02.05.2017
Elif Ak – 150130009
Güneş Yurdakul - 150140141

Istanbul Technical University
Department of Computer Engineering

BLG 454E – Learning From Data
Homework 2

Solutions

Question - 1

The dataset consists of two features, which are given in the first 2 columns of the .mat file. The third column represents the labels, which are 0 and 1. This dataset can be classified logistic regression, since we have two classes.

Cost function

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

Vectorized cost function

$$h = g(X\theta)$$

$$J(\theta) = \frac{1}{m}(-y^T\log(h) - (1-y)^T\log(1-h))$$

Gradiant Descent

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

Vectorized Gradient Descent Function

$$\theta := \theta - \frac{\alpha}{m}X^T(g(X\theta) - y)$$

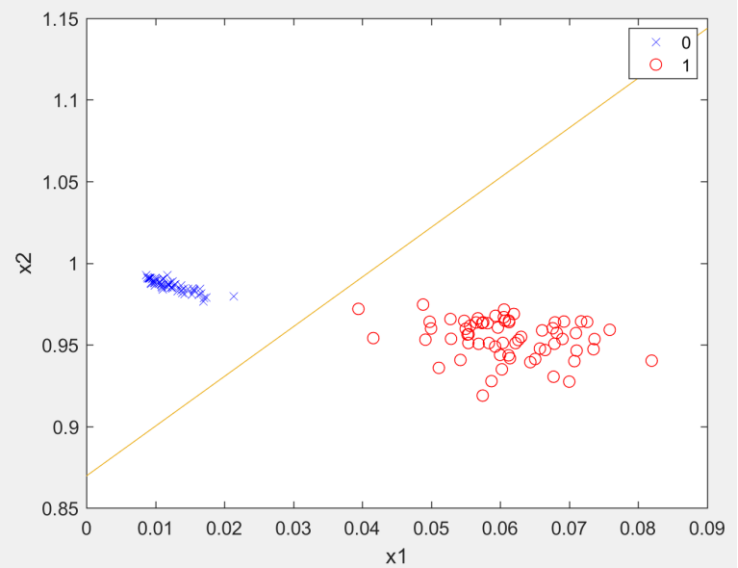
We should iterate over the gradient function while updating θ. If the cost function using updated theta decreases by less than 10^-7 in 1 iteration, the minima is found. Theta value which minimizes the cost function is what we are trying to calculate.

If we use the theta found using gradient descent function to calculate sigmoid function, we can succesfully classify new data. If the $h_\theta \geq 0.5$, y=1, else y=0.

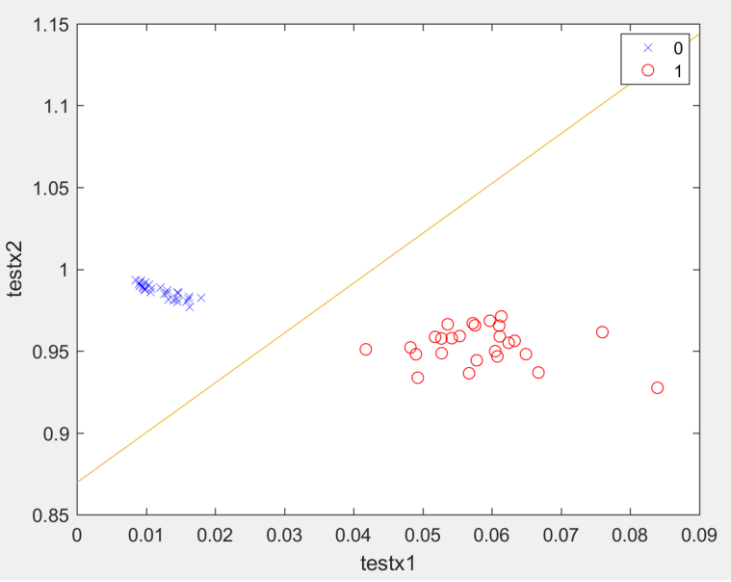
$$h_\theta = \frac{1}{1 + e^{-x\theta}}$$

Decision boundary is calculated using theta values too.

Desicion boundary: $\theta(1) + x_1\theta(2) + x_2\theta(3)$



Plot of the training dataset



Plot of the test dataset after classification using logistic regression

As it can be seen all data is classified correctly. Classification accuracy is 100%.

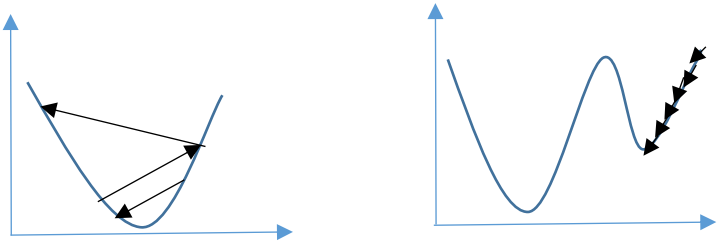
02.05.2017
Elif Ak – 150130009
Güneş Yurdakul - 150140141

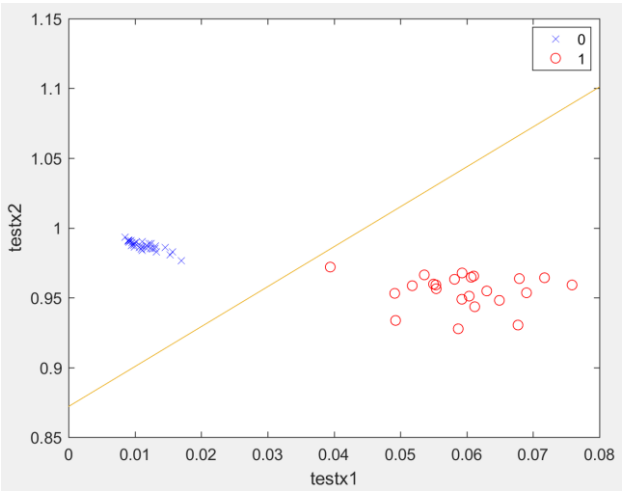
Question – 2

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

α determines the step size while approaching the minimum of cost function.
If α is too large, there is a risk of skipping the minima.
If α is too small, too many iterations will be needed to converge to minima. Also there is a risk to be trapped in local minima.

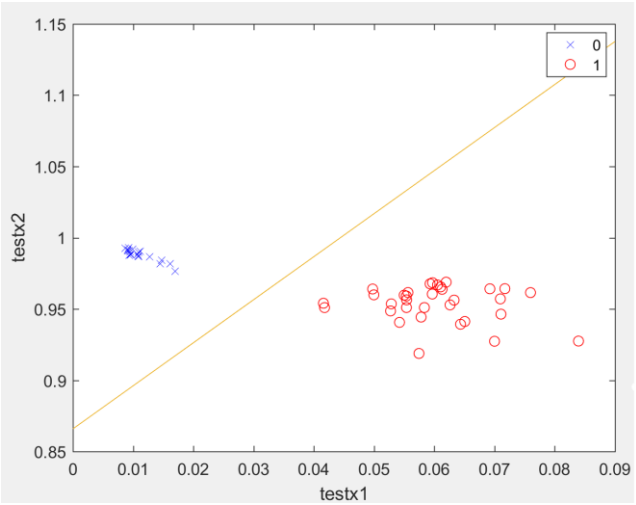
Large α Small α





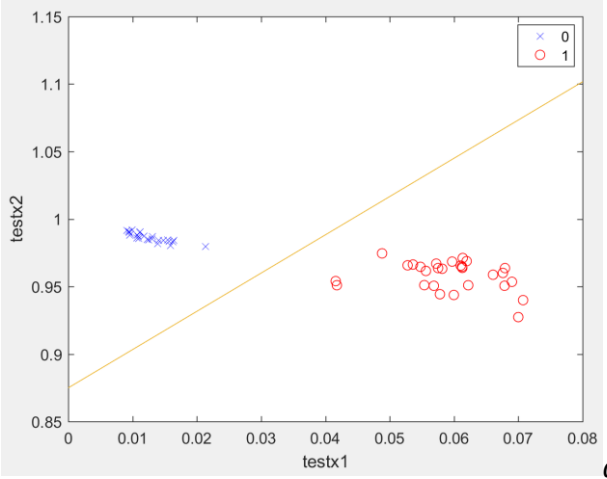
$\alpha = 0.5$

The classification is accurate(100%), but it took 195648 number of iterations to converge.



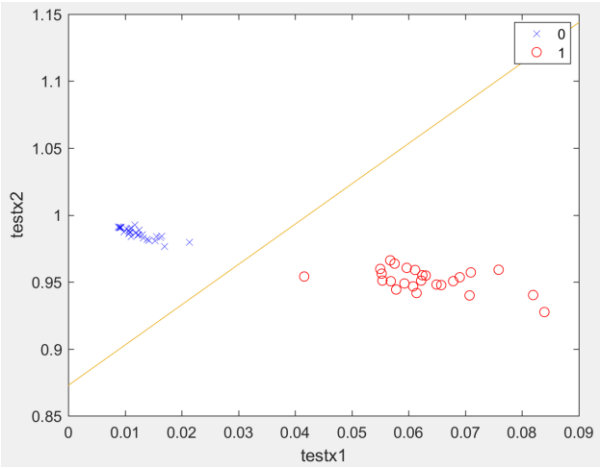
$\alpha = 1$

The classification is accurate, but it took 140862 number of iterations to converge.



