IBM - COURSERA CAPSTONE PROJECT PREDICTING ACCIDENT SEVERITY REPORT

August 30, 2020

1 INTRODUCTION/BUSINESS PROBLEM

Road accidents are very fatal to human lives and the environment. In this project we seek to solve the problem of reducing accidents by predicting the possibility and severity of accidents in terms of human fatality, traffic delay, property damage, or any other type of bad impacts from accidents. This is very useful to humans since once can tell the probability of a road accident occuring and one will take precautions. Therefore decide whether or not to take that road or drive out. This will go a long way to save lots of lives.

This report follows the following layout. In Section 2, I will discuss the data used in this project. I will then throw light on the methodology in Section 3, give details on the results and observations in Section 4 and then finally conclude in Section 5

2 DATA

The data used in this project is the accident severity data for Seattle city. It was downloaded online from the course materials. The data has 37 attributes and a target known as ”severitycode”. Some of these attributes are the weather, the condition of the road at that time of collision, the pesdestrian count on the road, the number of vehicles on the road, if a parked car was hit or not and so on. The data contains 194673 rows.

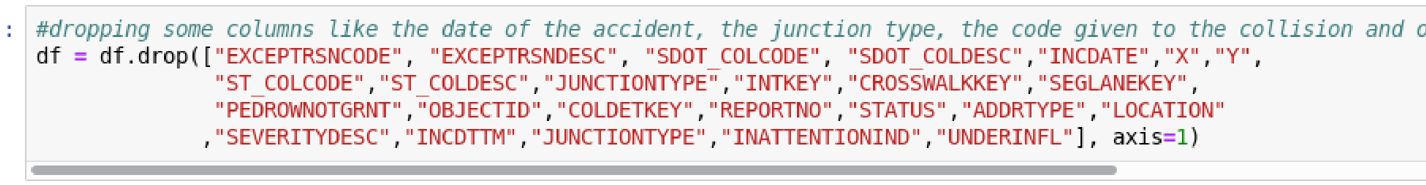
3 METHODOLOGY

In this section, I will discuss the data preprocessing that was done in Subsection 3.1, the type of machine learning that was used and the justifications for using that type in Subsection 3.2.

3.1 DATA PREPROCESSING

Data preprocessing involves all the steps and methods employed in preparing the data sets to be readily used for the model.

In the preprocessing process, I deleted some columns in the datasets which had a lot of empty entries and some of the columns which had no effect on severity of the accident. Some of the deleted columns are the date of the accident, the time of the collison, the type of junction and the code given to the collision type and so on. All the columns deleted can be seen in Figure 1 below:



```
:  #dropping some columns like the date of the accident, the junction type, the code given to the collision and o
   df = df.drop(["EXCEPTRSNCODE", "EXCEPTRSNDESC", "SDOT_COLCODE", "SDOT_COLDESC","INCDATE","X","Y",
                 "ST_COLCODE","ST_COLDESC","JUNCTIONTYPE","INTKEY","CROSSWALKKEY","SEGLANEKEY",
                 "PEDROWNOTGRNT","OBJECTID","COLDETKEY","REPORTNO","STATUS","ADDRTYPE","LOCATION"
                ,"SEVERITYDESC","INCDTTM","JUNCTIONTYPE","INATTENTIONIND","UNDERINFL"], axis=1)
```

Figure 1: Dropping some of the columns

I then looked at the types of the features or attributes for the remaining columns and noted that some of them were categorical variables. These were the weather, road condition, light condition, the type of collision and the hit parked car attributes. I therefore used one hot encoding to convert this variables into dummy variables and multiple binary features to be suitable for the model.

Moreover, given that our datasets had a lot of entries, I deleted all rows with any missing variables and this did not affect the datasets much. After deleting there were still 114837 rows left which is good enough for the model. Also the target variable which is the severitycode column has two class, that is 1 and 2. I replaced the class 1 with a 0 and class 2 with a 1 to make this a binary classification problem. This is now a regression problem with two classes, 0 and 1. I therefore used logistic regression model. More details will be given in the next subsection.

