Overview

One of the most popular financial tools for online payments and transactions is the credit card, which gives consumers an easy way to handle their money. However, there is a risk associated with using credit cards, especially when it comes to credit card fraud, which is when someone else's credit card or credit card information is used without authorisation in order to make transactions or take money out of the account.

It is crucial that credit card firms are able to detect fraudulent credit card transactions because of the possibility of unauthorised transactions and the expenses that come with them. The use of credit cards has increased with the introduction of digital transactions, which has increased the likelihood of fraudulent behaviour. Financial institutions have suffered large financial losses as a result of this trend, underscoring the significance of differentiating between fraudulent and non-fraudulent activities.

Given these factors, it is crucial to create and implement efficient systems for analysing and spotting fraudulent transactions. By reducing the risks of credit card fraud, these techniques will help credit card firms maintain the integrity of online financial transactions. Our goal in this project is to use Machine

Learning Ensemble Algorithms to create a classification model that can determine whether a credit card transaction is fraudulent or not.

INDEX

CONTENTS

Introduction

A. PROBLEM STATEMENT:

Credit card transactions performed by European cardholders in September 2013 make up the dataset. Only 0.172% of all transactions are fraudulent, making the sample extremely unbalanced with 492 frauds out of 284,807 transactions. Our objective is to create a classification model that can reliably discern between transactions that are fraudulent and those that are real.

B. OBJECTIVE:

The project's goal is to create a machine learning model that can reliably identify fraudulent credit card transactions. The goal is to develop a strong fraud detection system that can detect fraudulent activity with high recall and precision by utilising methods including feature engineering, data balance, exploratory data analysis, and model training. By successfully identifying and stopping fraudulent transactions, the ultimate objective is to improve credit card users' financial security and reduce losses for credit card firms.

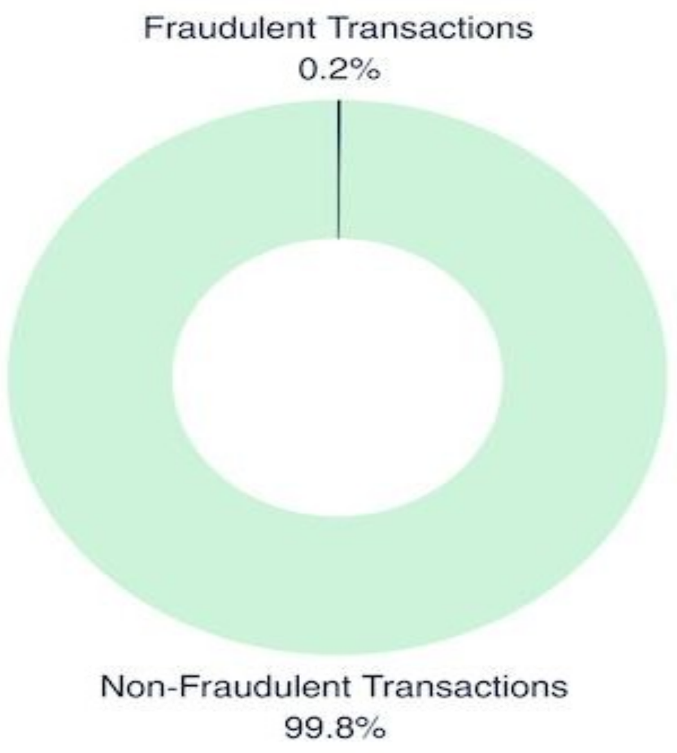
METHODOLOGY

Exploratory Data Analysis (EDA)

To comprehend the underlying patterns and abnormalities in the dataset, exploratory data analysis is essential. We evaluate the quality of the data, deal with outliers and missing values, and make sure date columns have the correct datatype. Finding connections and patterns that guide our next actions is made possible in large part by visualisations.

Dealing with Imbalanced Data

We use methods like oversampling, under-sampling, or synthetic data generation techniques like SMOTE to build a balanced dataset that is suitable for model training because of the dataset's extreme imbalance.



Feature Engineering

To improve model performance, feature engineering entails developing new features and altering preexisting ones. In order to extract valuable information from the dataset, this step is essential. To improve the information available for classification, we generate new features like "Transaction_hour" and "Normalized_amount" from the available data.

Model Selection

We assess several categorisation models, including as Random Forest, Gradient Boosting Machines, Decision Trees, and Logistic Regression, that are appropriate for binary prediction problems. The model was chosen based on its interpretability, performance indicators, and capacity to manage unbalanced data. Because Random Forest is strong against overfitting and can effectively handle imbalanced data, we have chosen it.

Model Training and Evaluation

After dividing the dataset into training and test sets, we used the former to train the Random Forest model. GridSearchCV is used for hyperparameter optimisation in order to maximise model performance. Assessing the model on the test set guarantees its capacity for generalisation and detects any possible problems, including overfitting.

Model Deployment

Following training and validation, the model is put into use in a production setting to detect fraudulent transactions in real time. Real-time inference is made dependable and scalable by deploying machine learning models on AWS SageMaker. The heavy work is taken care of by SageMaker's managed infrastructure, giving you more time to concentrate on providing value to your users.

Results

The model performs exceptionally well, attaining a 99.69% accuracy rate. This suggests that transactions, both fraudulent and lawful, are classified with high accuracy. The model's near-perfect precision, recall, and F1-score show that it can accurately identify both positive and negative situations.

Future Work

Even while our current model shows encouraging results, it can yet be improved. In order to improve decision-making, future work could concentrate on investigating more sophisticated anomaly detection methods, adding new characteristics, and enhancing model interpretability.

Source Code

```
pip install imbalanced-learn

# Importing necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split, cross_val_score,
GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.feature_selection import SelectKBest, f_classif, SelectFromModel
from sklearn.decomposition import PCA

import joblib
import warnings
warnings.filterwarnings('ignore')

#load dataset
data=pd.read_csv("creditcard.csv")

print(data.shape)
data.head()

# Data Exploration and Analysis
# Check for missing values3322
missing_values = data.isnull().sum()
print("Missing Values:\n", missing_values)

# Data Preprocessing
data['Time'] = pd.to_datetime(data['Time']) # Convert 'Time' column to datetime
datatype

# Define features (X) and target variable (y)
```

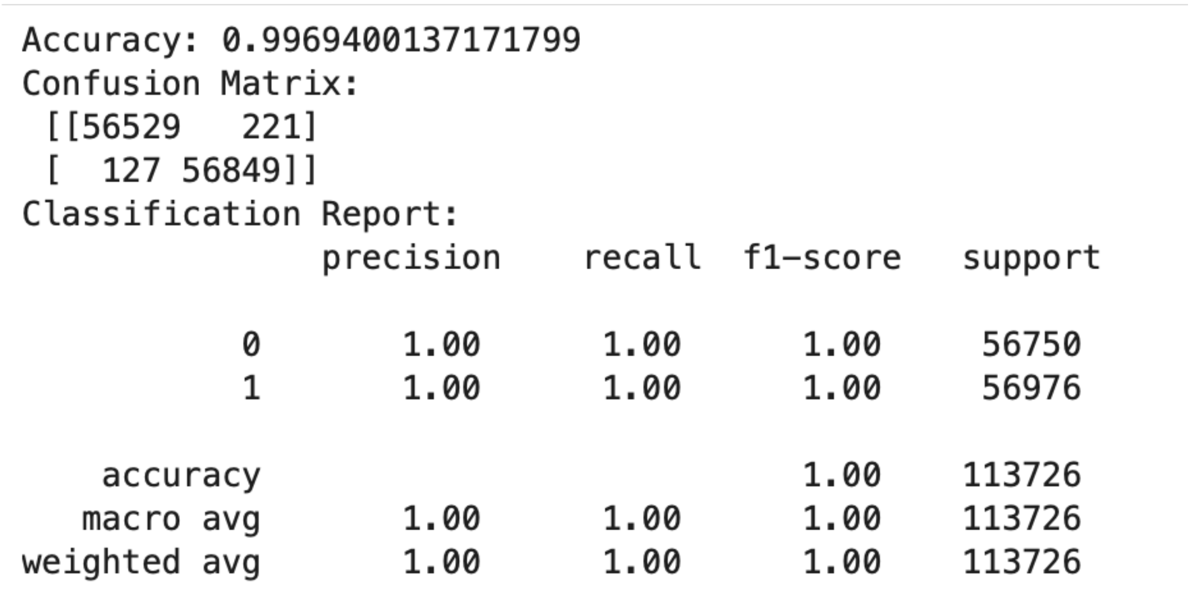
```
X = data.drop(['Class', 'Time'], axis=1) # Features
y = data['Class'] # Target variable

# Remove constant features
data = data.loc[:, data.apply(pd.Series.nunique) != 1]
# Visualize the distribution of 'Class' (target variable)
plt.figure(figsize=(8, 6))
data['Class'].value_counts().plot(kind='bar', color=['blue', 'red'])
plt.title('Distribution of Class (0: Non-fraudulent, 1: Fraudulent)')
plt.xlabel('Class')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.show()

from imblearn.over_sampling import RandomOverSampler

# Oversample the minority class
oversample = RandomOverSampler()
X, y = oversample.fit_resample(X, y)

# Feature Engineering
# Add new features
data['Transaction_hour'] = pd.to_datetime(data['Time'], unit='s').dt.hour
data['Normalized_amount'] = (data['Amount'] - data['Amount'].mean()) /
data['Amount'].std()
```

```
# Feature Selection
# Select features
selected_features = ['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',
                     'Transaction_hour', 'Normalized_amount']

from sklearn.decomposition import PCA
from sklearn.feature_selection import SelectKBest, f_classif

# Define features (X) and target variable (y)
X = data[selected_features]
y = data['Class']

# Perform PCA for dimensionality reduction
n_components = min(X.shape[0], X.shape[1])  # Number of components should
be less than or equal to the minimum of samples or features
pca = PCA(n_components=n_components)
X_pca = pca.fit_transform(X)

# Perform feature selection on the PCA-transformed data
k_best_selector = SelectKBest(score_func=f_classif, k=5)  # Adjust k as needed
X_k_best = k_best_selector.fit_transform(X_pca, y)

# Get the indices of selected features
selected_indices = k_best_selector.get_support(indices=True)
```

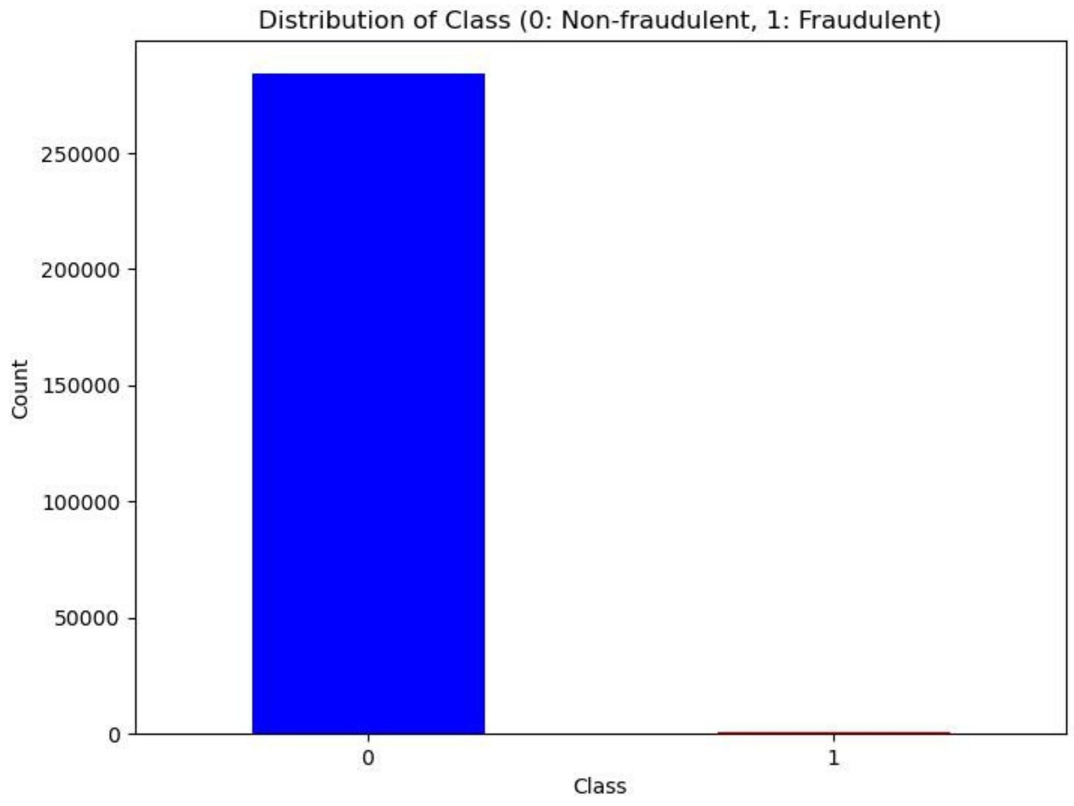
```
# Map selected PCA components back to original feature names
selected_features = [selected_features[i] for i in selected_indices]

print("Selected features using ANOVA F-test after PCA:")
print(selected_features)
# Split the data into training and validation sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Model Selection and Training
model = RandomForestClassifier(n_estimators=100, random_state=42)
# Train the model on the training data
model.fit(X_train, y_train)

# Cross-validation
cv_scores = cross_val_score(model, X_train, y_train, cv=5)
print("Cross-validation Scores:", cv_scores)
print("Mean Cross-validation Score:", np.mean(cv_scores))

# Model Evaluation
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)


# Hyperparameter Tuning using GridSearchCV
param_grid = {
    'n_estimators': [1, 5, 10],
    'max_depth': [None, 5, 10, 20],
    'min_samples_split': [2, 5, 10]
}
grid_search = GridSearchCV(model, param_grid, cv=5, n_jobs=-1)
grid_search.fit(X_train, y_train)
best_params = grid_search.best_params_
best_model = grid_search.best_estimator_
print("Best Hyperparameters:", best_params)


# Model Evaluation
y_pred = best_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)


print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)


# Save the best model
joblib.dump(best_model, 'credit_card_fraud_detection_model.pkl')
```

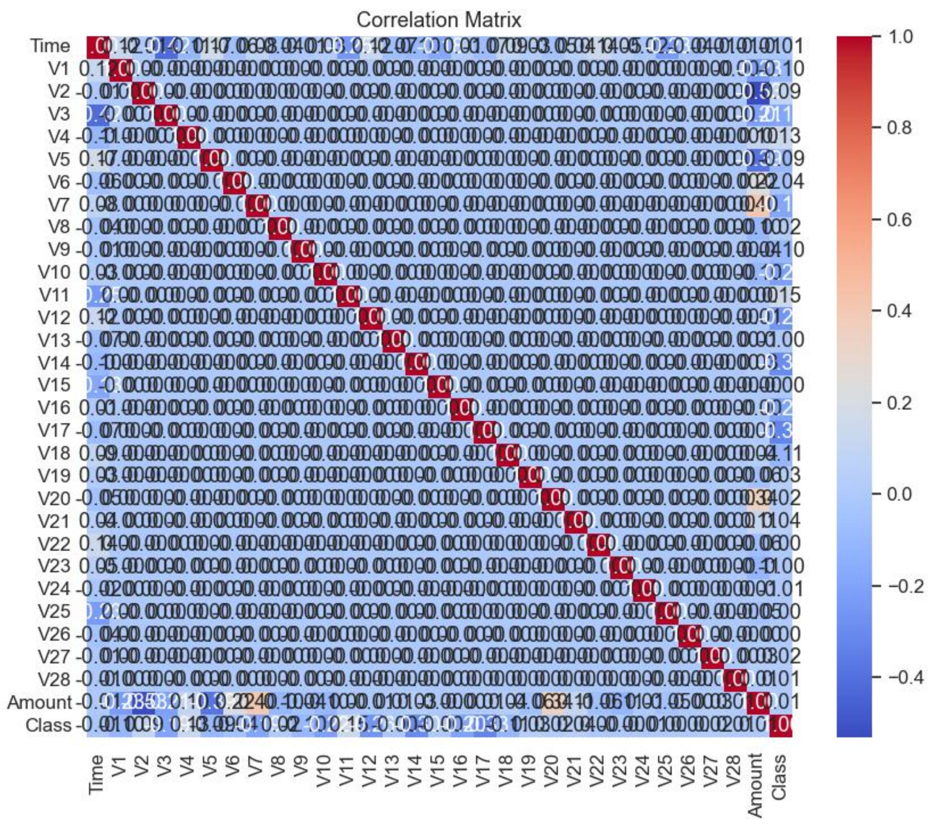
```
# Additional Visualizations

# Creating a heatmap for correlation matrix

plt.figure(figsize=(10, 8))

sns.heatmap(data.corr(), annot=True, cmap='coolwarm', fmt=".2f")

plt.title('Correlation Matrix')

plt.show()


# Scatter plot to visualize the actual vs. predicted classes for test data

plt.figure(figsize=(15, 6))

plt.scatter(range(len(y_test)), y_test, color='blue', marker='o', label='Actual')

plt.scatter(range(len(y_test)), y_pred, color='red', marker='x', label='Predicted')

plt.xlabel('Transaction Index')

plt.ylabel('Class (0: Non-fraudulent, 1: Fraudulent)')

plt.title('Actual vs. Predicted Classes for Test Data')

plt.legend()

plt.show()


# Plot the transaction volume over time

plt.figure(figsize=(12, 6))

plt.plot(data['Time'], data['Amount'], color='blue', alpha=0.5)

plt.title('Transaction Volume Over Time')

plt.xlabel('Time')

plt.ylabel('Transaction Amount')

plt.grid(True)

plt.show()
```

Visualisation:

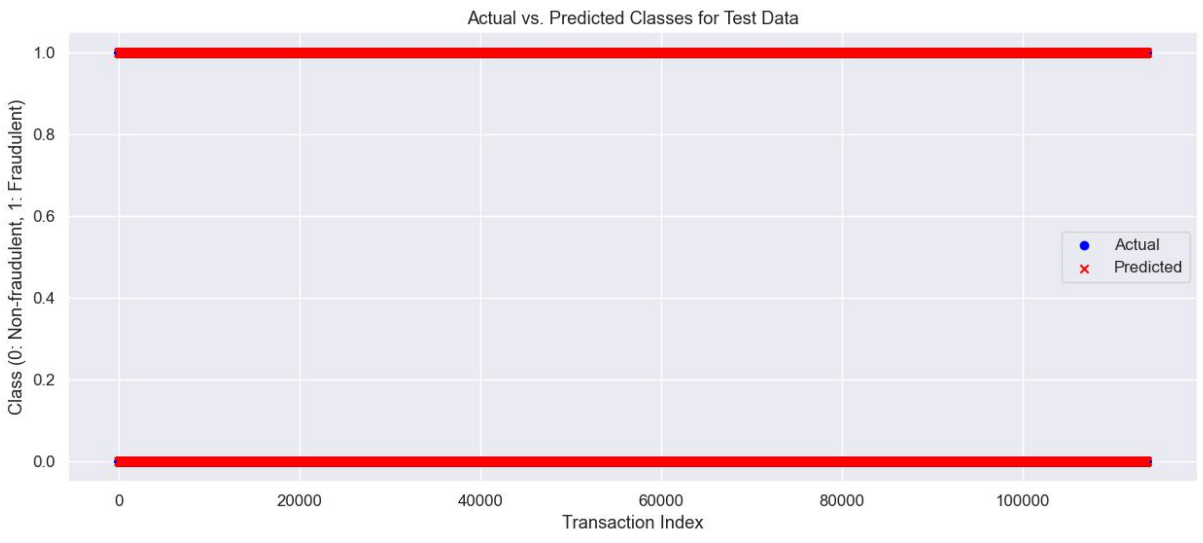


```
Accuracy: 0.9969400137171799
Confusion Matrix:
 [[56529   221]
 [  127 56849]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56750
           1       1.00      1.00      1.00     56976

    accuracy                           1.00    113726
   macro avg       1.00      1.00      1.00    113726
weighted avg       1.00      1.00      1.00    113726
```

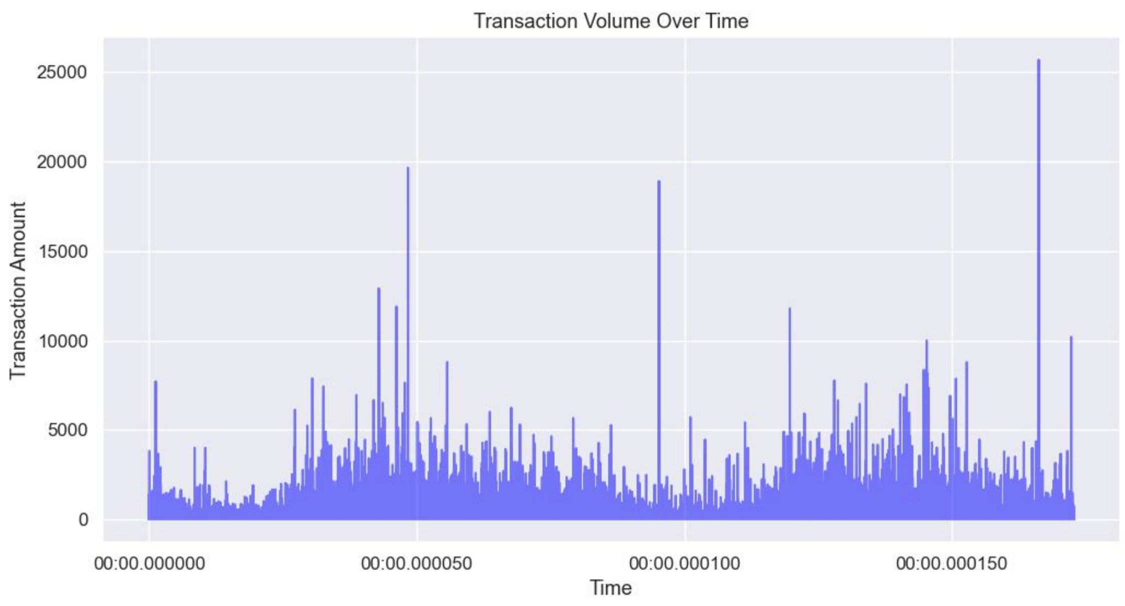




Correlation Matrix
Time
V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16
V17
V18
V19
V20
V21
V22
V23
V24
V25
V26
V27
V28
Amount
Class
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4



Actual vs. Predicted Classes for Test Data
Class (0: Non-fraudulent, 1: Fraudulent)
Transaction Index
● Actual
× Predicted
0
20000
40000
60000
80000
100000
0.0
0.2
0.4
0.6
0.8
1.0



Transaction Volume Over Time
Transaction Amount
Time
0
5000
10000
15000
20000
25000
00:00.000000
00:00.000050
00:00.000100
00:00.000150

CONCLUSION

Through this project, we have used predictive modelling approaches to create a reliable system for detecting credit card fraud. Our goal is to improve the financial integrity of organisations and consumers by correctly recognising fraudulent transactions. Our dedication to using data science to advance our business and its stakeholders is demonstrated by this effort.

Consumers and financial institutions are at serious danger from credit card theft. In this report, we describe how we are using predictive modelling tools to address this difficulty. Our goal is to create a reliable system that can minimise false positives while precisely detecting fraudulent transactions.

Consumers and financial institutions are at serious danger from credit card theft. In this report, we describe how we are using predictive modelling tools to address this difficulty. Our goal is to create a reliable system that can minimise false positives while precisely detecting fraudulent transactions.

Contact Information

Kanak Acharya

kanak.acharya1995@gmail.com