

**MINI PROJECT**  
**(2021-2022)**  
**“WINE QUALITY ANALYSIS”**  
**Project Report**



**Institute of Engineering & Technology**

**Submitted By -**

Utkarsh Kulshrestha (191500879)

Rishika Sharma (191500656)

**Under the Supervision Of**

**Mr. Farmanul Haque**

**Technical Trainer**

**Department of Computer Engineering & Applications**

# CONTENTS

Cover Page.....	
Declaration.....	
Certificate.....	
Acknowledgement.....	
Abstract.....	
Details about the hardware and the software.....	
Chapter 1- Data Science Introduction-	
Data Science.....	
Stages of Data Science.....	
Chapter 2- Python Programming Language Basics-	
Why Python?.....	
Operators.....	
Conditional Statements.....	
Modules, Packages and Functions.....	

## Chapter 3- Libraries in Python-

Matplotlib.....

Pandas.....

Numpy.....

## Chapter 4- Data Collection-

Red Wine Classification using Regression.....

Flow Diagram.....

Snapshots from Project.....

Conclusions.....

Bibliography.....



Department of Computer Engineering and Applications  
GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,  
Chaumuhan, Mathura – 281406 U.P (India)

### **Declaration**

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project “**Wine Quality Analysis**”, in partial fulfillment of the requirements for the award of the **Bachelor of Technology** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Mr. Farmanul Haque, Technical Trainer, Dept. of CEA, GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

**Sign:** *UtkarshKulshrestha*

**Name of Candidate:** Utkarsh Kulshrestha

**University Roll No.:** 191500879

**Sign:** *RishikaSharma*

**Name of Candidate:** Rishika Sharma

**University Roll No:** 191500656



**Department of Computer Engineering and Applications**  
**GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,**  
**Chaumuhan, Mathura – 281406 U.P (India)**

## **Certificate**

This is to certify that the project entitled “Wine Quality Analysis”, carried out in Mini Project – II, is a bonafide work by Utkarsh Kulshrestha and Rishika Sharma and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

**Signature of Supervisor:**

**Name of Supervisor:** Mr. Farmanul Haque.

**Date**



Department of Computer Engineering and Applications  
GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,  
Chaumuhan, Mathura – 281406 U.P (India)

## ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor Mr. Farmanul Haque, our technical trainer and supervisor.

He has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn't have been able to complete this project.

And at last but not the least we would like to thank our dear parents for helping us to grab this opportunity to get trained and also my colleagues who helped me find resources during the training.

Thanking You

**Sign:** *UtkarshKulshrestha*

**Name of Candidate:** Utkarsh Kulshrestha

**University Roll No.:** 191500879

**Sign:** *RishikaSharma*

**Name of Candidate:** Rishika Sharma

**University Roll No:** 191500656

# ABSTRACT

As each and every sector of the market is growing, data is building up day by day, we need to keep the record of the data which can be helpful for the analytics and evaluation. Now we don't have data in gigabyte or terabyte but in zetta byte and petabyte and this data cannot be handled with the day to day software such as Excel or Matlab. Therefore in this report we will be dealing with large data sets with the high-level programming language 'Python'. The main goal of this project is to aggregate and analyze the data collected from the different data sources available on the internet. This project mainly focuses on the usage of the python programming language in the field of Wine Quality Check. This language has not only its application in the field of just analyzing the data but also for the prediction of the upcoming scenarios in the Wine Prodction. The purpose of using this specific language is due to its versatility, vast libraries (Pandas, Numpy, Matplotlib, etc.), speed limitations, and ease of learning. We will be analyzing large wine data sets in this project which cannot be easily analyzed in other tools as compared to python. Python does not have its limitation to only data analytics but also in many other fields such as Artificial intelligence, Machine learning, and many more.

# Details about the Hardware and the Software

***System Requirements: - Windows 7/8/10***

## **Software Required:**

- Technology Implemented: Machine Learning and Data Science.
- Language : Python.
- Database: Google Collab, VS Code
- Browser: Google Chrome

## ***Hardware Requirements: -***

- Processor: Intel i3
- Operating System: Windows 7/8/10
- RAM: 4+GB
- Hard Disk: 64 GB
- Hardware Devices: Computer System.



# CHAPTER 1 – DATA SCIENCE INTRODUCTION

## DATA SCIENCE

Data science is the field of data analytics and data visualization in which raw data or the unstructured data is cleaned and made ready for the analysis purpose. Data scientists use this data to get the required information for the future purpose. "Data science uses many processes and methods on the big data, the data may be structured or unstructured". Data frames available on the internet is the raw data we get. It may be either in unstructured or semi structured format. This data is further filtered, cleaned and then number of required task are performed for the analysis with the use of the high programming language. This data is further analyzed and then presented for our better understanding and evaluation.

One must be clear that data science is not about making complicated models or making awesome visualization neither it is about writing code but about using the data to create an impact for your company, for this impact we need tools like complicated data models and data visualization.

## STAGES OF DATA SCIENCE

There are many tools used to handle the big data available to us. "Data scientists use programming tools such as Python, R, SAS, Java, Perl, and C/C++ to extract knowledge from prepared data".

Data scientists use many algorithms and mathematical models on the data.

Following are the stages and their cycle performed on the unstructured data.

- Identifying the problem.
- Identify available data sources
- Identify available data sources
- Identify if additional data sources are needed.
- Statistical analysis
- Implementation, development
- Communicate results
- Maintenance



7 steps that together constitute this life-cycle model of Data science

Data science finds its application in many fields. With the assistance of data science it is easy to get the search query on search engines in plenty of time. A role of the data scientist is to have a deep understanding of the data as well as a good command on the programming language, he should also know how to work with the raw data extracted from the data source. Many programming languages are used to analyze and evaluate the data such as Python, Java, MATLAB, Scala, Julia, R., SQL and Tensor Flow. Among which python is the most user friendly and vastly used programming language in the field of data Science .

This life cycle is applied in each and every field , in this project we will be considering all this seven stages of data science to analyze the data. The process will be starting from data collection, data preparation, data modeling and finally data evaluation.

# CHAPTER 2- PYTHON PROGRAMMING LANGUAGE BASICS.

## WHY PYTHON?

"Python is an interpreted, object-oriented, high-level programming language with dynamic semantics . This language consist of mainly data structures which make it very easy for the data scientists to analyse the data very effectively. It does not only help in forecasting and analysis it also helps in connecting the two different languages. Two best features of this programming language is that it does not have any compilation step as compared to the other programming language in which compilation is done before the program is being executed and other one is the reuse of the code, it consist of modules and packages due to which we can use the previously written code anywhere in between the program whenever is required.

There are multiple languages for example R., Java, SQL, Julia, Scala, MATLAB available in market which can be used to analyze and evaluate the data, but due to some outstanding features python is the most famous language used in the field of data science.

Python is mostly used and easy among all other programming languages is due to the following reasons.

# Data structures in Python

Data structures are the way of storing the data so that we can easily perform different operations on the data whenever it's required. When the data has been collected from the data source the data is available in different forms. So later it is easy for the data scientists to perform different operations on the data once it is sorted in to different data structures.

Data structures are mainly classified in to two categories and then further their subcategories shown below.

## 1. Primitive Data Structures.

They are also called as basic data structures. This type of data structures contains simple values of the data.[7]

- **Integers** - All the whole numbers from negative infinity to positive infinity comes under integer data types. for example 4,9,-2,-6.
- **Float** - The decimal figure numbers or rational numbers comes under float data types. for example 3.1,2.2,8.96
- **Strings** - Collection of alphabets or characters are called strings. We enclose the string either in single or double quotes in python. for example 'hello' and "bread".
- **Boolean** - These are the built in data types which take two values that are 'True' and 'False'. True represents the 1 and False represents 0 in python.

## 2. Non-Primitive Data Structures

These are the derived type or reference variables data structures. They are called derived data structures because they derived from the basic data structures such as integer and float. Python has mainly five types of data structures.

Following are the non primitive data structures.

- **Array** - Array is the collection of data types of same type. Arrays data structures are used mostly in the NumPy library of python. In the below example we have first imported the pack-age array from numpy library and defined the array as variable arr then divided the array by 7 and we have printed our array to get output.


 jupyter Array example Last Checkpoint: 5 minutes ago (unsaved changes)

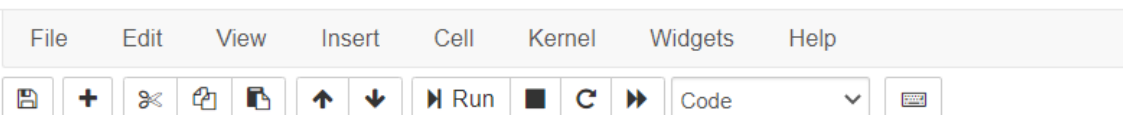


```
In [3]: from numpy import array
arr = array([28,35,42,56])
arr = arr/7
print(arr)

[4.  5.  6.  8.]
```

- **List** - A list is a value that contains multiple values in an ordered sequence". Values in the list referred to list itself, that is the value can be stored in a variable or passed to a function. List are changeable and values in the list are enclosed inside a square bracket, we can perform multiple operations such as indexing, slicing, adding and multiplying.

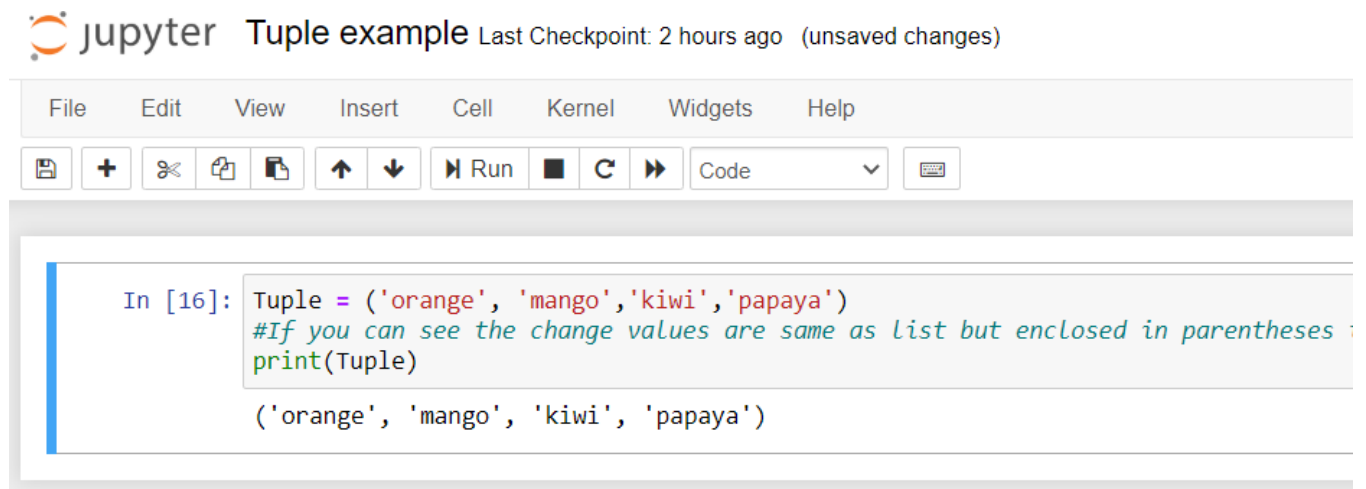
 jupyter List example Last Checkpoint: an hour ago (unsaved changes)



```
In [13]: List = ['orange', 'mango', 'kiwi', 'papaya']
print(List)

['orange', 'mango', 'kiwi', 'papaya']
```

- **Tuple** - A tuple is a list of non changeable objects. The differences between tuples and lists are that the tuples cannot be changed, tuples use parentheses, whereas list uses squarebrackets.



