Data Science and Machine Learning Internship
(HDLC Technologies)

Name: *B. Deva SriRama Sai Ganesh*
Email: bandarusaiganesh2003@gmail.com

Mini Project: Use KNN algorithm for the classification of Iris flower data set

Solutions:

Python code:

```
#Data pre-processing
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as mtp
import warnings
warnings.filterwarnings('ignore')

#load the dataSet
iris_dataSet = pd.read_csv('C:\\Users\\banda\\Downloads\\Iris dataset.csv')
print("iris_dataSet: \n")
print(iris_dataSet.to_string())

print("\n Displaying summary statistics: \n")
print(iris_dataSet.describe())

print("\n Displaying information about the dataset: \n")
print(iris_dataSet.info())

print("\n Visualizing data: \n")
sns.set_style('whitegrid')
sns.FacetGrid(iris_dataSet, hue = "species", height = 6).map(mtp.scatter,
'sepal_length', 'sepal_width').add_legend()
sns.pairplot(iris_dataSet, hue = 'species', height = 2).add_legend()
mtp.show()
```

```
#Extracting independent variable x and dependent variable y
X = iris_dataSet.iloc[:, :4].values
y = iris_dataSet.iloc[:, -1].values
print(f'Displaying first 5 values of independent variable x: \n {X[:5]}')
print(f'\n Displaying first 5 values of dependent variable y: \n{y[:5]}')

#Splitting the dataset into training and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, shuffle = True, random_state = 0)
print(f'\n Training set size: {X_train.shape[0]} samples \n Test set size: {X_test.shape[0]} samples')

#feature Scaling
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
print(f'\n Printing X_train: \n\n {X_train}')

#Fitting KNN classifier to the training set
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3, metric = 'minkowski', p = 2)
knn.fit(X_train, y_train)
print(f'\n Printing X_test: \n\n {X_test}')

#predicting the test set result
y_pred = knn.predict(X_test)
print(f'\n Printing y_pred: \n\n {y_pred}')

#Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
ac = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
print(f'\n Accuracy is {ac}')
print(f'\n Confusion matrix: \n {cm}')
```

In [1]:
```
#Data pre-processing
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as mtp
import warnings
warnings.filterwarnings('ignore')

#Load the dataSet
iris_dataSet = pd.read_csv('C:\\Users\\banda\\Downloads\\Iris dataset.csv')
print("iris_dataSet: \n")
print(iris_dataSet.to_string())
```

```
iris_dataSet:

     sepal_length  sepal_width  petal_length  petal_width      species
0             5.1          3.5           1.4          0.2       setosa
1             4.9          3.0           1.4          0.2       setosa
2             4.7          3.2           1.3          0.2       setosa
3             4.6          3.1           1.5          0.2       setosa
4             5.0          3.6           1.4          0.2       setosa
5             5.4          3.9           1.7          0.4       setosa
6             4.6          3.4           1.4          0.3       setosa
7             5.0          3.4           1.5          0.2       setosa
8             4.4          2.9           1.4          0.2       setosa
9             4.9          3.1           1.5          0.1       setosa
10            5.4          3.7           1.5          0.2       setosa
11            4.8          3.4           1.6          0.2       setosa
12            4.8          3.0           1.4          0.1       setosa
13            4.3          3.0           1.1          0.1       setosa
14            5.8          4.0           1.2          0.2       setosa
15            5.7          4.4           1.5          0.4       setosa
16            5.4          3.9           1.3          0.4       setosa
17            5.1          3.5           1.4          0.3       setosa
18            5.7          3.8           1.7          0.3       setosa
19            5.1          3.8           1.5          0.3       setosa
20            5.4          3.4           1.7          0.2       setosa
21            5.1          3.7           1.5          0.4       setosa
22            4.6          3.6           1.0          0.2       setosa
23            5.1          3.3           1.7          0.5       setosa
24            4.8          3.4           1.9          0.2       setosa
25            5.0          3.0           1.6          0.2       setosa
26            5.0          3.4           1.6          0.4       setosa
27            5.2          3.5           1.5          0.2       setosa
28            5.2          3.4           1.4          0.2       setosa
29            4.7          3.2           1.6          0.2       setosa
30            4.8          3.1           1.6          0.2       setosa
```

