VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**“JnanaSangama”, Belgaum -590014, Karnataka.**



# LAB REPORTON

**MACHINE LEARNING**

***Submitted by***

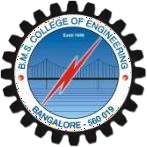
# Imran Wadrali(1BM21CS077)

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



**B. M. S. College of Engineering,**

**Bull Temple Road, Bangalore 560019(March 2024 to June 2024)**



**B. M. S. College of Engineering,**

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

## Department of Computer Science and Engineering

**CERTIFICATE**

This is to certify that the Lab work entitled “**MACHINE LEARNING**” is carried out by **Imran Wadrali (1BM21CS077)** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visveswaraya Technological University, Belgaum during the year 2023-2024. The lab report has been approved as it satisfies the academic requirements in respect of **Machine Learning Lab - (22CS3PCMAL)** work prescribed for thesaid degree.

|  |  |
| --- | --- |
| **Dr. K. Panimozhi** | **Dr. Jyothi S Nayak** |
| Assistant Professor | Prof.& Head, Dept. of CSE |
| BMSCE, Bengaluru | BMSCE, Bengaluru |

**Index**

|  |  |  |
| --- | --- | --- |
| **Sl.**  **No.** | **Experiment Title** | **Page No.** |
| 1 | Write a python program to import and export data using Pandas library functions | 4 |
| 2 | Demonstrate various data pre-processing techniques for a given dataset | 7 |
| 3 | Use an appropriate data set for building the decision tree (ID3) and apply this knowledge to classify a new sample. | 10 |
| 4 | Implement Linear and Multi-Linear Regression algorithm using appropriate dataset | 14 |
| 5 | Build KNN Classification model for a given dataset. | 17 |
| 6 | Build Logistic Regression Model for a given dataset | 19 |
| 7 | Build Support vector machine model for a given dataset. | 22 |
| 8 | Build k-Means algorithm to cluster a set of data stored in a .CSV file. | 25 |
| 9 | Implement Dimensionality reduction using Principle Component Analysis (PCA) method. | 27 |
| 10 | Build Artificial Neural Network model with back propagation on a given dataset | 29 |
| 11 | 1. Implement Random forest ensemble method on a given dataset. 2. Implement Boosting ensemble method on a given dataset. | 32 |

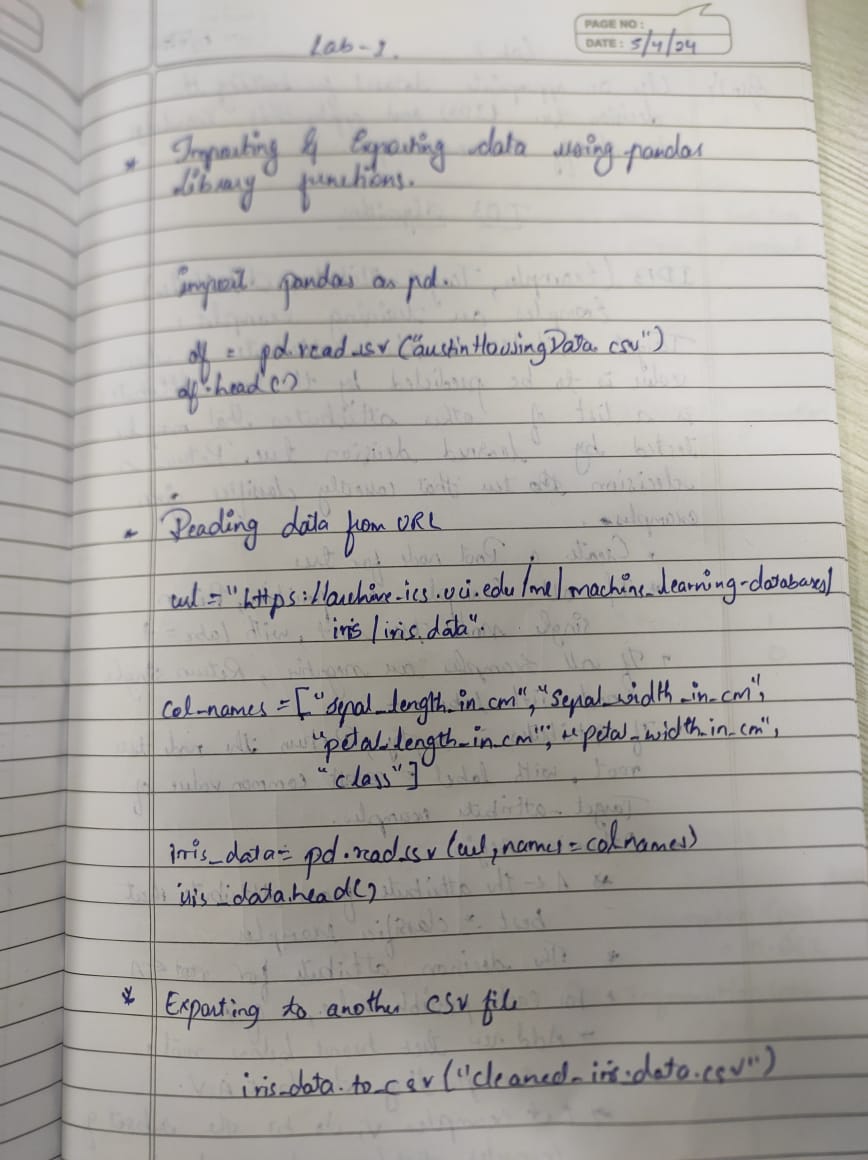
**Course outcomes:**

|  |  |
| --- | --- |
| CO1 | Apply machine learning techniques in computing systems |
| CO2 | Evaluate the model using metrics |
| CO3 | Design a model using machine learning to solve a problem |
| CO4 | Conduct experiments to solve real-world problems using appropriate machine learning techniques |

**Lab 1**

1. Write a python program to import and export data using Pandas library functions.

Algorithm (Observation book):



Code

import pandas as pd

df=pd.read\_csv("/content/austinHousingData.csv")

df.head(5)

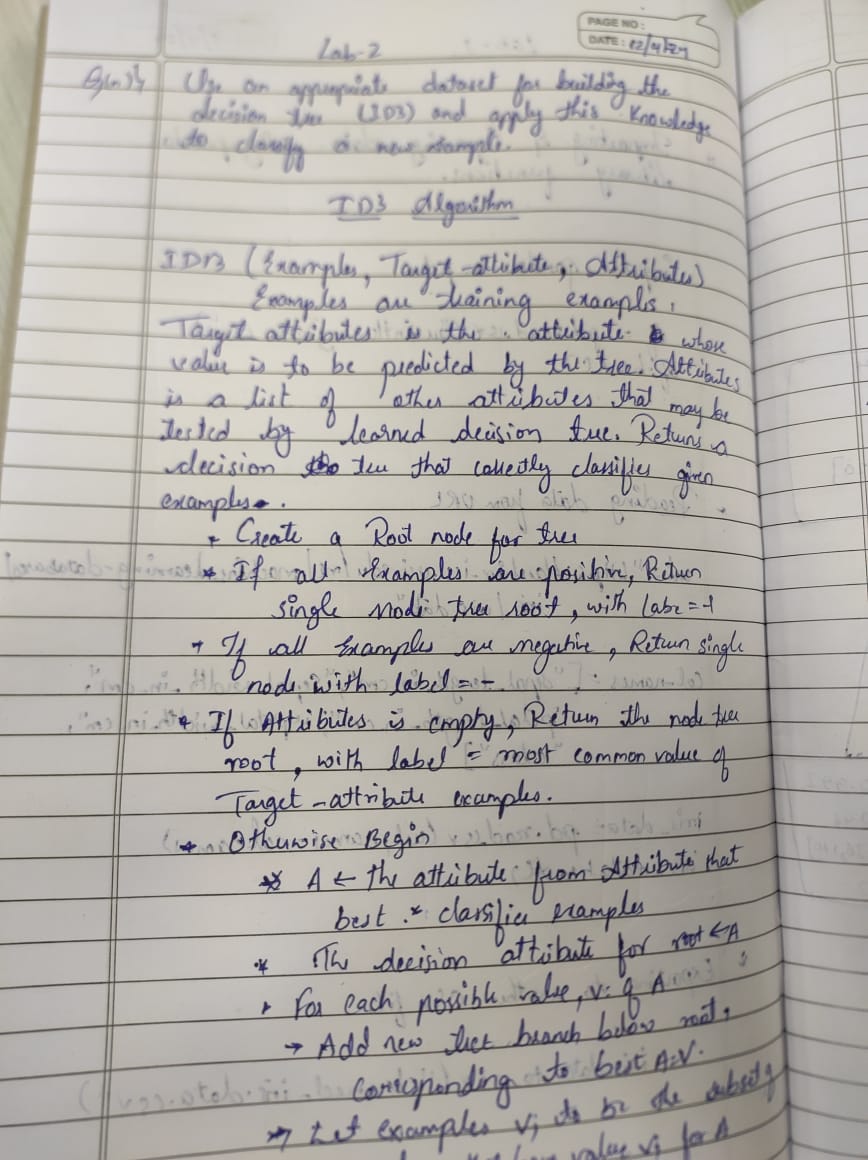
Output:

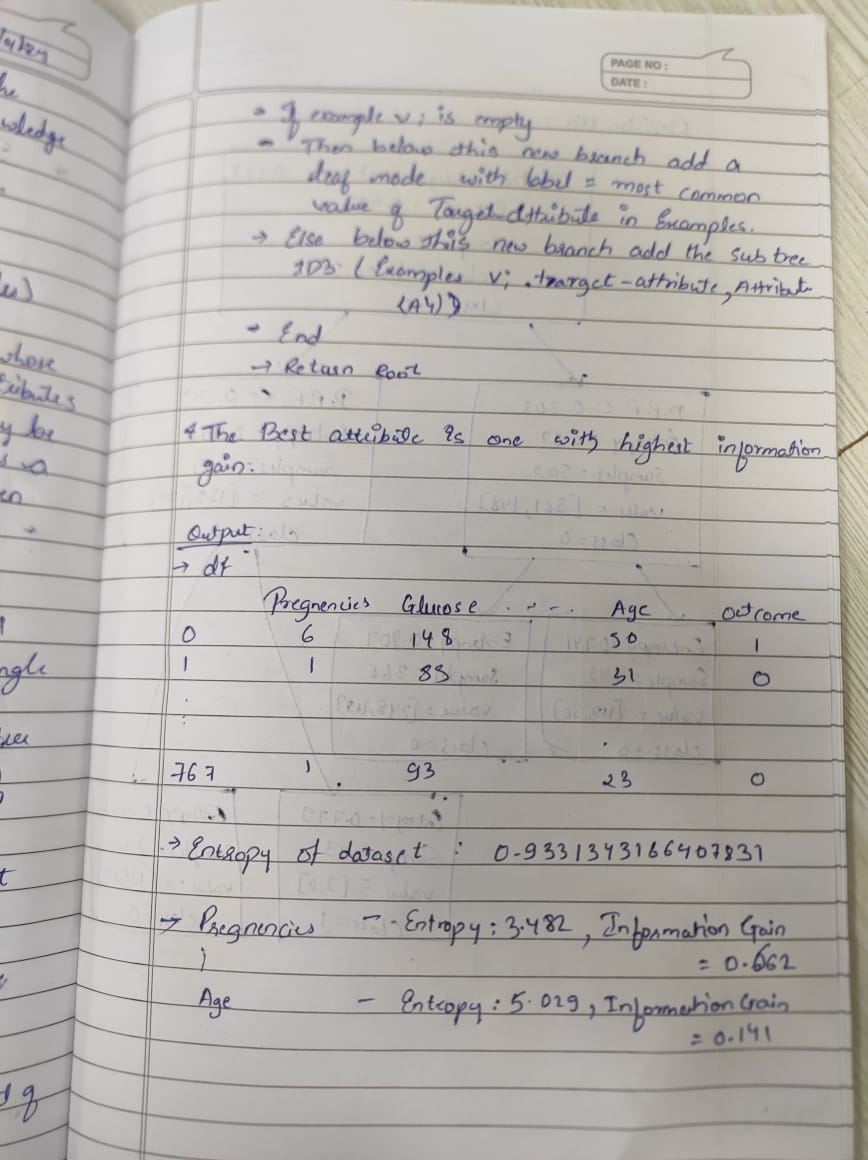


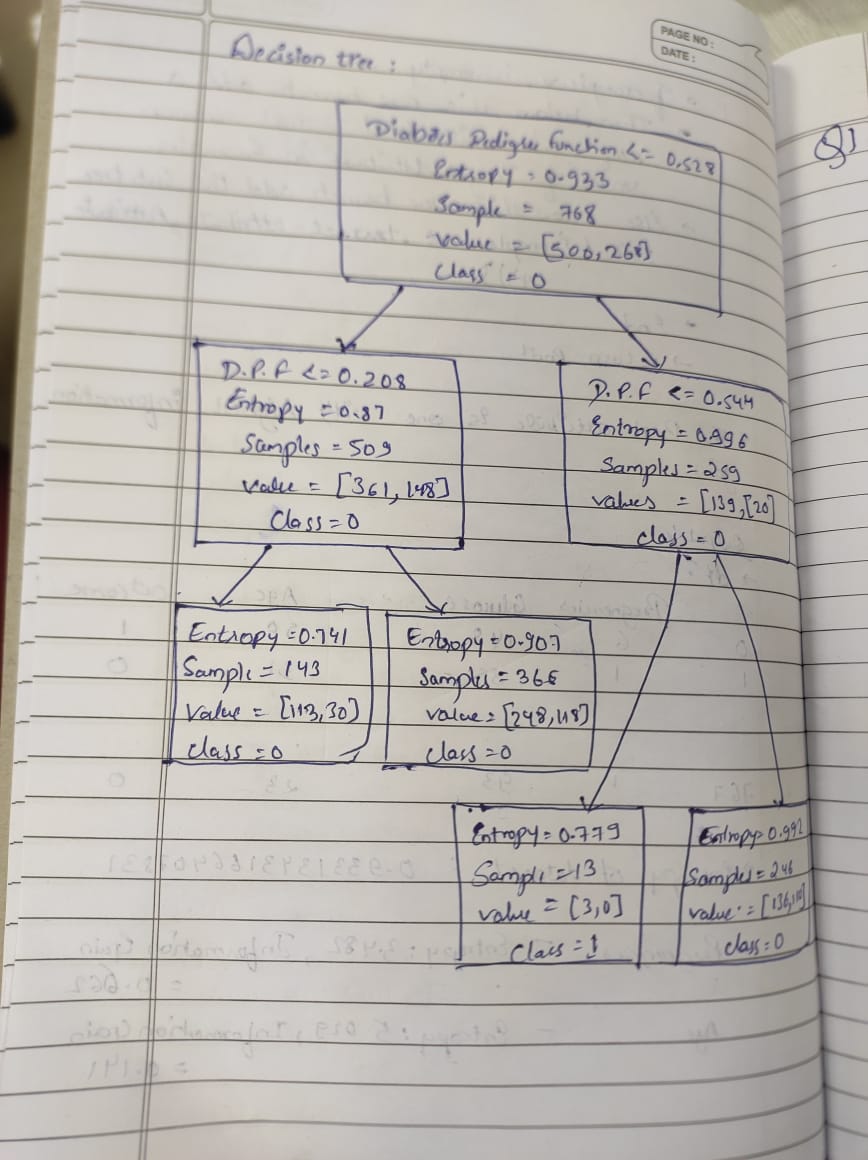
Exporting data

1. Use an appropriate dataset for building the decision tree (ID3) and apply this knowledge to classify a new sample.

Algorithm (Observation book):







Code:

import pandas as pd

from sklearn.tree import DecisionTreeClassifier, plot\_tree

import matplotlib.pyplot as plt

import math

df = pd.read\_csv('/content/diabetes.csv')

def calculate\_entropy(data, target\_column):

total\_rows = len(data)

target\_values = data[target\_column].unique()

entropy = 0

for value in target\_values:

# Calculate the proportion of instances with the current value

value\_count = len(data[data[target\_column] == value])

proportion = value\_count / total\_rows

entropy -= proportion \* math.log2(proportion)

return entropy

entropy\_outcome = calculate\_entropy(df, 'Outcome')

print(f"Entropy of the dataset: {entropy\_outcome}")

def calculate\_entropy(data, target\_column): # for each categorical variable

total\_rows = len(data)

target\_values = data[target\_column].unique()

entropy = 0

for value in target\_values:

# Calculate the proportion of instances with the current value

value\_count = len(data[data[target\_column] == value])

proportion = value\_count / total\_rows

entropy -= proportion \* math.log2(proportion) if proportion != 0 else 0

return entropy

def calculate\_information\_gain(data, feature, target\_column):

# Calculate weighted average entropy for the feature

unique\_values = data[feature].unique()

weighted\_entropy = 0

for value in unique\_values:

subset = data[data[feature] == value]

proportion = len(subset) / len(data)

weighted\_entropy += proportion \* calculate\_entropy(subset, target\_column)

# Calculate information gain

information\_gain = entropy\_outcome - weighted\_entropy

return information\_gain

for column in df.columns[:-1]:

entropy = calculate\_entropy(df, column)

information\_gain = calculate\_information\_gain(df, column, 'Outcome')

print(f"{column} - Entropy: {entropy:.3f}, Information Gain: {information\_gain:.3f}")

# Feature selection for the first step in making decision tree

selected\_feature = 'DiabetesPedigreeFunction'

# Create a decision tree

clf = DecisionTreeClassifier(criterion='entropy', max\_depth=1)

X = df[[selected\_feature]]

y = df['Outcome']

clf.fit(X, y)

plt.figure(figsize=(8, 6))

plot\_tree(clf, feature\_names=[selected\_feature], class\_names=['0', '1'], filled=True, rounded=True)

plt.show()

def id3(data, target\_column, features):

if len(data[target\_column].unique()) == 1:

return data[target\_column].iloc[0]

if len(features) == 0:

return data[target\_column].mode().iloc[0]

best\_feature = max(features, key=lambda x: calculate\_information\_gain(data, x, target\_column))

tree = {best\_feature: {}}

features = [f for f in features if f != best\_feature]

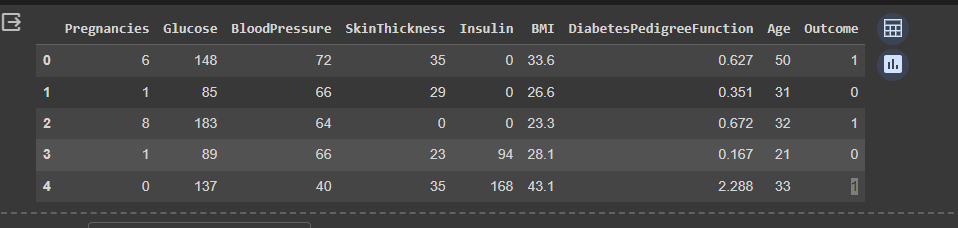
for value in data[best\_feature].unique():

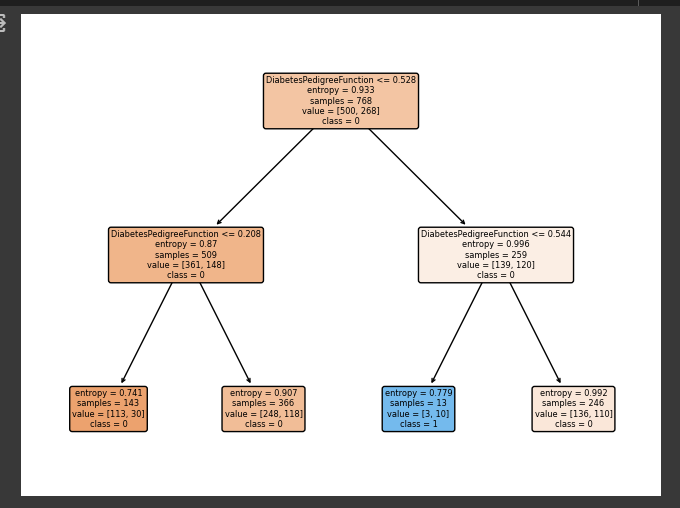
subset = data[data[best\_feature] == value]

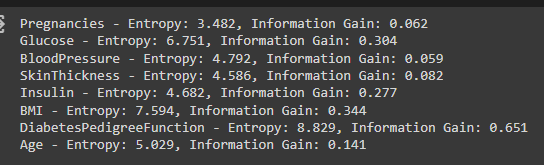
tree[best\_feature][value] = id3(subset, target\_column, features)

return tree

id3(df, 'Outcome', ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age'] )

