# Forecast Model Design



## 文档更新记录

- 相关文档

    - alice organic BRD： https://docs.google.com/document/d/1aFdFpbZRAp-4M3XIHCdKApeC58ur_HUAnARJP19WawM/edit#heading=h.bnd7714bpvwi
    - alice的campaign BRD: https://docs.google.com/document/d/1-6nTFCDFzu7hmStB6bBwA36WDrsxgFSnayJiSxNVVLY/edit
    - dylan 的data BRD: https://docs.google.com/document/d/1IGYYGdtf6n9EVmh5lKtDmILWbit5TwtLsWWSzB0YUNE/edit#heading=h.kujo3z6p51p5
    - kezhen 的数据需求文档： Forecasting Data Requirement Document
    - jiaqi 的数据流流程：https://drive.google.com/file/d/1NdA8hh56gXbMy5wDXznpRuM6BgUZLQJY/view?ts=62736da3
    - jiaqi手机数据字段信息： https://docs.google.com/spreadsheets/d/1je7f1XdMRGCMM_9oejG3lmI22mNAGRMKDS7r4GfG53k/edit#gid=1811647913
- 04-30对更新模型文档：
    - 问题跟踪记录文档： https://docs.google.com/document/d/10iaNE0g_BhYydt-YnCUShMdgu5EC2ybepNlm_ZJpyc0/edit
    - organic forecast的Pipeline 文档： https://docs.google.com/document/d/12cEH5yOiZ3BdG375Y5eCLwQuRijLGs8vE8TxYcN8Jag/edit#
    - organic forecast 的说明ppt: https://docs.google.com/presentation/d/1T3swVbc4z5N7aue2KP-WK6CCuMAjh-aY/edit#slide=id.p4
- 05-25 BRD更新
    - 文档: https://docs.google.com/document/d/1IGYYGdtf6n9EVmh5lKtDmILWbit5TwtLsWWSzB0YUNE/edit#
    - metric: https://docs.google.com/spreadsheets/d/1O7WgooaZeyILukGv2m4HQ2-tZ0bJLFkHZ6eCwZh7Tkg/edit#gid=493120267
- 05-30 Alice的验收标准文档： https://docs.google.com/presentation/d/196Q5CxY3N9XV438byLycA_bFxZ-SOU0fTVIhWG4ahmw/edit#slide=id.g12edffb3cbf_0_51
- 05-31 jiiaqi DRD 文档： https://confluence.shopee.io/pages/viewpage.action?pageId=1114958779
- 06-10 lifu data 技术文档： Sales Forecasting数据方案设计 , 具体字段设计 https://docs.google.com/spreadsheets/d/1NDGqg_8q665bVrAsNkILZf9vvZpcXtd9SCXesgKGVms/edit#gid=397945128
- 业务的campaign calendar 文档： https://docs.google.com/spreadsheets/d/1HSZcAobXiVBZ5cNSi0j0RRzsSsYtFTndrk8vrEpIq8k/edit#gid=465213523
- forecast metric :
    - BRD:https://docs.google.com/document/d/1Ddr8Efrz8-49CxAM8g3sCYuQbVlsyWixRA5lHLsTZB8/edit#heading=h.t1gragyg4kqv
    - Metrics Design:https://docs.google.com/spreadsheets/d/1GMs_-BWu-y3dC-W1SFoYq8J9s2OkAAh9ztVm-0iFYO0/edit#gid=1948572567

# 主要业务补充说明

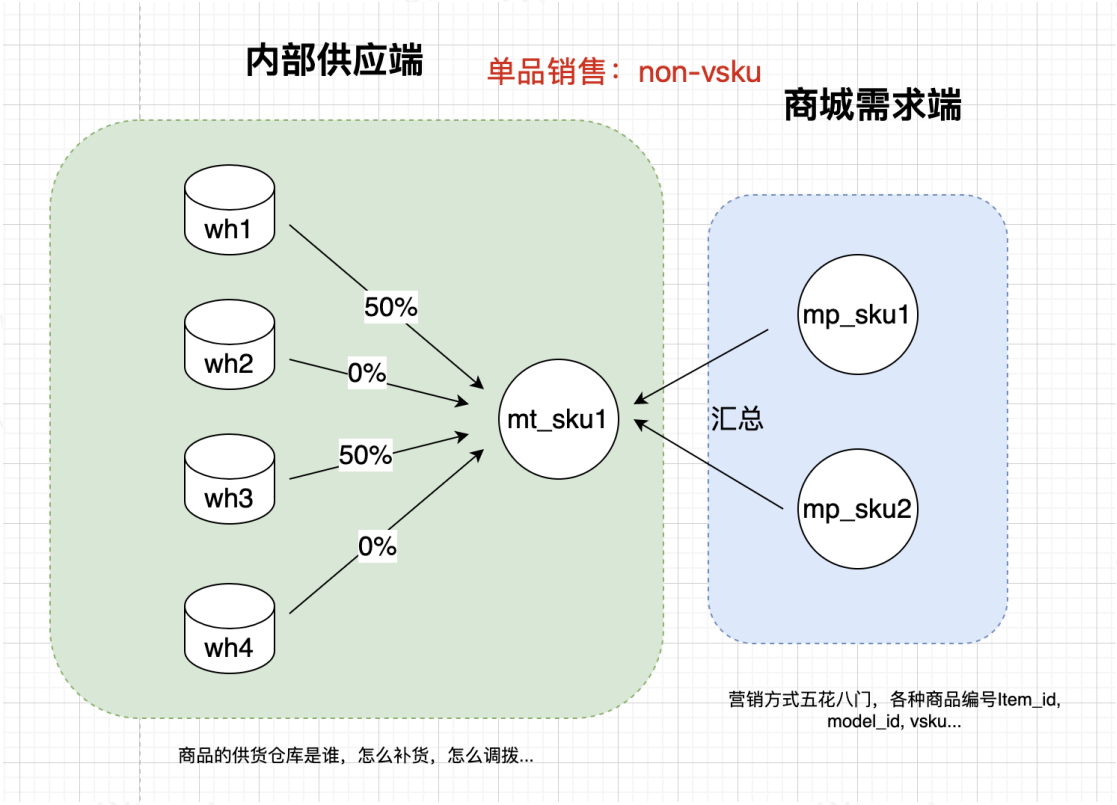
整体项目概述参考 project overview

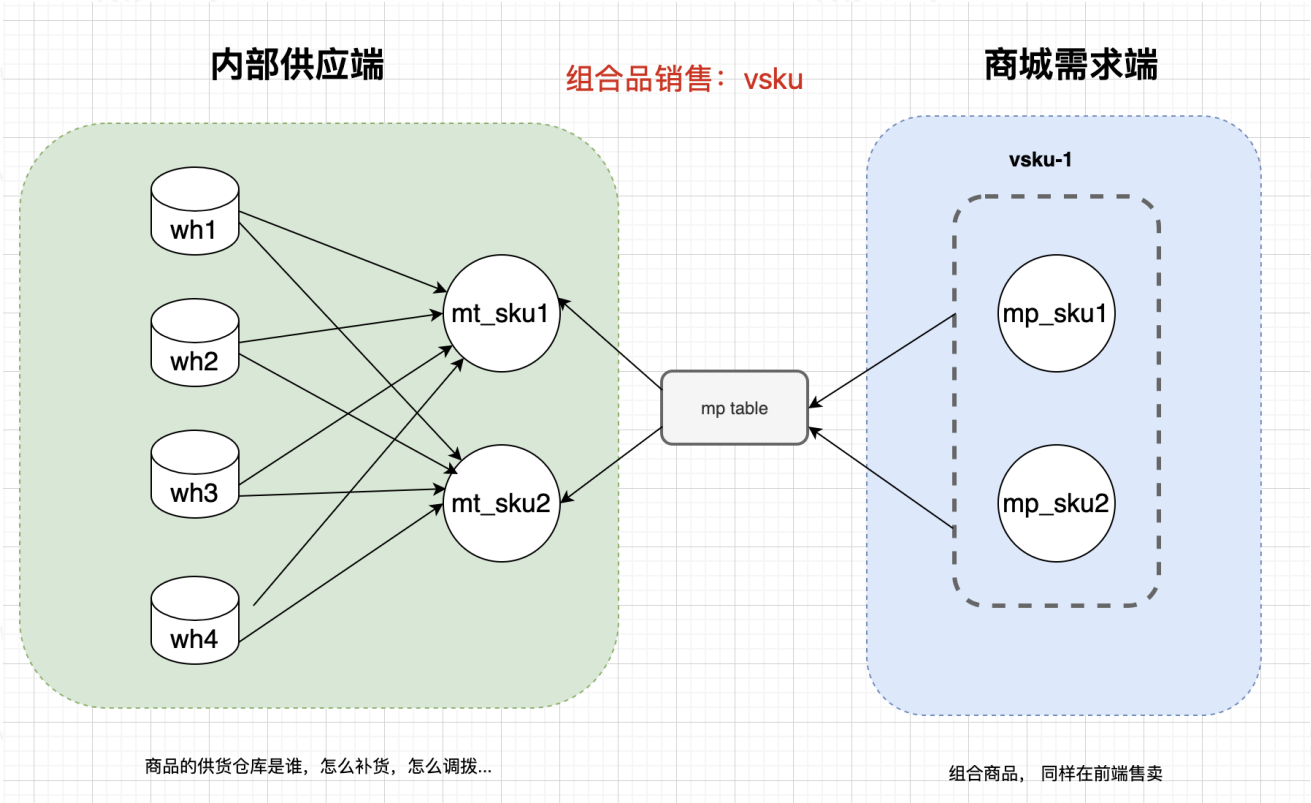
## organic vs campaign

- organic: 排除掉促销日的销量预测，也就是我们常说的“平销”
- campaign: 促销预测，主要分成两类：
    - shop-wised campagin: 店铺级别的促销活动，如 super brand day， 一般是到shop + mp_sku级别的价格折扣
    - platform-wised campagin： 全平台级别的促销活动，如double day，可以理解为国内淘宝的双11, 全平台有大规模的引流，不一定有价格优惠

## mp sku vs mt sku

- mp sku: marketplace sku, 即商城列表上可供购买的商品
- mt sku: merchant sku, 补货商品




举个简单例子，mp_sku1 是shopee旗舰店1的“iphone 13 白色”， mp_sku2 是shopee旗舰店2的“iphone 13 白色”， 对应的mt_sku1就是"iphone 13 白色", 并且该商品50% 由wh1供货，50% 由wh2供货，

**注意：**

- mp sku是商城界面可供选择的商品， 由item_id + model_id 构成， 其复杂性更高，上述例子只是简单说明
- 对于mt_sku 和 wharehouse的对应关系， 上述例子中的比例是需要根据历史数据进行拆分计算的，当前拆分逻辑是按照过去30/60/90天的平均销量。 但常规来说供货规则如果固定下来会减少供应的不确定性，有利于补货，待后续跟进！

- 当前系统设计：
  - organic: 由于运营人员兼顾了需求计划和补货计划，所以organic 预测直接在mt_sku颗粒度进行，但耦合性高，看如何在后续的系统侧迭代
  - campaign: 各类促销活动都是和mp_sku挂钩的，因此campagin端预测是在mp_sku颗粒度上进行，后将mp_sku的预测结果汇总到mt_sku + wh进行补货

## city warehouse vs physical warehouse

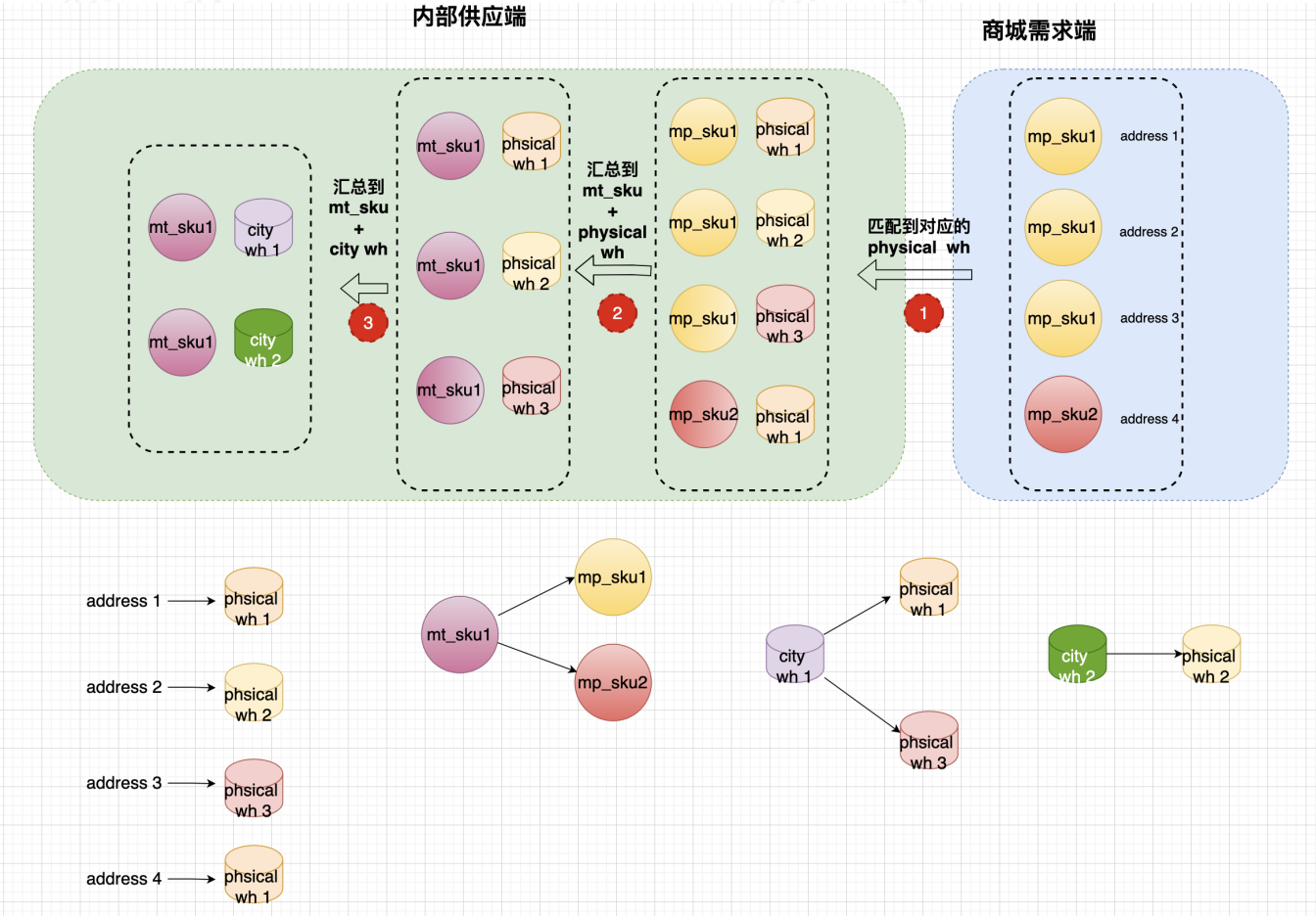
city warehouse代表的是一个虚拟仓，目前来看在ID,一个城市就是一个虚拟仓， 并且实际可能对应多个实体仓(physical warehouse)。

但是注意由于当前补货的决定是基于city warehouse, 所以我们在**下文的 warehosue都代表city_warehouse**

| City/City Warehouse | Actually Warehouse/Buyer Warehouse |
|---|---|
| JKT | ('IDG', 'IDL', 'IDE', 'IDF', 'IDH', 'IDC', 'IDA') |
| SBY | 'IDS' |
| MDN | 'IDM' |
| MKS | 'IDK' |
| BPN | 'IDN' |
| SMG | 'IDR' |
| PLB | 'IDP' |

## 结合mp_sku, mt_sku, physical_warehouse, city_warehouse的图示关系

下图是从结合mp/mt，physical/city 的数据流转图（注意这是最细颗粒度的数据流转图，实际的预测流转图不一定按照这样的方式进行）




对于存在vsku的场景，逻辑上跟上述一致，不过在步骤2中会多一张map表，用来关联vsku, mp_sku, mt_sku的关系。

## 不同场景下的预测颗粒度对比情况

注意下面的颗粒度**仍待讨论**:

| | organic | | campaign | |
|---|---|---|---|---|
| 阶段 | as-is | to-be | as-is | to-be |
| 预测接粒度 | mt_sku + city_wh | mp_sku | mp_sku | mp_sku |
| 基础数据 | mt_sku + city_wh的adis统计表 | mp历史销量表 | mp历史销量表 | mp历史销量表 |
| 如何转化为mt+city_wh的结果 | 不需要转换 | 1：汇总到mt_sku<br>2: 根据历史销售情况按比例拆分到mt_sku + city_wh | 1：汇总到mt_sku<br>2: 根据历史销售情况按比例拆分到mt_sku + city_wh | 1：汇总到mt_sku<br>2: 根据历史销售情况按比例拆分到mt_sku + city_wh |
| 谁来转换 | 不需要转换 | PMS BE | PMS BE | PMS BE |

对于不同场景下的**历史销量统计的需求**:

| | organic | | campaign | |
|---|---|---|---|---|
| 场景 | 长尾 | 短尾 | SBD | DD |
| 颗粒度 | mt_sku + city_wh | mt_sku + city_wh | mp_sku | mp_sku |
| 统计字段 | purchasable_days/asp/adis | rolling features<br>lag features | purchasable_days/asp/adis | purchasable_days/asp/adis |
| 目的 | 作为长尾的预测结果<br>计算volatility= (future adis + campaign forecast /confirmation period) / final adis | 对历史数据做滚动计算<br>作为短尾商品预测的特征 | 作为计算uplift的基准，因为最后预测的y值是uplift | 计算上一次DD的l1级别的uplift系数的基准，作为下一次DD的参考 |

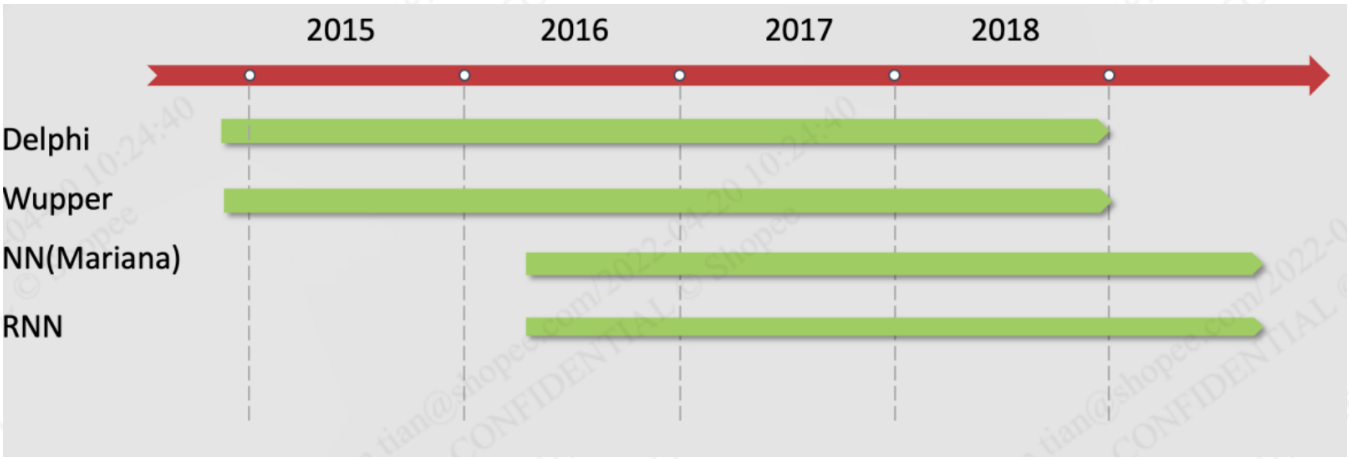
# 竞品方案调研

需求预测是供应链优化的开始， 重要性不言而喻， 对于各大电商平台都是一块重大的模块， 国内阿里， 京东， 美团，字节等重金投入，国外的亚马逊更是较早完成科技转型，成为行业标杆

## 主要竞品

### 亚马逊

亚马逊的预测模块开发周期和投资都较大，已经形成一套完整的预测框架和针对不同预测场景的优化方案:




- **Step1**(Delphi): 以**可解释性**和**满足业务基本**需求为主，主要focus在**非softline**的且**低预测难度（历史销量规律性明显）**的商品， 模型选型上以**时间序列模型**为主
- **Step2**(Wupper): 主要focus在**softline商品（历史销量规律不明显）**， 除了历史销量外，需要**更多外部因素**作为预测输入，以**机器学习算法模型**为主， 可解释性没有时间序列强
- **Step3**(Mariana(NN related)): 主要focus 在**新品**以及**较难预测的促销商品**，以**深度学习模型**为主，**可解释性较差**

**阿里供应链**


阿里数字供应链的技术路线于之类似， 每一代解决问题的侧重点和发力点不一样。 总结下来：

| 代数 | 业务范围 | 预测难度 | 稳定性 | 选择模型 | 可解释性 | 业务价值 | 其他 |
|---|---|---|---|---|---|---|---|
| 第一代 | 以非softline类的商品为主，算法可结果可复用于多数商品 | 低 | 高 | 时间序列模型（如Holt-winters,prophet） | 高 | 可解决大部分商品预测问题， 对于预测不准的商品也可产生兜底方案供运营参考 | 经久不衰，可解释性强于一切 |
| 第二代 | 以softline商品为主，处理外部因素导致的销量波动 | 高 | 中 | 机器学习模型（如LGB, RF等） | 中 | 对预测难度较大的商品提供更精确的预测 | 逐步使用开来，对于有一定解释性的模型来说已经普遍使用 |
| 第三代 | 处理新品场景以及更复杂的场景如直播，团购等 | 很高 | 低 | 深度学习模型（如DeepAR, N-beats等） | 低 | 对新品和复杂营销规则下的销量提供更精准的预测 | 这一代解决特殊场景下的问题 |

**注意：**

- 虽然号称迭代了三代，但是每一代都有适应的场景， 不存在某一代完全替代的情况， 因为真实业务场景中纷繁复杂，没有一个大而全的模型框架
- 每一代的演进都伴随着整个业务架构和成熟度的提升，例如业务人员对预测场景的熟悉，对系统的操作熟练，对S&OP流程的理解...
- 国内大的电商平台已经大部分都完成了前两代的部署，近几年都是在第三代的尝试和发力

## shopee的位置和方向

### 当前目标

当前shopee 的阶段仍比较初级，可以理解为还属于原始的线下计算（excel 数据操作）， 当前最大的两个目标是：

- **2022年度顺利完成线上化的目标，培养业务人员的系统使用粘性**
- 未来1-2年逐步搭建第一代预测框架，对于特定业务场景可选择性尝试第二代/第三代技术解决部分场景

### 需求预测 vs 需求计划

需求预测是如何产生预测值得过程， 包括人为拍脑袋，统计规则，统计模型， 机器学习，深度学习...

| | **需求预测** | **需求计划** |
|---|---|---|
| 功能 | 如何产生预测值得过程， 包括人为拍脑袋，统计规则，统计模型， 机器学习，深度学习... | S&OP中讨论较多的计划，需要多方合议进行修改和追踪，好的需求计划是公司供应链能力的强力体现 |
| 参与人员 | 销售人员， 需求计划员， 数据分析师 | 销售人员， 需求计划员， 数据分析师， 市场，财务，供应计划员 |
| 衡量指标 | 预测准确率， 偏差率 | 预测准确率， 偏差率， 达成率 |

虽然作为算法，我们主要参与到需求预测这样的过程中， 但是需求计划中协同流程对最终预测结果重要性也是不言而喻。 有时候为了解决业务问题， 我们可以换一种思路帮助业务人员解决预测不准的问题！

### 需求计划和供应计划

当前需求计划员和供应计划员有一人担当， 因此也没有传统的S&OP流程， 全部在该业务员的脑袋中. 主要原因包括：

- 当前retail的业务量在shopee中仍然不大，一人仍有余地完成工作
- 当前retail仍处于快速扩张阶段，对成本相对敏感性不高， 因此对准确率和偏差率没有硬性要求

但是这两点原因在retail高速成长的过程肯定会暴露问题， 因此会逐步走向规范化。 从我的经验来看， 需求计划系统和供应计划系统绝不是两个孤立的系统， 算法上的融通性也很多，供应计划部分retail也在

规划， 孙泽 会负责供应计划部分

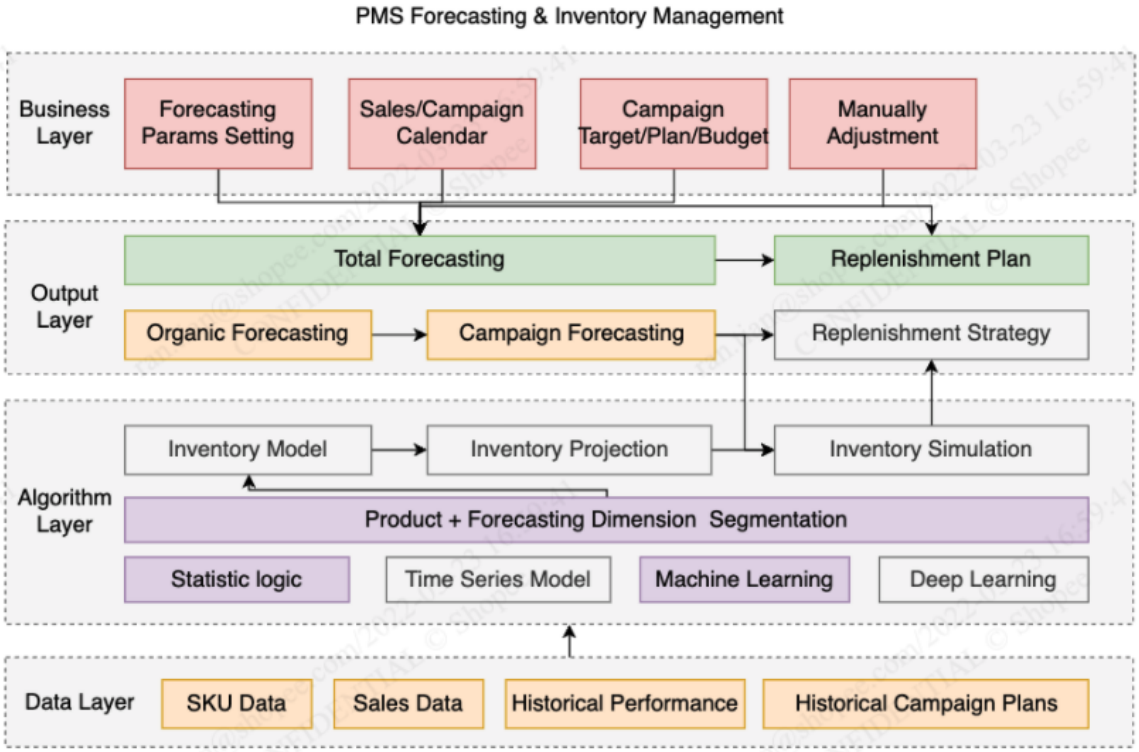
### 算法介入的挑战

算法团队从2022年加入retail的优化过程中， 参与的第一个项目就是该预测项目， 在我接触的几个月来， 目前比较大的挑战包括：

- 当前推进的预测系统的设计， 耦合性较高， 夹带着需求预测和补货逻辑， 尤其是数据流端的设计
- 营销端的手段和方式五花八门， 对预测模型是个大挑战
- 数据端的规范性不高， 有些数据的dependency都较高（甚至现在对于商品主数据中心都没有集中管理: vsku, mp, mt.. 每个系统的逻辑都不一样）

虽然挑战较多，但是机会也众多， 一旦打开了retail的优化之路， 那我们的价值会得到更充分的体现！

# Big picture




inventory 相关的模型考虑在下一步产品规划的范围内.

