

任务五：实战—阿里新手天池赛

目录

- 赛题背景：2
- 赛题任务：2
- 字段表：2
- 数据探索：3
- 数据预处理：4
- 数据集划分：（见 data_split.py 文件）4
- 特征工程：（见 feature_extract.py 文件）5
- 模型设计：（见 xgb.py 文件）7
- 天池信息.....8

生活大实惠：O2O 优惠券使用预测

赛题背景：

随着移动设备的完善和普及，移动互联网+各行各业进入了高速发展阶段，这其中以 O2O (Online to Offline) 消费最为吸引眼球。据不完全统计，O2O 行业估值上亿的创业公司至少有 10 家，也不乏百亿巨头的身影。O2O 行业天然关联数亿消费者，各类 APP 每天记录了超过百亿条用户行为和位置记录，因而成为大数据科研和商业化运营的最佳结合点之一。以优惠券盘活老用户或吸引新客户进店消费是 O2O 的一种重要营销方式。然而随机投放的优惠券对多数用户造成无意义的干扰。对商家而言，滥发的优惠券可能降低品牌声誉，同时难以估算营销成本。个性化投放是提高优惠券核销率的重要技术，它可以让具有一定偏好的消费者得到真正的实惠，同时赋予商家更强的营销能力。本次大赛为参赛选手提供了 O2O 场景相关的丰富数据，希望参赛选手通过分析建模，精准预测用户是否会在规定时间内使用相应优惠券。

赛题任务：

通过分析线上和线下的数据分析，预测用户在 2016 年 7 月领取优惠券后 15 天以内的使用情况。

提供数据：ccf_offline_stagel_test_revised.csv,
ccf_offline_stagel_train.csv, ccf_online_stagel_train.csv,
sample_submission.csv

字段表：

Table 1: 用户线下消费和优惠券领取行为(ccf_offline_stagel_train.csv)

Field	Description
User_id	用户 ID
Merchant_id	商户 ID
Coupon_id	优惠券 ID: null 表示无优惠券消费，此时 Discount_rate 和 Date_received 字段无意义
Discount_rate	优惠率: $x \in [0,1]$ 代表折扣率; $x:y$ 表示满 x 减 y 。单位是元
Distance	user 经常活动的地点离该 merchant 的最近门店距离是 $x*500$ 米 (如果是连锁店，则取最近的一家门店), $x \in [0,10]$; null 表示无此信息，0 表示低于 500 米，10 表示大于 5 公里;
Date_received	领取优惠券日期
Date	消费日期: 如果 Date=null & Coupon_id != null, 该记录表示领取优惠券但没有使用, 即负样本; 如果 Date!=null & Coupon_id = null, 则表示普通消费日期; 如果 Date!=null & Coupon_id != null, 则表示用优惠券消费日期, 即正样本;

Table 2: 用户线上点击/消费和优惠券领取行为(ccf_online_stagel_train.csv)

Field	Description
User_id	用户 ID

Merchant_id	商户 ID
Action	0 点击, 1 购买, 2 领取优惠券
Coupon_id	优惠券 ID: null 表示无优惠券消费, 此时 Discount_rate 和 Date_received 字段无意义。“fixed”表示该交易是限时低价活动。
Discount_rate	优惠率: $x \in [0,1]$ 代表折扣率; $x:y$ 表示满 x 减 y ; “fixed”表示低价限时优惠;
Date_received	领取优惠券日期
Date	消费日期: 如果 Date=null & Coupon_id != null, 该记录表示领取优惠券但没有使用; 如果 Date!=null & Coupon_id = null, 则表示普通消费日期; 如果 Date!=null & Coupon_id != null, 则表示用优惠券消费日期;

Table 3: 用户 020 线下优惠券使用预测样本
(ccf_offline_stagel_test_revised.csv)

Field	Description
User_id	用户 ID
Merchant_id	商户 ID
Coupon_id	优惠券 ID: null 表示无优惠券消费, 此时 Discount_rate 和 Date_received 字段无意义
Discount_rate	优惠率: $x \in [0,1]$ 代表折扣率; $x:y$ 表示满 x 减 y 。单位是元
Distance	user 经常活动的地点离该 merchant 的最近门店距离是 $x*500$ 米 (如果是连锁店, 则取最近的一家门店), $x \in [0,10]$; null 表示无此信息, 0 表示低于 500 米, 10 表示大于 5 公里;
Date_received	领取优惠券日期

Table 4: 选手提交文件字段, 其中 user_id, coupon_id 和 date_received 均来自 Table 3, 而 Probability 为预测值

Field	Description
User_id	用户 ID
Merchant_id	商户 ID
Coupon_id	优惠券 ID: null 表示无优惠券消费, 此时 Discount_rate 和 Date_received 字段无意义
Date_received	领取优惠券日期
Probability	15 天内用券概率, 由参赛选手给出

数据探索:

首先对所给的数据进行预处理分析, 对给予的几个基本数据集进行分析。

赛题主要给了 offtrain, ontrain 和 offtest 三个数据集。通过前两个数据集分析, 获得训练模型, 然后对 offtest 数据集中的记录进行预测。然而通过实际分析, 发现 offtest 中的关键属性 user_id 和 merchant_id 几乎都在 offtrain 中出现, 另一方面 ontrain 数据集中特有的 action 属性并存在于 offtest 数据集中, 并且缺乏了 distance 这个关键属性。理论上也确实最好使用线下的训练集来训练线下消费模型, 对线下消费情况进行分析, 在此题中

我并没有利用到线上数据集，猜测该数据集应该是在那个比赛的第二赛季起作用。

```
In[4]: import pandas as pd
In[5]: off_train = pd.read_csv(r'E:\PycharmProjects\O2O_work\ccf_offline_stage1_train.csv',
    ....: keep_default_na=False)
    ....: off_train.columns = ['user_id', 'merchant_id', 'coupon_id', 'discount_rate', 'distance', 'date_received', 'date']
    ....: off_test = pd.read_csv(r'E:\PycharmProjects\O2O_work\ccf_offline_stage1_test_revised.csv',
    ....: keep_default_na=False)
    ....: off_test.columns = ['user_id', 'merchant_id', 'coupon_id', 'discount_rate', 'distance', 'date_received']
In[6]: set(off_test.user_id) - set(off_train.user_id)
Out[6]: {1286474L, 2495873L}
In[7]: set(off_test.merchant_id) - set(off_train.merchant_id)
Out[7]: {5920L}
```

可见 offtest 中绝大部分用户和商户都出现在 offtrain 中，只有一个新商家和两个新用户，故只用 offtrain 做数据集。

数据预处理：

对于 offtrain 和 offtest 中都存在不少缺失值，且都以 ‘null’ 字符串存在。在 read_csv 读取时数据集通过设置 keep_default_na=False 使得缺失以原有的 ‘null’ 字符串保留，而不是设置为 np.nan。并没有使用其他方式在一开始进行缺失值填充也是后面使用 xgboost 模型时能自动处理缺失值。这样做导致了一个不好的后果就是在特征处理是先是使用 groupby 在通过 merge 左连接不可避免的产生了写 nan 类型空值，与原有的 ‘null’ 冲突，虽然可以通过 df.replace() 来替换，但是效率还是比较底下，这个问题希望在接下来的工作中解决统一。

数据集划分：（见 data_split.py 文件）

```
In[13]: df = off_train[['date', 'date_received']][off_train.date_received > '20160615']
In[14]: df.date_received.value_counts()
Out[14]:
null    701602
Name: date_received, dtype: int64
```

观察到 offtrain 数据集在 6.15 之后并没有记录任何领券行为。因此在 offtrain 的特征提取区间取 6.15 以前。最后决定使用划窗法采用两个训练集 dataset1 和 dataset2，测试集 dataset3。如下：

	预测区间（提取 Label）	特征区间（提取 Feature）
训练集 1	20160414~20160514	20160101~20160413
训练集 2	20160515~20160615	20160201~20160514
测试集	20160701~20160731	20160315~20160630

特征工程：（见 feature_extract.py 文件）

数据集拥有：'user_id', 'merchant_id', 'coupon_id',
'discount_rate', 'distance', 'date_received', 'date'

七大基本特征。通过这七个基本特征进行交叉组合又可得到新的特征群。如：user_merchant 组合特征可以表示用户对于商家的偏好，进一步细分可得用户在该商家消费次数，使用优惠券等消费次数等等特征。最终决定采用五个特征来划分特征。分别为用户特征，优惠券特征，商家特征，用户—商家特征，其他特征。实际上基本特征之间组合远不止五种，但由于个别特征群特征数量太少，便划分到其他特征上，比如我就把用户—距离特征归属到用户特征群上。总的特征列表如下：