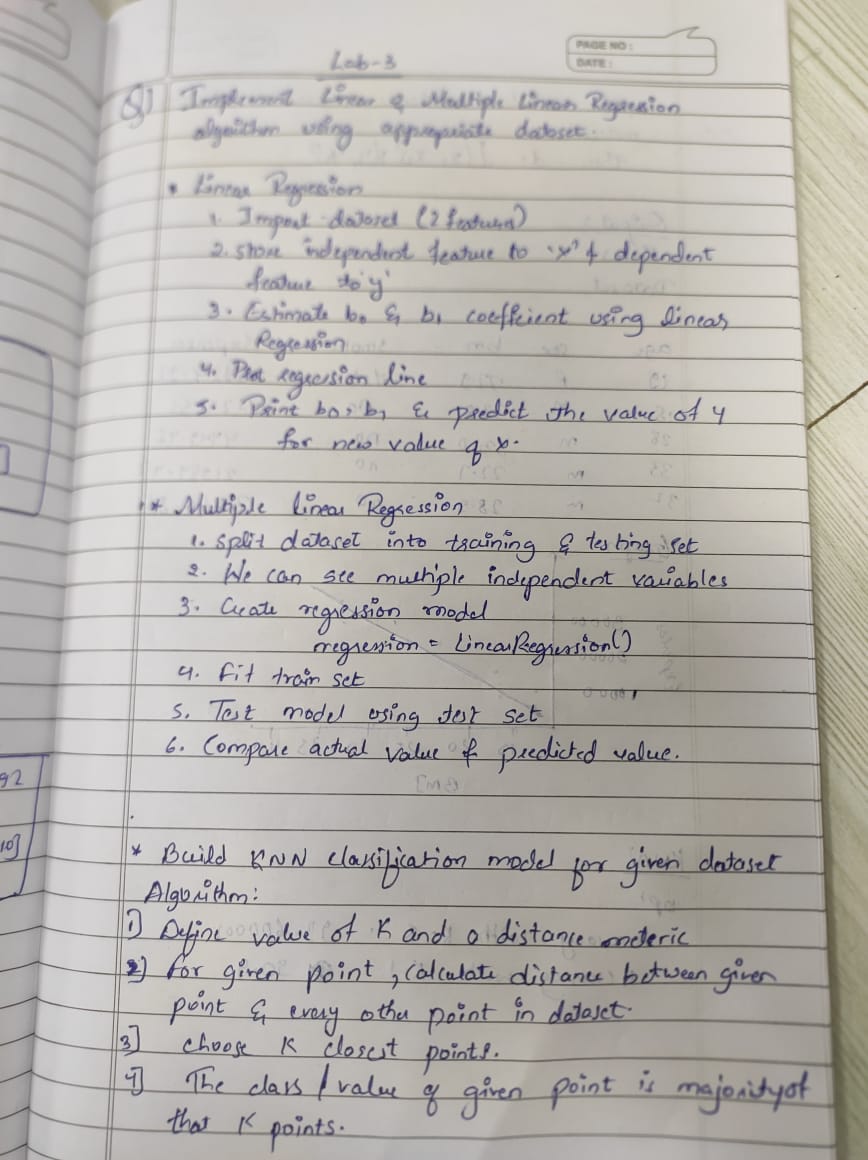




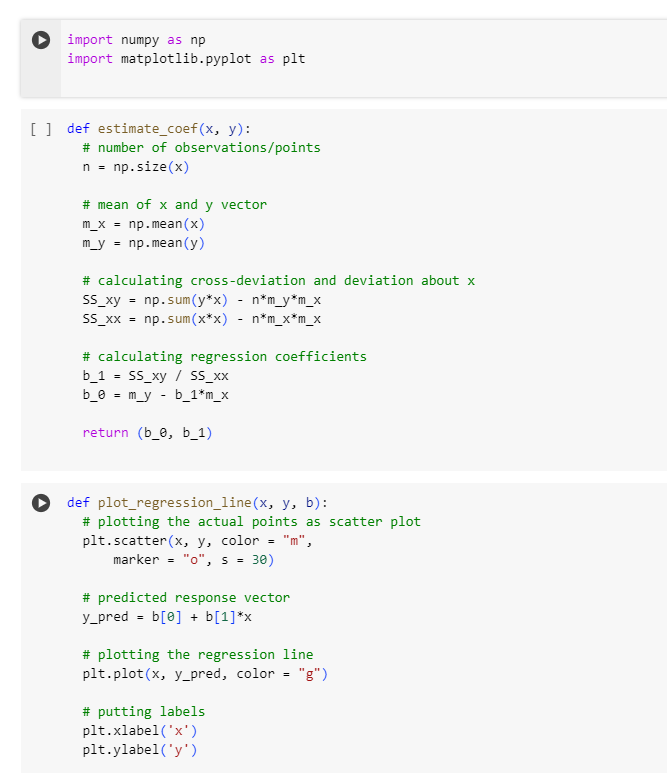


1. Implement Linear and Multi-Linear Regression algorithm using appropriate dataset

**Algorithm**

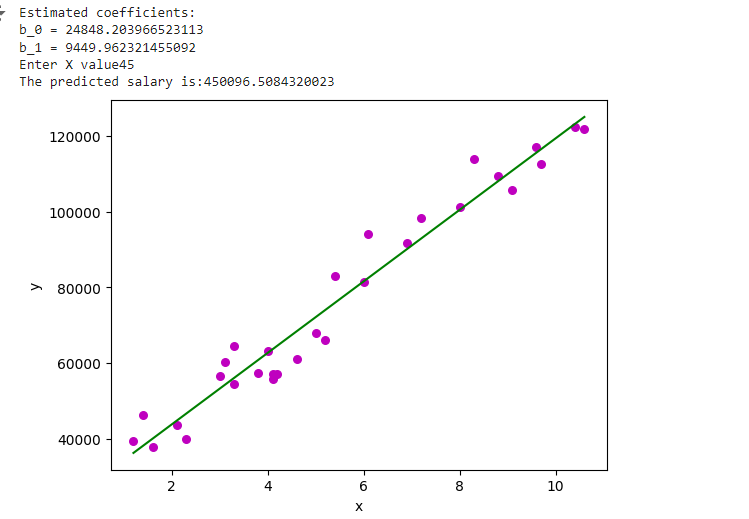


**Code:**

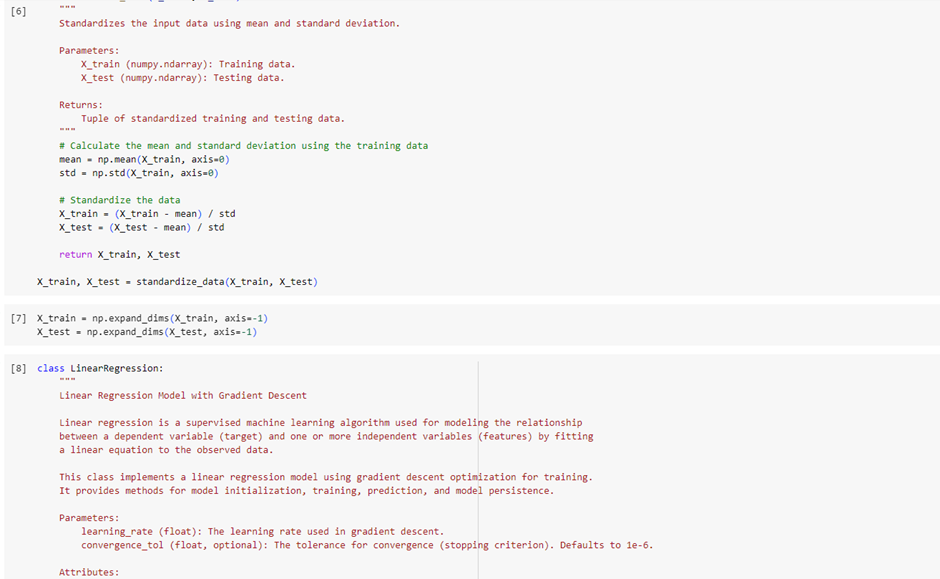


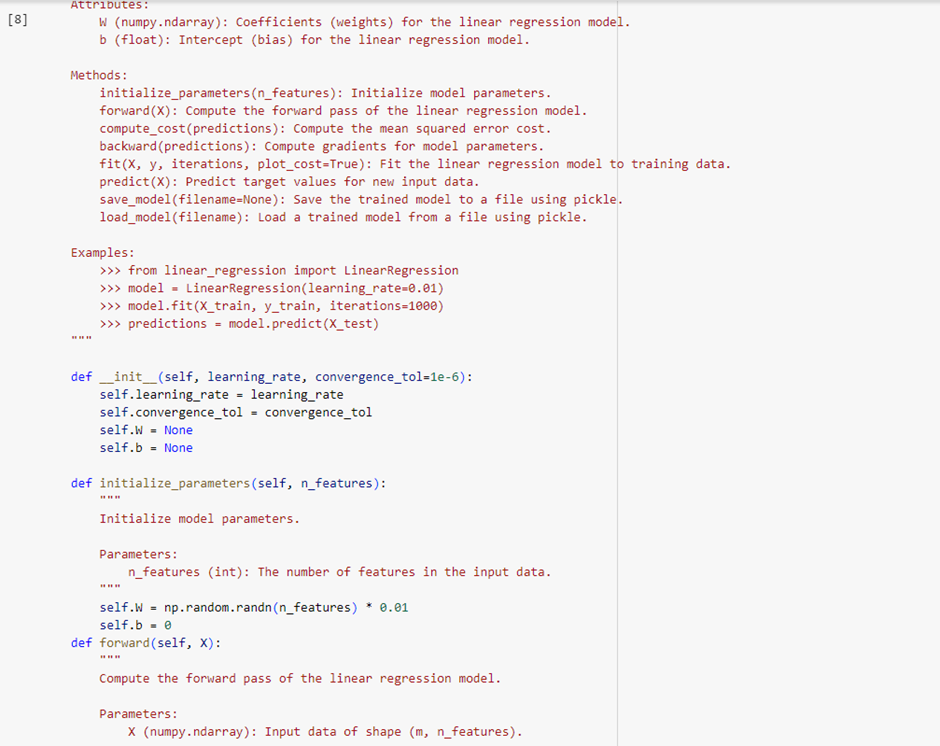


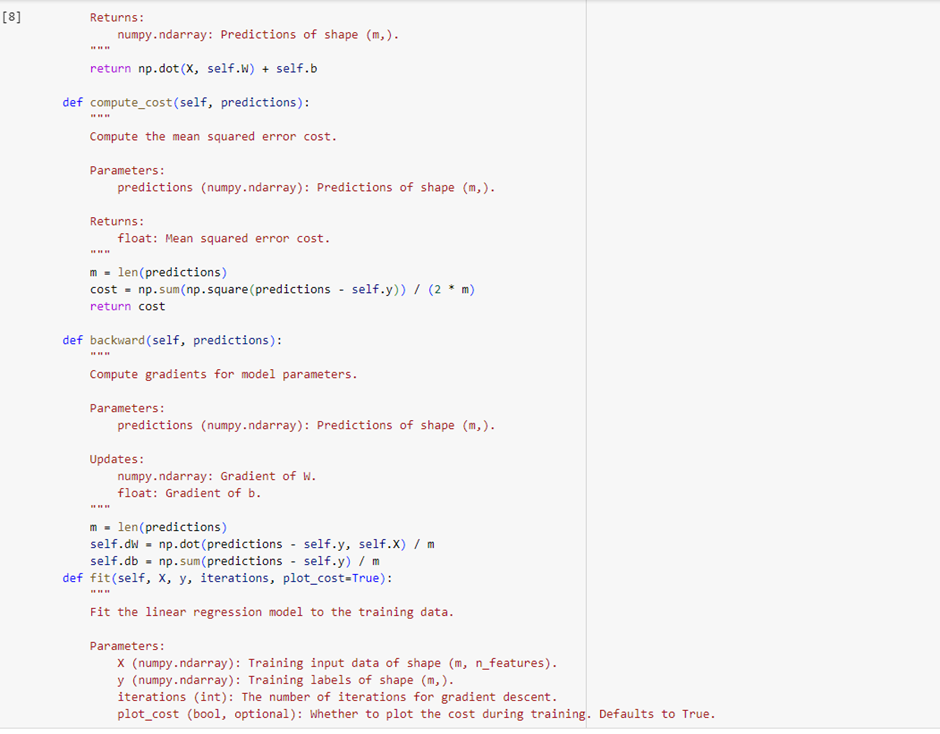
OUTPUT:

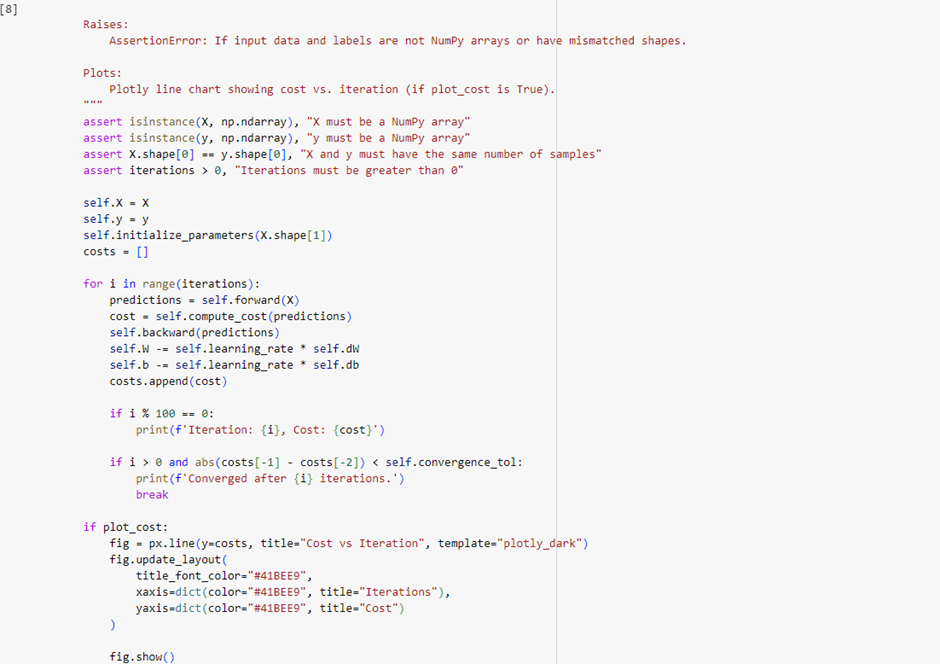


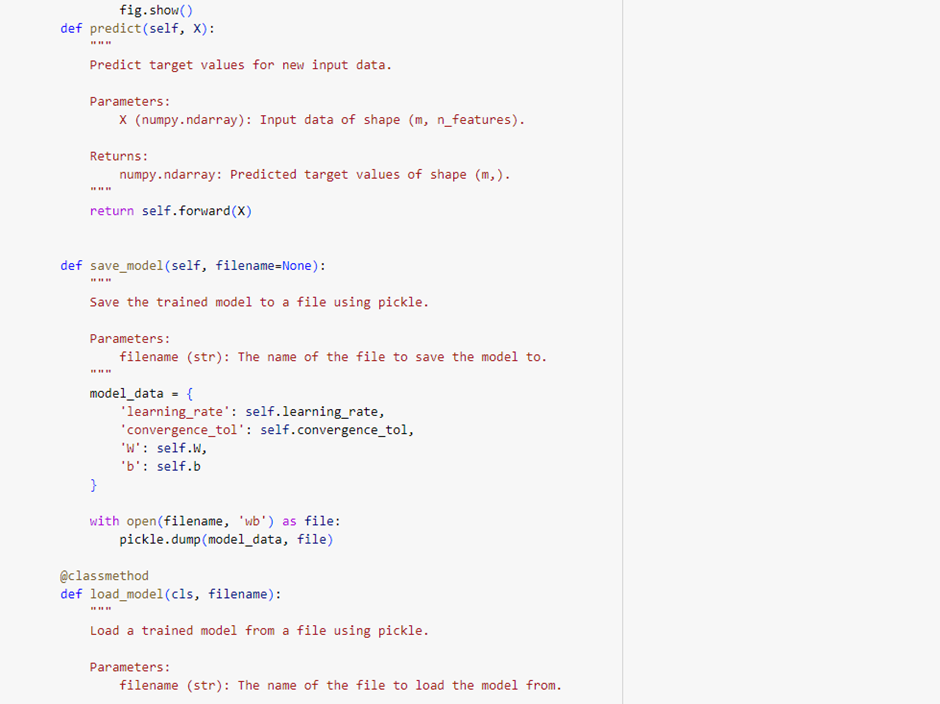
b) Implement Multi-Linear Regression algorithm using appropriate dataset

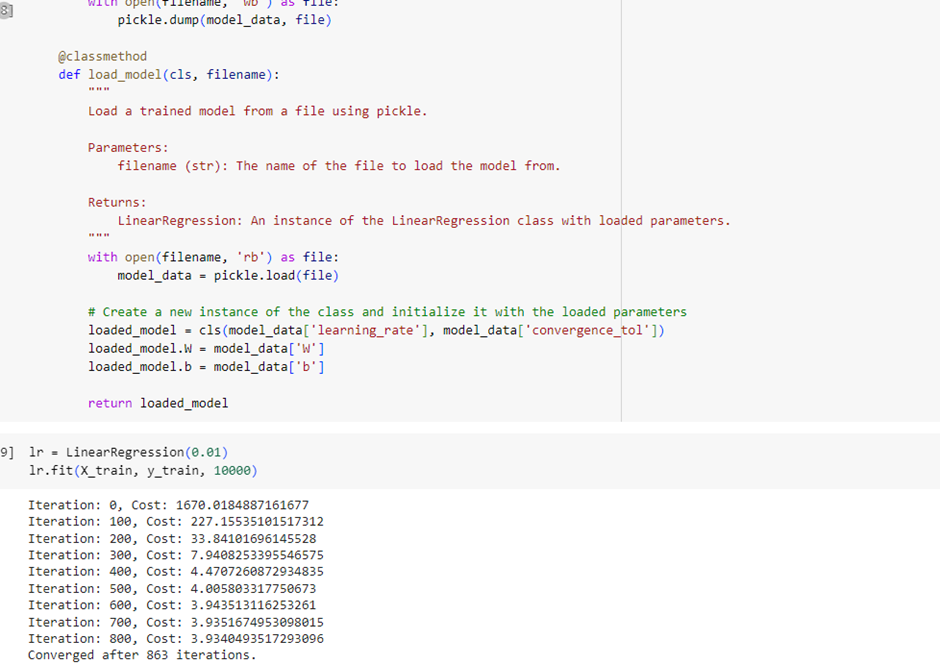






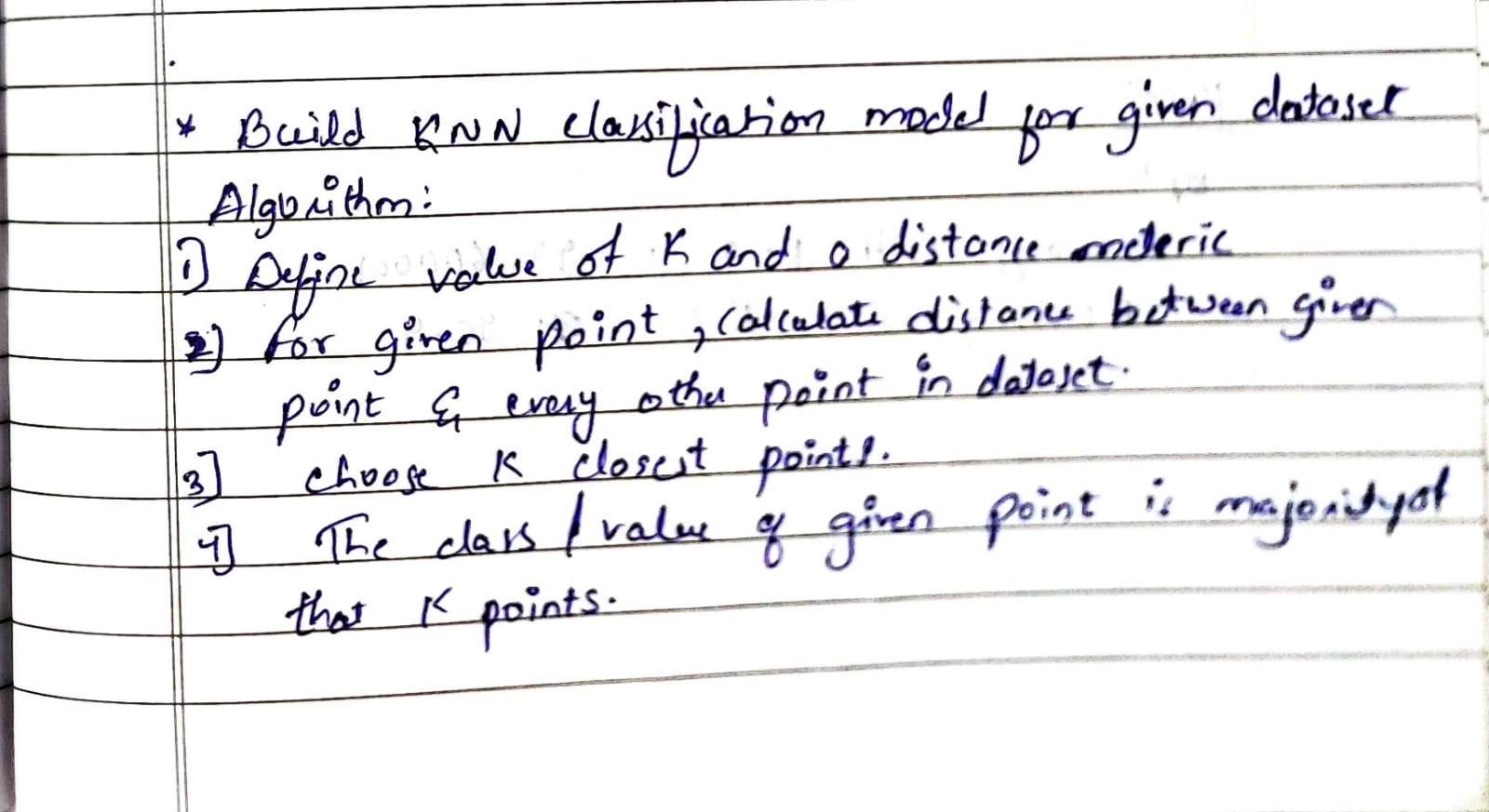




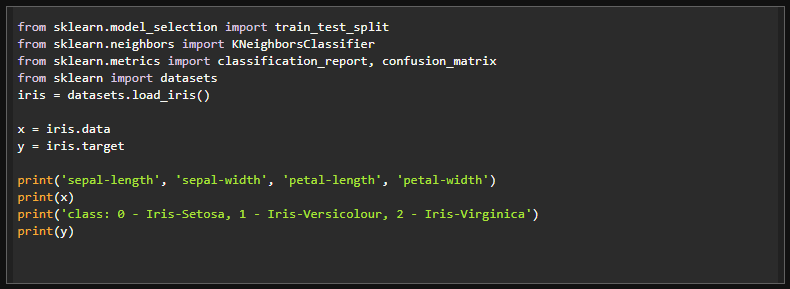


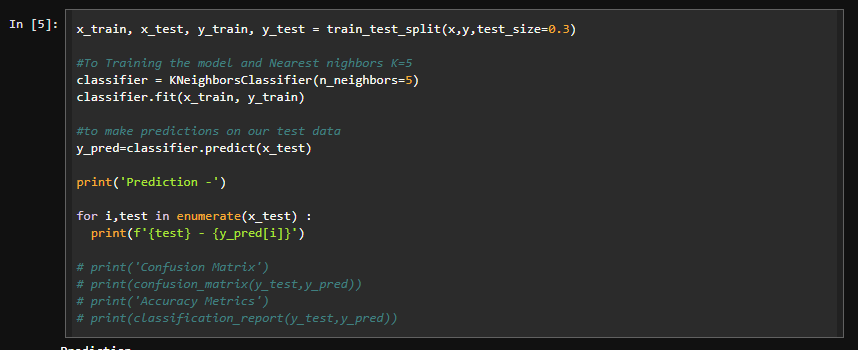
1. Build KNN Classification model for a given dataset.

Algorithm

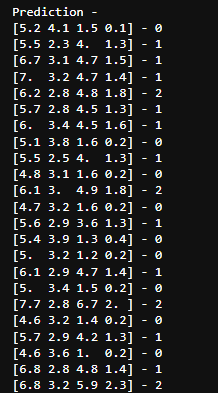


Code:



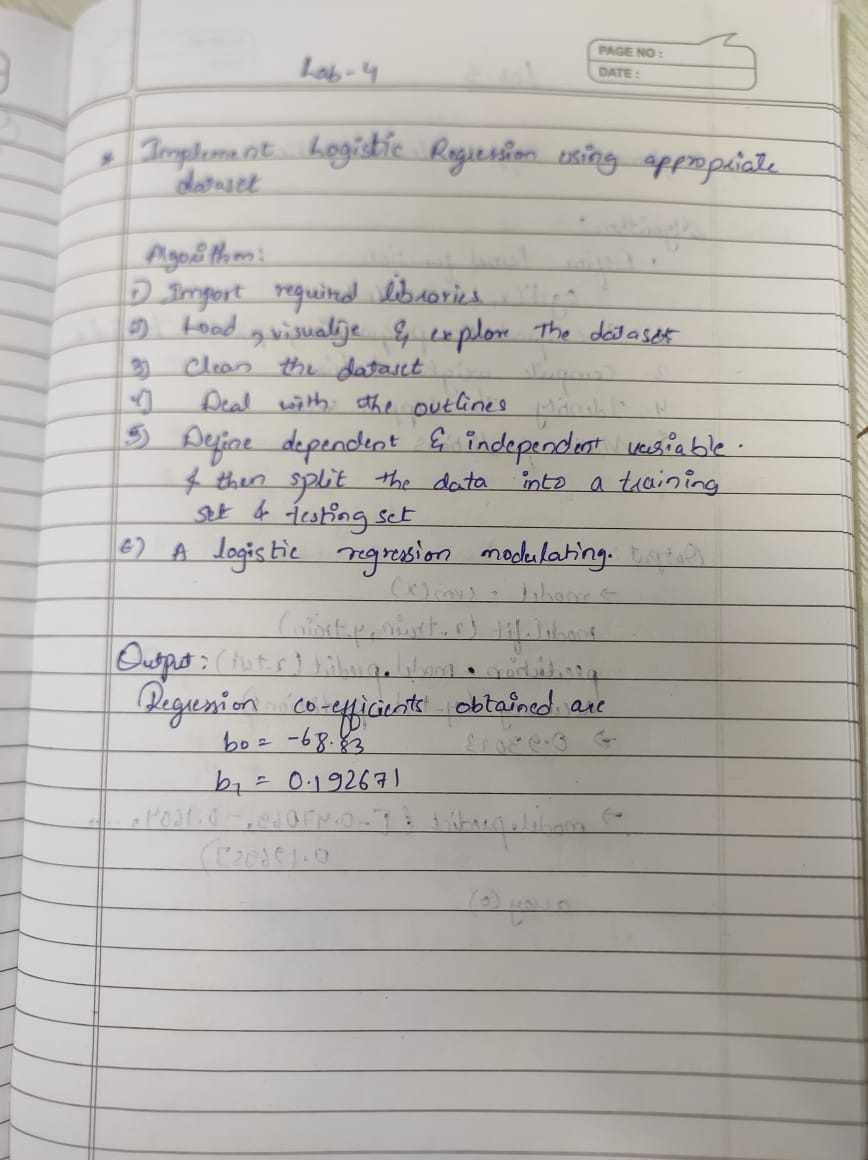


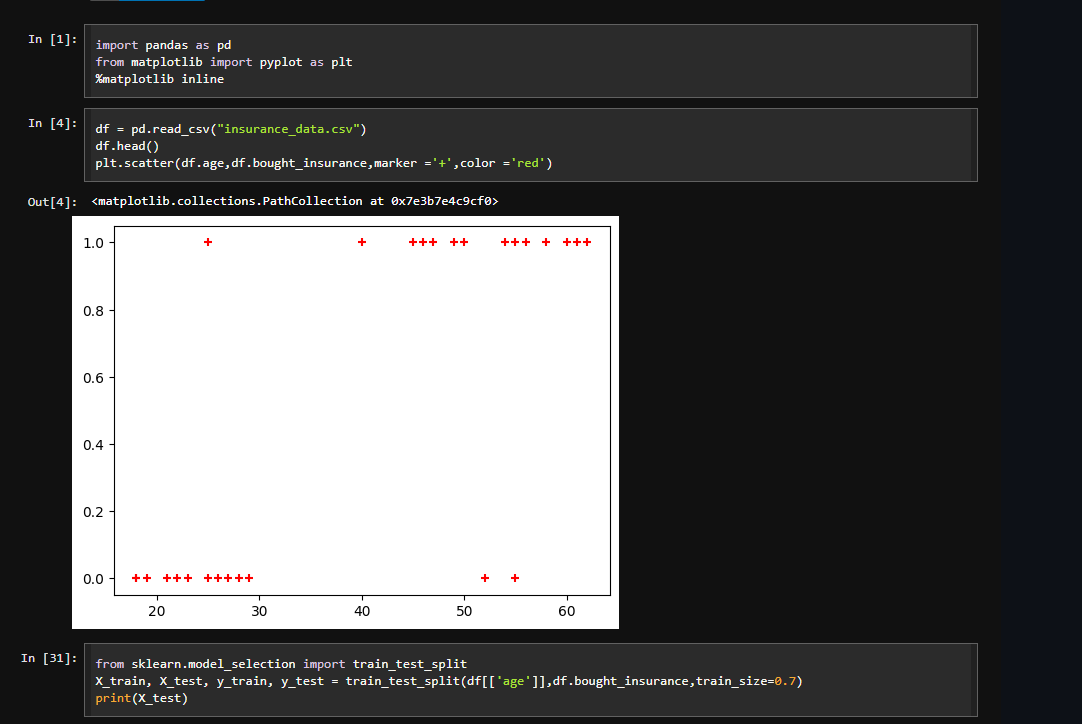
Results:

