

AY: 2020/21

CIS7031 – Programming for Data Analysis

Term 2

Student Name : Sandra Sai Kumar

Student ID : 20193067

Course : MSc Data Science

Module Leader : Dr Ambikesh Jayal

Module Tutor : Ali Shahaab-PhD

Assessment : Programming for Data Analysis

Assessment ID : WRIT1

Allotted Dataset : Police Stop and Search (Metropolitan police)

Table of Contents

[**Abstraction:** 3](#_Toc73008326)

[**Introduction:** 3](#_Toc73008327)

[**Data collection:** 4](#_Toc73008328)

[**Data exploration:** 5](#_Toc73008329)

[**Data preparation:** 6](#_Toc73008330)

[**Data cleaning** 6](#_Toc73008331)

[**Working with the null values:** 6](#_Toc73008332)

[**Data pre-processing** 9](#_Toc73008333)

[**Feature selection:** 9](#_Toc73008334)

[**Label Encoder:** 9](#_Toc73008335)

[**Ordinal Encoder:** 10](#_Toc73008336)

[**Correlation:** 11](#_Toc73008337)

[**Developing prediction model & Reviewing of techniques used:** 12](#_Toc73008338)

[**Hyper-parameter Optimization (Hyperopt):** 12](#_Toc73008339)

[**GridSearchCV for model tuning:** 14](#_Toc73008340)

[**Building a model using RandomforestClassifer Algorithm:** 15](#_Toc73008341)

[**Interpretation of results:** 16](#_Toc73008342)

[**Comparing the results with removing all the null values:** 16](#_Toc73008343)

[**Prediction values after using both parameters:** 18](#_Toc73008344)

[**Comparison of average training and testing errors:** 18](#_Toc73008345)

[**Confusion Matrix:** 19](#_Toc73008346)

[**Saving the decision trees generated by the random forest model:** 19](#_Toc73008347)

[**My Reflection:** 21](#_Toc73008348)

[**References:** 22](#_Toc73008349)

[**Appendixes:** 23](#_Toc73008350)

# **Abstraction:**

**Police Stop and Search** dataset is collection of data which is associated with the search operations performed in the England and Wales done by Metropolitan police under Police and Criminal Evidence Act 1984 and the Police Act 1996.

In this data set we deal with the categorical data which are related to each other to perform **Random forest classification** to make analysis so that our model can make predictions on the given input data which is trained on and also provide some interactive visible insights to the user.

# **Introduction:**

Police stop and search data set is the collection of huge number of data which includes the information of **'Type', 'Date', 'Part of a policing operation', 'Latitude', 'Longitude', 'Gender', 'Age range', 'Self-defined ethnicity', 'Officer-defined ethnicity', 'Legislation', 'Object of search', 'Outcome'** were each columns has its own meaning full data which are interlinked to each other so that we can make a model which takes some of them as input(X) and predict the output(y) which is target variable.

**To complete this task I have used the following:**

|  |  |
| --- | --- |
| **IDE** | **Jupyter note book** |
| **Language** | **python** |
| **Algorithm** | **RandomForestClassifier** |
| **Libraries** | **Pandas, NumPy, sklearn, plotly, missingno,**  **matplotlib, seaborn** |

**Jupyter** is a development environment which supports the python language in development.

**Python** is the widely used programming language which is used in data exploration and analysis mainly in the domain of machine learning and data science related aspects.

**RandomForestClassifier** is a collection of decision tree algorithms. It is an extension of bootstrap aggregation (bagging) of decision trees and can be used for classification and regression problems.

**Pandas** is a powerful library which helps to deal with the files or data of different formats.

**NumPy** is a numerical python which helps in performing the mathematical operations on data.

**Sklearn** is a scientific library which is a collection of different machine learning algorithms.

**Plotly, Matplotlib, Seaborn** is a visualization library which help the developer to develop the interactive insights of visualisations so that anyone can easily understand the data.

**Missingno** is used to represent the missing values in the dataset in matrix or heatmap etc.

# **Data collection:**

**Data** **collection** plays the key role in performing the data analysis, Data is of different formats from the different sources ex: Text, Binary, Images, Audios etc

In this analysis I have allotted with the dataset which belongs to **Police Stop and Search** of the UK government. Which has been downloaded from the source <https://data.police.uk/data/>.

Where I have taken data of 4 months from January -2020 to April -2020 which includes the stop and search data of the metropolitan services.

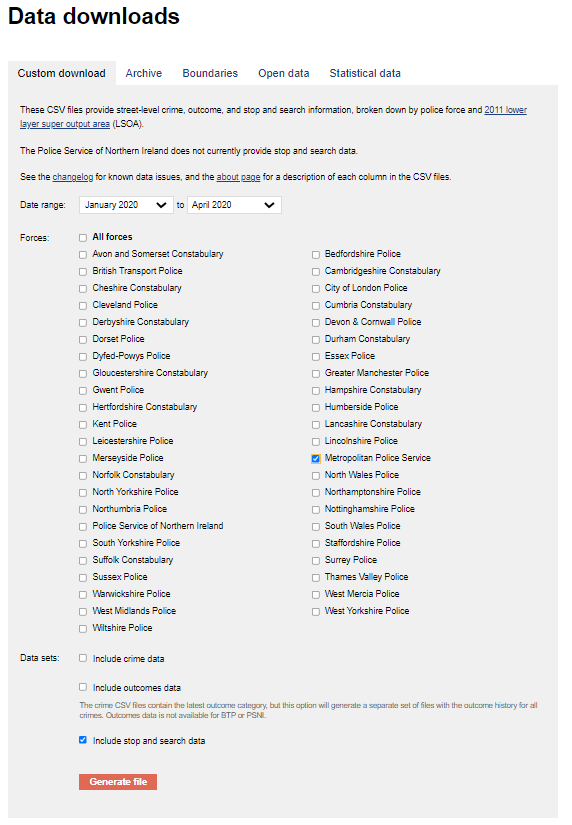


Fig: downloading data set

# **Data exploration:**

We have to perform **Exploratory Data Analysis** on the dataset so that we get the clean data to train the model.

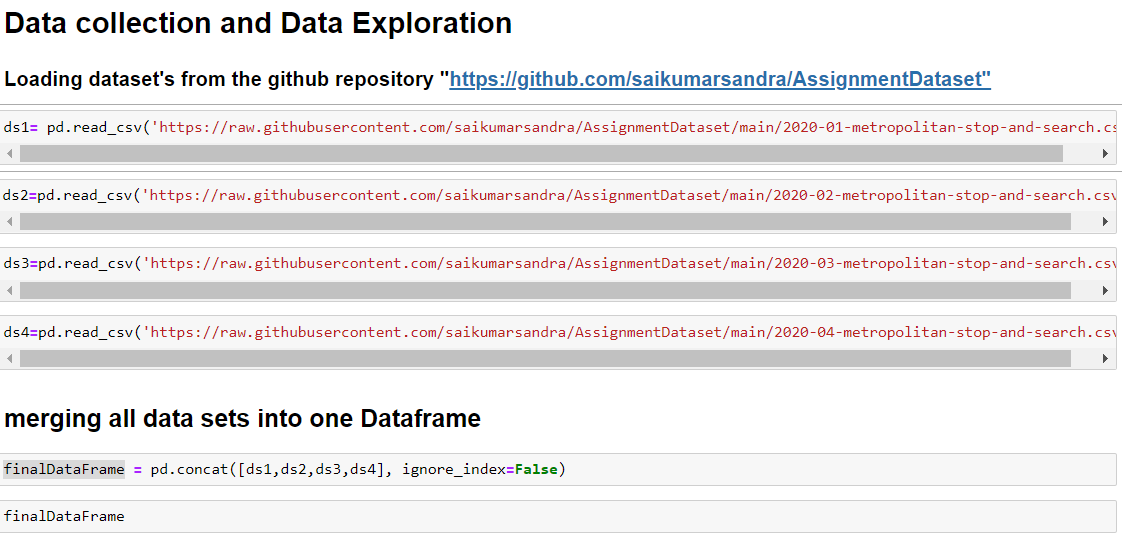
This includes the loading the data into the data frame and getting the data information of data type of the columns, dealing with the Null values, merging different data sets into the single data frame.

