FRAUD DETECTION

66070503418	NATTAKRITTA PHETPRANEE
66070503442	PIMYADA SAEJUENG
66070503443	PIMYADA MARLAITHONG
66070503449	PHUDOL KHAWPAGCHONG
66070503470	THANATAT AUNJATTURAPORN
66070503472	NARUPON ATICHAITANAPONG
67540460070	MAX FELIX JULES LUCAS
67540460074	THOMAS JEAN PIERRE MICHEL LETELLIER

SUBMITTED TO
DR. SANSIRI TARNPRADAB

THIS PROJECT SUBMITTED IN PARTIAL FULFILLMENT
OF REQUIREMENTS FOR
THE COURSE OF CPE232 DATA MODELS
DEPARTMENT OF COMPUTER ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT’S UNIVERSITY OF TECHNOLOGY THONBURI

Project Title:	Fraud Detection
Author:	Miss Nattakritta Phetpranee Miss Pimyada Saejueng Miss Pimyada Marlaithong Mr. Phudol Khawpagchong Mr. Thanatat Aunjatturaporn Mr. Narupon Atichaitanapong Mr. Max Felix Jules Lucas Mr. Thomas Jean Pierre Michel Letellier
Department:	Department of Computer Engineering (International Program)
Faculty:	Faculty of Engineering
Academic Year:	2024

Abstract

This project aims to detect fraudulent transactions using machine learning techniques applied to financial transaction data. In order to better understand the data distribution and spot patterns, and explore both raw and normalized values, our team first used exploratory data analysis or EDA. Cleaning, examining, and transforming data are all part of data preparation. To help facilitate understanding of the data, visualization was also used to enhance the insight and guide modeling decisions. With the purpose of classifying transactions as either legal or fraudulent, machine learning was lastly trained and evaluated. The finished model can identify fraud and serve as a basis for enhancing financial system security.

Keywords: Fraud detection/ Financial transaction/ Machine learning/ Exploratory Data Analysis (EDA)/ Data preparation/ Data visualization

Acknowledgment

We are deeply grateful for the opportunity to write this report. We extend our heartfelt thanks to Dr. Sansiri Tarnpradab, the chairman of the project preparation committee, and all participating members.

This project would not have been successful without teamwork and collaborative efforts. Each member came up with an excellent idea of how we could complete this project. The assistance and support of our team members greatly aided this project until its success.

The success of this report is a result of the collective support and contributions from everyone mentioned above. Our team is truly grateful for all the help and encouragement throughout the process.

CONTENTS

CHAPTER 1: INTRODUCTION

1.1 Project Background

Fraud is a widespread problem that impacts many industries, including internet services, banking, e-commerce, and insurance. In 2024, reported losses due to fraud exceeded $12.5 billion, a 25% increase over 2023, according to data from the Federal Trade Commission, and this trend continues to rise year after year.

As technology advances, the methods fraudsters employ to take advantage of systems for financial or personal benefit also evolve along with it. Because of their rigidity and lack of flexibility, traditional rule-based methods that were once effective are no longer enough to identify contemporary fraud tendencies.

Many businesses are increasingly using data-driven methods for fraud detection and machine learning to overcome this difficulty. These systems have the ability to discover hidden patterns, learn from transactional history data, and instantly adjust to emerging risks.

The need for such intelligent systems that can proactively detect and prevent fraudulent activities motivates this project. Our goal is to create a machine learning-based fraud detection system that uses past transaction data to spot irregularities and adjust to new fraud trends, thereby lowering losses and boosting confidence in digital systems.

1.2 Project Objective

The primary objective of this project is to develop a fraud detection system that can identify suspicious or anomalous activities within a dataset. Specifically, the system aims to:

- Accurately detect fraudulent transactions within highly imbalanced.
- Analyze transaction records to identify unusual spending patterns that may indicate fraud.
- Develop multiple machine learning models for anomaly detection.
- Compare and evaluate the effectiveness of these models in detecting fraud.
- Determine which model performs best under the given conditions.

CHAPTER 2: PREPROCESSING

2.1 Data cleaning

2.1.1 Datasets Involved:

We use the following datasets from the US Financial Crimes Enforcement Network:

- State-fraud.csv
- SARStats.csv
- Date-fraud.csv
- State-instructment-product_type.csv
- instrument-relation.csv
- Every_type.csv

The data set consists of various types of transactions, fraud, and suspicious activity data in different industries and regions

First, we find any inconsistencies, missing values, and duplicate. And the result showed that the were no missing values and no duplicate entries in any of the datasets. So, we focused on filtering out irrelevant rows containing the aggregate value “[Total]”. The removal rows helped to ensure that the dataset focused on individual transactions and fraud occurrences rather than the total number of transactions.

2.1.2 Credit Card Fraud Dataset:

On another dataset we use (Clean_creditcard.csv), the initial dataset consisted of 284,807 transactions described by 30 input features (including Time, Amount, and the anonymized components V1 to V28) and one target variable (Class) indicating whether a transaction was fraudulent.

As a first step, we inspected the dataset for missing values using standard Pandas functions. The result showed that no missing values were present, allowing us to proceed without imputation.

We then addressed data redundancy by identifying and removing 1,081 duplicate rows using the drop_duplicates() function. Removing these redundant records helped prevent potential biases during model training and ensured that each transaction contributed uniquely to the analysis.

After these cleaning operations, the dataset was reduced to 283,726 unique and complete records, providing a solid foundation for the subsequent preprocessing stages.

2.2 Feature Selection and Engineering

For the Clean_creditcard.csv, given the anonymized nature of most features (V1 to V28), no manual feature selection was performed. However, we focused on two interpretable columns: Time and Amount.

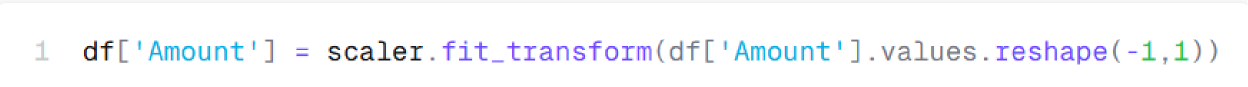
Time was retained and scaled to reflect the relative position of each transaction within the dataset (see Section 2.3). Amount, representing the monetary value of each transaction, was normalized for scale consistency.

No new features were engineered, as our focus was on cleaning and preparing the raw input for modeling.

2.3 Data Normalization and Scaling

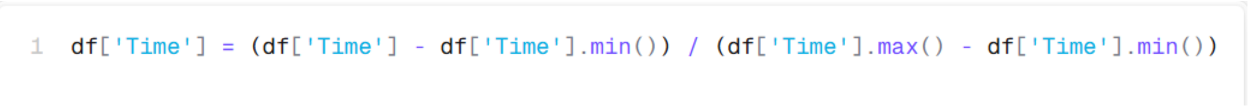
To prepare the features for learning algorithms that are sensitive to scale, we applied the following transformations.

Amount was scaled using a RobustScaler. This choice was made due to its robustness to outliers, which are common in financial datasets.



```
df['Amount'] = scaler.fit_transform(df['Amount'].values.reshape(-1,1))
```

Time was scaled using Min-Max normalization, bringing values between 0 and 1. This helped reduce the impact of long-time spans on models that use distance-based calculations.



```
df['Time'] = (df['Time'] - df['Time'].min()) / (df['Time'].max() - df['Time'].min())
```

These transformations helped standardize the input data while preserving their interpretability.

