STAT 440 Module 2 Report

Chris Sobczak (301335820)
Chris Liu (301299668)
Daijing Lin (301326091)
Sandy Wu (301273729)

November 20, 2020

1 Introduction

This second module for STAT 440 we were tasked to analyze simulated data with copious explanatory variables mapped to fifteen response variables. The original data set had 75 total covariates divided into 7 groups denoted by the leading letters `AXX` through `FXX`, `XX` denoting the number within the group so `A01` is the first explanatory variable in the first group and `B02` is the second variable in the second group. We had to produce 50,000 predictions on a variety of response variables.

The training set that we were given included 153,287 observations of all 79 explanatory variables and fifteen responses. The first two groups (`AXX` and `BXX`) range from negative 5 to positive 5 while the range for groups `CXX` through `F02` is negative 1 to positive 1, with some exceptions: `D12`, and `F03` through `F08` follow the first group ranging (-5,5) as well.

The test set included 50,000 observations, with explanatory variable ranges within that of the training set, to be used to predict select observations to select response variables. We used a gradient boosting model to produce our final predictions.

2 Methodology

Using the metrics root-mean square error (RMSE) and mean square error (MSE) we employed a variety of models to compare including multiple linear regression, gradient boosting, random forests, neural networks, deep learning and ensembles of these methods predominantly using the R API for h2o (`http://docs.h2o.ai/h2o/latest-stable/h2o-docs/index.html`). We implemented grid-search and other parameter seeking algorithms for each modeling technique, however the best results were consistently produced by our gradient boosted models, therefore we used a final version of it for the competition predictions.

2.1 Exploratory data analysis

Figure 2 shows the correlation matrix of all the data. One can note that there is significant multicolinearity within the `CXX` and `DXX` groups.

2.2 Data Manipulation

After carefully looked at the structures for every explanatory variables, we found most of the columns follow Uniform, Normal, or Bernoulli distribution with some human-created outliers(mostly -9 or 9). We first filled in the missing values with the median of each columns, and then we removed rows with extreme

human-created outliers(-9 or 9). Furthermore, after some data inspection on the X_test dataset, we found that the last six columns(F09 to G03) were useless since all values equal to -9. We will delete these columns from both training and testing set. By doing these three data manipulation processes, we ended up with a much nicer training set and only reduced our data from 153287 rows to 147006 rows with 53 explanatory variables.

2.3 Feature Analysis

Using multiple tools including Figure 2 and `h2o.ai`'s built in variable importance plot tools we programmatically selected variables after each training run, training separate models for each response variable `ZXX`. Most of the variable selection processes are automatically handled by `h2o.ai` and model pruning.

2.4 Model Building

Beginning work with the second module, machine learning tools that could automatically allocate jobs to different computing cores was necessary given the volume of data that we were give. Therefore, `h2o.ai` was a perfect candidate of software. We attempted each of the supervised learning methods from their documentation, so this included random forests, general linear models, gradient boosted machines (GBM), xgboosted machines, neural networks and stacked ensembles of each of these. From the starting point of trying all the models, we found GBM models and neural networks the most successful based on cross validated root-mean square errors.

2.5 Neural Network

Neural network has been applied onto the the manipulated data with a slight modification. The outliers are no longer manually removed but rather with z-score. For each column, only values with z-score of less than 4 will be kept. The replacement method of using median for NAN and "?" remained in the neural network approach. Upon analysing correlation, the variable C01 and D04 have been removed because has correlation of higher than 0.6 with C2 and D3. The training set is split into a sub-training set and a validation set.

2.5.1 Architecture

The architecture used in Module 2 is an attempt to recreate RESNET-18 in which instead of a ResNet Block, a linear block has been replaced. Inside this linear block, it contains a linear layer, RELU layer and then a dropout layer (x3). The probability of the dropout layer has been selected as 0.3 after series of testing from range from 0.3-0.8. Before it returns the result inside the block, it adds the result to a identity connection in which it's just a single layer of linear layer. This allows the model to skip the three linear layer if only performing one linear layer is better.

Linear Block
Linear Block(in_c, out_c)
ReLU()
Dropout(0.3)
Linear Block
Linear Block(out_c, out_c)
ReLU()
Dropout(0.3)
Linear Block
Linear Block(out_c, out_c)
ReLU()
Dropout(0.3)
+ Linear Block(in_c, out_c)

There are 19 layers in which 18 of them are linear blocks and last layer is a linear layer to each of the columns of y-set. The choice of number of perceptron are set to 128, then to 256 and then back to 128 following the idea of deep feed forward network. The loss function is chosen to be MSELoss, and optimizer has been set to Adam. Learning rate has been set to 0.0005, batch size has been set to 512, epoch has been set to 200 and weight decay has been set to 0.0001. See Appendix for visualization.

2.5.2 Model result

Results from neural network are not good and it has already been revised to combat overfitting. It only goes up to 1.1 on the public leader board. The validation MSE is 4.4 while training MSE has already gone down to 3.4. To combat overfitting, depth of the model increased/decreased, generation of missing value indicator, dropout layer, batch normalization, early stopping, tuning of learning rate with weight decays has all been attempted but validation MSE just remains to be around 4.4.

3 Results

Our most successful machine ended up being a GBM with explanatory variables `B13` through `F08` with 25,000 trees and a learning rate of 0.05 and specific prediction distributions for each of the response variables. For some of the specific response `ZXX` values, they are distributed as bernoilli, so identifying those response columns as such was important and allowed us to produce a final score on the competition set of 1.01474.

4 Conclusion and Future Improvements

Throughout the training process of the models we constantly struggled to run the scripts because of the copious amount of training data provided. To solve this problem when impossible to run the models on the full set, we read in the full data but then reduced the size of the sets by sampling randomly a given reduced size. With better hardware or optimized software and expense minded development we could probably improve our predictions and train the models on the full set of data.

5 Appendix

Figure 1: Neural network Architecture

```
self.fc1 = LinearBlock(n_features,128).cuda()
self.fc2 = LinearBlock(128,128).cuda()
self.fc3 = LinearBlock(128,256).cuda()
self.fc4 = LinearBlock(256,256).cuda()
self.fc5 = LinearBlock(256,256).cuda()
self.fc6 = LinearBlock(256,256).cuda()
self.fc7 = LinearBlock(256,256).cuda()
self.fc8 = LinearBlock(256,256).cuda()
self.fc9 = LinearBlock(256,256).cuda()
self.fc10 = LinearBlock(256,128).cuda()
self.fc11 = LinearBlock(128,128).cuda()
self.fc12 = LinearBlock(128,128).cuda()
self.fc13 = LinearBlock(128,128).cuda()
self.fc14 = LinearBlock(128,128).cuda()
self.fc15 = LinearBlock(128,128).cuda()
self.fc16 = LinearBlock(128,128).cuda()
self.fc17 = LinearBlock(128,128).cuda()
self.fc18 = LinearBlock(128,128).cuda()
self.fc19 = nn.Linear(128, 14)
```

Figure 2: Correlation Plot of all covariates to all responses