

第十二节课

<https://github.com/cystanford> 老师的GitHub

第十节课笔记		
讲义内容概括	知识点	Question?

预测全家桶

Project A：员工离职预测

Project B：银行产品推荐

分类算法：LR , SVM, KNN

矩阵分解：FunkSVD,

BiasSVD, SVD++

FM模型：FM, FFM, DeepFM,

NFM, AFM

树模型：GBDT, XGBoost,

LightGBM, CatBoost, NGBoost

Attention模型：DIN, DIEN, DSIN

机器学习四大神器

什么是集成学习

GBDT原理

XGBoost

LightGBM

CatBoost

NGBoost

针对Project A如何使用四大神器

Summary

每种模型都有适用的场景

LR优点:

- 实现简单, 广泛地应用于工业问题上;
- 分类时计算量非常小, 速度很快, 使用资源低;
- 方便处理样本概率分布;

LR缺点:

- 当特征空间很大时, LR的性能不是很好;
- 容易欠拟合, 准确度不太高;
- 不能很好地处理大量多类特征或变量;
- 通常只处理二分类问题, 多分类需要采用softmax (LR在多分类的推广), 且必须线性可分;
- 对于非线性特征, 需要进行转换;

SVM优点:

- 可以解决高维问题, 即大型特征空间;
- 能够处理非线性特征的相互作用;
- 需要先将数据进行归一化, 因为计算是基于距离的模型, 所以SVM和LR都需要对数据进行归一化处理

SVM缺点:

- 当样本很多时, 效率并不是很高;
- 对非线性问题没有通用解决方案, 可能会尝试找到合适核函数
- 对缺失数据敏感;

SVM核的选择是有技巧的, 样本数量<特征数, 线性核, 大于特征数使用非线性核

Summary

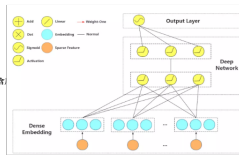
CTR模型及演化

- FM: 考虑了特征交叉, 即放数据非常稀疏的情况下, 依然能估计出可靠的参数并进行预测
- FFM: 在FM基础上引入了field (组), 使得每两组特征交叉的向量都是基独立的, 预测效果更好
- embedding + MLP: 深度学习CTR预估的通用框架
- Wide & Deep, LFM-DNN: 记忆能力+泛化能力
- DeepFM: 并行结构, FFM和DNN分开计算
- NFM: 串行架构, 将FM的结果作为DNN的输入
- AFM: 在FM的基础上增加Attention机制, 对量化后NFM进行加权求和
- DIN: Attention机制, 对用户历史行为序列进行挖掘



Summary

- embedding + MLP, 深度学习CTR预估的通用框架
- Step1, 对不同维度的one-hot特征进行嵌入(embedding), 使其降维成低维稠密特征
- Step2, 将这些特征向量拼接(concatenate)成一个向量
- Step3, 再不断叠象全连接层, 也就是多层感知机(Multilayer Perceptron, MLP, 也称为前馈神经网络)
- Step4, 最终输出预测的点比率
- FNN, 使用FMEmbedding层进行初始化

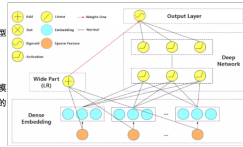


FNN使用FM进行参数初始化
FNN实际上是wide & deep模型中的deep模型

XGBoost LightGBM CatBoost NGBoost

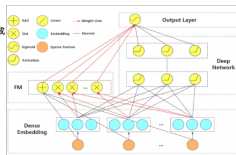
Summary

- Wide & Deep模型
- Wide模型, 采用Linear Regression, 解决模型的记忆能力(特征组合需要人来设计)
- Deep模型, 即FNN, 解决模型的泛化能力
- Joint Training, 同时训练Wide模型和Deep模型, 并将两个模型的结果的加权作为最终的预测结果



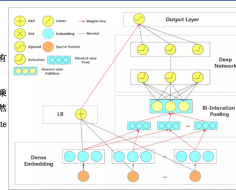
Summary

- DeepFM模型
- DeepFM是将Wide & Deep模型中的Wide替换成了FM模型
- $$y_{FM} = (w, x) + \sum_{i=1}^K \left(\sum_{j=1}^K \langle v_i, v_j \rangle x_{ij} \right)$$
$$a^{(l+1)} = \sigma(W^{(l)}a^{(l)} + b^{(l)})$$
$$y = \text{sigmoid}(y_{FM} + y_{DNN})$$