Fig: loading data sets

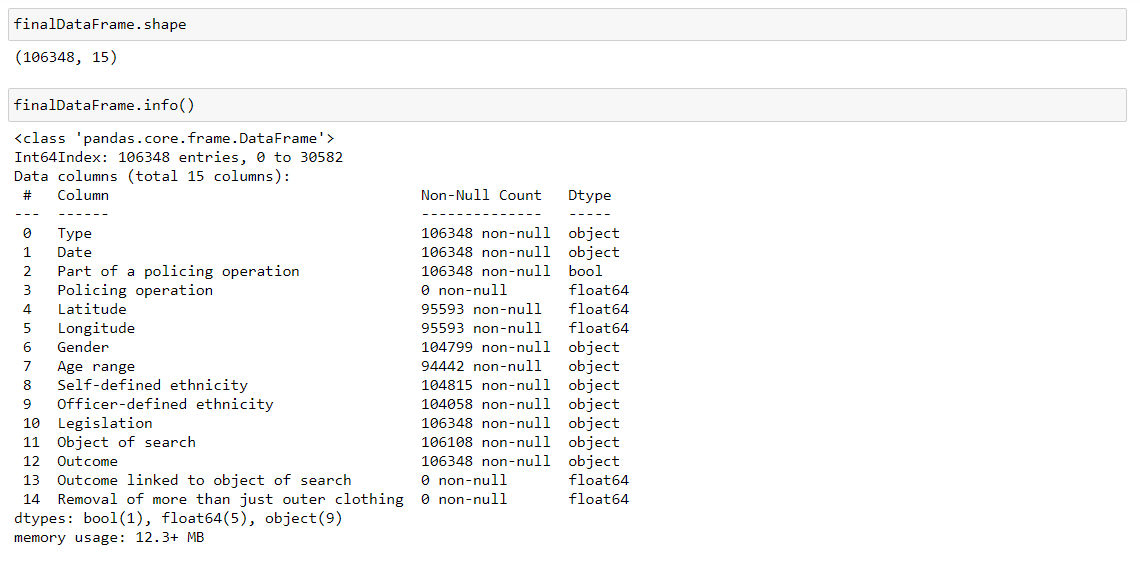
In above pick we have loaded the data set into the **Jupyter** notebook environment and merging all the data-frames into single data-frame.

Fig: info of the data set

**.shape ()** –method is used to get the “rows and columns” of the data-frame.

**.Info ()-** is used to get the complete information of the day data-frame which includes information of the column names and their data type and also with the count of null values.

# **Data preparation:**

Data preparation is the next important step in the data analysis which is done in the following 2 steps: -

1. **data cleaning**
2. **Data preprocessing**

## **Data cleaning**

Data Cleaning means making data-set clean from all types of unwanted data, misspelled data, Null values and making it into the meaning full data set so that we can able to train the model and predict the values more accurately. Now we will have look at some data cleaning operations and techniques.

Working with the null values is the major task for every data science enthusiast as the null values in the data set can cause a big trouble to the model to run the algorithm and make prediction.

Null values can also affect the performance and accuracy of the model.

### **Working with the null values:**

We use **missingno** library to visualize the missing values in the data-set.



Fig: In the above pic white spaces represents the missing values in the data-set.

**Data cleaning** can be done by using some built-in methods or by removing them completely from the data-set.

Some of the built-in methods or as follows:

1. **.fillna ()** – fill the missing values with the desired values.
2. **.replace ()** – replace the values with the methods like **‘pad’,’ffill’,’bfill’**.
3. **.dropna ()** – remove all the null values from the data-set.

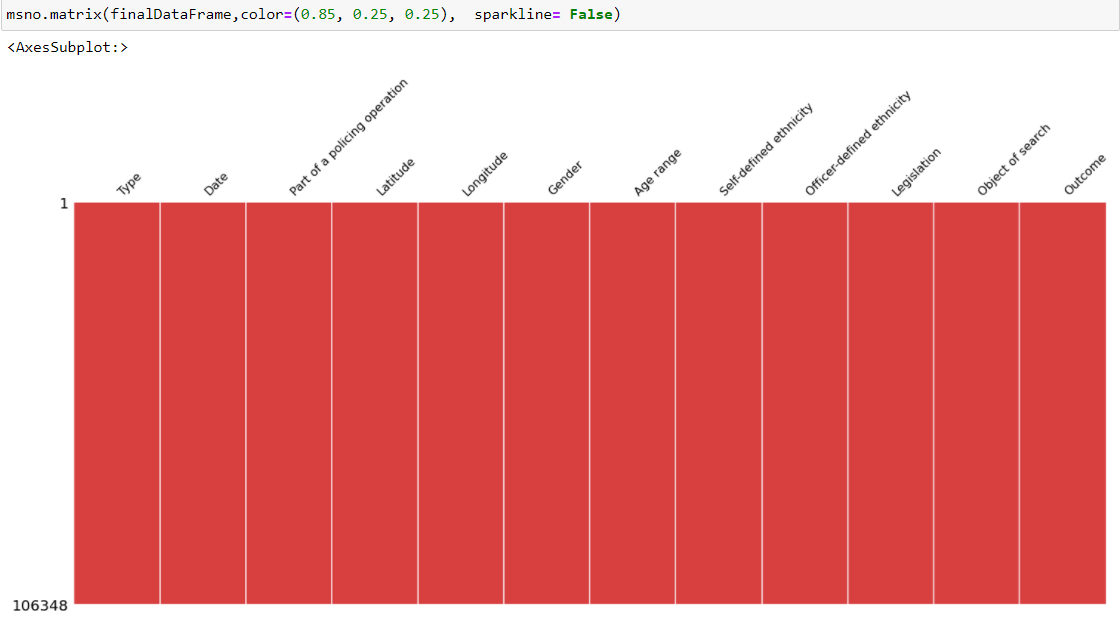
methods to deal with the missing values so that we can improve the accuracy of model prediction as every model cannot accept the missing values after performing the Data cleaning methods using “**.replace()** “we get clean dataset as follows:

Fig 5: clean data set

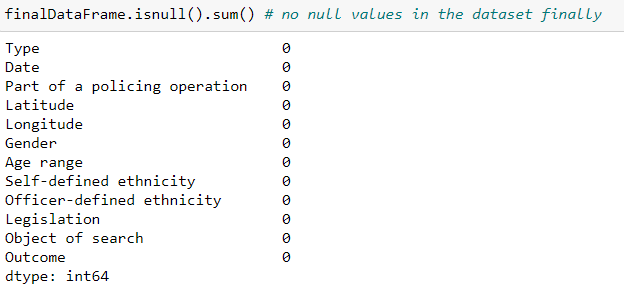


Fig: zero null values in the data set

**Some visual insights of the dataset using plotly:**

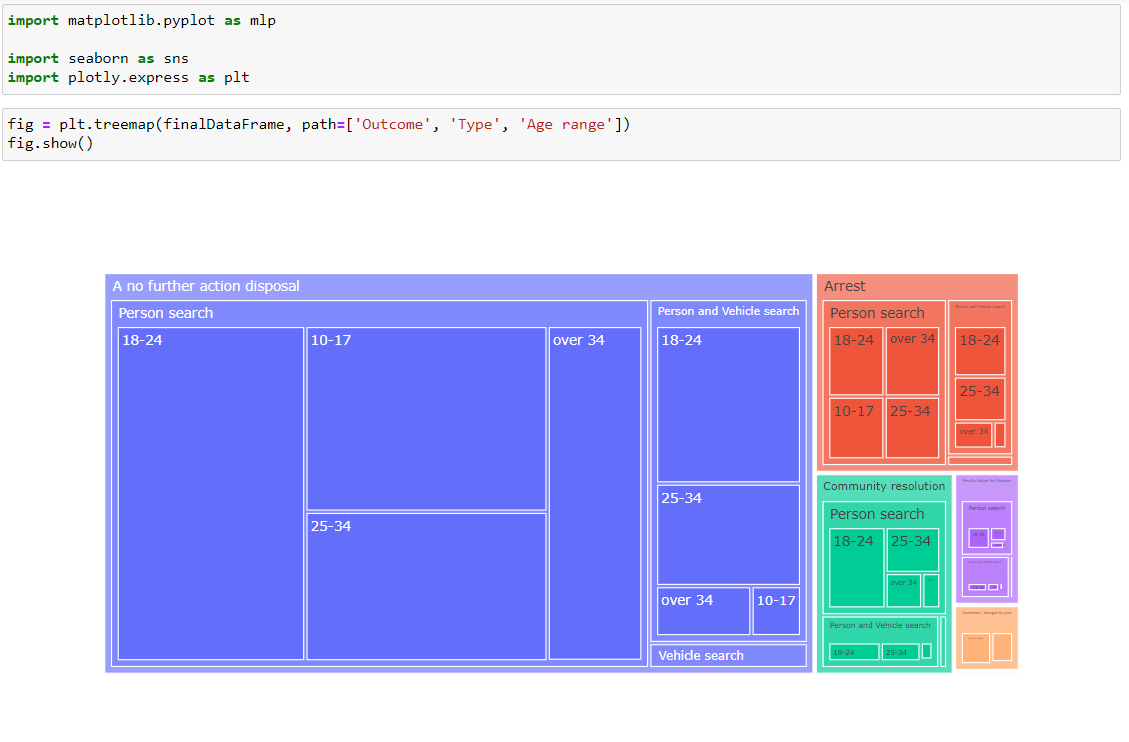


Fig: Tree map

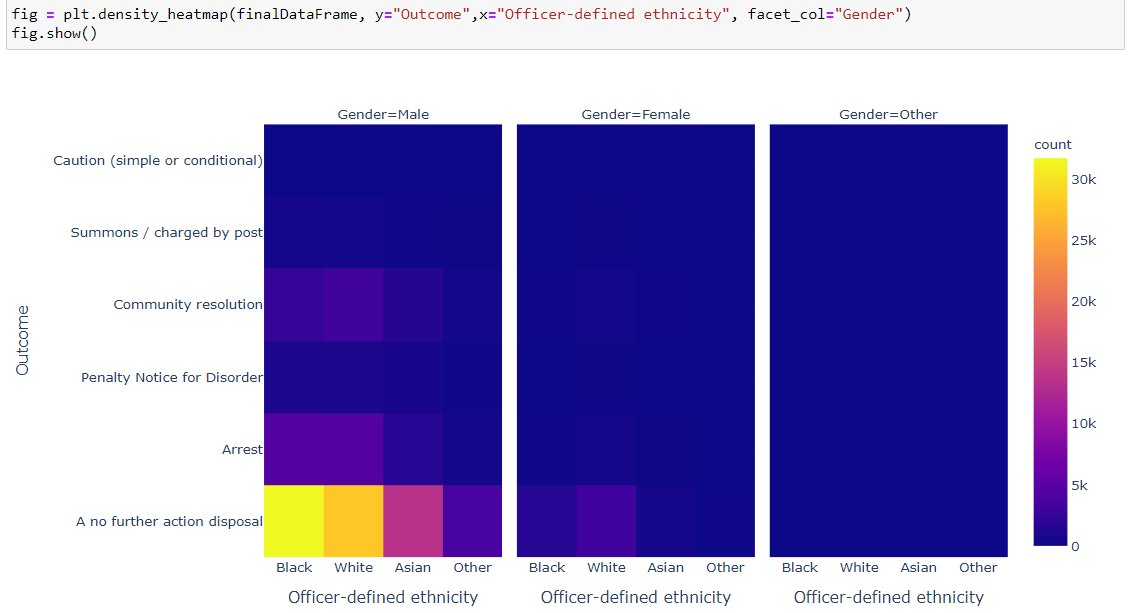


Fig: density heatmap

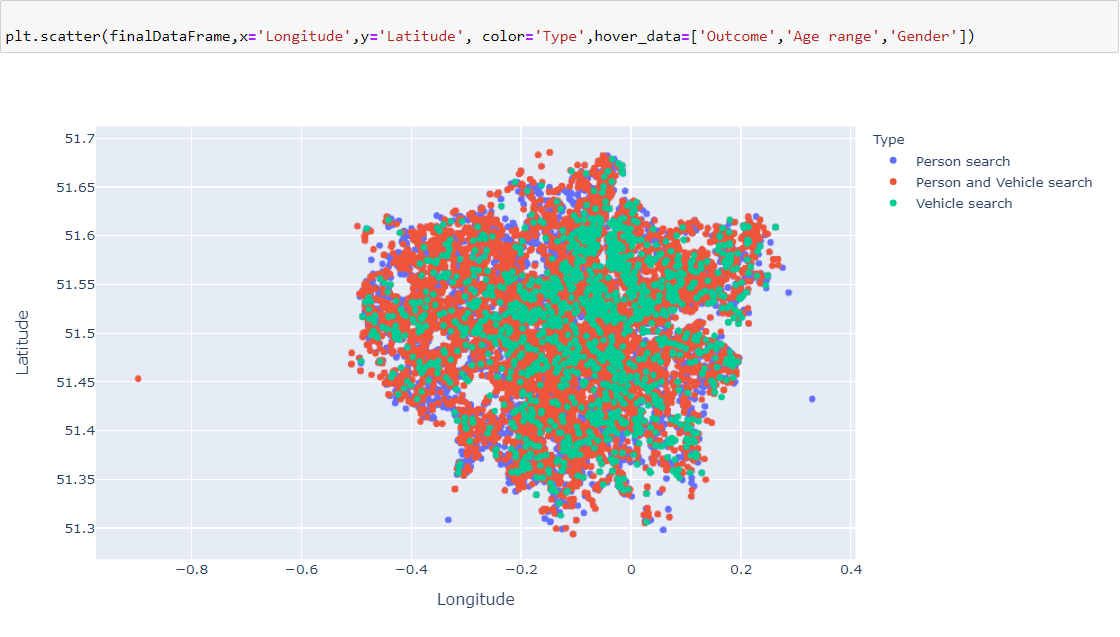


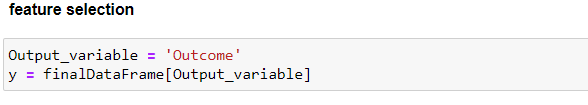
Fig: scatter plot

## **Data pre-processing**

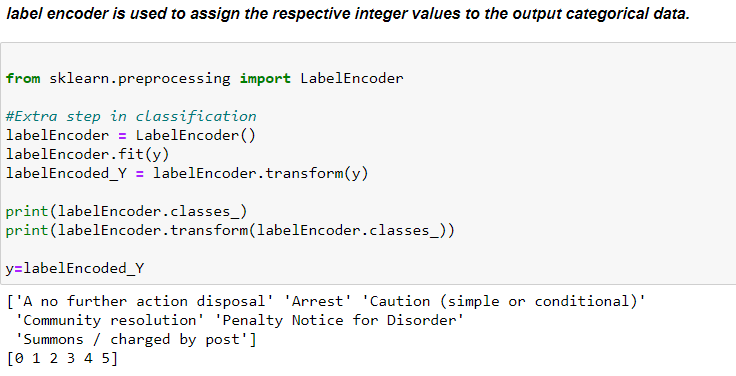
Data pre-processing deals with the preparation of the data so that the machine can take the input and make the accurate predictions.

### **Feature selection:**

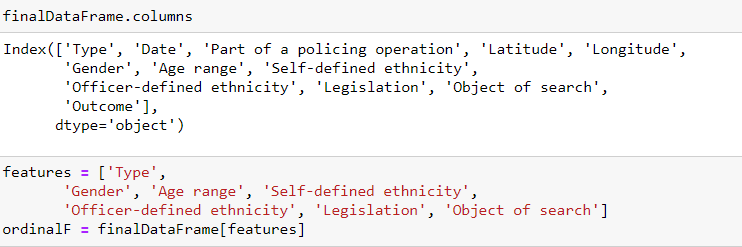
Feature selection involves in selection of the columns to take as input for a model and target variables as a output were model can predict it. Here target label is taken as ‘y’.



### **Label Encoder:**

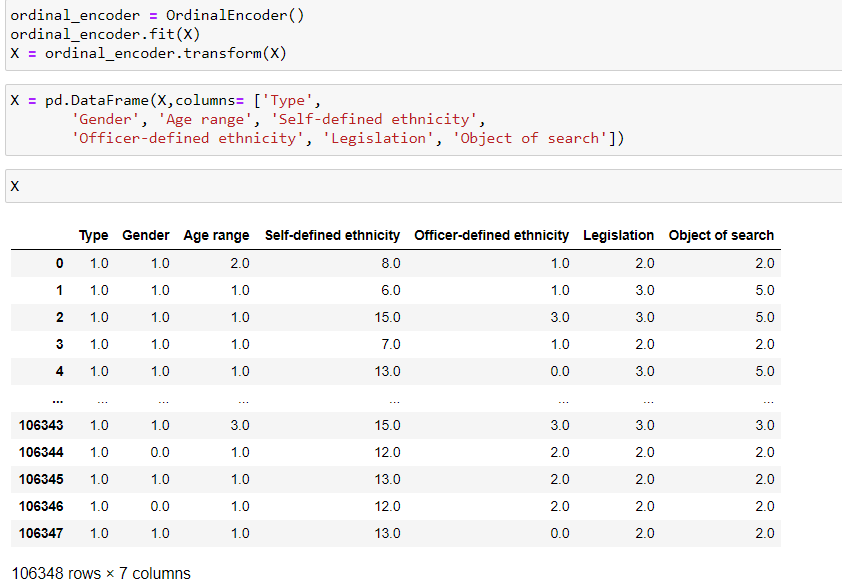
In this task I have used “**Label Encoder”** for target variable which has the categorical textual data. The label encoder helps us to assign the integer values to the textual data.

Next, I have selected the independent values as X where it is also a categorical data, so we need to convert it into the numeric data using some encoders which deals with categorical input.



### **Ordinal Encoder:**

I have used ordinal encoder for the independent values(X) so that it assigns the respective integer values to the categorical data.



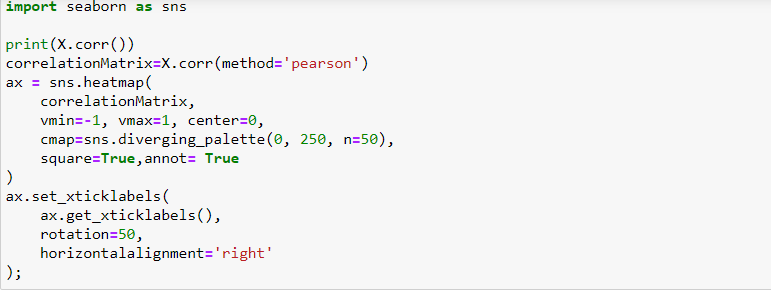
