

Logistic Regression Problem Instructions

Problem 1. The data comprises data related to studies on real participants in relation to heart disease.

The data was cleansed in STATA resulting in the original data of 4,239 data entries being reduced to 3,749.

Two datasets of 3 features each were selected after examining plots of the data:

- HeartData_01 – selected features are somewhat correlated
- HeartData_02 – selected features are somewhat correlated

Prevalent hypertension (prevalentHyp) was chosen as the predictor.

- a) Find the probability of prevalent hypertension for the following patients using HeartData_01 and HeartData_02:

diaBp	BMI	heartRate	prevalentHyp
140	33	95	
80	27	70	
60	21	65	

Note: Below are the steps that were followed in MATLAB to accomplish this assignment. The results are shown under the section titled, Results, published by the MATLAB code.

- 1) Initialize the weights at $t = 0$ to $\mathbf{w}(0)$.
- 2) For $t = 0, 1, 2, \dots$ do the following
- 3) Compute the gradient using the equation below.

$$\nabla E_{in} = -\frac{1}{N} \sum_{n=1}^N \frac{y_n X_n}{1 + e^{(y_n \mathbf{w}^T(t) X_n)}}$$

- 4) Update the weights: $\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \frac{\nabla E_{in}}{\|\nabla E_{in}\|}$
- 5) Iterate to the next step until it is time to stop.
- 6) Return the final weights \mathbf{w} .
- 7) Compute the sigmoid function (below) to obtain a value between 0 and 1, which can be used as probability.

$$P(y|x) = \theta(y\mathbf{w}^T x) , \text{Bayesian Estimation}$$

$$\theta(s) = \frac{1}{1 + e^{-s}} , \text{sigmoid function}$$

Logistic Regression Problem 1

Table of Contents

Importing Data from Excel File and Plotting the Data	1
Logistic Regression Algorithm	3
Initializing the Weights	3
Computing the Gradient and Updating the Weights	3
Finding the Probability Using the Sigmoid Function	4
Results	5
Functions	6
Weight Update Function	6
Excel Import File Function	6

