Soft Computing:ISCO630E

Report

# Assignment 3.1

**Anant Chaturvedi(IIT2016506)**

## Question Description

Using the data set of two examination results design a predictor using logistic regression for predicting whether a student can get an admission in the institution. Use regularizer to further tune the parameters. Use 70 % data for training and rest 30% data for testing your predictor and calculate the efficiency of the predictor/hypothesis.

Hints: 1. You can pre process the data for convenience

2. You must use Python program for evaluating parameters using batch gradient descent algorithm (GDA). No function should be used for GDA.

## Introduction

We are given a dataset comprising of marks obtained by various students in two subjects and a binary result of 0 if the student was not admitted based on his marks and 1 if the student was admitted.

## Concepts Used

### Hypothesis Function

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x, \qquad X = \begin{bmatrix} — (x^{(1)})^T — \\ — (x^{(2)})^T — \\ \vdots \\ — (x^{(m)})^T — \end{bmatrix}. \qquad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}.$$

A Linear Regression model can be represented by the equation.

$$h(x) = \theta^T x$$

We then apply the sigmoid function to the output of the linear regression

$$h(x) = \sigma(\theta^T x)$$

where the sigmoid function is represented by,

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

The hypothesis for logistic regression then becomes,

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

$$h(x) = \begin{cases} > 0.5, & \text{if } \theta^T x > 0 \\ < 0.5, & \text{if } \theta^T x < 0 \end{cases}$$

If the weighted sum of inputs is greater than zero, the predicted class is 1 and vice-versa. So the decision boundary separating both the classes can be found by setting the weighted sum of inputs to 0.

## Cost Function(Regularized)

The cost function for a single training example can be given by:

$$cost = \begin{cases} -log(h(x), & \text{if } y = 1 \\ -log(1 - h(x)), & \text{if } y = 0 \end{cases}$$

We can combine both of the equations using:

$$cost(h(x), y) = -ylog(h(x)) - (1 - y)log(1 - h(x))$$

Then with regularization the cost function for m samples becomes

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

## Gradient Descent(Regularized)

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta).$$

$$\frac{\partial J(\theta)}{\partial\theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial\theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \qquad \text{for } j \geq 1$$

Where

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

α = Learning rate

The convergence arises when $J_t(\Theta)$ - $J_{t+1}(\Theta)$ <= ERROR_THRESHOLD

Since the cost function is convex in nature, the local minima obtained for the cost function would be the global minima.

## RESULTS

75% of the data was used for training while the remaining was used for testing the predictor.

The convergence appeared to be faster for

α =0.01

Error_Threshold=0.0000001

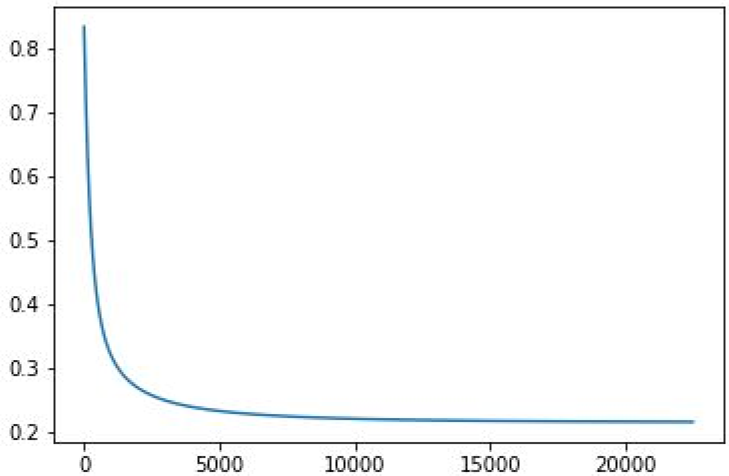
$\lambda$ =0.03

For the approach of Gradient Descent it took 22487

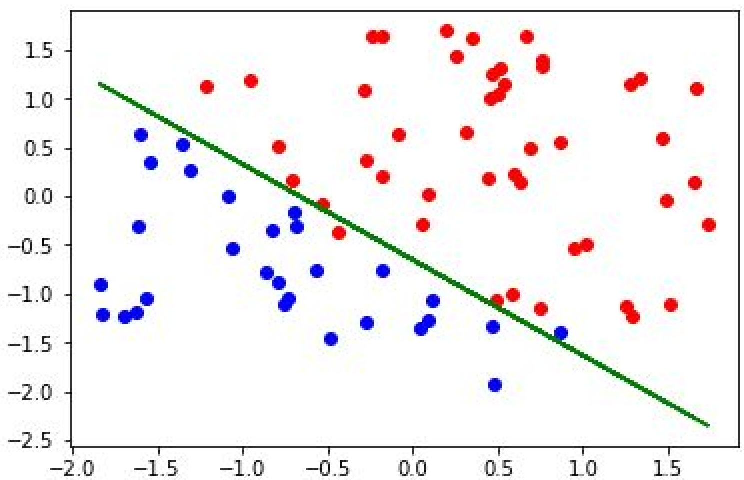
iterations to arrive at a final cost of 0.21437598380998046

The final theta values using Gradient Descent obtained were

Final theta values : [1.53007352 3.33438954 3.33394164]




The Decision Boundary along with the Data Points was obtained as follows




%accuracy for train data is 97.33333333333334

% accuracy for test data is 76.0