The image shows a Jupyter Notebook interface. At the top, the title bar says "jupyter Tuple example" followed by "Last Checkpoint: 2 hours ago (unsaved changes)". Below the title bar is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". Under the menu bar is a toolbar with icons for saving, adding, deleting, copying, pasting, undo, redo, and running code. The main area shows a code cell with the following content:

```
In [16]: Tuple = ('orange', 'mango', 'kiwi', 'papaya')
          #If you can see the change values are same as list but enclosed in parentheses
          print(Tuple)

          ('orange', 'mango', 'kiwi', 'papaya')
```

- **Dictionary**- These are nothing but a type of data structure which consist of key value pairs enclosed in the curly brackets. It is same as the any dictionary we use in day to day life in which we find the meaning of the particular words. So if I compare normal dictionary to this python dictionary data structure then the a word in a dictionary will be our key and its meaningwill be the value of the dictionary. In the figure name, occupation and hobby are the keys and Suraj, data analyst and vlogging are the values assigned to the keys.

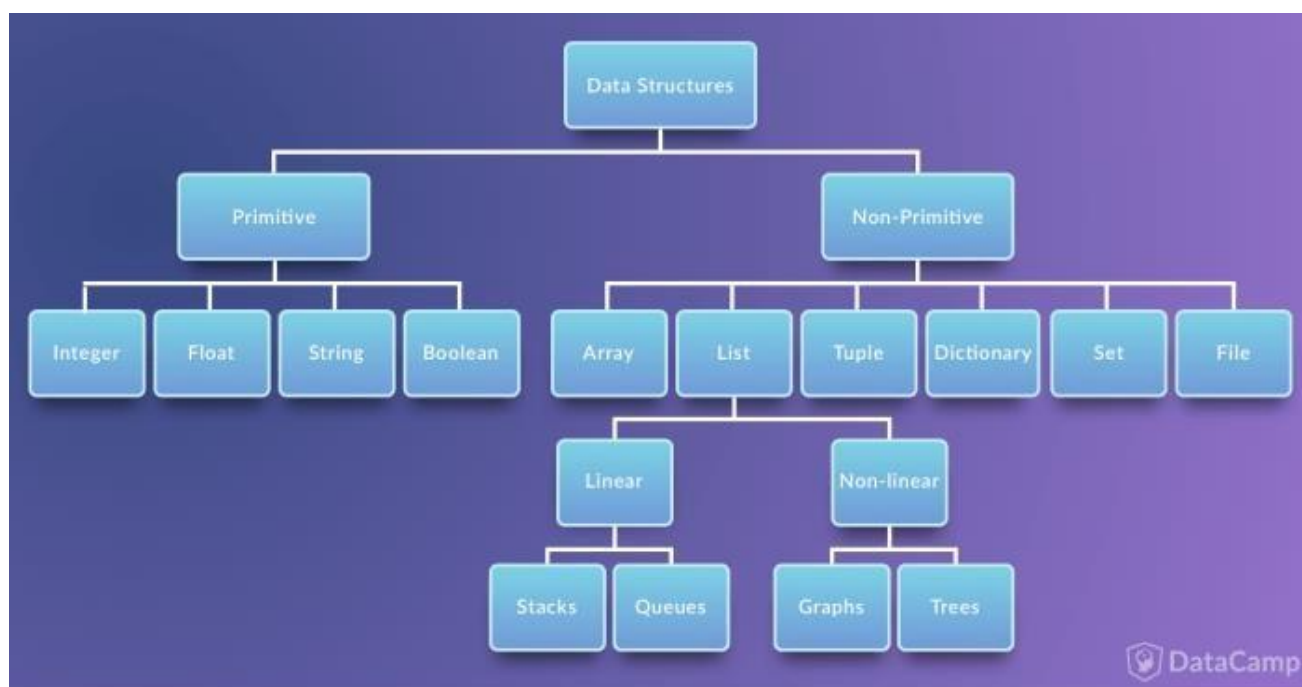
```
File Edit View Insert Cell Kernel Widgets Help
[Icons] Run Code [Dropdown]

In [25]: dictionary = {'name': 'Suraj', 'occupation': 'data analyst', 'hobby': 'vlogging'}
         dictionary['hobby']

Out[25]: 'vlogging'
```

- **Sets** - Set are used for calculating mathematical operation such as union, intersection and sym-metric difference.

Below is the data structure tree which explains the category and sub-category of each data types.



# Operators

**OPERATORS** - Operators are the symbols in python that are used to perform Arithmetic or logical operations. Following are the different types of operators in python.

**Arithmetic operators** - Arithmetic operators carry out mathematical operations and they are mostly used with the numeric values.

Arithmetic operators		
Operator	Name	Example
+	Addition	A+B
-	Subtraction	A-B
*	Multiplication	A*B
/	Division	A/B
%	Modulus	A%B
**	Exponentiation	A**B
//	Quotient	A//B

Table 2.1: Arithmetic

operators A and B are the numeric values.



**Assignment operators** - As the name decides this operators are used for assigning the value to the variables.

ASSIGNMENT OPERATORS		
Operator	Example	may also be writ-ten
=	a = 6	a = 6
+=	a += 3	a = a + 3
-=	a -= 4	a = a - 4
*=	a *= 5	a = a * 5
/=	a /= 6	a = a / 6
%=	a %= 7	a = a % 7
//=	a //= 8	a = a // 8
**=	a **= 9	a = a ** 9
&=	a &= 1	a = a & 1

Table 2.2: Assignment Operators

Here a is any value and number of operations are performed on this value.

**Logical operators** - These operators are used to join conditional statements

Logical Operators		
Operator	Description	Example
and	if both statements are true it returns true	x <5 <b>and</b> x <10
or	if any of the two statement is true it returns true	x <4 <b>or</b> x <8
not	if the result is true it reverses the result and gives false	<b>not</b> (x <4 <b>and</b> x <8)

Table 2.3: Logical Operators

Here a is any value provided by us and on which multiple operations can be performed.

**Comparison operators** - These operators are used to compare two different values.

Comparison operators		
Operator	Name	Example
==	Equal	a == b
!=	Not equal	a!=b
>	Greater than	a >b
<	less than	a <b
>=	Greater than equal to	a>= b
<=	less than equal to	a <=b

Table 2.4: Comparison operators

Here a and b are two different values and these values are compared.

**Membership operators** - These operators are used to check membership of a particular value. It is used to check whether a specific value is present in the object or not.

Membership operators		
Operator	Description	Example
in	it returns a True if the value is present inside the object	a <b>in</b> b
not in	it returns a True if the value is not present inside the object	a <b>not in</b> b

Table 2.5: Membership operators.

# Conditional statements

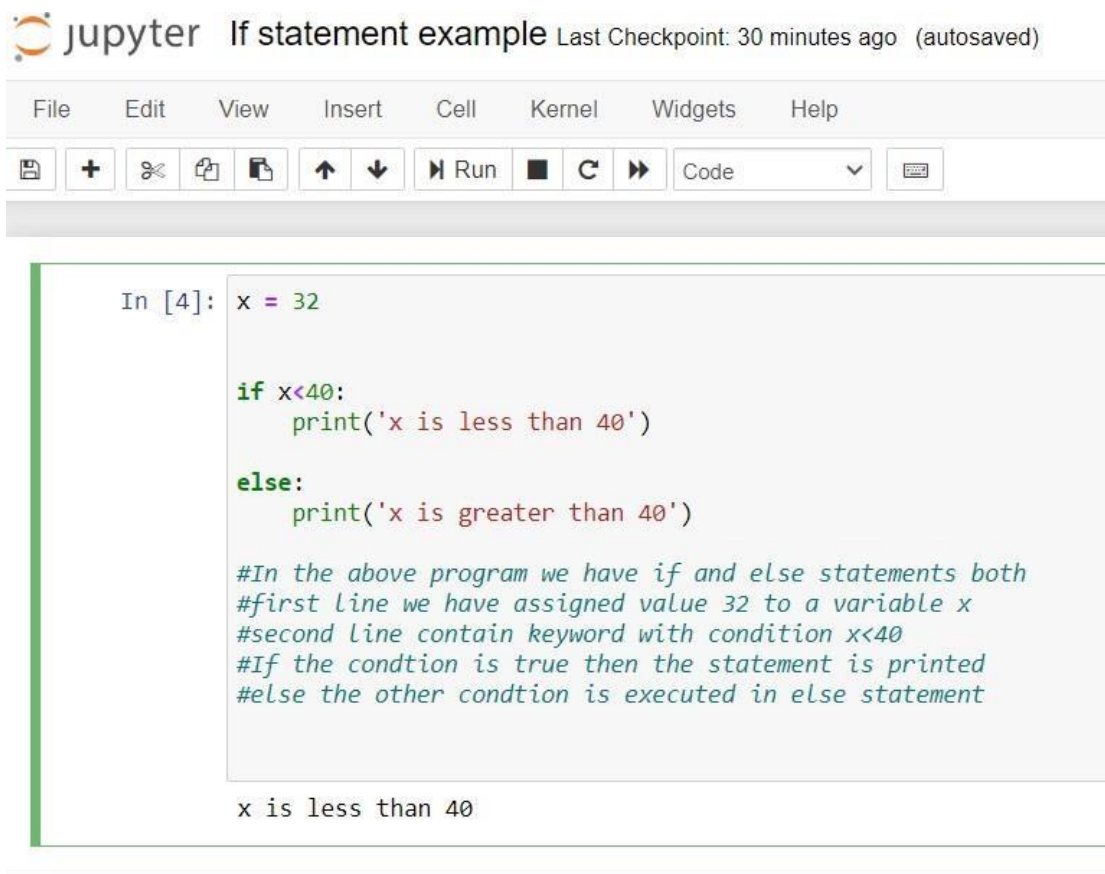
## If else statements

"The most common type of statement is the if statement. if statement consist of a block which is called as clause", it is the block after if statement, it executed the statement if the condition istrue. The statement is omitted if the condition is False. then the statement in the else part is printed

If statement consist of following -

- If keyword itself
- Condition which may be True or False
- Colon
- If clause or a block of code

Below is the figure shows how If and else statements are used with description inside it.



