**Name: Rohit Kumar Prajapati Roll No: 09**

**Subject: Machine Learning**

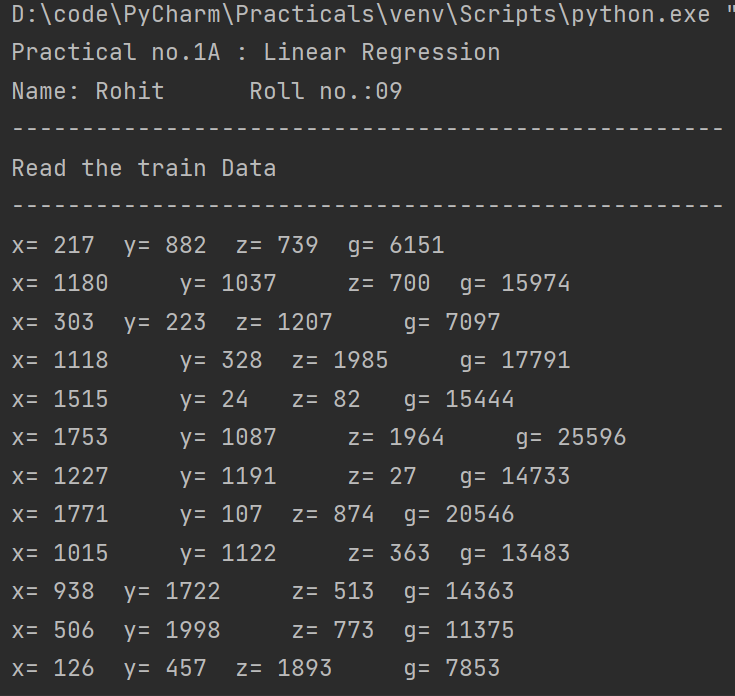
**Practical No – 1A**

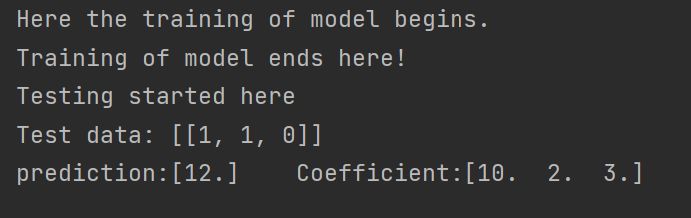
Design a simple machine learning model to train the training instances and test the same using Linear Regression model

**Code:**

*# Practical 1 # Linear Regression Model*print("Practical no.1A : Linear Regression \nName: Rohit \t Roll no.:09")  
import random  
from sklearn.linear\_model import LinearRegression  
  
print("Read the train Data")  
feature\_set = []  
target\_set = []  
no\_of\_rows = 200  
limit = 2000  
for i in range(0, no\_of\_rows):  
 x = random.randint(0, limit)  
 y = random.randint(0, limit)  
 z = random.randint(0, limit)  
 g = 10 \* x + 2 \* y + 3 \* z  
 print("x=", x, "\ty=", y, "\tz=", z, "\tg=", g)  
 feature\_set.append([x, y, z])  
 target\_set.append(g)  
print("Here the training of model begins. ")  
model = LinearRegression()  
model.fit(feature\_set, target\_set)  
print("Training of model ends here!")  
  
print("Testing started here")  
test\_data = [[1, 1, 0]]  
print("Test data:", test\_data)  
prediction = model.predict(test\_data)  
  
print("prediction:" + str(prediction) + '\t' + "Coefficient:" + str(model.coef\_))

**Output:**





**Practical No – 1B**

Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a .CSV file

**Dataset:**

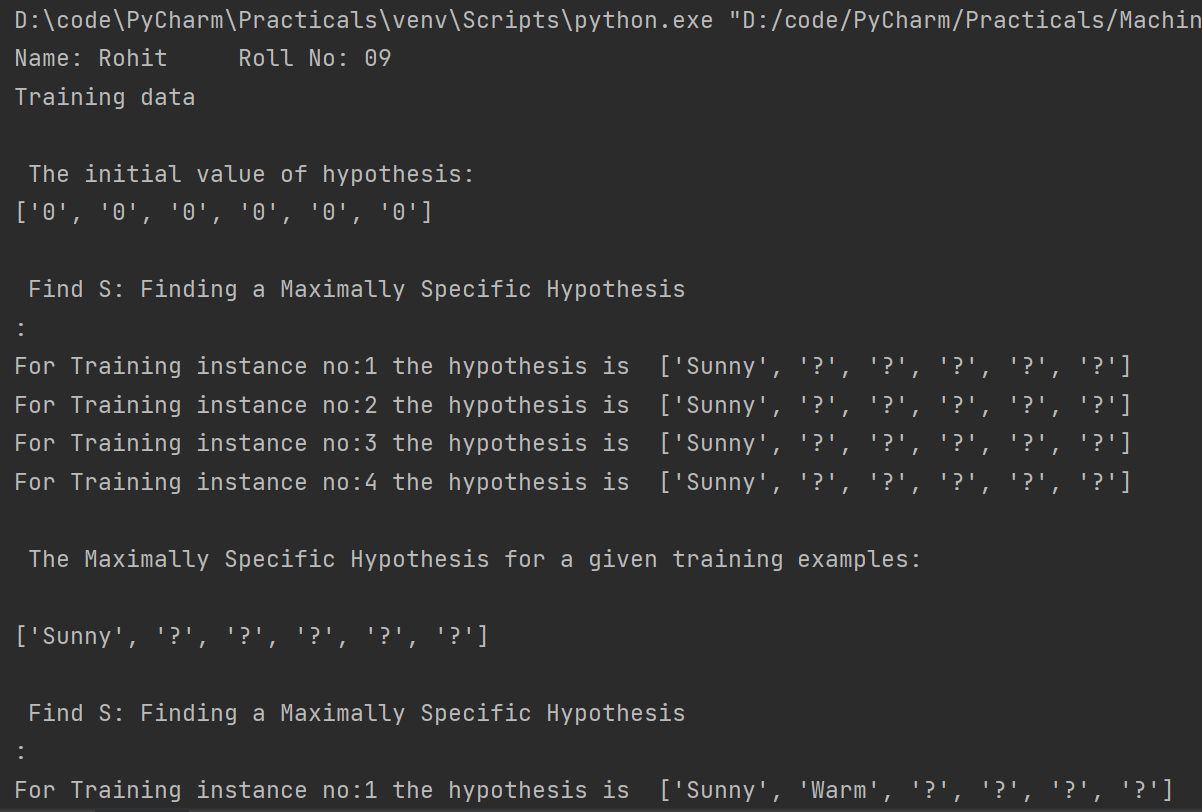
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **sky** | **air temp** | **humidity** | **wind** | **water** | **forecast** | **enjoy sport** |
| sunny | warm | normal | strong | warm | same | yes |
| sunny | warm | high | strong | warm | same | yes |
| rainy | cold | high | strong | warm | change | no |
| sunny | warm | high | strong | cool | change | yes |

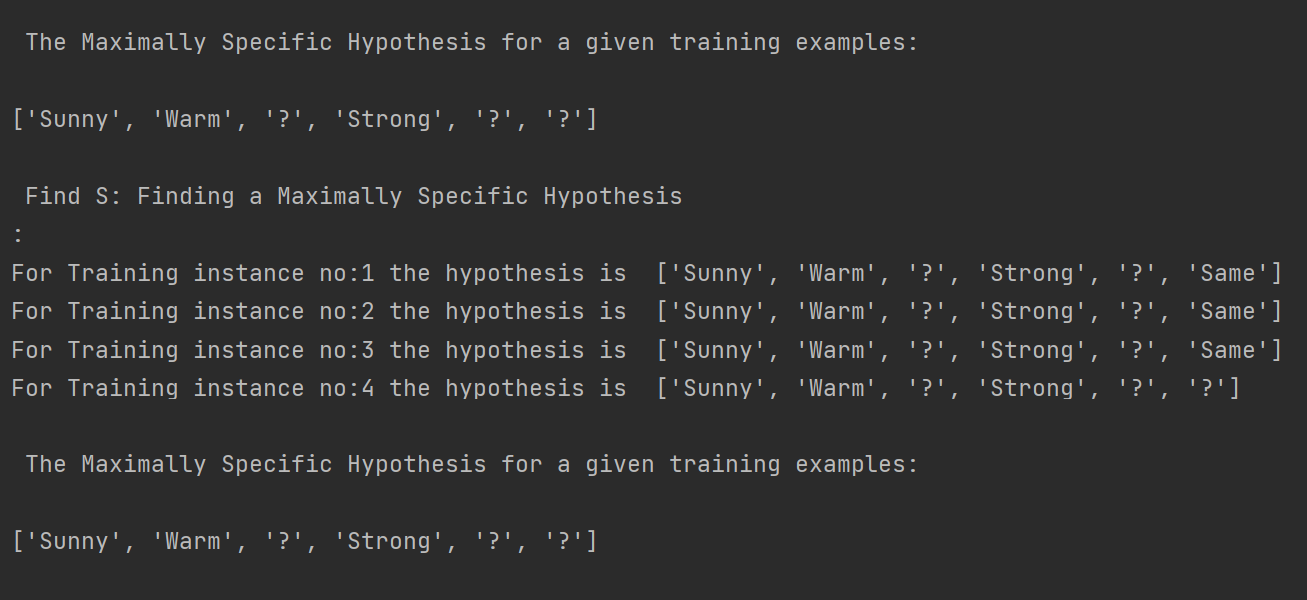
**Code:**

import csv  
print("Name: Rohit \tRoll No: 09")

num\_attributes = 6  
dataset = []  
print("Training data")  
  
with open(r"D:\code\PyCharm\Practicals\Machine Learning\data\_find\_s\_1b.csv", "r") as csvfile:  
 reader = csv.reader(csvfile)  
 for row in reader:  
 dataset.append(row)  
  
*# print(row)  
# print(dataset)*print("\n The initial value of hypothesis: ")  
hypothesis = ['0'] \* num\_attributes  
print(hypothesis)  
  
for j in range(0, num\_attributes):  
 hypothesis[j] = dataset[1][j]  
 print("\n Find S: Finding a Maximally Specific Hypothesis \n:")  
 for i in range(1, len(dataset)):  
 if dataset[i][num\_attributes] == 'Yes':  
 for j in range(0, num\_attributes):  
 if dataset[i][j] != hypothesis[j]:  
 hypothesis[j] = '?'  
 else:  
 hypothesis[j] = dataset[i][j]  
 print("For Training instance no:{0} the hypothesis is ".format(i), hypothesis)  
 print("\n The Maximally Specific Hypothesis for a given training examples: \n")  
 print(hypothesis)

**Output:**





**Practical No – 2A**

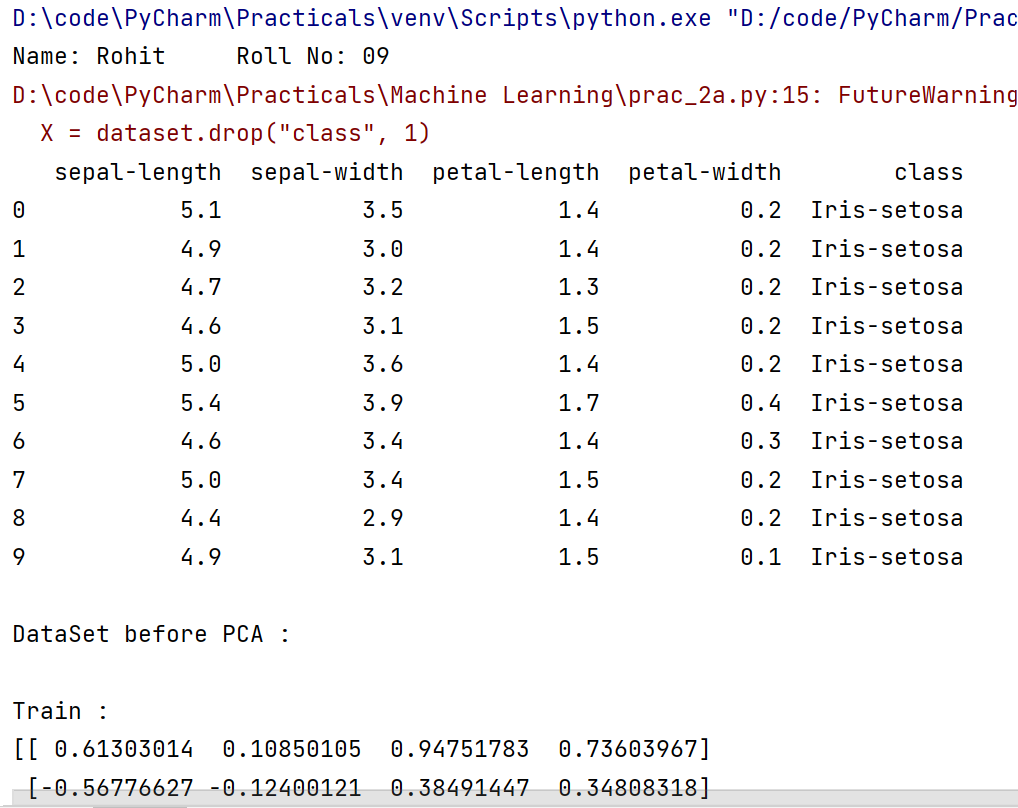
Perform Data Loading, Feature selection (Principal Component analysis) and Feature Scoring and Ranking.

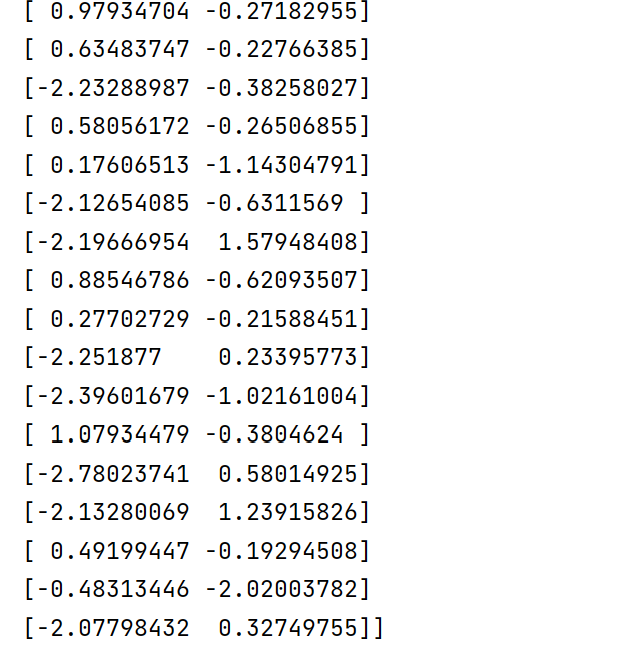
**Code:**

import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.decomposition import PCA  
from sklearn.preprocessing import StandardScaler  
  
print("Name: Rohit \tRoll No: 09")  
  
*# Reading dataset using pandas*url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'  
names = ["sepal-length", "sepal-width", "petal-length", "petal-width", "class"]  
dataset = pd.read\_csv(url, names=names)  
  
*# Display dataset*print(dataset.head(10))  
X = dataset.drop("class", 1)  
Y = dataset["class"]  
  
*# Splitting the train and test datasets*x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=0)  
  
sc = StandardScaler()  
x\_train = sc.fit\_transform(x\_train)  
x\_test = sc.transform(x\_test)  
  
*# Display Training and testing data*print(f"\nDataSet before PCA :\n\nTrain :\n{x\_train}\n\nTest :\n{x\_test}")  
*# print(x\_train)  
# print(x\_test)*

*# Creating PCA*pca = PCA()  
x\_train = pca.fit\_transform(x\_train)  
x\_test = pca.transform(x\_test)  
  
*# Giving a principal feature to model*pca = PCA(n\_components=2)  
x\_train = pca.fit\_transform(x\_train)  
x\_test = pca.transform(x\_test)  
  
print(f"\nDataSet After PCA :\n\nTrain :\n{x\_train}\n\nTest :\n{x\_test}")

**Output:**

****

****

**Practical No – 2B**

For a given set of training data examples stored in .CSV file, implement and demonstrate the Candidate- Elimination algorithm to output a description of the set of all hypotheses consistent with the training examples.

