Nick Montana

Professor Yan Yan

Intro to Machine Learning

3/10/2017

Project 2 Report

Introduction: Project 2 consists of completing various functions. Significant code snippets and outputs can be found below:

costFunction.m:

```
% ====================== YOUR CODE HERE ======================
% Instructions: Compute the cost of a particular choice of
theta.
%               You should set J to the cost.
%               Compute the partial derivatives and set grad to
the partial
%               derivatives of the cost w.r.t. each parameter in
theta
%
% Note: grad should have the same dimensions as theta
%

z = X * theta;
h = sigmoid (z);

cost_sum = 0;
for i = 1:m

    a = -y(i)*log(h(i));
    b = (1 - y(i))*log (1 - h (i));
    cost_sum = cost_sum + (a - b);
end

J = cost_sum / m;

for j = 1:size (grad)
    pd_sum = 0;
```

```
    for i = 1:m
        a = (h(i) - y (i)) * X (i, j);
        pd_sum = pd_sum + a;
    end
    grad (j) = pd_sum / m;
end

% =============================================================
```

costFunctionReg.m:

```
% ====================== YOUR CODE HERE ======================
% Instructions: Compute the cost of a particular choice of
theta.
%               You should set J to the cost.
%               Compute the partial derivatives and set grad to
the partial
%               derivatives of the cost w.r.t. each parameter in
theta

z = X * theta;
h = sigmoid (z);

cost_sum = 0;
for i = 1:m

    a = -y(i)*log(h(i));
    b = (1 - y(i))*log (1 - h (i));
    cost_sum = cost_sum + (a - b);
end

theta_sum = 0;
for j = 1:n
    a = theta (j);
    theta_sum = theta_sum + a^2;
end

lambda_part = theta_sum * (lambda / (2 * m));

J = (cost_sum / m) + lambda_part;

pd_sum = 0;
for i = 1:m
    a = (h(i) - y (i)) * X (i, 1);
    pd_sum = pd_sum + a;
end
grad (1) = pd_sum / m;
```

```
for j = 2:size (grad)
    pd_sum = 0;
    for i = 1:m
        a = (h(i) - y (i)) * X (i, j);
        pd_sum = pd_sum + a;
    end
    grad (j) = (pd_sum / m) + (lambda / m) * theta (j);
end


% ==========================================================
```

sigmoid.m:

```
% ====================== YOUR CODE HERE ======================
% Instructions: Compute the sigmoid of each value of z (z can be
a matrix,
%               vector or scalar).

for i = 1:size (z, 1)
    for j = 1:size (z, 2)
        g (i, j) = 1/(1 + exp (-z (i,j)));
    end
end

% =============================================================
```

plotData.m:

```
% ====================== YOUR CODE HERE ======================
% Instructions: Plot the positive and negative examples on a
%               2D plot, using the option 'k+' for the positive
%               examples and 'ko' for the negative examples.
%

% Find Indices of Positive and Negative Examples
pos = find(y==1); neg = find(y == 0);
% Plot Examples
plot(X(pos, 1), X(pos, 2), 'k+','LineWidth', 2, ...
    'MarkerSize', 7);
plot(X(neg, 1), X(neg, 2), 'ko', 'MarkerFaceColor', 'y', ...
    'MarkerSize', 7);
```

```
%
======================================================================
=========
```

predict.m:

```
% ====================== YOUR CODE HERE ======================
% Instructions: Complete the following code to make predictions
using
%               your learned logistic regression parameters.
%               You should set p to a vector of 0's and 1's
%

z = X * theta;

for i = 1:m
    if (sigmoid (z (i)) >= 0.5)
        p (i) = 1;
    end
end

%
======================================================================
=========
```

Figure 1 for ex2:

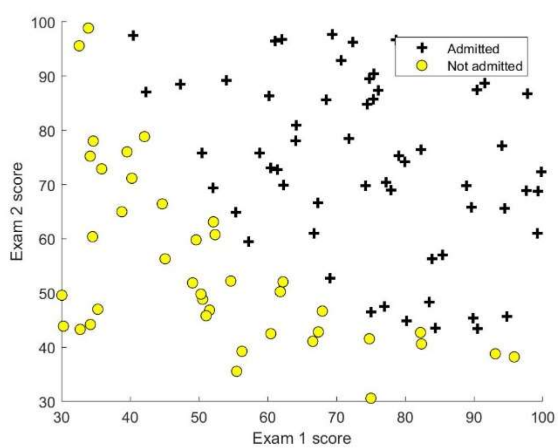
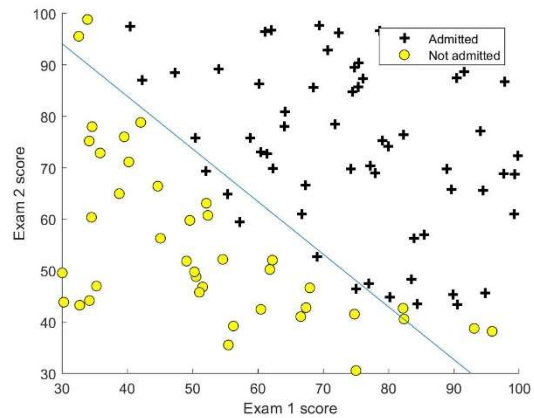


Figure 2 for ex2:



ex2 Output:

