




IBM DATA SCIENCE CAPSTONE PROJECT

CAR ACCIDENT SEVERITY IN SEATTLE CITY

By
Sam Joseph



INTRODUCTION

Seattle, a city in US, is surrounded by water, mountains and evergreen forests, and contains thousands of acres of parkland. Washington State's largest city, it's home to a large tech industry. Seattle is known to have well established roads network that caters to heavy traffic.



BUSINESS PROBLEM

This project aims to reduce the severity of accidents in the city of Seattle; hence we need to build an algorithm to predict the severity of an accident based on the current weather, road and visibility conditions. The main data attributes which we will use for the analysis will be

- Weather Condition
- Car Speeding
- Light Condition
- Road Condition

TARGETED AUDIENCE

This project will benefit the drivers in Seattle city in deciding the route to be taken based on the weather and traffic conditions by reviewing the accident severity prediction model during adverse weather conditions.

DATA SOURCES

To proceed with this project, the data has been sourced from below repository

<https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv>

The dataset has information gathered on the road traffic accidents of Seattle City.

APPROACH

- Different data visualization methods are used to study and analyze the data.
- The predictor or target variable will be 'SEVERITYCODE' because it is used to measure the severity of an accident from 1 to 2 within the dataset. Other attributes used to weigh the severity of an accident are 'WEATHER', 'ROADCOND', 'LIGHTCOND' and 'SPEEDING'.
- Severity code 1 stands for property damage and
- Severity code 2 stands for injury during an accident.

METHODOLOGY

DATA SCIENCE TOOLS USED

- The tools used are pandas, numpy, matplotlib libraries.
- Seaborn graphics library was used for general statistics to display graphs.
- Matplotlib Library was used to generate bar chart.
- Used K-Means algorithm from Scikit-Learn Library for clustering attributes on generated data.
- Folium Library was used to create maps with popups labels to allow quick identification of accident locations with popup info on Severity, weather condition, road condition etc .

DATA WRANGLING

Uncleaned Data

	SEVERITYCODE	LAT	LON	ADDRTYPE	COLLISIONTYPE	WEATHER	ROADCOND	LIGHTCOND	SPEEDING
0	2	-122.323148	47.703140	Intersection	Angles	Overcast	Wet	Daylight	NaN
1	1	-122.347294	47.647172	Block	Sideswipe	Raining	Wet	Dark - Street Lights On	NaN
2	1	-122.334540	47.607871	Block	Parked Car	Overcast	Dry	Daylight	NaN
3	1	-122.334803	47.604803	Block	Other	Clear	Dry	Daylight	NaN
4	2	-122.306426	47.545739	Intersection	Angles	Raining	Wet	Daylight	NaN

Cleaned Data

	SEVERITYCODE	LAT	LON	ADDRTYPE	COLLISIONTYPE	WEATHER	ROADCOND	LIGHTCOND	SPEEDING
0	2	-122.323148	47.703140	Intersection	Angles	5	8	6	0
1	1	-122.347294	47.647172	Block	Sideswipe	7	8	3	0
2	1	-122.334540	47.607871	Block	Parked Car	5	1	6	0
3	1	-122.334803	47.604803	Block	Other	2	1	6	0
4	2	-122.306426	47.545739	Intersection	Angles	7	8	6	0

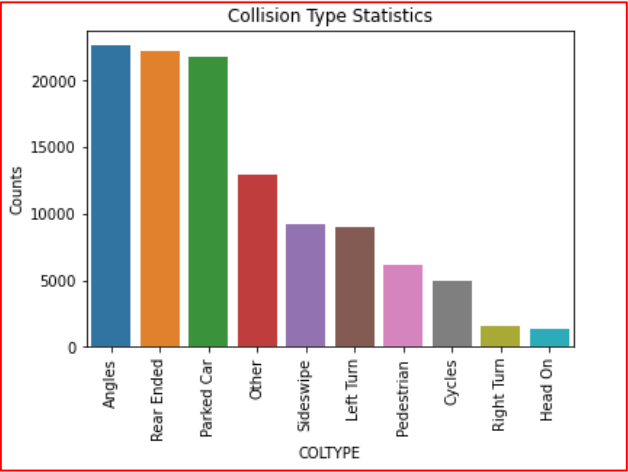
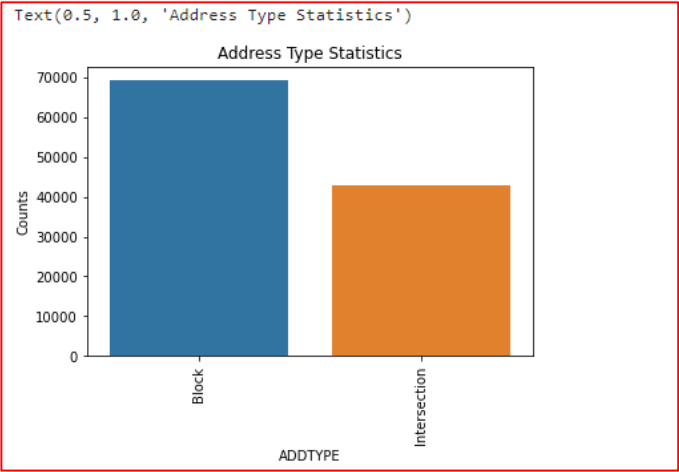
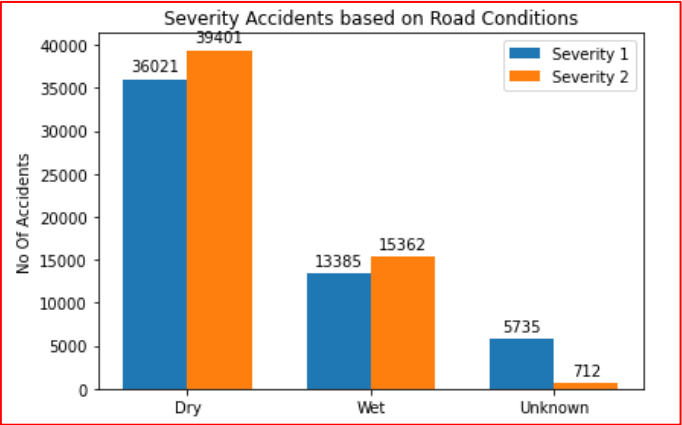
Cleaning Process

```
#df1['WEATHER'].value_counts()
df1["WEATHER"].replace(" ", np.nan, inplace = True)
df1["WEATHER"].replace("0", np.nan, inplace = True)
df1["LIGHTCOND"].replace(" ", np.nan, inplace = True)
df1["LIGHTCOND"].replace("0", np.nan, inplace = True)
df1["ROADCOND"].replace(" ", np.nan, inplace = True)
df1["ROADCOND"].replace("0", np.nan, inplace = True)
df1["ADDRTYPE"].replace(" ", np.nan, inplace = True)
```

```
# Dropping rows where value is NAN
df1.dropna(subset=["WEATHER"], axis=0, inplace=True)
df1.dropna(subset=["ROADCOND"], axis=0, inplace=True)
df1.dropna(subset=["LIGHTCOND"], axis=0, inplace=True)
df1.dropna(subset=["LAT"], axis=0, inplace=True)
```