## 数据流模块设计

参考 数据需求PRD

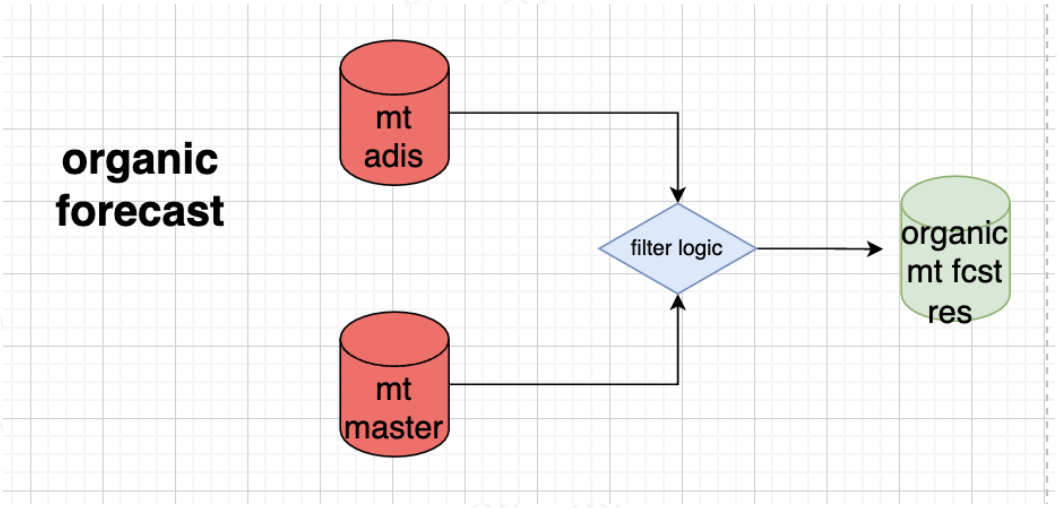
## 当前RegBD预测流程

下面图形的流程链接： https://app.diagrams.net/#G1rnKdKQ5P8ycHzk18_Wl8jhXWf3TOariX

## Organic

**Organic 预测老逻辑（0427之前， 已放弃）**

时间序列预测，给出预测horizon， 输入数据， 滚动方式等， 汇总到一个示意图上.




具体计算的逻辑参考： https://docs.google.com/spreadsheets/d/1OwbqrVzq7VHWeUlHG8GyZNunovmZOGRjZ3HdvAIDZmk/edit#gid=923256373

**计算最终的adis**:

- 计算mt_sku + city_warehouse的相应字段：
  - purchasable days
  - average selling price(asp)
  - price diff
  - average daily item sold(adis)
- 计算三组上述指标，
  - group A:  last 30 days exclude campaign
  - group B:  last 90 days exclude campaign
  - group C: last 90 days include campaign
- 根据price diff 和 purchasable days 筛选最终的adis作为未来的预测值

最终的adis结果就作为未来销量的参考值

**计算projected coverage**:

- 统计M1, M2, M3三个时间段
- 计算三个时间段的指标：
  - net_qty_sold
  - purchasable days
  - net_price
- 根据 net_price 和 last_buy_price 计算price diff
- 对比每个时间段的price diff 和 price band(配置得到)， 筛选符合的时间段
- 计算符合时间段的 project_coverage

| | 1. available in data mart? all sales include organic and campaign | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Calculation example:** | | | | | | | | | | |
| SKU ID | net_qty_sold_m1 | net_qty_sold_m2 | net_qty_sold_m3 | purchasable_da | purchasable_da | purchasable_da | net_price_m1 | net_price_m2 | net_price_m3 | buying price after tax |
| 123_456 | 30 | 40 | 100 | 28 | 28 | 28 | 24000 | 22000 | 10000 | 19000 |
| | | | | | | | | | | |
| SKU ID | price band m1 | price band m2 | price band m3 | | | | | | | |
| 123_456 | 26.32% | 15.79% | 47.37% | | | | | | | |
| | | | | | | | | | | |
| SKU ID | final adis for coverage calculation | | | | | | | | | |
| 123_456 | 1.25 | | | | | | | | | |

**Organic 更新逻辑（0427之后采用）**

参考模型说明文档： https://docs.google.com/document/d/12cEH5yOiZ3BdG375Y5eCLwQuRijLGs8vE8TxYcN8Jag/editc

参考模型代码：https://drive.google.com/file/d/118UHYrUpM-3nLa9FWu6PkQrdy6ctiDqu/view

- 长尾商品： adis
- 短尾商品： random forest 的模型， 主要包括：

# Organic Sales Forecasting: Revised organic forecast model is using Random Forest machine learning model with various key business data inputs to predict sales

**Context**

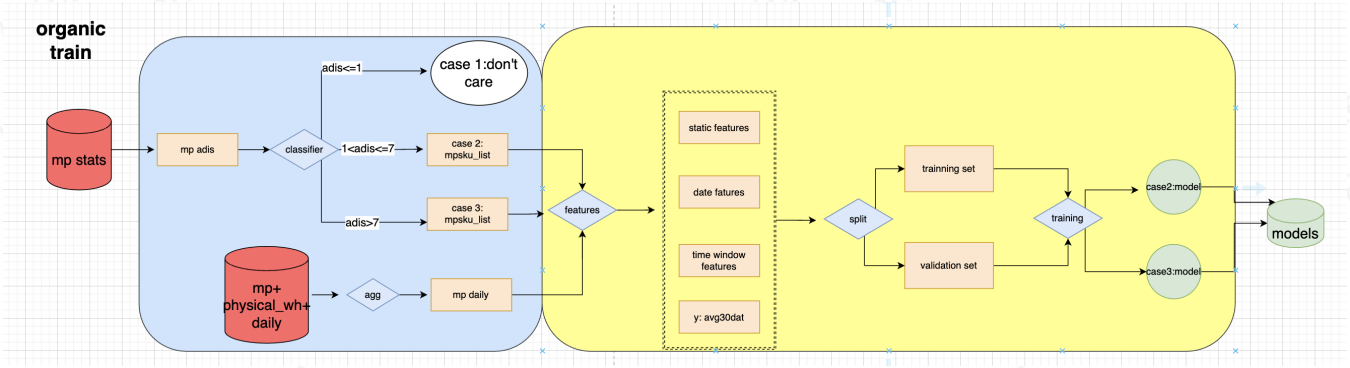
- Multiple types of model back tested, including Random forest, Lasso regression, support vector regression, Xgboost and LSTM (Long-short term memory)
- Random forest model with following features resulted in best accuracy
  - Constant model inputs to be applied for every region first
  - Customized features to be added if helpful for accuracy country by country
- Output table: regbd_general.shopee_bd_sbs_organic_forecasting_{cty}

| | Data type | Model inputs | Rationale |
|---|---|---|---|
| **Baseline model inputs** | **Historical sales** | L30D ADIS | Key feature that is the basis of the previous organic forecast model |
| | | L60D-L30D ADIS | Indicator to incorporate with L30D ADIS to show month on month trend |
| | | L30D Sales Volatility | Indicator to categorize SKU based on volatility using standard deviation |
| | | L7D Sales EMA* | Indicator for recent one week sales trend |
| | **Historical promotion** | L30D Avg Discount | Indicator for SKU's promotion level in L30D |
| | | L60D-30D Avg Discount | Indicator to incorporate with L30D Avg Discount to show month on month trend |
| | | L7D Avg Discount | Indicator to incorporate with L30D Avg Discount to show most recent week's discount trend |
| **Country specific model inputs** | **Historical sales** | Customized features | Model takes a hyper-localized approach and identifies key model inputs that are specific for each region and has insights into the country's unique sales patterns.<br><br>For example, for countries that have more long tail SKUs, a L90D ADIS sales trend may be included where as a country like ID where sales are generally faster, L15D or L7D ADIS may be included |

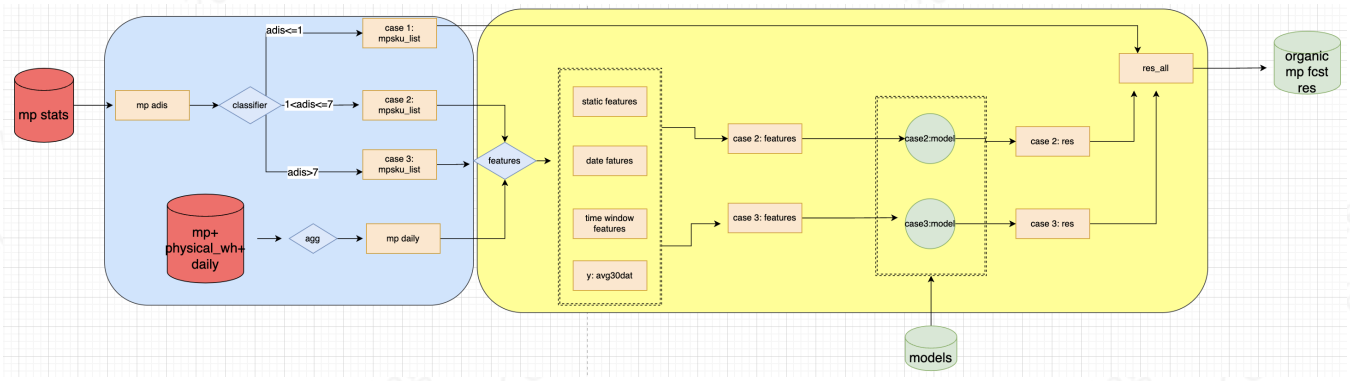
**EMA: exponential moving average, a classic time series model for demand forecasting which can reflect the most recent trend*



## Oraganic training



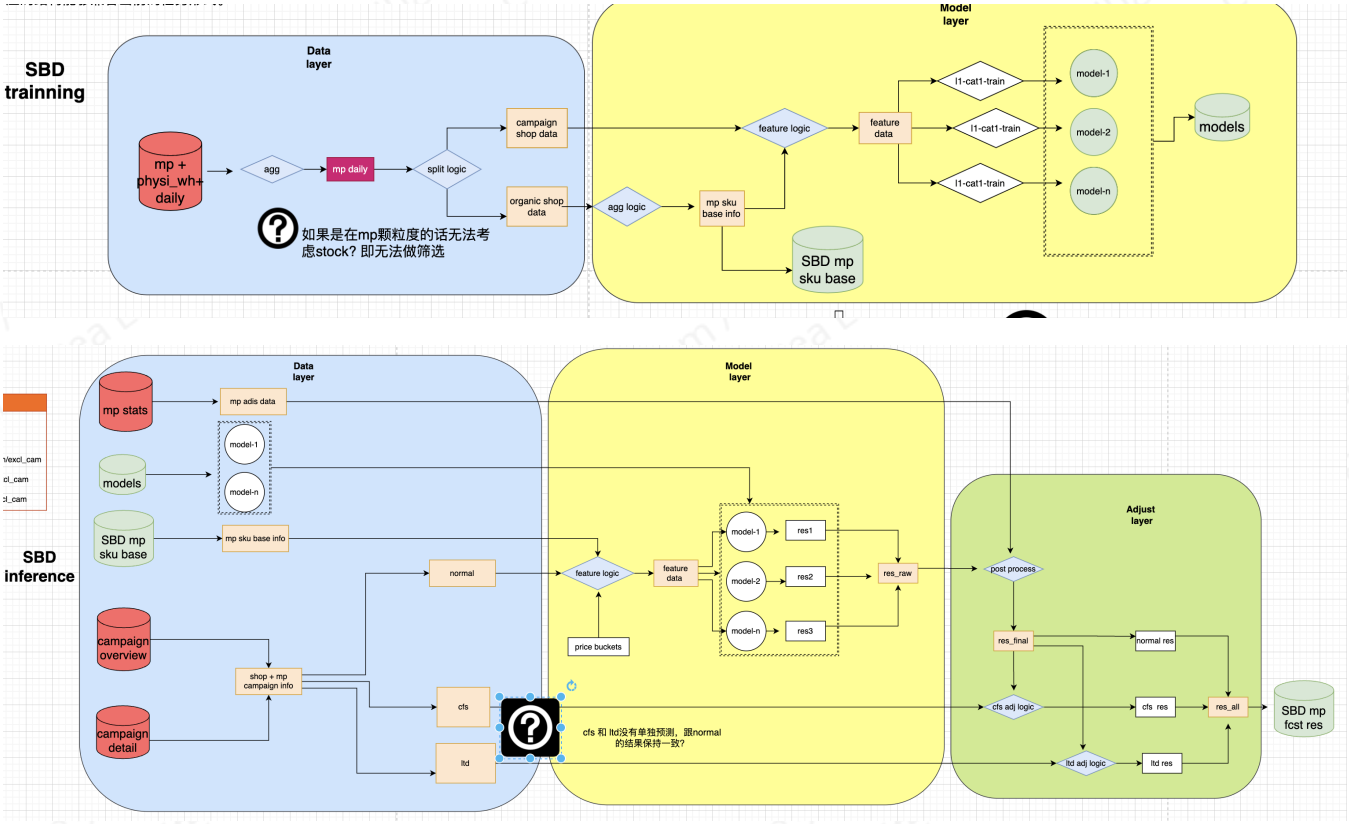

## Organic inference




# In-Organic (Campaign)

## SBD

当前regbd团队关于SBD的 methodology 文档




## DD

当前regbd团队关于DD的 DD methodology 文档

**Context**

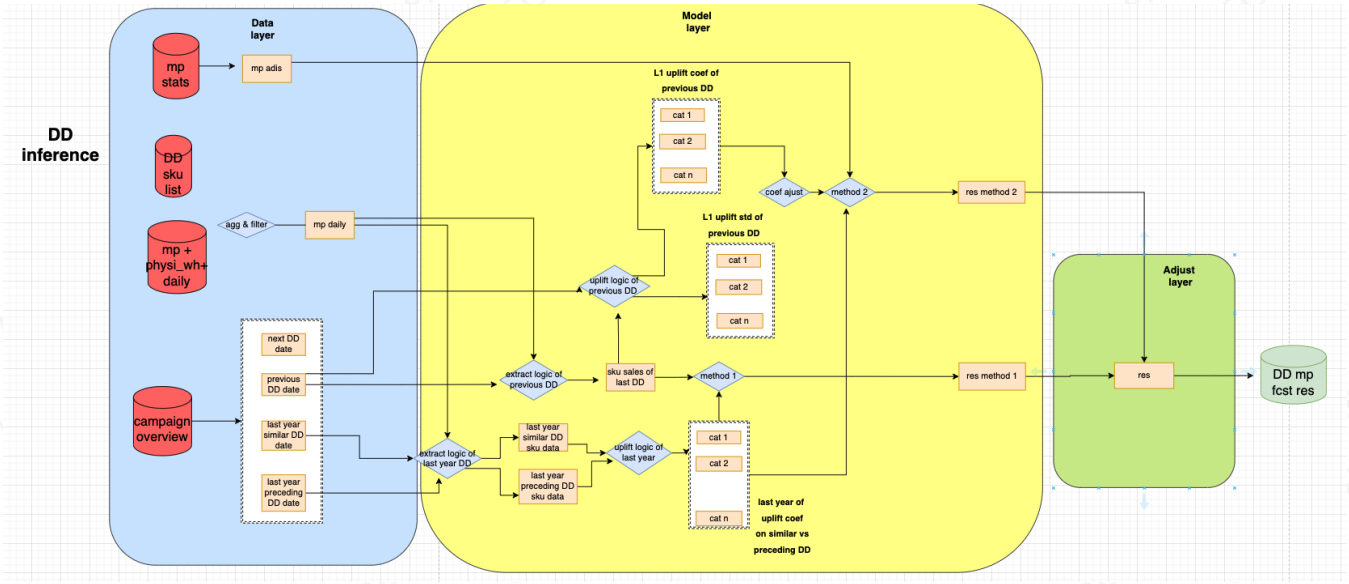
Reg SBS has collected countries' methodologies to forecast double-double (DD) sales per SKU. Reg SBS has developed a model based on various inputs considered by countries.

**Current Approach** [1]

(2022-04-04 is set as an example to illustrate)

| Approach | | Details |
|---|---|---|
| 1 | Basis | Historical sales of SKU on the last double-double |
| | Formula | Sales of SKU on last DD (Sales of SKU on 2022-03-03) × Last year's uplift of L1 on similar vs preceding DD [2] (Uplift of L1 on 2021-04-04 vs 2021-03-03) |
| | Findings | As last DD sales information is very specific for the SKU, relying on this input solely leads to overfitting model. There were swings in uplift on last DD and the same uplift trend of a SKU on last DD may not hold for next DD. |
| 2 | Basis | Historical uplift of L1 category on the last double-double |
| | Formula | [ Uplift of L1 last DD [3] (Uplift of L1 on 2021-03-03) + Uplift standard deviation of L1 last DD [3] (Uplift std dev of L1 on 2021-03-03) ] × L30D SKU organic ADIS [4] (L30D SKU organic ADIS) × Last year's uplift of L1 on similar vs preceding DD [2] (Uplift of L1 on 2021-04-04 vs 2021-03-03) |
| | Findings | Standard deviation of L1 uplift on last DD is added to cover the sales of SKUs with higher uplift on next DD. As uplift on DD is evaluated on L1 level, the model provides more generic information about the uplift on next DD. As such, using this input alone leads to underfitting model. |
| 3 (Used in the model) | Formula | Average of the approach 1 and 2 |
| | Findings | Averaging the 2 approaches above gives the best result since it prevents the model from being both overfitting & underfitting |

Note: 1. The SKUs evaluated in the model are on a listed SKU level on SBS Outright.
2. Last year's uplift of L1 on similar vs preceding DD is evaluated on Shopee platform SKU universe.
3. Uplift and uplift standard deviation of L1 on last DD is evaluated on SBS Outright SKU universe vs L30D organic ADIS below.
4. L30D organic ADIS is calculated 45 days before the next double-double to provide sufficient time for countries for replenishment planning.



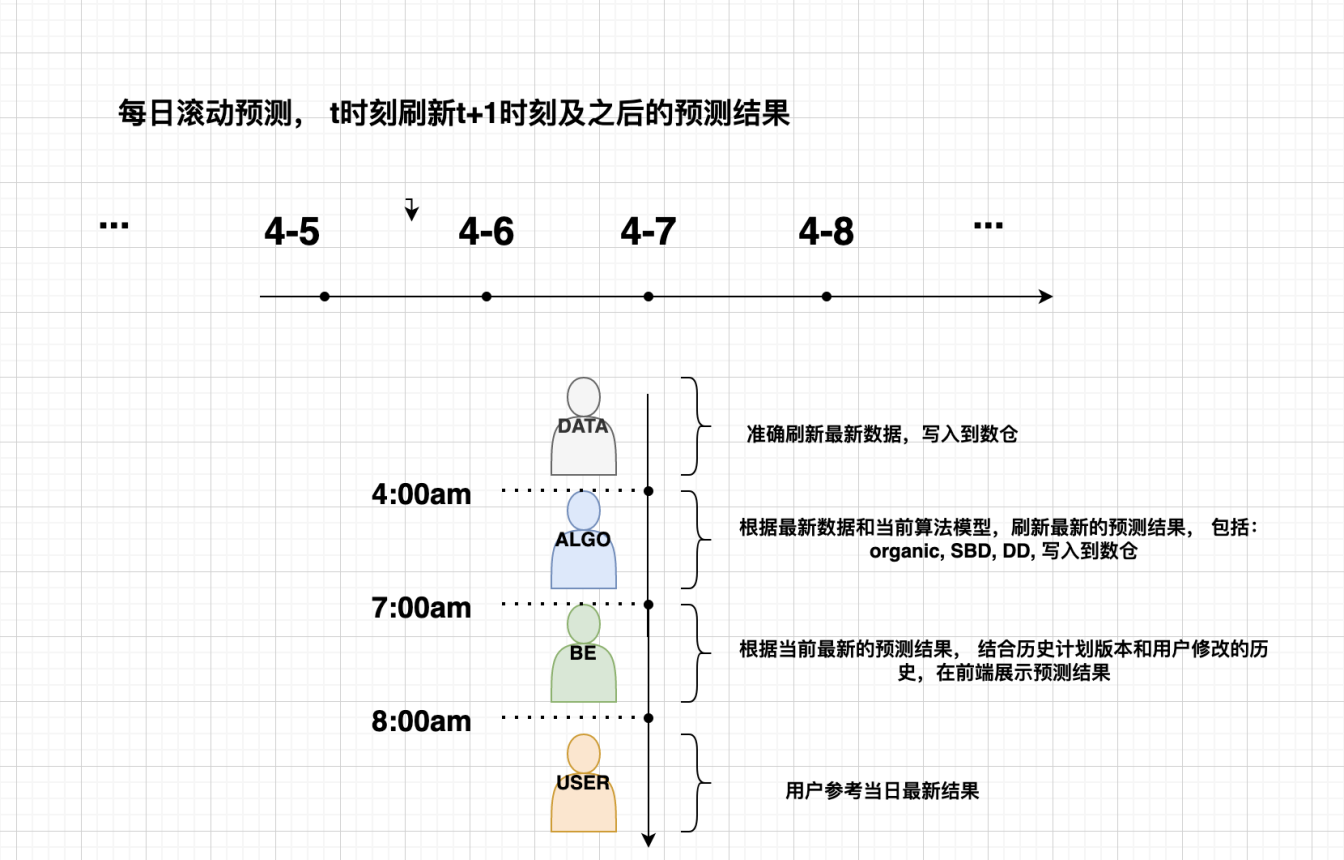

# 算法运行

## 业务需求

当前业务的需求主要包括：

- 每日需要在工作前（8:00am前）看到最新的预测结果， 其他时间
- 不会有触发式的使用需求，统一采用 t+1的滚动预测模式（当日算法刷新数据后，今日不会根据允许用户主动触发算法重新计算）
- 时效性要求不高，在当地时间凌晨离线计算完成即可， 无线上直接预测的场景

下图以一日为例，介绍了各个团队的任务和时间线：




## 算法触发方式

根据上述的业务需求， 算法和各大团队的交互相对简单，主要**集中在数据表的准备和输出**， 因此算法的调度执行由算法团队控制， 主要采用**定时任务调度**以下内容：

- job1: 每日定时预测organic的预测结果
- job2: 每日定时预测SBD的预测结果
- job3:  每日定时预测DD的预测结果
- job4[optional]: 每个月定时更新SBD的模型
- job5[optional] 每日定时更新mp的adis

对于定时任务调度框架， 优先采用**`data suite 上的scheduler`**模块， 如果仍不支持 python/pyspark 的话采取**`airflow`**过渡使用

## 日志收集

to-be decided

# 算法模块设计

## 模块介绍

为了解耦模块，为后续迭代提供基础，算法模块主要分成四大模块：

**模块整体内容：**

| 模块 | 作用 | 输入 | 输出 | 其他 |
| --- | --- | --- | --- | --- |
| **`data layer`**<br><br>数据层 | 承接输入数据load以及清洗相关数据的模块 | ▪ 原始的DB数据，<br>▪ 配置数据<br>▪ ... | 建模需要的数据模板 | |
| **`model layer`** | 针对已有的数据模板搭建整个建模流程， 注意此模型层可能包含着业务的流程和规则，是一个广义的模型概念 | data layer层的建模数据模板 | 模型初步的预测结果 | |

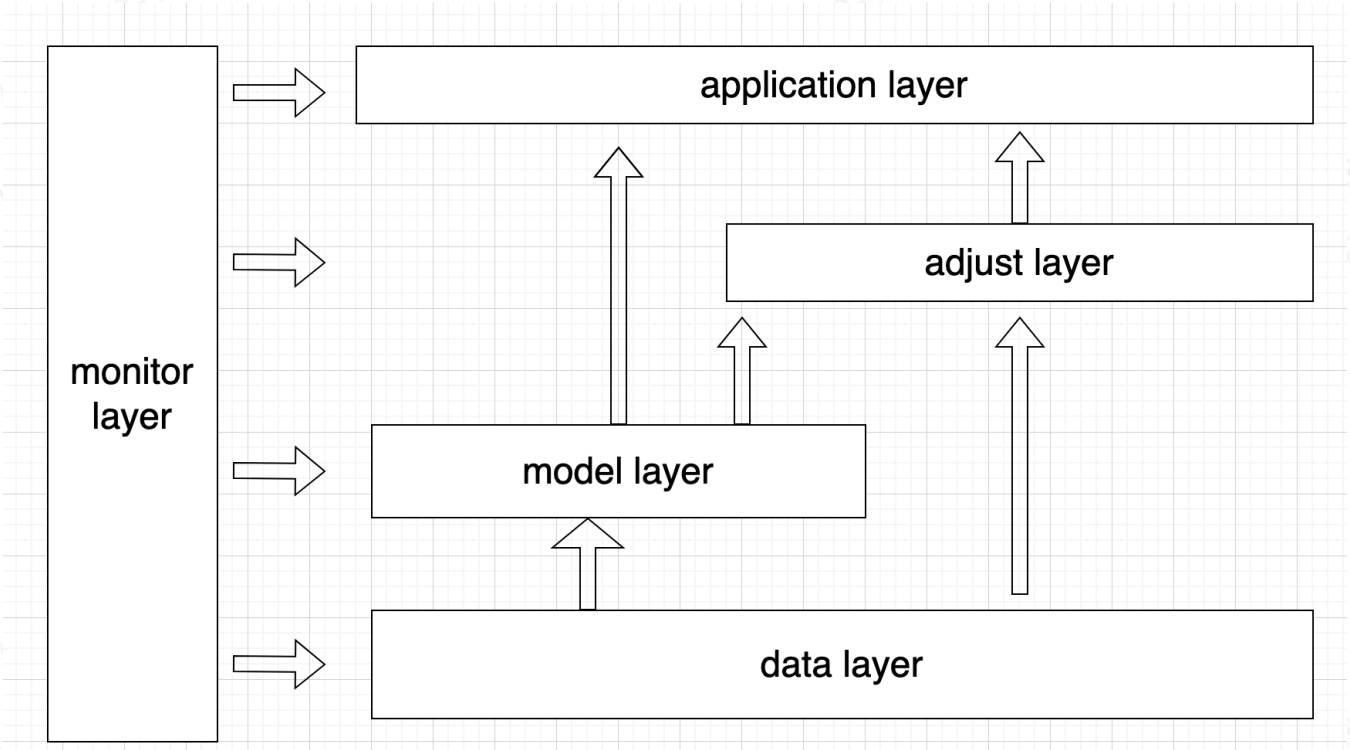
| | | | | |
|---|---|---|---|---|
| 模型层 | | | | |
| **adjust layer**<br><br>调整层 | 针对模型层的输出结果进行逻辑调整/选择等 | ▪ data layer层的一些汇总数据<br>▪ 模型初步的预测结果 | 模型调整后的预测结果 | |
| **monitor layer**<br><br>监控层 | 针对输出的结果，进行持续的数据质量监控， 预警等 | 各个layer | ~ | 主要模块包括：<br>• 数据质量监控<br>• 模型错误处理方案<br>• 准确率监控<br>• 稳定性监控 |

注意：

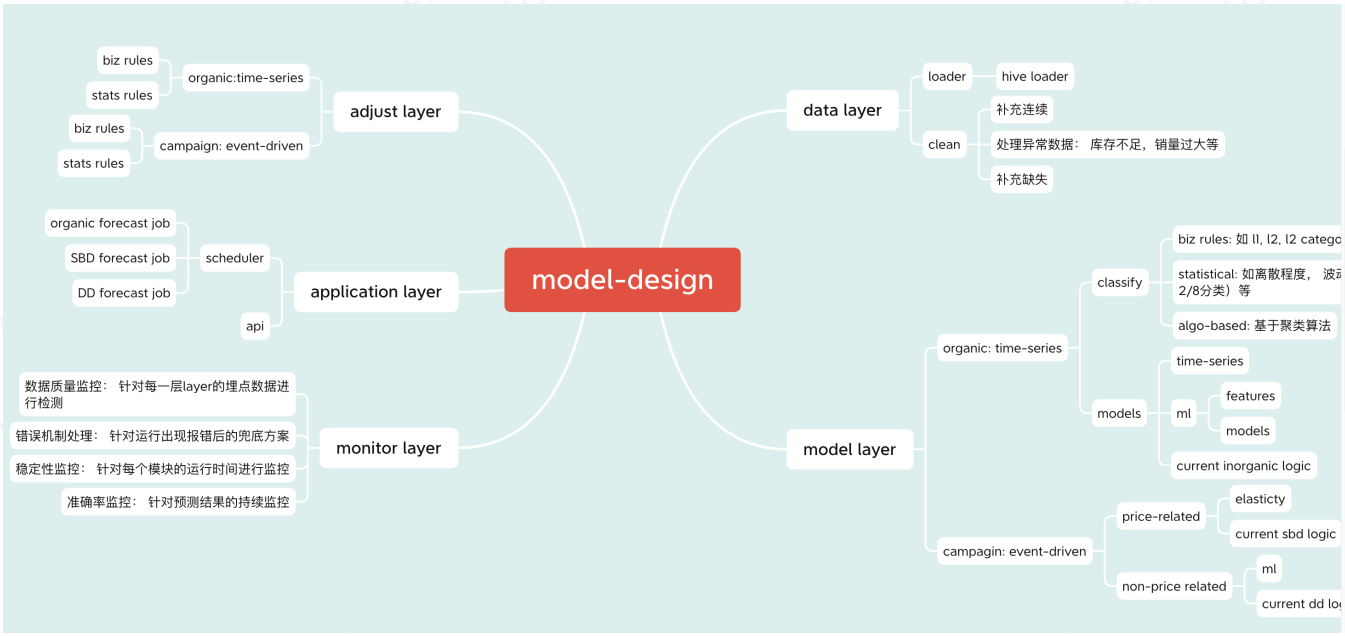
- 上述Layer分层是粗颗粒的分层，旨在将比较大的功能解耦， 减少重复代码量
- 对于比较细分的模型流程（例如preprocess, features, model_train/test, post_process等）， 在此框架不做严格细分，因为**此类预测模型和业务规则/需求是严格绑定**，不做过多抽象

