# 第 1 章
# 机器学习概述

本章内容分为3节。第1节主要介绍机器学习（Machine Learning，ML）的基本概念、机器学习三要素和核心、机器学习开发流程、机器学习模型评价指标。在详细学习机器学习之前，需要了解什么是机器学习、机器学习的开发流程以及机器学习模型评价指标。第2节主要介绍机器学习的发展及分类。从传统机器学习到现代机器学习，机器学习已经在人工智能、自然语言处理、图像识别、医学、金融等领域得到广泛的应用，成为当今信息技术发展中的热点和趋势之一。那么，机器学习这一路的发展历程是什么样的呢？在第2节我们会详细探讨这个问题。第3节主要介绍机器学习中常用的术语。机器学习是一个复杂性、专业性很强的技术领域，它大量应用了概率论、统计学、逼近论、算法复杂度等多门学科的知识，因此会出现很多专业性很强的词汇。在探索机器学习的初级阶段，理解这些专业术语是学习过程中的第一件重要任务，因此第3节将详细介绍机器学习中常用的术语，为后续的知识学习打下坚实的基础。

本章主要知识点：

- ❖ 机器学习简介
- ❖ 机器学习的发展史和分类
- ❖ 机器学习常用术语

## 1.1 机器学习简介

在正式学习机器学习之前，先了解什么是机器学习、机器学习三要素和核心、机器学习的开发流程、机器学习模型评价指标，以及机器学习项目开发步骤。

### 1.1.1 什么是机器学习

机器学习是人工智能的一个分支。人工智能的研究历史有着一条从以“推理”为重点，到以“知识”为重点，再到以“学习”为重点的自然、清晰的脉络。显然，机器学习是实现人工

智能的一个途径，即以机器学习为手段解决人工智能中的问题。机器学习经过30多年发展，已经成为一门多领域交叉学科，涉及概率论、统计学、逼近论、凸分析（Convex Analysis）、计算复杂性理论等多门学科。机器学习理论主要是设计和分析一些让计算机可以自动“学习”的算法。机器学习算法是一类从数据中自动分析获得规律，并利用规律对未知数据进行预测的算法。因为学习算法中涉及大量的统计学理论，所以机器学习与推断统计学联系尤为密切，因此也被称为统计学习理论。算法设计方面，机器学习理论关注可以实现的、行之有效的学习算法。很多推论问题无程序可循，难度较大，所以部分机器学习的研究是开发容易处理的近似算法。

机器学习已广泛应用于数据挖掘、计算机视觉、自然语言处理、生物特征识别、搜索引擎、医学诊断、检测信用卡欺诈、证券市场分析、DNA序列测序、语音和手写识别、战略游戏和机器人等领域。

简单地说，机器学习是一门从数据中研究算法的科学学科，它根据已有的数据进行算法的选择，并基于算法和数据结构构建模型，最终对未来进行预测。它的实质是通过数据构建一个模型并用于预测未知属性。

### 1.1.2　机器学习三要素和核心

#### 1. 机器学习三要素

按照统计机器学习的观点，任何一个机器学习方法都是由模型（Model）、策略（Strategy）和算法（Algorithm）三个要素构成的。具体可理解为机器学习模型在一定的优化策略下，使用相应求解算法来达到最优目标的过程。

机器学习的第一个要素是模型。机器学习中的模型就是要学习的决策函数或者条件概率分布，一般用假设空间（Hypothesis Space）$F$来描述所有可能的决策函数或条件概率分布。

当模型是一个决策函数时，如线性模型的线性决策函数，$F$可以表示为若干决策函数的集合，$F=\{f \mid Y=f(X)\}$，其中$X$和$Y$为定义在输入空间和输出空间中的变量。

当模型是一个条件概率分布时，如决策树是定义在特征空间和类空间中的条件概率分布，F可以表示为条件概率分布的集合，$F=\{P \mid P=Y \mid X\}$，其中$X$和$Y$为定义在输入空间和输出空间中的随机变量。

机器学习的第二个要素是策略。简单来说，就是在假设空间的众多模型中，机器学习需要按照什么标准选择最优模型。对于给定模型，模型输出$f(X)$和真实输出$Y$之间的误差可以用一个损失函数（Loss Function），也就是$L(Y, F(X))$来度量。

不同的机器学习任务都有对应的损失函数，回归任务一般使用均方误差，分类任务一般使用对数损失函数或者交叉熵损失函数等。

机器学习的最后一个要素是算法。这里的算法有别于所谓的“机器学习算法”，在没有特别说明的情况下，“机器学习算法”实际上指的是模型。作为机器学习三要素之一的算法，指的是学习模型的具体优化方法。当机器学习的模型和损失函数确定时，机器学习就可以具体地形式化为一个最优化问题，可以通过常用的优化算法，例如随机梯度下降法（Stochastic Gradient Descent，SGD）、牛顿法、拟牛顿法等，进行模型参数的优化求解。

当一个机器学习问题的模型、策略和算法都确定了，相应的机器学习方法也就确定了，因而这三者也叫作“机器学习三要素”。

### 2. 机器学习的核心

机器学习的目的在于训练模型，使其不仅能够对已知数据有很好的预测效果。而且能对未知数据有较好的预测能力。当模型对已知数据预测效果很好，但对未知数据预测效果很差的时候，就引出了机器学习的核心问题之一——过拟合（Over-Fitting）。

先来看一下监督机器学习的核心哲学。总的来说，所有监督机器学习都可以用如下公式来概括：

$$\theta^* = \operatorname{argmin} \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, f\left(x_i; \theta\right)\right) + \lambda \phi_\theta$$

上面的公式就是监督机器学习中的损失函数计算公式，其中，第一项为针对训练集的经验误差项，即我们常说的训练误差；第二项为正则化项，也称为惩罚项，用于对模型复杂度进行约束和惩罚。

因此，所有监督机器学习的核心任务就是在正则化参数的同时最小化经验误差。多么简约的哲学啊！各类机器学习模型的差别无非就是变着方式改变经验误差项，即我们常说的损失函数。不信你看：当第一项是平方损失（Square Loss）时，机器学习模型便是线性回归；当第一项变成指数损失（Exponential Loss）时，模型则是著名的AdaBoost（一种集成学习树模型算法）；而当损失函数为合页损失（Hinge Loss）时，便是大名鼎鼎的SVM了！

综上所述，第一项“经验误差项”很重要，它能变着法儿改变模型形式，我们在训练模型时要最大限度地把它变小。但在很多时候，决定机器学习模型质量的关键不是第一项，而是第二项“正则化项”。正则化项通过对模型参数施加约束和惩罚，让模型时时刻刻保持对过拟合的警惕。

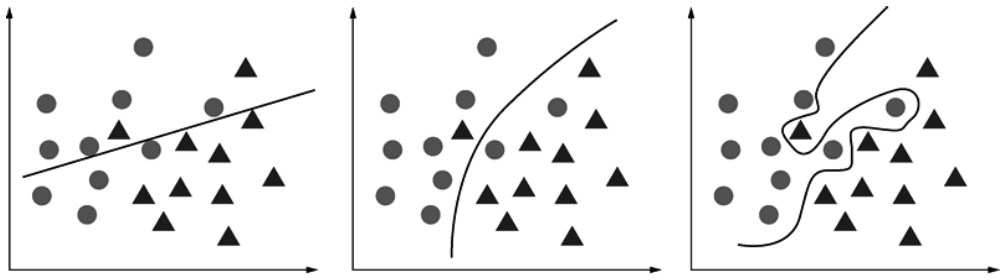
我们再回到前面提到的监督机器学习的核心任务：正则化参数的同时最小化经验误差。通俗来讲，就是训练集误差小，测试集误差也小，模型有着较好的泛化能力；或者模型偏差小，方差也小。

但是很多时候模型的训练并不尽如人意。当你在机器学习领域摸爬滚打已久时，想必更能体会到模型训练的艰辛，要想训练集和测试集的性能表现高度一致实在太难了。很多时候，我们已把经验损失（即训练误差）降到极低，但模型一到测试集上，瞬间“天崩地裂”，表现得一塌糊涂。这种情况便是下面要谈的主题——过拟合。

所谓过拟合，指在机器学习模型训练的过程中，模型对训练数据学习过度，将数据中包含的噪声和误差也学习了，使得模型在训练集上表现很好，而在测试集上表现很差的一种现象。机器学习简单而言就是归纳学习数据中的普遍规律，这里一定得是普遍规律，像这种将数据中的噪声也一起学习了的，归纳出来的便不是普遍规律，而是过拟合。

欠拟合、正常拟合与过拟合的表现形式如图1-1所示。

鉴于过拟合十分普遍并且关乎模型的质量，因此在机器学习实践中，与过拟合长期坚持不懈地斗争是机器学习的核心。而机器学习的一些其他问题，诸如特征工程、扩大训练集数量、算法设计和超参数调优等，都是为防止过拟合这个核心问题而服务的。



欠拟合　　正常拟合　　过拟合

图 1-1　欠拟合、正常拟合、过拟合的表现形式

### 1.1.3　机器学习开发流程

机器学习开发流程在实际操作层面一共分为8步：需求分析、数据收集、数据预处理、数据分析与可视化、建模调优、特征工程、模型结果展示与分析报告、模型部署与上线反馈优化。如图1-2所示。

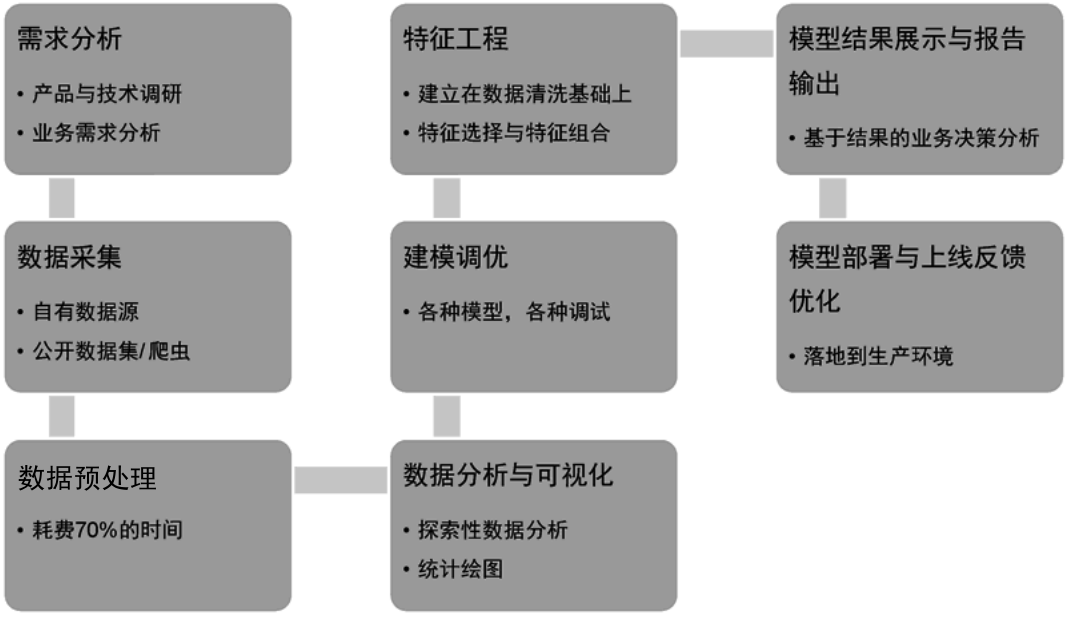



图 1-2　机器学习开发流程

#### 1. 需求分析

很多算法工程师可能觉得需求分析没有技术含量，因而不太重视项目启动前的需求分析工作，这对于一个项目而言其实是非常危险的。

需求分析的主要目的是为项目确定方向和目标，为整个项目的顺利开展制订计划和设立里程碑。我们需要明确机器学习的目标输入是什么，目标输出是什么，是回归任务还是分类任务，关键性能指标都有哪些，是结构化的机器学习任务还是基于深度学习的图像和文本识别任务，市面上项目相关的产品都有哪些，对应的SOTA（State Of The Art）模型有哪些，相关领域的

前沿研究和进展都到什么程度了，项目有哪些有利条件和风险。这些都需要在需求分析阶段认真考虑。

### 2. 数据采集

一个机器学习项目要开展下去，最关键的资源就是数据。

在数据资源相对丰富的领域，例如电商、O2O、直播以及短视频等行业，企业一般会有自己的数据源，业务部门提出相关需求后，数据工程师可直接根据需求从数据库中提取数据。但对于本身数据资源就贫乏或者数据隐私性较强的行业，例如医疗行业，一般很难获得大量数据，并且医疗数据的标注也比较专业，因此高质量的标注数据尤为难得。对于这种情况，我们可以先获取一些公开数据集或者竞赛数据集进行算法开发。

还有一种情况是目标数据在网页端，例如我们想了解杭州二手房的价格信息，找出影响杭州二手房价格的关键因素，这时候可能需要使用爬虫一类的数据采集技术来获取相关数据。

（1）数据来源：

- 用户访问行为数据。
- 用户业务数据。
- 外部第三方数据（网络爬虫等）。

（2）数据存储：

- 需要存储的数据：原始数据、预处理后的数据、模型结果。
- 存储设施：MySQL、HDFS、HBase、Solr、Elasticsearch、Kafka、Redis等。

（3）数据收集方式：

- Flume。
- Kafka。

（4）机器学习可用的公开数据集：

在实际工作中，我们可以使用业务数据进行机器学习开发，但是在学习过程中没有业务数据，此时可以使用公开数据集进行开发。常用的公开数据集网址如下：

- http://archive.ics.uci.edu/ml/datasets.html
- https://aws.amazon.com/cn/public-datasets/
- https://www.kaggle.com/competitions
- http://www.kdnuggets.com/datasets/index.html
- http://www.sogou.com/labs/resource/list_pingce.php
- https://tianchi.aliyun.com/datalab/index.htm
- http://www.pkbigdata.com/common/cmptIndex.html

### 3. 数据预处理

由于公开数据集和一些竞赛数据集非常“干净”，有的甚至可以直接用于模型训练，因此一些机器学习初学者认为只需专注于模型与算法设计就可以了，其实不然。在生产环境下，

我们拿到的数据都会比较“脏”，以至于需要花大量时间去清洗数据，有些人甚至认为数据清洗和特征工程要占用项目70%以上的时间。

- 数据预处理是实际生产环境中机器学习比较耗时的一部分。
- 大部分的机器学习模型所处理的都是特征，特征通常是输入变量所对应的可用于模型的数值表示。
- 大部分情况下，收集得到的数据需要经过预处理后才能够为算法所用。

数据预处理的操作主要包括以下几个部分：

- 数据过滤。
- 处理数据缺失。
- 处理可能的异常、错误或者异常值。
- 合并多个数据源数据。
- 数据汇总。

#### 4. 数据分析与可视化

数据清洗完后，一般不建议直接对数据进行训练。这时候我们对于要训练的数据还是非常陌生的。数据都有哪些特征？是否有很多类别特征？目标变量分布如何？各自变量与目标变量的关系是否需要可视化展示？数据中各变量缺失值的情况如何？怎样处理缺失值？这些问题都需要在探索性数据分析（Exploratory Data Analysis，EDA）和数据可视化过程中找到答案。

#### 5. 建模调优与特征工程

数据初步分析完后，对数据会有一个整体的认识，一般就可以着手训练机器学习模型了。但建模通常不是一锤子买卖，训练完一个基线（Baseline）模型之后，需要花大量时间进行模型调参和优化。

有关模型评价指标的相关内容参见下一小节。

#### 6. 模型结果展示与分析报告

经过一定的特征工程和模型调优之后，一般会有一个阶段性的最优模型结果，模型对应的关键性能指标都会达到最优状态。这时候需要通过一定的方式呈现模型，并对模型的业务含义进行解释。如果需要给上级领导和业务部门作决策参考，一般还需要生成一份有价值的分析报告。

#### 7. 模型部署与上线反馈优化

给出一份分析报告不是一个机器学习项目的最终目的，将模型部署到生产环境并能切实产生收益，才是机器学习的最终价值所在。

如果新上线的推荐算法能让用户的广告点击率上升 0.5%，为企业带来的收益也是巨大的。该阶段更多的是需要进行工程方面的一些考虑，是以 Web 接口的形式提供给开发部门，还是以脚本的形式嵌入软件中，后续如何收集反馈并提供产品迭代参考，这些都是需要在模型部署和上线之后考虑的。

### 1.1.4　机器学习模型评价指标

关于模型调优，结合业务的精细化特征工程工作比模型调参更能改善模型表现。建模调优与特征工程之间本身是个交互性的过程，在实际工作中我们可以一边进行调参，一边进行特征设计，交替进行，相互促进，共同改善模型表现。在调优过程中同时包括对模型的评估，即调优的模型是否更优秀。评估的指标主要有准确率、召回率、精确率、F1值、混淆矩阵、ROC曲线、ROC曲线下的面积等，如表1-1所示。

表1-1　评估指标

| 指　　标 | 描　　述 | sklearn函数 |
|---|---|---|
| Accuracy | 准确率 | from sklearn metrics import accuracy_score |
| Precision | 精确率 | from sklearn metrics import precision_score |
| Recall | 召回率 | from sklearn metrics import recall_score |
| F1 | F1值 | from sklearn metrics import f1_score |
| Confusion Matrix | 混淆矩阵 | from sklearn metrics import confusion_matrix |
| ROC | ROC曲线 | from sklearn metrics import roc |
| AUC | ROC曲线下的面积 | from sklearn metrics import auc |

#### 1. 准确率

准确率是指分类正确的样本占总样本个数的比例，计算公式为：

准确率=提取出的正确样本数/总样本数

准确率具有局限性。准确率是分类问题中最简单也是最直观的评价指标，但存在明显的缺陷，当不同种类的样本比例非常不均衡时，占比大的类别往往成为影响准确率的最主要因素。例如：当负样本占99%时，即使分类器把所有样本都预测为负样本，也可以得到99%的准确率，换句话说，总体准确率高，并不代表类别比例小的准确率也高。

#### 2. 召回率与精确率

召回率是指正确分类的正样本个数占真正的正样本数的比例。精确率是指正确分类的正样本个数占分类器判定为正样本的样本个数的比例。

召回率=正确的正例样本数/样本中的正例样本数
精确率=正确的正例样本数/预测为正例的样本数

召回率和精确率是既矛盾又统一的两个指标，为了提高精确率的值，分类器需要尽量在“更有把握”时才把样本预测为正样本，但此时往往会因为过于保守而漏掉很多“没有把握”的正样本，导致召回率值降低。

#### 3. F值

F值是精确率和召回率的谐波平均值，正常的平均值会平等对待所有的值，而谐波平均值会给予较低的值更高的权重。因此，只有当召回率和精确率都很高时，分类器才能得到较高的F值。

F值=精确率×召回率×2/（精确率＋召回率）

这个公式即表示F值为精确率和召回率的调和平均值。F值对那些具有相近的精确率和召回率的分类器更为有利。但这并不一定能符合我们的期望，在某些情况下，我们更关心的是精确率，而另一些情况下，我们可能真正关心的是召回率。精确率与召回率的权衡将是很值得思考的问题。

真实值与预测值的关系如表1-2所示。

表1-2　真实值与预测值

| 真 实 值 | 预　测　值 | |
|---|---|---|
| | 正　　例 | 负　　例 |
| 正例 | 真正例（A） | 假负例（B） |
| 负例 | 假正例（C） | 真负例（D） |

表中A和D预测正确，B和C预测错误，测试计算结果为（其中#表示样本个数，如#（A）表示真正例个数）：

$$\text{Accuracy}=\frac{\#(A)+\#(D)}{\#(A)+\#B+\#C+\#(D)}$$

$$\text{Recall}=\frac{\#(A)}{\#(A)+\#(B)}\quad\text{Precision}=\frac{\#(A)}{\#(A)+\#(C)}\quad F=\frac{2\times\text{Recall}\times\text{Precision}}{\text{Recall}+\text{Precision}}$$

### 4. ROC曲线

二值分类器是机器学习领域中最常见也是应用最广泛的分类器。评价二值分类器的指标很多，例如精确率、召回率、F1值、P-R曲线等，但这些指标或多或少只能反映模型在某一方面的性能。相比而言，ROC曲线则有很多优点，经常作为评估二值分类器最重要的指标之一。ROC曲线是Receiver Operating Characteristic Curve的简称，中文名为受试者工作特征曲线。

ROC曲线的横坐标为假阳率FPR，纵坐标为真阳率TPR，FPR和TPR的计算方法分别为：

$$\text{FPR}=\frac{FP}{N}\qquad\text{TPR}=\frac{TP}{P}$$

$P$是真实的正样本数量，$N$是真实的负样本数量，$TP$是$P$个正样本中被分类器预测为正样本的个数，$FP$为$N$个负样本中被预测为正样本的个数。

下面来演示ROC曲线的绘制。首先，创建数据集：

【程序 1.1】roc_data.py

```
import pandas as pd
column_name = ['真实标签','模型输出概率']
datasets = [['p',0.9],['p',0.8],['n',0.7],['p',0.6],
    ['p',0.55],['p',0.54],['n',0.53],['n',0.52],
    ['p',0.51],['n',0.505],['p',0.4],['p',0.39],
    ['p',0.38],['n',0.37],['n',0.36],['n',0.35],
    ['p',0.34],['n',0.33],['p',0.30],['n',0.1]]
```

```
    data = pd.DataFrame(datasets,index = [i for i in range(1,21,1)],
                        columns=column_name)
    print(data)
```

然后，绘制ROC曲线：

【程序 1.2】roc_plt.py

```
    import matplotlib.pyplot as plt
    # 计算各种概率情况下对应的(假阳率，真阳率)
    points = {0.1:[1,1],0.3:[0.9,1],0.33:[0.9,0.9],0.34:[0.8,0.9], 0.35:[0.8,0.8],
0.36:[0.7,0.8],0.37:[0.6,0.8],0.38:[0.5,0.8], 0.39:[0.5,0.7],0.40:[0.4,0.7],
0.505:[0.4,0.6],0.51:[0.3,0.6], 0.52:[0.3,0.5], 0.53:[0.2,0.5],0.54:[0.1,0.5],
0.55:[0.1,0.4], 0.6:[0.1,0.3], 0.7:[0.1,0.2],0.8:[0,0.2],0.9:[0,0.1]}
    X = []
    Y = []
    for value in points.values():
        X.append(value[0])
        Y.append(value[1])
    plt.scatter(X,Y,c = 'r',marker = 'o')
    plt.plot(X,Y)
    plt.xlim(0,1)
    plt.ylim(0,1)
    plt.xlabel('FPR')
    plt.ylabel('TPR')
    plt.show()
```

代码运行结果如图1-3所示。

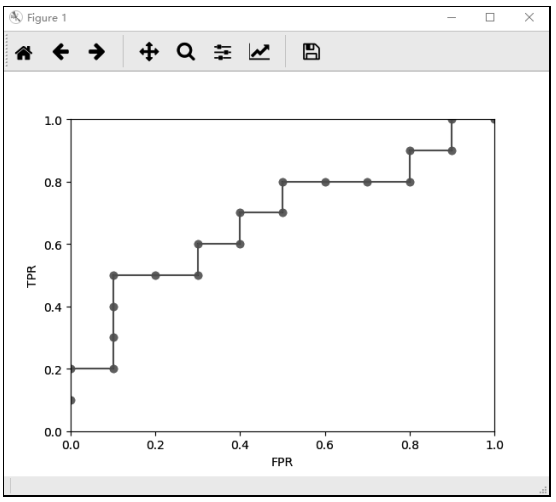



图 1-3　ROC 曲线

AUC指ROC曲线下的面积大小，该值能够量化地反映基于ROC曲线衡量出的模型性能。AUC越大，说明分类器越可能把真正的正样本排在前面，分类性能越好。

ROC曲线相比P-R曲线来说，当正负样本的分布发生变化时，ROC曲线的形状能够基本保持不变，而P-R曲线的形状一般会发生激烈的变化。这个特点让ROC曲线能够尽量降低不同测试集带来的干扰，更加客观地衡量模型本身的性能。

### 1.1.5　机器学习项目开发步骤

假设我们有个机器学习任务，是通过酒精度和颜色来区分红酒和啤酒。下面以机器学习如何区分啤酒和红酒为例（见图1-4），详细介绍一下机器学习中每一个步骤是如何工作的。


图 1-4　区分红酒和啤酒案例

#### 1. 数据收集与存储

我们先在超市买来一堆不同种类的啤酒和红酒，然后买来测量颜色的光谱仪和用于测量酒精度的设备，最后把买来的所有酒都标记出相应的颜色和酒精度，会形成下面这张表格（见表1-3）。

表1-3　选取数据特征

| 颜　色 | 酒　精　度 | 种　类 |
|---|---|---|
| 610 | 5 | 啤酒 |
| 599 | 13 | 红酒 |
| 693 | 14 | 红酒 |
| … | … | … |

这一步非常重要，因为数据的数量和质量直接决定了预测模型的好坏。

#### 2. 数据预处理

在本例中，数据是很工整的，但是在实际情况中，我们收集到的数据会有很多问题，所以会涉及数据清洗等数据预处理工作，如图1-5所示。

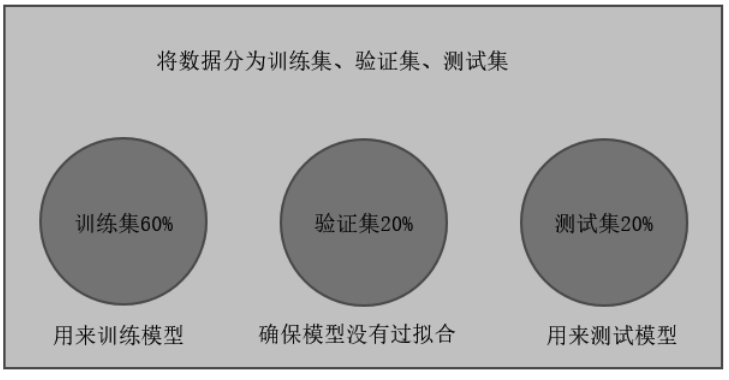



图 1-5　数据预处理

当数据本身没有什么问题后，我们将数据分成3个部分：训练集（60%）、验证集（20%）、测试集（20%），用于后面的验证和评估工作。

#### 3. 选择一个模型

研究人员和数据科学家创造了许多模型。我们可以根据不同的数据特征选择不同的模型，有些模型非常适合图像数据，有些非常适合序列（如文本或音乐），有些适合数字数据，有些适合文本数据。

在本例中，由于只有两个特征——颜色和酒精度，因此我们可以使用一个小的线性模型，这是一个相当简单的模型。

#### 4. 训练

大部分人都认为训练这一步是最重要的部分，其实并非如此，数据的数量、质量，以及模型的选择比训练本身更重要。将原始数据分为训练集和测试集（交叉验证），并利用训练集训练模型，这个过程不需要人来参与，机器可以独立完成，整个过程就像做算术题。因为机器学习的本质就是一个将现实问题转换为数学问题，然后解答数学题的过程。

#### 5. 模型评估

一旦训练完成，就可以评估模型是否有用。这是我们之前预留的验证集和测试集发挥作用的地方。这个过程可以让我们看到模型是如何预测的，即模型在现实世界中是如何表现的。

#### 6. 参数调整

完成模型评估后，我们可能希望了解是否可以使用任意方式进一步改进训练，这些可以通过调整参数来做到。当模型进行训练时，我们隐含地假设了一些参数，可以通过人为调整这些参数来让模型表现得更出色。

#### 7. 预测

前面的6个步骤都是围绕预测来服务的，这也是机器学习的价值。在这一步，当我们买来一瓶新的酒，只要告诉机器酒的颜色和酒精度，模型就会告诉我们这瓶酒是啤酒还是红酒了。

## 1.2　机器学习的发展史和分类

本节简单介绍一下机器学习的发展史及其分类。

### 1.2.1　机器学习的发展史

虽然人工智能并不是最近几年才兴起的，但是它一直作为科幻元素出现在大众视野中。自2016年AlphaGo战胜李世石之后，人工智能突然间成了公众热议的话题，仿佛人类已经创造出了超越人类智慧的机器。人工智能的核心技术——机器学习及其子领域深度学习迅速成为瞩目的焦点。面对这个似乎从天而降的新现象，乐观者有之，悲观者亦有之。

追溯历史，我们会发现机器学习的技术爆发有其历史必然性，属于技术发展的必然产物。而理清机器学习的发展脉络，有助于我们整体把握机器学习或者人工智能的技术框架，有助于从“道”的层面来理解这一技术领域。本节就先从三大究极哲学问题中的后两个——“从哪来”“到哪去”入手，整体把握机器学习，而后从“术”的角度深入学习，解决“是谁”的问题。

机器学习的发展并不是一帆风顺的，它的起源可以追溯到赫布理论的诞生。赫布于1949年基于神经心理提出了一种学习方法，该方法被称为赫布学习理论。大致可描述为：假设反射活动的持续性或反复性会导致细胞的持续性变化并增加其稳定性，当一个神经元A能持续或反复激发神经元B时，其中一个或两个神经元的生长或代谢过程都会变化。

从人工神经元或人工神经网络角度来看，赫布学习理论简单地解释了循环神经网络（Recurrent Neural Network，RNN）中节点之间的相关性关系（权重），即如果两个节点同时发生变化（无论是Positive还是Negative），那么节点之间有很强的正相关性（Positive Weight）；如果两者变化相反，那么说明有负相关性（Negative Weight）。

赫布学习理论在20世纪70年代曾陷入了瓶颈期，而后大数据时代开始，机器学习也在大数据的支持下复兴，因此我们可以大致将它的理念和运作模式按大数据时代前后分为浅层学习和深度学习。浅层学习起源20世纪70年代人工神经网络的反向传播（Back Propagation，BP）算法的发明，使得基于统计的机器学习算法大行其道，虽然这时候的人工神经网络算法也被称为多层感知机，但由于多层网络训练困难，因此通常都是只有一层隐含层的浅层模型。而深度学习则是多层神经网络的感知机，它的实质便是通过海量的数据进行更有效的训练，从而获得更精确的分类或预测。

### 1. 小数据时代（浅层学习）

1949年，Donald Hebb提出的赫布理论解释了人类学习过程中大脑神经元所发生的变化。赫布理论的诞生标志着机器学习领域迈出了第一步。1950年，阿兰·图灵创造了图灵测试（见图1-6）来判定计算机是否智能。图灵测试认为，如果一台机器能够与人类展开对话（通过电传设备）而不能被辨别出其机器身份，那么这台机器具有智能。这一简化使得图灵能够令人信服地说明“思考的机器”是可能的。

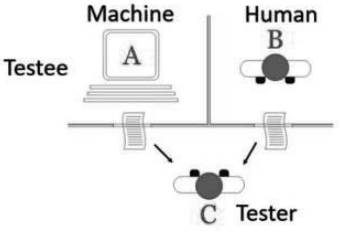


图 1-6　图灵测试

2014年6月8日，一个叫作尤金·古斯特曼的聊天机器人成功地让人类相信它是一个13岁的男孩，成为有史以来首台通过图灵测试的计算机。这被认为是人工智能发展的一个里程碑事件。

1952年，IBM科学家亚瑟·塞缪尔（见图1-7）开发了一个跳棋程序。该程序能够观察当前位置，并学习一个隐含的模型，从而为后续动作提供更好的指导。塞缪尔发现，伴随着程序运行时间的增加，程序可以实现越来越好的后续指导。通过这个程序，塞缪尔驳倒了普罗维登斯提出的机器无法超越人类，无法像人类一样写代码和学习的理论。塞缪尔创造了“机器学习”这一术语，并将它定义为：可以提供计算机能力而无须显式编程的研究领域。同时，IBM首次定义并解释了“机器学习”，将它非正式定义为“在不直接针对问题进行编程的情况下，赋予计算机学习能力的一个研究领域”。

图 1-7　塞缪尔工作照

1957年，罗森·布拉特基于神经感知科学背景提出了第二模型，非常类似于今天的机器学习模型。这在当时是一个非常令人兴奋的发现，它比赫布的想法更适用。基于这个模型，罗森·布拉特设计出了第一个计算机神经网络——感知机（Perceptron），它模拟了人脑的运作方式。

罗森·布拉特对感知机的定义如下：感知机旨在说明一般智能系统的一些基本属性，它不会被个别特例或通常不知道的东西束缚，也不会因为那些个别生物有机体的情况而陷入混乱。

1960年，威德罗提出了Delta学习规则，也就是差量学习规则，即如今的最小二乘问题。这种学习规则随即被应用到感知机模型中，创建出更精确的线性分类器。随后机器学习的发展出现了瓶颈。

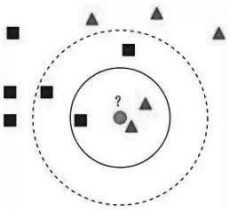


图 1-8　KNN 算法

1967年，K邻近算法（K-Nearest Neighbor Algorithm，KNN）出现，使计算机可以进行简单的模式识别。KNN算法的核心思想是，如果一个样本在特征空间中与其$k$个最相邻的样本中的大多数都属于某一个类别，则该样本也属于这个类别，并具有该类别样本的特性，如图1-8所示。这就是所谓的“少数服从多数”原则。

1969年，明斯基提出了异域问题，指出了感知机的本质缺陷——面对线性不可分问题时无力，即当空间内的点无法被直线分类时，感知机便会束手无措。直到20世纪80年代末此算法才开始被接纳使用，并给机器学习带来了希望。人们发现，神经网络反向传播算法可以帮助机器从大量数据统计中整理规律，从而面对未知的事件做出推测。这时候的感知机只是一种含有一层隐藏层节点的浅层模型，这个时代的浅层学习也因此而得名。到了20世纪90年代，浅层学习进入黄金时代，各种各样的浅层学习模型被相继提出，这些模型大多数在实际运用中都取得了巨大的成功。

### 2. 大数据时代（深度学习）

随着人类对数据信息的收集和应用逐渐娴熟，对数据的掌控力逐渐提升，伟博斯在1981年的神经网络反向传播算法中具体提出了多层感知机模型。虽然BP算法早在1970年就已经以“自动微分的反向模型（Reverse Mode of Automatic Differentiation）”为名提出来了，但直到此时才真正发挥效用，并且直到今天，BP算法仍然是神经网络架构的关键因素。有了这些新思想，神经网络的研究又加快了。多层感知机如图1-9所示。

1985－1986年，机器学习在海量数据的支持下攀上了新的高峰，神经网络研究人员鲁梅尔哈特、辛顿、威廉姆斯、尼尔森相继提出了使用BP算法训练的多参数线性规划（MLP）的理念，成为后来深度学习的基石。

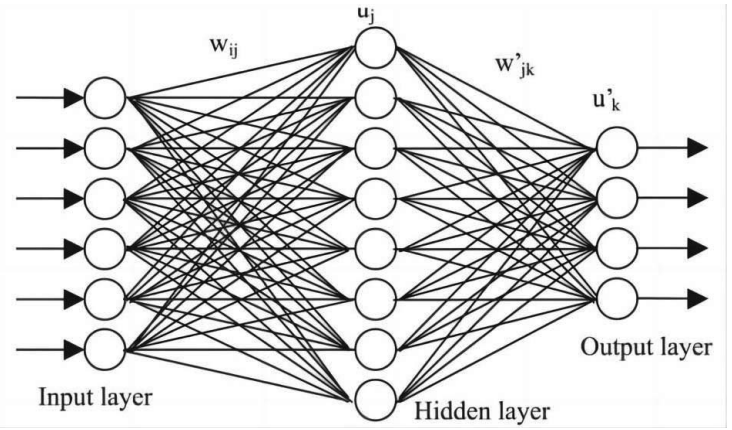



图 1-9　多层感知机

在另一个谱系中，昆兰于1986年提出了一种非常出名的机器学习算法，我们称之为决策树（见图1-10），更具体地说是ID3算法。这是另一个主流机器学习算法的突破点。此外，ID3算法也被发布成为一款软件，它能以简单的规划和明确的推论找到更多的现实案例，而这一特性正好和神经网络黑箱模型相反。

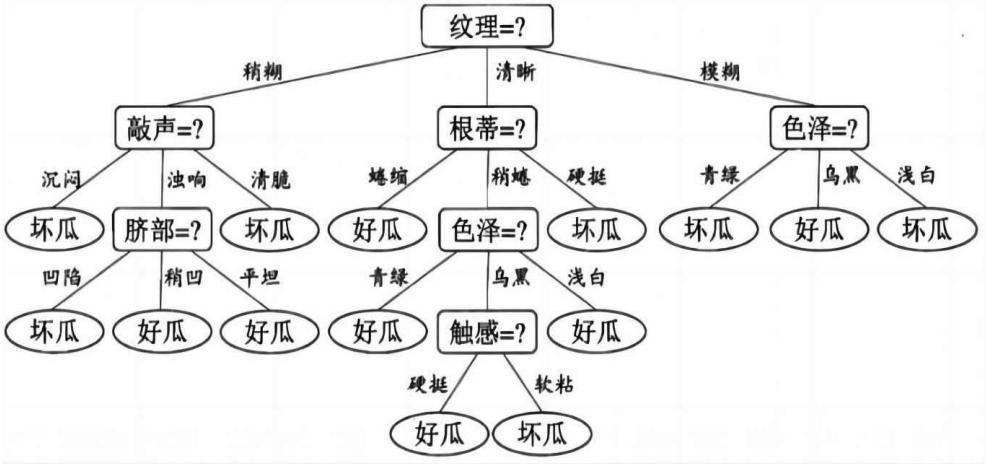



图 1-10　决策树

决策树是一个预测模型，它代表的是对象属性与对象值之间的一种映射关系。树中每个节点表示某个对象，而每个分叉路径则代表某个可能的属性值，每个叶节点则对应从根节点到该叶节点所经历的路径所表示的对象的值。

决策树仅有单一输出，若欲有复数输出，则可以建立独立的决策树以处理不同输出。在数据挖掘中，决策树是一种经常被使用的技术，可以用于分析数据，同样也可以用来进行预测。

在ID3算法提出来以后，研究社区已经探索了许多不同的选择或改进（如ID4、回归树、CART算法等），这些算法仍然活跃在机器学习领域中。

1990年，Schapire最先构造出一种多项式级的算法，这就是最初的Boosting算法。一年后，Freund提出一种效率更高的Boosting算法。但是，这两种算法存在共同的实践上的缺陷，那就是都要求事先知道弱学习算法学习正确的下限。

1995年，Freund和Schapire改进了Boosting算法，提出了AdaBoost（Adaptive Boosting）算法，该算法效率和Freund于1991年提出的Boosting算法几乎相同，但它不需要任何关于弱学习器的先验知识，因而更容易应用到实际问题当中。

Boosting方法是一种用来提高弱分类算法准确度的方法，这种方法先构造一个预测函数系列，然后以一定的方式将它们组合成一个预测函数。它是一种框架算法，主要通过对样本集的操作获得样本子集，然后用弱分类算法在样本子集上训练生成一系列的基分类器。

支持向量机（Support Vector Machine，SVM）的出现是机器学习领域的另一个重要突破，该算法具有非常强大的理论地位和实证结果。自SVM出现之后，机器学习研究也分为神经网络（Neural Network，NN）和支持向量机两派。

然而，自2000年左右提出了带核函数的支持向量机后，SVM在许多以前由NN为主的任务中获得了更好的效果。此外，相对于NN来说，SVM还能利用所有关于凸优化、泛化边际理论和核函数的深厚知识。因此，SVM可以从不同的学科中大力推动理论和实践的改进。

此时，神经网络又遭受到一个质疑。1991年Hochreiter等人和2001年Hochreiter等人的研究表明，在应用BP算法学习时，NN神经元饱和后会出现梯度损失（Gradient Loss）的情况。简单地说，在一定数量的epochs训练后，NN会产生过拟合现象。因此，这一时期与SVM相比，NN处于劣势地位。

决策树模型由布雷曼博士在2001年提出，它是一种通过集成学习的思想将多棵树集成的算法，它的基本单元是决策树，而它的本质属于机器学习的一大分支——集成学习（Ensemble Learning）方法。随机森林的名称中有两个关键词，一个是“随机”，另一个是“森林”。“森林”很好理解，一棵叫作树，那么成百上千棵就可以叫作森林了，这样的比喻还是很贴切的，其实这也是随机森林的主要思想——集成思想的体现。

从直观角度来解释，每棵决策树都是一个分类器（假设现在针对的是分类问题），那么对于一个输入样本，$N$棵树会有$N$个分类结果。而随机森林集成了所有的分类投票结果，并将投票次数最多的类别指定为最终的输出，这就是一种最简单的集成学习思想。

2006年，神经网络研究领域的领军者Hinton提出了神经网络Deep Learning算法，使神经网络的能力大大提高，并向支持向量机发出了挑战。同年，Hinton和他的学生Salakhutdinov在顶尖学术刊物*Science*上发表了一篇文章，开启了深度学习在学术界和工业界的浪潮。

这篇文章有两个主要的信息：①有很多隐藏层的人工神经网络具有优异的特征学习能力，学习到的特征对数据有更本质的刻画，从而有利于可视化或分类；②深度神经网络在训练上的难度可以通过“逐层初始化”（Layer-wise Pre-training）来有效克服。在这篇文章中，逐层初始化是通过无监督学习（Unsupervised Learning）来实现的。

2015年，为纪念人工智能概念提出60周年，LeCun、Bengio和Hinton推出了深度学习综述。

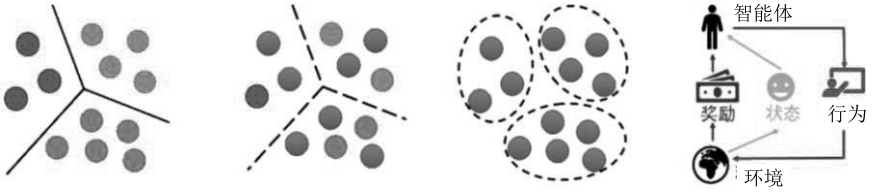
深度学习可以让那些拥有多个处理层的计算模型来学习具有多层次抽象的数据的表示。这些方法为人工智能的许多方面都带来了显著的改善，包括最先进的语音识别、视觉对象识别、对象检测和许多其他领域，例如药物发现等。深度学习能够发现大数据中的复杂结构。它利用BP算法来完成这个发现过程。BP算法能够指导机器如何从前一层获取误差，从而改变本层的内部参数，这些内部参数可以用于计算表示。深度卷积网络为图像、视频、语音和音频处理方面带来了突破，而递归网络在处理序列数据（例如文本和语音方面）时表现出闪亮的一面。

当前，统计学习领域热门的方法主要有Deep Learning和SVM，它们是统计学习的代表方法。可以认为神经网络与支持向量机都源自于感知机，但它们一直处于“竞争”关系。SVM应用核函数的展开定理，无须知道非线性映射的显式表达式；由于是在高维特征空间中建立线性学习机，因此与线性模型相比，不仅几乎不增加计算的复杂性，而且在某种程度上避免了“维数灾难”。而早先的神经网络算法比较容易过训练，需要设置大量的经验参数；训练速度比较慢，在层次比较少（小于或等于3）的情况下效果并不比其他方法更优。

神经网络模型似乎能够实现更加艰难的任务，如目标识别、语音识别、自然语言处理等。但是，这绝对不意味着其他机器学习方法已终结。尽管深度学习的成功案例迅速增长，但是对这些模型的训练成本是相当高的，调整外部参数也很麻烦。同时，SVM的简单性促使其仍然是使用最为广泛的机器学习方式。

### 1.2.2　机器学习分类

根据算法类型，机器学习可以分为4类，即监督学习（Supervised Learning）、无监督学习（Unsupervised Learning）、半监督学习（Semi-Supervised Learning）和强化学习（Reinforcement Learning）。机器学习分类示意图如图1-11所示，图中用灰色圆点代表没有标签的数据，其他颜色的圆点代表不同类别的有标签数据。




监督学习（有标签）　半监督学习（部分标签）　无监督学习（有标签）　强化学习（评级标签）

图 1-11　机器学习分类

#### 1. 监督学习

监督学习使用标记过的数据进行训练。所谓标记过的数据，指的是包含已知输入和输出的原始数据。其中输入数据中的每个变量都称为一个特征（Feature）值，而输出数据则是针对这些输入数据的对应输出的期望值，也称标签值。在监督学习中，计算机使用输入数据计算输出值，然后对比标签值计算误差，最后通过迭代寻找最佳模型参数。监督学习通常用于基于历史数据的未来事件预测，主要解决两类问题，即回归（Regression）和分类（Classification）。例如在天气预报中使用历史数据预测未来几天的温度、湿度和降雨量等就是典型的回归问题，其输出的数据是连续的。而分类问题的输出是不连续的离散值，例如，使用历史数据判断航班是否晚点就是一种二元分类问题，其输出值只有“是”和“非”两种。在实际情况中，有些场景既可以看作回归问题，也可以看作分类问题，例如在天气预报中将利用回归计算得到的温度值转换为“炎热”和“凉爽”的分类问题。

简单来说，监督学习是指我们给算法一个数据集，并且给定正确答案，机器通过数据集来学习正确答案的计算方法。

举例来说，我们准备了一大堆猫和狗的照片，想让机器学会识别猫和狗。当使用监督学习的时候，我们需要给这些照片打上标签，如图1-12所示。

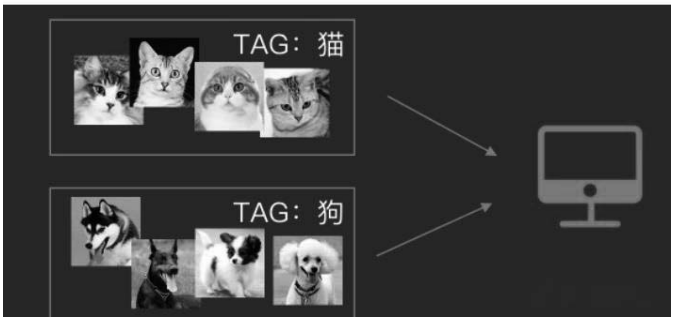



图 1-12　给猫和狗的照片打标签

给照片打的标签就是“正确答案”，机器通过大量学习，就可以学会如何识别出猫和狗，如图1-13所示。

图 1-13　识别猫和狗的机器学习过程

这种通过大量人工打标签来帮助机器学习的方式就是监督学习。这种学习方式效果非常好，但是成本也非常高。

常用的监督学习算法包括K邻近算法（K-Nearest Neighbors，KNN）、线性回归（Linear Regression）、逻辑回归（Logistic Regression）、支持向量机（Support Vector Machine，SVM）、朴素贝叶斯（Naive Bayes）、决策树（Decision Tree）、随机森林（Random Forest）、神经网络（Neural Network）和卷积神经网络（Convolutional Neural Networks，CNN）等。

### 2. 半监督学习

半监督学习与监督学习的应用场景相同，主要面向分类和回归。但半监督学习使用的原始数据只有一部分有标签，因为无标签数据的获取成本更低。在实际场景中，用户会倾向于使用少量的标签数据与大量的无标签数据进行训练。例如，在图像识别领域，先在大量含有特定物体的原始图像中挑选部分图像进行手工标注，然后就可以使用半监督学习对数据集进行训练，从而得到能够从图像中准确识别物体的模型。

常用的半监督学习算法包括协同训练（Co-Training）和转导支持向量机（Transductive Support Vector Machine，TSVM）等。

### 3. 无监督学习

与监督学习不同，无监督学习所使用的原始数据的输出部分没有标签，也就是说，在训练的时候并不知道期望的输出是什么。所以，无监督学习并不像监督学习那样预测输出结果，而

是解决输入数据的聚类（Clustering）和特征关联（Correlation）问题，目标是通过训练来发现输入数据中存在的共性特征，或者发现特征值之间的关联关系。其中，聚类算法根据对象属性进行分组。

简单来说，无监督学习中，给定的数据集没有“正确答案”，所有的数据都是一样的，无监督学习的任务是从给定的数据集中挖掘出潜在的结构。

举个例子，我们把一堆猫和狗的照片给机器，不给这些照片打任何标签，但是我们希望机器能够将这些照片分类，如图1-14所示。

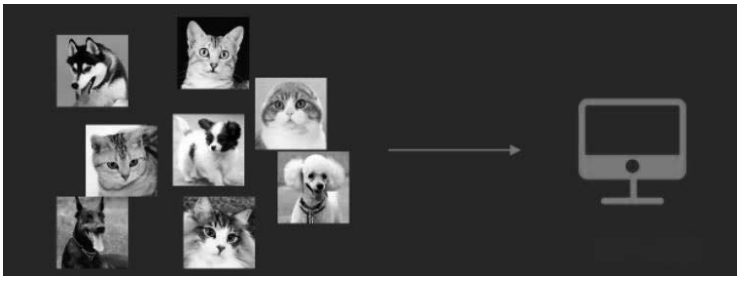


图 1-14　数据输入机器

通过学习，机器会把这些照片分为两类，一类都是猫的照片，另一类都是狗的照片，如图1-15所示。虽然跟监督学习的结果看上去差不多，但二者有着本质的差别：无监督学习中，虽然照片分为了猫和狗，但是机器并不知道哪个是猫，哪个是狗。对于机器来说，相当于分成了A、B两类。

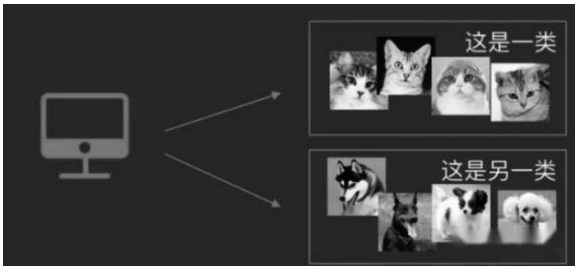



图 1-15　机器分类

常用的无监督学习算法包括K均值聚类（K-Means Clustering）、主成分分析（Principal Component Analysis，PCA）算法、自组织映射（Self-Organizing Map，SOM）神经网络和受限玻尔兹曼机（Restricted Boltzmann Machine，RBM）等。

#### 4. 强化学习

强化学习主要由智能体（Agent）、环境（Environment）、状态（State）、动作（Action）、奖励（Reward）组成。智能体执行了某个动作后，环境将会转换到一个新的状态，并对该新的状态给出奖励信号（正奖励或者负奖励），随后，智能体根据新的状态和环境反馈的奖励，按照一定的策略执行新的动作。上述过程为智能体和环境通过状态、动作、奖励进行交互的方式。

智能体通过强化学习，可以知道自己在什么状态下，应该采取什么样的动作，使得自身获得最大奖励。由于智能体与环境的交互方式与人类与环境的交互方式类似，因此可以认为强化

学习是一套通用的学习框架，可以用来解决通用人工智能的问题。由此，强化学习也被称为通用人工智能的机器学习方法。

强化学习面向决策链问题，在不断变化的状态下，强化学习的目的是确定当前状态下的最佳决策。因为当前的决策往往无法立刻被验证和评估，所以强化学习往往没有大量的原始数据，计算机需要进行大量的试错学习，基于错误发现哪些行动能产生最大的回报，再根据规则找到生成最佳结果的最优路径。强化学习的目标是学习最好的策略，通常用于机器人、自动驾驶、游戏和棋类等，最典型的场景就是打游戏。例如《王者荣耀》里面的那些人机，都是训练出来的，我们不同段位的玩家遇到的人机的能力也是有区别的。

## 1.3　机器学习常用术语

机器学习是一门专业性很强的技术，它大量地应用了数学、统计学上的知识，因此总会有一些陌生的词汇，这些词汇就像“拦路虎”一样阻碍着我们前进，甚至把我们吓跑。本节就来介绍机器学习中常用的术语，为后续的学习打下坚实的基础。

### 1. 模型

模型这一词语将会贯穿整个教程的始末，它是机器学习中的核心概念。可以把它看作一个“魔法盒”，我们向它许愿（输入数据），它就会帮我们实现愿望（输出预测结果）。整个机器学习的过程都将围绕模型展开，训练出一个最优质的“魔法盒”，它可以尽量精准地实现我们许的“愿望”，这就是机器学习的目标。

### 2. 数据集

数据集，从字面意思很容易理解，它表示一个承载数据的集合。如果说“模型”是“魔法盒”，那么数据集就是负责给它充电的“能量电池”。简单地说，如果缺少了数据集，那么模型就没有存在的意义了。数据集可划分为“训练集”和“测试集”，它们分别在机器学习的“训练阶段”和“预测输出阶段”起着重要的作用。

### 3. 样本与特征

样本指的是数据集中的数据，一条数据被称为“一个样本”。通常情况下，样本会包含多个特征值，用来描述数据，例如现在有一组描述人体形态的数据“180 70 25”，如果单看数据我们会感到茫然，但是用“特征”描述后就会变得容易理解，如表1-4所示。

表1-4　样本与特征

| 身高（cm） | 体重（kg） | 年龄 |
| --- | --- | --- |
| 180 | 70 | 25 |

由上表可知，数据集的构成是“一行一样本，一列一特征”。特征值也可以理解为数据的相关性，每一列的数据都与这一列的特征值相关。

### 4. 向量

“向量”是一个常用的数学术语，也是机器学习的关键术语。向量在线性代数中有着严格的定义。向量也称欧几里得向量、几何向量、矢量，指具有大小和方向的量。可以形象地把它理解为带箭头的线段，箭头代表向量的方向，线段长度代表向量的大小。与向量对应的量叫作数量（物理学中称标量），数量只有大小，没有方向。

在机器学习中，模型算法的运算均基于线性代数运算法则，例如行列式、矩阵运算、线性方程等。这些运算法则学习起来其实并不难，它们都有一定的运算规则，只需套用即可，因此读者不必彷徨，可以参考向量运算法则。向量的计算可采NumPy库来实现，示例如下。

【程序 1.3】num_cal.py

```
import numpy as np
#构建向量数组
a=np.array([-1,2])
b=np.array([3,-1])
#加法
a_b=a+b
#数乘
a2=a*2
b3=b*(-3)
#减法
b_a=a-b
print(a_b,a2,b3,b_a)
```

简而言之，数据集中的每一个样本都是一条具有向量形式的数据。

### 5. 矩阵

矩阵也是一个常用的数学术语，可以把它看作由向量组成的二维数组。数据集就是以二维矩阵的形式存储数据的，可以把它形象地理解为电子表格，“一行一样本，一列一特征”，表现形式如表1-5所示。

表1-5　数据特征描述

| 样本序号 | A特征 | B特征 | C特征 | D特征 | E结果 |
|---|---|---|---|---|---|
| 1 | x1 | x2 | x3 | x4 | y1 |
| 2 | x1 | x2 | x3 | x4 | y2 |
| 3 | x1 | x2 | x3 | x4 | y3 |
| 4 | x1 | x2 | x3 | x4 | y4 |
| 5 | x1 | x2 | x3 | x4 | y5 |
| 6 | x1 | x2 | x3 | x4 | y6 |

如果用二维矩阵的表示的话，其格式如下：

$$\begin{bmatrix} & A & B & C & D & E \\ 1 & x1 & x2 & x3 & x4 & y1 \\ 2 & x1 & x2 & x3 & x4 & y2 \\ 3 & x1 & x2 & x3 & x4 & y3 \\ 4 & x1 & x2 & x3 & x4 & y4 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

### 6. 假设函数与损失函数

机器学习在构建模型的过程中会应用大量的数学函数，正因为如此，很多初学者对它产生畏惧，那么它们真的有这么可怕吗？其实笔者认为至少没有想象中的那么可怕。从编程角度来看，这些函数就相当于模块中内置好的方法，只需要调用相应的方法，就可以达成想要的目的。而要说难点，首先就是要理解应用场景，然后根据实际的场景去调用相应的方法，这才是我们更应该关注的问题。

假设函数（Hypothesis Function）和损失函数（Loss Function）是机器学习中的两个概念，它并非某个模块下的函数或方法，而是我们根据实际应用场景确定的一种函数形式，就像我们解决数学应用题一样，根据题意写出解决问题的方程组。下面分别来看一下它们的含义。

1）假设函数

假设函数可表述为$y=f(x)$，其中$x$表示输入数据，$y$表示输出的预测结果，而这个结果需要不断地优化才会达到预期的结果，否则会与实际值偏差较大。

2）损失函数

损失函数又叫目标函数，简写为$L(x)$。这个$L(x)$的值是假设函数得出的预测结果$y$，如果$L(x)$的返回值越大，就表示预测结果与实际偏差越大；如果$L(x)$的返回值越小，则证明预测值越来越“逼近”真实值，这才是机器学习最终的目的。因此，损失函数就像一个度量尺，让我们知道“假设函数”预测结果的优劣，从而做出相应的优化策略。

3）优化方法

“优化方法”可以理解为假设函数和损失函数之间的沟通桥梁。通过$L(x)$可以得知假设函数输出的预测结果与实际值的偏差值，当该值较大时，就需要做出相应的调整，这个调整的过程叫作“参数优化”。而如何实现优化呢？这也是机器学习过程中的难点。其实为了解决这一问题，数学家们早就给出了相应的解决方案，例如梯度下降、牛顿法与拟牛顿法、共轭梯度法等。因此，我们要做的就是理解并掌握“科学巨人”留下的理论、方法。

对于优化方法，我们要根据具体的应用场景来选择，因为每一种方法都有自己的优劣，只有合适的才是最好的。

上述函数的关系如图1-16所示。

### 7. 拟合、过拟合与欠拟合

拟合是机器学习中的重要概念，也可以说机器学习的研究对象就是让模型能更好地拟合数据。那么，到底如何理解“拟合”这个词呢？

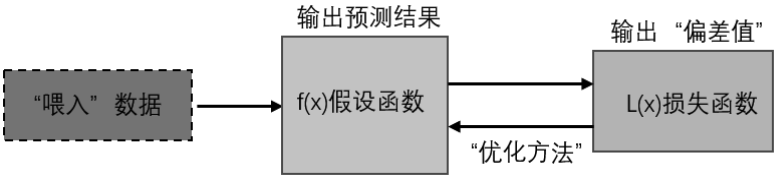



图 1-16　函数关系图

1）拟合

形象地说，“拟合”就是把平面坐标系中一系列散落的点，用一条光滑的曲线连接起来，因此拟合也被称为“曲线拟合”。拟合的曲线一般用函数来表示，但是由于拟合曲线会存在许多种连接方式，因此就会出现多种拟合函数。通过研究、比较确定一条最佳的“曲线”也是机器学习中一个重要的任务。如图1-17所示，展示了一条拟合曲线。


很多和数学相关的编程语言都内置了计算拟合曲线的函数，例如 MATLAB、Python SciPy等，在后续内容中还会介绍。

2）过拟合

过拟合是机器学习模型训练过程中经常遇到的问题。所谓过拟合，通俗来讲就是模型的泛化能力较差，也就是过拟合的模型在训练样本中表现优越，但是在验证数据以及测试数据集中表现不佳。

举一个简单的例子，例如训练一个识别狗狗照片的模型，如果只用金毛犬的照片训练，那么该模型就只吸纳了金毛犬的相关特征，此时让训练好的模型识别一条泰迪犬，那么结果可想而知，该模型会认为泰迪犬不是一条狗。过拟合曲线如图1-18所示。

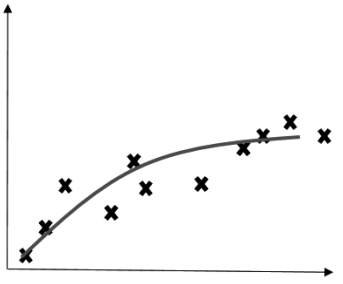


图 1-17　拟合曲线

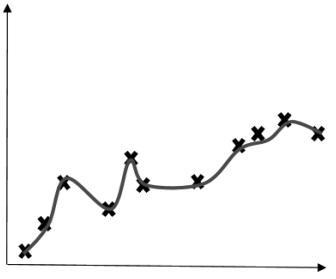


图 1-18　过拟合曲线

过拟合问题之所以在机器学习中经常遇道，主要是因为训练时样本过少、特征值过多导致的，本书后续还会详细介绍。

3）欠拟合

欠拟合（Underfitting）恰好与过拟合相反，它指的是“曲线”不能很好地“拟合”数据。在训练和测试阶段，欠拟合模型表现均较差，无法输出理想的预测结果。欠拟合曲线如图1-19所示。

造成欠拟合的主要原因是没有选择好合适的特征值，例如使用一次函数（$y=kx+b$）去拟合具有对数特征的散落点（$y=\log 2x$），示例如图1-20所示。

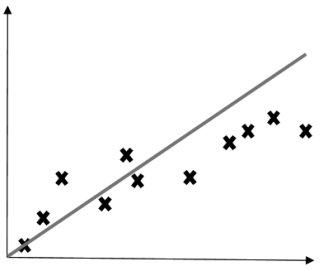


图 1-19　欠拟合曲线

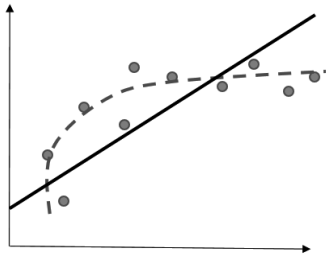


图 1-20 欠拟合示例

总之，过拟合可以理解为模型把非目标物识别成了目标物。我们想要识别的非目标物，它的一些特征没有那么明显，但因为模型被“喂”了过大的训练集或者训练轮数过多，所以模型把这些特征认为是目标物的特有特征。欠拟合是由于“喂”给模型的训练集过少、算法有缺陷等原因导致的。欠拟合可以理解为最终输出的模型不能完成我们希望模型完成的任务，也就是模型识别不出来我们想要识别的东西。

欠拟合和过拟合是机器学习中会遇到的问题，这两种情况都不是我们期望看到的，因此要避免。关于如何处理类似问题，我们在本书后续内容中还会陆续讲解，本节只需要熟悉并理解常见的机器学习术语和一些概念即可。

### 8. 激活函数（Activation Function）

激活函数（例如ReLU或Sigmoid）将前一层所有神经元激活值的加权和输入一个非线性函数中，然后向下一层传递该函数的输出值（典型的非线性）。

### 9. 反向传播（Backpropagation）

反向传播算法是神经网络中完成梯度下降的重要算法。首先，在前向传播的过程中计算每个节点的输出值；然后，在反向传播的过程中计算与每个参数对应的误差的偏导数。

### 10. 基线（Baseline）

基线是指用作比较参考的简单模型，它帮助模型开发者量化模型在特定问题上的预期表现。

### 11. 批量（Batch）

批量是指模型训练中一个迭代（指一次梯度更新）所使用的样本集。

### 12. 批量大小（Batch Size）

批量大小指一个批量中样本的数量。例如，SGD的批量大小为1，而mini-batch的批量大小通常为10~1000。批量大小通常在训练与推理的过程中确定，但是TensorFlow 框架不允许动态更改批量大小。

### 13. 二元分类器（Binary Classification）

二元分类器输出两个互斥（不相交）类别中的一个。例如，一个评估邮件信息并输出垃圾邮件或非垃圾邮件的机器学习模型就是一个二元分类器。

### 14. 标定层（Calibration Layer）

标定层是一种调整后期预测的结构，通常用于解释预测偏差。调整后的预期和概率必须匹配一个观察标签集的分布。

### 15. 候选采样（Candidate Sampling）

候选采样是一种在训练时进行的优化方法，使用Softmax等算法计算所有正标签的概率，同时只计算一些随机取样的负标签的概率。

### 16. 检查点（Checkpoint）

检查点指在特定时刻标记模型变量的状态的数据。检查点允许输出模型的权重，也允许通过多个阶段训练模型。检查点还允许跳过错误继续进行（例如，抢占作业）。注意，模型自身的图式并不包含于检查点内。

### 17. 类别（Class）

所有同类属性的目标值作为一个标签。

### 18. 类别不平衡数据集（Class-Imbalanced Data Set）

数据集样本类别极不平衡，一般针对二元分类问题，表示两个类别的标签的分布频率有很大的差异。

### 19. 分类模型（Classification）

机器学习模型的一种，将数据分离为两个或多个离散类别。例如，一个自然语言处理分类模型可以将一句话归类为法语、西班牙语或意大利语。分类模型与回归模型（Regression Model）成对比。

### 20. 分类阈值（Classification Threshold）

分类阈值指应用于模型的预测分数以分离正类别和负类别的一种标量值标准。当需要将逻辑回归的结果映射到二元分类模型中时，就需要使用分类阈值。

### 21. 混淆矩阵（Confusion Matrix）

混淆矩阵指总结分类模型的预测结果的表现水平（即标签和模型分类的匹配程度）的 $N \times N$ 维表格。混淆矩阵的一个轴列出模型预测的标签，另一个轴列出实际的标签。$N$表示类别的数量。

### 22. 连续特征（Continuous Feature）

连续特征拥有无限个取值点的浮点特征。和离散特征（Discrete Feature）相反。

### 23. 收敛（Convergence）

训练过程达到的某种状态，其中训练损失和验证损失在经过确定的迭代次数后，在每一次迭代中改变很小或完全不变。换句话说，当对当前数据继续训练而无法再提升模型的表现水平

的时候，就称模型已经收敛。在深度学习中，损失值在下降之前，有时候经过多次迭代仍保持常量或者接近常量，就会造成模型已经收敛的错觉。

24. 凸函数（Convex Function）

一种形状大致呈字母U形或碗形的函数。但是，在退化情形中，凸函数的形状就像一条线。

25. 交叉熵（Cross-Entropy）

多类别分类问题中对Log损失函数的推广。交叉熵量化两个概率分布之间的区别。参见困惑度（Perplexity）。

26. 数据集（Data Set）

样本的集合。

27. 决策边界（Decision Boundary）

在一个二元分类或多类别分类问题中，模型学习的类别之间的分离器。

28. 深度模型（Deep Model）

一种包含多个隐藏层的神经网络。深度模型依赖于其可训练的非线性性质。和宽度模型（Wide Model）对照。

29. 密集特征（Dense Feature）

大多数取值为非零的一种特征，通常用取浮点值的张量（Tensor）表示。和稀疏特征（Sparse Feature）相反。

30. Dropout正则化（Dropout Regularization）

训练神经网络时一种有用的正则化方法。Dropout正则化的过程是在单次梯度计算中删去一层网络中随机选取的固定数量的单元。删去的单元越多，正则化越强。

31. 动态模型（Dynamic Model）

动态模型是一种以连续更新的方式在线训练的模型，即数据连续不断地输入模型。

32. 早期停止法（Early Stopping）

一种正则化方法，在训练损失完成下降之前停止模型训练过程。当验证数据集（Validation Data Set）的损失开始上升的时候，即泛化表现变差的时候，就应该使用早期停止法。

33. 嵌入（Embeddings）

一类表示为连续值特征的明确的特征。嵌入通常指将高维向量转换到低维空间中。

34. 集成（Ensemble）

多个模型预测的综合考虑。

### 35. 评估器（Estimator）

评估器是一种封装了各种机器学习模型的工具，是拟合和训练数据的机器学习算法或者其他算法的抽象。

### 36. 样本（Example）

一个数据集的一行内容。一个样本包含了一个或多个特征，也可能是一个标签。参见标注样本（Labeled Example）和无标注样本（Unlabeled Example）。

### 37. 假负类（False Negative，FN）

被模型错误预测为负类的样本。例如，模型推断一封邮件为非垃圾邮件（负类），但实际上这封邮件是垃圾邮件。

### 38. 假正类（False Positive，FP）

被模型错误预测为正类的样本。例如，模型推断一封邮件为垃圾邮件（正类），但实际上这封邮件是非垃圾邮件。

### 39. 假正类率（False Positive Rate，FP率）

ROC曲线中的*x*轴。FP 率的计算公式是：假正率=假正类数/(假正类数+真负类数)。

### 40. 特征列（Feature Columns）

具有相关性的特征的集合，例如用户可能居住的所有国家的集合。一个样本的一个特征列中可能会有一个或者多个特征。

### 41. 特征集（Feature Set）

特征集指机器学习模型训练的时候使用的特征群。例如，邮政编码、面积要求和物业状况等，可以组成一个简单的特征集，使模型能预测房价。

### 42. 特征定义（Feature Spec）

特征指的是描述一个实例的属性或特征,也可以称为自变量（independent variable）或输入变量（input variable）。

### 43. 泛化（Generalization）

泛化是指模型利用新的没见过的数据而不是训练数据做出正确预测的能力。

### 44. 广义线性模型（Generalized Linear Model）

广义线性模型是线性模型的扩展，通过连接函数建立响应变量的数学期望值与线性组合的预测变量之间的关系。

### 45. 梯度（Gradient）

在机器学习中，梯度是模型函数的偏导数向量。梯度指向最陡峭的上升路线。

46. 梯度截断（Gradient Clipping）

在应用梯度之前先修饰数值，梯度截断有助于确保数值稳定性，防止梯度爆炸出现。

47. 梯度下降（Gradient Descent）

梯度下降通过计算模型的相关参数和损失函数的梯度来最小化损失，值取决于训练数据。梯度下降迭代地调整参量，逐渐靠近权重和偏置的最佳组合，从而最小化损失函数。

48. 图（Graph）

图是由节点（Node）和边（Edge）组成的一种数据结构,用于描述事物之间的关系。图近来正逐渐变成机器学习的一大核心领域，例如，可以通过图来预测潜在的连接，从而理解社交网络的结构、检测欺诈、理解汽车租赁服务的消费者行为，或者进行实时推荐。

## 1.4　本 章 小 结

本章从应用的角度介绍机器学习的基本概念、机器学习三要素和核心、机器学习开发流程、机器学习模型评价指标，以及机器学习的发展及分类。从了解机器学习到熟悉机器学习，再到精通机器学习，这是一个融会贯通的过程。机器学习已经在人工智能、自然语言处理、图像识别、医学、金融等领域得到广泛的应用，成为当今信息技术发展中的热点和趋势之一，这也使得机器学习成为各个领域和行业不可或缺的技术成分。从基础理论和应用的角度了解和学习机器学习，有利于学习者更快进入这个领域当中。