KNN iris dataset

December 18, 2022

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

[2]:
```
iris.feature_names
```

[2]:
```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

[3]:
```
iris.target_names
```

[3]:
```
array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

[4]:
```
df = pd.DataFrame(iris.data,columns=iris.feature_names)
df.head()
```

[4]:
```
   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
```

[5]:
```
df.shape
```

[5]:
```
(150, 4)
```

[6]:
```
df['target'] = iris.target
df.head()
```

[6]:
```
   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
```

```
   target
0       0
1       0
2       0
3       0
4       0
```

[7]:

```python
df[df.target==1].head()
```

[7]:

```
    sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
50                7.0               3.2                4.7               1.4
51                6.4               3.2                4.5               1.5
52                6.9               3.1                4.9               1.5
53                5.5               2.3                4.0               1.3
54                6.5               2.8                4.6               1.5

    target
50       1
51       1
52       1
53       1
54       1
```

[8]:

```python
df[df.target==2].head()
```

[8]:

```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
100                6.3               3.3                6.0               2.5
101                5.8               2.7                5.1               1.9
102                7.1               3.0                5.9               2.1
103                6.3               2.9                5.6               1.8
104                6.5               3.0                5.8               2.2

     target
100       2
101       2
102       2
103       2
104       2
```

[9]:

```python
df['flower_name'] =df.target.apply(lambda x: iris.target_names[x])
df.head()
```

[9]:

```
   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

   target flower_name
0       0      setosa
1       0      setosa
2       0      setosa
3       0      setosa
4       0      setosa
```

[10]:
```
df[45:55]
```

[10]:
```
    sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
45                4.8               3.0                1.4               0.3
46                5.1               3.8                1.6               0.2
47                4.6               3.2                1.4               0.2
48                5.3               3.7                1.5               0.2
49                5.0               3.3                1.4               0.2
50                7.0               3.2                4.7               1.4
51                6.4               3.2                4.5               1.5
52                6.9               3.1                4.9               1.5
53                5.5               2.3                4.0               1.3
54                6.5               2.8                4.6               1.5

    target flower_name
45       0      setosa
46       0      setosa
47       0      setosa
48       0      setosa
49       0      setosa
50       1  versicolor
51       1  versicolor
52       1  versicolor
53       1  versicolor
54       1  versicolor
```

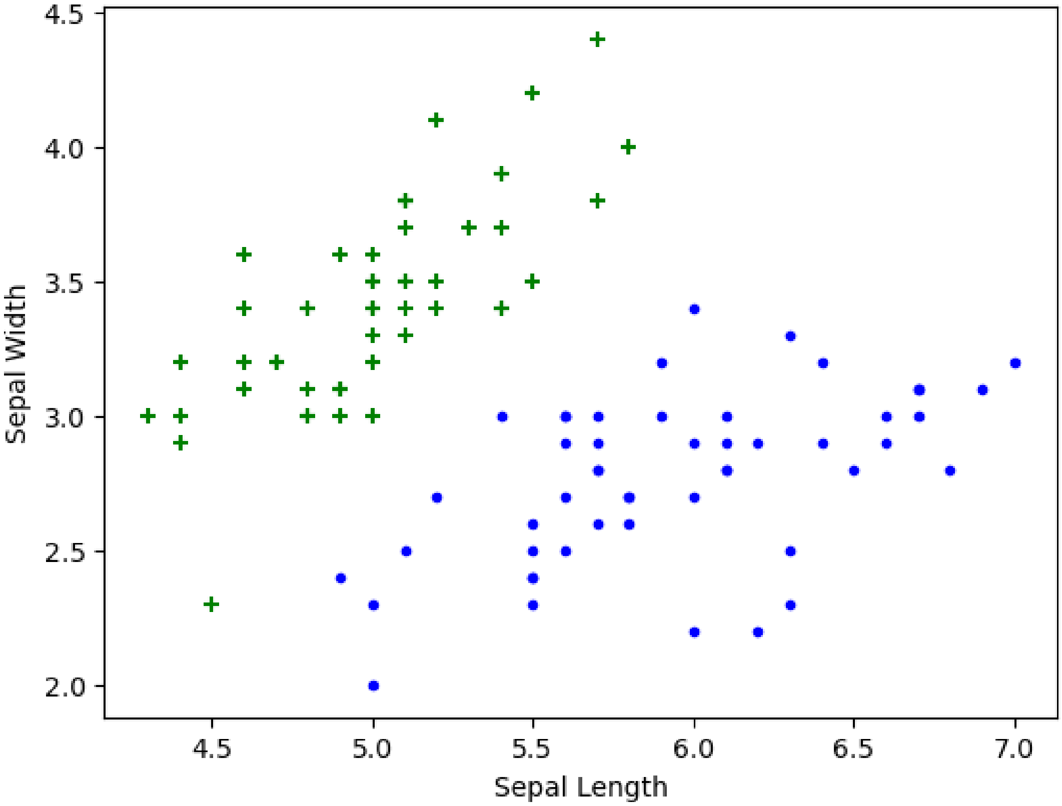
[11]:
```
df0 = df[:50]
df1 = df[50:100]
df2 = df[100:]
```

[12]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

[13]:
```
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.scatter(df0['sepal length (cm)'],df0['sepal width␣
 ↪(cm)'],color="green",marker='+')
```

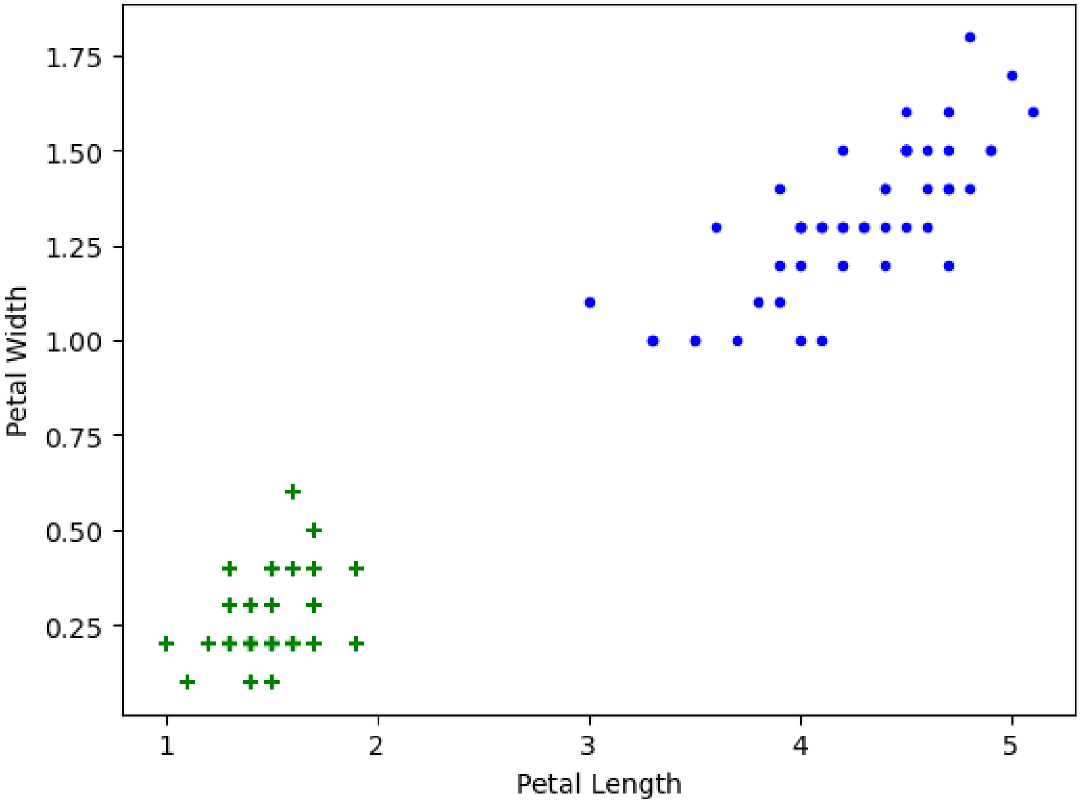
```
plt.scatter(df1['sepal length (cm)'],df1['sepal width␣
 ↪(cm)'],color="blue",marker='.')
```

```
[13]: <matplotlib.collections.PathCollection at 0x251add0ffa0>
```



```
[14]: plt.xlabel('Petal Length')
      plt.ylabel('Petal Width')
      plt.scatter(df0['petal length (cm)'],df0['petal width␣
       ↪(cm)'],color="green",marker='+')
      plt.scatter(df1['petal length (cm)'],df1['petal width␣
       ↪(cm)'],color="blue",marker='.')
```

```
[14]: <matplotlib.collections.PathCollection at 0x251ae40f1f0>
```



[15]:
```
from sklearn.model_selection import train_test_split
```

[16]:
```
X =df.drop(['target','flower_name'],axis='columns')
y = df.target
```

[17]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.
 ↪2,random_state=1)
```

[18]:
```
len(X_train)
```

[18]:
```
120
```

[19]:
```
len(X_test)
```

[19]:
```
30
```

[20]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=10)
```

[21]:
```
knn.fit(X_train, y_train)
```

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

```
[22]: knn.score(X_test, y_test)
```

```
C:\Users\Deepak\ana-conda-3\lib\site-
packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other
reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode`
typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will
change: the default value of `keepdims` will become False, the `axis` over which
the statistic is taken will be eliminated, and the value None will no longer be
accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

```
[22]: 0.9666666666666667
```

```
[23]: knn.predict([[4.8,3.0,1.5,0.3]])
```

```
C:\Users\Deepak\ana-conda-3\lib\site-packages\sklearn\base.py:450: UserWarning:
X does not have valid feature names, but KNeighborsClassifier was fitted with
feature names
  warnings.warn(
C:\Users\Deepak\ana-conda-3\lib\site-
packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other
reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode`
typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will
change: the default value of `keepdims` will become False, the `axis` over which
the statistic is taken will be eliminated, and the value None will no longer be
accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

```
[23]: array([0])
```

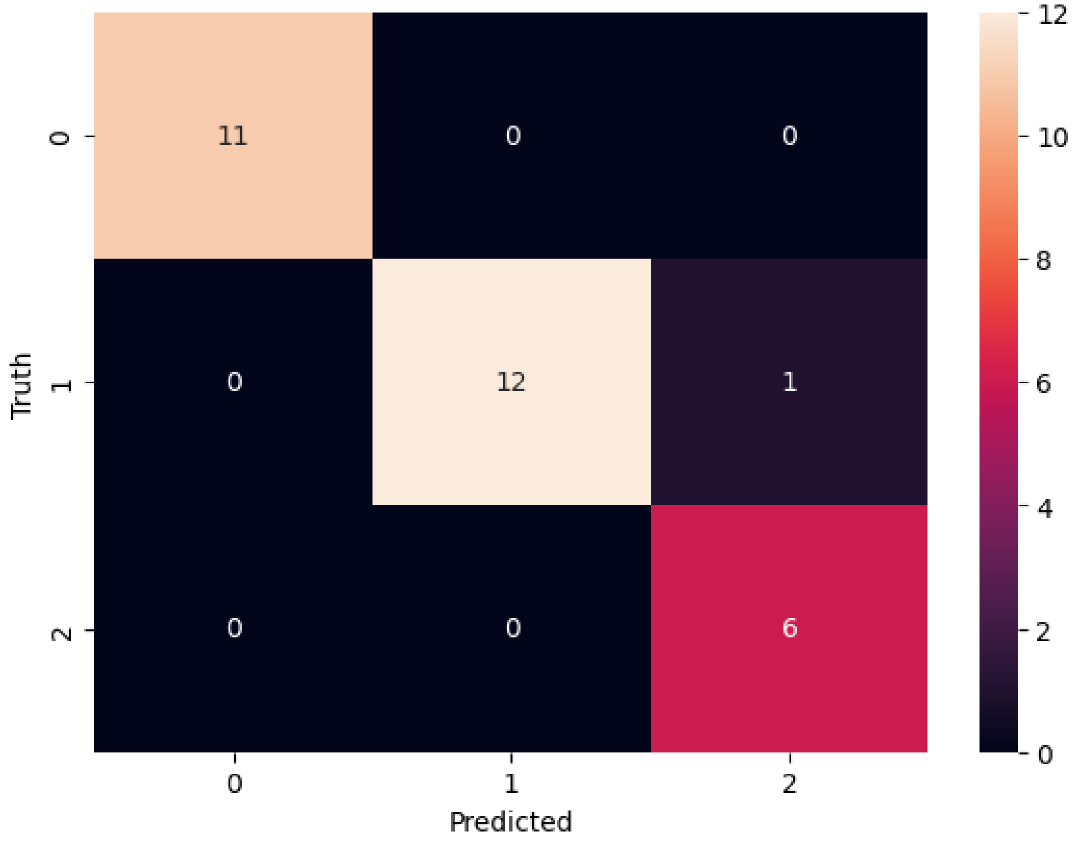
```
[24]: from sklearn.metrics import confusion_matrix
      y_pred = knn.predict(X_test)
      cm = confusion_matrix(y_test, y_pred)
      cm
```

```
C:\Users\Deepak\ana-conda-3\lib\site-
packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other
reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode`
typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will
change: the default value of `keepdims` will become False, the `axis` over which
the statistic is taken will be eliminated, and the value None will no longer be
accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

```
[24]: array([[11,  0,  0],
             [ 0, 12,  1],
             [ 0,  0,  6]], dtype=int64)
```

[25]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sn
plt.figure(figsize=(7,5))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

[25]:
```
Text(58.222222222222214, 0.5, 'Truth')
```



[26]:
```
from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       1.00      0.92      0.96        13
           2       0.86      1.00      0.92         6
```

```
    accuracy                           0.97        30
   macro avg       0.95      0.97      0.96        30
weighted avg       0.97      0.97      0.97        30
```

[ ]:

```

```