# Applied Capstone Project

Model and prediction of severity of road accidents in seattle

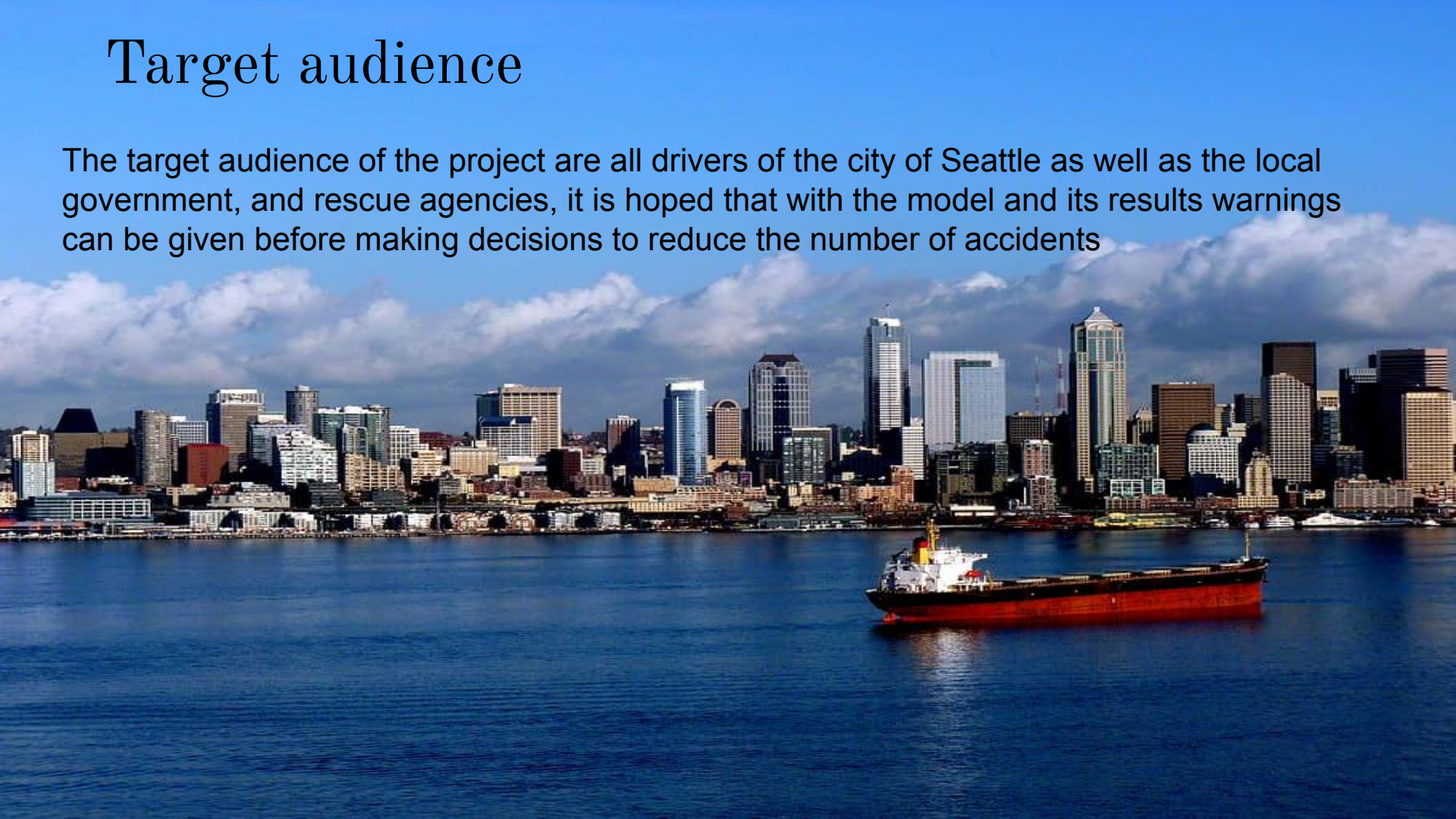
# Introduction and Business Understanding

"According to the World Health Organization, the United States ranks 41st among the 52 highest-income nations in terms of traffic fatalities, surpassed only by Qatar, Saudi Arabia and Russia. It is also "now the most dangerous rich country in which a child can be born," concluded a new report published by Health Affairs magazine, based in part on the high risk of a young person dying after a car accident. . Among American teens between the ages of 15 and 19, car accidents rank as the leading cause of death.