### 设计图

各大模块的数据见交互情况：




**对于每个模块包含的内容如下：**




**代码框架结构如下：**

结合上述分层的设计， 主要的代码框架参考 机器学习代码框架， 但根据需求预测的业务需求，做了一定的调整， 主要包括：

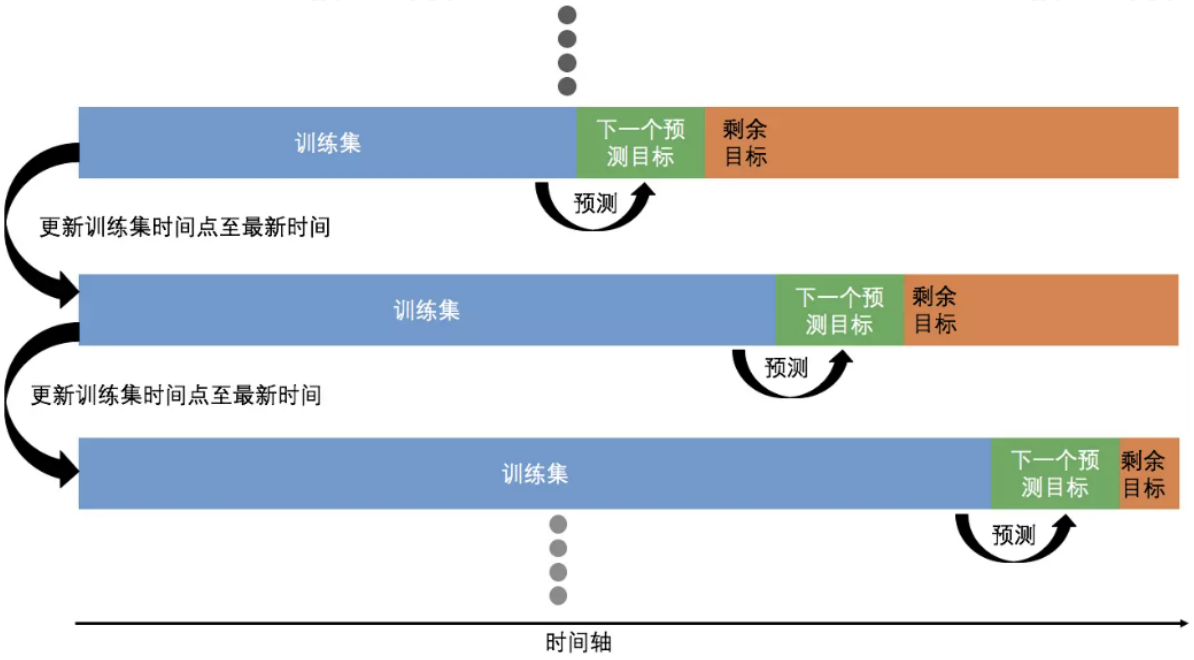
- [x] 结合电商需求预测的业务特点，对预测场景做出大的分类，包括时序相关的预测场景（organic）以及事件型相关的预测场景(campaign)
- [x] 需求预测的模型不仅仅包括机器学习，也包含一些规则和统计模型， 其和机器学习的框架有一定区别， 此框架不在每类模型做具体的规定，仅规范各类模型统一的接口(fit, predict)
- [x] 由于预测的场景较复杂， 往往会对需求进行分类，针对不同分类场景做适配性的定制， 因此对于全量的预测集合，不是单靠一类模型解决

主要结构如下：

参考 https://git.garena.com/shopee/bg-logistics/algo/retail/retail-forecast/-/tree/dev/ 的目录结构

**历史数据回测方式**

对于time-series的场景， 历史数据的回测方案主要是： time rolling cross validation, 图示如下：




对于event-driven(促销)的方案， 根据历史促销日期做相关的划分即可

## Regbd预测模型对应数据模型(0427日之后)

### 数据流程案例说明

以下是关于三类job对应的数据流案例：https://docs.google.com/spreadsheets/d/12u9BNKwgmbJ7glymxzbJCGut8MKFDyRJou2dWhsLCS4/edit#gid=0

### 数据环境和规范

#### 环境介绍

当前主要是三套环境, 与data保持一致， 但是算法的schema是在**sci_retail**内：

- dev：开发环境
- staging: 测试环境
- live: 实际线上环境

**注意：**

- data的环境和业务系统的环境不一定1对1保持一致，如data 的staging环境既支持pms端的test环境，又支持pms端的uat环境
- schema在**sci_retail**内, 见 https://datasuite.shopee.io/ram/project-management?code=sci_retail&tab=1
- 不同环境也对应不同的计算资源，一般除了live是实际的sci-prod资源，其余都是sci-dev资源

#### 命名规范

整体命名规范参考data的文档：https://docs.google.com/document/d/1PBNlBokUQbCKlRi30ssQAAILsCQsf7xCNZ8-xa16Kps/edit#heading=h.yshgiq5wshj2 ，需要注意的有以下几点：

主要遵循以下命名方式：**{env}_{schema}.{layer}_{app_category}_{job_type}_{description}_{type}_{region}**

- **env**: 环境，主要是dev,staging
- **schema**: 当前是sci_retail
- **layer**: 数据层，如 dim, ads
- **app_category**: retail业务下的应用范围，如 demand, supply, pricing... **当前预测均属于demand**
- **job_type**： 如organic_train, organic_infer, sbd_train, sbd_infer, dd_infer, all(代表所有的job都适用)
- **description** : 主要是表的描述，包括但不限于：
  - mpsku/mtsku
  - 1d/nd
  - ...
- **type**: 表的类型，如

    - tmp: 临时表
    - record: 记录表，记录表会把所有的运行结果都记录下来，
    - res: 结果表， 结果表只存最新的运行结果
    - config: 配置表
    - ...
- **region**: 区域， 如id，vn等

### 算法相关数据分类

此项目所有的数据设计见： https://docs.google.com/spreadsheets/d/1NDGqg_8q665bVrAsNkILZf9vvZpcXtd9SCXesgKGVms/edit#gid=499880166

下面是具体到算法需求相关的数据： **注意表的最终后缀 _{region} 在以下说明中忽略**

#### 输入数据

| job_type | table | instructions | notices |
|---|---|---|---|
| all | sci_retail.ads_demand_mode_config | 模型的配置数据 | |
| organic_train & organic_inference | sbs_mart.ads_sales_fcst_historical_sales_mpsku_1d | 历史mpsku销量表 | 1: 开始时间是2021-07-12<br>2：每日的grass_date 会更新前28天的net_sales数据 |
| | sbs_mart.ads_sales_fcst_historical_sales_shop_1d | 历史shop销量表 | 暂无 |
| | sbs_mart.ads_sales_fcst_historical_summary_mpsku_1d | 当前sku的历史时间窗数据的统计 | 1： 两个字段： mp_purchasable_days_excl_campaign_l30d 和 mp_net_purchasable_adis_excl_campaign_l30d<br>2： 选择grass_date 最大的数据作为最新的adis数据 |
| | sbs_mart.dim_mpsku_fcst_ext | 当前sku的属性以及预测范围 | 1: 选择grass_date 最大的数据作为最新的预测范围 |
| sbd_inference | sbs_mart.ads_sales_fcst_historical_sales_mpsku_1d | 历史mpsku销量表 | |
| | sbs_mart.ads_sales_fcst_historical_sales_shop_1d | 历史shop销量表 | |
| | sbs_mart.dim_mpsku_campaign_ext | sbd活动下待预测的sku范围 | 1： 跟 dim_campaign_calendar_shop 条件一致<br>2： 通过 campaign_id + campaign_date + shop_id 进行join |
| | sbs_mart.dim_mpsku_fcst_ext | sku的属性 | |
| | sbs_mart.dim_campaign_calendar_shop | 促销日历的overview数据 | 1： campaign_type =1<br>2: grass_date 选最新<br>3：is_fcst_ready = 1<br>4： camapaign_date > 今日 |
| | sci_retail.ads_demand_sbd_train_mpsku_base_info_record | 【算法自己生成】sbd下mpsku的base信息 | |
| sbd train | sbs_mart.ads_sales_fcst_historical_sales_mpsku_1d | 历史mpsku销量表 | |
| | sbs_mart.ads_sales_fcst_historical_sales_shop_1d | 历史shop销量表 | 1： 开始时间是2021-07-12<br>2： 使用的字段： net_order, net_gmv_usd, total_outright_mpsku_count |
| dd_inference | sbs_mart.ads_sales_fcst_historical_sales_mpsku_1d | 历史mpsku销量表 | |
| | sbs_mart.ads_sales_fcst_historical_sales_shop_1d | 历史shop销量表 | |

| | | | |
|---|---|---|---|
| | sbs_mart.dim_mpsku_campaign_ext<br>or<br>sbs_mart.<br>dim_campaign_calendar_mpsku | dd活动下待预测的sku范围 | 1： 根据campaign_id, campaign_date, shop_id 进行关联 |
| | sbs_mart.<br>dim_campaign_calendar_shop | 促销日历的overview数据 | 1： campaign_type =4<br>2: grass_date 选最新<br>3：is_fcst_ready = 1<br>4： camapaign_date > 今日 |
| | ads_sales_fcst_historical_summary_mpsku_1d | 当前sku的历史时间窗数据的统计 | |

### 中间数据

| job_type | table | instructions | notice |
|---|---|---|---|
| all | sci_retail.ads_demand_all_model_info_record | 模型信息表 | |
| | | | |
| organic | sci_retail.ads_demand_organic_train_feature_tabular_record | organic train时记录下的特征表， 跟train的周期保持一致 | |
| | sci_retail.ads_demand_organic_infer_feature_tabular_record | organic inference时记录下的特征表, 每天都会生成 | |
| | | | |
| sbd | sci_retail.<br>ads_demand_campagin_sbd_train_feature_tabular_record | sbd train时记录下的特征表， 跟train的周期保持一致 | |
| | sci_retail.ads_demand_campagin_sbd_train_mpsku_base_info_record | sbd train时记录的mpsku的信息表， 跟train的周期保持一致 | |
| | sci_retail.<br>ads_demand_campagin_sbd_infer_feature_tabular_record | sbd inference时记录下的特征表， 只有当未来calendar有预测任务时才会记录 | |
| | | | |
| dd | sci_retail.ads_demand_campagin_dd_infer_stats_record | dd inference时预测结果大宽表， 包含预测过程中各种系数和结果 | |
| | | | |
| | | | |

### 输出数据

对应的**record**表的关系为：

| job type | sci_retail | sbs_mart |
|---|---|---|
| campaign | sci_retail.ads_demand_campaign_fcst_mpsku_record | \ |
| organic | sci_retail.ads_demand_organic_fcst_mpsku_record | \ |

对应的**result**表的关系为：

| job type | sci_retail | sbs_mart |
|---|---|---|
| campaign | sci_retail.ads_demand_campaign_fcst_mpsku_res | sbs_mart.ads_algo_campaign_fcst_result_mpsku_1d |
| organic | sci_retail.ads_demand_organic_fcst_mpsku_res | sbs_mart.ads_algo_organic_fcst_result_mpsku_1d |

## Regbd预测模型对应数据介绍(0427日之前，目前已放弃)

以下模块介绍是针对当前BI模型在当前框架下的介绍：

**注意**：

1. 当前设计时采用的思路是基于历史大宽表进行处理
2. 下面提到的字段均是必须的字段，对于一些可选的字段没有列出

**其中一些字段的缩写如下：**

- excl: exclude
- incl: include
- wh: warehouse
- asp: average selling price
- adis: average daily item sold

## Organic预测

<table>
<tr><th>模块</th><th>模块内容</th><th>input</th><th>output</th></tr>
<tr><td>data</td><td>loader</td><td>
mt-wh 颗粒度下的adis切片：
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mtsku</td><td></td><td></td></tr>
<tr><td>city_wh</td><td></td><td></td></tr>
<tr><td>grass_region</td><td></td><td></td></tr>
<tr><td>grass_date</td><td></td><td></td></tr>
<tr><td>last_30days_excl_campagin_ + purchasable_days/asp/price_diff/adis</td><td>过去30天不包含促销的purchable_days/asp/price_diff/adis</td><td></td></tr>
<tr><td>last_90days_excl_campagin_ + purchasable_days/asp/price_diff/adis</td><td>过去90天不包含促销的purchable_days/asp/price_diff/adis</td><td></td></tr>
<tr><td>last_90days_incl_campagin_ + purchasable_days/asp/price_diff/adis</td><td>过去90天包含促销的purchable_days/asp/price_diff/adis</td><td></td></tr>
<tr><td>m1_ + net_qty_sold/purchasable_days/net_price</td><td>过去第一个月的net_qty_sold/purchasable_days/net_price</td><td></td></tr>
<tr><td>m2_ + net_qty_sold/purchasable_days/net_price</td><td>过去第二个月的net_qty_sold/purchasable_days/net_price</td><td></td></tr>
<tr><td>m3_ + net_qty_sold/purchasable_days/net_price</td><td>过去第三个月的net_qty_sold/purchasable_days/net_price</td><td></td></tr>
<tr><td>last_buy_price</td><td>该mt_sku + city_wh下的最新购买价格</td><td></td></tr>
</table>
price_band的配置数据：
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>grass_region</td><td></td><td></td></tr>
<tr><td>grass_date</td><td></td><td></td></tr>
<tr><td>price_band</td><td>用户配置的price_band数据</td><td></td></tr>
</table>
</td><td>
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mtsku</td><td></td><td></td></tr>
<tr><td>city_wh</td><td></td><td></td></tr>
<tr><td>grass_region</td><td></td><td></td></tr>
<tr><td>grass_date</td><td></td><td></td></tr>
<tr><td>last_30days_excl_campagin_ + purchasable_days/asp/price_diff/adis</td><td>过去30天不包含促销的purchable_days/asp/price_diff/adis</td><td></td></tr>
<tr><td>last_90days_excl_campagin_ + purchasable_days/asp/price_diff/adis</td><td>过去90天不包含促销的purchable_days/asp/price_diff/adis</td><td></td></tr>
<tr><td>last_90days_incl_campagin_ + purchasable_days/asp/price_diff/adis</td><td>过去90天包含促销的purchable_days/asp/price_diff/adis</td><td></td></tr>
<tr><td>m1_ + net_qty_sold/purchasable_days/net_price</td><td>过去第一个月的net_qty_sold/purchasable_days/net_price</td><td></td></tr>
<tr><td>m2_ + net_qty_sold/purchasable_days/net_price</td><td>过去第二个月的net_qty_sold/purchasable_days/net_price</td><td></td></tr>
<tr><td>m3_ + net_qty_sold/purchasable_days/net_price</td><td>过去第三个月的net_qty_sold/purchasable_days/net_price</td><td></td></tr>
<tr><td>last_buy_price</td><td>该mt_sku + city_wh下的最新购买价格</td><td></td></tr>
<tr><td>price_band</td><td>用户配置的price_band数据</td><td></td></tr>
</table>
</td></tr>
<tr><td>model</td><td>organic/adis_rule</td><td>
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mtsku</td><td></td><td></td></tr>
<tr><td>city_wh</td><td></td><td></td></tr>
<tr><td>grass_region</td><td></td><td></td></tr>
<tr><td>grass_date</td><td></td><td></td></tr>
</table>
</td><td>
mt + wh 级别的预测结果：
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mtsku</td><td></td><td></td></tr>
<tr><td>city_wh</td><td></td><td></td></tr>
<tr><td>grass_region</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

| | | | | | | |
|---|---|---|---|---|---|---|
| | | last_30days_excl_campagin_ + purchasable_days/asp/price_diff/adis | 过去30天不包含促销的purchable_days/asp/price_diff/adis | | grass_date | 数据刷新的日期 |
| | | last_90days_excl_campagin_ + purchasable_days/asp/price_diff/adis | 过去90天不包含促销的purchable_days/asp/price_diff/adis | | fcst_date | 待预测日的日期 |
| | | last_90days_incl_campagin_ + purchasable_days/asp/price_diff/adis | 过去90天包含促销的purchable_days/asp/price_diff/adis | | organic_fcst_adis | |
| | | m1_ + net_qty_sold/purchasable_days/net_price | 过去第一个月的net_qty_sold/purchasable_days/net_price | | organic_fcst_asp | |
| | | m2_ + net_qty_sold/purchasable_days/net_price | 过去第二个月的net_qty_sold/purchasable_days/net_price | | organic_fcst_price_diff | |
| | | m3_ + net_qty_sold/purchasable_days/net_price | 过去第三个月的net_qty_sold/purchasable_days/net_price | | organic_fcst_purchasable_days | |
| | | last_buy_price | 该mt_sku + city_wh下的最新购买价格 | | organic_projected_coverage | |
| | | price_band | 用户配置的price_band数据 | | | |
| adjust | ~ | ~ | | | ~ | |

## DD 促销预测

问题：

- mp 颗粒度的adis计算是否能够data 一起做了？因为逻辑和mt颗粒度的是一致的！ 历史数据的话，是否也可以一起算了？
- 如果是算法计算adis 的话，那是否也是一个job分别计算，因为sbd和dd都用到了其结果
  - 对于历史上的adis数据， 如果算法一起做了，可以一次性线下更新

| 模块 | 子模块 | 模块内容 | input | output |
|---|---|---|---|---|
| data | loader | 读取数据，对齐数据 | ▪ campagin_overview数据<br>▪ mp_历史销量数据 | **各大DD对应的日期：**<br>▪ fcst_DD_date: 待预测DD日期<br>▪ last_year_similar_DD_date： 去年同期的DD日期<br>▪ last_year_preceding_DD_date： 去年同期的上一期的DD日期<br>▪ previous_DD_date: 当下上一期的DD日期<br>**mp 历史销量数据** |
| | cleaner/mp_adis | 计算mp颗粒度下的adis | 待定 | 待定 |
| | cleaner/extract_sku_data_last_year | 提取**去年同期**的sku销量数据 | **输入参数：** last_year_similar_DD_date<br>**mp颗粒度下的历史销量数据：**<br>字段 / 描述 / 说明<br>mp_sku<br>l1_category<br>shop_id<br>date<br>is_purchasable：该mp_sku当日是否可销售<br>sku_net_gross_amount：当日总销量 | 选择last_year_similar_DD_date日期下的所有销量数据, 字段同输入 |

<table>
<tr><td></td><td></td><td></td><td><table>
<tr><td>sku_net_gross_amount_ltd</td><td>当日ltd销量</td><td></td></tr>
<tr><td>sku_net_gross_amount_cfs</td><td>当日cfs销量</td><td></td></tr>
<tr><td>sku_net_gross_amount_campaign</td><td>当日其他campagin类型的销量</td><td></td></tr>
</table></td><td></td></tr>
<tr><td></td><td>cleaner /extract_sku_data_last_year_preceding</td><td>提取去年同期的上一期sku销量数据</td><td>同上， 日期为 last_year_preceding_DD_date</td><td>同上,日期为 last_year_preceding_DD_date</td></tr>
<tr><td></td><td>cleaner /extract_sku_data_previous_DD</td><td>提取上期的sku销量数据</td><td>同上,日期为 previous_DD_date</td><td>同上,日期为 previous_DD_date</td></tr>
<tr><td>model</td><td>campaign/non_price /method1</td><td>当前的模型逻辑1</td><td>cleaner/extract_sku_data_last_year<br><br>cleaner /extract_sku_data_last_year_preceding<br><br>cleaner/extract_sku_data_previous_DD</td><td><table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mp_sku</td><td></td><td></td></tr>
<tr><td>l1_category</td><td></td><td></td></tr>
<tr><td>last_year_coef</td><td>去年同期相比于去年同期上一期的l1级别的提升系数</td><td>颗粒度是l1,即同一个l1级别的系数一样</td></tr>
<tr><td>amount_previous_DD_date</td><td>当前上期的销量</td><td>颗粒度是mp_sku</td></tr>
<tr><td>fcst_res_method1</td><td>method1的预测结果</td><td></td></tr>
</table></td></tr>
<tr><td></td><td>campaign/non_price /method2</td><td>当前的模型逻辑2</td><td>mp颗粒度下的adis<br><br>cleaner/extract_sku_data_previous_DD</td><td><table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mp_sku</td><td></td><td></td></tr>
<tr><td>l1_category</td><td></td><td></td></tr>
<tr><td>last_year_coef</td><td>去年同期相比于去年同期上一期的l1级别的提升系数</td><td>颗粒度是l1,即同一个l1级别的系数一样</td></tr>
<tr><td>previous_uplift_coef</td><td>上期DD下l1级别的提升系数</td><td>颗粒度是l1</td></tr>
<tr><td>previous_uplift_coef_std</td><td>上期DD下l1级别的提升系数标准差</td><td>颗粒度是l1</td></tr>
<tr><td>previous_uplift_coef_raw</td><td>=previous_uplift_coef+<br><br>previous_uplift_coef_std</td><td>颗粒度是l1</td></tr>
<tr><td>previous_uplift_coef_adjust</td><td>根据规则对previous_uplift_coef_raw进行调整</td><td>颗粒度是l1</td></tr>
<tr><td>fcst_res_method2</td><td>method2的预测结果</td><td></td></tr>
</table></td></tr>
<tr><td></td><td>campaign/non_price/</td><td>合并method1和method2的结果</td><td>method1的输出<br><br>method2的输出</td><td><table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mp_sku</td><td></td><td></td></tr>
<tr><td>l1_category</td><td></td><td></td></tr>
<tr><td>fcst_date</td><td>待预测的DD的日期</td><td>注意和预测的时间不一样！</td></tr>
<tr><td>fcst_res_method1</td><td></td><td></td></tr>
<tr><td>fcst_res_method2</td><td></td><td></td></tr>
<tr><td>fcst_res</td><td>最终融合的预测结果</td><td></td></tr>
</table></td></tr>
<tr><td>adjust</td><td>暂无</td><td>~</td><td>~</td><td>~</td></tr>
</table>

## SBD 促销预测

问题：

- SBD train 的频率不能是每天一次吧？ 没个月定时一次
- SBD train时保存两个离线数据
  - l1级别的预测模型
  - mp_sku 级别的基础汇总数据（用于后续inference阶段时构造特征）
- 对于shop_gmv 和 shop_orders 这两个指标，训练数据使用的是真实值，测试数据使用的是用户输入的预期值
- cfs和ltd的业务场景之后如何预测？
- 该SBD 模型不仅仅简单地给出推荐，而是业务需要调整和尝试（比如sku_discount_bin, shop_gmv_target, shop_order_target等）

<table>
<tr><th>模块</th><th>子模块</th><th>模块内容</th><th>input</th><th>output</th></tr>
<tr><td>data</td><td>loader</td><td>读取数据，对齐数据</td><td><ul><li>campaign overview</li><li>campaign detail</li></ul></td><td>mp_sku级别的促销数据：
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>campaign_id</td><td></td><td></td></tr>
<tr><td>campaign_type</td><td>campaign_type, selective: Super Brand Day, Brand of the Day, Double Double, Pay Day, Shop-wide Others, Platform-wide Others, Others(may not included in the current phase)</td><td></td></tr>
<tr><td>campaign_level</td><td>'shop-wide' when campaign_type in (Super Brand Day, Brand of the Day, Shop-wide Others),<br><br>'platform-wide' when campaign_type in (Double Double, Pay Day, Platform-wide Others)</td><td></td></tr>
<tr><td>shop_id</td><td></td><td></td></tr>
<tr><td>campaign_date</td><td></td><td></td></tr>
<tr><td>campaign_stage</td><td>campaign stage, when campaign_type in (Super Brand Day, Brand of the Day) then pass this value; or else N.A.</td><td></td></tr>
<tr><td>gross_order_target</td><td>campaign_shop_daily level gross order target; when campaign_level = platform-wide, this means the campaign_daily level gross order target</td><td></td></tr>
<tr><td>gmv_target</td><td>campaign_shop_daily level gmv target; when campaign_level = platform-wide, this means the campaign_daily level gmv target</td><td></td></tr>
<tr><td>mp_sku</td><td></td><td></td></tr>
<tr><td>l1_category</td><td></td><td></td></tr>
<tr><td>sku_original_selling_price</td><td></td><td></td></tr>
</table></td></tr>
<tr><td></td><td>cleaner /mp_adis</td><td>计算mp颗粒度下的adis</td><td>待定</td><td>待定</td></tr>
<tr><td></td><td>cleaner /sbd_clean</td><td>对原始的销量数据进行一定程度的清洗（这层要抽象出来因为后续的优化也是基于清洗后的数据）：<br><ul><li>yesterday_sku_stock>0&sku_stock > 0</li><li>is_purchasable == 1</li></ul></td><td>mp_sku历史销量表</td><td>清洗后mp_sku级别的历史销量表：
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mp_sku</td><td></td><td></td></tr>
<tr><td>shop</td><td></td><td></td></tr>
<tr><td>grass_date</td><td></td><td></td></tr>
<tr><td>shop_gross_orders/sales/gmv</td><td>当日shop级别的gross订单数/销量数/销售额</td><td>同一天同一shop下的量相同</td></tr>
<tr><td>shop_net_orders/sales/gmv</td><td>当日shop级别的net订单数/销量数/销售额</td><td>同一天同一shop下的量相同</td></tr>
<tr><td>l1_category</td><td></td><td></td></tr>
<tr><td>sku_gross_orders/sales/gmv</td><td>当日sku的gross订单数/销量数/销售额</td><td></td></tr>
<tr><td>sku_net_amount/sales/gmv</td><td>当日sku的net订单数/销量数/销售</td><td></td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td></td><td>
<table>
<tr><td></td><td>额</td><td></td></tr>
<tr><td>sku_gross_amount_ltd/cfs/campaign</td><td>当日sku由于ltd/cfs/其他方式的都的销量</td><td></td></tr>
<tr><td>sku_original_price</td><td></td><td></td></tr>
<tr><td>sku_actual_price</td><td></td><td></td></tr>
<tr><td>sku_discount</td><td></td><td></td></tr>
<tr><td>is_campaign</td><td></td><td></td></tr>
<tr><td>campaign_type</td><td></td><td></td></tr>
<tr><td>campaign_stage</td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td></td><td>cleaner/sbd_split</td><td>对清洗后的销量数据划分， 对于shop,前20%销量的日期内数据作为campaign shop, 后80%销量的日期内作为organic shop</td><td>cleaner/sbd_clean 的output</td><td>被标记为campaign shop的历史销量数据：字段同sbd_clean的output<br><br>被标记为organic shop的历史销量数据: 字段同sbd_clean的output</td></tr>
<tr><td>model</td><td>campaign/price/sbd_train</td><td>根据历史的数据在l1颗粒度下训练模型，保存模型</td><td>■ cleaner/sbd_split 的campaign shop<br>■ cleaner/sbd_split 的organic shop</td><td>各个l1-cat 下的模型pkl文件:<br><br>l1-cat1-model.pkl<br><br>l1-cat2-model.pkl<br><br>...<br><br>mp sku的基础信息表（按照mp_sku + discount_bucket + lag_bucket）:
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mp_sku</td><td></td><td></td></tr>
<tr><td>lag_no</td><td>lag长度的桶</td><td>start_date-45days: 0<br><br>start_date-90days: 1<br><br>...</td></tr>
<tr><td>discount_no</td><td>折扣粒度的桶, 总共20个桶</td><td>0~5%: 0<br><br>5%~10%: 1<br><br>...</td></tr>
<tr><td>sales_ref</td><td>日均销量</td><td></td></tr>
<tr><td>discount_ref</td><td>sku 具体折扣</td><td></td></tr>
<tr><td>shop_order_ref</td><td>mp_sku所在shop的日均订单数</td><td></td></tr>
<tr><td>shop_gmv_ref</td><td>mp_sku所在shop的日均gmv</td><td></td></tr>
<tr><td>sku_count_ref</td><td>mp_sku所在shop的日均sku个数</td><td></td></tr>
<tr><td>avg_order_size_ref</td><td>平均每个mp_sku日均订单数</td><td>这个特征酌情考虑一下？先按照shop+date进行transform统计sku+_count,再根据shop_gross_order / sku_count</td></tr>
<tr><td>original_price_ref</td><td>mp_sku 折价前的日均原价</td><td></td></tr>
<tr><td>item_view_ref</td><td>mp_sku 的日均view</td><td>该字段未来不会用</td></tr>
<tr><td>selling_price_ref</td><td>= original_price_ref *<br><br>(1-discount_ref)</td><td></td></tr>
</table>
</td></tr>
<tr><td></td><td>campaign/price/sbd_infer</td><td>根据未来的促销信息和已保存的模型， 对未来的促销进行预测</td><td>• loader的output下的待预测促销数据<br>• campaign/price/sbd_train 的output下的各个模型</td><td>mp sku的测试数据的特征数据表:
<table>
<tr><th>字段</th><th>描述</th><th>说明</th></tr>
<tr><td>mp_sku</td><td></td><td></td></tr>
</table>
</td></tr>
</table>

