Algorithms

Our data has 7 features and 1 target variable. The target variable is split into a binary classification of fraud or not fraud. Out of the 7 features, 3 are numerical and continuous, and 4 of them are categorical (but also numerical). Based on our problem statement, we are looking to classify credit card transactions in our data as either fraud or not fraud. Taking into account our goal and the constraints of our data, we thought it would be best to use 3 different classification algorithms: k-Nearest Neighbors, logistic regression, and support vector machine.

kNN is the simplest classification algorithm, and is fairly easy to implement with the help of scikit-learn's modules. kNN uses the entire sampled dataset as model representation, and makes predictions straight off of the training data that you split. KNN works by looking at the training data and finding the "nearest" neighbors to each record. Then, it uses those neighbors to predict the label for the new data. It is a straightforward classification algorithm that we can use to make predictions on our dataset and even evaluate the strength of the features we use based on the accuracy of the predictions.

The next algorithm we will use is logistic regression. Logistic regression has a few assumptions that must be fulfilled before the model can be constructed. First, it requires the target variable to be binary. Second, only meaningful variables should be included. Third, logistic regression requires a large sample size. We fulfill all three of these conditions, and can move forward with using logistic regression as our second classification algorithm.

Finally, we will construct an SVM as our last classification algorithm. SVMs are utilized mostly when evaluating in high dimensional spaces, but even for our dataset with 7 features, it can be an effective classification algorithm. However, when choosing kernel functions for the SVM, we must make sure to avoid over-fitting. We can sample our data, as to where we have a balance between fraud and non-fraud data, so the SVM can be more effective.

K-Nearest Neighbor

We chose KNN because it is a simple algorithm that can be easily applied to this problem. The model will be trained on a dataset of credit card transactions. The labels will be whether or not the transaction was fraudulent. The model will be evaluated by seeing how many frauds it correctly detects. In terms of train/test split, while k-fold cross validation can be done to optimize the kNN further with more nuanced splitting of the training and testing data, we chose to simply split 30% of the data for testing and the rest for training. We are going to use the recall of fraud predictions to measure the accuracy of the model, because the consequences of a false negative are much worse than the consequences of a false positive. We are alright with flagging non-fraud as fraud, but letting fraud get through unflagged defeats the purpose of the model. Therefore we optimize for highest recall to minimize the amount of false negatives that get past our knn. The parameters that will be tuned are the number of neighbors to use (k) and the distance metric. We have already completed the kNN algorithm to classify our data, and we found the best k value to be 8. When evaluated on the test set, we observed a recall of 0.99 for labeling fraud. This indicates that the model is high-accuracy.

Logistic Regression

We will split the data into training and testing sets, with 70% of the data in the training set and 30% in the testing set. We will then train a logistic regression model on the training set and evaluate its accuracy on the testing set. One key component is RFE (reverse feature elimination) that is similar to a genetic algorithm. This algorithm removes features that add noise to the data while maintaining those that add signal. In our implementation, we are not planning to use RFE since we only have 7 features and should try and evaluate all of them. We are planning to use StatsModel.API to perform logistic regression on features to allow us to manually eliminate features that have a low P-value. Then, for the actual model, we will use the more traditional scikit-learn. In the model, the parameters that we will tune are the regularization parameter C, and the solver (algorithm) used to optimize the logistic regression model. For this model as well, we will be using recall of the fraud predictions to measure the accuracy of the model. We expect high accuracy from this model as well, since we expect fraud data to be linearly separable from non-fraudulent data. Our plots in the EDA back up this prediction as well.

Support Vector Machine

We will do a simple 70-30 split of the labeled data into training and testing sets. Our dataset already has all categorical variables (including the class label) as numerical values 0 or 1, so that encoding step is already done for us. We will perform scaling on the quantitative variables, and then train an SVM model with a linear kernel on the training set. We expect a linear kernel to work best with a binary classification problem. The parameters that we will tune are the regularization parameter C, and the kernel function. We will generate a classification report and, as before, use recall as the metric to evaluate model performance. We expect the model to have a high accuracy, since we expect the fraud data to be linearly separable from the non-fraud data.