Enhancing Credit Card Fraud Detection through Machine Learning and Deep Learning: A Comparative Analysis

Ya-Wei (Jeremy) Tsai

May 18, 2024

Abstract

This proposal outlines the approach for enhancing credit card fraud detection using advanced machine learning (ML) and deep learning (DL) techniques to combat increasing fraud sophistication. Utilizing real-life credit card transaction data, the project will apply a variety of ML and DL models and will be supported by Python and its libraries for data manipulation and model implementation.

1 Executive Summary

This project aimed to develop an effective system for detecting credit card fraud using various machine learning techniques. We experimented with models such as Random Forest, XGBoost, and Convolutional Neural Networks (CNN), focusing on their performance with tabular data. Our findings revealed that Random Forest and XGBoost significantly outperformed other models, including CNN, which did not perform well due to the lack of spatial features in the dataset. This report details the methodologies used, presents performance results, and provides recommendations for future research to enhance fraud detection systems. The project's significance lies in its potential to reduce financial losses and protect consumers by improving the accuracy and efficiency of fraud detection.

2 Introduction

Credit card fraud is a major issue in the financial sector, causing significant financial losses to both consumers and institutions. With the increasing volume of online transactions, the sophistication of fraudulent activities has also grown, making traditional detection methods less effective. Detecting fraudulent transactions promptly is crucial for minimizing these losses and protecting consumers' financial data. This project explores advanced machine learning techniques to develop a robust fraud detection system. By leveraging these techniques, we

aim to improve the accuracy and efficiency of detecting fraudulent transactions, thereby enhancing the security and trust in financial transactions.

3 Related Work

Several studies have addressed credit card fraud detection using different machine learning approaches.

3.1 Credit Card Fraud Detection Using Machine Learning Techniques: A Comparative Analysis

This study addressed the challenge of class imbalance inherent in fraud detection datasets. Researchers preprocessed real-life credit card transaction data using a hybrid approach that included both undersampling and oversampling techniques. They developed models using Naïve Bayes, K-nearest Neighbor, and Logistic Regression, and evaluated them using metrics such as accuracy, sensitivity, specificity, precision, and the Matthews Correlation Coefficient. The K-nearest Neighbor model showed significant performance, indicating the effectiveness of the hybrid sampling approach.

3.2 Fraud Detection using Machine Learning and Deep Learning

This research explored ensemble learning in fraud detection using European credit card transaction data. The data was processed with Principal Component Analysis (PCA) to obtain 30 main input features. The study used a meta-classification strategy by integrating several base classifiers, including decision trees, Naïve Bayesian, and K-nearest neighbor algorithms. The ensemble approach showed a 28% performance improvement when the Naïve Bayesian algorithm was applied at the meta-level.

3.3 Fraud Detection using Machine Learning and Deep Learning by Raghavan and El Gayar

Raghavan and El Gayar presented a comprehensive comparison of machine learning and deep learning techniques using datasets from European, Australian, and German domains. They benchmarked methods such as K-nearest Neighbor, Random Forest, Support Vector Machines (SVM), and deep learning techniques like autoencoders and Convolutional Neural Networks (CNN). Their evaluation employed metrics such as the Area Under the ROC Curve (AUC) and the Matthews Correlation Coefficient (MCC). The study concluded that ensemble models integrating SVM, Random Forest, and KNN can enhance performance on smaller datasets, while autoencoders showed potential in identifying fraud in dynamic environments.

4 Solution Description

4.1 Data Used

The dataset used in this project consists of credit card transactions labeled as fraudulent or non-fraudulent. The data was sourced from the Kaggle credit card dataset, which includes features such as transaction time, amount, and anonymized features.

4.2 Software and Libraries

The implementation was done using Python and various libraries, including:

- `scikit-learn` for K-Nearest Neighbors, Logistic Regression, Linear Regression, Random Forest, and preprocessing
- `xgboost` for XGBoost implementation
- `pytorch`, `keras`, and `tensorflow` for CNN and autoencoder models
- `pandas` and `numpy` for data manipulation

4.3 Model Descriptions

Convolutional Neural Network (CNN) The CNN model was designed to detect fraud in credit card transactions by learning hierarchical representations of the input data. The architecture of the CNN model is as follows:

- `conv1`: A 1D convolutional layer with 16 filters, kernel size of 3, stride of 1, and padding of 1.
- `pool`: A max-pooling layer with a pool size of 2.
- `conv2`: Another 1D convolutional layer with 32 filters, kernel size of 3, stride of 1, and padding of 1.
- `fc1`: A fully connected layer with 64 units.
- `fc2`: A fully connected layer with 1 unit, outputting the probability of a transaction being fraudulent.

The model was trained using the Adam optimizer and binary cross-entropy loss. Despite its complexity, the CNN did not perform as well on tabular data due to the lack of spatial features.

Autoencoder The autoencoder was employed to learn a compressed representation of normal transactions and detect anomalies. The architecture of the autoencoder is as follows:

- `encoder`:
  - A linear layer with input dimension equal to the number of features and output dimension of the encoding size.
  - A ReLU activation function.
  - A linear layer with output dimension of encoding size divided by 2.
  - A ReLU activation function.
- `decoder`:
  - A linear layer with input dimension of encoding size divided by 2 and output dimension of the encoding size.
  - A ReLU activation function.
  - A linear layer with output dimension equal to the number of features.
  - A Sigmoid activation function to ensure output values are between 0 and 1.

