RUTGERS
UNIVERSITY | NEW BRUNSWICK

Utsav Patel (RUID - 211009880, NetID - upp10)
Manan Vakta (RUID - 206003851, NetID - mv651),
Sneh Desai (RUID - 211003857, NetID - sd1324),
Harshal Sinha (RUID - 203008538, NetID - hs1030)

Master of Science in Computer Science

Vehicular Accidents Effects on Traffic

December 2021

The school of Graduate Studies

Contents

List of Figures

Chapter 1

Goals and Motivations

1.1 Motivations

There are around 6 million vehicular accidents every year in the U.S.A. 90 people everyday die because of accidents and around 3 million people are injured every year by the same. Reducing this number would help out a significant portion of the population. To achieve this, we want to find the main contributing factors in the 'severity' of an accident. By identifying and controlling these factors, we might be able to reduce the overall severity of an accident.

1.2 Goal

Identify the key features that contribute to high severity accidents by predicting the severity of accidents and then using feature importance to see which features contributed to the high severity accidents the most.

Chapter 2

Data Exploration, Cleaning and Analysis

2.1 Data Source and Details

This Data set contains data from accidents from all around the USA. We obtained this data from Kaggle.com. The data in this data set has been collected February 2016 to December 2020 using multiple APIs that provide streaming traffic incident (or event) data. These APIs broadcast traffic data captured by a variety of entities, such as the US and state departments of transportation, law enforcement agencies, traffic cameras, and traffic sensors within the road-networks. There are about 1.5 million accident records in this dataset. Each row contains 47 different features. Description of each feature is given in our project proposal if further reading is required.

2.2 Data Preprocessing

We start off by loading our csv data into python using pandas and checking its shape -

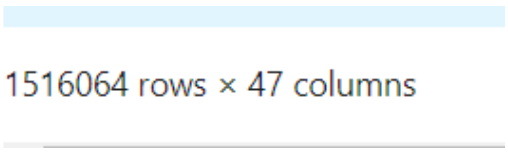

Figure 2.1: Preview of data

We see that our data has 1.5 million rows and 47 columns. We then go over all 47 features

present in our data -

```
Index(['ID', 'Severity', 'Start_Time', 'End_Time', 'Start_Lat', 'Start_Lng',
       'End_Lat', 'End_Lng', 'Distance(mi)', 'Description', 'Number', 'Street',
       'Side', 'City', 'County', 'State', 'Zipcode', 'Country', 'Timezone',
       'Airport_Code', 'Weather_Timestamp', 'Temperature(F)', 'Wind_Chill(F)',
       'Humidity(%)', 'Pressure(in)', 'Visibility(mi)', 'Wind_Direction',
       'Wind_Speed(mph)', 'Precipitation(in)', 'Weather_Condition', 'Amenity',
       'Bump', 'Crossing', 'Give_Way', 'Junction', 'No_Exit', 'Railway',
       'Roundabout', 'Station', 'Stop', 'Traffic_Calming', 'Traffic_Signal',
       'Turning_Loop', 'Sunrise_Sunset', 'Civil_Twilight', 'Nautical_Twilight',
       'Astronomical_Twilight'],
      dtype='object')
```

Figure 2.2: Features

By going through this data a little bit, we can clearly see that a few of the features present in our data are only collected *after* the accident has occurred and are thus not very useful to us while we predict the severity of an accident. Thus we drop these features. We then continue to perform pre-processing and check how many unique values each of our features contains -

```
Unique count of values in features:
Severity 4
Start_Time 1037092
Start_Lat 590611
Start_Lng 603369
Number 37593
Street 93048
Side 2
City 10658
County 1671
State 49
Zipcode 177197
Country 1
Timezone 5
Airport_Code 1986
Weather_Timestamp 331749
Temperature(F) 776
Wind_Chill(F) 884
Humidity(%) 101
Pressure(in) 1008
Visibility(mi) 75
Wind_Direction 25
Wind_Speed(mph) 122
Precipitation(in) 188
Weather_Condition 117
Amenity 2
Bump 2
Crossing 2
Give_Way 2
Junction 2
No_Exit 2
Railway 2
Roundabout 2
Station 2
Stop 2
Traffic_Calming 2
Traffic_Signal 2
Turning_Loop 1
Sunrise_Sunset 3
Civil_Twilight 3
Nautical_Twilight 3
Astronomical_Twilight 3
```

Figure 2.3: Unique Features

From the above we clearly see that Country and Turning-Loop features have only 1 distinct value, thus these features can be dropped from our data without affecting our prediction capabilities at all.

We also see that we have a few features that have only a few distinct values and thus have a high chance of being categorical features. These features are - 'Side', 'Timezone', 'Amenity', 'Bump', 'Crossing', 'Give_Way', 'Junction', 'No_Exit', 'Railway', 'Roundabout', 'Station', 'Stop', 'Traffic_Calming', 'Traffic_Signal', 'Sunrise_Sunset', 'Civil_Twilight', 'Nautical_Twilight', 'Astronomical_Twilight', 'Wind_Direction'. Let us take a closer look at the categorical values in these features and see if we can perform any data cleaning on them -

```
Feature:  Severity [3 2 4 1]

Feature:  Side ['R' 'L']

Feature:  Timezone ['US/Eastern' 'US/Pacific' nan 'US/Central' 'US/Mountain']

Feature:  Amenity [False  True]

Feature:  Bump [False  True]

Feature:  Crossing [False  True]

Feature:  Give_Way [False  True]

Feature:  Junction [False  True]

Feature:  No_Exit [False  True]

Feature:  Railway [False  True]

Feature:  Roundabout [False  True]

Feature:  Station [False  True]

Feature:  Stop [False  True]

Feature:  Traffic_Calming [False  True]

Feature:  Traffic_Signal [False  True]

Feature:  Sunrise_Sunset ['Night' 'Day' nan]

Feature:  Civil_Twilight ['Night' 'Day' nan]

Feature:  Nautical_Twilight ['Night' 'Day' nan]

Feature:  Astronomical_Twilight ['Night' 'Day' nan]

Feature:  Wind_Direction ['SW' 'Calm' 'WSW' 'WNW' 'West' 'NNW' 'South' 'W' 'NW' 'North' 'SSE' 'SSW'
 'ESE' 'SE' nan 'East' 'Variable' 'NNE' 'NE' 'ENE' 'CALM' 'S' 'VAR' 'N'
 'E']
```

Figure 2.4: Unique Features

From the above we can see that Wind_direction has multiple features that are redundant. We clean up these values into easily readable categorical values -

Unique values for Wind_Direction after cleaning up: ['SW' 'CALM' 'W' 'N' 'S' 'NW' 'E' 'SE' nan 'VAR' 'NE']

By seeing the number of unique values our data contains, we see that weather condition contains 117 different values. We know that weather condition will play a big factor in determining

accident severity and thus we must take some steps to reduce the amount of weather condition values into a set of more clear and concise values. Upon reading through other research (According to Road Weather Management Program) related to the analysis of car accidents in conjunction with weather conditions we came to the conclusion that the following weather conditions are of interest for our analysis as these conditions are more likely to cause an accident. These weather conditions of interest are: 'Clear', 'Cloud', 'Rain', 'Heavy_Rain', 'Snow', 'Heavy_Snow' and 'Fog'.

In order to focus our analysis with regards to the correlation of car accidents and the corresponding weather conditions, we created separate features for the above mentioned weather conditions.

Creating separate features(boolean) for: 'Clear', 'Cloud', 'Rain', 'Heavy_Rain', 'Snow', 'Heavy_Snow' and 'Fog'

In essence, for the rows where the value for the feature 'Weather_Condition' is equal to that particular weather condition of interest then it will be assigned the value true for that feature, else it is assigned the value false.

Looking at our time features, to be able to handle them in python we first convert them to python date_time datatype. We then check how much time difference exists between the Start_time feature and Weather_timestamp feature -

```
Mean difference between 'Start_Time' and 'Weather_Timestamp':  0 days 00:01:29.597799165
```

Figure 2.5: Unique Features

There is around a 1 minute of average difference between the two features. Because this is a very small amount of difference, we can safely drop one of the features and just use the other.

Now we want to **handle missing values** in our data set. First let us check how many missing or NaN values our features contain. Below we have shown the features with the most amount of missing data -

4	Number	69.000715
5	Street	0.000000
6	Side	0.000000
7	City	0.005475
8	County	0.000000
9	State	0.000000
10	Zipcode	0.061673
11	Timezone	0.151841
12	Airport_Code	0.280199
13	Temperature(F)	2.838469
14	Wind_Chill(F)	29.637007
15	Humidity(%)	3.001786
16	Pressure(in)	2.392643
17	Visibility(mi)	2.916170
18	Wind_Direction	2.760965
19	Wind_Speed(mph)	8.499773
20	Precipitation(in)	33.675953
21	Weather_Condition	2.902714

Figure 2.6: Unique Features

From the above we can see that the feature 'Number' is more than 60% of its values. This feature 'Number' is actually the street number where the accident took place and thus is not an import feature to us. We just drop this feature and continue with our processing.

We also notice that Precipitation and Wind_chill contain around 30% missing values. Since both of those features are important for our analysis and there is no smart way to fill these missing values, we simply drop all the rows where the data of precipitation and wind_chill are missing.

Now for the remaining features where the amount of missing data is fairly small -

	Feature	Missing_Percent(%)
14	Humidity(%)	0.201091
15	Pressure(in)	0.070967
16	Visibility(mi)	0.259088
17	Wind_Direction	0.001160
20	Weather_Condition	0.218595

Figure 2.7: Unique Features

We first group these missing data points by airport code (location) and Month (time) and fill in these grouped values using the median of the values in these groups. After this step we simply drop all remaining values that still contain missing values.

After performing all this preprocessing we have 1 more thing that we must check. Since we want to predict 'Severity', let us check the count of each category of severity in our data -

Severity 1 = 26340 Severity 2 = 1173892 Severity 3 = 270141 Severity 4 = 45691

From the above we can see that our data is highly skewed toward severity 2 accidents and using the data as is would cause us problems when predicting the severity of an accident. We also notice that the number of Severity 1 accidents are very low and are also of lower concern to us since the actual severity of level 1 accidents is very low. We thus decide to drop all Severity 1 accidents and then undersample our data. We take 45691 values from each of our severities to get an equal amount of data from all 3 remaining severities so that we don't have any skewness in our data.

2.3 Data Analysis

Now that we have pre-processed and cleaned our data, let us create some graphs to help us visualize the data.

2.3.1 Time Feature Analysis

In this section we have created graphs to show how time affected the severity of accidents.

