IBM Coursera Capstone

INTRODUCTION/BUSINESS PROBLEM

It is extremely essential for proper understanding of public safety when it comes to constructing roads by the municipal corporation or any other private entity. If the officials have access to various meaningful insights regarding these road accidents, the future construction of roads could be done in a manner that would ultimately lead to a safer and more seamless experience for the public. After studying and fabricating trends from the previous years, these entities will be at a much better position to make decisions which greatly benefit the public and the corresponding stakeholders. For the existing roads, these organisations can also put up various signs at strategic areas to further alert the civilians. Various infographics related to these car accidents could be issued in public interest to further alert the civilians regarding this and hence spread awareness.
We know how effective Machine Learning is when it comes to predicting/classifying based on some previous trends. By using some of the machine learning models I plan to contribute towards the safety of the civilians and elaborate various factors that go into a road accident.
Using inferences from the previous trends we can alert the public with some key findings and thus make them more careful towards car accidents thereby reducing it.
So my business problem aims to aid the road-building organisations to be more aware and educated about the car accidents before constructing newer roads in the city of Seattle so that these accidents don't repeat as often. It will compel the officials to strategically come up with various junction types to reduce the number of accidents accordingly.
This is no way is restricted only to Seattle, since all the cities that have roads similar to Seattle can take some inferences from this work as well (with a few modifications of course).

DATA DESCRIPTION

The dataset that I am going to be working with is a Collision dataset that records various factors when an accident takes place at different locations in the city of Seattle. These accidents have been recorded since the year 2004. The data for analysis was retrieved from the Road Accident Severity Data from the Seattle State Department of Transport from Data-Collisions. It is a CSV(Comma Separated Value) file that contains 194673 rows and 38 columns.
There are 37 other attributes that are a mixture of text and numbers, i.e., both categorical and numerical data types are present. The label is chosen to be the "accident severity" and is encoded as follows:

- 3—fatality
- 2b—serious injury
- 2—injury
- 1—prop damage
- 0—unknown

Here is a more detailed version description of the dataset as a whole:

```
In [11]: df.shape
Out[11]: (194673, 38)
```

```
In [16]: df.describe()
Out[16]:
```

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	INTKEY	SEVERITYCODE.1	PERSONCOUNT	PEI
count	194673.000000	189339.000000	189339.000000	194673.000000	194673.000000	194673.000000	65070.000000	194673.000000	194673.000000	19467
mean	1.298901	-122.330518	47.619543	108479.364930	141091.456350	141298.811381	37558.450576	1.298901	2.444427	(
std	0.457778	0.029976	0.056157	62649.722558	86634.402737	86986.542110	51745.990273	0.457778	1.345929	(
min	1.000000	-122.419091	47.495573	1.000000	1001.000000	1001.000000	23807.000000	1.000000	0.000000	(
25%	1.000000	-122.348673	47.575956	54267.000000	70383.000000	70383.000000	28667.000000	1.000000	2.000000	(
50%	1.000000	-122.330224	47.615369	106912.000000	123363.000000	123363.000000	29973.000000	1.000000	2.000000	(
75%	2.000000	-122.311937	47.663664	162272.000000	203319.000000	203459.000000	33973.000000	2.000000	3.000000	(
max	2.000000	-122.238949	47.734142	219547.000000	331454.000000	332954.000000	757580.000000	2.000000	81.000000	(

Other attributes:

```
In [11]: df.shape
Out[11]: (194673, 38)
```

```
In [16]: df.describe()
Out[16]:
```

INTKEY	SEVERITYCODE.1	PERSONCOUNT	PEDCOUNT	PEDCYLCOUNT	VEHCOUNT	SDOT_COLCODE	SDOTCOLNUM	SEGLANEKEY	CROSSWALKKEY
0.000000	194673.000000	194673.000000	194673.000000	194673.000000	194673.000000	194673.000000	1.149360e+05	194673.000000	1.946730e+05
8.450576	1.298901	2.444427	0.037139	0.028391	1.920780	13.867768	7.972521e+06	269.401114	9.782452e+03
5.990273	0.457778	1.345929	0.198150	0.167413	0.631047	6.868755	2.553533e+06	3315.776055	7.226926e+04
7.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.007024e+06	0.000000	0.000000e+00
7.000000	1.000000	2.000000	0.000000	0.000000	2.000000	11.000000	6.040015e+06	0.000000	0.000000e+00
3.000000	1.000000	2.000000	0.000000	0.000000	2.000000	13.000000	8.023022e+06	0.000000	0.000000e+00
3.000000	2.000000	3.000000	0.000000	0.000000	2.000000	14.000000	1.015501e+07	0.000000	0.000000e+00
0.000000	2.000000	81.000000	6.000000	2.000000	12.000000	69.000000	1.307202e+07	525241.000000	5.239700e+06

