bcp-logistic

August 29, 2024

[1]:
```
import numpy as np
import pandas as pd
import sklearn.datasets
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

[2]:
```
#Data Collection And Processing
```

[3]:
```
breast_cancer_dataset=sklearn.datasets.load_breast_cancer()
```

[4]:
```
print(breast_cancer_dataset)
```

```
{'data': array([[1.799e+01, 1.038e+01, 1.228e+02, …, 2.654e-01, 4.601e-01,
        1.189e-01],
       [2.057e+01, 1.777e+01, 1.329e+02, …, 1.860e-01, 2.750e-01,
        8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, …, 2.430e-01, 3.613e-01,
        8.758e-02],
       …,
       [1.660e+01, 2.808e+01, 1.083e+02, …, 1.418e-01, 2.218e-01,
        7.820e-02],
       [2.060e+01, 2.933e+01, 1.401e+02, …, 2.650e-01, 4.087e-01,
        1.240e-01],
       [7.760e+00, 2.454e+01, 4.792e+01, …, 0.000e+00, 2.871e-01,
        7.039e-02]]), 'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0,
       1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
       0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0,
       0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0,
       0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1,
       1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
       1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 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, 0, 0, 0, 0, 0, 0, 1]), 'frame': None,
'target_names': array(['malignant', 'benign'], dtype='<U9'), 'DESCR': '..
_breast_cancer_dataset:\n\nBreast cancer wisconsin (diagnostic)
dataset\n--------------------------------------------\n\n**Data Set
Characteristics:**\n\n:Number of Instances: 569\n\n:Number of Attributes: 30
numeric, predictive attributes and the class\n\n:Attribute Information:\n    -
radius (mean of distances from center to points on the perimeter)\n    - texture
(standard deviation of gray-scale values)\n    - perimeter\n    - area\n    -
smoothness (local variation in radius lengths)\n    - compactness (perimeter^2 /
area - 1.0)\n    - concavity (severity of concave portions of the contour)\n
- concave points (number of concave portions of the contour)\n    - symmetry\n
- fractal dimension ("coastline approximation" - 1)\n\n    The mean, standard
error, and "worst" or largest (mean of the three\n    worst/largest values) of
these features were computed for each image,\n    resulting in 30 features.  For
instance, field 0 is Mean Radius, field\n    10 is Radius SE, field 20 is Worst
Radius.\n\n    - class:\n            - WDBC-Malignant\n            - WDBC-
Benign\n\n:Summary Statistics:\n\n===================================== ======
======\n                                        Min
Max\n===================================== ====== ======\nradius (mean):
6.981  28.11\ntexture (mean):                       9.71   39.28\nperimeter
(mean):                     43.79  188.5\narea (mean):
143.5  2501.0\nsmoothness (mean):                    0.053  0.163\ncompactness
(mean):                   0.019  0.345\nconcavity (mean):
0.0    0.427\nconcave points (mean):               0.0    0.201\nsymmetry
(mean):                     0.106  0.304\nfractal dimension (mean):
0.05   0.097\nradius (standard error):             0.112  2.873\ntexture
(standard error):            0.36   4.885\nperimeter (standard error):
0.757  21.98\narea (standard error):               6.802  542.2\nsmoothness
(standard error):          0.002  0.031\ncompactness (standard error):
0.002  0.135\nconcavity (standard error):          0.0    0.396\nconcave points
(standard error):      0.0    0.053\nsymmetry (standard error):           0.008
0.079\nfractal dimension (standard error):   0.001  0.03\nradius (worst):
7.93   36.04\ntexture (worst):                      12.02  49.54\nperimeter
(worst):                    50.41  251.2\narea (worst):
185.2  4254.0\nsmoothness (worst):                   0.071  0.223\ncompactness
```

