University
for the
Creative Arts

BERLIN SCHOOL OF
BUSINESS & INNOVATION

Assignment Title: Detecting Fraud Using Machine Learning and Data Visualization

Programme Title: Information Technology Management

Name: Safa Orhan

Number: Q1065727

Year: 2025

1. **Describe the importance of fraud detection in financial transactions and explain why synthetic datasets are used (e.g., privacy concerns, availability of real data). Highlight the benefits and limitations of using synthetic data.**

There are over 700 billion transactions facilitating per year according to Capital One Shopping research report[1]. When it is compared to the videos uploaded on Youtube (1 billion per year), It is a massive amount of data and has to be monitored, controlled in order to prevent fraudulent behavior. With the increasing volume and complexity of financial transactions, use of more advanced technologies has become a critical impact for fraud detection[2]. To implement machine learning strategies, fraud detection models should be trained well with various types of data in terms of data quality, variety and prevent underfitting scenarios. While machine learning mostly relies on data, real world data might not provide intended quality and there is always difficulties in data access, process due to privacy concerns and strict data protection regulations in the world[3]. With these challenges, synthetic data generation can provide an alternative solution for data sharing where that real world data can not facilitate. The main reason for using synthetic datasets is because of the ability to work with confidential and sensitive data. Another benefit is that synthetic data relies on real world data. While this might be a limitation for other fields, financial transactions have a variety of completed datasets so synthetic data can imitate it. Synthetic data generative models look for common trends and patterns in real world data but it can miss potential anomalies that are already presented in real data which results that it may not be completely accurate and reliable[4]. One of the most crucial challenges with using synthetic data is that potential for bias and privacy concerns datasets do not expose sensitive information or propagate biases.

2. **Select a suitable machine learning model for fraud detection (e.g., Logistic Regression, Random Forest, Gradient Boosting, Neural Networks, or Anomaly Detection techniques). Explain the rationale behind your choice, considering the characteristics of synthetic data, such as distribution and feature variability.**

In order to select one of the suitable machine learning models, synthetic data should be observed. Dataset has columns like transaction type, amount, old-new balances of destination and origin, name of origin and destination. Given these circumstances, fraud possibility is dependent on the complex relationships with various sources and different probabilities. When transaction amount increases, the possibility of fraud is not expected to increase, it depends on numerous variables to be identified as fraud behavior. This shows that there is not a linear relationship therefore logistic regression is not very suitable. In order to handle non linear relationships, decision tree algorithms should be conducted. Deep learning algorithms require very large datasets and powerful graphic processing units. Considering neither we have, it can

[1] capitaloneshopping.com. (2024). Number of Credit Card Transactions per Second & Year: 2023 Data. [online] Available at: https://capitaloneshopping.com/research/number-of-credit-card-transactions/ (Accessed: 25 December 2024).

[2] Kaur, A. and Kaur, B. (2021) 'Fraud detection in financial transactions using machine learning (abstract)', International Journal of Smart Device and Computing Science, 9(3). Available at: https://ijsdcs.com/index.php/ijsdcs/article/view/639/253 (Accessed: 25 December 2024).

[3] Wang, Y., Chen, Z. and Zhou, Z. (2021) 'A comprehensive survey on financial fraud detection (abstract)', Journal of LaTeX Class Files, 14(8), p. 1.

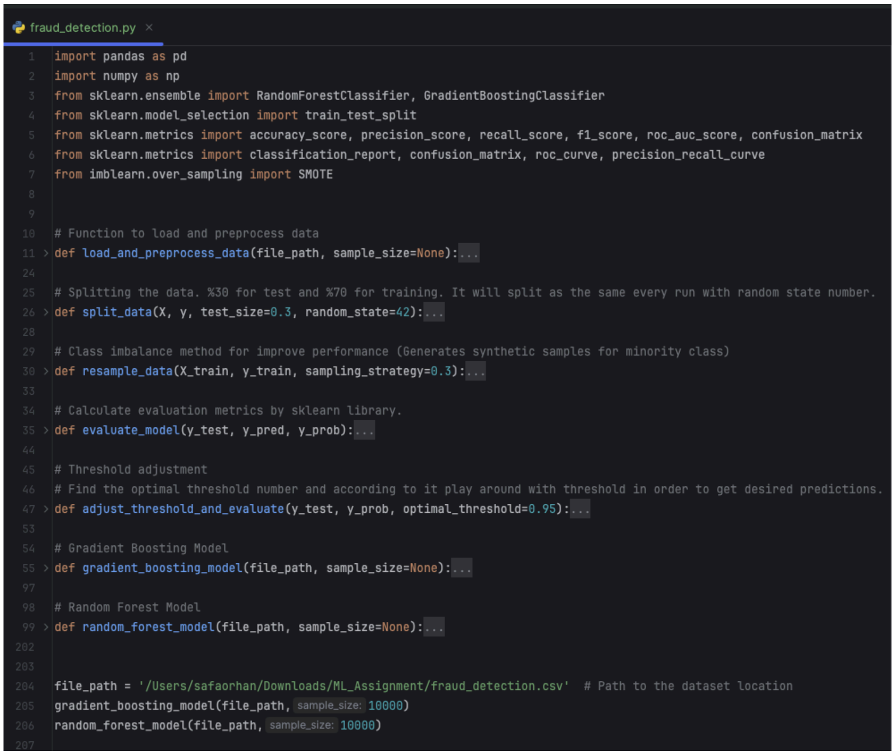
[4] Lamberti, A. (2023). The benefits and limitations of generating synthetic data. [online] syntheticus.ai. Available at: https://syntheticus.ai/blog/the-benefits-and-limitations-of-generating-synthetic-data (Accessed: 25 December 2024).

be eliminated for this assignment. This would leave a decision between random forest and gradient boosting. Both of the algorithms will be implemented and the results will be observed.

3. **Implement the chosen model using Python on a synthetic financial dataset. Provide the code and detail each step: data preprocessing (handling class imbalance, scaling features), training, and model evaluation. Include evaluation metrics like Precision, Recall, F1-Score, and AUC-ROC.**

IntelliJ IDEA Community Edition integrated development environment (IDE) with python (py) programming language is used for training the models.

In the single py file it consists of 3 main structures. As can be observed from Image 1, first section, import the necessary libraries to be able use the methods for training. Second part consists of methods to pre-process, train, and calculate the synthetic data. Third section includes file path definition and calling the training algorithm methods to run the program.

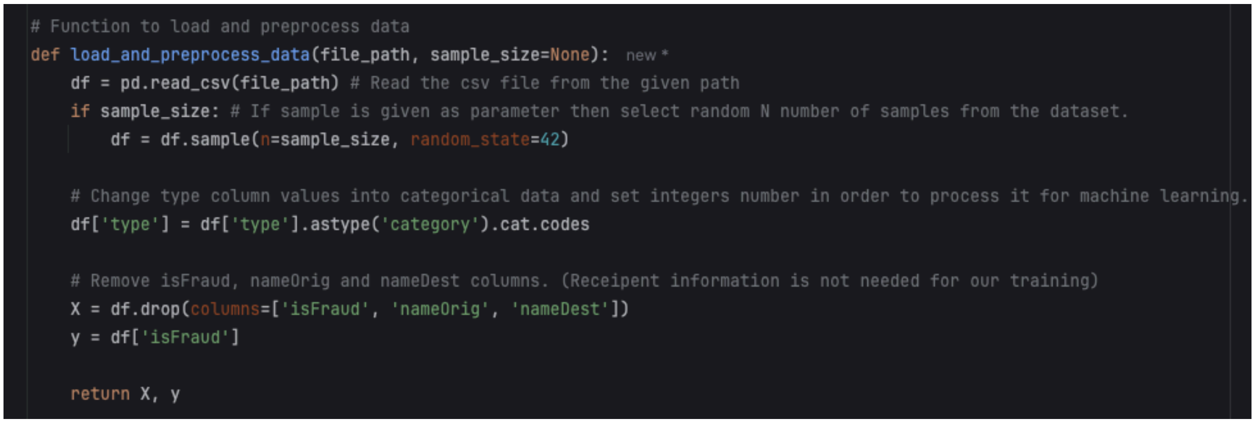


```python
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confusion_matrix
from sklearn.metrics import classification_report, confusion_matrix, roc_curve, precision_recall_curve
from imblearn.over_sampling import SMOTE


# Function to load and preprocess data
def load_and_preprocess_data(file_path, sample_size=None):...

# Splitting the data. %30 for test and %70 for training. It will split as the same every run with random state number.
def split_data(X, y, test_size=0.3, random_state=42):...

# Class imbalance method for improve performance (Generates synthetic samples for minority class)
def resample_data(X_train, y_train, sampling_strategy=0.3):...

# Calculate evaluation metrics by sklearn library.
def evaluate_model(y_test, y_pred, y_prob):...

# Threshold adjustment
# Find the optimal threshold number and according to it play around with threshold in order to get desired predictions.
def adjust_threshold_and_evaluate(y_test, y_prob, optimal_threshold=0.95):...

# Gradient Boosting Model
def gradient_boosting_model(file_path, sample_size=None):...

# Random Forest Model
def random_forest_model(file_path, sample_size=None):...


file_path = '/Users/safaorhan/Downloads/ML_Assignment/fraud_detection.csv'  # Path to the dataset location
gradient_boosting_model(file_path, sample_size: 10000)
random_forest_model(file_path, sample_size: 10000)
```

Image 1: Overall layout of the python program.

To have a better readability and to avoid repetitive tasks, 7 methods are created.

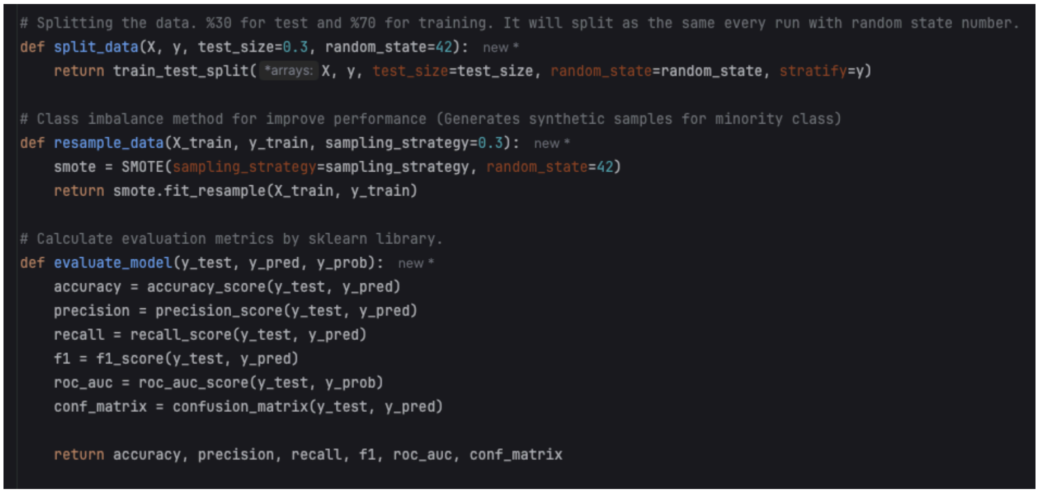
Method 1: Load and preprocess data
First action is to load the data and process it. In the comma separated values (csv) data, type column has string values and to be able to train the model, type values are changed to categorical data with random integers numbers (Image 2). “nameOrig” and “nameDest” columns are removed from the csv file as recipient information is not needed for model training which impacts 0 for predictions of fraud activities. “isFraud” column and its values are moved to y variable as it is the target variable.



```
# Function to load and preprocess data
def load_and_preprocess_data(file_path, sample_size=None):  new *
    df = pd.read_csv(file_path) # Read the csv file from the given path
    if sample_size: # If sample is given as parameter then select random N number of samples from the dataset.
        df = df.sample(n=sample_size, random_state=42)

    # Change type column values into categorical data and set integers number in order to process it for machine learning.
    df['type'] = df['type'].astype('category').cat.codes

    # Remove isFraud, nameOrig and nameDest columns. (Receipent information is not needed for our training)
    X = df.drop(columns=['isFraud', 'nameOrig', 'nameDest'])
    y = df['isFraud']

    return X, y
```

Image 2: Load and preprocess data

Method 2-3-4: Split, resample and evaluate
Split data method splits the processed data, %30 for test, %70 for training.
Resample data method generates new synthetic data for minority classes and updates training data to have a balance between class distributions.
Evaluate model method calculates evaluation metrics with the results of y predictions and probabilities.

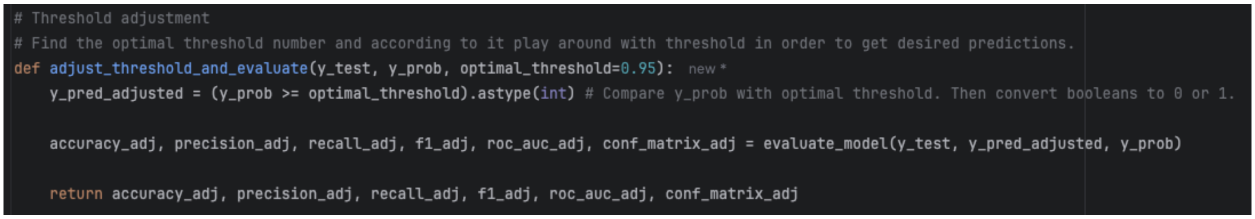


```
# Splitting the data. %30 for test and %70 for training. It will split as the same every run with random state number.
def split_data(X, y, test_size=0.3, random_state=42):  new *
    return train_test_split( *arrays: X, y, test_size=test_size, random_state=random_state, stratify=y)

# Class imbalance method for improve performance (Generates synthetic samples for minority class)
def resample_data(X_train, y_train, sampling_strategy=0.3):  new *
    smote = SMOTE(sampling_strategy=sampling_strategy, random_state=42)
    return smote.fit_resample(X_train, y_train)

# Calculate evaluation metrics by sklearn library.
def evaluate_model(y_test, y_pred, y_prob):  new *
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    roc_auc = roc_auc_score(y_test, y_prob)
    conf_matrix = confusion_matrix(y_test, y_pred)

    return accuracy, precision, recall, f1, roc_auc, conf_matrix
```

Image 3: Process data for training

Method 5: Adjust threshold and re-evaluate

In this method, it adjusts the threshold value to classify predictions. In this method, the threshold value is set to 0.95 which means only predictions with a probability higher than 0.95 will be classified as fraud. This makes the model stricter and cautious. After adjusting the threshold, the performance of the model is evaluated again and metrics are calculated to see how the model results.

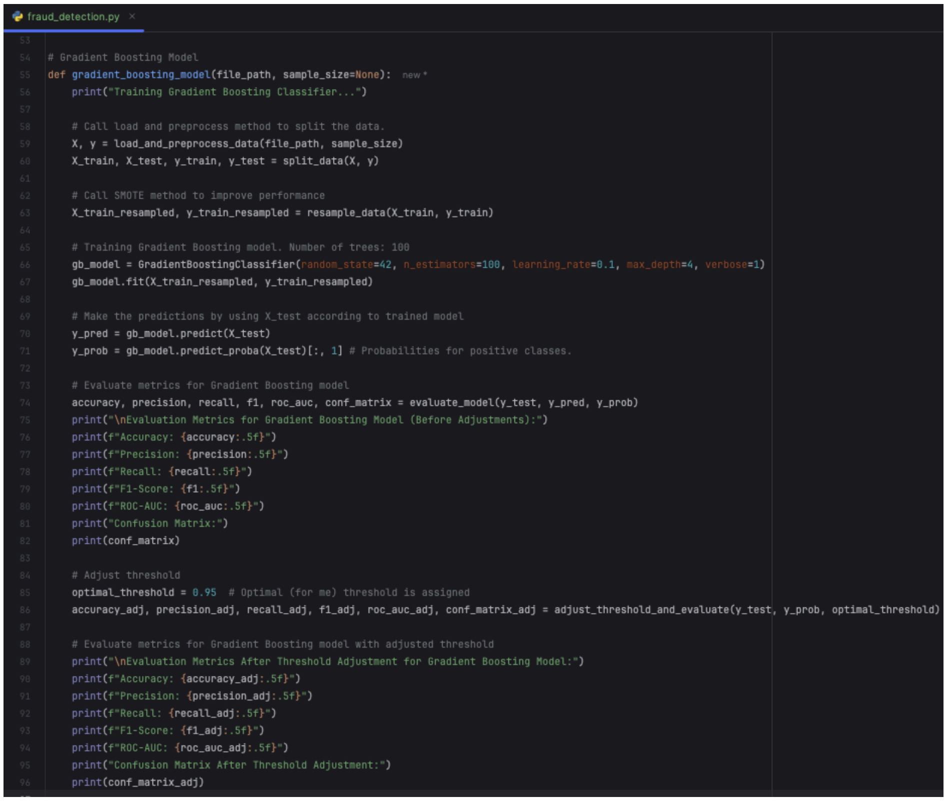


```python
# Threshold adjustment
# Find the optimal threshold number and according to it play around with threshold in order to get desired predictions.
def adjust_threshold_and_evaluate(y_test, y_prob, optimal_threshold=0.95):
    y_pred_adjusted = (y_prob >= optimal_threshold).astype(int) # Compare y_prob with optimal threshold. Then convert booleans to 0 or 1.

    accuracy_adj, precision_adj, recall_adj, f1_adj, roc_auc_adj, conf_matrix_adj = evaluate_model(y_test, y_pred_adjusted, y_prob)

    return accuracy_adj, precision_adj, recall_adj, f1_adj, roc_auc_adj, conf_matrix_adj
```

Image 4: Threshold calculation

Method 6: Gradient boosting algorithm

In this method training and predictions are completed with gradient boosting algorithm. Load and preprocess data, split data, resample data methods are called in order. Model is trained with gradient boosting method with 100 trees and 4 depths. When more than 100 trees and 4 depths are selected it was consuming a lot of time to train the model. After evaluation metrics are printed, threshold method is called and evaluation metrics with threshold value is printed again.

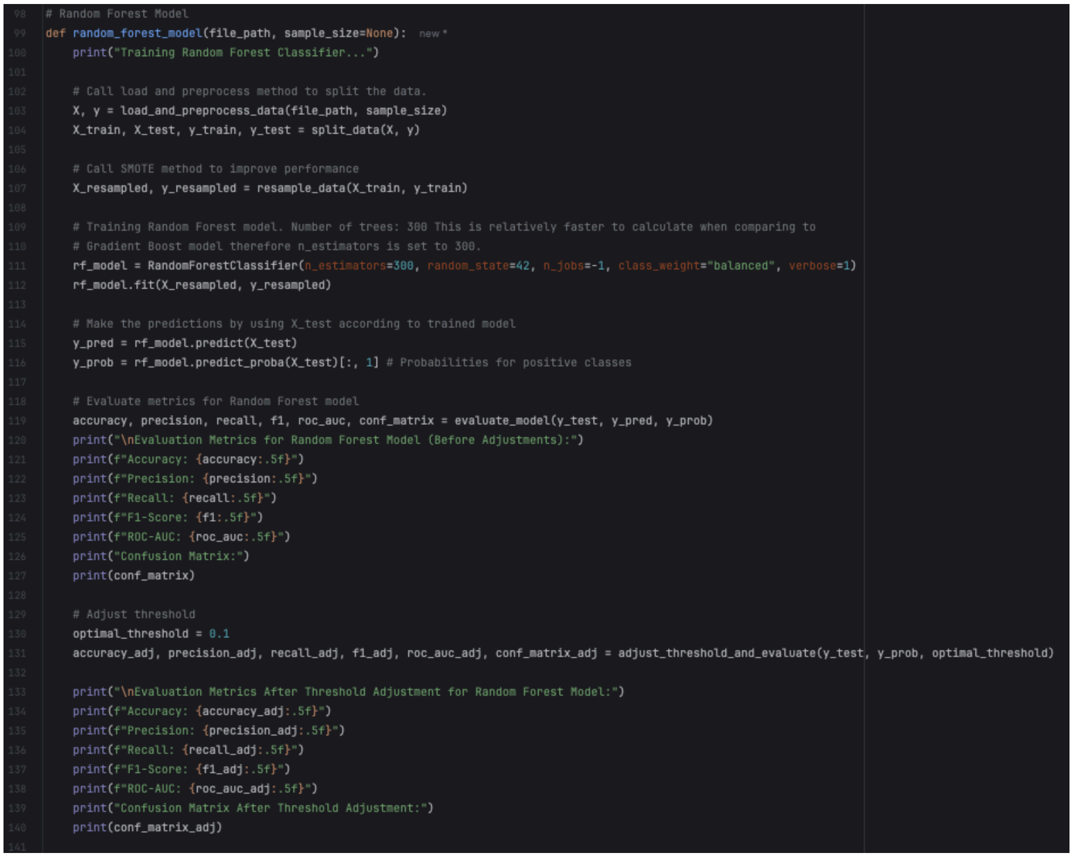


```python
# Gradient Boosting Model
def gradient_boosting_model(file_path, sample_size=None):
    print("Training Gradient Boosting Classifier...")

    # Call load and preprocess method to split the data.
    X, y = load_and_preprocess_data(file_path, sample_size)
    X_train, X_test, y_train, y_test = split_data(X, y)

    # Call SMOTE method to improve performance
    X_train_resampled, y_train_resampled = resample_data(X_train, y_train)

    # Training Gradient Boosting model. Number of trees: 100
    gb_model = GradientBoostingClassifier(random_state=42, n_estimators=100, learning_rate=0.1, max_depth=4, verbose=1)
    gb_model.fit(X_train_resampled, y_train_resampled)

    # Make the predictions by using X_test according to trained model
    y_pred = gb_model.predict(X_test)
    y_prob = gb_model.predict_proba(X_test)[:, 1] # Probabilities for positive classes.

    # Evaluate metrics for Gradient Boosting model
    accuracy, precision, recall, f1, roc_auc, conf_matrix = evaluate_model(y_test, y_pred, y_prob)
    print("\nEvaluation Metrics for Gradient Boosting Model (Before Adjustments):")
    print(f"Accuracy: {accuracy:.5f}")
    print(f"Precision: {precision:.5f}")
    print(f"Recall: {recall:.5f}")
    print(f"F1-Score: {f1:.5f}")
    print(f"ROC-AUC: {roc_auc:.5f}")
    print("Confusion Matrix:")
    print(conf_matrix)

    # Adjust threshold
    optimal_threshold = 0.95  # Optimal (for me) threshold is assigned
    accuracy_adj, precision_adj, recall_adj, f1_adj, roc_auc_adj, conf_matrix_adj = adjust_threshold_and_evaluate(y_test, y_prob, optimal_threshold)

    # Evaluate metrics for Gradient Boosting model with adjusted threshold
    print("\nEvaluation Metrics After Threshold Adjustment for Gradient Boosting Model:")
    print(f"Accuracy: {accuracy_adj:.5f}")
    print(f"Precision: {precision_adj:.5f}")
    print(f"Recall: {recall_adj:.5f}")
    print(f"F1-Score: {f1_adj:.5f}")
    print(f"ROC-AUC: {roc_auc_adj:.5f}")
    print("Confusion Matrix After Threshold Adjustment:")
    print(conf_matrix_adj)
```

Image 5: Gradient boosting training and metrics

Method 7: Random forest algorithm

Same as method 6 steps are written for random forest algorithm. Only difference is that 300 trees are defined for training random forest algorithm because it is relatively much faster to train compared to gradient boosting algorithm.



```python
# Random Forest Model
def random_forest_model(file_path, sample_size=None):
    print("Training Random Forest Classifier...")

    # Call load and preprocess method to split the data.
    X, y = load_and_preprocess_data(file_path, sample_size)
    X_train, X_test, y_train, y_test = split_data(X, y)

    # Call SMOTE method to improve performance
    X_resampled, y_resampled = resample_data(X_train, y_train)

    # Training Random Forest model. Number of trees: 300 This is relatively faster to calculate when comparing to
    # Gradient Boost model therefore n_estimators is set to 300.
    rf_model = RandomForestClassifier(n_estimators=300, random_state=42, n_jobs=-1, class_weight="balanced", verbose=1)
    rf_model.fit(X_resampled, y_resampled)

    # Make the predictions by using X_test according to trained model
    y_pred = rf_model.predict(X_test)
    y_prob = rf_model.predict_proba(X_test)[:, 1] # Probabilities for positive classes

    # Evaluate metrics for Random Forest model
    accuracy, precision, recall, f1, roc_auc, conf_matrix = evaluate_model(y_test, y_pred, y_prob)
    print("\nEvaluation Metrics for Random Forest Model (Before Adjustments):")
    print(f"Accuracy: {accuracy:.5f}")
    print(f"Precision: {precision:.5f}")
    print(f"Recall: {recall:.5f}")
    print(f"F1-Score: {f1:.5f}")
    print(f"ROC-AUC: {roc_auc:.5f}")
    print("Confusion Matrix:")
    print(conf_matrix)

    # Adjust threshold
    optimal_threshold = 0.1
    accuracy_adj, precision_adj, recall_adj, f1_adj, roc_auc_adj, conf_matrix_adj = adjust_threshold_and_evaluate(y_test, y_prob, optimal_threshold)

    print("\nEvaluation Metrics After Threshold Adjustment for Random Forest Model:")
    print(f"Accuracy: {accuracy_adj:.5f}")
    print(f"Precision: {precision_adj:.5f}")
    print(f"Recall: {recall_adj:.5f}")
    print(f"F1-Score: {f1_adj:.5f}")
    print(f"ROC-AUC: {roc_auc_adj:.5f}")
    print("Confusion Matrix After Threshold Adjustment:")
    print(conf_matrix_adj)
```

Image 6: Random forest training and metrics

Evaluation Metrics for Algorithms

In order to create visualizations on Tableau tool, with every threshold value, evaluation metrics are calculated and saved in a chart.

Model	Threshold	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Gradient Boosting	0	0.00127	0.00127	1	0.00253	0.99961
Gradient Boosting	0.1	0.9824	0.06713	1	0.12581	0.99961
Gradient Boosting	0.2	0.982	0.0656	0.99737	0.12311	0.99927
Gradient Boosting	0.3	0.98794	0.09446	0.99211	0.1725	0.99927
Gradient Boosting	0.4	0.99128	0.12433	0.97368	0.2205	0.99927
Gradient Boosting	0.5	0.99371	0.16286	0.95789	0.27839	0.99927
Gradient Boosting	0.6	0.99574	0.22181	0.94211	0.35908	0.99927
Gradient Boosting	0.7	0.9973	0.30899	0.91316	0.46174	0.99927
Gradient Boosting	0.8	0.99861	0.47443	0.87895	0.61624	0.99927
Gradient Boosting	0.9	0.99928	0.67742	0.82895	0.74556	0.99927
Gradient Boosting	0.95	0.99961	0.92283	0.75526	0.83068	0.99927
Gradient Boosting	1.0	0.99873	0	0	0	0.99927

Table 1: Evaluation metrics chart for gradient boosting algorithm

Model	Actual	Predicted	Count
Gradient Boosting	Legitimate	Legitimate	299.587
Gradient Boosting	Legitimate	Fraud	33
Gradient Boosting	Fraud	Legitimate	65
Gradient Boosting	Fraud	Fraud	315

Table 2: Confusion matrix with threshold 0.95 for gradient boosting algorithm

Model	Threshold	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Random Forest	0	0.00127	0.00127	1	0.00253	0.99958
Random Forest	0.1	0.9958	0.2276	0.9737	0.3689	0.9996
Random Forest	0.2	0.9976	0.3396	0.95	0.50035	0.99958
Random Forest	0.3	0.99842	0.44169	0.93684	0.60034	0.99958
Random Forest	0.4	0.99886	0.52968	0.91579	0.67117	0.99958
Random Forest	0.5	0.9991	0.6131	0.8632	0.7169	0.9996
Random Forest	0.6	0.9993	0.6834	0.8237	0.7470	0.9996
Random Forest	0.7	0.9995	0.7911	0.7974	0.7942	0.9996
Random Forest	0.8	0.9996	0.9003	0.7605	0.8245	0.9996
Random Forest	0.9	0.9996	0.9511	0.7158	0.8168	0.9996
Random Forest	1.0	0.9993	1.0000	0.4711	0.6404	0.9996

Table 3: Evaluation metrics chart for random forest algorithm

Model	Actual	Predicted	Count
Random Forest	Legitimate	Legitimate	299.540
Random Forest	Legitimate	Fraud	77
Random Forest	Fraud	Legitimate	80
Random Forest	Fraud	Fraud	303

Table 4: Confusion matrix with threshold 0.8 for random forest algorithm

4. Displays transaction volumes, fraud rates, and key trends in the synthetic dataset.

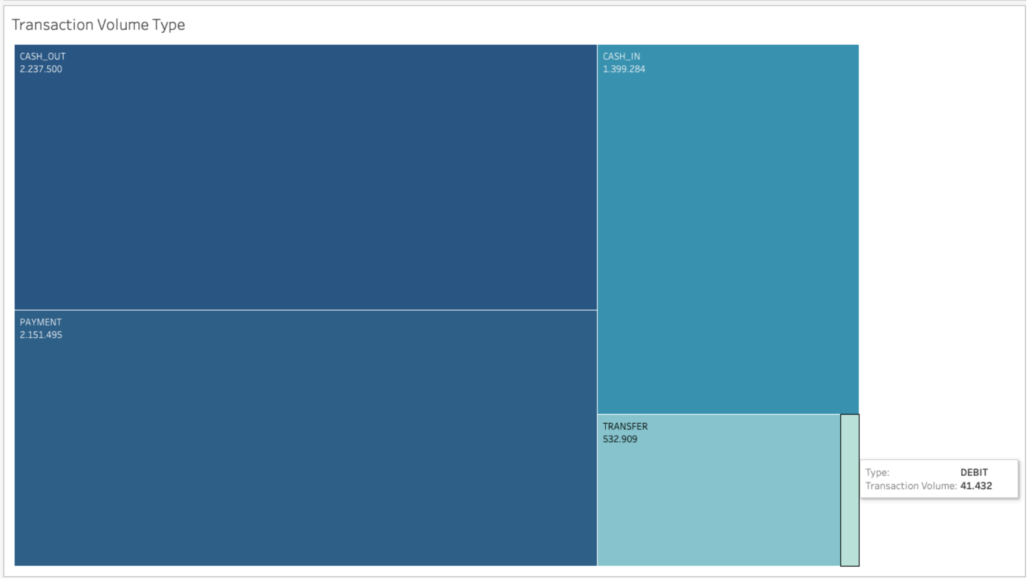


Image 7: Transaction Volumes per Type

In this tree map chart, more than 60% of the transactions belong to cash out and payment types and the least number of transaction volume is debit type.

