### Data Description

After arriving at a problem, we had to understand our dataset. The schema given to us was as follows:

* Medallion - This is a transferrable license (from one person to the other) giving someone the ability to drive a cab. We only used this to join the two datasets together.
* Hack\_license - This is another license, but it is specific to an individual (non-transferrable). We only used this to join the datasets together.
* Vendor\_ID - A code for the payment company of the particular cab. We did not use this in our analysis.
* 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 in our analysis.
* Pickup\_Datetime – The date and time the passengers were picked up. We used this to:
  + Estimate traffic (number of cabs in area recently)
  + Add weather data
  + Segment into hours/days/months for datetime 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 felt this was immaterial to our analysis.
* Trip\_time\_in\_secs- The length of time a trip took. This is simply the difference in the two datetimes above.
* Trip\_Distance- the number of miles for a particular trip.
* Pickup\_Latitude and Pickup\_Longitude – The latitude and longitude for the pickup location for a trip.
* Dropoff\_Latitude and Dropoff\_Longitude – The latitude and longitude for the dropoff location for a trip. We used all of 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 notuse this in our analysis.
* Fare\_Amount – The base value for a trip (minus tolls and other extras). We did not use any money figures for our analysis.
* Surcharge – Monetary value of certain extra for a trip.
* MTA\_Tax – City-mandated tax for the trip.
* Tolls\_Amount – The fee for tolls across the city.
* Total\_Amount – The total amount paid for the trip, including the tip.
* Tip\_Amount – The amount of the tip paid for the trip.

### Data Cleansing

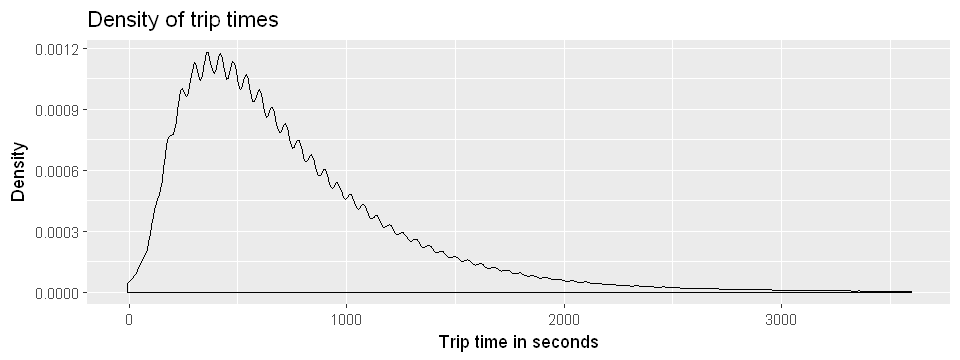
Our dataset was mostly clean to begin with, but we noticed a few things we had to take care of. There were:

* 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 or extremely long trip times and distances – These extremes were removed.
* Very high MPH (trip\_distance over time) – Trips averaging 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

After augmenting our dataset with other features (weather, time bins), we wanted to take a look at a few things. One was to observe the distribution of trip time and trip distance, and these are shown below:



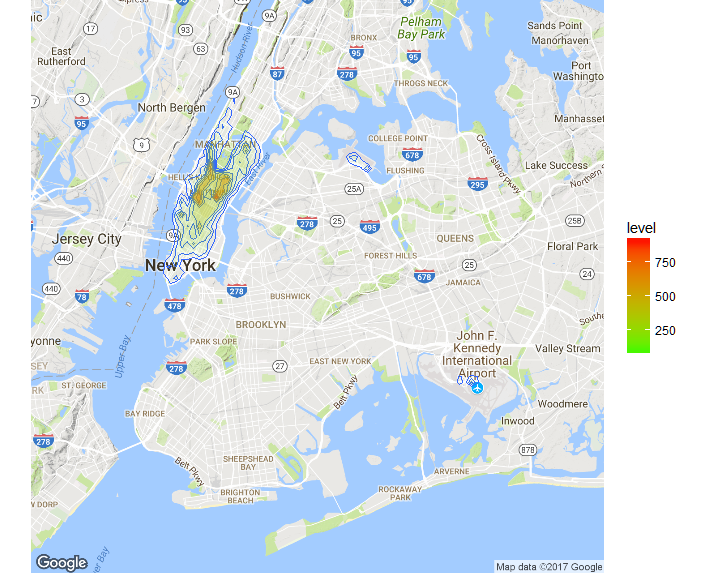
### 

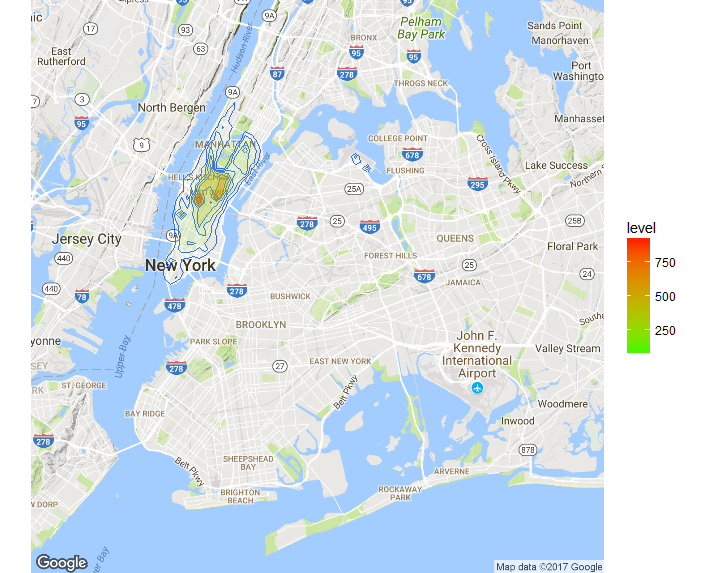
### A few things to note here:

* There are zero Time and Distance values here, these 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

We also wanted to look at the distributions of the taxi pickup and drop offs.

Pickups:



Drop offs:

Virtually all of the trips are in Manhattan, with a couple small pockets due east of there. There are two incredibly heavy areas in Manhattan, with a few smaller clusters to the north and south. We clustered the data using K-means.

Admittedly, there was not much that jumped out at us aside from a strong correlation between trip distance and trip time (more on that later).

### Feature Selection and Engineering

Feature selection is important when building a machine learning model for several reasons:

* It helps control overfitting by reducing the variance of the model
* Fewer features lead to a simpler model
* Simpler models are faster to train and easier to explain

For this problem, because of the choice to use tree based models (which inherently perform feature selection), the majority of the feature selection process was done from a logistical and business standpoint. Since we are trying to predict the overall trip time, we should only be using information that would be known to a taxi driver at the beginning of a trip. This means using no features associated with the dropoff portion of the trip, except for the dropoff location itself. In addition, no fare information can be used because knowing the cost of the ride requires knowledge of the overall trip time. This is what we are trying to predict, so using cost as a feature would be a form of 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.)
  + This was divided into seven categories, so as to put them into larger bins (fewer categories) than the pickup hour
* Whether it is rush hour (binary feature)
  + Rush Hour was derived from the New York section of the article <https://en.wikipedia.org/wiki/Rush_hour>
* Whether it is a weekend or a holiday (binary features)
  + We used the federal public holidays in the United States
* Pickup and dropoff location (ZIP codes and location clusters from kmeans algorithm)
* Traffic within the pickup and dropoff cluster (at pickup time)
  + This was defined as the number of cabs that had a pickup or drop-off in the last hour
* Trip direction (N, NNE, NE, ENE, etc.)
  + This was simply the angle made from the x-axis and the vector from pickup to drop-off, rolled up into 16 categories
* Current weather (is it raining?)
  + Weather was sourced from hourly climatological data for the JFK airport <https://www.ncdc.noaa.gov/cdo-web/>
  + Hours that had no rain data were extrapolated by using the nearest populated value (example: if 8 AM had no value, but 7 AM and 9 AM showed rain, then 8 AM also had rain)

We would have liked to use data for “big events” but we thought that a) we would discriminate against certain events since we did not know of a good, all-encompassing dataset and b) we thought the “traffic” feature would be a good analog (more cabs for bigger events).

### 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. Tree based models, on the other hand, are extremely adept at modeling nonlinear behavior. The following tree based 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)

For all models, the trip data was randomly 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 summarized in the table on the following page. 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 the final model selection, and the next step is to perform hyperparameter tuning to see if the performance can be improved.

|  |  |
| --- | --- |
| **Model** | **Validation RMSE (seconds)** |
| Regression Tree | 281 |
| Random Forest | 208 |
| Boosted Regression Tree | 364 |
| XGBoost Regression Tree | 209 |

### 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, 10% of the rides were chosen at random for the tuning models (approximately 150,000 rides).

For the random forest model, the parameters were the number of features to use for each tree, and the minimum number of observations allowed in a node. The combination that produced the lowest RMSE was: 3 features for each tree, and a minimum of 6 observations in a node. The model was relatively insensitive to the value of the parameters, perhaps because so many trees were grown (500). After tuning, the validation RMSE of the random forest model was still 208 seconds.

For the xgboost model, the parameters were the learning rate (the amount that the results of each tree are scaled by before being added to the current approximation) and the maximum tree depth. The combination that produced the lowest RMSE was: a learning rate of 0.2, and a max tree depth of 7 layers. The model was more slightly more sensitive to the value of the parameters compared to the random forest, and after tuning, the validation RMSE of the xgboost model dropped to 207 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:

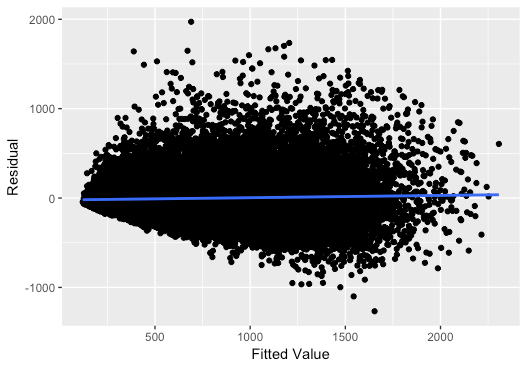
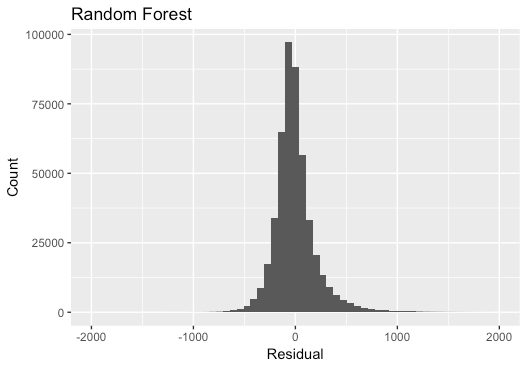
* Random forest models are easier to interpret.
* Random forest models grow many bagged trees, which reduces the variance of the model.
* Random forest models can be highly parallelized, since each tree is grown independently from every other tree. Boosted trees, on the other hand, are usually grown in series, since the error from the previous model is part of the input for the next model.

## Model Performance

The overall performance of the random forest model is summarized below:

* Number of trees: 500
* Number of features used: 16
* Validation RMSE: 208 seconds (~3.5 minutes)
* R2 value: 0.76

The residuals of the predicted trip times are shown below, along with a plot of the residual versus fitted values. Note that the residuals are normally distributed and centered about zero, and the residuals versus fitted plot shows some heteroscedasticity.



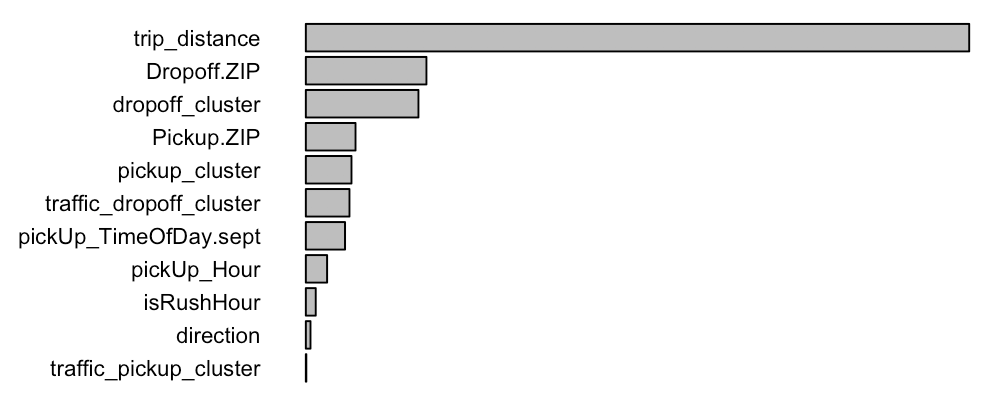
Given that the median trip time is 10 minutes, a RMSE of ~3.5 minutes is encouraging, but still a somewhat large error. The model is able to capture 76% of the variance in the trip time, which is significant considering the dependence of trip time on so many features. To put the RMSE value in context, 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
* Long trips (over 3 miles) occur on the highway with an average speed of 45 mph

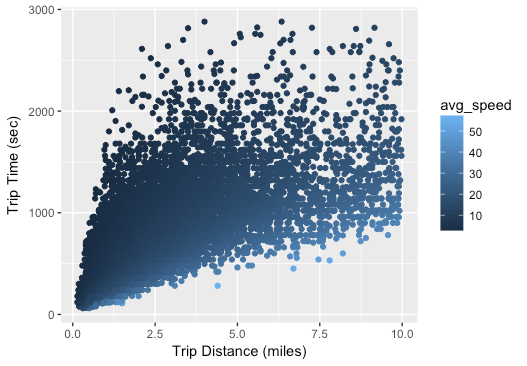
The RMSE of this ‘crude’ model is just under 480 seconds, and serves as a sanity check that the random forest model is an improvement over a system that doesn’t require a model at all. However, as stated, the error is still somewhat large.

### Regression vs Classification

Upon evaluating the regression model, the results were encouraging, but predicting the trip time down to the second is a difficult task. Consider the most important features (obtained from the simple decision tree) that are listed below.

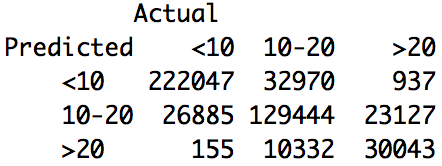


As expected, the trip distance is by far the strongest predictor of the trip time. With that in mind, the plot below shows trip time versus trip distance, colored by average speed. Note that although average speed generally increases with trip distance, most trips are on the slower end.

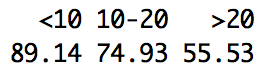


There is a moderately strong positive correlation between trip distance and trip time (the correlation coefficient is approximately 0.7). Some of the variance is due to local speed limits, but most of it is due to traffic. Keep in mind that this feature is (by far!) the strongest predictor of trip time, and there is still a great deal of noise. For a fixed trip distance, of say five miles, the trip time can vary from 500 seconds to nearly 3000 seconds!

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 more accurate predictions, the idea of converting the problem from a regression task to a simpler classification task was explored. Trips were binned into the following categories: Less than 10 minutes (52% of trips), 10-20 minutes (37% of trips), and over 20 minutes (11% of trips). A random forest classification model was built with the same features and parameters as the regression model. The validation set performance is summarized below:



The overall accuracy for each predicted category is the number of correct predictions divided by the actual number of trips in that category:



Given that just over 50% percent of trips are under 10 minutes, a prediction accuracy of nearly 90% is very encouraging. The next category covers 37% of trips, and the prediction accuracy was just under 75%. The prediction accuracy slipped a bit on the longest trip category, at just over 55%, but keep in mind only 11% of trips fit into this category. The model tends to under-predict longer trips, which once again is most likely due to not having accurate enough measures of the traffic along a given trip. Overall, these results are very encouraging, and suggest that a classification based approach be used over a regression model given the data available.