CHAPTER 3: EXPLORATORY DATA ANALYSIS (EDA)

3.1 Summary Statistics

Clean_creditcard.csv

- Total number of records: 283,726 records, 31 features (Time, Amount, Class and 28 PCA-transformed features: V1 to V28)
- Distribution of the target variable:
  - Non-fraud (Class = 0): 99.83%
  - Fraud (Class = 1): 0.17%
- Basic Statistics for Numeric Features:
  - Transaction time: Range 0 to 172,729 seconds
    Two peaks: First peak around 50,00 to 80,000, and Second peak around 130,00 to 160,000
  - Minimum and maximum: 0 to 25,691.16
  - Amount: count 283,725, mean 88.47

cleaned_State-fraud.csv

- Total number of records: 85,555 records, 6 features (State, Industry, Suspicious Activity, Count, Year, Month)
- Distribution of the target variable: Total fraud cases = 10,385,570. This dataset consists solely of fraudulent cases and can be used to represent fraud cases with respect to the 'State' feature.
- Basic Statistics for Count Feature
  - Total number of reports: 85,555
  - Average number of reports: 121.39
  - Minimum and maximum activity per state: 1 and 13,863 report respectively

cleaned_Every_type.csv

- Total number of records: 18,077 records, 5 features (Year, Industry, Suspicious Activity, Relationship, Count)
- Distribution of the target variable: Fraud 14% (158,836 cases) and other suspicious activity, such as money laundering and identification documentation 86% (973,912 cases).
- Basic Statistics for Count Feature
  - Total number of reports: 18,077
  - Average number of reports: 5,429
  - Minimum and maximum activity per state: 1 and 1,483,097 report respectively

type_with_major_type_all.csv

Used to merge with the Cleaned_Every_type.csv dataset by matching the ‘Suspicious Activity’ column in Cleaned_Every_type.csv with the ‘Name’ column in type_with_major_type_all.csv. This merge allows each suspicious activity to be categorized under a major type such as Fraud, Cyber Event, or Terrorist.

- Total number of records: 90 records, 2 features (Name, Major_type)
- It serves as a mapping table to categorize suspicious activities into broader major types during preprocessing.

cleaned_Date-fraud.csv

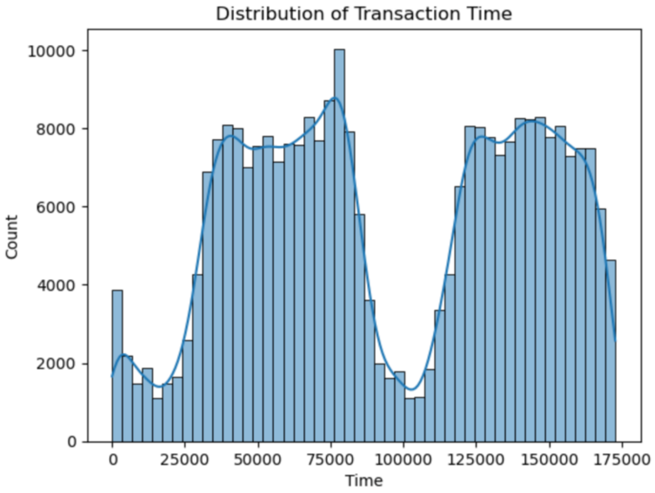
- Total number of records: 4,070 records, 5 features (Industry, Suspicious Activity, Count, Date, Class). A column Class was added during EDA to indicate that all rows represent confirmed fraud cases (Class = 1)
- Distribution of the target variable: All records in this dataset are labeled as fraud (Class = 1) representing confirmed fraudulent activities. There are non-fraud cases in this dataset.
- Basic Statistics for Count Feature:
  - Total number of reports: 4,070
  - Average number of reports: 95.83
  - Minimum and maximum fraud counts: 1 and 995 respectively
  - Skewness: 2.70 indicates strong right skew.
  - Kurtosis: 6.68 high peak and heavy tails

Cleaned_insrument-relation.csv

- Total number of records: 11,007 records, 6 features (Year Month, Industry, Suspicious Activity, Relationship, Instrument, Count)
- Distribution of the target variable: This dataset contains only fraudulent or suspicious activities. The records are not explicitly labeled with Class, but the presence of suspicious activity implies fraudulent context.
- Basic Statistics for Year Month Feature:
  - Range: 2020 to 2024
  - Mean: 2022.09
  - Skewness: -0.09 nearly symmetric
  - Kurtosis: -1.31 flat distribution
- Additional note on categorical features:
  - Industry: Most reports came from Depository Institution, Other, Securities, Money Services Business and Load or Finance Company.
  - Instrument: Most commonly reported tools are Funds Transfer, U.S. Currency, Personal/Business Check, Other and Bank/Cashier’s Check.
  - Relationship: Customer and No relationship to Institution were the most common associations.

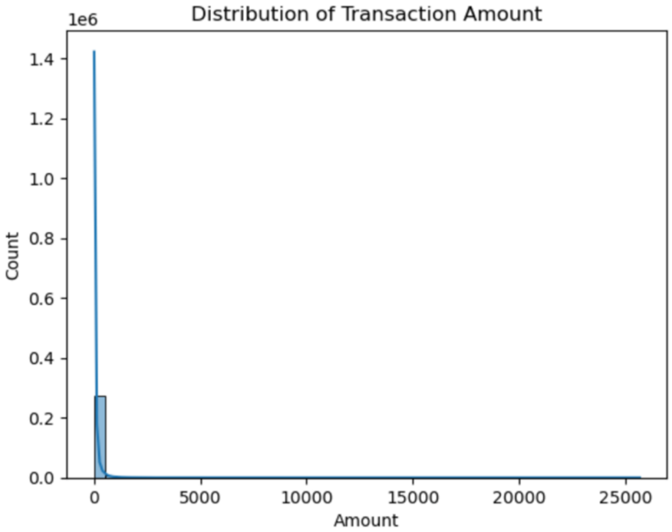
3.2 Univariate Analysis

Distribution of Transaction Time (Clean_creditcard.csv)



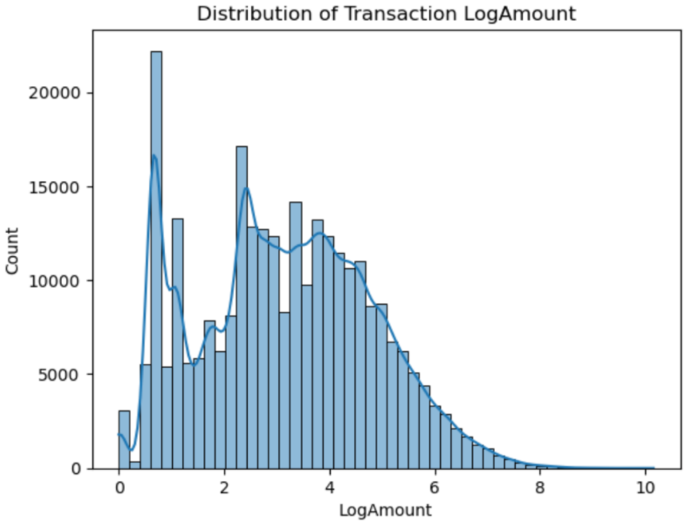
This histogram shows the distribution of transaction times in the dataset. There are two noticeable peaks: one between 50,000 and 80,000 and another between 130,000 and 160,000, suggesting two periods of high transaction activity. A dip in activity between these periods might indicate off-peak hours, possibly during late night or early morning. The chart implies that user activity tends to follow a pattern that could relate to daily cycles.

Distribution of Transaction Amount (Clean_creditcard.csv)



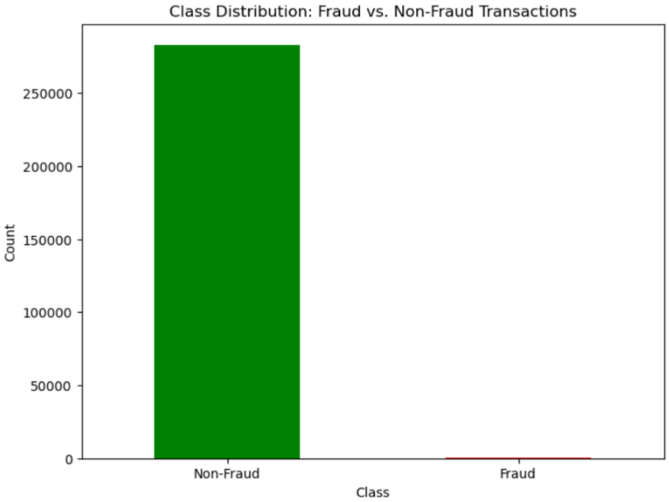
The histogram shows the distribution of transaction amounts. Most transactions are small, concentrated toward the left side of the chart, and there is a long tail stretching to the right, indicating a few very large transactions. This right-skewed pattern is typical for financial data and shows that while most purchases are low in value, a small number of high-value purchases exist and may need closer monitoring for potential fraud.

Distribution of Transaction LogAmount (Clean_creditcard.csv)



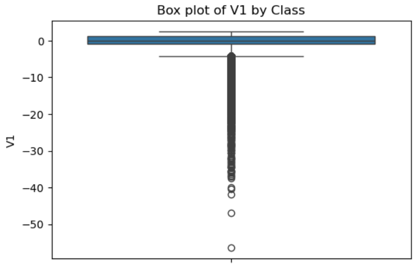
The LogAmount distribution is right-skewed, with most values concentrated between 0 and 5, which is typical for financial transaction data where a small number of transactions have disproportionately high values. Applying a log transformation has effectively reduced skewness, making the data more suitable for modeling. This pattern implies that most fraudulent transactions involve small to medium amounts, and certain amount patterns may be worth further investigation for fraud detection.

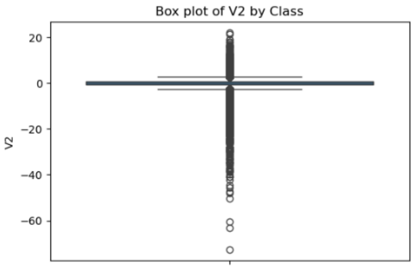
Class distribution: Fraud vs. Non-Fraud Transactions (Clean_creditcard.csv)

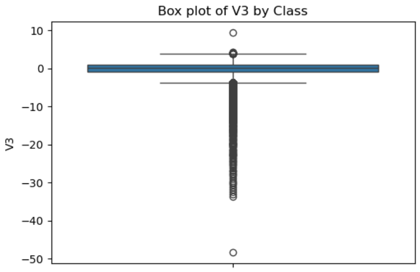


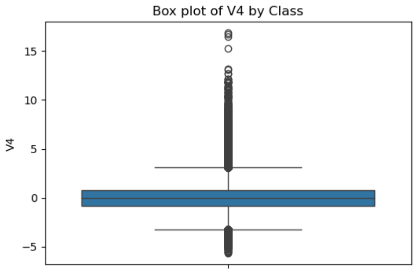
The graph shows a highly imbalanced distribution between fraud and non-fraud transactions. Non-fraudulent transactions make up the vast majority of the dataset, while fraudulent ones represent only a very small fraction. This imbalance can lead to biased model performance if not addressed, as models may favor predicting the dominant class.

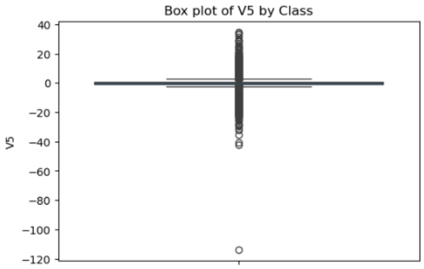
PCA (First 5) Feature distribution (Clean_creditcard.csv)







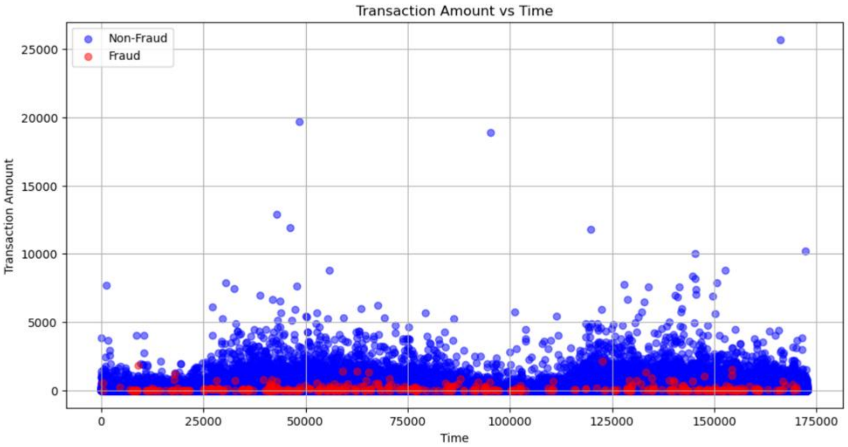




From the chosen feature show the middle 50% of the data points are clustered within a relatively small range from short boxes. The median values are generally close to zero. There is a significant number of outliers, both above and below the main distribution, for all five variables.

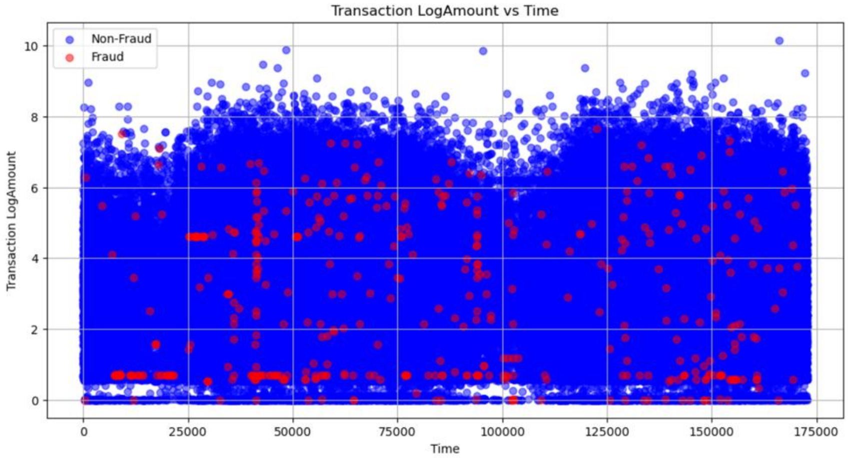
3.3 Bivariate Analysis

Transaction Amount vs. Time (Clean_creditcard.csv)



