

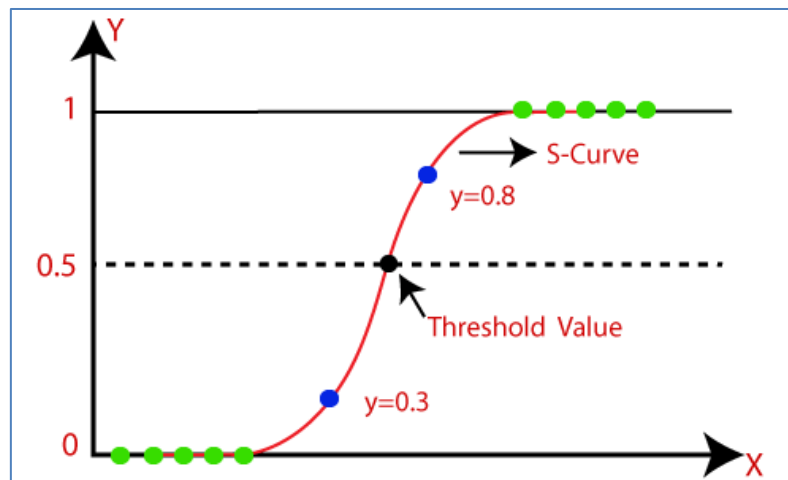
Experiment No: 02

● **Aim:** To implement Supervised Learning using Logistic regression algorithm.

● **Theory:**

Logistic regression

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

- We know the equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}; 0 \text{ for } y=0, \text{ and infinity for } y=1$$

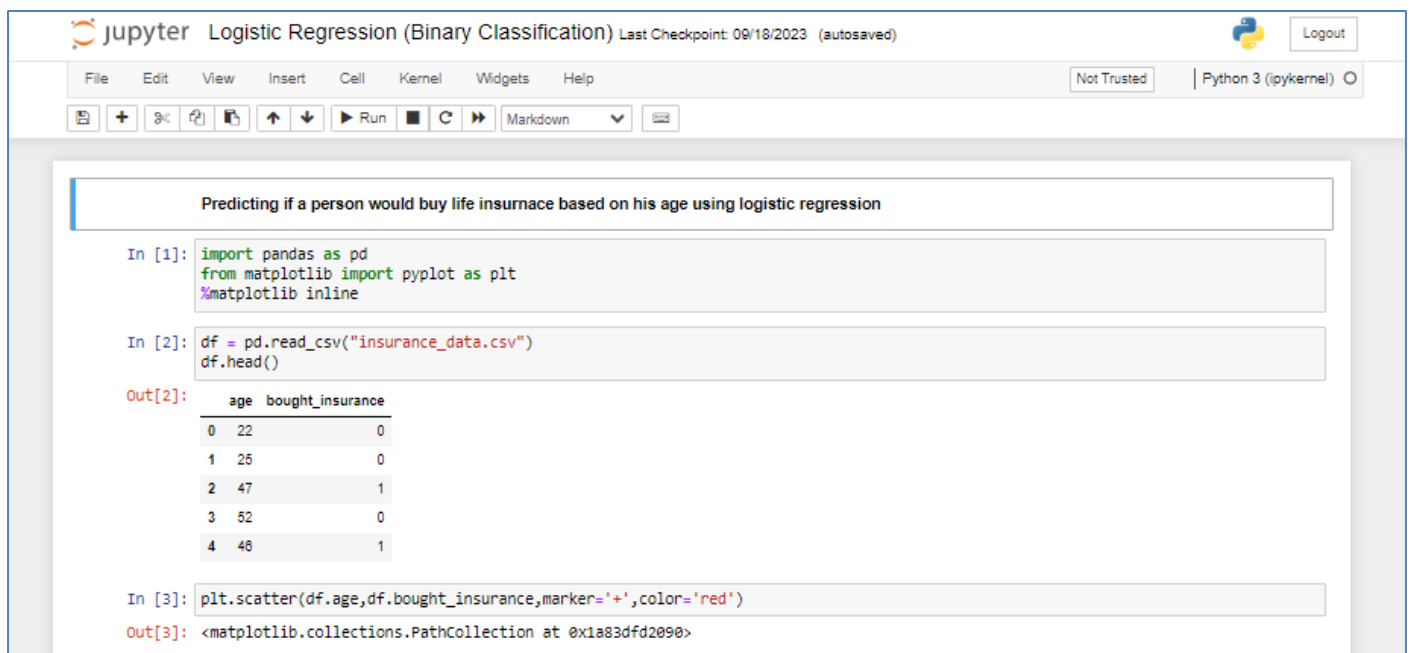
- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$\log \left[\frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

