IBM Capstone Project

Car accident severity

Introduction

Car accidents can occur all the time, however there are some conditions were the probabilities of have an accident arise due multiple variables.

This report has as purpose develop a model for Seattle government to predict the probabilities of have a car accident and severity, based on different conditions as weather or road conditions.

The information was provided by Seattle Police Department from 2004 to 2020.

Business Problem

Identify the conditions that can cause future car accidents in order to alarm the people with anticipation to be aware and drive more carefully.

The avoid of accidents will result in multiples benefits:

- Save lives as main benefit
- Reduce costs in damage infrastructure
- Reduce cost from police and paramedics to attend each accident

---

Data

The information comes from Seattle Police Department and recorded by Traffic Records and include Collisions at intersection or mid-block of a segment. The period information is from 2004 to May 2020.

The information is organized in a CSV file with 37 attributes and originally 194673 rows, the information is labeled and unbalanced. Additionally, document with the description of each column were given.

Due our information is labeled we know the result for each record, we have selected the column **SEVERITYCODE** as D*ependent* variable. The possible values are:

- 0—unknown

- 1—prop damage
- 2—injury
- 2b—serious injury
- 3—fatality

The information is unbalanced by the difference in samples for each accident type. In our case there are only two types of accidents from the five possible options.

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

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

Data Understanding

During the data exploration the following columns were excluded due to the lack of relevance as dates, IDs or codes, fields that describe collisions as the type of injuries.

Columns excluded		
X	SEVERITYDESC	INATTENTIONIND
Y	PEDCOUNT	PEDROWNOTGRNT
OBJECTID	PEDCYLCOUNT	SDOTCOLNUM
INCKEY	VEHCOUNT	SPEEDING
COLDETKEY	INCDATE	ST_COLCODE
REPORTNO	INCDTTM	ST_COLDESC
STATUS	JUNCTIONTYPE	SEGLANEKEY
INTKEY	SDOT_COLCODE	CROSSWALKKEY
EXCEPTRSNCODE	SDOT_COLDESC	HITPARKEDCAR
PERSONCOUNT	ADDRTYPE	COLLISIONTYPE

The column LOCATION initially was considered as part of the variables to identify a possible accident but after a review in detail the information does not have a standard, so is difficult to apply a technique as One hot.

Examples:

- 6TH AVE AND JAMES ST
- ALASKAN WY VI NB BETWEEN S ROYAL BROUGHAM WAY ON RP AND SENECA ST OF F RP
- 30TH AVE BETWEEN E SPRUCE S ST AND E SPRUCE N ST

The column EXCEPTRSNDESC was used to filter the records and exclude the ones with value “Not Enough Information, or Insufficient Location Information”

Finally, the data was balanced using down sampling technique and the variables to predict the car accidents were:

- UNDERINFL
- WEATHER
- ROADCOND
- LIGHTCOND

Data frame balanced:

```
In [29]: df_result["SEVERITYCODE"].value_counts()

Out[29]: 2    57673
         1    57673
         Name: SEVERITYCODE, dtype: int64
```

Modeling

The models that can apply due we have a label dataset are:

- KNN
- SVM
- Decision Tree

In each model we are going to test different values in order to get the best result per model and then compare among each type.

Data Transformation

Due the columns selected the information appears as labeled so we need to change it to a code of numbers. Using factorize we did the change in all the labels and reset the index.

```
import numpy as np
df_x['WEATHER_num'] = pd.factorize(df_x['WEATHER'])[0]
```

	UNDERINFL	WEATHER_num	ROADCOND_num	LIGHTCOND_num
0	0	0	0	0
1	0	1	0	0
2	0	0	0	0
3	0	0	0	0
4	0	2	1	0

Training and test set

After clean and transform our information we split the data in 80% for training and 20% for testing.

In [27]:
```
from sklearn.model_selection import train_test_split
```

In [28]:
```
cols_X = ["UNDERINFL", "WEATHER", "ROADCOND", "LIGHTCOND"]

df_x = df_result[cols]
df_y = df_result["SEVERITYCODE"]

#Create test set
x_train, x_test, y_train, y_test = train_test_split(df_x, df_y, test_size=0.2, random_state=1)
```

KNN

For this model we try different values for K from 10 to 45, having as best result 14 as the best value for neighbor value.

KNN

```
k_initial=10
jaccard_array = []
f1_score_array = []

for k in range(k_initial, 45, 1):
    knn = KNeighborsClassifier(n_neighbors = k).fit(x_train, y_train)
    yhat = knn.predict(x_test)

    #Jaccard
    j = jaccard_similarity_score( y_test ,yhat)
    jaccard_array.append( j )
#     print(j)

    # f1_score
    f1 = f1_score(y_test, yhat, average='macro')
    f1_score_array.append(f1)



print(f'Best value for k= {jaccard_array.index( max(jaccard_array))+k_initial}')
print(f'Jaccard = {max(jaccard_array)}'  )
print(f'Jaccard = {max(f1_score_array)}'  )
```

```
Best value for k= 14
Jaccard = 0.5565596896490919
Jaccard = 0.5469972491792049
```

SVM

SVM

```
from sklearn import svm

svm_m = svm.SVC(kernel='rbf', gamma='scale')
svm_m.fit(x_train, y_train)
s_yhat = svm_m.predict(x_test)


jaccard = jaccard_similarity_score( y_test ,s_yhat)
f1 = f1_score(y_test, s_yhat, average='weighted')

print(f"Evaluation Jaccard : {jaccard}")
print(f"Evaluation F1 Score : {f1}")
```

```
Evaluation Jaccard : 0.5555898430611885
Evaluation F1 Score : 0.5249654288033598
```

Decision Tree

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier

jaccard_array = []
f1_score_array = []

md_initial = 3
for md in range(md_initial,10,1):

    dt = DecisionTreeClassifier(criterion='entropy', max_depth=md)
    dt.fit(x_train, y_train)
    dt_yhat = dt.predict(x_test)

    jaccard = jaccard_similarity_score( y_test ,s_yhat)
    f1 = f1_score(y_test, s_yhat, average='weighted')

    jaccard_array.append(jaccard)
    f1_score_array.append(f1)

print(f'Best value for max depth= {jaccard_array.index( max(jaccard_array))+md_initial}')
print(f"Evaluation Jaccard : {jaccard}")
print(f"Evaluation F1 Score : {f1}")
```

```
Best value for max depth= 3
Evaluation Jaccard : 0.5555898430611885
Evaluation F1 Score : 0.5249654288033598
```

Results

According with the models evaluated above the results are the following:

	Jaccard	F1 Score
KNN	0.5565	0.5469
SVM	0.5555	0.5249
Decision Tree	0.5555	0.5249

Considering the table, the best model to use is KNN.

Discussion

The information after analysis is reduced drastically from 37 columns to only use four relevant columns and the number of records used had to be reduced due down sampling technique in order to minimize execution time in each model evaluation.

For the first reduction, alternative sources should be considered for better predictions also to improve the number of independent variables. The goal is to have a tool that can predict in the best way when the accidents can occur.

Additional for each model more evaluations can be run changing the parameters as neighbor numbers in KNN using another range, type of kernel and gamma for SVM, type of criterion and max_depth for decision tree.

By last, other models can be explored to have evaluations using Jaccard or F1 Score closest to 1.

Conclusion

The information provided by Seattle Police Department is a first step to prove that a model can be generated to predict future accidents on the road and identify the type of information (independent variables) that can be used.

Another project can start to collect more information from other sources or directly from the roads.