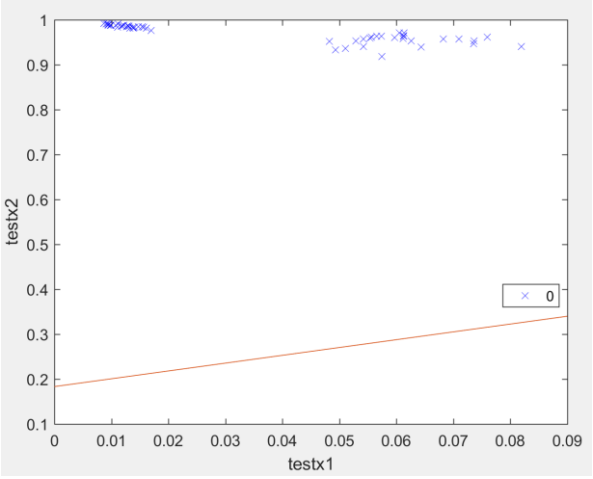
$\alpha = 3$

The classification is still accurate and in only 81821 number of iterations the cost function has converged.



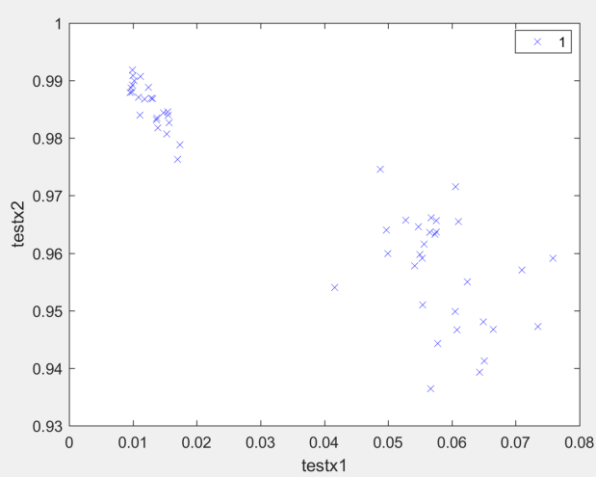
$\alpha = 4$

The classification is still accurate and in only 74777 number of iterations the cost function has converged.



$\alpha = 5$

The classification is not accurate. Since the learning rate is too large, the minima is skipped. Number of iterations:3



$\alpha = 10$

The classification is not accurate. Since the learning rate is too large, the minima is skipped.
Number of iterations:1

Question - 3

When it is asked to find sigmoid function or gradient descent update model for logistic regression where the number of classes are greater than two, we should solve the problem in case of logistic regression which means 1 or 0. So that for more classes, it is need to update formulations (cost function, sigmoid fuction etc.) under logistic regression logic. In order to do that for each class, logistic regression is thought as one and rest logic. For example, lets say there are 3 classes naming 1, 2 and 3. Regression is operated as 1 and others, 2 and others, 3 and other to find which one is correct. So, it is needed to k outcomes and probabilities of each of the outcomes such as $\phi_1, \phi_2, \dots, \phi_k$.

$\phi_i = p(y = i; \phi)$, and k^{th}class can be written as following because of probability summation will be 1 .

$$p(y = k; \phi) = 1 - \sum_{i=1}^{k-1} \phi_i$$

$T(y)$ is used to express the multinomial as an exponential family distribution. $T(y)_i$ denotes the i^{th} element of the vector $T(y)$.

Under the conditional probability description,

$$p(y; \phi) = \phi_1^{T(y)_1} \phi_1^{T(y)_1} \dots \phi_k^{1 - \sum_{i=1}^{k-1} T(y)_i}$$

$$= e^{T(y)_1 \log(\phi_1) + T(y)_2 \log(\phi_2) + \dots + \left(1 - \sum_{i=1}^{k-1} T(y)_i\right) \log(\phi_k)}$$

Note that $e^{\log(a)} = a$, in our formula a indicates $\phi_1^{T(y)_1}$ for the first element and it means that $e^{T(y)_1 \log(\phi_1)} = e^{\log\left(\phi_1^{T(y)_1}\right)} = \phi_1^{T(y)_1}$. Besides the explanations, continuous generalization in the following,

$$= e^{T(y)_1 \log\left(\frac{\phi_1}{\phi_k}\right) + T(y)_2 \log\left(\frac{\phi_2}{\phi_k}\right) + \dots + T(y)_{k-1} \log(\phi_{k-1}/\phi_k) + \log(\phi_k)}$$

