公式：

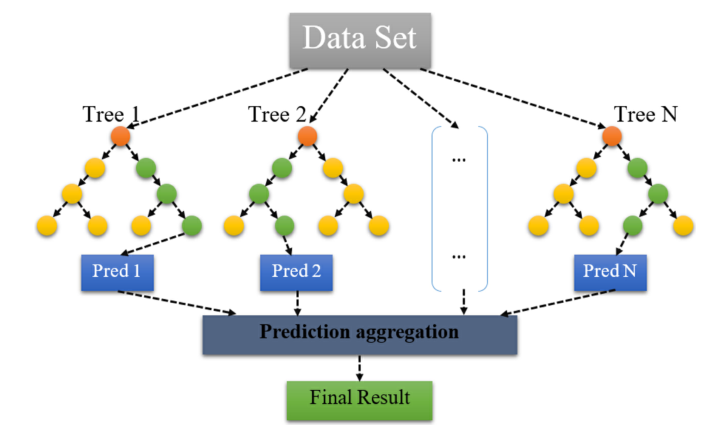




链接1：<https://zhuanlan.zhihu.com/p/26683576>

链接https://blog.csdn.net/zaishijizhidian/article/details/88066882?spm=1001.2101.3001.6650.2&utm\_medium=distribute.pc\_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-2-88066882-blog-99980795.pc\_relevant\_multi\_platform\_whitelistv3&depth\_1-utm\_source=distribute.pc\_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7ERate-2-88066882-blog-99980795.pc\_relevant\_multi\_platform\_whitelistv3&utm\_relevant\_index=5

链接：



**1.**

(1)bagging算法：bagging的个体弱学习器的训练集是通过随机采样得到的。通过T次的随机采样，我们就可以得到T个采样集，对于这T个采样集，我们可以分别独立的训练出T个弱学习器，再对这T个弱学习器通过集合策略来得到最终的强学习器。

(2)GBDT的子采样是无放回采样，而Bagging的子采样是放回采样。

(3)随机森林的应用场景：随机森林既可以用于分类，也可以用于回归。一般适用于数据维度较低，同时对准确性要求较高的场景中。

(4)GBDT的应用场景：它能解决各种分类、回归和排序问题，能优秀地处理定性和定量特征，针对outlier的鲁棒性很强，数值不需要normalize。而且，LightGBM和XGBoost等算法库已经解决了GBDT并行计算的问题（然而scikit-learn暂不支持并行）。

链接2：<https://zhuanlan.zhihu.com/p/26704531>

clf **=** GradientBoostingClassifier(n\_estimators**=**100, learning\_rate**=**1.0, max\_depth**=**1, random\_state**=**0)

n\_estimators：

是弱学习器的最大迭代次数，或者说最大的弱学习器的个数。一般来说n\_estimators太小，容易欠拟合，n\_estimators太大，又容易过拟合，一般选择一个适中的数值。默认是100。在实际调参的过程中，我们常常将n\_estimators和下面介绍的参数learning\_rate一起考虑

learning\_rate：

在建造每一个弱预测器时，这个弱预测器的预测结果都会按照learning\_rate（从0到1.0的浮点数）缩减。learning\_rate与n\_estimators之间存在权衡。

max\_depth：

决策树最大深度，默认可以不输入，如果不输入的话，默认值是3。一般来说，数据少或者特征少的时候可以不管这个值。如果模型样本量多，特征也多的情况下，推荐限制这个最大深度，具体的取值取决于数据的分布。常用的可以取值10-100之间。

random\_state：

是一个[随机种子](https://so.csdn.net/so/search?q=%E9%9A%8F%E6%9C%BA%E7%A7%8D%E5%AD%90&spm=1001.2101.3001.7020)，是在任意带有随机性的类或函数里作为参数来控制随机模式。当random\_state取某一个值时，也就确定了一种规则。

**GBDT梯度提升算法：**

输入：训练数据集T={(x1,y1),(x2,y2),...,(xN,yN)},xi∈X⊆**R**n，yi∈Y⊆**R**；损失函数L(y,f(x))；

输出：回归树(x)

1. 初始化





(2)对m=1,2，...，M

(a)对i=1,2，...，N，计算





(b)对rmi拟合一个回归树，得到第m棵树的叶结点区域Rmj，j=1,2，...，J。

(c)对j=1,2，...，J。计算





(d)更新



(3)得到回归树





<https://blog.csdn.net/u010366748/article/details/111060108>

<https://blog.csdn.net/weixin_39611389/article/details/110597328>

1. **决策树回归：基尼指数；最大深度；后剪枝算法；k折交叉验证Kfold；定义，**

https://blog.csdn.net/sealir/article/details/85085573?ops\_request\_misc=&request\_id=&biz\_id=102&utm\_term=%E5%86%B3%E7%AD%96%E6%A0%91%E5%9F%BA%E5%B0%BC%E6%8C%87%E6%95%B0%E5%AE%9A%E4%B9%89&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduweb~default-3-85085573.142^v52^pc\_rank\_34\_queryrelevant25,201^v3^add\_ask&spm=1018.2226.3001.4187

