带您启航机器学习的风帆
用简单的方式讲述复杂的算法
提供完整Java代码及实验数据下载

肖云鹏 卢星宇 许明 汪浩瀚 吴斌 刘宴兵 著

机器学习
经典算法实践

Classical Machine Learning
Algorithms in Practice

清华大学出版社

肖云鹏 卢星宇 许明 汪浩瀚 吴斌 刘宴兵 著

机器学习
经典算法实践

Classical Machine Learning Algorithms in Practice

清华大学出版社
北京

内 容 提 要

机器学习是数据分析、智能技术的核心课程，本书作为该领域的入门教程，选择了机器学习领域的十大经典算法，讲原理、给数据、给源码、给实验，带入门。正如本书封面表达的那样，本书希望带您启航机器学习的风帆，用简单的方式讲述复杂的算法，提供完整 Java 代码及实验数据下载。

本书可作为高等院校计算机、软件工程及自动化相关专业的本科生或研究生教材，也可作为对机器学习感兴趣的研究人员和工程技术人员的参考读物。

图书在版编目(CIP)数据

机器学习经典算法实践/肖云鹏等著. —北京：清华大学出版社，2018
ISBN 978-7-302-49333-4

Ⅰ. ①机…　Ⅱ. ①肖…　Ⅲ. ①机器学习－算法－高等学校－教材
Ⅳ. ①TP181

中国版本图书馆 CIP 数据核字(2018)第 004237 号

责任编辑：贾　斌　薛　阳
封面设计：常雪影
责任校对：李建庄
责任印制：丛怀宇

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 装 者：三河市国英印务有限公司
经　　销：全国新华书店
开　　本：210mm×235mm　　**印　　张**：12.5　　**字　　数**：200 千字
版　　次：2018 年 5 月第 1 版　　**印　　次**：2018 年 5 月第1次印刷
印　　数：1～1500
定　　价：49.00 元

产品编号：075180-01

FOREWORD ◀

前　言

现在，大数据、社交网络、计算智能、深度学习等词汇都已经成为人们日常生活中经常看到的热门专业名词。如果我们考虑这些领域的共性，那么机器学习一定是重要的交集部分。很多来自不同领域、不同角色的学生、工作人员都在加入学习机器学习的队伍。

本书的编写面向正走在或即将走向学习机器学习路上的广大读者。我们在日常教学和培养研究生过程中发现，很多学生一方面想学、愿意学机器学习，另一方面又遇到入门难的问题，希望能有一本书、一本教材讲原理、给数据、给源码、给实验，带着入门。鉴此我们编写了这本书，选择了机器学习领域的十大经典算法，把我们平常培养刚入校研究生的算法材料进行整理，提供给广大希望学习的读者朋友们。

本书在整体章节的安排上，按照监督{KNN(分类)，Bayes(分类)，C4.5(分类)，SVM(分类)，AdaBoost(分类)，CART(回归)}和无监督{K-Means(聚类)，Apriori(关联规则)，PageRank(排序)，EM(参数估计)}的顺序组织。在每一章的讲解中，从讲故事开始讲解算法原理，接着分别从算法实现类/方法流程图、类/方法说明表、关键代码讲解算法实现，然后给出实验数据，最后给出实验结果与分析，尽量做到简单易懂。每章完整的源代码扫描下面二维码即可下载，每个算法对应一个 Java 工程，实验数据都在每个工程的 data 文件夹下。代码风格尽量保持一致，让读者更容易理解。

本书的写作工作是由我们实验室两位老师(肖云鹏和刘宴兵教授)以及复旦大学卢星宇博士、清华大学许明博士、CMU 汪浩瀚博士和北京邮电大学吴斌教授共同完成，几位作者都是长期在机器学习领域从事科学研究、工程实践、项目合作的科研

扫码下载完整代码及实验数据

人员和高校工作者。我们的想法是通过努力，以开放的心态，帮助更多的希望学习机器学习的读者。

即使只是作为一本入门级的学习读物，整个书稿前前后后也修改了几十稿。同时我们也参考学习了很多机器学习方面的书籍和网络资源，真高兴当下国内有许多学者、产业界人员和互联网热心人提供这么多优秀的学习资源。诚然，即便是我们非常努力地完善书稿，由于水平有限和时间仓促，书中可能还会有这样或那样的问题，请读者批评指正。另外，算法自身也在不断更新，凡是内容有更新的地方都会体现在本书的后继版本中，我们也希望本书的第二版、第三版等不仅是内容的进一步完善，还会加入更多有趣的算法，从传统机器学习到深度学习、增强学习。其实，机器学习经典算法又何止这十大呢！

最后，感谢我的家人对我工作的支持，感谢实验室学生们在本书的写作过程中帮着收集材料、提意见、讨论书稿，所有的过程都是美好回忆。

本书的完成得到国家 973 重点基础研究发展计划（No. 2013CB329606）、重庆市重点研发项目（No. cstc2017zdcy-zdyf0299，No. cstc2017zdcy-zdyf0436）、重庆市基础科学与前沿技术研究项目（No. cstc2017jcyjAX0099）和重庆邮电大学出版基金资助。

肖云鹏

2018 年 4 月

图书资源支持

感谢您一直以来对清华版图书的支持和爱护。为了配合本书的使用，本书提供配套的资源，有需求的读者请扫描下方的“书圈”微信公众号二维码，在图书专区下载，也可以拨打电话或发送电子邮件咨询。

如果您在使用本书的过程中遇到了什么问题，或者有相关图书出版计划，也请您发邮件告诉我们，以便我们更好地为您服务。

我们的联系方式：

地　　址：北京海淀区双清路学研大厦 A 座 707

邮　　编：100084

电　　话：010－62770175－4604

资源下载：http://www.tup.com.cn

电子邮件：weijj@tup.tsinghua.edu.cn

QQ：883604(请写明您的单位和姓名)

资源下载、样书申请

书圈

用微信扫一扫右边的二维码，即可关注清华大学出版社公众号“书圈”。

目 录

CONTENTS ◀

第1章

KNN

1.1 KNN 算法原理

如果已知一个人的大部分朋友的爱好，要把这个人的爱好用最简单的分类问题做预测（分类），办法就是通过统计他最亲密（Nearest Neighbor，某种距离函数方法确定中的最近）的 K 个朋友中最多的爱好，这就是 KNN 算法。在这个算法中，已知朋友越多（训练数据完备性越好）、朋友圈子分离越大（不同簇的距离越大），算法越好。由于该算法原理简单、易于理解，目前应用领域至少包括文本处理、模式识别、计算机视觉、通信工程、生物工程，甚至 NBA 等体育数据分析。

1.1.1 算法引入

假定某个人有 20 个亲密的朋友，其中有 9 个人的爱好是打篮球，6 个人的爱好是打乒乓球，5 个人的爱好是打排球，那么就可以猜测这个人更可能喜欢打篮球。

这个过程就是利用 KNN 算法思想进行分类的过程，其标准的描述如下：假定有三类体育运动分别是篮球、乒乓球和排球，要求判断这个用户喜爱的运动。根据上述过程，要做出这个判断，首先得找出用户亲密的朋友，而且数量是 20 个，然后根据这

20 个亲密的朋友的爱好做出判断。

由此类推到 KNN 算法。K-近邻算法是一款简单实用的分类算法，通过测量不同样本之间的距离，然后根据距离最近的 K 个邻居来进行分类。整个分类过程主要有三个步骤。第一，算距离。要判断哪些是用户亲密的朋友，需要一个邻近度量方法。第二，求近邻。“20 个”就是用户近邻用户，为什么是 20 个？这就是 K 值的选择问题。第三，做决策。用户的爱好最终是与近邻用户人数最多的一个类别，这里采用了多数表决的方法进行分类决策，可以概括为“随大流”的思想。

1.1.2 科学问题

问题输入：训练数据集

$$Z = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\} \tag{1-1}$$

其中，$x_i \in \chi \subseteq \mathbf{R}^n$ 为实例样本的特征向量，$y_i \in \{c_1, c_2, \cdots, c_n\}$ 为实例的类别；实例特征向量 x；近邻用户个数 K。

问题输出：实例 x 的类别 y。

1.1.3 算法流程

构建 K-近邻算法主要分为三个步骤：算距离，取近邻，做决策。下面详细解释这三个步骤。

(1) 算距离：计算测量值与样本集中每个数据的距离。常见距离的度量方法包括欧几里得距离和夹角余弦等。一般来说，文本分类使用夹角余弦比欧几里得距离更加合适。

(2) 取近邻：将计算好的距离排序，选择 K 个距离最近的样本点。选择合适的 K 值，对算法分类的效果尤为重要。如果 K 值太小，则分离器容易受到训练数据中的噪声影响；如果 K 值太大，分类器可能会误分类测试样本。可利用交叉验证的方案来选择 K 值。

(3) 做决策：得到近邻列表后，采用多数表决的方法对测试样本进行分类。在多数表决中，每个近邻对分类的影响都一样，这使得算法对 K 的选择很敏感。降低 K

的影响的一种途径是根据每个近邻距离的不同对其作用加权。

1.1.4 算法描述

算法 1-1 是对 K-近邻算法的描述。算法首先对每个测试样本实例 $x_{si} \in X$ 计算与所有训练集 $(x_i, y_i) \in Z$ 之间的距离，得到近邻列表 D_z，然后根据近邻列表的分类情况以多数判决的规则决定测试样本的分类。算法的伪代码如下。

算法 1-1 K-近邻分类算法。

输入：训练数据集 Z；

可调参数 K；测试样例集的特征向量 X；

1： **for all** $x_{si} \in X$ **do**

2： 计算测试样本 x_{si} 到每个训练集 $(x_{ij}, y_{ij}) \in Z$ 之间的距离 d

3： 以距离为特征对训练集排序，得到距离最近的 K 个近邻集合 D_z

4： 多数表决 $y_{si} = \underset{c}{\operatorname{argmax}} \sum_{(x_{ij}, y_{ij}) \in D_z} I(c = y_{ij})$

5： **end for**

输出：实例 x_{si} 的类别 y_{si}

其中，c 是类别标号，$I(c = y_{ij})$ 为指示函数，如果参数为真，则值为 1，如果参数为假，则值为 0。

KNN 的优点在易于理解，模型使用高效(不代表存储量低，但是遍历计算复杂问题有很多工程方法解决)，有一定鲁棒性(K 值较大时明显抗噪能力强)；算法的不足在于大多数情况下并没有那么好的训练集(例如小簇对大簇存在分类劣势)。

1.1.5 补充说明

1. 欧几里得距离与余弦距离

设特征空间 χ 是 n 维实数向量空间 $\mathbf{R}^n$，$x_i, x_j \in \chi$，$x_i = (x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(n)})$，$x_j =$

$(x_j^{(1)}, x_j^{(2)}, \cdots, x_j^{(n)})$，则特征向量之间的欧几里得距离为

$$d(x_i, x_j) = \sqrt{\sum_{l=1}^{n} (x_i^{(l)} - x_j^{(l)})^2} \tag{1-2}$$

余弦距离为

$$\cos(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\boldsymbol{x}_i \cdot \boldsymbol{x}_j}{\| x_i \| \cdot \| x_j \|} \tag{1-3}$$

其中，欧几里得距离表达的是两向量的绝对距离，计算的是两向量中各维度的绝对差值。因此，计算欧几里得距离的时候要求各维度指标有相同的刻度级别，在使用之前需要对底层数据进行标准化处理。而余弦距离注重两向量方向上的差异，而非位置。

邻近度度量公式有许多，如皮尔逊相关系数、Jaccard 相关系数等，距离度量的类型的选取需要与数据类型相适应。各种类型的邻近度度量公式的区别及适用条件可见参考文献。

2. 距离加权表决

在多数表决中，K 个近邻用户对最终决策的贡献是一样的，这使得 K 值对决策结果很敏感，降低这种敏感的有效手段之一是使用距离加权表决。其形式化如下：

$$y = \underset{c}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} w_i \times I(c = y_i) \tag{1-4}$$

1.2 KNN 算法实现

本节讲述如何使用 Java 实现 K-近邻算法，并开发 KNN 算法的简单应用，以加深读者对构建 KNN 算法的三个主要步骤的理解。

1.2.1 简介

本算法的 Java 实现主要包括数据处理和算法模块。数据处理模块的主要内容有数据的加载及预处理、训练集和测试集的划分；算法模块的主要内容有计算欧几

里得距离、选取近邻数据及决策。下面将详细介绍 Java 类的设计情况。

算法设计流程图如图 1-1 所示。

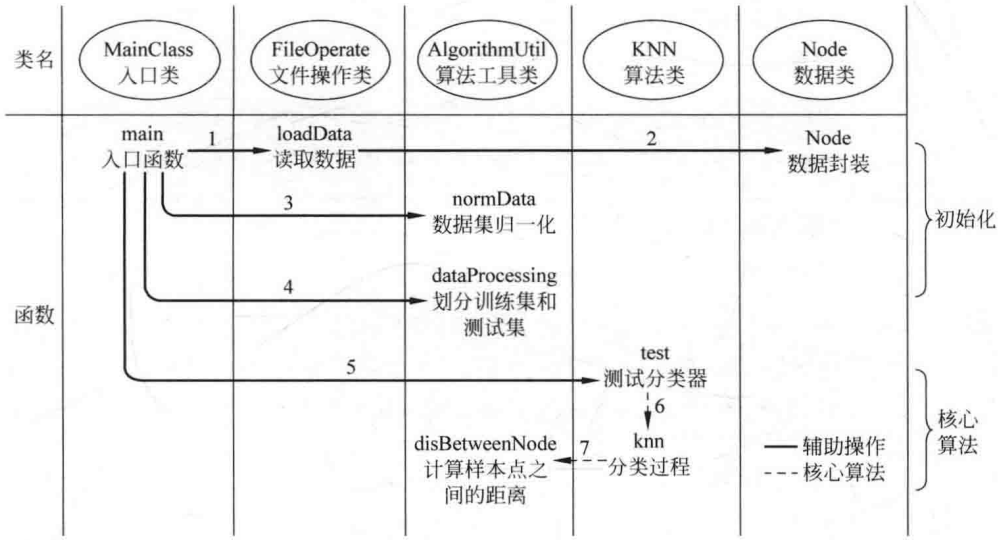

图 1-1 算法设计流程图

类名称及其描述如表 1-1 所示。

表 1-1 类名称及其描述

类 名 称	类 描 述

Node	(描述一个数据点)
	成员变量:
	`private ArrayList<Double> property;  //数据点的属性向量`
	`private String label;  //数据点的标签`

AlgorithmUtil	(集成算法所需要的工具类)
	函数:
	`/** 归一化数值 */`
	`Public static double normNum(double oldValue,double max, double K){…}`
	`/** 计算两个样本之间的邻近度(欧氏距离) */`
	`public static double disBetweenPoint(Node o1,Node o2){…}`
	`/** 将数据集归一化处理 */`
	`Public static void normData(ArrayList<Node> dataList){…}`
	`/** 划分训练集 */`
	`Public static Array<ArrayList<Node>> dataProcessing(ArrayList<Node> dataSetList, double trainRate){…}`