山东大学 计算机科学与技术学院

机器学习课程实验报告

学　　号　　： 201700301042	姓名： 陈佳睿	班级：　17 智能班
实验题目：逻辑回归和牛顿法		
实验学时：2 学时	实验日期：　　2019.11.6	

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，并使用牛顿法进行参数 theta 的更新，最终得到对于数据拟合程度较好的参数，并对未在 train 数据集中新的数据进行预测

硬件环境：

MacBook pro 2017

8GB 内存

软件环境：

Macos ， matlab 2017_a

实验步骤与内容：

（以下的一部分实验报告使用 markdown 编写所以格式前后不太一致）

一．实验所给的数据的特点

ex4Data.zip 解压后得到 ex4x.dat 和 ex4y.dat 两个文件

一共有 40+40 个样本 （包括 40 个没有被大学录取的人和被大学录取的人）

ex4x.dat 中 两列数据表示每个人标化考试的成绩

ex4y.dat 中的数据是对应该同学有没有被大学录取的 label （值为 1 则被录取，值为 0 则没有被录取）

二．绘制所给的数据

add the x0 = 1 intercept term into your x matrix.

实验指导书指出要在我的 x 数据矩阵中加入 x0=1 的截断项

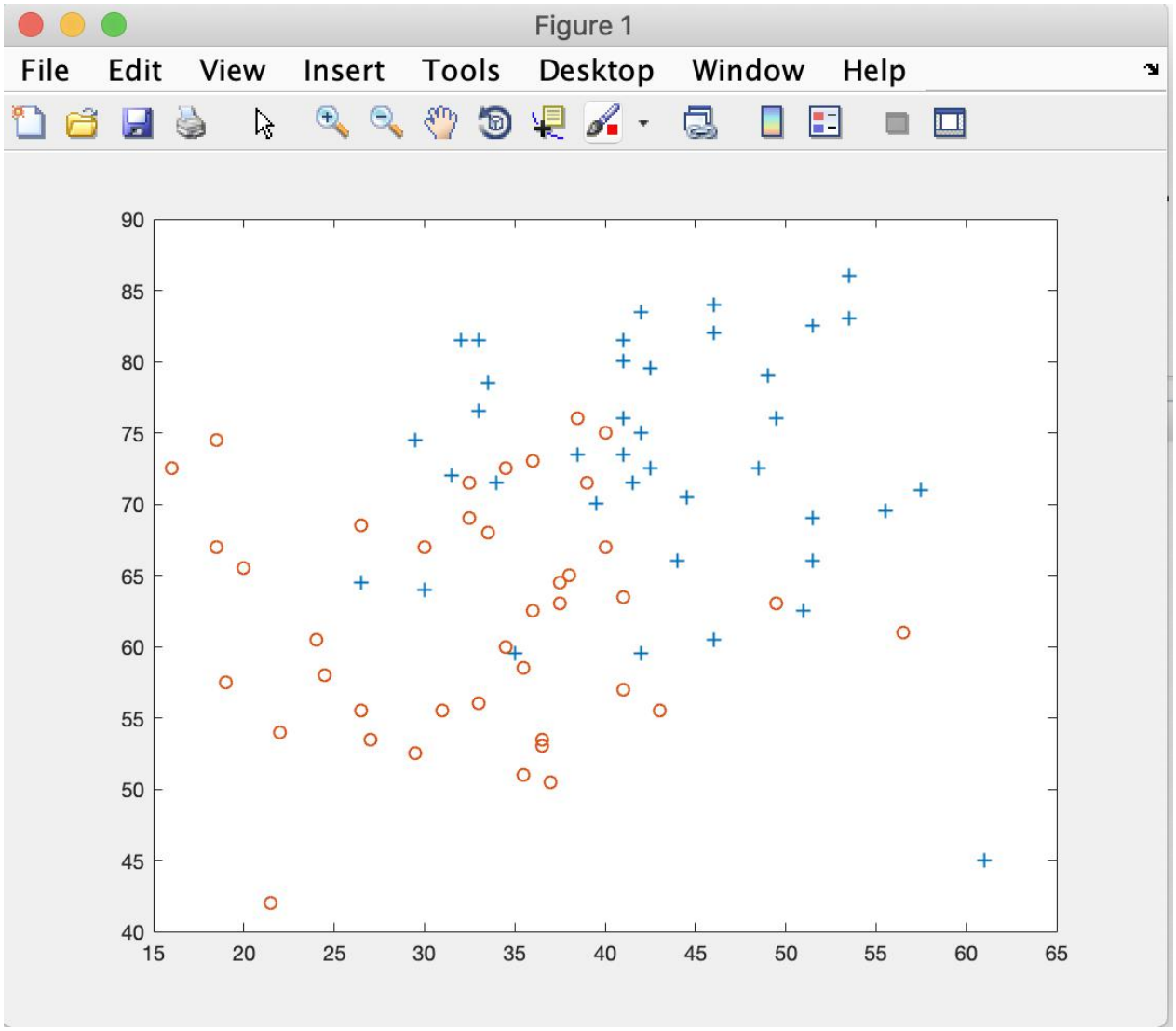
所以在实验中我对于 x 进行扩展操作

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

使用 find 命令来对正样本和负样本进行分类

在坐标系中绘制出 80 个点，正样本用‘+’标识，负样本使用‘o’标识

得到的效果图如下：



可以看出数据点是较为离散的一组值

Recall that in logistic regression, the hypothesis function is

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

在线性回归的假设中 ，我们需要学习的数值只有 theta
而我们的任务就是根据给定的数据和标签来进行学习和近似，来对新的测试数据提供更高的鲁棒性
因为我们得到的数据值是离散的但我们假设的函数是连续的，所以我们将假设问题变为预测问题，得到的结果为概率
我使用 sigmod 函数对于函数的值域进行压缩，将其取值放到【0，1】之间，然后将其二值化，大于 0.5 的为 1，小于 0.5 的记为 1，实现了我们对于数据的逻辑回归。
三．牛顿法
使用牛顿法，我们要最小化代价函数，通过循环迭代不断的对于目标函数的导函数求解零点，并最终将结果逼近到一个很小的区间之内，我们就默认函数在该点处即为所求