**基尼指数**：基尼指数（基尼不纯度），表示在样本集合中一个随机选中的样本被分错的概率。

基尼指数（基尼不纯度）= 样本被选中的概率 \* 样本被分错的概率



1.pk表示选中的样本属于k类别的概率，则这个样本被分错的概率是(1− pk)

2.样本集合中有K个类别，一个随机选中的样本可以属于这k个类别中的任意一个，因而对类别就加和  
3. 当为二分类是，Gini(p)=2p(1−p)

**最大深度**：树中的最大层级数量

**后剪枝算法**：

后剪枝是先从训练集生成一颗完整的决策树，然后自底向上的对决策树进行剪枝。

后剪枝则是先从训练集生成一颗完整的决策树，然后自底向上地对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能的提升，则将该子树替换为叶结点。后剪枝决策树通常比预剪枝决策树保留了更多的分支。一般情况下，后剪枝决策树生长更为充分，因此欠拟合风险很小，泛化性能也往往优于预剪枝决策树。当然，由于对树的生长过程不加限制，并且要自底向上地对所有叶结点进行一一考察，所以后剪枝决策树的

训练时间和复杂程度都要大于预剪枝决策树。

文献：[1]路逸行. 决策树误差降低剪枝算法的改进研究[D].山东大学,2020.DOI:10.27272/d.cnki.gshdu.2020.001746.

**k折交叉验证Kfold:**

KFold 将所有的样例划分为 k 个组，称为折叠 (fold) ，每组数据都具有相同的大小。每一次分割会将其中的 K-1 组作为训练数据，剩下的一组用作测试数据，一共会分割K次。

**3.梯度搜索算法：gridsearchcv；**

[**https://zhuanlan.zhihu.com/p/375632445**](https://zhuanlan.zhihu.com/p/375632445)

**https://www.cnblogs.com/dalege/p/14175192.html**

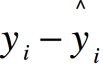
即网格搜索和交叉验证,网格搜索算法是一种通过遍历给定的参数组合来优化模型表现的方法。通过交叉验证确定参数。

GridSearchCV存在的意义就是自动调参,可以保证在指定的参数范围内找到精度最高的参数，但是这也是网格搜索的缺陷所在，他要求遍历所有可能参数的组合，在面对大数据集和多参数的情况下，很难得出结果。

**4.模型评价指数：r2 score**

https://blog.csdn.net/u012735708/article/details/84337262?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522166531726716782395340277%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=166531726716782395340277&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~top\_positive~default-1-84337262-null-null.142^v52^pc\_rank\_34\_queryrelevant25,201^v3^add\_ask&utm\_term=R2&spm=1018.2226.3001.4187



