STAT 652: Predicting Flight Delays Project

Vincent Chiu

11/26/2019

Contents

Hi there My plot is called Figure 2. Figure 1. bye

1. Introduction (brief)
2. Data (brief)
3. Methods
4. Results
5. Conclusions and Discussion

1 Introduction

The goal of this project is to predict the response variable, departure delays for a particular flight given the explanatory variables.

2 Data

The dataset consists of information about all the flights leaving from New York City in 2013. The dataset contains 43 variables in total. The dataset is an algamation of several datasets including datasets containing information on weather, the airports, the flights, and the models of airplanes. The training dataset provided to us contains 200,000 observations.

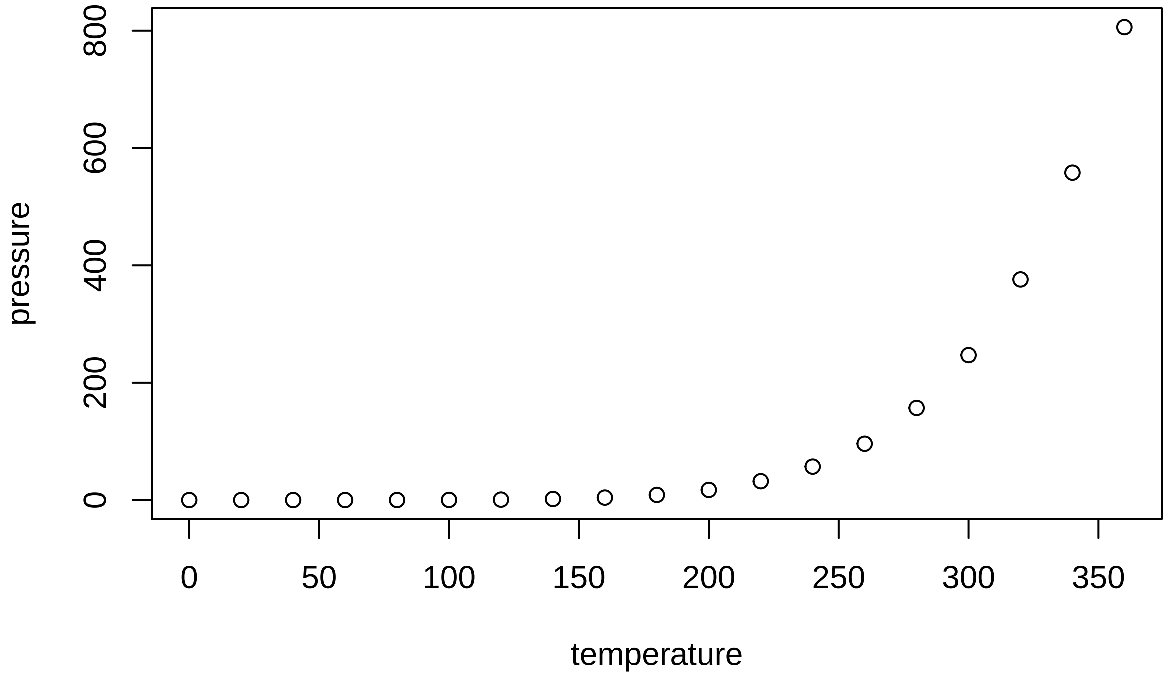

Figure 1: My caption

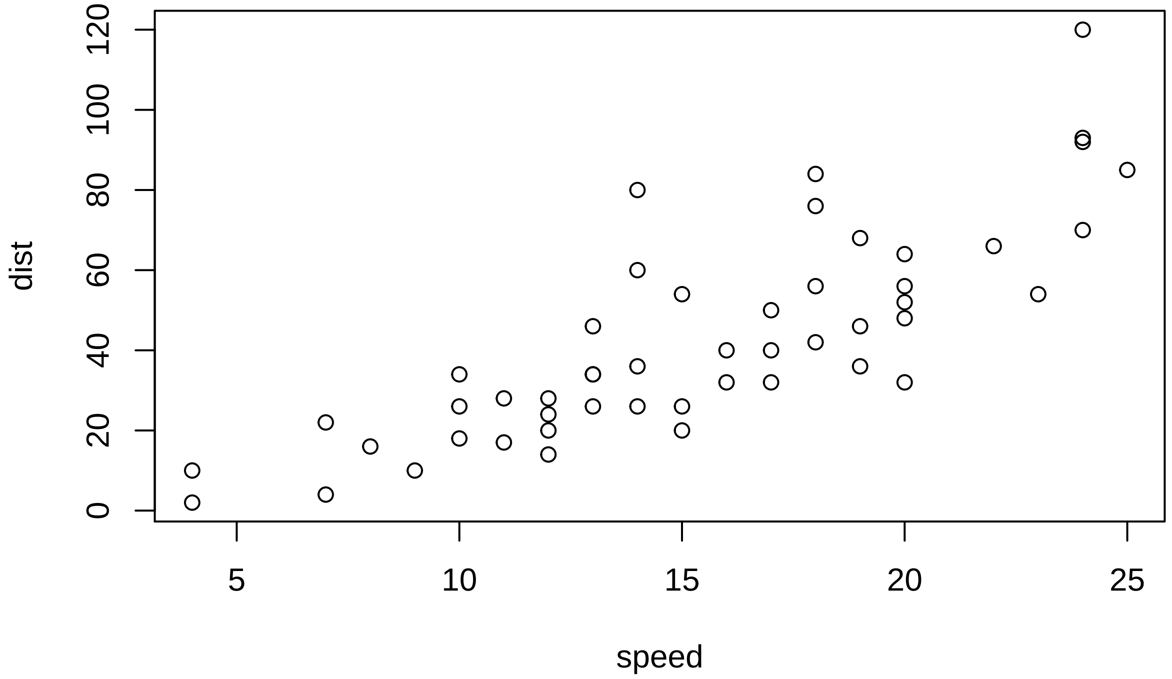

Figure 2: My caption

3 Methods:

We will now outline the various methods used to clean and perform prediction on the data. We will discuss our techniques for data preprocessing, cross validation and the different models that we tried.

3.1 Data Preprocessing

I performed data preprocessing. My data preprocessing steps included the following:

- loading the data from a csv
- setting the random seed for reproducibility of results
- casting all the columns with the character data type into the factor data type
- converting the shed_arr_time and sched_dep_time columns into the POSIX time format so that I could accurately take the difference of them.
- Dropping columns that contain data from after the planes' departure which may leak information about the response variable dep_delay. We also drop columns that contain data from after the planes' departure which may leak information about the response variable dep_delay. We drop the columns "dep_time", "arr_time", "air_time", "arr_delay", because that leaks the response variable. We drop column "year.x" because all the values are 2013 We also drop tailnum because it produces too many dummy variable columns for one hot encoding.
- Dropping columns which consists of over 50% NAs which include the speed column
- Impute NAs for the remaining columns using the imputeMissings library, adding a boolean flag which indicates 1 if the associated value was 1 and 0 otherwise. For example the model_flag for a given row is 1 if the model value is NA for that given row.
- Scaling the data to work well with methods like lasso regression.
- Only kept data which had a departure delay of less than 30 minutes late, which reduced the dataset from 200,000 rows to approximately 170,000. This is because I assume that extreme delays of of over 30 minutes late are freak accidents which cannot be accurately predicted by the available explanatory variables.

3.2 Cross Validation

Initially, I used the most basic cross validation technique where I have a training dataset and a cross validation dataset. I split the original data into a ratio of 2/3 train and 1/3 of the data for cross validation. There is a additional data which would be provided by the professor at a later date which we will use as the holdout test set. I believe that 2/3 of the data gives enough data for the models to train on while 1/3 is enough data for us to get an accurate assessment of the error. k-folds cross validation was not initially used in order to save on compute time as we were intially only exploring the models. k-folds cross validation would increase training time for the models by a factor of k.

