Data Analytics

For this part, we will deploy various fraud detection models to predict whether a given transaction is fraud or non-fraud with the 7 attributes (all the attributes provided by the dataset minus the Fraud label) as inputs. We will comment on the strengths & weaknesses of each classifier we use and evaluate their suitableness to our dataset based on the prediction performance. We will also make inferences on each variable’s importance and give detailed graphical descriptions on how they play the role in deciding the outcome.

1. SVM

The first model we consider using is Support vector machine model (SVM). It has the merit of being able to capture the non-linear structure of the dataset by kernel transformation, as well as being resilient to outliers and correlated features, which are particularly preferable under this case as the EDA showed there’re certain extreme observations existing in those continuous attributes, and the distance from home and whether the retailer is a repeated tend to be related to one another.

Before building the SVM we rescale each attribute to have a mean 0 and standard deviation 1. Besides, we noticed that the time required to build the SVM model is proportional to the cost parameter C, hence we chose a small value of C (0.01) due to time complexity concerns. We use the gaussian kernel to do the transformation and the gamma is set to 0.5 by cross validation.

We build the SVM under two scenarios: the first one is to use the whole dataset, corresponding to the two figures below:

Chart, treemap chart

Description automatically generatedChart, treemap chart

Description automatically generated

*.  Precision:0.996 Recall:0.928 Accuracy:0.993 F1:0.961 Precision:0.997 Recall:0.892 Accuracy:0.99 F1:0.941*

And the second is to use the down-sampled dataset after balancing the fraud and non-fraud cases, corresponding to the figure below:

Chart, treemap chart

Description automatically generated

*Precison:0.747 Recall:0.995 Accuracy:0.97 F1:0.853*

For each scenario we consider two different training set proportions: 0.8 vs 0.4, in order to see whether the model performance is sensitive to the number of trainings instances.

From the former two figures, we can see that the SVM performs well on the whole dataset with training-set percentage of 80%, whereas the recall still seems to have space for improvement. As the training percentage is tuned down to 40%, its performance also gets slightly worse. From the above figure, we can see the trade-off between label balancing and down-sampling: Since the proportion of fraud cases is elevated, the model does achieve a higher recall rate meaning it becomes more capable of capturing those fraud transactions. On the other hand, as the effective sample size gets smaller after down-sampling, the overall performance gets poorer. In particular, there is a sharp drop in precision value, meaning that the model now generates too many false positive predictions.

A significant drawback of SVM, which has also been mentioned above, is the heavy computational cost, which limits the scopes of fine-tuning hence makes it hard for us the find the optimal value of hyperparameters. Besides, SVM assumes that the two transaction classes can be divided by a hyperplane, which heavily depends on the structure of the dataset and may not hold true.

1. KNN

In our case, the dataset is of large-scale with only a few attributes, and if the structure tends to be clustered instead of being separable, the k-nearest-Neighbor (KNN) is supposed to be a more suitable approach.

To build KNN, we also do the rescaling first. We consider balancing to be a must for KNN to achieve desired performance, so we only build our model on the down-sampled dataset. By repeatedly try all acceptable k values and check the recall rates, we find the best k as 5 for this dataset, and the figures below shows the results of KNN evaluated on the test set:

Chart, treemap chart

Description automatically generated

*Precision: 0.999 Recall: 1 Accuracy:1 F1:1*

And on the whole set:

Chart, treemap chart

Description automatically generated*.* Chart, treemap chart

Description automatically generated

*Precison:0.994 Recall:1 Accuracy:0.999 F1:0.997 Precision: 0.994 Recall:1 Accuracy:0.999 F1:0.997*

The KNN model performs almost perfectly on the test set with only 13 cases misclassified. Its performance drops down a little bit while evaluating on the whole dataset, but still obtain full recall and all the other indicators close to 1. Even if we drag down the training set proportion to 40%, the performance still maintains at roughly the same quality except only a few more false positives, which indicates that KNN can learn the task quite well even with a small percentage of training instances.

The goal of detecting fraud cases has been achieved, we then need to consider detailed attributes analytics and some further situations like getting new records containing new features.

1. Tree-based Models

As we know, fraudulent will never stop even though we have finished a good job at fraud detection, and there will always be new fraud techniques created as fraudulent prevention methods developing. In order to extract features from the new fraud records rapidly, we need to deploy some models that don’t rely on the size of the dataset. Thus, one widely used tree-based algorithm, random forest, is good to use.

According to the characteristics of random forest, we don’t need to do data pre-processing and even don’t need to do many configurations to the model. To validate our idea, we still trained this model on different proportions of the dataset, and the results are amazing as well.

Chart

Description automatically generated A picture containing chart

Description automatically generated

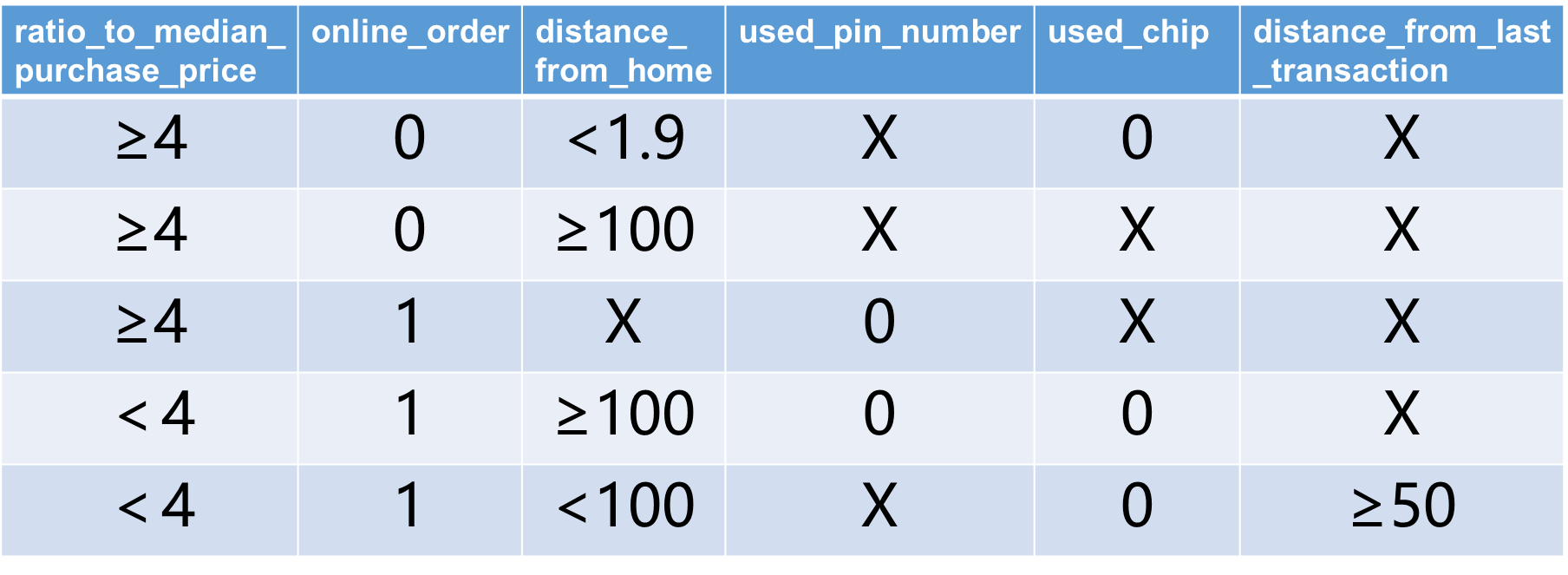
*Precision: 0.999 Recall: 1 Accuracy:1 F1:1 Precison:0.994 Recall:1 Accuracy:0.999 F1:0.997*

Although we cannot take the place of KNN model, the advantage of fast training speed of random forest makes it be able to be used as an auxiliary model to help us improve the prediction effect. The other purpose of using random forest is to check the attributes importance to verify the information we get from EDA process and to make further analysis.

Chart, funnel chart

Description automatically generated

Another model we used to help us analyze the results is decision tree, which is highly interpretable. We can simplify the whole model to specific rules by deleting the nodes leading to non-fraud records and constructing paths from root to fraud nodes. The five rules we get are shown below:



Summarize the five rules to fraud cases, they should be merchant abuse, stealing, deception, leaked account and leaked pin number. Then we can use these rules to give our customers some advices and to add some restrictions to our system to decrease the fraud rate.