SOFT COMPUTING (CSE-458) PROJECT

ADVANCED FRAUD DETECTION IN FINANCIAL SYSTEMS

Group Number 66
Anshika Yadav (22124010)
Devika S Menon (22124014)
Shruti Agrawal (22124050)

INTRODUCTION

- This project presents an **advanced fraud detection system** for **credit card** transactions. Credit card companies need to detect fraudulent transactions to prevent customers from being charged for unauthorized purchases.

- By combining **deep learning** with **genetic algorithms** for feature selection, the model enhances detection accuracy, reduces false positives, and ensures reliable, real-time identification of fraudulent activity in financial systems.

- It aims to optimize performance by selecting the **most relevant features** and training a robust neural network model.

DATASET INFORMATION

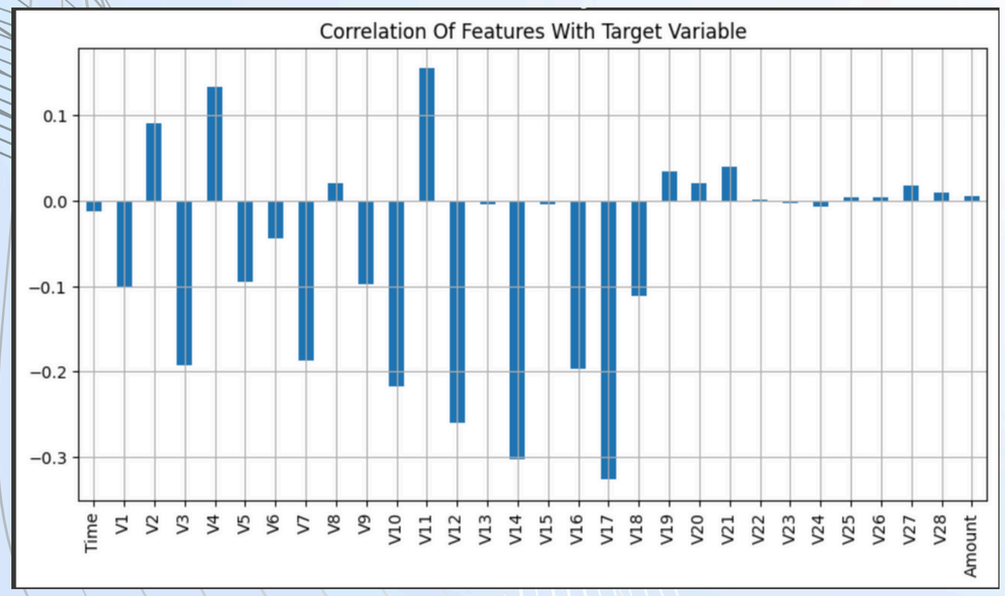
- **Dataset:** Transactions made by European cardholders in September 2013.
- **Duration:** Two days, with 492 frauds out of 284,807 transactions.
- **Class Imbalance:** Fraudulent transactions (positive class) account for 0.172% of all transactions. Hence it is an imbalanced dataset.
- **Features:**
  - Numerical input variables resulting from **PCA** transformation.
  - 'Time': Seconds elapsed between each transaction and the first transaction.
  - 'Amount': Transaction amount, suitable for cost-sensitive learning.
- **Target:**
  - 'Class': Response variable, 1 for fraud, 0 otherwise (non-fraud).

- In the dataset, we have a total of 31 features with 3 of them being time, amount and class and the rest of 28 features being numerical inputs which are a result of PCA transformation, due to confidentiality issues.

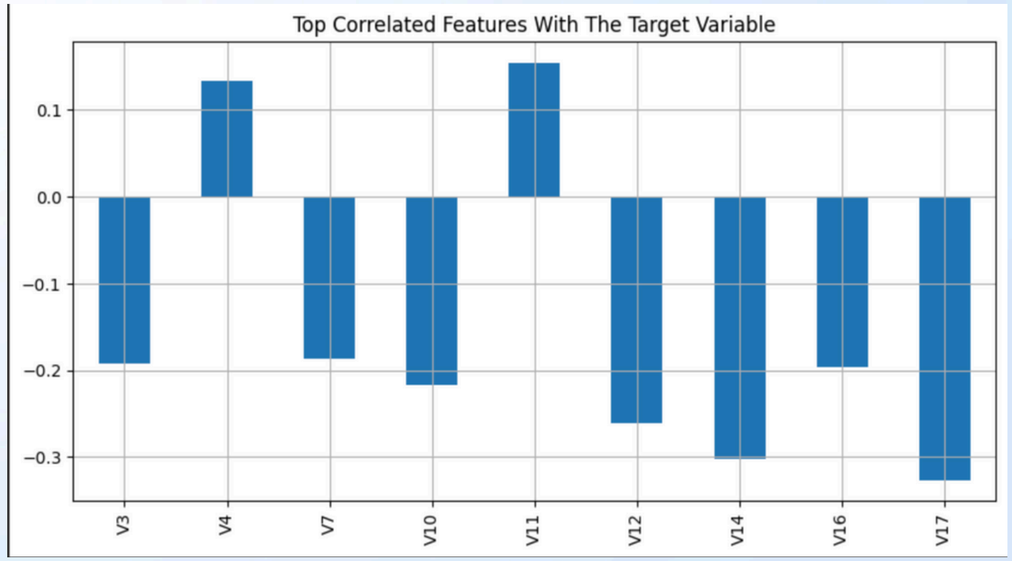
- Credit card fraud datasets, including this one, are typically highly imbalanced because occurrences of fraud are rare compared to normal transactions. Hence, we employed **feature selection** using a **Genetic Algorithm** to focus on the most relevant attributes, enhancing model performance and reducing overfitting.

FEATURE SELECTION

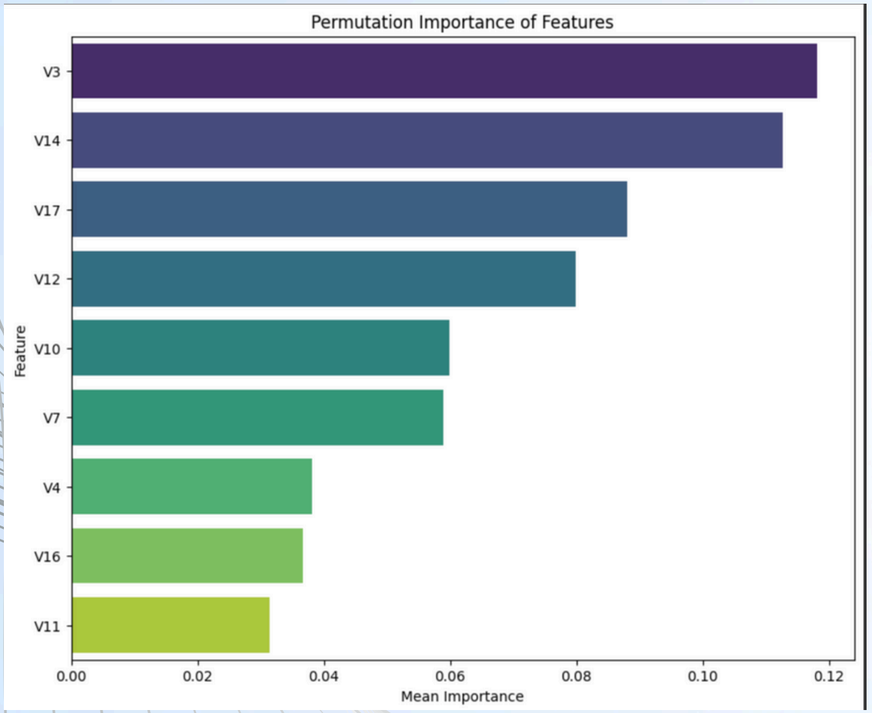
The correlation coefficient between each feature and the target variable ('Class') was calculated. Features with low correlation were identified as having minimal impact on the final prediction and were removed. This allowed us to focus on the most influential features for improved model performance.