DATA CLEANING

As discussed in the previous IBM Data Science modules, data preprocessing is actually the most time consuming process. There was quite a few cleaning that had gone into this dataset particularly.
These were the columns that were kept from the original dataset:

```
In [10]: new_df = df[['X', 'Y', 'ADDRTYPE', 'LOCATION', 'COLLISIONTYPE',
                'PERSONCOUNT', 'PEDCOUNT', 'PEDCYLCOUNT', 'VEHCOUNT',
                'INCDATE', 'JUNCTIONTYPE',
                'INATTENTIONIND', 'UNDERINFL', 'WEATHER', 'ROADCOND', 'LIGHTCOND',
                'PEDROWNOTGRNT', 'SPEEDING', 'HITPARKEDCAR', 'SEVERITYCODE']]
```

```
Y                              float64
ADDRTYPE                        object
LOCATION                        object
COLLISIONTYPE                   object
PERSONCOUNT                      int64
PEDCOUNT                         int64
PEDCYLCOUNT                      int64
VEHCOUNT                         int64
INCDATE            datetime64[ns, UTC]
JUNCTIONTYPE                    object
INATTENTIONIND                   int64
```


```
Y                              float64
ADDRTYPE                        object
LOCATION                        object
COLLISIONTYPE                   object
PERSONCOUNT                      int64
PEDCOUNT                         int64
PEDCYLCOUNT                      int64
VEHCOUNT                         int64
INCDATE            datetime64[ns, UTC]
JUNCTIONTYPE                    object
INATTENTIONIND                   int64
```

These were some of the values that were dropped from each of the columns based on a closer look into the data. The reason behind dropping was to avoid unnecessary noise and allow the classifiers to perform well. They also didn't convey any significant meaning to the outputs.

In [24]:
```
new_df = new_df[new_df.ROADCOND != 'Other']
new_df = new_df[new_df.LIGHTCOND != 'Other']
new_df = new_df[new_df.ROADCOND != 'Standing Water']
new_df = new_df[new_df.ROADCOND != 'Sand/Mud/Dirt']
new_df = new_df[new_df.ROADCOND != 'Oil']
new_df = new_df[new_df.LIGHTCOND != 'Dark - Unknown Lighting']
new_df = new_df[new_df.COLLISIONTYPE != 'Pedestrian']
new_df = new_df[new_df.JUNCTIONTYPE != 'Ramp Junction']

new_df = new_df[new_df.COLLISIONTYPE != 'Cycles']
new_df = new_df[new_df.COLLISIONTYPE != 'Right Turn']
new_df = new_df[new_df.COLLISIONTYPE != 'Head On']
new_df = new_df[new_df.COLLISIONTYPE != 'Other']
new_df = new_df[new_df.ROADCOND != 'Unknown']
new_df = new_df[new_df.JUNCTIONTYPE != 'Unknown']
new_df = new_df[new_df.WEATHER != 'Unknown']
new_df = new_df[new_df.WEATHER != 'Fog/Smog/Smoke']
new_df = new_df[new_df.WEATHER != 'Sleet/Hail/Freezing Rain']
new_df = new_df[new_df.WEATHER != 'Blowing Sand/Dirt']
new_df = new_df[new_df.WEATHER != 'Severe Crosswind']
new_df = new_df[new_df.WEATHER != 'Partly Cloudy']
new_df = new_df[new_df.WEATHER != 'Other']
new_df = new_df[new_df.LIGHTCOND != 'Unknown']
```

In [25]:
```
new_df = new_df.dropna(subset=["ADDRTYPE","ROADCOND","LIGHTCOND","WEATHER","LOCATION","X","Y", "COLLISIONTYPE"],axis=0)
```

In [26]:
```
new_df.shape
```

Out[26]:
```
(128131, 21)
```

Some more cleaning:

```
new_df["UNDERINFL"] = new_df["UNDERINFL"].replace(['N', '0', '1', 'Y'], [0,0,1,1])
new_df["UNDERINFL"] = new_df["UNDERINFL"].replace([np.nan], [0])
```

In [30]:
```
new_df["INATTENTIONIND"] = new_df["INATTENTIONIND"].replace([np.nan, 'Y'], [0,1])
```

In [31]:
```
new_df["SPEEDING"] = new_df["SPEEDING"].replace([np.nan, 'Y'], [0, 1])
```

In [32]:
```
new_df["PEDROWNOTGRNT"] = new_df["PEDROWNOTGRNT"].replace([np.nan, 'Y'], [0, 1])
```

In [33]:
```
new_df["HITPARKEDCAR"] = new_df["HITPARKEDCAR"].replace(['N', 'Y'], [0, 1])
```