First let us take a look at accidents per year -

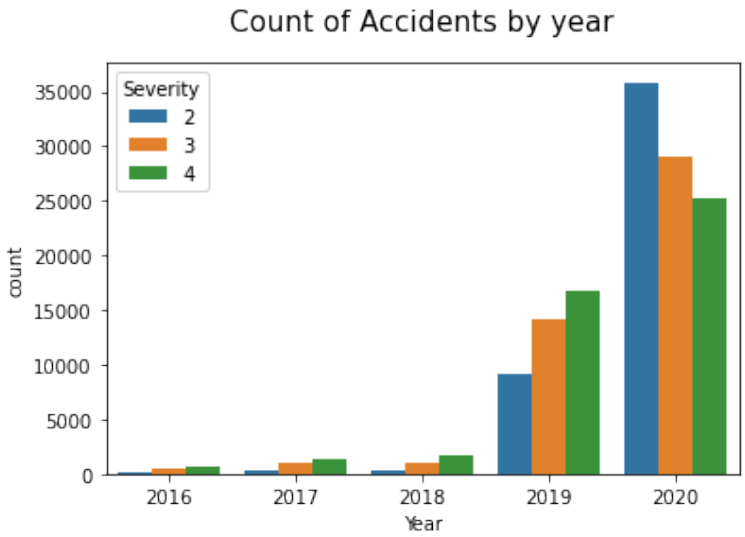

Figure 2.8: Accidents Per Year

From the above graph it may seem like the number of accidents per year are growing exponentially, but the nature of this graph is simply because of the fact that data collection became easier and more widespread in the year 2019 and 2020 and thus we have more data for those particular years and the other years.

Now let us take a look at the distribution of accidents by the month in which they occurred -

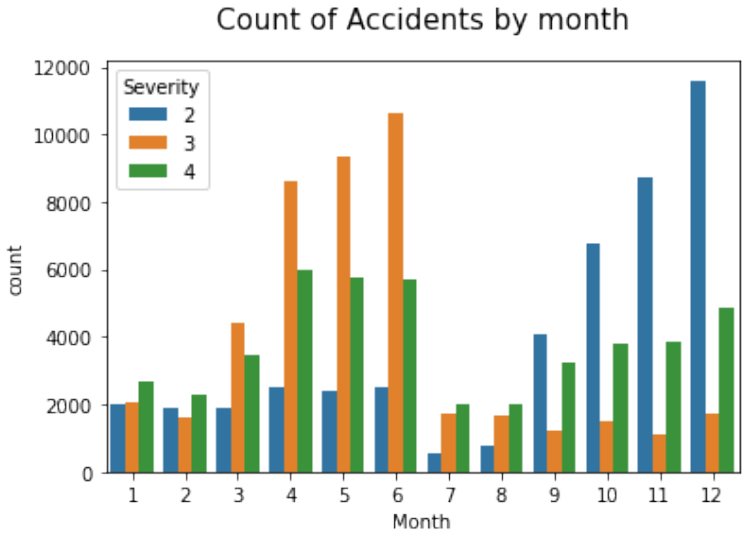

Figure 2.9: Accidents Per Month

We see something fairly interesting in these graphs, The months January, February, July and August (1,2,7 and 8) have on average much fewer accidents than the other months. This can probably be attributed to those months having a lot of holidays and thus lesser number of vehicles on the road. The Months September, October, November and December have more

severity 2 accidents than severity 3 or 4 and months April, May and June have more severity 3 accidents than 2 or 4. This is probably because in the Winter parts of the year people are more careful while driving and also usually drive slower thus even if accidents do occur they are less sever than accidents that occur during the summer months when people tend to be less careful and drive more recklessly.

Let us now take a look at the accident distribution by day of the week -

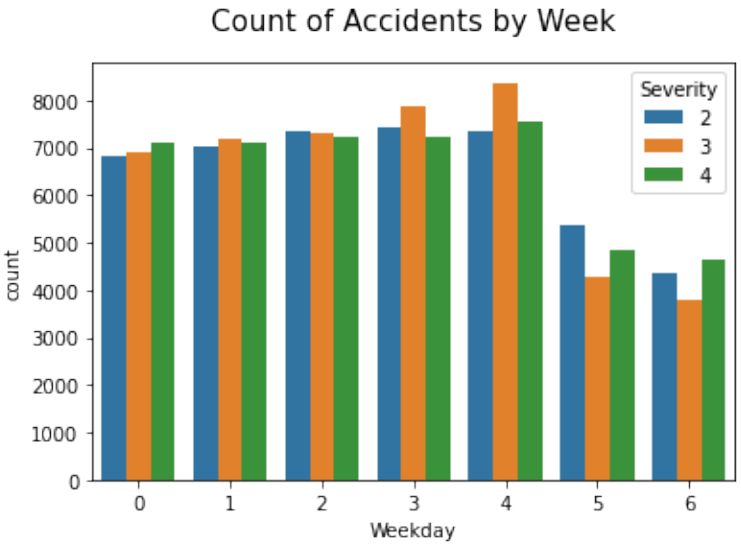

Figure 2.10: Accidents Per Day

As we can see from the above, weekdays tend to have more accidents on average than weekends since there are more cars on the road during the weekday because of work and this leads to higher probability of accidents occurring.

Let us take a look at accident distribution by hour of the day -

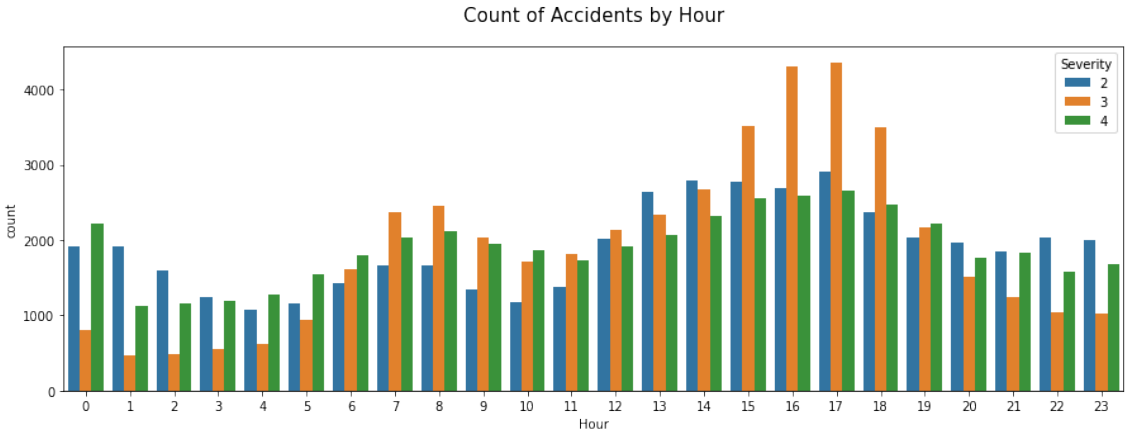

Figure 2.11: Accidents Per Hour

As we can see from the above graph, more accidents tend to happen from 3 PM to 6 PM than any other part of the day. This is largely owing to the fact that most jobs finish around that time and this causes the number of vehicles on the road during that time frame to be high.

Now let us take a look at accident distribution by minute per hour -

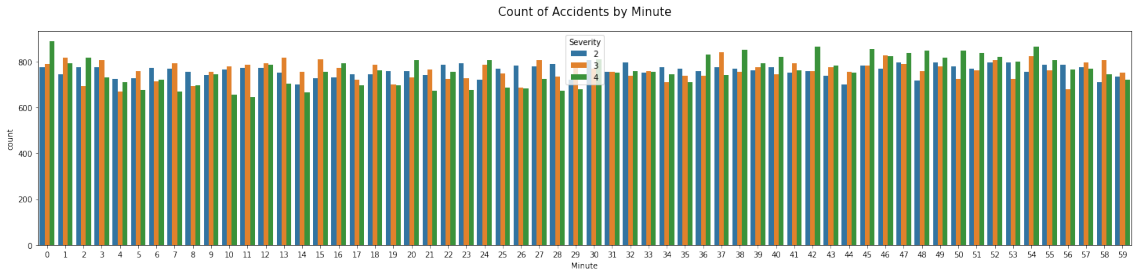

Figure 2.12: Accidents Per Hour

As expected, we have accidents almost equally distributed throughout the hour.

Let us take a look at the number of accidents by day and night where we define Day as 8 AM to 8 PM and Night as 8 PM to 8 AM. Here we noticed that 5 features in our data depicted this Day Night cycle, these were - Sunrise, Sunset, Nautical_Twilight, Astronomical_Twilight, and Civil_Twilight. When we plotted graphs for all these features, they all turned out to be the same -

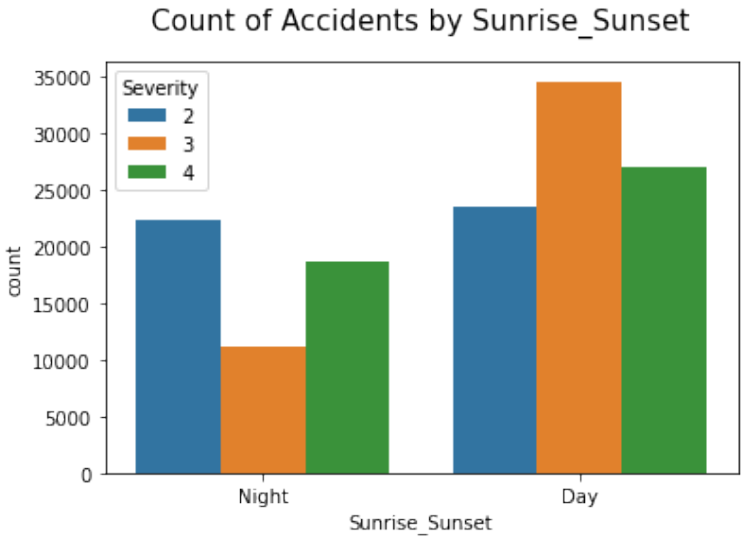

Figure 2.13: Accidents By Day and Night

As we can see, more accidents happen during the day on average than in the night simply because there are a lot more vehicles on the road during that day than at night.

