Predicting the Severity of Car Accidents

Youssef Saleh

September 7, 2020

1. Introduction

1.1 Background

Car accidents without doubt are one of the most tragic incidents that take place repeatedly, they lead to losses in both money and lives. Although many governments have tried hard to prevent accidents from taking place through safety precautions and organizational matters, accidents still take place and results in heavy damages and costly losses. Moreover, most of car companies have developed safety measurements and installed new technologies into the vehicles, which has successfully in recent years reduced the number of casualties during car accidents and also sometimes helped in preventing accidents from taking place in the first place. However, it will be also useful if governments were able to group accidents according to their severity by taking into consideration various variables, this will enable governments to understand the reasons for each accidents group and concentrate the efforts on dealing with these reasons and specially with the most severe accident group. Then it would be beneficial to predict the severity of accidents in the future and try to deal with causes.

1.2 Problem

Dataset that will help in determining the severity of the car accidents will contain various attributes representing many aspects and conditions of car accidents like: weather, road, driver's condition, etc. All of this will help in the aim of this project which is predicting the severity group of car accidents.

1.3 Interests

Obviously, this project and the created model would be very helpful for governments, as they can use it as reference when establishing new roads, reorganizing traffic of the cities or trying to understand the main reasons behind different severity of car accidents to deal with them. Although attributes of accidents differ from country to another, this project illustrates to any country, how to implement and create a model with their own conditions.

2. Data Acquisition and Data Cleaning

2.1 Data

The target of this project is accidents severity, hence a labeled dataset with accidents severity types is needed and of course various attributes to create a more inclusive model. There are many sources to get such dataset, however a sample dataset provided by Corsera will be used during the course of this notebook and can be found here. The dataset consists of all types of accidents in the city of Seattle since the year 2004. It has 38 attributes, 194673 observations and contains attribute for accidents severity (our target). Moreover, the significance of attributes will be assessed and then chosen accordingly. The unused attributes will be dropped out of our dataset and the missing values will be dealt with during preprocessing. Since grouping of already labeled accidents severity is the objective, Classification learning algorithm will be implemented (Supervised learning) to create the intended model.

2.2 Data cleaning

```
print(df['SEVERITYCODE'].value_counts())
df['SEVERITYDESC'].value_counts(normalize=True)
```

```
1    136485
2     58188
Name: SEVERITYCODE, dtype: int64
Property Damage Only Collision    0.701099
Injury Collision                  0.298901
Name: SEVERITYDESC, dtype: float64
```

By examining the target column, it is found out to be grouped into two labeled groups:

- Group1: Property Damage Only collision
- Group2: Injury Collisions

The objective of this project is to create a model to group accidents severity accordingly.

Furthermore, it is clear that our target variable is imbalanced, where more than half of the observations lie in severity group 1 and this could affect our model and make it more biased to the greater class. Hence, we need to figure out an approach to handle such imbalance.

There are many ways we can handle such problem such as:

- Up sampling the minority class (value=2) where we try to replicate random values of smaller class to increase their number and balance with greater class.
- Down sampling the majority class (value=2) where we try to remove random values of greater class to decrease their number and balance with smaller class.
- Use specific classification algorithm, which works well with imbalanced data such as Decision Tree.
- Divide the greater class into two distinct groups, then apply the model on both groups, then evaluate the accuracy of both models and get the average.

During this project last option will be used to balance the target for simplicity and saving time

```
major_class=attributes[attributes['SEVERITYCODE']==1]
minor_class=attributes[attributes['SEVERITYCODE']==2]

#dividing the majority class
major_class_first,major_class_second=train_test_split(major_class,test_size=0.5,random_state=0)

#Models from which we can set our two Y variables (dependant variables) and X variables (independant variables)
model1=minor_class.append(major_class_first)
model2=minor_class.append(major_class_second)
```

On examining the data types of all columns, it was shown that most columns were of type object. Since the target of this project is creating a classification model, all the object values need to be changed to numeric values. This can be done through either get_dummies() method or using LabelEncoder. It is also very helpful to visualize data to have a better understanding of the features, however since the dataset used for this project is mostly of type object (string). Therefore, visualization in this project would be hard to implement.

Furthermore, on checking for null values in the dataset, there was many null values in different columns, which definitely would affect the feature selection phase, as some columns were completely null like EXCEPTRNDESC and EXCEPTRNCODE,

Also, there were columns, which unclear at all like: INATTENTIONIND, UNDERINFL, PEDROWNOTGRNT and SPEEDING. All these columns were a Y/N column however there are

only Y values and missing values, and it would be not perfectly sure to set the missing values to N. Thus, these columns were not used.

Moreover, some columns were not relevant like the X and Y columns as our target is to create a classification model and not a map. In addition, some columns were completely biased to one of its values, hence wasn't used as well.

2.3 Features Selection

Choosing the most suitable features for the model can be hard especially of the dataset used has many attributes, because choosing too many attributes might lead to overfitting, which affects the accuracy of our model.

Moreover, since this is a Car accident severity test then weather, road lighting, road conditions and main idea about the common locations of accidents must be taken into consideration. Also, it will be helpful to check for accidents during different days of the week. Hence, the features to be used in the model are:

- ADDRTYPE
- WEATHER
- LIGHTCOND
- ROADCOND
- INCDATE

Before starting our model, null values need to be dealt with. Since null values are object values then they can be dealt with either:

- Using probability and analyzing to predict the correct missing value like for example if the road is wet, then may be the weather was raining.
- Using the most frequent observation (mode)
- Dropping the rows containing these null values
- Just add 0 on using get_dummies() method.

Since this is a preliminary project, hence for simplicity and maintaining accuracy of our model, rows of missing values in multiple columns will be dropped, while rows with missing values in only one column will remain and get zero on applying get_dummies method().

```
#checking null values
df[['ADDRTYPE','WEATHER','LIGHTCOND','ROADCOND','INCDATE','SEVERITYCODE']].isnull().sum(axis=0)
```

```
ADDRTYPE        1926
WEATHER         5081
LIGHTCOND       5170
ROADCOND        5012
INCDATE            0
SEVERITYCODE       0
dtype: int64
```

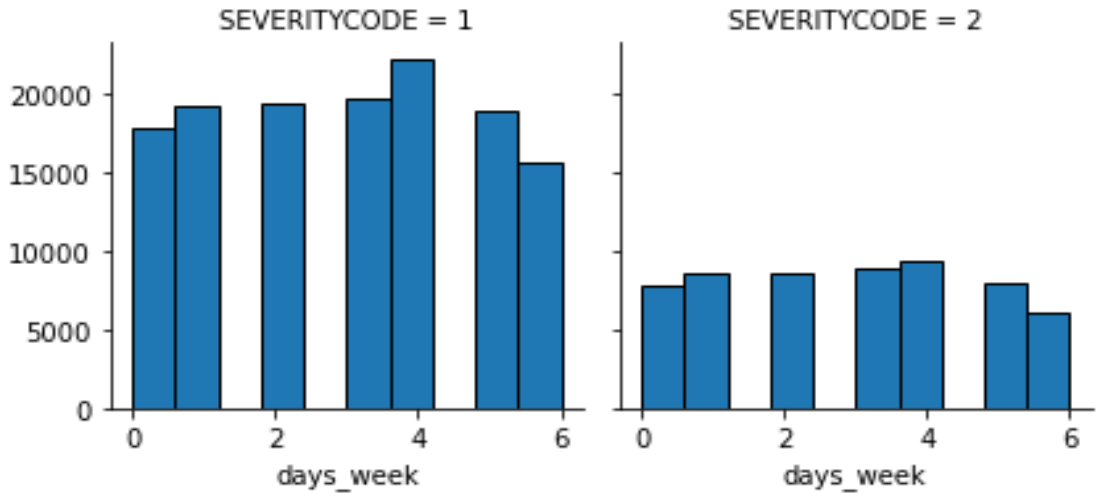
Since the null values in WEATHER, LIGHTCOND, ROADCOND are the most then the rows which have null values in these three columns at the same time will be removed.

On checking the values of LIGHTCOND feature, some values were found having the same meaning like: Dark - No Street Lights, Dark - Street Lights Off and Dark - Unknown Lighting. Hence, they were all combined in one column (Dark), and since the column will be changed to dummy variables Other and Unknown will be dropped.

On inspecting the values of ROADCOND feature, it was found that most of the values are either dry or wet, while other values had really small counts. Thus, on changing column into dummy variables only values wet and dry were used in the features data frame. The same was also applied on WEATHER feature and only values of Clear, Overcast and Raining were added.

A relatively different approach was utilized on converting ADDRTYPE feature to numeric, where since we want to minimize the number of columns to avoid overfitting, it is better to use LabelEncoder than get_dummies(). However, we can't use it while there are null values. So, we can add the value Alley to the null values since it is already too small compared to the others and even by adding null values to it, it will remain too small compared to other values. Hence, it won't affect the overall percentage of distribution much.

