# 数据挖掘比赛案例介绍

## 甜橙金融杯大数据竞赛

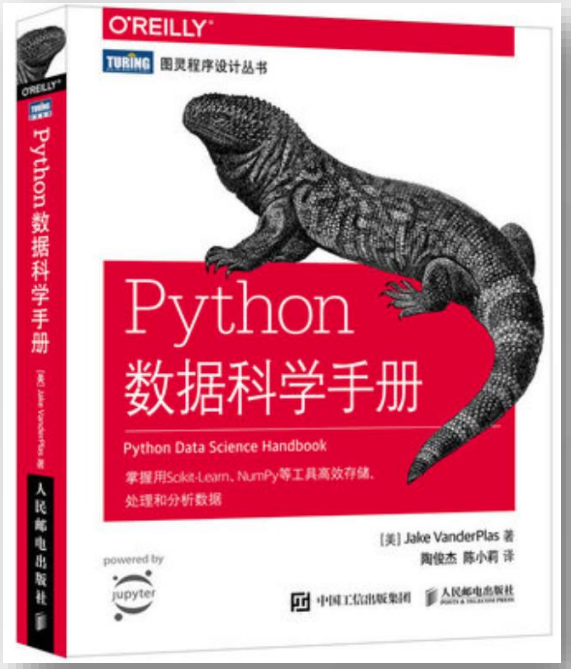
# 数据分析常用工具

1. Python、R、…

2.Anaconda、pycharm

3. numpy、pandas

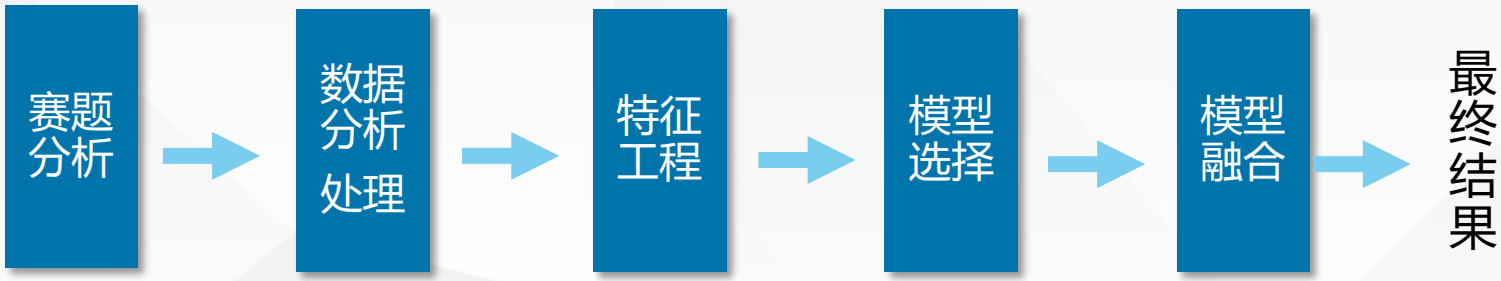
4. 可视化matplotlib、seaborm




或

网上搜索Python官方文档

# 基本流程



CONTANTS

# 怎么分析赛题

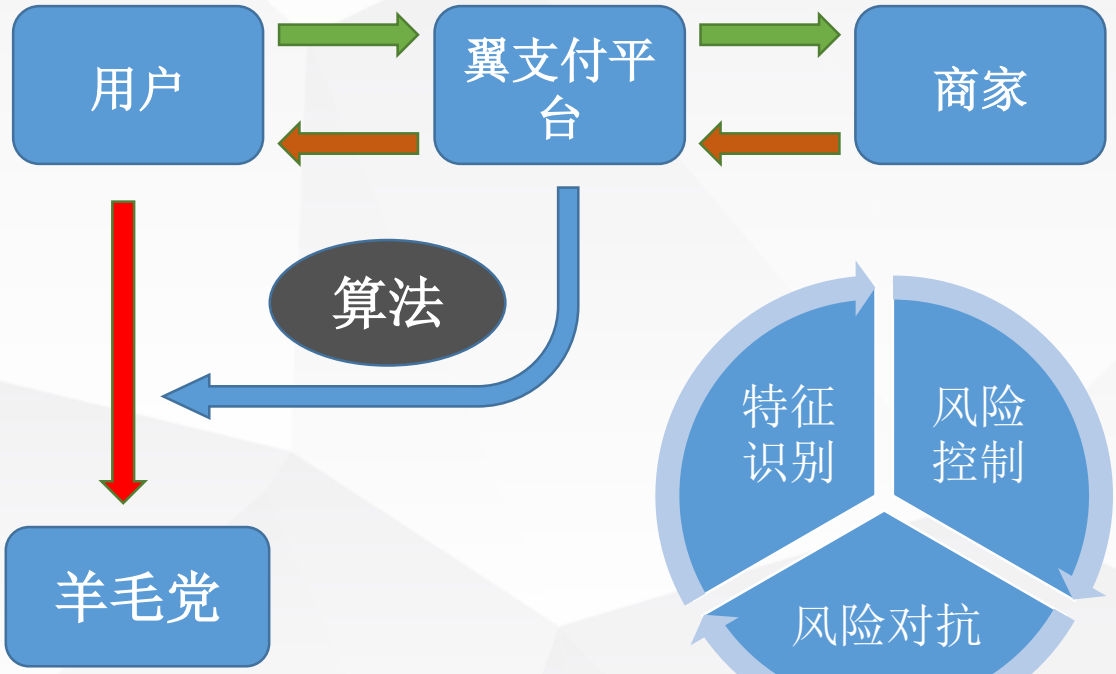
1. 背景？

2. 数据？

3.任务和评价指标？

# 赛题背景

支付平台应用机器学习算法识别羊毛党用户。



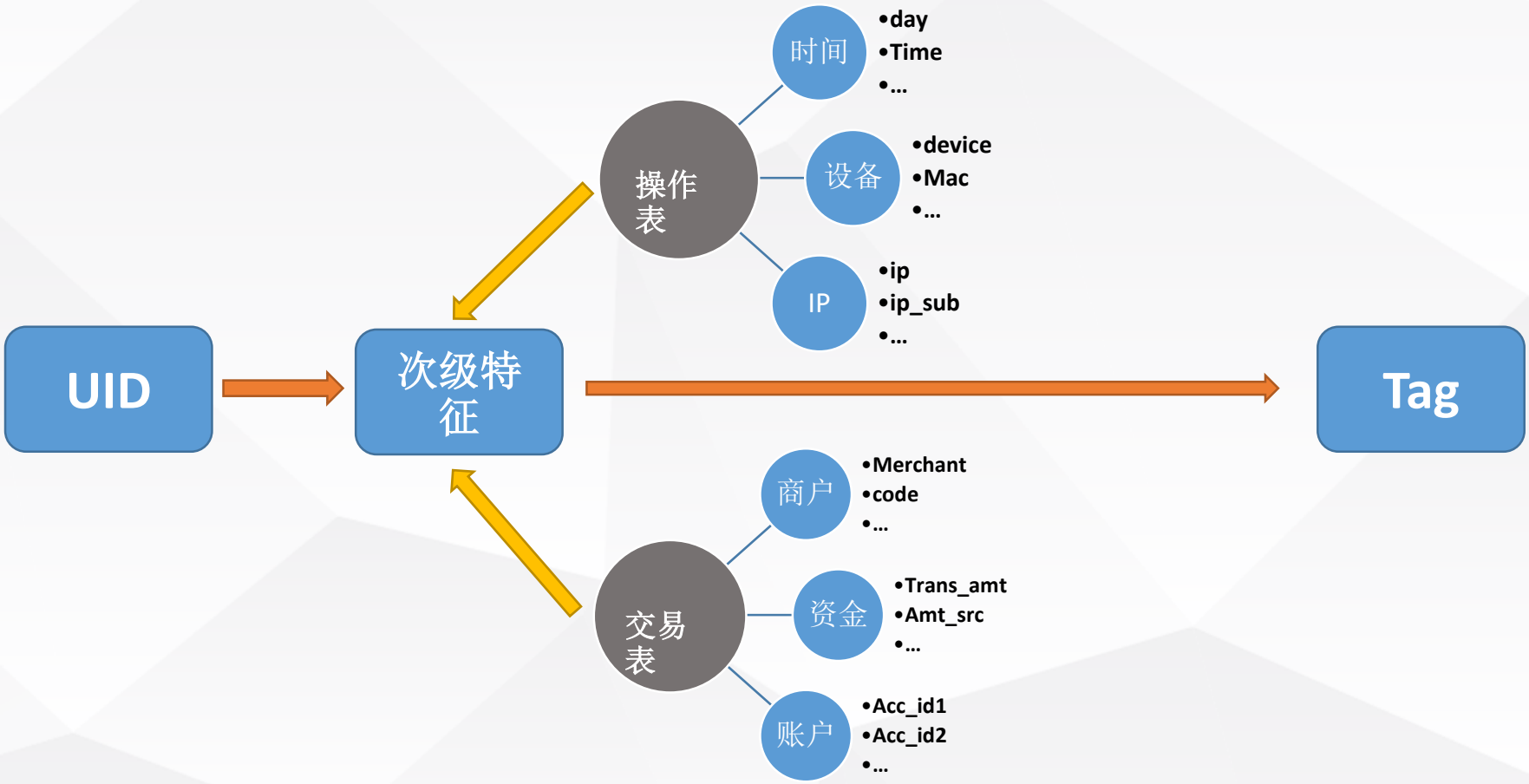
## 操作详单数据字典

| 字段名 | 中文解释 | 字段说明 |
| --- | --- | --- |
| UID | 用户编号 | |
| day | 操作日期 | 连续的日期标识，<br>E.g., 1为第一天，2为第二天，以此类推 |
| mode | 操作类型 | 操作类型（例如：修改密码、查询余额…） |
| success | 操作状态 | |
| time | 操作时间点 | |
| os | 操作系统 | |
| version | 客户端版本号 | |
| device1 | 操作设备参数1 | 设备名称加密，原字段如“Jack’s iphone” |
| device2 | 操作设备参数2 | 设备型号 |
| device_code1 | 操作设备唯一标识1 | 设备号唯一标识加密，可用于安卓类设备的唯一标识 |
| device_code2 | 操作设备唯一标识2 | 设备号唯一标识加密，可用于安卓类设备的唯一标识 |
| mac1 | MAC地址 | 操作设备MAC地址编码加密，<br>原字段如“38:XX:XX:XX:XX:92” |
| ip1 | IP地址 | 操作设备IP地址编码加密 |
| ip2 | IP地址 | 操作电脑IP地址编码加密 |
| device_code3 | 操作设备唯一标识3 | 设备号唯一标识加密，可用于苹果类设备的唯一标识 |
| mac2 | MAC地址 | WIFI MAC地址编码加密，<br>原字段如“02:XX:XX:XX:XX:03 |
| wifi | WIFI名称 | WIFI名称，原字段如“A的wifi” |
| geo_code | 地理位置 | 经纬度GeoHash编码 |
| ip1_sub | IP地址 | 前三位操作设备IP地址编码加密（ip1前三位IP地址）<br>比如，原字段为12，34，56，7和12，34，56，8的ip地址前三位都为12，34，56，故脱敏后的值是一样的 |
| ip2_sub | IP地址 | 前三位操作电脑IP地址编码加密（ip2前三位IP地址） |

# 交易详单数据字典

| 字段名 | 中文解释 | 字段说明 |
| --- | --- | --- |
| UID | 用户编号 | |
| channel | 平台 | 平台类型 |
| day | 交易日期 | 连续的日期标识，<br>1为第一天，2为第二天，以此类推 |
| time | 交易时间点 | |
| trans_amt | 脱敏后交易金额 | 保留大小关系 |
| amt_src1 | 资金类型 | 交易资金源类型，例如“余额”、“银行卡” |
| merchant | 商户标识 | 商户编码加密 |
| code1 | 商户标识 | 商户子门店编码加密 |
| code2 | 商户终端设备标识 | 商户交易终端设备编码加密 |
| trans_type1 | 交易类型1 | 交易类型，例如“消费”，“退款” |
| acc_id1 | 账户相关 | 用户交易账户号编码加密 |
| device_code1 | 操作设备唯一标识1 | 设备号唯一标识加密，可用于安卓类设备的唯一标识 |
| device_code2 | 操作设备唯一标识2 | 设备号唯一标识加密，可用于安卓类设备的唯一标识<br>（唯一标识码并不会只是一种 但都能达到效果） |
| device_code3 | 操作设备唯一标识3 | 设备号唯一标识加密，可用于苹果类设备的唯一标识 |
| device1 | 操作设备参数1 | 设备名称加密，原字段如“*Jack’s iphone*” |
| device2 | 操作设备参数2 | 设备型号 |
| mac1 | MAC地址 | 操作设备MAC地址编码加密，<br>原字段如“38:XX:XX:XX:XX:92” |
| ip1 | IP地址 | 操作设备IP地址编码加密 |
| bal | 脱敏后账户余额 | 保留大小关系 |
| amt_src2 | 资金类型 | 交易资金源类型，与1类型相似，2对银行卡做了细分 |
| acc_id2 | 账户相关 | 转账操作的转出账户号编码加密 |
| acc_id3 | 账户相关 | 转账操作的转入账户号编码加密 |
| geocode | 地理位置 | 经纬度GeoHash编码 |
| trans_type2 | 交易类型2 | 交易类型，例如“线上”、“线下”<br>trans_type2与trans_type1的维度和侧重不同 |
| market_code | 营销活动号编码 | 营销活动号编码加密 |
| market_type | 营销活动标识 | 营销活动类型 |
| ip1_sub | IP地址 | 前三位操作设备IP地址编码加密（ip1前三位IP地址） |

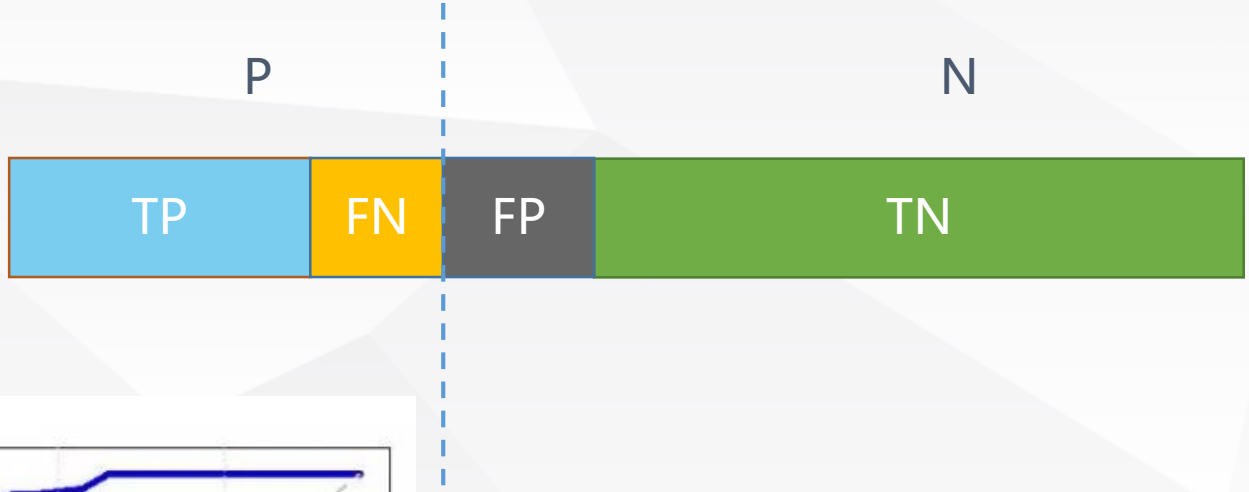
# 赛题任务

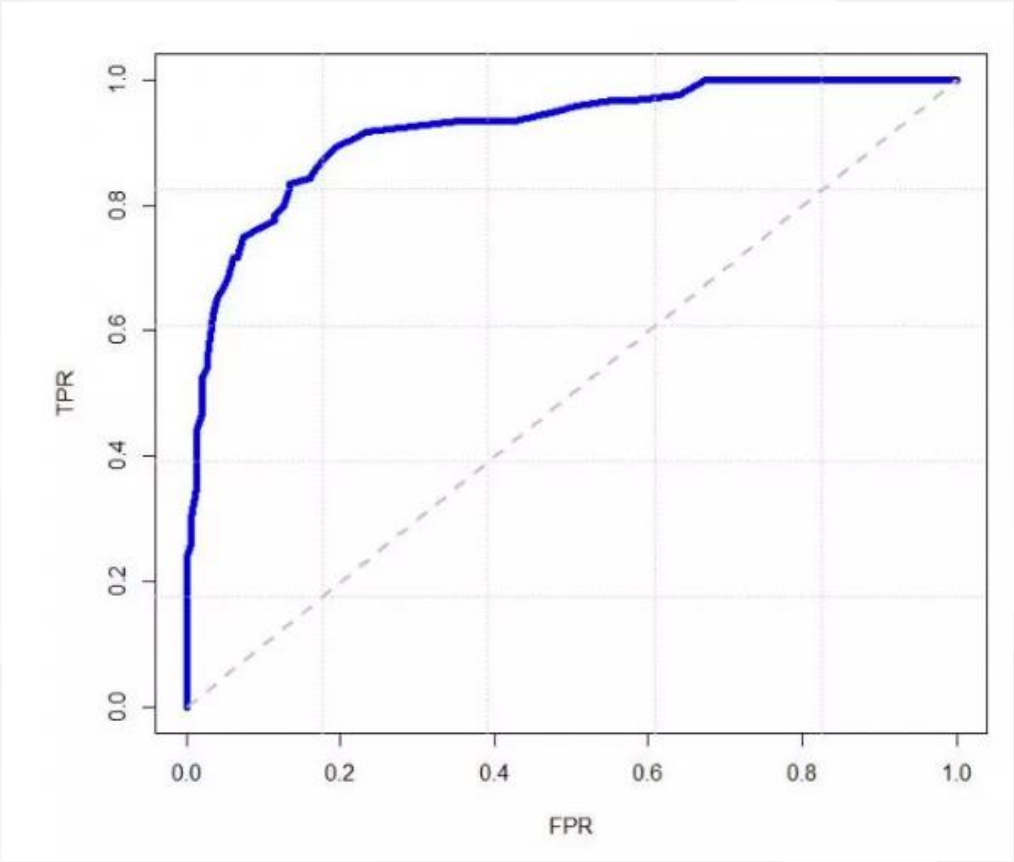



通过训练学习用户在消费过程中的关联操作、交易详单信息，来识别“羊毛党”

二分类问题

# 评价指标




TPR = TP / （TP + FN）
FPR = FP / （FP + TN）
其中，TP、FN、FP、TN分别为真正例、假反例、假正例、真反例。

TPR1：FPR=0.001时的TPR
TPR2：FPR=0.005时的TPR
TPR3：FPR=0.01时的TPR
score= 0.4*TPR1+0.3*TPR2+0.3*TPR3