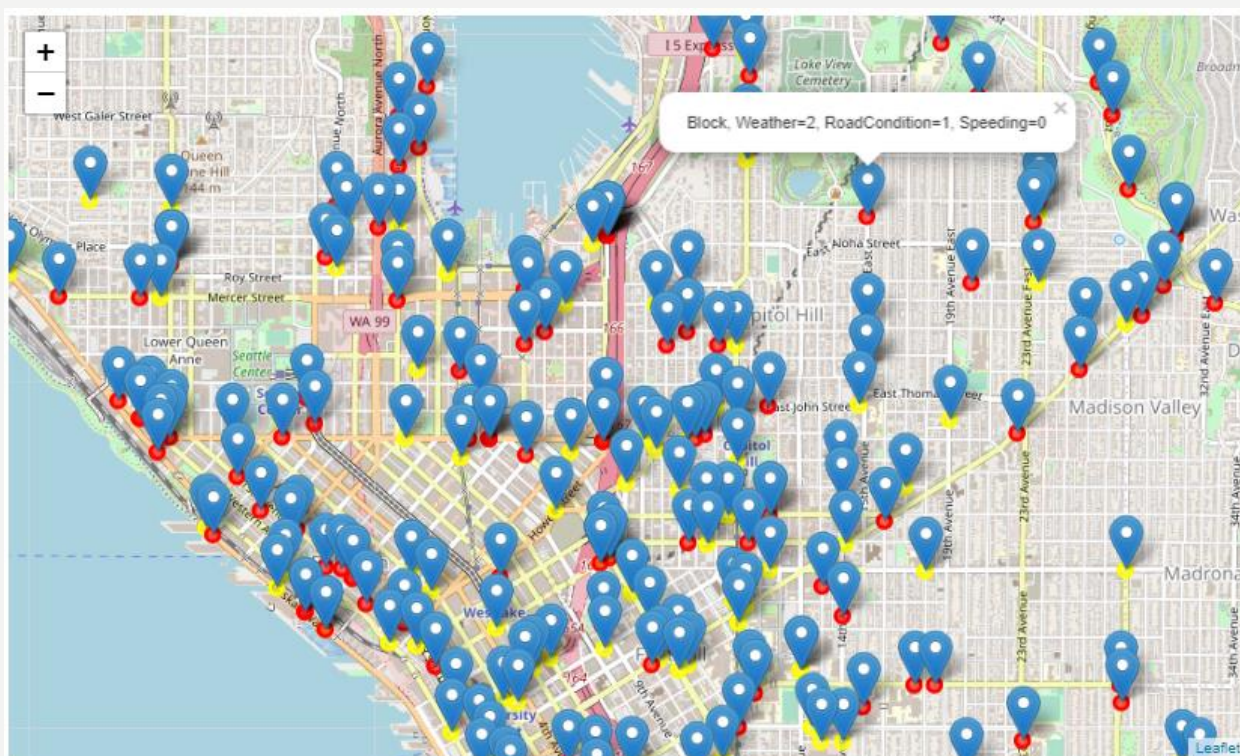
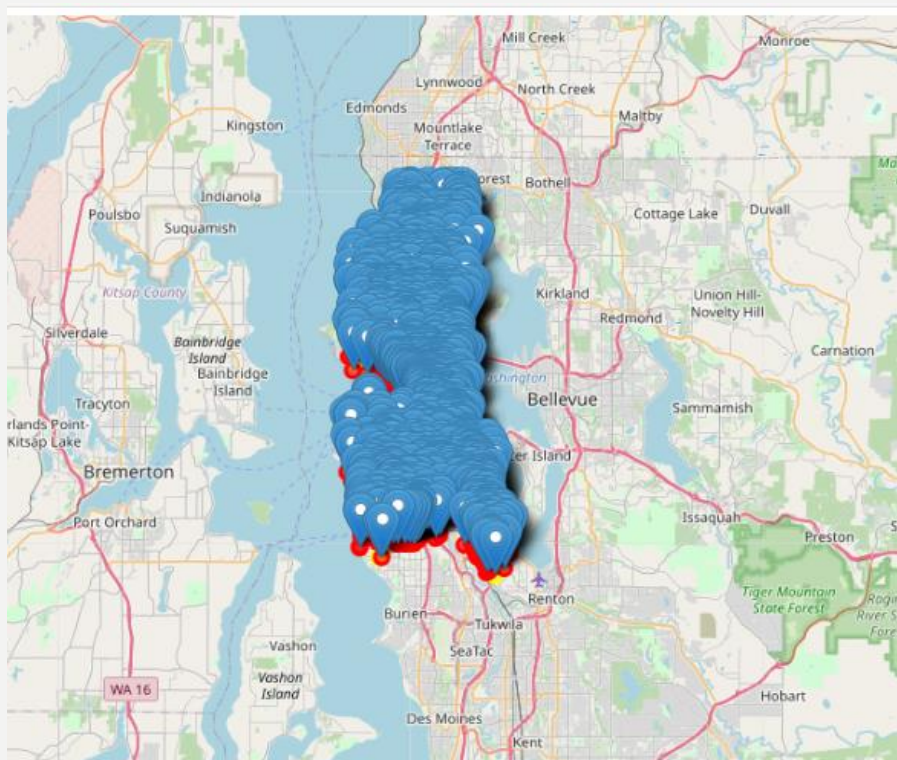

