**MACHINE LEARNING ASSIGNMENT -1** DINESHS(018)

**Dataset Name** : Diabetes

**Dataset link** : <https://drive.google.com/file/d/1I3vLLCYnnBOfksEA-7iewkEDCMYLY37l/view?usp=sharing>

**Feature Selection algorithm :**

**1 .PCA (Principal component analysis) :**

**Code:**

import pandas as pd

from sklearn.decomposition import PCA

from sklearn.preprocessing import StandardScaler

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn.tree import DecisionTreeClassifier

# Step 1: Load the dataset

data = pd.read\_csv('./diabetes1.csv')  # Update this with the correct path

# Step 2: Preprocess the data

# Separate features and target

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

y = data['Outcome']

# Step 3: Standardize the features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# Step 4: Apply PCA

pca = PCA(n\_components=2)  # Reduce to 2 principal components

X\_pca = pca.fit\_transform(X\_scaled)

# Step 5: Show the explained variance ratio

print("Explained variance ratio:", pca.explained\_variance\_ratio\_)

# Display the first few rows of the PCA-transformed dataset

print("PCA-transformed data (first 5 rows):\n", X\_pca[:20])

# # Step 6: Split the PCA data into train and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_pca, y, test\_size=0.2, random\_state=42)

clf = DecisionTreeClassifier(random\_state=42)

clf.fit(X\_train, y\_train)

# Step 5: Make Predictions

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

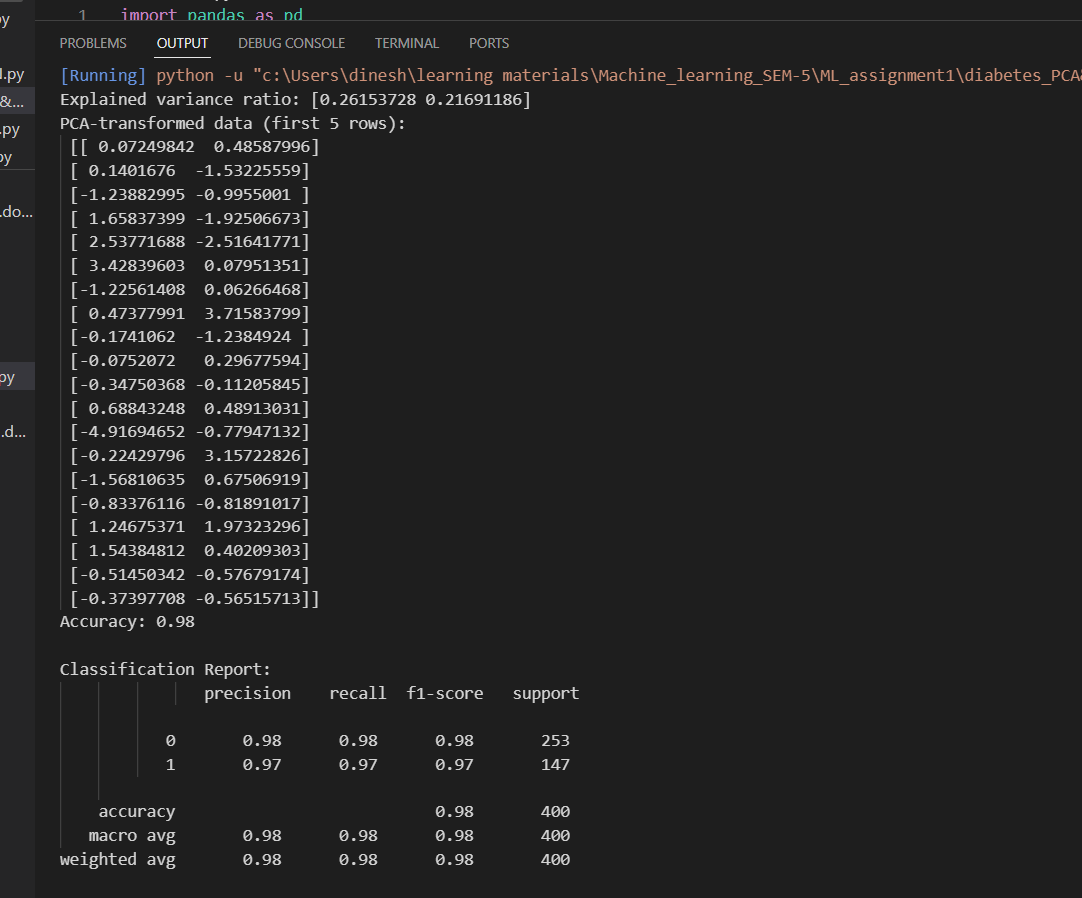
# Step 6: Evaluate the Model

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

print("Accuracy:", accuracy)

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

**Output:**

****

**2 . LDA (Linear discriminant analysis) :**

**Code:**

import pandas as pd

from sklearn.discriminant\_analysis import LinearDiscriminantAnalysis as LDA

from sklearn.preprocessing import StandardScaler

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, classification\_report

data = pd.read\_csv('./diabetes1.csv')  # Update this with the correct path

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

y = data['Outcome']

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

lda = LDA(n\_components=1)  # Reduce to 1 linear discriminant component

X\_lda = lda.fit\_transform(X\_scaled, y)

print("Explained variance ratio (Not applicable for LDA): N/A")  # LDA does not provide variance ratio like PCA

print("LDA-transformed data (first 5 rows):\n", X\_lda[:40])

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_lda, y, test\_size=0.2, random\_state=42)

model = LogisticRegression()

model.fit(X\_train, y\_train)

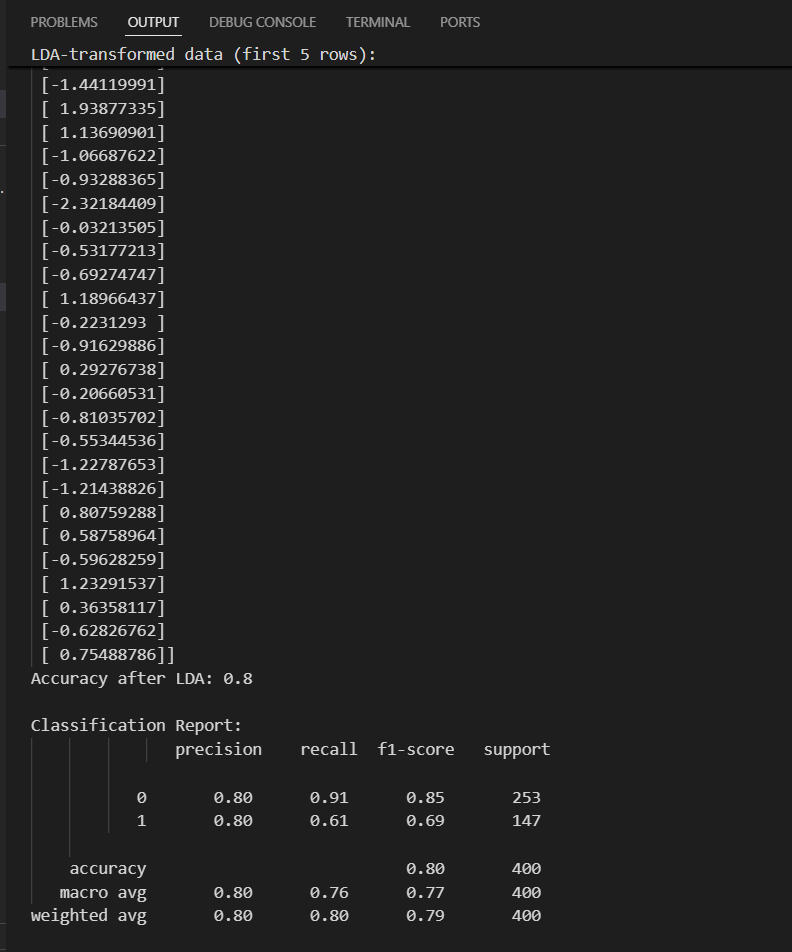
y\_pred = model.predict(X\_test)

