MLAPP 读书笔记 - 01 概论

> A Chinese Notes of MLAPP，MLAPP 中文笔记项目
> https://zhuanlan.zhihu.com/python-kivy

记笔记的人：cycleuser

2018年05月06日14:04:48

本来不想发的，但是 Github 上面的公式阅读有问题。
等会再看，我这会正在一个个修正公式呢，他妹的蝌蝌蛄叫唤老子还不种地了？
本文内容仅仅是读书笔记，并非对原文的忠实翻译，且不包含任何原文图件。

1.1 机器学习：概念和目的

> “我们被信息淹没，对知识感到饥饿。” John Naisbitt

人类进入大数据时代了。（译者注：现在有一种乱象，就是什么都说大数据啊人工智能啊，这些头猪已经就跟云计算一样四处飞啊，懂和不懂的都跟着吹捧。）

这么多数据就需要自动化方法来进行数据分析，因此需要机器学习。机器学习也就可以定义成一系列能够自动检测数据模式的方法集合，这些方法可以将发现的模式用于对未来数据的预测，或者对其他具有不确定性过程的选取提供参考（例如如何收集更多的数据等）。

1.1.1 机器学习的类别

机器学习一般分成两种主要类型。

第一种就是**预测类**或者说**监督学习**，通过学习算法得到一个从输入特征 x 到输出 y 的映射关系（mapping），给定一个带有标签的输入输出集合对 $D = \{(x_i, y_i)\}_{i=1}^{N}$。这样的集合 D 就叫训练集，N 就是训练集的样本数。

简单来说，每个训练样本的输入特征 x_i 是一个值为数值的 D 维向量，可能就是一个人的身高体重等等。这样的指标叫做特征（features）、属性（attributes）、或者 协变量（covariates）。不过这个 x_i 也可以是一个复杂结构对象，比如一个突破、一句话、一封邮件、一个时间序列、一个分子形状、一个图结构等等。

与此相似，输出（output）或 响应变量（response variable）也可以是任意形式的。

第二种就是**描述性**或者叫**无监督学习**。这里的给定训练集中只有输入值，$D = \{(x_i)\}_{i=1}^N$，目标是要从数据中找到感兴趣的模式。这种方法也被叫做知识发现（knowledge discovery）。

第三种**强化学习**（reinforcement learning），用于给定奖励或者惩罚信号下的行为响应。如小孩学走路等等。

1.2 监督学习

先开始讨论的是监督学习，这个在实践中用的最广泛。

1.2.1 分类问题

分类问题的核心就是对输入特征 x 和输出值 y 之间建立映射。
y 属于一个有限集合，有限集合的元素个数为C。
如果 C = 2，为二分类问题，可假设 y 属于集合{0,1}。
如果 C > 2，为多类别分类。

若分类标签不是互斥的，比如一个人可以即被形容为 富，同时也被形容为 丑，为多标签分类问题。

1.2.1.1 分类问题样本

本节简单又无聊，略

1.2.1.2 概率预测

要处理不确定的情况，就要用到概率了。如果对概率论的基本概念不熟悉了，可以等着本书第二章那里有回顾。
给定输入向量 x 和训练集 D，我们将在可能的分类标签上的概率分布表示为 $p(y|x, D)$。还记得 y 所属集合的元素个数 C 么？一般来说刚刚这个概率分布表示了一个长度为 C 的向量。如果只有两个分类，自然是所有分类的概率加一起等于1，$p(y = 1|x, D) + p(y = 0|x, D) = 1$，那么只用其中一个概率 $p(y = 1|x, D)$就足够了。这里用 $p(y|x, D)$来表示，|右边有 x 和 D，意思是这个概率是在测试样本中输入值 x 和训练集 D 上的条件概率分布。当选用不同的模型的时候，还可以加一个 M 在|右边，表示所选模型，也就是 $p(y|x, D, M)$。如果行文背景中已经对模型有充分讲述了，就可以把 M 去掉，只写作 $p(y|x, D)$了。

给定某个可能的概率化的输出，就可以计算出这个猜测是真实标签的概率了：

$$\hat{y} = \hat{f}(x) = argmax_{c=1}^{C} p(y = c|x, D) \quad (1.1)$$

上面这个形式对应的就是最可能的分类标签，也被叫做概率分布 $p(y|x, D)$的众数（mode），

也成为最大后验估计（MAP estimate， maximum a posteriori）。更正规详细的介绍在本书的5.7。

如果一个 $p(\hat{y}|x, D)$ 远小于1，也就是说这个答案不让人很有信心，这时候与其返回一个不太可信的结果就不如直接返回不确定了。在医疗和金融等对风险敏感的领域尤其如此。这一部分还列举了 IBM 的 Watson 和 Google 的 SmartASS 系统等等，在此忽略。

1.2.1.3 现实世界中的应用

分类可能是机器学习最广泛的用途了，用于很多有意思又难以人力解决的现实问题。

文档分类和垃圾邮件过滤

文档分类的目标是对一个文档，比如网页或者电子邮件信息，分到 C 种类别当中的某一种，也就是要计算 p(y=c|x,D)，其中的输入特征 x 是文本的某种信息。
垃圾邮件过滤，分类只有两种，是垃圾邮件，y=1；不是则 y=0.

大多数分类器都假设输入特征向量 x 有固定尺寸。可变长度文档的特征向量格式可以使用词汇袋（bag of words）来表示。细节可以参考本书的 3.4.4.1，不过基本思想还是很简单的，就是如果单词 j 出现在了文档 i 里面，则 $x_{ij} = 1$。如果把这种变换应用到数据集中的所有文档，就得到了一个由文档和词汇组成的二元共生矩阵，如图1.2所示。

这样就从一个文档分类问题简化到一个寻找模式中的细微变化的问题了。比如，可能很多邮件含有一些特定关键词，可以借助这些来分类。在练习8.1和8.2里面，你就要自己动手来使用不同分类技巧来进行垃圾邮件识别。

鸢尾花分类

图1.3是另外一个例子，来自统计学家 Ronald Fisher。生物学家已经将有用特征进行了统计，这些特征包括：萼片长度、宽度，花瓣长度、宽度。

从照片等复杂对象到数据这个过程叫做特征提取（feature extraction），这个过程非常重要，又特别困难。

特征提取还往往可能被人忽略，如果没有进行充分的特征提取，是根本不可能充分对分类对象进行分类的。本书后面的章节会讲到从数据中提取好的特征的一些方法。

如图1.4所示，对鸢尾花数据集进行散点图投图，很明显可以检查花瓣长宽来讲 setosas（红色圆圈）和其他两种区分开。然而另外两种 versicolor 和 virginica 的区分就稍微难一点了，需要至少使用两组特征。在进行机器学习之前，可以将数据先投图看看，这样可以进行一些探索性的数据分析，是个很好的办法。

图片分类和手写识别

接下来这个问题就更难了，要直接对图片进行分类，这些图片都是没有预处理的数据。可能需要先整体分类一下，比如是室内的还是室外场景，是横着还是竖着拍摄的，是否有小狗等等，这就叫做图形分类。

有一个特例就是判断手写的字母数字，例如对于邮编之类的，就可以进行这种手写识别。这个用途也有个标准数据集，叫做 MNIST，是 Modified National Institute of Standards 的缩写，这个数据集里面的图片都做了预处理，保证了大多数数值或者字幕都在图片中心位置。这个数据集有6000个训练样本图片和 10000 个测试样本图片，内容都是由不同的人写的数字0到9。每个图片都是 28x28 像素大小，灰度值都是从0到255的。图1.5（a）是一些样本。

很多通用分类方法会忽略输入特征的结构，例如空间布局等等。因此，这些方法可以用于处理图1.5（b） 所示的数据，这份数据是同一份数据对所有特征进行了随机排列。这个过程在练习1.1当中。这种灵活性既是好事，也是噩梦，是好事因为这些方法可以适用于通用目的，是噩梦因为这些方法忽略了很明显的有用信息。本书后面会讨论利用输入特征结构信息的方法。

人脸检测和识别

这个问题就比上一个更难了，要从一个图片中找到某些对象，这也叫做对象检测或者对象定位。一个例子就是人脸检测。解决这个问题的一种方法是把图片切分成叠覆在不同位置、大小和方向的小块，然后对这些个小块来检测是否含有人脸形状的结构。这种方法也叫做滑动窗口检测（sliding windows detector）。这个系统会返回像人脸概率最高的区域的位置。如图1.6所示。

这种方法目前应用于数码相机里面，比如索尼微单就有人脸识别和眼控对焦，都是基于这种思路。另外一种用途是在谷歌街景之类的系统中把人脸模糊掉。

找到了面孔之后，就可以继续进行人脸识别了，就是对该面孔所有者的身份进行识别。这个过程中用到的分类标签可能很多很多。另外这个过程用到的这些特征都和人脸检测问题中有所不同，例如在人脸识别的过程中，面孔的细微差别，例如发行等等，对于身份确定都是很重要的；而对于人脸检测来说，这类细节都是无关紧要的，而关注的核心是人脸与非人脸之间的差别。

1.2.2 回归

回归和分类其实挺相似的，核心区别在于分类的响应变量 y，也就是分类标签，是离散的，是一个有限集合中的元素；而回归中的响应变量 y 是连续的。
如图1.7所示，有一个单实数值的输入特征 $x_i \in R$，然后也有一个单实数值的响应变量 $y_i \in R$。我们可以考虑对这个样本数据使用两种模型进行拟合，一个如图1.7左图所示，用直线，另外一个如右图所示，用二次曲线拟合。其实这时候就能引出很多扩展问题了，比如如果输入特征是高维度的怎么办、异常值怎么处理、非光滑的响应变量怎么办等等。

现实世界中的回归问题有很多，比如：
* 根据给定的市场状况和其他方面的信息预测明天的股票市场价格

* 预测在 YouTube 上观看某个特定视频的观众年龄
* 预测一个机器人手臂在三维空间中的位置，对其一系列不同的马达发送控制信号控制扭矩
* 根据不同的临床检测结果来预测人体中前列腺特异抗原（prostate specific antigen, PSA）的规模

1.3 无监督学习

从数据中发现结构，也叫知识发现（knowledge discovery）。
要进行密度估计（density estimation），建立 $p(x_i|\theta)$的模型。

两个主要的区别：

1. 概率形式不一样，写成的是$p(x_i|\theta)$而不是$p(y_i|x_i,\theta)$。监督学习是条件密度估计，而无监督学习是非条件密度估计。
2. 无监督学习中的特征 x_i 是一个特征向量，要建立多元概率模型。而监督学习中的y_i通常是单值的，是用来去预测的。监督学习算法多用单变量概率模型，有与输入相关的参数（input-dependent parameters）。不过多输出分类的监督学习也涉及了多元概率模型。

无监督学习就跟人学习过程类似。应用领域更广，不需要人为对数据进行标签分类，成本低，信息密度大。分类的数据不仅很昂贵，而且包含的信息相对来说也少了些，对于复杂模型的参数估计来说就可能不够可靠了。多伦多大学的 Geoffrey Hinton 是机器学习领域的著名教授，他之前说：
当我们学着观看世界的时候，是没有人告诉我们什么是正确答案的，我们就是用眼睛去看而已。有时候可能你妈妈会告诉你那个是小狗，不过这只是非常少的信息。这样的过程中，你能得到几个 bit 的信息就挺幸运了，甚至哪怕每秒钟只有一个 bit 的信息都很不错了。你的大脑视觉系统有 10^14 数量级的神经元链接在一起，而你的生存时间只有 10^9 秒。所以每秒只有 1bit 信息获取的学习过程是没啥用的。你需要更多信息，比如每秒 10^5 bit 的信息。这么多信息只能从观察到的输入本身里面获得。---Geoffrey Hinton,1996,Quoted in Gorder 2006。

下面是无监督学习的一些例子。

1.3.1 聚类分析

此处看原书图1.8

聚类分析，简单说就是把数据分成不同的组。
设 K 表示了可分的组的数目。
首先对在分组数上的分布 p(K|D)估计，知道数据中是否具有子群。
通常用p(K|D)的众数（mode）来近似，得到$K \triangleq \arg\max_K p(K|D)$。

无监督学习里面，我们可以随意选择自己喜欢的分组数。
选一个具有“合适复杂度（right complexity）”的模型的过程就叫做模型选择。

接下来的目标就是估计每个点属于哪个组了。假设有 $z_i \in \{1, \ldots, K\}$，表示了每个点 i 所归属的组的序号。
z 就是潜在变量（hidden/latent variable），因为不能从训练集中直接观察到。
计算 $z_i \triangleq \arg\max_K p(z_i = k|x_i, D)$就行了。

本书重点是基于模型的聚类，也就是先对数据拟合一个概率模型，而不是直接运行特定算法。

聚类分析的应用范例：
* 天文学里对天文观测的聚类分析，自动分类系统（Cheeseman 等，1988）帮助发现了一种新恒星。
* 电子商务里面，基于购买类型或者网页浏览习惯等等，使用聚类将用户分组，发送定制广告（Berkhin 2006）。
* 生物学里面，对流式细胞术数据（flow-cytometry data）的聚类分组，将细胞分成不同子类（Lo 等，2009）。

1.3.2 发掘潜在变量

此处看原书图1.9

处理高维数据可能要把高维数据投影到包含了数据关键信息的低维度子空间内。这个过程就是降维。

背后思路是数据可能呈高维度状态，但与潜在变量对应的可能只有少数几个项目而已。

此处看原书图1.10

当作为对其他统计模型的输入的时候，降维通常能提高的预测准确性，因降维后聚焦在研究对象的“本质特征”上，滤除了次要信息。
另外，低维可用于快速邻域搜索，二维投影对于高维数据可视化也有帮助。

降维的常用主成分分析，PCA，principal components analysis。
可看做是无监督的（多输出）线性回归，观测的是高维度响应变量 y，而并不是低维度的“诱因” z。
模型是从 z 到 y 的对应关系。还要逆转这个对应关系，从观测到的高维 y 中推出潜在的低维 z，这部分在本书的12.1。

降维和 PCA，用的很多，比如：
* 生物学里面用 PCA 来解析基因片段数据，说明每个测试结果通常是多个基因通过所属的不同生物学路径协同行为的结果。

* 自然语言处理中，用一种叫做潜在语义分析的 PCA 衍生方法来进行文档恢复（本书27.2.2）。
* 在信号处理中，比如声学信号或者神经信号等等，常常使用 ICA（独立成分分析，主成分分析的一种变体）来区分不同的信号源（本书12.6）。
* 计算机图形学中，常会将动作捕捉数据投影到低维度空间，然后用来建立动画。参考本书的15.5。

1.3.3 发掘图结构量

此处看原书图1.11

对一系列的相关数据进行测试，找出某个变量和哪个变量最相关。
可以用图 G 来表示，图中的节点为各变量，线段表示变量的依赖关系，在本书第10章详细说。
可以从数据中使用学习算法得到这个图模型 $\hat{G} = \arg\max p(G|D)$。

稀疏图学习总体上两种应用：发掘新知识，更好的联合概率密度估计。

样例如下：
* 很多这种稀疏图学习算法的动机来自于系统生物学社区。例如，加入我们测试了一个细胞里蛋白质的磷酸关系状态（Sachs 等等，2005）。图1.11展示的就是这样得到的一个图结构，详细方法在本书26.7.2中有讲述。
* 某些情况下，对于解析图结构本身我们没多大兴趣，而只是想用它来对相关性进行建模然后来进行预测。例如金融证券组合管理，其中大规模的不同证券股票之间的相关性是极其重要的。

此处看原书图1.12

1.3.4 矩阵补全

数据缺失的情况，有的变量的值不知道。
例如，问卷调查，然后有的人可能没回答一些问题。
又如，有不同的传感器，某些可能不工作了。
结果就是矩阵中有“孔洞”，用 NaN 来表示，意思是 Not A Number，即值非数值。
对遗失值进行插补（imputation）就是要用合适的值来填充这些空白位置。这个过程也叫做矩阵补全。

1.3.4.1 图像修复

遗失值插补的一个例子就是图片修复。
图1.12所示，对图片降噪，然后插入像素填补了空白位置。
具体方法是对图中清楚的部分建立一个联合概率模型，然后推测给定变量（像素位置）的未知变量（像素）。
有点像购物篮分析（market basket analysis），除了数据是实数值和具有空间结构，用到的概率

模型也有所不同。更多细节参考本书的19.6.2.7和13.8.4。

1.3.4.2 协同筛选

此处看原书图1.13

协同筛选（collaborative filtering）。
基于人们对已看过电影的评分来预测他们想看的电影。
关键思路在于这个预测并不是基于电影或者用户的特征（虽然也可以用这些特征），而基本主要是基于一个评分矩阵 X，两个方向分别是电影和用户，矩阵的值就是用户i对电影i的评分，1到5。
X 当中很多值都可能是空或者未知，大多数用户未必对大多数电影打过分。
所以观察到只是小子集，预测的是另外一个子集。

本书的27.6.2详细讲述了相关内容。

1.3.4.3 购物篮分析

商业数据挖掘里面的一个很受关注的项目。

数据包含一个二值矩阵，规模很大又稀疏。
每一列表征的是一个商品，每行表示一笔交易。
如果在第 i 次交易中商品 j 被购买了，就设置 $x_{ij} = 1$，否则为0。

很多商品可能会被一起购买，比如面包和黄油等等，所以在这些数据中会有相关性。
给定一个新的观察到的部分向量，表征了消费者购买项目的一个子集，目标就是去预测其他位置的可能值，表示消费者是否会购买其他项目。

和协同筛选不同的是，这里通常假设没有缺失数据。因为每个消费者过去的购买行为都是已知的。

除了购买模式之外，这个问题也适用于很多其他领域。例如，类似的技术可以用于复杂软件系统中的文件相关性的建模分析。这时候任务目的就是根据给定的已被修改的文件自己来预测其他文件是否需要被上传来确保数据连续性（Hu 等2010）。

这个问题通常要用到频繁集挖掘算法（frequent itemset mining），这一方法建立关联规则（Hastie 等，2009）。另外也可以用一种概率方法，对向量拟合一个联合密度模型 $p(x_1, \ldots, x_D)$（Hu 等2010）。

相比关联规则，概率方法通常能够提供更精确的预测，不过解释起来就不如关联规则的方法好解释了。

数据挖掘和机器学习的一个显著区别：数据挖掘注重对是得到可解释的模型，机器学习注重得到

精确模型。

此处看原书图1.14

1.4 机器学习中的一些基本概念

1.4.1 参数化模型和非参数化模型

本书主要关注概率模型，形式是$p(y|X)$或者$p(x)$，取决于监督学习还是无监督学习。

最重要的区别是看模型是否有固定数目的参数，或者参数的数量是否随着训练集规模的增长而增长。
前一种就是参数化模型（parametric models），后一种则是非参数化模型（non-parametric models）。

参数化模型的优点是用起来更快速，而对数据分布的自然特征进行更强假设的时候就不如非参数化模型。
非参数化模型更加灵活，但对于大规模数据集来说在计算上比较困难。

1.4.2 K 最邻近算法，一个简单的非参数化模型分类器

此处看原书图1.15

这个方法就是“查看”训练集中与输入值 x 最邻近的 K 个点，然后计算样本中每一类有多少个成员包含于这个集合中，然后返回经验分数作为估计值，如图1.14所示。更正规的表示法如下所示：

$$p(y=c|x,D,K)=\frac{1}{K}\sum_{i\in N_{K(x,D)}}\prod(y_i=c) \quad (1.2)$$

其中的$N_{K(x,D)}$是在 D 中和点 x 最近的 K 个点的索引，而，$\prod(e)$则是指示函数，其中的 e 表示判断条件，如果 e 为真则$\prod(e)=1$，否则$\prod(e)=0$。

$$\prod(e)=\begin{cases}1 & \text{if e is true}\\ 0 & \text{if e is false}\end{cases}$$

(1.3)

这个方法属于一种基于记忆的学习（memory-based learning），也是基于实例的学习（instance-based learning）。具体的概率推导过程在本书的14.7.3。

此处看原书图1.16

最常用的距离矩阵就是欧氏距离，不过这也限制了适用的范围，因为要求数据必须是实数值的。

图1.15是上述方法的一个实例，其中输入是二维的，有三个类，设置了 K=10。图1.15（a）当中是对训练数据的投影，图1.15（b）则是对$p(y=1|x,D)$的投图，图1.15（c）则是对$p(y=2|x,D)$的投图。不用对$p(y=3|x,D)$再去投图，因为概率相加等于1，所以有其他两个就能确定$p(y=3|x,D)$了。图1.15（d）是对最大后验估计（MAP estimate）的$\hat{y}(x)=argmax_c p(y=c|x,D)$ 投图。

K =1 的 KNN 分类器就会生成一个沃罗诺伊镶嵌（Voronoi tessellation，Voronoi diagram），也叫狄利克雷镶嵌（Dirichlet tessellation）或泰森多边形（Thiessen polygon），如图1.14（b）所示。这个过程实际上是把整个的区域$V(x_i)$分配给每个点x_i，在$V(x_i)$中的所有点到 x_i的距离比其他所有点更近。在每个细胞格内，预测的标签就是训练集中对应点的标签。

1.4.3 维度诅咒

在得到一个比较好的距离矩阵并且有充分标签的训练集的情况下，KNN 分类器简单又好用。如果 N 趋向于无穷大，KNN 分类器的性能差不多是最佳性能的一半了（Cover and Hart 1967）。

不过当面对高维度输入的时候，KNN 分类器就不太好用了。这种高维度下的悲惨性能表现是由于维度诅咒（curse of dimensionality）。

具体解释很简单，略了。

1.4.4 分类和回归的参数模型

要解决维度诅咒这个难题，主要方法就是对于数据的自然特征进行一些假设，比如对于监督学习就是$p(y|x)$，对于无监督学习就是$p(x)$。这些假设也就是所谓的归纳偏见（inductive bias），经常是存在于参数模型当中。参数模型是有着固定参数数目的统计模型。下面会介绍两个广泛应用的例子，本书后文中会深入讲解包括这两个在内的更多模型。

1.4.5 线性回归

最广泛使用的回归模型就是线性回归（linear regression）。即设响应变量为输入变量的线性函数。写出来如下所示：

$$y(x)=w^Tx+\epsilon=\sum_{j=1}^{D}w_jx_j+\epsilon \text{(1.4)}$$

上式中的w^Tx表示的是输入向量 x 和模型的权重向量（weight vector）w（统计学中对回归权重更常用的记号是β）的内积（inner product）或者数积（scalar product）。而ϵ则是线性预测和真实值之间的残差（residual error）。

此处看原书图1.17

通常我们会假设残差ϵ遵循高斯分布，即正态分布。记作$\epsilon \sim N(\mu, \sigma^2)$，其中$\mu$是均值，$\sigma$是方差，更多细节第二章会讲。对这个分布投图，就会得到钟形曲线（bell curve），如图1.17所示。

要在线性回归和高斯分布之间建立更确切的联系，可以用下面这种形式重写模型：

$$p(y|x,\theta) = N(y|\mu(x), \sigma^2(x)) \text{(1.5)}$$

很明显，这个模型这样就是一个条件概率密度了。在最简单的情况下，可以假设μ是 x 的线性函数，所以$\mu = w^T x$，而设噪音为固定的，即$\sigma^2(x) = \sigma^2$。这样$\theta = (w, \sigma^2)$就是模型参数了。

假如输入特征x是1维的，就可以用下面的形式表示响应变量：
$$\mu(x) = w_0 + w_1 x = w^T X \text{(1.6)}$$
上式中的w_0就是偏项（bias term）,w_1是斜率（slope），此处的向量 X 定义为$X = (1, x)$。这里在输入向量上加一个预设常数1是一个常用技巧，这样就可以把截距项（intercept term）和模型中其他项目结合起来。如果w_1符号为正，就意味着输出随着输入的增大而增大。这如图1.17（b）所示；图1.17（a）所示的是响应变量均值和 x 之间关系的投图。

对非线性关系模型，可以把线性回归中的 x 替换成某种对输入的非线性函数$\phi(x)$，也就是如下面所示的形式：

$$p(y|x,\theta) = N(y|w^T \phi(x), \sigma^2) \text{ (1.7)}$$

此处看原书图1.18

这个过程即基函数扩展（basis function expansion）。例如图1.18所示，其中的$\phi(x) = [1, x, x^2, \ldots, x^d]$，图1.18（a）中 d=14，图1.18（b）中 d=20，这两个都是多项式回归（polynomial regression）。
本书的后面章节会提到其他的基函数。实际上很多流行的机器学习算法都可以看作是从数据估计基函数的不同方法而已，比如支持向量机（support vector machines）、神经网络（neural networks）、分类和回归树等等。这部分内容在本书14和16章。

1.4.6 逻辑回归

此处看原书图1.19

对于分类问题，尤其是二分类问题，可以对线性回归做两个修改。首先是对 y 不再使用高斯分布，而是换用伯努利分布，这在$y \in \{0, 1\}$二分类问题的情况下更近似。如下所示：

$$p(y|x,w) = Ber(y|\mu(x)) \text{(1.8)}$$

其中的$\mu(x) = E[y|x] = p(y = 1|x)$。接下来计算输入变量的一个线性组合，和之前的不同在于要通过一个函数来处理一下，以保证$0 \leq \mu(x) \leq 1$：

$$\mu(x) = sigm(w^T x) \text{(1.9)}$$

上面的$sigm(\eta)$是指 S 形函数（sigmoid function）， 也叫做逻辑函数（logistic function 或 logit function）。这个函数的定义如下所示：

$$sigm(\eta). = \frac{1}{1+exp(-\eta)} = \frac{e^\eta}{e^\eta+1} \text{(1.10)}$$

Sigmoid 的意思就是 s 形，如图1.19（a）所示。这个函数也叫做压缩函数（squashing function），因为此函数会把线条约束在0到1的闭区间内。这种约束很有必要，因为输出值要被表达成概率的形式，根据定义就能知道概率就是在这个0到1的闭区间内的。

把上面两个步骤结合起来就得到了下面的式子：

$$p(y|x,w) = Ber(y|sigm(w^T x)) \text{(1.11)}$$

这个就叫做逻辑回归，虽然看上去有点像线性回归，但实际上逻辑回归是一种分类，而并不是回归。

图1.19（b）展示了一个简单的逻辑回归样例，其中投影了下面的函数：

$$p(y_i = 1|x_i, w) = sigm(w_0 + w_1 x_i) \text{(1.12)}$$

图1.19（b）中的x_i是学生 i 的 SAT（理解为美国高考）的分数，而y_i是判定他们是否通过一门课。黑色实心点是训练集，红色圆点是$p(y = 1|x_i, \hat{w})$，这里的$\hat{w}$就是从训练集估计出来的参数，具体计算过程在本书的8.3.4。

如果我们在概率为0.5的位置分解，就引入了决策规则（decision rule），形式如下所示：
$$\hat{y}(x) = 1 \iff p(y = 1|X) > 0.5 \text{(1.13)}$$

如图1.19（b）中，$sigm(w_0 + w_i x) = 0.5$的时候，$x \approx 545 = x*$。所以可以在这个位置画一条竖线，作为所谓的决策边界（decision boundary）。这条边界左侧设为0，右侧设为1。

这里要注意到，决策规则即便在训练集上都有一个非零的错误比例（non-zero error rate）。这是因为数据本身并不是线性可分的（linearly separable），也就是说我们并不能画一条线区分0和1.使用基函数扩展，可以创建非线性决策边界的模型，就跟非线性回归类似。更多样例都在本书后文中。

1.4.7 过拟合

当拟合高灵活度模型的时候，要非常小心过拟合（overfit），也就是要避免去对输入的所有微小变动都进行建模，因为这些很可能是噪音而不是真正的信号。如图1.18（b）所示，其中用高次多项式拟合得到的曲线七扭八歪。真实的函数是不太可能有这么严重的振荡的。因此用这样的模型对未来输出就可能无法得到准确预测。

再举一个例子，比如 KNN 分类器，K 值的选择会对模型行为有很大影响。如果 K=1，那自然是在训练集上不会有误差，因为这就相当于只是返回了训练集各点本身的标签而已，但得到的与侧面就会很纠结了，如图1.20（a）所示.这样此方法对于预测未来数据就不一定很适合。在图1.20（b）当中设置了 K=5，这时候预测面就更光滑多了，因为在从一个更广泛的邻域来进行均值取值。随着 K 的增长，预测面会越来越光滑，知道 K 达到了临界值，也即是 K=N，这时候就是在预测整个数据集上的主要分类标签了。后文会详细讲解如何选择“正确”的 K 值。

此处看原书图1.20

1.4.8 模型选择

我们可能有很多可选的模型，复杂度各自不同，比如线性回归、逻辑回归，不同的多项式次数，或者 KNN 分类器使用不同 K 值，那么该怎么选择呢？一个很自然的方法是计算各个方法在训练样本上的误分类率（misclassification rate）。

这个误分类率(misclassification rate)的定义如下：

$$err(f, D) = \frac{1}{N} \sum_{i=1}^{N} \prod(f(x_i) \neq y_i) \text{(1.14)}$$

上式中的 f(x) 就是咱们的分类器了。如图1.21（a）中的是这个错误率和 KNN 分类器的 K 值之间关系的曲线。很明显随着 K 的增长训练集上的错误率会增加，因为这个过程增加了光滑程度。如上文所述，我们可以使用 K-1来让整个训练集上的错误率最低，这样就只是存储训练集数据而已。

泛化误差（generalization error），也就是未来数据的平均分类误差率的期望值，更多细节参考本书6.3。可以通过对大规模相互独立的测试数据集的误分类率进行计算而近似得到，而不是用在模型训练过程中的。在图1.21（a）当中用红色时薪曲线投了测试误差和 K 之间的曲线。这个曲线是 U 形的，对于复杂模型（小 K 值）过拟合，对于简单模型（大 K 值）欠拟合。

要选择测试误差最小的 K 值，在图1.21（a）当中从10到100都可以。

可以把训练集分成两部分，一部分用来训练模型，另一部分用作验证集（validation set）来挑选模型复杂度。
在训练集上拟合所有不同模型，然后在验证集上面评估不同模型的性能，从中择优。
选好了最佳模型之后，再用这个模型对全部数据进行拟合。
如果我们有一个单独的测试集，可以在这个测试集上面进行评估，来估计所选方法的准确性，更多相关细节在本书6.5.3。

通常从数据集中选择80%抽样作为训练集，另外的20%作为验证集。不过如果训练集规模很小，这样做训练集的数据可能不太充足，不够训练模型，甚至不能对未来的情况进行可靠估计了。

一种简单又很流行的解决方案就是使用交叉验证（cross validation，CV）。这个想法很简单：

把训练集分成 K 份，对每个$k \in \{1, \ldots, K\}$都训练除了第 k 份之外的所有数据，然后在第 k 份上进行验证，就这样轮流进行，如图1.21（b）所示。然后对所有份上的误差率计算均值，用这个作为测试误差的近似值。这里要注意每个点只被预测过一次，而在训练过程中则被用到了 K-1 次。这种方法就叫做 K 折交叉验证（K-fold CV），通常设置 K=5，称为5折交叉验证。如果设置 K=N，那么这样就成了留一法交叉验证（leave-one out cross validation，LOOCV），因为对于每一份 i 都训练了除了 i 样本之外的所有其他数据，而在 i 上进行测试。联系1.3就让你用5折交叉验证估计测试误差和 K 的关系。然后和图1.21（a）当中的先验误差（empirical error）进行比较。

上面这些挑选 KNN 分类器的 K 值问题其实是一种更广泛问题的特例，这类问题都是模型选择（model selection），其中我们的任务就是在不同灵活度的模型支架做出选择。交叉验证被广泛用于这类问题，当然本书后文还会介绍一些其他方法。

此处看原书图1.21

1.4.9 没有免费的午餐

所有的模型都是错的，不过有的模型还是有用的。———— George Box（Box and Draper 1987，p424）

机器学习多数关注的是推导出不同的模型，用不同的算法对这些模型进行拟合。我们可以用交叉验证之类的方法来先去选择一个对于所用问题最适合的方法。然而，并没有通用的最佳模型，这也被称为“没有免费的午餐理论（no free lunch theorem）”（Wolpert 1996）。这一现象是因为在某个领域比较合适的一些假设，用到其他情况就可能不太靠谱了。

因此，我们需要开发很多不同类别的模型，来覆盖现实世界中各种不同的数据。对于每个模型，都有很多不同的算法能用来训练该模型，还要在速度、准确性、复杂度之间进行权衡妥协。接下来的各个章节里面我们要学习的就是数据、模型、算法的结合。

练习 1

练习 1.1 对打散的MNIST数据使用 KNN 分类器

运行本书配套程序 pmtk3 中的 mnist1NNdemo，在前1000个测试案例上验证 K=1 的KNN 分类器的误分类率为3.8%。如果你运行到全部10,000个测试样本上，误差率应该是3.09%。
然后修改一下代码，对特征（训练和测试矩阵的列）进行随机置换，参考 shuffledDigitsDemo，然后再运行同一个分类器，验证一下误分类率没变。

练习 1.2 估计 KNN 分类器

使用https://github.com/mariusmuja/flann 这里的代码，原文中的链接http://people.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN 已经失效了，结合 mnist1NNdemo 来对 MNIST 数据进行分类。看看能加速多少，然后准确度是否有下降?

练习 1.3 对 KNN 分类器进行交叉验证

使用本书配套程序 pmtk3 中的 knnClassifyDemo 来投图测试集上误分类率的交叉验证估计。和图1.21（a）进行对比，讨论测试误差率的相似和不同。