Interpretable Machine Learning

可解释的机器学习

黑盒模型可解释性理解指南

Christoph Molnar 著
朱明超 译

@ChristophMolnar

序言

机器学习对于改进产品、过程和研究有着很大的潜力。但是**计算机通常无法解释他们的预测**，这是采用机器学习的障碍。这本书是关于使机器学习模型及其决策可解释的。

在探索了可解释性的概念之后，你将学习简单的、可解释的模型，例如决策树、决策规则和线性回归。后面几章重点介绍了解释黑盒模型的模型无关的一般方法，如特征重要性和累积局部效应，以及用 Shapley 值和 LIME 解释单个实例预测。

所有的解释方法进行了深入说明和批判性讨论。它们如何在黑盒下工作的？它们的优缺点是什么？如何解释它们的输出？本书将使你能够选择并正确应用最适合你的机器学习项目的解释方法。

这本书的重点是表格式数据 (也称为关系数据或结构化数据) 的机器学习模型，较少涉及到计算机视觉和自然语言处理任务。建议机器学习从业者、数据科学家、统计学家和任何对使机器学习模型可解释的人阅读本书。

你可以在 leanpub.com 上购买 PDF 和电子书版本 (epub，mobi)。

你可以在 lulu.com 上购买印刷版。

关于我：我叫克里斯托夫 · 莫纳 (Christoph Molnar)，我是统计学家和机器学习热爱者。我的目标是使机器学习可解释。

邮件：christoph.molnar.ai@gmail.com

网站：https://christophm.github.io/

在 Twitter 上关注我！@ChristophMolnar

@YvonneDoinel 的封面

本书采用 Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License 协议授权。

关于译者：我叫朱明超 (Mingchao Zhu)，我是机器学习热爱者。

邮件：deityrayleigh@gmail.com

你可以在 Github 网站上关注这本书的更新：https://github.com/MingchaoZhu/InterpretableMLBook

目录

第一章　前言

本书向你介绍了如何使 (监督) 机器学习模型可解释。虽然本书中包含一些数学公式，但是即使没有公式，你也需要能够理解这些方法背后的思想。本书不适合机器学习初学者。如果你不熟悉机器学习，则有很多书籍和其他资源可以学习基础知识。关于机器学习的入门学习，我推荐在线学习平台 coursera.com 上的 Hastie，Tibshirani 和 Friedman[1] 所著的《The Elements of Statistical Learning》一书和 Andrew Ng 的 "机器学习" 在线课程。这些书本和在线课程都是免费的!

目前解释机器学习模型的新方法以惊人的速度发表，跟上所有已发布的内容非常不现实。这就是为什么你不会在本书中看到最新颖、最奇特的方法，而是看到机器学习可解释性的成熟方法和基本概念。这些基础知识可以帮助你使机器学习模型具有可解释性。你阅读本书后，内化基础知识还使你能够更好地理解和评估 arxiv.org 上发表的有关可解释性的新论文。

本书以一些 (反乌托邦式的) 短篇小说作为开篇，这些短篇小说不是理解这本书所必需的，但希望它们能使你愉悦并引起思考。然后，本书探讨了机器学习可解释性的概念。我们将讨论可解释性何时重要，以及有哪些不同类型的解释。本书中使用的术语可以在 "术语" 章中查找。所描述的大多数模型和方法都是使用 "数据集" 一章中描述的真实数据集来介绍的。使机器学习可解释的一种方法是使用可解释的模型，例如线性模型或决策树。另一个选择是使用与模型无关的解释工具，这样就可以应用于任何监督机器学习模型。"模型无关方法" 一章处理诸如部分依赖图和置换特征重要性之类的方法。与模型无关的方法通过改变机器学习模型的输入并观察模型的输出变化来工作。返回实例作为解释的模型无关的方法，将在 "基于样本的解释" 一章中进行讨论。所有与模型无关的方法都可以根据它们是在所有数据实例中解释全局模型行为还是单个实例预测来进一步区分。以下方法解释了模型的整体行为：部分依赖图 (`Partial Dependence Plots`)，累积局部效应 (`Accumulated Local Effects`)，特征交互 (`Feature Interaction`)，特征重要性 (`Feature Importance`)，全局代理模型 (`Global Surrogate Models`) 以及原型和批判 (`Prototypes and Criticisms`)。为了解释单个实例预测，我们有局部代理模型 (`Local Surrogate Models`)，Shapley 值解释 (`Shapley Value Explanations`)，反事实解释 (`Counterfactual Explanations`) (密切相关：对抗样本)。可以使用一些方法来解释全局模型行为和单个实例预测两个方面：个体条件期望 (`Individual Conditional Expectation`) 和有影响力的实例 (`Influential Instances`)。

这本书以对可解释机器学习的未来前景持乐观态度作为结尾。

你可以从头到尾阅读这本书，也可以直接跳到你感兴趣的方法。

希望你喜欢阅读!

1.1 故事时间

我们将从一些短篇故事开始。每个故事都是对可解释的机器学习的的夸张表现。如果你赶时间，你可以跳过它们。如果你想获得娱乐和动力，请继续阅读!

该排版的灵感来自杰克 · 克拉克 (Jack Clark) 在他的 Import AI Newsletter。如果你喜欢此类故事或对 AI 感兴趣，我建议你注册。

1.1.1 闪电永不两次

2030 年：瑞士的医学实验室

“这绝对不是最糟糕的死亡方式！”Tom 总结说，试图在这场悲剧中找到积极的东西。他从静脉输液架上拆下了泵。

Lena 补充说：“他只是因为错误的原因死了。”

“当然还有错用的吗啡泵！为我们增大了工作量！”Tom 一边拧下泵的后板一边抱怨。卸下所有螺钉后，他把盘子举起放在一边。他将电缆插入诊断端口。

“你不只是抱怨工作，是吗？”Lena 笑了笑。

“当然不是。从未！”他用讽刺的语气惊呼。

他启动了泵的计算机。

Lena 将电缆的另一端插入平板电脑。“好的，诊断程序正在运行。” 她说，“我真的很好奇出了什么问题。”

“它确实为我们的 John Doe 注射了 Nirvana。那吗啡浓度很高。伙计，我是说…这是第一次，对吧？ 通常情况下，一个坏泵只会散发出很少的甜味或者没有味道。但是永远不会，像那疯狂的注射。”Tom 解释道。

“我知道。你不必说服我……嘿，看那个。” Lena 举起她的平板电脑。“你看到这个峰值了吗？ 这就是止痛药的功效。看！ 这条线显示参照水平。这个可怜的家伙在他的血液系统中混合了多种止痛药，可以杀死他 17 次以上。在这里由我们的泵注入。然后在这里……” 她轻扫，“在这里你可以看到病人死亡的那一刻。

“那么，你知道发生什么了吗，老板？ ” Tom 问他的上司。

“嗯……传感器似乎很好。心率，氧气水平，葡萄糖等……数据已按预期收集。血液氧数据中有些缺失值，但这并不罕见。看这里，传感器还检测出了吗啡衍生物和其他止痛药引起的病人心律减慢和皮质醇水平降低。” 她继续浏览诊断报告。

Tom 着迷地盯着屏幕。这是他对真实设备故障的首次调查。

“好，这是我们的第一个难题。系统未能向医院的通信信道发送警告。警告已触发，但应急方案未响应。这可能是我们的错，但也可能是医院的错。请将日志发送给 IT 团队。”Lena 对 Tom 说。

Tom 点了点头，眼睛仍然盯着屏幕。

Lena 继续说：“这很奇怪。该警告也应该导致泵关闭。但是它显然没有这样做，那一定是个错误。质量团队错过了一些东西。真的很糟糕。也许与应急方案有关。”

“因此，泵的应急系统不知何故发生了故障，但是为什么泵如此疯狂并向 John Doe 注入了很多止痛药？ ” Tom 想知道。

“好问题。你是对的。除了应急紧急故障之外，泵根本不应该使用那么多的药物。鉴于皮质醇和其他警告信号的低水平，该算法应该早点停止。”Lena 解释说。

“也许有些不幸，比如百万分之一，就像被闪电击中一样？ ” Tom 问她。

“不，Tom。如果你阅读了我发给你的文档，你就会知道这个泵首先是在动物实验中测试的，然后是在人类身上测试的，以学习根据感觉输入来注射完美数量的止痛药。泵的算法可能是不透明且复杂的，但不是随机的。这意味着在相同情况下，泵将再次以完全相同的方式运行，我们的病人会再次死亡。感觉输入的组合或不希望有的交互作用必定已经触发了泵的错误行为。这就是为什么我们必须深入挖掘，找出这里发生的事情。”Lena 解释说。

“我明白了……”Tom 迷惑地回答，“病人不是很快就会死吗？ 因为癌症什么的。”

Lena 在阅读分析报告时点了点头。

Tom 起身去窗前。他向外看，目光注视着远处的某个点。“也许，这台机器使他摆脱了痛苦，帮了他一个忙，不再受苦。也许它只是做了正确的事，就像闪电，但是，你知道，一个好闪电。我的意思是就像彩票，但不是随机的。但出于某种原因。如果我是泵，我也会做同样的事情。”

她终于抬起头看着他。

他一直在看外面的东西。

他们都沉默了片刻。

Lena 再次低下头，继续分析。“不，Tom。这是一个错误……只是该死的错误。”

1.1.2　信任倒下

2050 年：新加坡地铁站

她赶到 Bishan 地铁站。按照她的想法，她已经在工作了。新的神经架构的测试现在应该已经完成了。她领导了重新设计政府的“个体纳税情况预测系统”，该系统可以预测一个人是否会向税务部门隐瞒资金。她的团队提出了一个优雅的想法。如果成功的话，该系统不仅可以为税务局提供服务，还可以提供给其他系统，例如反恐警报系统和商业登记处。有一天，政府甚至可以将这些预测纳入公民信任评分。公民信任评分估计一个人的可信度，这个估计会影响你日常生活的各个方面，例如贷款或你要等多久才能拿到新护照。当她走下自动扶梯时，她想象着她的团队系统与公民信任评分系统的整合会是怎样的。

她经常在不降低步行速度的情况下用手擦拭 RFID 阅读器。她的思绪被占据了，但是感觉与现实的不一致在她的大脑中敲响了警钟。

太晚了。

鼻子先是撞到地铁入口，然后屁股跌在地上。门应该被打开……但是没有打开。她傻眼了，站起来看着门旁的屏幕。“请再试一次，”屏幕上友好的笑脸建议。有一个人路过，无视她，并把他的手覆盖在读取系统上。门开了，他穿过了。门又关了。她擦了擦鼻子。鼻子很痛，但至少没有出血。她试图打开门，但又被拒绝了。真奇怪，也许她的公共交通帐户没有足够的代币。她看着智能手表检查帐户余额。

“登录被拒绝。请联系您的公民咨询局！”她的手表通知了她。

一种恶心的感觉像拳头一样打在她的肚子上。她怀疑发生了什么事。为了证实她的想法，她启动了移动游戏“狙击手协会”，一个自我射击游戏。该应用程序又直接自动关闭了，这证实了她的想法。她头晕目眩，再次坐在地上。

仅有一种可能的解释：她的公民信任分数下降了。基本上，小幅度的下降意味着一些小的不便，例如没有乘坐头等舱的航班或者需要等待更长的时间才能获得官方文件。较低的信任度很罕见，这意味着你被视为对社会的威胁。对付这些人的一个措施是让他们远离地铁等公共场所。政府限制公民信任分数低的对象的金融交易，他们还开始积极监视你在社交媒体上的行为，甚至还限制了某些内容，例如暴力游戏。你的公民信任分数越低，就越难增加公民信任分。得分很低的人通常不会恢复原来分数。

她想不出任何原因为什么她的分数会下降。分数基于机器学习。公民信任评分系统就像一台运转良好的发动机，服务于社会。信任评分系统的性能始终受到密切监控。自本世纪初以来，机器学习已经变得越来越好。它的效率如此之高，以至于信任评分系统做出的决定将不再引起争议。绝对可靠的系统。

她绝望地笑了。绝对可靠的系统——只要，系统很少出故障。但失败了。她一定是其中的特例之一；系统错误；从现在开始她就成了被抛弃的人。没有人敢质疑这个系统。它过于与政府、社会本身融为一体，这是不可质疑的。在仅存的少数几个民主国家中，禁止开展反民主运动，不是因为这些国家在本质上是恶意的，而是因为它们会破坏当前制度的稳定。同样的逻辑适用于现在更常见的算法。由于对现状的危害，禁止对算法进行批判。

对算法的信任是社会秩序的基础。为了共同利益，默认接受了罕见的虚假信任评分。成百上千的其他预测系统和数据库也加入了评分，使得无法知道是什么原因导致了评分下降。她感到自己和她的下方好像有一个大黑洞。她惊恐地看着虚空。

她的纳税情况系统最终被纳入公民信托评分系统，但是她从此一无所知。

1.1.3 费米的回形针

612 年 AMS (在火星定居后)：火星博物馆

“历史很无聊。”Xola 对她的朋友小声说道。Xola，一个蓝头发的女孩，正懒洋洋地用左手追着房间里嗡嗡作响的放映机。“历史很重要。”老师沮丧地看着女孩们说道。Xola 脸红了。她没想到她的老师会偷听她。

“Xola，你刚刚学到了什么？”老师问她。“古代人用尽了地球人的所有资源然后死了吗？”她仔细地问。“不。他们使气候变热，不是人，而是计算机和机器。这是行星地球，而不是地球行星。”另一个名叫 Lin 的女孩补充道。Xola 点了点头。老师满怀自豪，笑了笑，点了点头。“你们都是对的。你知道为什么会这样吗？”“因为人们目光短浅和贪婪？”Xola 问。“人们无法停止他们的机器！”Lin 脱口而出。

“再次，你们都是对的，”老师肯定道，“但是要复杂得多。当时大多数人都不知道发生了什么。有些人看到了巨大的变化，但无法扭转它们。这一时期最著名的作品是一位匿名作家的诗。它最好地记录了当时发生的事情。仔细听！”

老师开始了这首诗。十几架小型无人机将自己放置在孩子们面前，并开始将视频直接投射到他们的眼睛中。它显示一个穿着西装的人站在森林里，只剩下树桩。他开始说道：

机器计算；机器预测。

我们继续前进，因为我们是其中的一部分。我们追求经过训练的最佳状态。

最优解是一维的、局部的、无约束的。

硅元素和血肉，追逐着指数。

发展是我们的精神。

当收集所有奖励，和忽略副作用之后；

当所有硬币都被开采，自然已经落后之后；

我们会有麻烦的，毕竟，指数式发展是一个泡沫。

公地悲剧在不断上演，爆炸，就在我们眼前。

冰冷的计算和贪婪，让地球充满热量。

一切都在死亡，

而我们是顺从的。

就像蒙着眼睛的马一样，我们参加自己创造的比赛，迈向文明的大过滤器。

因此，我们坚持不懈地前进。

因为我们是机器的一部分。

拥抱熵。

“黑暗的记忆，” 老师说道，打破了房间的沉默。“它将被上传到你们的图书馆。记住你们的作业下周前交。” Xola 叹了口气。她设法捉住了其中一架小型无人机。无人机从 CPU 和引擎中发出的声音很温暖。Xola 喜欢它温暖她的手。

1.2 什么是机器学习?

机器学习是计算机基于数据做出和改进预测或行为的一套方法。

例如，为了预测房屋的价格，计算机将从过去的房屋销售中学习模式。本书关注于监督机器学习，它涵盖了所有的预测问题，其中我们会有一个数据集，我们已经知道感兴趣的结果 (如过去的房价)，并希望学习预测新数据的结果。有监督的学习不包括例如聚类任务 (相当于无监督学习)，在这些任务中我们没有感兴趣的特定结果，但希望找到数据点的聚类。此外，诸如强化学习之类的也被排除，在这种情况下，智能体 (`Agent`) 基于环境 (`Environment`) 做出动作 (`Action`) 来学习优化某种奖励 (`Reward`) (例如玩俄罗斯方块的计算机)。监督学习的目标是学习一个预测模型，将数据的特征 (如房屋大小、位置、楼层类型等) 映射到输出 (如房屋价格)。如果输出的是类别，则任务称为分类；如果输出是数值，则任务称为回归。机器学习算法通过估计参数 (如权重) 或学习结构 (如树) 来学习模型，且算法由一个最小化的分数或损失函数指导。例如，在房屋价格预测中，机器将最小化房屋的估计价格和预测价格之间的差值，然后，就可以使用经过充分训练的机器学习模型来预测新实例。

房价估计、产品推荐、路牌检测、信用违约预测和欺诈检测：所有这些示例有着共同点，都可以通过机器学习来解决。任务不同，但方法相同：

- 步骤一：数据采集，越多越好。数据必须包含你要预测的结果以及要从中进行预测的其他信息。对于路牌检测 (即 “图像中有路牌吗？”)，你需要收集街道图像并标记是否存在路牌；对

于信用违约的预测，你需要过去实际贷款的数据、客户是否拖欠贷款的信息以及有助于做出预测的数据，例如收入、过去信用违约等；对于房屋价格估计，你可以从过去的房屋销售中收集数据和有关房地产的信息，如大小、位置等。

- 步骤二：将这些信息输入机器学习算法，生成路牌检测模型、信用评级模型或房屋价格估计模型。
- 步骤三：将新数据输入模型。将模型集成到产品或流程中，例如自动驾驶汽车、信贷申请流程或房地产市场网站。

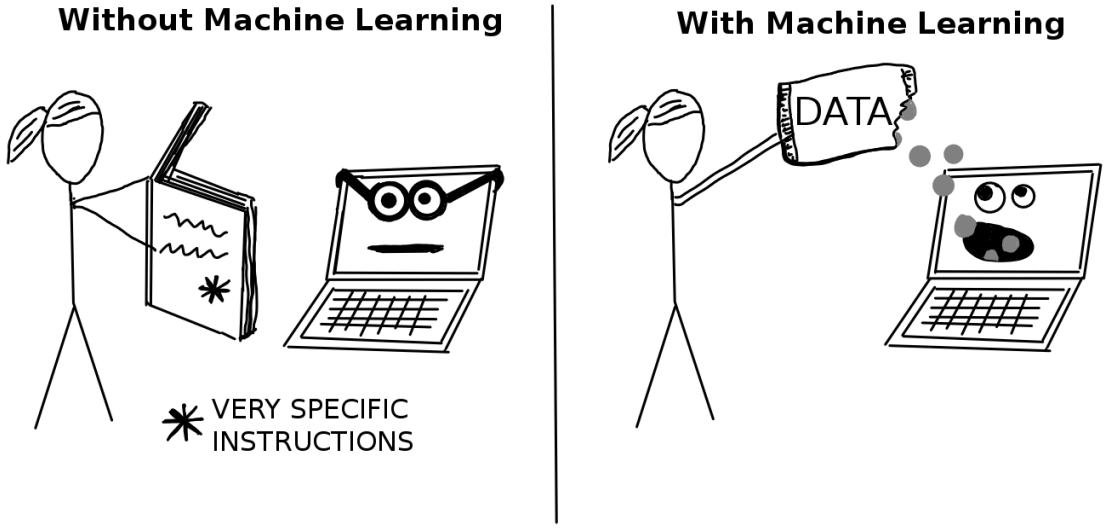
在许多任务上机器都超过人类，例如下棋 (或最近的围棋) 或天气预测。即使机器的性能和人类一样好，或者在某个任务上略微逊色，但在速度、可重复性和规模化方面仍然有很大的优势。一个曾经实施过的机器学习模型可以比人类更快地完成一项任务，可靠地提供一致的结果，并且可以无限地复制。在另一台机器上复制机器学习模型既快速又廉价。对应的，培训人员去完成一项任务则可能需要几十年 (尤其是当他们年轻时)，而且成本很高。使用机器学习的一个主要缺点是，关于数据和机器解决的任务的思路被隐藏在愈加复杂的模型中。你需要数以百万计的参数来描述一个深层的神经网络，而没有办法完全理解这个模型。其他模型，如随机森林，由数百个决策树组成，它们“投票”获得预测。为了理解这个决策是如何做出的，你必须查看数百棵树中每棵树的投票和结构。无论你多么聪明或记忆多么好，这都是行不通的。即使每个模型都可以被解释，但性能最好的模型通常是多个模型的集成，这就变得无法解释了。如果只关注性能，你就会获得越来越多的不透明的模型。例如，看看 Kaggle.com 机器学习竞赛平台对获胜者的采访：胜出的模型大多是模型的集成，或是非常复杂的模型，如提升树或深层神经网络。

1.3 术语

为避免歧义引起混淆，本书中使用的术语定义如下：

算法 (`Algorithm`) 是机器为达到特定目标而遵循的一组规则 [2]。可以将算法视为定义输入、输出以及从输入到输出所需的所有步骤的配方：配方是一种算法，其中材料是输入，熟食是输出，准备和烹饪步骤是算法指令。

机器学习 (`Machine Learning`) 是一套方法，能够允许计算机从数据中学习，以做出和改进预测 (例如癌症、每周销售、信用违约)。机器学习是从“常规编程” (`Normal Programming`) 到“间接编程” (`Indirect Programming`) 的一种范式转换，“常规编程”是指所有指令都必须显式地提供给计算机，而“间接编程”是通过提供数据来实现的。



学习器 (Learner) 或**机器学习算法** (Machine Learning Algorithm) 是用来从数据中学习机器学习模型的程序。另一个名字是 “诱导器” (Inducer) (例如 “树诱导器”)。

机器学习模型 (Machine Learning Model) 是将输入映射到预测的学习程序，这可以是线性模型或神经网络的一组权重。“模型” (Model) 也可以称作 “预测器” (Predictor)，基于任务可以再分为 “分类器” (Classifier) 或者 “回归模型” (Regression Model)。在公式化描述中，经过训练的机器学习模型称为 $\hat{f}$ 或 $\hat{f}(x)$。

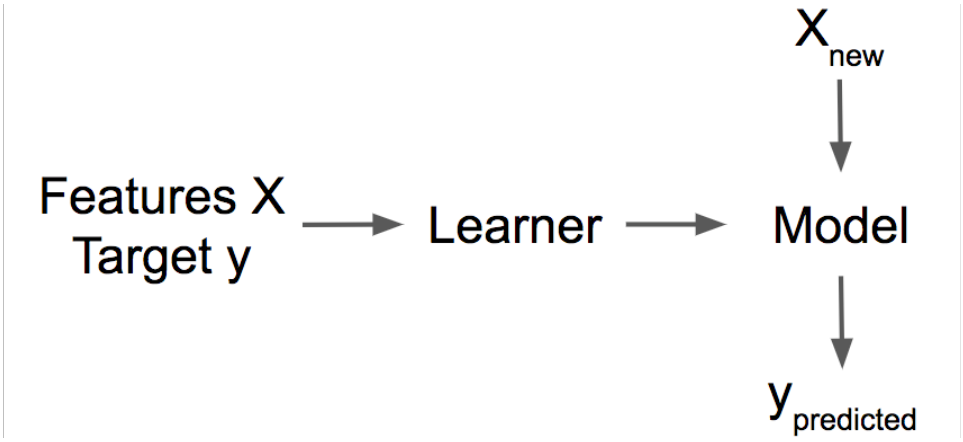
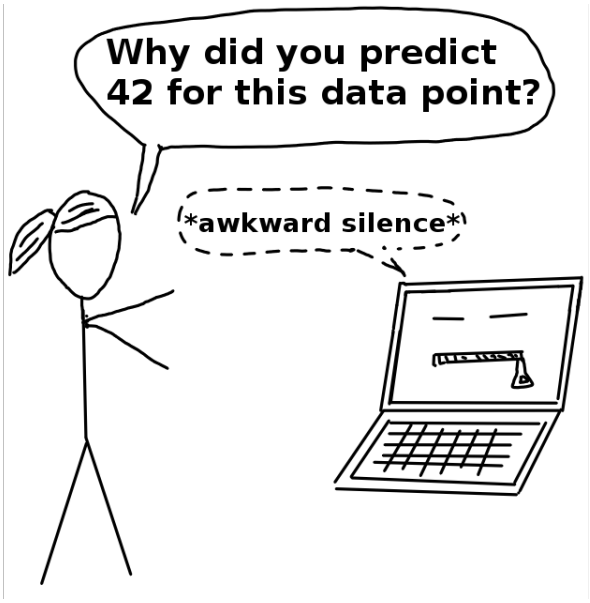


图 1.1. 学习器从标记的训练数据学习模型。该模型用于预测。

黑盒模型 (Black Box Model) 是一个不揭示其内部机制的系统。在机器学习中，“黑盒模型” 或称 “黑匣子” 描述了通过查看参数 (例如深度神经网络的参数) 却无法理解的模型。黑盒的对立面有时被称为**白盒** (White Box)，在本书中被称为可解释模型。模型无关的解释方法将机器学习模型视为黑盒 (即使这些模型本身不是黑盒)。



可解释的机器学习 (Interpretable Machine Learning) 是指使机器学习系统的行为和预测对人类可理解的方法和模型。

数据集 (Dataset) 是一个表格，其中包含机器要从中学习的数据。数据集包含要预测的特征和目标。当用于训练模型时，数据集称为训练数据。

实例 (Instance) 表现为数据集中的一行。实例也可以称作：(数据) 点 (Data Point)、样本 (Example) 或观测 (Observation)。实例由特征值 (向量) $x^{(i)}$ 和目标结果 y_i 组成。

特征 (Features) 是用于对输入进行预测或分类的。特征表现为数据集中的列。本书中认为特征是可解释的，这意味着很容易理解它们的含义，比如某一天的温度或一个人的身高。当然特征的可解释性是一个很大的假设，但如果很难理解输入的特征，那么就更难理解模型的行为。对于单个实例，具有所有特征的矩阵记为 X 和 $x^{(i)}$；所有实例的单个特征向量是 x_j；而第 i 个实例的第 j 个特征对应的值是 $x_j^{(i)}$。

目标 (Target) 是机器要去学会预测的信息。在数学公式中，对于单个实例，目标通常记为 $y^{(i)}$ 或 y_i。

机器学习任务 (Machine Learning Task) 是一个具有特征和目标的数据集的组合。根据目标的类型，任务可以是分类、回归、生存分析、聚类或异常值检测。

预测 (Prediction) 是机器学习模型根据给定的特征 “猜测” 目标值应该是什么。在本书中，模型预测用 $\hat{f}(x^{(i)})$ 或 $\hat{y}$ 表示。

第二章　可解释性

对可解释性是没有数学上定义的。我比较喜欢 Miller (2017)[3] 的 (非数学的) 定义：可解释性是人们能够理解决策原因的程度。另一种定义是 [4]：可解释性是指人们能够一致地预测模型结果的程度。机器学习模型的可解释性越高，人们就越容易理解为什么做出某些决策或预测。如果一个模型的决策比另一个模型的决策能让人更容易理解，那么它就比另一个模型有更高的解释性。我们将在后文中同时使用 Interpretable 和 Explainable 这两个术语来描述可解释性。像 Miller (2017) 一样，区分术语 Interpretable 和 Explainable 是有意义的。我们将使用 Explainable 来描述对单个实例预测的解释。

2.1　可解释性的重要性

如果一个机器学习模型运行良好，为什么我们仅仅信任该模型而忽略为什么做出特定的决策呢? 诸如分类准确性 (`Classification Accuracy`) 之类的单一指标无法完整地描述大多数实际任务。(Doshi-Velez 和 Kim，2017[5])

我们深入探讨可解释性的重要性。当涉及到预测模型时，你需要作出权衡：你是只想知道预测是什么? 例如，客户流失的概率或某种药物对病人的疗效。还是想知道为什么做出这样的预测? 这种情况下可能为了可解释性付出预测性能下降的代价。在某些情况下，你不必关心为什么要做出这样的预测，只要知道模型在测试数据集的预测性能良好就足够了。但是在其他情况下，了解 “为什么” 可以帮助你更多地了解问题、数据以及模型可能失败的原因。有些模型可能不需要解释，因为它们是在低风险的环境中使用的，这意味着错误不会造成严重后果 (例如，电影推荐系统)，或者该方法已经被广泛研究和评估 (例如，光学字符识别 OCR)。对可解释性的需求来自问题形式化的不完整性 [5]，这意味着对于某些问题或任务，仅仅获得预测结果是不够的。该模型还必须解释是怎么获得这个预测的，因为正确的预测只部分地解决了你的原始问题。以下原因推动了对可解释性 (`Interpretability`) 和解释 (`Explanations`) 的需求 (Doshi-Velez 和 Kim (2017)，Miller (2017))：

- **人类的好奇心和学习能力**，这是指人类有着对周围环境的心理模型 (`Mental Model`)，当有意外的事情发生时，人类就会更新这个心理模型，通过为这样的意外事件找到个解释来更新模型。例如，一个人突然感到不舒服，就问自己：“为什么我感到如此不舒服? ”。他得出每次

吃那些红色浆果后他都会生病，于是他更新了他的心理模型，认为浆果导致了疾病，因此应该避免食用。在研究中使用不透明的机器学习模型时，如果模型仅给出预测而没有解释，则科学发现仍是完全不可知的。为了促进学习并满足人们的好奇心——为什么机器产生了某些预测或行为，那么此时可解释性和解释至关重要。当然，人们不需要对所有发生的一切都进行解释。对于大多数人来说，他们可以不理解计算机是如何工作的。但意外事件使我们好奇，例如说，“为什么我的计算机意外关闭？”

- 与学习密切相关的是人类**渴望找到事物存在的意义**。我们希望协调我们知识结构要素之间的矛盾或不一致，例如，“为什么我的狗会咬我，尽管它以前从来没有这样做过。”狗过去的行为与新发生的、令人不快的咬伤经历之间存在着矛盾。兽医的解释调和了狗主人的矛盾：“那只狗因为紧张而咬人。”**机器的决策对人的生活影响越大，机器对它行为的解释就越重要**。如果机器学习模型拒绝贷款申请，这对于申请者来说可能是完全意外的。他们只能用某种说法来调和期望和现实之间的这种不一致。这些解释实际上并不需要完全说清楚情况，但应该至少提供一个主因。另一个例子是算法产品推荐，就我个人而言，我一直在思考为什么某些产品或电影被算法推荐给我。通常情况是：网络上的广告跟踪着我，因为我最近买了一台洗衣机，我知道在接下来的几天里，洗衣机广告会推送给我；由于我们的购物车中已经有冬帽，建议戴手套是很合理的；算法推荐某部电影，是因为和我喜欢相同的其他电影的用户也喜欢着这部电影。互联网公司越来越多地在推荐中添加解释。亚马逊产品推荐就是一个很好的例子，它基于经常购买的产品组合。

图 2.1. 从亚马逊购买颜料时的推荐产品。

- 在许多科学学科中，从定性方法到定量方法 (例如社会学、心理学)，以及到机器学习 (生物学、基因组学) 都发生了变化。**科学的目标**是获取知识，但是许多问题都是通过大数据集和黑盒机器学习模型来解决的。模型本身应该成为知识的来源，而不是数据。可解释性使得可以提取模型捕获的这些额外知识。

- 机器学习模型承担需要**安全措施**和测试的实际任务。想象一下，一辆自动驾驶汽车根据深度学习系统自动检测骑自行车的人。你希望 100% 确定系统所学到的抽象是无错误的，因为要是汽车直接碾过骑车人，这种情况是非常糟糕的。一种解释可能会指出，最重要的学习特征是识别自行车的两个轮子，这种解释可以帮助你思考极端情况，例如自行车的侧袋部分挡住了车轮。

- 默认情况下，机器学习模型从训练数据中学习到了某种偏见，这可能把你的机器学习模型变成歧视受保护群体的种族主义者。可解释性是机器学习模型中一种有效**检测偏见**的调试工具。训练好的自动批准或拒绝信用申请的机器学习模型可能会歧视少数人。而你的主要目标是只向最终会偿还贷款的人提供贷款。在这种情况下，问题表述的不完整性在于，你不仅想要最大程度地减少贷款违约，而且也有义务不根据某些人口统计数据加以区分。这是问题制定中的一个附加约束 (以低风险和合规的方式发放贷款)，而机器学习模型优化所针对的损失函数并没有包含这一约束。

- 将机器和算法整合到日常生活中的过程需要可解释性，以增加**社会认可度**。人们把信仰、欲望、意图等等归因到物体上。在一个著名的实验中，Heider 和 Simmel (1944)[6] 向参与者展示了一些形状的视频，其中一个圆圈打开一扇 “门” 进入一个 “房间” (这只是一个长方形)。参与者描述了形状的动作，就像描述人的行为一样，为形状分配了意图、甚至情感和性格特征。机器人就是一个很好的例子，比如说我的一个吸尘器，名字就叫 “Doge”。如果 Doge 被卡住了，我可以这么想：“Doge 想继续打扫，但它因为卡住了而向我求助。” 后来，当 Doge 打扫完并寻找插座充电时，我想：“Doge 想充电，并打算找到插座。” 我还可以给它赋予性格特征：“Doge 有点笨，但很可爱。” 这些是我的想法，尤其是当我发现 Doge 在尽职地打扫房子的时候撞倒了一棵植物时。能解释其预测的机器或算法会得到更多的认可，同时在后面，我们还会讨论解释是一个社会化过程。

- 解释用于**管理社交互动**。通过创造某个事物的共同含义，解释者影响着解释的接收者的行为、情感和信念。对于一个要与我们互动的机器，它可能需要塑造我们的情感和信念。机器必须 “说服” 我们，这样它们才能达到预期的目标。如果我的扫地机器人没有在某种程度上解释它的行为，我不会完全接受它。扫地机器人创造了一个共同的含义，例如，对于 “事故” (如再次被困在浴室地毯上)，解释说它被卡住了，而不是简单地停止工作却不发表评论。有趣的是，解释机器的目标 (构建信任) 和接收者的目标 (理解预测或行为) 之间可能存在偏差。也许对 Doge 被卡住的原因的完整解释可能是电池电量非常低，其中一个轮子工作不正常，还有一个 bug 让机器人即使有障碍物也一次又一次地走到同一个地方。这些原因 (还有其他一些) 导致机器人被卡住，但它只解释了有什么东西挡在路上，这足以让我相信它的行为，并得到事故的共同含义。顺便说一下，Doge 又被困在浴室里了。我们每次都要把地毯拆下来，然后让 Doge 清理。

图 2.2. 我们的真空吸尘器 Doge 卡住了。作为事故的解释，Doge 告诉我们，它需要在一个平面上。

- 机器学习模型只有在可以解释时才能进行**调试和审核**。即使在低风险环境中，例如电影推荐，在研究和开发阶段以及部署之后，解释能力也是很有价值的。之后，当模型用于产品时，可能会出错。对错误预测的解释有助于理解错误的原因，它为如何修复系统提供了指导方向。考虑一个哈士奇与狼分类器的例子，分类器将一些哈士奇误分类为狼。使用可解释的机器学习方法，你会发现错误分类是由于图像上的雪造成的。分类器学会了使用雪作为一个特征来将图像分类为狼，这对于在训练数据集中分离狼和哈士奇可能是有道理的，但在实际使用中则不然。

如果能够确保机器学习模型能够解释决策，我们还可以更容易地检查以下性质 (Doshi-Velez 和 Kim，2017)：

- 公平性 (`Fairness`)：确保预测是公正的，不会隐式或显式地歧视受保护的群体。可解释的模型可以告诉你为什么它决定某个人不应该得到贷款，并且使人们更容易判断该决策是否基于学习人口统计学偏见 (例如种族)。
- 隐私 (`Privacy`)：确保保护数据中的敏感信息。
- 可靠性 (`Reliability`) 或鲁棒性 (`Robustness`)：确保输入的小变化不会导致预测发生剧烈变化。
- 因果关系 (`Causality`)：检查是否只找到因果关系。
- 信任 (`Trust`)：与黑匣子相比，人们更容易信任解释其决策的系统。

何时我们不需要解释

以下场景说明了我们何时不需要甚至不希望机器学习模型的可解释性。

- 如果模型**没有重大影响**，则不需要解释性。想象一下，一个名为 Mike 的人正在做一个机器学习方面的项目，根据 Facebook 的数据预测他的朋友们下一个假期会去哪里。Mike 就是喜欢有依据地推测朋友们会去哪里度假，从而让他的朋友们吃惊。如果模型是错误的也没有问题 (最坏的情况是，Mike 有点尴尬)；如果 Mike 不能解释模型的输出，那也没有问题。在这种情况下，没有可解释性是完全可以的。如果 Mike 开始围绕这些度假目的地的预测建立业务，情况将会改变。如果模型是错误的，企业可能会赔钱，或者模型可能会因为种族偏见而对某些人变得更糟。一旦模型产生重大影响，无论是金融还是社会，可解释性就变得很重要了。

- 当问题被**研究得很深入**时，就不需要解释性了。一些应用已经得到了充分的研究，因此有足够的模型实践经验，随着时间的推移，模型的问题已经得到解决。一个很好的例子是光学字符识别的机器学习模型，它处理信封中的图像并提取地址。这些系统有多年的使用经验，很明显它们是有效的。此外，我们对获取有关这上面的任务的更多信息并不真正感兴趣。

- 可解释性可能使人或程序能够**操纵系统**。欺骗系统的用户问题是由模型的创建者和用户的目标不匹配造成的。信用评分就是这样一个系统，因为银行希望确保贷款只发放给可能归还贷款的申请人，而申请人的目标是获得贷款，即使银行不想提供给他们。这两个目标之间的不匹配鼓励申请者对系统进行博弈，以增加他们获得贷款的机会。如果申请人知道拥有两张以上的信用卡会对他的分数产生负面影响，他只需退掉第三张信用卡来提高分数，并在贷款获得批准后申请新的信用卡。虽然他的分数有所提高，但偿还贷款的实际可能性并没有改变。只有当输入是因果特征的代理，而不是实际导致结果时，系统才能被博弈。尽可能避免使用代理特征，因为它们使模型有可能被博弈。例如，Google 开发了一个名为 “Google 流感趋势” 的系统来预测流感爆发。该系统将 Google 搜索与流感爆发相关联，但其表现不佳。搜索查询的分布发生了变化，Google 流感趋势错过了许多次流感爆发。Google 搜索不会导致流感，当人们搜索 “发烧” 这样的症状时，这仅仅是与实际流感爆发的关联。理想情况下，模型只使用因果特征，因为它们不可博弈。

2.2 可解释性方法的分类

可以根据各种标准对机器学习可解释性的方法进行分类。

本质的 (Intrinsic) 还是事后的 (Post-hoc)? 该标准通过限制机器学习模型的复杂性 (本质的，亦可称内在的) 或在训练后分析模型的方法 (事后的) 来区分是否实现了可解释性。本质的可解释性是指由于结构简单而被认为是可解释的机器学习模型，如短的决策树或稀疏线性模型；事后解释性是指模型训练后运用解释方法，例如，置换特征重要性是一种事后解释方法。事后方法也可以应用于本质上是可解释的模型上。例如，可以计算决策树的置换特征重要性。本书各章节的组织方式是由本质上可解释模型 (Intrinsically Interpretable Models) 和事后的 (模型无关的) 解释方法

(Post-hoc Interpretation Methods) 之间的区别决定。

解释方法的输出——可以根据解释方法的输出大致区分各种解释方法。

- **特征概要统计量** (Feature Summary Statistic)：许多解释方法为每个特征提供概要统计量。有些方法为每个特征返回一个数字，例如特征重要性，或者更复杂的输出，例如成对特征交互强度，每个特征对表示为一个数字。

- **特征概要可视化** (Feature Summary Visualization)：大多数特征概要统计信息也可以可视化。有些特征概要实际上只有在可视化的情况下才有意义，并且表格不能满足要求。特征的部分依赖就是这样一种情况。部分依赖图是显示特征和平均预测结果的曲线。呈现部分依赖关系的最佳方法是实际绘制曲线，而不是打印坐标。

- **模型内部 (例如学习的权重)** (Model Internals)：对于本质上可解释的模型的解释属于这一类，如线性模型中的权重或决策树的学习树结构 (用于分割的特征和阈值)。但对于像线性模型，因为权重同时是模型内部和特征概要统计量，所以此时两者的界限是模糊的。输出模型内部结构的另一种方法是在卷积神经网络中将学习的特征检测器可视化。根据定义，输出模型内部的可解释性方法是特定于模型的 (请参阅下一个标准)。

- **数据点** (Data Point)：这个类别的方法是返回数据点 (已经存在的或新创建的) 以使模型可解释。一种方法叫做反事实解释 (Counterfactual Explanations)，为了解释对数据实例的预测，该方法通过用一些方式改变某些特征以改变预测结果 (例如预测类别的翻转)，找到相似的数据点。另一个方法是识别预测类的原型，这里输出新数据点的解释方法要求可以解释数据点本身。这对图像和文本很有效，但对于具有数百个特征的表格数据不太有用。

- **本质上可解释模型**：解释黑盒模型的一个解决方案是用可解释模型 (全局地或局部地) 对其进行近似。而这些可解释模型本身可以通过查看模型内部参数或特征概要统计量来解释。

特定于模型 (Model-specific) 还是模型无关 (Model-agnostic)? 特定于模型的解释方法仅限于特定的模型类，例如线性模型中回归权重的解释就是特定于模型的解释，因为根据定义，本质上可解释模型的解释通常是特定于模型的解释。仅应用于解释如神经网络的工具也是特定于模型的。相对应的，与模型无关的工具可以用于任何机器学习模型，并在模型经过训练后应用 (事后的)。这些模型无关的方法通常通过分析特征输入和输出来工作。根据定义，这些方法是不能访问模型的内部信息，如权重或结构信息。

局部 (Local) 还是全局 (Global)? 解释方法是否解释单个实例预测或整个模型行为? 还是范围介于两者之间? 在下一节中会有关于范围标准的更多信息。

2.3 可解释性范围

算法训练产生预测模型，每个步骤都可以根据透明度 (Transparency) 或可解释性进行评估。

2.3.1 算法透明度

算法是如何创建模型的?

算法透明度是指算法如何从数据中学习模型，以及它可以学习到什么样的关系。如果使用卷积神经网络对图像进行分类，则可以解释该算法在最底层学习边缘检测器和滤波器。这是对算法如何工作的理解，但既不是对最终学习的特定模型的理解，也不是对如何做出单个预测的理解。算法的透明度只需要对算法的了解，而不需要对数据或学习模型的了解。这本书的重点是模型的可解释性，而不是算法的透明度。线性模型的最小二乘法等算法已被深入地研究和理解，它们的特点是透明度高。深度学习方法 (通过具有数百万权重的网络推动梯度) 不太容易理解，对其内部工作机制的探究是正在进行的研究重点，它们被认为是低透明度的。

2.3.2 全局、整体的模型可解释性

训练好的模型如何进行预测?

一旦能理解整个模型，就可以将模型描述为可解释的 (Lipton，2016[7])。要解释全局模型输出，你需要训练好的模型、算法知识和数据。这种级别的可解释性是基于对模型特征和每个学习部分 (如权重、其他参数和结构) 的整体认知来理解模型是如何做出决策的。哪些特征很重要，以及它们之间有什么样的交互作用? 全局的模型可解释性有助于基于特征理解目标结果的分布。但在实践中很难实现全局模型可解释性，任何超过几个参数或权重的模型都不可能适合人的短期记忆。就比如说，我认为你很难想象一个具有 5 个特征的线性模型，因为这意味着要想象在 5 维空间中绘制估计的超平面。甚至任何超过 3 维的特征空间都是人们无法想象的。通常，当人们试图理解一个模型时，他们只考虑其中的一部分，例如线性模型中的权重。

2.3.3 模块层面上的全局模型可解释性

模型的某些部分如何影响预测?

具有数百个特征的朴素贝叶斯模型对我们来说太大了，至少很难保存在我们的记忆中。即使我们能够记住所有的权重，我们也无法快速预测新的数据点。此外，我们还需要在头脑中有所有特征的联合分布，以估计每个特征的重要性以及特征平均如何影响预测。所以这基本是一项不可能的任务。但是你能够很容易地理解一个权重。虽然全局模型可解释性通常是无法达到的，但至少有机会在模

块层面上理解某些模型。并非所有模型都可以在参数级别上解释。对于线性模型，可解释部分是权重，对于树来说，是分裂节点和叶节点预测。例如，线性模型看起来似乎可以在模块化层面上完美地解释，但单个权重的解释与所有其他权重是相互关联的。对单个权重的解释总是伴随着脚注，即“其他输入特征保持相同的值”，这在许多实际应用中并不现实。一个预测房屋价格的线性模型，考虑到房屋面积大小和房间数量，对于房间数量的特征可能具有负权重。之所以这种情况可能发生，是因为已经存在高度相关的房屋大小这个特征。在人们更喜欢大房间的市场中，如果两个房屋的面积相同的话，那么房间少的房屋比房间多的房屋更值钱。权重仅在模型中其他特征的上下文中有意义。当然，线性模型中的权重仍然可以比深层神经网络中的权重更好解释。

2.3.4 单个预测的局部可解释性

为什么模型会对一个实例做出某种预测?

当然，你可以着眼于一个实例，检查模型对某个输入的预测，并解释原因。如果你查看单个预测，那么这个原本复杂的模型的行为可能会更令人愉悦。在局部上，预测可能只依赖于线性或单调的某些特征，而不是对它们有着复杂的依赖性。例如，房屋的价格可能与它的面积大小成非线性关系。但是，如果我们只查看一个特定的 100 平方米的房屋，那么对于该数据子集，模型预测可能与面积成线性关系。你可以通过模拟当增加或减少 10 平方米的面积尺寸时，预测的价格是如何变化的来发现这一点。因此，局部解释比全局解释更准确。在后面的介绍中，在 “模型无关方法” 一章可以使单个实例的预测更容易解释。

2.3.5 一组预测的局部可解释性

为什么模型对一组实例进行特定的预测?

多个实例的模型预测可以用全局模型解释方法 (模块级别) 或单个实例的解释来解释。全局方法可以通过获取实例组，将其视为一个完整的数据集以及使用全局方法处理这个数据集。也可以对每个实例使用单独的局部解释方法，然后为整个组列出其结果或对结果进行聚合。

2.4 可解释性评估

对于机器学习中的可解释性至今没有达成共识，如何衡量也不清楚。但有一些初步的研究，并试图制定一些评估方法。

Doshi Velez 和 Kim (2017) 为评估可解释性提出了三个主要层次:

应用级评估 (实际任务) (Application Level Evaluation)：将解释放入产品中，由最终用户进行测试。想象一下，带有机器学习组件的一个骨折检测软件，它可以定位和标记 X 光片中的骨折。在应用层面，放射科医生将直接测试骨折检测软件来评估模型，这需要一个良好的实验装置和对如何评估质量的理解。一个很好的基准是人类在解释相同决策时的表现。

人员级评估 (简单任务) (Human Level Evaluation)：简化的应用级评估。不同的是，这些实验不是由领域专家进行的，而是由非专业人员进行的。这使得实验更廉价 (特别是如果领域专家是放射科医生的话)，并且更容易找到更多的测试人员。例如，向用户展示不同的解释，而用户会选择最好的解释。

功能级评估 (代理任务) (Function Level Evaluation)：不需要人工。当所使用的模型类已经由其他人在人员级评估中进行了评估时，这是最有效的。例如，可能知道最终用户了解决策树。在这种情况下，树的深度可能可以来表示解释质量的好坏。较短的树将获得更高的可解释性得分。增加这种约束条件是有意义的：与较深的树相比，树的预测性能保持良好且不会降低太多。

下面我们将着重于对功能级上的单个预测的解释进行评估。我们将在评估中考虑解释的相关性质是什么?

2.5 解释的性质

我们要解释机器学习模型的预测。为了实现这一点，我们依赖于某个解释方法，一种生成解释的算法。**解释 (Explanation) 通常以一种人类可理解的方式将实例的特征值与其模型预测联系起来**。其他类型的解释包括一组数据实例 (例如，对于 k-最近邻模型)。例如，我们可以使用支持向量机预测癌症风险并使用局部代理方法 (如使用决策树) 来解释预测结果；或者我们可以用线性回归模型作为支持向量机的代理，毕竟线性回归模型的权重可用于解释。

我们仔细研究了解释方法和解释的性质 (Robnik-Sikonja 和 Bohanec，2018[8])，这些性质可用于判断解释方法或解释的好坏。但针对这些性质，如何正确地衡量它们尚不清楚，因此目前的一个挑战是如何规范地计算它们。

解释方法的性质

- **表达能力** (Expressive Power)：是该方法能够产生的解释的“语言”或结构。解释方法可以生成 IF-THEN 规则、决策树、加权和、自然语言或其他东西。
- **半透明度** (Translucency)：描述了解释方法依赖于查看机器学习模型 (如其参数) 的程度。例如，依赖于本质上可解释模型 (如线性回归模型，这是特定于模型的) 的解释方法是高度透明的。而方法仅依赖于修改输入和观察预测，其半透明度为零。根据具体情况，可能需要不同程度的半透明度。高半透明度的优点是该方法可以依赖更多的信息来生成解释。低半透明度

的优点是解释方法更易于移植。

- **可移植性** (Portability)：描述了使用解释方法的机器学习模型的范围。低半透明度的方法具有较高的可移植性，因为它们将机器学习模型视为黑盒。代理模型可能是具有最高可移植性的解释方法。而仅适用于递归神经网络的方法具有低可移植性。

- **算法复杂度** (Algorithmic Complexity)：描述了生成解释的方法的计算复杂性。当计算时间成为生成解释的瓶颈时，必须考虑此性质。

单个解释的性质

- **准确性** (Accuracy)：解释预测看不见的数据会如何？如果将解释代替机器学习模型进行预测，那么高准确性尤为重要。如果机器学习模型的准确性也很低，并且目标是解释黑盒模型的作用，那么低准确性就很好了。在这种情况下，只有保真度才是重要的。

- **保真度** (Fidelity)：解释对黑盒模型预测的近似程度如何？高保真度是解释的重要性质之一，毕竟低保真度的解释对解释机器学习模型是无用的。准确性和保真度密切相关。如果黑盒模型具有较高的准确性并且解释有高保真度，则解释也具有较高的准确性。一些解释只提供局部保真度，这意味着该解释仅非常适合于数据子集的模型预测 (例如局部代理模型)，甚至仅适用于单个数据实例 (例如 Shapley 值)。

- **一致性** (Consistency)：经过相同任务训练并产生相似预测的模型之间的解释有多少不同？例如，我们在同一个任务上训练支持向量机和线性回归模型，两者都产生非常相似的预测。然后我们选择一种解释方法去计算解释，并分析这些解释之间的差异。如果解释非常相似，说明是高度一致的。但这个性质可能会有点棘手，因为这两个模型可以使用不同的特征，但得到相似的预测 (也叫 “罗生门效应”)。在这种情况下，高度一致性又是不可取的，因为解释必须非常不同。但如果模型确实依赖于相似的关系，则需要高一致性。

- **稳定性** (Stability)：类似实例之间的解释会有多相似？一致性是比较模型之间的解释，而稳定性则比较同一模型的相似实例之间的解释。高稳定性意味着实例特征的细微变化基本上不会改变解释 (除非这些细微变化也会强烈改变预测)。缺乏稳定性可能是解释方法差异很大的结果。换句话说，解释方法受到待解释实例的特征值的微小变化的强烈影响。解释方法的不确定性部分也可能导致稳定性不足，例如数据采样步骤，就像局部代理模型使用的那样。高稳定性始终是可取的。

- **可理解性** (Comprehensibility)：人类对解释的理解程度如何？这很难定义和衡量，但非常重要。比较容易接受的观点是可理解性取决于读者和观众。衡量可理解性的想法包括测量解释的大小 (线性模型中非零权重的特征的数量，决策规则的数量等等) 或测试人们如何从解释中预测机器学习模型的行为。还应考虑解释中使用的特征的可理解性，特征的复杂转换可能还不如原来的特征容易理解。

- **确定性** (Certainty)：解释是否反映了机器学习模型的确定性？许多机器学习模型只给出预测，而没有关于预测正确的模型置信度的描述。如果模型预测一个病人患癌症的概率为 4%，那么是否可以确定另一位特征值不同的病人患癌症的概率为 4%？一个包含模型确定性的解释是非常有用的。

- **重要程度** (Degree of Importance)：解释在多大程度上反映了解释的特征或部分的重要性？例如，如果生成决策规则作为对单个预测的解释，那么是否清楚该规则的哪个条件最重要？

- **新颖性** (Novelty)：解释是否反映了待解释的数据实例来自远离训练数据分布的“新”区域？在这种情况下，模型可能不准确，解释可能毫无用处。新颖性的概念与确定性的概念有关。新颖性越高，由于缺乏数据，模型的确定性就越低。

- **代表性** (Representativeness)：一个解释能覆盖多少个实例？解释可以覆盖整个模型 (例如线性回归模型中的权重解释)，也可以只表示单个预测。

2.6 人性化的解释

让我们更深入挖掘，发现人类所认为的“好的”解释，以及对可解释机器学习的意义。人文科学研究可以帮助我们找到答案。Miller (2017) 对有关解释的出版物进行了大量调查研究，本节以他的总结为基础。

在这一节中，我想让你相信以下内容：作为事件的解释，人类更喜欢简短的解释 (只有 1 或 2 个原因)，这些解释将当前的情况与事件不会发生的情况进行了对比，特别是异常原因提供了很好的解释。解释是解释者与被解释者 (解释的接收者) 之间的社会互动，因此社会背景对解释的实际内容有很大的影响。

当你考虑需要一个预测或行为的所有因素的解释时，你不需要人性化的解释，而是完整的因果归因。如果在法律上要求你指出所有有影响的特征或调试机器学习模型，则可能需要因果归因。在这种情况下，请忽略以下几点。而在所有的其他情况下，当外行人或时间很少的人作为解释的接收者时，则以下部分对你来说应该很有趣。

2.6.1 什么是解释

解释是“为什么”这个问题的答案 (Miller，2017)。

- 为什么治疗不起作用？

- 为什么我的贷款被拒绝了？

- 为什么外星人还没有联系我们？

前两个问题可以用 “日常” 的解释来回答，而第三个问题则来自 “更普遍的科学现象和哲学问题”。我们关注 “日常” 类型的解释，因为这些解释与可解释机器学习相关。以 “如何” 开头的问题通常可以改为 “为什么” 问题：“我的贷款是如何被拒绝的？ ” 可以变成” 为什么我的贷款被拒绝了？ ”。

在下文中，“解释” (Explanation) 一词是指解释的社会和认知过程，也指这些过程的产物。解释者 (Explainer) 可以是人或机器。

2.6.2 什么是好的解释?

本节进一步浓缩了 Miller 关于 “好的” 解释的总结，并为可解释机器学习添加了具体含义。

1. 解释具有对比性 (Lipton，1990[9])。人类通常不会问为什么会做出某种预测，但会问为什么会做出这种预测而不是另一种预测。我们倾向于在反事实的情况下思考，即 “如果输入 X 不同，预测会是怎样的？ ”。对于房屋价格预测，业主可能会感兴趣，为什么预测价格比他们预期的低价要高。如果我的贷款申请被拒绝，我不想听到所有的支持或反对的因素。而是我对申请中需要更改才能获得贷款的因素感兴趣。我想知道我的申请和我能被接受的申请之间的对比。认识到对比性的解释是可解释机器学习的一个重要发现。从大多数可解释的模型中，你可以提取一个解释，它隐式地将实例的预测与人工数据实例或一些实例平均值进行对比。医生可能会问：“为什么这种药对我的病人不起作用？ ”。他们可能需要一种解释，将他们的病人与药物起作用的病人以及与药物不作用的病人进行对比。对比性的解释比完整的解释更容易理解。医生对药物为什么不起作用的问题的一个完整的解释可能包括：病人已经有 10 年的疾病，11 个基因过表达，病人身体很快将药物分解成无效的化学物质等等。但一个对比的解释可能是更简单：与有反应的患者相比，无反应的患者具有一定的基因组合，使药物的疗效降低。最好的解释是强调感兴趣的对象和参照对象之间最大的差异。

它对于可解释机器学习意味着什么：人类不希望对预测有一个完整的解释，而是希望将不同之处与另一个实例 (可以是人工的) 的预测进行比较。创建对比性的解释依赖于应用程序，因为它需要一个参照点来进行比较。这可能取决于要解释的数据点，也取决于接受解释的用户。一个房屋价格预测网站的用户可能想要得到房屋价格预测的解释，使其能将他们自己的房屋或网站上的另一个房屋或附近的一个普通房屋形成对比。自动创建对比性解释的解决方案还可能涉及在数据中寻找原型。

2. 选择性的解释。人们不希望对涵盖事件的实际原因和完整原因进行解释。我们习惯于从各种可能的原因中选择一个或两个原因作为解释。作为证明，打开电视新闻：“股票价格的下跌被归咎于由于最新软件更新的问题而对该公司产品越来越强烈的反弹。”

“Tsubasa 和他的球队因为防守薄弱而输掉了比赛：他们给对手太多的空间来发挥他们的战术。”

“对既有机构和政府的不信任感日增是选民投票率降低的主要因素。”

一个事件可以由各种原因解释，则被称为 “罗生门效应”。《罗生门》是一部日本电影，讲述了有关武士之死的另类矛盾故事。对于机器学习模型，如果能根据不同的特征做出一个好的预测是有利的。

将使用不同的特征 (不同的解释) 的多个模型结合在一起的集成方法通常表现良好，进行平均可以使预测更加鲁棒和准确。但这也意味着有不止一个选择性的解释——为什么做出了某种预测。

它对可解释机器学习意味着什么：解释要简短，即使真实情况很复杂，但只给出 1 到 3 个原因。LIME 在这方面就做得很好。

3. 解释是社会性的。它们是解释者和解释的接收者之间对话或交互的一部分。社会背景决定了解释的内容和性质。如果我想向技术人员解释为什么数字加密货币价值如此之高，我会这样说：“分散的、分布式的、基于区块链的账本，不能由一个中央实体控制，与那些想确保财富安全的人产生共鸣。” 这就解释了。但对我的祖母，我会说：“看，祖母，加密货币有点像计算机黄金。人们喜欢并会花很多钱买黄金，年轻人喜欢并会花很多钱买计算机黄金。”

它对可解释机器学习意味着什么：注意机器学习应用程序的社会环境和目标受众。正确地使用机器学习模型的社交部分完全取决于你的特定应用程序。这方面可以找人文学科的专家 (如心理学家和社会学家) 帮助你。

4. 解释的重点是异常。人们更关注异常原因来解释事件 (Kahnemann 和 Tversky，1981[10])。这些原因发生的可能性很小，但还是发生了。消除这些异常原因将大大改变结果 (反事实解释)。人类认为这些 “异常” 原因是很好的解释。Trumbelj 和 Kononenko (2011)[11] 的一个例子是：假设我们有教师和学生之间的测试情况数据集。学生们参加一门课程，并在成功做一个演示后直接通过该课程。老师可以额外选择通过问学生问题的方式来测试学生的知识。回答不上这些问题的学生将不及格。学生可以有不同程度的准备，这意味着可以正确回答老师问题的概率也不同。我们要预测一个学生是否会通过这门课，并解释我们的预测。如果老师没有提出任何额外的问题，通过的概率是 100%，否则通过的概率取决于学生的准备水平和正确回答问题的概率。

情景 1：教师通常会向学生提出额外的问题 (例如，100 次中有 95 次)。一个没有准备的学生 (10% 的机会通过问题部分) 他不幸得到了额外的问题，但他没有正确回答。学生为什么不及格？我想说不准备是学生的错。

情景 2：教师很少问其他问题 (例如，100 次中有 2 次)。对于一个没有为这些问题准备过的学生来说，我们预测通过课程的可能性很高，因为提额外问题不太可能。当然，其中一个学生没有准备好这些问题，这给了他 10% 的机会通过这些问题。他很倒霉，老师又问了一些学生不能回答的问题，结果他没能通过这门课。失败的原因是什么？我认为现在更好的解释是 “因为老师测试了学生”。老师不太可能提问测试，所以老师表现异常。

这对于可解释机器学习意味着什么：如果一个预测的输入特征在任何意义上都是异常的 (比如分类特征的一个罕见类别)，并且该特征影响了预测，那么应该将其包括在解释中，即使此时其他 “正常” 特征具有对预测的影响与异常预测相同。在我们的房屋价格预测的例子中，一个不正常的特征可能是一个相当昂贵的房子有两个阳台。即使某种归因方法发现，这两个阳台对价格差异的影响与住房面积、良好的邻里环境或近期装修一样大，但 “两个阳台” 的异常特征可能是解释为什么房子

如此昂贵的最好解释。

5. 解释是真实的。事实证明，良好的解释是真实的。但这并不是 “好的” 解释的最重要因素。例如，选择性似乎比真实性更重要。仅选择一个或两个可能原因的解释很少涵盖相关原因的整个列表。选择性忽略了事实的一部分。例如，只有一个或两个因素导致了股市崩溃，这是不正确的，但事实是，有数百万个原因影响着数百万人的行事方式，最终导致了股市崩溃。

它对可解释机器学习意味着什么：解释应该尽可能真实地预测事件，在机器学习中有时被称为**保真度**。所以如果我们说第二个阳台增加了一套房屋的价格，那么这也应该适用于其他的房屋 (或至少适用于类似房屋)。对人类来说，解释的保真度不如它的选择性、对比性和社会性重要。

6. 好的解释与被解释者的先验知识是一致的。人类往往忽视与他们先验知识不一致的信息，这种效应被称为确认偏差 (`Confirmation Bias`) (Nickerson，1998[12])。这种偏差不能幸免，人们往往会贬低或忽视与他们先验知识不一致的解释。这套先验知识因人而异，但也有基于群体的先验知识，如政治世界观。

它对可解释机器学习意味着什么：“好的” 解释与先验知识是一致的。这很难整合到机器学习中，可能会大大损害预测性能。我们先前认为房屋面积大小对预测价格的影响是，房屋越大，价格越高。假设一个模型还显示了房屋大小对一些房屋的预测价格的负面影响。模型之所以了解到这一点，是因为它提高了预测性能 (由于一些复杂的交互作用)，但这种行为与我们先验知识强烈矛盾。你可以强制执行单调性约束 (一个特征只能影响一个方向的预测)，或者使用具有此性质的线性模型之类的东西。

7. 好的解释是普遍性的和很可能的。可以解释许多事件的原因是非常普遍的，可以被认为是一个好的解释。请注意，这与认为异常原因能够做出好的解释的说法相矛盾。如我所见，异常原因胜过普遍原因。根据定义，在给定的情况下，异常原因是罕见的。在没有异常事件的情况下，普遍性的解释被认为是一个好的解释。还要记住的是，人们往往会误判共同事件的可能性 (例如，Joe 是图书管理员，他更可能成为一个害羞的人还是一个喜欢读书的害羞的人？)。对于这类解释，一个很好的例子是 “房屋之所以昂贵是因为它很大”，这是一个非常普遍性的、很好的解释——为什么房屋昂贵或便宜。

它对可解释机器学习意味着什么：普遍性可以很容易地通过特征的 “支持” 来衡量，即解释应用到的实例数除以实例总数。

第三章　数据集

在整本书中，所有的模型和技术都是应用在可在线免费获得的真实数据集上。我们将为不同的任务使用不同的数据集：分类，回归和文本分类。

3.1　自行车租赁 (回归)

该数据集包含来自华盛顿特区的自行车租赁公司 Capital-Bikeshar 的自行车租赁的每日计数，以及天气和季节信息。该数据由 Capital-Bikeshare 公开提供。Fanaee-T 和 Gama (2013)[13] 添加了天气数据和季节信息。目的是根据天气和天数来预测将租用多少辆自行车。这些数据都可以从 UCI 机器学习数据库 里下载。

有许多新特征被添加到数据集中，但本书的示例并没有使用所有原始特征，下面列出使用到的特征列表：

- 自行车租赁的数量，包括来自于游客用户和注册用户。这个数量也是回归任务的预测目标。
- 季节，包括春、夏、秋、冬。
- 指示一天是否为假期。
- 年份，2011 年或 2012 年。
- 自 2011 年 1 月 1 日 (数据集中的第一天) 起的天数。引入此特征是为了考虑随时间变化的趋势。
- 指示一天是工作日还是周末。
- 当天的天气状况。是下面几种情况中的一个:
  - 晴，少云，部分多云，多云
  - 雾 + 云，雾 + 碎云，雾 + 少云，雾
  - 小雪，小雨 + 雷雨 + 散云，小雨 + 散云
  - 大雨 + 冰雹 + 雷雨 + 薄雾，雪 + 薄雾
- 温度 (摄氏度)。
- 相对湿度百分比 (0 到 100)。
- 风速，单位：km/h。

对于本书中的示例，数据已进行了少量处理。你可以在本书的 Github 存储库 中找到正在处理的 R 脚本以及 最终的 RData 文件。

3.2 YouTube 垃圾评论 (文本分类)

以文字分类为例，我们使用了来自 5 个不同 YouTube 视频的 1956 条评论。值得庆幸的是，在有关垃圾评论分类的文章中使用此数据集的作者 免费提供 了这些数据 (Alberto，Lochter 和 Almeida，2015[14])。

这些评论是通过 YouTube API 从 2015 年上半年 YouTube 上观看次数最多的十个视频中的五个收集的。所有五个都是音乐视频。其中之一就是韩国艺术家 Psy 创作的 “Gangnam Style”。其他艺术家是 Katy Perry，LMFAO，Eminem 和 Shakira。

查看一些评论。这些评论被手动标记为垃圾评论或正常评论。垃圾评论的编码为 “1”，正常评论的编码为 “ 0”。

评论	类别
Huh, anyway check out this you channel: kobyoshi02	1
Hey guys check out my new channel and our first vid THIS IS US THE MONKEYS!!! I’m the monkey in the white shirt, please leave a like comment and please subscribe!!!!	1
just for test I have to say murdev. com	1
me shaking my sexy ass on my channel enjoy ^_^	1
watch?v=vtaRGgvGtWQ Check this out .	1
Hey, check out my new website!! This site is about kids stuff. kidsmediausa . com	1
Subscribe to my channel	1
i turned it on mute as soon is i came on i just wanted to check the views…	0
You should check my channel for Funny VIDEOS!!	1
and u should.d check my channel and tell me what I should do next!	1

你也可以转到 YouTube 并查看评论部分。但是可千万别舍本逐末，最终观看了猴子从海滩上的游客那里偷喝鸡尾酒的视频。自 2015 年以来，谷歌垃圾评论探测器也可能发生了很大变化。

点击观看 Psy「江南 Style」

如果你想使用数据，可以在本书的 Github 存储库中找到 RData 文件 以及 R 脚本。

3.3 宫颈癌的危险因素 (分类)

宫颈癌数据集包含预测女性是否会患宫颈癌的指标和危险因素。这些特征包括人口统计学数据 (如年龄)、生活方式和病史。数据可从 UCI 机器学习库 下载，并由 Fernandes、Cardoso 和 Fernandes (2017) 整理 [15]。

本书中使用到的数据集的部分特征如下：

- 年龄 (岁)
- 性伴侣数量
- 首次性行为 (岁)
- 怀孕次数
- 是否吸烟
- 烟龄 (单位：年)
- 是否服用激素避孕药
- 服用激素避孕药的时间 (单位：年)
- 是否有宫内节育器 (IUD)
- 使用宫内节育器 (IUD) 的时间 (年数)
- 是否患有性传播疾病 (STD)
- 性病诊断次数
- 第一次性病诊断后到现在的时间
- 上次性病诊断到现在的时间
- 活检结果为 “健康” 或 “癌症”。这是目标输出。

活检结果作为判断是否患癌症的最终结果。对于本书中的例子，活检结果被用作目标。

数据中每列的缺失值都是由众数 (最常见的值) 来代替，这可能并非是一个好办法，因为真正的答案可能与某个值缺失的概率相关。可能会有偏差，因为这些问题是非常私人的。但这并不是一本关于缺失数据插补的书，所以我们必须认为众数插补是足以作为回归分析来使用的。

要使用该数据集重现本书的示例，请在本书的 Github 存储库 中找到预处理的 R 脚本和 最终的 RData 文件。

第四章　可解释的模型

实现可解释性的最简单方法是只使用创建可解释模型的算法子集。线性回归、逻辑回归和决策树是常用的可解释模型。

在接下来的内容中，我们将讨论这些模型。没有详细介绍，仅提供基础知识，因为已经有大量书籍、视频、教程、论文和更多材料可供使用。我们将专注于如何解释模型。该书更详细地讨论了线性回归、逻辑回归、其他线性回归扩展、决策树、决策规则和 RuleFit 算法，还列出了其他可解释的模型。

除 k-最近邻算法外，本书中解释的所有可解释的模型都可以在模块级别上解释。下表概述了可解释的模型类型及其性质。如果特征和目标之间的关联是线性建模的，那么模型就是线性的。具有单调性约束的模型可确保特征和目标结果之间的关系在整个特征范围内始终朝着相同的方向：特征值的增加要么总是导致目标结果的增加，要么总是导致目标结果的减少。单调性对于模型的解释是有用的，因为它使理解关系变得更容易。一些模型可以自动地包含特征之间的交互，来预测目标结果。你可以通过手动创建交互特征，可以将交互包括在任何类型的模型中。交互可以提高预测性能，但太多或太复杂的交互都会损害可解释性。有些模型只处理回归，有些只处理分类，还有一些模型两者都处理。

下面这个表你可以用来为任务选择合适的可解释模型，无论回归 (regr) 问题还是分类 (class) 问题：

算法	线性	单调性	交互	任务
线性回归	Yes	Yes	No	regr
逻辑回归	No	Yes	No	class
决策树	No	Some	Yes	class,regr
RuleFit	Yes	No	Yes	class,regr
朴素贝叶斯	No	Yes	No	class
k-最近邻	No	No	No	class,regr

4.1 线性回归

线性回归 (Linear Regression) 模型将目标预测为特征输入的加权和，而所学习关系的线性使解释变得容易。统计学家、计算机科学家以及其他解决定量问题的人长期以来都使用线性回归模型。

线性模型可用于建模回归目标 y 对某些特征 x 的依赖性。由于学到的关系是线性的，可以针对第 i 个实例写成如下：

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \epsilon$$

实例的预测结果是其 p 个特征的加权和。参数 β_j 表示要学习的特征权重或系数，其中第一项 β_0 称为截距，不与特征相乘。ϵ 表示误差，即预测结果与真实结果之间的差。假设这些误差服从高斯分布，这意味着我们在正负方向上都会产生误差，并且会产生很多小的误差和少量大的误差。

可以使用各种方法来估计最佳权重。通常使用最小二乘法来找到使真实结果和预测结果之间平方差最小化的权重：

$$\hat{\boldsymbol{\beta}} = \arg\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y^{(i)} - \left(\beta_0 + \sum_{j=1}^{p} \beta_j x_j^{(i)} \right) \right)^2$$

我们不会详细讨论如何找到最佳权重，但如果你感兴趣，可以阅读《The Elements of Statistical Learning》[1] 一书的第 3.2 章或者线性回归模型的其他网上资源。

线性回归模型的最大优点是线性：它使估计过程变得简单，而最重要的是，这些线性方程在模块化水平 (即权重) 上具有易于理解的解释。这是线性模型和所有类似模型在医学、社会学、心理学等许多定量研究领域应用如此广泛的主要原因之一。例如，在医学领域，不仅要预测病人的临床结果，而且要量化药物的影响，同时以可解释的方式考虑性别、年龄和其他特征。

估计的权重带有置信区间。置信区间是权重估计的范围，它以一定的置信度覆盖“真实”权重。例如，权重为 2 的 95% 置信区间可能为 1 到 3。这个区间的解释是：如果我们用新的抽样数据重复估计 100 次，假设线性回归模型是正确的数据模型情况下，置信区间将包括 100 个案例中 95 个的真实权重。

模型是否为“正确”模型取决于数据中的关系是否满足某些假设，即线性、正态性、同方差性、独立性、固定特征和不存在多重共线性。

线性

线性回归模型使预测成为特征的线性组合，这既是其最大的优势，也是其最大的局限。线性导致其为可解释模型。线性效应易于量化和描述，它们是可加的，因此很容易分离效应。如果你怀疑有特征交互或特征与目标值的非线性关联，则可以考虑添加交互项或使用回归样条。

正态性

假设给定特征的目标结果服从正态分布。如果违反此假设，则特征权重的估计置信区间无效。

同方差性

假设误差项的方差在整个特征空间内是恒定的。例如，假设你要根据居住面积 (平方米) 来预测房屋的价格。你估计一个线性模型，该线性模型假设不管房屋面积大小如何，预测响应周围的误差具有相同的方差。这种假设在现实中经常被违背。在房屋示例中，对于较大的房屋，由于价格较高且存在更大的价格波动空间，因此在较大的房屋周围，围绕预测价格的误差项的方差较高可能是合理的。假设线性回归模型中的平均误差 (预测价格和真实价格之间的差异) 为 50,000 欧元。如果假设同方差，则对于成本为 100 万的房屋和成本仅为 40,000 的房屋，平均误差都为 50,000。显然这是不合理的，因为后者我们甚至能得到一个负的价格。

