Predicting severe collision events in Seattle

Sayan Das

# Introduction and background

The safety of drivers, passengers, pedestrians and property is an ever-growing concern when it comes to the operation of vehicles on the road. With **6,452,000 motor vehicle crashes** and **37,133 crash-related deaths** in 2017 alone, significant costs in terms of life, money and property are incurred as a result of collisions each year.

To make things more complicated, the number of vehicles in the US keeps growing every year, creating a need to adopt greater safety measures on the road. Insurance companies play a significant role in minimizing and handling automobile collisions, and they do so through services provided by charging insurance premiums. These insurance rates could be more accurately predicted based on factors such as location, weather, road and light conditions, date/time, whether the driver was DUI etc., most of which are dependent on the location.

In this report, we use collision data from the city of Seattle to implement a machine learning model to predict the severity of collisions using the above variables and others. Furthermore, we explore the variables in detail to find patterns and insights related to the geography of the area, trends across time and traffic variables.

Some of the biggest insurers in the US such as *Statefarm*, *Esurance* and *Allstate* have stated location as one of their top criteria for determining rates. These and other insurance companies could use this model to determine more accurate rates for their customers. Moreover, they could also provide useful information to their clients such as looking out for red flags when buying expensive cars in more accident prone locations which would likely increase their insurance premiums. Another audience for this model could be rideshare companies as well as companies such as *Google* and *Apple* who provide mapping apps. Furthermore, traffic control departments could collect this data firsthand and make it available to the aforementioned companies.

# Data description

The data was acquired from the [City of Seattle Open Data Portal](https://data.seattle.gov/) and consists of **212,760 instances** of vehicle collisions in Seattle while the timeframe of the data spans from **2004-2018**. Spanning mostly the Seattle area, the data for each variable was extracted using *ArcGIS REST API* in *.csv* format and made available on the [Seattle GeoData](http://data-seattlecitygis.opendata.arcgis.com/datasets/5b5c745e0f1f48e7a53acec63a0022ab_0/geoservice) page.

A number of collision event features such as date of collision, location of collision, type of collision (sideswipe, rear-end, parked car etc.), number of people involved, number of fatalities, junction type (intersection, driveway junction, mid-block etc.), weather conditions, road conditions, light conditions, collision description etc. are described which are analyzed in this report.

In total, **40 variables** are included in the dataset but this number will be reduced to include only the ones crucial for building the model. As location would plays an important role in the modeling process, addresses for each instance were extracted using Python’s *reverse\_geocoder* library and added to the dataset. This library was used with the latitudes and longitudes as inputs to retrieve the neighborhood for each instance, stored in a variable called *Neighborhood* in the dataframe.

The *Tomtom API* was used to retrieve the free-flow traffic speed at the closest road for each coordinate. This data was retrieved as a *JSON* file which was then parsed using the *requests* and *json* libraries in Python and finally stored as *Speed* in the dataframe. Additionally, the *HERE API* was used to extract the road length and road congestion data. Both these sets of information are estimated using the closest road to the provided coordinate and stored as *Road Length* and *Road Congestion* respectively in the main dataset.

Since the data covers collisions only in the Seattle area and majority of the features are location-based, the results of the model for predicting severe collisions cannot be generalized to the US. However, similar analytical techniques can be adopted for different cities and states across the country as well as other countries to create useful models and gain insights through exploratory data analysis (EDA). It can be assumed that random sampling was adopted during the retrieval of this data for purposes of any statistical analysis.

# Data wrangling and cleaning

The original dataset called *Collisions.csv* was loaded as a dataframe into a Jupyter Notebook, hosted by the *SageMaker* machine learning (ML) platform on *Amazon Web Services (AWS).* Once the data was loaded, the shape of the dataset was confirmed to be 212,760 rows by 40 columns.

## Selecting initial variables

An initial glance at the dataset revealed certain variables to be redundant, contain many missing values or contain many unique categories, all of which would provide little information during further analysis or during the modeling process. The following variables were excluded from our dataset with the reasoning provided beside each one:

## Redundant variables

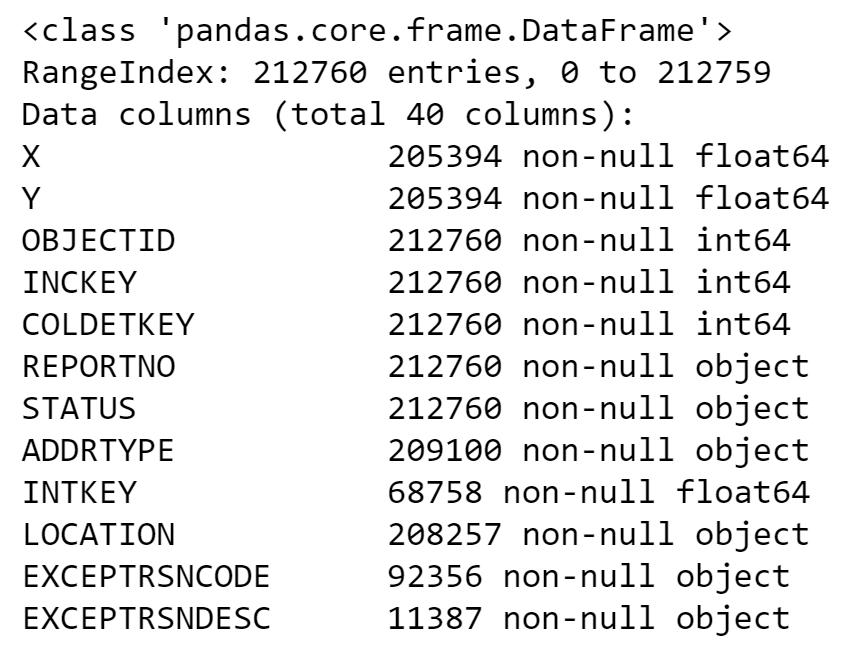
* *LOCATION*: Provides specific and descriptive addresses which is difficult to interpret. Instead, the *Latitude* and *Longitude* variables are used for more accurate location data.
* *INCDATE*: Only contains date information while the variable *INCDTTM* contains both date and time data making it more useful for analysis.
* *SDOT\_COLCODE*: Contains­­ the same information as *SDOT\_COLDESC* which contains more detailed information.
* *ST\_COLCODE*: Contains the same information as *ST\_COLDESC* which contains more detailed descriptions for each collision type.
* *SEVERITYCODE*: Contains the same information as *SEVERITYDESC* except it contains character codes instead of texts. Hence, *SEVERITYDESC* provides more information regarding the different severity levels which can be converted to discrete values.

## Variables with many unique categories

* *OBJECTID*: Contains the same values as the default dataframe indexes with as many unique values as the number of rows in the dataset which would not provide much information for our model.
* *INCKEY*: Contains unique values for each collision incident similar to *OBJECTID*.
* *COLDETKEY*: Same reasons as above
* *REPORTNO*: Same reasons as above
* *SEGLANEKEY*: Contains more than 2000 unique categories making it difficult to analyze
* *CROSSWALK*: Same reason as *SEGLANEKEY*

## Variables with many missing values

The number of missing values for each feature was determined using the *df.info()* method in Python which provided a list of all the variables along with the number of non-null values in each as well as the data type of each variable providing a good overview of the structure of the dataset. *Fig 1* shows a snippet of the output.



**Fig 1.** Snippet showing the number of missing values and data type for each variable

Using the above table, variables with many missing values were excluded. The complete list of such variables is provided below:

* *INTKEY*: Only 68,578 instances
* *EXCEPTRSNCODE*: Only 92,356 entries.
* *EXCEPTRSNDESC*: Only 11,387 entries.
* *INATTENTIONIND*: Too few entries - 29,116. Filling the missing values will not make sense as there are not enough meaningful values to use as reference in the first place.
* *PEDROWNOTGRNT*: Only 4983 entries.
* *SDOTCOLNUM*: Only 127205 entries.
* *SPEEDING*: Only 9492 entries.

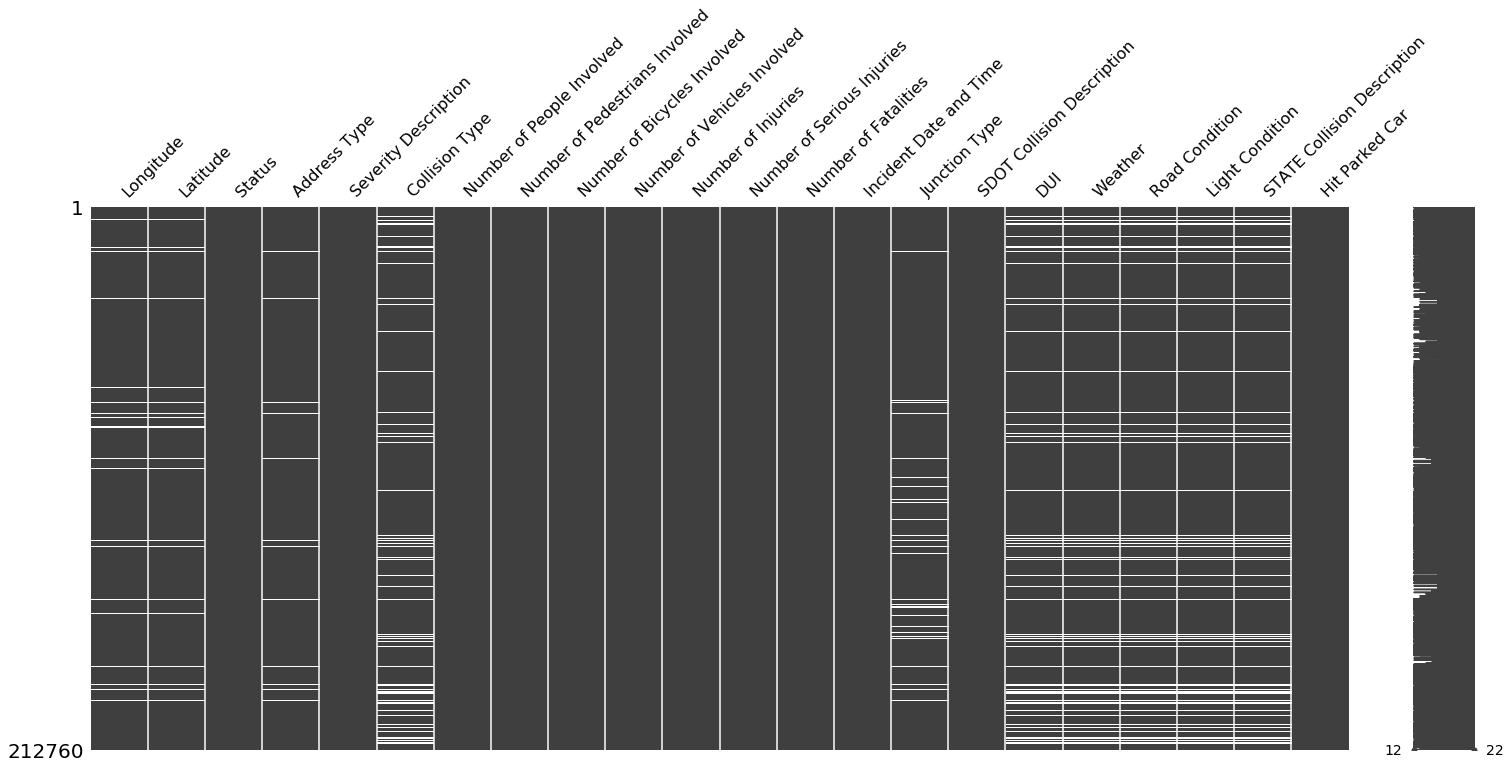
Following the initial variable filtering process, 22 important features remain in the dataset with 18 having been removed.

## Renaming columns

Following the removal of the undesired features, the columns for the remaining variables are renamed to make them more human-readable and less cryptic. For example, the name *INCDTTM* was converted to *Incident Date and Time* whereas *SDOT\_COLDESC* was changed to *SDOT Collision Description*. Note that ‘SDOT’ stands for Seattle Department of Transportation.

## Handling missing values and formatting

From the remaining 22 variables, 11 contained missing values which was visualized using the *missingno* library in Python. This provides a clearer understanding of the missing values relative to each variable as shown in *Fig 2*.



**Fig 2.** Visualization of missing values across all variables in the dataset.

Several patterns were noticeable from the image above. Variables such as *DUI, Weather, Road Condition* and *Light Condition* have missing values for essentially the same rows. Another variable that matches these variables in terms of missing value but is not immediately obvious is *Collision Type*, which is the seventh column from the left.

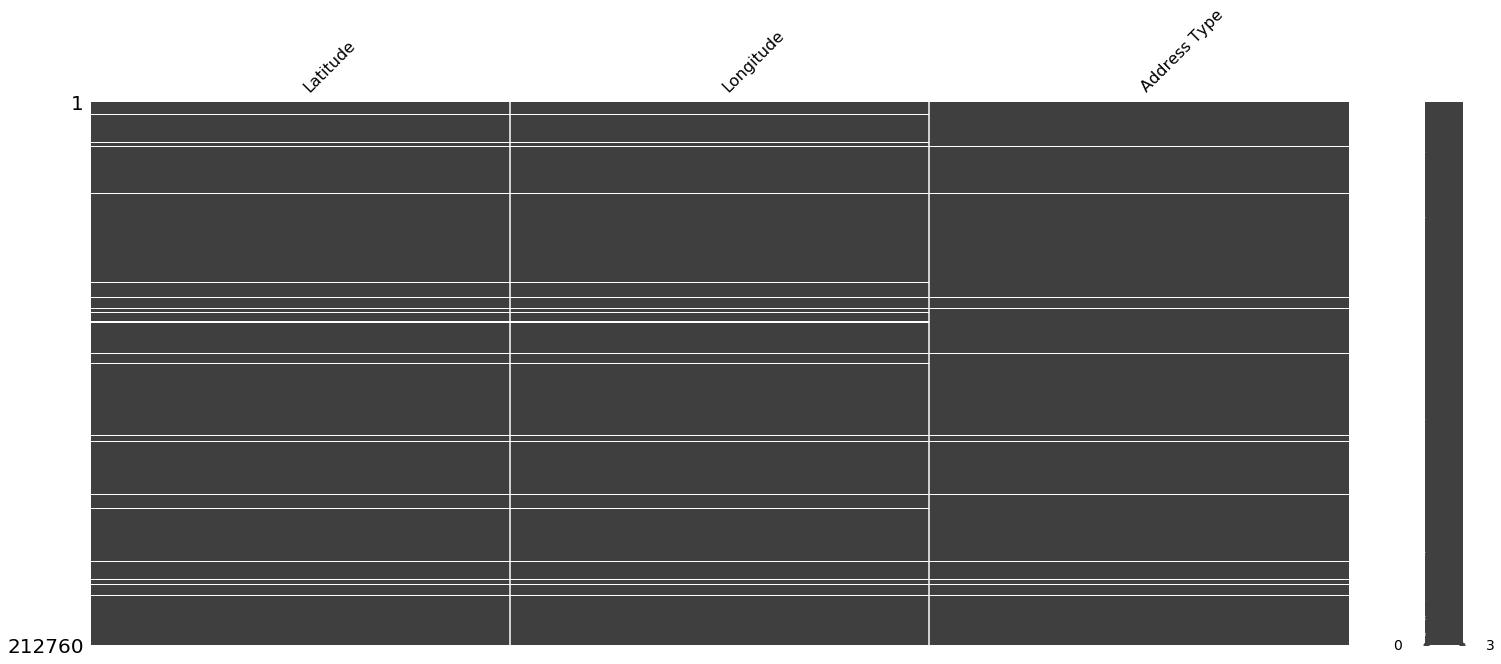
Variables *Longitude* and ­­­*Latitude* share the same missing values while *Address Type* shares missing values for certain rows but not all. *Junction Type* consists of a unique set of missing values which will be interesting to handle. Other variables such as *Status*, *Severity Description* and *Number of Injuries* contain almost no missing values. Variables containing very few missing values cannot be distinguished well using this matrix but this information can be confirmed from the table formed using the *.info()* function.

## Latitude and Longitude

Since only 7,366 instances were missing for this variable, these observations were simply dropped as the total number of instances would still be 205,394 which is significant. Attempting to fill in these values would be a tedious task provided they are specific location data.

## Address Type

Creating the missing values matrix again, only for coordinates and address type, provides a clearer depiction of the relation between the variables in terms of null values as shown in *Fig 3*.



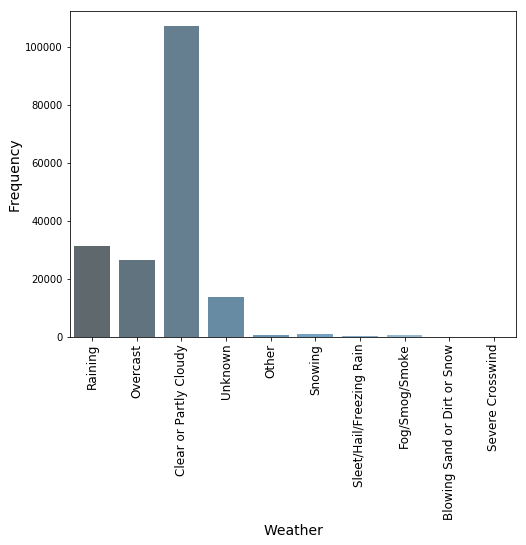
**Fig 3.** Visualization of missing values for latitude, longitude and address type.

In the plot above, we can notice that the missing values for Address Type are contained within the missing values for the coordinates. This would indicate that if we dropped the null values for the coordinates, that would likely result in the null values for Address Type also being dropped. This was confirmed by verifying that the total non-null values for *Address Type* was 205,394.

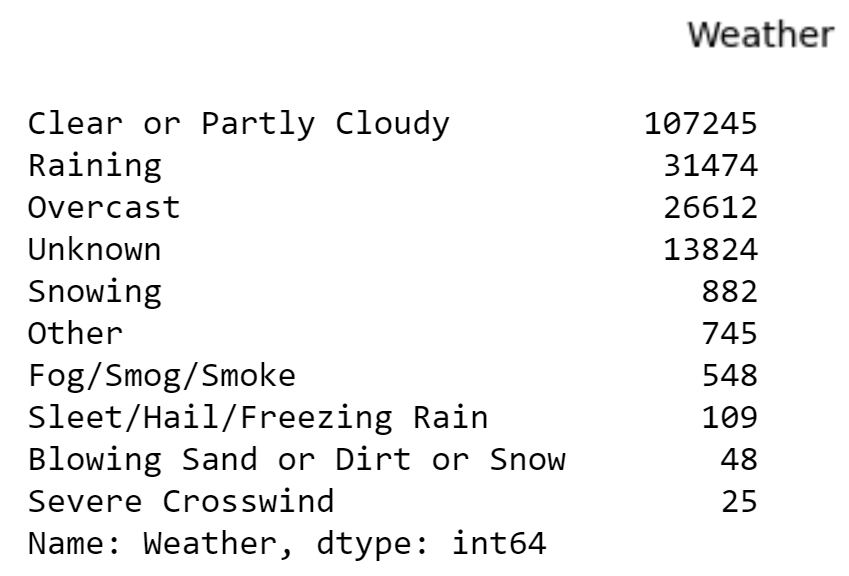
## Weather

The weather-related variables provided the greatest challenge when it came to dealing with missing values. Not only did they contain a decent amount of null values but also contained categories such as ‘Unknown’ and ‘Other’ which essentially represented missing values. *Fig 4* shows the distribution of categories for *Weather* while *Fig 5* provides the actual frequencies for each category in a table.

*Fig 4* was generated using the *countplot* method of Python’s *seaborn* library while *Fig 5* was created using the *value\_counts()* pandas dataframe method.



**Fig 4.** Visualization of weather categories

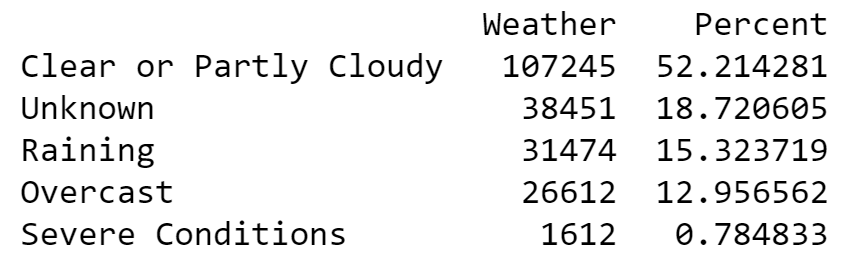


**Fig 5.** Table showing frequencies for each weather category

Several cleaning techniques were adopted for this variable. All missing values (23,882 instances), the ‘other’ category (745 instances) and the ‘unknown’ category (13,824 instances) were grouped into a single category called *Unknown* consisting of 38,451 observations.

Since categories such as ‘Snowing’, ‘Fog/Smog/Smoke’, ‘Severe Crosswind’,*‘*Sleet/Hail/Freezing Rain’*and ‘*Blowing Sand or Dirt or Snow’ had distinctly lower frequencies as compared to other categories they were grouped into a single category called ‘Severe Conditions’, making the categories more interpretable. This was done using the *.replace()* pandas dataframe function.

Also, due to the low frequencies, the median of the category frequency distribution was not affected which is important for modeling purposes. *Fig 6* shows the frequencies of the newly formed weather categories.



**Fig 6.** Frequencies of new weather categories

The table above indicates that despite combining all the severe categories into one, it’s share of instances within the variable is only 0.785%. Another noticeable category is ‘Unknown’ which has now risen to a sizeable 18.72% and is the second largest category. Removing this category would eliminate a sizeable chunk of instances. Instead we decided to keep this category.

## Road Condition

Like *Weather*, Road Condition also contained a significant amount of missing values as well as some categories which could be combined for better interpretation and handling. Once again, the ‘unknown’ and ‘other’ categories were combined with the null values to form a ‘Unknown’ category. ‘Snow/Slush’ and ‘Ice’ were combined into a new category called ‘Snow/Ice’ whereas ‘Standing Water’ and ‘Oil’ were combined with the existing ‘Wet’ category. Finally, ‘Sand/Mud/Dirt’ went into the existing ‘Dry’ category.

## Light Condition

Categories ‘Dark - No Street Lights’ and ‘Dark - Street Lights Off’ were combined into a single all-encompassing category which represents lighting in the dark with the absence of street lights. ‘Dusk’ and ‘Dawn’ are combined into a single category called ‘Dusk/Dawn’ since both refer to similar lighting conditions. The biggest difference between the two are the actual times of the day when they occur but our Time variable should account for that variability.

## DUI

This variable had 2 different types of binary distinctions when it came to whether the driver was under the influence or not. The first binary distinction contained the classes ‘Y’ and ‘N’, while the second one contained ‘1’ and ‘0’. Since It was self-explanatory that ‘1’ corresponds to ‘Y’ and ‘0’ corresponds to ‘N’, we simply replaced the letters to the numbers which will not only come in handy during the modeling process but is also easier to understand. It also contained 23,705 missing values which were converted to the ‘Unknown’ category.

## Junction Type, Collision Type and SDOT Collision Description

For these variables, the only cleaning involved adding the missing values to an ‘Unknown’ category.

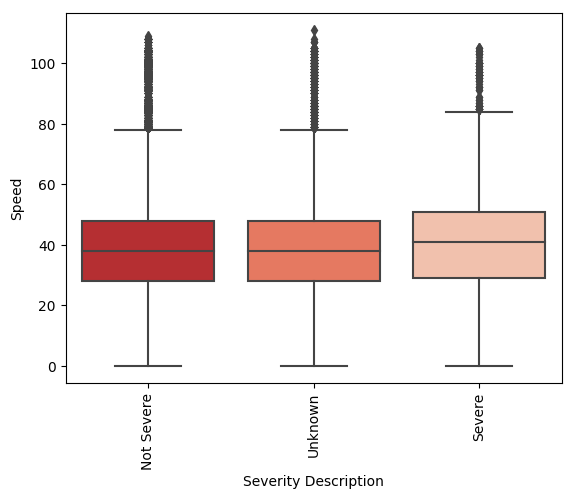
## Severity Description

Since this is the target variable and what we are trying to predict, it was crucial to handle it appropriately. Fortunately, it contained no missing values but the categories had to be re-aligned. An initial look at the frequency for each class revealed that the more severe categories such as ‘Serious Injury Collision’ and ‘Fatality Collision’ had much fewer instances than the non-severe cases – 1.42% and 0.15% respectively. This caused a class imbalance problem which would make it challenging to predict the severe classes.

Therefore to increase the severe class representation, the ‘Fatality Collision’ and ‘Serious Injury Collision’ categories were combined to form the ‘Severe’ category whereas ‘Property Damage Only Collision’ and ‘Injury Collision’ were converted to a ‘Non-severe’ class. Another benefit of having binary classes is that it simplifies the problem and makes the variable easier to work with. It would also make the results more actionable by making the interpretation of the results easier.

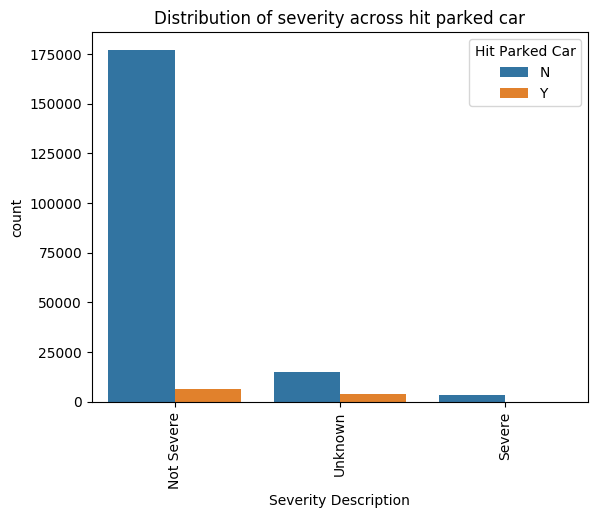
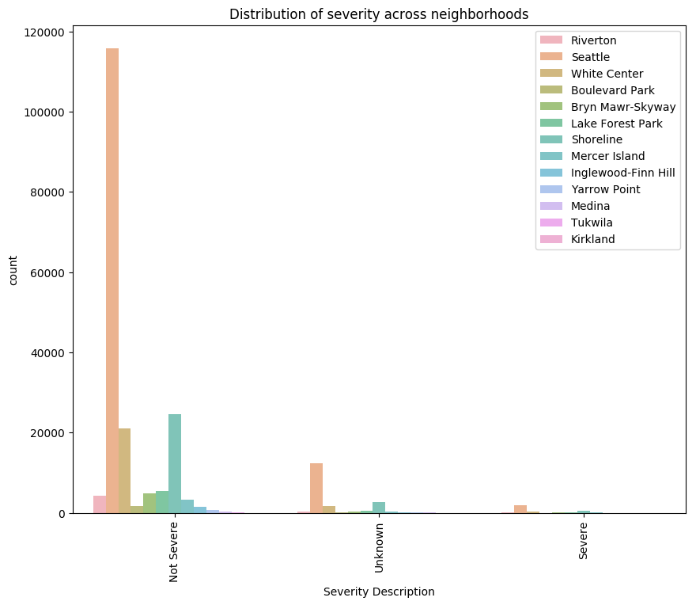
Finally, we also had to deal with the ‘Unknown’ category for this variable. This category was combined with the ‘Not Severe’ category for the following reasons:

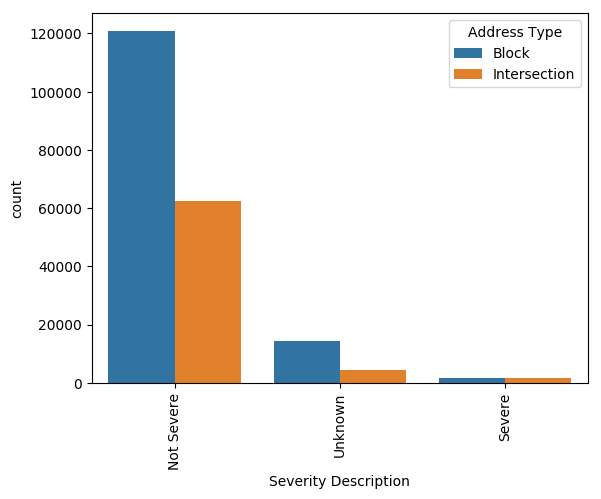
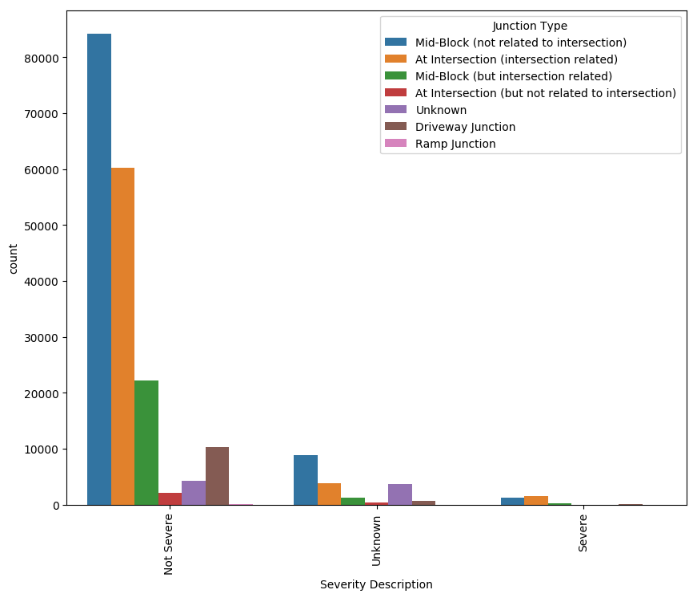
* For crucial variables such as *Speed*, the distribution for the unknown category was similar to the non-severe one, both of which have a median of roughly 40 miles per hour. The median for severe is distinctly higher, see *Fig 7*.



**Fig 7.** Distribution of severe categories for the speed variable

* The distributions for the ‘Unknown’ category, as seen from the multivariate analysis of several variables with the severity variable above, are more similar to the non-severe distributions than the severe ones. For example, if we look at variables such as ‘Hit Parked Car’, ‘Neighborhood’, ‘Address Type’ and ‘Junction Type’, the distribution of the ‘Unknown’ category matches more with the non-severe class, see *Fig 8*. Hence it seemed more rational to combine these two categories.

**Fig 8.** Countplots for Hit Parked Car, Neighborhood, Address Type and Junction Type against Severity

* Also, there are more chances of mis-classification if the unknown category is combined with severe as that would push that category from ~1.5% to ~10% which is a big jump and would potentially provide false results. On the other hand, combining with the non-severe category would increase the size from ~89% to ~98% which would not have such a major impact on the results as the non-severe category is already much larger.

## Transforming date variables

The *Incident Date and Time* variable was split into *Year*, *Month*, *Date* (day of the month), *Day of the Week* and *Hour* variables. This was done by first converting the *Incident Date and Time* variable from ‘string’ to ‘datetime’ format using the *.to\_datetime()* function in pandas.

After that, the aforementioned new date and time variables were extracted from *Incident Date and Time* using attributes of the datetime format. For example, the year for a datetime value can be extracted by using the *.year* attribute. The other variables were extracted in similar fashion and had integer values. The *Day of the Week* variable was ordered from 0-6 with Monday being the first day.

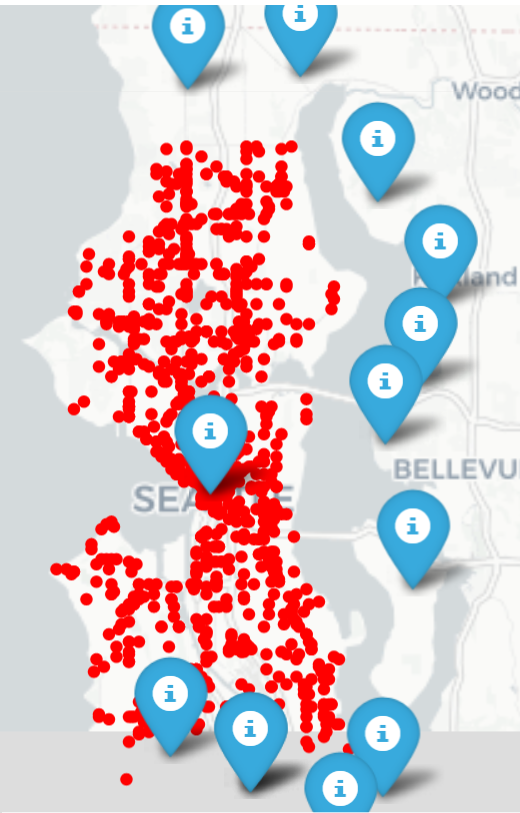
## Dealing with outliers

The only variables where we had to deal with outliers were *Road Length* and *Road Congestion* as they contained a few ‘999’ values which were purposely added in order to catch error returns from the HERE API during data retrieval.

# Exploratory data analysis

## Geographical analysis

In this section, the coordinates are used to visualize the crash locations across Seattle in order to get an idea of the areas of high collision density as well as areas of greater severity.

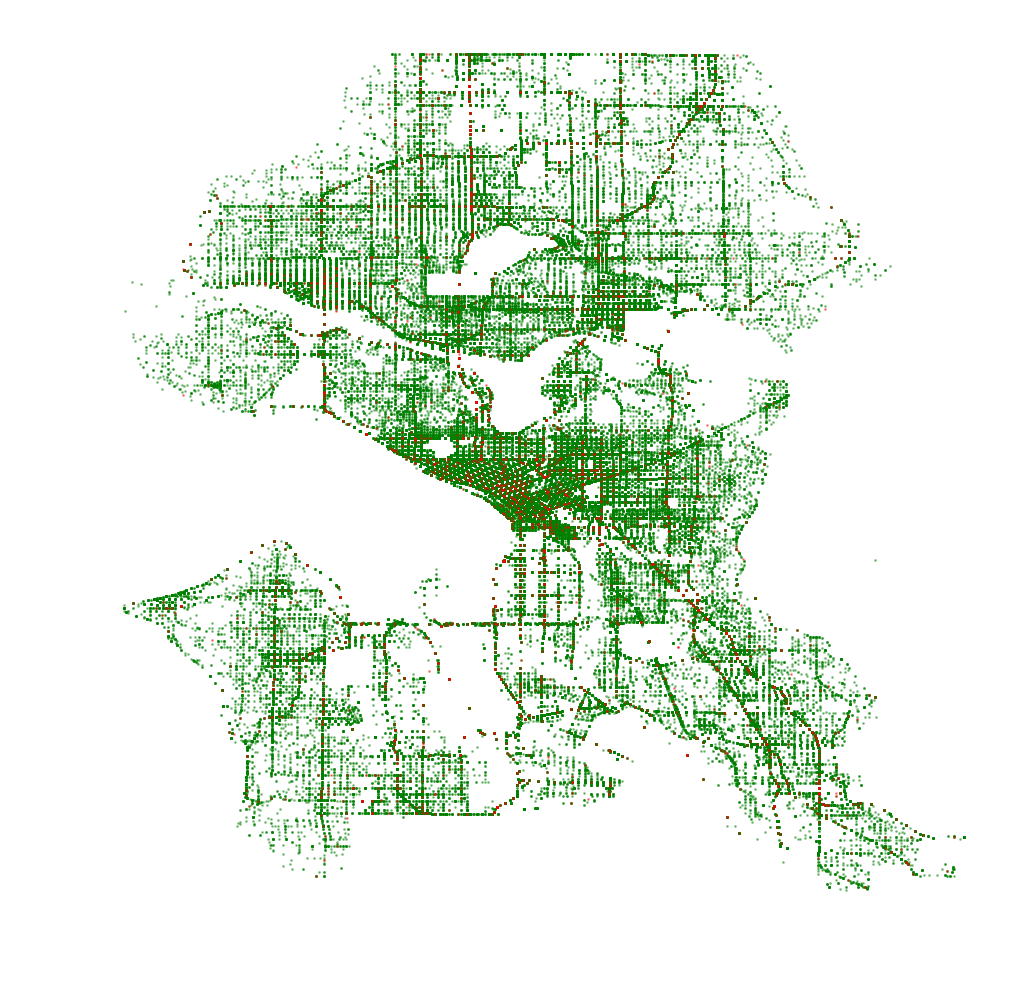


**Fig 9.** Folium map of Seattle showing the spread of neighborhoods across the city

In *Fig 9*, all the neighborhoods in Seattle are visualized on a map created using Python’s *Folium* library. This visualization was created by mapping each neighborhood to it’s specific coordinates and storing this data in a new dataframe. The neighborhood names are then used as labels along with the coordinates in the *folium.Marker* function. The red circles on the map are just a subset of the collision incidents which were plotted using the *folium.CircleMarker* function.

It is instantly noticeable that the neighborhood right at the center of the map contains majority of the collision instances with the other neighborhoods lying at the fringes of the city.

*Fig 10* provides a clearer visualization of all the collisions as a scatter plot of the coordinates. Due to the fairly large number of instances, the plot essentially maps out the entire Seattle area. This can be confirmed by looking at the folium map in *Fig 9*. On the map, the green points denote non-severe collisions while the red points denote severe cases. As expected, the green instances are much more prevalent. Despite the sparse nature of the severe points, they can still be seen across the map.

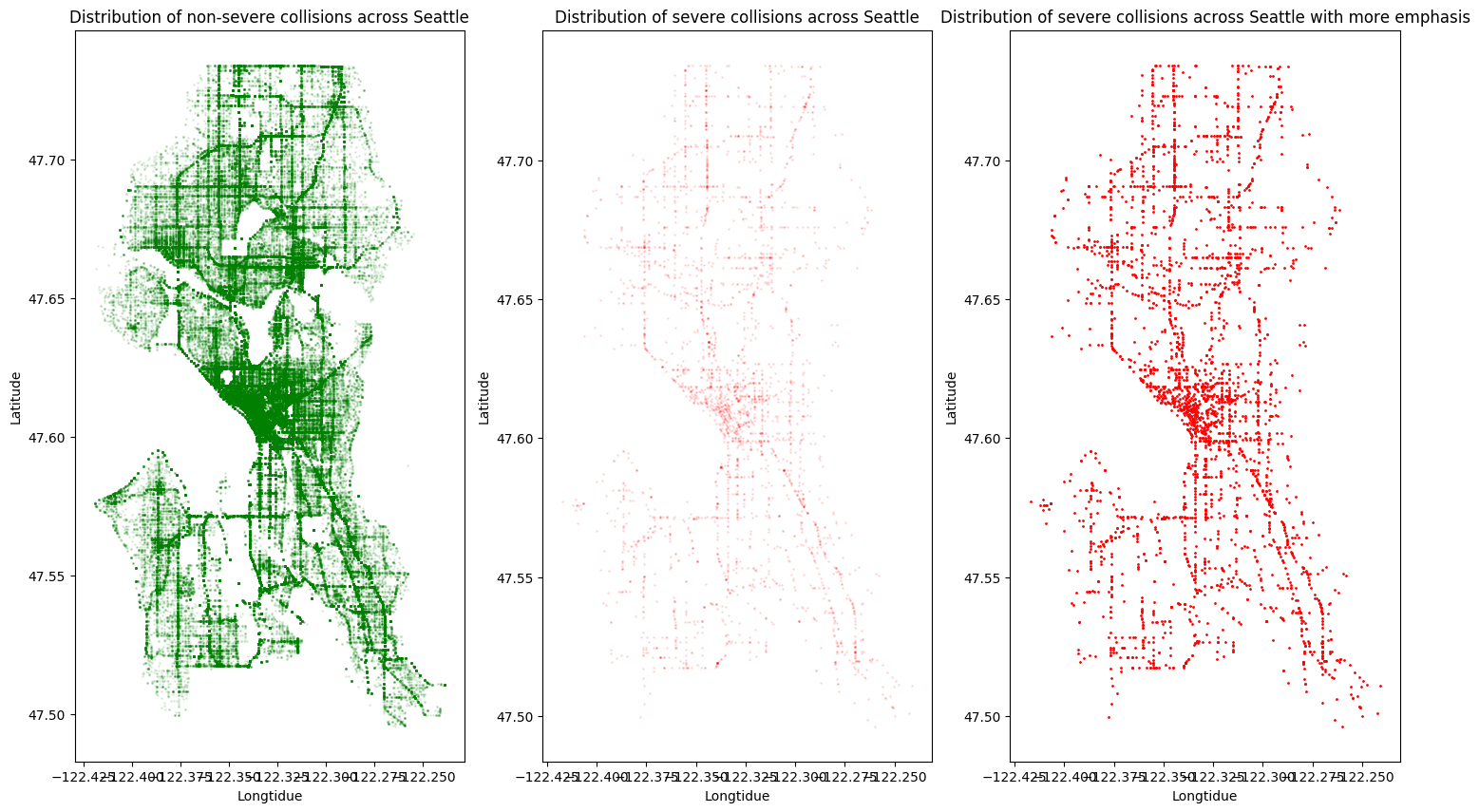


**Fig 10.** Scatterplot representation of the collision incidents across Seattle using coordinates

The severe and non-severe incident locations are plotted separately in *Fig 11* to get a better understanding of the location trends. The first plot, shown in green, displays the non-severe cases. The second plot shows the severe cases but they appear sparse due to the low frequency of the severe class. The third plot is the same as the second one but with the transparency increased so that the points can be visualized better.

In the first plot, the center of Seattle sees the highest density of collisions with several streets towards the north also depicting dense regions. The density is relatively lower on the southern side. For the severe plot, once again the center of the city seems to have the highest density of crashes while *Aurora Ave North* (road going north) and *Rainier Ave South* (road going south-east) see the next most dense areas. Other roads that have relatively denser severe collisions are *15th Ave Northwest* (road going north-west), *Lake City Way Northeast* (road going north-east) and *24th Ave East* (towards the east).

Most of these areas are also dense for the non-severe cases but the non-severe plot also has other regions of higher density as opposed to the severe plot which mostly emphasizes just these areas.

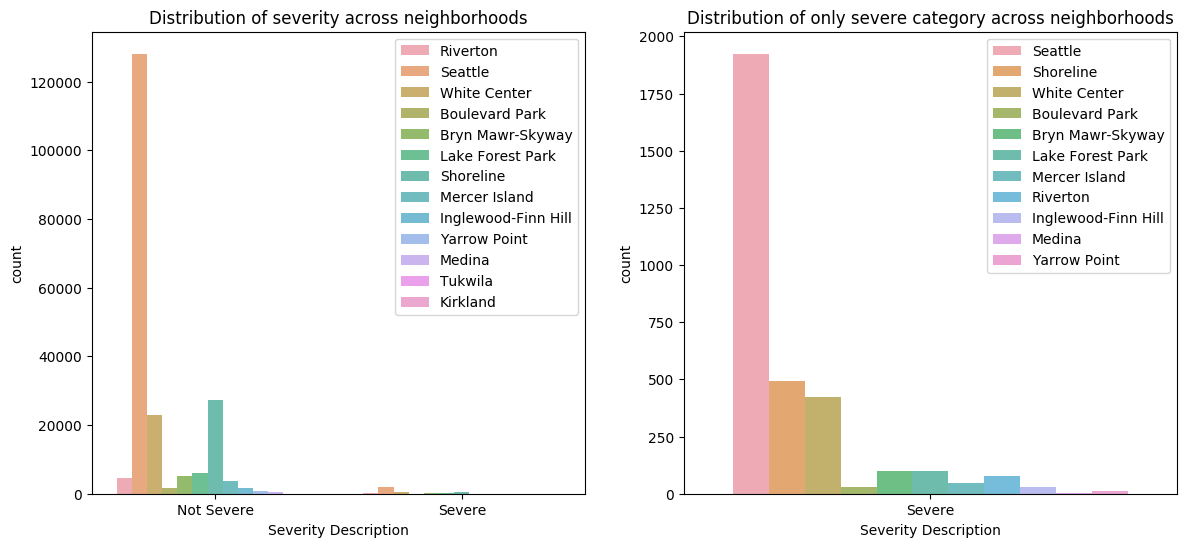


**Fig 11.** Separate scatterplots of only severe and non-severe incident locations

*Fig 12* shows the distribution of severe and non-severe cases across all the neighborhoods. A glaring issue with the first plot in the figure is that the distribution of the severe category can barely be noticed. We therefore visualize this category separately in the second plot. For both severity cases, **Seattle** is by far the most prevalent neighborhood. This makes sense as most of the other neighborhoods are at the edges of the city as we observed from the folium map as well as the scatter plot maps earlier.

The next two most frequent neighborhoods are **Shoreline** and **White Center** for both severity categories. After that, neighborhoods such as **Lake Forest Park***,* **Bryn Mawr-Skyway**and**Riverton** follow. The important point to note here is that the distribution of neighborhoods for both severity cases are almost identical. Hence, there is not much information to differentiate between non-severe and severe instances.

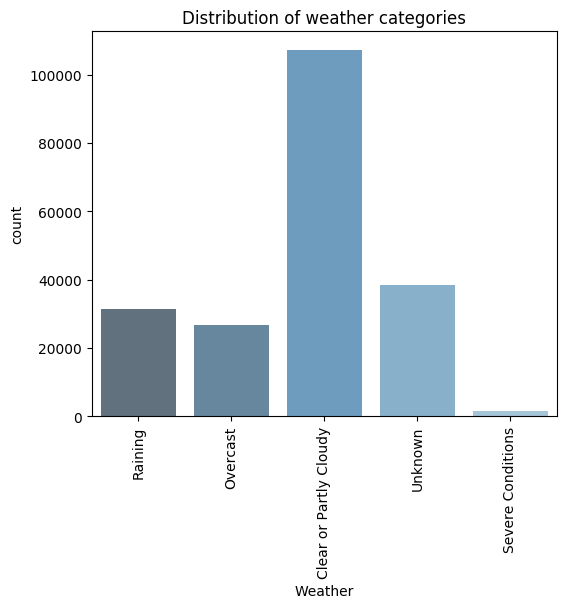
This plot was once again created using seaborn’s *countplot* function.



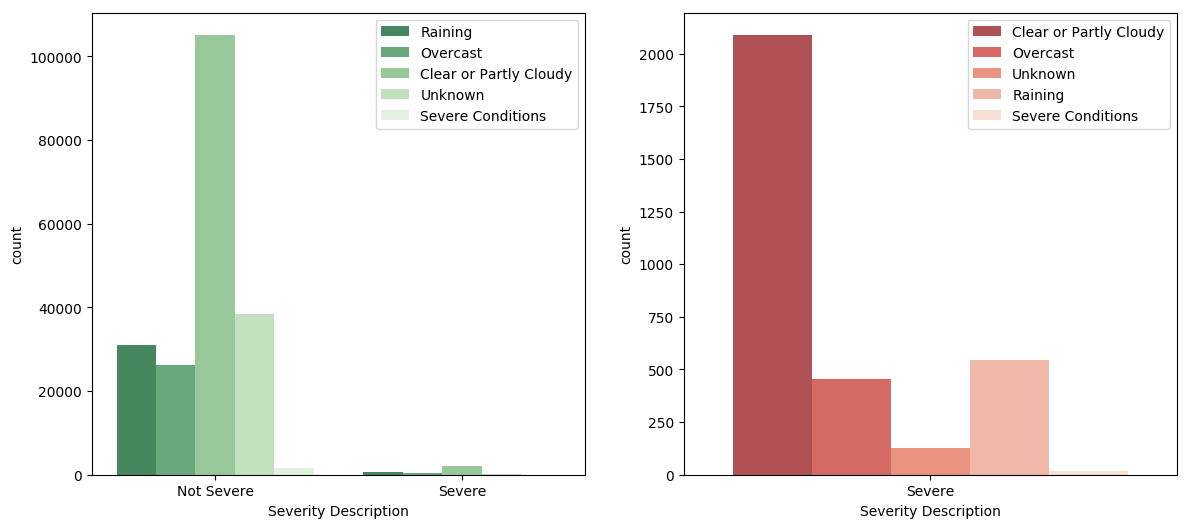
**Fig 12.** Distribution of non-severe and severe collisions across neighborhoods

## Analyzing weather related variables

Once again we look at the weather variables (*fig 13*), this time with clearer categories and binary severity classes (*fig 14*). For both cases, **Clear or Partly Cloudy** conditions are the most prevalent possibly because there is more traffic on the roads under those conditions which in turn causes more collisions. In order to normalize this issue, we would need to determine the rate of traffic flow under each condition and then divide the amount for each category by the corresponding traffic rate. Unfortunately, we don't have access to such data at the moment. When it comes to the other weather categories, the distribution is once again similar for both severity cases.

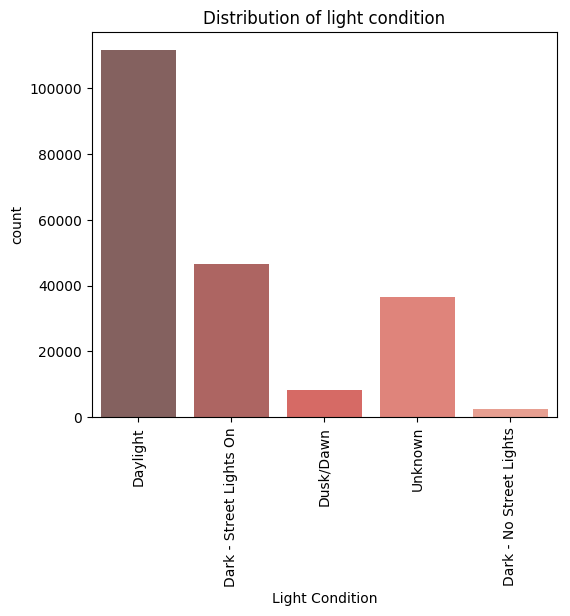


**Fig 13.** Distribution of weather categories.

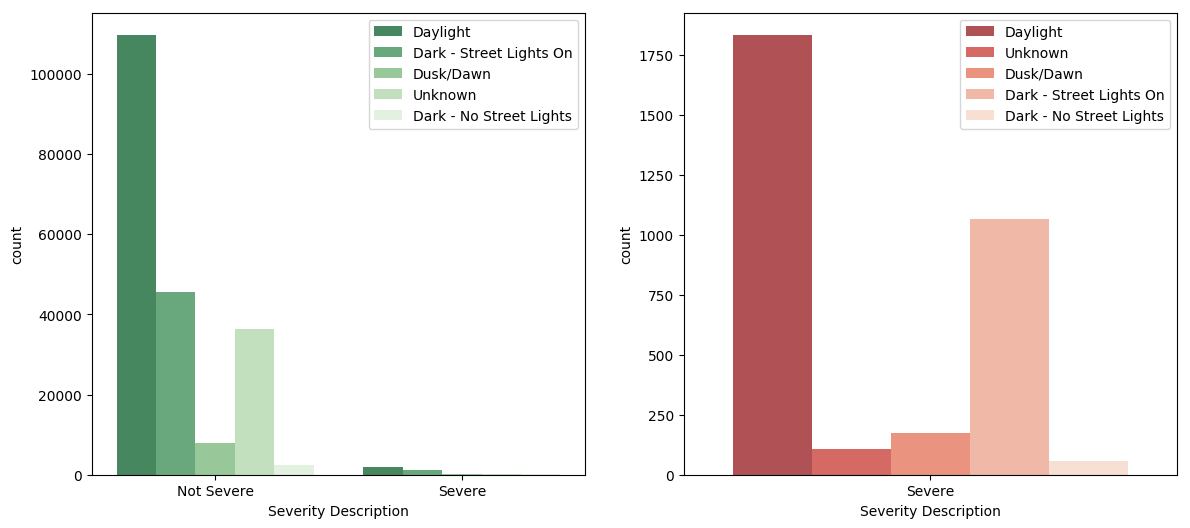


**Fig 14.** Distribution of non-severe and severe collisions across weather.

The case is the same for Light Condition as most of the collisions are during the daytime (*Fig 15*) and there is nothing to indicate a difference between severe and non-severe cases. The spread of light conditions across severity categories are similar, as shown in *Fig 16*.

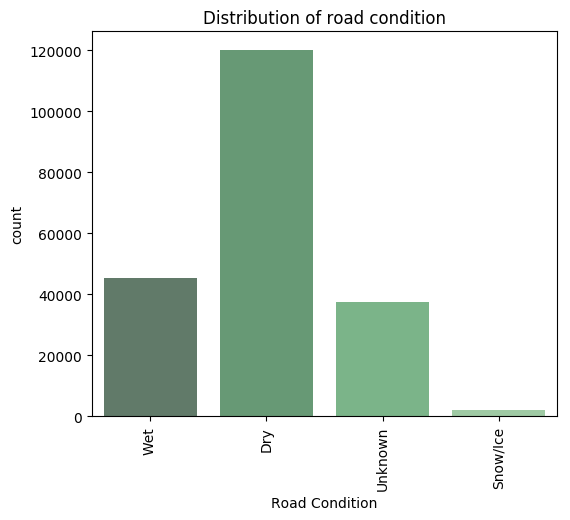


**Fig 15.** Distribution of light condition categories.

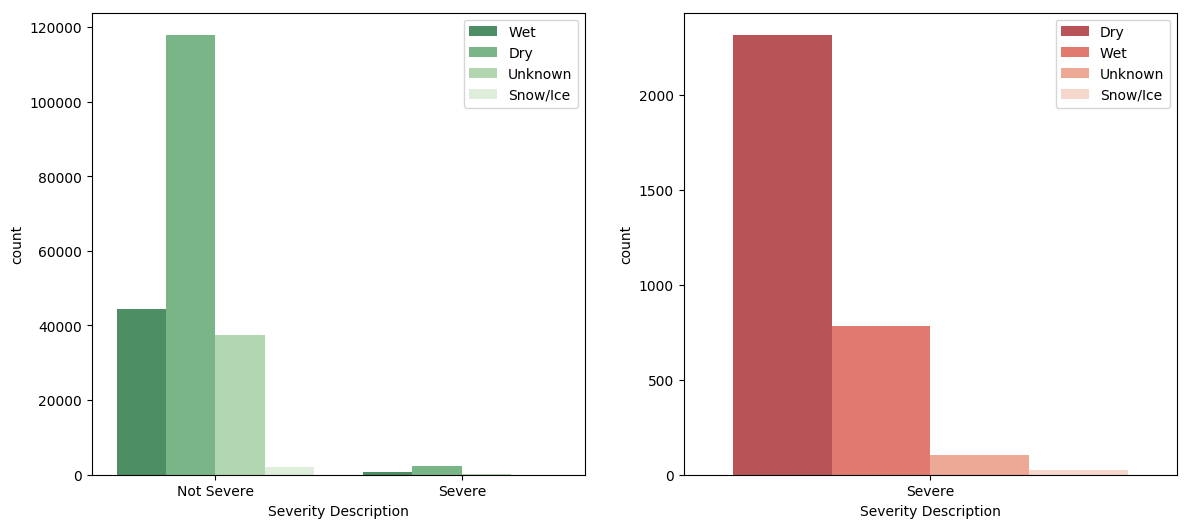


**Fig 16.** Distribution of non-severe and severe collisions across light condition.

Road conditions follow the same trend as weather and light conditions. The analysis of road condition class is shown in *Fig 17* while the trend across severity is shown in *Fig 18*. The ‘Dry’ category is the most predominant which could again be due to the fact that most cars operate during these conditions whereby causing more accidents. For both severity classes, wet conditions are less frequent while icy conditions are rare.



**Fig 17.** Distribution of road condition categories.



**Fig 18.** Distribution of non-severe and severe collisions across road condition

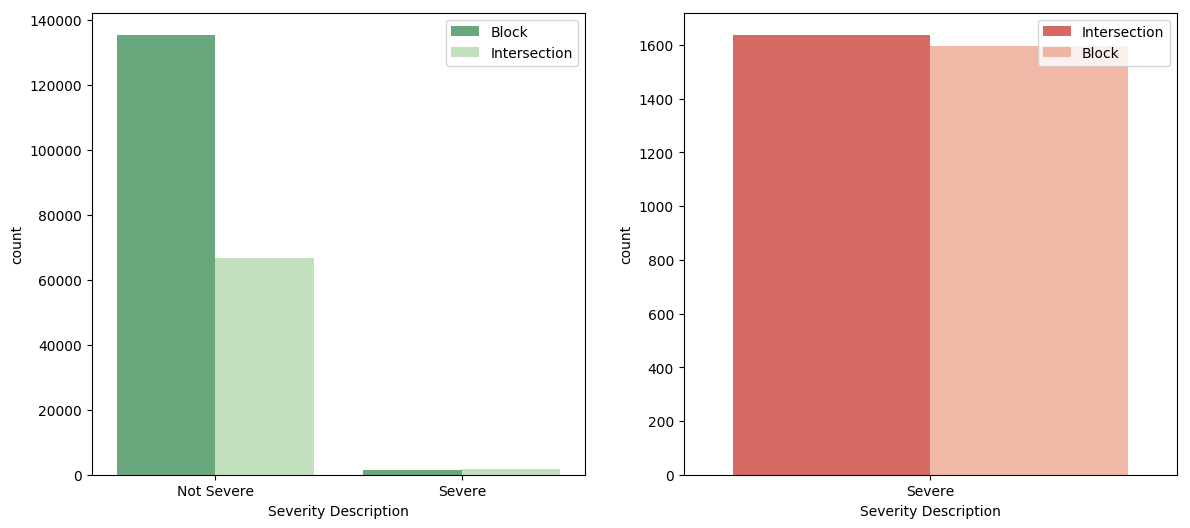
## Analyzing address related variables

The analysis of *Address Type* shows that the ‘Intersection’ category contains half the number of instances as that for the ‘Block’ category, shown in *Fig 19*.



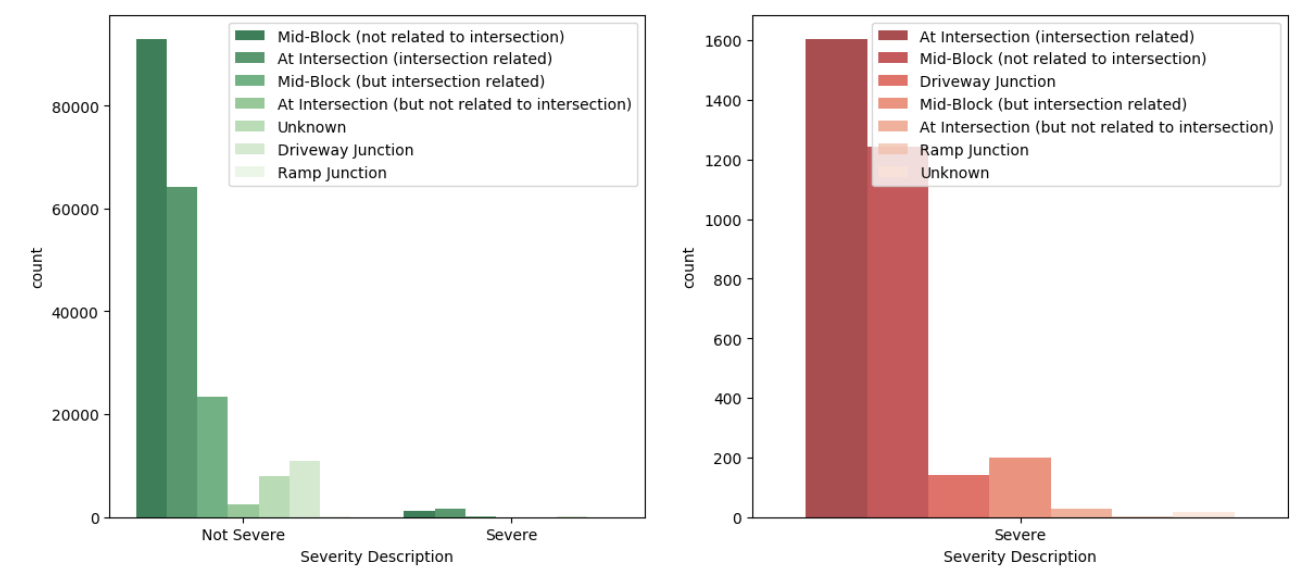
**Fig 19.** Distribution of address type classes.

When bivariate analysis is performed with severity, it appears that ‘Intersection’ has a slightly higher frequency than ‘Block’ for the severe class whereas the non-severe class shows double the number of instances for ‘Block’ as that for intersection which also matches the overall trend for *Address Type*, as shown in *Fig 20*. Hence, this could be a telling distinction for our model.



**Fig 20.** Distribution of severity across address type.

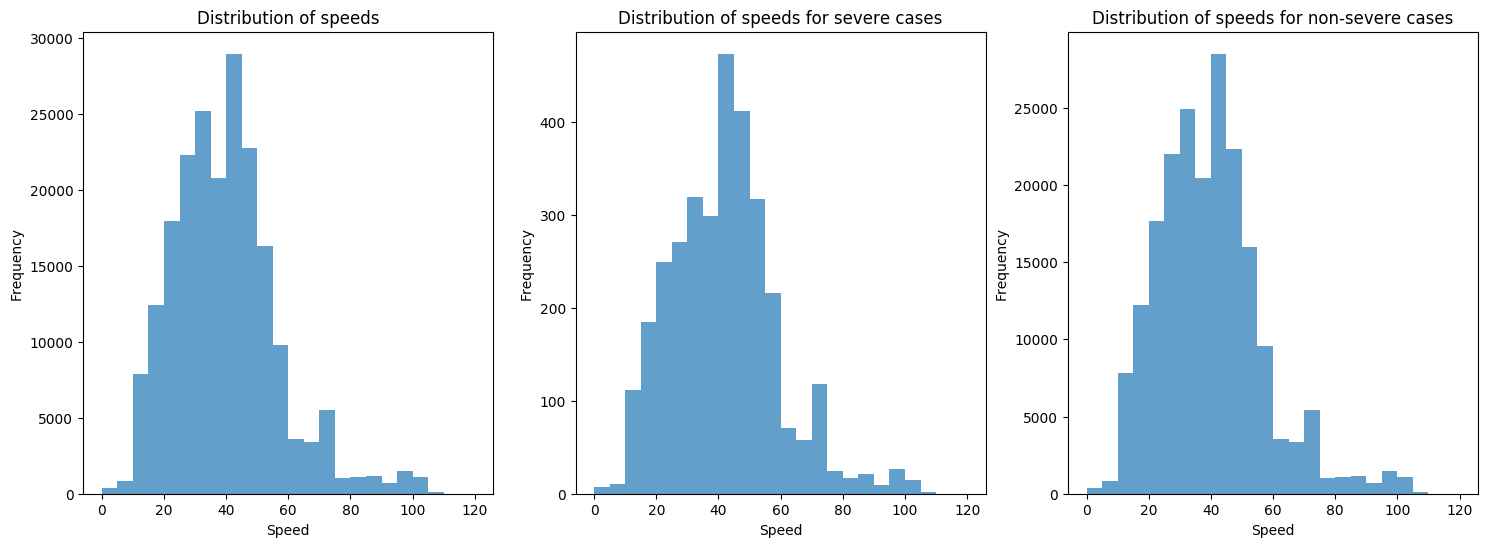
Looking at *Junction Type*, once again ‘Intersection’ collisions are more frequent for the severe class whereas ‘Mid-Block’ is more prevalent for non-severe cases, as shown in *Fig 21*. We are mostly considering the two most frequent class for each severity case as they cover much of the distribution.



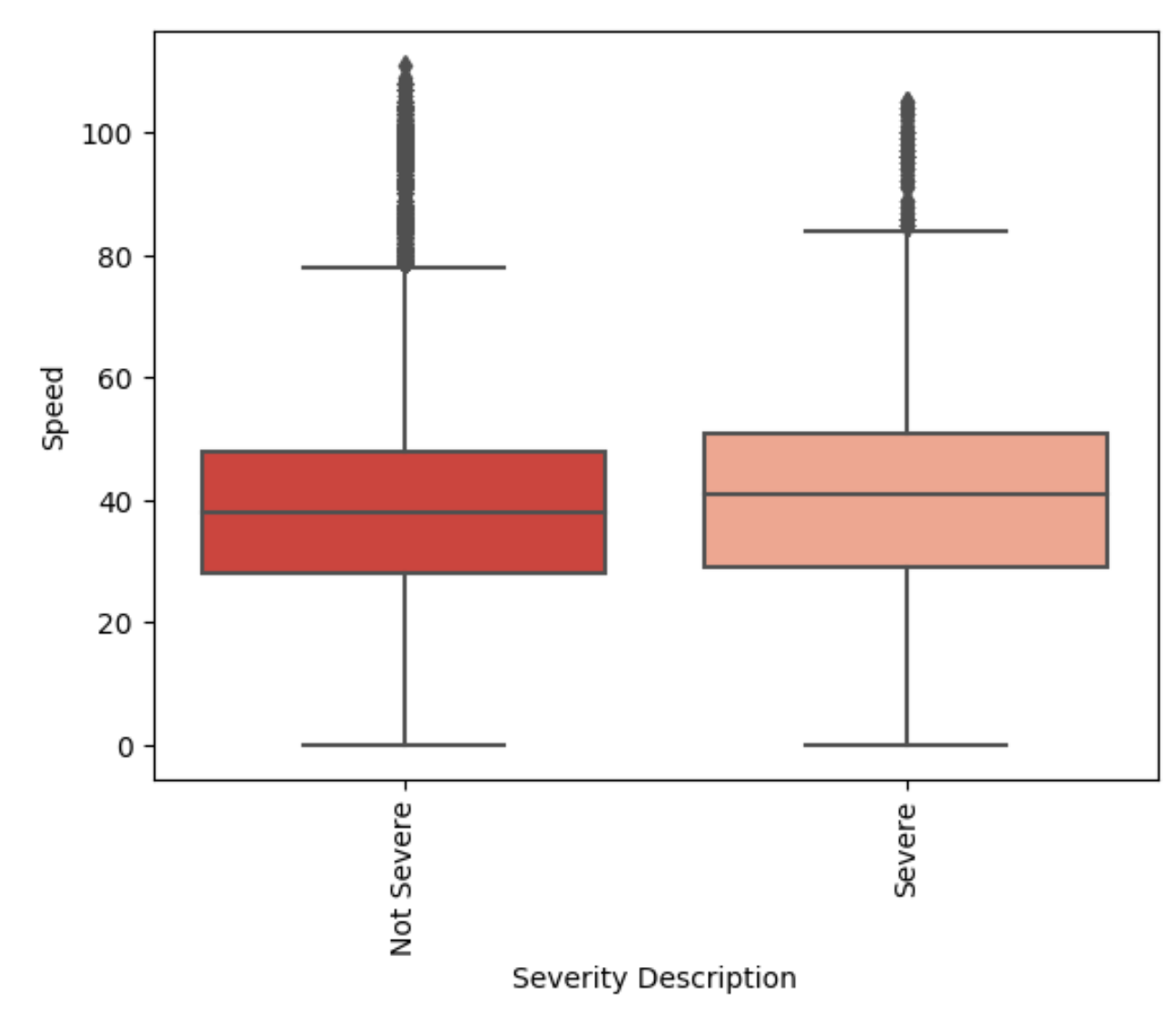
**Fig 21.** Distribution of severity across junction type.

## Analyzing variables related to the road

For the speed variable, the histograms for the two severity cases as well as the overall speed distribution are almost identical, as shown in *Fig 22*. Hence, not much information can be gathered from this data. However, the boxplot in *Fig 23* reveals that the severe cases have a slightly higher median. The severe class has a median of ~41mph while non-severe has a median of ~38mph as confirmed by the table below.

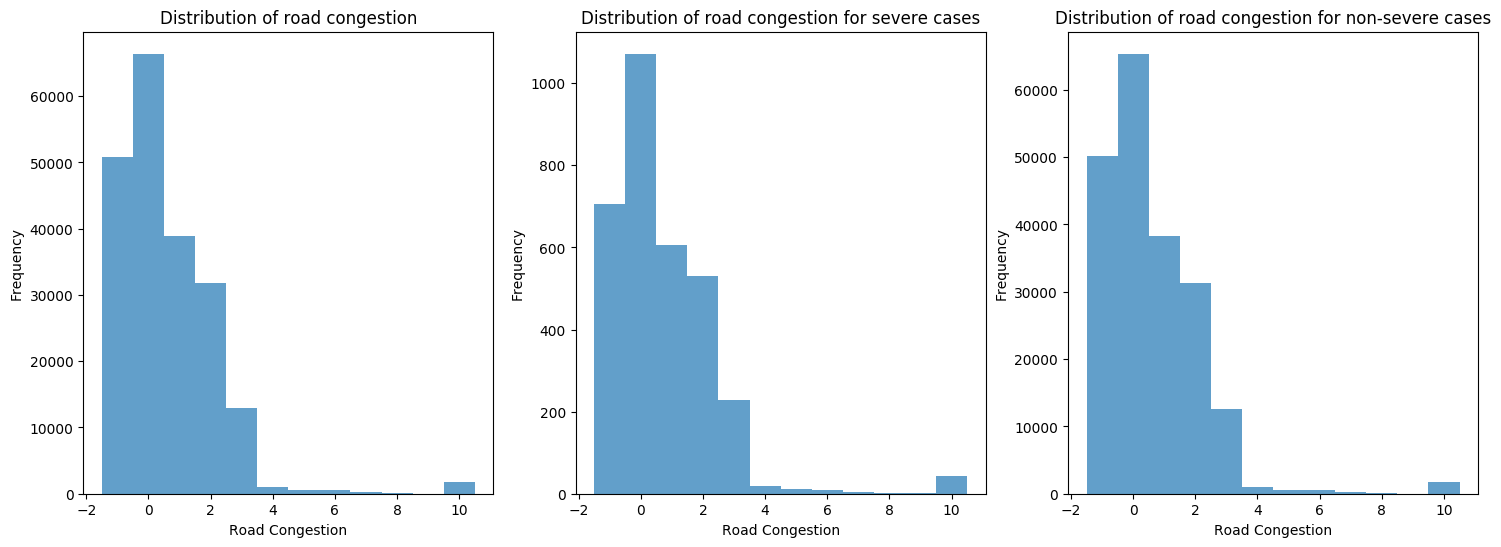


**Fig 22.** Distributions speed for different severity classes.

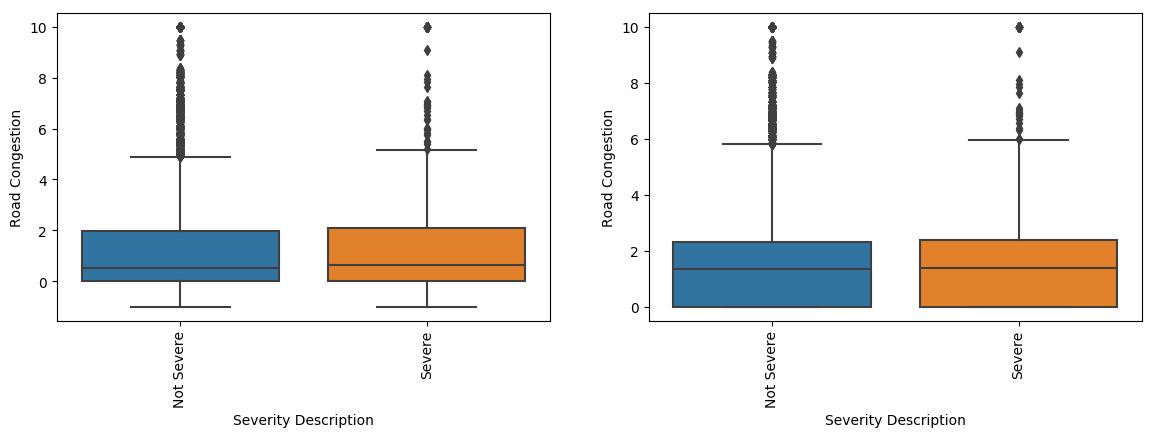


**Fig 23.** Boxplot of the speed variable across severity.

The distributions of road congestion across severity are almost identical. The median for non-severe instances is slightly lower (0.50823) than the severe (0.63665). Note that for this metric, values **closer to 1 have higher congestion** rates while those **closer to 0 have lesser congestion**.

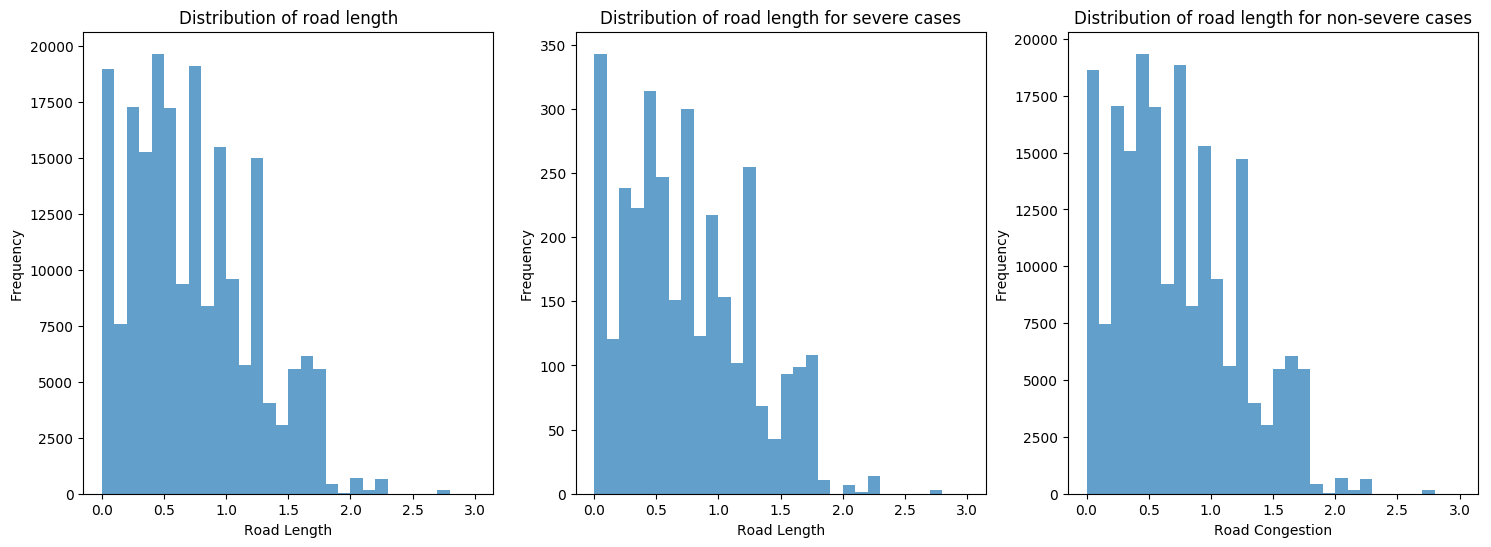


**Fig 24.** Distributions of road congestion for different severity classes.

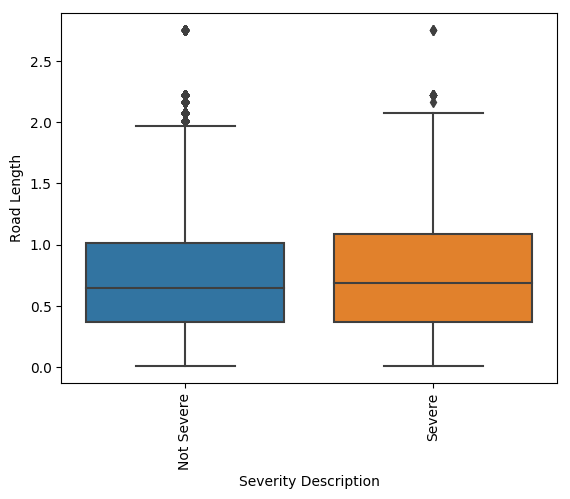


**Fig 25.** Boxplots of road congestion with the ‘-1’ value included (left) and excluded (right)

Similar to road congestion, the distributions for road length is also tight across the severity classes (*Fig 26*) where the median only differs by ~0.04 (*Fig 27*). Overall, despite the fact that speed, road congestion and road length don't provide a clear distinction between the severity levels, they could still prove to be useful for our model.



**Fig 26.** Distribution of road length across severity classes

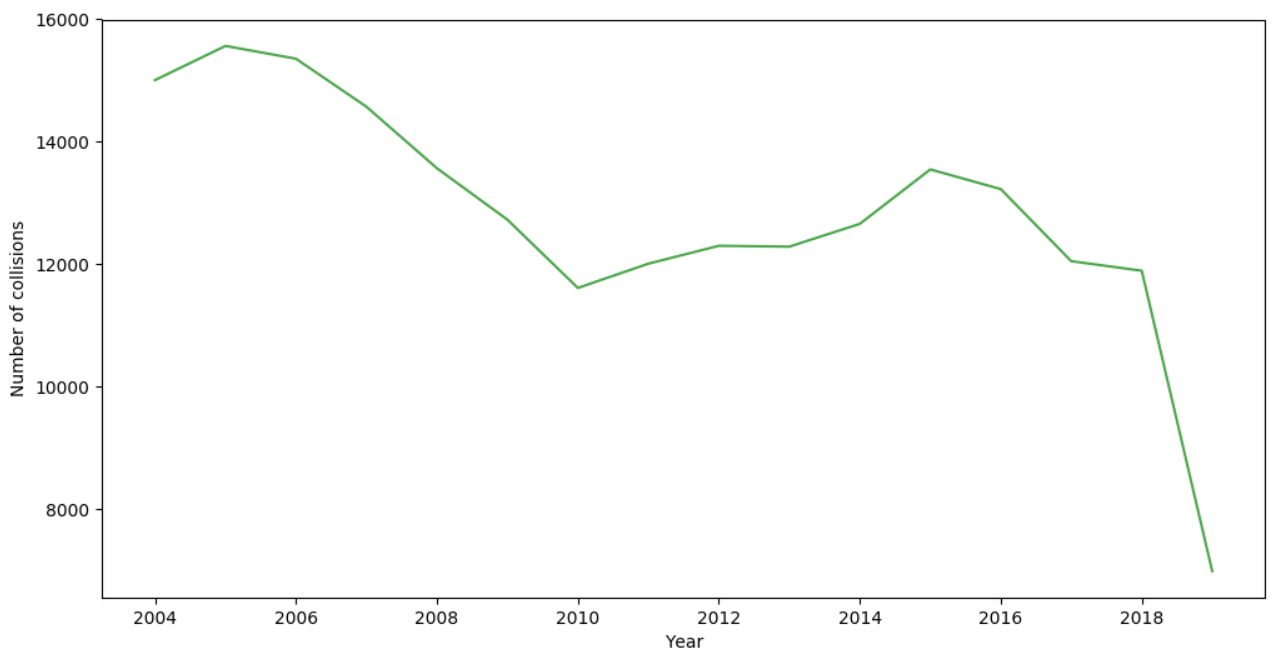


**Fig 27.** Boxplot of road length across severity.

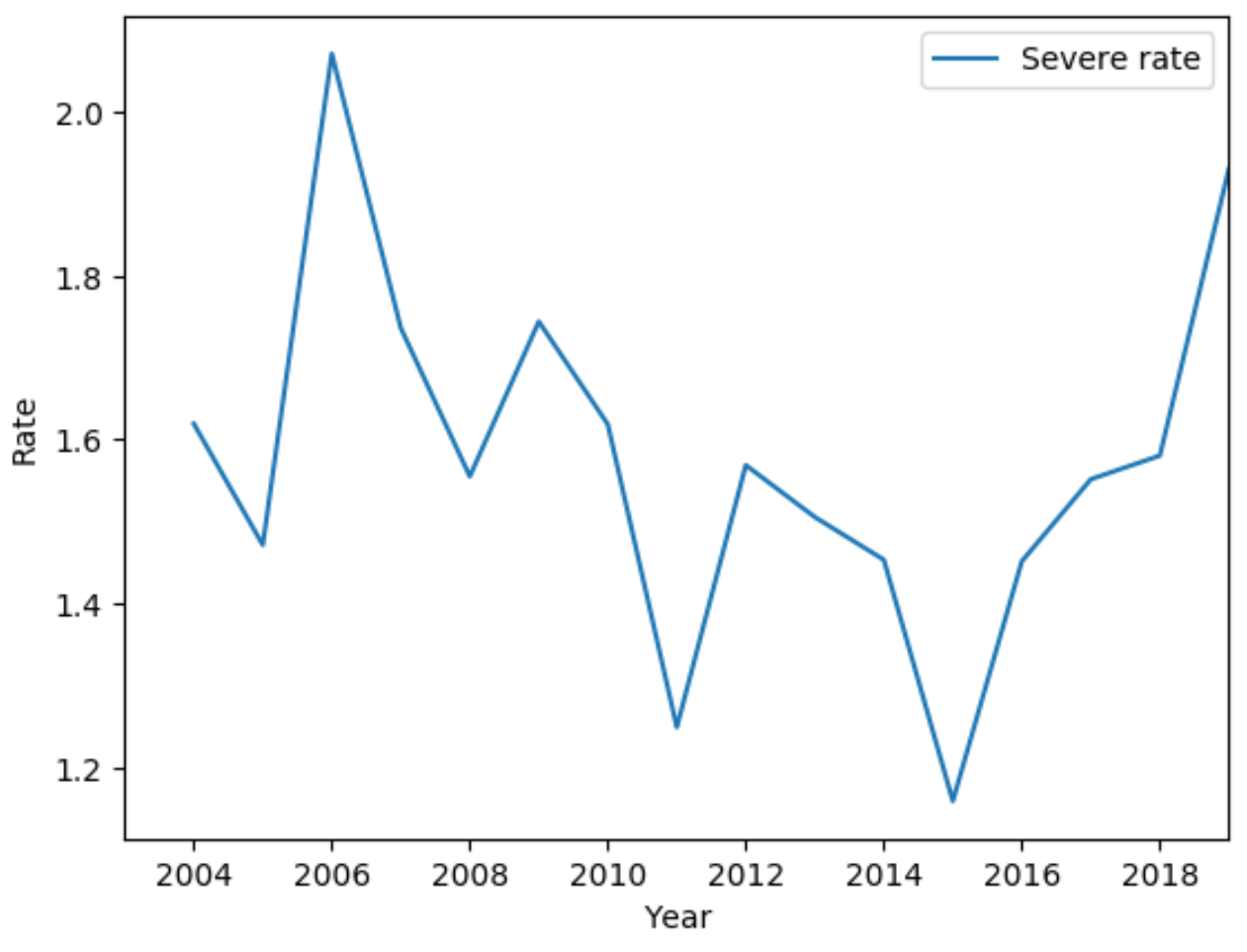
## Timeseries analysis

The timeseries plot in *Fig 28* displays the frequency of collisions across years which shows an overall downward pattern. Although, if observed in more detail, there is a downward trend from 2006 to 2011 followed by an upward pattern till 2016. From 2017 to 2019, there is again a downward movement.

Looking at the average severe collisions rate over the years in *Fig 29*, there is a sharp increase from 2005-2006 followed by a slump till 2008. After another increase till 2009, there was a downward trend till 2011. There was another one year increase till 2012 followed by yet another downward pattern till 2015 followed by a continuous increase till 2019. **Overall, there is a downward trend from 2006 to 2015 followed by an upward trend since.**

