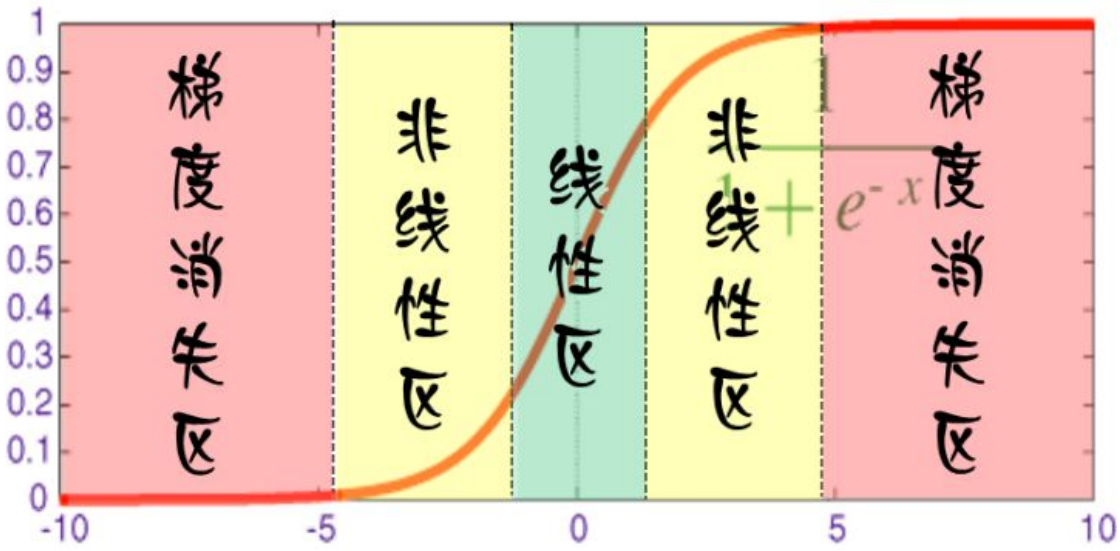
使用Typora添加数学公式https://blog.csdn.net/mingzhuo_126/article/details/82722455

机器学习

机器学习基础

- 介绍下欠拟合与过拟合：1. 过度拟合训练样本，模型在训练集上效果好；在测试集上效果差。模型泛化能力弱。2. 欠拟合正好相反，训练样本不能很好的拟合，同样泛化能力差
- 机器学习中用来防止过拟合的方法有哪些?

1. 更多的训练数据，要给足够多的有效数据，让少数噪音数据得干扰达到最低。获得方式包括：从数据源头获取更多数据；数据增强。
2. 合理处理模型，包括减少网络层数；限制训练时间；加入正则化；
3. 集成的方法: bagging，boosting，stacking，dropout，在训练时，**每次**随机（如50%概率）忽略隐层的某些节点；这样，我们相当于随机从2^H个模型中采样选择模型；同时，由于每个网络只见过一个训练数据（每次都是随机的新网络），类似 **bagging** 的做法。
4. 贝叶斯的方法

> 限制训练时间的原因：当网络权值较小时，神经元的激活函数工作在线性区，此时神经元的拟合能力较弱。一般都是初始为较小的权值。训练时间越长，部分网络权值可能越大，使得模型拟合能力越强。
>
> 
>
>
> 知乎问题解答：https://www.zhihu.com/question/59201590

- L1或L2是如何防止过拟合的? 1. L1权重向量在最优化的过程中变得稀疏（即非常接近0）2. L2正则化可以直观理解为它对于大数值的权重向量进行严厉惩罚，倾向于更加分散的权重向量。3. 用L1正则化和L2正则化降低模型的复杂度。符合奥卡姆剃刀 (Occam's razor)原理。奥卡姆剃刀原理应用于模型选择时变为以下想法：在所有可能选择的模型中，能够很好地解释已知数据并且十分简单才是最好的模型，也就是应该选择的模型。
- LR正则化与数据先验分布有什么关系？1. L2对应高斯分布，L1对应拉普拉斯分布 2.如果在用概率推导线性回归的时候，参数 w 引入协方差为 α 的零均值高斯先验。推导的结果就是岭回归，也就是损失函数加上L2正则 2. 如果在用概率推导线性回归的时候，参数 w 引入协方差为 α 的**拉普拉斯分布**先验。推导的结果就是LASSO ，也就是损失函数加上L1正则

