

# 机器学习观止

## 核心原理与实践

Machine Learning

林学森 著

清华大学出版社

北 京

## 内 容 简 介

本书在写作伊始,就把读者设想为一位虽然没有任何AI基础,但对技术本身抱有浓厚兴趣、喜欢“抽丝剥茧”、探究真相的“有识之士”。有别于市面上部分AI技术书籍从一开始就直接讲解各种“高深莫测”算法的叙述手法,本书尝试先从零开始构建基础技术点,而后“循序渐进”地引领读者前进,最终“直捣黄龙”,赢取最后的胜利。

全书据此分为5篇,共31章,内容基本覆盖了由AI发展历史、数学基础知识、机器学习算法等经典知识点以及深度学习、深度强化学习等较新理论知识所组成的AI核心技术。同时注重“理论联系实践”,通过多个章节重点介绍了如何在工程项目中运用AI来解决问题的诸多经验以及相应的模型算法,以期让读者既能享受到“知其所以然”的乐趣,还能体会到“知其然”的轻松和愉悦。

本书适合对AI感兴趣的读者阅读,从事AI领域工作的研究人员、工程开发人员、高校本科生和研究生都可以从本书中学到机器学习的相关知识。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。举报:010-62782989, beiqinquan@tup.tsinghua.edu.cn。

### 图书在版编目(CIP)数据

机器学习观止:核心原理与实践/林学森著. —北京:清华大学出版社, 2021.3

ISBN 978-7-302-55744-9

I. ①机… II. ①林… III. ①机器学习 IV. ①TP181

中国版本图书馆CIP数据核字(2020)第104785号

责任编辑:李万红 施 猛

封面设计:熊仁丹

版式设计:方加青

责任校对:马遥遥

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](mailto:c-service@tup.tsinghua.edu.cn)

质 量 反 馈: 010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

印 装 者: 三河市铭诚印务有限公司

经 销: 全国新华书店

开 本: 210mm×285mm

印 张: 48.25

字 数: 1460千字

版 次: 2021年3月第1版

印 次: 2021年3月第1次印刷

定 价: 168.00元

---

产品编号: 086756-01



## 前言

这本书前后写了五年，终成稿付梓。

过去五年是以机器学习，特别是深度学习为代表的人工智能技术爆炸性增长的一段关键时期。在本书写作期间，“深度学习三巨头”Yoshua Bengio、Geoffrey Hinton和Yann LeCun共同获得了图灵奖，更是把本轮人工智能浪潮推向了巅峰。可以说，我们大家都在见证，并将在未来几十年里持续见证人工智能的各种突破性研究成果，以及它们在产业界各个角落的落地开花。

那么，本轮人工智能(不论以何种分支为载体)的兴起，真的可以如人类已经经历的几次工业革命一样，给整个社会带来颠覆性的影响吗？

提出这种疑问的人，可能对人工智能在过去几十年的“风雨飘摇”还记忆犹新。

人工智能起源于20世纪50年代，除了Alan Mathison Turing提出的著名的“图灵测试”外，还有几个重要事件，比如：世界上第一台神经网络计算机的建造；1956年在美国Dartmouth College召开的会议上，人们第一次提出人工智能(Artificial Intelligence, AI)术语，并将其确立为一门独立的学科；世界上第一座人工智能实验室，即MIT AI LAB的建立等。

随后的二十年是人工智能的第一次高峰，人们认为只需要较短的时间，机器就会取代人类完成很多工作。然而事实证明，当时的这个观点显然过于乐观了——由于AI研究长期的“雷声大雨点小”，于是1973年一份著名的《莱特希尔报告》(Lighthill Report)就成了压死骆驼的最后一根稻草，直接将AI逼进了第一个寒冬(1974—1980年)。

不过这似乎并没有完全浇灭人们对于人工智能的“希望之火”——果然，几年后的1980年，AI又以“专家系统”重出江湖了，而且这一次还是伴随着商业机遇出现的(据悉1985年AI市场规模已经达到了10亿美元级别)。人们在看到新希望的同时，也催生了AI的再次繁荣。

然而好景不长，AI的再次复出仍然没有能够达到人们的预期。正所谓“希望越大，失望越大”，于是，伴随着LISP机(LISP machine)市场的萎缩，AI又迅速地成了一个“弃儿”(1987—1993年)。

至此，AI实际上已经是“四十岁”的“中年人”了。人们常说“四十不惑”，或许AI在经历了这样曲折的“人生阅历”后，也可以逐步成熟稳重起来。事实证明，AI也确实在“凤凰涅槃”，1993年之后的它逐步在多个领域展开了拳脚，并且在学术界和工业界都取得了不小的成绩。特别是步入21世纪以后，计算机运算能力的指数级增长以及大数据的繁荣使得AI“内功”得到了极大的增强，于是以深度学

习为代表的人工智能在多年的理论沉淀后终于得以“厚积薄发”。毫不夸张地说，人工智能已经成为当今最具热度的一个研究方向之一。业界预判AI将是激发新一轮产业革命的突破性技术，如图1所示。

