**Practical: 6**

**Aim: To Implement principal component analysis**

**Code:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn import metrics, preprocessing

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

import matplotlib.pyplot as plt

import numpy as np

data = pd.read\_csv('diabetes.csv')

# Data Preprocessing

preprocessing = False

#preprocessing = True

if preprocessing:

data['Glucose'].fillna(data['Glucose'].mean(), inplace=True)

data['BloodPressure'].fillna(data['BloodPressure'].mean(), inplace=True)

data['SkinThickness'].fillna(data['SkinThickness'].mean(), inplace=True)

data['Insulin'].fillna(data['Insulin'].mean(), inplace=True)

data['BMI'].fillna(data['BMI'].mean(), inplace=True)

# Correlation Matrix for Diabetes Data

corr = data.corr()

# print(corr.values)

plt.figure(figsize=(17, 7))

X = data.drop(['Outcome'], axis=1).values

y = data['Outcome'].values

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,

y,

test\_size=0.2,

random\_state=0)

print('training data : ', X\_train.shape, y\_train.shape)

print('testing data : ', X\_test.shape, y\_test.shape)

# Decision Tree Classifier

dtc = DecisionTreeClassifier()

dtc = dtc.fit(X\_train, y\_train)

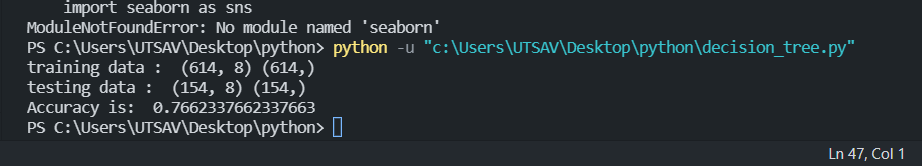
y\_pred = dtc.predict(X\_test)

acc = accuracy\_score(y\_pred, y\_test)

print('Accuracy is: ', acc)

**# Ouput:**

1. Without preprocessing



1. With preprocessing