Before training the model, I splitted the data sets into 70% for training and 30% for testing to allow for better evaluation of my model on unseen data sets.

However, the target variable was imbalanced, the first class (class 0) was two times more than that of the second class (class 1). I therefore used a resampling technique (from scikit-learn), that is both upward sampling of the minority class (class 1) and downward sampling of the majority class (class 0) before running the model and then checked the results.

3.2 MACHINE LEARNING TYPE

The logistic regression model was used, this is because our target classes are 0 and 1. Also logistic regression allows for two discrete outputs, which are 0 and 1 (binary classification problem). I used the built in function "LogisticRegression()" to run the model in Scikit-learn. I tried the logistic regression model with and without weights.

4 RESULTS AND OBSERVATIONS

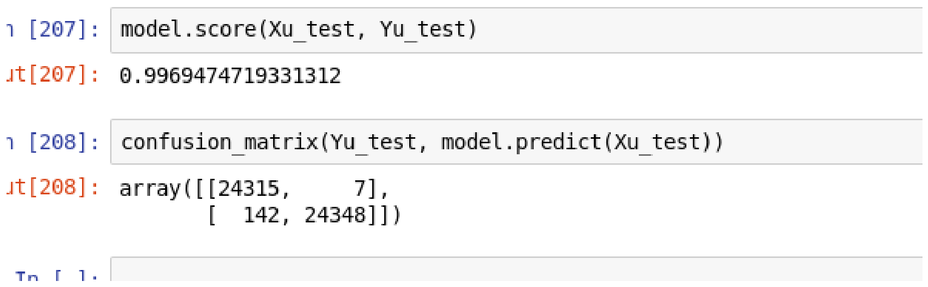
In this section I will discuss the results of the model in Subsection 4.1 and then talk about the observations I made while running the model in Subsection 4.2

4.1 RESULTS

Firstly, I run the logistic regression model without resampling the imbalanced data and my model only predicted the majority class by looking at the confusion matrix even though the accuracy was 71%. This was expected since the data sets was biased. I then decided to add more weights in the same proportion as the imbalanced data sets to the minority class. This however did not help the model in any way. It was still predicting only class 0.

I then downsampled the majority class and that gave better accuracy. When I added weights to the logistic regression model the accuracy improved and the model predicted well on both classes.

Finally, I upsampled the minority class and then added class weights for 60 each for both classes to the logistic regression model. This model was the best for this datasets. It gave the highest accuracy of 99.6% and the confusiom matrix was very good. It predicted 24315 instances of the first class correctly and 24348 instances of the second class correctly. This can be seen in Figure 2 below:



```
ı [207]: model.score(Xu_test, Yu_test)
ıt[207]: 0.9969474719331312

ı [208]: confusion_matrix(Yu_test, model.predict(Xu_test))
ıt[208]: array([[24315,     7],
                [  142, 24348]])
```

Figure 2: Accuracy and confusion matrix

4.2 DISCUSSIONS

I observed that, resampling the imbalanced data and adding weights to the classes improved on the model's predictions. For the downsampling technique as I increased the weights of the classes from 1 to 20 the accuracy and prediction improved but beyond 20, the accuracy begun to fall. Also, for upsampling the class weights beyond 60 for the classes gave us bad predictions and lower accuracy. 60 was the best class weight to use.

5 CONCLUSION

In conclusion, logistic regression with class weights and an upward resampling technique was very good to use for this datasets and problem. It gave us better accuracy and predicted well. With this model, Individuals can therefore predict the severity of an accident given data on the weather conditions, the road conditions, the number of pedestrian and vehicles and so on.

The code can be seen in the following link : https://github.com/sahada19/Coursera_Capstone/blob/master/capstone_notebook.ipynb