Finally, on checking for the severity of accidents during different days of the week.



It is shown on the graphs, that there is a slight increase in the number of accidents after fourth day, hence end of the week. Therefore, a new column was created (weekend) representing if the date is at the beginning or end of the week in the form of binary values.

3. Modeling

Different Classification algorithms would be implemented in this project and their respective accuracy would be calculated and compared.

Since the data has a lot of observations and not to consume too much time we will use:

- Decision Trees
- Logistic Regression

3.1 Decision Tree Algorithm

Decision tree learning is one of the predictive modelling approaches used in statistics, data mining and machine learning. It uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees. Which is the type of trees we intend to create.

```
#model 1 and accuaracy calculations
drugTree_1 = DecisionTreeClassifier(criterion="entropy").fit(X1_train,Y1_train)
Y_hat_model1=drugTree_1.predict(X1_test)
print('Jaccard index for model 1 is', metrics.jaccard_similarity_score(Y1_test,Y_hat_model1))
print('F1_score for model 1 is', metrics.f1_score(Y1_test,Y_hat_model1))

#model 2 and accuaracy calculations
drugTree_2 = DecisionTreeClassifier(criterion="entropy").fit(X2_train,Y2_train)
Y_hat_model2=drugTree_2.predict(X2_test)
print('\nJaccard index for model 2 is', metrics.jaccard_similarity_score(Y2_test,Y_hat_model2))
print('F1_score for model 2 is', metrics.f1_score(Y2_test,Y_hat_model2))
```

As shown in the image the algorithm for decision tree was implemented on both models and two metrics were applied to evaluate both models, where these metrics values will be averaged and inserted in a table in the evaluation section.

It is always preferable to plot the flow chart of the decision trees, as this is the essence and strength of using, which is showing the selection and evaluation order. However, on trying to do so, the result was unclear, duo to it’s huge widens, because a large number of features were used during modeling.

3.2 Logistic regression

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. Although the target of our project is grouping into groups of 1 or 2 and not 0 and 1, this is considered to be binary variable too like Yes/No questions, hence that is a perfect match with the project. In addition, this method enables probability predictions, which is extremely useful on evaluating.

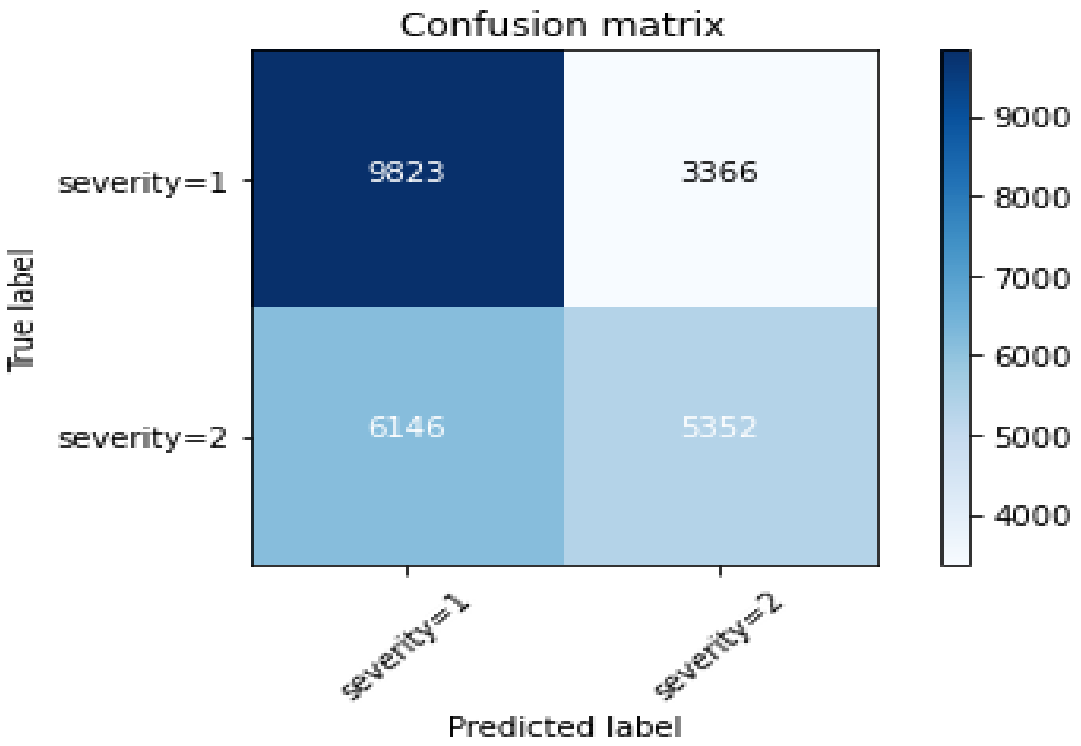
```
#model 1 and accuaracy calculations
LR_model1=LogisticRegression(solver='liblinear').fit(X1_train,Y1_train)
Y_hat_model1=LR_model1.predict(X1_test)
Y_hat_prob1=LR_model1.predict_proba(X1_test)
print('Jaccard index for model 1 is', metrics.jaccard_similarity_score(Y1_test,Y_hat_model1))
print('F1_score for model 1 is', metrics.f1_score(Y1_test,Y_hat_model1))
print('Log Loss for model 1 is ',metrics.log_loss(Y1_test,Y_hat_prob1))
#model 2 and accuaracy calculations
LR_model2=LogisticRegression(solver='liblinear').fit(X2_train,Y2_train)
Y_hat_model2=LR_model2.predict(X2_test)
Y_hat_prob2=LR_model2.predict_proba(X2_test)
print('\nJaccard index for model 2 is', metrics.jaccard_similarity_score(Y2_test,Y_hat_model2))
print('F1_score for model 2 is', metrics.f1_score(Y2_test,Y_hat_model2))
print('Log Loss for model 2 is ',metrics.log_loss(Y2_test,Y_hat_prob2))
```