31	5.4	3.4	1.5	0.4	setosa
32	5.2	4.1	1.5	0.1	setosa
33	5.5	4.2	1.4	0.2	setosa
34	4.9	3.1	1.5	0.1	setosa
35	5.0	3.2	1.2	0.2	setosa
36	5.5	3.5	1.3	0.2	setosa
37	4.9	3.1	1.5	0.1	setosa
38	4.4	3.0	1.3	0.2	setosa
39	5.1	3.4	1.5	0.2	setosa
40	5.0	3.5	1.3	0.3	setosa
41	4.5	2.3	1.3	0.3	setosa
42	4.4	3.2	1.3	0.2	setosa
43	5.0	3.5	1.6	0.6	setosa
44	5.1	3.8	1.9	0.4	setosa
45	4.8	3.0	1.4	0.3	setosa
46	5.1	3.8	1.6	0.2	setosa
47	4.6	3.2	1.4	0.2	setosa
48	5.3	3.7	1.5	0.2	setosa
49	5.0	3.3	1.4	0.2	setosa
50	7.0	3.2	4.7	1.4	versicolor
51	6.4	3.2	4.5	1.5	versicolor
52	6.9	3.1	4.9	1.5	versicolor
53	5.5	2.3	4.0	1.3	versicolor
54	6.5	2.8	4.6	1.5	versicolor
55	5.7	2.8	4.5	1.3	versicolor
56	6.3	3.3	4.7	1.6	versicolor
57	4.9	2.4	3.3	1.0	versicolor
58	6.6	2.9	4.6	1.3	versicolor
59	5.2	2.7	3.9	1.4	versicolor
60	5.0	2.0	3.5	1.0	versicolor
61	5.9	3.0	4.2	1.5	versicolor
62	6.0	2.2	4.0	1.0	versicolor
63	6.1	2.9	4.7	1.4	versicolor
64	5.6	2.9	3.6	1.3	versicolor
65	6.7	3.1	4.4	1.4	versicolor
66	5.6	3.0	4.5	1.5	versicolor
67	5.8	2.7	4.1	1.0	versicolor
68	6.2	2.2	4.5	1.5	versicolor
69	5.6	2.5	3.9	1.1	versicolor
70	5.9	3.2	4.8	1.8	versicolor
71	6.1	2.8	4.0	1.3	versicolor
72	6.3	2.5	4.9	1.5	versicolor
73	6.1	2.8	4.7	1.2	versicolor
74	6.4	2.9	4.3	1.3	versicolor
75	6.6	3.0	4.4	1.4	versicolor
76	6.8	2.8	4.8	1.4	versicolor
77	6.7	3.0	5.0	1.7	versicolor
78	6.0	2.9	4.5	1.5	versicolor
79	5.7	2.6	3.5	1.0	versicolor
80	5.5	2.4	3.8	1.1	versicolor
81	5.5	2.4	3.7	1.0	versicolor
82	5.8	2.7	3.9	1.2	versicolor
83	6.0	2.7	5.1	1.6	versicolor
84	5.4	3.0	4.5	1.5	versicolor
85	6.0	3.4	4.5	1.6	versicolor
86	6.7	3.1	4.7	1.5	versicolor
87	6.3	2.3	4.4	1.3	versicolor
88	5.6	3.0	4.1	1.3	versicolor
89	5.5	2.5	4.0	1.3	versicolor

90	5.5	2.6	4.4	1.2	versicolor
91	6.1	3.0	4.6	1.4	versicolor
92	5.8	2.6	4.0	1.2	versicolor
93	5.0	2.3	3.3	1.0	versicolor
94	5.6	2.7	4.2	1.3	versicolor
95	5.7	3.0	4.2	1.2	versicolor
96	5.7	2.9	4.2	1.3	versicolor
97	6.2	2.9	4.3	1.3	versicolor
98	5.1	2.5	3.0	1.1	versicolor
99	5.7	2.8	4.1	1.3	versicolor
100	6.3	3.3	6.0	2.5	virginica
101	5.8	2.7	5.1	1.9	virginica
102	7.1	3.0	5.9	2.1	virginica
103	6.3	2.9	5.6	1.8	virginica
104	6.5	3.0	5.8	2.2	virginica
105	7.6	3.0	6.6	2.1	virginica
106	4.9	2.5	4.5	1.7	virginica
107	7.3	2.9	6.3	1.8	virginica
108	6.7	2.5	5.8	1.8	virginica
109	7.2	3.6	6.1	2.5	virginica
110	6.5	3.2	5.1	2.0	virginica
111	6.4	2.7	5.3	1.9	virginica
112	6.8	3.0	5.5	2.1	virginica
113	5.7	2.5	5.0	2.0	virginica
114	5.8	2.8	5.1	2.4	virginica
115	6.4	3.2	5.3	2.3	virginica
116	6.5	3.0	5.5	1.8	virginica
117	7.7	3.8	6.7	2.2	virginica
118	7.7	2.6	6.9	2.3	virginica
119	6.0	2.2	5.0	1.5	virginica
120	6.9	3.2	5.7	2.3	virginica
121	5.6	2.8	4.9	2.0	virginica
122	7.7	2.8	6.7	2.0	virginica
123	6.3	2.7	4.9	1.8	virginica
124	6.7	3.3	5.7	2.1	virginica
125	7.2	3.2	6.0	1.8	virginica
126	6.2	2.8	4.8	1.8	virginica
127	6.1	3.0	4.9	1.8	virginica
128	6.4	2.8	5.6	2.1	virginica
129	7.2	3.0	5.8	1.6	virginica
130	7.4	2.8	6.1	1.9	virginica
131	7.9	3.8	6.4	2.0	virginica
132	6.4	2.8	5.6	2.2	virginica
133	6.3	2.8	5.1	1.5	virginica
134	6.1	2.6	5.6	1.4	virginica
135	7.7	3.0	6.1	2.3	virginica
136	6.3	3.4	5.6	2.4	virginica
137	6.4	3.1	5.5	1.8	virginica
138	6.0	3.0	4.8	1.8	virginica
139	6.9	3.1	5.4	2.1	virginica
140	6.7	3.1	5.6	2.4	virginica
141	6.9	3.1	5.1	2.3	virginica
142	5.8	2.7	5.1	1.9	virginica
143	6.8	3.2	5.9	2.3	virginica
144	6.7	3.3	5.7	2.5	virginica
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

In [2]:
```
print("\n Displaying summary statistics: \n")
print(iris_dataSet.describe())
```

```
 Displaying summary statistics:

       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
```

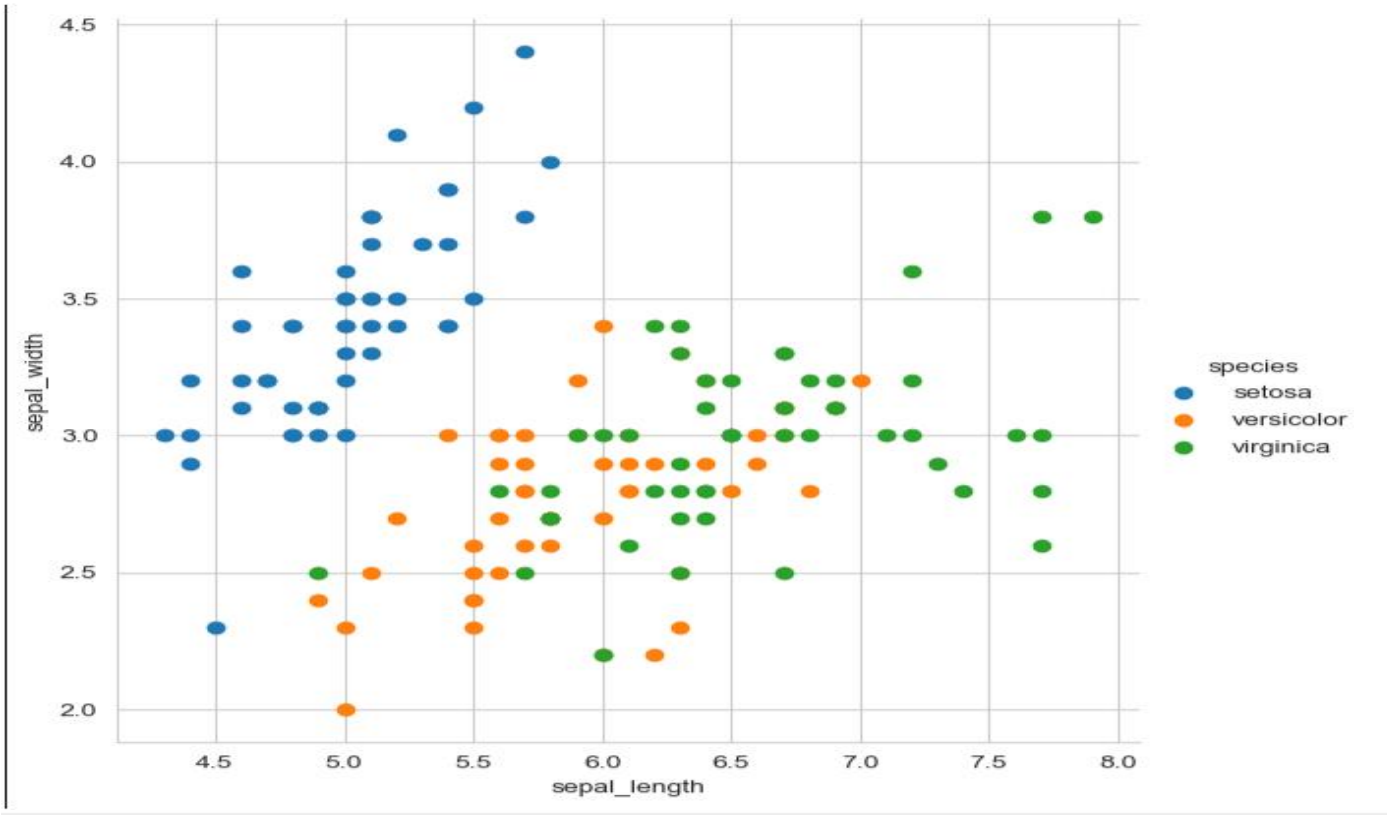
In [3]:
```
print("\n Displaying information about the dataset: \n")
print(iris_dataSet.info())
```

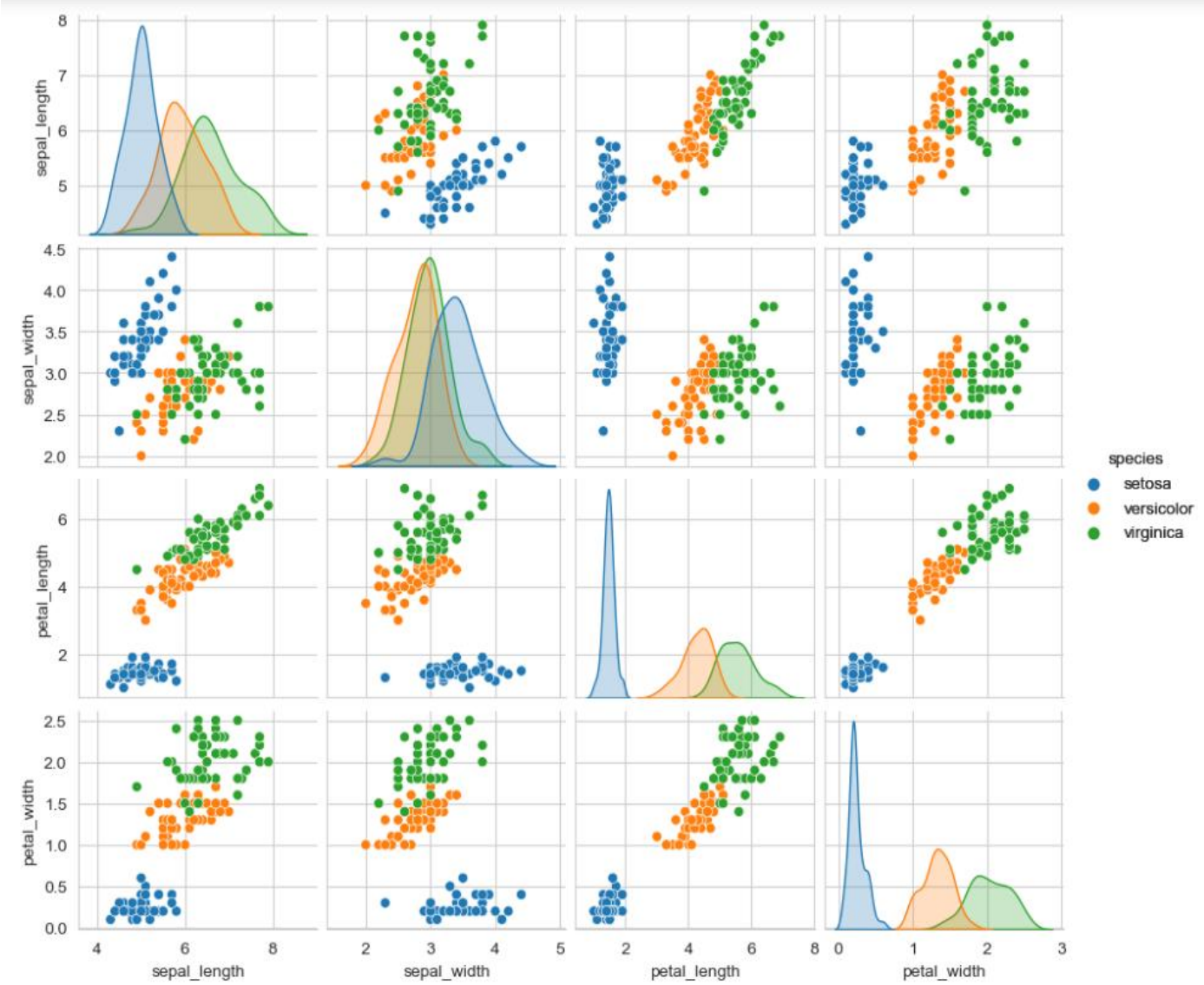
```
 Displaying information about the dataset:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
None
```

In [4]:
```
print("\n Visualizing data: \n")
sns.set_style('whitegrid')
sns.FacetGrid(iris_dataSet, hue = "species", height = 6).map(mtp.scatter, 'sepal_length', 'sepal_width').add_legend()
sns.pairplot(iris_dataSet, hue = 'species', height = 2).add_legend()
mtp.show()
```

Visualizing data:





In [5]:
```
#Extracting independent variable x and dependent variable y
X = iris_dataSet.iloc[:, :4].values
y = iris_dataSet.iloc[:, -1].values
print(f'Displaying first 5 values of independent variable x: \n {X[:5]}')
print(f'\n Displaying first 5 values of dependent variable y: \n{y[:5]}')
```

```
Displaying first 5 values of independent variable x:
 [[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]]

 Displaying first 5 values of dependent variable y:
['setosa' 'setosa' 'setosa' 'setosa' 'setosa']
```

In [6]:
```
#Splitting the dataset into training and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, shuffle = True, random_state = 0)
print(f'\n Training set size: {X_train.shape[0]} samples \n Test set size: {X_test.shape[0]} samples')
```

```
Training set size: 120 samples
Test set size: 30 samples
```

In [7]:
```
#feature Scaling
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
print(f'\n Printing X_train: \n\n {X_train}')
```

```
Printing X_train:

[[ 0.61303014  0.10850105  0.94751783  0.73603967]
[-0.56776627 -0.12400121  0.38491447  0.34808318]
[-0.80392556  1.03851009 -1.30289562 -1.3330616 ]
[ 0.25879121 -0.12400121  0.60995581  0.73603967]
[ 0.61303014 -0.58900572  1.00377816  1.25331499]
[-0.80392556 -0.82150798  0.04735245  0.21876435]
[-0.21352735  1.73601687 -1.19037495 -1.20374277]
[ 0.14071157 -0.82150798  0.72247648  0.47740201]
[ 0.02263193 -0.12400121  0.21613346  0.34808318]
[-0.09544771 -1.05401024  0.10361279 -0.03987331]
[ 1.0853487  -0.12400121  0.94751783  1.12399616]
[-1.39432376  0.34100331 -1.41541629 -1.3330616 ]
[ 1.20342834  0.10850105  0.72247648  1.38263382]
[-1.04008484  1.03851009 -1.24663528 -0.81578628]
[-0.56776627  1.50351461 -1.30289562 -1.3330616 ]
[-1.04008484 -2.4490238  -0.1776889  -0.29851096]
[ 0.73110978 -0.12400121  0.94751783  0.73603967]
[ 0.96726906  0.57350557  1.0600385   1.64127148]
[ 0.14071157 -1.98401928  0.66621615  0.34808318]
[ 0.96726906 -1.2865125   1.11629884  0.73603967]
[-0.33160699 -1.2865125   0.04735245 -0.16919214]
[ 2.14806547 -0.12400121  1.28507985  1.38263382]
[ 0.49495049  0.57350557  0.49743514  0.47740201]
[-0.44968663 -1.51901476 -0.00890789 -0.16919214]
[ 0.49495049 -0.82150798  0.60995581  0.73603967]
[ 0.49495049 -0.58900572  0.72247648  0.34808318]
[-1.15816448 -1.2865125   0.38491447  0.60672084]
[ 0.49495049 -1.2865125   0.66621615  0.8653585 ]
[ 1.32150798  0.34100331  0.49743514  0.21876435]
[ 0.73110978 -0.12400121  0.77873682  0.99467733]
[ 0.14071157  0.80600783  0.38491447  0.47740201]
[-1.27624412  0.10850105 -1.24663528 -1.3330616 ]
[-0.09544771 -0.82150798  0.72247648  0.8653585 ]
[-0.33160699 -0.82150798  0.21613346  0.08944552]
[-0.33160699 -0.35650346 -0.12142856  0.08944552]
[-0.44968663 -1.2865125   0.10361279  0.08944552]
[ 0.25879121 -0.12400121  0.4411748   0.21876435]
[ 1.55766726  0.34100331  1.22881951  0.73603967]
[-0.68584591  1.50351461 -1.30289562 -1.3330616 ]
[-1.86664232 -0.12400121 -1.52793696 -1.46238043]
[ 0.61303014 -0.82150798  0.83499716  0.8653585 ]
```

```
[-0.21352735 -0.12400121  0.21613346 -0.03987331]
[-0.56776627  0.80600783 -1.19037495 -1.3330616 ]
[-0.21352735  3.13103043 -1.30289562 -1.07442394]
[ 1.20342834  0.10850105  0.60995581  0.34808318]
[-1.5124034   0.10850105 -1.30289562 -1.3330616 ]
[ 0.02263193 -0.12400121  0.72247648  0.73603967]
[-0.9220052  -1.2865125  -0.45899058 -0.16919214]
[-1.5124034   0.80600783 -1.35915595 -1.20374277]
[ 0.37687085 -1.98401928  0.38491447  0.34808318]
[ 1.55766726  1.27101235  1.28507985  1.64127148]
[-0.21352735 -0.35650346  0.21613346  0.08944552]
[-1.27624412 -0.12400121 -1.35915595 -1.46238043]
[ 1.43958762 -0.12400121  1.17255917  1.12399616]
[ 1.20342834  0.34100331  1.0600385   1.38263382]
[ 0.73110978 -0.12400121  1.11629884  1.25331499]
[ 0.61303014 -0.58900572  1.00377816  1.12399616]
[-0.9220052   1.73601687 -1.24663528 -1.3330616 ]
[-1.27624412  0.80600783 -1.24663528 -1.3330616 ]
[ 0.73110978  0.34100331  0.72247648  0.99467733]
[ 0.96726906  0.57350557  1.0600385   1.12399616]
[-1.63048304 -1.75151702 -1.41541629 -1.20374277]
[ 0.37687085  0.80600783  0.89125749  1.38263382]
[-1.15816448 -0.12400121 -1.35915595 -1.3330616 ]
[-0.21352735 -1.2865125   0.66621615  0.99467733]
[ 1.20342834  0.10850105  0.89125749  1.12399616]
[-1.74856268  0.34100331 -1.41541629 -1.3330616 ]
[-1.04008484  1.27101235 -1.35915595 -1.3330616 ]
[ 1.55766726 -0.12400121  1.11629884  0.47740201]
[-0.9220052   1.03851009 -1.35915595 -1.20374277]
[-1.74856268 -0.12400121 -1.41541629 -1.3330616 ]
[-0.56776627  1.96851913 -1.19037495 -1.07442394]
[-0.44968663 -1.75151702  0.10361279  0.08944552]
[ 1.0853487   0.34100331  1.17255917  1.38263382]
[ 2.02998583 -0.12400121  1.56638153  1.12399616]
[-0.9220052   1.03851009 -1.35915595 -1.3330616 ]
[-1.15816448  0.10850105 -1.30289562 -1.46238043]
[-0.80392556  0.80600783 -1.35915595 -1.3330616 ]
[-0.21352735 -0.58900572  0.38491447  0.08944552]
[ 0.84918942 -0.12400121  0.32865413  0.21876435]
[-1.04008484  0.34100331 -1.47167663 -1.3330616 ]
```

```
[-0.9220052   0.57350557 -1.19037495 -0.94510511]
 [ 0.61303014 -0.35650346  0.27239379  0.08944552]
 [-0.56776627  0.80600783 -1.30289562 -1.07442394]
 [ 2.14806547 -1.05401024  1.73516253  1.38263382]
 [-1.15816448 -1.51901476 -0.29020957 -0.29851096]
 [ 2.38422475  1.73601687  1.45386085  0.99467733]
 [ 0.96726906  0.10850105  0.32865413  0.21876435]
 [-0.80392556  2.43352365 -1.30289562 -1.46238043]
 [ 0.14071157 -0.12400121  0.55369548  0.73603967]
 [-0.09544771  2.20102139 -1.47167663 -1.3330616 ]
 [ 2.14806547 -0.58900572  1.62264186  0.99467733]
 [-0.9220052   1.73601687 -1.30289562 -1.20374277]
 [-1.39432376  0.34100331 -1.24663528 -1.3330616 ]
 [ 1.79382654 -0.58900572  1.28507985  0.8653585 ]
 [-1.04008484  0.57350557 -1.35915595 -1.3330616 ]
 [ 0.49495049  0.80600783  1.00377816  1.51195265]
 [-0.21352735 -0.58900572  0.15987312  0.08944552]
 [-0.09544771 -0.82150798  0.04735245 -0.03987331]
 [-0.21352735 -1.05401024 -0.1776889  -0.29851096]
 [ 0.61303014  0.34100331  0.83499716  1.38263382]
 [ 0.96726906 -0.12400121  0.77873682  1.38263382]
 [ 0.49495049 -1.2865125   0.60995581  0.34808318]
 [ 0.96726906 -0.12400121  0.66621615  0.60672084]
 [-1.04008484 -0.12400121 -1.24663528 -1.3330616 ]
 [-0.44968663 -1.51901476 -0.06516822 -0.29851096]
 [ 0.96726906  0.10850105  1.00377816  1.51195265]
 [-0.09544771 -0.82150798  0.72247648  0.8653585 ]
 [-0.9220052   0.80600783 -1.30289562 -1.3330616 ]
 [ 0.84918942 -0.35650346  0.4411748   0.08944552]
 [-0.33160699 -0.12400121  0.15987312  0.08944552]
 [ 0.02263193  0.34100331  0.55369548  0.73603967]
 [ 0.49495049 -1.75151702  0.32865413  0.08944552]
 [-0.44968663  1.03851009 -1.41541629 -1.3330616 ]
 [-0.9220052   1.50351461 -1.30289562 -1.07442394]
 [-1.15816448  0.10850105 -1.30289562 -1.46238043]
 [ 0.49495049 -0.35650346  1.00377816  0.73603967]
 [-0.09544771 -0.82150798  0.15987312 -0.29851096]
 [ 2.14806547  1.73601687  1.62264186  1.25331499]
 [-1.5124034   0.34100331 -1.35915595 -1.3330616 ]]
```

In [8]:
```
#Fitting KNN classifier to the training set
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3, metric = 'minkowski', p = 2)
knn.fit(X_train, y_train)
print(f'\n Printing X_test: \n\n {X_test}')
```

```
 Printing X_test:

[[-0.09544771 -0.58900572  0.72247648  1.51195265]
 [ 0.14071157 -1.98401928  0.10361279 -0.29851096]
 [-0.44968663  2.66602591 -1.35915595 -1.3330616 ]
 [ 1.6757469  -0.35650346  1.39760052  0.73603967]
 [-1.04008484  0.80600783 -1.30289562 -1.3330616 ]
 [ 0.49495049  0.57350557  1.22881951  1.64127148]
 [-1.04008484  1.03851009 -1.41541629 -1.20374277]
 [ 0.96726906  0.10850105  0.49743514  0.34808318]
 [ 1.0853487  -0.58900572  0.55369548  0.21876435]
 [ 0.25879121 -0.58900572  0.10361279  0.08944552]
 [ 0.25879121 -1.05401024  1.00377816  0.21876435]
 [ 0.61303014  0.34100331  0.38491447  0.34808318]
 [ 0.25879121 -0.58900572  0.49743514 -0.03987331]
 [ 0.73110978 -0.58900572  0.4411748   0.34808318]
 [ 0.25879121 -0.35650346  0.49743514  0.21876435]
 [-1.15816448  0.10850105 -1.30289562 -1.46238043]
 [ 0.14071157 -0.35650346  0.38491447  0.34808318]
 [-0.44968663 -1.05401024  0.32865413 -0.03987331]
 [-1.27624412 -0.12400121 -1.35915595 -1.20374277]
 [-0.56776627  1.96851913 -1.41541629 -1.07442394]
 [-0.33160699 -0.58900572  0.60995581  0.99467733]
 [-0.33160699 -0.12400121  0.38491447  0.34808318]
 [-1.27624412  0.80600783 -1.07785427 -1.3330616 ]
 [-1.74856268 -0.35650346 -1.35915595 -1.3330616 ]
 [ 0.37687085 -0.58900572  0.55369548  0.73603967]
 [-1.5124034   1.27101235 -1.5841973  -1.3330616 ]
 [-0.9220052   1.73601687 -1.07785427 -1.07442394]
 [ 0.37687085 -0.35650346  0.27239379  0.08944552]
 [-1.04008484 -1.75151702 -0.29020957 -0.29851096]
 [-1.04008484  0.80600783 -1.24663528 -1.07442394]]
```

In [9]:
```
#predicting the test set result
y_pred = knn.predict(X_test)
print(f'\n Printing y_pred: \n\n {y_pred}')
```

```
Printing y_pred:

['virginica' 'versicolor' 'setosa' 'virginica' 'setosa' 'virginica'
'setosa' 'versicolor' 'versicolor' 'versicolor' 'versicolor' 'versicolor'
'versicolor' 'versicolor' 'versicolor' 'setosa' 'versicolor' 'versicolor'
'setosa' 'setosa' 'virginica' 'versicolor' 'setosa' 'setosa' 'virginica'
'setosa' 'setosa' 'versicolor' 'versicolor' 'setosa']
```

In [10]:
```
#Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
ac = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
print(f'\n Accuracy is {ac}')
print(f'\n Confusion matrix: \n {cm}')
```

```
Accuracy is 0.9666666666666667

Confusion matrix:
[[11  0  0]
 [ 0 13  0]
 [ 0  1  5]]
```