2.3.2 Location Analysis

In this section we want to see how location played a role in accidents in the USA. We first wanted to plot all accidents by Latitude and Longitude to see the locations of accidents on a sort of map -

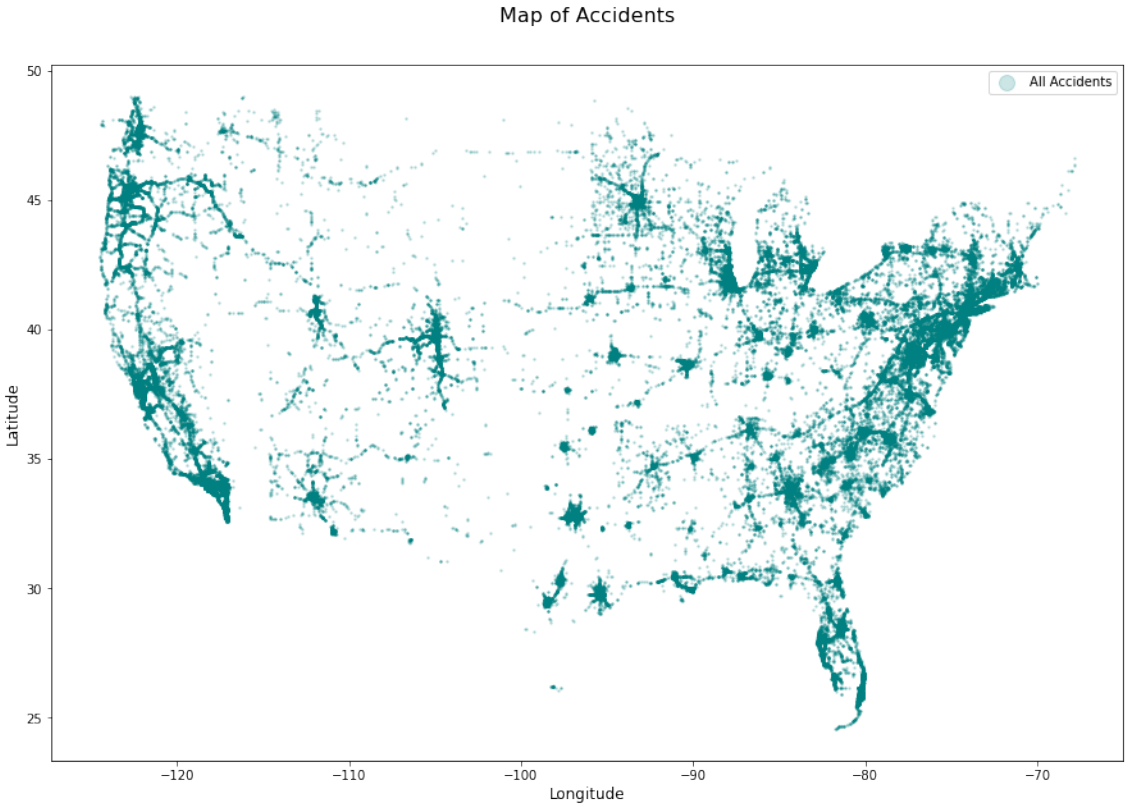

Figure 2.14: Accidents By Longitude and Latitude

As we can see from above most of the recorded accidents are focused around the east and west coast. This is expected because those are the most densely populated areas of the US and thus will contain the most number of accidents.

Now let us take a look at accidents per state in the US separated by severity where we order them in descending order of the number of accidents recorded for that severity.

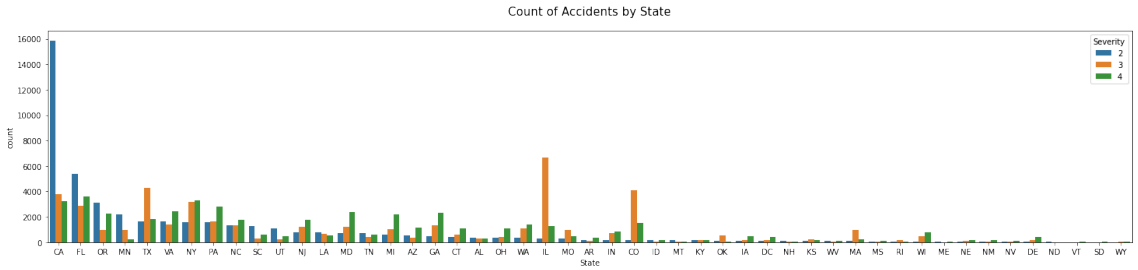

Figure 2.15: Accidents By State in order of Severity 2

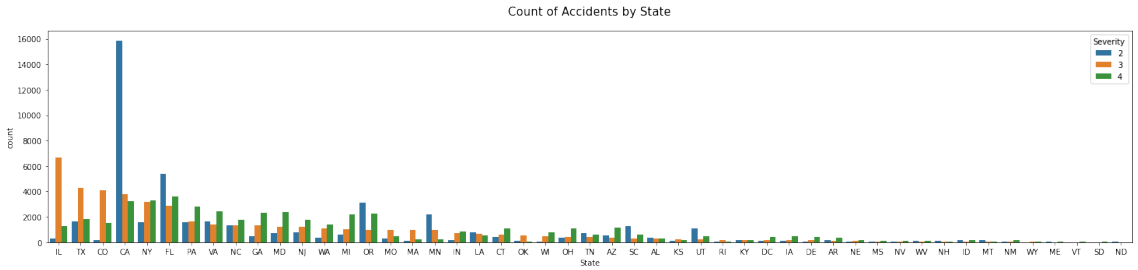

Figure 2.16: Accidents By State in order of Severity 3

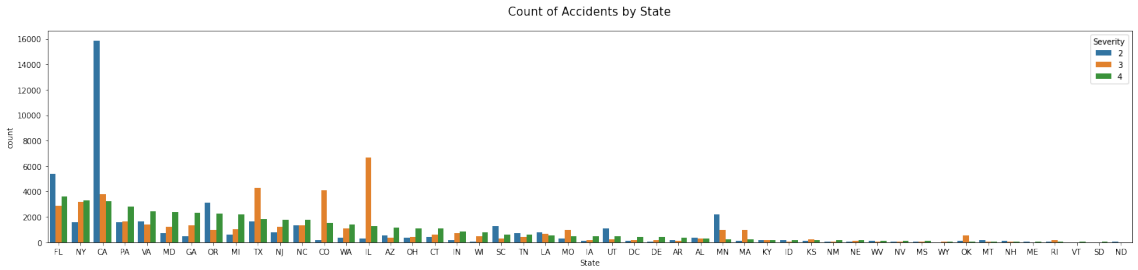

Figure 2.17: Accidents By State in order of Severity 4

As we can see states with generally higher population densities experience higher number of accidents.

We found another interesting feature in our data that states if the accident happened on the left side of the road or the right side of the road -

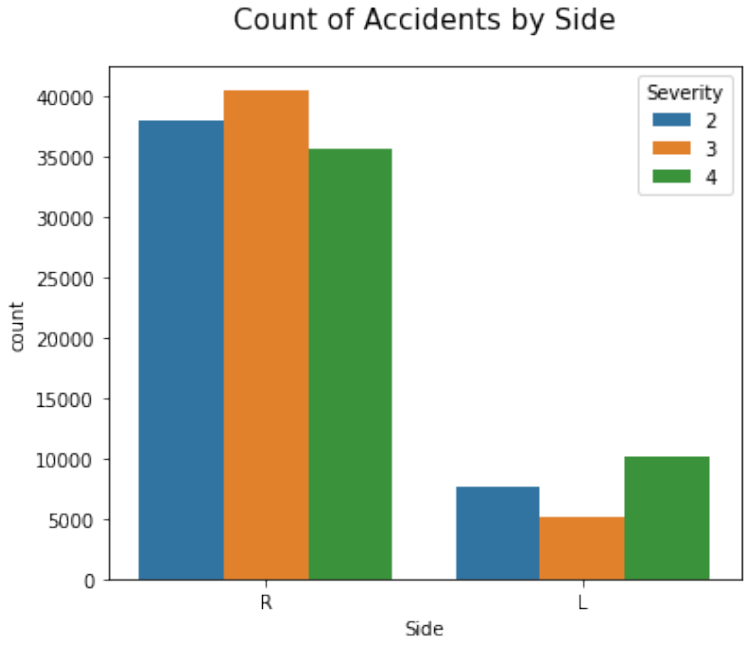

Figure 2.18: Accidents By Side of the Road

Interestingly, we see that a lot more accidents occur on the right side of the road than the left.

This seems to be the case because most right handed turns limit the visibility of the driver whereas left handed turns are usually easier to take.

We performed a little more analysis on location features in our data but didn't get any new insights from them and thus have not included those in our report. The analysis can still be found in our code.

2.3.3 Weather Condition Analysis

We found multiple features that conveyed the weather conditions at the time of the accident. We first plotted the distribution of the accidents based on the temperature, separated by the severity of the accident -

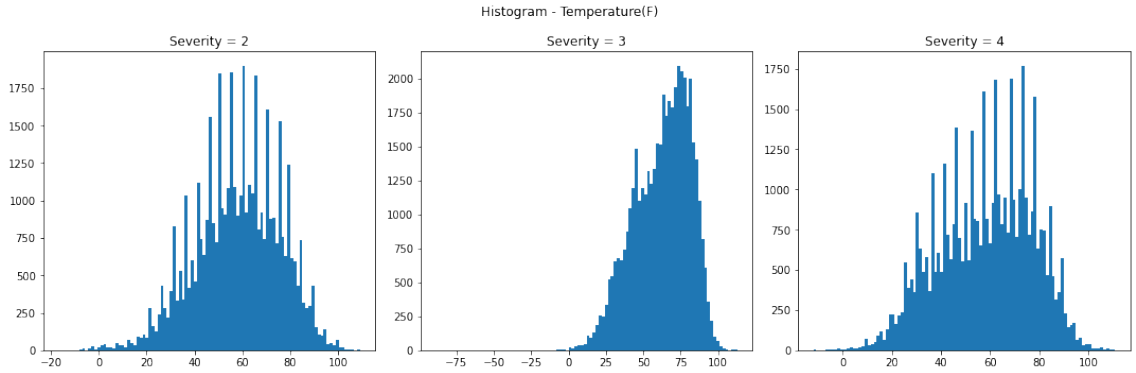

Figure 2.19: Accidents By Temperature

As we can see, most accidents occur when the temperature is around 40-80 Fahrenheit which indicates that more people travel using vehicles in pleasant and warm weather. Most accidents falling in this particular range of temperature is not very noteworthy because the average temperature across the US over the course of a year is around 40-80 Fahrenheit.

Now let us take a look at how wind chill affects accidents -

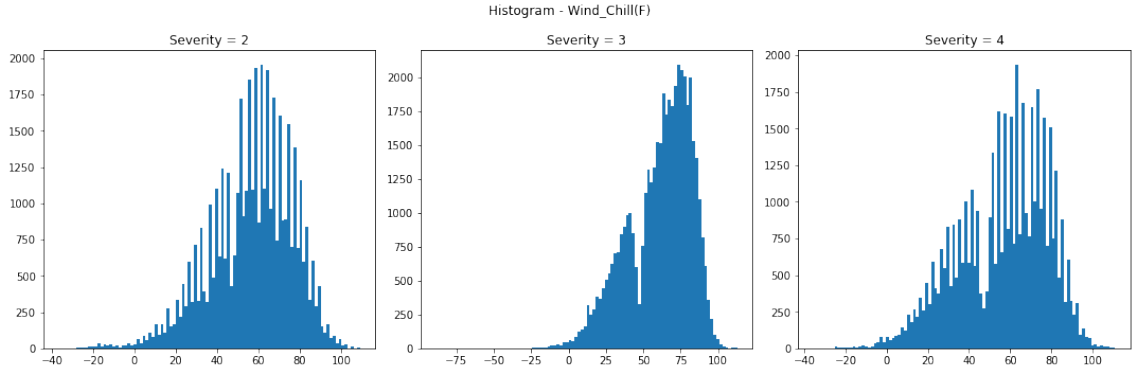

Figure 2.20: Accidents By Wind Chill

As expected, our graph looks very similar to the graph we have for temperature. This is because temperature and wind chill factor are directly related to each other.

Next we want to talk about the effect of humidity on accidents -

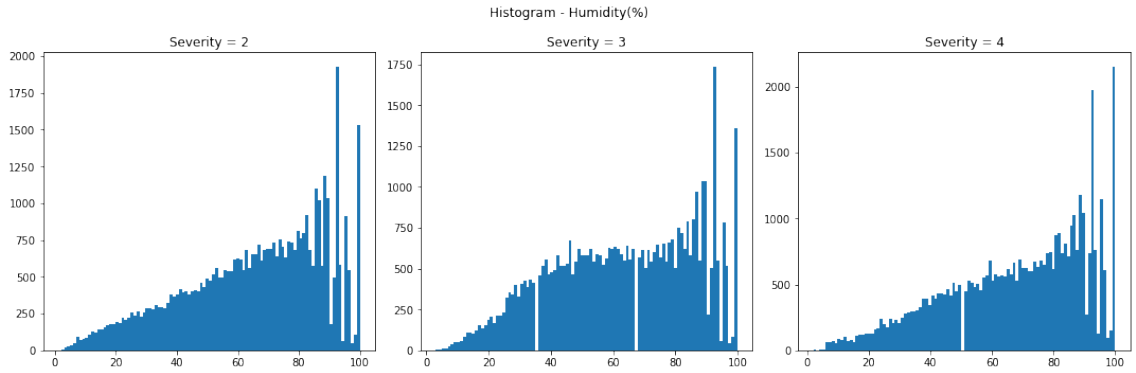

Figure 2.21: Accidents By Humidity

We know that Humidity is a good indicator of the chance to rain on a particular day. And interestingly we see that accidents rise almost perfectly linearly with a rise in humidity. This can be attributed to bad road conditions making the area more accident prone.

Let us take a look at how air pressure affect accidents -

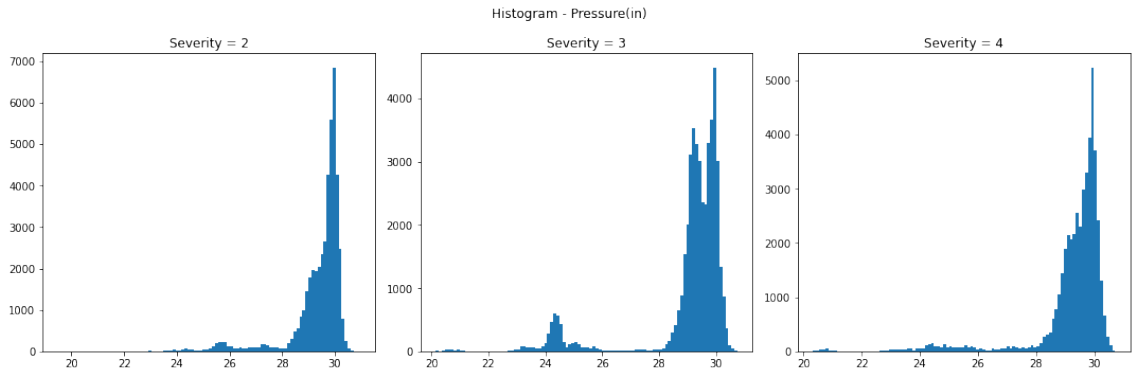

Figure 2.22: Accidents By Pressure

Since the pressure scale is very sensitive, we can see that our pressure data is concentrated around 30 BAR which is the normal amount of pressure. Low pressure situations like 24-26 BAR happen rarely but are still a cause for accidents since they cause heavy rains and sometimes storms and hurricanes.

Lastly, let us take a look at how wind-speed plays a role in accidents -

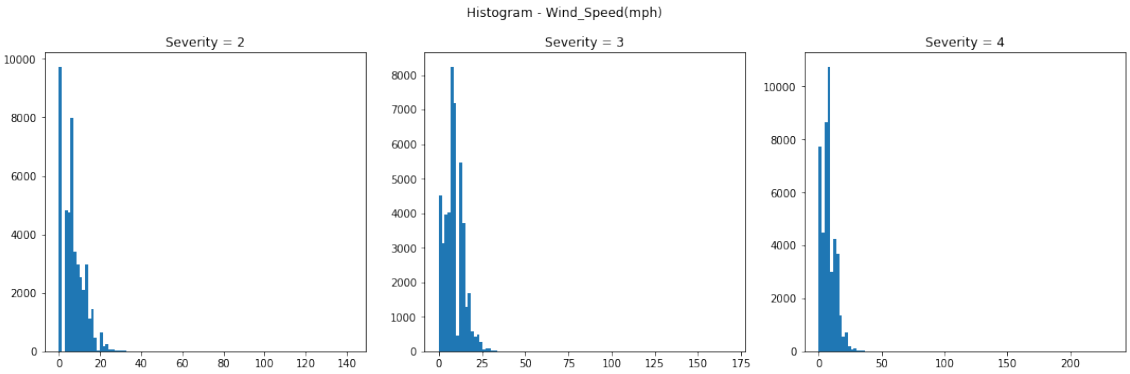

Figure 2.23: Accidents By Wind-Speed

We see a high number of accidents at a low wind speed not because wind speed doesn't affect number of accidents but because high speed winds only happen on very few days throughout the year. If we could plot data between days that there were high winds and there were accidents vs days there were slow winds and there were accidents we would be able to see a much clearer picture on how wind speed affects accidents. But we do not have access to this data and thus were unable to make such a graph.

We wanted to see how our categorical weather conditions (clear, cloudy, rain, heavy rain, snow, heavy snow and fog) affected accidents -

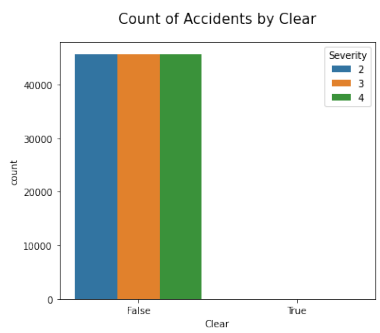

Figure 2.24: Accidents By Clear Weather

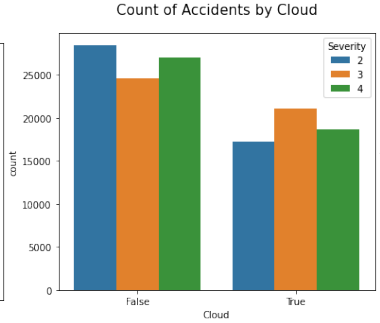

Figure 2.25: Accidents By Cloudy Weather

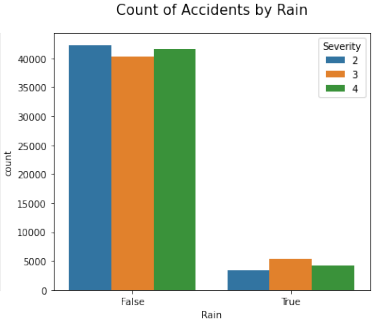

Figure 2.26: Accidents By Rainy Weather

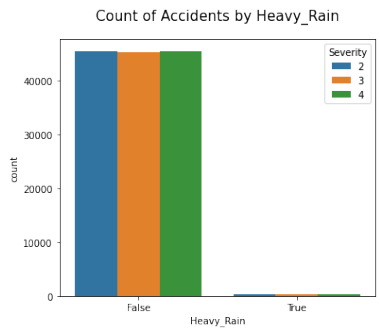

Figure 2.27: Accidents By Heavy-Rain

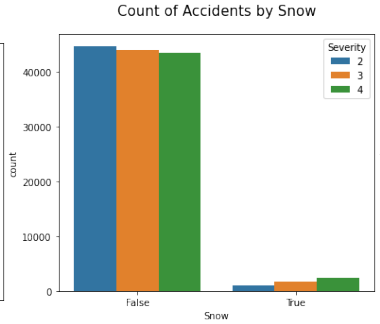

Figure 2.28: Accidents By Snowy Weather

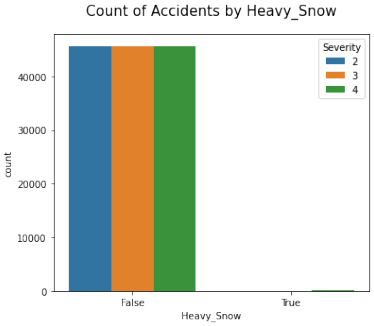

Figure 2.29: Accidents By Heavy-Snow Weather

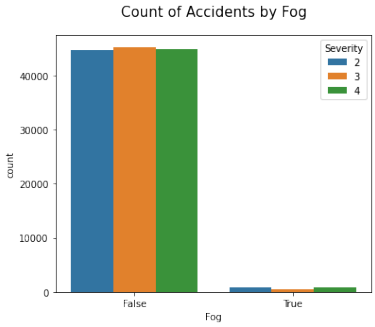

Figure 2.30: Accidents By Foggy Weather

As we can see from the above, more extreme weather conditions led to more severe accidents like in heavy-rain, snow, heavy-snow and fog. The other weather conditions had more even distributions of accident severity.

Now we want to check the effect of wind-direction on accident severity -

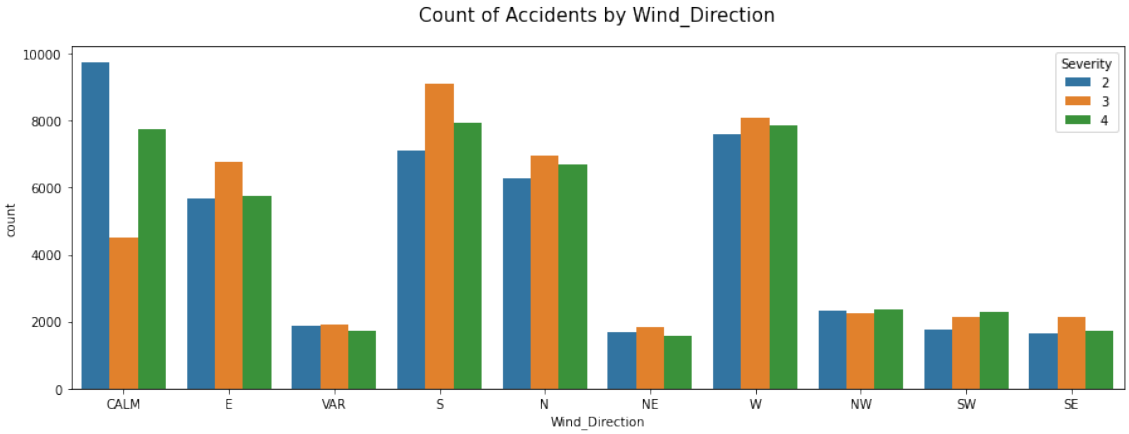

Figure 2.31: Accidents By Wind-Direction

As we can see, wind direction doesn't seem to have a particular impact on accident severity. After this we decided to drop the feature since it does not seem important to us.

