编程练习 2:逻辑回归

机器学习

介绍

在本练习中,您将实现逻辑回归并将其应用于两个不同的数据集。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复习题。

要开始练习,您需要下载起始代码并将其内容解压缩到您希望完成练习的目录。如果需要,在开始本练习之前使用 Octave/MATLAB 中的 cd 命令切换到该目录。

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

本练习中包含的文件

ex2.m - 指导您完成练习的 Octave/MATLAB 脚本 ex2 reg.m - 练习后面部分的 Octave/MATLAB 脚本 ex2data1.txt - 练习前半部分的训练集 ex2data2.txt - 训练集练习的后半部分 submit.m - 将您的解决方案发送到我们的服务器的提交脚本 mapFeature.m - 生成多项式特征的函数 plotDecisionBoundary.m - 绘制分类器决策界限的函数

ary
[?] plotData.m - 绘制 2D 分类数据的函数 [?] sigmoid.m - Sigmoid 函数 [?] costFunction.m - 逻辑回归成本函数 [?] predict.m - 逻辑回归预测函数 [?] costFunctionReg。 m - 正则化逻辑回归成本

?表示您需要完成的文件

在整个练习中,您将使用脚本 ex2.m 和 ex2 reg.m。
这些脚本为问题设置数据集并调用您将编写的函数。您不需要修改其中任何一个。您只需按照本作业中的说明修改其他文件中的函数。

从哪里获得帮助本课程的练习

使用 Octave1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,请参考课程网站“环境设置说明”中的安装说明。

在 Octave/MATLAB 命令行中,键入 help 后跟一个函数名称会显示内置函数的文档。例如,help plot 会调出绘图的帮助信息。 Octave 函数的更多文档可以在Octave 文档页面找到。 MATLAB 文档可以在MATLAB 文档页面中找到。

我们也强烈鼓励使用在线讨论与其他学生讨论练习。但是,请勿查看他人编写的任何源代码或与他人共享您的源代码。

---

1 逻辑回归

在这部分练习中,您将建立一个逻辑回归模型来预测学生是否被大学录取。

假设您是大学系的管理员,并且您想根据每个申请人在两次考试中的成绩来确定他们的录取机会。您拥有以前申请者的历史数据,可用作逻辑回归的训练集。对于每个培训示例,您都有申请人在两次考试中的分数和录取决定。

你的任务是建立一个分类模型,根据这两个考试的分数来估计申请人的录取概率。此大纲和 ex2.m 中的框架代码将指导您完成练习。

1Octave 是 MATLAB 的免费替代品。对于编程练习,您可以自由使用 Octave 或 MATLAB。

1.1 数据可视化

在开始实施任何学习算法之前,如果可能,最好将数据可视化。在 ex2.m 的第一部分,代码将加载数据并通过调用函数 plotData 将其显示在二维图上。

您现在将完成 plotData 中的代码,使其显示类似于图1 的图形,其中轴是两个考试分数,正例和负例以不同的标记显示。

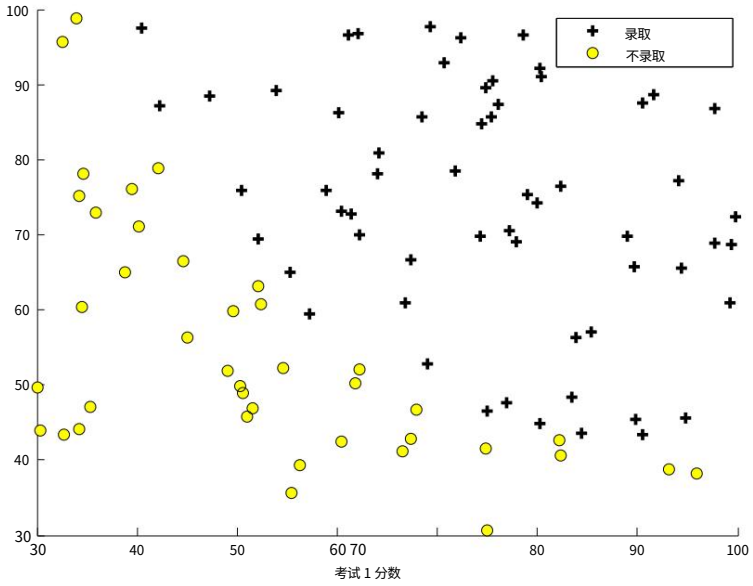

图 1:训练数据的散点图

为了帮助您更熟悉绘图,我们将 plotData.m 留空,因此您可以尝试自己实现它。但是,这是一个可选的（未评分的)练习。我们还在下面提供了我们的实现,因此您可以复制或参考它。如果您选择复制我们的示例,请确保通过查阅 Octave/MATLAB 文档了解其每个命令的作用。

```
% 查找正例和负例的索引pos = find(y==1);否定 = 查找（y == 0）；


% 绘图示例plot(X(pos, 1),
X(pos, 2),    k+    ,    LineWidth    , 2, ...
              标记大小    , 7);
情节（X（否定,1）,X（否定,2）,    ko    ,    MarkerFaceColor    ,    y    ,    ...
              标记大小    , 7);
```

1.2 实施

1.2.1 热身练习:sigmoid函数

在开始实际成本函数之前,请回忆一下逻辑回归假设定义为:

$$h\theta(x) = g(\theta^{T} x),$$

其中函数 g 是 sigmoid 函数。 sigmoid 函数定义为:

$$g(z) = \frac{1}{1 + e^{-z}}.$$

您的第一步是在 sigmoid.m 中实现此函数,以便程序的其余部分可以调用它。完成后,尝试通过在 Octave/MATLAB 命令行调用 sigmoid(x) 来测试一些值。对于较大的 x 正值,sigmoid 应该接近 1,而对于较大的负值,sigmoid 应该接近 0。评估 sigmoid(0) 应该正好给你 0.5。您的代码也应该适用于向量和矩阵。对于矩阵,您的函数应该对每个元素执行 sigmoid 函数。

您可以通过在 Octave/MATLAB 命令行中输入 submit 来提交您的解决方案以进行评分。提交脚本将提示您输入登录电子邮件和提交令牌,并询问您要提交哪些文件。您可以从网页获取作业的提交令牌。

您现在应该提交您的解决方案。

1.2.2 成本函数和梯度

现在您将实现逻辑回归的成本函数和梯度。
完成 costFunction.m 中的代码以返回成本和梯度。

回想一下,逻辑回归中的成本函数是

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} -y^{(i)} \log(h\theta(x^{(i)})) - (1 - y^{(i)}) \log(1 - h\theta(x^{(i)})),$$

并且成本的梯度是一个与 θ 长度相同的向量,其中第 j^{th} 个元素(对于 $j = 0, 1, \ldots, n$)定义如下:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

请注意,虽然这个梯度看起来与线性回归梯度相同,但公式实际上是不同的,因为线性回归和逻辑回归对$h_\theta(x)$ 的定义不同。

完成后,ex2.m 将使用初始值调用您的 costFunction
θ 的参数。您应该看到成本约为 0.693。

您现在应该提交您的解决方案。

1.2.3 使用 fminunc 学习参数

在之前的作业中,您通过实施梯度下降找到了线性回归模型的最佳参数。您编写了一个成本函数并计算了它的梯度,然后相应地采取了梯度下降步骤。

这一次,您将使用一个名为 fminunc 的 Octave/MATLAB 内置函数,而不是采用梯度下降步骤。

Octave/MATLAB 的 fminunc 是一个优化求解器,可以找到 unconstrained[2] 函数的最小值。对于逻辑回归,您希望使用参数 θ 优化成本函数 J(θ)。

具体来说,在给定固定数据集（X 和 y 值)的情况下,您将使用 fminunc 为逻辑回归成本函数找到最佳参数 θ。您将向 fminunc 传递以下输入:

- 我们试图优化的参数的初始值。
- 一个函数,当给定训练集和特定 θ 时,计算数据集相对于 θ 的逻辑回归成本和梯度

  (X, y)

在 ex2.m 中,我们已经编写了使用正确参数调用 fminunc 的代码。

2优化中的约束通常是指对参数的约束,例如,约束 θ 可以取的可能值（例如,$\theta \leqslant 1$）。逻辑回归没有这样的约束,因为 θ 可以取任何实际值。

```
% 为 fminunc 设置选项options =
optimset( GradObj , on , MaxIter ,400);

% 运行 fminunc 以获得最优的 theta
% 这个函数将返回 theta 和成本
[theta, 成本] = fminunc(@(t) ...
        (costFunction(t, X, y)), 初始 theta, 选项);                    –
```

