```
 1 import numpy as np
 2 import pandas as pd
 3 import matplotlib.pyplot as plt
 4 import seaborn as sns
 5 from scipy.stats import sem
 6 from sklearn import preprocessing
 7 from sklearn.decomposition import PCA
 8
 9 # load the dataset(sl = sepal length, sw = sepal width, pl =
   petal length, pw = petal width,class = Target)
10 df = pd.read_csv('C:\\Users\\Authorized User1\\Desktop\\Data
   mining\\assignment 2\\iris.txt',
11                  names=["sl","sw","pl","pw","class"])
12 print(df.info())
13 print (df.head())
14
15 # 2D scatter plots of the four features
16 sns.set_theme(style="darkgrid")
17 sns.pairplot(df,hue='class',diag_kind="kde",markers=['o','v','s
   '])
18 plt.show()
19
20 # 3D scatter plot of three features: sepal length, sepal width
   , petal width.
21 fig = plt.figure()
22 ax = fig.add_subplot(projection='3d')
23 ax.scatter(df['sl'], df['sw'], df['pw'], color = 'green',s = 20
   ,depthshade=True,marker='o')
24 plt.title("3D scatter plot")
25 ax.set_xlabel('Sepal Length')
26 ax.set_ylabel('Sepal Width')
27 ax.set_zlabel('Petal Width')
28 plt.show()
29
30 # Visualization of the feature matrix (column 1-4) as an image.
31 dfd = df.drop('class',axis=1)
32 plt.imshow(np.asmatrix(dfd), aspect = 'auto')
33 plt.show()
34
35 # For each class, generate histograms for the four features.
36 fig, ax = plt.subplots(2,2)
37 sns.histplot(ax=ax[0,0],data= df,x='sl',hue='class',bins= 20,
   alpha = 0.3)
38 sns.histplot(ax=ax[0,1],data= df,x='sw',hue='class',bins= 20,
   alpha = 0.3)
39 sns.histplot(ax=ax[1,0],data= df,x='pl',hue='class',bins= 20,
   alpha = 0.3)
40 sns.histplot(ax=ax[1,1],data= df,x='pw',hue='class',bins= 20,
   alpha = 0.3)
41 plt.show()
42
43 #For each class, generate boxplots of the four features.
```

```
44 fig, ax = plt.subplots(2,2)
45 sns.boxplot(ax=ax[0,0],data= df,x = 'class',y='sl')
46 sns.boxplot(ax=ax[0,1],data= df,y='sw',x='class')
47 sns.boxplot(ax=ax[1,0],data= df,y='pl',x='class')
48 sns.boxplot(ax=ax[1,1],data= df,y='pw',x='class')
49 plt.show()
50
51 #Calculate the correlation matrix of the four features
52 print (df.corr())
53
54 #Visualize the correlation matrix as an image.
55 sns.heatmap(df.corr())
56 plt.show()
57
58 #Create a parallel coordinate plot of the four features.
59 plt.style.context(("ggplot", "seaborn"))
60 pd.plotting.parallel_coordinates(df, 'class', color=('red', '
   blue', 'black'),alpha = 0.5)
61 plt.title("parallel coordinate plot")
62 plt.legend()
63 plt.show()
64
65
66
67
68 #Make a function for Minkowski Distance.
69 def minkowski(x,y,r):
70     a = []
71     for (i, j, k, l) in zip(df[x[0]], df[x[1]], df[x[2]], df[x[
   3]]):
72         a.append((((abs(i - y[0])) ** r) + ((abs(j - y[1])) **
   r) + ((abs(k - y[2])) ** r) + ((abs(l - y[3])) ** r)) ** ( 1 /
   r))
73     return a
74
75
76
77
78 #Make a function for T-statistics Distance.
79 dt = pd.read_csv('C:\\Users\\Authorized User1\\Desktop\\Data
   mining\\assignment 2\\hw2_ts.txt',
80                  names=["c1","c2"])
81 def ttest(d1, d2):
82     return ((np.mean(d1) - np.mean(d2)) / (np.sqrt((sem(d1))**2
    + sem(d2)**2)))
83
84
85
86
87 #Make a function for Mahalanobis Distance
88 def mahalnobis(x,y,m):
89     a = []
```

```
 90     for (i, j, k, l) in zip(df[x[0]], df[x[1]], df[x[2]], df[x
    [3]]):
 91         x = np.array([i, j, k, l])
 92         y = np.array(y)
 93         v = np.array([(x[0] - y[0]), (x[1] - y[1]), (x[2] - y[
    2]), (x[3] - y[3])])
 94         a.append(np.sqrt(np.dot(np.dot(v, m), v.transpose())))
 95     return a
 96
 97 #Make a function for Minkowski Distance and plot.
 98 def minkowski(x,y,r):
 99     a = []
100     for (i, j, k, l) in zip(df[x[0]], df[x[1]], df[x[2]], df[x
    [3]]):
101         a.append((((abs(i - y[0])) ** r) + ((abs(j - y[1
    ])) ** r) + ((abs(k - y[2])) ** r) + ((abs(l - y[3])) ** r
    )) ** ( 1 / r))
102     return a
103
104 q = [5.0000, 3.5000, 1.4600, 0.2540]
105 p= ['sl','sw','pl','pw']
106 # r = 1
107 sns.scatterplot(range(len(minkowski(p,q,1))),minkowski(p,q,1),
    hue=df['class'])
108 plt.xlabel('Number')
109 plt.ylabel('Minkowski distance for (r = 1)')
110 plt.show()
111 # r = 2
112 sns.scatterplot(range(len(minkowski(p,q,2))),minkowski(p,q,2),
    hue=df['class'])
113 plt.xlabel('Number')
114 plt.ylabel('Minkowski distance for (r = 2)')
115 plt.show()
116 # r = 10
117 sns.scatterplot(range(len(minkowski(p,q,10))),minkowski(p,q,10
    ),hue=df['class'])
118 plt.xlabel('Number')
119 plt.ylabel('Minkowski distance for (r = 10)')
120 plt.show()
121
122
123 #Calculate Mahalanobis distances and plot
124 r = np.linalg.inv(np.cov(np.matrix([df['sl'],df['sw'],df['pl'
    ],df['pw']])))
125 p = ['sl','sw','pl','pw']
126 q = [5.0000, 3.5000, 1.4600, 0.2540]
127
128 sns.scatterplot(range(len(mahalnobis(p,q,r))),(mahalnobis(p,q,
    r)),hue=df['class'])
129 plt.xlabel('Number')
130 plt.ylabel('Mahalonobis distance')
131 plt.show()
```

```
132
133 #Normalize the feature matrix of the IRIS dataset
134 normalize = df.drop('class',axis=1)
135 normalize = (((normalize-(np.mean(normalize)))/np.std(
    normalize)))
136 print (normalize)
137
138 #Calculate the correlation matrix
139 print (normalize.corr())
140
141 # load the dataset
142 df = pd.read_csv('C:\\Users\\Authorized User1\\Desktop\\Data
    mining\\assignment 2\\iris.txt',
143                  names=["sl","sw","pl","pw","class"])
144 dr = df.drop('class',axis=1)
145 pca = PCA(n_components=4)
146 dr_fit = pca.fit(dr)
147 dr_transform = pca.transform(dr)
148 dr_df = pd.DataFrame(dr_transform, columns = ['p1','p2','p3','
    p4'])
149
150 #Create 2D scatter plots of each pair of the four components
151 sns.pairplot(dr_df)
152 plt.show()
153
154 #3D scatter plot of the first three components
155 fig = plt.figure()
156 ax = fig.add_subplot(projection='3d')
157 ax.scatter(dr_df['p1'], dr_df['p2'], dr_df['p3'], color = '
    green',s = 20,depthshade=True,marker='v',)
158 plt.title("3D scatter plot")
159 ax.set_xlabel('p1')
160 ax.set_ylabel('p2')
161 ax.set_zlabel('p3')
162 plt.show()
163
164 #Obtain the variance of each component and visualize in a
    figure plot.
165 print(pca.explained_variance_ratio_)
166 plt.plot(np.cumsum(pca.explained_variance_ratio_))
167 plt.xlabel('number of components')
168 plt.ylabel('cumulative explained variance')
169 plt.show()
170
171 #Calculate the correlation matrix of the four components
172 print (dr_df.corr())
173
174 #load the dataset
175 dt = pd.read_csv('C:\\Users\\Authorized User1\\Desktop\\Data
    mining\\assignment 2\\hw2_ts.txt',names=["c1","c2"])
176 print (dt.head())
177
```

```
178 #Visualize the two time series in one figure plot.
179 dt.plot()
180 plt.xlabel('number of observations')
181 plt.ylabel('values')
182 plt.show()
183
184 #Calculate the T-statistics distance
185 print (ttest(dt['c1'],dt['c2']))
186
187 #Calculate the correlation of the two time series
188 print (dt.corr())
189
190
```