```
(worst):                  0.027  1.058\nconcavity (worst):
0.0    1.252\nconcave points (worst):              0.0    0.291\nsymmetry
(worst):                     0.156  0.664\nfractal dimension (worst):
0.055  0.208\n===================================== ====== ======\n\n:Missing
Attribute Values: None\n\n:Class Distribution: 212 - Malignant, 357 -
Benign\n\n:Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L.
Mangasarian\n\n:Donor: Nick Street\n\n:Date: November, 1995\n\nThis is a copy of
UCI ML Breast Cancer Wisconsin (Diagnostic)
datasets.\nhttps://goo.gl/U2Uwz2\n\nFeatures are computed from a digitized image
of a fine needle\naspirate (FNA) of a breast mass.  They
describe\ncharacteristics of the cell nuclei present in the image.\n\nSeparating
plane described above was obtained using\nMultisurface Method-Tree (MSM-T) [K.
P. Bennett, "Decision Tree\nConstruction Via Linear Programming." Proceedings of
the 4th\nMidwest Artificial Intelligence and Cognitive Science Society,\npp.
97-101, 1992], a classification method which uses linear\nprogramming to
construct a decision tree.  Relevant features\nwere selected using an exhaustive
search in the space of 1-4\nfeatures and 1-3 separating planes.\n\nThe actual
linear program used to obtain the separating plane\nin the 3-dimensional space
is that described in:\n[K. P. Bennett and O. L. Mangasarian: "Robust
Linear\nProgramming Discrimination of Two Linearly Inseparable
Sets",\nOptimization Methods and Software 1, 1992, 23-34].\n\nThis database is
also available through the UW CS ftp server:\n\nftp ftp.cs.wisc.edu\ncd math-
prog/cpo-dataset/machine-learn/WDBC/\n\n|details-
start|\n**References**\n|details-split|\n\n- W.N. Street, W.H. Wolberg and O.L.
Mangasarian. Nuclear feature extraction\n  for breast tumor diagnosis. IS&T/SPIE
1993 International Symposium on\n  Electronic Imaging: Science and Technology,
volume 1905, pages 861-870,\n  San Jose, CA, 1993.\n- O.L. Mangasarian, W.N.
Street and W.H. Wolberg. Breast cancer diagnosis and\n  prognosis via linear
programming. Operations Research, 43(4), pages 570-577,\n  July-August 1995.\n-
W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques\n
to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994)\n
163-171.\n\n|details-end|\n', 'feature_names': 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'), 'filename':
'breast_cancer.csv', 'data_module': 'sklearn.datasets.data'}
```

[5]:
```
#loading the data into data frame
```

[6]:
```
df=pd.DataFrame(breast_cancer_dataset.data,columns=breast_cancer_dataset.
↪feature_names)
```

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

[7]:

	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]

[8]:
```
#dding the target column
```

[9]:
```
df['label']=breast_cancer_dataset.target
```

[10]:
```
#Print last 5 rows
```

[11]:
```
df.tail()
```

[11]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	\
564	21.56	22.39	142.00	1479.0	0.11100	
565	20.13	28.25	131.20	1261.0	0.09780	
566	16.60	28.08	108.30	858.1	0.08455	
567	20.60	29.33	140.10	1265.0	0.11780	
568	7.76	24.54	47.92	181.0	0.05263	

	mean compactness	mean concavity	mean concave points	mean symmetry	\
564	0.11590	0.24390	0.13890	0.1726	
565	0.10340	0.14400	0.09791	0.1752	
566	0.10230	0.09251	0.05302	0.1590	
567	0.27700	0.35140	0.15200	0.2397	
568	0.04362	0.00000	0.00000	0.1587	

	mean fractal dimension	…	worst texture	worst perimeter	worst area	\
564	0.05623	…	26.40	166.10	2027.0	
565	0.05533	…	38.25	155.00	1731.0	
566	0.05648	…	34.12	126.70	1124.0	
567	0.07016	…	39.42	184.60	1821.0	
568	0.05884	…	30.37	59.16	268.6	

	worst smoothness	worst compactness	worst concavity	\
564	0.14100	0.21130	0.4107	
565	0.11660	0.19220	0.3215	
566	0.11390	0.30940	0.3403	
567	0.16500	0.86810	0.9387	
568	0.08996	0.06444	0.0000	

	worst concave points	worst symmetry	worst fractal dimension	label
564	0.2216	0.2060	0.07115	0
565	0.1628	0.2572	0.06637	0
566	0.1418	0.2218	0.07820	0
567	0.2650	0.4087	0.12400	0
568	0.0000	0.2871	0.07039	1

[5 rows x 31 columns]

[12]:
```
#No of cols and rows in the dataset
```

[13]:
```
df.shape
```

[13]:
```
(569, 31)
```

[14]:
```
#Getting info about the data
```

[15]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
 30  label                    569 non-null    int32  
dtypes: float64(30), int32(1)
memory usage: 135.7 KB
```