2.3.4 Remaining Features Analysis

We had a bunch of remaining features that we wanted to plot graphs for and check together, these are -

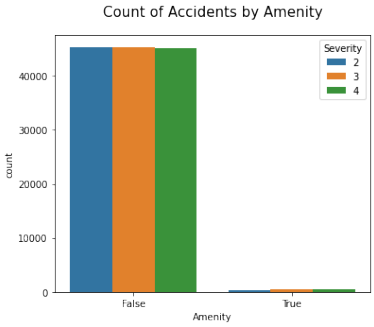

Figure 2.32: Accidents By Amenity

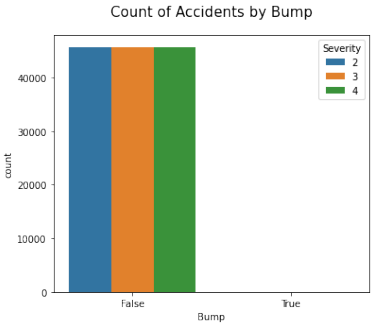

Figure 2.33: Accidents By Bump

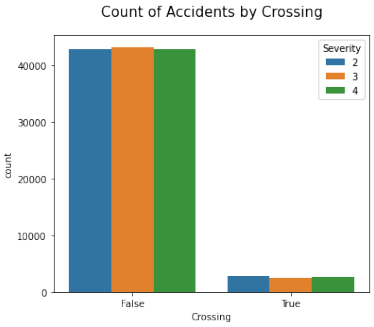

Figure 2.34: Accidents By Crossing

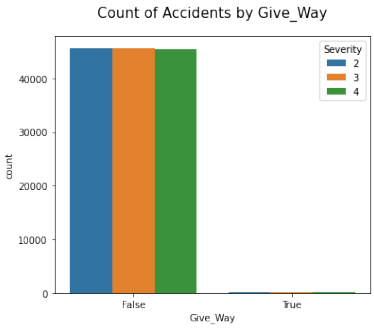

Figure 2.35: Accidents By Give-Way

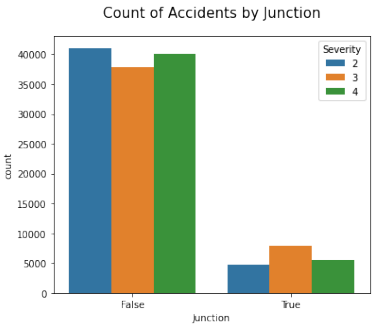

Figure 2.36: Accidents By Junction

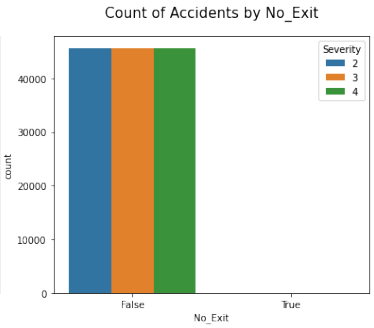

Figure 2.37: Accidents By Exit

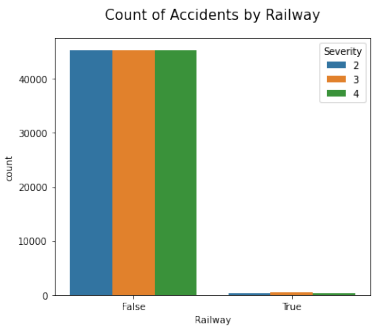

Figure 2.38: Accidents By Railway

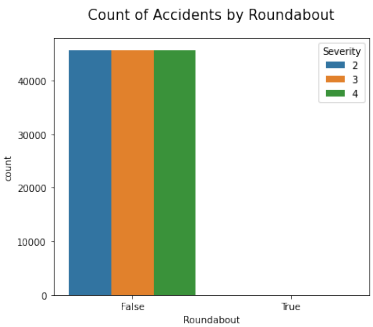

Figure 2.39: Accidents By Roundabout

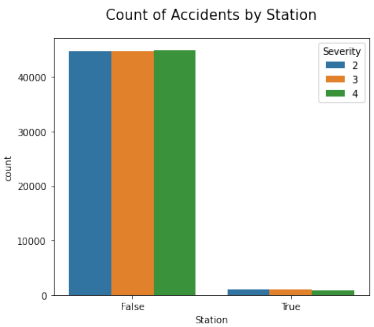

Figure 2.40: Accidents By Station

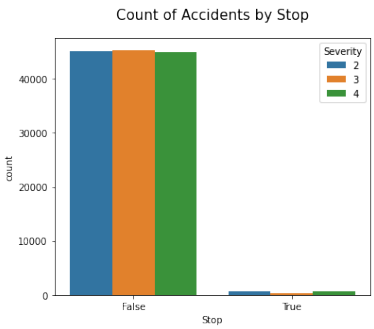

Figure 2.41: Accidents By Stop

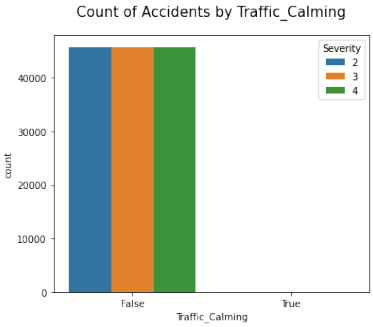

Figure 2.42: Accidents By Traffic-Calming

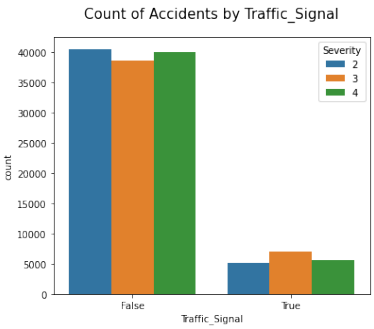

Figure 2.43: Accidents By Traffic-Signal

As we can see, most of the above features talk about if the accident happened near a traffic-signal, or a stop or a train station or a highway exit etc. These features do not seem to affect

the severity of the accident by a significant margin and thus are not very important for our purposes.

2.3.5 Correlation Analysis

For the final part of our analysis we want to plot the correlation between all the features in our data with respect to severity -

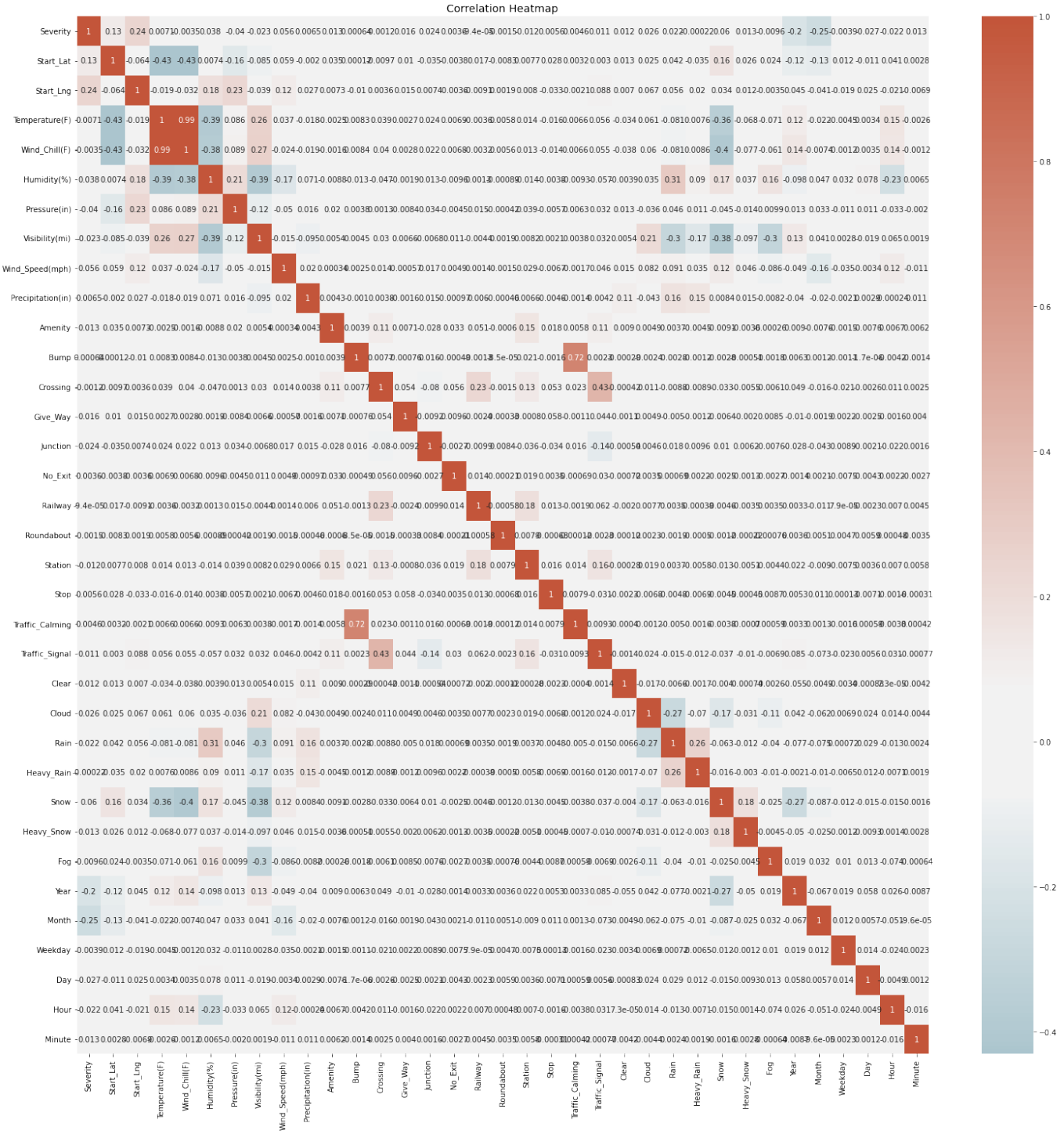

Figure 2.44: Accidents By Correlation Heat map

From the above we can deduce that Longitude, Latitude, Year and Month are the highest impacting features when looking at the severity of the accident.

Let us now fit our data to some models and see if our models can pick up anything that we have missed in our analysis and also if our models can accurately predict the severity of an accident based on some of the data we give to it.

Chapter 3

Models

Before we fit our data to any models, we must split our data in training sets and test sets so that we are not training on the exact same data that we are testing on. We conduct this split and get a final shape of our data that looks like the following -

```
Original data shape (137073, 50)
Training input shape (134331, 49)
Training output shape (134331,)
Testing input shape (2742, 49)
Testing output shape (2742,)
```

Figure 3.1: Shape of Original, Test and Training Sets

Thus we create a 98% and 2% train test split to fit to our models.

We will be testing the accuracy of 5 models - Random Forest Classifier, ADA Boost Classifier, Logistic Regression, Neural Networks and XGBoost. After this we will pick the model with the highest accuracy and calculate the feature importance according to that model.

3.1 Random Forest Classifier

The first model we tested is the Random Forest Classifier.

```
Optimal parameters RandomForestClassifier(max_depth=33, max_features='sqrt', n_estimators=46)
Best score 0.7733247806027291
Number of splits 5
```

Figure 3.2: Optional Parameters, score and splits

After testing a variety of parameters, we got the best results when using a max-depth of 33,

max-features of square root and 48 as our number of estimators.

This resulted in a Training accuracy and Training Confusion Matrix as shown below -

```
Training Accuracy: 0.9928460295836404
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.99      0.99      0.99     44784
           3       0.99      1.00      0.99     44718
           4       0.99      0.99      0.99     44829

    accuracy                           0.99    134331
   macro avg       0.99      0.99      0.99    134331
weighted avg       0.99      0.99      0.99    134331
```

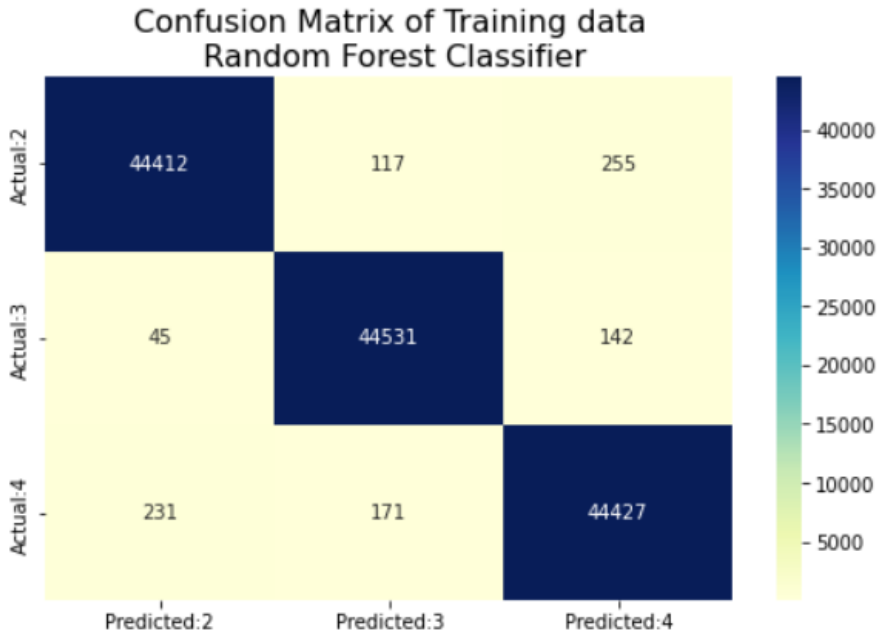

Figure 3.3: Training Accuracy and Training Confusion Matrix

As we can see, random forest classifier gets a very high precision recall and F1-Score on the training data and also a very high Training Accuracy of 99%. But how well does it perform on the test data ?

```
Testing Accuracy: 0.7811816192560175
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.84      0.79      0.82       907
           3       0.76      0.81      0.78       973
           4       0.74      0.74      0.74       862

    accuracy                           0.78      2742
   macro avg       0.78      0.78      0.78      2742
weighted avg       0.78      0.78      0.78      2742
```

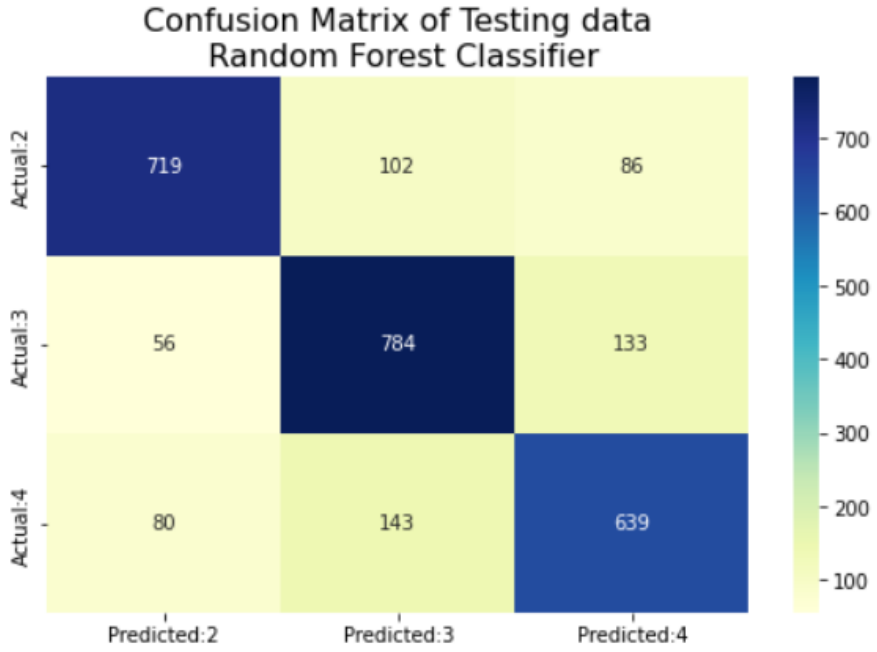

Figure 3.4: Testing Accuracy and Testing Confusion Matrix

On the testing data we see that the random forest classifier has a fairly lower Precision Recall and F1-score and a overall Testing accuracy of 78%

3.2 ADA Boost Classifier

The second model we tested is the ADA Boost Classifier.

```
Optimal parameters: AdaBoostClassifier(n_estimators=83)
Best score: 0.6120104106813583
Number of splits: 5
```

Figure 3.5: Optional Parameters, score and splits

After testing a variety of parameters, we got the best results when using 83 as our number of estimators.

This resulted in a Training accuracy and Training Confusion Matrix as shown below -

```
Training Accuracy: 0.619134823681801
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.70      0.75      0.72     44784
           3       0.60      0.68      0.64     44718
           4       0.54      0.43      0.48     44829

    accuracy                           0.62    134331
   macro avg       0.61      0.62      0.61    134331
weighted avg       0.61      0.62      0.61    134331
```

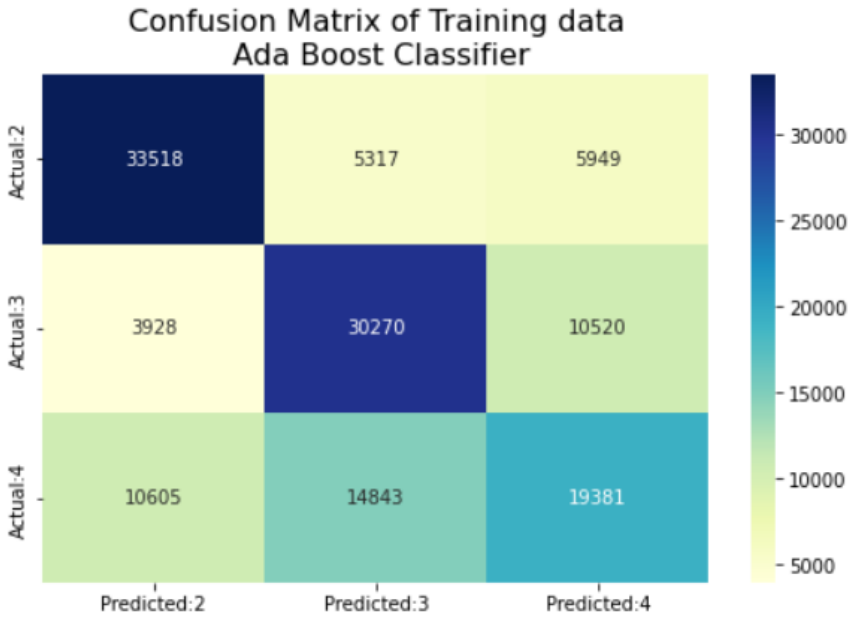

Figure 3.6: Training Accuracy and Training Confusion Matrix

As we can see, ADA Boost classifier gets fairly low precision recall and F1-Score on the training data and also a low Training Accuracy of 62%. But how well does it perform on the test data ?

```
Testing Accuracy: 0.6199854121079504
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.70      0.73      0.71       907
           3       0.62      0.67      0.64       973
           4       0.52      0.45      0.48       862

    accuracy                           0.62      2742
   macro avg       0.61      0.62      0.61      2742
weighted avg       0.61      0.62      0.62      2742
```

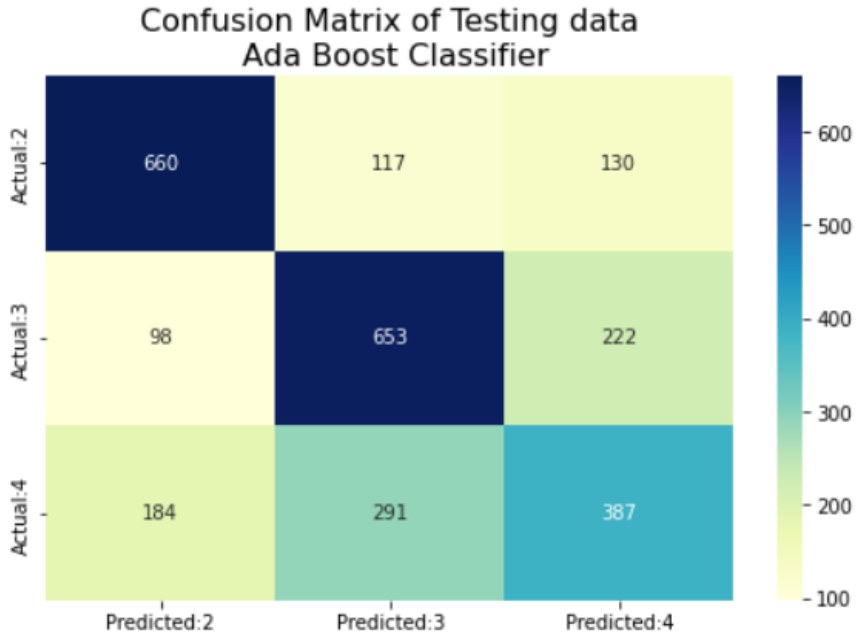

Figure 3.7: Testing Accuracy and Testing Confusion Matrix

On the testing data we see that the ADA Boost classifier has about the same low Precision Recall and F1-score and a overall Testing accuracy of 62%

3.3 Logistic Regression

The third model we tested is Logistic Regression.

After testing a variety of parameters, we got the best results when using Logistic Regression with a multi-class multionomial.

