ml-lab2-8thfeb-2

February 8, 2024

[1]:
```
import pandas as pd
from sklearn.datasets import load_iris
iris = load_iris()
```

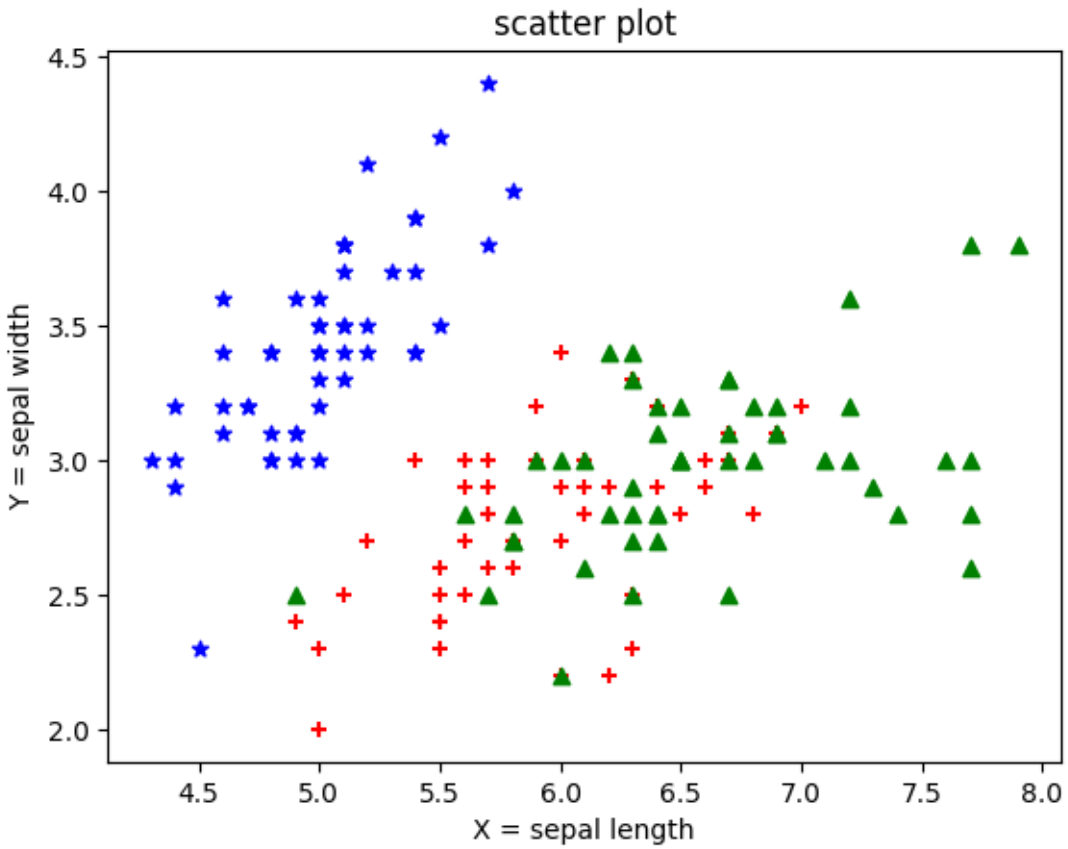
[2]:
```
iris.target_names
df = pd.DataFrame(iris.data , columns=iris.feature_names)
df_target = pd.DataFrame(iris.target_names)
print("Features: \n", df.head())

print(" \nIris Dataset: \n", df_target)
```

```
Features:
    sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

Iris Dataset:
             0
0       setosa
1   versicolor
2    virginica
```

[3]:
```
import matplotlib.pyplot as plt
df0=df[:50]
df1=df[50:100]
df2=df[100:150]

plt.title("scatter plot")
plt.xlabel('X = sepal length')
plt.ylabel('Y = sepal width')
plt.scatter(df0['sepal length (cm)'],df0['sepal width (cm)'], color = 'blue',␣
 ↪marker='*')
plt.scatter(df1['sepal length (cm)'],df1['sepal width (cm)'], color = "red",␣
 ↪marker = "+")
```

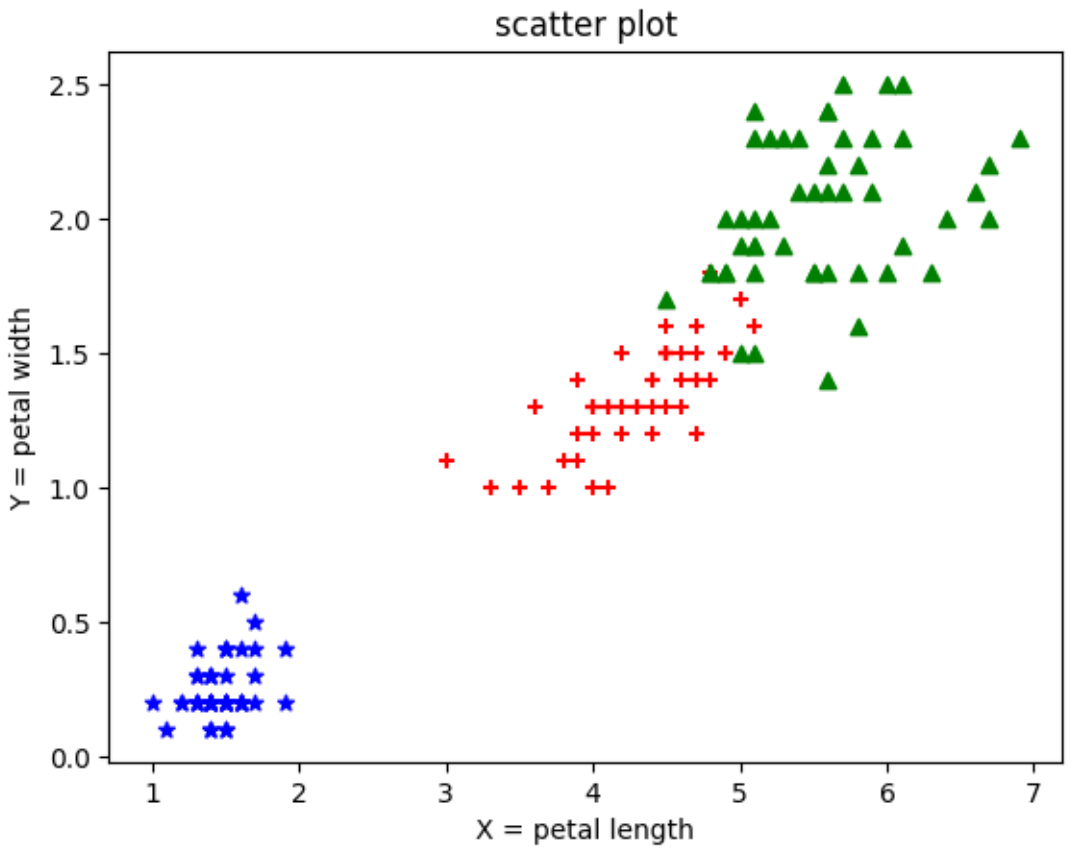
```
plt.scatter(df2['sepal length (cm)'],df2['sepal width (cm)'],color = "green",␣
 ↪marker ="^")
```

[3]: <matplotlib.collections.PathCollection at 0x7fab95aab070>



[4]:
```
import matplotlib.pyplot as plt
df0=df[:50]
df1=df[50:100]
df2=df[100:150]

plt.title("scatter plot")
plt.xlabel('X = petal length')
plt.ylabel('Y = petal width')
plt.scatter(df0['petal length (cm)'],df0['petal width (cm)'], color = 'blue',␣
 ↪marker='*')
plt.scatter(df1['petal length (cm)'],df1['petal width (cm)'], color = "red",␣
 ↪marker = "+")
plt.scatter(df2['petal length (cm)'],df2['petal width (cm)'],color = "green",␣
 ↪marker ="^")
```

[4]: <matplotlib.collections.PathCollection at 0x7fab9594f100>



[5]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
# Load the Iris dataset
X = df
y = iris.target
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,␣
 ↪random_state=42)

print(X_train)
print(X_test)
print(y_train)
print(y_test)
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
96	5.7	2.9	4.2	1.3
105	7.6	3.0	6.6	2.1
66	5.6	3.0	4.5	1.5

```
0                  5.1               3.5                1.4               0.2
122                7.7               2.8                6.7               2.0
..                 …                 …                  …                 …
71                 6.1               2.8                4.0               1.3
106                4.9               2.5                4.5               1.7
14                 5.8               4.0                1.2               0.2
92                 5.8               2.6                4.0               1.2
102                7.1               3.0                5.9               2.1

[100 rows x 4 columns]
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
73                 6.1               2.8                4.7               1.2
18                 5.7               3.8                1.7               0.3
118                7.7               2.6                6.9               2.3
78                 6.0               2.9                4.5               1.5
76                 6.8               2.8                4.8               1.4
31                 5.4               3.4                1.5               0.4
64                 5.6               2.9                3.6               1.3
141                6.9               3.1                5.1               2.3
68                 6.2               2.2                4.5               1.5
82                 5.8               2.7                3.9               1.2
110                6.5               3.2                5.1               2.0
12                 4.8               3.0                1.4               0.1
36                 5.5               3.5                1.3               0.2
9                  4.9               3.1                1.5               0.1
19                 5.1               3.8                1.5               0.3
56                 6.3               3.3                4.7               1.6
104                6.5               3.0                5.8               2.2
69                 5.6               2.5                3.9               1.1
55                 5.7               2.8                4.5               1.3
132                6.4               2.8                5.6               2.2
29                 4.7               3.2                1.6               0.2
127                6.1               3.0                4.9               1.8
26                 5.0               3.4                1.6               0.4
128                6.4               2.8                5.6               2.1
131                7.9               3.8                6.4               2.0
145                6.7               3.0                5.2               2.3
108                6.7               2.5                5.8               1.8
143                6.8               3.2                5.9               2.3
45                 4.8               3.0                1.4               0.3
30                 4.8               3.1                1.6               0.2
22                 4.6               3.6                1.0               0.2
15                 5.7               4.4                1.5               0.4
65                 6.7               3.1                4.4               1.4
11                 4.8               3.4                1.6               0.2
42                 4.4               3.2                1.3               0.2
146                6.3               2.5                5.0               1.9
51                 6.4               3.2                4.5               1.5
```

```
27                5.2               3.5               1.5               0.2
4                 5.0               3.6               1.4               0.2
32                5.2               4.1               1.5               0.1
142               5.8               2.7               5.1               1.9
85                6.0               3.4               4.5               1.6
86                6.7               3.1               4.7               1.5
16                5.4               3.9               1.3               0.4
10                5.4               3.7               1.5               0.2
81                5.5               2.4               3.7               1.0
133               6.3               2.8               5.1               1.5
137               6.4               3.1               5.5               1.8
75                6.6               3.0               4.4               1.4
109               7.2               3.6               6.1               2.5
[1 2 1 0 2 1 0 0 0 1 2 0 0 0 1 0 1 2 0 1 2 0 2 2 1 1 2 1 0 1 2 0 0 1 1 0 2
 0 0 1 1 2 1 2 2 1 0 0 2 2 0 0 0 1 2 0 2 2 0 1 1 2 1 2 0 2 1 2 1 1 1 0 1 1
 0 1 2 2 0 1 2 2 0 2 0 1 2 2 1 2 1 1 2 2 0 1 2 0 1 2]
[1 0 2 1 1 0 1 2 1 1 2 0 0 0 0 1 2 1 1 2 0 2 0 2 2 2 2 2 2 0 0 0 0 1 0 0 2 1
 0 0 0 2 1 1 0 0 1 2 2 1 2]
```

[6]:
```
# Initialize and train the KNN classifier
#importing the knn model
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=10)
```

[7]:
```
#fitting the model
neigh.fit(X_train, y_train)
```

[7]:
```
KNeighborsClassifier(n_neighbors=10)
```

[8]:
```
#Seeing the score of the model
neigh.score(X_test,y_test)
```

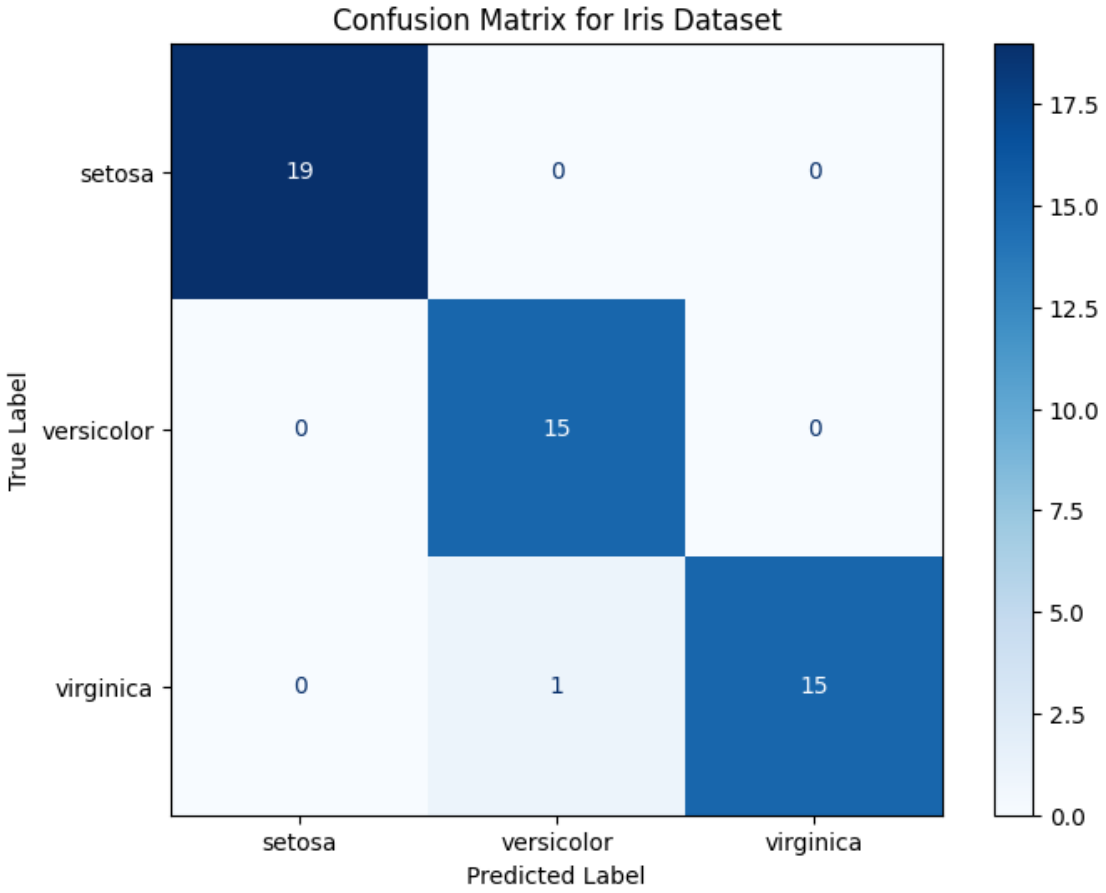
[8]:
```
0.98
```

[9]:
```
#predicting the model
neigh.predict(X_test)
```

[9]:
```
array([1, 0, 2, 1, 1, 0, 1, 2, 1, 1, 2, 0, 0, 0, 0, 1, 2, 1, 1, 2, 0, 2,
       0, 2, 2, 2, 2, 2, 0, 0, 0, 0, 1, 0, 0, 2, 1, 0, 0, 0, 2, 1, 1, 0,
       0, 1, 1, 2, 1, 2])
```

[12]:
```
#predicting and generating the confusion matrix
from sklearn.metrics import confusion_matrix
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
y_pred = neigh.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
```

```
cm
# Plot confusion matrix
fig, ax = plt.subplots(figsize=(8, 6))
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=iris.
 ↪target_names)
disp.plot(ax=ax, cmap=plt.cm.Blues)
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix for Iris Dataset')
plt.show()
```



[ ]:
```
# Generate and print the classification report

from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
```

```
print("Classification Report:")
print(classification_report(y_test, y_pred, target_names=iris.target_names))
```

```
Classification Report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        19
  versicolor       0.94      1.00      0.97        15
   virginica       1.00      0.94      0.97        16

    accuracy                           0.98        50
   macro avg       0.98      0.98      0.98        50
weighted avg       0.98      0.98      0.98        50
```