编程练习 8：

异常检测和推荐系统

机器学习

# 介绍

在本练习中，您将实现异常情况检测算法，并将其应用于检测网络上出现故障的服务器。在第二部分中，您将使用协同过滤来构建电影的推荐器系统。在开始编程练习之前 ，我们强烈建议您观看视频讲座并完成相关主题的评论问题。

要开始练习，您需要下载入门代码并将其内容解压缩到您希望完成练习的目录中。如果需要，请在 开始本练习之前使用 Octave/MATLAB 中的 cd 命令更改为此目录。

您还可以在课程网站的“环境设置说明”中找到有关安装Octave / MATLAB的说明。

## 本练习中包含的 Fi les

ex8.m - 练习第一部分的 Octave/MATLAB 脚本 ex8 cofi.m - 练习第二部分的 Octave/MATLAB 脚本 ex8data1.mat - 第一个示例异常检测数据集 ex8data2.mat - 异常检测的第二个示例数据集 ex8 movies.mat - 电影评论数据集 ex8 movieParams.mat - 为调试提供的参数multivariateGaussian.m - 计算高斯分布的概率密度函数 visualizeFit.m - 高斯分布和数据集的二维图 CheckCostFunction.m - 用于协同过滤计算的梯度检查NumericalGradient.m - 数值计算梯度 fmincg.m - 函数最小化例程（类似于 fminunc） loadMovieList.m - 将电影列表加载到 cell-array 电影 ID 中.txt - 电影规范化列表Ratings.m - 协作过滤的平均规范化 submit.m - 将您的解决方案发送到我们服务器的提交脚本 [*？*] estimateGaussian.m - 使用对角协方差矩阵估计高斯分布的参数

[*?*] 选择阈值 .m - 查找异常检测的阈值 [*？*] cofiCostFunc.m - 实现协同过滤的成本函数

*?* 表示您需要完成的文件

在整个练习的第一部分（异常检测）中，您将在 8.m 前使用脚本。 对于协同过滤的第二部分，您将使用 ex8 cofi.m。这些脚本为问题设置数据集，并调用您将要编写的函数。您只需按照此赋值中的说明修改其他文件中的函数。

## 从何处获取帮助

本课程中的练习使用Octave[[1]](#footnote-1) 或MATLAB，这是一种非常适合数值计算的高级编程语言。如果您没有安装Octave或MATLAB，请参阅课程网站“环境设置说明”中的安装说明。

在 Octave/MATLAB 命令行中，键入 help 后跟函数名称将显示内置函数的文档。例如，帮助绘图将显示绘图的帮助信息。有关倍频程函数的更多文档，请参阅[倍频程文档页面。](http://www.gnu.org/software/octave/doc/interpreter/) MATLAB 文档可以在 [MATLAB 文档页面上](http://www.mathworks.com/help/matlab/?refresh=true)找到[。](http://www.mathworks.com/help/matlab/?refresh=true)

我们还强烈建议使用在线 **讨论** 与其他学生讨论练习。但是，不要查看任何由others编写的源代码或与他人共享您的源代码。

# 异常检测

在本练习中，您将实现异常情况检测算法来检测服务器计算机中的异常行为。这些功能可测量每个服务器的吞吐量 （mb/s） 和响应延迟 （ms）。当您的服务器运行时，您收集了 *m* = 307 个示例，了解它们的行为方式，因此具有未标记的数据集 {*x*（1）*,...，x*（*m*）}.您怀疑这些示例中的绝大多数是服务器正常运行的“正常”（非异常）示例 ，但也可能有一些服务器在此数据集中异常运行的示例。

您将使用高斯模型来检测数据集中的异常示例。您将首先从 2D 数据集开始，该数据集将允许您可视化算法正在执行的操作。在该数据集上，您将拟合高斯分布，然后找到概率非常低的值，因此可以将其视为异常。之后，您将异常情况检测算法应用于具有多个维度的较大数据 et。您将在这部分练习中使用 ex8.m。