STATISTICAL ANALYSIS

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



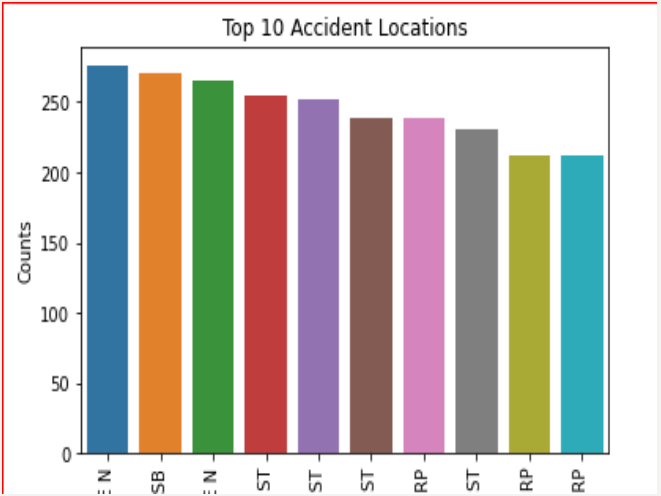
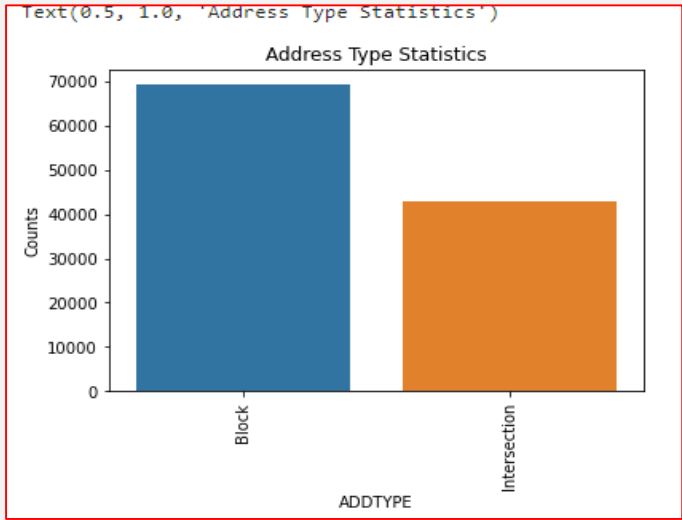
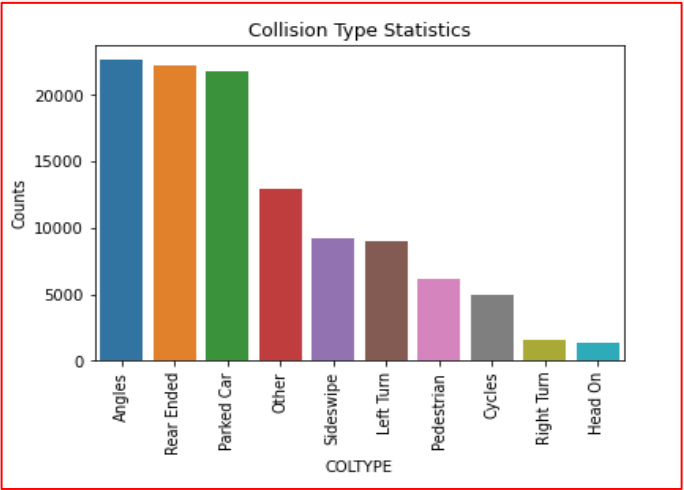
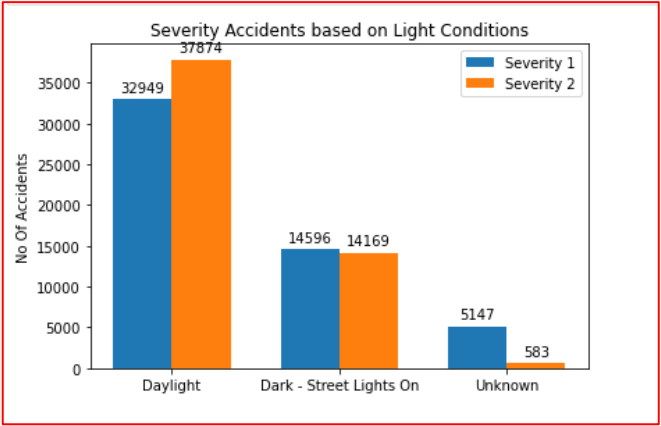
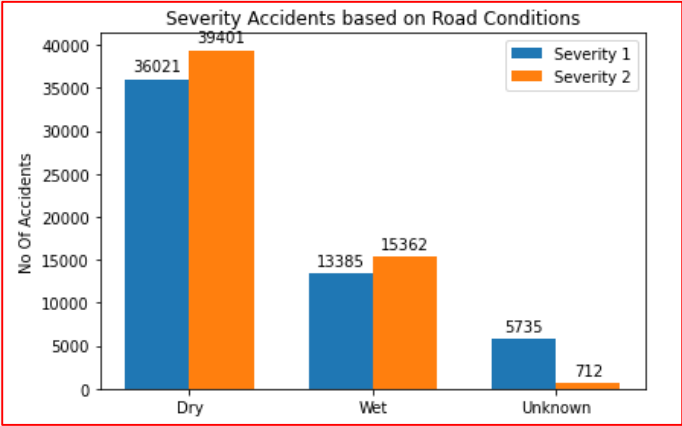
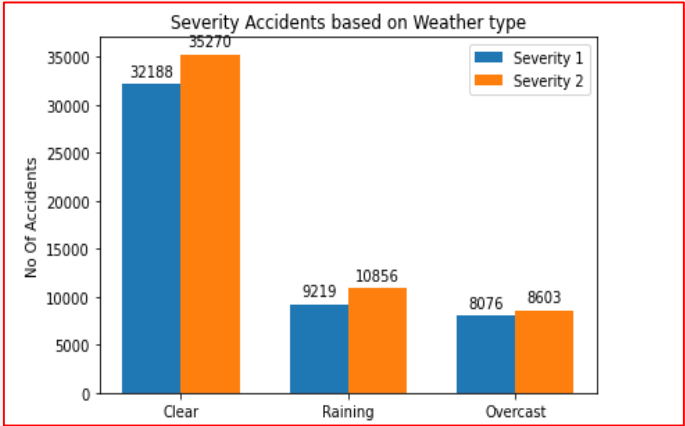
DATA VISUALIZATION

- Accident locations are super imposed on the map with severity code 1 marked in yellow circle and severity code 2 marked in red circle. Click on the popup to view the Address type, Weather, Road condition and speeding status at the location. Zoom in to the map to see the roads to see where more accidents happen.



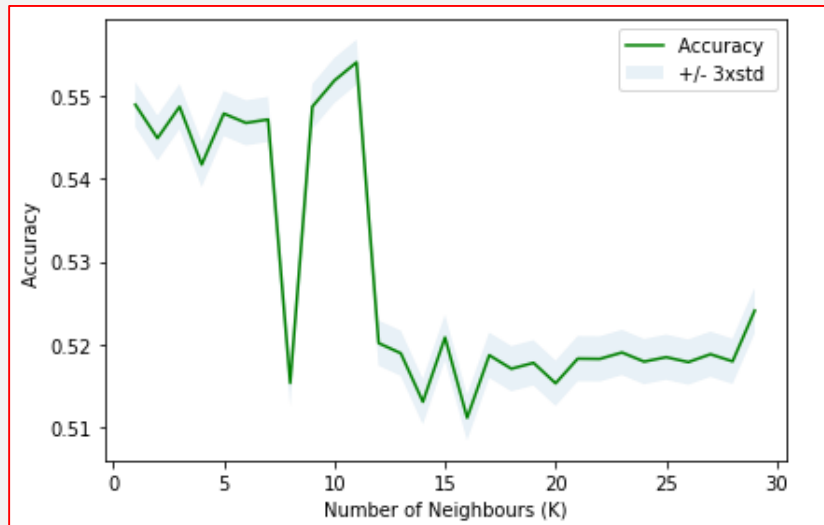
DETAILED STUDY

- Detailed study was done to identify the impact of various attributes of weather, road and light condition in relation with the severity of accident.



MODELLING AND PREDICTIONS

Best value of K calculated. Plotted the graph using matplotlib library



Decision Tree

```
#Train Model & Predict
dtyhat=dtree.predict(X_test)
dtyhat[0:5]

array([2, 2, 2, 2, 1])
```

KNN Model

```
k = 11
neigh = KNeighborsClassifier(n_neighbors = k).fit(X_train,y_train)
kyhat = neigh.predict(X_test)
kyhat[0:5]

array([2, 2, 2, 2, 1])
```

Logistic Regression

```
LRyhat = LR.predict(X_test)
LRyhat

LRyhat_prob = LR.predict_proba(X_test)
LRyhat_prob

array([[0.47702251, 0.52297749],
       [0.47702251, 0.52297749],
       [0.47702251, 0.52297749],
       ...,
       [0.53045002, 0.46954998],
       [0.47702251, 0.52297749],
       [0.40193133, 0.59806867]])
```

RESULTS

Accuracy of our models.

Evaluation metrics used to test the accuracy of models were jaccard index, f-1 score and logloss for logistic regression. Choosing different k, max depth and hyperparameter C values helped to improve our accuracy to be the best possible.

```
print("KNN Jaccard index: %.2f" % jaccard_similarity_score(y_test, kyhat))
print("KNN F1-score: %.2f" % f1_score(y_test, kyhat, average='weighted') )

KNN Jaccard index: 0.55
KNN F1-score: 0.54

print("Decision Tree Jaccard index: %.2f" % jaccard_similarity_score(y_test, dtyhat))
print("Decision Tree F1-score: %.2f" % f1_score(y_test, dtyhat, average='weighted') )

Decision Tree Jaccard index: 0.56
Decision Tree F1-score: 0.53

print("LR Jaccard index: %.2f" % jaccard_similarity_score(y_test, LRyhat))
print("LR F1-score: %.2f" % f1_score(y_test, LRyhat, average='weighted') )
print("LR LogLoss: %.2f" % log_loss(y_test, LRyhat_prob))

LR Jaccard index: 0.55
LR F1-score: 0.53
LR LogLoss: 0.68
```

Algorithm	Jaccard	F1-score	LogLoss
KNN	0.55	0.54	NA
Decision Tree	0.56	0.53	NA
LogisticRegression	0.55	0.53	0.68

DISCUSSION

- The initial data had categorical data that was of type 'object'. This is not a data type that we could have fed through an algorithm, so label encoding was used to create new classes that were of type int; a numerical data type. After solving that issue we were presented with another - imbalanced data. As mentioned earlier, severity 1 was nearly three times larger than severity 2. The solution to this was down sampling the majority severity with sklearn's resample tool. The data was down sampled to match the minority severity class.
- After data wrangling and analysis of the data, it was then fed through three ML models; K-Nearest Neighbor, Decision Tree and Logistic Regression. Although the first two are ideal for this project, logistic regression made most sense because of its binary nature.
- Evaluation metrics used to test the accuracy of our models were jaccard index, f-1 score and logloss for logistic regression. Choosing different k, max depth and hyperparameter C values helped to improve our accuracy to be the best possible.

CONCLUSION

- Based on historical data from weather conditions pointing to certain classes, we can conclude that weather conditions have impact on accident severity. The top location of accidents shown will help drivers to be careful. It also points out that during clear weather a greater number of accidents happen with severity 2 and the cause related is speeding and road conditions. This information is beneficial for people looking out for low cost residential apartments in Dubai.
- Be Safe – Be Careful & Be vigilant. Choose the safest route while driving based on this analysis published. All the best.

Thank You!!!