This resulted in a Training accuracy and Training Confusion Matrix as shown below -

```
Training Accuracy: 0.5259843223083279
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.58      0.65      0.62     44784
           3       0.51      0.61      0.56     44718
           4       0.45      0.31      0.37     44829

    accuracy                           0.53    134331
   macro avg       0.52      0.53      0.51    134331
weighted avg       0.52      0.53      0.51    134331
```

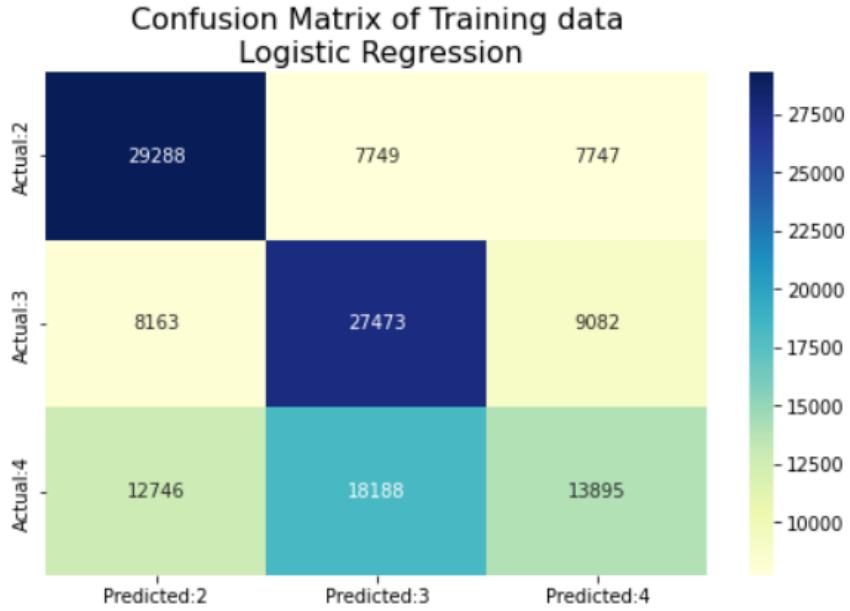

Figure 3.8: Training Accuracy and Training Confusion Matrix

As we can see, Logistic Regression gets a very low precision recall and F1-Score on the training data and also a low Training Accuracy of 52%. But how well does it perform on the test data ?

```
Testing Accuracy: 0.5448577680525164
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.59      0.67      0.63       907
           3       0.55      0.64      0.59       973
           4       0.45      0.31      0.37       862

    accuracy                           0.54      2742
   macro avg       0.53      0.54      0.53      2742
weighted avg       0.53      0.54      0.53      2742
```

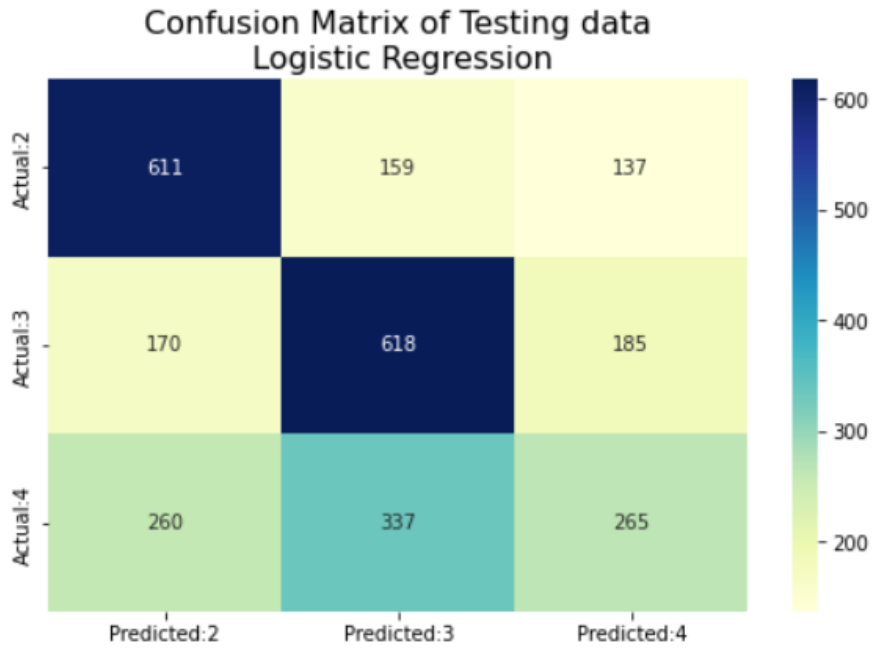

Figure 3.9: Testing Accuracy and Testing Confusion Matrix

On the testing data we see that Logistic Regression still has a bad and low Precision Recall and F1-score and a overall Testing accuracy of 54%

3.4 Neural Networks

The fourth model we tested are Neural Networks.

After testing a variety of parameters, we got the best results when using a Neural Network with 5 Dense Layers and 1 Output Layer.

Number of neurons in 5 hidden layers is fixed as: 256, 128, 64, 32 and 16 respectively.

This resulted in a Training accuracy and Training Confusion Matrix as shown below -

```
Training Accuracy: 0.6580982796227229
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.73      0.73      0.73     44784
           3       0.65      0.70      0.67     44718
           4       0.59      0.55      0.57     44829

    accuracy                           0.66    134331
   macro avg       0.66      0.66      0.66    134331
weighted avg       0.66      0.66      0.66    134331
```

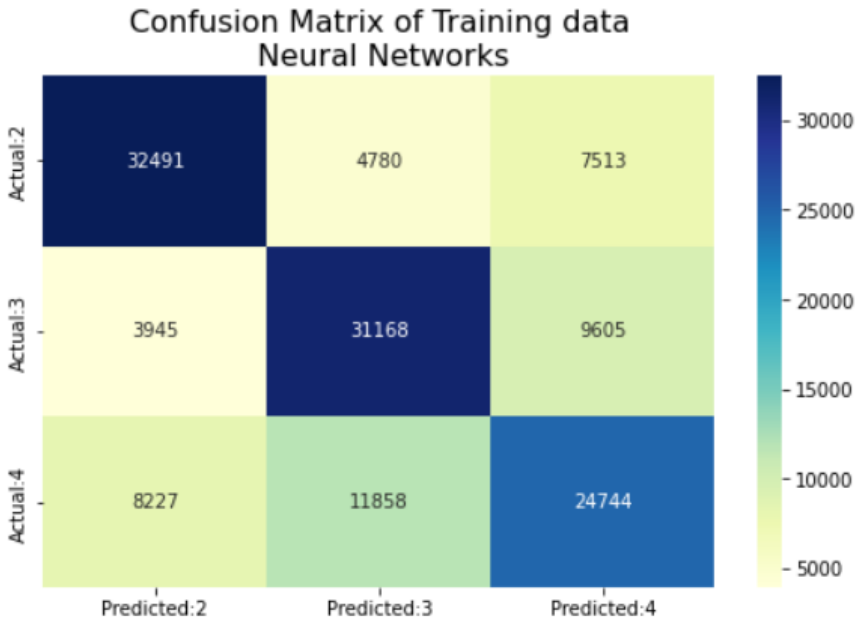

Figure 3.10: Training Accuracy and Training Confusion Matrix

As we can see, Neural Networks get a fairly average precision recall and F1-Score on the training data and a decent Training Accuracy of 65%.

```
Testing Accuracy: 0.6564551422319475
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.74      0.72      0.73       907
           3       0.65      0.71      0.68       973
           4       0.57      0.53      0.55       862

    accuracy                           0.66      2742
   macro avg       0.65      0.65      0.65      2742
weighted avg       0.66      0.66      0.66      2742
```

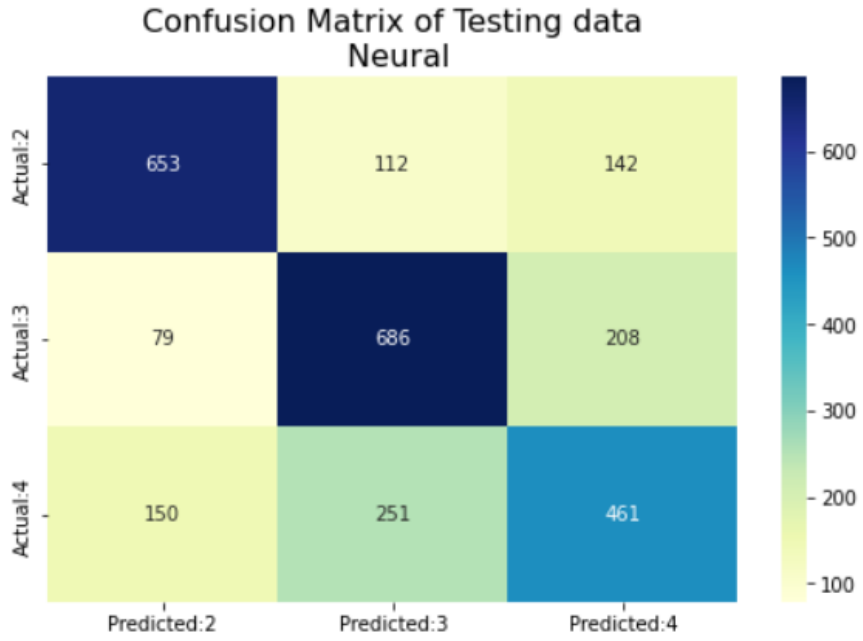

Figure 3.11: Testing Accuracy and Testing Confusion Matrix

On the testing data we see that Neural Networks have about the same average Precision Recall and F1-score and a overall decent Testing accuracy of 66%

3.5 XGBoost

The fifth model we tested is XGBoost.

After testing a variety of parameters, we got the best results by using a max depth of 12 and number of estimators as 150 -

This resulted in a Training accuracy and Training Confusion Matrix as shown below -

```
Training Accuracy: 0.9893620981009595
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.99      0.99      0.99     44784
           3       0.98      1.00      0.99     44718
           4       0.99      0.98      0.99     44829

    accuracy                           0.99    134331
   macro avg       0.99      0.99      0.99    134331
weighted avg       0.99      0.99      0.99    134331
```

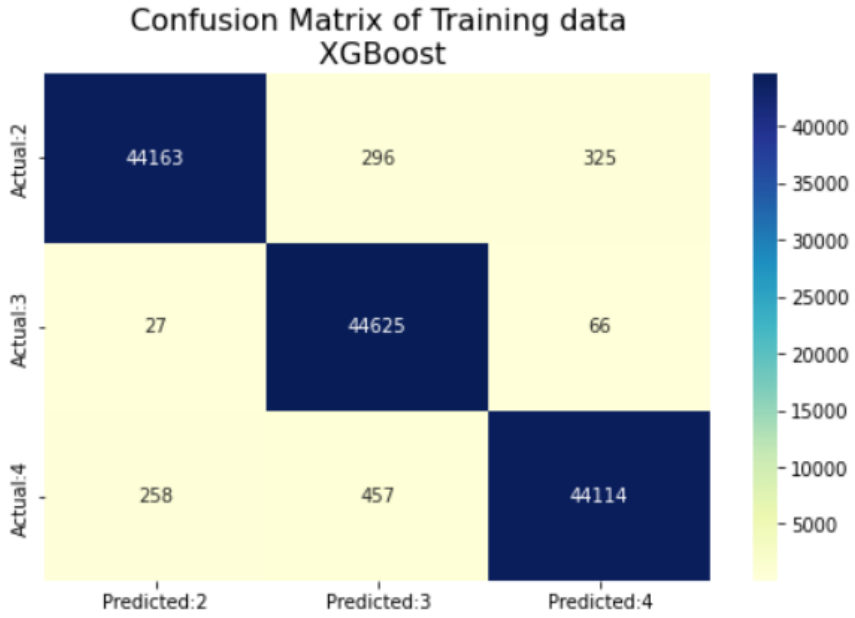

Figure 3.12: Training Accuracy and Training Confusion Matrix

As we can see, XGBoost gets a superb precision recall and F1-Score on the training data and an almost perfect Training Accuracy of 99%.

```
Testing Accuracy: 0.8026987600291758
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.85      0.81      0.83       907
           3       0.79      0.80      0.80       973
           4       0.77      0.79      0.78       862

    accuracy                           0.80      2742
   macro avg       0.80      0.80      0.80      2742
weighted avg       0.80      0.80      0.80      2742
```