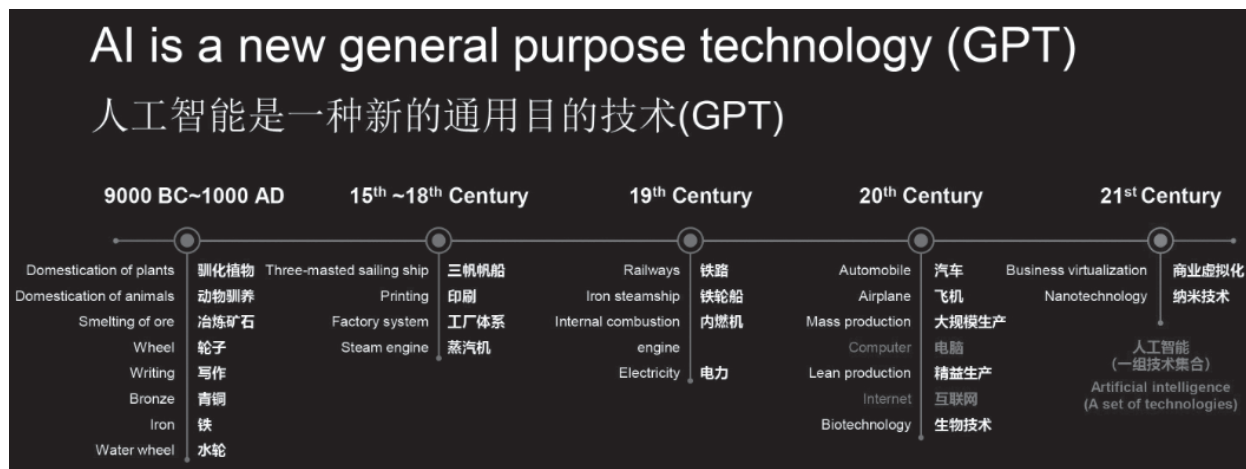


图1 业界预判AI将是激发新一轮产业革命的突破性技术

(材料引用自：华为2018年全联接大会keynote)

那么，人工智能是否会进入下一个“寒冬”，或者说它在本轮热潮中还可以走多远呢？

相信没有人可以给出准确的答案，毕竟我们谁也不可能预知未来。但是，这并不应该成为大家对它热爱与否的前提条件，因为我们相信：

**人工智能必将取得全面成功。**

或许还需要10年、50年甚至更长的时间，但这个趋势是不可逆转的。与其去纠结人工智能寒冬是否会到来，不如静下心来好好研究一下AI技术是否可以帮助我们解决一些实实在在的工作中的问题。

总的来说，人工智能的发展历史还在继续，而我们这一代人则有幸正在参与它的谱写过程。这也是本书的创作目的之一，希望通过由浅入深的叙述方式，来带领更多的AI爱好者进入这个领域，并在可能的情况下为AI的发展“添砖加瓦”。

## 致谢

感谢清华大学出版社的编辑，你们的专业态度和处理问题的人性化，是所有作者的“福音”。

感谢我的家人林进跃、张建山、林美玉、杨惠萍、林惠忠、林月明，没有你们的鼓励与理解，就没有本书的顺利出版。

感谢我的妻子张白杨的默默付出，是你工作之外还无怨无悔地照顾着我们可爱的宝宝，才让我有充足的时间和精力来写作。

感谢所有读者的支持，是你们赋予了我写作的动力。

编者  
2021 年元旦



## 目 录

### 机器学习基础知识篇

第1章 人工智能概述 .....	002
1.1 人工智能的定义 .....	002
1.2 人工智能发展简史 .....	003
1.2.1 史前文明，曙光初现(1956年前) .....	004
1.2.2 初出茅庐，一战成名(1956—1974年) .....	008
1.2.3 寒风凛冽，首次入冬(1974—1980年) .....	010
1.2.4 卷土重来，威震八方(1980—1987年) .....	010
1.2.5 失望弥漫，再度入冬(1987—1993年) .....	012
1.2.6 重出江湖，渐入佳境(1993年至今) .....	013
1.3 人工智能经典流派 .....	016
1.3.1 符号主义 .....	018
1.3.2 连接主义 .....	019
1.3.3 行为主义 .....	023
1.3.4 贝叶斯派 .....	026
1.4 人工智能与机器学习 .....	027
1.5 如何选择机器学习算法 .....	029
1.5.1 没有免费的午餐理论 .....	030
1.5.2 Scikit Learn小抄 .....	031
1.5.3 Microsoft Azure小抄 .....	032
1.6 机器学习的典型应用场景 .....	032
1.6.1 计算机图像领域 .....	035
1.6.2 自然语言处理简述及其应用 .....	036
1.6.3 制造业中的预测性维护 .....	038

1.6.4	软件自动化开发和测试	042
1.7	本书的组织结构	043
第2章	机器学习中的数学基础	045
2.1	微分学	045
2.1.1	链式求导法则	045
2.1.2	对数函数求导	045
2.1.3	梯度和梯度下降算法	046
2.2	线性代数	047
2.2.1	向量	047
2.2.2	矩阵拼接	052
2.2.3	特征值和特征向量	057
2.2.4	仿射变换	059
2.3	概率论	060
2.3.1	概率分布	061
2.3.2	先验/后验概率	062
2.3.3	最大似然估计	063
2.3.4	贝叶斯法则	064
2.4	统计学	065
2.4.1	数据的标准化和归一化	065
2.4.2	标准差	066
2.4.3	方差和偏差	066
2.4.4	协方差和协方差矩阵	067
2.5	最优化理论	068
2.5.1	概述	068
2.5.2	函数等高线	070
2.5.3	拉格朗日乘子法	071
2.5.4	拉格朗日对偶性	074
2.5.5	KKT	079
2.6	其他	088
2.6.1	训练、验证和测试数据集	088
2.6.2	过拟合和欠拟合	090
2.6.3	奥卡姆的剃刀	092
2.6.4	信息熵	093
2.6.5	IOU	094
2.6.6	NMS	095
2.6.7	Huffman树	096
第3章	机器学习模型的度量指标	099
3.1	Precision、Recall和mAP	099
3.2	$F_1$ Score	101
3.3	混淆矩阵	102

3.4 ROC	103
3.5 AUC	105
3.6 PRC	107
3.7 工业界使用的典型AI指标	108

## 经典机器学习篇

第4章 回归算法	112
4.1 回归分析	112
4.2 线性回归	112
4.2.1 线性回归的定义	112
4.2.2 线性回归的损失函数	113
4.2.3 线性回归范例	113
4.3 逻辑回归	115
4.3.1 逻辑回归——二分类	115
4.3.2 逻辑回归——多分类及Softmax	119
第5章 K-NN算法	122
5.1 K-NN概述	122
5.2 K-NN分类算法	123
5.3 K-NN回归算法	124
5.4 K-NN的优缺点	125
5.4.1 K-NN的优点	125
5.4.2 K-NN的缺点	126
5.5 K-NN工程范例	126
第6章 k-means	129
6.1 k-means概述	129
6.2 k-means核心算法	129
6.3 k-means算法的优缺点	131
6.3.1 k-means算法的优点	131
6.3.2 k-means算法的缺点	131
6.4 k-means工程范例	132
第7章 朴素贝叶斯	135
7.1 朴素贝叶斯分类算法	135
7.2 朴素贝叶斯的实际应用	137
第8章 决策树和随机森林	141
8.1 决策树	141
8.1.1 决策树的主要组成元素	141
8.1.2 决策树的经典算法	141
8.1.3 决策树的优缺点	145
8.1.4 决策树的过拟合和剪枝	145

8.2 随机森林	146
第9章 支持向量机	149
9.1 SVM可以做什么	149
9.2 SVM的数学表述	151
9.2.1 决策面的数学表述	151
9.2.2 分类间隔的数学表述	152
9.2.3 比较超平面的数学公式	153
9.2.4 最优决策面的数学表述	159
9.3 SVM相关的最优化理论	160
9.3.1 感知机学习算法	160
9.3.2 SVM最优化问题	173
9.4 硬间隔SVM	174
9.5 软间隔SVM	177
9.6 核函数技巧	182
9.7 多分类SVM	187
9.8 SVM实践	193
第10章 PCA降维	196
10.1 降维概述	196
10.2 PCA降维实现原理	197
10.2.1 PCA的直观理解	197
10.2.2 PCA的理论基础——最大方差理论	199
10.2.3 PCA的核心处理过程	199
10.3 PCA实例	200
第11章 集成学习	202
11.1 集成学习概述	202
11.2 集成学习架构	203
11.2.1 聚合法	203
11.2.2 提升法	204
11.2.3 堆叠法	205
11.3 典型的集成方法	206
11.3.1 平均法	206
11.3.2 投票法	207
11.3.3 学习法	208

## 深度学习进阶篇

第12章 深度神经网络	212
12.1 神经元	212
12.2 激活函数	214
12.2.1 Sigmoid	214



12.2.2	tanh .....	216
12.2.3	ReLU .....	217
12.2.4	Leaky ReLU .....	218
12.2.5	ReLU的其他变种 .....	219
12.2.6	激活函数的选择 .....	220
12.3	前向传播和后向传播算法 .....	220
12.4	损失函数 .....	224
12.4.1	分类场景 .....	225
12.4.2	回归场景 .....	228
12.4.3	其他任务类型的损失函数 .....	230
第13章	卷积神经网络 .....	232
13.1	CNN发展历史简述 .....	232
13.2	CNN的核心组成元素 .....	233
13.2.1	卷积层 .....	233
13.2.2	池化层 .....	235
13.2.3	全连接层 .....	236
13.2.4	Softmax层 .....	237
13.3	CNN经典框架 .....	237
13.3.1	LeNet .....	237
13.3.2	AlexNet .....	238
13.3.3	VGG .....	240
13.3.4	GoogLeNet .....	242
13.3.5	ResNet .....	245
13.4	CNN的典型特性 .....	249
13.4.1	CNN位移不变性 .....	250
13.4.2	CNN尺度不变性 .....	252
13.4.3	CNN旋转不变性 .....	253
13.4.4	CNN视角不变性 .....	255
第14章	RNN与LSTM .....	256
14.1	RNN .....	256
14.2	RNN的多种形态 .....	257
14.3	RNN存在的不足 .....	258
14.4	LSTM .....	259
14.5	LSTM核心框架 .....	259
14.5.1	遗忘门 .....	261
14.5.2	输入门 .....	261
14.5.3	输出门 .....	262
14.6	GRU .....	263
第15章	深度强化学习 .....	265
15.1	强化学习和MDP .....	265

