Capstone Project - Car accident severity

Table of Contents

[**Introduction:** 1](#_Toc49298616)

[**Project Objective:** 2](#_Toc49298617)

[**Data Description:** 2](#_Toc49298618)

[**Data understanding** 2](#_Toc49298619)

[**Data Preparation** 2](#_Toc49298620)

[**Methodology** 3](#_Toc49298621)

[**1.** **Obtaining and Viewing the Data** 3](#_Toc49298622)

[**2.** **Pre-processing the Data** 4](#_Toc49298623)

[**2.1 Handling Date and Time** 4](#_Toc49298624)

[**2.2Handling Missing Values** 5](#_Toc49298625)

[**2.3 Merging Dataframes** 6](#_Toc49298626)

[**2.4 Handling Numerical Data** 6](#_Toc49298627)

[**2.5 Handling Categorical Data** 7](#_Toc49298628)

[**3.** **Modelling the Data** 8](#_Toc49298629)

[**3.1Train-Test-Split** 8](#_Toc49298630)

[**3.2 Handling Imbalanced Classes** 8](#_Toc49298631)

[**3.3 Training and Evaluating Random Forest Classifier with class weight** 9](#_Toc49298632)

[**3.4 Training and Evaluating Random Forest Classifier with SMOTE** 9](#_Toc49298633)

[**Results and Discussion** 10](#_Toc49298634)

# **Introduction:**

In this capstone I will explore and analyze a dataset collected about traffic accidents across the UK. The UK government collects and publishes (usually on an annual basis) detailed information about traffic accidents across the country. This information includes, but is not limited to, geographical locations, weather conditions, type of vehicles, number of casualties and vehicle manoeuvres, making this a very interesting and comprehensive dataset for analysis and research.

The data for this project is available on Kaggle as UK Road Safety: Traffic Accidents and Vehicles (<https://www.kaggle.com/tsiaras/uk-road-safety-accidents-and-vehicles>)

The traffic accidents have a countless effect on the society due to mortality rate and damages due. The traffic accidents are one of the leading causes of death due to injury. In this capstone attention will be paid to identify factors affecting the road accident and severity of these factors. Applying machine learning model traffic accident data records can help to understand the characteristics of driver's behaviour, roadway condition and weather condition that were causally connected with different injury severity.

# **Project Objective:**

The objective of our project is as follows:

* Find the factors affecting the road accident severity in UK.
* Through visualizations and machine learning algorithms, build a model to predict the seriousness of road accident based on Weather Conditions and other important variables as accurately as possible.

# **Data Description:**

* The data for this project is obtained from a user on Kaggle and was composed from information on the United Kingdom’s Government Open Data website.
* It consists of two different datasets that contain information from 2005 2017 that were combined on a common field (Accident\_Index).
  + Vehicle\_Information.csv: A file containing information about the vehicles, point of impact, maneuvers made, driver information, etc.
  + Accident\_Information.csv: A file containing details about the accident that include location, junction details, date

## **Data understanding**

In this phase, dataset will be collected and extracted from a source’s csv file. Then, attributes (columns) will be selected that will use to train your machine learning model. Also, assess the condition of chosen attributes by looking for trends, certain patterns, skewed information, correlations, and so on.

## **Data Preparation**

The data preparation includes all the required activities to construct the final dataset which will be fed into the modelling tools. Data preparation can be performed multiple times and it includes balancing the labelled data, transformation, filling missing data, and cleaning the dataset.

# **Methodology**

Here I will discuss the methodology used in this capstone containing main components, *Data Collection*, *Data Understanding*, *Data pre-processing*, *Modelling*, and *Evaluation and Testing*. This methodology consists of the following main steps:

1. Obtaining and Viewing the Data
2. Pre-processing the Data
   1. Handling Date and Time
   2. Handling Missing Values
   3. Merging Dataframes
   4. Handling Numerical Data
   5. Handling Categorical Data
3. Modelling the Data
   1. Train-Test-Split
   2. Handling Imbalanced Classes
   3. Training and Evaluating Random Forest Classifier with class weight
   4. Training and Evaluating Random Forest Classifier with SMOTE

## **Obtaining and Viewing the Data**

I downloaded the two data set in csv form from Kaggle as UK Road Safety: Traffic Accidents and Vehicles (<https://www.kaggle.com/tsiaras/uk-road-safety-accidents-and-vehicles>). First data set is about Vehicle\_Information and second data set is about Accident\_Information.

