Ryan Blosser
Greg DeVore
Deriving Knowledge from Data at Scale, Fall 2017

Data Science Capstone Project: Predicting NYC Taxi Trip Times

Problem Definition

The primary data source for this project was a set of approximately 1.7 million taxi rides from NYC, circa 2013. We were tasked with proposing a business question that could be answered using this data, and we decided to focus on a subject that would have a significant business impact: Can we accurately predict the length of a taxi ride given data that would only be known at the beginning of the trip? Doing this could have a profound impact on driver scheduling, as companies could ensure they had the necessary amount of drivers to handle the expected volume of rides at any given time. Companies could use this information to predict the fare of a ride, so passengers would know what to expect up front.

Both regression and classification models will be explored. The regression model (predict trip time in seconds) gives an average prediction error of 4 minutes. The classification model (predict trips as short, medium, or long) gives an overall prediction accuracy of 80%. Given these results, and the ability to model traffic (which has the largest impact on trip time), the classification approach is recommended.

Data Description

After arriving at a problem, we first had to understand our dataset. The taxi data features are:

- Medallion/Hack_license - These are transferrable and non-transferable license IDs, respectively. The first gives someone the ability to drive a cab, the second is specific to an individual. We only used these to join the two datasets together.
- Vendor_ID - A code for the payment company of the particular cab. We did not use this.
- Rate_Code - The rate category charged by the taxi. We interpreted this in-line with the notion within the industry: “special trips, such as to a specific airport or end location have a different rate”. We did not use this.
- Pickup_Datetime - The date and time the passengers were picked up. We used this to estimate traffic (number of cabs in the area within the pickup hour), add weather data, and segment into hours/days/months for date/time feature engineering,
- Dropoff_Datetime - The date and time the passengers were dropped off. As the cab driver would not know this ahead of time (this is essentially what we were predicting), we did not use this in our analysis other than to validate our model.
- Passenger_Count - The number of passengers picked up in a given trip. We did not use this.
- Trip_time_in_secs - The length of time a trip took. This is simply the difference in the two datetimes above, and is the response variable in our analysis (what we are trying to predict).
- Trip_Distance - The number of miles for a particular trip.
- Pickup_Latitude/Longitude and Dropoff_Latitude/Longitude - The latitude and longitude for the pickup and dropoff location for a trip. We used these coordinates to:
    - Cluster the locations so as to give this variable fewer values
    - Get ZIP Codes for the locations
    - Derive a cardinal direction (from pickup to dropoff)
- Payment_Type - The method of payment for a particular trip. We did not use this.
- Fare_Amount - The base cost for the trip. We did not use this.
- Surcharge - Monetary value of certain extra for a trip. We used this as a binary feature.

- MTA_Tax - City-mandated tax for the trip. We did not use this.
- Tolls_Amount - The fee for tolls across the city. We used this as a binary feature.
- Total_Amount - The total amount paid for the trip, including the tip. We did not use this.
- Tip_Amount - The amount of the tip paid for the trip. We did not use this.

In addition to the taxi data, we used zip code latitude and longitude from https://gist.github.com/erichurst/7882666 to convert the pickup and dropoff locations to the nearest zip code. We also used hourly weather data from https://www.ncdc.noaa.gov/cdo-web/ to add weather related features to each trip (primarily, whether or not it was raining).

Data Cleansing

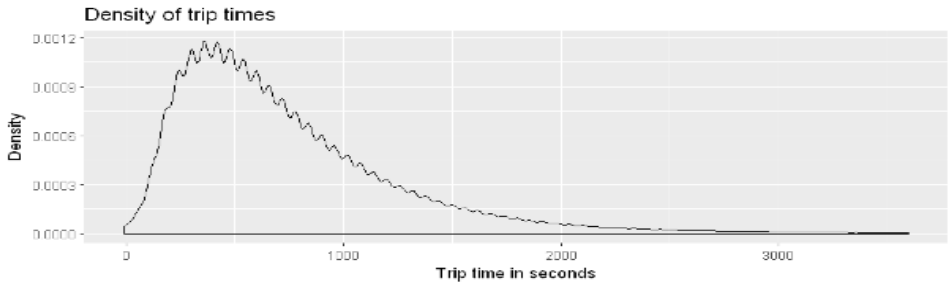
Our dataset was mostly clean to begin with, but we noticed a few things we had to take care of:

