1.向量化逻辑函数

（其实在上周的练习中已经接近完成）

```
function [J, grad] = lrCostFunction(theta, X, y, lambda)
%LRCOSTFUNCTION Compute cost and gradient for logistic regression with
%regularization
%   J = LRCOSTFUNCTION(theta, X, y, lambda) computes the cost of using
%   theta as the parameter for regularized logistic regression and the
%   gradient of the cost w.r.t. to the parameters.

% Initialize some useful values
m = length(y); % number of training examples

% You need to return the following variables correctly
J = 0;
grad = zeros(size(theta));

% ====================== 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
%
% Hint: The computation of the cost function and gradients can be
%       efficiently vectorized. For example, consider the computation
%
%           sigmoid(X * theta)
%
%       Each row of the resulting matrix will contain the value of the
%       prediction for that example. You can make use of this to vectorize
%       the cost function and gradient computations.
%
% Hint: When computing the gradient of the regularized cost function,
%       there're many possible vectorized solutions, but one solution
%       looks like:
%           grad = (unregularized gradient for logistic regression)
%           temp = theta;
%           temp(1) = 0;   % because we don't add anything for j = 0
%           grad = grad + YOUR_CODE_HERE (using the temp variable)
```

```
%
hypo = sigmoid(X * theta);
J = (1/m) * sum(-y' * log(hypo) - (1 - y)' * log(1 - hypo)) + (lambda/(2 * m)) *
(sum((theta(2:length(theta))).^2));
%grad = (1/m) * sum(repmat((hypo - y), 1, size(X, 2)) .* X);
grad = ((1/m) * X' * (hypo - y))';注意这里，向量化的写法，其余内容和Week3最后的练习没有区别
grad(2:length(grad)) = grad(2:length(grad)) + (lambda/m) * (theta(2:length(theta)))';

% =============================================================

grad = grad(:);

end
```

2.多元分类算法

```
function [all_theta] = oneVsAll(X, y, num_labels, lambda)
%ONEVSALL trains multiple logistic regression classifiers and returns all
%the classifiers in a matrix all_theta, where the i-th row of all_theta
%corresponds to the classifier for label i
%   [all_theta] = ONEVSALL(X, y, num_labels, lambda) trains num_labels
%   logistic regression classifiers and returns each of these classifiers
%   in a matrix all_theta, where the i-th row of all_theta corresponds
%   to the classifier for label i

% Some useful variables
m = size(X, 1);
n = size(X, 2);

% You need to return the following variables correctly
all_theta = zeros(num_labels, n + 1);

% Add ones to the X data matrix
X = [ones(m, 1) X];

% ========================== YOUR CODE HERE ==========================
% Instructions: You should complete the following code to train num_labels
%               logistic regression classifiers with regularization
%               parameter lambda.
```

```
%
% Hint: theta(:) will return a column vector.
%
% Hint: You can use y == c to obtain a vector of 1's and 0's that tell you
%       whether the ground truth is true/false for this class.
%
% Note: For this assignment, we recommend using fmincg to optimize the cost
%       function. It is okay to use a for-loop (for c = 1:num_labels) to
%       loop over the different classes.
%
%       fmincg works similarly to fminunc, but is more efficient when we
%       are dealing with large number of parameters.
%
% Example Code for fmincg:
%
%     % Set Initial theta
%     initial_theta = zeros(n + 1, 1);
%
%     % Set options for fminunc
%     options = optimset('GradObj', 'on', 'MaxIter', 50);
%
%     % Run fmincg to obtain the optimal theta
%     % This function will return theta and the cost
%     [theta] = ...
%         fmincg (@(t)(lrCostFunction(t, X, (y == c), lambda)), ...
%                 initial_theta, options);
%
K = num_labels;
initial_theta = zeros(n + 1, 1);
options = optimset('GradObj', 'on', 'MaxIter', 50);
for k = 1:K,
all_theta(k, :) = ...
(fmincg (@(t)(lrCostFunction(t, X, (y == k), lambda)), initial_theta, options))';
```

1. K个预测值1,2,3,...,k，y == k就是把相应预测值的y代入lrCostFunction
2. fmincg用来计算能把函数（第一个引数）最小化的theta值，具体说明如下

To specify the actual function we are minimizing, we use a “short-hand” for specifying functions with the @(t) (costFunction(t, X, y)) . This creates a function, with argument t, which calls your costFunction. This allows us to wrap the costFunction for use with fminunc.

```
%  Set options for fminunc
options = optimset('GradObj', 'on', 'MaxIter', 400);

%  Run fminunc to obtain the optimal theta
%  This function will return theta and the cost
[theta, cost] = ...
    fminunc(@(t)(costFunction(t, X, y)), initial_theta, options);
```

end

%
==========

end

3.用神经网络进行预测

```
function p = predict(Theta1, Theta2, X)
%PREDICT Predict the label of an input given a trained neural network
%   p = PREDICT(Theta1, Theta2, X) outputs the predicted label of X given the
%   trained weights of a neural network (Theta1, Theta2)

% Useful values
m = size(X, 1);
num_labels = size(Theta2, 1);

% You need to return the following variables correctly
p = zeros(size(X, 1), 1);

% ====================== YOUR CODE HERE ======================
% Instructions: Complete the following code to make predictions using
%               your learned neural network. You should set p to a
%               vector containing labels between 1 to num_labels.
%
% Hint: The max function might come in useful. In particular, the max
%       function can also return the index of the max element, for more
%       information see 'help max'. If your examples are in rows, then, you
%       can use max(A, [], 2) to obtain the max for each row.
%
```

```
X = [ones(m, 1) X];
layerB = sigmoid(X * Theta1'); 第二层的值
ayerB = [ones(m, 1) layerB]; 给第二层加上偏差项
layerC = sigmoid(layerB * Theta2'); 第三层的值（预测出的概率）
[a, p] = max(layerC, [], 2); 计算矩阵layerC每一行的最大值，并返回位置p，由于我们预测的是数字，
而这里的位置又刚好和数字一一对应（除了用0表示10），于是在这个特殊的案例中，返回的p值也就是预测的结果
如果不是这样的特殊情况我们可能需要基于得到的p值进行进一步的处理
%
==========================================================================


end
```