On feature Selection



FEAUTURE IMPORTANCE



Permutation importance measures the change in a model's performance when a single feature's values are randomly shuffled.

The idea is simple, if shuffling a feature drops performance significantly, it means the model relied heavily on that feature and hence we cannot drop that feature.

HANDLING DATA IMBALANCE

Models trained on **imbalanced data** may prioritize accuracy on the majority class while neglecting the minority class. This can result in poor performance in detecting fraud.

	Predicted Non-Fraudulent	Predicted Fraudulent
Actual Non-Fraudulent	71,078	200
Actual Fraudulent	50	40

The model fails to detect most fraudulent transactions (low recall for class 1).

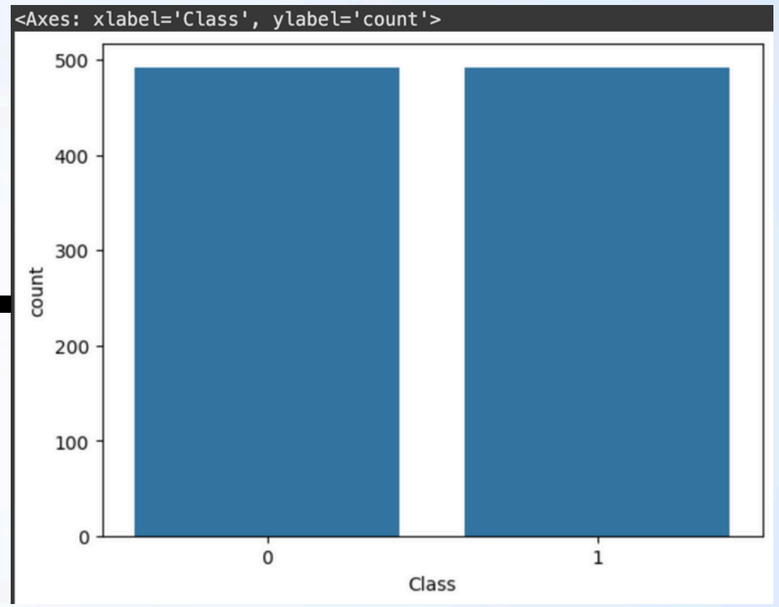
Oversampling techniques artificially inflate the minority class by generating synthetic examples. But, this can lead to overfitting and the introduction of noise, especially in cases where the minority class is already sparsely represented.

	Predicted Non-Fraudulent	Predicted Fraudulent
Actual Non-Fraudulent	70,800	478
Actual Fraudulent	10	80

High false positives due to synthetic examples, leading to overfitting and reduced precision for fraud detection.

Downsampling involves randomly reducing the number of samples in the majority class to balance it with the minority class. This encourages the model to learn from both classes equally, improving its ability to accurately detect fraudulent transactions.

Downsampled Data
(Both classes have equal number of data points)

