Modeling San Francisco COVID Traffic Post-Lockdown

Ethan Ikegami
Calvin Wong
Aryan Gupta

November 2021

Problem

The problem Uber faced was that the company's pre-existing traffic estimates are wildly incorrect given the COVID-19 lockdown.

This is interesting because solving the problem will save the company millions of dollars and allow us to reduce traffic congestion, which will help both the environment, and save people time - who likes being stuck in traffic?

Hypothesis

We have two hypotheses:

1. We can predict average travel time with good (90%) accuracy using part or all of the following features:
    - Location coordinates (latitude and longitude)
    - time (day)
    - size of geometry objects (area of multipolygon objects)
    - average speed
    - street speed limit
    - schoolzone proximity
    - distance between origin and destination (calculated using coordinates)
    - elevation (terrain)
2. Given spatial structuring using census tracts, linear relationships between features, and limited data, linear regression models will be better at predicting average speeds and travel times than classification models (Random Forest in particular).

Predicting travel times will address Uber's need for a new model, and if we prove that linear regression models are better, then Uber can rely on a more successful model to estimate travel times given limited data and features.

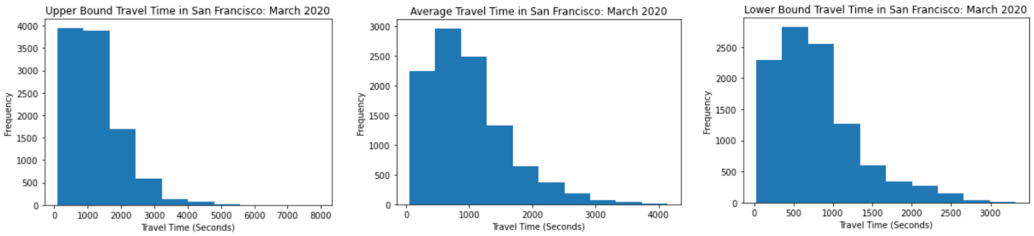
As for testing our hypothesis, our main metrics for measuring our hypothesis's validity would be RMSE (Root Mean Squared Error) and R^2 score for our linear models, and simple accuracy for classification models. If our R^2 score is much lower than the accuracy, then our hypothesis will be rejected, and we will either need to change our feature set or accept the higher error. Otherwise, we will fail to reject the hypothesis.

Results

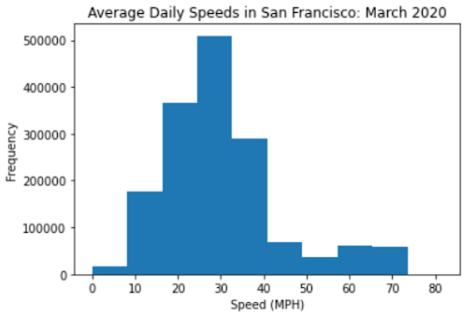
We confirmed both of our hypotheses: our second model (Linear Regression) had an improvement achieving 99.13% R^2 score and RMSE of 2.11 seconds while the Random Forest Classifier did much worse (50% accuracy) in comparison.

External Datasets

1. Daily travel times from Uber Movement data in March 2020 from San Francisco, by census tract:
    - This gives us the travel times (mean, upper bound, lower bound), days, and start and end locations. Here are its distributions:



    - This dataset was useful for our hypothesis because we saw that the day feature had a negative relationship with travel time, and thus was an important feature to include.
    - Additionally, we can see that certain locations had more travel time reduction than others.
    - We noticed the end locations of San Jose, Santa Clara, and Sunnyvale had the most travel time reduction.
2. Daily traffic speeds from Uber Movement data in Q1 2020 from San Francisco, between OSM nodes
    - It describes daily average speeds along Uber ride paths. (This dataset is the condensed version of Uber's hourly data)
    - Features:
        - The starting node/point
        - The ending node/point
        - The n^{th} day of March 2020 (eg: day 2 means March 2, 2020)
        - The average speed between the start and end nodes.
        - Below is its distribution:



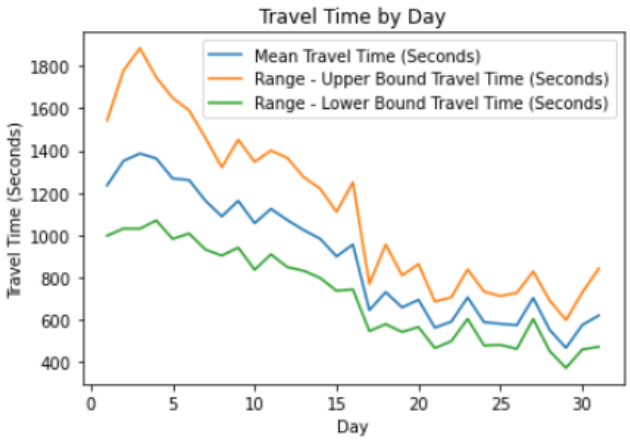
    - This dataset was useful for our hypothesis because we directly saw how speed related to the other features of the dataset.
    - In particular, we used it to compare how plus codes effectiveness versus census tracts, by comparing how uniform each cluster of the respective spatial structure was.
3. Census tracts dividing San Francisco by GPS coordinates

- This dataset further divides regions of interest into smaller zones.
- Features:
  - The name with which the geographical region is referred to.
  - Geometry: A polygon formed by points demarcating the geographical region wherein each point is in the form of (longitude, latitude).
- This is useful for our hypothesis because we can use smaller regions as features in our hypothesis modelling, along with their geographical coordinates.
- Combining this dataset with the data on average speeds and displaying the results on a map shows lower average speeds in the city centre of San Francisco accompanied by gradually increasing average speeds away from the city, in suburbs and larger neighbourhoods.

