**A CAPSTONE PROJECT**

**ON**

**STUDENT ADMISSION ANALYSIS USING ML**

**ALGORITHMS AND POWERBI**

Submitted By,

DHANAVEL K

# ABSTRACT

Education always been as priority for the students, to get placed in their dream companies the students should complete their higher education in a top colleges and universities. To get admission in top colleges the students marks are evaluated, and the filtered lists are taken for further admission process. Here the filtration process is made through cross checking the marks scored by the students in their schoolings and according to that they are been applicated for the college. As an optional step some set of students can grab their seats in top universities using their sports certifications. This proposed system makes a prediction which results in identifying the average possibilities for the sports certified students to get their seats in top colleges. This proposed systems also proves the importance of effective sports participation for the students during their schoolings.

Dataset Link: https://github.com/dhanavel2021/My-Capstone-Project/blob/main/final%20Csv.csv

Source Code: https://github.com/dhanavel2021/My-Capstone-Project/blob/main/Capstone%20Project%20Code%20Sheet.ipynb

# ACKNOWLEDGEMENTS

I am using this opportunity to express my gratitude to everyone who supported me throughout the course of my capstone project. I am thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on several issues related to the project.

Further, I have fortunate to have Mr. PRASAD as my mentor. He has readily shared his immense knowledge in data analytics and guide me in a manner that the outcome resulted in enhancing my data skills.

I wish to thank all the faculties, as this project utilized knowledge gained from every course that formed the DSP program.

I certify that the work done by me for conceptualizing and completing this project is original and authentic.

Date: July 10, 2022 Name: DHANAVEL K

# CERTIFICATE OF COMPLETION

I hereby certify that the project titled “STUDENT ADMISSION ANALYSIS USING ML ALGORITHMS AND POWERBI” was undertaken and completed under supervision of Mr. Prasad from the batch of DSP (Apr 2018)

Mentor: Mr. Prasad Date: April 1, 2018 Place–Karur

**TABLE OF CONTENTS**

[ABSTRACT 2](#_Toc109933485)

[ACKNOWLEDGEMENTS 3](#_Toc109933486)

[CERTIFICATE OF COMPLETION 4](#_Toc109933487)

[CHAPTER 1 11](#_Toc109933488)

[INTRODUCTION 11](#_Toc109933489)

[CHAPTER 2 12](#_Toc109933490)

[DATA COLLECTON 12](#_Toc109933491)

[CHAPTER 3 13](#_Toc109933492)

[INTEGRATING SQL SERVER WITH JUPYTER NOTEBOOK USING PYTHON LIBRARIES 13](#_Toc109933493)

[CHAPTER 4 14](#_Toc109933494)

[DATA PRE-PROCESSING 14](#_Toc109933495)

[4.1 CHECKING FOR NULL VALUES 14](#_Toc109933496)

[4.2 VIEWING DATATYPES OF EACH COLUMN 15](#_Toc109933497)

[4.3 VIEWING COUNT OF DISTINCT VALUES IN EACH COLUMN 15](#_Toc109933498)

[4.4 CONVERTING NON-NUMERICAL DATA INTO NUMERIC DATA 16](#_Toc109933499)

[CHAPTER 5 17](#_Toc109933500)

[EXPLORATORY DATA ANALYSIS 17](#_Toc109933501)

[5.1 DATA SUMMARIZATION 17](#_Toc109933502)

[5.2 CORRELATION BETWEEN VARIABLES 18](#_Toc109933503)

[5.3 VISUALIZING THE RELATION BETWEEN THE COLUMNS 18](#_Toc109933504)

[CHAPTER 6 21](#_Toc109933505)

[SPLITTING ATTRIBUTES 21](#_Toc109933506)

[CHAPTER 7 22](#_Toc109933507)

[TRAINING AND TESTING THE SPLITTED DATASET 22](#_Toc109933508)

[CHAPTER 8 23](#_Toc109933509)

[MODEL SELECTION 23](#_Toc109933510)

[8.1 SUPERVISED LEARNING 24](#_Toc109933511)

[i) Classification 24](#_Toc109933512)

[ii) Regression 25](#_Toc109933513)

[8.2 UNSUPERVISED LEARNING 25](#_Toc109933514)

[i) Clustering 25](#_Toc109933515)

[ii) Association 26](#_Toc109933516)

[8.3 REINFORCEMENT LEARNING 26](#_Toc109933517)

[8.4 SEMI-SUPERVISED LEARNING 26](#_Toc109933518)

[CHAPTER 9 28](#_Toc109933519)

[FITTING THE TRAINING DATASET INTO THE MODELS 28](#_Toc109933520)

[9.1 DECISION TREE CLASSIFIER 28](#_Toc109933521)

[9.2 RANDOM FOREST CLASIFIER 29](#_Toc109933522)

[9.3 GRADIENT BOOSTING CLASSIFIER 29](#_Toc109933523)

[9.4 SUPPORT VECTOR MACHINE 30](#_Toc109933524)

[9.5 KNN 30](#_Toc109933525)

[9.6 NAIVE-BAYES 31](#_Toc109933526)

[9.7 LOGISTIC REGRESSION 31](#_Toc109933527)

[CHAPTER 10 32](#_Toc109933528)

[EVALUATION AND PARAMETER TUNING (if required) 32](#_Toc109933529)

[10.1 DECISION TREE 33](#_Toc109933530)

[10.1.1 HYPER PARAMETER TUNING FOR DECISION TREE MODEL 34](#_Toc109933531)

[10.2 RANDOM FOREST 35](#_Toc109933532)

[10.2.1 HYPER PARAMETER TUNING FOR RANDOM FOREST MODEL 36](#_Toc109933533)

[10.3 GRADIENT BOOSTING 37](#_Toc109933534)

[10.4 SUPPORT VECTOR MACHINE 38](#_Toc109933535)

[38](#_Toc109933536)

[10.5 KNEIGHBORS 39](#_Toc109933537)

[10.6 NAIVE-BAYES 40](#_Toc109933538)

[10.7 LOGISTIC REGRESION 41](#_Toc109933539)

[10.8 ACCURACY SCORES OF EACH MODELS 42](#_Toc109933540)

[CHAPTER 11 43](#_Toc109933541)

[INTEGRATION OF POWERBI IN JUPYTER USING PYTHON LIBRARIES 43](#_Toc109933542)

[POWERBI DASHBOARD 44](#_Toc109933543)

[CONCLUSION 45](#_Toc109933544)

[REFERENCE 46](#_Toc109933545)

**LIST OF FIGURES**

[Figure 1 Checking Null Values 11](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908319)

[Figure 2 Source code for checking columns 12](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908320)

[Figure 3 Source code for viewing count of distinct values 12](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908321)

[Figure 4 Source code for converting non-numerical data into numerical data 14](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908322)

[Figure 5 Source code for statistical analysis of dataset 14](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908323)

[Figure 6 Source code for calculating relationship between columns. 15](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908324)

[Figure 7 Scatter plot 16](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908325)

[Figure 8 Histogram plot for every column 16](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908326)

[Figure 9 Pair Plot the dataset 17](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908327)

[Figure 10 Source code for splitting attributes for ML process. 19](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908328)

[Figure 11 Source code for train, test split 19](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908329)

[Figure 12 Classification and Regression in Supervised Learning 21](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908330)

[Figure 13 Unsupervised learning Clustering 23](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908331)

[Figure 14 Source code for decision tree classifier 26](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908332)

[Figure 15 Source code for Random Forest Classifier 26](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908333)

[Figure 16 Source code for gradient boosting classifier 27](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908334)

[Figure 17 Source code for SVC 27](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908335)

[Figure 18 Source Code for KNN 28](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908336)

[Figure 19 Source code for Naïve-Bayes 28](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908337)

[Figure 20 Source Code for Logistic Regression 29](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908338)

[Figure 21 Source code for evaluation of decision tree classifier 30](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908339)

[Figure 22 Source code for Hyper parameter tuning – Decision tree Model 31](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908340)

[Figure 23 Source code for Random Forest Classifier evaluation 32](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908341)

[Figure 24 Source code for Hyper parameter tuning – Random Forest Model 33](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908342)

[Figure 25 Source code for Gradient Boosting evaluation 34](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908343)

[Figure 26 Source code for KNN evaluation 36](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908344)

[Figure 27 Source code for NAIVE-BAYES evaluation 37](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908345)

[Figure 28 Source code for LOGISTIC REGRESSION evaluation 38](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908346)

[Figure 29 Accuracy Scores of each model 39](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908347)

[Figure 30 Installing PowerBI library in Jupyter Notebook. 40](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908348)

[Figure 31 Source code for importing powerBI client 41](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908349)

[Figure 32 PowerBI Dashboard 41](file:///C:\Users\DHANAVEL\OneDrive\Desktop\Capstone\CAPSTONE%20PROJECT%20REPORT.docx#_Toc109908350)

# CHAPTER 1

# INTRODUCTION

As we all know that education plays an important role in setting a good career for each person. It can be acquired by good education. For good education there are n number of institutions offering different courses. Students, after completing their higher secondary schoolings, they show interest in joining institutions based on their individual interest. Educational percentage scored in their final examinations plays a vital role in getting admission in high quality educational institutions. Not only marks play a vital role, but also every student’s active sports participation with higher level certification helps them to acquire admission in a good institution. In this project, we are going to analyse and predict that the sportsmen/women have high chance of getting admission with above average grades or not. So, we all came across a situation when we are not allowed to participate in sports competition during our HSC period.

This project is also used to showcase the importance of sports, which play a role in admission process using machine learning algorithms.

I have done integration of MySQL server and PowerBI dashboard in Jupyter Notebook Using Python Libraries.

# CHAPTER 2

# DATA COLLECTON

The dataset consists of variables:

* AGE – Integer datatype.
* SPORT\_PARTICIPANT – Integer datatype.
* EDUCATIONAL\_PERCENTAGE – Integer datatype.
* FITNESS – Object datatype.
* HANDICAP - Object datatype.
* SEX - Object datatype.
* POSSIBILITY\_OF\_ADMISSION – Integer datatype.

# CHAPTER 3

# INTEGRATING SQL SERVER WITH JUPYTER NOTEBOOK USING PYTHON LIBRARIES

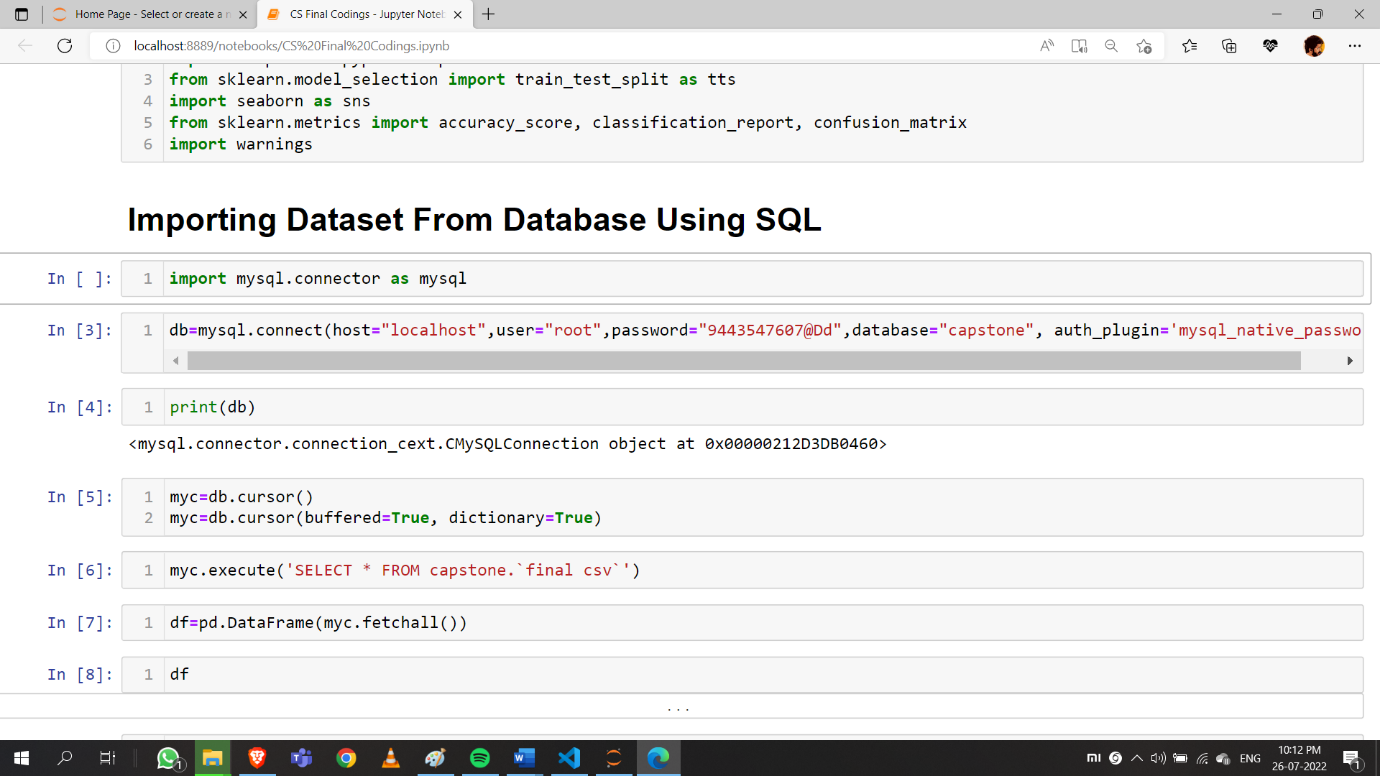
* SQL(Structured Query Language) is most used software for data analytics in which we can create dataset using MySQL commands and analyse it.
* Created a database for my project and imported my dataset into the dataset.
* Used python libraries to integrate MySQL server and Jupyter Notebook.
* Library used for integration is “**mysql.connector**”.

Fig 3.1 Integrating MySQL server with Jupyter Notebook

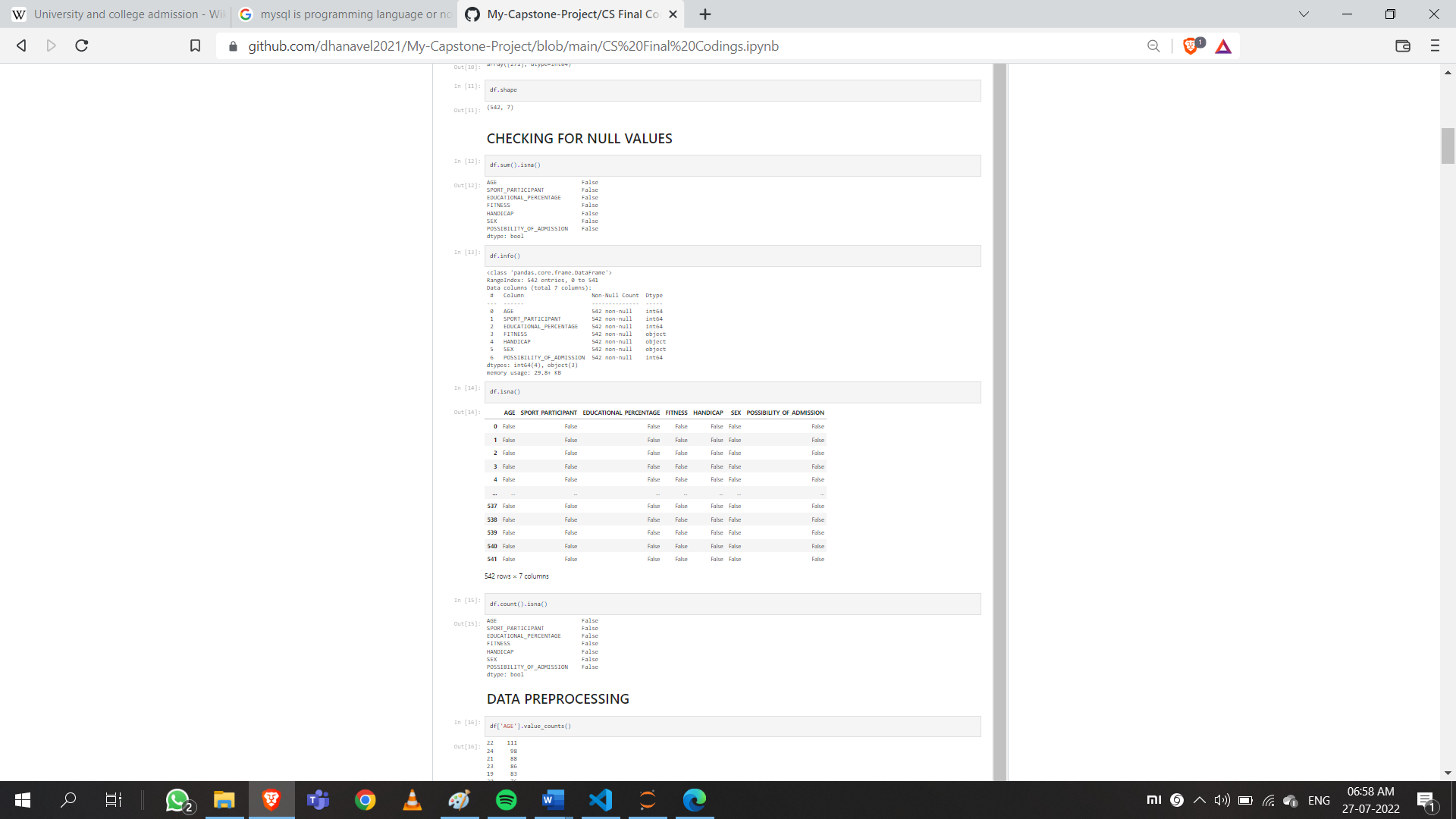
# CHAPTER 4

# DATA PRE-PROCESSING

Data pre-processing is a process of checking for null values in the dataset, checking outliers, duplicates, missing values and filling the null values, missing values or drop the row which has null or missing values or duplicates.

At first, I have checked for any null values

# 4.1 CHECKING FOR NULL VALUES

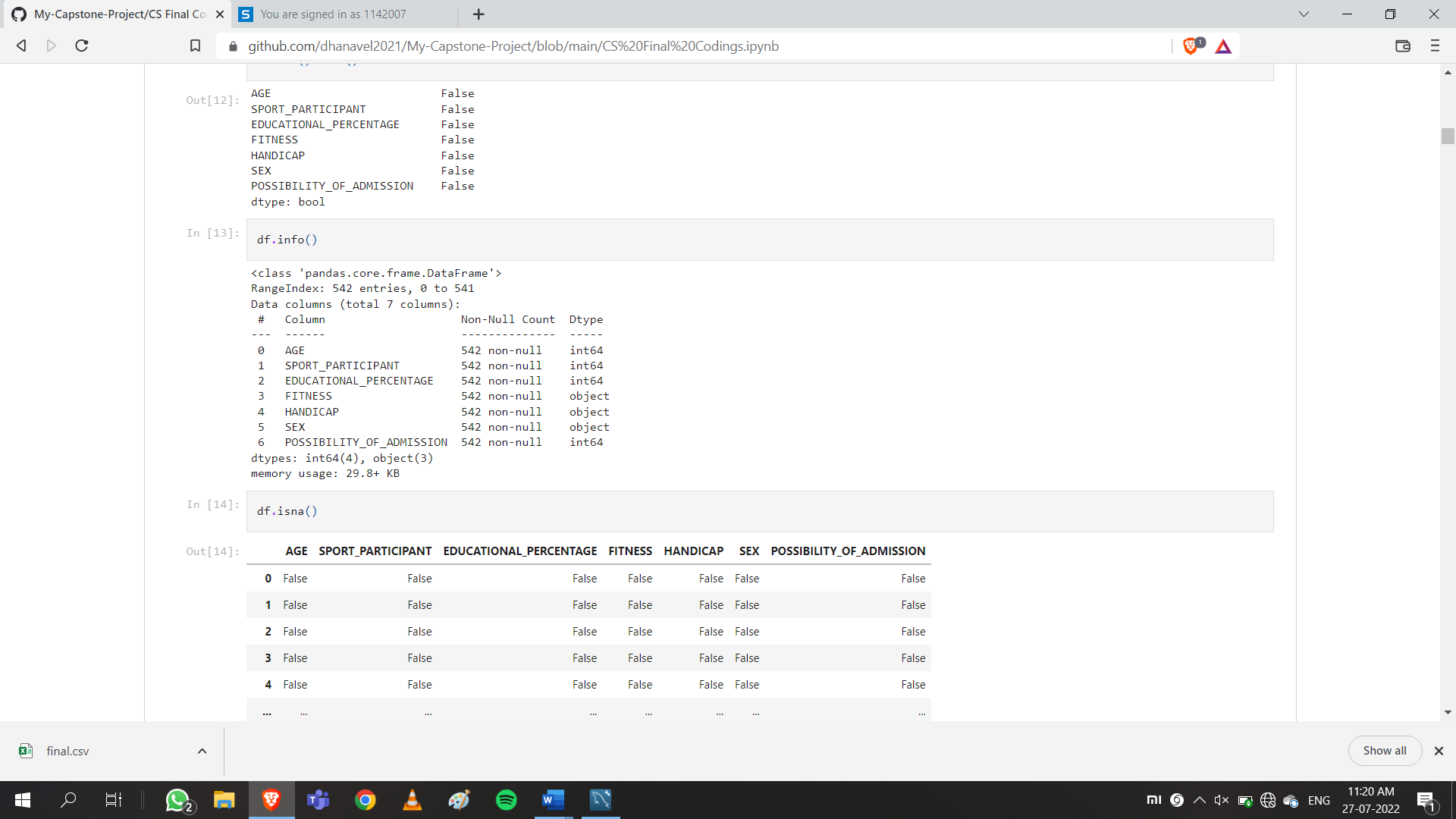


**Figure 1 Checking Null Values**

Fig 4.1.1 Checking Null Values(Source Code)

There are no null values in the dataset.

# 4.2 VIEWING DATATYPES OF EACH COLUMN



**Figure 2 Source code for checking columns**

# 4.3 VIEWING COUNT OF DISTINCT VALUES IN EACH COLUMN

A screenshot of a computer

Description automatically generated

**Figure 3 Source code for viewing count of distinct values**

# 4.4 CONVERTING NON-NUMERICAL DATA INTO NUMERIC DATA

While doing machine learning algorithms, non-numeric data cannot be processed and get predicted. So, we are changing the non-numeric data into categorical data or numeric data.

Graphical user interface, text, application

Description automatically generated

**Figure 4 Source code for converting non-numerical data into numerical data**

# CHAPTER 5

# EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis(EDA) in data science is used to investigate the characteristics of datasets and relationship between variables, summarize the dataset.

Often EDA is done by visualizing the datasets.

Libraries used for visualization are: “**matplotlib.plyplot**” and “seaborn”

# 5.1 DATA SUMMARIZATION

Using describe() keyword we can see a detailed statistical analysis of the dataset.

**Figure 5 Source code for statistical analysis of dataset**

# 5.2 CORRELATION BETWEEN VARIABLES

Using the keyword “**dataset.corr()”,** relationship between the columns can be calculated.

**Figure 6 Source code for calculating relationship between columns.**

# 5.3 VISUALIZING THE RELATION BETWEEN THE COLUMNS

**Graphical user interface

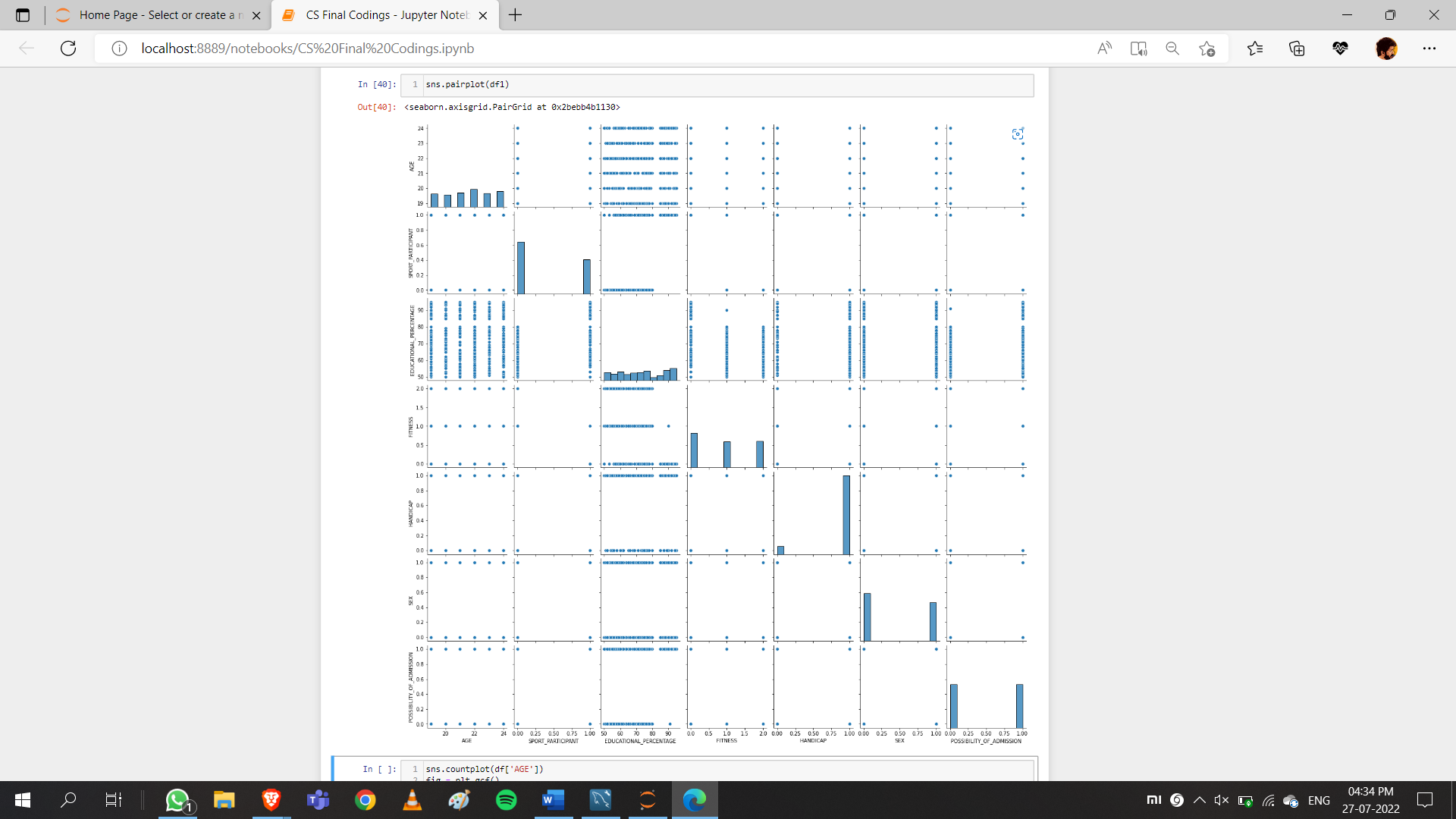
Description automatically generated**

**Figure 7 Scatter plot**

Graphical user interface, chart, box and whisker chart

Description automatically generated

**Figure 8 Histogram plot for every column**



**Figure 9 Pair Plot the dataset**

# CHAPTER 6

# SPLITTING ATTRIBUTES

Before processing machine learning algorithms, the dataset should be split into two variables such as, one variable (x) contains data except dependent variable and other(y) with only the independent data.

For that case, first we need to notice that the purpose of this project. In my case, predicting the possibility of admission of students is the target, so my target variable should contain only the variable which gives the required data.

I have used x and y as my two different variables for splitting.

y– Dependent Variable

x – Independent Variable



**Figure 10 Source code for splitting attributes for ML process.**

# CHAPTER 7

# TRAINING AND TESTING THE SPLITTED DATASET

After splitting the data set into dependent and independent variables, next step is to split both variables into train and test data.