### **Correlation:**

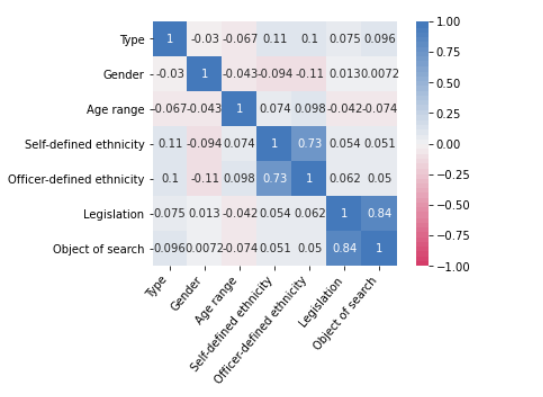
**Correlation** is used to find the correlation of a dataset where the two or more column’s in the dataset related to each other. It is represented by the “**Corr ()”** method on the data frame.

Correlation have different inbuilt methods they are:

* + - * **Pearson**: standard correlation coefficient
      * **Kendall**: Kendall Tau correlation coefficient
      * **spearman**: Spearman rank correlation
      * **callable**: callable with input two 1d ndarray and returning a float

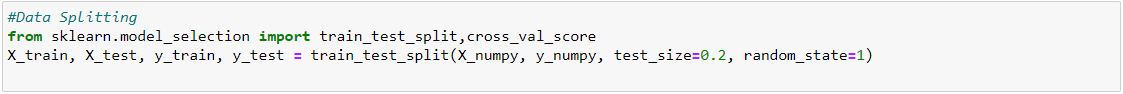
Using seaborn I have plotted the correlation of the input columns (X).





# **Developing prediction model & Reviewing of techniques used:**

In developing prediction model mainly involves in training the model and testing the accuracy of the model.

The “**sklearn”** provides the sub module called **“model\_selection”** from it we import “**train\_test\_split**” module where we can able to divide the data into two splits:

* Train data
* Test data
* In the above image we can see that the complete data is divided into two that is 80% of total data as a **train data** where model is trained.
* 20% of total data is used to **test** the model which it is trained with the 80% earlier.

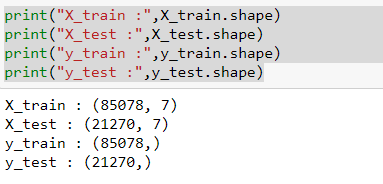


Fig: how the data is split into train and test data

## **Hyper-parameter Optimization (Hyperopt):**

HyperOpt was developed by James Bergstra as a Python open-source optimization library.

