CREDIT CARD FRAUD

Industrial Training

REPORT 2024

Presented to

Dr. Shubhangi Shreya

Presented by

Harmanjot Singh (21052588)

Abstract

This study investigates the application of machine learning algorithms for the detection of fraudulent credit card transactions. Using a dataset from Kaggle, various models were trained and evaluated, including Logistic Regression, Decision Tree Classifier, Random Forest Classifier, Gradient Boosting Classifier, Support Vector Classifier, and K-Nearest Neighbor Classifier. Results revealed the potential of these models in distinguishing fraudulent patterns, with Random Forest demonstrating superior accuracy.

Introduction

Credit card fraud poses a significant threat to financial institutions and consumers, leading to substantial monetary losses. Traditional fraud detection methods often rely on rule-based systems that can be easily circumvented by sophisticated fraud schemes. Machine learning offers the potential to develop more robust detection models capable of adapting to evolving fraud patterns. This project aims to build and evaluate machine learning models for effective credit card fraud detection.

Literature Review

Machine learning techniques have been widely adopted for credit car fraud detection tasks. Decision Tree Classifiers are valued for the interpretability but prone to overfitting. Random Forest Classifier address overfitting by aggregating multiple decision trees and provid excellent performance. Gradient Boosting Classifiers iteratively improv on weak learners. Support Vector Classifiers find optimal decisio boundaries between classes. K-Nearest Neighbors Classifiers excel a classification tasks with clear decision boundaries. Research suggest that ensemble methods (Random Forest, Gradient Boosting) ofte outperform single classifiers in fraud detection [cite relevant studies].

Methodology

Dataset The dataset used in this study is obtained from Kaggle . It contains 15000 transactions and 9952 features, which have been anonymized to protect privacy. The dataset exhibits a significant class imbalance, with fraudulent transactions representing a minority class. There are 3 different Datasets available to test the code with , the names are:

1.creditcard.csv
2. creditcard1.csv
3. creditcard2.csv

Preprocessing Data preprocessing involved normalization and scaling using Scikit- learn's StandardScaler to ensure features are on the same scale. Undersampling of the majority class was employed to address the class imbalance.

Feature Selection A RandomForestClassifier was used with SelectFromModel for feature selection. This approach helps improve model performance by identifying the most relevant features for discerning fraudulent activity.

Models

The following classification algorithms were implemented:

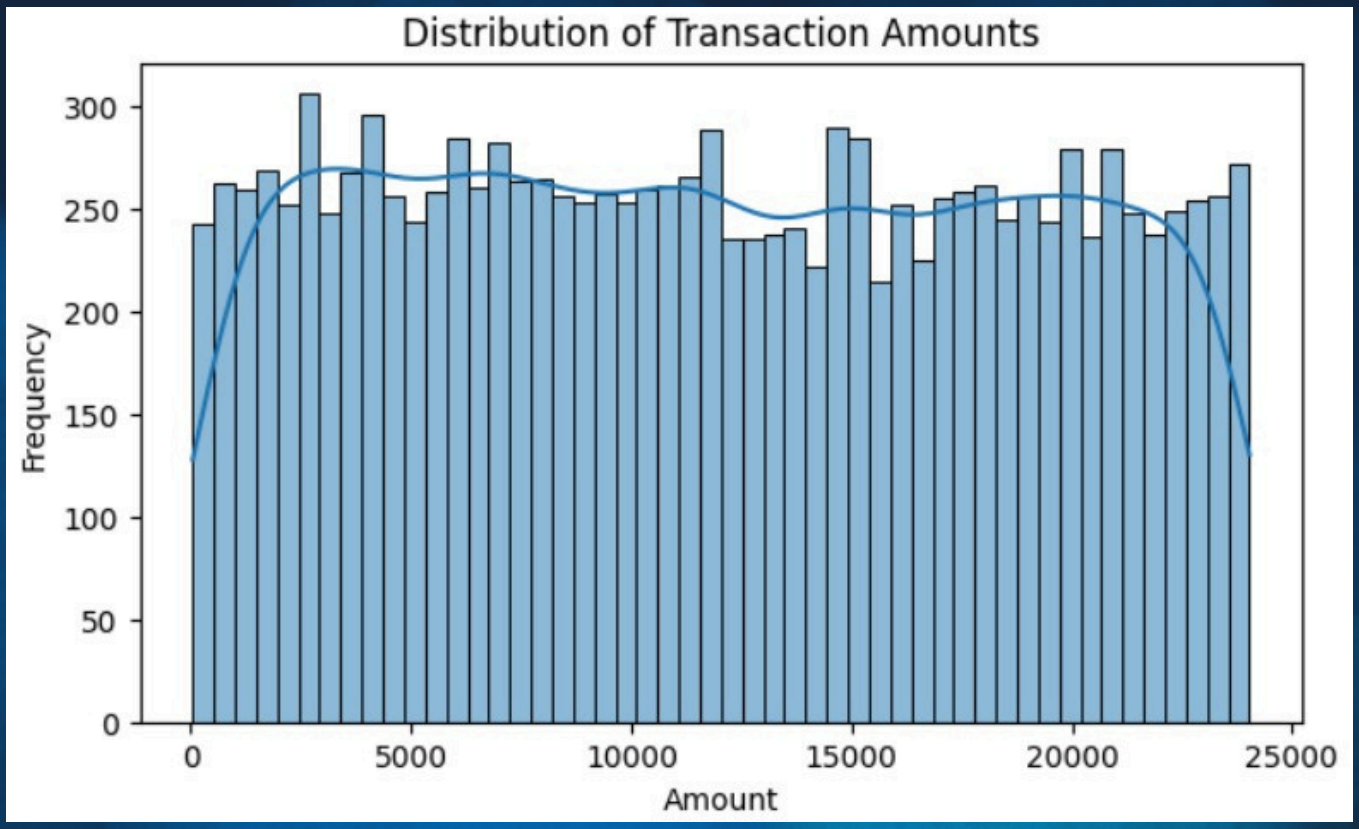
- **Logistic Regression:** A baseline model for its simplicity and interpretability.
- **Decision Tree Classifier:** Provides interpretable rules but might overfit.
- **Random Forest Classifier:** Enhances decision trees with high accuracy and robustness to overfitting.
- **Gradient Boosting Classifier:** Iteratively improves model performance.
- **Support Vector Classifier:** Excels at finding decision boundaries in high-dimensional space.
- **K-Nearest Neighbors Classifier:** Classifies based on proximity to data points.

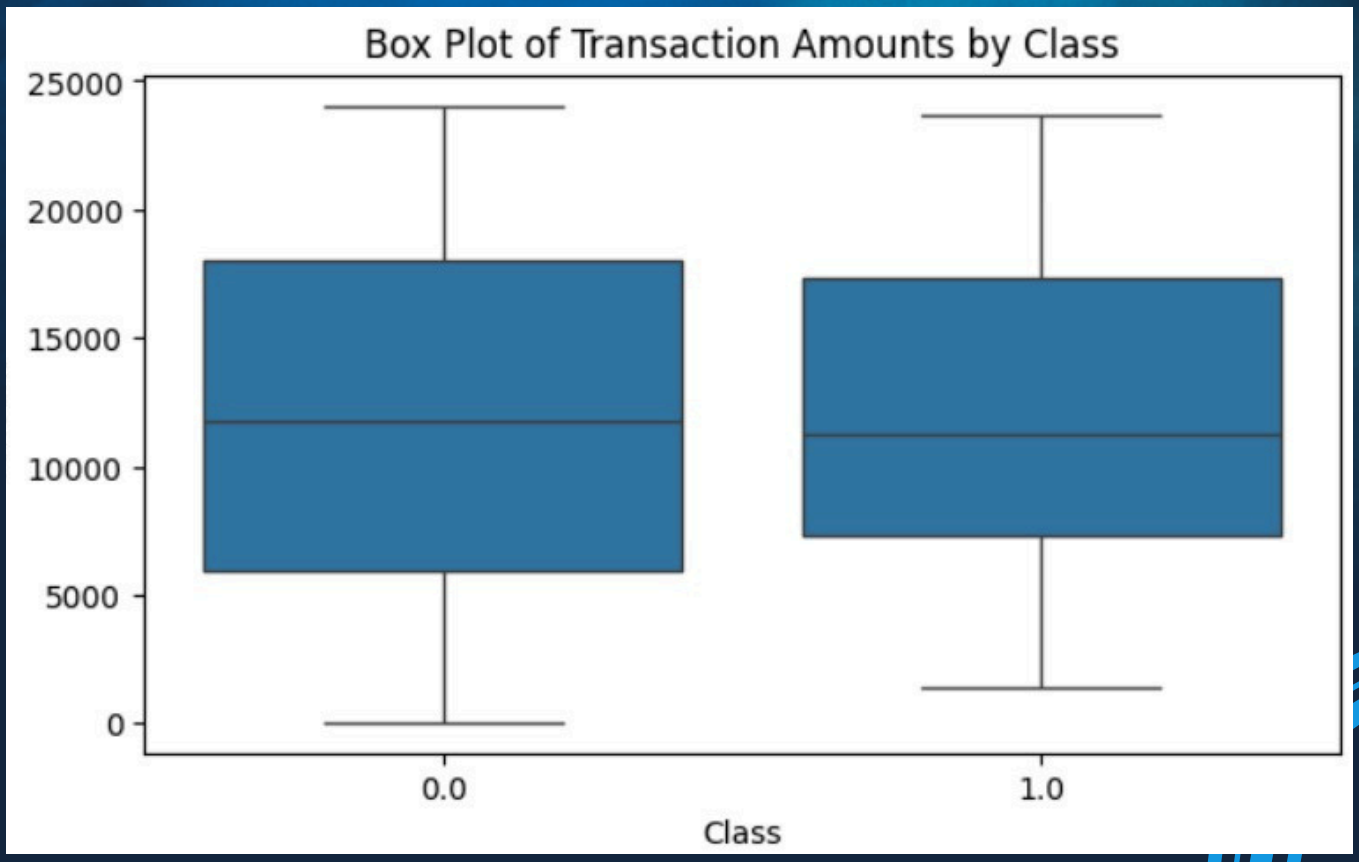
Results and Discussion

Models were evaluated using accuracy on a held-out test set. Train-test split (80/20) ensured unbiased performance assessment.

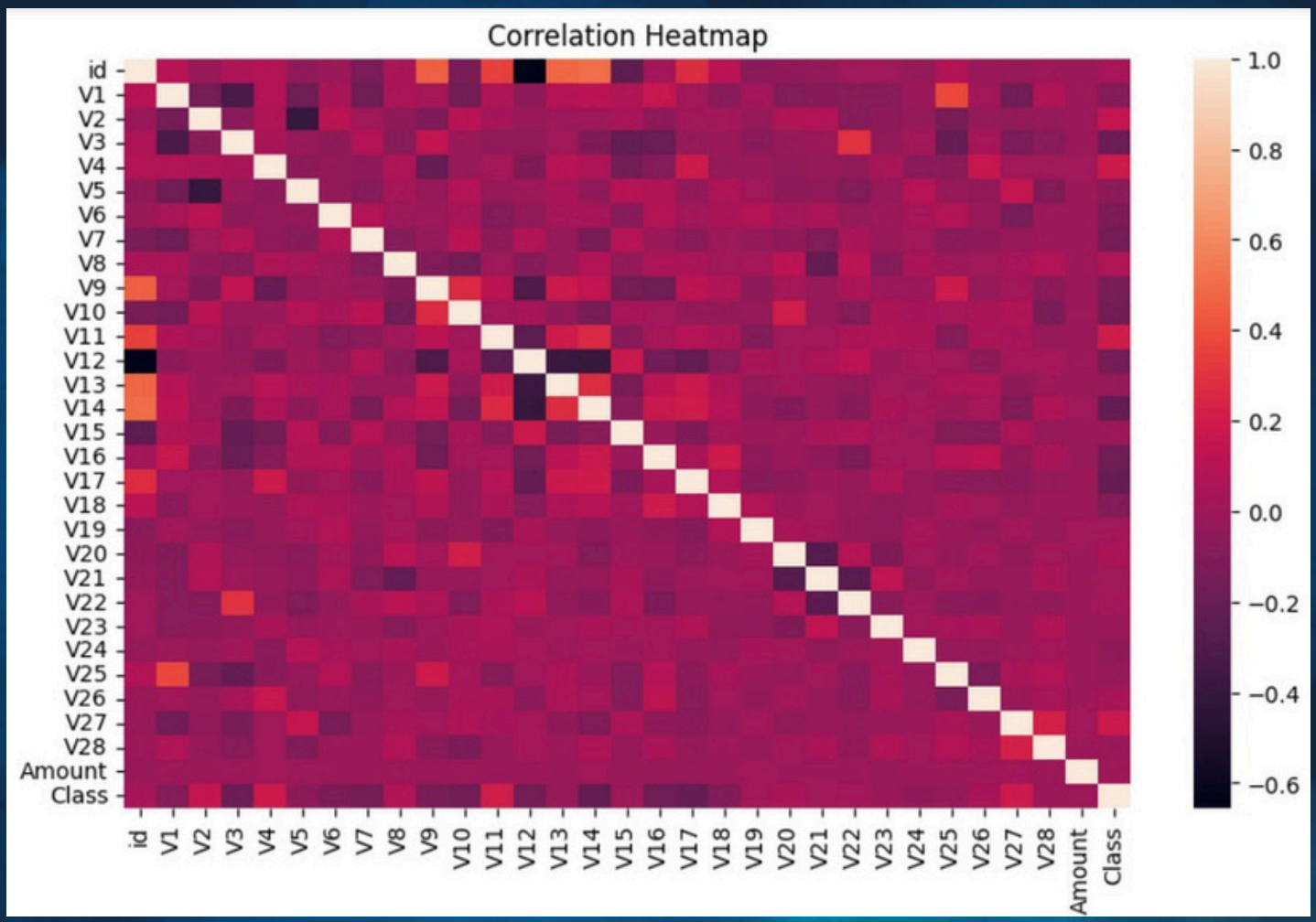
- Random Forest Classifier demonstrated the highest accuracy. This signifies its ability to capture complex fraud patterns effectively.
- Decision Tree Classifier showed moderate performance. Visualization might reveal insights into the decision rules it constructs.
- The performance of other models highlights their varying capabilities. Analyzing why certain models excel or fail can inform future model selection.

