CAR ACCIDENT SEVERITY

IBM DATA SCIENCE
Capstone Project

INTRODUCTION

- Car accident is one of the most important reason that cause people's dead every year.
- External factors such as bad weather, pedestrians who do not obey traffic rules, and animals that appear suddenly.
- Internal factors are mostly caused by improper driving, such as fatigue driving, drunk driving, driving after taking drugs and so on.

BUSINESS PROBLEM

- Traffic accident data can help to find out the causes of most traffic accidents, including weather, speed, road conditions and so on.
- The aim of the project is to use the data and build a machine learning model.
- When there gives some data, like weather, light condition, road condition, the model will classify the data, and make prediction about its possible damage: property damage or injury.

DATA METHODOLOGY

- The dataset include 194673 records of car accident, and 37 attributes.

- Clean and prepare the data in the data set.
- Normalize the data
- Split data into train and test
- KNN, Decision Tree and Logistic Regression.
- F1-score and Jaccard index to evaluate each of the method.

CODE FOR ANALYSING

Data cleaning and preparation

```
In [70]: df_group_one = df_car[['SEVERITYCODE','WEATHER','ROADCOND','LIGHTCOND']]
         df_group_one.head()
```

Out[70]:

	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

CODE FOR ANALYSING

Data Modeling-KNN

```
k =11
#Train Model and Predict
neigh = KNeighborsClassifier(n_neighbors = k).fit(x_train,y_train)
neigh
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=11, p=2,
           weights='uniform')
```

```
yhat = neigh.predict(x_test)
yhat[0:5]
```

```
array([2, 2, 2, 2, 2])
```

CODE FOR ANALYSING

Data Modeling-Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
accidentTree = DecisionTreeClassifier(criterion="entropy", max_depth = 7)
accidentTree # it shows the default parameters
```

```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=7,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

```
accidentTree.fit(x_train,y_train)
predTree = accidentTree.predict(x_test)
print (predTree [0:10])
print (y_test [0:10])
```

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

CODE FOR ANALYSING

Data Modeling-Logistic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix

LR = LogisticRegression(C=6, solver='liblinear').fit(x_train,y_train)
LR
```

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

```
yhat = LR.predict(x_test)
yhat
```

```
array([1, 2, 2, ..., 2, 2, 2])
```

```
yhat_prob = LR.predict_proba(x_test)
yhat_prob
```

```
array([[0.573691  , 0.426309  ],
       [0.47157777, 0.52842223],
       [0.47157777, 0.52842223],
       ...,
       [0.47157777, 0.52842223],
       [0.46167199, 0.53832801],
       [0.47157777, 0.52842223]])
```

CODE FOR ANALYSING

Model Evaluation

```
from sklearn.metrics import log_loss# predicted y
yhat_knn = neigh.predict(x_test)

# jaccard
jaccard_knn = jaccard_similarity_score(y_test, yhat_knn)
print("KNN Jaccard index: ", jaccard_knn)

# f1_score
f1_score_knn = f1_score(y_test, yhat_knn, average='weighted')
print("KNN F1-score: ", f1_score_knn)
```

```
KNN Jaccard index:  0.563026293177522
KNN F1-score:  0.5471223872325656
```

```
# predicted y
yhat_dt = accidentTree.predict(x_test)

# jaccard
jaccard_dt = jaccard_similarity_score(y_test, yhat_dt)
print("DT Jaccard index: ", jaccard_dt)

# f1_score
f1_score_dt = f1_score(y_test, yhat_dt, average='weighted')
print("DT F1-score: ", f1_score_dt)
```

```
DT Jaccard index:  0.5628544423440454
DT F1-score:  0.5347732983835123
```

Logistic Regression

```
# predicted y
yhat_lg = LR.predict(x_test)
yhat_lg_prob = LR.predict_proba(x_test)

# jaccard
jaccard_lg = jaccard_similarity_score(y_test, yhat_lg)
print("LR Jaccard index: ", jaccard_lg)

# f1_score
f1_score_lg = f1_score(y_test, yhat_lg, average='weighted')
print("LR F1-score: ", f1_score_lg)

# logloss
logloss_lg = log_loss(y_test, yhat_lg_prob)
print("LR log loss: ", logloss_lg)
```

```
LR Jaccard index:  0.5243598556452999
LR F1-score:  0.5091466260926107
LR log loss:  0.6856325273218654
```

RESULT

ALGORITHM	JACCARD	F1-SCORE	LOGLOSS
KNN	0.56302	0.547122	NA
Decision Tree	0.56285	0.534773	NA
LogisticRegression	0.52435	0.509146	0.68563

DISCUSSION AND CONCLUSION

- The Jaccard index and F1-Score are closed among these three method.
- In conclusion, the Logistic Regression is the best way to classify this dataset.
- The dataset in this capstone about weather, road condition, and light condition can directly pointing to certain classes, therefore we can conclude that particular conditions have an impact on whether or not an accident could result in property damage (class 1) or injury (class 2).