实验三

**function [J, grad] = lrCostFunction(theta, X, y, lambda)**

计算正则项系数为lambda的二范数正则的逻辑回归的损失函数和对theta的偏导数

代码：

（同练习2costFunctionReg(theta, X, y, lambda) ）

hypothesis = sigmoid(X \* theta);

J = -1 / m \* sum( y .\* log(hypothesis) + (1 - y) .\* log(1 - hypothesis) ) ...

+ lambda / 2 / m \* (sum(theta .^ 2) - theta(1)^2);

grad = 1 / m \* X' \* (hypothesis - y) + lambda / m \* theta;

grad(1) = grad(1) - lambda / m \* theta(1);

% no need to regularize theta 0

**function [all\_theta] = oneVsAll(X, y, num\_labels, lambda)**

训练一对多逻辑回归分类器，返回最终权重(shape = num\_labels\*(n+1),个人认为返回(n+1)\*num\_labels大小的矩阵更方便，可以省掉此处以及predictOneVsAll里的转置)

代码：

for i = 1:num\_labels

initial\_theta = zeros(n + 1, 1);

options = optimset('GradObj', 'on', 'MaxIter', 50);

initial\_theta = fmincg ( ...

@(t)(lrCostFunction(t, X, (y == i), lambda)), initial\_theta, options ...

);

% The i th row of all\_theta corresponds to

% the vector of weight of i th lable’s LR\_Classifier.

all\_theta(i, :) = initial\_theta';

endfor

**function p = predictOneVsAll(all\_theta, X)**

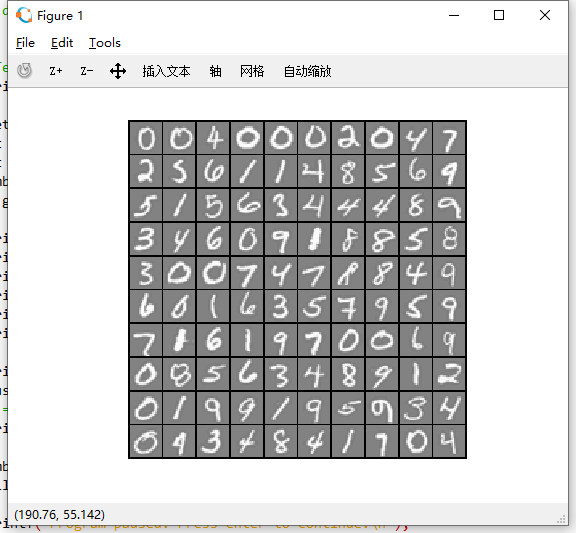
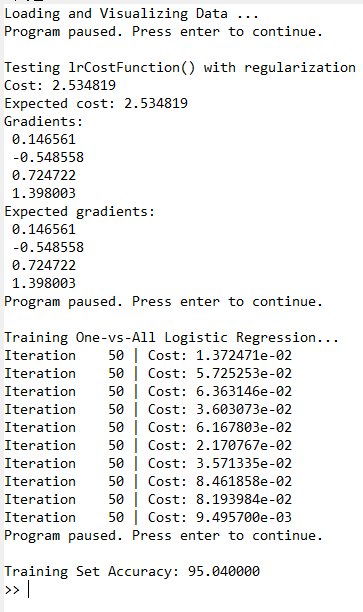
用一对多分类训练得到的权重进行预测

代码：

predictions = X \* all\_theta';

[max\_val, p] = max(predictions, [], 2);

**ex3运行结果**

**function p = predict(Theta1, Theta2, X)**

用预训练好的两层神经网络的权重进行预测

代码：

X = [ones(m, 1) X];

hidden = sigmoid(X \* Theta1');

hidden = [ones(m, 1) hidden];

predictions = sigmoid(hidden \* Theta2');

[max\_val, p] = max(predictions, [], 2);

**ex3\_nn运行结果**

