|  |
| --- |
| **Github账号：www.github.com/zmA0** |
| **实验题目:梯度下降算法及代价函数的应用** |
| **实验摘要：**   1. 实验目的：   利用matlab或者octave，掌握对机器学习中的梯度下降算法和代价函数的代码的编写，以及特征的归一化等知识点的编写。   1. 实验要求：   利用matlab或者octave的数学编程语言进行书写，按照代码文件中的要求编写并提交文件，使得程序运行通过并且得到足够的分数即为通过。   1. 代码文件要求：   至少需要编写的文件为  warmupExercise.m,  gradientDesent.m,  computeCost.m，  这三个文件编写通过实验分数即可达标，还有几个同样也可以编写的文件，不过没有分数，它们为：  normalEqn.m,  featureNormalize.m,  gradientDescentMulti.m,  computeCostMulti.m |
| **题目描述**   1. warmupExercise.m:   这道题的instructions部分说明需要返回一个5x5的单位矩阵   1. computeCost.m   这道题开头定义了一个代价函数即J，参数有X，y，theta。同时还声明了样本长度值m和将J初始化。根据instruction部分，这道题需要我们编写J函数的主体部分，即代价函数中的差的平方和部分。   1. gradientDescent.m   首先这道题开始定义了一个返回值有两个的函数，参数有X,y，theta，alpha和num\_iters即迭代步数。之后同样有样本长度值m，和J\_history函数（即上一题的J函数）。而且这道题还加了一个for循环，循环变量是迭代步数，明显，这道题需要我们编写迭代算法的主体部分，即theta值不断循环自减的部分。  4.normalEqn.m  这道题也是定义出来一个求theta的函数，参数是X，y。通过文件名以及内部注释我们可以知道这个文件需要用到线性回归中求theta的另一个方法，即正规方程法。利用X，y求出合适的theta。   1. featureNormalize.m   关于特征归一化的编写，这道题定义了一个函数参数为X矩阵，同时还定义了一个X\_norm矩阵和mu，sigma两个1x2维的行向量。之后阅读instruction部分可以知道，mu是用来接收均值的，sigma是用来接收标准差的，同时还可以知道提示步骤就是以X的列为单位，先用mean函数计算每个特征x对应的均值，再用std函数计算对应的标准差，然后用X\_norm同样以每个特征x即每列为单位进行新标准值得接收。   1. computeCostMulti.m   题目描述与之前一题一样，不过这题意图需要进行多变量的代价函数的编写，即尽量使用向量化编程思想才能较为方便的完成。（由于之前单变量的代价函数我们已经使用了矩阵的操作，这是适用于本题的，所以代码一样。）   1. gradientDescent.m   题目描述和上面与之对应的单变量题目一样，同理还是需要用向量化的编程思想来完成theta的梯度下降算法。（同样，由于之前已经应用了矩阵和向量的编写规则，所以这里重复之前的代码同样适用） |
| **实验过程**   1. warmupExercise.m   根据上面的题目描述可以知道，这道题仅需要返回一个固定维度的单位矩阵，我这里用的是matlab，因此可以直接利用内置的函数eye直接生成，用法就是A=eye(5)，这样一来在代码编写区域直接写上：  A=eye(5)就可以满足题意。   1. computeCost.m   根据之前所学的课程，可以知道代价函数J就是用假设函数H和样本结果y作差的平方和。在这道题中我们的X，y，theta都是向量或者矩阵。假设若有n个特征，则X为mx(n+1)维矩阵，y为mx1维列向量，theta为(n+1)x1维列向量。所以根据H的线性代数写法，我们可以写出H的表达式H=X \* theta;之后我们就是要写出差的平方和项，其中需要注意H-y是向量，不能直接进行求幂运算，因此需要进行一个点运算，即(H-y).^2,这样一来，就是代表对矩阵中的元素各自进行平方运算；最后我们求和可以直接利用matlab的内置函数sum，直接求和。综上，这道题的代码部分如下所示：  h = X \* theta;  res=sum((h-y).^2);  J= 1/(2\*m) \* res;   1. gradientDescent.m   同理，这道题需要用向量化编程的思想。根据之前所学的课程知道theta需要进行迭代自减运算，减数是一个常数alpha乘以之前J的导函数。以线性代数的角度可以知道，本题J导函数部分的附加乘项x可以利用矩阵乘法来进行书写的化简，因为矩阵乘法中会自带一个相加总和的运算；当然，我们也可以利用sum函数和点运算对每一项进行相加。考虑到（H-y）是一个mx1维列向量，theta是（n+1）x1维列向量，X是mx（n+1）维矩阵，再加之我们的附加乘项x代表将一个特征中的各个样本的x值与（H-y）相乘最后累加，因此我们可以先让X转置，即变成（n+1）xm维之后与（H-y）相乘，即可满足题意。  （其实也可以直接利用sum函数进行运算，具体思路是再编写一个for  训话） |
| 循环，对theta每一项进行运算，此时减数的编写需要用到sum函数和点运算即minus=sum((H-y).\*X(:,1));即为theta（1）的减数部分。）  代码如下：  x = X';  temp = x \* (X \* theta - y);  theta = theta - (alpha/m)\* temp;  4.normalEqn.m  之前的学习中对于正规方程法我们并没有学习它的原理，而且这里因为只涉及求值运算，我们可以直接按照公式的写法书写即可。  代码入下：  theta=(X' \* X)^(-1) \* X' \* y;   1. featureNormalize.m   通过上面对题目的大致分析，可以知道这道题主要是三个部分，一个用来操作的for循环，一个mean函数，一个std函数，由于mu和sigma之前已经定义好为行向量，因此可以直接用行向量的下标对应X矩阵的列标进行for循环的遍历，通过help命令可以了解到上述两个主要函数的作用，同时还要注意X一列代表一种特征，就对应有一个平均值，一个标准差。所以代码如下：  for i =1:size(X,2)  mu(i)=mean(X(:,i));  sigma(i)=std(X(:,i));  X\_norm(:,i)=(X(:,i)-mu(i))/sigma(i);  end   1. computeCostMulti.m   根据之前的描述，上面单变量的代码这里多变量仍然适用  H=X\*theta;  J=1/(2\*m) \* sum((H-y).^2);   1. gradientDescent.m   重复之前代码即可  temp=X' \* (X\*theta-y);  theta=theta-alpha/m \* temp; |
| **实验总结**  第一次编程实验总体来说还算比较简单，都是利用现有的公式，按照注释要求去完成。代码量也不大，但是对机器学习中代价函数和梯度下降算法有了一个直观的认识，在编写过程中确实对向量化编程的思想比较不适应，这种思想确实能够很好简化代码的书写，不过需要对各个矩阵，向量的维数都能够十分熟悉，而且要对matlab中矩阵的各种运算都很好的利用。在出现困难错误的时候，我往往会在matlab命令行中单行书写，这样一来，及时报错才能发现是哪里出了问题，也会对整体的一个书写思路和方法有更多的认识。 |
| **参考文献**  **<https://wenku.baidu.com/view/f328b62b69dc5022abea0068.html>**  （关于这个课程一个笔记，图文并茂，可以当做一个汇总来看）  <http://cn.mathworks.com/help/matlab/index.html?s_tid=gn_loc_drop>  （MATLAB帮助手册） |