# 评价指标的分析

| 样本 | 概率 | 标签 |
| --- | --- | --- |
| 1 | 0.99 | 1 |
| 2 | 0.92 | 0 |
| 3 | 0.91 | 1 |
| 4 | 0.90 | 0 |
| 5 | 0.85 | 1 |
| 6 | 0.84 | 0 |
| 7 | 0.83 | 0 |
| 8 | 0.75 | 1 |
| 9 | 0.63 | 0 |
| 10 | 0.50 | 0 |

TPR1：FPR=0.001时的TPR
TPR2：FPR=0.005时的TPR
TPR3：FPR=0.01时的TPR
score= 0.4*TPR1+0.3*TPR2+0.3*TPR3

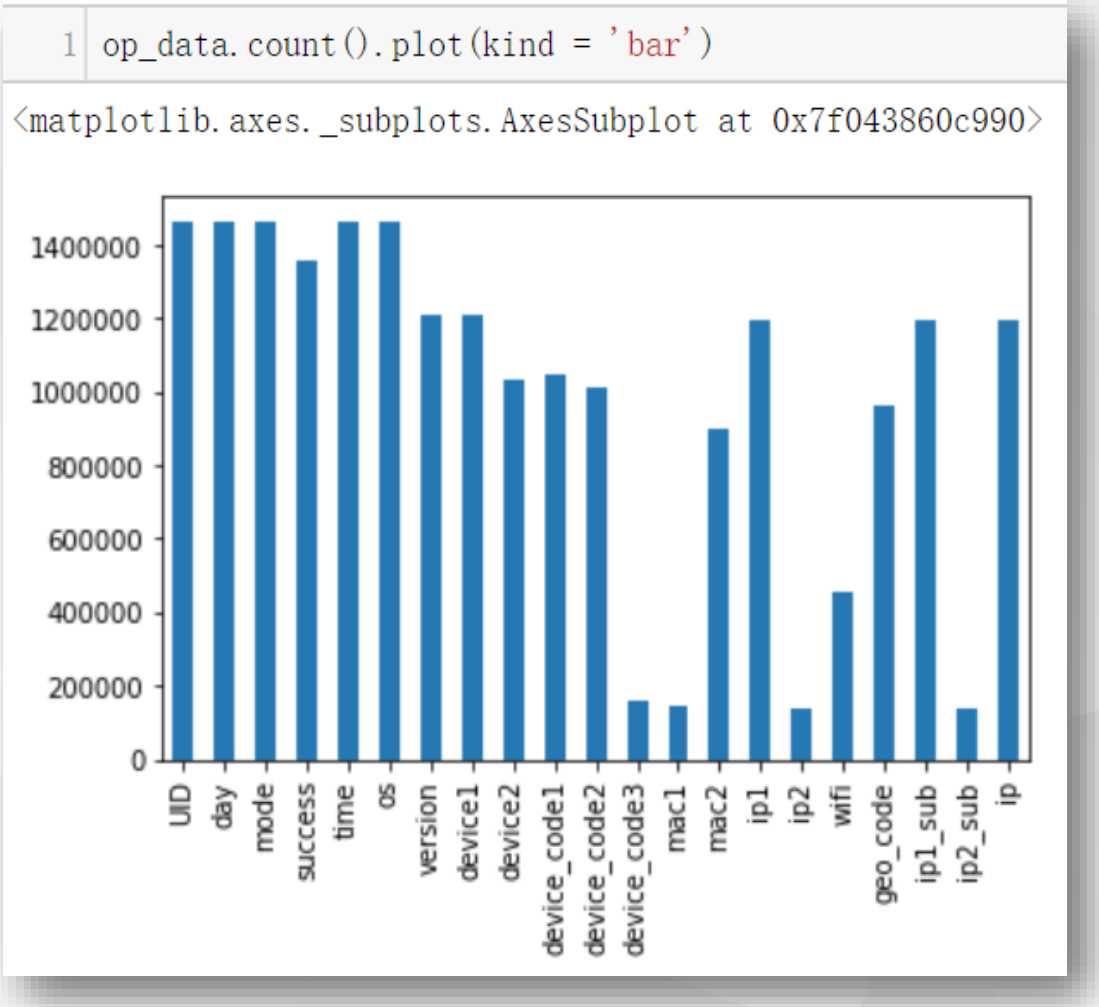
FPR小意味着此时被冤枉的为“羊毛党”的用户少

TPR大意味着只要是“羊毛党”，都应该尽可能找出来

# 数据探索

```
1 op_data.count().plot(kind = 'bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f043860c990>
```

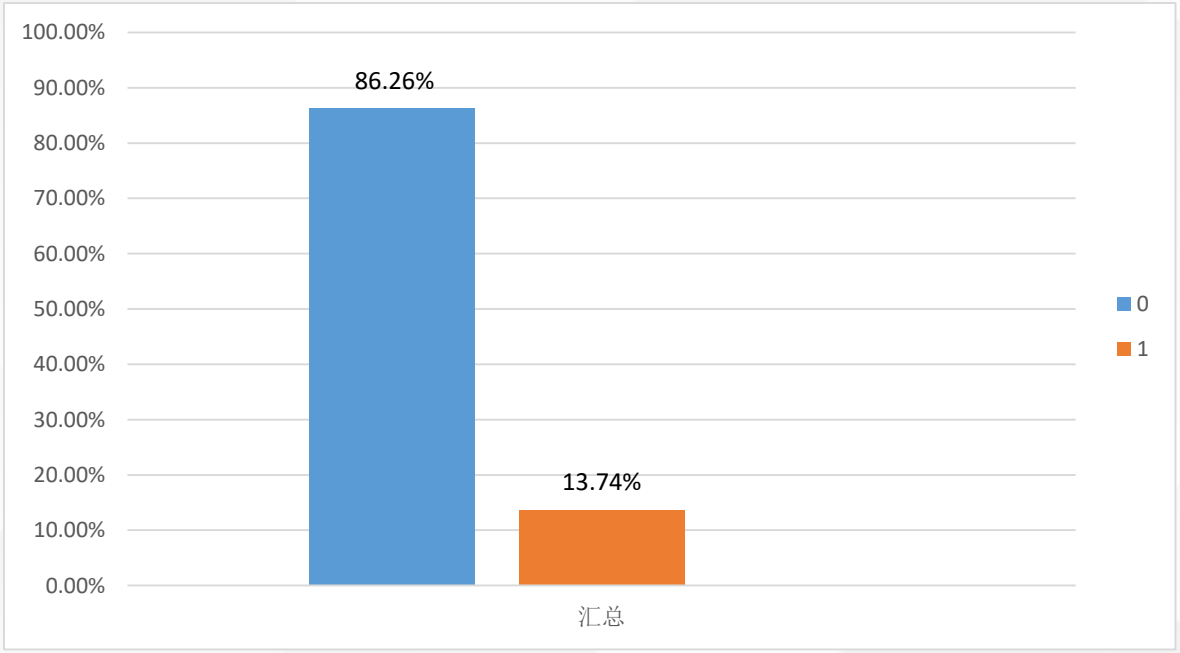



1. 删除字段

2. 数据补齐

# 数据探索

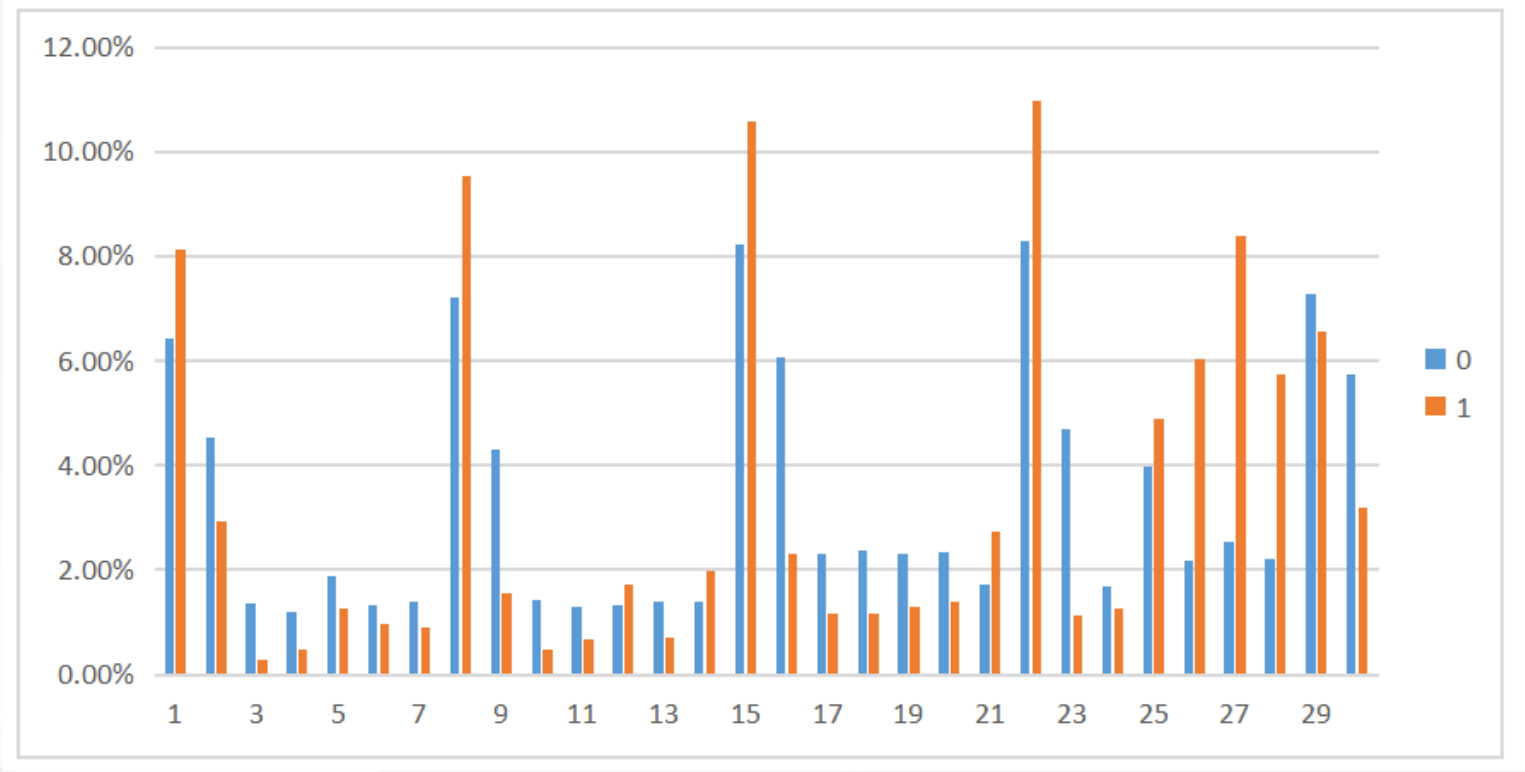
## 训练集正负样本比例




羊毛党样本比例小，且复赛测试集中羊毛党比例更小。

# 数据探索

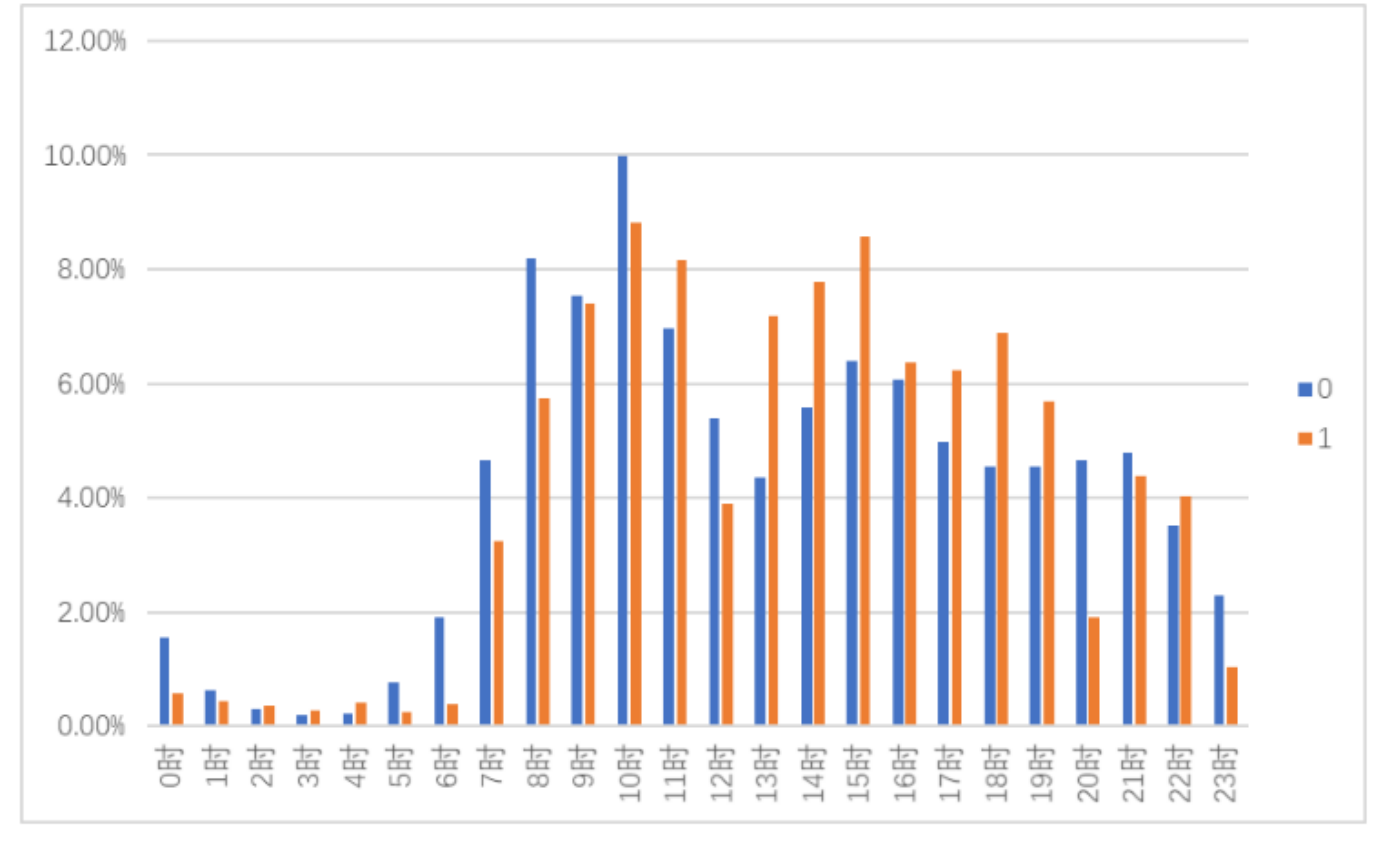
Day




可以看出，用户行为数据在day上呈现出了周期性的特征，可以作为判断是周几的依据。

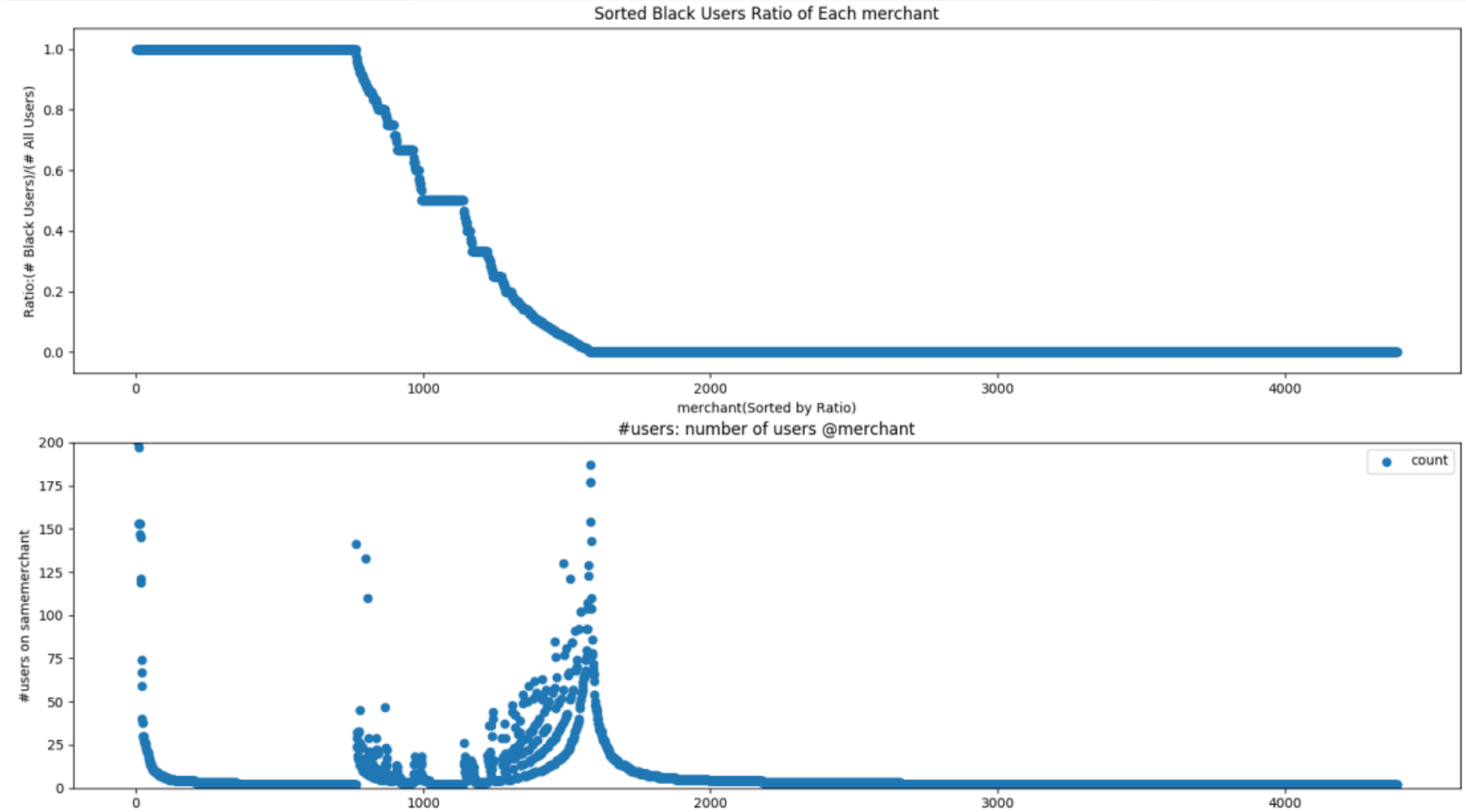
# 数据探索

Time


12.00%
10.00%
8.00%
6.00%
4.00%
2.00%
0.00%
0时
1时
2时
3时
4时
5时
6时
7时
8时
9时
10时
11时
12时
13时
14时
15时
16时
17时
18时
19时
20时
21时
22时
23时
0
1