ex8.m 的第一部分将可视化数据集，如图 1 所示。

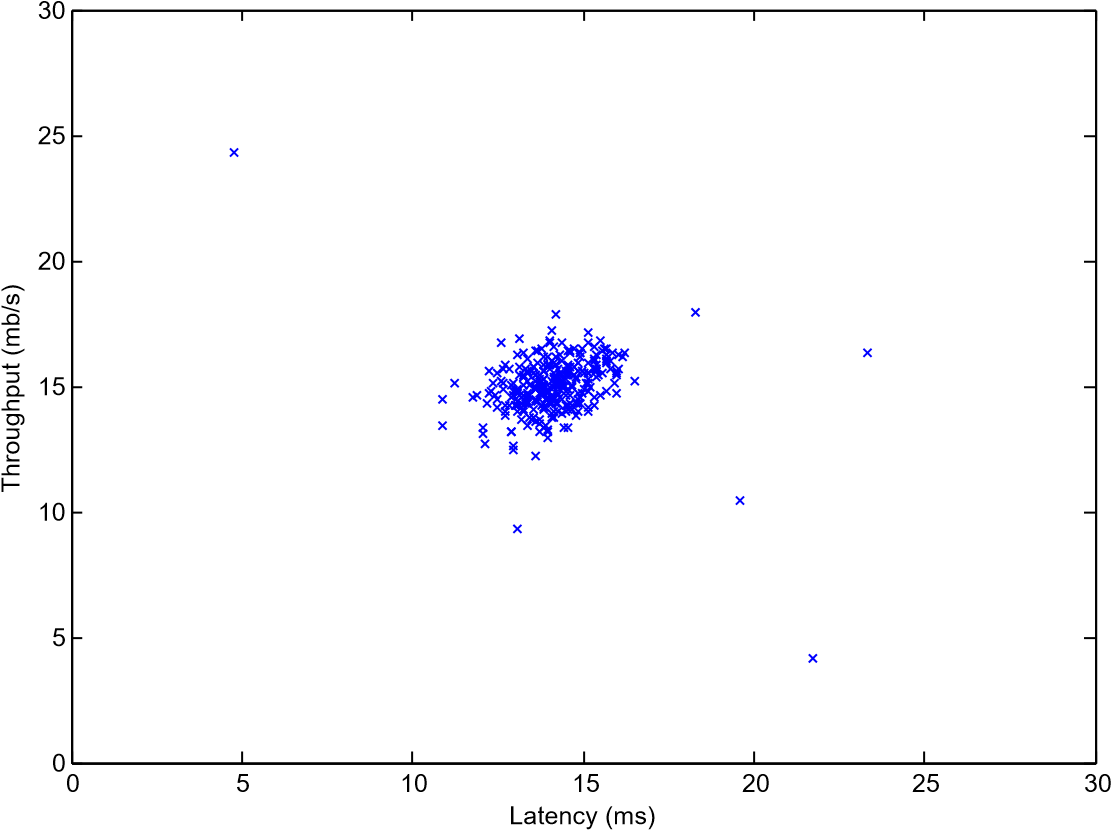
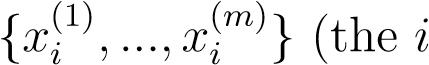


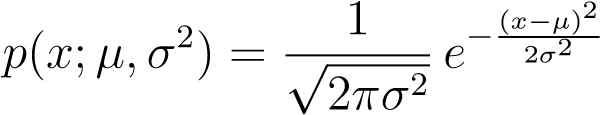
图 1：第一个数据集。

## 正态分布

若要执行异常检测，首先需要将模型拟合到数据的分布。

给定一个训练集 {*x*（1）*,...，x*（*m*）} （其中 *x*（*i*） ∈ R*n*），您希望估计每个特征 *xi* 的高斯分布。对于每个特征 *i* = 1*...n*，您需要找到参数*μi* 和 *σi*2，这些参数适合每个示例的第 *i* 维 -th 维中的数据）。

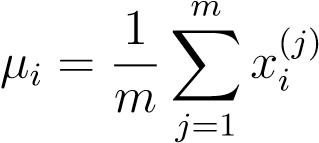
高斯分布由下式给出

*,*

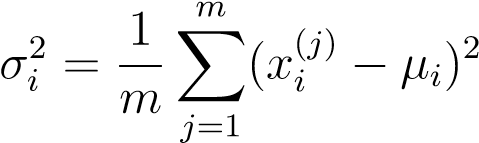
其中 *μ* 是均值， *σ*2 控制方差。

## 估计高斯的参数

您可以使用以下公式估计第 *i* 个特征的参数 （*μi*，*σ* *i* 2）。要估计平均值，您将使用：

*，* （1）

对于方差，您将使用：

*.* （二）

您的任务是完成估计Gaussian.m中的代码。此函数将数据矩阵 X 作为输入 ，并应输出一个*保存所有 n* 个特征均值的 *n* 维向量 mu 和另一个 保存所有特征方差的 n 维向量 sigma2。你可以在每个特征和每个训练示例上使用for-loop来实现这一点（尽管矢量化实现可能更有效;如果你愿意，可以随意使用矢量化实现）。请注意，在Octave/MATLAB中，var函数（默认情况下）在计算*σi*2时使用，而不是。

完成 estimateGaussian.m 中的代码后，ex8.m 的下一部分将可视化拟合高斯分布的等值线。您应得到一个类似于图 2 的绘图。从您的绘图中，您可以看到大多数示例位于概率最高的区域中，而异常示例位于概率较低的区域 s 中。

*您现在应该提交您的解决方案。*

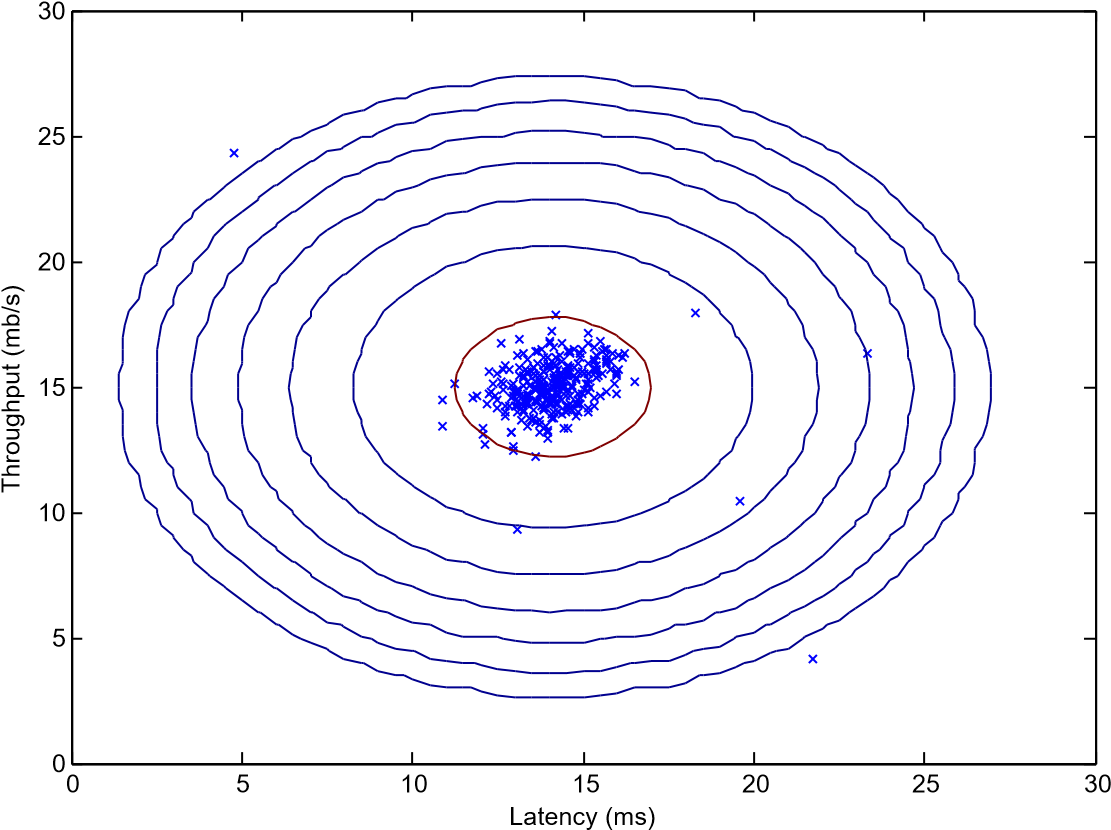
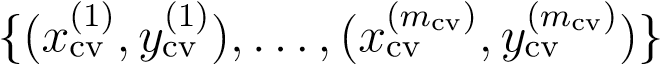
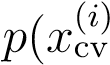
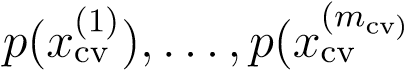


图 2：分布的高斯分布等值线与数据集拟合。

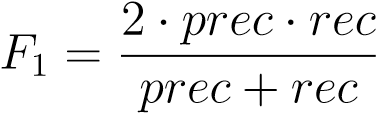
## 选择阈值， *ε*

现在您已经估计了高斯参数，您可以研究给定此分布的概率非常高的示例，并且哪些示例的概率非常低。低概率示例更有可能成为我们数据集中的异常。确定哪些示例是异常的一种方法是根据交叉验证集选择阈值。在本部分练习中，您将实现一种算法，以使用交叉验证集上的 *F*1  分数*ε*选择阈值。