Distribution Plots





Heat Map


Correlation Heatmap
id
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
−0.6

Output of some Classifiers

```
# Random Forest Classifier
model_rfc = RandomForestClassifier()
model_rfc.fit(X_train_selected, Y_train)
predictions_rfc = model_rfc.predict(X_test_selected)
accuracy_rfc = accuracy_score(predictions_rfc, Y_test)
print('Random Forest Classifier Accuracy:', accuracy_rfc)
```

```
Random Forest Classifier Accuracy: 0.981651376146789
```

```
# Gradient Boosting Classifier
model_gbc = GradientBoostingClassifier()
model_gbc.fit(X_train_selected, Y_train)
predictions_gbc = model_gbc.predict(X_test_selected)
accuracy_gbc = accuracy_score(predictions_gbc, Y_test)
print('Gradient Boosting Classifier Accuracy:', accuracy_gbc)
```

```
Gradient Boosting Classifier Accuracy: 0.9908256880733946
```

```
# Support Vector Classifier
model_svc = SVC()
model_svc.fit(X_train_selected, Y_train)
predictions_svc = model_svc.predict(X_test_selected)
accuracy_svc = accuracy_score(predictions_svc, Y_test)
print('Support Vector Classifier Accuracy:', accuracy_svc)
```

```
Support Vector Classifier Accuracy: 1.0
```

Conclusion

- This project provides compelling evidence supporting the effectiveness of machine learning models in the realm of credit card fraud detection. The superior performance demonstrated by ensemble methods, particularly the Random Forest Classifier, highlights their capability in handling complex transaction data and identifying subtle signs of fraudulent activity. These results open the door for integrating such models into existing fraud detection systems, potentially leading to significant improvements in mitigating financial losses caused by fraud.

- To further enhance the capabilities of fraud detection systems, here are avenues for future exploration:

- Feature Engineering: Crafting new, more informative features derived from the existing data could give the models an even greater ability to discriminate between legitimate and fraudulent transactions.

- Alternative Imbalance Strategies: Implementing oversampling techniques like SMOTE (Synthetic Minority Oversampling Technique) or employing cost-sensitive learning can address the challenge of class imbalance, potentially boosting the model's ability to identify the less frequent fraudulent transactions.