Statistics101C Final Report

[cyc00]

[TEAM ID:717009]

Joshua Oh(joshua.sh.oh1@gmail.com)

Instructed by: Akram Almohalwas

Spring 2017

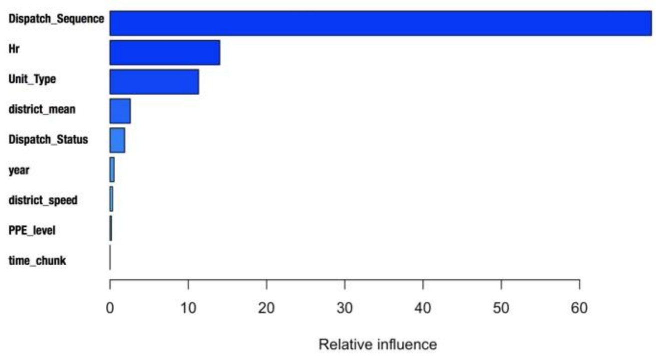
1. Data Cleaning and Transformations

- **Data Cleaning:** I deleted the variable "Emergency Dispatch Code" from the dataset, because it only had one level, Emergency. I also removed the variable "incident.ID", since it was just a useless random ID number. There were missing values in both the training dataset and the testing dataset. Alongside, since most NAs in the training dataset were contained in the response variable, I simply deleted those observations. After removing NAs in the training data, there was only around 0.1% of data containing missing values for predictors so I decided to remove those observations because of the low percentage. For the testing data, there were missing values in the predictor "Dispatch Sequence" and so I used linear regression for the training data to predict the dispatch sequence for these observations.

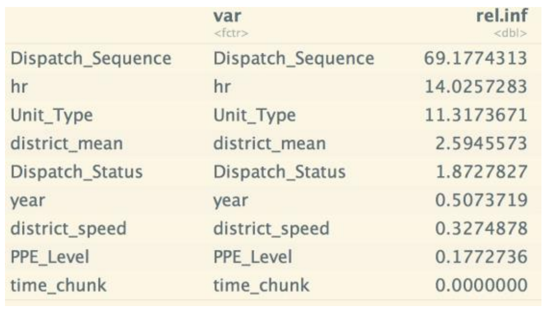
- **Data Transformation:** I created a new categorical variable called "hr" by extracting the hour of the incident from the predictor "Incident Creation Time (GMT)". After grouping the average elapsed time by hour, I created a new predictor called "time_chunk", which classifies the observations into one of the four levels: fastest, fast, slow or slowest. I also changed the predictor "year" and "First in District" into categorical variables. By grouping the median elapsed time by "First_in_District"(the fire station), I created "district_speed" which ranked each observation with a numerical score from 1 to 9, indicating the speed of that station. I took the log transformation on the predictor "Dispatch_sequence", since it represented the "waiting time" of vehicles to go to incident places and may follow an exponential distribution. Since the categorical variable "First_in_District" has more than one hundred levels, I replaced it with the mean response value to avoid overfitting. For features "Dispatch Status", "Unit Type" and "PPE Level" and "row.id", I just left them as they are.

2. The Model

```
bst3<-gbm(elapsed_time~.-row.id, interaction.depth = 3,
data=l[-idna,],distribution="gaussian",n.trees =300,verbose =F, shrinkage =0.15)
```



I used a boost model with the 9 predictors mentioned above. I tuned the parameters manually by creating a



	var	rel.inf
	<fctr>	<dbl>
Dispatch_Sequence	Dispatch_Sequence	69.1774313
hr	hr	14.0257283
Unit_Type	Unit_Type	11.3173671
district_mean	district_mean	2.5945573
Dispatch_Status	Dispatch_Status	1.8727827
year	year	0.5073719
district_speed	district_speed	0.3274878
PPE_Level	PPE_Level	0.1772736
time_chunk	time_chunk	0.0000000

testing set of the actual size of the testing data and tried it for almost 20 possible models. The model "bst3" with a shrinkage value of 0.15, 300 trees, and an interaction depth of 3. Together this usually produced the smallest MSE for the testing set I separated from the training data. After realizing bst3 was the best model, I used all of the training data to fit this model, thereby producing the final model shown above. The chart above is the relative importance of our predictors for the model.

3. Best MSE

The best score on Kaggle I inputted was 1392287.13605.

4. Strengths and Weaknesses of the Model

- **Strengths:** Compared to other models that I tried(PCA, PLS, random forest, tree, Lasso, and ridge), the gradient boosting model had the highest predictive power, since it minimized the MSE. Furthermore, it is relatively computational inexpensive in comparison to random forest, making it possible for me to use all of the training data to build and tune our model. Our boosting model also avoids overfitting, since I tuned the parameter shrinkage, creating a penalty for less useful predictors. Unlike trees, boosting models allowed me to use the categorical predictor "Unit_Type" which has around 40 levels. Therefore, I avoided losing information by reducing the number of levels in predictors (i.e. replacing levels of very low frequency with levels of high frequency).

- **Weaknesses**: I searched online for possible external data, such as the location of the fire station. But it turned out none of them were helpful in terms of decreasing MSE. If I could find any useful external information, the predictive power would have increased. Furthermore, I tuned the parameters manually by trying some possible combinations. (Because it would have taken a very long time to use package caret to find the best combination) Due to the limited number of combinations I tried, the combination I used for parameters may be a set of values that just produced a "local minimum" MSE. If I actually tuned the parameters using functions in caret, I may have gotten better results.