Capstone Report

Introduction:

Road safety is one of the most important problems in the world but is often not considered by people going about their daily lives. Vehicular accidents are one of the most common causes for fatalities and injuries, being caused by distracted driving, driving under the influence or even seemingly benign factors such as weather, visibility or road conditions.

This project aims to build a model that can predict the severity of accidents based on information such as its location, the weather, road conditions and light conditions. The target audience of this project is the local Seattle government, police, and city planners. The model and its results could help provide advice for making decisions in reducing the number of accidents and injuries in the city.

Data:

The data used for this study is given by the Applied Data Science Capstone course on Coursera.org via the following
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. The initial dataset consists of 38 columns and 194673 rows. Python packages will be used to conduct this study. The dataset will be cleaned according to the requirements of this project. Missing data information will either be substituted using valid means or dropped – considering the amount of missing data and the description of individual elements.

Firstly, we want to drop the columns that aren't relevant to our data

In [4]:
```
# We only want the following columns in our data SEVERITYCODE, SEVERITYDESC, WEATHER, ROADCOND, LIGHTCOND, WEATHER_CAT, ROADCOND_CAT, LIGHTCOND_CAT

df.drop(df.columns.difference(['SEVERITYCODE','SEVERITYDESC', 'ADDRTYPE', 'JUNCTIONTYPE', 'SDOT_COLDESC', 'WEATHER', 'LIGHTCOND', 'ROADCOND_CAT']), axis=1, inplace=True)
df.head()
```

Out[4]:

	SEVERITYCODE	ADDRTYPE	SEVERITYDESC	JUNCTIONTYPE	SDOT_COLDESC	WEATHER	LIGHTCOND
0	2	Intersection	Injury Collision	At Intersection (intersection related)	MOTOR VEHICLE STRUCK MOTOR VEHICLE, FRONT END ...	Overcast	Daylight
1	1	Block	Property Damage Only Collision	Mid-Block (not related to intersection)	MOTOR VEHICLE STRUCK MOTOR VEHICLE, LEFT SIDE ...	Raining	Dark - Street Lights On
2	1	Block	Property Damage Only Collision	Mid-Block (not related to intersection)	MOTOR VEHICLE STRUCK MOTOR VEHICLE, REAR END	Overcast	Daylight
3	1	Block	Property Damage Only Collision	Mid-Block (not related to intersection)	MOTOR VEHICLE STRUCK MOTOR VEHICLE, FRONT END ...	Clear	Daylight
4	2	Intersection	Injury Collision	At Intersection (intersection related)	MOTOR VEHICLE STRUCK MOTOR VEHICLE, FRONT END ...	Raining	Daylight

And due to the relatively low amount of rows with null values, I elected to just drop those rows

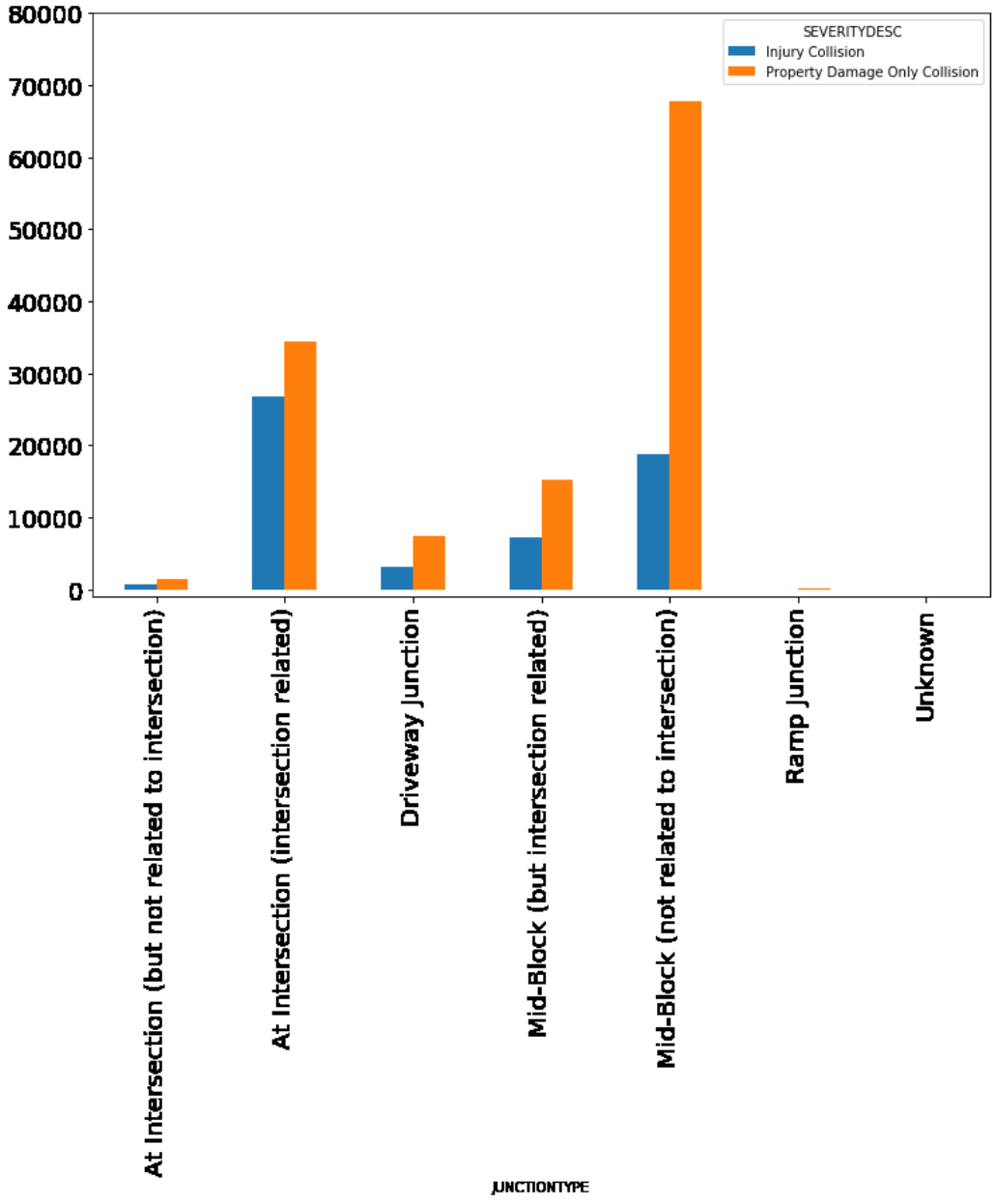
```
In [6]: df['SEVERITYCODE'].value_counts()

Out[6]: 1    136485
        2     58188
        Name: SEVERITYCODE, dtype: int64
```

Since there isn't that many null data, I'll just remove the rows with null values.

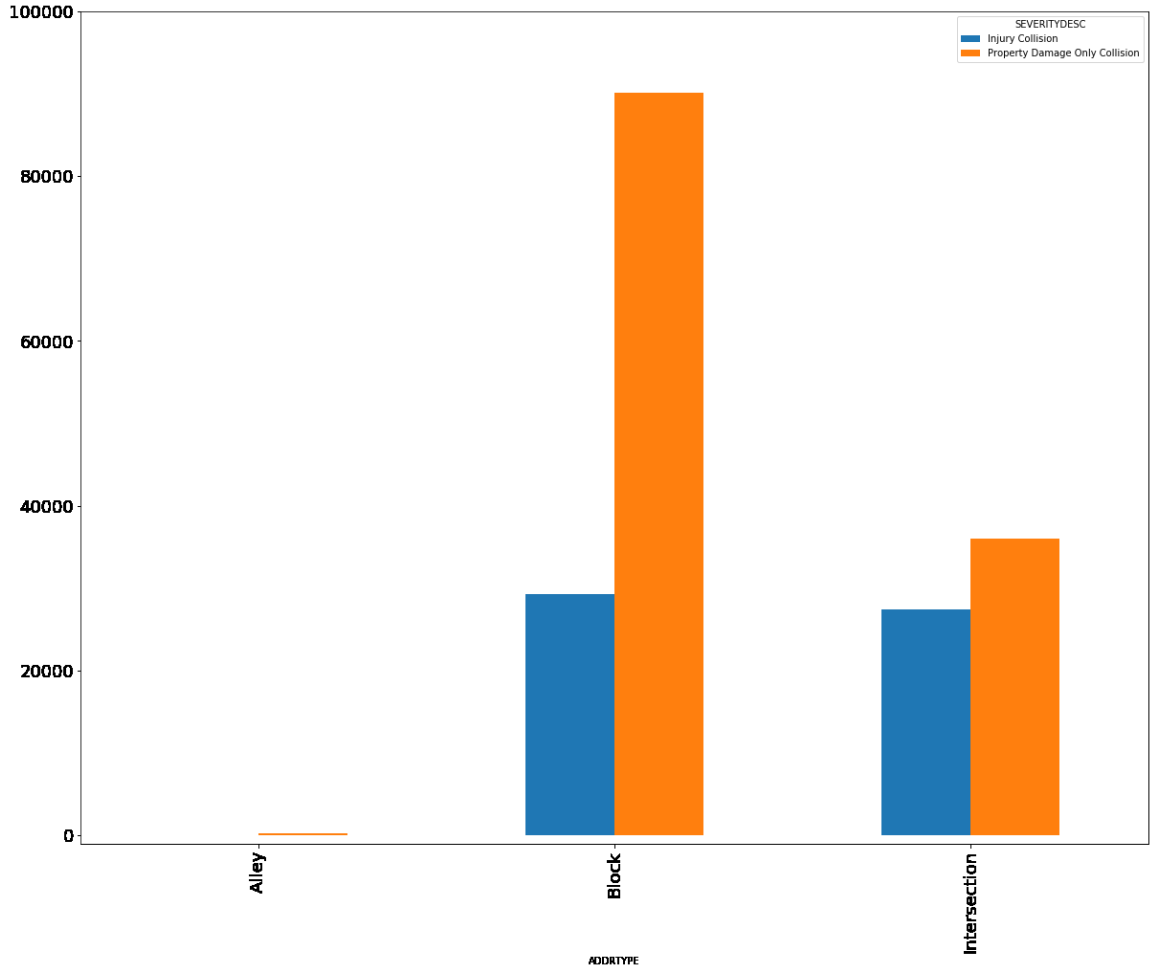
```
In [7]: df = df.dropna()
        df.shape

Out[7]: (182954, 7)

In [8]: # we will resit index to correct rows numbers
        df=df.reset_index(drop=True)
        df.isnull().sum()

Out[8]: SEVERITYCODE    0
        ADDRTYPE        0
        SEVERITYDESC    0
        JUNCTIONTYPE    0
        SDOT_COLDESC    0
        WEATHER         0
        LIGHTCOND       0
        dtype: int64
```