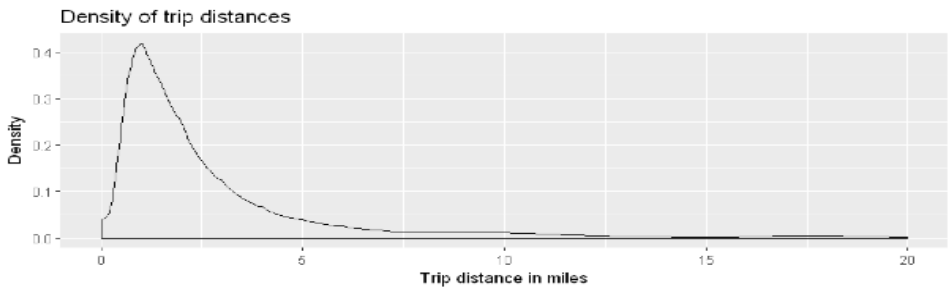
- Duplicate rows - There were a few rows (0.01%) with more than one Medallion + Hack_License + Pickup_datetime. These were removed so as not to double count a trip.
- Bad latitude and longitude values - In some cases, the latitude and longitude were simply reversed (and fixable). In a couple other cases, they took on impossible values and had to be removed.
- Zero-length, negative, or extremely long trip times and distances - These extremes were removed.
- Very low or high MPH (trip_distance over time) - Trips averaging less than 1 MPH or over 60 MPH were removed.

There were other rows with problematic money values, but we did not use these features in our analysis so these rows were not removed if everything else appeared valid.

Data Visualization

The distribution of trip time (the response variable) and distance (the most correlated predictor) are:

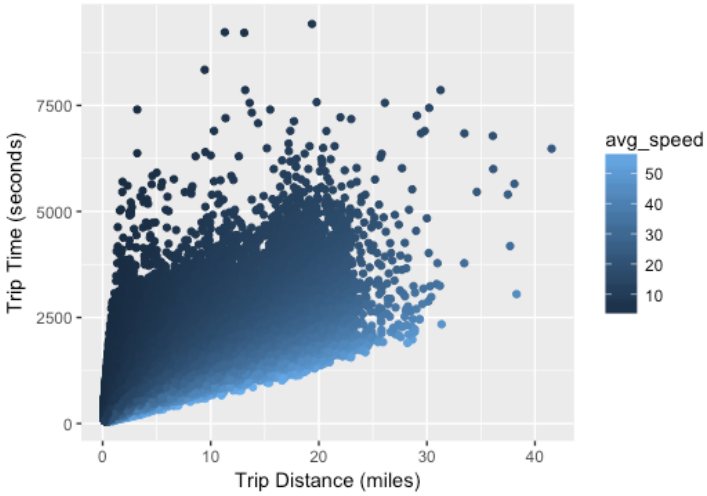




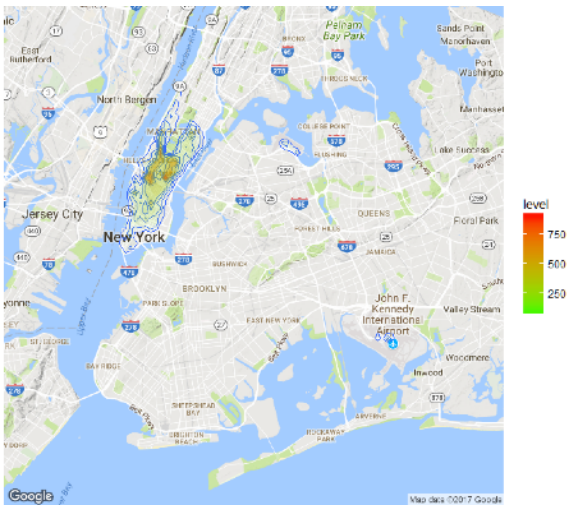
A few things to note:

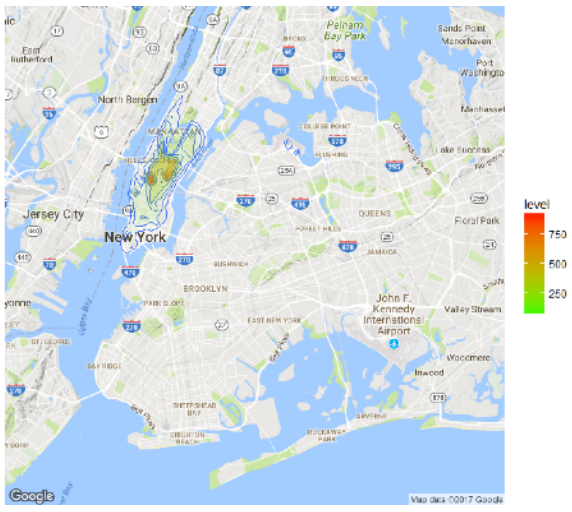
- There are zero time and distance values here, which were removed for our analysis.
- The jagged appearance of the trip time is due to a large number of the trips having times rounded to the minute (60 seconds, 120 seconds, and so on).
- The distributions have a long right tail and could have a more normal distribution with a logarithmic transformation, which was not done here.
- The median trip time is 600 seconds (10 minutes), and the median distance is just under 2 miles.