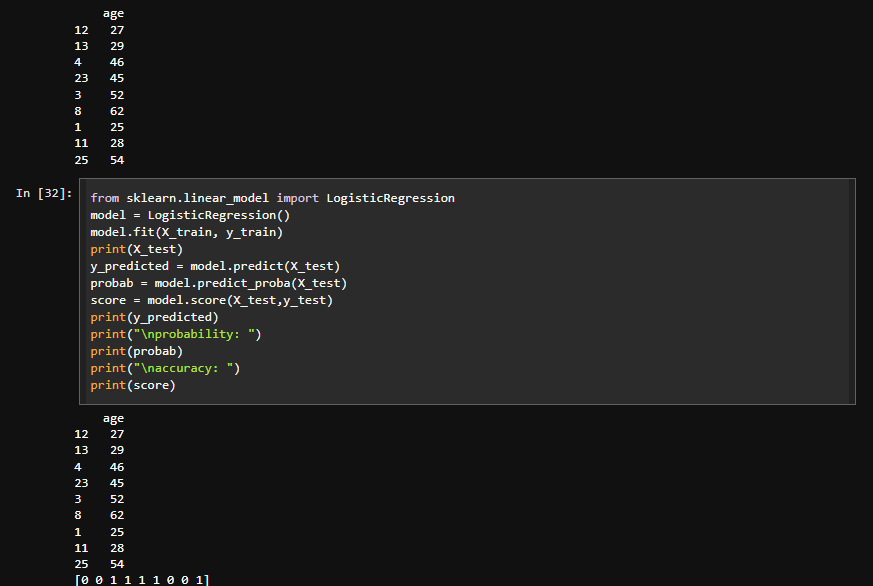


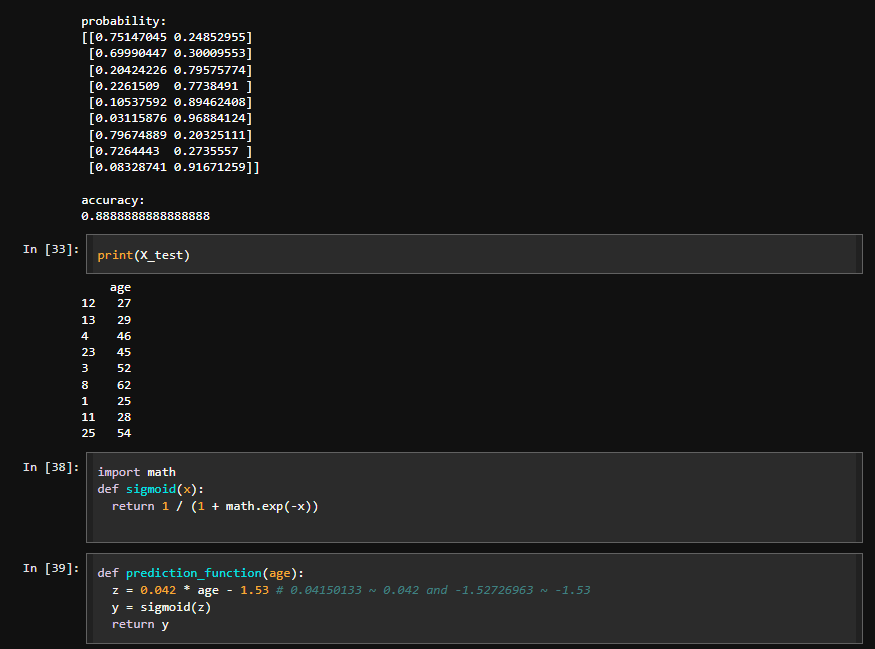
1. Build Logistic Regression Model for a given dataset

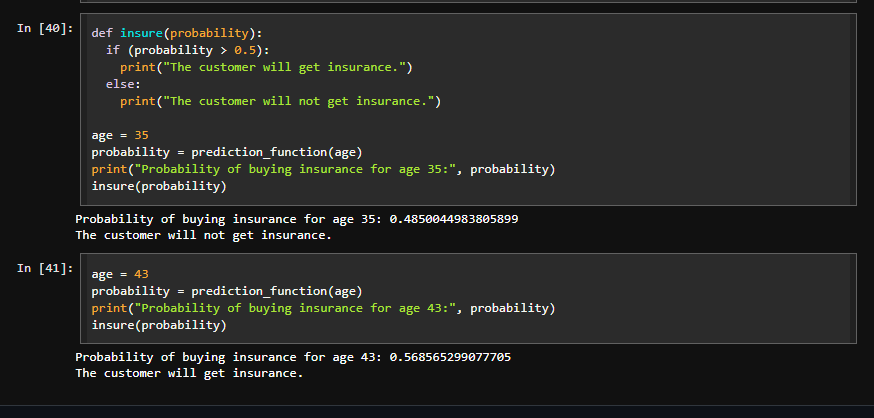
Algorithm:



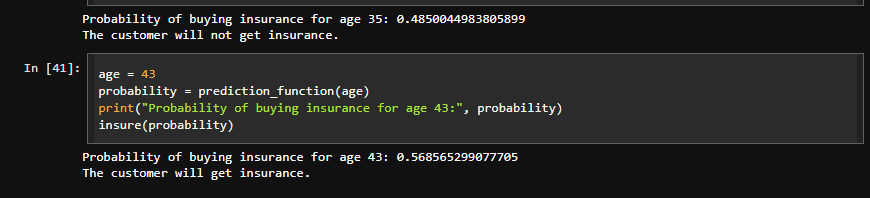
Code:





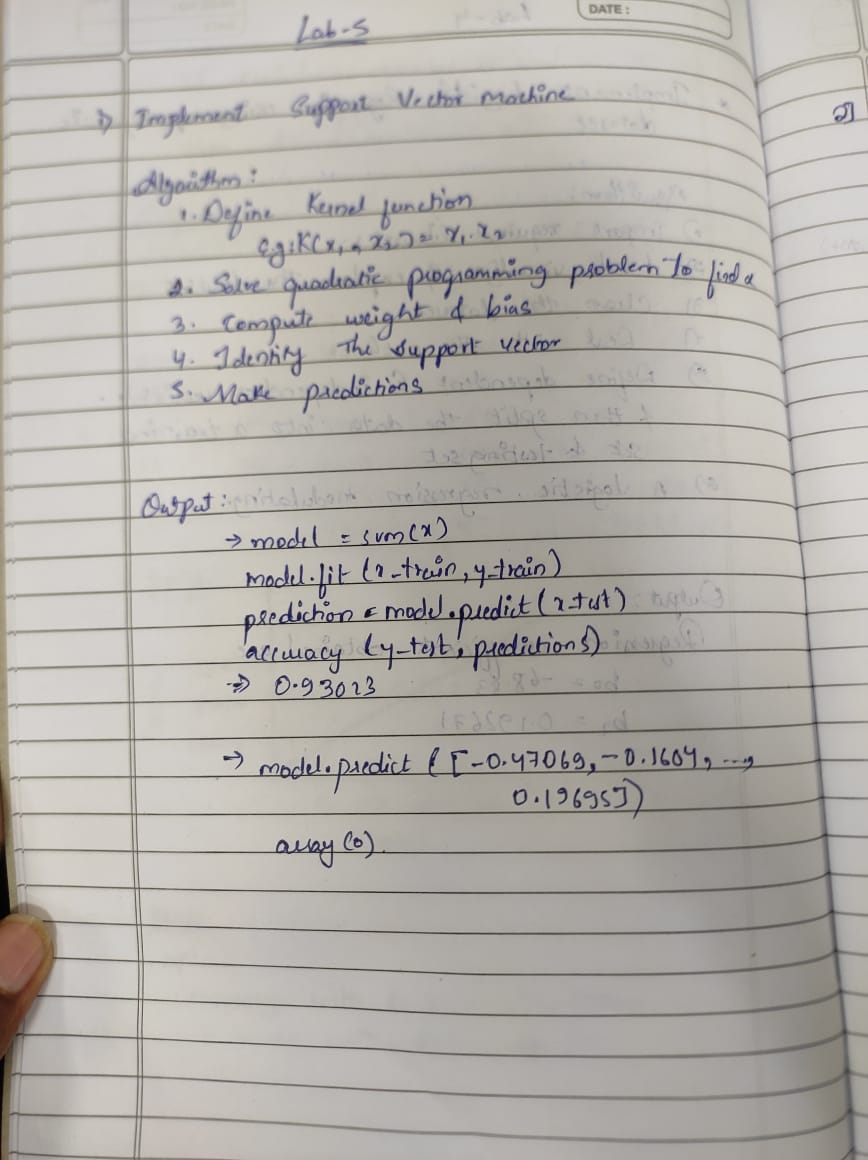


Results:

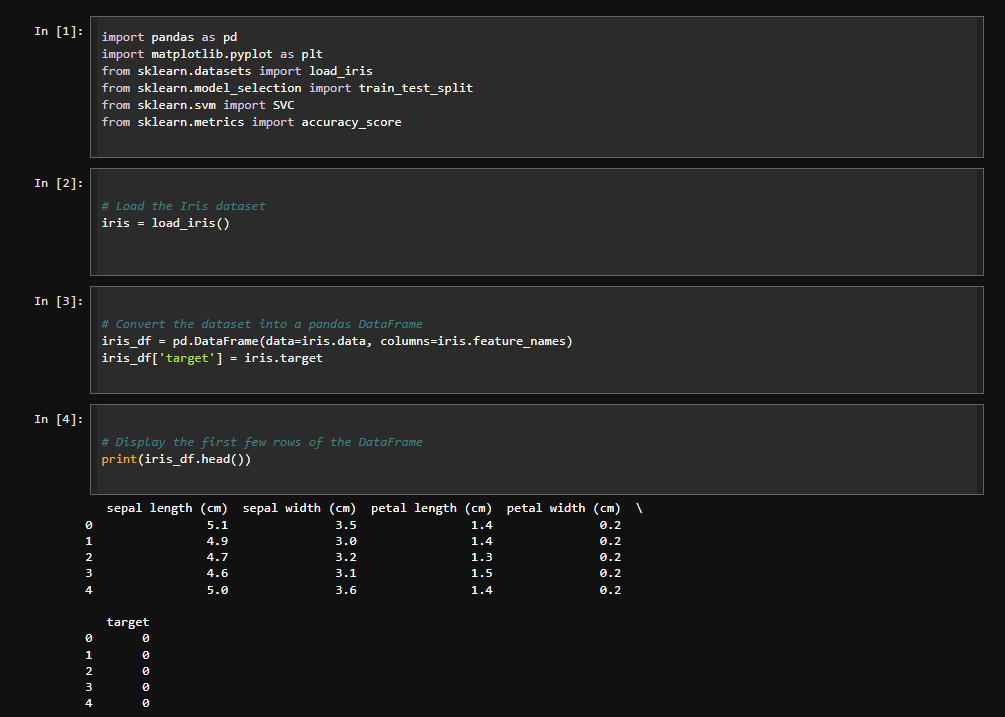


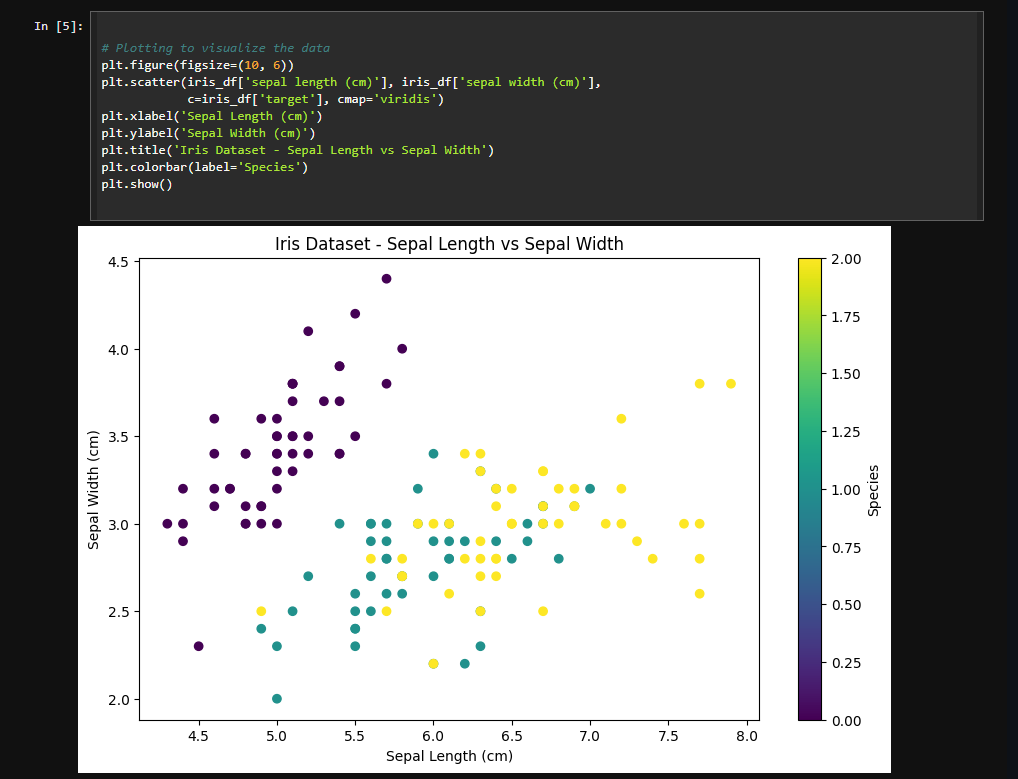
1. Build Support vector machine model for a given dataset

