Lavina Golani

Batch A

16

**Experiment 2**

Aim:To implement PCA(Principal Component Analysis)

Theory: PCA is used to reduce your dataset.It combines features to give new features. It is used for visualization and to reduce dataset size.

Code:

import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

import numpy as np

# import matplotlib

# matplotlib.use('agg')

import seaborn as sns

import matplotlib.pyplot as plt

import json

file = '/home/lavina/Desktop/pima-indians-diabetes-database/diabetes.csv'

# load data set into frame

dataFrame = pd.read\_csv(file, names=['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'])

features = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']

# Separating out the features

x = dataFrame.loc[:, features].values

xplot =x

# Separating out the target

y = dataFrame.loc[:, ['Outcome']].values

# Standardizing the features mean=0 variance =1

x = StandardScaler().fit\_transform(x)

pca = PCA(n\_components=8)

Components = pca.fit\_transform(x)

print(np.cumsum(pca.explained\_variance\_ratio\_))

plt.plot(np.cumsum(pca.explained\_variance\_ratio\_))

plt.xlabel('no of components')

plt.ylabel('cum explained variance')

plt.show()

# so 5 components show 90% of the data

pca1 = PCA(n\_components=5)

principalComponents = pca1.fit\_transform(x)

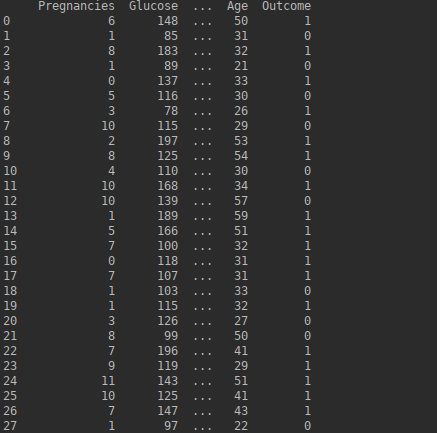
principalDf = pd.DataFrame(data=principalComponents, columns=['pc1', 'pc2', 'pc3', 'pc4', 'pc5'])

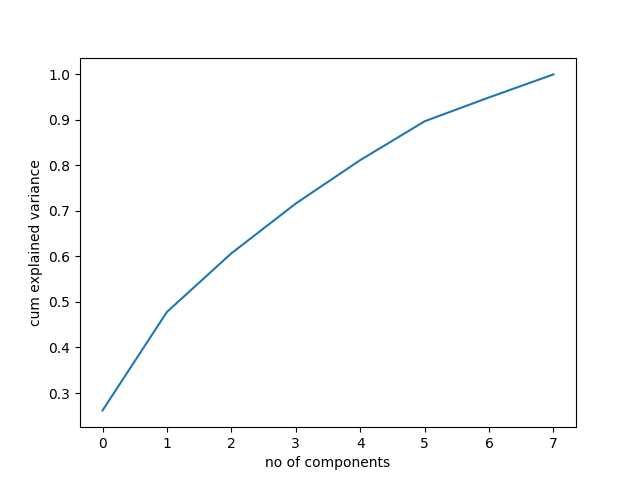
finalDf = pd.concat([principalDf, dataFrame[['Outcome']]], axis=1)

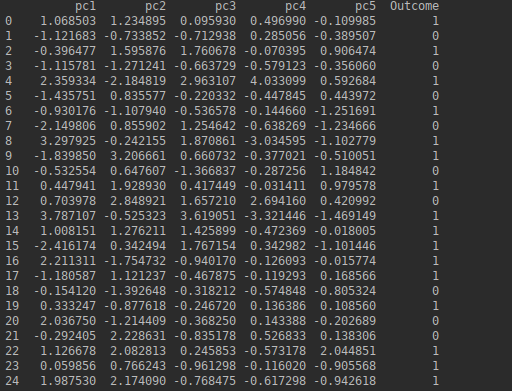
print(finalDf)

Output:

Initial dataset



Dataset after PCA



Conclusion :Hence we have successfully implemented PCA