1. Introduction/Business Problem

In the capstone project for IBM certificate course, the aim is to analyse the accident "severity" whether in terms of fatality or traffic or accidents or any other type of bad impact. All records were provided by SPD and recorded by Traffic Records and provided to me by Coursera. This dataset covers a period of 2004 to present. It contains information such as severity code, address type, weather, road condition, speeding, etc.

The audience will be road users who will benefit from information that will possibly help reduce traffic collision/accidents in the future and also assist with travel plans. This model can alert/assist road users know when the road is safe or bad so they can plan accordingly

2. Data Understanding

There are 194,673 observations and 38 attributes in the dataset. The aim is to identify the severity of an accident and the possible factors that can lead to one. From the dataset, the SEVERITYCODE attribute will be used as the target variable (Y). The two types of severity are 1-prop damage and 2-injury Feature Engineering process will be applied to ascertain the attributes that impact the severity of an accident. Missing value will either be replaced or removed from the dataset as well.

The columns that I will be using for the analysis are as follows:

- COLLISIONTYPE, which describes the type of crash,
- WEATHER, description of the weather conditions during the time of the collision
- ROADCOND, condition of the road during the collision.
- LIGHTCOND, light conditions during the collision,
- INATTENTIONIND, whether or not collision was due to inattention. (Y/N)
- UNDERINFL, whether or not a driver involved was under the influence of drugs or alcohol.
- SEVERITYCODE, corresponds to the severity of the collision and assigns a crash a value of 1, which means property damage collison, and 2, indicating injury

3. Methodology

The objective is to predict the severity of a traffic accident based on certain factors. The python libraries that will be used are for both data visualization and manipulation such as PANDAS, SEABORN, MATPLOTLIB. This will precede the use of SCIKIT learn libraries and machine learning algorithms.

Data analysis will be carried out to ascertain the best algorithms to be used to predict and determine what attributes in the dataset are relevant to achieve this.

Libraries imported:

```
#conda install -c conda-forge imbalanced-learn
```

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from imblearn.under_sampling import RandomUnderSampler
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import jaccard_similarity_score
from sklearn.metrics import f1_score
```

Importing data to be used:

```
collision = pd.read_csv("Data-Collisions.csv")
collision.head()
```

Dropping attributes that will not be used:

```
collision.drop(['X','Y','INTKEY','EXCEPTRSNCODE','EXCEPTRSNDESC','PEDROWNOTGRNT',
                'SDOTCOLNUM','SPEEDING','INCKEY','COLDETKEY','INCDATE','INCDTTM','SDOTCOLNUM',
               'ST_COLCODE','ST_COLDESC','OBJECTID','STATUS','REPORTNO','LOCATION','SEVERITYDESC'
               ,'JUNCTIONTYPE','SDOT_COLDESC'],inplace=True, axis=1)
```

Converting text attributes to categorical data using one-hot encoding:

```
collision = pd.get_dummies(collision, columns=['WEATHER','COLLISIONTYPE','LIGHTCOND','ROADCOND','UNDERINFL',
                                               'INATTENTIONIND','ADDRTYPE','HITPARKEDCAR'])
```

Training the model and using Logistic Regression and K-Nearest Neighbor algorithms:

Logistic Regression: Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable

K-Nearest Neighbor: K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (based on distance)

```
X_train, X_test, y_train, y_test = train_test_split(X_rus,y_rus, test_size=0.33, random_state=42)
```

```
#Using Logistic Regression Classifier
lr = LogisticRegression()
lr.fit(X_train,y_train)
pred_lr = lr.predict(X_test)
print(classification_report(y_test,pred_lr))
print(confusion_matrix(y_test,pred_lr))
print(jaccard_similarity_score(y_test, pred_lr))
print ("training accuracy: {:.2f}".format(lr.score(X_train,y_train)))
print ("test accuracy: {:.2f}".format(lr.score(X_test,y_test)))
```

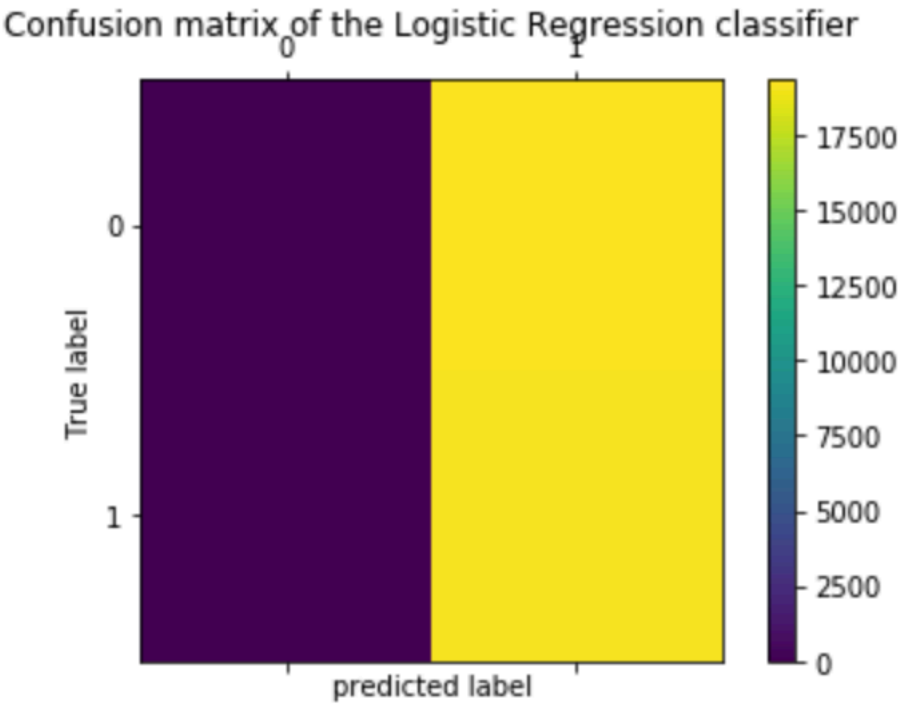
```
#Using K-Nearest Neighbor
from sklearn.model_selection import cross_val_score

#import the train and test split function
cross_val_scores = []

neighbors = np.arange(1,30,2)
for i in neighbors:
    knn = KNeighborsClassifier(n_neighbors=i)
    #perform cross-validation with a given model
    scores = cross_val_score(knn, X_train, y_train, cv=10)
    cross_val_scores.append(np.mean(scores))

print("best cross-validation score: {:.3f}".format(np.max(cross_val_scores)))
best_n_neighbors = neighbors[np.argmax(cross_val_scores)]
print("best n_neighbors           : {}".format(best_n_neighbors))

knn = KNeighborsClassifier(n_neighbors=best_n_neighbors)
knn.fit(X_train, y_train)
print("test-set score             : {:.3f}".format(knn.score(X_test, y_test)))
```

4. Results

The two models had varying results with one of them working well to predict both negatives and positives.

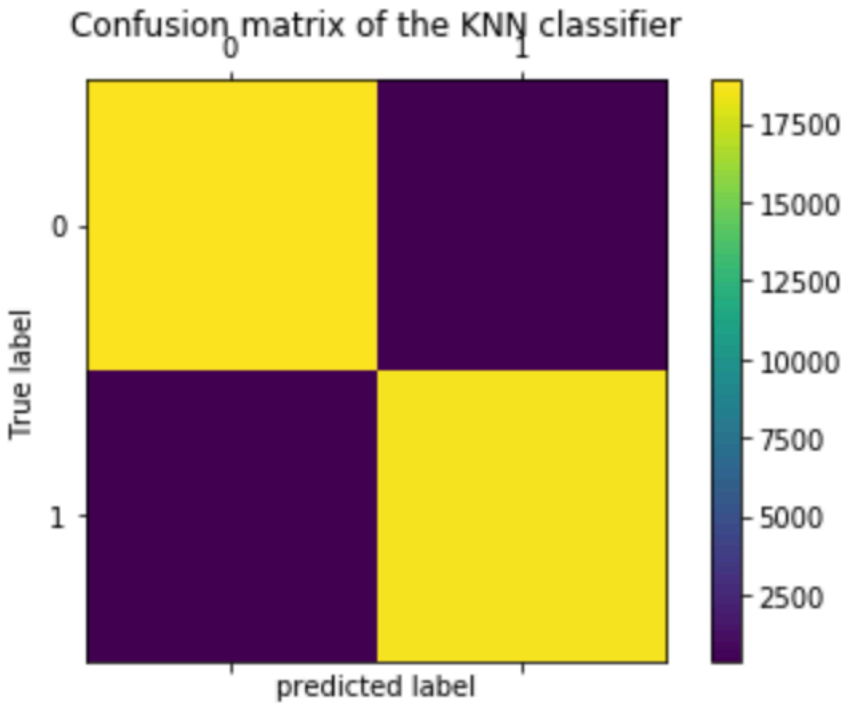
Logistic Regression using the default parameters:

	Precision	Recall	F1-score
1	1.00	0.00	0.00
2	0.50	1.00	0.66
Accuracy	0.50		
Micro Average	0.50	0.50	0.50
Macro Average	0.75	0.50	0.33
Weighted Average	0.75	0.50	0.33



K-Nearest Neighbors using best n-neighbors of 11:

	Precision	Recall	F1-score
1	0.98	0.98	0.98
2	0.98	0.98	0.98
Accuracy	0.98		
Micro Average	0.98	0.98	0.98
Macro Average	0.98	0.98	0.98
Weighted Average	0.98	0.98	0.98



5. Discussion

Three key metrices were used to ascertain the best result:

- **Precision:** Precision refers to the percentage of results which are relevant, in simpler terms it can be seen as how many of the selected items from the model are relevant. Mathematically, it is calculated by dividing true positives by true positive and false positive
- **Recall:** Recall refers to the percentage of total relevant results correctly classified by the algorithm. In simpler terms, it tells how many relevant items were selected. It is calculated by dividing true positives by true positive and false negative
- **F1-Score:** f1-score is a measure of accuracy of the model, which is the harmonic mean of the model's precision and recall. Perfect precision and recall is shown by the f1-score as 1, which is the highest value for the f1-score, whereas the lowest possible value is 0 which means that either precision or recall is 0

Algorithm	Average F1-score	Property damage (1) vs Injury (2)	Precision	Recall
Logistic Regression	0.33	1	1.00	0.00
		2	0.50	1.00
K-Nearest Neighbors (KNN)	0.98	1	0.98	0.98
		2	0.98	0.98

From the above, KNN performed best in predicting car accident severity. The Recall and Precision are best from the KNN model as it favours all the resources that will be necessary to determine the severity. The accuracy score of 0.98 from the KNN model is way higher than the 0.50 from the Logistic Regression.

6. Conclusion

In this project, I analysed the severity of an accident and some of the characteristics surrounding the situation involving the accident. It can be concluded that particular conditions have a somewhat impact on whether road users can travel or not and the possible dangers on the road depending on the conditions such as weather, road condition, light conditions and even if the driver is drunk or not. I built two models and compared their results to determine if a road accident will have a severity 1 (property damage) or 2 (injury). This model can be used by Rescue agencies like the fire department to predict the type of accident that has happened before getting to the scene of the accident. It can also be used by the Seattle department to plan road works to prevent accident depending on the locations that meet certain conditions.

Drivers will also benefit as they can know when to take the necessary precautions giving the circumstances of light condition, road condition and even the weather, in order to avoid an accident.