The cost function $J(\theta)$ is defined as

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

对于高维的情况，牛顿法的迭代公式为：

$$\theta^{(t+1)} = \theta^{(t)} - H^{-1}\nabla_\theta J$$

其中 H 为 hessian 矩阵，定义为

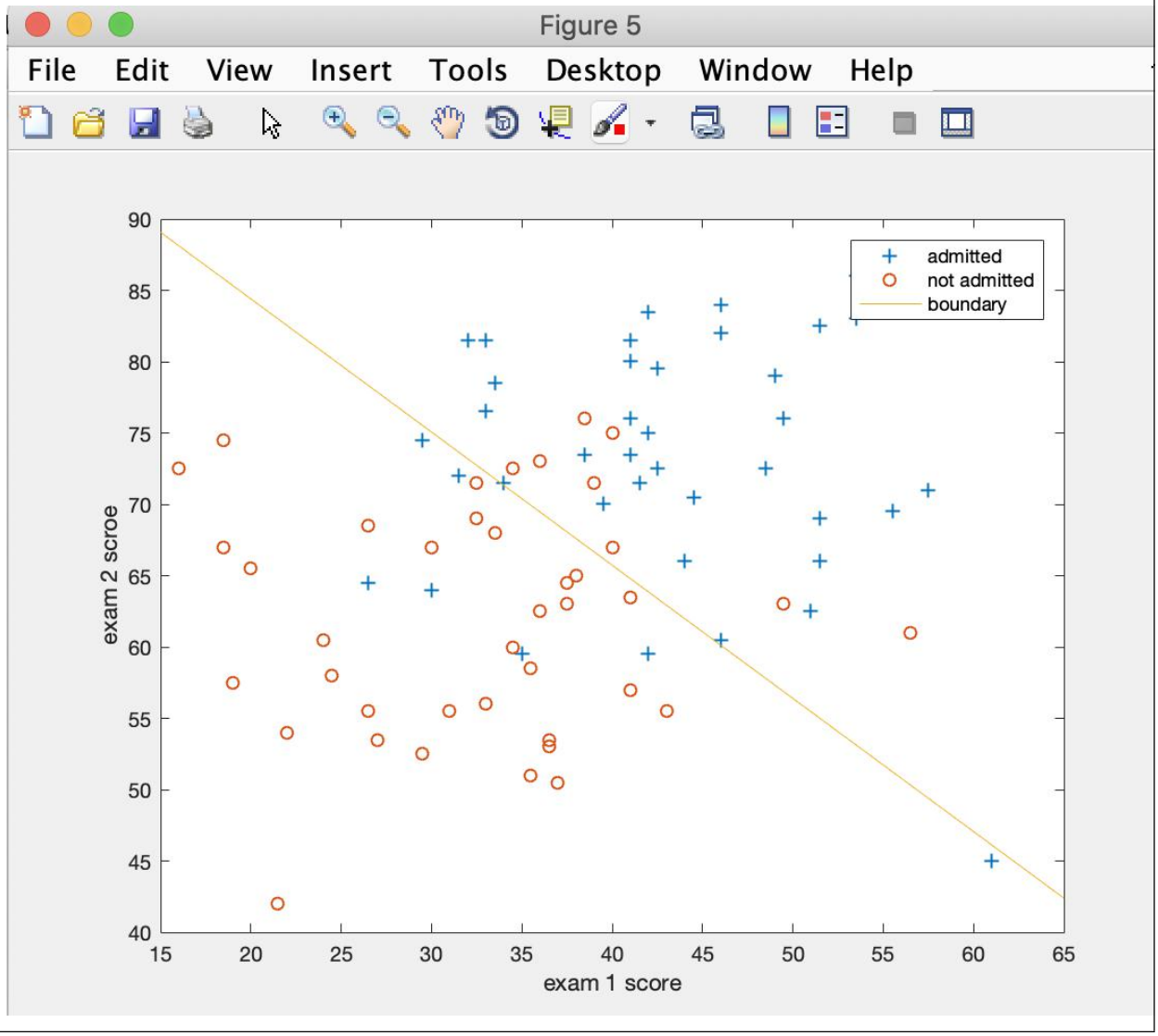
$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

$$H = \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x_i)\left(1-h_\theta(x_i)\right)x_i x_i^T \right]$$