Algorithm:



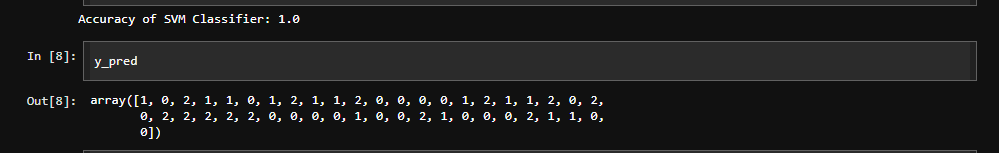
Code:





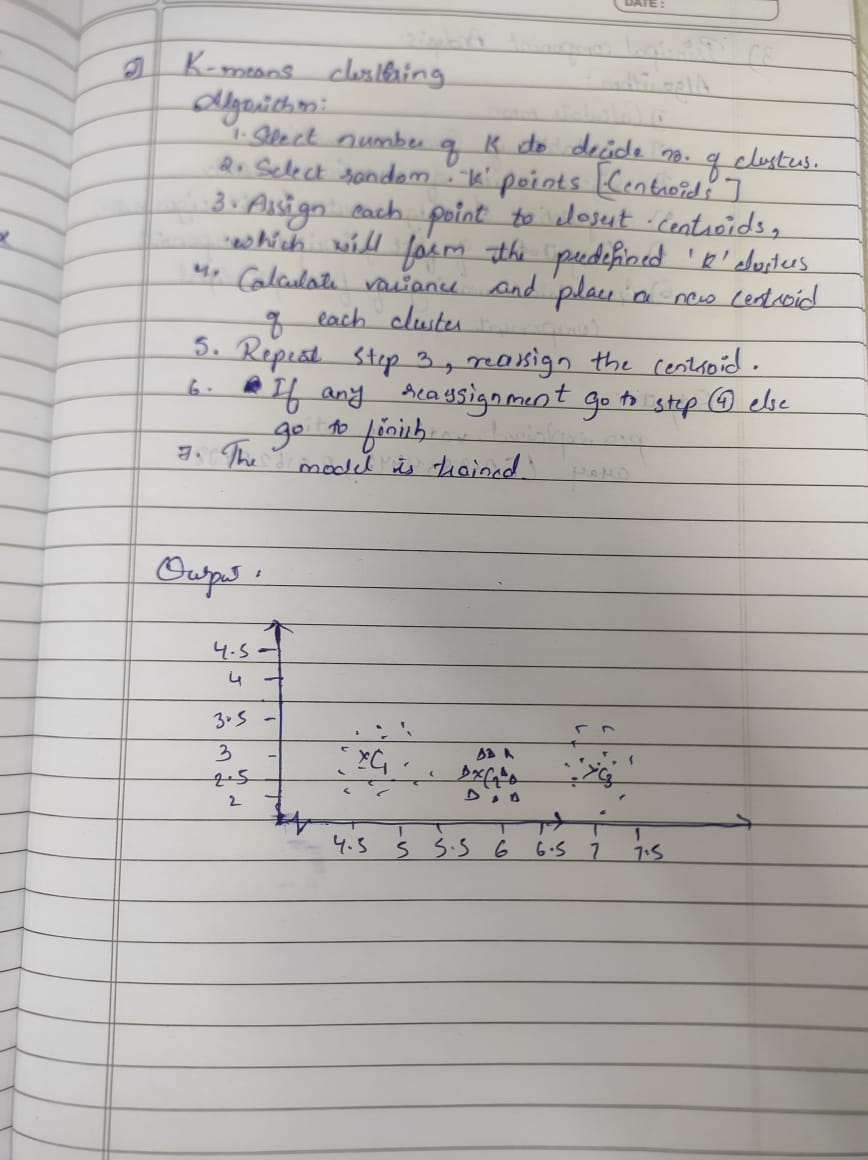


Results:

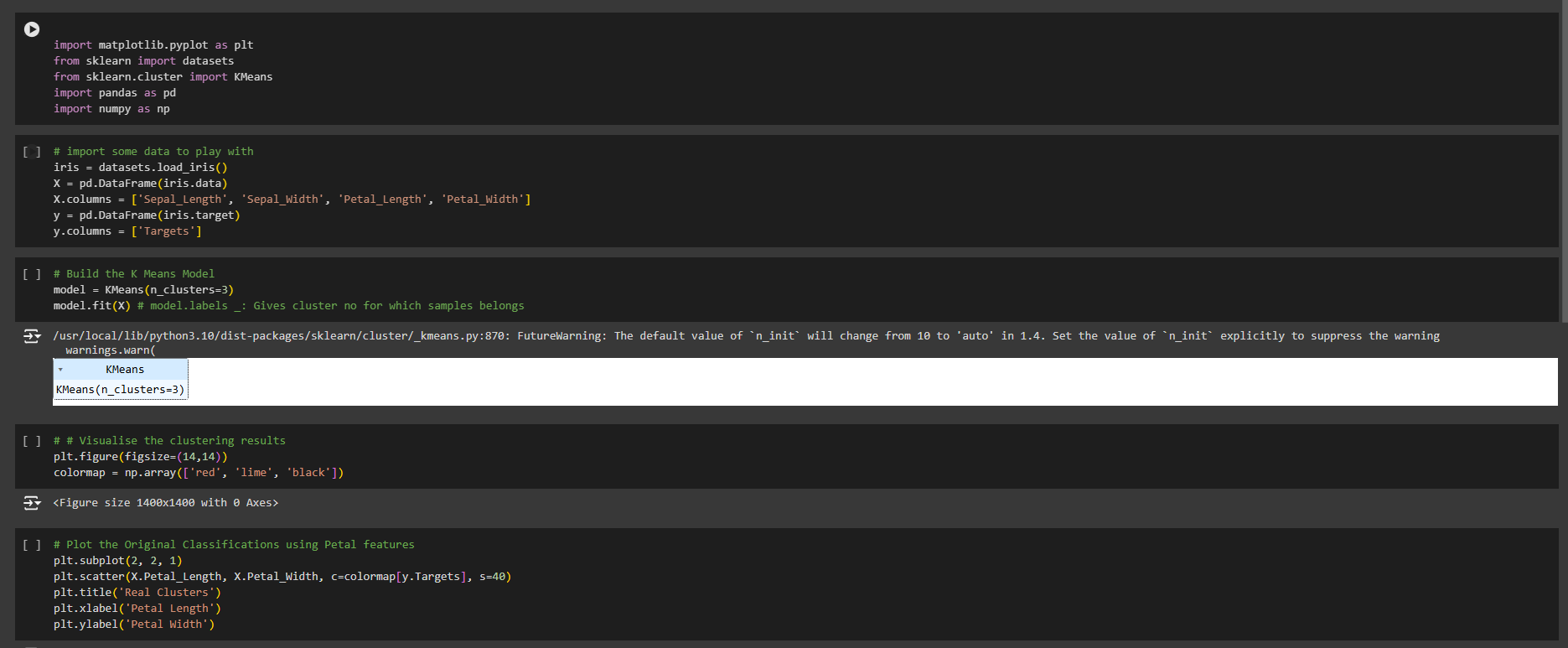


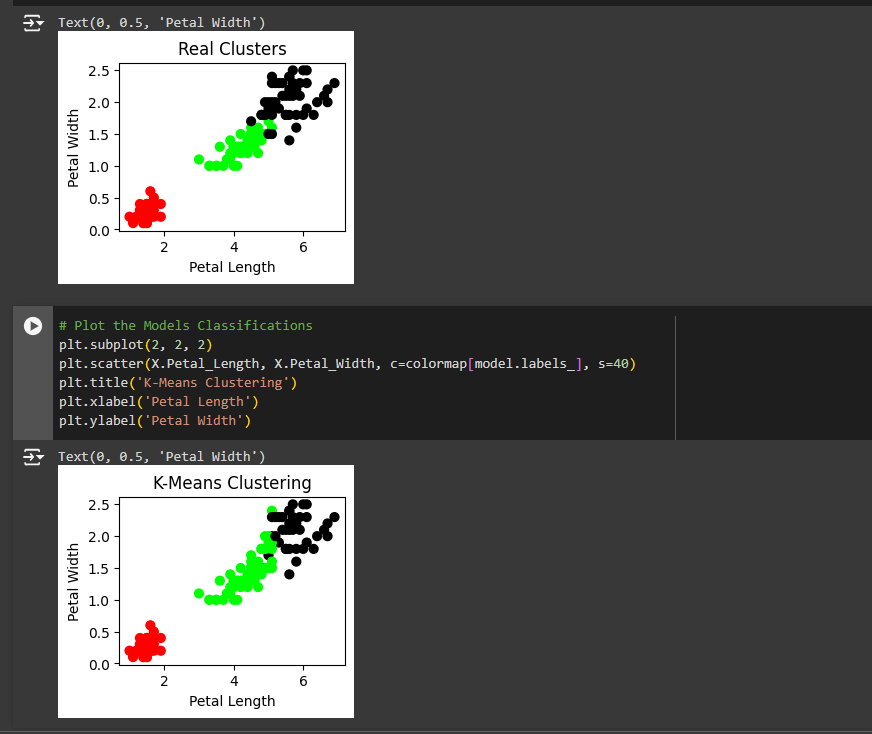
1. Build k-Means algorithm to cluster a set of data stored in a .CSV file.

