The Prediction of Car Accidents Severity

IBM Applied Data Science Capstone

By: Mickael Aghajarian

September 2020

1. Introduction

Motor vehicle crashes have been resulted in many deaths and injuries in the world. For example, based on the U.S. National Highway Traffic Safety Administration (NHTSA), there were 37,461 people killed in crashes on U.S. roadways during 2016, an increase from 35,485 in 2015. As another example, distracted driving caused 1.25 million deaths worldwide in 2015, with an estimated 3,477 deaths in the United States alone [1].

By analyzing some factors such as weather condition, road condition, and the speed of cars, some effective machine learning algorithms can be developed to predict the severity of car accidents before they would actually occur. Such intelligent systems can mitigate the number of deaths and injuries by warning drivers so that they would drive more carefully or even take another route if they could. As a result, the driving safety could be improved dramatically.

1.2 Business problem

Governments would be highly interested in such systems with the ability to predict the severity of car accidents ahead of time because they could save more lives by reducing the time of transporting the injured people to nearby hospitals for treatment. The faster the proper medical care is provided, the greater the chance of survival is for injured people in car accidents. Hospitals could also reap the benefits of such systems by getting prepared in advanced to admit injured people involved in accidents. Car companies would also be interested in equipping their products with such intelligent systems to alert drivers in advance and in turn increase the car safety. Thus, the main objective of this project is to train an effective machine learning model by which the severity of car accidents can be predicted.

2. Data

The dataset of car accidents in the Seattle city (from 2004 to present) can be found from the below link. All collisions have been provided by the Seattle Police Department and recorded by Traffic Records.

https://s3.us.cloud-object-storage.appdomain.cloud/cf-courses-data/CognitiveClass/DP0701EN/version-2/Data-Collisions.csv

The dependent variable is the accident severity in terms of 37 independent variables such as weather, road, and light conditions during the time of the collision. The target variable (SEVERITYCODE) is an integer number whose value can be either 0 (unknown), 1 (property damage), 2 (injury), 2b (serious injury), or 3(fatality).

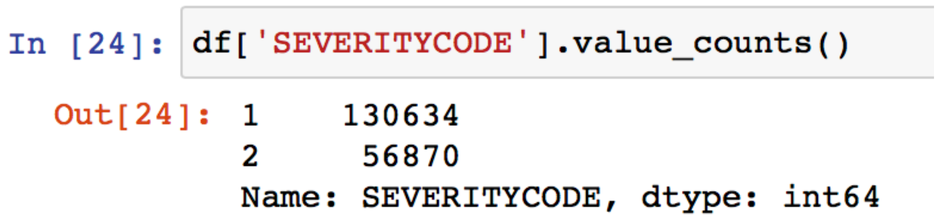
2.1 Feature Selection

In the dataset, there are 37 independent features some of which might not be helpful to be used for training the machine learning model. In this work, five features were selected to predict the severity of car accidents. The first feature is "ADDRTYPE" that determines the address type of the collision ("Alley", "Block", and "Intersection"). The second feature is "COLLISIONTYPE" that is the collision type (e.g., sideswipe, parked car). The third feature is "WEATHER" that describes the weather condition during the time of the collision. The fourth feature is "ROADCOND" that is the condition of the road during the collision. The fifth feature is "LIGHTCOND" which describes the light conditions during the collision. The type of all independent features is *object* (i.e., text) while the type of the target variable is *int64*.

2.2 Data Preprocessing

Before using the data to train the model, the data should be prepared. First, the unnecessary columns were dropped from the dataset. Since the type of the independent features was *object*, we needed to convert their type to numerical data types (i.e., *int64*). Also, there were some missing values among the data all of which were removed from the dataset. Next, we needed to check whether or not the dataset was balanced. As can be seen from Figure 1, there were 130,634 samples with property damage while there were only 56,870 samples with injury. Thus, the dataset was imbalance which meant we needed to balance it to prevent training a biased model. Figure 2 shows the histogram of the imbalance target variable.

