Capstone Project - Car Accident Severity

Keerthi Prakash

September 2020

Introduction

Every year car accidents cause hundreds of thousands of deaths worldwide. According to a research conducted by the World Health Organization (WHO) there were 1.35 million road tra c deaths globally in 2016, with millions more sustaining serious injuries and living with long-term adverse health consequences. Globally, road tra c crashes are a leading cause of death among young people, and the main cause of death among those aged 15{29 years. Road tra c injuries are currently estimated to be the eighth leading cause of death across all age groups globally, and are predicted to become the seventh leading cause of death by 2030.

For the final capstone project in the IBM certificate course, we want to analyze the accident "severity" in terms of human fatality, traffic delay, property damage, or any other type of accident bad impact. The data was collected by Seattle SPOT Traffic Management Division and provided by Coursera via a link. This dataset is updated weekly and is from 2004 to present. It contains information such as severity code, address type, location, collision type, weather, road condition, speeding, among others.

The target audiences of this study are those people who really care about the traffic records, especially in the transportation department. Also, we want to figure out the reason for collisions and help to reduce accidents in the future.

Data

The data was collected by Seattle SPOT Traffic Management Division and provided by Coursera via a link. There are 194,673 observations and 38 variables in this data set. Since we would like to identify the factors that cause the accident and the level of severity, we will use SEVERITYCODE as our dependent variable Y (Target Variable) , and try different combinations of independent variables X to get the result. Since the observations are quite large, we may need to filter out the

missing value and delete the unrelated columns first. Then we can select the factor which may have more impact on the accidents such as weather, road condition, and light condition.

The target Data to be predicted under (SEVERITYCODE 1-prop damage 2-injury) label.

Other important variables include in the given dataset:

• ADDRTYPE: Collision address type: Alley, Block, Intersection

• LOCATION: Description of the general location of the collision

• PERSONCOUNT: The total number of people involved in the collision helps identify severity involved

• PEDCOUNT: The number of pedestrians involved in the collision helps identify severity involved

• PEDCYLCOUNT: The number of bicycles involved in the collision helps identify severity involved

• VEHCOUNT: The number of vehicles involved in the collision identify severity involved

• JUNCTIONTYPE: Category of junction at which collision took place helps identify where most collisions occur

• WEATHER: A description of the weather conditions during the time of the collision

• ROADCOND: The condition of the road during the collision

• LIGHTCOND: The light conditions during the collision

• SPEEDING: Whether or not speeding was a factor in the collision (Y/N)

• SEGLANEKEY: A key for the lane segment in which the collision occurred

• CROSSWALKKEY: A key for the crosswalk at which the collision occurred

• HITPARKEDCAR: Whether or not the collision involved hitting a parked car

Furthermore, because of the existence of null values in some records, the data needs to be preprocessed before any further processing.

Feature Selection

For implementing the solution, I have used Github as a repository and running Jupyter Notebook to preprocess data and build Machine Learning models. Regarding coding, I have used Python and its popular packages such as Pandas, NumPy and Sklearn.

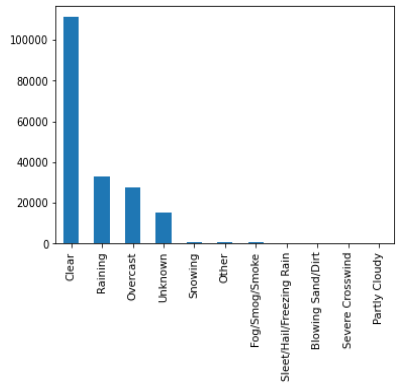
Once I have load data into Pandas Dataframe, used '*dtypes'* attribute to check the feature names and their data types. Then I have selected the most important features to predict the severity of accidents in Seattle. Among all the features, the following features have the most influence in the accuracy of the predictions:

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

Also, as I mentioned earlier, "SEVERITYCODE" is the target variable.

I have run a value count on road ('ROADCOND') and weather condition ('WEATHER') to get ideas of the different road and weather conditions. I also have run a value count on light condition ('LIGHTCOND'), to see the breakdowns of accidents occurring during the different light conditions. I have run a value count on Location. The results can be seen below:

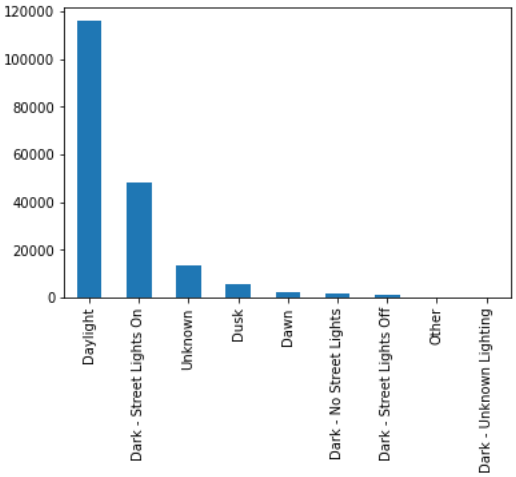
```
In [30]: df['WEATHER'].value_counts().plot(kind ='bar')
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x1f90c12b850>
```




```
In [29]: df['WEATHER'].value_counts().to_frame()
Out[29]:
```

	WEATHER
Clear	111135
Raining	33145
Overcast	27714
Unknown	15091
Snowing	907
Other	832
Fog/Smog/Smoke	569
Sleet/Hail/Freezing Rain	113
Blowing Sand/Dirt	56
Severe Crosswind	25
Partly Cloudy	5

```
In [40]: df['LIGHTCOND'].value_counts().plot(kind ='bar')
Out[40]: <matplotlib.axes._subplots.AxesSubplot at 0x1f910d849d0>
```

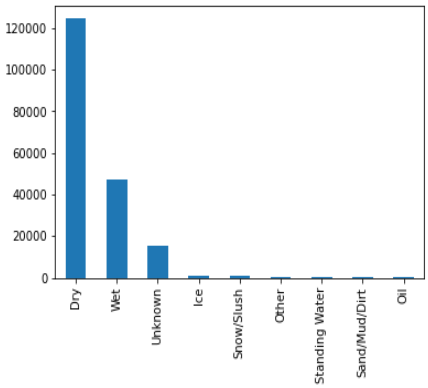



```
In [39]: df['LIGHTCOND'].value_counts().to_frame()
Out[39]:
```

	LIGHTCOND
Daylight	116137
Dark - Street Lights On	48507
Unknown	13473
Dusk	5902
Dawn	2502
Dark - No Street Lights	1537
Dark - Street Lights Off	1199
Other	235
Dark - Unknown Lighting	11

```
In [35]: df['ROADCOND'].value_counts().plot(kind ='bar')
```

```
Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x1f910dec100>
```




```
In [36]: df['ROADCOND'].value_counts().to_frame()
```

Out[36]:

	ROADCOND
Dry	124510
Wet	47474
Unknown	15078
Ice	1209
Snow/Slush	1004
Other	132
Standing Water	115
Sand/Mud/Dirt	75
Oil	64

```
In [48]: df['LOCATION'].value_counts().to_frame()
```

Out[48]:

	LOCATION
BATTERY ST TUNNEL NB BETWEEN ALASKAN WY VI NB AND AURORA AVE N	276
BATTERY ST TUNNEL SB BETWEEN AURORA AVE N AND ALASKAN WY VI SB	271
N NORTHGATE WAY BETWEEN MERIDIAN AVE N AND CORLISS AVE N	265
AURORA AVE N BETWEEN N 117TH PL AND N 125TH ST	254
6TH AVE AND JAMES ST	252
...	...
27TH AVE W BETWEEN W RAYE ST AND W ARMOUR ST	1
NW 80TH ST BETWEEN 27TH AVE NW AND EARL AVE NW	1
23RD AVE NW BETWEEN NW 67TH ST AND NW 70TH ST	1
S RIVER ST BETWEEN 5TH AVE S AND 5TH PL S	1
AIRPORT NB WAY S BETWEEN AIRPORT WAY NB OFF RP AND S ADDITION ST	1

24102 rows × 1 columns

I visualized the data in the form of bar graphs from original dataset. Most crashes happened in clear, dry, and bright conditions. Most days are clear, dry, and bright, so it's no surprise that most car crashes occur under these conditions. I also found out that crashes with a distracted driver or an impaired driver are statistically more likely to result in injury, which is also not a surprise.

I thought that maybe weather, road, and light condition may cause more accidents. However, we do figure out that the accidents are highly related to some specific locations. Thus, the traffic management division could try to improve the safety instructions or some other factors that could reduce the accidents.

Data Preprocessing

The dataset in the original form is not ready for data analysis. In order to prepare the data, first, we need to drop the non-relevant columns. In addition, most of the features are of object data types that need to be converted into numerical data types.

After analyzing the data set, I have decided to focus on only four features, severity, weather conditions, road conditions, and light conditions, among others.

In it's original form, this data is not fit for analysis. For one, there are many columns that we will not use for this model. Also, most of the features are of type object, when they should be numerical type.

We must use label encoding to covert the features to our desired data type.

Resampling Data

To get a good understanding of the dataset, I have checked different values in the features. The results show, the target feature SEVERITYCODE is imbalance, so we use a simple statistical technique to balance it.

In fact, severitycode in class 1 is nearly three times the size of class 2.

