Enhancing Fairness: Adversarial Debiasing in Fraud Detection

Final Report

Dylan Steimel

DSA 5900
Summer 2023
July 15, 2023
Faculty Supervisor: Dr. Trafalis

Abstract

This paper addresses the challenge of building a fair fraud detection model for imbalanced datasets, where the minority class (fraudulent transactions) is severely underrepresented. We explore the use of adversarial debiasing, a technique that combines adversarial learning and gradient-based optimization, to promote fairness in the model's predictions while maintaining classification performance. Our study focuses on the impact of various classification techniques, such as bias initialization, class weights, and over/under sampling, on the model's ability to detect fraud. We also investigate debiasing techniques, including dimension reduction and adversarial debiasing, to mitigate potential bias in the model's predictions. The performance of the models is evaluated on a large-scale fraud dataset, and fairness is assessed based on the false positive rates across different age groups. The results demonstrate that adversarial debiasing, when combined with appropriate classification techniques, can effectively reduce bias and improve fairness in fraud detection models. Overall, this study provides valuable insights into the challenges and potential solutions for building fair fraud detection models.

1. Introduction

The rapid advancement of digital technologies in our society has brought forth a corresponding increase in the risk of fraudulent activities. This heightened concern holds particular significance for the banking industry, as countless individuals rely on these institutions to safeguard their hard-earned savings.

While it is essential for banks to establish robust security measures, it is equally crucial to maintain the accessibility of financial resources. Indiscriminately flagging every transaction or credit application as fraudulent and subsequently blocking them would undermine the trust and usability of banking services. Therefore, it becomes imperative to strike a delicate balance that ensures both adequate security and accessibility.

Adding to the complexity of this issue is the ethical dimension associated with the development of models. While it is possible to create models that maximize performance, they may inadvertently yield biased outcomes. For instance, such models might disproportionately deny individuals over the age of 50 access to new lines of credit, leading to unjust discrimination. Consequently, careful consideration must be given to these ethical implications before deploying any model of this nature.

The primary objective of this project is to construct a comprehensive model that addresses these considerations. To achieve this, the Bank Account Fraud Dataset Suite (NeurIDP 2022), developed by Feedzai, will be utilized for training and evaluating the model based on the aforementioned criteria. This dataset has been specifically designed to simulate real-world credit applications and banking data, incorporating discrepancies between groups, with a particular focus on customer age.

2. Objectives

2.1. Metrics

The main objective of this paper is to explore adversarial debiasing as a method of creating a fair model, comparing it with other techniques and considering its additional value. The performance of different methods will be assessed based on two primary goals: maintaining the model's effectiveness as a classification model and ensuring its fairness.

To evaluate the classification performance of the model, several key metrics will be considered, namely Recall, Precision, and AUROC. These metrics have been chosen due to their relevance to the problem at hand and are definied below.

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

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

AUROC: (Area Under the Receiver Operating Characteristic Curve) evaluates the model's ability to discriminate between positive and negative cases across various probability thresholds.

2.2. Evaluation Criteria

For this paper, stricter restrictions will be placed on Recall, with a required value of 0.4 or greater, to ensure that a minimal number of fraudulent transactions are missed. Moreover, Precision should exceed 0.035, representing the financial institution's cost of processing a false positive transaction. Additionally, confusion matrices will be employed to gain a better understanding of the model's predictions. These minimum values serve as the initial criteria for the classification model, which will then be halved when evaluating the validity of the de-biased model. Any models falling outside of these ranges will be considered invalid and excluded from this paper.

It should be noted that these values and their relative importance may vary depending on the specific needs of the financial institution and the model under development. The chosen values were determined following a discussion with industry expert Donna Turner, who highlighted that the weight of a true positive versus false positive prediction can range from 3:1 to 30:1, depending on the needs and regulatory considerations of a given model.

When determining the fairness of a model, two different techniques will be employed. Firstly, as provided by Feedzai, the false positive rate across all sensitive groups must differ by no more than 5%. Additionally, a stricter requirement will be imposed by testing for False Positive Parity, which similarly assesses the equality of false positive rates across groups (Hardt et al., 2016). In this paper, the sensitive groups will be defined based on different age categories as established by Feedzai. This choice was made due to the observed variation in false positive rates across age groups in the baseline model compared to the other suggested sensitive groups of income and employment status. By focusing on a single group, the aim is to facilitate a clear understanding of the findings. A model demonstrating parity across all groups will be considered valid, while a model that falls within the 5% range of difference but lacks parity will be deemed "plausible."

It may initially appear counter-intuitive to employ a low precision cut-off when the goal is to assess False Positive Rates (FPR) within each group. However, it is important to note that the objective is to achieve equal FPRs across all groups, rather than aiming for an overall low FPR.

3. Data

The dataset utilized in this study, as documented by Feedzai (Jesus et al., 2022), comprises a comprehensive collection of 32 distinct features. These features encompass a range of data types, including categorical variables such as housing status, payment type, and transaction source, alongside numerical variables such as account velocity, session length, and credit risk score. The dataset encompasses a substantial sample size of 1,000,000 individual transactions, providing an extensive foundation for analysis and model development.

3.1. Ingestion

The data ingestion process was straightforward for this project. As the purpose of the dataset provided by Feedzai was to promote accessibility to typically classified data, acquiring it involved a simple download and importation into my Jupyter notebook. However, it's worth noting that the dataset is quite large, and managing its memory became a challenge. To address this issue, I simply compressed the dataset into a zip file.

3.2. Exploration

The dataset utilized in this project comprises 30 features, in addition to the target column. These features can be categorized into different sets.

Our target variable is binary, with "True" representing a fraudulent credit application and "False" indicating a non-fraudulent transaction. However, it is important to note that the distribution of the target variable is heavily imbalanced, as depicted in Figure 1. Approximately 1% of the transactions in our dataset are labeled as fraudulent, while the majority are labeled as non-fraudulent.

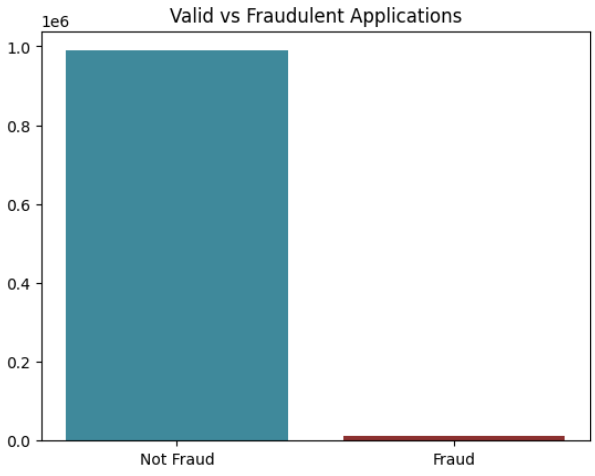

Figure 1. Target Value Frequencies

With regard to our dataset's features, firstly there are six categorical features: "payment type," "employment status," "housing status," "application source," "month," and "device operating system." Figure 13 illustrates the distribution of these categories within the dataset.

Secondly, there are six boolean features: "email is free," "valid home phone," "valid mobile phone," "has other cards," "foreign request," and "keep alive session." Figure 14 showcases the ratio of True versus False values for each feature separated into fraudulent and non-fraudulent cases.