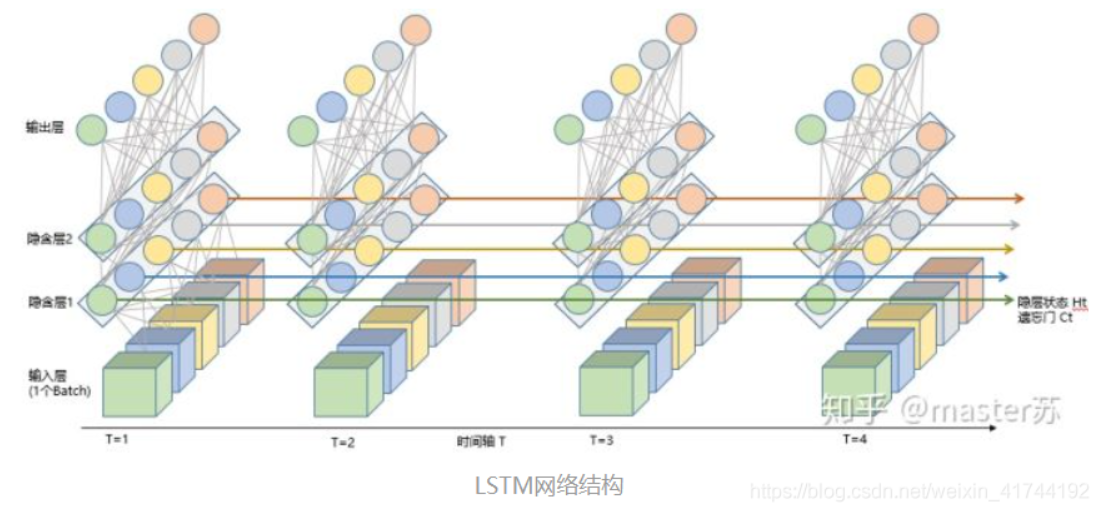
其中，为测试集上真实值-预测值。

可以看出，RMSE=sqrt（MSE）。

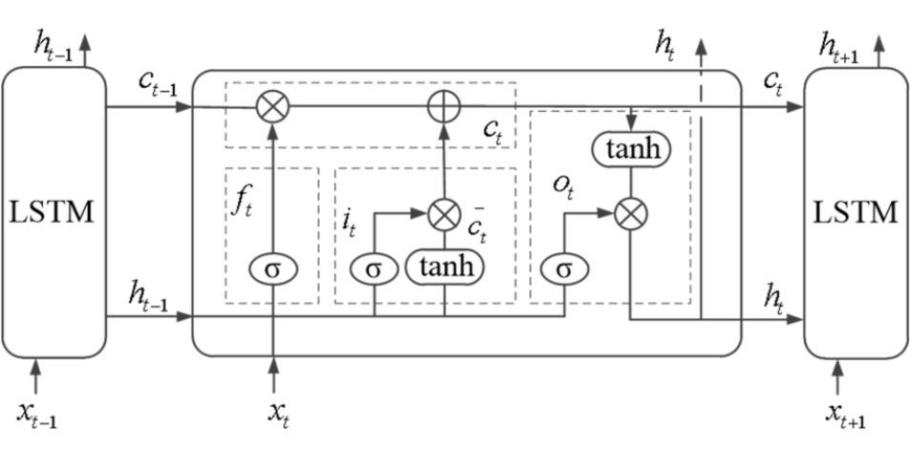


**5.LSTM：**

https://blog.csdn.net/weixin\_41744192/article/details/115270178?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522166531801716782425185739%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request\_id=166531801716782425185739&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~first\_rank\_ecpm\_v1~pc\_rank\_34-18-115270178-null-null.142^v52^pc\_rank\_34\_queryrelevant25,201^v3^add\_ask&utm\_term=LSTM&spm=1018.2226.3001.4187



[1]陈斌,刘悦,尹开浪,方珣.冰雪天气下基于LSTM的跑道温度数据-机理联合预测[J/OL].北京航空航天大学学报:1-16[2022-10-09].DOI:10.13700/j.bh.1001-5965.2022.0579.



**6.CART：**

[1]第一步：输入数据集X=（x1，x2，......，xn)，设变量为V =（v1，v2，......，vk），设定最小Gini 值的阈值，在V中选出目标变量v\*；

[2]第二步：处理变量，将变量V 中每一vj都处理成二元属性变量的形式。其中，若vj已经是一元变量，则不用更变：如果vj，不是元元属性变量，是多元分类 （属性）变量，或者是离散、连续数值型变量，则需要通过最小 Gini 法确定最优分割，即在某一个分割条件下,计算得到的 Gini 值最小，Gini 值的公式为：



该公式表示目标变量v\*按变量vj划分后的 Gimi 值，其中n1和n2表示变量vj,二元化后两个类别(设为1和2）所包含的数据点个数，p1，p2表示vj第1（2）个类别中属于目标变量v\*的第1（2）个类別的概率，同理q1，q2。选择日标变量v\*按变量vj划分后的 Gini 值最小的变量vj作为最优划分变量;

[3]第三步：对划分好的决策树重复进行第二步，直到无法继续划分，或满足收敛条件：

141第四步：输出最终的 CART 回归二叉树，作为最终的决策树，用于分类预测。

**7.Bagging算法流程**

<https://blog.csdn.net/qq_41870157/article/details/103395732>

输入为样本集D={(x,y1),(x2,y2),…(xm,ym)}，弱学习器算法, 弱分类器迭代次数T。

输出为最终的强分类器f(x)

(1)对于t=1,2…,T:

(a)对训练集进行第t次随机采样，共采集m次，得到包含m个样本的采样集Dt

(b)用采样集Dt训练第t个弱学习器Gt(x)

(2)如果是分类算法预测，则T个弱学习器投出最多票数的类别或者类别之一为最终类别。如果是回归算法，T个弱学习器得到的回归结果进行算术平均得到的值为最终的模型输出。

**9.随机森林算法流程**

遍历随机森林的大小 K 次：  
(1)从训练集 T 中有放回抽样的方式，取样N 次形成一个新子训练集 D  
(2)随机选择 m 个特征，其中 m < M  
(3)使用新的训练集 D 和 m 个特征，学习出一个完整的决策树  
得到随机森林