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

print("Accuracy after LDA:", accuracy)

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

**output:**



**Classification algorithm :**

1. **Decision Tree :**

**Code :**

import pandas as pd

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

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, classification\_report

data = pd.read\_csv("C:/Users/dinesh/learning materials/Machine\_learning\_SEM-5/ML\_assignment1/diabetes1.csv")  # Replace with your actual CSV file path

data[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']] = data[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']].replace(0, pd.NA)

data.fillna(data.mean(), inplace=True)

X = data.drop('Outcome', axis=1)  # Features (all columns except Outcome)

y = data['Outcome']  # Labels (Outcome column)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

clf = DecisionTreeClassifier(random\_state=42)

clf.fit(X\_train, y\_train)

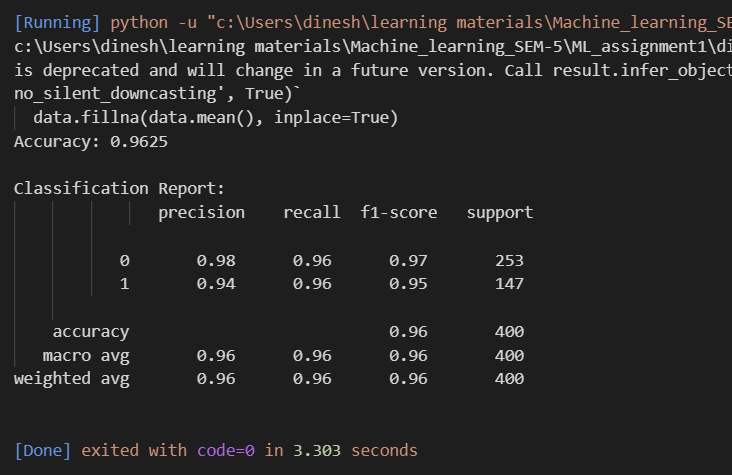
y\_pred = clf.predict(X\_test)

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

print("Accuracy:", accuracy)

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

**output :**



**2 . KNN (K-Nearest neighbour) algorithm :**

**Code :**

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score, classification\_report

data = pd.read\_csv('./diabetes1.csv')  # Update file path if needed

columns\_to\_replace = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']

data[columns\_to\_replace] = data[columns\_to\_replace].replace(0, pd.NA)

data.fillna(data.mean(), inplace=True)

X = data.drop('Outcome', axis=1)  # Features (all columns except Outcome)

y = data['Outcome']  # Target label (Outcome column)

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

knn\_clf = KNeighborsClassifier(n\_neighbors=20)  # You can change the value of k (number of neighbors)

knn\_clf.fit(X\_train, y\_train)

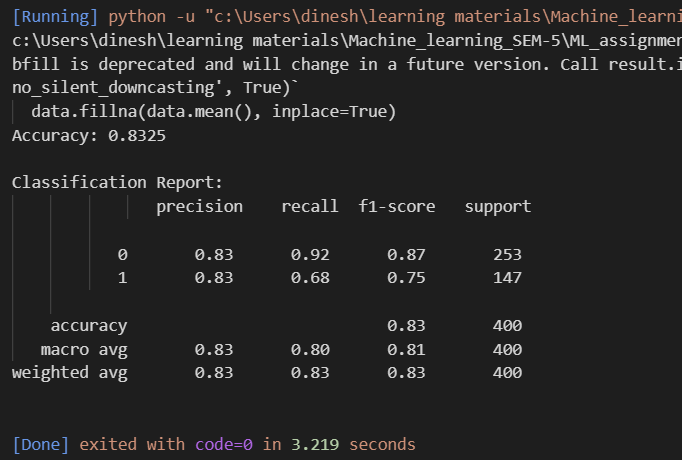
y\_pred = knn\_clf.predict(X\_test)

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

print("Accuracy:", accuracy)

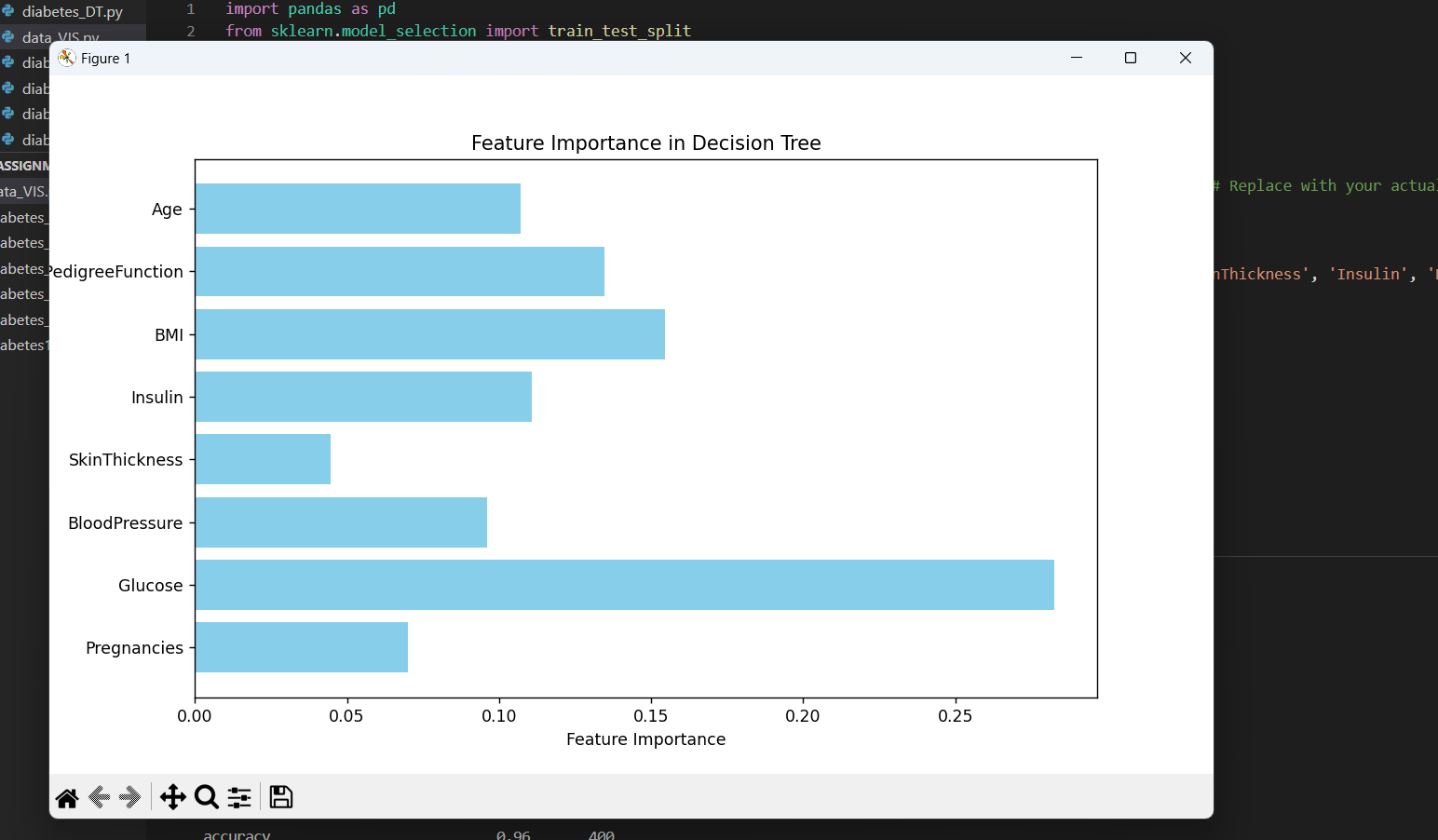
print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

**output :**



**DATA VISUALIZATION:**

**In Decision Tree :**

****