ML project: Classifying credit card fraud

September 2022

1. Introduction

A credit card is a payment card issued to users (cardholders) by a bank or financial services company that allows cardholders to borrow short-term loans with which to pay for goods and services with merchants that accept cards for payment. Despite its convenience, there is a risk of fraud which is known as credit card fraud. The purpose is to obtain goods and services or to make payment to another account, which is controlled by a criminal. In 2019, the total value of fraudulent transactions using cards issued within SEPA and acquired worldwide amounted to €1.87 billion [1]. Therefore, regulators, card providers, and banks take considerable time and effort to collaborate with investigators worldwide to ensure fraudsters are not successful. With the advances in technology, machine learning is one approach to detecting credit card fraud that I will use to implement in this project.

Section 1 of the ML project report is an introduction to the real-life scenario of fraud detection in the financial service domain. Section 2 discusses the application of the problem and explains the source of the dataset. Section 3 defines two different methods that solve the problem and the motivation behind them. Section 4 reports the results and evaluates those based on some metrics. Section 5 summarizes the report and discusses what could be improved.

2. Problem Formulation

The dataset collected transactions made by credit cards in September 2013 by European cardholders (and was obtained on Kaggle [2]). For two days, 284,807 transactions were recorded, and each transaction represents a data point in the dataset.

Each data point has 30 numerical features and 1 binary label. Due to confidentiality issues, most of the features have been transformed with PCA to V1, V2, …, and V28, except two features ‘Time’ and ‘Amount’. The feature ‘Time’ describes the seconds elapsed between each transaction and the first transaction in the dataset. The feature ‘Amount’ is the transaction amount. Using these features, the objective is to classify the label ‘Class’, whether a transaction is a fraud (class ‘1’) or not (class ‘0’). Therefore, this is a supervised learning problem.

3. Methods

3.1. Preprocessing

The features mostly contain variables that are the results of the PCA transformation [3]. By plotting the correlation heatmap, we can see that they are not correlated with each other, but a few of them either has a positive or negative correlation with the target variable. Therefore, they are chosen to be the features matrix, along with the variable ‘Amount’ since it’s intuitive that the ‘Amount’ variable can influence the result of the ‘Class’ variable. In other word, I will only drop the ‘Time’ feature and use the rest of the features for building the models.

The dataset is split into two parts. The first part is the training set (80%) that uses to learn a hypothesis. The second part is the test set (20%) that use to test the hypothesis based on the recall, the accuracy and the confusion matrix.

The dataset reflects a real-life scenario that it is highly imbalance with 284,315 datapoints labeled non-fraud and only 492 labeled fraud. Because most machine learning techniques (including both Logistic Regression and Decision Tree) don't consider the number of each class, they end up having poor performance on the minority class, although typically the performance on the minority class is the most important. Therefore, in the training set, the minority class is resampled to match the majority class using SMOTE (Synthetic Minority Oversampling Technique). Basically, SMOTE will create new data points by synthesizing from existing data points. The result is the training set with 454,902 data points (284,315 in each class) with each data point represented by 29 features (from 'V1' to 'V28' and 'Amount') and 1 label ('Class').

3.2. Logistic Regression [3]

The Logistic Regression is a supervised binary classification method that classifies data points based on the probability into either of two classes. It uses a feature space and binary labels to learn a hypothesis out of the linear hypothesis space. The loss function used to assess the quality of a particular hypothesis is the logistic loss function. Because the logistic loss is applied in the ready-made library for Logistic Regression in the Scikit-learn library, it is chosen as the loss function for the algorithm.

Because the obtained dataset only contains numeric features and binary labels, the dataset meets the requirements for the method's input. In addition, the Logistic Regression can also be prone to overfitting, mainly when there is a high number of predictor variables. Therefore, the Logistic Regression is the first implemented model for the project.

3.3.Decision Tree [4]

The Decision Tree is a supervised ML model that can be used for both regression and classification. For classification, the Decision Tree is a method that classifies data points based on a series of "if/else" decisions in a flowchart-like tree. A decision tree consists of two components, the nodes, and the edges. Each node represents a condition of a feature. The nodes will be split into edges, which represent whether a condition is true or false. If a node doesn't split anymore, it is called a leaf which represents a decision or a prediction. To assess the quality of a particular decision tree (hypothesis), there are many loss functions that one can choose from ('Gini impurity', 'Information gain', or 'Entropy'). Because the 'Gini impurity' is already implemented in a default model of the Decision Tree Classifier, I will choose it as my loss function for the model.

For a binary classification problem, the Decision Tree model takes input as a feature space and binary labels which match the dataset used for this problem. But unlike the Logistic Regression, the Decision Tree model is more human-made because it is based on the decision just like a human, which is less complex and therefore easy to interpret. Thus, the Decision Tree model is the second method for this project.

4. Results

The results of both methods are printed using the 'classification_report' method of Scikit-learn. Based on those, the Logistic Regression achieved 98% accuracy and 91% recall in the fraud class (=1) and the Decision Tree achieved 100% accuracy and 0.81% recall in the fraud class (=1). Although the Decision Tree model achieved 100% accuracy, it doesn't mean it performs perfectly. By looking at the confusion matrix of the Decision Tree, the model correctly classified 79 out of 98 fraud transactions and misclassified 105 normal transactions. In the confusion matrix of the Logistic Regression, the correct classified cases are 89 out of

98, while the misclassified cases are much larger than the Decision Tree model, 1054. Therefore, no model outperforms the other, but one can consider both models. If the company values the correctly detected transaction more than misclassification, such a company may choose the Logistic Regression model.

5. Conclusion

These are just the results of only two models out of many models in the landscape of ML models. The performance can be improved by implementing other methods (Support Vector Machine, Deep Learning, etc.)

6. References

[1] European central bank (2021).
https://www.ecb.europa.eu/pub/cardfraud/html/ecb.cardfraudreport202110~cac4c418e8.en.html

[2] Kaggle dataset
https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud/code?datasetId=310&sortBy=voteCount

[3] Scikit-learn Logistic Regression
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html?highlight=logistic#sklearn.linear_model.LogisticRegression

[4] Scikit-learn Decision Tree Classifier
https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html?highlight=tree

[5] Information about machine learning models
https://github.com/alexjungaalto/MachineLearningTheBasics/blob/master/MLBasicsBook.pdf

7. Appendix