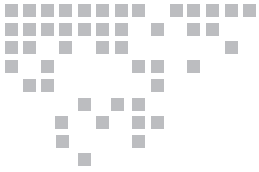
第 1 部分 Part 1

实时机器学习方法论



Chapter 1 第 1 章

实时机器学习综述

1.1 什么是机器学习

相信本书的读者都已经接触过一点机器学习了，或者听说过各种新奇的机器学习方法，或者通过相关新闻了解过机器学习的应用场景。那么，大家是否了解机器学习的定义呢？事实上，对它的定义层出不穷，不同领域的大咖往往都会有一个从自己角度出发的特别“机灵”的定义。比如，吴恩达（Andrew Ng）是深度学习的先驱者之一，他对机器学习的定义是从计算机从业者的角度出发的，他的定义是：

> 机器学习是一门科学，它旨在让计算机自主化工作，而不需要刻意编程。

而从统计和数据分析的角度出发，世界领先的统计软件公司 SAS 对机器学习的定义是：

> 机器学习是一种方法，它旨在用数据分析自动化模型的建立。

笔者个人从学术和工业界应用的角度出发，认为机器学习的定义应该包括以下三个方面。

- 用数据说话：在常规计算机编程中，所有的逻辑都是人为设定的。而机器学习方法是试图让观测到的数据和现象成为编撰逻辑的依据，不同模型之间的衡量标准也试图尽量达到标准化，以使得人为干预最小化。
- 高度自动化：机器学习模型往往会在工业应用中不断重复更新，所以机器学习建模生存期中的每个步骤往往都是可以高度自动化的。
- 鲁棒性：虽然教科书中很少提及，但鲁棒性（又称稳定性，Robustness）确实是机器

学习方法论中隐含的一个巨大要求。由于模型建立高度自动化，因此我们需要运用的机器学习模型在面对极端数据的时候只会受到较少影响，不需要人为排错。

根据笔者的经验，以上三点是一个组织成功运用机器学习的必要条件，但是一定要以用户体验为出发点来进行均衡。

在工业应用中，上面这三点的重要性总是在不断得到印证。下面就通过两个应用中的有名案例来体会一下。

1. 谷歌通过机器学习和人工干预进行网页筛查

谷歌等搜索引擎公司每天需要处理上百万个新网页信息。为了向用户快速提供这些信息，谷歌多年来通过不懈的努力开发出了 Caffeine 平台，将提供实时新闻搜索结果的延迟从一天缩短到了若干分钟。机器学习数据驱动、高度自动化的特点让谷歌用户受益不少。就连微软在通过记者发布会宣布发行 Windows10 的时候，谷歌搜索引擎也比微软自有的必应搜索引擎更快地呈现了与 Windows10 相关的信息。同时为了满足鲁棒性的要求，谷歌通过第三方人工服务，不断进行人工抽样审查了大量的网页内容。

2. Yelp 机器学习模型的失败

Yelp 类似于国内的大众点评网，其内容多为用户生成，对餐馆、娱乐、家装等行业都有很全面的覆盖。由于大量商家的成败都取决于 Yelp，因此市场上出现了冒充消费者进行刷点的评论师。评论师会按照商家的要求对商户进行不公正的点评，从而对消费者产生误导。Yelp 意识到了这样的问题，并且建立了机器学习模型进行自动化侦测。但可能是建模数据出现了问题（比如，建模的时候使用了评论师的数据），因此生成的模型并没有阻挡评论师的进攻，真正的用户所产生的评论反而会被屏蔽掉，用户体验大打折扣。

通过这样的案例，我们可以意识到基本数据采集对机器学习模型的重要性。如果数据出现了问题，那么后面的模型、架构再强大也没有办法产生效益。

1.2 机器学习发展的前世今生

1.2.1 历史上机器学习无法调和的难题

早在 2011 年，笔者之一彭河森正在谷歌总部实习的时候，机器学习的应用还主要集中在几个互联网巨头手里。当时，机器学习的大规模应用存在以下三个方面的限制。

1. 运维工具欠缺

就拿灵活开发流程来说吧，早在 2011 年，谷歌、亚马逊等公司开发了内部自有的协同部署工具，而开源协同部署工具 Jenkins 才刚刚起步，不少公司对服务器集群的管理还停留在 rsync 和 ssh 脚本阶段。机器学习的应用往往需要多台服务器各司其职、协同作业，这也增加了机器学习开发、部署的难度。这也解释了为什么早期的机器学习软件包（如 Weka

等）都是单机版的，因为服务器配置真的是太麻烦了。

一个机器学习系统的上线运行，需要前端、后端多个组成部分协调工作。在缺乏运维工具的年代，这样的工作量会大得惊人。人力物力的超高成本投入，有限而且不确定的回报率，这些都让机器学习从业人员在实际应用中难以生存。另外一些机器学习专用工具（如Hadoop）在早年是很少有人懂得其部署步骤的，一般的工程师都不愿意去主动接触它。在这种情况下，两位笔者都曾经为自己的部门搭建过 Hadoop 平台。

2. 模型尚未标准化

早在 2011 年，机器学习模型仍然是门派林立，SVM、神经网络等大家之作往往需要与作者直接书写的 libsvm 等软件库相对应，而统一标准化语言的软件包才刚刚出现。当时已经有很多线性模型的软件包，但是如果需要使用随机森林，那么还需要再安装其他软件包，不同软件包的用法又是不一样的。这样的非标准化工作大大加重了开发的工作量，减缓了工作的进度。

机器学习软件行业这种“军阀割据”的格局，导致机器学习从业人员必须对每个对应的模型都要进行二次开发。理论上来说，再好的机器学习模型，在实际系统里面，其地位也就应该是一个可以随时替换的小插件。可是实际上，由于二次开发，往往导致一个模型很难替代和更新，只能与系统黏在一起。

3. 全栈人才欠缺

如果有了足够多的全栈人才，那么上面的挑战就都不是问题了。可是由于机器学习的门槛比较高，往往需要拥有多年理论训练的人才能胜任相关工作。如果没有多年的工作经验，那么这样的人员往往对系统运维等工作一窍不通。 找到懂机器学习的人容易，找到懂系统运维的人也不难，可是要找到两样都会的人就非常困难了，这样的情况直接导致全栈人才极度缺乏。如果大家有幸能够供职于一些积累了多年机器学习实战经验的大公司，对机器学习系统架构进行“考古”，就会发现这个公司的机器学习系统架构设计大多取决于该公司架构人员的学历背景，每个公司在重模型还是重架构方面都有自己的倾向性。

1.2.2 现代机器学习的新融合

上面三个问题在 2012 年得到了转变，而这一年，也是两位笔者在亚马逊公司相遇的年份。能站在机器学习应用的前线，目睹这一革命的发生，我们感到非常欣喜。我们有幸成为这个大潮中的第一批用户，而且通过实际经历，了解了这一变化的来龙去脉。经过这几年的历练，我们熟悉了机器学习架构应用的相关知识。更重要的是，我们通过广泛的实验和讨论，总结出了对机器学习架构各个组成部分进行选择、预判的规律，让我们可以通过关键点分析对机器学习的浪潮进行预测评估。这也是我们希望能够通过本书与大家进行分享的经验。

从 2012 年到 2016 年，机器学习领域主要发生了以下这些变化。

1. 轻量化运维工具成为主流

运维过于复杂，已经成为众多互联网企业的心头之患。这个痛点在 2012 年到 2016 年这几年之间得到了革命性的解决，其中一马当先的急先锋是 Docker，Docker 是一款轻量化容器虚拟机生态系统（本书的第 6 章会详细介绍 Docker，并且其应用实例也会贯穿全书）。在 Docker 出现以前，调试部署的工作往往会占用开发人员大量的时间，例如在开发人员电脑上能够成功运行的程序，部署好了之后却不能正常运行。Docker 的出现，使得开发人员电脑上和生产服务器上面的虚拟机镜像内容完全相同，从而彻底杜绝了这种悲剧的发生，大大缩短了开发部署的周期。