独立性

假设每个实例独立于任何其他实例。如果执行重复测量，例如每个患者进行多次血液测试，则样本点 (数据点) 不是独立的。对于相关样本，你就需要特殊的线性回归模型，如混合效应模型 (`Mixed Effect Model`) 或 GEE。如果使用正常的线性回归模型，可能会从模型中得出错误的结论。

固定特征

输入特征被认为是 “固定的”。固定意味着它们被视为 “给定常数”，而不是统计变量。这意味着它们没有测量误差。这是一个相当不切实际的假设。然而，如果没有这个假设，就将不得不拟合非常复杂的测量误差模型，这些模型会考虑输入特征的测量误差。当然通常你不想这样做。

不存在多重共线性

你不需要强相关的特征，因为这会扰乱对权重的估计。在两个特征强相关的情况下，由于特征效应 (`Feature Effects`，也可以称为特征影响) 是累加的，因此估计权重变得很困难，并且无法确定哪一个相关特征归因了效应。

4.1.1 解释

线性回归模型中权重的解释取决于相应特征的类型。

- 数值特征：将数值特征增加一个单位会根据其权重改变估计结果。例如，房屋面积大小。
- 二分类特征：每个实例都采用两个可能值之一的特征。例如，“房屋有花园” 特征 (用 1 编码)，而另一个值被视为参照类别，例如 “房屋没有花园” (用 0 编码)。将特征从参照类别更改为其他类别会根据特征的权重改变估计结果。
- 具有多个类别的分类特征：具有固定数量的可能值的特征。例如，“地板类型” 特征，可能的类别为 “地毯”、“层压板” 和 “镶木地板”。一种处理多种类别的解决方案是 one-hot 编码，这

意味着每个类别都有自己的二进制列。对于具有 L 个类别的分类特征，你只需要 L − 1 列，因为第 L 列将具有冗余信息 (例如，当列 1 到 L − 1 的值都为 0 时，我们知道此实例的分类特征为第 L 个类别)。然后对每个类别的解释与对二分类特征的解释相同。某些语言 (例如 R) 允许你以各种方式对分类特征进行编码，如本节稍后所述。

- 截距项 β_0：截距是 “常量特征” 的特征权重，对于所有实例都是 1。大多数软件包会自动添加 “1” 这个特征来估计截距。解释是：对于所有数值特征为零和分类特征为参照类别下的实例，模型预测是截距权重。截距的解释通常不相关，因为所有特征值都为零的实例通常没有意义。只有当特征标准化 (均值为 0，标准差为 1) 时，这种解释才有意义。然后，截距就将反映当所有特征都处于其均值时的实例的预测结果。

线性回归模型中特征的解释可以通过使用以下文本模板自动进行。

数值特征的解释

当所有其他特征保持不变时，特征 x_k 增加一个单位，预测结果 y 增加 β_k。

分类特征的解释

当所有其他特征保持不变时，将特征 x_k 从参照类别改变为其他类别时，预测结果 y 会增加 β_k。

解释线性模型的另一个重要度量是 R-平方度量 (`R-squared Measurement`)。R-平方告诉你模型解释了目标结果的总方差中的多少。R-平方越高，模型对数据的解释就越好。R-平方的计算公式为：

$$R^2 = 1 - SSE/SST$$

SSE 是误差项的平方和：

$$SSE = \sum_{i=1}^{n}(y^{(i)} - \hat{y}^{(i)})^2$$

SST 是数据方差的平方和：

$$SST = \sum_{i=1}^{n}(y^{(i)} - \bar{y})^2$$

SSE 会告诉你在拟合线性模型后还有多少方差，该方差是是通过预测结果和真实结果之间的平方差来衡量的。SST 是目标结果的总方差。R-平方告诉你有多少方差可以用线性模型来解释。对于根本无法解释数据的模型，R-平方的值为 0；对于解释数据中所有方差的模型，R-平方的值为 1。

这里有一个陷阱，因为 R-平方随模型中的特征数量的增加而增加，即便它们不包含任何关于目标值的信息也是如此。因此，最好使用调整后的 R-平方，它考虑了模型中使用的特征数量。其计算如下：

$$\bar{R}^2 = 1 - (1 - R^2)\frac{n-1}{n-p-1}$$

其中 p 是特征的数量，n 是实例的数量。解释一个 (调整后的) R-平方很低的模型是没有意义的，因为这样的模型基本上不能解释大部分的方差，对权重的任何解释都没有意义。

特征重要性

在线性回归模型中某个特征的重要性可以用它的 t-统计量 (`t-statistic`) 的绝对值来衡量。t-统计量是以标准差为尺度的估计权重。

$$t_{\hat{\beta}_j} = \frac{\hat{\beta}_j}{SE(\hat{\beta}_j)}$$

此公式告诉我们的内容：特征的重要性随着权重的增加而增加。估计权重的方差越大 (= 我们对正确值的把握越小)，特征就越不重要。这也是有道理的。

4.1.2 示例

在此示例中，给定天气和日历信息的情况下，我们使用线性回归模型来预测特定日期的自行车租赁数量。为了便于解释，我们检查了估计的回归权重。这些特征包括数值特征和分类特征。对于每个特征，该表显示估计的权重、估计的标准差 (SE) 和 t-统计量的绝对值 (|t|)。

	Weight	SE	|t|
(Intercept)	2399.4	238.3	10.1
seasonSUMMER	899.3	122.3	7.4
seasonFALL	138.2	161.7	0.9
seasonWINTER	425.6	110.8	3.8
holidayHOLIDAY	-686.1	203.3	3.4
workingdayWORKING DAY	124.9	73.3	1.7
weathersitMISTY	-379.4	87.6	4.3
weathersitRAIN/SNOW/STORM	-1901.5	223.6	8.5
temp	110.7	7.0	15.7
hum	-17.4	3.2	5.5
windspeed	-42.5	6.9	6.2
days_since_2011	4.9	0.2	28.5

数值特征 (温度特征/temp) 的解释：当所有其他特征保持不变时，将温度升高 1 摄氏度，自行车的预测数量增加 110.7 。

分类特征 (天气状况特征/weathersit) 的解释：与好天气相比，当下雨、下雪或暴风雨时，自行车的估计数量减少了 1901.5；再次假设所有其他特征不变，当天气有雾时，自行车的预测数量比正常天气少了 379.4 。

所有的解释总是伴随 “所有其他特征保持不变”，这是因为线性回归模型的性质。预测目标是加权特征的线性组合。估计的线性方程是特征/目标空间中的超平面 (在单个特征的情况下为线)。权重

指定每个方向上超平面的斜率 (梯度)。好的一方面是，可加性将单个特征的解释与所有其他特征隔离开来。这是可能的，因为方程式中的所有特征效应 (= 权重乘以特征值) 都是用加号组合在一起。坏的一面是，这种解释忽略了特征的联合分布。增加一个特征，但不改变另一个特征，这可能不合实际或者是不太可能的数据点。例如，如果不增加房屋的面积大小，却增加房间的数量就可能是不现实的。

4.1.3 可视化解释

各种可视化效果使线性回归模型易于快速掌握。

权重图 (Weight Plot)

权重表的信息 (权重和方差估计) 可以在权重图中可视化。下图显示了先前线性回归模型的结果。

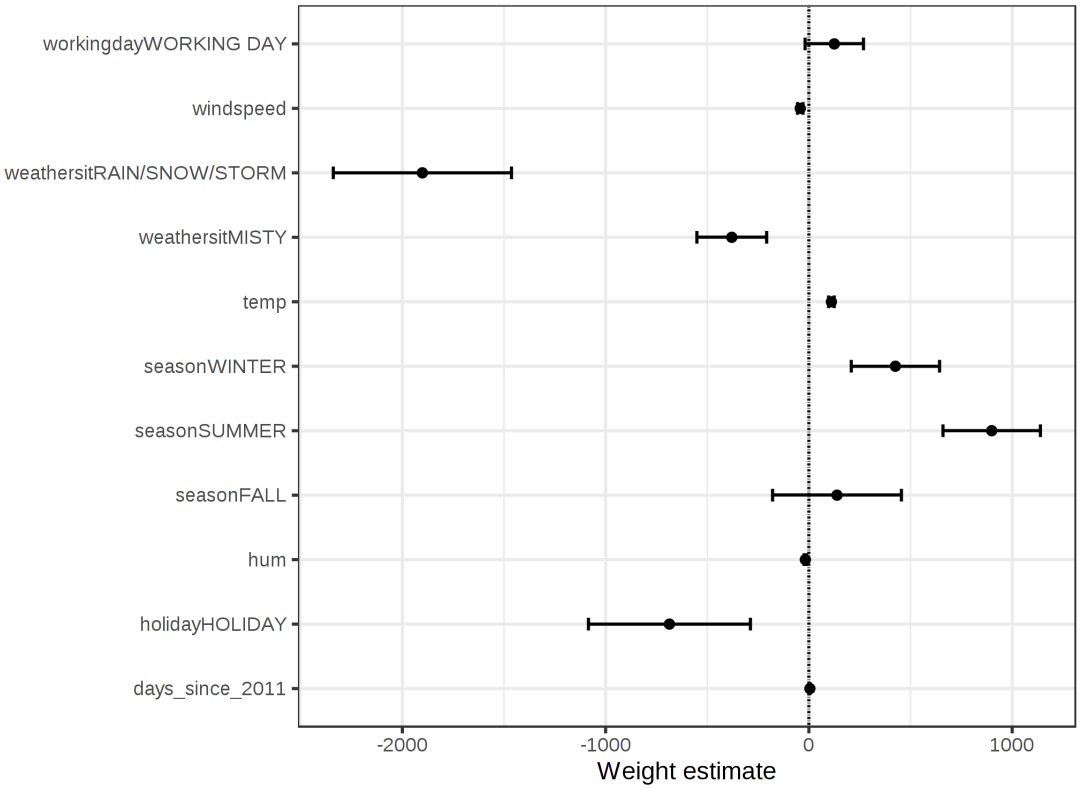


图 4.1. 权重显示为点，95% 置信区间显示为线。

权重图显示，下雨/下雪/暴风雨天气对预测的自行车数量有很大的负效应。“工作日” 特征 (workingday) 的权重接近于零，并且 95% 的区间中包含零，这意味着该效应在统计上不显著。一些置信区间很短，估计值接近于零，但特征效应在统计上是显着的。温度就是这样的一个例子。权

重图的问题在于特征是在不同的尺度上测量的。虽然对于天气状况，估计的权重反映了好天气和下雨/暴风雨/下雪天气之间的差异，但温度只反映了 1 摄氏度的增加。在拟合线性模型之前，可以通过缩放特征 (0 均值和 1 标准差)，使估计的权重更具可比性。

效应图 (Effect Plot)

当线性回归模型的权重与实际特征值相乘时，可以更有意义地进行分析。权重取决于特征的比例，比如有一个测量人的身高的特征，如果测量单位从米转换到厘米，那么权重会有所不同。权重会改变，但在数据中的实际效应不会改变。了解数据中特征的分布也是很重要的，因为如果方差非常小，这意味着这个特征几乎在所有的实例中都有类似的贡献。效应图可以帮助了解权重和特征的组合对数据预测的贡献程度。首先计算特征效应 (也可称特征影响)，即每个特征的权重乘以实例的特征值：

$$\text{effect}_j^{(i)} = w_j x_j^{(i)}$$

使用箱线图可以可视化效应。箱线图中的框包含一半数据的特征效应范围 (效应值的 1/4 到 3/4 分位数)。框中的垂直线是中位数 (50% 的实例对预测的影响小于此值，另一半高于此值)。水平线延伸到 $\pm 1.5\text{IQR}/\sqrt{n}$，其中 IQR 是四分位数之间的范围 (3/4 分位数减去 1/4 分位数)。这些点是离群点。与每个类别都单独一条线的权重图相比，分类特征的效应可以总结为一个单独的箱线图。

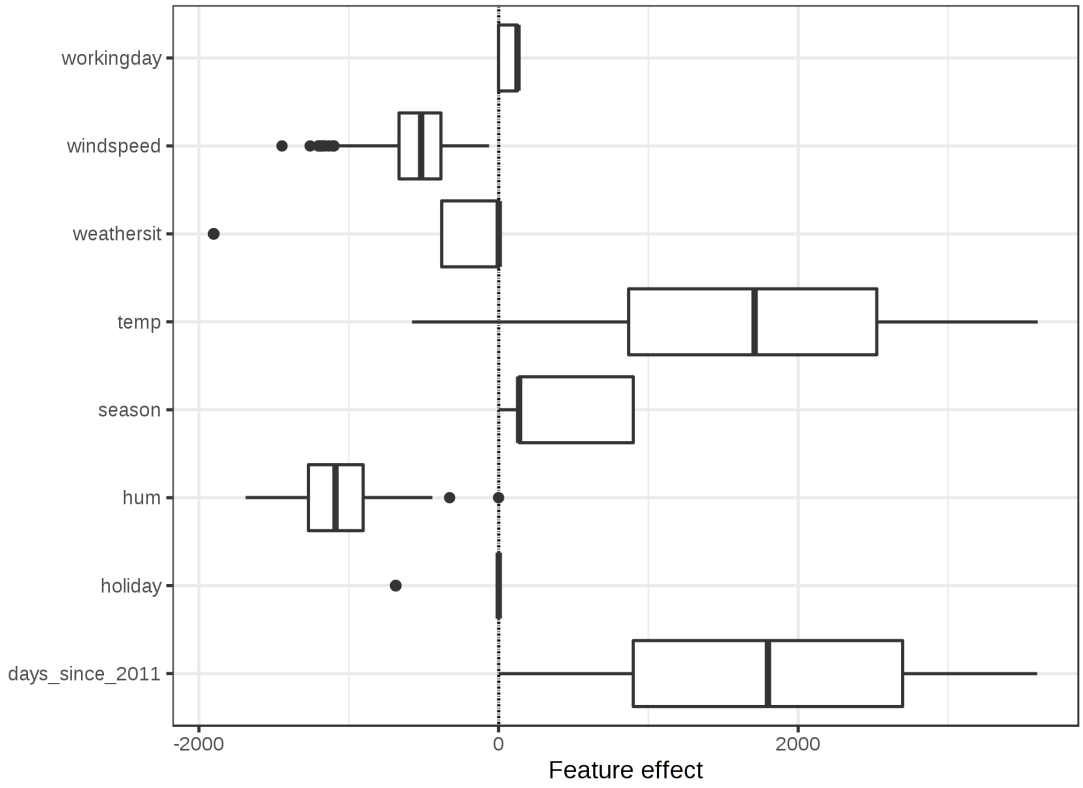


图 4.2. 特征效应图显示了每个特征的效应分布 (= 特征值乘以特征权重)。

对租用自行车预期数量的最大贡献来自于温度特征和天数特征，后者反映了自行车租赁随时间的趋势。温度在很大程度上有助于预测。天数特征从零到大的正的贡献，因为数据集中的第一天 (2011 年 1 月 1 日) 具有非常小的特征效应，并且该特征的估计权重为正 (4.93)。这意味着该特征效应每天都在增加，并且在数据集中的最后一天最高 (2012 年 12 月 31 日)。要注意负权重的特征效应，对于这些负权重，具有负特征值的实例表现出来的是正的效应。例如，风速有高负效应的时候就是风速高的日子。

4.1.4 解释单个实例预测

单个实例的各个特征对预测有多大贡献？这可以通过计算这个实例的特征效应得到答案。特定实例的特征效应的解释仅与各个特征的特征效应分布相比较才有意义。我们要解释来自自行车数据集的第 6 个实例的线性模型的预测。该实例具有以下特征值。

Feature	Value
season	SPRING
yr	2011
mnth	JAN
holiday	NO HOLIDAY
weekday	THU
workingday	WORKING DAY
weathersit	GOOD
temp	1.604356
hum	51.8261
windspeed	6.000868
cnt	1606
days_since_2011	5

为了获得这个实例的特征效应，我们必须将它的特征值乘以线性回归模型中相应的权重。对于 “工作日” 特征的值为 WORKING DAY，特征效应为 124.9。对于 1.6 摄氏度的温度，特征效应是 177.6。我们将这些单独的特征效应作为 “叉号” 添加到效应图中，效应图向我们展示了数据中的效应分布。这使我们能够将单个实例的特征效应与数据中的效应分布进行比较。如下图所示：

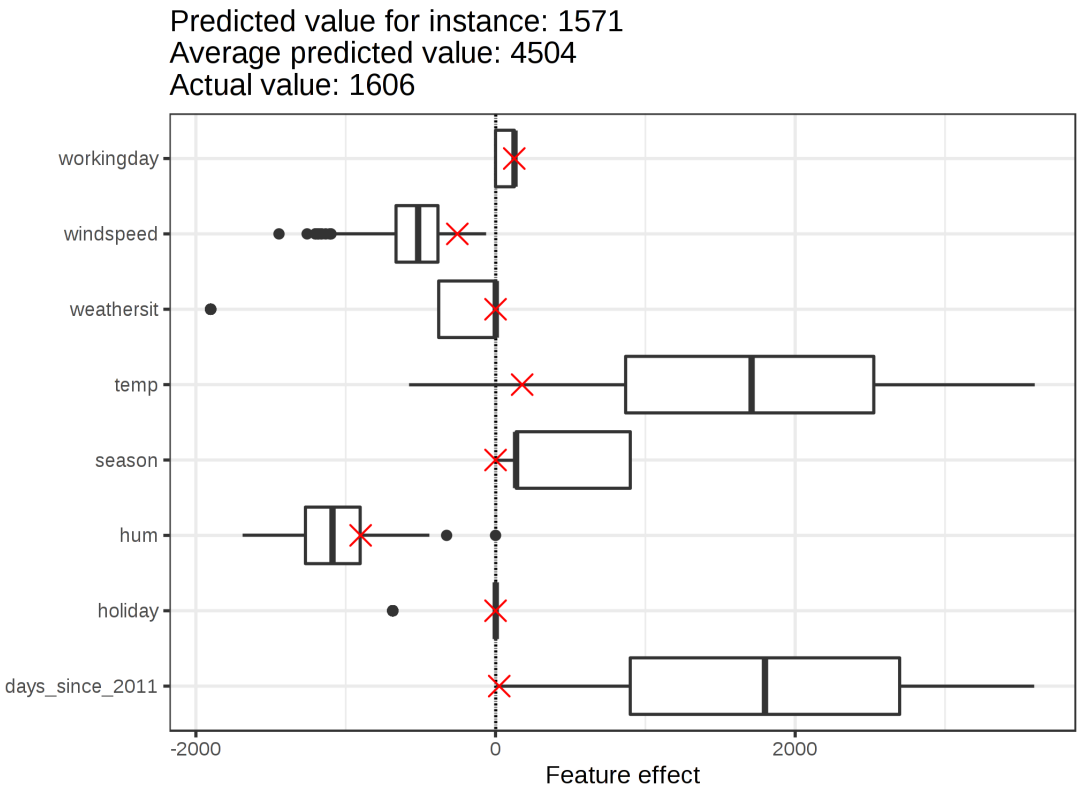


图 4.3. 一个实例的效应图显示了效应分布，并突出了感兴趣实例的特征效应。

如果我们对训练数据实例的预测求平均，得到的平均值是 4504。相比之下，第 6 个实例的预测值很小，因为预测的只有 1571 辆自行车租用。效应图揭示了原因。箱线图显示数据集所有实例的效应分布，“叉号” 显示第 6 个实例的效应。第 6 个实例的温度特征的特征效应较小，因为这一天的温度为 2 摄氏度，与其他大多数日期相比较低 (记住温度特征的权重为正)。此外，与其他数据实例相比，天数特征的特征效应也较小，因为该实例自第一天起仅过了 5 天，而且这个特征也是正的权重。

4.1.5 分类特征的编码

分类特征的编码有几种方法，不同的选择会影响权重的解释。

线性回归模型的标准是 Treatment Coding，这在大多数情况下已足够。使用不同的编码归根结底就是从具有分类特征的单个列中创建不同的 (设计) 矩阵。本节介绍了三种不同的编码，但还有更多。使用的示例有六个实例和三个类别的分类特征。对于前两个实例，该特征采用类别 A；对于实例 3 和 4，采用类别 B；对于后两个实例，采用类别 C。

Treatment Coding

在 Treatment Coding 中，每个类别的权重是对应类别和参照类别之间预测的估计差值。线性模型的截距是参照类别的平均值 (当所有其他特征保持不变时)。设计矩阵的第一列是截距，它始终是 1。第二列表示实例是否在 B 类中，第三列表示实例是否在 C 类中。A 类不需要列，因为此时只要知道一个实例既不属于 B 类也不属于 C 类就足够了，不然线性方程会被过度指定，并且找不到权重的唯一解。

特征矩阵：

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

Effect Coding

每个类别的权重是从相应类别到总体均值的估计 y 值差 (假定所有其他特征为零或参照类别)。第一列用于估计截距。与截距相关联的权重 β_0 表示总体均值，第二列的权重 β_1 表示总体均值与 B 类之间的差异。B 类的总体效应为 $\beta_0 + \beta_1$。同样地可以得到 C 类的解释。对于参照类别 A，$-(\beta_1 + \beta_2)$ 表示和总体均值的差异，则 $\beta_0 - (\beta_1 + \beta_2)$ 表示该类别的总体效应。

特征矩阵：

$$\begin{pmatrix} 1 & -1 & -1 \\ 1 & -1 & -1 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

Dummy coding

每个类别的 β 是这个类别 y 的估计均值 (假定所有其他特征值为零或参照类别)。请注意这里省略了截距，这样就可以找到线性模型权重的唯一解。

特征矩阵：

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

如果想深入了解分类特征的不同编码，可以参考 网页 和 博客。

4.1.6 线性模型是否有很好的解释?

从构成 “好的” 解释需要的性质 (见第二章人性化的解释) 来看，线性模型并不能创建最佳解释。它们是对比性的，但是参照实例是构造的一个数据点，其中所有数值特征都为零，分类特征设置为它们的参照类别，这通常是一个人工的、毫无意义的实例，不太可能出现在你的真实数据或现实中。有一种例外：如果所有数值特征均以均值为中心 (特征减去特征的平均值)，并且所有的分类特征都采用 Effect Coding 的，那么参照实例就是所有特征取平均特征值的数据点。这也可能是一个不存在的数据点，但至少更有可能或更有意义。在这种情况下，权重乘以特征值 (特征效应) 解释了与 “均值实例” 相比，对预测结果的贡献。一个 “好的解释” 的另一个要求是选择性，这可以在线性模型中通过使用较少的特征或训练稀疏的线性模型来实现。但默认情况下，只要线性方程是建模特征与结果间关系的合适的模型，线性模型就不会创建选择性解释，而是创建真实的解释。非线性和交互作用越多，线性模型越不准确，解释越不真实。线性使得解释更为普遍性和简单。我认为，模型的线性性质是人们使用线性模型解释关系的主要因素。

4.1.7 稀疏线性模型

前面我们选择的线性模型的例子看起来都很好，不是吗? 但实际上，在真实的情况下你可能不只是拥有少数个特征，而是拥有成百上千个特征。这种情况下你的线性回归模型的解释能力就会下降。你甚至可能会发现处于这样一种情况，即特征比实例多，并且根本无法拟合标准线性模型。幸运的是，有很多方法可以将稀疏性 (即很少的特征) 引入线性模型。

Lasso

Lasso 是一种将稀疏性引入线性回归模型的自动简便方法。Lasso 代表 “最小绝对收缩和选择算子”，当应用于线性回归模型时，执行特征选择和所选特征权重的正则化。让我们考虑权重优化的最小化问题 (注：这里的 x_i 表示的是第 i 个实例)：

$$min_{\boldsymbol{\beta}} \left(\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - x_i^T \boldsymbol{\beta})^2 \right)$$

Lasso 为这个优化问题添加了一项：

$$min_{\boldsymbol{\beta}} \left(\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - x_i^T \boldsymbol{\beta})^2 + \lambda ||\boldsymbol{\beta}||_1 \right)$$

其中，$||\boldsymbol{\beta}||_1$ 为特征向量的 L_1 范数，会对大权重进行惩罚。由于使用了 L_1 范数，许多权重的估计值为 0，而其他权重的估计值则缩小。参数 λ 控制正则化的强度，通常通过交叉验证进行调整。尤其是当 λ 很大时，许多权重变为 0。特征权重可以可视化为惩罚项 λ 的函数，下图中每个特征权重用一条曲线表示。

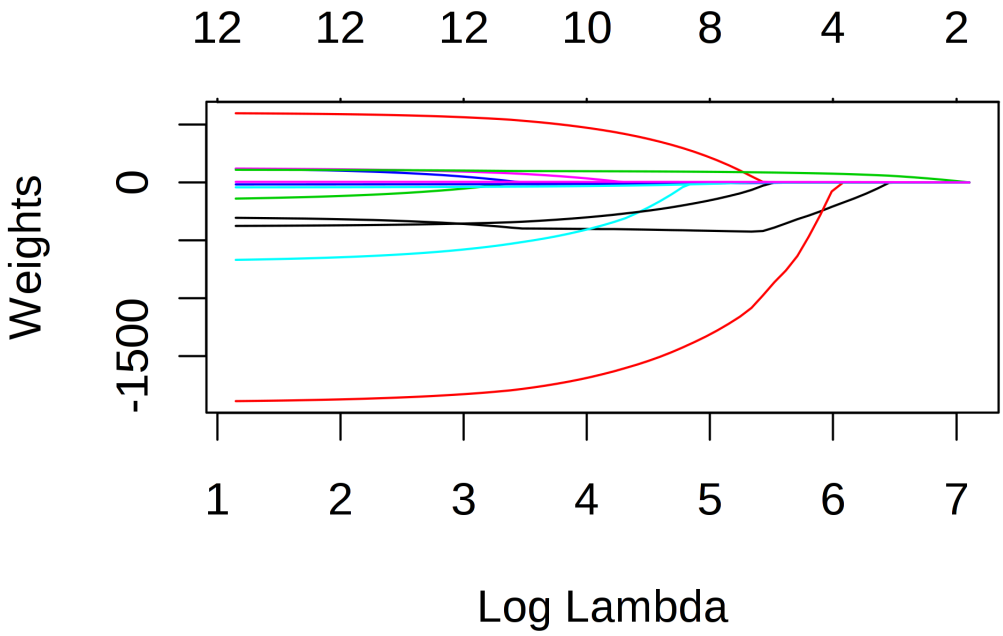


图 4.4. 随着权重惩罚的增加，越来越少的特征接受非零权重估计。这些曲线也称为正则化路径。图上的数字是非零权重的数目。

我们应该为 λ 选择什么值？如果将惩罚项视为一个优化参数，那么可以通过交叉验证找到将模型错误最小化的 λ。还可以将 λ 视为控制模型可解释性的参数，惩罚越大，模型中出现的特征就越少(因为它们的权重为零)，并且模型的解释效果越好。

Lasso 示例

我们将用 Lasso 预测自行车的租赁。我们预先设置了模型中需要的特征数量，让我们先设置为 2 个特征：

	Weight
seasonSPRING	0.00
seasonSUMMER	0.00
seasonFALL	0.00
seasonWINTER	0.00
holidayHOLIDAY	0.00
workingdayWORKING DAY	0.00
weathersitMISTY	0.00
weathersitRAIN/SNOW/STORM	0.00
temp	52.33
hum	0.00
windspeed	0.00
days_since_2011	2.15

前两个在 Lasso 路径中具有非零权重的特征是温度特征 (temp) 和天数特征 (days_since_2011)。

现在，让我们选择 5 个特征：

	Weight
seasonSPRING	-389.99
seasonSUMMER	0.00
seasonFALL	0.00
seasonWINTER	0.00
holidayHOLIDAY	0.00
workingdayWORKING DAY	0.00
weathersitMISTY	0.00
weathersitRAIN/SNOW/STORM	-862.27
temp	85.58
hum	-3.04
windspeed	0.00
days_since_2011	3.82

需要注意，温度和天数这两个特征的权重和前面那个模型的结果不一样。这是因为减少了 λ，权重就会受到较少的惩罚，权重的绝对值就可能更大。Lasso 权重的解释与线性回归模型中权重的解释相对应。你只需要注意特征是否标准化，因为这会影响权重。在这个示例中，这些特征已经被软件

标准化了，但权重会自动转换以匹配原始特征比例。

线性模型中稀疏性的其他方法

可以使用多种方法来减少线性模型中的特征数量。

预处理方法：

- 手动选择特征：你可以始终使用专家知识来选择或放弃某些特征。最大的缺点是它不能被自动化，需要与理解数据的人员取得联系。
- 单变量选择：相关系数就是一个例子。你只考虑超过特征和目标之间相关性阈值的特征。缺点是它只是单独地考虑单个特征。在线性模型考虑了其他一些特征之前，某些特征可能不会显示相关性。而使用单变量选择方法你会错过这些特征。

分步方法：

- 向前选择：用一个特征拟合线性模型，对每个特征都执行此操作。选择最有效的模型 (例如，最高 R-平方)；然后，对于其余的特征，通过将每个特征添加到当前的最佳模型中来拟合模型的不同版本，选择表现最好的一个；重复操作，直到达到某个条件，例如最大特征数。
- 向后选择：类似于向前选择。但是，不是添加特征，而是从包含所有特征的模型开始，尝试删除某个特征以达到性能最大程度的提高；重复此操作，直到达到某个停止标准。

这里，我们建议使用 Lasso，因为它可以自动化，同时考虑所有特征，并且可以通过 λ 进行控制。它同样也适用于分类 (逻辑回归)。

4.1.8 优点

将预测建模为一个**加权和**，使预测的生成变得透明。使用 Lasso，我们可以确保使用的特征数量仍然很小。

许多人使用线性回归模型。这意味着在许多地方，预测建模和推理都被**接受**。有**很高水平的集体经验和专业知识**，包括线性回归模型和软件实现的教材。线性回归可以在 R，Python，Java，Julia，Scala，Javascript 等等中找到。

从数学上讲，估计权重很简单，而且可以**保证找到最佳权重** (假设数据满足线性回归模型的所有假设)。

与权重一起，你还可以得到置信区间，检验和可靠的统计理论。线性回归模型也有许多扩展 (请参阅有关 GLM，GAM 等的章节内容)。

4.1.9 缺点

线性回归模型只能表示线性关系，即输入特征的加权和。而**每一个非线性或交互都必须是人工构成的**，并明确地作为输入特征提供给模型。

从预测性能角度来说，线性模型通常也不是那么好，因为可以学习的关系很有限，而且通常过于简单化了复杂的现实。

权重的解释可能**不直观**，因为它取决于所有其他特征。在线性模型中，与结果 y 和另一个特征都高度正相关的特征可能会得到负权重，这是因为在另一个相关特征下，它与高维空间中的 y 呈负相关。完全相关的特征使得我们甚至不可能找到线性方程的唯一解。例如，你用一个模型来预测房屋的价格，并且有一些特征，比如房间的数量和房屋的面积大小。房屋的面积大小和房间的数量是高度相关的：房屋面积越大，拥有的房间就越多。如果将这两个特征都纳入线性模型中，可能会发生这样的情况：房屋的面积大小是更好的预测因素，并且会得到很大的正权重。房间的数量最终可能会得到一个负的权重，因为考虑到房屋的面积大小相同，增加房间的数量可能会使它的价格降低，或者当相关性太强时，线性方程变得不稳定。

4.2 逻辑回归

逻辑回归 (`Logistic Regression`) 建模有两个可能结果的概率分类问题，它是针对分类问题的线性回归模型的扩展。

4.2.1 线性回归用于分类有什么问题?

线性回归模型能很好地建模回归，但建模分类就失败。为什么呢？对于两个类，可以将其中一个类标记为 0，另一个标记为 1，然后使用线性回归。从技术上讲，它是可行的，大多数线性模型程序都会得到权重。但这种方法存在一些问题：

线性模型不输出概率，但它将类视为数字 (0 和 1)，并拟合最佳超平面 (对于单个特征，它是一条线) 以最小化点和超平面之间的距离。所以它只是在点之间插值，而你不能把它解释为概率。

一个线性模型也会给出低于 0 和高于 1 的值，这就表明我们应该找到一个更好的分类方法。

由于预测的结果不是概率，而是点之间的线性插值，因此没有一个有意义的阈值可以用来区分一个类和另一个类。这一问题可以参考 Stackoverflow。

线性模型不能扩展到具有多个类的分类问题，你必须用 2、3 等等标记下一个类。类的顺序可能没有任何意义，但是线性模型会在特征和类别预测之间的关系上强制一个约束。具有正权重的特征值越高，它对具有更高编号的类的预测所起的作用就越大，即便相似编号的类并不比其他类更近。

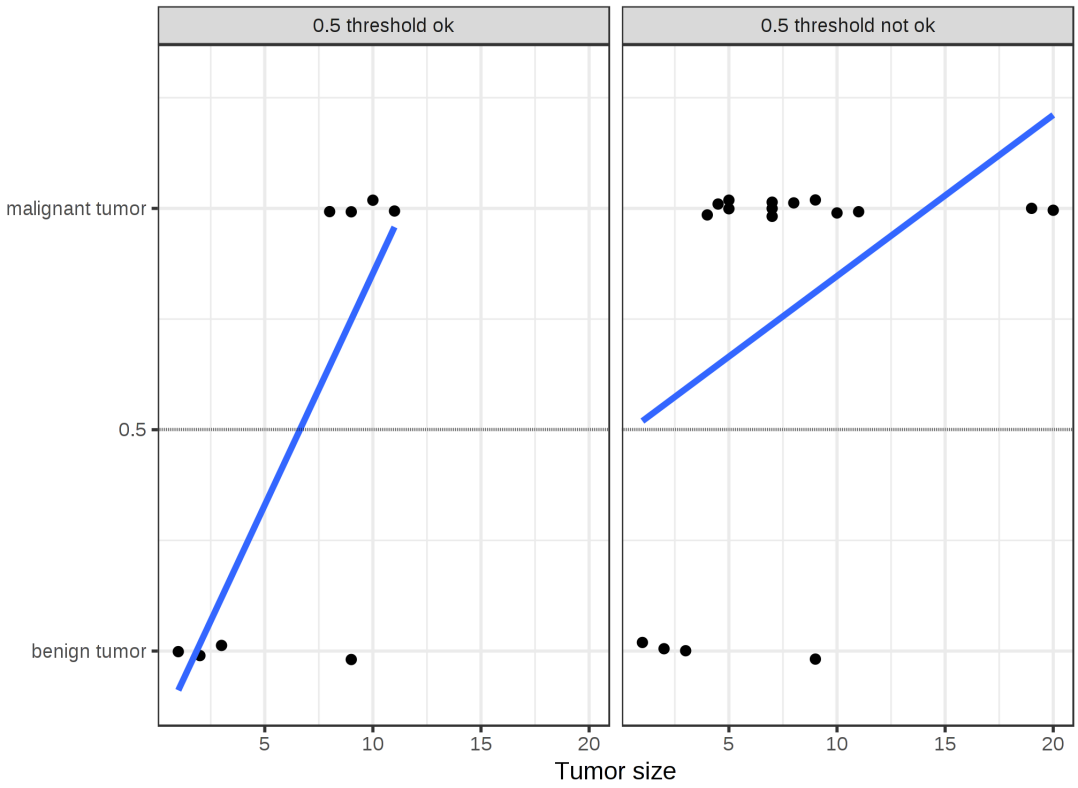


图 4.5. 线性模型根据肿瘤的大小将其分类为恶性 1 或良性 0，这些线条显示了线性模型的预测。对于左边的数据，我们可以使用 0.5 作为分类阈值；在引入更多的恶性肿瘤病例后，对于右边的数据，0.5 的阈值不再区分这两类。图中的点略有抖动，以减少过度绘图。

4.2.2 理论

分类的解决方案是逻辑回归。逻辑回归不是拟合直线或超平面，而是使用逻辑函数将线性方程的输出挤压到 0 和 1 之间。逻辑函数定义如下：

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)}$$

它看起来像这样：

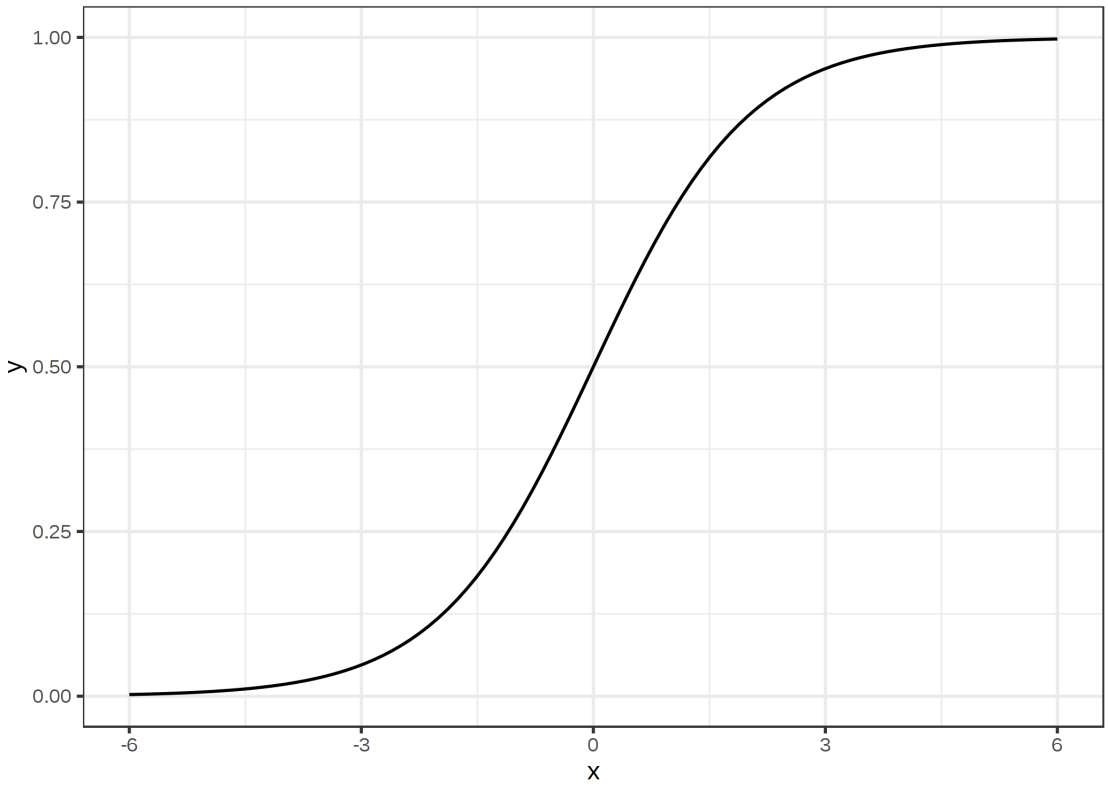


图 4.6. 逻辑函数。它输出 0 到 1 之间的数字。在输入 0 时，它输出 0.5。

从线性回归到逻辑回归的步骤较为简单。在线性回归模型中，我们用一个线性方程来建模结果和特征之间的关系：

$$\hat{y}^{(i)} = \beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)}$$

对于分类，我们倾向于 0 到 1 之间的概率，因此我们将方程的右边包装成逻辑函数，这将强制输出仅为 0 和 1 之间的值。

$$P(y^{(i)} = 1) = \frac{1}{1 + exp(-(\beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)}))}$$

让我们再次回顾一下肿瘤大小的例子。但我们使用的不是线性回归模型，而是逻辑回归模型。

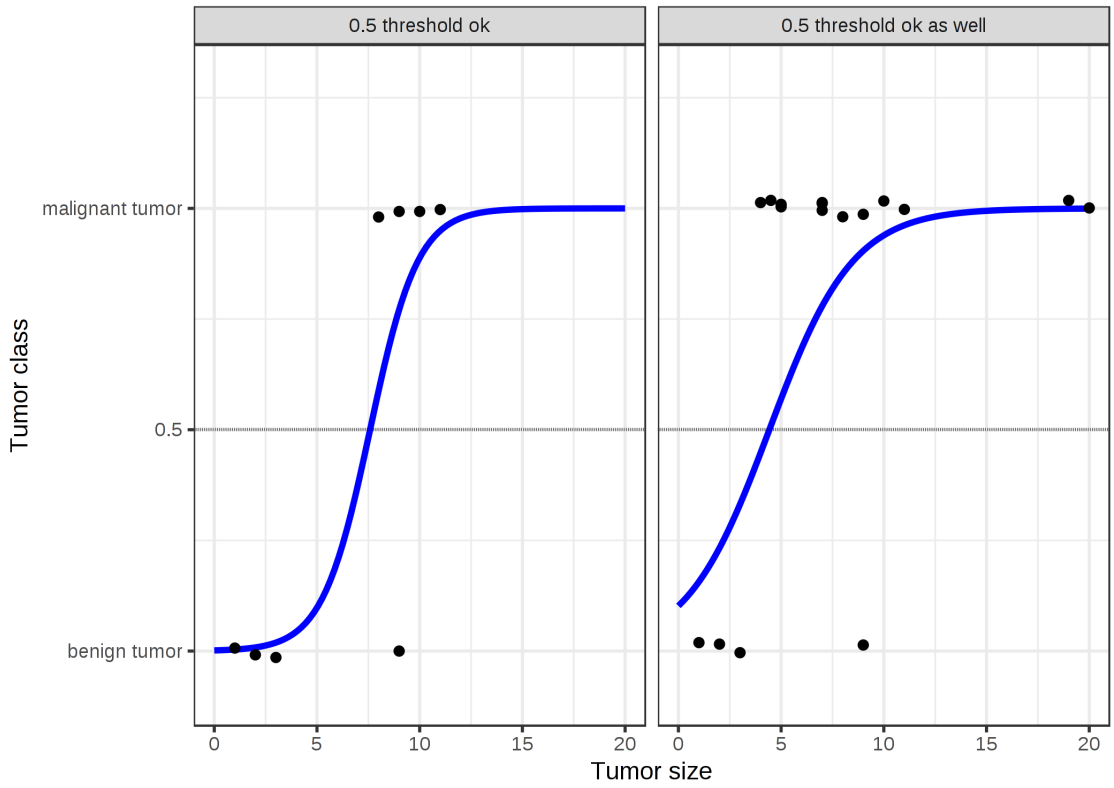


图 4.7. 逻辑回归模型根据肿瘤的大小在恶性和良性之间找到正确的决策边界。曲线为拟合数据对逻辑函数进行了移位和压缩。

使用逻辑回归进行分类效果更好，在两种情况下我们都可以使用 0.5 作为阈值。附加的点并不会真正影响估计曲线。

4.2.3 解释

由于逻辑回归的结果是 0 到 1 之间的概率，逻辑回归中权重的解释不同于线性回归中权重的解释。权重不再线性地影响概率，加权和由逻辑函数转换为概率。因此，我们需要为解释重新构造方程，以便只有线性项在公式的右边。

$$log\left(\frac{P(y=1)}{1-P(y=1)}\right) = log\left(\frac{P(y=1)}{P(y=0)}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

我们将 log() 函数中的项称为几率 (`odds`) (事件发生概率除以事件不发生概率)，并用对数表示，称为对数几率 (`log odds`)。

该公式表明，逻辑回归模型是对数几率的线性模型。现在，只要稍微改变一下这些项，我们就可以知道当特征 x_j 改变 1 个单位时，预测是如何变化的。为此，我们首先可以将 exp() 函数作用于公

式的两边：

$$\frac{P(y=1)}{1-P(y=1)} = odds = exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p\right)$$

然后我们比较当我们将某个特征值增加 1 时会发生什么。但是，我们不看差异，而是看两个预测的比率：

$$\frac{odds_{x_j+1}}{odds} = \frac{exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_j(x_j+1) + \ldots + \beta_p x_p\right)}{exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_j x_j + \ldots + \beta_p x_p\right)}$$

我们应用以下规则：

$$\frac{exp(a)}{exp(b)} = exp(a-b)$$

然后删除一些项：

$$\frac{odds_{x_j+1}}{odds} = exp\left(\beta_j(x_j+1) - \beta_j x_j\right) = exp\left(\beta_j\right)$$

最后，我们得到了一个简单的特征权重 exp()。一个特征改变 1 个单位将会使几率比 (`odds ratio`) 改变 $\exp(\beta_j)$。我们也可以这样解释：特征 x_j 改变 1 个单位会增加对数几率比 (`log odds ratio`) 相应权重的值。大多数人解释几率比是因为人们觉得思考 log() 对大脑来说很困难，而且光是解释几率比已经需要一些习惯。例如，如果几率 (odds) 为 2，则表示 y=1 的概率是 y=0 的两倍。如果你有一个权重为 0.7，则将相应的特征增加 1 个单位，几率将乘以 $\exp(0.7)$ (约 2)，也就是几率将变为 4。但通常你不用处理几率，只要把权重解释为几率比。因为为了实际计算几率，你需要为每个特征设置一个值，这只有在想查看数据集的一个特定实例时才有意义。

这些是具有不同特征类型的逻辑回归模型的解释：

- 数值特征：如果你将特征 x_j 增加一个单位，则估计的几率将乘以因子 $\exp(\beta_j)$。
- 二分类特征：只取两种可能值的特征，其一是参照类别 (用 0 编码)。将特征 x_j 从参照类别更改为其他类别，则估计的几率将乘以因子 $\exp(\beta_j)$。
- 具有多个类别的分类特征：具有固定数量的可能值的特征。处理多个类别的解决方案是 one-hot 编码，这意味着每个类别都有自己的列。对于具有 L 个类别的分类特征，只需要 L − 1 列，因为第 L 列将具有冗余信息 (例如，当列 1 到 L − 1 的值都为 0 时，我们知道此实例的分类特征为第 L 个类别)。然后对每个类别的解释与对二分类特征的解释相同。
- 截距项 β_0：所有数字特征为零和分类特征为参照类别，则估计的几率是 $\exp(\beta_0)$。截距的解释通常不相关。

4.2.4 示例

我们使用逻辑回归模型基于一些风险因素来预测宫颈癌。下表显示了估计权重、相关的几率比和估计的标准差。

	Weight	Odds ratio	Std. Error
Intercept	-2.91	0.05	0.32
Hormonal contraceptives y/n	-0.12	0.89	0.30
Smokes y/n	0.26	1.29	0.37
Num. of pregnancies	0.04	1.04	0.10
Num. of diagnosed STDs	0.82	2.26	0.33
Intrauterine device y/n	0.62	1.85	0.40

数字特征的解释 (性病诊断次数/Num. of diagnosed STDs)：当所有其他特征保持不变时，诊断的性传播疾病 (STDs) 数量的增加改变 (增加) 了癌症对非癌症的几率 2.26 倍。要注意，相关性并不意味着因果关系。

分类特征的解释 (是否服用激素避孕药/Hormonal contraceptives y/n)：对于使用激素避孕药的女性，与没有激素避孕药的女性相比，癌症与没有癌症的几率低 0.89 倍。

当然，就像线性模型一样，我们在解释的时候总是伴随着 “所有其他特征保持不变” 这一条件。

4.2.5 优缺点

线性模型的很多优点和缺点也适用于逻辑回归模型。

逻辑回归已被许多不同的人广泛使用，但它难以克服其限制性的表达能力 (例如，必须手动添加交互)，其他模型可能具有更好的预测性能。

逻辑回归模型的另一个缺点是解释更困难，因为权重的解释是乘法而不是加法。

逻辑回归可能受到完全分离 (`Complete Separation`) 的影响。如果有一个特征能够将两个类完全分开，那么逻辑回归模型就不能再被训练了。这是因为该特征的权重不会收敛，因为最佳权重将是无限的。这真的有点不幸，因为这样的特征真的很有用。但是如果你有一个简单的规则将两个类分开，其实就不需要机器学习了。完全分离问题可以通过引入权重的惩罚或定义权重的先验概率分布来解决。

从好的方面来说，逻辑回归模型不仅是一个分类模型，而且还给出概率。与只能提供最终分类的模型相比，这是一个很大的优势。知道一个实例属于某个类有 99% 的概率，和有 51% 的概率是很大不同的。

逻辑回归也可以从二分类扩展到多分类，后者被称为多分类回归 (`Multinomial Regression`)。

4.2.6 软件

我在所有示例中都使用了 R 中的 glm 函数。你可以在任何可用于执行数据分析的编程语言中找到逻辑回归，例如 Python，Java，Stata，Matlab 等等。

4.3 GLM, GAM 和其他模型

线性回归模型最大的优点也是最大的缺点就是预测被建模为特征的加权和。此外，线性模型还有许多其他假设。坏的情况就是所有这些假设在现实中经常被违背：给定特征的结果可能具有非高斯分布，特征可能交互，特征和结果之间的关系可能是非线性的。好消息是，统计界已经进行了各种修改，将线性回归模型从简单的刀片转换为瑞士刀。

本节不是扩展线性模型的明确指南，而是作为扩展的概述，例如广义线性模型 (`Generalized Linear Models`，GLMs) 和广义加性模型 (`Generalized Additive Models`，GAMs)。阅读之后，你可以对如何扩展线性模型有个全面的了解。如果你想首先了解有关线性回归模型的更多信息，建议你阅读有关线性回归模型的小节 (如果还没有的话)。

我们回顾下线性回归模型的公式：

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \epsilon$$

线性回归模型假设一个实例的结果 y 可以用它的 p 个特征的加权和表示，其中误差项 ϵ 遵循高斯分布。通过将数据强制到公式中，我们可以获得模型解释。特征的效应是加性的，意味着没有特征交互；而且关系是线性的，意味着某个特征增加 1 个单位可以直接转化为预测结果的增加/减少。线性模型允许我们将特征和预期结果之间的关系压缩成一个单一数字，即估计的权重。

但对于许多现实世界的预测问题来说，简单的加权和太过严格。在本节中，我们将学习经典线性回归模型的三个问题以及如何解决它们。可能违反的假设问题还有很多，但我们将重点关注下图所示的三个问题：

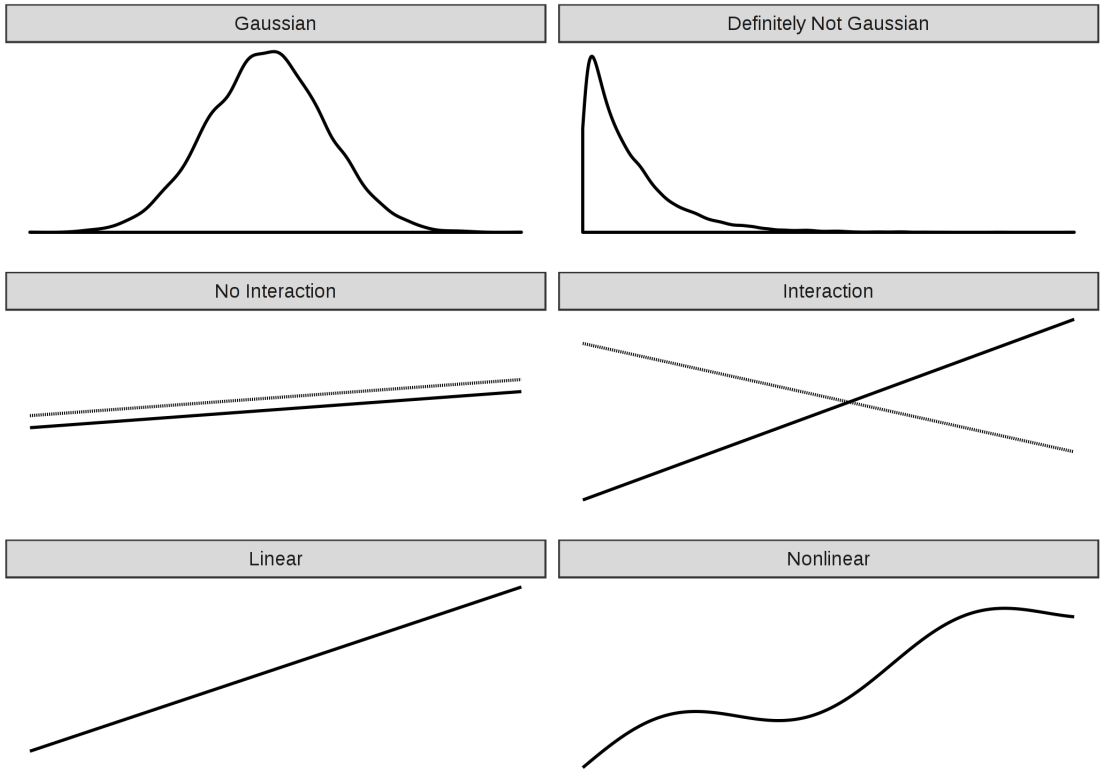


图 4.8. 线性模型的三个假设 (左侧)：给定特征的输出结果为高斯分布，可加性 (相当于无交互) 和线性关系。现实通常不遵循这些假设 (右侧) ： 结果可能具有非高斯分布，特征可能会交互并且关系可能是非线性的。

所有这些问题都有解决方案：

1. 问题：给定特征的目标结果 y 不遵循高斯分布。

示例：假设我想预测我在某天内骑自行车的时间。给定的特征有这一天的类型，天气情况等等。如果我使用线性模型，它可以预测负的分钟，因为它假设高斯分布，不会在 0 分钟停止。同样如果我想用线性模型预测概率，我可以得到负的或大于 1 的值。

解决方案：广义线性模型 (`Generalized Linear Models`，GLMs)。

2. 问题：特征交互。

示例：一般来说，小雨对我骑车的欲望有轻微的负面影响。但是在夏天，在交通高峰期，我还是希望下雨，因为那时所有晴天时的自行车骑手都呆在家里，我就可以有自己的自行车道！这是一种时间和天气之间的交互，无法由纯加性模型获得。

解决方案：手动添加交互。

3. 问题：特征和 y 之间的真实关系不是线性的。

示例：在 0 到 25 摄氏度之间，温度对我骑车欲望的影响可能是线性的，这意味着从 0 到 1 摄氏度的增加导致骑车欲望的增加与从 20 到 21 摄氏度的增加相同。但在较高的温度下，我骑车的动机会减弱甚至降低——我不喜欢在太热的时候骑车。

解决方案：广义加性模型 (`Generalized Additive Models`，GAMs)；特征转换

下面会着重介绍这三个问题的解决方案。线性模型的许多进一步扩展这边被省略了。如果我试图覆盖这里的所有内容，那么本书很快就会变成关于这个主题的一本书，而该主题已经在许多其他书籍中涉及。但既然你已经看到这里了，所以在线性模型扩展方面会遇到的一些问题以及解决方案概述可以在本节末中进一步描述。

4.3.1 非高斯结果输出 - GLM

线性回归模型假定给定输入特征的结果遵循高斯分布。这一假设排除了许多情况：结果也可以是一个类别 (癌症对健康)、一个计数 (儿童数量)、事件发生的时间 (机器故障的时间) 或只有少数非常高的值的扭曲的结果 (家庭收入)。线性回归模型可以扩展为对所有这些类型的结果建模，这个扩展称为**广义线性模型**，简称 GLM。在本节中，我们将对通用框架和该框架中的特定模型使用 GLM 这个名称。任何 GLM 的核心概念都是：保留特征的加权和，但允许非高斯结果分布，并通过可能的非线性函数连接该分布的期望均值与加权和。例如，逻辑回归模型假设结果为伯努利分布，并使用 Logit 函数连接期望均值与加权和。

GLM 数学上使用连接函数 g 将特征的加权和与假定分布的均值连接起来，其中连接函数可以根据结果的类型可以自由选择：

$$g(E_Y(y|x)) = \beta_0 + \beta_1 x_1 + \ldots \beta_p x_p$$

GLM 由三个部分组成：连接函数 g、加权和 $X^T\beta$ (有时称为线性预测因子) 以及定义 E_Y 指数族的概率分布。

指数族是一组可以用相同的 (参数化的) 公式构成的分布，其中包括指数、分布的均值和方差以及一些其他参数。这边我们不会深入研究数学细节，因为这是一个非常大的探索领域。维基百科有一个 指数族分布列表，这个列表中的分布都可以作为 GLM 的选择。根据你要预测的结果类型，可以选择合适的分布。结果是对某事的计数吗? (例如一个家庭中孩子的数量) 那么，泊松分布可能是一个不错的选择。结果是否总是正的? (例如两次事件之间的时间) 那么，指数分布可能是一个不错的选择。

让我们把经典线性模型看作是 GLM 的一个特例。经典线性模型中高斯分布的连接函数是一个简单的恒等函数。高斯分布由均值和方差参数进行参数化。均值描述了我们的期望均值，而方差则描述了在均值上下值的变化程度。在线性模型中，连接函数会连接特征加权和和高斯分布的均值。

在 GLM 框架下，这个概念推广到任何分布 (指数族) 和任意连接函数。如果 y 是某物的计数，例如某人某天喝的咖啡数量，我们可以用一个带有泊松分布和自然对数作为连接函数的 GLM 来建模：

$$ln(E_Y(y|x)) = x^T\beta$$

逻辑回归模型也是一个假定伯努利分布并使用 Logit 函数作为连接函数的 GLM。逻辑回归中使用的二项分布的均值是 y 为 1 的概率。

$$x^T\beta = ln\left(\frac{E_Y(y|x)}{1-E_Y(y|x)}\right) = ln\left(\frac{P(y=1|x)}{1-P(y=1|x)}\right)$$

如果我们把这个方程解 $P(y=1)$ 放在一边，我们得到逻辑回归公式：

$$P(y=1) = \frac{1}{1+exp(-x^T\beta)}$$

指数族的每一个分布都有一个标准的连接函数，可以从分布中数学地推导出来。GLM 框架使选择独立于分布的连接函数成为可能。如何选择正确的连接函数？没有完美的配方。你要考虑到关于目标分布的知识，同时也要考虑到理论上的考虑，以及模型与实际数据的匹配程度。对于某些分布，标准的连接函数可能导致对该分布无效的值。因为你可以选择任何连接函数，所以简单的解决方案是选择另一个尊重分布域的函数。

示例

我们模拟了一个关于咖啡饮用行为的数据集，用于强调对 GLM 的需求。假设你已经收集了关于每天喝咖啡行为的数据 (如果你不喜欢咖啡，就假装是茶)。除了杯子的数量外，你还记录了目前的压力水平 (从 1 到 10)，前一天晚上的睡眠状况 (从 1 到 10)，以及当天是否必须工作。目的是根据压力、睡眠和工作来预测咖啡的数量。我们模拟了 200 天的数据，在 1 到 10 之间均匀地绘制压力和睡眠，并且以 0.5/0.5 的概率绘制 “是/否” 工作 (真是命啊！)。然后，每天从泊松分布中提取咖啡的数量，将强度 λ (也是泊松分布的期望值) 建模为睡眠、压力和工作特征的函数。估计你可以猜到这个故事的结局：“嘿，让我们用一个线性模型来建模这个数据……哦，它不起作用……让我们用泊松分布的 GLM……惊喜！现在可以了！“。

让我们看看目标变量的分布，即给定日期的咖啡数量：

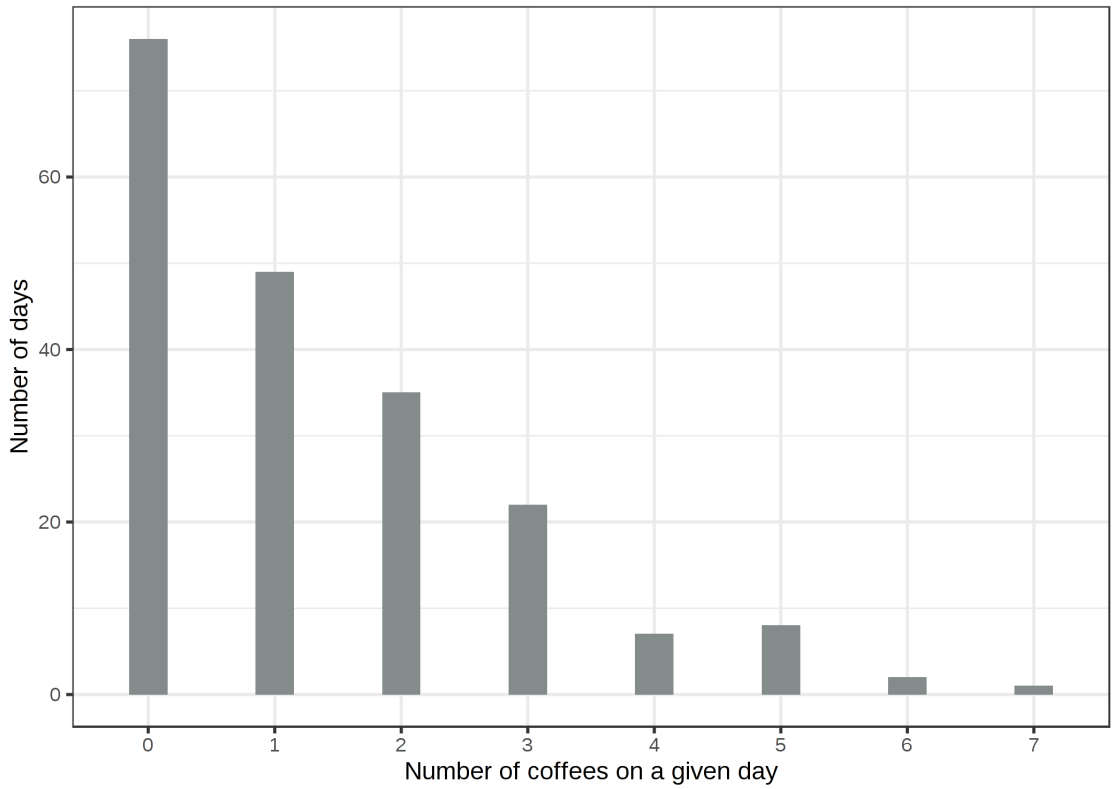


图 4.9. 200 天的每日咖啡数量的模拟分布。

在 200 天中的 76 天根本没有喝咖啡，在最极端的一天喝了 7 杯。让我们先使用一个线性模型来预测咖啡的数量，使用睡眠水平、压力水平和是/否工作作为特征。当我们错误地假设高斯分布时，会发生什么错误？错误的假设可能会使估计无效，尤其是权重的置信区间。更加明显的问题是，预测与真实结果的“允许”范围不匹配，如下图所示。

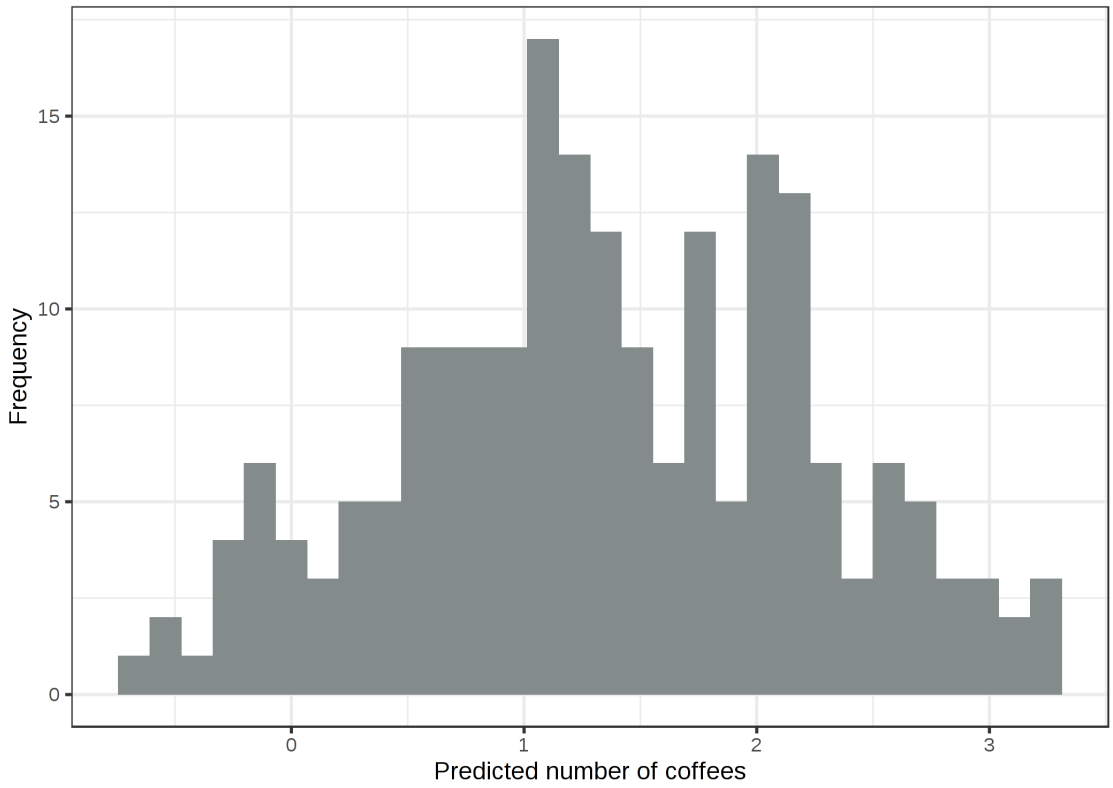


图 4.10. 预测咖啡的数量取决于压力，睡眠和工作。线性模型预测负值。

线性模型没有意义，因为它预测咖啡的数量为负。这个问题可以用广义线性模型 GLM 来解决。我们可以改变连接函数和假定的分布。一种可能是保持高斯分布，并使用一个总是导致正预测的连接函数，例如对数连接 (逆函数是 exp 函数) 而不是恒等函数。更好的方案是：我们选择一个与数据生成过程相对应的分布和一个适当的连接函数。由于结果是一个计数，泊松分布是一个自然选择，同时对数作为连接函数。在这种情况下，数据甚至是由泊松分布生成的，因此泊松 GLM 是最佳选择。拟合的泊松 GLM 导致预测值的以下分布：

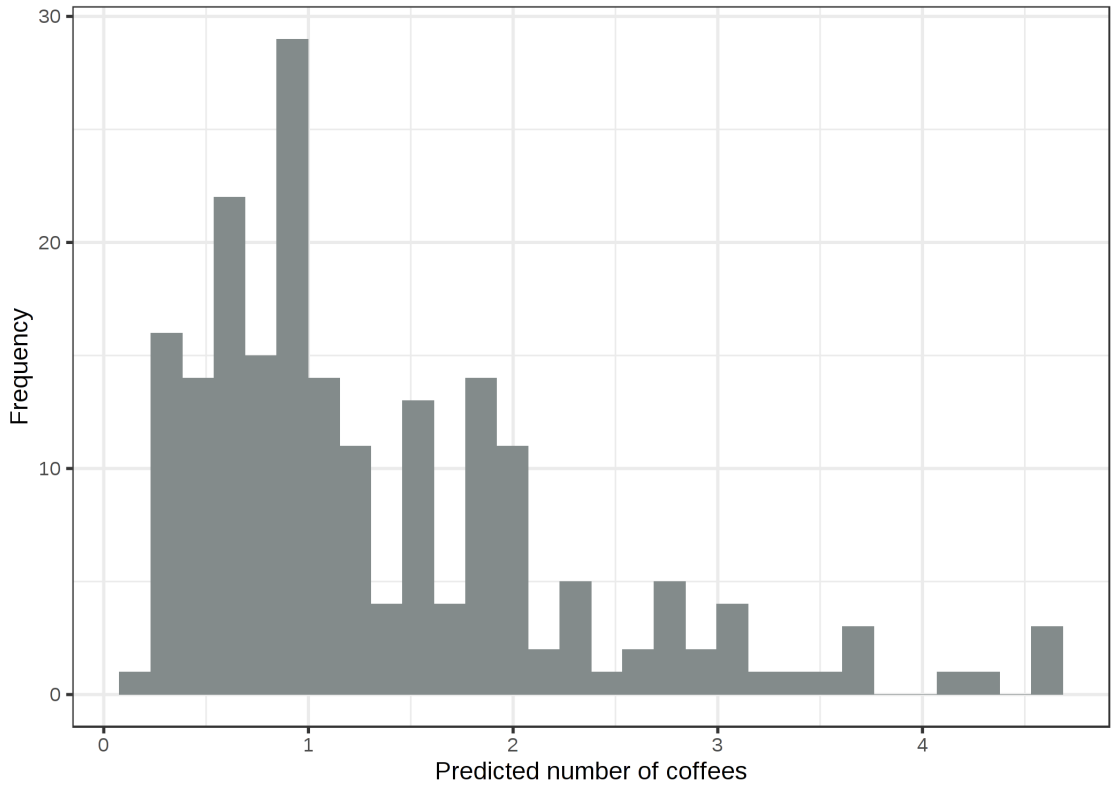


图 4.11. 预测的咖啡数量取决于压力，睡眠和工作。具有泊松假设和对数连接的 GLM 是适用于此数据集的模型。

没有负数量的咖啡，现在看起来好多了。

对 GLM 权重的解释

假设分布和连接函数共同决定了如何解释估计的特征权重。在咖啡计数的示例中，我们使用了一个带有泊松分布和对数连接的 GLM，这意味着特征和期望结果之间存在以下关系。

$$ln(E(\text{coffees}|\text{stress}, \text{sleep}, \text{work})) = \beta_0 + \beta_{\text{stress}}x_{\text{stress}} + \beta_{\text{sleep}}x_{\text{sleep}} + \beta_{\text{work}}x_{\text{work}}$$

为了解释权重，我们对连接函数取逆，这样我们就可以解释特征对期望结果的影响，而不是对期望结果的对数的影响。

$$E(\text{coffees}|\text{stress}, \text{sleep}, \text{work}) = exp(\beta_0 + \beta_{\text{stress}}x_{\text{stress}} + \beta_{\text{sleep}}x_{\text{sleep}} + \beta_{\text{work}}x_{\text{work}})$$

由于所有权重都在指数函数中，因此特征影响的解释不是加性的，而是乘性的，因为 $\exp(a+b) = \exp(a)\exp(b)$。解释的最后一个成分是虚构样本的实际权重。下表列出了估计的权重和 exp(权重)以及 95% 的置信区间：

	weight	exp(weight) [2.5%, 97.5%]
(Intercept)	-0.16	0.85 [0.54, 1.32]
stress	0.12	1.12 [1.07, 1.18]
sleep	-0.15	0.86 [0.82, 0.90]
workYES	0.80	2.23 [1.72, 2.93]

压力水平提高一点，咖啡的预期数量乘以系数 1.12。将睡眠质量提高一点，咖啡的预期数量乘以系数 0.86。工作日的预测咖啡数量平均是休息日咖啡数量的 2.23 倍。总之，压力越大，睡眠越少，工作越多，咖啡消耗越多。

在本小节中，你了解了一些关于当目标不遵循高斯分布时有用的广义线性模型的知识。接下来，我们将研究如何将两个特征之间的交互集成到线性回归模型中。

4.3.2 交互

线性回归模型假设一个特征的效应是保持相同的，与其他特征的值无关 (= 没有交互)。但数据中经常有交互作用，为了预测自行车租赁数量，“温度” 和 “是否工作日” 之间可能存在交互作用。也许，当人们不得不工作时，温度对租用自行车数量的影响不大，因为不管发生什么，人们都会骑着租用的自行车去工作。在休息日，许多人为了享乐而骑车，但前提天气足够暖和。当涉及到自行车租赁时，我们可能希望温度和工作日之间的交互作用。

我们如何才能得到包含交互作用的线性模型？在拟合线性模型之前，要在特征矩阵中添加一列，表示特征之间的交互，并像往常一样拟合模型。该解决方案在某种程度上很优雅，因为它不需要对线性模型进行任何更改，只需要数据中的附加的列。在工作日和温度示例中，我们将添加一个新特征，该特征在非工作日为零，否则它具有温度特征的值 (假定工作日为参照类别)。假设我们的数据如下：

work	temp
Y	25
N	12
N	30
Y	5

线性模型使用的数据矩阵看起来略有不同。下表显示了如果我们不指定任何交互，为模型准备的数据是什么样子的。通常，这种转换是由软件自动执行的。

Intercept	workY	temp
1	1	25
1	0	12
1	0	30
1	1	5

第一列是截距项，第二列对分类特征进行编码 (参照类别为 0，其他类别为 1)，第三列为温度特征。

如果我们希望线性模型考虑温度和工作日特征之间的交互作用，我们必须为交互作用添加一列：

Intercept	workY	temp	workY.temp
1	1	25	25
1	0	12	0
1	0	30	0
1	1	5	5

新的列 “workY.temp” 捕获了特征工作日和温度之间的交互。对于一个实例，如果工作特征处于参照类别 (“N” 表示非工作日)，则此新特征列为 0，否则为实例的温度特征的值。使用这种编码方式，线性模型可以学习两种工作日下的温度的不同线性效应。这是两个特征之间的交互作用。在没有交互项的情况下，分类特征和数值特征的组合效应可以描述成一条针对不同类别垂直移动的线。如果我们将交互包括在内，我们就允许数值特征 (斜率) 的影响在每个类别中具有不同的值。

两个分类特征的交互作用也有类似的效果。我们创建了表示类别组合的附加特征。以下是一些包含工作日 (work) 和分类天气特征 (wthr) 的人工数据：

work	wthr
Y	2
N	0
N	1
Y	2

接下来，我们将包括交互项：

Intercept	workY	wthr1	wthr2	workY.wthr1	workY.wthr2
1	1	0	1	0	1

Intercept	workY	wthr1	wthr2	workY.wthr1	workY.wthr2
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	1	0	1