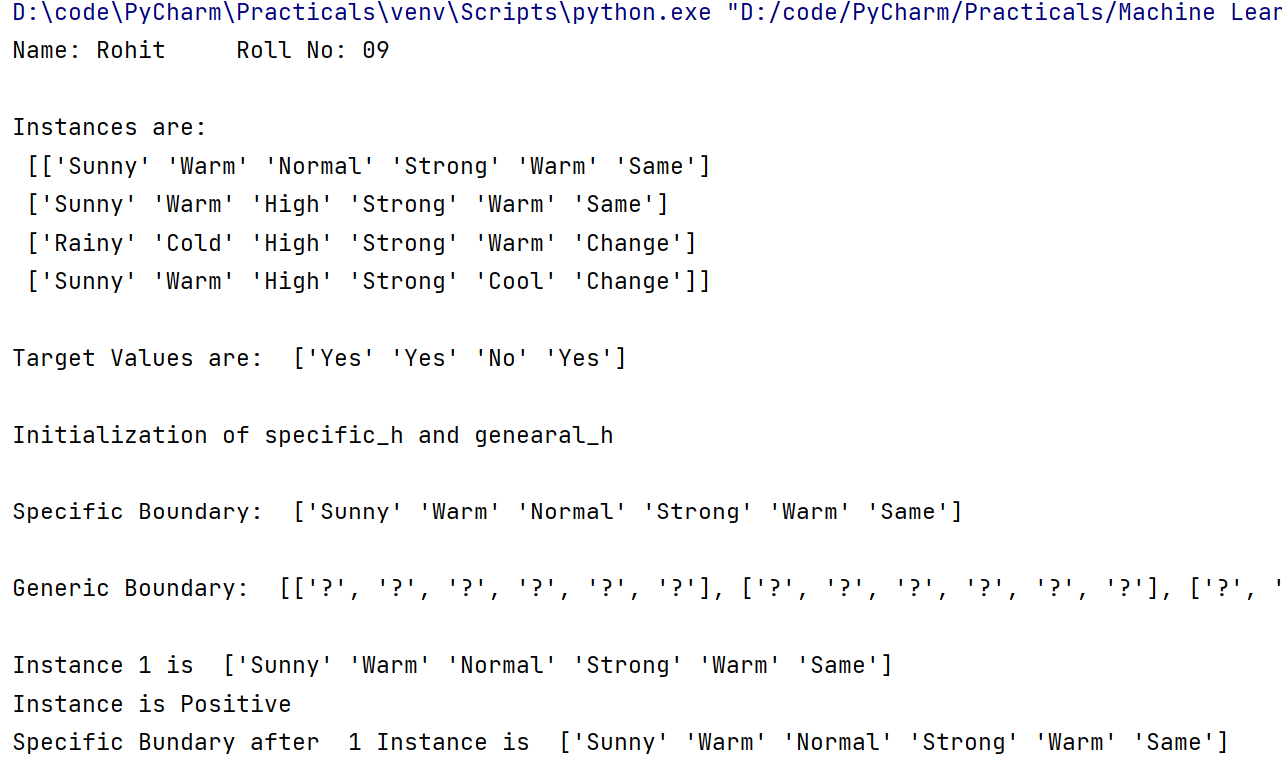
**Dataset:**

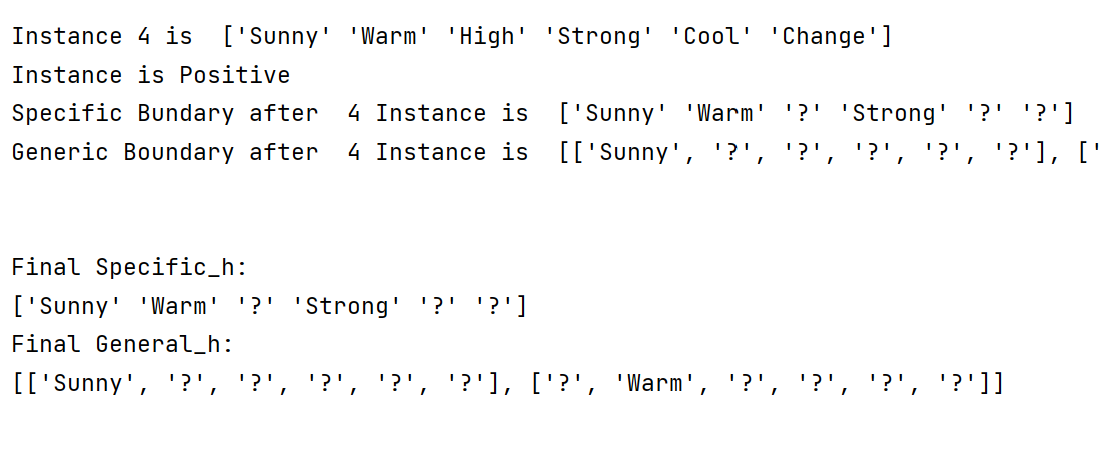
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **sky** | **air temp** | **humidity** | **wind** | **water** | **forecast** | **enjoy sport** |
| sunny | warm | normal | strong | warm | same | yes |
| sunny | warm | high | strong | warm | same | yes |
| rainy | cold | high | strong | warm | change | no |
| sunny | warm | high | strong | cool | change | yes |

**Code:**

import numpy as np  
import pandas as pd  
  
data = pd.read\_csv(r'D:\code\PyCharm\Practicals\Machine Learning\data\_find\_s\_1b.csv')  
concepts = np.array(data.iloc[:, 0:-1])  
print("\nInstances are:\n", concepts)  
target = np.array(data.iloc[:, -1])  
print("\nTarget Values are: ", target)  
  
  
def learn(concepts, target):  
 specific\_h = concepts[0].copy()  
 print("\nInitialization of specific\_h and genearal\_h")  
 print("\nSpecific Boundary: ", specific\_h)  
 general\_h = [["?" for i in range(len(specific\_h))] for i in range(len(specific\_h))]  
 print("\nGeneric Boundary: ", general\_h)  
  
 for i, h in enumerate(concepts):  
 print("\nInstance", i + 1, "is ", h)  
 if target[i] == "yes":  
 print("Instance is Positive ")  
 for x in range(len(specific\_h)):  
 if h[x] != specific\_h[x]:  
 specific\_h[x] = '?'  
 general\_h[x][x] = '?'  
  
 if target[i] == "no":  
 print("Instance is Negative ")  
 for x in range(len(specific\_h)):  
 if h[x] != specific\_h[x]:  
 general\_h[x][x] = specific\_h[x]  
 else:  
 general\_h[x][x] = '?'  
  
 print("Specific Bundary after ", i + 1, "Instance is ", specific\_h)  
 print("Generic Boundary after ", i + 1, "Instance is ", general\_h)  
 print("\n")  
  
 indices = [i for i, val in enumerate(general\_h) if val == ['?', '?', '?', '?', '?', '?']]  
 for i in indices:  
 general\_h.remove(['?', '?', '?', '?', '?', '?'])  
 return specific\_h, general\_h  
  
  
s\_final, g\_final = learn(concepts, target)  
  
print("Final Specific\_h: ", s\_final, sep="\n")  
print("Final General\_h: ", g\_final, sep="\n")

