Solving Real World Problems with Data Science

D. Laskarzewski, Dr. S. Hutt

Introduction

This research attempts to find relationships between the weather, car crashes, and public transportation using publicly available data for the greater Boston area. Key questions included:

- Can the lateness of the light and heavy commuter rails be determined by knowledge of a crash and the current weather?
- Can fatal injuries be predicted by knowing basic crash and weather information?

Methods

Data Collection

Weather data was obtained from Open-Meteo with data points for every hour and every day between January 1, 2019 and January 1, 2023. Hourly variables of interest included temperature, apparent temperature, precipitation, rainfall, snowfall, cloud cover, windspeed, wind direction, wind gusts, and soil moisture as a proxy for previous rainfall. Daily variables of interest included maximum temperature, minimum temperature, total rainfall, total snowfall, precipitation hours, maximum windspeed and maximum wind gusts. All data was specific to the city of Boston, MA.

Crash level data was obtained from the Massachusetts Department of Transportation for all crashes between January 1, 2019, and January 1, 2023. Initial variables included information on basic crash identifiers, time indices, demographic information, environmental variables, and road

descriptors. Data was collected on an individual year basis and compiled together to form the 4-year range.

MBTA data was obtained through the Massachusetts Bay Transportation Authority's Data Portal. The data of interest pertained to the light and heavy commuter rails through Boston. MBTA does provide vehicle travel times between stops, but in looking for terminal-to-terminal travel times, the general arrival and departure events proved to be a greater help. The information provided with each event included the service data, route id, trip id, direction, stop id, type of stop performed, vehicle id and label, event type (arrival or departure), and event time. Bus and commuter rail data were considered as well, but ultimately decided against due to time constraints. It would be an interesting proxy for traffic and general road travel times as an expansion to this project though.

Data Processing

Because the time frames and the variables for the weather data were initially chosen, it was mostly pre-processed with little else to do. The snowfall was measured in centimeters while everything else used imperial measurements, so that conversion was most of the processing done for that dataset.

The crash data from Mass.gov was for the entire state of Massachusetts, not just the Boston area. The data was first filtered down based on location, only keeping the crashes with a location field that matched the city of Boston. From there, the data was farther filtered on number of missing fields. Columns with more than 1000 rows of missing data were removed from the set.

Redundant columns having any overlap with the weather data were also removed from the set, as well as columns in unhelpful formats (dates in string format while dates in datetime were also in the data). Columns with data that are difficult to parse but possibly interesting for research purposes were left in at this time. Examples of this include specific roadway information where the crash took place. Columns similar to this have too many unique values to easily work with but would be another interesting project expansion.

MBTA data processing was arduous given the total quantity of data. The data was separated annually and by quarter, so combination happened before processing. Data of interest was end-to-end travel times of the MBTA rail systems. Given the hundreds of millions of events across the selected years, the data processing was done in SQL instead of Python for the query optimization. The first step was to find the stop ids of the terminals for each line. From there, I attempted to use vehicle ids to find each terminal-to-terminal trip. Unfortunately, MBTA rotates their vehicle ids daily and between terminals. The trip id variable also proved to be inconsistent and unreliable with trip ids not being unique. Additionally, the terminal events were not always recorded depending on the line. While the blue line was good at recording and remained mostly in the Boston area, for the orange, red, and green lines, stations closer to downtown Boston were selected instead of the actual line terminals. This provided much more reliable data and made matching trips much more doable, as well as aligning more closely with the crash data for urban Boston. Ultimately, the query used to pair departure events with arrival events required matched vehicle ids, matched vehicle labels, different stop ids, matched service dates, matched direction ids, an arrival time later than the departure time, and a matched trip id where possible. With the occasional inconsistency in the pairing from missing or duplicate vehicle ids and trip ids,

additional processing was required to ensure the arrival matched the departure correctly. Occasionally the query would match multiple departures to the same arrival, in which case the departure closest to the arrival was correct and used. The other longer trips were associated with different trains that failed to record their arrival events. This mostly happened in March 2020 during the start of the pandemic when information recording became sparser. There were a few thousand of these duplicate trips that were discarded out of a few million matched trips. "Backwards" trips were also discarded – although the trains run both directions, the northbound platforms had an occasional southbound trip and vice versa. There were no more than a couple hundred backwards trips in the dataset.

Once the endpoint events were matched into trips, the overall travel times for each line were averaged. The travel times for every hour from 12:00am Jan 1, 2019, to 11:00pm Dec 31, 2022, for each line were also averaged. The number of trips that were taken in each hour and used in averaging were also added to the dataset. From there, it was possible to normalize the travel times into seconds above or below the line average per hour. In order to use this information with the machine learning methods, the normalized travel times needed to be categorized to be "early" or "late" depending on how fast or slow the average was compared to average. A few thresholds were tested for this, including 3 minutes on either side of average and 5 minutes. I attempted 10 minutes as well but there weren't enough data points to oversample for training.

Data Compilation

Crash and weather data were combined in the wide format pairing hourly weather data from the hour of every crash. Crash categorical data was then broken down into a variety of dummy

variables for each category. The raw hourly average travel times of every rail line was then added for the associated crash hour, as well as the number of trains observed for each line and a normalized travel time for the hour relative to the overall average travel time. The columns if a train qualified as “early” or “late” were also appended. It’s worth noting that the crash and weather data alone is longer than when combined with the MBTA data as the MBTA trains do not run 24 hours per day and couldn’t always be paired with a crash.

Compiled Data Overview

The overall fully combined dataset including the additional dummy variables from the crash data resulted in 200 fields with 12,757 rows. Because of the hourly nature of the weather data and the MBTA data, rows are defined by crashes. Every crash has the weather and MBTA data for the hour associated with it. This dataset was used for predicting train lateness and thus only includes crashes during hours where MBTA trains were running.

Note: I still get overwhelmed by descriptive statistics on such a complex dataset. I’ve been practicing still in Tableau and R and can describe a couple variables well at a time, but when there are so many fields, I’m not sure yet what’s important to include here. This is one of the areas I’m behind on but working on improvements with. If you have time for feedback on what you’d like to see from descriptive statistics on this dataset, then I’d really appreciate that as a starting point for working on sets as complex as this. I know grades are due and it wouldn’t matter towards that, but I would like to improve and get feedback for later with this section. I thought about doing descriptive statistics similar to what I tried in the weekly slides for each data

type – one for crashes, one for weather, and one for MBTA, but I don't know how accurately that reflects the refined combined dataset actually used.

Machine Learning Methods

Three different machine learning methods were used to attempt to predict the answers to the research questions. A baseline Dummy classifier is used as well as a Random Forest and XGBoost. SMOTE was used to oversample the interesting variables in every case since there was so much data skew. I found 4 folds worked the best for training and testing the model with more folds resulting in insufficient data to be used to oversample per fold. A Shapley Interrogation was performed on the results of the XGBoost classifier to see which of the many features had the most weight in the end prediction. I gave all possible fields of information for each test, although being selective about factors that determine rail lateness could be interesting too.

For the Rail Lateness tests, each rail line was run individually through the classifiers to predict the lateness of that route specifically. From there, the per-line results were averaged across the folds, leading to single Accuracy, Recall, Precision, and AUROC values for each classifier. For the sake of the overarching research questions of predicting overall train lateness, these values were then averaged across all train lines. This allows one Accuracy, Recall, Precision, and AUROC value to be used to describe the entire experiment, providing one result for each classifier for easy comparison across methods.

Results and Analysis

As mentioned, the results from the classifiers were broken into Accuracy, Recall, and Precision. The differences between them are subtle but significant and will be contextualized within each specific result below, but a general overview is as follows:

Accuracy – How often was the model correct overall?

Precision – How many trains were predicted late correctly out of all late trains?

Recall – For all late trains, how many did it find?

Rail Lateness (3 minutes)

These were the most successful results because there was more data to oversample with SMOTE per fold. Below is a brief look at the compiled results across all lines.

Classifier	NumFeatures	Accuracy	Precision	Recall	AUROC
DummyClassifier	115	69.96%	4.58%	28.71%	49.67%
RandomForestClassifier	115	95.96%	72.93%	13.95%	92.70%
XGBClassifier	115	96.66%	65.12%	45.55%	94.43%

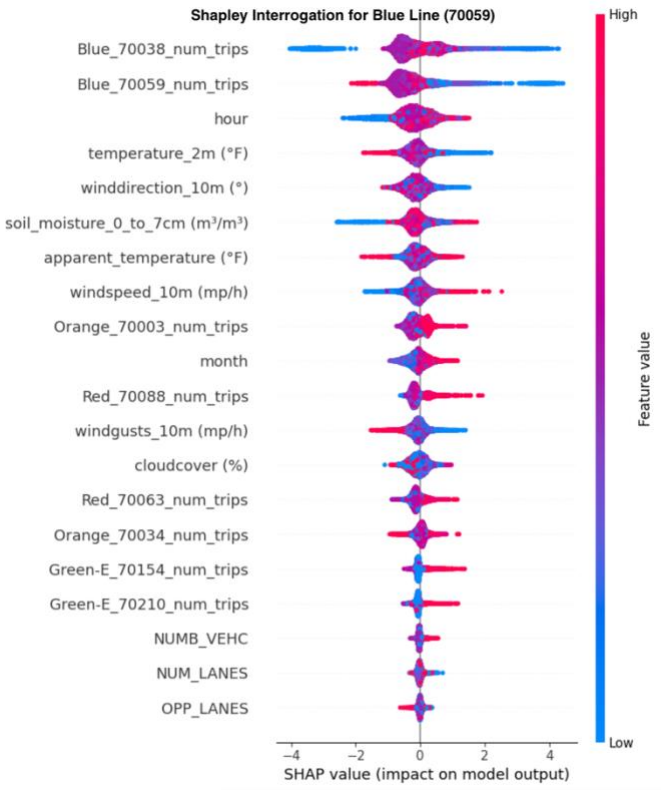
Looking closer at this, we can see the Dummy classifier performed as expected with an AUROC of almost exactly 0.5, the equivalent of a coin flip. At first glance the Random Forest and XGBoost classifiers do significantly better with these predictions, both with AUROC scores over 90% and very high accuracy overall. As outlined above, an accuracy score of around 96% means the model predicted correctly 96% of the time, as seen in both the Random Forest and XGBoost results. Interestingly, the Dummy predicted with fairly high accuracy, although it's worth noting that each of the training folds were oversampled but not the test fold, so the precision and recall scores were significantly lower than the accuracy in all instances.

The Random Forest results show a significantly higher precision than recall. This means it's correctly predicting 73% of the late trains, but only finding 14% of all late trains. I would consider this a more conservative model – it's very selective about which trains it thinks will be late (only finding 14% of them) but is usually right when it thinks one is late (73% of the time). This could be a sign of overtraining on oversampled data. If there were more late train data samples, the oversampling might have had a greater variety among the variables. With Random Forests, variable consistency can lead to selective and rigid decision paths, looking for very specific criteria based on the oversampled amounts. Since the test data hasn't been oversampled, fewer of those data points match with the decision tree, making the model less inclined to think a train is late. Once a data point does come close to lateness criteria, it's fairly accurate given the rigidity of the decision path.

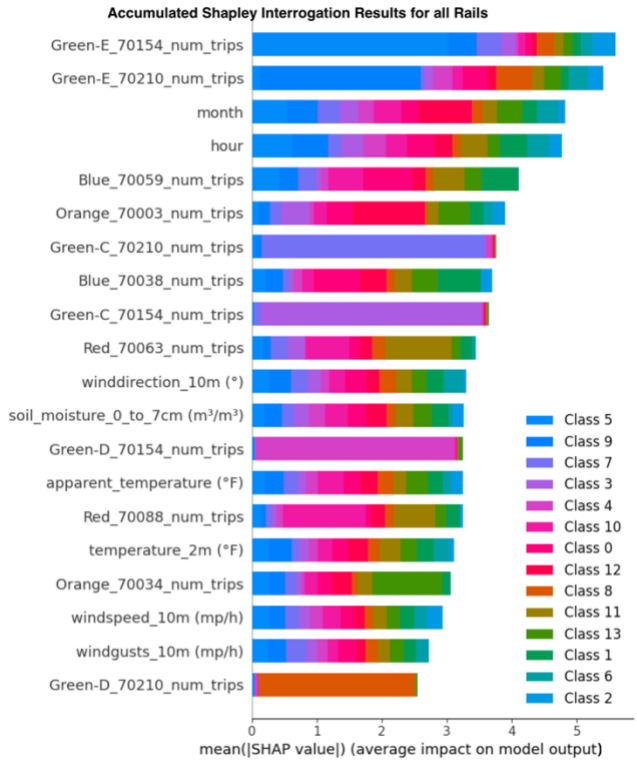
XGBoost performed very well overall, especially given the oversampling in the training data but not the test data. Part of this can be attributed to the model complexity, with XGBoost generally performing better than the Random Forest in more complex situations. The precision of this classifier almost matched the selective precision of the Random Forest by correctly predicting 65% of all late trains. Notably, it was also able to find over 45% of actual late trains, a significant improvement over the Random Forest.

The Shapley Interrogation yielded some very interesting results for each line. Unsurprisingly, one of the most influential factors in train lateness was number of trains running on the line at that time. Below is the Shapley plot for the most impactful fields in predicting the lateness of the Blue line from the 70059 station (chosen arbitrarily).

The results here show some interesting variables having surprising weight in the model. While the number of trips on the blue line is to be expected, the number of trips of a handful of other lines are also surprisingly influential. The other impactful factors are mostly time and weather related which makes logical sense. At the bottom are a few crash related variables that would generally influence the amount of car traffic and could play a role in number of commuters on public transportation like the trains. Variables like cloud cover and soil moisture are surprising to see so high on the list, although they show up for almost all the lines with larger amounts of data.



I also accumulated all Shapley numbers into a full rail predictor Shapley plot, shown here. Each Class represents a different rail line and can mostly be seen reflected in the number of trips for that line. The other variables are similar to what's seen in the Shapley for just the blue line with one notable exception: there's no crash data in

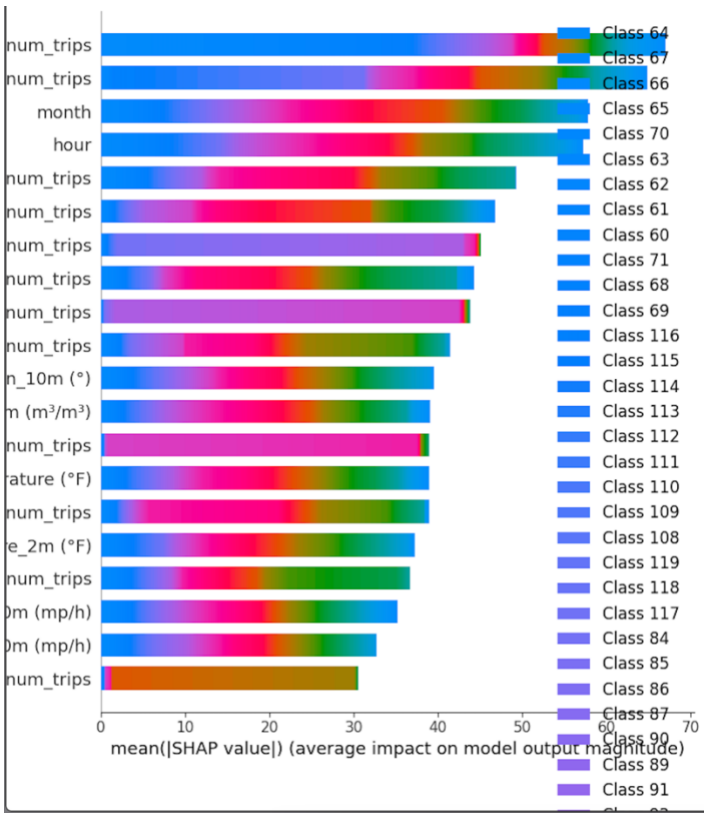


the aggregate. That isn't to necessarily say that crashes don't influence the rail system, but they don't seem to play a very large role when looking at all the rail lines at once. They do seem to be more influential in some lines than in others, but as an aggregate the number of trains and the general weather matters most. These results are not necessarily surprising and still help to answer the research question:

> Can the lateness of the light and heavy commuter rails be determined by knowledge of a crash and the current weather?

Based on this data, it seems that the lateness can be predicted fairly well overall and weather influences that prediction more than crashes.

Note: Just for fun, I also ran a Shapley interrogation across every training and testing fold on the three-minute schedule. I know these results aren't very helpful or meaningful, but I thought it was interesting to see what the model was considering as it was training. The plot is formatted extremely poorly because of the sheer amount of data, but the results are identical to the plot above that only used the testing fold. I was a bit surprised to see



completely identical results, I thought all the training models would have some slight variation on popular factors. It's possible that's the case and they're just not in the top list.

Rail Lateness (5 minutes)

This was a less successful test overall because of how sparse the data for trains being at least five minutes late was. The overview below shows similar results to the 3-minute tests.

Classifier	NumFeatures	Accuracy	Precision	Recall	AUROC
DummyClassifier	115	72.36%	2.29%	28.42%	50.27%
RandomForestClassifier	115	97.99%	36.80%	7.13%	90.89%
XGBClassifier	115	98.33%	54.18%	28.82%	94.13%

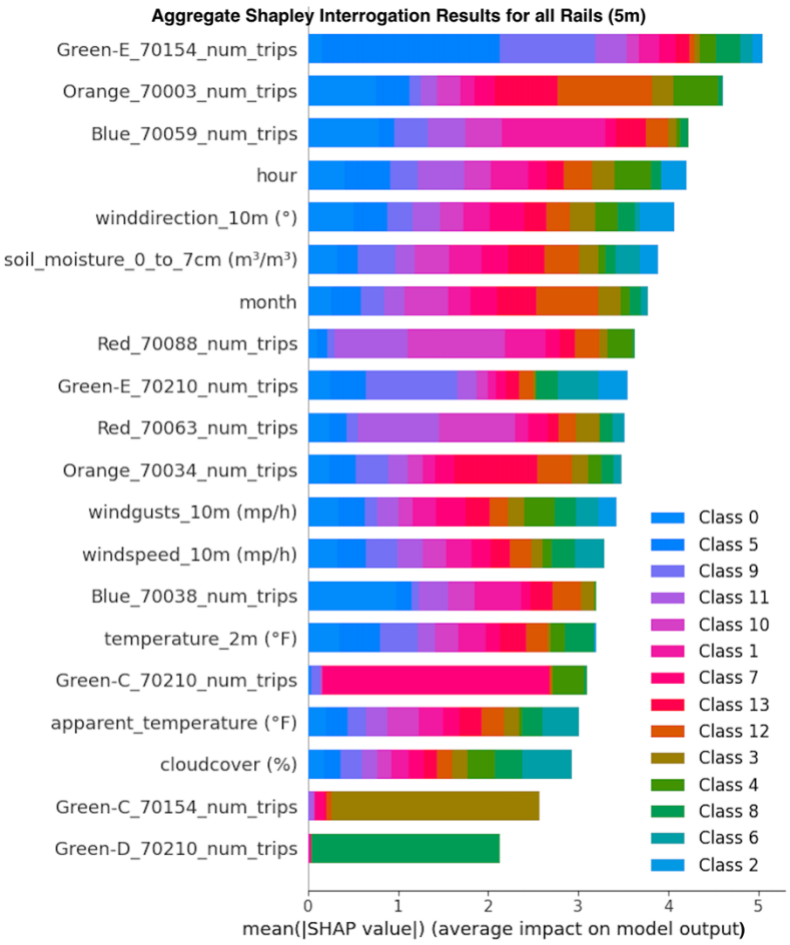
The Dummy once again performed exactly as expected with extremely similar numbers to the 3-minute runs. The Random Forest and XGBoost results are also similar with some interesting marginal changes.

The Random Forest accuracy improved over the 3-minute by over 2% at the cost of almost half the precision and recall. This model correctly predicted 37% of late trains but only was able to identify 7% of all late trains. I do think a lot of this is attributed to overfitting the oversampling. There were significantly fewer trains that were a full 5 minutes late in reality, so the oversampling was likely very specifically fitted to those samples. I think the reasoning is like the explanation from the 3-minute but exaggerated here. The model predicted much worse overall given the smaller sample size in the test fold.

The XGBoost classifier stayed quite strong, gaining a little accuracy also at the cost of some precision and recall. Overall, given the high accuracy and AUROC and relatively strong

precision and recall over the Random Forest, I do think this model is still usable to explain some train lateness. The data is much more specific, with 5-minute late trains few and far between compared to 3-minute late trains, but it was still able to find over a quarter of the late trains and identify over half correctly. I think this has more potential utility, especially if the model worked with some reliability on a 10-minute frame. Three minutes late just isn't very significant in practicality even though the model predicts it more reliably.

The aggregate Shapley Plot for 5-minute late rails had very similar results overall to the 3-minute rails. Wind direction and soil moisture play a bigger role, both of which could be proxies for larger storms that rolled through with a lot of recent past precipitation. There are more differences between this and the 3-minute Shapley Interrogation than I expected with how much more relevant the weather events were despite being the same fields, although it does make sense that weather plays more of a potential role in more significant train delays.