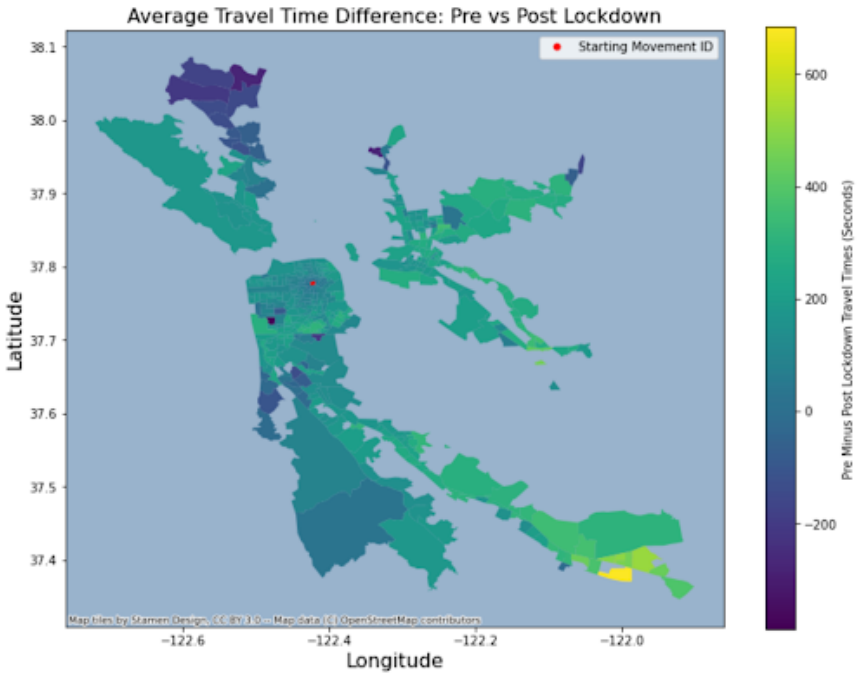
4. Mapping from OSM nodes to GPS coordinates
   - This was used with the other datasets to attach GPS coordinates to the Uber start and end nodes/positions.
   - This was necessary to create heatmap visualizations and compare speeds based on location.
5. Speed Limit Dataset for San Francisco street segments:
   - https://data.sfgov.org/Transportation/Speed-Limits-per-Street-Segment/3t7b-gebn
6. Elevation
   - https://data.sfgov.org/Energy-and-Environment/Elevation-Contours/rnbg-2qxw
   - https://www.kaggle.com/elmadj/analyzing-the-sf-fire-department-calls/data

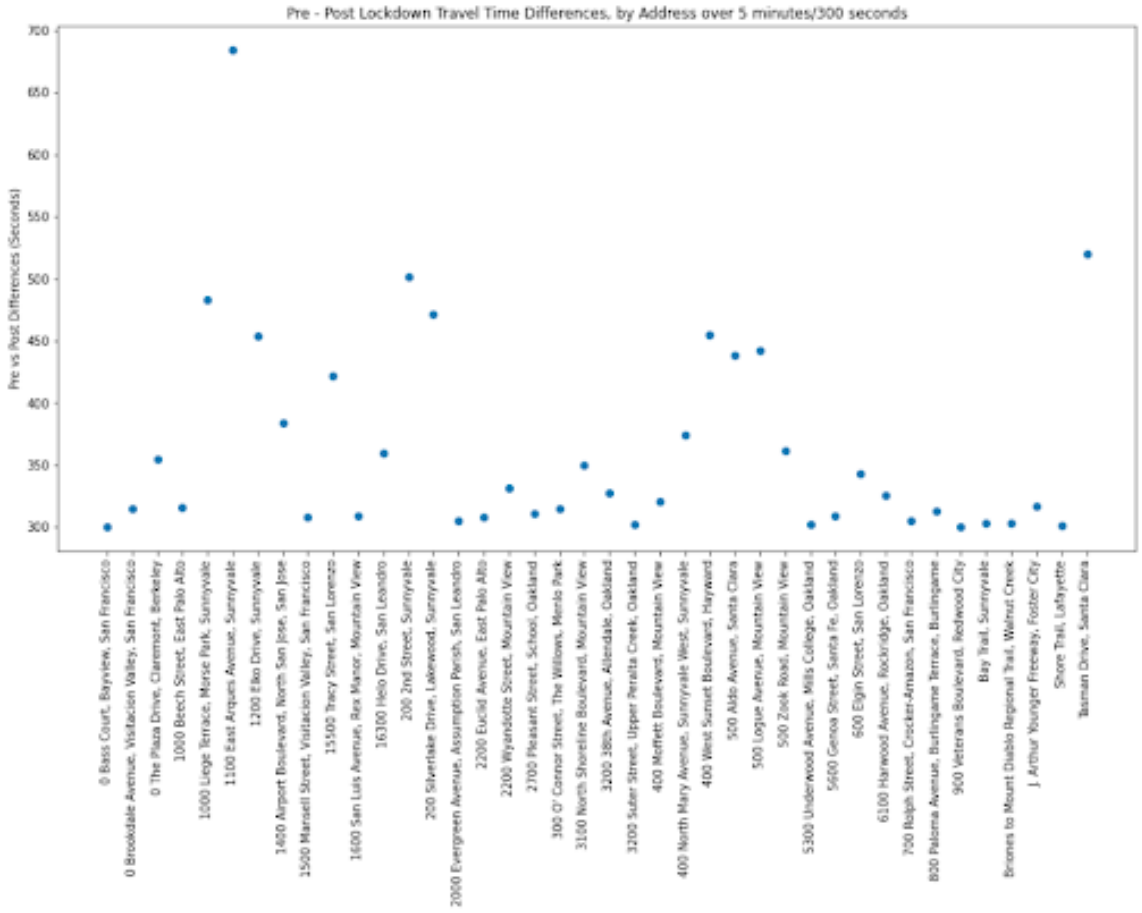
Exploratory Data Analysis

We explored how lockdown affected travel times by graphing the averages of the mean, upper bound, and lower bound travel times, for each day.Clearly, from the graph of the various travel times, we can see that average travel time was decreasing overall, with a sharp spike around the 17th, when lockdown was imposed. Interestingly, travel time decrease was much steeper pre lockdown, and almost plateaued after the initial spike from imposing lockdown. We can see that the mean, upper, and lower bound for travel times follow the same relationship. This further reinforced the fact that day was a key feature that we should include in our model, and most likely will be the most important feature, which we can verify via PCA or regularization.



To investigate how destination travel times were affected by lockdown, we explored how location impacts travel time by creating three separate groupby's - address, neighborhood, and city. This involved regexing the address, which also held the neighborhood, as well as the city. From these groupby's, we plotted the pre lockdown minus the post lockdown travel time for each group, averaged over the entire month. We also created heatmaps for these groups. Below is one of the three groupby visualizations.





Lighter colors represent shorter average travel time, and the red dot is the starting point of all travel. The address visualization showed that for the most part, as we travel away from the main city the average travel time increases. Generally, areas that were very close to the starting point did not seem to have much of a change in average travel times, confirming from our hypothesis that location could be a key feature in determining traffic speed. Interestingly, the northwest region has much longer times, but the southeast region of the map has much shorter times. This is likely because the northwest has a hospital, which is more active in COVID times. We see certain neighborhoods, such as in Sunnyvale, have more differences than others. Again, location, in particular certain groupings of locations, may be key features in predicting average speed, per our hypothesis.

This brought up more questions about the data:

- What locational grouping were the best predictors? Although the above showed certain locations had significant differences, some locations were similar travel times compared to others. We may need to do some more regex/clustering to see if we can surpass census tracts.
- How could we visualize more regions? (Getting it to work for the heatmaps was hard, and we don't have API knowledge/datasets)
- How would these heatmaps look like outside SF, or even the US.

First Model

This model uses multivariate linear regression (supervised) to predict/output mean travel time. For this model, to see how we would do on different dataset sizes, and because using the full dataset killed the kernel due to a merge, we sampled from our dataset and treated that sample as our true dataset. The train-test split is 90-10 since we do not have too much data. This dataset was created by merging speeds_to_tract with times_to_tract.

For all inputs, a row in our data is an uber trip between two census tracts.

Inputs and explanations:

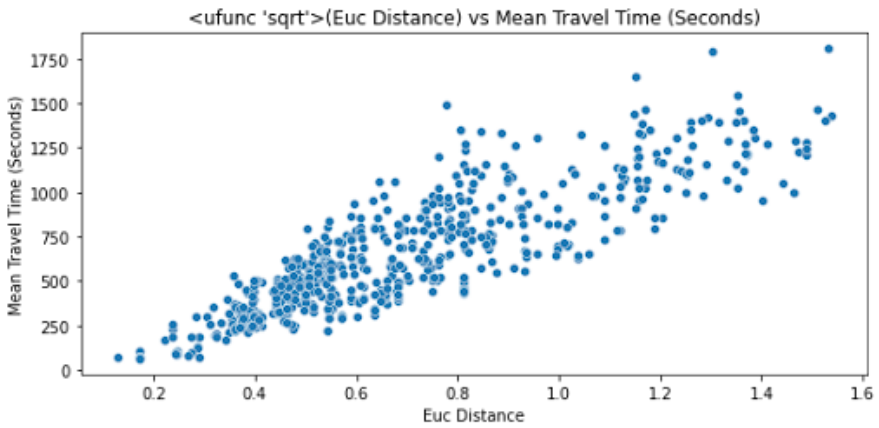
- Euclidean distance - The exact route of the trip was unavailable, so the straight line distance was chosen as a reasonable approximation
- Day - May not be the best feature but as we entered late-March, travel time decreased. May only be reliable for the March period.
- Area of destination's polygon - The larger the destination, the more area that could be traveled. Plus, perhaps the size of the census tract could indicate if it's rural or urban (urban tends to be more crowded, with larger travel times per distance)
- Area of origin's polygon - We thought this would make a difference but then realized all the origins are the same. Thus, this was dropped but would be useful otherwise.
- Weekday Indicator - Generally, weekdays tend to be more crowded in pre-lockdown times. Perhaps that might change with lockdown since people are cooped up inside, which could lead to more weekend traffic post-lockdown.
- Lower Bound Travel Time (Seconds)*
- Upper Bound Travel Time (Seconds)*

*Upper/lower bound travel times would be "cheating" since those are similar to what we're estimating. Tried using them as features in a separate model.