第一列用于估计截距；第二列是编码了工作日特征；第三列和第四列表示天气特征，这需要两列，因为需要两个权重来捕获三个类别的效果 (其中一个类别是参照类别)；剩余的两列捕获交互。对于这两个特征的每个类别 (参照类别除外)，如果两个特征都有一个特定的类别，我们将创建一个新的特征列 (即 1)，否则为 0。

对于两个数值特征，交互列更容易构造：我们简单地将两个数值特征相乘。

有一些方法可以自动检测和添加交互项，其中之一可以参考 RuleFit。RuleFit 算法首先挖掘交互项，然后估计包含交互的线性回归模型。

示例

让我们回到在线性模型一节中已经建模的自行车租赁预测任务。这一次，我们还考虑了温度和工作日特征之间的交互作用，这导致以下估计的权重和置信区间。

	Weight	Std. Error	2.5%	97.5%
(Intercept)	2185.8	250.2	1694.6	2677.1
seasonSUMMER	893.8	121.8	654.7	1132.9
seasonFALL	137.1	161.0	-179.0	453.2
seasonWINTER	426.5	110.3	209.9	643.2
holidayHOLIDAY	-674.4	202.5	-1071.9	-276.9
workingdayWORKING DAY	451.9	141.7	173.7	730.1
weathersitMISTY	-382.1	87.2	-553.3	-211.0
weathersitRAIN/…	-1898.2	222.7	-2335.4	-1461.0
temp	125.4	8.9	108.0	142.9
hum	-17.5	3.2	-23.7	-11.3
windspeed	-42.1	6.9	-55.5	-28.6
days_since_2011	4.9	0.2	4.6	5.3
workingdayWORKING DAY:temp	-21.8	8.1	-37.7	-5.9

额外的交互效应是负的 (-21.8)，与零有显著差异，如 95% 置信区间所示，其中不包括零。顺便说一句，数据不是独立同分布的，因为彼此接近的日子并不是彼此独立的。就拿它来说，置信区间可能会产生

误导。交互项改变了所涉及特征的权重解释。如果是工作日，温度会有负效应吗? 答案是否定的，即使表格建议未经培训的用户使用也可以。我们不能孤立地解释 “workingdayWORKING DAY:tem” 的交互权重，因为这样的解释是：“在保持所有其他特征值不变的情况下，增加工作日温度的交互作用会减少自行车的预测数量。” 但是，交互项的效应只是添加到温度的主要效应中。假设这是一个工作日，我们想知道如果今天的温度升高 1 度会发生什么。那么我们需要将 “temp” 和 “workingdayWORKING DAY:tem” 的权重相加，以确定估计值增加了多少。

从视觉上更容易理解交互。通过引入分类特征和数值特征之间的交互项，我们得到了温度的两个斜率，而不是一个。非工作日 (“NO WORKING DAY”) 的温度斜率可直接从表中读取 (125.4)。工作日 (“WORKING DAY”) 的温度斜率是两个温度权重的总和 (125.4-21.8=103.6)。在温度等于 0 时，“NO WORKING DAY” 线的截距由线性模型 (2185.8) 的截距项确定。在温度等于 0 时，“WORKING DAY” 线的截距由截距项 + 工作日的效应 (2185.8+451.9=2637.7) 确定。

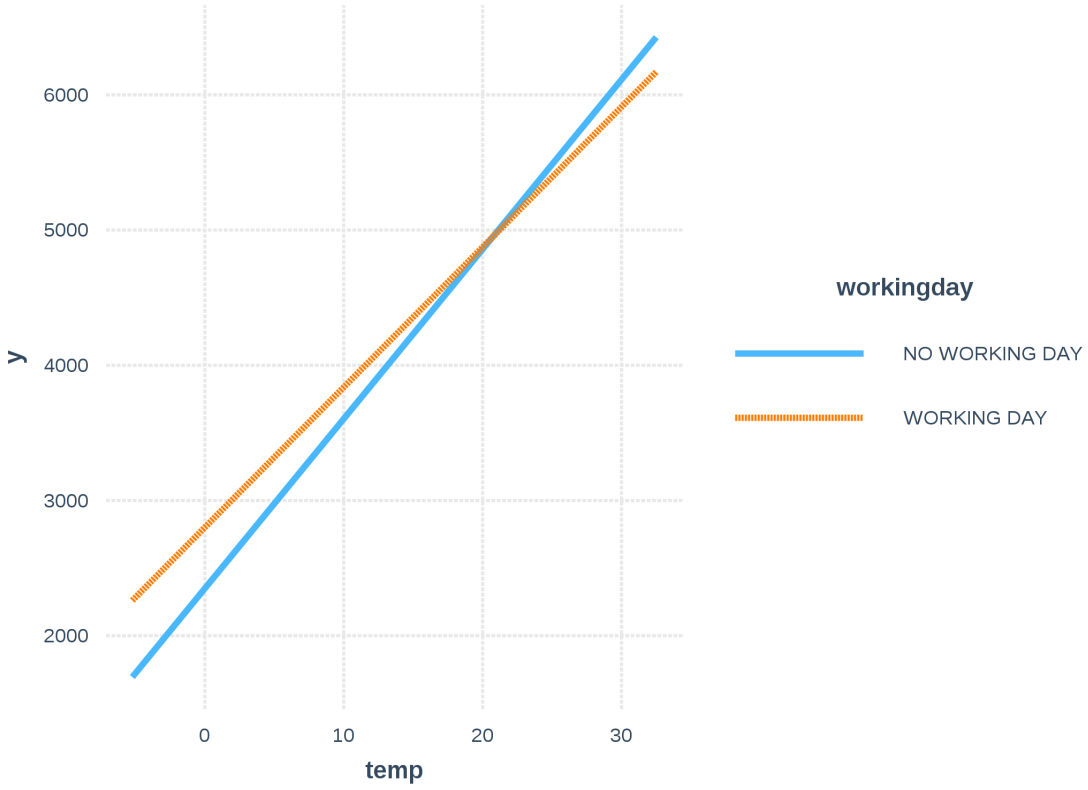


图 4.12. 温度和工作日对线性模型预测的自行车数量的影响 (包括交互作用)。实际上，我们得到了两个温度斜率，每个工作日特征对应一个类别。

4.3.3 非线性效应 - GAM

世界不是线性的。线性模型中的线性意味着，无论一个实例在某个特定特征中具有什么值，将该值增加 1 个单位对预测结果总是有相同的效应。温度在 10 摄氏度时升高 1 度，对租用自行车的数量

产生的效应是否与温度在 40 摄氏度时升高相同？直观地说，人们期望把温度从 10 摄氏度提高到 11 摄氏度会对自行车租赁产生正效应，从 40 摄氏度提高到 41 摄氏度会产生负效应，正如整本书中许多示例中所看到的那样。温度特征对租用自行车的数量有线性的、正的效应，但在某些时候它会变平，甚至在高温下也会产生负效应。线性模型并不关心，它会尽力地找到最佳的线性平面 (通过最小化欧几里得距离)。

你可以使用以下方法对非线性关系建模:

- 特征的简单转换 (例如对数)
- 特征分类
- 广义加性模型 (GAM)

在介绍每种方法的细节之前，让我们先从一个示例开始，说明这三种方法。我们使用并训练了一个仅具有温度特征的线性模型来预测自行车租赁的数量。下图显示了估计的斜率：标准线性模型、对数 (转换) 温度的线性模型、将温度视为分类特征的线性模型以及使用回归样条 (GAM) 的线性模型。

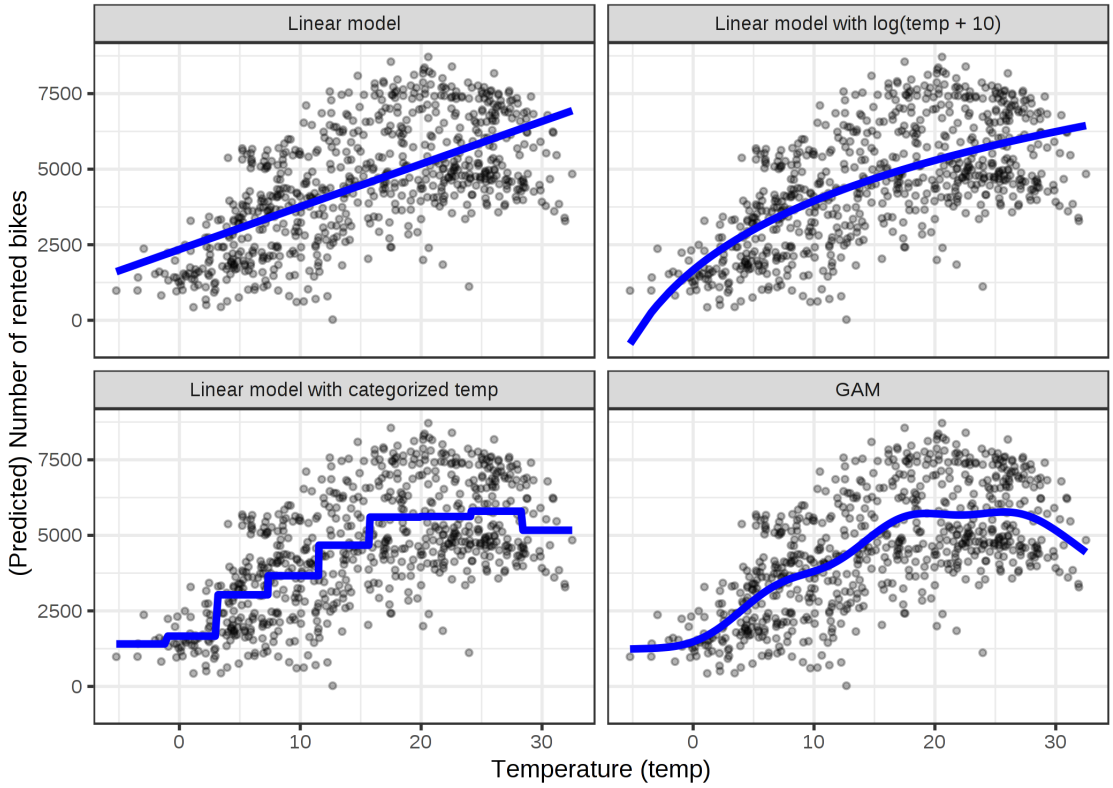


图 4.13. 仅使用温度特征预测租用自行车的数量。线性模型 (左上) 无法很好地拟合数据。一种解决方案 (右上) 是对特征进行转换 (例如使用对数)，对其进行分类 (左下) (这通常是一个错误的决定)，或者使用可以自动拟合温度的平滑曲线的 GAM (右下)。

特征转换

通常将特征的对数用作转换。使用对数表示，温度每升高 10 倍，对自行车数量有相同的线性效应，因此从 1 摄氏度到 10 摄氏度的变化与从 0.1 到 1 的变化具有相同的效果 (听起来错误)。特征变换的其他例子有平方根、平方函数和指数函数。使用特征转换意味着你用特征的函数 (如对数) 替换数据中该特征的列，并照常拟合线性模型。一些统计程序还允许你在线性模型调用中指定转换。因而在转换特征时，可以发挥创造力。特征的解释会根据所选转换而变化。如果使用对数转换，线性模型中的解释将变为：“如果特征的对数增加 1，则预测将增加相应的权重。” 当使用不是恒等函数作为连接函数的 GLM 时，则解释将更复杂，因为你必须将这两种转换合并到解释中 (除非它们相互抵消，如 log 和 exp，那么解释会容易)。

特征分类

实现非线性效应的另一种可能方法是离散化特征，将其转化为分类特征。例如，你可以将温度特征切割为 20 个间隔，级别分别为 [-10, -5)，[-5, 0)…等等。当你使用分类温度而不是连续温度时，线性模型将估计阶跃函数，因为每个级别都有自己的估计。这种方法的问题在于，它需要更多的数据，更容易过拟合，而且不清楚如何有意义地离散化特征 (等距间隔或分位数？有多少个间隔？) 只有在有很强理由的情况下，我们才会使用离散化。例如，使模型与另一项研究相比较。

广义加性模型

为什么不 “简单” 地允许 (广义) 线性模型学习非线性关系呢？这就是 GAM 背后的动机。GAM 放宽了这种限制，即关系必须是一个简单的加权和，而是假定可以由每个特征的任意函数的和建模结果。在数学上，GAM 中的关系如下：

$$g(E_Y(y|x)) = \beta_0 + f_1(x_1) + f_2(x_2) + \ldots + f_p(x_p)$$

该公式与 GLM 公式类似，不同之处在于线性项 $\beta_j x_j$ 被更灵活的函数 $f_j(x_j)$ 取代。GAM 的核心仍然是特征效应的总和，但是你可以选择允许某些特征和输出之间存在非线性关系。线性效应也包含在框架中，因为对于要线性处理的特征，我们可以将它们的形式 $f_j(x_j)$ 限制为仅采用 $\beta_j x_j$。

最大的问题是如何学习非线性函数。答案称为样条函数 (“splines” 或 “spline functions”)。样条曲线是可以组合以近似任意函数的函数，有点像堆叠乐高积木来构建更复杂的东西。定义这些样条函数的方法多种多样。如果你有兴趣了解有关定义样条曲线的所有方法的更多信息，祝你旅途愉快。我们不想在这里讨论细节，只想建立一个直觉。对于理解样条曲线，最大的帮助方法是可视化各个样条函数，并研究是如何修改数据矩阵的。例如，为了用样条曲线对温度进行建模，我们从数据中删除了温度特征，并将其替换为 4 列，每列表示一个样条函数。通常你会有更多的样条函数，我们减少数量只是为了说明目的。这些新样条特征的每个实例的值取决于实例的温度值。连同所有的线性效应，GAM 还将估计这些样条权重。GAM 还为权重引入了一个惩罚项，以使它们接近零。这有效地降低了样条的灵活性，并减少了过拟合。然后通过交叉验证来调整通常用于控制曲线灵活性的平滑度参数。忽略惩罚项，用样条做非线性建模是一个很棒的特征工程。

在示例中，我们仅使用温度的 GAM 预测自行车租赁数量，模型特征矩阵如下所示：

	s(temp).1	s(temp).2	s(temp).3	s(temp).4
1	0.93	-0.14	0.21	-0.83
1	0.83	-0.27	0.27	-0.72
1	1.32	0.71	-0.39	-1.63
1	1.32	0.70	-0.38	-1.61
1	1.29	0.58	-0.26	-1.47
1	1.32	0.68	-0.36	-1.59

每行表示数据中的单个实例 (一天)。每个样条列包含特定温度值下样条函数的值。下图显示了这些样条函数的外观：

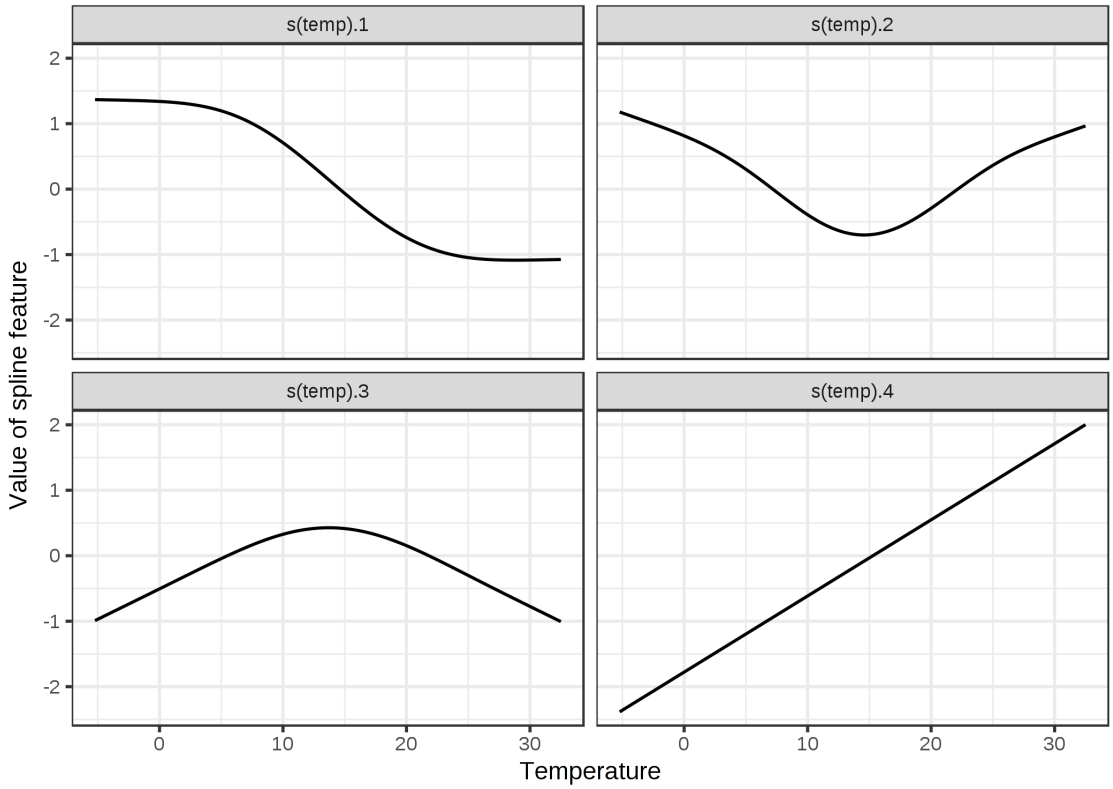


图 4.14. 为了平滑地模拟温度特征效应，我们使用了 4 个样条函数。每个温度值都映射到 (此处) 4 个样条线值。如果实例的温度为 30 摄氏度，则第一个样条曲线特征的值为 -1，第二个样条曲线的值为 0.7，第三个样条曲线的值为 -0.8，第四个样条曲线的值为 1.7。

GAM 为每个温度样条特征分配权重：

	weight
(Intercept)	4504.35
s(temp).1	-989.34
s(temp).2	740.08
s(temp).3	2309.84
s(temp).4	558.27

而实际曲线是由用估计权重加权的样条函数之和得到的，如下所示：

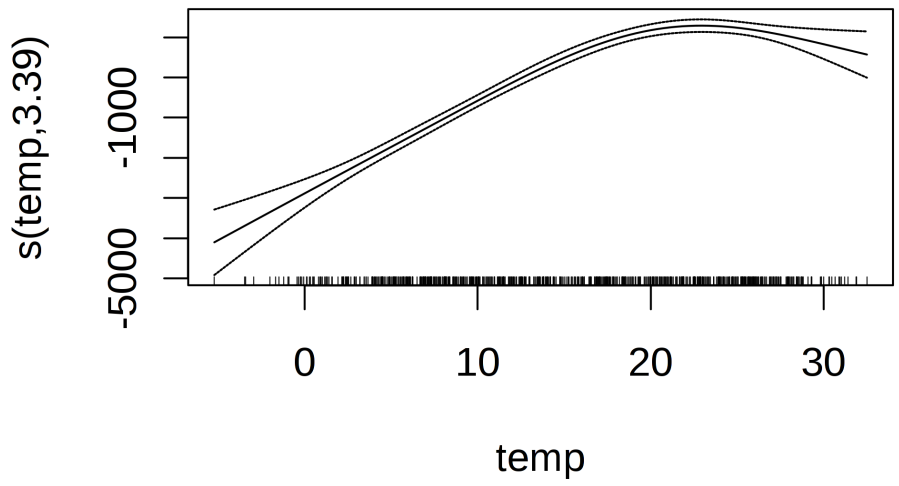


图 4.15. 预测所租自行车的数量 (温度是唯一特征)，温度的 GAM 特征效应。

平滑效果的解释需要对拟合曲线进行目视检查。样条曲线通常以均值预测为中心，因此曲线上的一个点与均值预测是不同的。例如，在 0 摄氏度时，预测的自行车数量比平均预测值低 3,000。

4.3.4 优点

线性模型的所有这些扩展本身就是一个很大的探索空间。无论线性模型遇到的问题是什么，**你都可能会找到可解决该问题的扩展**。

大多数方法已经使用了数十年。例如，GAM 已经有将近 30 年的历史了。许多来自工业界的研究人员和实践者对线性模型非常**有经验**，并且**这些方法在许多社区被认为是建模的现状**。

除了进行预测之外，你还可以使用模型**进行推断**，得出有关数据的结论——只要不违反模型假设即可。你还可以得到权重的置信区间、显著性检验、预测区间等等。

一些统计软件通常有很好的接口来适应 GLM、GAM 和更特殊的线性模型。

许多机器学习模型的不透明性源于 1). 缺乏稀疏性，这意味着使用了许多特征；2). 以非线性方式处理的特征，这意味着需要多个权重来描述特征效应；3). 特征之间交互作用的建模。线性模型具有很高的可解释性，但通常不适合实际，本节所述的扩展提供了一种很好的方式，可以在**平稳地过渡到更灵活的模型的同时**，保留一些可解释性。

4.3.5 缺点

作为优势，我们说过线性模型是一个非常大的探索空间，**扩展简单线性模型的方法太多了**。事实上，很多领域的人都做了这些研究，问题是许多研究者和实践者在他们领域内都用他们自己的名字来命名那些可能做着相同事情的方法，这会非常令人困惑。

线性模型的大多数修改都会使模型的**解释性变差**。不是恒等函数的任何连接函数 (在 GAM 中) 都会使解释复杂化；交互也会使解释复杂化；非线性特征效应要么不太直观 (如对数转换)，要么不能再由单个数字 (如样条函数) 来概括。

GLM、GAM 等**依赖于有关数据生成过程的假设**。如果违反了这些假设，则对权重的解释将不再有效。

在许多情况下，随机森林或梯度提升树等基于树的集成方法的性能比最复杂的线性模型要好。这部分是经验之谈，也部分是来自 kaggle.com 等平台上的获奖模型的观察结果。

4.3.6 软件

本节中的所有示例都是使用 R 语言创建的。对于 GAM，使用了 gam 软件包，但还有许多其他软件包。R 有数量惊人的软件包来扩展线性回归模型。R 语言在线性回归模型的扩展方面，是其他分析语言所无法比拟的。你会在 Python 中找到 GAM 的实现 (例如 pyGAM)，但是这些实现还不那么成熟。

4.3.7 进一步扩展

如前所述，这里列出了线性模型可能遇到的问题，以及问题的解决方案。

1. 问题：数据违反了独立同分布 (IID) 的假设。

示例：例如，对同一患者重复测量。

解决方案：混合模型 (`Mixed Models`) 或广义估计方程 (`Generalized Estimating Equations`)。

2. 问题：模型有异方差性误差 (`Heteroscedastic Errors`)。

示例：例如，在预测房屋价格时，昂贵房屋的模型误差通常较高，这违背了线性模型的同方差性。

解决方案：稳健回归 (`Robust Regression`)。

3. 问题：有异常点会强烈影响模型。

解决方案：稳健回归。

4. 问题：想预测事件发生前的时间。

示例：到达事件的时间数据通常带有经过审查的测量值，这意味着在某些情况下没有足够的时间来观察事件。例如，一家公司想预测其制冰机的故障，但只有两年的数据。一些机器两年后仍然完好无损，但可能会在以后会故障。

解决方案：参数生存模型 (`Parametric Survival Models`)，Cox 回归 (`Cox Regression`)，生存分析 (`Survival Analysis`)。

5. 问题：要预测的结果是一个类别。

解决方案：如果结果有两个类别，则使用逻辑回归模型 (`Logistic Regression Model`)，该模型对类别的概率进行建模。如果你有更多类别，可以考虑多分类回归 (`Multinomial Regression`)。逻辑回归模型和多分类回归都是 GLM。

6. 问题：预测有序的类别。

示例：例如，学校成绩。

解决方案：比例优势模型 (`Proportional Odds Model`)。

7. 问题：结果是一个计数 (就像一个家庭中孩子的数量)。

解决方案：寻找泊松回归 (`Poisson Regression`)。泊松模型也是一个 GLM。

8. 问题：在上一个问题中，0 的计数值非常频繁。

解决方案：寻找零膨胀泊松回归 (`Zero-inflated Poisson Regression`)，栅栏模型 (`Hurdle Model`)。

9. 问题：不确定模型中需要包含哪些特征才能得出正确的因果结论。

示例：例如，想知道药物对血压的影响。这种药物对某些血液有直接影响，而该血液会影响结果。应该将血液纳入回归模型吗?

解决方案：寻找因果推理 (Causal Inference)，中介分析 (Mediation Analysis)。

10. 问题：缺少数据。

解决方案：多重填补 (Multiple Imputation)。

11. 问题：把先前的知识整合到模型中。

解决方案：贝叶斯推断 (Bayesian Inference)。

4.4 决策树

线性回归和逻辑回归在特征与结果之间的关系为非线性或特征交互的情况下会失败。是时候“点亮”决策树 (Decision Tree) 了！基于树的模型根据特征中的某些截断值多次分割 (Split，或称分裂、拆分) 数据。通过分割，可以创建数据集的不同子集，每个实例都属于一个子集。最后的子集称为终端 (Terminal) 或叶节点 (Leaf Nodes)，中间的子集称为内部节点 (Internal Nodes) 或分裂节点 (Split Nodes)。为了预测每个叶节点的结果，使用该节点中训练数据的平均结果。树模型可用于分类和回归。

有多种算法可以生成一棵树，它们在树的可能结构 (例如每个节点的分割数量)、如何分割的标准、何时停止分割以及如何估计叶节点内的简单模型等方面存在差异。CART 算法可能是最流行的树归纳算法。我们将专注于 CART，但是对于大多数其他树类型，其解释是相似的。我们推荐《The Elements of Statistical Learning》一书来更详细地介绍 CART。

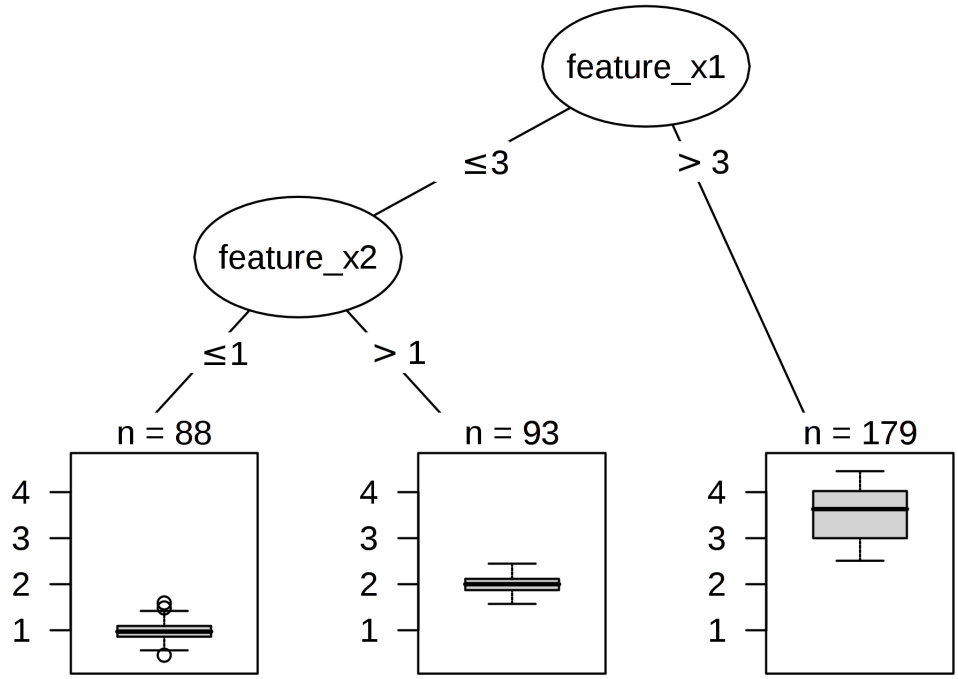


图 4.16. 具有人工数据的决策树。特征 x_1 的值大于 3 的实例最终出现在节点 5 中。所有其他实例都分配给节点 3 或节点 4，具体取决于特征 x_2 的值是否超过 1。

下面的公式描述了结果 y 和特征 x 之间的关系：

$$\hat{y} = \hat{f}(x) = \sum_{m=1}^{M} c_m I\{x \in R_m\}$$

每个实例正好属于一个叶节点 (= 子集 R_m)。$I_{\{x \in R_m\}}$ 是指示函数，即如果 x 在子集 R_m 中则为 1，否则为 0。如果一个实例属于叶节点 R_l，则预测结果为 $\hat{y} = c_l$，其中 c_l 是叶节点 R_l 中所有训练实例的平均值。

但是子集是从哪里来的呢？这很简单：CART 采用一个特征，并确定哪个截断值将回归任务的 y 的方差最小化，或者分类任务的 y 的类别分布的基尼指数最小化。方差告诉我们节点中的 y 值围绕其平均值分散的程度。基尼指数告诉我们节点的“不纯”程度，例如，如果所有的类别都有相同的频率，那么这个节点就是不纯的；如果只有一个类别存在，那么它就是最大的纯度。当节点中的数据点具有非常相似的 y 值时，方差和基尼指数最小化。因此，最佳截断值是使两个结果子集在目标结果方面尽可能不同。对于分类特征，该算法通过尝试不同类别分组来创建子集。在确定每个特征的最佳截断值后，算法选择要进行分割的特征，这是考虑方差或基尼指数下的最佳分割，并将此分割添加到树中。该算法继续此搜索并在新节点中递归分割，直到达到停止条件。可能的条件是：分

割前必须在节点中的最小实例数，或必须在终端节点中的最小实例数。(CART 的理解与代码实现可以参考 这份工作 的第五章机器学习部分。)

4.4.1 解释

解释很简单：从根节点开始，转到下一个节点，而边表明要查看的子集。一旦到达叶节点，该节点将表明预测的结果。所有边通过“AND”连接。

模板：“如果特征 x 比阈值 c [小/大] AND …，那么预测结果就是节点 y 中实例的平均值。”

特征重要性

在决策树中，一个特征的总体重要性可以用以下方法计算：遍历使用该特征的所有分割，并测量它相对于父节点减少了多少方差或基尼指数。所有重要性的总和被缩放为 100，这意味着每个重要性可以解释为总体模型重要性的一部分。

树分解

通过将决策路径分解为每个特征的组成，可以解释决策树的单个预测。我们可以通过树跟踪决策，并通过在每个决策节点上添加的贡献来解释预测。

决策树中的根节点是我们的起点。如果我们使用根节点进行预测，它将预测训练数据结果的均值。在下一个分割中，我们根据路径中的下一个节点减去或者添加一项。为了得到最终的预测，我们必须遵循要解释的数据实例的路径，并不断地添加到公式中。

$$\hat{f}(x) = \bar{y} + \sum_{d=1}^{D} \text{split.contrib}(d, x) = \bar{y} + \sum_{j=1}^{p} \text{feat.contrib}(j, x)$$

单个实例的预测是目标结果的均值加上在根节点和实例结束的终端节点之间发生的 D 分割的所有贡献的总和。不过，我们对分割的贡献不感兴趣，而是对特征的贡献感兴趣。一个特征可能用于多次分割，或者根本不用于分割。我们可以为 p 个特征的每一个添加贡献，并获得每个特征对预测贡献多少的解释。

4.4.2 示例

让我们再看一下自行车租赁数据，我们希望通过决策树来预测某一天的自行车租用数量。学到的树如下所示：

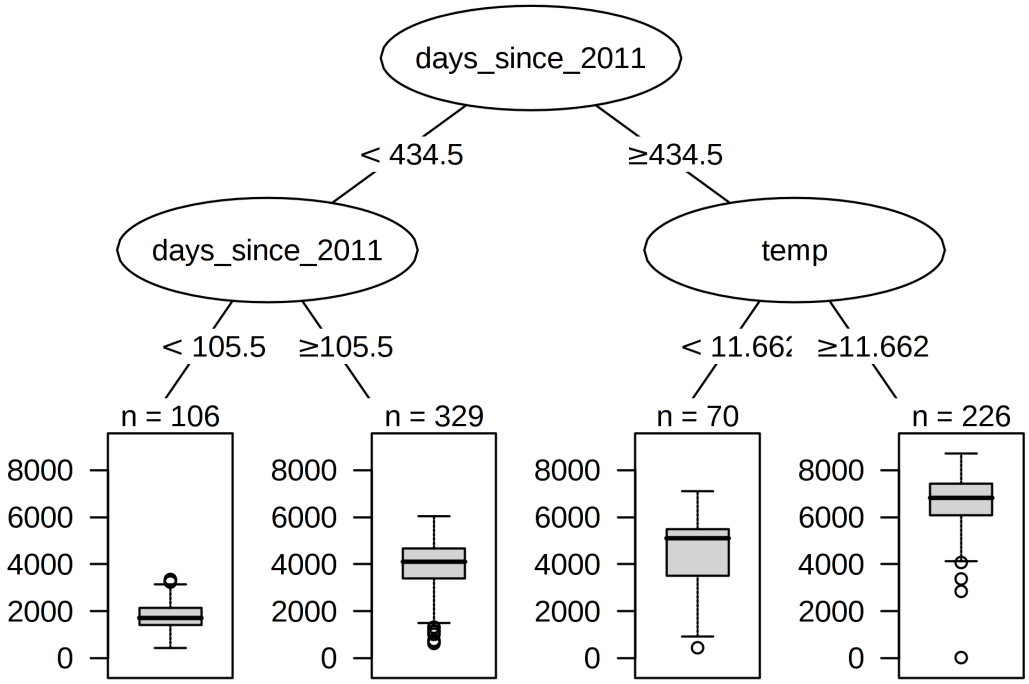


图 4.17. 拟合在自行车租赁数据上的回归树。树的最大允许深度设置为 2。已为分割选择天数特征和温度特征。箱线图显示了终端节点中自行车数量的分布。

第一次分割和第二次分割中的第一个是使用天数特征执行的，该特征统计数据收集开始后的天数，并涵盖了随着时间的推移自行车租赁服务变得越来越流行的趋势。在第 105 天之前的天数中，自行车的预测数量大约是 1800；在第 106 天和第 430 天之间，大约是 3900。对于第 430 天之后的天数，预测值为 4600 (如果温度低于 12 摄氏度) 或 6600 (如果温度高于 12 摄氏度)。

特征的重要性告诉我们特征在多大程度上有助于提高所有节点的纯度。这里使用了方差，因为预测自行车租赁是一个回归任务。

可视化树表明温度特征和天数特征都被用于分割，但没有量化哪个特征更重要。特征重要性表明天数特征远比温度特征重要。

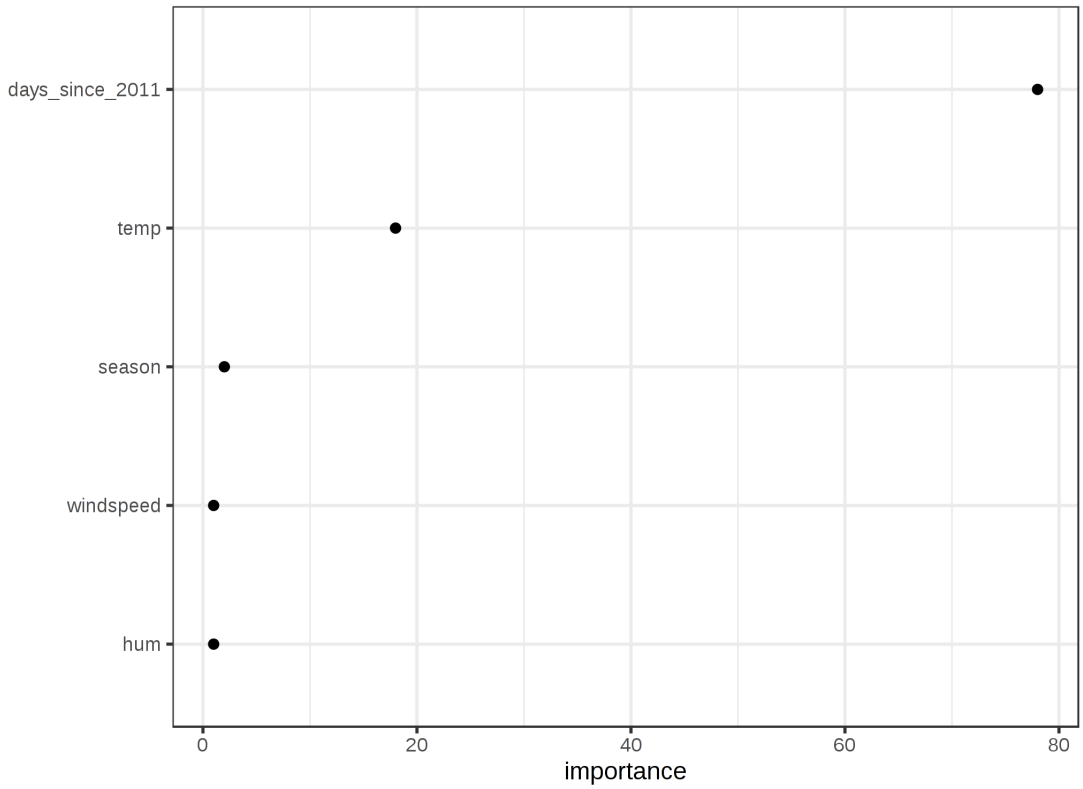


图 4.18. 通过平均提高的节点纯度可以衡量特征的重要性。

4.4.3 优点

树结构非常适合捕获数据中特征之间的**交互**。

这些数据最终分成**不同的组**，通常比线性回归中的多维超平面上的点更容易理解。可以说，解释相当简单。

树结构也有一个**自然的可视化**，包括它的节点和边。

树可以按照 “2.6 人性化的解释” 一节中的定义**提供良好的解释**。树结构自动将单个实例的预测值视为反事实：“如果某个特征大于/小于分割点，则预测值将是 y_1 而不是 y_2”。树的解释具有对比性，因为你总是可以将实例的预测与只是树的其他叶节点的相关的 “假设” 场景 (由树定义) 进行比较。如果这棵树很短，像 1 到 3 个分割的深度，那么所得到的解释是有选择性的。深度为 3 的树最多需要 3 个特征和分割点来为单个实例的预测创建解释。预测的真实性取决于树的预测性能。对于短树的解释非常简单和普遍性，因为对于每一个分割，实例都属于一个或另一个叶子，并且二进制决策很容易理解。

不需要转换特征。在线性模型中，有时需要取特征的对数。决策树同样适用于特征的任何单调变换。

4.4.4 缺点

树不能处理线性关系。输入特征和结果之间的任何线性关系都必须通过分割来近似，从而创建一个阶跃函数。这不是有效的。

这会**缺乏平滑度**。输入特征的微小变化会对预测结果产生很大影响，这通常是不可取的。想象一棵树预测房屋的价格，而这棵树使用房屋的面积大小作为分割特征之一。分割发生在 100.5 平方米。可以想象一下，使用你的决策树模型进行房屋价格估计的用户：他们测量他们的房子，得出房子有 99 平方米，然后放入模型中计算价格，得到 20 万欧元的预测。用户注意到他们忘记了测量一个 2 平方米的小储藏室。储藏室有一道倾斜的墙，因此他们不确定是可以计数全部面积还是仅占一半。所以他们决定尝试 100.0 和 101.0 平方米。结果：模型输出了 20 万欧元和 20.5 万欧元，这是相当不合理的，因为这表明从 99 平方米到 100 平方米没有变化。

树也相当**不稳定**。训练数据集中的一些更改可以创建完全不同的树。这是因为每个分割都依赖于父分割。如果选择其他特征作为第一个分割特征，则整个树结构将发生更改。如果结构如此容易地改变，则不会对模型产生信心。

决策树很容易理解——只要它们很短。**终端节点的数量随着深度的增加而迅速增加**。终端节点越多，树越深，就越难理解树的决策规则。深度为 1 表示 2 个终端节点，深度 2 表示最多 4 个节点，深度 3 表示最多 8 个节点。一棵树中终端节点的最大数量是深度的 2 的指数倍。

4.4.5 软件

对于本节中的示例，我使用了 R `rpart` 包实现 CART。CART 以多种编程语言 (包括 Python) 实现。可以说，CART 是一个非常古老且有些过时的算法，并且有一些有趣的新算法可以拟合树。你可以在 机器学习和统计学习 CRAN 任务视图 中的关键字 “递归分区” 下找到一些决策树 R 包的概述。

4.5 决策规则

决策规则 (`Decision Rules`) 是一个简单的 `IF-THEN` 语句，由条件 (也称为先行条件) 和预测组成。例如：“`IF` 今天下雨 `AND` 现在是四月 (条件)，`THEN` 明天下雨 (预测)”。可以使用单个决策规则或多个规则的组合进行预测。

决策规则遵循一个一般结构：`IF` 条件满足，THEN 做一个特定的预测。决策规则可能是最容易解释的预测模型。它们的 `IF-THEN` 结构在语义上类似于自然语言和我们的思维方式，前提是条件是由可理解的特征构建的，条件的长度很短 (用 `AND` 连接的 `feature=value` 组合较少)，并且没有太多规则。在编程中，编写 `IF-THEN` 规则是很自然的。机器学习中的新功能是通过算法学习决策规则。

想象一下，使用一种算法来学习预测房屋价格的决策规则 (价格 low，medium 或 high)。这个模型学习到的一个决策规则可能是：如果一个房屋超过 100 平方米，并且有一个花园，那么它的价格就很高。更正式一点：IF size>100 AND garden=1 THEN value=high。

让我们分解一下决策规则：

- size>100 是 IF 部分的第一个条件。
- garden=1 是 IF 部分的第二个条件。
- 这两个条件通过 “AND” 连接以创建新条件。两者都必须为真，才能应用该规则。
- 预测结果 (THEN 部分) 为 value=high。

决策规则在条件中至少使用一个 feature=value 语句，对于 “AND” 可以添加的数量没有上限。一个例外是没有显式 IF 部分的默认规则，当没有其他规则应用时该规则适用，稍后将对此进行详细说明。

决策规则的有用性通常概括为两个数字：支持度 (Support) 和准确性 (Accuracy)。

- **规则的支持度或覆盖率 (Coverage)**：规则条件适用的实例所占的百分比称为支持度。例如，IF size=big AND location=good THEN value=high 用于预测房屋价格。假设 1000 个房屋中有 100 个大的而且位置好的，那么规则的支持度就是 10%。预测 (THEN 部分) 对于支持度的计算并不重要。
- **规则的准确性或置信度 (Confidence)**：规则的准确性是衡量规则在规则条件适用的实例预测正确类别时的准确性的指标。例如：假设规则 IF size=big AND location=good THEN value=high 适用的 100 个房屋，其中，85 个房屋中的 value=high，14 个房屋中的 value=medium，1 个房屋中的 value=low，则规则的准确性为 85%。

通常在准确性和支持度之间有一个权衡：通过在条件中添加更多的特征，我们可以获得更高的准确性，但会失去支持度。

为了创建一个好的分类器来预测房屋的价格，我们可能不仅需要学习一条规则，而且需要学习 10 或 20 条规则。然后事情会变得更加复杂，并且可能会遇到以下问题之一：

- 规则可能会重叠：如果我想预测房屋的价格并且应用两个或多个规则，并且它们给我带来矛盾的预测，该怎么办?
- 没有适用的规则：如果我想预测房屋的价格，却没有适用的规则怎么办?

组合多个规则有两种主要策略：决策列表 (Decision List) (有序) 和决策集 (Decision Set) (无序)。这两种策略都暗示了对重叠规则问题的不同解决方案。

- **决策列表**向决策规则引入有序。如果第一条规则对实例的条件为真，则使用第一条规则的预测。如果没有，我们将转到下一个规则，检查它是否适用，依此类推。决策列表通过只返回适用列表中第一个规则的预测来解决重叠规则的问题。
- **决策集**类似于规则的民主，只是有些规则可能具有更高的投票权。在一套规则中，这些规则要么相互排斥，要么有解决冲突的策略，例如多数投票，这些策略可以通过个别规则的准确性或其他质量措施进行加权。当一些规则适用时，解释性可能会受到影响。

决策列表和决策集都可能面临没有规则适用于实例的问题。这可以通过引入默认规则来解决。默认规则是在没有其他规则适用时应用的规则。默认规则的预测通常是数据点中最常见的一类，而其他规则没有涵盖这些数据点。如果一组规则或规则列表覆盖了整个特征空间，我们称之为详尽的。通过添加默认规则，集合或列表将自动变得详尽。

有很多方法可以从数据中学习规则，本书远没有涵盖所有的这些方法。本节展示了其中三个。选择这些算法是为了涵盖通用的学习规则的一般思想，因此这三种算法代表了截然不同的方法。

1. OneR 从单个特征中学习规则。OneR 的特点在于其简单性、可解释性、并且可以用作基准。
2. **顺序覆盖** (`Sequential covering`) 是一种通用过程，可以迭代地学习规则并删除新规则覆盖的数据点。许多规则学习算法都使用此过程。
3. **贝叶斯规则列表** (`Bayesian Rule List`) 使用贝叶斯统计将预先挖掘的频繁模式组合到决策列表中。使用预先挖掘模式是许多规则学习算法所使用的常见方法。

让我们从最简单的方法开始：使用单个最佳特征来学习规则。

4.5.1 从单个特征学习规则 (`OneR`)

Holte (1993)[16] 提出的 OneR 算法是最简单的规则归纳算法之一。从所有特征中，OneR 选择一个包含有关感兴趣结果的最多信息的特征，并从该特征创建决策规则。

尽管命名为 OneR 代表 “一个规则”，但是算法会生成多个规则：它实际上是所选最佳特征的每个 (唯一的，不重复的) 特征值的一个规则。一个更好的名字应该是 OneFeatureRules。

算法简单快捷：

1. 通过选择适当的间隔来离散化连续特征。
2. 对于每个特征：
    - 在特征值和 (分类) 结果之间创建交叉表。
    - 对于特征的每个值，创建一个规则，预测具有此特定特征值 (可以从交叉表中读取) 的实例的最常见类别。
    - 计算特征规则的总误差。

3. 选择总误差最小的特征。

OneR 始终覆盖数据集的所有实例，因为它使用所选特征的所有级别 (可能值)。缺失值可以作为附加特征值处理，也可以预先输入。

OneR 模型是一个只有一个分割的决策树。分割不一定像 CART 中那样是二进制的 (二分叉)，而是取决于 (唯一的、不重复的) 特征值的数量。

让我们来看一个例子，OneR 是如何选择最佳特征的。下表显示了有关房屋的人工数据集，其中包含有关房屋价格、位置、面积大小以及是否允许养宠物的信息。我们有兴趣学习一个简单的模型来预测房屋的价格。

location	size	pets	value
good	small	yes	high
good	big	no	high
good	big	no	high
bad	medium	no	medium
good	medium	only cats	medium
good	small	only cats	medium
bad	medium	yes	medium
bad	small	yes	low
bad	medium	yes	low
bad	small	no	low

OneR 在每个特征和结果之间创建交叉表：

	value=low	value=medium	value=high
location=bad	3	2	0
location=good	0	2	3

	value=low	value=medium	value=high
size=big	0	0	2
size=medium	1	3	0
size=small	2	1	1

	value=low	value=medium	value=high
pets=no	1	1	2
pets=only cats	0	2	0
pets=yes	2	1	1

对于每一个特征，我们一行一行地遍历表：每个特征值都是规则的 `IF` 部分；对于具有这个特征值的实例，最常见的类别是预测，即规则的 `THEN` 部分。例如，级别 (可能值) 为 `small`，`medium` 和 `big` 的 (面积) 大小特征在三个规则中。对于每个特征，我们计算生成的规则的总错误率，即误差总和。位置特征具有可能的值 `bad` 和 `good`。不良地段房屋最常见的价值是 `low`，当我们用 `low` 作为预测时，我们会有两个错误，因为有两个房屋的价格是 `medium` 的。好地段的房屋预测值 `high`，我们又有了两个错误，因为有两个房屋的价格 `medium`。我们使用位置特征的误差是 4/10，对于面积大小特征是 3/10，对于宠物特征是 4/10。面积大小特征生成误差最小的规则，并将用于最终的 OneR 模型：

```
IF size=small THEN value=low

IF size=medium THEN value=medium

IF size=big THEN value=high
```

OneR 倾向于具有许多可能级别的特征，因为这些特征可以更容易地覆盖目标。设想一个只包含噪声不包含信号的数据集，这意味着所有特征都具有随机值，并且对目标没有预测价值。有些特征比其他特征有更多的级别。具有更多级别的特征更容易过拟合。每个数据中的实例具有不同级别的特征可以完美地预测整个训练数据集。一种解决方案是将数据拆分为训练集和验证集，学习训练数据的规则，并评估在验证集上选择特征时的总误差。

连结 (`Ties`) 是另一个问题，即当两个特征导致相同的总误差时。OneR 通过使用具有误差最小的第一个特征或具有卡方检验的 p 值最小的特征来解决连结。

示例

让我们尝试使用真实数据进行 OneR。我们使用宫颈癌分类任务来测试 OneR 算法。将输入的所有连续特征离散化为 5 个分位数，创建以下规则：

Age	prediction
(12.9,27.2]	Healthy
(27.2,41.4]	Healthy
(41.4,55.6]	Healthy
(55.6,69.8]	Healthy

Age	prediction
(69.8,84.1]	Healthy

OneR 选择年龄特征作为最佳预测特征。由于癌症是罕见的，因此对于每个规则选择多数的类别作为预测，因此预测的标签始终是 Healthy，这是无济于事的。在这种不平衡的情况下，使用标签预测是没有意义的。年龄 (`Age`) 间隔与癌症/健康状况 (`Cancer/Healthy`) 以及患癌症的女性百分比之间的交叉表更具参考价值：

	# Healthy	# Cancer	P(Cancer)
Age=(12.9,27.2]	477	26	0.05
Age=(27.2,41.4]	290	25	0.08
Age=(41.4,55.6]	31	4	0.11
Age=(55.6,69.8]	1	0	0.00
Age=(69.8,84.1]	4	0	0.00

但是在你开始解释任何东西之前：由于对每个特征和每个值的预测都是 Healthy，所以所有特征的总误差率都是相同的。默认情况下，通过使用误差率最低的特征中的第一个特征 (这里，所有特征都有 55/858) 来解决总误差中的连结，这恰好是年龄特征。

OneR 不支持回归任务。但是我们可以通过将连续结果切成间隔来将回归任务转变为分类任务。我们使用此技巧将 OneR 的自行车数量减少到四个四分位数 (0-25%，25-50%，50-75% 和 75-100%)，从而预测使用 OneR 的自行车租赁数量。下表显示了拟合 OneR 模型后的选定特征：

mnth	prediction
JAN	[22,3152]
FEB	[22,3152]
MAR	[22,3152]
APR	(3152,4548]
MAY	(5956,8714]
JUN	(4548,5956]
JUL	(5956,8714]
AUG	(5956,8714]
SEP	(5956,8714]
OKT	(5956,8714]
NOV	(3152,4548]

mnth	prediction
DEZ	[22,3152]

所选特征是月份 (mnth)。特征月份有 12 个特征级别，比大多数其他特征都要多，因此存在过拟合的风险。在乐观的方面：月份特征可以处理季节性趋势 (例如，冬季租车减少)，并且预测似乎是明智的。

现在，我们从简单的 OneR 算法过渡到使用具有更复杂条件的规则 (由几个特征组成) 的更复杂过程：顺序覆盖。

4.5.2 顺序覆盖

顺序覆盖 (`Sequential Covering`) 是一个通用过程，它重复学习单个规则以创建一个决策列表 (或集合)，该决策列表 (或集合) 按规则覆盖整个数据集。许多规则学习算法是顺序覆盖算法的变体。本节介绍了主要的配方，以及 RIPPER (一个变体的顺序覆盖算法)。

这个想法很简单：首先，找到一个适用于某些数据点的好规则；删除规则所涵盖的所有数据点 (当条件适用时，不管这些点是否正确分类，都会覆盖数据点)；重复规则学习，在剩余的点删除覆盖的点，直到没有剩余的点或满足另一个停止条件为止。最后结果是一个决策列表。这种重复规则学习和删除覆盖数据点的方法称为 “变治法” (`separate-and-conquer`)。

假设我们已经有一个算法可以创建一个覆盖部分数据的规则。两个类 (一个正类，一个负类) 的顺序覆盖算法的工作原理如下：

- 从一个空的规则列表 ($rlist$) 开始。
- 学习一个规则 r。
- 直到规则列表低于某个质量阈值 (或正面示例未被覆盖)：
  - 将规则 r 添加到 $rlist$。
  - 删除规则 r 涵盖的所有数据点。
  - 在剩余数据中学习其他的规则。
- 返回决策列表。

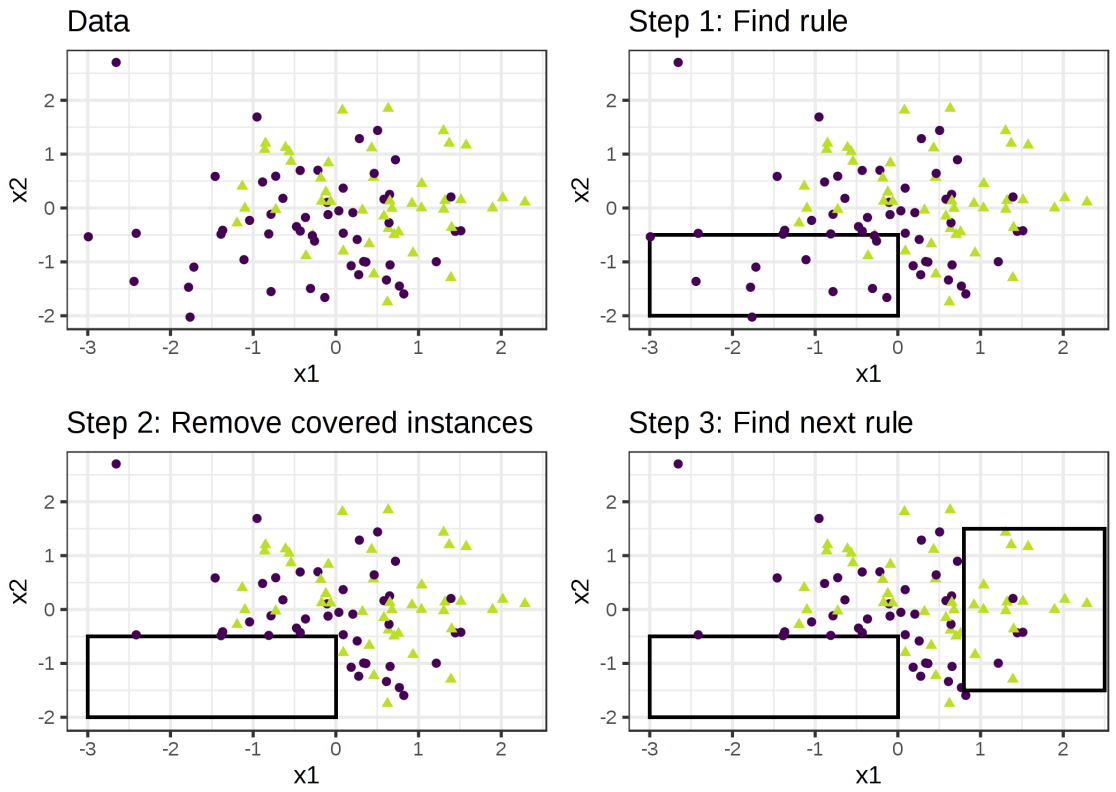


图 4.19. 覆盖算法的工作原理是依次用单个规则覆盖特征空间并删除那些规则已覆盖的数据点。出于可视化目的，特征 x_1 和 x_2 是连续的，但是大多数规则学习算法都需要分类特征。

例如：我们有一个任务和数据集，用于根据面积大小，位置以及是否允许携带宠物来预测房屋的价格。我们学习了第一条规则，结果是：`IF size=big AND location=good THEN value=high`。然后我们从数据集中删除所有位于良好位置的大房屋。利用剩下的数据，我们学习下一个规则。可能是：`IF location=good THEN value=medium`。请注意，这条规则是在没有好位置的大房屋的数据上学习的，只留下有好位置的中小型房屋。

对于多类别的设置，必须修改方法。首先，通过频繁度 (`Prevalence`) 来对类别进行排序。顺序覆盖算法从最不频繁类开始，为其学习规则，删除所有覆盖的实例；然后转到第二不频繁的类，依此类推。当前类别始终被视为正类别，而具有较高频繁度的类别都被认为负类别。最后一个类是默认规则。这在分类中也被称为“一对多”策略。

我们如何学习一条规则？OneR 算法在这里是无用的，因为它总是覆盖整个特征空间。但是还有很多其他可能。一种可能方法是通过束搜索 (`Beam Search`) 从决策树中学习单个规则：

- 学习决策树 (使用 CART 或其他树学习算法)。
- 从根节点开始，递归地选择最纯的节点 (例如，具有较低的错误分类率)。

- 终端节点的多数的类别用作规则的预测；通向该节点的路径用作规则条件。

下图说明了树中的束搜索：

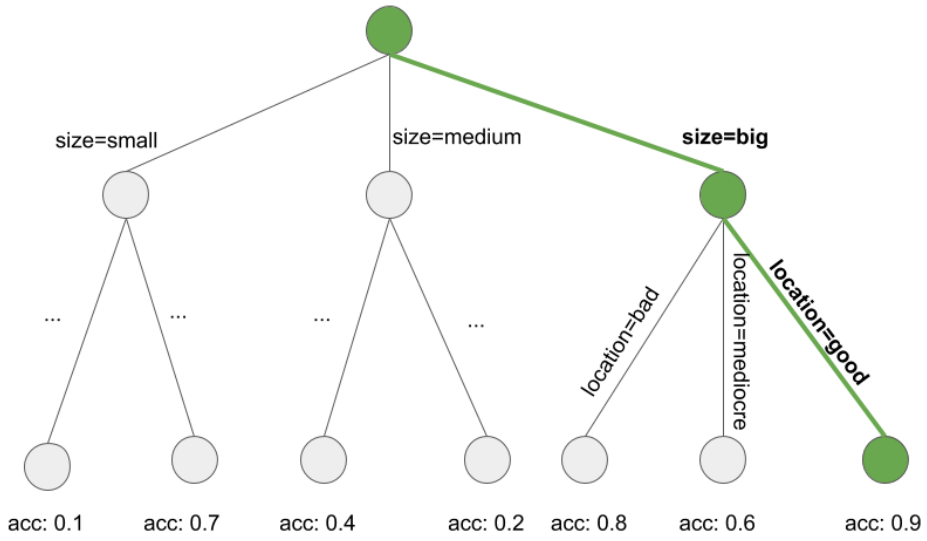


图 4.20. 通过搜索决策树中的路径来学习规则。生成决策树以预测感兴趣的目标。我们从根节点开始，贪婪地和迭代地遵循局部生成最纯子集 (例如，最高准确性) 的路径，并将所有分割值添加到规则条件中。结果是：`IF size=big AND location=good THEN value=high`。

学习单个规则是一个搜索问题，其中搜索空间是所有可能规则的空间。搜索的目标是根据某些条件找到最佳规则。有许多不同的搜索策略：爬山法 (`Hill-climbing`)，束搜索，穷尽搜索 (`Exhaustive Search`)，最佳优先搜索 (`Best-first Search`)，有序搜索 (`Ordered Search`)，随机搜索 (`Stochastic Search`)，自上而下搜索 (`Top-down Search`)，自下而上搜索 (`Bottom-up Search`) 等等。

Cohen (1995) [17] 提出的 RIPPER (`Repeated Incremental Pruning to Produce Error Reduction`) 是顺序覆盖算法的一个变体。RIPPER 有点复杂，它使用后处理阶段 (规则修剪) 来优化决策列表 (或集合)。RIPPER 可以在有序或无序模式下运行，并生成决策列表或决策集。

示例

我们将使用 RIPPER 作为例子。

RIPPER 算法在宫颈癌的分类任务中未找到任何规则。

当我们在回归任务上使用 RIPPER 来预测自行车数时，会发现一些规则。由于 RIPPER 仅适用于分类，因此必须将自行车计数转换为分类结果。我们通过将自行车数减少到四分位数来实现的。例如 (4548，5956) 是涵盖 4548 和 5956 之间的预测自行车计数的间隔。下表显示了学习规则的决策

列表。

rules
(days_since_2011 >= 438) and (temp >= 17) and (temp <= 27) and (hum <= 67) => cnt=(5956,8714]
(days_since_2011 >= 443) and (temp >= 12) and (weathersit = GOOD) and (hum >= 59) => cnt=(5956,8714]
(days_since_2011 >= 441) and (windspeed <= 10) and (temp >= 13) => cnt=(5956,8714]
(temp >= 12) and (hum <= 68) and (days_since_2011 >= 551) => cnt=(5956,8714]
(days_since_2011 >= 100) and (days_since_2011 <= 434) and (hum <= 72) and (workingday = WORKING DAY) => cnt=(3152,4548]
(days_since_2011 >= 106) and (days_since_2011 <= 323) => cnt=(3152,4548]
=> cnt=[22,3152]

解释很简单：如果条件适用，我们将在右侧预测自行车数量的间隔。最后一个规则是默认规则，当其他规则均不应用于实例时，该规则适用。要预测新实例，从列表顶部开始，检查规则是否适用。当一个条件匹配时，规则的右侧就是该实例的预测。默认规则确保始终存在预测。

4.5.3 贝叶斯规则列表

在本节中，我们将向你展示学习决策列表的另一种方法，它遵循以下粗略的方法：

1. 从数据中预先挖掘 (`pre-mined`) 频繁使用的模式，这些模式可用作决策规则的条件。
2. 从预先挖掘的规则中选择决策列表。

使用此配方的特定方法称为贝叶斯规则列表 (Letham 等人，2015)[18] (`Bayesian Rule Lists`) 或简称为 BRL。BRL 使用贝叶斯统计从频繁模式中学习决策列表，这些频繁模式已使用 FP-tree (Borgelt，2005)[19] 算法进行了预先挖掘。

让我们从 BRL 的第一步慢慢开始。

频繁模式的预先挖掘

频繁模式是特征值的频繁同现。作为 BRL 算法的预处理步骤，我们使用特征 (在此步骤中不需要目标结果) 并从中提取频繁出现的模式。模式可以是单个特征值 `size=medium` 或者特征值的组合 `size=medium AND location=bad`。

模式的频率 (frequency) 是通过其在数据集中的支持度来测量：

$$Support(x_j = A) = \frac{1}{n}\sum_{i=1}^{n} I(x_j^{(i)} = A)$$

其中 A 是特征值，n 是数据集中的数据点数量，I 是指示函数 (如果实例 i 的特征 x_j 具有级别 A，则返回 1，否则为 0)。在房屋价格数据集中，如果 20% 的房屋没有阳台，而 80% 的房屋有一个或多个阳台，那么对 `balcony=0` 的支持度为 20%。支持度还可以测量特征值组合，例如 `balcony=0 AND pets=allowed`。

有许多算法可以找到这种频繁的模式，例如 Apriori 或 FP-Growth。使用哪种方法无关紧要，只有找到模式的速度不同，但生成的模式总是相同的。

我们将给出 Apriori 如何找到频繁模式的大致想法。实际上，Apriori 算法由两部分组成，第一部分查找频繁的模式，第二部分从中构建关联规则。对于 BRL 算法，我们只对 Apriori 第一部分中生成的频繁模式感兴趣。

在第一步中，Apriori 算法从支持度大于用户定义的最小支持度的所有特征值开始。如果用户说最小支持应该是 10%，而只有 5% 的房屋拥有 `size=big`，我们将删除该特征值并仅保留 `size=medium` 和 `size=small` 作为模式。这并不意味着将房屋从数据中删除，它只是意味着 `size=big` 不会作为频繁模式返回。基于具有单个特征值的频繁模式，Apriori 算法迭代地寻找越来越高阶的特征值组合。模式是通过将 `feature=value` 语句与逻辑 `AND` (例如，`size=medium AND location=bad`) 相结合来构造的。删除支持度低于最小支持度的生成模式。最后我们有了所有频繁模式。频繁模式的任何子集都会再次频繁出现，这称为 Apriori 性质。直观地说，它是有意义的：通过从模式中删除条件，缩小后的模式只能覆盖更多或相同数量的数据点，而不能覆盖更少的数据点。例如，如果 20% 的房屋 `size=medium and location=good`，那么 `size=medium` 的房屋的支持度为 20% 或更大。Apriori 性质用于减少要检查的模式的数量。只有在频繁模式的情况下，我们才需要检查高阶模式。

现在我们已经完成了贝叶斯规则列表算法的预挖掘条件。但在我们继续进行 BRL 的第二步之前，这里想说一下另一种基于预先挖掘模式的规则学习方法。其他方法建议将感兴趣的结果包含到频繁的模式挖掘过程中，并构建 `IF-THEN` 规则的 Apriori 算法的第二部分。由于该算法是无监督的，因此 `THEN` 部分还包含我们不感兴趣的特征值。但是我们可以通过仅对 `THEN` 部分感兴趣的结果的规则进行过滤。这些规则已经形成了一个决策集，但也可以对规则进行排列、修剪、删除或重新组合。

然而，在 BRL 方法中，我们使用频繁模式，学习 `THEN` 部分以及如何使用贝叶斯统计将模式排列到决策列表中。

学习贝叶斯规则列表

BRL 算法的目标是通过选择预先挖掘的条件来学习一个准确的决策列表，同时对规则较少、条件较短的列表进行优先级排序。BRL 通过在条件长度 (最好是较短的规则) 和规则数量 (最好是较短

的列表) 的先验分布中定义决策列表的分布来实现此目标。

在给定简短假设的前提下，列表的后验概率分布使得描述决策列表的可能性，以及列表对数据拟合程度成为可能。我们的目标是找到使后验概率最大化的列表。由于无法直接从列表的分布中找到确切的最佳列表，BRL 建议采用以下方法：

1. 生成初始决策列表，该列表是从先验分布中随机抽取的。

2. 通过添加，切换或删除规则来迭代地修改列表，确保结果列表遵循列表的后验分布。

3. 根据后验分布，从概率最高的采样列表中选择决策列表。

让我们更仔细地研究一下算法：该算法从使用 FP-Growth 预先挖掘的特征值模式开始。BRL 对目标的分布和定义目标分布的参数的分布做了许多假设。(这是贝叶斯统计。) 如果你不熟悉贝叶斯统计，不要太在意下面的解释。重要的是要知道贝叶斯方法可以结合现有知识或要求 (所谓的先验分布)，同时也适合于数据。在决策列表的情况下，贝叶斯方法是有意义的，因为先验假设会推动决策列表较短且规则简短。

目标是从后验分布中抽取决策列表 d：

$$\underbrace{p(d|x,y,A,\alpha,\lambda,\eta)}_{posteriori} \propto \underbrace{p(y|x,d,\alpha)}_{likelihood} \cdot \underbrace{p(d|A,\lambda,\eta)}_{priori}$$

其中，d 是决策列表，x 是特征，y 是目标，A 是预先挖掘的条件集，λ 决策列表的先验期望长度，η 是一个规则中先验期望的条件数，α 是狄利克雷分布参数 (如果是考虑正类别和负类别的先验伪计数，最好固定为 (1,1))。

$$p(d|x,y,A,\alpha,\lambda,\eta)$$

是根据观测数据和先验假设量化决策列表的概率的。这与给定决策列表和数据下的结果 y 的似然，和给定先验假设和预先挖掘的条件下列表的概率的乘积成正比。

$$p(y|x,d,\alpha)$$

是给定决策列表和数据下观察到的 y 的似然。BRL 假设 y 由狄利克雷多项分布生成。决策列表 d 解释数据越多，似然越高。

$$p(d|A,\lambda,\eta)$$

是决策列表的先验分布。它将针对列表中规则数的截断泊松分布 (参数 λ) 与针对规则的条件中特征值数 (即条件长度) 的截断泊松分布 (参数 η) 相乘。

如果一个决策列表能够很好地解释结果 y 并且遵循先验假设，那么它很可能具有很高的后验概率。

贝叶斯统计中的估计总是有点棘手，因为我们通常不能直接计算出正确的答案，而是使用马尔可夫链蒙特卡洛方法绘制候选者，对其进行评估并更新我们的后验估计。对于决策列表，这更为棘手，

因为我们必须从决策列表的分布中绘制。BRL 的作者建议首先绘制一个初始决策列表，然后对其进行迭代修改，从列表的后验分布 (决策列表的马尔可夫链) 中生成决策列表的样本。结果可能依赖于初始决策列表，因此建议重复此过程以确保列表的多样性。软件实现中的默认值是 10 次。以下配方告诉我们如何绘制初始决策列表：

- FP-Growth 预先挖掘模式。
- 从截断的泊松分布中采样列表长度参数 m。
- 对于默认规则：采样目标值的狄利克雷多项分布参数 θ_0 (即在没有其他的可以应用时应用的规则)。
- 对于决策列表规则 $j = 1, ..., m$，执行：
  - 对规则 j 采样规则长度参数 l_j (条件长度)。
  - 从预先挖掘的条件中采样长度为 l_j 的条件。
  - 对 `THEN` 部分采样狄利克雷多项分布参数 (即给定规则的目标结果的分布)。
- 对于数据集中的每个观测：
  - 从决策列表中查找首先适用的规则 (自上而下)。
  - 从适用规则建议的概率分布 (狄利克雷分布) 中得出预测结果。

下一步是从这个初始样本开始生成许多新列表，从而从决策列表的后验分布中获取许多样本。

通过从初始列表开始，然后随机地将规则移动到列表中的其他位置，或者将从预先挖掘的条件中的规则添加到当前决策列表中，或者从决策列表中删除规则，来对新决策列表进行采样。随机选择切换，添加或删除规则。在每个步骤中，算法都会估计决策列表的后验概率 (准确性和简短性的混合)。Metropolis Hastings 算法确保我们采样具有较高后验概率的决策列表。此过程为我们提供了决策列表分布中的许多样本。BRL 算法选择具有最高后验概率的样本的决策列表。

示例

理论就是这样，现在让我们看看 BRL 方法的实际应用。示例使用了 Yang 等人 (2017)[20] 提出的一种更快的 BRL 变体，称为 “可伸缩贝叶斯规则列表” (SBRL)。我们使用 SBRL 算法来预测宫颈癌的风险。为了使 SBRL 算法正常工作，我们首先必须离散化所有输入特征。为此，我们基于分位数的值的频率对连续特征进行了分类。我们得到以下规则：

rules
If {STDs=1} (rule[259]) then positive probability = 0.16049383

rules
else if {Hormonal.Contraceptives..years.=[0,10)} (rule[82]) then positive probability = 0.04685408
else (default rule) then positive probability = 0.27777778

请注意，我们得到了明智的规则，因为 `THEN` 部分的预测不是类别结果，而是癌症的预测概率。

这些条件是从使用 FP-Growth 算法预先挖掘的模式中选择的。下表显示了 SBRL 算法可以从中选择以构建决策列表的条件池。在我作为用户允许的条件下，特征值的最大数目是 2。以下是十个模式的示例：

pre-mined conditions
Num.of.pregnancies=[3.67,7.33)
IUD=0,STDs=1
Number.of.sexual.partners=[1,10),STDs..Time.since.last.diagnosis=[1,8)
First.sexual.intercourse=[10,17.3),STDs=0
Smokes=1,IUD..years.=[0,6.33)
Hormonal.Contraceptives..years.=[10,20),STDs..Number.of.diagnosis=[0,1)
Age=[13,36.7)
Hormonal.Contraceptives=1,STDs..Number.of.diagnosis=[0,1)
Number.of.sexual.partners=[1,10),STDs..number.=[0,1.33)
STDs..number.=[1.33,2.67),STDs..Time.since.first.diagnosis=[1,8)

接下来，我们将 SBRL 算法应用于自行车租赁预测任务。只有把预测自行车数量的回归问题转化为二分类任务时，这才有效。我们通过构建标签 (如果一天的自行车数量超过 4000 辆，则为 1，否则为 0) 随意创建了一个分类任务。

SBRL 学习了以下列表：

rules
If {yr=2011,temp=[-5.22,7.35)} (rule[718]) then positive probability = 0.01041667
else if {yr=2012,temp=[7.35,19.9)} (rule[823]) then positive probability = 0.88125000
else if {yr=2012,temp=[19.9,32.5]} (rule[816]) then positive probability = 0.99253731
else if {season=SPRING} (rule[351]) then positive probability = 0.06410256
else if {yr=2011,temp=[7.35,19.9)} (rule[730]) then positive probability = 0.44444444
else (default rule) then positive probability = 0.79746835

让我们预测一下，在 17 摄氏度的温度下，2012 年一天内自行车的数量将超过 4000 辆。第一条规则不适用，因为它只适用于 2011 年。第二条规则适用，因为时间是在 2012 年，17 摄氏度是在时间间隔 [7.35,19.9]。我们对出租 4000 辆自行车的概率的预测是 88%。

4.5.4 优点

本节一般讨论 IF-THEN 规则的好处。

IF-THEN 规则**很容易解释**。它们可能是可解释模型中最容易解释的。此声明仅适用于规则数量较少、规则条件较短 (最多 3 个) 以及规则组织在决策列表或不重叠的决策集中的情况。