To balance the dataset, the samples with property damage (i.e., the majority samples) were down-sampled so that their number of samples was reduced from 130,634 to 56,870. As a result, the number of samples in both groups was equal that meant the dataset became balanced. Figure 3 shows the histogram of the balanced target variable. After doing these preprocessing steps, the data was ready to be used to train the model. Figure 4 demonstrates the first five rows of the data after preprocessing.



```
In [24]: df['SEVERITYCODE'].value_counts()

Out[24]: 1    130634
         2     56870
         Name: SEVERITYCODE, dtype: int64
```

Figure 1. unbalanced dataset

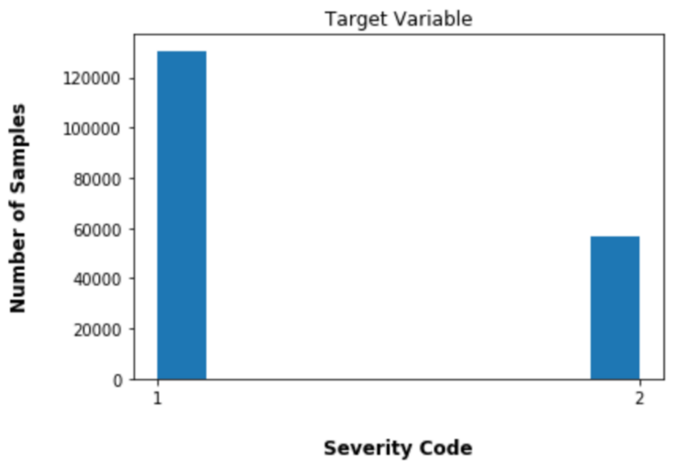


Figure 2. histogram of the imbalance target variable

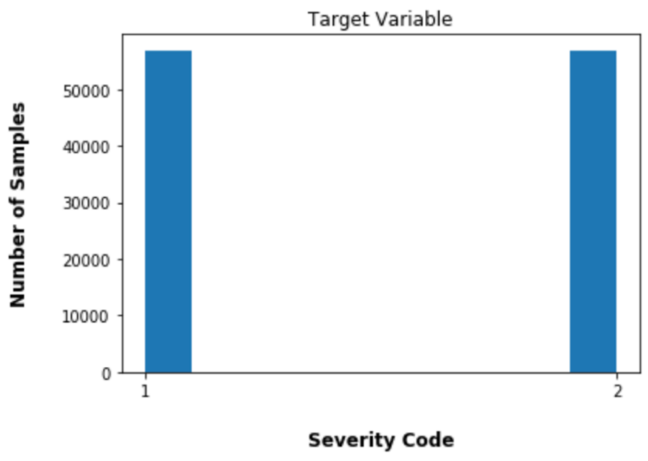
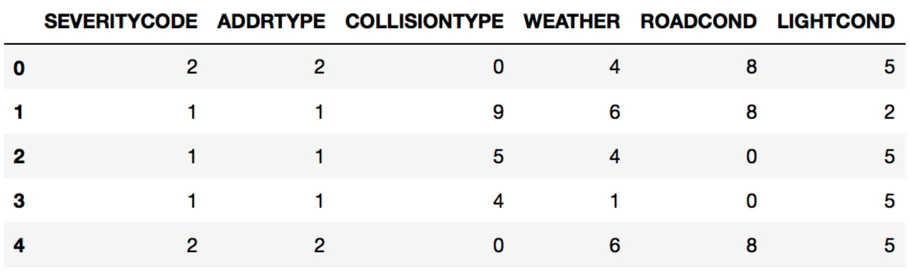


Figure 3. histogram of the balanced target variable



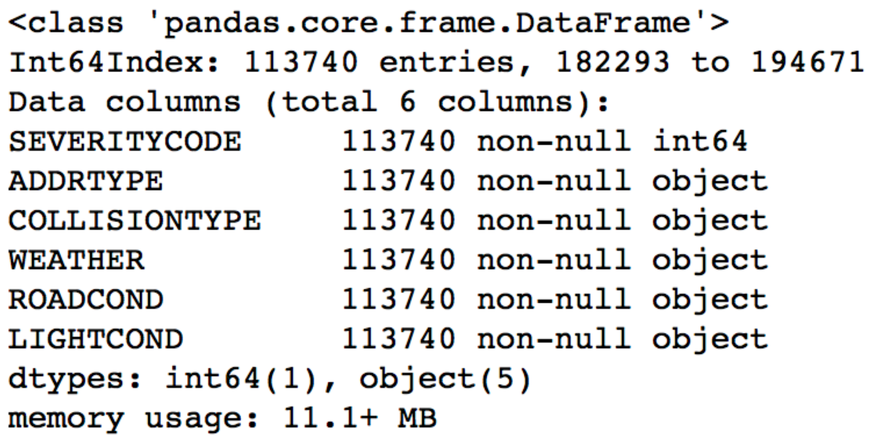
	SEVERITYCODE	ADDRTYPE	COLLISIONTYPE	WEATHER	ROADCOND	LIGHTCOND
0	2	2	0	4	8	5
1	1	1	9	6	8	2
2	1	1	5	4	0	5
3	1	1	4	1	0	5
4	2	2	0	6	8	5

Figure 4. the first five row of the data after preprocessing

2.3 Exploratory Data Analysis

We applied some exploratory data analysis to have a better understanding of the data. Since all the selected features were categorical variables, statistical summary of the data (e.g., *df.describe*) would not be very helpful. Figure 5 shows a summary of the balanced data set.

Figure 6 depicts the *countplot* for five independent variables. By looking at these plots, we gain a better understanding of the data set. For examples, by looking at the *countplot* of the weather condition variable, it can be seen that most of the accidents occurred when the weather condition was clear. As another example, by looking at the *countplot* of the road condition variable, it can be seen that most of the accidents occurred when the road condition was dry.