* Hyperopt is a Python library for serial and parallel optimization over awkward search spaces, which may include real-valued, discrete, and conditional dimensions.
* It is intended for large-scale model optimization with hundreds of parameters and can be used for multi-core and multi-machine optimization.
* Explicitly, the library has been used to refine machine learning pipelines including data preparedness and model collection.

There are many common algorithms of optimization, including:

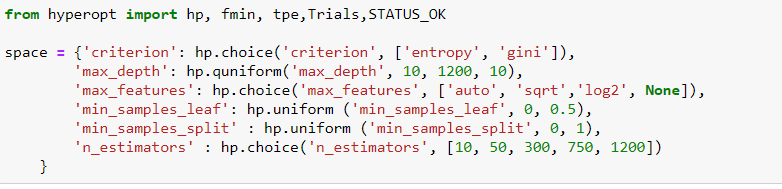
* Random Search
* Tree of Parzen Estimators
* Annealing
* Tree
* Gaussian Process Tree and model hyperparameters.

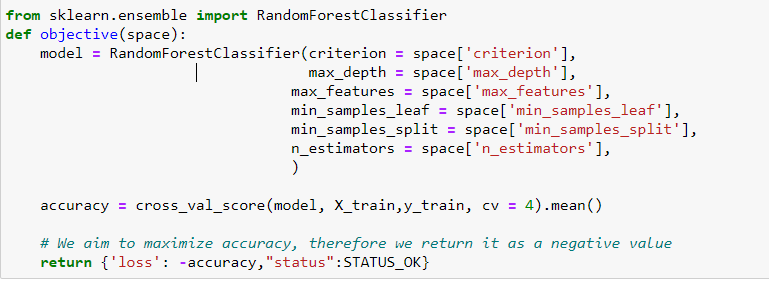
Above all those algorithms the “**Tree of Parzen Estimators (TPE)”** is a good default optimization algorithm used in the “Hyperopt”

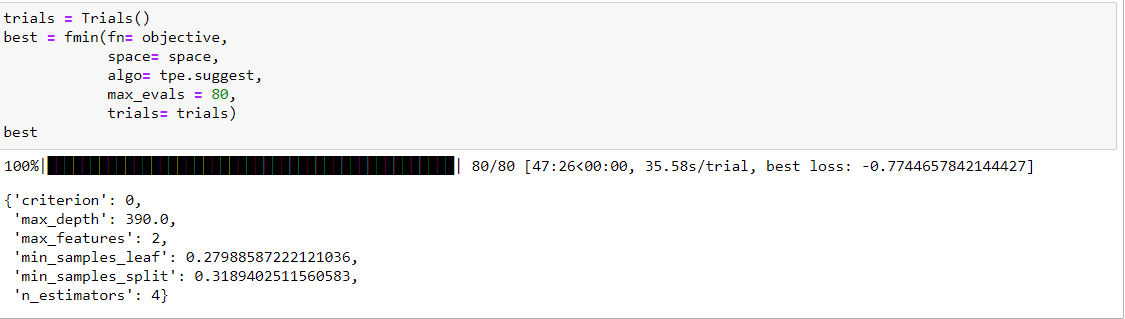
An ML algorithm usually has a loss/cost function that must be minimised by considering the algorithm's hyperparameters and parameters. Weights and biases are determined by the ML algorithm's Optimization mechanism. Choosing good hyperparameters reduces the loss function much more and makes the model more stable and reliable, improving the model's overall performance.

First, I used **hp. choice ()** to return one of the two dictionaries. now, since we need to take parameters based on the classifier. I allocated hyperopt spaces to various variables, after which I described the model and fitted x\_train and y\_train to the model.

The **fmin()** function will iterate over various sets of algorithms and their hyperparameters and return the set with the lowest loss. You'll notice that I've reduced the loss on x\_train and y\_train, you can use cross validation and other techniques to avoid overfitting.







In above we can see the suggestions from the hyperopt to get the accuracy of “0.7744657842144427”:

{

'criterion': 0,

'max\_depth': 390.0,

'max\_features': 2,

'min\_samples\_leaf': 0.27988587222121036,

'min\_samples\_split': 0.3189402511560583,

'n\_estimators': 4

}

We use these suggestions in our model to improve our accuracy.

## **GridSearchCV for model tuning:**

Grid-search is a method for determining the best hyperparameters of a model that result in the most 'accurate' predictions. **GridSearchCV** exhaustively considers all parameter combinations

In the specified distribution.

**Hyperparameter:** Model configuration argument specified by the developer to guide the learning process for a specific dataset.



The above algorithm suggested the following parameters which can be used in our model to improve the accuracy.

Best score = 0.7744657849810608,

Best Hyperparameters = {

'criterion': 'entropy',

'max\_depth': 4,

'max\_features': 7,

'max\_samples': 200,

'n\_estimators': 20

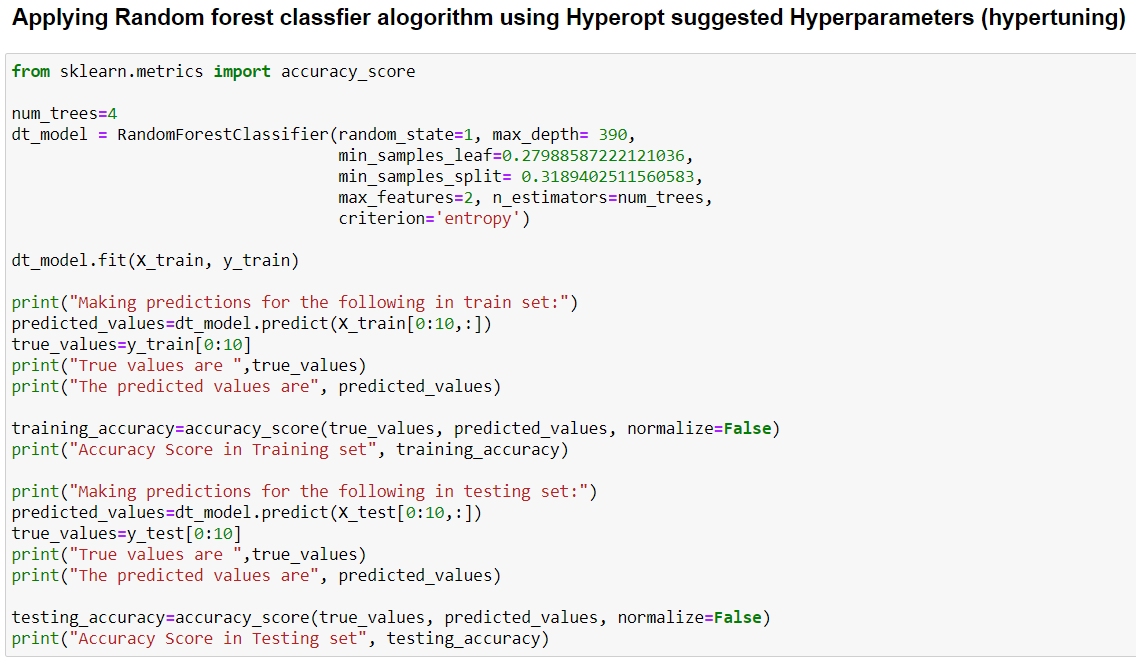
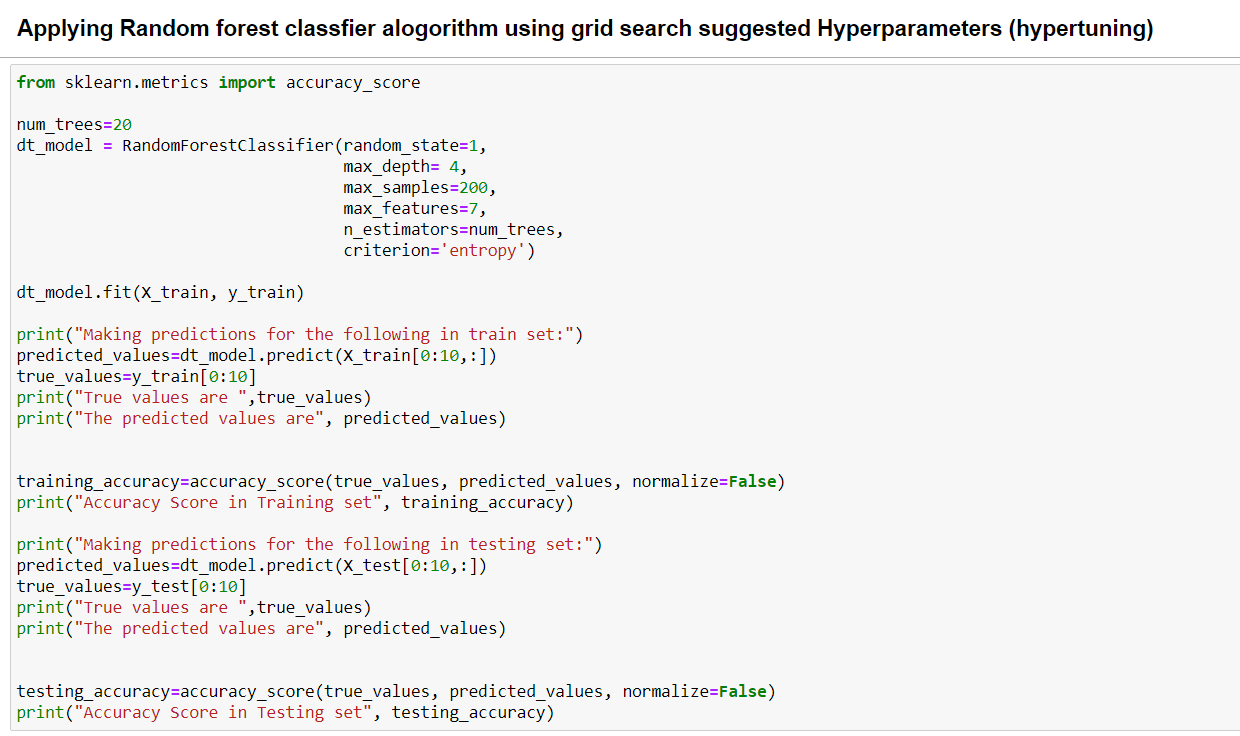
}