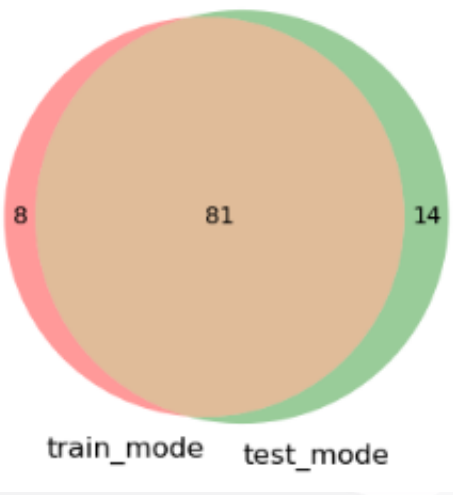
# 数据探索

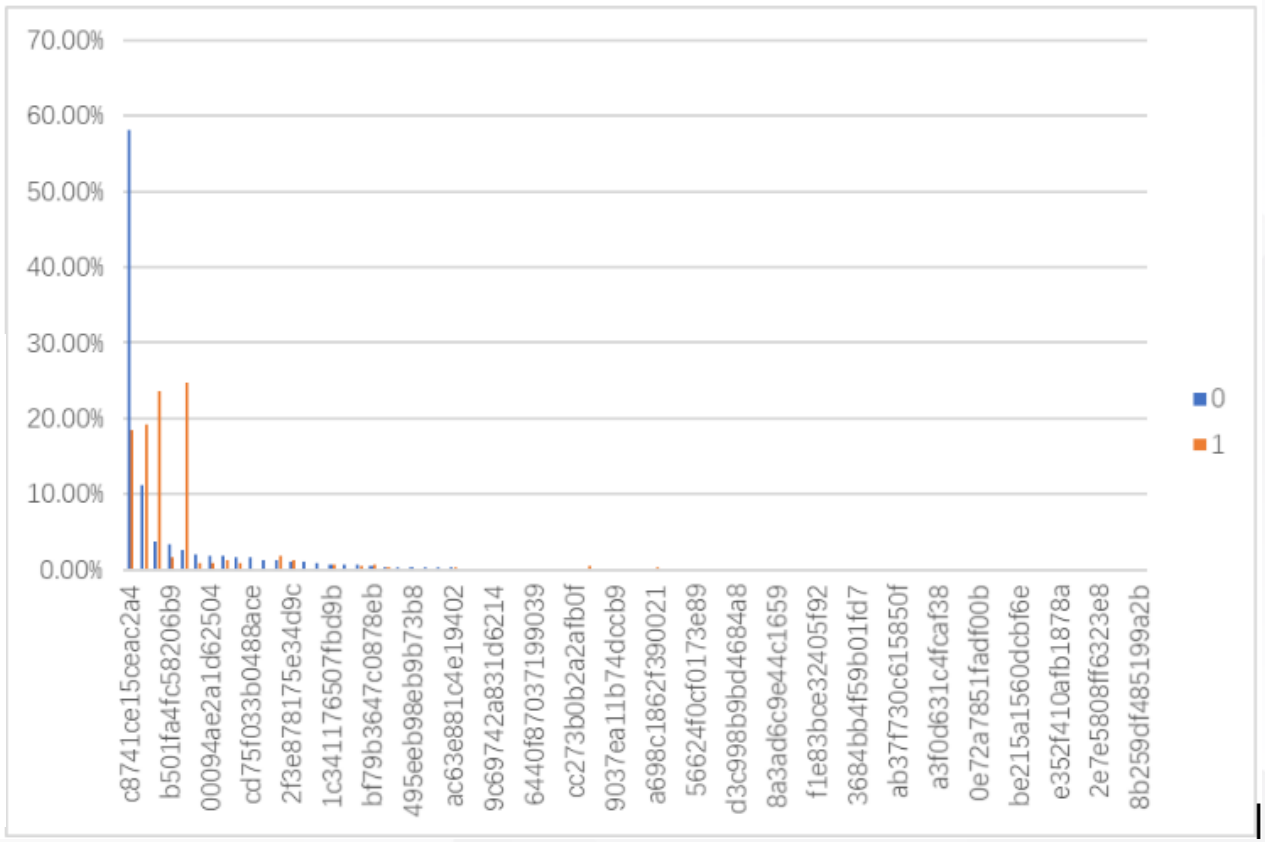
Merchant



# 数据探索

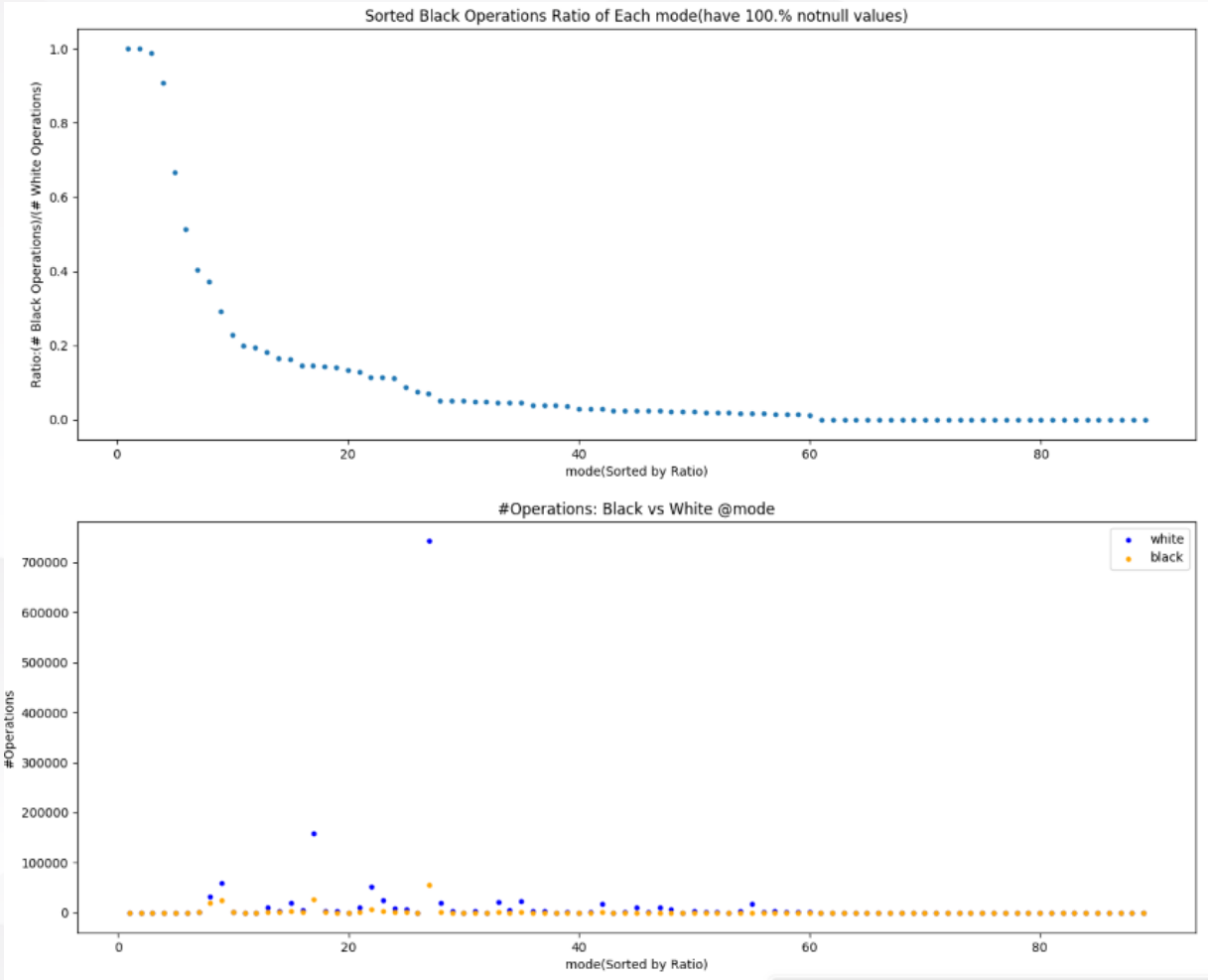
举例：对operation中正负样本在mode上的数据探索



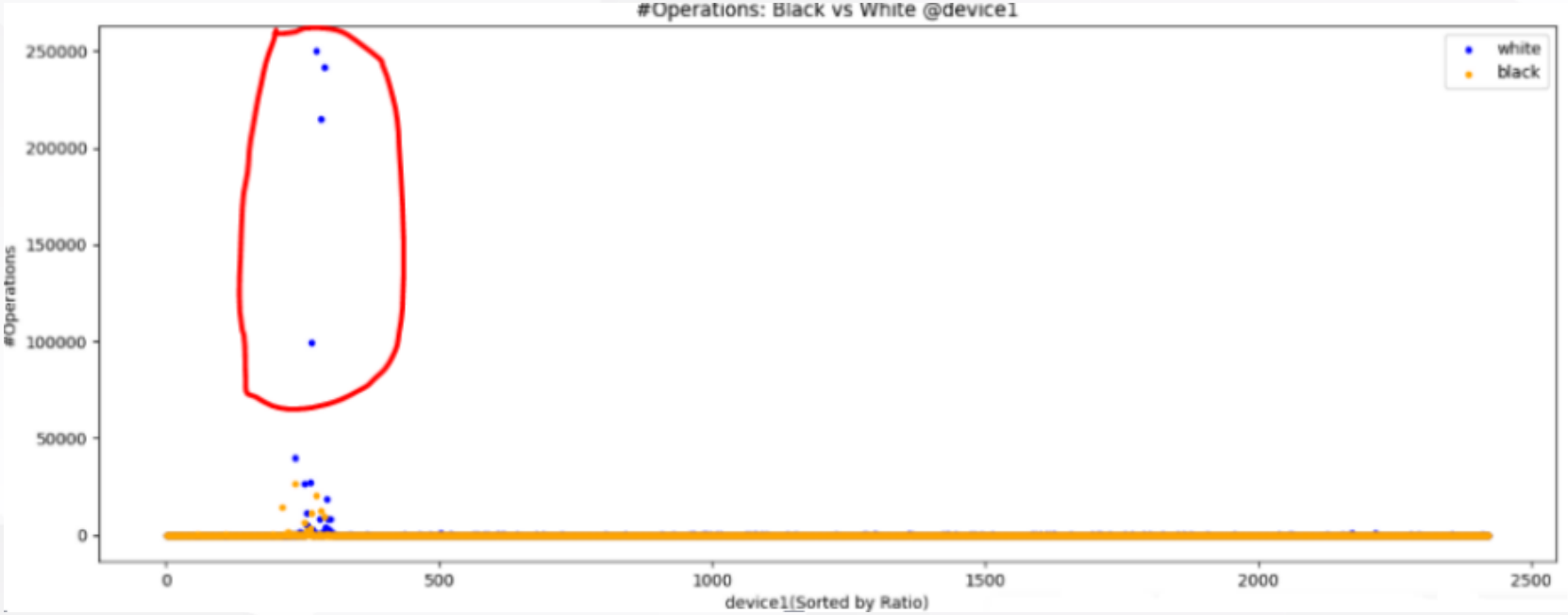
# 数据探索

举例：对operation中正负样本在mode上的数据探索



# 数据探索

异常数据处理


#Operations: Black vs White @device1
white
black
250000
200000
150000
100000
50000
0
#Operations
0
500
1000
1500
2000
2500
device1(Sorted by Ratio)

# 特征工程

# 频次统计

os
mode
trans_type
…

Groupby用户id

① 最小值
② 最大值
③ 累加值
④ 平均数

| | UID | os | os_count |
|---|---|---|---|
| 0 | 10000 | 103 | 9 |
| 1 | 10001 | 102 | 49 |
| 2 | 10001 | 200 | 17 |
| 3 | 10001 | 104 | 1 |
| 4 | 10002 | 102 | 10 |
| 5 | 10002 | 101 | 1 |
| 6 | 10003 | 102 | 15 |
| 7 | 10004 | 102 | 34 |
| 8 | 10006 | 102 | 4 |
| 9 | 10006 | 200 | 3 |
| 10 | 10007 | 102 | 13 |

| | UID | max | min | mean |
|---|---|---|---|---|
| 0 | 10000 | 26 | 13 | 20.222222 |
| 1 | 10001 | 13 | 2 | 6.522388 |
| 2 | 10002 | 29 | 29 | 29.000000 |
| 3 | 10003 | 17 | 17 | 17.000000 |
| 4 | 10004 | 29 | 1 | 12.647059 |
| 5 | 10006 | 8 | 7 | 7.571429 |
| 6 | 10007 | 26 | 18 | 22.307692 |

# 频次统计

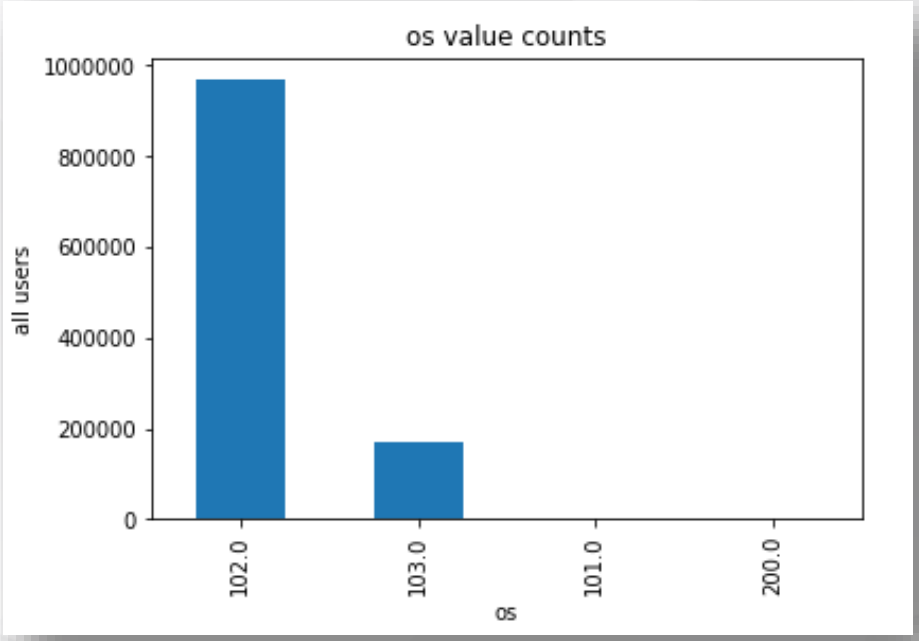
将类别字段每个取值频次都作为特征

```
op_Data.groupby('UID').os.value_counts().unstack().reset_index().fillna(0
)
```

| | UID | os | os_count |
|---|---|---|---|
| 0 | 10000 | 103 | 9 |
| 1 | 10001 | 102 | 49 |
| 2 | 10001 | 200 | 17 |
| 3 | 10001 | 104 | 1 |
| 4 | 10002 | 102 | 10 |
| 5 | 10002 | 101 | 1 |
| 6 | 10003 | 102 | 15 |
| 7 | 10004 | 102 | 34 |
| 8 | 10006 | 102 | 4 |
| 9 | 10006 | 200 | 3 |
| 10 | 10007 | 102 | 13 |

| os | UID | 101 | 102 | 103 | 104 | 105 | 107 | 200 |
|---|---|---|---|---|---|---|---|---|
| 0 | 10000 | NaN | NaN | 9.0 | NaN | NaN | NaN | NaN |
| 1 | 10001 | NaN | 49.0 | NaN | 1.0 | NaN | NaN | 17.0 |
| 2 | 10002 | 1.0 | 10.0 | NaN | NaN | NaN | NaN | NaN |
| 3 | 10003 | NaN | 15.0 | NaN | NaN | NaN | NaN | NaN |
| 4 | 10004 | NaN | 34.0 | NaN | NaN | NaN | NaN | NaN |
| 5 | 10006 | NaN | 4.0 | NaN | NaN | NaN | NaN | 3.0 |
| 6 | 10007 | NaN | 13.0 | NaN | NaN | NaN | NaN | NaN |

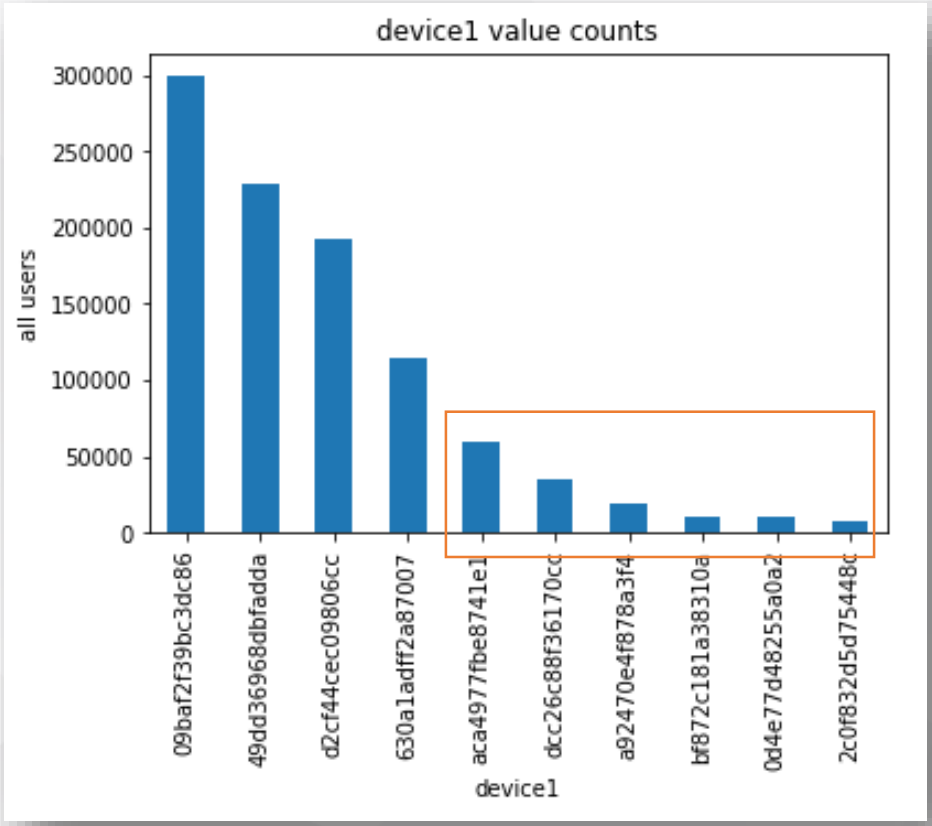
# 频次统计




## 直接统计

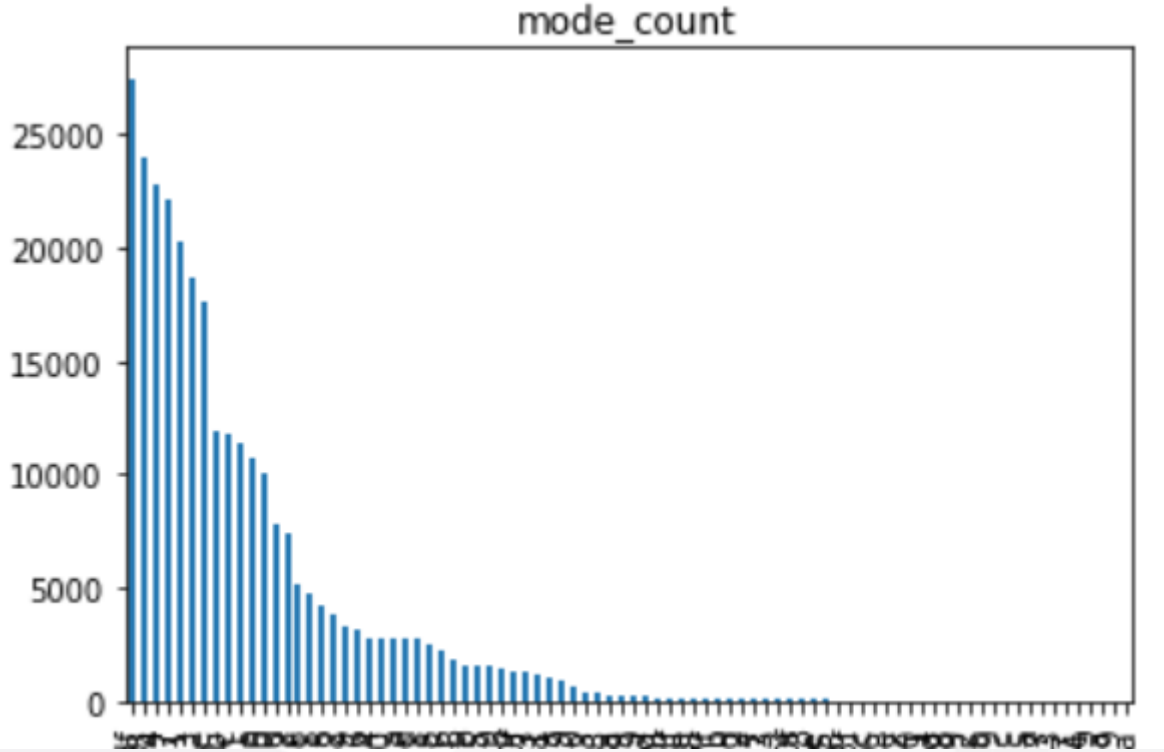
类别种类较少
类别值基本不变

# 频次统计




Topk：类别种类很多，类别值变化大，将出现较少的归为“其它取值”

# 频次统计

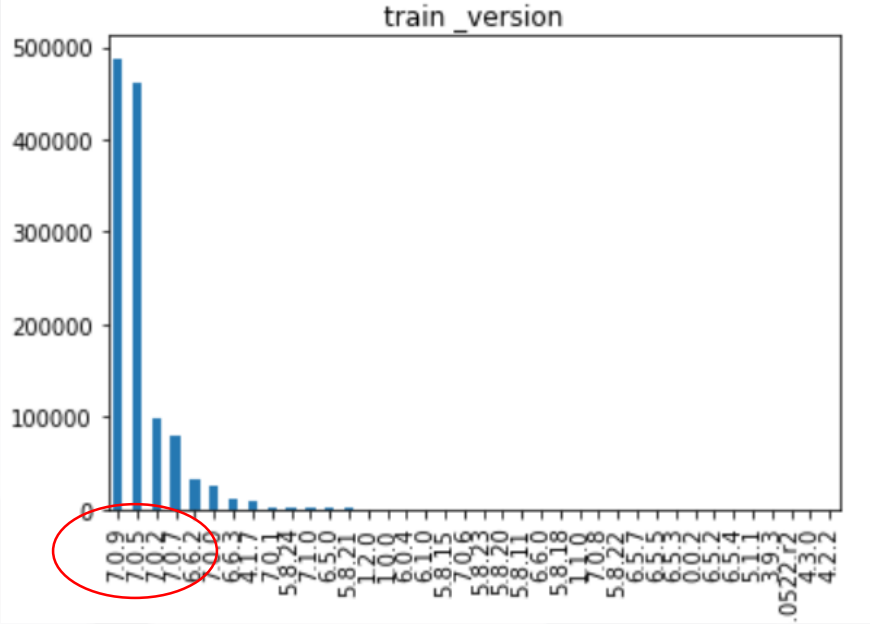


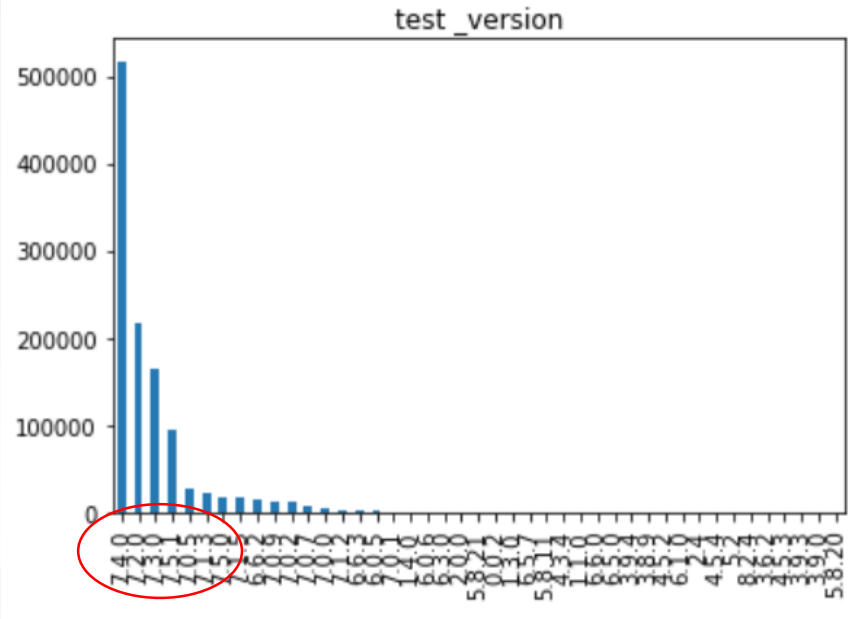

分箱

pd.cut(x, bins)

pd.qcut(x, q)

# 流行度


train _version
500000
400000
300000
200000
100000
0


test _version
500000
400000
300000
200000
100000
0

# 流行度

含义

字段映射为数值， 数值反映字段取值的用户数量

| version | 流行度 |
| --- | --- |
| 7.0.9 | 17432 |
| 7.0.5 | 16753 |
| 7.0.7 | 4004 |
| 7.0.2 | 3463 |
| 6.6.2 | 1331 |

Train

| version | 流行度 |
| --- | --- |
| 7.4.0 | 17521 |
| 7.2.0 | 14659 |
| 7.3.0 | 7819 |
| 7.5.1 | 4932 |
| 7.5.0 | 1657 |

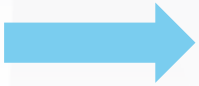
Test

# 地理信息字段处理

geohash编码：二维的经纬度转换成字符串

| | geo_code |
|---|---|
| 0 | wskx |
| 3 | wm4v |
| 5 | ws90 |
| 6 | ws45 |
| 7 | ws2x |
| 9 | ws82 |
| 10 | wkjm |
| 11 | wwdj |
| 12 | wmxb |
| 13 | wwew |
| 14 | wtgf |

```
import geohash
lambda x:geohash.decode(x)
```



| | geo_code |
|---|---|
| 0 | (25, 119) |
| 3 | (29, 105) |
| 5 | (25, 114) |
| 6 | (23, 115) |
| 7 | (25, 113) |
| 9 | (25, 113) |
| 10 | (23, 109) |
| 11 | (38, 115) |
| 12 | (31, 112) |
| 13 | (38, 118) |
| 14 | (33, 118) |

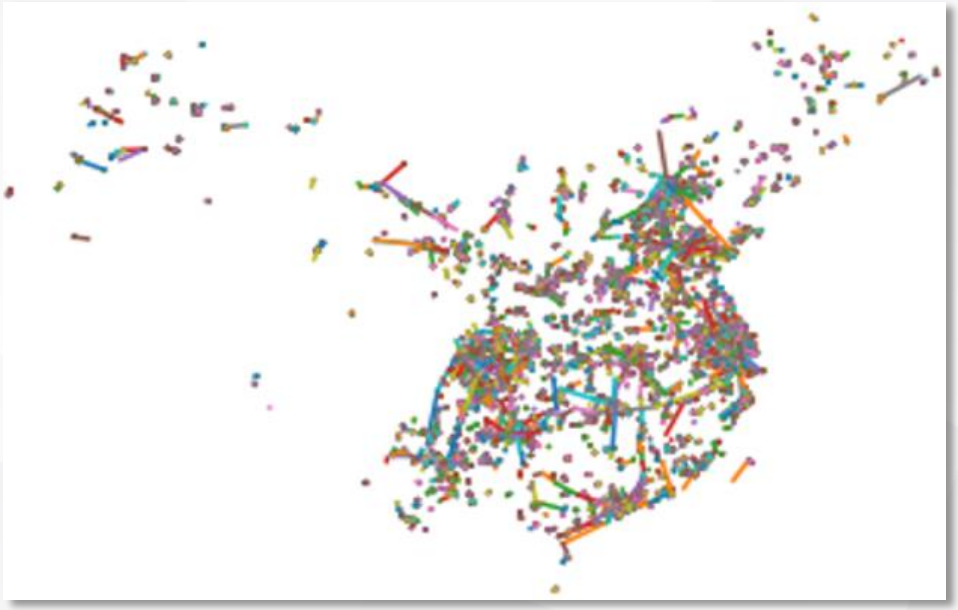
tips：一般的工具需要手动安装geohash包

python的话在cmd输入命令“pip install Geohash”

# 地理信息字段处理

**“羊毛党”**

- 恶意篡改地理位置信息
- 使用vpn



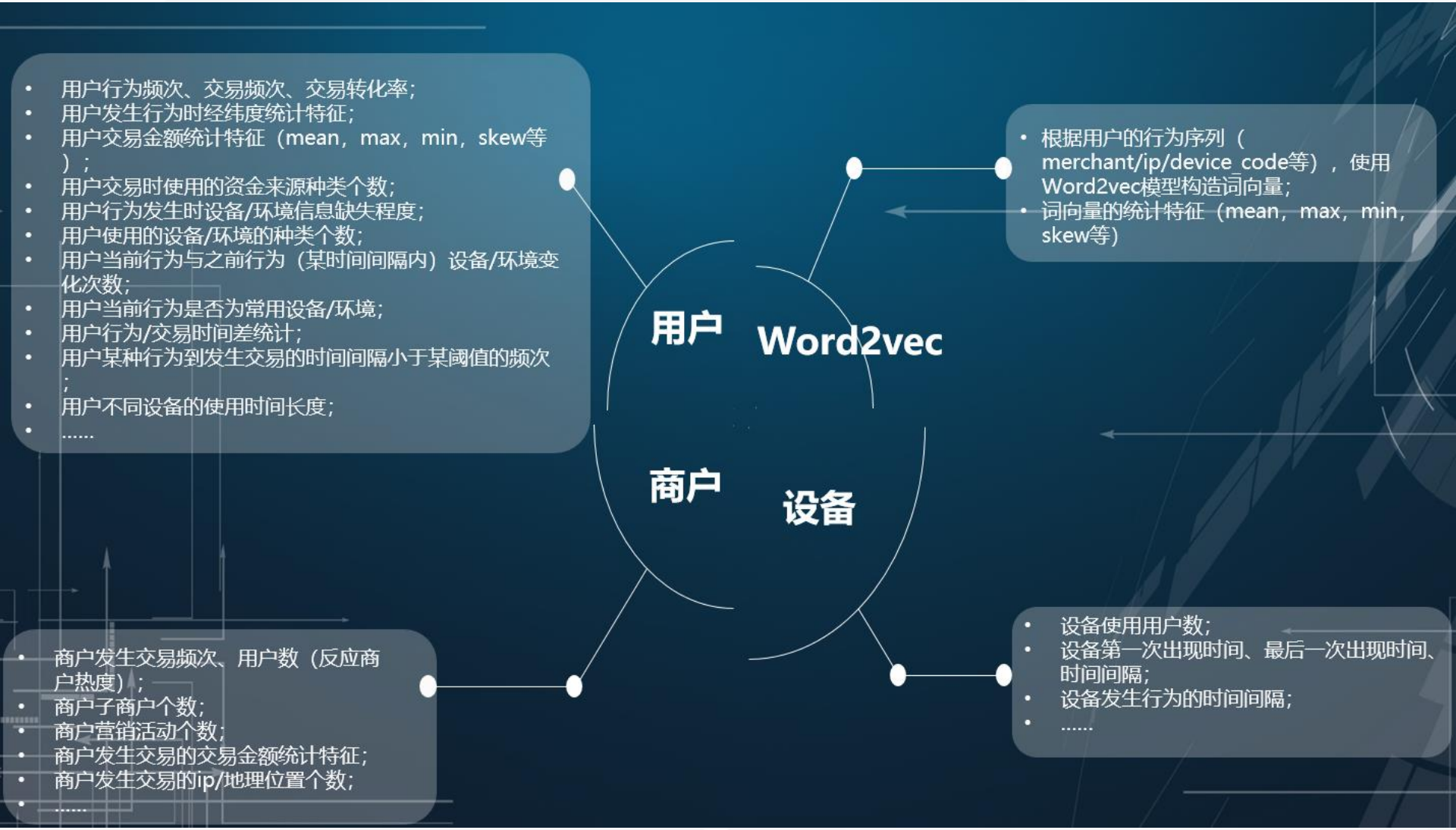
经纬度

按月统计　　按天统计

极值、跨度、方差等特征

活动范围、活动位置变化

# 特征工程


用户行为频次、交易频次、交易转化率；
用户发生行为时经纬度统计特征；
用户交易金额统计特征（mean，max，min，skew等）；
用户交易时使用的资金来源种类个数；
用户行为发生时设备/环境信息缺失程度；
用户使用的设备/环境的种类个数；
用户当前行为与之前行为（某时间间隔内）设备/环境变化次数；
用户当前行为是否为常用设备/环境；
用户行为/交易时间差统计；
用户某种行为到发生交易的时间间隔小于某阈值的频次；
用户不同设备的使用时间长度；
……
根据用户的行为序列（merchant/ip/device_code等），使用Word2vec模型构造词向量；
词向量的统计特征（mean，max，min，skew等）
用户
Word2vec
商户
设备
商户发生交易频次、用户数（反应商户热度）；
商户子商户个数；
商户营销活动个数；
商户发生交易的交易金额统计特征；
商户发生交易的ip/地理位置个数；
……
设备使用用户数；
设备第一次出现时间、最后一次出现时间、时间间隔；
设备发生行为的时间间隔；
……

# 模型选择

## 树模型

- XGBoost
- LightGBM
- CatBoost
- RandomForest
- ExtraTree
- Regularized Greedy Forest (RGF)
- GradientBoostingClassifier

- 可解释性强
- 可处理混合类型特征
- 具体伸缩不变性（不用归一化特征）
- 有特征组合的作用
- 可自然地处理缺失值
- 对异常点鲁棒
- 有特征选择作用
- 可扩展性强，容易并行

- 缺乏平滑性（回归预测时输出值只能输出有限的若干种数值）
- 不适合处理高维稀疏数据

## 其他模型　　LR、朴素贝叶斯等

# 模型选择

- 选择的模型及模型参数：
- lgb.LGBMClassifier (boosting_type='gbdt', num_leaves=200, reg_alpha=3,reg_lambda=5, max_depth=-1,n_estimators=5000, objective='binary', subsample=0.9, colsample_bytree=0.77,subsample_freq=1, learning_rate=0.05,random_state=1000, n_jobs=16, min_child_weight=4, min_child_samples=5, min_split_gain=0)

# 线下验证

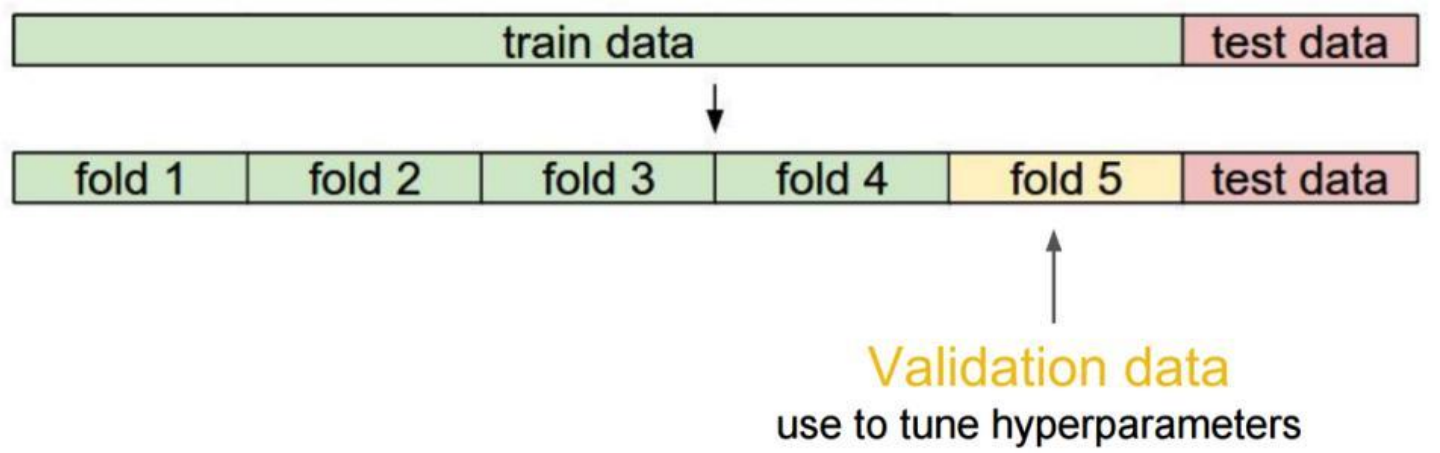
**提交次数有限**

**实际应用需求**

# 线下验证

- 根据数据分布抽样：例如分层抽样

- CV交叉验证



# 扩展思考

| id | 句子 |
| --- | --- |
| 1 | [i, like, data, mining, … , etc] |
| 2 | [i, don’t, know, what, … ,etc] |
| … | … |

| id | 一月内操作类型序列 |
| --- | --- |
| 1 | [mode1, mode1, mode2, mode4, … ,mode1] |
| 2 | [mode4, mode5, mode4, mode4, … ,mode2] |
| … | … |

# 扩展思考

- 只用统计特征是不够的，因为测试数据和训练数据时间相隔太久，复赛测试数据与训练数据时间相隔更久。
- 该问题的本质是探索羊毛党的行为模式以及羊毛党之间的关联关系。

# 扩展思考

- 关联图谱、图数据库
- 社区发现算法，比如COPRA算法
- 异构信息网络，比如NetClus网络聚类

# 磨刀功夫

**注意记录：避免重复工作、避免遗漏信息**

- 分析数据、业务——潜在的信息
- 特征改动——线上、线下效果
- 参数调整——线上、线下效果

自动日志：python的logging包

手动记录

# 磨刀功夫

## 多看评论和赛后解法

- 每一次比赛后其实都对实际的Data Mining问题有新的认识，总结自己的方案和其他top的方案，这是在比赛方面提高的关键

## 不断学习

- 不断接触前沿paper，不断学习当下最新算法，可以参考工业界有效的方案

## 切记只看不做

- 数据挖掘不等于机器学习

# 要求

- 到Data Castle报名参赛，单人参赛。注意队名为：DM+学号
  http://www.dcjingsai.com/common/cmpt
- 下载数据，完成比赛
- 考核：线上分数+模型亮点
- baseline会发到微信群里

http://www.inpluslab.com

移动互联网与金融大数据实验室