Lesson 7 CTR预估算法与基于流行度的推荐

FM与MF;GBDT+LR模型; Wide&Deep模型,NFM 都经常被用来预测点击率

FM使用更加频繁:因为它的泛化能力更强;

效果提升与Cost的trade-off:根据具体工作场景而定

MF只考虑了UserID,ItemID的特征

FM可以考虑多个特征

逻辑回归:

Predict-prob可以转化为 predict(0,1): 大于0.5为1,小于0.5为0

但是当样本数量大时,LR的缺点在于时间成本很高,这时我们用决策树来解决.

GBDT + LR:

Stacking 思想的二分类:

GBDT: Input Featrures, Tree Splits; Transformed Features

LR: linear Classifer (L1来防止过拟合)

Transformed Features:GBDT的输出,LR的输入

GBDT:回归树既可以做回归也能做分类(加sigmoid); Gradient boosting decision tree

GBDT:基于残差来学习,多棵树进行学习,每棵树学习上一棵树留下的残差,直到收敛

Adaboost:样本分布的原理,每棵树将上一课树学习的错误部分权重提升进行学习,所以可能两棵树之间的样本交叉部分很多

GBDT的树是CART树,但树的deepth更浅,并且有多棵树,不会太容易过拟合

NE: Normalised Cross-Entropy交叉信息熵 其实就是logloss/background CTR的熵, 所以越小越好

GBDT+LR 大约提升了3%

GBDT其实是在做特征构造,一棵树其实就是做一个one-hot,最后由LR来做分类任务.

评价指标:

NE. NE对background CTR不敏感,所有要看logloss

Calibration:预估CTR/实际CTR 越接近1,效果越好

AUC:正样本排在前面的大小和负样本排在前面的大小; 所以越大越好

调包: from sklearn.ensemble import GradientBoostingRegressor, GradientBoostingClassifier

Xgboost: max\_depth 一般设成7-9 一般来说设置得要比GBDT要多

n\_estimator: 不一定越多越好,因为太大容易过拟合

xgboost 实际是在GBDT的基础上做了正则化向(防止过拟合),所以迭代次数可以设的比GBDT要大

LR实际是一个分类器,理论上可以用SVM替代;同理lightbgm,xgboost可以替代GBDT做特征提取

集成学习方法(RF与GBDT的区别):

Boosting:通过将弱分类器提升为强分类器的集成方法来提升预测精度(adaboost,gbdt)

这个也可以叫做串行.即在之前学习的内容基础上继续来学习进行提升.即第一个学习后,第二个会在第一个学习的基础上继续学习.

Bagging: 第一个学习的内容和第二个学习的内容没有任何关系,所以叫做并行

通过自助采样的方法生成众多并行式的分类器，通过“少数服从多数”的原则来确定最终的结果（比如Random Forest）

这里的自助采样指一定数量的树,每棵树采用不同的样本,在训练后放入bag中,再随机取出来.

对于分类任务来说,是采用’’少数服从多数’’来决定输出结果;

对于回归任务来说,是取所有结果的平均值.

Wide&Deep算法

LR+DNN

Wide推荐:可解释性强,但是需要人工做特征工程

整个架构和DeepFM是一样的

Deep的架构:先输入continuous features和 categorical features,然后对categorical features做完embedding, 然后再和continuous features一起做集中embedding,然后扔进三层relu的框架进行学习

最后就是LR的结果和Deep的结果做了ensemble,但是这个效果比两个单独的模型solo好,从AUC看出来的.

Wide&Deep是一种并行算法,即LR和DNN各自运行后,然后对两个结果放在一起后做了一个sigmoid.

FNN

CTR模型可以基本都用于二分类问题

在CTR中特征大多都是离散,高维并且稀疏的,所以要去做embedding(one-hot维度太大,降维).

FNN实际是Wide&Deep中的Deep,模型,也就是DNN,但在Deep模型之前做了FM,目的是填充空缺矩阵,使稀疏矩阵变得稠密

NFM: FM+DNN

NFM其实算是一个串行.机制是这样的:将Dense Embedding分别喂给LR和Bi-Interaction Pooling(这个其实就是FM),因为FM处理的其实是二阶特征,这个可以算是高阶特征中的低阶特征,把这个结果再扔进DNN中做模拟,最后把LR的结果和DNN的结果做融合.

NFM解决了FM对非线性和高阶特征交叉能力不足的问题

实际上,FM可以看做是NFM中DNN的hidden layer为0的一种特殊情况.

FM做排序比较多,也可以做召回;

基于流行度的推荐下节课再讲.