题目：Logistic Regression

姓名:张胤民 学号:201694069

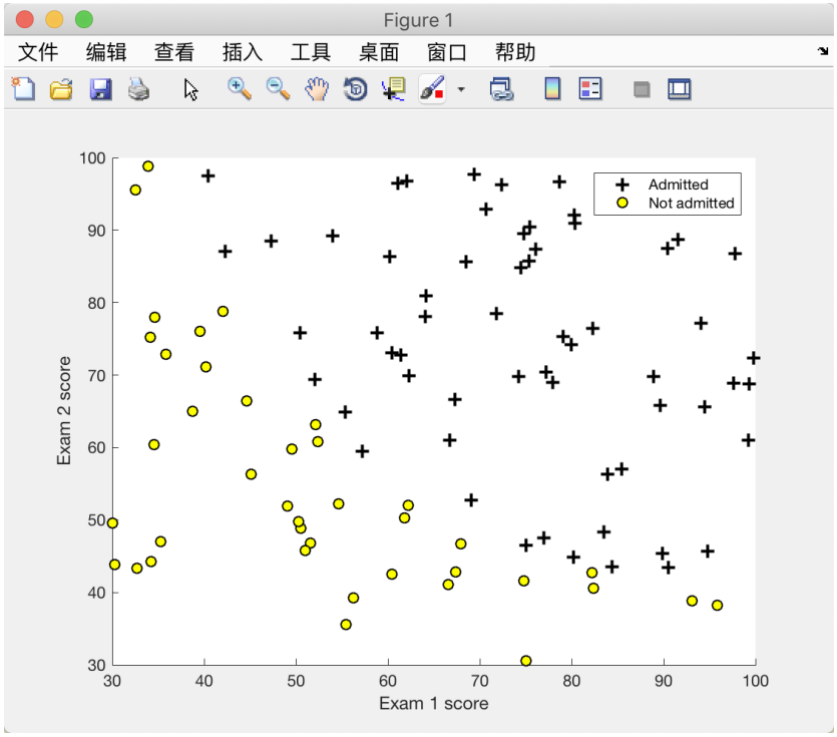
一、 实现功能简介

主要功能:

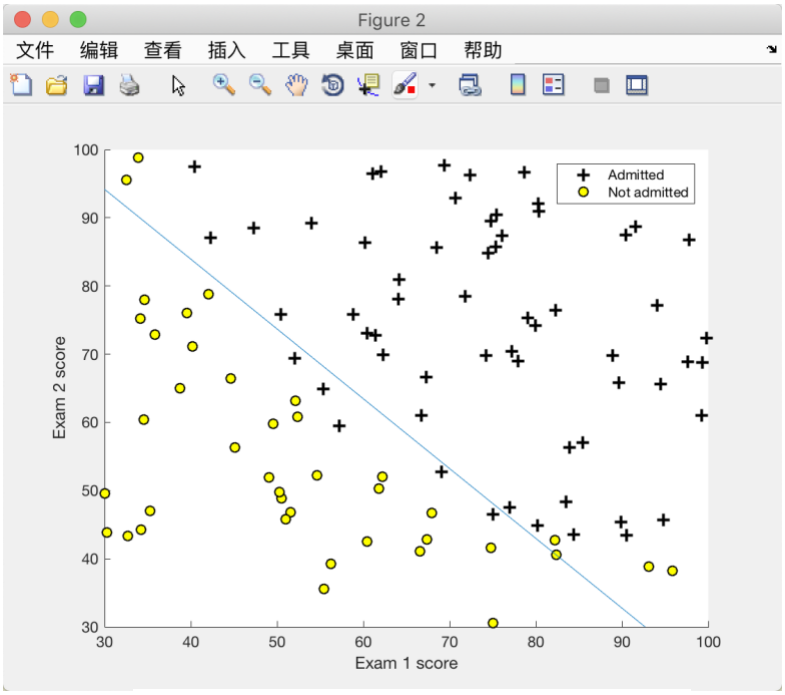
1. 实现逻辑回归的**误差计算(log-likehood loss function)**
2. 实现带有**正则化(regularization)**的逻辑回归**误差计算**
3. 实现特**征映射函数**将输入特征映射到多项式特征
4. 实现 **Sigmoid 函数**计算
5. 实现**分类结果预测**
6. 实现数据显示

二、 具体编写代码及结果展示以及代码功能描述

1. 结果展示

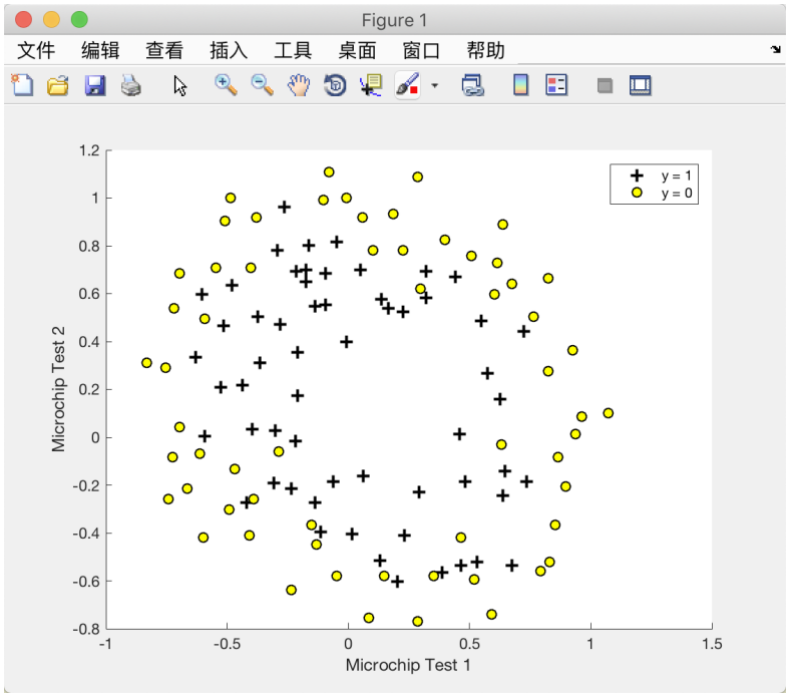


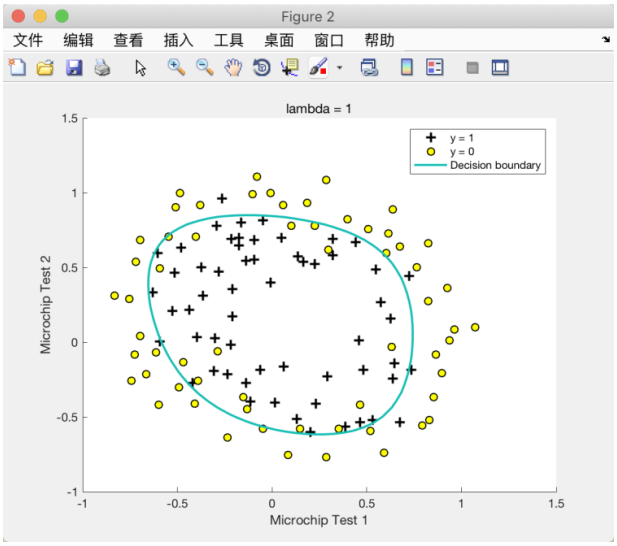
```
Cost at initial theta (zeros): 0.693147
Gradient at initial theta (zeros):
 -0.100000
 -12.009217
 -11.262842
```



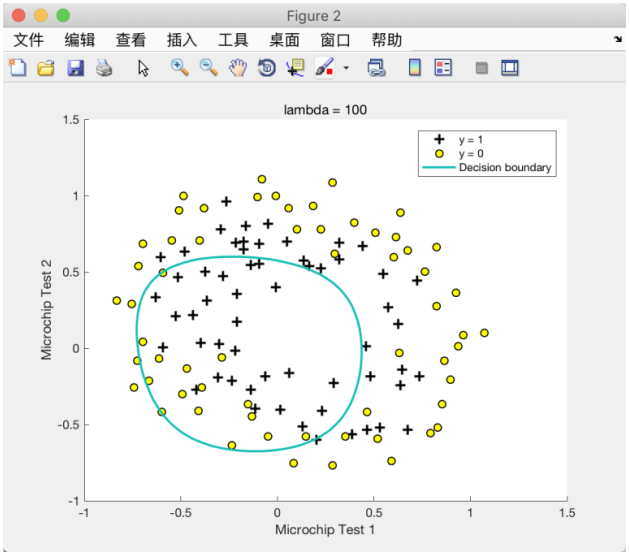

```
Train Accuracy: 89.000000
```

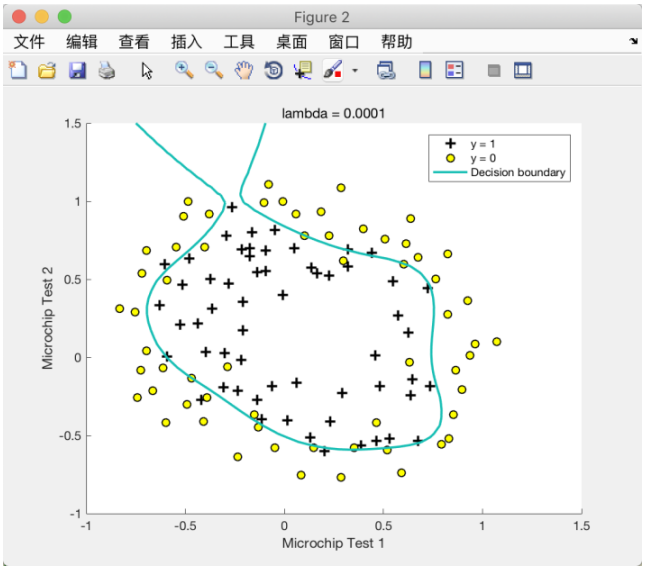
ex2_reg.m






```
Train Accuracy: 83.050847
```





2. 代码实现

a) costFunction.m

代码：

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

实现功能：计算逻辑回归的损失函数以及梯度结果

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

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

b) costFunctionReg.m

代码：

```
J = 1 / m * (- y' * log(sigmoid(X * theta)) - (1.- y)' * log(1.- sigmoid(X * theta))) + lambda / (2 * m) * theta(2:end,end)' * theta(2:end,end);
grad(1,1) = 1 / m * (X(:,1)' * (sigmoid(X * theta) - y))
grad(2:end,1) = (1 / m * (X(:,2:end)' * (sigmoid(X * theta) - y)) + lambda / m * theta(2:end));
```

实现功能:正则化逻辑回归的损失函数和梯度(Regularized logistic regression)

c) mapFeature.m

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

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

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

代码：

```
degree = 6;
out = ones(size(X1(:,1)));
for i = 1:degree
    for j = 0:i
        out(:, end+1) = (X1.^(i-j)).*(X2.^j);
    end
end
```

实现功能:实现特征映射

一种更好地确定数据的方法是从每个数据点创建更多的特性。在提供的函数 mapFeature.m 中，我们将把这些特征映射到 x_1 和 x_2 的所有多项式项中，直到第六次幂。

d) plotData.m

代码:

```
% 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);
```

实现功能:绘制数据正负类样本分布图

e) plotDecisionBoundary.m

代码:

```
% Plot Data
plotData(X(:,2:3), y);
hold on
if size(X, 2) <= 3
    % Only need 2 points to define a line, so choose two endpoints
    plot_x = [min(X(:,2))-2,  max(X(:,2))+2];
    % Calculate the decision boundary line
    plot_y = (-1./theta(3)).*(theta(2).*plot_x + theta(1));
    % Plot, and adjust axes for better viewing
    plot(plot_x, plot_y)

    % Legend, specific for the exercise
    legend('Admitted', 'Not admitted', 'Decision Boundary')
    axis([30, 100, 30, 100])
else
    % Here is the grid range
    u = linspace(-1, 1.5, 50);
    v = linspace(-1, 1.5, 50);
    z = zeros(length(u), length(v));
    % Evaluate z = theta*x over the grid
    for i = 1:length(u)
```

```
        for j = 1:length(v)
            z(i,j) = mapFeature(u(i), v(j))*theta;
        end
    end
    z = z'; % important to transpose z before calling contour
    % Plot z = 0
    % Notice you need to specify the range [0, 0]
    contour(u, v, z, [0, 0], 'LineWidth', 2)
end
```

实现功能:绘制决策边界

f) predict.m

代码:

```
p = (sigmoid(X * theta) >= 0.5);
```

实现功能:预测类别标签

g) Sigmoid.m

代码:

```
e=exp(1);
g = 1 ./ (1.+(e .^ (-1 .* z)))
```

实现功能: Sigmoid 函数计算

$$g(z) = \frac{1}{1+e^{-z}}.$$

三、 小结（包括通过本内容的认识以及其他）

1. 认识:

1) 逻辑回归(Logistic Regression)线性回归(Linear Regression)的区别如下：

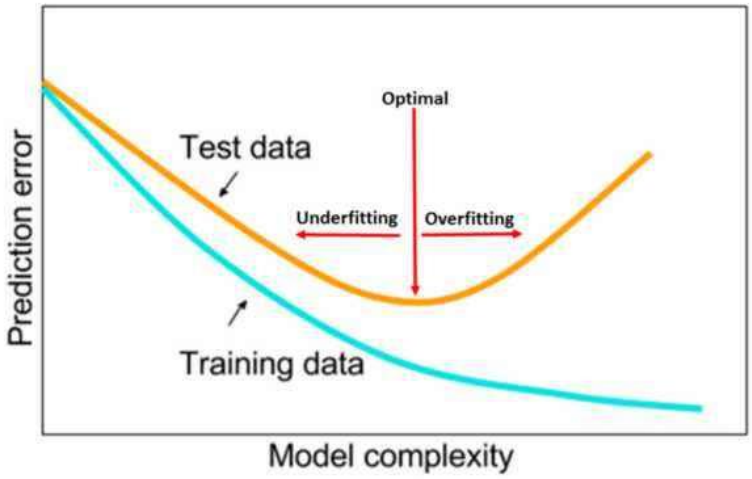
- 普通线性回归主要用于连续变量的**预测**，即，线性回归的输出 y 的取值范围是整个实数区间（y∈R）
- 逻辑回归用于离散变量的**分类**，即它的输出 y 的取值范围是一个离散的集合，主要用于类的判别，而且其输出值 y 表示属于某一类的概率.逻辑回归主要用于分类问题，常用来预测概率，如知道一个人的年龄、体重、身

高、血压等信息，预测其患心脏病的概率是多少.经典的 LR 用于二分类问题.

- 损失函数(cost function):
  - 线性回归: **平方损失函数（quadratic loss function)**
  - 逻辑回归: **对数似然损失函数(log-likehood loss function)**

2) 过拟合(Overfitting),欠拟合(Underfitting)以及正则化(Regularization)

- 过拟合: 当某个模型过度的学习训练数据中的细节和噪音，以至于模型在新的数据上(**预测**)表现很差.常常采用重采样技术 **k 折交叉验证**,**正则化**等方法应对.
- 欠拟合: 欠拟合指的是模型在**训练**和**预测**时表现都不好的情况.应对方法是继续学习并且试着更换机器学习算法.



- 正则化: 对于规模庞大的特征集,重要的特征可能并不多,所以需要减少无关特征的影响,减少后的模型也会有更强的可解释性;L_2 正则可以用来减小权重参数的值,当权重参数取值很大时,导致其导数或者说斜率也会很大,斜率偏大会使模型在较小的区间里产生较大的波动.加入 L_2 正则后,可使得到的模型更平滑.

2. 心得:

1) 适用场景:分类问题(传统逻辑回归适合二分类问题,softmax 回归以及多个逻辑回归分类器适合多分类问题)

2) 损失函数以及梯度下降法:

   a) 使用对数似然损失函数保证逻辑回归是凸函数(convex logistic regression cost function),保证梯度下降法的适用.

   b) 参数调整参照逻辑回归.

3) 特征映射: 一种更好地确定数据的方法是从每个数据点创建更多的特性。高的 Degree 可以创建更多特征.

4) 关于**正则化系数(lamda)**:

- 使用一个小的 λ,分类器几乎可以得到每个训练示例的正确结果,但绘制了一个非常复杂的边界,从而过度定义了数据.这不是一个好的决策边界(如图 3-1)它预测 X=(-0.25,1.5)处的一个点被接受(Y=1),这似乎是给定训练集的错误决策.

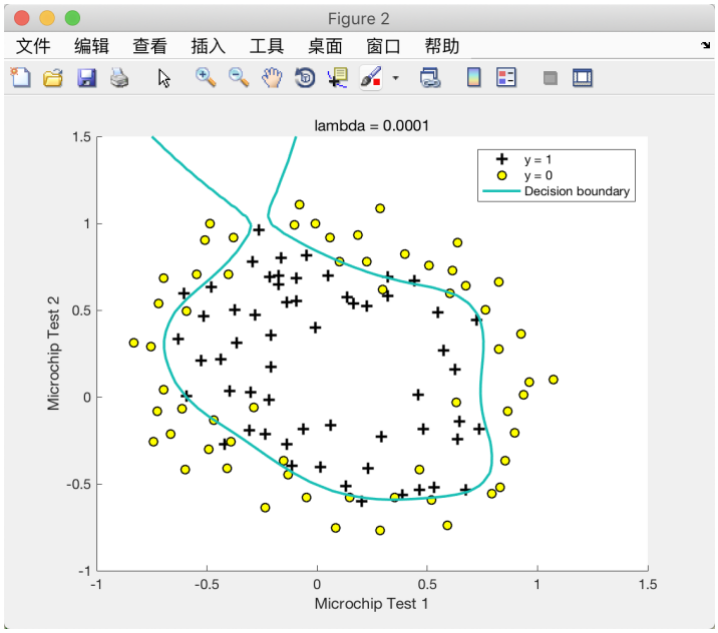


Figure3-1

- 对于较大的 λ,会产生一个更简单的决策边界,它仍然能够很好地区分正负。(如图 3-2)然而,如果将 λ 设置为过高的值,将得不到很好的结果,决策边界也不会很好地跟踪数据,从而导致数据的欠拟合.(如图 3-3)

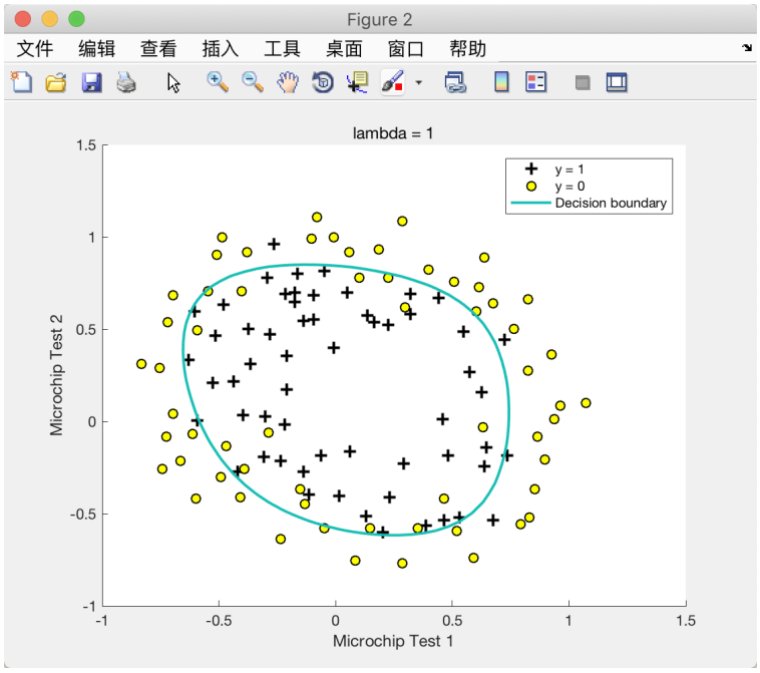


Figure3-2

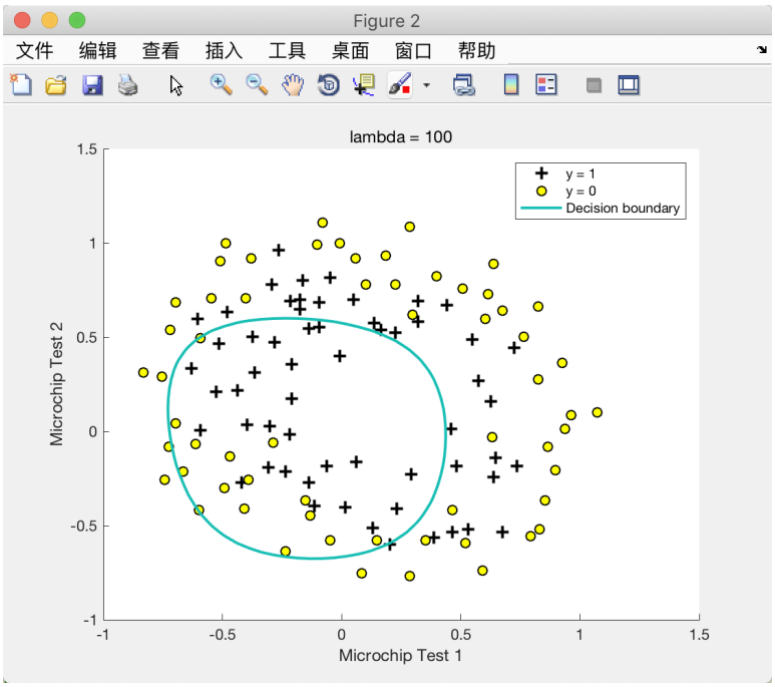


Figure3-3