CAR ACCIDENT SEVERITY REPORT

1. INTRIDUCTION

Car accidents can post economical and social harm to the society. Accidents can be caused by different factors, including weather, road conditions, light conditions, not paying enough attention during driving, driving at high speed, and under influence of drugs or alcohol. In order to learn and reduce the frequency of car accidents, prediction of the severity of the accidents must be performed under different conditions.

The target audience of this project is local Seattle government, police, and car drivers. The model and its results are going to predict the severity of an accident given some factors. Also, it can give some suggestions to the target audiences about how to reduce the number of accidents.

2. DATA

The data used for this project was collected based on the car accidents which have taken place in the city of Seattle, Washington from the year 2004 to 2020. Our target variable will be 'SEVERITYCODE', which contains numbers that correspond to different levels of severity caused by an accident from 0 to 4.

Severity codes are as follows:

- 0: Little to no Probability (Clear Conditions)
- 1: Very Low Probability — Chance or Property Damage
- 2: Low Probability — Chance of Injury
- 3: Mild Probability — Chance of Serious Injury
- 4: High Probability — Chance of Fatality

The target variable SEVERITYCODE is imbalance. The SEVERITYCODE in class 1 is almost three times bigger than that in class 2.

```
In [9]: accident_df["SEVERITYCODE"].value_counts()

Out[9]: 1    136485
        2     58188
        Name: SEVERITYCODE, dtype: int64
```

This issue can be solved by downsampling the class 1. After downsampling the class 1, the target variable becomes balance.

```
balanced_df["SEVERITYCODE"].value_counts()

Out[12]: 2    58188
         1    58188
         Name: SEVERITYCODE, dtype: int64
```

The original dataset is not ready for analysis. There are many columns are non-relevant and we will need to drop these columns first. Since for this project, 'INATTENTIONIND', 'UNDERINFL', 'WEATHER', 'ROADCOND', 'LIGHTCOND', and 'SPEEDING' were selected to be the features along with the target variable 'SEVERITYCODE', these will be the useful columns and all other columns need to be dropped.

- INATTENTIONIND: whether or not collision was due to inattention.
- UNDERINFL: whether or not a driver involved was under the influence of drugs or alcohol.
- WEATHER: weather conditions during the time of collision.
- ROADCOND: the conditions of road during collision.
- LIGHTCOND: the light conditions during the collision.
- SPEEDING: whether or not speeding was a factor in the collision.

Also, most of the features are of object data types, which need to be converted into numerical data types. This can be done by encoding numerical data to different situation in each feature. For example, value of 1 was assigned to "Y" and value of 0 was assigned to "N" for feature INATTENTIONIND, UNDERINFL, and SPEEDING. For feature WEATHER, value of 0 was assigned to clear, value of 1 was assigned to raining or snowing, value of 2 was assigned to overcast or cloudy, value of 3 was assigned to wind related conditions, and value of 4 was assigned to unknown or other conditions. For feature LIGHTCOND, value of 0 was assigned to daylight, value of 1 was assigned to dark, value of 2 was assigned to medium light, and value of 3 was assigned to unknown or other conditions. . For feature ROADCOND, value of 0 was assigned to wet roads, value of 1 was assigned to dry, value of 2 was assigned to muddy conditions, and value of 3 was assigned to unknown or other conditions.

The dataset also contains many missing values. To solve this issue, the missing values was replaced by the values that appear the most often for each feature.

3. METHODOLOGY

After the data processing step, the dataset was ready to be fit into machine learning models. The machine learning models used were logistic regression, decision tree, k-Nearest neighbor.

Before fitting the data into the model, data needed to be normalized, Also, the data was split the data into train data and test data with 70% used for training and 30% used for testing.

A. Logistic Regression

Logistic regression is used to predict a binary variable. The logistic regression model was built as follow:

```
#build LR model
LR = LogisticRegression(C=0.01, solver='liblinear').fit(X_train,y_train)
```

B. Decision Tree

Decision tree breaks down the dataset into smaller subsets, it classifies the data by sorting it them down the tree from toot to leaf node, with the leaf node provides the classification to the data. The decision tree model is built with the criterion chosen to be “entropy” and maximum depth of the tree to be 4.

```
#build Decision Tree
Tree = DecisionTreeClassifier(criterion="entropy", max_depth = 4)
Tree.fit(X_train,y_train)
```

C. k-Nearest Neighbor (kNN)

kNN is a method for classifying cases based on their similarity to other cases. To decide the value for k to be used for building kNN model, an iterative process was done to calculate the accuracy of prediction using different k values.

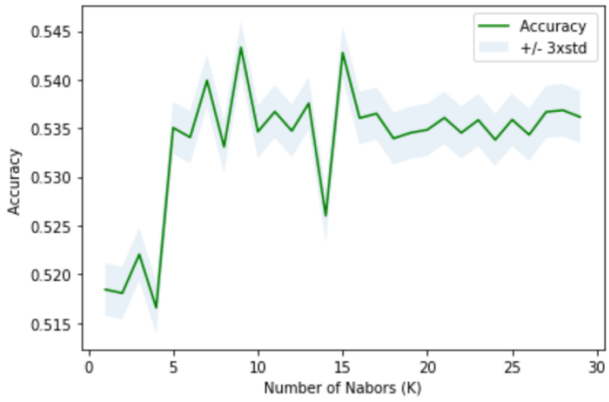
```
#Build KNN model
Ks = 30
mean_acc = np.zeros((Ks-1))
std_acc = np.zeros((Ks-1))
ConfustionMx = [];
for n in range(1,Ks):

    #Train Model and Predict
    neigh = KNeighborsClassifier(n_neighbors = n).fit(X_train,y_train)
    yhat=neigh.predict(X_test)
    mean_acc[n-1] = metrics.accuracy_score(y_test, yhat)


    std_acc[n-1]=np.std(yhat==y_test)/np.sqrt(yhat.shape[0])

#plot model accuracy for different k
plt.plot(range(1,Ks),mean_acc,'g')
plt.fill_between(range(1,Ks),mean_acc - 1 * std_acc,mean_acc + 1 * std_acc, alpha=0.10)
plt.legend(('Accuracy ', '+/- 3xstd'))
plt.ylabel('Accuracy ')
plt.xlabel('Number of Nabors (K)')
plt.tight_layout()
plt.show()
print( "The best accuracy was with", mean_acc.max(), "with k=", mean_acc.argmax()+1)
```

By plotting the accuracy and values of k, the best accuracy was 0.5433 with k value equals to 9.



With the value of k being 9, a kNN model was built as following:

```
#kNN with k = 9
k = 9
#Train Model and Predict
neigh = KNeighborsClassifier(n_neighbors = k).fit(X_train,y_train)
```

4. RESULTS

To evaluate different models, Jaccard score and F1-score were calculated for all three models. Also, log loss was calculated for logistic regression model. The results are summarized in the following table:

	Algorithm	Jacard	F1-score	LogLoss
0	KNN	0.415206	0.538047	NA
1	Decision Tree	0.123197	0.454320	NA
2	Logistic Regression	0.288168	0.539320	0.669559

5. DISCUSSION

From the result table, we can see that the Jaccard score is the highest for kNN model at 0.415 and lowest for decision tree model at 0.123. The F1-score is the highest for logistic regression model at 0.539 and lowest for decision tree model at 0.454. From the result, all these models perform well with the decision tree model performed the worst. However, these models could have a better performance if it is a balance dataset for target variable and if there were fewer missing values in the dataset. In addition, changing the maximum depth of the tree might improve the accuracy of the decision tree model.

6. CONCLUSION

Based on the dataset and the machine learning models analysis, we can conclude that factors like weather conditions, road conditions, light conditions, not paying enough attention during driving, driving at high speed, and under influence of drugs or alcohol have some impact on the severity of the accidents.