Methodology:

Firstly I did some data visualization by creating a graph that displays the Junction type with the Severity Description



And the address type with the severity description



I then converted the data from text description types to just category codes.

```
In [11]: df['ADDRTYPE']=df['ADDRTYPE'].astype('category').cat.codes
         df['JUNCTIONTYPE']=df['JUNCTIONTYPE'].astype('category').cat.codes
         df['SDOT_COLDESC']=df['SDOT_COLDESC'].astype('category').cat.codes
         df['WEATHER']=df['WEATHER'].astype('category').cat.codes
         df['LIGHTCOND']=df['LIGHTCOND'].astype('category').cat.codes
```

```
In [12]: Feature = df['SEVERITYDESC']
         Feature=pd.concat([Feature, pd.get_dummies(df[['ADDRTYPE','JUNCTIONTYPE','SDOT_COLDESC','WEATHER','LIGHTCOND']])], axis=1)
         Feature.drop(['SEVERITYDESC'], axis=1, inplace=True)
         Feature
```

Out[12]:

	ADDRTYPE	JUNCTIONTYPE	SDOT_COLDESC	WEATHER	LIGHTCOND
0	2	1	14	4	5
1	1	4	16	6	2
2	1	4	17	4	5
3	1	4	14	1	5
4	2	1	14	6	5
5	2	1	14	1	5
6	2	1	14	6	5
7	2	1	29	1	5
8	1	4	14	1	5

For implementing the model I used a Jupyter notebook using Python and some libraries such as Pandas, Numpy and SKLearn to process the data and to build the machine learning model.

A decision tree model gives us a layout of all possible outcomes so we can fully analyze the consequences of a decision. In context, the decision tree observes all possible outcomes of different weather conditions.

I employed the machine learning model of Decision Trees due to the multitude of factors that contribute to a severity, I feel that a decision tree would be the best.

Decision tree model

```
In [15]: from sklearn.tree import DecisionTreeClassifier
         from sklearn import preprocessing
```

```
In [16]: DT_model = DecisionTreeClassifier(criterion='entropy', max_depth=5)
         DT_model.fit(X_train, y_train)
```

```
Out[16]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=5,
                                max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                                splitter='best')
```

```
In [17]: DT_predict = DT_model.predict(X_test)
         print(DT_predict[0:5])
         print(y_test [0:5])
```

```
['Property Damage Only Collision' 'Property Damage Only Collision'
 'Property Damage Only Collision' 'Property Damage Only Collision'
 'Property Damage Only Collision']
['Property Damage Only Collision' 'Injury Collision'
 'Property Damage Only Collision' 'Property Damage Only Collision'
 'Injury Collision']
```

Results

With the Decision tree, after splitting the data into training and test data it seems that it's able to correctly predict the correct severity ~70% of the time. And has a Jaccard and F1-Score of 0.739991 and 0.673699 respectively.

```
In [18]: from sklearn import metrics
         print("Decision Tress Accuracy:", metrics.accuracy_score(y_test, DT_predict))
```

```
Decision Tress Accuracy: 0.7399907080976197
```

Evaluation

In [19]:
```
from sklearn.metrics import jaccard_similarity_score
from sklearn.metrics import f1_score
```

In [20]:
```
jc=jaccard_similarity_score(y_test, DT_predict)
fs=f1_score(y_test, DT_predict, average='weighted')
```

In [21]:
```
list_jc = [jc]
list_fs = [fs]

df = pd.DataFrame(list_jc, index=['Decision Tree'])

df.columns = ['Jaccard']
df.insert(loc=1, column='F1-score', value=list_fs)

df
```

Out[21]:

	Jaccard	F1-score
Decision Tree	0.739991	0.673699

Discussion

Conclusion

Based on the dataset and the decision tree created for this project, we can conclude that address types, junction types, weather, lighting conditions and road conditions have a significant impact on the severity of car crashes in the city of Seattle.