Artificial Intelligence Experimental Manual

人工智能实验课程手册（下册）

作者：Yunlong Yu

组织：浙江大学信电学院

时间：June 29, 2022

浙江大学

目录

特别声明

欢迎选修人工智能实验课程！

自 2020 年担任《人工智能实验课程》任课老师以来，一直打算撰写适合于机器学习和人工智能课程的实验手册，一方面是课程教学的需求，希望通过此手册使得学生可以更方便地学习人工智能和机器学习的入门知识；另一方面，希望通过撰写这个手册，对基本的知识进行梳理，进一步巩固自己的知识体系，促进科学研究。

本手册为内部资料，作为浙江大学《人工智能实验课程》的参考教材使用，**请勿传播！** 本手册的内容大多来源于《机器学习实战》以及互联网上大量的优秀资料，如维基百科、知乎等网站，在此表示感谢。

本手册分为上下两册。上册主要介绍 Python 语言以及常用的标准库，下册主要介绍常用的人工智能（机器学习）算法及实现过程。

此外，本手册可能存在不少错误，如果发现错误欢迎大家及时反馈至本人邮箱.

Yunlong Yu

前沿

本手册读者对象

选修人工智能实验课程的大三学生。

如果您是人工智能领域的初学者，不必担心，我们会由浅入深地向您介绍人工智能课程的基本方法和所需要的基础知识。

如果您是一位有人工智能或机器学习领域知识的人，可以不必在已掌握的知识上浪费时间，而是直接学习所需要的知识。

本手册主要内容

本手册主要介绍 **Python** 以及基于**Sklearn**的各种机器学习算法实现。本手册服务于人工智能实验课程，因此以实验为主。不过，在尽可能的情况下，本手册将提供人工智能及机器学习的基本的原理。

本手册结构

本手册首先介绍 Python 这一编程语言。从这里开始，会介绍 Python 的基本语法以及编程规则，进而利用 Python 这一工具实现机器学习的基本算法。本手册分为如下 3 个部分。

- 了解 **Python**。首先介绍 Python 的开发历史和优势，然后介绍 Python 的基本语言，包括它的语法和模块。
- 机器学习入门之 **Sklearn** 介绍。
-

第 1 章 机器学习基础

内容提要

在本章中，我们将回顾机器学习中的基础概念，比较监督学习和无监督学习，讨论训练数据、测试数据和验证数据的用法，并了解机器学习应用

1.1 定义机器学习

计算机科学家汤姆·米切尔对机器学习给出了一个更加正式的定义：“如果一个程序的性能在‘T’中体现，通过‘P’来衡量，并通过经验‘E’来提升，那么该程序可以被视为针对一些任务类型‘T’和性能衡量‘P’从经验‘E’中进行学习”。例如，假设你有一个图片集合，每一张图片描绘了一只狗或一只猫。任务是将图片分为狗图片类和猫图片类，而程序可以通过观察已经被分类好的图片来学习执行这个任务，同时它可以通过计算分类图片的正确比例来提升性能。

我们将使用米切尔关于机器学习的定义来组织本章内容。首先，我们将讨论经验的类型，包括监督学习和无监督学习。然后，我们将讨论可以用机器学习系统解决的常见任务。

1.2 从经验中学习

机器学习系统经常被描述为在人类监督或无监督之下从经验中学习。在**监督学习**问题中，一个程序会通过标记的输入和输出进行学习，并从一个输入预测一个输出。也就是说，程序从“正确答案”的例子中学习。在**无监督学习**中，一个程序不会从标记数据中学习。相反，它尝试在数据中发现模式。例如，假设你已经收集了描述人身高体重的数据。一个无监督学习的例子是将数据划分到不同的组中。一个程序可能会产出对应到男性和女性，或者儿童和成人的组。现在假设数据也标记了性别。一个监督学习的例子是归纳出一个规则，基于一个人

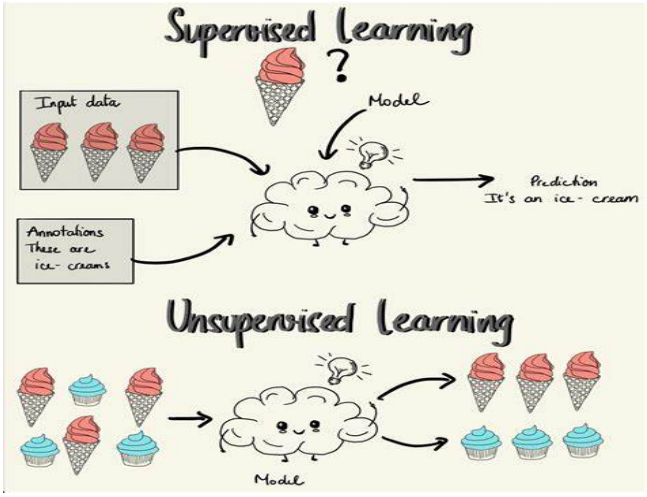


图 1.1: 监督学习和无监督学习示例图

的身高和体重来预测一个人是男性还是女性。我们将在后面的章节中讨论监督学习和无监督学习的算法和例子。

监督学习和无监督学习可以被认为是一个范围的两端。一些类型的问题，被称为半监督学习问题，这些问题同时使用监督学习的数据和无监督学习的数据，位于监督学习和无监督学习之间。**强化学习**靠近监督学习一端。和无监督学习不同，强化学习程序不会从标记的输出对中进行学习。相反，它从决策中接收反馈，但是错误并不会显式地被更正。例如，一个学习去玩像超级玛丽兄弟这样的横向卷轴游戏的强化学习程序，当完成一个关卡或者达到一个特定分数时，可能会接收到一个奖励，而当失去一次生命时会受到惩罚。然而，这样的监督反馈并不会附带一个特定的决策去指挥角色跑动，躲开栗子怪，或者捡起一朵火焰花。我们将主要关注监督学习和无监督学习，因为这两个类别包含了最常见的机器学习问题。在下一节中，我们将更细致地审阅监督学习和无监督学习。

一个监督学习程序从标记输出的例子中进行学习，这些输出例子应该由对应的输入产出。一个机器学习程序的输出有很多名字，在机器学习中汇集了一些学科，许多学科都会使用自己的术语。在本书中，我们将把输出称为响应变量。响应变量的其他名字包括“依赖变量”“回归变量”“标准变量”“测定变量”“应答变量”“被解释变量”“输出变量”“实验变量”“标签”和“输出变量”。类似的，输入变量也有很多名字。在本书中，我们将输入变量称为特征，它们代表的现象称为解释变量。解释变量的其他名字包括“预测器”“回归器”“控制变量”和“暴露变量”。响应变量和解释变量可以是实数值或离散值。

组成监督学习经验的实例集合称为一个训练集。一个用于衡量程序性能的实例集合称为一个测试集。响应变量可以被看作是由解释变量引发问题的回答，监督学习问题会从一个针对不同问题回答的集合中进行学习。也就是说，监督学习程序会被提供正确的答案，而它需要学习去正确地回答没见过的类似问题。

1.3 机器学习任务

两种最常见的监督机器学习任务是分类和回归。在分类任务中，程序必须学习去从一个或多个特征去预测一个或多个响应变量的离散值。也就是说，程序必须为新观测值预测最可能的分类、类别或者标签。分类的应用包括预测一只股票的价格会上涨或下跌，或者决定一篇新闻文章属于政治主题板块还是休闲娱乐板块。在回归问题中，程序必须从一个或多个特征预测一个或多个连续响应变量值。回归问题的例子包括预测一个新产品的销售收入，或者基于一个职位的描述预测其薪水。和分类问题一样，回归问题也需要监督学习。

一个常见的无监督学习任务是在数据集内发现互相关联的观测值群组，称之为聚类。该项任务称为聚类或者聚类分析，会基于一些相似性衡量标准，把观测值放入和其他群组相比相互之间更加类似的群组中。聚类经常用于探索一个数据集。例如，对于一个电影评论集合，一个聚类算法可找出正向评价和负向评价。系统不会将聚合的类标记为正向或者负向。由于缺乏监督，系统只能通过一些衡量标准来判断聚合的观测值相互之间很类似。聚类的一个常见应用是在市场中为一个产品发现客户群体。通过了解特定客户群体的共同属性，销售人员可以决定应该注重销售活动的哪个方面。聚类也应用于网络广播服务中。对于一个歌曲集合，聚类算法可以根据歌曲的特征将歌曲划分为不同的分组。通过使用不同的相似性衡量标准，同

样的聚类算法可以通过歌曲的音调，或者通过歌曲中包含的乐器来为歌曲划分不同的组。

降维是另一种常见的使用无监督学习完成的任务。一些问题可能包含数千或者上百万个特征，这会导致计算能力的极大消耗。另外，如果一些特征涉及噪声或者和潜在的关系无关，程序的泛化能力将会减弱。降维是发现对响应变量变化影响最大的特征的过程。降维还可以用于数据可视化。通过房屋面积预测房屋价格这样的回归问题的可视化很简单，房屋的面积可以作为图的 x 轴，价格可以作为 y 轴。当为房屋价格回归问题添加第二个特征以后，可视化依然很简单，房屋的浴室数量可以作为 z 轴。然而，对于一个包含千万个特征的问题，可视化几乎是不可能的。

1.4 训练数据、测试数据和验证数据

正如前面提到的，一个训练集是一个观测值集合。这些观测值组成了算法用来学习的经验。在监督学习问题中，每一个观测值包含一个观测响应变量和一个或多个观测解释变量特征。测试集是一个类似的观测值集合。测试集被用于使用一些衡量标准来评估模型性能。不把训练集中的观测值包含在测试集中是非常重要的。如果测试集中包含来自训练集中的例子，我们很难评估算法是真的从训练集中学习到了泛化能力，还是只是简单地记住了训练例子。一个能够很好地泛化的程序可以有效地执行一个包含新数据的任务。相反，一个通过学习过于复杂的模型记住了训练数据的程序可以准确地预测训练集中的响应变量，但是无法预测新例子中的响应变量值。对训练集产生记忆称为过拟合。一个对观测值产生记忆的程序会记住训练数据中保持一致的关系和结构因此并不能很好地完成任务。

除了训练数据和测试数据，我们经常需要第三个观测值集合，称为验证集或者保留集。验证集常用来微调被称为超参数的变量，超参数用于控制算法如何从训练数据中学习。在现实世界中程序依然会在测试集上评估，以提供对其性能的估计。由于程序已经被微调过，可以以某种方式从训练数据中学习以提高在验证数据上的得分，因此验证集不应该用来估计现实世界的性能。在现实世界中程序并不会具备在验证数据上的优势。

通常一个监督观测值集合会被划分为训练集、验证集和测试集。划分的每个部分的数量并不会作要求，根据可用数据的数量划分的比例将会有所不同。通常来说，训练集占 50%~75%，测试集占 10%~25%，剩下的则是验证集。

一些训练集可能只包含几百个观测值，其他有的则可能包含数百万个。廉价的存储设备，增强的网络连通性，以及带有传感器智能手机的普及，造就了现代大数据“帝国”，或者说包含数以百万甚至数以十亿计的实例训练集。虽然本课程不会处理需要在几十台乃至数百台计算机上并行处理的数据集，许多机器学习算法预测能力的提升依赖于训练数据数量的增加。然而，机器学习算法同时遵循格言“无用数据入、无用数据出”。假如一个学生通过阅读一本错误百出、令人困惑的大部头教材来准备考试，他的考试成绩并不会比阅读篇幅短小但内容质量较高的教材的学生的成绩好。类似地，在现实世界中，一个算法如果在一个包含噪声、不相关或者错误标签数据的集上进行训练，其表现并不会比一个在包含更能代表问题的小数据集上训练的模型表现好。

许多监督训练数据集需要通过手动或者半自动处理来准备。在一些领域，创建一个大型监督数据集代价不菲。幸运的是，scikit-learn 类库包含了一些数据集，这让开发者可以专注于模

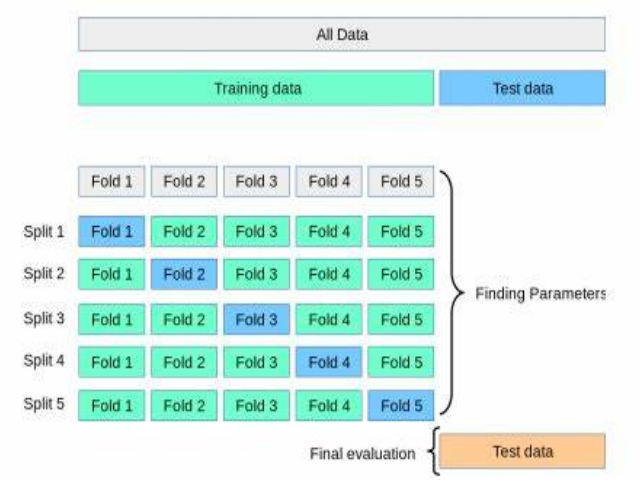


图 1.2: 5-重交叉验证

型实验。在开发过程中，尤其是当训练数据很缺乏时，一种称为交叉验证的实战技巧可用于在同样的数据上训练和验证一个模型。在交叉验证过程中，训练数据被分割为几部分。模型在除了一个部分以外的数据上进行训练，并在剩余的部分上测试。划分被转换几次以便模型可以在全部数据上训练和评估。在现实世界中，每个划分上的模型性能估计得分均值会优于单一的训练/测试划分。图1.2描绘了 5-重交叉验证：

原始数据集被划分为 5 个数量相等的子集，标记 A。最开始模型在划分 B，在划分 A 上测试。在下一次迭代中，模型在划分 A、C、D 和 E 上进行训练，在划分 B 上测试。接着划分被转换直到模型已经在所有的划分上进行训练和测试。相比在单一模型划分上进行测试，交叉验证能为模型提供更准确的性能预估。

1.5 开发机器学习应用程序的步骤

本书学习和使用机器学习算法开发应用程序，通常遵循以下的步骤。

1. **收集数据**。我们可以使用很多方法收集样本数据，如：制作网络爬虫从网站上抽取数据、从 RSS 反馈或者 API 中得到信息、设备发送过来的实测数据（风速、血糖等）。提取数据的方法非常多，为了节省时间与精力，可以使用公开可用的数据源。
2. **准备输入数据**。得到数据之后，还必须确保数据格式符合要求，本书采用的格式是 Python 语言的 List。使用这种标准数据格式可以融合算法和数据源，方便匹配操作。此外还需要为机器学习算法准备特定的数据格式，如某些算法要求特征值使用特定的格式，一些算法要求目标变量和特征值是字符串类型，而另一些算法则可能要求是整数类型。后续章节我们还要讨论这个问题，但是与收集数据的格式相比，处理特殊算法要求的格式相对简单得多。
3. **分析输入数据**。此步骤主要是人工分析以前得到的数据。为了确保前两步有效，最简单的方法是用文本编辑器打开数据文件，查看得到的数据是否为空值。此外，还可以进一步浏览数据，分析是否可以识别出模式；数据中是否存在明显的异常值，如某些数据点与数据集中的其他值存在明显的差异。通过一维、二维或三维图形展示数据也是不错的

方法，然而大多数时候我们得到数据的特征值都不会低于三个，无法一次图形化展示所有特征。本书的后续章节将会介绍提炼数据的方法，使得多维数据可以压缩到二维或三维，方便我们图形化展示数据。这一步的主要作用是确保数据集中没有垃圾数据。如果是在产品化系统中使用机器学习算法并且算法可以处理系统产生的数据格式，或者我们信任数据来源，可以直接跳过第 3 步。此步骤需要人工干预，如果在自动化系统中还需要人工干预，显然就降低了系统的价值。

4. **训练算法**。机器学习算法从这一步才真正开始学习。根据算法的不同，第 4 步和第 5 步是机器学习算法的核心。我们将前两步得到的格式化数据输入到算法，从中抽取知识或信息。这里得到的知识需要存储为计算机可以处理的格式，方便后续步骤使用。如果使用无监督学习算法，由于不存在目标变量值，故而也不需要训练算法，所有与算法相关的内容都集中在第 5 步。
5. **测试算法**。这一步将实际使用第 4 步机器学习得到的知识信息。为了评估算法，必须测试算法工作的效果。对于监督学习，必须已知用于评估算法的目标变量值；对于无监督学习，也必须用其他的评测手段来检验算法的成功率。无论哪种情形，如果不满意算法的输出结果，则可以回到第 4 步，改正并加以测试。问题常常会跟数据的收集和准备有关，这时你就必须跳回第 1 步重新开始。
6. **使用算法**。将机器学习算法转换为应用程序，执行实际任务，以检验上述步骤是否可以在实际环境中正常工作。此时如果碰到新的数据问题，同样需要重复执行上述的步骤。

第 2 章 简单线性回归

在本章中，我们将介绍第一个模型——简单线性回归。简单线性回归围绕一个响应变量和解释变量的某个特征之间的关系进行建模。我们将讨论如何对模型进行拟合，同时也会解决一个玩具问题。虽然简单线性回归对于现实世界的问题几乎不具有可用性，但是理解简单线性回归是理解许多其他模型的关键。在后面的章节中，我们将学到简单线性回归的一般化模型，并将它们运用于现实世界的数据集。

2.1 简单线性回归

在前面的章节中，我们学到了在监督学习问题中用训练数据估计一个模型的参数。用解释变量的观察值及其对应的响应变量组成训练数据，训练好的模型可用于预测未被观测到的解释变量值对应的响应变量值。回顾一下，回归问题的目标是去预测一个连续响应变量的值。在本章中，我们将检验简单线性回归，它常用于对一个响应变量和解释变量的特征之间的关系进行建模。

假设你希望了解披萨的价格。你可能会简单地查看菜单。然而，本书是一本关于机器学习的图书，因此我们将基于能观测到的披萨的属性或者说解释变量，来预测披萨的价格。让我们来对披萨的尺寸和价格之间的关系进行建模。首先，我们将使用 scikit-learn 编写一段程序，通过提供的披萨尺寸来预测其价格。接着我们将讨论简单线性回归如何运行以及如何将其泛化来解决其他类型的问题。

假设你已经记录下了已经吃过的披萨的直径和价格。表**??**中的观测值组成了我们的训练数据。

训练实例	直径（英寸）	价格（美元）
1	6	7
2	8	9
3	10	13
4	14	17.5
5	18	18

我们可以使用 matplotlib 作图来将训练数据可视化:

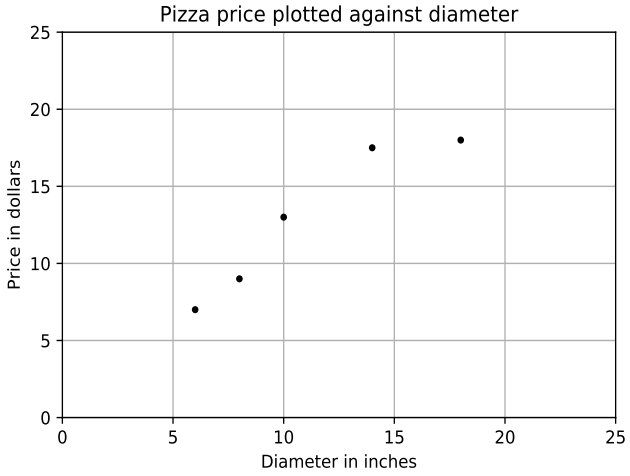
```
import numpy as np
import matplotlib.pyplot as plt

X = np.array([[6], [8], [10], [14], [18]]).reshape(-1, 1)
y = [7, 9, 13, 17.5, 18]

plt.figure()
plt.title('Pizza price plotted against diameter')
plt.xlabel('Diameter in inches')
plt.ylabel('Price in dollars')
plt.plot(X, y, 'k.')
plt.axis([0, 25, 0, 25])
```

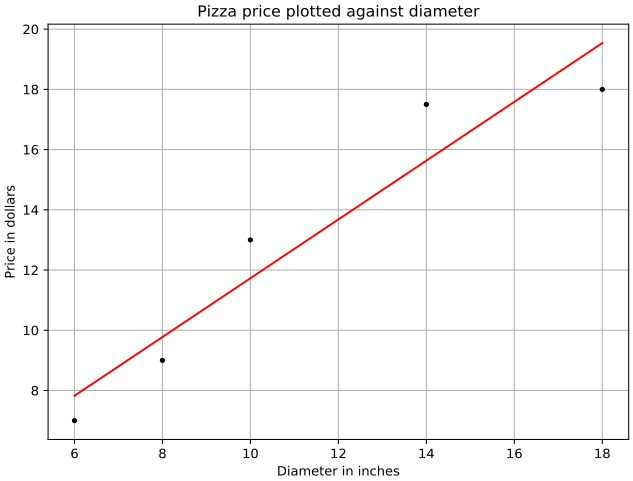
```
plt.grid(True)
plt.show()
```

简单线性模型假设响应变量和解释变量之前存在线性关系，它使用一个被称为超平面的线性面来对这种关系进行建模。一个超平面是一个子空间，它比组成它的环绕空间小一个维度。在简单线性回归中共有两个维度，一个维度表示响应变量，另一个维度表示解释变量。因此，回归超平面只有一个维度，一个一维的超平面是一条直线。



LinearRegression类是一个估计器。估计器基于观测到的数据预测一个值。在 scikit-learn 中，所有的估计器都实现了 fit 方法和 predict 方法。前者用于学习模型的参数，后者使用学习到的参数来预测一个解释变量对应的响应变量值。使用 scikit-learn 可以非常简单地对不同模型进行实验，因为所有的估计器都实现了 fit 和 predict 方法，尝试新的模型只需要简单地修改一行代码。LinearRegression 的 fit 方法学习了公式2.1简单线性回归模型的参数：

$$y = \alpha + \beta x \tag{2.1}$$

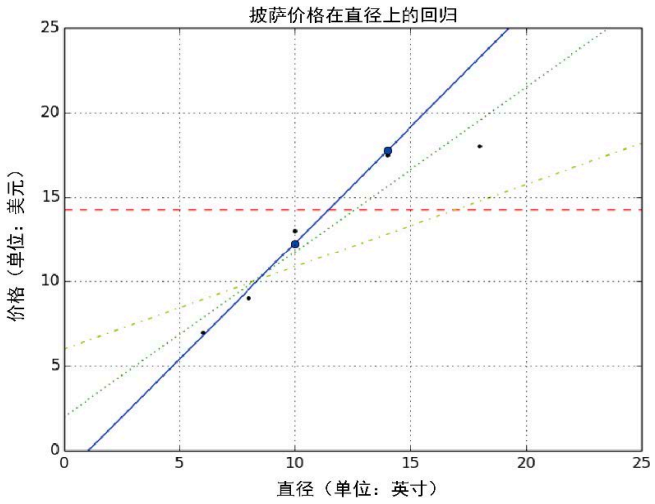


在上面的公式中，y 是响应变量的预测值，在这个例子里，它表示披萨的预测价格。x 表示解释变量。截断项 α 和系数 β 都是可以通过学习算法学到的模型参数。在图**??**中，绘制的

超平面对一个披萨的价格和尺寸之间的关系进行建模。

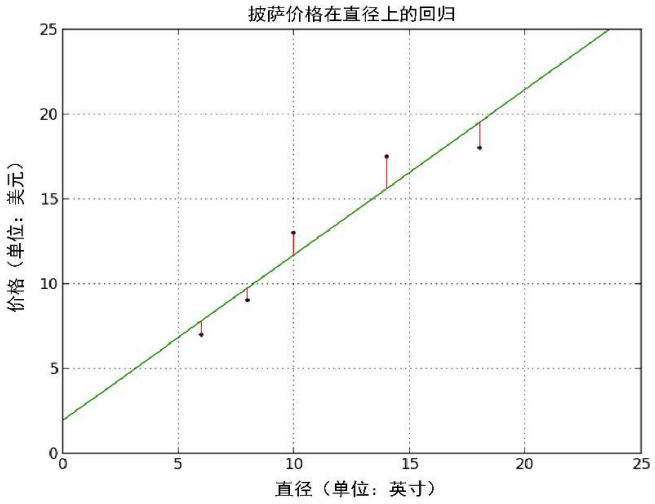
利用训练数据学习产生最佳拟合模型的简单线性回归的参数值称为普通最小二乘（Ordinary Lease Squares，OLS）或线性最小二乘法。接下来我们将讨论一种解析出模型参数值的方法。

2.1.1 用代价函数评价模型的拟合性



在图**??**中我们根据一些参数集合的值绘制出几条回归线。然而我们如何去评估哪组参数值产生了最佳拟合回归线呢?

代价函数，也被称为**损失函数**，它用于定义和衡量一个模型的误差。由模型预测出的价格和在训练数据集中观测到的披萨价格之间的差值被称为残差或者训练误差。稍后，我们将使用一个单独的测试数据集来评价模型。在测试数据中预测值和观测值之间的差值叫作预测误差或者测试误差。在图**??**中，模型的残差由训练实例点和回归超平面之间垂直线表示。



我们可以通过最小化残差的和来生成最佳披萨价格预测器。也就是说，对于所有训练数据而言，如果模型预测的响应变量都接近观测值，那么模型就是拟合的，这种衡量模型拟合的方法叫作残差平方和（RSS）代价函数。在形式上，该函数通过对所有训练数据的残差平方求

和来衡量模型的拟合性。RSS 由下面方程的公式2.2计算出，其中 y_i 是观测值，$f(x_i)$ 是预测值：

$$SS_{res} = \sum_{i=1}^{n}(y_i - f(x_i))^2 \tag{2.2}$$

```
from sklearn.linear_model import LinearRegression
modelRegL = LinearRegression()
modelRegL.fit(X, y)
yFit = modelRegL.predict(X)
print('Residual sum of squares: %.2f' \% np.mean((modelRegL.predict(X) - y)**2))
```

2.1.2 求解简单线性回归的 OLS

笔记 OLS: Ordinary Least Square

在这一部分中，我们将求解出简单线性回归的 OLS。回想一下，简单线性回归由方程 $y = \alpha + \beta x$ 给出，而我们的目标是通过求代价函数的极小值来求解出 β 和 α 的值。首先我们将解出 β 值，为了达到目的，我们将计算 x 的方差以及 x 和 y 的协方差。方差用来衡量一组值偏离程度，如果集合中的所有数值都相等，那么这组值的方差为 0。方差小意味着这组值都很接近总体均值，而如果集合中包含偏离均值很远的数据则集合会有很大的方差。方差可以使用下面的公式2.3算出：

$$var(x) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1} \tag{2.3}$$

$\bar{x}$ 表示 x 的均值，x_i 是训练数据中的第 i 个 x 的值，n 表示训练数据的总量。我们来计算一下训练数据中披萨直径的方差。

```
import numpy as np
X = np.array([[6], [8], [10], [14], [18]]).reshape(-1, 1)
x_bar = X.mean()
print(x_bar)
# 注意我们在计算样本方差的时候将样本的数量减去1
# 这项技巧称为贝塞尔校正，它纠正了对样本中总体方差估计的偏差
variance = ((X -x_bar)**2).sum() / (X.shape[0] - 1)
print(variance)
```

Numpy 库也提供了一个叫作 var 的方法来计算方差。计算样本方差时关键字参数 ddof 可以设置贝塞尔校正：

```
import numpy as np
print(np.var(X, ddof=1))
```

协方差用来衡量两个变量如何一同变化。如果变量一起增加，它们的协方差为正。如果一个变量增加时另一个变量减少，它们的协方差为负。如果两个变量之间没有线性关系，它们的协方差为 0，它们是线性无关的但不一定是相对独立的。协方差可以使用下面的公式2.4计算：

$$cov(x,y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n-1} \tag{2.4}$$

和方差一样，x_i 表示训练数据中第 i 个 x 的值，表示直径的均值，表示价格的均值，y_i 表示训练数据中第 i 个 y 的值，n 表示训练数据的总量。我们来计算一下训练数据中披萨半径和价格的协方差，

```
# 之前我们使用一个列表表示y
# 在这里我们改为使用一个Numpy多位数组，它包含了几个计算样本均值的方法
y = np.array([7, 9, 13, 17.5, 18])
y_bar = y.mean()
# 我们将X转置，因为所有的操作都必须是行向量
covariance = np.multiply((X - x_bar).transpose(), y - y_bar).sum()/(X.shape[0] - 1)
print(covariance)
print(np.cov(X.transpose(), y)[0][1])
```

现在我们已经计算出了解释变量的方差以及解释变量和响应变量之间的协方差，可以使用公式2.5解出 β 值：

$$\begin{aligned} \beta &= \frac{cov(x,y)}{var(x)} \\ \beta &= \frac{22.65}{23.2} = 0.98 \end{aligned} \tag{2.5}$$

解出 β 值以后我们可以使用公式2.6解出 α 值：

$$\alpha = \bar{y} - \beta\bar{x} \tag{2.6}$$

现在我们已经通过求代价函数的极小值解出了模型的参数值，可以带入披萨的直径预测它们的价格。例如，一个 11 英寸的披萨预计花费 12.70 美元，一个 18 英寸的披萨预计花费 19.54 美元。恭喜！你已经使用简单线性回归预测了披萨的价格。

2.2 评价模型

我们已经使用了一种学习算法从训练数据中估计出了模型的参数。我们如何评估模型是否很好地表达了现实中解释变量和响应变量之间的关系呢？假设你找到了另一页披萨菜单，我们将使用这页菜单中的条目作为测试数据集来衡量模型的表现。表**??**是一个包含 4 列数据的表格，其中包含了由我们的模型预测出的披萨价格。

训练实例	直径（英寸）	真实价格（美元）	预测价格（美元）
1	8	11	9.7759
2	9	8.5	10.7522
3	11	15	12.7048
4	16	18	17.4863
5	12	11	13.6811

我们可以使用一些衡量方法来评估模型的预测能力。在此我们使用一种叫作 **R** 方的方法

来评估披萨价格预测器。R 方，也被称为**决定系数**，它用来衡量数据和回归线的贴近程度。计算 R 方的方法有多种，在简单线性回归模型中，R 方等于**皮尔森积差相关系数（PPMCC）**的平方，也被称为皮尔森相关系数 r 的平方。使用该计算方法，R 方必须是 0 和 1 之间的正数，其原因很直观：如果 R 方描述的是由模型解释的响应变量中的方差的比例，这个比例不能大于 1 或者小于 0。其他一些计算方法，包括 scikit-learn 库使用的方法，不使用皮尔森相关系数 r 的平方公式计算 R 方。如果模型的表现非常差，由这些计算方法求出的 R 方可能为负值。了解性能指标的局限性非常重要，R 方对于异常值尤其敏感，当新的特征增加到模型中时，它常常会出现异样的增长。

我们通过 scikit-learn 使用的方法来计算披萨价格预测器的 **R** 方。首先我们需要算出平方总和。y_i 是第 i 个测试实例的响应变量观测值，是响应变量的观测值均值，如公式2.7所示。

$$\begin{aligned} SS_{tot} &= \sum_{i=1}^{n}(y_i - \bar{y})^2 \\ SS_{tot} &= (11 - 12.7)^2 + (8.5 - 12.7)^2 + ... + (11 - 12.7)^2 = 56.8 \end{aligned} \tag{2.7}$$

其次我们需要算出 RSS。回顾一下此公式和前面提到的代价函数的计算公式相同，如公式2.8所示：

$$\begin{aligned} SS_{res} &= \sum_{i=1}^{n}(y_i - f(x_i))^2 \\ SS_{res} &= (11 - 9.78)^2 + (8.5 - 10.75)^2 + ... + (11 - 13.68)^2 = 19.20 \end{aligned} \tag{2.8}$$

最后，我们使用公式2.9计算出 R 方：

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{19.20}{56.8} = 0.66 \tag{2.9}$$

R^2 计算得分为 0.662，这表明测试实例价格变量的方差很大比例上是可以被模型解释的。现在用 scikit-learn 类库来印证我们的计算结果。

LinearRegression 类的 score 方法返回了模型的 R 方值。

```
import numpy as np
from sklearn.linear_model import LinearRegression
X_train = np.array([6, 8, 10, 14, 18]).reshape(-1, 1)
y_train = [7, 9, 13, 17.5, 18]
X_test = np.array([8, 9, 11, 16, 12]).reshape(-1, 1)
y_test = [11, 8.5, 15, 18, 11]
model = LinearRegression()
model.fit(X_train, y_train)
r_squared = model.score(X_test, y_test)
print(r_squared)
```

第 2 章 练习

1. 有 20 名学生，研究学生在考前复习时间与得到的成绩之间的相关关系。

```
from collections import OrderedDict
import pandas as pd
examDict={
    '学习时间':[0.50,0.75,1.00,1.25,1.50,1.75,1.75,2.00,2.25,
          2.50,2.75,3.00,3.25,3.50,4.00,4.25,4.50,4.75,5.00,5.50],
    '分数':   [10, 22, 13, 43, 20, 22, 33, 50, 62,
           48, 55, 75, 62, 73, 81, 76, 64, 82, 90, 93]
}
examOrderDict=OrderedDict(examDict)
exam=pd.DataFrame(examOrderDict)

>>> exam.head()
   学习时间  分数
0  0.50  10
1  0.75  22
2  1.00  13
3  1.25  43
4  1.50  20
```

问题一：此数据适不适合用线性回归的模型？为什么？

问题二：划分训练集与测试集

问题三：模型的简单线性回归方程

问题四：评估模型精度。

问题五：分析模型。