

Author: MANISH SINGH 1. Importing Required Libraries

In [1]: from sklearn import datasets import matplotlib.pyplot as plt import pandas as pd import numpy as np

from sklearn.cluster import KMeans import matplotlib.patches as mpatches import sklearn.metrics as sm from mpl_toolkits.mplot3d import Axes3D

from scipy.cluster.hierarchy import linkage,dendrogram from sklearn.cluster import DBSCAN from sklearn.decomposition import PCA import warnings

warnings.filterwarnings('ignore') print(" All required packages included successfully!") All required packages included successfully!

2 - Loading the dataset In [2]: iris = datasets.load_iris()

print(iris.data)

[[5.1 3.5 1.4 0.2] [4.9 3. 1.4 0.2] [4.7 3.2 1.3 0.2]

[4.6 3.1 1.5 0.2]

[5. 3.6 1.4 0.2] [5.4 3.9 1.7 0.4]

[4.6 3.4 1.4 0.3] [5. 3.4 1.5 0.2] [4.4 2.9 1.4 0.2]

[4.9 3.1 1.5 0.1] [5.4 3.7 1.5 0.2] [4.8 3.4 1.6 0.2] [4.8 3. 1.4 0.1]

[4.3 3. 1.1 0.1] [5.8 4. 1.2 0.2] [5.7 4.4 1.5 0.4] [5.4 3.9 1.3 0.4] [5.1 3.5 1.4 0.3]

[5.7 3.8 1.7 0.3] [5.1 3.8 1.5 0.3]

[5.4 3.4 1.7 0.2] [5.1 3.7 1.5 0.4] [4.6 3.6 1. 0.2] [5.1 3.3 1.7 0.5]

[5. 3.4 1.6 0.4] [5.2 3.5 1.5 0.2] [5.2 3.4 1.4 0.2] [4.7 3.2 1.6 0.2] [4.8 3.1 1.6 0.2]

[4.8 3.4 1.9 0.2] [5. 3. 1.6 0.2]

[5.4 3.4 1.5 0.4]

[5.2 4.1 1.5 0.1] [5.5 4.2 1.4 0.2] [4.9 3.1 1.5 0.2] [5. 3.2 1.2 0.2] [5.5 3.5 1.3 0.2] [4.9 3.6 1.4 0.1]

[4.4 3. 1.3 0.2] [5.1 3.4 1.5 0.2] [5. 3.5 1.3 0.3] [4.5 2.3 1.3 0.3] [4.4 3.2 1.3 0.2] [5. 3.5 1.6 0.6] [5.1 3.8 1.9 0.4]

[4.8 3. 1.4 0.3] [5.1 3.8 1.6 0.2] [4.6 3.2 1.4 0.2] [5.3 3.7 1.5 0.2] [5. 3.3 1.4 0.2] [7. 3.2 4.7 1.4] [6.4 3.2 4.5 1.5]

[6.3 3.3 4.7 1.6] [4.9 2.4 3.3 1.] [6.6 2.9 4.6 1.3] [5.2 2.7 3.9 1.4] [5. 2. 3.5 1.] [5.9 3. 4.2 1.5]

[6.9 3.1 4.9 1.5] [5.5 2.3 4. 1.3] [6.5 2.8 4.6 1.5] [5.7 2.8 4.5 1.3]

[6. 2.2 4. 1.]

[6.2 2.2 4.5 1.5] [5.6 2.5 3.9 1.1] [5.9 3.2 4.8 1.8]

[6.7 3. 5. 1.7] [6. 2.9 4.5 1.5] [5.7 2.6 3.5 1.]

[5.6 3. 4.1 1.3] [5.5 2.5 4. 1.3] [5.5 2.6 4.4 1.2]

[5.8 2.8 5.1 2.4] [6.4 3.2 5.3 2.3] [6.5 3. 5.5 1.8] [7.7 3.8 6.7 2.2] [7.7 2.6 6.9 2.3] [6. 2.2 5. 1.5] [6.9 3.2 5.7 2.3] [5.6 2.8 4.9 2.] [7.7 2.8 6.7 2.] [6.3 2.7 4.9 1.8] [6.7 3.3 5.7 2.1] [7.2 3.2 6. 1.8] [6.2 2.8 4.8 1.8] [6.1 3. 4.9 1.8] [6.4 2.8 5.6 2.1] [7.2 3. 5.8 1.6] [7.4 2.8 6.1 1.9] [7.9 3.8 6.4 2.] [6.4 2.8 5.6 2.2] [6.3 2.8 5.1 1.5] [6.1 2.6 5.6 1.4] [7.7 3. 6.1 2.3] [6.3 3.4 5.6 2.4] [6.4 3.1 5.5 1.8] [6. 3. 4.8 1.8] [6.9 3.1 5.4 2.1] [6.7 3.1 5.6 2.4] [6.9 3.1 5.1 2.3] [5.8 2.7 5.1 1.9] [6.8 3.2 5.9 2.3] [6.7 3.3 5.7 2.5] [6.7 3. 5.2 2.3] [6.3 2.5 5. 1.9] [6.5 3. 5.2 2.] [6.2 3.4 5.4 2.3] [5.9 3. 5.1 1.8]]

print(iris.target_names)

print(iris.target)

2 2]

x = iris.datay = iris.target

#Plotting

plt.show()

 $max_d=7.08$

dendrogram(hier,

plt.show()

2.0

30

25

20

15

10

#Hierachy Clustering hier=linkage(x, "ward")

plt.xlabel('Species') plt.ylabel('distance')

['setosa' 'versicolor' 'virginica']

fig = plt.figure(1, figsize=(7,5))

ax.set_xlabel("Petal width") ax.set_ylabel("Sepal length") ax.set_zlabel("Petal length")

plt.figure(figsize=(20,10))

truncate_mode='lastp',

leaf_rotation=90., leaf_font_size=8.,

plt.axhline(y=max_d, c='k')

2 - Visualizing the input data and its Hierarchy

plt.title("Iris Clustering K Means=3", fontsize=16)

plt.title('Iris Hierarchical Clustering Dendrogram')

Iris Clustering K Means=3

ax = Axes3D(fig, rect=[0, 0, 0.95, 1], elev=48, azim=134)ax.scatter(x[:, 3], x[:, 0], x[:, 2], edgecolor="k", s=50)

[6.1 2.9 4.7 1.4] [5.6 2.9 3.6 1.3] [6.7 3.1 4.4 1.4] [5.6 3. 4.5 1.5] [5.8 2.7 4.1 1.]

[6.1 2.8 4. 1.3] [6.3 2.5 4.9 1.5] [6.1 2.8 4.7 1.2] [6.4 2.9 4.3 1.3] [6.6 3. 4.4 1.4] [6.8 2.8 4.8 1.4]

[5.5 2.4 3.8 1.1] [5.5 2.4 3.7 1.] [5.8 2.7 3.9 1.2] [6. 2.7 5.1 1.6] [5.4 3. 4.5 1.5] [6. 3.4 4.5 1.6] [6.7 3.1 4.7 1.5] [6.3 2.3 4.4 1.3]

[6.1 3. 4.6 1.4] [5.8 2.6 4. 1.2] [5. 2.3 3.3 1.] [5.6 2.7 4.2 1.3] [5.7 3. 4.2 1.2] [5.7 2.9 4.2 1.3] [6.2 2.9 4.3 1.3]

[5.1 2.5 3. 1.1] [5.7 2.8 4.1 1.3] [6.3 3.3 6. 2.5] [5.8 2.7 5.1 1.9] [7.1 3. 5.9 2.1] [6.3 2.9 5.6 1.8] [6.5 3. 5.8 2.2]

[7.6 3. 6.6 2.1] [4.9 2.5 4.5 1.7] [7.3 2.9 6.3 1.8] [6.7 2.5 5.8 1.8] [7.2 3.6 6.1 2.5] [6.5 3.2 5.1 2.] [6.4 2.7 5.3 1.9] [6.8 3. 5.5 2.1] [5.7 2.5 5. 2.]

In [5]: In [6]:

In [4]:

Out[14]: In [15] I was able to successfully carry-out prediction using Unsupervised Machine Learning task and was able to evaluate the model's clustering accuracy score. THANK YOU SO MUCH!!

(2) 3: Data Preprocessing In [7]: x = pd.DataFrame(iris.data, columns=['Sepal Length', 'Sepal Width', 'Petal Length', 'Petal Width']) y = pd.DataFrame(iris.target, columns=['Target']) In [8]: x.head() Sepal Length Sepal Width Petal Length Petal Width Out[8]: 5.1 3.5 0.2 4.9 3.0 0.2 4.7 1.3 0.2 3.2 1.5 4.6 3.1 0.2

Iris Hierarchical Clustering Dendrogram

0 4: Model Training iris_k_mean_model = KMeans(n_clusters=3) iris_k_mean_model.fit(x) KMeans(n_clusters=3) print(iris_k_mean_model.labels_) 0 2] In [12]: print(iris_k_mean_model.cluster_centers_)

In [13]: patches = [mpatches.Patch(color=colors[i], label="{:s}".format(texts[i])) for i in range(len(texts))] plt.figure(figsize=(14,6)) colors = np.array(['red', 'green', 'blue'])

plt.subplot(1, 2, 1)

In [9]:

Out[9]:

y.head()

0

0

[[6.85 3.07368421 5.74210526 2.07105263] [5.006 3.428 1.462 0.246 [5.9016129 2.7483871 4.39354839 1.43387097]] 5 : Visualizing the Model Cluster import matplotlib.patches as mpatches colors = ["r", "g", "b"] texts = ["Setosa", "Versicolor", "Virginica"]

predictedY = np.choose(iris_k_mean_model.labels_, [1, 0, 2]).astype(np.int64)

plt.scatter(x['Petal Length'], x['Petal Width'], c=colors[y['Target']])

plt.title('Before classification') #plt.legend(handles=[red_patch, green_patch, blue_patch]) #plt.legend(handles= patches) plt.legend(handles=patches, loc='best', ncol=1) #plt.legend(facecolor="plum") plt.subplot(1, 2, 2) plt.scatter(x['Petal Length'], x['Petal Width'], c=colors[predictedY]) plt.title("Model's classification") #plt.legend(handles= patches) plt.legend(handles=patches, loc=2, ncol=1) plt.show() Before classification Versicolor Virginica 2.0 2.0 1.5 1.5 1.0 1.0

Versicolor Virginica 6 - Calculating the Accuracy and Confusion Matrix sm.accuracy_score(predictedY, y['Target']) sm.confusion_matrix(predictedY, y['Target']) array([[50, 0, 0], [0, 2, 36], [0, 48, 14]], dtype=int64) 7 - Conclusion

Model's classification