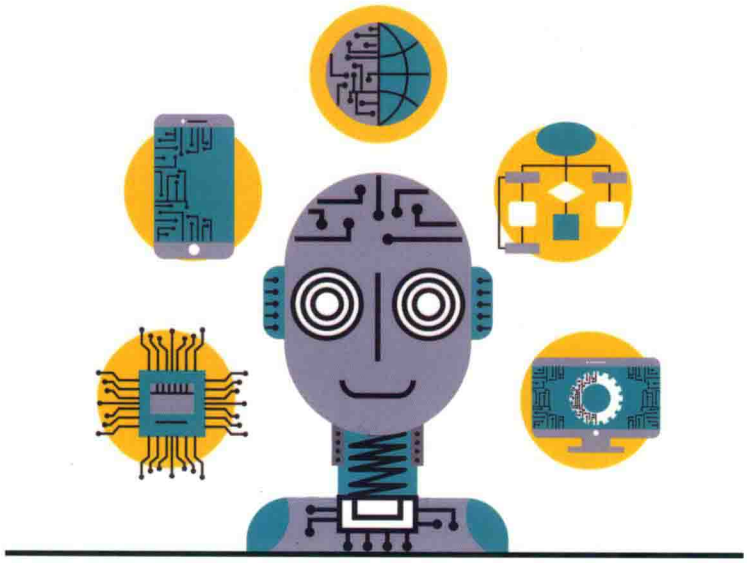
-SaaS 课程 | 实验 | 题库

教育部产学合作协同育人项目成果教材
西普教育研究院 IT 前沿技术方向高校系列教材

"十三五"江苏省高等学校重点教材



Machine Learning and Application

机器学习及应用

在线实验 + 在线自测

李克清 时允田 主编
高小惠 王勤宏 杨梦铎 林雪纲 副主编

实验吧
让实验更简单

-SaaS
开放实验云平台

- 内容新颖，可操作性强，层层深入，简明易懂。
- 从实用角度出发，重点培养动手解决问题的能力。
- 提供体系完整的在线实验，即学即练，书网结合。

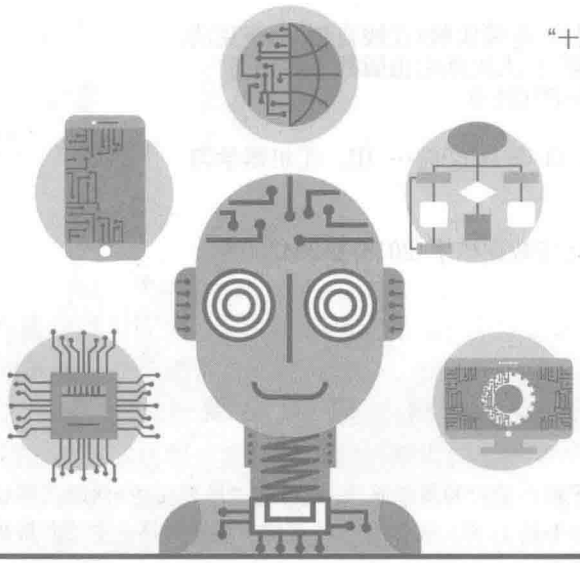
扫描书中二维码随时进行在线测试

中国工信出版集团
人民邮电出版社 POSTS & TELECOM PRESS

课程 | 实验 | 题库

教育部产学合作协同育人项目成果教材
西普教育研究院 IT 前沿技术方向高校系列教材

“十三五”江苏省高等学校重点教材
（编号：2018-2-020）



Machine Learning and Application

机器学习及应用

在线实验 + 在线自测

李克清 时允田 主编
高小惠 王勤宏 杨梦铎 林雪纲 副主编

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

机器学习及应用 ：在线实验+在线自测 / 李克清，时允田主编. — 北京 ：人民邮电出版社，2019.5
ISBN 978-7-115-50134-9

Ⅰ. ①机… Ⅱ. ①李… ②时… Ⅲ. ①机器学习
Ⅳ. ①TP181

中国版本图书馆CIP数据核字(2018)第264250号

内 容 提 要

本书详细地介绍了机器学习的基本原理，并采用“原理简述+问题实例+实际代码+运行结果”的模式介绍常用算法。全书共 11 章，主要包括决策树、神经网络、支持向量机、贝叶斯分类器、集成学习、聚类、降维等内容。

本书可以作为高校计算机及相关专业的教材，也可以作为机器学习培训班教材，并适合作为从事机器学习及应用的专业人员和广大机器学习爱好者的自学用书。

◆ 主　　编　李克清　时允田
副 主 编　高小惠　王勤宏　杨梦铎　林雪纲
责任编辑　左仲海
责任印制　焦志炜

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京圣夫亚美印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：16.25　　2019 年 5 月第 1 版
字数：396 千字　　2019 年 5 月北京第 1 次印刷

定价：49.80 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

前 言 FOREWORD

机器学习是计算机科学与人工智能的重要分支领域。这门学科所关注的问题是，计算机程序如何随着经验累积自动提高性能。机器学习使用实例数据或过去的经验训练计算机，以优化性能标准。当人们不能通过直接编写计算机程序来解决给定的问题，而是需要借助于实例数据或经验时，就需要用到机器学习。

.这是一本面向中文读者的机器学习教科书，为了使尽可能多的读者通过本书对机器学习有所了解，本书没有从理论的角度来揭示机器学习算法背后的数学原理。编者试图尽可能少地使用数学知识，主要通过“原理简述+问题实例+实际代码+运行结果”的模式来介绍每一个算法。然而，少量的概率、统计、代数、优化、逻辑知识似乎不可避免。学习计算机的人都知道，计算机是一门实践学科，没有真正实际运行，很难真正理解算法的精髓。本书的最大好处就是边学边用，非常适合于迈进机器学习领域的人员学习。实际上，即使对于那些对机器学习有所了解的人，通过代码实现也能进一步加深对机器学习算法的理解。

本书在内容上尽可能涵盖机器学习的基础知识，但作为机器学习入门的读物，并且受到授课时间的限制，很多重要的、前沿的材料未能覆盖，覆盖的部分也仅是管中窥豹。书中每章都给出了相应的习题，有的习题可帮助读者巩固本章学习，有的则是为了引导读者扩展相关知识。

全书共 11 章，高小惠编写了第 2、8、9 章，沈韬编写了第 11 章的后半部分，王勤宏编写了第 3、6 章，吴建平参与编写了第 5 章的部分内容，杨梦铎编写了第 1 章和第 11 章的前半部分，李克清编写了剩余的其他部分并统稿，时允田、林雪纲、薛乔毓进行了内容校验并提供 U-SaaS 在线实验平台资源支持。本书在写作过程中得到了编者单位的同事、学生和朋友的支持与帮助，在此对他们的无私奉献表示衷心的感谢。同时，也要感谢北京西普阳光教育科技股份有限公司的“教育部 2016 年西普教育产学合作协同育人项目”的资助，为本书的正式出版和发行提供了全力保障和后勤服务。

机器学习的发展极其迅速，目前已成为一个广袤的学科。编者才疏学浅，精力有限，书中疏漏和不足之处在所难免，恳请读者不吝告知，将不胜感激。

编 者

2018 年 7 月

平台支撑 PLATFORM SUPPORT

北京西普阳光教育科技股份有限公司（简称西普教育）开发的在线教育平台——实验吧（http://www.shiyanbar.com），提供了强大的集成实验环境及海量的在线教学资源，把配套的实验搬到线上，可以让读者更方便地结合本书进行实践。

1. 如何学习本书中的配套实验课程

（1）购买本书后，找到粘贴在本书封底的刮刮卡，刮开并获得学号。

（2）登录实验吧网站（http://www.shiyanbar.com），完成网站注册。

（3）登录人邮学院在线实验中心（http://rymooc.shiyanbar.com），输入在实验吧注册的账户及密码，完成登录（见图 1）。

（4）输入刮刮卡中的学号，姓名填写“人邮学院”，单击“保存”按钮，完成绑定（见图 2）。

图 1　登录在线实验平台

图 2　绑定学生信息

（5）完成绑定后，自动登录进入在线实验中心，开始学习本书配套的课程资源。

2. 如何学习本书中的配套练习题

实验吧教研团队为本书配置了丰富的课后练习题，读者通过扫描本书各项目里配置的习题二维码，即可进行在线自测，提交后自动判断正误，并提供正确答案。

目录 CONTENTS

第 1 章 导论 …… 1

1.1 引言 …… 1
1.2 基本术语 …… 2
1.3 概念学习与假设空间 …… 3
1.4 归纳偏好 …… 4
1.5 经验误差与过拟合 …… 5
1.6 模型评估与选择 …… 5
1.7 性能度量 …… 6
1.8 发展历程 …… 8
1.9 应用现状 …… 10
习题 1 …… 11

第 2 章 Python 初步 …… 12

2.1 Python 概述 …… 12
2.2 NumPy 库介绍 …… 12
2.2.1 ndarray 对象 …… 12
2.2.2 ufunc 函数 …… 14
2.2.3 常用函数库 …… 15
2.3 Matplotlib 库介绍 …… 19
2.3.1 快速绘制二维图表 …… 19
2.3.2 Artist 对象 …… 21
2.3.3 配置属性 …… 24
2.3.4 绘制三维图表 …… 24
2.4 SciPy 库函数 …… 26
2.4.1 线性代数模块 …… 26
2.4.2 优化和拟合模块 …… 28
2.4.3 统计模块 …… 30
2.4.4 稀疏矩阵模块 …… 32
2.5 scikit-learn 库函数 …… 35
2.5.1 sklearn.datasets …… 35
2.5.2 模型选择与评价 …… 36
2.5.3 scikit-learn 的机器学习 …… 44
习题 2 …… 46

第 3 章 决策树 …… 48

3.1 引言 …… 48
3.1.1 决策树的基本思想 …… 48
3.1.2 决策树的构造 …… 49
3.1.3 决策树的算法框架 …… 54
3.1.4 信息增益 …… 54
3.2 ID3 决策树 …… 57
3.2.1 ID3 算法 …… 57
3.2.2 ID3 的实现 …… 59
3.3 C4.5 决策树 …… 63
3.3.1 C4.5 算法 …… 63
3.3.2 C4.5 的实现 …… 64
3.4 sklearn 与回归树 …… 68
3.4.1 回归算法原理 …… 68
3.4.2 最小剩余方差法 …… 69
3.4.3 剪枝策略 …… 69
3.4.4 sklearn 实现 …… 70
习题 3 …… 72

第 4 章 神经网络 …… 73

4.1 引言 …… 73
4.1.1 人工神经网络的发展历程 …… 73
4.1.2 人工神经网络的特点 …… 74
4.1.3 人工神经网络的分类 …… 75
4.2 神经元模型 …… 75
4.3 感知机与多层神经网络 …… 77
4.3.1 感知机 …… 77
4.3.2 梯度下降法 …… 81

4.3.3 随机梯度下降法 …… 85
4.3.4 多层神经网络 …… 86
4.4 误差反向传播算法 …… 90
4.4.1 BP 神经网络学习算法 …… 90
4.4.2 BP 神经网络实验 …… 93
4.5 玻耳兹曼机 …… 95
4.5.1 BM 的拓扑结构 …… 96
4.5.2 BM 的学习过程 …… 96
4.6 综合案例 …… 99
习题 4 …… 101
第 5 章 支持向量机 …… 103
5.1 引言 …… 103
5.2 线性分类 …… 104
5.2.1 函数间隔与几何间隔 …… 104
5.2.2 对偶问题 …… 107
5.3 线性支持向量机 …… 108
5.4 非线性支持向量机 …… 111
5.4.1 核技巧 …… 111
5.4.2 sklearn SVC …… 113
5.5 序列最小优化算法 …… 117
5.6 综合案例 …… 119
习题 5 …… 125
第 6 章 贝叶斯分类器 …… 127
6.1 引言 …… 127
6.2 朴素贝叶斯分类 …… 128
6.2.1 朴素贝叶斯算法 …… 128
6.2.2 朴素贝叶斯分类算法 …… 129
6.2.3 朴素贝叶斯分类算法的 Python 实现 …… 131
6.2.4 sklearn 的朴素贝叶斯方法 …… 135
6.3 极大似然估计 …… 137
6.3.1 EM 算法 …… 138
6.3.2 EM 算法步骤 …… 140
6.3.3 三硬币的 EM 求解 …… 140
6.3.4 sklearn 的 EM 方法 …… 142
6.4 贝叶斯网络 …… 146
6.4.1 贝叶斯网络的构造和学习 …… 146
6.4.2 贝叶斯网络应用举例 …… 147
习题 6 …… 150
第 7 章 集成学习 …… 152
7.1 引言 …… 152
7.2 Voting …… 153
7.3 Bagging …… 156
7.4 Boosting …… 161
7.4.1 AdaBoost 法 …… 161
7.4.2 Gradient Boosting …… 165
7.5 综合案例 …… 168
习题 7 …… 171
第 8 章 聚类 …… 172
8.1 引言 …… 172
8.1.1 聚类的概念 …… 172
8.1.2 典型应用 …… 172
8.1.3 常见算法分类 …… 172
8.1.4 聚类算法中存在的问题 …… 173
8.2 距离计算 …… 173
8.2.1 闵可夫斯基距离 …… 173
8.2.2 欧几里得距离 …… 174
8.2.3 曼哈顿距离 …… 174
8.2.4 切比雪夫距离 …… 175
8.2.5 皮尔逊相关系数 …… 175
8.2.6 余弦相似度 …… 175
8.2.7 杰卡德相似系数 …… 176
8.3 *k*-means 聚类 …… 176
8.3.1 算法思想 …… 176
8.3.2 辅助函数 …… 177
8.3.3 编程实现 *k*-means 算法 …… 178
8.3.4 scikit-learn 中的 *k*-means 方法 …… 179
8.3.5 算法评价 …… 181
8.3.6 算法改进 *k*-means++ …… 181
8.4 密度聚类 …… 182
8.4.1 密度聚类算法思想 …… 182

8.4.2 DBSCAN 算法 …… 182
8.4.3 密度峰值聚类 …… 185
8.5 层次聚类 …… 187
8.5.1 层次聚类思想 …… 187
8.5.2 层次聚类实现 …… 188
8.6 综合实例 …… 190
8.6.1 聚类算法性能比较 …… 190
8.6.2 算法总结 …… 193
习题 8 …… 193
第 9 章 降维 …… 195
9.1 引言 …… 195
9.1.1 降维的概念 …… 195
9.1.2 常见算法分类 …… 195
9.2 *k*-近邻学习 …… 196
9.2.1 算法实现 …… 197
9.2.2 算法实例 …… 199
9.2.3 算法关键 …… 200
9.3 主成分分析 …… 201
9.3.1 算法思想 …… 201
9.3.2 算法实例 …… 202
9.4 低维嵌入 …… 205
9.4.1 算法原理 …… 205
9.4.2 算法实例 …… 206
9.4.3 算法评价 …… 208
9.5 奇异值分解 …… 209
9.5.1 SVD 算法原理 …… 209
9.5.2 SVD 算法及应用示例 …… 210
9.6 综合实例 …… 215
9.6.1 PCA 实例 …… 215
9.6.2 SVD 实例 …… 218
习题 9 …… 219
第 10 章 概率图模型 …… 221
10.1 引言 …… 221
10.2 马尔科夫过程 …… 222
10.2.1 基本概念 …… 222
10.2.2 隐马尔科夫模型 …… 225
10.3 Viterbi 算法 …… 227
10.4 综合案例 …… 231
习题 10 …… 233
第 11 章 深度学习初步 …… 235
11.1 引言 …… 235
11.2 表示问题 …… 235
11.3 学习问题 …… 236
11.4 优化问题 …… 238
11.5 认知问题 …… 238
11.6 基本模型 …… 239
11.6.1 自编码器 …… 239
11.6.2 受限玻耳兹曼机 …… 240
11.6.3 卷积神经网络 …… 242
11.7 TensorFlow 的简介与安装 …… 243
11.7.1 Python 3 环境 …… 243
11.7.2 安装 TensorFlow …… 243
11.7.3 验证 …… 243
11.8 TensorFlow 的基本使用 …… 243
11.9 基于卷积神经网络的 MNIST 手写体识别实验 …… 245
11.9.1 conv2d 函数 …… 245
11.9.2 max_pool 函数 …… 246
11.9.3 示例程序 …… 246
习题 11 …… 249
参考文献 …… 250

第1章 导论

本章知识点

- 机器学习的概念
- 机器学习基本术语
- 概念学习与假设空间
- 模型评估与选择

1.1 引言

什么是机器学习？从广义上来说，机器学习（Machine Learning）是计算机程序随着经验积累自动提高性能或系统自我改进的过程。以一个更形式化的定义来说，对于某类任务T和性能标准P，如果一个计算机程序在T上以P衡量性能，随着经验E而自我完善，就称这个计算机程序从经验E中学习。例如，对于手写识别学习的问题，任务T是识别和分类图像中的手写文字，性能标准P是分类的正确率，训练经验E是已知分类的手写文字数据库。这就是说，为了很好地定义一个计算机学习问题，学习问题的标准描述包括了3个基本的特征，即任务的种类、衡量任务提高的标准及经验的来源。简而言之，学习就是通过经验提高性能的某类程序。

机器学习正是致力于研究如何通过计算的手段及利用经验来改善系统自身性能的一门学科。为了在计算机上解决问题，通常需要一定的算法。这些对于特定任务设计的算法需要一定的输入，依据特定的指令序列对输入进行变换，得到输出。例如，我们可以设计一种排序算法，输入一个数字集合，输出这个数字集合的一个有序列表。然而，对于现实中的很多任务，我们并没有确定的算法，例如为每天收取的邮件进行分类，区分其是垃圾邮件还是正常邮件。尽管我们知道输入是邮件文档，输出为是否是垃圾邮件，然而并不知道应该按照怎样的规则将这种输入变换成输出。

事实正是如此，机器学习所面临的绝大多数问题都是这种没有确定算法的学习问题。对于这一类学习任务，我们希望计算机自动地为学习任务提取相应的算法。换言之，在排序的例子中，我们不需要学习如何将数字集合进行排序，因为已经有了排序的算法。然而在垃圾邮件的例子中，我们确实没有既定的算法，这需要计算机从过去的经验中进行学习，自动地提取出能够分类邮件的学习算法。

在计算机系统中，经验通常以数据的形式存在。为了能够自动地从经验中提取出学习

算法，需要获得过去大量的邮件实例来作为数据。接下来要做的事情就是从实例数据中学习出垃圾邮件的模型，以此作为判断的依据。因此，机器学习所研究的主要内容，是如何在计算机上从数据中产生模型的算法，即学习算法。有了学习算法，我们可将经验以数据的形式提供给计算机，计算机就能基于这些数据产生相应的模型。继而在面对新的情况时，学习到的模型能够提供相应的判断，比如计算机能够正确分类一封新邮件是否是垃圾邮件。从这个意义上来说，机器学习是研究学习算法的学问，机器学习的过程是从大量数据中自动地寻找有用模型的过程。

1.2 基本术语

要进行机器学习，首先需要有数据。考虑手写体数字识别的问题，假设每个数字对应一个 28 像素×28 像素的灰度图像，依照矩阵逐列首尾相接拼成向量的方式，每一幅数字图像可以表示为一个由 784 个实数组成的向量 $\boldsymbol{x}$。假定收集了一组手写体数字的图像，均以向量的形式表示，这组图像向量的集合称为一个数据集（Data Set），其中的每个向量是关于一幅手写体数字图像的描述，称为一个实例（Instance）或样本（Sample）。784 维向量中的每一维反映了图像在某个特定方面的表现性质，称为属性（Attribute）或特征（Feature）；属性的取值，也就是向量中每个元素对应的实数值，称为属性值（Attribute Value）。这些属性所张成的空间称为属性空间（Attribute Space），也叫样本空间（Sample Space）或输入空间（Input Space）。由此，所有特征张成一个用于描述手写体数字的 784 维空间，在这个属性空间中，每一幅图像对应了该空间中的一个点。由于空间中的每个点可以用一个坐标向量来表示，因此也把一个样本称为一个特征向量（Feature Vector）。

一般的，令 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$ 表示包含 m 个样本的数据集，每个样本由 d 个属性描述，则每个样本 $\boldsymbol{x}_i=(x_{i1},x_{i2},\cdots,x_{id})$ 是 d 维属性空间 $\boldsymbol{X}$ 中的一个向量，$\boldsymbol{x}_i\in\boldsymbol{X}$，其中 x_{ij} 是 $\boldsymbol{x}_i$ 在第 j 个属性上的取值，d 称为样本 $\boldsymbol{x}_i$ 的维数（Dimensionality）。

手写体数字识别问题的目标是建立一个机器，能够以代表一幅手写体数字图像的向量 $\boldsymbol{x}$ 作为输入，以 0～9 中的某一个数字作为输出。这不是一个简单的问题。通过人工编写规则解决这样的问题经常会给出较差的结果，原因是手写体数字形态变化多端。而使用机器学习的方法可以得到好得多的结果。

机器学习试图从数据中寻找特定的模型，这种从数据中学得模型的过程称为学习（Learning）或训练（Training）。在学习算法中，一个由 N 个数字组成的大的集合 $\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N\}$ 被称作训练集（Training Set），用来调节模型的参数。这些在训练过程中使用的数据也称为训练数据（Training Data），其中的每个样本称为一个训练样本（Training Sample）。训练集就是所有训练样本组成的集合。训练集中数字的类别称为目标向量（Target Vector），用来代表训练数据的标记信息或标签（Label）。拥有了标记信息的样本称为样例（Example）。一般的，用 $(\boldsymbol{x}_i,y_i)$ 表示第 i 个样例，其中，$y_i\in Y$，y_i 是样本 $\boldsymbol{x}_i$ 的标记，Y 是所有标记的集合，亦称标记空间（Label Space）或输出空间（Output Space）。依据训练数据是否拥有标记信息，机器学习任务可大致划分为监督学习（Supervised Learning）和无监督学习（Unsupervised Learning）两大类。

机器学习算法可以表示为一个函数 $y=f(x)$，假设以手写体图像 x 作为输入，向量 y 作为输出，其中，向量 y 的形式与目标向量的形式相同。在训练数据的基础上，函数 $f(x)$ 的精确形式在训练阶段或学习阶段被确定。一旦模型被训练出来，它就能确定新的手写体数字集合中的图像标签。这些新的手写体数字图像组成的集合称为测试集（Test Set）。使用学习得到的模型进行预测的过程称为测试（Testing），被预测的样本称为测试样本（Testing Sample）。学习得到的模型适用于新样本的能力称为泛化（Generalization）能力。

如果希望预测的结果是离散值，如这里的 0～9 这 10 个数字，此类学习任务称为分类（Classification）；如果希望预测的结果是连续值，此类学习任务称为回归（Regression）。在分类任务中，将只涉及两个类别的学习任务称为二分类（Binary Classification）任务。通常称其中一个类为正类（Positive Class），相应的样例称为正例（Positive Example）；称另一个类为负类或反类（Negative Class），相应的样例称为负例或反例（Negative Example）。当涉及多个类别时，学习任务则称为多分类（Multi-Class Classification）任务。如果在学习过程中使用的训练样本不具备标记信息，我们依然希望可以将训练集中的数据分成若干组，这样的学习任务称为聚类（Clustering）。聚类任务中的每个组称为一个簇（Cluster），这些自动形成的簇对应一些潜在概念的划分，有助于我们了解数据内在的规律。从训练数据是否拥有标记信息的角度我们可以知道，分类和回归是监督学习的代表，而聚类则是非监督学习的代表。

1.3 概念学习与假设空间

归纳（Induction）与演绎（Deduction）是科学推理的两大基本手段。归纳是从特殊到一般的泛化过程，即从具体的事实归结出一般性规律；演绎则是从一般到特殊的特化（Specialization）过程，即从基础原理推演出具体情况。从特殊的训练样例中归纳出一般函数是机器学习的中心问题，该归纳过程称为归纳学习（Inductive Learning）。

归纳学习有广义与狭义之分，前者通常指的是从样例中学习的归纳过程，而后者指的是从训练数据中学得概念（Concept），因此狭义的归纳学习也称为概念学习（Concept Learning）。对概念学习有所了解，有助于理解机器学习的一些基础思想。

概念学习考虑的问题是，给定一样例集合及每个样例是否属于某一概念的标记，怎样自动推断出该概念的一般定义。概念定义在一个实例（Instance）集合之上，这个集合表示为 X。在学习目标概念时，必须提供一套训练样例（Training Examples），每个样例为 X 中的一个实例 $\boldsymbol{x}$ 及它的目标概念值 $c(\boldsymbol{x})$。通常用序偶 $\langle \boldsymbol{x}, c(\boldsymbol{x}) \rangle$ 来描述训练样例，表示包含了实例 $\boldsymbol{x}$ 和目标概念值 $c(\boldsymbol{x})$。符号 D 用来表示训练样例的集合。

一旦给定目标概念 c 的训练样例集，学习器面临的问题就是假设或估计 c。可以把学习过程看作一个在所有可能假设（All Possible Hypotheses）的集合上进行搜索的过程，搜索的目标是找到与训练集匹配（Match）或拟合（Fit）的假设。这些所有可能的假设（Hypothesis）组成的空间称为假设空间（Hypothesis Space）。假设空间中的假设集合才是确定目标概念所考虑的范围，通常使用符号 $\mathcal{H}$ 来表示。机器学习的目标就是寻找一个假设 h，使对于 X 中的所有 $\boldsymbol{x}$ 有 $h(\boldsymbol{x})=c(\boldsymbol{x})$。换言之，机器学习的任务是在整个实例集合 X 上确定与目标概念

c 相同的假设 h。

事实上，目标概念 c 仅仅是训练样例上的信息，没有包含测试样例。因此，归纳学习算法最多只能保证输出的假设能与训练样例相拟合。如果没有更多的信息，我们只能假定，对于未见实例，最好的假设就是与训练数据最佳拟合的假设。由此引出归纳学习的一个基本假设，即归纳学习假设：**任一假设如果在足够大的训练样例集中很好地逼近目标函数，它也能在未见实例中很好地逼近目标函数。**

概念学习可以看为一个搜索过程，范围是假设的表示所隐含定义的整个空间。搜索的目标是寻找能最好地拟合训练样例的假设。自然的，对学习算法的研究需要考查假设空间搜索的不同策略。特别引起我们兴趣的算法应能有效地搜索非常大的或无限大的假设空间，以找到最佳拟合训练数据的假设。有很多策略可以对这个假设空间进行搜索，如自顶向下、从一般到特殊，或是自底向上、从特殊到一般。搜索过程中可以不断删除与正例不一致的假设和（或）与反例一致的假设，最终获得与训练集一致的假设，即对所有训练样本能够进行正确判断的假设，这就是我们学得的结果。

1.4 归纳偏好

必须注意到，当假设的表示形式选定后，也就隐含地为学习算法确定了所有假设的空间。这些假设是学习程序所能表示的，也是它能够学习的。通常情况下，当给定正确的训练样例且保证初始假设空间包含目标概念时，学习算法可以收敛到目标概念。如果要保证假设空间包含目标概念，一个明显的方法是扩大假设空间，使每个可能的假设都包含在内。在现实问题中，通常会面临很大的假设空间，但学习过程是基于有限样本训练集进行的，因此，可能有多个假设与训练集一致，即存在着一个与训练集一致的假设集合（Hypothesis Set），称为版本空间（Version Space）或变形空间，因为它包含了目标概念所有合理的变形。然而，对于一个具体的学习算法而言，它必须产生一个模型。这时，学习算法本身的偏好（Bias）就会起到关键的作用。机器学习算法在学习过程中对某种类型假设的偏好，称为归纳偏好（Inductive Bias）。

任何一个有效的机器学习算法必有其归纳偏好，否则它将被假设空间中看似在训练集上等效的假设所迷惑，而无法产生确定的学习结果。这说明了归纳学习的一个基本属性：**学习器如果不对目标概念的形式做预先的假定，从根本上就无法对未见实例进行分类。**可以说，归纳偏好是学习器从训练样例中泛化并在推断新实例的分类过程中所采用的策略。归纳偏好可看作学习算法在对所有假设进行选择时的“价值观”。一种算法的有偏性越强，那么它的归纳能力就越强，可以分类更多的未见实例。

引导学习算法确立“正确”偏好的一个一般性原则是奥卡姆剃刀（Occam's Razor）原则，即优先选择拟合数据的最简单的假设。这就是说，当有多个假设与观察一致时，则选择最简单的那个。这里的“一致”是指假设能够正确分类训练样例集合 D 中的每一个样例，即对于 $\forall\langle \boldsymbol{x}, c(\boldsymbol{x})\rangle \in D$，都有 $h(\boldsymbol{x})=c(\boldsymbol{x})$。奥卡姆剃刀的一种解释是短假设基于简单的参数组合，因此其数量少于长假设的数量，所以找到一个短的但同时与训练数据拟合的假设的可能性较小。当然，奥卡姆剃刀并非是唯一可行的原则。

事实上，归纳偏好对应了学习算法本身所做出的关于“什么样的模型更好”的假设。在具体的现实问题中，这个假设是否成立，即算法的归纳偏好是否与本问题自身匹配，大多数时候直接决定了算法能否取得好的性能。此外，必须认识到，脱离具体的问题，空泛地谈论“什么学习算法更好”毫无意义。要谈论算法的相对优劣，必须要针对具体的学习问题。在某些问题上表现好的学习算法，在另一些问题上却可能不尽如人意。学习算法自身的归纳偏好与问题是否相配，往往会起到决定性的作用。

1.5 经验误差与过拟合

通常，把分类错误的样本数占样本总数的比例称为错误率（Error Rate），即如果在 m 个样本中有 a 个样本分类错误，则错误率为 $E=a/m$；相应的，$1-E$ 称为精度，即精度=1−错误率。更一般的，把学习器的实际预测输出与样本的真实输出之间的差异称为误差（Error），学习器在训练集上的误差称为训练误差（Training Error）或经验误差（Empirical Error），在新样本上的误差称为测试误差（Testing Error）或泛化误差（Generalization Error）。显然，希望得到泛化误差小的学习器。然而，事先并不知道新样本是什么样的，实际能做的是努力使经验误差最小化。为了能让学习器在新样本上表现得出色，应该从训练样本中尽可能学习到适用于所有潜在样本的普遍规律，这样才能在遇到新样本时做出正确的判断。

然而，当学习器把训练样本学得“太好”了的时候，很可能已经把训练样本自身的一些特点当作了所有潜在样本都会具有的一般性质，这样就会导致泛化性能下降。这也就是说，对于一个假设，当存在其他假设对训练样例的拟合比它差，但在训练集以外的实例上表现得更好时，就说这个假设过度拟合训练样例。这种现象在机器学习中称为过拟合（Overfitting）。与过拟合相对的是欠拟合（Underfitting），这是指学习器对训练样本的一般性质尚未学好。

有很多种因素可能导致过拟合情况的发生。一种可能原因是训练样例含有随机错误或噪声，当假设试图拟合含有噪声的训练样例后，学习器的泛化能力自然会受到影响。事实上，当训练数据没有噪声时，过拟合也有可能发生。一种最常见的情况是由于学习器的学习能力过于强大，以至于把训练样例所包含的不太一般的特性都学到了。还有一种情况是训练样例太少，很可能出现巧合的规律性，使得一些属性恰巧可以很好地分割样例，但却与实际的目标函数无关系。一旦这种巧合的规律性存在，就有过拟合的风险。过拟合是机器学习面临的关键障碍，各类学习算法都必然带有一些针对过拟合的措施。然而，我们必须认识到，过拟合是无法彻底避免的，只能够尽量地缓解过拟合或者减小过拟合带来的风险。

1.6 模型评估与选择

在现实任务中，往往有多种学习算法可供选择，甚至对同一个学习算法，当使用不同的参数配置时，也会产生不同的模型。那么，该选用哪一个学习算法，使用哪一种参数配置呢？这就是机器学习中的模型选择（Model Selection）问题。

从归纳偏好的角度来说，模型选择要解决的问题实际上是如何选择正确的归纳偏好。

对于这种问题的解答，应当记住，机器学习的目标不是复制训练数据，而是预测新情况。也就是说，希望对于训练集之外的输入能够产生正确的输出，而这个正确的输出并没有在训练集中给出。换言之，希望训练的模型能够很好地泛化。通常，可使用评估方法来对学习器的泛化误差进行评估，进而进行模型选择。

如果访问训练集以外的数据，就能够度量假设的泛化能力，即它的归纳偏好的质量。通过将已有的训练集划分为两部分来模拟这一过程：使用一部分来做训练，即拟合一个假设；而剩下的部分称作验证集（Validation Set），用来检验假设的泛化能力。也就是说，给定可能的假设类的集合 $\mathcal{H}_i$，对于每一个集合，我们在训练集上拟合最佳的 $h_i \in \mathcal{H}_i$。假定训练集和验证集都足够大，则在验证集上最准确的假设就是最好的假设，即具有最佳归纳偏好的假设。这一过程称为交叉验证（Cross-Validation）。

交叉验证法是一种常见的评估方法，通常把交叉验证法称为 k-折交叉验证或 k-倍交叉验证（k-fold Cross-Validation）。具体的做法是，先将数据集 D 划分为 k 个大小相似的互斥子集，即 $D = D_1 \cup D_2 \cup \cdots \cup D_k$，$D_i \cap D_j = \phi(i \neq j)$。每个子集 $D_i(i = 1,2,\cdots,k)$ 都尽可能保持数据分布的一致性，避免因数据划分过程中引入额外的偏差而对最终结果产生影响。如果从采样（Sampling）的角度来看待数据集的划分过程，则保留类别比例的采样方式通常称为分层采样（Stratified Sampling），即每个子集 D_i 都从 D 中分层采样得到。然后，每次用 $k-1$ 个子集的并集作为训练集，余下的那个子集作为验证集。这样就可以获得 k 个组训练/验证集，从而可进行 k 次训练和验证，最终返回的是这 k 个验证结果的均值。

值得一提的是，验证集是在模型评估与选择中用于评估测试的数据集，而学得模型在实际使用中遇到的数据集称为测试集，它包含在训练阶段或验证阶段未使用过的数据中。如果需要报告学得模型的期望误差，就不应该使用验证误差，而应该使用测试误差。在研究对比不同算法的泛化性能时，用测试集上的判别效果来估计模型在实际使用时的泛化能力，而把训练数据另外划分为训练集合验证集，基于验证集上的性能来进行模型选择。另外，需注意的是，也不能一直使用相同的训练集和验证集划分，因为一旦使用一次，验证集实际上就已经成为训练集的一部分了。

一定要记住，使用的训练数据是一个随机样本。也就是说，对于相同的应用，如果多次收集数据，则将得到稍微不同的数据集，拟合的 h 也稍微不同，并且具有稍微不同的验证误差。或者，如果把固定的数据集划分成训练集、验证集和测试集，则依赖于如何划分，会有不同的误差。这些微小的不同可以估计多大的差别可以看作是显著的而非偶然的。换言之，在假设类 $\mathcal{H}_i$ 和 $\mathcal{H}_j$ 之间进行选择时，我们将在大量训练集和验证集上多次使用它们，并检查 h_i 和 h_j 的平均误差之差是否大于多个 h_i 之间的平均差。

1.7 性能度量

对学习器的泛化性能进行评估，不仅需要有效可行的实验估计方法，还需要有衡量模型泛化能力的评价标准，这就是性能度量（Performance Measure）。对于预测任务，给定样例集 $D = \{(\boldsymbol{x}_1, y_1),(\boldsymbol{x}_2, y_2),\cdots,(\boldsymbol{x}_m, y_m)\}$，其中，$y_m$ 是实例 $\boldsymbol{x}_m$ 的真实标记。要评估学习器 f 的性能，就要把学习器预测结果 $f(\boldsymbol{x})$ 与真实标记 y 进行比较。

在分类任务中，错误率和精度是最常用的两种性能度量，既适用于二分类任务，也适用于多分类任务。对于样例集 D，分类错误率定义为式（1.1）。

$$E(f;D)=\frac{1}{m}\sum_{i=1}^{m}I\left[f(\boldsymbol{x}_i)\neq y_i\right] \tag{1.1}$$

精度则定义为式（1.2）。

$$\mathrm{acc}(f;D)=\frac{1}{m}\sum_{i=1}^{m}I\left[f(\boldsymbol{x}_i)=y_i\right]=1-E(f;D) \tag{1.2}$$

其中，$I(\bullet)$ 是指示函数，若 $\bullet$ 为真则取值为 1，否则取值为 0。更一般的，对于数据分布 D 和概率密度函数 $p(\bullet)$，错误率可描述为式（1.3）。

$$E(f;D)=\int_{\boldsymbol{x}\in D}I\left[f(\boldsymbol{x})\neq y\right]p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \tag{1.3}$$

精度可描述为式（1.4）。

$$\mathrm{acc}(f;D)=\int_{\boldsymbol{x}\in D}I\left[f(\boldsymbol{x})=y\right]p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}=1-E(f;D) \tag{1.4}$$

在回归任务中，最常用的性能度量是均方误差（Mean Squared Error），其定义为式（1.5）。

$$E(f;D)=\frac{1}{m}\sum_{i=1}^{m}\left[f(\boldsymbol{x}_i)-y_i\right]^2 \tag{1.5}$$

更一般的，对于数据分布 D 和概率密度 $p(\bullet)$，均方误差可描述为式（1.6）。

$$E(f;D)=\int_{\boldsymbol{x}\in D}\left[f(\boldsymbol{x})-y\right]^2p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \tag{1.6}$$

错误率和精度虽然常用，但是并不能满足所有的任务需求。例如在信息检索中，经常会关心“检索出的信息中有多少比例是用户感兴趣的”“用户感兴趣的信息中有多少被检索出来了”。更适用于此类需求的性能度量是查准率（P）与查全率（R）。

对于二分类问题，可将样例根据其真实类别与学习器预测类别的组合划分为真正例（True Positive，TP）、假正例（False Positive，FP）、真负例（True Negative，TN）、假负例（False Negative，FN）4 种情形。对于一个正实例，如果预测也是正的，则它是一个真正例；如果对正实例的预测是负的，则它是一个假负例。对于一个负实例，如果预测也是负的，则它是一个真负例；如果将负实例预测为正的，则它是一个假正例。在二分类问题中有两种类型的错误，即假正例和假负例。让我们设想一种身份认证应用，其中用户通过声音登录其账户。假正例是错误地允许冒名顶替者（模仿真实用户的声音）登录，而假负例则会拒绝合法用户。

令 TP、FP、TN、FN 分别表示真正例、假正例、真负例、假负例对应的样例数，则显然有 TP+ FP+ TN+ FN = 样例总数 。分类结果的混淆矩阵（Confusion Matrix）如表 1-1 所示。

表 1-1 分类结果的混淆矩阵

真实情况 \ 预测结果	正　例	负　例
正例	TP（真正例的样例数）	FN（假负例的样例数）
负例	FP（假正例的样例数）	TN（真负例的样例数）

查准率 P 与查全率 R 分别定义为式（1.7）和式（1.8）。

$$P=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{1.7}$$

$$R=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{1.8}$$

查准率和查全率是一对矛盾的度量。一般来说，查准率高时，查全率往往偏低；而查全率高时，查准率往往偏低。在一些应用中，对查准率和查全率的重视程度有所不同。例如在商品推荐系统中，为了尽可能少地打扰用户，更希望推荐的内容的确是用户感兴趣的，此时查准率更重要；而在逃犯信息检索系统中，更希望尽可能少地漏掉逃犯，此时查全率更重要。

在聚类任务中，样本集 $D=\{x_1,x_2,\cdots,x_m\}$ 包含 m 个无标记样本。评估学习器 f 的性能标准是尽量使得聚类结果的簇内相似度（Intra-Cluster Similarity）高，并且簇间相似度（Inter-Cluster Similarity）低。聚类性能度量亦称聚类有效性指标（Validity Index），大致可以分为两类：一类是将聚类结果与某个参考模型（Reference Model）进行比较，称为外部指标（External Index）；另一类是直接考察聚类结果而不利用任何参考模型，称为内部指标（Internal Index）。

1.8 发展历程

机器学习的发展离不开人工智能（Artificial Intelligence，AI）研究的推动。随着人工智能领域发展到一定的阶段，机器学习应运而生。

20 世纪 50 年代到 70 年代初，是人工智能领域的“推理期”。这个时代的普遍思想是，只要赋予机器一定的逻辑推理能力，机器就能够具有智能（Intelligent）。代表性的工作包括纽厄尔（A. Newell）和西蒙（H. Simon）的逻辑理论家程序及通用问题求解程序等。这些程序在对一些著名数学定理的证明上取得了令人振奋的成果。例如，1952 年，逻辑理论家程序证明了数学家罗素（Bertrand Russell）和怀特海（Alfred North Whitehead）的名著《数学原理》中的 38 条定理；1963 年证明了这本著作中的全部 52 条定理。

然而，随着研究的逐渐发展，人们认识到仅仅利用推理能力来实现人工智能是远远不够的。人工智能的实现还需要设法让机器拥有知识。

20 世纪 70 年代中期开始，是人工智能的“知识期”。大量的专家系统是这个时期的代表性产物，为很多应用领域做出了巨大贡献。知识工程之父费根鲍姆（E.A. Feigenbaum）在 1994 年获得了图灵奖（A.M. Turing Award）。

但是，随着知识工程瓶颈的到来，专家系统很难将人类总结出来的知识教授给计算机。于是，一个被学者广泛接受的想法随之产生：让机器自己学习知识。

实际上，早在 1950 年，图灵（A.M.Turing）就已经提出了机器学习的可能性。20 世纪 50 年代，研究者陆续开展了有关机器学习的研究工作。代表性工作主要有罗森勃拉特（F. Rosenblatt）的感知机及威德罗（B. Widrow）的自适应线性神经元（Adaline）。20 世纪 60 年代至 70 年代，以决策理论为基础的统计学习技术及强化学习技术得到了初步发展，基于逻辑或图结构表示的符号学习技术也开始出现。前者的代表性工作主要有塞缪尔（A.L.

Samuel）的跳棋程序及尼尔森（N.J. Nilsson）的学习机器等；后者的代表性工作主要有温斯顿（P. Winston）的结构学习系统、麦可尔斯基（R.S. Michalski）等人的基于逻辑的归纳学习系统、亨特（E.B. Hunt）等人的概念学习系统等。

20 世纪 80 年代，机器学习已经成为一个独立的学科领域，并且作为解决知识工程瓶颈问题的关键开始显现出巨大的张力。

1980 年的夏天，第一届机器学习研讨会在美国卡内基梅隆大学举办；同年，《策略分析与信息系统》（*Strategic analysis and information systems*）连续出版三期机器学习专辑；1983 年，Tioga 出版社出版了米哈尔斯基（R.S. Michalski）、加博内尔（J.G. Carbonell）和米切尔（T.M. Mitchell）主编的《机器学习：一种人工智能途径》（*Machine learning: an artificial intelligence approach*）一书，书中汇集的学者文章对当时的机器学习研究工作进行了总结，引起了很大反响；1986 年，《机器学习》（*Machine Learning*）创刊；1989 年，Artificial Intelligence 出版了机器学习专辑，当时的一些比较活跃的研究工作后来被引用在 J.G. Carbonell 主编的 MIT 出版社 1990 年出版的《机器学习：风范与方法》一书中。

20 世纪 90 年代中期之前，受到专家系统发展的影响，逻辑知识表示与归纳逻辑程序设计成了这一时期的主流。马格莱顿（S.H. Muggleton）主编的书对这一时期的归纳逻辑程序设计方面的研究工作做了总结。然而，归纳程序设计因为过强的表示能力而面临很多假设空间问题，因此很快关于这方面的研究就陷入了低谷。与此同时，鲁梅尔哈特（D.E. Rumelhart）、欣顿（G.E. Hinton）和威廉姆斯（R.J. Williams）发明了著名的 BP 算法，基于神经网络的连接主义学习由于 BP（Back Propagation）算法的成功而迅速兴起。相比于归纳逻辑程序设计，连接主义学习所面临的假设空间要小得多，很多实际问题都得到了解决。当然，连接主义学习的局限性也很快被认识到：大量的经验参数需要通过试错的方式来调试，学习效果很多时候依赖于一个好的参数设置。

20 世纪 90 年代中期，统计学习开始占据机器学习的主流，统计学习理论为学习器提供了强有力的理论基础。特别的，有效的支持向量机算法在 20 世纪 90 年代被波沙（B.E. Boser）、居翁（I. Guyon）和瓦普尼克（V.N. Vapnik）提出，而其优越的性能也在阿基姆（T. Joachims）等人对文本分类的研究中显现了出来，使得支持向量机算法红极一时。支持向量机的广泛使用，使得核方法（Kernel Method）逐渐成为机器学习中被普遍使用的一种基本技巧。统计学习在 20 世纪 90 年代中期开始成为机器学习的主流技术。

然而，利用统计学习描绘学习问题有时候又过于理想化。统计特性或简化问题时经常需要做出一定的假设，但是这些假设往往在真实世界中是难以成立的。另外，虽然理论上说把原始空间利用核技巧转化到一个新的特征空间可以解决十分复杂的问题，但困难的是如何选择合适的核映射，这经常有着浓重的经验主义色彩。

21 世纪初，连接主义学习以深度学习（Deep Learning）算法的形式卷土重来，卓越的性能迅速引发了学术界和工业界的热潮。究其原因，一是硬件水平的发展使得设备的计算能力得到强有力的提升，二是大数据（Big Data）时代的到来为深度学习模型提供了大储量的数据样本。当然，严格的理论基础的缺乏是深度学习一个无法回避的软肋。此外，连接主义学习很难有效地表示出复杂数据之间的关系，这一方面使得领域知识的使用变得困难，另一方面也让学习结果具有黑箱的性质。不过，换个角度来说，深度学习严格的理论基础