Los Angeles Fire Department Responding Time Model Report

by Xin Jin, Yaxi Lou, Yu Zhang

For this predictive model, we tried two different approaches, including Extreme Gradient Boosting (xgboost) and Random Forest, and we recode our datasets accordingly.

1.Code

In order to construct an efficient and precise model, we need to convert all categorical variables into numeric variables since xgboost requires a numeric matrix. We are able to find the corresponding zipcodes to the variable "First in District", which is the fire station where the incident happened nearby. This variable is important since some fire stations are located closely.

For time variable, we simply split them into hours, minutes, and seconds. We think hour is more important than minutes, so we decide to recode minutes so that if it's over 40 we add one more unit to its corresponding hour (i.e, 10:40 will be code as 11).

We add a new variable called traffic based on the variable hour and the typical rush hour, because we think a heavier traffic may influence the travel time of LAFD. Hours 7-10 and 15-9 are coded as 3, 11-14 and 20-23 are coded as 2, the rest is coded as 1.

We add one more variable which is frequency of the incident id. We make the table of frequency of incident id with corresponding median of response variable. We find out that larger the frequency larger the median of elapsed time. Thus, we decide to use frequency of incident id as our new variable.

For Unit Type, there are some differences between testing and training data. There are some types existing in training data set, and some are only in testing data set. So we recode these elements as other, to reduce the error.

Because variable Dispatch Status has many levels, we have to recode it in order to maintain less levels. We group dispatch status with its frequency, mean, median, minimum, and maximum of response variable, and make the table. We combine different Dispatch Status together with similar median elapsed time. We used the same approach to recode the unit type.

We also replace the missing value of Dispatch Sequence by 0, and delete other NAs in training dataset.

The final dataset contains 18 variables, including incident.ID, row.id, year, First in District, Emergency Dispatch Code, Dispatch Sequence, Dispatch Status, Unit Type, PPE Level, Incident Creation Time (GMT), hour, minute, timelevel, frequency, zip, traffic, speed and elapsed_time. (Testing data does not have elapsed_time.)

2. Model

a)Random Forest

Random Forests is an ensemble learning method for classification, regression, and other tasks by constructing decision trees for training and outputting the class that is the mode of classification or regression of the individual trees.

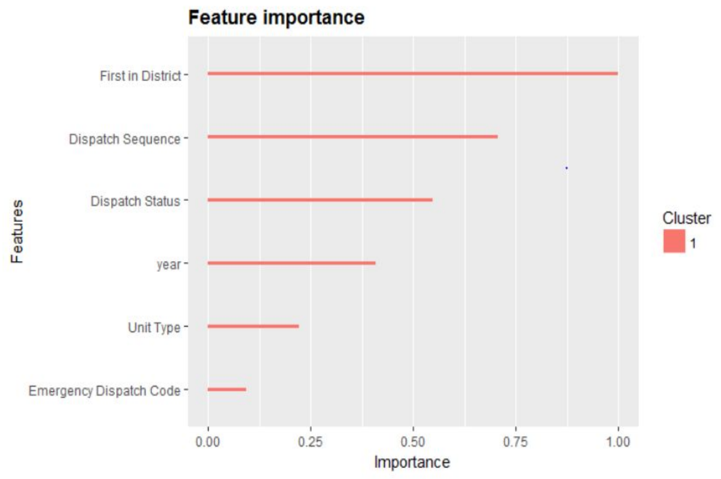
For random forest, we use Dispatch sequence, PPE level, zipcode, frequency, traffic, Dispatch Status, and Unit type as our predictors of the model.

Because of the limitation of the memory we have to use ntree = 500 with max_depth = 20 for training the model.

b) Extreme Gradient Boosting (xgboost)

XGBoost is an optimized distributed gradient boosting system designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. It is a tree-based model which is faster than random forest and precisely avoids overfitting.

For xgboost model, we use xgb.importance() to do variable selection. We obtain the plot below:



We try to use features above, but finally we decide to use predictors Dispatch Sequence, PPE Level, zipcode, frequency, and traffic because the combination of these variables provides the most accurate prediction for us. We also try other variables such as year, Dispatch Status and Unit Type, but these variables seem to be insignificant and produce larger errors.

Applying gboost model in R requires inputting proper parameters including eta, gamma, max_depth, min_child_weight and nround. After trying many combinations, we choose eta=0.7, gamma=0, max_depth=10, min_child_weight=1 and nround=50.

3. MSE

We get the minimum training error of 1037 for xgboost model. (We are able to get smaller training error, but it seems provide the overfitting issue since the prediction is not good enough.) For Random Forests, we get the training error of 1161.151. The xgboost prediction is scored

1441701.32929 on Kaggle, which is an estimator of MSE. The Random Forest prediction is scored around 1,460,000.

Since we take two different approaches for model construction, we take the mean of two predictions (with MSEs over 1,440,000 and 1,460,000), which is surprisingly good with a Kaggle score 1399072.83235 publicly. We examine the two prediction and find that they have similar patterns but one has smaller values and the other provides larger values. Taking the mean will help us to balance the error.

4. What do we think

Both random forest and xgboost are efficient and accurate for making predictions. The two models are able to produce descent approaches to the right prediction. The combination of the two model works better for us.

There are also some problems getting us into troubles. For instance, the original dataset contains many categorical variables, but we need numeric variables. There might be some errors during recoding process which caused by levels of those variables, so we are not able to get a perfect prediction by applying models repeatedly.

In addition, our personal computers have limited memory and computing speed so we cannot apply the two models to the entire dataset. We can ty to use AWS next time for a better result.