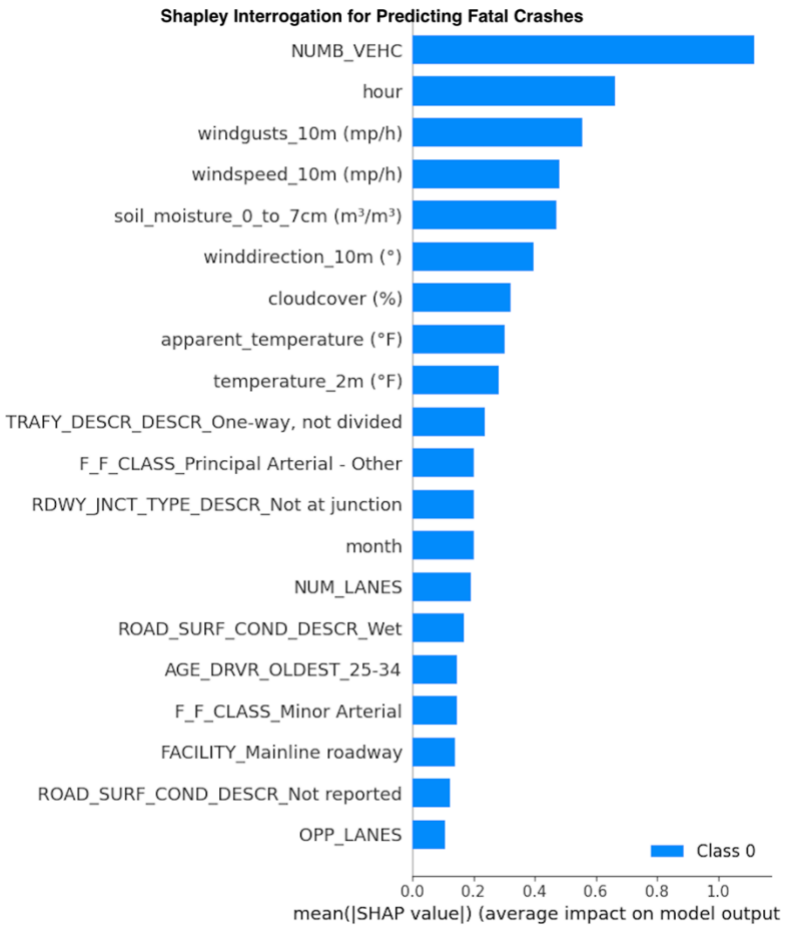
Fatal Crashes

For this analysis, I used the dataset without the MBTA data since it had more rows of relevant crash data. The goal was to predict if there were any fatalities as a direct result of the accident based on other crash information and weather conditions. A result summary is below.

Classifier	NumFeatures	Accuracy	Precision	Recall	AUROC
DummyClassifier	87	74.72%	0.59%	27.06%	47.76%
RandomForestClassifier	87	99.38%	12.50%	1.19%	79.09%
XGBClassifier	87	99.34%	20.83%	3.19%	80.96%

Despite the very high accuracy and solid AUROC score, these results have quite low precision and recalls. I think the complexity here worked in favor of XGBoost again, but it still was only able to identify 3% of fatal crashes. The high accuracy is likely due to the extremely low number of fatalities as a whole, allowing the model to guess there are no fatalities correctly over 99% of the time. The precision of 21% isn't terrible given how well fit the data must have been from oversampling, but a 3% recall is contextually concerning. If a system like this were used in practice, it would likely be better to slightly overestimate the harm done to get the required emergency services on site faster. A 3% recall means the model is guessing more conservatively and could have dangerous consequences in practice.

The most interesting part of this model was the Shapley Interrogation. It's nice to see predictors that make logical sense being used and which weather conditions impact crash severity the most. I think soil moisture here is being used as a proxy for past precipitation and could possibly indicate slicker roads. It's again interesting to see how much of a role wind plays in the decision though but most of the other predictors are expected and reassuring to find here.



Discussion and Final Thoughts

This project has a lot of farther potential from this point. A huge part of this research went into data pre-processing and learning better techniques for data manipulation. Given more time, I would have loved to incorporate buses and the other MBTA public transportation systems that are more influenced by traffic patterns. Adding historical traffic information like what's found on Google Maps would also allow for more in-depth research questions especially pertaining to a tradeoff between commuter traffic and public transportation delays. Adding more weather information, like weather in the last 6 hours or the last full day could also have more of an

impact than initially thought, especially with the importance of soil moisture in all the models above. For crashes, having a larger sample size would allow for more variation and less oversampling monotony like found in the fatality question. Working in the specific road where the crash occurred would also have practical implications when predicting injury and public transport time.

This is still a personal interest of mine and I'll likely continue working on these questions moving forward (after a short graduation break). I was very happy to get more experience working with more complex SQL queries and datasets, learning Tableau and introductory R, and getting more practice with pandas. I've continued the tutorials for Tableau and R and have enjoyed playing with the included datasets although I still have some trouble with datasets as complex as this one (hence the lack of those graphics in this report). Moving forward I'll be working on presenting huge datasets in clearer ways using these tools and would love to update this report to reflect that progress as I make it. The data analysis side of the project was challenging but rewarding and I would love to do more data work like that. I also appreciated the machine learning side of the project; it was very interesting reading about how the different classifiers work and having a template program for moving forward. Getting predictable and logical results felt good and was a great way to check if the model was working as expected, especially for an introductory program. I think some of the research here that's more centered around injuries and fatalities for crashes could really be beneficial in larger cities in anticipating the resources needed at a given crash and refining this dataset could be a good start in getting there.