Python Library used for splitting train and test data is : “SKLEARN”

Package used : “**train\_test\_split**”

Graphical user interface, application

Description automatically generated

**Figure 11 Source code for train, test split**

Here I have splitted the dependent variable x into xtrain and xtest and independent variable y as ytrain and ytest.

The size(percentage) of data that is to be tested after trained is mentioned inside parenthesis as test\_size = .2 which indicates 80% of both x and y datas are trained and tested with remaining 20% of data.

# CHAPTER 8

# MODEL SELECTION

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.

There are three types in Machine Learning,

* Supervised Learning
* Classification Algorithms
* Regression Algorithms
* Unsupervised Learning
* Clustering Algorithm
* Association
* Reinforcement Learning
* Semi-Supervised Learning

# 8.1 SUPERVISED LEARNING

Chart, scatter chart

Description automatically generatedA supervised learning algorithm takes a known set of input data and known responses to the data and forms a model to generate reasonable predictions for the response to the new input data. Use supervised learning if we have existing data for the output you are trying to predict.

**Figure 12 Classification and Regression in Supervised Learning**

# i) Classification

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations based on training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into a few classes or groups.

# ii) Regression

Regression is a supervised machine learning technique which is used to predict continuous values. The goal of the regression algorithm is to plot a best-fit line or a curve between the data. The three main metrics that are used for evaluating the trained regression model are variance, bias and error.

# 8.2 UNSUPERVISED LEARNING

Unsupervised learning uses machine learning algorithms to analyse and cluster unlabelled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention.

# i) Clustering

Clustering is an unsupervised method that works on datasets in which there is no outcome (target) variable nor is anything known about the relationship between the observations, that is, unlabelled data.

Diagram

Description automatically generated

**Figure 13 Unsupervised learning Clustering**

# ii) Association

Association rule is unsupervised learning where algorithm tries to learn without a teacher as data are not labelled. Association rule is descriptive not the predictive method, generally used to discover interesting relationship hidden in large datasets.

# 8.3 REINFORCEMENT LEARNING

Reinforcement learning is a machine learning training method based on rewarding desired behaviours and/or punishing undesired ones. In general, a reinforcement learning agent can perceive and interpret its environment, take actions and learn through trial and error.

# 8.4 SEMI-SUPERVISED LEARNING

Semi-supervised machine learning is a combination of supervised and unsupervised machine learning methods. With more common supervised machine learning methods, you train a machine learning algorithm on a “labelled” dataset in which each record includes the outcome information. But there can be unlabelled data, so first we need to label the unlabelled data and continue with supervised learning algorithms.

After analysing using EDA, the dataset collected will be suitable for performing supervised classification algorithms for predicting the results.

**Models used in this project are :**

* Decision Tree Cassifier
* Random Forest Classifier
* Gradient Boosting Classifier
* KNN
* Support Vector Machine
* Naïve-Bayes
* Logistic Regression

# CHAPTER 9

# FITTING THE TRAINING DATASET INTO THE MODELS

Sklearn python library is most used for importing machine learning algorithms

# 9.1 DECISION TREE CLASSIFIER

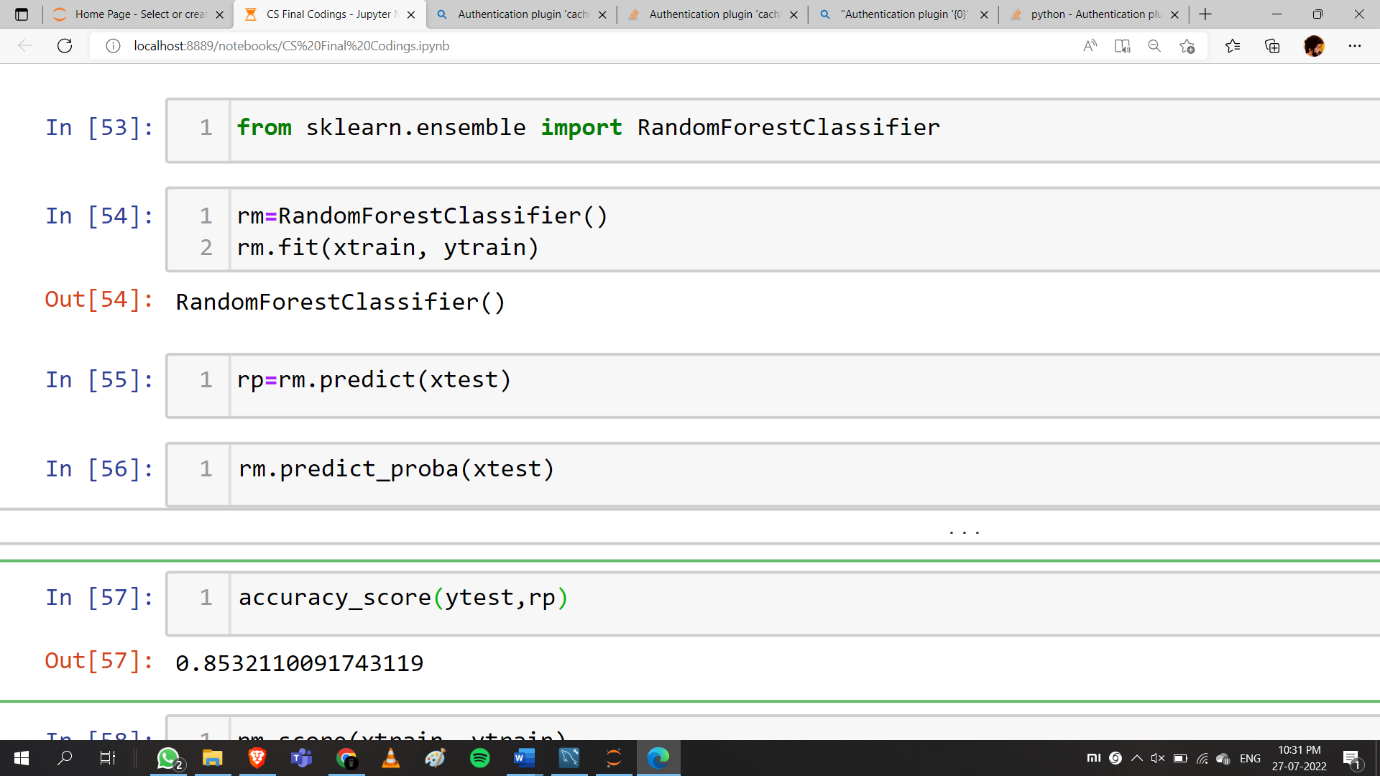
A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes.

Using the package “**DecisionTreeClassifier**” in “**tree**” package, the training dataset is fit into the model

**Figure 14 Source code for decision tree classifier**

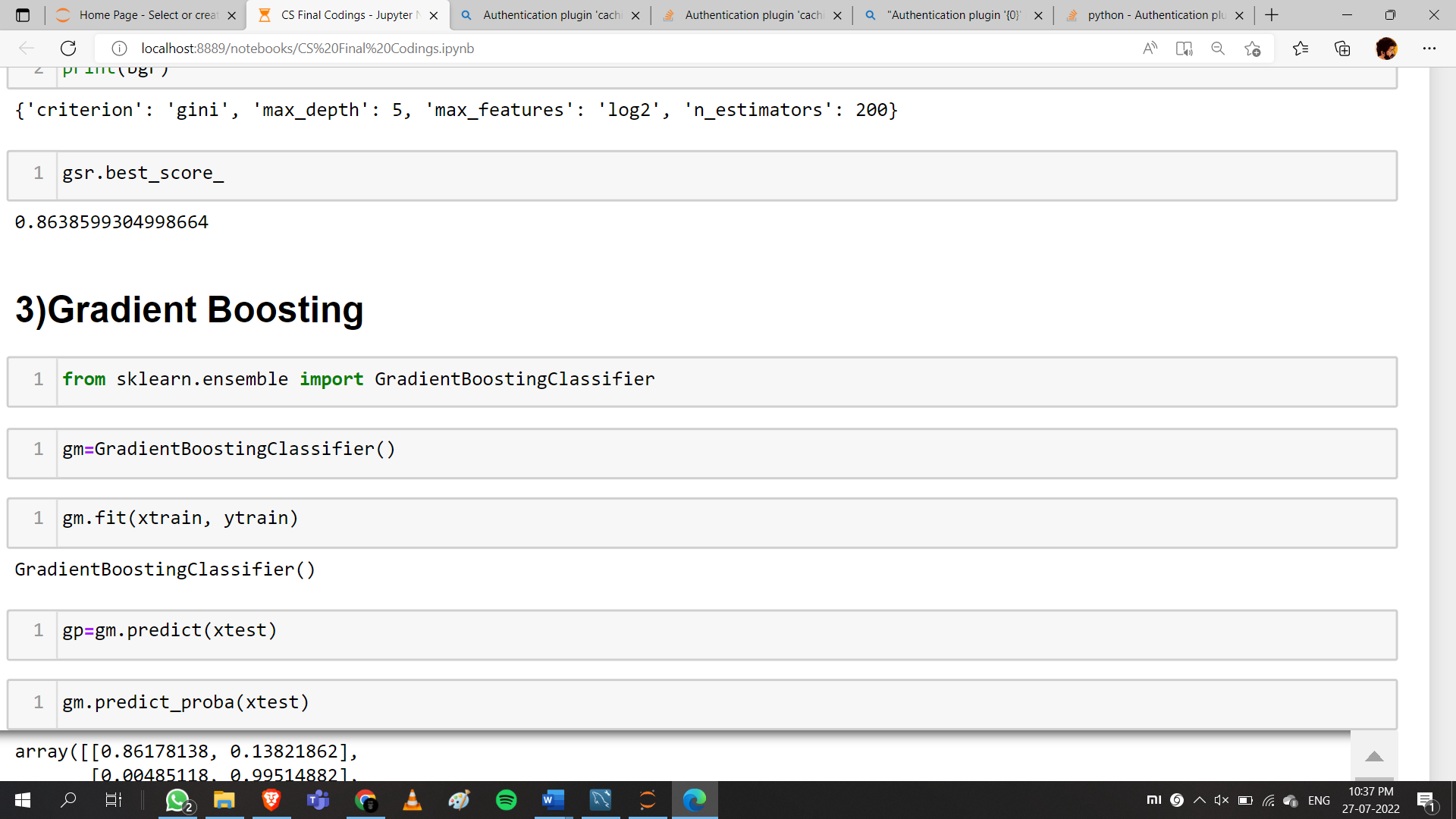
# 9.2 RANDOM FOREST CLASIFIER

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

Using “**RandomForestClassifier**” in “**ensemble**” package, the training dataset is fit into the model

**Figure 15 Source code for Random Forest Classifier**

# 9.3 GRADIENT BOOSTING CLASSIFIER

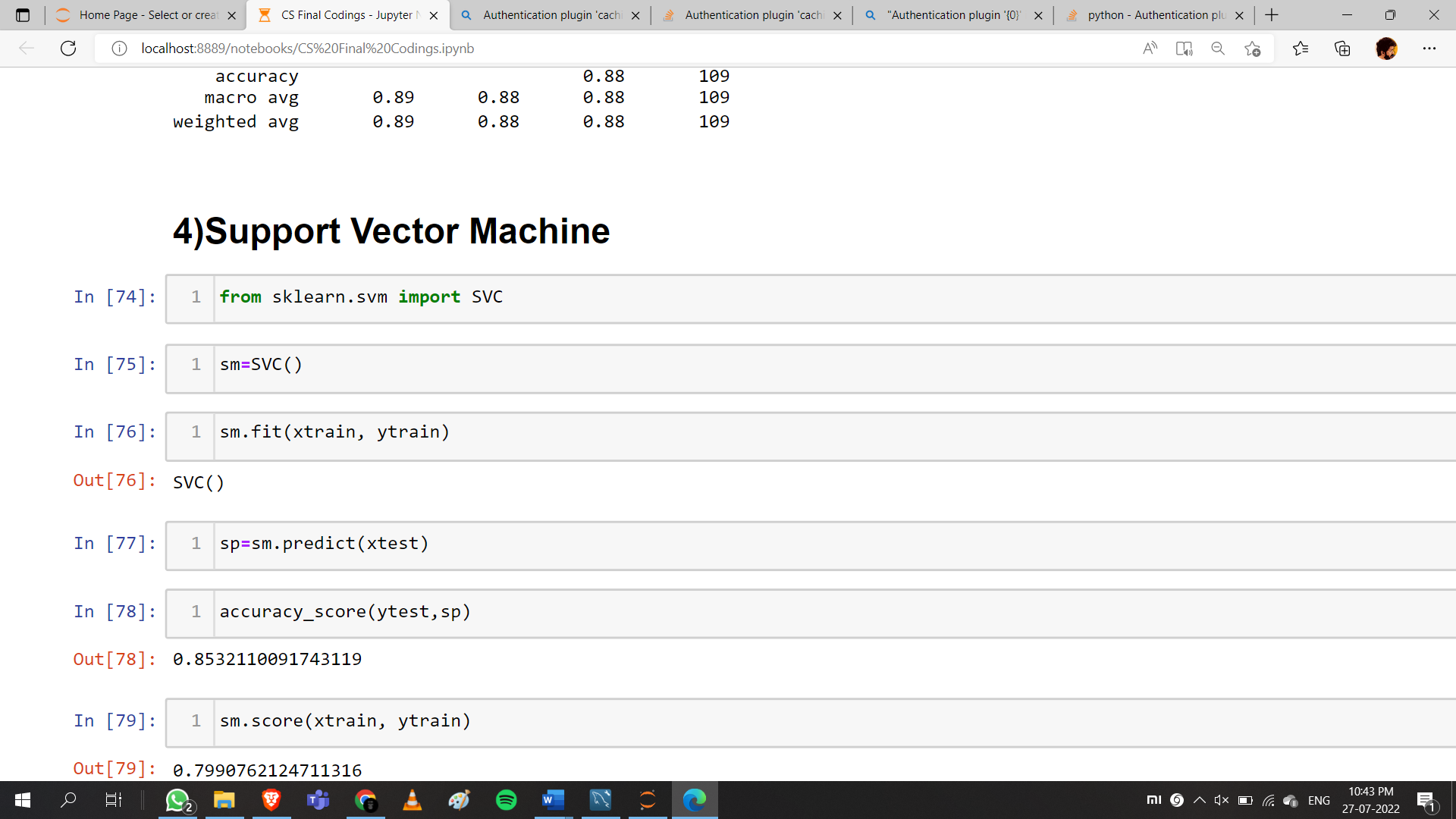
Using “**GradientBoostingClassifier**” in “**ensemble**” package, the training dataset is fit into the model

**Figure 16 Source code for gradient boosting classifier**

# 9.4 SUPPORT VECTOR MACHINE

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points.

Using the “SVC” in “**svm**” package, the training dataset is fit into the model.



**Figure 17 Source code for SVC**

# 9.5 KNN

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

Using the “**KNeighborsClassifier**” in “**neighbors**” package, the training dataset is fit into the model.

**Figure 18 Source Code for KNN**

# 9.6 NAIVE-BAYES

Graphical user interface, text, application, email

Description automatically generatedUsing the “**GaussianNB**” in “**naïve\_bayes**” package, the training dataset is fit into the model.

**Figure 19 Source code for Naïve-Bayes**

# 9.7 LOGISTIC REGRESSION

Using the “**LogisticRegression**” in “**linear\_model**” package, the training dataset is fit into the model.



**Figure 20 Source Code for Logistic Regression**

# CHAPTER 10

# EVALUATION AND PARAMETER TUNING (if required)

Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses. Model evaluation is important to assess the efficiency of a model during initial research phases, and it also plays a role in model monitoring.

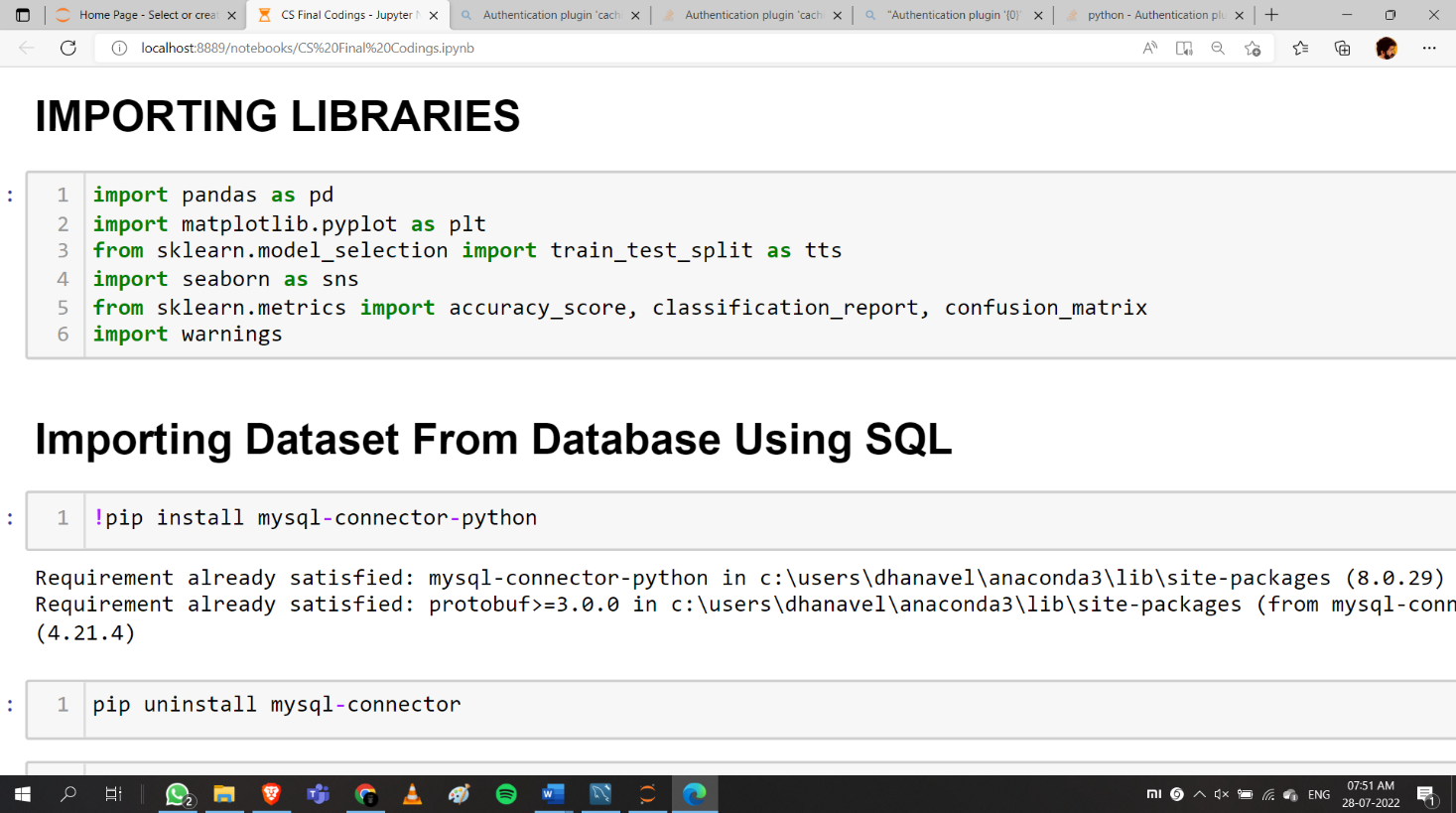
The Evaluation of classification models can be done by using “**accuracy\_score**”, “**confusion\_matrix**”, “**classification\_report**” using “**metrics**” package form “**sklearn**” library.