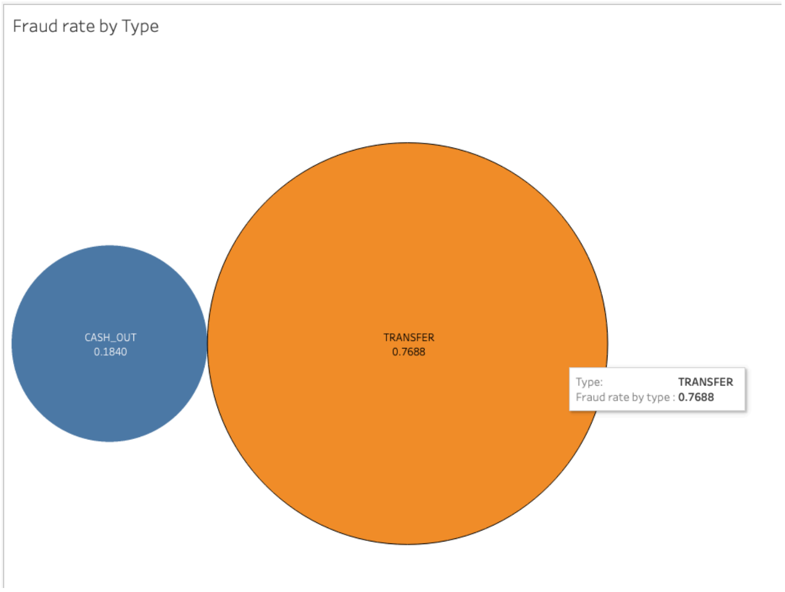
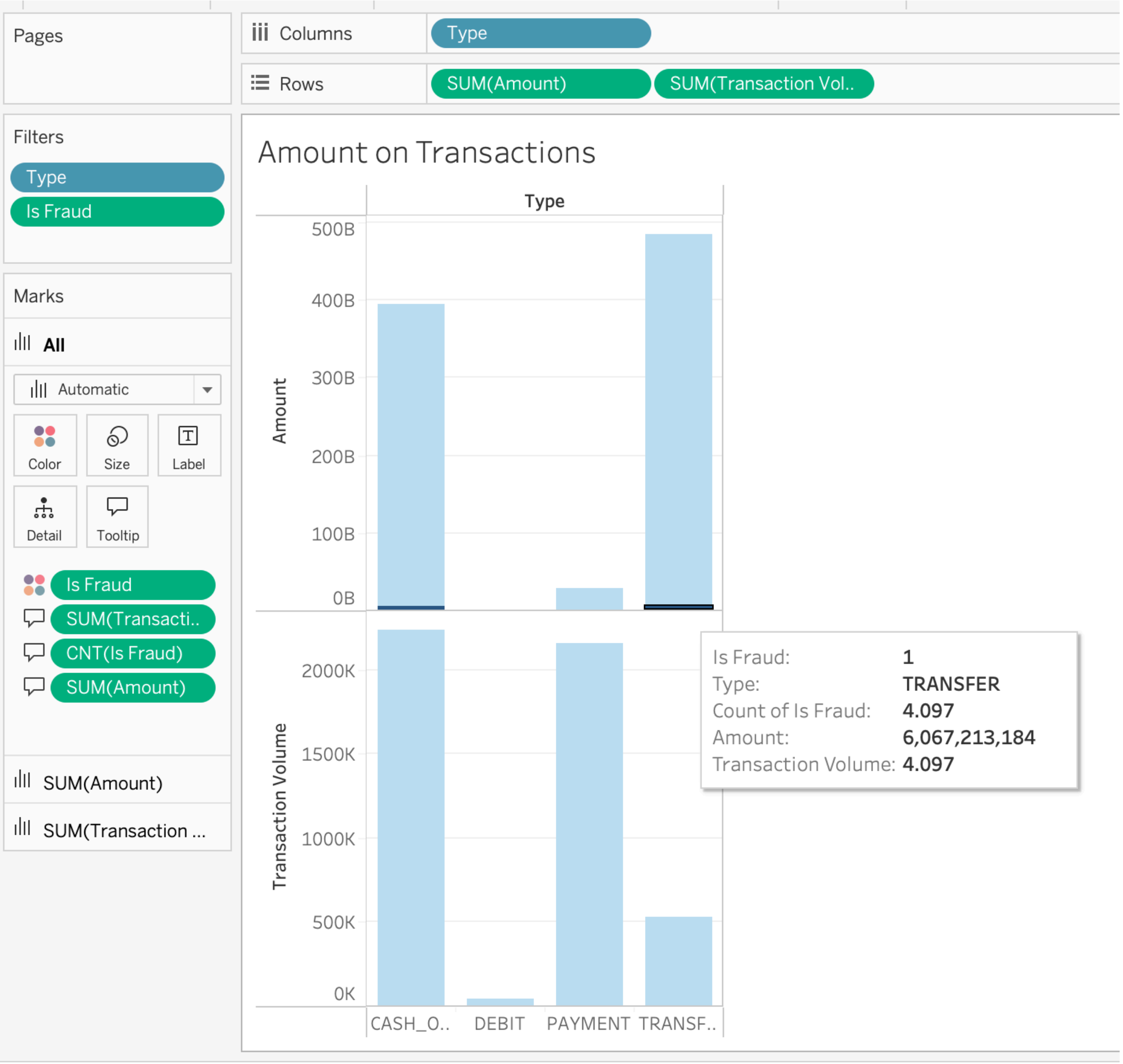


Image 8: Fraud rate by Type

To be able to see the fraud rates, packed bubble chart is chosen. While transfer type fraud rate is 0.7, cash out type fraud rate is close to 0.2 and the other transaction types has 0 fraud rate which shows that especially transfer transaction type should be carefully observed and make sure to alert if any fraudal activity is sensed.

5. **Highlights patterns of fraud with filters for transaction type, amount, and account features.**



Columns: Type

Rows: SUM(Amount) SUM(Transaction Vol..

Filters: Type, Is Fraud

Marks: Is Fraud, SUM(Transacti.., CNT(Is Fraud), SUM(Amount)

SUM(Amount)

SUM(Transaction ...

Is Fraud: 1
Type: TRANSFER
Count of Is Fraud: 4.097
Amount: 6,067,213,184
Transaction Volume: 4.097

Image 9: Balance Change on Transaction per Type

In this bar chart the amount of transactions and transaction volumes are compared in terms of transaction type. While the total amount of fraud transfer activity is 6.067.213; total amount of fraud cash out type activity is 5.989.202 which shows that there is a very high number of transactions occurring for cash out type when we compare it with transfer type which can be observed from *Image 5: Transaction Volumes per Type.* Even though transfer transaction volume is very low, the total amount of fraudal events is the highest which shows again that securing these operations is crucial and high valued accounts might being targeted.

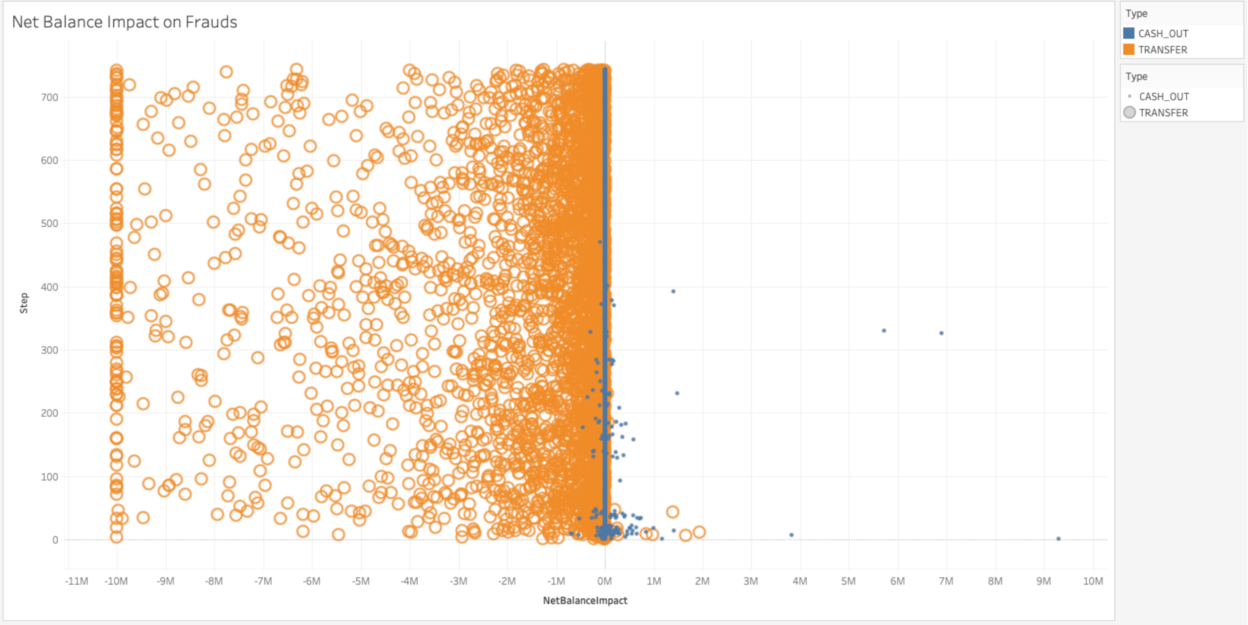


Image 10: Net Balance Impact per Type

In this scatter chart, most of the transfer-typed fraudulent activities are on the negative part of the net balance impact axis which is showing a pattern to link between fraud events and negative balanced transfers. Blue dots show cash out typed fraudulent activities with no specific pattern.

