ML Evaluation on Bank Account Fraud Dataset Suite

Shashank Vankadari[1][0000-1111-2222-3333]

[1] Lehigh University, Bethlehem PA 18015, USA
shv222@lehigh.edu

Abstract. This work makes use of a synthetically developed bank account fraud detection data suite which is developed using GAN network based on original fraud detection dataset. We make use of these datasets and perform analysis of what factors affect the fraudulent bank account opening. We then build various binary classification models and compare them using appropriate metrics to choose the best performing model. As the final step, we define fairness in the context of fraud detection and choose the suitable that can assess fairness in this use case. This is a realistic dataset used in the context of fairness research.

Keywords: Fraud detection, Machine Learning, Class imbalance, Recall, Fairness evaluation.

1 Introduction

In the era of increasing financial transactions and digitalization, safeguarding businesses against fraudulent activities presents a formidable challenge. Fraud detection finds its application in many spheres such as government, healthcare, insurance etc. Any business that deals with a high number of online transactions runs a significant risk of fraud. Banking is one such sector where fraud detection and prevention is a crucial aspect to maintain trust, as it can lead to financial losses both for banks as well as its customers. The banks must ensure that financial services are extended only to the legitimate applicants. To improve the overall customer experience, it is also important that a legitimate applicant should not be considered fraud. Using machine learning technologies can help us strike the balance between these two aspects of providing optimal customer experience for legitimate applications and denial of services for fraudsters. It is also very important that fraud detection algorithms perform well for all the customers without any bias. The algorithm should not be biased as it can result in bad consequences, disproportionately affecting certain demographic groups. Achieving fairness is not just a technical challenge but a moral obligation, especially in the financial sector where trust and reliability are crucial. By prioritizing fairness, we not only enhance the algorithm's reliability but also contribute to building a financial ecosystem that treats every individual with respect. Thus, the emphasis on fairness in fraud detection algorithms is a major step in improving transparency, trust, and equality in financial services.

This study implements Fair Machine Learning (ML) evaluation using a bank account fraud detection dataset suite to detect fraud applications. We do Exploratory

Data Analysis (EDA), model construction, the selection of appropriate metrics for evaluation, and fairness assessment to make sure that the model is performing fairly without any bias. Through comprehensive analysis and evaluation, this report aims to provide insights into the fairness and effectiveness of the implemented model. We propose suitable recommendations, based on fairness metrics which can be useful for future work involving ethical machine learning practices.

1.1 What is Fair ML?

ML fairness is a recently established area of machine learning that studies how to ensure that biases in the data and model inaccuracies do not lead to models that treat individuals unfavorably based on characteristics such as e.g. race, gender, disabilities, and sexual or political orientation. [1]

2 About the dataset

The Bank Account Fraud (BAF)[1] suite of datasets has been published at NeurIPS 2022. The Conference and Workshop on Neural Information Processing Systems (abbreviated as NeurIPS and formerly NIPS) is a machine learning and computational neuroscience conference held every December[2]. The BAF suite comprises of one base dataset with 5 additional variants which has predetermined controlled type of bias patterns, and these were generated from a real-world online bank account opening fraud detection dataset using Generative Adversarial Network (CTGAN) models. This is done to preserve the privacy of individual applicants. Each instance of dataset represents an individual, but since these are synthetically generated, they are not real people. Each variant follows the same underlying distribution as the base dataset, i.e., each instance of the suite is sampled from the same generative model.

We are using 3 variants – Base, Variant I, Variant II for our analysis. Each dataset has 1 million instances, with 32 attributes. The target variable (‘fraud_bool’) is binary which determines whether the instance is classified as fraud or not. The positive class represents a fraud instance. Following are the features of the dataset.

- Realistic, based on a present-day real-world dataset for fraud detection.
- Biased, each dataset has distinct controlled types of bias.
- Imbalanced, we have only 1% of fraud instances in the datasets.
- Dynamic, with temporal data and observed distribution shifts. We have the bank account opening data for 8 months from the month of February to September.
- Privacy preservation, to protect the identity of potential applicants they applied differential privacy techniques (noise addition), feature encoding and trained a generative model (CTGAN)[3]. It contains synthetic instances generated using a CTGAN, trained on a real bank account opening dataset, where each application is

1 https://www.kaggle.com/datasets/sgpjesus/bank-account-fraud-dataset-neurips-2022/data
2 https://en.wikipedia.org/wiki/Conference_on_Neural_Information_Processing_Systems
3 https://github.com/feedzai/bank-account-fraud/blob/main/notebooks/empirical_results.ipynb

related to an individual. In that sense, it is related to people, but the presented individuals are not real.

2.1 Data Attributes

Each variant in the data suite has the following fields:

- **income** (numeric): Annual income of the applicant (in decile form). Ranges between [0.1, 0.9].
- **name_email_similarity** (numeric): Metric of similarity between email and name. Higher values represent higher similarity. Ranges between [0, 1].
- **prev_address_months_count** (numeric): Number of months in previous registered address of the applicant, i.e. the applicant's previous residence, if applicable. Ranges between $[-1, 380]$ months (-1 is a missing value).
- **current_address_months_count** (numeric): Months in currently registered address of the applicant. Ranges between $[-1, 429]$ months (-1 is a missing value).
- **customer_age** (numeric): Applicant's age in years, rounded to the decade. Ranges between [10, 90] years.
- **days_since_request** (numeric): Number of days passed since application was done. Ranges between [0, 79] days.
- **intended_balcon_amount** (numeric): Initial transferred amount for application. Ranges between $[-16, 114]$ (negatives are missing values).
- **payment_type** (categorical): Credit payment plan type. 5 possible (anonymized) values.
- **zip_count_4w** (numeric): Number of applications within same zip code in last 4 weeks. Ranges between [1, 6830].
- **velocity_6h** (numeric): Velocity of total applications made in last 6 hours i.e., average number of applications per hour in the last 6 hours. Ranges between $[-175, 16818]$.
- **velocity_24h** (numeric): Velocity of total applications made in last 24 hours i.e., average number of applications per hour in the last 24 hours. Ranges between [1297, 9586]
- **velocity_4w** (numeric): Velocity of total applications made in last 4 weeks, i.e., average number of applications per hour in the last 4 weeks. Ranges between [2825, 7020].
- **bank_branch_count_8w** (numeric): Number of total applications in the selected bank branch in last 8 weeks. Ranges between [0, 2404].
- **date_of_birth_distinct_emails_4w** (numeric): Number of emails for applicants with same date of birth in last 4 weeks. Ranges between [0, 39].
- **employment_status** (categorical): Employment status of the applicant. 7 possible (annonymized) values.
- **credit_risk_score** (numeric): Internal score of application risk. Ranges between $[-191, 389]$.
- **email_is_free** (binary): Domain of application email (either free or paid).

- **housing_status** (categorical): Current residential status for applicant. 7 possible (annonymized) values.
- **phone_home_valid** (binary): Validity of provided home phone.
- **phone_mobile_valid** (binary): Validity of provided mobile phone.
- **bank_months_count** (numeric): How old is previous account (if held) in months. Ranges between [−1, 32] months (-1 is a missing value).
- **proposed_credit_limit (**numeric): Applicant's proposed credit limit. Ranges between [200, 2000].
- **foreign_request** (binary): If origin country of request is different from bank's country.
- **source** (categorical): Online source of application. Either browser (INTERNET) or app (TELEAPP).
- **session_length_in_minutes** (numeric): Length of user session in banking website in minutes. Ranges between [−1, 107] minutes (-1 is a missing value).
- **device_os** (categorical): Operative system of device that made request. Possible values are: Windows, macOS, Linux, X11, or other.
- **keep_alive_session** (binary): User option on session logout.
- **device_distinct_emails** (numeric): Number of distinct emails in banking website from the used device in last 8 weeks. Ranges between [−1, 2] emails (-1 is a missing value).
- **device_fraud_count** (numeric): Number of fraudulent applications with used device. Ranges between [0, 1].
- **month** (numeric): Month where the application was made. Ranges between [0, 7].
- **fraud_bool** (binary): If the application is fraudulent or not. (0 is legit, 1 is Fraud)

2.2 Types of bias

The three variants have different kinds of bias induced in it. It offers various challenges for fairness evaluation. We measure the bias in our datasets using these metrics in the fairness evaluation section.

Group size disparity. represents uneven distribution of different group-wise frequencies in a protected attribute (Age, gender etc.). In fairness evaluation, group size disparity can impact the reliability of metrics. A model may perform good for a majority group, but if the dataset has a significant imbalance in group sizes, it might not work well to underrepresented groups.

Prevalence disparity. refers to the imbalance in the distribution of positive and negative instances across different groups or categories within a dataset. In our context, we can have higher fraud datapoints for a particular group compared to another group within the protected attribute.

3 Exploratory Data Analysis

3.1 Checking Class imbalance

We have almost 98.8% of the data in negative class and only 1.2% of data in positive class. So, this is highly imbalanced data. We perform model building without doing sampling as that is not realistic in fraud detection use case. Refer the figure 1 below.

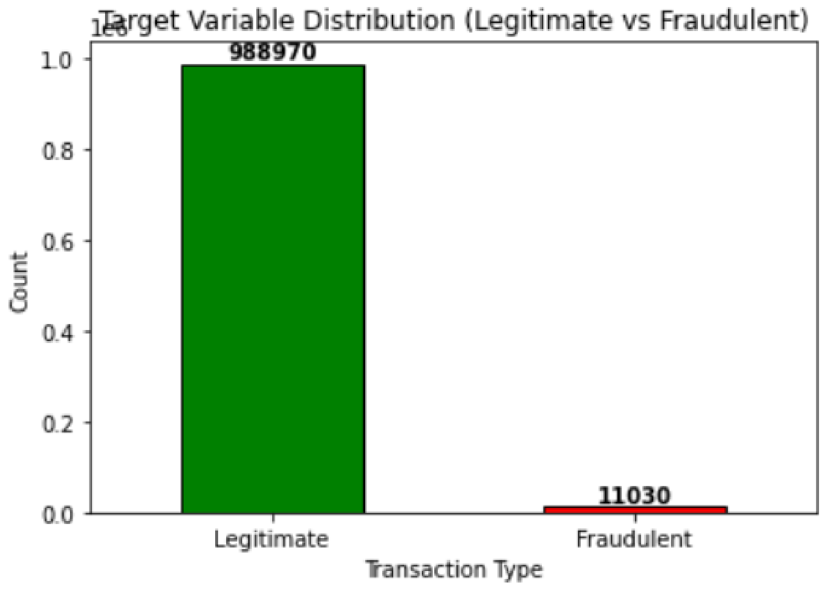


Fig. 1. Class distribution of all the data variants

3.2 Dealing with missing values

Missing values are represented by -1 in all the datasets and some variables have negative values as missing. For feature selection, the three columns prev_address_months_count, bank_months_count, intended_balcon_amount are removed as 70% of the data is missing in those attributes for all the three datasets. Also, the variable device_fraud_count has been removed as it only contains the value of zero and adds no importance to the model.

3.3 Checking skewness with respect to target variable

As in figure 2, the numerical features are positively skewed with respect to target variable. So, we perform min max scaling for all the numeric features to maintain the normality of the dataset.

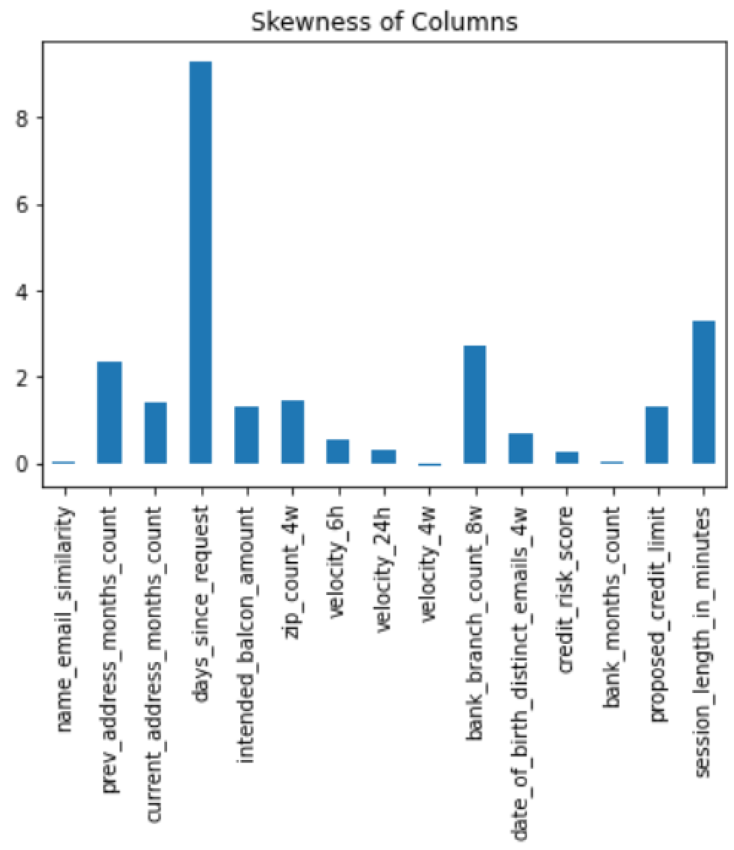


Fig. 2. Variable skewness in base dataset

3.4 Handling categorical features which have strings.

We perform one hot encoding for all the categorical features to convert it to numerical variables. Below features are one hot encoded.
['payment_type', 'employment_status', 'housing_status', 'source', 'device_os']

3.5 Checking the fraud instances by month

We make use of violin plot to check the distribution of month. Refer the figure 3 below. We can see that fraudulent data points are concentrated for months 2 and 6. So we can leverage this for data split during model training which is described in the next section.

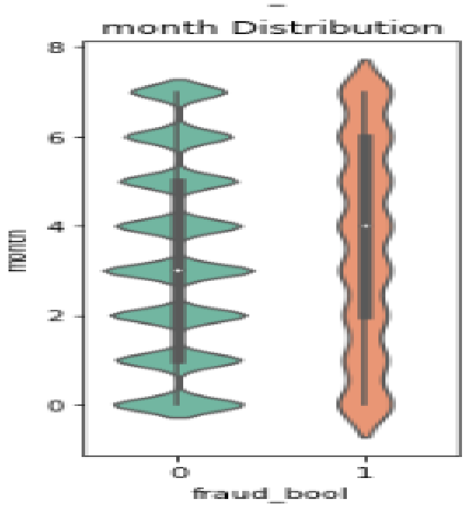


Fig. 3. Distribution of fraudulent instances by month

3.6 Protected attribute

A protected attribute, in the context of fairness and ethics in machine learning, refers to a characteristic or feature of an individual that is considered sensitive or could potentially be subject to discrimination. Protected attributes often include demographic information such as race, gender, age, or ethnicity. These attributes are labeled as "protected" because using them to make decisions, especially in predictive modeling or algorithmic decision-making, can lead to biased or unfair outcomes. The concept is rooted in the desire to prevent discrimination and promote equity by ensuring that certain characteristics are not used to disadvantage or advantage individuals in various processes, particularly those involving automated systems or artificial intelligence.

In fairness evaluation, we consider `'Age'` [4]as the protected attribute and divide the age into two groups. For this we create a separate column `'age_categorical'` where the value 1 represents the instances which has less than 50 years of age and the value 0 represents the instances where the age is greater than 50. Using this as the protected attribute, we evaluate the predictive equality in fairness evaluation. I am using the same process as described in the main paper which I am referencing. We check the Group disparity and prevalence disparity considering these two groups in the data. In table-1 we can see that, for the base dataset, 80% of the instances fall under the age group of less than 50 and only 20% of the people are from the above 50 age group. Also, for the prevalence of disparity, the second group has twice the number of fraudulent applicants compared to first group.

[4] https://www.sentrient.com.au/blog/age-is-a-personal-attribute-protected-by-law

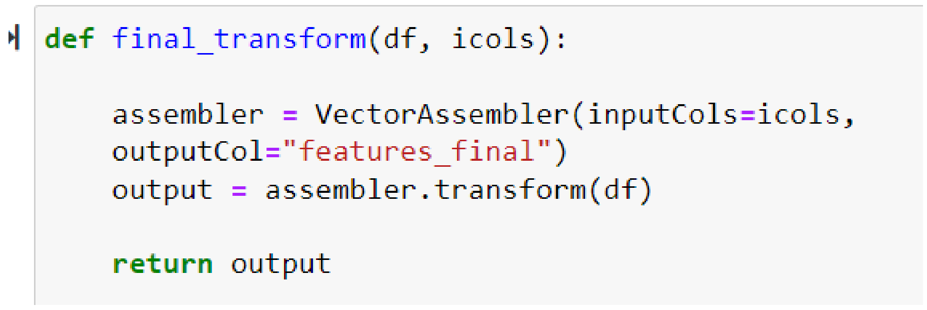
For Variant-1, the group wise disparity is very huge while prevalence disparity is not aggravated. For variant 3, the group sizes are equal, but the prevalence disparity is very aggravated.

Table 1. Disparity for the datasets for various groups for protected attribute (Age)

Dataset	Group size disparity	Prevalence disparity
Base	80-20	0.88%-1.76%
Variant I	90-10	1.08%-1.11%
Variant II	50-50	0.45%-1.92%

3.7 Feature vector

After the above tasks, we convert all the input features to a feature vector which is used for model training. We do it using a vector assembler in pyspark. Below figure shows the pseudo code for converting the input features to a feature vector. This is performed for all the three variants of datasets.



```
def final_transform(df, icols):

    assembler = VectorAssembler(inputCols=icols,
    outputCol="features_final")
    output = assembler.transform(df)

    return output
```

Fig. 4. Code for converting input features to a feature vector.

4 Model training strategy

4.1 Train-test split:

Since the dataset has the data for 8 months, we train our model on the first 6 months and test the model on the last 2 months of data. We divide this based on `'month'` column. There is a good distribution of fraudulent instances across various months, (Refer section 3) so it is a good strategy. [1]

4.2 Performance metrics

We only have 1% of fraudulent instances in the data. So, accuracy will not be a suitable evaluation metric in this use case. Since the negative labels are too high, a classifier may achieve a very high accuracy, but cannot be able to predict any fraud. As mentioned in the introduction, we want to strike a balance between identifying fraud while offering seamless services to legitimate applicants. A legitimate applicant should not be considered as fraudulent (False positive rate). So, we make use of ROC-AUC curve, Precision and Recall in evaluating our model. We put a threshold of 5% of FPR and calculate TPR (Fraudulent transactions) at that point. The aim is to achieve a high Recall (True Positive Rate) at 5% FPR as it implies that the model is effectively capturing a substantial portion of actual fraud cases while eliminating the inconvenience to legitimate customers.

4.3 Fairness metric

We select predictive equality as a suitable fairness metric for evaluating the fairness of our model. For the choice of fairness metric, we measure the FPR difference (or ratio) across the protected groups in the dataset using the thresholds that allow for 5% FPR globally. We selected this metric due to the punitive nature of the classification scenario. A false positive corresponds to a denied bank account to a legitimate applicant (with consequent societal impacts), while a false negative incurs mostly losses to the bank company.

Predictive Equality (PE) is a fairness metric that quantifies the balance in positive prediction rates across different groups. In mathematical terms, it can be expressed as the ratio of minimum FPR to the maximum FPR observed across the groups.

$$\text{PE} = (\text{Minimum FPR}) / (\text{Maximum FPR}) \tag{1}$$

5 Model Evaluation

5.1 ZeroR and OneR

We first evaluate the models using base line models such as ZeroR and OneR models so that we can a benchmark to compare these results with more complex models. Refer the Table-1 below for the model accuracies for ZeroR and OneR model on the Base dataset.

ZeroR model makes predictions based on the most frequent class in the training and OneR selects a single feature to build a predictive rule for the target variable. This rule is chosen based on the feature that leads to the most accurate predictions when used in isolation.

Table 2. Evaluation of ZeroR and OneR model

Model	Accuracy
ZeroR	98.7%
OneR	98.6%

We got very high accuracy for both the models, as it is evident with the fact that we have a greater number of datapoints in the negative class. So, we implemented the logistic regression as the baseline model and check recall at 5% FPR for all the 3 datasets.

5.2 Logistic Regression:

Since this is a binary classification problem, logistic regression can be a suitable baseline model to evaluate the data. It models the probability of an event occurring by applying the logistic function to a linear combination of input features. The logistic function transforms the output into a range between 0 and 1, representing the likelihood of the positive class. The model is trained using a logistic loss function, and its coefficients indicate the impact of each feature on the log-odds of the event, providing interpretable insights into the relationships between variables. We used maxIter=10 as the hyperparameter while training. Refer Table 2 of the results of Logistic Regression model.

Table 3. Results of Logistic Regression model.

Dataset	AUC	Recall at 5% FPR
Base	0.87	0.48
Variant I	0.85	0.43
Variant II	0.85	0.43

Below are the visualizations for ROC-AUC curves for 3 datasets.

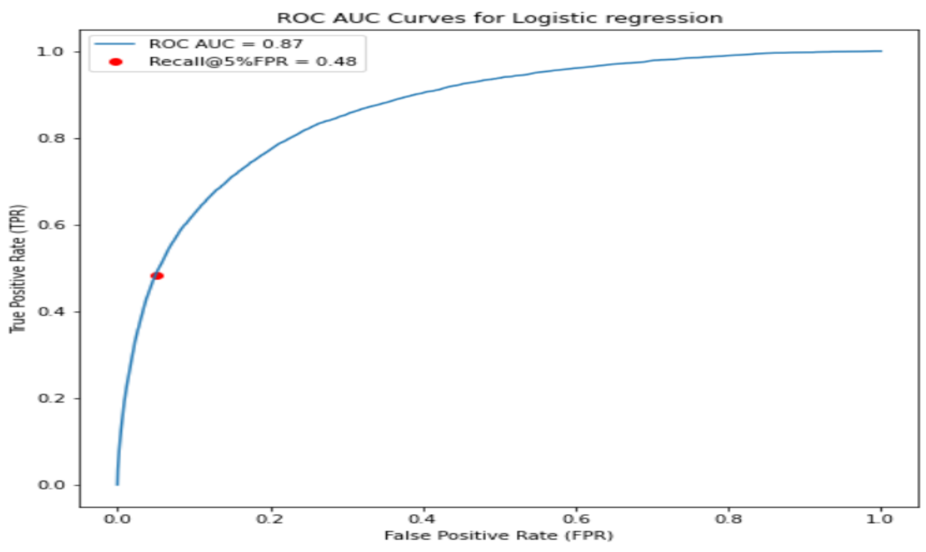


Fig. 5. ROC-AUC for Base dataset

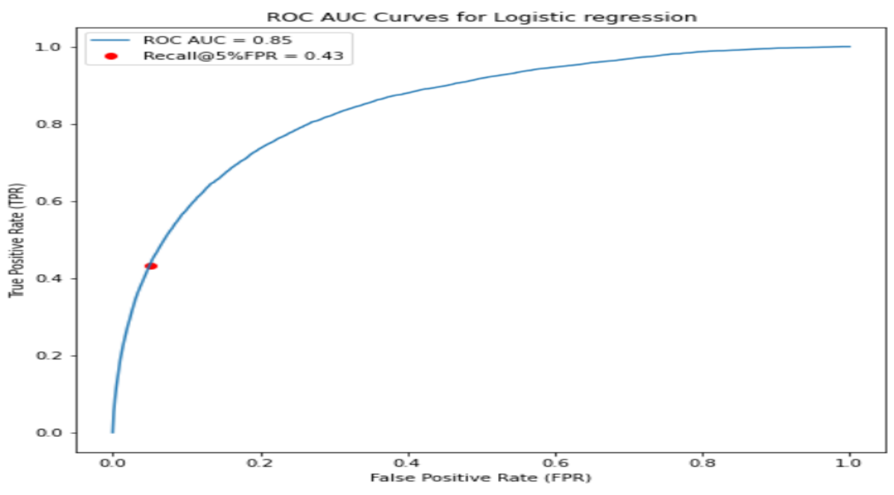


Fig. 6. ROC-AUC for Variant I dataset

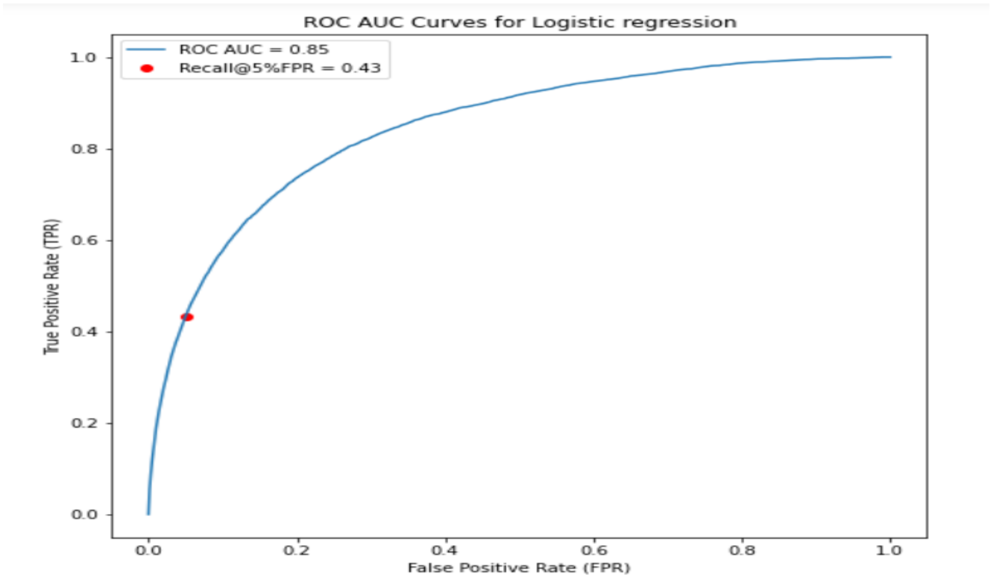


Fig. 7. ROC-AUC for Variant II dataset

The base variant has good performance but both variants have same value of metric.

5.3 Random Forest

After baseline model, we implemented Random Forest model, as this is an ensemble technique which can handle class imbalances effectively. It introduces randomness in the tree-building process by using a subset of features and data points for each tree, which helps prevent overfitting and enhances generalization. Refer Table 3 for results of random forest model.

Hyper parameter tuning. We implemented hyperparameter tuning for all the three datasets with maxDepth of [5, 10] and numTrees of [10, 20] and used the best practice of cross validation of 3 folds for training and validation. The best hyperparameters turned out to be 10 and 20 for maxDepth and numTrees respectively.

Table 4. Metrics for Random Forest model

Dataset	AUC	Recall at 5% FPR
Base	0.85	0.53
Variant I	0.84	0.47
Variant II	0.86	0.52

Below are the visualizations for Random Forest model.

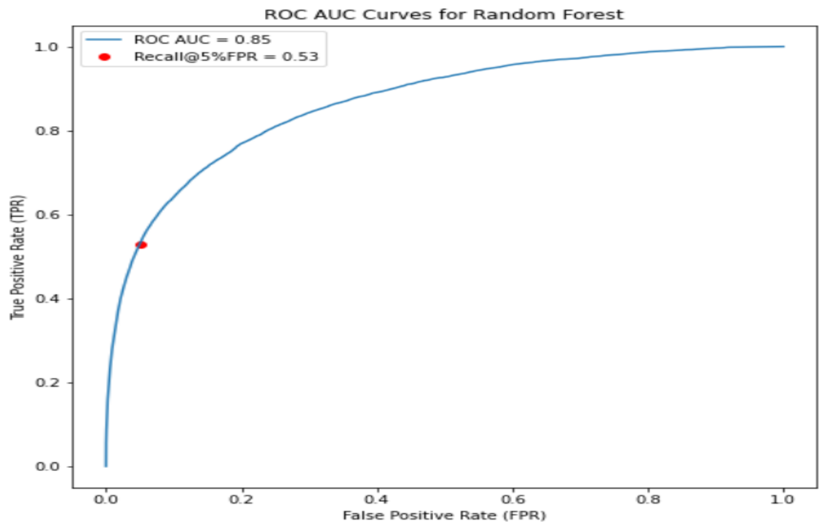


Fig. 8. ROC AUC of Base dataset for Random Forest model

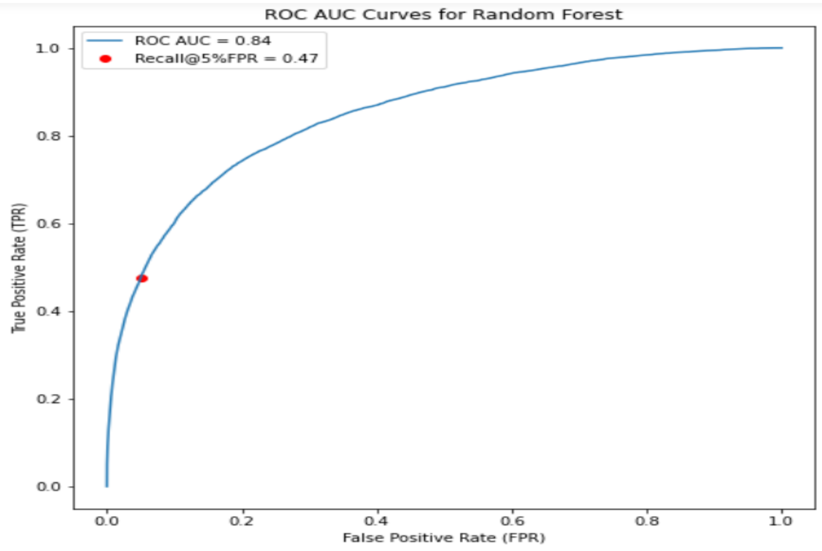


Fig. 9. ROC-AUC for Variant I for Random Forest Model

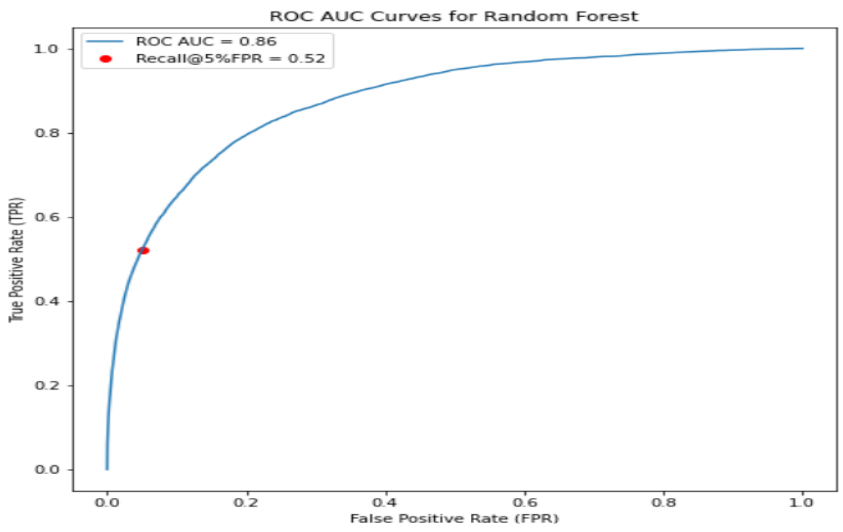


Fig. 10. ROC AUC of Variant II for Random Forest model

The AUC is almost similar for both Logistic regression and Random Forest models, but the Recall at 5% FPR is high for Random Forest model compared to Logistic Regression model. So, this can be considered the best performing model, and we use this model for fairness evaluation of these 3 datasets.

6 Model Fairness

As the Random Forest model turned out to be the best, we use this model to evaluate fairness across the protected attribute. Fairness in the context of fraud detection use case can be explained as the model's ability to ensure that it does not lead to biased outcomes or discriminate against individuals based on their age. That means the ability of a model in detecting fraud or the model wrongly classifying a legitimate applicant as fraud should be independent of the distribution of groups in the protected attribute. [2]

As discussed above, we make use of Predictive equality in predicting fairness, which means that we run the trained model separately on each group in the test set and calculate the FPRs independently. We then calculate the predictive parity as described in the previous section. Below are the results of predictive parity. The below results tell that both the disparities (group wise disparity as well as predictive parity) play an equal role in determining model's fairness. It is observed that the model has produced more fairness for Variant I compared to Base and Variant II. The base dataset and variant 2 have almost equal levels of fairness results. Fairness can never be completely achieved [4] but it can be improved by sampling the data properly during data collection process.

Table 5. Predictive parity

Dataset	Predictive parity
Base	85.66%
Variant-1	90.44%
Variant-2	87.52%

References

1. Sérgio Jesus: Turning the Tables: Biased, Imbalanced, Dynamic Tabular Datasets for ML Evaluation - https://arxiv.org/pdf/2211.13358v2.pdf
2. NINAREH MEHRABI., RED MORSTATTER,.A Survey on Bias and Fairness in Machine Learning (arxiv.org) Sections 3 and 4
3. https://medium.com/arize-ai/evaluating-model-fairness-a-primer-c85a7b0dd0d2
4. Sam Corbett-Davies., Sharad Goel., The Measure and Mismeasure of Fairness: A Critical Review of Fair Machine Learning - https://arxiv.org/abs/1808.00023v2