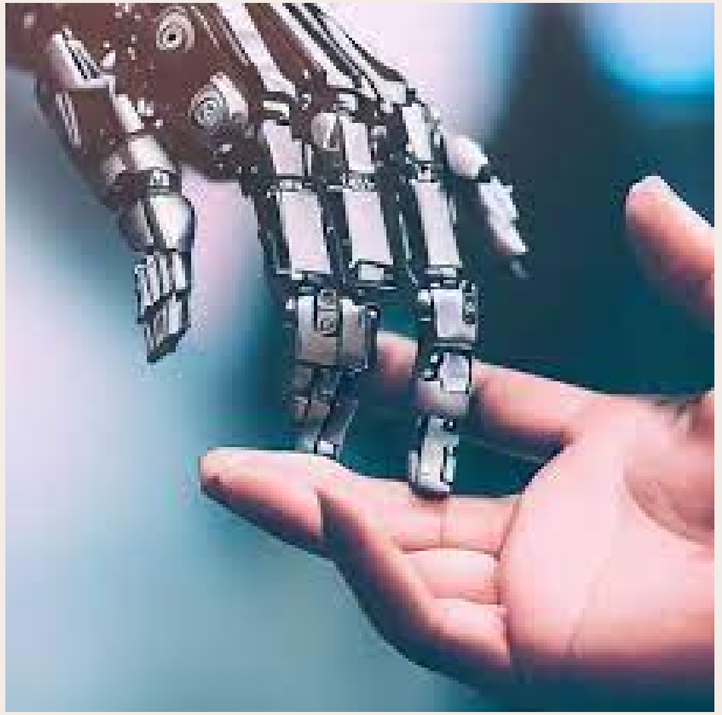
Artificial Intelligence Workshop Project



TOPIC: CREDIT CARD FRAUD DETECTION

Team Members

Amgoth Sai Teja

Anushka Singh

Lives Kumar Singh

Rupali Das

Table of Content

Introduction

- Artificial intelligence and machine learning have provided solutions to complex problems and changed the way we approach data analysis and decision-making processes.

- The proliferation of credit card transactions offers consumers and businesses unparalleled convenience in an increasingly connected digital economy. However, as convenience increases, so does the threat of fraud.

Objective

This study summarizes a carefully planned AI/ML project that addresses the critical problem of credit card fraud detection. By using the dataset from Kaggle we have successfully formed a project using Logistic regression model to detect fraudulent transaction through credit card.

Problem Statement

- The problem statement for a credit card fraud detection project in AI/ML typically involves developing a system that can accurately identify and prevent fraudulent transactions in real-time.

- The goal is to create a robust model that can effectively distinguish between legitimate and fraudulent activities to minimize financial losses for both credit card users and financial institutions.

Motivation

- With a surge in fraudulent activities, the existing rule-based systems fall short in combatting sophisticated fraud tactics.

- AI/ML empowers adaptive, real-time analysis, swiftly detecting anomalies and potentially fraudulent patterns in massive transaction datasets.

Logistic Regression

- We used the Logistic Regression model to implement this credit card fraud detection project.
- Logistic Regression is a fundamental statistical method used for binary classification tasks.
- Operating on the logistic function (sigmoid function) principle, Logistic Regression maps inputs to an output range between 0 and 1, representing probabilities.
- It models the relationship between the independent variables (features) and the probability of a certain class or outcome occurring.

Advantages Of Logistic Regression

- **Interpretability:** Provides clear and interpretable results, allowing the understanding of how each input feature contributes to the prediction.

- **Efficiency and Speed:** It's computationally efficient and less complex than other more advanced algorithms.

- **Ease of Implementation:** Relatively easy to implement and doesn't require extensive computational resources or sophisticated tuning.

Logistic Regression Process

1. **Data Preprocessing:** Handling missing values, encoding categorical variables, and scaling features for better model performance.

2. **Model Training:** Optimizing the model's coefficients by iteratively updating them based on the error between predictions and actual outcomes.

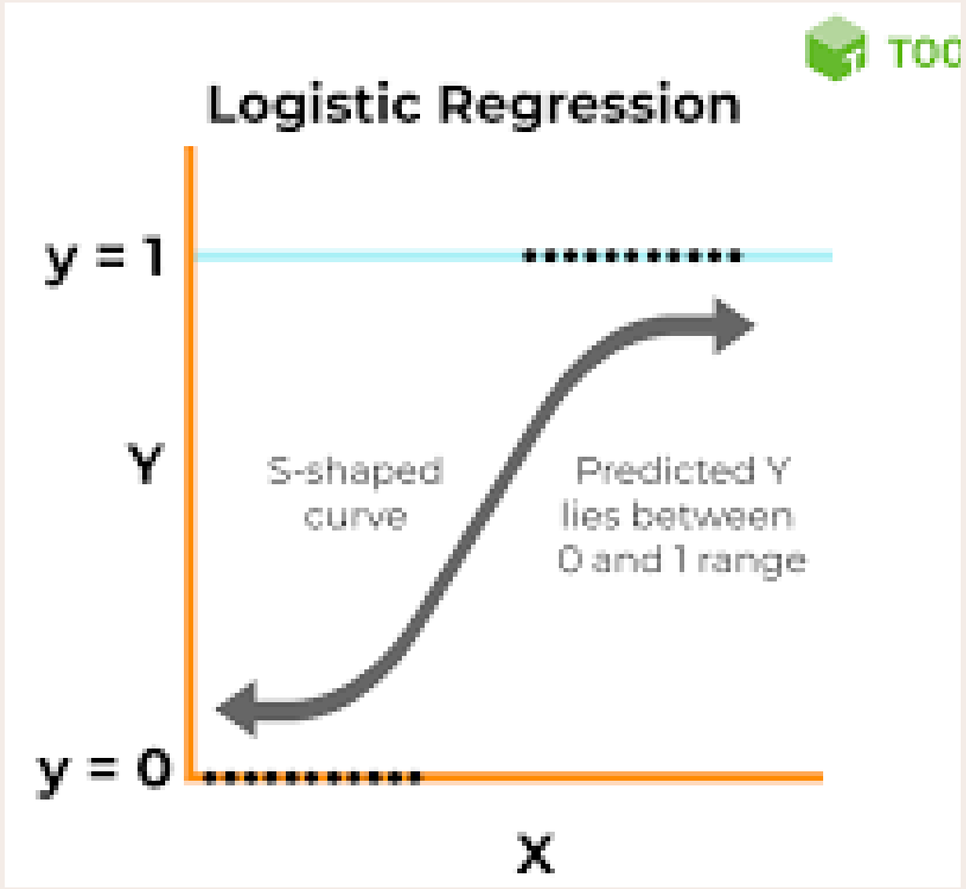
3. **Regularization:** Regularization techniques like L1 (Lasso) and L2 (Ridge) regularisation are often applied to prevent overfitting.

4. **Model Evaluation:** Using various metrics and techniques, like confusion matrices, precision-recall curves, or ROC curves, to assess the model's performance.

Dataset Description

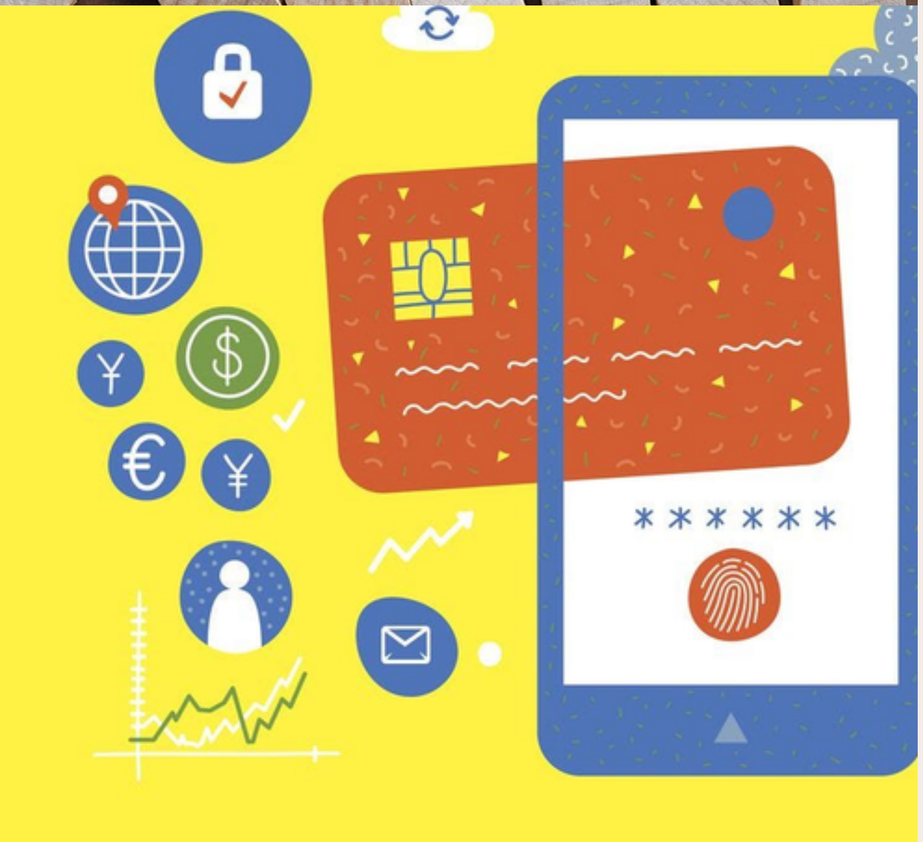
- This dataset contains credit card transactions made by European cardholders in the year 2023. It comprises over 550,000 records, and the data has been anonymized to protect the cardholders' identities.

- The primary objective of this dataset is to facilitate the development of fraud detection algorithms and models to identify potentially fraudulent transactions.



Key variables In Data

- **id:** Unique identifier for each transaction
- **V1-V28:** Anonymized features representing various transaction attributes (e.g., time, location, etc.)
- **Amount:** The transaction amount
- **Class:** Binary label indicating whether the transaction is fraudulent (1) or not (0)



```
credit_card_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 568630 entries, 0 to 568629
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   id      568630 non-null  int64  
 1   V1      568630 non-null  float64
 2   V2      568630 non-null  float64
 3   V3      568630 non-null  float64
 4   V4      568630 non-null  float64
 5   V5      568630 non-null  float64
 6   V6      568630 non-null  float64
 7   V7      568630 non-null  float64
 8   V8      568630 non-null  float64
 9   V9      568630 non-null  float64
 10  V10     568630 non-null  float64
 11  V11     568630 non-null  float64
 12  V12     568630 non-null  float64
 13  V13     568630 non-null  float64
 14  V14     568630 non-null  float64
 15  V15     568630 non-null  float64
 16  V16     568630 non-null  float64
 17  V17     568630 non-null  float64
 18  V18     568630 non-null  float64
 19  V19     568630 non-null  float64
 20  V20     568630 non-null  float64
 21  V21     568630 non-null  float64
 22  V22     568630 non-null  float64
 23  V23     568630 non-null  float64
 24  V24     568630 non-null  float64
 25  V25     568630 non-null  float64
 26  V26     568630 non-null  float64
 27  V27     568630 non-null  float64
 28  V28     568630 non-null  float64
 29  Amount  568630 non-null  float64
 30  Class   568630 non-null  int64  
dtypes: float64(29), int64(2)
memory usage: 134.5 MB
```

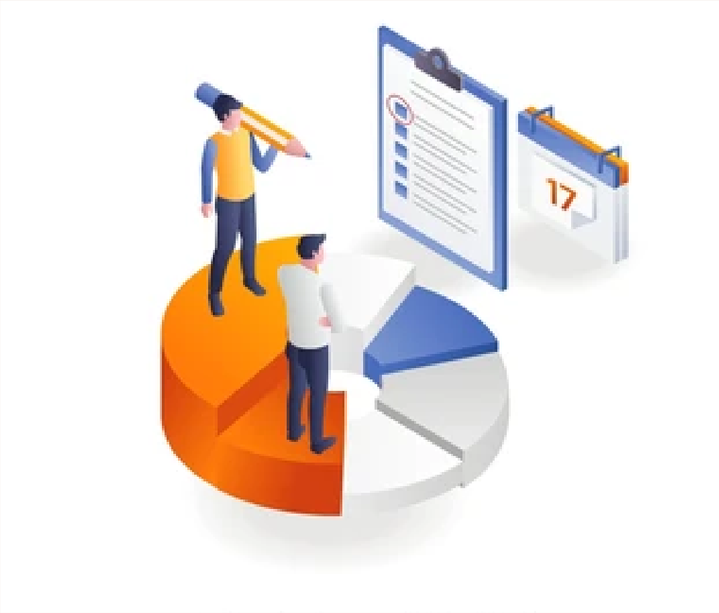
Experiment Description

- We train the data with three models: Logistic Regression, Decision Tree and Random Forest.

- After that, the result of the models is compared based on the accuracy obtained on training and test sets.

Results and Discussion

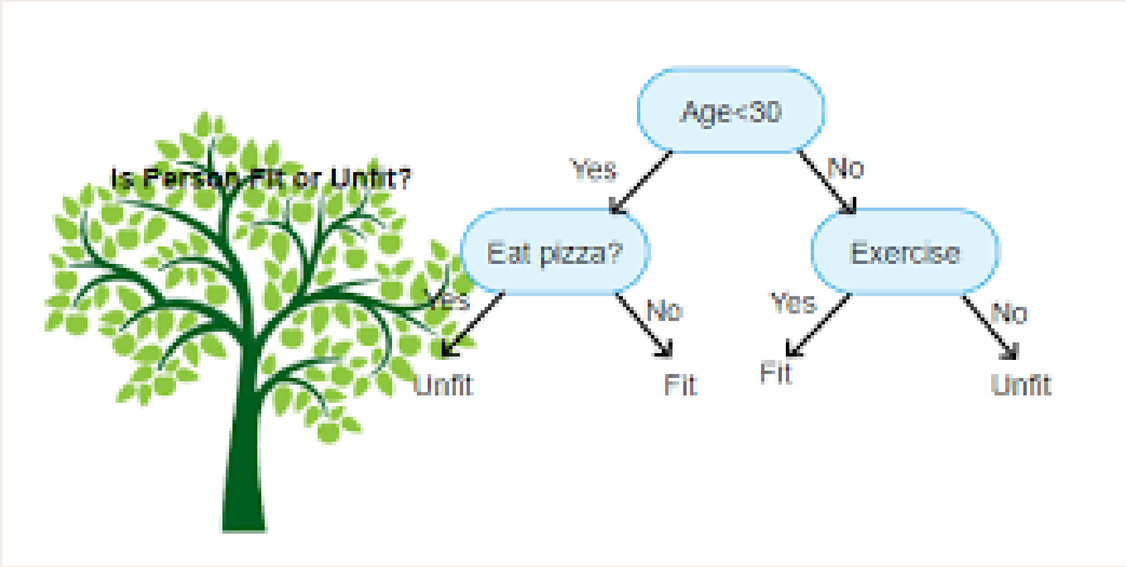
A) Logistic Regression:




```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
model = LogisticRegression()
model.fit(X_train, Y_train)
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print('Accuracy on Training data : ', training_data_accuracy)
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Accuracy score on Test Data : ', test_data_accuracy)
```

```
Accuracy on Training data :  0.9980425289121991
Accuracy score on Test Data :  0.9982971103542713
```

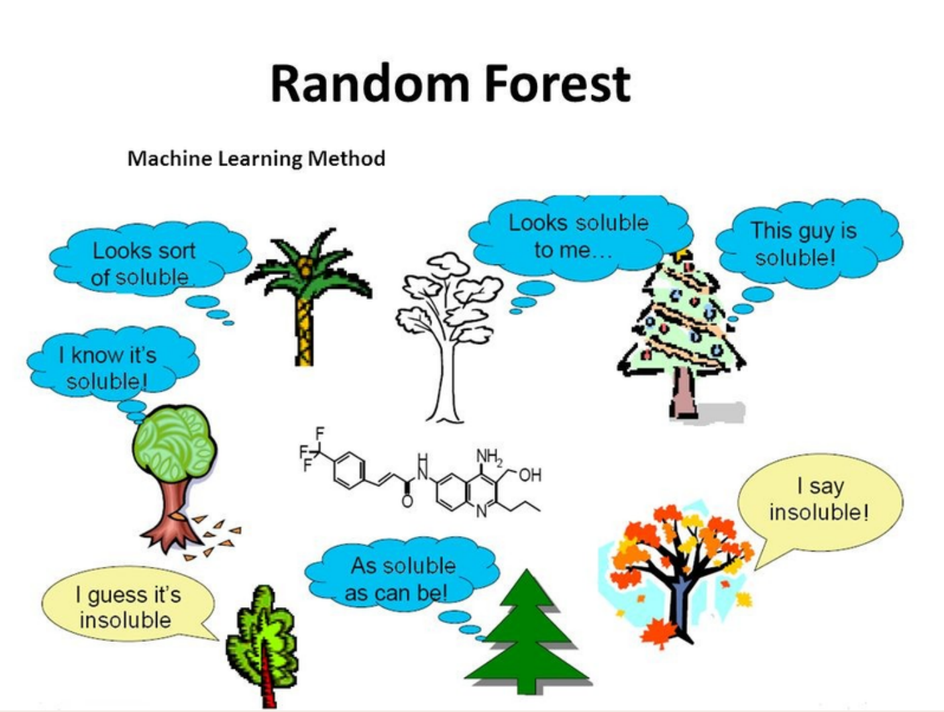
B) Decision Tree




```
from sklearn.tree import DecisionTreeClassifier

dt_model = DecisionTreeClassifier(random_state=2)
dt_model.fit(X_train, Y_train)
dt_train_predictions = dt_model.predict(X_train)
dt_training_accuracy = accuracy_score(dt_train_predictions, Y_train)
dt_test_predictions = dt_model.predict(X_test)
dt_test_accuracy = accuracy_score(dt_test_predictions, Y_test)

print('Decision Tree - Accuracy on Training Data:', dt_training_accuracy)
print('Decision Tree - Accuracy on Test Data:', dt_test_accuracy)
```

```
Decision Tree - Accuracy on Training Data: 1.0
Decision Tree - Accuracy on Test Data: 0.9996488887328394
```

C) Random Forest



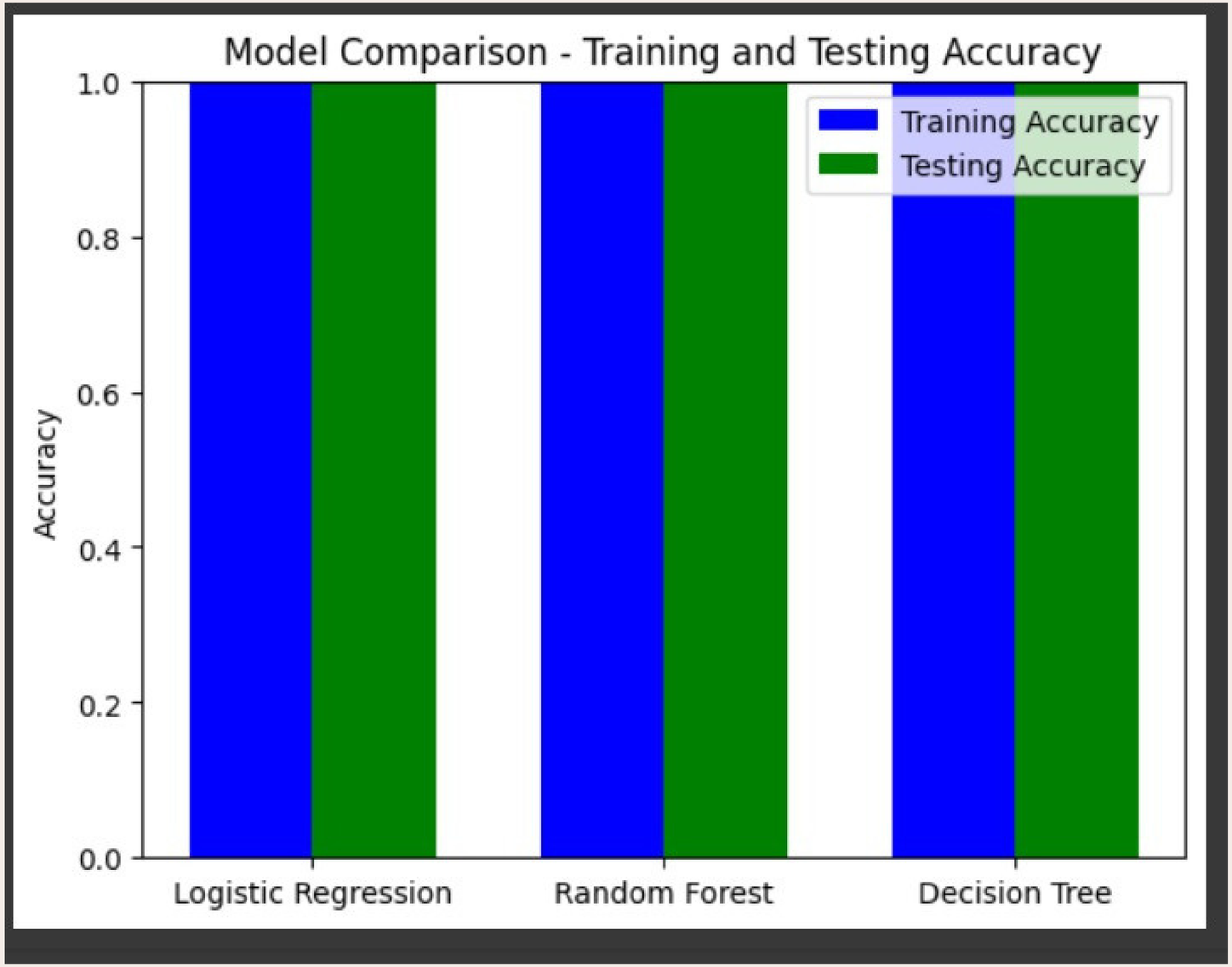

```python
from sklearn.ensemble import RandomForestClassifier

rf_model = RandomForestClassifier(random_state=2)
rf_model.fit(X_train, Y_train)
rf_train_predictions = rf_model.predict(X_train)
rf_training_accuracy = accuracy_score(rf_train_predictions, Y_train)
rf_test_predictions = rf_model.predict(X_test)
rf_test_accuracy = accuracy_score(rf_test_predictions, Y_test)

print('Random Forest - Accuracy on Training Data:', rf_training_accuracy)
print('Random Forest - Accuracy on Test Data:', rf_test_accuracy)
```

```
Random Forest - Accuracy on Training Data: 1.0
Random Forest - Accuracy on Test Data: 0.9998946666198518
```

Accuracy of The models used

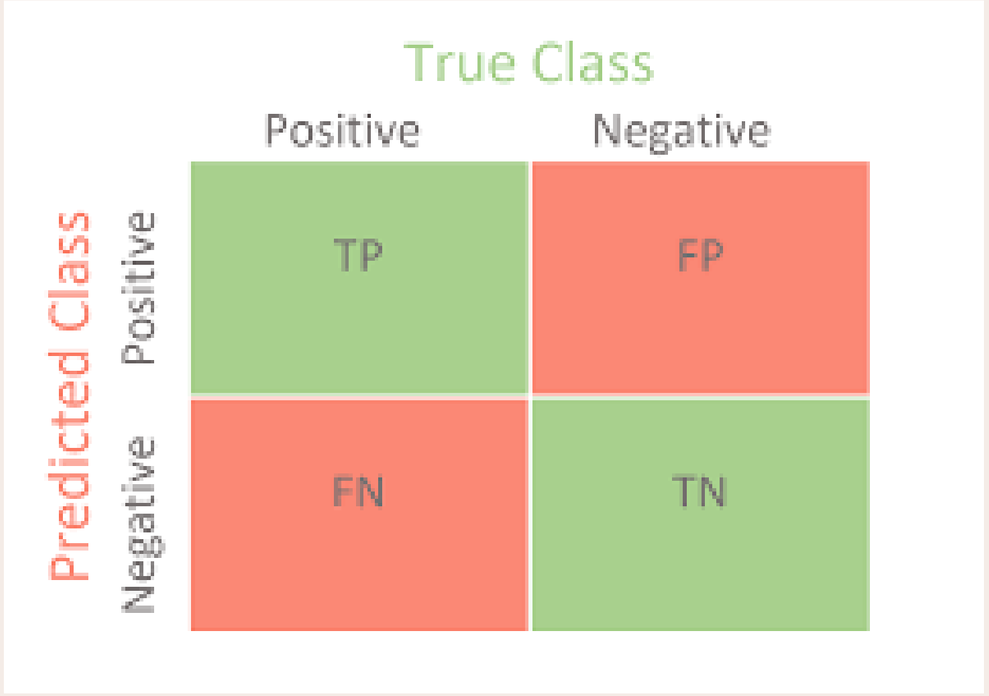
```
models = ['Logistic Regression', 'Random Forest', 'Decision Tree']
train_accuracies = [lr_training_data_accuracy, rf_training_accuracy,dt_training_accuracy]
test_accuracies = [lr_test_data_accuracy, rf_test_accuracy, dt_test_accuracy]
x = np.arange(len(models))
width = 0.35
fig, ax = plt.subplots()
rects1 = ax.bar(x - width/2, train_accuracies, width, label='Training Accuracy', color='b')
rects2 = ax.bar(x + width/2, test_accuracies, width, label='Testing Accuracy', color='g')
ax.set_ylabel('Accuracy')
ax.set_title('Model Comparison - Training and Testing Accuracy')
ax.set_xticks(x)
ax.set_xticklabels(models)
ax.legend()
plt.ylim(0.0, 1.0)
```

Accuracy of The models used

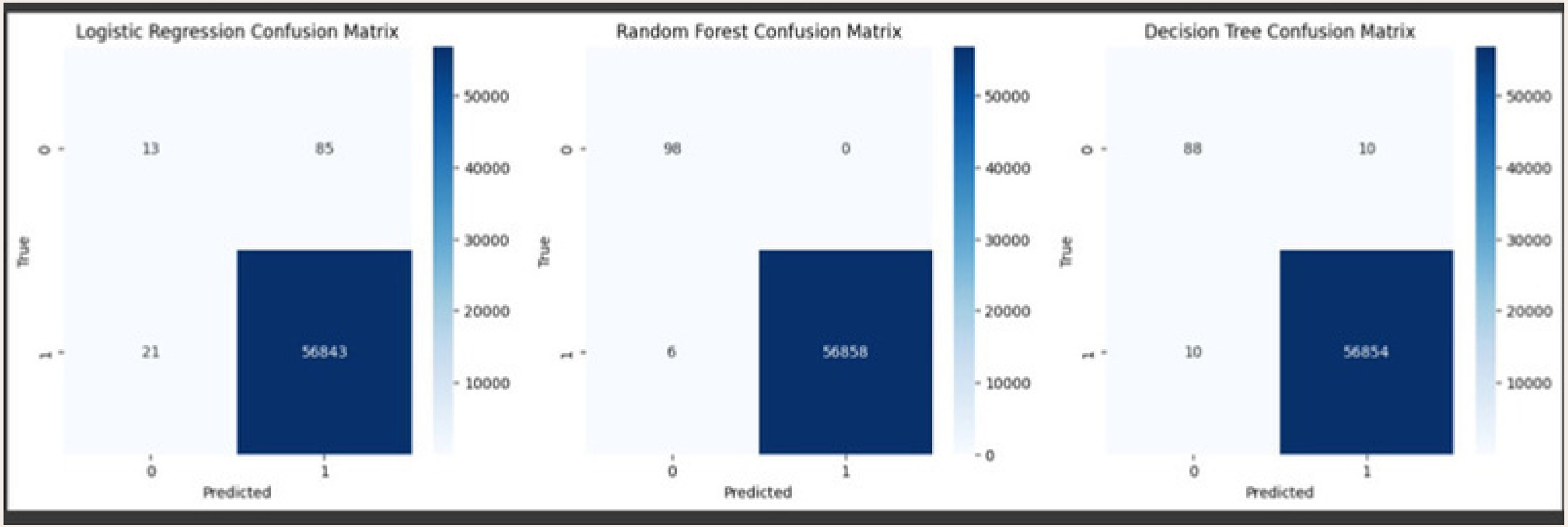


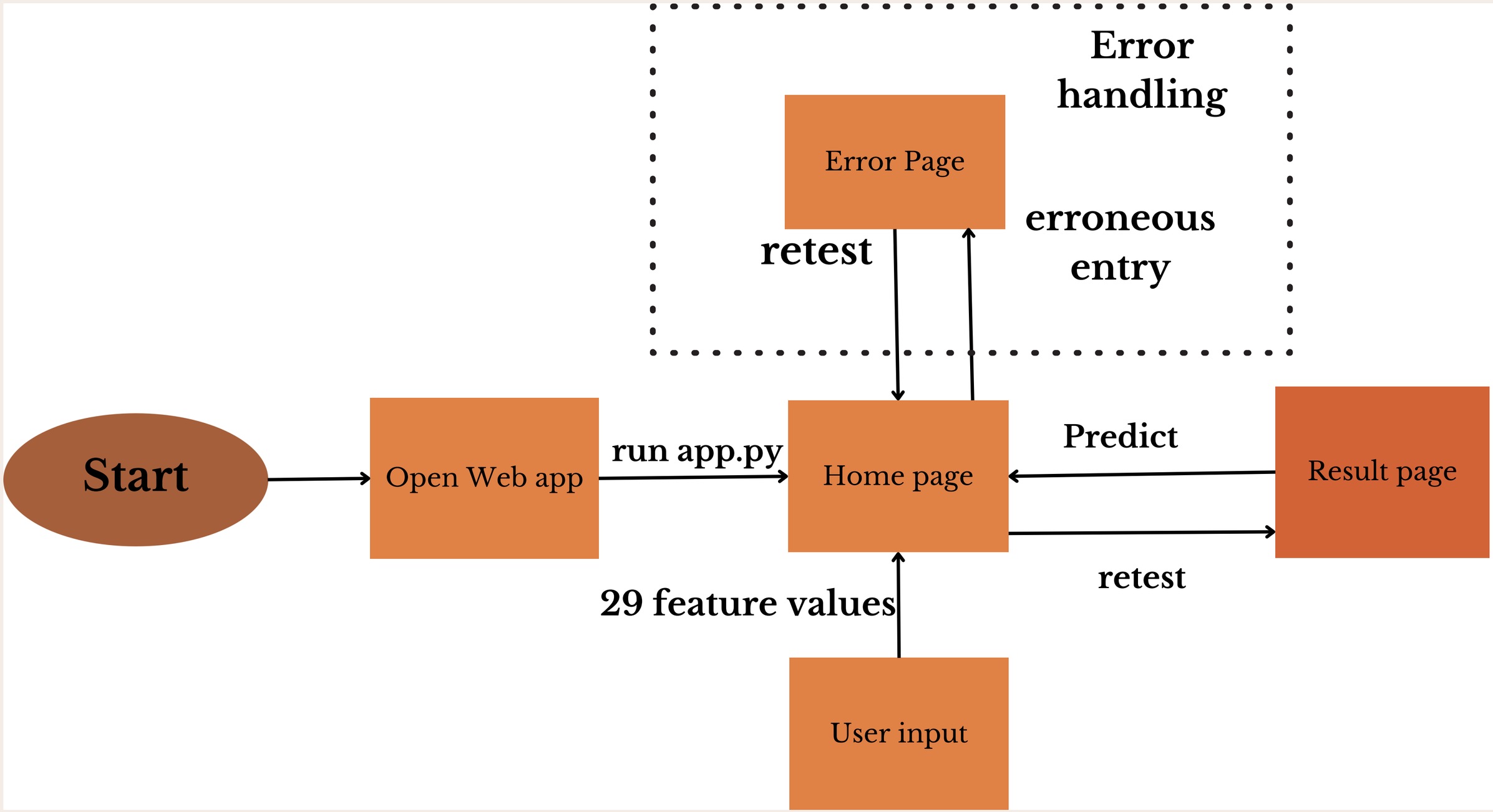
```python
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

lr_confusion_matrix = confusion_matrix(Y_test, X_test_prediction)

rf_confusion_matrix = confusion_matrix(Y_test, rf_test_predictions)

dt_confusion_matrix = confusion_matrix(Y_test, dt_test_predictions)

plt.figure(figsize=(18, 5))

plt.subplot(131)
sns.heatmap(lr_confusion_matrix, annot=True, cmap="Blues", fmt="d")
plt.title("Logistic Regression Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("True")

plt.subplot(132)
sns.heatmap(rf_confusion_matrix, annot=True, cmap="Blues", fmt="d")
plt.title("Random Forest Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("True")

plt.subplot(133)
sns.heatmap(dt_confusion_matrix, annot=True, cmap="Blues", fmt="d")
plt.title("Decision Tree Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("True")

plt.show()
```

Confusion Matrix



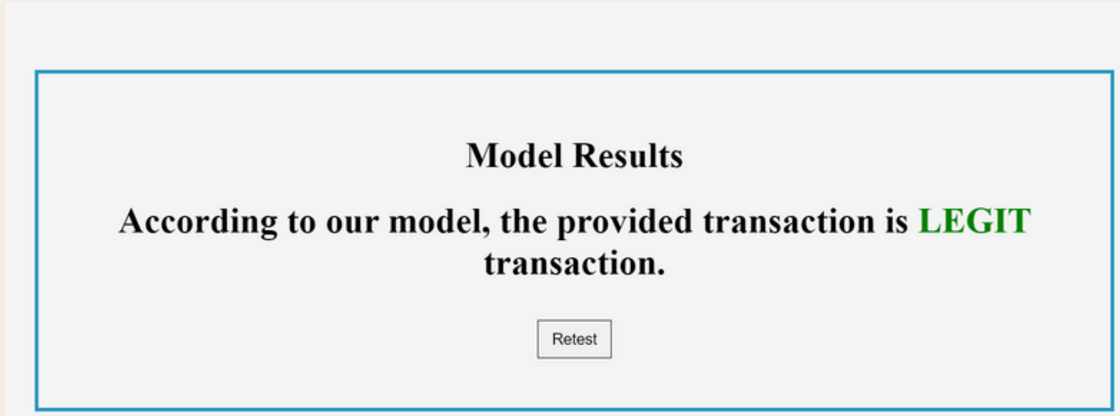
Output As Confusion Matrix

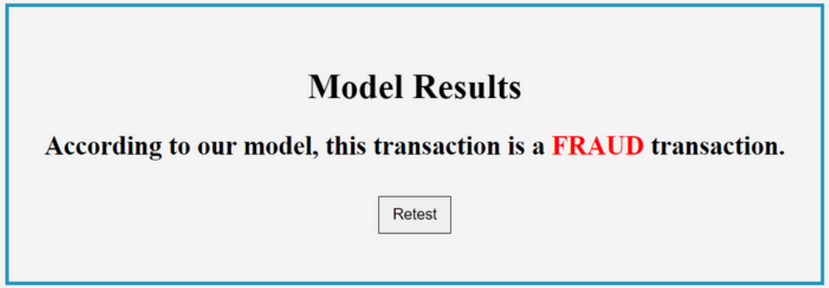


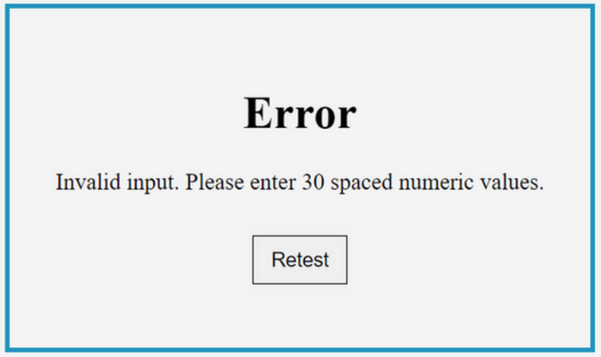


Credit Card Fraud Detection Web app

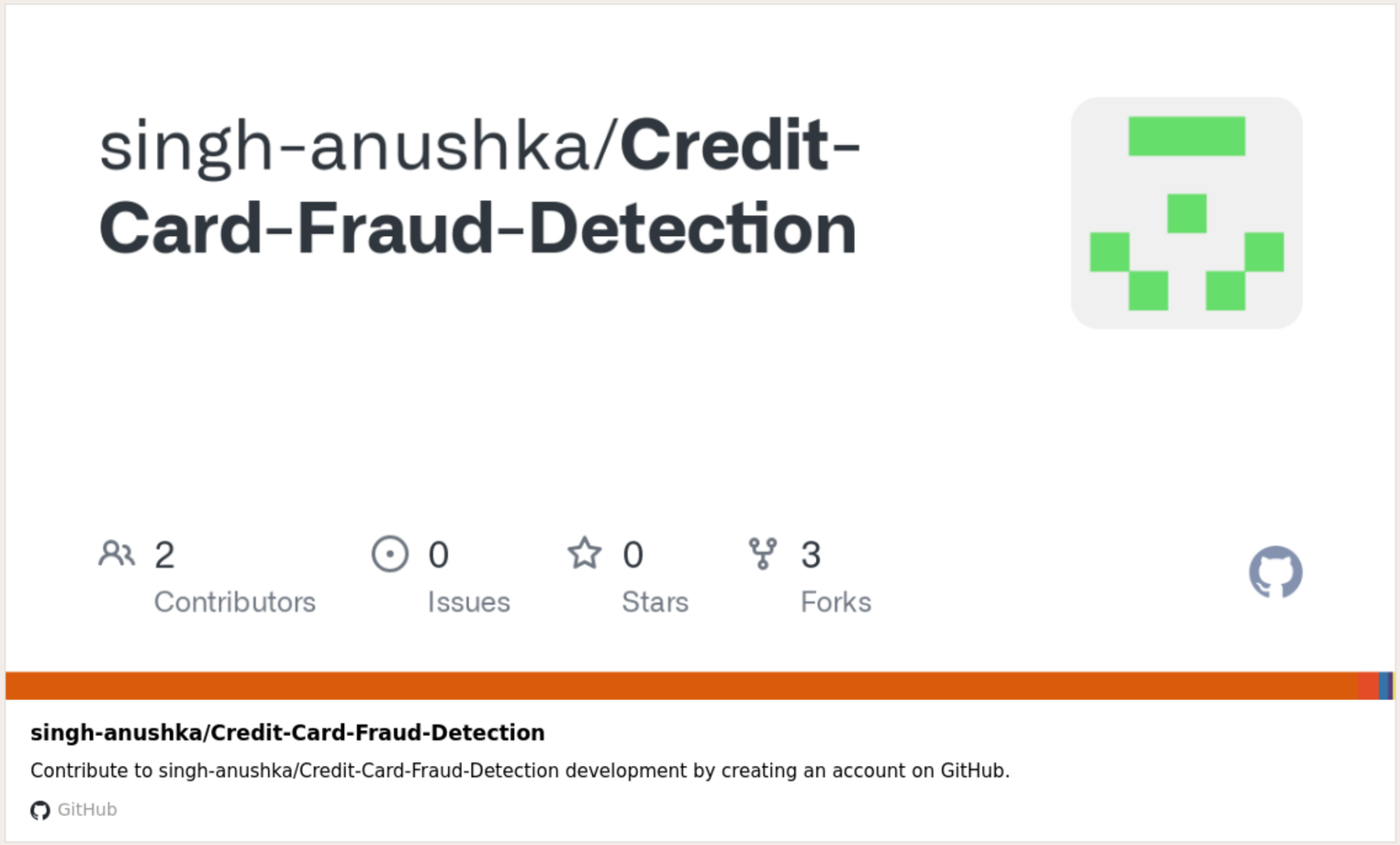
Sneak Peak to our App Interface







GitHub Repository



Limitations

- **Assumption of Linearity:** Logistic regression assumes that the relationship between the features and the log-odds of the response variable is linear. This assumption may not hold in complex fraud scenarios, where fraudulent activities can be highly nonlinear and involve intricate patterns.

- **Imbalanced Data**: Logistic regression can struggle with imbalanced datasets, with a large number of legitimate transactions and a small number of fraudulent ones as it tends to favor the majority class and may have difficulty identifying the minority class (fraudulent transactions).

Limitations

- **Limited to Binary Classification**: Logistic regression is a binary classification model that can distinguish between two classes- either true or false. So, More complex scenarios, such as detecting different types of fraud or identifying multiple classes of fraudulent activity, may require more advanced models.

- **Limited Feature Engineering**: While logistic regression is effective with carefully selected features, it may not perform well if complex, high-dimensional feature engineering is required to capture intricate fraud patterns.

Conclusion

- In conclusion, the logistic regression model for credit card fraud detection is a valuable tool for financial institutions seeking to protect their customers from fraudulent transactions.

- However, it's important to complement logistic regression with other advanced techniques and strategies to enhance detection rates and reduce false positives.

- Regular model retraining and monitoring are essential to maintain the model's effectiveness in an ever-changing landscape of fraudulent activities.

References

1.https://jovian.com/biraj/deploying-a-machine-learning-model

2.https://www.kaggle.com/datasets/kartik2112/fraud-detection

3.https://css-tricks.com/snippets/css/a-guide-to-flexbox/

4.https://www.youtube.com/watch?v=NCgjcHLFNDg

Thank You