Fig 10.1 Source code for importing Evaluation modules

Hyperparameter optimization or tuning is the process of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. Hyperparameters tuning is crucial as they control the overall behaviour of a machine learning model. Hyperparameter tuning can be very advantageous to improve the accuracy of machine learning models.

In this project I have used GridSearcCV method for the parameter tuning of the model.

# 10.1 DECISION TREE

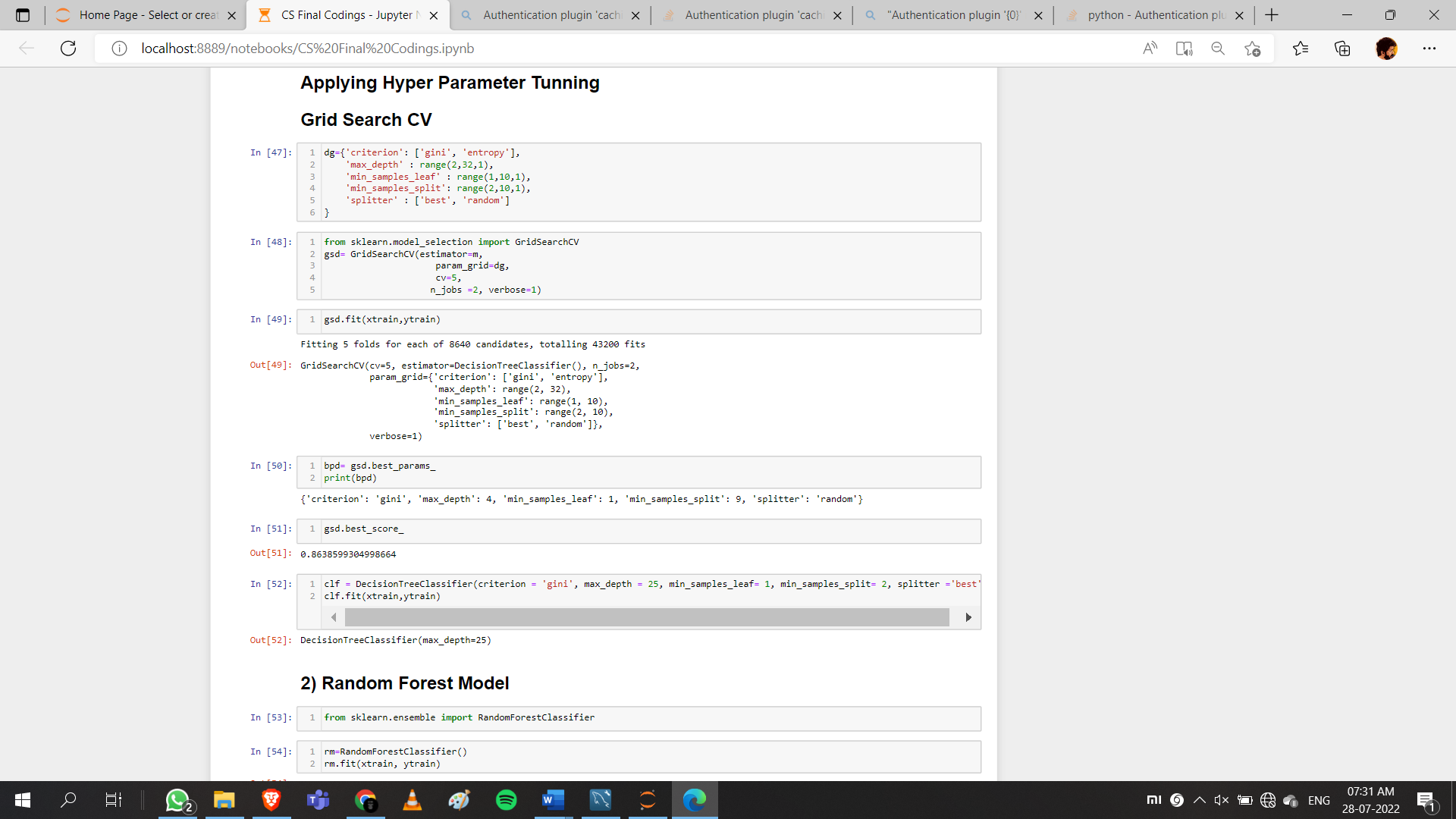


**Figure 21 Source code for evaluation of decision tree classifier**

# 10.1.1 HYPER PARAMETER TUNING FOR DECISION TREE MODEL

**i) GridSearchCV**

After training the model using the Decision Tree Classifier and then the model is tuned by using GridSearchCV for the purpose of increasing the accuracy of the model.

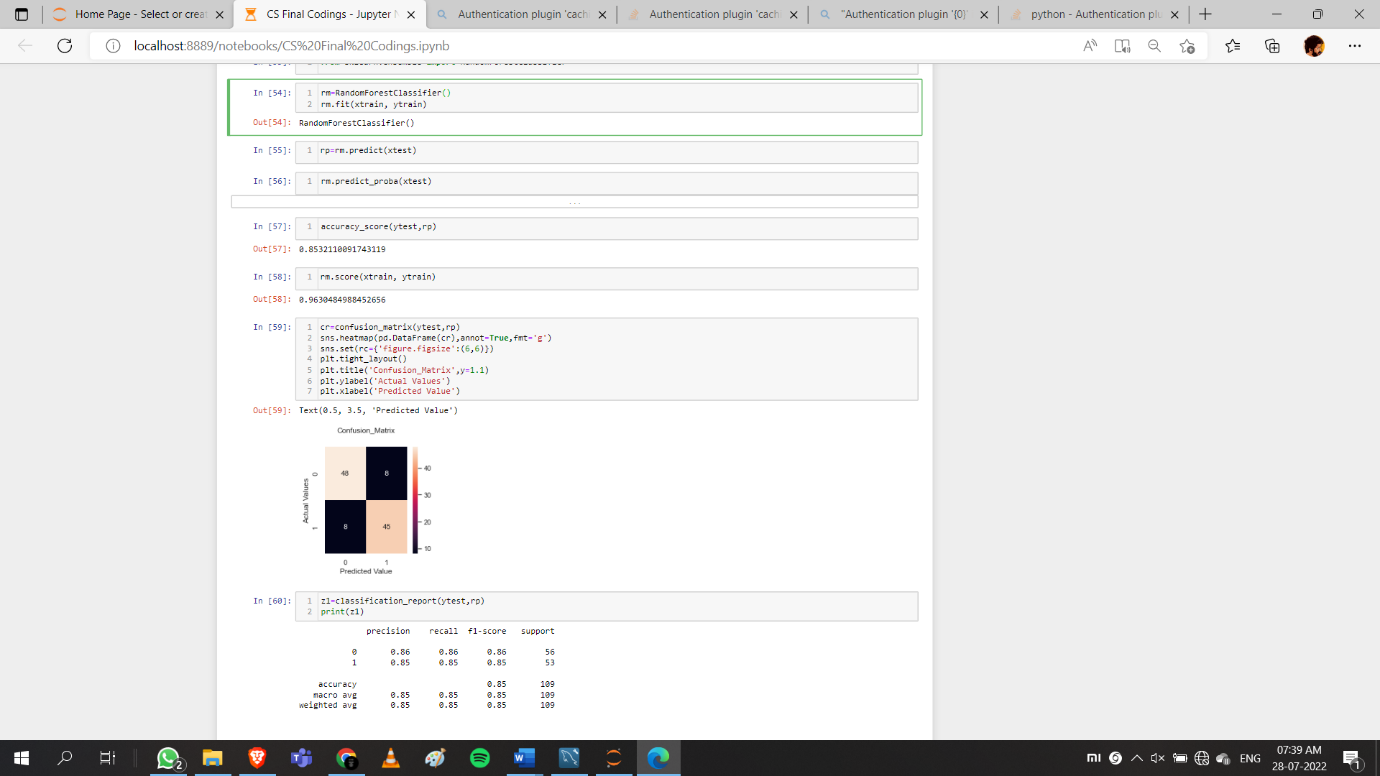


**Figure 22 Source code for Hyper parameter tuning – Decision tree Model**

The Accuracy\_Score before Hyperparameter tuning is : 0.8165

The Accuracy\_Score before Hyperparameter tuning is : 0.8638

# 10.2 RANDOM FOREST



**Figure 23 Source code for Random Forest Classifier evaluation**

# 10.2.1 HYPER PARAMETER TUNING FOR RANDOM FOREST MODEL

**i) GridSearchCV**

Graphical user interface, text, application, email

Description automatically generatedAfter training the model using the Random Forest Classifier and then the model is tuned by using GridSearchCV for the purpose of increasing the accuracy of the model.

