```
终于把XGBoost总结写出来了!
Original 王茂霖 Datawhale 5/2
                             ↑↑↑关注后"<mark>星标</mark>"Datawhale
                            每日干货 & 每月组队学习,不错过
                                  Datawhale干货
                         作者:王茂霖,华中科技大学,Datawhale成员
内容概括
XGBoost模型及调参总结
 1. XGBoost原理
 2. XGBoost优势总结
 3. XGBoost参数详解
 4. XGBoost快速使用
 5. XGBoost调参方法
PPT下载:后台回复"210502"可获取
 XGBoost模型介绍
                                                    TIACHI天地 🍆 Datawhale
                          XGBoost模型介绍
                                 一、XGBoost原理
                                                    TIACHI天池 🍑 Datawhale
   XGBoost
                      XGBoost: A Scalable Tree Boosting System
                           Tianqi Chen
                                                 Carlos Guestrin
                       University of Washington 
tqchen@cs.washington.edu
                                             University of Washington 
guestrin@cs.washington.edu
       XGBoost的目标函数如下:
                                     正则项包括了叶子节点数目T和leaf score的L2模的平方
           Training loss
         其中正则项控制着模型的复杂度。
                                 XGBoost在优化目标函数的同时相当于做了预剪枝
XGBoost是2016年由华盛顿大学陈天奇老师带领开发的一个可扩展机器学习系统。严格意义上讲
XGBoost并不是一种模型,而是一个可供用户轻松解决分类、回归或排序问题的软件包。它内部
实现了梯度提升树(GBDT)模型,并对模型中的算法进行了诸多优化,在取得高精度的同时又保持
了极快的速度。
                              二、XGBoost优势总结
                                                    TIACHI天地 Contambale
   XGBoost
  XGBoost的优势:
                                                    相对于GBDT模型, XGBoost里面
                                                    采用了如下的优化:
   •正则化
                                                    • 对目标函数进行了泰勒展示的
       • 可有效防止模型过拟合
                                                      二阶展开, 可以更加高效拟合
   •并行处理
                                                      误差。
       • XGBoost并行处理数据的速度要比普通提升树
                                                    • 提出了一种估计分裂点的算法
         算法更快.
                                                      加速CART树的构建过程, 同时
   •自动处理缺失数据
                                                      可以处理稀疏数据。
       • 不做切分点, 但会把缺失值放进左右子树看效
                                                    • 提出了一种树的并行策略加速
       果
                                                      迭代。
  •剪枝策略
                                                    • 为模型的分布式算法进行了底
       • 普通的提升采用的是贪心算法,只有在不再有
                                                      层优化。
         增益时才会停止分裂
  •可以在目前树模型上继续迭代
      • XGBoost可以在训练中使用不同的迭代策略
XGBoost模型调参
                              一、XGBoost参数详解
1.一般参数
                                                    TIACHI天地 🎾 Datawhale
   XGBoost
         一般参数 [General Parameters] https://xgboost.readthedocs.io/en/latest/parameter.html

    booster [default: gbtree]

            * 表示应用的调学习器的类型,推荐用默认参数
            * 可选的有gbtree, dart, gblinear
              gblinear是线性模型,表现很差,接近一个LASSO
              dart是柯模型的一种,思想是每次训练新柯的时候,随机从前m轮的柯中扔掉一些,来避免过报合
              gbtree即是论文中主要讨论的柯模型(推荐)
          verbosity [default: 1]
           * 训练过程中打印的日志等级, 0 (silent), 1 (warning), 2 (info), 3 (debug)
          3. nthread [default: 最大可用线程数][alias: n_jobs]
           * 训练过程中的并行线程数
            * 如果用的是sklearn的api, 那么使用n_jobs来代替
        每个分类器算法参数 [Tree Booster Parameters]

    eta [default: 0.3] [alias: learning_rate] [range: [0, 1]]

           * 就是常说的学习速率,控制每一次学习的权重缩减,恰后来的模型提供更多的学习空间
         gamma [default: 0] [alias: min_split_loss] [range: [0, inf]]
           * 叶子节点分裂时所需要的最小的损失减少量,这个值越大,叶子节点越难分裂,所以算法就越保守
         	ext{Gain} = rac{G_L^2}{H_L + \lambda} + rac{G_R^2}{H_R + \lambda} - rac{\left(G_L + G_R
ight)^2}{H_L + H_R + \lambda} - \gamma
        3. max_depth [default: 6] [range: [0, inf]]
         * 柯的最大深度
          * 这个值对结果的影响算是比较大的了,值越大,柯的深度越深,模型的复杂度就越高,就越容易过拟合
          * 注意如果这个值被设置的较大,会吃掉大量的内存
          * 一般来说比价合适的取值区间为[3, 10]

    min_child_weight [default: 1] [range: [0, inf]]

          * 最小的叶子节点权重
          * 在普通的GBM中,叶子节点样本没有权重的概念,其实就是等权重的,也就相当于叶子节点样本个数
          * 越小越没有限制,容易过报合,太高容易欠报合
        5. max_delta_step [default: 0] [range: [0, inf]]
          * 适用于正负样本不均衡情况下,控制学习速率(类似eta)最大为某个值,不能超过这个阈值
          * 首先我们有参数eta来控制学习速率,为了后面学习到更多,每一步在权重上乘上这个因子,降低速度
          * 但是在正负样本不均衡的情况下eta不足够,因为此时由于二阶导接近于0的原因,权重会特别大
          * 这个参数就是用来控制学习速率最大不能超过这个数值
        6. sub_sample [default: 1] [range: (0, 1]]
          * 样本抽样比例
          * 在每次训练的随机选取sub_sample比例的样本来作为训练样本
        colsample_bytree [default: 1]
          * 这里实际上有3个参数,借助了随机森林的特征抽样的思想,3个参数可以同时使用
          * colsample_bytree 更常用, 每棵树的特征抽样比例
          * colsample_bylevel 每一层深度的柯特征抽样比例
          * colsample_bynode 每一个节点的特征抽样比例
       lambda [default: 1] [alias: reg_lambda]
         * 损失函数中的L2正则化项的系数,类似RidgeRegression,减轻过报合
       9. alpha [default: 0] [alias: reg_alpha]
         * 损失函数中的L1正则化项的系数,类似LASSO,减轻过拟合

    scale_pos_weight [default: 1]

          * 在正负样本不均衡的情况下,此参数需要设置,通常为: sum(负样本) / sum(正样本)
2.学习目标参数
            学习目标参数 [Learning Task Parameters]
             1. objective [default: reg:squarederror(均方误意)]
               * 目标函数的选择。默认为均方误差损失,当然还有很多其他的。这里列举几个主要的
               * reg:squarederror 均方误差
               * reg:squaredlogerror 均方对数误差
               * reg:logistic 对数几率损失。参考对数几率回归(逻辑回归)
               * binary:logistic 二分类对数几率回归。输出概率值
              * binary:hinge
                           二分类合页损失,此时不输出概率值,而是0或1
               * multi:softmax
                           多分类softmax损失。此时需要设置num_class参数
             2. eval_metric [default: 根据objective而定]
              * 模型性能度量方法,主要根据objective而定,也可以自定义一些,下面列举一些常见的
               * rmse : root mean square error 也就是平方误差和开银号
              * aucpr: area under the pr curve pr曲线下面积
3.工具包参数
      工具包参数 [XGB Packages Parameters]

    num_boost_round

          * 迭代次数, 这货其实跟sklearn中的n_estimators是一样的
          * sklearn的api中用n_estimators,原始xgb中用num_boost_round
       evals
          * 训练过程中通过计算验证集的指标,观察模型性能的数据集
          * 指标就是通过eval_metric参数来制定的
       early_stopping_rounds
          * 在num_boost_round的轮调练中,如果过程中指标经过early_stopping_rounds轮还没有减少
          * 指标是通过evals的验证集, 计算eval_metric的指标
                              二、XGBoost快速使用
 XGBoost提供了方便的Sklearn接口供我们使用:
   # 分割数据集
   from sklearn.model_selection import train_test_split
   x_train,x_valid,y_train,y_vaild = train_test_split(data, target,
                                 test_size = 0.2, random_state = 0)
   # 建立XGBoost回归模型并预测
   from xgboost import XGBRegressor
   model = XGBRegressor(
        learning_rate = 0.1,
        n_{estimators} = 300,
        max_depth = 7,
        min_child_weight = 3,
        subsample = 0.8,
        colsample_bytree = 0.8,
        seed = 0
   model.fit( X_train, y_train, eval_metric='mae',
        eval_set=[(X_train, y_train), (X_valid, y_valid)],
        early_stopping_rounds=20 )
   y_pred = model.predict( X_valid )
 原生XGBoost
   import xgboost as xgb
   params = {
          'eval_metric': 'mae',
          'eta':
                                        0.1,
          'max_depth':
          'min_child_weight': 3,
          'subsample':
                                        0.8,
          'colsample_bytree': 0.8,
          'seed':
   n_rounds = 300
   dtrain = xgb.DMatrix( X_train, y_train )
   dvalid = xgb.DMatrix( X_valid, y_valid )
   model = xgb.train( params, dtrain, n_rounds,
         evals=[(dtrain, 'train'), (dvalid, 'valid')],
         early_stopping_rounds=20 )
   y_pred = model.predict( dvalid )
                        三、XGBoost调参方法(贝叶斯优化)
Hyperopt是一个sklearn的python库,在搜索空间上进行串行和并行优化,搜索空间可以是实
值,离散和条件维度。
1.初始化空间所需的值范围
 步骤1: 初始化空间或所需的值范围:
  import hyperopt
  space={'max_depth': hp.quniform("max_depth", 3, 18, 1),
        'gamma': hp.uniform ('gamma', 1,9),
        'reg_alpha' : hp.quniform('reg_alpha', 40,180,1),
        'reg_lambda' : hp.uniform('reg_lambda', 0,1),
        'colsample_bytree' : hp.uniform('colsample_bytree', 0.5,1),
        'min_child_weight' : hp.quniform('min_child_weight', 0, 10, 1),
        'n_estimators': 180
2.定义目标函数
 步骤2: 定义目标函数:
  # Regression:
  def hyperparameter_tuning(space):
     model=xgb.XGBRegressor(n_estimators =space['n_estimators'], max_depth = int(space['max_depth']), gamma =
  space['gamma'],
                    reg_alpha = int(space['reg_alpha'],min_child_weight=space['min_child_weight'],
                    colsample_bytree=space['colsample_bytree']))
     evaluation = [( x_train, y_train), ( x_test, y_test)]
     model.fit(x_train, y_train,
           eval_set=evaluation, eval_metric="rmse",
           early_stopping_rounds=10,verbose=False)
     pred = model.predict(x_test)
     mse= mean_squared_error(y_test, pred)
     print ("SCORE:", mse)
     #change the metric if you like
     return {'loss':mse, 'status': STATUS_OK, 'model': model}
3.运行hyperopt功能
 步骤3: 运行Hyperopt功能:
```

Scan to Follow

```
trials = Trials()
best = fmin(fn=hyperparameter_tuning,
            space=space,
            algo=tpe.suggest,
            max_evals=100,
            trials=trials)
print (best)
```

王茂霖,Datawhale重要贡献成员,Datawhale&天池数据挖掘学习赛开源内容贡献者,阅读超 10w。

本文作者

参赛30余次,获得DCIC-数字中国创新创业大赛亚军,全球城市计算AI挑战赛,Alibaba Cloud German Al Challenge等多项Top10。



喜欢此内容的人还喜欢 刷爆全网的动态条形图,原来5行Python代码就能实现! Python大数据分析 论文推荐|[CVPR 2021] 基于傅里叶轮廓嵌入的任意形状文本检测(有 源码) CSIG文档图像分析与识别专委会