Algorithm:



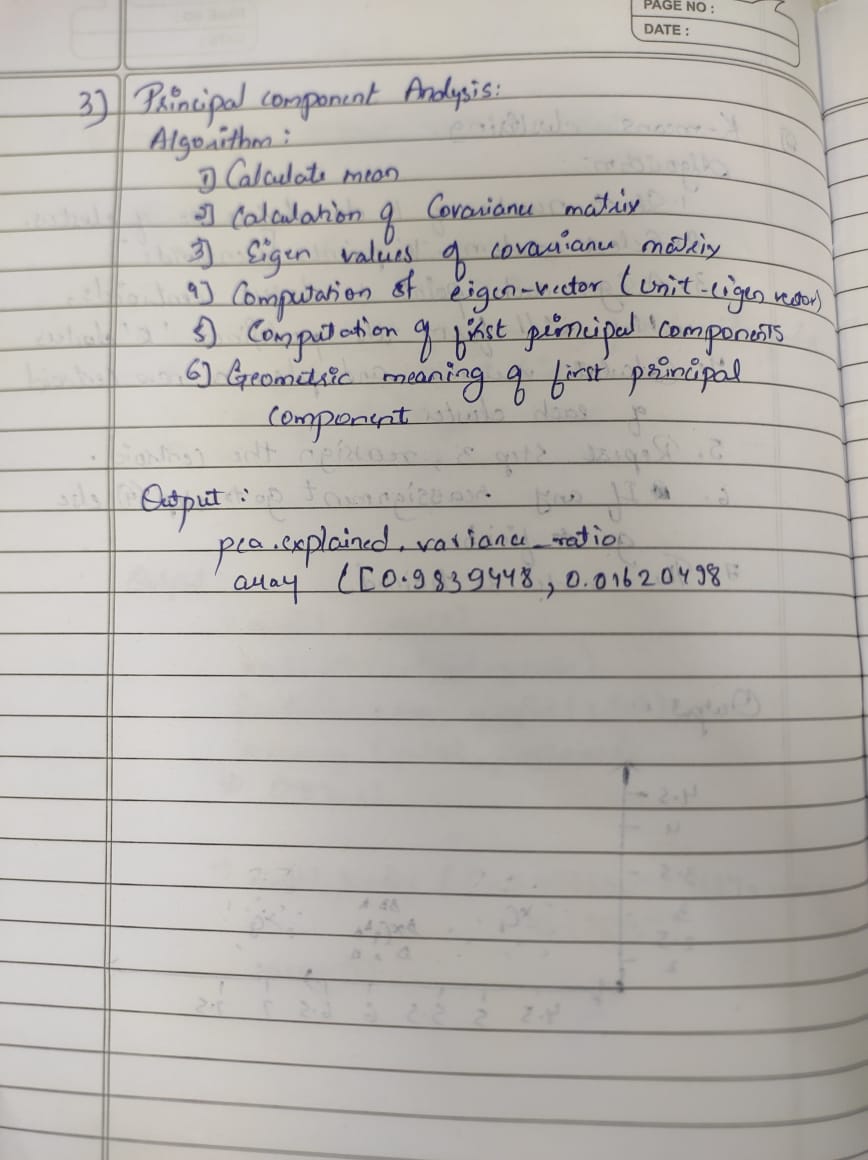
Code:



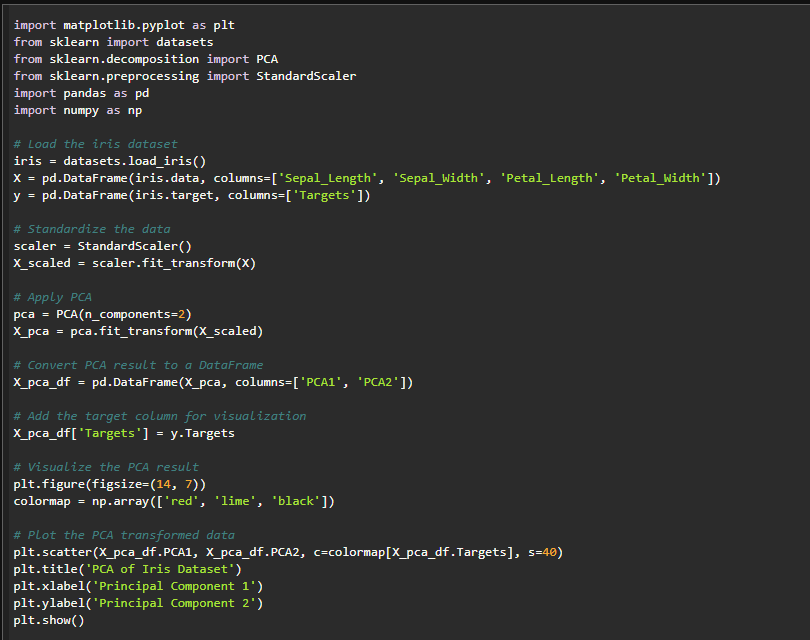


1. Implement Dimensionality reduction using Principle Component Analysis (PCA)  
   method.

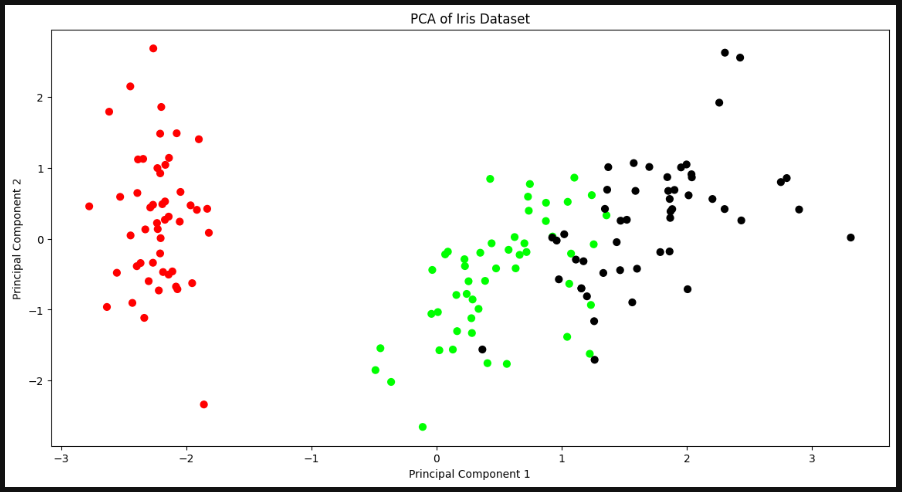
Algorithm:



Code:

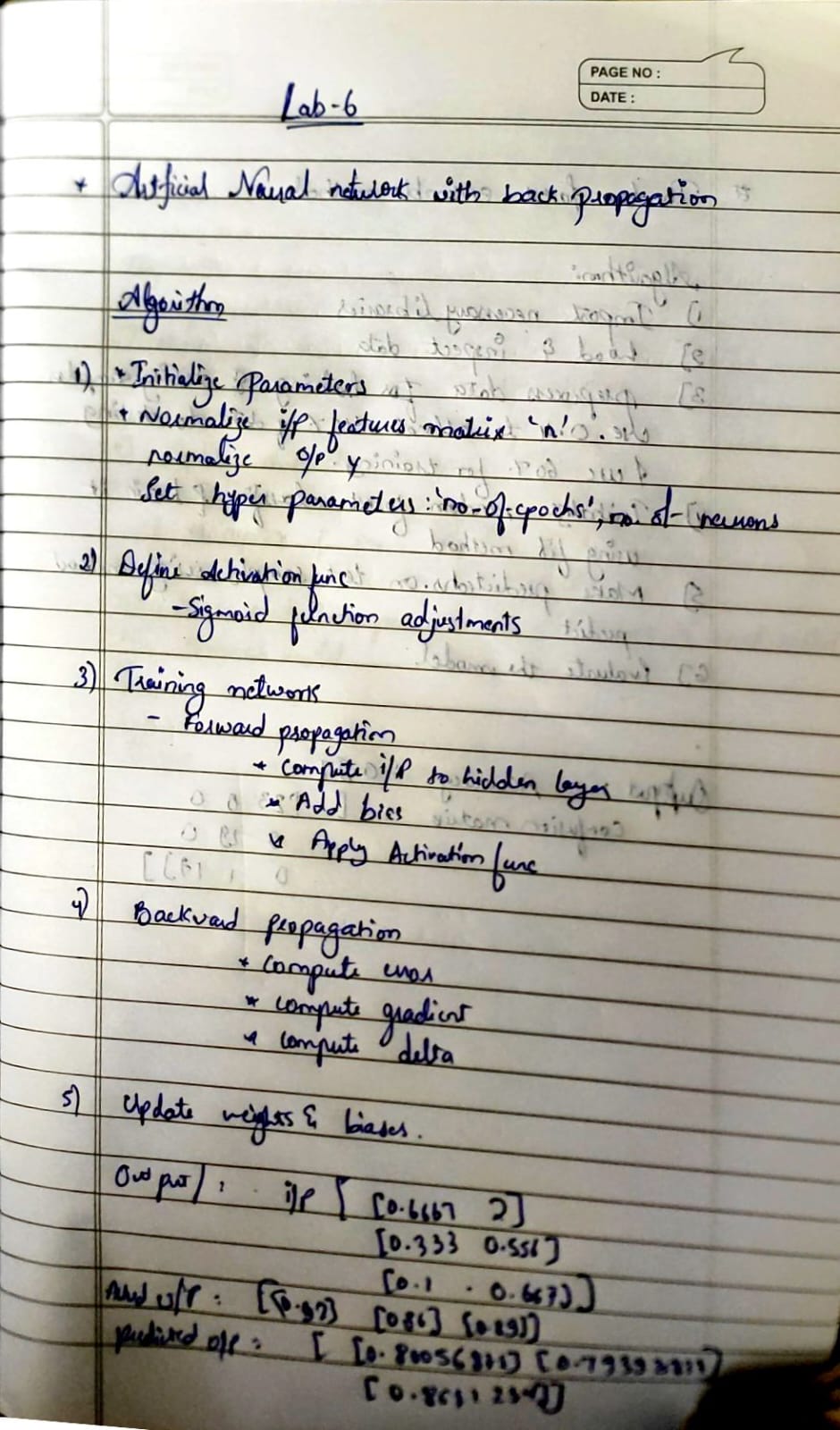


Results:

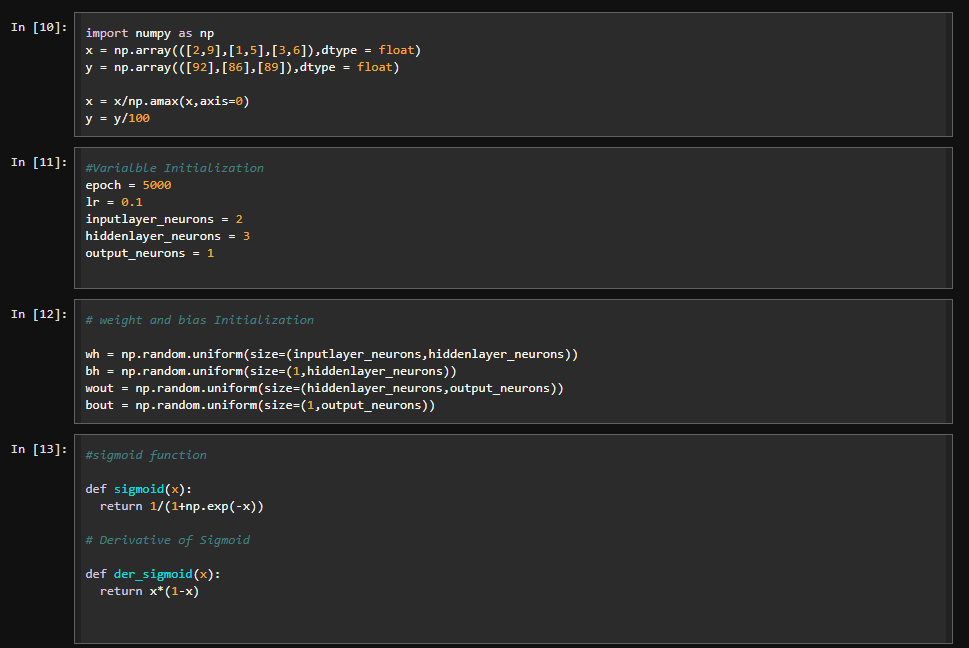


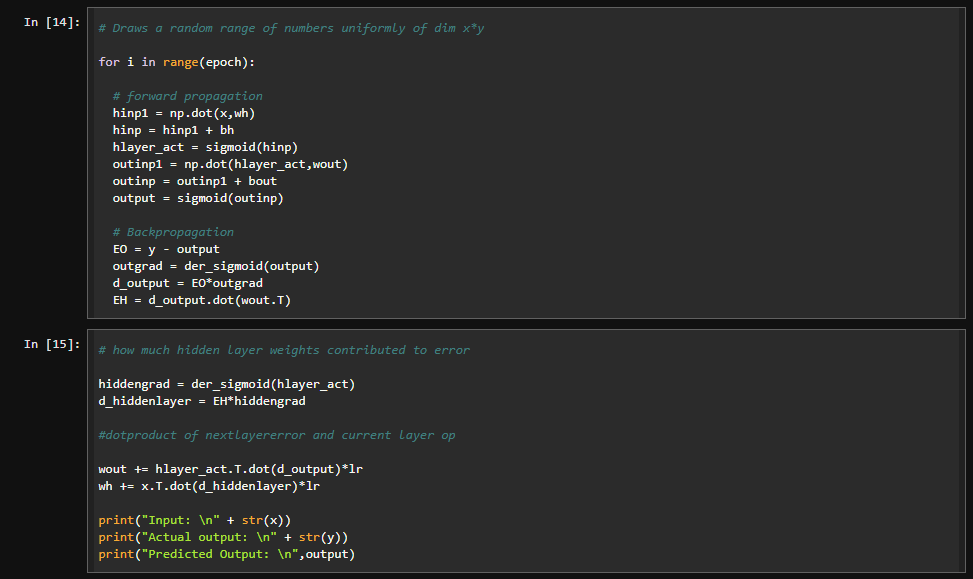
1. Build Artificial Neural Network model with back propagation on a given dataset

Algorithm:

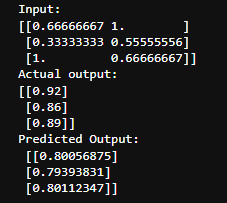


Code:



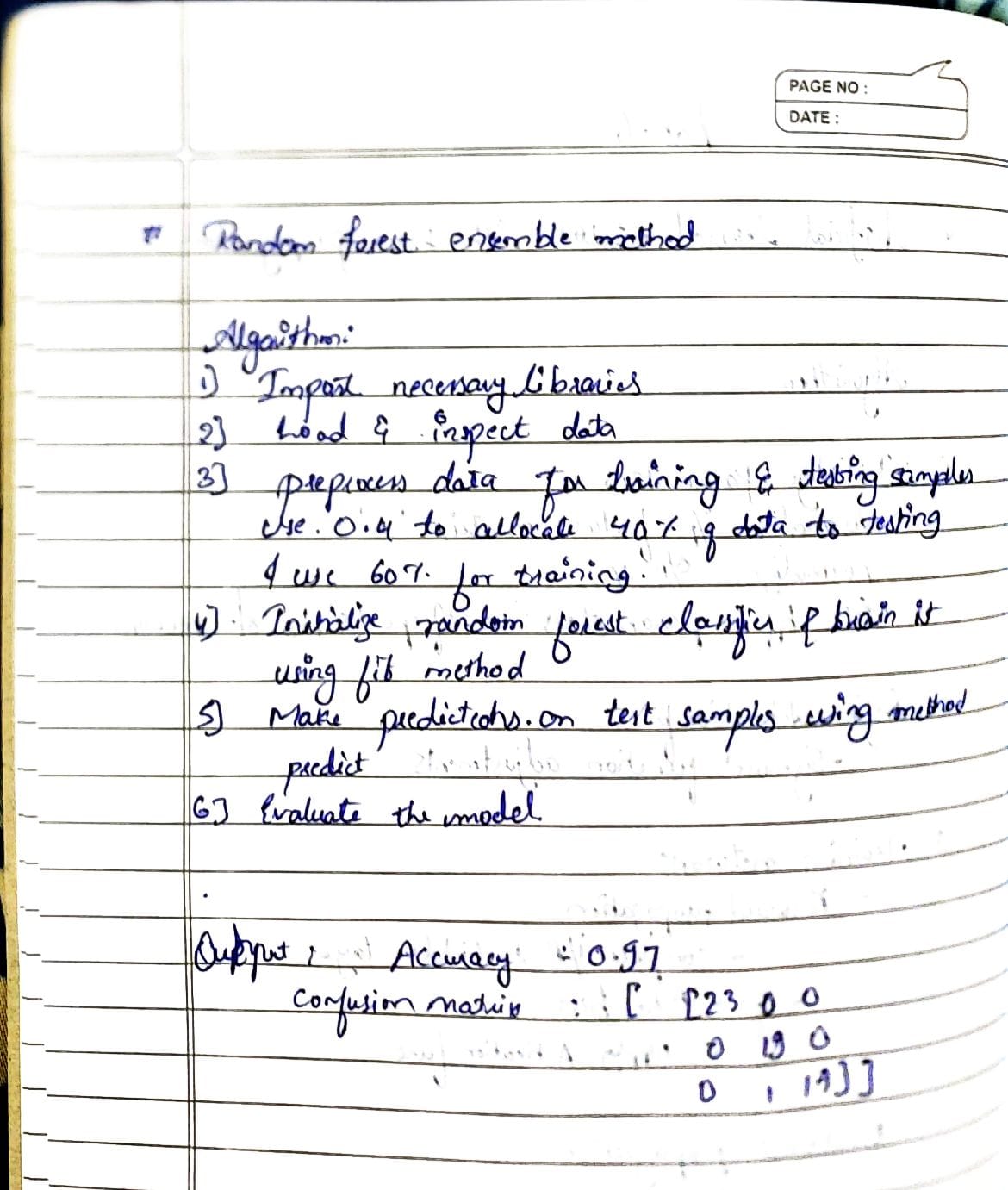


Results

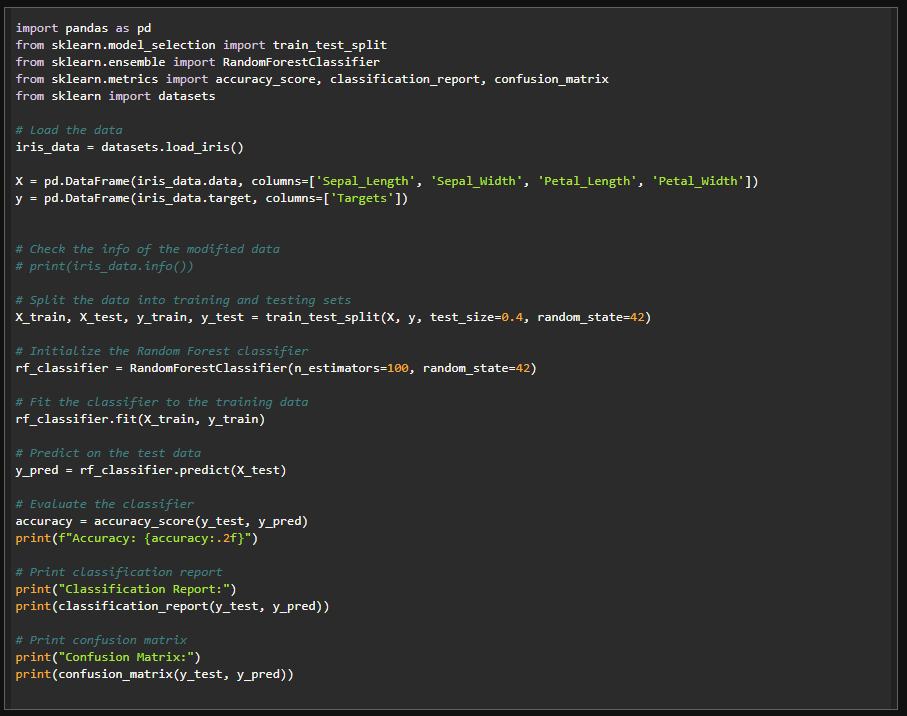


1. Implement Random forest ensemble method on a given dataset.

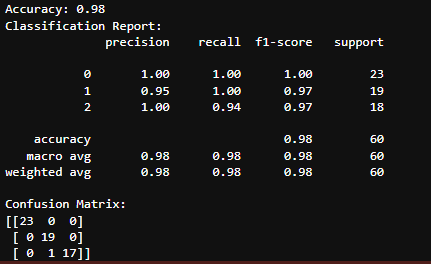
Algorithm:



Code:

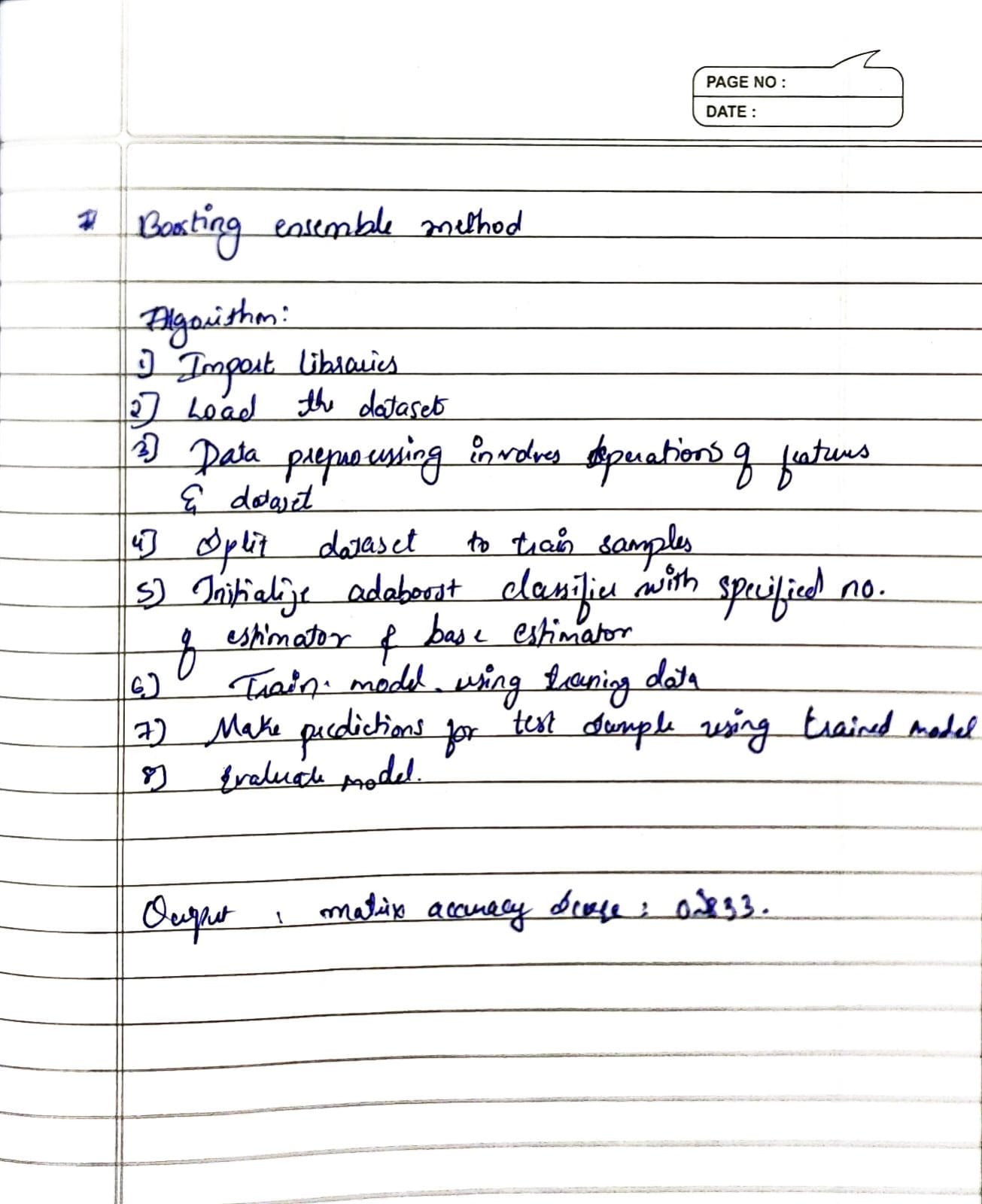


Results:

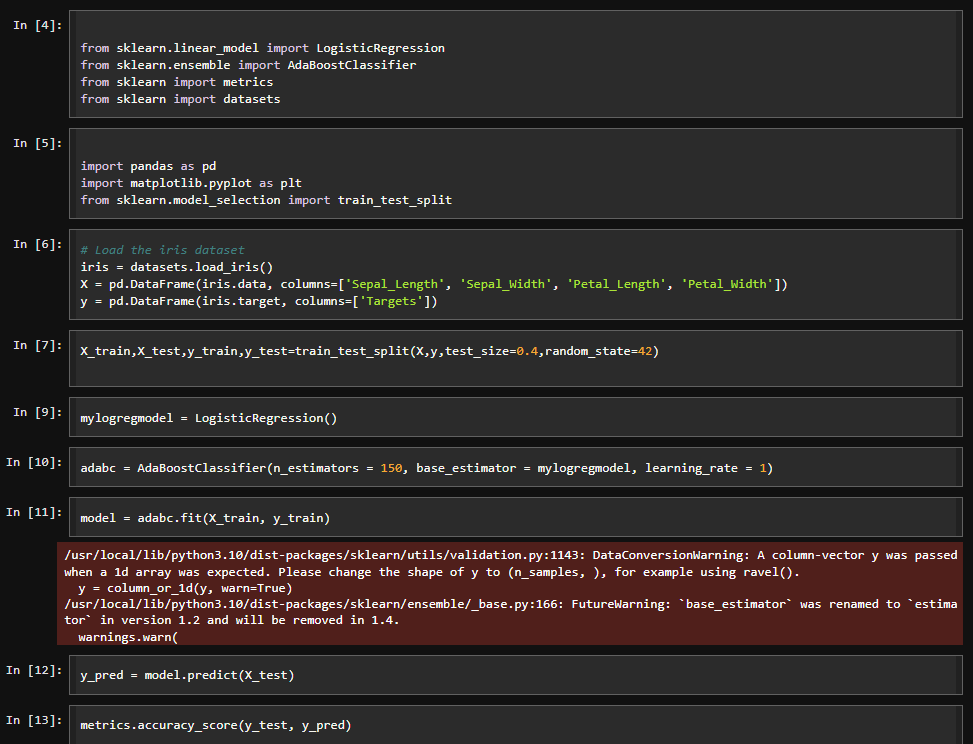


1. Implement Boosting ensemble method on a given dataset.

Algorithm



Code:



Results:

