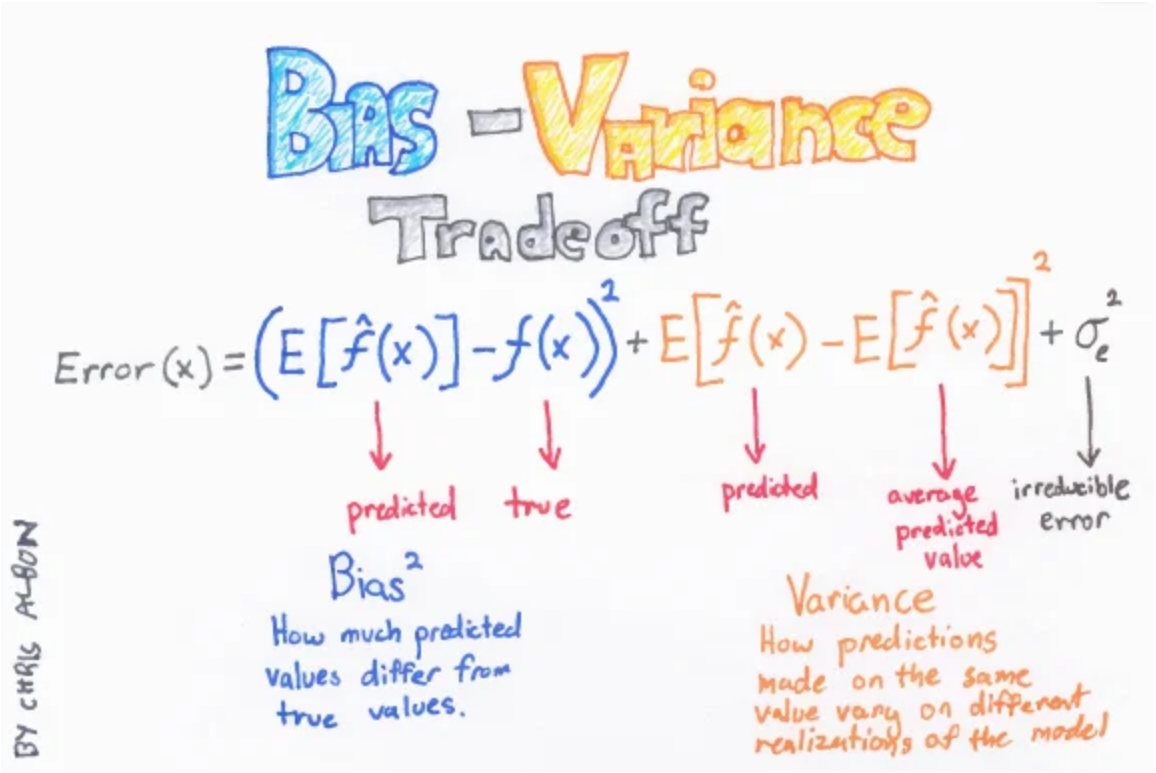
集成学习

Minimal, System, Insight, Programming

比赛中集成学习，但要用好模型集成可能并不是那么简单，本文将讨论讨论：

1. 模型总体误差分解，过拟合，欠拟合
2. 为什么集成学习能降低总体误差
3. Stacking
4. Bagging
5. Boosting

Bias and Variance



假设有训练数据集D，包含样本点$(x_1, y_1), \ldots, (x_n, y_n)$，存在一个带噪音的真实函数$y = f(x) + \epsilon$，噪音$\epsilon$均值为0，方差为$\sigma^2$，我们希望通过数据集$D$训练模型$\hat{f}\,(x; D)$尽可能毕竟真实函数$f$，使得任意训练数据集以外的样本误差最小化，即最小化误差函数MSE:

$$\mathrm{E}_{D,\varepsilon}\left[\left(y - \hat{f}\,(x; D)\right)^2\right] = \left(\mathrm{Bias}_D\left[\hat{f}\,(x; D)\right]\right)^2 + \mathrm{Var}_D\left[\hat{f}\,(x; D)\right] + \sigma^2$$

where 偏差(bias)部分：

$$\mathrm{Bias}_D\left[\hat{f}\,(x; D)\right] = \mathrm{E}_D\left[\hat{f}\,(x; D)\right] - f(x)$$

and 方差(variance)部分:

$$\mathrm{Var}_D\left[\hat{f}\,(x; D)\right] = \mathrm{E}_D[(\mathrm{E}_D[\hat{f}\,(x; D)] - \hat{f}\,(x; D))^2].$$