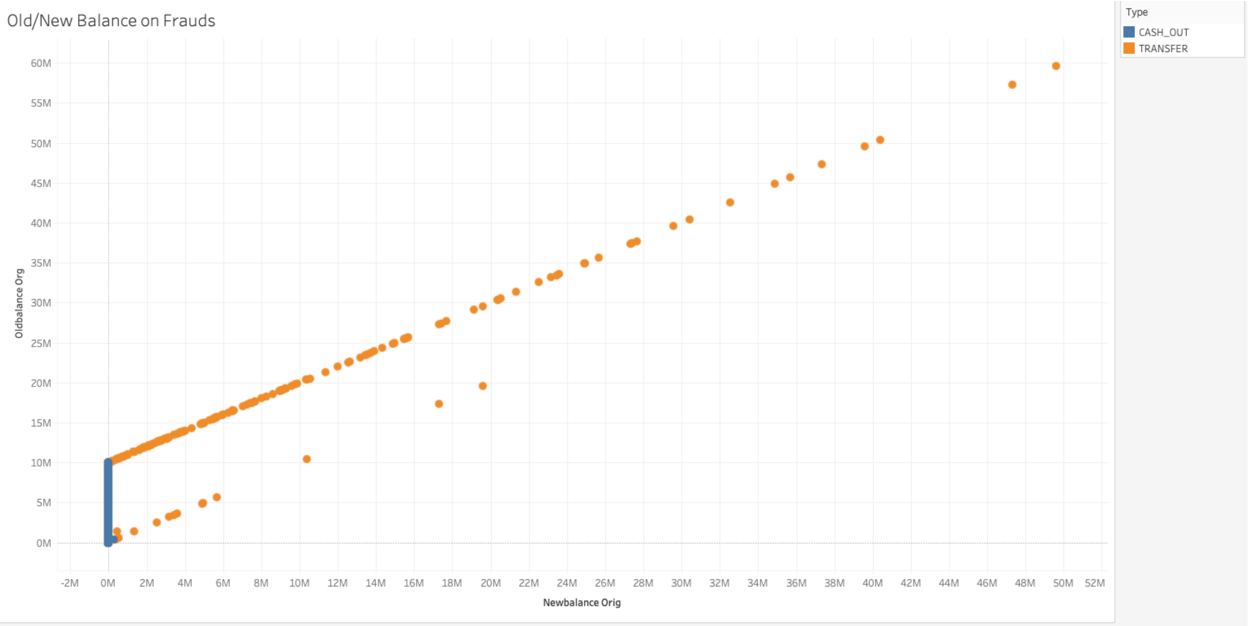
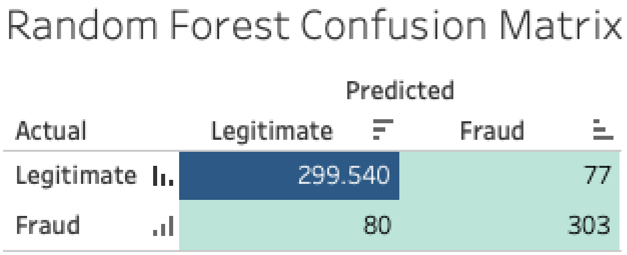


Image 11: Old/New Balance on Frauds

In this scatter chart, transfer-typed fraudulent events are drawing an almost linear line which shows there is a meaningful correlation between the accounts of old and new balances. Blue dots show that almost all of the cash out typed fraudulent events are resulting with zero balanced new accounts.

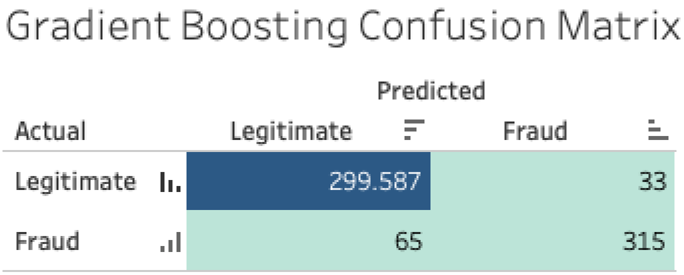
6. **Visualizes model metrics like confusion matrix and Precision-Recall curves to assess fraud detection effectiveness.**



Random Forest Confusion Matrix

	Predicted	
Actual	Legitimate	Fraud
Legitimate	299.540	77
Fraud	80	303

Image 12: Random Forest Confusion Matrix



Gradient Boosting Confusion Matrix

	Predicted	
Actual	Legitimate	Fraud
Legitimate	299.587	33
Fraud	65	315

Image 13: Gradient Boosting Confusion Matrix

For fraudulent activities the most important topic is to catch fraudulent activity which is false negative cases. While random forest has 80 missed cases, gradient boosting has 65 which is showing it is an average trained model because false negatives should be close to 0. False positive cases are important as well but if the operation is able to handle these cases, high volumes are tolerable. Overall this confusion matrix shows that gradient boosting is slightly a better trained model compared to random forest.

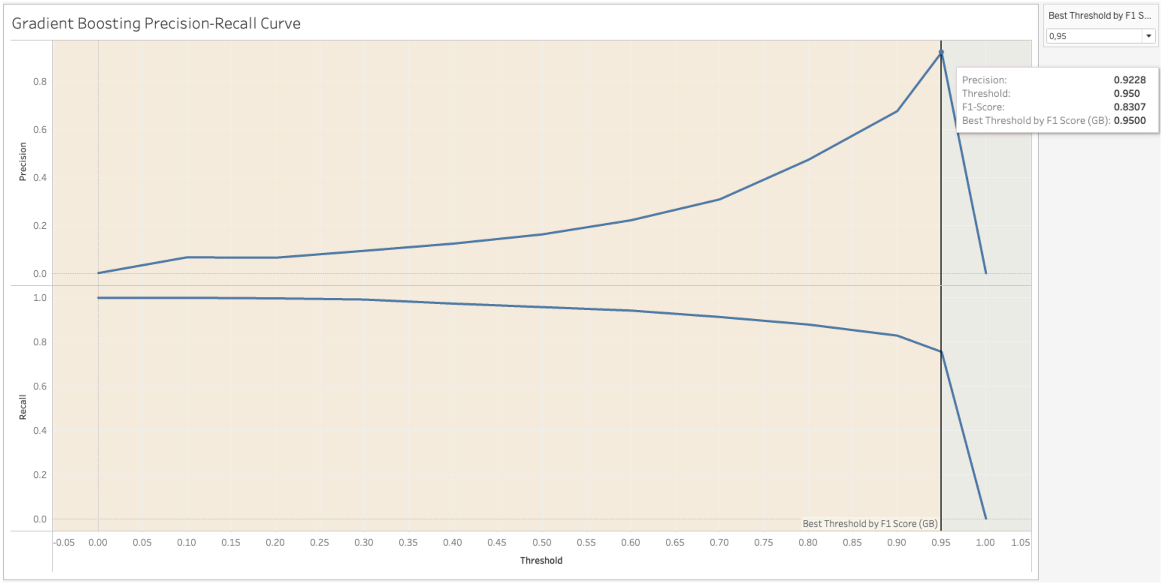


Image 14: Gradient Boosting Precision-Recall Curve

For gradient boosting algorithm, on every threshold value precision and recall metrics are calculated and results at 0.95 value gives the best F1 score high precision and recall. It is observed that when threshold value increases, it is sacrificed from recall to increase precision. Even though recall (catching false negative cases) is important, there should be a balance between precision and recall therefore the threshold value which gives the best F1 score is used to train the model. When the threshold value is getting closer to 1, the model is not able to catch the false negative cases anymore and it results a significant drop in recall value.

