ECS171 Winter 2018

Final Project Report

Group Member: Guowei Yang 913205497, Anze Wang 912777492, Stella Xiao 912192093

The Loan Default Competition

1. Overview

We use a two-step method to predict the loss. In the first step, we predict the probability of default using classification methods. In the second step, we predict the actual loss, using regression, on the set of data points that we predicted that the data would default in the first step.

2. Cleaning the data

Before we start our training process, we clean our data following the process below:

a. Cleaning the Training Data based:

1. Delete any feature that has the same value based on the first 25,000 data points.
2. Find the median values for all features and replace 'NA' with the mean value of the feature which it belongs to.

b. Cleaning the Testing Data:

1. Delete features that we delete in the first step of cleaning the training data.
2. Find the median values for all features and replace 'NA' with the mean value of the feature which it belongs to.

3. Feature Selection:

We first find the correlation coefficients between the features and the loss and if the coefficients > 0.011, we then selected those features for later use and for the sake of clarification in this report, we name this set of features as feature1 (We are not using this name in our code).

Because the project is based on loan default and features are transaction data, we guessed that maybe some feature could be repayment and some feature could be loan. So, maybe the differences of the features may tell us something about the probability of default. We noticed that the differences of certain feature pairs can result in a high classification accuracy, e.g. f527 and f528. We soon found out that those feature pairs are high correlated. So, we searched in the training data to find all highly correlated feature pairs. During this step, we also found out that there are some repetitions in the feature, i.e. a feature is identical to another feature. So, we excluded those feature pairs who have correlation coefficient > 0.99999999999999. We selected the feature pairs with correlation coefficients > 0.999999 and keeped their differences as a set of new features. We then find the correlation coefficients between the new features and the loss and selected features with correlation coefficients > 0.02. Again, for the sake of clarification in this report, we name this set of features as feature2 (We are not using this name in our code).

Now, we combine feature1 and feature2 named as feature_class (We are not using this name in our code). Sort all the features in the feature_class by their feature importance (using feature_importances[1] in Gradient Boosting Classifier) in a descending order. Pick the top 25 features as the features we used in training our data.

4. The First Step (Classification):

A. Model(s) that we used:

We use gradient boosting tree to train the data, with parameters: n_estimators=160, max_depth=5, max_feature=sqrt(number of features).

5. The Second Step (Regression):

A. Model(s) that we used:

We used Keras sequential model to implement a simple neural network to train the data. We, at first, make baseline prediction without a hidden layer but then add two layers to the model, forming: input layer→wider layer→narrow layer →output layer.The first hidden layer has around twice as many neurons as the input layer, and the second hidden layer has around square root of number of neuron as in the first hidden layer. In the train set, for data that have positive loss, guessing all zero would give an average loss of 8. Guessing the average would give an average loss of 6, and we take this as our baseline. Our neural network would give an loss of around 4.8-4.9 in the train set when doing 5-fold cross validation. Though there's still space to improve, this is a pretty good result.

6. How to generate the result using our code, i.e. README file:

We used numpy, keras, and scikit-learn packages. So, make sure that those packages are installed. Run our code following the steps below:

1. python data_preprocess.py to get the clean data with features we selected.
2. python model.py to generate the "submission.csv".

Reference List:

1. "1.13. Feature Selection¶." 1.13. Feature Selection - Scikit-Learn 0.19.1 Documentation, scikit-learn.org/stable/modules/feature_selection.html.