Importing Data from Excel File and Plotting the Data

The excel data file with **patients** having certain conditions in relation to heart disease that was used for this assignment was provided by the instructor

```
% Getting the Data from the HeartData Excel Sheet
% diaBP = diastolic blood pressure
% BMI = body mass index
% prevalentHyp = prevalent hypertension
[diaBP, BMI, heartRate, prevalentHyp] =
    importfile('HeartData_01.xlsx');

% creating a column vector of ones to be multiplied to w0 when doing
% matrix multiplication
heart_x0 = ones(3749,1);

% concatenation of the data into one matrix
x = [heart_x0, diaBP, BMI, heartRate];
Y = prevalentHyp; % passing prevalent hypertension to Y

% initializing column vectors to save values if hypertension is
% positive.
% the parameters below will be plot in red, so they can be identified
% as positive hypertension
diaBP_pos = zeros(1,1);
BMI_pos = zeros(1,1);
heartRate_pos = zeros(1,1);

% initializing column vectors to save values if hypertension is
% negative
% the parameters below will be plot in blue, so they can be identified
% as negative hypertension
diaBP_neg = zeros(1,1);
BMI_neg = zeros(1,1);
heartRate_neg = zeros(1,1);
```

```

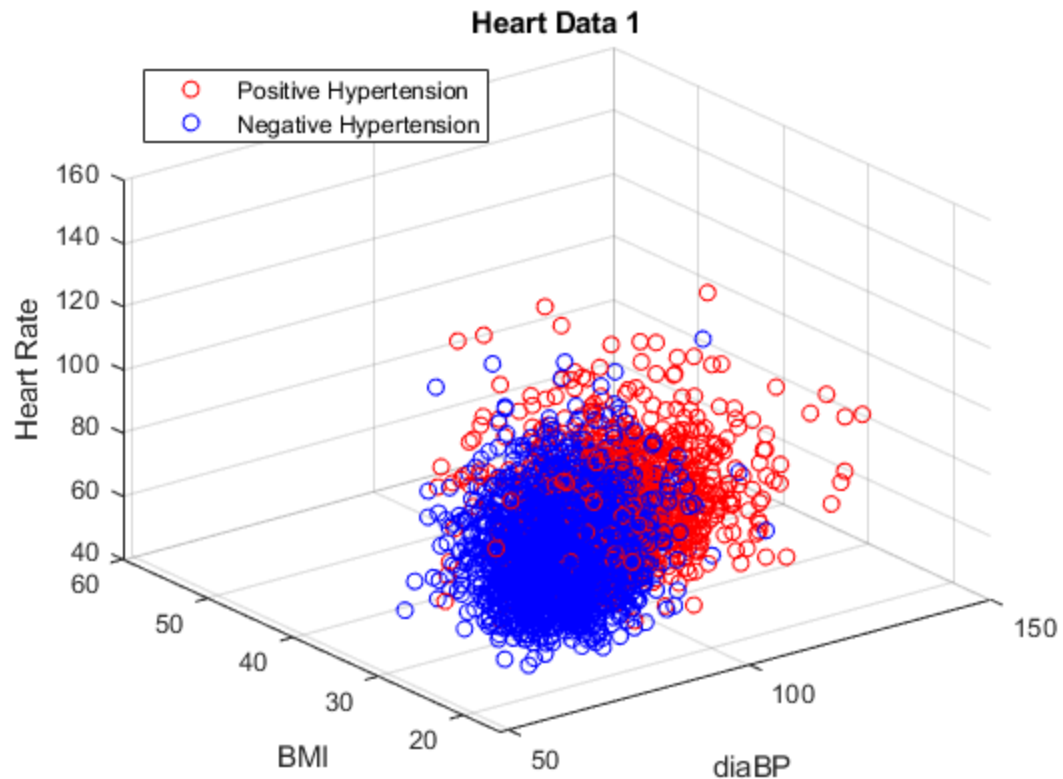
%y_pos = zeros(1,1);
%y_neg = zeros(1,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Separating the Data: Positive and Negative Hypertension
% for loop below separates the data in respect to positive
% hypertension
% or negative hypertension
j = 0;
j_n = 0;
for i = 1:3749

    if prevalentHyp(i) > 0
        j = j + 1;
        diaBP_pos(j) = diaBP(i);
        BMI_pos(j) = BMI(i);
        heartRate_pos(j) = heartRate(i);
        Y(i) = prevalentHyp(i);
    else
        j_n = j_n + 1;
        diaBP_neg(j_n) = diaBP(i);
        BMI_neg(j_n) = BMI(i);
        heartRate_neg(j_n) = heartRate(i);
        Y(i) = -1;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
% figure below plots the data onto a 3D plane with red dots indicating
% positive hypertension and blue dots indicating negative hypertension
figure(1)
scatter3(diaBP_pos, BMI_pos, heartRate_pos, 'red')
hold on
scatter3(diaBP_neg, BMI_neg, heartRate_neg, 'blue')
title('Heart Data 1')
xlabel('diaBP');
ylabel('BMI');
zlabel('Heart Rate');
legend('Positive Hypertension','Negative
Hypertension', 'location','nw');
hold off

```



Logistic Regression Algorithm

Initializing the Weights

```
w_0 = 0;  
w_1 = 0;  
w_2 = 0;  
w_3 = 0;  
w = [w_0; w_1; w_2; w_3];  
  
N = 3749; % number of samples per iteration  
xt = transpose(x); % transpose of x
```

Computing the Gradient and Updating the Weights

```
% the forloop below computes the gradient and updates the weights  
for t = 1:1000 % number of iterations  
  
    gradi = (zeros(4,1));  
    new_gradiw = (zeros(1,4));  
    for i = 1:3749
```

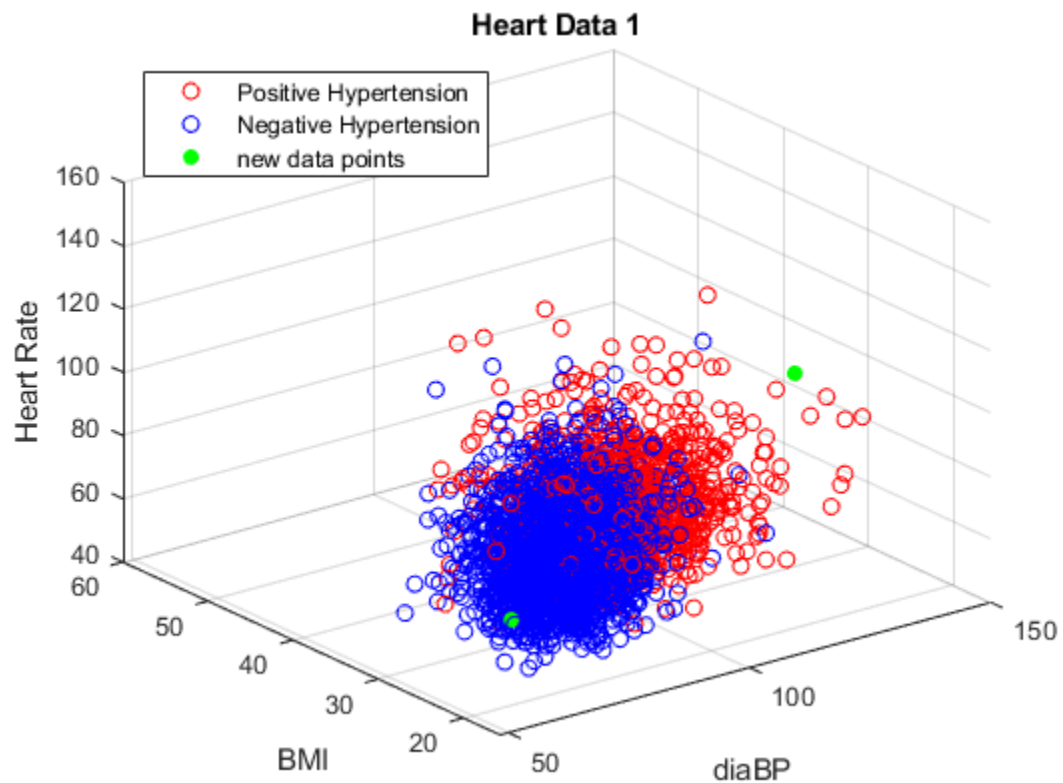
```
        gradi = gradi + ((Y(i)*xt(:,i)))/(1+  
exp(Y(i)*transpose(w)*xt(:,i)));  
  
    end  
    % the lines below update the weights  
    gradient = -1*gradi/N;  
    wnew = w_update(w,gradient);  
    w = wnew;  
  
end
```

Finding the Probability Using the Sigmoid Function

the for loop below takes new data and gives a probability output based on the weights that were previously found using the training data

```
prob1 = zeros(3,1);  
for i = 1:3  
    % the new data set, below, is the test data given in the instructions  
    for  
    % part a) to find the probability of prevalent hypertension  
    x1_new_test = [1 140 33 95;  
                   1 80 27 70;  
                   1 60 21 65];  
  
    x1_test_2_tran = transpose(x1_new_test(i,:));  
  
    % below is the sigmoid equation used to find the probability of  
    % positive  
    % hypertension based on the information given for 3 patients, which  
    % are  
    % above in the matrix named x1_new_test, and using the w's as well  
    prob1(i) = 1/(1+ exp(-1*transpose(w)*x1_test_2_tran)); %sigmoid  
    function  
end  
  
figure(2)  
scatter3(diaBP_pos, BMI_pos, heartRate_pos, 'red')  
hold on  
scatter3(diaBP_neg, BMI_neg, heartRate_neg, 'blue')  
hold on  
title('Heart Data 1')  
xlabel('diaBP');  
ylabel('BMI');  
zlabel('Heart Rate');  
  
% plotting the test points which are given in the instructions for  
% part a)  
scatter3(x1_new_test(:,2),x1_new_test(:,3),  
        x1_new_test(:,4), 'filled','green')
```

```
legend('Positive Hypertension','Negative Hypertension','new data
points','location','nw');
hold off
```



Results

```
% below are the weights being printed to the command window
disp('Below are the weights produced by the algorithm After Training
Data');
disp(w);

% output probability results
disp('Output Result from Test Data to Find Positive prevalentHyp
Probability');
disp(probl);

prevalentHypertension = probl;
diaBp = [140; 80; 60];
BMI = [33; 27; 21];
HeartRate = [95; 70; 65];
% Below is a table with the probability results
disp('The table below displays the patients probabilities of having
prevalent hypertension');
T = table(diaBp, BMI,HeartRate, prevalentHypertension);
disp(T)
% x_test_2_tran = transpose(x_test_2)
```

```
% prob1 = 1/(1+ exp(-1*transpose(w)*x1_test_2_tran))
```

Below are the weights produced by the algorithm After Training Data

```
-0.0206
0.0738
-0.0855
-0.0543
```

Output Result from Test Data to Find Positive prevalentHyp Probability

```
0.9117
0.4440
0.2856
```

The table below displays the patients probabilities of having prevalent hypertension

<i>diaBp</i>	<i>BMI</i>	<i>HeartRate</i>	<i>prevalentHypertension</i>
-----	-----	-----	-----
140	33	95	0.91167
80	27	70	0.44404
60	21	65	0.28563

Functions

Weight Update Function

```
function w_new = w_update(w, gradient)
    n = 0.01;
    new_w = w - ((n*gradient)/norm(gradient));
    w_new = new_w;
end
```

Excel Import File Function

```
function [diaBP1, BMI, heartRate, prevalentHyp] =
    importfile(workbookFile, sheetName, dataLines)
%IMPORTFILE1 Import data from a spreadsheet
% [DIABP1, BMI, HEARTRATE, PREVALENTHYP] = IMPORTFILE1(FILE) reads
data
% from the first worksheet in the Microsoft Excel spreadsheet file
% named FILE. Returns the data as column vectors.
%
% [DIABP1, BMI, HEARTRATE, PREVALENTHYP] = IMPORTFILE1(FILE, SHEET)
% reads from the specified worksheet.
%
% [DIABP1, BMI, HEARTRATE, PREVALENTHYP] = IMPORTFILE1(FILE, SHEET,
% DATALINES) reads from the specified worksheet for the specified row
% interval(s). Specify DATALINES as a positive scalar integer or a
% N-by-2 array of positive scalar integers for dis-contiguous row
% intervals.
```

```
%
% Example:
% [diaBP1, BMI, heartRate, prevalentHyp] = importfile1("C:\Users
\Baker\Desktop\School\ECG703\Midterm\problem_1_dataset_given
\HeartData_01.xlsx", "Sheet1", [2, 3750]);
%
% See also READTABLE.
%
% Auto-generated by MATLAB on 30-Mar-2021 02:02:04

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
% Input handling

% If no sheet is specified, read first sheet
if nargin == 1 || isempty(sheetName)
    sheetName = 1;
end

% If row start and end points are not specified, define defaults
if nargin <= 2
    dataLines = [2, 3750];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%

% Setup the Import Options
opts = spreadsheetImportOptions("NumVariables", 4);

% Specify sheet and range
opts.Sheet = sheetName;
opts.DataRange = "A" + dataLines(1, 1) + ":D" + dataLines(1, 2);

% Specify column names and types
opts.VariableNames = ["diaBP1", "BMI", "heartRate", "prevalentHyp"];
opts.SelectedVariableNames =
    ["diaBP1", "BMI", "heartRate", "prevalentHyp"];
opts.VariableTypes = ["double", "double", "double", "double"];

% Import the data
tbl = readtable(workbookFile, opts, "UseExcel", false);

for idx = 2:size(dataLines, 1)
    opts.DataRange = "A" + dataLines(idx, 1) + ":D" + dataLines(idx,
2);
    tb = readtable(workbookFile, opts, "UseExcel", false);
    tbl = [tbl; tb]; %#ok<AGROW>
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%

% Convert to output type
```



```
diaBP1 = tbl.diaBP1;  
BMI = tbl.BMI;  
heartRate = tbl.heartRate;  
prevalentHyp = tbl.prevalentHyp;  
end
```

Published with MATLAB® R2019a