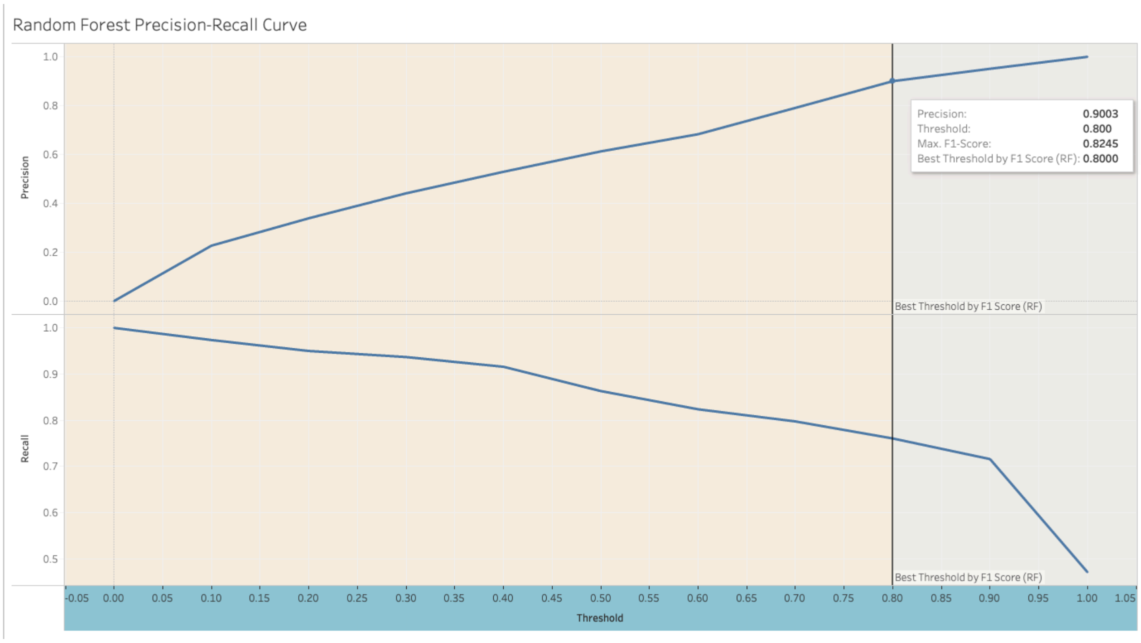


Image 15: Random Forest Precision-Recall Curve

For random forest algorithm, the threshold value of 0.8 gives the best F1 score. Same as the gradient boosting curve, while it is closer to 1 recall value significantly decreases because of false predicted fraudulent cases.

7. **Use Tableau dashboards to narrate fraud detection, covering data exploration, model results, and insights.**

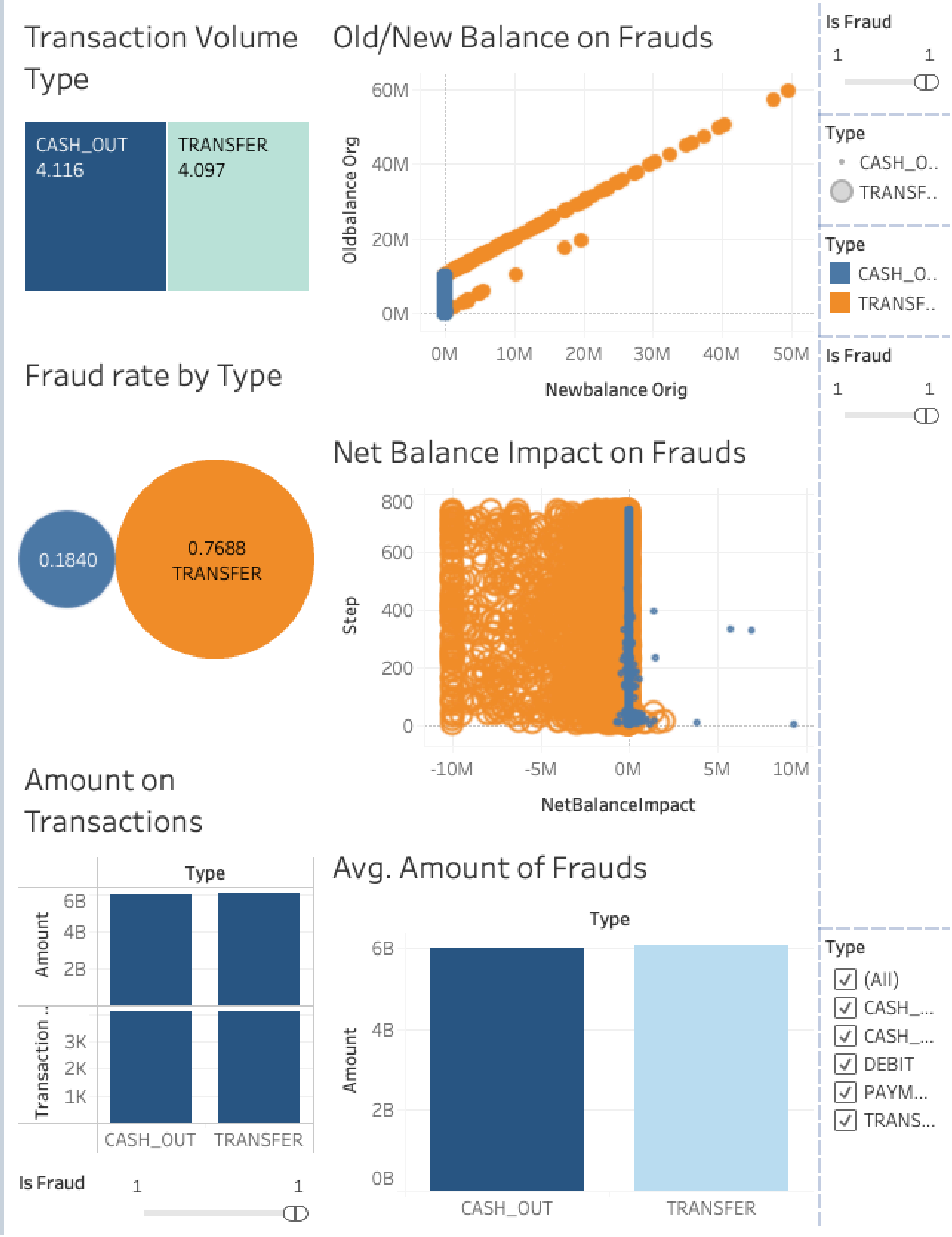
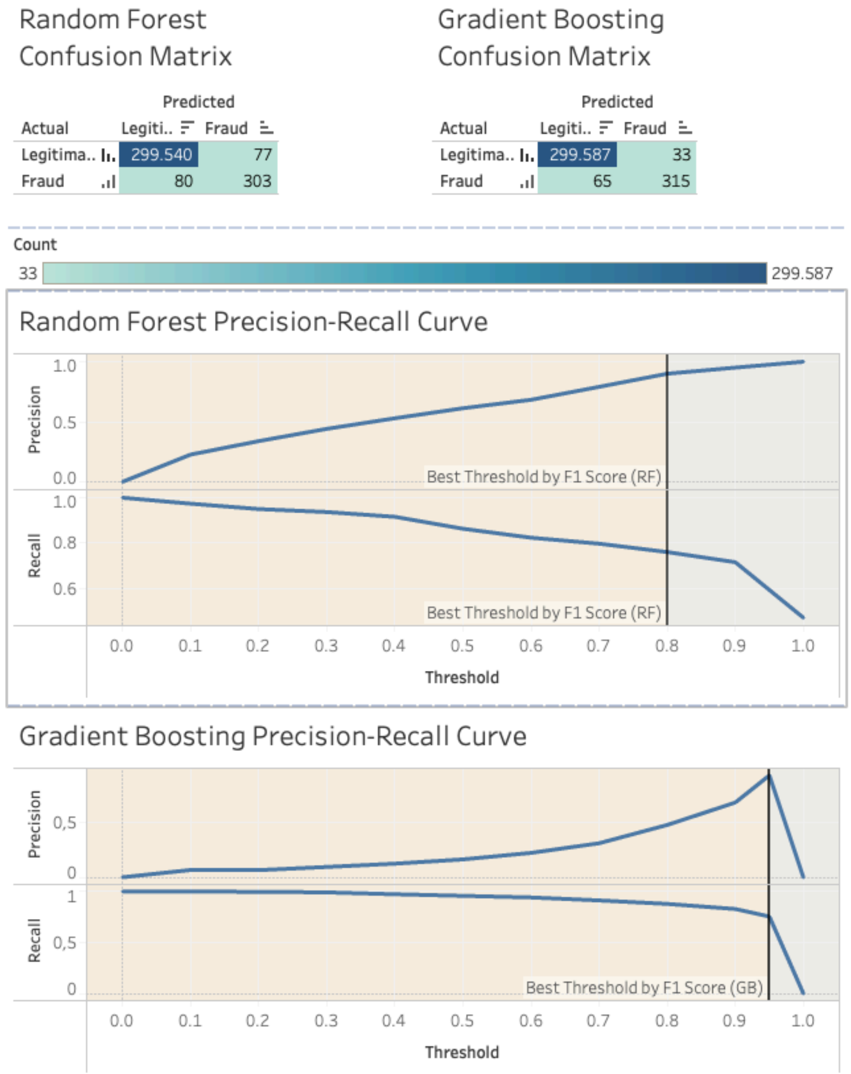


Image 16: Synthetic dataset dashboard

Fraud detection is the most critical attribute for financial companies in order to assure customer trust and financial loss. This analysis aims to identify fraudulent activities by using machine learning techniques with feeding a synthetic dataset. Model results data and synthetic data is exported and used for visualizations on Tableau. Dashboards are created to be able to get insights from the dataset.

From the dashboard when only fraudulent events are filtered, cash out and transfer type fraud events almost equal around 4100 but fraud rates differ from each other. Fraudulent events' amount when cash out and transfer are compared, transfer amount is slightly higher than cash out. From the dashboard we understand that transfer type is the most invulnerable for fraud and it needs to be taken extra attention for activities. In order to get more insight fraud events of transfer type should be checked.



Random Forest Confusion Matrix

	Predicted	
Actual	Legiti..	Fraud
Legitima..	299.540	77
Fraud	80	303

Gradient Boosting Confusion Matrix

	Predicted	
Actual	Legiti..	Fraud
Legitima..	299.587	33
Fraud	65	315

Random Forest Precision-Recall Curve

Gradient Boosting Precision-Recall Curve

Image 17: Model results dashboard

Random forest and gradient boosting algorithms results are compared by Tableau dashboard. With different threshold values precision-recall curves are compared. From the confusion matrix, gradient boosting trained better than random forest by recall and precision numbers.

8. Propose actionable steps to prevent fraud, highlighting real-world applications for financial institutions.

In order to prevent and handle fraud, advanced machine learning models should be integrated and real time monitoring should be used to detect transactions. When fraudulent activities can be detected, advanced security measures should be integrated like 2 factor authentication if the transaction is possibly marked as fraud or very high transfers. The real time monitoring should be forwarded to visualization tool dashboards and be observed in order to be consistent with key performance indicator (KPI) measures.

According to the Intel Corporation[5] PayPal reduced the number of missed fraudulent transactions by 30X by improving Service Level Agreement (SLA) adherence to 99.95% up from 98.5%. Paypal uses advanced deep learning models in order to prevent fraud. The analytics graph is used to uncover unknown patterns using graph algorithms and graph technologies have proven to be very effective in fraud detection and prevention. Today, almost all PayPal risk models are using graph features[6].

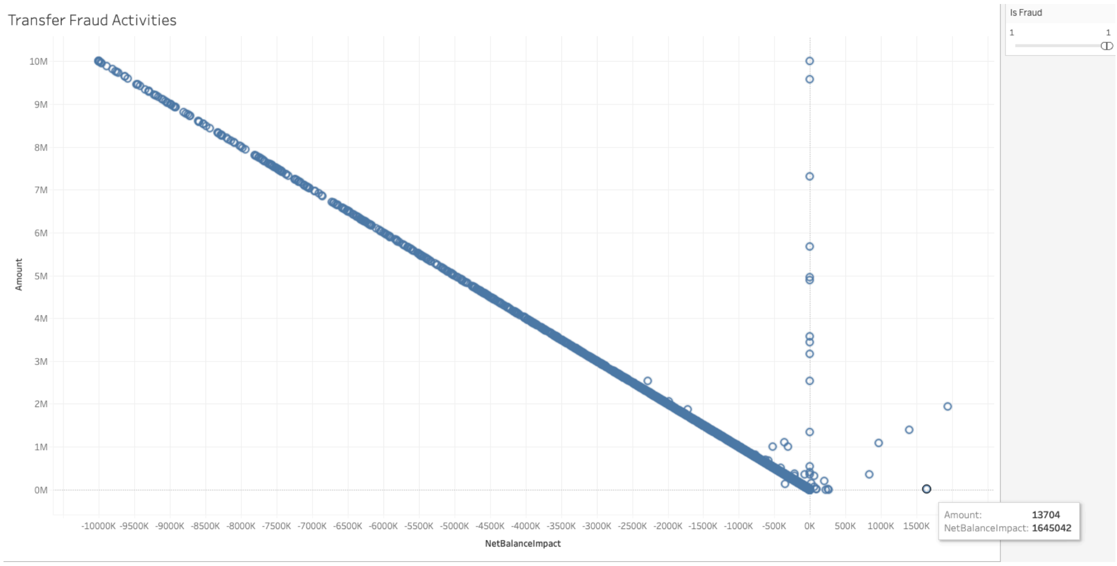
9. Discuss limitations of synthetic data and suggest improvements to enhance model realism and effectiveness.

One of the synthetic data limitations is that it can not capture complex real world examples therefore when comparing to real world examples, synthetic data is simpler and non-diverse. However our model results are satisfying when the model is implemented it would not perform well with real data. Synthetic data quality is very important to reflect real data therefore the tools or algorithms that are being used to create synthetic data should be designed to reflect real data.

To analyze further the most fraudulent transaction type, amount - net balance impact scatter plot is drawn.

[5] Intel Corporation. (2023) 'PayPal Solves Fraud Challenges with Aerospike® and Intel® Optane™ Persistent Memory'. Available at: https://www.intel.com/content/dam/www/central-libraries/us/en/documents/2023-08/aerospike-paypal-case-studyl.pdf (Accessed: 26 January 2025).

[6] Zuo, Q. (2021) 'How PayPal uses real-time graph database and graph analysis to fight fraud', PayPal Tech. Available at: https://medium.com/paypal-tech/how-paypal-uses-real-time-graph-database-and-graph-analysis-to-fight-fraud-96a2b918619a (Accessed: 26 January 2025).



Is Fraud
1 1

Amount: 13704
NetBalanceImpact: 1645042

Image 18: Transfer type fraud activities

In the transfer fraud activities chart, there is a negative correlation with amount and net balance impact which is expected for fraudulent events although there are data points where net balance impact is on the positive sides which supports why synthetic data should be qualified with real world examples. These anomalies could be related to how this synthetic data is generated.

In order to enhance model realism, adding controlled noise and random outliers to synthetic data to better reflect real world scenarios.

References

1. capitaloneshopping.com (2024) 'Number of Credit Card Transactions per Second & Year: 2023 Data'. [online] Available at: https://capitaloneshopping.com/research/number-of-credit-card-transactions/ (Accessed: 25 December 2024).
2. Kaur, A. and Kaur, B. (2021) 'Fraud detection in financial transactions using machine learning (abstract)', *International Journal of Smart Device and Computing Science*, 9(3). Available at: https://ijsdcs.com/index.php/ijsdcs/article/view/639/253 (Accessed: 25 December 2024).
3. Wang, Y., Chen, Z. and Zhou, Z. (2021) 'A comprehensive survey on financial fraud detection (abstract)', *Journal of LaTeX Class Files*, 14(8), p. 1.
4. Lamberti, A. (2023) 'The benefits and limitations of generating synthetic data'. [online] *syntheticus.ai*. Available at: https://syntheticus.ai/blog/the-benefits-and-limitations-of-generating-synthetic-data (Accessed: 25 December 2024).
5. Intel Corporation (2023) 'PayPal Solves Fraud Challenges with Aerospike® and Intel® Optane™ Persistent Memory'. Available at: https://www.intel.com/content/dam/www/central-libraries/us/en/documents/2023-08/aerospike-paypal-case-studyl.pdf (Accessed: 26 January 2025).
6. Zuo, Q. (2021) 'How PayPal uses real-time graph database and graph analysis to fight fraud', *PayPal Tech*. Available at: https://medium.com/paypal-tech/how-paypal-uses-real-time-graph-database-and-graph-analysis-to-fight-fraud-96a2b918619a (Accessed: 26 January 2025).

Appendix

Project Repository:
The source code and related data for this project can be accessed at the following link:
https://github.com/Q1065727/FraudDetection

Dataset:
Synthetic dataset csv zip file is uploaded to google drive because of the high size.
https://drive.google.com/file/d/14pjR23Uuqh_dNpJWDyYQYvdF_r2xGWrk/view?usp=sharing