1. Data Engineering

1.1 Drop Columns to ignore irrelevant information

Trans_date_trans_time: I use UNIX Time to extract information later, which is overlapped with this column, so I drop it.

Lat and long: The city or state variable can already aggregate the information of location of credit card holder. The coordinates are numerical value which the absolute value is meanless.

Merchant Name: Perhaps there is little correlation between some certain merchants and fraud. But the unique value of merchants is too much, which will cause curse of dimension and increase the load of model.

Street: There are too many streets and the street is low-level information.

Zip: Same as above reason

City: The state variable is higher-level information of location. Ignore City

First: There are many different and overlapped first name. The first name can not indicate the fraud.

Last: Same as reason above.

Trans_num: it is an identification number. Irrelevant to fraud.

1.2 Compute age period from date of birth

A str of birthdate is useless. We can compute the real age from the birthdate. But a small difference of age values cannot make sense. A age interval with length = 10 is more pursuable.

1.3 Map cc_num to its length

The credit card number is just a i.d. We can use its length to represent it to reduce dimensions

1.4 Map UNIX Time to time information

From UNIX time, I extract year, month, and time period. Time period has four unique values: Earlymorning is 0 a.m. to 6 a.m., Morning is 6.am to 12 p.m. , Afternoon is 12 p.m. to 18 p.m., and Night is 18 p.m. to 0 a.m.

1.5 Map job to numerical value

There are too many unique job values, which may cause curse of dimensions using one-hot encoding. According to intuition, the important of something should inverse to its frequency. Thus, I count the time of appearance of each job value, and map job string to 1/(appearance times / N), where N is total numbe of training set.

1.6 One-hot Encoding

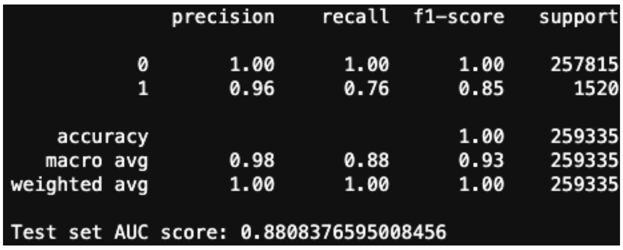
For the rest of str variables, use One-Hot Encoding to map str to binary variables.

2. Model

Random Forest is Used as model Since it can greatly reduce the variance. It is very suitable for the anomaly detection problem, where labels are highly imbalance.

The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble is comprised of a data sample drawn from a training set with replacement, called the bootstrap sample. Of that training sample, one-third of it is set aside as test data, known as the out-of-bag (oob) sample, which we'll come back to later. Another instance of randomness is then injected through feature bagging, adding more diversity to the dataset and reducing the correlation among decision trees. Depending on the type of problem, the determination of the prediction will vary. For a regression task, the individual decision trees will be averaged, and for a classification task, a majority vote—i.e. the most frequent categorical variable—will yield the predicted class. Finally, the oob sample is then used for cross-validation, finalizing that prediction.

In the coding, the number of estimators is set to 200. Training data and testing data are split using proportion of 8 : 2. The classification report on validation set is:



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    257815
           1       0.96      0.76      0.85      1520

    accuracy                           1.00    259335
   macro avg       0.98      0.88      0.93    259335
weighted avg       1.00      1.00      1.00    259335

Test set AUC score: 0.8808376595008456
```

And AUC Score is 0.88.

Then, I train the random forest using full dataset and save in local file.