LAB ACTIVITY 4: REGULARIZATION and TITANIC

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

I.INTRODUCTION

Regularization is another method in implementing machine learning. In this lab activity, regularization to both linear and logistical regression will be covered. The titanic disaster will be also covered by implementing regularization logistical regression in real-world scenarios. Regularization is presenting additional information to the plot. This method will determine if the prediction is 'fit' on the given data. *Underfitting* is a linear function; one example is on linear regression. While increasing the polynomial, the curve will *overfit* the function. Thus too many features will try to fit all the training data. Similarly, the sigmoidal function of logistical regression is an *underfit*. However, increasing the value of the polynomial will result to an *overfit*. Regularization basically, retains all the features while reducing the parameters of theta (polynomial). This method handles more features, for example, the titanic disaster where many people died. For linear regression, the hypothesis will be a high order polynomial, see equation 1.

Equation 1. Hypothesis Regularization (Linear)

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \theta_5 x^5$$

Equation 2. Regularization Cost Function (Linear)

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

where λ is the regularization parameter

Lambda will be the regularization parameter wherein it will control the fit of the training data.

In regularization logistical regression, newton's method will be used again.

Equation 3. Regularization Cost Function (Logistical)

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Similar with an un-regularized cost function, but a regularized term was added on the last equation, see equation 3. The Newton's method will be using Hessian and the gradient.

Equation 4. Regularization Newton's Law

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

Hessian is almost similar with the cost function of linear regression with an additional regularization term and an identity matrix

Equation 5. Hessian Equation

$$H = \frac{1}{m}\left[\sum_{i=1}^{m} h_\theta(x^{(i)})\left(1-h_\theta(x^{(i)})\right) x^{(i)}\left(x^{(i)}\right)^T\right] + \frac{\lambda}{m}\begin{bmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}$$

II.PROCEDURE

4.1 (See Code 1)

1. Load the two data files in Octave and assign the two training data to x and y.
2. Plot the training data into a two dimensional plot.
3. Initialize fitting parameters by equating the features with powers of x from x^0 to x^5.
4. Set the regularization parameter which is lambda = 0;
5. Input the closed form solution from the normal equations and the extra regularization terms.
6. Plot the linear fit with a much denser array of x-values.

4.2

1. Using the Normal equation, find values for θ
using the three regularization parameters below:
a. λ = 0 (this is the same case as nonregularized
linear regression)
b. λ = 1
c. λ = 10

4.3

1. When you have found the answers for θ, verify them with the values in the solutions.
2. In addition to listing the values for each element θ_j of the θ vector, we will also provide the L2-norm of θ so you can quickly check if your answer is correct.
3. In Octave, you can calculate the L2-norm of a vector x using the command norm(x).

4.4

1. Plot the training data by equating x to ex5Logx.dat and y to ex5Logy.dat.
2. Use different markers for positives and negatives.

4.5

1. Run Newton's Method using the three values of lambda below:
   a. λ = 0 (this is the same case as nonregularized logistic regression)
   b. λ = 1
   c. λ = 10

4.6

1. Print out the value of J(θ) during each iteration.
2. J(θ) should not be decreasing at any point during Newton's Method.
3. If it is, check that you have defined J(θ) correctly.
4. Also check your definitions of the gradient and Hessian to make sure there are no mistakes in the regularization parts.

4.7

1. After convergence, use your values of theta to find the decision boundary in the classification problem.
2. The decision boundary is defined as the line where P(y =1|x;theta) = 0.5 → theta^T = 0

4.8

1. Finally, because there are 28 elements θ, we will not provide an element-by-element comparison in the solutions.
2. Instead, use norm(theta) to calculate the L2-norm of θ, and check it against the norm in the solutions.

III.RESULTS AND DISCUSSION

PART 1: Regularized Linear Regression

Procedure 4.1:

Fig. 1. shows the plot of the training data wherein the group assumed that a linear curve would not be able to fit the data. Alternatively, the group used a higher-order polynomial to obtain a more appropriate curve to apprehend better the various data points.