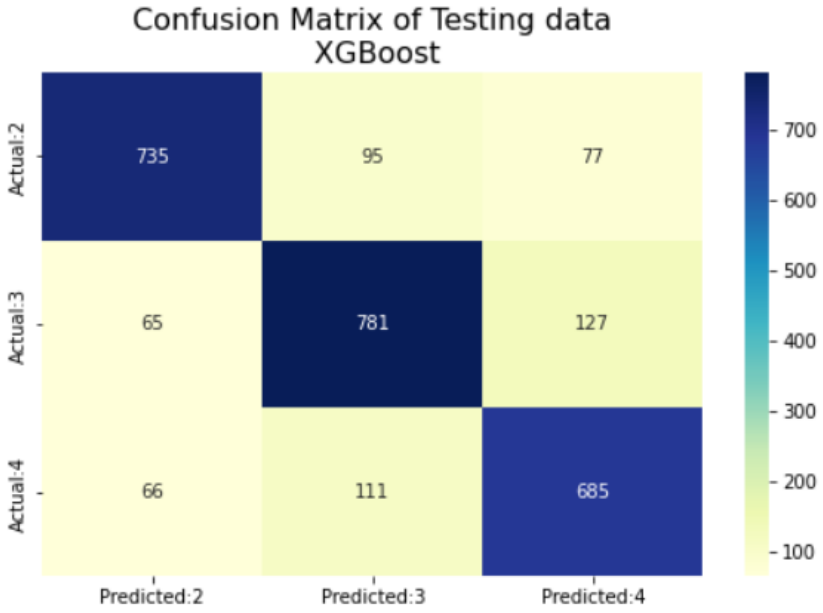

Figure 3.13: Testing Accuracy and Testing Confusion Matrix

On the testing data we see that XGBoost performs much lower in terms of Precision Recall and F1-score and a lower but still the best Testing accuracy of 80%

3.6 Gradient Boosting Classifier

The final model we tested is a Gradient Boosting Classifier.

After testing a variety of parameters, we got the best results by using a max depth of 13 and number of estimators as 120 -

This resulted in a Training accuracy and Training Confusion Matrix as shown below -

```
Training Accuracy: 0.9776522172841712
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.98      0.97      0.98     44784
           3       0.96      0.99      0.98     44718
           4       0.99      0.97      0.98     44829

    accuracy                           0.98    134331
   macro avg       0.98      0.98      0.98    134331
weighted avg       0.98      0.98      0.98    134331
```

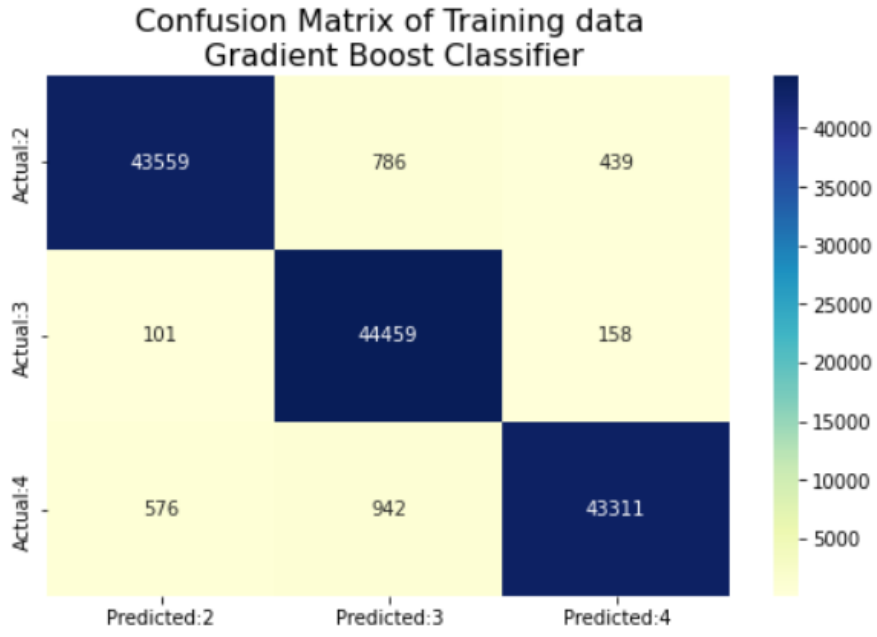

Figure 3.14: Training Accuracy and Training Confusion Matrix

As we can see, Gradient Boosting Classifier again gets a superb precision recall and F1-Score on the training data and an almost perfect Training Accuracy of 97%.

```
Testing Accuracy: 0.7972283005105762
Precision, Recall and f1-scores
              precision    recall  f1-score   support

           2       0.86      0.81      0.83       907
           3       0.79      0.80      0.79       973
           4       0.75      0.79      0.77       862

    accuracy                           0.80      2742
   macro avg       0.80      0.80      0.80      2742
weighted avg       0.80      0.80      0.80      2742
```

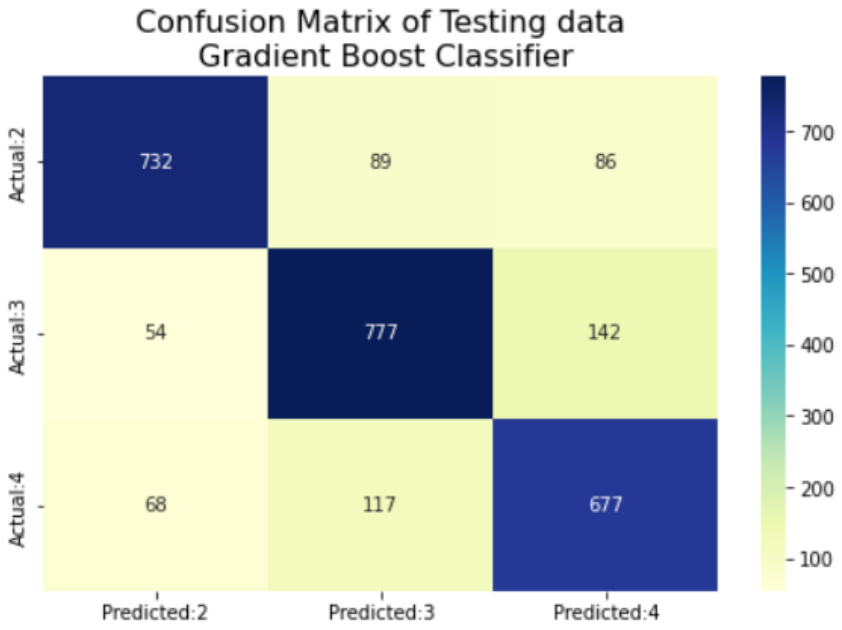

Figure 3.15: Testing Accuracy and Testing Confusion Matrix

On the testing data we see that Gradient Boosting Classifier performs much lower in terms of Precision Recall and F1-score and a lower but still fairly good Testing accuracy of 79%

Chapter 4

Feature Importance and Conclusion

4.1 Feature Importance

Finally we now use our best performing model - XGBoost to calculate the feature importance in our data set so that we can see what are best way to prevent high severity accidents. This is the graph we get when we plot the importance of our features using XGBoost -

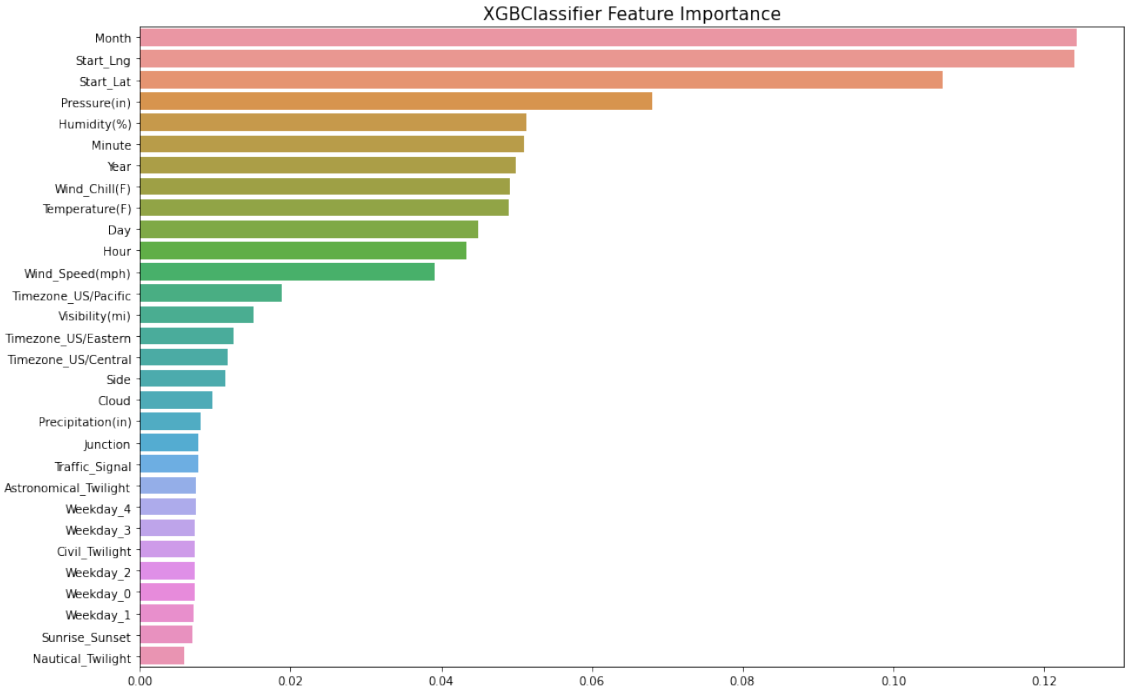

Figure 4.1: Performance of features

We can clearly see from the above plot that the month, latitude longitude and weather conditions are the most important features when looking at the severity of accidents.

4.2 Conclusion

We can see that the location of a person is a big factor in determining the severity of an accident. The month of the year during which a lot of travel takes place and also the weather conditions on a particular day are also major factors in determining the severity of an accident. There are multiple ways to help prevent high severity accidents -

- Build bigger and safer roads in densely populated areas so that the risk of accidents is lower.
- Making sure roads and cars are properly equipped to handle bad weather conditions.
- Avoiding travel unless necessary in extreme weather conditions.
- Building better infrastructure to help work time rush hour traffic.

After conducting this research we believe that all of us and the government must play a part so that the number and severity of accidents are reduced.