We read the dataset that we collect about UK Road Safety: Traffic Accidents and Vehicles into a pandas’ data frame and display the first 5 rows of it as follows in figures 1 and 2:

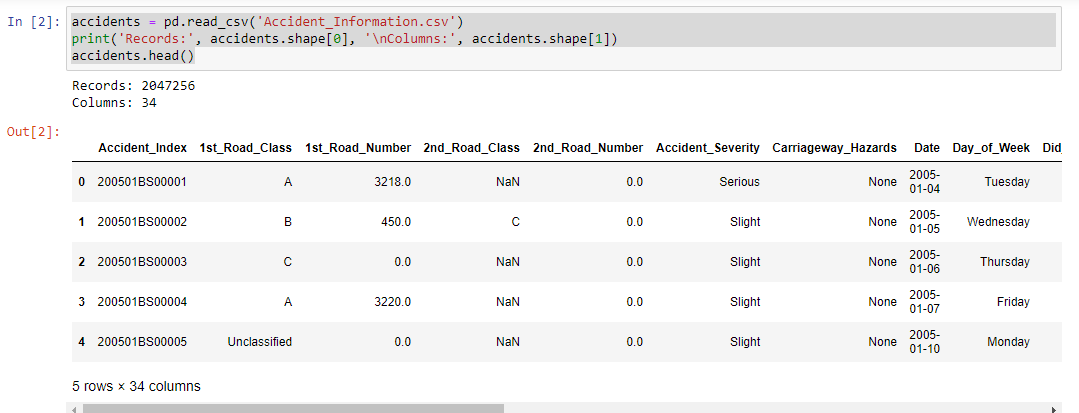


Figure 1Accident Table

Accidents table has Records: 2047256 and Columns: 34

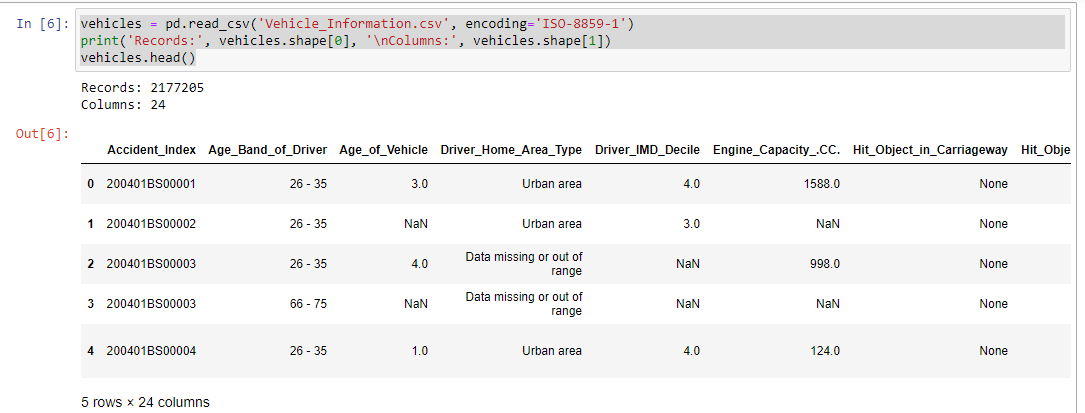


Figure 2Vehicles Table

Vehicles table has Records: Records: 2177205 and Columns: 24

## **Pre-processing the Data**

The collected data are not ready for the analysis approach and need to be explored and organized. we have to clean the data for the modelling process where we choose the machine learning algorithms.

### **2.1 Handling Date and Time**

The Date column with values not properly stored which need to be processed in the correct format as shown in figure 3.

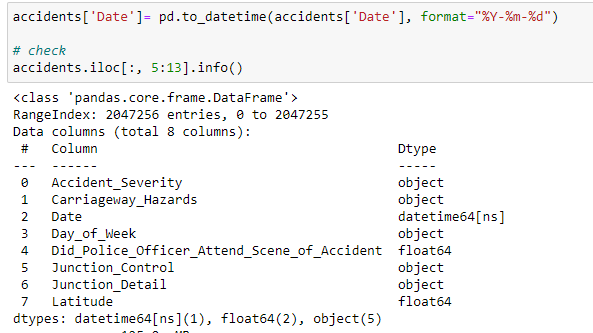


Figure 3 date format

Next, let's define a new column that groups the Time the accidents happened into one of five options:

Morning Rush from 5am to 10am --> value 1

Office Hours from 10am to 3pm (or: 10:00 - 15:00) --> value 2

Afternoon Rush from 3pm to 7pm (or: 15:00 - 19:00) --> value 3

Evening from 7pm to 11pm (or: 19:00 - 23:00) --> value 4

Night from 11pm to 5am (or: 23:00 - 05:00) --> value 5

*For more details please refer notebook Capstone-Week 2.ipynb*

### **2.2Handling Missing Values**

Proportion of missing values in accidents table will be addressed here as shown in figures 4 and 5:

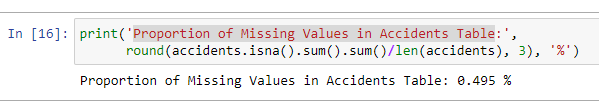


Figure 4Handling missing values in accident table

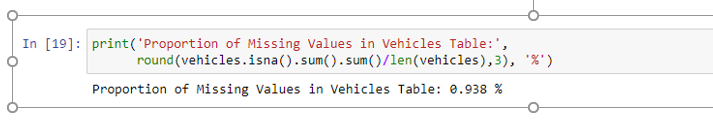


Figure 5Handling missing values in vehicles table

### **2.3 Merging Dataframes**

Here the accidents with the vehicles tables will be merged as shown in Figure 6.

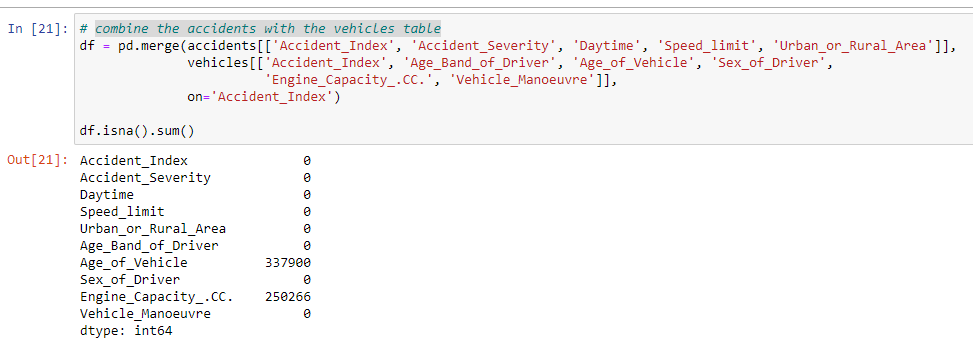


Figure 6 merging two tables

### **2.4 Handling Numerical Data**

In this part outliers will be identified and handled as shown in Figure 7.



Figure 7Outliers

Feature Scaling: The distance-based algorithm won’t work properly for the numerical features’ range with high variations unless the range of all features is normalized.

Tree based models, which will be used here, are not distance based and can handle varying ranges of features. Therefore, scaling is not required.

### **2.5 Handling Categorical Data**

A lot of important information are hidden in the Categorical variables in the given data set. Therefore, it is important to deal with categorical variable by encoding them .as shown in Figure 8.



Figure 8 Handling categorical variable

## **Modelling the Data**

### **3.1Train-Test-Split**

Split arrays or matrices into random train and test subsets [ [1]

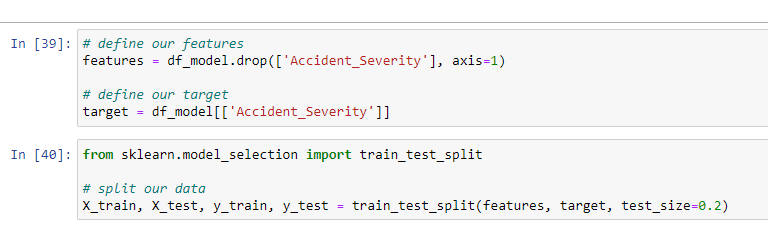


Figure 9Train-Test-Split

### **3.2 Handling Imbalanced Classes**

We can't apply the best strategy and simply can't collect more data, especially from the minority class, we need to find other ways to deal with imbalanced classes.

A second strategy is to use a model evaluation metric better suited to imbalances classes: confusion matrices, precision, recall, F1 scores, or ROC curves instead of accuracy.

A third strategy is to use the class weighing parameter included in implementations of some models. This allows us to have the algorithm adjust for imbalanced classes.

The fourth and fifth strategies are related: downsampling and upsampling. Several of these resampling strategies are well summarized in this [blog post by Chris Remmel](https://calremmel.github.io/fraud-detection-part-one.html) [ [2] .

Here main focus on class **weight parameters**: The Random Forest Classifier will be used which is a popular classification algorithm and includes a class\_weight parameter.

### **3.3 Training and Evaluating Random Forest Classifier with class weight**

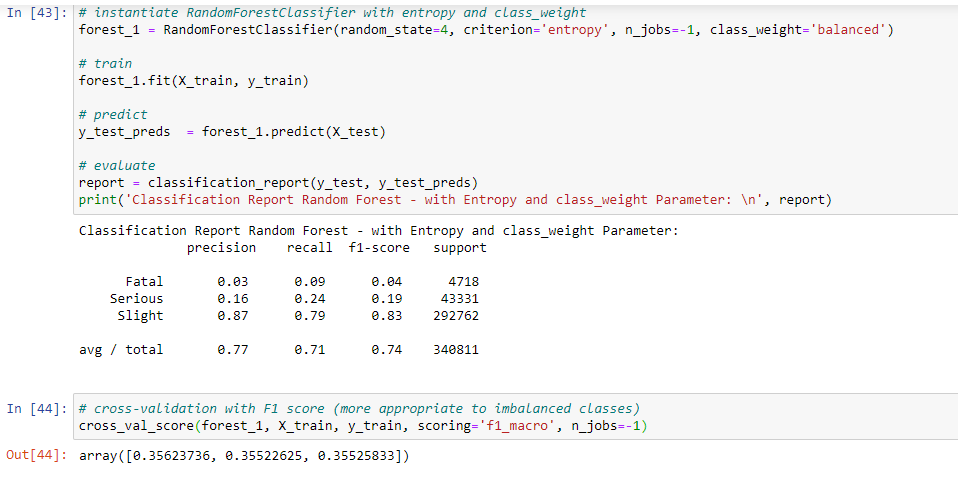


Figure 10 Random Forest with Class\_Weight

### **3.4 Training and Evaluating Random Forest Classifier with SMOTE**

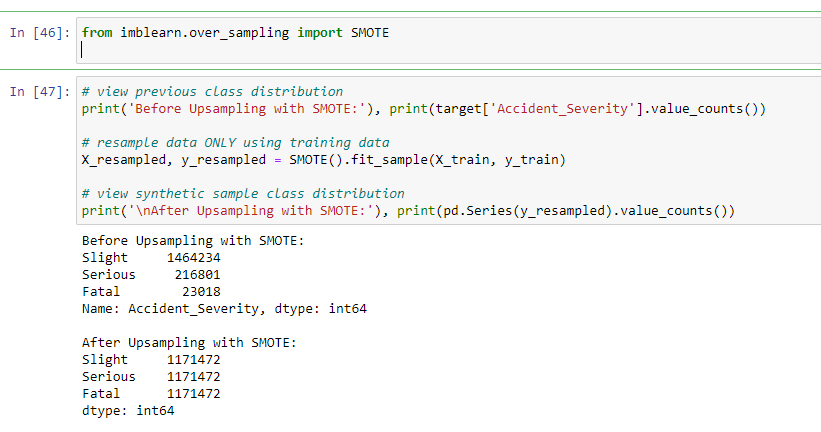


Figure 11Random Forest with SMOTE

# **Results and Discussion**

In this section, we can discus some results that we have got from the analysis and modelling sections. We have started by examining the Accident\_Severity that we have in the dataset. We found as shown in Figure 12.

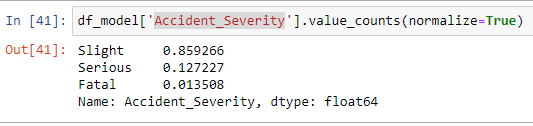


Figure 12 Accident Severity

The result of Random Forest Classifier algorithm with a class\_weight parameter is shown in Figures 13 and 14.