As expected, there is a strong correlation between trip distance and trip time (the correlation coefficient is approximately 0.77). The plot below shows trip time versus distance for a sample of the data set, with points colored by the average trip speed (in MPH). Note that although average speed generally increases with trip distance, most trips are on the slower end. Some of the variance in trip time is due to local speed limits, but most is due to traffic. For a fixed trip distance of five miles, the trip time can vary from 400 seconds to over 5000 seconds. The impact of this will be discussed in the Model Performance section.



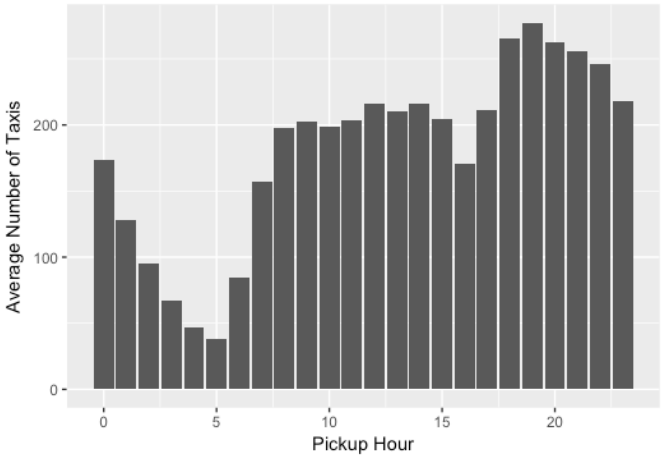
We also wanted to look at the distributions of the taxi pickup and drop offs, which are shown below (pickups left, dropoffs right). The data was clustered using K-means. Virtually all of the trips are in Manhattan, with a couple small pockets at the regions two major airports (LaGuardia and JFK). There are two incredibly heavy areas in Manhattan, with a few smaller clusters to the north and south.





Feature Engineering

In addition to adding zip code and weather data, several additional features were created from the taxi data itself. Most of these had to do with modeling traffic, which is the greatest predictor of how long a given trip distance will take. Consider the plot of the average number of taxis over each hour of the day:



The average number of taxis varies significantly throughout the day, and no features in the baseline data set explicitly capture this. To remedy this deficiency, the following features were created:

- Pickup and Dropoff Cluster - The K-means algorithm was used to the cluster pickup and dropoff locations into 14 clusters.
- Pickup and Dropoff Cluster Traffic - The total number of taxi rides in the pickup and dropoff clusters at the pickup hour. This helps to model the traffic in the local area.
- Trip Direction - One of 16 cardinal directions (N, NNE, NE, ENE, etc.) that points from the pickup to dropoff location.
- Rush Hour - A binary feature indicating whether the trip occurred during rush hour (8 am - 3pm, or 6pm - 10pm). Rush hour was determined from the bar plot above.
- Surcharge and Tolls - Binary features indicating whether a toll or surcharge was applied.

Feature Selection

Feature selection is important when building a machine learning model: It helps control overfitting by reducing the variance of the model, and it leads to a simpler model, which are faster to train and easier to explain. Because tree based models were ultimately used (see the next section), the majority of the feature selection process was done from a business standpoint. As discussed, we are only using information that would be known to a driver at the beginning of a trip. Other than tolls and surcharge data, no fare information can be used. The cost of the ride implies knowledge of the trip time, which would be target leakage. With that in mind, the following features were used:

- Pickup hour, day of the week, month
- Pickup time of day (early morning, afternoon, evening, etc.)
- Whether it is a weekend or a holiday (binary features)
- Pickup and dropoff location (ZIP codes)
- Traffic (at pickup time) within the pickup and dropoff clusters
- Whether it is rush hour
- The trip direction
- Current weather (is it raining?)
- Whether a toll or surcharge was paid (binary features)

Model Selection

Because of the nonlinear relationship between the various features and the trip time, it was decided that a tree based method would be the most appropriate. Linear regression models would not be able to capture the complex interactions between traffic, pickup location, time of day, weather, and the overall trip time. On the other hand, tree based models are very adept at modeling nonlinear behavior. The following algorithms were considered:

- Simple regression tree (from 'rpart' package)
- Random forest (from 'ranger' package)
- Boosted regression tree (from 'gbm' package)
- Extreme gradient boosted tree (from 'xgboost' package)

To perform model selection, 10% of the data was randomly sampled, and further split into a training set and a validation set (using a 70/30 split). The models were built on the training set and evaluated on the validation set. The root mean squared error (RMSE) on the validation set for each model is:

Model	Validation RMSE (seconds)
Regression Tree	336
Random Forest	260
Boosted Regression Tree	468
XGBoost Regression Tree	256

The RMSE is the standard deviation of the residuals, and is in the same units as the response variable (seconds, in this case). A lower RMSE means lower prediction error, and the random forest and xgboost regression trees performed best in this case. They will be considered for final model selection, and hyperparameter tuning will be used to see if performance can be improved.

Hyperparameter Tuning

For each combination of parameters, 10 fold cross validation was used to evaluate the performance of the model, with the minimum RMSE taken over the 10 folds. To increase the tuning speed, 1% of the rides were chosen at random for the tuning models (approximately 17,000 rides).

For the random forest model, the parameters were the number of trees, the number of features to use for each tree, and the minimum number of observations allowed in a node. The lowest RMSE was obtained with 200 trees, 5 features for each tree, and a minimum of 6 observations in a node. The default RMSE on the full validation set was 248 seconds, and after tuning, the RMSE dropped to 247 seconds.

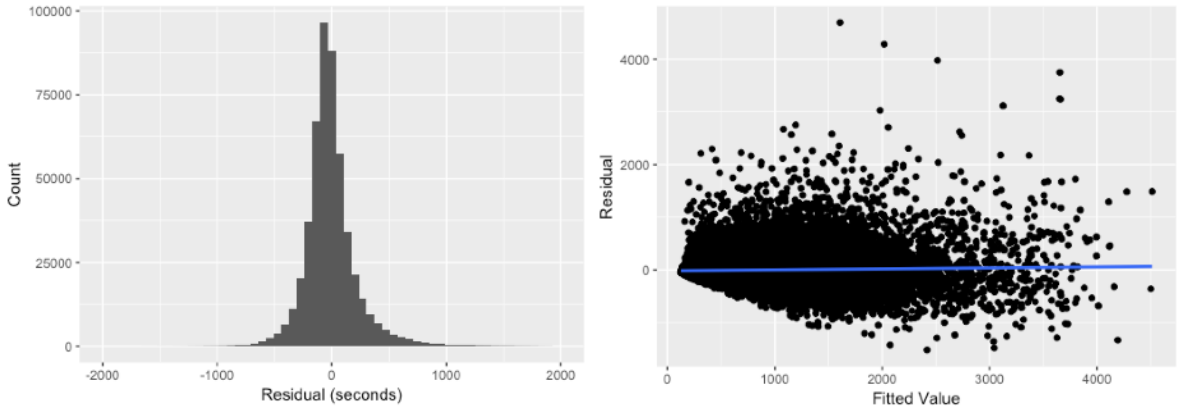
For the xgboost model, the parameters were the learning rate and the maximum tree depth. The lowest RMSE was obtained with a learning rate of 0.1, and a max tree depth of 8 layers. The default RMSE on the full validation set was 245 seconds, and after tuning, the RMSE dropped to 244 seconds.

Given that the results of the parameter tuning didn't improve either model by a significant amount, the random forest model will be chosen for several reasons: they are easier to interpret, they grow many bagged trees, which reduces the variance of the model, and they can be highly parallelized, since each tree is grown independently from every other tree.

Model Performance

The final random forest model used 200 trees, 16 features, 5 random features for each tree, and a minimum of 6 rides in a node. The validation RMSE was just over 4 minutes, with an R^2 value of 0.80.

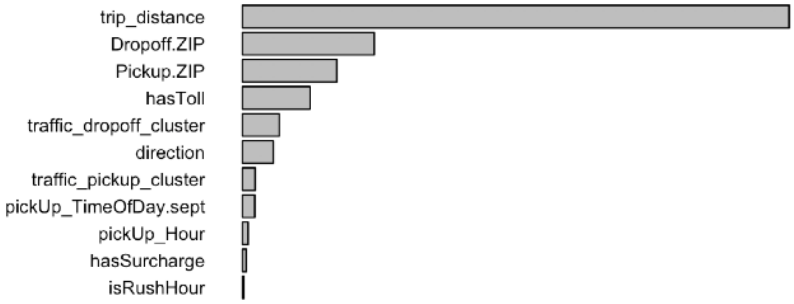
The residuals of the predicted trip times and residuals versus fitted values are shown below. The residuals are normally distributed and centered about zero, and the residuals versus fitted plot shows slight heteroscedasticity. Given that the median trip time is 10 minutes, and that 25% of the trips are longer than 16 minutes, a RMSE of ~4 minutes is encouraging, but still a somewhat large error. The model is able to capture 80% of the variance in the trip time, which is significant considering the dependence of trip time on so many features.



To put the RMSE value in context, consider a 'crude' model that predicts trip times with the following assumptions: short trips (less than 1.5 miles) occur in the city with an average speed of 10 mph, medium trips (over 1.5 miles but less than 3 miles) are mostly in the city with an average speed of 15 mph, and long trips (over 3 miles) occur on the highway with an average speed of 45 mph.The RMSE of this 'crude' model is just under 550 seconds, and serves as a sanity check that the random forest model is an improvement over a system that doesn't require a model at all.

Regression vs Classification

As noted above, predicting the trip time in seconds is a difficult task. The most important features (obtained from the simple decision tree) are:



As expected, the trip distance is by far the strongest predictor of the trip time, and there is still a great deal of noise (see the plot of trip time versus distance in the Data Visualization section). In addition to the inherent difficulty presented by the large variance in trip times, predicting the trip time in seconds may not be that useful from a business standpoint. Rather, it may be more useful to be able to predict the length of

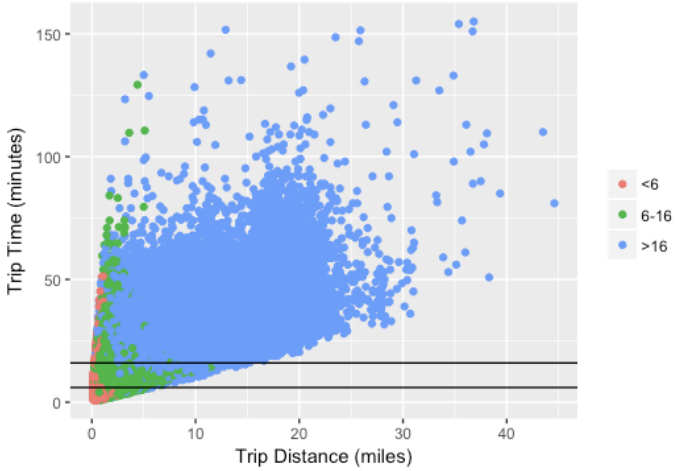
a trip in a categorical sense. Without more accurate traffic based features to help explain more of the variance in the trip time and obtain better predictions, the idea of converting the problem from a regression task to a simpler classification task was explored.

Trip times were binned using the 4-quantiles: Under 6 minutes (25% of trips), 6-16 minutes (50% of trips), over 16 minutes (the remaining 25%). A random forest classification model was built with the same inputs as the regression model. The confusion matrix (left) and validation set accuracy (right) are:

Predicted \ Actual	<6	6-16	>16
<6	93215	23558	621
6-16	28147	213281	30126
>16	98	18552	92800

<6	6-16	>16
76.75	83.51	75.11

The accuracy is the number of correct predictions in each category divided by the actual number of trips in that category The weighted prediction accuracy across all categories was just under 80%, which is very encouraging given the prediction errors encountered in the regression model. The plot below shows trip time versus distance, colored by the predicted category, with lines drawn at 6 and 16 minutes.



Most misclassifications occur for short trips that are long in duration (i.e., during heavy traffic). Once again, this is most likely due to not having accurate enough measures of the traffic along a given route. Overall, the classification results are encouraging, and it is recommended this approach be used over a regression model, given the data available and the current ability to model traffic.

Summary and Conclusion

The goal of this project was to determine if the length of a taxi trip could be accurately predicted. In addition to the baseline data set, weather and locations features were added, and additional features were engineered to try and account for the traffic in the region. A random forest regression model was chosen to predict the trip times, and an average accuracy of ~4 minutes was achieved. Given the difficulty in accurately modeling traffic, a simpler classification problem was explored. The results were encouraging, with an average prediction accuracy of nearly 80%. A classification based approach is recommended given the data available and the current ability to model traffic. As a future exercise more accurate measures of traffic should be explored to increase the accuracy of the model.

Member Contributions

Ryan Blosser primarily worked on data description, data cleansing, and feature engineering.
Greg DeVore primarily worked on feature selection, model selection, and model performance.