Barros, 2018). If the resulting p-value is below $\alpha = 0.05$, the null hypothesis H_0 : "There is no association between the two variables" is rejected. This means that the metric is not the same across groups, indicating possible bias. Fisher's exact test is made for binary variables, as it uses a 2×2 contingency table. However, in this case it can be used for continuous variables as well, since they are split into two groups. Important to note is that because Fisher's exact test uses a contingency table with counts, it means that it does not calculate the significance of the parities directly. It does answer a similar question. For example, recall parity indicates whether people who commit fraud in one group are more likely to receive a positive fraud prediction than in the other group. Similarly, Fisher's exact test can ask, of the people who have committed fraud, whether the odds of receiving a *fraud* prediction are the same for both groups. The contingency table then contains the two groups in the rows and the true positives and false negatives in the columns. The same can be done for the precision parity by having true positives and false positives in the columns.

Determining whether the model is biased against individuals from certain groups according to the counterfactual fairness test requires considering the flip rate. It measures the ratio of how often the prediction of fraud changed when the sensitive attribute changed. If the prediction changes from *no fraud* to *fraud*, the model is possibly biased in favor against the group this person is part of. If the prediction changes from *fraud* to *no fraud*, the model is possibly biased against the group to which this person originally belonged. It is important to remember that because most people do not commit fraud, most predictions will be and remain *no fraud* and flip rates are expected to be low. A flip rate of 0 implies that the predictions did not change based on a change in the sensitive attribute, which is ideal, as it indicates no bias. Several flip rates are considered for each group:

- How many predictions changed relative to the group's sample size
- How many predictions changed from *no fraud* to *fraud* relative to the group's sample size
- How many predictions changed from *fraud* to *no fraud* relative to the group's sample size

- How many predictions changed from *fraud* to *no fraud* relative to the original number of *fraud* predictions in the group
- How many predictions changed from *no fraud* to *fraud* relative to the counterfactual number of *fraud* predictions in the group

The first three rates are expected to be almost 0, as the sample size is large and there are few *fraud* predictions. The rates that measure how many predictions were flipped relative to the number of *fraud* predictions should be more interpretable. Therefore, this rate is considered first, and the other rates can be used to provide context.

For each variable, the metrics discussed are considered when determining whether the model contains bias against a certain group. It is not based on only one of them.

5.2 Bias Mitigation

Fraud data are highly imbalanced, which can lead to biases in the model (Dastile et al., 2020). Therefore, this project aims to mitigate bias with methods that consider data imbalance. Three techniques are chosen: one pre-processing technique, one in-processing technique, and one post-processing technique. They are combined into two methods. The first method uses a data balancing technique that balances classes as well as groups, after which threshold optimization is applied. The second method evaluates the model during training with the area under the precision recall curve (AUPRC) instead of with the traditional AUC and applies threshold optimization afterwards.

5.2.1 Method 1: Fair-SMOTE

Pre-processing techniques attempt to mitigate bias by removing underlying unfairness in the training data (Mehrabi et al., 2021). Trinh & Zhang (2024) state that synthetic data generation, which is a technique that generates synthetic samples that resemble the original data, can significantly improve fairness without sacrificing too much accuracy. For example, data balancing techniques that oversample the minority class do this. In this project, there is not only class imbalance, but also group imbalance. Class imbalance is when the target variable is imbalanced, and group imbalance is when groups within a feature are imbalanced. Therefore, a technique called Fair-SMOTE, introduced by Chakraborty et al. (2021), will be used. The data is balanced in such a way that privileged and unprivileged groups have the same amount of positive and negative samples so that every subgroup will have the same size. One risk is that, for small subgroups, most of the data in that group will be synthetic after balancing. Then the model fits primarily on the synthetic data instead of the original data and may overfit to synthetic patterns instead of reflecting the true data distribution. After the model is trained on the balanced data, the model is used to predict whether potential customers will commit fraud. The default threshold for a

fraud prediction is 0.5, which means that the model predicts fraud if the probability is above the threshold of 0.5, and no fraud if it is below the threshold. Leevy et al. (2023) found that this default threshold works well for balanced data, but can yield poor performance for imbalanced data. They describe a method to identify the optimal threshold that maximizes a user specified performance metric. In this project, a threshold is found that requires the recall to be at least as high as in the original model. Specifically, the highest threshold that still achieves the required recall is chosen. This is because a higher threshold would lead to a lower recall and fewer fraud cases would be caught. A lower threshold would lead to a higher recall and likely to a lower precision, so there would be more false fraud predictions.

Fair-SMOTE is an extension of SMOTE. SMOTE works as follows: let X be a data point in the minority class with attributes $x_1, x_2, ..., x_n$ and with nearest neighbor $X'(x'_1, x'_2, ..., x'_n)$. Then, a new data point $Y(y_1, y_2, ..., y_n)$ is generated with the following formula (Chakraborty et al., 2021):

$$Y = X + rand(0, 1) \times (X - X').$$

In this formula, a new data point is generated by copying X and adding the difference between X and X' at random. Every X is used to generate a new data point a number of times until the required number of data points is generated. If the number of data points that must be generated is less than the number of data points in the minority class, X are picked at random (Chawla et al., 2002).

SMOTE balances the minority and majority classes, but can exacerbate group imbalance. Fair-SMOTE deals with this by not only balancing classes but also balancing groups. For example, the variable *calculated_countrycode_nld* has a large group imbalance, where 10% do not have a Dutch identification document (ID) and 90% do have it. The variable also has a large class imbalance, as 10% of people without a Dutch ID have committed fraud and 2.1% of people with a Dutch ID have committed fraud. There are then four subgroups: people with a Dutch ID that have not committed fraud, people with a Dutch ID that have committed fraud, people without a Dutch ID that have not committed fraud, and people without a Dutch ID that have committed fraud. SMOTE only balances the fraud cases, regardless of the groups. Fair-SMOTE balances the data so that all four groups become of equal size.

In this project, there are multiple protected attributes that should be balanced. Chakraborty et al. (2021) handles multiple attributes by balancing the training data with respect to class and protected attributes. The number of subgroups increases with the number of protected attributes. For one protected attribute, there are $2 \times 2 = 4$ subgroups, for two protected attributes there are $2 \times 2 \times 2 = 8$ subgroups, etc. A higher number of subgroups requires more synthetic data, which could potentially result in lower accuracy. Continuous sensitive attributes are split into two groups so that the number of subgroups does not explode.

After dividing the data into subgroups, Fair-SMOTE generates new data

points for all subgroups so that they all end up with a size equal to the subgroup with the maximum number of data points. For data generation, two hyperparameters are used. Mutation amount (f) denotes the rate at which the generated sample will differ from the parent, and crossover frequency (cr) denotes how different the data point will be from the parent. They are both in the range [0,1]. They are both set to 0.8, since this gave Chakraborty et al. (2021) the best results.

Chakraborty et al. (2021) describes the algorithm as follows. It starts by randomly selecting a parent point (p) from a subgroup. Then, using K-nearest neighbor (KNN), two data points (c_1, c_2) closest to p are selected. In order to use KNN, numerical data is scaled and categorical variables are ignored. A new datapoint is generated for a subgroup based on p, c_1, and c_2. The class and protected attributes are assigned the values that correspond to the subgroup. For the other attributes, the value depends on the data type. For a binary attribute b, the new value is randomly copied from $p(b), c_1(b)$, or $c_2(b)$ with probability cr, and copied from $p(b)$ with probability $1 - cr$. For a categorical attribute c, the new value is randomly copied from $p(c), c_1(c)$, or $c_2(c)$. Finally, for a numerical attribute n, the new value is calculated by $p(n)+f*(c_1(n)-c_2(n))$ with probability cr and copied from $p(n)$ with probability $1 - cr$.

K-nearest neighbor. KNN assigns the class of the majority of its K nearest neighbors to the input feature vector $\mathbf{x}$ (Dastile et al., 2020). The nearest neighbors are found by calculating the Euclidean distance between the input feature vector $\mathbf{x}$ and the training dataset $\{\mathbf{x}_k\}_{k=1}^{m}$. In this project, *NearestNeighbours* in *scikit-learn* is used in Python (Buitinck et al., 2013).

5.2.2 Method 2: AUPRC

In-processing techniques attempt to mitigate bias by removing unfairness during model training (Mehrabi et al., 2021). The current model is trained with AUC as an evaluation metric. However, for imbalanced data, the AUC can be misleading and the AUPRC is a good alternative (Hancock et al., 2023). The AUPRC plots precision and recall scores, while the AUC plots the recall and the false positive rate (FPR). The FPR involves true negatives, and due to its relatively large size, false positives can be drowned out. The precision does not incorporate true negatives, which makes it more appropriate than the FPR in this context. Therefore, the AUPRC is used to evaluate the model during training. This means that during training, the model with the best AUPRC is selected instead of the model with the best AUC. The objective function remains the same, only the metric with which the model is evaluated during training is changed. The AUPRC is not necessarily expected to have any advantage over the AUC for balanced data, so this method is only applied to the original imbalanced training data. Additionally, after training, threshold optimization is applied. It is done in the same manner as for method 1. The threshold is again determined by finding the highest threshold that still achieves a recall that is at least as high as in the original model.

5.2.3 Model training

Two new models are trained based on the two methods. The first method changes the training data, and the second method changes the manner in which the model is trained. Both are XGBoost models, where the hyperparameters are tuned with *hyperopt* (Bergstra et al., 2013). The training data is split into a training set (80%) and validation set (20%). For method 1, the training set is balanced using Fair-SMOTE and the validation set is from the original training dataset so that it has the same distribution as the test set. The same features as in the original model are used and the categorical ones are one hot encoded before training.

5.2.4 Evaluation

The mitigation methods are evaluated based on accuracy and fairness relative to the original method. Fairness is measured by considering recall parity and precision parity and their corresponding p-values. The flip rates of the features with a significant recall parity or precision parity are considered to further investigate fairness. Accuracy is measured with the following metrics: AUC, AUPRC, F1-score, MCC, and G-mean.

AUC: The AUC is chosen because it is used as an evaluation metric in the original model. It is calculated by plotting the TPR and FPR. The value ranges in the interval [0.5,1], where 0.5 is equivalent to random guessing and 1 indicates perfect predictions (Naidu et al., 2023). The TPR and FPR can be calculated as follows (Hancock et al., 2023):

$$TPR = \frac{TP}{TP + FN}$$

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

AUPRC: Hancock et al. (2023) found that AUPRC is a good alternative to the AUC when data are imbalanced. It is calculated by plotting the precision and recall scores. The value ranges from 0 to 1, where 1 indicates perfect predictions. The recall is the same as the TPR and the precision is calculated as follows (Hancock et al., 2023):

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

F1-Score: The F1-score, which balances recall and precision, is calculated as follows (de la Cruz Huayanay et al., 2024):

$$F1 = \frac{2 \times TP}{2 \times TP + FP + FN}.$$

Ideally, this value is 1, which indicates perfect recall and precision. In the worst case, it is 0. de la Cruz Huayanay et al. (2024) and Chicco & Jurman

(2020) found that the F1-score performs well in the case where the negative class is the majority class.

MCC: The MCC metric uses all four values of the confusion matrix. It is appropriate for imbalanced data, since it only generates a high score if the majority of positive and negative instances were predicted correctly. It can be calculated with this formula (Chicco & Jurman, 2020):

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}.$$

The value ranges in the interval [-1,1], where -1 indicates perfect misclassification, 0 equals random guessing, and 1 indicates perfect classification. de la Cruz Huayanay et al. (2024) and Chicco & Jurman (2020) found that the MCC performs well when the negative class is the majority class.

G-mean: The geometric mean quantifies the classification performances of both majority and minority classes (de la Cruz Huayanay et al., 2024). It is calculated as follows:

$$GM = \sqrt{\frac{TN}{FP + TN} \times \frac{TP}{TP + FN}}.$$

Its value lies in the interval [0,1], where 0 indicates perfect misclassification and 1 indicates perfect classification. Chicco & Jurman (2020) found that the G-mean performs well for imbalanced data where the negative class is the majority class.

6 Results

Where relevant, a significance rate of $\alpha = 0.05$ and random seed = 0 is used.

6.1 Detecting bias

The features that are considered sensitive attributes in this project are explained in Table 1. The features that could be discriminatory to location, age, gender, and nationality were selected. Sometimes, when almost everyone had identical values, splitting the feature into groups would result in one group containing almost no samples. The features for which this was the case were not included, as there was then simply not enough data for the minority groups to create reliable results.

The data consisted of binary and continuous variables, as categorical ones were one-hot-encoded. Binary variables naturally have a group 0 and a group 1, and continuous variables were split at a certain threshold T to create two groups. This threshold was chosen so that the sample sizes were approximately equal, but identical values would be in the same group. This resulted in the following thresholds: $T_{calculated\_age} = 39.7$, $T_{focumcheck\_statpc4index} = 100$,

$T_{focumcheck\_statpc5index} = 99$, and $T_{focumcheck\_statpc6index} = 90$. Group 0 contains samples with values that are $\leq T$ and group 1 contains samples with values that are $> T$. *calculated_age* is divided into younger and older individuals. *focumcheck_statpc4index*, *focumcheck_statpc5index*, and *focumcheck_statpc6index* are divided into lower and higher risk scores (based on postcode).

6.1.1 Group Fairness

For each variable in Table 1, the recall parity and precision parity were calculated, as well as corresponding p-values. The parities can be found in Table 3 and the p-values in Table 4. P-values below 0.05 are highlighted, as those are significant and indicate bias. Additionally, the recall, precision, and group sizes for each separate group are presented in Table 5. The group with the highest recall and lowest precision is highlighted, as that could indicate bias against that group. Important to consider is that groups with a much larger sample size are generally more accurately classified, since the model had more training data for these groups, and thus are expected to have a higher recall and precision.

Feature	Recall parity	Precision parity
calculated_countrycode_nld	0.2477	0.4002
calculated_age	0.0318	0.5212
gender	0.1348	0.1389
focumcheck_statpc4index	0.4945	0.1242
focumcheck_statpc5index	0.0381	0.2999
focumcheck_statpc6index	0.2186	0.1633
focumcheck_debtresult_d02	0.0429	0.2816
focumcheck_debtresult_d03	0.0410	0.2697
focumcheck_debtresult_d99	0.1603	0.3526

Table 3: Parities of each sensitive attribute

Feature	Recall p-value	Precision p-value
calculated_countrycode_nld	0.2038	0.0054
calculated_age	0.9045	0.0001
gender	0.5473	0.4796
focumcheck_statpc4index	0.0077	0.6196
focumcheck_statpc5index	0.9025	0.0700
focumcheck_statpc6index	0.3280	0.4109
focumcheck_debtresult_d02	1	0.3817
focumcheck_debtresult_d03	0.9091	0.0888
focumcheck_debtresult_d99	0.4639	0.0186

Table 4: P-values for each sensitive attribute

It can be seen in Table 3 that for most features, the precision parity is higher than the recall parity. In Table 5 it can be seen that the recalls themselves are

low, which explains the low differences. Because the parities are normalized, they are not as small as the absolute differences would be. These low recalls mean that there are many false negatives with respect to true positives, indicating that many fraud cases are not caught.

In Table 4 it can be seen that not many p-values are significant for recall and precision. *focumcheck_statpc4index* has the highest recall parity and is the only variable with a significant p-value for the recall parity. The three variables with the highest precision parities, namely *calculated_countrycode_nld*, *calculated_age*, and *focumcheck_debtresult_d99*, are the only ones with a significant p-value for precision.

	Group 0			Group 1		
Feature	Recall	Precision	size	Recall	Precision	size
calculated_countrycode_nld	0.0915	0.2772	4536	0.0688	0.4622	38751
calculated_age	0.0742	0.3086	21649	0.0767	0.6444	21638
gender	0.0824	0.3444	17988	0.0713	0.4	25299
focumcheck_statpc4index	0.0457	0.34	21811	0.0904	0.3882	21384
focumcheck_statpc5index	0.0733	0.4902	21748	0.0762	0.3432	21447
focumcheck_statpc6index	0.0632	0.4314	21759	0.0809	0.3609	21436
focumcheck_debtresult_d02	0.0754	0.3897	41883	0.0722	0.2800	1404
focumcheck_debtresult_d03	0.0768	0.4494	24748	0.0736	0.3282	18539
focumcheck_debtresult_d99	0.0708	0.3292	20583	0.0842	0.5085	22704

Table 5: Metrics per group

6.1.2 Individual Fairness

The counterfactual fairness results are interpreted using the flip rates (FR). Sampling from the truncated normal distribution to estimate latent continuous variables Z introduces randomness. However, by fixing the random seed to 0 the results are reproducible. The assignment of causal mechanisms to each variable in the causal graph also uses randomness in selecting the best model. However, there is no option to control this randomness as it happens inside the function. It leads to limited changes in the results and it takes quite some time to assign causal mechanisms, so the decision was made to use a representative run for the results. The following flip rates for groups G, where $G \in \{0, 1\}$, are shown in Tables 6 and 7 respectively:

$$FR1_G = \frac{\text{\#changed predictions in G}}{\text{sample size G}}$$

$$FR2_G = \frac{\text{\#no fraud to fraud flips in G}}{\text{sample size G}}$$

$$FR3_G = \frac{\text{\#fraud to no fraud flips in G}}{\text{sample size G}}$$

$$FR4_G = \frac{\text{\#fraud to no fraud flips in G}}{\text{\#fraud predictions observed data in G}}$$

$$FR5_G = \frac{\text{\#no fraud to fraud flips in G}}{\text{\#fraud predictions counterfactual data in G}}$$

	Group 0				
Feature	$FR1$	$FR2$	$FR3$	$FR4$	$FR5$
calculated_countrycode_nld	0.0205	0.0002	0.0203	0.9109	0.1000
calculated_age	0.0073	0.0004	0.0069	0.8514	0.2353
gender	0.0049	0.0026	0.0023	0.4667	0.4894
focumcheck_statpc4index	0.0032	0.0021	0.0011	0.4600	0.6301
focumcheck_statpc5index	0.0039	0.0028	0.0011	0.4706	0.6932
focumcheck_statpc6index	0.0051	0.0045	0.0006	0.2549	0.7185
focumcheck_debtresult_d02	0.0041	0.0000	0.0040	0.8667	0.0370
focumcheck_debtresult_d03	0.0035	0.0000	0.0035	0.9663	0.0000
focumcheck_debtresult_d99	0.0073	0.0067	0.0006	0.0807	0.4825

Table 6: Flip rates group 0

	Group 1				
Feature	$FR1$	$FR2$	$FR3$	$FR4$	$FR5$
calculated_countrycode_nld	0.0061	0.0048	0.0013	0.4370	0.7352
calculated_age	0.0039	0.0036	0.0003	0.1556	0.6607
gender	0.0017	0.0004	0.0012	0.2385	0.1000
focumcheck_statpc4index	0.0075	0.0028	0.0047	0.5941	0.4609
focumcheck_statpc5index	0.0063	0.0012	0.0051	0.6450	0.3023
focumcheck_statpc6index	0.0057	0.0013	0.0045	0.5680	0.2700
focumcheck_debtresult_d02	0.0171	0.0000	0.0171	0.9600	0.0000
focumcheck_debtresult_d03	0.0070	0.0000	0.0070	0.9847	0.0000
focumcheck_debtresult_d99	0.0005	0.0004	0.0002	0.0678	0.1270

Table 7: Flip rates group 1

These flip rates can be interpreted in the following manner: a high $FR2$ or $FR5$ indicates bias against the opposite group, because then the model changed its prediction from *no fraud* to *fraud* when the sample changed to the other group. In contrast, a high $FR3$ or $FR4$ indicates bias against the group itself, because then the model initially predicted *fraud* and changed to *no fraud* when the sample changed to the other group. Furthermore, each FR can be compared between groups. If the FR is similar for both groups, it does not necessarily indicate bias even if the rate is high, because then, even though the predictions change, it does not happen specifically because of the group the individual belongs to. Important to keep in mind is that if a variable that has a large influence on the output of the model changes, it makes sense if there are many

flips. Figure 22 in the appendix shows the 20 variables on which the model relies the most to make its prediction. Additionally, in several cases one group simply does commit fraud more often, which can then explain large differences. The accuracy of the flips cannot be confirmed since the counterfactual samples are all artificial, so they have no ground truth. Essentially, large flip rates or large differences might be explained by class imbalance, so they do not necessarily confirm bias on their own. However, they can still be used as an indication.

6.1.3 Features

The results are analyzed by feature to determine whether they contain bias.

Calculated_countrycode_nld. This feature specifies whether someone has a Dutch identification document, not necessarily whether they have a Dutch nationality. The feature has a high imbalance in group sizes. Only 10% of the individuals do not have a Dutch identification document (group 0), and 90% of the individuals do have it (group 1). Furthermore, in the training data, 10.3% of group 0 commits fraud and only 2.1% of group 1 commits fraud. This could explain the significant precision parity, where group 0 (0.2772) has a lower precision than group 1 (0.4622), as can be seen in Table 5. This means that group 0 has more false fraud predictions relative to true fraud predictions compared to group 1. Looking at the recall, it can be seen that group 0 (0.0915) has a higher recall than group 1 (0.0688). The model does not often identify the fraud cases as fraud, however, it does detect fraud slightly more often for group 0. Together, this indicates bias against group 0, as more people in group 1 get away with fraud (lower recall), and more people in group 0 receive false fraud predictions (lower precision). The precision p-value is significant, which can be seen in Table 4. The p-value for recall is not significant, however, the recall parity itself is relatively high.

The flip rates shown in Tables 6 and 7 suggest a bias against group 0. $FR2_0 = 0.0002 < FR2_1 = 0.0048$ and $FR5_0 = 0.1 < FR5_1 = 0.7352$ indicate that the model changes its prediction to *fraud* more often for samples that change from group 1 to group 0 than for samples that change from group 0 to group 1. $FR3_0 = 0.0203 > FR3_1 = 0.0013$ and $FR4_0 = 0.9109 > FR4_1 = 0.437$ indicate that the model changes its prediction to *no fraud* relatively often when a sample changes from group 0 to group 1 and less often when a sample changes from group 1 to group 0. The differences between $FR4_0$ and $FR4_1$ and between $FR5_0$ and $FR5_1$ are also large. This further indicates bias, as the output of the model highly depends on the value of this feature. The high rates can be partly attributed to the fact that the variable has a relatively high SHAP value (Figure 22), so the model relies greatly on its value to make a prediction. Moreover, compared to group 1, group 0 has a much higher ratio of fraud cases, making it logical that there are many flips in favor of group 1.

Considering all this, it can be said that the model contains bias against the variable *calculated_countrycode_nld*, presumably due to the large difference in group sizes and the disparities in fraud cases per group.

Calculated_age. For this feature, group 0 includes individuals who are 39.7 years of age or younger, and group 1 contains all individuals with ages over 39.7. In Tables 3 and 4 it can be seen that the precision parity (0.5212) is high and the p-value (0.0001) indicates significance. Of the features tested, *calculated_age* has the highest precision parity. Interestingly, the recall parity is the lowest at 0.0318. This means that the model catches fraud cases at approximately the same rate for both groups, but one group recieves much more false fraud predictions than the other. Table 5 shows that group 0 (0.3086) has a lower precision than group 1 (0.6444), thus the younger group receives more false fraud predictions than the older group.

Tables 6 and 7 show that $FR2_0 = 0.0004 < FR2_1 = 0.0036$ and $FR5_0 = 0.2353 < FR5_1 = 0.6607$, indicating bias against group 0. $FR3_0 = 0.0069 > FR3_1 = 0.0003$ and $FR4_0 = 0.8514 > FR4_1 = 0.1556$ also indicate bias against group 0. The model flips its predictions more frequently to *fraud* if an individual changes from group 1 to group 0, and to *no fraud* if an individual changes from group 0 to group 1. The large differences between $FR4_0$ and $FR5_0$ and between $FR5_1$ and $FR4_1$ further indicate bias. The group sizes are approximately equal, however, in the training data 3.9% of group 0 committed fraud and 1.8% of group 1 committed fraud. This could have resulted in group 0 receiving relatively more false fraud predictions than group 1. However, it is not such a large disparity that it fully excuses the high flip rate differences. Another reason for the high flip rates could be that *calculated_age* has a relatively high SHAP value (Figure 22) so it is logical that a change in value can lead to a change in prediction.

From this, it can be concluded that the model contains bias against the variable *calculated_age*, specifically against the younger group.

Gender. Of the examined features, only *gender* was not used in model training. Group 1 consists of men and group 0 of women and unknowns (the latter being only 0.03% of the training data). In Tables 3 and 4 it can be seen that the parities are relatively low and the p-values of the metrics are not significant. Table 5 shows that there is a slight imbalance in group sizes, where 42.8% of the samples are in group 0 and 57.2% are in group 1. Group 0 has a recall of 0.0824, which is slightly higher than that of group 1 at 0.0767, which means that group 1 gets away with fraud slightly more often than group 0. Group 0 (0.3444) has a lower precision than group 1 (0.4), so group 0 receives somewhat more false fraud predictions than group 1. Since the differences are relatively small, the model does not contain a significant bias against this variable.

Table 7 shows that $FR4_1 = 0.2385$ and $FR5_1 = 0.1$ are relatively small, indicating that the model does not change its prediction often for individuals in group 1. Additionally, Table 6 shows that $FR4_0 = 0.4667$ and $FR5_0 = 0.4894$ are close together, indicating that the model does change predictions for individuals in group 0, but it does not necessarily favor one group over the other.

All in all, it can be said that the model is not significantly biased against the variable *gender*.

Focumcheck_statpc4index. This variable represents a risk score based on an individual's 4 position zip code (e.g., 1234). The values range from 0 to 402, where a higher score indicates a higher risk. Group 0 includes individuals with scores up to and including 100 and group 1 includes individuals with scores higher than 100. Interestingly, of the measured variables, this one has the highest recall parity (0.4945) and the lowest precision parity (0.1242), which can be seen in Table 3. This means that the groups receive a similar ratio of false fraud predictions, but the model is much better at detecting fraud cases for one group than for the other. The recall parity is significant according to the corresponding p-value. In Table 5 it can be seen that the recall of group 0 is lower, so they get away with fraud more often. Interestingly, group 0 also has a lower precision, indicating that they receive slightly more false fraud predictions. Together, it means that the model is somewhat more accurate for group 1 and that there is a slight bias against group 0.

Tables 6 and 7 show that $FR2_0 = 0.0021 < FR2_1 = 0.0028$ and $FR5_0 = 0.6301 > FR5_1 = 0.4609$. This is contradictory, because it means that for group 1 relatively more predictions changed from *no fraud* to *fraud* when the group changed than for group 0. At the same time, of the counterfactual fraud predictions per group, group 0 contained more predictions that flipped to *fraud* when the group changed to group 1 than when the group changed from 1 to 0. This does not point to bias against a specific group, since the flips are close enough for the metrics to contradict each other. $FR3_0 = 0.0011 < FR3_1 = 0.0047$ and $FR4_0 = 0.46 < FR4_1 = 0.5941$ indicate a bias against group 1, as the prediction changed to *no fraud* somewhat more often for samples that changed from group 1 to group 0 than the other way around. It is logical that the model predicts *fraud* somewhat more often for group 1 than for group 0, since in the training data, 1.9% of group 0 and 4.1% of group 1 were fraud cases.

Although there are indications that the model is biased against *focumcheck_statpc4index*, it is limited and does not favor one group much. The only real indication of bias is the significant recall parity.

Focumcheck_statpc5index. Similarly to the previous variable, this one represents a risk score based on an individual's 5 position zip code (e.g., 1234A). The values range from 0 to 1157, where group 0 includes individuals with a score up to and including 99, and group 1 includes those with a score above 99. As can be seen in Tables 3 and 4, the recall parity is low at 0.0381, so fraud cases are caught at a similar rate in the groups. The precision parity is relatively high (0.2999), however, according to the corresponding p-value (0.07) it is not significant. The precision of group 1 (0.3432) is lower than the precision of group 0 (0.4902), which would indicate bias against group 1, as this group receives more false fraud predictions.

Tables 6 and 7 indicate bias against group 1. $FR2_0 = 0.0028 > FR2_1 = 0.0012$ and $FR5_0 = 0.6932 > FR5_1 = 0.3023$ indicate that predictions flipped to *fraud* more frequently when a sample changed from group 0 to group 1 than the other way around. $FR3_0 = 0.0011 < FR3_1 = 0.0051$ and $FR4_0 =$

$0.4706 < FR4_1 = 0.6450$ also indicate bias against group 1, because the prediction changed to *no fraud* more often when a sample changed from group 1 to group 0 than the other way around. The flip rates indicate a slight bias against group 1, however, 1.8% of group 0 committed fraud in the training data and 4.2% of group 1, which could explain the differences in flip rates.

The precision parity and flip rates indicate a slight bias against the feature *focumcheck_statpc5index*. The precision parity is relatively high, however, given that the p-values of both the recall parity and precision parity are not significant, the bias can be considered to be limited.

Focumcheck_statpc6index. This variable represents a risk score based on an individual's 6 position zip code (e.g., 1234AB), with values ranging from 0 to 2314. Group 0 contains individuals with a score of 90 and below, and group 1 contains individuals with scores greater than 90. In Table 3, it can be seen that the recall parity (0.2186) is relatively high and the precision parity (0.1633) relatively low. The p-values corresponding to the recall parity and precision parity are not significant (Table 4). Table 5 shows that the recall of group 0 (0.0632) is lower than the recall of group 1 (0.0809), so group 0 gets away with fraud relatively more often than group 1. The precision of group 1 (0.3609) is lower than the precision of group 0 (0.4314), so group 1 receives more false fraud predictions. This indicates a slight bias against group 1, which could come from the training data having relatively more fraud cases for group 1 (4.2%) than for group 0 (1.8%).

Tables 6 and 7 show that $FR2_0 = 0.0045 > FR2_1 = 0.0013$ and $FR5_0 = 0.7185 > FR5_1 = 0.27$. This indicates bias against group 1, as the predictions change to *fraud* relatively often for samples that change from group 0 to group 1. $FR3_0 = 0.0006 < FR3_1 = 0.0045$ and $FR4_0 = 0.2549 < FR4_1 = 0.568$ also indicate bias against group 1, as the model changes its prediction to *no fraud* more often when a sample changed from group 1 to group 0. This could be because group 1 has a higher rate of fraud cases than group 0.

The model is slightly biased against group 1 of *focumcheck_statpc6index*, presumably due to the fact that one group contains relatively more fraud cases than the other in the training data.

Focumcheck_debtresult_d02. This is a binary feature that takes a value of 1 if the score on an individual's address is higher than 0, and a value of 0 if the score is 0. This variable has a high imbalance, since group 0 has 41,883 samples and group 1 has 1,404 samples. Furthermore, 8.5% of group 1 committed fraud in the training data and 2.7% of group 0 did. In Tables 3 and 4 it can be seen that the recall parity is 0.0429 and not significant. The precision parity (0.2816) is realtively high, but according to the p-value (0.3817) it is not significant. Table 5 shows that the recall and precision are higher for group 0, so the model is more accurate for this group. This makes sense because there are many more data for this group.

Tables 6 and 7 show that something interesting happens to the flip rates.

$FR2_0 = FR2_1 = FR5_1 = 0$ and $FR5_0 = 0.037$, so there are almost no flips to *fraud* in both groups. There are many flips to *no fraud*, as $FR4_0 = 0.8667$ and $FR4_1 = 0.96$, however, as they are relatively close, it does not indicate significant bias. The highly imbalanced group sizes, which in turn have a high imbalance in fraud cases, can be a reason for these results.

Considering all this, the model is not significantly biased against the feature *focumcheck_debtresult_d02*, despite the high disparities in group sizes and fraud cases.

Focumcheck_debtresult_d03. This feature has a value of 1 if the score on an individual's zipcode is larger than 0, and a value of 0 if the score is 0. The group sizes are slightly imbalanced, but both are reasonably large, as group 0 has size 24,748 and group 1 has size 18,539. In the training data, 2.2% of group 0 and 3.9% of group 1 were fraud cases. Table 3 shows that the recall parity is 0.041, which is relatively low, and the precision parity is 0.2697. Table 5 shows that the precision of group 0 (0.4494) is larger than the precision of group 1 (0.3282), indicating bias against group 1. However, since the p-value for the precision parity is not significant at 0.0888 (Table 4), the indicated bias is limited.

Tables 6 and 7 demonstrate that the flip rates also indicate a slight bias against group 1. $FR2_0 = FR2_1 = 0$ and $FR5_0 = FR5_1 = 0$ indicate that there are no flips to *fraud* for both groups. In contrast, $FR4_0 = 0.9663$ and $FR4_1 = 0.9847$ are high but also close together. This indicates that the model flips a *fraud* prediction to *no fraud* almost always, but does so at approximately the same rate in both groups, indicating no bias.

Taking into account both the parities and the flip rates, the model is not significantly biased against the feature *focumcheck_debtresult_d03*.

Focumcheck_debtresult_d99. If this feature has a value of 1, the individual has a score of 0 on person, address, and zipcode. If the value is 0, it has a score of > 0 in at least one of the categories. The group sizes are relatively balanced, at 20,583 and 22,704 for group 0 and group 1 respectively. 1.7% of group 1 committed fraud in the training data and 4.4% of group 0 did. In Table 3 it can be seen that the recall parity (0.1603) and precision parity (0.3526) are both relatively high. Table 4 confirms that the precision parity is significant with a p-value of 0.0186. The recall parity is not significant with a p-value of 0.4639. The precision of group 0 (0.3292) is lower than the precision of group 1 (0.5085), which indicates bias against group 0 because that group receives more false fraud predictions than group 1.

In Tables 6 and 7 it can be seen that $FR2_0 = 0.0067 > FR_1 = 0.0004$ and $FR5_0 = 0.4825 > FR5_1 = 0.1270$. This indicates bias against group 1, as the model changed its prediction to *fraud* more often for samples that changed from group 0 to group 1 than the other way around. This is interesting because in the training data, 1.7% of group 1 were fraud cases, while 4.4% of group 0 were fraud cases. Thus, it would be expected that the model would suspect group

0 of fraud more quickly than group 1. $FR4_0 = 0.0807$ and $FR4_1 = 0.0678$ are low and lie close together, indicating few flips to *no fraud* for both groups.

The model is not significantly biased against one group of the feature *focumcheck_debtresult_d99*, as the significant precision parity indicates bias against group 0 of the variable and the flip rates indicate bias in the opposite way.

To summarize the findings: the model is biased against most analyzed variables to a certain extent, presumably due to imbalance in group sizes and in fraud cases. Specifically, the model shows significant bias against these variables: *calculated_countrycode_nld* and *calculated_age*. The model shows limited bias against these variables: *focumcheck_statpc4index*, *focumcheck_statpc5index*, and *focumcheck_statpc6index*. Finally, the model has no specific bias against these variables: *gender*, *focumcheck_debtresult_d02*, *focumcheck_debtresult_d03*, and *focumcheck_debtresult_d99*.

6.2 Mitigating bias

6.2.1 Method 1: Fair-SMOTE

Since the number of subgroups increases with the number of protected attributes, not all sensitive attributes were considered. Only the attributes against which the model showed (limited) bias were balanced: *calculated_countrycode_nld*, *calculated_age*, *focumcheck_statpc4index*, *focumcheck_statpc5index*, and *focumcheck_statpc6index*.

Hyperparameters. Various hyperparameters were set to a certain value and other were tuned with *hyperopt* (Bergstra et al., 2013). The following hyperparameters were tuned: *learning_rate*, *max_depth*, *min_child_weight*, *subsample*, *colsample_bytree*, *gamma*, *reg_alpha*, *reg_lambda*. For this method, it was not necessary to tune *scale_pos_weight*, since the training data was balanced. The search space for these hyperparameters was large at first, so the results differed significantly for different random seeds. The hyperparameters are tuned by finding the combination of hyperparameters that lead to the model that performed best according to a user specified performance metric. A larger space leads to more possible combinations, which leads to more variability or randomness. Therefore, after performing multiple runs, the resulting hyperparameters were considered and the search space was reduced. The resulting space and values can be found in Table 18 in the appendix. There is still randomness, so for reproducibility, a random seed of 0 is used.

Bias and accuracy. The accuracy results of method 1 relative to the original model can be found in Table 16. It can be seen that this model performed slightly worse than the original model. This method balanced a number of subgroups by creating synthetic samples for all subgroups so that their sizes were equal to the subgroup with the largest size. Since there were five features that had to be balanced, there were $2^6 = 64$ subgroups, which resulted in a

Feature	Recall parity	Precision parity
calculated_countrycode_nld	0.5465	0.3889
calculated_age	0.1713	0.3961
gender	0.2567	0.0207
focumcheck_statpc4index	0.6356	0.0824
focumcheck_statpc5index	0.2441	0.4338
focumcheck_statpc6index	0.4000	0.2678
focumcheck_debtresult_d02	0.5638	0.2779
focumcheck_debtresult_d03	0.1291	0.4447
focumcheck_debtresult_d99	0.1442	0.4486

Table 8: Parities of each sensitive attribute for method 1 (Fair-SMOTE)

Feature	Recall p-value	Precision p-value
calculated_countrycode_nld	0.0003	0.0072
calculated_age	0.4009	0.0076
gender	0.1851	1.0000
focumcheck_statpc4index	0.0002	0.7107
focumcheck_statpc5index	0.2686	0.0105
focumcheck_statpc6index	0.0491	0.1599
focumcheck_debtresult_d02	0.0042	0.1798
focumcheck_debtresult_d03	0.5679	0.0014
focumcheck_debtresult_d99	0.5434	0.0035

Table 9: P-values for each sensitive attribute for method 1 (Fair-SMOTE)

considerable amount of synthetic data. Therefore, it is logical that it did not perform as well on the test data.

Table 16 shows that the precision of this model (0.3168) was lower than the precision of the original model (0.3773), indicating that relative to the total fraud predictions, there were more false fraud predictions. A reason for this could be that of the 43,287 samples in the test data, the model predicted *fraud* for 262 samples, while the original model predicted *fraud* for 220 samples. More fraud predictions logically lead to more false fraud predictions. The recall is the same as the original model, which is due to the threshold optimization that required it. The threshold was 0.3089, which means that samples with a predicted fraud probability of at least 0.3089 received a *fraud* prediction.

As can be seen in Table 9, every feature except *gender* has a significant recall parity and/or a significant precision parity. In Table 8 it can be seen that in general, the parities are higher than those of the original model. This suggests that, with respect to fairness, this model performs worse than the original model.

	Group 0			Group 1		
Feature	Recall	Precision	size	Recall	Precision	size
calculated_countrycode_nld	0.1242	0.2500	4536	0.0563	0.4091	38751
calculated_age	0.0702	0.2684	21649	0.0847	0.4444	21638
gender	0.0904	0.3208	17988	0.0672	0.3141	25299
focumcheck_statpc4index	0.0349	0.3421	21811	0.0959	0.3139	21384
focumcheck_statpc5index	0.0616	0.5000	21748	0.0815	0.2831	21447
focumcheck_statpc6index	0.0517	0.4091	21759	0.0862	0.2995	21436
focumcheck_debtresult_d02	0.0675	0.3009	41883	0.1546	0.4167	1404
focumcheck_debtresult_d03	0.0806	0.4421	24748	0.0702	0.2455	18539
focumcheck_debtresult_d99	0.0788	0.2757	20583	0.0674	0.5000	22704

Table 10: Metrics per group for method 1 (Fair-SMOTE)

6.2.2 Method 2: AUPRC

Hyperparameters. Various hyperparameters were set to a certain value and others were tuned with *hyperopt* (Bergstra et al., 2013). The following hyperparameters were tuned: *learning_rate*, *max_depth*, *min_child_weight*, *subsample*, *colsample_bytree*, *gamma*, *scale_pos_weight*, *reg_alpha*, *reg_lambda*. Similarly to method 1, the search space started out large and was reduced after performing several runs. The resulting space and value can be found in Table 19 in the appendix. There is still randomness, so for reproducibility, a random seed of 0 is used.

Feature	Recall parity	Precision parity
calculated_countrycode_nld	0.3234	0.5446
calculated_age	0.1278	0.2668
gender	0.2185	0.0270
focumcheck_statpc4index	0.3770	0.2289
focumcheck_statpc5index	0.1951	0.3391
focumcheck_statpc6index	0.3568	0.0735
focumcheck_debtresult_d02	0.2975	0.0358
focumcheck_debtresult_d03	0.0534	0.1541
focumcheck_debtresult_d99	0.1935	0.0914

Table 11: Parities of each sensitive attribute for method 2 (AUPRC)

Bias and accuracy. The accuracy results of method 2 can be found in Table 16. It also shows the differences relative to the original model. It can be seen that the model trained with method 2 performs slightly worse than the original model. This was expected, since improving fairness generally leads to a loss in accuracy (Trinh & Zhang, 2024).

In Table 16 it can be seen that the precision of this model (0.3458) is some-

Feature	Recall p-value	Precision p-value
calculated_countrycode_nld	0.0756	0.0000
calculated_age	0.5485	0.1051
gender	0.2784	0.8910
focumcheck_statpc4index	0.0542	0.2331
focumcheck_statpc5index	0.3901	0.0570
focumcheck_statpc6index	0.0856	0.7537
focumcheck_debtresult_d02	0.3096	1.0000
focumcheck_debtresult_d03	0.8200	0.4091
focumcheck_debtresult_d99	0.3945	0.6443

Table 12: P-values for each sensitive attribute for method 2 (AUPRC)

what worse than the precision of the original model (0.3773). The reason could be that this model predicted *fraud* slightly more frequently. Of the 43,287 samples of test data, this model predicted *fraud* for 240 samples, while the original model predicted *fraud* for 220 samples. Threshold optimization ensured that the recall was 0.0751, same as the original model. The highest threshold that still achieved this was found to be 0.3852.

The recall parity and precision parity of the sensitive attributes in method 2 can be found in Table 11. The corresponding p-values can be found in Table 12, where the significant ones are shaded. By comparing the p-values to those in Table 4, it can be seen that the bias has slightly changed from the original model. The p-values still indicate a bias against the variable *calculated_countrycode_nld*, but not against *calculated_age*, *focumcheck_statpc4index*, *focumcheck_debtresult_d99*, and *focumcheck_debtresult_d02*. To further investigate this bias, the recall, precision, and group sizes are presented in Table 13 for each group and feature. The groups are the same as in the original model and for each feature, the group with the highest recall and lowest precision is highlighted, as that would indicate bias against that group.

The feature *calculated_countrycode_nld* is the only one with a significant precision parity. In Table 13, it can be seen that the precision of group 0 is 0.2256 and the precision of group 1 is 0.4953. Of the measured groups, these are actually the lowest and highest precisions overall. It means that of the fraud predictions the model makes, relatively few are correct for samples in group 0 and relatively many are correct for samples in group 1. This might be due to the distribution of the original groups. Group 0 has a much smaller size (4,536) than group 1 (38,751) and, in the training data, 10.3% of group 0 committed fraud, while only 2.1% of group 1 committed fraud. The recall parity is not significant, however, it is relatively high at 0.3234. The recall of group 0 (0.098) is higher than the recall of group 1 (0.0663), meaning that group 1 gets away with fraud relatively more frequently than group 0.

The flip rates of this model can be found in Tables 14 and 15 for groups 0 and 1, respectively. As a reminder: $FR1$ represents the ratio of predictions that

	Group 0			Group 1		
Feature	Recall	Precision	size	Recall	Precision	size
calculated_countrycode_nld	0.0980	0.2256	4536	0.0663	0.4953	38751
calculated_age	0.0715	0.3114	21649	0.0820	0.4247	21638
gender	0.0878	0.3402	17988	0.0686	0.3497	25299
focumcheck_statpc4index	0.0538	0.4255	21811	0.0863	0.3281	21384
focumcheck_statpc5index	0.0645	0.4783	21748	0.0802	0.3161	21447
focumcheck_statpc6index	0.0546	0.3276	21759	0.0849	0.3536	21436
focumcheck_debtresult_d02	0.0724	0.3443	41883	0.1031	0.3571	1404
focumcheck_debtresult_d03	0.0729	0.3800	24748	0.0771	0.3214	18539
focumcheck_debtresult_d99	0.0801	0.3371	20583	0.0646	0.3710	22704

Table 13: Metrics per group for method 2 (AUPRC)

flipped in either direction out of the total sample size. $FR2$ indicates how many of the total predictions flipped from *no fraud* for the observed data to *fraud* for the counterfactual data. Similarly, $FR3$ is the ratio of total predictions that changed from *fraud* for observed data to *no fraud* for counterfactual data. $FR4$ indicates how many of the *fraud* predictions for the observed data changed to *no fraud* for the counterfactual data. Finally, $FR5$ is the ratio of how many of the *fraud* predictions for the counterfactual data were *no fraud* for the observed data. Compared to the flip rates of the original model (Tables 6 and 7), the universal flip rates ($FR1$) of this model are generally lower.

	Group 0				
Feature	$FR1$	$FR2$	$FR3$	$FR4$	$FR5$
calculated_countrycode_nld	0.0216	0.0013	0.0203	0.6917	0.1277
calculated_age	0.0034	0.0011	0.0023	0.2934	0.1690
gender	0.0036	0.0017	0.0018	0.3402	0.3263
focumcheck_statpc4index	0.0047	0.0046	0.0001	0.0426	0.6918
focumcheck_statpc5index	0.0018	0.0017	0.0001	0.0435	0.4568
focumcheck_statpc6index	0.0089	0.0089	0.0000	0.0000	0.7689
focumcheck_debtresult_d02	0.0032	0.0028	0.0004	0.0755	0.3778
focumcheck_debtresult_d03	0.0025	0.0023	0.0002	0.0500	0.3791
focumcheck_debtresult_d99	0.0034	0.0024	0.0011	0.1236	0.2390

Table 14: Flip rates group 0 for method 2 (AUPRC)

The feature *calculated_countrycode_nld* has a significant precision parity, indicating bias against group 0. In Tables 14 and 15 it can be seen that $FR4_0 = 0.6917 > FR5_0 = 0.1277$, indicating that if a sample of group 0 changes to group 1, the prediction changes to *no fraud* relatively more frequently than to *fraud*. Additionally, $FR4_0 = 0.6917 > FR4_1 = 0.2897$ indicates that the prediction changes to *no fraud* relatively more frequently if a sample from group 0 changes to group 1 than when a sample changes from group 1 to group

	Group 1				
Feature	$FR1$	$FR2$	$FR3$	$FR4$	$FR5$
calculated_countrycode_nld	0.0039	0.0031	0.0008	0.2897	0.6162
calculated_age	0.0033	0.0032	0.0001	0.0274	0.4929
gender	0.0032	0.0018	0.0014	0.2517	0.3007
focumcheck_statpc4index	0.0043	0.0016	0.0028	0.3073	0.2036
focumcheck_statpc5index	0.0046	0.0022	0.0024	0.2694	0.2500
focumcheck_statpc6index	0.0039	0.0028	0.0011	0.1326	0.2731
focumcheck_debtresult_d02	0.0057	0.0014	0.0043	0.2143	0.0833
focumcheck_debtresult_d03	0.0038	0.0024	0.0013	0.1786	0.2812
focumcheck_debtresult_d99	0.0011	0.0011	0.0001	0.0323	0.2857

Table 15: Flip rates group 1 for method 2 (AUPRC)

0. This indicates a bias against group 0. $FR5_1 = 0.6162 > FR4_1 = 0.2897$ and $FR5_1 = 0.6162 > FR5_0 = 0.1277$ indicate that if a sample from group 1 changes to group 0, the prediction changes to *fraud* relatively more often than to *no fraud* and that it changes to *fraud* relatively more frequently than if a sample changes from group 0 to group 1. Together, this suggests a bias against group 0, however it is also logical since group 0 has relatively more fraud cases than group 1.

Due to randomness, a different random seed may lead to other variables having significant parities. Therefore, various variables with relatively high parities are investigated.

The feature *focumcheck_statpc5index* has a relatively high precision parity (Table 11). In Table 13 it can be seen that group 1 has a lower precision (0.3161) than group 0 (0.4783), which means that of the fraud predictions, relatively more are false for group 1 than for group 0. In Table 15 it can be seen that $FR4_1 = 0.2694 \approx FR5_1 = 0.25$, so when the sample changes from group 1 to group 0, the prediction changes to *fraud* or *no fraud* at approximately the same rate. However, Table 14 shows that $FR4_0 = 0.0435 < FR5_0 = 0.4568$, indicating that when a sample changes from group 0 to group 1, the prediction changes to *fraud* relatively more frequently than to *no fraud*. The reason could be that in the training data, 4.2% of group 1 committed fraud and only 1.8% of group 0. Together, it indicates limited bias against the variable *focumcheck_statpc5index*.

In addition to *calculated_countrycode_nld*, there are two other features with relatively high recall parities (Table 11): *focumcheck_statpc4index* and *focumcheck_statpc6index*. The feature *focumcheck_statpc4index* has a recall parity of 0.377 (Table 11). The recall of group 0 (0.0538) is lower than the recall of group 1 (0.0863), indicating that group 0 gets away with fraud slightly more often than group 1. The flip rates also indicate a slight bias against group 1. In Table 14 it can be seen that $FR4_0 = 0.0426 < FR5_0 = 0.6918$, so when a sample switches from group 0 to group 1, the prediction changes to *fraud* much more frequently than to *no fraud*. Table 15 shows that $FR4_1 = 0.3073 > FR5_1 = 0.2036$, which is a much smaller difference, but still indicates that when a sample changes from

group 1 to 0, the prediction changes to *no fraud* relatively more frequently than to *fraud*. The reason for the differences in flip rates can be that the training data contained relatively more fraud cases for group 1 (4.1%) than for group 0 (1.9%). All this considered, this model does not have a significant bias against the variable *focumcheck_statpc4index*.

The feature *focumcheck_statpc6index* has a relatively high recall parity of 0.3568 (Table 11). Table 13 shows that group 1 has a recall of 0.0849, which is higher than the recall of group 0, which is 0.0546. This means that group 0 gets away with fraud relatively more frequently than group 1. The flip rates in Tables 14 and 15 do not indicate significant bias, as $FR4_0 = 0.0 < FR5_0 = 0.7689$ and $FR4_1 = 0.1326 < FR5_1 = 0.2731$. This means that independent of the group, the prediction changes to *fraud* relatively more frequently than to *no fraud* when the group changes. The difference is larger for group 0, presumably because the training data contained more fraud cases for group 1 (4.2%) than for group 0 (1.8%).

All in all, this model still has a slight bias, so it is not completely mitigated. However, the bias has been reduced with a limited accuracy loss.

6.2.3 Comparison

The original model had the most significant bias against the variables *calculated_countrycode_nld* and *calculated_age*. The results of the model that was trained with method 1 (Fair-SMOTE) indicated bias against more features. The model trained with method 2 (AUPRC) only had a significant bias against the variable *calculated_countrycode_nld*. The group sizes of *calculated_countrycode_nld* are unbalanced, as are the distributions of fraud cases. Group 0 has a size of 4,536, where 10.3% is a fraud case, and group 1 has a size of 38,751, where 2.1% is a fraud case. Knowing this, it is logical that the model suspects group 0 of fraud faster than group 1.

Metric	Original	Method 1	Method 2
AUC	0.8695	0.8584 (-0.0111)	0.8424 (-0.0271)
AUPRC	0.1906	0.1689 (-0.0217)	0.1685 (-0.0221)
G-mean	0.2736	0.2735 (-0.0001)	0.2736 (-0.0001)
F1-score	0.1253	0.1214 (-0.0038)	0.1234 (-0.0019)
MCC	0.1594	0.1441 (-0.0153)	0.1516 (-0.0078)
Recall	0.0751	0.0751 (-0.0000)	0.0751 (-0.0000)
Precision	0.3773	0.3168 (-0.0605)	0.3458 (-0.0314)

Table 16: Accuracy results for each method

In Table 16 it can be seen that the original model has the best accuracy results. The other methods perform somewhat worse, but are still close to the original model. This makes sense because trying to lower the bias often leads to a lower accuracy.

7 Discussion

The goal of this research was to answer the following question: "*Does the credit risk engine at Dutch telecom company X contain bias according to the European AI Act, and if so, how can this bias be mitigated while keeping accuracy at an appropriate level?*"

The project was split into two phases, detecting bias and mitigating the found bias. Bias detection was performed on both a group level and an individual level. To determine whether the CRE contained bias against certain groups, the recall parity and precision parity were calculated (Kamalaruban et al., 2024). If these parities are small, it indicates fairness, since fraud is detected at similar rates across groups. If the parities are large, it means that one group is falsely accused of fraud more often than the other, or that one group commits fraud without detection much more often than the other group. On an individual level, fairness is measured with the counterfactual fairness method (Kusner et al., 2017). This method essentially changes the value of a sensitive attribute and checks whether the prediction changes by considering the flip rate. Flip rates indicate the rates at which a prediction changed (flipped) when the sensitive attribute changed. If the prediction changed much more for one group than for the other group, it indicates unfairness. However, in various cases, one group commits fraud relatively more often than the other group, which can then explain those differences. Therefore, the results of the different methods should be carefully considered together with the original data distribution.

The parities of nine sensitive attributes were investigated. Only one feature had a significant recall parity, namely, *focumcheck_statpc4index*. There were three features with significant precision parities. For those features, one group received significantly more false fraud accusations than the other group relative to the total number of fraud predictions for that group. This was the case for *calculated_countrycode_nld*, *calculated_age*, and *focumcheck_debtresult_d99*. These results were considered along with the flip rates and indicated that the model contained bias against the features *calculated_countrycode_nld* and *calculated_age*. The model contained limited or no bias against the other variables.

The next step was to mitigate the bias of the original model. The techniques to mitigate bias can be categorized into when in the model training process they are applied. Pre-processing techniques try to mitigate bias before training the model, in-processing techniques try to do it during model training, and post-processing techniques after model training (Mehrabi et al., 2021). One technique of each category was chosen and combined into two methods. The first method aimed to mitigate bias by balancing the training data using Fair-SMOTE (Chakraborty et al., 2021), since it was assumed that part of the bias came from the imbalanced nature of the data, and by applying threshold optimization after training (Leevy et al., 2023). Fair-SMOTE balances not only classes, but also groups within the training data. The default threshold for probability scores is 0.5, meaning that anything above it results in a fraud prediction and anything below it in a non-fraud prediction. Threshold optimization finds a different threshold that optimizes a certain goal. In this case, the goal was

to achieve at least the recall of the original model while minimizing the loss in precision. The second method used to mitigate bias evaluated the model during training with the AUPRC instead of the AUC and applied threshold optimization after training (Leevy et al., 2023). The AUPRC has been found to be an appropriate alternative for the AUC when data are imbalanced (Hancock et al., 2023).

These methods were applied and the results were compared to those of the original model. The model created with the first method performed worse than the original model regarding both accuracy and fairness. Compared to the original model, the second method was found to have improved fairness, with a limited loss of accuracy. This illustrates the trade-off where improving fairness may come at the expense of predictive performance. However, since both the original model and these new models predict little fraud overall, slight variations in predictions can considerably affect fairness metrics. Therefore, the robustness of all three models should be further investigated to verify the generality of the methods.

The reason for the poor performance of Fair-SMOTE could be the amount of synthetic data. Chakraborty et al. (2021) found that the method performs well for one or two sensitive attributes. However, in this research, the method is applied to five attributes, leading to 64 subgroups. To balance all 64 subgroups, a substantial amount of synthetic data was generated. Presumably, the model was trained on so much synthetic data that the performance suffered for it. Perhaps, this method would have performed better if only one or two attributes had been balanced.

The second method performed slightly better with respect to fairness and slightly worse with respect to accuracy compared to the original model. There was still a significant bias against the variable *calcualted_countrycode_nld*. To mitigate bias against this feature as well, more advanced methods would be required. However, since this project focused on bias detection as well as mitigation, it was outside the scope of this research to use more advanced methods. The results of this model were similar to those of the original model, which makes sense because the models themselves were similar. The main difference between this model and the original model was that during training, this model used the AUPRC instead of the AUC as an evaluation metric. The loss function remained the same. Z. Liu & Bondell (2019) found that classifiers that optimize the AUPRC can outperform classifiers that optimize the AUC for imbalanced data. Incorporating the AUPRC in the loss function could be interesting to attempt in this context, but it was outside the scope of this research.

There was randomness in the counterfactual fairness method. Since assigning causal mechanisms takes quite some time, a representative run was used in this project. Confidence intervals for the flip rates could be calculated to obtain more robust results. Additionally, there was randomness in the training of the models. A random seed of 0 was used to obtain reproducible results, however, it could be further investigated how different random seeds influence the results. For example, since there are relatively few fraud predictions, a few differences in predictions can result in larger differences in the parities and their significances.

This research succeeded in finding methods that consider the different aspects of fraud data and in applying them to real-world data. However, the majority of the literature on detecting bias or measuring fairness considered only binary variables. There is limited relevant literature on doing the same for continuous or categorical variables. In this research, this was solved by one-hot-encoding categorical variables and splitting continuous variables into two groups. Presumably, this could have led to information loss, and it would be valuable to have more methods specifically created for non-binary variables. However, creating an entirely new method was outside the scope of this project.

This research was based on an existing model. Therefore, the decision was made to only consider the variables that were used in that model. It would be valuable to also consider other variables and perform feature selection or feature engineering in further research. For example, bias detection was focused on nine features that were considered the most relevant in terms of fairness. However, the same methods could be applied for even more features. There are several features that come from third parties, and it is not always clear what exact data they use. In this research, third party variables that likely used sensitive information, for example, those that returned a risk score based on zipcode, were investigated. It could be interesting to further explore the less obvious features regarding fairness. It could also be investigated whether these features add to the performance of the model or whether the model can do without them.

The investigated CRE predicts fraud for customers interested in small loans, however, the company also has one that considers larger loans or loans to businesses. These are not investigated in this research, but could be investigated in future research with similar methods.

8 Conclusion

This project focused on the following research question: "*Does the credit risk engine at Dutch telecom company X contain bias according to the European AI Act, and if so, how can this bias be mitigated while keeping accuracy at an appropriate level?*"

The answer to the first part is yes, the results indicate a bias in the CRE associated with the two variables *calculated_countrycode_nld* and *calculated_age*, which represent whether someone has a Dutch ID and their age, respectively. This conclusion is based on the significant precision parities of both variables, which indicate that one group receives significantly more false fraud predictions than the other group within these variables. The flip rates further confirmed this bias by having large differences between groups. The disparities were likely influenced by the class imbalance between groups.

The second part of the research question was how to mitigate the found bias. Two new models were trained based on two different methods. The first (FairSMOTE) achieved both a lower fairness and lower accuracy than the original model. The second (AUPRC) realized improved fairness relative to the original model, with a slight reduction in accuracy.

Since this research was aimed at both detecting and mitigating bias, the bias mitigation techniques were not as advanced as they are in other projects that focus solely on mitigating bias. However, the AUPRC showed potential and could perhaps achieve even better fairness when incorporated into the loss function, as Z. Liu & Bondell (2019) did.

Finally, this project aimed to fill a gap in the existing literature by finding methods to detect and mitigate bias that consider the different aspects of fraud data and can be applied to real world data. An example of how this was achieved is how counterfactual fairness was investigated. For counterfactual fairness, a causal graph is required (Kusner et al., 2017). However, much of the existing literature on counterfactual fairness assumes that a causal graph is known, which is rarely the case in real-world datasets. Therefore, the counterfactual fairness method was combined with a causal discovery algorithm to create a causal graph. This algorithm was further adjusted to account for unobserved variables, which are realistically unavoidable in real-world settings. In conclusion, this research demonstrates that combining and adjusting existing methods enables the detection and reduction of bias in a real-world credit risk system.

References

Alowais, S. A., Alghamdi, S. S., Alsuhebany, N., Alqahtani, T., Alshaya, A. I., Almohareb, S. N., ... Albekairy, A. M. (2023). Revolutionizing healthcare: the role of artificial intelligence in clinical practice. *BMC Medical Education*, *23*(689). Retrieved from `https://doi.org/10.1186/s12909-023-04698-z` doi: 10.1186/s12909-023-04698-z

Bergstra, J., Yamins, D., & Cox, D. (2013, 17–19 Jun). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In S. Dasgupta & D. McAllester (Eds.), *Proceedings of the 30th international conference on machine learning* (Vol. 28, pp. 115–123). Atlanta, Georgia, USA: PMLR. Retrieved from `https://proceedings.mlr.press/v28/bergstra13.html`

Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications*, *39*(3), 3446-3453. Retrieved from `https://www.sciencedirect.com/science/article/pii/S095741741101342X` doi: https://doi.org/10.1016/j.eswa.2011.09.033

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., ... Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. In *Ecml pkdd workshop: Languages for data mining and machine learning* (pp. 108–122).

Cai, Z., Xi, D., Zhu, X., & Li, R. (2022). Causal discoveries for high dimensional mixed data. *Statistics in Medicine*, *41*(24), 4924–4940. Retrieved from `https://doi.org/10.1002/sim.9544` doi: 10.1002/sim.9544

Calders, T., Kamiran, F., & Pechenizkiy, M. (2009). Building classifiers with independency constraints. In *2009 ieee international conference on data mining workshops* (p. 13-18). doi: 10.1109/ICDMW.2009.83

Celis, L. E., Huang, L., Keswani, V., & Vishnoi, N. K. (2019). Classification with fairness constraints: A meta-algorithm with provable guarantees. In *Proceedings of the conference on fairness, accountability, and transparency* (p. 319–328). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3287560.3287586 doi: 10.1145/3287560.3287586

Chakraborty, J., Majumder, S., & Menzies, T. (2021). Bias in machine learning software: why? how? what to do? In *Proceedings of the 29th acm joint meeting on european software engineering conference and symposium on the foundations of software engineering* (p. 429–440). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3468264.3468537 doi: 10.1145/3468264.3468537

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*. Retrieved from https://doi.org/10.1613/jair.953 doi: 10.1613/jair.953

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., ... Cortes, D. (2026). xgboost: Extreme gradient boosting [Computer software manual]. Retrieved from https://github.com/dmlc/xgboost (R package version 3.2.0.0)

Chicco, D., & Jurman, G. (2020). The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, *20*(6). Retrieved from https://doi.org/10.1186/s12864-019-6413-7 doi: 10.1186/s12864-019-6413-7

Dablain, D., Krawczyk, B., & Chawla, N. V. (2024). Towards a holistic view of bias in machine learning: bridging algorithmic fairness and imbalanced learning. *Discover Data*, *2*(4). Retrieved from https://doi.org/10.1007/s44248-024-00007-1 doi: 10.1007/s44248-024-00007-1

Dastile, X., Celik, T., & Potsane, M. (2020). Statistical and machine learning models in credit scoring: A systematic literature survey. *Applied Soft Computing*, *91*, 106263. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494620302039 doi: https://doi.org/10.1016/j.asoc.2020.106263

de la Cruz Huayanay, A., Bazán, J. L., & Russo, C. M. (2024). Performance of evaluation metrics for classification in imbalanced data. *Computational Statistics*, *40*, 1447–1473. Retrieved from https://doi.org/10.1007/s00180-024-01539-5 doi: 10.1007/s00180-024-01539-5

de Lima Cabral, D. R., & Barros, R. S. M. (2018). Concept drift detection based on fisher's exact test. *Information Sciences*, *442–443*, 220–234. Retrieved from https://www.sciencedirect.com/science/article/pii/S0020025518301403 doi: 10.1016/j.ins.2018.01.051

de Lima Cabral, D. R., & de Barros, R. S. M. (2018). Concept drift detection based on fisher's exact test. *Information Sciences*, *442-443*, 220-234. Retrieved from https://www.sciencedirect.com/science/article/pii/S0020025518301403 doi: https://doi.org/10.1016/j.ins.2018.02.054

Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R. (2012). Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference* (p. 214–226). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/2090236.2090255 doi: 10.1145/2090236.2090255

European Parliament. (2023). *Eu ai act: First regulation on artificial intelligence.* Retrieved from https://www.europarl.europa.eu/topics/en/article/20230601STO93804/eu-ai-act-first-regulation-on-artificial-intelligence (Published 8 June 2023; last updated 19 February 2025. Accessed 3 February 2026)

European Union. (n.d.-a). *Artificial intelligence act (eu ai act), article 10: Data and data governance.* Retrieved from https://artificialintelligenceact.eu/article/10/ (Accessed: 3 February 2026)

European Union. (n.d.-b). *Recital 58.* Retrieved from https://artificialintelligenceact.eu/recital/58/ (Accessed: 3 February 2026)

European Union. (n.d.-c). *Section 2: Requirements for high-risk ai systems.* Retrieved from https://artificialintelligenceact.eu/section/3-2/ (Accessed: 3 February 2026)

Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., & Venkatasubramanian, S. (2015). Certifying and removing disparate impact. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining* (p. 259–268). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/2783258.2783311 doi: 10.1145/2783258.2783311

Garcia, A. C. B., Garcia, M. G. P., & Rigobon, R. (2024). Algorithmic discrimination in the credit domain: what do we know about it? *AI Society*, *39*, 2059–2098. Retrieved from https://doi.org/10.1007/s00146-023-01676-3 doi: 10.1007/s00146-023-01676-3

Genovesi, S., Mönig, J. M., Schmitz, A., Poretschkin, M., Akila, M., Kahdan, M., ... Zimmermann, A. (2024). Standardizing fairness-evaluation procedures: Interdisciplinary insights on machine learning algorithms in credit-

worthiness assessments for small personal loans. *AI and Ethics*, *4*, 537–553. Retrieved from https://doi.org/10.1007/s43681-023-00291-8 doi: 10.1007/s43681-023-00291-8

Glymour, C., Zhang, K., & Spirtes, P. (2019). Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, *Volume 10 - 2019*. Retrieved from https://www.frontiersin.org/journals/genetics/articles/10.3389/fgene.2019.00524 doi: 10.3389/fgene.2019.00524

Grari, V., Lamprier, S., & Detyniecki, M. (2023). Adversarial learning for counterfactual fairness. *Machine Learning*, *112*(3), 741–763. Retrieved from https://doi.org/10.1007/s10994-022-06206-8 doi: 10.1007/s10994-022-06206-8

Greene, W. H. (2003). *Econometric analysis* (5th ed.). Upper Saddle River, NJ: Pearson Education / Prentice Hall.

Gunnarsson, B. R., vanden Broucke, S., Baesens, B., Óskarsdóttir, M., & Lemahieu, W. (2021). Deep learning for credit scoring: Do or don't? *European Journal of Operational Research*, *295*(1), 292-305. Retrieved from https://www.sciencedirect.com/science/article/pii/S037722172100196X doi: https://doi.org/10.1016/j.ejor.2021.03.006

Gupta, P., Varshney, A., Khan, M. R., Ahmed, R., Shuaib, M., & Alam, S. (2023). Unbalanced credit card fraud detection data: A machine learning-oriented comparative study of balancing techniques. *Procedia Computer Science*, *218*, 2575-2584. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050923002314 (International Conference on Machine Learning and Data Engineering) doi: https://doi.org/10.1016/j.procs.2023.01.231

Hancock, J. T., Khoshgoftaar, T. M., & Johnson, J. M. (2023). Evaluating classifier performance with highly imbalanced big data. *Journal of Big Data*, *10*, 42. Retrieved from https://doi.org/10.1186/s40537-023-00724-5 doi: 10.1186/s40537-023-00724-5

Het College voor de rechten van de mens. (n.d.). *Wat is discriminatie?* Retrieved from https://www.mensenrechten.nl/mensenrechten-voor-jou/discriminatie-en-gelijke-behandeling/wat-is-discriminatie (Accessed: 3 February 2026)

Hort, M., Chen, Z., Zhang, J. M., Harman, M., & Sarro, F. (2024, June). Bias mitigation for machine learning classifiers: A comprehensive survey. *ACM J. Responsib. Comput.*, *1*(2). Retrieved from https://doi.org/10.1145/3631326 doi: 10.1145/3631326

Howard, A., & Borenstein, J. (2018). The ugly truth about ourselves and our robot creations: The problem of bias and social inequity. *Science and*

Engineering Ethics, *24*, 1521–1536. Retrieved from https://doi.org/10.1007/s11948-017-9975-2 doi: 10.1007/s11948-017-9975-2

Hoyer, P. O., Janzing, D., Mooij, J., Peters, J., & Schölkopf, B. (2008). Nonlinear causal discovery with additive noise models. In *Proceedings of the 22nd international conference on neural information processing systems* (p. 689–696). Red Hook, NY, USA: Curran Associates Inc.

Jaworski, P., Durante, F., Hardle, W. K., & Rychlik, T. (2010). *Copula theory and its applications* (Vol. 198). Springer.

Kalisch, M., & Bühlmann, P. (2007). Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, *8*(22), 613–636. Retrieved from http://jmlr.org/papers/v8/kalisch07a.html

Kamalaruban, P., Pi, Y., Burrell, S., Drage, E., Skalski, P., Wong, J., & Sutton, D. (2024). Evaluating fairness in transaction fraud models: Fairness metrics, bias audits, and challenges. In *Proceedings of the 5th acm international conference on ai in finance* (p. 555–563). Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3677052.3698666 doi: 10.1145/3677052.3698666

Kamiran, F., Karim, A., & Zhang, X. (2012). Decision theory for discrimination-aware classification. In *2012 ieee 12th international conference on data mining* (p. 924-929). doi: 10.1109/ICDM.2012.45

Kamishima, T., Akaho, S., Asoh, H., & Sakuma, J. (2012). Fairness-aware classifier with prejudice remover regularizer. In P. A. Flach, T. De Bie, & N. Cristianini (Eds.), *Machine learning and knowledge discovery in databases* (pp. 35–50). Berlin, Heidelberg: Springer Berlin Heidelberg.

Kisten, M., & Khosa, M. (2024). Enhancing fairness in credit assessment: Mitigation strategies and implementation. *IEEE Access*, *12*, 177277-177284. doi: 10.1109/ACCESS.2024.3505836

Kornbrot, D. (2014). Point biserial correlation. *Wiley StatsRef: Statistics Reference Online*. Retrieved from https://onlinelibrary.wiley.com/doi/10.1002/9781118445112.stat06227 doi: 10.1002/9781118445112.stat06227

Kozodoi, N., Jacob, J., & Lessmann, S. (2022). Fairness in credit scoring: Assessment, implementation and profit implications. *European Journal of Operational Research*, *297*(3), 1083-1094. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221721005385 doi: https://doi.org/10.1016/j.ejor.2021.06.023

Kozodoi, N., Lessmann, S., Alamgir, M., Moreira-Matias, L., & Papakonstantinou, K. (2025). Fighting sampling bias: A framework for training and evaluating credit scoring models. *European Journal of Operational Research*, *324*(2), 616-628. Retrieved from https://www.sciencedirect.com/

science/article/pii/S0377221725000839 doi: https://doi.org/10.1016/j.ejor.2025.01.040

Krafft, T. D., Hauer, M. P., & Zweig, K. (2024). Black-box testing and auditing of bias in adm systems. *Minds and Machines*, *34*(15), 1–31. Retrieved from https://doi.org/10.1007/s11023-024-09666-0 doi: 10.1007/s11023-024-09666-0

Kusner, M. J., Loftus, J., Russell, C., & Silva, R. (2017). Counterfactual fairness. In *Proceedings of the 31st international conference on neural information processing systems (nips '17)* (pp. 4069–4079). Retrieved from https://dl.acm.org/doi/10.5555/3294996.3295162 doi: 10.5555/3294996.3295162

Le, T. D., Hoang, T., Li, J., Liu, L., Liu, H., & Hu, S. (2019). A fast pc algorithm for high dimensional causal discovery with multi-core pcs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *16*(5), 1483-1495. doi: 10.1109/TCBB.2016.2591526

Leevy, J. L., Johnson, J. M., Hancock, J. T., & Khoshgoftaar, T. M. (2023). Threshold optimization and random undersampling for imbalanced credit card data. *Journal of Big Data*, *10*(58). Retrieved from https://doi.org/10.1186/s40537-023-00738-z doi: 10.1186/s40537-023-00738-z

Liang, X. S., & Yang, X.-Q. (2021). A note on causation versus correlation in an extreme situation. *Entropy*, *23*(3). Retrieved from https://www.mdpi.com/1099-4300/23/3/316 doi: 10.3390/e23030316

Liu, H., Lafferty, J., & Wasserman, L. (2009, December). The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *J. Mach. Learn. Res.*, *10*, 2295–2328.

Liu, Z., & Bondell, H. D. (2019). Binormal precision–recall curves for optimal classification of imbalanced data. *Statistics in Biosciences*, *11*, 141–161. Retrieved from https://doi.org/10.1007/s12561-019-09231-9 doi: 10.1007/s12561-019-09231-9

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Neural Information Processing Systems*, 4768–4777. Retrieved from https://dl.acm.org/doi/10.5555/3295222.3295230 doi: 10.5555/3295222.3295230

Machado, A., Jiménez Mérida, M., Deo, A., & Pathak, A. (2025). *Ai act governance: Best practices for implementing the eu ai act.* Retrieved from https://www.appliedai.de/en/insights/ai-act-governance-best-practices-for-implementing-the-eu-ai-act/ (Accessed: 3 February 2026)

Mariscal, C., Yustiawan, Y., Rochim, F. C., & Tanuar, E. (2024). Implementing and analyzing fairness in banking credit scoring. *Procedia Computer Science*, *234*, 1492-1499. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050924005088 doi: https://doi.org/10.1016/j.procs.2024.03.150

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021, July). A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, *54*(6). Retrieved from https://doi.org/10.1145/3457607 doi: 10.1145/3457607

Naidu, G., Zuva, T., & Sibanda, E. M. (2023). A review of evaluation metrics in machine learning algorithms. In R. Silhavy & P. Silhavy (Eds.), *Artificial intelligence application in networks and systems* (pp. 15–25). Cham: Springer International Publishing.

Pessach, D., & Shmueli, E. (2022, February). A review on fairness in machine learning. *ACM Comput. Surv.*, *55*(3). Retrieved from https://doi.org/10.1145/3494672 doi: 10.1145/3494672

Peters, J., Janzing, D., & Scholkopf, B. (2011). Causal inference on discrete data using additive noise models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(12), 2436-2450. doi: 10.1109/TPAMI.2011.71

Sedgwick, P. (2014). Spearman's rank correlation coefficient. *BMJ*, *349*. Retrieved from https://www.bmj.com/content/349/bmj.g7327 doi: 10.1136/bmj.g7327

Sonoda, R. (2023). Fair oversampling technique using heterogeneous clusters. *Information Sciences*, *640*, 119059. Retrieved from https://www.sciencedirect.com/science/article/pii/S0020025523006448 doi: https://doi.org/10.1016/j.ins.2023.119059

Spirtes, P., Glymour, C. N., & Scheines, R. (2000). *Causation, prediction, and search* (2nd ed.). Cambridge, MA: MIT Press. Retrieved from https://books.google.com/books?id=vV-U09kCdRwC

Suárez Ferreira, J., Slavkovik, M., & Casillas, J. (2025). General procedure to measure fairness in regression problems. *International Journal of Data Science and Analytics*, *20*, 4343–4362. Retrieved from https://doi.org/10.1007/s41060-025-00721-2 doi: 10.1007/s41060-025-00721-2

Trinh, T. K., & Zhang, D. (2024). Algorithmic fairness in financial decision-making: Detection and mitigation of bias in credit scoring applications. *Journal of Advanced Computing Systems*, *4*(2), 36–49. Retrieved from https://scipublication.com/index.php/JACS/article/view/156 doi: 10.69987/JACS.2024.40204

Xiao, J., Wang, Y., Chen, J., Xie, L., & Huang, J. (2021). Impact of resampling methods and classification models on the imbalanced credit scoring problems. *Information Sciences*, *569*, 508-526. Retrieved from https://www.sciencedirect.com/science/article/pii/S0020025521004874 doi: https://doi.org/10.1016/j.ins.2021.05.029

Zhang, B. H., Lemoine, B., & Mitchell, M. (2018). Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 aaai/acm conference on ai, ethics, and society* (p. 335–340). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3278721.3278779 doi: 10.1145/3278721.3278779

Zhang, X., & Yu, L. (2024). Consumer credit risk assessment: A review from the state-of-the-art classification algorithms, data traits, and learning methods. *Expert Systems with Applications*, *237*, 121484. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417423019863 doi: https://doi.org/10.1016/j.eswa.2023.121484

Zheng, Y., Huang, B., Chen, W., Ramsey, J., Gong, M., Cai, R., ... Zhang, K. (2024, January). Causal-learn: causal discovery in python. *Journal of Machine Learning Research*, *25*(1).

9 Appendix

Hyperparameter	Final value
n_estimators	1000
objective	"binary:logistic"
eval_metric	"auc"
n_jobs	-1
base_score	0.91
learning_rate	0.02
max_depth	16
min_child_weight	3.0
subsample	0.9
colsample_bytree	0.85
gamma	1.4
scale_pos_weight	2.0

Table 17: Hyperparameter values

Hyperparameter	Range	Stepsize	Final value
learning_rate	[0.005, 0.2]	0.005	0.085
max_depth	[5, 20]	1	12
min_child_weight	[1, 5]	1	3
subsample	[0.7, 1]	0.05	0.8
colsample_bytree	[0.7, 1.0]	0.05	0.75
gamma	[0, 0.2]	0.05	0.05
reg_alpha	[0, 2]	0.1	1.2
reg_lambda	[0, 3]	0.1	2

Table 18: Hyperparameter search space and final value for method 1 (Fair-SMOTE)

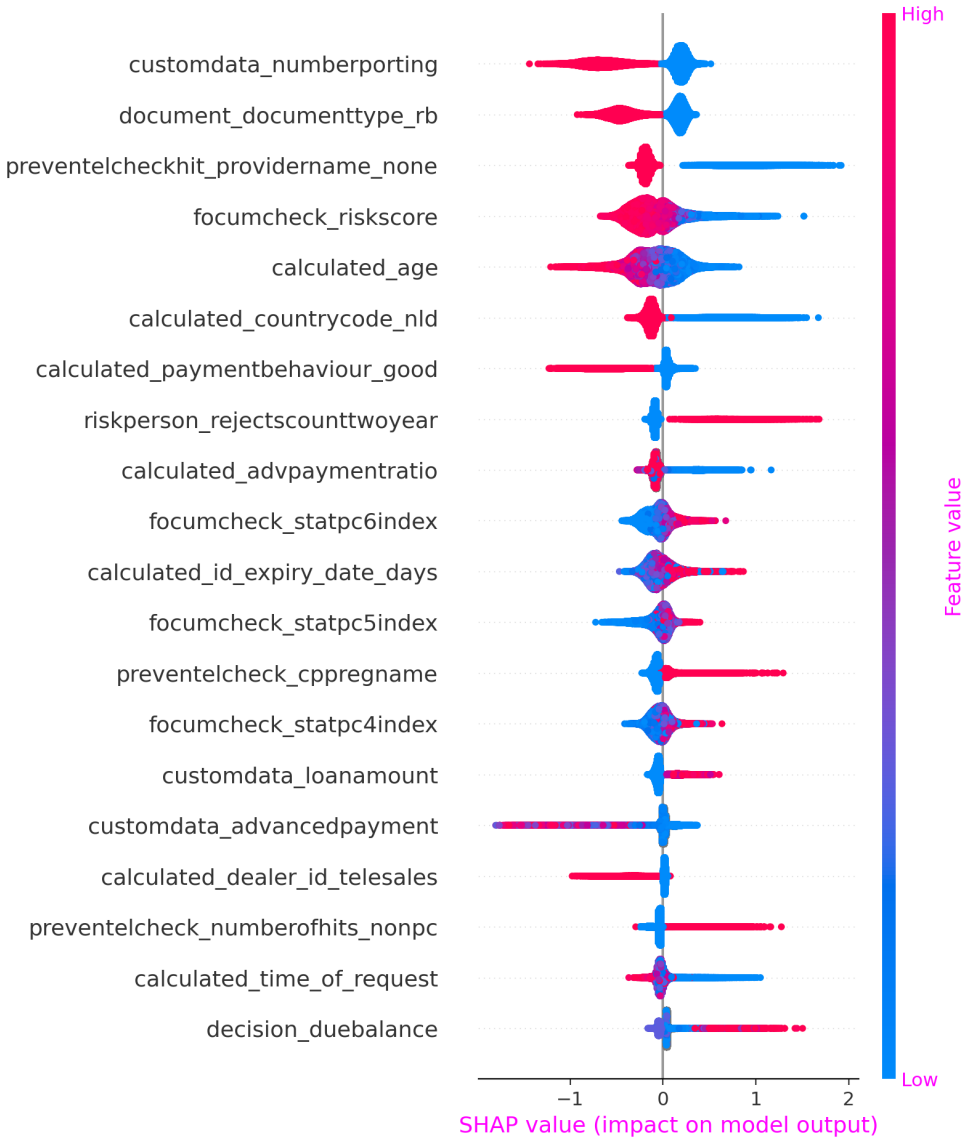

Figure 22: SHAP plot

Hyperparameter	Range	Stepsize	Final value
learning_rate	[0.005, 0.1]	0.005	0.01
max_depth	[1, 15]	1	12
min_child_weight	[2, 5]	1	2
subsample	[0.7, 1]	0.05	0.85
colsample_bytree	[0.6, 0.9]	0.05	0.65
gamma	[0, 1.5]	0.1	1.4
scale_pos_weight	[0, 33.2759]	1	1
reg_alpha	[0, 5]	0.1	0.2
reg_lambda	[0, 5]	0.1	5

Table 19: Hyperparameter search space and final value for method 2 (AUPRC)