The same as in decision tree algorithm was carried out on this algorithm, but with one more additional metric which is (log loss), since logistic regression enables probability, then we can evaluate this metric too.

To visualize the accuracy of this model, a confusion matrix was created and plotted, showing numbers of true positives, true negatives, false positives and false negatives. This will be shown during evaluation.

4. Evaluation

On visualizing the confusion matrix for the Logistic regression model, it was found out



The first row is for accidents whose actual severity value in test set is 1. Obviously, out of 24687 accidents, the severity value of 13189 of them is 1. And out of these 13189, the classifier correctly predicted 9823 of them as 1, and 3366 of them as 2. It means, for 9823 accidents, the actual severity value was 1 in test set, and classifier also correctly predicted those as 1. However, while the actual severity of 3366 accidents were 1, the classifier predicted those as 2, which is all in total relatively good. While the second row. It looks like there were 11498 accidents whom their severity value was 2. The classifier correctly predicted 5352 of them as 2, and 6146 of them wrongly as 1. So, it has not done a good job in predicting the accidents with severity value 2. We can consider it as error of model for second row.

Nevertheless, the calculated metrics during both models were averaged and added to the following table.

Algorithm	Jaccard	F1-score	LogLoss
Decision Tree	0.61	0.67	NA
LogisticRegression	0.61	0.67	0.64

The Evaluation of our model resulted in not very good scores, as it is always preferable to have accuracy scores at least above 75%. Besides, the dataset used in this project had a lot of null values, which we don't have enough information to be able to predict them. Moreover, the target variable was greatly imbalanced, which affected our model significantly, although we tried to minimize its effect. Furthermore, many of the data observations were labeled Other and Unknown in some features, thus we had to drop them. Finally, the dataset lacked numeric data, which blocked us from using visualization or correlation in choosing the features.

5. Conclusion and Recommendation

In my opinion more observations need to be collected accurately without Unknown, Other or Null values. Also, it would be very useful to add more relatable numeric attributes like: horse power of the car, speed upon impact and age of the driver. Also starting to collect more information about drivers like gender, vision and social class because sometimes rich people are careless while driving. All of this information would be very helpful as they will allow correlation and visualization testing to take place, in order to find the most adequate features to use and create a more accurate model.

To sum up, I have analyzed the dataset completely, applied an approach to deal with the imbalance in the dataset, chosen the most suitable features covering the circumstances of car accidents, dealt with null, Other, Unknown and values having same meaning, created classification models using two different algorithms and evaluated them using different metrics. The result was that most of the car accidents are of lower severity, however it can still be lethal. We can use our model in learning the circumstances that leads to accidents of different severity especially the higher ones, so that we remain careful and aware while designing new roads and building new cities. We will need to collect data more carefully and add new features like those suggested.