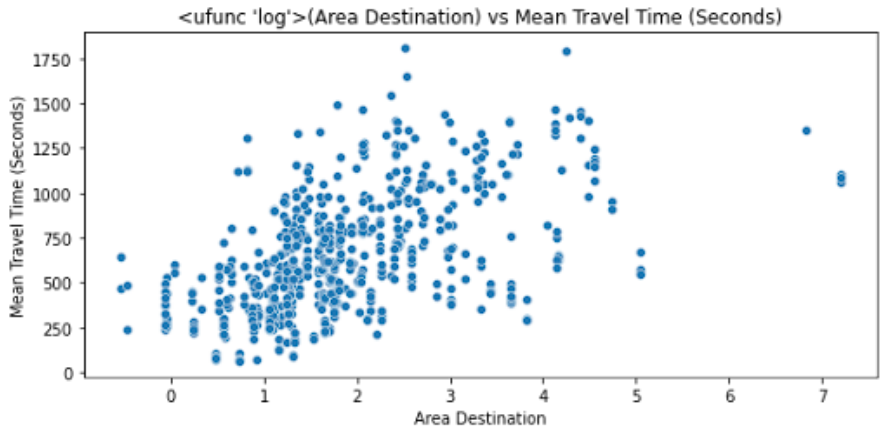
Since we did not have an abundance of features, we did not perform cross validation, since this is more useful if we want to filter out features. This is also why we included day as a feature.

We chose linear regression since we noticed linear relationships between travel time and our inputs. Below is more concrete evidence for why we chose certain features:

- Linearizing Euclidean Distance by square rooting:

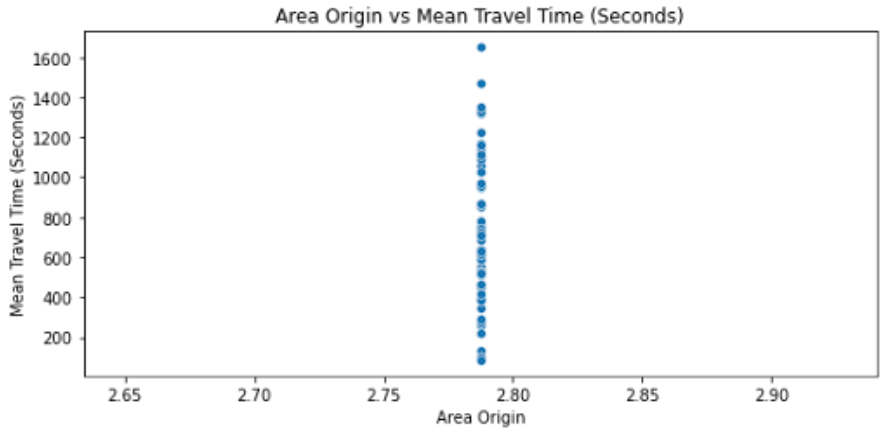


- Linearizing Area Destination using log:

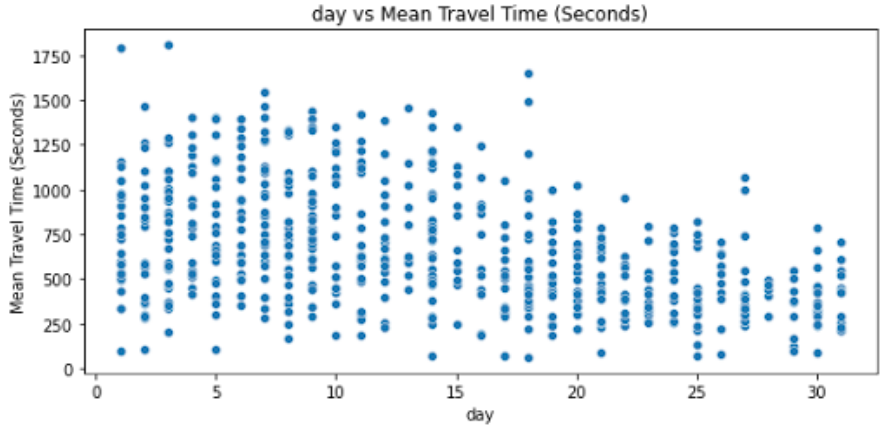


However, others simply could not be transformed into a relatively linear relationship:

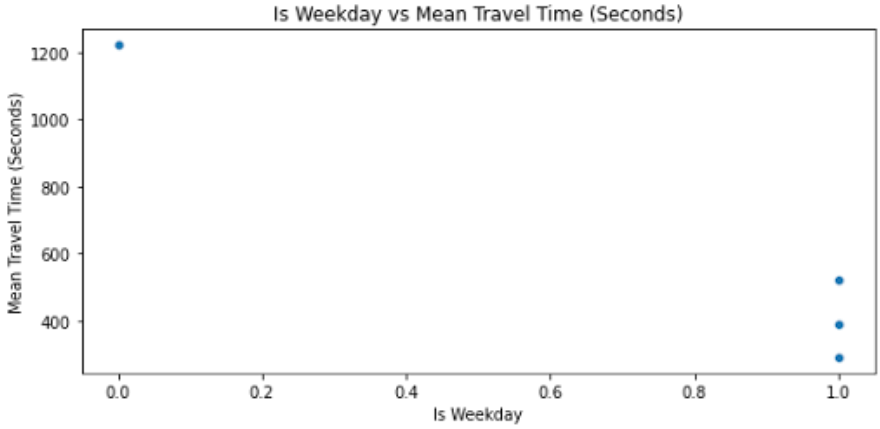
- Area of the origin: will always be the same because all our trips originate from one spot.



- Day: will probably not be easily transformed due to the amount of variance it has (it is probably doable, with a function that squashes the lower left up). However, we didn't bother since day wasn't really a good feature to rely on.



- Weekdays: may be better for logistic regression, but we see they did indeed have lower travel times.