**Figure 24 Source code for Hyper parameter tuning – Random Forest Model**

The Accuracy\_Score before Hyperparameter tuning is : 0.8532

The Accuracy\_Score before Hyperparameter tuning is : 0.8638

# 10.3 GRADIENT BOOSTING

Graphical user interface, application

Description automatically generated

**Figure 25 Source code for Gradient Boosting evaluation**

The Accuracy\_Score : 0.8807

# 10.4 SUPPORT VECTOR MACHINE

# 

Figure 26 Source code for SVM evaluation

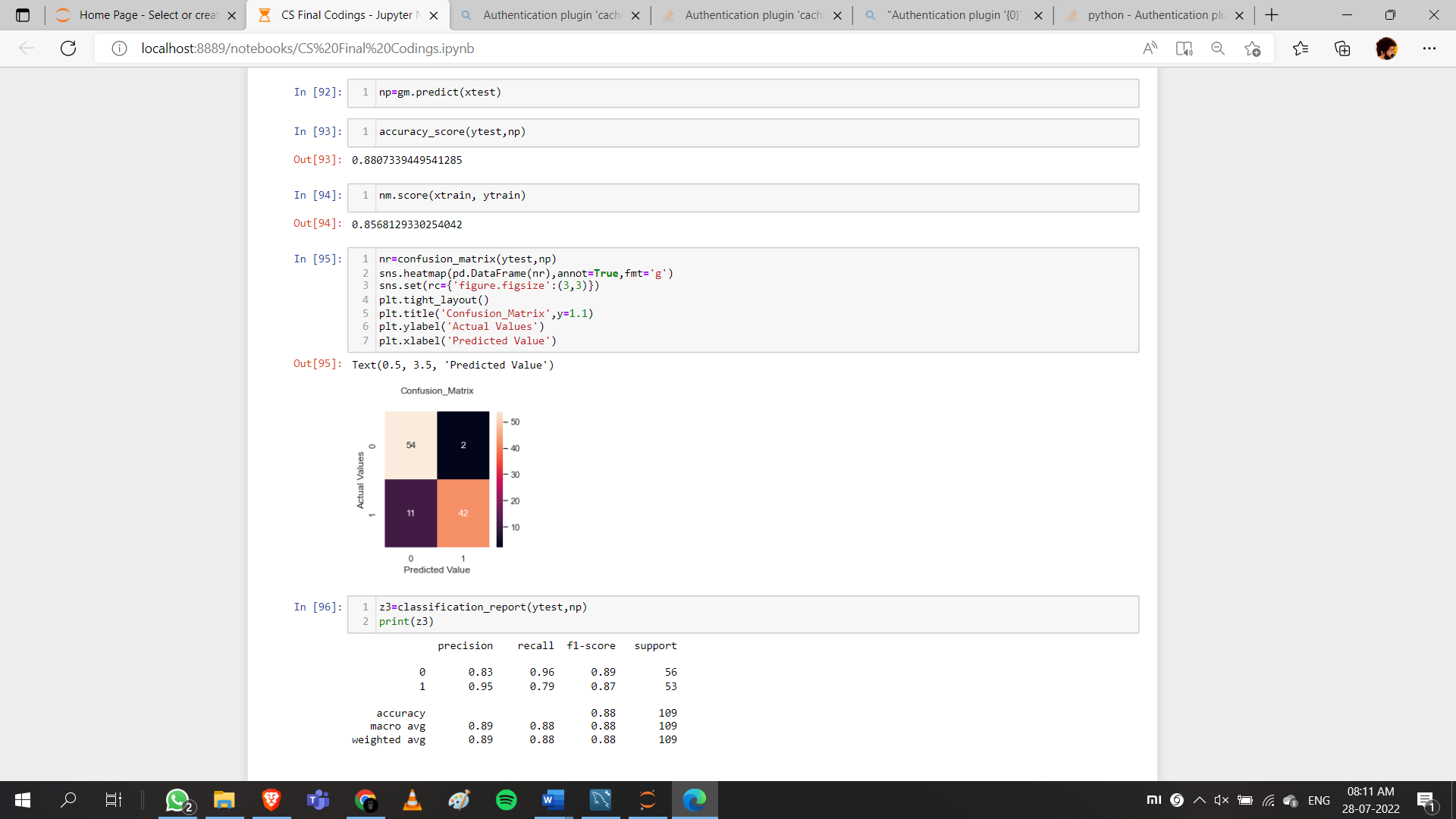
The Accuracy\_Score : 0.8532

# 10.5 KNEIGHBORS

**Figure 26 Source code for KNN evaluation**

The Accuracy\_Score : 0.8532

# 10.6 NAIVE-BAYES



**Figure 27 Source code for NAIVE-BAYES evaluation**

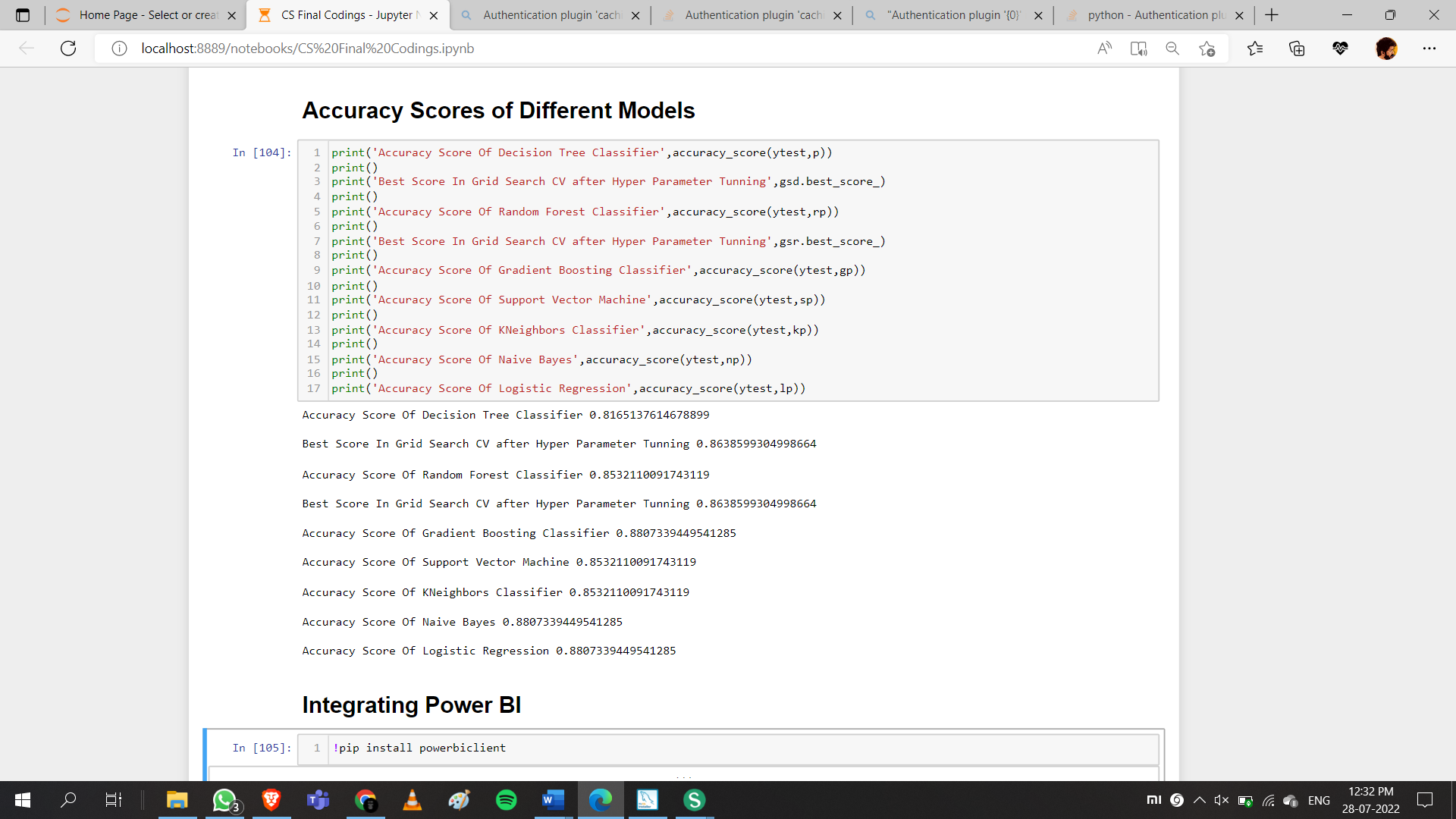
The Accuracy\_Score : 0.8807

# 10.7 LOGISTIC REGRESION

**Figure 28 Source code for LOGISTIC REGRESSION evaluation**

The Accuracy\_Score : 0.8807

# 10.8 ACCURACY SCORES OF EACH MODELS



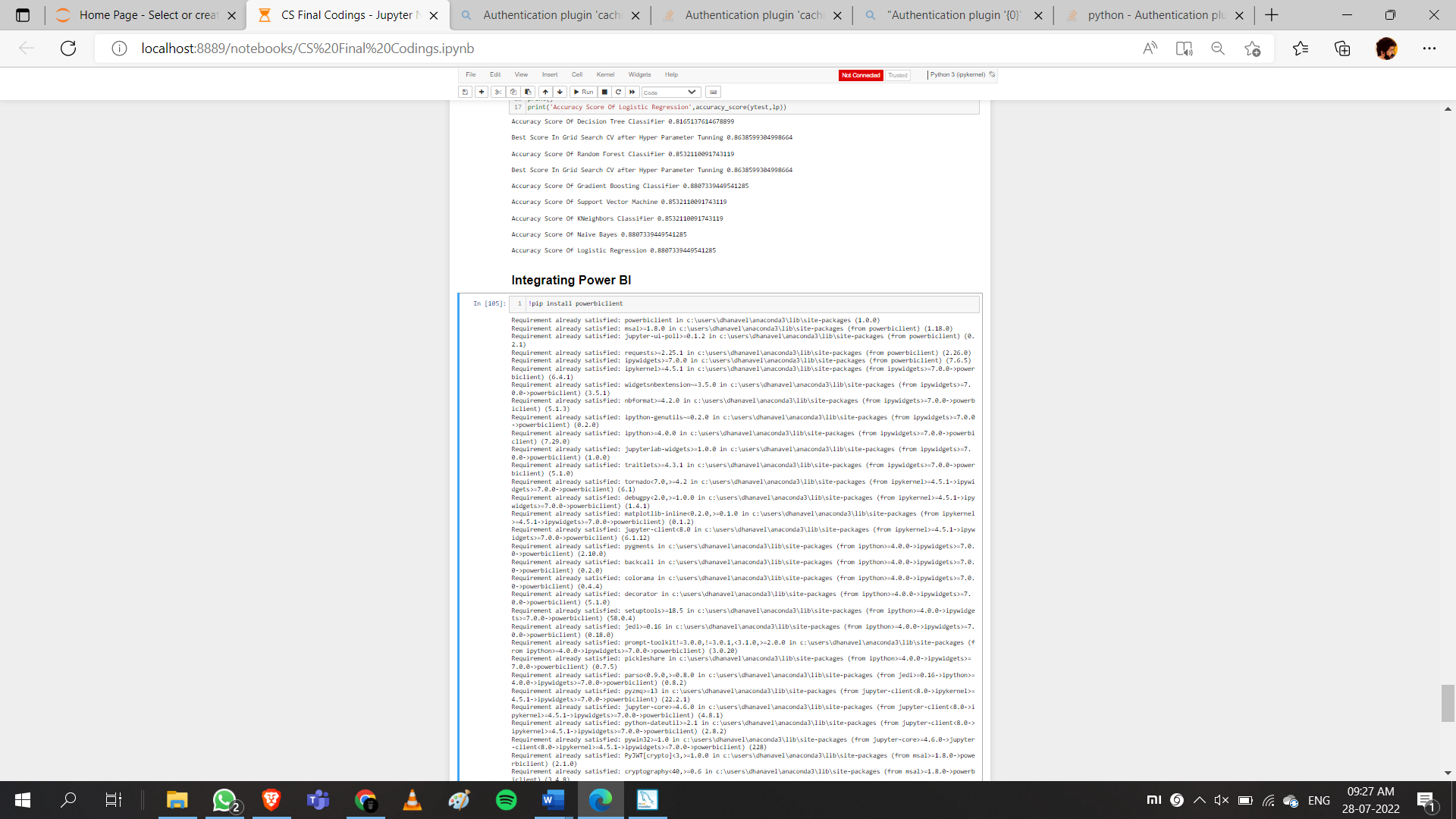
**Figure 29 Accuracy Scores of each model**

# CHAPTER 11

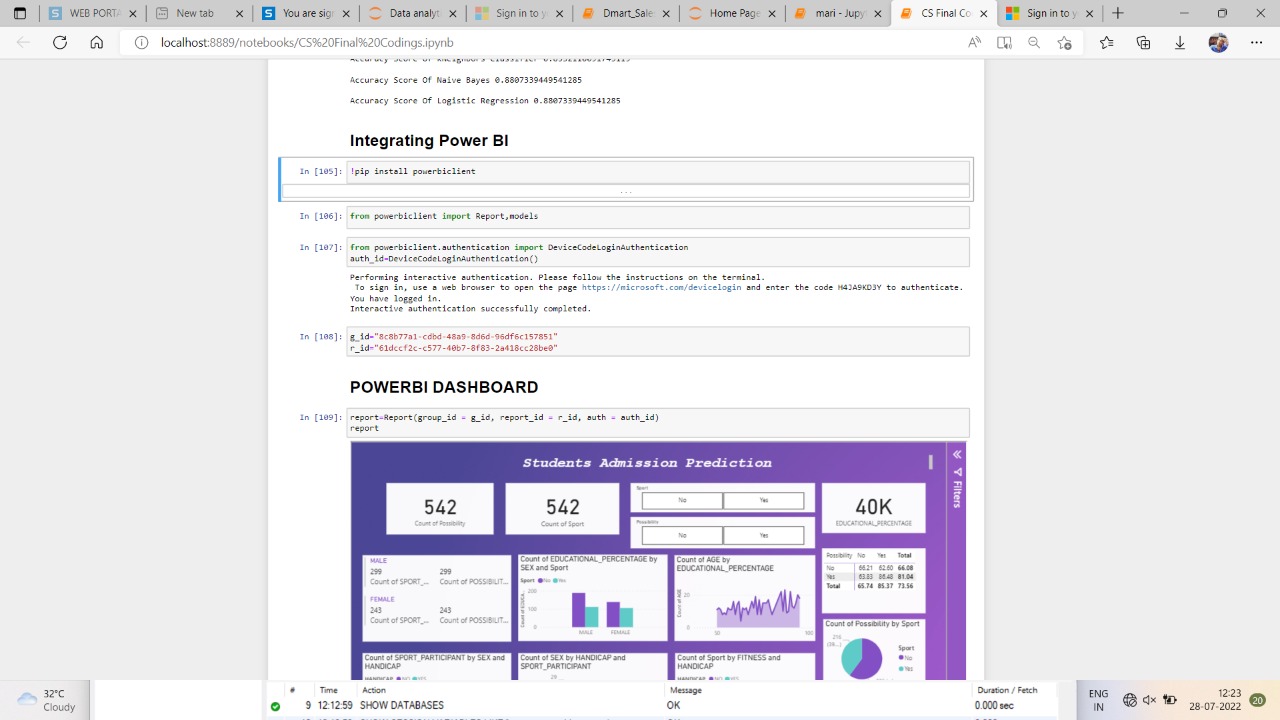
# INTEGRATION OF POWERBI IN JUPYTER USING PYTHON LIBRARIES

Power BI service is a secure Microsoft hosted cloud service that lets users view dashboards, reports, and Power BI apps —a type of content that combines related dashboards and reports — using a web browser or via mobile apps for Windows, iOS, and Android.

Power BI is a collection of software services, apps, and connectors that work together to turn your unrelated sources of data into coherent, visually immersive, and interactive insights.

Power bi can be integrated with python by using some of the python codes and libraries, Power bi is used to visually represent the predictions and conclusions that are made using the machine learning models.

**Figure 30 Installing PowerBI library in Jupyter Notebook.**



**Figure 31 Source code for importing PowerBI client**

# POWERBI DASHBOARD

Graphical user interface, application, Teams