用户特征

用户特征	feature about user
用户去过的商家数	user_count_merchants
用户使用优惠券的最小距离	user_min_distance
用户使用优惠券的最大距离	user_max_distance
用户使用优惠券最多的距离	user_most_distance
用户使用优惠券的平均距离	user_avg_distance
用户总的消费次数	buy_total
用户使用优惠券的次数	buy_use_coupon
用户领取到的优惠券的数量	user_coupon_received
用户使用优惠券的时间间隔	user_time_gap
用户使用优惠券的平均时间间隔	user_avg_time_gap
用户使用优惠券的最长时间间隔	user_max_time_gap
用户使用优惠券的最短时间间隔	user_min_time_gap
用户使用优惠券的最多的时间间隔	user_most_time_gap
用户核销优惠券的最高折扣率	user_max_discount_rate
用户核销优惠券的最低折扣率	user_min_discount_rate
用户核销优惠券的平均折扣率	user_avg_discount_rate
用户使用优惠券购买的比例	user_buy_use_coupon_rate
用户领取到的优惠券核销率	user_coupon_trans_rate
用户平均在每个商家的核销优惠券次数	user_avg_cons_of_merchant
用户平均在每个商家的核销率	user_avg_coupon_of_merchant

商家特征

商家特征	feature about merchant
商家总的交易次数	total_sales
商家的优惠券发行数量	total_coupon
商家发行的优惠券种类数	total_coupon_types
商家优惠券交易的数量	coupon_sales

商家优惠券交易占比率	merchant_coupon_sales_rate
商家优惠券被使用率	merchant_coupon_used_rate
商家优惠券交易的平均距离	merchant_avg_distance
商家优惠券交易的最大距离	merchant_max_distance
商家优惠券交易的最小距离	merchant_min_distance
商家优惠券交易的众数距离	merchant_most_distance
商家发行优惠券的最高折扣率	merchant_max_discount_rate
商家发行优惠券的最低折扣率	merchant_min_discount_rate
商家发行优惠券的平均折扣率	merchant_avg_discount_rate
商家发行优惠券的众数折扣率	merchant_most_discount_rate

优惠券特征

优惠券特征	feature about coupon
优惠券类型（是否为满减类）	is_man_jian
相同折扣率的优惠券发行数量	discount_rate_total_nums
计算相同折扣率的优惠券核销数量	discount_rate_used_nums
满减的满额	discount_man
满减的减额	discount_jian
真实折扣率	discount_rate
优惠券领取时的工作日	day_of_week
该优惠券领取时的月份	day_of_month
相同 id 的优惠券的数量	coupon_count

用户-商家特征

用户-商家特征	feature between user and merchant
用户在该商家的总消费次数	user_merchant_buy_total
用户在该商家收到的优惠券总数	user_merchant_received
用户在该商家购物使用的优惠券数量	user_merchant_buy_use_coupon
用户在该商家购物不使用的优惠券数量	user_merchant_buy_no_coupon
用户商家总记录条数	user_merchant_records
商家发行优惠券核销率	user_merchant_coupon_trans_rate
用户在该商家使用优惠券消费比例	user_merchant_use_coupon_rate
用户在该商家不使用优惠券消费比例	user_merchant_no_coupon_rate

其他特征

其他特征	other feature
当天收到的优惠券数量	nowday_receive_coupon
收到第一张同样优惠券的时间	user_first_same_coupon_date
收到最后一张同样优惠券的时间	user_last_same_coupon_date
优惠券(coupon_id)是第一张优惠券	is_first
优惠券(coupon_id)是最后张优惠券	is_last

对于优惠券特征 `day_of_week` 使用了 `get_dummies()` 方法进行独热编码 (One-Hot encoding)。分解为周一至周日七个属性。特征之间的距离计算变得更加合理。并在此基础上添加了 '`is_weekend`' 特征。

通过对数据集的特征提取，合并

模型设计：（见 `xgb.py` 文件）

网上观看论坛以及和同学讨论，得知在单模型下 `xgboost` 的效果最好，于是偷懒只使用了 `xgboost` 来融合而模型。事实上 `xgboost` 对本赛题真的非常契合，效果拔群。其优势有以下几点：`xgboost` 内置处理缺失值规则，解决了本赛题最终数据集中大量缺失值问题。`xgboost` 会一直分裂到指定的最大深度 (`max_depth`)，然后回过头来剪枝，使得 `xgboost` 相对于其他采用贪心算法剪枝的模型效果要好很多。