In this step, simplification is done using *log* property which is $\log(a) - \log(b) = \log(a/b)$. And in our generalization step,

$$\left(1 - \sum_{i=1}^{k-1} T(y)_i\right) \log(\emptyset_k) = \log(\emptyset_k) - \sum_{i=1}^{k-1} T(y)_i \log(\emptyset_k)$$

So that summation cause following simplification for first element,

$$= e^{T(y)_1 \log(\emptyset_1) - T(y)_1 \log(\emptyset_k)} = e^{T(y)_1 \log(\emptyset_1/\emptyset_k)}$$

Vectorized implementation:

$$= e^{\eta^T T(y) + \log(\phi_k)}$$

Where,

$$\eta = \begin{bmatrix} \log\left(\frac{\phi_1}{\phi_k}\right) \\ \log\left(\frac{\phi_2}{\phi_k}\right) \\ . \\ . \\ . \\ \log\left(\frac{\phi_{k-1}}{\phi_k}\right) \end{bmatrix}$$

$$\eta_i = \log\left(\frac{\phi_i}{\phi_k}\right),\ e^{\eta_i} = \frac{\phi_i}{\phi_k}$$

$$\phi_k e^{\eta_i} = \phi_i$$

$$\phi_k \sum_{i=1}^{k} e^{\eta_i} = \sum_{i=1}^{k} \phi_i$$

And summation of probabilities will be one, which means that;

$$\sum_{i=1}^{k} \phi_i = 1 \; so \; that$$

$\phi_k \sum_{i=1}^{k} e^{\eta_i} = \sum_{i=1}^{k} \phi_i = 1$ and this implies that $\phi_k = 1/\sum_{i=1}^{k} e^{\eta_i}$

And we can substitute ϕ_k where $\phi_k e^{\eta_i} = \phi_i$ with $\phi_k = 1/\sum_{i=1}^{k} e^{\eta_i}$ so new equation will be

$$\boxed{\phi_i = \frac{e^{\eta_i}}{\sum_{i=1}^{k} e^{\eta_i}}}$$

If we back to the start point,

$$p(y = i|x;\theta) = \phi_i$$

$$= \frac{e^{\eta_i}}{\sum_{j=1}^{k} e^{\eta_j}}$$

$$= \frac{e^{\theta_i^T x}}{\sum_{j=1}^{k} e^{\theta_j^T x}}$$

Finally,

$$h_\theta(x) = \begin{bmatrix} \phi_1 \\ \phi_2 \\ . \\ . \\ . \\ \phi_{k-1} \end{bmatrix} = \begin{bmatrix} \frac{e^{\theta_1^T x}}{\sum_{j=1}^{k} e^{\theta_j^T x}} \\ \frac{e^{\theta_2^T x}}{\sum_{j=1}^{k} e^{\theta_j^T x}} \\ . \\ . \\ . \\ \frac{e^{\theta_{k-1}^T x}}{\sum_{j=1}^{k} e^{\theta_j^T x}} \end{bmatrix}$$

Note that this forms implies ***softmax regression*** model [1] or ***softmax function*** [2].

Cost function can be written as we used in *part 1*

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

$$= -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{1} 1\{y^{(i)} = j\}\log(p(y^{(i)} = j|x^{(i)};\theta))\right]$$

As you can see that, the softmax cost function will be similar, except that it includes j's 0 to *k which is number of classes. So that* ***softmax cost function is:***

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{k} 1\{y^{(i)} = j\}\log\left(p\left(y^{(i)} = j|x^{(i)};\theta\right)\right)\right]$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{k} 1\{y^{(i)} = j\}\log\left(\frac{e^{\theta_i^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\right)\right]$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}\left(1\{y^{(i)} = j\} - \log\left(\frac{e^{\theta_i^T x^{(i)}}}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\right)\right)\right]$$

As we know, our purpose is to minimize $J(\theta)$ and for each iteration we should update

$\theta_j := \theta_j - \alpha\frac{\partial J(\theta)}{\partial \theta_j}, \quad j = 1, \dots, k$. ***This is the update rule of the gradient descent for softmax regression (logistic regression with more than two classes).***

References

1. Anon. 2013. Softmax Regression. (April 2013). Retrieved May 2, 2017 from http://ufldl.stanford.edu/wiki/index.php/Softmax_Regression
2. Ethem Alpaydin. 2010. Introduction to Machine Learning (2nd ed.). The MIT Press.