The dataset also includes nine features representing count data. These features include "months at previous address," "months at current address," "customer age," "number of applications in zip code over 4 weeks," "bank branch count 8 weeks," "date of birth distinct emails," "credit risk score," "bank months count," and "device distinct emails." Additionally, there are nine continuous value features: "income," "name email similarity," "days since request," "intended balcony amount," "velocity 6 hours," "velocity 24 hours," "velocity 4 weeks," "proposed credit limit," and "session length in minutes." To compare the distributions of fraud versus non-fraud cases, kernel density plots were generated for these continuous value features (Figure 15).

3.3. Preparation

3.3.1. Missingness

The initial step in data preparation involved addressing missing values. The "device fraud count" column, which was not included in the previous description, was entirely composed of 0 values and thus provided no valuable information. Consequently, this column was removed from the dataset.

Three columns, namely "current address month count," "session length in minutes," and "distinct device emails 8 weeks," contained placeholder values of -1 to represent missing data. To determine whether these missing values occurred randomly or non-randomly, a chi-squared contingency test was conducted for each column. The test compared the distribution of missing values between instances labeled as fraud and those labeled as legitimate. A significance level (alpha) of 0.01 was chosen for the test.

Results indicated that two columns, "session length in minutes" and "device distinct emails," exhibited missing values randomly and had relatively few instances with missing data compared to the overall dataset size. Consequently, instances with missing values in these columns were removed from further analysis.

However, the third column showed non-random missingness, with a p-value of 0.004. To address the non-random missingness in the "current address month count" column, we utilized the IterativeImputer class from the scikit-learn

library. This approach leverages an iterative process to estimate missing values based on the observed data. The imputer fits a regression model to the dataset, predicting the missing values while considering the other features as predictors (Pedregosa et al., 2011). This iterative process ensures that the imputation accounts for the relationships and patterns present in the data.

Importantly, the fitted imputer can be stored and applied to new data during the transformation stage before feeding it into the model. By doing so, we ensure consistency and maintain the integrity of the imputation process, even when working with new, unseen data.

Understanding whether missing values occur at random or not is crucial for accurately analyzing and modeling fraudulent credit applications. Ignoring missing not at random (MNAR) values, which may indicate deliberate omissions related to fraudulent transactions, can potentially introduce biases and compromise the effectiveness of our model. By acknowledging the possibility of MNAR missingness and employing appropriate imputation techniques that capture the potential information within those missing values, we can enhance the integrity of our analysis and ensure that valuable insights regarding fraudulent activities are not overlooked. Effectively addressing MNAR missingness allows us to develop a more robust and comprehensive model that accurately captures the complex dynamics of fraud detection in the banking industry.

3.3.2. SCALING

After handling missing values, the next important step in our data preparation process was to scale our numerical features. This step is crucial because our model lacks inherent context about the magnitude and range of the features, making it challenging for the model to learn effectively when features have significantly different value scales. For instance, in our dataset, the "name_email_similarity" feature ranges from -1 to 1, while the "credit_risk_score" feature can go up to 400.

To address this issue, we applied a simple min-max scaler to the numerical data. This scaling technique linearly transforms the values of each feature, setting the minimum value to 0 and the maximum value to 1, while maintaining the proportionality of the values in between.

Another common issue in numerical data is the presence of heavily skewed features. When a feature is skewed, it can make it challenging for a model to make accurate predictions on the "tail-side" values. To address this behavior, we performed a log transformation on the skewed features. The log transform helps to reduce the impact of extreme values and brings the distribution closer to a normal distribution.

Figure 2 illustrates an example of the effects of log transformation and min-max scaling applied to our dataset. These scaling techniques ensure that the features are on a comparable scale and reduce the impact of skewed distributions, facilitating the learning process for our model.

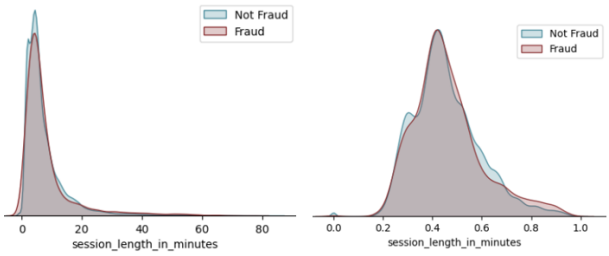

Figure 2. Visual of Numerical Transformation

3.3.3. ENCODING

The final step in preparing our data to work effectively with our model is to encode the categorical features. This process is necessary because our model lacks contextual understanding of the categorical values. For instance, payment types represented as A, B, C, or D hold no inherent meaning for our model. Therefore, encoding allows us to transform these categorical values into a numeric representation.

In this project, we employed one-hot encoding as the chosen method. One-hot encoding involves creating boolean columns for each unique value in all of our categorical features. One-hot encoding was preferred over other methods, such as label encoding, because the categories should not be represented as being "more" or "less" than each other, as they are nominal categories. It is important to note that nominal categories lack a natural ordering or hierarchy. For instance, it makes no sense to compare categories using mathematical operations, such as:

- $Check > Card$
- $2 * Checks = Card$
- $Check + Card = Cash$

Therefore, one-hot encoding is a suitable approach as it represents the categories as binary values of "is" or "is not."

Fortunately, in our project, the number of unique values in our categorical features is not extensive. Thus, the common issue associated with one-hot encoding, where an excessive number of columns is created, is not a concern. By applying one-hot encoding, we retain the necessary information encoded in a format that our model can effectively learn from, ensuring the categorical features contribute meaningfully to our predictive modeling efforts.

4. Methodology

4.1. Classification Techniques

When working with highly imbalanced data, such as our fraud dataset where only 1% of the data is labeled as fraud, it can be challenging for a machine learning model to effectively learn patterns for the minority class. To address this challenge, I employed a combination of techniques to help our model better recognize instances of fraud. These techniques included initializing bias, using class weights, and considering over/under sampling.

Bias initialization is a technique where the model's initial output is set to reflect the ratio of fraud instances to non-fraud instances in the dataset. By providing this initial bias, the model can focus on learning to classify the imbalanced data effectively, rather than spending a significant number of iterations to understand the data distribution. The effects of implementing an initial output bias on the loss function of our baseline model are demonstrated in Figure 3.

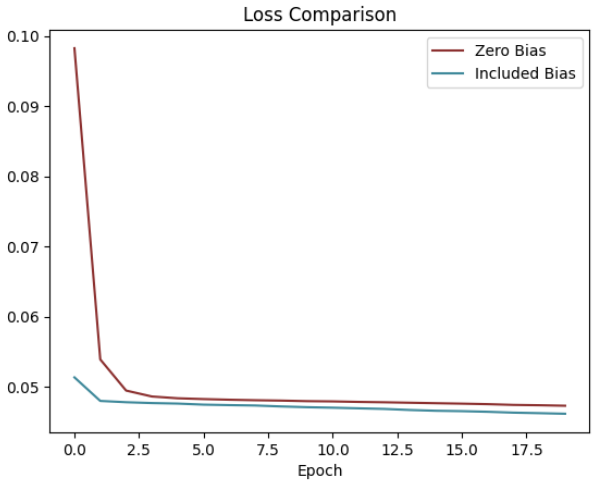

Figure 3. Effects of bias initialization on the loss function

Another technique used is the application of class weights, which adjusts the value of the loss function to penalize or reward certain classes. In our case, the baseline model uses Binary Cross Entropy (BCE) as the loss function.

$$BCE = \frac{-1}{N}\sum_{i=1}^{N}\left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right)$$

$$N = number\,of\,instances$$
$$y_i = instance\,target\,value$$
$$p_i = prediction\,probability$$

However, by introducing class weights, we transform it into Weighted Binary Cross Entropy (WBCE). The WBCE multiplies the resulting value by a weight that depends on the class of the true value. A higher weight is assigned to the fraudulent class (class 1) to increase the penalty for misclassifying fraud instances.

$$WBCE = \frac{-1}{N}\sum_{i=1}^{N}\left(w_1 * y_i \log(p_i) + w_0 * (1-y_i)\log(1-p_i)\right)$$

$$w_0 = weight\,for\,class\,0\,(Non-Fraudulent)$$
$$w_1 = weight\,for\,class\,1\,(Fraudulent)$$

The weighted BCE greatly increases the number of fraudulent cases identified by our model compared to the non-weighted BCE. The confusion matrices for both loss functions, shown in Figures 4 and 5, demonstrate the improved performance of the weighted BCE in identifying fraudulent cases.

		Actual	
		False	True
Predicted	False	185033	2008
	True	18	41

Figure 4. Normal BCE

		Actual	
		False	True
Predicted	False	151864	425
	True	33187	1624

Figure 5. Weighted BCE

Additionally, I mention the technique of over/under sampling for completeness. However, this technique did not result in any valid models meeting our defined criteria. Oversampling involves adding duplicate samples of the underrepresented class, while under-sampling removes samples of the overrepresented class. Although this technique aims to balance the class ratio, it can lead to the model learning that there are more positive cases in the data than there really are, resulting in a high false positive rate. Due to the negative impact on precision and the violation of our precision requirements, this technique was not implemented in any final tests.

By leveraging these classification techniques, we improve the model's ability to learn and accurately classify the minority class of fraudulent transactions. The initialization of bias, utilization of class weights, and consideration of over/under sampling are effective strategies for addressing the challenges posed by imbalanced data and promoting the detection of fraudulent instances. These techniques, as well as other methods of handling imbalanced data, are covered extensively in (He & Garcia, 2009).

4.2. Debiasing Techniques

4.2.1. Dimension Reduction

To promote fairness in the model, a common technique is to remove the sensitive groups, thereby preventing the model from utilizing that data as a predictor, thus reducing the chances of discrimination against those groups. However, it is important to consider that the remaining features used for training the model may still be correlated with the removed feature or introduce bias into the predictor.

In line with this rationale, I created a few separate datasets to explore the impact of removing the sensitive category and reducing the number of potentially correlated features. One dataset involved simply removing the sensitive category "customer age." Another dataset utilized principal component analysis (PCA) to reduce the dimensionality of the dataset to eight features, aiming to minimize the potential correlation between features and the age variable. These datasets were saved to be used for comparison with and combination with adversarial debiasing.

I should note that I conducted similar iterations, such as removing highly correlated features and various combinations of feature removal and PCA. However, these iterations did not perform well as classification models in their baseline tests and will not be further discussed in this paper.

By exploring different dimension reduction methods and other debiasing methods, we can better understand the impact of removing sensitive groups and reducing feature correlation on the fairness and predictive power of our model. This comprehensive exploration contributes to the robustness and reliability of our findings and allows us to make informed decisions regarding the most suitable approach for promoting fairness in our model.

4.2.2. Adversarial Debiasing

I soon realized that feature engineering and other manual processes might not be a practical solution due to the significant disparity between groups. To overcome these challenges, I adopted adversarial learning, a method described in the literature (Zhang et al., 2018) for promoting fairness in machine learning models. Adversarial learning involves training two separate models: the deployment model and the adversarial model. The deployment model is trained using standard techniques for the desired task, such as fraud classification. The adversarial model is trained on the predictions made by the deployment model, predicting the sensitive groups that should be treated fairly.

By incorporating the loss of the adversarial model into the overall loss function of the deployment model, we encourage the deployment model to make predictions that are more difficult for the adversarial model to identify. In other words, the deployment model becomes more robust and less dependent on sensitive attributes when making predictions. This approach helps to promote fairness by ensuring that the deployment model provides more equitable predictions across different classes, without the need for extensive feature engineering or the loss of significant amounts of data.

By adopting adversarial debiasing, we can address the challenges posed by class imbalance and enhance the fairness of our fraud classification model. This method leverages the power of adversarial learning to encourage the deployment model to make unbiased predictions while maintaining high predictive performance.

4.3. Procedures

4.3.1. Data Preprocessing

Due to the large size of the dataset, it was possible to split it into training, validation, and test sets without concern about insufficient samples. The dataset was divided using a 64/16/20% split, with 64% allocated for training, 16% for validation, and 20% for testing. The training set was utilized to train our model, the validation set was employed for performance measurement during training and tuning, and the test set was exclusively used for final evaluations to assess the model's performance on unseen data.

Following the dataset splitting, the data transformations described in the data preparation section were applied. Initially, the imputer and scalers were fitted to the training set only. This approach was adopted because the validation and test sets represent unseen or future inputs, and referencing them during the fitting process would compromise the in-

tegrity of the evaluation. Consequently, it is expected that a few values in the validation and test sets may fall outside the 0 to 1 range after min-max scaling. However, this accurately reflects how our model will respond to new inputs that lie outside the initial extremes.

Once the transformations were fitted to the training set, they were consistently applied across all datasets. This ensures that the same transformation rules learned from the training set are uniformly applied to the validation and test sets. By maintaining this consistency, a fair and representative evaluation of the model's performance on unseen data is achieved.

By following this procedure, we ensure that our model is trained on a sufficiently large dataset, and the data transformations are applied consistently across all relevant subsets, enabling accurate evaluation and analysis of the model's performance.

4.3.2. Model Selection

In order for adversarial debiasing to be applied, the selected architecture must be trained using a gradient-based learning method. In this study, two architectures, namely a Dense Neural Network and Logistic Regression, were compared for model selection.

Logistic regression was initially considered due to its suitability for binary classification tasks, such as our target variable with "True" or "False" labels. However, it was found that logistic regression was not a viable option for our problem, likely due to the non-linear nature of the data and the target variable.

As a result, a Dense Neural Network model was chosen as the architecture for the remainder of this paper. Dense Neural Networks are well-suited for capturing non-linear patterns in data, making them a suitable choice for our problem. Additionally, these models meet the requirement of being trained using a gradient-based learning method. The Dense Neural Network also performed well as a baseline model, further supporting its selection for the subsequent analysis and application of adversarial debiasing.

For the adversarial model, we also employed a Dense Neural Network architecture. The final layer of this adversarial model utilizes a softmax activation function, which outputs probabilities for each age value based on the predictions of our main model.

4.3.3. Training and Adversarial Debiasing

The training of the models involved the following procedures.

Firstly, the deployment model was trained on the transformed dataset using the Weighted Binary Cross Entropy loss function, as mentioned before. This step aimed to create a classification model that is effective at detecting fraud.

Next, the predictions of the deployment model were used as input for the adversarial model to predict the customer age for each instance. The loss function used for the adversarial model is Categorical Cross Entropy, which is similar to Binary Cross Entropy but summed over all classes.

This completes the pretraining step of the adversarial debiasing process.

After the initial training of both models, the loss function of the deployment model was adjusted to incorporate the loss of the adversarial model. Specifically, the weighted binary cross-entropy (WBCE) loss function was used for the deployment model, while the adversarial model utilized categorical cross-entropy (CCE) as its loss function. The hyperparameter α controls the strength of the impact of the adversarial loss on the overall loss. Additionally, the projection of the deployment model's loss onto the adversarial model's loss was included to prevent the weights of the deployment model from moving in a direction that benefits the adversary. Figure 6, provided by (Zhang et al., 2018), offers a visual representation of this concept.

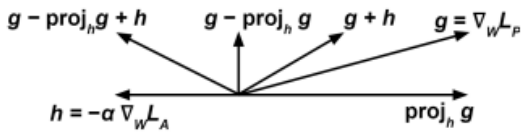

Figure 6. Vector Representation of Adversarial Loss and the projection

Therefore, the updated deployment loss function can be expressed as follows:

$$loss = WBCE(d) - \alpha * CCE(a) - proj_{\nabla_W L_a}(\nabla_W L_d)$$

The debiasing process involves iterating over the following steps until a stopping criterion is met:

1. Train the adversarial model using the deployment model's predictions while keeping the weights of the deployment model constant.
2. Train the deployment model with its new loss function, taking into account the predictions of the adversarial model and keeping the weights of the adversarial model constant.

The stopping criterion used in this project is either 10 iterations with less than a 0.00001 change in either loss function or 10 iterations that fall outside the defined classification parameters. Weights are then reset to their values in the best performing iteration.

4.3.4. HYPER-PARAMETER TUNING

To determine the optimal combination of hyper-parameters, a simple grid-search technique was implemented. This technique involved iterating over all possible combinations of hyper-parameters and performing the training and adversarial debiasing steps described earlier. Notably, the hyper-parameter tuning was conducted in two separate steps. First, the grid search was performed to find the best hyper-parameters for the classification models, and then another grid search was conducted to determine the optimal values for the debiasing process.

For the initial training of the deployment model, the following hyper-parameters were included. Class weights and initial bias, as described in the 4.1 Classification Techniques section, were considered. Different combinations of class weights, emphasizing varying degrees of importance on the fraud class, were tested. It is important to note that heavier weights were never applied to the non-fraudulent class. The initial bias was either included or not, as the aim was to provide the model with an accurate understanding of the desired output.

Additionally, for the deployment model, an array of hidden layers was provided, with variations in the number of layers and the number of nodes in each layer. Hidden layers play a crucial role in neural networks, as they are responsible for finding complex patterns and enabling the network to perform well. Each hidden layer also had an activation function, which transformed the weighted sum of the nodes in the previous layer using a non-linear function. The grid search included a comparison between the "Tanh" and "ReLU" activation functions.

The same hyper-parameters were iterated over for the adversarial model, with the exception of class weights.

As gradient-based models are trained, the learning rate determines the speed at which the weights are adjusted based on the loss function. Choosing an appropriate learning rate is critical for model performance. To take advantage of both quick changes with a large learning rate and convergence abilities with lower learning rates, the TensorFlow ReduceLROnPlateau class was utilized. This class allowed the monitoring of loss and the reduction of the learning rate over time (Abadi et al., 2015). The grid search involved testing different initial learning rates for each model, noting that each model had its own learning rate.

In the debiasing grid search, different learning rates were also iterated over for each model. However, unlike the classification models, the learning rate for the debiasing step was not reduced over time due to the changing states of the adversarial model. The hyper-parameter α in the debiasing process was also included in the grid search.

Lastly, although not considered a hyper-parameter, both the full dataset and the dataset with the sensitive group excluded were tested with these methods. The dataset obtained through Principal Component Analysis (PCA) was not included in this analysis due to time constraints and is left for future research.

During the hyper-parameter tuning process, several models were saved based on specific evaluation metrics. The metrics used for model selection were Recall, Precision, AUROC, and the sum of these three metrics. These evaluation metrics were calculated on the validation set, which the models were not trained on. If a model outperformed the previous best model in terms of any of these metrics and met the validity criteria, it was saved for further analysis. The validity criteria were determined based on the constraints set for the classification model.

After saving multiple models, the confusion matrices and metric values were examined to evaluate their performance. The model with the best overall performance, considering all the evaluation metrics and the constraints, was chosen as the final model.

5. Results and Analysis

Once the hyper-parameters were selected through the tuning process, the chosen models were retrained using the combined training and validation sets. This step allowed the models to benefit from a larger amount of labeled data for training. After retraining, the performance of the models was evaluated on the unseen test set. This evaluation on the test set provided an unbiased assessment of the models' generalization ability and their performance on new, unseen data.

5.1. Classification Results

In the evaluation of our models, we compared their classification performance before and after debiasing.

For the full dataset, including the customer age feature, the pretrained model achieved a recall of 0.547 and a precision of 0.103, as can be computed using the confusion matrix in Figure 7. After debiasing, the model achieved a recall of 0.216 and a precision of 0.229, as can be computed using in the confusion matrix in Figure 8. Although there is a decrease in recall, the precision has significantly improved, and both values fall within the valid classification range.

In terms of AUROC score, both the pretrained and debiased models outperformed the highest Feedzai baseline score of 0.877, achieving scores of 0.891 and 0.890, respectively. However, it is important to note that AUROC can be misleading for highly imbalanced datasets, and it is recommended to consider multiple metrics when evaluating such

		Actual	
		False	True
Predicted	False	233688	1240
	True	13039	1496

Figure 7. Confusion Matrix of Final Model on Full Dataset before debiasing

		Actual	
		False	True
Predicted	False	244738	2145
	True	1989	591

Figure 8. Confusion Matrix of Final Model on Full Dataset after debiasing

data (Bekkar et al., 2013).

Regarding the model trained on the dataset where the age feature was removed, the results prior to debiasing showed a recall of 0.459, a precision of 0.105, and the confusion matrix in Figure 9. This model did not surpass the baseline with an AUROC score of 0.876. Additionally, after debiasing, this model did not meet the criteria for both valid classification and plausible debiasing. Due to time restrictions and its lack of validity, the model was not trained on the combination of datasets, and Figure 9 shows its results on the validation dataset.

5.2. Fairness Results

For this project, fairness is defined in terms of the false positive rate (FPR) across different customer age groups. The age groups, binned by Feedzai, are represented in our

		Actual	
		False	True
Predicted	False	176628	1154
	True	8357	979

Figure 9. Confusion Matrix of Final Model on Partial Dataset before debiasing

dataset as the values 10, 20, 30, 40, 50, 60, 70, 80, and 90. We aim to ensure fairness by examining and minimizing the differences in FPRs across these age groups.

To assess fairness, we set a maximum allowable difference of 5% across all FPRs, as defined by the dataset's creators. Additionally, we apply a stricter requirement by testing for False Positive Parity, which evaluates the equality of false positive rates across groups.

