UNIVERSITY OF VICTORIA
Department of Electrical and Computer Engineering
ECE 403/503 Optimization for Machine Learning
LABORATORY REPORT

Experiment No: 4
Title: Breast Cancer Diagnosis via Logistic Regression
Date of Experiment: 11/21/2019
Report Submitted on:11/28/2019
To: TA
Laboratory Group No.:
Name(s): Kurt Elliott & Raj Deepak S. N.

Objectives:

In this experiment, we investigate using a logistic regression to build a model for breast cancer patients that can classify new samples as either benign or malignant. Then a gradient descent algorithm with backtracking line search will be used to optimize the model for best results.

Introduction:

The logistic regression model is a regression technique that uses statistical probability to classify a model. The model classifies based on probability, if the values of the model is < 0.5 the model is classified as -1 (benign), if the model is > 0.5 the model is classified as 1(malignant). A Gradient descent algorithm is then applied to the model to find the optimum solution. A backtracking line search algorithm is also applied to the gradient descent to insure that it takes the most optimal path to the optimized solution of the logistic regression model.

The dataset to be used has 30 feature values from each of 569 patients of which 357 patients were diagnosed as benign while the other 212 patients were diagnosed as malignant.

Results:

The following is the MATLAB result from the script used in this lab, and a brief description of how the program is implemented. The script was sliced into sections for readability, therefore refer to appendix for full script.

The script used in this lab consists of three primary sections. The first section of the code is used to load and extract the training and testing datasets. The datasets consists of input vectors **x**, which consist of 30 components

(features), and **y** output vectors, which consist of 2 components (malignant & benign).

```
%% Step 1
% load datasets into matlab
load D_bc_te
load D_bc_tr

%% Step 2
% format testing and training datasets and normalization dataset

NumFeatures = 30;
K = 75;

Xtrain = zeros(30,480);
for i = 1:30
 xi = D_bc_tr(i,:);
 mi = mean(xi);
 vi = sqrt(var(xi));
 Xtrain(i,:) = (xi - mi)/vi;
end
Xtest = zeros(30,89);
for i = 1:30
 xi = D_bc_te(i,:);
 mi = mean(xi);
 vi = sqrt(var(xi));
 Xtest(i,:) = (xi - mi)/vi;
end

ytrain = D_bc_tr(31,:);
ytest = D_bc_te(31,:);

Dtrain = [Xtrain;ytrain];
Dtest = [Xtest;ytest];
```

The next step is to build the model, this is done through two functions that were made in matlab. First function is the “objective function” (refer to lab manual E4.2).

```
function f = f_wcdc(w,D)
X = D(1:30,:);
y = D(31,:);
N = length(y);
f = 0;
wt = w(:)';
for i = 1:N
    xi = [X(:,i);1];
    fi = log(1+exp(-y(i)*(wt*xi)));
    f = f+fi;
end
f = f/N;
```

The second function is the gradient of the objective function (refer to lab manual E4.3).

```
function g = g_wcdc(w,D)
X = D(1:30,:);
y = D(31,:);
N = length(y);
g = zeros(31,1);
wt = w(:)';
for i = 1:N
    xi = [X(:,i);1];
    ei = exp(-y(i)*(wt*xi));
    ci = -y(i)*ei/(1+ei);
    gi = ci*xi;
    g = g+gi;
end
g = g/N;
```

Next is the model training section. The data is formatted into matrices and the learning rate and the direction of the gradient descent are obtained. Than the model is trained to a episonal threshold.

```
%% Step 4
% Minimize f(wh)
eps = 10^-9;
w = zeros(NumFeatures+1,1);
f = zeros(1,K);
k = 1;

% step 2: find d
gk = g_wcdc(w,Dtrain);
dk = -gk;

% step 3: find alpha
alpha = bt_lsearch(w,dk,'f_wcdc','g_wcdc',Dtrain);

%% Training
while ((norm(alpha*dk) >= eps) && (k<+K))

    w = w + alpha*dk;

    % step 2: find d
    gk = g_wcdc(w,Dtrain);
    dk = -gk;

    % step 3: find alpha
    alpha = bt_lsearch(w,dk,'f_wcdc','g_wcdc',Dtrain);

    %check if f is decreasing.
    f(k) = f_wcdc(w,Dtrain);

    k = k + 1;
end
```

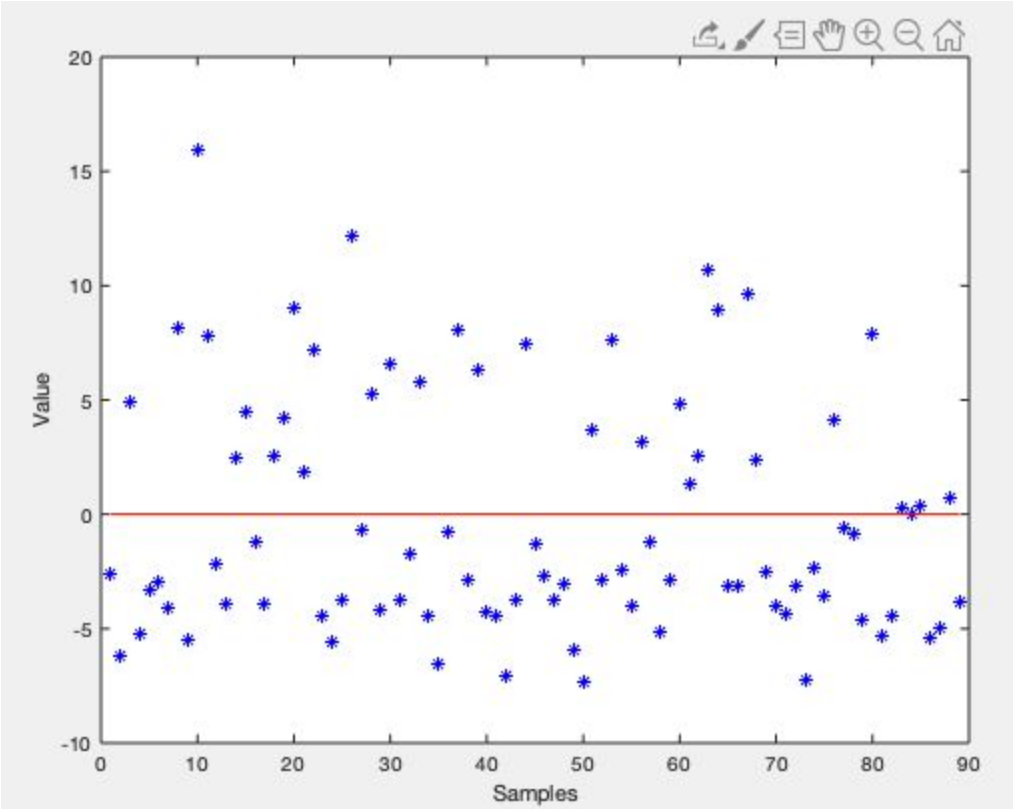
The final part is testing the logistic regression model by comparing prediction values with the test model prediction in the test dataset. This was done by checking the predictions label, than checking it will the tests label and counting the false positive and false negatives. A misclassification ratio was than obtained by adding the false positives and negatives together and dividing by 100.

