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：基于牛顿法迭代的逻辑回归 | | | |
| 实验学时：2小时 | | 实验日期： 2018/10/22 | |
| 实验目的：本实验是在给定的两个feature（两门考试成绩）下，预测是否被通过（y=0或y=1）的问题。在给定的数据集x（两个特征）与标签y下，实现逻辑回归，并牛顿法的数值方法进行参数的更新，最终得到参数，并进行预测 | | | |
| 硬件环境： 16 GB 内存 | | | |
| 软件环境：mac os，matlab 2017b | | | |
| 实验步骤与内容：   1. 逻辑回归 2. 在线性回归的假设中，我们令y=wx+b，对应y的值域在本问题下显然不合适，本问题是通过或者不通过的两个标签的问题，是个离散的结果，我们的假设的函数是连续的，所以可以将该假设变为预测问题，得到的结果为概率。 3. 适合作为概率预测的函数，我们选择了sigmoid函数，sigmoid函数如下：     sigmoid函数具有对称性等良好性质，可以将z=wx+b映射在0到1的区间内，然后我们将其二值化，大于0.5为1，小于0.5为0，0.5自由处理，这样就实现了逻辑回归。   1. 逻辑回归问题中，我们定义的目标函数如下：     观察可知，该函数在h与y最接近时是最大的，所以我们的目的是最大化该函数，得到的h才能接近真实的y。但该函数在求导和计算上有一些问题和困难，容易出现向下溢出等等问题，而且也与一般问题中最小化目标函数不同，所以我们对其取对数并取反，得到了我们定义的损失函数如下：    对损失函数求导可得：     1. 牛顿法参数更新 2. 在之前梯度下降中，我们参数的更新公式为：     其中埃塔为学习率，是超参数，我们设置了迭代停止的条件如下：    其中埃普西隆为设置的阈值，该问题设置的阈值为1e-9，在学习过程中我们发现最好的学习率下都需要几百次迭代，我们将数据标准化后迭代次数达到了10次（5.83的学习率下）。   1. 本实验我们使用牛顿法的数值方法进行参数更新，更新公式如下：     其中H为hession矩阵，对L各个参数的二阶求导组成的矩阵，hession矩阵公式如下:    在该方法下，我们只需要8次就可以达到收敛。   1. 边界确定   在逻辑回归中分界线是h=0.5，对应的是theta\*x=0，得到参数后将直线theta\*x=0在图像上表示出来  四．实验内容与部分代码   1. 数据读取、分类与显示：   %% data load  x=load('ex4x.dat');  y=load('ex4y.dat');  m=length(y);  X=[ones(m,1),x];    %% show the data  pos=find(y==1);  neg=find(y==0);  figure;  plot(X(pos,2),X(pos,3),'+');  hold on  plot(X(neg,2),X(neg,3),'o');  xlabel('exam 1 score');  ylabel('exam 2 scroe');   1. 牛顿法迭代代码如下：   %% train(newton's method)  e=1e-9;  theta=zeros(3,1);  loss=zeros(1,5);  num=1;  loss(1,num)=inf;  % repeat  num=num+1;  h=1./(1+exp(-X\*theta));  H=(1/m)\*X'\*(h.\*(1-h).\*X);  hinv=inv(H);  loss(1,num)=-(1/m)\*sum((1-y).\*log(1-h)+y.\*log(h));  theta=theta-hinv\*(1/m)\*(X'\*(h-y));  while abs(loss(1,num)-loss(1,num-1))>e  num=num+1;  h=1./(1+exp(-X\*theta));  H=(1/m)\*X'\*(h.\*(1-h).\*X);  hinv=inv(H);  loss(1,num)=-(1/m)\*sum((1-y).\*log(1-h)+y.\*log(h));  theta=theta-hinv\*(1/m)\*(X'\*(h-y));  end   1. 训练结果图像绘制：   % theta\*X=0,show the decision boundary(newton's method)  x1=[15:0.1:65];  x2=(theta(1)+theta(2)\*x1)/(-theta(3));  hold on  plot(x1,x2,'-');  legend('admitted','not admitted','boundary');    %% plot  figure  plot([1:num],loss,'-');  xlabel('iteration');  ylabel('loss');  legend('newton');  得到关于边界和迭代过程中误差变化的图像如下：  ../../f1.jpg ../../fa.jpg   1. 问题求解：   %% pre  x0=[1,20,80];  y0=1/(1+exp(-x0\*theta));    %% show the result  theta  1-y0  num  结果保留四位小数，最终得到的theta为：[-16.3787,0.1483,0.1589]  跌代次数为8  不被通过的概率为0.6680  五．问题改进与探究  我们对原数据进行标准化之后，设置学习率为5.830，与牛顿法收敛速度进行对比如下：  ../../fff.jpg  可见只有在最好的学习率与良好的数据条件下，手动设置学习率的梯度下降方法才能略差与牛顿法（迭代10次与迭代8次），而复杂的问题下找到最好的学习率是很困难的，所以牛顿法是效率极高的一种方法。 | | | |
| 结论分析与体会： 本次实验实现了逻辑回归问题，通过对牛顿法的梯度下降的实现，我们对优化逻辑回归有了更好的掌握 | | | |

附录：程序源代码

ex4.m

clear,clc;

%% data load

x=load('ex4x.dat');

y=load('ex4y.dat');

m=length(y);

X=[ones(m,1),x];

%% show the data

pos=find(y==1);

neg=find(y==0);

figure;

plot(X(pos,2),X(pos,3),'+');

hold on

plot(X(neg,2),X(neg,3),'o');

xlabel('exam 1 score');

ylabel('exam 2 scroe');

%% train(newton's method)

e=1e-9;

theta=zeros(3,1);

loss=zeros(1,5);

num=1;

loss(1,num)=inf;

% repeat

num=num+1;

h=1./(1+exp(-X\*theta));

H=(1/m)\*X'\*(h.\*(1-h).\*X);

hinv=inv(H);

loss(1,num)=-(1/m)\*sum((1-y).\*log(1-h)+y.\*log(h));

theta=theta-hinv\*(1/m)\*(X'\*(h-y));

while abs(loss(1,num)-loss(1,num-1))>e

num=num+1;

h=1./(1+exp(-X\*theta));

H=(1/m)\*X'\*(h.\*(1-h).\*X);

hinv=inv(H);

loss(1,num)=-(1/m)\*sum((1-y).\*log(1-h)+y.\*log(h));

theta=theta-hinv\*(1/m)\*(X'\*(h-y));

end

% theta\*X=0,show the decision boundary(newton's method)

x1=[15:0.1:65];

x2=(theta(1)+theta(2)\*x1)/(-theta(3));

hold on

plot(x1,x2,'-');

legend('admitted','not admitted','boundary');

%% plot

figure

plot([1:num],loss,'-');

xlabel('iteration');

ylabel('loss');

legend('newton');

%% pre

x0=[1,20,80];

y0=1/(1+exp(-x0\*theta));

%% show the result

theta

1-y0

num