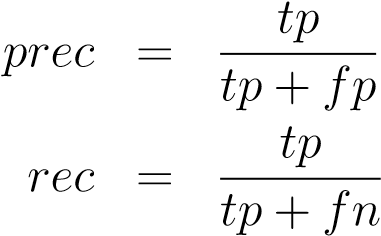
现在，您应该在 selectThreshold.m 中完成代码。为此，我们将使用交叉验证集 ，其中标签 *y* = 1 对应于异常示例，*y* = 0 对应于正常示例。对于每个交叉验证示例，我们将进行计算）。所有这些概率的向量）被传递到 向量 pval 中的 selectThreshold.m。相应的标签 *y*cv（1）*,...，y*cv（*m*cv） 被传递到向量 yval 中的相同函数。

函数 selectThreshold.m 应返回两个值;第一个是所选阈值*ε*。如果示例 *x*  的*概率 p*（*x*） *< ε*较低，则将其视为异常。该函数还应返回*F*1分数，which告诉您在给定某个阈值的情况下在查找基本真值异常方面做得如何。对于许多不同的*ε*值，您将通过计算当前阈值正确分类且不均匀分类的示例数来计算生成的 *F*1  分数。

*F*1 分数使用精度 （*prec*） 和召回率 （*rec） 计算*：

*，* （3）

您可以通过以下方式计算精度和召回率：

(4)

*，* （5）

哪里

* *tp* 是真阳性的数量：基本事实标签说这是异常，我们的算法正确地将其归类为异常。
* *fp* 是误报的数量：基本事实标签说它不是异常，但是我们的算法错误地将其归类为异常。
* *fn* 是假阴性的数量：基本事实标签说这是一个异常，但是我们的算法错误地将其分类为不是异常的。

在提供的代码 selectThreshold.m 中，已经有一个循环，它将尝试许多不同的*ε*值，并根据 *F*1 分数选择最佳*ε*。

现在，您应该在 selectThreshold.m 中完成代码。您可以使用所有交叉验证示例的 for 循环来实现 F1 分数的 计算（以计算值 *tp*、*fp*、*fn*）。您应该看到 epsilon 的值约为 8.99e-05。

|  |
| --- |
| **实现说明：**为了计算 *tp*、*fp* 和 *fn*，您可以使用矢量化实现，而不是遍历所有示例。这可以通过Octave / MATLAB在向量和单个数字之间的相等性测试来实现。如果在 *{*0*，*1}n 维二进制向量 *v* ∈有多个二进制值，则可以使用以下方法找出此向量中有多少个值为 0 。您还可以将逻辑和运算符应用于此类二元向量。例如，假设 cvPredictions 是交叉验证集数大小的二进制向量，其中*第 i* 个元素为 1（如果算法考虑异常），否则为 0。例如，您可以使用以下公式计算误报数：fp = sum（（cvPredictions == 1） & （yval == 0））。 |