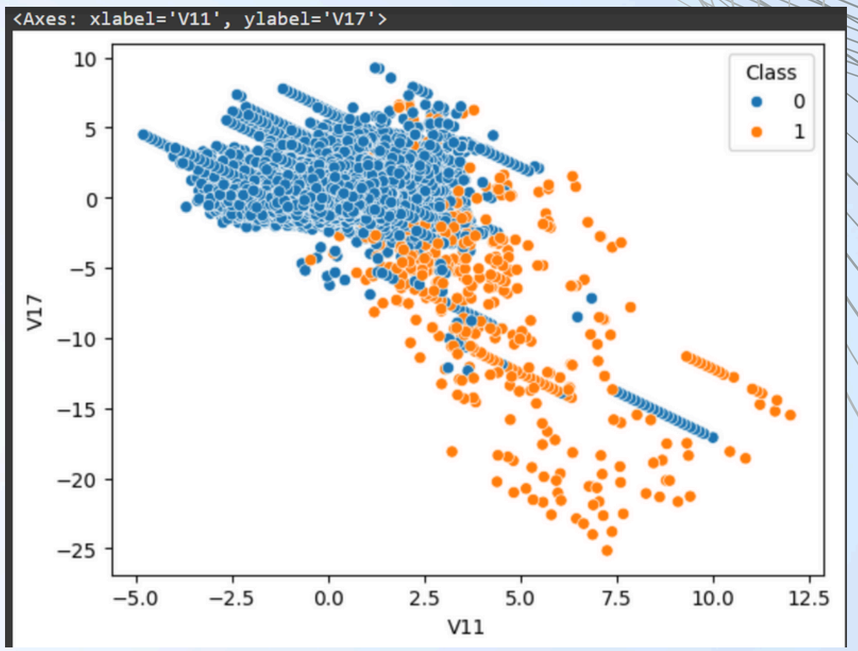


We downsampled the dataset to balance the classes for training. But since real-world data is imbalanced, we **test** the model on the unbalancedto check how it performs in real situations.

HANDLING OUTLIERS

Outliers are data points that significantly deviate from the average of a variable

The **blue dots** represent **normal** transactions, tightly clustered with very few outliers.
In contrast, the **orange dots** represent **fraud transactions**, which do not form a distinct cluster, making outlier detection challenging.
Attempting to manage outliers individually for each variable could result in transforming or **deleting over 70%** of the fraud transactions.



For these reasons, no records will be deleted from fraudulent transactions. Additionally, it's important to consider that fraudulent transactions are rare, making each record valuable.

DATA SPLITTING

Stratified split: This ensures that the imbalance is maintained in both the training and test sets, preventing the risk of ending up with a dataset without fraud transactions, and hence we use this method for splitting the dataset.

We have **downsampled** the data, and also tried to **avoid data leakage**, that is there is no overlap between the final training and test datasets to prevent information from the test set influencing the training process.

QUICK TEST USING LAZYPREDICT

LazyPredict is a Python library that quickly **compares** the **performance** of **multiple ML models** without writing detailed code for each one. It helps identify the best-performing models by providing accuracy, F1 score, and training time—making model selection faster and easier in the initial stages of experimentation.

Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
LogisticRegression	0.97	0.95	0.95	0.99	0.11
LinearSVC	0.97	0.95	0.95	0.98	0.13
SVC	0.99	0.95	0.95	0.99	1.37
SGDClassifier	0.99	0.95	0.95	0.99	0.24
CalibratedClassifierCV	0.98	0.95	0.95	0.99	0.18
PassiveAggressiveClassifier	0.98	0.94	0.94	0.99	0.11
ExtraTreesClassifier	0.97	0.94	0.94	0.98	1.29
KNeighborsClassifier	0.98	0.94	0.94	0.99	11.47
RandomForestClassifier	0.96	0.94	0.94	0.98	1.89
XGBClassifier	0.95	0.93	0.93	0.97	0.26
LGBMClassifier	0.95	0.93	0.93	0.97	0.75
GaussianNB	0.98	0.93	0.93	0.99	0.11
QuadraticDiscriminantAnalysis	0.96	0.93	0.93	0.98	0.13
BaggingClassifier	0.96	0.93	0.93	0.98	0.30
AdaBoostClassifier	0.95	0.93	0.93	0.97	0.87
BernoulliNB	1.00	0.93	0.93	1.00	0.20
LabelSpreading	0.96	0.93	0.93	0.98	2.05
LabelPropagation	0.96	0.93	0.93	0.98	2.01
ExtraTreeClassifier	0.91	0.93	0.93	0.95	0.08
NuSVC	1.00	0.92	0.92	1.00	3.58
DecisionTreeClassifier	0.91	0.91	0.91	0.95	0.08
RidgeClassifierCV	0.99	0.91	0.91	1.00	0.20
RidgeClassifier	0.99	0.90	0.90	1.00	0.08
LinearDiscriminantAnalysis	0.99	0.90	0.90	1.00	0.13
NearestCentroid	1.00	0.90	0.90	1.00	0.14
Perceptron	0.64	0.81	0.81	0.78	0.17
DummyClassifier	1.00	0.50	0.50	1.00	0.05

Here, the best results are given by BernoulliNB, NuSVC and NearestCentroid. However, since detecting normal transactions is not challenging for machine learning models, we will focus on the F1 score for fraud transactions (Class == 1).