The autoencoder was trained using the Adam optimizer and mean squared error loss. Its performance was limited due to the inability to capture complex patterns in the tabular data.

K-Nearest Neighbors (KNN) The KNN model was implemented using scikit-learn, and the optimal number of neighbors (K=3) was selected through cross-validation. This model achieved high accuracy and MCC scores, indicating its effectiveness in detecting fraudulent transactions.

Random Forest The Random Forest model is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes for classification. It effectively handles tabular data and captures complex interactions between features. The model was implemented using scikit-learn, and it achieved high performance due to its robustness and ability to avoid overfitting.

XGBoost XGBoost (Extreme Gradient Boosting) is an optimized gradient boosting algorithm that provides high performance and speed. It handles missing data, regularizes the model to avoid overfitting, and supports parallel processing. The model was implemented using the xgboost library and achieved the highest accuracy and MCC scores among all the models tested.

Logistic Regression Logistic Regression is a linear model for binary classification that estimates the probability of a binary response based on one or more predictor variables. It was implemented using scikit-learn and provided a baseline for comparing more complex models. Despite its simplicity, it offered valuable insights into the importance of different features.

Linear Regression Linear Regression is a basic yet powerful model used to predict a continuous dependent variable based on one or more independent variables. Although more suited for regression tasks, it was used here as a baseline model to understand its performance on binary classification tasks when adapted.

Lasso Regression Lasso Regression is a type of linear regression that includes a regularization term to prevent overfitting by shrinking less important feature coefficients to zero. This model was implemented using scikit-learn and provided insights into the significance of regularization in preventing overfitting in the dataset.

5 Evaluation of Work

To effectively evaluate the performance of our model in the context of fraud detection using machine learning and deep learning techniques, we employed two critical metrics: Accuracy and the Matthews Correlation Coefficient (MCC). These metrics were chosen due to their relevance and effectiveness in assessing the quality of binary classification models, which is central to fraud detection tasks.

The Accuracy is defined as the ratio of correctly predicted observations to the total observations:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

However, in the case of imbalanced datasets, which is a common scenario in fraud detection where fraudulent transactions are much less frequent than legitimate ones, Accuracy alone can be misleading. Therefore, we also employed the Matthews Correlation Coefficient (MCC), which is a more reliable statistical rate that produces a high score only if the prediction obtained good results in all of the four confusion matrix categories (true positives, false negatives, true negatives, and false positives), proportionally both to the size of positive elements and the size of negative elements in the dataset.

The MCC is calculated using the formula:

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}$$

The use of MCC is particularly advantageous in our project as it provides a more nuanced view of the model's performance, especially important in our imbalanced dataset where the prevalence of non-fraudulent transactions could otherwise overshadow the less frequent but more important fraudulent cases. By using MCC, we ensure that both classes are taken into account and the measure is not biased by the majority class, hence giving us a balanced measure of our model's effectiveness.

Model	Accuracy	MCC
CNN (Benchmark)	0.9500	0.7981
Random Forest	0.9996	0.8350
XGBoost	0.9998	0.9101
Autoencoder	0.9499	0.0671
Linear Regression (Full predictors)	0.9970	0.3251
Lasso (Full predictors)	0.9967	0.0000
Logistic Regression	0.9973	0.4419
KNN (K=3)	0.9995	0.7976
Ensemble (Random Forest + XGBoost)	0.9997	0.8773

Table 1: Performance Metrics of Different Models

The Random Forest and XGBoost models achieved the highest accuracy and MCC scores. The CNN and autoencoder models underperformed, which can be attributed to the lack of structured features in the tabular data. The ensemble learning approach, while promising, did not outperform the single models of Random Forest and XGBoost.

5.1 Ideal Data

In an ideal world, additional data such as real-time transaction sequences and user behavior patterns would be beneficial. These could help in identifying temporal and sequential patterns in fraudulent transactions, potentially improving the model's performance.

5.2 Future Work

In the next 1-3 months, future work could explore models on datasets with temporal features to capture sequential patterns. Integrating real-time data streams for dynamic prediction and using graph-based methods to analyze relationships between transactions could also be beneficial.

5.3 Effort Description

This project involved extensive reading of research papers, coding in Python, and fine-tuning model parameters. Significant time was spent on data preprocessing and understanding the limitations of different models on tabular data.

- Learning new machine learning libraries and techniques
- Experimenting with different models and tuning hyperparameters
- Reading and summarizing related research papers
- Analyzing results and refining approaches

5.4 Challenges and Interesting Findings

One of the main challenges was handling the class imbalance in the dataset. Techniques like undersampling and oversampling were considered, but finding the optimal balance was crucial. Additionally, CNNs, which are typically effective for structured data, did not perform well on tabular data, highlighting the importance of choosing the right model for the data type.

References

[1] Awoyemi, John O. and Adetunmbi, Adebayo O. and Oluwadare, Credit card fraud detection using machine learning techniques: A comparative analysis, International Conference on Computing Networking and Informatics, 2017.

[2] Johan Perols, Financial Statement Fraud Detection: An Analysis of Statistical and Machine Learning Algorithms, American Accounting Association, 2011.

[3] Pradheepan Raghavan and Neamat El Gayar, Fraud Detection using Machine Learning and Deep Learning, International Conference on Computational Intelligence and Knowledge Economy, 2019.