day45-knn-classification-iris

November 14, 2023

Day45: KNN Classification (Iris) By: Loga Aswin

[4]:
```
#importing libraries
import numpy as np
import pandas as pd
```

Data Pre-processing :

[5]:
```
# Importing the dataset
df = pd.read_csv('/content/IRIS.csv')
df
```

[5]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
..	…	…	…	…	…
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

[150 rows x 5 columns]

[6]:
```
df.shape
```

[6]:
```
(150, 5)
```

[7]:
```
df.head()
```

[7]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

[8]:
```
df.info()
```

```
<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
```

[9]:
```
df.species.unique()
```

[9]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

[10]:
```
df.species.value_counts()
```

[10]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: species, dtype: int64
```

[11]:
```
df['species'] = df['species'].replace({'Iris-setosa':1, 'Iris-versicolor':2,␣
↪'Iris-virginica':3})
```

[12]:
```
df.head()
```

[12]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	1
1	4.9	3.0	1.4	0.2	1
2	4.7	3.2	1.3	0.2	1
3	4.6	3.1	1.5	0.2	1
4	5.0	3.6	1.4	0.2	1

[13]:
```
df.tail()
```

[13]:

	sepal_length	sepal_width	petal_length	petal_width	species
145	6.7	3.0	5.2	2.3	3
146	6.3	2.5	5.0	1.9	3
147	6.5	3.0	5.2	2.0	3
148	6.2	3.4	5.4	2.3	3
149	5.9	3.0	5.1	1.8	3

[14]:
```
X = df.drop('species', axis=1)
Y = df['species']
```

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

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.25,␣
 ↪random_state = 0)
```

[17]:
```
from sklearn.neighbors import KNeighborsClassifier
```

Creating and Training KNN Model:

[19]:
```
knn_clf = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
knn_clf.fit(X_train, y_train)
```

[19]:
```
KNeighborsClassifier()
```

Predict Test Results:

[21]:
```
y_pred = knn_clf.predict(X_test)
```

[22]:
```
pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
```

[22]:

	Actual	Predicted
114	3	3
62	2	2
33	1	1
107	3	3
7	1	1
100	3	3
40	1	1
86	2	2
76	2	2
71	2	2
134	3	3
51	2	2
73	2	2
54	2	2
63	2	2
37	1	1
78	2	2
90	2	2
45	1	1
16	1	1
121	3	3
66	2	2
24	1	1
8	1	1

126	3	3
22	1	1
44	1	1
97	2	2
93	2	2
26	1	1
137	3	3
84	2	2
27	1	1
127	3	3
132	3	3
59	2	2
18	1	1
83	2	3

Model Evaluation Metrics

[23]:
```
from sklearn.metrics import confusion_matrix, classification_report,␣
↪accuracy_score

score = accuracy_score(y_test, y_pred)
print(score)
```

```
0.9736842105263158
```

[24]:
```
report = classification_report(y_test, y_pred)
print(report)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        13
           2       1.00      0.94      0.97        16
           3       0.90      1.00      0.95         9

    accuracy                           0.97        38
   macro avg       0.97      0.98      0.97        38
weighted avg       0.98      0.97      0.97        38
```

[25]:
```
m1 = confusion_matrix(y_test, y_pred)
print(m1)
```

```
[[13  0  0]
 [ 0 15  1]
 [ 0  0  9]]
```