在这个代码片段中,我们首先定义了与 fminunc 一起使用的选项。
具体来说,我们将 GradObj 选项设置为 on,它告诉 fminunc 我们的函数返回成本和梯度。这允许 fminunc 在最小化函数时使用梯度。此外,我们将 MaxIter 选项设置为 400,以便 fminunc 在终止之前最多运行 400 步。

为了指定我们要最小化的实际函数,我们使用“速记”来指定带有 @(t) (costFunction(t, X, y)) 的函数。这将创建一个带有参数 t 的函数,该函数调用您的 costFunction。这允许我们包装 costFunction 以与 fminunc 一起使用。

如果您正确完成了 costFunction,fminunc 将收敛到正确的优化参数并返回成本和 θ 的最终值。请注意,通过使用 fminunc,您不必自己编写任何循环,也不必像在梯度下降中那样设置学习率。这一切都由 fminunc 完成:您只需要提供一个计算成本和梯度的函数。

一旦 fminunc 完成,ex2.m 将使用 θ 的最佳参数调用您的 costFunction 函数。您应该看到成本约为 0.203。

然后,这个最终的 θ 值将用于在训练数据上绘制决策边界,得到类似于图2 的图形。我们还鼓励您查看 plotDecisionBoundary.m 中的代码,了解如何使用θ 值。

1.2.4 评估逻辑回归

学习参数后,您可以使用该模型来预测特定学生是否会被录取。对于考试 1 得分为 45 和考试 2 得分为 85 的学生,您应该期望看到 0.776 的录取概率。

评估我们发现的参数质量的另一种方法是查看学习模型在我们的训练集上的预测效果。在这个

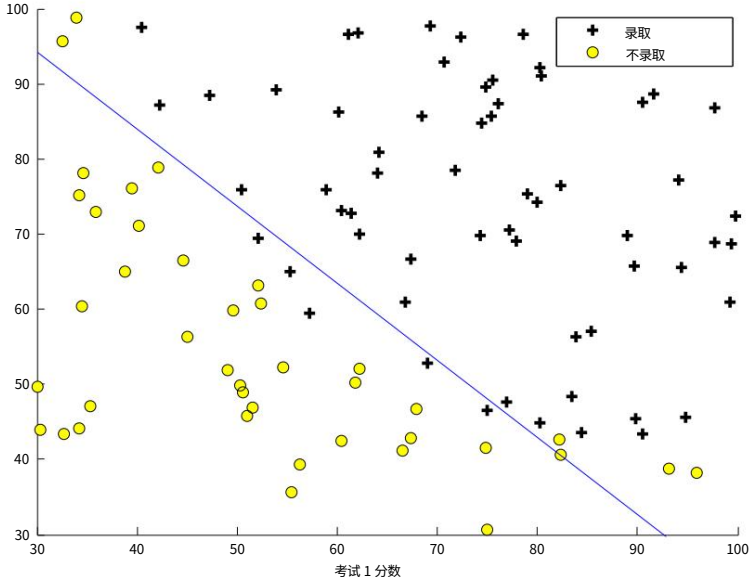

图 2:具有决策边界的训练数据

部分,您的任务是完成 predict.m 中的代码。预测功能
给定数据集和学习参数,将产生“1”或“0”预测
向量θ。

完成 predict.m 中的代码后,ex2.m 脚本将
继续通过计算来报告分类器的训练准确性
它得到正确的例子的百分比。

您现在应该提交您的解决方案。

2 正则化逻辑回归

在这部分练习中,您将实现正则化逻辑回归
预测制造厂的微芯片是否通过质量保证 (QA)。在 QA 期间,每个微芯片都经过各种测试,以确保

它运行正常。

假设你是工厂的产品经理,你有
一些微芯片在两种不同测试中的测试结果。从这两个测试中,
您想确定是否应接受微芯片或
被拒绝。为了帮助您做出决定,您有一个测试结果数据集
在过去的微芯片上,您可以从中构建逻辑回归模型。

您将使用另一个脚本 ex2 reg.m 来完成这部分练习。

2.1 数据可视化

与本练习的前面部分类似,plotData 用于生成如图3 所示的图形,其中轴是两个测试分数,正面（y = 1,接受)和负面（y = 0,拒绝)示例是用不同的标记显示。

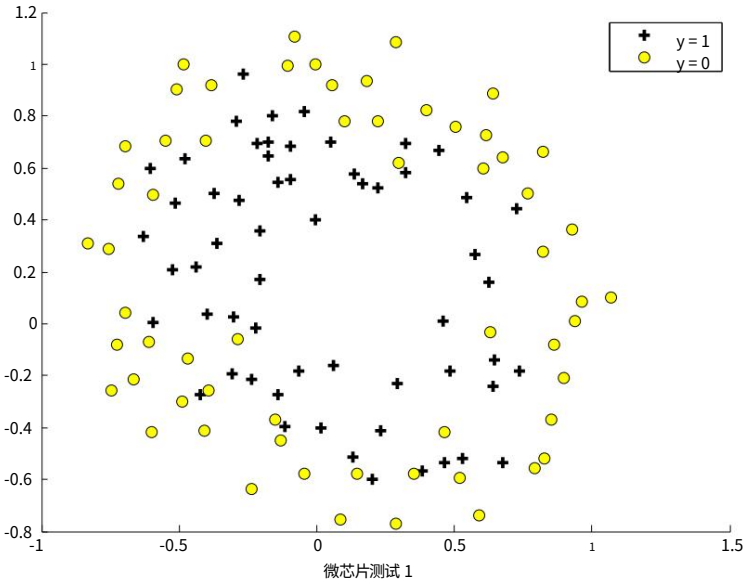

图 3:训练数据图

图3显示我们的数据集不能通过图中的直线分为正样本和负样本。因此,逻辑回归的直接应用在该数据集上表现不佳,因为逻辑回归只能找到线性决策边界。

2.2 特征映射

更好地拟合数据的一种方法是从每个数据点创建更多特征。在提供的函数 mapFeature.m 中，我们将特征映射到x1和x2的所有多项式项,直到六次方。

x1
x2
2 x1
×2 2 3
×1
×1×2 5 6
×2

2.3 成本函数和梯度

— X米 Xn 2
我=1 j=1

th

——— — X米 (hθ(x (i)) - y (i))x j (一世)
我=1

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{米} \sum_{我=1} (h\theta(x(i)) - y(i))x^{(一世)}_{!\ \_+} \frac{\lambda}{米} \theta j \text{对于 } j \geqslant 1$$

完成后,ex2 reg.m 将使用 θ 的初始值（初始化为全零)调用 costFunctionReg 函数。您应该看到成本约为 0.693。

您现在应该提交您的解决方案。

2.3.1 使用 fminunc 学习参数

与前面部分类似,您将使用 fminunc 来学习最优参数 θ。如果您已正确完成正则化逻辑回归 (costFunctionReg.m) 的成本和梯度,您应该能够逐步完成 ex2 reg.m 的下一部分,以使用 fminunc 学习参数 θ。

2.4 绘制决策边界

为了帮助您可视化此分类器学习的模型,我们提供了函数 plotDecisionBoundary.m,它绘制了将正例和负例分开的（非线性)决策边界。在 plotDecisionBoundary.m 中,我们通过在均匀间隔的网格上计算分类器的预测来绘制非线性决策边界,然后绘制预测从 y = 0 变为 y = 1 的等高线图。

在学习了参数 θ 之后,ex reg.m 中的下一步将绘制类似于图4 的决策边界。

2.5 可选（未分级)练习

在这部分练习中,您将尝试为数据集尝试不同的正则化参数,以了解正则化如何防止过度拟合。

注意当你改变 λ 时决策边界的变化。使用较小的 λ,您应该会发现分类器几乎可以使每个训练示例都正确,但会绘制非常复杂的边界,从而过度拟合数据（图5）。这不是一个好的决策边界:例如,它预测 x = (−0.25, 1.5) 处的点被接受（y = 1）,这在给定训练集的情况下似乎是一个不正确的决策。

使用较大的 λ,您应该会看到一个显示更简单决策边界的图,该图仍然可以很好地区分正面和负面。但是,如果 λ 设置得太高,您将无法获得良好的拟合,并且决策边界不会很好地遵循数据,从而导致数据拟合不足（图6）。

您无需为这些可选（未评分)练习提交任何解决方案。

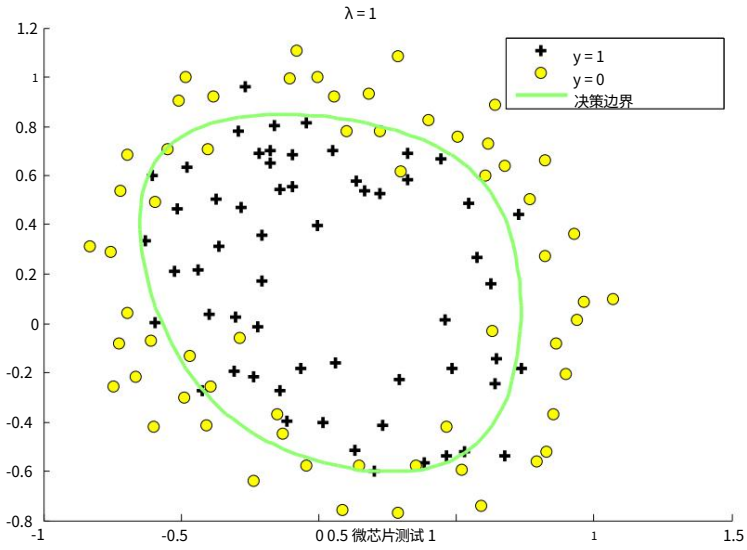

图 4:具有决策边界的训练数据 (λ = 1)

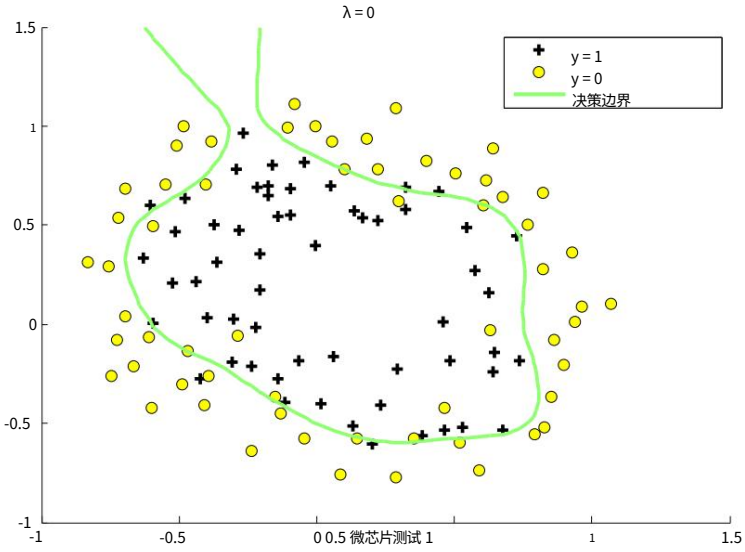

图 5:无正则化（过拟合）（λ = 0）

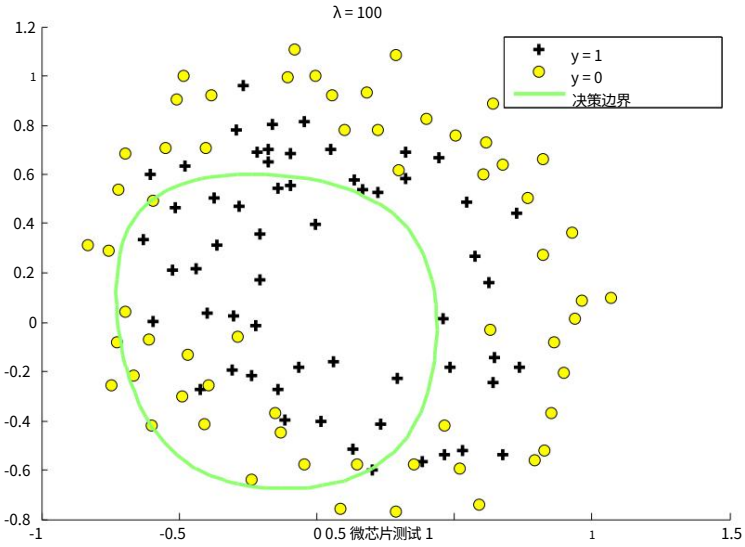

图 6:过多的正则化（欠拟合）（λ = 100）

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。以下是一个
此练习的每个部分的评分方式的细分。

部分	提交文件	积分
Sigmoid 函数 5 分	sigmoid.m 逻	
辑回归的计算成本 costFunction.m 30 分 逻辑回归的梯度 30 分 预测功能 5 分	成本函数.m 预测.m	
正则化 LR costFunctionReg.m 的计算成本 15 分 正则化 LR costFunctionReg.m 的梯度 15 分		
总分 100 分		

您可以多次提交您的解决方案,我们将采取
只考虑最高分。