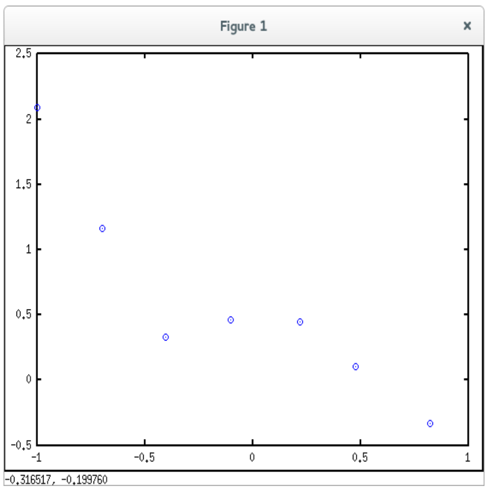

Fig. 1. Plot Data of Linx and Liny .dat file

Procedure 4.2 & 4.3:

Table 1. Values for θ for Varying λ			
	λ = 0	**λ = 1**	**λ = 10**
θ1	0.47253	0.39760	0.520471
θ2	0.68135	-0.42067	-0.182507
θ3	-1.38013	0.12959	0.060643
θ4	-5.97769	-0.39747	-0.148177
θ5	2.44173	0.17526	0.074330
θ6	4.73711	-0.33939	-0.127957
Norm θ	8.1687	0.80977	0.59307

In Fig. 2. where **λ** = 0 shows that the curve hit all the training data, but is not likely good to show all the movement or direction. This plot is an *overfitting* curve.

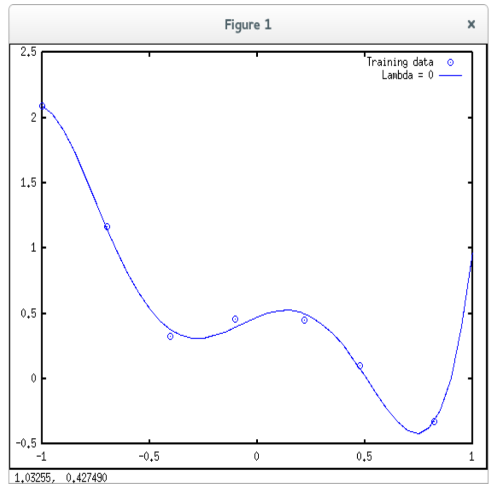

Fig. 2. Plot of Training Data with **λ** = 0

In Fig. 3. where **λ** = 1 displays that the *overfitting* curve was reduced and the new curve only hits some of the training data. This plot is just the right curve for the training data.

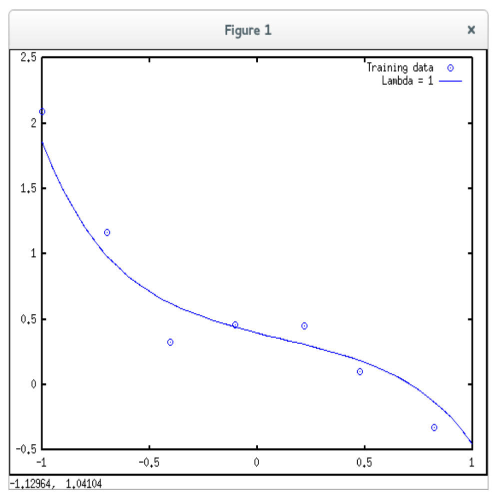

Fig. 3. Plot of Training Data with **λ** = 1

And in Fig. 4. where **λ** = 10 exhibits that the new curve was not able to follow the correct movement or direction of the training data points. This plot is an *underfitting* curve.

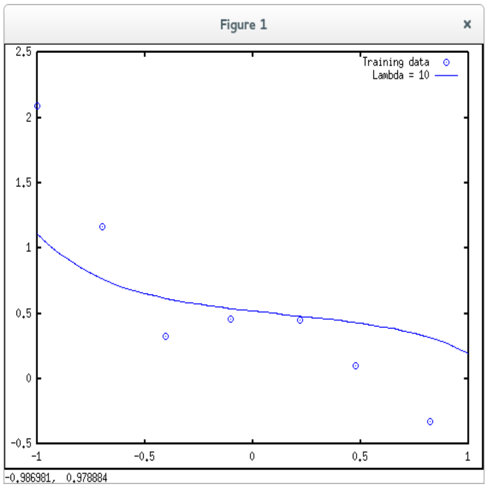

Fig. 4. Plot of Training Data with **λ** = 10

From looking at the previous graphs, what conclusions can you make about how the regularization parameter affects **λ** *your model?*

<u>It is evident that when lambda (**λ**) increases, the norms of theta (θ) decreases. Also the fitting for each lambda (**λ**) value varies from, overfitting, just right and underfitting.</u>

PART 11: Regularized Logistic Regression

Procedure 4.4:

Fig. 5. shows the plot of training data wherein data when y=1 are represented by '+' and data when y=0 are displayed as 'o'.

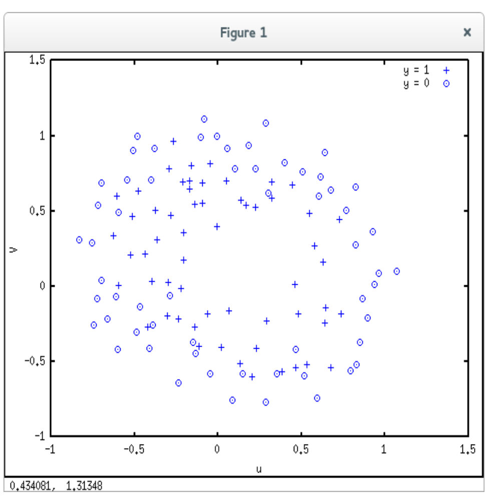

Fig. 5. Plot Data of Logx and Logy .dat file

Procedure 4.5, 4.6 and 4.7:

Table 2. Values for Norm θ for Varying λ			
	λ = 0	**λ = 1**	**λ = 10**
Norm θ			

In Fig. 6. where **λ** = 0 shows that the decision boundary is very exact that it even created an isolated area y=0 inside the larger area y=1.

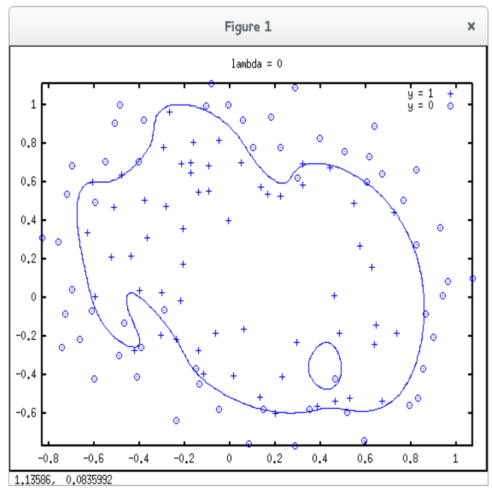

Fig. 6. Plot of Training Data with **λ** = 0

In Fig. 7. where **λ** = 1 displays a much simple decision boundary for the training data. The boundary still separates values in '+' and 'o', though the previous isolated area is not present anymore in the plot.