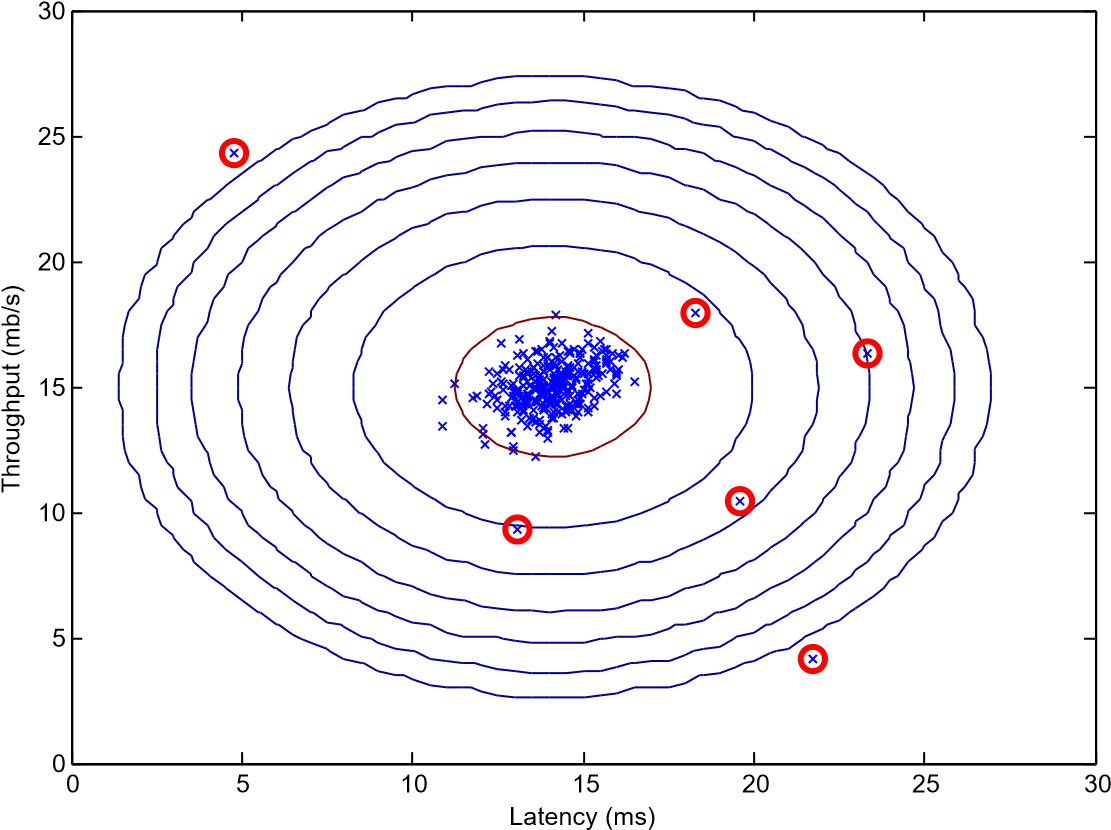


图 3：分类异常。

完成 selectThreshold.m 中的代码后，ex8.m 中的下一步将运行异常情况检测代码并圈出图中的异常（图 3）。

*您现在应该提交您的解决方案。*

## 高维 dataset

脚本 ex8.m 的最后一部分将运行你在更真实、更困难的数据集上实现的异常情况检测算法。在此数据集中，每个示例由 11 个功能描述，捕获计算服务器的更多属性。

该脚本将使用您的代码来估计高斯参数（*μi* 和 *σi*2），评估从中估计高斯参数的训练数据 X 的概率，并对交叉验证集 Xval 执行此操作。最后，它将使用 selectThreshold 来查找ε的最佳阈值。

您应看到 epsilon 的值约为 1.38e-18，并发现 117 个异常。

# 推荐者 S系统

在本部分练习中，您将实现协同过滤学习算法，并将其应用于电影评级数据集。[[2]](#footnote-2) 此数据集由 1 到 5 的等级评级组成。数据集有 *n u*= 943 个用户， *nm* = 1682 部电影。在本部分练习中，您将使用脚本 ex8 cofi.m。

在本练习的下一部分中，您将实现计算协同拟合目标函数和梯度的函数 cofiCostFunc.m。在实现成本函数和梯度后，您将使用 fmincg.m 来学习协同过滤的参数。

## 电影评级数据集

脚本 ex8 cofi.m 的第一部分将加载数据集 ex8 movies.mat，在 Octave/MATLAB 环境中提供变量 Y 和 R i。

矩阵 *Y*（ 数字电影×数字用户矩阵）存储评级 *y*（*i，j*）

（从 1 到 5）。矩阵 *R* 是一个二值指标矩阵，其中 *R*（*i，j*） = 1（如果用户 *j* 给电影 i 一个评级，R（*i，j*） = 0 否则。 协同过滤的目的是预测用户尚未分级的电影的电影分级，即 *R*（*i，j*） = 0 的条目。这将使我们能够向用户推荐具有最高预测评分的电影。

为了帮助您处理矩阵 Y，脚本 ex8 cofi.m 将计算第一部电影（玩具总动员）的平均电影评级，并将平均评级输出到屏幕上。

在本部分练习中，您还将使用矩阵 X 和 θ：

|  |  |
| --- | --- |
|  （1））*T* —   — （*x*   — （*x*（2））*T* —  X =  ... *，*    — （*x*（*nm*））*T* — |  （1））*T* —   — （*i*   — （*θ*（2））*T* —  Theta =  ... *。*    — （*θ*（*nu*））*T* — |

对于第 *i* 个用户，X 的第 i 行对应于第 i 个电影的特征向量 *x*（*i*），而 Theta 的第 *j* 行对应于一个参数向量 *θ*（j）。对于第 *1 个*用户。*x*（*i*） 和 *θ*（*j*） 都是 *n* 维向量。在本练习中，您将使用 *n* = 100，因此，*x*（*i*） ∈ R100，*θ*（*j*） ∈ R100。相应地，X是n *m*×100矩阵，Theta是n *u*×100矩阵。