Description automatically generated

**Figure 32 PowerBI Dashboard**

# CONCLUSION

In this project, I have analysed that the possibility of acquiring admission of student based on their academic performance and active sports participation, the dataset is integrated from the SQL server using python libraries and then the dataset is cleaned and pre-processed for the exploratory data analysis and trained and tested using various classification models and accuracy of the model is further increased by performing parameter tuning and then we integrated power bi with python and done visualization on powerBI dashboard.

# REFERENCE

[1] Dr. R. S. Kamath, Dr. S. S. Jamsandekar, and Dr. P. G. Naik, “Machine Learning Approach for Employee Attrition Analysis,” *International Journal of Trend in Scientific Research and Development*, no. Special Issue-FIIIIPM2019, pp. 62–67, Mar. 2019, doi: 10.31142/ijtsrd23065.

[2] “Employee Attrition Prediction Using Machine Learning and Sentiment Analysis,” *International Journal of Advanced Trends in Computer Science and Engineering*, no. 5, pp. 7550–7557, Oct. 2020, doi: 10.30534/ijatcse/2020/91952020.

[3] S. S. Et.al, “Analysis of Employee Attrition using for Machine Learning Techniques,” *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, no. 6, pp. 28–31, Apr. 2021, doi: 10.17762/turcomat.v12i6.1253.

[4] A. Qutub, A. Al-Mehmadi, M. Al-Hssan, R. Aljohani, and H. S. Alghamdi, “Prediction of Employee Attrition Using Machine Learning and Ensemble Methods,” *International Journal of Machine Learning and Computing*, no. 2, pp. 110–114, Mar. 2021,

doi: 10.18178/ijmlc.2021.11.2.1022.

[5] A. Qutub, A. Al-Mehmadi, M. Al-Hssan, R. Aljohani, and H. S. Alghamdi, “Prediction of Employee Attrition Using Machine Learning and Ensemble Methods,” *International Journal of Machine Learning and Computing*, no. 2, pp. 110–114, Mar. 2021,

doi: 10.18178/ijmlc.2021.11.2.1022.