Summary

- NFM模型
- FNN, Wide & Deep, DeepFM都是在DNN部分, 对embedding之后的特征进行concatenate, 没有充分进行特征交叉计算。
- NFM算法是对embedding直接采用对位相乘(element-wise)后相加起来作为交叉特征。然后通过DNN直接将特征压缩, 最后concatenate linear部分和deep部分的特征。
- 两种FM和DNN的结合方式:
- DeepFM, 并行结构, FM和DNN分开计算
- NFM, 串行架构, 将FM的结果作为DNN的输入



Summary

- FM和SVM的区别:
- SVM的二次特征交叉参数是独立的, 而FM的二次特征交叉参数是两+维的向量w, v, 交叉参数就不是独立地, 而是相互影响的
- 在数据稀疏的情况下, 线性SVM比多项式SVM结果比较差, 线性SVM只有一维特征, 不能挖掘深层次组合特征, 而多项式SVM, 交叉的多个特征需要在训练集上共现才能被学习到, 否则对应的参数就为0
- 而FM不一样, 通过向量化的交叉, 可以学习到不同特征之间的交互, 进行挖掘到更深层次的抽象意义

- FM和LR的区别
- LR是从组合特征的角度去描述特征之间的交互组合, FM实际上是从模型(latent factor model)的角度来做的, 即FM中特征的交互是模型参数的一部分。
- FM是过拟合的思想, 基于latent factor, 来降低交叉项参数学习不充分的影响
- FM能很大程度上避免了数据稀疏行造成参数估计不准确的影响

Summary

- 每种模型都有适用的场景
- 可以使用LR模型作为预测的Baseline
- FM衍生模型在推荐系统, 尤其是CTR预估中有广泛应用, 弥补了LR模型的不足(需要人工组合特征, 耗费大量时间和人力)
- Attention机制, 对于Diversity多样性的情况, Attention机制可以提升效率, 并且得出更好的结果
- Tree Ensemble模型, 比如GBDT, 使用广泛, 因为训练模型更可控
- 对于LR模型, 如果大组合, 需要增加feature, 才能提高准确率, 而对于tree-ensemble来说, 解决方法是训练更多的决策树tree, Kaggle比赛中使用较多

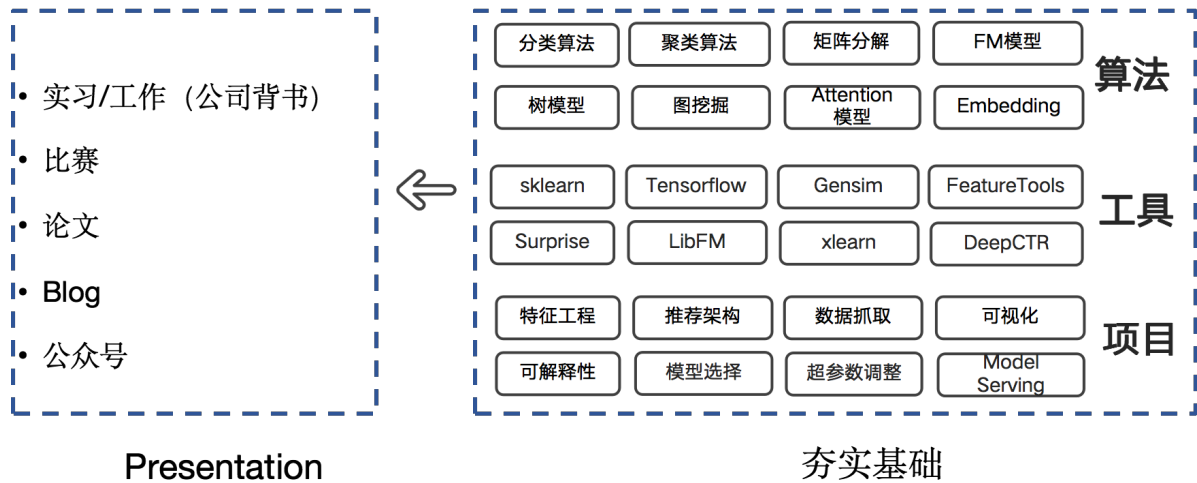
- 常用预测模型:
- 分类算法: LR, SVM, KNN
- 矩阵分解: FunkSVD, BiasSVD, SVD++
- FM模型: FM, FFM, DeepFM, NFM, AFM
- 树模型: GBDT, XGBoost, LightGBM, CatBoost, NGBoost
- Attention模型: DIN, DFN, DSIN

总结Summary

- 记录作业内容

- Thinking1: 电商定向广告和搜索广告有怎样的区别，算法模型是否有差别
-

TIPS: 我们要PK的是和我们一起竞争的人



•