山东大学 ______计算机科学与技术______ 学院

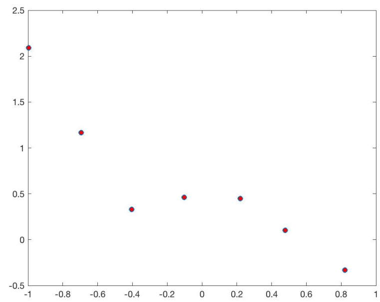
机器学习与模式识别 课程实验报告

学号：201600301304	姓名：贾乘兴	班级： 人工智能 16
实验题目：正则化		
实验学时：2 小时	实验日期： 2018/11/9	
实验目的：在给定的数据集下，实现线性回归与逻辑回归的正则化		
硬件环境： 16 GB 内存		
软件环境：mac os，matlab 2017b		

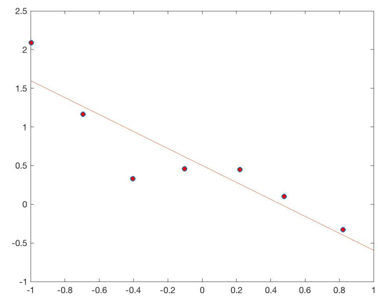
实验步骤与内容：

一．正则化的线性回归

1. 问题分析：在线性回归问题中，对于一些较为复杂的非线性的点的分布，如下图



如果我们仍然使用线性的假设作线性回归，那么得到的参数结果，可知线性函数 y=wx+b 的形式显然是误差较大的，如下图



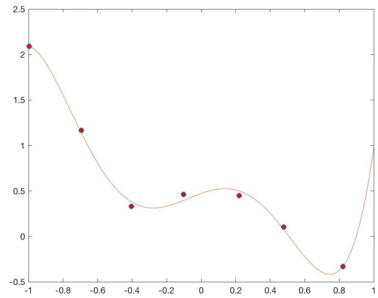
所以我们认为该分布用一些更为复杂的函数作为假设会更合适一些，在该分布下，我们认为使用高次的函数作为假设会更好，在这里我们假设分布满足五次函数的模型，五次函数的假设如下

$$h_\theta(x)=\theta_0+\theta_1x+\theta_2x^2+\theta_3x^3+\theta_4x^4+\theta_5x^5=\sum_{i=0}^{5}\theta_ix^i$$

在该假设下，我们使用一维的数据生成了六维的样本（常数维计作一维下），线性回归得到参数的解如下

$$\theta=\left(X^TX\right)^{-1}X^Ty$$

最终我们可以得到一个非常符合给定数据的曲线 h



但是，在实际问题中，很多时候数据存在大量的噪声，使得我们的算法在学习函数的过程中不只学到了该问题下的 feature，还学到了训练集自己特有的的 feature，这样就会导致得到的模型的泛化能力不够强，在训练集的误差较小，但在测试集的误差反而会增大，所以这个时候需要我们进行特征选择，正则化是一种较好的嵌入式特征选择的方法，在线性回归时，我们对目标函数加入一个正则项如下

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

这样做的好处就是，对于一些不重要的 feature，在训练过程中，参数对应的那一项减小对目标函数的最小化来说，较前面的 l2 误差的增大来讲更明显，这样就起到了减小不重要的 feature 的权重，抑制过拟合的作用，其中 lamda 为人为给定的超参数，用于调整正则项的比重的，这样我们得到新的线性回归的参数结果如下

$$\theta=(X^TX+\lambda\begin{bmatrix}0 & & & \\ & 1 & & \\ & & \cdots & \\ & & & 1\end{bmatrix})^{-1}X^Ty$$

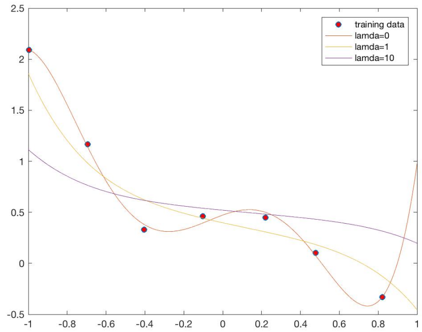
2. 实验过程

首先导入数据，训练集导入，实现正则化下的线性回归，然后对不同的 lamda 进行测试，观察不同的 lamda 对应的图像，lamda 分别取 0（无正则化）、1、10，实验部分代码如下所示

```
clear,clc;
%% liner data load and preprocess
x=load('ex5Linx.dat');
y=load('ex5Liny.dat');
figure
plot(x,y,'o','MarkerFaceColor','r');
m=length(y);
X=[ones(m,1),x,x.^2,x.^3,x.^4,x.^5];
I0=eye(6);
I0(1,1)=0;
%% loss=(sum((X*theta-y).^2)+lamda*theta'*I0*theta)/2m
% lamda=0
lamda=0;
theta=inv(X'*X+lamda*I0)*X'*y;
hold on
x0=[-1:0.01:1]';
m0=length(x0);
x0=[ones(m0,1),x0,x0.^2,x0.^3,x0.^4,x0.^5];
plot(x0(:,2),x0*theta,'-');
% lamda=1
lamda=1;
theta=inv(X'*X+lamda*I0)*X'*y;
hold on
x0=[-1:0.01:1]';
m0=length(x0);
x0=[ones(m0,1),x0,x0.^2,x0.^3,x0.^4,x0.^5];
plot(x0(:,2),x0*theta,'-');
% lamda=10
lamda=10;
theta=inv(X'*X+lamda*I0)*X'*y;
```

```
hold on
x0=[-1:0.01:1]';
m0=length(x0);
x0=[ones(m0,1),x0,x0.^2,x0.^3,x0.^4,x0.^5];
plot(x0(:,2),x0*theta,'-');
legend('training data','lamda=0','lamda=1','lamda=10');
```

3. 实验结果，不同取值的曲线绘制图像的对比如下



4. 结果分析：可见随着 lamda 增大，得到的曲线越平滑，lamda 等于 0 时易出现过拟合问题，但是如果 lamda 过大也会出现问题，我们的假设将像一开始的线性函数一样较大的偏离实际，所以只有在 lamda 给定的较好的情形下（如 lamda=1），训练效果才算是不错

二．正则化的逻辑回归

1. 问题分析：在逻辑回归中我们也可以采用类似的形式来进行特征选择，选取较好的特征逻辑回归的假设如下

$$h_\theta(x)=g\left(\theta^T x\right)=\frac{1}{1+e^{-\theta^T x}}=P\left(y=1|x;\theta\right)$$

在目标函数中加入正则项，新的目标函数如下

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

在该问题下我们无法得到解析解，我们需要用梯度下降的方法进行求解

$$\theta:=\theta-H^{-1}\nabla_\theta J$$

其中，H 为 hession 矩阵

$$H=\frac{1}{m}\left[\sum_{i=1}^{m}h_\theta\left(x_i\right)\left(1-h_\theta\left(x_i\right)\right)x_i x_i^T\right]+\frac{\lambda}{m}\begin{bmatrix}0 & & & \\ & 1 & & \\ & & \dots & \\ & & & 1\end{bmatrix}$$

而损失函数对 theta 求导得到

$$\nabla_\theta J=\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x_i\right)-y_i\right)x_i+\begin{bmatrix}0 & & & \\ & 1 & & \\ & & \dots & \\ & & & 1\end{bmatrix}\theta$$

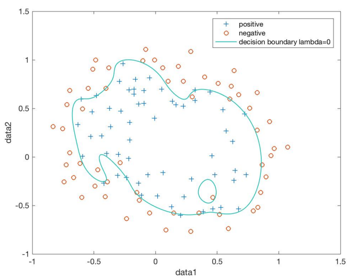
其中我们的 x 是由两个维度特征生成的 28 维的特征，最后训练得到的参数的结果，我们取 theta*x=0 的曲线，该曲线下，假设函数的取值为 0.5，即概率为 0.5，该曲线可以作为分类的曲线。