与此同时，新近出现的连续化部署（Continuous Integration，CI）工具，如 Jenkins，自动化了机器学习模型的训练和部署流程，大大提高了模型训练的效率。

2. 机器学习工具标准化

近几年机器学习软件的出现仍然呈现爆炸式增长的趋势，但是领军软件已经崭露头角。在单机机器学习处理方面，基于 Python 的 Scikit-learn 工具已经成为了监督式机器学习模型的主流。Scikit-learn 具有丰富的机器学习模型模块，并且非常易于进行系统整合，我们在本书的第 4 章将对其进行详细介绍。与此同时，大数据机器学习方面，Spark 和 MLLib 也成为了主流，MLLib 几乎涵盖了所有的主流分布式机器学习模块，并且非常易于扩展。这些新工具的出现，让开发人员不再需要对模型进行二次开发，从而大大提高了效率。

3. 全栈人才登上历史舞台

机器学习领域人才大战正酣，吸引了众多优质青年投身于这一领域。我们非常欣喜地看到众多全栈型人才的出现。所谓全栈型人才，即为上可建模型、调参数，下可搭集群、做部署，左可开讲座、熬鸡汤，右可面风投、拉项目的“多才多艺”型人才。全栈型人才在组织中可以起到掌控技术全局的作用，可以大大缩短开发所需要的时间，减少系统反复修改带来的浪费。

1.3 机器学习领域分类

从方法论的角度来讲，机器学习分为监督式学习、非监督式学习和新兴机器学习课题三大方面。

1. 监督式学习

监督式机器学习的主要任务是通过机器学习模型和已有信息，对感兴趣的变量进行预测，或者对相关对象进行分类。监督式机器学习的一些应用场景包括：对网页访问进行分类，通过声音、文字、表情等信息对用户心情进行判断，对天气进行预测等。常用的监督式机器学习方法包括线性模型、最近邻估计、神经网络、决策树等。最近特别火热的深度学习在图像分类等场景的应用也是监督式学习的一种。

2. 非监督式学习

非监督式学习的主要任务是对数据进行描述。在非监督式学习的应用场景中，所有变量几乎都处于同等地位，不存在一个需要进行预测和分类的目标。故此非监督式学习主要用于机器学习建模前期对数据的分析和可视化处理，其在生产环境中的应用较少。非监督式学习的主要方法包括聚类分析、隐含因子分析等。

3. 新兴的机器学习课题

最近五年，强化学习（reinforcement learning）领域在深度学习的带领下得到了飞速的发展。强化学习旨在通过对实际事件的观察得到行为优化的结论，例如，AlphaGo 通过强化学习优化下围棋的策略。到目前为止，强化学习暂时还主要停留在学院派研究中，实际应用暂时有限。

本书将着重讲述机器学习方法在实时场景中的应用，我们将会简要介绍主流监督式学习的方法和应用。另外值得一提的是，在 IT 工业界应用中，自然语义处理、推荐系统和搜索引擎由于其专业领域深度和应用的难度，在各种文献中它们往往被列为独立的大方向。本书的第 9 章和第 12 章会对自然语言的处理进行简单的介绍。

1.4 实时是个“万灵丹”

成长会解决一切问题。如果一个企业正在飞速成长，大家步调一致、同心齐力，那么内斗或管理混乱等问题将是难以出现的。而当企业的成长受到了制约，停滞不前的时候，往往就会出现众多非技术性原因造成的悲剧。

我们强调机器学习的实时性，就是为了保证应用机器学习的企业能够利用机器学习的资源大踏步向前，而不会被早早地制约，徘徊不前。机器学习就已经够有挑战性的了，为什么还要采用实时机器学习？根据我们的经验，实时机器学习上马应该越早越好，原因具体有以下三点。

1. 实时架构稳定性可以得到保证

Fail fast（快速失败）强调如果有问题，那么应让问题尽早出现，使得问题可以得到尽早修复，这是软件工程里面一个重要的思想。如果系统有问题，就应该让问题尽早暴露，而不是往后拖。实时机器学习架构强调连续运行，设计、实施中的任何问题一般都可以在部署上线后的几个小时内暴露出来，以及时得到更正。

非实时架构往往会在每天的某一个固定时刻进行数据处理、建模等工作。如果前一天开发人员部署了问题程序，到了第二天运行的时候才发现，打好补丁就到了第三天，然后验证补丁是否正确又到了第四天……在流程的反复中，宝贵的时间就这样浪费下去了。

2. 代码、架构质量可以得到保证

与非实时架构不同，实时架构设计假设数据是无限量连续到来的。这时候系统的设计

和开发必须从一开始就设计好全局步骤，而不是走一步算一步，由此可以大大提高架构设计的质量。与此同时，连续交付的要求需要代码能够事先考虑到所有边际情况，这样我们所得到的代码质量也会更高。

3. 数据驱动的组织文化可以得到加强

由于机器学习具有实时性，因此所有有关业务效果的讨论都可以基于实时数据，而不是凭空根据大佬的主观臆断。与此相对的，没有采用实时机器学习的组织往往只会定期手动进行数据分析，得到真相的速度大大减慢，不利于商业决策的正确执行。另外，非实时架构企业的数据处理往往会经过相关人员之手，数据的原始性和真实性很难得到保证，最终用户拿到数据的时候，数据可能已经失去了使用的价值。

1.5 实时机器学习的分类

按照实际应用中采用的方式不同，实时机器学习可以分为硬实时、软实时和批实时三种模式，下面将分别进行介绍。

1.5.1 硬实时机器学习

硬实时的定义是：响应系统在接收到请求之后，能够马上对请求进行响应反馈，做出处理。硬实时机器学习的主要应用场景是网页浏览、在线游戏、高频交易等对时效性要求非常高的领域。在这些领域中，我们往往需要将相应延迟控制在若干毫秒以下。对于高频交易等场景，更是有不少计算机软件、硬件专家，开发出了各种专有模块以在更短的时间内完成交易，获得超额利润。

在本书写作之时，计算机网络的传输速度仍然是响应延迟的一大主要因素。硬实时机器学习的响应架构往往会试图尽量减少请求处理过程中的网络传输步骤。与此同时，为了达到硬实时的要求，在请求突然增加的时候，往往会采取负载均衡的方法，靠增加服务器的数量来减少响应延迟。

1.5.2 软实时机器学习

软实时的定义是：响应系统在接收到请求的时候，立即开始对响应进行处理，并且在较短时间内进行反馈。软实时机器学习只要求系统立即对请求开始进行处理，最后处理完成所消耗的时间比较少，但是要求不如硬实时严格。软实时机器学习的主要应用场景是物流运输、较为频繁的数量金融交易等领域。例如某物流企业在接到订单之后需要对运输时间、物品风险进行预估，其中需要和多个系统服务进行交互读取，这个时候我们需要系统能够实时地做出处理，但是处理结果可能需要经过数秒才能得到。

由于软实时机器学习对响应延迟的要求有所放松，因此往往会在处理架构中加入分布

式队列这一组成部件。处理的任务会被实时地传输到分布式队列中，而后端的处理程序能响应式地对任务进行处理。与此同时，在请求增加的时候，可以通过分布式队列缓冲到达的任务，也可以通过负载均衡的方法增加处理单元，以保证低延迟。

1.5.3 批实时机器学习

硬实时机器学习和软实时机器学习都是针对具体的单个事件进行处理。与此相对应的，批实时机器学习是指对成批到达的数据进行实时的处理。批实时机器学习的应用场景往往处于后端机器学习模型的训练和数据处理加工上。通过实时训练的模型将会被部署到硬、软实时机器学习架构中，对数据进行处理。

由于批实时机器学习需要对一定时间窗口内的所有数据进行处理，因此批实时机器学习架构中往往也会有一个分布式队列，对时间窗口内的数据进行缓冲和加工。在数据流向增加的时候，可以通过加大分布式队列的容量，提高分布式队列的处理能力；也可以通过增加处理单元的方法来提高处理能力，以保证低延迟。

1.6 实时应用对机器学习的要求

现今每年都会发表成千上万的机器学习相关的论文，其中不乏表现突出的方法论，但是并不是所有的机器学习模型在实际应用中都适用。实时机器学习的应用主要有以下几个方面的要求。

1. 模型可扩展性

模型可扩展性需要整个机器学习应用的各个部分均可以轻易地根据实际需要进行扩展。这里的扩展可能是增加新的预测变量，也可能是在新的市场、人群和用户界面中进行使用，还有可能是加入新的架构部件，进行可视化等操作。

2. 模型运用低延迟性

低延迟性是实时机器学习应用区别于其他机器学习应用的核心。根据定的义的不同，低延迟的界定也会有所不同。对于网页、交互式游戏等应用场景，低延迟需要整个机器学习后台在少于 10 个微秒内完成反应；与此相对应的是，对于后台数据分析、作弊检测等场景，低延迟要求整个机器学习后台能在少于一分钟内完成作业即可。

3. 训练数据私密性

训练数据私密性是指，模型的用户能否通过逆向工程的办法，倒推出模型训练数据集的内容。如果训练数据集的内容可以被轻松倒推出来，那么可能会对训练集数据提供者的隐私和经济利益带来负面影响。这是近几年刚被机器学习业界意识到的一个重要问题。

1.7 案例：Netflix 在机器学习竞赛中学到的经验

美国领先的付费视频公司 Netflix 在机器学习、系统推荐方面都做出了卓越的贡献，早在 2007 年，Netflix 就率先提出了百万美元大奖，奖励在 Netflix Prize 竞赛中优胜的队伍。Netflix Prize 通过为期三年的竞赛，积累了机器学习宝贵的第一手资料，成为了机器学习中的经典案例，这里我们介绍以下两个方面。

1.7.1 Netflix 用户信息被逆向工程

Netflix Prize 进行影片推荐预测时，使用的数据包括用户名、影片名、评价日期、评价等级等信息，为了防止泄露用户个人的隐私信息，Netflix 对用户名进行了加密处理。

尽管如此，德州大学的研究人员仍然通过逆向工程成功得到了一些用户的个人信息。他们是怎么做到的呢？原来 Netflix 用户在评价一个影片的时候，往往还会去互联网影片库 IMDB 上转载自己的评论。德州大学的研究人员将 Netflix 数据集中的评论和 IMDB 中的评论按照评论日期进行配对，很快就发现了具有上面行为的若干用户，其中不乏具有隐秘性取向的用户。这一研究结果一经发出之后，这些用户的生命安全直接受到了威胁，这也直接导致了 Netflix 在 2010 年遭到了以上用户的起诉，并且取消了 2010 年以后的所有竞赛。

通过这一案例，我们意识到了在设计机器学习应用的时候一定要把用户隐私保护放在第一位。一些社会边缘个体特别容易因为自己的行为特征与大众不同而被模型泄露。

1.7.2 Netflix 最终胜出者模型无法在生产环境中使用

2009 年 Netflix 最终胜出的队伍为 BellKor，该队伍是由四个队伍混合而成的。为什么要混合队伍呢？笔者曾有幸亲自向 BellKor 成员之一的 Michael Jahrer 请教。故事是这样的，在比赛进行到了白热化阶段之后，来自雅虎、贝尔实验室、Commendo Research and Consulting 和 Pragmatic Theory 这四个队伍得到的结果都不相上下，这个时候，往往要在进行大量的参数调校后，模型才会有很少一点的提升。

2009 年的时候，机器学习领域已经出现了 Emsemble 的概念。Emsemble 的意思是通过混搭来源不同的模型的结果，取长补短，以得到更为强大的模型。很自然的，上面这四支队伍先后决定合并成为一个大集体，最后取得了 Netflix 比赛的最终胜利。

比赛确实是结束了，运用 Emsemble 过程带来的负面影响是，最终模型是由上百个小模型组成的，每个小模型都可能是由不同的语言来写成的，需要自己特殊的预处理程序，而且还需要独立的模型训练架构。虽然按照约定，Netflix 享有最终模型的使用权，但是实际上由于训练和运用模型的复杂性，Netflix 至今也没有将上述模型运用到实际应用中去。

通过这一案例，我们可以学到，先进、前沿的机器学习模型固然很重要，得在运用的时候仍然要考虑到训练、运用的复杂性。一切从实际出发，也是本书全文的贯穿思想。

1.8 实时机器学习模型的生存期

进行实时机器学习开发必须考虑生存期。生存期是指一个系统从提出、设计、开发、测试到部署运用、维护、更新升级或退役的整个过程。若在生存期设计上出现了数据，那么在后面的使用中就会出现各种各样的瓶颈阻碍应用产生价值。

从软件工程的角度上讲，开发实时机器学习也遵从构思、分析、设计、实现和维护五个步骤，这五个步骤可能会循环往复，随着业务的发展进行多次迭代。实时机器学习模型的应用由于其技术的特殊性，也具有自己的小型生存期，其中包括数据收集、数据分析、离线手工建模评测、上线自动化建模评测这四个方面。如图 1-1 所示，离线手工建模评测、上线自动化建模评测这两个部分主要是靠监督式机器学习。而数据分析主要是依靠非监督式机器学习和统计数据分析。

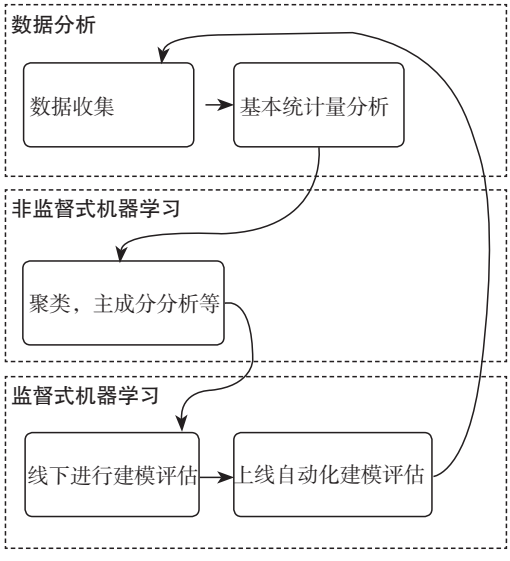


图 1-1 实时机器学习模型的生存期

值得一提的是，进行上面这四个步骤的前提是机器学习模型能够给组织和用户带来价值。但是，众多开发人员甚至是领导层都不愿意面对的一个问题是：我的模型真的有用吗？

对于一些非机器学习大数据类的初创公司来说，在用户数量并不太多的情况下，用非监督式机器学习进行少量数据分析，然后用人力进行反馈，反而有可能会取得更优良的投资回报率。笔者道听途说得知国内一些门户视频网站，就算在公司都已经上市之后，仍然还在使用人工选择的方式进行视频推介，甚至还取得了尚可的效果。

如果机器学习不能给组织带来直接效果，就算有高层支持，对于机器学习从业人员来说也不是很好的职业选择。在机器学习能为组织带来效益的情况下，让数据说话，从业人员才能够不断进行深挖，并得到更多的锻炼和领域洞见；与此相反，如果所建立的系统听

起来很好，但是却没能带来相对应的效益，那么这样岗位上从业人员的工作重心就会像浮萍一样随波逐流，被公司政治利益驱动，长期来说这样很不利于从业人员的个人发展。

机器学习实战的最高境界，就是知行合一，在创造科技前沿作品的同时，能够为个人、组织和社会带来效益，这也是本书写作的指导思想。

在下面的章节里，我们将会从更实际的角度出发来探索实时机器学习的应用。其中，第 2 章到第 4 章，我们将会介绍监督式机器学习模型，并且学习建模的工具 Pandas 和 Scikit-learn；第 6 章到第 9 章，我们将会介绍实时机器学习的架构，并且学习使用 Docker、RabbitMQ、Elasticsearch 及数据库等重要组成部分。