We can fix this by downsampling the majority class.

```
Out[6]: 2    57052
        1    57052
        Name: SEVERITYCODE, dtype: int64
```

Define X and y

```
In [7]: X = balanced_df[["ROADCOND","WEATHER","LIGHTCOND"]].values
        y = balanced_df.SEVERITYCODE
```

Normalize the data

```
]: from sklearn import preprocessing

]: X = preprocessing.StandardScaler().fit(X).transform(X)
```

Splitting our Data Set into Training Data and Test Data

We will use 30% of our data for testing and 70% for training.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)

print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)

(79872, 3) (34232, 3) (79872,) (34232,)
```

Methodology

After balancing SEVERITYCODE feature, and standardizing the input feature, the data has been ready for building machine learning models. I have employed three machine learning models:

K-Nearest Neighbor (KNN)

KNN will help us predict the severity code of an outcome by finding the most similar to data point within k distance.

Decision Tree

A decision tree model gives us a layout of all possible outcomes so we can fully analyze the concequences of a decision. In context, the decision tree observes all possible outcomes of different weather conditions.

Logistic Regression

Because our dataset only provides us with two severity code outcomes, our model will only predict one of those two classes. This makes our data binary, which is perfect to use with logistic regression.

K-Nearest Neighbor (KNN)

```
: #Train Model & predict
  knn_model  = knn(n_neighbors = 14).fit(X_train,y_train)
  knn_yhat = knn_model.predict(X_test)
  knn_yhat[0:5]
```

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

```
: # Jaccard
  j1=jaccard_score(y_test, knn_yhat)
  j1
```

```
: 0.4503557673477381
```

```
: from sklearn.metrics import f1_score

  f1=f1_score(y_test, knn_yhat, average ='macro')
  f1
```

```
: 0.4600894929150905
```

```
: print(classification_report(y_test,knn_yhat))
```

```
              precision    recall  f1-score   support

           1       0.51      0.80      0.62     17214
           2       0.51      0.21      0.30     17018

    accuracy                           0.51     34232
   macro avg       0.51      0.51      0.46     34232
weighted avg       0.51      0.51      0.46     34232
```

Decision Tree

```
#Libraries

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

```
LR_y_pred = LR.predict(X_test)
```

```
LR_y_prob = LR.predict_proba(X_test)
```

```
log_loss(y_test, LR_y_prob)
```

```
0.692454836719203
```

```
j3=jaccard_score(y_test, LR_y_pred)
j3
```

```
0.27325174825174825
```

```
f3=f1_score(y_test,LR_y_pred , average ='macro')
f3
```

```
0.5032293174917176
```

```
print(classification_report(y_test, LR_y_pred))
```

```
              precision    recall  f1-score   support

           1       0.52      0.36      0.43     17214
           2       0.51      0.67      0.58     17018

    accuracy                           0.51     34232
   macro avg       0.52      0.52      0.50     34232
weighted avg       0.52      0.51      0.50     34232
```

Logistic Regression

```
from sklearn.tree import DecisionTreeClassifier
```

```
# Main Model

dt_model = DecisionTreeClassifier(criterion = 'entropy', max_depth = 7)
dt_model.fit(X_train, y_train)
dt_model
```

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

```
dt_y_pred = dt.predict(X_test)
```

```
j2=jaccard_score(y_test, dt_y_pred)
j2
```

```
0.2657691458839108
```

```
f2=f1_score(y_test, dt_y_pred, average ='macro')
f2
```

```
0.500933648294112
```

```
print(classification_report(y_test, dt_y_pred))
```

```
              precision    recall  f1-score   support

           1       0.53      0.35      0.42     17214
           2       0.51      0.68      0.58     17018

    accuracy                           0.51     34232
   macro avg       0.52      0.52      0.50     34232
weighted avg       0.52      0.51      0.50     34232
```

Evaluation

Algorithm	Jaccard	F1-score
KNN	0.450356	0.460089
Decision Tree	0.265769	0.500934
Logistic Regression	0.273252	0.503229

In the beginning of the notebook, we had categorical data that was of type 'object'. This is not a data type that we could have fed through an algoritim, so label encoding was used to created new classes that were of type int8; a numerical data type.

After solving that issue we were presented with another - imbalanced data. As mentioned earlier, class 1 was nearly three times larger than class 2. The solution to this was downsampling the majority class with sklearn's resample tool. We downsampled to match the minority class exactly with 58188 values each.

Once we analyzed and cleaned the data, it was then fed through three ML models; K-Nearest Neighbor, Decision Tree and Logistic Regression. The logistic regression, KNN, and SVM models have similar accuracy that we already seen in classification report. 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 score, f-1 score. Choosing different k, max depth and hyparameter C values helped to improve our accuracy to be the best possible.

Conclusion

In this study, I analyzed the relationship between severity of an accident and some characteristics which describe the situation that involved the accident.I built and compared 3 different classification models to predict whether an accident would have a high or low severity. These models can have multiple application in real life.

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

By identifying the features that favor the most the gravity of an accident, these could be tackled by improving road conditions or increasing the awareness of the population.Furthermore, there are some places which has more accidents during the dark time. For those places, adding lights might be a good solution to reduce the collisions. Also, when more cars involved in the accident, it seems that the level of severity will increase. They may need to be responded immediately to save more life.