`Xgboost` 参数设置，第一，采用 `gbtree` 的 `booster` 迭代模式。`Pairwise` 中 Rank 排序算法。根据赛题要求对于有效数据的度量方法 (`eval_metric`) 采用 `auc`，曲线下面积来度量。至于其他参数则在推荐范围区间内进行手动微调。参数 `params` 最终如下：

```
params={'booster': 'gbtree',
        'objective': 'rank:pairwise',
        'eval_metric': 'auc',
        'gamma': 0.1,
        'min_child_weight': 1,
        'max_depth': 5,
        'lambda': 10,
        'subsample': 0.7,
        'colsample_bytree': 0.8,
        'colsample_bylevel': 0.8,
        'eta': 0.01,
        'tree_method': 'exact',
        'seed': 0,
        'nthread': 12
    }
```

设置迭代的次数为 3500，最终模拟模型 `auc` 为 0.895 左右，线上成绩为 0.76564078。

```

7 dataset2 = pd.read_csv(r'E:\PycharmProjects\O2O_work\dataset2.csv')
8 dataset2.label.replace(-1, 0, inplace=True)
9 dataset3 = pd.read_csv(r'E:\PycharmProjects\O2O_work\dataset3.csv')
10
11 dataset1.drop_duplicates(inplace=True)
12 dataset2.drop_duplicates(inplace=True)
13 dataset3.drop_duplicates(inplace=True)
14
15 dataset12 = pd.concat([dataset1, dataset2], axis=0)
16
17 dataset1_y = dataset1.label
18 dataset1_x = dataset1.drop(['user_id', 'label'], axis=1) # 'day_gap_before', 'day_gap_after' cause overfitting, 0
19 dataset2_y = dataset2.label
20 dataset2_x = dataset2.drop(['user_id', 'label'], axis=1)
21 dataset12_y = dataset12.label
22 dataset12_x = dataset12.drop(['user_id', 'label'], axis=1)
23 dataset3_preds = dataset3[['user_id', 'coupon_id', 'date_received']]
24 dataset3_x = dataset3.drop(['user_id', 'coupon_id', 'date_received'], axis=1)

```

Run console output:

```

[3447] train-auc:0.894686
[13:13:33] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 58 extra nodes, 0 pruned nodes, max_depth=5
[3448] train-auc:0.894689
[13:13:33] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 46 extra nodes, 0 pruned nodes, max_depth=5
[3449] train-auc:0.894691
[13:13:34] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 60 extra nodes, 0 pruned nodes, max_depth=5
[3450] train-auc:0.894693
[13:13:34] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 34 extra nodes, 0 pruned nodes, max_depth=5
[3451] train-auc:0.894694
[13:13:34] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 62 extra nodes, 0 pruned nodes, max_depth=5

```



天池信息

最终成绩如下：天池 ID : cauchyguo

个人中心	比赛 2	数据集 0	帖子 0
 Hi, cauchyguo 积分/排名: 0/7715 我的主页 参加的活动 个人信息 认证 个人简历	新浪微博互动预测-挑战Baseline <hr/> 参加时间: 2018-05-30 参加形式: cauchyguo 主办方: 微博 <div>🕒 进行中</div>	天池新人实战赛o2o优惠券使用预测 <hr/> 参加时间: 2018-05-29 参加形式: cauchyguo 第1赛季排名: 217/9334 主办方: 阿里云 <div>🕒 进行中</div>	

历史提交记录:

			0217 / 0.76564	
0				
时间		auc	当天排名	
2018-07-27 21:05:12		0.76564078 ↑	16	
2018-07-27 09:44:26		0.76483738 ↑	24	
2018-07-26 17:06:50		0.61334438 ↓	29	
2018-07-25 17:44:06		0.61864690 ↑	27	
2018-07-25 00:39:27		0.60391883 ↓	18	
			<< 1 2 > >> <input type="text"/> GO	

时间		auc	当天排名	
2018-07-24 21:14:09		0.60799194 ↑	23	
2018-07-24 01:05:59		0.59782489	17	