Coursera Capstone - Predicting Accident Severity

Arabinda Behera

September 11, 2020

1 Introduction

1.1 Background

The World Health Organization describes the road traffic system as the most complex and the most dangerous system with which people have to deal every day. Millions of people around the world die in road accidents each year. The affected people and their family go through mental stress and face financial problems. Road accidents also puts pressure on police and insurance agencies. They have to be prepared and be efficient in dealing with unexpected cases. The study of road accidents is very crucial and some models for predicting accident severity would be very beneficial for drivers, police, insurance and other agencies to deal with accidents efficiently. It would also be helpful for the awareness of general public on the important causes and patterns and may help prevent accidents to some level.

1.2 Problem Statement

The numbers and the severity of the accidents can depend on several factors like location, time, traffic conditions, weather conditions, state of the vehicle, driver's physical and mental state, etc. Predicting the probability and severity of accidents using simple statistical models is difficult. A huge amount of open access data is available on road accidents from different governments around the world. The important question is can this vast data be used to predict accident severity?

This project aims on developing machine learning models using available and labeled data to predict severity of road accidents.

2 Data

2.1 Data Source

The data for this project is taken from Kaggle (`https://www.kaggle.com/`). The data originally comes from the Open Data website of UK government (`https://data.gov.uk/`) published by the Department of Transport. The dataset contains detailed information on the road accidents from different places in the country. The dataset contains two csv files :

- `Accident_Information.csv`: each row represents a unique traffic accident (identified by the `Accident_Index` column), featuring various properties related to the accident as columns. Date range: 2005-2017

- `Vehicle_Information.csv`: each row represents the involvement of a unique vehicle in a unique traffic accident, featuring various vehicle and passenger properties as columns. Date range: 2004-2016

The project is implemented in Jupyter notebook using python libraries - numpy, pandas, sklearn, matplotlib and seaborn.

2.2 Data merging

The ".csv" data files are loaded in the Jupyter notebook using pandas dataframes. The two data frames were then joined using "inner-join" on the column "`Accident_Index`". The top five rows of the merged dataframe obtained using the head() command of pandas is shown in Figure 1.

Out[3]:

	Accident_Index	Age_Band_of_Driver	Age_of_Vehicle	Driver_Home_Area_Type	Driver_IMD_Decile	Engine_Capacity_.CC.	Hit_Object_in_Carriageway	Hit_Object
0	200501BS00002	36 - 45	3.0	Data missing or out of range	NaN	8268.0	None	
1	200501BS00003	26 - 35	5.0	Urban area	3.0	8300.0	Parked vehicle	
2	200501BS00004	46 - 55	4.0	Urban area	1.0	1769.0	None	
3	200501BS00005	46 - 55	10.0	Data missing or out of range	NaN	85.0	Kerb	
4	200501BS00006	46 - 55	1.0	Urban area	4.0	2976.0	None	

5 rows × 57 columns

Figure 1: First 5 rows in the dataframe

2.3 Data reduction

There are 2,058,408 rows in the dataframe which is quite huge for the analysis. So, the number of rows are reduced by selecting a 30% sample of the data with weights. The columns which are most important features for the analysis are retained and others are dropped. The final dataset now consists of 617,522 rows and 25 columns.

3 Methodology

3.1 Exploratory Data Analysis

The dataframe is explored to make several plots for data visualisatiion. Some of the important ones are discussed below. The target column is "`Accident_Severity`". The value counts of the three values in this columns are shown in Figure 2. The null values in the dataframe are checked for different columns. The list of columns and number of null values are shown in Figure 3. The distribution of the "`Age_of_Vehicle`" column with years as units is shown in Figure 4.. The minimum vehicle age is 1 year, maximum age is 104 years and the median age is 7 years. The distribution of

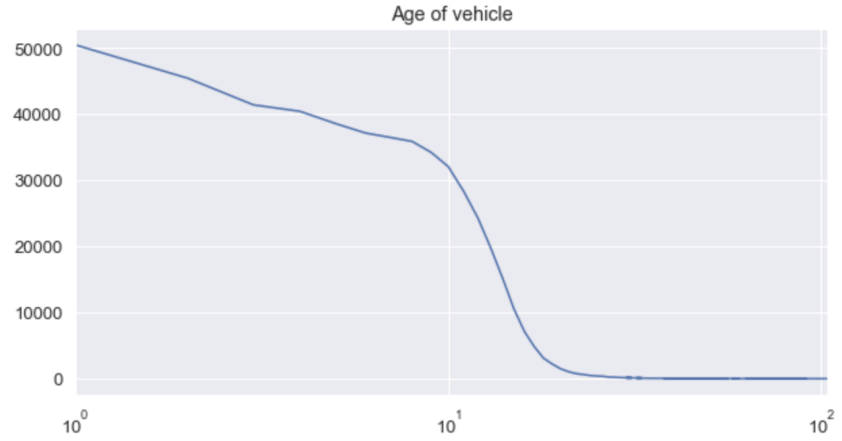
Figure 2: Accident severity counts

```
In [8]: #Null values in the columns
        null_count=df.isnull().sum()
        null_count[null_count>0]
```

```
Out[8]: Age_of_Vehicle                                  102870
        Driver_IMD_Decile                               207055
        Engine_Capacity_.CC.                             76015
        make                                             34408
        model                                            96452
        Propulsion_Code                                  70780
        Vehicle_Location.Restricted_Lane                   307
        2nd_Road_Class                                  263792
        2nd_Road_Number                                   5879
        Did_Police_Officer_Attend_Scene_of_Accident         42
        Latitude                                            28
        Location_Easting_OSGR                               28
        Location_Northing_OSGR                              28
        Longitude                                           28
        LSOA_of_Accident_Location                        44609
        Pedestrian_Crossing-Human_Control                  202
        Pedestrian_Crossing-Physical_Facilities            364
        Speed_limit                                         13
        Time                                                62
        InScotland                                          14
        dtype: int64
```

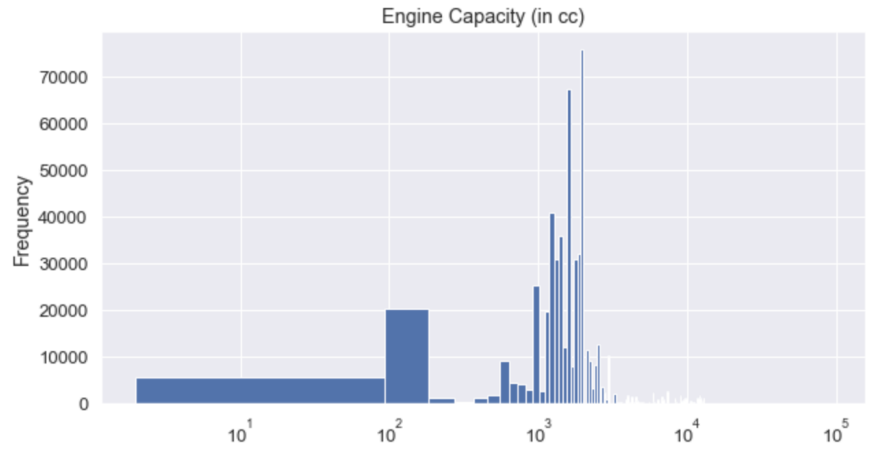
Figure 3: Null values in different features

the "`Engine_Capacity_.CC.`" column with "c.c" as units is shown in Figure 5. The minimum engine capacity is 2 c.c, maximum is 91,000 c.c and the median is 1,598 c.c The heatmap of the correlation matrix from the dataframe is shown in Figure 6. The distribution of the "`Number_of_Vehicles`" column is shown in Figure 7.The minimum number of vehicles involved in accident is 1, maximum is 67 and the median is 2. The distribution of the "`Number_of_Casualties`" column is shown in Figure 8. The




```
Min: 1.0
Max: 104.0
Median: 7.0
```

Figure 4: Age of vehicles in years




```
Min: 2.0
Max: 91000.0
Median: 1598.0
```

Figure 5: Engine capacity in c.c

minimum number of casualties in accident is 1, maximum is 87 and the median is 1. The distribution of the accidents at different times over the day is shown in Figure 9. Minimum number of accidents occured during early morning from 12 am to 5 am. Maximum number of accidents occured in the evening around 5 pm.

```
In [12]: plt.figure(figsize=(10,5))
         sns.heatmap(df2.corr(),cmap='YlGnBu')
         plt.show()
```

