logistic回归是回归分析的一种，函数表达式为   
y = 1/(1+exp(-x))   
在matlab中可以画出其graph：

x = -10:0.1:10;

y = 1./(exp(-x)+1);

plot(x,y,'g-x');

title('logistic function');

xlabel('x');ylabel('y');

以上是一维的情况。对于多维变量，可以定义一个超平面 代入原来的变量x中，得到：对于任意变量x，可以代入上式计算出y值并与0.5比较进行分类，　分类式为：

其中sgn(x) 为符号函数。

为了演示logistic函数是版怎样用于分类的，　假定我们有一组数据,分别对应的类别为。　定义平方和（或L2－norm)代价函数为：

通过最小化代价函数可以得到模型的参数w和b 。最小化的方法有很多种，　在下面的代码中给出一个最简单的梯度下降法。其基本思想是利用代价函数对w和b的一阶导数。关于导数如何求得请大家参考下面的Matlab代码。

%% generate random data

shift = 2;

n = 2;%2 dimN = 200;

x = [randn(n,N/2)-shift, randn(n,N/2)\*2+shift];

y = [zeros(N/2,1);ones(N/2,1)];

%show the data

figure;

plot(x(1,1:N/2),x(2,1:N/2),'rs');

hold on;

plot(x(1,1+N/2:N),x(2,1+N/2:N),'go');

title('2d training data');

上述code segment运行结果得到：

有了数据和模型，下面的代码将进行模型参数（即w和b)估计：

function [w,b,cost] = logistic\_train(x,y,tol,max\_iter)

fprintf('training started...\n');

n = size(x,1);

N = size(x,2);

w = ones(n,1)/n;

b = 1;

cost = [];

count = 1;while 1

*%find gradient*

partial\_w = zeros(n,1);

partial\_b = 0;

for i=1:N

a = exp(w'\*x(:,i)+b);

partial\_w = partial\_w + (1/(a+1)-y(i))\*(-1)/((1+a)\*(1+a))\*a\*x(:,i);

partial\_b = partial\_b + (1/(a+1)-y(i))\*(-1)/((1+a)\*(1+a))\*a;

end

*%find step size*

old\_cost = logistic\_cost(x,y,w,b);

step = 1;

while step > 1e-12

w1 = w - step\*partial\_w;

b1 = b - step\*partial\_b;

new\_cost = logistic\_cost(x,y,w1,b1);

if new\_cost < old\_cost

break;

end

step = step \* 0.1;

end

if step <= 1e-12

fprintf('finished seraching the step size\n');

break;

end

w = w1;

b = b1;

cost = [cost,new\_cost];

if new\_cost < tol

fprintf('converged after %d iterates!',count);

break;

end

if count > max\_iter

fprintf('training stoped after %d iterates, not converged to desired precision!',count);

break;

end

count = count + 1;

end

调用上面logistic\_train函数就能得到w,b。　如前述，训练方法为梯度下降法。

*%%%training..*[w,b,cost] = logistic\_train(x,y,1e-6,100);*%%show convergence cureve*disp('training ended');

figure;

plot(cost,'g-s');

xlabel('iterate');

ylabel('cost');

set(gca,'YScale','log');

title('convergence curve');

返回值中的cost为每一步迭代后的代价值，其曲线图如下：

训练结束后，得到的w = [ -30.1994, -30.4356], b = [2.7303]. （由于样本是随机人工生成的，每次得到的w,b都会有些不同。）

　最后，有了模型（即logistic function)和模型参数w,b，　就能进行分类了。下面的代码演示训练样本进行分类

%% plot the training data

figure;

plot(x(1,1:N/2),x(2,1:N/2),'rs');

hold on;

plot(x(1,N/2+1:N),x(2,N/2+1:N),'go');

%% visualize the classification area

hold on;for x = -shift\*5:0.5:shift\*5

for y=-shift\*5:0.5:shift\*5

if 1/(1+exp(w(1:2)'\*[x;y]+b)) > 0.5

plot(x,y,'g.');

else

plot(x,y,'r.');

end

end

end

title('classification result on training data');