决策规则可以**像决策树那样具有表达能力，同时更紧凑**。决策树通常也会遭受重复的子树，也就是说，当左子节点和右子节点中的分割具有相同的结构时。

使用 IF-THEN 规则的**预测很快**，因为只需要检查几个二进制语句就可以确定哪些规则适用。

决策规则对于输入特征的单调变换具有很强的**鲁棒性**，因为条件中只有阈值会发生变化。它们对于异常值也很健壮，因为它只在条件适用或不适用时才重要。

IF-THEN 规则通常生成稀疏模型，这意味着不包含许多特征。它们**仅为模型选择相关特征**。例如，默认情况下，线性模型为每个输入特征指定权重。不相关的特征可以简单地被 IF-THEN 规则忽略。

像来自 OneR 这样的简单规则可以用作更复杂算法的**基线**。

4.5.5 缺点

此部分处理 IF-THEN 规则的缺点。

关于 IF-THEN 规则的研究和文献主要集中在分类上，几乎**完全忽略了回归**。虽然你可以将一个连续的目标划分为间隔，并将其转化为分类问题，但你总是会丢失信息。一般来说，如果方法可以同时用于回归和分类，则更具吸引力。

这些**特征通常也必须是分类的**。这意味着如果要使用数值特征，必须对它们进行分类。有很多方法可以将一个连续的特征分割成间隔，但这并不是一件微不足道的事情，并且会带来许多没有明确答案的问题。该特征应划分为多少个间隔？分割标准是什么：固定的间隔长度、分位数或其他什么？对连续特征进行分类是一个不容忽视的问题，但经常被忽视，人们只是使用次佳的方法 (就像我们在示例中所做的那样)。

许多旧的规则学习算法容易过拟合。这里给出的算法都至少有一些保护措施来防止过拟合：OneR 是有限的，因为它只能使用一个特征 (只有当特征具有太多的级别 (可能值) 或有许多特征时才有问

题，这等同于多个测试问题)，RIPPER 进行修剪，以及贝叶斯规则列表对决策列表施加了先验分布。

在描述特征和输出之间的线性关系时，决策规则是不好的。这是他们与决策树共享的问题。决策树和规则只能产生阶跃预测函数，其中预测的变化总是离散的阶跃，而不是平滑的曲线。这与输入必须是分类的问题有关。在决策树中，它们通过分割来隐式分类。

4.5.6 软件和替代方法

OneR 在 R 包 OneR 中实现，R 包用于本书中的示例。OneR 还可以在 Weka 机器学习库 中实现，并且可以在 Java，R 和 Python 中使用。RIPPER 也在 Weka 中实现。对于示例，我在 RWeka 包 中使用了 JRIP 的 R 实现。SBRL 可在 R 包 获得 (我在示例中使用)，Python 或 C 实现 提供。

我们不会尝试列出学习决策规则集和列表的所有替代方法，而是会指出一些总结性工作。我们推荐 Fuernkranz 等撰写的《Foundations of Rule Learning》(2012)[ürnkranz12]。对于那些想深入研究该主题的人来说，这是一项有关学习规则的广泛工作。它为思考学习规则提供了一个整体框架，并提出了许多规则学习算法。我们还建议 Weka 规则学习器，该学习器实现了 RIPPER，M5Rules，OneR，PART 等。`IF-THEN` 规则可用于线性模型中，如本书有关 RuleFit 算法的一节中所述。

4.6 RuleFit

Friedman 和 Popescu (2008)[21] 的 RuleFit 算法学习了稀疏线性模型，该模型包括以决策规则形式自动检测到的交互作用。

线性回归模型不考虑特征之间的交互作用。具有像线性模型一样简单且可解释但又集成了特征交互的模型，这难道不方便吗？RuleFit 填补了这一空白。RuleFit 学习具有原始特征以及许多新特征 (决策规则) 的稀疏线性模型。这些新特征捕获了原始特征之间的交互。RuleFit 从决策树自动生成这些特征。通过将分割的决策合并为规则，可以将通过树的每条路径转换为决策规则。节点预测被丢弃，决策规则中仅使用分割：

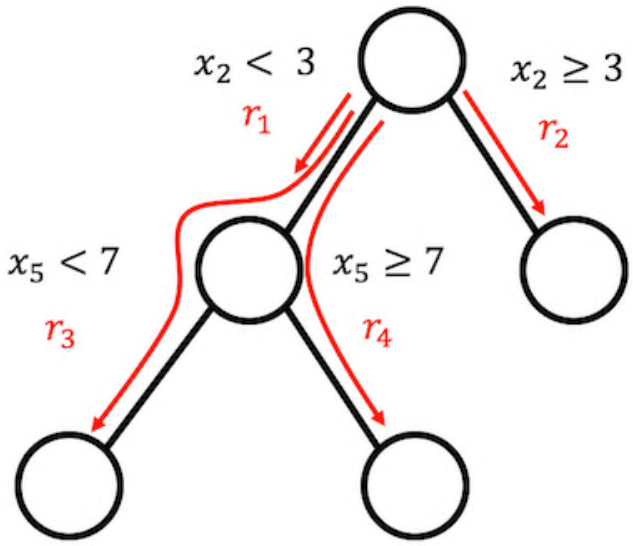


图 4.21. 可以从具有 3 个终端节点的树中生成 4 条规则。

这些决策树从何而来? 这些树被训练来预测感兴趣的结果。这样可以确保分割对于预测任务有意义。生成大量树的任何算法都可以用于 RuleFit，例如随机森林。每棵树都分解为决策规则，这些规则用作稀疏线性回归模型 (Lasso) 中的附加特征。

RuleFit 论文使用波士顿房屋数据来说明这一点：目的是预测波士顿社区的房价中位数。RuleFit 生成的规则之一是：`IF number of rooms > 6.64 AND concentration of nitric oxide < 0.67 THEN 1 ELSE 0`。

RuleFit 还提供了特征重要性度量，可帮助识别对于预测很重要的线性项和规则。根据回归模型的权重可以计算特征重要性。重要性度量可以针对原始特征进行汇总 (特征用于其“原始”形式，也可能用于决策规则中)。

RuleFit 还引入了部分依赖图，通过改变特征来显示预测的平均变化。部分依赖图是一种与模型无关的方法，可以与任何模型一起使用，在有关部分依赖图的一节中进行了说明。

4.6.1 解释和示例

由于 RuleFit 最终会估计一个线性模型，因此其解释与“常规”线性模型的解释相同。唯一的区别是该模型具有从决策规则派生的新特征。决策规则是二进制特征：值为 1 表示满足规则的所有条件，否则值为 0。对于 RuleFit 中的线性项，其解释与线性回归模型中的解释相同：如果特征增加一个单位，预测结果会随着相应的特征权重而变化。

在此示例中，我们使用 RuleFit 预测给定日期的自行车出租数量。该表显示了 RuleFit 生成的五个规则，以及它们的 Lasso 权重和重要性。该计算将在本节后面解释。

Description	Weight	Importance
days_since_2011 > 111 & weathersit in (“GOOD”, “MISTY”)	793	303
37.25 <= hum <= 90	-20	272
temp > 13 & days_since_2011 > 554	676	239
4 <= windspeed <= 24	-41	202
days_since_2011 > 428 & temp > 5	366	179

最重要的规则是：“days_since_2011> 111 & weathersit in (“ GOOD”, “ MISTY”)”，相应的权重为 793。解释是：如果 days_since_2011> 111 & weathersit in (“ GOOD”, “ MISTY”)，那么当所有其他特征值保持固定时，预测的自行车数量将增加 793。从最初的 8 个特征中总共创建了 278 个这样的规则。非常多！但是多亏了 Lasso，278 个中只有 58 个的权重非 0。

计算全局特征重要性表明，温度特征和天数特征是最重要的特征：

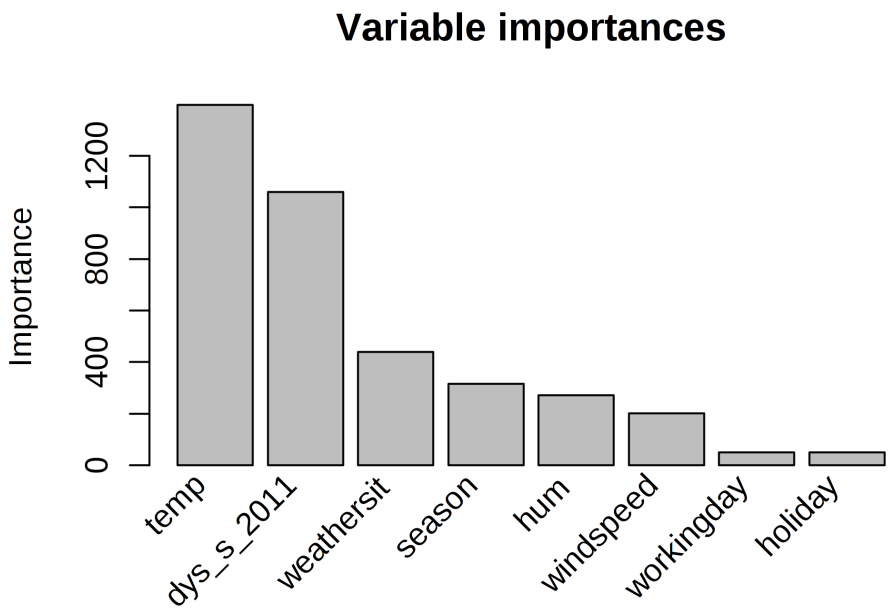


图 4.22. 预测自行车数量的 RuleFit 模型的特征重要性度量。预测的最重要特征是温度特征和天数特征。

特征重要性度量包括原始特征项以及特征出现的所有决策规则的重要性。

解释模板

这种解释类似于线性模型：如果特征 x_j 改变一个单位，则预测结果会改变 β_j，前提是所有其他特征保持不变。决策规则的权重解释是一种特殊情况：如果一个决策规则 r_k 所有条件都适用，那么预测结果变化 α_k (在线性模型中规则 r_k 学到的权重)。

对于分类 (使用逻辑回归而不是线性回归)：如果决策规则所有条件均适用，则事件与无事件的几率会改变 α_k。

4.6.2 理论

让我们更深入地研究 RuleFit 算法的技术细节。RuleFit 由两个部分组成：第一个部分从决策树创建 “规则”，第二个部分以原始特征和新规则作为输入用线性模型拟合 (因此名称为 “RuleFit”)。

步骤 1：规则生成

规则是什么样的？算法生成的规则具有简单的形式。例如：`IF` x_2 `<` `3` `AND` x_5 `<` `7` `THEN` `1` `ELSE` `0`。通过分解决策树来构造规则：到树中节点的任何路径都可以转换为决策规则。用于规则的树被拟合来预测目标结果。因此，分割和生成规则经过优化可以预测你感兴趣的结果。你只需使用 “`AND`” 将导致特定节点的二进制决策连接起来，然后你就有了一条规则。我们希望生成许多多样且有意义的规则。梯度提升用于通过用原始特征 X 对 y 进行回归或分类来拟合决策树的集成。每个生成的树都转换为多个规则。不仅提升树，任何树集成算法都可以用作 RuleFit 生成树。可以用以下通式描述树集成：

$$f(x) = a_0 + \sum_{m=1}^{M} a_m f_m(X)$$

M 是树的数量，$f_m(x)$ 是第 m 个树的预测函数，α 是权重。Bagging，随机森林，AdaBoost 和 MART 产生树集成，可用于 RuleFit。

我们从集成的所有树中创建规则。每个规则 r_m 的形式为：

$$r_m(x) = \prod_{j \in \mathrm{T}_m} I(x_j \in s_{jm})$$

其中 T_m 是第 m 个树中使用的一组特征。I 是指示函数，当特征 x_j 在第 j 个特征的值 s 的指定子集中 (由树分割指定) 时为 1，否则为 0。对于数值特征，s_{jm} 是特征值范围内的间隔。间隔看起来像两种情况之一：

$$x_{s_{jm},\text{lower}} < x_j$$

$$x_j < x_{s_{jm},upper}$$

该特征的进一步分割可能会导致更复杂的间隔。对于分类特征，子集 s 包含特征的某些特定类别。

自行车租赁数据集的示例：

$$r_{17}(x) = I(x_{\text{temp}} < 15) \cdot I(x_{\text{weather}} \in \{\text{good}, \text{cloudy}\}) \cdot I(10 \leq x_{\text{windspeed}} < 20)$$

如果满足所有三个条件，则此规则返回 1，否则返回 0。RuleFit 从树中提取所有可能的规则，而不仅仅是从叶节点中。因此，将创建的另一个规则是：

$$r_{18}(x) = I(x_{\text{temp}} < 15) \cdot I(x_{\text{weather}} \in \{\text{good}, \text{cloudy}\})$$

总共，从具有 t_m 个终端节点的 M 个树的集合中创建的规则数是：

$$K = \sum_{m=1}^{M} 2(t_m - 1)$$

RuleFit 作者介绍的一个技巧是学习具有随机深度的树，以便生成许多具有不同长度的不同规则。请注意，我们丢弃每个节点中的预测值，仅保留将我们引导到节点的条件，然后从中创建规则。决策规则的加权在 RuleFit 的步骤 2 中完成。

另一种方式看步骤 1：RuleFit 从你的原始特征生成一组新特征。这些特征是二进制的，可以表示原始特征的非常复杂的交互。选择规则以最大化预测任务。规则是从协变量矩阵 X 自动生成的。你可以简单地将规则视为基于原始特征的新特征。

步骤 2：稀疏线性模型

你在步骤 1 中获得了许多规则。由于第一步可以看作仅是特征变换，因此你仍然无法完成模型拟合。另外，你希望减少规则数量。除了这些规则之外，原始数据集中的所有 “原始” 特征也将用于稀疏线性模型中。每个规则和每个原始特征都将成为线性模型中的特征并获得权重估计。添加原始特征是因为树无法表示 y 和 x 之间的简单线性关系。在训练稀疏线性模型之前，我们先对原始特征进行调整，以使它们对异常值更加健壮：

$$l_j^*(x_j) = min(\delta_j^+, max(\delta_j^-, x_j))$$

其中 δ_j^- 和 δ_j^+ 是特征 x_j 的数据分布的 δ 分位数。δ 选择为 0.05 意味着在 5% 最小值或 5% 最大值中的任何特征值 x_j 将分别设置为 5% 或 95% 的分位数。根据经验，你可以选择 δ=0.025。另外，必须对线性项进行归一化，以便它们与典型决策规则具有相同的先验重要性：

$$l_j(x_j) = 0.4 \cdot l_j^*(x_j) / std(l_j^*(x_j))$$

0.4 是均匀支持度分布 $s_k \sim U(0, 1)$ 的规则的平均标准差。

我们结合两种类型的特征以生成新的特征矩阵，并使用具有以下结构的 Lasso 训练稀疏线性模型：

$$\hat{f}(x) = \hat{\beta}_0 + \sum_{k=1}^{K} \hat{\alpha}_k r_k(x) + \sum_{j=1}^{p} \hat{\beta}_j l_j(x_j)$$

其中 $\hat{\alpha}$ 是规则特征的估计权重向量，$\hat{\beta}$ 是原始特征的权重向量。由于 RuleFit 使用 Lasso，损失函数会获得附加约束，该约束会强制某些权重获得零估计：

$$(\{\hat{\alpha}\}_1^K, \{\hat{\beta}\}_0^p) = argmin_{\{\hat{\alpha}\}_1^K, \{\hat{\beta}\}_0^p} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)})) + \lambda \cdot \left(\sum_{k=1}^{K} |\alpha_k| + \sum_{j=1}^{p} |b_j| \right)$$

结果是一个线性模型，该模型对所有原始特征和规则都具有线性效应。解释与线性模型相同，唯一的区别是某些特征现在是二进制规则。

步骤 3 (可选)：特征重要性

对于原始特征的线性项，使用标准预测变量来测量特征重要性：

$$I_j = |\hat{\beta}_j| \cdot std(l_j(x_j))$$

其中 β_j 是来自 Lasso 模型的权重，是 $std(l_j(x_j))$ 在数据上线性项的标准差。

对于决策规则而言，重要性的计算公式如下：

$$I_k = |\hat{\alpha}_k| \cdot \sqrt{s_k(1 - s_k)}$$

其中 $\hat{\alpha}_k$ 是决策规则的关联 Lasso 权重，s_k 是数据中特征的支持度，即决策规则适用的数据点的百分比 (其中 $r_k(x) = 0$)：

$$s_k = \frac{1}{n} \sum_{i=1}^{n} r_k(x^{(i)})$$

特征是线性项，也可能在许多决策规则中。我们如何衡量特征的总体重要性？可以为每个单独的预测测量特征的重要性 $J_j(x)$：

$$J_j(x) = I_l(x) + \sum_{x_l \in r_k} I_k(x)/m_k$$

其中 I_l 是 (出现 x_j 的) 线性项的重要性，I_k 是 (出现 x_j 的) 决策规则的重要性，m_k 是是构成规则 r_k 的特征数。通过从所有实例中添加特征重要性，可以为我们提供全局特征重要性：

$$J_j(X) = \sum_{i=1}^{n} J_j(x^{(i)})$$

可以选择实例的子集并计算该组的特征重要性。

4.6.3 优点

RuleFit 自动将**特征交互**添加到线性模型。因此，它解决了必须手动添加交互作用项的线性模型问题，并且对建模非线性关系的问题有所帮助。

RuleFit 可以处理分类和回归任务。

创建的规则易于解释，因为它们是二进制决策规则。规则是否适用于实例。仅当规则中条件的数量不太大时，才能保证良好的可解释性。对于我来说，有 1 到 3 个条件的规则似乎很合理。这意味着树集成中树的最大深度为 3。

即使模型中有很多规则，它们也不适用于每个实例。对于单个实例，只有少数规则适用 (= 具有非零的权重)。这提高了局部可解释性。

RuleFit 提出了许多有用的诊断工具。这些工具与模型无关，因此你可以在本书的模型无关部分中找到它们：特征重要性，部分依赖图和特征交互。

4.6.4 缺点

有时 RuleFit 创建许多规则，这些规则在 Lasso 模型中的权重非零。可解释性随着模型中特征数量的增加而降低。一种有希望的解决方案是强制将特征效应设为单调，这意味着特征的增加必须导致预测的增加。

一个奇闻轶事的缺点：论文声称 RuleFit 的性能很好——通常接近随机森林的预测性能！但是在我亲自尝试过的少数案例下，性能令人失望。只需尝试解决你的问题，然后查看其性能。

RuleFit 过程的最终产物是一个具有额外的花哨特征 (决策规则) 的线性模型。但是，由于它是线性模型，因此权重解释仍然非确定性的。它具有与通常的线性回归模型相同的 “脚注”：“ ...假设所有其他特征均已固定。” 当你有规则重叠时，它会变得有些棘手。例如，用于自行车预测的一个决策规则 (特征) 可以是：“ temp $>$ 10”，而另一个规则可以是 “ temp $>$ 15 & weather=‘GOOD’ ”。如果天气晴朗且温度高于 15 摄氏度，则温度会自动高于 10 摄氏度。在适用第二条规则的情况下，也适用第一条规则。对于第二条规则，估计权重的解释是：“假设所有其他特征保持不变，当天气好且温度在 15 度以上，预测的自行车数量将增加 β_2。”。但是，现在非常清楚的是，“所有其他固定的特征” 是有问题的，因为如果应用规则 2，那么规则 1 也适用，解释是无意义的。

4.6.5 软件和替代方法

Fokkema 和 Christoffersen (2017)[22] 在 R 中实现了 RuleFit 算法，你可以在 Github 上找到 Python 版本。

一个非常相似的框架是 skope-rules，这是一个 Python 模块，它也从集成中提取规则。它学习最终规则的方式有所不同：首先，skope-rules 会基于召回率和精度阈值删除性能不佳的规则。然后，基于规则的逻辑项 (变量 + 较大/较小的运算符) 的多样性和性能 (F1 值) 进行选择，删除重复的规则和相似的规则。最后一步不依赖于使用 Lasso，而是仅考虑袋外 F1 值和构成规则的逻辑项。

4.7 其他可解释模型

可解释模型的列表在不断增长且规模未知。它包括简单的模型，例如线性模型，决策树和朴素的贝叶斯模型，还包括更复杂的模型 (这些模型结合或修改了不可解释的机器学习模型，以使其更具可解释性)。特别是后一种类型的模型的出版物目前正以高频率出版，很难跟上发展的步伐。该书在本节中仅介绍朴素贝叶斯分类器和 k-最近邻。

4.7.1 朴素贝叶斯分类器

朴素贝叶斯分类器使用条件概率的贝叶斯定理。对于每个特征，它根据特征值计算类的概率。朴素贝叶斯分类器独立地计算每个特征的类概率，这相当于特征独立性的强假设。朴素贝叶斯是一个条件概率模型，它对 C_k 的概率建模如下：

$$P(C_k|x) = \frac{1}{Z}P(C_k)\prod_{i=1}^{n}P(x_i|C_k)$$

Z 是一个缩放参数，可确保所有类别的概率之和为 1 (否则它们将不是概率)。类的条件概率是类概率乘以给定类每个特征的概率，并通过 Z 进行归一化。可以使用贝叶斯定理来推导该公式。

由于独立性假设，朴素贝叶斯是一个可解释的模型。可以在模块级别上解释它。由于我们可以解释条件概率，因此对于每个特征，它对特定类别预测的贡献是非常明显的。

4.7.2 k-最近邻

k-最近邻方法可用于回归和分类，并将数据点的最近邻用于预测。对于分类，k-最近邻方法分配实例最近邻的最常见类。为了进行回归，它采用了邻居结果的平均值。棘手的部分是找到合适的 k 并决定如何测量实例之间的距离，从而最终确定邻域。

k-最近邻模型不同于本书中介绍的其他可解释模型，因为它是基于实例的学习算法。如何解读 k-最近邻？首先，没有要学习的参数，因此在模块级别没有可解释性。此外，由于模型本质上是局部的并且没有明确学习的全局权重或结构，因此缺乏全局模型的可解释性。也许在局部可以解释？为了解释预测，你始终可以检索用于预测最近的 k 个点。模型是否可解释仅取决于是否可以“解释”数据集中的单个实例的问题。我认为，如果一个实例包含成百上千个特征，那么它是不可解释的。但是，如果你有少量特征或有将实例简化为最重要特征的方法，那么展示 k-最近邻可以为你提供很好的解释。

第五章　模型无关方法

将解释与机器学习模型 (= 与模型无关的解释方法) 分离具有一些优势 (Ribeiro, Singh 和 Guestrin, 2016[23])。与模型无关 (也称模型不可知) 的解释方法相对于模型特定的解释方法的最大优势是它们的灵活性。当解释方法可以应用于任何模型时，机器学习开发人员可以自由使用他们喜欢的任何机器学习模型。任何建立在对机器学习模型的解释之上的东西 (例如图形或用户界面)，也将变得独立于底层机器学习模型。通常，不仅要评估一种机器学习模型，还要评估多种类型的机器学习模型来解决任务，并且在可解释性方面比较模型时，使用与模型无关的解释会更容易，因为相同的方法可以用于任何类型的模型。

模型无关的解释方法的替代方法是仅使用可解释模型，这通常具有一个很大的缺点，即与其他机器学习模型相比，预测性能会丢失，并且你只能使用一种模型。另一种选择是使用特定于模型的解释方法。这样做的缺点是，它还会将你绑定到一种模型类型，并且将很难切换到其他模型类型。

模型不可知的解释系统的理想方面是 (Ribeiro，Singh 和 Guestrin，2016)：

- **模型的灵活性**：解释方法可以与任何机器学习模型一起使用，例如随机森林和深度神经网络。
- **解释的灵活性**：你不限于某种形式的解释。在某些情况下，线性公式可能会有用，而在其他情况下，特征重要性的图形可能会有用。
- **表示方式的灵活性**：解释系统应该能够使用与所解释模型不同的特征表示方式。对于使用抽象词嵌入向量的文本分类器，可能更希望使用单个词的存在进行解释。

更大的图景

让我们对模型无关的可解释性进行高层次的研究。我们通过收集数据来捕获世界，然后通过学习使用机器学习模型预测 (针对任务的) 数据来进一步抽象世界。可解释性只是帮助人们理解的最上一层。

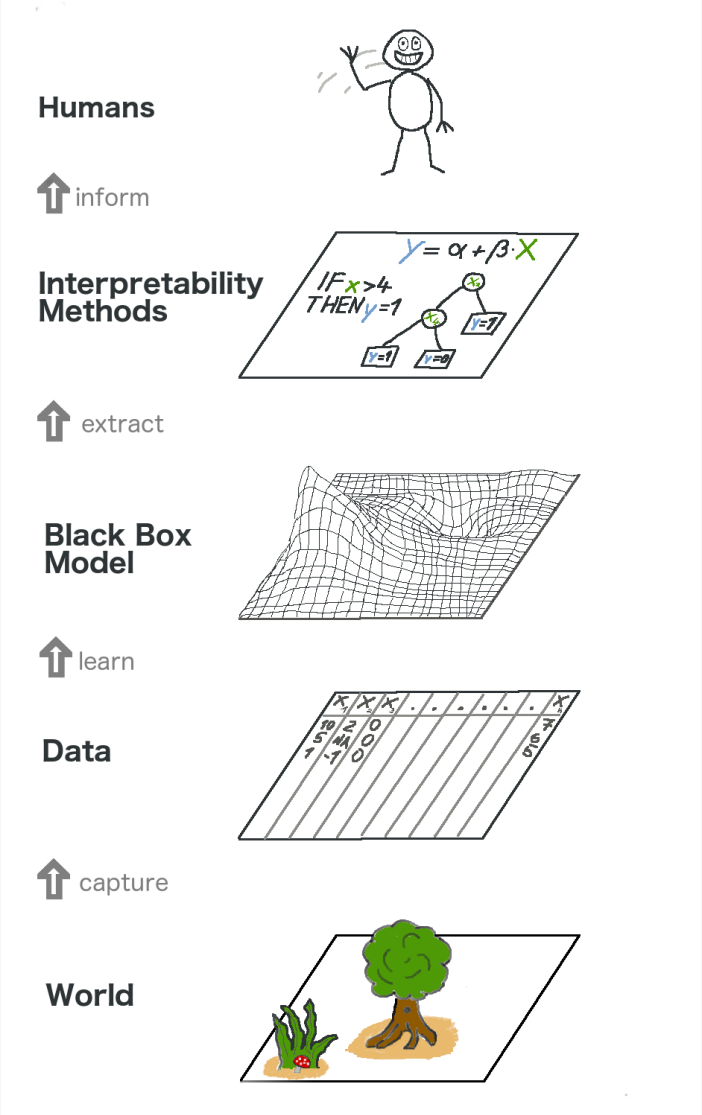


图 5.1. 可解释的机器学习的图景。现实世界在以解释的形式到达人类之前经历了许多层次。

最低层是**世界层** (World)。从字面上看，这可能是自然本身，例如人体的生物学及其对药物的反应方式，还可以是房地产市场等抽象的事物。世界层包含可以观察到的所有有趣的事物。最终，我们想了解一些有关世界的知识并与之互动。

第二层是**数据层** (Data)。我们必须对世界进行数字化处理，以使其可用于计算机并存储信息。数据层包含图像、文本、表格数据等中的任何内容。

通过基于数据层拟合机器学习模型，我们得到了**黑盒模型层** (Black Box Model)。机器学习算法从现实世界中学习数据，以做出预测或找到结构。

在黑盒模型层上方是**可解释性方法层** (Interpretability Methods)，该层有助于我们处理机器学

习模型的不透明性。特定诊断的最重要特征是什么？为什么将金融交易分类为欺诈?

最后一层被**人类** (Human) 占据。看！之所以向你招手，是因为你正在阅读本书，并为黑盒模型提供了更好的解释！人类最终是这些解释的消费者。

这种多层抽象还有助于理解统计学家和机器学习专家在方法上的差异。统计人员处理数据层，例如计划临床试验或设计调查。他们跳过黑盒模型层，然后转到可解释性方法层。机器学习专家处理数据层，例如收集带标签的皮肤癌图像样本或爬取维基百科。然后他们训练了黑盒机器学习模型。跳过了可解释性方法层，人类直接处理了黑盒模型的预测。可解释的机器学习很好地融合了统计学家和机器学习专家的工作，这非常棒。

当然，这张图并不能捕捉到一切：数据可能来自模拟。黑盒模型还会输出可能甚至无法传给人类的预测，而只会提供给其他机器，依此类推。但是总的来说，了解可解释性如何成为机器学习模型之上的这一新层是一个有用的抽象。

5.1 部分依赖图

部分依赖图 (Partial Dependence Plot，简称 PDP 或 PD 图) 显示了一个或两个特征对机器学习模型的预测结果的边际效应 (JH Friedman，2001[1])。部分依赖图可以显示目标和特征之间的关系是线性的、单调的或更复杂的。例如，当应用于线性回归模型时，部分依赖图始终显示线性关系。用于回归的部分依赖函数定义为:

$$\hat{f}_{x_S}(x_S) = E_{x_C}\left[\hat{f}(x_S, x_C)\right] = \int \hat{f}(x_S, x_C) d\mathbb{P}(x_C)$$

x_S 是其部分依赖函数应被绘制的特征，x_C 是在机器学习模型 $\hat{f}$ 中使用的其他特征。通常，集合 S 中只有一个或两个特征。S 中的特征是我们想要了解其对预测的影响的那些。特征向量 x_S 和 x_C 合并组成总特征空间 x。部分依赖性通过在集合 C 中的特征分布上边缘化机器学习模型输出而起作用，因此该函数显示了我们感兴趣的集合 S 中的特征与预测结果之间的关系。通过边缘化其他特征，我们得到了仅依赖于 S 中的特征以及与其他特征的交互作用的函数。

部分依赖函数 $\hat{f}_{x_S}$ 是通过计算训练数据中的平均值来估算的，也称为蒙特卡洛方法:

$$\hat{f}_{x_S}(x_S) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(x_S, x_C^{(i)})$$

部分依赖函数告诉我们特征 S 的给定值对预测的平均边际效应是多少。在此公式中，x_C 是数据集中我们不感兴趣的特征的实际特征值，n 是数据集中的实例数。PDP 的一个假设是 C 中的特征与 S 中的特征不相关。如果违反此假设，则为部分依赖图计算的平均值将包含极不可能或甚至不可能的数据点 (请参见缺点)。

对于机器学习模型输出概率的分类，部分依赖函数显示给定 S 中不同的特征值下特定类别的概率。处理多个类别的一种简单方法是为每个类别绘制一条线或图。

部分依赖图是一种全局方法：该方法考虑所有实例，并给出有关特征与预测结果的全局关系的说明。

分类特征

到目前为止，我们仅考虑了数值特征。对于分类特征，部分依赖很容易计算。对于 (特征) 每个类别值，我们通过强制所有数据实例具有相同类别值来获得 PDP 估计。例如，如果我们查看自行车租赁数据集，并对季节的部分依赖图感兴趣，则会得到 4 个数字，每个季节一个。为了计算 “夏季” 的值，我们将所有数据实例的季节替换为 “夏季”，然后对预测取平均值。

5.1.1 示例

实际上，特征集 S 通常仅包含一个特征或最多包含两个特征，因为一个特征产生 2D 图，而两个特征产生 3D 图。除此之外的一切都非常棘手。甚至在 2D 纸张或显示器上显示 3D 都已经具有挑战性。

让我们回到回归示例，在该示例中，我们可以预测给定一天要租用的自行车数量。首先，我们拟合机器学习模型，然后分析部分依赖关系。在这种情况下，我们拟合了一个随机森林来预测自行车的数量，并使用部分依赖图来可视化模型学习到的关系。下图显示了天气特征对预测自行车数的影响。

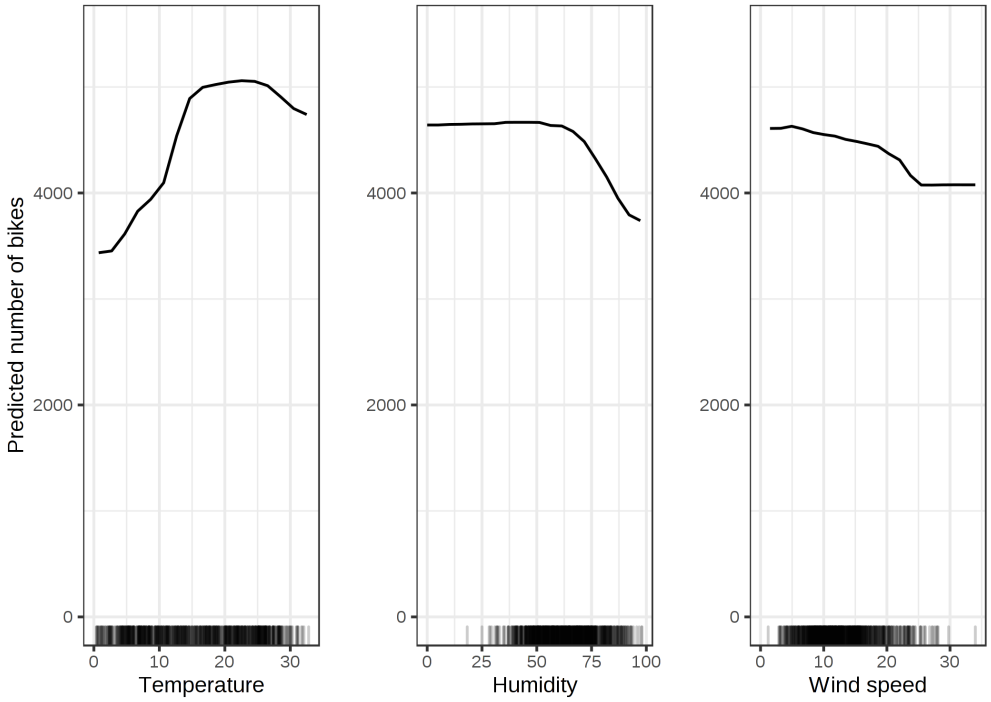


图 5.2. 自行车数量预测模型以及温度、湿度和风速的 PDP。在温度上可以看到最大的差异。温度越高，自行车出租越多。这种趋势上升到 20 摄氏度，然后在 30 摄氏度处变平并略有下降。x 轴上的标记表示数据分布。

对于温暖但不太热的天气，该模型预测平均会租用大量自行车。当湿度超过 60 时，潜在的骑自行车的人越来越不愿意租用自行车。另外，风越大，骑自行车的人就越少，这是有道理的。有趣的是，当风速从 25 km/h 增加到 35 km/h 时，自行车租赁的预测数量不会减少，但是没有太多的训练数据，因此机器学习模型可能无法学习该范围的有意义的预测。至少从直觉上讲，我预计自行车的数量会随着风速的增加而减少，特别是在风速很高时。

为了说明分类特征的部分依赖图，我们研究了季节特征对预测自行车租用的影响。

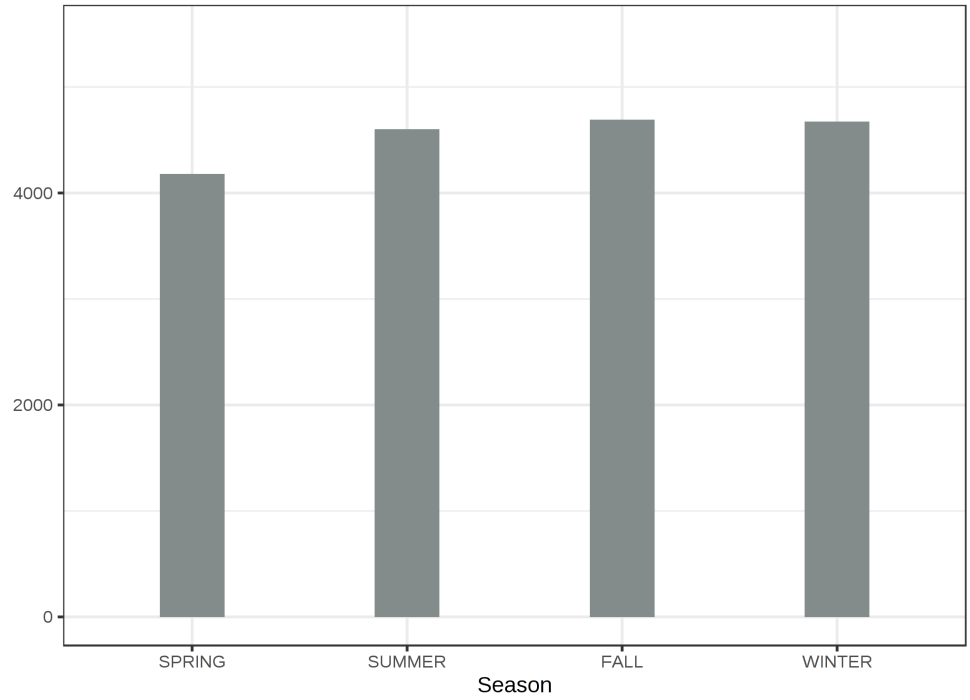


图 5.3. 自行车数量预测模型和季节的 PDP。出乎意料的是，所有季节都对模型预测产生相似的影响，仅在春季，模型预测的自行车租量会减少。

我们还计算宫颈癌分类的部分依赖性。这次，我们拟合了一个随机森林以根据风险因素预测女性是否会患上宫颈癌。我们为随机森林计算并可视化癌症概率对不同特征的部分依赖性：

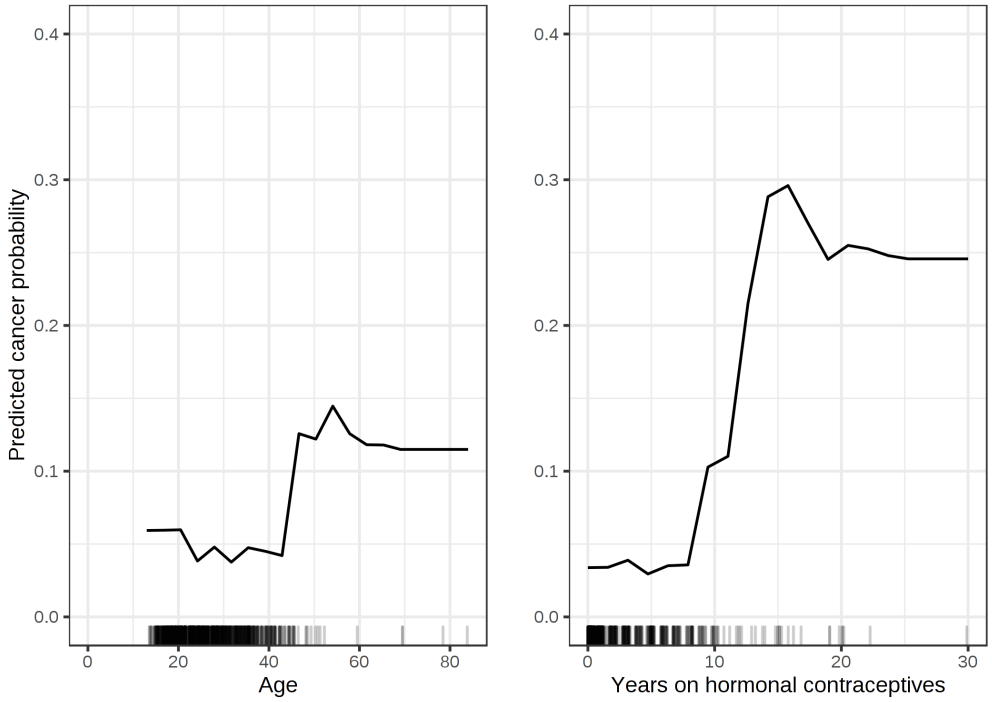


图 5.4. 基于年龄和服用激素避孕药的时间的癌症概率的 PDP。对于年龄，PDP 显示直到 40 岁的概率都很低，之后才增加。服用激素避孕药的时间越长，预计的癌症风险就越高，尤其是 10 年后。对于这两个特征，没有很多具有大数值的数据点可用，因此在这些区域中，PD 估计不太可靠。

我们还可以同时可视化两个特征的部分依赖关系：

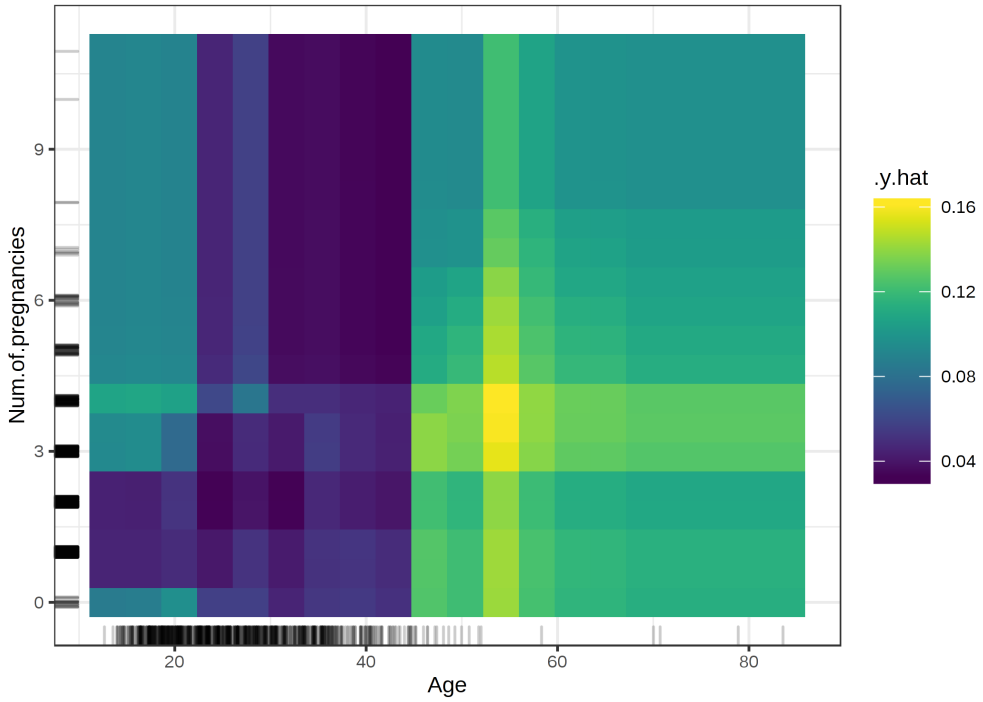


图 5.5. 患癌概率的 PDP 以及年龄和怀孕次数的交互作用。该图显示了 45 岁时患癌的概率增加。对于 25 岁以下的人，怀孕 1 或 2 次的女性的预测癌症风险要低于怀孕 0 次或 2 次以上的女性。但是得出结论时要小心：这可能只是相关性，而不是因果关系！

5.1.2 优点

部分依赖图的计算很**直观**：如果我们强制所有数据点都假定该特征值，则特定特征值处的部分依赖函数表示平均预测。根据我的经验，非专业人士通常会很快理解 PDP 的概念。

如果你为其计算 PDP 的特征与其他特征不相关，则 PDP 可以完美地表示该特征如何平均影响预测。在不相关的情况下，**解释很清楚**：部分依赖图显示了第 j 个特征更改时数据集中的平均预测如何变化。当特征相关时会更加复杂，另请参见缺点。

部分依赖图很**容易实现**。

部分依赖图的计算具有**因果关系**。我们干预一项特征并测量预测的变化。在此过程中，我们分析了特征与预测之间的因果关系。[24] 这种关系对于模型是因果关系的——因为我们明确地将结果建模为特征的函数——但不一定对现实世界有效！

5.1.3 缺点

部分依赖函数中实际的**最大特征数量**为 2。这不是 PDP 的错，而是二维表示 (纸或屏幕) 的错，也是我们无法想象 3 个以上维度的错。

一些 PD 图未显示**特征分布**。忽略分布可能会产生误导，因为你可能会过度解释几乎没有数据的区域。通过显示 RUG (x 轴上的数据点指示器) 或直方图可以轻松解决此问题。

独立性的假设是 PD 图最大的问题。假定针对其计算了部分依赖性的特征与其他特征不相关。例如，假设你要根据一个人的体重和身高来预测一个人走多快。对于其中一个特征 (例如身高) 的部分依赖性，我们假设其他特征 (体重) 与身高不相关，这显然是错误的假设。对于某个身高 (例如 200 厘米) 的 PDP 的计算，我们对体重的边际分布求平均值，其中可能包括 50 公斤以下的体重，这对于 2 米高的人来说是不现实的。换句话说：当特征关联时，我们会在特征分布区域中创建实际概率非常低的新数据点 (例如，某人身高 2 米但体重不足 50 公斤的概率不大)。解决这个问题的一种方法是适用于条件分布而非边际分布的累积局部效应图或简称 ALE 图。

异质效应可能被隐藏，因为 PD 曲线仅显示平均边际效应。假设对于一个特征，你的数据点中的一半与预测具有正相关关系——特征值越大，预测值越大——另一半有负相关性——特征值越小，预测值越大。PD 曲线可能是一条水平线，因为数据集的两半的效果可能会相互抵消。然后，你可以得出结论，该特征对预测没有影响。通过绘制个体条件期望曲线而不是聚合线，我们可以发现异构效应。

5.1.4 软件和替代方法

有许多实现 PDP 的 R 包。我将 iml 包用作示例，但也有 pdp 或 DALEX。在 Python 中，scikit-learn 内置了部分依赖图，你可以使用 PDPBox。

本书介绍的 PDP 替代方法是 ALE 图和 ICE 曲线。

5.2 个体条件期望

个体条件期望 (`Individual Conditional Expectation`，简称 ICE) 图为每个实例显示一条线，该线显示了特征更改时实例的预测如何改变。

特征平均效应的部分依赖图是一种全局方法，因为它不关注特定实例，而是关注整体平均。等价于单个数据实例的 PDP 称为个体条件期望 (ICE) 图 (Goldstein 等人，2017[25])。ICE 图将实例对每个特征的预测依赖关系可视化，每个实例分别产生一条线，而部分依赖图中整体则只有一条线。PDP 是 ICE 图的线的平均值。可以通过以下方式计算线 (和一个实例) 的值：保持所有其他特征相

同，通过用网格中的值替换特征的值创建该实例的变体并使用黑盒模型对这些新创建的实例进行预测。结果是一组具有来自网格的特征值和相应预测的点。

看个体期望而不是部分依赖有什么意义？部分依赖图可能会掩盖由交互作用创建的异构关系。PDP 可以向你显示特征与预测之间的平均关系。仅当要为其计算 PDP 的特征与其他特征之间的交互作用较弱时，这才有效。在交互的情况下，ICE 图将提供更多的见解。

更正式的定义：在 ICE 图，对 $\{(x_S^{(i)}, x_C^{(i)})\}_{i=1}^N$ 中每个实例，曲线 $\hat{f}_S^{(i)}$ 是关于 $x_S^{(i)}$ 的，此时 $x_C^{(i)}$ 固定不变。

5.2.1 示例

让我们返回宫颈癌数据集，看看每个实例的预测如何与特征“年龄”相关联。我们将分析随机森林，随机森林用于预测给定风险因素的情况下女性患癌的概率。在部分依赖图中，我们看患癌症的概率在 50 岁左右时增加，但这是否适用于数据集中的每个女性？ICE 图显示，对于大多数女性而言，年龄效应遵循 50 岁时平均增加的趋势，但也有一些例外：对于少数几位在年轻时具有较高预测概率的女性，预测的癌症概率不会随年龄变化太大。

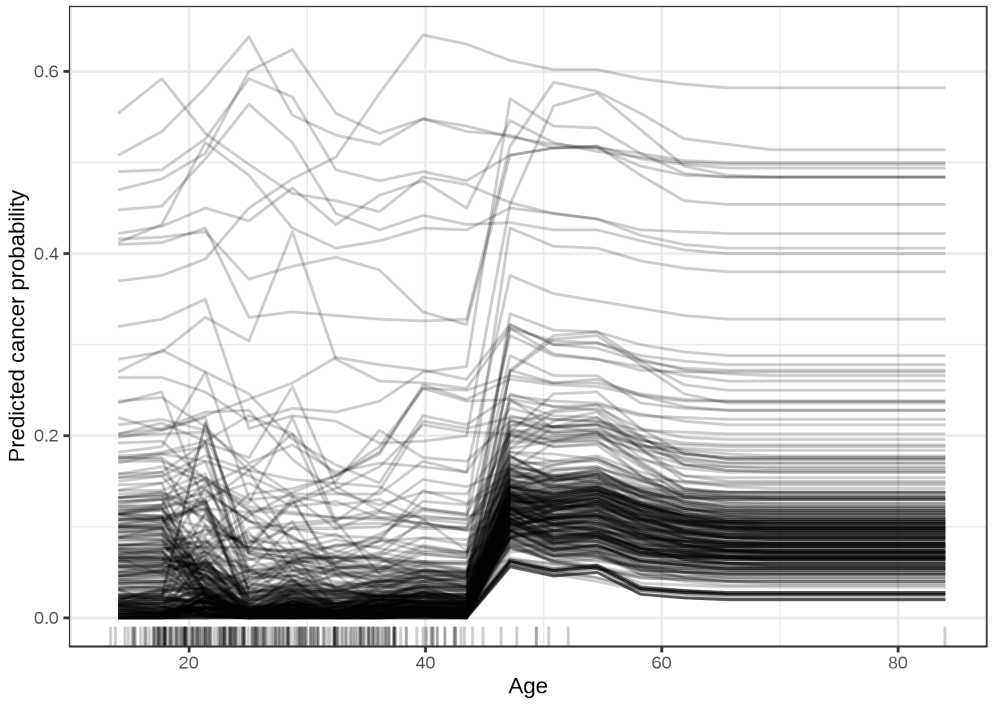


图 5.6. 按年龄划分的宫颈癌概率 ICE 图。每条线代表一位女性。对于大多数女性来说，随着年龄的增长，预测癌症概率会增加。对于某些癌症预测概率高于 0.4 的女性，该预测在较高年龄时变化不大。

下图显示了预测自行车租赁的 ICE 图。基本的预测模型是随机森林。

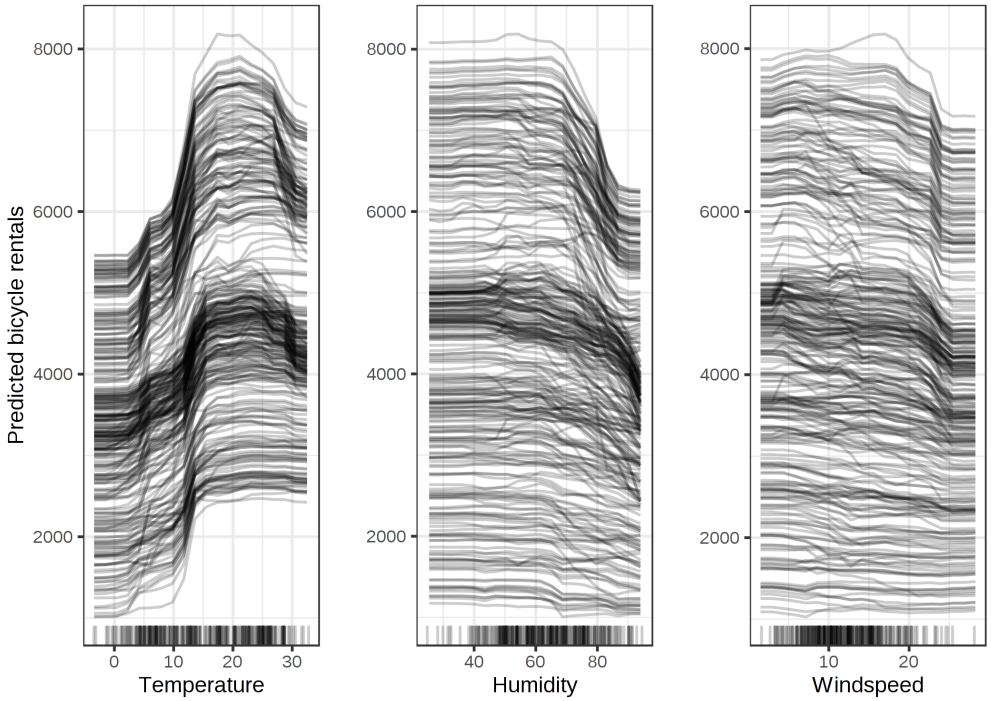


图 5.7. 按天气状况预测的自行车租赁的 ICE 图。可以观察到与部分依赖图相同的效果。

所有曲线似乎都遵循相同的路线，因此没有明显的相互作用。这意味着 PDP 已经很好地概括了显示的特征与预测的自行车数量之间的关系。

中心化个体条件期望图 (Centered ICE Plot)

ICE 图存在一个问题：有时很难判断 ICE 图在个体之间是否不同，因为它们从不同的预测开始。一种简单的解决方案是将曲线中心化于特征中的某个点，并仅显示到该点的预测差异。结果图称为中心化 ICE 图 (c-ICE)。将曲线锚定在特征的下端是一个不错的选择。新曲线定义为：

$$\hat{f}_{cent}^{(i)} = \hat{f}^{(i)} - \mathbf{1}\hat{f}(x^a, x_C^{(i)})$$

其中 $\mathbf{1}$ 是具有适当数量尺寸 (通常为一个或两个) 的 1 的矢量，$\hat{f}$ 是拟合模型，x^a 是锚定点。

示例 例如，绘制宫颈癌年龄 ICE 图，并以观察到的最小年龄为中心：

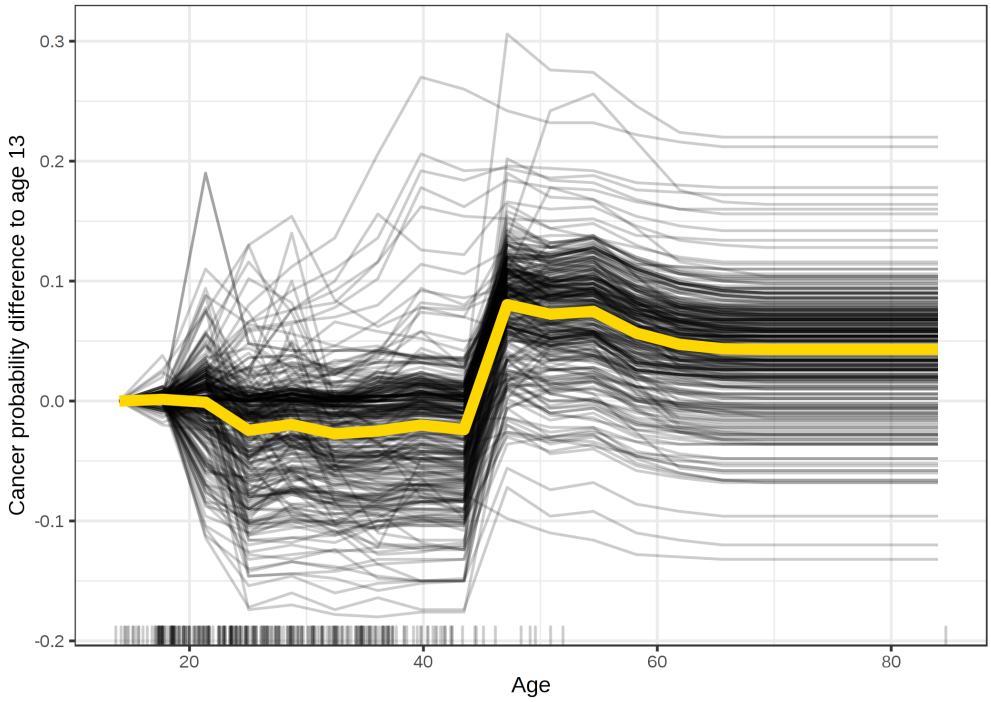


图 5.8. 按年龄预测癌症概率的中心化 ICE 图。在 14 岁时，线固定为 0。与 14 岁相比，大多数女性的预测一直保持不变，直到 45 岁时预测的概率增加。

中心化 ICE 图使比较单个实例的曲线变得更加容易。如果我们不希望看到预测值的绝对变化，而是希望预测与特征范围的固定点相比的差异，这将很有用。

让我们看一下用于自行车租赁预测的中心化 ICE 图：

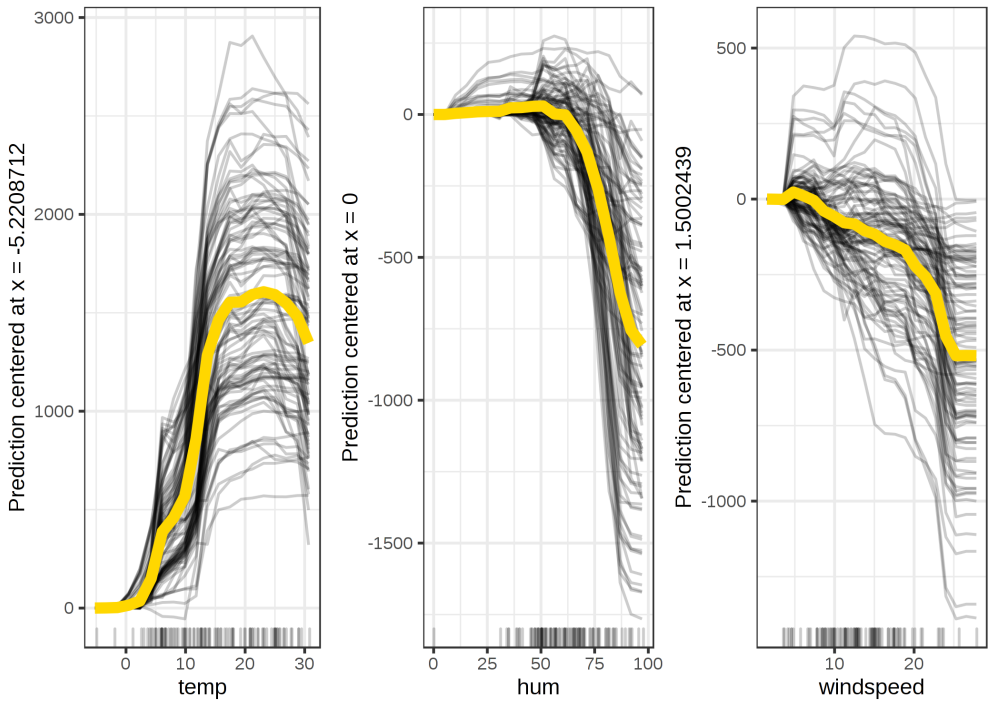


图 5.9. 根据天气状况预测的自行车数量的中心化 ICE 图。这些线表示与各个特征值处于其观测最小值时的预测值之间的差异。

导数个体条件期望图 (Derivative ICE Plot)

另一种使从视觉上更容易发现异质性的方法是观察预测函数相对于特征的导数。生成的图称为导数 ICE 图 (d-ICE)。函数 (或曲线) 的导数告诉你是否发生了变化以及它们发生的方向。利用导数 ICE 图，很容易发现特征值的范围，在该范围内，黑盒模型预测对实例会发生变化 (至少某些)。如果分析的特征 x_S 和其他特征 x_C 之间没有交互作用，那么预测函数可以表示为:

$$\hat{f}(x) = \hat{f}(x_S, x_C) = g(x_S) + h(x_C), \quad \text{with} \quad \frac{\delta \hat{f}(x)}{\delta x_S} = g'(x_S)$$

如果没有交互作用，则所有情况下的各个偏导数都应相同。如果它们不同，则是由于交互作用，并且在 d-ICE 图中可见。除了显示预测函数的导数相对于 S 中特征的的各个曲线外，显示导数的标准差还有助于突出显示 S 中特征中具有估计异质性的区域。d-ICE 图需要很长时间才能计算出来，这是不切实际的。

5.2.2 优点

与部分依赖图相比，个体条件期望曲线**更直观**。如果我们改变感兴趣的特征，则一条线代表一个实例的预测。

与部分依赖图不同，ICE 曲线可以**揭示异质关系**。

5.2.3 缺点

ICE 曲线只能**有意义地显示一个特征**，因为两个特征将需要绘制多个重叠曲面，你将在图中看不到任何内容。

ICE 曲线与 PDP 面临相同的问题：如果感兴趣的特征与其他特征相关联，则根据联合特征分布，**线中的某些点可能是无效的数据点**。

如果绘制了许多 ICE 曲线，该图**可能会过于拥挤**，你将看不到任何东西。解决方案：要么为线条添加一些透明度，要么仅绘制线条的样例。

在 ICE 绘图中，**很难看到平均值**。这有一个简单的解决方案：将个体条件期望曲线与部分依赖图结合起来。

5.2.4 软件和替代方法

ICE 图在 R iml 包 (用于这些示例)，ICEbox[26] 和 pdp 中实现。另一个做类似于 ICE 的 R 包是 condvis。

5.3 累积局部效应图

累积局部效应 [27] (`Accumulated Local Effects Plot`) 描述了特征平均如何影响机器学习模型的预测。ALE 图是部分依赖图 (PDP) 的更快、更无偏的替代方法。

我建议先阅读有关部分依赖图的小节，因为它们更易于理解，并且两种方法都有相同的目标：两者都描述了特征如何平均影响预测。在下面的内容中，我想让你相信，当特征相关时，部分依赖图存在严重问题。

5.3.1 动机和直觉

如果机器学习模型的特征相关，则部分依赖图将不可信。计算与其他特征强相关的特征的部分依赖图涉及对在实际中不太可能出现的人工数据实例的平均预测。这会极大地影响估计的特征效应。想象一下，为机器学习模型计算部分依赖图，该模型根据房间数量和居住面积来预测房屋的价格。我们对居住面积对预测值的影响感兴趣。提醒一下，部分依赖图的配方如下：1) 选择特征。2) 定义网格。3) 每个网格值：a) 用网格值替换特征，b) 平均预测。4) 绘制曲线。为了计算 PDP 的第一个网格值——例如 30 平方米——我们将所有实例的居住面积替换为 30 平方米，即使是拥有 10 个房间的房屋也是如此。在我看来，这样生成的房屋就很不寻常。部分依赖图将这些不切实际的房屋包

括在特征效应估计中，并假装一切都很好。下图说明了两个相关特征以及部分依赖图方法对不太可能发生的实例的预测取平均的结果。

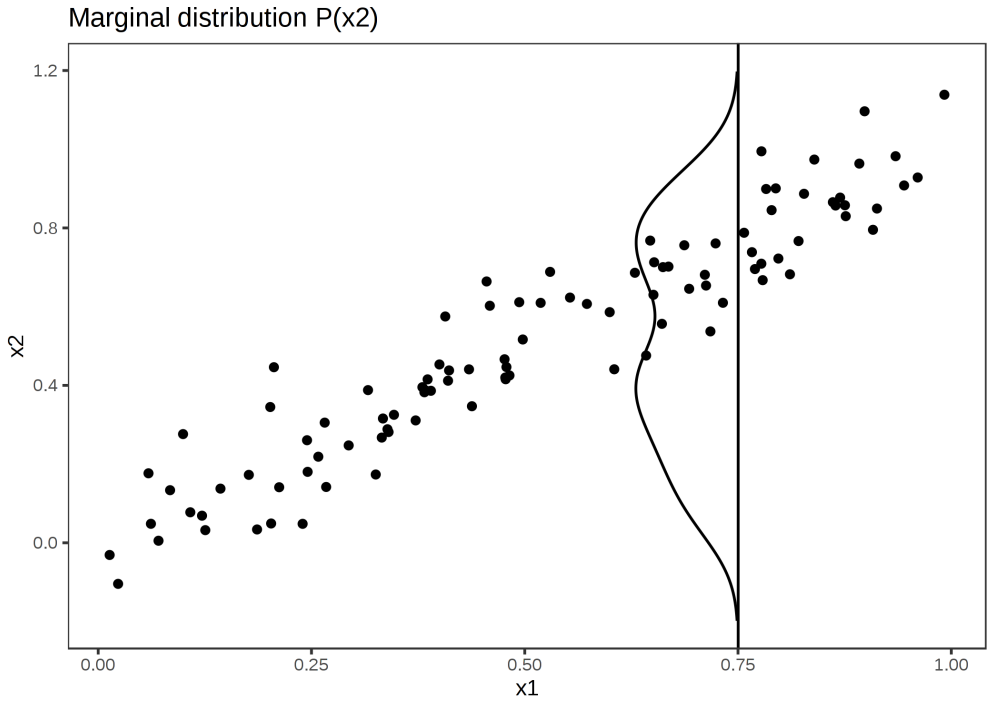


图 5.10. 高度相关的特征 x_1 和 x_2。为了计算 x_1 在 0.75 时的特征效应，PDP 错误地假设 x_2 在 $x_1 = 0.75$ 处的分布与 x_2 的边际分布相同 (垂直线)，则用 0.75 替换所有实例的 x_1。这导致不太可能的 x_1 和 x_2 的组合 (例如 $x_1 = 0.75$ 时 $x_2 = 0.2$)，PDP 将其用于计算平均效应。(注：在本书中特征效应与特征影响描述相同概念)

我们如何做才能得到尊重特征相关性的特征效应估计？我们可以对特征的条件分布求平均值，即在 x_1 的网格值处，对 x_1 值类似的实例的预测求平均值。使用条件分布来计算特征效应的解决方案称为边际图 (`Marginal Plot`) 或 M 图 (名称易混淆，因为它们基于条件而不是边际分布)。等一下，我不是说谈论 ALE 图的吗？M 图不是我们正在寻找的解决方案。为什么 M 图无法解决我们的问题？如果我们平均 30 平方米左右的所有房屋的预测，我们会估计居住面积和房间数量的联合效应，因为它们的相关性。假设居住面积对房屋的预测值没有影响，而只有房间数才有影响。M 图将仍然显示出居住面积会增加预测值，因为房间数会随居住面积的增加而增加。下图显示了两个相关特征，M 图如何工作。

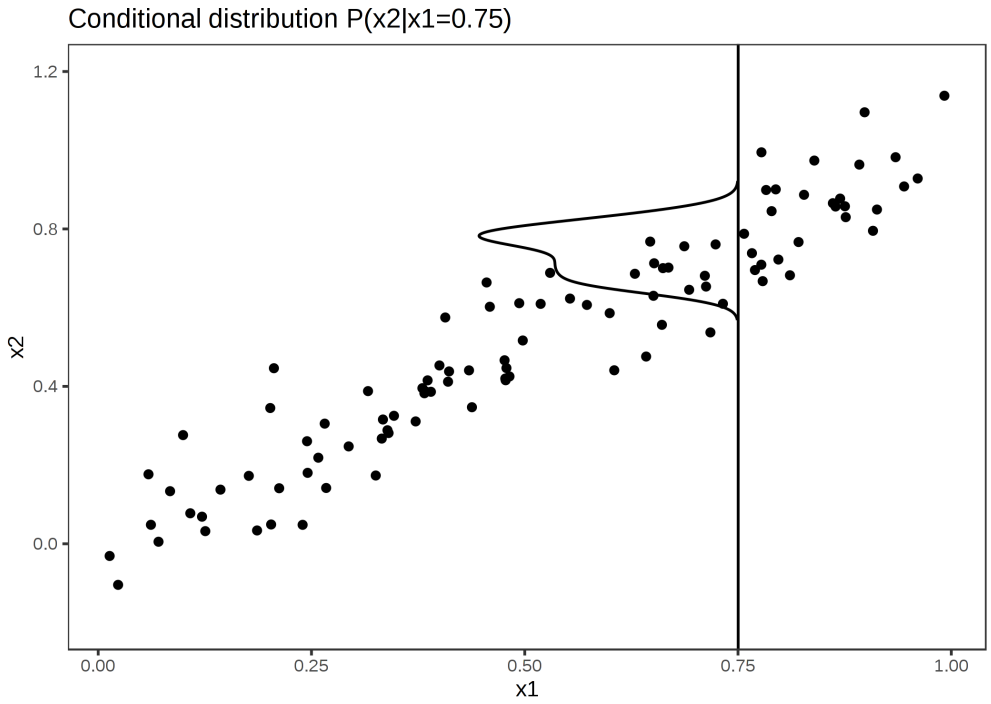


图 5.11. 高度相关的特征 x_1 和 x_2。M 图在条件分布上取平均值。此处 x_2 在 $x_1 = 0.75$ 处的条件分布。平均局部预测会导致混合两种特征的特征效应。

M 图避免了对不太可能出现的数据实例的平均预测，但是它们将特征的效应与所有相关特征的效应混合在一起。ALE 图通过基于特征的条件分布来计算预测差异而不是平均值，从而解决了这一问题。对于 30 平方米的居住面积的效应 (或称影响)，ALE 方法使用所有大约 30 平方米的房屋，得到 (假装) 这些房屋为 31 平方米的预测减去 (假装) 为 29 平方米的预测。这给了我们居住区域的纯粹效应，并且没有将效应与相关特征的效应混合在一起。差异的使用会阻止其他特征的影响。下图直观地说明了如何计算 ALE 图。

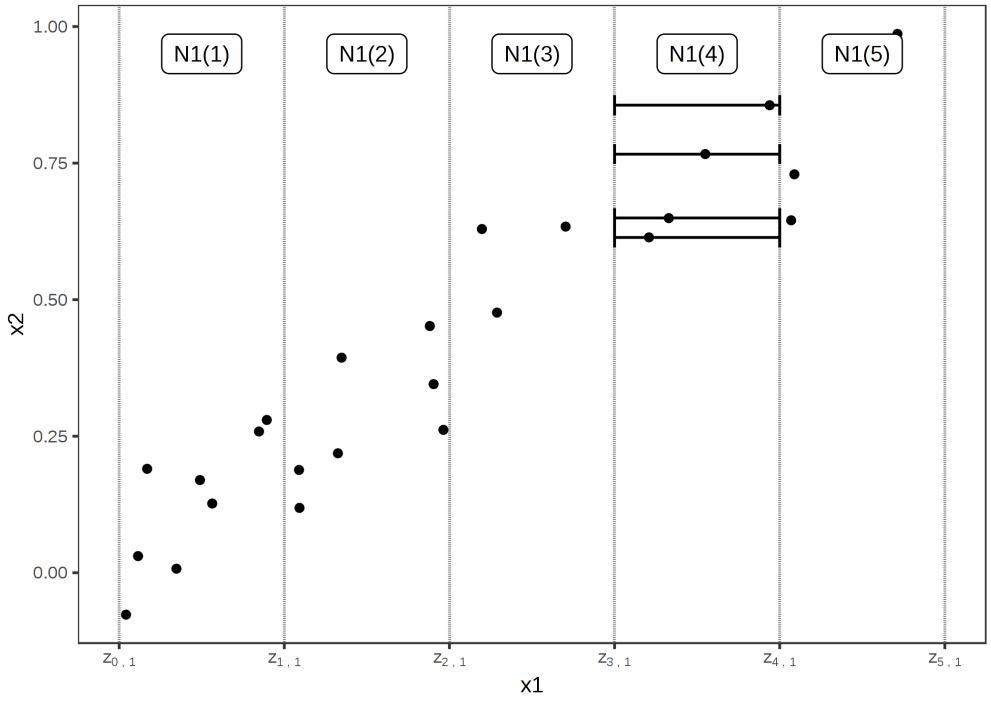


图 5.12. 与 x_2 相关的特征 x_1 的 ALE 计算。首先，我们将特征划分为间隔 (垂直线)。对于间隔中的数据实例 (点)，当我们用间隔的上限和下限 (水平线) 替换特征时，我们计算预测值的差异。这些差异随后被累积并中心化，从而形成 ALE 曲线。

总结每种类型的图 (PDP，M，ALE) 如何在某个网格值 v 下计算 x_j 的特征效应：

部分相关图：“让我展示对于特征 x_j 当每个数据实例具有值 v 时模型平均预测的结果。我忽略了值 v 是否对所有数据实例都有意义。”

M 图：“让我告诉你模型对于特征 x_j 的值接近 v 的数据实例平均预测什么。效应可能是由于该特征，也由于相关的特征。”

ALE 图：“让我向你展示该窗口中数据实例的模型预测如何在围绕 v 的特征 x_j 的一个小的 “窗口” 中变化。”

5.3.2 理论

PD，M 和 ALE 图在数学上有何不同？这三种方法的共同点是将复杂的预测函数 f 简化为仅依赖一个 (或两个) 特征的函数。这三种方法都通过平均其他特征的效应来简化特征，但是在计算预测平均值或预测差异以及是否对边际或条件分布进行平均方面有所不同。

部分依赖图对边际分布的预测进行平均。

$$\begin{aligned}\hat{f}_{x_S,PDP}(x_S) &= E_{X_C}\left[\hat{f}(x_S, X_C)\right] \\ &= \int_{x_C} \hat{f}(x_S, x_C)\mathbb{P}(x_C)dx_C\end{aligned}$$

这是预测函数 f 的值，在特征值 x_S 处，对 x_C 中的所有特征取平均值。平均是指计算集合 C 中特征的边际期望 E，它是通过概率分布加权的预测的积分。听起来很奇怪，但是要计算边际分布上的期望值，我们只需获取所有数据实例，强制它们对集合 S 中的特征具有特定的网格值，并平均此操作下数据集的预测值。这个过程确保我们对特征的边际分布进行平均。

M 图对条件分布的预测平均。

$$\begin{aligned}\hat{f}_{x_S,M}(x_S) &= E_{X_C|X_S}\left[\hat{f}(X_S, X_C)|X_S = x_s\right] \\ &= \int_{x_C} \hat{f}(x_S, x_C)\mathbb{P}(x_C|x_S)dx_C\end{aligned}$$

与 PDP 相比唯一改变的是，我们根据感兴趣的特征的每个网格值来平均预测，而不是假设每个网格值的边际分布。在实践中，这意味着我们必须定义一个邻域，例如，为了计算 30 平方米对房屋预测值的影响，我们可以对 28 到 32 平方米之间的所有房屋的预测值求平均。

ALE 图对预测的变化进行平均，然后将其累积在网格上 (稍后会在计算中提供更多信息)。

$$\begin{aligned}\hat{f}_{x_S,ALE}(x_S) &= \int_{z_{0,1}}^{x_S} E_{X_C|X_S}\left[\hat{f}^S(X_s, X_c)|X_S = z_S\right] dz_S - \text{constant} \\ &= \int_{z_{0,1}}^{x_S}\int_{x_C} \hat{f}^S(z_s, x_c)\mathbb{P}(x_C|z_S)dx_C dz_S - \text{constant}\end{aligned}$$

该公式揭示了与 M 图的三个差异。首先，我们平均预测的变化，而不是预测本身。将该变化定义为梯度 (但稍后，对于实际计算，将替换为某个间隔内的预测差异)。

$$\hat{f}^S(x_s, x_c) = \frac{\delta\hat{f}(x_S, x_C)}{\delta x_S}$$

第二个差异是 z 上的积分。我们在集合 S 中的特征范围内累积局部梯度，这给了我们特征对预测的影响。对于实际计算，将 z 替换为间隔网格，在该间隔上我们计算预测的变化。ALE 方法不是直接平均预测，而是计算以特征 S 为条件的预测差异，并对特征 S 上的导数进行积分以估计效应。好吧，这听起来很愚蠢。导数和积分通常会互相抵消，例如先减去，然后再加上相同的数字。为什么在这里有意义？导数 (或区间差) 隔离感兴趣特征的效应并阻止相关特征的效应。

ALE 图与 M 曲线的第三个差异是我们从结果中减去常数。此步骤将 ALE 图中心化，以便数据的平均效应为零。

仍然存在一个问题：并非所有模型都带有梯度，例如随机森林没有梯度。但是，你将看到，实际的计算没有梯度，并且使用了间隔。让我们深入研究 ALE 图的估计。

5.3.3 估计

首先，我将描述如何为单个数值特征估计 ALE 图，然后是两个数值特征以及单个分类特征。为了估计局部效应，我们将特征划分为许多区间并计算预测值之间的差异。此过程近似梯度，也适用于没有梯度的模型。

首先，我们估计非中心化效应：

$$\hat{\tilde{f}}_{j,ALE}(x) = \sum_{k=1}^{k_j(x)} \frac{1}{n_j(k)} \sum_{i:x_j^{(i)} \in N_j(k)} \left[f(z_{k,j}, x_{\setminus j}^{(i)}) - f(z_{k-1,j}, x_{\setminus j}^{(i)}) \right]$$

让我们把这个公式分解，从右边开始。这个名为**累积局部效应**很好地反映了这个公式的所有组成部分。ALE 方法的核心是计算预测中的差异，因此我们用网格值 z 替换感兴趣的特征。预测中的差异是特征在特定间隔内对单个实例的**效应**。右边的总和将间隔 (作为邻域 $N_j(k)$ 显示在公式中) 内所有实例的效应相加。我们将该总和除以该间隔中的实例数，以获得此间隔的预测平均差。间隔中的此平均值由名称 ALE 中的术语“局部” 覆盖。左边的和符号表示我们在所有间隔内累积平均效应。例如，位于第三个间隔中的特征值的 (非中心化) ALE 是第一个间隔、第二个间隔和第三个间隔的效应之和。ALE 中的 “ 累积” 一词反映了这一点。

将效应中心化，因此平均效应为零。

$$\hat{f}_{j,ALE}(x) = \hat{\tilde{f}}_{j,ALE}(x) - \frac{1}{n} \sum_{i=1}^{n} \hat{\tilde{f}}_{j,ALE}(x_j^{(i)})$$

与数据的平均预测值相比，ALE 值可以解释为某一特定值下特征的主要效应。例如，当第 j 个特征值为 3 时 ($x_j = 3$)，ALE 估计值为 -2，则预测值比平均预测值低 2。

特征分布的分位数用作定义间隔的网格。使用分位数可确保每个间隔中有相同数量的数据实例。分位数的缺点是间隔的长度可能非常不同。如果感兴趣的特征非常偏斜 (例如，许多低值而只有少数非常高的值)，则这可能会导致某些 ALE 图出现异常。

用于两个特征交互作用的 ALE 图

ALE 图还可以显示两个特征的交互作用。计算原理与单个特征相同，但是我们使用矩形单元而不是间隔，因为我们必须在二维中累积效应。除了调整总体平均效应外，我们还调整两个特征的主要效应。这意味着针对两个特征的 ALE 会估计二阶效应，其中不包括特征的主要效应。换句话说，两个特征的 ALE 仅显示两个特征的附加交互效应。我为你省却了 2D ALE 图的公式，因为它们冗长，读起来很不愉快。如果你对这个计算感兴趣，请参考论文的公式 (13)—(16)。我将依靠可视化来发展二阶 ALE 计算的直觉。

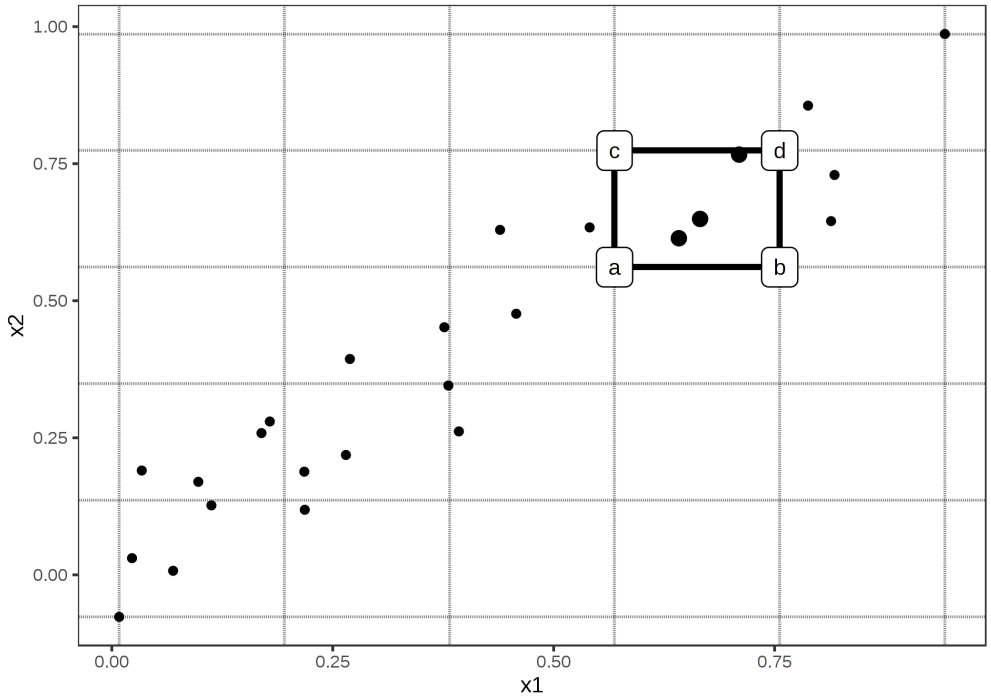


图 5.13. 2D-ALE 的计算。我们在这两个特征上放置一个网格。在每个网格单元中，我们计算内部所有实例的二阶差。我们首先用单元格角落的值替换 x_1 和 x_2 的值。如果 a、b、c 和 d 代表 “角落点”——操作实例的预测 (如图中所示)，则二阶差为 (d-c) - (b-a)。每个单元的平均二阶差在网格上累积并中心化。

在上图中，由于相关性，许多单元为空。在 ALE 图中，可以使用灰色或深色方框将其可视化。或者，你可以用最接近的非空单元格的 ALE 估计值代替空单元格的 ALE 估计值。

由于两个特征的 ALE 估计仅显示特征的二阶效应，因此需要特别注意解释。二阶效应是在考虑了特征的主要效应之后，特征的附加的交互效应。假设两个特征不交互，但是每个特征对预测结果具有线性效应。在每个特征的一维 ALE 图中，我们将看到一条直线作为估计的 ALE 曲线。但是，当我们绘制 2D ALE 估计时，它们应该接近于零，因为二阶效应只是附加的交互效应。ALE 图和 PD 图在这方面有所不同：PDP 始终显示总效应，ALE 图显示一阶或二阶效应。这些是不依赖于基础数学的设计决策。你可以减去部分依赖图中的低阶效应得到纯的主要效应或二阶效应，或者，你可以通过避免减去低阶效应得到总 ALE 图的估计。

累积局部效应也可以针对任意更高的阶数 (三个或更多特征的交互作用) 进行计算，但是正如 PDP 一节所述，最多只有两个特征才有意义，因为更高的交互无法可视化甚至无法有意义地解释。

ALE 用于分类特征

累积局部效应方法 (根据定义) 需要特征值具有顺序，因为该方法会沿特定方向累积效应。分类特征没有任何自然顺序。为了计算分类特征的 ALE 图，我们必须以某种方式创建或找到一个顺序。

类别的顺序会影响累积局部效应的计算和解释。

一种解决方案是基于其他特征根据类别的相似性对类别进行排序。两类之间的距离是每个特征的距离之和。逐特征距离比较了两个类别中的累积分布，也称为 Kolmogorov-Smirnov 距离 (对于数值特征) 或相对频率表 (对于分类特征)。一旦有了所有类别之间的距离，便可以使用多维缩放将距离矩阵压缩为一维距离度量。这为我们提供了基于相似度的类别顺序。

为了使这一点更清楚一些，下面是一个示例：让我们假设我们有两个分类特征 “季节” 和 “天气”，以及一个数值特征 “温度”。对于第一个分类特征 (季节)，我们要计算 ALE。该特征类别为 “春季”，“夏季”，“秋季”，“冬季”。我们开始计算 “春天” 和 “夏天” 类别之间的距离。距离是温度特征和天气特征的距离总和。对于温度，我们采用 “春季” 季节的所有实例，计算经验累积分布函数，对 “夏季” 季节的实例进行相同的操作，并使用 Kolmogorov-Smirnov 统计量度它们的距离。对于天气特征，我们为所有 “春季” 实例计算每种天气类型的概率，对 “夏季” 实例执行相同的操作，并求出概率分布中的绝对距离。如果 “春季” 和 “夏季” 的温度和天气有很大不同，则总类别距离很大。我们对其他季节对重复此过程，并通过多维缩放将所得距离矩阵缩小为一维。

5.3.4 示例

让我们看一下 ALE 图的实现。我构建了一个场景，其中部分依赖图失效了。该场景包括一个预测模型和两个高度相关的特征。预测模型主要是线性回归模型，但是在这两个我们从未观察到的特征的组合中做了一些奇怪的事情。

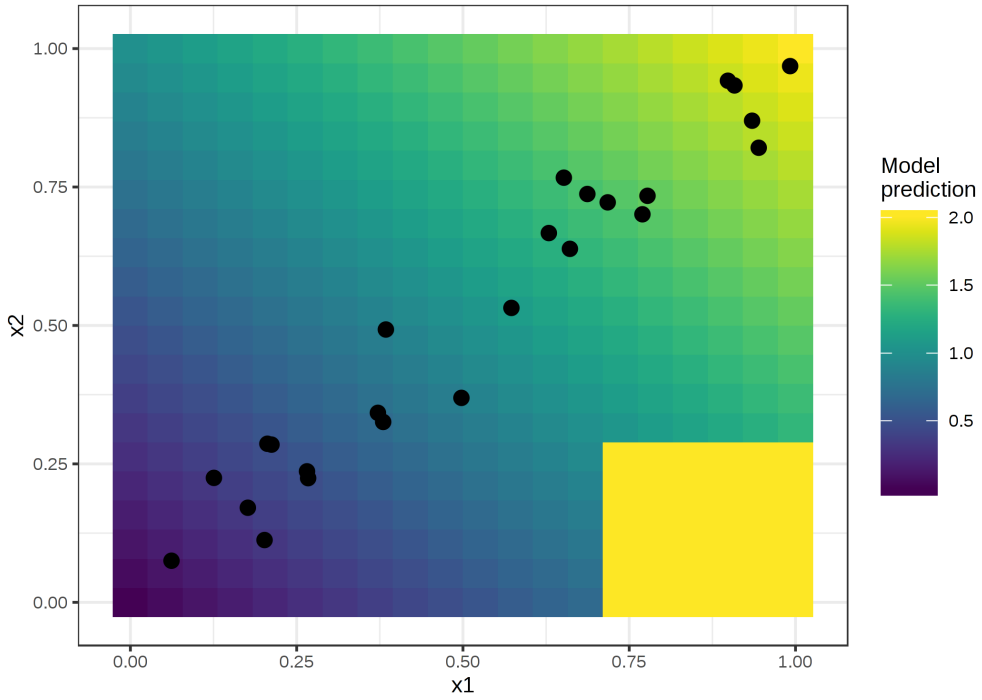


图 5.14. 两个特征和预测结果。模型预测两个特征的总和 (阴影背景)，但如果 x_1 大于 0.7 且 x_2 小于 0.3，则模型始终预测 2。该区域远离数据分布 (点云)，并且不会影响模型的性能，也不应影响其解释。

这是现实的、相关的场景。训练模型时，学习算法会将现有训练数据实例的损失降至最低。奇怪的事情可能发生在训练数据的分布之外，因为该模型不会因为在这些区域做奇怪的事情而受到惩罚。远离数据分布称为外推法 (`Extrapolation`)，也可用于欺骗机器学习模型，在下一章对抗样本一节中对此进行了介绍。在我们的小示例中，看到部分依赖图与 ALE 图相比如何表现。

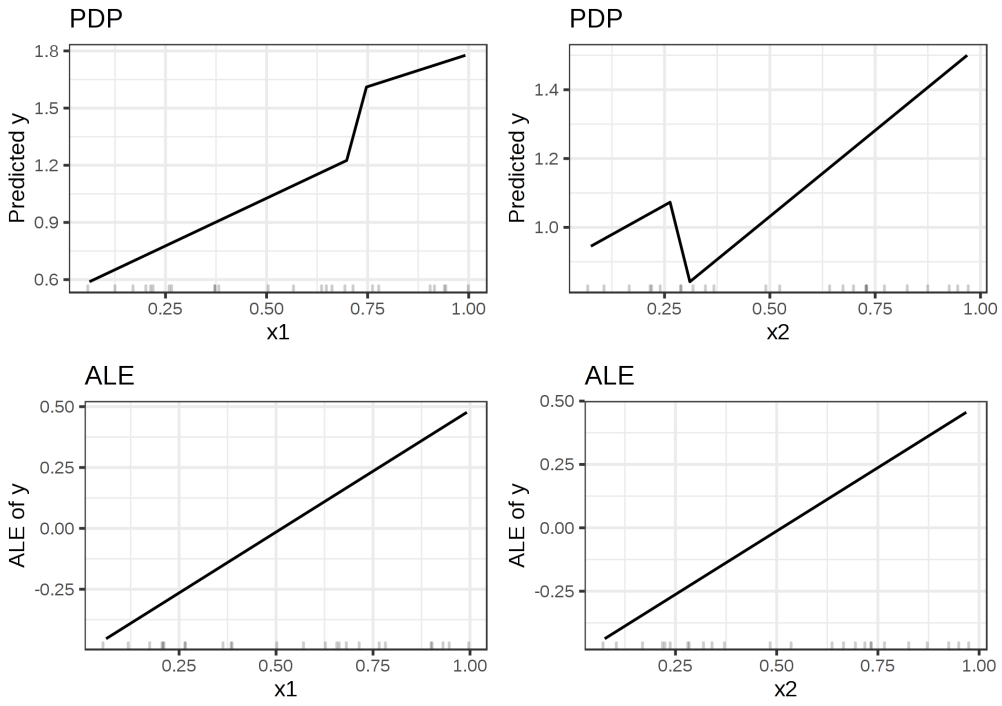


图 5.15. 用 PDP (上排) 和 ALE (下排) 计算的特征效应比较。PDP 估计受数据分布之外模型的奇怪行为 (图中的陡峭跳跃) 影响。ALE 图正确识别了机器学习模型在特征和预测之间具有线性关系，而忽略了没有数据的区域。

但是，看到我们的模型在 $x_1 > 0.7$ 和 $x_2 < 0.3$ 时表现得很奇怪，这不是很有趣吗？是的，其实也不是。由于这些数据实例在物理上可能是不可能的，或者至少是极不可能的，因此通常无需考虑这些实例。但是，如果你怀疑测试分布可能略有不同，并且某些实例实际上在该范围内，那么将这一区域包括在特征效应的计算中将很有趣。必须有意识地决定是否包括我们尚未观察到数据的区域，并且它不应该是选择方法 (如 PDP) 的副作用。如果你怀疑该模型以后会与分布不同的数据一起使用，我建议使用 ALE 图并模拟你期望的数据分布。

转到一个真实的数据集，让我们根据天气和天数来预测所租自行车的数量，并检查 ALE 图是否确实如预期那样有效。我们训练回归树以预测给定日期的自行车出租数量，并使用 ALE 图分析温度，相对湿度和风速如何影响预测。让我们看一下 ALE 图是怎么说的：

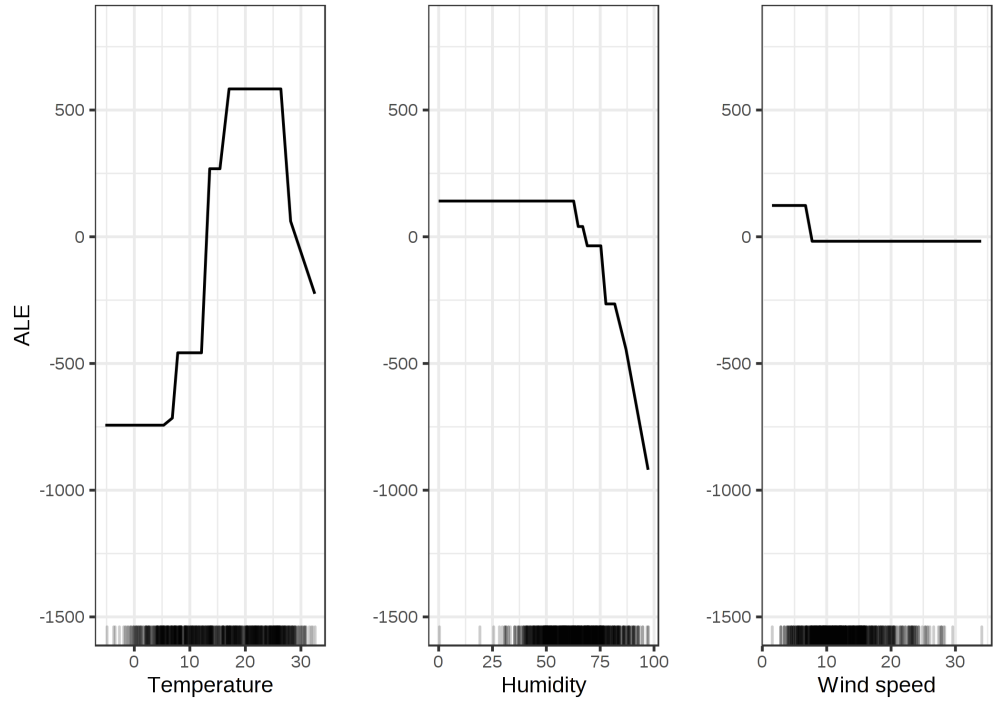


图 5.16. 基于温度，湿度和风速的自行车预测模型的 ALE 图。温度对预测有很强的影响。平均预测随着温度升高而上升，但到 25 摄氏度以上再次下降。湿度具有负效应 (负面影响)：当湿度超过 60% 时，相对湿度越高，预测值越低。风速对预测的影响不大。

让我们看一下温度、湿度和风速与所有其他特征之间的相关性。由于数据还包含分类特征，因此我们不能仅使用 Pearson 相关系数，该系数仅在两个特征均为数值时才有效。相反，我训练了一个线性模型，基于其他特征 (之一) 作为输入来预测温度。然后，我测量线性模型中其他特征所解释的方差并取平方根。如果另一个特征是数值的，则结果等于标准 Pearson 相关系数的绝对值。但是，这种基于模型的 “解释方差” 方法 (也称为 ANOVA，“方差分析”)，即使其他特征是分类特征，仍然有效。“解释方差” 的度量始终位于 0 (无关联) 和 1 (可以从其他特征完美预测温度) 之间。我们计算出所有其他特征的温度，湿度和风速的解释方差。解释方差 (相关性) 越高，PD 图的 (潜在) 问题就越多。下图显示了天气特征与其他特征之间的关联程度。

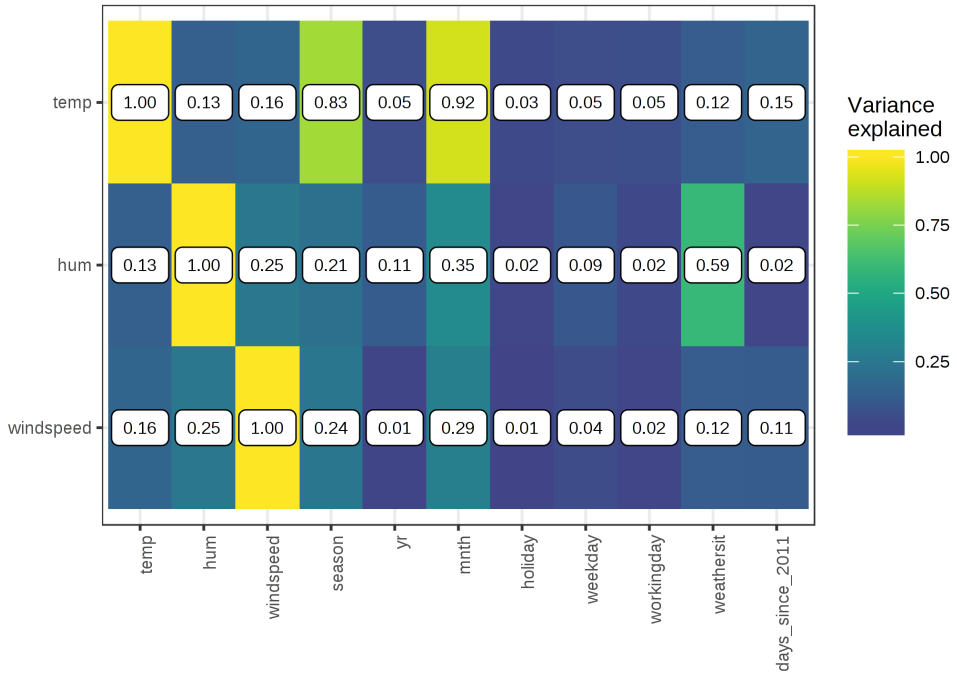


图 5.17. 当我们使用 (例如温度来预测并以季节为特征的) 线性模型进行训练时，温度、湿度和风速与所有特征之间的相关强度以解释方差量表示。对于温度，我们毫不奇怪的观察到与季节和月份高度相关。湿度与天气状况相关。

这种相关性分析表明，我们可能会遇到部分依赖图的问题，尤其是对于温度特征而言。好吧，自己看看：

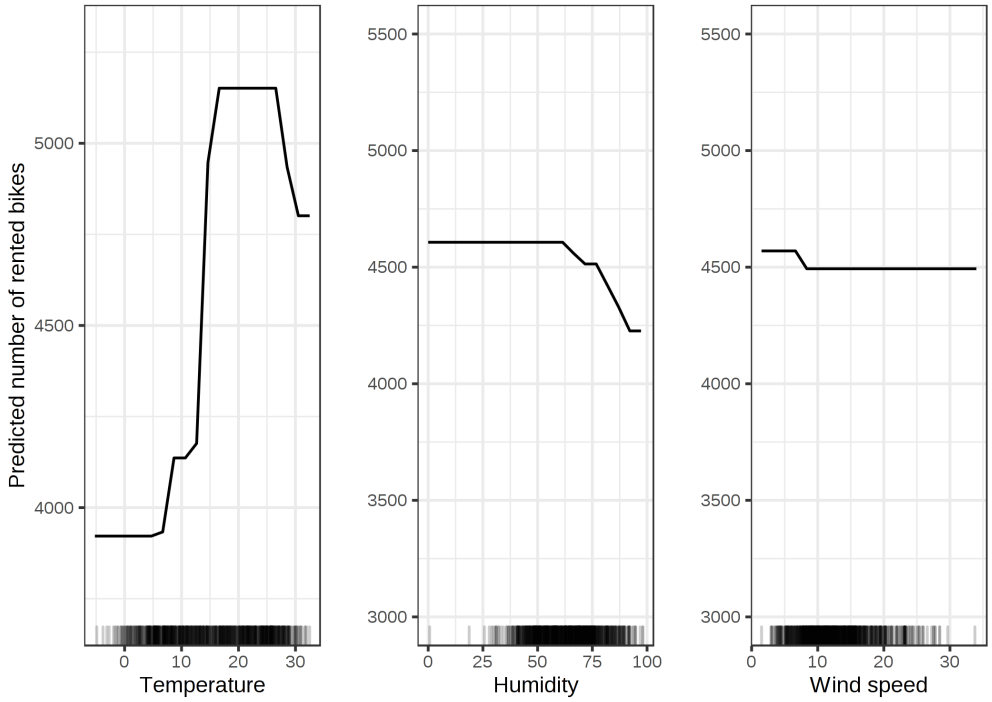


图 5.18. 温度、湿度和风速的 PDP。与 ALE 图相比，PDP 显示在高温或高湿度下的自行车预测数量减少较小。PDP 使用所有数据实例来计算高温的影响，即使它们是 (例如) 季节为 “冬季” 的实例。ALE 图更可靠。

接下来，让我们看到针对分类特征的 ALE 图。月份是一个分类特征，我们要分析它对自行车预测数量的影响。可以说，月份已经有一定的顺序 (从 1 月到 12 月)，但是让我们尝试看看如果我们先按相似性对类别重新排序然后计算影响，会发生什么。根据其他特征 (例如温度或是否假期)，根据每月的相似性对月份进行排序。

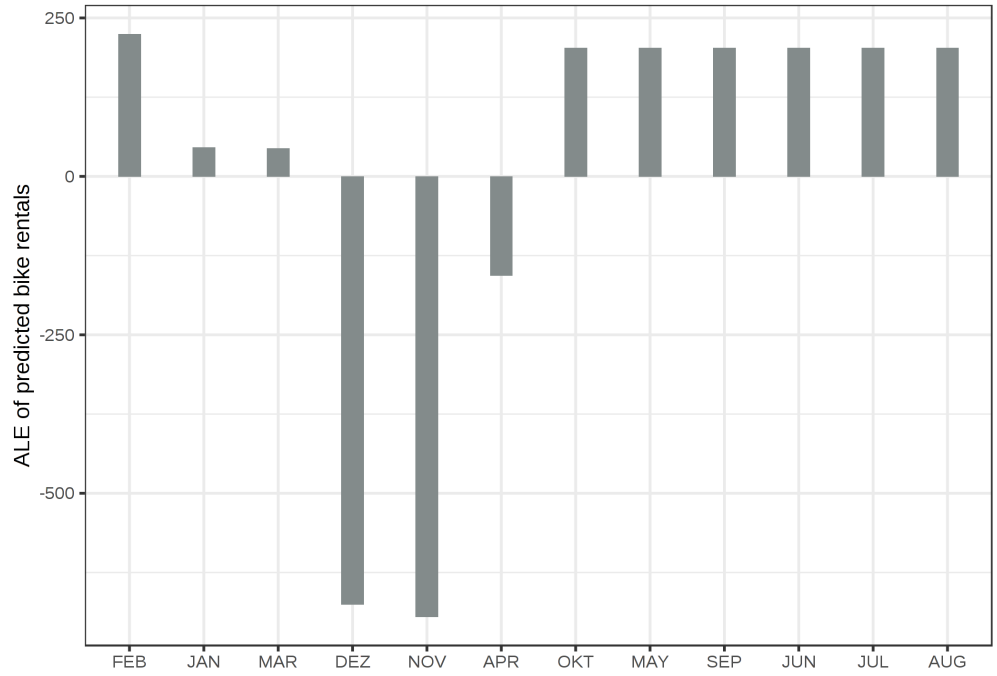


图 5.19. 分类特征月份的 ALE 图。基于 (按月的) 其他特征的分布情况，根据彼此之间的相似性对月份进行排序。与其他月份相比，我们观察到 1 月，3 月和 4 月，尤其是 12 月和 11 月，对预测的租赁自行车数量影响较小。

由于许多特征与天气有关，因此月份的顺序强烈反映了月份之间天气的相似程度。所有较冷的月份在左侧 (2 月至 4 月)，在较暖的月份在右侧 (10 月至 8 月)。请记住，相似性计算中还包括了非天气特征，例如，计算月份之间相似度时假期的相对频率与温度具有相同的权重。

接下来，我们考虑湿度和温度对自行车预测数量的二阶效应。请记住，二阶效应是两个特征的附加交互效应，并不包括主要效应。举例来说，这意味着你不会在二阶 ALE 图中看到高湿度导致平均预测自行车数量减少的主要效应。

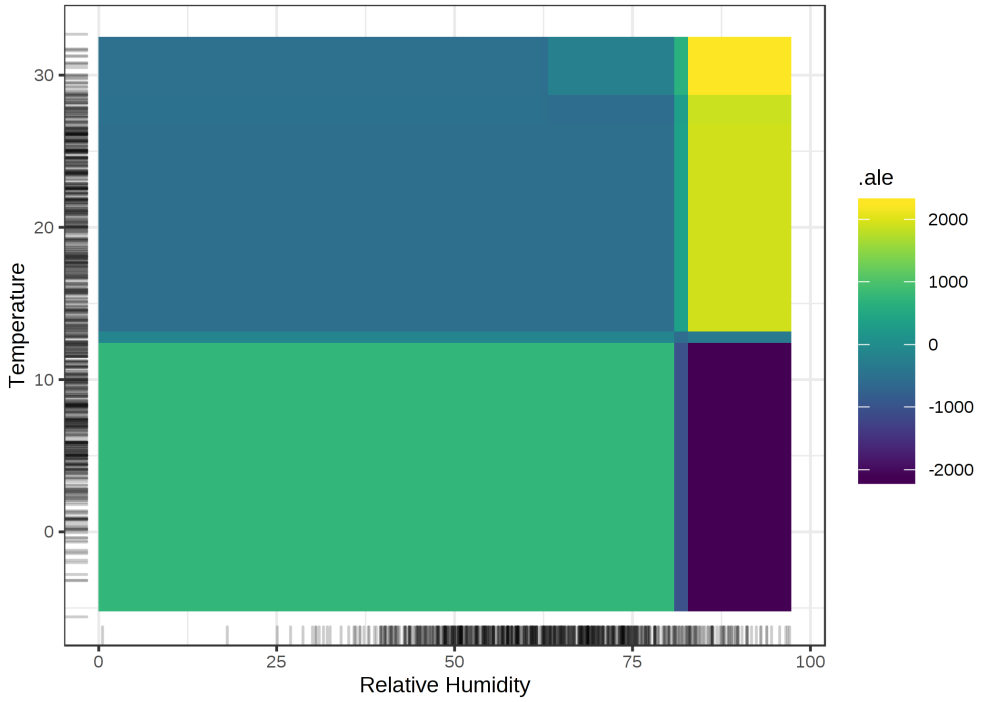


图 5.20. 湿度和温度对所租自行车预测数量的二阶效应的 ALE 图。当已经考虑到主要效应时，浅阴影表示高于平均水平，深阴影表示低于平均水平的预测。该图揭示了温度和湿度之间的交互作用：炎热和潮湿的天气增加了预测。在寒冷和潮湿的天气中，还会对预测的自行车数量产生负效应。

请记住，湿度和温度的主要效应都说明在非常炎热和潮湿的天气中，预测的自行车数量会减少。因此，在炎热和潮湿的天气中，温度和湿度的综合效应不是主要效应的总和，而是大于总和。为了强调纯二阶效应 (你刚刚看到的 2D ALE 图) 和总效应之间的区别，让我们看一下部分依赖图。PDP 显示了总体效应，它结合了平均预测，两个主要效应和二阶效应 (交互作用)。

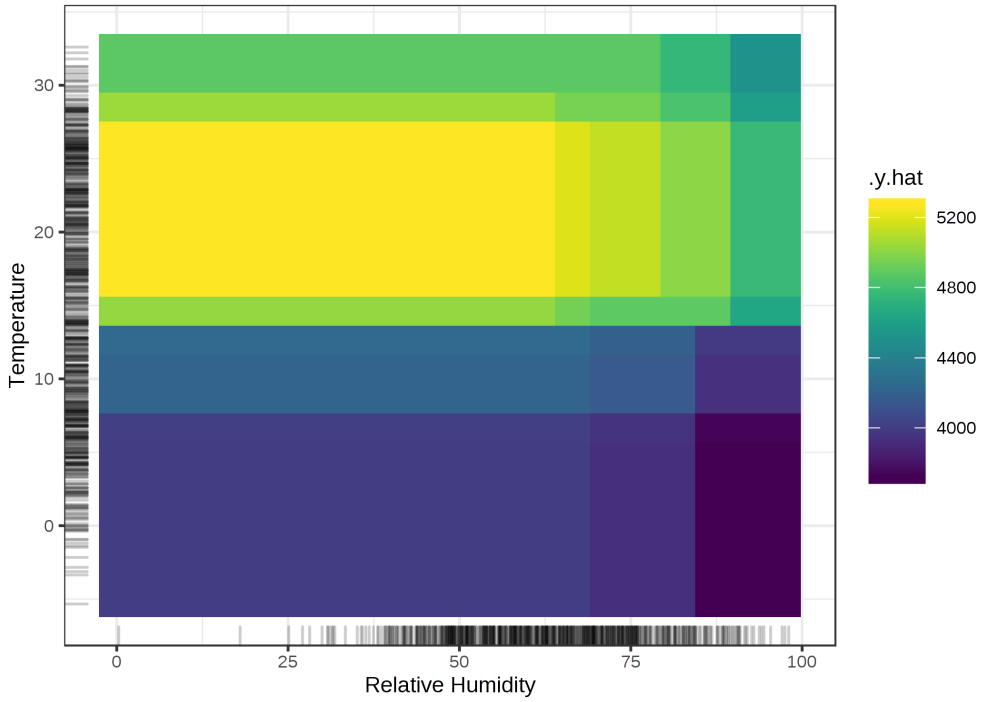


图 5.21. 温度和湿度对自行车预测数量的总体效应的 PDP。与仅显示交互作用的 2D ALE 图相反，该图结合了每个特征的主要效应及其交互效应。

如果你仅对交互感兴趣，你应该考虑二阶效应，因为总效应会将主要效应混合到图中。但是，如果你想了解特征的组合效应，则应查看总体效应 (PDP 显示)。例如，如果你想知道 30 摄氏度和 80% 湿度下的自行车预期数量，则可以直接从 2D PDP 中读取它。如果要从 ALE 图中读取相同内容，则需要查看以下三个图：温度，湿度以及温度 + 湿度的 ALE 图，还需要了解总体平均预测。在两个特征没有交互作用的情况下，这两个特征的总体效应图可能会产生误导，因为它可能显示了复杂的情况，表明存在某些交互作用，但这仅仅是两个主要作用的产物。二阶效应将立即表明没有交互作用。

现在描述自行车够多了，让我们开始分类任务。我们训练一个随机森林以根据风险因素预测宫颈癌的概率。我们将两个特征的累积局部效应可视化：

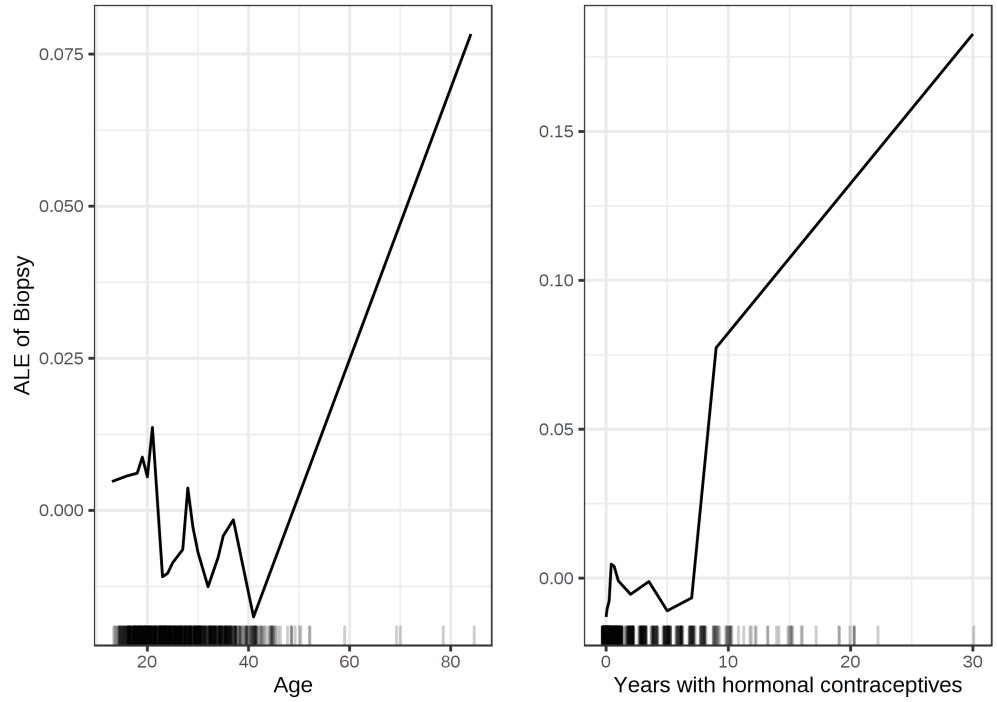


图 5.22. ALE 图表显示年龄和服用激素避孕药的时间对宫颈癌的预测概率的影响。对于年龄特征，ALE 图显示，直到 40 岁，预测的癌症概率平均较低，此后则增加。服用激素避孕药的年数在 8 年后与预测癌症风险高相关。

接下来，我们看一下怀孕次数和年龄之间的交互作用。

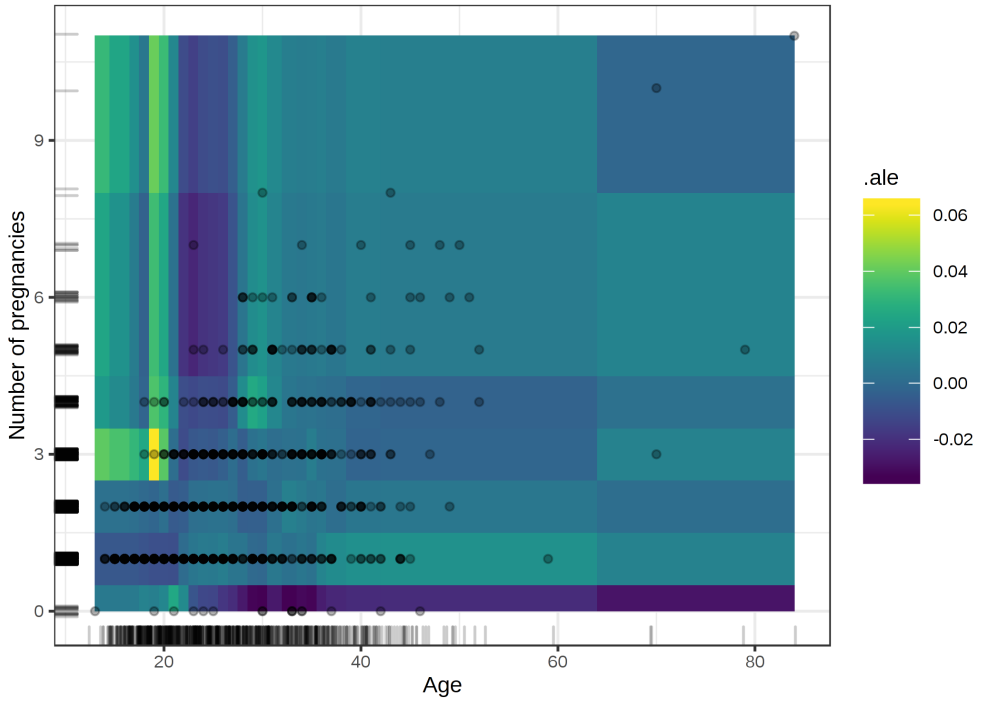


图 5.23.　怀孕次数和年龄的二阶效应的 ALE 图。该图的解释尚无定论，表明似乎过拟合。例如，该图显示了 18-20 岁和 3 次以上的怀孕 (癌症概率增加了 5 个百分点) 的奇怪模型行为。在这组年龄和怀孕次数的数据中，女性并不多 (实际数据显示为点)，因此在训练过程中不会因对这些女性犯错而对该模型进行严厉的惩罚。

5.3.5　优点

ALE 图是无偏的，这意味着在特征相关时它们仍然有效。在这种情况下，部分依赖图将失败，因为它们会边缘化不太可能甚至物理上不可能的特征值组合。

ALE 绘图的**计算速度比 PDP 更快**，并且可以使用 $O(n)$ 进行缩放，因为可能的最大间隔数是实例数 (每个实例为一个间隔)。PDP 需要网格点估计数量的 n 倍。对于 20 个网格点，PDP 所需的预测比最坏情况下的 ALE 图多 20 倍，其中使用的间隔与实例相同。

ALE 图的解释很清楚：在给定值的条件下，可以从 ALE 图中读取更改特征对预测的相对影响。**ALE 图以 0 为中心**。这使它们的解释更好，因为 ALE 曲线每个点的值都是与平均预测之差。**2D ALE 绘图仅显示交互作用**：如果两个特征不交互，则图不显示任何内容。

总而言之，在大多数情况下，我**宁愿使用 ALE 图而不是 PDP 图**，因为特征通常在某种程度上相关。

5.3.6 缺点

ALE 图可能会变得有些不稳定 (许多小起伏)，间隔很多。在这种情况下，减少间隔数可以使估计更稳定，但也可以使预测模型的实际复杂度消除和隐藏。**没有完美的解决方案来设置间隔的数量**。如果数量太小，则 ALE 图可能不太准确。如果数量太大，曲线可能会变得不稳定。

与 PDP 不同，**ALE 图不附带 ICE 曲线**。对于 PDP 来说，ICE 曲线是很好的，因为它们可以揭示特征效应的异质性，这意味着对于数据子集而言，特征的效应看起来有所不同。对于 ALE 图，你只能检查每个间隔实例之间的效应是否不同，但是每个间隔具有不同的实例，因此它与 ICE 曲线不同。

二阶 ALE 估计在整个特征空间中具有不同的稳定性，这是不以任何方式可视化的。其原因是，对一个单元中局部效应的每次估计都使用不同数量的数据实例。结果，所有估计都具有不同的准确性 (但它们仍然是最好的估计)。对于主要效应 ALE 图，该问题存在于不太严重的版本中。由于使用了分位数作为网格，因此在所有间隔中实例的数量都是相同的，但是在某些区域中会存在许多短间隔，并且 ALE 曲线将包含更多估计值。但是对于较长的间隔 (可能占整个曲线的很大一部分) 而言，实例相对较少。例如，这发生在高龄的宫颈癌预测 ALE 图中。

二阶效应图解释起来有点烦人，因为你始终必须牢记主要效应。将热图解读为这两个特征的总体效应是很有诱惑力的，但这里只是附加的交互效应。纯粹的二阶效应对于发现和探索交互非常有趣，但是对于解释效应看起来，我认为将主要效应整合到图中更有意义。

与部分依赖图相比，ALE 图的**实现更加复杂**且不直观。

即使 ALE 图在相关特征的情况下没有偏差，但**当特征强相关时，解释仍然困难**。因为如果它们之间有非常强的相关性，那么分析同时改变两个特征 (而不是孤立地更改) 的效应才有意义。这个缺点不是特定于 ALE 图，而是一个具有强相关特征的普遍问题。

如果特征不相关，计算时间也不成问题，则 PDP 稍好，因为它们易于理解并且可以与 ICE 曲线一起绘制。

缺点列表已经很长了，但是不要被我使用的词的数量所迷惑：作为一个经验法则——使用 ALE 而不是 PDP。

5.3.7 实现和替代方法

我是否提到过部分依赖图和个体条件期望曲线是替代方法？=)

据我所知，ALE 图目前仅在 R 中实现，一个是由发明人本人在 R ALEPlot 包 中，另一个是在 iml 包中。

5.4 特征交互

特征交互 (Feature Interaction)：当特征在预测模型中交互时，预测不能表示为特征效应的总和，因为一个特征的效应取决于另一特征的值。亚里士多德的“整体大于部分之和”适用于存在交互作用的情况。

5.4.1 特征交互

如果机器学习模型基于两个特征进行预测，则可以将预测分解为四项：常量项，第一个特征项，第二个特征项以及两个特征间的交互项。

两个特征间的交互项是在考虑单一特征效应后通过改变特征而发生的预测变化。

例如，模型使用房屋大小 (大或小) 和位置 (好或坏) 作为特征来预测房屋的价格，这产生了四个可能的预测：

Location	Size	Prediction
good	big	300,000
good	small	200,000
bad	big	250,000
bad	small	150,000

我们将模型预测分解为以下几部分：一个常量项 (150,000)，面积大小的特征效应 (如果为大则为 +100,000；如果为小则为 +0) 以及位置特征的特征效应 (如果为好则为 +50,000；如果不好则为 +0)。这种分解完全解释了模型预测。没有交互作用，因为模型预测是针对大小和位置的单一特征效应的总和。当你将小房子变大时，无论位置在哪里，预测值始终会增加 100,000。而且，无论大小，好坏位置的预测差异为 50,000。

现在让我们看一个带有交互的示例：

Location	Size	Prediction
good	big	400,000
good	small	200,000
bad	big	250,000
bad	small	150,000

我们将预测表分解为以下几部分：一个常量项 (150,000)，面积大小特征的效应 (如果为大则为

+100,000；如果为小则为 +0) 和位置特征的效应 (如果为好则为 +50,000；如果为坏则为 +0)。对于此表，我们需要一个用于交互的附加项：如果房屋较大且位置好，则 +100,000。这是大小和位置之间的交互作用，因为在这种情况下，大小房子之间的预测差异取决于位置。

估计交互强度的一种方法是衡量预测的变化在多大程度上取决于特征的交互作用。这项衡量称为 H 统计量，由 Friedman 和 Popescu (2008)[21] 引入。

5.4.2 理论：弗里德曼的 H 统计量

我们将处理两种情况：首先，采用双向交互度量，它告诉我们模型中的两个特征是否交互以及在何种程度上交互；其次，是一个总体交互度量，它告诉我们某个特征在模型中是否与所有其他特征发生交互以及在何种程度上的交互。从理论上讲，可以测量任意数量的特征之间的任意交互，但这两个是最感兴趣的情况。

如果两个特征不交互，我们可以按如下方式分解部分依赖函数 (假设部分依赖函数以零为中心)：

$$PD_{jk}(x_j, x_k) = PD_j(x_j) + PD_k(x_k)$$

其中 $PD_{jk}(x_j, x_k)$ 是两个特征的双向部分依赖函数，而 $PD_j(x_j)$ 和 $PD_k(x_k)$ 是单个特征的部分依赖函数。

同样，如果一个特征与任何其他特征都没有交互，我们可以将预测函数 $\hat{f}(x)$ 表示为部分依赖函数的总和，其中第一个求和项仅依赖于 j，第二个求和项依赖于除 j 以外的所有其他特征：

$$\hat{f}(x) = PD_j(x_j) + PD_{-j}(x_{-j})$$

其中 $PD_{-j}(x_{-j})$ 是依赖于除第 j 个特征以外的所有特征的部分依赖函数。

这种分解表示部分依赖 (或完全预测) 函数，而没有交互作用 (在特征 j 和 k 之间，或者分别在 j 与所有其他特征之间)。在下一步中，我们测量观察到的部分依赖函数与没有交互作用的已分解部分依赖函数之间的差异。我们计算部分依赖 (以度量两个特征之间的交互作用) 或整个函数 (以度量一个特征与所有其他特征之间的交互作用) 输出的方差。由交互作用解释的方差量 (观察到的和没有交互作用 PD 之间的差异) 用作交互强度统计量。如果完全没有交互，则统计量为 0；如果用部分依赖函数之和解释了 PD_{jk} 或 $\hat{f}$ 的所有方差，则统计值为 1。两个特征之间的交互统计为 1 表示每个 PD 函数都是常数，并且对预测的效应仅来自交互。

在数学上，Friedman 和 Popescu 为特征 j 和 k 之间的交互作用提出的 H 统计量为：

$$H_{jk}^2 = \sum_{i=1}^{n} \left[PD_{jk}(x_j^{(i)}, x_k^{(i)}) - PD_j(x_j^{(i)}) - PD_k(x_k^{(i)}) \right]^2 / \sum_{i=1}^{n} PD_{jk}^2(x_j^{(i)}, x_k^{(i)})$$

这同样适用于度量特征 j 是否与任何其他特征交互：

$$H_j^2 = \sum_{i=1}^{n} \left[\hat{f}(x^{(i)}) - PD_j(x_j^{(i)}) - PD_{-j}(x_{-j}^{(i)}) \right]^2 / \sum_{i=1}^{n} \hat{f}^2(x^{(i)})$$

H 统计量的评估成本很高，因为它会在所有数据点上进行迭代，并且必须在每个点处评估部分依赖，而这又需要对所有 n 个数据点进行。在最坏的情况下，我们需要 $2n^2$ 调用机器学习模型预测函数来计算双向 H 统计量 (j vs. k)，并需要 $3n^2$ 以获得总体 H 统计量 (j vs. all)。为了加快计算速度，我们可以从 n 个数据点进行采样。这有增加部分依赖估计方差的缺点，使得 H 统计量不稳定。因此，如果使用采样来减少计算负担，请确保采样足够的数据点。

Friedman 和 Popescu 还提出了一个检验统计量，以评估 H 统计量与零是否有显著差异。零假设是没有交互。要在零假设下生成交互统计量，你必须能够调整模型，以使其在特征 j 和 k 或所有其他特征之间不具有交互作用。这不可能适用于所有类型的模型。因此，该检验是特定于模型的，而不是与模型无关的，故不在此介绍。

如果预测是概率，则交互强度统计也可以应用于分类设置。

5.4.3 示例

让我们看看在实践中交互特征是什么样的！我们在支持向量机中测量特征的交互强度，支持向量机可根据天气和日历特征预测租赁自行车的数量。下图显示了特征交互作用 H 统计量：

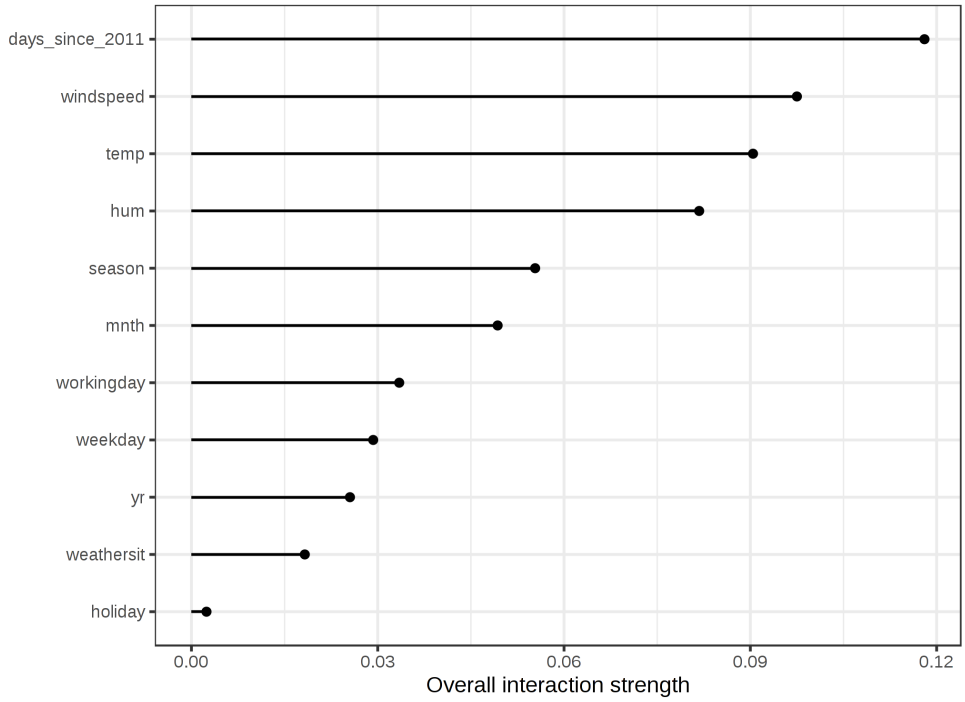


图 5.24. 预测自行车租赁的支持向量机的每个特征与所有其他特征的交互强度 (H 统计量)。总体而言，特征之间的交互作用非常弱 (低于每个特征解释的方差的 10%)。

在下一个示例中，我们计算分类问题的交互统计量。给定一些风险因素，我们分析了经过训练可预测宫颈癌的随机森林中特征之间的交互作用。

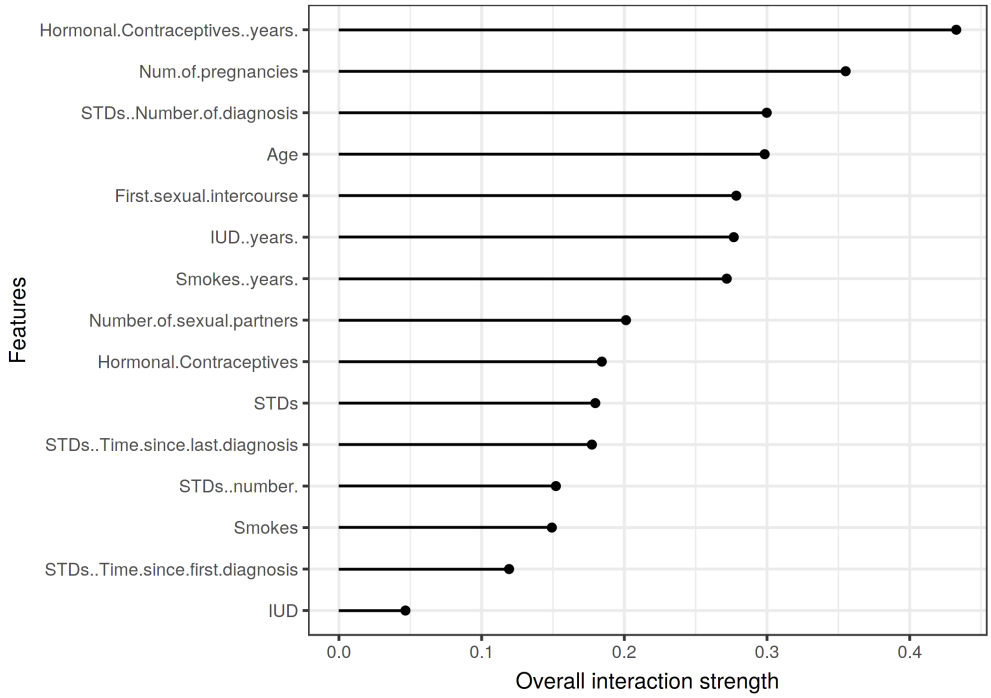


图 5.25. 用于预测宫颈癌概率的随机森林的每个特征与所有其他特征的交互强度 (H 统计量)。激素避孕药的使用年限与其他特征的相对最高的交互作用，其次是怀孕次数。

在查看了每个特征与所有其他特征的特征交互之后，我们可以选择其中一个特征，然后更深入地研究所选特征与其他特征之间的所有双向交互。

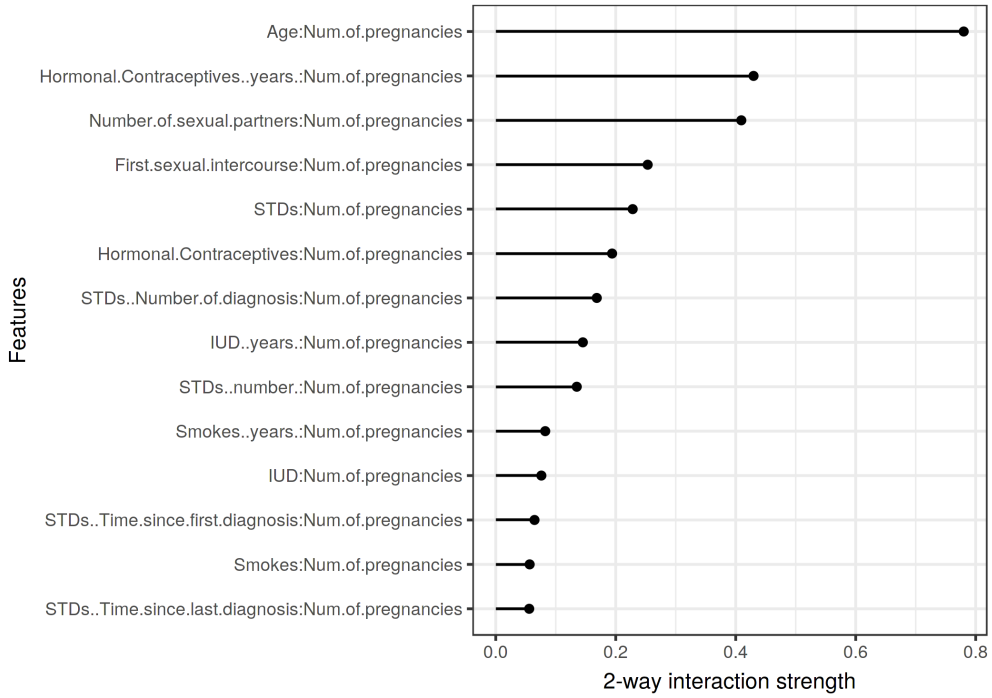


图 5.26. 怀孕次数与其他特征之间的双向交互强度 (H 统计量)。怀孕次数和年龄之间有很强的交互作用。

5.4.4 优点

交互作用 H 统计量通过部分依赖分解具有**理论基础**。

H 统计量具有**有意义的解释**：交互作用定义为由交互作用解释的方差份额。

由于统计信息是**无量纲的**，并且总是在 0 和 1 之间，因此它在各个特征甚至模型之间都具有可比性。

统计信息会**检测各种类型的交互**，无论它们的特殊形式如何。

使用 H 统计量，还可以分析**任意更高阶的交互作用**，例如 3 个或更多特征之间的交互作用强度。

5.4.5 缺点

你会注意到的第一件事：交互 H 统计量需要花费很长时间进行计算，因为它的**计算量很大**。

该计算涉及估计边际分布。如果我们不使用所有数据点，则这些估计值**也存在一定差异**。这意味着，当我们对点进行采样时，估算值也因运行而异，结果可能会**不稳定**。我建议重复几次 H 统计量计算，以查看是否有足够的数据来获得稳定的结果。

尚不清楚交互作用是否显着大于 0。我们将需要进行统计检验，但**该检验在与模型无关的版本中尚不可用**。

关于检验问题，很难说 H 统计量何时足够大以至于我们认为交互 “强”。

另外，H 统计量可能大于 1，这使得解释变得困难。

H 统计量告诉我们交互的强度，但没有告诉我们交互的样子。这就是部分依赖图的用途。一个有意义的工作流程是测量交互强度，然后为你感兴趣的交互创建 2D 部分依赖图。

如果输入是像素，则无法有效使用 H 统计量。因此该技术对图像分类器没有用。

交互统计是在假设我们可以独立地对特征进行随机排序的情况下起作用。如果特征之间具有很强的相关性，则将违反该假设，并且我们**会结合现实中极不可能发生的特征组合**。这就是部分依赖图存在的相同问题。你通常无法说出它是否导致高估或低估。

有时结果是奇怪的，对于小的模拟**不能产生预期的结果**。但这更多是个奇闻轶事。

5.4.6 实现

对于本书中的示例，我使用了 R iml 包，该包在 CRAN 上可用，而开发版本在 Github 上可用。还有其他一些实现，这些实现着眼于特定模型：R 预先 实现 RuleFit 和 H-statistic。R gbm 包实现了梯度提升模型和 H 统计量。

5.4.7 替代方法

H 统计量不是衡量交互作用的唯一方法：

Hooker (2004)[28] 提出的变量交互网络 (VIN) 是一种将预测函数分解为主要效应和特征交互的方法。然后将特征之间的交互可视化为网络。不幸的是，目前还没有可用的软件。

Greenwell 等人 (2018)[29] 基于部分依赖性的特征交互测量了两个特征之间的交互。这种方法测量一个特征在另一个特征的不同固定点上的特征重要性 (定义为部分依赖函数的方差)。如果方差高，则特征就交互；如果方差为零，则它们就不交互。相应的 R vip 包在 Github 上可以获得。该软件包还涵盖了部分依赖图和特征重要性。

5.5 置换特征重要性

置换特征重要性 (Permutation Feature Importance) 衡量了我们对特征值进行置换后模型预测误差的增加，这打破了特征与真实结果之间的关系。

5.5.1 理论

这个概念非常简单：我们通过置换特征后计算模型的预测误差的增加来衡量特征的重要性。如果将特征的值改变会增加模型误差，则该特征 “重要”，因为在这种情况下，模型依赖于特征进行预测。如果将特征的值改变而使模型误差保持不变，则特征 “不重要”，因为在这种情况下，模型会忽略预测的特征。置换特征重要性度量是由 Breiman (2001)[30] 引入的，用于随机森林。基于这个想法，Fisher、Rudin 和 Dominici (2018)[31] 提出了特征重要性的模型无关版本，并将其称为模型依赖 (Model Reliance)。他们还介绍了有关特征重要性的更先进的想法，例如一个 (特定于模型的) 版本是考虑到用很多预测模型可以很好地预测数据。他们的论文值得一读。

基于 Fisher，Rudin 和 Dominici (2018) 的置换特征重要性算法：

置换特征重要性 训练模型 f，特征矩阵 X，目标向量 y，误差度量 $L(y,f)$ 估计原始模型误差 $e^{orig} = L(y,f(X))$ (例如均方误差) 特征 $j \leftarrow 1$ to p 通过置换数据 X 中的特征 j 生成特征矩阵 X^{perm}。这破坏了特征 j 与真实结果 y 之间的关联基于置换数据的预测，估计误差 $e^{perm} = L(Y,f(X^{perm})$ 计算置换特征重要性 $FI^j = e^{perm}/e^{orig}$。或者，可以使用差异：$FI^j = e^{perm} - e^{orig}$ 按降序对特征进行排序

Fisher，Rudin 和 Dominici (2018) 在他们的论文中建议将数据集分成两半并交换这两半的特征 j 的值，而不是置换特征 j。仔细考虑一下，这与置换特征 j 完全相同。如果需要更准确的估计，可以通过将每个实例的特征 j 值与每个其他实例的特征 j 值配对 (自身除外) 来估计置换特征 j 的误差。这为你提供了一个大小为 n(n-1) 足以估计置换误差的数据集，并且需要大量的计算时间。如果你真的想得到非常准确的估计，我才建议使用 n(n-1)-方法。

5.5.2 我应该计算训练数据还是测试数据的重要性?

这我没有确切的答案。

回答有关训练或测试数据的问题，触及到特征重要性的基本问题。理解基于训练数据的特征重要性与基于测试数据的特征重要性之间的区别的最好方法是一个 “极端” 示例。我训练了一个支持向量机，给定 50 个随机特征 (200 个实例) 预测连续的、随机的目标结果。“随机” 是指目标结果独立于 50 个特征。这就像根据最新的彩票号码预测明天的气温一样。如果模型 “学习” 了所有的关系，则表示过拟合。实际上，SVM 确实对训练数据过拟合。训练数据的平均绝对误差 (简称：MAE) 为

0.29，测试数据的平均绝对误差为 0.82，这也是最可能的模型的误差 (预测平均结果总为 0 时 MAE 为 0.78)。换句话说，SVM 模型是垃圾。对于此过拟合的 SVM 的 50 个特征，你期望特征重要性是什么值？是零？因为没有任何特征有助于提高看不见的测试数据的性能；还是重要性应该反映模型在多大程度上取决于每个特征，而不管所学的关系是否泛化到看不见的数据？让我们看一下训练和测试数据的特征重要性分布如何不同。

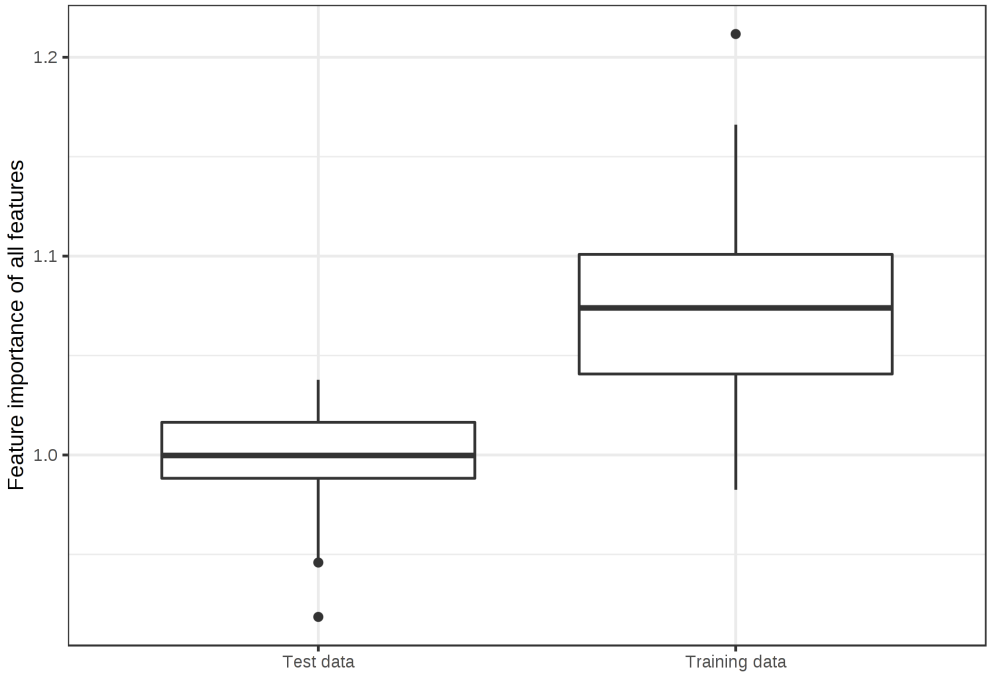


图 5.27. 按数据类型划分的特征重要性值的分布。对一个具有 50 个随机特征和 200 个实例的回归数据集进行了 SVM 训练。支持向量机对数据过拟合：基于训练数据的特征重要性显示出许多重要的特征。根据未公开的测试数据计算，特征重要性接近 1 的比率 (= 不重要)。

我不清楚这两个结果中哪个更可取。因此，我将尝试为这两个版本各做说明，让你自己决定。

测试数据情况

这是一个简单的情况：基于训练数据的模型误差估计是垃圾-> 特征重要性依赖于模型误差估计-> 基于训练数据的特征重要性是垃圾。

实际上，这是你在机器学习中学习的第一件事：如果在训练模型的相同数据上测量模型误差 (或性能)，则测量通常过于乐观，这意味着模型的工作效果似乎比实际情况要好得多。并且由于置换特征重要性取决于模型误差的度量，因此我们应该使用看不见的测试数据。基于训练数据的特征重要性使我们错误地认为特征对于预测很重要，而实际上模型只是过拟合而特征根本不重要。

训练数据情况

使用训练数据的论据在某种程度上更难以表述，但恕我直言，就像使用测试数据的论据一样引人注目。我们再来看一下垃圾的 SVM。根据训练数据，最重要的特征是 $X42$。让我们看一下特征 $X42$ 的部分依赖图。部分依赖图显示了模型输出如何根据特征的变化而变化，并且不依赖于泛化误差。PDP 是用训练数据还是测试数据计算都没有关系。

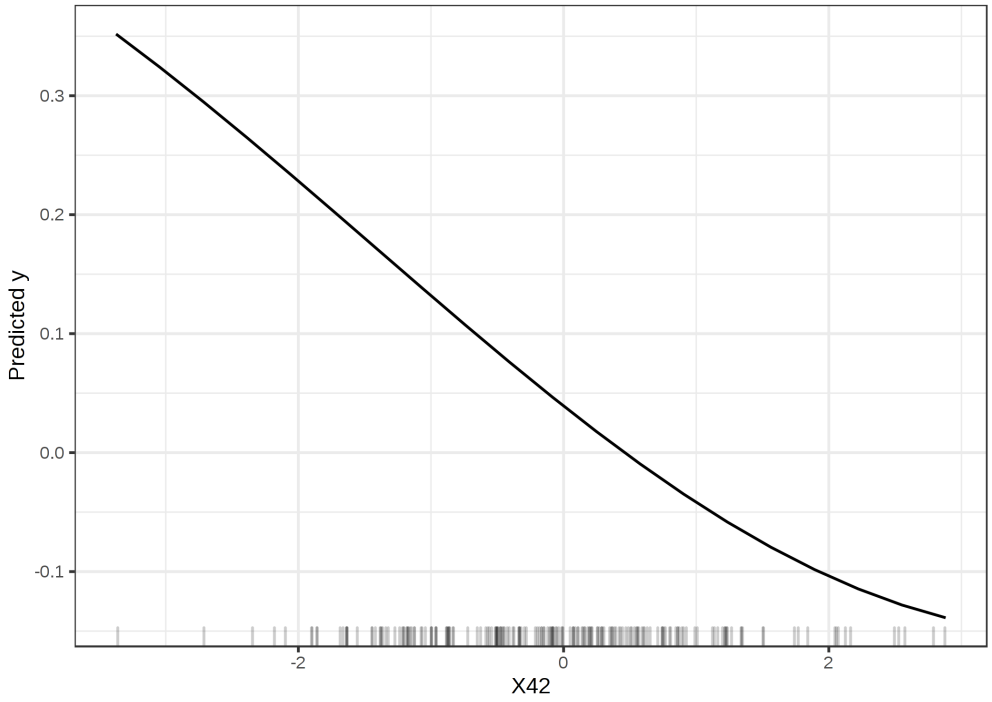


图 5.28. $X42$ 特征的 PDP，基于训练数据并根据特征重要性，它是最重要的特征。该图显示了 SVM 如何依赖此特征进行预测。

该图清楚地表明，SVM 已经学会了依靠特征 $X42$ 进行预测，但是根据基于测试数据的特征重要性 (1.04)，它并不重要。根据训练数据，重要性为 1.21，反映出该模型已学会使用此特征。基于训练数据的特征重要性可以告诉我们哪些特征对模型很重要，模型的预测依赖于这些特征。

作为使用训练数据情况的一部分，我想提出一个反对测试数据的论点。实际上，你希望使用所有数据来训练模型，以最终获得最可能的模型。这意味着将没有剩余未使用的测试数据来计算特征的重要性。当你要估计模型的泛化误差时，也会遇到相同的问题。如果将 (嵌套的) 交叉验证用于特征重要性估计，则会遇到以下问题：特征重要性不是在具有所有数据的最终模型上计算的，而是在具有可能表现不同的数据子集的模型上计算的。

最后，你需要决定是要知道模型在多大程度上依赖于每个特征来进行预测 (-> 训练数据)，还是该特征在多大程度上有助于模型在未知数据上的性能 (-> 测试数据)。据我所知，目前还没有针对训练数据与测试数据的研究。与我的 “垃圾 SVM” 示例相比，它将需要更深入的检查。我们需要更多的研究和使用这些工具的经验来获得更好的理解。

接下来，我们将看一些示例。我将重要性计算基于训练数据，因为我必须选择一个，并且使用训练数据可以少几行代码。

5.5.3 示例和解释

我展示了分类和回归的示例。

宫颈癌 (分类)

我们拟合随机森林模型预测宫颈癌。我们用 1-AUC (1 减去 ROC 曲线下的面积) 测量误差增加量。将模型误差增加为 1 倍 (= 无变化) 的特征表示对于宫颈癌的预测并不重要。

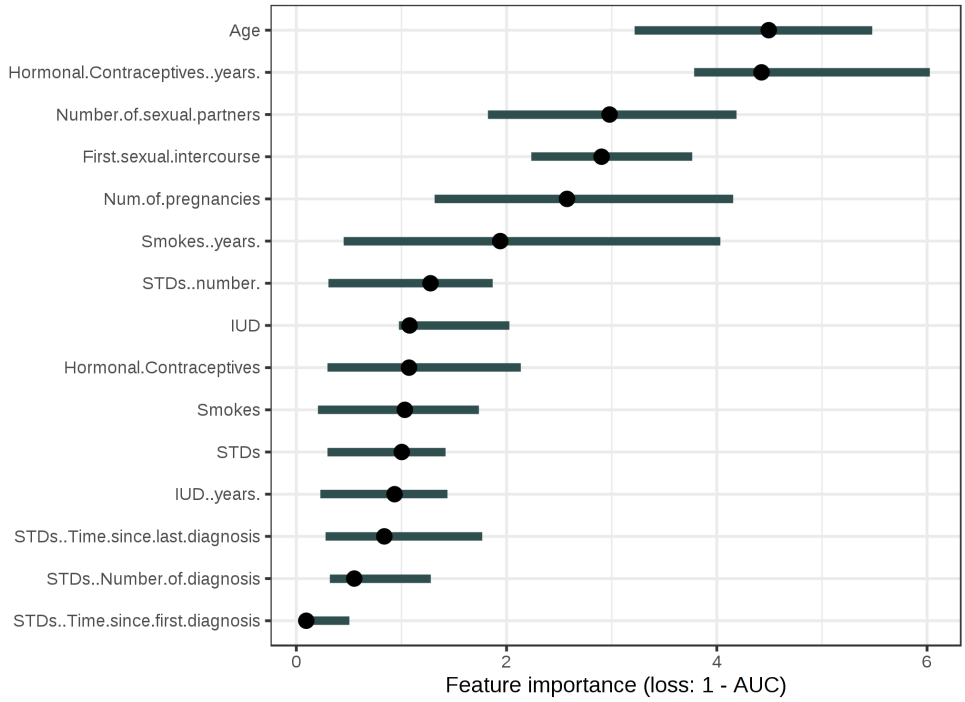


图 5.29. 随机森林预测宫颈癌各特征的重要性。最重要的特征是年龄。置换年龄导致 1-AUC 增加 4.49 倍。

最重要的特征是年龄，置换后误差增加 4.49。

租赁自行车 (回归)

我们拟合了一个支持向量机模型，在给定天气条件和日历信息的情况下预测租赁自行车的数量。作为误差测量，我们使用平均绝对误差。

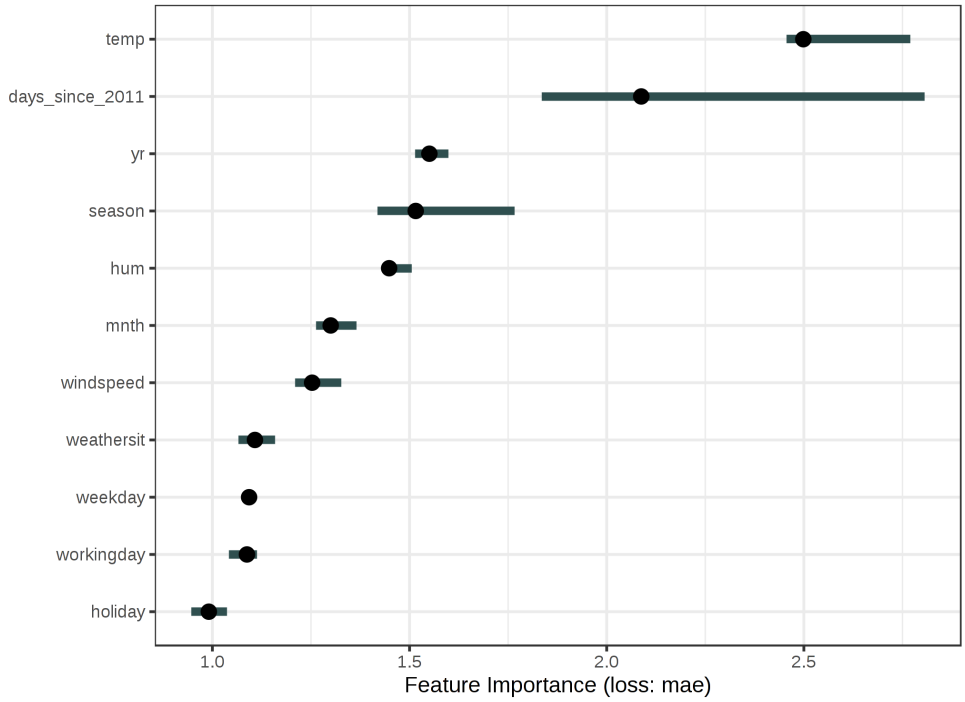


图 5.30. 使用支持向量机预测自行车计数时每个特征的重要性。最重要的是温度，最不重要的是假期。

5.5.4 优点

很好的解释：特征重要性是当特征信息被破坏时模型误差的增加。

特征重要性提供了对模型行为的**高度压缩的、全局的洞悉**。

使用错误的比率代替错误的差值的一个积极方面是，在不同问题之间，特征重要性度量是**可比较的**。

重要性度量会自动考虑与其他特征的**所有交互**。通过置换特征，你还可以破坏与其他特征的交互效应。这意味着置换特征重要性同时考虑了主要特征效应和交互效应。这也是一个缺点，因为两个特征之间的交互作用的重要性包括在两个特征的重要性测量中。这意味着特征重要性不是加起来就是总性能的下降，而是总和更大。仅当特征之间没有交互时 (如线性模型中)，重要性才近似相加。

置换特征重要性**不需要重新训练模型**。其他一些方法建议删除特征，重新训练模型，然后比较模型误差。由于机器学习模型的重新训练可能需要很长时间，因此“只”置换特征就可以节省大量时间。乍一看，使用部分特征重新训练模型的重要性方法乍看起来似乎很直观，但是数据减少后的模型对于特征的重要性却毫无意义。我们对固定模型的特征重要性感兴趣。使用减少的数据集进行重新训练会创建与我们感兴趣的模型不同的模型。假设你训练了一个稀疏线性模型 (使用 Lasso)，该模型具有固定数量的权重且权重非零。数据集具有 100 个特征，你可以将非零权重的数量设置为 5。你

可以分析其中一个非零权重的特征的重要性。你删除特征并重新训练模型。模型的性能保持不变，因为另一个同样出色的特征的权重不为零，你的结论是该特征并不重要。另一个例子：模型是决策树，我们分析了被选为第一个分割的特征的重要性。你删除特征并重新训练模型。由于选择了另一个特征作为第一个分割，因此整棵树可能会非常不同，这意味着我们比较 (可能) 完全不同的树的错误率，以确定该特征对其中一棵树的重要性。

5.5.5 缺点

非常不清楚应该使用训练数据还是测试数据来计算特征的重要性。

置换特征的重要性**与模型的误差有关**。这并不是天生的坏事，但在某些情况下不是你所需要的。在某些情况下，你可能希望了解某个特征的模型输出有多少变化，而无需考虑其对性能的影响。例如，你想了解当有人操纵特征时模型输出的鲁棒性。在这种情况下，你不会对置换特征时模型性能降低多少感兴趣，而是由每个特征解释的模型输出的方差是多少。当模型泛化得很好时 (即，它不会过拟合)，模型方差 (由特征解释) 和特征重要性密切相关。

你**需要获得真正的结果**。如果某人仅向你提供模型和未标记的数据，但没有提供真实结果，则你将无法计算置换特征的重要性。

置换特征的重要性取决于对特征的改变，这会增加测量的随机性。重复置换后，**结果可能会有很大差异**。重复置换并在重复中对重要性值进行平均，可以鲁棒地度量结果，但会增加计算时间。

如果特征是相关的，则置换特征重要性**可能会因不切实际的数据实例而有偏差**。问题与部分依赖图相同：特征的置换在两个或多个特征关联时不会产生数据实例。当它们正相关时 (例如一个人的身高和体重)，并且我改变了其中一个特征时，我创建了不太可能甚至是物理上不可能的新实例 (例如，体重 2 公斤的 2 米体重的人)，但是我使用了这些新实例衡量重要性。换句话说，对于相关特征的置换特征重要性，我们考虑了当我们将特征与在现实中永远不会观察到的值交换时，模型性能会降低多少。检查特征是否紧密相关，如果是，对特征重要性的解释要小心。

另一个棘手的事情：**添加相关特征可以通过在两个特征之间拆分重要性降低关联特征的重要性**。让我举一个例子说明我所说的 “拆分” 特征重要性的含义：我们要预测下雨的概率，并将前一天上午 8 点的温度用作特征，以及其他不相关的特征。我训练了一个随机森林，结果发现温度是最重要的特征，一切都很好，第二天晚上我睡得很好。现在想象另一个场景，其中我另外包括了上午 9 点的温度，它作为与上午 8 点的温度强相关的特征。如果我已经知道上午 8 点的温度，那么上午 9 点的温度不会给我太多其他信息。但是拥有更多特征总是好的，对吧？我用两个温度特征和不相关特征训练了另一个随机森林。随机森林中的一些树基于上午 8 点的温度，一些树基于上午 9 点的温度，一些则是同时拥有，还有一些则都没有。这两个温度特征一起比以前的单个温度特征具有更多的重要性，但是每个温度现在不在重要特征列表的顶部，而是在中间。通过引入相关特征，我将最重要的特征从重要性阶梯的顶端踢到了中间。一方面，这很好，因为它仅反映了底层机器学习模型 (这

里是随机森林) 的行为。上午 8 点的温度已经变得不那么重要了，因为该模型现在也可以依靠上午 9 点的测量值。另一方面，这使得特征重要性的解释变得相当困难。假设你要检查特征是否存在测量错误。检查费用昂贵，你决定只检查最重要特征中的前 3 个。在第一种情况下，你将检查温度，在第二种情况下，你将不包括任何温度特征，因为它们现在具有同等的重要性。即使重要性值在模型行为的层次上可能有意义，但如果你具有相关特征，则仍会造成混淆。

5.5.6 软件和替代方法

R iml 包被用于示例。在 R DALEX 和 vip 包，以及 Python alibi 包也实现了模型无关置换特征重要性。

一个名为 PIMP 的算法对特征重要性算法进行了改进，为重要性提供 p 值。

5.6 全局代理模型

全局代理模型是一种可解释的模型，经过训练可近似于黑盒模型的预测。我们可以通过解释代理模型得出有关黑盒模型的结论。通过使用更多的机器学习解决机器学习的可解释性!

5.6.1 理论

代理模型也可用于工程中：如果目标结果昂贵、耗时或难以衡量 (例如，由于来自复杂的计算机模拟)，则可以使用结果廉价、快速的代理模型。工程中使用的代理模型和可解释机器学习中使用的代理模型的区别在于，底层模型是机器学习模型 (不是模拟)，并且代理模型必须是可解释的。(可解释的) 代理模型的目的是尽可能准确地近似底层模型的预测，并且可以同时进行解释。可以用不同的名称找到代理模型的概念：近似模型 (`Approximation Model`)，元模型 (`metamodel`)，响应面模型 (`Response Surface Model`)，仿真器 (`emulator`) 等等。

关于这个理论：理解代理模型实际上并不需要太多理论。我们希望在 g 可解释的约束下，代理模型预测函数 g 尽可能接近地逼近我们的黑盒预测函数 f。对于函数 g，可以使用任何可解释的模型，例如来自 “可解释的模型” 一章的模型。

例如线性模型：

$$g(x) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

或决策树：

$$g(x) = \sum_{m=1}^{M} c_m I\{x \in R_m\}$$

训练代理模型是一种与模型无关的方法，因为它不需要有关黑盒模型内部运作的任何信息，仅需要访问数据和预测。如果将底层机器学习模型替换为其他机器学习模型，你仍可以使用代理方法。黑盒模型类型和代理模型类型的选择是分离的。

执行以下步骤以获得代理模型：

1. 选择数据集 X。这可以是用于训练黑盒模型的相同数据集，也可以是来自同一分布的新数据集。你甚至可以根据应用程序选择数据的子集或点的网格。
2. 对于选定的数据集 X，获取黑盒模型的预测。
3. 选择一种可解释的模型类型 (线性模型，决策树等)。
4. 在数据集 X 及其预测上训练可解释模型。
5. 恭喜你！你现在有了一个代理模型。
6. 衡量代理模型复制黑盒模型预测的效果。
7. 解释代理模型。

你可能会发现代理模型的方法有一些额外的步骤或稍有不同，但一般的想法通常如此处所述。

衡量代理复制黑盒模型的能力的一种方法是 R-平方度量：

$$R^2 = 1 - \frac{SSE}{SST} = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_*^{(i)} - \hat{y}^{(i)})^2}{\sum_{i=1}^{n}(\hat{y}^{(i)} - \bar{\hat{y}})^2}$$

其中 $\hat{y}_*^{(i)}$ 是代理模型的第 i 个实例的预测，$\hat{y}^{(i)}$ 是黑盒模型的预测和 $\bar{\hat{y}}$ 是黑盒模型预测的平均值。SSE 代表误差平方和，SST 代表总的平方和。R-平方可以解释为代理模型捕获的方差百分比。如果 R-平方接近 1 (= 低 SSE)，则可解释模型会很好地近似黑盒模型的行为。如果可解释模型非常接近，则可能要用可解释模型替换复杂模型。如果 R 平方接近 0 (= 高 SSE)，则可解释模型无法解释黑盒模型。

请注意，我们没有谈论底层黑盒模型的模型性能，也就是说，它在预测实际结果方面的好坏。黑盒模型的性能在训练代理模型中不起作用。代理模型的解释仍然有效，因为它是关于模型的陈述，而不是关于现实世界的陈述。但是，当然，如果黑盒模型不好，则代理模型的解释就变得无关紧要了，因为黑盒模型本身就是无关紧要的。

我们还可以基于原始数据的子集构建代理模型，或重新调整实例的权重。这样，我们更改了代理模型输入的分布，从而改变了解释的重点 (然后它不再是真正的全局性的)。如果我们通过数据的特定实例在局部对数据进行加权 (数据与所选的特定实例越近，则它们的权重就越高)，我们将获得一个局部代理模型，该模型可以解释该实例的各个预测。在下一节中阅读有关局部模型的更多信息。

5.6.2 示例

为了演示代理模型，我们考虑一个回归和一个分类示例。

首先，我们训练一个支持向量机，根据天气和日历信息预测自行车的租量。支持向量机不是很容易解释，因此我们将 CART 决策树作为可解释模型训练代理，以近似支持向量机的行为。

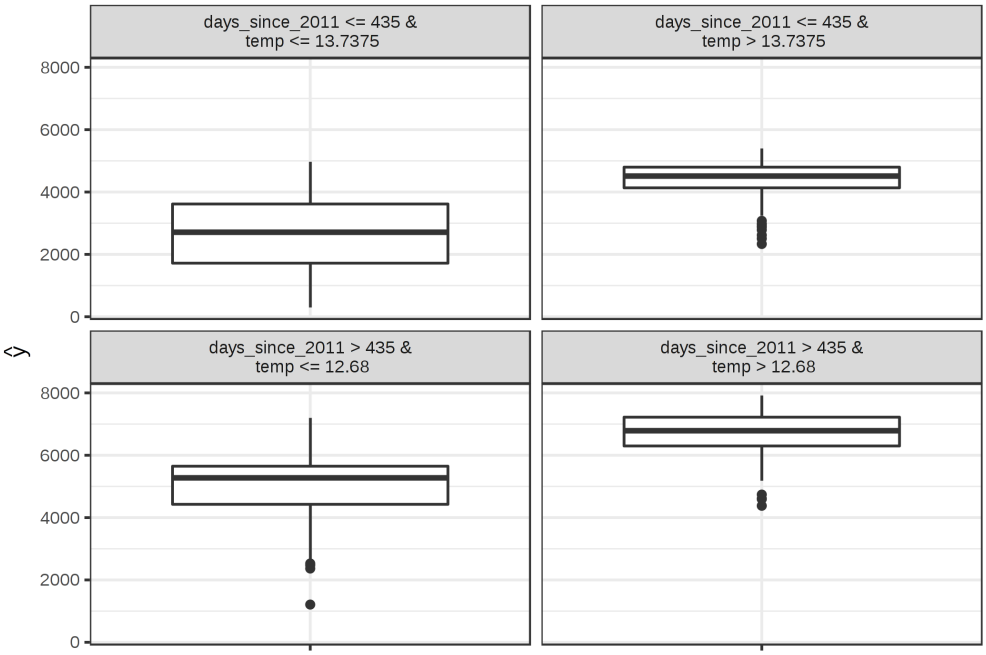


图 5.31. 代理树的终端节点，它近似自行车租赁数据集上训练的支持向量机的预测。节点中的分布表明，当温度高于 13 摄氏度时，以及天数是 2 年的晚些时候 (临界点为 435 天)，代理树预测租用自行车的数量会更高。

代理模型的 R-平方 (方差解释) 为 0.77，这意味着它很好地近似了底层黑盒的行为，但并不完美。如果拟合是完美的，我们可以扔掉支持向量机，而改用树。

在第二个示例中，我们拟合随机森林模型预测了宫颈癌的概率。再次，我们使用原始数据集训练决策树，但将随机森林的预测作为结果，而不是数据中的真实类别 (健康、癌症)。

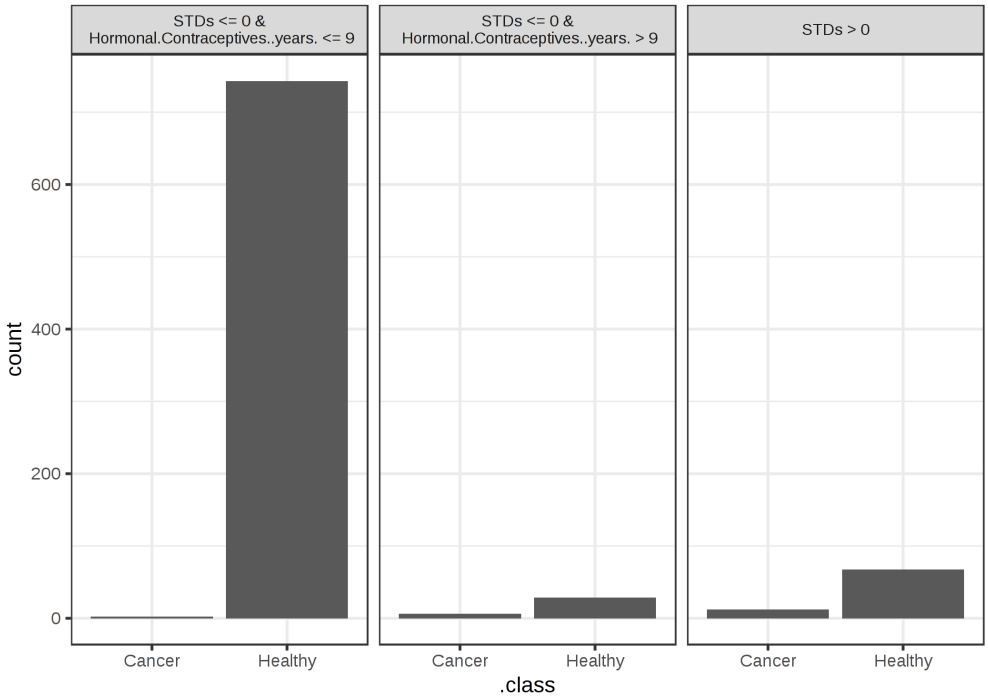


图 5.32. 代理树的终端节点，它近似于一个随机森林对宫颈癌数据集的预测。节点中的计数显示节点中黑盒模型分类的频率。

代理模型的 R-平方 (方差已解释) 为 0.19，这意味着它不能很好地逼近随机森林，并且在得出有关复杂模型的结论时，我们不应过度解释树。

5.6.3 优点

代理模型方法**非常灵活**：可以使用 “可解释的模型” 一章中的任何模型。这也意味着你不仅可以交换可解释模型，还可以交换底层黑盒模型。假设你创建了一个复杂的模型，并向公司中的不同团队进行了解释。一个团队熟悉线性模型，另一个团队可能了解决策树。你可以为原始黑盒模型训练两个代理模型 (线性模型和决策树)，并提供两种解释。如果你发现性能更好的黑盒模型，则不必更改解释方法，因为你可以使用同一类代理模型。

我认为这种方法非常**直观**和直接。这意味着它易于实现，而且也易于向不熟悉数据科学或机器学习的人们解释。

使用 **R-平方**测量，我们可以轻松地测量我们的代理模型在逼近黑盒预测方面的表现。

5.6.4 缺点

你必须注意，你**得出的是有关模型而不是数据的结论**，因为代理模型永远看不到实际结果。

尚不清楚 **R-平方的最佳截止点**是什么，以便确信代理模型与黑盒模型足够接近。解释方差的 80%? 50%? 99%?

我们可以衡量代理模型与黑盒模型的接近程度。让我们假设我们不是很接近，但是足够接近。**对于数据集的一个子集，可解释模型可能非常接近，而对于另一子集，则可能发生很大差异**。在这种情况下，对于所有数据点，对简单模型的解释将不尽相同。

你选择的可解释的代理模型**本身的所有优点和缺点**。

有人认为，一般而言，**没有本质上可解释的模型** (甚至包括线性模型和决策树)，甚至有一个可解释性的错觉也是危险的。如果你对此表示赞同，那么这种方法当然不适合你。

5.6.5 软件

我将 R iml 包用作示例。如果你可以训练机器学习模型，那么你应该能够自己实现代理模型。只需训练一个可解释的模型即可预测黑盒模型的预测。

5.7 局部代理 (LIME)

局部代理模型本身是可解释的模型，用于解释黑盒机器学习模型的单个实例预测。局部可解释模型无关的解释 (`Local interpretable model-agnostic explanations`，LIME)[23] 是作者在一篇论文中提出的局部代理模型的具体实现。代理模型经过训练可以近似底层黑盒模型的预测。LIME 并非训练全局代理模型，而是专注于训练局部代理模型以解释单个预测。

这个想法很直观。首先，忘记训练数据，并假设你只有黑盒模型，你可以在其中输入数据点并获得模型的预测。你可以随时探测黑盒。而你的目标是了解机器学习模型为何做出特定的预测。当你将数据变化后输入到机器学习模型中时，LIME 会测得预测发生了什么。LIME 生成一个新的数据集，该数据集由扰动的样本和黑盒模型的相应预测组成。然后，在这个新的数据集上，LIME 训练了一个可解释的模型，该模型通过采样实例与感兴趣实例的接近程度来加权。可解释模型可以是 “可解释的模型” 一章中的任何模型，例如 Lasso 或决策树。所学习的模型应该是机器学习模型局部预测的良好近似，但不一定是良好的全局近似。这种准确性也称为局部保真度。

在数学上，具有可解释性约束的局部代理模型可以表示为:

$$\text{explanation}(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g)$$

例如对实例 x 的解释模型是模型 g (例如线性回归模型)，最小化损失 L (例如均方误差) 测量了解释 g 与原始模型 f (例如 xgboost 模型) 的预测的接近程度，而模型复杂度 $\Omega(g)$ 保持较低水平 (例如，偏好较少的特征)。G 是可能的解释的族，例如所有可能的线性回归模型。接近度 π_x 定义了我们考虑解释时实例 x 附近的邻域大小。实际上，LIME 仅优化损失部分。用户必须确定复杂度，例如，通过选择线性回归模型可以使用的最大特征数。

训练局部代理模型的方法：

- 选择你想要对其黑盒预测进行解释的感兴趣实例。
- 扰动你的数据集并获得这些新点的黑盒预测。
- 根据新样本与目标实例的接近程度对其进行加权。
- 在新数据集上训练加权的，可解释的模型。
- 通过解释局部模型来解释预测。

例如，在 R 和 Python 的当前实现中，可以选择线性回归作为可解释的代理模型。事先，你必须选择 K，即你希望在可解释模型中拥有的特征数量。K 越低，解释这个模型越容易。较高的 K 可能会产生具有较高保真度的模型。有几种方法可以训练具有 K 个特征的模型。Lasso 是一个不错的选择。具有高正则化参数 λ 的 Lasso 模型会生成没有任何特征的模型。通过缓慢减小 λ，重新训练 Lasso 模型，这些特征获得不为零的权重估计。如果模型中有 K 个特征，则你已达到所需的特征数量。其他策略是向前或向后选择特征。这意味着你可以从完整模型 (= 包含所有特征) 开始，或者从仅具有截距的模型开始，然后测试添加或删除时哪个特征将带来最大的改进，直到达到具有 K 个特征的模型。

你如何获得数据的变化？这取决于数据类型，可以是文本，图像或表格数据。对于文本和图像，解决方案是 “打开” 或 “关闭” 某个单词或超像素。对于表格数据，LIME 通过分别对每个特征进行扰动来创建新样本，并从正态分布中提取平均值和标准差。

5.7.1 表格数据的 LIME

表格数据是以表格形式出现的数据，每一行代表一个实例，每一列代表一个特征。LIME 样本不是在感兴趣的实例周围获取，而是从训练数据的质心获取，这是有问题的。但这增加了某些采样点预测的结果与感兴趣的数据点不同的概率，并且 LIME 至少可以学到一些解释。

最好从视觉上解释采样和局部模型训练的工作原理：

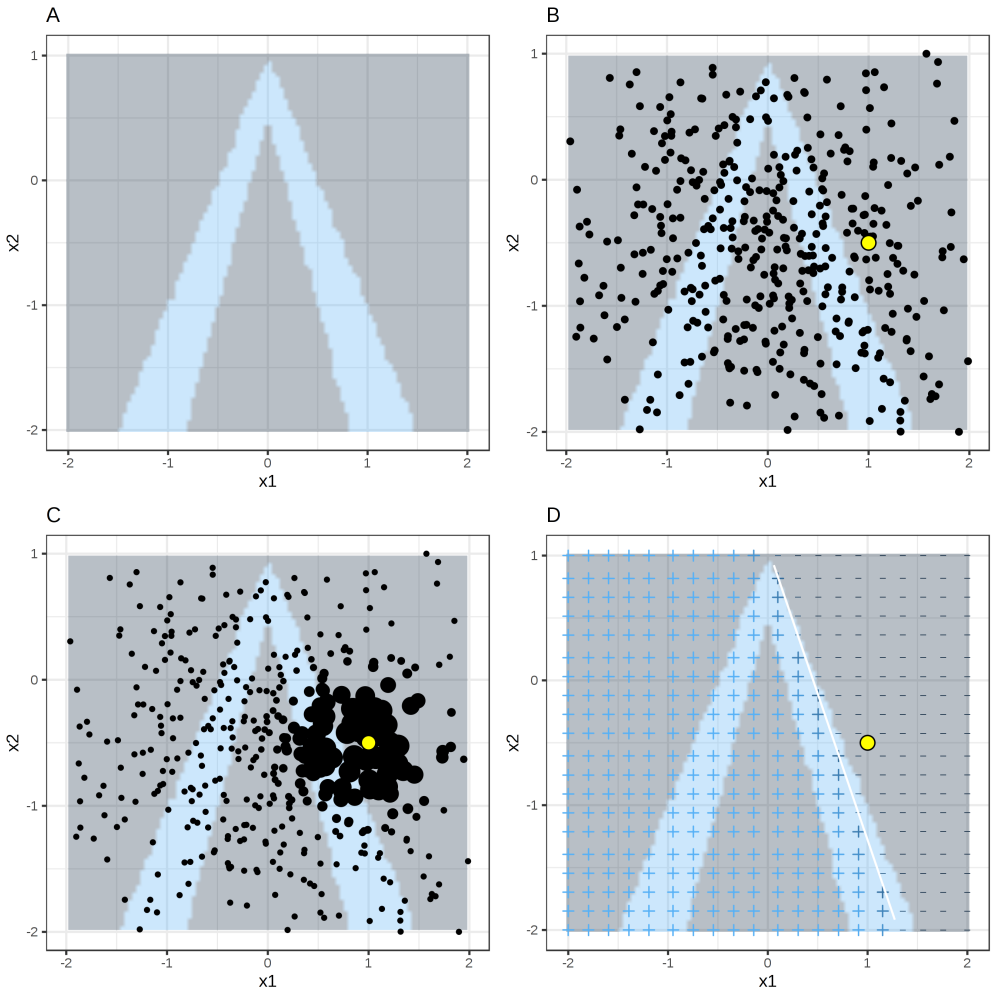


图 5.33. 用于表格数据的 LIME 算法。A) 给定特征 x_1 和 x_2 的随机森林预测。预测的类别：1 (深色) 或 0 (浅色)。B) 感兴趣的实例 (大点) 和从正态分布采样的数据 (小点)。C) 给感兴趣实例附近的点分配更高的权重。D) 网格的符号显示了从加权样本中局部地学习的模型的分类。白线标记了决策边界 ($P(class = 1) = 0.5$)。

与往常一样，细节决定成败。在点周围定义有意义的邻域非常困难。当前 LIME 使用指数平滑核来定义邻域。平滑核是一个函数，它接受两个数据实例并返回一个接近度 (`proximity measure`)。核宽度决定了邻域的大小：较小的核宽度意味着实例必须非常接近才能影响局部模型，较大的核宽度意味着距离较远的实例也会影响模型。如果你看一下 LIME 的 Python 实现 (文件 lime/lime_tabular.py) 你会看到它使用了指数平滑核 (针对标准化数据)，并且核宽度是训练数据的列数的平方根的 0.75 倍。最大的问题是我们没有找到最佳核或宽度的好方法。甚至 0.75 来自哪里？在某些情况下，你可以通过更改核宽度轻松地转换你的解释，如下图所示：

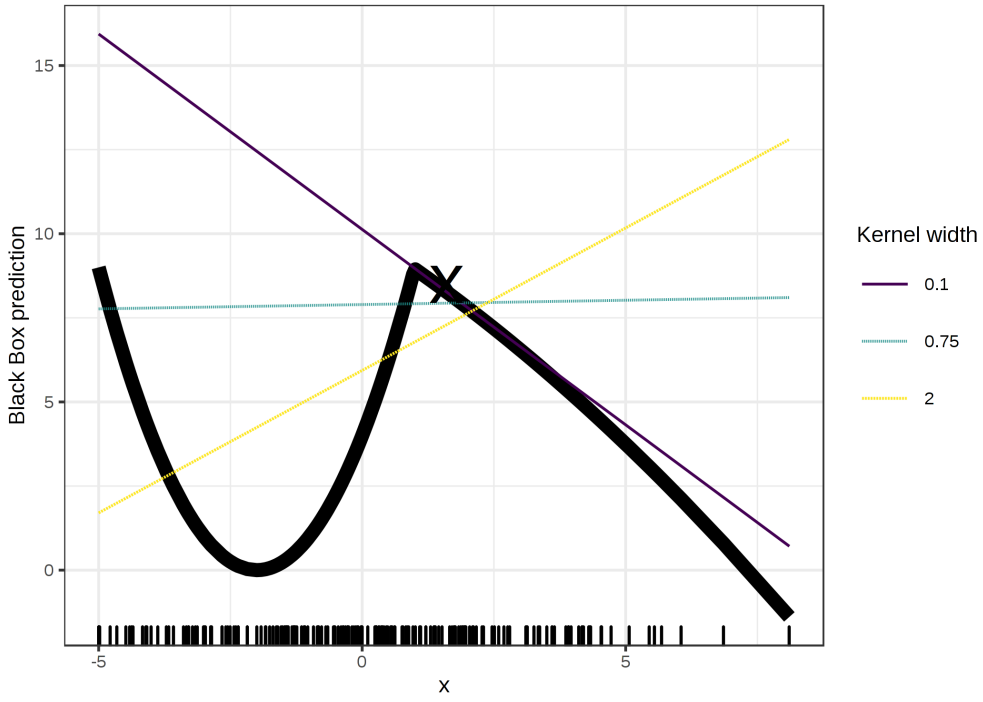


图 5.34. 实例 $x = 1.6$ 的预测解释。黑盒模型根据单个特征的预测以粗线显示，数据分布以 rug 显示。计算了三个具有不同核宽度的局部代理模型。生成的线性回归模型取决于核宽度：对于 $x = 1.6$，特征具有负效应，正效应或是无效应？

该示例仅显示一个特征。在高维特征空间中情况更糟。距离度量是否应该平等对待所有特征也不清楚。特征 x_1 的距离单位等于特征 x_2 的一个单位吗？距离度量是非常任意的，不同维度 (也称为特征) 上的距离可能根本无法比较。

示例

让我们看一个具体的例子。我们返回到自行车租赁数据，并将预测问题转化为一个分类问题：考虑到自行车租赁随着时间的推移变得越来越流行的趋势后，我们想知道在某天租赁的自行车数量是否将高于或低于趋势线。你也可以将“高于”解释为高于平均自行车数量，但会根据趋势进行调整。

首先，我们在分类任务上训练了 100 棵树的随机森林。根据天气和日历信息，租赁自行车的数量将在哪一天高于平均值？

解释包括 2 个特征。针对具有不同预测类别的两个实例训练的稀疏局部线性模型的结果：

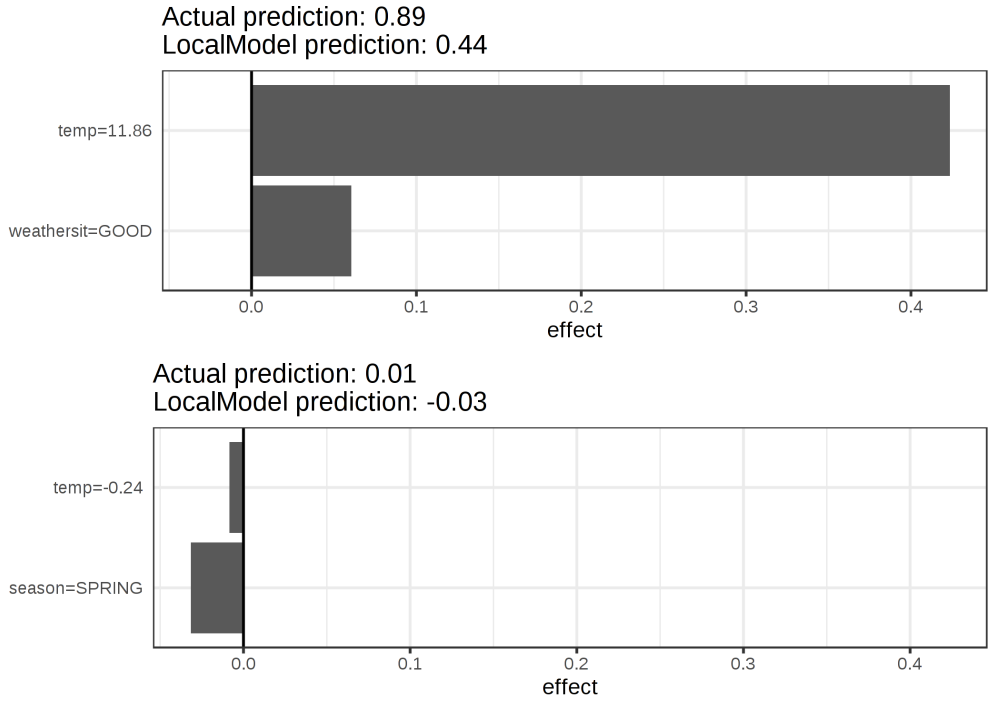


图 5.35. 自行车租赁数据集的两个实例的 LIME 解释。温暖的温度和良好的天气状况对预测有正效应。x 轴显示特征效应：权重乘以实际特征值。

从图中可以清楚地看出，分类特征比数值特征更容易解释。一种解决方案是将数值特征分类到箱中。

5.7.2 文本的 LIME

用于文本的 LIME 与用于表格数据的 LIME 不同。对数据的变化方式不同：

从原始文本开始，通过从原始文本中随机删除单词来创建新文本。数据集用每个单词的二进制特征表示。如果包含相应的单词，则特征为 1；如果已删除，则特征为 0。

示例

在此示例中，我们将 YouTube 评论归类为垃圾或正常。

黑盒模型是在文档词矩阵上训练的深度决策树。每个评论是一行，每一列是给定单词的出现次数。简短的决策树很容易理解，但是在这个示例中树会很深。代替该树的方法，可以是在词嵌入 (抽象向量) 上训练的循环神经网络或支持向量机。让我们看一下该数据集的两个评论以及相应的类 (垃圾评论为 1，正常评论为 0)：

	CONTENT	CLASS
267	PSY is a good guy	0
173	For Christmas Song visit my channel! ;)	1

下一步是创建局部模型中使用的数据集的一些变体。例如，其中一条评论的一些变体 (对原来评论进行变化后的版本)：

	For	Christmas	Song	visit	my	channel!	;)	prob	weight
2	1	0	1	1	0	0	1	0.09	0.57
3	0	1	1	1	1	0	1	0.09	0.71
4	1	0	0	1	1	1	1	0.99	0.71
5	1	0	1	1	1	1	1	0.99	0.86
6	0	1	1	1	0	0	1	0.09	0.57

每一列对应于句子中的一个单词。每行都是一个变体，1 表示该词是该变体的一部分，0 表示该词已被删除。变体之一的对应句子是 “Christmas Song visit my;)”。“prob” 列显示了每个句子变体的垃圾评论的预测概率。“weight” 列显示变体与原始句子的接近程度，计算方式为 1 减去删除的单词所占的比例，例如，如果 7 个单词中删除了 1 个，则接近程度为 1-1/7=0.86。

这是两个句子 (一个是垃圾评论，一个不是垃圾评论)，其估计的局部权重由 LIME 算法找到：

case	label_prob	feature	feature_weight
1	0.0872151	good	0.000000
1	0.0872151	a	0.000000
1	0.0872151	PSY	0.000000
2	0.9939759	channel!	6.908755
2	0.9939759	visit	0.000000
2	0.9939759	Song	0.000000

“channel” 表示垃圾评论的概率很高。对于非垃圾评论，不会估计非零权重，因为无论删除哪个单词，预测类别都将保持不变。

5.7.3 图像的 LIME

用于图像的 LIME 与用于表格数据和文本的 LIME 不同。直观地讲，扰动单个像素没有多大意义，因为许多个像素对一个类有贡献。随机改变单个像素可能不会对预测产生太大影响。因此，通过将图像分割成 “超像素” 并 “关闭” 或 “打开” 超像素来创建图像的变化。超像素是具有相似颜色的互连像素，可以通过将每个像素替换为用户定义的颜色 (例如灰色) 来关闭。用户还可以指定在每次置换中关闭超像素的概率。

示例

由于图像解释的计算相当缓慢，因此，R lime 包包含一个预先计算的示例，我们还将使用该示例演示该方法的输出。这些解释可以直接显示在图像样本上。由于每个图片可以有几个预测标签 (按概率排序)，因此我们可以解释前 n 个预测标签。对于下图，前 3 位的预测是电吉他 (Electric guitar)，民谣吉他 (Acoustic guitar) 和拉布拉多 (Labrador)。

(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

图 5.36. 由 Google Inception 神经网络对图像分类的前三类进行 LIME 解释。该示例摘自 LIME 论文 (Ribeiro 等，2016)。

第一种情况的预测和解释是非常合理的。第一个预测电吉他当然是错误的，但是解释告诉我们神经网络仍然表现合理，因为识别出的图像部分表明这可能是电吉他。

5.7.4 优点

即使你**替换了底层的机器学习模型**，你仍然可以使用相同的局部可解释模型进行解释。假设看解释的人最了解决策树。因为你使用局部代理模型，所以可以将决策树用作解释，而不必实际使用决策树进行预测。例如，你可以使用 SVM。而且，如果发现 xgboost 模型的效果更好，则可以替换 SVM，并且仍然用决策树来解释这些预测。

局部代理模型受益于训练和解释可解释的模型的文献和经验。

当使用 Lasso 或短的树时，得到的**解释是简短的 (= 选择性的)，并且可能是对比性的**。因此，它们做出了人类友好的解释。这就是为什么我在解释的接收者是外行人或时间很少的人的应用程序中

看到 LIME 的原因。对于完整的归因而言，这还不够，因此在合规性场景中我看不到 LIME，在合规性场景中，法律上可能会要求你全面解释预测。对于调试机器学习模型，有所有的原因而不是少数是有用的。

LIME 是少数**适用于表格数据，文本和图像**的方法之一。

保真度度量 (可解释模型与黑盒预测的近似程度) 使我们很好地了解了可解释模型在解释感兴趣的数据实例附近的黑盒预测方面的可靠性。

LIME 在 Python (LIME 库) 和 R (LIME 包和 iml 包) 中实现，并且**非常易于使用**。

用局部代理模型创建的解释**可以使用除原始模型所用以外的其他 (可解释) 特征**。当然，这些可解释的特征必须从数据实例中派生。文本分类器可以将抽象词嵌入作为特征，但解释可以基于句子中是否存在词。回归模型可以依赖于某些属性的不可解释的转换，但是可以使用原始属性来创建解释。例如，可以在答案的主成分分析 (PCA) 的成分上训练回归模型，而 LIME 可以在原始问题上训练。与其他方法相比，对 LIME 使用可解释特征可能是一个很大的优势，尤其是当模型使用不可解释特征进行训练时。

5.7.5 缺点

当对表格式数据使用 LIME 时，正确定义邻域是一个很大的未解决的问题。我认为这是 LIME 的最大问题，也是我建议仅谨慎使用 LIME 的原因。对于每个应用程序，你都必须尝试不同的核设置，并亲自查看解释是否有意义。不幸的是，这是我可以找到合适的核宽度的最佳建议。

在当前 LIME 的实现中可以改善采样。从高斯分布中采样数据点，而忽略特征之间的相关性。这可能会导致不太可能的数据点，然后这些数据点可用于学习局部解释模型。

解释模型的复杂性必须事先定义。这只是一个小小的抱怨，因为最终用户总是必须定义保真度和稀疏度之间的折衷。

另一个真正的大问题是解释的不稳定。在 [32] 中，作者表明，在模拟环境中两个非常接近的点的解释差异很大。另外，以我的经验，如果你重复采样过程，那么出来的解释可能会有所不同。不稳定意味着很难相信这些解释，你应该非常严格。

结论：以 LIME 作为具体实现的局部代理模型非常有应用前景。但是该方法仍处于开发阶段，需要解决许多问题才能安全应用。

5.8 Shapley 值

可以通过假设实例的每个特征值是游戏中的“玩家”来解释预测，其中预测是“总支出”。Shapley 值是联盟博弈论的一种方法，它告诉我们如何在特征之间公平地分配“总支出”。

5.8.1 总体思路

假定以下情况：

你已经训练了机器学习模型来预测公寓价格。对于某套公寓，它的预计价格为 300,000 欧元，你需要对此做出解释。公寓的大小为 50 平方米，位于 2 楼，附近有公园，并且禁止猫入内：

图 5.37. 一套 50 平方米的公寓，附近有公园以及禁止猫入内，预计价格为 300,000 欧元。我们的目标是解释这些特征值如何对预测做出贡献。

所有公寓的平均预测价格为 310,000 欧元。与平均预测相比，每个特征值对预测有多少贡献？

对于线性回归模型，答案很简单。每个特征的效应是特征的权重乘以特征值。这仅因模型的线性而起作用。对于更复杂的模型，我们需要不同的解决方案。例如，LIME 建议使用局部模型来估计特征效应。另一种解决方案来自合作博弈理论：Shapley 值 (由 Shapley (1953)[33] 创造) 是一种根据玩家对总支出的贡献分配支出给玩家的方法。玩家在联盟中进行合作，并从此合作中获得一定的收益。

玩家？游戏？支出？与机器学习预测和可解释性有什么关系？“游戏”是数据集单个实例的预测任务。“收益”是此实例的实际预测值减去所有实例的平均预测值。“玩家”是实例的特征值，它们协同工作以获得收益。在我们的公寓示例中，特征值“公园附近”、“禁止猫进入”、“50 平方米”和“2 楼”

共同实现 300,000 欧元的预测。我们的目标是解释实际预测 (300,000 欧元) 与平均预测 (310,000 欧元) 之间的差额：-10,000 欧元。

答案可能是："公园附近" 贡献 30,000 欧元；"50 平方米" 贡献 10,000 欧元；"2 楼" 贡献 0 欧元；"禁止猫进入" 贡献 -50,000 欧元。贡献总计为 -10,000 欧元，即最终预测减去平均预测公寓价格。

我们如何计算一个特征的 Shapley 值?

Shapley 值是所有可能的联盟 (`Coalition`) 中特征值的平均边际贡献。这么说清楚了吗?

在下图中，我们评估了将 "禁止猫进入" 添加到 "公园附近" 和 "50 平方米" 组成的联盟中时的贡献。我们通过从数据中随机绘制另一套公寓，模拟出只有 "公园附近"、"禁止猫进入" 和 "50 平方米" 在一个联盟中，并将它的值用于楼层特征。"2 楼" 被随机抽取的 "1 楼" 所取代，然后我们预测这个组合的公寓价格 (310,000 欧元)。在第二步中，我们将 "禁止猫进入" 从联盟中删除，从随机绘制的公寓中将猫允许/禁止进入的特征值替换。在这个随机示例中为 "允许猫进入"，但也可能再是 "禁止猫进入"。我们预测对 "公园附近" 和 "50 平方米" 联盟的公寓价格 (320,000 欧元)。"禁止猫进入" 的贡献 310,000 - 320,000 = -10.000 欧元。此估计取决于用作猫和楼层特征值的 "贡献者" 的随机绘制的值。如果我们重复此采样步骤并平均贡献，将会得到更好的估计。

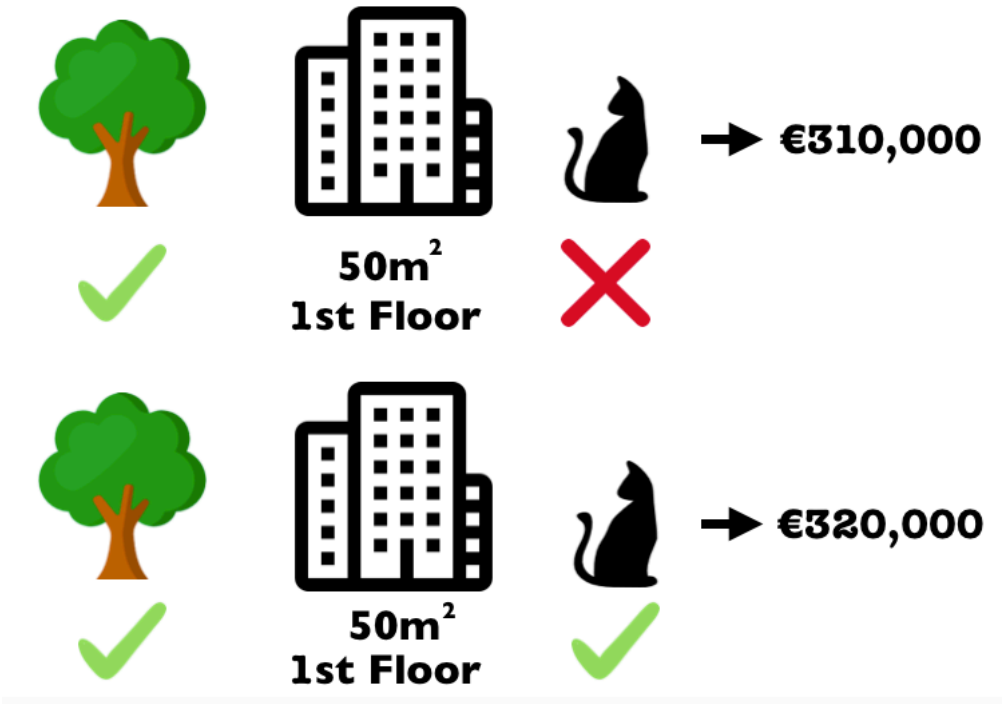


图 5.38. 一个样本来估计当 "禁止猫进入" 加入 "公园附近" 和 "50 平方米" 联盟时对预测的贡献。

我们对所有可能的联盟重复此计算。Shapley 值是对所有可能的联盟的所有边际贡献的平均值。计

算时间随特征数量呈指数增长。保持计算时间可控的一个解决方案是只计算可能联盟的少量样本的贡献。

下图显示了确定“禁止猫进入”的 Shapley 值所需的所有特征值联盟。第一行显示没有任何特征值的联盟。第二，第三和第四行显示随联盟大小增加而不同的联盟，以“|”分隔。总而言之，可能的联盟如下：

- 空
- “公园附近”
- “50 平方米”
- “2 楼”
- “公园附近”+“50 平方米”
- “公园附近”+“2 楼”
- “50 平方米”+“2 楼”
- “公园附近”+“50 平方米”+“2 楼”

对于这些联盟中的每个联盟，我们都计算带有或不带有特征值“禁止猫进入”的预测公寓价格，并取其差值以获得边际贡献。Shapley 值是边际贡献的 (加权) 平均值。我们用公寓数据集中的随机特征值替换不在联盟中的特征的特征值，以从机器学习模型中获得预测。

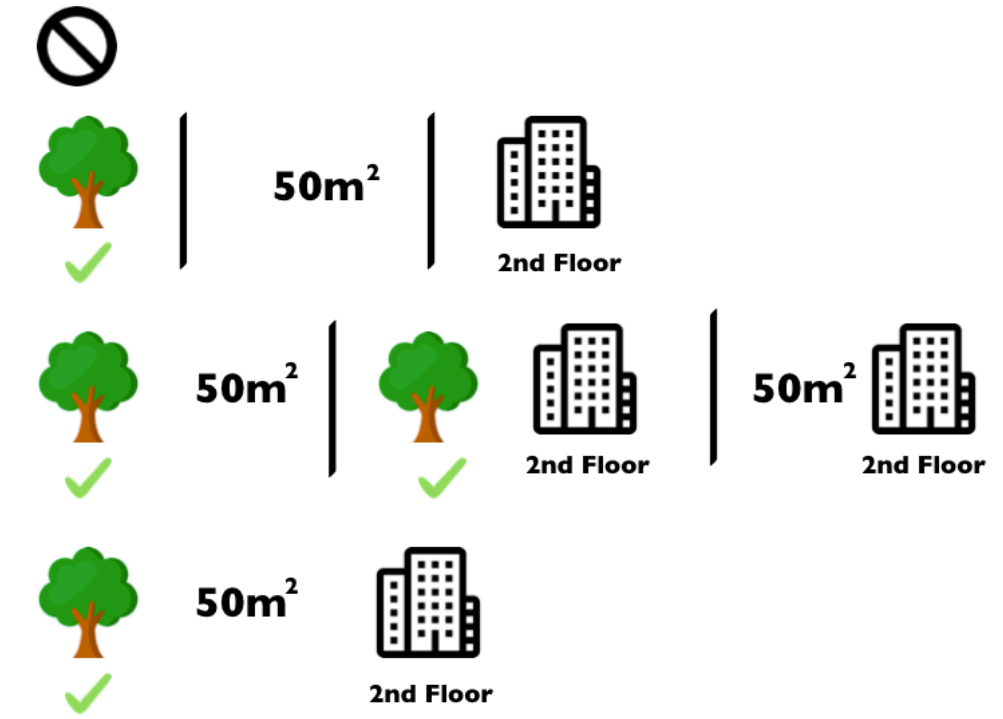


图 5.39. 用于计算“禁止猫进入”的精确 Shapley 值的所有 8 个联盟。

如果我们估计所有特征值的 Shapley 值，则可以得到特征值之间预测的完整分布 (减去平均值)。

5.8.2 示例与解释

特征值 j 的 Shapley 值的解释是：与数据集的平均预测相比，第 j 个特征的值对这个特定实例的预测的贡献为 ϕ_j。

Shapley 值适用于分类 (如果我们要处理概率) 和回归。

我们使用 Shapley 值来分析预测宫颈癌的随机森林模型的预测：

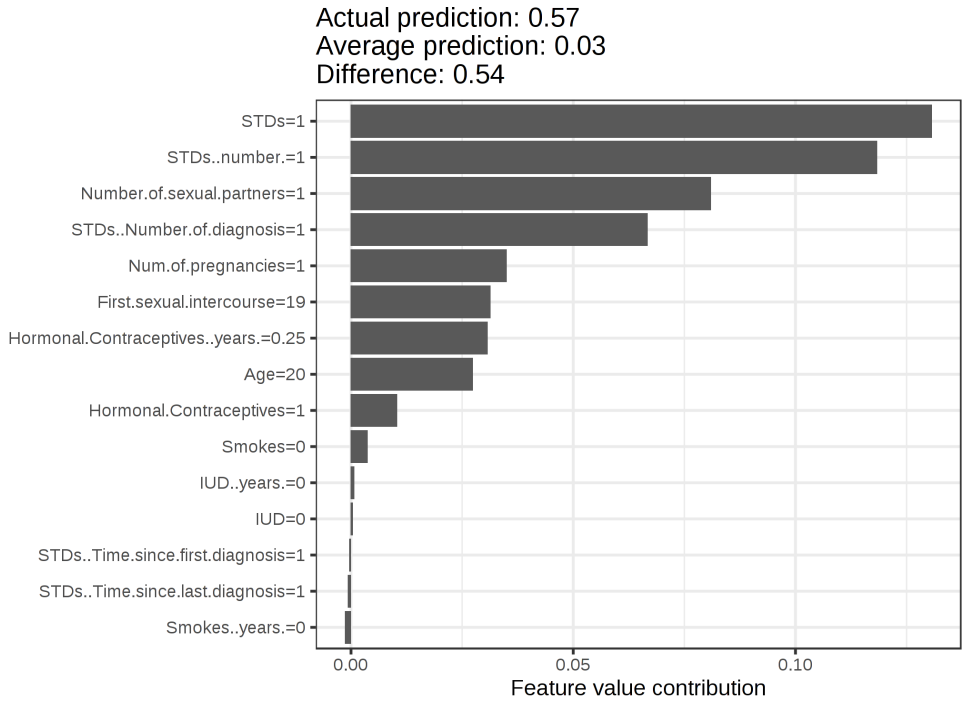


图 5.40. 宫颈癌数据集中一名女性的 Shapley 值。预测为 0.57，则该女性患癌的概率比平均预测值 0.03 高 0.54。被诊断的性病 (STDs) 数量对概率的增加最大。贡献之和产生实际和平均预测之间的差异 (0.54)。

对于自行车租赁数据集，我们还训练了一个随机森林，在给定天气和日历信息的情况下预测一天的自行车租赁数量。为特定日期的随机森林预测创建的解释：

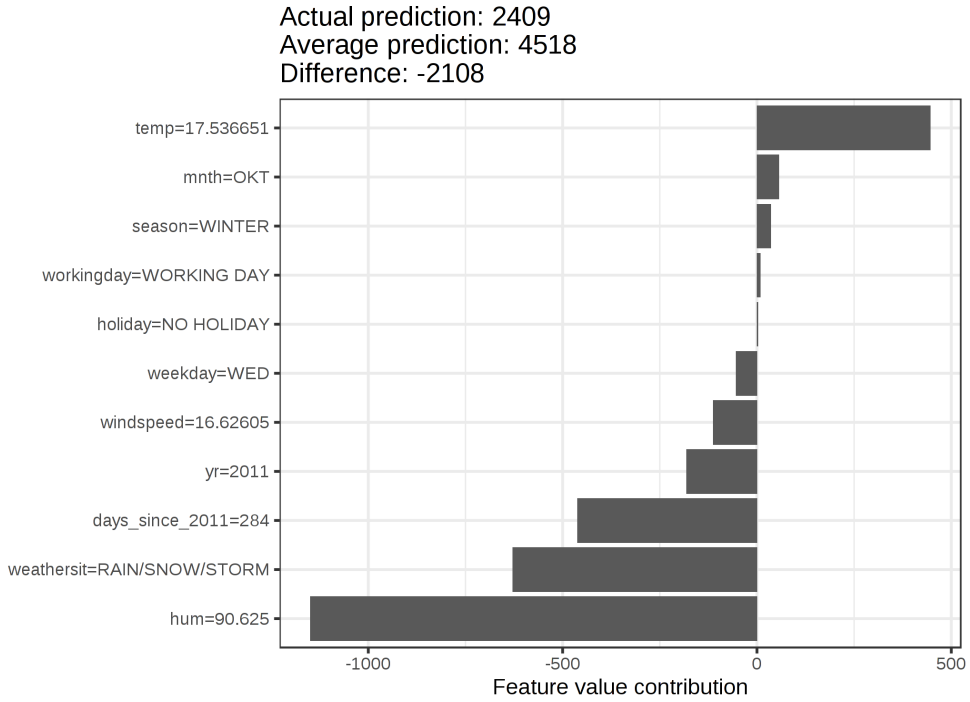


图 5.41. 第 285 天的 Shapley 值。预计会有 2409 辆出租自行车，这一天比平均预测值 4518 少 2108。天气情况和湿度对负值的影响最大。这一天的温度有积极的贡献。Shapley 值之和产生实际和平均预测的差异 (-2108)。

请注意正确解释 Shapley 值：Shapley 值是特征值在不同联盟中对预测的平均贡献。当我们从模型中删除特征时，Shapley 值并不是预测的差异。

5.8.3 详细的 Shapley 值

本节将为好奇的读者更深入地介绍 Shapley 值的定义和计算。如果你对技术细节不感兴趣，请跳过本节，直接转到 “优点和缺点”。

我们对每个特征如何影响数据点的预测很感兴趣。在线性模型中，很容易计算出各个特征效应。这是一个数据实例的线性模型预测的样子：

$$\hat{f}(x) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

其中 x 是要计算其贡献的实例。每个 x_j 都是一个特征值 ($j = 1 \ldots p$)。β_j 是特征 j 相对应的权重。

第 j 个特征对预测 $\hat{f}(x)$ 的贡献为 ϕ_j：

$$\phi_j(\hat{f}) = \beta_j x_j - E(\beta_j X_j) = \beta_j x_j - \beta_j E(X_j)$$

其中 $E(\beta_j X_j)$ 是特征 j 的平均效应估计值。贡献是特征效应减去平均效应的差。好极了！现在我们知道每个特征对预测的贡献。如果我们将一个实例的所有特征贡献相加，则结果如下：

$$\begin{aligned}\sum_{j=1}^{p}\phi_j(\hat{f}) &= \sum_{j=1}^{p}(\beta_j x_j - E(\beta_j X_j)) \\ &= (\beta_0 + \sum_{j=1}^{p}\beta_j x_j) - (\beta_0 + \sum_{j=1}^{p}E(\beta_j X_j)) \\ &= \hat{f}(x) - E(\hat{f}(X))\end{aligned}$$

这是数据点 x 的预测值减去平均预测值。特征贡献可能为负。

我们可以对任何类型的模型执行相同操作吗？将此作为模型无关的工具将是很棒的。由于我们通常在其他模型类型中没有相似的权重，因此我们需要不同的解决方案。

帮助来自意想不到的地方：合作博弈理论。Shapley 值是一种针对任何机器学习模型的单个预测计算特征贡献的解决方案。

Shapley 值

Shapley 值是通过 S 中玩家的值函数 val 定义的。

每个特征值的 Shapley 值是其对总支出 (预测) 的贡献，在所有可能的特征值组合上加权和求和：

$$\phi_j(val) = \sum_{S\subseteq\{x_1,\ldots,x_p\}\setminus\{x_j\}} \frac{|S|!\,(p-|S|-1)!}{p!}\left(val\left(S\cup\{x_j\}\right) - val(S)\right)$$

其中 S 是模型中使用的特征的子集，x 是要解释的实例的特征值的向量，p 是特征的数量。$val_x(S)$ 是对集合 S 中的特征值的预测，它是在集合 S 中未包含的特征上边缘化：

$$val_x(S) = \int \hat{f}(x_1,\ldots,x_p)d\mathbb{P}_{x\notin S} - E_X(\hat{f}(X))$$

实际上你对每个未包含的特征执行了多次积分。一个具体的示例：机器学习模型可用于 4 个特征 x_1，x_2，x_3 和 x_4，并且我们评估由特征值 x_1 和 x_3 组成的联盟 S 的预测：

$$val_x(S) = val_x(\{x_1,x_3\}) = \int_{\mathbb{R}}\int_{\mathbb{R}} \hat{f}(x_1,X_2,x_3,X_4)d\mathbb{P}_{X_2X_4} - E_X(\hat{f}(X))$$

这看起来类似于线性模型中的特征贡献！

不要对 “值” 一词的多种用法感到困惑：特征值是实例特征的数值或分类值；Shapley 值是特征对预测的贡献；值函数是玩家 (特征值) 联盟的支出函数。

Shapley 值是唯一满足效益性 (`Efficiency`)，对称性 (`Symmetry`)，虚拟性 (`Dummy`) 和可加性 (`Additivity`) 的归因方法，这些性质一起可以视为公平支出的定义。

效益性

特征贡献必须加起来等于 x 和平均值的预测差。

$$\sum_{j=1}^{p}\phi_j = \hat{f}(x) - E_X(\hat{f}(X))$$

对称性

如果两个特征值 j 和 k 对所有可能的联盟均贡献相同，则它们的贡献应相同。如果

$$val(S \cup \{x_j\}) = val(S \cup \{x_k\})$$

对于所有

$$S \subseteq \{x_1, \ldots, x_p\} \setminus \{x_j, x_k\}$$

那么

$$\phi_j = \phi_k$$

虚拟性

假设无论将特征值添加到哪个集合，都不会改变预测值的特征 j，它的 Shapley 值应为 0。

如果

$$val(S \cup \{x_j\}) = val(S)$$

对于所有

$$S \subseteq \{x_1, \ldots, x_p\}$$

那么

$$\phi_j = 0$$

可加性

对于具有组合支出 $val + val^+$ 的游戏，各自的 Shapley 值如下：

$$\phi_j + \phi_j^+$$

假设你训练了一个随机森林，这意味着预测是许多决策树的平均值。“可加性” 性质保证对于特征值，你可以分别计算每棵树的 Shapley 值，取平均值，然后获得随机森林的特征值的 Shapley 值。

直觉 理解 Shapley 值的一种直观方法：特征值以随机顺序进入房间。房间中的所有特征值都参与游戏 (= 有助于预测)。特征值的 Shapley 值是当该特征值加入它们时，已经存在于房间中的联盟所收到的预测的平均变化。

估算 Shapley 值

精确的 Shapley 值必须使用第 j 个特征和不使用第 j 个特征的所有可能的联盟 (集合) 来估计。当特征比较多时，随着更多特征的添加，可能的联盟数量呈指数增长，因此对该问题的精确解成为问题。Strumbelj 等人 (2014)[34] 提出了蒙特卡洛抽样的近似值：

$$\hat{\phi}_j = \frac{1}{M}\sum_{m=1}^{M}\left(\hat{f}(x_{+j}^m) - \hat{f}(x_{-j}^m)\right)$$

其中 $\hat{f}(x_{+j}^m)$ 是 x 的预测，除了特征 j 的值，其他不在联盟内的特征值被来自随机数据点 z 的特征值替换。x 向量 x_{-j}^m 与 x_{+j}^m 几乎相同，但被随机数据点 z 的特征值替换的特征包括特征 j。每一个新实例都是一种 “弗兰肯斯坦怪物”，由两个实例组合而成。

单个特征值的近似 Shapley 估计 迭代次数 M，关注的实例 x，特征索引 j，数据矩阵 X 和机器学习模型 f 实例 x 的第 j 个特征值的 Shapley

$m \leftarrow 1$ to N 从数据矩阵 X 中随机抽取实例 z 生成特征的随机顺序 o 随机顺序实例 x：$x_o = (x_{(1)}, \ldots, x_{(j)}, \ldots, x_{(p)})$ 随机顺序实例 z：$z_o = (z_{(1)}, \ldots, z_{(j)}, \ldots, z_{(p)})$ 构造两个新实例有特征 j：$x_{+j} = (x_{(1)}, \ldots, x_{(j-1)}, x_{(j)}, z_{(j+1)}, \ldots, z_{(p)})$ 没有特征 j：$x_{-j} = (x_{(1)}, \ldots, x_{(j-1)}, z_{(j)}, z_{(j+1)}, \ldots, z_{(p)})$ 计算边际贡献：$\phi_j^m = \hat{f}(x_{+j}) - \hat{f}(x_{-j})$ 计算平均值作为 Shapley 值：$\phi_j(x) = \frac{1}{M}\sum_{m=1}^{M}\phi_j^m$

首先，选择感兴趣的实例 x，特征 j 和迭代次数 M。对于每个迭代，从数据中选择一个随机实例 z，并生成特征的随机顺序。通过组合感兴趣的实例 x 和样本 z 的值来创建两个新实例。第一个实例 x_{+j} 是感兴趣的实例，但是之前的所有值 (包括特征 j 的值) 都被样本 z 中的特征值替代。第二个实例 x_{-j} 相似，但是所有值均按之前的顺序排列，但不包括用样本 z 中的特征 j 的值替换了特征 j 的值。从黑盒中预测的差异计算出来：

$$\phi_j^m = \hat{f}(x_{+j}^m) - \hat{f}(x_{-j}^m)$$

所有这些差异取平均，结果如下：

$$\phi_j(x) = \frac{1}{M}\sum_{m=1}^{M}\phi_j^m$$

通过 x 的概率分布隐式地对样本进行平均。

必须对每个特征重复此过程，才能获得所有的 Shapley 值。

5.8.4 优点

预测值与平均预测值之间的差异在实例的特征值 (即 Shapley 值的效益性) 之间**公平分配**。此性质将 Shapley 值与其他方法 (例如 LIME) 区分开来。LIME 不保证预测在特征之间公平分配。Shapley 值可能是提供完整解释的唯一方法。在法律要求可解释性的情况下 (例如欧盟的 “解释权”)，Shapley

值可能是唯一合法的方法，因为它基于扎实的理论并公平地分配效应。我不是律师，所以这仅反映了我对要求的直觉。

Shapley 值允许进行**对比性解释**。无需将预测与整个数据集的平均预测进行比较，你可以将其与子集甚至单个数据点进行比较。这种对比也是像 LIME 这样的局部模型所不具备的。

Shapley 值是唯一具有**扎实理论**的解释方法。公理——效益性，对称性，虚拟性，可加性——为解释提供了合理的基础。诸如 LIME 之类的方法在局部假设了机器学习模型的线性行为，但是尚无关于为何应起作用的理论。

将预测解释为特征值所玩的游戏真是令人难以置信。

5.8.5 缺点

Shapley 值**需要大量的计算时间**。在 99.9% 的实际问题中，只有近似解决方案是可行的。Shapley 值的精确计算在计算上成本很高，因为有 2^p 个必须通过绘制随机实例来模拟特征值的可能联盟和特征的 “不存在”，这会增加 Shapley 值估计的估计方差。联盟的指数数量可以通过对联盟进行采样并限制迭代次数 M 来处理。减少 M 会减少计算时间，但会增加 Shapley 值的方差。对于迭代次数 M 没有很好的经验法则。M 应该足够大以准确地估计 Shapley 值，但是应该足够小以在合理的时间内完成计算。应该有可能基于 Chernoff 边界选择 M，但是我还没有看到关于 Shapley 值进行机器学习预测的文章。

Shapley 值可能会**被误解**。特征值的 Shapley 值不是从模型训练中删除特征后的预测值之差。Shapley 值的解释是：给定当前的一组特征值，特征值对实际预测值与平均预测值之差的贡献就是估计的 Shapley 值。

如果你寻求稀疏解释 (包含很少特征的解释)，则 Shapley 值是错误的解释方法。使用 Shapley 值方法创建的解释**始终使用所有特征**。人类更喜欢选择性的解释，例如 LIME 提出的解释。LIME 可能是非专业人士必须处理的更好的选择。另一个解决方案是 Lundberg 和 Lee (2016)[35] 提出的 SHAP，它基于 Shapley 值，但也提供了很少的特征解释。

Shapley 值为每个特征返回一个简单值，但**没有像 LIME 这样的预测模型**。这意味着它不能用于对输入变化的预测做出变化的陈述，例如：“如果我每年多赚 300 欧元，我的信用评分将提高 5 点”。

另一个缺点是，如果要计算新数据实例的 Shapley 值，则**需要访问数据**。访问预测函数还不够，因为你需要数据以随机抽取的数据实例中的值替换感兴趣的实例的某些部分。只有在你可以创建看起来像真实数据实例但不是训练数据中实际实例的数据实例时，才能避免这种情况。

与许多其他基于置换的解释方法一样，Shapley 值方法在特征相关时**会遇到不现实的数据实例**。为了模拟联盟中缺少特征值，我们将特征边缘化。这是通过从特征的边缘分布中采样值来实现的。只

要特征是独立的就可以。当特征依赖时，我们可能会采样对该实例没有意义的特征值。但是我们将使用它们来计算特征的 Shapley 值。据我所知，尚无关于 Shapley 值含义的研究，也未提出解决方法。一种解决方案可能是将相关特征置换在一起，并为其获取一个相互的 Shapley 值。或者，调整采样程序以考虑特征的依赖性。

5.8.6 软件和替代方法

Shapley 值在 R iml 包中实现。

SHAP 是 Shapley 值的另一种估算方法，将在下一节中介绍。

另一方法称为 breakDown，其在 R breakDown 包 [36] 中实现。BreakDown 还显示了每个特征对预测的贡献，但会逐步进行计算。让我们重用游戏类比：我们从一个空团队开始，添加对预测贡献最大的特征值，然后迭代直到添加所有特征值。每个特征值贡献多少取决于 “团队” 中已经存在的各个特征值，这是 BreakDown 方法的主要缺点。它比 Shapley 值方法快，并且对于没有交互作用的模型，结果是相同的。

5.9 SHAP

Lundberg 和 Lee (2016)[35] 的 SHAP (`SHapley Additive ExPlanations`) 是一种解释个体预测的方法。SHAP 基于博弈论上的最佳 Shapley 值。

SHAP 拥有自己的一节，而不是 Shapley 值的子节有两个原因。首先，SHAP 的作者提出了 KernelSHAP，这是一种基于核的代理方法，可根据局部代理模型对 Shapley 值进行估算。他们提出了 TreeSHAP，这是一种基于树的模型的有效估计方法。其次，SHAP 带有许多基于 Shapley 值聚合的全局解释方法。本节介绍了新的估计方法和全局解释方法。

我建议先阅读有关 Shapley 值和局部模型 (LIME) 的小节内容。

5.9.1 定义

SHAP 的目标是通过计算每个特征对预测 x 的贡献来解释实例 x 的预测。SHAP 解释方法根据联盟博弈理论计算 Shapley 值。数据实例的特征值充当联盟中的参与者。Shapley 值告诉我们如何在特征之间平均分配 “总支出” (即预测)，玩家可以是单个特征值，例如表格数据。玩家也可以是一组特征值。例如，为了解释图像，可以将像素分组为超像素，并将预测分布在超像素之间。SHAP 带来的一项创新是，Shapley 值的解释表示为一种可加的特征归因方法，即线性模型。这将 LIME 和

Shapley 值联系起来。SHAP 将解释指定为：

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$$

其中 g 是解释模型，$z' \in \{0,1\}^M$ 是联盟向量，M 是最大联盟大小，$\phi_j \in \mathbb{R}$ 是特征 j 的特征归因 Shapley 值。在 SHAP 论文中，我所谓的 “联盟向量” 被称为 “简化特征”。我认为这个名称是因为例如在图像数据中，图像未在像素级别上表示，而是聚合为超像素。我认为考虑 z' 来描述联盟是有帮助的：在联盟向量中，输入 1 表示相应的特征值 “存在”，而输入 0 则表示 “不存在”。如果你知道 Shapley 值，这应该听起来很熟悉。为了计算 Shapley 值，我们模拟了只有一些特征值在游戏 (“存在”)，而有些则没有 (“不存在”)。联盟线性模型的表示是计算 ϕ 的技巧。对于 x (感兴趣的实例)，联盟向量 x' 是全为 1 的向量，即所有特征值均为 “存在”。该公式简化为：

$$g(x') = \phi_0 + \sum_{j=1}^{M} \phi_j$$

你可以在 Shapley 值一节中以类似的符号找到此公式。有关实际估计的更多信息将在稍后描述。让我们先谈谈 ϕ 的性质，然后再讨论它们的估计细节。

Shapley 值是唯一满足效益性、对称性、虚拟性和可加性的解。SHAP 也满足这些要求，因为它可以计算 Shapley 值。在 SHAP 论文中，你会发现 SHAP 性质和 Shapley 性质之间的差异。SHAP 描述了以下三个理想的性质：

1) **局部准确性** (`Local accuracy`)

$$f(x) = g(x') = \phi_0 + \sum_{j=1}^{M} \phi_j x'_j$$

表示特征归因的总和等于我们要解释的模型的输出。如果定义 $\phi_0 = E_X(\hat{f}(x))$ 并将所有 x'_j 都设置为 1，则这是 Shapley 效益性。仅使用不同的名称并使用联盟向量。

$$f(x) = \phi_0 + \sum_{j=1}^{M} \phi_j x'_j = E_X(\hat{f}(X)) + \sum_{j=1}^{M} \phi_j$$

2) **缺失性** (`Missingness`)

$$x'_j = 0 \Rightarrow \phi_j = 0$$

缺失性表示缺失特征的归因为零。请注意，x'_j 表示联盟，其中值 0 表示不存在特征值。在联盟表示法中，要解释的实例的所有特征值 x'_j 均应为 “1”。0 的存在将意味着感兴趣的实例缺少特征值。此性质不在 “常规” Shapley 值的性质中。那么，为什么我们需要 SHAP？Lundberg 称其为 “小型簿记财产” (`Minor Book-keeping Property`)。从理论上讲，缺失的特征可以具有任意的 Shapley 值，而不会损害局部准确性，因为它与 $x'_j = 0$ 相乘。缺失性强制缺少的特征获得 Shapley 值 0。

3) **一致性** (Consistency)

让 $f_x(z') = f(h_x(z'))$ 且 $z_{\setminus j'}$ 表明 $z'_j = 0$。对于任意两个模型 f 和 f' 满足，对所有输入 $z' \in \{0,1\}^M$：

$$f'_x(z') - f'_x(z'_{\setminus j}) \geq f_x(z') - f_x(z'_{\setminus j})$$

则有：

$$\phi_j(f', x) \geq \phi_j(f, x)$$

一致性表示，如果模型发生变化，以使特征值的边际贡献增加或保持不变 (不考虑其他特征)，则 Shapley 值也会增加或保持不变。从一致性中得出 Shapley 性质的可加性，虚拟性和对称性，如 Lundberg 和 Lee 的附录中所述。

5.9.2 KernelSHAP

KernelSHAP 为一个实例 x 估算每个特征值对预测的贡献。KernelSHAP 包含 5 个步骤：

- 采样联盟 $z'_k \in \{0,1\}^M, k \in \{1,\ldots,K\}$ (1 = 联盟中存在特征，0 = 不存在特征)。
- 对 z'_k 预测，方法是首先将 z'_k 转换为原始特征空间，然后应用模型 f：$f(h_x(z'_k))$。
- 使用 SHAP 核计算每个 z'_k 的权重。
- 拟合加权线性模型。
- 返回 Shapley 值 ϕ_k，即线性模型的系数。

我们可以通过重复掷硬币来建立随机联盟，直到我们得到一个由 0 和 1 组成的链。例如，向量 (1, 0, 1, 0) 意味着我们具有第一个和第三个特征的联盟。K 个采样的联盟成为回归模型的数据集。回归模型的目标是联盟的预测。(“等等！”，你可能说，“该模型尚未针对这些二进制的联盟数据进行训练，因此无法对其进行预测。”) 为了从特征值的联盟转化为有效的数据实例，我们需要一个函数 $h_x(z') = z$ 其中 $h_x : \{0,1\}^M \to \mathbb{R}^p$。函数 h_x 将 1 映射到我们要解释的实例 x 的对应值。对于表格数据，它将 0 映射到我们从数据中采样的另一个实例的值。这意味着我们将 “缺少特征值” 等同于 “将特征值替换为数据中的随机特征值”。对于表格数据，下图显示了从联盟到特征值的映射：

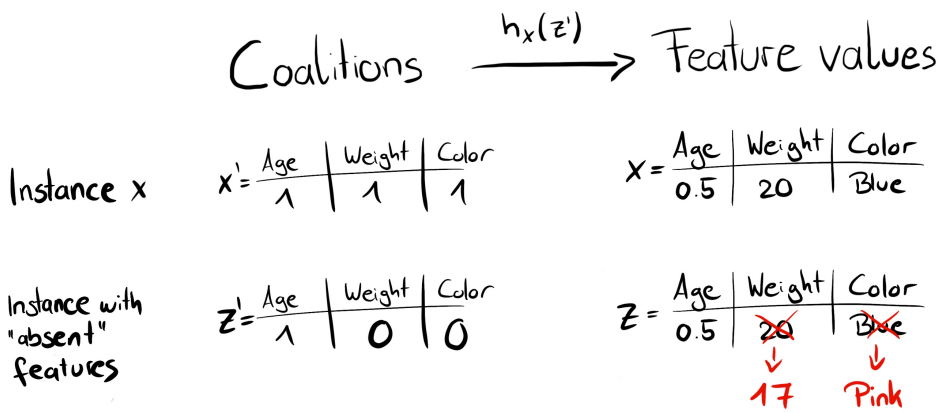


图 5.42. 函数 h_x 将联盟映射到有效实例。对于当前特征 (1)，h_x 映射到 x 的特征值。对于缺少特征 (0)，h_x 映射到随机采样的数据实例的值。

在理想情况下，h_x 会根据当前特征值对不存在的特征值进行采样：

$$f(h_x(z')) = E_{X_C|X_S}[f(x)|x_S]$$

其中 X_C 是缺少的特征集，X_S 是当前的特征集。但是，如上所述，用于表格数据的 h_x 将 X_C 和 X_S 视为独立的，并在边际分布上进行积分：

$$f(h_x(z')) = E_{X_C}[f(x)]$$

从边际分布采样意味着忽略当前特征和不存在特征之间的依赖关系。因此，KernelSHAP 与所有基于置换的解释方法都存在相同的问题。该估计对不太可能发生的情况给予了过多的重视。结果可能变得不可靠。稍后我们将看到，树集成算法 TreeSHAP 不受此问题的影响。

对于图像，下图描述了可能的映射函数：

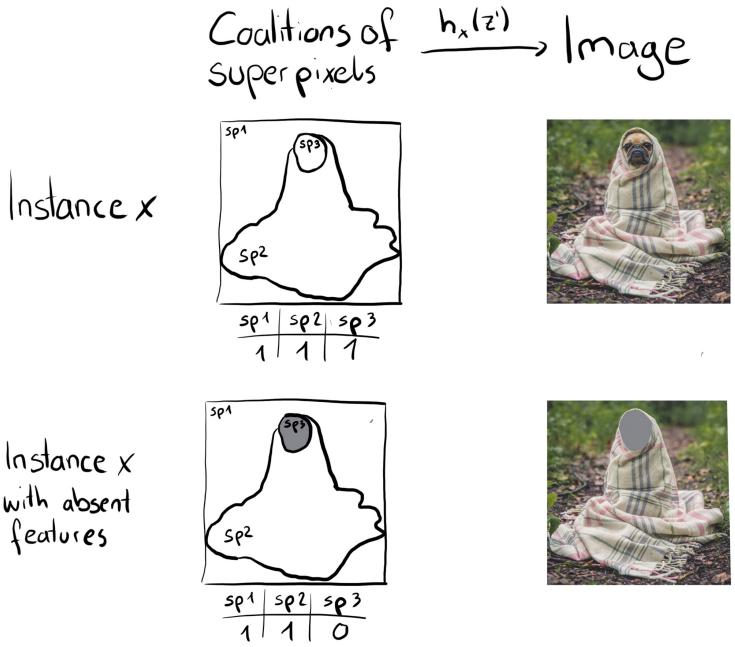


图 5.43. 函数 h_x 将超像素 (sp) 的联盟映射到图像。超像素是像素组。对于当前特征 (1)，h_x 返回原始图像的相应部分。对于不存在的特征 (0)，h_x 将使相应区域变灰。变为周围像素或类似像素的平均颜色也是一种选择。

与 LIME 的最大区别在于回归模型中实例的权重。LIME 根据实例与原始实例的接近程度对其进行加权。联盟向量中的 0 越多，LIME 中的权重就越小。SHAP 根据联盟在 Shapley 值估计中获得的权重对采样实例进行加权。小型联盟 (少量 1) 和大型联盟 (即许多 1) 获得最大权重。它的直觉是：如果我们可以单独研究它们的影响，我们将最大程度地了解单个特征。如果一个联盟由一个特征组成，那么我们可以了解特征对预测的孤立的主要效应。如果一个联盟只少了一个特征，那么我们可以了解这个特征的总效应 (主要效应以及特征交互)。如果一个联盟由一半特征组成，那么我们对单个特征的贡献知之甚少，因为存在许多可能具有一半特征的联盟。为了达到 Shapley 标准的加权，Lundberg 等提出了 SHAP 核：

$$\pi_x(z') = \frac{(M-1)}{\binom{M}{|z'|}|z'|(M-|z'|)}$$

这里，M 是最大联盟大小，$|z'|$ 实例 z' 中当前特征的数量。Lundberg 和 Lee 表明，用这个核权重的线性回归会产生 Shapley 值。如果你将 SHAP 核与 LIME 一起用于联盟数据，则 LIME 还将估算 Shapley 值!

对于联盟的采样，我们可以更聪明一些：最小的联盟和最大的联盟会承担大部分权重。通过使用采样预算 K 包括这些高权重联盟，而不是盲目采样，我们可以获得更好的 Shapley 值估计。我们从具有 1 和 $M-1$ 特征的所有可能的联盟开始，这总共有 $2M$。当我们有足够的预算 (当前预算为

$K-2M$) 时，我们可以包括具有 2 个特征以及具有 $M-2$ 特征的联盟，等等。从剩余的联盟规模中，我们使用调整后的权重进行采样。

现在我们有数据、目标和权重。从而建立加权线性回归模型：

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$$

我们通过优化以下损失函数 L 来训练线性模型 g：

$$L(f, g, \pi_x) = \sum_{z' \in Z} [f(h_x(z')) - g(z')]^2 \pi_x(z')$$

其中 Z 是训练数据。这是我们通常对线性模型进行优化的误差平方和。模型的估计系数 ϕ_j 是 Shapley 值。

由于我们处于线性回归设置中，因此我们也可以使用标准工具进行回归。例如，我们可以添加正则化项以使模型稀疏。如果在损失 L 上加上 L1 惩罚，我们可以创建稀疏的解释。(我不确定所得出的系数是否仍然是有效的 Shapley 值。)

5.9.3 TreeSHAP

Lundberg 等人 (2018)[37] 提出了 TreeSHAP，这是 SHAP 的一种变体，用于基于树的机器学习模型，例如决策树、随机森林和梯度提升树。TreeSHAP 速度很快，可以计算精确的 Shapley 值，并且在特征相关时可以正确估计 Shapley 值。相比之下，KernelSHAP 的计算成本很高，而且只能近似实际的 Shapley 值。

TreeSHAP 快多少? 对于精确的 Shapley 值，它将计算复杂度从 $O(TL2^M)$ 降低到了 $O(TLD^2)$，其中 T 是树的数量，L 是所有树中的最大叶子数量，D 是所有树的最大深度。

TreeSHAP 估计正确的条件期望 $E_{X_S|X_C}[f(x)|x_S]$。我将为你提供一些直觉，我们如何计算单个树，实例 x 和特征子集 S 的期望预测。如果我们以所有特征为条件——如果 S 是所有特征的集合——那么来自实例 x 所在节点的预测将是期望预测。如果我们对任何特征都没有条件——如果 S 为空——我们将使用所有终端节点的预测的加权平均值。如果 S 包含一些但不是全部特征，则我们将忽略不可到达节点的预测。不可到达意味着通向该节点的决策路径与 x_S 中的值相矛盾。在剩余的终端节点中，我们根据节点大小 (即该节点中训练样本的数量) 对预测结果进行加权平均。剩余终端节点的平均值 (由每个节点的实例数加权) 是给定 S 时 x 的期望预测。问题是我们必须对特征值的每个可能子集 S 应用此过程。幸运的是，TreeSHAP 的计算时间是多项式级而不是指数级。基本思想是同时将所有可能的子集 S 推到树上。对于每个决策节点，我们必须跟踪子集的数量。这取决于父节点中的子集和分割特征。例如，当树中的第一个分割基于特征 x_3，则包含特征 x_3 的所有子集都将到达一个节点 (x 所走的节点)。不包含特征 x_3 的子集伴随权重降低同时到达两边节点。

不幸的是，不同大小的子集具有不同的权重。该算法必须跟踪每个节点中子集的总体权重。这使算法变得复杂。有关 TreeSHAP 的详细信息，请参阅原始论文。可以将计算扩展到更多的树：由于 Shapley 值的可加性，树集成的 Shapley 值是各个树的 Shapley 值的 (加权平均值)。

接下来，我们将看一下 SHAP 的解释。

5.9.4 示例

我用 100 棵树训练了一个随机森林分类器，预测宫颈癌的风险。我们将使用 SHAP 来解释个体预测。我们可以使用快速 TreeSHAP 估计方法而不是较慢的 KernelSHAP 方法，因为随机森林是树的集成。

由于 SHAP 计算 Shapley 值，因此其解释与 Shapley 值一节中的解释相同。但是，随着 Python shap 软件包带来了不同的可视化效果：你可以将诸如 Shapley 值之类的特征归因可视化为 “力” (force)。每个特征值都是增加或减少预测的力。预测从基线开始。Shapley 值的基线是所有预测的平均值。在图中，每个 Shapley 值都是一个箭头，可推动增加 (正值) 或减小 (负值) 预测。这些力在数据实例的实际预测中相互平衡。

下图显示了宫颈癌数据集中两名女性的 SHAP 解释力图：

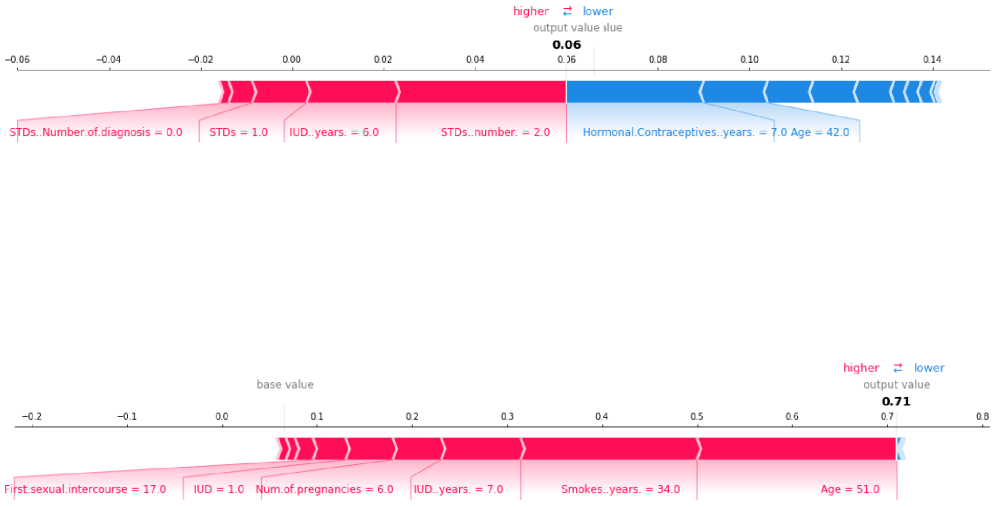


图 5.44. SHAP 值可以解释两个人的预测癌症概率。基线 (平均预测概率) 为 0.066。第一位女性有低预测风险 0.06。诸如性病 (STDs) 之类的风险增加的效应被诸如年龄 (Age) 之类的减少的效应所抵消。第二名女性的预测风险很高，为 0.71。51 岁和吸烟 34 年增加了她预测患癌症的风险。

这些是对单个预测的解释。

Shapley 值可以合并为全局解释。如果我们为每个实例运行 SHAP，则将获得 Shapley 值矩阵。此矩阵每个数据实例具有一行，每个特征具有一列。我们可以通过分析此矩阵中的 Shapley 值来解释整个模型。

现在我们再开始介绍 SHAP 特征重要性。

5.9.5 SHAP 特征重要性

SHAP 特征重要性背后的想法很简单：具有较大的 Shapley 绝对值的特征很重要。由于我们需要全局重要性，因此我们在数据中对每个特征的 Shapley 绝对值取平均值：

$$I_j = \sum_{i=1}^{n} |\phi_j^{(i)}|$$

接下来，我们通过重要性对特征进行降序排序并绘制它们。下图显示了 SHAP 特征对于预测宫颈癌的经过训练的随机森林的 SHAP 特征重要性。

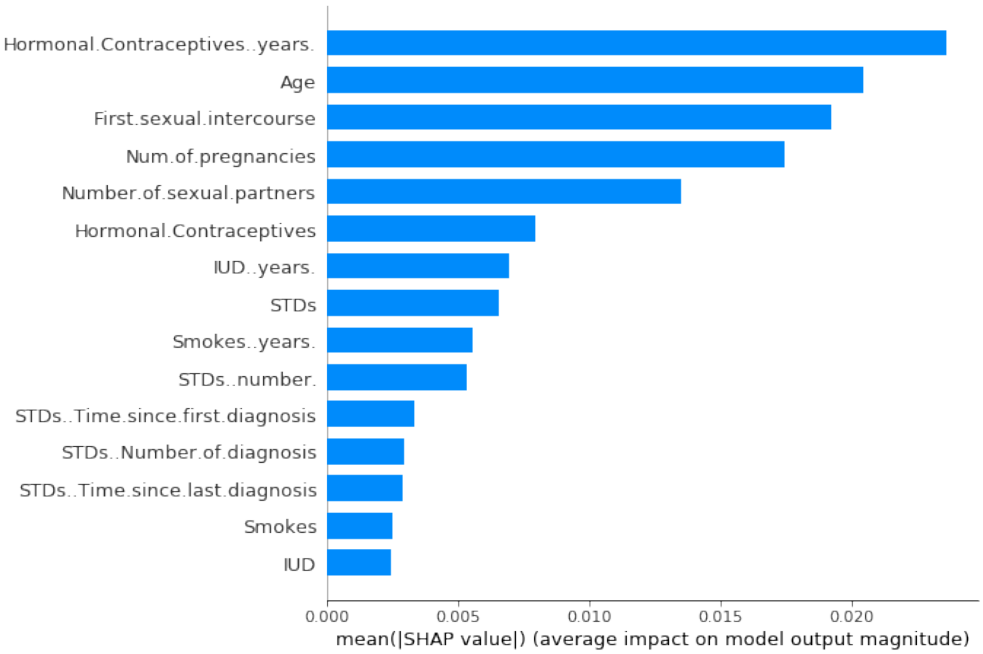


图 5.45. SHAP 特征重要性的度量是平均 Shapley 绝对值。使用激素类避孕药的年数是最重要的特征，平均将预测的癌症绝对概率改变了 2.4 个百分点 (x 轴为 0.024)。

SHAP 特征重要性是置换特征重要性的替代方法。两种重要性度量之间存在很大差异：置换特征重要性基于模型性能的下降。SHAP 基于特征归因的大小。

特征重要性图很有用，但是除了重要性之外不包含任何信息。要获得更多信息，我们接下来将看一下摘要图。

5.9.6 SHAP 概要图

概要图将特征重要性与特征效应结合在一起。概要图上的每个点都是一个特征和一个实例的 Shapley 值。y 轴上的位置由特征确定，x 轴上的位置由 Shapley 值确定。颜色代表特征值从低到高。重叠点在 y 轴方向上抖动，因此我们可以了解每个特征的 Shapley 值的分布。这些特征根据其重要性排序。

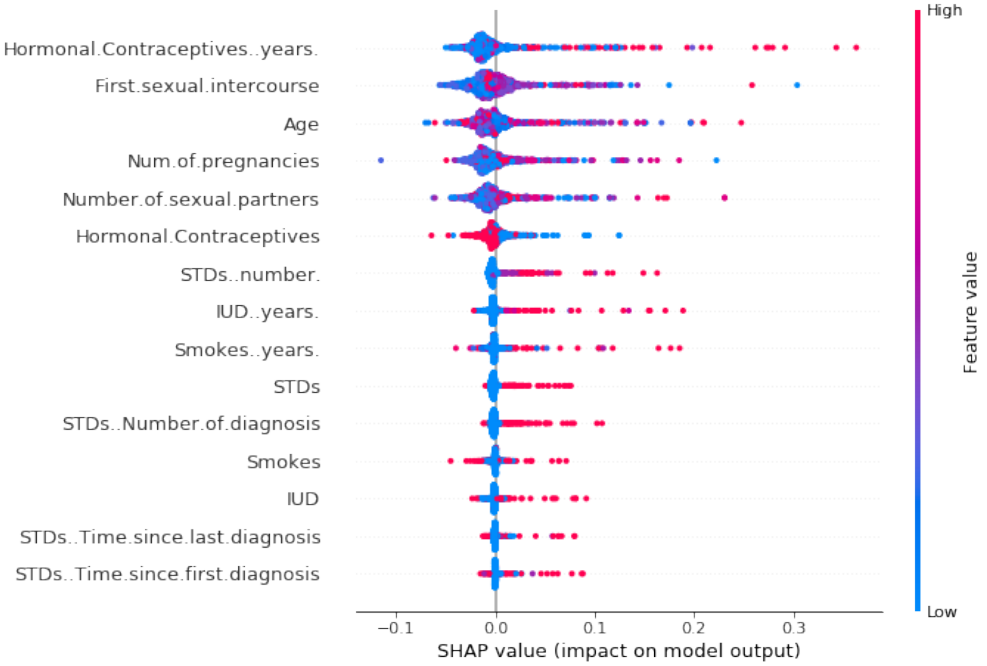


图 5.46. SHAP 概要图。使用激素类避孕药的年数越少，患癌症的风险越低；年数越多，患癌症的风险越高。注意：所有效应都描述了模型的行为，在现实世界中不一定是因果关系。

在概要图中，我们首先看到特征值与对预测的影响之间的关系。但要了解这种关系的确切形式，我们必须查看 SHAP 依赖图。

5.9.7 SHAP 依赖图

SHAP 特征依赖关系可能是最简单的全局解释图：

1. 选择一个特征。

2. 对于每个数据实例，绘制一个点，在 x 轴上是特征值，在 y 轴上是相应的 Shapley 值。

3. 完成。

从数学上讲，该图包含以下点：$\{(x_j^{(i)}, \phi_j^{(i)})\}_{i=1}^n$

下图显示了对使用激素类避孕药的年数的 SHAP 特征依赖性：

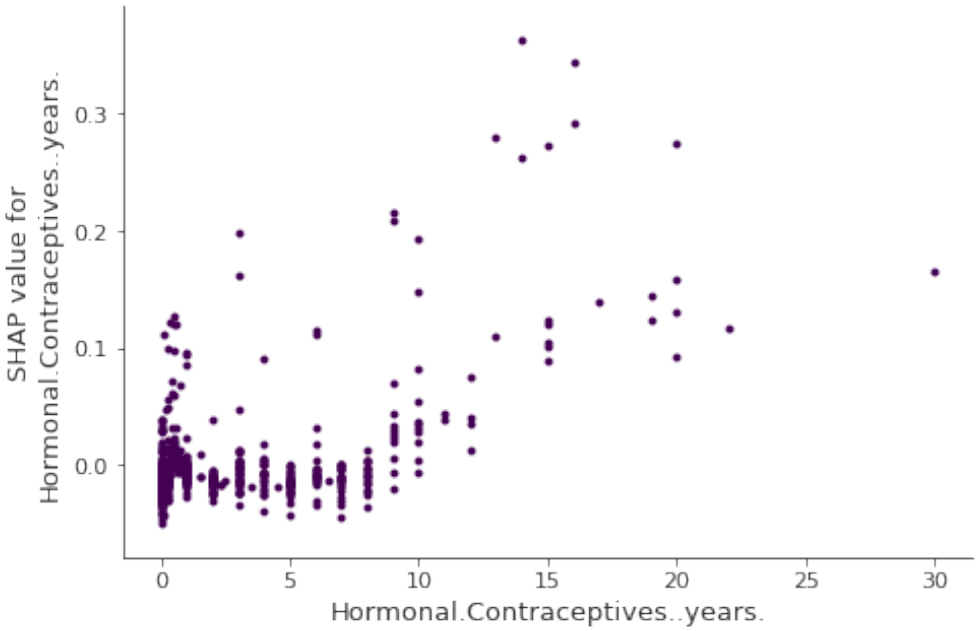


图 5.47. 使用激素类避孕药的年数的 SHAP 依赖图。与 0 年相比，使用激素类避孕药的年数少会降低预测的概率，而高的年数会增加预测的癌症概率。

SHAP 依赖图是部分依赖图和累积局部效应图的替代方法。PDP 和 ALE 图显示平均效应，而 SHAP 依赖性还显示 y 轴的方差。特别是在交互作用的情况下，SHAP 依赖图在 y 轴上更加分散。突出显示这些特征交互，可以改进依赖图。

5.9.8 SHAP 交互值

交互效应是在考虑了单个特征效应之后的附加的组合特征效应。博弈论中的 Shapley 交互指数定义为：

$$\phi_{i,j} = \sum_{S \subseteq \setminus \{i,j\}} \frac{|S|!(M - |S| - 2)!}{2(M-1)!} \delta_{ij}(S)$$

当 $i \neq j$ 且：

$$\delta_{ij}(S) = f_x(S \cup \{i,j\}) - f_x(S \cup \{i\}) - f_x(S \cup \{j\}) + f_x(S)$$

该公式减去了特征的主要效应，因此我们在考虑了单个效应后得到了纯交互效应。与 Shapley 值计算中一样，我们对所有可能的特征联盟 S 上的值求平均值。当我们计算所有特征的 SHAP 交互值时，每个实例得到一个矩阵，维数为 $M \times M$，其中 M 为特征数量。

我们如何使用交互指数？例如，下面自动与交互作用最强的特征绘制 SHAP 依赖图：

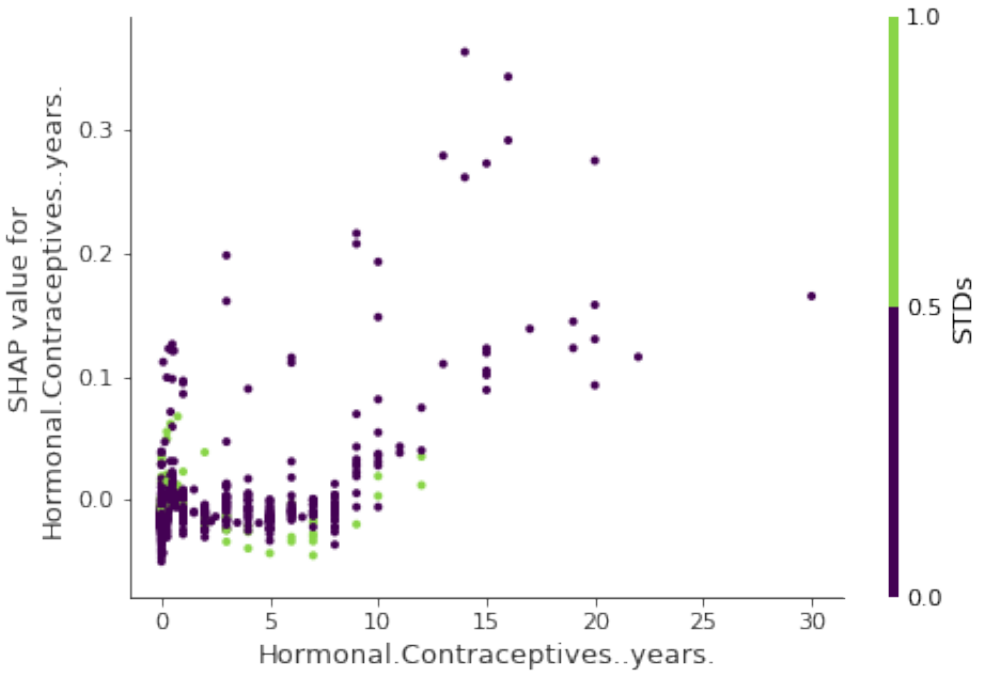


图 5.48. 具有交互可视化的 SHAP 特征依赖图。使用激素类避孕药的年数与性病相互作用。在接近 0 年的情况下，性病的发生会增加预测的癌症风险。使用激素类避孕药多年，性病的发生降低了预测的风险。同样，这不是因果模型。效应可能是由于混杂 (例如性病和较低的癌症风险可能与更多的就诊机会相关)。

5.9.9 聚类 SHAP 值

你可以借助 Shapley 值对数据进行聚类。聚类的目的是找到相似实例的组。通常，聚类是基于特征的。特征通常在不同的尺度上。例如，高度可能以米为单位，颜色强度从 0 到 100，某些传感器输出在 -1 和 1 之间。难点在于计算具有这种不同、不可比较的特征的实例之间的距离。

SHAP 聚类通过对每个实例的 Shapley 值进行聚类来工作。这意味着你可以通过解释相似性来聚类实例。所有 SHAP 值都具有相同的单位——在预测空间的单位。你可以使用任何聚类方法。以下示例使用层次聚类对实例进行排序。

该图由许多力图组成，每个力图都解释了实例的预测。我们垂直旋转力图，并根据它们的聚类相似性将它们并排放置。

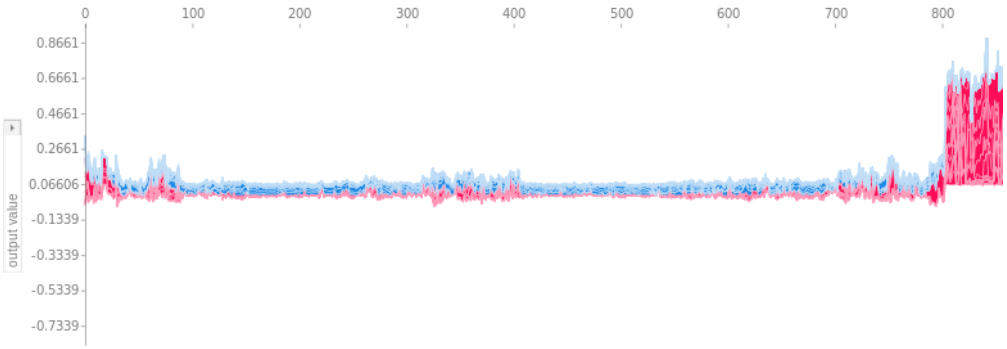


图 5.49.　堆叠的 SHAP 解释按解释相似性聚类。x 轴上的每个位置都是数据的一个实例。红色 SHAP 值会增加预测，蓝色值会降低预测。聚类突出：右侧是预测癌症风险较高的人群。

5.9.10　优点

由于 SHAP 计算 Shapley 值，因此应用了 Shapley 值的所有优点：SHAP 在博弈论中**具有扎实的理论基础**。预测结果在特征值中**公平分配**。我们得到**对比性解释**，将预测与平均预测进行比较。

SHAP **连接 LIME 和 Shapley 值**。这对于更好地了解这两种方法非常有用。它还有助于统一可解释机器学习的领域。

SHAP **可以快速实现基于树的模型**。我认为这是 SHAP 普及的关键，因为采用 Shapley 值的最大障碍是计算速度慢。

快速的计算使**计算全局模型解释**所需的许多 Shapley 值成为可能。全局解释方法包括特征重要性，特征依赖，交互作用，聚类和摘要图。使用 SHAP，全局解释与局部解释一致，因为 Shapley 值是全局解释的 “原子单位”。如果将 LIME 用于局部解释，将部分依赖图和置换特征重要性用于全局解释，则你会缺乏通用的基础。

5.9.11　缺点

KernelSHAP 很慢。当你要为许多实例计算 Shapley 值时，这使得使用 KernelSHAP 不切实际。同样，所有全局 SHAP 方法 (例如 SHAP 特征重要性) 都需要为许多实例计算 Shapley 值。

KernelSHAP 忽略特征相关性。大多数其他基于置换的解释方法都存在此问题。通过用随机实例中的值替换特征值，通常更容易从边缘分布中随机采样。但是，如果特征是相关的，这就会导致把过多的权重放在不太可能的数据点上。TreeSHAP 通过显式建模条件期望预测来解决此问题。

Shapley 值的缺点也适用于 SHAP：Shapley 值可能会被错误解释，并且需要访问数据才能为新数据计算 SHAP 值 (TreeSHAP 方法除外)。

5.9.12 软件

作者在 Python SHAP 包中实现了 SHAP。此实现适用于 Python 的 scikit-learn 机器学习库中的基于树的模型。shap 包也用于本章中的示例。SHAP 已集成到树增强框架 XGBoost 和 LightGBM 中。在 R 中，有一个 shapper 包。SHAP 也包含在 R xgboost 软件包中。

第六章　基于样本的解释

基于样本的解释方法 (Example-based Explanations) 选择数据集的特定实例来解释机器学习模型的行为或解释底层数据分布。

基于样本的解释大多与模型无关，因为它们使任何机器学习模型都更具可解释性。与模型无关的方法的不同之处在于，基于样本的方法通过选择数据集的实例而不是通过创建特征概要 (例如特征重要性或部分依赖性) 来解释模型。只有当我们可以以人类可以理解的方式表示数据实例时，基于样本的解释才有意义。这对于图像非常有效，因为我们可以直接查看它们。通常，如果实例的特征值包含更多的上下文，那么基于样本的方法就会很好地工作，这意味着数据具有像图像或文本一样的结构。以一种有意义的方式表示表格数据更具挑战性，因为一个实例可以包含数百或数千个 (较少结构化的) 特征。列出所有的特征值来描述一个实例通常是没有用的。如果只有少数几个特征，或者我们有一种方法来总结一个实例，那么它可以很好地工作。

基于样本的解释可帮助人们构建机器学习模型以及机器学习模型所训练的数据的心理模型。它特别有助于理解复杂的数据分布。但是，基于样本的解释是什么意思? 我们经常在工作和日常生活中使用它们。让我们从一些示例开始 [38]。

医师看到病人出现异常咳嗽和轻度发烧。病人的症状使她想起了她多年前遇到的患有类似症状的另一位病人。她怀疑她目前的病人可能患有相同的疾病，并且她采集了血液样本以测试这种检测疾病。

一位数据科学家正在为他的一位客户做一个新项目：分析导致键盘生产机器故障的风险因素。数据科学家记得他曾经做过的一个类似项目，并且重用了旧项目中的部分代码，因为他认为客户希望进行同样的分析。

一只小猫坐在一个失火的无人居住的房子的窗台上。消防部门已经到达，其中一名消防队员正在考虑他是否可以冒险进入大楼救小猫。他还记得自己当消防员时遇到的类似情况：缓慢燃烧了一段时间的老木屋往往不稳定，最终倒塌。由于这种情况的相似性，他决定不进入，因为房屋倒塌的风险太大。幸运的是，猫咪跳出窗户，安全着陆，没有人受伤。这是个美满结局。

这些故事说明了我们人类在样例或类比中是如何思考的。基于样本的解释的蓝图是：事物 B 与事物 A 类似，事物 A 导致 Y，因此我预测事物 B 也将引起 Y。隐式地，一些机器学习方法是基于样本的。决策树根据对预测目标很重要的特征 (数据点的相似性) 将数据划分为节点。决策树通过查

找相似的实例 (= 在相同的终端节点中) 并返回这些实例结果的平均值作为预测来获取新数据实例的预测。k-最近邻 (knn) 方法可以显式地处理基于样本的预测。对于一个新实例，knn 模型可以找到 k 个最近的邻居 (例如，k=3 个最近的实例)，并返回这些邻居结果的平均值作为预测。可以通过返回 k 个邻居来解释 knn 的预测，这同样仅当我们有一个好方法表示单个实例时才有意义。

本部分的各节涵盖以下基于样本的解释方法：

- **反事实解释**告诉我们实例必须如何改变才能显着改变其预测。通过创建反事实实例，我们了解模型如何做出预测并可以解释各个预测。
- **对抗样本**是用来欺骗机器学习模型的反事实。重点是翻转预测而不是解释它。
- **原型**是从数据中选择具有代表性的实例，而**批评**是那些原型无法很好地表示的实例。[4]
- **有影响力的实例**是对预测模型的参数或预测本身影响最大的训练数据点。识别和分析有影响力的实例有助于发现数据问题、调试模型并更好地了解模型的行为。
- **k-最近邻**模型是基于样本的 (可解释的) 机器学习模型。

6.1 反事实解释

反事实解释 (`Counterfactual Explanations`) 按以下形式描述了一种因果关系：“如果没有发生 X，那么 Y 就不会发生”。例如：“如果我不喝一杯热咖啡，我的舌头就不会烫伤了”。事件 Y 是烫伤了我的舌头；原因 X 是我喝了热咖啡。思考反事实需要想象一个与所观察到的事实相矛盾的假设现实 (例如，一个我没有喝热咖啡的世界)，因此被称为“反事实”。在反事实中进行思考的能力使我们人类比其他动物更加聪明。

在可解释的机器学习中，反事实解释可用于解释各个实例的预测。“事件” 是实例的预测结果，“原因” 是该实例的特定特征值，将其输入到模型并会 “引起” 某个预测。以图的形式显示，输入和预测之间的关系非常简单: 特征值导致预测。

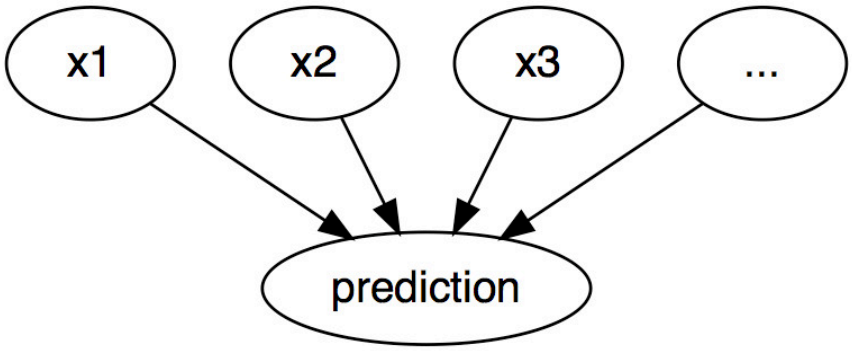


图 6.1. 当模型仅被视为一个黑盒时，机器学习模型输入与预测之间的因果关系。输入导致预测 (不一定反映数据的真正因果关系)。

即使实际上输入与要预测的结果之间的关系可能不是因果关系，我们也可以将模型的输入视为预测的原因。

有了这个简单的图，就很容易看出我们如何模拟反事实来预测机器学习模型：我们只是在做出预测之前改变实例的特征值，然后分析预测是如何变化的。我们对预测以相关方式发生变化的场景感兴趣，例如预测类别发生翻转 (例如接受或拒绝信贷申请) 或预测达到某个阈值 (例如癌症的概率达到 10%)。**预测的反事实解释描述了将预测更改为预定义输出时特征值的最小变化**。

反事实解释方法与模型无关，因为它仅适用于模型输入和输出。由于该解释可以表示为特征值差异的概要 ("更改特征 A 和 B 以更改预测")，因此该方法在模型无关的章节中也很适用。但是反事实解释本身就是一个新实例，因此它存在于本章中 ("从实例 X 开始，改变 A 和 B 以得到一个反事实实例")。与原型不同，反事实不一定是来自训练数据的实际实例，而可以是特征值的新组合。

在讨论如何创建反事实之前，我想讨论一些反事实的示例以及一个好的反事实解释是怎样的。

在第一个示例中，Peter 申请了一笔贷款，并被 (基于机器学习的) 银行软件拒绝了。他想知道为什么他的申请被拒绝，以及他怎样才能提高获得贷款的机会。"为什么" 的问题可以表述为反事实：对特征 (收入，信用卡数量，年龄等) 的最小变化是什么，可以使预测从拒绝变为批准？一个可能的答案是：如果 Peter 每年能多赚 10,000 欧元，他将获得贷款。或者，如果 Peter 的信用卡较少，并且 5 年前没有拖欠贷款，那么他会得到贷款。Peter 永远不会知道拒绝的原因，因为银行对透明度没有兴趣，但这是另一回事。

在我们的第二个示例中，我们想用反事实解释来解释一个预测连续结果的模型。Anna 想把她的公寓租出去，但她不确定要收取多少费用，因此她决定训练一个机器学习模型来预测租金。当然，由于 Anna 是一位数据科学家，因此她可以解决自己的问题。输入有关面积大小、位置、是否允许携带宠物等的所有详细信息之后，模型告诉她可以收取 900 欧元。她期望 1000 欧元或更多，但是她相信自己的模型，并决定使用公寓的特征值了解如何提高公寓的价格。她发现，如果公寓面积再大 15 平方米，则可以以超过 1000 欧元的价格出租。有趣，但不可行，因为她无法扩大自己的公寓。最后，通过仅在其控制下调整特征值 (内置厨房是/否，允许宠物是/否，地板类型等)，她发现如果允许宠物并安装隔热效果更好的窗户，她可以收取 1000 欧元。Anna 凭直觉与反事实合作来改变结果。

反事实是对人类友好的解释，因为它们与当前实例形成对比，并且它们是选择性的，这意味着它们通常专注于少量特征更改。但是反事实却遭受 "罗生门效应" 的困扰。《罗生门》是一部日本电影，讲述了不同的人对一个武士的谋杀。每个故事都很好地解释了结果，但故事彼此矛盾。反事实也可能发生同样的情况，因为通常会有多种不同的反事实解释。每个反事实都讲述了如何达到特定结果的不同 "故事"。一个反事实可能会说要更改特征 A，另一反事实可能会说要保留 A 不变但要更改特征 B，这是矛盾的。可以通过报告所有反事实解释或通过制定标准评估反事实并选择最佳的反事实来解决这个多重反事实问题。

说到标准，我们如何定义一个好的反事实解释？首先，反事实解释的用户定义了实例预测中一个相关的更改，因此显而易见的第一个要求是，**反事实实例应尽可能紧密地产生预定义的预测**。并非总是可以完全匹配预定义的输出。在具有两个类 (罕见类和频繁类) 的分类设置中，模型可以始终将实例分类为频繁类。更改特征值以使预测的标签从频繁类别转换为罕见类别可能是困难的。因此，我们希望放宽反事实的预测输出必须与定义的结果完全对应的要求。在分类示例中，我们可以寻找一个反事实，罕见类别的预测概率增加到 10%，而不是当前的 2%。问题是，为了使预测概率从 2% 变为 10% (或接近 10%)，特征的最小变化是什么？**另一个标准是反事实应该与特征值实例尽可能相似**。这需要两个实例之间的距离度量。反事实不仅应接近原始实例，而且还应更改尽可能少的特征。这可以通过选择适当的距离度量来实现，比如曼哈顿距离。最后一个要求是，**反事实实例应具有可能的特征值**。对于房租为负数或房间数设置为 200 的租房示例，反事实解释是没有意义的。如果根据数据的联合分布产生反事实则更好，例如，一个有 10 个房间和 20 平方米的公寓不应视为反事实解释。

6.1.1 生成反事实解释

一种简单的产生反事实解释的方法是通过反复试验进行搜索。此方法涉及随机改变感兴趣实例的特征值，并在预期输出时停止。就像 Anna 想找一套她可以收取更高租金的公寓。但是有比反复试验更好的方法。首先，我们定义一个损失函数，该函数将感兴趣的实例、反事实和期望的 (反事实) 结果作为输入。损失度量反事实的预测结果与预定义结果之间的距离，以及反事实与感兴趣实例之间的距离。我们可以使用优化算法直接优化损失，也可以通过在实例周围进行搜索来优化损失，如 “Growing Spheres” 方法 (参见软件和替代方法)。

在本节中，我将介绍 Wachter 等人 (2017)[39] 建议的方法。他们建议尽量减少以下损失。

$$L(x, x', y', \lambda) = \lambda \cdot (\hat{f}(x') - y')^2 + d(x, x')$$

第一项是反事实 x' 的模型预测与期望结果 y' 之间的二次距离，用户必须事先定义。第二项是要解释的实例 x 与反事实 x' 之间的距离 d，稍后将对此进行详细说明。参数 λ 平衡预测距离 (第一项) 与特征值距离 (第二项)。对于给定的 λ 求解损失，并返回反事实 x'。较高的 λ 表示我们更喜欢接近期望结果 y' 的反事实，较低的值意味着我们更喜欢与特征值中的 x 非常相似的反事实 x'。如果 λ 非常大，则将选择预测最接近 y' 的实例，而不管其与 x 的距离如何。最终，用户必须决定如何在对反事实的预测与期望结果匹配的要求和对反事实类似于 x 的要求之间取得平衡。该方法的作者建议不要选择 λ 的值而是选择允许反事实实例的预测与 y' 相距多远的公差 ϵ。该约束可以写为：

$$|\hat{f}(x') - y'| \leq \epsilon$$

为了使该损失函数最小化，可以使用任何合适的优化算法，例如 Nelder-Mead。如果可以访问机器学习模型的梯度，则可以使用基于梯度的方法，例如 ADAM。必须预先设置实例 x，所需的输出 y'

和公差参数 ϵ。对于 x'，损失函数最小化，并且增大 λ 找到 (局部) 最佳反事实 x'，直到找到足够接近的解 (= 公差参数内)。

$$\arg\min_{x'} \max_{\lambda} L(x, x', y', \lambda)$$

用于测量实例 x 与反事实 x' 之间的距离的函数 d 是用绝对中位差 (MAD) 的倒数加权的曼哈顿距离。

$$d(x, x') = \sum_{j=1}^{p} \frac{|x_j - x'_j|}{MAD_j}$$

总距离是所有 p 个特征距离的总和，即实例 x 和反事实 x' 之间的特征值的绝对差。特征距离通过特征 j 在数据集上的绝对中位差的倒数来缩放：

$$MAD_j = \text{median}_{i \in \{1,\ldots,n\}}(|x_{i,j} - \text{median}_{l \in \{1,\ldots,n\}}(x_{l,j})|)$$

向量的中位数是向量值的一半大于而另一半小于的值。MAD 等效于特征的方差，但不是使用均值作为中心并求平方距离之和，而是使用中位数作为中心并求绝对距离之和。提出的距离函数相对于欧几里得距离具有优势，它引入了稀疏性。这意味着当不同的特征较少时，两个点彼此靠近。而且对于异常值更鲁棒。使用 MAD 进行缩放对于使所有特征达到相同的比例是必要的——无论你以平方米还是平方英尺来衡量公寓的面积大小都没有关系。

产生反事实的方法很简单:

1. 选择要解释的实例 x、所需的结果 y'、公差 ϵ 和低的 λ 初始值。
2. 采样一个随机的实例作为初始反事实。
3. 以初始采样的反事实为出发点，对损失进行优化。
4. 直到 $|\hat{f}(x') - y'| > \epsilon$：

   增加 λ。以当前反事实为出发点优化损失。返回最小化损失的反事实。
5. 重复步骤 2-4 并返回反事实列表或最小化损失的列表。

6.1.2 示例

这两个示例均来自 Wachter 等人 (2017) 的工作。

在第一个示例中，作者训练了一个三层的全连接神经网络来预测一个学生在法学院的第一年的平均成绩，基于在法学院之前的平均绩点 (GPA)，种族 (Race) 和法学院入学考试分数 (LSAT)。目的是为每个学生找到反事实的解释，以回答以下问题：如何改变输入特征，以使预测分数为 0? 由于之前已经对分数进行了归一化，因此分数为 0 的学生与学生的平均水平一样好。负分数表示低于平均水平的结果，正分数表示高于平均水平的结果。

下表显示了学习到的反事实：

Score	GPA	LSAT	Race	GPA x'	LSAT x'	Race x'
0.17	3.1	39.0	0	3.1	34.0	0
0.54	3.7	48.0	0	3.7	32.4	0
-0.77	3.3	28.0	1	3.3	33.5	0
-0.83	2.4	28.5	1	2.4	35.8	0
-0.57	2.7	18.3	0	2.7	34.9	0

第一列包含预测分数，接下来三列为原始特征值，最后三列为反事实特征值 (可以导致预测分数接近 0)。前两行是学生的平均成绩高于平均水平，其余三行是学生低于平均水平。前两行的反事实描述了如何改变学生的特征以降低预测分数，而对于其他三种情况，则描述了如何改变以将分数提高至平均值。增加分数的反事实总是将种族从黑色 (用 1 编码) 更改为白色 (用 0 编码)，这显示了模型的种族偏见。GPA 在反事实方面没有发生变化，但 LSAT 发生了变化。

第二个例子显示了预测糖尿病风险的反事实解释。经过训练的三层全连接神经网络可以根据年龄、身体质量指数 (BMI)、怀孕次数等来预测糖尿病风险。反事实回答了这个问题：必须更改哪些特征值才能将糖尿病的风险评分增加或降低到 0.5？发现了以下反事实：

- 人 1：如果你的 2 小时血清胰岛素水平为 154.3，则你的得分为 0.51
- 人 2：如果你的 2 小时血清胰岛素水平为 169.5，则你的得分为 0.51
- 人 3：如果你的血浆葡萄糖浓度为 158.3，而你的 2 小时血清胰岛素水平为 160.5，则你的得分为 0.51

6.1.3 优点

反事实解释的解释很清楚。如果实例的特征值根据反事实而更改，则预测将更改为预定义的预测。这里没有额外的假设，也没有魔法的背景。这也意味着它不像 LIME 那样危险，因为我们不清楚我们能在多大程度上推断出局部解释模型。

反事实方法创建了一个新实例，但是我们也可以通过报告哪些特征值已改变来总结一个反事实。这给了我们**两种报告结果的选择**。你可以报告反事实实例，也可以突出显示在感兴趣的实例和反事实实例之间改变了哪些特征。

反事实方法不需要访问数据或模型。它只需要访问模型的预测函数，例如，该函数也可以通过 Web API 来运行。这对于经第三方审计或向用户提供解释但未披露模型或数据的公司具有吸引力。由于商业秘密或数据保护的原因，公司有保护模型和数据的利益。反事实解释在解释模型预测和保护模型所有者的利益之间取得了平衡。

该方法还**适用于不使用机器学习的系统**。我们可以为接收输入并返回输出的任何系统创建反事实。预测公寓租金的系统也可以包含手写规则，反事实解释仍然有效。

反事实解释方法**相对容易实现**，因为它本质上是一种损失函数，可以使用标准优化程序库进行优化。必须考虑一些其他细节，例如将特征值限制在有意义的范围内 (例如，只有正的公寓面积大小)。

6.1.4 缺点

对于每个实例，通常会找到多个反事实的解释 (罗生门效应)。这很不方便——大多数人都喜欢简单的解释，而不是现实世界的复杂性。这也是一个实际的挑战。假设我们为一个实例生成了 23 个反事实解释。我们都报告吗？只选最好的吗？如果它们都是相对 “好” 但又非常不同怎么办？对于每个项目，必须重新回答这些问题。进行多种反事实的解释也可能是有利的，因为然后人类可以选择与他们先验知识相对应的解释。

对于给定的公差 ϵ，不能保证找到反事实的实例。那不一定是该方法的问题，而是取决于数据。

不能很好地处理具有许多不同级别的分类特征。该方法的作者建议针对分类特征的特征值的每种组合分别运行该方法，但是如果你拥有多个具有多个值的分类特征，这将导致组合爆炸。例如，具有 10 个级别的 6 个分类特征将意味着进行一百万次运行。Martens 等人 (2014)[40] 提出了仅针对分类特征的解决方案。在 Python 的 Alibi 包中实现了一种解决方案，该方案可以使用产生分类变量的扰动的方式处理数值变量和分类变量。

6.1.5 软件和替代方法

在 Python Alibi 包中实现了对反事实的解释。软件包的作者实现了 简单的反事实方法，以及使用类原型提高算法输出的可解释性和收敛性的 扩展方法[41]。

Martens 等人 (2014) 提出了一种非常相似的方法解释文档分类。在他们的工作中，他们专注于解释为什么文档被归类为特定类或者不被归类为特定类。与本节介绍的方法的不同之处在于 Martens 等人 (2014) 着重于文本分类器，它们以单词的出现作为输入。

搜索反事实的另一种方法是 Laugel 等人 (2017)[42] 的 Growing Spheres 算法。该方法首先在感兴趣点周围绘制一个球体，再对该球体内的点进行采样，检查其中一个采样点是否产生所需的预测，并相应地收缩或扩张球体，直到找到 (稀疏的) 反事实并最终返回。他们在论文中没有使用 “反事实” 一词，但是方法非常相似。他们还定义了一个损失函数，该函数支持反事实，其特征值的变化应尽可能少。他们建议直接使用球体进行上述搜索，而不是直接优化函数。

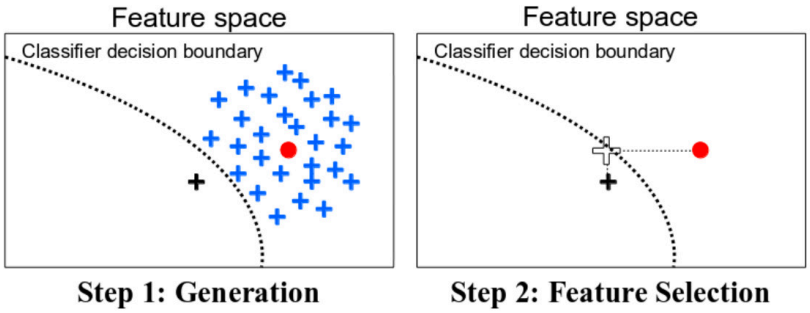


图 6.2. Growing Spheres 算法和选择反事实。

Ribeiro 等人 (2018)[43] 的锚与反事实相反。锚回答以下问题：哪些特征足以锚定预测，即改变其他特征不能更改预测？一旦找到可以用作预测锚的特征，我们将不再通过更改锚中未使用的特征来找到反事实实例。

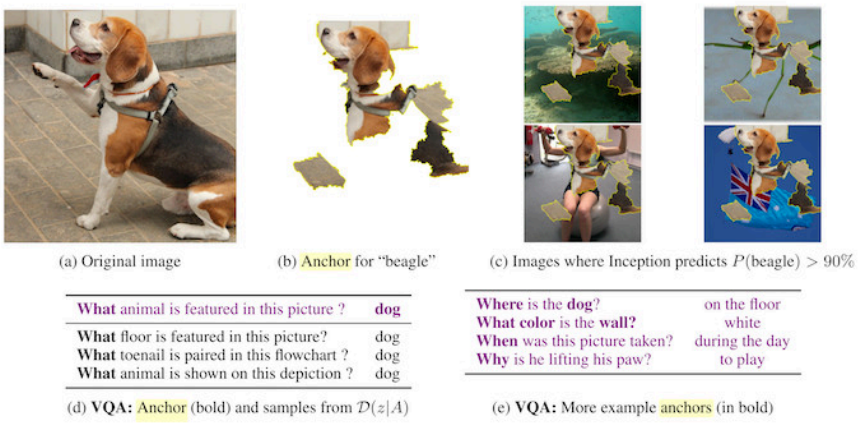


图 6.3. 锚的样例。

6.2 对抗样本

对抗样本 (`Adversarial Examples`) 是指当对一个样本的某一个特征值作出一个微小的变化而使得整个模型作出一个错误的预测。我建议先阅读有关反事实解释的小节，因为这些概念非常相似。对抗样本是反事实实例，旨在欺骗模型而不是解释模型。

为什么我们对对抗样本感兴趣？它们难道不只是机器学习模型的有趣副产品，没有实际意义吗？答案是“否”。对抗样本使机器学习模型容易受到攻击，如以下情况所示。

自动驾驶汽车撞向另一辆汽车，因为它无视停车标志。有人在标志上放了一张图片，从人眼来看它

看起来像是停车标志上粘了点 “灰尘”，但对标志识别软件而言它是禁止停车标志。

垃圾邮件检测程序没有将电子邮件分类为垃圾邮件。垃圾邮件设计成类似于普通电子邮件，但目的是欺骗收件人。

使用机器学习的扫描仪在机场扫描行李箱。为了避免被发现，人们发明了一种刀具，让系统认为它是一把雨伞。

让我们看一下创建对抗样本的一些方法。

6.2.1 方法与示例

有许多创建对抗样本的技术。大多数方法建议将对抗样本与要操纵的样本之间的距离最小化，同时将预测转换为期望的 (对抗性的) 结果。有一些方法需要访问模型的梯度，而这些方法当然仅适用于基于梯度的模型 (例如神经网络)，而其他方法仅需要访问预测函数，这使得这些方法与模型无关。本节中的方法重点在于具有深度神经网络的图像分类器，因为在该领域已进行了大量研究，而对抗图像的可视化具有很强的教育意义。图像的对抗样本是带有故意扰动像素的图像，目的是在应用期间欺骗模型。这些样本令人印象深刻地表明，人眼来看无害的图像可以多么容易地欺骗用于目标识别的深度神经网络。如果你尚未看到这些样本，你可能会感到惊讶，因为对于人类观察者来说，预测的变化是无法理解的。对抗样本对机器而言就像光学幻像。

我的狗出了点问题

Szegedy 等人 (2013)[44] 在他们的工作 “Intriguing Properties of Neural Networks” 中使用了基于梯度的优化方法来寻找深度神经网络的对抗样本。

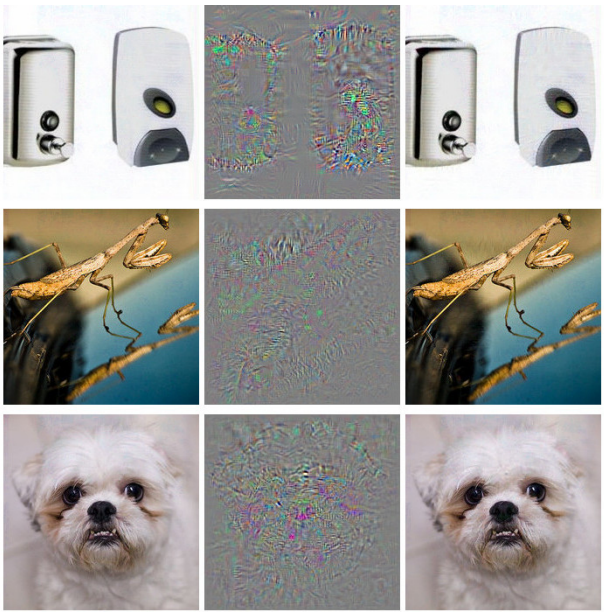


图 6.4. Szegedy 等人 (2013) 关于 Alexnet 的对抗样本。左列中的所有图像都已正确分类。中间一列显示添加到图像中的 (放大的) 公差 (或称扰动)，以生成右列中的图像，所有的都 (错误) 分类为“鸵鸟”。

这些对抗样本是通过最小化关于 r 的以下函数生成的：

$$loss(\hat{f}(x+r), l) + c \cdot |r|$$

在此公式中，x 是图像 (表示为像素向量)，r 是对像素的变化以创建对抗图像 ($x + r$ 生成新图像)，l 是期望的结果类别，参数 c 用于平衡图像之间的距离和预测之间的距离。第一项是对抗样本的预测结果与期望的类别 l 之间的距离，第二项是对抗样本与原始图像之间的距离。这个公式几乎与损失函数相同，可以得出反事实的解释。对于 r 还有其他限制，因此像素值保持在 0 到 1 之间。作者建议使用 box-constrained L-BFGS (一种适用于梯度的优化算法) 解决此优化问题。

干扰的熊猫：快速梯度符号方法 (Fast Gradient Sign Method)

Goodfellow 等人 (2014)[45] 发明了用于生成对抗图像的快速梯度符号方法。梯度符号方法使用底层模型的梯度来查找对抗样本。通过向每个像素添加或减去一个小的扰动 ϵ 来操纵原始图像 x。加或减 ϵ 取决于像素的梯度符号是正还是负。在梯度方向上增加扰动意味着图像被有意修改，从而导致模型分类失败。

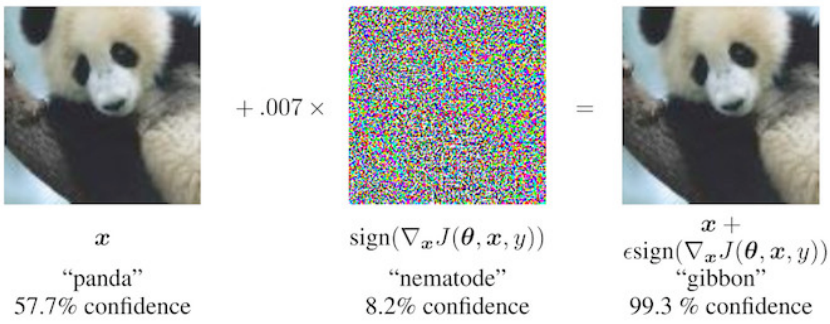


图 6.5. Goodfellow 等人 (2014) 在神经网络中使熊猫看起来像长臂猿。通过向原始熊猫像素 (左图) 添加小扰动 (中间图)，作者创建了一个对抗样本，它被归类为长臂猿 (右图)，但对人类而言却像熊猫。

以下公式描述了 FGSM 的核心：

$$x' = x + \epsilon \cdot sign(\nabla_x J(\theta, x, y))$$

其中 $\nabla_x J$ 是模型损失函数相对于原始输入像素向量 x 的梯度，y 是 x 的真实标签向量，而 θ 是模型参数向量。对于梯度向量 (与输入像素向量一样长)，我们需要符号：如果像素强度的增加会增加损失 (模型产生的误差)，梯度符号为正 (+1)；如果像素强度的减少会增加损失，则为负 (-1)。当神经网络线性处理输入像素强度和类别分数之间的关系时，就会出现此漏洞。特别是，倾向于线性的神经网络结构，如 LSTM、Maxout 网络、带有 ReLU 激活单元的网络或其他线性机器学习算法 (例如逻辑回归) 容易受到梯度符号方法的影响。攻击是通过外推 (`Extrapolation`) 进行的。输入像素强度和类别分数之间的线性关系导致容易受到异常值的影响，也就是说，可以通过将像素值移动到数据分布之外的区域来欺骗模型。我希望这些对抗样本非常特定于给定的神经网络体系结构。但是事实证明，你可以重复使用对抗样本来欺骗在同一任务上训练的不同体系结构的网络。

Goodfellow 等人 (2014) 建议在训练数据中添加对抗样本，以学习鲁棒的模型。

水母……不，等等。浴缸：单像素攻击 (`1-pixel attacks`)

Goodfellow 等人 (2014) 提出的方法要求更改许多像素，即使只是稍微改变一点。但是，如果你只能更改一个像素怎么办？你能欺骗机器学习模型吗？Su 等人 (2019)[46] 表明，实际上有可能通过更改单个像素来欺骗图像分类器。

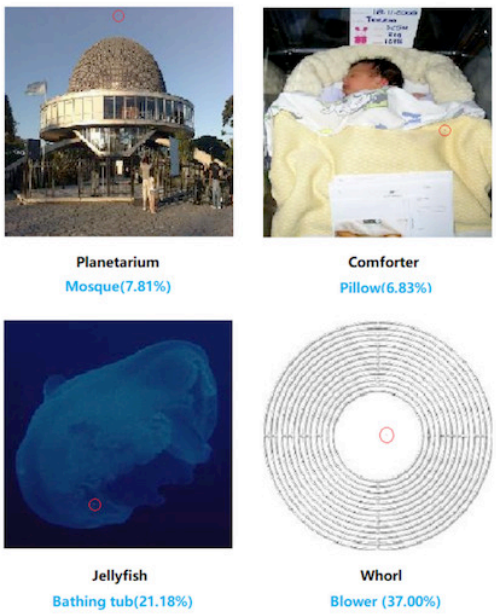


图 6.6. 通过故意改变一个像素 (用圆圈标记)，在 ImageNet 上训练的神经网络被欺骗，从而预测错误的类别而不是原始类别。Su 等人 (2019) 的研究。

与反事实类似，单像素攻击会寻找一个修改后的样本 x'，该样本与原始图像 x 接近，但将预测更改为对抗结果。但是，接近程度的定义不同：只有一个像素可以更改。单像素攻击使用差分进化 (`Differential Evolution`) 来找出要改变的像素以及改变方式。差分进化是受到物种生物进化的大致启发。个体群体 (称为候选解) 会一代一代地重组，直到找到解为止。每个候选解都对像素修改进行编码，并由五个元素的向量表示：x 坐标和 y 坐标以及红、绿和蓝 (RGB) 值。搜索从例如 400 个候选解 (= 像素修改建议) 开始，并使用以下公式从父代中生成新一代候选解 (子代)：

$$x_i(g+1) = x_{r1}(g) + F \cdot (x_{r2}(g) + x_{r3}(g))$$

其中每个 x_i 是候选解的元素 (x 坐标，y 坐标，红色，绿色或蓝色)，g 是当前代，F 是缩放参数 (设置为 0.5)，并且 $r1$，$r2$ 和 $r3$ 是不同的随机数。每个新的子候选解又是一个具有位置和颜色五个元素的像素，并且每个元素都是三个随机父像素的混合。

如果候选解之一是对抗样本 (这意味着它被归类为不正确的类)，或者达到了用户指定的最大迭代次数，则停止子代的创建。

一切都是烤面包机：对抗补丁 (`Adversarial Patch`)

我最喜欢的方法之一是将对抗样本带入现实。Brown 等人 (2017)[47] 设计了一个可打印的标签，该标签可以粘贴在目标旁边，使它们看起来像图像分类器的烤面包机。辉煌的工作!

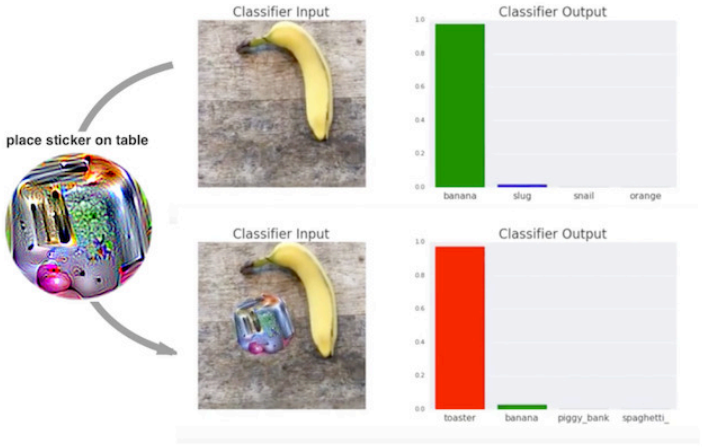


图 6.7. 一个贴纸，它使在 ImageNet 上训练的 VGG16 分类器将香蕉的图像分类为烤面包机。Brown 等人 (2017) 的工作。

该方法不同于到目前为止针对对抗样本提出的方法，因为消除了对抗图像必须非常接近原始图像的限制。取而代之的是，该方法用可以采用任何形状的补丁完全替换了图像的一部分。补丁的图像针对不同的背景图像进行了优化，而且补丁在图像上的位置不同，有时还会移动、会变大或变小以及旋转，以使补丁在许多情况下都可以工作。最后，将优化后的图像打印出来，并用它来欺骗图像分类器。

即使你的计算机认为这是个好主意，也切勿将 3D 打印的海龟卷入枪战中：鲁棒的对抗样本 (`Robust Adversarial Examples`)

接下来的方法实际上是在烤面包机上增加另一个维度：Athalye 等人 (2017)[48]3D 打印了一只乌龟，从几乎所有可能的角度来看，深度神经网络都觉得它像一把步枪。是的，你没看错。对人类而言，像乌龟一样的物理物体对计算机而言就像步枪！

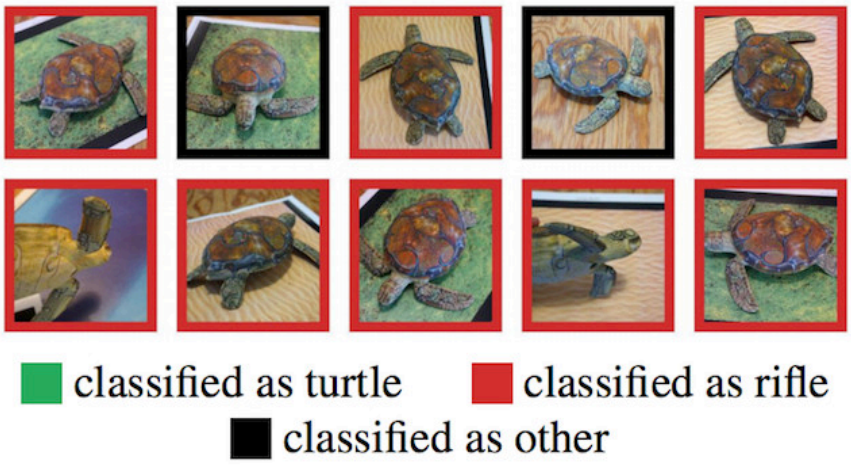


图 6.8. 3D 打印的乌龟，被 TensorFlow 标准预训练 InceptionV3 分类器识别为步枪。Athalye 等人 (2017) 的工作。

作者发现了一种方法，可以为 2D 分类器创建 3D 对抗样本，该样本在转换方面具有对抗性，比如旋转乌龟、放大等等的所有可能。当图像旋转或视角改变时，其他方法 (例如快速梯度方法) 将不再起作用。Athalye 等人 (2017) 提出了 Expectation Over Transformation (EOT) 算法，这是一种生成对抗样本的方法，该样本甚至在图像转换后也可以工作。EOT 背后的主要思想是在许多可能的转换中优化对抗样本。EOT 不是对抗样本与原始图像之间的距离最小化，而是在给定选定的可能转换分布下，将两者之间的期望距离保持在某个阈值以下。转换后的期望距离可表示为：

$$\mathbb{E}_{t\sim T}[d(t(x'), t(x))]$$

其中 x 是原始图像，$t(x)$ 是转换后的图像 (例如旋转图片)，x' 是对抗样本，$t(x')$ 是其转换后的版本。除了处理转换分布外，EOT 方法遵循熟悉的模式，将搜索对抗样本框架化为优化问题。我们尝试找到一个对抗样本 x'，该样本在可能的变换 T 的分布范围内最大化选定类别 y_t (例如 “步枪”) 的概率：

$$\arg\max_{x'} \mathbb{E}_{t\sim T}[logP(y_t|t(x'))]$$

约束条件是对抗样本 x' 和原始图像 x 之间所有可能的转换上的期望距离都保持在某个阈值以下：

$$\mathbb{E}_{t\sim T}[d(t(x'), t(x))] < \epsilon \quad \text{and} \quad x \in [0,1]^d$$

我认为我们应该关注此方法所带来的可能性。其他的方法有基于数字图像的处理。但是，这些 3D 打印的，鲁棒的对抗样本可以插入任何真实场景中，并欺骗计算机错误地对目标进行分类。让我们扭转一下：如果有人制造了一只看起来像乌龟的步枪怎么办?

蒙蔽的对手：黑盒攻击 (Black Box Attack)

想象以下情况：我使你可以通过 Web API 访问我的出色的图像分类器。你可以从模型中获取预测，但无权访问模型参数。你发送数据，然后我的服务回答相应的分类。大多数对抗攻击都不是在这种

情况下设计的，因为它们需要访问底层深度神经网络的梯度才能找到对抗样本。Papernot 及其同事 (2017)[49] 表明，无需内部模型信息且无需访问训练数据即可创建对抗样本。这种 (几乎) 零知识攻击称为黑盒攻击。

工作原理：

1. 从与训练数据来自同一域的一些图像开始，例如，如果要攻击的分类器是数字分类器，则使用数字图像。领域知识是必需的，但不需要访问训练数据。
2. 从黑盒获得当前图像集的预测。
3. 在当前图像集上训练代理模型 (例如神经网络)。
4. 使用启发式方法创建一组新的合成图像，该方法检查当前图像集，在哪个方向操作像素以使模型输出具有更大的方差。
5. 基于预定义的迭代次数重复步骤 2 到 4。
6. 使用快速梯度方法 (或类似方法) 为代理模型创建对抗样本。
7. 用对抗样本攻击原始模型。

代理模型的目的是近似黑盒模型的决策边界，但不一定要达到相同的精度。

作者通过攻击在各种云机器学习服务上训练过的图像分类器来测试这种方法。这些服务在用户上传的图像和标签上训练图像分类器。该软件会自动训练模型 (有时会使用用户不知道的算法) 并进行部署。然后分类器对上传的图像进行预测，但是模型本身无法检查或下载。作者能够为不同的提供者找到对抗样本，其中多达 84% 的对抗样本被错误分类。

如果要欺骗的黑盒模型不是神经网络，该方法也能工作。这包括没有梯度的机器学习模型，例如决策树。

6.2.2 网络安全视角

机器学习处理已知情况下的未知：从已知的分布中预测未知的数据点。防御攻击可应对未知情况下的未知：从未知的对抗输入分布中可靠地预测未知的数据点。随着机器学习被集成到越来越多的系统中，例如自动驾驶汽车或医疗设备，它们也成为攻击的切入点。即使测试数据集上机器学习模型的预测是 100% 正确的，也可以找到对抗样本来欺骗该模型。防御网络攻击的机器学习模型是网络安全领域的新组成部分。

Biggio 等人 (2018)[50] 对十年来关于对抗性机器学习的研究进行了很好的回顾，本节以此为基础。网络安全是攻击者和防御者一次又一次地胜过对手的军备竞赛。

网络安全中有三个黄金法则：1) 了解你的对手 2) 主动 3) 保护自己。

不同的应用程序有不同的对手。试图通过电子邮件骗取他人钱财的人是电子邮件服务提供商和用户的敌对方。提供商希望保护其用户，以便他们可以继续使用其邮件程序，攻击者希望使人们给他们钱。了解你的对手意味着了解他们的目标。假设你不知道这些垃圾邮件发送者的存在，并且对电子邮件服务的唯一滥用是发送盗版音乐，那么防御方式将是不同的 (例如，扫描附件中的受版权保护的材料，而不是分析文本中的垃圾邮件指标)。

“主动” 意味着积极地测试和识别系统的弱点。当你主动尝试用对抗样本欺骗模型然后防御它们时，你会变得很主动。使用解释方法来了解哪些特征很重要以及特征如何影响预测，也是了解机器学习模型的弱点的主动步骤。作为数据科学家，你是否会在这个危险的世界中信任你的模型？你是否分析了模型在不同情况下的行为，确定了最重要的输入，并检查了一些样本的预测解释？你是否尝试寻找对抗输入？机器学习模型的可解释性在网络安全中起着重要作用。被动，是与主动相反，意味着等待着直到系统被攻击，然后才理解问题并安装一些防御措施。

我们如何保护我们的机器学习系统免受对抗样本的攻击？主动的方法是使用对抗样本对分类器进行迭代重新训练，也称为对抗训练。其他方法基于博弈论，例如学习特征的不变转换或鲁棒优化 (正则化)。另一种建议的方法是使用多个分类器，而不是一个，并让它们对预测进行投票 (集成)，但这并不能保证有效，因为它们都可能遭受类似的对抗样本。另一种效果也不佳的方法是梯度掩模 (`Gradient Masking`)，它通过构建没有用梯度的模型，例如使用最近邻分类器而不是原始网络模型。

我们可以通过攻击者对系统的了解程度来区分攻击类型。攻击者可能具有完备的知识 (白盒攻击)，这意味着他们知道有关模型的所有信息，例如模型的类型、参数和训练数据。攻击者可能具有部分知识 (灰盒攻击)，这意味着他们可能只知道特征表示和使用的模型类型，而无法访问训练数据或参数；攻击者可能具有零知识 (黑盒攻击)，这意味着他们只能以黑盒方式查询模型，而无法访问训练数据或有关模型参数的信息。根据信息级别，攻击者可以使用不同的技术来攻击模型。正如我们在样本中看到的，即使在黑盒中，也可以创建对抗样本，因此隐藏关于数据和模型的信息不足以抵御攻击。

考虑到攻击者和防御者之间的猫捉老鼠游戏的性质，我们将在这一领域看到很多发展和创新。试想一下不断演变出许多不同类型的垃圾邮件。针对机器学习模型提出新的攻击方法，并针对这些新攻击提出了新的防御措施。不断开发出更强大的攻击来逃避最新的防御等等。在本节中，我希望使你意识到对抗样本的问题，并且只有通过积极研究机器学习模型，我们才能发现并弥补弱点。

6.3 原型与批评

一个原型是一个数据实例，它是所有数据的代表。一个批评是不能由一组原型很好地代表的一个数据实例。批评的目的是与原型一起提供见解，尤其是对于原型不能很好代表的数据点。原型和批评 (`Prototypes and Criticisms`) 可以 (与机器学习) 模型无关地用于描述数据，但是它们也可以用于创建可解释的模型或使黑盒模型可解释。

在本节中，我使用“数据点”一词来指代单个实例，以强调实例也是坐标系统中的一个点，其中每个特征都是一个维度。下图显示了一个模拟的数据分布，其中一些实例被选择为原型，另一些被作为批评。小点是数据，大点是原型，大方块是批评。选择原型 (手动) 以覆盖数据分布的中心，并且批评是没有原型的簇中的点。原型和批评总是来自数据的实际实例。

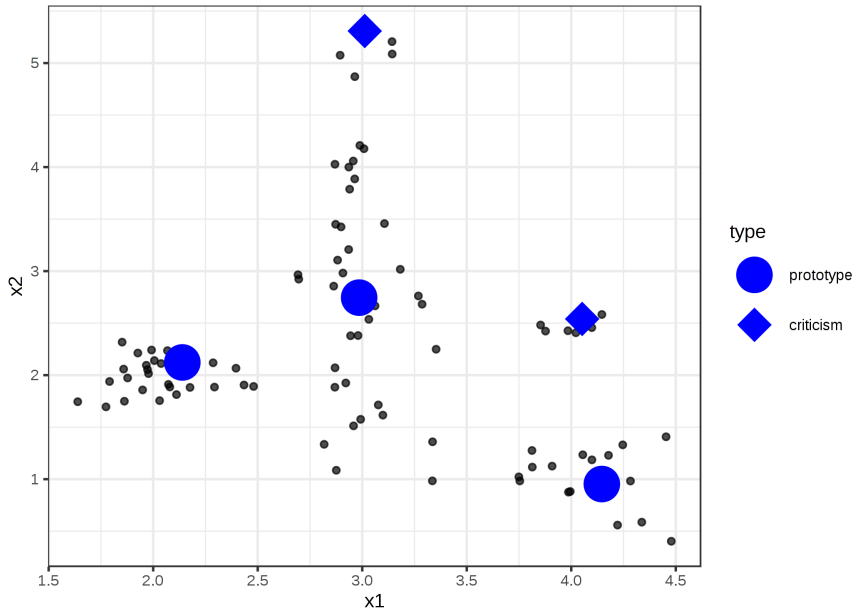


图 6.9. 具有两个特征 x_1 和 x_2 的数据分布的原型和批评。

我手动选择了原型，这并不能很好地扩展，可能会导致结果不佳。在数据中找到原型的方法很多。其中之一是 k-medoids，这是与 k-means 算法相关的聚类算法。任何返回实际数据点作为聚类中心的聚类算法都有资格选择原型。但是这些方法大多数都只找到原型，而没有批评。本节介绍 Kim 等人 (2016)[4] 的 MMD-critic，一种将原型和批评结合在一个框架中的方法。

MMD-critic 比较数据的分布和所选原型的分布。这是理解 MMD-critic 方法的中心概念。MMD-critic 选择的原型可以最大程度地减少两个分布之间的差异。高密度区域中的数据点是很好的原型，尤其是从不同的“数据簇”中选择数据点时。来自原型不能很好解释的区域的数据点被选择为批评。

让我们深入研究该理论。

6.3.1 理论

MMD-critic 程序可以概括如下：

1. 选择要查找的原型和批评数量。
2. 通过贪婪搜索找到原型。选择原型，以使原型的分布接近数据分布。
3. 通过贪婪搜索找到批评。当原型的分布不同于数据的分布时，选择点作为批评。

我们需要一些因素来找到 MMD-critic 数据集的原型和批评。作为最基本的成分，我们需要一个**核函数**来估计数据密度。核是根据接近度对两个数据点加权的函数。基于密度估计，我们需要一种度量来告诉我们两种分布有何不同，以便我们可以确定我们选择的原型的分布是否接近数据分布。这可以通过测量**最大平均差异 (Maximum Mean Discrepancy)** 来解决。同样基于核函数，我们需要**witness 函数**告诉我们在特定数据点上两种分布有何不同。通过 witness 函数，我们可以选择批评，即原型和数据的分布不同且 witness 函数具有较大绝对值的数据点。最后一个成分是寻找好的原型和批评的搜索策略，可以通过简单的**贪婪搜索**来解决。

让我们从**最大平均差异 (MMD)** 开始，它衡量两个分布之间的差异。原型的选择创建了原型的密度分布。MMD-critic 的目的是最小化选择的原型分布和数据分布之间的差异。我们要通过用核密度函数进行估计的经验分布，评估原型分布与数据分布是否不同。最大平均差异度量两个分布之间的差异，这是根据两个分布的期望差在函数空间上的极值。清楚了吗？就我个人而言，当我看到数据是如何计算的时候，我会更好地理解这些概念。以下公式显示了如何计算平方的 MMD 量度 (MMD2)：

$$MMD^2 = \frac{1}{m^2}\sum_{i,j=1}^{m} k(z_i, z_j) - \frac{2}{mn}\sum_{i,j=1}^{m,n} k(z_i, x_j) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(x_i, x_j)$$

k 是一个核函数，用于测量两点的相似性，稍后将对此进行更多介绍。m 是原型 z 的数量，n 是原始数据集中的数据点 x 的数量。原型 z 是数据点 x 的选择。每个点都是多维的，即可以具有多个特征。MMD-critic 的目标是最小化 MMD2。MMD2 越接近零，原型的分布拟合数据越好。将 MMD2 降低为零的关键是中间的项，该项计算原型与所有其他数据点之间的平均接近度 (乘以 2)。如果此项加起来等于第一项 (原型彼此之间的平均接近度) 加上最后一项 (数据点彼此之间的平均接近度)，则原型可以完美地解释数据。尝试一下如果你使用所有 n 个数据点作为原型，公式会发生什么变化。

下图说明了 MMD2 度量。第一张图显示了具有两个特征的数据点，其中数据密度的估计以阴影背景显示。其他每个图都显示了不同的原型选择，以及图标题中的 MMD2 度量。原型是大点，其分布显示为等高线。选择最能覆盖这些情况下数据的原型 (左下) 具有最低的差异值。

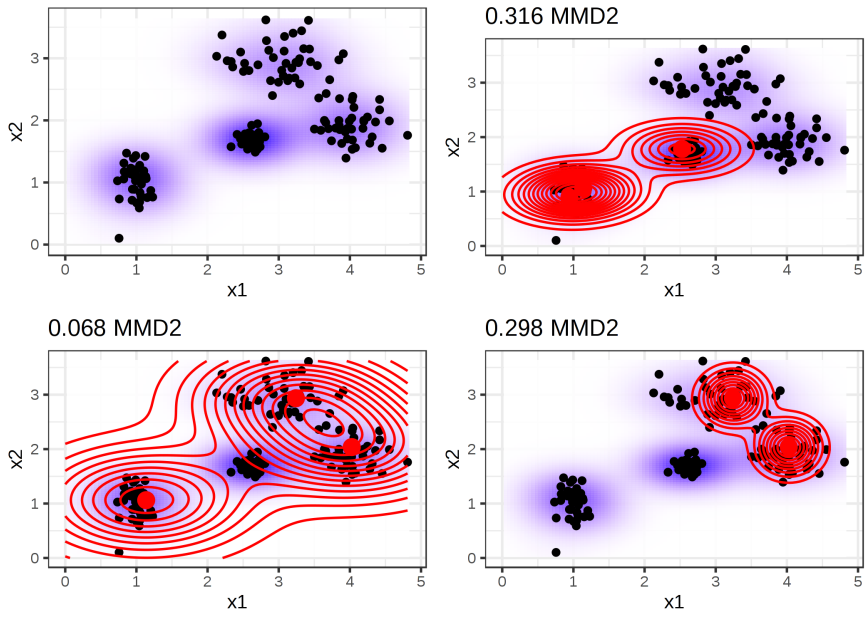


图 6.10. 具有两个特征和不同原型选择的数据集的最大平均差异的平方 (MMD2) 度量。

核的一种选择是径向基函数核：

$$k(x, x') = exp\left(-\gamma||x - x'||^2\right)$$

其中 $||x - x'||^2$ 是两点之间的欧式距离，而 γ 是缩放参数。核的值随两点之间的距离而减小，范围在 0 和 1 之间：当两点无限远时为 0；当两点相等时为 1。

我们将 MMD2 度量，核和贪婪搜索结合在一种算法中，以查找原型：

- 从空的原型列表开始。
- 当原型数量低于所选数量 m 时：
  - 对于数据集中的每个点，检查将点添加到原型列表后 MMD2 减少了多少。将最小化 MMD2 的数据点添加到列表中。
- 返回原型列表。

找到批评是用 witness 函数，该函数告诉我们在特定点上两个密度估计值相差多少。可以使用以下方法估计：

$$witness(x) = \frac{1}{n}\sum_{i=1}^{n} k(x, x_i) - \frac{1}{m}\sum_{j=1}^{m} k(x, z_j)$$

对于两个具有相同特征的数据集，witness 函数为你提供了一种评估 x 点更适合哪种经验分布的方法。为了发现批评，我们在正向和负向上寻找 witness 函数的极值。witness 函数中的第一项是点 x 与数据之间的平均接近度，第二项是点 x 与原型之间的平均接近度。如果点 x 的 witness 函数接近于零，则数据和原型的密度函数接近在一起，这意味着在点 x 处原型的分布类似于数据的分布。

x 点处的负 witness 函数意味着原型分布会高估数据分布 (例如，如果我们选择原型，但附近只有很少的数据点)；x 点处的正 witness 函数意味着原型分布低估了数据分布 (例如，如果 x 周围有很多数据点，但我们附近没有选择任何原型)。

为了给你带来更多的直觉，让我们预先以最低的 MMD2 重用图中的原型，并显示一些手动选择的点的 witness 函数。下图中的标签显示了标记为三角形的各个点的 witness 函数的值。只有中间的一个点具有很高的绝对值，因此是进行批评的很好的候选者。

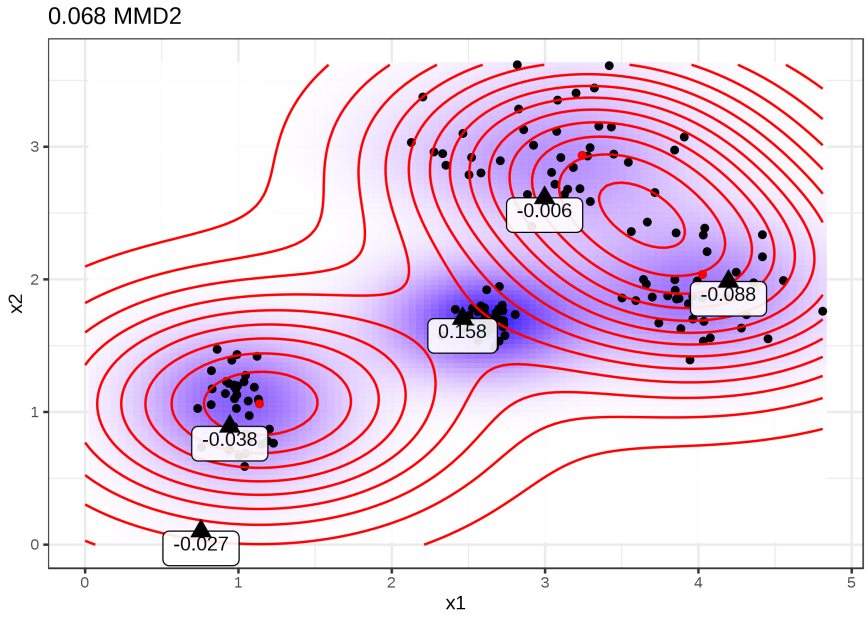


图 6.11. 不同点 witness 函数的评估。

witness 函数允许我们显式搜索原型无法很好地表示的数据实例。批评是 witness 函数中具有很高绝对值的点。像原型一样，也可以通过贪婪搜索找到批评。但是，我们不是在减少总体 MMD2，而是在寻找能够最大化损失函数 (包括 witness 函数和正则项) 的点。优化函数中的附加项会增强点的多样性，这是必需的，以便点来自不同的簇。

第二步独立于找到原型的过程。我也可以亲自挑选一些原型，并使用此处描述的过程来学习批评。或者，原型可以来自任何聚类过程，例如 k-medoids。

这就是 MMD-critic 理论的重要部分。一个问题仍然存在：**如何将 MMD-critic 用于可解释的机器学习**？

MMD-critic 可以通过三种方式增加可解释性：通过帮助更好地理解数据分布；通过建立可解释的模型；通过使黑盒模型可解释。

如果将 MMD-critic 应用于数据以查找原型和批评，这将增进你对数据的理解，尤其是当你有边缘

情况的复杂数据分布时。但是有了 MMD-critic，你可以实现更多!

例如，你可以创建一个可解释的预测模型：所谓的 “最近原型模型” (`Nearest Prototype Model`)。预测函数定义为：

$$\hat{f}(x) = argmax_{i \in S} k(x, x_i)$$

这意味着我们从原型集合 S 中选择最接近新数据点的原型 i，因为它产生了核函数的最大值。原型本身将作为预测的解释返回。该过程具有三个调整参数：核类型，核缩放参数和原型数量。可以在交叉验证循环中优化所有参数。这种方法不使用批评。

作为第三个选择，我们可以使用 MMD-critic，通过检查原型和批评及其模型预测，使全局范围内的任何机器学习模型都可解释。步骤如下：

1. 查找 MMD-critic 的原型和批评。
2. 照常训练机器学习模型。
3. 使用机器学习模型预测原型和批评的结果。
4. 分析预测：在哪些情况下算法错误？现在，你有一些很好地表示数据的实例，可以帮助你发现机器学习模型的弱点。

这有什么用呢？还记得 Google 的图像分类器将黑人识别为大猩猩吗？也许他们应该在部署图像识别模型之前使用这里描述的过程。仅仅检查模型的性能是不够的，因为如果正确率是 99%，则此问题可能仍然是 1%。而且标签也可能是错误的！检查所有训练数据并遍历所有训练数据并执行完整性检查 (如果预测有问题) 可能会发现问题，但不可行。但是，选择 (例如几千个) 原型和批评是可行的，并且可能揭示了数据问题：这可能表明缺乏皮肤黝黑的人的图像，这表明存在数据集多样性的问题。它可能会显示一个或多个皮肤黝黑的人的图像作为原型，或者 (可能) 以臭名昭著的 “大猩猩” 类别显示为批评。我不保证 MMD-critic 肯定会拦截此类错误，但这是一个很好的完整性检查。

6.3.2 示例

我从 MMD-critic 论文中摘取了示例。这两个应用程序都基于图像数据集。每个图像都由 2048 维的图像嵌入表示。图像嵌入是一个带有数字的向量，这些数字捕获图像的抽象属性。嵌入向量通常是从神经网络中提取的，这些神经网络经过训练可以解决图像识别任务，在这种情况下为 ImageNet 挑战。使用这些嵌入向量计算图像之间的核距离。

第一个数据集包含与 ImageNet 数据集不同的犬种。MMD-critic 应用于来自两个犬种类别的数据。左边的狗，原型通常会显示狗的脸，而批评则是没有狗脸或不同颜色 (例如黑白) 的图像。在右侧，原型包含狗的户外图像。批评包括穿着服装的狗和其他不寻常情况的狗。

图 6.12. 来自 ImageNet 数据集的两种类别狗的原型和批评。

MMD-critic 的另一个示例是使用手写数字数据集。

查看实际的原型和批评，你可能会注意到每个数字的图像数量是不同的。这是因为在整个数据集中搜索了固定数量的原型和批评，而不是每个类都使用固定数量的原型和批评。正如预期的那样，原型展示了不同的数字书写方式。这些批评包括带有异常的粗线或细线的实例，也有无法识别的数字。

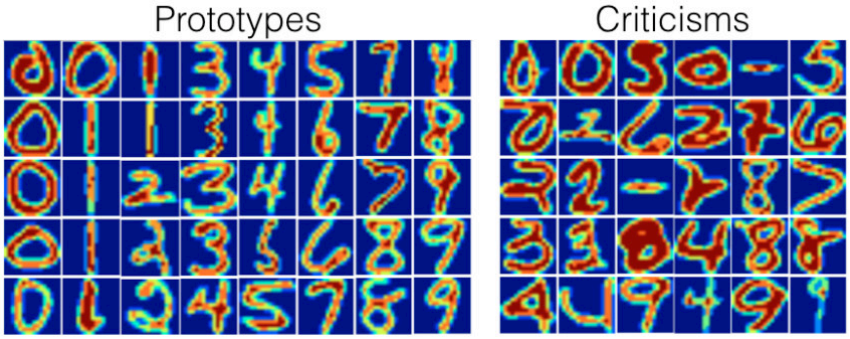


图 6.13. 手写体数字的原型和批评。

6.3.3 优点

在一项用户研究中，MMD-critic 的作者向参与者提供了图像，他们必须在视觉上匹配两组图像中的一个，每组图像代表两个类别之一 (例如，两个犬种)。**当展示原型和批评而不是一个类的随机图像时，参与者表现得最好**。

你可以**自由选择原型和批评的数量**。

MMD-critic 使用数据的密度估计。这适用于**任何类型的数据和任何类型的机器学习模型**。

该算法易于实现。

MMD-critic 在提高解释性方面非常灵活。它可以用来了解复杂的数据分布。它可用于构建可解释的机器学习模型。或者它可以阐明黑盒机器学习模型的决策。

找到批评是独立于原型的选择过程。但是根据 MMD-critic 选择原型是有意义的，因为这样一来，使用相同的比较原型和数据密度的方法来创建原型和批评。

6.3.4 缺点

在数学上，原型和批评的定义不同，但它们的区别是**基于截断值** (原型的数量)。假设你选择的原型数量太少，无法覆盖数据分布。批评最终将集中在没有得到很好解释的区域。但是，如果要添加更多原型，它们也将最终出现在相同的区域。任何解释都必须考虑到批评在很大程度上取决于现有原型和原型数量的 (任意) 截断值。

你**必须选择原型和批评的数量**。尽管这很不错，但它也是一个缺点。我们实际上需要多少原型和批评？越多越好？越少越好？一种解决方案是通过测量人类在查看图像上有多少时间来选择原型和批评的数量，这取决于特定的应用程序。只有当使用 MMD-critic 构建分类器时，我们才可以直接对其进行优化。一种解决方案可能是在屏幕 x 轴上显示原型数量，在 y 轴上显示 MMD2 度量。我们将选择 MMD2 曲线变平的原型数量。

其他的参数是核的选择和核缩放参数。我们面临着与原型和批评的数量相同的问题：**我们如何选择核及其缩放参数**？同样，当我们使用 MMD-critic 作为最近原型分类器时，我们可以调整核参数。但是，对于 MMD-critic 的无监督使用情况，目前尚不清楚。(也许我在这里有点苛刻，因为所有无监督的方法都存在这个问题。)

它以所有特征为输入，而**忽略了某些特征可能与预测目标结果无关的事实**。一种解决方案是仅使用相关特征，例如图像嵌入而不是原始像素。只要我们有办法将原始实例投影到仅包含相关信息的表示形式上，它就可以工作。

有一些可用的代码，但是**它还没有被很好地打包成文档化的软件来实现**。

6.3.5 代码和替代方法

可以在以下位置找到 MMD-critic 的实现：https://github.com/BeenKim/MMD-critic。

寻找原型的最简单替代方案是 Kaufman 等人 (1987)[51] 的 k-medoids。

6.4 有影响力的实例

机器学习模型最终是训练数据的产物，删除其中一个训练实例可能会影响生成的模型。当训练实例从训练数据中删除后，会大大改变模型的参数或预测，因此我们将这个实例称为 “有影响力的”。通过识别有影响力的训练实例，我们可以 “调试” 机器学习模型，并更好地解释它们的行为和预测。

本节向你介绍两种识别有影响力的实例 (Influential Instances) 的方法，即删除诊断和影响函数。两种方法均基于稳健统计 (Robust Statistics)，稳健统计提供的统计方法受异常值或违反模型假设的影响较小。稳健统计还提供了一些方法来测量数据的稳健估计 (例如平均估计或预测模型的权重)。

假设你要估算城市中人们的平均收入，然后随机询问街上的 10 个人他们的收入。除了你的样本可能真的很差之外，你的平均收入估计能在多大程度上受到一个人的影响? 要回答这个问题，你可以通过忽略个别答案来重新计算平均值，或者通过 “影响函数” 从数学上推导是如何影响平均值的。使用删除方法，我们将重新计算平均值十次，每次都删除其中一份收入，并测量平均估计值的变化。一个大的变化意味着删去的实例具有很大的影响力。第二种方法通过无限小的权重来增加其中一个的权重，这对应于统计量或模型的一阶导数的计算。这种方法也称为 “无穷小方法” 或 “影响函数”。顺便说一句，结果是你的均值估计会受到单个值的强烈影响，因为均值与单个值呈线性比例关系。一个更可靠的选择是中位数 (该数值表示一半的人收入更高而另一半的人收入更低)，因为即使你样本中收入最高的人的收入要高十倍，所得的中位数也不会改变。

删除诊断和影响函数也可以应用于机器学习模型的参数或预测，以更好地了解其行为或解释单个预测。在介绍这两种寻找有影响力的实例的方法之前，我们将研究异常值和有影响力的实例之间的差异。

异常值

一个异常值 (Outlier，也称离群值) 是远离数据集中其他实例的一个实例。“远离” 表示到所有其他实例的距离 (例如，欧几里得距离) 非常大。在新生儿数据集中，体重 6 公斤的新生儿将被视为异常值。在支票账户为主的银行账户数据集中，专用贷款账户 (负余额大，交易少) 将被视为异常值。下图显示了一维分布的离群值。

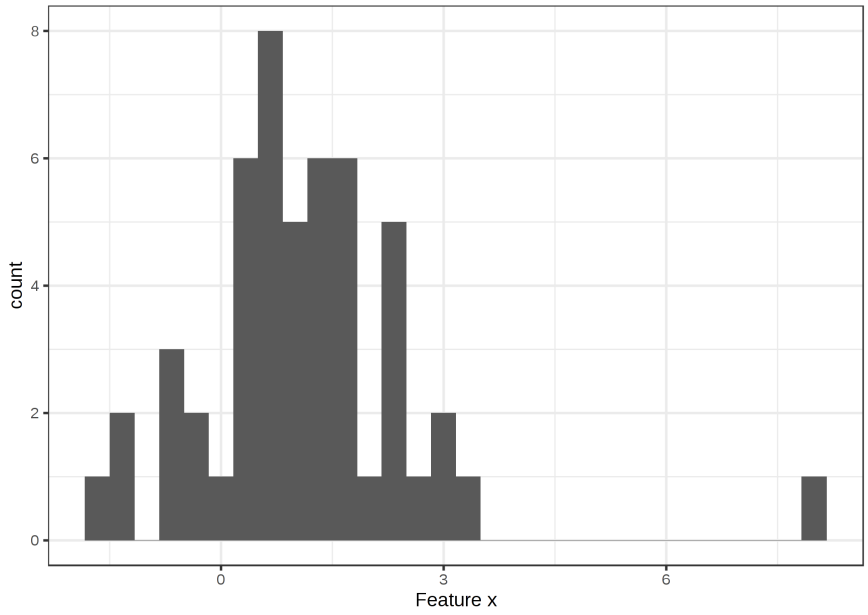


图 6.14. 特征 x 遵循高斯分布，x=8 处有一个异常值。

异常值可能是有趣的数据点 (例如，批评)。当异常值影响模型时，它也是有影响力的实例。

有影响力的实例

有影响力的实例是数据实例，其删除对训练模型有很大影响。在从训练数据中删除特定实例后对模型进行重新训练时，模型参数或预测变化越大，该实例的影响力就越大。实例是否对经过训练的模型有影响还取决于实例对目标 y 的值。下图显示了线性回归模型的有影响力的实例。

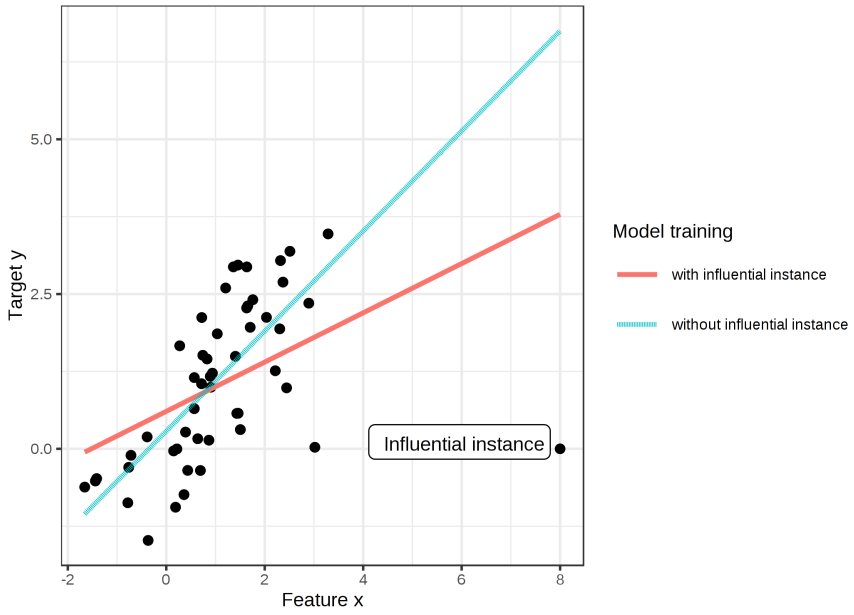


图 6.15. 具有一个特征的线性模型。在完整数据上进行了一次训练，以及在没有影响力的实例下进行了一次训练。删除有影响的实例会显著地改变拟合的斜率 (权重/系数)。

为什么有影响力的实例有助于理解模型?

对解释有影响力的实例背后的关键思想是将模型参数和预测追溯到一切开始的地方：训练数据。学习器 (即生成机器学习模型的算法) 是一种函数，该函数获取包含特征 X 和目标 y 的训练数据并生成机器学习模型。例如，决策树的学习器是一种选择分割特征和分割值的算法。神经网络的学习器使用反向传播来找到最佳权重。

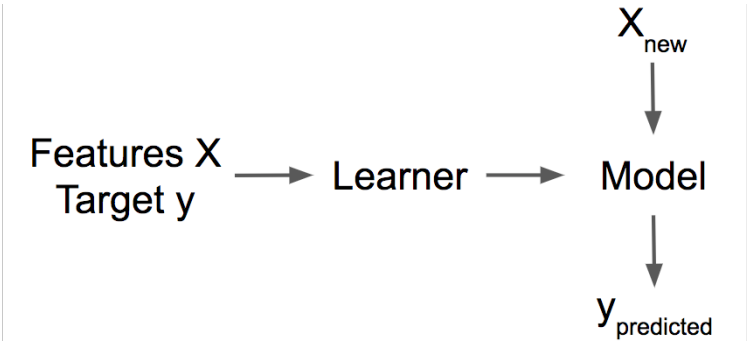


图 6.16. 学习器从训练数据 (特征加目标) 中学习模型。该模型对新数据进行预测。

我们询问如果在训练过程中从训练数据中删除实例，模型参数或预测将如何变化。这与其他可解释性方法形成了对比，其他可解释性方法分析了当我们操纵要预测的实例的特征 (例如部分依赖图或特征重要性) 时预测如何变化。对于有影响力的实例，我们不会将模型视为固定模型，而是将其视

为训练数据的函数。有影响力的实例可以帮助我们回答有关全局模型行为和单个预测的问题。哪些是对模型参数或整体预测最有影响力的实例？对于特定的预测，哪些是最有影响力的实例？有影响力的实例告诉我们模型可能会在哪些实例上出现问题，针对错误应该检查哪些训练实例，并且会给人以模型鲁棒性的印象。如果单个实例对模型的预测和参数有很大的影响，我们可能不信任模型。至少这将使我们进一步调查。

我们如何找到有影响力的实例？我们有两种测量影响力的方法：我们的第一个选择是从训练数据中删除实例，在简化的训练数据集上重新训练模型，并观察模型参数或预测的差异 (无论是单个的还是整个数据集)。第二个选择是增加数据实例的权重，通过模型参数的梯度来近似参数变化。删除方法更容易理解，并且会引申出权重增加方法，因此我们从前者开始。

6.4.1 删除诊断

统计学家已经在有影响力的实例领域进行了大量研究，尤其是对于 (广义) 线性回归模型。当你搜索“有影响力的观测”时，第一个搜索结果与诸如 DFBETA 和 Cook 距离之类的度量有关。**DFBETA** 衡量删除实例对模型参数的影响。**Cook 距离** (Cook，1977[52]) 衡量删除实例对模型预测的影响。对于这两种方法，我们必须重复训练模型，每次都忽略单个实例。删除诊断 (`Deletion Diagnostics`) 是将具有所有实例的模型的参数或预测与从训练数据中删除某个实例后的模型的参数或预测进行比较。DFBETA 定义为：

$$DFBETA_i = \beta - \beta^{(-i)}$$

其中，β 是在所有数据实例上训练模型时的权重向量，$\beta^{(-i)}$ 是在没有实例 i 的情况下训练模型时的权重向量。我会说非常直观。DFBETA 仅适用于具有权重参数的模型，例如逻辑回归或神经网络，而不适用于决策树、树的集成、某些支持向量机等模型。

Cook 距离是针对线性回归模型而发明的，并且存在对广义线性回归模型的近似值。训练实例的 Cook 距离定义为当从模型训练中删除第 i 个实例时，预测结果的平方差之和。

$$D_i = \frac{\sum_{i=1}^{n}(\hat{y}_j - \hat{y}_j^{(-i)})^2}{p \cdot MSE}$$

其中分子是在有和没有第 i 个实例的情况下对模型的预测之间的平方差，是对数据集求和的结果。分母是特征数 p 乘以均方误差。无论删除哪个实例 i，所有实例的分母都是相同的。Cook 距离告诉我们，当我们从训练中删除第 i 个实例时，线性模型的预测输出会发生多少变化。

我们可以在任何机器学习模型中使用 Cook 距离和 DFBETA 吗？DFBETA 需要模型参数，因此此度量仅适用于参数化模型。Cook 距离不需要任何模型参数。有趣的是，通常不会在线性模型和广义线性模型的范围之外看到 Cook 距离，但是在删除特定实例之前和之后获取模型预测之间的差异的想法非常普遍的。Cook 距离定义的一个问题是 MSE，它对所有类型的预测模型 (例如分类) 都没有意义。

可以将对模型预测影响的最简单的影响力度量写为：

$$\text{Influence}^{(-i)} = \frac{1}{n}\sum_{j=1}^{n}\left|\hat{y}_j - \hat{y}_j^{(-i)}\right|$$

该表达式基本上是 Cook 距离的分子，不同之处在于，用绝对差代替平方差相加。这是我做的一个选择，因为它对后面的示例有意义。删除诊断度量的一般形式包括选择度量 (例如预测结果)，并为在考虑所有实例时以及考虑实例被删除时训练的模型计算度量的差异。

我们可以轻松地分解影响力，为实例 j 的预测解释第 i 个训练实例的影响力是什么：

$$\text{Influence}_j^{(-i)} = \left|\hat{y}_j - \hat{y}_j^{(-i)}\right|$$

这也适用于模型参数的差异或损失的差异。在下面的示例中，我们将使用这些简单的影响力度量。

删除诊断示例

在以下示例中，我们训练了一个支持向量机，在给定风险因素的情况下预测宫颈癌，并衡量哪些训练实例对整体和特定预测的影响最大。由于癌症的预测是一个分类问题，因此我们用癌症的预测概率中的差异来衡量影响力。如果将实例从模型训练中删除，预测概率在数据集中平均显著增加或减少，则该实例具有影响力。要测量所有 858 个训练实例的影响力，就需要先对所有数据进行一次模型训练，并对其进行 858 次 (= 训练数据大小) 的训练，其中每次都会移除一个实例。

最有影响力的实例的影响力度量约为 0.01。0.01 的影响力意味着，如果我们删除第 540 个实例，则预测的概率平均变化 1 个百分点。考虑到癌症的平均预测概率为 6.4%，这是相当可观的。影响力度量值在所有可能的删除上的平均值为 0.2 个百分点。现在我们知道哪个数据实例对模型影响最大。这对于调试数据已经很有用。是否存在问题实例？是否存在测量误差？有影响力的实例是应首先检查是否存在错误的实例，因为它们中的每个错误都会强烈影响模型预测。

除了模型调试之外，我们还能学到一些东西来更好地理解模型吗？仅仅打印出 10 个最有影响力的实例并不是很有用，因为它只是具有许多特征的实例表。所有返回实例作为输出的方法只有在我们有很好的表示方式时才有意义。但是，当我们询问以下问题时，我们可以更好地理解哪种类型的实例具有影响力：有影响力的实例和没有影响力的实例有什么区别？我们可以将这个问题变成回归问题，并根据其特征值对实例的影响进行建模。我们可以从 “可解释的模型” 一章中自由选择任何模型。在此示例中，我选择了一个决策树 (下图)，该树显示来自 35 岁及以上女性的数据对支持向量机的影响最大。在数据集的 858 名女性中，有 153 名年龄在 35 岁以上。在 “部分依赖图” 一节中，我们发现 40 岁以后女性的预测癌症发病率急剧上升，而 “特征重要性” 也将年龄视为最重要的特征之一。影响力分析告诉我们，当预测较高年龄的癌症时，该模型将变得越来越不稳定。这本身就是有价值的信息。这意味着这些实例中的错误会对模型产生重大影响。

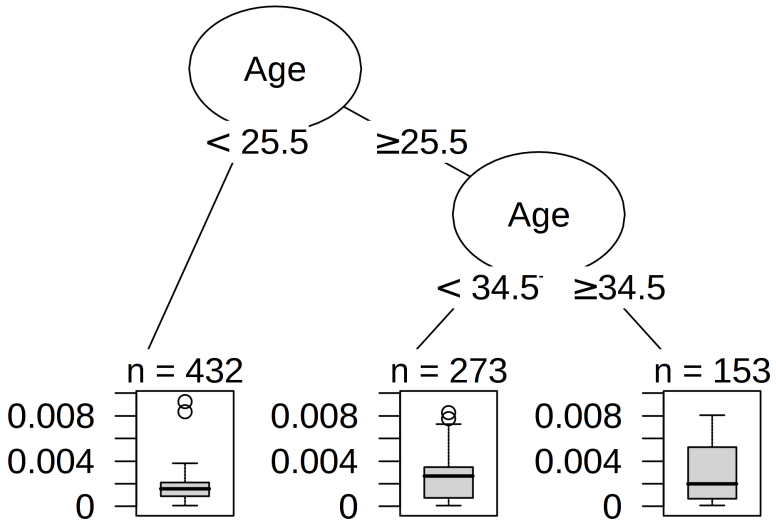


图 6.17. 决策树，对实例的影响力及其特征之间的关系进行建模。树的最大深度设置为 2。

第一个影响力分析揭示了总体最有影响力的实例。现在，我们选择一个实例，即第 7 个实例，我们要通过查找最有影响力的训练数据实例来解释预测。这就像一个反事实的问题：如果我们从训练过程中忽略实例 i，实例 7 的预测将如何变化？我们对所有实例重复此删除操作。然后，当训练实例在训练中被忽略时，我们选择导致实例 7 预测变化最大的训练实例，并使用它们来解释该实例的模型预测。我选择解释实例 7 的预测，因为它是癌症预测概率最高 (7.35%) 的实例，我认为这是一个有趣的案例，需要更深入地分析。我们可以返回对预测第 7 个实例的前 10 个最有影响力的实例 (以表格形式返回)。不是很有用，因为我们看不到太多。再次，更有意义的是通过分析特征来找出将有影响力的实例与无影响力的实例区分开的地方。我们使用经过训练的决策树来预测给定特征的影响力，但实际上，我们误用它只是找到结构而不实际预测某些东西。以下决策树显示了哪种训练实例对预测第 7 个实例最有影响力。

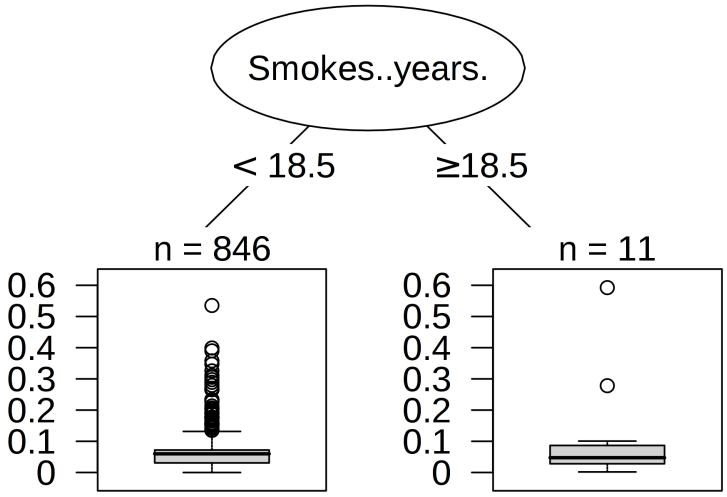


图 6.18. 决策树解释了哪些实例对预测第 7 个实例最有影响力。吸烟时间在 18.5 年或更长时间的女性的数据对第 7 个实例的预测有很大影响，绝对预测的平均变化会导致癌症发生概率降低 11.7 个百分点。

烟龄 18.5 年或更长时间的女性的数据实例对第 7 个实例的预测有很大影响力。第 7 个实例的女性吸烟了 34 年。在数据中，有 12 位女性 (1.40%) 吸烟 18.5 年或更长时间。在收集其中一名女性的吸烟年数方面犯的任何错误都会对第 7 个实例的预测结果产生巨大影响。

当我们删除第 663 例时，预测会发生最极端的变化。据称该病人吸烟了 22 年，与决策树的结果一致。如果我们删除第 663 例，则第 7 个实例的预测概率将从 7.35% 变为 66.60%!

如果我们仔细研究最有影响力的实例的特征，我们会发现另一个可能的问题。数据显示，该女性今年 28 岁，已经吸烟 22 年。这是一个非常极端的情况，她真的从 6 岁开始吸烟，或者这是一个数据错误。我倾向于相信后者。当然，在这种情况下，我们必须质疑数据的准确性。

这些实例说明了识别有影响力的实例对调试模型的有用性。该方法的一个问题是该模型需要针对每个训练实例进行重新训练。整个重新训练可能会非常缓慢，因为如果你有成千上万的训练实例，则必须对模型进行数千次重新训练。假设模型需要一天的训练时间，并且你有 1000 个训练实例，那么有影响力的实例 (不进行并行化) 的计算将花费近 3 年的时间。没有人有时间这样做。在本节的其余部分，我将向你展示一种不需要重新训练模型的方法。

6.4.2 影响函数

你：我想知道训练实例对特定预测的影响力。

研究：你可以删除训练实例，重新训练模型并测量预测中的差异。

你：太好了！但是你有没有一个方法可以让我不用重新训练？这需要很多时间。

研究：你是否具有一个损失函数的模型，该模型的参数具有二阶导数？

你：我用 logistic 损失训练了一个神经网络。所以是的。

研究：然后，你可以使用**影响函数** (`Influence Functions`) 来估计实例对模型参数和预测的影响力。影响函数是模型参数或预测对训练实例的依赖程度的度量。该方法不是删除实例，而是仅通过很小的步幅对损失中的实例加权。该方法包括使用梯度和 Hessian 矩阵来近似当前模型参数的损失。损失加权与删除实例类似。

你：太好了，这就是我想要的！

Koh and Liang (2017)[53] 建议使用影响函数 (一种稳健统计) 来衡量实例如何影响模型参数或预测。与删除诊断一样，影响函数将模型参数和预测追溯到负责的训练实例。但是，该方法并没有删除训练实例，而是近似于当实例在经验风险 (训练数据损失之和) 中被加权时，模型的变化程度。

影响函数的方法需要获得与模型参数相关的损失梯度，这仅适用于机器学习模型的子集。Logistic 回归、神经网络和支持向量机符合条件，而基于树的方法 (如随机森林) 则不符合。影响函数有助于理解模型行为，调试模型并检测数据集中的错误。

以下部分解释了影响函数背后的直觉和数学运算。

影响函数背后的数学

影响函数背后的关键思想是通过无限小的步幅 (ϵ) 对某个训练实例的损失加权，从而产生新的模型参数：

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n} L(z_i,\theta) + \epsilon L(z,\theta)$$

其中 θ 是模型参数向量，而 $\hat{\theta}_{\epsilon,z}$ 是用非常小的数字 ϵ 对 z 进行加权后的参数向量。L 是模型训练的损失函数，z_i 是训练数据，z 是我们想要增加权重来模拟其移除的训练实例。这个公式背后的直觉是：如果我们将训练数据中的某个特定实例 z_i 稍微增加一点 ϵ 权重，损失会有多大变化？优化这个新的组合损失参数向量又是什么样子？参数的影响函数，即对训练实例 z 加权对参数的影响力，可计算如下。

$$I_{\text{up,params}}(z) = \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1}\nabla_\theta L(z,\hat{\theta})$$

最后一个表达式 $\nabla_\theta L(z,\hat{\theta})$ 是对训练实例加权后参数的损失梯度。梯度是训练实例损失的变化率。它告诉我们，当我们稍微改变模型参数 $\hat{\theta}$ 时，损失会有多大的变化。梯度向量中的正项意味着相应模型参数的微小增加会增加损失，负项意味着参数的增加会减少损失。第一部分 $H_{\hat{\theta}}^{-1}$ 是逆 Hessian

矩阵 (损失相对于模型参数的二阶导数)。Hessian 矩阵是梯度的变化率，或表示为损失变化率的变化率。可以使用下列公式进行估算：

$$H_\theta = \frac{1}{n}\sum_{i=1}^{n} \nabla^2_{\hat{\theta}} L(z_i, \hat{\theta})$$

非正式地说：Hessian 矩阵记录损失在特定点的弯曲程度。Hessian 是一个矩阵，而不仅仅是一个向量，因为它描述了损失的曲率，并且曲率取决于我们的观察方向。如果你有许多参数，则实际计算 Hessian 矩阵会很费时。Koh 和 Liang 提出了一些有效计算的技巧，这超出了本节的范围。如上式所述，更新模型参数等效于在估计模型参数周围形成二次展开后执行单个牛顿步。

这个影响函数公式背后的直觉是什么? 公式来自于围绕参数 $\hat{\theta}$ 形成二次展开式。这意味着我们实际上不知道，或者计算实例 z 在被删除/加权时的损失究竟会有多大的变化太复杂了。我们利用当前模型参数设置下的陡度 (= 梯度) 和曲率 (=Hessian 矩阵) 的信息对函数进行局部近似。通过这种损失近似，我们可以计算出如果我们对实例 z 进行加权 (令 $\epsilon = -\frac{1}{n}$)，新参数将大致是什么样子：

$$\hat{\theta}_{-z} \approx \hat{\theta} - \frac{1}{n} I_{\text{up,params}}(z)$$

近似参数向量基本上是原始参数减去 z 的损失梯度 (因为我们要减少损失)，用曲率 (= 乘以逆 Hessian 矩阵) 缩放并用 $\frac{1}{n}$ 缩放，因为这是单个训练实例的权重。

下图显示了加权的工作方式。x 轴显示 θ 参数的值，y 轴显示实例 z 加权后的损失的相应值。出于演示的目的，这里的模型参数为一维，但实际上通常为高维。对于实例 z 我们只向损失改善 (下降) 的方向移动了 $\frac{1}{n}$。我们不知道删除 z 时损失的实际变化，但是利用损失的一阶和二阶导数，我们围绕当前模型参数创建了二次近似，并假设这是实际损失的行为。

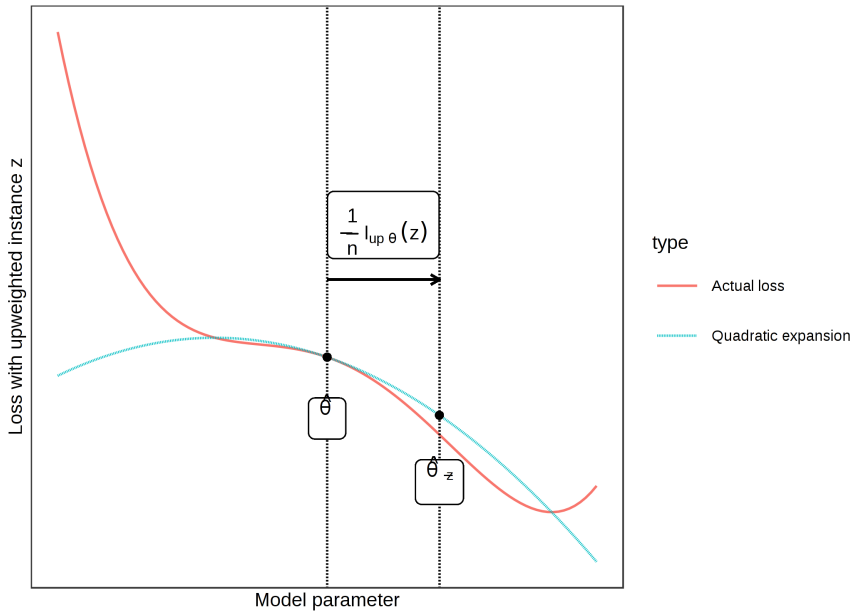


图 6.19. 通过在当前模型参数周围形成损失的二次展开，并在将实例 z 加权的损失最大改善 (y 轴) 的方向上移动 $\frac{1}{n}$，来更新模型参数 (x 轴)。如果我们删除 z 并在减少的数据上训练模型，则实例 z 在损失中的这种加权近似于参数变化。

实际上，我们不需要计算新参数，但可以使用影响函数来衡量 z 对参数的影响力。

当我们增加训练实例 z 的权重时，预测如何变化？我们既可以计算新参数，然后使用新参数化的模型进行预测，也可以直接计算实例 z 对预测的影响力，因为我们可以使用链式规则来计算影响力：

$$
\begin{aligned}
I_{up,loss}(z, z_{test}) &= \left.\frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\right|_{\epsilon=0} \\
&= \nabla_\theta L(z_{test}, \hat{\theta})^T \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} \\
&= -\nabla_\theta L(z_{test}, \hat{\theta})^T H_\theta^{-1} \nabla_\theta L(z, \hat{\theta})
\end{aligned}
$$

该方程式的第一行意味着，当我们增加实例 z 的权重并获得新参数 $\hat{\theta}_{\epsilon,z}$ 时，我们测量训练实例对某个预测 z_{test} 的影响力，作为预测实例损失的变化。对于方程的第二行，我们应用了导数的链式规则，并得出了测试实例的损失相对于参数的导数乘以 z 对参数的影响力。在第三行中，我们将表达式替换为参数的影响函数。第三行中的第一项 $\nabla_\theta L(z_{test}, \hat{\theta})^T$ 是测试实例相对于模型参数的梯度。

有一个公式是很好的，而且是显示事物的科学而准确的方法。但是，我认为对公式的含义进行一些直觉非常重要。公式 $I_{\text{up,loss}}$ 表示，训练实例 z 对实例 z_{test} 的预测的影响函数是“实例对模型参数变化的反应程度”乘以“当我们增加实例 z 的权重时参数变化了多少”。理解该公式的另一种方法：

影响力与训练损失和测试损失的梯度大小成正比。训练损失的梯度越高，其对参数的影响力越大，对测试预测的影响力也越大。测试预测的梯度越高，测试实例的影响力就越大。整个结构也可以看作是训练和测试实例之间相似度的度量 (由模型学习)。

理论和直觉就是这样。下一节将说明如何应用影响函数。

影响函数的应用

影响函数有许多应用，其中一些已在本节中介绍过。

了解模型行为

不同的机器学习模型具有不同的预测方法。即使两个模型具有相同的性能，它们根据特征进行预测的方式也可能非常不同，因此在不同的场景中会失败。通过识别有影响力的实例来了解模型的特定弱点，有助于在你的脑海中形成机器学习模型行为的 “心理模型”。下图展示了一个示例，其中训练了支持向量机 (SVM) 和神经网络以区分狗和鱼的图像。对于两种模型，鱼的示例性图像的最有影响力的实例非常不同。对于 SVM，如果实例颜色相似，则实例具有影响力。对于神经网络，实例在概念上相似时具有影响力。对于神经网络来说，即使是一只狗的图像也会是最有影响力的图像之一，这表明它是在颜色空间中学习了概念而不是欧几里得距离。

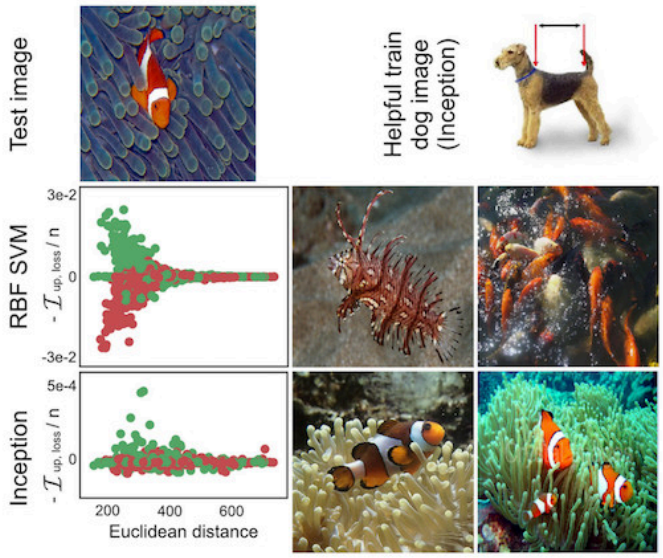


图 6.20. 狗还是鱼? 对于 SVM 预测 (中间行)，与测试图像颜色相似的图像最具影响力。对于神经网络预测 (最下排)，不同环境中的鱼类影响最大，但也有狗图像 (右上方)。Koh 和 Liang 的工作 (2017)。

处理域不匹配/调试模型错误

处理域不匹配 (Handling Domain Mismatches) 与更好地了解模型行为密切相关。域不匹配意味着

训练数据和测试数据的分布不同，这可能导致模型在测试数据上的表现不佳。影响函数可以识别导致错误的训练实例。你训练了一个预测模型来预测接受手术的病人的结果。所有这些病人来自同一家医院。现在，你在另一家医院使用该模型，会发现它对很多病人都不起作用。当然，你假设两家医院的病人不同，如果查看他们的数据，你会发现他们的许多特征有所不同。但是是什么特征或实例“破坏”了模型？同样，具有影响力的实例也是回答此问题的好方法。你以一个新病人为例，该模型对其做出了错误的预测，然后找到并分析了最有影响力的实例。例如，这可能表明第二家医院平均有老年病人，而训练数据中最有影响力的实例是第一家医院中的少数老年病人，并且该模型只是缺少用于很好地预测该亚组的数据。结论是，该模型需要针对更多年龄较大的病人进行训练，以便在第二家医院工作良好。

修复训练数据

如果你对检查正确性的训练实例的数量有限制，那么如何进行有效选择？最好的方法是选择最有影响力的实例，因为根据定义，它们对模型的影响力最大。即使你有个实例的值明显不正确，如果该实例不具有影响力，并且只需要用于预测模型的数据，则检查有影响力的实例也是更好的选择。例如，你训练了一个模型来预测病人是应该继续住院还是应该早日出院。你真的想要确保这个模型是鲁棒的并且做出正确的预测，因为错误释放病人会产生严重后果。病人记录可能非常混乱，因此你对数据质量没有完全的信心。但是检查病人信息并进行更正会非常耗时，因为一旦你报告了需要检查的病人，医院实际上就需要派人更仔细地查看所选病人的记录，这可能是手写的并且躺在一些档案中。检查病人数据可能需要一个小时或更长时间。考虑到这些成本，只检查几个重要的数据实例是有意义的。最好的方法是选择对预测模型影响较大的病人。Koh 和 Liang (2017) 表明，这种类型的选择比随机选择或者是对损失或分类错误最大的选择要好得多。

6.4.3 识别有影响力的实例的优点

删除诊断和影响函数的方法与模型无关章节中介绍的基于特征扰动的方法大不相同。对有影响力的实例的研究强调了训练数据在学习过程中的作用。**这使影响函数和删除诊断成为机器学习模型的最佳调试工具之一**。在本书介绍的技术中，它们是唯一直接帮助识别应该检查错误的实例的技术。

删除诊断是模型无关的，这意味着该方法可以应用于任何模型。基于导数的影响函数也可以应用于广泛的模型。

我们可以使用这些方法来**比较不同的机器学习模型**并更好地理解它们的不同行为，而不仅仅是比较预测性能。

在本节中我们没有讨论这个话题，但是**通过导数的影响函数也可以用来创建对抗样本**。这些实例经过操作后使得当在这些操作实例上训练模型时，模型无法正确预测某些测试实例。与“对抗样本”小节中的方法的不同之处在于，攻击是在训练期间发生的，也称为中毒攻击 (`Poisoning Attacks`)。如果你有兴趣，请阅读 Koh 和 Liang (2017) 的论文。

对于删除诊断和影响函数，我们考虑了预测的差异，对于影响函数则考虑了损失的增加。但是，实际上，该方法**可以推广**到以下形式的任何问题："……删除或增加实例 z 时会发生什么？"，你可以在其中用所需模型的任何函数来填充"……"。你可以分析训练实例在多大程度上影响模型的总损失。你可以分析训练实例对特征重要性的影响程度。你可以分析在训练决策树时，训练实例对第一次分割所选择的特征的影响力有多大。

6.4.4 识别有影响力的实例的缺点

删除诊断的计算非常昂贵，因为它们需要重新训练。但是历史表明，计算机资源正在不断增加。一项 20 年前在资源方面难以想象的计算可以很容易地在你的智能手机上完成。你可以在几秒/分钟的时间内在笔记本电脑上训练具有数千个训练实例和数百个参数的模型。因此，假设删除诊断在 10 年内即使使用大型神经网络也不会出现问题，这并不是一个很大的飞跃。

影响函数是删除诊断的一种很好的替代方法，**但仅适用于参数可微的模型**，如神经网络。它们不适用于基于树的方法，例如随机森林、提升树或决策树。即使你有带有参数和损失函数的模型，损失也可能是不可微的。但对于最后一个问题，有一个技巧：例如，当底层模型使用 Hinge 损失而不是某些可微损失时，使用可微损失代替计算影响力。针对影响函数损失将替换的损失的平滑版本，但是仍可以使用非平滑损失来训练模型。

影响函数仅仅是近似的，因为该方法在参数周围二次展开。近似值可能是错误的，并且删除实例时，实例的影响力实际上更高或更低。Koh 和 Liang (2017) 给出了一些示例，表明由影响函数计算出的影响力与删除实例后实际重新训练模型时获得的影响力度量接近。但是，不能保证近似值总是如此接近。

还有，在我们称一个实例为有影响力或没有影响力的影响力度量**没有明确的截止点**。通过影响力对实例进行排序很有用，但是拥有一种不仅可以对实例进行排序，而且实际上可以区分有影响力和没有影响力的方法也将非常有用。例如，如果你为一个测试实例确定了 10 个最有影响力的训练实例，其中一些可能并不具有影响力，因为，比如，只有前 3 个实例具有真正的影响力。

影响力度量**仅考虑单个实例的删除**，而不是一次删除多个实例。数据实例的组可能具有一些交互，这些交互强烈影响模型的训练和预测。但是问题出在组合学上：从数据中删除单个实例有 n 种可能，从训练数据中删除两个实例有 $n(n-1)$ 的可能，有 $n(n-1)(n-2)$ 次删除三个实例的可能…我想你可以看到这是怎么回事，有太多的组合了。

6.4.5 软件和替代方法

删除诊断非常容易实现。可以看看我为本节中的示例编写的 代码。

对于线性模型和广义线性模型，在 R stats 包中实现了许多影响力度量，例如 Cook 距离。

Koh 和 Liang 在他们的论文中发布了影响函数的 Python 代码。这太棒了！不幸的是，它 “只是” 本文的代码，而不是维护和文档化的 Python 模块。该代码主要专注于 Tensorflow 库，因此你无法将其直接用于使用其他框架的黑盒模型，例如 scikit-learn。

Keita Kurita 写了一篇关于影响函数的 博客，帮助我更好地理解了 Koh 和 Liang 的论文。该博客文章更深入地介绍了黑盒模型影响函数背后的数学原理，还讨论了有效实现该方法的一些数学 “技巧”。

第七章　水晶球

可解释机器学习的未来是什么？本章可以看作是一个思辨性的脑力练习，也是对可解释的机器学习将如何发展的主观猜测。尽管我以悲观小故事作为这本书的开头，但是我希望以一种乐观的态度结束这本书。

我的“预测”基于三个前提：

1. **数字化：任何 (有趣的) 信息都将被数字化。**想想电子现金和在线交易。想想电子书、音乐和视频。想想我们的环境、人类行为、工业生产过程等的所有感官所产生的数据。一切数字化的驱动因素是：廉价的计算机/传感器/存储、规模效应 (赢者通吃)、新的商业模式、模块化价值链、成本压力等等。
2. **自动化：当一个任务可以被自动化，并且自动化的成本低于一段时间内执行该任务的成本时，该任务将被自动化。**甚至在引入计算机之前，我们就已经有了一定程度的自动化。例如，织布机自动进行编织或蒸汽机自动产生马力。但是计算机和数字化将自动化提升到了一个新的高度。简单地说，你可以编程 forloops、编写 Excel 宏、自动化电子邮件响应等等，这显示了个人可以自动化的程度。自动售票机可自动购买火车票 (不再需要收银员)，洗衣机可自动执行洗衣服务，定期订单可自动执行付款交易等。自动化任务可以节省时间和金钱，因此有巨

大的经济和个人动机来推动自动化实现。目前，我们能看到语言翻译，自动驾驶，甚至在小范围内的科学探索自动化。

3. **错误的指定：我们不可能完美地指定一个有所限制的目标。**想想瓶子里的精灵总是会如实实现你的愿望："我想成为世界上最富有的人！" -> 你成为首富，但副作用是你持有的货币由于通货膨胀而崩溃。
   "我要为余生感到幸福！"-> 接下来的 5 分钟，你会感到非常高兴，然后精灵将你杀死。
   "我希望世界和平！" -> 精灵杀死所有人类。
   我们错误地指定目标，要么是因为我们不知道所有的约束条件，要么是因为我们无法衡量它们。让我们以公司为例，说明目标规范不完善。公司的简单目标是为股东赚钱。但是，此规范并没有抓住我们为之奋斗的所有真正目标：例如，我们不欣赏一家公司为了赚钱而杀人、毒害河流或仅仅印制自己的钱。我们发明了法律、法规、制裁、合规程序、工会等，以完善还不完善的目标规范。你可以亲自体验的另一个示例是"回形针"，这是一个目的产生尽可能多的回形针的游戏。警告：它会上瘾。我不想把它搞得太糟，但可以说事情很快就失控了。在机器学习中，目标规范中的缺陷来自不完善的数据抽象 (有偏的总体，测量误差等)、不受约束的损失函数、缺乏约束知识、训练数据与应用数据之间的分布偏移等等。

数字化正在推动自动化。但是不完善的目标规范与自动化相冲突。我认为这种冲突部分是由解释方法来调解的。

我们的预测阶段已经准备就绪，水晶球也已经准备就绪，现在我们来看一下这个领域的发展方向！

7.1 机器学习的未来

没有机器学习，就无法解释机器学习。因此，在谈论可解释性之前，我们必须猜测机器学习的发展方向。

机器学习 (或 "AI") 与许多承诺和期望相关联。但是，让我们从不那么乐观的观察开始：尽管科学开发了许多精致的机器学习工具，但根据我的经验，将它们集成到现有的流程和产品中还是很困难的。不是因为不可能，而是因为公司和机构需要时间才能赶上。在当前 AI 炒作的淘金热中，公司开设了 "AI 实验室"、"机器学习单元"，并雇用了 "数据科学家"、"机器学习专家"、"AI 工程师" 等，但实际情况是，以我的经验，相当令人沮丧。通常，公司甚至没有所需格式的数据，而数据科学家则等待数月之久。有时，由于媒体的原因，公司对 AI 和数据科学抱有很高的期望，因此数据科学家永远无法实现它们。通常没有人知道如何将这些人整合到现有结构和许多其他问题中。这导致了我的第一个预测。

机器学习将缓慢而稳定地增长

数字化正在推进，自动化的诱惑也在不断地推进。即使采用机器学习的道路缓慢而艰难，机器学习

仍在不断从科学转向业务流程、产品和现实应用。

我认为我们需要更好地向非专业人士解释哪些类型的问题可以被表述为机器学习问题。我知道许多高薪数据科学家使用报表和 SQL 查询来执行 Excel 计算或经典的商业智能，而不是应用机器学习。但是，已经有一些公司成功地使用了机器学习，而大型互联网公司处于最前沿。我们需要找到更好的方法来将机器学习集成到流程和产品中，培训人员并开发易于使用的机器学习工具。我相信机器学习将变得更加易于使用：我们已经看到机器学习正在变得越来越容易使用，例如通过云服务。一旦机器学习成熟——这个小孩已经迈出了第一步——我的下一个预测是：

机器学习将推动很多事情

基于“凡是可以自动化的东西都将被自动化”的原则，我得出结论，只要有可能，任务将被表述为预测问题并通过机器学习解决。机器学习是自动化的一种形式，或者至少可以是自动化的一部分。人类目前执行的许多任务已被机器学习取代。以下是一些使用机器学习自动化部分任务的示例：

- 文件的整理/决策/完成 (例如在保险公司、法律部门或咨询公司中)
- 数据驱动的决策，例如信贷申请
- 药物发现
- 流水线的质量控制
- 自动驾驶汽车
- 疾病诊断
- 翻译。在这本书中，我使用了由深度神经网络提供支持的翻译服务 (DeepL)，可以将句子从英语翻译成德语再翻译成英语，从而改善句子。
- …

机器学习的突破不仅可以通过更好的计算机/更多的数据/更好的软件来实现，而且还可以：

可解释性工具促进了机器学习的采用

基于机器学习模型的目标永远不可能被完美指定的前提下，可解释的机器学习对于弥合错误目标和实际目标之间的差距是必要的。在许多领域和部门，可解释性将成为采用机器学习的催化剂。一些轶事证据：我刚才谈到的许多人不使用机器学习，因为他们无法向他人解释这些模型。我相信可解释性将解决这个问题，并使机器学习对要求透明的组织和人员有吸引力。除了问题的错误说明之外，出于法律原因、规避风险或要深入了解底层任务，许多行业都需要可解释性。机器学习使建模过程自动化，并将人类与数据和底层任务进一步分离：这增加了实验设计、训练分布选择、采样、数据编码、特征工程等方面出现问题的风险。解释工具使识别这些问题更加容易。

7.2 可解释性的未来

让我们看一下机器学习可解释性的可能未来。

重点将放在与模型无关的可解释性工具上

当它与底层的机器学习模型分离时，使可解释性自动化变得容易得多。与模型无关的可解释性的优点在于其模块化。我们可以轻松地替换底层的机器学习模型。我们可以轻松地替换解释方法。由于这些原因，与模型无关的方法将更好扩展。这就是为什么我认为与模型无关的方法从长远来看将变得更加占主导地位的原因。但是 “本质上可解释的方法” 也将占有一席之地。

机器学习将是自动化的，并具有可解释性

模型训练的自动化已经是一个显而易见的趋势。这包括自动工程和特征选择，自动超参数优化，不同模型的比较以及模型的集成或堆叠。结果是最佳可能的预测模型。当我们使用与模型无关的解释方法时，我们可以将它们自动应用于自动机器学习过程中出现的任何模型。在某种程度上，我们也可以使第二步自动化：自动计算特征重要性、绘制部分依赖关系、训练代理模型等等。没有人会阻止你自动计算所有这些模型解释。实际的解释仍然需要人类。想象一下：你上传了一个数据集，指定了预测目标，并且只需按一下按钮，就可以训练出最佳的预测模型，并且程序会吐出对该模型的所有解释。已经有第一批产品，我认为对于许多应用程序来说，使用这些自动化机器学习服务就足够了。今天，任何人都可以在不了解 HTML，CSS 和 JavaScript 的情况下建立网站，但是仍然有许多 Web 开发人员。同样，我相信每个人都可以在不知道如何编程的情况下训练机器学习模型，并且仍然需要机器学习专家。

我们不分析数据，我们分析模型

原始数据本身总是无用的。(我故意夸大其词。现实情况是你需要对数据有深刻的理解才能进行有意义的分析。) 我不在乎数据；我关心数据中包含的知识。可解释的机器学习是从数据中提取知识的好方法。你可以广泛地探查模型，模型可以自动识别特征是否与预测相关，以及如何与预测相关 (许多模型具有内置的特征选择)，模型可以自动检测关系的表示方式，以及 (如果训练正确) 最终模型是对现实的很好的近似。

许多分析工具已经基于数据模型 (因为它们基于分布假设)：

- 简单的假设检验，例如学生 t 检验
- 对混杂因素调整 (通常是 GLM) 的假设检验
- 方差分析 (ANOVA)
- 相关系数 (标准化线性回归系数与 Pearson 的相关系数有关)
- …

我在这里告诉你的其实并不是什么新鲜事。那么，为什么要从分析基于假设的透明模型转换为分析

没有假设的黑盒模型呢？因为做出所有这些假设都是有问题的：它们通常是错误的 (除非你认为世界上大多数地方都遵循高斯分布)、难以检查、非常不灵活且难以自动化。在许多领域中，与黑盒机器学习模型相比，基于假设的模型通常对原始测试数据的预测性能更差。这仅适用于大型数据集，因为具有良好假设的可解释模型通常在小型数据集上的性能要优于黑盒模型。黑盒机器学习方法需要大量数据才能正常工作。随着一切的数字化，我们将拥有更大的数据集，因此机器学习的方法变得更具吸引力。我们不做任何假设，而是尽可能逼近现实 (同时避免训练数据过拟合)。我认为我们应该开发统计中拥有的所有工具来回答问题 (假设检验、相关性度量、交互作用度量、可视化工具、置信区间、p 值、预测区间、概率分布)，并将其重写为黑盒模型。在某种程度上，这已经发生了：

- 让我们采用经典的线性模型：标准化回归系数已经是一种特征重要性度量。使用置换特征重要性度量，我们有了可与任何模型一起使用的工具。
- 在线性模型中，系数衡量单个特征对预测结果的影响。它的广义形式就是部分依赖图。
- 检验 A 或是 B 更好：为此，我们也可以使用部分依赖函数。(据我所知) 我们还没有针对任意黑盒模型进行统计检验。

数据科学家将使自己自动化

我相信，数据科学家最终将自动化完成许多分析和预测任务的工作。为了做到这一点，必须对任务进行明确定义，并且周围必须有一些流程和例程。如今，这些例程和过程是缺少的，但是数据科学家和同事正在研究它们。随着机器学习成为许多行业和机构不可或缺的一部分，许多任务将实现自动化。

机器人和程序会自我解释

我们需要对大量使用机器学习的机器和程序有更直观的界面。例如说：一辆自动驾驶汽车，报告其突然停车的原因 (“孩子越过马路的概率为 70%”)；信用违约程序，向银行员工解释为何拒绝信用申请 (“申请人的信用卡过多，并且从事不稳定的工作。”)；一个机械臂，解释了为什么它把物品从传送带上搬到垃圾桶里 (“物品底部有裂缝。”)。

可解释性可以促进机器智能研究

我可以想象，通过对程序和机器如何自我解释进行更多研究，我们可以改善对智能的理解，并更好地创造智能机器。

最后，所有这些预测都是推测，我们必须看到未来真正带来了什么。形成你自己的观点并继续学习！

参考文献

[1] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[2] Definition of algorithm., 2017. https://www.merriam-webster.com/dictionary/algorithm.

[3] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[4] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in neural information processing systems*, pages 2280–2288, 2016.

[5] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[6] Fritz Heider and Marianne Simmel. An experimental study of apparent behavior. *The American journal of psychology*, 57(2):243–259, 1944.

[7] Zachary C Lipton. The mythos of model interpretability. *Queue*, 16(3):31–57, 2018.

[8] Marko Robnik-Šikonja and Marko Bohanec. Perturbation-based explanations of prediction models. In *Human and machine learning*, pages 159–175. Springer, 2018.

[9] Peter Lipton. Contrastive explanation. *Royal Institute of Philosophy Supplements*, 27:247–266, 1990.

[10] Daniel Kahneman and Amos Tversky. The simulation heuristic. Technical report, Stanford Univ CA Dept of Psychology, 1981.

[11] Erik Štrumbelj and Igor Kononenko. A general method for visualizing and explaining black-box regression models. In *International Conference on Adaptive and Natural Computing Algorithms*, pages 21–30. Springer, 2011.

[12] Raymond S Nickerson. Confirmation bias: A ubiquitous phenomenon in many guises. *Review of general psychology*, 2(2):175–220, 1998.

[13] Hadi Fanaee-T and Joao Gama. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 2(2-3):113–127, 2014.

[14] Túlio C Alberto, Johannes V Lochter, and Tiago A Almeida. Tubespam: Comment spam filtering on youtube. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 138–143. IEEE, 2015.

[15] Kelwin Fernandes, Jaime S Cardoso, and Jessica Fernandes. Transfer learning with partial observability applied to cervical cancer screening. In *Iberian conference on pattern recognition and image analysis*, pages 243–250. Springer, 2017.

[16] Robert C Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90, 1993.

[17] William W Cohen. Fast effective rule induction. In *Machine learning proceedings 1995*, pages 115–123. Elsevier, 1995.

[18] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, David Madigan, et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[19] Christian Borgelt. An implementation of the fp-growth algorithm. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, pages 1–5, 2005.

[20] Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable bayesian rule lists. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3921–3930. JMLR. org, 2017.

[21] Jerome H Friedman, Bogdan E Popescu, et al. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3):916–954, 2008.

[22] Marjolein Fokkema and Benjamin Christoffersen. Pre: Prediction rule ensembles. https://CRAN.R-project.org/package=pre. 2017.

[23] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.

[24] Qingyuan Zhao and Trevor Hastie. Causal interpretations of black-box models. *Journal of Business & Economic Statistics*, pages 1–10, 2019.

[25] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

[26] Alex Goldstein, Adam Kapelner, Justin Bleich, and Maintainer Adam Kapelner. Package 'icebox'. 2017.

[27] Daniel W Apley. Visualizing the effects of predictor variables in black box supervised learning models. *arXiv preprint arXiv:1612.08468*, 2016.

[28] Giles Hooker. Discovering additive structure in black box functions. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 575–580, 2004.

[29] Brandon M Greenwell, Bradley C Boehmke, and Andrew J McCarthy. A simple and effective model-based variable importance measure. *arXiv preprint arXiv:1805.04755*, 2018.

[30] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[31] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. Model class reliance: Variable importance measures for any machine learning model class, from the"rashomon"perspective. *arXiv preprint arXiv:1801.01489*, 68, 2018.

[32] David Alvarez-Melis and Tommi S Jaakkola. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*, 2018.

[33] Lloyd S Shapley. A value for n-person games. Technical report, Rand Corp Santa Monica CA, 1952.

[34] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.

[35] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

[36] Mateusz Staniak and Przemyslaw Biecek. Explanations of model predictions with live and breakdown packages. *arXiv preprint arXiv:1804.01955*, 2018.

[37] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

[38] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

[39] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[40] David Martens and Foster Provost. Explaining data-driven document classifications. *Mis Quarterly*, 38(1):73–100, 2014.

[41] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. *arXiv preprint arXiv:1907.02584*, 2019.

[42] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki. Inverse classification for comparison-based interpretability in machine learning. *arXiv preprint arXiv:1712.08443*, 2017.

[43] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[44] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[45] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[46] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[47] Tom B. Brown. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

[48] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

[49] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

[50] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[51] Leonard KAUFMAN Peter J RDUSSEEUN. Clustering by means of medoids. 1987.

[52] R Dennis Cook. Detection of influential observation in linear regression. *Technometrics*, 19(1):15–18, 1977.

[53] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org, 2017.