The image shows a Jupyter Notebook interface. The title bar reads "jupyter If statement example" followed by "Last Checkpoint: 30 minutes ago (autosaved)". The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The toolbar contains icons for saving, adding cells, undo, redo, running, and other standard Jupyter actions. The code cell, labeled "In [4]:", contains the following Python code:

```
x = 32

if x<40:
    print('x is less than 40')

else:
    print('x is greater than 40')
```

Below the code, there are several lines of comments explaining the logic:

```
#In the above program we have if and else statements both
#first line we have assigned value 32 to a variable x
#second line contain keyword with condition x<40
#If the condition is true then the statement is printed
#else the other condition is executed in else statement
```

The output of the code cell is "x is less than 40".

x

## **elif statements**

In this statement only one statement is executed, There are many cases in which there is only one possibility to execute. "The elif statement is an else if statement that always follows an if or another elif statement"[8]. The elif statement provides another condition that is checked only if any of the previous conditions were False. In code, an elif statement always consists of the following:. The only difference between if else and elif statement is that in elif statement we have the condition where as in else statement we do not have any condition.

elif statement consist of following -

- elif keyword itself
- Condition which may be True or False
- Colon
- elif clause or a block of code

Below is the figure shows how elif statement is used with description inside it.

```
File Edit View Insert Cell Kernel Widgets Help
```

```

In [9]: var = 't'

if var == 'a':
    print('this is the vowel a')
elif var == 'e':
    print('this is the vowel e')
elif var == 'i':
    print('this is the vowel i')
elif var == 'o':
    print('this is the vowel o')
elif var == 'u':
    print('this is the vowel u')
else:
    print('The value in variable var is constant')

#In the above program we have if, else and elif statements.
#first line we have assigned value t to a variable 'var'
#second line contain keyword if with condition if var==a
#If the condtion is true then the statement is printed
#elif the other condtion is executed in elif statement in furhter lines
#if all the conditions are not true then we come to else statement
#and then statement in else block is printed on the output

```

The value in variable var is constant

Figure 2.7: elif example



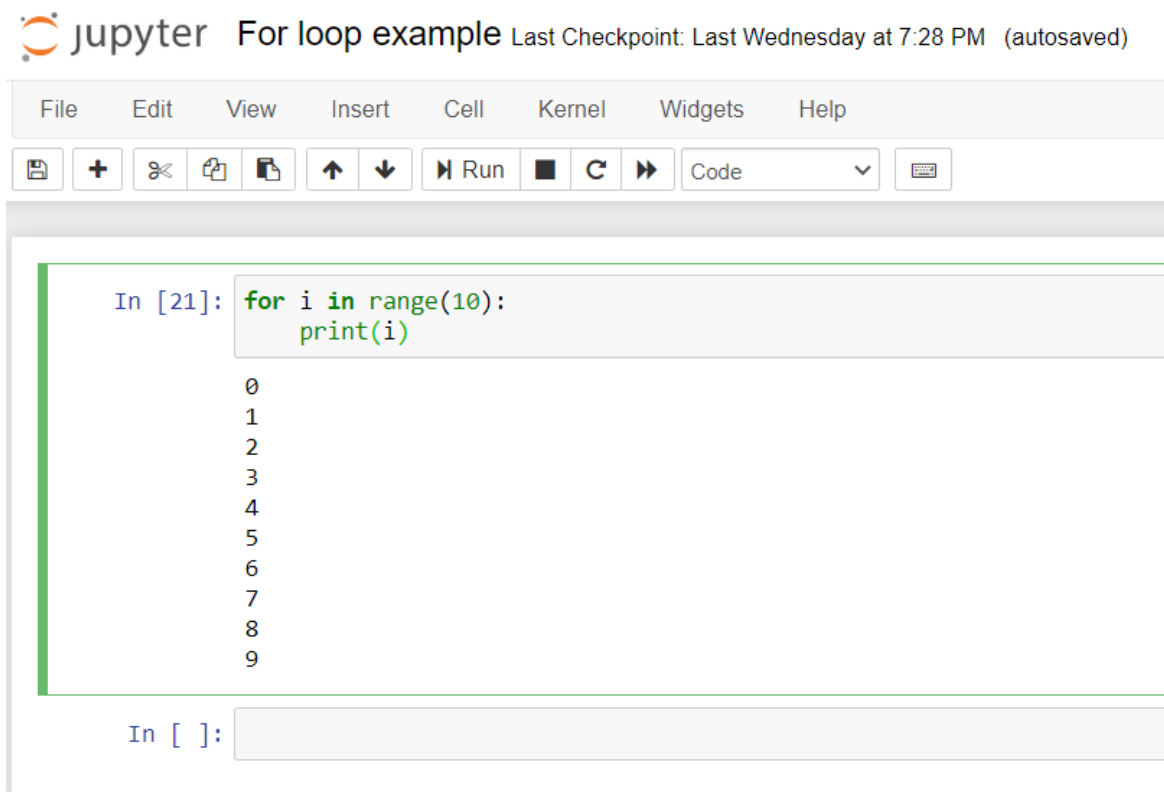
# Loops in python

## For loop

### When do we use for loops ?

for loops are traditionally used when you have a block of code which you want to repeat a fixed number of times. The Python for statement iterates over the members of a sequence in order, executing the block each time.[9]

**Range statement** - This statement 'range()' is used with for loop statements where you can specify one value. For example, if you specify 10, the loop statement starts from 1 and ends with 9, which is n-1. Also, you can specify the start and end values. The following examples demonstrate loop statements.

A screenshot of a Jupyter Notebook interface. The title bar says "jupyter For loop example" and "Last Checkpoint: Last Wednesday at 7:28 PM (autosaved)". The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The toolbar contains icons for saving, adding, deleting, copying, pasting, undo, redo, and running code. The code cell shows the following Python code:

```
In [21]: for i in range(10):  
         print(i)
```

The output of the code is displayed below the code cell, showing the numbers 0 through 9, each on a new line.

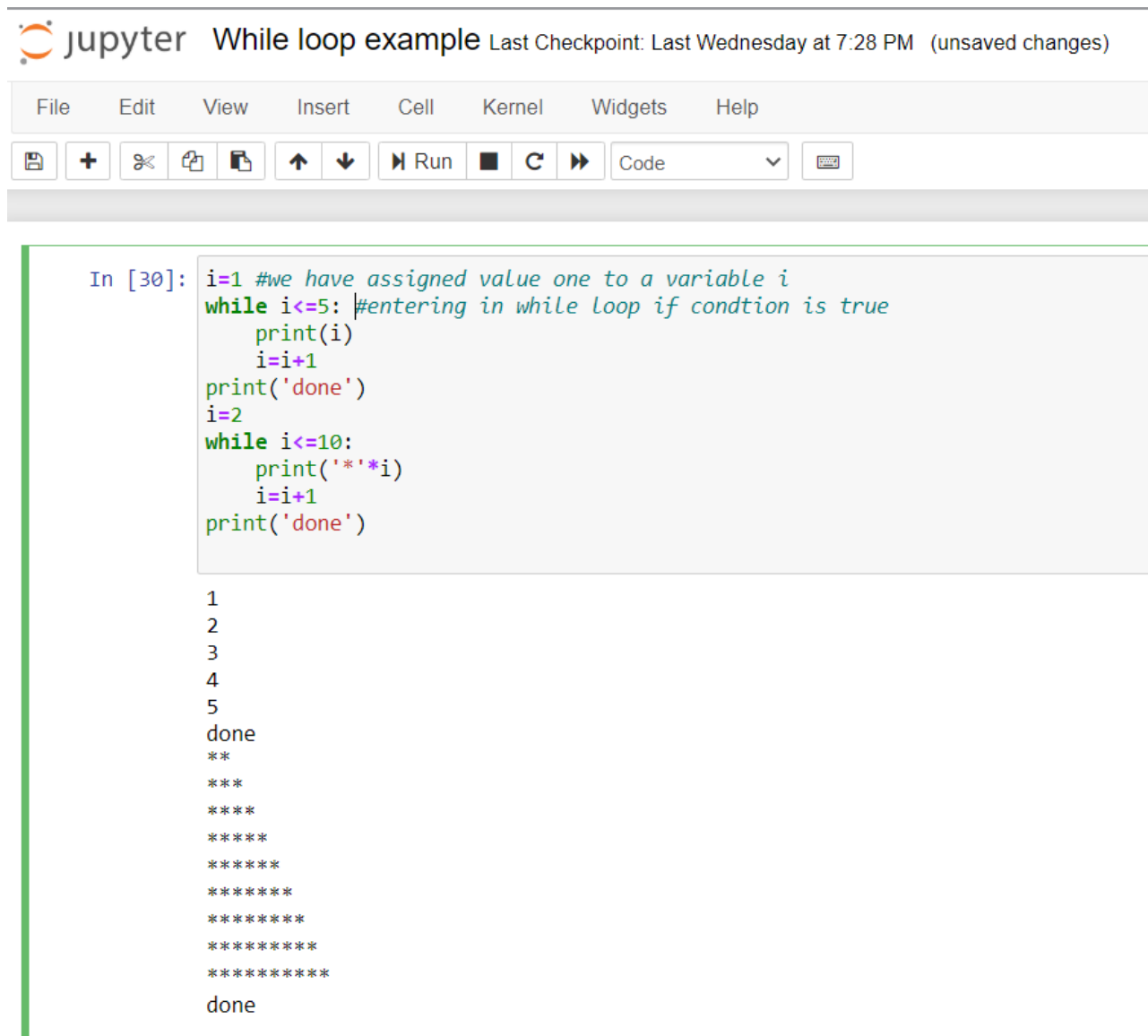
```
0  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

Below the output, there is an input prompt "In [ ]:" followed by an empty text box.

Figure 2.8: for example with range statement

# While loop

While loops are used for repeating the section of code but not same as for loop, the while loop does not run n times, but until a defined condition is no longer met. If the condition is initially false, the loop body will not be executed at all.

The image shows a Jupyter Notebook interface. At the top, the Jupyter logo is followed by the text "While loop example" and "Last Checkpoint: Last Wednesday at 7:28 PM (unsaved changes)". Below this is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". Under the menu bar is a toolbar with icons for saving, adding cells, undo, redo, running, and other functions. The main area contains a code cell with the following Python code:

```
In [30]: i=1 #we have assigned value one to a variable i
while i<=5: #entering in while loop if condtion is true
    print(i)
    i=i+1
print('done')
i=2
while i<=10:
    print('*'*i)
    i=i+1
print('done')
```

The output of the code is displayed below the code cell:

```
1
2
3
4
5
done
**
***
****
*****
*****
*****
*****
*****
*****
*****
done
```

Figure 2.9: While loop example

# Module, Package and Functions

- **Module**

Modules are Python files which has extension as .py. The name of the module will be the name of the file. A Python module can have a set of functions, classes or variables defined and implemented.

Module has some python codes, this codes can define the classes, functions and variables. The reason behind using the module is that it organizes your python code by grouping the python code so that it is easier to use.

- **Package**

A package consist of the collection of modules in which python codes are written with name init.py. It means that each python code inside of the python path, which contains a file named init.py, will be treated as a package by Python. Packages are used for organizing the module by using dotted names.

for example -

We have a package named simple package which consist of two modules a and b. We will import the module from package in following way.

```
from simple package import a, b
```

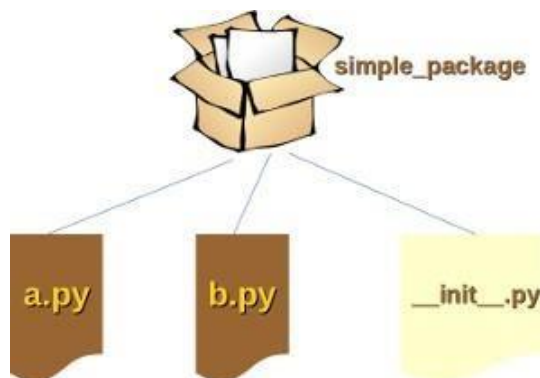
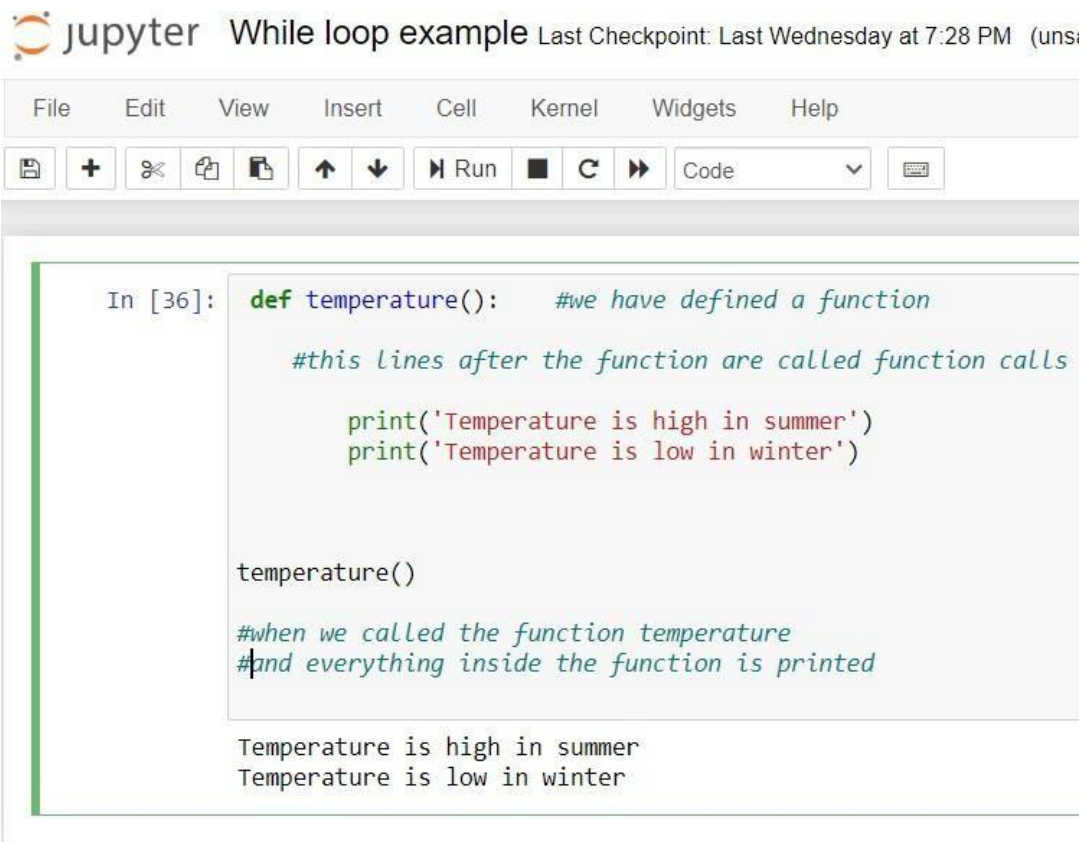


Figure 2.10: packages example [10]

- **Functions**

A function is a python code which can be reused at any anytime in the whole python code. Function performs specific task whenever it is called during the program. With the help of function the program is divided in to multiple codes.

- **Built in functions** - The functions which are already in the python programming and have specific action to perform are called as built in functions. This function are immutable. Some examples of this functions are -  
chr() - used to get string  
print() - used to print an object in terminal min()  
- used to get minimum value interterminal
- **User defined functions** - This functions are user to defined functions and it starts with the key word 'def' as shown in the example below. We have defined the function names temperature and its task to be performed when called. Below is the example of it.



The image shows a Jupyter Notebook interface. At the top, the title bar reads "jupyter While loop example" followed by "Last Checkpoint: Last Wednesday at 7:28 PM (unsaved)". Below the title bar is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Under the menu bar is a toolbar with icons for saving, adding a new cell, deleting a cell, copying, pasting, undo, redo, and running the cell. The main area of the notebook contains a code cell with the following text:

```
In [36]: def temperature():    #we have defined a function

        #this lines after the function are called function calls

        print('Temperature is high in summer')
        print('Temperature is low in winter')

temperature()

#when we called the function temperature
#and everything inside the function is printed

Temperature is high in summer
Temperature is low in winter
```

## Chapter 3

# Libraries in Python

Python library is vast. There are built in functions in the library which are written in C language. This library provide access to system functionality such as file input output and that is not accessible to Python programmers. This modules and library provide solution to themany problems in programming.

Following are some Python

libraries.Matplotlib

Pandas

333333

TensorFlo

wNumpy

Keras

PyTorch

LightGBM

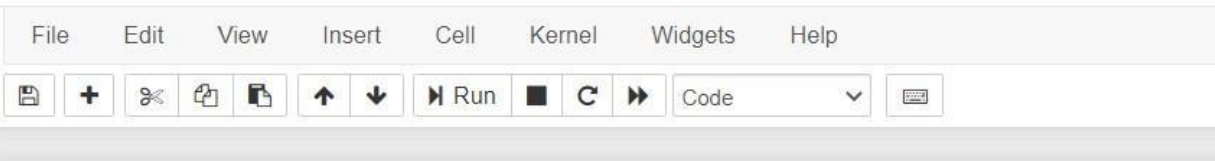
Eli5

SciPy

# Matplotlib

"Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy"[11]. Matlab provides an application that is used in graphical user interface tool kits. Another such library is pylab which is almost same as MATLAB.

It is a library for 2D graphics, it finds its application in web application servers, graphical user interface toolkit and shell. Below is the example of a basic plot in python.



```
In [10]: import matplotlib.pyplot as plt
#we have imported matplotlib library first
#we have imported the pyplot module from matplotlib library
#we have give name 'plt' instead of using whole function name
plt.plot([1,2,3],[1,3,4])
#we used plot function to plot a graph
#we have take simple list in plot function
plt.xlabel('x label') #This function is used to name x axis
plt.ylabel('y label') #This function is used to name y axis
plt.title('basic plot') #This function used for title of grapho
plt.show() #This function show the graph
```

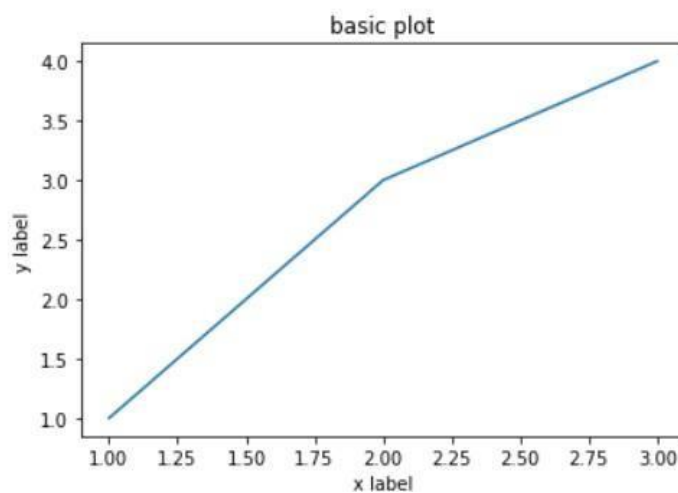


Figure 3.1: Matplotlib basic example



# Pandas

Pandas is also a library or a data analysis tool in python which is written in python programming language. It is mostly used for data analysis and data manipulation. It is also used for datastructures and time series.

We can see the application of python in many fields such as - Economics, Recommendation Systems - Spotify, Netflix and Amazon, Stock Prediction, Neuroscience, Statistics, Advertising, Analytics, Natural Language Processing. Data can be analyzed in pandas in two ways -

**Data frames** - In this data is two dimensional and consist of multiple series. Data is always represented in rectangular table.

**Series** - In this data is one dimensional and consist of single list with index.

File
Edit
View
Insert
Cell
Kernel
Widgets
Help

```

In [7]: #SERIES
import pandas as pd
#We are first importing the Librabry
#and keeping its name as 'pd' for our convenience
odd_numbers = pd.Series([3,9,13,15])
#we have imported series array from pandas
odd_numbers

```

```

Out[7]: 0      3
        1      9
        2     13
        3     15
        dtype: int64

```

```

In [32]: #DATAFRAME - IT HAS TWO OR MORE ARRAYS IN IT
info_stu = {'students':['john','mike','harry','robert'],
            'age':[28,26,29,25],
            'country':['spain','rome','holand','russia']}
#here we have defind our 3 differnet arrays

#then we have imported 'DataFrame' from pandas

info =pd.DataFrame(info_stu)

info

#0,1,2,3 are the index number of different rows

```

```

Out[32]:

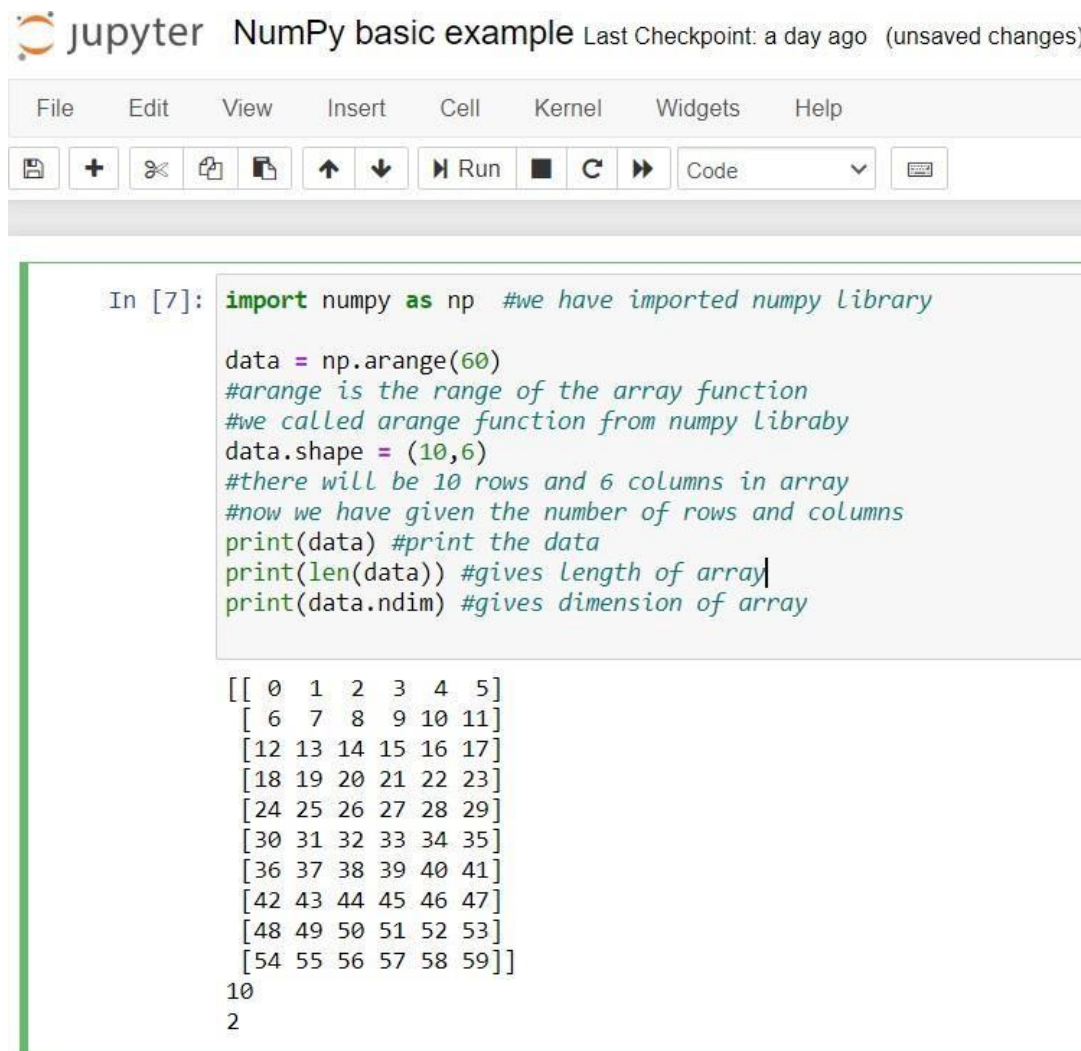
```

	students	age	country
0	john	28	spain
1	mike	26	rome
2	harry	29	holand
3	robert	25	russia

Figure 3.2: series and data frame in pandas

# NumPy

"NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays". The previous similar programming of NumPy is Numeric, and this language was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. [\[12\]](#) It is an open source library and free of cost.



The image shows a Jupyter Notebook interface with the title "NumPy basic example". The interface includes a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". Below the menu bar is a toolbar with icons for saving, adding, deleting, and running code. The main area displays a code cell with the following Python code:

```
In [7]: import numpy as np #we have imported numpy library

data = np.arange(60)
#arange is the range of the array function
#we called arange function from numpy library
data.shape = (10,6)
#there will be 10 rows and 6 columns in array
#now we have given the number of rows and columns
print(data) #print the data
print(len(data)) #gives length of array
print(data.ndim) #gives dimension of array
```

The output of the code is displayed below the code cell:

```
[[ 0  1  2  3  4  5]
 [ 6  7  8  9 10 11]
 [12 13 14 15 16 17]
 [18 19 20 21 22 23]
 [24 25 26 27 28 29]
 [30 31 32 33 34 35]
 [36 37 38 39 40 41]
 [42 43 44 45 46 47]
 [48 49 50 51 52 53]
 [54 55 56 57 58 59]]
10
2
```

Figure 3.3: NumPy basic example

# Chapter 4

## Data collection

### RED WINE CLASSIFICATION USING REGRESSION

#### PROCEDURE:

- Here first we Imported Library then we loaded wine data then we moved on to check if there is any null or missing value in the dataset here this wasn't the case we had preprocessed dataset so we moved on to visualize the correlation of the parameter present in the dataset using `corr ()` after that we plotted it using `matplotlib` and we went to visualize whole dataset using `hist ()` to better insight of the dataset.
- There was one minor issue: so first after we visualized data we took Quality of wine and converted it to categorical value (True or False) since we can say that if a wine has quality value over 5 it's a good wine otherwise it's a not good wine (Here are just classifying wine based on True and False i.e. if it has quality score over 5 and not on values from 3-8) so the next step to change the value of quality.
- After that we can now split our data into training data and testing data and now we use `StandardScaler` to standardize our both data (training as well as testing).
- After we start to build our logistic regression model, then feed our training data and then we move forward to predictions.
- After predictions are done on testing data now it's time to calculate Accuracy score, Precision, Recall and Confusion matrix.

## ABOUT DATA:

fixed acidity: most acids involved with wine or fixed or nonvolatile (do not evaporate readily) volatile acidity : the amount of acetic acid in wine, which at too high of levels can lead

to an unpleasant, vinegar taste citric acid : found in small quantities, citric acid can add 'freshness' and flavor to wines residual sugar: the amount of sugar remaining after fermentation

stops, it's rare to find wines with less than 1 gram/liter and chlorides: the amount of salt in the wine free sulfur dioxide: the free form of  $\text{SO}_2$  exists in equilibrium between molecular

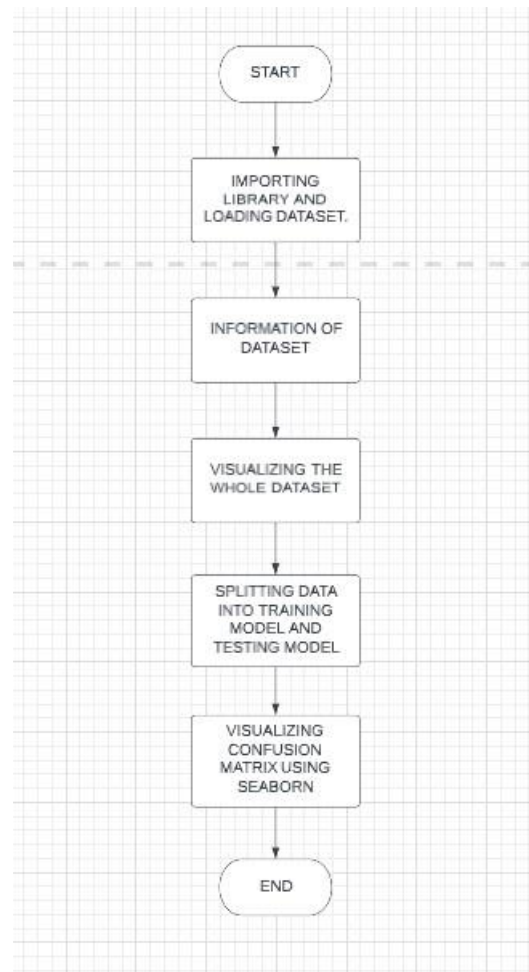
$\text{SO}_2$  (as a dissolved gas) and bisulfate ion; it prevents total sulfur dioxide: amount of free and bound forms of  $\text{SO}_2$ ; in low concentrations,  $\text{SO}_2$  is mostly undetectable in wine, but at free

$\text{SO}_2$  density :the density of water is close to that of water depending on the percent alcohol and sugar content pH: describes how acidic or basic a wine is on a scale from 0 (very acidic)

to 14 (very basic); most wines are between 3-4 on the sulphates: a wine additive which can contribute to sulfur dioxide gas ( $\text{SO}_2$ ) levels, which acts as an antimicrobial an alcohol:

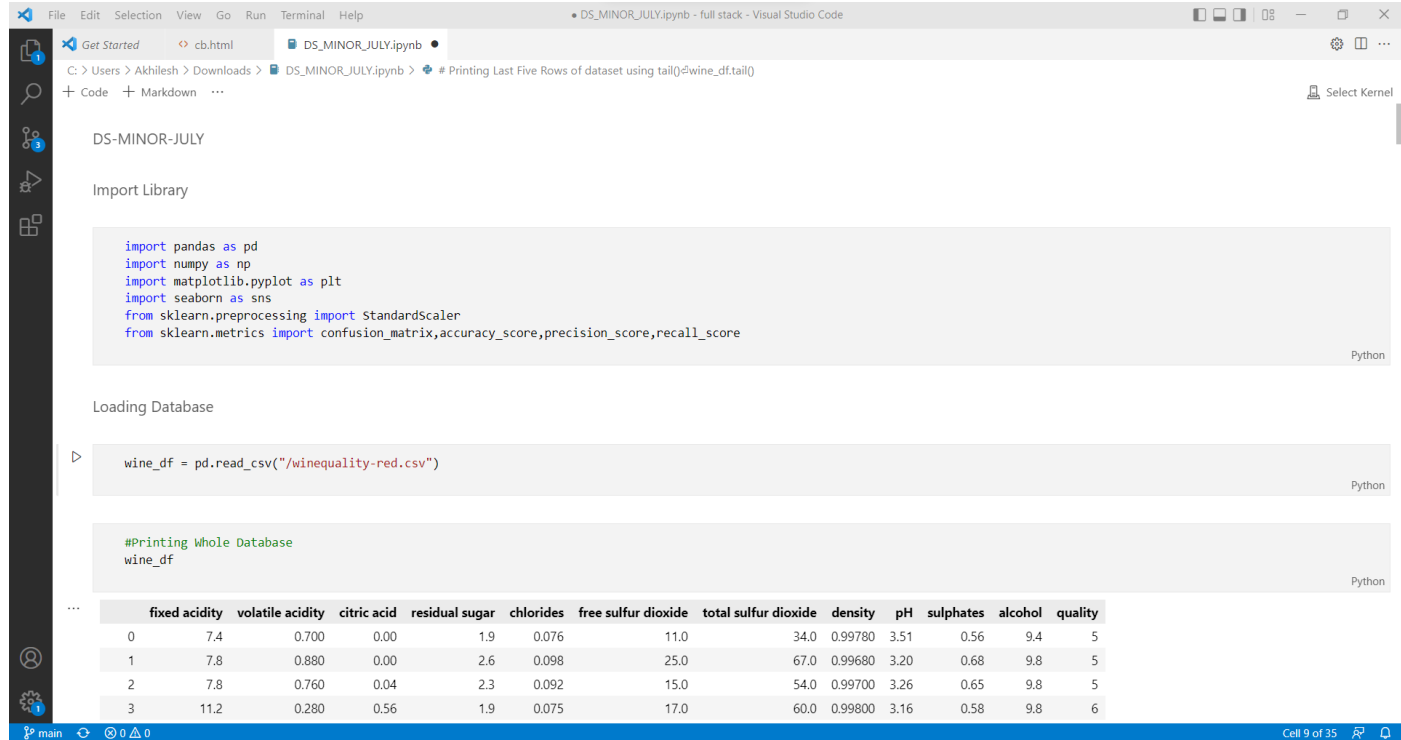
the percent alcohol content of the wine quality: output variable (based on sensory data, score between 0 to 10.

# FLOW DIAGRAM OF PROJECT



# SNAPSHOTS OF PROJECT:-

## 1. Importing Library and Loading Database



The screenshot shows a Jupyter Notebook titled "DS\_MINOR\_JULY.ipynb" in Visual Studio Code. The notebook is divided into three sections: "DS-MINOR-JULY", "Import Library", and "Loading Database".

**DS-MINOR-JULY**

**Import Library**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
```