# **Building a model using RandomforestClassifer Algorithm:**

**RandomforestClassifer Algorithm:**

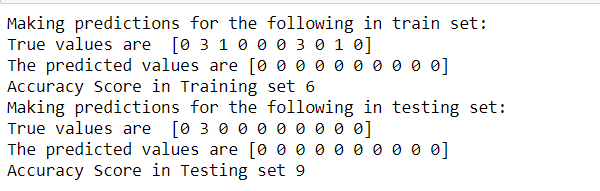
Random forest is a supervised learning method that may be used for classification and regression. However, it is mostly employed to solve categorization related data problems.

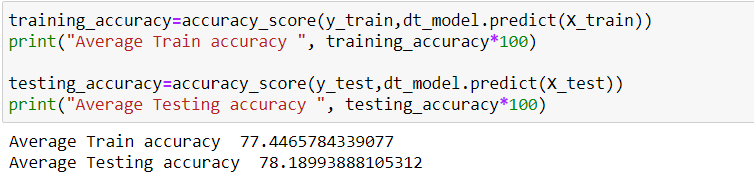
A Random forest generates decision trees from data samples, then receives predictions from each of them, and eventually votes to choose the best option. It is an ensemble approach that outperforms a single decision tree because it eliminates over-fitting by averaging the results.

In my model I have used the parameters suggested by the “**Hyperopot”** and “**GridSearchCV”** to improve the accuracy of my model.

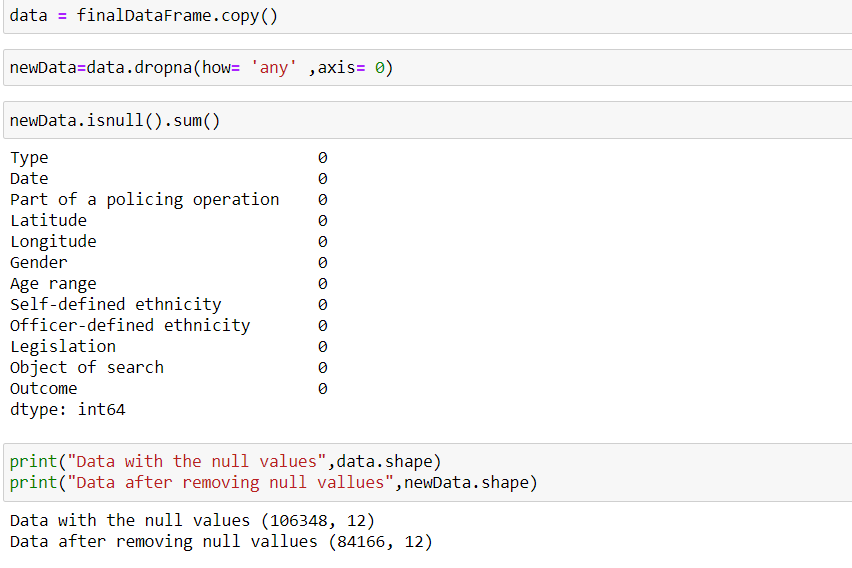
# **Interpretation of results:**

In above section I have used the “Hyperopt” and “GridsearchCV” parameters to improve the accuracy of my model below or the Screen Shots of the accuracy provided by both models as follows:

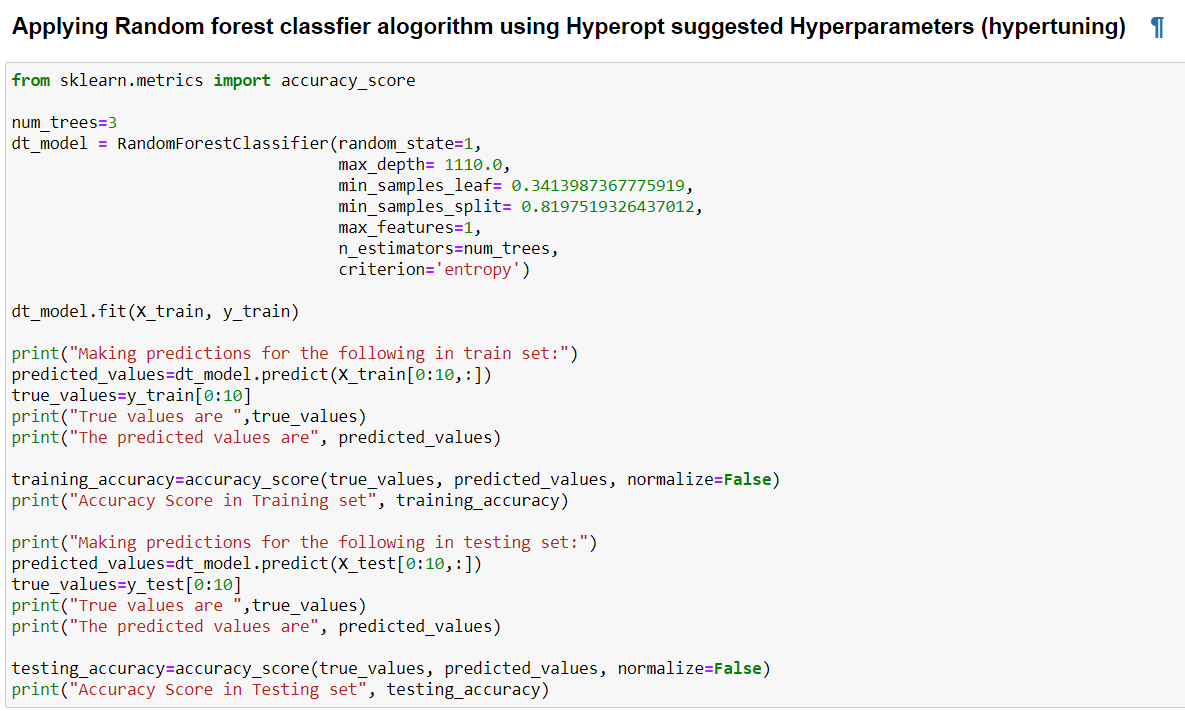


**Accuracy score of the model is as follows:**

## **Comparing the results with removing all the null values:**

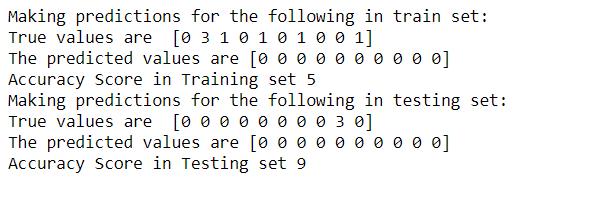
After completing above all the steps now I have performed all the above operations on the dataset removing all the null values with out performing the data cleaning operations.

Now I have performed the feature selection, label and ordinal encoders on the X and y, later I have performed hyperopt and GridsearchCV and got the model parameters next, I have added this model parameters on the model and got the following results:

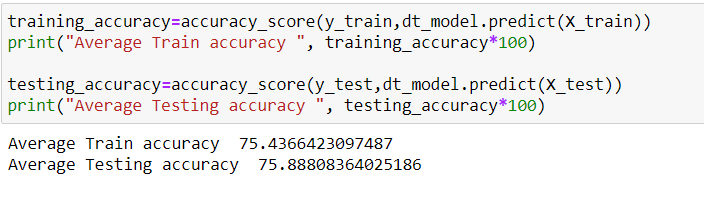




## **Prediction values after using both parameters:**

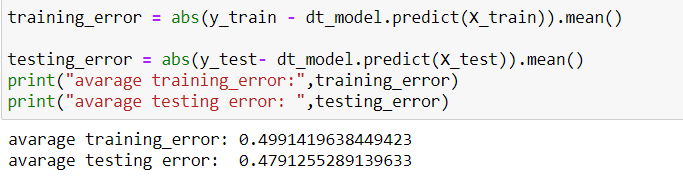


Accuracy of the model without removing of null values:

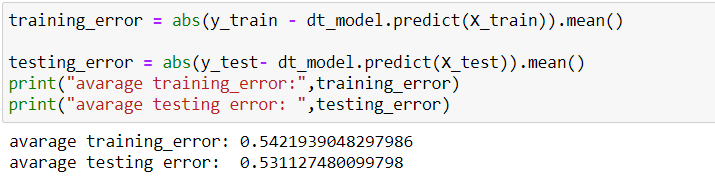


## **Comparison of average training and testing errors:**

**Average train & test error of model which has performed the data cleaning:**

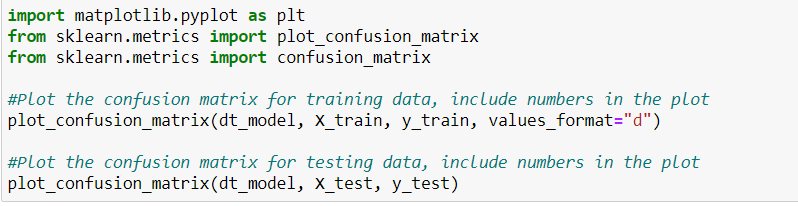


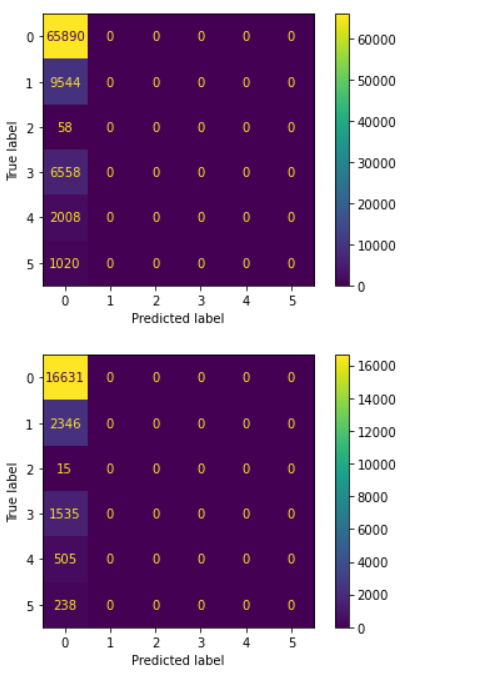
**Average train & test error of model which has removed the data Null values in dataset:**



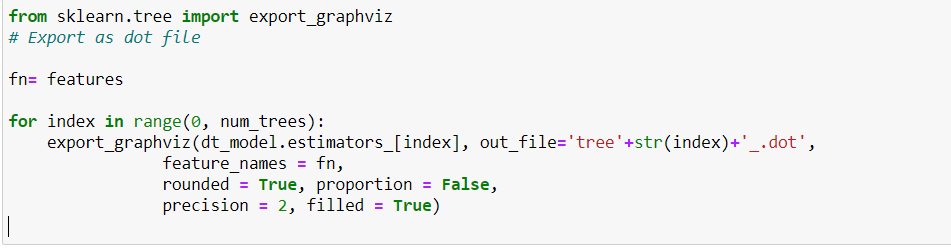
## **Confusion Matrix:**

A Confusion matrix is a N x N matrix that is used to assess the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values to the machine learning model's predictions. This provides us with a comprehensive picture of how well our classification model is performing and the types of errors it is making.





