

Credit Card Fraud Detection : A Comparison using Random Forest, SVM and ANN

Simi M.J.

Student, Dept. of Computer Science, St. Joseph's College (Autonomous) Irinjalakuda, Thrissur, Kerala, India

Abstract - An effective credit card fraud detection system is essential for today. Because everyone uses credit cards today's day to day life. We use three algorithms in machine learning and compare the accuracy of system in credit card fraud detection. It uses full historical transactions of a person including normal transaction data or fraudulent to get normal/fraud transaction features and then these features are used to check whether a transaction is normal or not. The three machine learning supervised algorithms are Random Forest, Support Vector Machine (SVM) and Artificial Neural Network (ANN). The final result will be based on the most accurate algorithm used.

Key Words : Random Forest, SVM, ANN, decision tree, credit card fraud

1. INTRODUCTION

In today's world most of the people uses credit cards for transactions. Changes in mobile intelligent devices and e-commerce made the people to use the credit cards for easy transactions. Credit cards are very useful. It is very easy to carry. Card – not – present transactions or online transactions are very famous today.

Like any other technology, the credit cards have both advantages and disadvantages. The users of credit cards may normal or fraudulent. So it is important to detect to which category a user belongs to. There are many chances to perform fraudulent activities using a credit card.

Fraud detection is a process of monitoring the transaction behaviour of a cardholder in order to detect whether an incoming transaction is done by the cardholder or others [1]. It uses full historical transactions of a person including normal transaction data or fraudulent to get normal/fraud transaction features and then these features are used to check whether a transaction is normal or not.

This paper uses three machine learning algorithms Forest, Support Vector Machine (SVM) and Artificial Neural Network (ANN). These algorithms are supervised algorithms. Hence it has training and testing phases.

2. RELATED WORK

An understanding of fraud detection technologies will help us to solve the problem of credit card fraud. The credit card frauds can be categorized into two: application fraud and behavior fraud. When criminals collect new credit cards

from issuing companies by submitting wrong information or legitimate cardholder information is called application fraud. When the criminals steal the account and password of a legitimate user and use that account and card details for their personal luxury life is known as behavior fraud.

Almost in all banks the behaviors of the cardholder is tested. For this the transaction patterns of each user is collected. These patterns are tested for finding whether the user is normal or not.

Srivastava et.al. [2] model the sequence of transaction features in credit card transaction processing using a hidden markov model (HMM) and demonstrate its effectiveness on the detection of frauds. An HMM is initially trained with the normal behaviour of the cardholder. If the current transaction is not accepted by the trained HMM with a high probability, it is considered to be fraudulent. However, they only consider the transaction amount as the feature in the transaction process. Amlan et.al [3] propose a method using two-stage sequence alignment which combines both misuse detection and anomaly detection [4]. In their method, a profile analyser is used to determine the similarity of an incoming sequence of transaction on a given credit card with the legitimate cardholder's past spending sequence. Then, the unusual transactions traced by the profile analyser are passed to a deviation analyser for possible alignment with the past fraudulent behaviour. The final decision about the nature of a transaction is taken on the basis of the observations by the two analysers. However, this method cannot detect frauds in real time.

3. RANDOM FOREST

Random Forest is a machine learning algorithm. It is a supervised algorithm. It is a tree based algorithm. It creates several decision trees and combines their outputs to produce a good model. The process of combining the decision trees is known as ensemble process.

Advantages and Disadvantages of Random Forest[5]

Advantages are as follows:

1. It is robust to correlated predictors.
2. It is used to solve both regression and classification problems.

3. It can be also used to solve unsupervised ML problems.
4. It can handle thousands of input variables without variable selection.
5. It can be used as a feature selection tool using its variable importance plot.
6. It takes care of missing data internally in an effective manner.

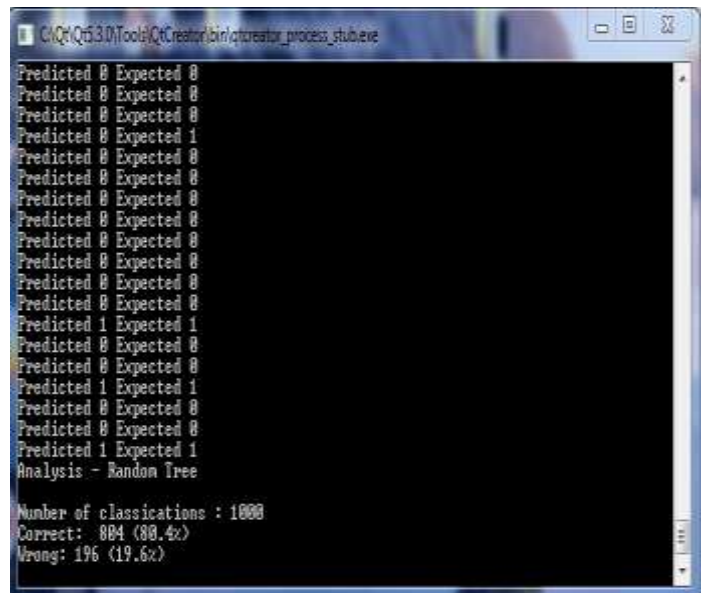
Disadvantages are as follows:

1. The Random Forest model is difficult to interpret.
2. It tends to return erratic predictions for observations out of range of training data. For example, the training data contains two variable x and y. The range of x variable is 30 to 70. If the test data has x = 200, random forest would give an unreliable prediction.
3. It can take longer than expected time to computer a large number of trees.

The code for training data set

```
void RandomForest::training(Mat trainData,Mat trainLabel)
{
Ptr<RTrees> rfClassifier =RTrees::create();
rfClassifier->setMaxDepth(5);
rfClassifier->setMinSampleCount(3);
rfClassifier->setTermCriteria(TermCriteria(TermCriteria::MAX_ITER,10,1e-6));
cout<<"Training started"<<endl;
trainData.convertTo(trainData,CV_32FC1);
rfClassifier->train(trainData , ROW_SAMPLE , trainLabel);
rfClassifier->save("frauddetection.xml");
cout<<"Training completed"<<endl;
}
```

The result of using random forest on a set of data for testing is



4. SUPPORT VECTOR MACHINE

SVM is a supervised algorithm associated with machine learning. It is defined by separating hyperplane. Important terms in SVM

- a) Hyperplane : a line that separates and classifies a set of data.
- b) Support Vector: data points nearest to the hyperplane.
- c) Margin : distance between hyperplane and nearest data point

Advantages:

1. High-Dimensionality
2. Memory Efficiency
3. Versatility

Disadvantage:

1. Non-Probabilistic

The code used for training a set of data

```
void SupportVectorMachine::training(Mat trainData ,Mat
trainLabels)
{
Ptr<cv::ml::SVM> svm= cv::ml::SVM::create();
svm->setType(cv::ml::SVM::C_SVC);
svm->setKernel(cv::ml::SVM::RBF);
```

```
svm>setTermCriteria(cv::TermCriteria(TermCriteria::MAX_ITER,1000, 1e-6));

svm->setGamma(3);

svm->setC(1);

trainData.convertTo(trainData, CV_32FC1);

cv::Ptr<cv::ml::TrainData>td=cv::ml::TrainData::create(trainData, cv::ml::ROW_SAMPLE, trainLabels);

svm->train(td);

// auto train

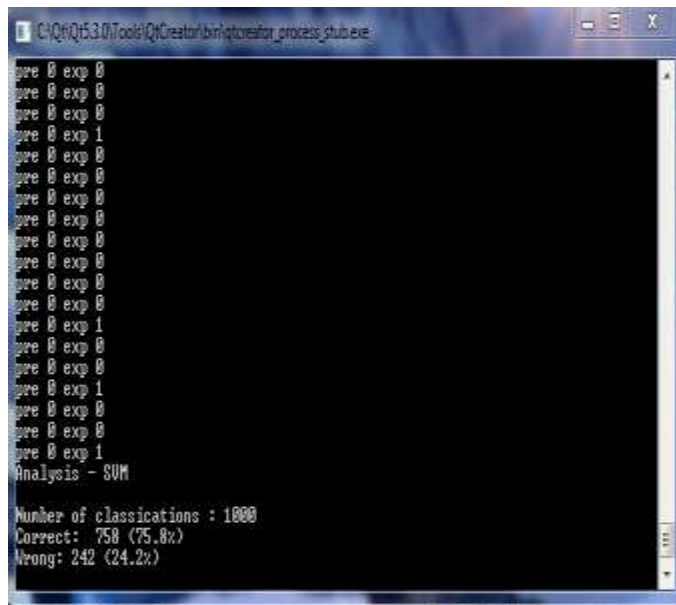
svm->trainAuto(td);

svm->save("svm.xml");

cout << "Training Completed" << endl;

}
```

Result of testing data set using SVM



5. ARTIFICIAL NEURAL NETWORK

ANN is a computational algorithm. It is used for both machine learning and pattern recognition. It is like a system of interconnected neurons which can be used to calculate the values from inputs. It is a machine learning algorithm based on the model of human neurons. It process information like a human brain works.

ANN contains several connected processing units. These processing units are used to process information. The neural network contain three layers

1. Input Layer
2. Hidden Layer
3. Output Layer

The code for training data set using ANN

```
void ArtificialNeuralNetwork::training(cv::Mat trainData,cv::Mat trainLabels)
{
trainData.convertTo(trainData, CV_32FC1);

Mat layers(1, 3, CV_32SC1);

// input layer

layers.at<int>(0) = trainData.cols;

// hidden layer

layers.at<int>(1) = 65;

// output layer

layers.at<int>(2) = 2;

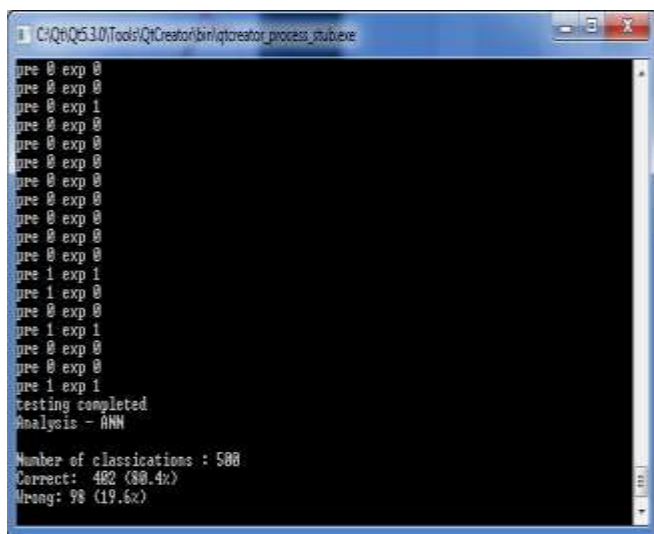
// prepare trainclases

Mat trainClasses(trainData.rows, 2, CV_32FC1);

for(int i=0 ; i<trainClasses.rows; i++)
{
for(int k=0; k<trainClasses.cols; k++)
{
if(k == trainLabels.at<int>(i))
{
trainClasses.at<float>(i,k) = 1;
}
else
{
trainClasses.at<float>(i,k) = 0;
}
}
}
}
```

```
cv::Ptr<cv::ml::ANN_MLP>ann=cv::ml::ANN_MLP::  
create() ;  
ann->setLayerSizes(layers);  
ann->setActivationFunction(cv::ml::ANN_MLP::SIGMOID_SYM,  
1, 1);  
ann->setTermCriteria(cv::TermCriteria(cv::TermCriteria::MA  
X_ITER, 1000, 1e-6));  
ann->setTrainMethod(cv::ml::ANN_MLP::BACKPROP,  
0.0001, 0.001);  
Ptr<cv::ml::TrainData>Data=cv::ml::TrainData::create(  
trainData, cv::ml::ROW_SAMPLE, trainClasses);  
ann->train(Data);  
ann->save("AnnResult.xml");  
cout<<"training completed"<<endl;  
}
```

The result of testing data set using ANN



```
C:\QRC5.10\Tools\QtCreator\bin\qtcreator_process_stu.exe  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 1  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 0 exp 0  
pre 1 exp 1  
pre 1 exp 0  
pre 0 exp 0  
pre 1 exp 1  
pre 0 exp 0  
pre 0 exp 0  
pre 1 exp 1  
testing completed  
Analysis - ANN  
Number of classifications : 500  
Correct: 482 (96.4%)  
Wrong: 18 (3.6%)
```

6. CONCLUSION

This paper uses three machine learning algorithms. Random Forest, SVM and ANN. From the comparison we reach in a conclusion that Random Forest have more accuracy than SVM and less accuracy than ANN in fraud detection. So the final result will be calculated based on Artificial Neural Network(ANN). In future we can add more algorithms to improve the functioning of credit card fraud detection systems.

REFERENCES

- [1] Duman, E., and Ozcelik, M. H. (2011). Detecting credit card fraud by genetic algorithm and scatter search. *Expert Systems with Applications*, 38(10), 13057-13063.
- [2] Srivastava, A., Kundu, A., Sural, S., and Majumdar, A. (2008). Creditcard fraud detection using hidden markov model. *IEEE Transactions on Dependable and Secure Computing*, 5(1), 37-48.
- [3] Kundu, A., Panigrahi, S., Sural, S., and Majumdar, A. K. (2009). Blastssaha hybridization for credit card fraud detection. *IEEE Transactions on Dependable and Secure Computing*, 6(4), 309-315.
- [4] Ju, W. H., and Vardi, Y. (2001). A hybrid high-order markov chain model for computer intrusion detection. *Journal of Computational and Graphical Statistics*, 10(2), 277-295.
- [5] Manish Saraswat
<https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/tutorial-random-forest-parameter-tuning-r/tutorial/>