> 参考：https://www.zhihu.com/question/23536142

- L1和L2正则的区别和应用场景：1. L1对异常点不太敏感，而L2则会对异常点存在放大效果。2. L1的变动很大，而L2的则整体变动不大。也就是L2更加稳定 3. L1 nrom几乎没有比L2 norm表现好的时候，优先使用L2 norm是比较好的选择，数据特征比较多时，不乏相关性特征，用L1可以起到筛选特征的作用，消除一波共线性问题，使得参数稀疏

  > 参考：https://blog.csdn.net/fantacy10000/article/details/90647686

- 介绍下偏差方差分解：1. 泛化误差(generalization error)可以分解为三个部分: 偏差(bias), 方差(variance) 和噪声(noise). 在估计学习算法性能的过程中, 我们主要关注偏差与方差. 因为噪声属于不可约减的误差 (irreducible error). 2. 偏差：度量了模型的期望预测和真实结果的偏离程度，刻画了**模型本身的拟合能力。** 3. 方差：度量了同样大小的训练集的变动所导致的学习性能的变化，即**刻画了数据扰动所造成的影响。** 4. 噪声：表达了当前任务上任何模型所能达到的期望泛化误差的下界，**刻画了学习问题本身的难度。** 5. 随着模型复杂度的提升, 偏差逐渐减小, 方差逐渐增大.模型的方差会偏高，模型倾向于过拟合，而模型的偏差会偏高，模型倾向于过拟合。

  > 参考：https://www.cnblogs.com/makefile/p/bias-var.html

- 介绍一般分类的指标：1. 准确度：分类正确的样本占全部样本的比例，精确度：某类的精确度就是：该类正确分类数占分类为该类的样本数，召回度：某类的召回率表示该类正确分类数占该类的总数 2. F1值：某类的F1值表示该类精确度和召回度的调和均值，ROC曲线：表示：横坐标是1-负样本召回率，纵坐标是正样本的召回率，曲线上的点表示的是在大于不同阈值就表示正类的样本数，AUC：ROC曲线下的面积。

F1值：

$$\frac{2}{F_1} = \frac{1}{\mathrm{Precision}} + \frac{1}{\mathrm{Recall}}$$

- 数据不平衡用accuracy、PR、ROC指标会咋样:1.
- 数据不平衡时的解决方式：1. 在评价标准上需要考虑，用ROC，不用准确率 2. 从数据上解决问题采样，主要包括随机欠采样，随机过采样，SMOTE过采样（具体见下方）EasyEnsemble算法（随机森林相似，它将多数类样本随机划分为N份，分别与少数类组成一个训练集训练N个模型，以N个模型的平均作为结果）和混合采样 3. 调整class_weight以及sample_weight的样本权重，一般设置为“balanced” 4. 设计损失函数，设计一个损失函数来惩罚少数类的错误分类，比如Focal Loss，在目标检测领域取得了良好的处理类别不平衡效果和改善误分类的效果

smote采样：

1. 对于少数类中每一个样本x，以欧氏距离为标准计算它到少数类样本集S_{min}中所有样本的距离，得到其k近邻。
2. 根据样本不平衡比例设置一个采样比例以确定采样倍率N，对于每一个少数类样本x，从其k近邻中随机选择若干个样本，假设选择的近邻为xn。
3. 对于每一个随机选出的近邻xn，分别与原样本按照如下的公式构建新的样本

$$x_{new} = x + rand(0,1) * |x - xn|$$

- 生成模型和判别模型的区别：1.生成模型(generative model)**通过学习先验分布来推导后验分布而进行分类**，而判定模型(discriminative model)**直接学习后验分布来进行分类。**2.对于分类模型，一般有三种建模方法：**生成模型，判定模型和直接学习决策边界。**举例来说，如果信用卡公司需要通过建模学习如何通过个人收入(X)来检测其是否会成为坏账(Y)。一种建模的方法是先学习整体的坏账率(P(Y))以及在坏账与非坏账的群体中，个人收入是如何分布的(P(X|Y))。在学习这两种概率后，通过Bayes theorem得到后验分布(P(Y|X))，即所有收入在某个区间的贷款人成为坏账的概率。这种方法被称为**生成模型**；通过对于先验分布的学习与建模来得到后验分布，从而对其进行分类。而另一种更加直观的建模方法是直接对后验分布进行学习，例如如果我们假设坏账率与个人收入是线性的关系，我们可以通过logistic regression来直接预测在该收入范围内的坏账率。这种方法则被称为**判定模型**；直接学习后验分布P(Y|X)而不在乎X与Y的联合分布甚至先验分布。当然，我们也可以**直接学习决策边界**，(decision boundary)，SVM便是直接学习边界的例子。

  > 参考：https://www.zhihu.com/question/20446337/answer/411353610

- 说下极大似然估计的原理：1. 是概率论在统计学中的应用。极大似然估计提供了一种给定观察数据来评估模型参数的方法，即：**“模型已定，参数未知”。**通过若干次试验，观察其结果，利用试验结果得到某个参数值能够使样本出现的概率为最大，则称为极大似然估计。 2.计算方式就是写出似然函数，对对似然函数取对数，并求导得到最佳参数。

- 损失函数有几种？原理是什么？有什么特点？1. 损失函数是**经验风险函数**的核心部分，也是**结构风险函数**重要组成部分。模型的结构风险函数包括了经验风险项和正则项 2. 0-1损失函数是指，预测值和目标值不相等为1，否则为0 3. 交叉熵损失函数 4. 平方损失函数,一般用于线性回归 5. 指数损失函数，AdaBoost就是一指数损失函数为损失函数的。 6. Hinge loss主要用于支持向量机中，
- 分类与回归的区别：1. 浅层： 两者的的预测目标变量类型不同，回归问题是连续变量，分类问题连续变量。 2. 中层：回归问题是定量问题，分类问题是定性问题。 3. 高层： 回归与分类的根本区别在于输出空间是否为一个度量空间。

  > 参考：https://www.cnblogs.com/gaowenxingxing/p/12360125.html

数据预处理

- 介绍下常见的数据预处理的方法：1. 数据清洗，包括缺失值的处理，异常值的处理，2. 数据规约，规范化，3. 特征生成和选择。

  > https://zhuanlan.zhihu.com/p/51131210

- 归一化，标准化：
- 特征选择的方法：卡方检验：：
- PCA和SVD算法：
- 时间序列数据怎么构建训练集和验证集，要注意什么 1.在划分训练集与测试集时不能够随机划分， 因为在构造样本时选取的时间步长一般很短，那么某些样本之间的差异会很小，如果随机划分会导致验证集的结果过于乐观。 2. 如果时间序列数据存在明显的趋势或者周期性的话，一般采用forward chain方法，3. 如果时间序列数据沿着时间轴没有明显的周期和趋势的话，即该时间序列是稳态的，那么其实是可以使用hv交叉验证方法的

  > https://blog.csdn.net/g1027785756/article/details/98481123

线性回归

- 线性回归为什么用均方误差：可以由概率推导出用均方误差的合理性，假设数据符合高斯分布，利用极大似然推导之后，得到的最终目标函数即是均方误差
- 梯度下降，随机，mini-batch
- 极大似然估计：
- **<手推最小二乘>**
- lasso和ridge的区别以及如何做特征选择：1.

逻辑回归，softmax回归

- 手推LR：https://zhuanlan.zhihu.com/p/38186102和https://blog.csdn.net/Candy_GL/article/details/87885465
- **代码实现LR的前向和反向传播**
- 写一下softmax的公式
- LR（逻辑回归） 为什么使用sigmoid函数：1. Sigmoid 函数自身的性质，也就是sigmoid 函数连续，单调递增，是非线性的，增强模型拟合能力 2. 是由概率推出的，逻辑回归认为函数其概率服从伯努利分布，将其写成指数族分布，可以推出sigmoid函数的形式。
- 交叉熵损失和均方误差对比：1. square mean在更新w，b时候，w,b的梯度跟激活函数的梯度成正比，Simoid函数在值非常高时候，梯度是很小的，比较平缓。cross entropy在更新w,b时候，预测值跟实际值差距越大，那么w,b调整就越快，收敛就越快。所以交叉熵的收敛速度应该更加快。2. 二次代价函数存在很多局部最小点，而交叉熵就不会。
- 逻辑回归特征之间关联程度大会有什么问题：1. 相关性高的特征太多放大了噪声的作用，多个特征实际上“平分”了这类特征对模型的贡献，这样就导致模型对于数据的变动更加敏感，泛化误差增大。 2. 对于变量的分析造成影响，衡量变量的重要性或贡献的时候存在困难，比如上面写的lgb的问题。

  > https://zhuanlan.zhihu.com/p/70124378?from_voters_page=true

- 交叉熵函数的形式：https://www.cnblogs.com/lijie-blog/p/10166002.html

决策树

- 介绍一下ID3和C4.5算法：1. ID3指的是利用信息增益来作为分裂节点的标准。ID3对某个类别较多的特征有偏向 2. c4.5指的利用信息增量率作为分裂节点的标准
- 介绍CART算法：1. 这是一类分类与回归树的总称，假设是一颗二叉树 2. 分类是利用gini系数找分裂特征和分裂值，3. 回归是用最小二乘来找到最优化节点和分割值
- 介绍下决策树的剪枝：1. 预剪枝 决策树生成过程中，如果这个节点进行划分，**不能带来泛化性能的提升**，则**停止划分**并将该节点设置为叶子节点。优点降低了过拟合，减少训练时间，缺点容易带来欠拟合2. 后剪枝：先训练好一棵树，然后自底向上对非叶子节点进行考察，如果将该节点对应的**子树替换为叶节点能不能带来泛化性能的提升**，能就将该子树替换为叶节点。优点不会欠拟合，缺点训练开销高于预剪枝决策树。
- 讲讲决策树是如何对对连续属性和对缺失属性的处理：1. 对于连续值，连续值a在D上出现了n个不同的取值，也就是连续值离散化处理。2. 对于缺失值，通过修改信息增益的公式，在原始计算前添加了占比系数作为权重。
- **<手推>** 信息增益的定义是什么，公式是什么；熵的定义什么，公式是什么

SVM，博客https://blog.csdn.net/macyang/article/details/38782399

- 简单介绍下SVM：1. SVM是一个二类分类器，主要分为线性最大硬间隔SVM, 软间隔SVM也就是，在硬间隔的基础上加一个松弛变量 2.支持向量表示与超平面最近的点，假设约束函数间隔是1，所有点都大于等于1，那么支持向量就是刚好满足到超平面等于1的点 3. SVM的有两种解释，一个是决策函数sign(wx+b) ,学习策略是软间隔最大化，算法是凸二次规划，一个是决策函数sign(wx+b)，学习策略是最优化折页损失函数，算法可以是梯度下降等
- 函数间隔与几何间隔：1. 几何间隔除以一个范数等于函数间隔2.不用函数间隔，因为几何间隔不随w，b等比例变化而变化。(w，b等比例变化,但超平面不变)
- svm的核函数有哪些，有什么区别和应用场景：1. 线性核，只能解决线性可分问题，但简单容易理解，一般是m(数据量)比较小而n(特征数量)比较大的时候 2. 多项式核依靠升维使得原本线性不可分的数据线性可分，优点是可解决非线性问题，但是比较多的参数要选择 3. 高斯核，优点是可以映射到无限维，只有一个参数，相比多项式核容易选择，缺点是可解释性差(无限多维的转换，无法算w)，计算速度比较慢和容易过拟合。**且需要进行特征缩放, 因为高斯核需要两个点之间的欧式距离, 如果不进行特征缩放的话数量级较大的特征对核函数的结果有决定性影响, 而数量级小的特征会被忽略.**4. 总结就是一般在样本数量可观、特征少选择高斯核，特征维数高选择线性核。
- 说说SVM相对于LR有什么不同? 1.样本点对于模型的作用不同. SVM中只有支持向量对模型有影响, 而LR中所有点都有影响. 2.损失函数不同. SVM采用的是合页损失函数, LR采用的是对率损失函数. 3.输出不同. LR输出的是对样本点是否属于给定分类的概率, SVM只能判断是否属于某一分类. 4.处理非线性问题的能力不同. SVM可以通过核函数灵活的将非线性问题转化为线性问题求解, 而LR一般不能非线性分类，需要手动进行特征转换。
- SVM的优点和缺点分别是什么：1.用内积核函数代替向高维空间的非线性映射. 2.间隔最大化的思想, 取得最优分类超平面. 3.最终结果只由支持向量决定, 剔除了大量的冗余样本, 因此具有一定的鲁棒性. 4.决策函数也只由支持向量决定, 降低了计算复杂度. 5.有坚实的理论基础. 6. 传统的SVM只能给出二分类的结果, 对于多分类需要需要用到组合分类器 7. 对数据缺失敏感, 对参数与核函数选择敏感. 8.
- SVM怎么进行多分类：1. 将某一类归为正类，其余全部是负类。该方法的最大缺陷是数据集的不平衡，因为某一类的实例往往只占一小部分。当然解决不平衡的问题可以进行降采样或者上采样，但是上采样中数据集过多重合，易导致过拟合，而降采样使得数据利用率不足，不能学习整个模型的特性 2. k类的数据集中，单独为每两类的样本设计SVM，进行分类。最终必须设计k(k-1)/2个分类器，最终用投票的方式进行选择。这也是libsvm采用的方法，但是当类别比较多的时候, 效率很低。
- svm模型C参数越大，代表模型的偏差大还是方差大？1.
- **<手推>**：https://zhuanlan.zhihu.com/p/45444502加数学优化（二）

EM

- 介绍下EM算法：1.极大似然估计，用现有对一个已知特定分布估计参数 2.EM也是已知分布，但数据非完全观测，一种特殊的极大似然。3. 具体来说，猜想隐含参数代表EM 算法的 E 步，具体来说就是计算联合分布的条件概率期望。然后基于观察数据和猜测的隐含参数一起来极大化对数似然，求解我们的模型参数（EM算法的M步。之后重复迭代E步和M步，直到收敛。 4. 一个最直观了解 EM 算法思路的是 K-Means 算法。在 K-Means 聚类时，每个聚类簇的质心是隐含数据。我们会假设 K 个初始化质心，即 EM 算法的 E 步；然后计算得到每个样本最近的质心，并把样本聚类到最近的这个质心，即 EM 算法的 M 步。重复这个 E 步和 M 步，直到质心不再变化为止，这样就完成了 K-Means 聚类。

> 参考https://zhuanlan.zhihu.com/p/36331115

Ensemble

boosting

- 介绍一下boosting：1.boosting指的是多个弱分类器（弱分类器一般选用单层决策树）进行合理的串行加权叠加，使其成为一个强分类器的一类算法。2. 常用的boosting包括Adaboost，GBDT的一类算法。
- 介绍下adaboost： 1. 一般使用单层决策树作为其弱分类器，adaboost包括数据的权重，是弱分类器的权重。其中，数据的权重主要用于弱分类器寻找其分类误差最小的决策点，数据的权重在训练过程中不断变化，却决于之前分类器对该数据的分类精确度。找到之后用这个最小误差计算出该弱分类器的权重（发言权），分类器权重越大说明该弱分类器在最终决策时拥有更大的发言权。2. adaboost可以表示成模型是加法模型，学习算法为前向分布算法，损失函数是指数函数推导出来。
- 讲讲在应用adaboost要注意的点：1. 异常点非常敏感，异常点非常容易错分，会影响后续若干个弱分类器。
- SVM跟AdaBoost联系：SVM仅取决于支持向量，那么AdaBoost呢？AdaBoost更关注之前被分错的样本，所以它们的共同点在于都不是一视同仁的看待所有样本。

bagging

- 介绍下随机森林算法：1. 用随机的方式建立一个森林，森林里面有很多的决策树组成，随机森林的每一棵决策树之间是没有关联的 2. 具体来说，对于每个决策树，随机森林都要对对输入的数据要进行行、列的采样，行采样就是对样本数据进行又放回的采样（使得相同数据在多个分类器训练，保证分类器的效果，有保证分类器的多样性），列采样就是在N个特征中选取其中M个特征来建立决策树。3. 在多棵决策树建立并预测之后，通过投票法来决定最终的分类结果。
- 随机森林和adaboost的异同：1. 随机森林是bagging的代表算法，子分类器是并行的，最终结果用投票法决定；而adaboost是boosting的代表算法，子分类器是串行，最终结果是加权累加的结果 2. 随机森林对异常值不敏感，而adaboost对异常值比较敏感。3. 随机森林是通过减少模型方差提高性能，adaboost是通过减少模型偏差提高性能。
- 说下随机森林有什么局限：1. 随机森林算法在训练和预测时都比较慢 2. 区分的类别十分多，随机森林的表现并不会很好 3. 其他bagging的一些缺陷。
- bagging为什么能减小方差？boosting为什么减小偏差：1. bagging是对许多强（甚至过强）的分类器求平均。在这里，每个单独的分类器的bias都是低的，平均之后bias依然低；而每个单独的分类器都强到可能产生overfitting的程度，也就是variance高，求平均的操作起到的作用就是降低这个variance。2. 是把许多弱的分类器组合成一个强的分类器。弱的分类器bias高，而强的分类器bias低，所以说boosting起到了降低bias的作用。variance不是boosting的主要考虑因素。Boosting 则是迭代算法，每一次迭代都根据上一次迭代的预测结果对样本进行加权，所以随着迭代不断进行，误差会越来越小，所以模型的 bias 会不断降低。
- 除了树模型，Bagging能不能接其他的基模型:1. 可以，还可以用k近邻分类器作为基分类器，这个在Sklearn那边做了封装，

blending&stacking

- 介绍下blending和stacking： 1. blending指的是原始的训练集先分成两部分，比如70%的数据作为新的训练集，剩下30%的数据作为测试集。第一层我们在这70%的数据上训练多个模型，然后去预测那30%数据的label。在第二层里，我们就直接用这30%数据在第一层预测的结果做为新特征

继续训练即可。2. stacking直接用所有的训练数据对第一层多个模型进行k折交叉验证，这样每个模型在训练集上都有一个预测值，然后将这些预测值做为新特征对第二层的模型进行训练。相比blending，stacking两层模型都使用了全部的训练数据。3. 两者的主要区别在于，blending相比stacking简单，同时不会数据泄露，但stacking使用多次的CV会比较稳健，blending由于第二层数据较少，容易过拟合。

GBDT

博客：https://zhuanlan.zhihu.com/p/29765582和https://www.zybuluo.com/yxd/note/611571

- 简单介绍GBDT： 1. 一种基于树的boosting集成算法 2.基础树一般是cart回归树 3. 加法模型， 当损失函数是均方误差的时候，每一步新增一棵树，拟合上一步的残差。4. 其他损失函数的时候一般就是拟合梯度。
- GBDT特征分裂规则：1.选取误差下降最多的
- GBDT和RF的区别： 1. GBDT是加法穿行，RF是并行，也就是一个是boosting一个是bagging 2. GBDT一般用cart回归树，RF都有 3. RF不用对特征进行预处理归一化，而GBDT需要（一般树形结构都不需要，最优化求解的都需要）
- GBDT可以并行计算的部分有哪些：1. 计算每个样本的负梯度
- GBDT和随机森林有什么区别：1. 随机森林是bagging的代表算法，子分类器是并行的，最终结果用投票法决定；而GBDT是boosting的代表算法，子分类器是串行，最终结果是加权累加的结果 2. 组成随机森林的树可以是分类树，也可以是回归树；而GBDT只由回归树组成 。 3. 随机森林是通过减少模型方差提高性能，GBDT是通过减少模型偏差提高性能。4. 随机森林对异常值不敏感，而GBDT对异常值比较敏感