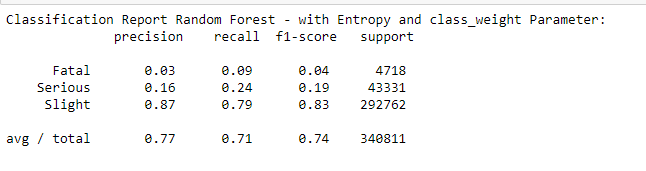


Figure 13 f1-Score of Random Forest with Class\_Weight

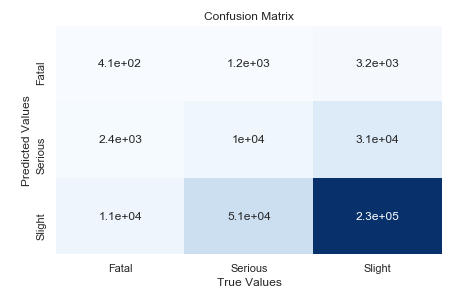


Figure 14 Confusion Matrix

The Random Forest using the weight\_class parameter did not perform very well on classifying the severity. So, let's try one of the resampling strategies to deal properly with our imbalances target classes: **Synthetic Minority Over-sampling Technique (SMOTE)**. Here we're repeatedly sample with replacement from the minority class to make it of equal size as the majority class. To be more specific: We're creating new synthetic data for the minority class - that is representative but not exact duplicate - using K-Nearest Neighbors.

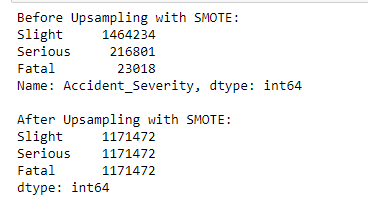


Figure 15 accident Severity Using Random Forest SMOTE

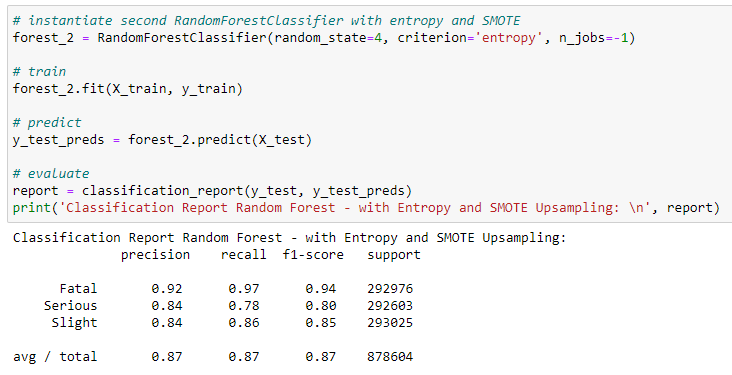


Figure 16 f1-Score Random Forest With SMOTE

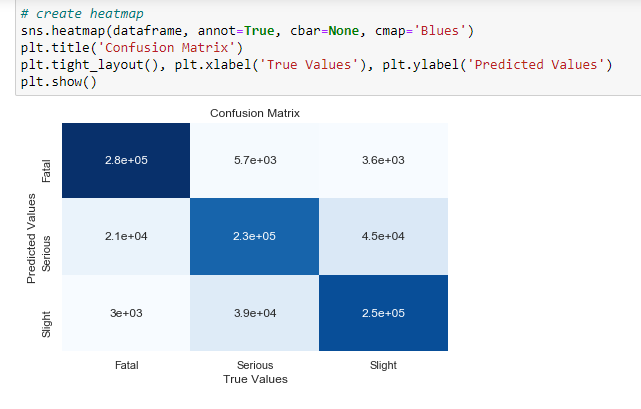


Figure 17 Confusion Matrix Random Forest with SMOTE

It can be observed that Random SMOTE f1-score is better than the other method.

**References**

|  |  |
| --- | --- |
| [1] | "sklearn.model\_selection.train\_test\_split," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model\_selection.train\_test\_split.html. [Accessed 26 08 2020]. |
| [2] | C. Remmel, "Fraud Detection in Python: Part One," 04 2019. [Online]. Available: https://calremmel.github.io/fraud-detection-part-one.html. [Accessed 26 08 2020]. |