Comparison of Algorithms for Predicting Time and Accuracy of Credit Card Fraud

Assignment Presented to
Professor Jean-Francois Plante

Statistical Learning
60603A – Fall 2020

Huai Jun Sun, 11287405
Saba Daftari, 11254107
Xinyuan Qi, 11272794

Due Date
November 12th, 2020

Abstract

The problem of credit card fraud can have a major impact on enterprises. In real life, the prediction model of credit card fraud detection has been widely used. However, different machine learning algorithms also have a certain influence on the prediction process and results. This paper will focus on studying the credit card fraud prediction of different models by evaluating four algorithms, including logistic regression, random forest algorithm, k-nearest-neighbours classification algorithm, naive bayes classification algorithm, to compare their training speed, prediction speed and accuracy for different training sets of different sizes. We also compare how the model reacts to changes in data nature. Ultimately we seek to determine which model is most suitable for a latency sensitive application such as credit card fraud. In this project, the basic data set will be randomly generated using Monte Carlo simulation.

1. Context

The number of credit card transactions has been on the rise in recent years. As one of the main components of the payment system, credit card fraud also comes with the problem. According to MERCHANT SAVVY's report, global losses from payment fraud has tripled from $9.84 in 2011 to $32.39 in 2000. Meanwhile, all over the world, 47% of companies surveyed said they had experienced fraud in the past 2 years. In order to avoid additional losses and the further spread of credit card fraud, it is important for enterprises to predict credit card fraud. In this project, we are going to optimize the credit card fraud prediction from two perspectives.

Credit card fraud prediction times for transactions can be extremely important for companies, since shorter transaction times will improve customer satisfaction and mitigate churn. However, accuracy levels are also important to ensure that the company minimizes their losses. The most accurate models often use the most data, however we know that increased sizes in training sets and complex models can increase training and prediction times. Thus, we wish to determine the training and prediction times of various models with varying sizes of training data, while ensuring an acceptable level of accuracy.

2. Methodology

2.1. Models Selected

In this project, we will evaluate the performance of four models. The first method is logistic regression. Generally speaking, logistic regression is a widely used forecasting model, which is used to forecast fraud in some industries nowadays. The advantage of Logistic regression is that the prediction is fast and relatively simple. Thus, it serves as the baseline model in this project. For the second method, we chose the random forest algorithm. The random forest algorithm is mainly affected by the two types of parameters, the number of decision trees and subset size. For the third method, we chose to use the k-nearest-neighbours classification algorithm. We will use k equal to 10 for this particular simulation. For the fourth method is the Naive Bayes algorithm.

For all models, we will leave all optional parameters at default values and the training set will consist of balanced 50-50 data, as some of our models do not perform well with the realistically low proportion of fraudulent transactions.

2.2. Method Description

There are four functions *logistic_regression, randomForestmd, knn_md, nbc,* associated with each of the algorithms. Each function calculates the train and test time of the algorithm used for predicting the two datasets passed through the arguments.

The *data_change_simulation* function takes three datasets (old, new, test) and a function as arguments. First it creates 10 fractions for the old dataset and 10 fractions for the new dataset. In a for loop we managed to combine two fractions from two datasets, and store it as the train set. predicting using the function passed through the argument and returning the list of accuracy of all ten fraction sets.

The *streaming* function takes a dataset and two numbers and a function as arguments. It will split the dataset into two parts by one of these numbers and then take one part as the train set and the other as the test set. Between the two numbers we increment 1000 each time to see what happens with the predictor function that has been passed as the argument. Finally it returns the accuracy list for each increment.

The *n_observation_analysis* function takes a dataset, three numbers and a function as arguments. First we call the *streaming* function by the same arguments. Then it creates and returns a result data frame containing *observation size*, *train time*, *predict time*, and *accuracy*.

In the modeling section, given 0 to 2000 observations from data frame 1, we take min = 20 and max = 1000 testing observations from the data frame 2, incrementing by rate of 5. The same configure for all four algorithms and the performance is plot as the following. This will repeat for training time, testing time, and accuracy level versus data set size for each of the algorithms.

Lastly as an additional analysis using function *changing_data_analysis*, we will look at the changes in accuracy when the composition of a dataset is changed between data of two different natures. We vary the parameters outlined below in data generation to produce a secondary dataset. We then gradually increase the number of rows of the secondary dataset in the training data, and later reduce the number of rows of the primary dataset.

3. Research Question

Logistic regression, Random forest algorithm, K-nearest-neighbours classification algorithm and Naive Bayes algorithm were evaluated in this project. Our research question revolves around understanding the strengths and weaknesses of the four different models mentioned above applied in the context of credit card fraud detection. We know that generally larger datasets are favored as they tend to yield higher performances, yet conversely we also know that models will generally take longer to train and sometimes predict with greater amounts of training data. Thus, we look to simulate various different models with various different training set sizes to observe the behaviour in terms of prediction time, training time, and accuracy. In the end, we wish to make informed recommendations for a real-time credit card fraud prediction model.

4. Monte Carlo Simulation

4.1. Monte Carlo Simulation

We will use Monte Carlo simulation to generate our basic data set. And we will generate two data sets based on the distribution hyperparameters as well as the rules on different kinds of fraud results and also the decision rules. Use one data set to contaminate the other one to generate different subsets. Then, we will measure the accuracy and prediction time of the prediction results provided for each training set, and then calculate the average estimate of all repetitions for each training set size. Finally, we hope to find the best predictive model for training sets of different sizes.

4.2. Data Generation

In this project, there are two datasets, each of size 1 million. The differences between two datasets are mainly the distribution hyperparameters as well as the rules on different kinds of fraud results and also the decision rules. The goal is to use one dataset to contaminate the other one so that we can see the fastest reaction from the different methods of prediction. To generate these two data sets, we will do the following operation.

We generate a basic data set randomly through a normal distribution and a binomial distribution. And then we are going to process the data.

Independent variables (binomial) :

Variable Name	Description
TransactionAmount	amount of purchase 0 = less than 1000; 1 = great than or equal to 1000
TransactionTime	transaction is not done during 12 - 5 A.M. 0 = during 12 – 5 A.M.; 1 = during other time
CreditLimit	amount of purchase is less than credit limit 0 = less than limit; 1 = beyond limit
Overdraft	buyer has not had overdraft during the past 6 months 0 = not had overdraft; 1 = had overdraft
ShoppedBefore	buyer has shopped there before 0 = shopped there before; 1 = first shop
HowMuch	amount of purchase if they shopped there before 0 = less than 1000; 1 = great than or equal to 1000
ShoppingFreq	buyer frequently purchases from this shop 0 = frequently purchase; 1 = rarely purchase
ShopProximite	shop is near where the buyer lives 0 = near; 1 = not near
UsualProximite	transaction happens near where they usually spend 0 = near; 1 = not near
PrevTranProx	transaction happens near their previous transactions 0 = near; 1 = not near
OnlineTransaction	transaction is not online 0 = at shop; 1 = online
BillPayment	transaction is a bill payment 0 = bill payment; 1 = not bill
PreAuthorized	payment is a pre-authorized 0 = pre-authorized; 1 = not pre-authorized

assumption about probability of occurrence approaching to the real-life probability:

We assume that the probability of a transaction is done where the buyer lives is 60%.

```
TransactionLocation <- rbinom(10000, 1, 0.4)
                                    # = 0 if the transaction is done where the buyer lives
```

Second, we assume that there are some dependencies between variables and that these data dependencies are represented during the data generation process. For example:

We assumed that if at least one of the following variables is equal to ‘1’, then we can know if the transaction is done domestically.

```
sim1_1M[(sim1_1M$TransactionLocation == 1 ||
          sim1_1M$ShoppedBefore == 1 ||
          sim1_1M$ShopProximite == 1 ||
          sim1_1M$UsualProximite  == 1 ||
          sim1_1M$PrevTranProx == 1), "Country"] <- 1
```

Third, to examine a fraud result, we defined a function that will decide if the transaction is a fraud or not based on other variables. we defined a fraud column that its value is obtained from a linear regression from other variables as follows:

```
sim1_1M[,"Fraud"] <- 3*sim1_1M$TransactionAmount+ 2*sim1_1M$TransactionLocation+ 3*sim1_1M$TransactionTime+
    4*sim1_1M$CreditLimit+ 2*sim1_1M$Overdraft+ 3*sim1_1M$Country+ sim1_1M$ShoppedBefore+ 2*sim1_1M$HowMuch+
    2*sim1_1M$ShoppingFreq+ sim1_1M$ShopProximite+ sim1_1M$UsualProximite+ sim1_1M$PrevTranProx+
    3*sim1_1M$OnlineTransaction+ sim1_1M$BillPayment+ sim1_1M$PreAuthorized
```

Then if the fraud amount for each transaction is below 6 then this is not a *fraud* (FraudResults=0). As you can see each of the variables has a weighted contribution to the decision we make.

To reduce the effect of linearity we also considered some other assumptions such as:

1. if we have no overdraft, the person lives in the same country, and they have shopped before in that store, then it is definitely a non-fraud transaction.
2. if it is an online transaction, and a bill payment and it is pre-authorized, then it is definitely a non-fraud transaction.
3. if this transaction is not close to where they shop usually, and it is not near where they live and it is not near where their other transactions happen, and it is not near their previous transaction, then it is definitely a fraud transaction.

Fourth, since some of the models, like logistic regression and random forest algorithm, used in this project required data classification, we decided to balance the data in order to increase the accuracy of the models. However, there is a large difference between the number of fraud and non-fraud transactions, which is a typical unbalanced data. Considering the non-balanced data is not easily divided, it is necessary to balance the data. In this project, in the second data set, we have a proportion of 15% fraud and 85% non-fraud cases in our basic data set. We will balance the data by random resampling at a proportion of 50% fraud and 50% non-fraud cases:

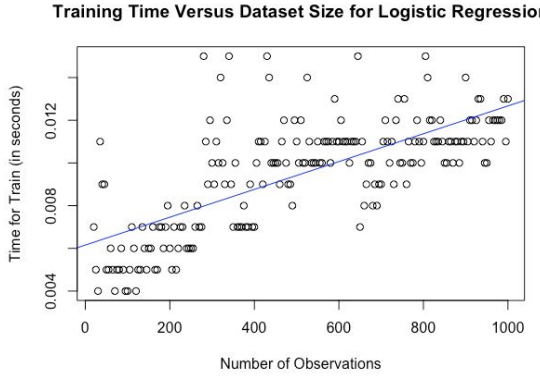
```
new_data = rbind(data_v2_full[sample(which(data_v2_full$FraudResults == 0),500000),],
                 data_v2_full[sample(which(data_v2_full$FraudResults == 1),500000),])
rows = sample(nrow(new_data))
new_data = new_data[rows,]
rownames(new_data) = NULL
```

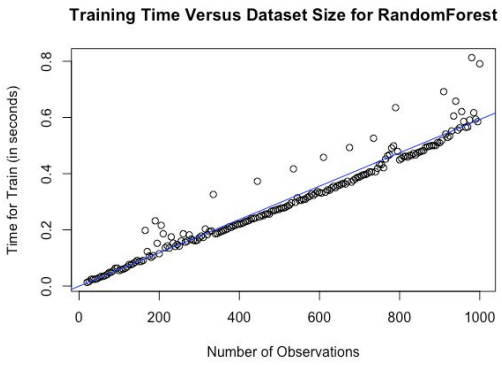
Fifth, while we have balanced the data, we cannot guarantee the accuracy of the model. In order to compare the models, we use accuracy since it is a binary target variable.

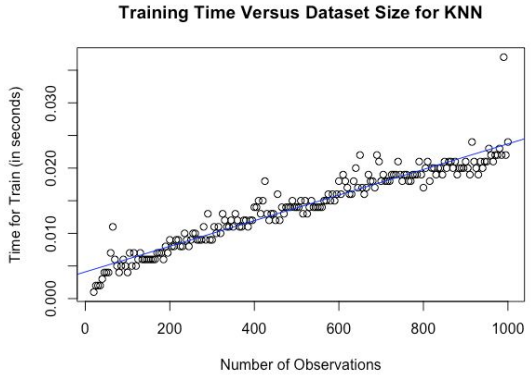
4.3. Results

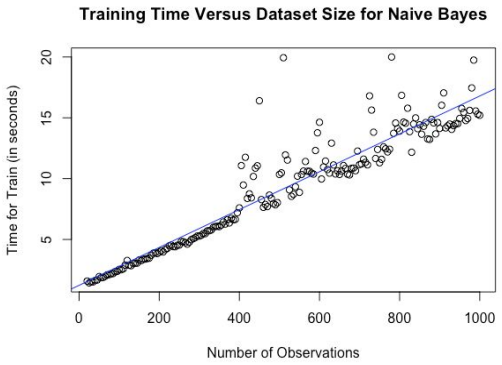
4.3.1. Training Time

As we see in the plots, generally when we increase the number of observations, we expect more time to train the model. However, the naive Bayes' and logistic regression are not as consistent as random forest and K-nearest neighbor. In naive Bayes' we see that after 400 observations around 420 observations there is a rise to 13 seconds which is not consistent and comes back to what can be predicted at around 440 observations. These rises are periodically happening throughout increasing the number of observations and is also depicted in the random forest. The difference is for logistic regression it has high variance and it seems there are categories of time that some observation sizes are falling into those categories (if you draw horizontal lines, you detect the categories).



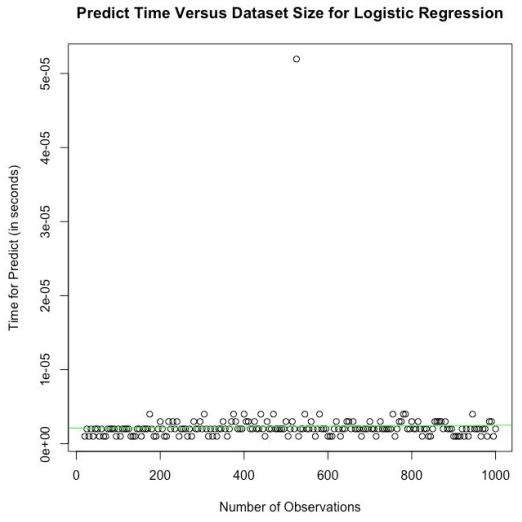


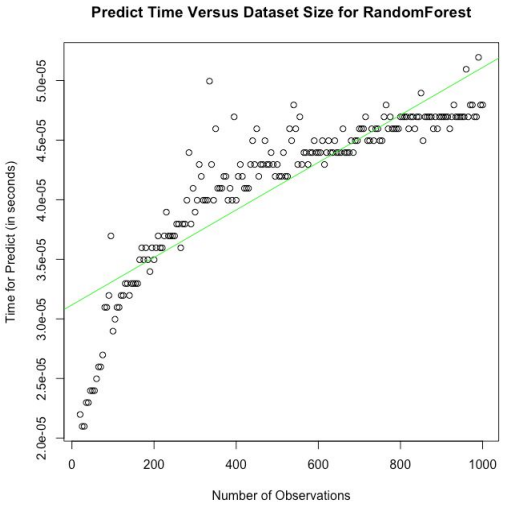


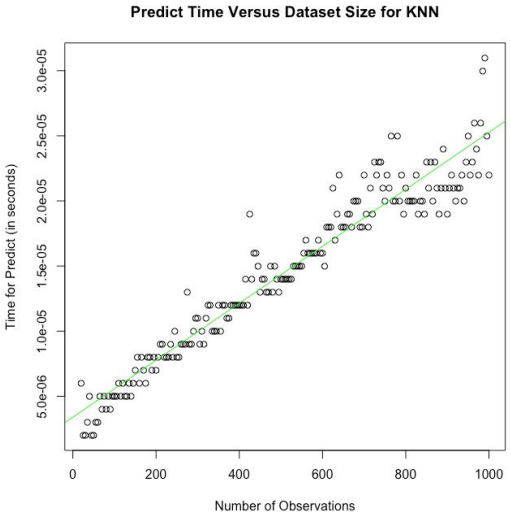


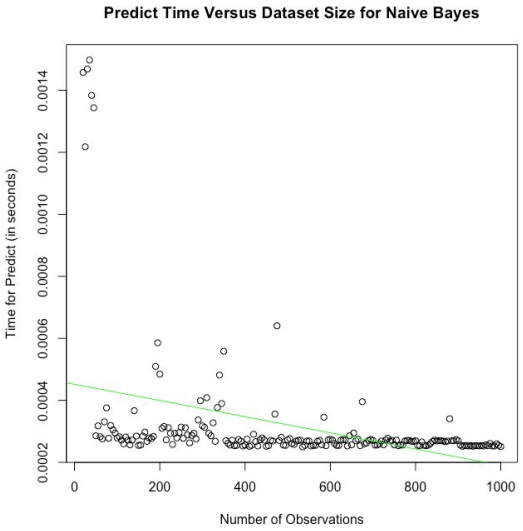
4.3.2. Predict Time

Prediction time for logistic regression remains consistent as we increase the number of observations. For Naive Bayes' is still the same behavior as regression, however, as you increase up to 400 and more, there is a consistency and density in the plot. For KNN it looks like it is a polynomial graph which after around 720 of observations, we again see the time to predict for observations fall into categories. Interestingly, the random forests seem to have a logarithmic look, which is probably due to the tree structures and depth of the trees.



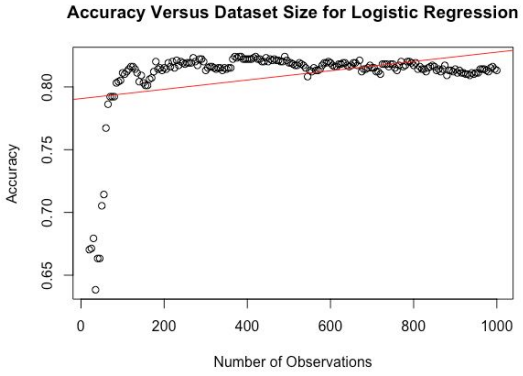


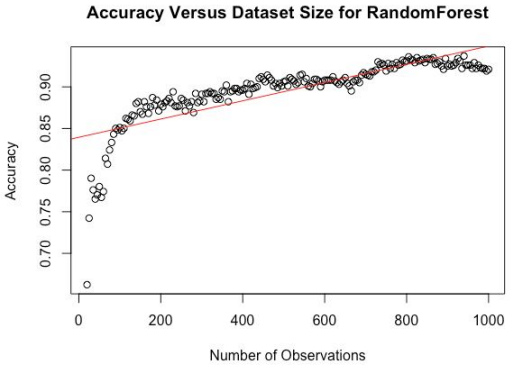


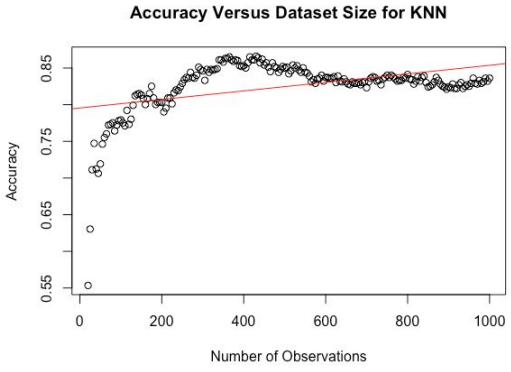


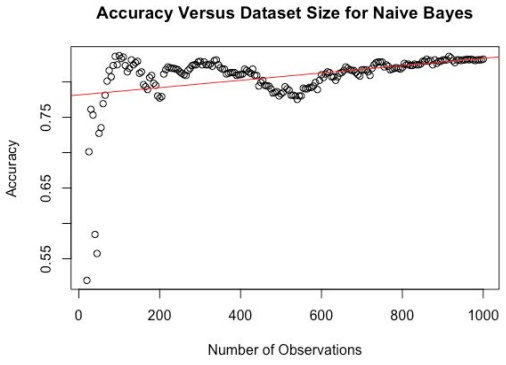
4.3.3. Accuracy

For logistic regression we see that the highest accuracy happens around 400 observations and it will remain the same after that. For randomforest, we see as much as you feed the algorithm, it gives you higher accuracy. Just around 640 of observations we see a little fall off the accuracy level. In KNN, we see that the maximum accuracy happens around 380-420. After that it only gets lower to around 0.826 and stays the same despite increasing the dataset size. For naive Bayes' is somehow again close to logistic regression; it does not require a lot of data to hit the highest accuracy, even 100 observations is enough for NB.





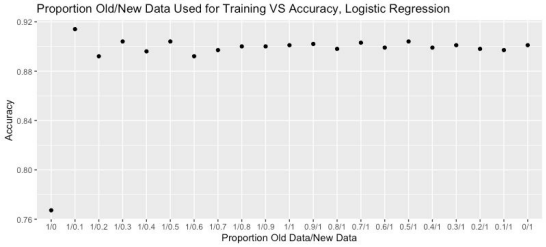


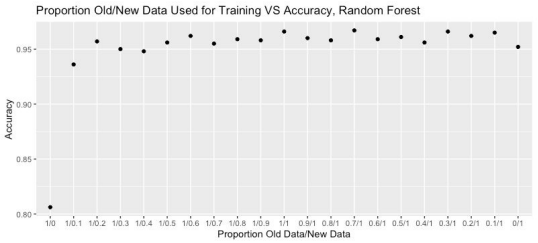


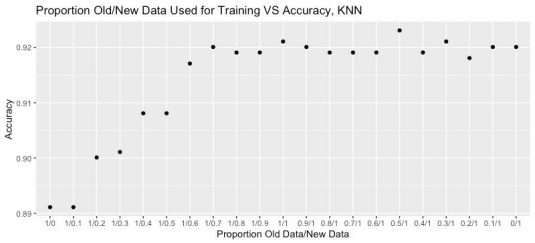
4.3.4. Changing Data Over Time

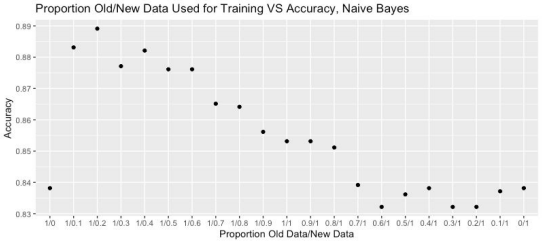
As described in the Model Selected, we have mainly used methods of K-nearest neighbor, logistic

regression, random forests, and naive bayes' algorithms. We want to see which one of these algorithms react faster to the dataset contamination. Reacting here means in terms of level of accuracy, prediction and training time. We will evaluate each separately.









Another avenue to explore is the time required for a model to detect changes to the nature of the data. Upon further examination, it is not a question of time needed for the model to reflect a change in data, rather how many observations following the new nature is required in the training dataset. Thus, we examine the accuracy rates of the four models with varying compositions. On the left we begin with a full dataset for the outdated data (in this instance, 9000 observations) and no new data. Moving right, the percentage of new data included in the training set is increased by increments of 10% until both datasets are included in full at the center. Lastly, towards the end the proportion of old data is reduced by increments of 10% until only the new dataset is used for training. Generating predictions for a test data from the new data, we see that accuracy is immediately low for all models. For Logistic Regression and Random Forest, including just 10% of the new dataset will allow for high accuracy rates, which remain fairly stable for the remainder of the data composition shifts. KNN takes a longer time to regain accuracy rates, only doing so when 70% of the new dataset is included in the training set. Lastly, Naive Bayes displays a strange behaviour of a quick jump is accuracy at 10% of new data included, but actually begins to decrease as time goes on until coming to a plateau at all of the new data and 60% of the old data included.

5. Conclusion

Model	**Training Time**	**Prediction Time**	**Accuracy**
Logistic Regression	Fastest, High Variance	Does not increase w.r.t observation size, Low Variance	Increases quickly, Low Variance
Random Forest	Slower, Medium Variance	Increases non-linearly w.r.t observation size, Medium Variance	Increases slowly, Low Variance
KNN	Faster, Medium Variance	Increase linearly w.r.t observation size,	Increases slowly, High Variance

		Medium Variance	
Naive Bayes	Slowest, Mixed Variance	Does not increase w.r.t observation size, Low-Medium Variance	Increases quickly, High Variance

Logistic Regression algorithm

1. The predictive time performance of logistic regression is great. Not only is the prediction time the shortest among the models used in this project, but it also remains stable as the size of the data set increases. It fully proves its advantages of simple calculation.
2. The accuracy of logistic regression performed well in the early stage. As the size of the data set increases, the accuracy of logistic regression gradually lags behind the accuracy of the K-nearest-neighbours classification algorithm and the random forest algorithm. But it still maintained a high accuracy.

Random forest algorithm

From the results, we can prove the following points.

1. The accuracy of the random forest algorithm is the best among the models used in this project. This is also as we predicted in the model selected.
2. When the sample size is small, the accuracy and prediction time of the random forest algorithm are not good, and it deviates from the regression line more obviously than other models. This also proves that the random forest algorithm does not perform well in data sets with small size.
3. The prediction time of the random forest algorithm is relatively slow, and it shows an obvious trend of increasing with the increase of sample size.

K-nearest-neighbours classification algorithm

1. The accuracy of the K-nearest-neighbours classification algorithm is good. The accuracy is high, second only to the random forest algorithm. This proves its high precision characteristics.
2. Although its prediction speed surprised us, it is second only to the prediction speed of logistic regression. However, there is a significant difference in the prediction time between it and logistic regression, and the prediction time is still relatively long.
3. Its prediction time also shows a trend of increasing as the sample size increases.

Naive bayes algorithm

It can be seen from figures that the speed of naive bayes' algorithm is much slower than that of other models. And the accuracy is also the lowest among the models we use. This is strange because the logic of naive bayes' algorithm is relatively simple to compute. We believe that this result in this project is due to the following reasons.

1. Naive bayes' algorithm cannot learn the interactions between features. However, due to the obvious correlation between some variables in this project, the independence assumption of naive bayes algorithm cannot be satisfied. Therefore, the prediction process of naive bayes' algorithm is affected, resulting in the prediction time anomalies and the relatively low accuracy.
2. Naive bayes' algorithm is greatly influenced by prior probabilities, and the prior probabilities often depend on assumptions. Since no specific characteristics are given in this project, the problem of hypothetical conditions causes abnormalities in the naive bayes algorithm to predict time.

Through a comprehensive comparison of the above analysis, the logistic regression model is considered to perform best in this project. First, its forecast time is stable. In real life, since data exists in the form of streaming data, the stability of the prediction time as the size of the data set increases is very important for fraud prediction. In addition, the accuracy of the logistic regression model has been maintained at a very high level. Therefore, in this project, we believe the logistic regression model is most suitable for the prediction of credit card fraud.

References

MerchantSavvy. *The Continued Rise Of Payment & Card Fraud.* Available at https://www.merchantsavvy.co.uk/payment-fraud-statistics/?matchtype=b&device=c&utm_keyword=%2Bcard%20%2Bfraud%20%2Bdata&utm_source=google&utm_medium=cpc&gclid=CjwKCAiA4o79BRBvEiwAjteoYJKNUprSeTNy3M0w8dpA9k0mLptPUr0O3pJir5SSFlS1Rphfqa56HxoCJ88QAvD_BwE [Last accessed November 2020]