## 协同过滤学习算法

现在，您将开始实现协同过滤学习算法。您将从实现成本函数（不正则化）开始。

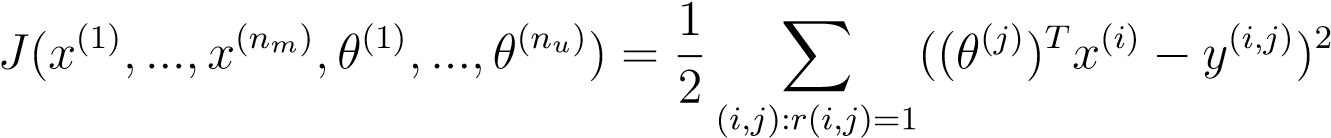
电影推荐设置中的协同过滤算法考虑了一组n-d的参数向量*x*（1）*,...* *，x*（*nm*） 和 *θ*（1）*,...,θ*（*nu*），其中模型预测 用户 *j*  的电影 *i* 的评级为 *y*（*i，j*） = （*θ*（*j*））*T x*（*i*）.给定一个数据集，其中包含一些用户对某些电影产生的一组评级，您希望学习参数向量

*x*（1）*,...，x*（*nm*）*，θ*（1）*,...,θ*（*nu*） 产生最佳拟合（最小化平方误差）。

您将在 cofiCostFunc.m 中完成代码，以计算协同筛选的成本函数和梯度。请注意，函数的参数（即，您尝试学习的值）是 X 和 Theta。为了使用现成的最小化器（如 fmincg），cost 函数 ha已被设置为将参数展开为单个向量参数。您之前在神经网络编程练习中使用相同的向量展开方法。

### 协同过滤成本函数

协同滤波成本函数（without 正则化）由下式给出

*.*

您现在应该修改 cofiCostFunc.m 以在变量 J 中返回此成本。请注意， 仅当 *R*（*i，j*） = 1 时，才应累积用户 *j* 和 movie *i* 的成本。

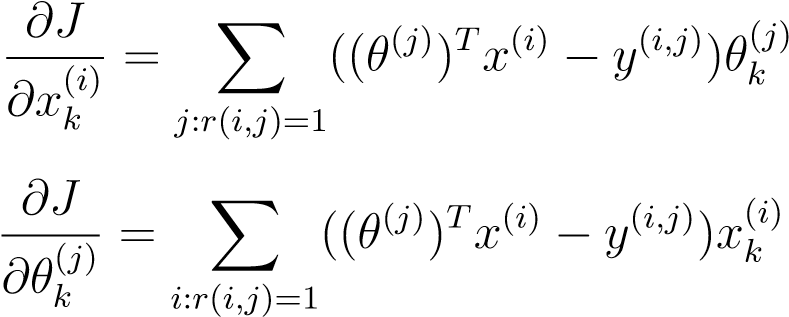
完成函数后，脚本 ex8 cofi.m 将运行您的成本函数。您应该期望看到 22 的输出*。*22.

*您现在应该提交您的解决方案。*

**实现说明：**我们强烈建议您使用矢量化实现来计算*J*，因为它稍后会被优化包fmincg多次调用。像往常一样，最简单的方法是首先编写一个非矢量化的实现（以确保你有正确的答案），然后将其修改为矢量化实现（检查矢量化步骤是否不会改变算法的输出）。要提出一个矢量化的实现，以下提示可能会有所帮助：您可以使用R矩阵to将所选条目设置为0。例如，R .\* M 将在 M 和 R 之间执行按元素的乘法;由于 R 只有值为 0 或 1 的元素，因此仅当 R 中的相应值为 0 时，这才具有将 M 的元素设置为 0 的效果。因此，sum（sum（R.\*M）） 是 M 中所有元素的总和 ，其中 R 中的相应元素 等于 1。

### Col劳动过滤梯度

现在，您应该实现渐变（不带正则化）。具体来说，您应该在 cofiCostFunc.m 中完成代码以返回变量 X grad 和 Theta grad。请注意，X grad 应该是与 X 具有相同 siz e 的矩阵 ，同样，Theta grad 是与 Theta 大小相同的矩阵。成本函数的梯度 由下式给出：

*.*

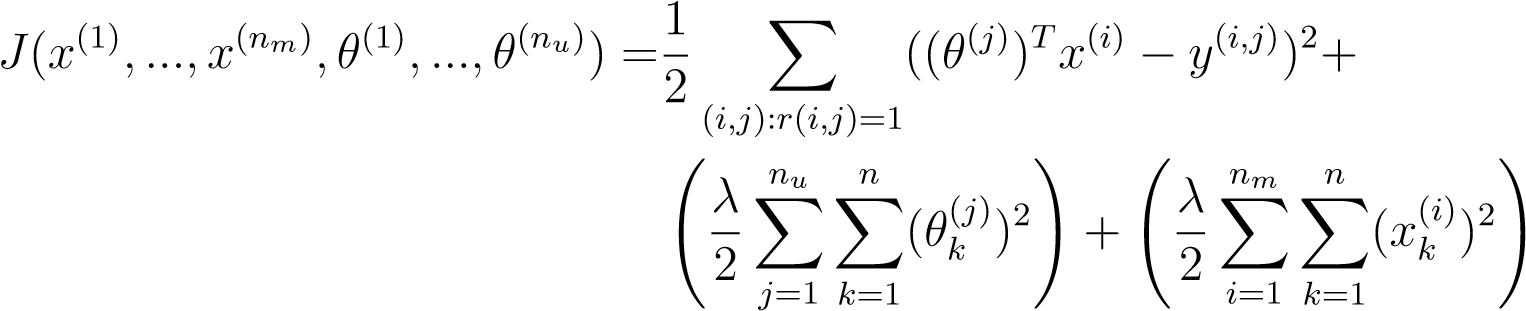
请注意，该函数通过将两组变量展开到单个向量中来返回这两组变量的梯度。完成代码以计算梯度后，脚本 ex8 cofi.m 将运行梯度检查 （checkCostFunction） 以数字方式检查梯度的实现。[[3]](#footnote-3) 如果你的实现是正确的，你应该发现分析和数值 gradients 非常匹配。

*您现在应该提交您的解决方案。*

|  |
| --- |
| **实现说明：** 您可以在不使用矢量化实现的情况下获得此分配的全部功劳，但是您的代码将运行得更慢（几个小时），因此我们建议您尝试矢量化您的实现。  首先，您可以使用电影上的for-loop（用于 计算） 和for-loop over users（用于计算）来实现梯度。首次实现梯度时，可以从 未矢量化 版本开始，通过实现另一个计算求和中每个元素的内部 for-l oop。以这种方式完成梯度计算后，您应该尝试对实现进行矢量化（对内部 for 循环进行矢量化），以便只剩下两个 for 循环（一个用于循环 电影以 计算 每部电影，另一个用于循环用户以  计算每个用户）。 |

|  |
| --- |
| **实现提示：**要执行矢量化，您可能会发现这很有帮助：您应该提出一种方法来计算与特征向量x（i）相关的所有导数（即与特征向量*x*（*i*）关联的导数项）。让我们将第 *i* 个电影的特征向量的导数定义为：  （X度  要对上述表达式进行矢量化，可以先索引到 θ 和 Y 中，以仅选择感兴趣的元素（即 *r*（*i，j*） = 1 的元素）。直观地说，当您考虑第*i*部电影的功能时，您只需要关注为电影评分的用户，这允许您从Theta和Y中删除所有其他用户。  具体来说，你可以将idx = find（R（i， :)==1）设置为所有对电影i进行评级的用户的列表。这将允许您创建临时矩阵 Thetatemp = Theta（idx*，*:)和 Ytemp = Y（i*，*idx），将其索引到 Theta 和 Y 中，仅为您提供对第 *i* 部电影进行评级的用户集。这个wi将允许您将衍生品编写为：  Xgrad（i*，*:) = （X（i*，*:) ∗ ΘTtemp − Ytemp） ∗ Thetatemp*。*  （注意：上面的矢量化计算返回行向量。  在对导数相对于 *x*（*i*） 的计算进行矢量化后，还应使用类似的方法对 θ（j） 的导数进行矢量化 。 |

### 正则化成本函数

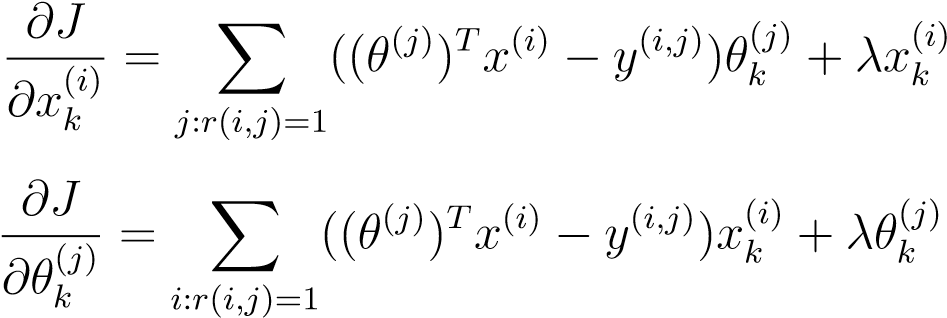
正则化协同过滤的成本函数由 给出  *。*

现在，您应该将正则化添加到成本函数 *J* 的原始计算中。完成后，脚本 ex8 cofi.m 将运行正则化成本函数，您应该期望看到大约 31.34 的成本。

*您现在应该提交您的解决方案。*

### 正则化梯度

现在，您已经实现了正则化成本函数，您应该继续为梯度实现正则化。您应该在 cofiCostFunc.m 中添加 r 实现 ，以通过添加正则化项的贡献来返回正则化梯度。请注意，正则化成本函数的梯度由下式给出：

*.*

这意味着你只需要将 *λx*（*i*） 添加到 X grad（i，:) 前面描述的变量，并将 *λθ*（*j*） 添加到 Theta grad（j，:) 前面描述的变量。

完成计算梯度的代码后，脚本 ex8 cofi.m 将运行另一个梯度检查 （checkCostFunction） 以数字方式检查梯度的实现。

*您现在应该提交您的解决方案。*

## 学习影片推荐

在完成协作过滤成本函数和梯度后，您现在可以开始训练自己是一个lgorithm 为自己制作电影推荐。在 ex8 cofi.m 脚本的下一部分，您可以输入自己的电影首选项，以便以后在算法运行时，您可以获得自己的电影推荐！我们已经根据自己的喜好填写了一些值，但您应该根据自己的喜好进行更改。数据集中所有电影的列表及其编号可以在文件电影 idx.txt 中找到。

### 建议

为您提供的最佳建议：

电影《泰坦尼克号》（1997）的预测评分为9.0

预测评级 8.9 电影星球大战 （1977）

预测评级8.8的电影肖申克救赎，（1994）

预测电影《尽善尽美》（1997）的评分8.5

Predicti ng rating 8.5 for Movie Good Will Hunting （1997）

预测评级8.5的电影普通嫌疑人，（1995）

电影《辛德勒的名单》（1993）的预测评分8.5

电影《夺宝奇兵》（Raiders of the Lost Ark，1981）的预测评分为8.4

预测评级8.4为mov，即帝国反击战，（1980）

预测评级8.4的电影勇敢的心（1995）

提供的原始评级：

4 分 （1995）

3 for Twelve Monkeys （1995）

《普通嫌疑人》第5名（1995）

疫情等级为 4 （1995）

1994 年 Sh awshank Redemption（ The （1994） 評分 5 名

《When You Are Sleeping》評級 3 （1995）

5 级 Forrest Gump （1994）

《沉默的羔羊》（Silence of the Lambs） 評級 2 （1991）

额定 4 为 外星人 （1979）

Die Hard 2 （1990） 第 5 名

5 级球体 （1998）

图 4：电影推荐

将其他评级添加到数据集后，脚本将继续训练协作筛选模型。这将学习参数 X 和 Theta。要预测 用户*j*的电影*i*的评级，您需要计算（*θ*（*j*））*T x*（*i*）。。脚本的下一部分计算所有电影和用户的评级，并根据脚本前面输入的评级显示它推荐的电影（图 4）。请注意，由于不同的随机初始化，您可能会获得一组不同的预测ns。

# 提交和评分

完成作业的各个部分后，请务必使用提交功能系统将您的解决方案提交到我们的服务器。以下是本 exercise 的每个部分的评分方式的细分。

|  |  |  |
| --- | --- | --- |
| **部分** | **提交的文件** | **点** |
| 估计高斯参数 | estimateGuassian.m | 15 积分 |
| 选择阈值 | selectThreshold.m | 15 积分 |
| 协同过滤成本 | cofiCostFunc.m | 20 积分 |
| 协同过滤梯度 | cofiCostFunc.m | 30 积分 |
| 正则化成本 | cofiCostFunc.m | 10 积分 |
| 具有正则化的梯度 | cofiCostFunc.m | 10 积分 |
| 总积分 |  | 100 积分 |

您可以多次提交解决方案，我们将仅考虑最高分。

1. Octave 是 MATLAB 的免费替代品。对于编程练习，您可以自由使用Octave或MATLAB。 [↑](#footnote-ref-1)
2. [电影长度 100k 数据集](http://www.grouplens.org/node/73/) 从GroupLens研究。 [↑](#footnote-ref-2)
3. 这类似于您在神经网络练习中使用的数值校验。 [↑](#footnote-ref-3)