**Loading Database**

```
wine_df = pd.read_csv("/winequality-red.csv")
```

**#Printing Whole Database**

```
wine_df
```

The output of the code is a table with 13 columns and 4 rows (index 0 to 3).

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6

Cell 9 of 35

## 2. Running Some Operations

The screenshot shows a Jupyter Notebook titled "DS\_MINOR\_JULY.ipynb" in Visual Studio Code. The notebook contains three cells. The first cell runs `wine_df.head()` and displays the first five rows of the dataset. The second cell runs `wine_df.tail()` and displays the last five rows. The third cell runs `wine_df.info()` to show the overall information about the dataset.

**Cell 1: Printing First Five Rows of dataset using head()**

```
wine_df.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

**Cell 2: Printing Last Five Rows of dataset using tail()**

```
wine_df.tail()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

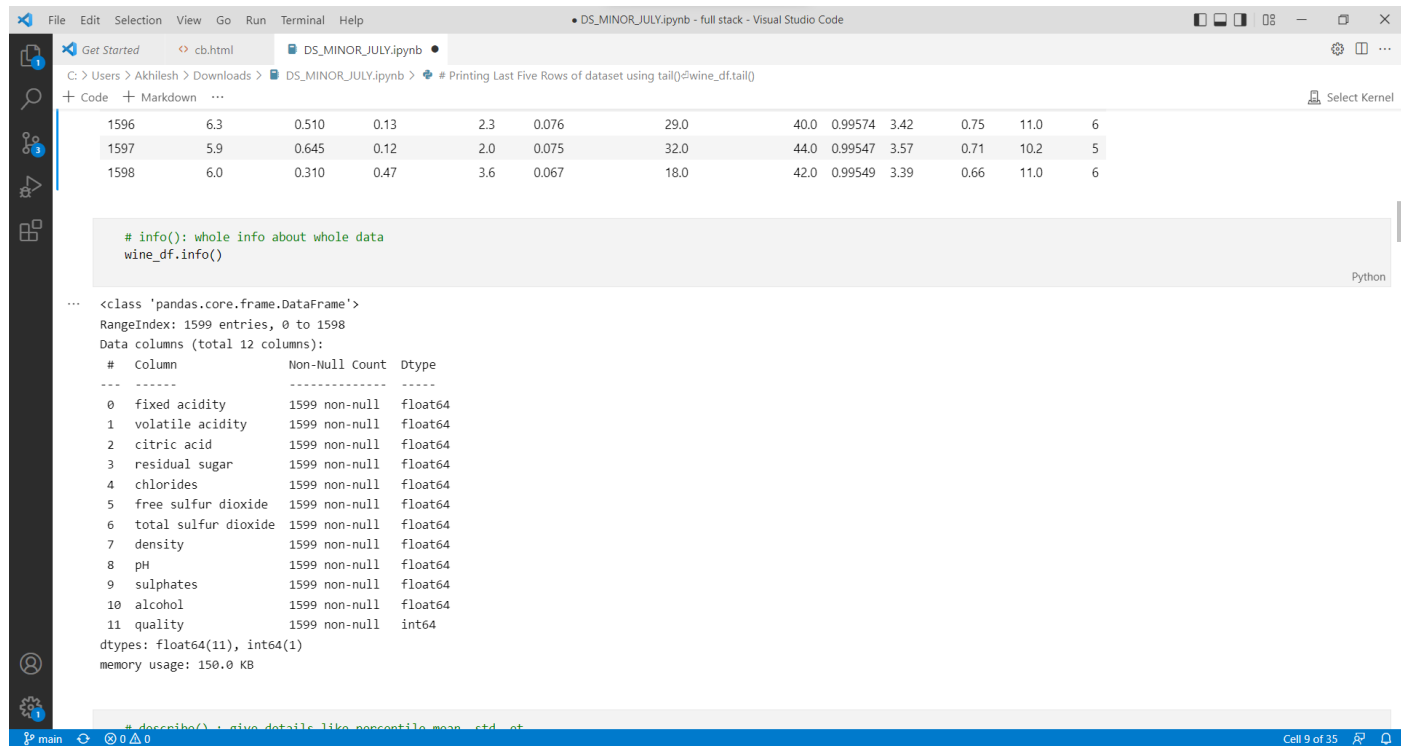
**Cell 3: info(): whole info about whole data**

```
wine_df.info()
```

The status bar at the bottom indicates "main" and "Cell 9 of 35".



### 3. Information of Dataset



The screenshot shows a Jupyter Notebook titled "DS\_MINOR\_JULY.ipynb" in Visual Studio Code. The notebook is open to a cell containing the following code and output:

```
# info(): whole info about whole data
wine_df.info()
```

The output of the `wine_df.info()` function is displayed below the code cell:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   fixed acidity          1599 non-null   float64
1   volatile acidity       1599 non-null   float64
2   citric acid            1599 non-null   float64
3   residual sugar         1599 non-null   float64
4   chlorides              1599 non-null   float64
5   free sulfur dioxide    1599 non-null   float64
6   total sulfur dioxide   1599 non-null   float64
7   density               1599 non-null   float64
8   pH                    1599 non-null   float64
9   sulphates             1599 non-null   float64
10  alcohol               1599 non-null   float64
11  quality               1599 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

Below the output, the code for the next cell is visible:

```
# describe(): give details like percentile, mean, std, etc.
```

The status bar at the bottom indicates "Cell 9 of 35".

## 4. Checking for Null Value

The screenshot shows a Jupyter Notebook interface in Visual Studio Code. The notebook is titled "DS\_MINOR\_JULY.ipynb" and is running on a "Python" kernel. The current cell displays a summary statistics table for a dataset, followed by a code cell that checks for null values.

Summary Statistics Table:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.422983	5.636023
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.065668	0.807569
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.000000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.500000	5.000000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.200000	6.000000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.100000	6.000000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.000000

Code cell content:

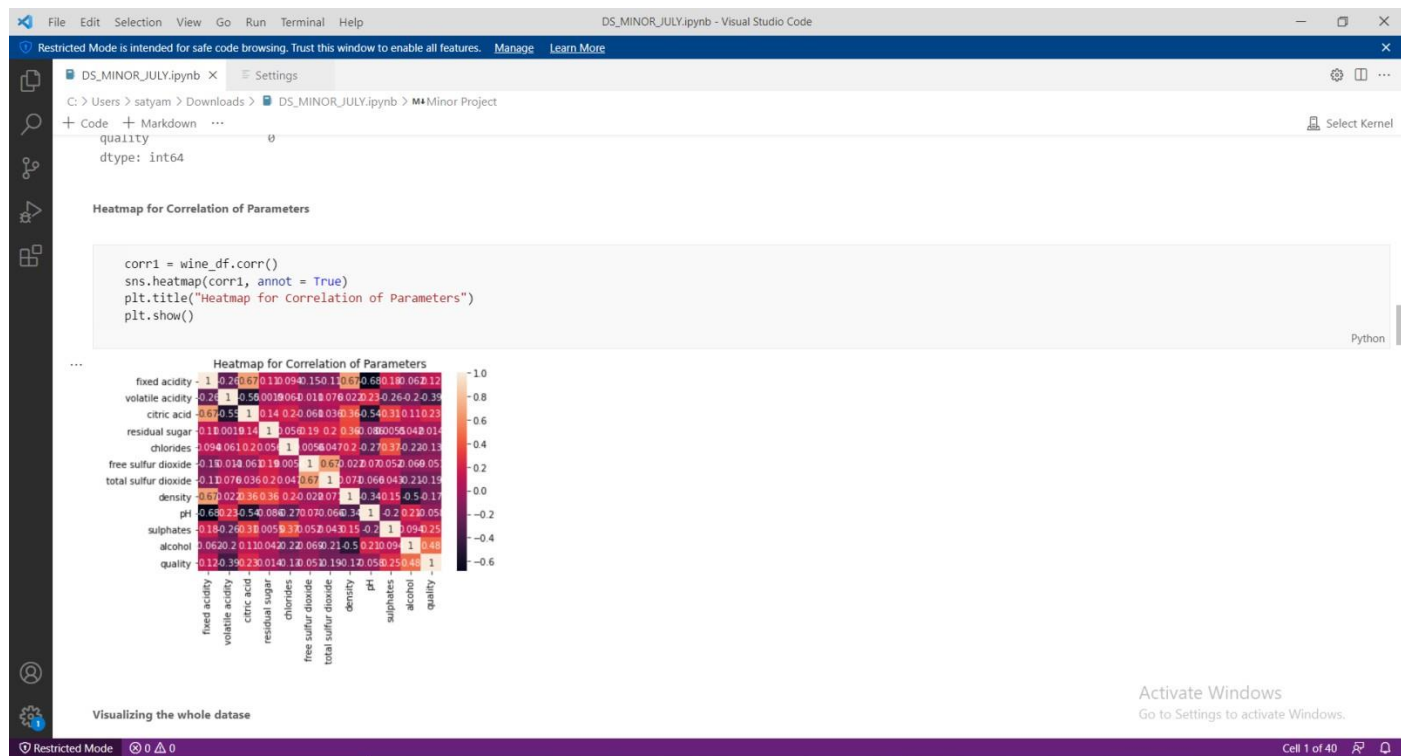
```
# isnull().sum() : Checking for any null/missing value
wine_df.isnull().sum()
```