Command Window

New to MATLAB? See resources for Getting Started.

```
Plotting data with + indicating (y = 1) examples and o indicating (y = 0) examples.

Program paused. Press enter to continue.
Cost at initial theta (zeros): 0.693147
Expected cost (approx): 0.693
Gradient at initial theta (zeros):
 -0.100000
 -12.009217
 -11.262842
Expected gradients (approx):
 -0.1000
 -12.0092
 -11.2628

Cost at test theta: 0.218330
Expected cost (approx): 0.218
Gradient at test theta:
 0.042903
 2.566234
 2.646797
Expected gradients (approx):
 0.043
 2.566
 2.647

Program paused. Press enter to continue.
Warning: Your current settings will run a different algorithm ('quasi-newton') in a future
release. Either use optimoptions to set options (recommended), or set option Algorithm to
'trust-region' using optimset.
> In throwFminuncGradObjandLargeScaleWarning (line 18)
  In fminunc (line 170)
  In ex2 (line 99)
fx
```

```
In ex2 (line 99)

Local minimum possible.

fminunc stopped because the final change in function value relative to
its initial value is less than the default value of the function tolerance.

<stopping criteria details>

Cost at theta found by fminunc: 0.203506
Expected cost (approx): 0.203
theta:
 -24.932774
 0.204406
 0.199616
Expected theta (approx):
 -25.161
 0.206
 0.201

Program paused. Press enter to continue.
For a student with scores 45 and 85, we predict an admission probability of 0.774321
Expected value: 0.775 +/- 0.002

Train Accuracy: 89.000000
Expected accuracy (approx): 89.0

fx >>
```

Figure 1 for ex2_reg:

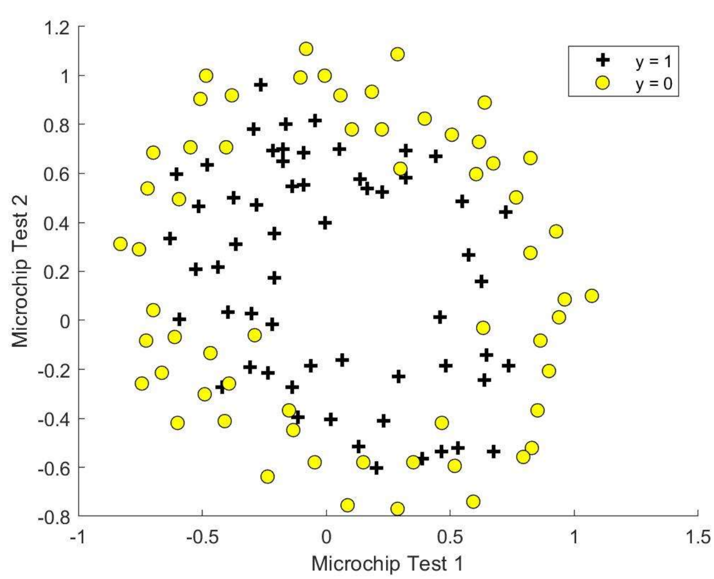
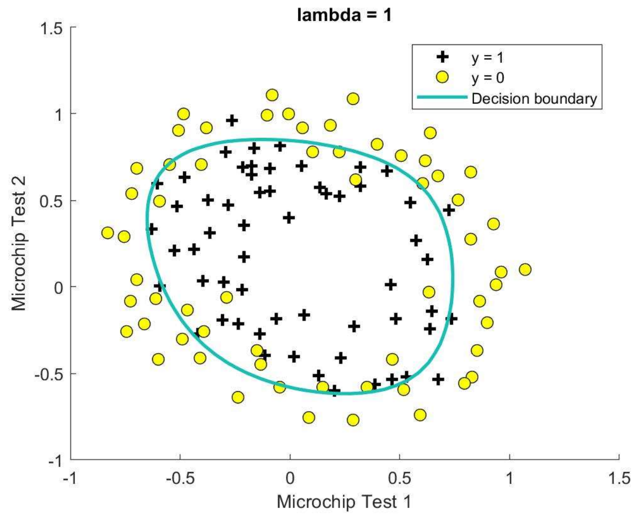


Figure 2 for ex2_reg:



ex2_reg Output:

Command Window

New to MATLAB? See resources for Getting Started.

```
Cost at initial theta (zeros): 0.693147
Expected cost (approx): 0.693
Gradient at initial theta (zeros) - first five values only:
 0.008475
 0.018788
 0.000078
 0.050345
 0.011501
Expected gradients (approx) - first five values only:
 0.0085
 0.0188
 0.0001
 0.0503
 0.0115

Program paused. Press enter to continue.

Cost at test theta (with lambda = 10): 3.206882
Expected cost (approx): 3.16
Gradient at test theta - first five values only:
 0.346045
 0.161352
 0.194796
 0.226863
 0.092186
Expected gradients (approx) - first five values only:
 0.3460
 0.1614
 0.1948
 0.2269
 0.0922

fx Program paused. Press enter to continue.
```

```
Expected gradients (approx) - first five values only:
 0.3460
 0.1614
 0.1948
 0.2269
 0.0922

Program paused. Press enter to continue.
Warning: Your current settings will run a different algorithm ('quasi-newton') in a future
release. Either use optimoptions to set options (recommended), or set option Algorithm to
'trust-region' using optimset.
> In throwFminuncGradObjandLargeScaleWarning (line 18)
  In fminunc (line 170)
  In ex2_reg (line 117)

Local minimum possible.

fminunc stopped because the size of the current step is less than
the default value of the step size tolerance.

<stopping criteria details>

Train Accuracy: 83.050847
Expected accuracy (with lambda = 1): 83.1 (approx)
>>
```