## Experiment **No. 3**

Logistic Regression

**OBJECTIVE:**

* To get familiar with the concepts of Logistic Regression and Gradient Descent using python.
* To be able to use Logistic Regression for Regression Problems using python.

**Logistic Regression:**

Logistic regression is a supervised machine learning algorithm used for classification tasks where the goal is to predict the probability that an instance belongs to a given class or not. Logistic regression is a statistical algorithm which analyze the relationship between two data factors.

Logistic regression is used for binary classification where we use sigmoid function, that takes input as independent variables and produces a probability value between 0 and 1.

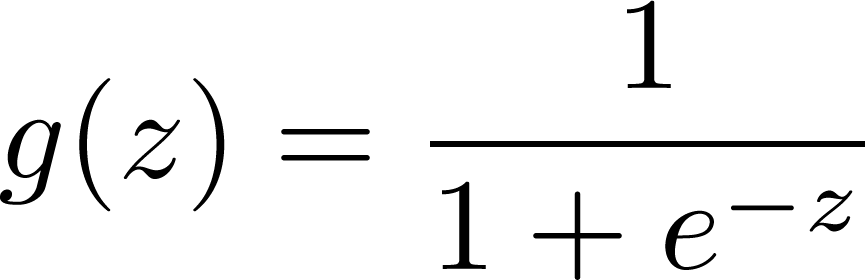
For example, we have two classes Class 0 and Class 1 if the value of the logistic function for an input is greater than 0.5 (threshold value) then it belongs to Class 1 it belongs to Class 0. It’s referred to as regression because it is the extension of linear regression but is mainly used for classification problems.

**Key-Points:**

* 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.
* In Logistic regression, instead of fitting a regression line, we fit an “S” shaped logistic function, which predicts two maximum values (0 or 1).

**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.



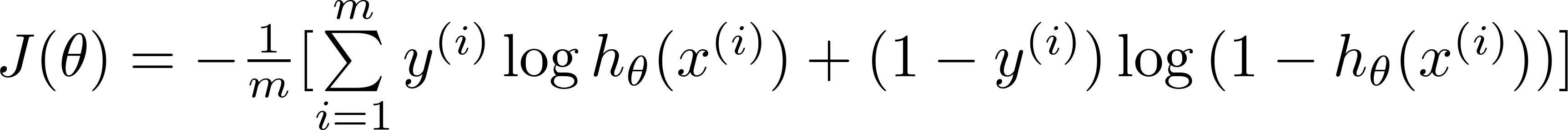
**Performing logistic regression using Gradient Descent:**

Performing logistic regression using gradient descent is a fundamental concept in machine learning and optimization. Gradient descent is an iterative optimization algorithm used to minimize a function by adjusting its parameters iteratively. In the context of logistic regression, the goal is to find the decision boundary for a given set of data points by minimizing the Error between the predicted values and the actual values.

Here's how it works:

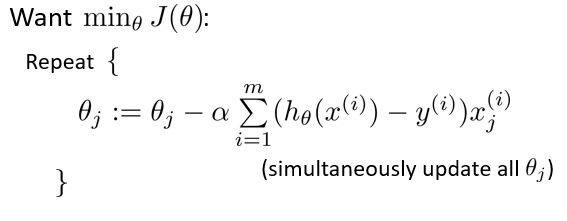
1. **Cost Function:**

The first step is to define a cost function (also known as the loss function or objective function) that measures the difference between the predicted values and the actual values. For logistic regression, the cost function is **Binary Cross-entropy function**:

****

1. **Gradient Descent Algorithm:**

The gradient descent algorithm aims to minimize the cost function by adjusting the parameters θ iteratively. The update rule for gradient descent is as follows:



**Implementation Steps:**

1. **Initialize Parameters:** Start with initial values for θ using numpy function random(random.rand() ).
2. **Compute Predictions:** Calculate  for all training examples.
3. **Compute Gradient:** Compute the gradient of the cost function with respect to θ.
4. **Update Parameters:** Update the weights using the gradient descent algorithm
5. **Iterate:** Repeat the above steps until the cost converges to a minimum or for a predefined number of iterations.
6. **Evaluate:** Use evaluation measures defined above to evaluate model.

**Linear Regression using Scikit-Learn:**

from sklearn.datasets import load\_iris

from sklearn.linear\_model import LogisticRegression

X, y = load\_iris(return\_X\_y=True)

clf = LogisticRegression(random\_state=0).fit(X, y)

clf.predict(X[:2, :])

clf.predict\_proba(X[:2, :])

clf.score(X, y)

**Exercise - 1**

Read the given data by using Pandas Library and write the python code (From Scratch) to perform the Logistic Regression with X1 and X2 as the Input variables and Y as the target. You need to do the following:

1. Use the given Cost function to optimize the values of the parameters on the training set.
2. Gradient Descent must be used for the back-propagation and update of the parameters where you can use the learning rate to be 0.02. Predict the values for the model on the test set.

**Training Set:**

|  |  |  |
| --- | --- | --- |
| **X1** | **X2** | **Y** |
| 60 | 22 | 0 |
| 62 | 25 | 0 |
| 67 | 24 | 0 |
| 70 | 20 | 0 |
| 71 | 15 | 1 |
| 72 | 14 | 1 |
| 75 | 14 | 1 |
| 78 | 11 | 1 |

**Test Set:**

|  |  |  |
| --- | --- | --- |
| **X1** | **X2** | **Y** |
| 61 | 23 | 0 |
| 71 | 19 | 0 |
| 73 | 15 | 1 |
| 79 | 13 | 1 |

**Exercise – 2**

Repeat Exercise-1 by changing the learning rate to 0.01, 0.001, and 0.1. What happens when learning rate is changed? Does it affect Learning?

**Answer:**

**Exercise – 3**

Use Built-in Library (Scikit-Learn) to do the task given in Exercise -1 and compare the results.