本次课程讲解了模型集成、模型蒸馏和模型部署三个方面，最后部分进行了代码讲解以及多模型集成的方案运行，从而引出本次课程的作业。

模型集成的主要点是将多个模型的结果进行融合，从而得到更好的模型，其主要的思路是训练多个模型解决相同的问题，并将它们结合起来获得更好的集成结果，从而得到更精确和/或更鲁棒的模型，其数学思想是将弱学习器的偏差和方差结合起来，创建一个强学习器，从而获得更好的性能。

偏差又称为拟合能力，度量学习算法的期望预测与真实结果的偏离程度，描述的是预测值的期望与真实值的差距。

方差度量了同样大小的数据集的变动所导致的学习性能的变化，即刻画了数据扰动造成的影响，描述的是预测值的变化范围。

偏差所导致的欠拟合是可以通过挖掘更好更多的特征进行解决；方差所导致的是过拟合的情况，一般是通过增加数据、减少特征、换用简单模型、正则化、dropout等以及交叉验证的训练方式都可以帮助解决过拟合的问题。

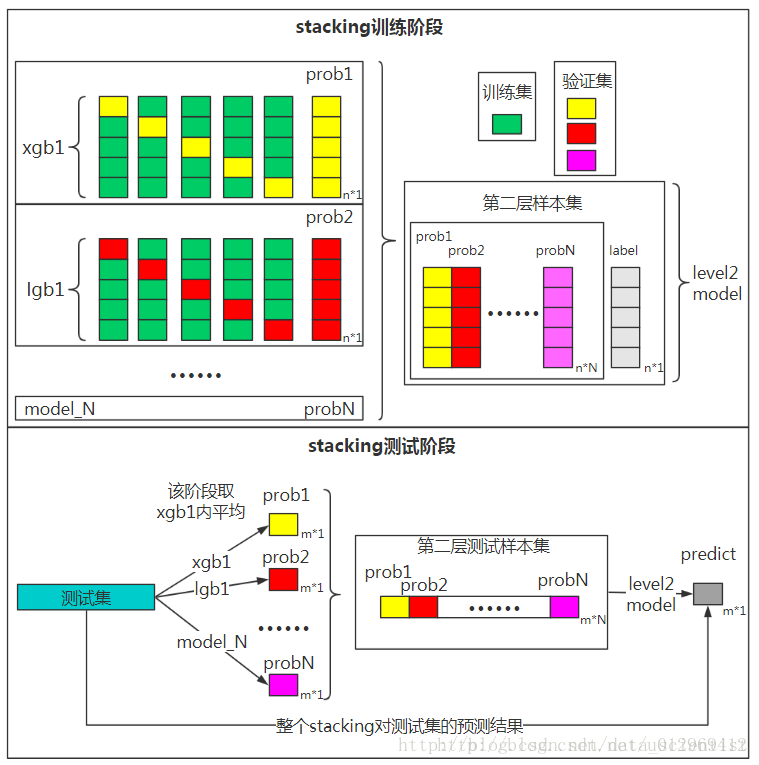
Bagging的思想是基于投票思想的多数票机制，属于套袋集成技术，较常用于深度学习方面；而boosting思想在传统机器学习方面用的比较多；分层模型集成框架stacking（叠加算法）主要用于工业上的子模型集成，将子模型的结果当作特征作为stacking框架中形成下一个子模型的结果。

基于投票的思想是分别使用全部语料训练n个弱分类器，对每个弱分类器的输出结果进行投票，每个样本取投票数最多的那个预测为该样本的最终分类预测，投票思想不一定得到更好的结果，也有可能带来性能的下降。

Bagging思想的目的是为了降低方差，Bagging的投票思想有所不同，每个分类器使用的是全部样本的随机抽样，每个分类器都是使用不同的样本进行训练得到n个弱分类器，对每个弱分类器输出预测结果并投票，每个样本取投票最多的预测作为该样本最终分类预测。