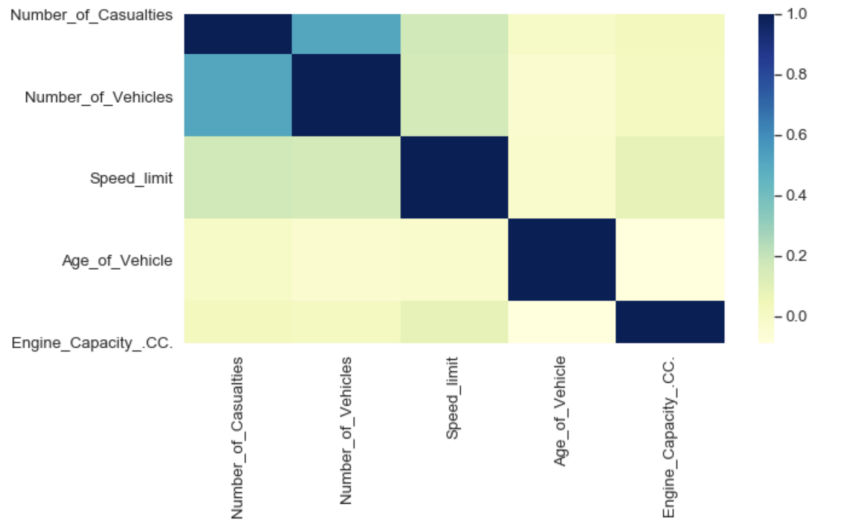
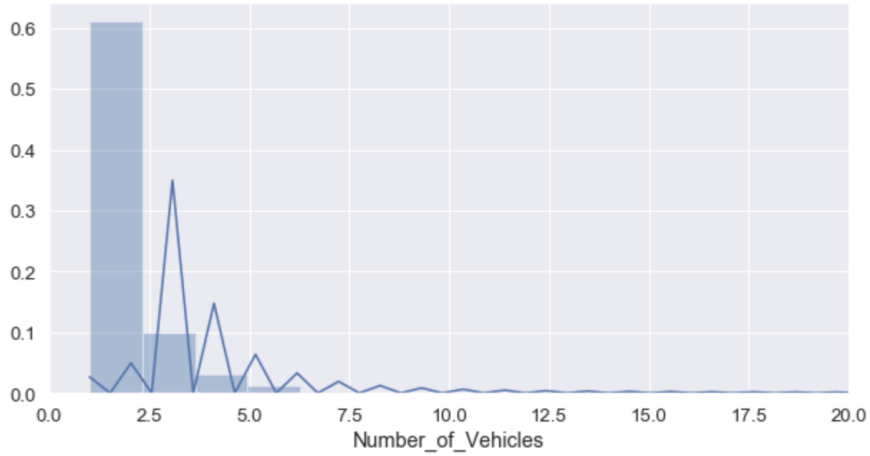
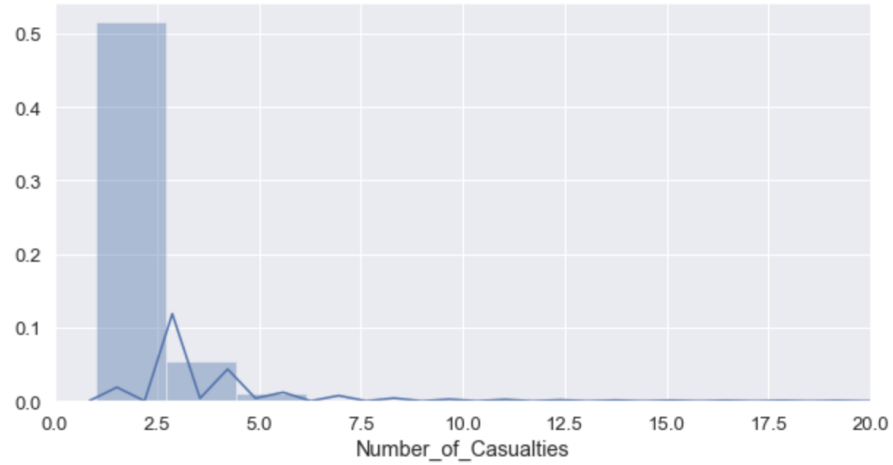


Figure 6: Correlation matrix of features - heatmap




```
Min: 1
Max: 67
Median: 2.0
```

