Class_16.04.2023_Naive_Bayes_Classifier

April 16, 2023

- Bayes theorem
- Gaussian Distribution/Normal distribution

[48]:
```
#importing the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_breast_cancer
```

[49]:
```
cancer = load_breast_cancer()
```

[50]:
```
cancer.keys()
```

[50]:
```
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names',
'filename', 'data_module'])
```

[51]:
```
cancer.feature_names
```

[51]:
```
array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter error', 'area error',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error',
       'fractal dimension error', 'worst radius', 'worst texture',
       'worst perimeter', 'worst area', 'worst smoothness',
       'worst compactness', 'worst concavity', 'worst concave points',
       'worst symmetry', 'worst fractal dimension'], dtype='<U23')
```

[52]:
```
cancer.target_names
```

[52]:
```
array(['malignant', 'benign'], dtype='<U9')
```

[53]:
```
print(cancer.DESCR)
```

```
.. _breast_cancer_dataset:

Breast cancer wisconsin (diagnostic) dataset
--------------------------------------------
```

Data Set Characteristics:

    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and the class

    :Attribute Information:
        - radius (mean of distances from center to points on the perimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
        - area
        - smoothness (local variation in radius lengths)
        - compactness (perimeter^2 / area - 1.0)
        - concavity (severity of concave portions of the contour)
        - concave points (number of concave portions of the contour)
        - symmetry
        - fractal dimension ("coastline approximation" - 1)

        The mean, standard error, and "worst" or largest (mean of the three
        worst/largest values) of these features were computed for each image,
        resulting in 30 features. For instance, field 0 is Mean Radius, field
        10 is Radius SE, field 20 is Worst Radius.

        - class:
                - WDBC-Malignant
                - WDBC-Benign

    :Summary Statistics:

    ===================================== ====== ======
                                           Min   Max
    ===================================== ====== ======
    radius (mean):   6.981 28.11
    texture (mean):   9.71   39.28
    perimeter (mean):   43.79 188.5
    area (mean):   143.5 2501.0
    smoothness (mean):   0.053 0.163
    compactness (mean):   0.019 0.345
    concavity (mean):   0.0   0.427
    concave points (mean):   0.0   0.201
    symmetry (mean):   0.106 0.304
    fractal dimension (mean):   0.05   0.097
    radius (standard error):   0.112 2.873
    texture (standard error):   0.36   4.885
    perimeter (standard error):   0.757 21.98
    area (standard error):   6.802 542.2
    smoothness (standard error):   0.002 0.031
    compactness (standard error):   0.002 0.135

```
    concavity (standard error):           0.0    0.396
    concave points (standard error):      0.0    0.053
    symmetry (standard error):            0.008  0.079
    fractal dimension (standard error):   0.001  0.03
    radius (worst):                       7.93   36.04
    texture (worst):                      12.02  49.54
    perimeter (worst):                    50.41  251.2
    area (worst):                         185.2  4254.0
    smoothness (worst):                   0.071  0.223
    compactness (worst):                  0.027  1.058
    concavity (worst):                    0.0    1.252
    concave points (worst):               0.0    0.291
    symmetry (worst):                     0.156  0.664
    fractal dimension (worst):            0.055  0.208
    ===================================== ====== ======

    :Missing Attribute Values: None

    :Class Distribution: 212 - Malignant, 357 - Benign

    :Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian

    :Donor: Nick Street

    :Date: November, 1995

This is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.
https://goo.gl/U2Uwz2

Features are computed from a digitized image of a fine needle
aspirate (FNA) of a breast mass.  They describe
characteristics of the cell nuclei present in the image.

Separating plane described above was obtained using
Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree
Construction Via Linear Programming." Proceedings of the 4th
Midwest Artificial Intelligence and Cognitive Science Society,
pp. 97-101, 1992], a classification method which uses linear
programming to construct a decision tree.  Relevant features
were selected using an exhaustive search in the space of 1-4
features and 1-3 separating planes.

The actual linear program used to obtain the separating plane
in the 3-dimensional space is that described in:
[K. P. Bennett and O. L. Mangasarian: "Robust Linear
Programming Discrimination of Two Linearly Inseparable Sets",
Optimization Methods and Software 1, 1992, 23-34].
```

```
This database is also available through the UW CS ftp server:

ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

.. topic:: References

   - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction
     for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on
     Electronic Imaging: Science and Technology, volume 1905, pages 861-870,
     San Jose, CA, 1993.
   - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and
     prognosis via linear programming. Operations Research, 43(4), pages
570-577,
     July-August 1995.
   - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning
techniques
     to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77
(1994)
     163-171.
```

[54]:
```
cancer_df = pd.DataFrame(data = cancer.data , columns = cancer.feature_names)
```

[55]:
```
cancer_df.head()
```

[55]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	\
0	17.99	10.38	122.80	1001.0	0.11840	
1	20.57	17.77	132.90	1326.0	0.08474	
2	19.69	21.25	130.00	1203.0	0.10960	
3	11.42	20.38	77.58	386.1	0.14250	
4	20.29	14.34	135.10	1297.0	0.10030	

	mean compactness	mean concavity	mean concave points	mean symmetry	\
0	0.27760	0.3001	0.14710	0.2419	
1	0.07864	0.0869	0.07017	0.1812	
2	0.15990	0.1974	0.12790	0.2069	
3	0.28390	0.2414	0.10520	0.2597	
4	0.13280	0.1980	0.10430	0.1809	

	mean fractal dimension	…	worst radius	worst texture	worst perimeter	\
0	0.07871	…	25.38	17.33	184.60	
1	0.05667	…	24.99	23.41	158.80	
2	0.05999	…	23.57	25.53	152.50	
3	0.09744	…	14.91	26.50	98.87	
4	0.05883	…	22.54	16.67	152.20	

	worst area	worst smoothness	worst compactness	worst concavity	\

```
0      2019.0            0.1622             0.6656           0.7119
1      1956.0            0.1238             0.1866           0.2416
2      1709.0            0.1444             0.4245           0.4504
3       567.7            0.2098             0.8663           0.6869
4      1575.0            0.1374             0.2050           0.4000

   worst concave points  worst symmetry  worst fractal dimension
0                0.2654          0.4601                  0.11890
1                0.1860          0.2750                  0.08902
2                0.2430          0.3613                  0.08758
3                0.2575          0.6638                  0.17300
4                0.1625          0.2364                  0.07678

[5 rows x 30 columns]
```

[56]:
```
cancer_df['target'] = cancer.target
```

[57]:
```
cancer_df.head(20)
```

[57]:
```
    mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0         17.99         10.38          122.80     1001.0          0.11840
1         20.57         17.77          132.90     1326.0          0.08474
2         19.69         21.25          130.00     1203.0          0.10960
3         11.42         20.38           77.58      386.1          0.14250
4         20.29         14.34          135.10     1297.0          0.10030
5         12.45         15.70           82.57      477.1          0.12780
6         18.25         19.98          119.60     1040.0          0.09463
7         13.71         20.83           90.20      577.9          0.11890
8         13.00         21.82           87.50      519.8          0.12730
9         12.46         24.04           83.97      475.9          0.11860
10        16.02         23.24          102.70      797.8          0.08206
11        15.78         17.89          103.60      781.0          0.09710
12        19.17         24.80          132.40     1123.0          0.09740
13        15.85         23.95          103.70      782.7          0.08401
14        13.73         22.61           93.60      578.3          0.11310
15        14.54         27.54           96.73      658.8          0.11390
16        14.68         20.13           94.74      684.5          0.09867
17        16.13         20.68          108.10      798.8          0.11700
18        19.81         22.15          130.00     1260.0          0.09831
19        13.54         14.36           87.46      566.3          0.09779

    mean compactness  mean concavity  mean concave points  mean symmetry  \
0            0.27760         0.30010              0.14710         0.2419
1            0.07864         0.08690              0.07017         0.1812
2            0.15990         0.19740              0.12790         0.2069
3            0.28390         0.24140              0.10520         0.2597
4            0.13280         0.19800              0.10430         0.1809
```

5	0.17000	0.15780	0.08089	0.2087
6	0.10900	0.11270	0.07400	0.1794
7	0.16450	0.09366	0.05985	0.2196
8	0.19320	0.18590	0.09353	0.2350
9	0.23960	0.22730	0.08543	0.2030
10	0.06669	0.03299	0.03323	0.1528
11	0.12920	0.09954	0.06606	0.1842
12	0.24580	0.20650	0.11180	0.2397
13	0.10020	0.09938	0.05364	0.1847
14	0.22930	0.21280	0.08025	0.2069
15	0.15950	0.16390	0.07364	0.2303
16	0.07200	0.07395	0.05259	0.1586
17	0.20220	0.17220	0.10280	0.2164
18	0.10270	0.14790	0.09498	0.1582
19	0.08129	0.06664	0.04781	0.1885

	mean fractal dimension	…	worst texture	worst perimeter	worst area	\
0	0.07871	…	17.33	184.60	2019.0	
1	0.05667	…	23.41	158.80	1956.0	
2	0.05999	…	25.53	152.50	1709.0	
3	0.09744	…	26.50	98.87	567.7	
4	0.05883	…	16.67	152.20	1575.0	
5	0.07613	…	23.75	103.40	741.6	
6	0.05742	…	27.66	153.20	1606.0	
7	0.07451	…	28.14	110.60	897.0	
8	0.07389	…	30.73	106.20	739.3	
9	0.08243	…	40.68	97.65	711.4	
10	0.05697	…	33.88	123.80	1150.0	
11	0.06082	…	27.28	136.50	1299.0	
12	0.07800	…	29.94	151.70	1332.0	
13	0.05338	…	27.66	112.00	876.5	
14	0.07682	…	32.01	108.80	697.7	
15	0.07077	…	37.13	124.10	943.2	
16	0.05922	…	30.88	123.40	1138.0	
17	0.07356	…	31.48	136.80	1315.0	
18	0.05395	…	30.88	186.80	2398.0	
19	0.05766	…	19.26	99.70	711.2	

	worst smoothness	worst compactness	worst concavity	\
0	0.1622	0.6656	0.7119	
1	0.1238	0.1866	0.2416	
2	0.1444	0.4245	0.4504	
3	0.2098	0.8663	0.6869	
4	0.1374	0.2050	0.4000	
5	0.1791	0.5249	0.5355	
6	0.1442	0.2576	0.3784	
7	0.1654	0.3682	0.2678	

```
8            0.1703            0.5401            0.5390
9            0.1853            1.0580            1.1050
10           0.1181            0.1551            0.1459
11           0.1396            0.5609            0.3965
12           0.1037            0.3903            0.3639
13           0.1131            0.1924            0.2322
14           0.1651            0.7725            0.6943
15           0.1678            0.6577            0.7026
16           0.1464            0.1871            0.2914
17           0.1789            0.4233            0.4784
18           0.1512            0.3150            0.5372
19           0.1440            0.1773            0.2390

    worst concave points  worst symmetry  worst fractal dimension  target
0                0.26540          0.4601                  0.11890       0
1                0.18600          0.2750                  0.08902       0
2                0.24300          0.3613                  0.08758       0
3                0.25750          0.6638                  0.17300       0
4                0.16250          0.2364                  0.07678       0
5                0.17410          0.3985                  0.12440       0
6                0.19320          0.3063                  0.08368       0
7                0.15560          0.3196                  0.11510       0
8                0.20600          0.4378                  0.10720       0
9                0.22100          0.4366                  0.20750       0
10               0.09975          0.2948                  0.08452       0
11               0.18100          0.3792                  0.10480       0
12               0.17670          0.3176                  0.10230       0
13               0.11190          0.2809                  0.06287       0
14               0.22080          0.3596                  0.14310       0
15               0.17120          0.4218                  0.13410       0
16               0.16090          0.3029                  0.08216       0
17               0.20730          0.3706                  0.11420       0
18               0.23880          0.2768                  0.07615       0
19               0.12880          0.2977                  0.07259       1

[20 rows x 31 columns]
```

[58]:
```
x = cancer_df.drop('target',axis = 'columns')
y = cancer_df.target
```

[59]:
```
x.shape
```

[59]: (569, 30)

[60]:
```
y.shape
```

[60]: (569,)

[61]:
```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size = 0.3)
```

[62]:
```
xtrain.shape
```

[62]: (398, 30)

[63]:
```
xtest.shape
```

[63]: (171, 30)

[64]:
```
ytrain.shape
```

[64]: (398,)

[65]:
```
ytest.shape
```

[65]: (171,)

[66]:
```
from sklearn.naive_bayes import GaussianNB
```

[67]:
```
nb = GaussianNB()
nb.fit(xtrain,ytrain)
```

[67]: GaussianNB()

[68]:
```
nb.score(xtest,ytest)
```

[68]: 0.9298245614035088

[69]:
```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score , confusion_matrix ,␣
↪ConfusionMatrixDisplay
```

[70]:
```
#accuracy testing

''' On training data'''
predict_train = nb.predict(xtrain)
accuracy_train = accuracy_score(ytrain,predict_train)

print(f'the training data accuracy is {accuracy_train}')
```

```
the training data accuracy is 0.9447236180904522
```

[71]:
```
'''On test data'''
predict_test = nb.predict(xtest)
accuracy_test = accuracy_score(ytest,predict_test)
print(f'the test data accuracy is {accuracy_test}')
```

```
the test data accuracy is 0.9298245614035088
```

[72]:
```
# finding cross validation score
cv_score = cross_val_score(nb,x,y,cv = 5)
```

[73]:
```
print(cv_score)
```

```
[0.92105263 0.92105263 0.94736842 0.94736842 0.95575221]
```

[83]:
```
# Build a confusion matrix
prediction = nb.predict(xtest)
prediction
```

[83]:
```
array([1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0,
       1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0,
       1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1,
       1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1])
```

[84]:
```
ytest
```

[84]:
```
173    1
489    0
561    1
500    1
528    1
      ..
148    1
76     1
47     0
553    1
305    1
Name: target, Length: 171, dtype: int32
```

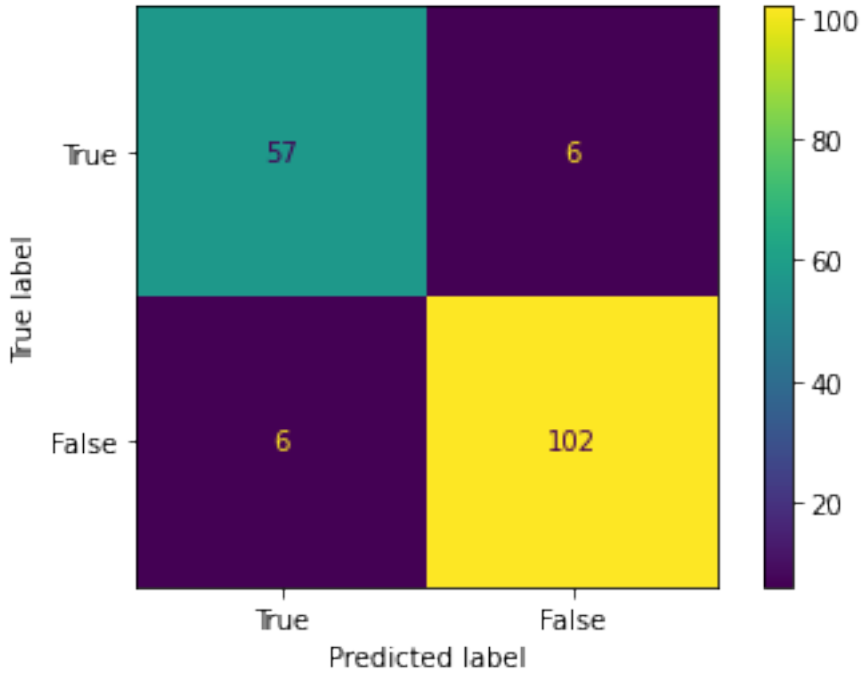
[85]:
```
cm = confusion_matrix(ytest,prediction)
```

[86]:
```
cm
```

[86]:
```
array([[ 57,   6],
       [  6, 102]], dtype=int64)
```

[89]:
```
disp = ConfusionMatrixDisplay(confusion_matrix=cm ,␣
 ↪display_labels=['True','False'])
```

[88]:
```
disp.plot()
plt.show()
```


True
False
True label
57
6
6
102
True
False
Predicted label
100
80
60
40
20

[ ]: