lab_3

September 11, 2024

1 Lab 3 - KNN

1.1 Setup

[3]:

```
import numpy as np
import pandas as pd
from sklearn import datasets
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

1.2 Load and Look at Data

[12]:

```
iris = datasets.load_iris()
# print(iris)
```

1.3 Get X, Y, Names from Dataset

[13]:

```
x = iris.data
y = iris.target
names_array = iris.target_names
# print(x)
# print(y)
# print(names_array)
```

1.4 Split Data into Train/Test (80/20)

[14]:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_state=81)
# print(x_train)
# print(y_train)
# print(x_test)
# print(y_test)
```

1.5 Fit KNN Alogorithm (K = 7)

[15]:
```
knn_7 = KNeighborsClassifier(n_neighbors=7)
knn_7.fit(x_train, y_train)
```

[15]:
```
KNeighborsClassifier(n_neighbors=7)
```

1.6 Run Predictions on Testing Set

[16]:
```
print(y_test)
print(names_array)
```

```
[2 0 2 1 0 0 1 0 2 0 1 1 2 2 0 2 0 0 0 1 1 1 0 0 1 1 1 2 0 0]
['setosa' 'versicolor' 'virginica']
```

[17]:
```
y_test_pred = knn_7.predict(x_test)
print(y_test_pred)
```

```
[2 0 2 1 0 0 1 0 2 0 1 1 2 2 0 2 0 0 0 1 1 1 0 0 1 1 1 2 0 0]
```

1.7 Evaluate Model

[18]:
```
confusion_matrix(y_test, y_test_pred)
```

[18]:
```
array([[13,  0,  0],
       [ 0, 10,  0],
       [ 0,  0,  7]])
```

[19]:
```
report = classification_report(y_test, y_test_pred, target_names=names_array)
print(report)
```

```
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        13
  versicolor       1.00      1.00      1.00        10
   virginica       1.00      1.00      1.00         7

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

1.8 Trying with fewer neighbors (K = 3)

[20]:
```
knn_3 = KNeighborsClassifier(n_neighbors=3)
knn_3.fit(x_train, y_train)

y_test_pred_3 = knn_3.predict(x_test)
print(y_test_pred_3)
```

```
[2 0 2 1 0 0 1 0 2 0 1 1 2 2 0 2 0 0 0 1 1 1 0 0 1 1 1 2 0 0]
```

[21]:

```python
confusion_matrix(y_test, y_test_pred_3)
```

[21]:

```
array([[13,  0,  0],
       [ 0, 10,  0],
       [ 0,  0,  7]])
```

[22]:

```python
report_3 = classification_report(y_test, y_test_pred, target_names=names_array)
print(report_3)
```

```
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        13
  versicolor       1.00      1.00      1.00        10
   virginica       1.00      1.00      1.00         7

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

1.9 Trying with 70/30 Split and K = 7

[23]:

```python
x_train_70, x_test_30, y_train_70, y_test_30 = train_test_split(x, y,␣
 ↪test_size=0.30, random_state=72)
```

[24]:

```python
knn_7_70 = KNeighborsClassifier(n_neighbors=7)
knn_7_70.fit(x_train_70, y_train_70)
```

[24]:

```
KNeighborsClassifier(n_neighbors=7)
```

[25]:

```python
y_test_30_pred = knn_7_70.predict(x_test_30)
confusion_matrix(y_test_30, y_test_30_pred)
```

[25]:

```
array([[17,  0,  0],
       [ 0, 13,  0],
       [ 0,  1, 14]])
```

[26]:

```python
report_70 = classification_report(y_test_30, y_test_30_pred,␣
 ↪target_names=names_array)
print(report_70)
```

```
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        17
  versicolor       0.93      1.00      0.96        13
   virginica       1.00      0.93      0.97        15
```

```
    accuracy                           0.98        45
   macro avg       0.98      0.98      0.98        45
weighted avg       0.98      0.98      0.98        45
```