# Step 1: Import necessary libraries

import numpy as np

import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

# Step 2: Load the Iris dataset

iris = load\_iris()

iris\_data = iris.data

iris\_target = iris.target

iris\_target\_names = iris.target\_names

# Step 3: Standardize the data

scaler = StandardScaler()

iris\_data\_scaled = scaler.fit\_transform(iris\_data)

# Step 4: Perform PCA

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

iris\_pca = pca.fit\_transform(iris\_data\_scaled)

# Step 5: Visualize the results

plt.figure(figsize=(8, 6))

scatter = plt.scatter(iris\_pca[:, 0], iris\_pca[:, 1], c=iris\_target, cmap='viridis', edgecolor='k', s=100)

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.title('PCA of Iris Dataset By211P002 Rohitkumar')

# Modified legend creation

# Instead of using scatter.legend\_elements(), manually create the legend handles

# using the unique target values and corresponding labels.

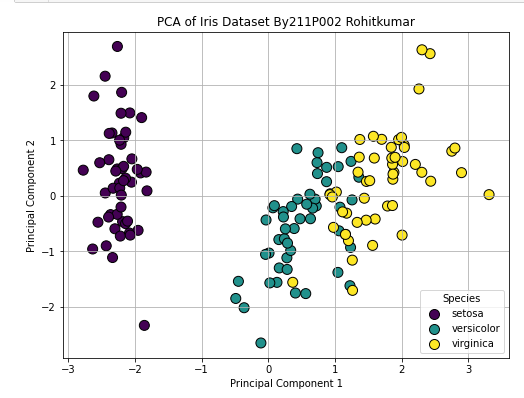
for target, target\_name in zip(np.unique(iris\_target), iris\_target\_names):

plt.scatter([], [], c=[plt.cm.viridis(target / 2)], label=target\_name, edgecolor='k', s=100)

plt.legend(title="Species")

plt.grid()

plt.show()



# Step 1: Import necessary libraries

import numpy as np

import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d import Axes3D

# Step 2: Load the Iris dataset

iris = load\_iris()

iris\_data = iris.data

iris\_target = iris.target

iris\_target\_names = iris.target\_names

# Step 3: Standardize the data

scaler = StandardScaler()

iris\_data\_scaled = scaler.fit\_transform(iris\_data)

# Step 4: Perform PCA

pca = PCA(n\_components=3) # Reduce to 3 principal components for more features

iris\_pca = pca.fit\_transform(iris\_data\_scaled)

# Step 5: Visualize the cumulative explained variance

explained\_variance = pca.explained\_variance\_ratio\_

cumulative\_explained\_variance = np.cumsum(explained\_variance)

plt.figure(figsize=(8, 5))

plt.plot(range(1, 4), cumulative\_explained\_variance, marker='o', linestyle='--', color='b')

plt.title('Cumulative Explained Variance by Principal Components\nRohitkumar Pandey 211P002')

plt.xlabel('Number of Principal Components')

plt.ylabel('Cumulative Explained Variance')

plt.grid()

plt.show()

# Step 6: 2D Visualization of the first two principal components

plt.figure(figsize=(10, 7))

scatter = plt.scatter(iris\_pca[:, 0], iris\_pca[:, 1], c=iris\_target, cmap='viridis', edgecolor='k', s=100)

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.title('2D PCA of Iris Dataset By Rohitkumar Pandey 211P002')

# Manually create legend handles for each target class

handles = []

for i in range(len(iris\_target\_names)):

handles.append(plt.scatter([], [], marker='o', s=100, edgecolor='k', c=[plt.cm.viridis(i / (len(iris\_target\_names) - 1))]))

# Use colormap to match scatter plot colors

plt.legend(handles, iris\_target\_names, title="Species") # Use the created handles

plt.grid()

plt.show()

# Step 7: 3D Visualization of the first three principal components

fig = plt.figure(figsize=(10, 7))

ax = fig.add\_subplot(111, projection='3d')

scatter = ax.scatter(iris\_pca[:, 0], iris\_pca[:, 1], iris\_pca[:, 2], c=iris\_target, cmap='viridis', edgecolor='k', s=100)

ax.set\_xlabel('Principal Component 1')

ax.set\_ylabel('Principal Component 2')

ax.set\_zlabel('Principal Component 3')

ax.set\_title('3D PCA of Iris Dataset ByRohitkumar Pandey 211P002')

# Manually create legend handles for each target class

handles\_3d = []

for i in range(len(iris\_target\_names)):

handles\_3d.append(ax.scatter([], [], [], marker='o', s=100, edgecolor='k', c=[plt.cm.viridis(i / (len(iris\_target\_names) - 1))]))

ax.legend(handles\_3d, iris\_target\_names, title="Species") # Use created handles 3D plot

plt.show()

# Step 8: Print explained variance ratio for each principal component

print("Explained variance ratio for each principal component:", explained\_variance)