It didn't use to be this way: Before the 1980s, the United States was at the forefront of driving safety, outperforming several wealthy countries. Today, the traffic accident fatality rate in the US, per 100,000 inhabitants, is almost double that of Canada and Australia, and is almost three and a half times that of the United Kingdom. Slovenia, Italy and Greece are also doing better" Bliss,L(2018,13 March) The highway crisis in the US: high mortality and poor safety Retrieved from https://www.univision.com/noticias/citylab-transporte/la-crisis-de-las-autopistas-en-eeuu-alta-mortalidad-y-escasa-seguridad

# Target audience

The target audience of the project are all drivers of the city of Seattle as well as the local government, and rescue agencies, it is hoped that with the model and its results warnings can be given before making decisions to reduce the number of accidents



# Data Preprocessing

The data in the initial form is not ready for analysis. For this reason, first, we are going to remove the non-relevant columns. In addition to this, many of the features are from object data types that must be converted to numeric data types.

In analyzing the data set, it has been decided to work with only four characteristics: severity, weather conditions, road conditions and light conditions, "For now ".

In order to understand the data set well, different values have been verified in the characteristics.The results show, the target feature is imbalance, so we use a simple statistical technique to balance it

```
In [1]: import csv

In [2]: import numpy as np
        import pandas as pd #primary data structure library
        from __future__ import print_function #adds compatibility to python 2

In [79]: # Read the csv file from other path (pc)
         df = pd.read_csv('C:/Users/ASUS/Desktop/Python/Data-Collisions.csv',sep=','
                          )

         print ('Data read into a pandas dataframe!')

         Data read into a pandas dataframe!

         C:\Users\ASUS\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:2717: DtypeWarning: Columns (33) have mixed types. S
         pecify dtype option on import or set low_memory=False.
           interactivity=interactivity, compiler=compiler, result=result)

In [80]: df
```

Out[80]:

| | SEVERITYCODE | X | Y | OBJECTID | INCKEY | COLDETKEY | REPORTNO | STATUS | ADDRTYPE | INTKEY | ... | ROADCOND | LIGHTCO |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | -122.323148 | 47.703140 | 1 | 1307 | 1307 | 3502005 | Matched | Intersection | 37475.0 | ... | Wet | Day |
| 1 | 1 | -122.347294 | 47.647172 | 2 | 52200 | 52200 | 2607959 | Matched | Block | NaN | ... | Wet | Dark - St Lights |

# Data Preprocessing

The dataset in the original form is not ready for data analysis. In order to prepare the data, first, we need to drop the non-relevant columns. In addition, most of the features are of object data types that need to be converted into numerical data types.

After analyzing the data set, I have decided to focus on only four features, severity, weather conditions, road conditions, and light conditions, among others.

To get a good understanding of the dataset, I have checked different values in the features. The results show, the target feature is imbalance, so we use a simple statistical technique to balance it.

```
In [5]: df["SEVERITYCODE"].value_counts()
```

```
Out[5]: 1    136485
        2     58188
        Name: SEVERITYCODE, dtype: int64
```

As you can see, the number of rows in class 1 is almost three times big the class 1.

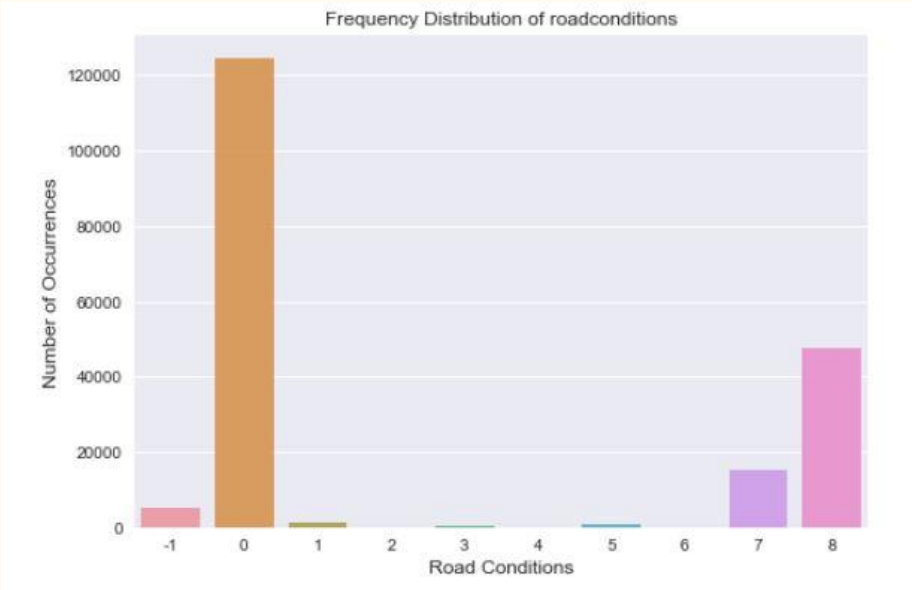
```
In [6]: from sklearn.utils import resample
```

```
In [7]: df_maj=df[df.SEVERITYCODE==1]
        df_min=df[df.SEVERITYCODE==2]
        df_maj_dsample=resample(df_maj, replace=False,
                                n_samples=58188,
                                random_state=123)
        balanced_df=pd.concat([df_maj_dsample,df_min])
        balanced_df.SEVERITYCODE.value_counts()
```

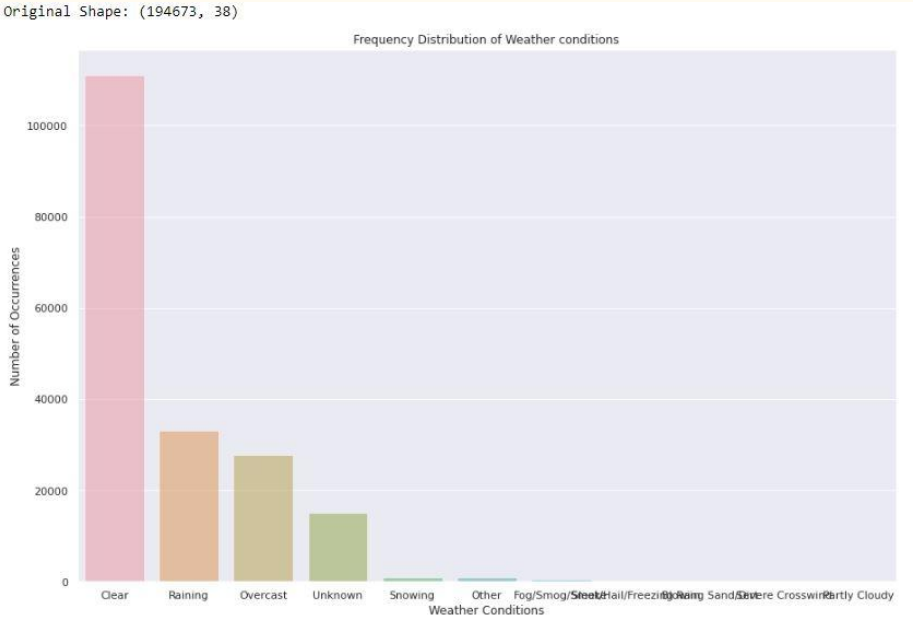
```
Out[7]: 2    58188
        1    58188
        Name: SEVERITYCODE, dtype: int64
```

```
In [8]: y=np.asarray(balanced_df["SEVERITYCODE"])
        y[0:5]
```

```
Out[8]: array([1, 1, 1, 1, 1], dtype=int64)
```

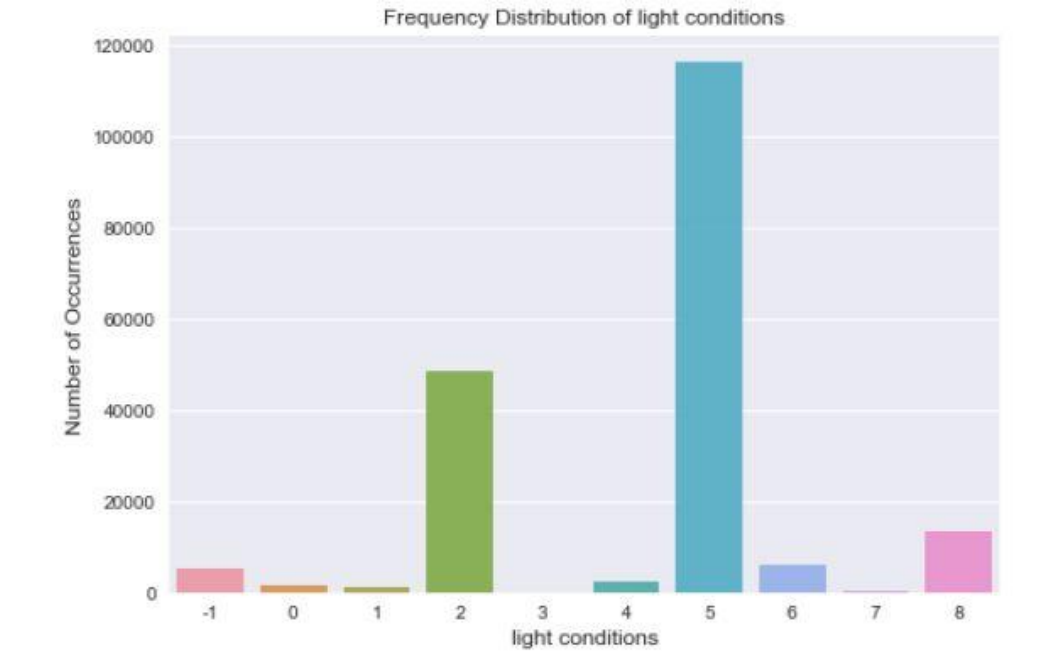



```
Original Shape: (194673, 38)
```




0- Dry
8 – Wet
7 - Unknown

| | | | |
|---|---|---|---|
| 0 | Blowing Sand/Dirt | 6 | Raining |
| 1 | Clear | 7 | Severe Crosswind |
| 2 | Fog/Smog/Smoke | 8 | Sleet/Hail/Freezing Rain |
| 3 | Other | 9 | Snowing |
| 4 | Overcast | 10 | Unknown |
| 5 | Partly Cloudy | | |




:

```
%matplotlib inline
import seaborn as sns
import matplotlib.pyplot as plt

roadcond_count = df['LIGHTCOND'].value_counts()
sns.set(style="darkgrid")
sns.barplot(roadcond_count.index, roadcond_count.values, alpha=0.9)
plt.title('Frequency Distribution of light conditions')
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('light conditions', fontsize=12)
plt.show()
```

# Methodology

## 1) Train/Test Split

```
from sklearn.model_selection import train_test_split
```

```
# split a dataset into train and test sets
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
# create dataset
X, y = make_blobs(n_samples=116376)
# split into train test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,random_state=1)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(81463, 2) (34913, 2) (81463,) (34913,)
```

```
#X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.3,random_state=1)
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.3, random_state=1)
print("Train set:",X_train.shape, y_train.shape)
print("Test set:",X_test.shape, y_test.shape)
```

```
Train set: (81463, 2) (81463,)
Test set: (34913, 2) (34913,)
```

# 2)Machine learning models:

After balancing SEVERITYCODE feature, and standardizing the input feature, the data has been ready for building machine learning models.
I have employed three machine learning models:

## *K Nearest Neighbour (KNN)*

## *Decision Tree*

## *Linear Regression*

After importing necessary packages and splitting preprocessed data into test and train sets, for each machine learning model, I have built and evaluated the model and shown the results as follow:

# KKN Evaluation

Method 1

[34]:
```
import numpy as np
from sklearn.metrics import jaccard_similarity_score
```

[35]:
```
#jaccard similarity score
jaccard_similarity_score(y_test,knn_y_pred)
```

[35]: 0.99607596024403522

[36]:
```
from sklearn.metrics import f1_score
```

[37]:
```
#f1 SCORE
f1_score(y_test,knn_y_pred,average="macro")
```

[37]: 0.99610437073534663

# Decision tree

40]:
```
#Building the Decision Tree
from sklearn.tree import DecisionTreeClassifier
colDataTree= DecisionTreeClassifier(criterion="entropy",max_depth=7)
colDataTree
colDataTree.fit(X_train,y_train)
```

40]:
```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=7,
            max_features=None, max_leaf_nodes=None,
            min_impurity_split=1e-07, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='best')
```

41]:
```
#Train Model and Predict
DTyhat=colDataTree.predict(X_test)
print(DTyhat[0:5])
print(y_test[0:5])
```

```
[2 0 0 0 1]
[2 0 0 0 1]
```

# Logistic Regression

48]:
```
#Building the LR Model
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
LR=LogisticRegression(C=6,solver="liblinear").fit(X_train,y_train)
LR
```

48]:
```
LogisticRegression(C=6, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

49]:
```
#Train Model and predicr
LRyhat=LR.predict(X_test)
LRyhat
```

49]:
```
array([2, 0, 0, ..., 0, 1, 2])
```

50]:
```
yhat_prob=LR.predict_proba(X_test)
yhat_prob
```

50]:
```
array([[  2.84077420e-28,   1.53100757e-01,   8.46899243e-01],
```

# Results and Evaluations

```
import pandas as pd
data = [['KNN',0.999943,0.99994271474808816],['Decision Tree',0.999914,0.99991
df = pd.DataFrame(data,columns=['Mode','F1 Score','jaccard similarity score'])
print(df)
```

```
                Mode  F1 Score  jaccard similarity score
0                KNN  0.999943                  0.999943
1      Decision Tree  0.999914                  0.999914
2  Linear Regresion   0.999770                  0.999771
```

Based on the above table, KNN is the best model to predict car accident severity.

# Conclusion

Based on the data set provided for this project which are weather, road and light conditions, we can conclude that the particular conditions have a certain impact on whether the trip could result in property damage.

Thanks to all the readers.