IBM DATA SCIENCE

Car Accident Severity Capstone Project

Tim Skates

September 2020

Data Science Capstone

Car Accident Severity

Introduction

Last year in the United States, almost 39,000 people lost their lives in car accidents. In addition to that, an estimated 4.4 million people were hurt badly enough that they had to seek medical attention. While car accidents can happen at any time and any place, we also know that there may be external factors that may result in a higher frequency of crashes and/or a greater chance of injury. The goal of this project is to determine if we could predict the severity or risk of an accident and what factors may be at play. The ultimate goal for this data and model would be to determine if we could reduce harm by having the driver avoid such factors, such as driving in better conditions, choosing another time of day or avoiding weather.

The Data

The data for this project comes from the city of Seattle and has vast amounts of information on the accidents from this area. It is available at https://s3.us.cloud-object-storage.appdomain.cloud/cfcoursesdata/CognitiveClass /DP0701EN/version-2/Data-Collisions.csv.

The dataset that we used was provided by the Seattle Department of Transportation and included 194,673 accidents and included 37 features such as location or address of accident, weather conditions, driver impairment, and collision type. Each of these accidents was given a severity code. The severity of each accident was ranked in the following manner: fatality (3), serious injury (2b), i injury (2), prop damage (1) and unknown (0).

The dataset is unbalanced and needed to be cleaned. In order to do this, we enlisted the drop function of Pandas to take out all rows (accidents) with incomplete data, which will allow us to build a more accurate model.

The Methodology

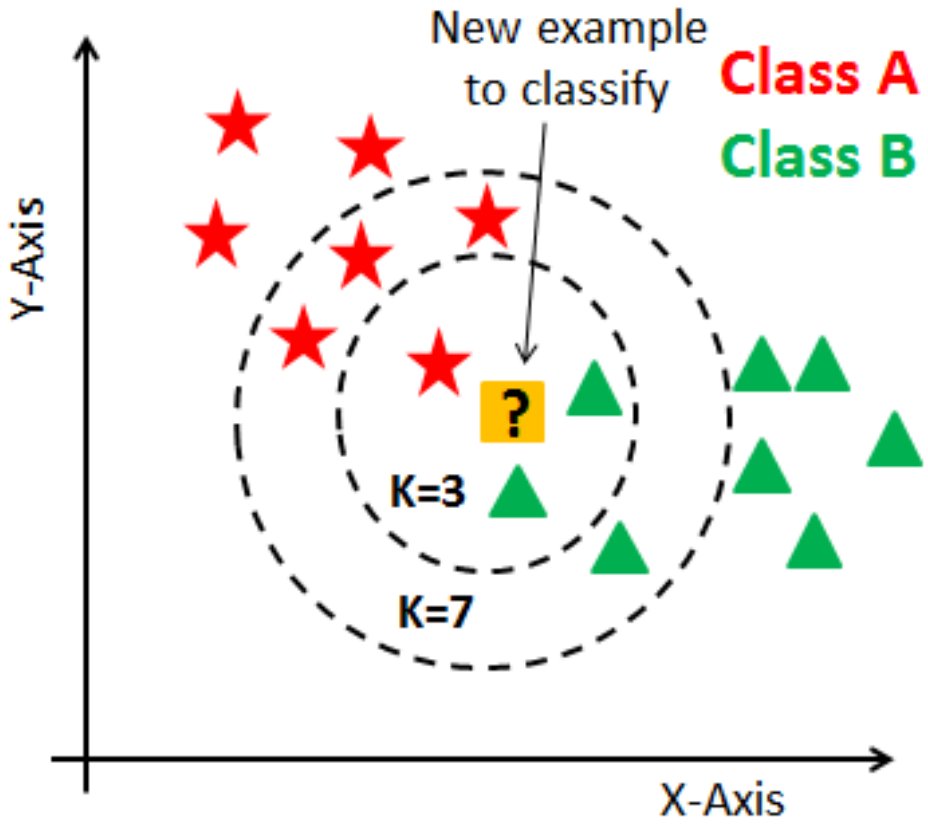
Data cleaning, by definition, is "the process of preparing data for analysis by removing or modifying data that is incorrect, incomplete, irrelevant, duplicated, or improperly formatted." In order to use this data we had to clean and drop the missing values. We also needed clean noisy data and transform our data. We also used a Pandas function to replace NAN values in the dataset. Finally, we used three Machine Learning Algorithms to determine what model best fitted our data and therefore which one we could use to predict the severity of car accidents based on our factors.

Machine Learning Algorithms

1. KNN - K Nearest Neighbor

The KNN algorithm assumes that similar things exist in close together or are near each other.

http://res.cloudinary.com/dyd911kmh/image/upload/f_auto,q_auto:best/v1531424125/Knn_k1_z96jba.png



As you can see in the above diagram, the general idea is that we have a new example and we will want to classify it with it's closest neighbors under the premise that closeness means they will be alike. We can also choose the 'k' which is how far out we want the algorithm to look and we should pick a 'k' that minimizes the noise in our data. KNN takes immediate action on the dataset and assigns the new data point in the group that is closest to it, sometimes earning it the nickname "Lazy Learner"

Code

K-Nearest Neighbors

```
from sklearn.neighbors import KNeighborsClassifier
k=23

kneighbors = KNeighborsClassifier(n_neighbors = k).fit(x_train, y_train)
k_Predict = kneighbors.predict(x_test)
k_Predict[0:5]
```

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

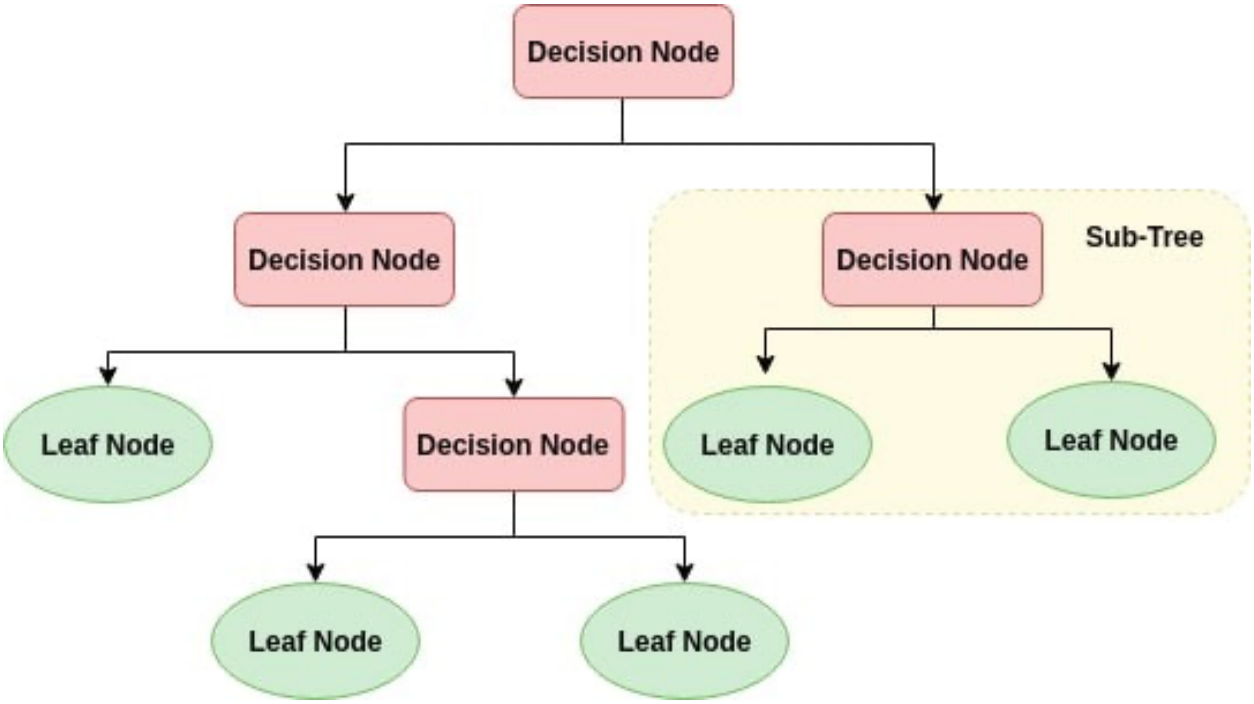
```
j=jaccard_score(y_test,k_Predict)
f=f1_score(y_test, k_Predict, average = 'macro')
print("Jaccard Score: ",j)
print("F1 Score: ",f)
```

```
Jaccard Score:  0.42309313105773283
F1 Score:  0.6074617239040984
```

2. Decision Tree

Decision Tree is a supervised learning technique that is used for regression and classification problems, but it is mostly used for classification. It is structured like an upside down tree and the nodes represent features of the data set, the branches represent the rules and the leaf nodes represent the outcome. The following diagram shows how this works.

https://res.cloudinary.com/dyd911kmh/image/upload/f_auto,q_auto:best/v1545934190/1_r5ikdb.png



Code

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier (criterion = 'entropy', max_depth = 9)

dt.fit(x_train, y_train)
dt_predict = dt.predict(x_test)
dt_predict[0:5]
```

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

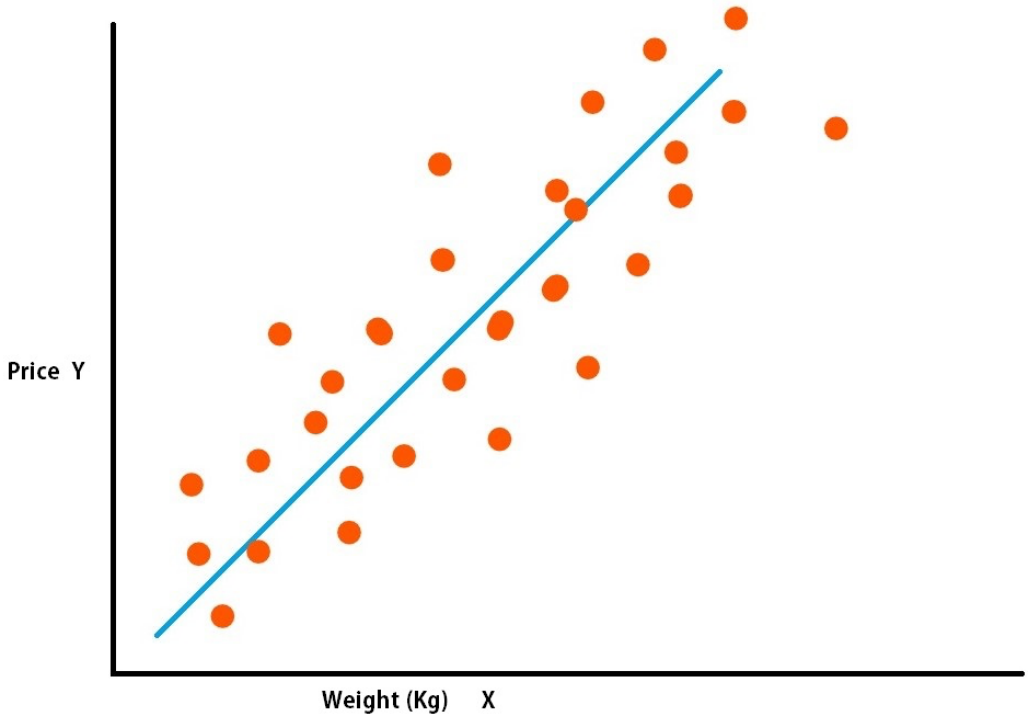
```
j2=jaccard_score(y_test,dt_predict)
f2=f1_score(y_test,dt_predict, average = 'macro')
print("Jaccard Score: ",j2)
print("F1 Score: ",f2)
```

```
Jaccard Score:  0.4391772524606713
F1 Score:  0.617955163978534
```

3. Linear Regression

The Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables. It finds out how the dependent variable is changing as the value of the independent variable or variables change. The linear regression model shows a sloped straight line representing the relationship between the variables as shown below.

https://codespeedy.com/wp-content/uploads/2019/05/linear-regression-diagram-Python.jpg



Code

Linear Regression

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

logreg_predict = LogReg.predict(x_test)
logreg_prob= LogReg.predict_proba(x_test)
logreg_prob
```

```
array([[0.3285926 , 0.6714074 ],
       [0.65971659, 0.34028341],
       [0.58208952, 0.41791048],
       ...,
       [0.59150057, 0.40849943],
       [0.38589354, 0.61410646],
       [0.50987312, 0.49012688]])
```

```
j1=jaccard_score(y_test,logreg_predict)
f1=f1_score(y_test,logreg_predict, average = 'macro')
print("Jaccard Score: ",j1)
print("F1 Score: ",f1)
print("Log Loss: ",log_loss(y_test,logreg_prob))
```

```
Jaccard Score:  0.4585652934627008
F1 Score:  0.6090004494835676
Log Loss:  0.6554544694261182
```

Model Development

With the data having categorical variables, we have used a classification model. To do this we used K-Nearest Neighbor, Linear Regression and Decision Tree algorithms and determined the Jaccard Score and the F1 score. The Jaccard score is a measure of how close the predicted values are to the actual true values. The score is on a scale of 0 to 1, with 1 being the most similar or accurate. The F1 score comes from the precision and recall of the test values. Precision is the number of correctly identified positive results divided by the number of all positive results, (including those not identified correctly) and the recall is the number of correctly identified positive results divided by the number of all samples that should have been identified as positive. This score is also from 0 to 1, with 1 being the most accurate.

Result

	ML Model	Jaccard Score	F1 Score
0	KNN	0.423093	0.607462
1	Linear Regression	0.458565	0.609000
2	Decision Tree	0.439177	0.617955

Conclusion

The goal of this project was to determine if we could predict accident severity based on certain factors, such as intersection type, weather, road conditions, etc. Based on our dataset, we found that a Linear Regression Model was the best model to use based on the Jaccard and F1 scores. We can use this Linear Regression model to predict the severity of a car crash and what factors may be at play. Authorities could use this data and model to determine what actions could be taken to reduce risk and minimize harm caused by car accidents.