Figure 7: Number of vehicles




```
Min: 1
Max: 87
Median: 1.0
```

Figure 8: Number of casualties

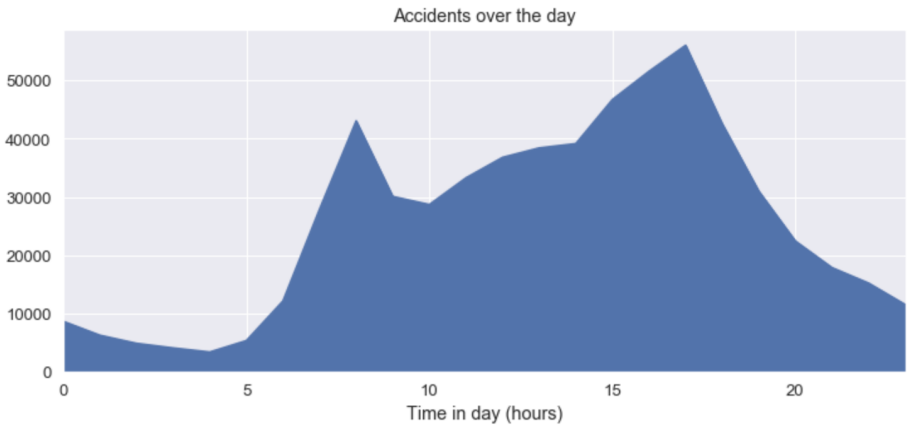


Figure 9: Accidents over different times in a day

3.2 Multi-class to Two-Class Transformation

There are three classes in the column "`Accident_Severity`" namely 'Slight', 'Serious' and 'Fatal'. To make the classification problem simpler I am combining the 'Serious' and 'Fatal' into one class. So, basically one new binary column "`Accident_Severity_Slight`" is created using one-hot encoding with value 1 for 'Slight' and '0' for 'Severe'. The number of casualties distributions by accident severity is shown in Figure 10. The

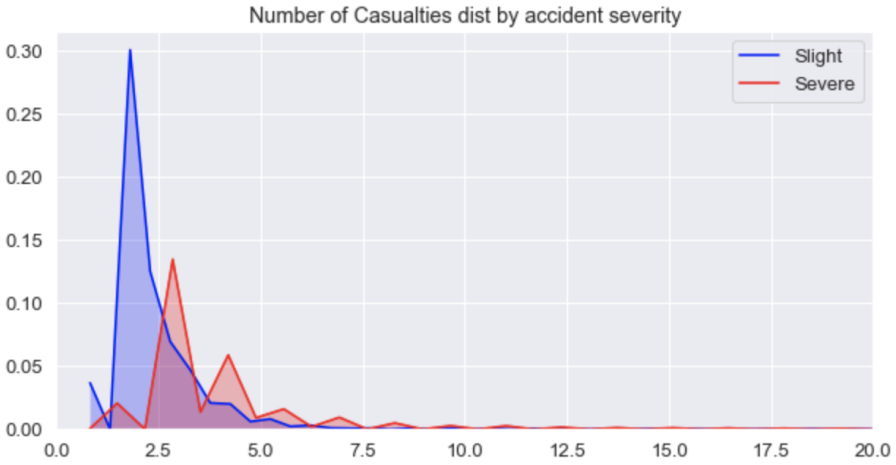


Figure 10: Number of casualties distributions by accident severity

number of vehicles distributions by accident severity is shown in Figure 11.

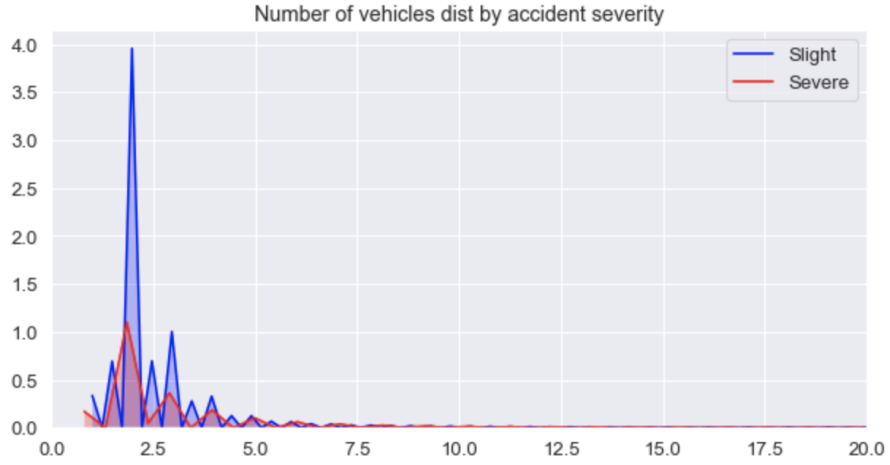


Figure 11: Number of vehicles distributions by accident severity

3.3 Feature-Target Split

The columns are split into features and target. The column "`Accident_Severity_Slight`" is selected as the target and the rest are selected and feature set.

