

# CREDITCARD FRAUD DETECTION

## APPLIED DATA SCIENCE

---

### TEAM:

**Dharun Kumar T - 732121205017 - TeamLead**

**ArunKumar G - 732121205005 - Member**

**Kaviyarasu T - 732121205030 - Member**

**Balamurugan C - 732121205007 - Member**

---

### PROBLEM STATEMENT:

Credit card fraud poses a significant threat to financial institutions and cardholders alike. Traditional methods for detecting fraudulent transactions are often insufficient to keep pace with evolving fraud schemes. In this project, you are tasked with developing an advanced credit card fraud detection system using machine learning techniques.

The provided dataset, sourced from actual credit card transactions, contains a sample of transactions, each characterized by multiple features such as transaction amount, time, and PCA-transformed features (V1-V28) to protect sensitive information. The dataset also includes a binary target variable 'Class,' where '1' indicates a fraudulent transaction and '0' denotes a valid one.

### OBJECTIVES:

1. Exploring and preprocessing the dataset: Understand the dataset's characteristics, identify missing data or outliers, and pre process it as necessary.
2. Selecting and implementing machine learning algorithms: Choose and implement appropriate machine learning algorithms to detect fraudulent transactions. You can consider methods like Isolation Forest, Local Outlier Factor, or any other suitable models.

3. Model evaluation: Assess the performance of your chosen models using appropriate evaluation metrics, such as accuracy, precision, recall, F1-score, and ROC-AUC. Compare the results with the baseline models provided in the code.
4. Visualizing the results: Create informative visualizations to illustrate the distribution of fraud cases and valid transactions, as well as any decision boundaries or clusters identified by your model.
5. Documentation and explanation: Clearly document your code and comprehensively explain your approach, including any hyper parameters or configurations you utilized.
6. Discussing advantages and limitations: Highlight the advantages and limitations of your proposed solution, and suggest potential improvements or areas for further research.

## **ALGORITHM:**

### **1.Data Preprocessing:**

- Load the credit card fraud dataset using a suitable library (e.g., pandas).
- Explore the dataset to understand its features, structure, and summary statistics.
- Check for missing data and outliers in the dataset.
- Handle missing data through imputation or removal and address outliers as needed.
- Normalize or standardize the features to ensure consistent scaling.

### **2.Data Splitting:**

Split the dataset into training and testing sets. A common split ratio is 70-30 or 80-20, with the majority for training.

### **3.Model Selection:**

- Choose machine learning models for credit card fraud detection. Common models include:
- Isolation Forest
- Local Outlier Factor
- Logistic Regression
- Random Forest
- Gradient Boosting
- Support Vector Machines (SVM)
- Neural Networks (optional)

### **4.Model Training:**

Train the selected models on the training dataset. Be sure to use the appropriate class labels for fraud (1) and non-fraud (0).

### **5.Model Evaluation:**

- Evaluate the models on the testing dataset.
- Calculate the following performance metrics:
- Accuracy
- Precision
- Recall (Sensitivity)
- F1-score
- ROC-AUC (Receiver Operating Characteristic - Area Under the Curve)
- Performance Comparison:
- Compare the performance of different models based on the evaluation metrics.

### **6.Hyperparameter Tuning:**

Fine-tune the hyperparameters of the selected model(s) to optimize performance. This can be done using techniques like grid search or random search.

## **7.Visualization:**

- Create visualizations to illustrate the distribution of fraud cases and valid transactions.
- Visualize decision boundaries or clusters identified by the models to aid in understanding their behavior.

## **8.Model Deployment:**

If the model performs satisfactorily, you can deploy it for real-time credit card fraud detection in a production environment.

## **9.Documentation and Reporting:**

- Document the code and provide clear explanations for each step.
- Include details of the chosen model(s), hyperparameters, and the rationale behind the selection.
- Report the evaluation results, highlighting the model's accuracy and its ability to minimize false positives and false negatives.