Type of Logistic Regression:

Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

Implementation of Logistic Regression (Binomial):



The screenshot shows a Jupyter Notebook titled "Logistic Regression (Binary Classification)" with a last checkpoint of 09/18/2023. The notebook contains the following code and output:

```

In [1]: import pandas as pd
        from matplotlib import pyplot as plt
        %matplotlib inline

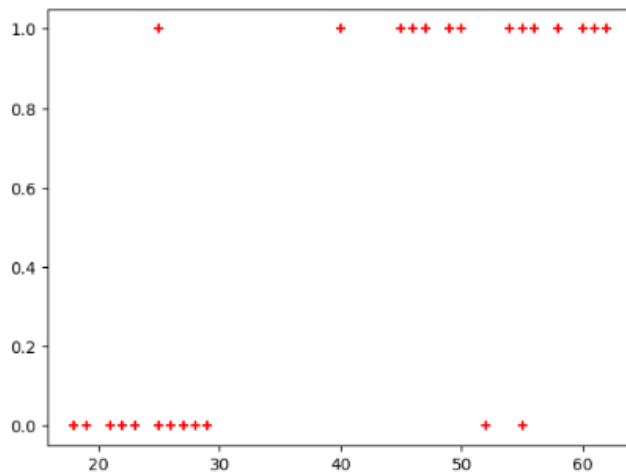
In [2]: df = pd.read_csv("insurance_data.csv")
        df.head()

Out[2]:
   age  bought_insurance
0   22                 0
1   25                 0
2   47                 1
3   52                 0
4   46                 1

In [3]: plt.scatter(df.age, df.bought_insurance, marker='+', color='red')

Out[3]: <matplotlib.collections.PathCollection at 0x1a83dfd2090>

```



```
In [13]: df.shape
Out[13]: (27, 2)

In [14]: from sklearn.model_selection import train_test_split

In [17]: X_train, X_test, y_train, y_test = train_test_split(df[['age']],df.bought_insurance,train_size=0.9)

In [18]: X_test
Out[18]:
```

```
In [18]: X_test
Out[18]:
   age
21  26
20  21
10  18

In [19]: from sklearn.linear_model import LogisticRegression

In [20]: model = LogisticRegression()

In [21]: model.fit(X_train, y_train)
Out[21]: LogisticRegression()
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [22]: model.predict(X_test)
Out[22]: array([0, 0, 0], dtype=int64)

In [23]: model.score(X_test,y_test)
Out[23]: 1.0

In [24]: model.predict_proba(X_test)
Out[24]: array([[0.81604245, 0.18395755],
                [0.89233197, 0.10766803],
                [0.92342366, 0.07657634]])
```

Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep".

Implementation of Logistic Regression (Multinomial):

```

jupyter Logistic Regression Multiclass Classification Last Checkpoint: 09/18/2023 (autosaved)
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

In [1]: from sklearn.datasets import load_digits
        %matplotlib inline
        import matplotlib.pyplot as plt
        digits = load_digits()

In [2]: digits = load_digits()

In [3]: dir(digits)
Out[3]: ['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_names']

In [4]: digits.data[0]
Out[4]: array([ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.,  0.,  0., 13., 15., 10.,
        15.,  5.,  0.,  0.,  3., 15.,  2.,  0., 11.,  8.,  0.,  0.,  4.,
        12.,  0.,  0.,  8.,  8.,  0.,  0.,  5.,  8.,  0.,  0.,  9.,  8.,
         0.,  0.,  4., 11.,  0.,  1., 12.,  7.,  0.,  0.,  2., 14.,  5.,
        10., 12.,  0.,  0.,  0.,  0.,  6., 13., 10.,  0.,  0.,  0.])

In [5]: plt.gray()
        for i in range(5):
            plt.matshow(digits.images[i])
  
```



```

In [6]: digits.target[0:5]
Out[6]: array([0, 1, 2, 3, 4])

In [7]: from sklearn.model_selection import train_test_split

In [8]: X_train, X_test, y_train, y_test = train_test_split(digits.data, digits.target, test_size=0.2)

In [9]: len(X_train)
Out[9]: 1437

In [10]: len(X_test)
Out[10]: 360
  
```

Create and train logistic regression model

```
In [11]: from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

```
In [12]: model.fit(X_train, y_train)
```

C:\ProgramData\anaconda3\Lib\site-packages\sklearn\linear_model_logistic.py:458: 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()