XGBOOT

1. 博客https://blog.csdn.net/v_JULY_v/article/details/81410574

- 介绍下XGboost：1. 本质上是GDBT框架下的一类变体，基于boosting增强策略的加法模型，训练的时候采用前向分布算法进行贪婪的学习，每次迭代都学习一棵CART树来拟合之前 t-1 棵树的预测结果与训练样本真实值的残差。 2. GBDT进行了一系列优化，比如损失函数进行了二阶泰勒展开、目标函数加入正则项、支持并行和默认缺失值处理等，在可扩展性和训练速度上有了巨大的提升，但其核心思想没有大的变化。
- 说下XGboost是怎么分裂结点：1. 利用贪心策略来分裂。（1. 对所有特征都进行预排序特征值，用目标函数计算Gain收益来计算每个特征最佳分裂点 2. 选取最大增益Gain的特征和其收益点分裂 3. 总之，gain计算要计算特征数*对每个特征含有的划分方法）2. 使用近似算法来计算
- XGboost与GBDT的区别：1. 对于目标函数，xgboost利用泰勒展开近似的用到函数2阶导数和1阶导数，GBDT只用到1阶导数；另外，xgboost目标函数用到表示树复杂程度的正则化项，防止过拟合。2. xgboost可以在特征粒度上并行，保存预排序后的特征，然后保存为block(块)结构，后面的迭代中重复地使用这个结构。并行计算每个特征的带来的gain收益。3. 加速的其他两个方法：Shrinkage（缩减）也就是在每一步生成boosting树时添加一个删减参数：η，通SGD中的学习率类似，这个参数可以减少单颗树的作用，列抽样 （column subsampling），这点和随机森林类似，也就是对特征进行随机采样后建立树。 4. XGBoost的基分类器不仅支持CART决策树，还支持线性分类器，此时XGBoost相当于带L1和L2正则化项的Logistic回归（分类问题）或者线性回归（回归问题）# https://www.zhihu.com/question/41354392

- XGBoost的参数调优有哪些经验：1. 选择较高的**学习速率(learning rate)**。一般情况下，学习速率的值为0.1。但是，对于不同的问题，理想的学习速率有时候会在0.05到0.3之间波动。选择**对应于此学习速率的理想决策树数量**。XGBoost有一个很有用的函数“cv”，这个函数可以在每一次迭代中使用交叉验证，并返回理想的决策树数量。2. 给定的学习速率和决策树数量，进行**决策树特定参数调优**(max_depth, min_child_weight, gamma, subsample, colsample_bytree)。在确定一棵树的过程中，我们可以选择不同的参数 3. **正则化参数**的调优。(lambda, alpha)。这些参数可以降低模型的复杂度，从而提高模型的表现。# 参考文章https://www.cnblogs.com/mfryf/p/6293814.html
- 讲讲xgboost的特征重要性是怎么计算的：特征的重要性在xgboost主要由三部分组成。1. gain增益：意味着相应的特征对通过对模型中的每个树采取每个特征的贡献而计算出的模型的相对贡献。与其他特征相比，此度量值的较高值意味着它对于生成预测更为重要。2. 覆盖度量：指的是与此功能相关的观测的相对数量。例如，如果您有100个观察值，4个特征和3棵树，并且假设特征1分别用于决定树1，树2和树3中10个，5个和2个观察值的叶节点;那么该度量将计算此功能的覆盖范围为10 + 5 + 2 = 17个观测值。 3. 频率：表示特定特征在模型树中发生的相对次数的百分比，在上面的例子中，如果feature1发生在2个分裂中，1个分裂和3个分裂在每个树1，树2和树3中;那么特征1的权重将是2 + 1 + 3 = 6。特征1的频率被计算为其在所有特征的权重上的百分比权重 4. gain增益是解释每个特征的相对重要性的最相关属性。在xgb实现中，一般就是以gain增量来作为重要性考虑。

LightGBM

- 介绍下lightGBM：1. 也是GBDT框架下的一类变体，追求高效，便捷 2. 主要在树的生成做了改进，相对于xgboost，LightGBM引入直方图分布算法，
- 介绍下直方图分布算法：1. 基本思想是先把连续的浮点特征值离散化成k个整数（其实又是分桶的思想，而这些桶称为bin，比如[0,0.1)→0, [0.1,0.3)→1），同时构造一个宽度为k的直方图。对于分类特征来说，则是每一种取值放入一个bin，且当取值的个数大于max bin数时，会忽略那些很少出现的category值。2. 当遍历一次数据后，直方图累积了需要的统计量，然后根据直方图的离散值，遍历寻找最优的分割点。也就是相比于xgboost，不需要按照预排序算法那样，对于每个特征都计算#data遍了，而是只需要计算#bins遍