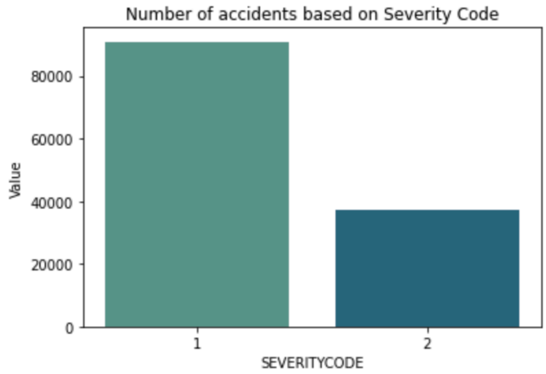
After performing the above steps, the data was ready to be explored to gain some meaningful insights.

DATA EXPLORATION

Data Exploration is an essential process to understand the data that we are going to work with. There were quite a few insights that I had gained after exploring the data and visualising the results. It did break a few common misconceptions that I at least had when it came to road accidents. We will see them below:

Figure 1.

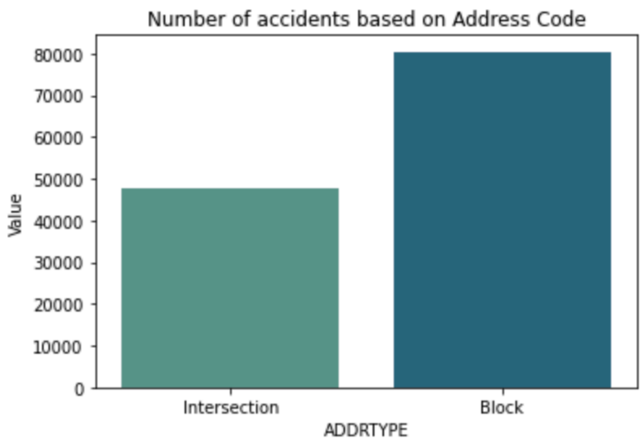
Shows the number of accidents for each severity type, where 1 means property damage and 2 means injury. It is surprising to know that there have been no reported serious injuries or fatalities in car accidents.




```
1    90937
2    37194
Name: SEVERITYCODE, dtype: int64
```

Figure 2.

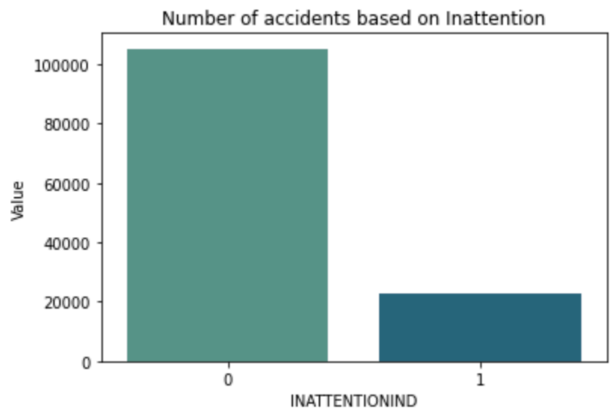
Shows the number of accidents in different addresses. Here, there are Blocks and Junctions where the majority of the accidents have taken place.




```
Block           80455
Intersection    47676
Name: ADDRTYPE, dtype: int64
```

Figure 3.

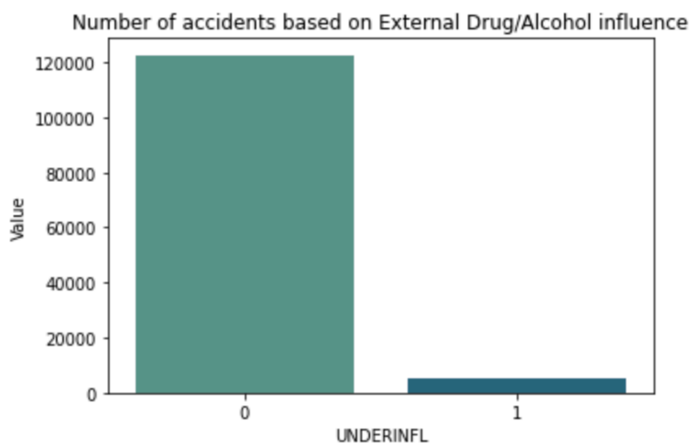
Shows the number of accidents caused due to Inattention. Here, 0 stands for No and 1 stands for Yes.




```
0     105315
1      22816
Name: INATTENTIONIND, dtype: int64
```

Figure 4.

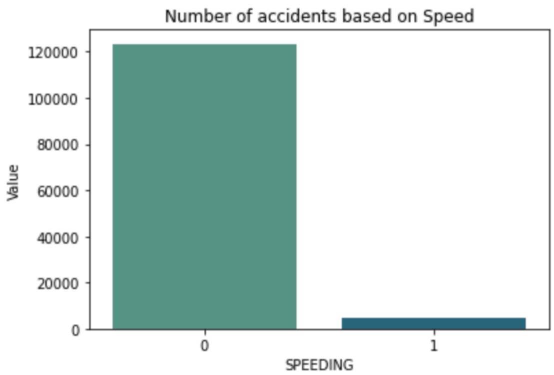
Shows the number of accidents caused due to external drug/alcohol influence. Here, 0 stands for No and 1 stands for Yes.
It is quite surprising to see that very accidents have occurred that have involved people under the influence of drugs or alcohol. This could mean that people are acting responsible and not driving when they are under the influence and that the cops are doing a great job at enforcing the rules out there in the city of Seattle.




```
0     122625
1       5506
Name: UNDERINFL, dtype: int64
```

Figure 5.

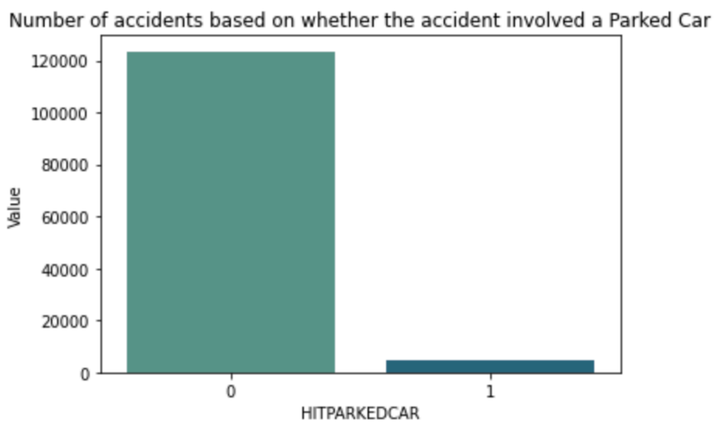
Shows the number of accidents caused due to Speeding. Here, 0 stands for No and 1 stands for Yes.
As we can see, most of the accidents have not occurred due to speeding. This might be the reason why there are no serious injuries or fatalities due to these accidents.




```
0    123403
1      4728
Name: SPEEDING, dtype: int64
```

Figure 6.

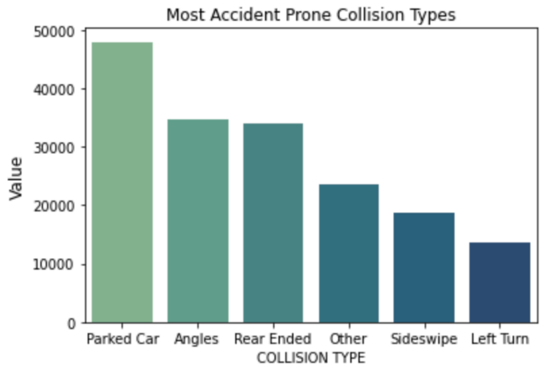
Shows the number of accidents involving a parked car. Here, 0 stands for No and 1 stands for Yes.




```
0    123483
1      4648
Name: HITPARKEDCAR, dtype: int64
```

Figure 7.

Shows the number of accidents and their types of collision. Here, we can see that highest type of collisions have occurred to parked cars followed by collisions at an angle.




```
Angles         33450
Parked Car     33014
Rear Ended     31413
Sideswipe      17025
Left Turn      13229
Name: COLLISIONTYPE, dtype: int64
```

Figure 8.

Shows the number of accidents in different weather conditions. Here, we can see that highest number of collisions have occurred when the weather conditions were clear. Again contrary to what a normal human instinct would be. This might also tell us that the most common weather condition in Seattle is clear. The two bars are separated based on the Severity Type of the accident just for a better comparison.

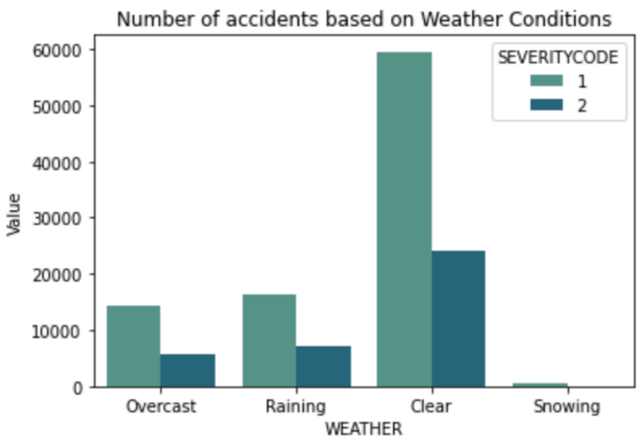
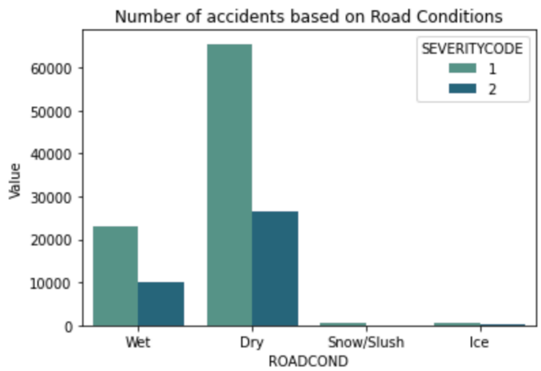


Figure 9.

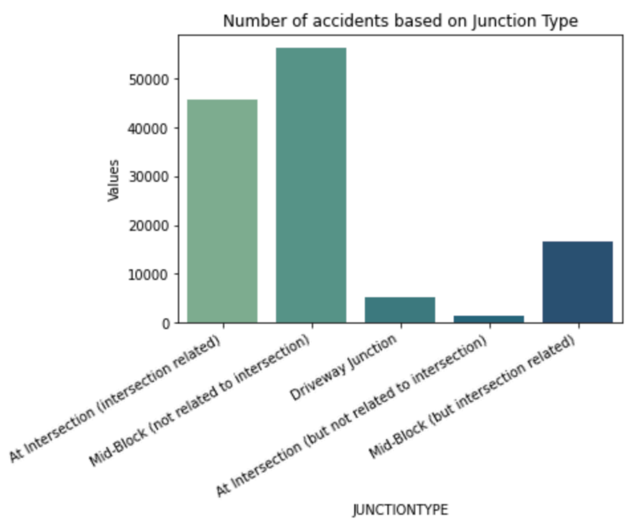
Shows the number of accidents in different road conditions. Here, we can see that highest number of collisions have occurred when the roads were dry.




```
Dry           92106
Wet           33238
Ice             540
Snow/Slush      531
Name: ROADCOND, dtype: int64
```

Figure 10.

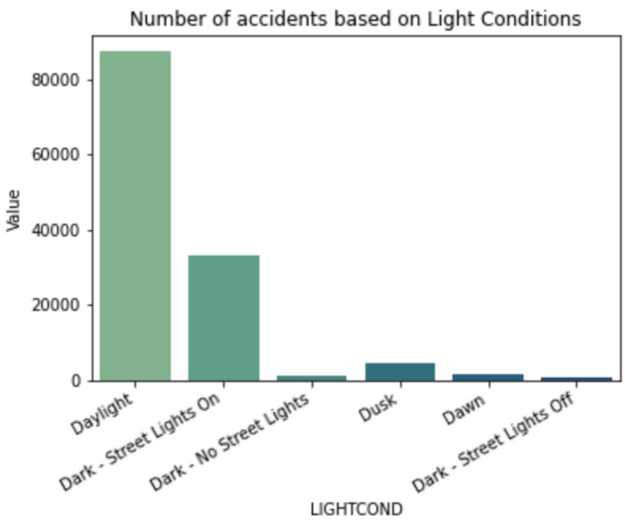
Shows the number of accidents in different types of junctions. Here, we can see that highest number of collisions have occurred in the Mid-block Junction closely followed by at intersections.




```
Mid-Block (not related to intersection)              56358
At Intersection (intersection related)               45763
Mid-Block (but intersection related)                 16639
Driveway Junction                                     5100
At Intersection (but not related to intersection)     1421
Name: JUNCTIONTYPE, dtype: int64
```

Figure 11.

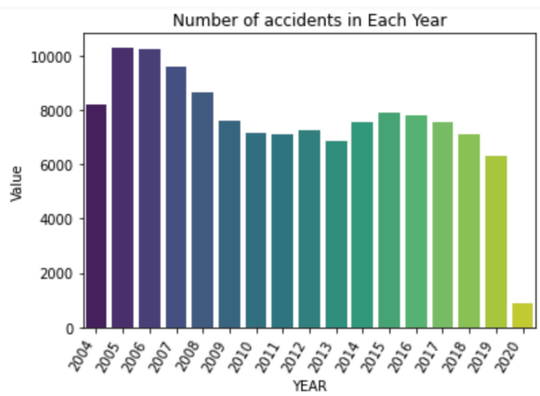
Shows the number of accidents in different light conditions. Here, we can see that highest number of collisions have occurred in daylight. Again this is contrary to human instincts, but this might imply that the drivers are more careful in the night and thus less prone to accidents. Also, there are very few cases where street lights were off or not present. This shows the efficiency of the road managing corporation of Seattle.




```
Daylight                    87234
Dark - Street Lights On     33260
Dusk                         4309
Dawn                         1612
Dark - No Street Lights       932
Dark - Street Lights Off      784
Name: LIGHTCOND, dtype: int64
```

Figure 12.

Shows the number of accidents over the different years. Here, we can see that highest number of collisions have occurred in 2005 closely followed by 2006. In the past 5 years, there has been a successive decline in the number of accidents in Seattle which is a good sign. In 2020, there are significantly lesser number of accidents which might be a result of the global pandemic response by us humans.




```
2005    10305
2006    10242
2007     9599
2008     8638
2004     8186
2015     7917
2016     7798
2009     7629
2017     7579
2014     7534
2012     7247
2010     7141
2011     7115
2018     7096
2013     6881
2019     6324
2020      900
Name: YEAR, dtype: int64
```

Figure 13.

This is a heat map that shows the correlation between the various features of the dataset. This was purely done to understand the relationships between the various attributes in the acquired dataset.

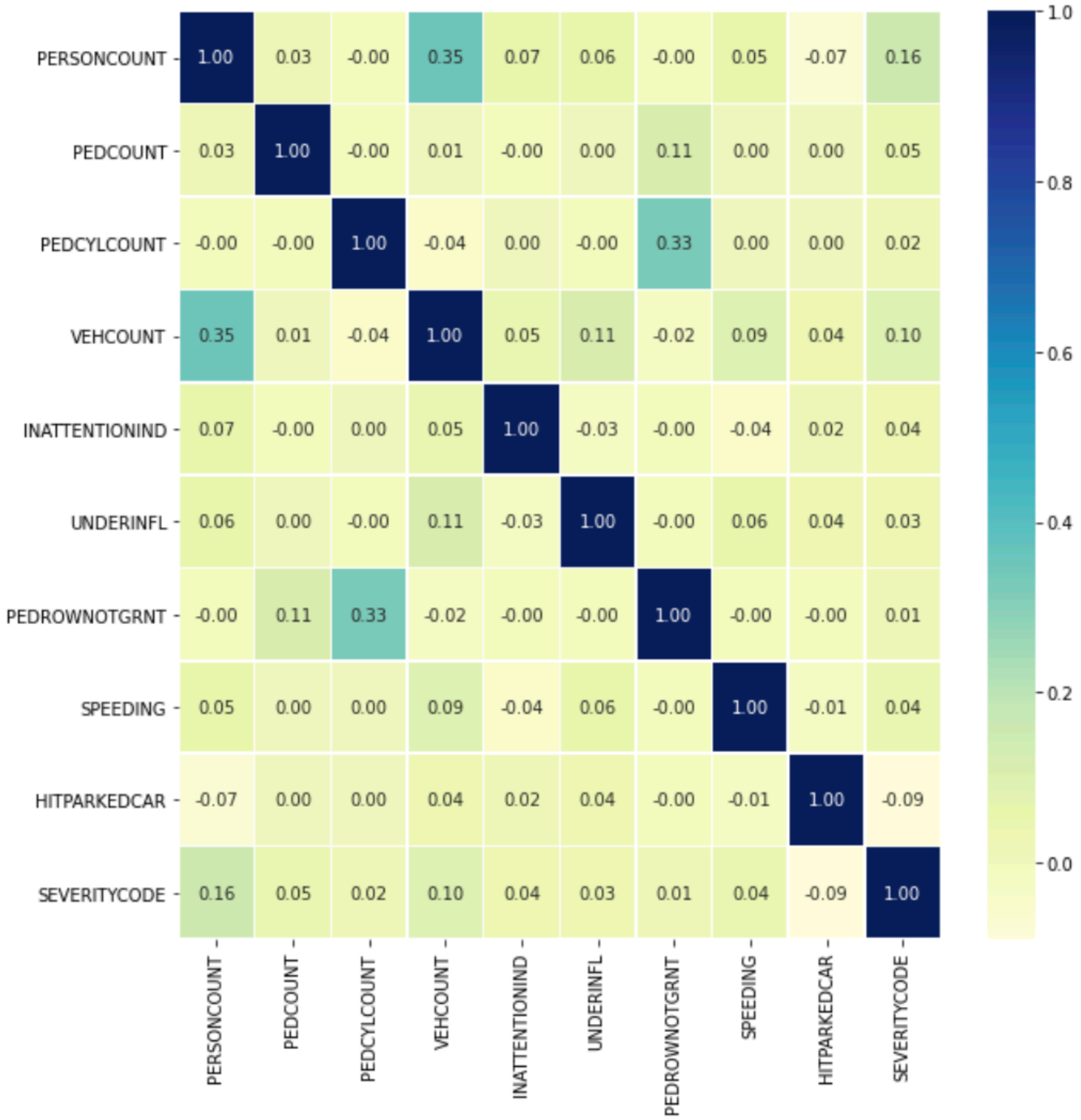
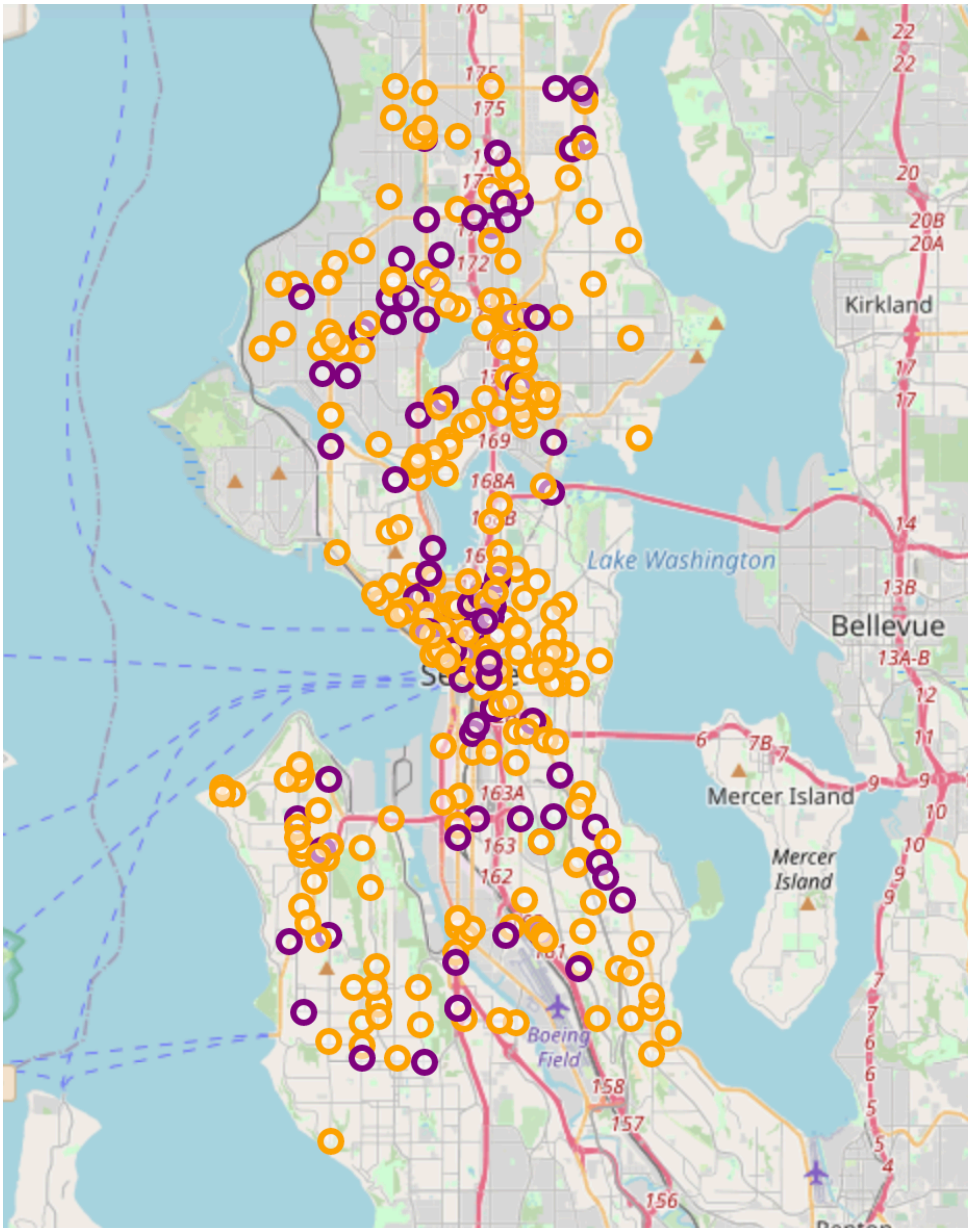


Figure 14.

This is a map of Seattle that shows the longitudes and latitudes of the exact accident spots in Seattle. 500 locations are selected so that map looks presentable. As it is visible, there is a pretty dense spot in the city-centre.



So this was the thorough Data Exploration done on the dataset. Let us move on the Data Preprocessing for finally fitting in the machine learning models.

DATA PREPROCESSING

There were quite a few categorical variables in the dataset and which I believed did impact the severity of the accident. This was also indicated from the heat map. Hence, to be able to use these categorical features in the machine learning models I applied a technique called One Hot Encoding. It was done using the get_dummies method in pandas. Here is an example:

```
In [63]: oh1 = pd.get_dummies(new_df['ADDRTYPE'])
         oh1.head()
```

Out[63]:

	Block	Intersection
0	0	1
1	1	0
2	1	0
4	0	1
5	0	1

[ADDRTYPE, COLLISIONTYPE, JUNCTIONTYPE, WEATHER, ROADCOND, LIGHTCOND] were the attributes that I have applied OHE on. Here is how the final list of features look like after OHE all the categorical attributes:

```
In [78]: features = final_df[['PERSONCOUNT', 'PEDCOUNT', 'PEDCYLCOUNT',
                'VEHCOUNT', 'INATTENTIONIND', 'UNDERINFL', 'PEDROWNOTGRNT',
                'SPEEDING', 'HITPARKEDCAR', 'Block',
                'Intersection', 'Angles', 'Left Turn', 'Parked Car', 'Rear Ended',
                'Sideswipe', 'At Intersection (but not related to intersection)',
                'At Intersection (intersection related)', 'Driveway Junction',
                'Mid-Block (but intersection related)',
                'Mid-Block (not related to intersection)', 'Clear', 'Overcast',
                'Raining', 'Snowing', 'Dry', 'Ice', 'Snow/Slush', 'Wet',
                'Dark - Street Lights On', 'Dawn', 'Daylight', 'Dusk']]

In [79]: X = np.asarray(features)
```

This is the label:

```
In [80]: label = final_df[['SEVERITYCODE']]

In [81]: Y = np.asarray(label)
```

Here is the final shapes of the training and testing dataset:

```
In [83]: print(X.shape)
         print(Y.shape)

         (126415, 33)
         (126415, 1)

In [86]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=21)

In [87]: print(X_train.shape)
         print(X_test.shape)
         print(y_train.shape)
         print(y_test.shape)

         (101132, 33)
         (25283, 33)
         (101132, 1)
         (25283, 1)
```

DATA MODELLING

I have used 4 classifiers to predict the severity of the accident based on the above given features.
The 4 classifiers used are:

1. Logistic Regression
2. Decision Tree Classifier
3. Random Forres Ensemble Classifier
4. K Nearest Neighbour

LOGISTIC REGRESSION:

In [111]:
```
from sklearn.linear_model import LogisticRegression

LR = LogisticRegression(C = 0.01, solver = 'liblinear').fit(X_train, y_train)
yPredLR = LR.predict(X_test)
```

```
/Library/Python/3.7/site-packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector y was passed
when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
```

DECISION TREE:

In [91]:
```
from sklearn.tree import DecisionTreeClassifier

DTree = DecisionTreeClassifier(criterion="entropy", max_depth=4)
DTree.fit(X_train, y_train)
yPredTree = DTree.predict(X_test)
yPredTree[0:10]
```

Out[91]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

RANDOM FORREST:

In [93]:
```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(n_estimators=100)
clf.fit(X_train,y_train)
yPredForrest = clf.predict(X_test)
yPredForrest[0:10]
```

```
/Library/Python/3.7/site-packages/ipykernel_launcher.py:3: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  This is separate from the ipykernel package so we can avoid doing imports until
```

Out[93]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 2, 1])
```

K Nearest Neighbour:

In [95]:
```
from sklearn.neighbors import KNeighborsClassifier
KNN_model = KNeighborsClassifier(n_neighbors=3)
KNN_model.fit(X_train, y_train)
yPredKNN = KNN_model.predict(X_test)
```

```
/Library/Python/3.7/site-packages/ipykernel_launcher.py:3: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  This is separate from the ipykernel package so we can avoid doing imports until
```

RESULTS

LOGISTIC REGRESSION:

Evaluation Metric	Value
Accuracy	0.723
F1 Score	0.661
Jaccard Similarity	0.709

DECISION TREE:

Evaluation Metric	Value
Accuracy	0.721
F1 Score	0.655
Jaccard Similarity	0.709

RANDOM FORREST:

Evaluation Metric	Value
Accuracy	0.713
F1 Score	0.671
Jaccard Similarity	0.694

K NEAREST NEIGHBOUR:

Evaluation Metric	Value
Accuracy	0.691
F1 Score	0.670
Jaccard Similarity	0.662

CONCLUSION

As it is evident from the results, Logistic Regression model slightly outperforms the decision tree classifier for predicting the severity of the accidents.
The entire data exploration along with all the cycles of the Data Science methodology have given us a lot of insights that can greatly benefit the stakeholders and the people of Seattle. If these insights are looked into and decisions are made based on it, the government will indeed save a lot of money by reducing the numerous property damage due to these accidents. Civilians will also feel safer walking the roads as the number of injuries incurred to them will reduce!

Overall this was a great project assigned to us which tested all the skills learnt as a part of this specialisation course.