Experiment 2: Logistic Regression and Newton's Method

August 29, 2018

1 Description

In this exercise, you will use Newton's Method to implement logistic regression on a classification problem.

2 Data

To begin, download data2.zip and extract the files from the zip file. For this exercise, suppose that a high school has a dataset representing 40 students who were admitted to college and 40 students who were not admitted. Each $(x^{(i)}, y^{(i)})$ training example contains a student's score on two standardized exams and a label of whether the student was admitted.

Your task is to build a binary classification model that estimates college admission chances based on a student's scores on two exams. In your training data, the first column of your x array represents all Test 1 scores, and the second column represents all Test 2 scores, and the y vector uses "1" to label a student who was admitted and "0" to label a student who was not admitted.

3 Plot the Data

Load the data for the training examples into your program and add the $x_0 = 1$ intercept term into your x matrix.

Before beginning Newton's Method, we will first plot the data using different symbols to represent the two classes. In Matlab/Octave, you can separate the positive class and the negative class using the find command:

```
% 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 and 3rd
% columns of x
plot(x(pos, 2), x(pos,3), '+'); hold on
plot(x(neg, 2), x(neg, 3), 'o')
```

Your plot should look like the following:
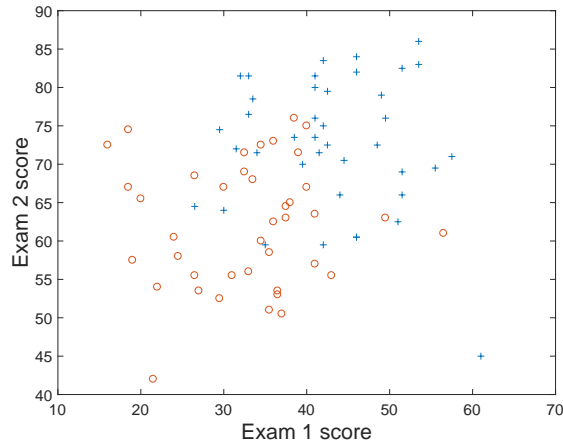
4 Logistic Regression

Recall that in logistic regression, the hypothesis function is

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}} = P(y = 1 \mid x; \theta) \tag{1}$$

Matlab/Octave does not have a library function for the sigmoid, so you will have to define it yourself. The easiest way to do this is through an inline expression:

```
g = inline('1.0 ./ (1.0 + exp(-z))');
% Usage: To find the value of the sigmoid
% evaluated at 2, call g(2)
```

Given a set of training data $\{x^{(i)}\}_{i=1,\cdots,m}$, we define a likelihood function as

$$J(\theta) = \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1 - h_\theta(x^{(i)}))^{1-y^{(i)}} \tag{2}$$

To facilitate the computation, we maximize the following log likelihood function

$$L(\theta) = \frac{1}{m} \sum_{i=1}^{m} [y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))] \tag{3}$$

Note that maximizing (3) is equivalent to minimizing its negative. Then, our problem becomes [1]

$$\min_\theta \quad L(\theta) = \frac{1}{m} \sum_{i=1}^{m} [-y^{(i)} \log(h_\theta(x^{(i)})) - (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))] \tag{4}$$

where $\nabla_\theta L$ is the gradient of L and can be defined as

$$\nabla_\theta L = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)} \tag{5}$$

---

[1]Note that it is not a must to perform this transformation, since both gradient ascent algorithm and Newton's method can be applied to resolve maximization problems.

One approach to minimize the above objective function is gradient descent algorithm, where we update θ iteratively according to the following rule

$$\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta) \tag{6}$$

until the difference between the objective function values in successive iterations is less than (or equal to) some threshold ϵ, i.e.

$$|L^+(\theta) - L(\theta)| \leq \epsilon \tag{7}$$

Try to resolve the logistic regression problem using gradient descent method with the initialization $\theta = 0$, and answer the following questions:

1. Assume $\epsilon = 10^{-6}$. How many iterations are required to achieve convergence? Note that gradient descent method has a very slow convergence rate and may take a long while to achieve the minimum.

2. What values of θ did you get after achieving the convergence?

3. Calculate $L(\theta)$ in each iteration and illustrate how $L(\theta)$ is decreased iteratively in the gradient descent method.

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

   Plotting the decision boundary is equivalent to plotting the $\theta^T x = 0$ line. When you are finished, your plot should appear like the figure below. Note that the figures may be slightly different under different parameter settings.
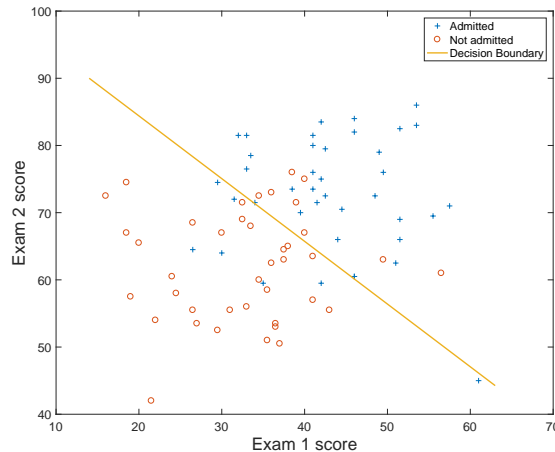
5. What is the probability that a student with a score of 20 on Exam 1 and a score of 80 on Exam 2 will not be admitted?

5   Newton's Method

Our goal is to use Newton's method to minimize this function. Recall that the update rule for Newton's method is

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

In logistic regression, the Hessian is

$$H = \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)})\left(1 - h_\theta(x^{(i)})\right)x^{(i)}\left(x^{(i)}\right)^T\right] \tag{8}$$

Note that the formulas presented above are the vectorized versions. Specifically, this means that $x^{(i)} \in R^{n+1}$, $x^{(i)}\left(x^{(i)}\right)^T \in R^{(n+1)\times(n+1)}$, while $h_\theta(x^{(i)})$ and $y^{(i)}$ are scalars.

Now, implement Newton's method in your program, starting with the initial value of $\theta = \vec{0}$. Use the same stopping condition (7) as the gradient descent method. To determine how many iterations to use, calculate $L(\theta)$ for each iteration and plot your results. Newton's method often converges in 5-15 iterations.

Finally, answer the following questions.

1. What values of θ did you get when achieving convergence?

2. Show how L is decreased iteratively by Newton's method.

3. Plot the decision boundary.

4. 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?

5. What did you learn from comparing gradient descent method with Newton's method?