3.4 Pipelines

Pipelines are created to do the following tasks :

- Addressing the null values in the columns
- Applying transformation on some features
- Applying scaling on features.
- Creating One-hot encoding of the features.

Some of these are discussed below.

Filling null values

The null values in the features are filled in the pipelines using the SimpleImputer() function with different strategies. The strategies used for some of the features are given below :

- Speed Limit - '`most_frequent`'
- Age of Vehicle - '`median`'
- Engine Capacity - '`most_frequent`'

Feature Transformations

Different transformations are applied on some of the features to improve their efficiency for the classification. Some of them are given below :

- The feature "Time" is grouped into different classes - [Night, Early Morning, Morning, Midday, Afternoon, Evening, Late Evening]
- In the feature "Make" of the car, the companies with less than 2000 counts are grouped under one make - 'Other'.
- The feature "`Engine_Capacity_.CC.`" is spread over a very broad range. So, the range is rebinned into 7 bins.

One-hot Encoding

One-hot encodings are applied on the categorical features using the OneHotEncoder() function in the pipelines.

3.5 Modeling

The dataset is split into training samples (`X_train` and `y_train`) and testing samples (`X_test` and `y_test`) using the `train_test_split()` function in the ratio 3:1. The feature sets from training and testing samples (`X_train` and `X_test`) are passed through the pipelines for transformation.

Three machine learning models are used for classification of accident severity - Logistic Regression, Random Forest and Boosting Gradient. For the Logistic Regression Classifier the hyperparameter `class_weight="balanced"`. For Random Forest Classifier the hyperparameters `n_estimators=100` and `n_jobs=3`. For the Gradient Boosting Classifier the hyperparameter `random_state=0`. The three Classifiers were trained separately using the `X_train` and `y_train`. The Gradient Boosting Classifier took the longest time to train. After training, the Classifiers were applied on `X_test` to get the predictions `y_pred`.

4 Results

The models were finally evaluated using three evaluation metrics
- `jaccard_similarity_score`, `f1_score` and `roc_auc_score`. The results are presented in Table 1.

Metric	Logistic Regression	Gradient Boosting	Random Forest
Jaccard Similarity Score	0.66	0.67	0.80
F1 Score	0.70	0.76	0.84
ROC-AUC Score	0.71	0.71	0.86

Table 1: Evaluation metrics table for the Classifiers

So from the evaluation of the classifiers it is concluded that the best model in this project is the Random Forest Classifier with 86% `roc_auc_score`.

5 Discussion

The three models are able to do some good prediction of the accident severity. The Random Forest model has the best performance of 86% accuracy which is pretty good. There is scope of improvement for the performance. Only 30% of the dataset is used in the project. The full dataset can give improvement in the performance. A more detailed hyperparameter search for the models using cross-validation can further improve the performance. More advanced models like the Artificial Neural Networks can significantly boost the performance.

6 Conclusion

In summary, Machine learning classification models were constructed in this project to analyze the open data of car accidents provided by the UK government. Data

was loaded using pandas dataframes and steps were taken for data cleaning and data reduction. Important features for the classification problem were selected and a new dataframe was created with them. In the data exploration and visualisation stage several plots were made to understand the features in the data. Several pipelines were created for handling null values in the features and also do feature transformation. The features and target set were then split into training and testing samples in the ratio 3:1. The features train and test sets were passed through the pipelines to apply the transformations. Three classification models - Logistic Regression, Gradient Boosting and Random Forest were trained on the training sample. The trained models were then applied on the test sample to make the predictions. Finally, the models were evaluated using three metrics - Jaccard Similarity Score, F1 Score and RAC-AUC score. From the evaluation scores it is found that the Random model has the best performs with highest RAC-AUC score of 86%. So, we have developed a model that can predict accident severity with 86% accuracy. There is scope of improvement for the performance.