

## Experiment No.: 07

**Title:** Write a program for implementation of logistic regression.

**Objectives:** To learn logistic regression

**Theory:**

Logistic regression is a method for classifying data into discrete outcomes. For example, we might use logistic regression to classify an email as spam or not spam. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labeled "0" and "1". In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names. Analogous models with a different sigmoid function instead of the logistic function can also be used, such as the probit model; the defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each independent variable having its own parameter; for a binary dependent variable this generalizes the odds ratio.

### Hypothesis Representation

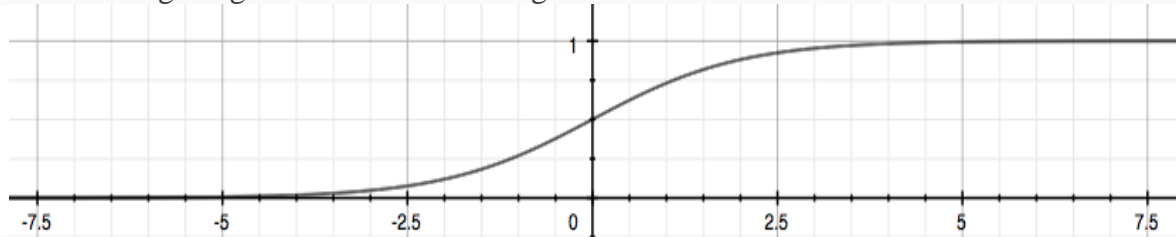
We could approach the classification problem ignoring the fact that  $y$  is discrete-valued, and use our old linear regression algorithm to try to predict  $y$  given  $x$ . However, it is easy to construct examples where this method performs very poorly. Intuitively, it also doesn't make sense for  $h_{\theta}(x)$  to take values larger than 1 or smaller than 0 when we know that  $y \in \{0, 1\}$ . To fix this, let's change the form for our hypotheses  $h_{\theta}(x)$  to satisfy  $0 \leq h_{\theta}(x) \leq 1$ . This is accomplished by plugging  $\theta^T \cdot x$  into the Logistic Function. Our new form uses the "Sigmoid Function," also called the "Logistic Function":

$$h_{\theta}(\mathbf{x}) = g(\theta^T \cdot \mathbf{x})$$

$$\mathbf{z} = \theta^T \cdot \mathbf{x}$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

The following image shows us what the sigmoid function looks like:



The function  $g(z)$ , shown here, maps any real number to the  $(0, 1)$  interval, making it useful for transforming an arbitrary-valued function into a function better suited for classification.

$h_{\theta}(\mathbf{x})$  will give us the **probability** that our output is 1. For example  $h_{\theta}(\mathbf{x}) = 0.7$  gives us a probability of 70% that our output is 1. Our probability that our prediction is 0 is just the complement of our probability that it is 1 (e.g. if probability that it is 1 is 70%, then the probability that it is 0 is 30%).

$$h_{\theta}(\mathbf{x}) = P(y=1 | \mathbf{x}; \theta) = 1 - P(y=0 | \mathbf{x}; \theta)$$

$$P(y=0 | \mathbf{x}; \theta) + P(y=1 | \mathbf{x}; \theta) = 1$$

#### Hypothesis Function

- Hypothesis Function for Logistic regression is,
- $h_{\theta}(\mathbf{x}) = g(\theta^T \mathbf{x})$

$$h_{\theta}(\mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}}$$

i.e.

#### Cost Function

- Cost Function for Logistic regression is

$$J(\theta) = -\frac{1}{m} \left[ \sum_{i=1}^m y^{(i)} \log h_{\theta}(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_{\theta}(x^{(i)})) \right]$$

#### Gradient Descent

- Gradient Descent Function for logistic regression is

Repeat {

$$\theta_j := \theta_j - \alpha \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

}

(simultaneously update all  $\theta_j$ )