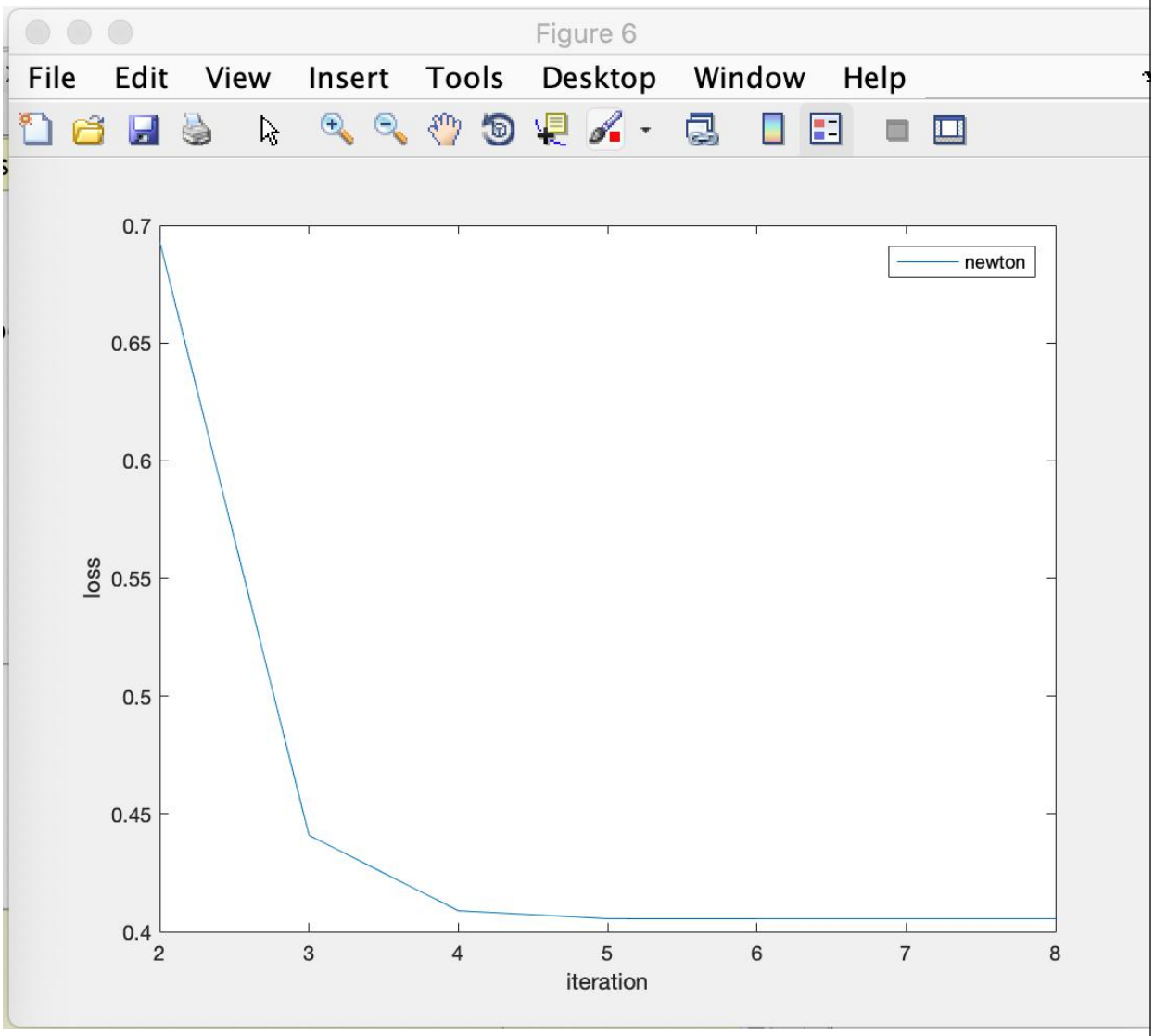
hessian 矩阵的更新方法如上所示
使用 matlab 内置的函数 inv 对于 hessian 矩阵求逆
接着根据上面所给的参数更新公式 我们进行迭代求解
并每次更新 loss 函数来判读是否已经可以认为此解接近真实解
以下是第一次循环得到的 hessian 矩阵的值

1	2	3
0.2500	9.4625	16.8453
9.4625	382.4656	643.6656
16.8453	643.6656	1.1592e...

使用牛顿法迭代得到两个类数据的边界



迭代过程中误差变化的图像如下：



实验指导书中提出的问题：

Finally, record your answers to these questions.

1. What values of θ did you get? How many iterations were required for convergence?
2. What is the probability that a student with a score of 20 on Exam 1 and a score of 80 on Exam 2 will not be admitted?

```
>> theta

theta =
```

```
   -16.3787
     0.1483
     0.1589

>> 1-y0

ans =0.6680

>> num

num =8
```

以上是使用迭代之后使用新数据预测得到的结果
迭代次数是 8 次
不被通过的概率为 0.6680

结论分析与体会：
本次试验使用牛顿法来求解 loss（代价函数）的最优化方法，并最终用于迭代更新求解的参数 theta，相比之前梯度下降的实验中我手动调节学习率来进行求解的方法要简便不少，对于二维的 hessian 矩阵我比较清晰，但是对于维度更高的 hessian 矩阵我的思路就相对变得比较模糊，只能跟着推导出的公式和我求解过程中变量的维度数来进行实验，总的来说，感觉机器学习的实验相比其他应用开发不太一样，中间结果的数学变量还是比较抽象。

附录：程序源代码

```
%% 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

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));
```