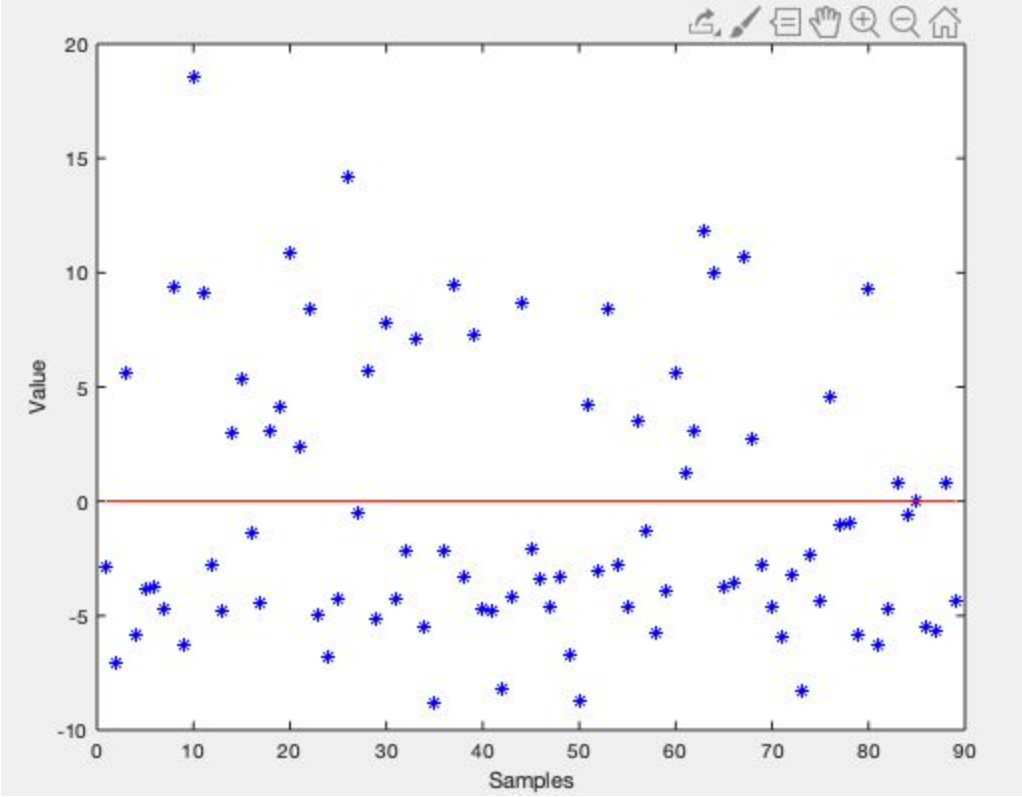
```
%% Testing
Dt = [Xtest; ones(1,89)];
Result = w'*Dt;
TestLabel = zeros(1,length(Result));
FalsePos = 0;
FalseNeg = 0;

if PLOT == 1
    AfterTr = Result;
    Z = zeros(1,length(AfterTr));

    figure, plot(AfterTr,'b*');
    hold on
    plot(Z,'r-');
    xlabel('Samples');
    ylabel('Value');
end

for ii = 1:length(Result)
    if Result(ii) < 0
        TestLabel(ii) = -1;
        if ytest(ii) > 0
            FalseNeg = FalseNeg + 1;
        end
    else
        TestLabel(ii) = 1;
        if ytest(ii) > 0
            FalsePos = FalsePos + 1;
        end
    end
end
```

