



数据科学家成长之路

Python大战机器学习

数据科学家的第一个小目标

华校专 王正林 编著



中国工信出版集团



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn



数据科学家成长之路

Python大战机器学习

数据科学家的第一个小目标

华校专 王正林 编著

电子工业出版社

Publishing House of Electronics Industry

北京•BEIJING

内 容 简 介

数据科学家是当下炙手可热的职业，机器学习则是他们的必备技能。机器学习在大数据分析中居于核心地位，在互联网、金融保险、制造业、零售业、医疗等产业领域发挥了越来越大的作用且日益受到关注。

Python 是最好最热门的编程语言之一，以简单易学、应用广泛、类库强大而著称，是实现机器学习算法的首选语言。

本书以快速上手、四分理论六分实践为出发点，讲述机器学习的算法和 Python 编程实践，采用“原理笔记精华 + 算法 Python 实现 + 问题实例 + 代码实战 + 运行调参”的形式展开，理论与实践结合，算法原理与编程实战并重。

全书从内容上分为 13 章分 4 篇展开：第一篇：机器学习基础篇（第 1~6 章），讲述机器学习的基础算法，包括线性模型、决策树、贝叶斯分类、k 近邻法、数据降维、聚类 and EM 算法；第二篇：机器学习高级篇（第 7~10 章），讲述经典而常用的高级机器学习算法，包括支持向量机、人工神经网络、半监督学习和集成学习；第三篇：机器学习工程篇（第 11~12 章），讲述机器学习工程中的实际技术，包括数据预处理，模型评估、选择与验证等；第四篇：Kaggle 实战篇（第 13 章），讲述一个 Kaggle 竞赛题目的实战。

本书内容丰富、深入浅出，算法与代码双管齐下，无论你是新手还是有经验的读者，都能快速学到你想要的知识。本书可供为高等院校计算机、金融、信息、自动化及相关理工科专业的本科生或研究生使用，也可供对机器学习感兴趣的研究人员和工程技术人员阅读参考。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有，侵权必究。

图书在版编目（CIP）数据

Python 大战机器学习：数据科学家的第一个小目标 / 华校专，王正林编著. —北京：电子工业出版社，2017.3

（数据科学家成长之路）

ISBN 978-7-121-30894-9

I. ① P…II. ① 华…② 王…III. ① 软件工具—程序设计 IV. ① TP311.561

中国版本图书馆 CIP 数据核字（2017）第 022099 号

策划编辑：张月萍

责任编辑：徐津平

特约编辑：顾慧芳

印 刷：北京京科印刷有限公司

装 订：三河市良远印务有限公司

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱

邮编：100036

开 本：787×1092 1/16 印张：28

字数：716 千字

版 次：2017 年 3 月第 1 版

印 次：2017 年 5 月第 3 次印刷

定 价：69.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：（010）51260888-819 faq@phei.com.cn。

自序

本人华校专，性别男，爱好女^{^_^}，湖北黄冈人。2004 年我考入清华大学航天学院工程力学系，喜欢读书，在清华的四年我每年的成绩都是本系的 Top 1。在清华园里我养成了记笔记的习惯，大学四年我做了大量的学习笔记，这一良好的习惯一直坚持到现在，十多年的时间我已经记了三千多页的笔记，如图 1 和图 2 所示。

研究生阶段我被免试保送到国防科大计算机学院读研并入伍，研究生毕业之后一直在某部队工作至 2016 年。工作期间我阅读了大量的计算机书籍并编写了大量的代码，从操作系统底层开发到应用 App 开发。这个阶段是我从学生到工程师的一个转变阶段，也是我个人知识体系的建设阶段。对于不理解的内容我反复读、反复研究。记得学习《算法导论》的时候，我阅读了不下四轮，做了两轮笔记，并且仿照 C++ STL 的风格实现了其中的各种算法（算法导论的 C++ 实现我已经放在个人的 [github](#) 上）。

我个人超喜欢研究算法，也比较有优势。一是我数学能力比较强，作为曾经的清华学霸^{^_^}，我数学相关的课程平均分不低于 95 分（我本科四年的平均学分积不低于 90 分）。另一方面是我编程功底比较强，尤其是精通 C/C++/Python 三门语言。在自学机器学习时，理论方面我结合了斯坦福大学的机器学习公开课，李航老师的《统计学习方法》和周志华老师的《机器学习》等课程，实践方面我使用 Python 的 scikit-learn 包提供的 API 函数，包里面所涵盖的算法接口非常全面，更令人振奋的是，其用户手册写得非常好，我发现这是一条快捷的学习路径。

机器学习是一门理论与实践结合非常紧密的学科，理论提供了各种算法处理问题的边界，即有的算法适合处理问题 A，不适合问题 B；而另外一些算法适合处理问题 B 不适合处理问题 A。如果不懂得理论，那么对于某个具体问题，你就完全不知道应该采用哪种算法，以及当你采用了某个算法时各类超参数的物理意义。如果没有扎实的写代码实践，那么你可能采用了一个看起来很美好的算法，但是实际操作中因为各种条件不满足，最后要么预测性能很差，要么运行时间能让你崩溃，停留在“看上去很美”的尴尬状态。

2016 年，我顺利从部队退役，一次在北京旅游时，抱着试试的态度，我轻松拿到了阿里的算法 offer，2017 年年初，我在美丽的杭州入职！

学习机器学习绝不是“吃饭，睡觉，打豆豆”就能搞定，我觉得一定要做好三件事：看书、做笔记、编程，我已经验证了，该你了！

在机器学习的路上，我们一起同行！

2017 年 1 月，入职日于杭州阿里总部大楼前

= 1

函数的高级特性

1. 在 Python 中, 函数是递归通常比 for 循环要慢且空间消耗大, 但递归的优点是可以遍历任意形状的结构。
2. Python 函数是对象, 自身存储在内存块中, 它可以自由地传递与引用。
 - ① 函数对象支持一个特殊操作: 可以带括号及参数列表调用。
 - ② 我们可以通用地检查函数对象的属性。

如 `func.__name__`, `func.__code__` ... 可以利用这些信息来管理函数。
 - ③ 可以给函数附加任意的用户自定义属性。

如 `func.count = 0` ... 这样的属性可以用来直接把状态信息附加到函数对象。类似于 C 语言的静态局部变量 `static`。
- ④ Python 3.0 中, 可以给函数对象附加注释。注释不做任何事情, 它是可选的, 只是附加在函数对象的 `--annotations--` 属性中。

图 1 《Python 学习手册》(中文第 4 版) 笔记摘选

10. 每对顶点间的最短路径:

a. 解法一: 对 N 个顶点, 每次选其中一个作为源点来运行单源最短路径算法, 则运行 N 次之后即可求解。共用 Bellman-Ford 算法, 则总时间为 $O(N^3E)$ 。

b. 本问题的数据描述:

邻接矩阵表示权值: $W = (w_{ij})_{n \times n}$, 其中 $w_{ij} = \begin{cases} 0 & \text{若 } i=j \\ \text{有边 } (i,j) \text{ 的权值} & \text{若 } i \neq j \text{ 且 } (i,j) \in E \\ \infty & \text{若 } i \neq j \text{ 且 } (i,j) \notin E \end{cases}$

结果输出为 $D = (d_{ij})_{n \times n}$, 其中 d_{ij} 是从 i 到 j 的最短路径权值。

结果前驱矩阵 $\Pi = (\pi_{ij})_{n \times n}$, 其中:

$\pi_{i,j} = \begin{cases} \text{NIL} & \text{若 } i=j \text{ 或 } i \text{ 到 } j \text{ 没有通路} \\ \text{从 } i \text{ 出发的最短路径上} & \text{若 } (i,j) \text{ 有通路} \\ \text{ } j \text{ 的前驱顶点} & \end{cases}$

从图 2 可知, Π 的每行导出图是以 i 为根的一棵最短路径树。

对于顶点 i 的前驱子图 $G_{\pi,i} = (V_{\pi,i}, E_{\pi,i})$, 其中:

$V_{\pi,i} = \{v \mid (v,i) \in E_{\pi,i}\}$, $E_{\pi,i} = \{(v,i) \mid (v,i) \in E \text{ 且 } \pi_{v,i} \neq \text{NIL}\}$