```
Out[12]: LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

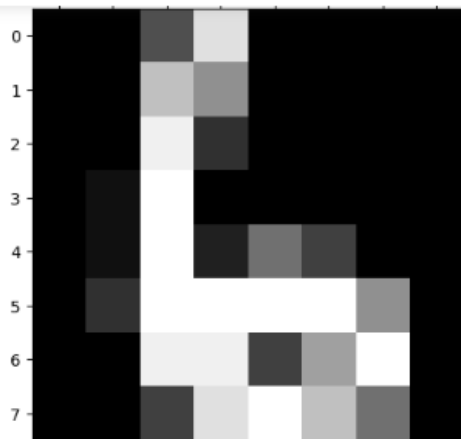
Measure accuracy of our model

```
In [13]: model.score(X_test, y_test)
```

```
Out[13]: 0.9611111111111111
```

```
In [14]: plt.matshow(digits.images[67])
```

```
Out[14]: <matplotlib.image.AxesImage at 0x2452a361190>
```



```
In [15]: digits.target[67]
```

```
Out[15]: 6
```

```
In [16]: model.predict([digits.data[67]])
```

```
Out[16]: array([6])
```

```
In [18]: model.predict(digits.data[0:5])
```

```
Out[18]: array([0, 1, 2, 3, 4])
```

Confusion Matrix

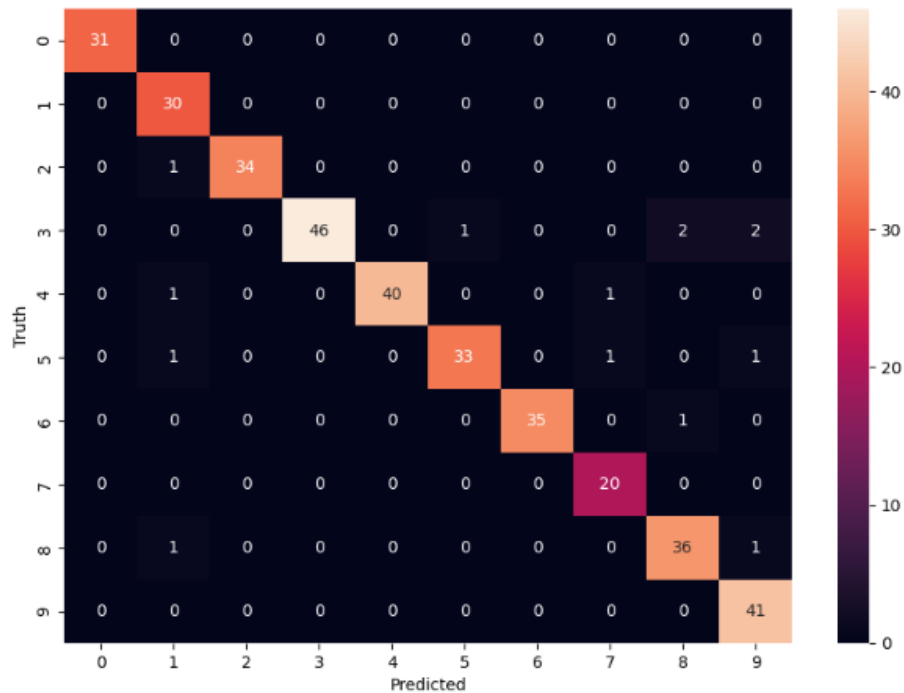
```
In [19]: y_predicted = model.predict(X_test)

In [20]: from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_predicted)
cm

Out[20]: array([[31,  0,  0,  0,  0,  0,  0,  0,  0,  0],
 [ 0, 30,  0,  0,  0,  0,  0,  0,  0,  0],
 [ 0,  1, 34,  0,  0,  0,  0,  0,  0,  0],
 [ 0,  0,  0, 46,  0,  1,  0,  0,  2,  2],
 [ 0,  1,  0,  0, 40,  0,  0,  1,  0,  0],
 [ 0,  1,  0,  0,  0, 33,  0,  1,  0,  1],
 [ 0,  0,  0,  0,  0,  0, 35,  0,  1,  0],
 [ 0,  0,  0,  0,  0,  0,  0, 20,  0,  0],
 [ 0,  1,  0,  0,  0,  0,  0,  0, 36,  1],
 [ 0,  0,  0,  0,  0,  0,  0,  0,  0, 41]], dtype=int64)

In [21]: import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')

Out[21]: Text(95.7222222222221, 0.5, 'Truth')
```



● Conclusion:

Logistic regression is widely used in various fields, including healthcare, finance, marketing, and more, for tasks such as spam detection, customer churn prediction, and medical diagnosis. It's a foundational algorithm in supervised learning for binary and multiclass classification.