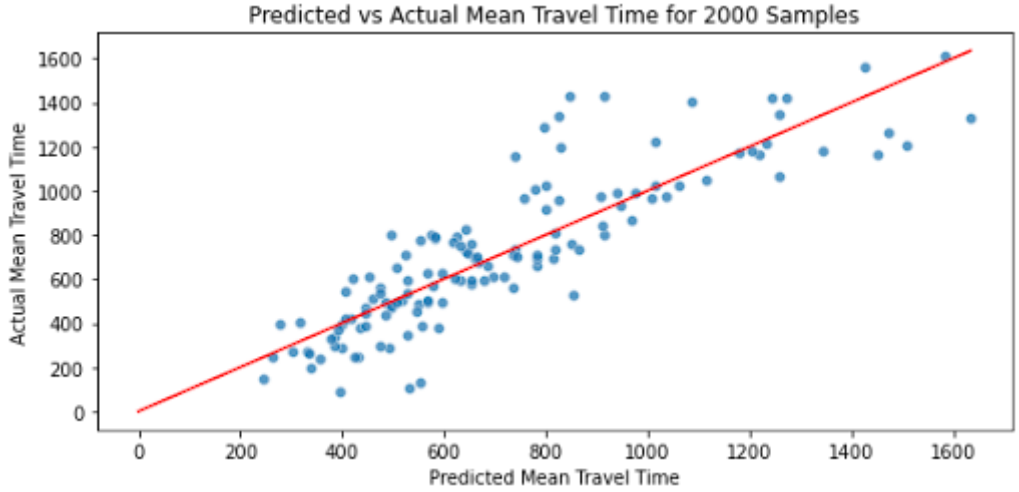
First Model Evaluation and Analysis

This model was evaluated according to its R^2 score and Root Mean Squared Error (in seconds). These are appropriate measures because R^2 measures how linearly related the predicted and actual values are, which is what we are looking for in a linear regression model (the closer R^2 is to 1, the better the prediction). RMSE allows us to quantify in real world terms how far off our values are on average.

The naive implementation (no transformations, 100 samples) was inefficient, with an R^2 score of 0.737, and test R^2 score of 0.294, and RMSE of 289 seconds. This was terrible, since the average of mean travel times is 717 seconds, so our RMSE was 40% of our trip. We would be better off asking the locals how long it takes to get from point A to point B. Plus, .294 is abysmally small, indicating we couldn't predict travel time accurately.

Increasing the number of samples had better results (300 samples): train and test R^2 score increased to 0.781 and 0.758, respectively, with an RMSE of 173 seconds. However, this means that any variations in the data threw off this naive model by a lot.

For the largest sample size we could use without crashing the kernel (2000), the R^2 score was 0.753 for both train and test. Here is predicted vs actual mean travel time for the naive implementation:



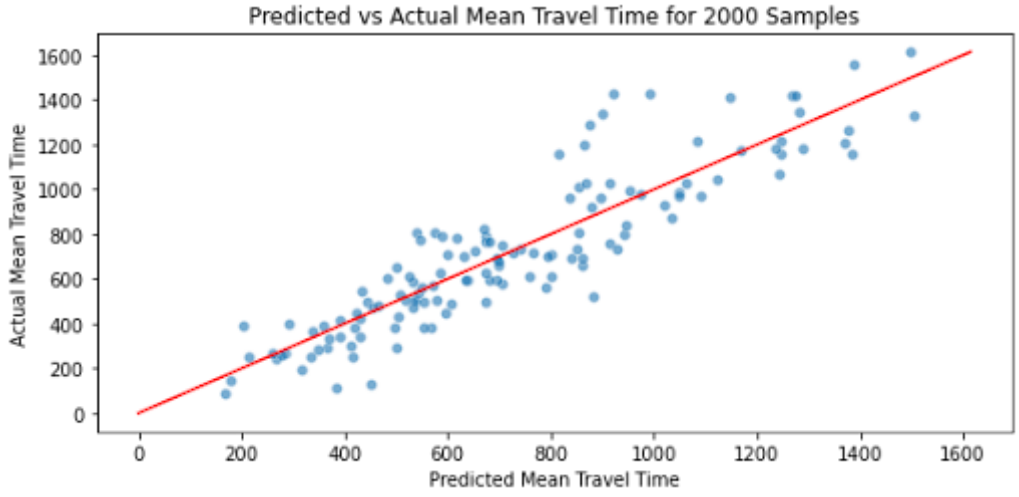
Takeaway/Connection to Project: The red line is where we want points to be, because then we are solving Uber's problem of predicting mean travel time from A to B. Although the model

isn't bad, we can clearly see that a lot of points are close to the line, so our model predictions aren't quite useless, especially considering how little features we have. However, the data is a little wave-y, motivating the following:

Improvement 1:

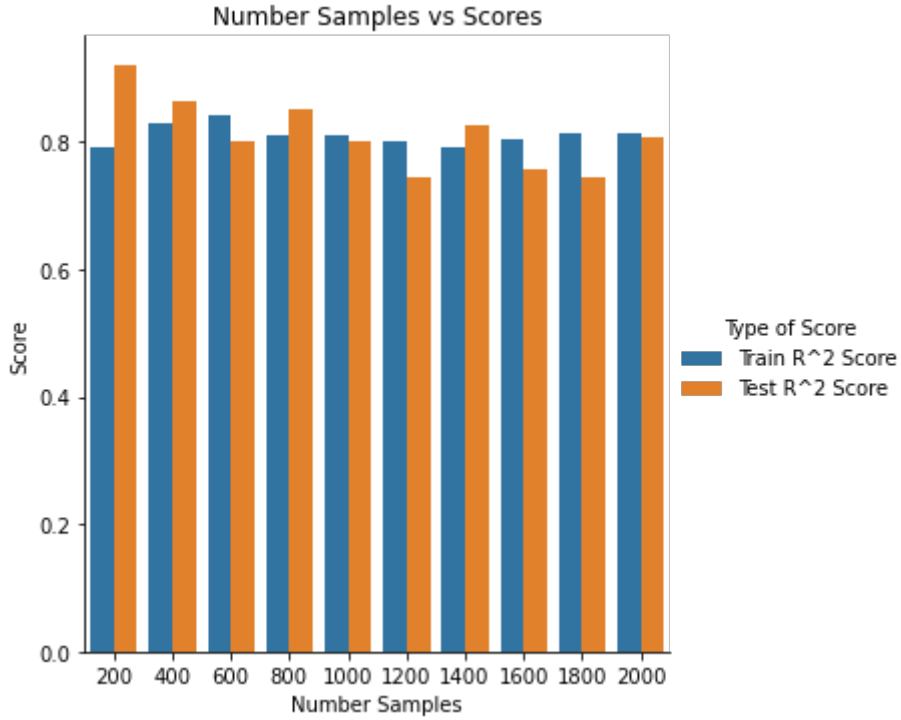
Problem: Again, since some of our data was not linearly related without transformations, we got the waviness above. The EDA from above led to these improvements.

Solution: Transforming the data (log, square root), we get the following plot of predicted vs actual travel time for 2000 samples:



Takeaway: We can also clearly see that the points are closer to the line, and less wavy.

The RMSE for the model on 2000 samples is about 150.5 seconds, which is 2.5 minutes. This is not bad since the average mean travel time is 717 seconds, so this error is about 21% of the average mean trip time.



Result: As expected, our accuracy improves, but we see 200 samples actually did really well. Interestingly as well, we see that as our dataset grows, our test R^2 starts to fall and converge at around slightly below .8 (at 2000 samples it is .806, which is a 5% improvement over our naive model).

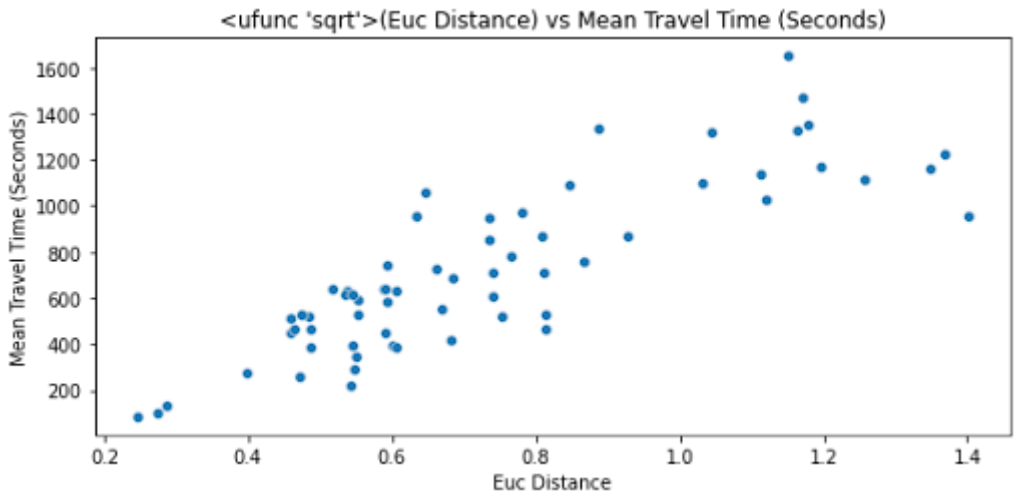
This is interesting because the fact that R^2 doesn't continue to decrease indicates our model is probably robust against the size of the dataset, despite the fact that we had to cut 80% of our data due to lack of memory.

Connection to project: Again, we are trying to estimate within the constraints of our memory - as sample size increases, the dataset gets closer to the actual dataset. The above shows that we are doing fine in that regard.

Improvement 2:

Problem: We still could not really improve accuracy to consistently hit 90%, even with our transformations. Thus, we switched to removing outliers could help, since they tend to drag the line away from other points.

Solution: From our scatterplots, we saw that perhaps we could shave some of Euclidean Distance, dropping everything from 1.23 and beyond, because of the below graph. Moreover, larger Euclidean distances may have shortcuts or less straight routes between the origin and destination, so they could resemble outlier-like points.



Result: Our new RMSE is 152.18, which is about 1.8 seconds higher than without removing outliers. This did not work likely due to the sample used in the scatter plot not being representative of the population dataframe and because minimizing the error of the non-outliers did not offset the increase in error of the outliers. Overall, the tradeoff is probably not worth it, since we have no significant change in RMSE.

Improvement 3:

Problem: We still wanted to consistently hit 90%, and our next guess was some features, such as day, were doing more harm than good.

Solution: We tried using Lasso regression rather than outright dropping features. We wanted to limit the penalty of having a complicated model, motivating our choice of alpha to be between 0.8 and 1.2.

Results: It turns out the default alpha 1.0 resulted in the best RMSE of 152.805, which did not beat our RMSE of 150.5 from earlier. Our understanding is that the day did not actually harm anything, though it would be interesting to see if this is the case with data beyond March. Not worth doing since, again, no significant change in RMSE.

Overall, these improvements weren't really good, but they are very effective for the second model.

Second Model

This model uses multivariate Linear Regression with LASSO regularization. We used this since, again, we noticed linear relationships between travel time and input variables - this time we will predict travel time from past travel times, since our features above weren't enough. We also used the same data to try classification (further down).

