IBM Data Science Capstone: Car Accident Severity Report

Methawut Apikitmetha

September 27, 2020

1. Introduction

The objective of this project is to predict the severity of an accident by using several factors such as weather, road condition, light condition and so on.

To reduce the frequency of car accident in a community. So, this predictive model can give you a warning signals to avoid the accident, to drive more carefully or recommend you to change your routing

2. Data Preparation

Seattle Police Department (SPD) has been collecting detailed data about cars collisions. It will be used for developing model.

The data consists of 37 independent variables and 194,673 rows. "SEVERITYCODE" contains numbers that correspond to different levels of severity caused by an accident from 0 to 4.

- 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

I have decided to focus on 3 variables to predict severity. As following;

- "WEATHER"
- "ROADCOND"
- "LIGHTCOND"

"Unknown" and "Other" data has to be deleted. Then I need to downsampling due to the number of severity code is not balance.

3. Methodology

Once I have load data into Dataframe and do data preparation already.

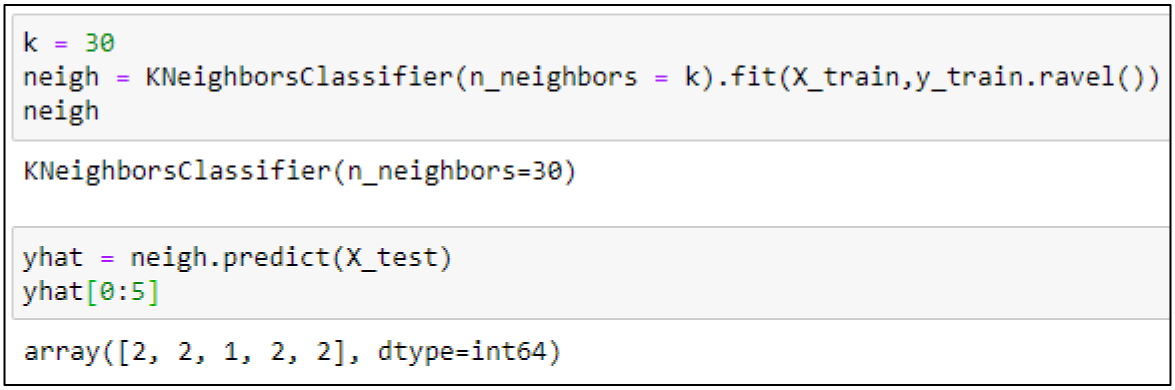
I will use the following models:

1. K-Nearest Neighbor (KNN)
2. Decision Tree
3. Logistic Regression

I define X and Y dataset then I split 80% for training and 20% for testing. And I have to convert object values into numerical values.

3.1 K-Nearest Neighbor (KNN)

I found that the most accurate k number is 30.



```
k = 30
neigh = KNeighborsClassifier(n_neighbors = k).fit(X_train,y_train.ravel())
neigh
```

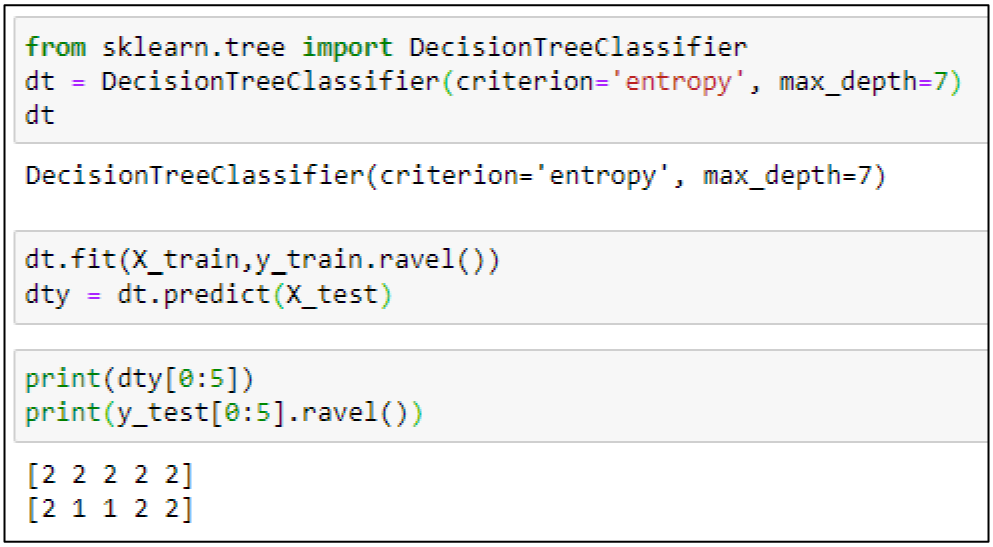
```
KNeighborsClassifier(n_neighbors=30)
```

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

```
array([2, 2, 1, 2, 2], dtype=int64)
```

3.2 Decision Tree

This model is most accurate with a max depth of 7.



```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier(criterion='entropy', max_depth=7)
dt
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=7)
```

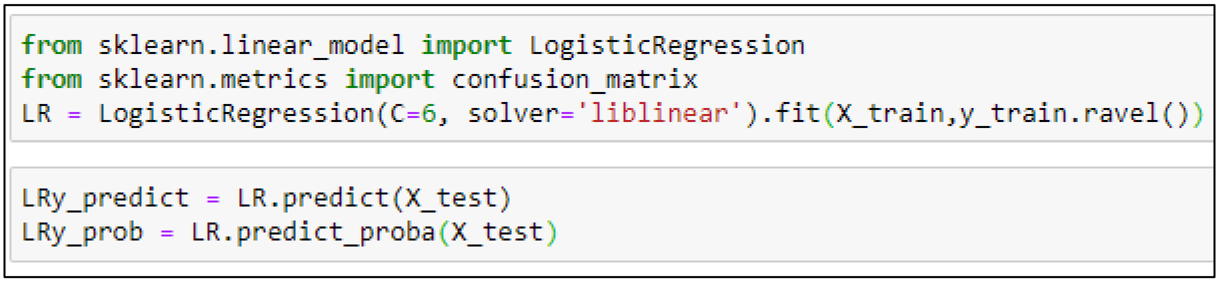
```
dt.fit(X_train,y_train.ravel())
dty = dt.predict(X_test)
```

```
print(dty[0:5])
print(y_test[0:5].ravel())
```

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

3.3 Logistic Regression

This model is most accurate when hyperparameter C is 6.



```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
LR = LogisticRegression(C=6, solver='liblinear').fit(X_train,y_train.ravel())
```

```
LRy_predict = LR.predict(X_test)
LRy_prob = LR.predict_proba(X_test)
```

4. Results

The results of the model evaluations are summarized as following;

K-Nearest Neighbor (KNN)

- Jaccard index: 0.25
- f1-score: 0.50

Decision Tree

- Jaccard index: 0.21
- f1-score: 0.48

Logistic Regression

- Jaccard index: 0.24
- f1-score: 0.48
- Log-Loss: 0.69

Based on the above results, KNN is the best model to predict car accident severity.

5. Discussion

Once we analyzed and cleaned the data, it was then fed through three ML models; K-Nearest Neighbor, Decision Tree and Logistic Regression. Although the first two are ideal for this project, logistic regression made most sense because of its binary nature.

Evaluation metrics used to test the accuracy of our models were Jaccard index, f-1 score and log-loss for logistic regression. Choosing different k, max depth and hyperparameter C values helped to improve our accuracy to be the best possible.

6. Conclusion

Based on historical data from weather, road and light conditions pointing to certain classes, we can conclude that particular weather conditions have a somewhat impact on whether or not travel could result in property damage (class 1) or injury (class 2).