在经过了与业务方多次沟通和迭代后，模型的效果终于获得了大家的一致认可，我们的模型进入了生产待命的状态，即将迎来曙光。不过需要注意的一点就是，我们的目标是业务需求，而数据挖掘产出的结果，不管是预测型的还是关联型的，都要结合业务场景，融入业务流程中去。

模型部署

我们的业务形态不同，部署的方案也就不同。你的模型可能独立部署成服务运行，也可能嵌入到其他的项目代码中去，但是都逃不脱一个本质，那就是**回归业务**。所以，在这个阶段，我们就要考虑具体的业务场景了：模型如何保存？如何根据业务需求优化？以及如何最终上线服务？下面为大家详细解答。

模型的保存

在有了优秀的模型之后，首先就是要把它保存好，以方便应用。我们要给它定义一个好的名字，甚至需要维护一个详细的文档来记录模型所使用的算法、训练数据、评估结果等信息。因为在整个过程中会进行很多次训练，产生很多的模型，或者要把很多的模型组合在生产中使用，同时还需要跟后面的重新训练进行效果的对比，有时候模型的训练和部署可能由不同的人来实施，如果保存时没有注意到这些问题，很有可能导致出现混乱的情况。

所以我们要制定好模型保存的规范，包括存放的位置、名字的定义、模型所使用的算法、参数、数据、效果等内容，防止发生比如遗忘、丢失、误删除，甚至是服务器崩坏等人为的事故，造成不必要的损失。

模型的优化

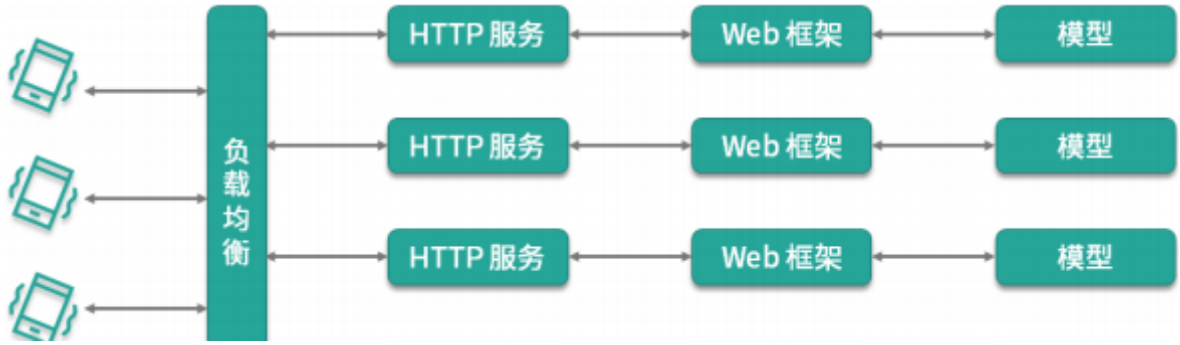
在模型训练阶段已经讲了一部分模型优化或者说提升效果的方法，为什么这里又出现了模型的优化呢？这主要是因为在模型部署应用阶段的很多限制条件在模型训练阶段并不会显现出来，模型训练阶段优化所追求的目标是效果要尽量好；而在模型应用阶段优化所追求的目标是在效果尽量不降低的前提下，适配应用的限制。

比如，在对时延要求比较高的场景下，如果业务应用无法忍受模型的响应时间，那么我们就需要想办法解决，是增加机器还是降低模型的复杂度以提高速度；还有，在对模型大小要求比较高的场景下，我们期望把人脸识别模型部署到一个摄像装置的小型存储芯片上面，那么模型的大小就会受到限制，需要考虑降低模型的参数维度等。

离线应用还是在线应用？

想想我们的业务需求，如果是要使用新闻分类的类别标签结果，实时分发到用户 App 中，那我们的分类模型就需要部署成在线的应用服务以实时响应新的内容请求。如果我们只是需要对一批已有的新闻数据进行分类处理，而之后只是使用这些结果而不会新增新闻内容，那我们的模型就可以离线运行，把存储的新闻处理完就可以了，或者是每隔一段时间去处理一下新的数据。

这里我主要来说一下在线应用。随着算力和业务需求的不断提升，~在~公司里有越来越多的在线服务需要数据挖掘模型的支撑。这里我画了一幅可能的服务架构：



在通常的业务中，有很多客户端在发起请求，我们要在不同的服务器或者 Docker 中部署多个环境及模型，然后使用 Web 框架和 HTTP 服务响应请求，当然中间还有一层负载均衡去处理请求负载转发，以平均服务器的压力。

一个方案

通常算法工程师或者数据挖掘工程师都忙于解决模型问题，到了模型部署阶段就头疼不已，尤其是需要大规模并行的线上服务，可能会耗费很多时间。我在这里介绍一个简单的部署方案，希望能够为大家节约一点时间。

Flask Web 框架：在日常的任务中可以使用 Flask 作为构建我们的 Web 服务框架，它是用 Python 来实现的。

Gunicorn HTTP 服务：可以理解成 HTTP 服务器，需要注意的是 Gunicorn 只能运行在 Linux 服务器上面。

Nginx 负载均衡：Nginx 是一个功能很强大的 Web 服务项目，它可以用作负载均衡器，很多大公司都在使用。负载均衡用于通过集群中的多个服务器或实例将工作负载进行分布，目的是避免任何单一资源发生过载，进而将响应时间最小化、程序吞吐量最大化。在上图中，负载均衡器是面向客户端的实体，会把来自客户端的所有请求分配到集群中的多台服务器上。

客户端：业务的具体场景，可能是手机 App，也可能是其他服务器应用，客户端会向托管用于模型预测的架构服务器发送请求。比如今日头条 App 页面下拉，将会调用推荐算法模型进行推荐内容的计算。

当然，这里的方案并不是唯一的，在实际的工作中也有很多其他的工具具备同样的功能，可以根据自己环境和需求灵活选用。如果是在一些大公司，这些环节可能甚至不需要你考虑，会有一些成熟的平台项目来帮你实施算法模块，不过了解一下对自己也有帮助。

项目总结

记录项目经历，学会总结和反思

在项目部署上线之后，我们的项目算是告一小小的段落了，但是不要忘了对我们的工作进行总结，整理一下文档。总结的内容包括：从项目的需求发起，到数据准备，再到模型训练、评估、上线，这些环节都遇到了什么样的问题，我们解决了什么问题，又有哪些问题尚未解决，如果在时间等条件充裕的情况下还可以做哪些尝试。同时认真地做一下反思，把整个项目中的重点知识内化成自己的能力。

良好的项目总结文档会带给我们很多便利，方便我们在项目迭代时查阅，同时也是对自己工作的总结，在做过很多项目之后，这些积累将成为你宝贵的经验与财富。

多考虑一点，如何适合更多场景

当你完成了一个又一个需求之后，会发现很多需求似曾相识，但是又总有不一样的点。所以我们的数据挖掘模型或结果能不能做成统一的服务，能不能应用在更多的地方？比如，我们在做标签系统之初，业务 A 有一个分类标签的需求，业务 B 有一个分类标签的需求，业务 C 有一个分类标签的需求，那我们就要一个一个地去做，A 模型给 A 用，B 模型给 B 用，C 模型给 C 用。但是过了不久又有 D 来提一个

类似 AB 的需求，所以我们就开始规划一个面向全公司更底层的标签体系架构，以应对各种类似的业务。多考虑一点就可以把数据挖掘前置到业务需求之前，最终形成完整的业务数据闭环，而不需要再进行冗余的开发。

监控与迭代

模型终于部署上线，但这不是说我们的数据挖掘工作就已经结束了。如今社会变动的速度也十分迅速，我们已经做好的模型很可能在经过一段时间的运行之后就不再符合当前的线上情况了，有可能我们的产品形态、业务需求发生了变化，这在互联网公司再正常不过了。为了我们的模型保持良好的效果，需要有一份迭代计划去维护和更新我们的模型。

模型的监控

要了解什么时候该做出调整，我们需要有一套监控的策略。这里所说的模型监控不仅仅是对服务运行状况的监控，更是对于模型输出的结果进行监控，以查看模型在线上运行的效果是否符合预期，是否有比较大的变化。假设我们的新闻分类模型已经部署上线，每一条新闻流过将会被标注上若干个标签，然后进到推荐系统被召回排序分发给用户。

模型的监控可以从三个方面入手，分别是结果监控、人工定期复审以及 Case 收集与样本积累。

结果监控

结果监控主要是针对一些具体的指标进行监控，包括我们在评估环节用到的准确率、召回率等，另外还可以根据具体产出的结果在业务中的效果进行监控。

首先可以想到的是针对每天新闻的分类标签进行排名统计，来查看每个标签的占比情况与我们的初始数据是否接近；还可以监控到一些数据较少的类别是否能够被预测出来，这属于稳定性指标。

其次，在推荐系统中，我们还会有 CTR（点击率预估）这样的总体指标，可以对标签与 CTR 的关系进行计算，来查看每个标签的 CTR 情况。

在一些 App 中还会有主动负反馈，让用户自己选择不喜欢的标签。通过分析这些数据可以知道我们的模型结果是否还符合业务场景，是否需要做出一些调整。在一些长期的大型项目中，我们甚至需要建设一些 Debug 平台，对这些数据进行持续的、可视化的监控，观察每天每周的变化情况。

人工定期复审

第二个方法就是定期进行人工复审，该方法主要针对业务需求准确率的情况进行评估，查看当前的模型效果是否还满足业务的需求，准确率情况是否有所变化，同时也可以跟业务进行沟通评估，确认当前的情况是否需要对模型进行重新训练。

Case 收集与样本积累

第三个，在整个监控的过程中，也需要进行 Case 的收集。通过具体的 Case 我们可以知道当前的模型存在哪些问题，有些 Case 可能是由于模型本身的问题造成的；有些 Case 是由于我们的业务场景数据发生了变化造成的。通过对收集的 Case 进行分析，也可以知道我们需要往哪个方向去优化模型。同时，收集足够的 Case 也可以作为我们重新训练的样本积累，所以要注意在前期准备数据时遇到的数据不充分或者不准确的情况，也可以在收集环节重点关注，以补全我们上一版训练时的一些缺失情况，这样在下一次训练迭代时能够有更好的样本集。

重新开启

此时此刻，再回想一下最开始画的那张数据挖掘流程图。虽然我这里按照顺序一步步进行讲解，但实际上，每一个步骤并不是严格独立存在的，比如我们在准备数据阶段发现数据无法解决业务需求时，那就要返回去重新讨论业务需求与数据的问题；在训练模型阶段发现数据与模型无法匹配，或者如果要更换其他模型时，那要回到准备数据环节；如果在模型评估的时候发现效果达不到预期，那可能要回到准备

数据环节重新处理数据，甚至要回到理解业务阶段。而在这里，我们的模型已经上线应用，但不代表工作已经完全结束了。

我们的公司无时无刻不在发生着变化，用户群体、用户行为也在发生变化，数据采集的技术也在不断更新，所以我们的模型也不是一成不变的，在经过一段时间的运行之后，它的能力可能无法适配当前的情况，此时我们又要回到故事的开头，去思考业务，理解数据了。

所以说，时刻准备好重新开始。

总结

到了模型应用这一课时，关于数据挖掘流程相关的部分就已经讲完了。在这一课时，我介绍了一些关于模型保存、模型优化、模型部署的思路，又讲了关于项目总结，乃至模型监控等内容，知识点比较零散，难免会有一些疏漏和不足，不过这些都是平时工作中的一些总结，权当抛砖引玉。如果关于这部分内容有什么好的想法和实践，也欢迎你提出来，集思广益、共同进步。

下一课时，我们将进入具体的算法模块，介绍算法的思想、优缺点，看看如何使用这些算法。