3.3 Models

We first explored some basic models to establish a baseline performance and compared it to our most sophisticated model, the Generalized Boosted Regression Model (GBM).

3.3.1 Basic Models

dep_delay is the number of minutes that the plane either departs early or late. Negative numbers are for early departures and positive numbers are for the number of minutes the plane is late. First, I used a basic model of simply predicting the dep_delay to always be 0. This was done to establish baseline performance. This model had an root mean squared error (RMSE) of 8.30571. TODO The model in which I predicted the mean for all the predictions had an RMSE of TODO.

3.3.2 Linear Regression

Then I tried linear regression with dep_delay as the response variables and all the other remaining variables as the explanatory variables. This model was better than predicting the mean with an RMSE of TODO. This suggests that there is some relationship between the dep_delay and the explanatory variables.

3.3.3 Generalized Boosted Regression Model (GBM)

Afterwards, we tried a Generalized Boosted Regression Model (GBM). This model had the lowest RMSE on the test dataset after it was tuned to have a shrinkage of 0.01 and around 16,000 trees. Shrinkage is proportional to the learning rate. 16,000 trees is the number of trees used in the model. Each iteration uses 1 tree, so 16,000 trees also refers to the number of iterations. According to the vignette, the RMSE can always be improved by decreasing shrinkage, but this provides diminishing returns. A good strategy would be to pick a small shrinkage that balances performance and compute time. Then with this fixed shrinkage value, increase the number of trees until you get diminishing returns. We decided to follow the aforementioned strategy.

4 Results

In regression and gbm, I found different features to be important. For the best gbm model, dest which refers to which airport a given plane is going to was the most important feature. However, the one hot encoding versions of carrier were the most important features for regression. On the other hand, dest does appear as an important feature in linear regression as well but it is not the most important feature. I surmise that if we can somehow sum up all the contributions from each of the one hot encoded variables derived from dest then, it might appear as the most important feature for linear regression as well. We can try Anova in order to measure the statistical significance of dest. Performing anova on comparing linear regression model with and without dest, it was determined that due to the low p-value of 0.0001863 associated with having dest that keeping at least one of the one hot categorical variables derived from dest is beneficial for the linear regression model.

TODO: try interaction terms , try anova.

5 Conclusion and Discussion

5.1 Discussion

We considered removing outliers in train but not in test, then use k-folds cross validation on test to determine how many outliers we should remove to boost performance on the cross validation set. We considered removing points that are outliers aka points with high cook's distance in order to train a better model. Howevever, this was infeasible as we did not have enough computational resources available and it took too long.

5.2 Conclusion

In conclusion, out of the methods that we covered in class, I found gradient boosted models to provide the best performance based on having the lowest root mean squared error on the hold out test set.

Based on the relative influence scores provided by the gbm, some of the most important feature variables include dest, model, and sched_dep_time_num_minute.

The dest column contains the airport code for where a given flight is flying to. Based on my run of gbm with a shrinkage of 0.01 and 16834 trees, dest was the most important feature with 49.56 relative influence.("Gradient Boosting Machines · UC Business Analytics R Programming Guide" 2019).

5.3 Future Work

TODO: remove points that are outliers ie dep_delay > 200 or 300 etc. or remove less than x number of points. then use k-folds cross validation on cross validation set where no points were removed. can repeat k-folds for different seeds. can just try this on my quickest model, ie linear regression. should be bowl shape vs rmse vs. number of points removed. theoretically

I also considered removing based on cook's distance but this took too long to compute.

5 folds with 10 different random seeds

have train, CV and test setå 1/3 train, 1/3 CV, 1/3 test 2/3% train, 1/3%CV, wait for prof test set

try lasso regression

6 Code

In separate file.

References

"Gradient Boosting Machines · UC Business Analytics R Programming Guide." 2019. http://uc-r.github.io/gbm_regression#h2o.