LAB ACTIVITY 3: LOGISTIC REGRESSION and NEWTON'S METHOD

CARANDO, Paolo Creas
GOMENDOZA, Frank
LIM, Theo Paolo
LORENA, Neil John
ZABAT, Juan Paolo

I.INTRODUCTION

Lab Activity 3 will be about implementing Logistic Regression with the use of Newton's Method.

A new regression will be used. Logistic regression uses probabilities as well as classification. This is one the sought-out method used in tools such as applied statistics or data mining. Logistic regression measures the relationship between a categorical dependent variable and one or more independent variable by estimating its probabilities. For example, it can take the probability of a class and predict who will pass or fail depending on the score. Unlike linear regression, logistic regression predicts binary dependent variables rather than a continuous output.

Newton's method will used to calculate the line of the logistical regression. It is a method to find better successive approximation/prediction to the roots of a real valued function. The idea on the Newton's method is it will start first with an initial guess which is close to the root, and then the function will approximate the later prediction by its tangent line and computes the x-intercept of its tangent line. This is a helpful method, because the method converges on the root. The problem with this method, it requires the derivative to be calculated directly.

Newton's method will find the minimum derivative of the cost function, similarly with gradient descent.

II.PROCEDURE

Procedure 3.1

1. From the training data, load and plot the data in Matlab/Octave by using "+" to mark students who was admitted and "o" for those who are not admitted.
2. Add x_0 intercept into the data's x matrix with a value of 1.
3. Separate your plot by using *find* command. *(Optional, See Code 1 in Appendices)*
4. Define the sigmoid using the *inline* function in Matlab/Octave. *(See Code 2)*

Procedure 3.2

1. Plot the Cost Function J(θ) that is defined as:

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

2. After plotting, minimize the Cost Function using the Newton's Method

The hypothesis function in logistic regression:

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

$$= P(y=1|x;\theta)$$

Update rule for Newton's Method:

$$\theta^{(t+1)} = \theta^{(t)} - H^{-1}\nabla_\theta J$$

Hint: theta = theta - H / gradient

The gradient in logistic regression:

$$\nabla_\theta J = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$$

Hessian Matrix of f is defined as:

$$H = \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)})\left(1-h_\theta(x^{(i)})\right)x^{(i)}\left(x^{(i)}\right)^T\right]$$

3. Calculate J(θ) for every iterations to run and plot the data similar to Laboratory Exercise 1. Start with an initial value of θ = 0.

4. Record the values of theta that will be use in finding the decision boundary of the plot.

Decision boundary is defined as:

$$P(y=1|x;\theta) = g(\theta^T x) = 0.5$$

$$\theta^T x = 0$$

Procedure 3.3

1. Obtain the value of theta.
2. Identify the number of iterations needed for convergence.
3. Compute for the probability of the problem: *What is the probability that a student with a score of 20 on Exam 1 and a score of 80 on Exam 2 will not be admitted?*

Procedure 4.

Repeat the laboratory work with the group's own data.

III.RESULTS AND DISCUSSION

Procedure 3:

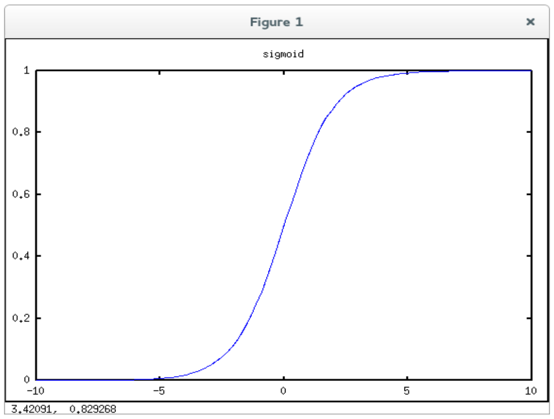

Fig. 1. Sigmoid Graph (Exercise)

Procedure 3.1:

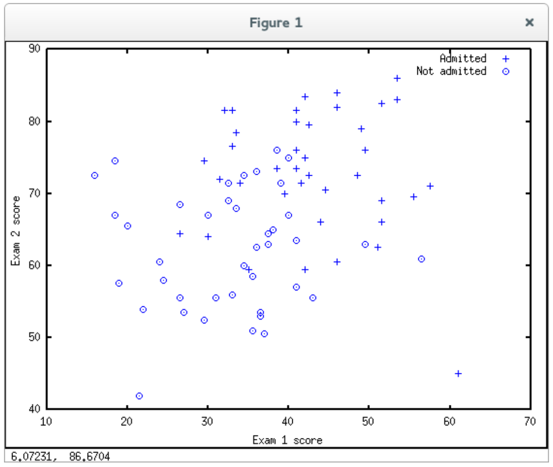

Fig. 2. Test 1 Scores vs Test 2 Scores

Procedure 3.2:

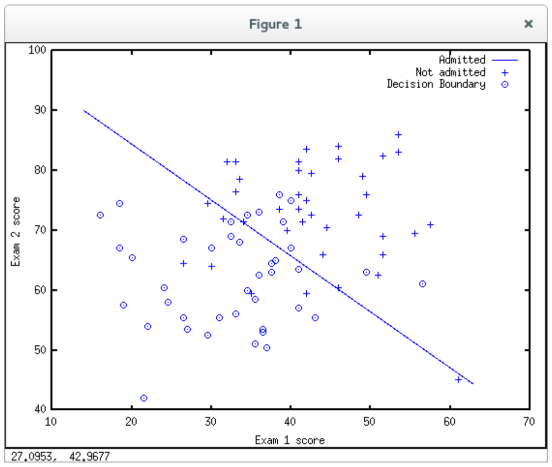

Fig. 3. Cost Function J(θ) Plot with Decision Boundary

Procedure 3.3:

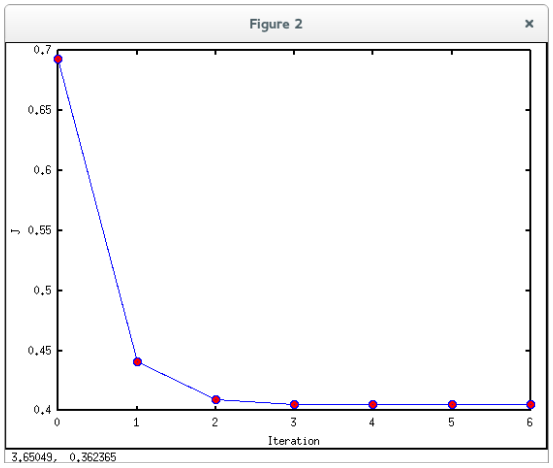

Fig. 4. J(θ) vs # of Iterations Plot

1. θ = -16.37874, 0.14834, 0.15891
2. How many iterations were required for convergence?
   _Five iterations were needed for convergence__
3. What is the probability that a student with a score of 20 on Exam 1 and a score of 80 on Exam 2 will not be admitted?
   _The probability is 0.66802_________________

Procedure 4.

Please refer to Data 1, Data 2 and Code 4 in Appendices for the group's data and code.

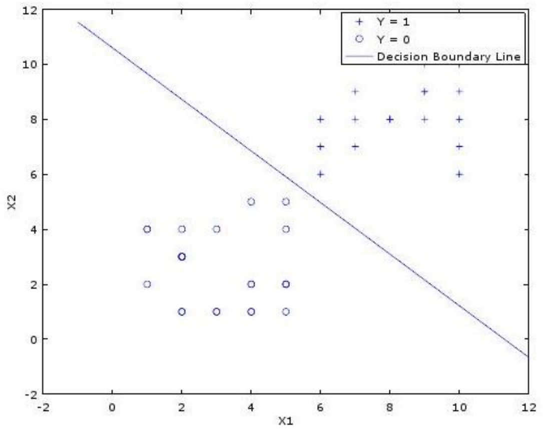

Fig. 5. Cost Function J(θ) Plot with Decision Boundary

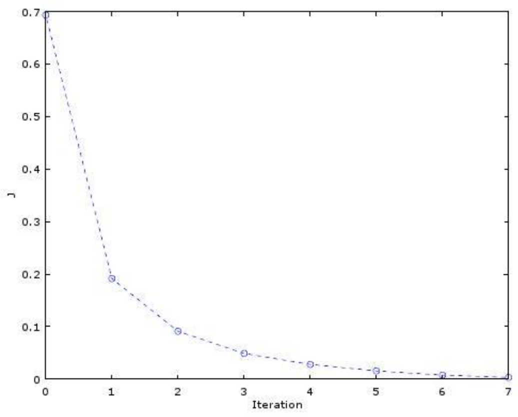

Fig. 6. J(θ) vs # of Iterations Plot

IV.CONCLUSION

The group concluded that Logistical Regression is an ideal method in classifying different data. Logistic Regression is a dynamic way of predicting values because it uses binary dependent variables instead of continuous values. The Newton's Method is applied to get the cost function of the data then finds the root by using the x-intercept of its tangent line. By using Newton's Method, we obtained the value of theta which is critical in finding the number of iterations needed for convergence of the graph. Logistical Regression is an optimum way of predicting probabilities of data.

V.APPENDICES

```
%find returns the indices of the
%rows meeting the specified condition
pos = find(y == 1);
neg = find(y == 0);

%Assume the features are in the 2nd & 3rd
%columns of x
plot(x(pos,2), x(pos,3), '+'); hold on
plot(x(neg,2), x(neg,3), 'o');
```

Code 1: *Find* Command Code

```
g = inline(‘1.0 ./ (1.0 + exp(-z))’);
```

Code 2: *Inline* Function Code

```
x = load('ex4x.dat');
y = load('ex4y.dat');
m = length(x);
x = [ones(m,1), x];
pos = find(y == 1);
neg = find(y == 0);
figure;
plot(x(pos, 2), x(pos,3), '+');
hold on;
plot(x(neg, 2), x(neg, 3), 'o');
g = inline('1.0 ./ (1.0 + exp(-z))');
theta = [0; 0; 0];
for i = 1:10
        h = g(x * theta);
  gradient = 1 / m .* x' * (h - y);
  hessian = 1 / m .* x' * diag(h) * diag(1 -
h) * x;
        J(i) = 1 / m * sum(-y' * log(h) - (1
- y)' * log(1 - h));
  theta = theta - hessian \ gradient;
end
probability = 1 - g([1 20 80] * theta);
plot_x = [min(x(:,2)) - 2,  max(x(:,2)) +
2];
plot_y = -1 / theta(3) * (theta(2) * plot_x
+ theta(1));
plot(plot_x, plot_y);
xlabel 'Exam 1 score';
ylabel 'Exam 2 score';
legend('Admitted', 'Not admitted',
'Decision Boundary');
hold off;
figure;
plot(0:9, J, 'o--');
xlabel 'Iteration';
ylabel 'J';
```

Code 3: Group’s Lab Work Code

Procedure 4:

```
%Rows 1-25
0
%Rows 26-50
1
```

Data 1: ‘logresy.dat’ Data

```
5       5
4       2
4       1
3       4
1       2
5       4
2       1
2       1
1       4
2       3
2       3
4       2
5       2
2       3
4       1
2       4
5       2
3       1
2       3
5       1
5       2
1       4
4       5
2       3
3       1
8       8
7       8
6       8
10      6
6       6
9       9
7       9
10      7
7       7
9       8
8       8
6       8
6       7
```

```
10	8
10	6
10	10
10	9
8	8
6	6
10	10
9	9
9	10
6	7
8	8
10	8
```

Data 2: ‘logresx.dat’ Data

```
x = load('logresx.dat');
y = load('logresy.dat');
m = length(x);
x = [ones(m,1), x];
pos = find(y == 1);
neg = find(y == 0);
figure;
plot(x(pos, 2), x(pos,3), '+');
hold on;
plot(x(neg, 2), x(neg, 3), 'o');
g = inline('1.0 ./ (1.0 + exp(-z))');
theta = [0; 0; 0];
for i = 1:9
	h = g(x * theta);
   gradient = 1 / m .* x' * (h - y);
   hessian = 1 / m .* x' * diag(h) * diag(1 -
h) * x;
	J(i) = 1 / m * sum(-y' * log(h) - (1
- y)' * log(1 - h));
   theta = theta - hessian \ gradient;
end
probability = 1 - g([1 4 9] * theta);
plot_x = [min(x(:,2)) - 2,  max(x(:,2)) +
2];
plot_y = -1 / theta(3) * (theta(2) * plot_x
+ theta(1));
plot(plot_x, plot_y);
xlabel 'x1';
ylabel 'x2';
legend('y = 1', 'y = 0', 'Decision
Boundary');
hold off;
figure;
plot(0:8, J, 'o--');
xlabel 'Iteration';
ylabel 'J';
```

Code 4: Group’s Own Data Code

VI.BIBLIOGRAPHY

[1] Ng, A. 2012. *Logistic Regression and Newton’s Method.* Machine Learning.