For the validation set, we compare the maximum difference in FPRs between our pretrained models on the full and partial datasets. The model trained on the full dataset had a maximum FPR difference of 14.35%, while the model trained on the partial dataset had a maximum difference of 11.37%. This confirms our previous observation that excluding certain features can help mitigate bias.

Although these differences still exceed the recommended 5% maximum, we address this by employing adversarial debiasing. Figures 10 and 11 illustrate the change in the FPR difference over the adversarial debiasing process, specifically on the unseen test set. We use the F1 score to balance recall and precision in selecting the best model for classification, and then we evaluate the FPR difference for fairness. The final model is chosen based on the iteration where the difference between the F1 score and FPR difference is maximized. The vertical lines in each figure represent the first epoch where the model is considered plausible in terms of fairness and the final epoch where the model satisfies our classification constraints.

Upon initial observation of the test set results, it appeared that we failed to meet the fairness plausibility requirements. However, further analysis of the confusion matrices revealed that the age group labeled as 90 had only a few positive predictions, with 2 false positives and a single true positive. To ensure a meaningful comparison, I implemented a sam-

ple size significance test to assess whether the groups had enough true predictions to be considered in comparisons. By considering only the groups with a significant sample size, we were able to eliminate large jumps in the differences between epochs, as depicted in Figure 11, and achieve plausible fairness results on our test set. This, combined with the results on our validation set, gives us sufficient confidence to classify the fairness performance of this model as 'plausible'.

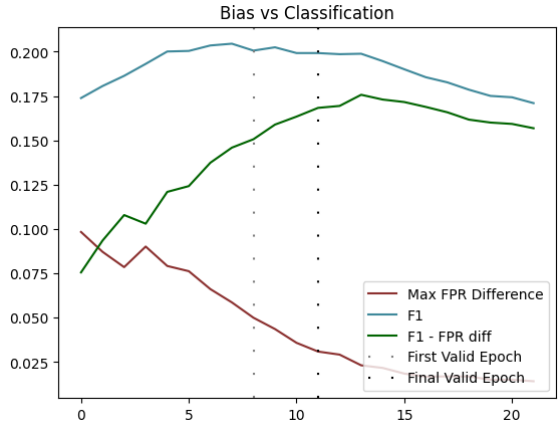

Figure 10. Debiasing process on the validation set.

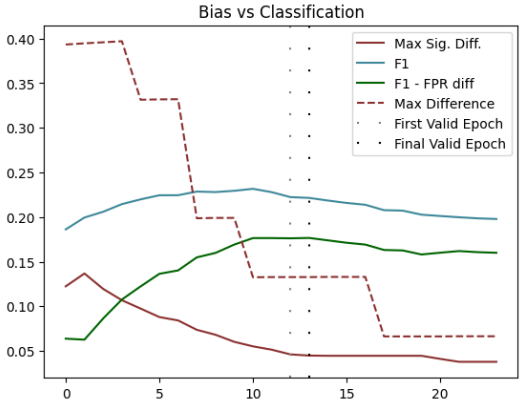

Figure 11. Debiasing process on the test set.

Upon conducting a comparison of false positive rates (FPR) across different age groups, it was found that parity, or equality, in FPR was only achieved in 3 out of the 36 possible pairs. To assess the statistical significance of the differences, a standard z-test was performed to compare proportions. A p-value greater than 0.01 was considered as an indication of equality between the FPRs.

Based on the results of the z-test, it was determined that the model did not meet the parity requirements, as the FPR differences were statistically significant. This means that there were significant disparities in the false positive rates across the different age groups, indicating potential bias in the model's predictions.

5.3. Model Analysis

In the model analysis, SHAP (SHapley Additive exPlanations) values were used to gain insights into the behavior of the Dense Neural Network model. SHAP values provide an understanding of feature importance by calculating the impact of each feature on the model's predictions. Figures 16 and 17 show the SHAP values for each feature before and after debiasing.

After debiasing, the SHAP values for all features were reduced, indicating that each feature had less influence on changing the predicted value. This is consistent with the observation of a significant decrease in false positives after debiasing and the imbalanced nature of the dataset. As the model became more cautious in predicting fraudulent cases, the overall importance of features decreased.

Additionally, the SHAP values also revealed a decrease in the relative importance of customer age compared to other features. This aligns with the debiasing step, where the model was trained to make it harder to identify the age group of the predicted values. Interestingly, there was a notable difference in the impact of the number of months at a previous address and the current address, with the current address showing a larger change. This could be attributed to the fact that the older population, who were more likely to be labeled as fraudulent before debiasing, had a longer duration at their previous address, which may have been a characteristic associated with older clients.

In addition to comparing the SHAP values before and after debiasing, the change in feature importance for each feature is also analyzed. This change is quantified by calculating the average SHAP values for each feature divided by the total of the inputs. This ratio represents how much each feature influences the prediction. The analysis is performed separately for the pretrained model and the debiased model, with the pretrained results subtracted from the debiased results. This approach allows for a more nuanced understanding of how the model's behavior changes during the debiasing process.

Figure 12 presents the features with the largest change in importance. The analysis reveals that "customer age" experiences the most significant decrease in influence, indicating that the model becomes less reliant on this feature for making predictions. On the other hand, "is foreign request" shows the largest increase in influence, suggesting that this feature gains more importance in the debiased model. Examining these shifts in feature importance provides valuable insights into how the model's behavior is altered during the debiasing process.

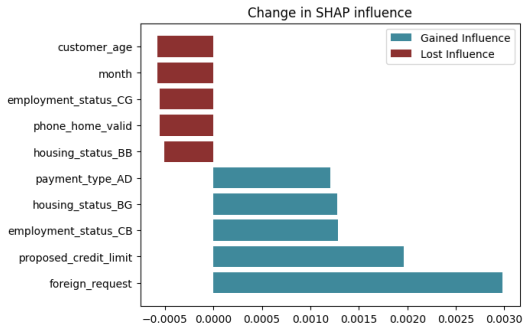

Figure 12. Effects of debiasing on feature influence.

A deeper understanding of how features were labeled would offer more intuitive insights into the changes observed alongside "customer age" during the debiasing process. However, to ensure the anonymity and privacy of the data used to train the generation formulas, the specific meanings of the labels are omitted. While this limitation restricts our ability to pinpoint the exact reasons for the changes in other features, the overall analysis still provides valuable insights into the effectiveness of adversarial debiasing and its influence on feature importance in the model.

5.4. Result Analysis

In conclusion, adversarial debiasing has been shown to be a plausible method for creating a fair fraud detection model. When combined with techniques for addressing imbalanced data, such as class weights and initial output bias, it maintains an acceptable level of performance while promoting fairness between different groups.

While combining adversarial debiasing with other techniques, such as removing sensitive features, resulted in a performance drop outside of the defined restraints, it still holds potential in other scenarios and should be explored further.

Depending on the specific requirements and goals of a financial institution, additional actions may be necessary to ensure a fully fair model. One possible approach could be setting different probability thresholds for each group based on the ROC curve, thereby achieving a desired false positive rate for each group. However, it is important to carefully consider the decision-making implications and potential biases introduced by such actions, as it goes beyond the scope of this paper to delve into the decision science behind these choices.

In summary, this study successfully built a robust classification model while significantly reducing bias, demonstrating the effectiveness of adversarial debiasing in promoting fairness in fraud detection.

5.5. Future Work

While this project demonstrated the effectiveness of Adversarial Debiasing in mitigating bias in imbalanced data, there are several avenues for future work that could further improve upon the results.

Firstly, it would be interesting to investigate the combined effects of Adversarial Debiasing and over/under sampling techniques. Although initial attempts with oversampling resulted in an increased number of false positives, falling outside our parameters for a valid model, the observed decrease in false positives after debiasing suggests that a combination of these techniques could potentially lead to improved fairness outcomes. Understanding the interplay between different approaches for handling imbalanced data and debiasing methods could provide valuable insights for developing more robust and effective models.

Additionally, there are other techniques for handling imbalanced data that were not explored in this project but are discussed in the literature (He & Garcia, 2009). It would be worthwhile to investigate the integration of these techniques with Adversarial Debiasing. For example, ensemble methods that combine multiple models could be used in conjunction with Adversarial Debiasing to enhance the overall classification performance.

By addressing these areas of future work, we can further advance the understanding and implementation of Adversarial Debiasing for promoting fairness in machine learning models.

6. Deliverables

The deliverables for this project include the trained transformers (imputer and scaler) and the final model. These deliverables will enable the financial institution to build a pipeline for fraud detection in credit applications. The pipeline will consist of the transformers for preprocessing new data and the trained model for classification. With these deliverables, the institution can efficiently process new credit applications and obtain predictions on their fraudulent or non-fraudulent nature.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y. & Zheng, X. (2015). TensorFlow:

Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Bekkar, M., Djemaa, H. & Alitouche, T. (2013). Evaluation measures for models assessment over imbalanced data sets. *Journal of Information Engineering and Applications*, *3(10)*.

Hardt, M., Price, E., Price, E. & Srebro, N. (2016). Equality of opportunity in supervised learning. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I. & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*, Volume 29. Curran Associates, Inc.

He, H. & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21(9)*, 1263–1284.

Jesus, S., Pombal, J., Alves, D., Cruz, A., Saleiro, P., Ribeiro, R. P., Gama, J. & Bizarro, P. (2022). Turning the tables: Biased, imbalanced, dynamic tabular datasets for ml evaluation.

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

Zhang, B. H., Lemoine, B. & Mitchell, M. (2018). Mitigating unwanted biases with adversarial learning.

7. Referenced Figures

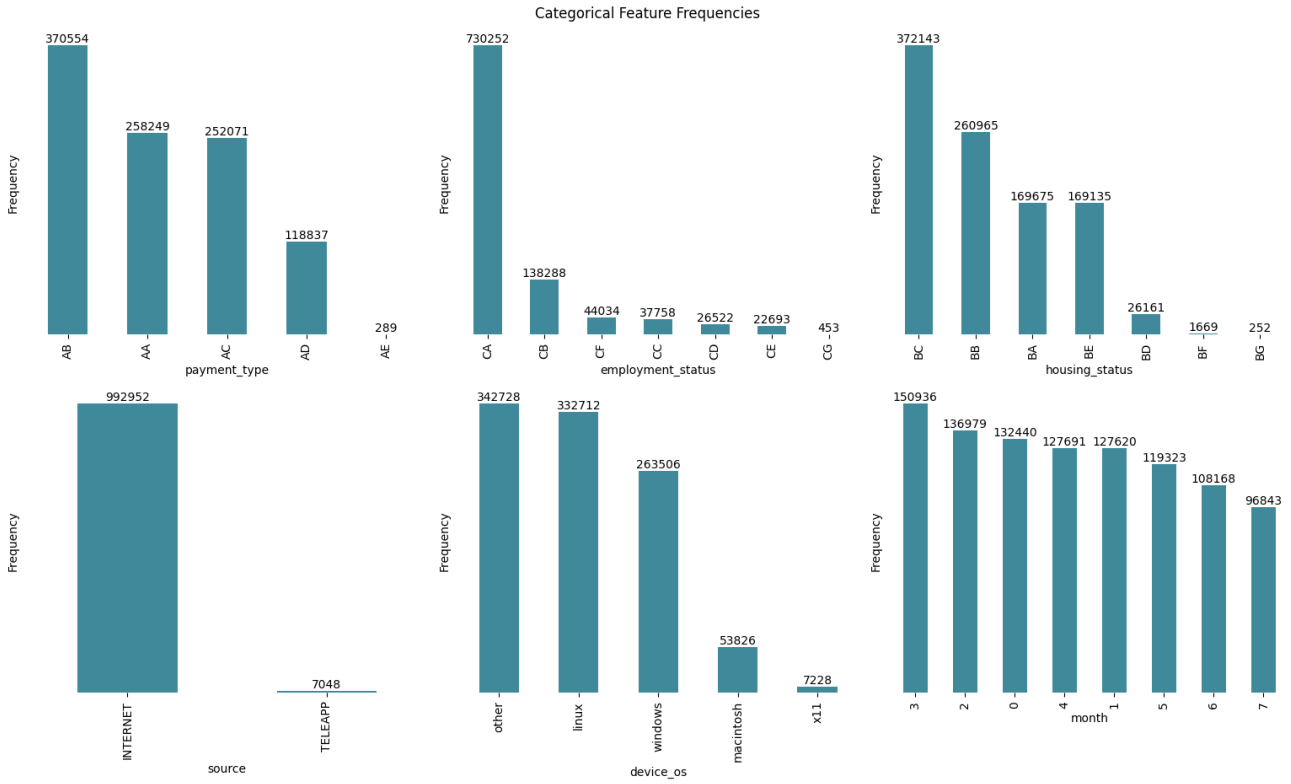

Figure 13. Categorical Feature Frequencies

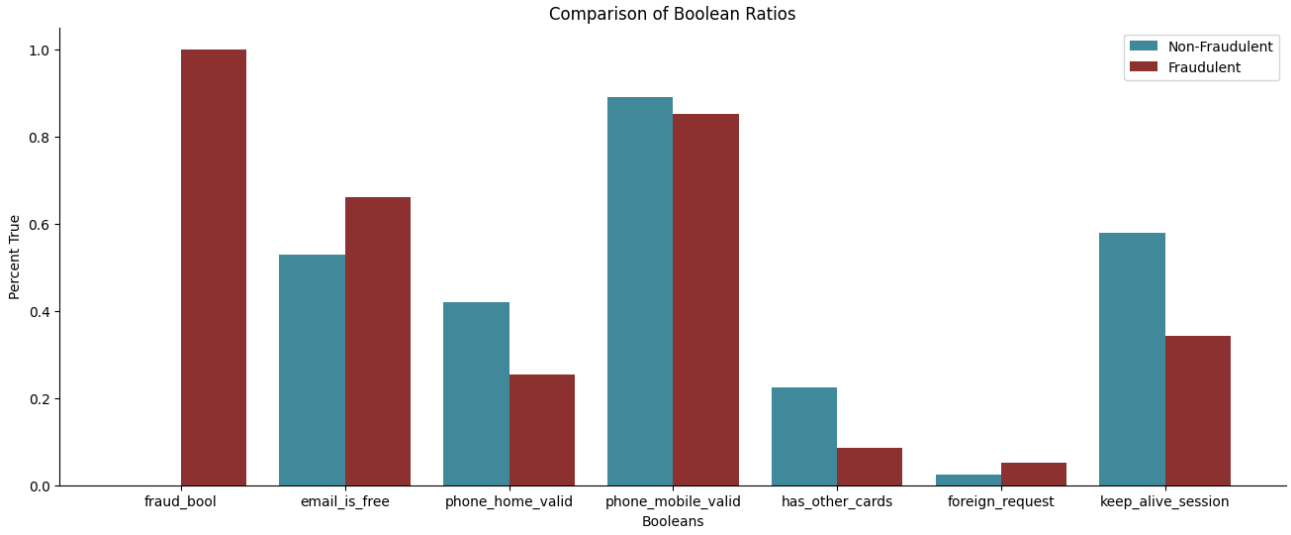

Figure 14. True : False, Boolean Feature Ratios

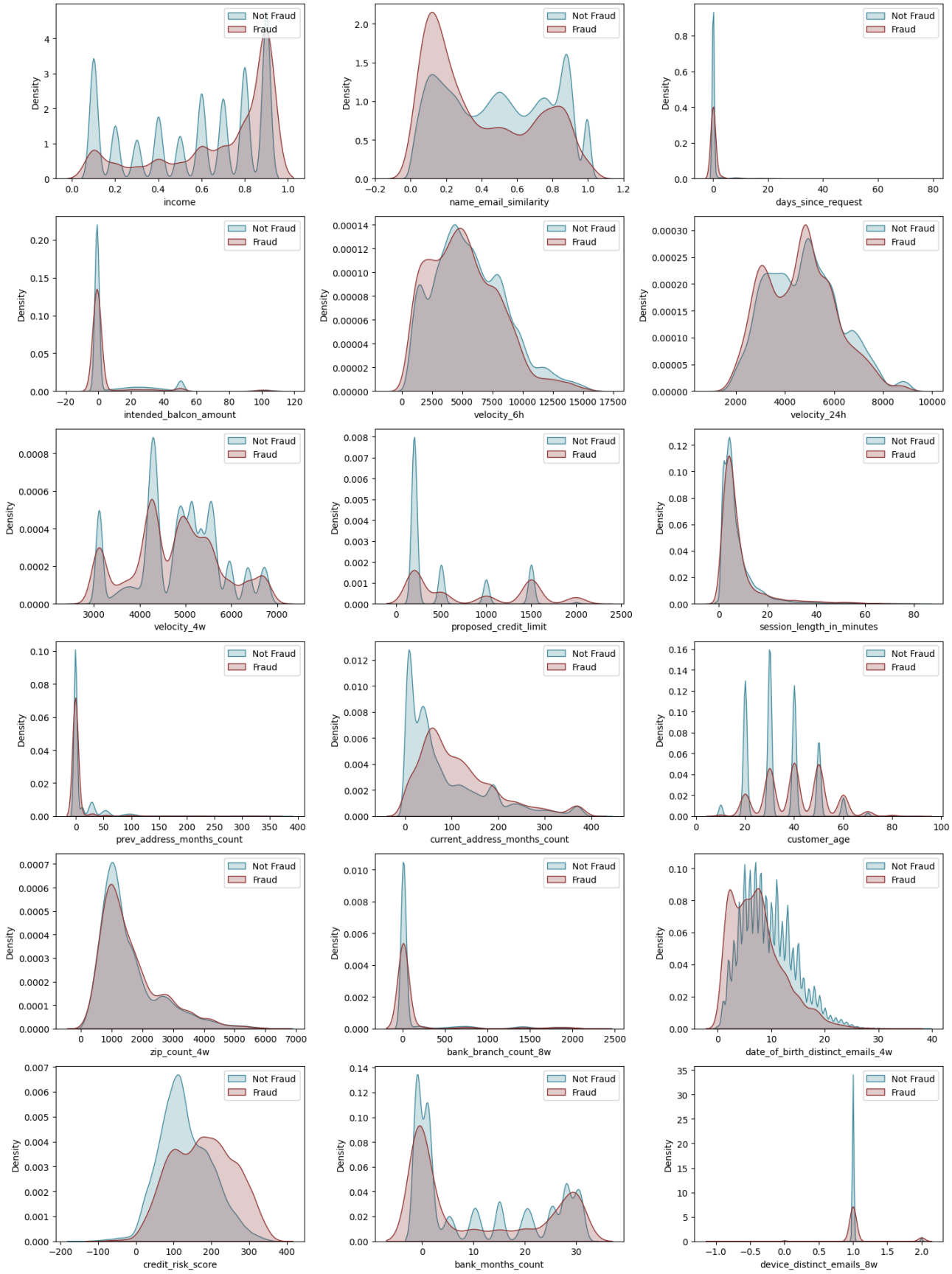

Figure 15. Numerical Feature Distributions

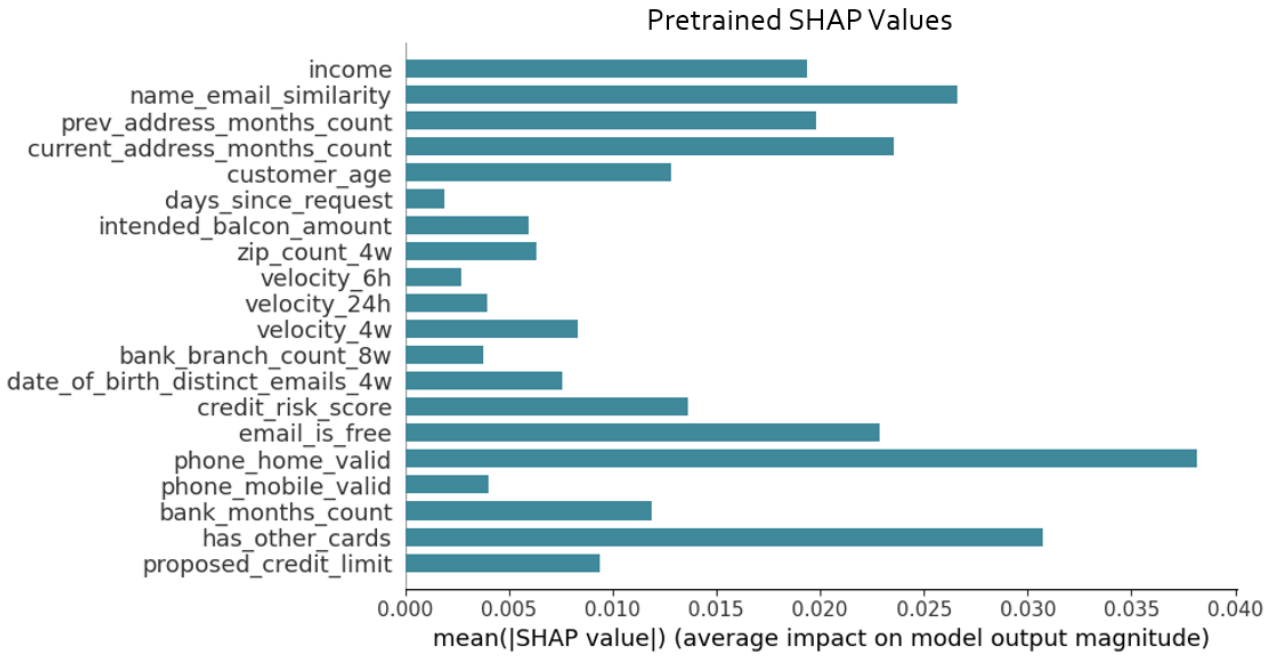

Figure 16. SHAP Values on the pretrained model.

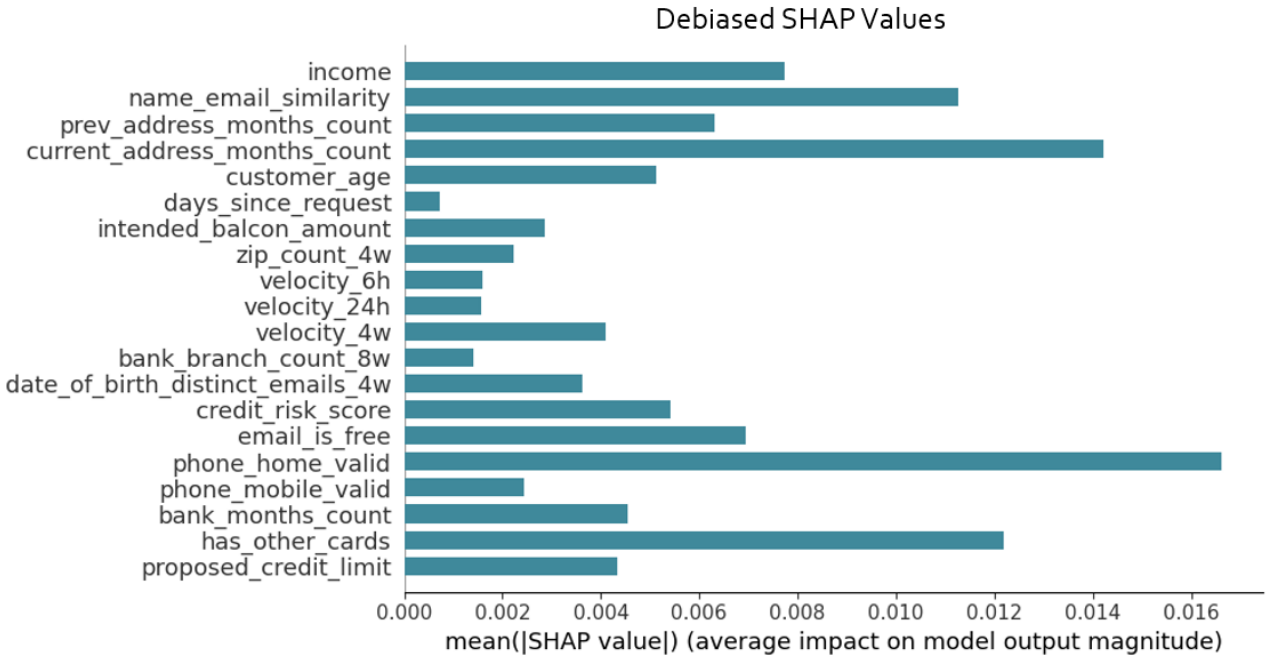

Figure 17. SHAP Values on the debiased model.

8. Self Assessment

This section is intended for the purpose of fulfilling project guidelines and providing specific details to my professors. It covers items outside the main scope of the project, such as self-reflection on personal learning objectives and course-related information. The content in this section is not intended for readers who are interested in the project itself, as it does not provide further insights into the project results or techniques employed.

8.1. Individual Learning Objectives

Throughout this project, I aimed to achieve specific learning objectives that were aligned with my personal interests and academic goals. Firstly, I sought to gain a deep understanding of techniques employed when dealing with highly imbalanced datasets. This involved learning about various methods for handling class imbalance and becoming familiar with the challenges and limitations associated with imbalanced data. The resources provided by Dr. Trafalis were instrumental in expanding my knowledge in this area, and I anticipate referring to them in future endeavors.

Secondly, I had a strong interest in exploring techniques for promoting fairness in machine learning models, an aspect that was not extensively covered in my coursework. I delved into the concept of fairness, exploring different fairness measures beyond those employed in this project. I also gained insights into methods aimed at enhancing fairness in models. As machine learning becomes increasingly integrated into workflows and decision-making processes, I believe this knowledge and practice in promoting fairness will prove valuable in addressing ethical considerations and ensuring equitable outcomes.

Overall, this project allowed me to deepen my understanding of imbalanced data handling techniques and expand my knowledge on fairness in machine learning models, thereby equipping me with valuable skills and perspectives for future endeavors.

8.2. DSA Skills

Throughout this project, several key ”Data Science” skills proved to be highly valuable. Firstly, my proficiency in Python played a crucial role in efficiently working with and manipulating the dataset. It allowed me to leverage various libraries, including TensorFlow, which was essential for implementing the adversarial debiasing technique. Python’s versatility and extensive ecosystem of data science libraries greatly facilitated the development and execution of the project.

Secondly, a solid understanding of the concepts and mathematics behind gradient-based loss functions was essential. This knowledge formed the foundation for implementing the adversarial debiasing approach, which relied heavily on optimizing the loss function to train the models effectively. Without a strong grasp of these concepts, it would have been challenging to navigate the intricacies of the technique and achieve the desired outcomes.

Lastly, a comprehensive understanding of machine learning metrics and statistical concepts played a vital role in analyzing the results. This included evaluating classification performance metrics such as precision, recall, and AUROC, as well as conducting hypothesis testing to assess the fairness of the models. These skills allowed for a rigorous evaluation of the model’s performance and provided valuable insights into its effectiveness and fairness.

Overall, proficiency in Python, knowledge of gradient-based loss functions, and a solid understanding of machine learning metrics and statistical concepts were pivotal in successfully completing the project and deriving meaningful insights from the results.

8.3. Learned Skills

One of the most valuable skills I acquired during this project was the ability to use the TensorFlow library in Python. Prior to this project, I had not worked extensively with TensorFlow, but its versatility and flexibility proved to be immensely useful. It allowed me to build and train complex models, customize loss functions, and implement adversarial debiasing techniques tailored to the specific needs of my problem. The experience gained from working with TensorFlow expanded my repertoire of tools for developing and deploying machine learning models.

Additionally, this project emphasized the importance of reading and learning from scholarly articles. While I had previously engaged with academic papers to support concepts learned in class, this project required a deeper level of engagement with the literature. I had to actively search for relevant articles, extract useful techniques and methodologies, and integrate them into my project. This skill of navigating scholarly articles, extracting knowledge, and synthesizing information from various

sources was instrumental in finding innovative solutions and advancing my understanding of the subject matter.

Overall, the hands-on experience with TensorFlow and the ability to effectively utilize scholarly articles were two key skills that I developed and honed throughout this project. These skills have not only enhanced my technical abilities but also equipped me with valuable research and problem-solving skills that will be beneficial in future data science endeavors.

8.4. Course Information

This research project was conducted as part of a 4-credit hour independent study course under the guidance of Dr. Trafalis (ttrafalis@ou.edu). The course allowed students to propose and solve a data science problem of their choice, with the faculty supervisor providing guidance and support throughout the project. The course was designed to provide students with practical hands-on experience in applying data science techniques to real-world problems and foster independent thinking and problem-solving skills.