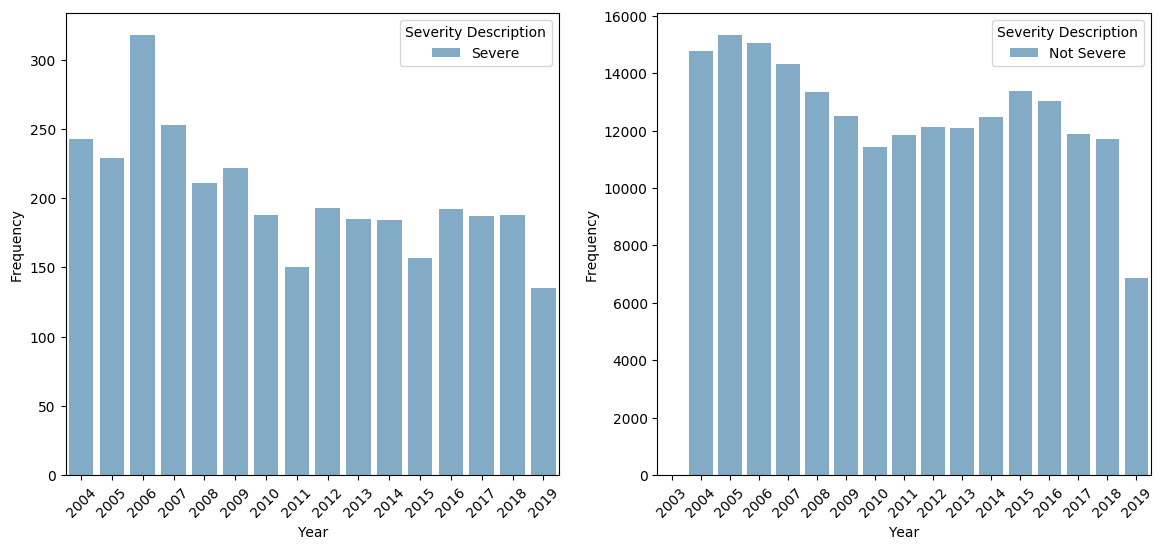


**Fig 28.** Yearly timeseries trend of collision frequencies.



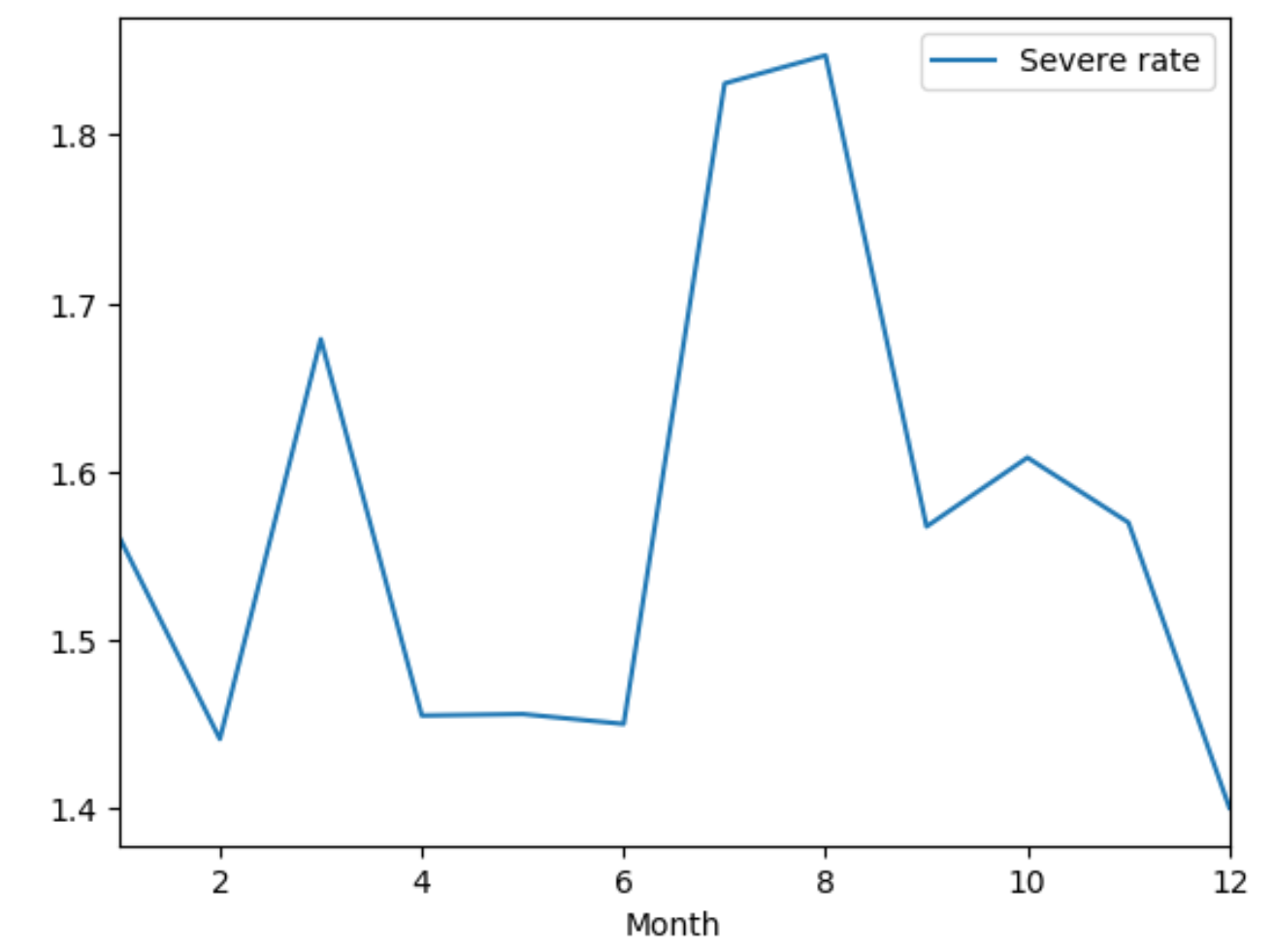
**Fig 29.** Average collision severity rates across years.

In *Fig 30*, the collision frequencies across years is plotted for both severe and non-severe cases separately. The trend for the non-severe plot was almost identical to the overall combined plot. Although, the severe plot had a general downward trend.



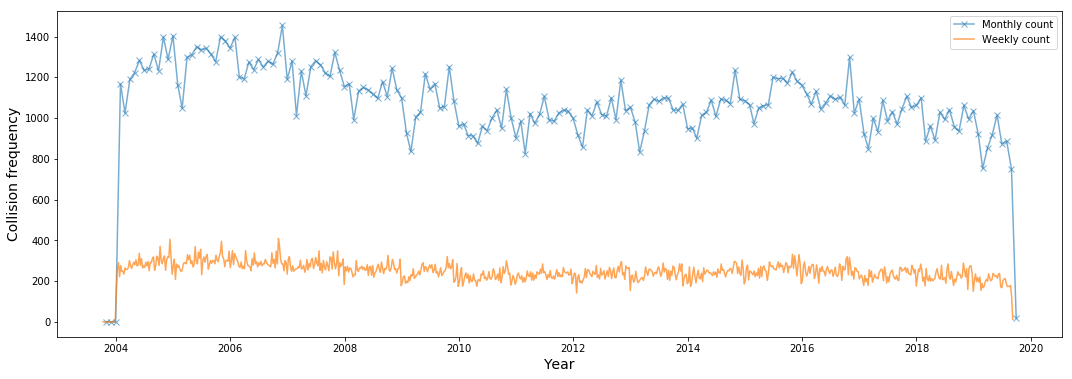
**Fig 30.** Distributions of collisions across years for both severe and non-severe cases.

*Fig 31* shows the average severity rate across months. From this plot, July and August clearly have a higher rate than the other months. April - June as well as the winter months of Feb and Dec see the lowest severe collisions.

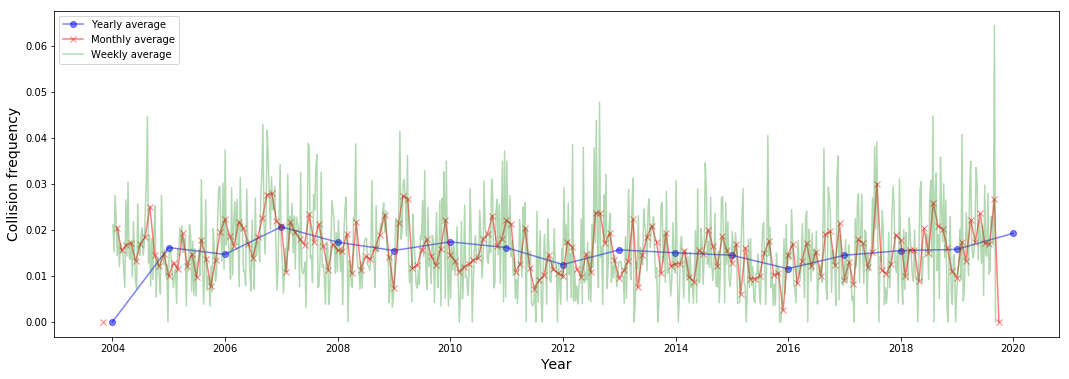


**Fig 31.** Average collision severity rates across months.

The monthly and weekly collision frequencies are plotted in *Fig 32* which shows the respective trends across time from 2004-2019.

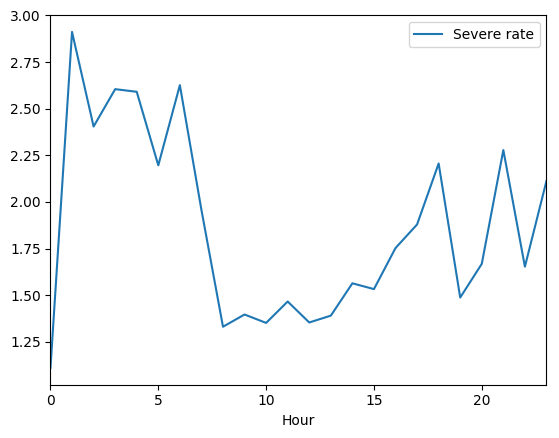


**Fig 32.** Monthly and weekly collision frequencies across time.



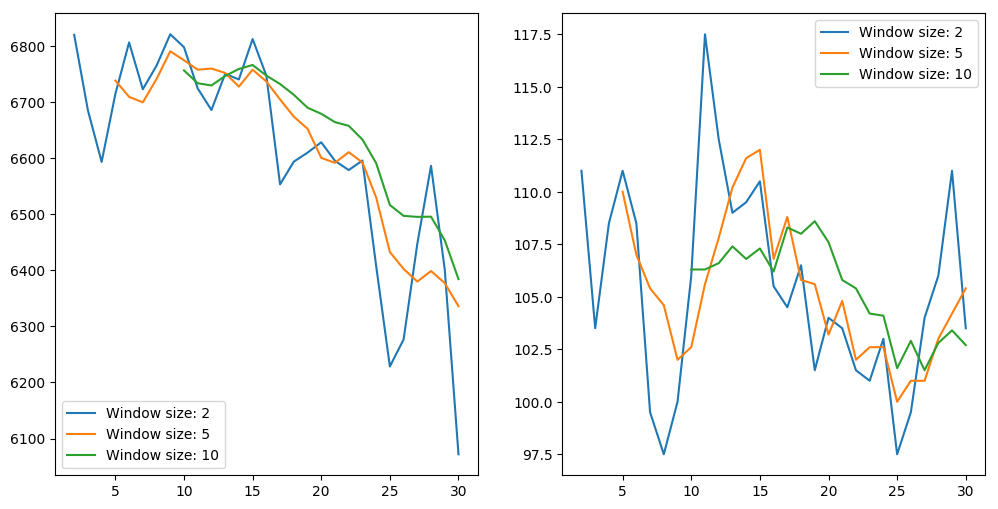
**Fig 33.** Average yearly, monthly and weekly collision rates across time.

*Fig 33* shows the average yearly, monthly and weekly collision rates across time. Note that the severe and non-severe class labels had to be converted to integers (1 for severe and 0 for non-severe) using the *LabelEncoder* function of the *sklearn* library in Python. Hence, points that are higher in value represent more severe collision instances than others. All the points have values much closer to 0 due to class imbalance. Due to the much larger proportion of non-severe instances in the dataset, an average of points at a certain point in time is likely to be skewed towards 0.



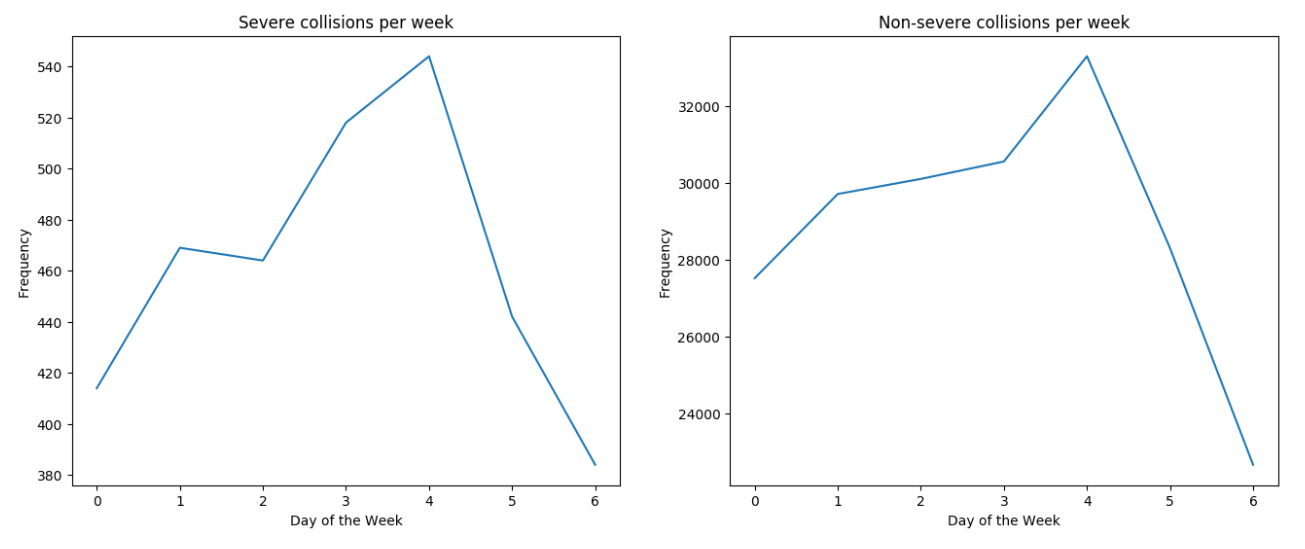
**Fig 34.** Average severity rate across hours of the day

The hourly average severity collision rates are plotted in *Fig 34* with **‘0’ indicating 12am and ‘23’ indicating 11pm**. Some trends clearly stand out. The hours between 1am and 6am see a higher rate of severe collisions. The case is similar for 4pm-6pm, 9pm and 11pm. Hence, between 4pm and 11pm, there is certainly an increasing trend in severe collision rate. Surprisingly the rate is the lowest at 12am. The hours between 8am and 3pm see the lowest severe collision rate which is somewhat expected since the light conditions tend to be the most ideal during this period.



**Fig 35.** Average non-severe (left) and severe (right) collision rates across days of the month.

From the plots in *Fig 35*, it can be seen that the downward trend for the non-severe case is much more pronounced with a higher moving average. The curve is smoother and has a distinct downward trend. For severe collisions, we can observe a general downward trend but a dip and then a rise between day 5 and 10 in the month. One way in which we can differentiate between the severity levels is by emphasizing the fact that the early days of the month see lesser severe collisions.



**Fig 36.** Average severe (left) and non-severe (right) collision rates across days of the week.

The trends for both severe and non-severe cases for day of the week are very similar. In both cases, Friday (4) sees the highest amount of collisions while Sunday (6) sees the lowest. These numbers make sense since as we progress through the week, the number of collisions increase gradually hitting the peak on Friday.

Below are the next steps.

# Statistical data analysis

# Modeling

# Results

# Conclusion

# Appendix: Links for data, jupyter notebooks