15.1.1	强化学习的基础概念	265
15.1.2	MDP	266
15.1.3	强化学习的核心三要素	267
15.2	MDP问题的解决方案分类	268
15.3	基于模型的动态规划算法	269
15.4	基于无模型的强化学习算法	272
15.4.1	蒙特·卡罗强化学习算法	272
15.4.2	时间差分算法	275
15.5	DQN	278
15.6	基于策略的强化学习算法	280
15.6.1	有限差分策略梯度	283
15.6.2	蒙特·卡罗策略梯度	283
第16章	MCTS	285
16.1	MCTS概述	285
16.2	MCTS算法核心处理过程	286
16.3	UCB和UCT	286
16.4	MCTS实例解析	288

## 机器学习应用实践及相关原理

第17章	数据集的建设	292
17.1	数据集建设的核心目标	292
17.2	数据采集和标注	294
17.2.1	数据从哪来	294
17.2.2	数据分布和多样性	296
17.2.3	如何扩大数据量	298
17.3	数据分析和处理	299
17.3.1	数据集分析的典型方法	299
17.3.2	标签类别合理性	301
17.3.3	数据清洗	303
第18章	CNN训练技巧	304
18.1	数据预处理	304
18.1.1	数据零中心化	304
18.1.2	数据标准化	306
18.1.3	尺寸调整	306
18.1.4	其他	307
18.2	数据增强	308
18.3	CNN核心组件择优	309
18.3.1	激活函数	309
18.3.2	超参数设定	309

18.4	参数初始化策略	310
18.4.1	全零初始化策略	310
18.4.2	随机初始化策略	311
18.4.3	采用预训练模型	319
18.5	模型过拟合解决方法	319
18.5.1	正则化	319
18.5.2	批标准化	320
18.6	模型的可解释性	328
18.6.1	反卷积网络	331
18.6.2	类别激活映射	334
18.6.3	LIME	339
18.6.4	可视化集成工具Darkon	344
18.7	Auto ML	346
第19章	CV和视觉识别经典模型	348
19.1	CV发展简史	348
19.2	视觉识别概述	353
19.3	R-CNN	359
19.3.1	R-CNN简述	359
19.3.2	R-CNN中的候选区域	360
19.3.3	R-CNN算法处理流程	361
19.4	Fast R-CNN	364
19.5	SPP-Net	365
19.5.1	空间金字塔池化	366
19.5.2	特征图和原图的映射关系	367
19.5.3	基于SPP-Net的目标识别	367
19.6	Faster R-CNN	368
19.6.1	Faster R-CNN简述	368
19.6.2	候选区域网络	370
19.6.3	分类器和边框回归	375
19.7	YOLO	375
19.8	SSD	383
19.8.1	SSD的网络框架	383
19.8.2	SSD的应用推理过程	384
19.8.3	SSD的性能评估和缺点	388
19.9	不基于CNN来实现目标识别	390
19.9.1	相关的OpenCV函数	390
19.9.2	利用OpenCV识别形状物体范例	394
第20章	自然语言处理和CNN	397
20.1	NLP简述	397
20.2	NLP发展历史	399

20.3	自然语言基础	400
20.4	词的表达方式	403
20.5	自然语言模型	405
20.5.1	基于N-Gram的语言模型	406
20.5.2	基于神经网络的语言模型——经典NNLM	409
20.5.3	基于神经网络的语言模型——NNLM的改进者CBOW模型	411
20.5.4	基于神经网络的语言模型——NNLM的改进者Skip-gram模型	414
20.6	word2vec	416
20.6.1	word2vec简介	416
20.6.2	word2vec源码与编译	417
20.6.3	word2vec使用范例	418
20.7	常用语料库	420
20.8	NLP应用：文本分类	424
20.8.1	传统的文本分类方法	424
20.8.2	基于深度学习的文本分类方法	425
第21章	自然语言处理和CNN	430
21.1	应用程序场景识别背景	430
21.2	特征向量	431
21.3	数据采集	432
21.4	算法模型	433
21.5	落地应用	433
第22章	软件自动修复	436
22.1	什么是软件自动修复	436
22.1.1	软件自动修复的定义	436
22.1.2	软件自动修复的价值	437
22.2	软件自动修复基础知识	437
22.2.1	软件自动修复技术分类	437
22.2.2	软件自动修复基础概念	439
22.3	阶段1：缺陷定位	441
22.3.1	基于程序频谱的缺陷定位	443
22.3.2	SFL中测试套件的构造	447
22.3.3	SFL中程序频谱的构造	451
22.4	阶段2：补丁生成	458
22.4.1	基于搜索的补丁生成和自动修复	459
22.4.2	基于模板的补丁生成和自动修复	460
22.5	APR领域经典框架	462
22.5.1	Facebook SapFix	463
22.5.2	Microsoft DeepCoder	465
22.5.3	GenProg	474

第23章 基于强化学习的经典应用——AlphaGO .....	479
23.1 AlphaGO简述 .....	479
23.2 AlphaGO核心原理 .....	480
23.3 策略网络 .....	481
23.4 估值网络 .....	483
23.5 MCTS .....	483