2. 实验过程

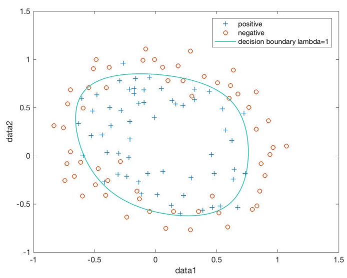
首先导入数据，训练集导入，实现正则化下的逻辑回归，然后对不同的 lamda 进行测试，观察不同的 lamda 对应的图像， lamda 分别取 0（无正则化）、1、10，实验部分代码如下所示

```
%% log data load and preprocess
clear,clc;
x=load('ex5Logx.dat');
y=load('ex5Logy.dat');
m=length(y);
pos=find(y==1);
neg=find(y==0);
figure
plot(x(pos,1),x(pos,2),'+');
hold on
plot(x(neg,1),x(neg,2),'o');
xlabel('data1');
ylabel('data2');

%% loss=-(1/m)*(ylog(h)+(1-y)log(1-h))+(2/m)*lamda*theta(2:n)^2
I=eye(28);
I(1,1)=0;
X=map_feature(x(:,1),x(:,2));

%% regulization
num=500;
%% lamda=0 %% 1 10
lamda=0;
theta=zeros(28,1);
for j=1:num
    h=1./(1+exp(-X*theta));
    H=(1/m)*(X'*(h.*(1-h).*X)+lamda*I);
    invH=inv(H);
    theta=theta-invH*(1/m)*(X'*(h-y)+lamda*I*theta);
end
% plot
u=linspace(-1,1.5,200);
v=linspace(-1,1.5,200);
z=zeros(length(u),length(v));
for j=1:length(u)
    for k=1:length(v)
        z(j,k)=map_feature(u(j),v(k))*theta;
    end
end
hold on
contour(u,v,z',[0,0],'linewidth',1);
```

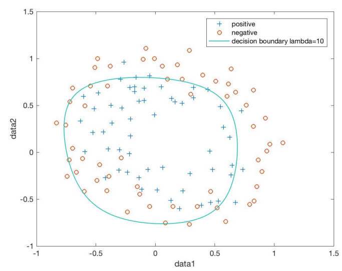
3. 实验结果与分析：不同 lambda 取值结果如下



lambda=0（无正则）

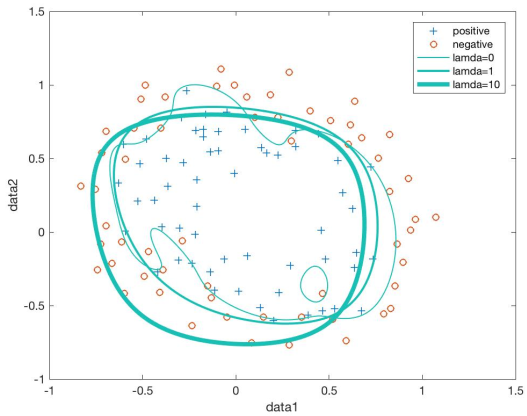


lambda=1



lambda=10

将各个取值边界进行对比图像如下



可见，当不加入正则化时，该函数学习到了噪声数据的一些固有的特征，而正则化系数较大时，该边界学习效果并不好，取合适的正则化系数分类效果较好
结论分析与体会： 通过本次实验，对正则化的学习，对正则化对结果的影响有了较为直观的认识，同时，复习了线性回归与逻辑回归