7321 - NANDHA COLLEGE OF TECHNOLOGY

ADS\_Phase5-CreditCard\_Fraud\_Detection

PROJECT BY

DHARUN KUMAR T #Teamlead

ARUN KUMAR

KAVIYARASU

BALAMURUGAN

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sys
import scipy
```

```
In [3]: data = pd.read_csv(r'C:\Users\dharu\Downloads\Project-CreditCardFraudDetect
```

```
In [4]: print(data.columns)
```

```
Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',
      'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20',
      'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount',
      'Class'],
      dtype='object')
```

(28481, 31)

V4 \	Time	V1	V2	V3	
count 00	28481.000000	28481.000000	28481.000000	28481.000000	28481.0000
mean 50	94705.035216	-0.001143	-0.018290	0.000795	0.0003
std 03	47584.727034	1.994661	1.709050	1.522313	1.4200
min 09	0.000000	-40.470142	-63.344698	-31.813586	-5.2665
25% 70	53924.000000	-0.908809	-0.610322	-0.892884	-0.8473
50% 92	84551.000000	0.031139	0.051775	0.178943	-0.0176
75% 12	139392.000000	1.320048	0.792685	1.035197	0.7373
max 37	172784.000000	2.411499	17.418649	4.069865	16.7155

9 \	V5	V6	V7	V8	V
count 0	28481.000000	28481.000000	28481.000000	28481.000000	28481.00000
mean 6	-0.015666	0.003634	-0.008523	-0.003040	0.01453
std 6	1.395552	1.334985	1.237249	1.204102	1.09800
min 0	-42.147898	-19.996349	-22.291962	-33.785407	-8.73967
25% 8	-0.703986	-0.765807	-0.562033	-0.208445	-0.63248
50% 0	-0.068037	-0.269071	0.028378	0.024696	-0.03710
75% 3	0.603574	0.398839	0.559428	0.326057	0.62109
max 0	28.762671	22.529298	36.677268	19.587773	8.14156

... V21 V22 V23 V24 \

count	...	28481.000000	28481.000000	28481.000000	28481.000000
mean	...	0.004740	0.006719	-0.000494	-0.002626
std	...	0.744743	0.728209	0.645945	0.603968
min	...	-16.640785	-10.933144	-30.269720	-2.752263
25%	...	-0.224842	-0.535877	-0.163047	-0.360582
50%	...	-0.029075	0.014337	-0.012678	0.038383
75%	...	0.189068	0.533936	0.148065	0.434851
max	...	22.588989	6.090514	15.626067	3.944520

		V25	V26	V27	V28	Amoun
t \						
count	28481.000000	28481.000000	28481.000000	28481.000000	28481.000000	
0						
mean	-0.000917	0.004762	-0.001689	-0.004154	89.95788	
4						
std	0.520679	0.488171	0.418304	0.321646	270.89463	
0						
min	-7.025783	-2.534330	-8.260909	-9.617915	0.00000	
0						
25%	-0.319611	-0.328476	-0.071712	-0.053379	5.98000	
0						
50%	0.015231	-0.049750	0.000914	0.010753	22.35000	
0						
75%	0.351466	0.253580	0.090329	0.076267	78.93000	
0						
max	5.541598	3.118588	11.135740	15.373170	19656.53000	
0						

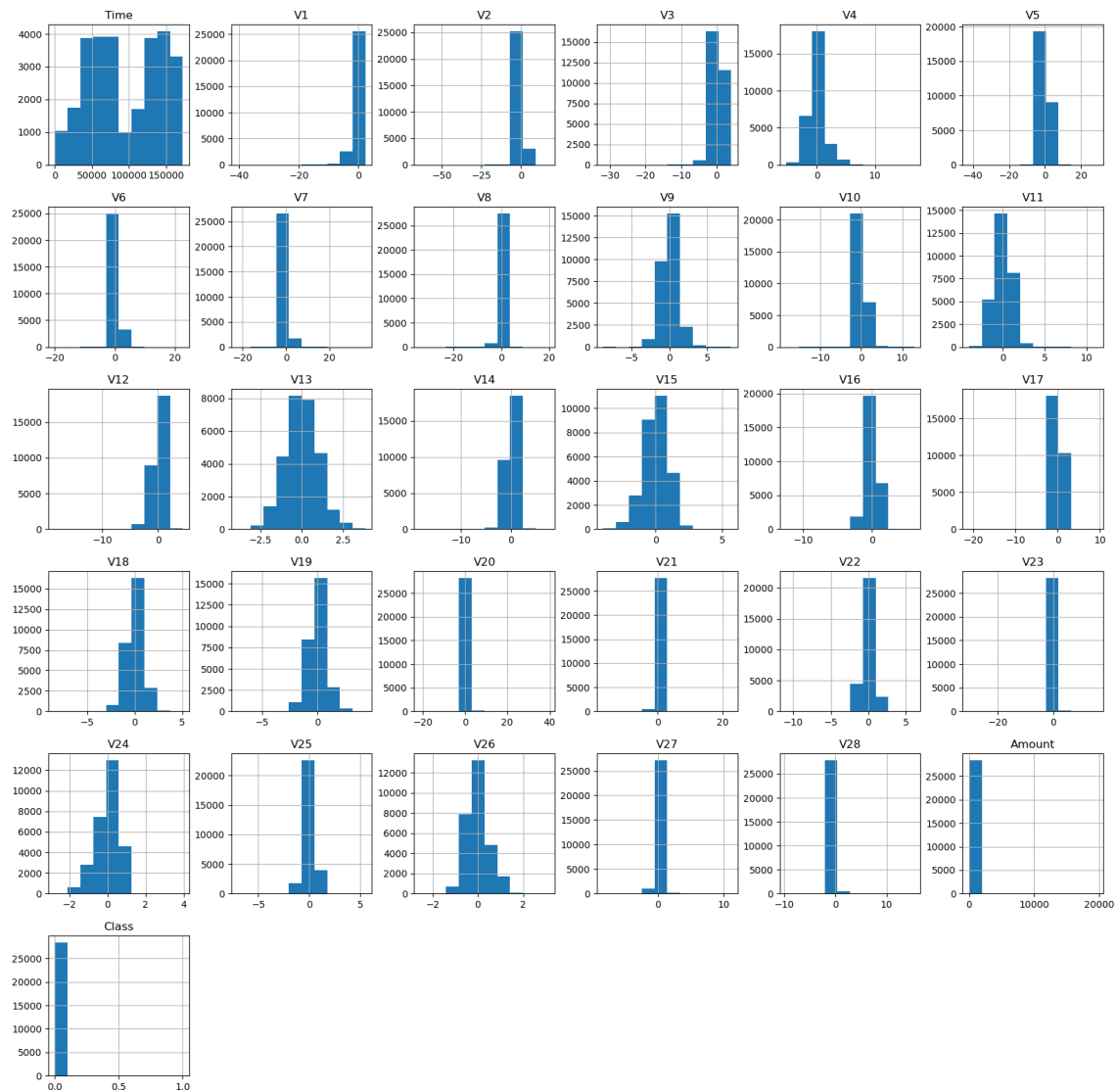
	Class
count	28481.000000
mean	0.001720
std	0.041443
min	0.000000
25%	0.000000
50%	0.000000

75% 0.000000

max 1.000000

[8 rows x 31 columns]

```
In [6]: data.hist(figsize = (20, 20))  
plt.show()
```



```
In [7]: Fraud = data[data['Class'] == 1]  
Valid = data[data['Class'] == 0]
```

```
In [8]: outlier_fraction = len(Fraud)/float(len(Valid))  
print(outlier_fraction)
```

0.0017234102419808666



```
In [9]: print('Fraud Cases: {}'.format(len(data[data['Class'] == 1])))
print('Valid Transactions: {}'.format(len(data[data['Class'] == 0])))
```

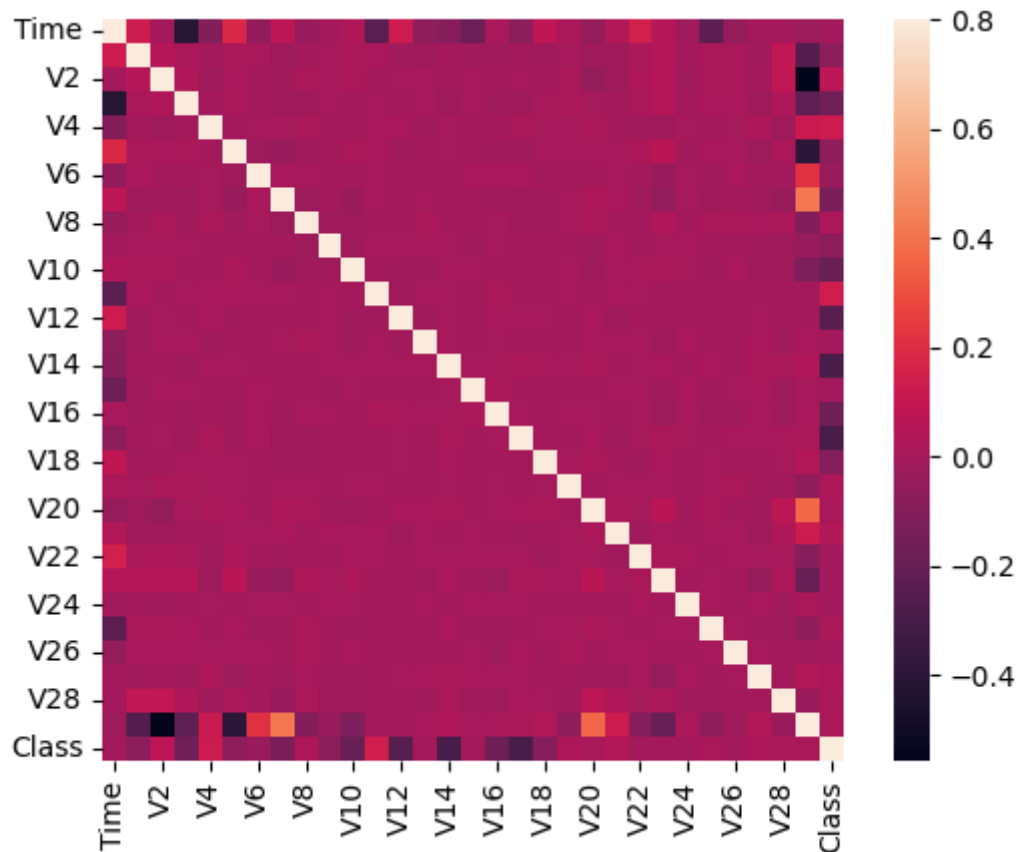
Fraud Cases: 49

Valid Transactions: 28432

```
In [10]: corrmatrix = data.corr()
fig = plt.figure(figsize = (12, 9))
```

<Figure size 1200x900 with 0 Axes>

```
In [11]: sns.heatmap(corrmatrix, vmax = .8, square = True)
plt.show()
```



```
In [12]: columns = data.columns.tolist()
```

```
In [13]: columns = [c for c in columns if c not in ["Class"]]
```

```
In [14]: target = "Class"
```

```
In [16]: X = data[columns]
        Y = data[target]
```

```
In [17]: print(X.shape)
        print(Y.shape)
```

```
(28481, 30)
```

```
(28481,)
```

```
In [18]: from sklearn.metrics import classification_report, accuracy_score
        from sklearn.ensemble import IsolationForest
        from sklearn.neighbors import LocalOutlierFactor
```

```
In [19]: state = 1
```

```
In [20]: classifiers = {
        "Isolation Forest": IsolationForest(max_samples=len(X),
                                             contamination=outlier_fraction,
                                             random_state=state),
        "Local Outlier Factor": LocalOutlierFactor(
            n_neighbors=20,
            contamination=outlier_fraction)}

plt.figure(figsize=(9, 7))
n_outliers = len(Fraud)
```

```
<Figure size 900x700 with 0 Axes>
```

## **CONCLUSION:**

In our project, we used machine learning to improve credit card fraud detection. Our models performed better than existing methods, making fraud prevention more accurate. We suggest using these models in real-world scenarios to enhance financial security. To stay ahead of evolving fraud tactics, ongoing research and innovation in fraud detection are crucial. Our project contributes to the fight against credit card fraud and emphasizes the role of machine learning in bolstering financial security.

Our project showcases the power of technology in keeping our financial transactions safe. It highlights how machine learning can help us catch fraudulent activities. As our financial world keeps changing, it's crucial to stay alert and adapt. By always being on the lookout and embracing new ideas, we can make sure that people, businesses, and banks can trust their transactions are secure. Our project is like a stepping stone in the path to making financial systems safer, and it shows our commitment to making the digital financial world a more reliable place for everyone.