[ ]:

[ ]:

[16]:
```
#Checking for missing values
df.isnull().sum()
```

[16]:
```
mean radius                0
mean texture               0
mean perimeter             0
mean area                  0
mean smoothness            0
mean compactness           0
mean concavity             0
mean concave points        0
mean symmetry              0
mean fractal dimension     0
radius error               0
texture error              0
perimeter error            0
area error                 0
smoothness error           0
compactness error          0
concavity error            0
concave points error       0
symmetry error             0
fractal dimension error    0
worst radius               0
worst texture              0
worst perimeter            0
worst area                 0
worst smoothness           0
worst compactness          0
worst concavity            0
worst concave points       0
worst symmetry             0
worst fractal dimension    0
label                      0
dtype: int64
```

[17]:
```
#Statistical measures about the data
```

[18]:
```
df.describe()
```

[18]:

	mean radius	mean texture	mean perimeter	mean area	\
count	569.000000	569.000000	569.000000	569.000000	
mean	14.127292	19.289649	91.969033	654.889104	
std	3.524049	4.301036	24.298981	351.914129	
min	6.981000	9.710000	43.790000	143.500000	
25%	11.700000	16.170000	75.170000	420.300000	
50%	13.370000	18.840000	86.240000	551.100000	
75%	15.780000	21.800000	104.100000	782.700000	
max	28.110000	39.280000	188.500000	2501.000000	

```
       mean smoothness  mean compactness  mean concavity  mean concave points  \
count       569.000000        569.000000      569.000000           569.000000
mean          0.096360          0.104341        0.088799             0.048919
std           0.014064          0.052813        0.079720             0.038803
min           0.052630          0.019380        0.000000             0.000000
25%           0.086370          0.064920        0.029560             0.020310
50%           0.095870          0.092630        0.061540             0.033500
75%           0.105300          0.130400        0.130700             0.074000
max           0.163400          0.345400        0.426800             0.201200

       mean symmetry  mean fractal dimension  …  worst texture  \
count     569.000000              569.000000  …     569.000000
mean        0.181162                0.062798  …      25.677223
std         0.027414                0.007060  …       6.146258
min         0.106000                0.049960  …      12.020000
25%         0.161900                0.057700  …      21.080000
50%         0.179200                0.061540  …      25.410000
75%         0.195700                0.066120  …      29.720000
max         0.304000                0.097440  …      49.540000

       worst perimeter   worst area  worst smoothness  worst compactness  \
count       569.000000   569.000000        569.000000         569.000000
mean        107.261213   880.583128          0.132369           0.254265
std          33.602542   569.356993          0.022832           0.157336
min          50.410000   185.200000          0.071170           0.027290
25%          84.110000   515.300000          0.116600           0.147200
50%          97.660000   686.500000          0.131300           0.211900
75%         125.400000  1084.000000          0.146000           0.339100
max         251.200000  4254.000000          0.222600           1.058000

       worst concavity  worst concave points  worst symmetry  \
count       569.000000            569.000000      569.000000
mean          0.272188              0.114606        0.290076
std           0.208624              0.065732        0.061867
min           0.000000              0.000000        0.156500
25%           0.114500              0.064930        0.250400
50%           0.226700              0.099930        0.282200
75%           0.382900              0.161400        0.317900
max           1.252000              0.291000        0.663800

       worst fractal dimension       label
count               569.000000  569.000000
mean                  0.083946    0.627417
std                   0.018061    0.483918
min                   0.055040    0.000000
25%                   0.071460    0.000000
50%                   0.080040    1.000000
```

```
75%                  0.092080    1.000000
max                  0.207500    1.000000

[8 rows x 31 columns]
```