附录：程序源代码

ex5.m

```
clear,clc;
%% liner data load and preprocess
x=load('ex5Linx.dat');
y=load('ex5Liny.dat');
figure
plot(x,y,'o','MarkerFaceColor','r');

m=length(y);
X=[ones(m,1),x,x.^2,x.^3,x.^4,x.^5];
I0=eye(6);
I0(1,1)=0;

%% loss=(sum((X*theta-y).^2)+lamda*theta'*I0*theta)/2m
% lamda=0
lamda=0;
theta=inv(X'*X+lamda*I0)*X'*y;
hold on
x0=[-1:0.01:1]';
m0=length(x0);
x0=[ones(m0,1),x0,x0.^2,x0.^3,x0.^4,x0.^5];
plot(x0(:,2),x0*theta,'-');

% lamda=1
lamda=1;
theta=inv(X'*X+lamda*I0)*X'*y;
hold on
x0=[-1:0.01:1]';
m0=length(x0);
x0=[ones(m0,1),x0,x0.^2,x0.^3,x0.^4,x0.^5];
```

```
plot(x0(:,2),x0*theta,'-');

% lamda=10
lamda=10;
theta=inv(X'*X+lamda*I0)*X'*y;
hold on
x0=[-1:0.01:1]';
m0=length(x0);
x0=[ones(m0,1),x0,x0.^2,x0.^3,x0.^4,x0.^5];
plot(x0(:,2),x0*theta,'-');

legend('training data','lamda=0','lamda=1','lamda=10');

%% log data load and preprocess
clear,clc;
x=load('ex5Logx.dat');
y=load('ex5Logy.dat');
m=length(y);
pos=find(y==1);
neg=find(y==0);
figure
plot(x(pos,1),x(pos,2),'+');
hold on
plot(x(neg,1),x(neg,2),'o');
xlabel('data1');
ylabel('data2');

%% loss=-(1/m)*(ylog(h)+(1-y)log(1-h))+(2/m)*lamda*theta(2:n)^2
I=eye(28);
I(1,1)=0;
X=map_feature(x(:,1),x(:,2));

%% regulization
num=500;
%% lamda=0
lamda=0;
theta=zeros(28,1);
for j=1:num
    h=1./(1+exp(-X*theta));
    H=(1/m)*(X'*(h.*(1-h).*X)+lamda*I);
    invH=inv(H);
    theta=theta-invH*(1/m)*(X'*(h-y)+lamda*I*theta);
end
% plot
u=linspace(-1,1.5,200);
v=linspace(-1,1.5,200);
```

```
z=zeros(length(u),length(v));
for j=1:length(u)
    for k=1:length(v)
        z(j,k)=map_feature(u(j),v(k))*theta;
    end
end
hold on
contour(u,v,z',[0,0],'linewidth',1);

%% lamda=1
lamda=1;
theta=zeros(28,1);
for j=1:num
    h=1./(1+exp(-X*theta));
    H=(1/m)*(X'*(h.*(1-h).*X)+lamda*I);
    invH=inv(H);
    theta=theta-invH*(1/m)*(X'*(h-y)+lamda*I*theta);
end
% plot
u=linspace(-1,1.5,200);
v=linspace(-1,1.5,200);
z=zeros(length(u),length(v));
for j=1:length(u)
    for k=1:length(v)
        z(j,k)=map_feature(u(j),v(k))*theta;
    end
end
hold on
contour(u,v,z',[0,0],'linewidth',2);

%% lamda=10
lamda=10;
theta=zeros(28,1);
for j=1:num
    h=1./(1+exp(-X*theta));
    H=(1/m)*(X'*(h.*(1-h).*X)+lamda*I);
    invH=inv(H);
    theta=theta-invH*(1/m)*(X'*(h-y)+lamda*I*theta);
end
% plot
u=linspace(-1,1.5,200);
v=linspace(-1,1.5,200);
z=zeros(length(u),length(v));
for j=1:length(u)
    for k=1:length(v)
        z(j,k)=map_feature(u(j),v(k))*theta;
```

```
    end
end
hold on
contour(u,v,z',[0,0],'linewidth',4);

legend('positive','negative','lamda=0','lamda=1','lamda=10');
```