Output of the code cell:

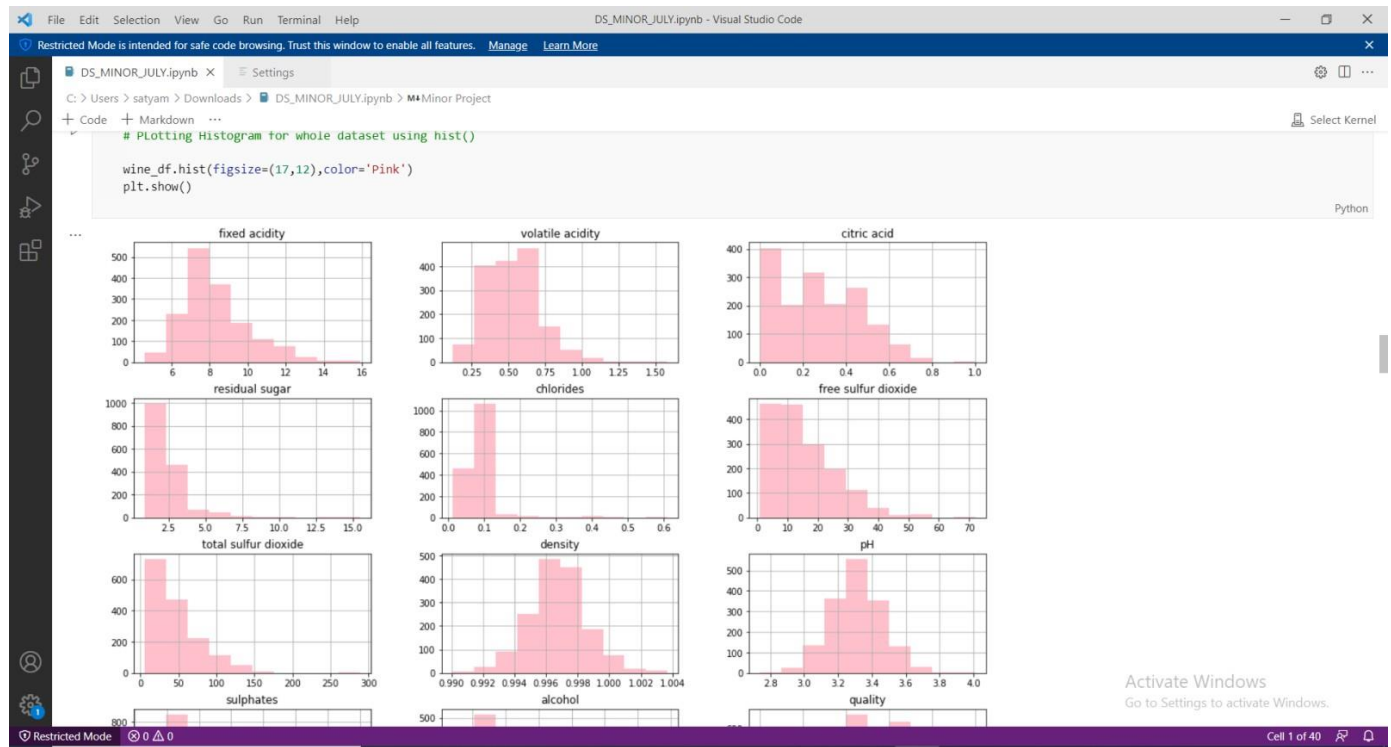
```
fixed acidity      0
volatile acidity   0
citric acid        0
residual sugar     0
chlorides          0
free sulfur dioxide 0
total sulfur dioxide 0
density            0
pH                 0
sulphates          0
alcohol            0
quality            0
dtype: int64
```

The status bar at the bottom indicates "main" and "Cell 9 of 35".

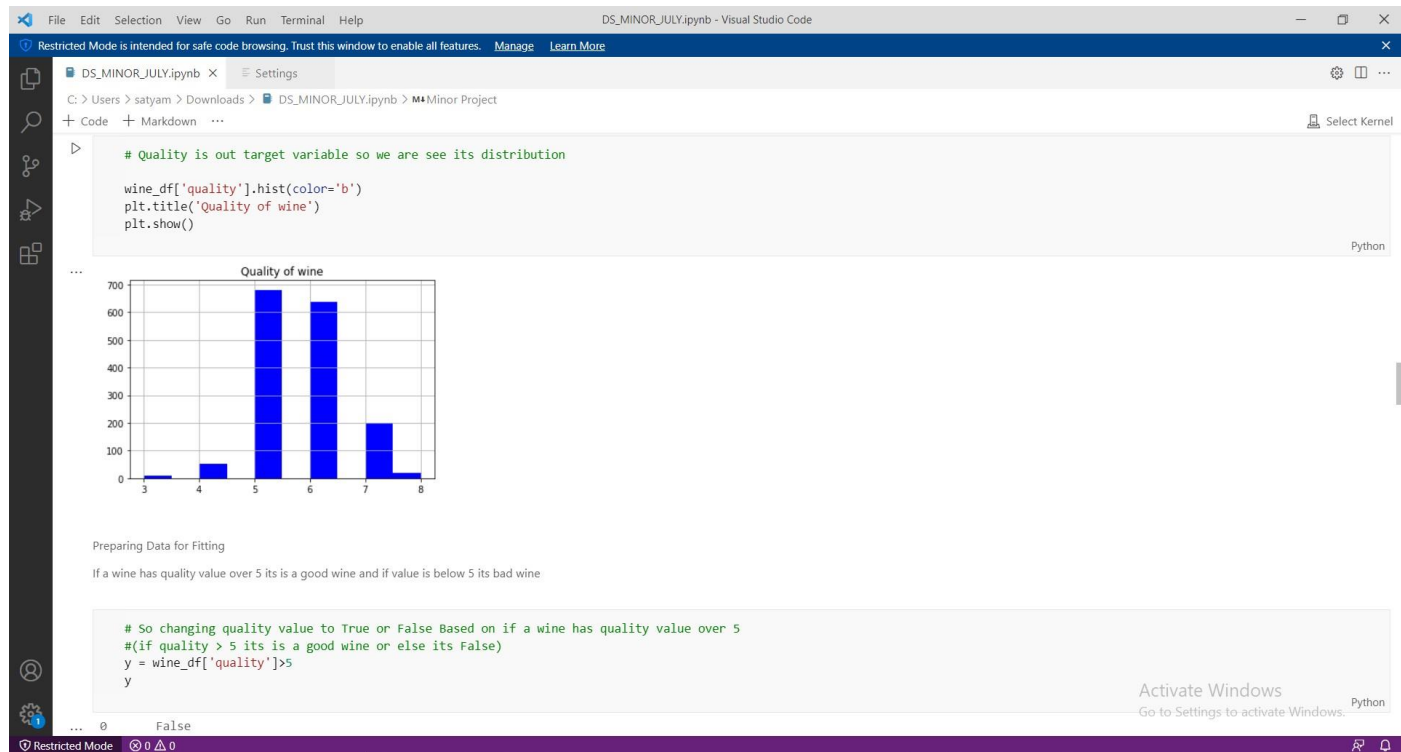
## 5. HeatMap for Correlation and Parameters



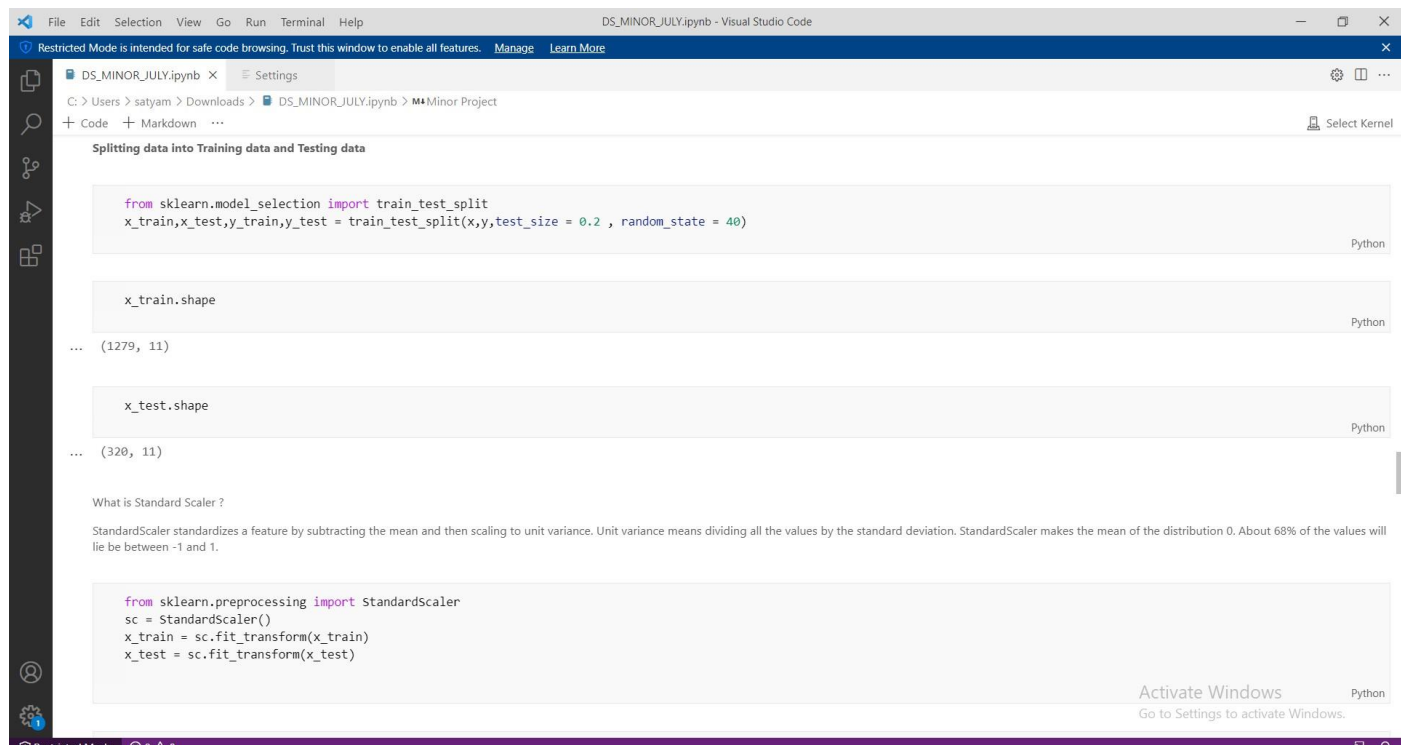
## 6. Visualizing the Whole Database



## 7. Quality Of Wine



## 8. Splitting Data into Training Data and Testing Data.



The screenshot shows a Jupyter Notebook titled "DS\_MINOR\_JULY.ipynb" in Visual Studio Code. The notebook is in "Restricted Mode" and contains the following content:

