Don’t Get Kicked Solution

February 5, 2018

1 Inspecting, Cleaning and Preparation of the Data

Upon load the data, I performed a preliminary inspection of the data available. First, I checked the shape of both the training and testing set. It can be see that the shapes are fairly similar with the only difference (aside from the amount of data) is that there's a column missing in the testing set which corresponds to the result of the testing set.

From the head and tail of the training data, it appears that PRIMEUNIT and AUCGUART are many entries. These columns were found to consist of 95% empty values. The column consists of categorical data of either YES or NO. The relative difference of whether a car will be lemon was somewhat significant so I decided to keep the columns but replace the missing data with the category of MAYBE. This MAYBE category had nearly $4\times$ as many lemons as the NO/Green category.

Additional missing values were checked for in the training and testing set that were not apparent in the head or tail of the data. A couple thousand data points had some sort of data missing. Sometimes the absence of data is insightful as we previously saw, so we filled the missing values with place holders of MAYBE for categorical data and the average of the column.

Next we'll set the categorical data to a numerical value that corresponds to a respective category. First we'll set all the categorical data to upper case. It was found that one column had less categories than was picked up by producing a dictionary due to inconsistent casing. A dictionary of indices was built from the training set and applied to the testing set. If the testing set had a category that didn't exist in the training set then the new category was assgined a value of -1.

Although there are definitely some features that don't have as much relevant information, we'll leave them in to be trained by a classifier and remove the features in further optimization.

2 First Pass Model: Logistic Regression

Logistic regression is a good choice for this since it would provide a first model due to the ease of implementation and that there's a certain probability that car will be a lemon.

We do a 80/20 split of the labled data into a training and vadidation set, respectively. From this we are more easiy able to evaluate the predicting power of our model. By training the model with 80% of the data, we find that our first pass model has an accuracy of ~88% on the training and testing data. Not a bad model for a first attempt.

Upon inspecting the prediction generated by the model, the high accuracy percent is misleading. The model predicts that every car is a good car which is pointless. Further inspection of the labels of the training set show that 88% of the data are cases in which the car is good (not a lemon).

The high mismatch of positive and negative data ponits is skewing our model to only predict that every car is good. In order to counter this we will create a new balanced training data set that is composed of 50% good cars and 50% lemons. All of the 'bad' data points from the training set (as 1 for IsBadBuy) were used and an equal number of randomly selected 'good' data points were used. The new balanced training set was created and the data points were randomly shuffled so not to bias the training.

A logistic regression model was fit to the balanced training set and found to a have a balanced training, balanced testing and overall testing accuracy of 67.3%, 67.3% and 71.6%. A quick check of the number of negative predictions shows that the model is making valid predictions and not just guessing that every car is valid.

The model scored a 0.18338 on Kaggle. The high score is 0.26719.

3 Future Directions

Next steps would be to inspect the data deeper, appropriately select the most relevant features, normalize the features appropriately, reduce feature size by combining features into insightful metrics, optimizing model parameters and using other model types (SVM, DNN, etc.).