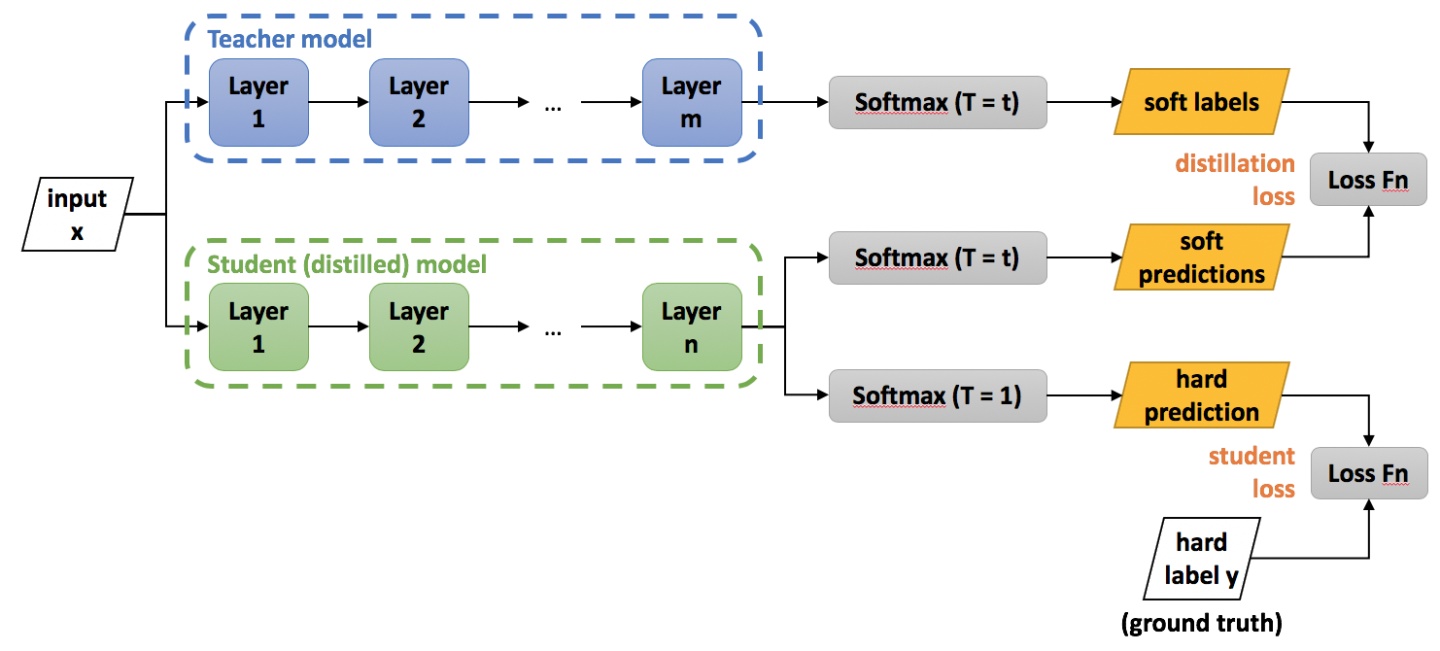
Boosting思想的目的是为了降低偏差，即提高弱的分类器的性能，按照顺序进行学习，主要是重点在学习之前分类错误样本进行多数票机制一步一步减小偏差。Adaboost思想是其中的自适应增强方法，首先初始化权重，然后对于每个轮数所有样本进行分类，计算权重错误率并根据错误率来重新更新下一步的权重。

Stacking集成算法可以理解为一个两层集成过程，第一层含有一个分类器并把预测结果提供给第二层，第二层分类器通常是逻辑回归。如下所示



Blending为了避免交叉验证出现的数据泄漏，而是建立一个holdout集，例如10%的训练数据，那么第二阶段就是基于第一阶段模型对这10%的训练数据的预测值进行拟合，其优点在于更加简单，避开了信息泄漏问题，不需要分享自己的随机种子；弱点也是比较容易看出来的，在于使用了很少的数据blender可能会过拟合（其实大概率是第一点导致的）stacking使用多次的CV会比较稳健。

模型蒸馏的来源是使用ensemble模型对大量的unlabeled data打标签，这部分样本去训练一个更加轻量级的模型，主要的研究突破是论文distilling the knowledge in a neural network，是用新的小模型去学习大模型的预测结果以及泛化能力。如下图所示



其中，hard labels指的是原始标签，soft labels是teacher model得到的标签，soft target作为温度，当T=1时就是一个传统的softmax函数，温度越高则softmax函数越平滑，反之T的温度越低softmax函数就越陡峭，从而尽可能将模型中的soft target比hard label携带更加有用更加丰富的信息。模型蒸馏一般会进行信息对齐，主要有三种对齐方式：1）对齐样本之间的关系；2）对齐网络中间的输出；3）对齐网络结构的信息。BERT的语义表示基本框架已经确定，后续大多数模型都是提高精度和速度，模型应用到线上服务的时候对服务的计算资源要求非常高，通过知识蒸馏之后得到只有原始模型40%大小的学生模型，节省时间，效果相近，有利于进行模型的部署。

项目部署：在线项目流程为技术调研->数据获取->算法训练模型持久化->模型部署->线上监控->提供服务，在项目部署中包含（特征在线化处理、模型在线化处理、线上环境的配置、模型版本的迭代、出错回滚、稳定性保证、服务降级、小流量实验、架构设计等等）， 算法工程师关注模型性能和效果，算法架构师则关注服务的稳定、可用性以及性能。模型的服务方式包括离线预测和在线预测两种方式。

部署方式包括两种方式：server framework + deep learning framework api 或者 server framework + deep learning serving

第一种方式中Server启动时进行模型加载和数据复制，而预测是根据请求推理；第二种方式 的server framework可以是任何语言，然后将输入转化为模型输入。

算法人员需要考虑的问题包括：长耗时操作、一次性操作、接口控制、处理时长、模型大小、服务降级策略、回滚兼容策略、实验方式、多模型并行或串行