  > 参考：https://blog.csdn.net/anshuai_aw1/article/details/83040541

- 介绍下相对于xgboost，LightGBM所作的改进有哪些：1. 带深度限制的Leaf-wise的叶子生长策略，也就是从xgboost的Level-wise相比，在分裂次数相同的情况下，Leaf-wise可以降低更多的误差，得到更好的精度。Leaf-wise的缺点是可能会长出比较深的决策树，产生过拟合。因此LightGBM在Leaf-wise之上增加了一个最大深度的限制，在保证高效率的同时防止过拟合。2. Histogram（直方图）做差加速，一个叶子的直方图可以由它的父亲节点的直方图与它兄弟的直方图做差得到。通常构造直方图，需要遍历该叶子上的所有数据，但直方图做差仅需遍历直方图的k个桶。利用这个方法，LightGBM可以在构造一个叶子的直方图后，可以用非常微小的代价得到它兄弟叶子的直方图，在速度上可以提升一倍。3. LightGBM可以直接支持类别特征,之前的模型一般需要一般需要把类别特征，转化到多维的0/1特征，降低了空间和时间的效率。

  > https://blog.csdn.net/weixin_41510260/article/details/95378749

HMM

- 介绍下隐马尔可夫模型：1. 是一种标注模型，隐马尔可夫模型是关于时序（顺序）的概率模型，描述由一个隐藏的马尔可夫链随机生成不可观测的状态随机序列，再由各个状态生成一个观测而产

生观测随机序列的过程。隐藏的马尔可夫链随机生成的状态的序列，称为状态序列（state sequence); 每个状态生成一个观测，而由此产生的观测的随机序列，称为观测序列（observation sequence)。序列的每一个位置又可以看作是一个时刻。用前向算法求解2. 隐马尔可夫有三个基本问题，包括概率计算问题：给定模型参数和观测序列，计算出现某个观测序列的概率，预测问题：给定模型参数和观测序列，求最有可能的状态序列，这个用近似或者维特比算法，学习问题：给定观测序列和状态序列，得到模型参数 用前向-后向算法求解。

CRF

- 介绍下CRF：一种标注模型，条件随机场是一种判别式无向图模型，即条件随机场是对条件概率分布建模（隐马尔可夫和马尔可夫随机场都是对联合概率分布建模，是生成模型）。条件随机场对输入的观测序列和标记序列建立条件概率模型。
- CRF和HMM的区别和联系：1.HMM是生成模型，CRF是判别模型 2.HMM是概率有向图，CRF是概率无向图 3.HMM求解过程可能是局部最优，CRF可以全局最优 4.CRF概率归一化较合理，HMM则会导致label bias 问题

> 参考https://zhuanlan.zhihu.com/p/31187060

聚类

- 说说你了解哪些聚类算法：1. K-Means聚类，随机初始化一个各自的聚类质心点（cluster centroids），计算每个数据点到质心的距离来进行分类，聚类里每个点到它的欧式距离平方和最小这个条件更新质点，不断迭代 2.均值漂移算法，工作原理基于质心，这意味着它的目标是定位每个簇/类的质心，即先算出当前点的偏移均值，将该点移动到此偏移均值，然后以此为新的起始点，继续移动，直到满足最终的条件（找出最密集的区域）3. 基于密度的聚类算法 4. EM聚类 5. 层次聚类

> 参考https://zhuanlan.zhihu.com/p/37381630

- **<手推K-Means>**
- KNN和K-means的区别和联系是什么：1. KNN是监督学习中的一种算法，K-Means是非监督学习中一种聚类算法，2. K的含义：来了一个样本x，要给它分类，即求出它的y，就从数据集中，在x附近找离它最近的K个数据点，这K个数据点，类别c占的个数最多，就把x的label设为c； K的含义：K是人工固定好的数字，假设数据集合可以分为K个簇，由于是依靠人工定好，需要一点先验知识 3. 相似点：都包含这样的过程，给定一个点，在数据集中找离它最近的点。即二者都用到了NN(Nears Neighbor)算法，一般用KD树来实现NN。