1 Multi-class Classification

1.1 Dataset

```
% Load saved matrices from file
load('ex3data1.mat');
```

1.3 Vectorizing Logistic Regression

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right].$$

To compute each element in the summation, we have to compute $h_\theta(x^{(i)})$ for every example i, where $h_\theta(x^{(i)}) = g(\theta^T x^{(i)})$ and $g(z) = \frac{1}{1+e^{-z}}$ is the

4

sigmoid function. It turns out that we can compute this quickly for all our examples by using matrix multiplication. Let us define X and θ as

$$X = \begin{bmatrix} - & (x^{(1)})^T & - \\ - & (x^{(2)})^T & - \\ & \vdots & \\ - & (x^{(m)})^T & - \end{bmatrix} \quad \text{and} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}.$$

Then, by computing the matrix product $X\theta$, we have

$$X\theta = \begin{bmatrix} - & (x^{(1)})^T\theta & - \\ - & (x^{(2)})^T\theta & - \\ & \vdots & \\ - & (x^{(m)})^T\theta & - \end{bmatrix} = \begin{bmatrix} - & \theta^T(x^{(1)}) & - \\ - & \theta^T(x^{(2)}) & - \\ & \vdots & \\ - & \theta^T(x^{(m)}) & - \end{bmatrix}.$$

这里X*Theta即可代表Theta'*X

Code:

```
h = sigmoid(X*theta);
J = sum(-y .* log(h) -(1-y) .* log(1-h))/m
```

$$\frac{\partial J}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left((h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right).$$

To vectorize this operation over the dataset, we start by writing out all

the partial derivatives explicitly for all θ_j,

$$\begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \\ \frac{\partial J}{\partial \theta_2} \\ \vdots \\ \frac{\partial J}{\partial \theta_n} \end{bmatrix} = \frac{1}{m}\begin{bmatrix} \sum_{i=1}^m \left((h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}\right) \\ \sum_{i=1}^m \left((h_\theta(x^{(i)}) - y^{(i)})x_1^{(i)}\right) \\ \sum_{i=1}^m \left((h_\theta(x^{(i)}) - y^{(i)})x_2^{(i)}\right) \\ \vdots \\ \sum_{i=1}^m \left((h_\theta(x^{(i)}) - y^{(i)})x_n^{(i)}\right) \end{bmatrix}$$
$$= \frac{1}{m}\sum_{i=1}^{m}\left((h_\theta(x^{(i)}) - y^{(i)})x^{(i)}\right)$$
$$= \frac{1}{m}X^T(h_\theta(x) - y). \qquad (1)$$

因此,

```
grad = X' * (h-y) /m
```

1.3.3 Vectorizing regularized logistic regression

而后面需要进行正则化如下:

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

Note that you should *not* be regularizing θ_0 which is used for the bias term.

Correspondingly, the partial derivative of regularized logistic regression cost for θ_j is defined as

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \qquad \text{for } j \geq 1$$

因此修改:

```
bias = (sum(theta .^ 2 )- theta(1)^2)*lambda/2/m;
J += bias;
temp = lambda*theta / m;
```

```
temp(0) = 0;
grad += temp;
```

1.4 One-vs-all Classification

这里需要先对每一个class进行优化，找到最优的theta

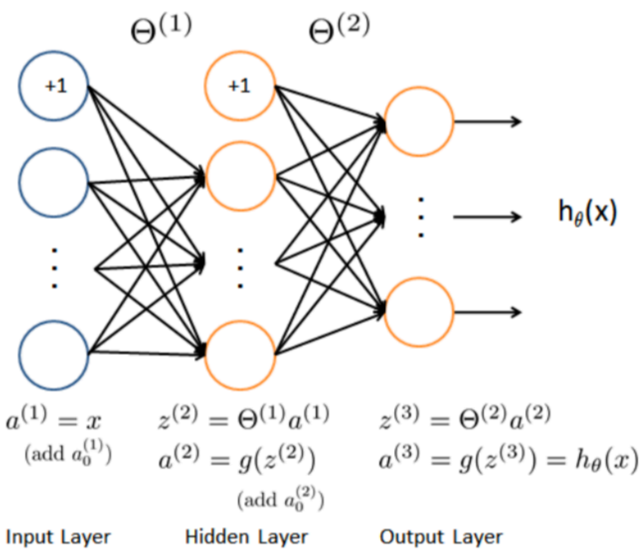
```
for c = 1:num_labels,
initial_theta = zeros(n+1,1);
options = optimset('GradObj','on','MaxIter',50);
[theta]=...
       fmincg(@(t)(lrCostFunction(t,X,(y==c),lambda))...
       initial_theta,options);
all_theta(c,:) = theta;
end
```

之后进行predict,这里的思路是每一列中最大项即是分类的类别

```
X = [ones(m,1) X];
h = X * all_theta';
h_max = max(h,[],2);
for i=1:m,
for j=1:num_labels,
if h(i,j)==h_max(i),
p(i)=j;
break;
end
 end
```

2 Netural Network

基本关系如下:



这里需要完成的是基于前向传播的预测，和上一个预测方法类似，也是对结果中每一行中取最大，找到对应的下标，即是预测的值。因此关根据上图中的关系计算出h

```
X = [ones(size(X,1),1),X];
z2 = X * Theta1' ;
```

```
a2 = sigmoid(z2);
a2 = [ones(size(a2,1),1),a2];
z3 = a2 * Theta2';
h = sigmoid(z3);

h_max = max(h,[],2);
for i=1:m,
 for j=1:num_labels,
 if h(i,j)==h_max(i),
 p(i)=j;
 break;
 end
 end
```