# Credit Card Fraud Detection Based on Support Vector Machine

Jianglin Xia*

Dept. of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA, USA 90089

*Corresponding author: jxia2404@usc.edu

**Abstract.** Due to the increasing popularity cashless transactions, credit card fraud has become one of the most common frauds and caused huge harm to the financial institutions and individuals in real life. In this academic paper, the algorithm Support Vector Machine (SVM) is used to build models to deal with the credit card fraud detection problem with the performance metrics AUC and F1-score. The experiment dataset is named Credit Card Transactions Fraud Detection Dataset from the Kaggle website. After the step of preprocessing, the dataset is split into the training, testing and validation dataset with 11 numerical features and a label feature called "is_fraud". The inner parameter "class_weight" of the SVM algorithm in Python is set as "balanced" to deal with the imbalanced datasets. The main method to find the optimized models is using the GridSearchCV function in Python library sklearn. After tuning the hyperparameters and handling the overfitting phenomenon, the optimized models for the two metrics are found. The parameter values of the best model for AUC are C=10, class_weight= "balanced", $\gamma$ =0.01, kernel = "rbf". The training AUC is 0.87 and testing AUC is 0.90. The parameter values of the final optimized model for F1-score are C=0.8, class_weight= "balanced", $\gamma$ =0.06, kernel = "rbf". The final training F-score is 0.305 and testing F-score is 0.260.

**Keywords:** Credit card, Fraud detection, Support Vector Machine, Imbalanced Data.

## 1. Introduction

In the information era, the financial companies are busy processing huge amounts of data from various resources and in different formats daily. If the data is not managed and secured well, it can be stolen and then cause harm to the entire business or individuals. There exist various types of frauds in the financial market. Due to the increasing popularity of cashless transactions, credit card fraud appears to be one of the most common frauds [1].

When someone uses a credit card fraudulently, they do so without the cardholders' knowledge and for their own purposes or requirements. This is referred to as credit card fraud. This is clearly an illegal act. Credit card fraud can be split into 2 types: inner card fraud and external card fraud. The inner card fraud aims to defraud the cash from false transactions as for the collusion between merchants and cardholders. Compared with the inner card fraud, the external card fraud represents the majority form. It focuses on using the stolen, fake, or counterfeit credit card to consume, or using cards to get cash in disguised forms, such as buying the expensive, small volume commodities or commodities that can be easily convertible into cash [2]. According to a BBC investigation, the estimated monetary value of worldwide credit card fraud rose from $21 billion in 2015, to $31 billion in 2020. Since the fraudsters tend to find more innovative and upgraded ways to implement frauds, the amount is more likely to increase in the next several years. Obviously, the problem is becoming more severe and requires urgent attention from more cutting-edge solutions.

Many scholars have proposed to use different algorithms from Machine Learning community to solve the credit card fraud detection problem. Among the algorithms, Support Vector Machine (SVM) becomes more and more popular due to its good theoretical foundations and good generalization capacity [3, 4]. It has been used a powerful tool for binary classification problems [5]. SVM and Random Forest Algorithms were compared in research to identify the fraudulent credit card transactions [6]. The result illustrated that SVM got the accuracy of 84.1%. The SVM turned out to be good though the accuracy was a bit lower than that of Random Forest which was 85.8%. Another study also used SVM and other algorithms like Logistic Regression (LR) and Quadratic Discriminant

Analysis (QDA) to detect credit card fraud [7]. After comparison, it became clear that SVM was superior to QDA and rather close to LR in terms of the accuracy. Therefore, SVM is demonstrated to be a comparatively great algorithm for credit card fraud detection.

The prior papers regarding the credit card fraud detection mainly focused on putting different kinds of Machine Learning algorithms into practice and finding the optimized one by carrying out the comparative analysis on the same performance metrics such as accuracy, sensitivity, precision, the Area Under Curve (AUC), F-score. Few of them were centered on using a specific algorithm to get the optimized model for the datasets through the process of tuning different hyperparameters within the algorithm. Thus, this paper concentrates on using a single algorithm called SVM to handle the credit card fraud detection problem instead of comparing different algorithms. The metrics AUC score and F-score are mainly used to evaluate the model performance. The whole model building process includes the steps of data cleaning, feature transformation, feature selection and hyperparameter tuning. After tuning the hyperparameters on the validation data, the best-found model would be checked if the overfitting or underfitting phenomenon exists on training and testing datasets and the optimized model is obtained finally.

## 2. Method

### 2.1. Dataset description and preprocessing

The dataset used in this paper is named Credit Card Transactions Fraud Detection Dataset, which is a simulated credit card transaction dataset. It contains legitimate and fraud transactions from 1st January 2019 to 31st December 2020 and includes credit cards of 1, 000 customers transacting with 800 merchants. The data source is from the Kaggle website [8].

The original dataset has about 1,850,000 items with no missing values for the combined training and testing datasets including the label feature, the number of features is 23. The binary target label is named “is_fraud”, so the main task of the research is to build the optimized models to detect the fraud credit card transactions.

In this research, to avoid the heavy calculation, the first 100,000 rows of data are selected from the original dataset. The train_test_split function is used to split the first 90,000 rows so that the training dataset has 80,000 rows and the testing dataset has 10,000 rows. The remaining 10,000 rows of data are employed as validation dataset.

Since the data has too many features, the preprocessing of feature selection and feature transformation needs to be done before building models. The “dob”, referring to date of birth of card holders, is transformed into age on the processing date. The “city_pop” feature is converted to “city_pop_segment” (city size) across the categories “Less Dense”, “Adequately Dense”, and “Densely Populated”. Additionally, the distance between the transaction location and merchant location is computed based on the features “displacement” and “location”, thus classified as “Nearby”, “Long Distance”, and “Far Away” based on the pre-set value ranges of displacement. Then the source variables generated during data processing are deleted, along with other non-relevant features. Finally, 11 features are retained for building models. The sample data with selected features is shown in Table 1 and the feature explanation is illustrated in Table 2. The Ordinal Encoder function is used to convert the object type data into numerical type data for its subsequent use, and a Standard Scaler function is then be used to scale and normalize the data.

**Table 1.** Sample data with selected features

| Feature name | Item-1 | Item-2 | Item-3 | Item-4 | Item-5 |
|---|---|---|---|---|---|
| merchant | 65.0 | 665.0 | 389.0 | 620.0 | 360.0 |
| category | 8.0 | 10.0 | 2.0 | 5.0 | 2.0 |
| amt | 791.35 | 180.92 | 81.20 | 120.59 | 68.29 |
| gender | 0 | 1 | 1 | 1 | 1 |
| zip | 48088 | 46346 | 51006 | 79842 | 77327 |
| job | 135.0 | 373.0 | 453.0 | 270.0 | 55.0 |
| unix_time | 1329605892 | 1327874422 | 1325834047 | 1327955741 | 1329820297 |
| age | 42 | 23 | 25 | 31 | 26 |
| displacement | 108.728778 | 71.657231 | 82.865748 | 83.770209 | 69.225902 |
| location | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| city_pop_segment | 1.0 | 2.0 | 2.0 | 2.0 | 0.0 |
| is_fraud (label) | 1 | 0 | 0 | 0 | 0 |

**Table 2.** Explanation for selected features

| Feature name | Description |
|---|---|
| merchant | merchant name |
| category | transaction category |
| amt | transaction amount |
| gender | sex of card holder |
| zip | transaction zip code |
| job | job of card holder |
| unix_time | time in Unix format |
| age | age of card holder |
| displacement | distance between transaction and merchant |
| location | level of the distance |
| city_pop_segment | size of the city |
| is_fraud (label) | target class |

### 2.2. Employed SVM model

Support Vector Machine, or simply SVM, is a powerful algorithm for binary classification problems [9]. The primal formulation of SVM is shown as following equation (1)

$$\min \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}\xi_i \tag{1}$$

$$\text{subject to} \quad y_i(\omega^T \cdot \varphi(x_i) + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0, i = 1, 2, \ldots, n$$

This formulation is called the soft-margin SVM. In essence, SVM picks a hyperplane to separate the data points from two classes with the largest margin, which means the smallest distance from all training points to the hyperplane.

There are mainly four kinds of hyperparameters for SVM to decide, which are regularization factor C, kernel function type, kernel efficient $\gamma$ and degree of kernel functions. In particular, the degree is for polynomial kernel functions. Therefore, with different kernel functions, different hyperparameters need to be tuned to find the best model. The following kernel functions are popularly used ones [10].

- Polynomial kernel with degree d: $K(x_i, x_j) = \gamma({x_i}^T x_j + 1)^d, \gamma > 0$

- Radial basis function kernel is given as: $K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \gamma > 0$
- Sigmoid kernel function is given as: $K(x_i, x_j) = tanh(\gamma {x_i}^T x_j + r)$

The method to find the optimized model is using the GridSearchCV function in Python library sklearn. The inner parameter "class_weight" is set as "balanced" to deal with the imbalanced datasets. The first step is to find out the best kernel function to fit the validation data because various kernel functions have different hyperparameters. In this research, the best kernel function for the validation data is Radial basis function kernel for both metrics AUC and F-score. Therefore, the remaining hyperparameters are regularization factor C and kernel efficient γ .

The second step is to use the GridSearchCV function to tune these two kinds of hyperparameters. The value range for C is 0.001, 0.01, 0.1, 1, 10, 100 and value range for γ is 0.001, 0.01, 0.1, 1, 10, 100, "scale", "auto". Specifically, the default "scale" is 1/(n_features * X.var( )) and "auto" is 1/n_features. With the chosen evaluation metrics AUC and F-score, this powerful function will automatically find the best value sets of the hyperparameters for the validation dataset. Then the best-found model will be fitted into the training dataset and tested on the testing dataset. Finally, the overfitting or underfitting phenomenon would be checked over the training and testing results. If it exists, the values of hyperparameters would be adjusted a little bit to solve this issue.

## 3. Result and discussion

**Table 3.** Result for AUC

| Model Parameters | Training AUC | Testing AUC |
|---|---|---|
| C=10, class_weight="balanced", γ =0.01, kernel = "rbf" | 0.87 | 0.90 |

**Table 4.** Result for F1-score

| Model Parameters | Training F1-score | Testing F1-score |
|---|---|---|
| C=100, class_weight="balanced", γ ="auto", kernel = "rbf" | 0.793 | 0.446 |
| C=0.8, class_weight="balanced", γ ="auto", kernel = "rbf" | 0.311 | 0.243 |
| C=100, class_weight="balanced", γ =0.02, kernel = "rbf" | 0.272 | 0.205 |
| C=0.8, class_weight="balanced", γ =0.06, kernel = "rbf" | 0.305 | 0.260 |

Table 3 and Table 4 present the result of AUC and F1-score. In terms of AUC, the parameter values of the best model are C=10, class_weight="balanced", γ =0.01, kernel = "rbf". The training AUC is 0.87 and testing AUC is 0.90 so there is no overfitting or underfitting phenomenon for this model.

As for the F1-score, the original parameter values of the optimized model from the GridSearchCV function are C=100, class_weight="balanced", γ ="auto", kernel = "rbf". However, the corresponding training F-score is 0.793 and testing F-score is 0.446. Thus, the overfitting phenomenon needs to be solved. After adjusting the values of C and γ , the parameter values of the final optimized model are C=0.8, class_weight="balanced", γ =0.06, kernel = "rbf". Finally, the training F-score is 0.305 and testing F-score is 0.260.

From the above results, it is obvious that SVM has great performance on the AUC score with testing AUC 0.90. The metric result turns out to be excellent and there is no overfitting phenomenon between training and testing.

However, when using SVM on the F1-score, the overfitting phenomenon exists in the original best model and the difference between training and testing is relatively large. Then the main strategies to fix this problem are decreasing the value of C or γ. In Table 4, the second model is to decrease C only

and the third model is to decrease γ only. Nevertheless, the difference between training and testing is not small and the testing F-score is not satisfactory. Therefore, the final best model tries to decrease C and γ at the same time, and the overfitting problem is solved.

## 4. Conclusion

This research focuses on using a single algorithm called SVM to deal with the credit card fraud detection problem with the performance metrics AUC and F1-score. After tuning hyperparameters and dealing with overfitting problems on the dataset, the optimized models for the two metrics are found. As for the AUC, the parameter values of the best model are C=10, class_weight="balanced", γ =0.01, kernel = "rbf". The training AUC is 0.87 and testing AUC is 0.90. As for the F1-score, the parameter values of the final optimized model are C=0.8, class_weight="balanced", γ =0.06, kernel = "rbf". The final training F-score is 0.305 and testing F-score is 0.260. The main limitations of this research are not enough in terms of the data size and employed balancing method. Partial data from the original dataset is selected to build the models and the imbalanced data is handled by the inner parameter "class_weight". Therefore, the future plan is to enlarge the data size and try other resampling method like Synthetic Minority Oversampling Technique (SMOTE) to enhance the model performance.

## References

[1] Varmedja, D., et al. "Credit card fraud detection-machine learning methods," 2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH), IEEE, 1-5 (2019).

[2] Shen, A., et al. "Application of classification models on credit card fraud detection," 2007 International Conference on Service Systems and Service Management, IEEE, 1-4 (2007).

[3] Noble, S., "What is a support vector machine?" Nature biotechnology 24.12 (2006): 1565-1567.

[4] Suthaharan, S., "Support vector machine." Machine learning models and algorithms for big data classification. Springer, Boston, MA, 2016. 207-235.

[5] Cervantes, J., et al. "A comprehensive survey on support vector machine classification: Applications, challenges and trends," Neurocomputing 408, 189-215 (2020).

[6] Hussain, S.S., et al. "Fraud Detection in Credit Card Transactions Using SVM and Random Forest Algorithms," 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), IEEE, 1013-1017 (2021).

[7] Naveen, P. et al. "Relative Analysis of ML Algorithm QDA, LR and SVM for Credit Card Fraud Detection Dataset," 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), IEEE, 976-981 (2020).

[8] Shenoy, K., "Credit Card Transactions Fraud Detection Dataset," Kaggle, 5 Aug 2020, < https://www.kaggle.com/datasets/kartik2112/fraud-detection>.

[9] Yu, H. et al. "SVM Tutorial-Classification, Regression and Ranking," Handbook of Natural Computing 1, 479-506 (2012).

[10] Hussain, M., et al. "A comparison of SVM kernel functions for breast cancer detection," 2011 Eighth International Conference Computer Graphics, Imaging and Visualization, IEEE, 145-150 (2011).