lab3_logistic_regression

November 15, 2022

LOGISTIC REGRESSION

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import preprocessing
from sklearn.metrics import␣
 ↪accuracy_score,confusion_matrix,recall_score,precision_score
```

[ ]:
```
data=pd.read_csv("C:\Coding\ML_python\machine-learning-lab-main\datasets\iris.
 ↪csv")
data.head()
```

[ ]:

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

[ ]:
```
X=data.iloc[:,:4]
y=data.iloc[:,4]
print(X)
print(y)
```

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

```
[150 rows x 4 columns]
0         setosa
1         setosa
2         setosa
3         setosa
4         setosa
          …
145    virginica
146    virginica
147    virginica
148    virginica
149    virginica
Name: species, Length: 150, dtype: object
```

[ ]:
```
le = preprocessing.LabelEncoder()
y=le.fit_transform(y)
y
```

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

[ ]:
```
xtrain, xtest, ytrain, ytest = train_test_split(X, y, test_size=0.3,␣
 ↪random_state=0)
print("training dataset:\n",xtrain,"\n",ytrain)
print("\ntesting dataset:\n",xtest,"\n",ytest)
```

```
training dataset:
      sepal_length  sepal_width  petal_length  petal_width
60             5.0          2.0           3.5          1.0
116            6.5          3.0           5.5          1.8
144            6.7          3.3           5.7          2.5
119            6.0          2.2           5.0          1.5
108            6.7          2.5           5.8          1.8
..             …            …             …            …
9              4.9          3.1           1.5          0.1
103            6.3          2.9           5.6          1.8
67             5.8          2.7           4.1          1.0
117            7.7          3.8           6.7          2.2
47             4.6          3.2           1.4          0.2

[105 rows x 4 columns]
 [1 2 2 2 2 1 2 1 1 2 2 2 2 1 2 1 0 2 1 1 1 1 2 0 0 2 1 0 0 1 0 2 1 0 1 2 1
```

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

testing dataset:

	sepal_length	sepal_width	petal_length	petal_width
114	5.8	2.8	5.1	2.4
62	6.0	2.2	4.0	1.0
33	5.5	4.2	1.4	0.2
107	7.3	2.9	6.3	1.8
7	5.0	3.4	1.5	0.2
100	6.3	3.3	6.0	2.5
40	5.0	3.5	1.3	0.3
86	6.7	3.1	4.7	1.5
76	6.8	2.8	4.8	1.4
71	6.1	2.8	4.0	1.3
134	6.1	2.6	5.6	1.4
51	6.4	3.2	4.5	1.5
73	6.1	2.8	4.7	1.2
54	6.5	2.8	4.6	1.5
63	6.1	2.9	4.7	1.4
37	4.9	3.1	1.5	0.1
78	6.0	2.9	4.5	1.5
90	5.5	2.6	4.4	1.2
45	4.8	3.0	1.4	0.3
16	5.4	3.9	1.3	0.4
121	5.6	2.8	4.9	2.0
66	5.6	3.0	4.5	1.5
24	4.8	3.4	1.9	0.2
8	4.4	2.9	1.4	0.2
126	6.2	2.8	4.8	1.8
22	4.6	3.6	1.0	0.2
44	5.1	3.8	1.9	0.4
97	6.2	2.9	4.3	1.3
93	5.0	2.3	3.3	1.0
26	5.0	3.4	1.6	0.4
137	6.4	3.1	5.5	1.8
84	5.4	3.0	4.5	1.5
27	5.2	3.5	1.5	0.2
127	6.1	3.0	4.9	1.8
132	6.4	2.8	5.6	2.2
59	5.2	2.7	3.9	1.4
18	5.7	3.8	1.7	0.3
83	6.0	2.7	5.1	1.6
61	5.9	3.0	4.2	1.5
92	5.8	2.6	4.0	1.2
112	6.8	3.0	5.5	2.1
2	4.7	3.2	1.3	0.2
141	6.9	3.1	5.1	2.3

```
43            5.0           3.5            1.6           0.6
10            5.4           3.7            1.5           0.2
 [2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0 2 1 0 2 2 1 0
 1 1 1 2 0 2 0 0]
```

[ ]:
```
logr= LogisticRegression().fit(xtrain, ytrain)
ypred=logr.predict(xtest)
print("the predicted y values: ",ypred)
print("the test y values: ",ytest)
print("the score for the training data set is: ",logr.score(xtest,ytest))
```

```
the predicted y values:  [2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0
1 1 0 2 1 0 2 2 1 0
 2 1 1 2 0 2 0 0]
the test y values:  [2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0
2 1 0 2 2 1 0
 1 1 1 2 0 2 0 0]
the score for the training data set is:  0.9777777777777777
```

[ ]:
```
print("the accuracy score: ",accuracy_score(ytest,ypred))
mat=confusion_matrix(ytest,ypred)
print("the confusion matrix is:\n",mat)
```

```
the accuracy score:  0.9777777777777777
the confusion matrix is:
 [[16  0  0]
 [ 0 17  1]
 [ 0  0 11]]
```

[ ]:
```
print("the recall score: ",recall_score(ytest,ypred,average='macro'))
print("the precision score: ",precision_score(ytest,ypred,average='macro'))
```

```
the recall score:  0.9814814814814815
the precision score:  0.9722222222222222
```

[ ]:
```
from sklearn.metrics import classification_report
print(classification_report(ytest, ypred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        16
           1       1.00      0.94      0.97        18
           2       0.92      1.00      0.96        11

    accuracy                           0.98        45
   macro avg       0.97      0.98      0.98        45
weighted avg       0.98      0.98      0.98        45
```