day53-random-forest-iris-ipynb

November 26, 2023

Day53 Random Forest(Iris) By: Loga Aswin

[3]:
```
# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[4]:
```
#Load dataset
df = pd.read_csv('/content/IRIS..csv')
```

Exploratory Data Analysis(EDA):

[5]:
```
df.head()
```

[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

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

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

[7]:
```
df.isnull().sum()
```

[7]:
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

[9]:
```
df['species'].unique
```

[9]:
```
<bound method Series.unique of 0         Iris-setosa
1         Iris-setosa
2         Iris-setosa
```

```
3          Iris-setosa
4          Iris-setosa
             …
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: species, Length: 150, dtype: object>
```

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

[12]:
```
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    int64
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
```

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

[13]:

	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

[15]:
```
X = df.drop('species', axis=1)
y = df['species']
```

Split into train and test data:

[16]:
```
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,␣
↪random_state=42)
```

Using RandomForestClassifier Model to train data :

[18]:
```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(n_estimators=10, criterion='entropy')
model.fit(X_train, y_train)
```

[18]:
```
RandomForestClassifier(criterion='entropy', n_estimators=10)
```

[20]:
```
# predict test results
y_pred = model.predict(X_test)
```

[34]:
```
pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
```

[34]:

	Actual	Predicted
73	2	2
18	1	1
118	3	3
78	2	2
76	2	2
31	1	1
64	2	2
141	3	3
68	2	2
82	2	2
110	3	3
12	1	1
36	1	1
9	1	1
19	1	1
56	2	2
104	3	3
69	2	2
55	2	2
132	3	3
29	1	1
127	3	3
26	1	1
128	3	3
131	3	3
145	3	3
108	3	3
143	3	3
45	1	1
30	1	1

Model Evaluation Metrics:

[32]:
```
from sklearn.metrics import confusion_matrix, classification_report

matrix = confusion_matrix(y_test, y_pred)
matrix
```

[32]:
```
array([[10,  0,  0],
       [ 0,  9,  0],
       [ 0,  0, 11]])
```

[33]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        10
           2       1.00      1.00      1.00         9
           3       1.00      1.00      1.00        11

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

[ ]: