计算机科学与技术 学院

机器学习 课程实验报告

学号：	姓名：	班级：
实验题目：Multivariate Linear Regression		
实验学时：2.0	实验日期： 2018/9/30	

实验目的：

学习多元线性回归的基本算法应用，熟悉 matla 等软件的使用，锻炼解决实际问题的能力.

硬件环境：

操作系统

Windows 10 家庭中文版 64-bit

CPU

Intel Core i5 7200U @ 2.50GHz 41 ° C

Kaby Lake-U/Y 14nm 工艺

RAM

8.00GB 单个的-通道 未知 (15-15-15-35)

主板

HP 81D1 (U3E1)

图像

Generic PnP Monitor (1920x1080@60Hz)

Intel HD Graphics 620 (HP)

存储器

476GB NVMe THNSN5512GPUK TO (未知)

40GB Microsoft 虚拟磁盘 (File-backed Virtual)

光盘驱动器

没有检测到光纤磁盘驱动

音频

Conexant ISST Audio

软件环境：

Win10 + matlabR2016a

实验步骤与内容：

2.1 载入数据集：

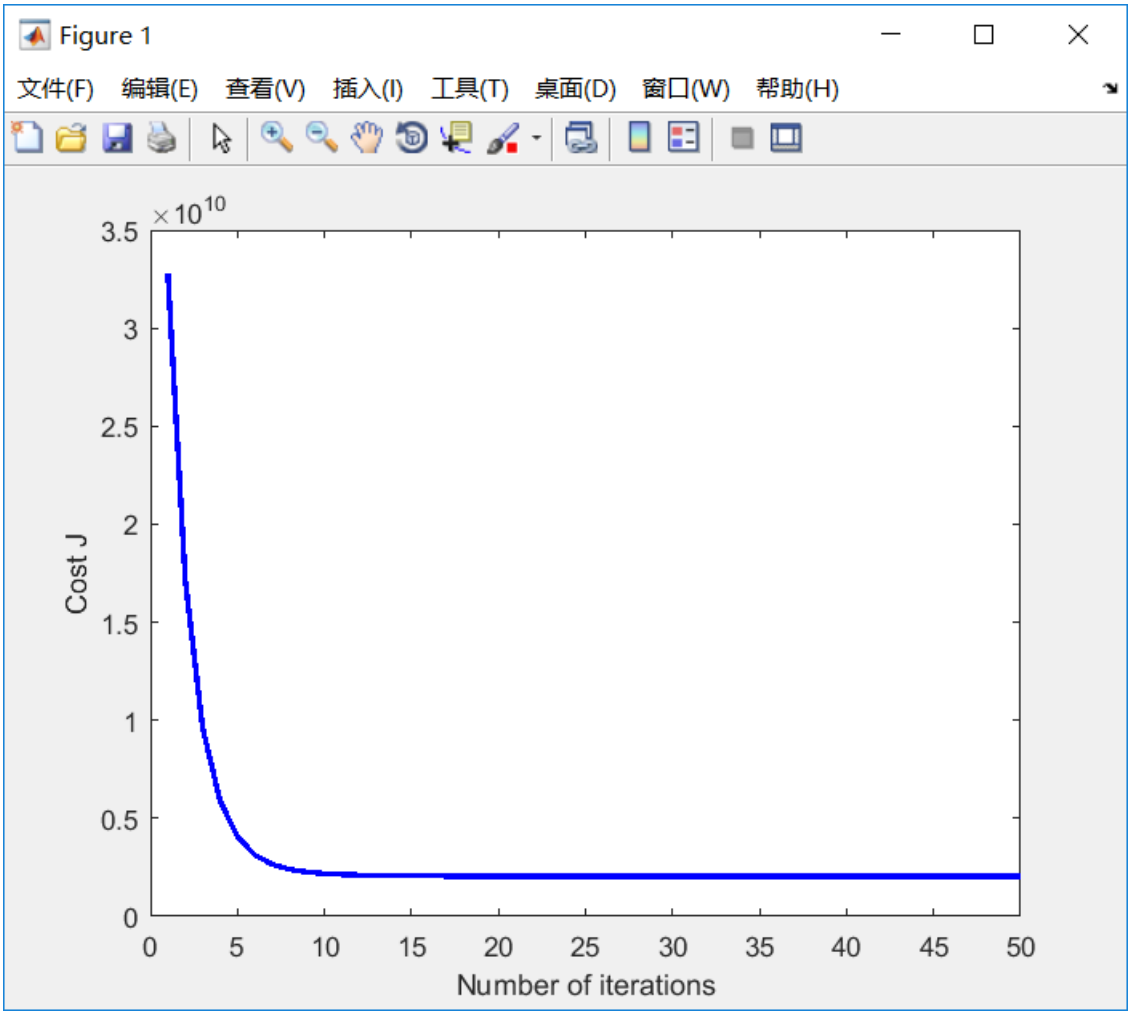
```
>> x=load('ex2x.dat');
>> y=load('ex2y.dat');
>> m=length(y);
```

2.2 数据归一化：

```
>> [X mu sigma]=featureNormalize(x);
>> X=[ones(m,1),X]
```

2.3 进行梯度下降：

```
>> alpha=0.3;
>> theta=zeros(3,1);
>> [theta, J_history] = gradientDescentMulti(X, y, theta, alpha,
num_iters);
>> figure;
>> plot(1:numel(J_history), J_history, '-b', 'LineWidth', 2);
>> xlabel('Number of iterations');
>> ylabel('Cost J');
>> hold on;
```




```
>> alpha=0.2;
>> theta=zeros(3,1);
>> [theta, J_history] = gradientDescentMulti(X, y, theta, alpha,
num_iters);
>> plot(1:numel(J_history), J_history, '-r', 'LineWidth', 2);
>> hold on;
>> alpha=0.1;
>> theta=zeros(3,1);
>> [theta, J_history] = gradientDescentMulti(X, y, theta, alpha,
num_iters);
```

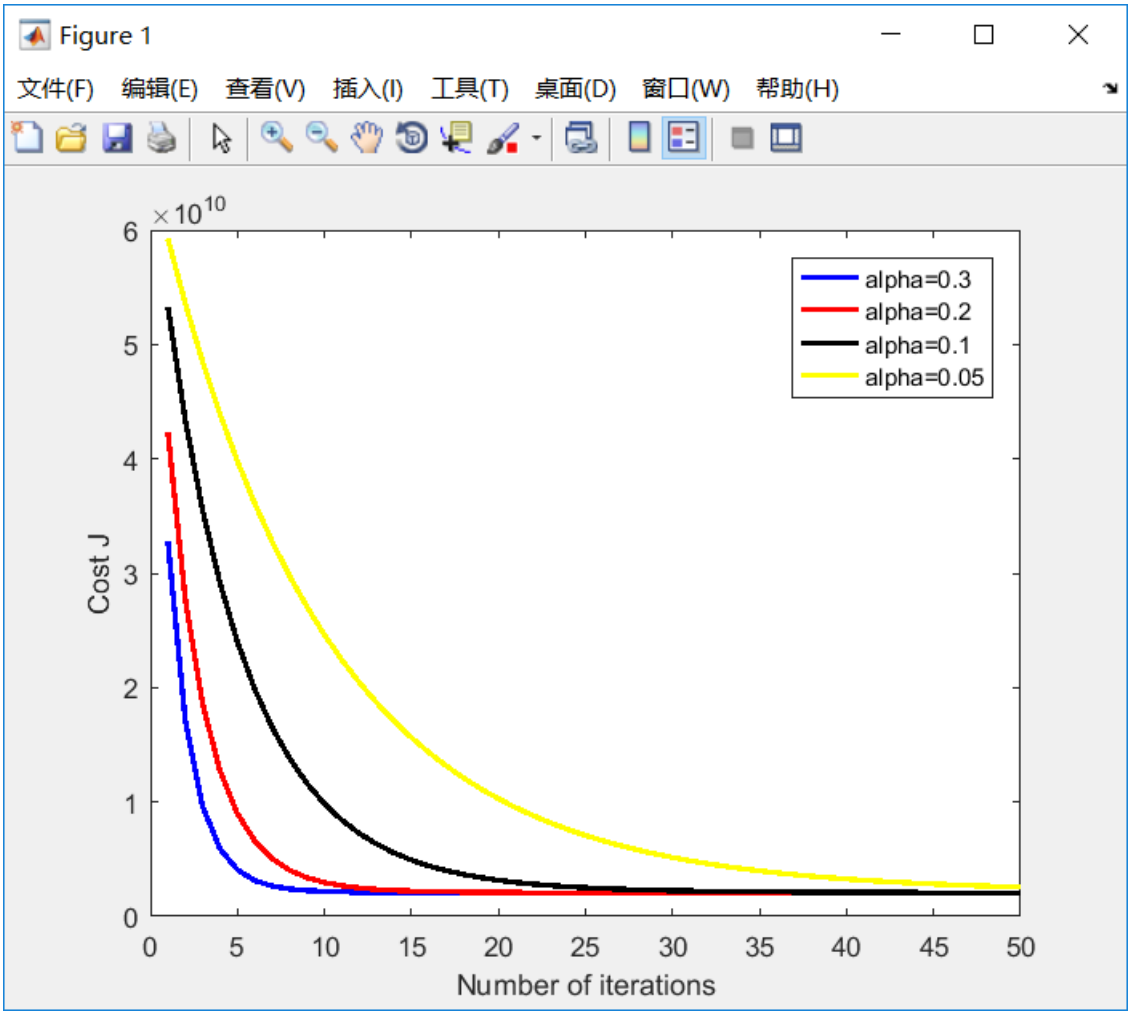
```
>> plot(1:numel(J_history), J_history, '-k', 'LineWidth', 2);
>> hold on;
>> alpha=0.05;
>> theta=zeros(3,1);
>> theta=zeros(3,1);
>> [theta, J_history] = gradientDescentMulti(X, y, theta, alpha,
num_iters);
>> plot(1:numel(J_history), J_history, '-y', 'LineWidth', 2);
>> legend('alpha=0.3', 'alpha=0.2','alpha=0.1','alpha=0.05');
>> hold off;
```

2.4 展示最终的 theta 值和不同学习率对梯度下降的影响：

```
>> fprintf(' %f \n', theta);
 340407.801043
 109981.937075
 -6000.362340
```

2.41 进行预测

```
>> pre1 =  [1, (1650-mu)/sigma, (3-mu)/sigma] *theta;
>> fprintf(' %f \n', pre1);
 307177.386185
```



2.5 使用 Normal Equations 方法对比

```
>> y=load('ex2y.dat');
>> x=load('ex2x.dat');
>> m=length(y);
>> x = [ones(m, 1) x];
>> theta = normalEqn(x,y);
```

2.51 进行预测：

```
>> pre1=[1,1650,3]*theta;
>> fprintf(' %f \n', pre1);
 293081.464335
```

%结果与梯度下降的相差不多。

结论分析与体会：

经过多元线性回归的基本实践，对线性回归的算法有了更深刻了解，对于学习率和迭代次数等也有了深入的理解。

附录：程序源代码

重要算法实现如下：

1. computeCostMulti 函数：

```
function J = computeCostMulti(x,y,theta)
%UNTITLED2 此处显示有关此函数的摘要
%   此处显示详细说明
m=length(y);
J = 0;
% 初始化
J = sum((x*theta - y).^2) / (2 * m);
% 计算损失
end
```

2. featureNormalize 函数（归一化）：

```
function [X_norm,mu,sigma] = featureNormalize(X)
%UNTITLED 此处显示有关此函数的摘要
%   此处显示详细说明
X_norm = X;
%(47,2)
mu = zeros(1, size(X, 2));
%size(X, 2)取 X 的第二维，mu 大小为（1,2）
sigma = zeros(1, size(X, 2));
%(1,2)
m = size(X,1);
```

```
%47
mu = mean(X);
%计算均值
for i = 1 : m,
    X_norm(i, :) = X(i , :) - mu;
end
sigma = std(X);
for i = 1 : m,
    X_norm(i, :) = X_norm(i, :) ./ sigma;
end
end
```

3. gradientDescentMulti 函数:（多元梯度下降）

```
function [theta, J_history] = gradientDescentMulti(X, y, theta, alpha,
num_iters)
m = length(y); % number of training examples
%m=47
J_history = zeros(num_iters, 1);
n = size(X , 2);
%n=3
for iter = 1:num_iters
    H = X * theta;
    %(47,3)*(3*1) = (47*1)
    T = zeros(n , 1);
    %3*1
    for i = 1 : m,
        T = T + (H(i) - y(i)) * X(i,:)';
    end
    theta = theta - (alpha * T) / m;
    % Save the cost J in every iteration
    J_history(iter) = computeCostMulti(X, y, theta);
end
end
```