HANDLING OUTLIERS

Hence when the three models with highest accuracy are chosen, and F1 scores for Class-1 (fraud) is calculated, we see that NearestCentroid performs the best.

```
F1 Scores for class == 1:
BernoulliNB: 0.0994
NuSVC: 0.5950
NearestCentroid: 0.8144
```

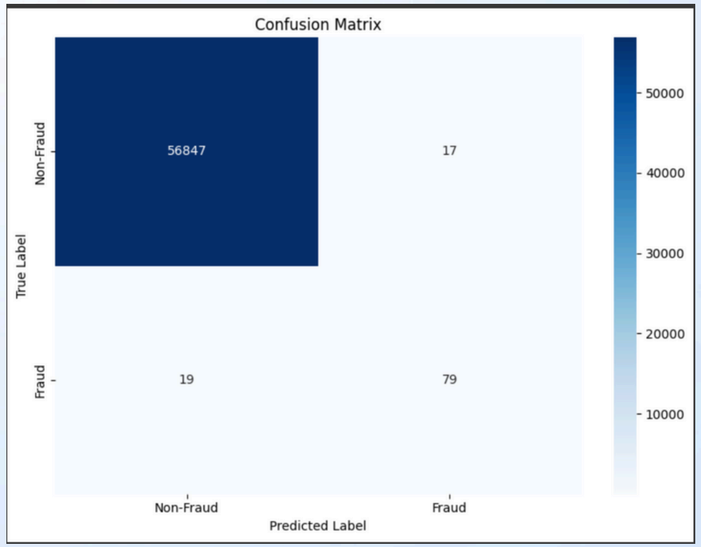
On building the final model on NearestCentroid classifier, we get

```
Classification Report:
              precision    recall  f1-score   support

   Non-Fraud       1.00      1.00      1.00     56864
       Fraud       0.82      0.81      0.81        98

    accuracy                           1.00     56962
   macro avg       0.91      0.90      0.91     56962
weighted avg       1.00      1.00      1.00     56962

Accuracy: 99.94%
```



We can see that the accuracy of our model is high. However, since the dataset is **imbalanced**, **accuracy is not a reliable metric**. Instead, evaluation should focus on F1-Score, Recall, and other metrics that better reflect performance on minority classes.
Hence we integrate the concept of Genetic Algorithms into the code and dataset workflow to enhance the model performance.

Applying Genetic Algorithm:

On the downsampled (balanced) dataset, we perform a train-test split and run the Genetic Algorithm with **Population Size** = 20, **Generations** = 5, **Crossover Probability** = 0.5, and **Mutation Probability** = 0.2 to select the optimal set of features.

Genetic Algorithm is doing the feature selection on its own by **evolving combinations of features** that give the best model performance (based on the fitness function like accuracy or F1 score).

Fitness function is the core of genetic algorithm. It evaluates how good a solution (i.e., an individual in the population) is at solving the problem. It **returns a score called fitness** that helps the algorithm decide which individuals are better, who gets to survive, reproduce or be discarded.

In the setup of the Fitness Function, we used Accuracy and F1 score as the evaluation metric. The Genetic Algorithm then selected the feature subset that maximized accuracy and F1 score, and the model was trained using only those selected features, producing the following performance results.

```
Final Test Accuracy: 0.9594
Recall: 0.9388
F1 Score: 0.9583

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.98      0.96        99
           1       0.98      0.94      0.96        98

    accuracy                           0.96       197
   macro avg       0.96      0.96      0.96       197
weighted avg       0.96      0.96      0.96       197
```

We can see that there is a significant improvement in the F1 score and recall by integrating genetic algorithm from that of the NearestCentroid classifier.

THANK YOU