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 113740 entries, 182293 to 194671
Data columns (total 6 columns):
SEVERITYCODE     113740 non-null int64
ADDRTYPE         113740 non-null object
COLLISIONTYPE    113740 non-null object
WEATHER          113740 non-null object
ROADCOND         113740 non-null object
LIGHTCOND        113740 non-null object
dtypes: int64(1), object(5)
memory usage: 11.1+ MB
```

Figure 5. summary of the balanced data set

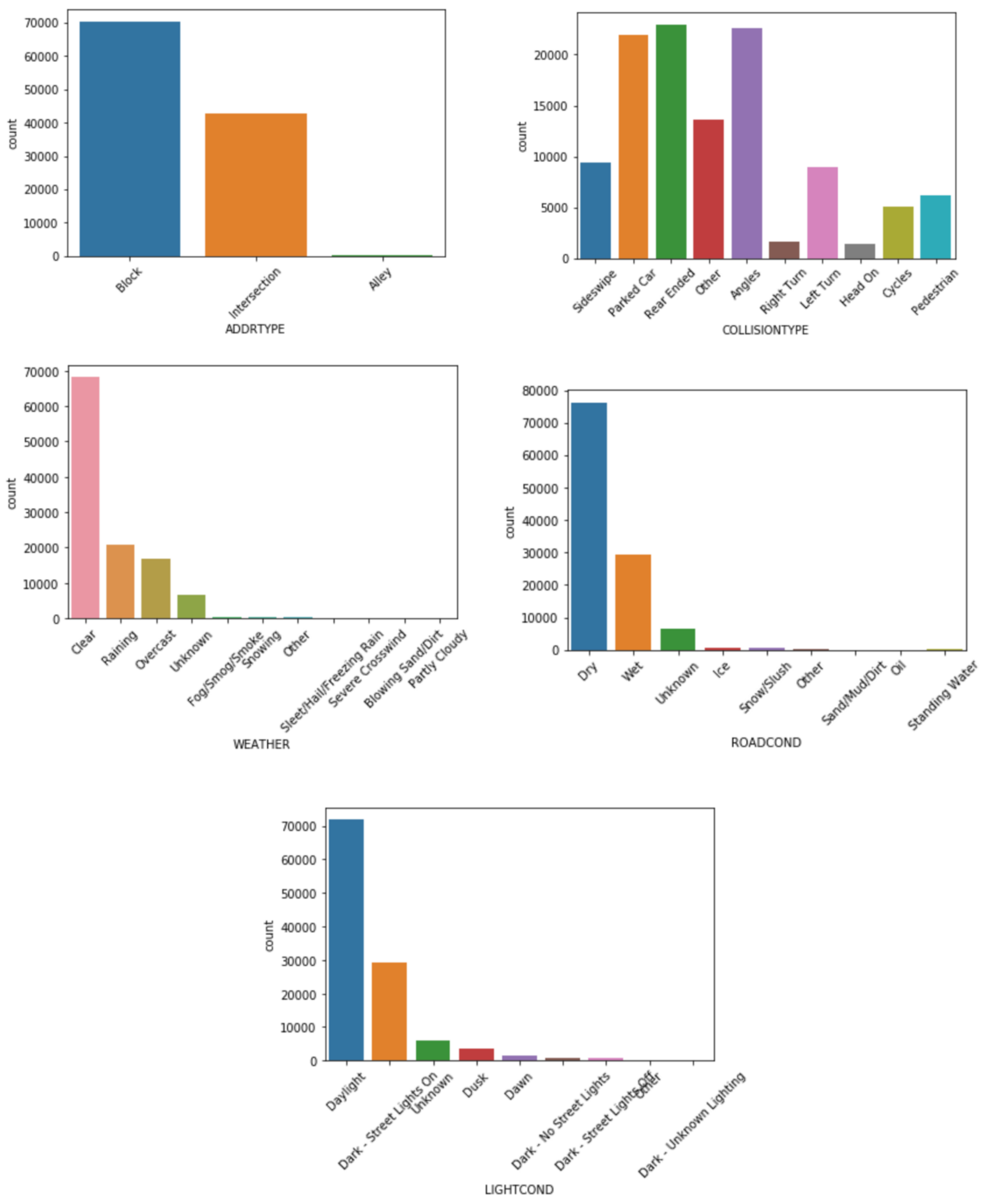


Figure 6. *countplot* of the independent variables

Although Figure 6 is helpful to gain a better understanding of the data set, it does not give us any information about the severity of accidents. To uncover relationships between five independents variable and the target variable, *Seaborn* library was used to generate plots shown in Figure 7 to Figure 11. Figure 7 tells us that most of the accidents with property damage occurred at blocks while most of the accidents with injury occurred at intersections. From Figures 9, 10, and 11, it can be seen that most of the accidents (with property damage or injury) occurred during the daylight hours when the weather condition was clear and the road condition was dry which was somehow surprising!

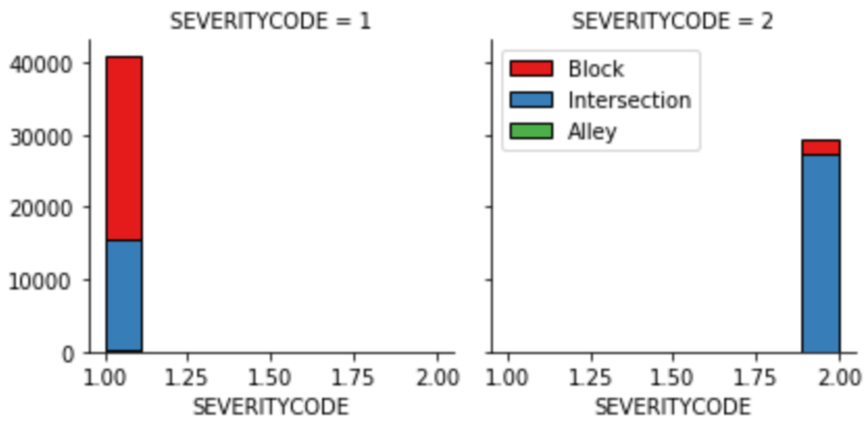


Figure 7. relationship between the address type and severity of accidents

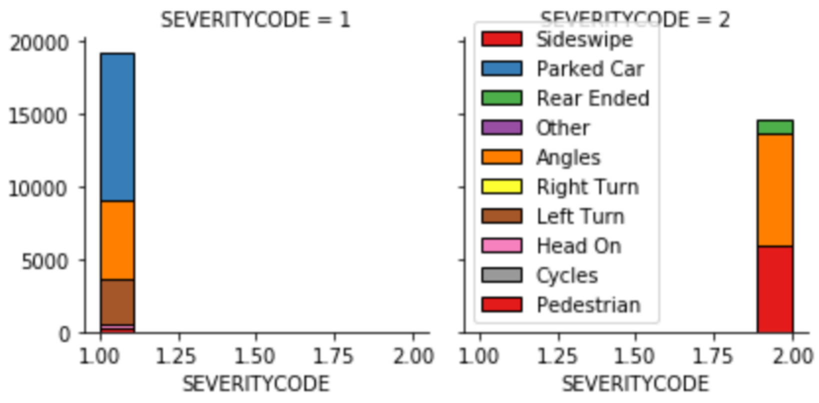


Figure 8. relationship between the collision type and severity of accidents

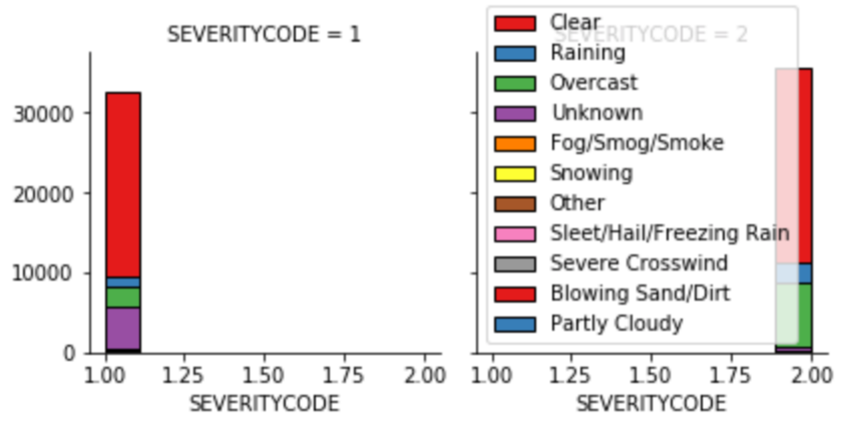


Figure 9. relationship between the weather condition and severity of accidents

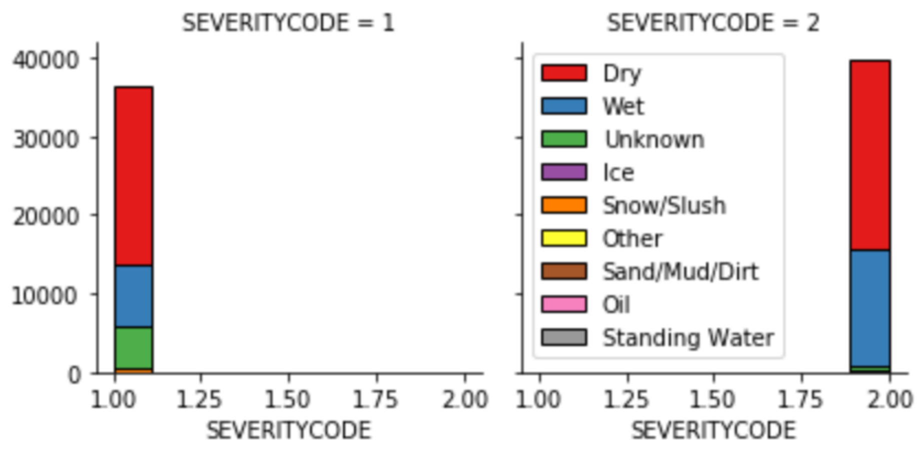


Figure 10. relationship between the read condition and severity of accidents

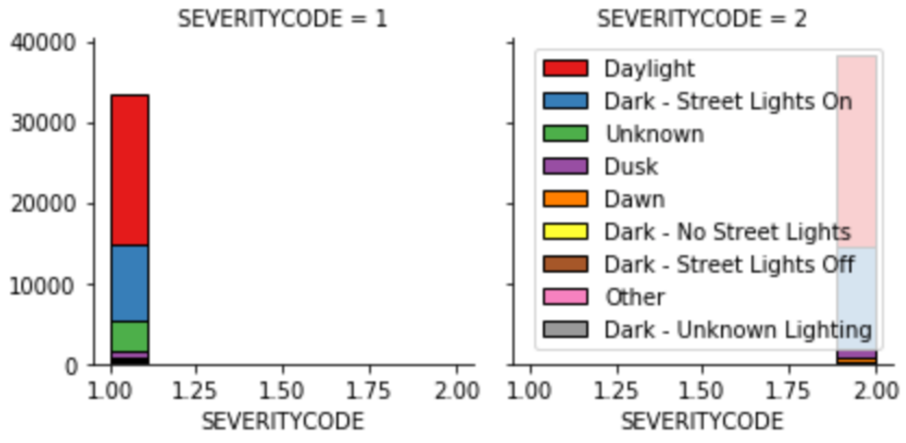


Figure 11. relationship between the light condition and severity of accidents

2.4 Data Normalization

In the last step of the data preprocessing, we normalized the data so that it had zero mean and unit variance. Finally, we split the data into training and test data. The training data was used to build the model while the test data was utilized to evaluate the performance of the trained model. It should be noted that 80% of the data was used for training while 20% of the data was used as the test data.

3. Methodology

We trained three different machine learning models (i.e., K-Nearest Neighbor, decision tree, and logistic regression) using the training data. In order to evaluate the performance of the trained models, we used the test data and then chose the model with the highest accuracy.

3.1 KNN

In order to train an effective KNN model, we needed to find the best *k* to build a model with the highest accuracy. To do so, we looped over the value of *k*, calculated the accuracy, and then picked the value of *k* with the highest accuracy. It is worth mentioning that the test data should NOT be used for finding the best *k*; however, we could split the training data into the training and cross-validation (or test) data to find the best *k*. By doing so, it turned out that the best value of *k* for our data set was 9 with 68.38% accuracy. Figure 12 shows the classification accuracy for different values of *k*.

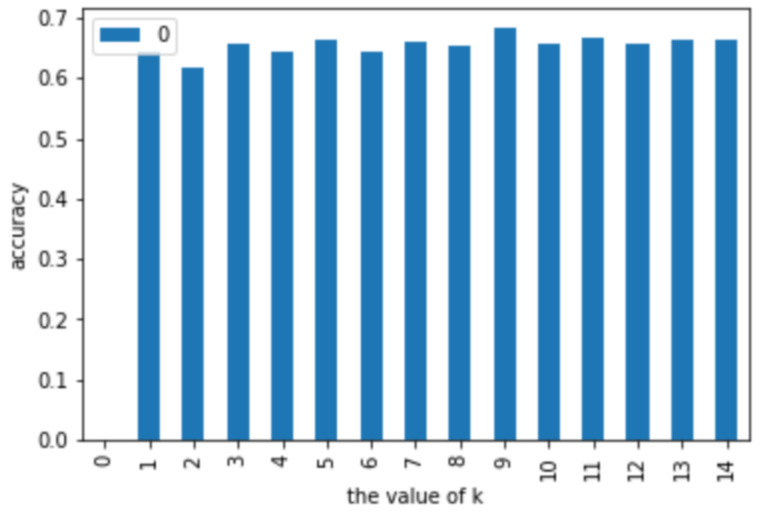


Figure 12. the classification accuracy for different values of *k*

3.1 Decision Tree

In order to train an efficient decision tree model, we needed to find the best depth to build a model with the highest accuracy. To do so, we looped over the value of the depth, calculated the accuracy, and then picked the value of depth with the highest accuracy. Similar to finding the best *k* for KNN, we used the cross-validation data set to find the best value of the depth. It turned out that the best value of depth for our data set was 5 with 70.66% accuracy. Figure 13 demonstrates the classification accuracy for different values of the depth.

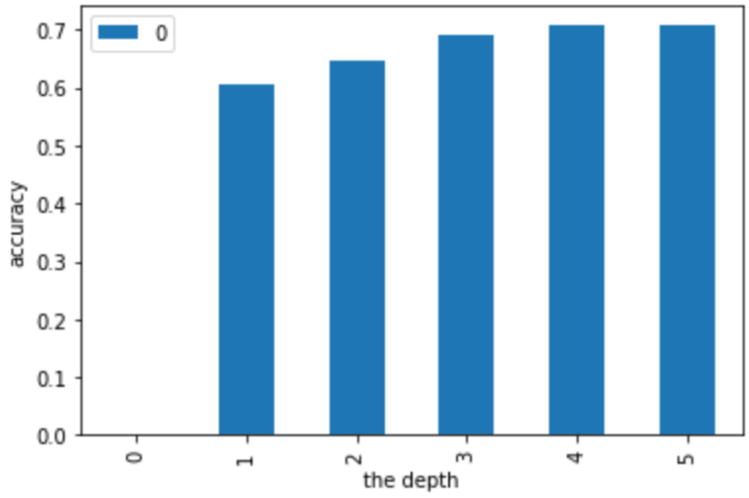


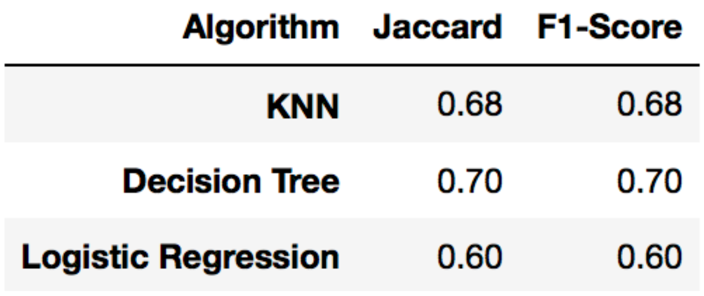
Figure 13. the classification accuracy for different values of the depth

3.1 Logistic Regression

To train a logistic regression model, we set the parameter *C* to 0.01 and tried different solvers (e.g., “lbfgs”, “saga”, “liblinear”, “newton-cg”, and “sag”). By using the “liblinear” solver, the classification accuracy was slightly better on the cross-validation data set, so we picked the “liblinear” solver for our decision tree model.

4. Results

In order to evaluate the performance of the trained model, the test data was used to calculate two evaluation metrics (the Jaccard index and F-1 score). Table 1 shows the values of evaluation metrics for three trained models. As can be seen, the decision three model achieved the highest accuracy compared to the KNN and logistic regression models.



Algorithm	Jaccard	F1-Score
KNN	0.68	0.68
Decision Tree	0.70	0.70
Logistic Regression	0.60	0.60

Table 1. the evaluation metrics for the trained models

5. Discussion

Although the achieved classification accuracy by the decision tree model is 70%, there might be still room for improvement. In the data preprocessing phase, we could try different methods; for example, instead of removing the missing values, we could replace them with an average of similar data points. In the feature selection phase, we could use more independent features to see if it would improve the classification accuracy. To split the data into train/cross-validation/test, we could use *k*-fold method to use the data more effectively. In the modeling phase, we could train other classifiers such as neural networks and support vector machine that might be able to improve the results.

5. Conclusion

In this capstone project, the dataset of car accidents in the Seattle city (from 2004 to present) was used to train a machine learning model with the ability to predict the severity of accidents. Five independent variables (Address type, collision type, weather/road/light conditions) were used to predict the target variable (i.e., the severity of an accident). After preprocessing the data, three classifiers (KNN, decision tree, and logistic regression) were trained and compared using the training/test data. It turns out that the decision tree classifier was able to achieve the highest accuracy.

References

[1] National Highway Traffic Safety Administration (NHTSA): 2015 Motor Vehicle Crashes: Overview (2016).