**Output:**

****

****

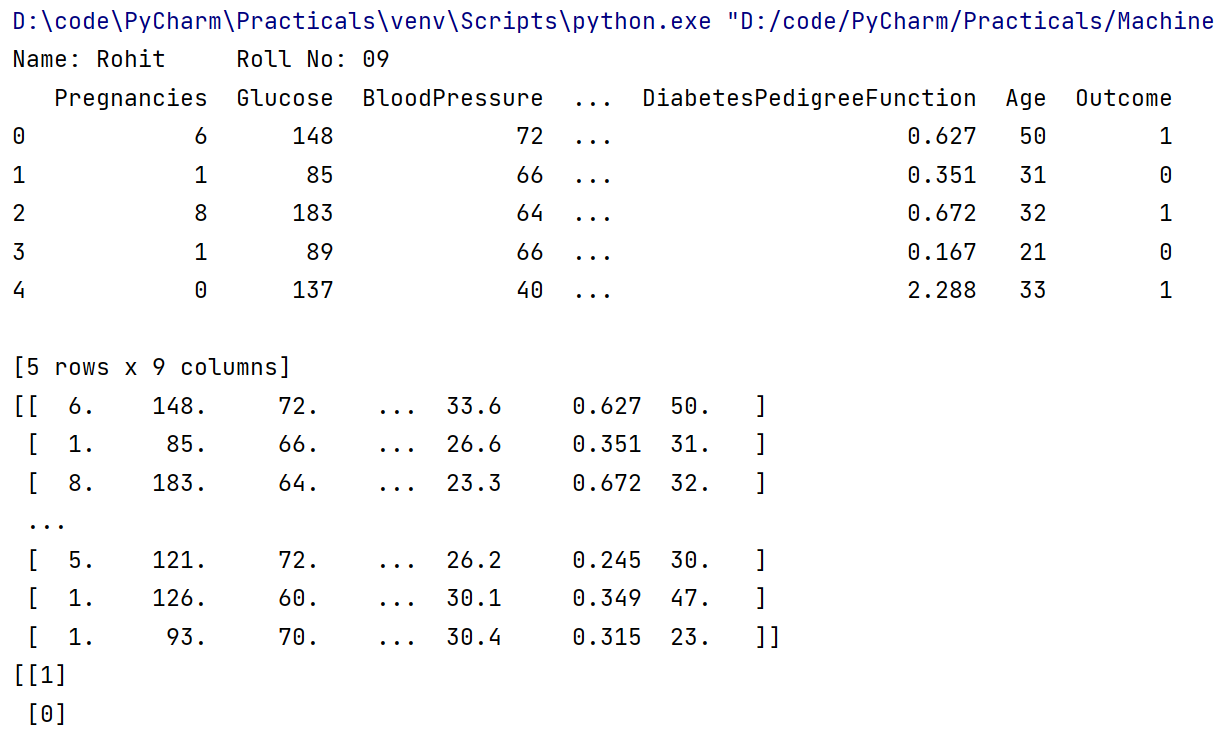
**Practical No – 4B**

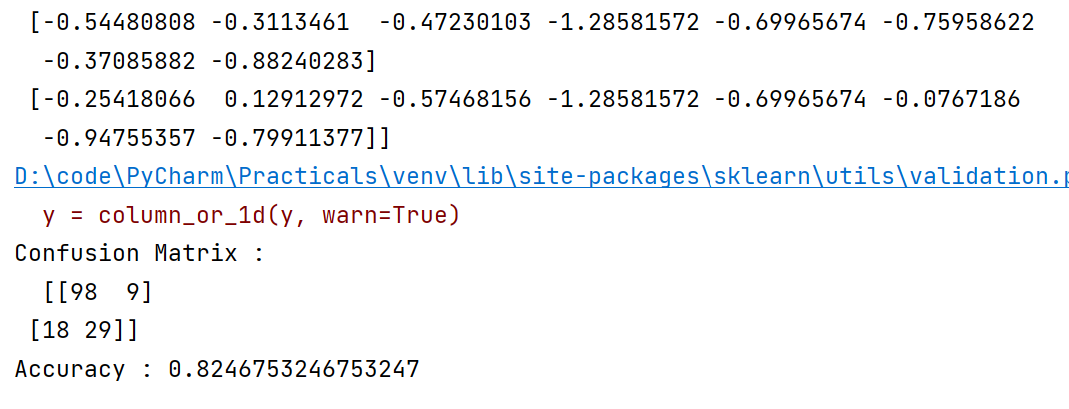
For a given set of training data examples stored in a .CSV file implement Logistic Regression algorithm. (Use Multivariate dataset )

**Code:**

import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.preprocessing import StandardScaler  
from sklearn.linear\_model import LogisticRegression  
from sklearn.metrics import confusion\_matrix, accuracy\_score  
  
  
print("Name: Rohit \tRoll No: 09")  
  
dataset = pd.read\_csv("https://raw.githubusercontent.com/plotly/datasets/master/diabetes.csv")  
print(dataset.head())  
x = dataset.iloc[:, [0, 1, 2, 3, 4, 5, 6, 7]].values  
y = dataset.iloc[:, [-1]].values  
print(x)  
print(y)  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=0)  
sc = StandardScaler()  
x\_train = sc.fit\_transform(x\_train)  
x\_test = sc.transform(x\_test)  
  
print(x\_train[0:15, :])  
  
classifier = LogisticRegression()  
classifier.fit(x\_train, y\_train)  
  
y\_pred = classifier.predict(x\_test)  
  
cm = confusion\_matrix(y\_test, y\_pred)  
print("Confusion Matrix :\n ", cm)  
  
print("Accuracy :", accuracy\_score(y\_test, y\_pred))

**Output:**

****

****