UNIVERSITY OF VICTORIA

Department of Electrical and Computer Engineering

ECE 403/503 Optimization for Machine Learning

LABORATORY REPORT

Experiment No: 04

Title: Breast Cancer Diagnosis via Logistic Regression

Date of Experiment: 16 July, 2019

Report Submitted on: 23 July, 2019

To: Mr. J. Zhan

Laboratory Group No.: B01 T 12

Name(s): Alvi Mahadi (V00912845)

1 Introduction and Objectives

The goal of this experiment is to develop a computer program for automatic diagnosis of breast cancer based on logistic regression where the logistic loss function is defined by a dataset provided by Dr. Wolberg from General Surgery Department, University of Wisconsin, Madison, WI in 1990s. The dataset contains 30 carefully selected features from each of 569 patients [R1],[R2]. The same dataset has also been made available from the UCI Machine Learning Repository [R3]. The parameters in the logistic regression model are optimized by minimizing the logistic loss function mentioned above. In this experiment, the optimization is performed using the gradient descent (GD) algorithm.

2 Implementation Steps and Results

2.1 Implementation Steps

We strictly followed the implementation steps stated in the laboratory manual [3].

2.2 Code

2.2.1 main.m

```
clc;
clear all;
close all;
D_bc_tr = load('/home/alvi/Documents/courses/ece503/labs/4/data/D_bc_tr.mat');
D_bc_tr = D_bc_tr.D_bc_tr;
D_bc_te = load('/home/alvi/Documents/courses/ece503/labs/4/data/D_bc_te.mat');
D_bc_te = D_bc_te.D_bc_te;
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,:);
w = zeros(30, 1);
b = 0;
w_hat = [w; b];
[xs,fs,k] = grad_desc('evaluate', 'gradient', w_hat, 5);
w_star = xs(1:30, :);
b_star = xs(31, :);
r = w_star' * Xtest + b_star;
false_positive = 0;
false_negetive = 0;
```

```
for i=1:89
if r(i) > 0
r(i) = 1;
elseif r(i) < 0
r(i) = -1;
end
if r(i) == 1 && r(i) ~= ytest(i)
false_positive = false_positive + 1;
elseif r(i) == -1 && r(i) ~= ytest(i)
false_negetive = false_negetive + 1;
end
end
error_rate = (false_positive + false_negetive) / 89 * 100;
disp('Result:');
fprintf('False Positive: %d\n', false_positive);
fprintf('Flase Negetive: %d\n', false_negetive);
fprintf('Error Rate: %f\n', error_rate);
```

2.2.2 evaluate.m

```
function f_w_hat = evaluate(w_hat)
        sum = 0;
        w_hat_transpose = w_hat';
        for i=1:480
        sum = sum + log(1 + exp((-ytrain(i) * w_hat_transpose * [Xtrain(:, i); 1
        end
        f_w_hat = sum / 480;
end
```

2.2.3 gradient.m

```
function g_w_hat = gradient(w_hat)
        w_hat_transpose = w_hat';
        for i=1:480
        calc_e = exp((ytrain(i) * w_hat_transpose * [Xtrain(:, i); 1]));
        sum = sum + (((ytrain(i) * calc_e) / (1 + calc_e)) * [Xtrain(:, i); 1]);
        end
        g_w_hat = -(sum / 480);
end
```

2.2.4 bt_lsearch.m

```
function a = bt_lsearch(x,s,F,G,p1,p2)
rho = 0.1;
gma = 0.5;
x = x(:);
s = s(:);
a = 1;
parameterstring ='';
% evaluate given parameters:
if nargin > 4,
if ischar(p1),
eval([p1 ';']);
else
```

```
parameterstring = ',p1';
end
end
if nargin > 5,
if ischar(p2),
eval([p2 ';']);
else
parameterstring = ',p2';
end
end
eval(['f0 = ' F '(x' parameterstring ');']);
eval(['g0 = ' G '(x' parameterstring ');']);
eval(['f1 = ' F '(x+a*s' parameterstring ');']);
f2 = f0 + rho*a*g0'*s;
er = f1 - f2;
while er > 0,
a = gma*a;
eval(['f1 = ' F '(x+a*s' parameterstring ');']);
f2 = f0 + rho*a*g0'*s;
er = f1 - f2;
end
if a < 1e-5,
a = 1e-5;
end
```

2.2.5 grad_desc.m

```
function [xs,fs,k] = grad_desc(fname,gname,x0,iterations)
format compact
format long
k = 1;
xk = x0;
gk = feval(gname, xk);
dk = -gk;
ak = bt_lsearch(xk,dk,fname,gname);
adk = ak*dk;
er = norm(adk);
for i=1:iterations
xk = xk + adk;
gk = feval(gname, xk);
dk = -gk;
ak = bt_lsearch(xk,dk,fname,gname);
adk = ak*dk;
er = norm(adk);
k = k + 1;
end
disp('solution:')
xs = xk + adk
disp('objective function at solution point:')
fs = feval(fname,xs)
format short
disp('number of iterations performed:')
k
```

2.3 Result

2.3.1 With K = 5

```
solution:
xs =
0.352638158468680
0.200203783957053
0.357948610209381
0.342299989673613
0.170708431987164
0.282443394937561
0.334037083162259
0.370899102676686
0.159603010445805
-0.004005604537492
0.266588609216825
-0.007723394236093
0.262505274824104
0.255681970225705
-0.033418042294543
0.151270526630695
0.135364512734198
0.200045137626640
-0.004182486405513
0.047971009456038
0.374238579225147
0.219405094580550
0.377049862171745
0.352804045330162
0.201982487316283
0.282912278788012
0.327028690537772
0.381015764530887
0.201311466653739
0.162042714197976
-0.125011175189046
objective function at solution point:
fs =
0.173566975819436
number of iterations performed:
k =
6
Result:
False Positive: 5
Flase Negetive: 2
Error Rate: 7.865169
```

2.3.2 With K = 12

```
solution:
xs =
0.352685501381469
```

0.200229189204410
0.357996943064542
0.342345947169986
0.170733666980612
0.282485669804677
0.334085248868133
0.370950956306149
0.159626964760263
−0.004001319412531
0.266624992597048
−0.007723790379885
0.262541534800715
0.255716608251548
−0.033421242003362
0.151296422534375
0.135388311108996
0.200076739735935
−0.004181085511795
0.047983744594538
0.374288593876488
0.219432620613280
0.377100596660160
0.352851231347517
0.202010346523521
0.282953226857858
0.327075421999848
0.381068802548124
0.201338892106307
0.162067828089840
−0.125026820762492
objective **function** at solution point:
fs =
0.173566094379326
number of iterations performed:
k =
13
Result:
False Positive: 5
Flase Negetive: 2
Error Rate: 7.865169

2.3.3 With k = 75

solution:
xs =
0.353111600599014
0.200457841401345
0.358431952470772
0.342759577677441
0.170960792412366
0.282866161794980
0.334518760391680
0.371417658470545

```
0.159842566747676
-0.003962743444460
0.266952454698131
-0.007727355050513
0.262867886676614
0.256028360766798
-0.033450034480577
0.151529500734340
0.135602509784567
0.200361173273865
-0.004168471259539
0.048098373742992
0.374738740419515
0.219680360873175
0.377557222333867
0.353275919536251
0.202261092185210
0.283321777086164
0.327496024623878
0.381546164551077
0.201585734692385
0.162293868755673
-0.125167647116726
objective function at solution point:
fs =
0.173558369319528
number of iterations performed:
k =
76
Result:
False Positive: 5
Flase Negetive: 2
Error Rate: 7.865169
```

3 References

[1] W. N. Street, W. H. Wolberg, and O. L. Mangasarian, Nuclear feature extraction for breast tumor diagnosis, in IS?T/SPIE Int. Symp. Electronic Imaging: Science and Technology, vol. 1905, pp. 861-870, San Jose, CA., 1993.

[2] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, Breast cancer diagnosis and prognosis via linear programming, AAAI Tech. Report SS-94-01, 1994.

[3] UCI Machine Learning, http://archive.ics.uci.edu/ml, University of California Irvine, School of Information and Computer Science.

[4] LABORATORY MANUAL, ECE 403/503 - OPTIMIZATION for MACHINE LEARNING, Prepared by: Wu-Sheng Lu, Department of Electrical and Computer Engineering, University of Victoria.