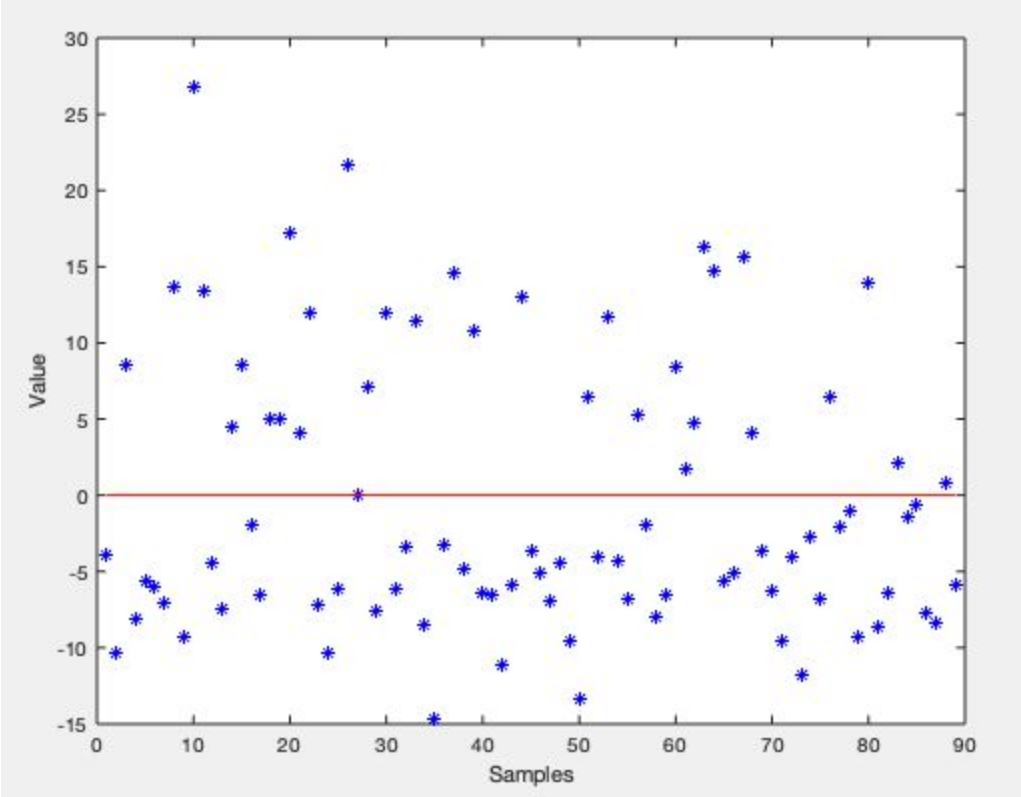
Discussion:

After training the model, analysis was done on the testing dataset and the results from the true values and the model predicted values were plotted. This procedure was done 3 times to evaluate different iteration values K=5, K=12, K=75. The different iteration values gave the following results for false positives and false negatives and are plotted below.

K-Iterations	K=5	K=12	K=75
False positive	31	31	32
False negative	1	1	0
mis-classification Ratio	0.32	0.32	0.32




Value
Samples
20
15
10
5
0
-5
-10
0
10
20
30
40
50
60
70
80
90


Value
Samples
30
25
20
15
10
5
0
-5
-10
-15
0
10
20
30
40
50
60
70
80
90

Conclusions:

The logistic regression classification model with gradient descent and backtracking line search for optimization was investigated for breast cancer diagnosis. The results show that all the points above and below the line are cases of benign and malignant respectively, with misclassified points on or in very close proximity to the line. The model is 68% efficient in separating benign and malignant cases. It has a high number of false positive results which means that a high number of patients are diagnosed with breast cancer when in actuality it is not true.

Appendix:

Table of Contents

Step 1

load datasets into matlab

```
load D_bc_te
load D_bc_tr
```

```
Error using evalin
Unrecognized function or variable 'experiment3'.
```

Step 2

format testing and training datasets and normalization dataset

```
NumFeatures = 30;
K = 75;

Xtrain = zeros(30,480);
for i = 1:30
 xi = D_bc_tr(i,:);
 mi = mean(xi);
 vi = sqrt(var(xi));
 Xtrain(i,:) = (xi - mi)/vi;
end
Xtest = zeros(30,89);
for i = 1:30
 xi = D_bc_te(i,:);
 mi = mean(xi);
 vi = sqrt(var(xi));
 Xtest(i,:) = (xi - mi)/vi;
end

ytrain = D_bc_tr(31,:);
ytest = D_bc_te(31,:);

Dtrain = [Xtrain;ytrain];
Dtest = [Xtest;ytest];
```

Step 3

Objective function and gradient