In [9]:

```python
"""
Importing the Dependencies
"""

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

"""Data Collection & Processing"""

# loading the data from sklearn
breast_cancer_dataset = sklearn.datasets.load_breast_cancer()

print(breast_cancer_dataset)

# loading the data to a data frame
data_frame = pd.DataFrame(breast_cancer_dataset.data, columns = breast_cancer_dataset.fe

# print the first 5 rows of the dataframe
data_frame.head()

# adding the 'target' column to the data frame
data_frame['label'] = breast_cancer_dataset.target

# print last 5 rows of the dataframe
data_frame.tail()

# number of rows and columns in the dataset
data_frame.shape

# getting some information about the data
data_frame.info()

# checking for missing values
data_frame.isnull().sum()

# statistical measures about the data
data_frame.describe()

# checking the distribution of Target Varibale
data_frame['label'].value_counts()

"""1 --> Benign

0 --> Malignant
"""

data_frame.groupby('label').mean()

"""Separating the features and target"""

X = data_frame.drop(columns='label', axis=1)
Y = data_frame['label']

print(X)

print(Y)

"""Splitting the data into training data & Testing data"""

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

```python
print(X.shape, X_train.shape, X_test.shape)

"""Model Training

Logistic Regression
"""

model = LogisticRegression()

# training the Logistic Regression model using Training data

model.fit(X_train, Y_train)

"""Model Evaluation

Accuracy Score
"""

# accuracy on training data
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 test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(Y_test, X_test_prediction)

print('Accuracy on test data = ', test_data_accuracy)

"""Building a Predictive System"""

input_data = (13.54,14.36,87.46,566.3,0.09779,0.08129,0.06664,0.04781,0.1885,0.05766,0.2

# change the input data to a numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the numpy array as we are predicting for one datapoint
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)
print(prediction)

if (prediction[0] == 0):
  print('The Breast cancer is Malignant')

else:
  print('The Breast Cancer is Benign')
```

```
{'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, 0, 0, 1, 0, 0, 0, 1, 0, 0,
        1, 1, 1, 1, 1, 0, 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, 0, 1, 1, 0,
        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, 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, 0, 0, 0, 0, 0, 0, 1]), 'frame': None, 'target
_names': array(['malignant', 'benign'], dtype='<U9'), 'DESCR': '.. _breast_cancer_datase
t:\n\nBreast cancer wisconsin (diagnostic) dataset\n-------------------------------------
--------\n\n**Data Set Characteristics:**\n\n    :Number of Instances: 569\n\n    :Numbe
r of Attributes: 30 numeric, predictive attributes and the class\n\n    :Attribute Infor
mation:\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 con
tour)\n        - concave points (number of concave portions of the contour)\n        - s
ymmetry\n        - fractal dimension ("coastline approximation" - 1)\n\n        The mea
n, standard error, and "worst" or largest (mean of the three\n        worst/largest valu
es) 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-Be
nign\n\n    :Summary Statistics:\n\n    ===================================== ====== ===
===\n                                           Min    Max\n    =======================
============= ====== ======\n    radius (mean):                        6.981  28.11\n
texture (mean):                       9.71   39.28\n    perimeter (mean):
      43.79  188.5\n    area (mean):                          143.5  2501.0\n    smoothn
ess (mean):                   0.053  0.163\n    compactness (mean):
0.019  0.345\n    concavity (mean):                     0.0    0.427\n    concave points
(mean):                0.0    0.201\n    symmetry (mean):                      0.106  0.
304\n    fractal dimension (mean):             0.05   0.097\n    radius (standard erro
r):              0.112  2.873\n    texture (standard error):             0.36   4.885\n
   perimeter (standard error):           0.757  21.98\n    area (standard error):
         6.802  542.2\n    smoothness (standard error):          0.002  0.031\n    compa
ctness (standard error):         0.002  0.135\n    concavity (standard error):
0.0    0.396\n    concave points (standard error):      0.0    0.053\n    symmetry (stan
dard error):             0.008  0.079\n    fractal dimension (standard error):   0.001
0.03\n    radius (worst):                       7.93   36.04\n    texture (worst):
               12.02  49.54\n    perimeter (worst):                    50.41  251.2\n
  area (worst):                         185.2  4254.0\n    smoothness (worst):
         0.071  0.223\n    compactness (worst):                  0.027  1.058\n    conca
vity (worst):                   0.0    1.252\n    concave points (worst):
0.0    0.291\n    symmetry (worst):                     0.156  0.664\n    fractal dimens
ion (worst):             0.055  0.208\n    ===================================== ====== =
=====\n\n    :Missing Attribute Values: None\n\n    :Class Distribution: 212 - Malignan
t, 357 - Benign\n\n    :Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangas
arian\n\n    :Donor: Nick Street\n\n    :Date: November, 1995\n\nThis is a copy of UCI M
L 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 d
escribed above was obtained using\nMultisurface Method-Tree (MSM-T) [K. P. Bennett, "Dec
```

```
ision Tree\nConstruction Via Linear Programming." Proceedings of the 4th\nMidwest Artifi
cial Intelligence and Cognitive Science Society,\npp. 97-101, 1992], a classification me
thod which uses linear\nprogramming to construct a decision tree.  Relevant features\nwe
re 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-dimen
sional space is that described in:\n[K. P. Bennett and O. L. Mangasarian: "Robust Linear
\nProgramming Discrimination of Two Linearly Inseparable Sets",\nOptimization Methods an
d Software 1, 1992, 23-34].\n\nThis database is also available through the UW CS ftp ser
ver:\n\nftp ftp.cs.wisc.edu\ncd math-prog/cpo-dataset/machine-learn/WDBC/\n\n.. topic::
References\n\n   - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extra
ction \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 Jos
e, CA, 1993.\n   - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnos
is and \n     prognosis via linear programming. Operations Research, 43(4), pages 570-57
7, \n     July-August 1995.\n   - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machi
ne learning techniques\n     to diagnose breast cancer from fine-needle aspirates. Cance
r Letters 77 (1994) \n     163-171.', 'feature_names': array(['mean radius', 'mean textu
re', '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'}
<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
```

```
     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]
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
(569, 30) (455, 30) (114, 30)
Accuracy on training data =  0.9384615384615385
Accuracy on test data =  0.9298245614035088
[1]
The Breast Cancer is Benign
```

```
C:\Users\vakap\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: Conver
genceWarning: 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(
C:\Users\vakap\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not
have valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
```

In [ ]:

In [ ]: