第1章 机器学习基础

1.1 何谓机器学习

简单的说，机器学习就是把无序的数据转换成有用的信息。

1.1.1 传感器和海量数据

移动计算和传感器产生的海量数据意味着未来我们将面临着越来越多的数据，如何从海量数据中抽取到有价值的信息将是一个非常重要的课题。

1.1.2 机器学习非常重要

针对具体任务搞懂所有相关数据的意义所在，这正成为基本的技能要求。

大量的经济活动都依赖于信息，我们不能在海量的数据中迷失，机器学习将有助于我们穿越数据雾霭，从中抽取出有用的信息。

1.2 关键术语

表1-1　基于四种特征的鸟物种分类表

	体重（克）	翼展（厘米）	脚　蹼	后背颜色	种　属
1	1000.1	125.0	无	棕色	红尾鵟
2	3000.7	200.0	无	灰色	鹭鹰
3	3300.0	220.3	无	灰色	鹭鹰
4	4100.0	136.0	有	黑色	普通潜鸟
5	3.0	11.0	无	绿色	瑰丽蜂鸟
6	570.0	75.0	无	黑色	象牙喙啄木鸟

下面测量的这四种值（体重、翼展、有无脚蹼以及后背颜色）称之为**特征**，也可以称作属性，但本书一律将其称为特征。表1–1中的每一行都是一个具有相关特征的**实例**。

机器学习的一个主要任务就是**分类**。根据实例的输入特征，判定实例的类别。

最终我们决定使用某个机器学习算法进行分类，首先需要做的是算法训练，即学习如何分类。通常我们为算法输入大量已分类数据作为算法的**训练集**。训练集是用于训练机器学习算法的数据样本集合，表1–1是包含六个训练样本的训练集，每个**训练样本**有4种特征（体重、翼展、有无脚蹼以及后背颜色）、一个**目标变量**（种属）。目标变量是机器学习算法的预测结果，在分类算法种目标变量的类型通常是标称型的，而在回归算法种通常是连续型的。训练样本集必须确定知道目标变量的值，以便机器学习算法可以发现特征和目标之间的关系。我们通常把分类问题中的目标变量称为**类别**，并假定分类问题只存在有限个数的类别。

注意：特征或着属性通常是训练样本集的列，它们是独立测量得到的结果，多个特征联系在一起共同组成一个训练样本。

为了测试机器学习算法的效果，通常使用两套独立的样本集：**训练数据**和**测试数据**。当机器学习程序开始运行时，使用训练样本集作为算法的输入，训练完成之后输入测试样本。输入测试样本时并不提供测试样本的目标变量，由程序决定样本属于哪个类别。比较测试样本预测的目标变量值与实际样本类别之间的差别，就可以得出算法的实际精确度。

假定这个鸟类分类程序，经过测试满足精确度要求，是否我们就可以看到机器已经学会了如何区分不同的鸟类了呢？这部分工作称之为**知识表示**，某些算法可以产生很容易理解的知识表示，而某些算法的知识表示也许只能为计算机所理解。

1.3 机器学习的主要任务

机器学习另一项任务是**回归**，它主要用于预测数值型数据。大多数人可能都见过回归的例子——数据拟合曲线：通过给定数据点得到最优的拟合曲线。分类和回归属于**监督学习**，之所以称之为监督学习，是因为这类算法必须知道预测什么，在本部分中即目标变量的分类信息。

与监督学习相对应的是**无监督学习**，此时数据没有类别信息，也不会给定目标值。在无监督学习中，将数据集合分成由类似的对象组成的多个类的过程被称为**聚类**；将寻找描述数据统计值的过程

称之为**密度估计**。此外，无监督学习还可以减少数据特征的维度，以便我们可以使用二维或三维图形更加直观地展示数据信息。表1-2列出了机器学习的主要任务，以及解决相应问题的算法。其中很多算法都可以用于解决同样的问题，下一节将回答算法选择的问题。

表1-2　用于执行分类、回归、聚类和密度估计的机器学习算法

监督学习的用途	
k-近邻算法	线性回归
朴素贝叶斯算法	局部加权线性回归
支持向量机	Ridge 回归
决策树	Lasso 最小回归系数估计
无监督学习的用途	
K-均值	最大期望算法
DBSCAN	Parzen窗设计

1.4 如何选择合适的算法

在选择实际可用的算法时，必须考虑下面两个问题：

i. 使用机器学习算法的目的，想要算法完成何种任务，比如是预测明天下雨的概率还是对投票者按照兴趣分组

ii. 需要分析或收集的数据是什么

首先考虑使用机器学习算法的目的。如果想要预测目标变量的值，则可以选择监督学习算法，否则可以选择无监督学习算法。确定选择监督学习算法之后，需要进一步确定目标变量类型，如果目标变量是离散型，如是/否、1/2/3、A/B/C或者红/黄/黑等，则可以选择**分类器算法**；如果目标变量是连续型的数值，如0.0～100.00、-999～999或者+∞～-∞等，则需要选择**回归算法**。

如果不想预测目标变量的值，则可以选择无监督学习算法。进一步分析是否需要将数据划分为离散的组。如果这是唯一的需求，则使用聚类算法；如果还需要估计数据与每个分组的相似程度，则需要使用密度估计算法。

其次需要考虑的是数据问题。我们应该充分了解数据，对实际数据了解得越充分，越容易创建符合实际需求的应用程序。主要应该了解数据的以下特性：特征值是离散型变量还是连续型变量，特征值中是否存在缺失的值，何种原因造成缺失值，数据中是否存在异常值，某个特征发生的频率如何（是否罕见得如同海底捞针），等等。充分了解上面提到的这些数据特性可以缩短选择机器学习算法的时间。

我们只能在一定程度上缩小算法的选择范围，一般并不存在最好的算法或者可以给出最好结果的算法，同时还要尝试不同算法的执行效果。对于所选的每种算法，都可以使用其他的机器学习技术来改进其性能。在处理输入数据之后，两个算法的相对性能也可能会发生变化。后续章节我们将进一步讨论此类问题，一般说来发现最好算法的关键环节是反复试错的迭代过程。

机器学习算法虽然各不相同，但是使用算法创建应用程序的步骤却基本类似，下一节将介绍 如何使用机器学习算法的通用步骤。

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。本书使用Python语言构造算法应用，不熟悉的读者可以学习附录A。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。

下节我们将讨论实现机器学习算法的编程语言Python。之所以选择Python，是因为它具有其他编程语言不具备的优势，如易于理解、丰富的函数库（尤其是矩阵操作）、活跃的开发者社区等。

1.6 Python 语言的优势

基于以下三个原因，我们选择Python作为实现机器学习算法的编程语言：(1) Python的语法清晰；(2) 易于操作纯文本文件；(3) 使用广泛，存在大量的开发文档。

1.6.1 可执行伪代码

Python具有清晰的语法结构，大家也把它称作可执行伪代码（executable pseudo-code）。默认安装的Python开发环境已经附带了很多高级数据类型，如列表、元组、字典、集合、队列等，无需进一步编程就可以使用这些数据类型的操作。使用这些数据类型使得实现抽象的数学概念非常简单。此外，读者还可以使用自己熟悉的编程风格，如面向对象编程、面向过程编程、或者函数式编程。不熟悉Python的读者可以参阅附录A，该附录详细介绍了Python语言、Python使用的数据类型以及安装指南。

Python语言处理和操作文本文件非常简单，非常易于处理非数值型数据。Python语言提供了丰富的正则表达式函数以及很多访问Web页面的函数库，使得从HTML中提取数据变得非常简单直观。

1.6.2 Python 比较流行

Python语言使用广泛，代码范例也很多，便于读者快速学习和掌握。此外，在开发实际应用程序时，也可以利用丰富的模块库缩短开发周期。

在科学和金融领域，Python语言得到了广泛应用。SciPy和NumPy等许多科学函数库都实现了向量和矩阵操作，这些函数库增加了代码的可读性，学过线性代数的人都可以看懂代码的实际功能。另外，科学函数库SciPy和NumPy使用底层语言（C和Fortran）编写，提高了相关应用程序的计算性能。本书将大量使用Python的NumPy。

Python的科学工具可以与绘图工具Matplotlib协同工作。Matplotlib可以绘制2D、3D图形，也可以处理科学研究中经常使用到的图形，所以本书也将大量使用Matplotlib。

Python开发环境还提供了交互式shell环境，允许用户开发程序时查看和检测程序内容。Python开发环境将来还会集成Pylab模块，它将NumPy、SciPy和Matplotlib合并为一个开发环境。在本书写作时，Pylab还没有并入Python环境，但是不远的将来我们肯定可以在Python开发环境找到它。

1.6.3 Python 语言的特色

Python语言是高级编程语言，我们可以花费更多的时间处理数据的内在含义，而无须花费太多精力解决计算机如何得到数据结果。Python语言使得我们很容易表达自己的目的。

1.6.3 Python 语言的缺点

Python语言唯一的不足是性能问题。Python程序运行的效率不如Java或者C代码高，但是我们可以使用Python调用C编译的代码。这样，我们就可以同时利用C和Python的优点，逐步地开发机器学习应用程序。我们可以首先使用Python编写实验程序，如果进一步想要在产品中实现机器学习，转换成C代码也不困难。

如果程序是按照模块化原则组织的，我们可以先构造可运行的Python程序，然后再逐步使用C代码替换核心代码以改进程序的性能。C++ Boost库就适合完成这个任务，其他类似于Cython和PyPy的工具也可以编写强类型的Python代码，改进一般Python程序的性能。如果程序的算法或者思想有缺陷，则无论程序的性能如何，都无法得到正确的结果。如果解决问题的思想存在问题，那么单纯通过提高程序的运行效率，扩展用户规模都无法解决这个核心问题。从这个角度来看，Python快速实现系统的优势就更加明显了，我们可以快速地检验算法或者思想是否正确，如果需要，再进一步优化代码。

本节大致介绍了本书选择Python语言实现机器学习算法的原因，下节我们将学习Python语言的shell开发环境以及NumPy函数库。

1.7 NumPy 函数库基础

机器学习算法涉及很多线性代数知识，因此本书在使用Python语言构造机器学习应用时，会经常使用NumPy函数库。如果不熟悉线性代数也不用着急，这里用到线性代数只是为了简化不同的数据点上执行的相同数学运算。将数据表示为矩阵形式，只需要执行简单的矩阵运算而不需要复杂的循环操作。在你使用本书开始学习机器学习算法之前，必须确保可以正确运行Python开发环境，同时正确安装了NumPy函数库。NumPy函数库是Python开发环境的一个独立模块，而且大多数Python发行版没有默认安装NumPy函数库，因此在安装Python之后必须单独安装NumPy函数库。

> **NumPy矩阵与数组的区别**
> NumPy函数库中存在两种不同的数据类型（矩阵matrix和数组array），都可以用于处理行列表示的数字元素。虽然它们看起来很相似，但是在这两个数据类型上执行相同的数学运算可能得到不同的结果，其中NumPy函数库中的matrix与MATLAB中matrices等价。
> 调用mat()函数可以将数组转化为矩阵。

1.8 本章小结

尽管没有引起大多数人的注意，但是机器学习算法已经广泛应用于我们的日常生活之中。每天我们需要处理的数据在不断地增加，能够深入理解数据背后的真实含义，是数据驱动产业必须具备的基本技能。

学习机器学习算法，必须了解数据实例，每个数据实例由多个特征值组成。分类是基本的机器学习任务，它分析未分类数据，以确定如何将其放入已知群组中。为了构建和训练分类器，必须首先输入大量已知分类的数据，我们将这些数据称为训练样本集。

尽管我们构造的鸟类识别专家系统无法像人类专家一样精确地识别不同的鸟类，然而构建接近专家水平的机器系统可以显著地改进我们的生活质量。如果我们可以构造的医生专家系统能够达到人类医生的准确率，则病人可以得到快速的治疗；如果我们可以改进天气预报，则可以减少水资源的短缺，提高食物供给。我们可以列举许许多多这样的例子，机器学习的应用前景几乎是无限的。

第一部分的后续6章主要研究分类问题，它是监督学习算法的一个分支，下一章我们将介绍第一个分类算法——k-近邻算法。