## 机器学习平台篇

第24章 分布式机器学习框架基础知识 .....	488
24.1 分布式机器学习核心理念 .....	488
24.2 GPU硬件设备 .....	491
24.2.1 GPU架构 .....	492
24.2.2 GPU的共享访问 .....	494
24.3 网络标准 .....	498
24.3.1 Ethernet .....	498
24.3.2 InfiniBand .....	499
24.4 分布式通信框架 .....	500
24.4.1 MPI .....	500
24.4.2 P2P和聚合通信 .....	503
24.4.3 NCCL .....	505
24.4.4 NV-Link .....	508
24.4.5 RDMA .....	510
24.5 经典分布式ML框架Caffe-MPI .....	511
第25章 Tensorflow .....	514
25.1 Tensorflow安装过程 .....	514
25.2 Tensorflow基础知识 .....	516
25.2.1 Tensorflow核心概念 .....	516
25.2.2 Tensorflow模型/数据的保存和恢复 .....	519
25.2.3 Tensorflow模型fine-tuning .....	523
25.2.4 Tensorflow模型调试 .....	526
25.2.5 Tensorflow的多语言支持 .....	528
25.2.6 可视化利器TensorBoard .....	529
25.3 Tensorflow分布式训练 .....	533
25.3.1 Tensorflow的分布式原理 .....	533
25.3.2 单机多GPU下的并行计算 .....	535
25.3.3 多机多GPU下的分布式计算 .....	542
25.4 Tensorflow分布式部署 .....	549
25.4.1 Tensorflow Serving概述 .....	549
25.4.2 基于GPU的Tensorflow Serving .....	549
25.4.3 Tensorflow Serving的核心概念 .....	552

25.4.4	Tensorflow模型分布式部署实例	553
25.5	Tensorflow范例解析	560
25.6	Tensorflow的“变种”	563
25.6.1	Tensorflow Lite	563
25.6.2	Tensorflow RS	565
第26章	Caffe	568
26.1	Caffe的安装	568
26.1.1	Ubuntu下安装Caffe	568
26.1.2	Windows下安装Caffe	572
26.2	Caffe支持的数据集格式	587
26.2.1	LevelDB	587
26.2.2	LMDB	590
26.2.3	数据库的生成	592
26.3	Caffe中的网络模型构建	594
26.4	Google Protocol Buffer	598
26.5	Caffe2源码结构	600
26.6	Caffe工程范例	601
26.7	Caffe中的Model Zoo	607
第27章	scikit-learn	609
27.1	scikit-learn的安装	610
27.2	scikit-learn中的机器学习算法	610
27.2.1	分类	610
27.2.2	回归	611
27.2.3	聚类	611
27.2.4	降维	612
27.3	scikit-learn中的Model selection	613
27.3.1	网络搜索	613
27.3.2	交叉验证	616
27.3.3	度量标准	616
27.4	scikit-learn中的预处理	619
27.4.1	数据标准化等预处理	619
27.4.2	数据特征提取预处理	621
第28章	主流AI云平台	628
28.1	Microsoft OpenPAI	628
28.2	Google Cloud	631
28.3	Baidu	631
28.3.1	百度AI云服务	632
28.3.2	PaddlePaddle	636
28.4	Alibaba	637
28.4.1	阿里飞天平台	638

28.4.2	MaxCompute平台	639
28.4.3	PAI	640
<b>第29章</b>	<b>图像处理基础</b>	<b>650</b>
29.1	光、色彩和人类视觉系统	650
29.2	图像的颜色模型	653
29.3	图像的基本属性	655
29.3.1	灰度值	655
29.3.2	亮度	656
29.3.3	对比度	657
29.3.4	色相	658
29.3.5	饱和度	658
29.4	图像特征	659
29.4.1	颜色特征	659
29.4.2	纹理特征	660
29.4.3	形状特征	661
29.5	图像的典型特征描述子	661
29.5.1	LBP	661
29.5.2	HOG	677
29.5.3	Haar-like 特征	681
29.5.4	图像的傅里叶变换	686
29.6	图像处理实例(图像质量检测)	690
<b>第30章</b>	<b>程序切片技术</b>	<b>693</b>
30.1	程序切片综述	693
30.2	程序切片基础知识	695
30.2.1	控制流图	695
30.2.2	控制流分析	699
30.2.3	数据流	706
30.3	静态切片技术	715
30.3.1	基本定义	715
30.3.2	静态切片算法	717
30.4	动态切片技术	721
30.4.1	动态切片基本概念	721
30.4.2	动态切片算法概述	723
30.4.3	基于PDG的动态切片算法	723
<b>第31章</b>	<b>业界主流数据集分析</b>	<b>726</b>
31.1	ImageNet简述	726
31.2	ImageNet的构建逻辑	726
31.3	ImageNet数据源的选择与处理	730
31.4	ImageNet的下载	733
	参考文献	736

## 5.1 K-NN概述

K-NN的全称为K-Nearest Neighbors(中文常被译作“K最近邻算法”),是模式识别领域应用相当广泛的一个算法。K-NN经典算法起源于20世纪70年代,它既可以用于分类任务,也能够支持回归任务。K-NN在处理这两种情况时的算法输入都是一样的,即包含特征空间的 $k$ 个最接近的训练样本,只不过它们在输出结果上有所差异。

### 1. K-NN分类

当K-NN用于分类任务时,一个对象 $A$ 的分类结果取决于训练集中与它最相近的 $k$ 个元素的投票结果(例如少数服从多数)。如果 $k$ 取值为1,那么意味着只要找寻与 $A$ 最相近的训练样本就可以确定它的类别,如图5-1所示。

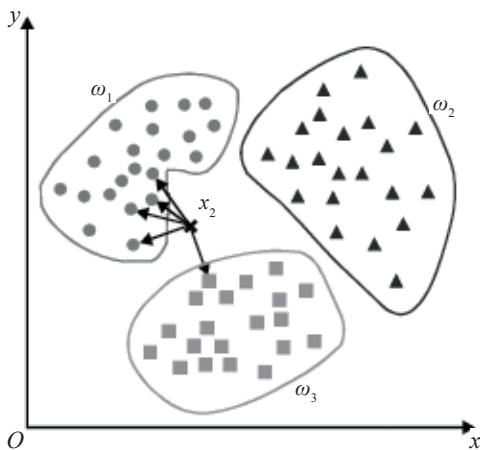


图5-1 K-NN分类图示

### 2. K-NN回归

当K-NN用于回归任务时,输出结果代表的是该对象 $A$ 的属性值(取与 $A$ 最相近的 $k$ 个样本的平均值),如图5-2所示。

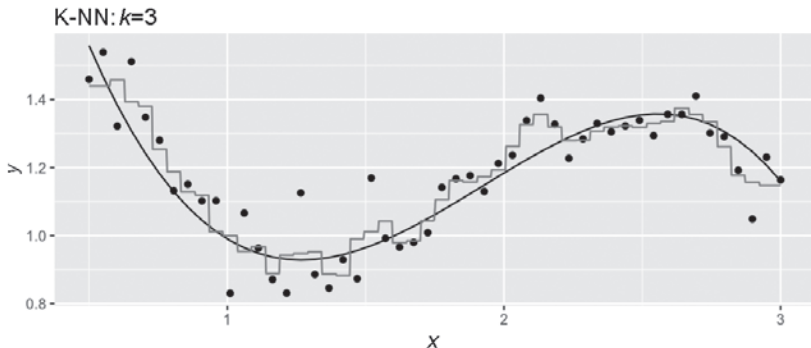


图5-2 K-NN回归

K-NN的另一个显著特点是它不需要训练,所有计算(分类/回归)都是在应用推理的时候才会执行,因而也被人们归于惰性学习法(Lazy Learning)——与之相对的是急性学习法(Eager Learning)。后者是指先利用训练数据进行训练得到一个目标函数,然后应用推理时只需利用训练好的函数进行决策。SVM等算法就属于这种学习方式。

这两种学习方法的区别如下。



- (1) 急性学习法需要基于所有样本开展训练过程，因而训练的时间较长，但推理决策时间较短。
- (2) 惰性学习法虽然在做最终决策时通常只用到了局部的几个样本，但由于它需要计算推理对象与所有样本之间的距离，复杂度还是达到了 $O(n)$ 。所以惰性学习法不仅需要较大的存储空间，而且决策过程比较慢(没有训练过程)。

## 5.2 K-NN分类算法

K-NN算法本身并不复杂，它主要由以下三个核心要素组成。

- (1)  $k$ 的取值。
- (2) 近邻算法。
- (3) 投票机制。

接下来围绕上述三个核心点来展开分析。

### 1. $k$ 的取值

理论上， $k$ 可以取合法范围内的任何数值——但需要思考的是， $k$ 取不同值会不会影响最终的结果呢？答案是肯定的，而且影响很大。

例如，如图5-3所示的经典案例。

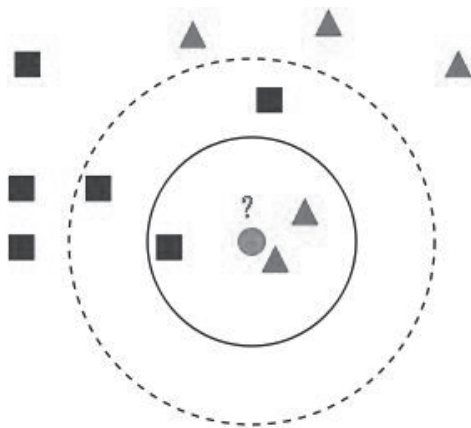


图5-3  $k$ 的取值对结果的影响示例

在图5-3的范例中，当 $k$ 取不同值时，结果如下。

(1)  $k=3$ 时：不难发现，此时中间的圆圈与两个三角形及一个正方形最近，所以如果按照投票法，它应该归属于三角形类别。

(2)  $k=5$ 时：此时圆圈与两个三角形及三个正方形最近，因而如果按照投票法，它就变成归属于正方形。

对于 $k$ 的取值，我们可以思考：

(1) 因为到目前为止还没有成熟的理论来支撑 $k$ 的选值过程，所以一般的做法是：采用交叉验证的方式来得到一个理论上最佳的 $k$ 值(类似于模型训练)。

(2)  $k$ 值较小时。

$k$ 值越小，代表参与最终决策的样本数量越少。这种情况下的训练误差可能会较小，但泛化误差会增大，或者说容易发生拟合。

(3)  $k$ 值较大时。

$k$ 值越大，代表参加最终决策的样本数量越多。其优点是可以减少噪声的影响，因而泛化误差较

小，但缺点是训练误差会增大。

(4)  $k$ 等于样本数 $n$ 时。

这是一种极端的情况，此时相当于没有分类。

(5) 根据实践经验来选取 $k$ 值。

从项目实践经验来看， $k$ 的取值一般都是： $k < \sqrt{n}$ ， $n$ 为样本数量。

## 2. 近邻算法

K-NN的可选近邻算法有很多种，包括但不限于：

1) 曼哈顿(Manhattan)距离

典型公式如下。

$$d_{12} = \sum_{k=1}^n |x_{1k} - x_{2k}|$$

2) 欧几里得(Euclidean)距离

典型公式如下。

$$d_{12} = \sqrt{\sum_{k=1}^n (x_{1k} - x_{2k})^2}$$

3) 马氏(Mahalanobis)距离

典型公式如下。

$$D(X_i, X_j) = \sqrt{(X_i - X_j)^T S^{-1} (X_i - X_j)}$$

其中， $S$ 为协方差矩阵。

4) 闵可夫斯基(Minkowsky)距离

典型公式如下。

$$d_{12} = \sqrt[p]{\sum_{k=1}^n |x_{1k} - x_{2k}|^p}$$

其中，“欧几里得距离”是K-NN中最常用的一种距离度量方法。

## 3. 投票机制

K-NN的投票机制也有多种选择，例如：

1) 少数服从多数

这是最简单的一种投票方式，但是它没有考虑 $k$ 个最近邻元素之间的“远近”差异，因而在效果上并不是最佳的。

2) 加权投票

和上述“人人平等”的投票方式不同，加权投票针对不同近邻元素的差异性进行了考量——例如，按照它们与被预测对象的距离远近分别赋予相应的权重，保证距离近的元素对最终决策的影响力更大一些，反之距离远的作用则被有目的地降低了。

综合来看，距离加权的投票方法是K-NN中采用最广泛的其中一种投票机制。

## 5.3 K-NN回归算法

K-NN回归算法和K-NN分类算法大致相同，接下来结合一个范例来进行实际的讲解。

假设要帮客户出租一套3室的房子，那么定价多少合适呢？

K-NN可以回答这个问题——可以基于市面上相同或者相近户型的房子的出租价格，来得出一个相对合理的答案，如表5-1所示。

表5-1 市面房租价格

序号	户型	出租价格/元
1	3室	1000
2	3室	1200
3	3室	1150
4	3室	1250
5	5室	2000

假如选择欧氏距离作为度量标准，那么：

1. 对于3室样本

此时欧氏距离公式中的 $n=1$ ，即

$$\begin{aligned}d_{12} &= \sqrt{\sum_{k=1}^n (x_{1k} - x_{2k})^2} \\&= x_1 - x_2 \\&= 3 - 3 \\&= 0\end{aligned}$$

2. 对于5室样本

同理，可得 $d=2$ 。

再假设 $k$ 的取值为4，换句话说，我们考虑的是4个最相近的房租样本，那么按照K-NN的算法思想可以得到待出租的3室的价格myPrice：

$$\begin{aligned}\text{myPrice} &= (\text{price1} + \text{price2} + \text{price3} + \text{price4}) / 4 \\&= (1000 + 1200 + 1150 + 1250) / 4 \\&= 1150\end{aligned}$$

可以看到上述计算过程中，使用的是针对 $k$ 个样本取均值的方式——不过和K-NN分类算法中的加权投票法类似，我们在回归场景中也可以考虑基于样本的差异性来区分对待，从而更好地发挥出样本数据的价值。

5.4 K-NN的优缺点

K-NN虽然是一个经典的算法，但它既有优点，同时也有不足之处。

5.4.1 K-NN的优点

K-NN算法的主要优点如下。

- (1) 算法简单，易于理解，易于实现。
- (2) 不需要训练过程。
- (3) 预测精度较高。
- (4) 通常对噪声数据不是特别敏感。

(5) 针对多分类问题(即对象具有多个类别标签的情况), K-NN算法的表现比较突出, 甚至超过SVM等算法。

### 5.4.2 K-NN的缺点

K-NN算法的主要缺点如下。

(1) 属于惰性学习法的一种。在前面已经分析过, 这种类型的算法虽然在做最终决策时只用到了局部的几个样本, 但由于它对每一个待分类的样本都需要计算其到全体样本数据的距离, 才能获取 $k$ 个最近邻点, 所以整体复杂度还是 $O(n)$ 。在实践中, 针对这一问题的一个常用解决方法是提前减除掉对分类结果作用不大的部分样本。

(2) 不仅决策过程比较慢(虽然理论上没有训练过程), 而且空间开销也较大。

(3) 可解释性较差。

(4) 当样本不均衡时(即归属于某些类别的样本数量很大, 而其他类别样本数量又很小的情况), 针对新样本的预测有可能出现偏差——因为与该样本最近邻的 $k$ 个数据中, 样本数量大的类别很可能会占绝对多数。

(5) 向量的维度越高, 那么欧式距离用于计算样本差异的能力就越弱, 由此可能导致预测结果的不准确性。

当然, K-NN的上述这些缺点并非完全无法解决。只要做到“知己知彼”, 那么就有可能在实际项目中采取有效的手段来规避或者减少这些缺点所带来的影响。

## 5.5 K-NN工程范例

本节中将通过scikit-learn的一个范例来讲解如何使用K-NN算法来解决一些实际问题。

scikit-learn框架下, 与K-NN算法相关的包是sklearn.neighbors, 这个包中不仅包含K-NN的经典实现, 还包含它的多种扩展实现, 如图5-4所示。

<code>neighbors.BallTree</code>	BallTree for fast generalized N-point problems
<code>neighbors.DistanceMetric</code>	DistanceMetric class
<code>neighbors.KDTree</code>	KDTree for fast generalized N-point problems
<code>neighbors.KernelDensity ([bandwidth, ...])</code>	Kernel Density Estimation
<code>neighbors.KNeighborsClassifier ([...])</code>	Classifier implementing the k-nearest neighbors vote.
<code>neighbors.KNeighborsRegressor ([n_neighbors, ...])</code>	Regression based on k-nearest neighbors.
<code>neighbors.LocalOutlierFactor ([n_neighbors, ...])</code>	Unsupervised Outlier Detection using Local Outlier Factor (LOF)
<code>neighbors.RadiusNeighborsClassifier ([...])</code>	Classifier implementing a vote among neighbors within a given radius

图5-4 K-NN扩展实现

其中, KNeighborsClassifier的原型如下。

```
class sklearn.neighbors. KNeighborsClassifier (n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30,
p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs) ¶ [source]
```

接下来的代码范例中就是基于KNeighborsClassifier来实现的, 核心代码节选以及详细释义如下。

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
```

```

from sklearn import neighbors, datasets
n_neighbors = 15 ##k值选取15
#加载iris(鸢尾花)数据集
iris = datasets.load_iris()
#iris是多维数据集，我们只选取两个维度进行示例
X = iris.data[:, :2]
y = iris.target

```

该数据集是关于鸢尾花的，大致是如图5-5所示的样式。

	花萼长度	花萼宽度	花瓣长度	花瓣宽度	类别
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
5	5.4	3.9	1.7	0.4	0

图5-5 iris数据集

```

h = .02 #通过步进值产生大量被预测对象
#一共有三种类型的iris，因而需要创建三种颜色
cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF'])
cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF'])
for weights in ['uniform', 'distance']:

```

这两种weights(uniform和distance)，分别对应前面讲解的“少数服从多数”和“加权投票”两种机制。

```

#紧接着利用scikit-learn提供的KNeighborsClassifier来创建K-NN分类器
clf = neighbors.KNeighborsClassifier(n_neighbors, weights=weights)
##将前面的iris数据与分类器建立关联
clf.fit(X, y)
##结合步进值，产生大量的网格点
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                     np.arange(y_min, y_max, h))

##利用K-NN分类器对这些网格点执行预测
Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
#将预测结果保存下来
Z = Z.reshape(xx.shape)
plt.figure()

```

```
##根据预测结果，分配不同的颜色值，从而形成decision boundary
plt.pcolormesh(xx, yy, Z, cmap=cmap_light)
#绘制所有数据样本
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cmap_bold,
            edgecolor='k', s=20)

plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())

plt.title("3-Class classification (k = %i, weights = '%s')" % (n_neighbors,
weights))
plt.show()
```

最终结果如图5-6所示。

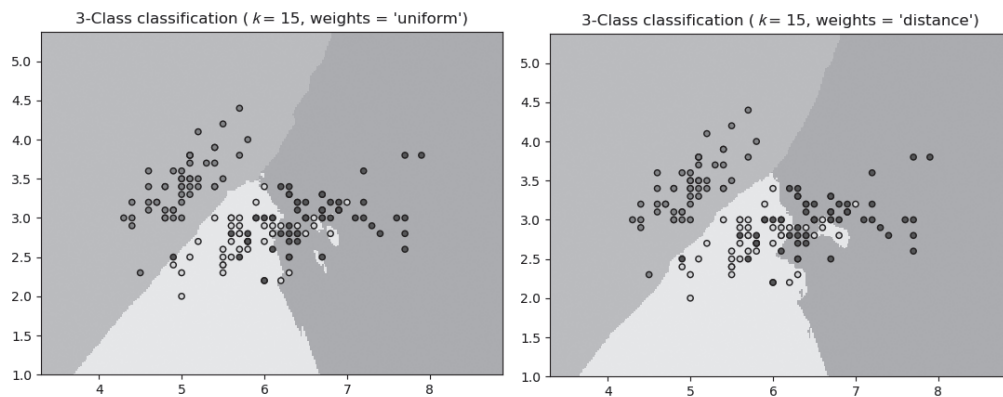


图5-6 基于K-NN算法的工程范例

读者可以仔细比较一下K-NN采用两种不同weights时在分类表现上的异同点。