**Splitting data into Training data and Testing data**

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 40)
```

**x\_train.shape**

```
... (1279, 11)
```

**x\_test.shape**

```
... (320, 11)
```

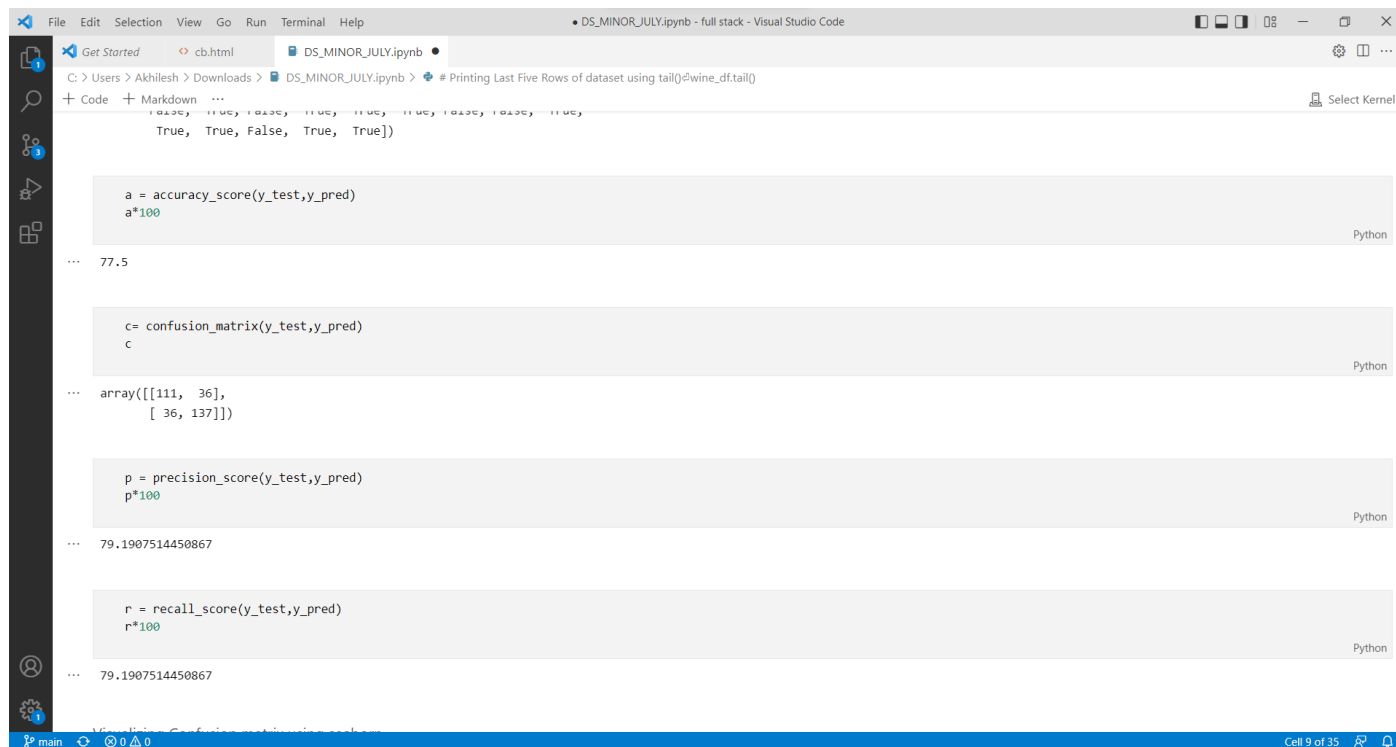
**What is Standard Scaler ?**

StandardScaler standardizes a feature by subtracting the mean and then scaling to unit variance. Unit variance means dividing all the values by the standard deviation. StandardScaler makes the mean of the distribution 0. About 68% of the values will lie between -1 and 1.

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

Activate Windows  
Go to Settings to activate Windows.

## 9. Conclusion ( Accuracy and precision)



The screenshot shows a Jupyter Notebook titled "DS\_MINOR\_JULY.ipynb" in Visual Studio Code. The notebook is open to a cell where the last five rows of a dataset are printed using `tail()`. The output shows a list of boolean values: `True, True, False, True, True`. Below this, four cells are shown, each calculating a different metric and multiplying the result by 100:

```
a = accuracy_score(y_test,y_pred)
a*100
```

Output: 77.5

```
c= confusion_matrix(y_test,y_pred)
c
```

Output: `array([[111, 36], [ 36, 137]])`

```
p = precision_score(y_test,y_pred)
p*100
```

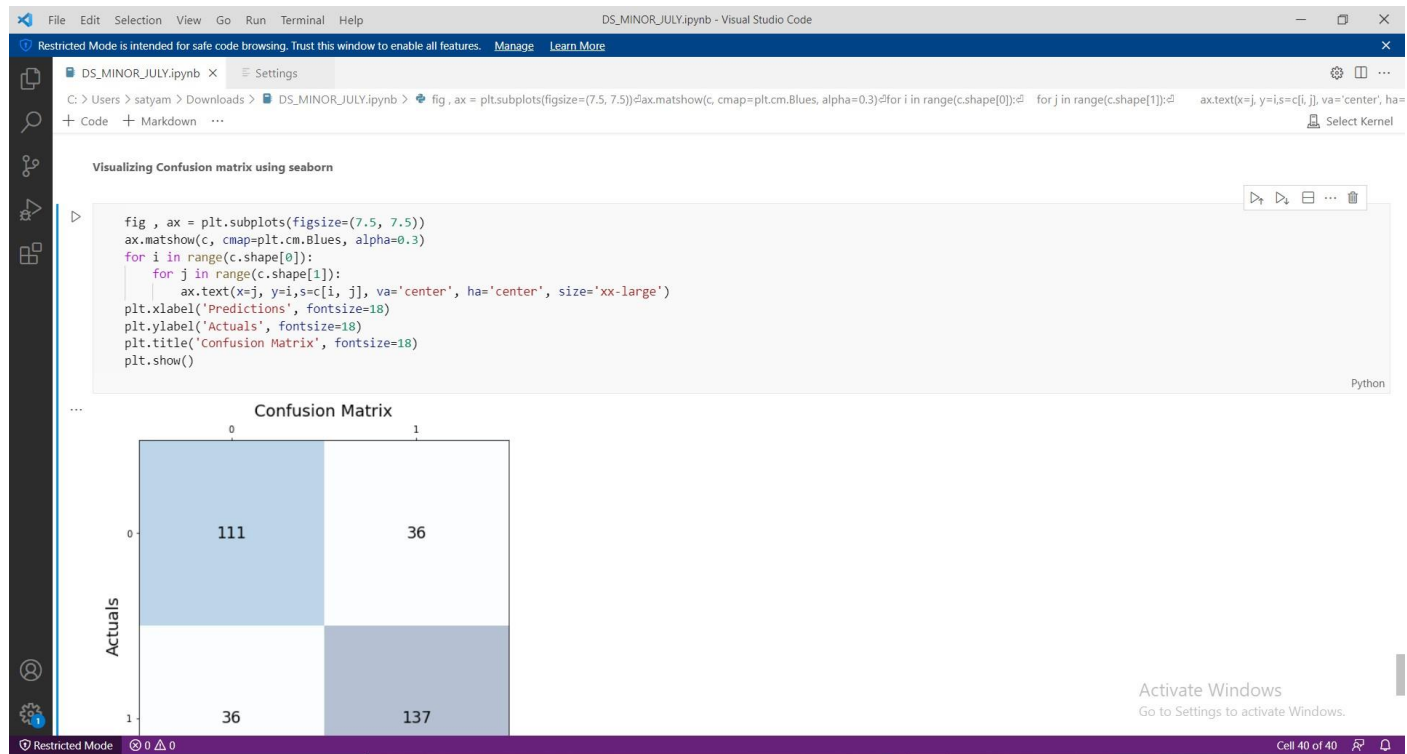
Output: 79.1907514450867

```
r = recall_score(y_test,y_pred)
r*100
```

Output: 79.1907514450867

The bottom status bar indicates the notebook is on the "main" branch, with 0 commits and 0 changes. The current cell is 9 of 35.

## 10. Visualizing Confusion Matrix Using Seaborn





## CONCLUSION:

Overall our logistic regression model performs quite well with:

- **Accuracy:** 77.5
- **Precision:** 79.1907514450867
- **Recall:** 79.1907514450867

## Bibliography:

[1] Data science [https://en.wikipedia.org/wiki/Data\\_science](https://en.wikipedia.org/wiki/Data_science)

[2] A book on data science by Dr. Ossama Embarak,  
[https://www.academia.edu/37886932/Data\\_Analysis\\_and\\_Visualization\\_Using\\_Python\\_-\\_Dr.\\_Ossama\\_Embarak.pdf](https://www.academia.edu/37886932/Data_Analysis_and_Visualization_Using_Python_-_Dr._Ossama_Embarak.pdf)

Python website <https://www.python.org/doc/essays/blurb/>

Matplotlib <https://en.wikipedia.org/wiki/Matplotlib>

Numpy online <https://en.wikipedia.org/wiki/NumPy>

<https://www.w3schools.com>

<https://stackoverflow.com>