图 2 《算法导论》(中文第 2 版) 笔记摘选

前言

拥抱智能时代

“大家还没搞清 PC 时代的时候，移动互联网来了，还没搞清移动互联网的时候，大数据时代来了。”马云在 2013 年淘宝十周年晚会上的这句话，仿佛一下子拉开了大数据时代的序幕。

2016 年下半年，著名的计算机科学家，超级畅销书《浪潮之巅》和《数学之美》的作者吴军博士，携他的全新力作《智能时代：大数据与智能革命重新定义未来》，宣告“智能时代，未来已来”，智能时代到来了！

新的时代，需要新的技术；新的技术，需要新的人才。2016 年年底，全球最著名的管理咨询公司麦肯锡在《分析的时代：在大数据的世界竞争（*The Age of Analytics: Competing in a Data-Driven World*）》的报告里指出“在美国，2012 年到 2014 年数据科学家的平均工资每年平均增长约 16%，远远高于美国劳工部统计的不到 2% 的所有工种的名义工资平均增长率。预测每年数据科学专业的应届毕业生将增加 7%，然而高质量项目对于专业数据科学家的需求每年增加 12%，这使得缺口约为 25 万人……到 2018 年，美国在‘深度分析’人才方面将面临 14 万至 19 万的人才缺口；在‘能够分析数据帮助公司做出商业决策’方面将面临 150 万的人才缺口”。清华大学计算机系教授武永卫 2016 年 5 月透露了一组数据：未来 3~5 年，中国需要 180 万数据人才，但目前只有约 30 万人。

麦肯锡还为此时代把脉“大数据分析正在改变竞争的基础，领先的公司如苹果、谷歌、亚马逊、Facebook、微软、通用以及阿里巴巴集团用自己的优势建立了全新的商业模式，数字化平台的网络结果在一些市场导致了“赢家通吃”的局面。……数据正在被商业化，而价值很大可能属于稀缺数据的所有者、用独特方式将数据整合起来的玩家、以及提供有价值的数据分析的人。”

大数据时代，做大数据分析的人有了一个更“性感”的名字，叫做数据科学家（Data Scientist）。《哈佛商业评论》声称，21 世纪最富挑战的工作是数据科学家。时下最热门的职业是数据科学家，而不是传统的信息科学家，也不是大数据工程师。

在数据科学家必备的技能中，机器学习和 Python 应该是位列前五的两项。机器学习炙手可热，在互联网、金融保险、电商、电信、制造业、零售业、医疗等产业领域发挥了越来越大的作用，关注度也越来越高。而 Python 则是最 in 的语言，“人生苦短，我用 Python”^^

怎么用这本书?

机器学习既有算法又有实现,还是比较高深的,算法太难,啃不动,代码太浅,钻不下去。我们的目标是让您快速上手,在内容组织上我们是动了心思的,采用“原理笔记精华+算法 Python 实现+问题实例+实际代码+运行调参”的形式,理论与实践交织着展开,算法原理与编程实战并重。

全书分 13 章进行展开,从内容上分为四篇:机器学习基础篇、机器学习高级篇、机器学习工程篇和 Kaggle 实战篇。

第一篇:机器学习基础篇(第 1~6 章)

包括线性模型、决策树、贝叶斯分类、k 近邻法、数据降维、聚类和 EM 算法等内容。

这些基础算法非常经典,原理也相对简单,是入门的最佳选择,掌握这些算法,才能更好地理解后续的高级算法。老司机可以直接忽略这部分。

第二篇:机器学习高级篇(第 7~10 章)

包括支持向量机、人工神经网络、半监督学习和集成学习等内容。

这些高级算法是目前应用非常广泛,也是效果不错的算法,需要深入理解算法的原理、优劣势等特点以及应用场景,要能达到应用自如的程度。

第三篇:机器学习工程篇(第 11~12 章)

讲述机器学习工程中的实际技术,包括数据预处理,模型评估、选择与验证等内容。

数据清洗、数据预处理和模型评估选择在实际中非常重要,在整个工程项目的开发过程中通常占到一半以上的时间,这部分给出的一些步骤和方法是实践的精华,值得熟练掌握。

第四篇:Kaggle 实战篇(第 13 章)

Step-by-step 讲述一个 Kaggle 竞赛题目的实战,有代码,有分析,有惊喜,有收获。

Kaggle 是目前顶级的数据科学比赛平台,很多机器学习的牛人都在这里玩过,咱们可以学习牛人好的算法,也可以启发自己的思路。对于梦想成为牛人的您,还是去里面混混先:)万一拿了个好的名次呢,拿个一流公司的 offer 还是很 easy 的。

本书的代码全部开源,请自行下载https://github.com/huaxz1986/git_book,也欢迎在这上面交流。

由于作者水平和经验有限,书中错漏之处在所难免,敬请读者指正,我的电子邮箱是 wa2003@126.com。

作者

2017 年元旦于北京

目录

第一篇 机器学习基础篇	1
第 1 章 线性模型	2
1.1 概述	2
1.2 算法笔记精华	2
1.2.1 普通线性回归	2
1.2.2 广义线性模型	5
1.2.3 逻辑回归	5
1.2.4 线性判别分析	7
1.3 Python 实战	10
1.3.1 线性回归模型	11
1.3.2 线性回归模型的正则化	12
1.3.3 逻辑回归	22
1.3.4 线性判别分析	26
第 2 章 决策树	30
2.1 概述	30
2.2 算法笔记精华	30
2.2.1 决策树原理	30
2.2.2 构建决策树的 3 个步骤	31
2.2.3 CART 算法	37
2.2.4 连续值和缺失值的处理	42
2.3 Python 实战	43
2.3.1 回归决策树 (DecisionTreeRegressor)	43
2.3.2 分类决策树 (DecisionTreeClassifier)	49
2.3.3 决策图	54
第 3 章 贝叶斯分类器	55
3.1 概述	55

3.2	算法笔记精华.....	55
3.2.1	贝叶斯定理.....	55
3.2.2	朴素贝叶斯法.....	56
3.3	Python 实战.....	59
3.3.1	高斯贝叶斯分类器 (GaussianNB)	61
3.3.2	多项式贝叶斯分类器 (MultinomialNB)	62
3.3.3	伯努利贝叶斯分类器 (BernoulliNB)	65
3.3.4	递增式学习 partial_fit 方法.....	69
第 4 章	k 近邻法	70
4.1	概述.....	70
4.2	算法笔记精华.....	70
4.2.1	kNN 三要素.....	70
4.2.2	k 近邻算法.....	72
4.2.3	kd 树.....	73
4.3	Python 实践.....	74
第 5 章	数据降维.....	83
5.1	概述.....	83
5.2	算法笔记精华.....	83
5.2.1	维度灾难与降维.....	83
5.2.2	主成分分析 (PCA)	84
5.2.3	SVD 降维.....	91
5.2.4	核化线性 (KPCA) 降维.....	91
5.2.5	流形学习降维.....	93
5.2.6	多维缩放 (MDS) 降维.....	93
5.2.7	等度量映射 (Isomap) 降维.....	96
5.2.8	局部线性嵌入 (LLE)	97
5.3	Python 实战.....	99
5.4	小结.....	118
第 6 章	聚类和 EM 算法	119
6.1	概述.....	119
6.2	算法笔记精华.....	120
6.2.1	聚类的有效性指标.....	120
6.2.2	距离度量.....	122
6.2.3	原型聚类.....	123
6.2.4	密度聚类.....	126

6.2.5 层次聚类.....	127
6.2.6 EM 算法.....	128
6.2.7 实际中的聚类要求.....	136
6.3 Python 实战	137
6.3.1 K 均值聚类 (KMeans)	138
6.3.2 密度聚类 (DBSCAN)	143
6.3.3 层次聚类 (AgglomerativeClustering)	146
6.3.4 混合高斯 (GaussianMixture) 模型.....	149
6.4 小结.....	153

第二篇 机器学习高级篇

155

第 7 章 支持向量机..... 156

7.1 概述.....	156
7.2 算法笔记精华.....	157
7.2.1 线性可分支持向量机.....	157
7.2.2 线性支持向量机.....	162
7.2.3 非线性支持向量机.....	166
7.2.4 支持向量回归.....	167
7.2.5 SVM 的优缺点	170
7.3 Python 实战	170
7.3.1 线性分类 SVM.....	171
7.3.2 非线性分类 SVM.....	175
7.3.3 线性回归 SVR.....	182
7.3.4 非线性回归 SVR.....	186

第 8 章 人工神经网络..... 192

8.1 概述.....	192
8.2 算法笔记精华.....	192
8.2.1 感知机模型.....	192
8.2.2 感知机学习算法.....	194
8.2.3 神经网络.....	197
8.3 Python 实战	205
8.3.1 感知机学习算法的原始形式.....	205
8.3.2 感知机学习算法的对偶形式.....	209
8.3.3 学习率与收敛速度.....	212
8.3.4 感知机与线性不可分数据集.....	213
8.3.5 多层神经网络.....	215

8.3.6	多层神经网络与线性不可分数据集.....	216
8.3.7	多层神经网络的应用.....	219
第9章	半监督学习.....	225
9.1	概述.....	225
9.2	算法笔记精华.....	226
9.2.1	生成式半监督学习方法.....	226
9.2.2	图半监督学习.....	228
9.3	Python 实战.....	234
9.4	小结.....	243
第10章	集成学习.....	244
10.1	概述.....	244
10.2	算法笔记精华.....	244
10.2.1	集成学习的原理及误差.....	244
10.2.2	Boosting 算法.....	246
10.2.3	AdaBoost 算法.....	246
10.2.4	AdaBoost 与加法模型.....	252
10.2.5	提升树.....	253
10.2.6	Bagging 算法.....	256
10.2.7	误差-分歧分解.....	257
10.2.8	多样性增强.....	259
10.3	Python 实战.....	260
10.3.1	AdaBoost.....	261
10.3.2	Gradient Tree Boosting.....	272
10.3.3	Random Forest.....	288
10.4	小结.....	298
第三篇	机器学习工程篇.....	299
第11章	数据预处理.....	300
11.1	概述.....	300
11.2	算法笔记精华.....	300
11.2.1	去除唯一属性.....	300
11.2.2	处理缺失值的三种方法.....	301
11.2.3	常见的缺失值补全方法.....	302
11.2.4	特征编码.....	307

11.2.5	数据标准化、正则化.....	308
11.2.6	特征选择.....	310
11.2.7	稀疏表示和字典学习.....	313
11.3	Python 实践.....	316
11.3.1	二元化.....	316
11.3.2	独热码.....	317
11.3.3	标准化.....	321
11.3.4	正则化.....	325
11.3.5	过滤式特征选取.....	326
11.3.6	包裹式特征选取.....	330
11.3.7	嵌入式特征选取.....	334
11.3.8	学习器流水线 (Pipeline)	339
11.3.9	字典学习.....	340
 第 12 章 模型评估、选择与验证.....		345
12.1	概述.....	345
12.2	算法笔记精华.....	346
12.2.1	损失函数和风险函数.....	346
12.2.2	模型评估方法.....	348
12.2.3	模型评估.....	349
12.2.4	性能度量.....	350
12.2.5	偏差方差分解.....	356
12.3	Python 实践.....	357
12.3.1	损失函数.....	357
12.3.2	数据集切分.....	359
12.3.3	性能度量.....	370
12.3.4	参数优化.....	387
 第四篇 Kaggle 实战篇.....		401
 第 13 章 Kaggle 牛刀小试.....		402
13.1	Kaggle 简介.....	402
13.2	清洗数据.....	403
13.2.1	加载数据.....	403
13.2.2	合并数据.....	406
13.2.3	拆分数据.....	407
13.2.4	去除唯一值.....	408
13.2.5	数据类型转换.....	410

13.2.6	Data_Cleaner 类	412
13.3	数据预处理.....	415
13.3.1	独热码编码.....	415
13.3.2	归一化处理.....	419
13.3.3	Data_Preprocessor 类.....	421
13.4	学习曲线和验证曲线.....	424
13.4.1	程序说明.....	424
13.4.2	运行结果.....	430
13.5	参数优化.....	433
13.6	小结.....	435
	全书符号	436

第一篇

机器学习基础篇

线性模型

1.1 概述

给定样本 $\vec{x}$ ，我们用列向量表示该样本 $\vec{x} = (x^{(1)}, x^{(2)}, \dots, x^{(n)})^T$ 。样本有 n 种特征，我们用 $x^{(i)}$ 表示样本 $\vec{x}$ 的第 i 个特征。线性模型 (linear model) 的形式为：

$$f(\vec{x}) = \vec{w} \cdot \vec{x} + b$$

其中 $\vec{w} = (w^{(1)}, w^{(2)}, \dots, w^{(n)})^T$ 为每个特征对应的权重生成的权重向量，称为权重向量，权重向量直观地表达了各个特征在预测中的重要性。

线性模型中的“线性”其实就是一系列一次特征的线性组合，在二维空间中是一条直线，在三维空间中是一个平面，然后推广到 n 维空间，这样可以理解为广义线性模型。

线性模型非常简单，易于建模，应用广泛，它还有多种推广形式，常见的有广义线性模型，包括岭回归、lasso 回归、Elastic Net、逻辑回归、线性判别分析等。本章将介绍这些模型的基本思想、优缺点以及如何用 Python 实现。

1.2 算法笔记精华

1.2.1 普通线性回归

线性回归是一种回归分析技术，回归分析本质上就是一个函数估计的问题（函数估计包括参数估计和非参数估计两类），就是找出因变量和自变量之间的因果关系。回归分析的因变量应该是连续变量，若因变量为离散变量，则问题转化为分类问题，回归分析是一个有监督学习的问题。

给定数据集 $T = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \dots, (\vec{x}_N, y_N)\}$, $\vec{x}_i \in \mathcal{X} \subseteq \mathbb{R}^n$, $y_i \in \mathcal{Y} \subseteq \mathbb{R}$, $i = 1, 2, \dots, N$, 其中 $\vec{x}_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(n)})^T$ 。我们需要学习的模型为:

$$f(\vec{x}) = \vec{w} \cdot \vec{x} + b$$

也即: 根据已知的数据集 T 来计算参数 $\vec{w}$ 和 b 。

对于给定的样本 $\vec{x}_i$, 其预测值为 $\hat{y}_i = f(\vec{x}_i) = \vec{w} \cdot \vec{x}_i + b$ 。我们采用平方损失函数, 则在训练集 T 上, 模型的损失函数为:

$$L(f) = \sum_{i=1}^N (\hat{y}_i - y_i)^2 = \sum_{i=1}^N (\vec{w} \cdot \vec{x}_i + b - y_i)^2$$

我们的目标是损失函数最小化, 即:

$$(\vec{w}^*, b^*) = \arg \min_{\vec{w}, b} \sum_{i=1}^N (\vec{w} \cdot \vec{x}_i + b - y_i)^2$$

可以用梯度下降法来求解上述最优化问题的数值解。在使用梯度下降法时, 要注意特征归一化 (Feature Scaling), 这也是许多机器学习模型都需要注意的问题, 这么重要的问题, 我们一定要讲三遍!

特征归一化有两个好处。(1) 提升模型的收敛速度, 比如两个特征 x_1 和 x_2 , x_1 的取值为 0~2000, 而 x_2 的取值为 1~5, 假如只有这两个特征, 对其进行优化时, 会得到一个窄长的椭圆形, 导致在梯度下降时, 梯度的方向为垂直等高线的方向而走之字形路线, 这样会使迭代很慢。相比之下, 归一化之后, 是一个圆形, 梯度的方向为直接指向圆心, 迭代就会很快。可见, 归一化可以大大减少寻找最优解的时间。(2) 提升模型精度, 归一化的另一好处是提高精度, 这在涉及一些距离计算的算法时效果显著, 比如算法要计算欧氏距离, 上面 x_2 的取值范围比较小, 涉及距离计算时其对结果的影响远比 x_1 带来的小, 所以这就会造成精度的损失。所以归一化很有必要, 它可以让各个特征对结果做出的贡献相同。在求解线性回归的模型时, 还有一个问题要注意, 那就是特征组合问题, 比如房子的长度和宽度作为两个特征参与模型的构造, 不如把其相乘得到面积作为一个特征来进行求解, 这样在特征选择上就做了减少维度的工作。

回过头来, 上述最优化问题实际上是有解析解的, 可以用最小二乘法求解解析解, 该问题称为多元线性回归 (multivariate linear regression)。

令:

$$\vec{w} = (w^{(1)}, w^{(2)}, \dots, w^{(n)}, b)^T = (\vec{w}^T, b)^T$$

$$\vec{x} = (x^{(1)}, x^{(2)}, \dots, x^{(n)}, 1)^T = (\vec{x}^T, 1)^T$$

$$\vec{y} = (y_1, y_2, \dots, y_N)^T$$

则有:

$$\sum_{i=1}^N (\vec{w} \cdot \vec{x}_i + b - y_i)^2 = \left(\vec{y} - (\vec{x}_1, \vec{x}_2, \dots, \vec{x}_N)^T \vec{w} \right)^T \left(\vec{y} - (\vec{x}_1, \vec{x}_2, \dots, \vec{x}_N)^T \vec{w} \right)$$

令:

$$\vec{x} = (\vec{x}_1, \vec{x}_2, \dots, \vec{x}_N)^T = \begin{bmatrix} \vec{x}_1^T \\ \vec{x}_2^T \\ \vdots \\ \vec{x}_N^T \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(n)} & 1 \\ x_2^{(1)} & x_2^{(2)} & \cdots & x_2^{(n)} & 1 \\ \vdots & \vdots & \ddots & \vdots & 1 \\ x_N^{(1)} & x_N^{(2)} & \cdots & x_N^{(n)} & 1 \end{bmatrix}$$

则:

$$\vec{w}^* = \arg \min_{\vec{w}} (\vec{y} - \vec{x}\vec{w})^T (\vec{y} - \vec{x}\vec{w})$$

令 $E_{\vec{w}} = (\vec{y} - \vec{x}\vec{w})^T (\vec{y} - \vec{x}\vec{w})$, 求它的极小值。对 $\vec{w}$ 求导令导数为零, 得到解析解:

$$\frac{\partial E_{\vec{w}}}{\partial \vec{w}} = 2\vec{x}^T (\vec{x}\vec{w} - \vec{y}) = \vec{0} \implies \vec{x}^T \vec{x}\vec{w} = \vec{x}^T \vec{y}$$

□ 当 $\vec{x}^T \vec{x}$ 为满秩矩阵或者正定矩阵时, 可得:

$$\vec{w}^* = (\vec{x}^T \vec{x})^{-1} \vec{x}^T \vec{y}$$

其中 $(\vec{x}^T \vec{x})^{-1}$ 为 $\vec{x}^T \vec{x}$ 的逆矩阵。于是学得的多元线性回归模型为:

$$f(\vec{x}_i) = \vec{x}_i^T \vec{w}^*$$

□ 当 $\vec{x}^T \vec{x}$ 不是满秩矩阵时。比如 $N < n$ (样本数量小于特征种类的数量), 根据 $\vec{x}$ 的秩小于等于 (N, n) 中的最小值, 即小于等于 N (矩阵的秩一定小于等于矩阵的行数和列数); 而矩阵 $\vec{x}^T \vec{x}$ 是 $n \times n$ 大小的, 它的秩一定小于等于 N , 因此不是满秩矩阵。此时存在多个解析解。常见的做法是引入正则化项, 如 L_1 正则化或者 L_2 正则化。以 L_2 正则化为例:

$$\vec{w}^* = \arg \min_{\vec{w}} \left[(\vec{y} - \vec{x}\vec{w})^T (\vec{y} - \vec{x}\vec{w}) + \lambda \|\vec{w}\|_2^2 \right]$$

其中, $\lambda > 0$ 调整正则化项与均方误差的比例; $\|\dots\|_2$ 为 L_2 范数。

根据上述原理, 我们得到多元线性回归算法:

□ 输入: 数据集 $T = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \dots, (\vec{x}_N, y_N)\}$, $\vec{x}_i \in \mathcal{X} \subseteq \mathbb{R}^n, y_i \in \mathcal{Y} \subseteq \mathbb{R}, i = 1, 2, \dots, N$, 正则化项系数 $\lambda > 0$ 。