Intersection Congestion Prediction

Shuang Song (ss2627) , Ken Jianhao Wu (jw2585), Pranshu Gupta (pg475)

1. Project Goals and Applications

We've all been there: stuck at a traffic light, only to be given mere seconds to pass through an intersection, behind a parade of other commuters. If the city planners and governments could anticipate traffic hot spots ahead of time they could reduce the stop-and-go stress of millions of commuters. Predicting congestion at intersections within cities has the power to improve safety, optimize operations, and identify opportunities for infrastructure challenges. The goal of our project is to predict congestion at intersections in the major cities in the US.

2. Dataset Description and Additional Features

The dataset we use contains trip-logging metrics by commercial vehicles from about 4,800 unique intersections in four major cities. It contains 27 features and more than 857k entries. These features include city, intersection ID's, coordinates, entry and exit street names, hour of day, weekend or not, month, direction of entries and exits and percentiles for total time stopped, distance from first stop and time from first stop. Most of the time records are zeros. We have much more intersection data for city 2 and 3, less for city 0 and 1. The dataset contains missing values only for the two features containing street names.

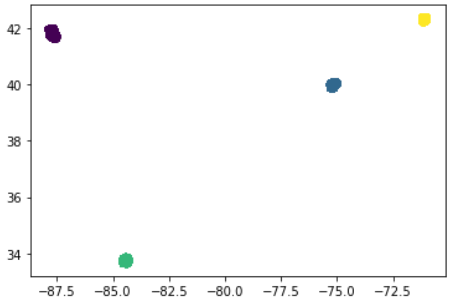
Apart from existing features, we also conducted further research and added additional features which might help in predicting congestion. The first feature we included was the percentage of rainfall/snowfall during that time. Another feature we included was the distance of that intersection to that city's downtown area. Furthermore, we also included the road type for a particular intersection. This road type includes information on whether the road is a street, lane, boulevard, broad, drive etc. Additionally, we included a feature that contains the distance of the particular intersection from the outskirts.

3. Data Exploration, Feature Transformation and Handling Missingness

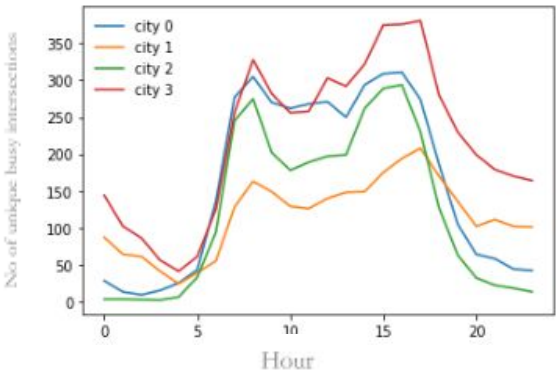
We analyse how many roads are linked to a particular intersection. Some roads might be one way and so the number of entry and exits are counted separately. All intersections have at least one entry and at least one exit. The number of entry and exit streets along with the difference in exit and entry streets for intersections are added to our training set.

For imputing missing values we first split data based on cities and then use Low Rank Models. The street names are label encoded and then imputed using different loss and regularization functions. Alternatively, we also tried using multinomial and one vs. all loss for nominal variables. However, these techniques did not give errors within a reasonable margin and hence the missing names were encoded as 'Unknown'.

Since the directions are related to each other, they are encoded to numerical values. Although there is a city name feature, we check if there are roads in between the cities etc. or mislabelled city name in the data. By using K-means method on the longitude and latitude, we easily cluster the data into four groups, and the graph shows that they are perfectly clustered. This means that we do not have to deal with issues like roads between cities etc.



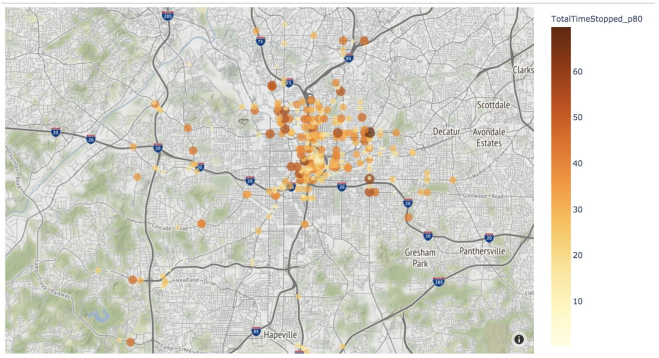
It is further observed that even with the separation of time, the data is highly unbalanced. So we try to find the busy streets first. We restrict it to be at least 30 minutes waiting time on average.

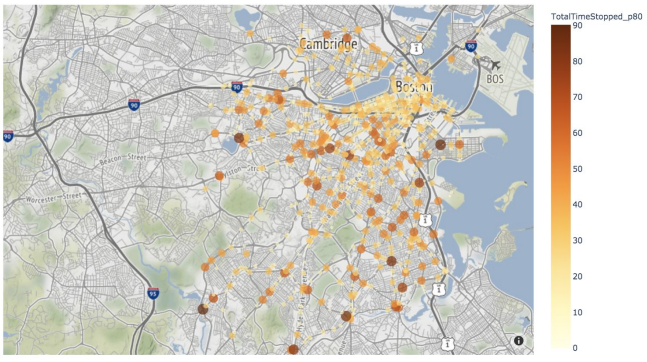


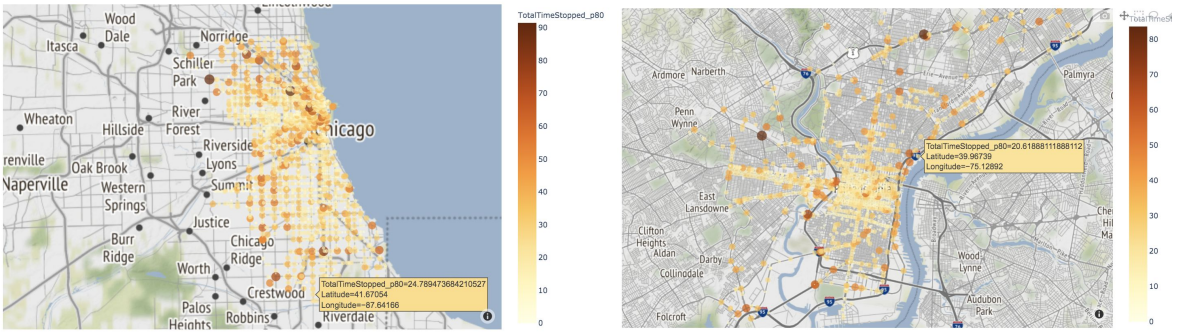
Some interesting observations from the above data and chart:

- Even though for city 0 we only have 973 unique intersections in our data, we see in its peak time, about a third of its intersections are busy. For city 1 we only have 377 unique intersections and at its peak time about half of them are busy
- The number of busy streets are actually closer to each other despite the number of total intersections in that city. A more stringent definition of 'busy' may bring these curves closer

Visualizing the busy intersections in for the cities of Atlanta, Boston, Chicago and Philadelphia (in order):







4. Technique 1: Multiple Linear Regression

Multiple linear regression generalizes linear regression, allowing the dependent variable(Y) to be a linear function of multiple independent variables(X's).

$$Y_i = \beta_0 + \beta_1 X_{1i} + \ldots + \beta_k X_{ki} + \epsilon_i$$

The validity of the model depends on whether or not the assumptions of the linear regression model are satisfied. These are:

- The independent variable is not random. This is true in our case.
- The variance of the error term is constant across observations. This is important for evaluating the goodness of the fit.
- Errors are non-autocorrelated. For this, the Durbin-Watson statistic must be close to 2 which is true in our analysis.
- Errors are normally distributed. If not, then we can't use some of the statistics, such as the F-test
- There is no multicollinearity. We ensure this by looking at the correlation matrix of features.

All these assumptions were rigorously tested by us and then the model was further used. In order to decide on a good model, we used stepwise regression. We tested plenty of combinations of features by adding or removing them one at a time. We added or removed these features based on our logical understanding and also by looking at the dataset. Finally, we selected the one that resulted in the best quality which we interpreted using AIC and BIC.

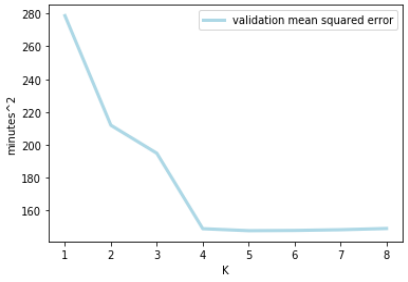
In our study, the outliers are anomalies that are important as they are the ones which are actual congestions. We implemented our model by using different loss and regularization functions with different parameters. The Huber loss is a loss function used in robust regression, that is less sensitive to outliers in data. On the other hand L2 loss is more sensitive to outliers and also provides a more stable and closed form solution. We tested first, huber loss with low values of the parameter delta. This is so that the outliers get penalized according to L1 and other smaller values are penalized according to L2. We further tested L2 loss as well. We finally proceeded with Huber loss with a low value of delta along with little regularization based on our results. Additionally, implementing cross validation enhanced our confidence in the model.

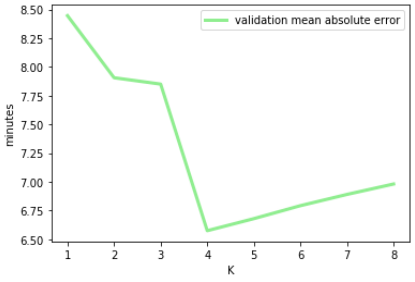
	MSE	MAE	Max Error
L2	226.8210	9.9459	297.3281
Huber	301.2180	7.6911	309.9721

Result and Applications: Linear regression is simple and easy to interpret, and it takes O(1) constant computation time for prediction. It could be used for various tasks. Huber Loss works well on the average waiting time prediction. The mean absolute error is only around 7.69 minutes, compared to 9.95 minutes from Least Squared Errors. On the other hand, Huber Loss method's errors for the outliers are significantly higher than the Least Squared Errors method's. However, the outliers tend to be rare occasions. Thus, we care more about the average waiting time than the specific outliers in this model. In this case, we are more confident to use Huber Loss to predict the 50 percentile waiting time.

5. Technique 2: K-Nearest Neighbour[4]

KNN is a distance based algorithm. It is a non-parametric algorithm, which means it does not make any assumption on the underlying data distribution. The only assumption it makes is that the data points are in a metric space and thus they have a notion of distance. Typically we use KNN algorithm for classification problems, but it could be used for regression problems as well. In our problem, it takes the average of a data ponit's k nearest neighbor's waiting time as the prediction. This is valid because our data lay in a feature space in its nature and using a distance based algorithm is intuitive. The following photos are the trend of the validation mean squared error, mean absolute error and max error as K increases for our baseline model. As we could see the trend, the errors start dropping first then start increasing again.





We used grid search to find the best combination of our parameters, and the best combination is using uniform weights, and euclidean metric. After some tuning and feature selection, we find the following two models to have the best validation results.

	MSE	MAE	Max Error
K = 7	135.4573	6.1995	260.0
K = 8	134.8533	6.2021	263.875

So on average we are about 6 mins off the real traffic time at each intersection, but this is largely due to the outliers as we see maximum error.
Application: KNN provides a good prediction on our problem, and it's simple and easy to interpret. One concern of the algorithm is that it takes a long time to run as its computation requires O(nd) runtime. This makes the algorithm not practical for real time traffic prediction usage, but it can still serve as a good tool for city traffic analysis and civil engineers planning.

6. **Technique 3: Gradient Boosting**[1]

Gradient boosting is a machine learning technique for regression and classification problems, which makes a prediction model as an ensemble of week prediction models. Boosting is an ensemble technique in which the predictors are made sequentially. The subsequent predictors learn from the mistakes of the previous predictors. The intuition behind gradient boosting algorithm is to repetitively leverage the patterns in residuals and strengthen a model with weak predictions and make it better. The only assumption it makes it's the independence of the observations. This is satisfied as the traffic in one intersection does not depend on another intersection. They could be correlated but our observations on each intersection are independent. For our implementation, we used MSE loss as the other two algorithms.

Results: We tested on different parameters using grid search, and we found the best model with parameters [n_estimators=100, max_depth=3, learning_rate = 0.1, loss='ls']. As we will discuss later in the feature importance, we find that our weather data is not useful for prediction and we run the model again without weather data.

	MSE	MAE	Max Error
With weather data	211.6662	9.4953	317.9179
Without weather data	211.5094	9.4929	325.3719

So on average we are about 10 mins off the real traffic time at each intersection, but this is largely due to the outliers as we see maximum error.

Application: Gradient Boosting does not have as good performance as KNN algorithm, but the runtime for prediction is much less. Its prediction computation complexity is only $O(pn_{estimators})$ where p is the number of test data points and $n_{estimators}$ is the number of estimators (we used 100), and it does not depend on the number of training data so the prediction can be done in much less time. So the Gradient Boosting algorithm could be used for real time prediction to get an approximate on which intersections could be busy at the time, or it can be used as a tool for other algorithms to find the feature importance of each feature. We will discuss our feature importance in the next section.

7. Feature Importance

For our Linear Regression model and KNN model, all features are treated as equally important. However, in our gradient boosting model, we were able to find and rank the feature importance of our features. The most important features are hour, weekend, ,number of entry streets, number of exit streets, latitude, longitude and distance to downtown center. Intuitively, the importance of these features does seem to be correct. The hour of day as well as weekend do determine the amount of traffic on the street. For example, at 8am and 5pm on weekdays, most business areas will have huge traffic. When the number of exit streets is less than the number of exit streets or vice versa, there is bound to be congestion. Additionally, downtown may be a crowded area and so may lead to congestion. The least important features are weather(which we removed later as our original dataset only has values for month but not day), month, and city.

8. Weapon of Math Destruction

This intersection congestion model is not a weapon of math destruction. We explore the following three points:

1. Are outcomes hard to measure?: According to our research, outcomes can be measured. We split the data into training and testing sets and are able to predict whether there will be congestion or not quite effectively for our test set. Also our test set contains intersections which are not there in our original training set and our models make effective predictions on them.
2. Could our predictions harm anyone?: Our predictions will enable the city planners to improve areas where there is congestion. This might lead them to pay much more attention and allocate much more resources on those busy areas as compared to the other non-busy areas. Also, this may cause the previously non-busy areas to get more traffic than they can handle. However, by using our model over the whole city and continuously checking and validating can easily address the above concerns.
3. Could it create a feedback loop?: Once our model predicts congestion at an intersection, people might prefer passing through elsewhere and thereby effectively spreading out. It also may lead to a previously busy intersection becoming non-busy and vice-versa. However, (like the models used by Google Maps) if our model is run after updating the data continuously, then the model doesn't create any feedback loop and in fact helps spread out traffic and help the population..

9. Fairness

Our model only predicts waiting time in real values. Even though we can still classify these outputs into {congestion, no congestion} by some threshold. The harm of false positives (actually no congestion) and false negatives (actually congestion) is quite low. These errors will change waiting time in our future data distribution, but this is a normal process of model calibration. Therefore in our analysis, fairness is not an important criterion to consider when choosing a model in our application.

10. **References**

1. Generalized Boosted Models: A guide to the gbm package, Greg Ridgeway, http://www.saedsayad.com/docs/gbm2.pdf

2. Comparison of loss functions for Linear Regression, Publisher:IEEE, V. Cherkassky ; Yunqian Ma https://ieeexplore.ieee.org/abstract/document/1379938

3. An investigation for loss functions widely used in machine learning, Feiping Ne, Hu Zhanxuan https://www.researchgate.net/publication/325634317_An_investigation_for_loss_functions_widely_used_in_machine_learning

4. Comparing K nearest neighbours methods and Linear regression-is there reason to select one Over the other? Arto Haara, Annika Susanna Kangas https://www.researchgate.net/publication/265978662_Comparing_K_nearest_neighbours_methods_and_Linear_regression-is_there_reason_to_select_one_Over_the_other