Inputs and explanations:

- average travel time from X previous days - to help predict future travel times.
- average speed (mph) from the X previous days - correlated with travel time
- Great Circle distance from origin to destination (miles) - same as the first model, but more accurate in terms of latitude and longitude using Haversine formula (2. Wikipedia).
- Area of the OSM node location - same as first model
- Elevation Change - flatter is generally faster travel times
- Speed Limits - lower means slower travel times
- Is in schoolzone - usually indicates slower travel times

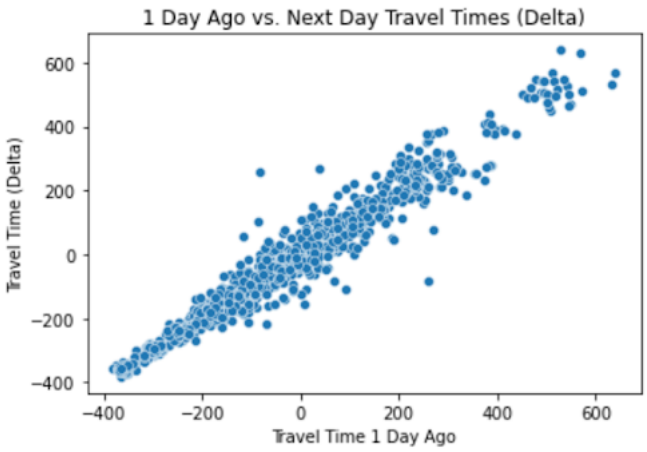
These features were based on the starting location 300 Hayes Street, Civic Center.

Used following datasets:

- Daily traffic speeds from Uber Movement data in Q1 2020 (San Francisco)
  - Feature-engineered the dataset by converting to a time series (described in the guided section). This allows us to include a specified amount of post lockdown days data in the training set.
- Daily travel times from Hayes Valley to all other census tracts around San Francisco in March 2020
  - Converted to a time series.

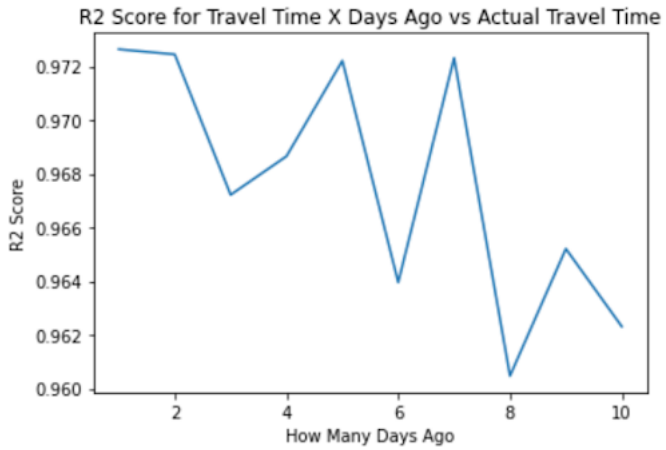
Outputs and explanations:

We predicted average travel time since the lockdown had a (naturally) significant effect on travel times.



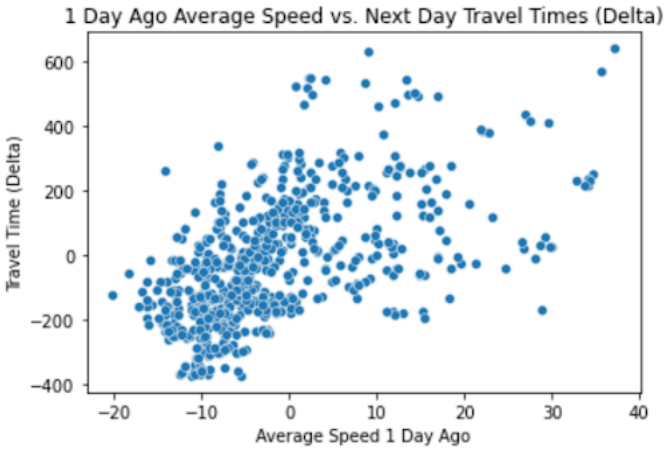
Above is a scatter plot showing the correlation between the travel time from 1 day ago vs the next day. The R^2 score was 0.9726 suggesting these two variables are highly correlated.Note the y axis is the difference between a point's travel time and the average travel time.

The following is a plot displaying the R^2 scores for X days ago vs the day we are trying to predict:



The overall trend is a decrease in R^2 score as we get further away from the day we are trying to predict. This is expected (note the y-axis ranges from 0.972 to 0.960). This is not a large range, suggesting that even travel times from 10 days ago are still highly correlated with the travel times of the day.

Lastly, below is a scatterplot displaying correlation between the average speed and travel time. There is not a high correlation (R^2 score of 0.5609):

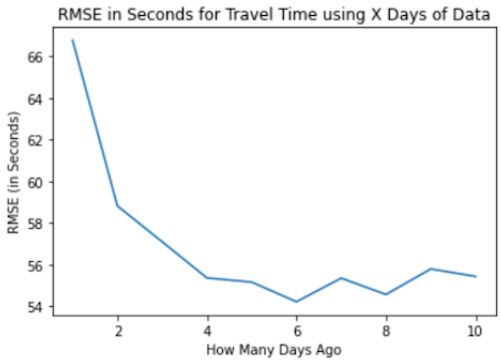


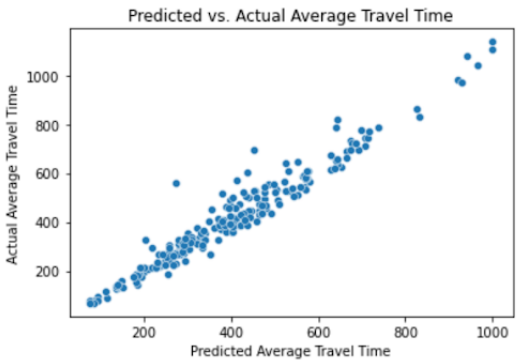
Note: there are negative average speeds because the average speeds are the average speed for each day minus an average speed for a movement ID (delta, same technique used with average travel time).