这里我们已经通过公式推导将目标函数分解成三项：

a) 偏差项（bias）是采用不同训练数据集D时，模型预测和真值的偏差，可以看成不同的模型假设带来的误差，比如真实函数是一个非线性函数f，而我们采用线性回归模型$\hat{f}$进行建模，由于这一假设，估计我们的线性模型估计$\hat{f}$将存在误差。通常越复杂的模型偏差更小，复杂模型的假设空间更大，对真实函数逼近能力更强，偏差更小。

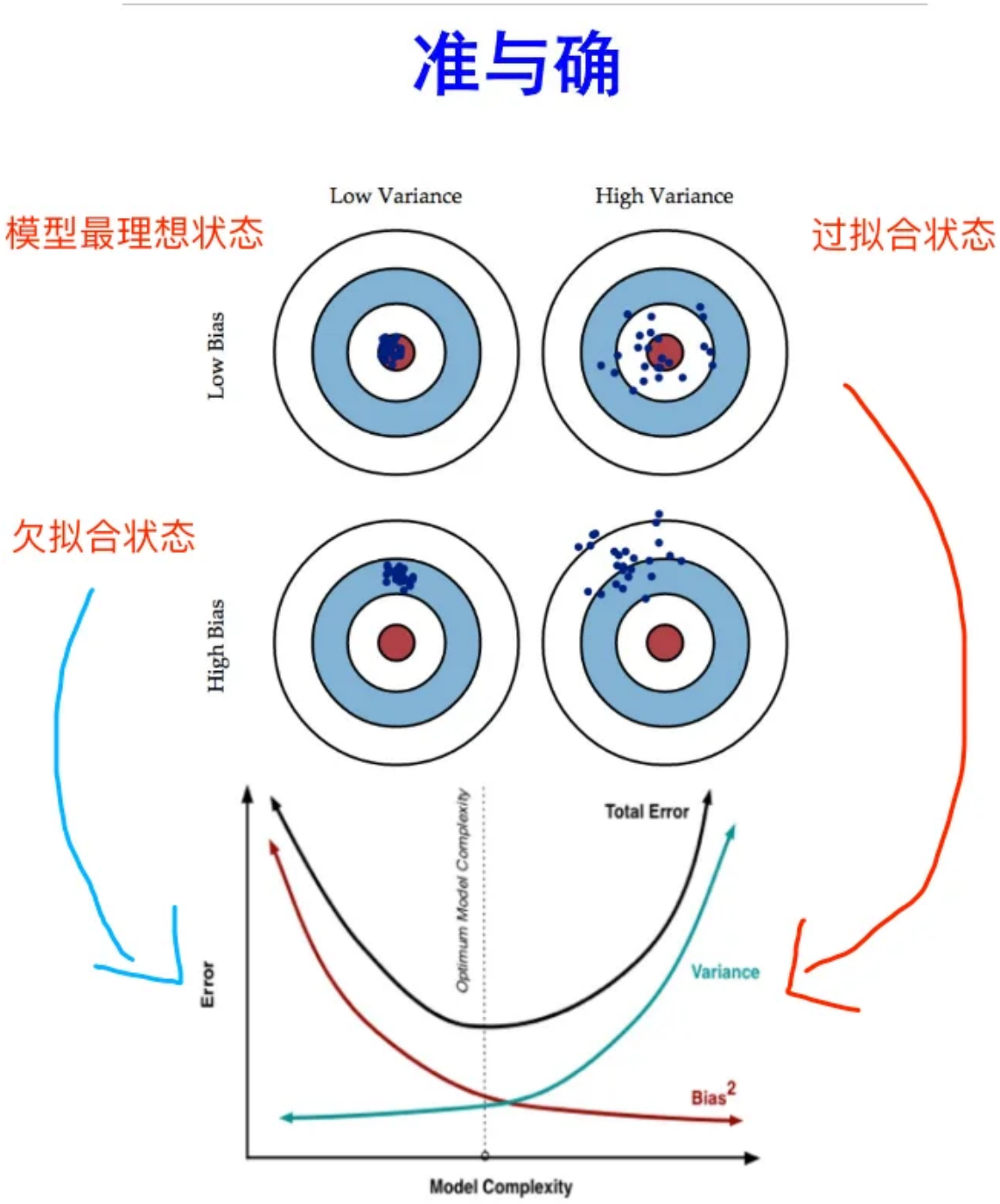
b) 方差项（variance）为采用不同训练数据集D训练模型时，模型对同一个样本预测值的波动大小，通常越复杂的模型越敏感，这意味着采用不同数据训练的模型差异会很大，导致对同一个样本预测值的波动较大，即方差很大。

c) 为随机误差项，无法预测。

过拟合（overfitting）与欠拟合（underfitting)

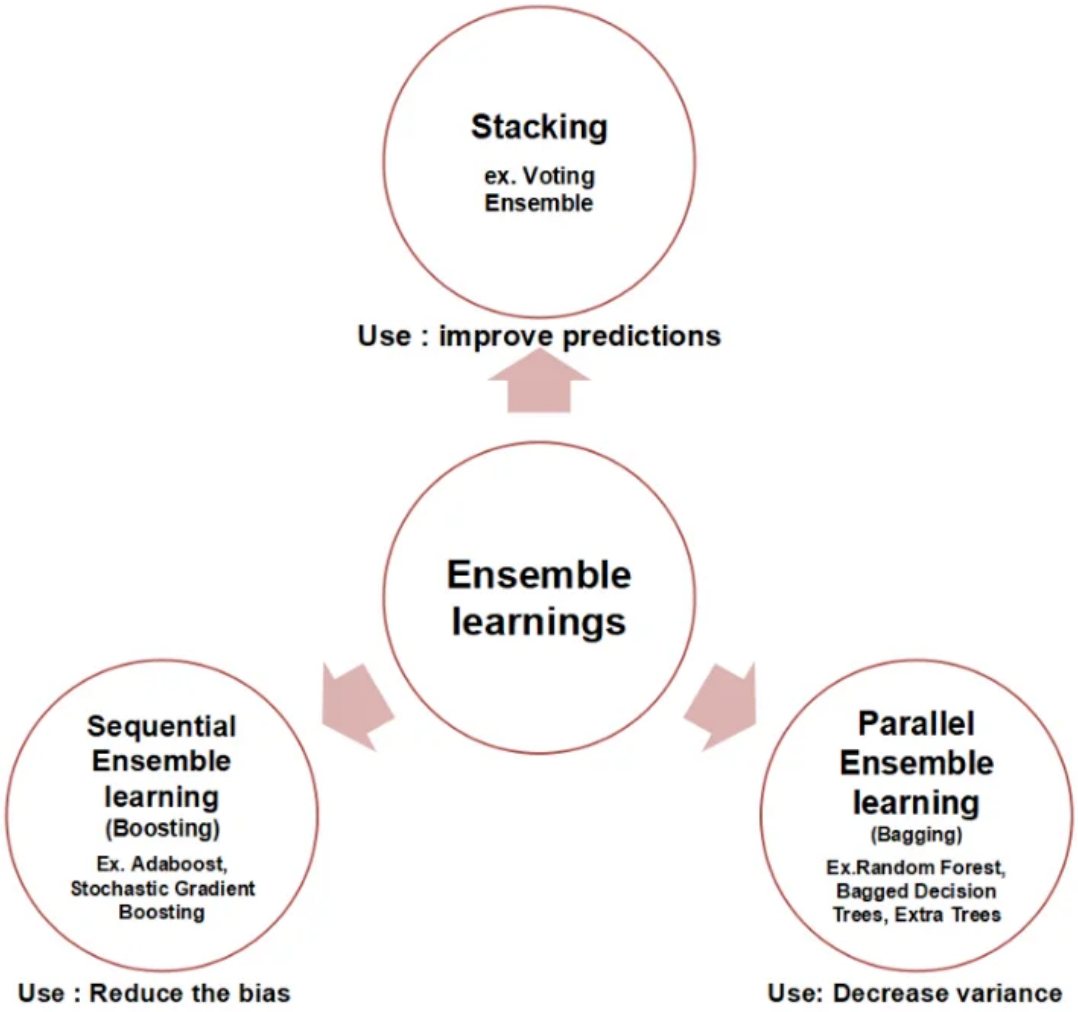
1. Overfitting，当我们的模型太复杂（比如很深的决策树、非常多的特征工程、大型深度学习网络等），模型偏差较低，方差较大；
2. Underfitting，当我们的模型太简单（线性回归，无特征工程，浅层感知机等），模型偏差较大，方差较小；

过拟合和欠拟合都不是我们想要的模型状态，如下图所示，最好的状态时偏差和方差都处于比较小的均衡状态，这时候总体误差最小。



集成学习为什么有效?

集成学习指建模时训练多个基模型，预测时候融合多个模型预测结果，降低总体误差的学习方法。集成学习方法有很多，总体上可以分为三类：Stacking，Bagging 及 Boosting。



降低模型方差

可以看出不管是哪种模型集成方法，都有一个共同特点：**融合多个模型预测结果**。由统计知识可知，多个独立同分布随机变量满足：

设 $X_1, X_2, \ldots, X_n$ 是独立同分布的，满足$E\left(X_i\right)=\mu, D\left(X_i\right)=\sigma^2, i=1,2,3, \cdots, n$

则随机变量$\overline{X}=\frac{1}{n} \sum_{i=1}^n X_i$ ，满足 $E(X)=\mu, D(\overline{X})=\frac{\sigma^2}{n}$ 。

下证：

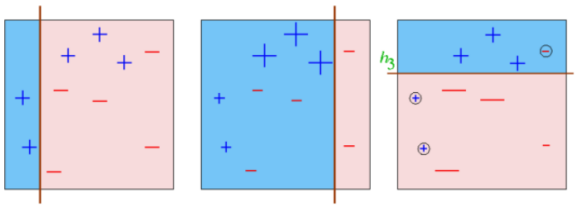
$$E(\overline{X})=E\left(\frac{1}{n} \sum_{i=1}^n X_i\right)=\frac{1}{n} \sum_{i=1}^n E\left(X_i\right)=\frac{1}{n} \cdot n \mu=\mu$$

$$D(\overline{X})=D\left(\frac{1}{n} \sum_{i=1}^n X_i\right)=\frac{1}{n^2} \sum_{i=1}^n D\left(X_i\right)=\frac{1}{n^2} n \sigma^2=\frac{\sigma^2}{n}$$

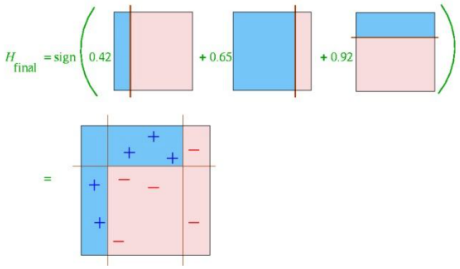
其中随机变量的简单算术平局$\bar{x}$可以看成模型融合结果，只要不同模型预测结果满足独立同分布，模型预测方差将从原本的σ^2变为$\frac{\sigma^2}{n}$，大幅度降低了总体误差中的方差部分。当然这是最理想情况了，一般来说模型间都存在一定的相关性，相关性越弱，融合后预测方差越低。因此集成学习时，构造模型间的差异化是重中之重，在stacking中，我们通过采用不同的模型类型构造差异化，在bagging中我们采用不同的训练数据构造差异化。