# **Saving the decision trees generated by the random forest model:**

We can save the decision trees generated by the random forest model which is used to make predictions and can view them in the tree structure

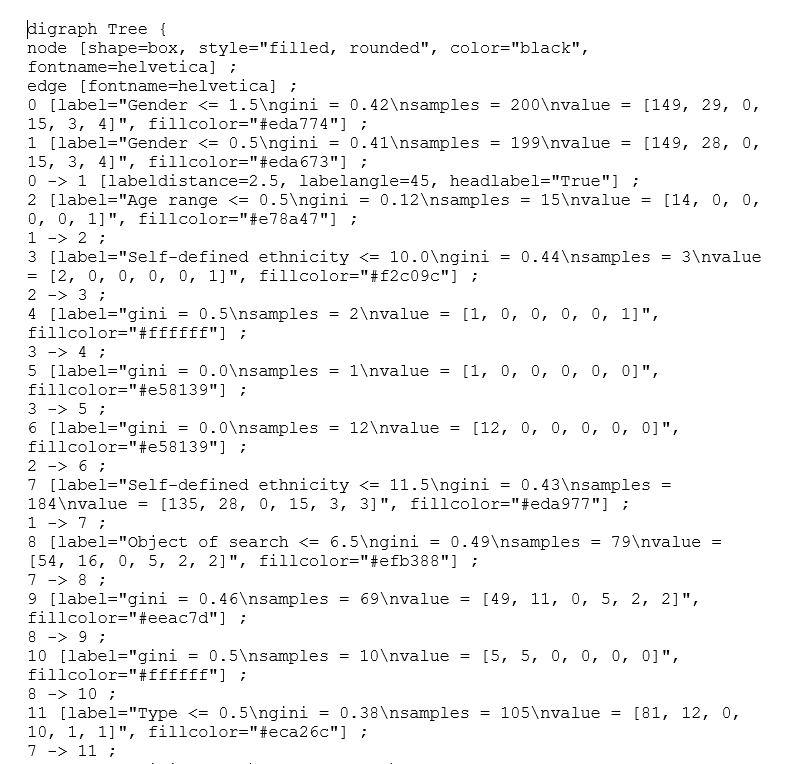
We can we decision trees here: <https://dreampuf.github.io/GraphvizOnline>

Fig: decision tree in text format

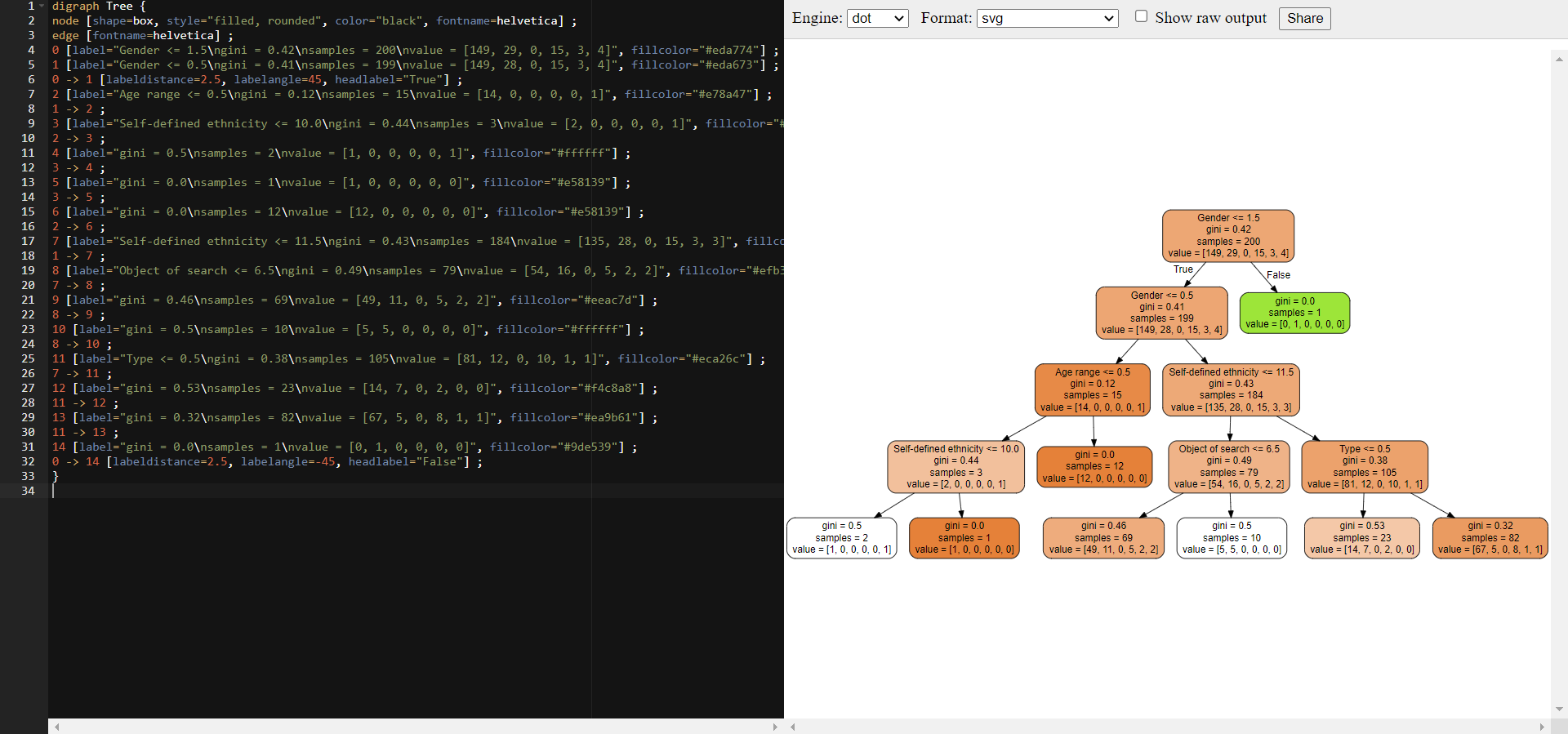


Fig: textual data represented in diagrammatically

# **My Reflection:**

Based on the dataset assigned to me I have performed the following tasks:

* Understanding the Data Set and what type of data present in it.
* Performed **Data cleaning,** worked with the Null values using built in methods.
* Evaluated the data set and represented it visually.
* Performed the **feature selection**
* Performed data **pre-processing** that deals with the categorical data using **encoders**.
* Performed Hyper parameters tuning using “**Hyperopt**”, “**GridSearchCV**” to improve the model accuracy.
* Applied **Random forest Classification** to build a model to make prediction.
* Calculated the test and training accuracy.
* Calculated the average test and train error.
* Saved the model generated decision trees into a file.

Finally, I conclude that the Random forest classification is the good model to make predictions towards the categorical data and I also conclude that removing the missing values instead of managing them effects the model accuracy as **Data Cleaning** plays the vital role in the model building.

In doing this task I have taken references from the previous classes which helped me mostly in understanding all the concepts.

# **References:**

Dr Ambikesh Jayal, Programme Director of the MSc Data Science.

Source:

I)Tutorial for Simple Classification Model with categorial input

as independent variables and Encoding.

II) Hyperparameter Tuning using GridSearch.

****Ali Shahaab-PhD., working with Datasets, pandas, NumPy, visualization.

Source:<https://learn.cardiffmet.ac.uk/course/view.php?id=30028&section=4>

**** Sujandutta,2019., Python | Visualize Missing Values (Nan) Values Using “**Missingno Library”,**

source: <https://www.geeksforgeeks.org/python-visualize-missing-values-nan-values-using-missingno-library/>.

Dr Jasonbrownlee,2017., Handling missing data using python

source: <https://machinelearningmastery.com/handle-missing-data-python/>.

Marco Santos,2020., How to Use Plotly for Data Visualization

Source:<https://towardsdatascience.com/how-to-use-plotly-for-data-visualization-f3d62bbcfd92>

Karandeep Singh,2020., Hypertune Machine Learning Model using HYPER OPT Library

Source: [https://github.com/karndeepsingh/HyperOpt](https://github.com/karndeepsingh/HyperOptb)

Bergstra, J., Yamins, D., Cox, D. D. (2013) Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. To appear in Proc. of the 30th International Conference on Machine Learning (ICML 2013)

# **Appendixes:**

1. Complete assignment data is available in my GitHub:

<https://github.com/saikumarsandra/AssignmentDataset>

1. Video explanation link:

<https://youtu.be/jpT39z-t6LA>