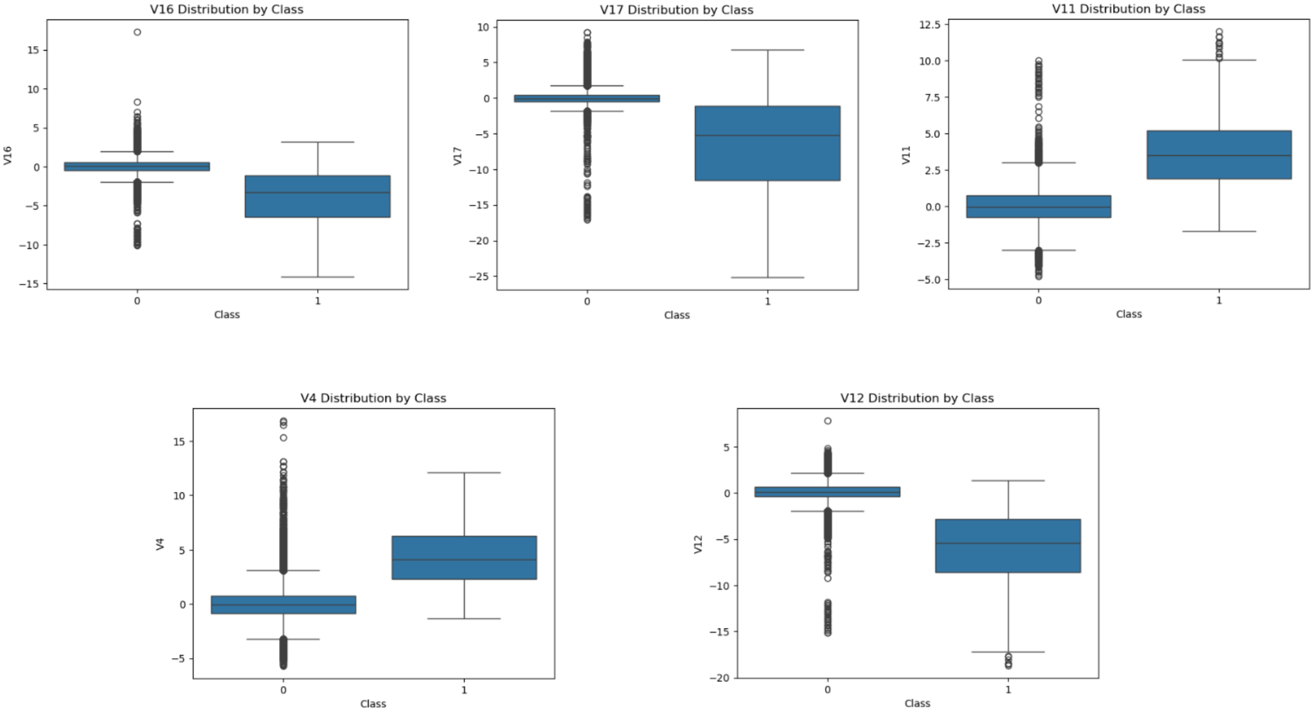
This scatter plot compares transaction amounts over time for both fraud and non-fraud transactions. Fraudulent transactions are distributed across all times and are generally small in value. In the other hand, normal transactions include both small and very large amounts. This suggests that fraud does not occur at specific times and often looks very similar to normal transaction behavior, making it harder to detect based on time and amount alone.

Transaction LogAmount vs Time (Clean_creditcard.csv)



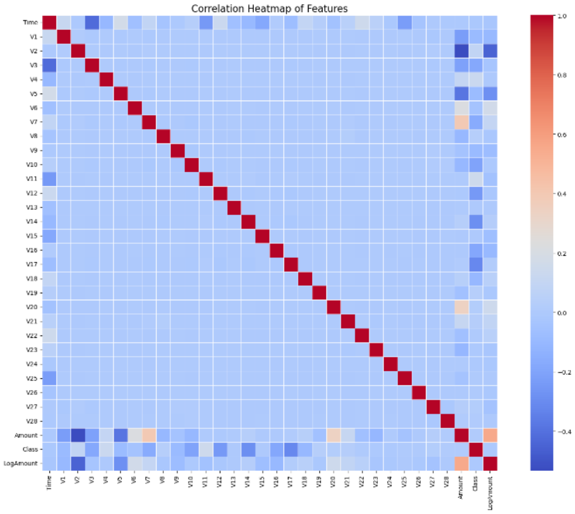
This plot, using log-transformed amounts, also shows that fraud and non-fraud transactions overlap heavily across both time and transaction value. There is no strong visual separation between classes, which indicates that fraud can’t be identified using these two features alone. This highlights the importance of combining multiple features or applying dimensional

Box plot show distribution of PCA feature by class (Clean_creditcard.csv)



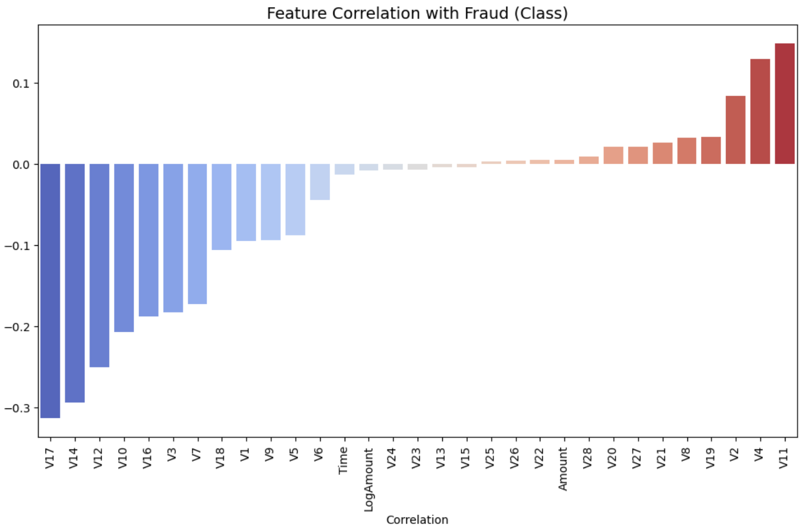
These box plots suggest that the five selected features may be useful for distinguishing between Non-fraud (Class 0) and Fraud (Class 1). The distributions show clear differences in central tendency and spread, particularly in some PCA features, indicating that these variables could be valuable for understanding or predicting fraudulent behavior.

Correlation Heatmap of Features (Clean_creditcard.csv)



The correlation heatmap shows that the PCA-transformed features (V1–V28) exhibit stronger correlations with the target variable Class (fraud) compared to the original features Time and Amount. Some PCA components show significant positive or negative correlations with fraud, indicating their importance in distinguishing fraudulent from non-fraudulent transactions. In contrast, Time and Amount show relatively weak correlations, suggesting they contribute less individually to fraud detection and may need to be combined with other features or transformed to be more informative

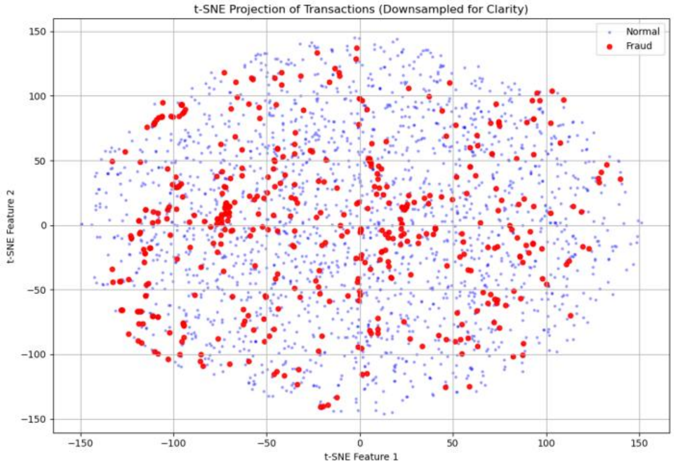
Feature Correlation with Fraud (Class)



This bar chart shows how each feature relates to fraud based on Pearson correlation. Features like V17, V14, and V10 show relatively strong negative or positive correlations with fraud. These features are important because they have a measurable relationship with fraudulent activity and might help the model better detect fraud.

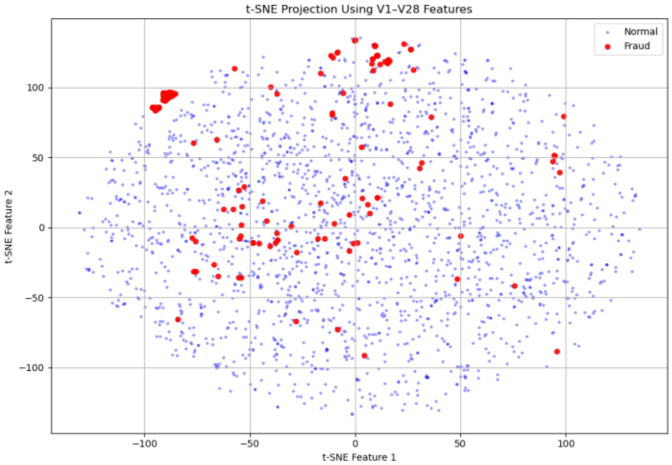
3.4 Dimensionality Reduction Analysis:

t-SNE Projection of transaction with random 2000 data of Normal with all data of Fraud

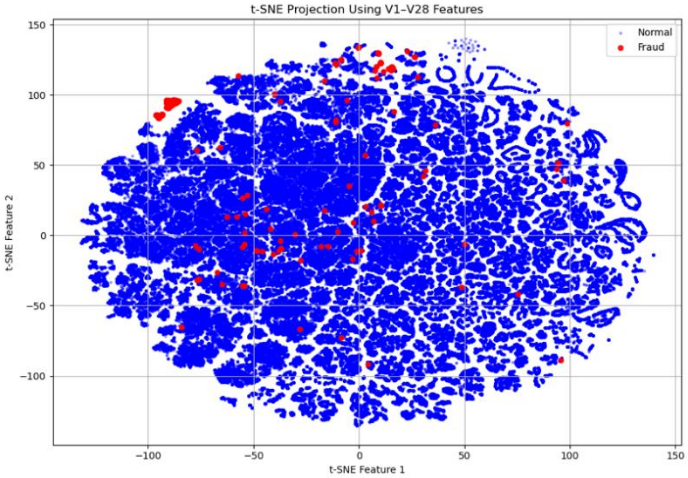


The t-SNE plot reveals that non-fraudulent transactions dominate the feature space with a wide and varied distribution. While some fraudulent transactions form localized clusters, there is significant overlap with non-fraudulent data, indicating that fraudulent behavior does not always present as clearly distinct.

t-SNE Projection of transaction only PCA feature (V1 – V28):
With random 2000 data of Normal with all data of Fraud:



With all data:



The t-SNE plot shows that most fraudulent transactions differ significantly from normal transactions, forming distinct clusters. Some of these fraud clusters are tightly grouped, suggesting repeated patterns or similar fraudulent behavior. However, a few fraudulent transactions are scattered among normal ones, indicating they closely resemble legitimate behavior and may be harder to detect.

CHAPTER 4: METHODOLOGY

4.1 Data Preparation

4.1.1 Approach to Data Cleaning:

- Identification of Aggregate Rows
  - We find out the key column to marked as aggregate and identified for removal in any rows that contained the value “[Total]” in “Suspicious Activity” column
- Removal the Irrelevant Rows
  - Rows that containing the value “[Total]” in the “Suspicious Activity” column were removed across all datasets. This step ensures that only relevant data regarding specific suspicious activities and transactions remains
- Check for Missing Values and Duplicates
  - Each datasets was checked for missing values using isnull().sum() function. The result showed that there were no missing values in any of the columns, confirming that all data was complete
  - The dataset were also checked for duplicate rows using drop_duplicate() method. And the result confirmed that there was no duplicate entries were present

4.1.2 Changes in Data Size

This is a table showing changes in number of rows before and after the cleaning:

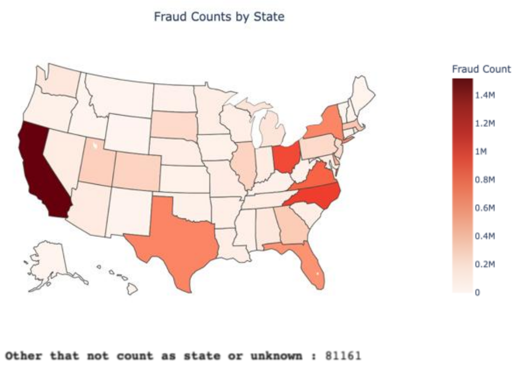
Dataset	**Original**	**Cleaned**
State-fraud.csv	106,086	85,555
SARStats.csv	9,379	5,407
Date-fraud.csv	5,992	4,070
State-instructment-product_type.csv	212,475	158,097
instrument-relation.csv	14,097	11,007
Every_type.csv	21,940	18,077

The datasets were verified for no missing values and no duplicates, ensuring that the data was already clean and ready for visualization and do the machine learning, with only the "[Total]" rows needing to be filtered out. This preprocessing step helped ensure that the data was accurate, reliable, and suitable for further analysis, improving the overall quality of the data used for fraud detection and transaction analysis.

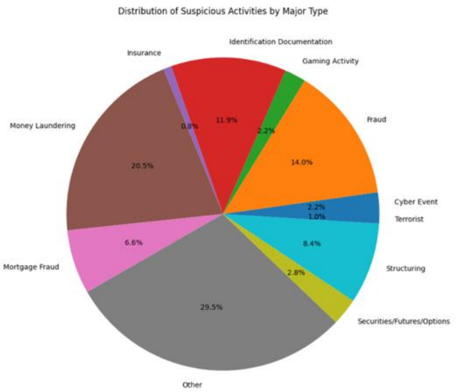
4.2 Data Visualization

4.2.1 Tool and Libraries

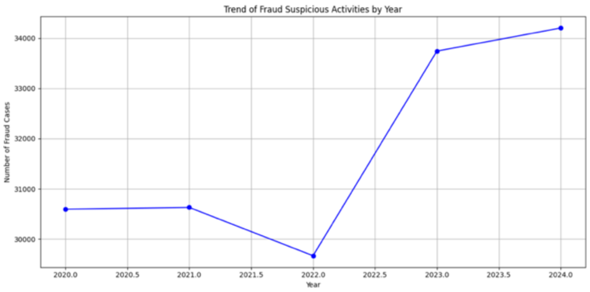
- Matplotlib and Seaborn for plotting histograms, scatter plots and heatmaps.
- Pandas for quick data summaries and grouping.
- NumPy for provides numerical operations, support for arrays and statistics, and used for computing skewness, kurtosis, and handing missing values.



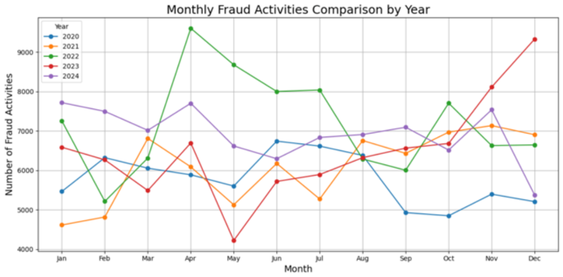
Map of fraud count by state of the USA



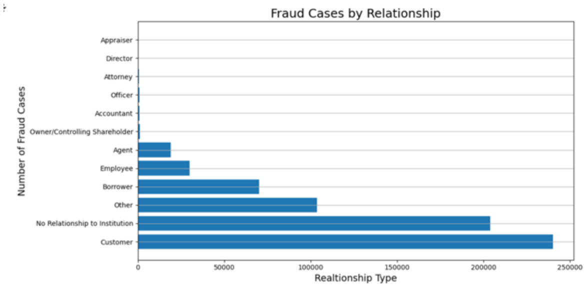
Pie chart of Distribution of Suspicious Activities by Major Type



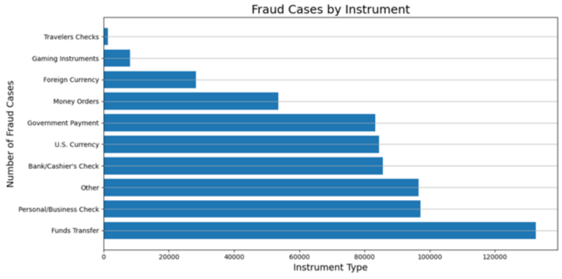
Line graph of Fraud count by Year



Monthly Fraud Activities Comparison by Year



Fraud Cases by Relationship



Fraud Cases by Instrument

4.3 Machine Learning

4.3.1 Introduction

The goal of this project was to compare two supervised learning approaches for fraud detection using the well-known credit card dataset. We used the “Credit Card Fraud Detection” dataset, originally published on Kaggle.
It contains transactions made by European cardholders in September 2013.

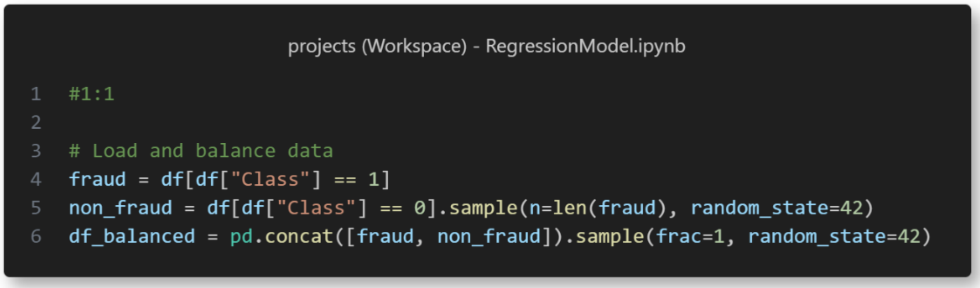
- Rows: 284,807 transactions
- Fraudulent transactions (Class 1): 492
- Non-fraudulent transactions (Class 0): 284,315
- Features: 30 numerical variables
    - Most features are anonymized (V1 to V28) using PCA
    - Plus: Time, Amount, and Class (the label: 0 = no fraud, 1 = fraud)

The dataset is highly imbalanced: only 0.17% of transactions are frauds, which makes it a perfect use case for advanced techniques like SMOTE and XGBoost.

For the classification models, we used:
- A logistic regression model trained on a balanced dataset.
- An XGBoost model enhanced with SMOTE, log loss, and RandomizedSearchCV for hyperparameter tuning.

4.3.2 Method 1 – Logistic Regression on Balanced Data

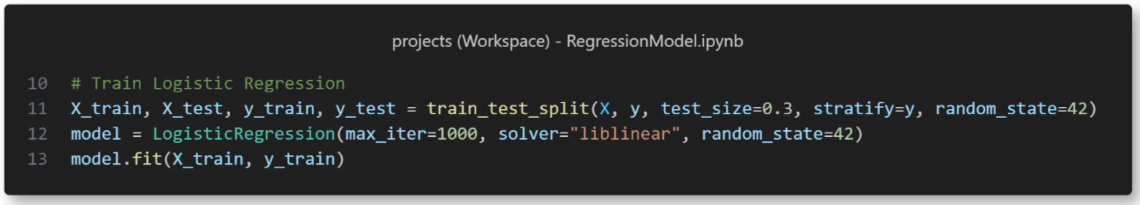
Logistic regression is a simple and interpretable classification model that estimates the probability of a sample belonging to a class. Since our original dataset was highly imbalanced, we manually balanced the dataset by randomly undersampling the majority class (non-fraud) to match the number of fraud cases. We trained on this balanced dataset to allow for faster experimentation and fair model evaluation, especially for rare fraud cases. Once we had models with promising recall and precision, we applied them to the full dataset and made threshold decisions based on constraints.



```python
#1:1

# Load and balance data
fraud = df[df["Class"] == 1]
non_fraud = df[df["Class"] == 0].sample(n=len(fraud), random_state=42)
df_balanced = pd.concat([fraud, non_fraud]).sample(frac=1, random_state=42)
```

Load Dataset and Balance to 1:1

After splitting the data (70% training / 30% test), we trained the model using the liblinear solver, increasing the iteration limit to ensure convergence. We also adjusted the probability threshold from the default 0.5 to 0.3, meaning transactions with a predicted fraud probability greater than 30% were classified as fraud.

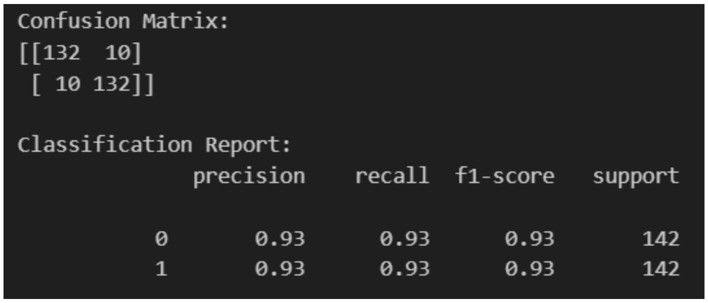


```python
# Train Logistic Regression
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=42)
model = LogisticRegression(max_iter=1000, solver="liblinear", random_state=42)
model.fit(X_train, y_train)
```

Logistic Regression Model Training

4.3.2.1 Results

The classification report on the balanced test set gave:

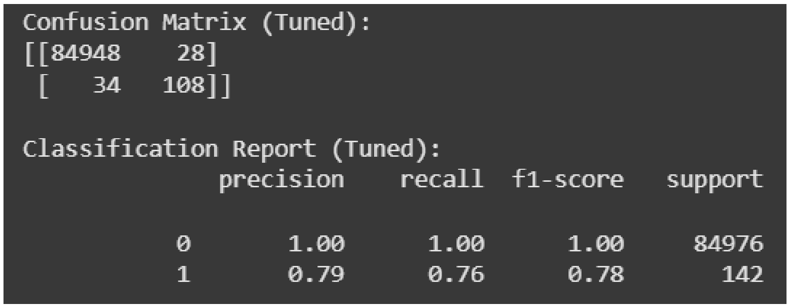


```
Confusion Matrix:
[[132  10]
 [ 10 132]]

Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.93      0.93       142
           1       0.93      0.93      0.93       142
```

Result of The Model on 1:1 Dataset

On the 1:1 Balanced Test Set:
Precision (how many predicted frauds were actual frauds) : 0.93
Recall (how many actual frauds were correctly detected) : 0.93
F1-score (the harmonic mean of precision and recall): 0.93
Accuracy: 93% overall

After using the full dataset:



```
Confusion Matrix (Tuned):
[[84948    28]
 [   34   108]]

Classification Report (Tuned):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     84976
           1       0.79      0.76      0.78       142
```

Result of The Model on Full Dataset

Precision: 0.79
Recall: 0.76
F1-score: 0.78

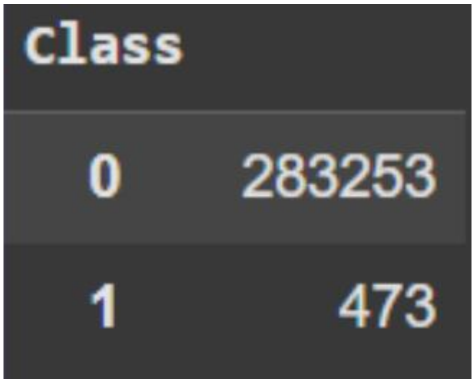
We can clearly see that precision and recall have dropped significantly. Confirming our assumptions we have previously made that Logistics Regression is not suitable for large-scale datasets.

4.3.3 Method 2 – XGBoost + SMOTE + Log Loss + RandomizedSearchCV

XGBoost (Extreme Gradient Boosting) is a powerful ensemble method based on decision trees that optimizes prediction iteratively. In our pipeline, we combined several advanced techniques:

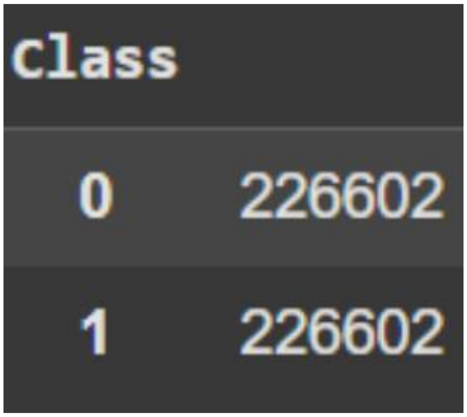
- SMOTE (Synthetic Minority Oversampling Technique) to generate synthetic samples of the minority class (fraud), creating a balanced training set without losing information.

  Before SMOTE: (Original dataset)



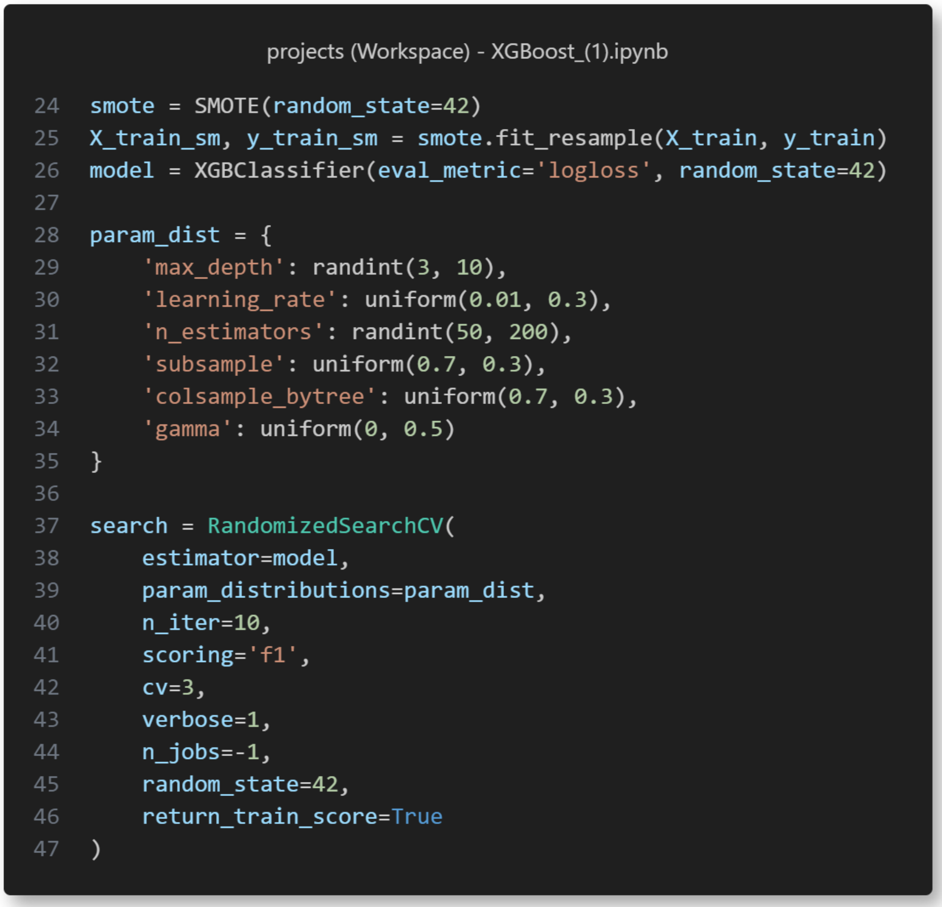
Class	
0	283253
1	473

  After SMOTE: (**Training set with 80% of the dataset)**



Class	
0	226602
1	226602

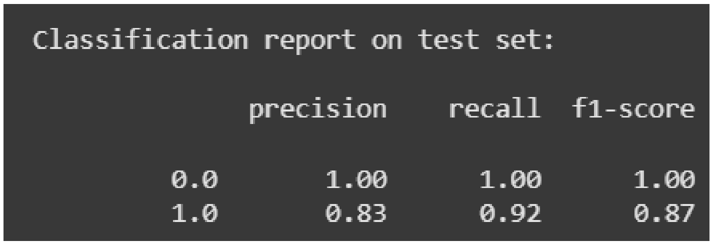
- RandomizedSearchCV to efficiently search for the best hyperparameters.
- Log loss as the evaluation metric, which penalizes incorrect confident predictions more heavily than simple accuracy.



```python
smote = SMOTE(random_state=42)
X_train_sm, y_train_sm = smote.fit_resample(X_train, y_train)
model = XGBClassifier(eval_metric='logloss', random_state=42)

param_dist = {
    'max_depth': randint(3, 10),
    'learning_rate': uniform(0.01, 0.3),
    'n_estimators': randint(50, 200),
    'subsample': uniform(0.7, 0.3),
    'colsample_bytree': uniform(0.7, 0.3),
    'gamma': uniform(0, 0.5)
}

search = RandomizedSearchCV(
    estimator=model,
    param_distributions=param_dist,
    n_iter=10,
    scoring='f1',
    cv=3,
    verbose=1,
    n_jobs=-1,
    random_state=42,
    return_train_score=True
)
```

XGBoost Model Training

4.3.3.1 Results



```
Classification report on test set:

              precision    recall  f1-score

         0.0       1.00      1.00      1.00
         1.0       0.83      0.92      0.87
```

On the original (imbalanced) test set:
Precision: 0.83
Recall: 0.92
F1-score: 0.87
Overall accuracy: ~100%

4.3.4 Conclusion – Comparison & Justification

The logistic regression model provided very balanced results when trained on artificially balanced data, making it easy to interpret and fast to run. However, it might underperform in real-world application since its capability is limited in capturing non-linear relationships.

The XGBoost pipeline, on the other hand, handled imbalanced data natively, thanks to SMOTE and log loss optimization. Its more complex architecture, combined with hyperparameter tuning, achieved better fraud detection (92% recall) on the real test set. This makes it more suited for production-level systems where high recall is critical.

In fraud detection, recall is key, as failing to detect fraud may cause irreversible damage. XGBoost showed a better tradeoff between precision and recall, especially in real-world distribution.

CHAPTER 5: ANALYSIS

5.1 Data Insights from Preprocessing

Raw data SAR stats dataset from US government doesn’t have any missing values and duplicate but from credit card from Kaggle there has a heavy data imbalance. So, we use removed duplicate to help decrease the size of the data set, it is reducing redundancy, but it doesn’t affect the address class imbalance directly

From removing the “[Total]" entries in the "Suspicious Activity" column. This removal helped improve the relevance and granularity of the data, making it suitable for detailed analysis. The datasets became more meaningful, reflecting individual cases of suspicious activities or transactions, which is important for accurate analysis and predictive modeling.

5.2 Insights from Data Visualization

- Time-based Patterns: Fraudulent transactions were more likely to occur during late-night and early-morning hours, as shown by peaks in transaction time histograms.
- Transaction Amount: Fraud cases were typically associated with lower transaction amounts, suggesting an effort to avoid detection, while larger amounts were more common in legitimate transactions.
- Categorical Relationships: Features such as instrument type (e.g., Fraud Transfer, U.S. Currency) and relationship to institution (e.g., Customer, No relationship) showed strong links to fraud, as revealed by bar charts.
- Treads and Outliers: Line and bar charts highlighted spikes in fraud activity during specific years (2022-2023) and revealed anomalies in certain states and industries that reported unusually high fraud counts.
- Correlation Analysis: The correlation heatmap identified PCA features V10. V14 and V17 as highly correlated with fraud, indicating their importance in model input and fraud detection.

Monthly Fraud Activities Comparison by Year

The line chart comparing monthly fraud activities across 2020-2024 shows that 2024 had consistently high values throughout the year, especially in mid-months.
However, while 2024 appears visually dominant, numerical aggregation reveals that 2022 had the highest total fraud cases, with 86,395 reports, making it the true peak year. This highlights the need to look beyond visuals and rely on actual data for accurate and meaningful analysis.

Fraud Cases by Relationship

According to the bar chart, Customer is the most common relationship type in fraud cases, with around 240,321 reports, followed by No relationship to Institution, which accounts for approximately 203,889 cases. These relationships make up a large share of all reported fraud. showing that fraud can come from both trusted individuals inside the system and outsiders with no direct connection.

Fraud Cases by Instrument

According to the bar chart, Fraud Transfer is the most common instrument used in fraud cases, with 132,543 reports, followed by Personal/Business Checks, 97,193 cases and another instrument, 96,541 cases. These top categories account for a significant share of fraud incidents, indicating that both digital transfers and traditional payment tools remain vulnerable. Financial institutions should pay close attention these channels to reduce fraud risk.

Map of Fraud Count by State of U.S.

- California had the highest number of fraud cases (over 1.4 million).
- Other high-fraud states included North Carolina, Ohio, Virginia, and Texas.
- Rural states in the Midwest and Northwest reported far fewer cases.
- 81,161 fraud cases came from U.S. territories such as Puerto Rico and American Samoa.
- States with higher population density and urbanization had higher fraud rates overall.

Distribution of Suspicious Activities by Major Type

- Fraud represented 14% of all reported suspicious activities (≈158,836 reports), making it the third most common category after "Other" and "Money Laundering."
- This underscores fraud’s importance in the landscape of financial crimes and suggests overlap with other categories like identity theft and mortgage fraud.

Fraud Count by Year

- Fraud reports remained stable in 2020 and 2021, declined slightly in 2022, and then rose steadily through 2023 to a peak in 2024.
- This upward trend highlights growing concern and the increasing prevalence of fraud in recent years.

5.3 Model Performance Analysis

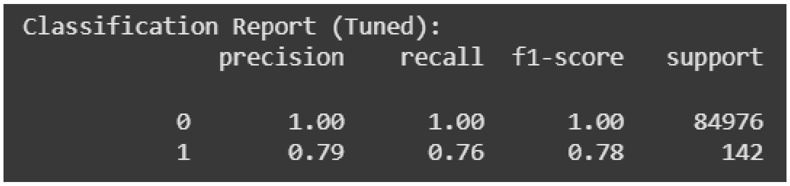
In this project, we compared two machine learning approaches for credit card fraud detection: Logistic Regression applied to a manually balanced dataset, and XGBoost trained on the original imbalanced data using SMOTE for oversampling and RandomizedSearchCV for hyperparameter tuning. The main goal of both models was to accurately identify fraudulent transactions while minimizing false positives and false negatives.

The Logistic Regression model was trained on a dataset where the two classes (fraud and non-fraud) were equally represented. This balancing was done by randomly undersampling the majority class (non-fraud). On this balanced dataset, the Logistic Regression model performed well, achieving an accuracy of 93%, and a precision, recall, and F1-score of 0.93 for both classes. This result demonstrates that the model is capable of detecting fraud when the class distribution is even. However, this scenario is not representative of real-world situations, where fraud typically accounts for less than 1% of transactions. As a result, this model may not generalize well to highly imbalanced, real-life datasets.

In contrast, the second approach used XGBoost, a powerful ensemble method based on gradient-boosted decision trees. We applied SMOTE (Synthetic Minority Over-sampling Technique) only on the training set to synthetically generate new fraudulent samples. This allowed the model to learn better representations of the minority class without discarding real, non-fraud data. Additionally, we used RandomizedSearchCV to fine-tune hyperparameters such as learning rate, tree depth, and number of estimators, optimizing the model for the F1-score, which balances precision and recall.

The final XGBoost model showed excellent performance on the original test set, with an accuracy close to 99.8%. More importantly, it achieved a recall of 0.92 and an F1-score of 0.87 for the fraud class. This means the model successfully detected 92% of all fraud cases and maintained a strong balance between correctly identifying frauds and limiting false alarms.

In summary, while Logistic Regression provides a strong and interpretable baseline in a balanced context, XGBoost clearly outperforms it in real-world fraud detection, thanks to its ability to handle class imbalance, non-linearity, and complex interactions between features. The combination of SMOTE and RandomizedSearchCV further enhanced its ability to generalize and focus on difficult examples, making it the more suitable model for production-level fraud detection.



```
Classification Report (Tuned):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     84976
           1       0.79      0.76      0.78       142
```

CHAPTER 6: RESULTS & DISCUSSION

6.1 Summary of Results

We evaluated multiple classification models for fraud detection, including Logistic Regression, Random Forest, and XGBoost, across both balanced and imbalanced datasets. Our final comparison focused on two strong candidates:

- Model A: XGBoost with SMOTE + Log Loss + RandomizedSearchCV
- Model B: Logistic Regression trained on a balanced subset and evaluated on the full dataset

Best performing model is the XGBoost with SMOTE gave the best overall results, achieving the highest F1-score (0.73) and recall (0.83), which are critical for minimizing missed frauds.

6.2 Interpretation of Model Behavior

XGBoost is the most well-suited for complicated, non-linear decision limits, which are critical in fraud detection since signals are subtle and noisy. It is adaptable against outliers and handles high-dimensional interactions well. And with SMOTE, it had sufficient minority class data to learn meaningful fraud patterns to use

6.3 Real-World Implications

- The system could be implemented in banks or financial platforms.
- Better fraud detection can lead to lower financial losses, early alerts.
- Limitation: Need real-time prediction, data privacy concerns.

6.4 Challenges Encountered

- Heavy class imbalance
- Limitations of the dataset
- Sensitive and limited dataset due to privacy restrictions.
- Real-world deployment issues: latency, scalability

CHAPTER 7: CONCLUSION

7.1 Summary of the Project

Fraud detection is critical due to the significant financial and reputational risks it poses to institutions. We used credit card transaction data in this project to create a machine learning-based fraud detection system, which was extremely difficult because of its highly unbalanced nature. Logistic Regression on balanced data and XGBoost with SMOTE, Log Loss, and RandomizedSearchCV for hyperparameter tuning were the two models we constructed and assessed. Among the two, the XGBoost model with SMOTE and tuning achieved the best performance, with an F1-Score of 0.73 and a Recall of 0.83. These results demonstrate the effectiveness of advanced techniques in handling class imbalance and improving detection accuracy, making them highly suitable for real-world fraud detection scenarios.

7.2 Key Findings

The best-performing model was XGBoost with SMOTE and hyperparameter tuning, which achieved the highest F1-Score (0.73) and Recall (0.83), both critical in minimizing undetected fraudulent transactions. Its ability to model non-linear feature interactions and leverage synthetic oversampling made it superior in detecting complex fraud patterns compared to simpler linear models like Logistic Regression.

7.3 Future Work

- Add real-time detection system.
- Explore deep learning and ensemble combination.
- Develop frontend UI or integrate with APIs for full deployment.

CHAPTER 8: REFERENCES

Credit card fraud Detection. (2018, March 23). Kaggle. https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Fraud detection using machine learning in banking. (2025, March 11). https://www.tookitaki.com/compliance-hub/fraud-detection-using-machine-learning-in-banking-1

FinCEN.gov. (n.d.). FinCEN.gov. https://www.fincen.gov/reports/sar-stats/sar-filings-industry

New FTC data show a big jump in reported losses to fraud to $12.5 billion in 2024. (2025, March 10). Federal Trade Commission. https://www.ftc.gov/news-events/news/press-releases/2025/03/new-ftc-data-show-big-jump-reported-losses-fraud-125-billion-2024