MACHINE LEARINING

LAB ASSESSMENT – V

15BCE0082 VOLETI RAVI

**Implement Principle Component Analysis for Dimensionality Reduction.**

**CODE:**

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns; sns.set()

from sklearn.decomposition import PCA

rng = np.random.RandomState(1)

X = np.dot(rng.rand(2, 2), rng.randn(2, 200)).T

plt.scatter(X[:, 0], X[:, 1])

plt.axis('equal');

pca = PCA(n\_components=2)

pca.fit(X)

print(pca.components\_)

print(pca.explained\_variance\_)

def draw\_vector(v0, v1, ax=None):

ax = ax or plt.gca()

arrowprops=dict(arrowstyle='->', linewidth=2, shrinkA=0, shrinkB=0)

ax.annotate('', v1, v0, arrowprops=arrowprops)

# plot data

plt.scatter(X[:, 0], X[:, 1], alpha=0.2)

for length, vector in zip(pca.explained\_variance\_, pca.components\_):

v = vector \* 3 \* np.sqrt(length)

draw\_vector(pca.mean\_, pca.mean\_ + v)

plt.axis('equal');

pca = PCA(n\_components=1)

pca.fit(X)

X\_pca = pca.transform(X)

print("original shape: ", X.shape)

print("transformed shape:", X\_pca.shape)

X\_new = pca.inverse\_transform(X\_pca)

plt.scatter(X[:, 0], X[:, 1], alpha=0.2)

plt.scatter(X\_new[:, 0], X\_new[:, 1], alpha=0.8)

plt.axis('equal');

**OUTPUT:**





