Credit Card Fraud Detection

Machine Learning Project- Phase 3

Malak Gaballa
900201683
Data Science
The American University in Cairo
malakhgaballa@aucegypt.edu

Masa Tantawy
900201312
Data Science
The American University in Cairo
masatantawy@aucegypt.edu

Introduction

In phase three, a pilot study of multiple supervised machine learning models were implemented on the prepared datasets. Exploratory data analysis, data preparation, and feature engineering were all done during phase 2, which is briefly described below. The prepared dataset was split into training and testing sets to facilitate machine learning implementation. Each machine learning model was trained and tested on the same datasets to allow reliable comparison and their performance were all evaluated using the same criteria, recall and specificity, to determine which one is the best and most suitable to the problem. A detailed description of the reason the choosing the evaluation metrics, the performance of each model, and the chosen final model are discussed in this phase.

Exploratory Data Analysis and Data Preparation (Phase 2 Revisited)

During this phase, the data was prepared for building a machine learning model. The initial dataset consists of simulated credit card transactions over the span of 2 years, from the 1st of January 2019 to the 31st of December 2020. It was initially divided into training and testing sets, which in total consist of 1,852,394 instances and 22 features, one of which is a binary label indicating whether each transaction is fraudulent (1) or legitimate (0). There are neither missing values nor duplicated instances in the dataset. The importance of each feature, statistical distributions, and feature correlations and relationships were examined in detail.

The features *credit card number, unix time of the transaction, transaction number, and the ZIP code of the customer* were dropped from the dataset due to their sparsity and high column variance. Likewise, the name of the merchant and of the customer, features *merchant, first* and *last*, were extremely sparse in addition to them being not beneficial or relevant to credit card fraud detection thus they were removed from the dataset.

Next, datetime features, which are *trans_date_trans_time* and *date of birth of the customer*, had to be transformed in order to facilitate the machine learning process. These were converted into datetime; the day, month, year, hour, and minute at which the transaction took place were extracted from the feature *trans_date_trans_time* as five new features. As for the variable dob, it was transformed into *age* then rounded down in order to accurately display the credit card holder's age. The aforementioned extracted features contained no outliers, but age had 0.0235% outliers; according to a threshold 0.03%, these instances were removed from the dataset. To ensure that all the numeric features in the dataset follow the same distribution, all the numeric features were similarly normalized.

For categorical features, they needed to be encoded. First, *gender* was label encoded: 1 for female and 0 for male and the feature *category* which represented the category of the transaction was one-hot encoded as it contained 14 classes. Similarly, the feature *state* was one-hot encoded since it may be an important indicator to the location of the customer; the dataset contained 51 states, all with very close number of instances, except for the states Rhode Island (RI) and Delaware (DE) which contained 745 and 9 instances respectively, which is considerably lower than all the other states, so they were merged together in a new feature *state_other*.

Other categorical features were more challenging to deal with since they contained a very large number of classes, indicating that one-hot encoding is not the optimal solution. Other encoding techniques were tried, such as label

All the tables mentioned can be found at the appendix at the end of this document.

encoding, frequency encoding and leave-one out encoding, yet these all resulted in collision hence could not be used. The feature *job* contained 497 different jobs for the customers; these were grouped into 9 categories depending on the industry: engineering and technical jobs, healthcare, creative and design jobs, legal jobs, education, business, media, agriculture, and social services. Each of these categories contained a similar number of jobs and were on-hot encoded into 9 new columns instead of the feature *job.* The features city and street were extremely sparse and they were not also strongly relevant to the classification thus were dropped.

Finally, the numeric feature *dist_km* was added representing the distance between the customer and the merchant using their latitudes and longitudes. After computing this distance, the latitude and longitude of the customer and the latitude and longitude of the merchant were no longer needed, so they were removed from the dataset. This new feature was also normalized to follow the same distribution as the rest of the numeric variables. Now, the dataset is ready to use. The final dataset contains 84 features, one of which is a binary indicator of fraudulent transactions.

Experimental Setup and Parameters of Choice

Now that the dataset is revamped, it had to be split into training and testing sets in order to apply each supervised learning model. The splitting was done using the *train_test_split* module in the *sklearn.model_selection* package where 75% of the dataset was training, 1388969 instances, and the remaining 25% were testing, 462990 instances. Since the splitting is random, these splits were saved in csv files so that different models can be completed in different notebook simultaneously.

After splitting the data, a function called *evaluate_perf()* was created in order to evaluate each machine learning algorithm by displaying the performance measure for each model; this function's arguments were the true and predicted values of the label. Accuracy and error rate are not suitable in this case, for the data is extremely unbalanced with the majority of the instances being non-fraudulent. As a result, all models will yield very high accuracy and very low error rates regardless of their performance. To allow evaluation of the actual performance of a model, the recall (true positive rate) and specificity (true negative rate) are considered the main metrics that will be used in the case of detecting fraudulent transactions. This is because more threat is posed if a fraudulent transaction is misclassified as a normal transaction (false negative) than if a legitimate transaction is misclassified as fraudulent (false positive) thus the main aim is to minimize the number of false negatives, which is identifying fraudulent transactions as non-fraudulent. Both the specificity and the recall (of class 1 which is fraudulent transaction) allow for this to be monitored and the best performing model is the one that maximizes both measures.

Algorithms

k-Nearest Neighbors

k-Nearest Neighbors was the first model implemented due to its simplicity and its ability to classify large data from any distribution. Given the large size of the dataset, a for loop was constructed to evaluate the performance of the classifier with different odd values of *k*. Since the rule of thumb is to choose a *k* that is less than the square root of the size of the dataset, the parameter K was tried from 1 to 1177, which is the square root of the size of the training set. Although there was no large variability in the specificity from 0.99 for all the values of k, a noticeable observation is that the recall started at 0.59 when *k=1* and decreased as *k* increased. Starting from *k=229* onwards, recall fell to zero. This can be explained by the unbalance of the label since the majority of the transactions are non-fraudulent therefore as k increases, the majority vote will tend to be non-fraudulent (*is_fraud* = 0) rather than fraudulent (*is_fraud* = 1). The value of *k* that returned the highest recall was 1 which makes the model ungeneralizable and very sensitive to noise as the closest instance is used for classification.

Linear and Logistic Regression

Next, a linear regression model was fitted to the training data and the threshold of 0.5 was chosen to classify all the predicted values that are less than 0.5 as non-fraudulent (*is_fraud* =0) and otherwise fraudulent (*is_fraud* =1). Linear regression fits a mathematical equation to predict the class that minimizes the error with a coefficient for each feature, so it is able to work with both numeric and categorical features. The model returned a recall value of 0 despite a very high specificity of almost 1 which means that all the fraudulent transactions were misclassified as non-fraudulent. This indicates that the model is not a good fit for the data since it was unable to detect any of the fraudulent transactions. One logical reason for this is that

the data is not linearly separable which makes this model fail.

Likewise, a logistic regression model was fit to the training set but, even though it returns the probability that each instance belongs to a class and does not just classify each instance, the recall was also very low. All the model parameters were set to their default values, for example the weights of both classes were equal and the maximum number of iterations to find the weights were 100. Even though the model had a specificity of almost 1, the recall was 0.007 which is extremely low and indicates the model's inability to identify fraudulent transactions. This explains why linear and logistic regression models are not a good fit for the data unlike in previous research discussed in phase 1 where logistic regression appeared to be among the best-performing models.This can be explained by the fact that the label has no linear correlation with the features which consequently affects the odds of success and leads to the model's poor performance.

Decision Trees

Decision trees was the most widespread approach in the literature review thus was implemented as one of the models in this phase. This is due to its simplicity to understand and implement as the tree can be drawn, clarifying the structure, such as which nodes are leaf nodes. However, this technique depends on binning numeric features into classes/ bins, so it ignores the importance of every unit change in a numeric feature and its effect on classification.

The module sickit-learn was used, which contained an optimized version of CART (classification and regression trees) used for classification. Since the data contained several numerical features with categorical features, this model automatically binned them to convert them to categorical features to maximise the information gain. All the parameters of the model were left to default, such as no maximum depth for the tree and equal weight for all the classes, except for the parameter criterion which is an attribute selection measure; a criterion is chosen from *gini* (default)*, entropy* and *log_loss*. These three criteria were implemented each in a model to evaluate their performance and compare any differences in performance. All the 3 models have a very close high specificity (almost 1), but they differed in recall.

The Gini index is an impurity measure which tries to minimize the probability of misclassification depending on the varying probabilities of each class. With this selection of criteria in the model, the recall measure was 0.83 which was high relative to the aforementioned models. This indicates that the model is able to classify fraudulent transactions correctly 83% of the time. As for the log-loss, it was used to fit another decision tree model. Log loss is known as cross-entropy loss which is a function used in multinomial logistic regression. The recall value was 0.84 for this model. Lastly, a decision tree model based on entropy was built. For decision trees, low entropy values are preferred since they indicate that the label is not random across the features and that the majority of instances belong to a certain class. Using entropy, the recall increased to 0.85 which is the highest value out of all the 3 decision tree models. Therefore, the best model is the entropy measure followed by log loss then gini according to the recall values.

Random Forests

Since the performance of decision trees appeared to exceed expectations, random forests were the following algorithm as it is known for its efficiency and ability to deal with high dimensional datasets that contain both categorical and numerical features. That being said, training of this algorithm is very slow due it the high computational power required as a result of numerous trees being built simultaneously before being combined for classification. Similar to decision trees, *entropy* and *log-loss* criterias were used to measure attribute selection in two different models; *gini* was excluded due to its weak performance in decision trees. A new parameter *n_estimators*, the preferred number of trees in the forest structure, was varied in a loop to isolate the highest performing model for each criterion. This parameter was tried for each number in the range 50 to 100 in a loop. For both *entropy* and *log_loss*, insignificant of the number of trees chosen, the specificity was always above 0.99. There were minor differences in the recall values.

For *entropy*, recall fluctuated between 0.73 and 0.75 as the number of estimators (trees) increased. The model that yielded the highest recall was when the number of trees was set to 71 since the recall was 0.75. As for the *log_loss*, recall was always 0.74 for all the values of the parameter *n_estimators*. This indicates that both criteria yield very close performance measures, and that the number of trees

in the random forest slightly affect the classification. Thus, the random forest model that used *entropy* as its measure and depended on 71 trees displayed the best performance, for it was able to correctly distinguish fraudulent transactions 75% of the time.

Naïve Bayes and Bayesian Networks

The gaussian naive bayes was used to fit a different model to the dataset. This model assumes that the classes follow a normal distribution, then uses their mean and standard deviation to classify each transaction as either fraudulent or legitimate from the probabilities of its details. Even if the data may not follow a normal distribution, the large sample size means that it converges to a gaussian distribution by the central limit theorem. This model yielded a specificity score of 0.90 and a recall of 0.58. Although the recall is not very low, it is not close to that of decision trees. This is because it does not perfectly fit the shape of the data due to the presence of a large number of numeric features, namely 9 numeric columns, compared to less initial categorical features before encoding, making the performance of naive bayes not optimal. It also neglects the dependency structure of the data as it assumes all the features in the dataset are unrelated and independent from each other which is not necessarily the case with credit card fraud detection. Accordingly, Bayesian networks were not built. This is for multiple reasons, such as that decision trees are better to use and our lack of domain knowledge which is vital to construct the network and select their dependencies.

Perceptron Learning and Neural Networks

Another approach tried was using perceptron learning. A linear perceptron classifier was built using the default parameters of *sklearn.linear_model* such as a maximum number of 1000 epochs (iterations over the training data), the data to be shuffled after each epoch and equal weights for all classes. Since the data is not linearly separable, the performance measures were low compared to the other algorithms; the specificity and recall of this model were approximately 1 and 0.38 respectively.

A multi-layer perceptron classifier (artificial neural networks) was also trained and tested on the data since this algorithm works on large samples or datasets which is the case with the transactions data. A combination of different solvers, such as stochastic gradient descent and the module's default stochastic gradient descent optimizer *adam* as well as different activation functions like *relu, tanh* and *logistic* were all fit in separate models. It is worth noting that the usage of stochastic gradient descent as the technique to learn the weights is not time efficient since the weights are updated after each instance and given the large sample size, weights were updated a massive number of times. Using the default solver *adam* and the *logistic* activation function, the specificity was almost 1 and the recall was 0.78 which surpassed the performance of all the other parameter combinations.

Conclusion

After trying 8 different machine learning algorithms with variations in their parameters, they were all evaluated based on the same performance measures, specificity and recall, after being trained on the same training set and tested on the same testing set. The top 3 best performing models were decision trees, neural networks, and random forests. It is worth noting that the specificity was very high and relatively close for these 3 models as 0.99 which is almost 1.

Random forests using the entropy measure and 71 trees was among the best performing models due to its high recall and being able to rightly classify 75% of fraudulent transactions. Multi-layer artificial neural networks also appeared to create a high functioning credit card fraud detection system; the model that used the default solver *adam* and the *logistic* activation function outperformed Random Forests by reaching a recall value of 0.78. This is because both these models are able to work with a large dataset size and take into account both categorical and numeric features without neglecting the importance of any. Likewise, both these techniques do not fail when working with extremely unbalanced data, which is the case with credit card transactions due to the very small number of fraudulent transactions compared to legitimate ones.

As for decision trees, they were the superior approach since this algorithm returned the highest recall values among its different criterias of building a tree. The 3 criteria evaluated were *entropy*, *gini*, and *log-loss*; the specificity for all 3 were extremely high, similar to random forests and neural networks, but the recall indicated that *entropy* was the best criterion followed by *log-loss* then *gini*. This is because it had the highest recall out of all criteria and of all the models fitted to the dataset (0.85). This algorithm performs the best because CART matches the shape of the dataset well as it takes into consideration both categorical and

numeric features, giving equal importance to all the features.

Based on these results, decision trees based on entropy is the algorithm that will be chosen to continue with due to its high performance and fit to the dataset shape and problem. This algorithm is also relatively easy to understand as the structure of the tree can be visualized, making it uncomplicated to be implemented at financial institutions and by those who do not have a strong technical background. After settling on this model, there will be multiple attempts to improve its performance. This can be done via examining the structure of the tree, tree pruning, trying to vary other parameters of the tree than criterion, and checking the automatic binning of numeric features and manually changing them to optimize classification.

References

[1]
Credit Card Transactions Fraud Detection Dataset. *Kaggle*. Retrieved February 13, 2022 from https://www.kaggle.com/datasets/kartik2112/fraud-detection?resource=download