深度学习在 Airbnb 搜索中的应用

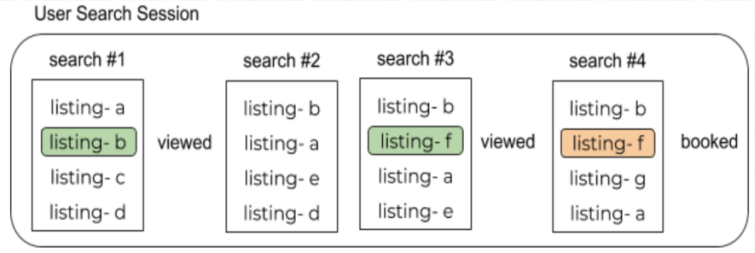
Liam Huang*

2018 年 11 月 27 日

*liamhuang0205@gmail.com

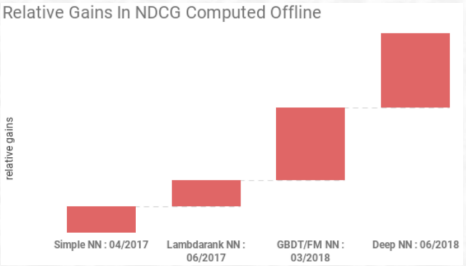
业务背景

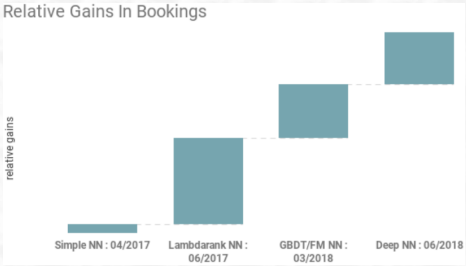
- Airbnb 中文名「爱彼迎」;
- 主要业务：收集民宿资源，向承租方（游客）提供租赁;
- 日常场景:
  - 用户输入 Query，或者直接选定地图上的位置;
  - Airbnb 给出符合要求的民宿列表;
  - 用户寻找满意民宿;
  - 下单预定.



不是传统的搜索，与推荐业务有一定相似性.

模型演变





- 迁向深度学习模型的过程并非一蹴而就;
- 横轴：历次迭代上线的 DL 模型;
- 纵轴：NDCG gain.

Simple NN

> 别去逞英雄！
>
> ——Andrej Karpathy (TESLA AI 总监)

- 设计超复杂的网络 $\rightarrow$ 花费大量时间精力 $\rightarrow$ 效果不好，不了了之

超简单的模型结构：

- 一个隐藏层，全连接；
- ReLU 激活函数；
- 输入特征与 GBDT 一致；
- 训练目标与 GBDT 一致：最小化 MSE.
  - 产生预定的 listing 记为正例 1；
  - 没有预定的 listing 记为负例 0.

结果与结论：

- 相对 GBDT，rank performance 没有太大提升；
- 验证了 NN 在线上的可行性.

LambdaRank NN

实际情况:

- 不逞英雄 → NN 上线 → 效果一般

修改名人名言:

- 「不要逞英雄」→ 「不要一开始就逞英雄」

修改模型结构:

- GBDT + LambdaRank → LambdaMART[1]
- Simple NN + LambdaRank → LambdaRank NN
- 构造偏序 listing-pair;
- 以 Sigmoid 函数构造偏序概率 → 最小化交叉熵损失
- 以 ΔNDCG 为 pair 的权重

结果与结论:

- 线下小幅提升 NDCG;
- 线上大幅提升.

[1] https://liam.page/2016/07/10/a-not-so-simple-introduction-to-lambdamart/

代码片段 1: LambdaRank NN 部分实现（TensorFlow™）

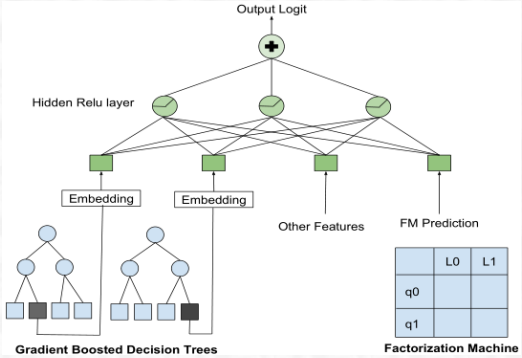
```
def apply_discount(x):
  return np.log(2.0) / np.log(2.0 + x)

def compute_weights(logit_op, session):
  logit_vals = session.run(logit_op)
  ranks = NUM_SAMPLES - 1 - logit_vals.argsort(axis = 1).argsort
      (axis = 1)
  discounted_non_booking = apply_discount(ranks[:, 1:])
  discounted_booking = apply_discount(np.expand_dims(ranks[:,
      0], axis = 1))
  discounted_weights = np.abs(discounted_booking -
      discounted_non_booking)
  return discounted_weight

# Compute the pairwise loss
pairwise_loss = tf.nn.sigmoid_cross_entropy_with_logits(targets
    = tf.ones_like(logit_op[:, 0]), logits = logit_op[:, 0] -
    logit_op[:, i:] )

# Compute the lambdarank weights based on delta ndcg
weights = compute_weights(logit_op, session)
# Multiply pairwise loss by lambdarank weights
loss = tf.reduce_mean(tf.multiply(pairwise_loss, weights))
```

Decesion Tree/Factorization Machine NN

在 Simple NN 的基础上进行 model stacking. 灵感:

- Simple NN 的效果和 GBDT/FM 基本一致;
- Simple NN 的排序结果和 GBDT/FM 差别很大;
- 让它们在一起.

- 原始特征;
- GBDT 输出结点的索引作为高维特征;
- FM 输出的预测值作为特征;
- 单隐层全连接 ReLU.

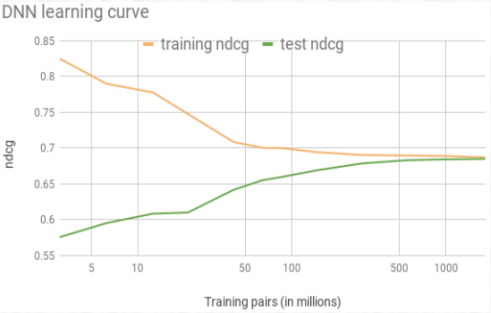


Deep NN

- 先前的模型复杂度：超复杂；
- 深度学习的真理之一：暴力堆数据.
  - 训练用数据量：$10\times$.

模型结构：

- 双层隐藏层；
- 输入 195 个（几乎是）基本的原始特征；
  - 当前 listing 与用户过去曾见过的 listing 的相似性；
  - ……
- 第一隐藏层：127 个 ReLU 全连接结点；
- 第二隐藏层：83 个 ReLU 全连接结点.



结果与结论：

- 离线、在线均有较大提升；
- 训练数据量达到 17 亿时，gap of train and test 消失.

失败的尝试

Embedding listing-ID

关于 item-ID 的成功尝试:

- NLP 中的 Word Embedding;
- recsys 中的 Video Embedding 和 User Embedding.

然而[2]，在 Airbnb 的场景中，listing-ID 带来的大都是过拟合.

[2] 「然而」、「但是」前面的都是废话.

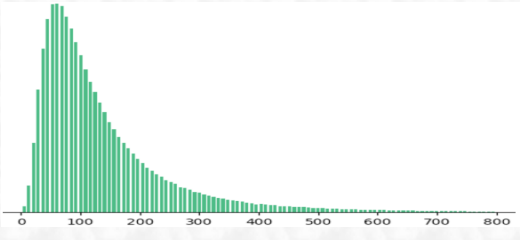
——匿名

原因分析:

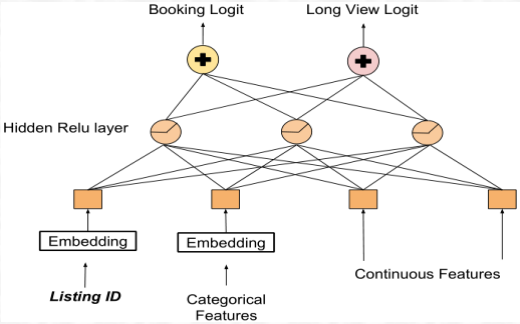
- Embedding 需要每个 item 都有大量训练数据才能合理收敛;
- item 可以无约束地大量重复 $\rightarrow$ success
  - 自然语言中的词;
  - 推荐系统里的内容、用户.
- item 无法无约束地大量重复 $\rightarrow$ failure
  - 最火爆的房子，一年也就只能被预定 365 次;
  - 其它房子量更小.
- Airbnb 场景的独特性导致的问题.

Multi-task learning

观察用户浏览行为：



- listing 的预定无法大量重复；
- 但用户对 listing 的浏览行为可以大量重复；
- 用户长时间的浏览行为和预定行为强相关.



- 双路输出；
- 以大量 user view 数据，修正 listing-ID 的过拟合；
- 线上实验 long view 大幅提升，但预定量无显著提升.

原因分析:

- 人工 case 分析
  - 高端但价格高的 listing;
  - 描述文字很长的 listing;
  - 非常特别，甚至滑稽的 listing.
- 再次：Airbnb 场景的独特性导致的问题.

特征工程

普遍认知：

- 传统机器学习：依赖大量的特征工程；
- 深度神经网络：让隐藏层的神经元去做特征工程.

然而，数据满足一定条件能让 NN 表现得更好.

特征归一化

- GBDT 对特征归一化不敏感 ← 按相对有序取分裂点
- NN 对特征的绝对数值敏感
  - 较大的数值变化，在 BP 中引起较大的梯度变化；
  - 改变 ReLU 的输入，导致 ReLU 激活函数永久关闭.

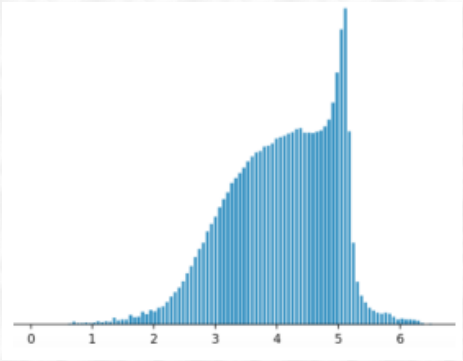
解决办法：将特征值映射到 $[-1, 1]$ 区间，且中值为 0：

- 满足正态分布的特征：$\frac{\text{feature\_value}-\mu}{\sigma}$；
- 符合幂律分布的特征：$\ln\left[\frac{1+\text{feature\_value}}{1+\text{median}}\right]$

特征分布

尽可能使特征分布平滑.（Airbnb 认为的）理由和好处:

- 便于排查错误 ← 异常的数据分布会呈现一个毛刺
- 便于泛化 ← NN 每一层都在使输出的分布更加平滑
- 便于检查特征完整性 ← 设计不合理的特征，其分布可能严重有偏



可预订的天数的分布

特征重要度分析

Airbnb 尝试了多种特征重要度分析手段，但发现都有各种问题：

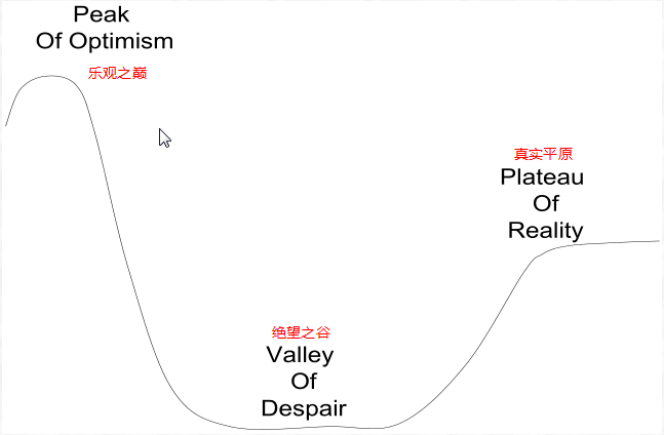
- Score Decomposition
- Ablation Test
- Permutation Test

最终，Airbnb 自研了一套名为 TopBot Analysis 的分析系统．逻辑：

- 保留一套测试集；
- 对每个 query 分别 rank；
- 取出所有 query 各自 rank 的 top 和 bottom 结果；
- 按不同特征，绘制 top/bottom 结果的分布；
- top/bottom 分布差异越大，说明模型对该特征越敏感．


Price distribution of top ranked listings
Price distribution of bottom ranked listings
Review count distribution of top ranked listings
Review count distribution of bottom ranked listings
Number of listings
Increasing price
Increasing review count

回顾



- 乐观之巅 ← 「GBDT 换成 NN，随随便便拿收益」
- 绝望之谷 ← 模型越来越复杂，收益总是拿不到
- 真实平原 ← 老老实实做特征、调超参

UGA

The opal codon.