降低模型偏差

1. 单独训练多个简单线性模型（分别边界是一条直线）

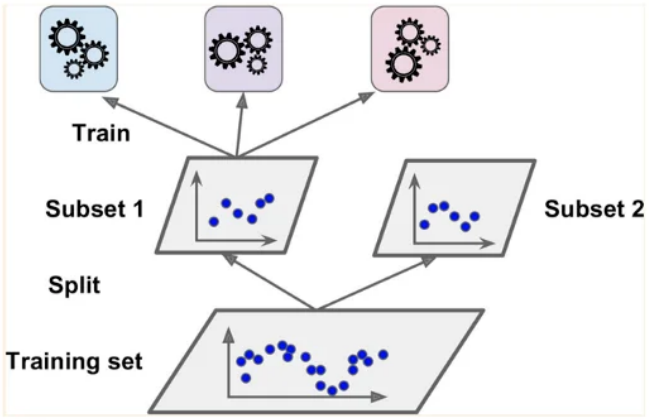


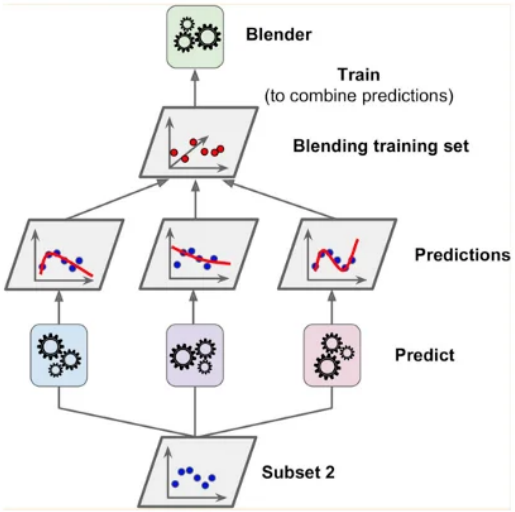
2. 融合多个简单线性模型，获得非线性决策边界



从上图可以看出，融合多个弱模型（相对概念，指复杂度比较低的模型），可以降低模型偏差，获得更高的模型精度。

Stakcing 模型集成





Stacking 模型集成方法通过一个meta model 对不同类别的基模型预测结果进行集成，不同于bagging & boosting，stack采用的基模型通常是不同种类的，算法流程如下：

1. 将数据集划分成3部分，subset1, subset2, subset3
2. 在subset1上训练不同种类的基模型(比如：逻辑回归、决策树等)
3. 在subset2上，基模型进行预测
4. 在subset2上，使用3)中的预测结果作为输入，训练一个meta模型，通常meta模型我们尽量采用复杂度较低的简单模型，比如线性回归或逻辑回归，避免过拟合
5. 在subset3上，基模型进行预测，预测结果输入到meta模型进行预测，得到最终预测结果，测试模型精度

Stacking 简化变体 Average / Weighted

很多时候，为了简化stacking模型集成或降低过拟合，我们不需要训练meta模型，可以直接将基模型预测值进行平均或进行加权平均即可。这个时候我们的工作流变成:

1. 将数据集划分成两部分，subset1和subset2
2. 在subset1上训练不同种类的基模型(比如：逻辑回归、决策树等)
3. 在subset2 使用基模型进行预测，并对所有基模型预测结果计算平均值（或进行加权平均）得到最终预测结果，测试模型精度

如果采用加权平均时，一般赋予精度更高的模型更高的权重，比如有模型A/B/C，精度模型A>B>C，可以给予权重（A: 0.5, B: 0.3, C: 0.2)。如果你试图搜索一个最佳权重，这时候可以等价于引入了一个线性模型，在搜索线性模型参数，为了防止过拟合，这个时候建议采用完整stacking模型集成方案。

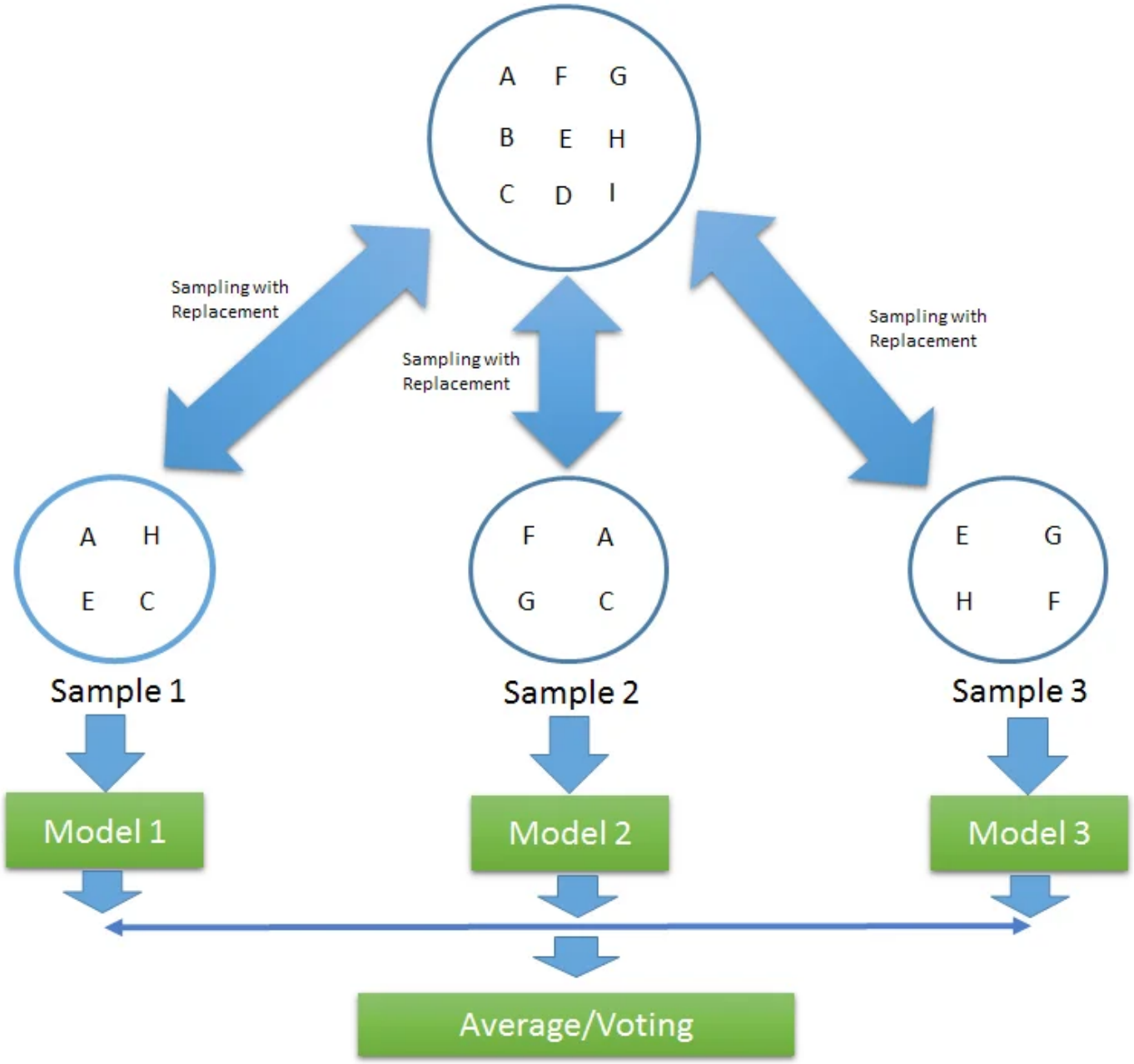
需要注意的是，在进行分类问题建模时候进行平均时，通常采用基模型预测的分类概率值进行融合，例如在使用逻辑回归进行二分类时，使用predict_prob方法可以得到预测概率结果，预测结果为N x 2维度，2为类别数量，分别代表输入负样本和正样本的概率。概率值融合后，使用 `np.argmax(pred_result, axis=1)` 即可得到每个样本最高概率的类别。

```
>>> from sklearn.datasets import load_iris
>>> from sklearn.linear_model import LogisticRegression
>>> X, y = load_iris(return_X_y=True)
>>> clf = LogisticRegression(random_state=0).fit(X, y)
>>> clf.predict(X[:2, :])
array([0, 0])
>>> clf.predict_proba(X[:2, :])
array([[9.8...e-01, 1.8...e-02, 1.4...e-08],
       [9.7...e-01, 2.8...e-02, ...e-08]])
>>> clf.score(X, y)
0.97...
```

stacking模型集成的优势：

1. 灵活，可以任意添加想要集成的模型，并通过设置二阶段模型融合的策略（模型，平均，加权平均）调节过拟合/欠拟合情况
2. 同时降低

Bagging模型集成



Bagging模型集成算法流程：

1. 通过放回抽样得到多个训练集
2. 在不同的训练集上训练基模型（一般采用相同类型的模型）
3. 对模型预测结果进行融合

一般的，我们将使用决策树作为基模型的Bagging集成学习方法叫做**随机森林**，同时由于决策树本身是一种拟合能力比较强的模型，为了最大化模型差异化，在训练每棵决策树时进行特征采样，使得不同子模型使用的数据集不仅样本不同，同时使用的特征也不完全一致。在sklearn库中已经有成熟的随机森林实现，可以直接调用。

```
from sklearn.ensemble import RandomForestClassifier

RandomForestClassifier(
    n_estimators=100,
    max_depth=None,
    max_features='auto',
    max_leaf_nodes=None,
)
```

参数解释：

参数	解释
n_estimators	设置基模型数量
max_depth	树的最大深度
max_features	寻找最佳分割时要考虑的特征数量
max_leaf_nodes	允许的最大叶子节点量