Model Evaluation Results:

As stated in the first model, this model was also evaluated according to its R^2 score and RMSE (in seconds). These are appropriate for the same reasons.

The resulting model yields a R^2 score of 0.9367 with a RMSE of 51.44 seconds. The model used an alpha value of 1 in order to avoid overfitting the test data and used data from 1-6 days back. This is a great model R^2 score and RMSE value because the R^2 score showed a high result with the predicted vs actual travel times and a low RMSE. In comparison, the first model we built had a R^2 score of 0.753 with a RMSE of 173 seconds which is a significant improvement.





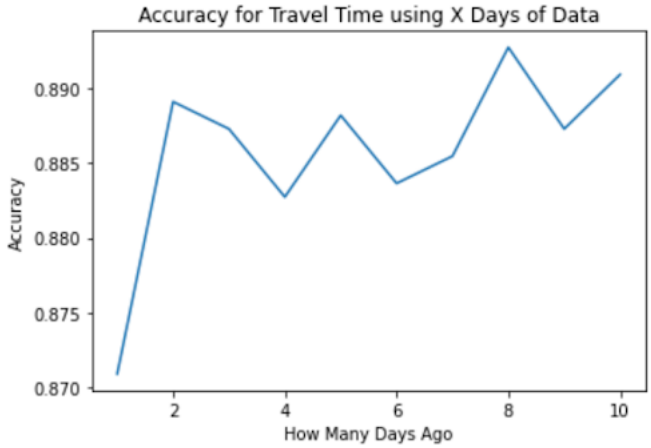
Takeaway/Connection to Project:

The line graph shows the RMSE score in seconds vs the amount of days we use in the model to predict the average travel time. As we increase the amount of days used in our prediction model, RMSE decreases. But when we use too many days, the RMSE increases - this concept is similar to an ROC curve. The optimal value seems to be using 6 days ago. This connects to the project by answering a potential question of whether older average travel time data would cloud the predictions made by the model or not. And as the graph shows, older data can actually improve model accuracy but using too much old data can negatively affect it. An optimum amount of days must be determined in order to optimize the model.

The scatterplot shows the correlation between the predicted and actual average travel time. The main takeaway is that the model predicts average travel time very well, but also shows that the model has a couple points in the range that are far off the line. This motivates further model improvements using classification or multiple models per Movement ID.

Improvement 1:

Problem: In the original model, we saw that there was a high accuracy in predicting the average travel time but there seemed to be points that were still very off from their actual average time. For example, one point was predicted to have an average of 300 seconds travel time but actually had a 500-second travel time.



Solution: We used Random Forest classification, with max depths of 1, 10, 20, 30, and infinite (which actually did not crash the kernel, surprisingly), in the hopes that it might be able to capture nonlinear relationships between variables, by grouping travel time into five minute intervals (i.e. 0-5, 5-10), believing it would allow the model to have a wider margin of error and yield a higher accuracy. (Other than that, the inputs were identical to this model).

Results: Accuracy of around 0.90. This is not as good as our 0.9367 R^2 value, likely because a lot of the travel times were close to the classification cutoffs. For example, a cutoff of 5 minutes could have multiple points close to this number, leading to easily incorrectly classifying points. This is supported by the data, as roughly 60% of average travel times were within 90 seconds of the cutoff values.

Furthermore, we saw that classifying by 1 minute intervals (which is closer to what linear regression would predict, since linear regression predicts the exact minute) resulted in accuracy at best around .53, which was pretty horrendous in comparison.

Moreover, classification is highly prone to overfitting, which is mitigated by having more features and data. Since linear data will look like a staircase in terms of decision boundaries, this explains why the test accuracy is so bad - the boundaries are too overfit. This supports our second hypothesis that classification is not effective given the small number of features and data, and the fact that there are linear relationships between our predicted variable and variables.

Model Improvement 2:

Problem: For the same reason as Improvement 1.

Solution: we decided that another approach could be to build multiple models for each Movement ID. This could get rid of unnecessary data that would be clouding the model's accuracy as well as fit parameters specific to each ID (such as how many days back we should use to predict travel time). Using this method, each Movement ID would have its own model with its own best parameters which could help improve prediction accuracy.

Results: We saw the best R^2 score yet of 0.9913 and a very low RMSE score of 2.11 seconds, proving our first hypothesis. Although this will need some data for new Movement IDs, it is very good at predicting once we get data.

Future Work

We wanted to try classification with more features, since we could get more "fitting" boundaries, since not all our data is linear with our predicted variable. Specifically, some sort of clustering by location seems like it could be a good idea, and we only tried comparing it to Random Forest, not clustering, and perhaps this could disprove our second hypothesis.

We probably try more carefully removing outliers for even more accuracy, or finding more features. It will obviously be hard to top 99.13% accuracy, but these could help our model be more robust to further traffic changes.

Additionally, incorporating the population density of each census tract is another interesting idea. We tried doing that but had issues with merging the existing census tract datasets to external ones, due to differing geometries and naming conventions, but it will likely be correlated positively with travel time, and could improve accuracy even more.