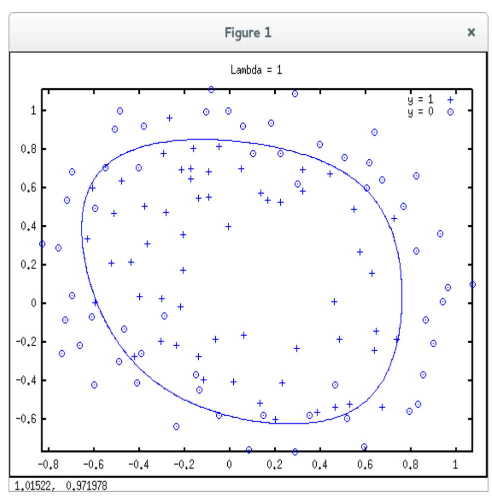

Fig. 7. Plot of Training Data with **λ** = 1

And in Fig. 8. where **λ** = 10 exhibits the decision boundary was not able to follow the training data in the plot as it mixed the '+' and 'o' values inside and outside the boundary. The most affected part of the boundary is seen in the left side where most 'o' values were included inside the boundary.

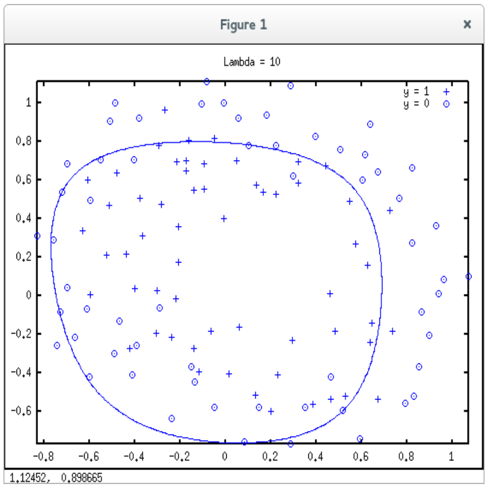

Fig. 8. Plot of Training Data with **λ** = 10

PART III: Real-world Application

IV. CONCLUSION

This experiment focuses on applying regularization for machine learning. Linear and logistical regression is used on the experiment to properly deal with problems where the number of variables is high. By applying regularization, line predictions are made labeling them as underfitting, just right, and overfitting. By varying the lambda for the regularization cost function, we obtained different fits for the training data. Based from the obtained results on parts I and II of the experiment, the optimum data prediction should be "just right".

V. APPENDICES

Linear Regression

```
x = load('ex5Linx.dat'); y =
load('ex5Liny.dat');
figure;
plot(x, y, 'o', 'MarkerFacecolor', 'r',
'MarkerSize', 8);
```

Code 1. Plotting

```
x = [ones(m, 1), x, x.^2, x.^3, x.^4, x.^5];
theta = zeros(size(x(1,:)))'
```

Code 2. Hypothesis

```
L = lambda.*eye(6);
L(1) = 0;
theta = (x' * x + L)\x' * y
norm_theta = norm(theta)
```

Code 3. Normal Equation

```
x_vals = (-1:0.05:1)';
features = [ones(size(x_vals)), x_vals,
x_vals.^2, x_vals.^3,...
        x_vals.^4, x_vals.^5];
plot(x_vals, features*theta, '--',
'LineWidth', 2)
legend('Training data', '5th order fit')
```

Code 4.

Logistical Regression

```
x = load('ex5Logx.dat');
y = load('ex5Logy.dat');
figure
pos = find(y); neg = find(y == 0);
plot(x(pos, 1), x(pos, 2), 'k+','LineWidth',
2, 'MarkerSize', 7)
hold on
plot(x(neg, 1), x(neg, 2), 'ko',
'MarkerFaceColor', 'y', 'MarkerSize', 7)
```

Code 5.

```
for i = 1:MAX_ITR
    z = x * theta;
```

```
    h = g(z)
```

Code 6.

```
J(i) =(1/m)*sum(-y.*log(h) - (1-y).*log(1-
h))+ ...
   (lambda/(2*m))*norm(theta([2:end]))^2;
G = (lambda/m).*theta; G(1) = 0;
L = (lambda/m).*eye(n); L(1) = 0
grad = ((1/m).*x' * (h-y)) + G;
 H = ((1/m).*x' * diag(h) * diag(1-h) * x)
+ L;
theta = theta - H\grad;
```

Code 7.

VI.BIBLIOGRAPHY

[1] Ng, A. 2012. *Regularization.* Machine Learning.