[19]:
```
#Checking the distribution of target Variable
```

[20]:
```
df['label'].value_counts()
```

[20]:
```
label
1    357
0    212
Name: count, dtype: int64
```

[24]:
```
#1-->Benign
#0-->Malignant
```

[25]:
```
df.groupby('label').mean()
```

[25]:
```
       mean radius  mean texture  mean perimeter   mean area  mean smoothness  \
label
0        17.462830     21.604906      115.365377  978.376415         0.102898
1        12.146524     17.914762       78.075406  462.790196         0.092478

       mean compactness  mean concavity  mean concave points  mean symmetry  \
label
0              0.145188        0.160775             0.087990       0.192909
1              0.080085        0.046058             0.025717       0.174186

       mean fractal dimension  …  worst radius  worst texture  \
label                          …
0                    0.062680  …     21.134811      29.318208
1                    0.062867  …     13.379801      23.515070

       worst perimeter   worst area  worst smoothness  worst compactness  \
label
0           141.370330  1422.286321          0.144845           0.374824
1            87.005938   558.899440          0.124959           0.182673

       worst concavity  worst concave points  worst symmetry  \
label
0             0.450606              0.182237        0.323468
1             0.166238              0.074444        0.270246

       worst fractal dimension
label
0                     0.091530
```

```
1                    0.079442

[2 rows x 30 columns]
```

[27]:
```
#Seperating the features and target variable
```

[28]:
```
X=df.drop(columns='label',axis=1)
Y=df['label']
```

[29]:
```
print(X)
```

```
     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
..           …             …               …          …               …
564        21.56         22.39          142.00     1479.0          0.11100
565        20.13         28.25          131.20     1261.0          0.09780
566        16.60         28.08          108.30      858.1          0.08455
567        20.60         29.33          140.10     1265.0          0.11780
568         7.76         24.54           47.92      181.0          0.05263

     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
..                …               …                    …              …
564           0.11590         0.24390              0.13890         0.1726
565           0.10340         0.14400              0.09791         0.1752
566           0.10230         0.09251              0.05302         0.1590
567           0.27700         0.35140              0.15200         0.2397
568           0.04362         0.00000              0.00000         0.1587

     mean fractal dimension  …  worst radius  worst texture  \
0                   0.07871  …        25.380          17.33
1                   0.05667  …        24.990          23.41
2                   0.05999  …        23.570          25.53
3                   0.09744  …        14.910          26.50
4                   0.05883  …        22.540          16.67
..                      …  …           …              …
564                 0.05623  …        25.450          26.40
565                 0.05533  …        23.690          38.25
566                 0.05648  …        18.980          34.12
```

```
567                0.07016  …        25.740          39.42
568                0.05884  …         9.456          30.37

     worst perimeter  worst area  worst smoothness  worst compactness  \
0             184.60      2019.0           0.16220            0.66560
1             158.80      1956.0           0.12380            0.18660
2             152.50      1709.0           0.14440            0.42450
3              98.87       567.7           0.20980            0.86630
4             152.20      1575.0           0.13740            0.20500
..               …         …                …                 …
564           166.10      2027.0           0.14100            0.21130
565           155.00      1731.0           0.11660            0.19220
566           126.70      1124.0           0.11390            0.30940
567           184.60      1821.0           0.16500            0.86810
568            59.16       268.6           0.08996            0.06444

     worst concavity  worst concave points  worst symmetry  \
0             0.7119                0.2654          0.4601
1             0.2416                0.1860          0.2750
2             0.4504                0.2430          0.3613
3             0.6869                0.2575          0.6638
4             0.4000                0.1625          0.2364
..               …                   …              …
564           0.4107                0.2216          0.2060
565           0.3215                0.1628          0.2572
566           0.3403                0.1418          0.2218
567           0.9387                0.2650          0.4087
568           0.0000                0.0000          0.2871

     worst fractal dimension
0                    0.11890
1                    0.08902
2                    0.08758
3                    0.17300
4                    0.07678
..                       …
564                  0.07115
565                  0.06637
566                  0.07820
567                  0.12400
568                  0.07039

[569 rows x 30 columns]
```

[30]:
```
print(Y)
```

```
0      0
1      0
```

```
2      0
3      0
4      0
      ..
564    0
565    0
566    0
567    0
568    1
Name: label, Length: 569, dtype: int32
```

[31]:
```
#Splitting the data into training and testing data
```

[33]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=2)
```

[34]:
```
print(X.shape,X_train.shape,X_test.shape)
```

```
(569, 30) (455, 30) (114, 30)
```

[35]:
```
#Model Training
```

[36]:
```
#Logistic Regression
```

[39]:
```
model=LogisticRegression()
```

[40]:
```
#Training the model using the training data
```

[42]:
```
model.fit(X_train,Y_train)
```

```
D:\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:469:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[42]:
```
LogisticRegression()
```

[43]:
```
#Model Evaluation
```

[44]:
```
#Accuracy Score
```

[50]:
```
X_train_prediction=model.predict(X_train)
training_data_accuracy=accuracy_score(Y_train,X_train_prediction)
print("Accuracy on training data=",training_data_accuracy)
```

```
Accuracy on training data= 0.9494505494505494
```

[51]:
```
X_test_prediction=model.predict(X_test)
test_data_accuracy=accuracy_score(Y_test,X_test_prediction)
print("Accuracy on testing data=",test_data_accuracy)
```

```
Accuracy on testing data= 0.9210526315789473
```

[52]:
```
#Building a predictive system
```

[57]:
```
input_data=(18.25,19.98,119.6,1040,0.09463,0.109,0.1127,0.074,0.1794,0.05742,0.
↪4467,0.7732,3.18,53.91,0.004314,0.01382,0.02254,0.01039,0.01369,0.002179,22.
↪88,27.66,153.2,1606,0.1442,0.2576,0.3784,0.1932,0.3063,0.08368
)
#Changing input data into numpy array
inp=np.asarray(input_data)
#Reshape the input
input_reshaped=inp.reshape(1,-1)
prediction=model.predict(input_reshaped)
print(prediction)
if(prediction[0]==0):
    print("The Breast Cancer is malignant")
else:
    print("The Breast Cancer is Benign")
```

```
[0]
The Breast Cancer is malignant

D:\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have
valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
```

[59]:
```
input_data=(9.504,12.44,60.34,273.9,0.1024,0.06492,0.02956,0.02076,0.1815,0.
↪06905,0.2773,0.9768,1.909,15.7,0.009606,0.01432,0.01985,0.01421,0.02027,0.
↪002968,10.23,15.66,65.13,314.9,0.1324,0.1148,0.08867,0.06227,0.245,0.07773
)
#Changing input data into numpy array
inp=np.asarray(input_data)
#Reshape the input
input_reshaped=inp.reshape(1,-1)
prediction=model.predict(input_reshaped)
print(prediction)

if(prediction[0]==0):
```

```
    print("The Breast Cancer is malignant")
else:
    print("The Breast Cancer is Benign")
```

```
[1]
The Breast Cancer is Benign
```

```
D:\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have
valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
```

[ ]: