Classification of Road Accident Severity

G Chan

14 Sep 2020

1. Introduction

1.1 Background

Road accident can occur anytime and anywhere and it may cause heavy casualties.

The cause of traffic accident could be attributed to various factors such as weather, road quality , drivers and pedestrian's awareness

To minimize the occurrence, we may leverage a machine learning model to classify the severity of the road accident to remind public

1.2 Problem

Data that might contribute to determining road safety might cover weather, road quality, drivers

and pedestrian's awareness. This project is to classify road accident severity.

1.3 Interest

Transportation Bureau , drivers and public would be very interested in indicative information about road accident severity . Maths , Statistic major students and data scientist would also be interested in the development of classification model

2. Data Understanding

2.1 Data Sources

The data is source by public transportation data for Seattle city

Link: https://github.com/garness/testing/blob/master/Data-Collisions.zip

Data source:Data - Collisions from coursera-IBM Data Science Course

In [3]:
```
import numpy as np
import pandas as pd
```

In [5]:
```
df=pd.read_csv('./DataCollisions.csv')
df.head()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3063: DtypeWarning: Columns (32) have mixed types.Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

Out[5]:

	SEVERITYCODE	X	Y	OBJECTID	INCKEY	COLDETKEY	REPORTNO	STATUS	ADDRTYPE	INTKEY	...	ROADCOND	LIGHTCOND
0	2	-122.323148	47.703140	1	1307	1307	3502005	Matched	Intersection	37475.0	...	Wet	Daylight
1	1	-122.347294	47.647172	2	52200	52200	2607959	Matched	Block	NaN	...	Wet	Dark - Street Lights On
2	1	-122.334540	47.607871	3	26700	26700	1482393	Matched	Block	NaN	...	Dry	Daylight
3	1	-122.334803	47.604803	4	1144	1144	3503937	Matched	Block	NaN	...	Dry	Daylight
4	2	-122.306426	47.545739	5	17700	17700	1807429	Matched	Intersection	34387.0	...	Wet	Daylight

2.2 Target Variable Definition

SEVERITYCODE is a target variable ''y'' for model development

```
In [8]: df.SEVERITYCODE.value_counts()
```

```
Out[8]: 1    108515
        2     45036
        Name: SEVERITYCODE, dtype: int64
```

1.Before data input, remove duplicated field"severitycode" in the original csv file

2.Severitycode is the target variable,y

3.Remove result related variables to avoid data leakage,eg.SEVERITYDESC

4.convert character to numeric data,eg.ROADCOND

2.3 Data Selection

Remove result related variables to avoid data leakage,eg.SEVERITYDESC

```
In [16]: df[['SEVERITYDESC','SEVERITYCODE']].groupby(['SEVERITYDESC']).agg(['min','max'])
```

Out[16]:

	SEVERITYCODE	
	min	max
SEVERITYDESC		
Injury Collision	2	2
Property Damage Only Collision	1	1

Other meaningless variable are also removed

```
In [23]: dataset=df.drop(columns=['X','Y','OBJECTID','INCKEY','COLDETKEY','REPORTNO',
            'STATUS','ADDRTYPE','INTKEY','LOCATION',
            'EXCEPTRSNCODE','EXCEPTRSNDESC','SEVERITYDESC','COLLISIONTYPE','PERSONCOUNT','PEDCOUNT',
             'PEDCYLCOUNT','VEHCOUNT','INCDATE','INCDTTM','JUNCTIONTYPE','SDOT_COLCODE','SDOT_COLDESC',
             'INATTENTIONIND','UNDERINFL','PEDROWNOTGRNT','SDOTCOLNUM','SPEEDING','ST_COLCODE'
             ,'ST_COLDESC','SEGLANEKEY','CROSSWALKKEY','HITPARKEDCAR'])
dataset.head()
```

Out[23]:

	SEVERITYCODE	WEATHER	ROADCOND	LIGHTCOND
0	2	Overcast	Wet	Daylight
1	1	Raining	Wet	Dark - Street Lights On
2	1	Overcast	Dry	Daylight
3	1	Clear	Dry	Daylight
4	2	Raining	Wet	Daylight

2.4 Data Manipulation

Assign missing to the field Weather,Roadcond,lightcond

```
In [76]: dataset['WEATHER']=dataset['WEATHER'].fillna(value='Unknown')
         dataset['WEATHER'].value_counts()

Out[76]: Clear                       86878
         Raining                     26468
         Overcast                    22300
         Unknown                     15912
         Snowing                       704
         Other                         701
         Fog/Smog/Smoke                442
         Sleet/Hail/Freezing Rain       96
         Blowing Sand/Dirt              34
         Severe Crosswind               16
         Name: WEATHER, dtype: int64
```

```
In [77]: dataset['ROADCOND']=dataset['ROADCOND'].fillna(value='Other')
         dataset['ROADCOND'].value_counts()

Out[77]: Dry              97842
         Wet              37741
         Unknown          13827
         Other             2085
         Ice               1035
         Snow/Slush         811
         Standing Water      96
         Sand/Mud/Dirt       66
         Oil                 48
         Name: ROADCOND, dtype: int64
```

```
In [78]: dataset['LIGHTCOND']=dataset['LIGHTCOND'].fillna(value='Other')
         dataset['LIGHTCOND'].value_counts()

Out[78]: Daylight                   91280
         Dark - Street Lights On    38829
         Unknown                    12464
         Dusk                        4707
```

Convert the string to number

```
In [79]: dataset['WEATHER_NUM']=0
         dataset.loc[dataset.WEATHER=='Clear','WEATHER_NUM']=4
         dataset.loc[dataset.WEATHER=='Raining','WEATHER_NUM']=3
         dataset.loc[dataset.WEATHER=='Overcast','WEATHER_NUM']=2
         dataset.loc[dataset.WEATHER=='Unknown','WEATHER_NUM']=1
         dataset.WEATHER_NUM.value_counts()

Out[79]: 4    86878
         3    26468
         2    22300
         1    15912
         0     1993
         Name: WEATHER_NUM, dtype: int64
```

```
In [80]: dataset['ROADCOND_NUM']=1
         dataset.loc[dataset.ROADCOND=='Dry','ROADCOND_NUM']=3
         dataset.loc[dataset.ROADCOND=='Wet','ROADCOND_NUM']=2
         dataset.loc[(dataset.ROADCOND=='Unknown')|(dataset.ROADCOND=='Other'),'ROADCOND_NUM']=0

         dataset.ROADCOND_NUM.value_counts()

Out[80]: 3    97842
         2    37741
         0    15912
         1     2056
         Name: ROADCOND_NUM, dtype: int64
```

```
In [81]: dataset['LIGHTCOND_NUM']=0
         dataset.loc[dataset.LIGHTCOND=='Daylight','LIGHTCOND_NUM']=5
         dataset.loc[dataset.LIGHTCOND=='Dusk','LIGHTCOND_NUM']=4
         dataset.loc[dataset.LIGHTCOND=='Dawn','LIGHTCOND_NUM']=3
         dataset.loc[dataset.LIGHTCOND=='Dark - Street Lights On','LIGHTCOND_NUM']=2
         dataset.loc[(dataset.LIGHTCOND=='Dark - No Street Lights')
                     |(dataset.LIGHTCOND=='Dark - Street Lights Off')
                     |(dataset.LIGHTCOND=='Dark - Unknown Lighting '),'LIGHTCOND_NUM']=1
```

3. Methodology

For model development, Variables X and Y are necessary

```
In [14]: X=dataset[['WEATHER_NUM','ROADCOND_NUM','LIGHTCOND_NUM']].values
         X[:5]

Out[14]: array([[2, 2, 5],
                [3, 2, 2],
                [2, 3, 5],
                [4, 3, 5],
                [3, 2, 5]])
```

```
In [15]: y=dataset['SEVERITYCODE'].values
         y[:5]

Out[15]: array([2, 1, 1, 1, 2])
```

As no continuous variables are involved, no need to apply normalization

Split the data into training set(70%) and testing set(30%)

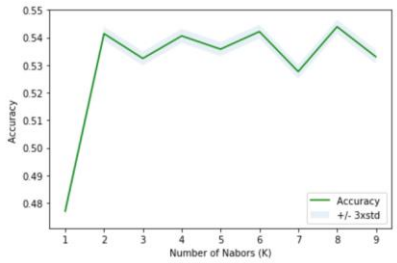
```
In [16]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=4)
         print ('Train set:', X_train.shape,  y_train.shape)
         print ('Test set:', X_test.shape,  y_test.shape)

             Train set: (155738, 3) (155738,)
             Test set: (38935, 3) (38935,)
```

4. Modelling

KNN would be used as machine learning model developement

K-Nearest Neighbors is an algorithm for supervised learning. Where the data is 'trained' with data points corresponding to their classification. Once a point is to be predicted, it takes into account the 'K' nearest points to it to determine it's classification.

```
In [23]: 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()
```




```
In [31]: # from sklearn import metrics
         # print("Train set Accuracy: ", metrics.accuracy_score(y_train, neigh.predict(X_train)))
         print("Test set Accuracy: ", metrics.accuracy_score(y_test, yhat))

             Test set Accuracy:  0.5439321946834468
```

When K=8, the accuracy reached the highest, accuracy rate around 55%

5. Result

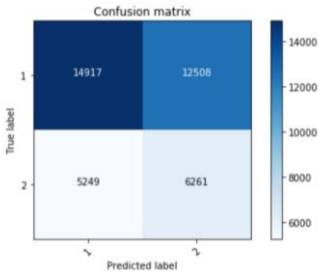
Apart from accuracy, there are other measurement about model performance.

1. Jaccard index: Similarity for the two sets of data , with a range 0% to 100%
2. F1 score:Model accuracy on a dataset

```
In [66]: from sklearn.metrics import f1_score
         f1_score(y_test, yhat, average='weighted')

Out[66]: 0.5638186373014934

In [67]: from sklearn.metrics import jaccard_similarity_score
         jaccard_similarity_score(y_test, yhat)

Out[67]: 0.5439321946834468
```

Confusion matrix:

```
              precision    recall  f1-score   support

           1       0.74      0.54      0.63     27425
           2       0.33      0.54      0.41     11510

   micro avg       0.54      0.54      0.54     38935
   macro avg       0.54      0.54      0.52     38935
weighted avg       0.62      0.54      0.56     38935

Confusion matrix, without normalization
[[14917 12508]
 [ 5249  6261]]
```



6. Discussion

1. The suitable data is limited. IF more data such as car type, drivers' condition, car speed are available in model development, there is a change the accuracy can be enhanced
2. Suggest number of KNN be limited within 10 to avoid wide range of group. It may be too difficult to apply in practical implement if number of K is too large

7. Conclusion

KNN with K=8 is selected to be a classification model about road severity in Seattle city