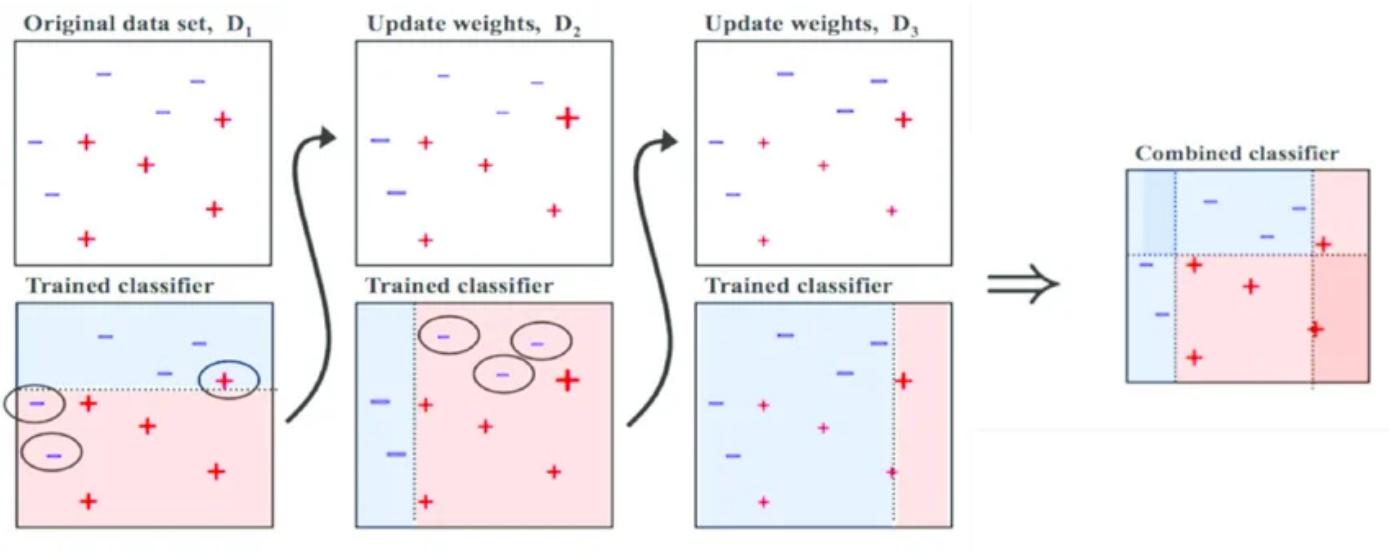
随机森林的特点：

1. 并行的集成框架，适合并行训练，训练速度快
2. 特征随机采样使得在样本特征维度很高的时候，仍然能有效的训练模型
3. 数据集随机采样&特征随机采样的引入，使得随机森林的基模型能构造较大的差异化，降低融合模型方差，提升泛化能力,避免过拟合
4. 因为bagging主要降低模型方差，因此一般选择偏差较小的大型决策树（深度较深，叶子节点数量高）作为基模型，来平衡偏差和方差

Boosting模型集成

与Bagging相反，Boosting采用串行的集成框架，常见的算法流程如下图所示：

1. 从数据集中均匀采样一个子集
2. 在子集上训练模型并对整个数据集进行预测
3. 计算每个样本误差
4. 根据误差大小对数据集样本赋权（误差越大，权重越大），并使用新的权重对数据集进行采样，得到一个新的子集
5. 重复2～4，直到达到预设的迭代次数



Boosting集成学习通过迭代训练，逐步降低模型偏差，因此我们一般采用比较简单的基模型（浅层决策树），避免过拟合发生。 `sklearn` 已经实现了Boosting集成学习方法，可以通过sklearn.ensemble.AdaBoostClassifier直接调用，使用方法和参数与随机森林类似。

Boosting模型集成特点：

1. 串行集成方法，比较难并行化，速度较慢
2. 能有效降低偏差，适合采用简单的基模型

拓展阅读：模型集成代码案例

作业

1. 回答在进行平均融合时，基模型具有怎样的特点能够最大化提升模型性能?
2. 在Kaggle泰坦尼克竞赛中使用逻辑回归、决策数、随机森林建模，对模型预测概率采用加权平均的方式进行模型融合，观察融合效果，是否较单模型有提升?