| | | | | |
|---|---|---|---|---|
| | | | • campaign/price /sbd_train 的output下的**mp sku的基础信息表** | shop | | |
| | | | | grass_date | 预测当时的日期 | |
| | | | | fcst_date | 待预测日的日期 | 注意和grass_date不一样 |
| | | | | lag_no | lag长度的桶 | start_date-45days: 0<br>start_date-90days: 1<br>... |
| | | | | discount_no | 折扣粒度的桶,总共20个桶 | 0~5%: 0<br>5%~10%: 1<br>... |
| | | | | sales_ref | | 对于ref数据的特征，是否有意义呢？ |
| | | | | discount_ref | | |
| | | | | shop_order_ref | | |
| | | | | shop_gmv_ref | | |
| | | | | sku_count_ref | | |
| | | | | avg_order_size_ref | | |
| | | | | original_price_ref | | |
| | | | | selling_price_ref | | |
| | | | | shop_order_uplift | | |
| | | | | shop_gmv_uplift | | |
| | | | | discount_uplift | | |
| | | | | order_size_uplift | | |
| | | | | avg_order_per_sku | | |
| | | | | selling_price | | |
| | | | | dayofweek | | |
| | | | | month | | |
| | | | | dayofmonth | | |
| | | | | weekofmonth | | |
| | | | | salef_uplift | net_amount / sales_ref | 待预测的y值 |
| | | | | **预测结果表:** | | |
| | | | | **字段** | **描述** | **说明** |
| | | | | mp_sku | | |
| | | | | grass_date | | |
| | | | | shop_id | | |
| | | | | l1_category | | |
| | | | | sku_discount | | 注意每个mp_sku + date都有20个buckets |
| | | | | fcst_date | | |
| | | | | fcst_res_total_raw | = fcst_res_cfs+<br>fcst_res_ltd+<br>fcst_res_normal | |
| | | | | fcst_res_cfs_raw | | |
| | | | | fcst_res_ltd_raw | | |
| | | | | fcst_res_normal_raw | | |
| adjust | campaign /adjust_sbd | 结合mp_sku的adis统计和模型的预测结果，对预测结果进行调整 | cleaner/mp_adis 下的output<br><br>campaign/price/sbd_test的output | **字段** | **描述** | **说明** |
| | | | | mp_sku | | |
| | | | | grass_date | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | shop_id | | |
| | | | | l1_category | | |
| | | | | sku_discount | | 注意每个mp_sku + date都有20个buckets |
| | | | | fcst_date | | |
| | | | | fcst_res_total_raw | = fcst_res_cfs_raw+<br>fcst_res_ltd_raw+<br>fcst_res_normal_raw | |
| | | | | fcst_res_cfs_raw | | |
| | | | | fcst_res_ltd_raw | | |
| | | | | fcst_res_normal_raw | | |
| | | | | fcst_res_total_final | = fcst_res_cfs_final+<br>fcst_res_ltd_final+<br>fcst_res_normal_final | |
| | | | | fcst_res_cfs_final | | |
| | | | | fcst_res_ltd_final | | |
| | | | | fcst_res_normal_final | | |

## 算法迭代模块初步构想（待后续和regbd团队一起讨论）

整体上业务同样遵循上述框架： organic 和 campaign 预测， 前者归结于time-series的预测类型， 后者归结于event-driven的预测类型。

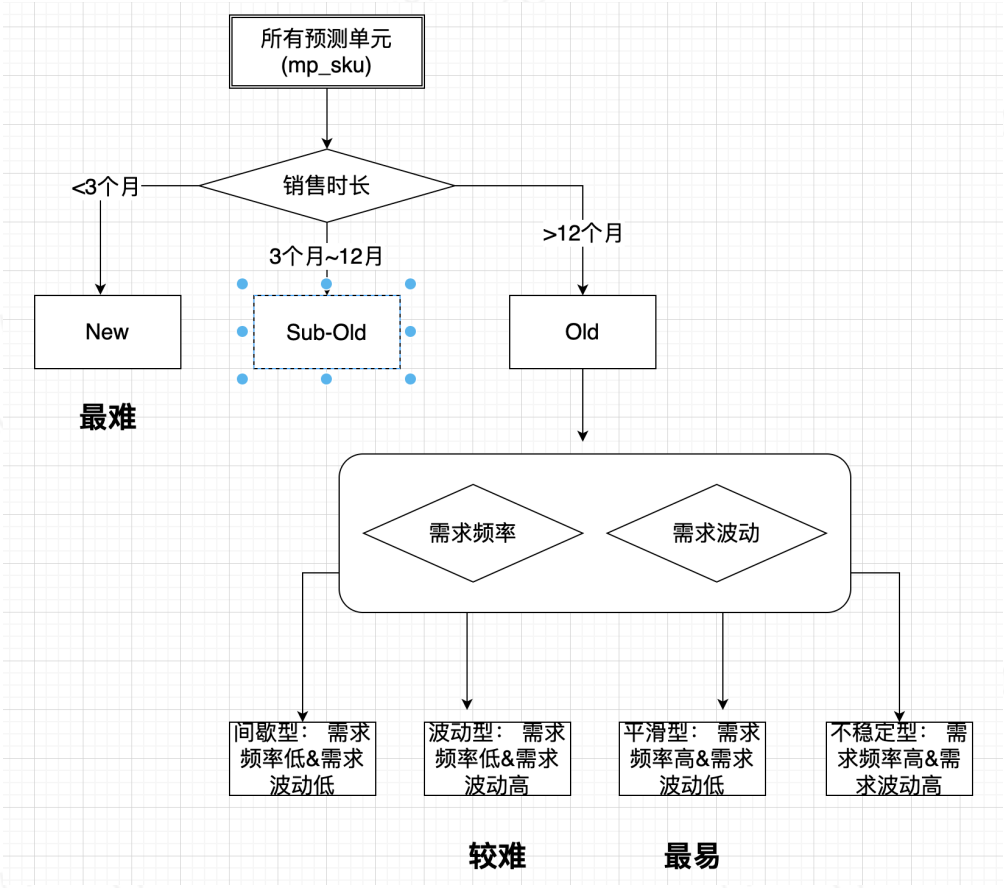
对于各类模型的特点如下

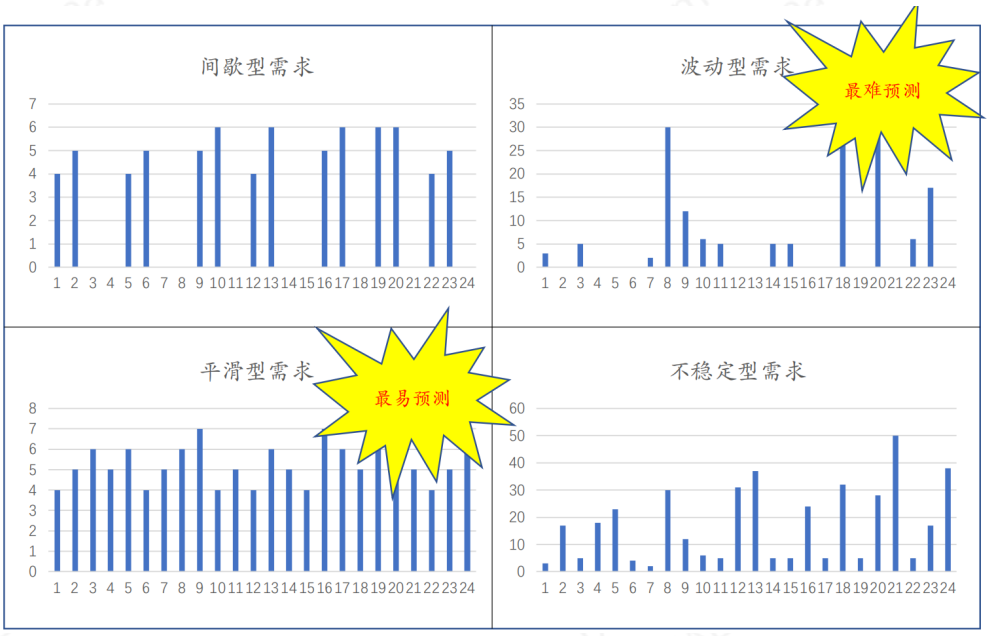
| | 规则类/销售人员预测 | 统计时序模型（单时序预测） | 机器学习/深度学习模型（多时序预测） |
|---|---|---|---|
| **适用问题** | 业务规则导向 | 拥有足够历史数据， 趋势 + 周期较显著的序列 | 多因素影响共振，历史数据量足够多 |
| **代表模型** | ▪ 销售人员根据前端销售情况拍脑袋<br>▪ 同环比模型 | prophet，STL，arima， holtwinters | Linear，LGB， RF，transformer... |
| **优点** | • 简单， 稳定， 可解释<br>• 销售人员可直观灵活地感知前端需求<br>• 销售人员脑中有一些无法量化的因子 | • 模型相对稳定<br>• 模型可解释较差 | • 相对准确<br>• 集中训练， 如果序列/sku较多的话， 训练的时间效应体现<br>• 可以交叉学习，例如如果商品历史数据不足的话，仍能给出预测 |
| **挑战** | • 个人经验可能不准<br>• 无法规模化，耗时耗力 | • 每次预测时需要重新fit, 比较费时<br>• 冷启动问题， 如果商品历史数据量不足，预测值参考意义不大<br>• 如果序列数较多，需要考虑分布式方式加速<br>• 多序列如何调参 | • 解释性较差<br>• 稳定性， 有时候模型输出的结果可能比较离谱，需要后处理adjust |

注意**我们的设计策略不会对一个预测场景仅使用一种模型， 会融合各个模型来应对场景的复杂性**

### Organic预测

根据电商场景的特点结合历史经验，需要对业务场景做一定的归类， 具体如下：




其中各类的对比如下：

| 场景 | 新品（new） | 准老品(sub-old) | 老品(old) |
| --- | --- | --- | --- |
| 特点 | 商品刚上新阶段，往往有一些营销活动 | 商品上新不足半年， 有一定历史数据，但可能无法完整学习到年度周期性 | 商品销量数据积累超过一年，可以有足够历史数据学习规律 |
| 模型选择 | 寻找同品类下的相似商品，建立机器学习模型 | ■ 常规时序类模型<br>■ 机器学习模型 | ■ 间歇型： croston 或者普通规则<br>■ 波动型： croston 或者 moving-average<br>■ 平滑型： 常规时序模型 或者 机器学习模型 |

| | | | |
|---|---|---|---|
| | | | ▪ 不稳定型： moving-average 或者 尝试使用时序模型 |
| 预测难度 | 最难 | 中 | ▪ 间歇型：中<br>▪ 波动型：最难<br>▪ 平滑型： 最易<br>▪ 不稳定型：中 |
| 备注 | 之后迭代可以考虑如何和产品融合 | 可以和old品界限没有那么严，有时候可以放在一起考虑 | |

注意， 以下思考可以在后续迭代中和业务，产品讨论：

- 对于历史数据中需求频率较小的分类中（间歇型和波动型）， 除了使用模型给出结果外，可以**考虑在粗颗粒度（例如l1/l2级别）聚合预测，然后按照一定比例拆分到sku**
- 对于新品和老品的交替和替代关系等，需要维护和整合，不然容易出错
- 对于新品的预测，模型可以基于相似品的机器学习模型给出参考值，另外一种思路是针对新品专门的工作台

## Campaign预测

### price相关促销（shop-wised: SBD）

由于运营人员需要输出同一sku下不同的discount的销量，如：

| 日期 | 2022-04-05 | | | | 2022-04-06 | | | | ... |
|---|---|---|---|---|---|---|---|---|---|
| | 0~5% | 5%~10% | ... | 95%~100% | 0~5% | 5%~10% | ... | 95%~100% | |
| sku1 | | | | | | | | | |
| sku2 | | | | | | | | | |
| ... | | | | | | | | | |

**价格弹性的预测模型**

这是比较直观且易解释的，其中弹性定义如下：

In economics, we define the price elasticity:

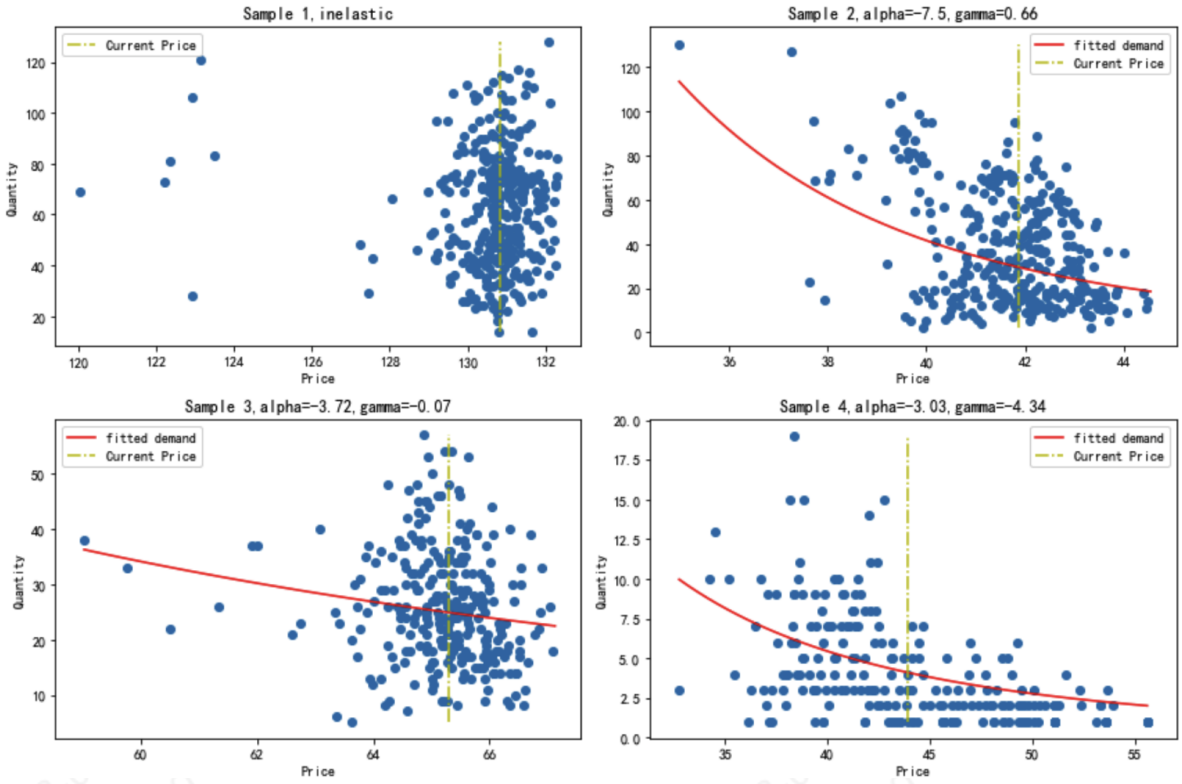
$$\epsilon = \frac{\%Change\ in\ demand}{\%Change\ in\ price} = \frac{\Delta Q/Q}{\Delta P/P} = \frac{(Q_2 - Q_1)/Q_1}{(P_2 - P_1)/P_1} = \frac{1 - Q_2/Q_1}{1 - P_2/P_1}$$

Mathematically,
$1 + x \approx lnx$,(when $x$ is small).
$\epsilon \approx \frac{ln(D_2/D_1)}{ln(P_2/P_1)}$
with some more mathematical transform, we can get:
$ln(D) = c - \epsilon ln(P)$
**We can approximate the price elasticity with regression analysis!**

根据sku的历史销量 vs 历史价格，进行线性回归，也可以加入其他因素，如下：

$$logQ = \epsilon logP + log(P_{sub}) + log(P_{com}) + log(P_{t-1}) + \sum \alpha_j PromotionType_j + Inventory + Trend + Seasonal + HolidayFactor$$

上述的epsilon即为该sku的需求弹性。以下是一些拟合案例： 但是sku级别是否有足够多的历史价格点？




注意：

- 价格弹性模型实际上依赖于历史上有足够多的价格变动点，**如果sku级别没有足够的历史变动数据，弹性则无法估计，可以考虑在粗颗粒度（l1/l2）进行计算**
- 价格弹性模型可以**保证销量和价格的单调性**（不会存在价跌量跌的情况）
- 线性模型有一定的线性外推，但是某些情况下的线性外推会造成异常销量预测， 需要在预测过程中约定边界！
- 弹性模型做预测时，可以加入其他因素，但注意量纲

**其他模型**

可以考虑使用其他非线性模型（如tree_based）， 但是注意：

- 销量和价格的单调性关系需要保持（不要出现价跌量跌的情况）

### non-price相关促销（platform-wised: DD）

对于DD这一类场景， 可以考虑建立机器学习模型，包含的特征包括：

- 商品信息
- shop信息
- 历史DD的表现
- 最近商品的非促平均销量（adis）

## 监控模块

### 模型错误处理方案

to-be discussed with dev

由于目前算法是离线计算输出到Hive表内，因此存在一定风险无法输出结果，导致业务无法获得最新预测展示，因此影响使用。具体错误方案处理在UAT-adoption中会试运营一段时间

| | organic-infer job | SBD-train job | SBD-infer job | DD-infer job |
|---|---|---|---|---|
| 原因 | ▪ 代码逻辑出错， 没有考虑到某些边界条件 | | | |

| | | | | |
|---|---|---|---|---|
| | ▪ 线上资源紧张，导致某些Job耗费时间较长， 没有在规定时间内给出结果 | | | |
| 监控预警 | ▪ 邮件预警<br>▪ 电话/信息/seatalk | 邮件预警 | ▪ 邮件预警<br>▪ 电话/信息/seatalk | ▪ 邮件预警<br>▪ 电话/信息/seatalk |
| 兜底方案 | 后端不读取当日的预测结果， 读取昨日的预测结果 | 使用上次训练的模型 | ▪ 算法摘取上次SBD的预测结果<br>▪ 不兜底 | ▪ 算法摘取上次DD的预测结果<br>▪ 不兜底 |
| 事后处理方案 | 算法此日解决问题 | 算法在一周内解决问题 | 算法此日解决问题 | 算法此日解决问题 |
| 时效性 | 高 | 中 | 高 | 高 |

注意上述所说的兜底方案是算法最终没有写入结果到hive表后（**按理说概率很小**）的处理方案，对于代码内部如何兜底处理的逻辑， 有job控制即可（try...except...时在except中处理异常逻辑）！

### 数据质量监控方案

对于每个layer之间的中间表， 对中间表的数据进行一定程度的汇总检查， 检查内容包括：

- sku的个数， 避免出现处理后漏掉/重复sku的情况
- 对于一些历史销量的处理， 从总量上对处理前后进行对比，避免出现不一致的情况（有时候表的各种Join, 可能存在重复/缺失）
- 异常数据的监控

### 准确率监控方案

#### 当前预测准确率评价指标

to-be-discussed with pm:

- organic 和 campaign的评价场景是否一样
- mp_sku or mt_sku
- 评价场景：
  - 口径： net or gross
  - lead time: 衡量第几版的预测值
  - horizon: 是未来一天的还是几天之和

目前预测准确率的指标

## 关于测试用例

需要和测试以及产品对其后确认相应用例，待后续补充

### Organic预测

### DD 促销预测

### SBD 促销预测

# Ref: 历史问题的一些记录

### 业务上的问题和澄清：

- 梳理sbd和dd的代码模块介绍
- 梳理sbd和dd能跑通的案例

- 如何对比说明当前业务上模型的可用性和提升程度
    - 等UAT阶段进行盲测对比
    - 历史预测结果获取进行回测
- 预测和补货这两件事的边界，体现在多个方面： 预测的颗粒度，操作人员干预的颗粒度...
    - organic 是mt + wh
    - campaign 是mp, 对应的mt下的mp的集合
- 算法业务边界：
    - organic 和 campagin data 分别给出相应的基础数据， 算法计算， 输出结果落表
    - 算法需要对接两个数据方向：
        - data 给的基础数据
        - dev 传入的一些参数型数据
    - 触发是自动触发，不支持及时触发，因此基本不存在前端交互型的接口
- 关于cfs 和 ltd 的模型结果如何产生？
    - cfs 和 ltd 目前的模板是一致，包括输入数据和输出数据
    - 输入：
        - 未来的促销计划（用户输入）： campaign_detail 中的 cfs/ltd_sku_discount, cfs/ltd_qty_limit
        - 历史上的mp + date颗粒度的数据： sku_net/gross_amount_ltd/cfs, 将日期均摊到每天的销量上（由于均摊，导致真实销量 和 fs 的销量关系切分的有一些不准？）
    - 输出：
        - forecasted_qty_campaign_cfs/ltd: 在 mp_sku + date 级别输出均摊后的结果
- 关于 gross 和 net 的统计口径？ 关于订单的状态的筛选（net 的口径是 ('ESCROW_CREATED','PAID','ESCROW_PAID','ESCROW_VERIFIED','ESCROW_PENDING','COMPLETED')）
    - gross 代表原始需求的销量， net 代表实际中卖了的量, 应该大部分时间是一致的，可能存在退货，对账，以及可能的延迟的情况！
    - 区分
        - gross :
            - 区分organic shop
            - 特征
        - net:
            - 计算mp颗粒度的adis
            - sbd模型中的计算y值使用的口径： net_amount / organic shop下对应的net_amount均值
- 关于 training set 和 test set 的数据集准备
    - 筛选逻辑
        - 哪些日期是促销日期： 根据shop 级别的数据
        - 根据pv数据
        - 根据销量的sigma原则
    - trainning set 的discount 核算逻辑是折算的，test set 的discount 是用户输入的
- campaign 的的触发方式， 目前是date - 45天【需要和韦奇对】
    - 算法读取campagin 日历判断是否触发促销日期之前的45天嘛？更新促销日历后
        - case 1: 促销时间 - 当前日期 > 45, 不跑
        - case 2: 促销时间 - 当前日期 = 45, 第一次跑
        - case 3: 促销时间 - 当前日期 < 45, 更新跑
    - 数据仍需要每天更新嘛？对于organic的数据是是每天更新的， 如果每个月一次的频率的话
- sbd 的模型中的 organic shop 和 campaign shop 的区分处理， 当前是针对campaign shop不足数据进行扩充的选择
- 关于mp的范围：organic mt sku 下的所有的mp sku
- 关于商品折扣信息， 目前夹杂了一些rebate的信息， 我们需要的折扣信息（未来提供预测实际折扣信息）， 这部分如何对应?
    - original pirce:**`before_discount_price_usd`** → sbs_sku_attr ， 单位是否匹配(美元和印尼盾的区别)？
    - actual price: **`list_price_usd`** → sbs_sku_attr
    - sku_discount: CASE WHEN sum(rebate_wo_fs+sku_price_wo_fs) = 0 THEN NULL ELSE sum(rebate_wo_fs)/sum (rebate_wo_fs+sku_price_wo_fs) END ， → order item的统计数据， 如果不为空，就用统计数据，如果为空，就用actual_price 和 original price 的差值
    - sku_real_discount: CASE WHEN sum(rebate_wo_fs+sku_price_wo_fs) = 0 THEN NULL ELSE sum(rebate_wo_fs + shopee_coin_rebate_usd+shopee_voucher_rebate_usd+seller_voucher_rebate_usd) / sum(rebate_wo_fs+sku_price_wo_fs) END， → order item的统计数据，
    - 算法参考的折扣数据是哪部分(rebate 这部分的逻辑我理解不需要考虑的)？未来预测的折扣数据口径是什么？
- 关于Purchsable 的逻辑，目前是两部分，这个需要区分嘛？还是目前揉在一起？对于purchable = 1的
    - 库存状态 'normal' 且 库存量 > 0
    - gross/net order > 0

```
    full_sku_df = pd.concat(sku_df_list)
    #
    raw_df['grass_date'] = pd.to_datetime(raw_df['grass_date'])
    raw_df2 = full_sku_df[['sku_id','grass_date','sku_stock', 'sku_status']].merge(raw_df[['sku_id','grass_date','net_amount']], on=
    raw_df2['is_purcasable'] = np.where(((raw_df2['net_amount'] > 0) | ((raw_df2['sku_stock']>0) & (raw_df2['sku_status']==1))), 1
    grouped_raw_df2 = raw_df2.groupby('sku_id').is_purcasable.sum().reset_index()
    grouped_raw_df2.columns = ['sku_id', 'purchasable_days']
    #
    adis = raw_df.groupby('sku_id').net_amount.sum().reset_index()
    adis.columns = ['sku_id','total_sales']
    adis = adis.merge(grouped_raw_df2,on=['sku_id'],how='left')
    adis['adis'] = adis['total_sales'] / (adis['purchasable_days'] + 0.001)
    #
    test_df = test_df.merge(adis,on=['sku_id'],how='inner')
    test_df['uplift_vs_adis'] = test_df['pred_sales'] / (test_df['adis'] + 0.001)
    test_df['grass_date_str'] = test_df['grass_date'].astype('str')
    #
    test_df['peak_day'] = test_df['peak_day'].astype('str')
    #
```

- mp 颗粒度的adis（DD 维度也是有这个需求的），是否需要和organic 的对其(因为后处理的时候需要针对adis做围栏限制)？ 还是算法根据历史销量表自行统计就好
    - sbd 模型中后处理逻辑
    - DD 模型中基础销量作为base
    - 如果算法维护的话，是否存成表做版本记录？
- 后处理逻辑SBD有，DD目前缺失，目前不需要统一?
- 替换关系的维护和数据， 是否有逻辑？ 对应的问题是mp-mt相关的对应关系
- 当前是每天都判断逻辑输出预测结果，如果用户修改了预测结果，那么页面展示的逻辑是什么？ 因为算法是每天都更新预测结果的【需要和韦奇对】
- DD fcst 在计算previous DD l1 uplift 系数 和 标准差时，使用了adis, 这个adis 是previous DD 的adis 还是当前节点的adis?
- 当前SBD 在设计计算 uplift 相关的特征时有数据泄露的问题（base info）的数据都是基于同一份数据，针对此种情况如何设计？
- 后端触发算法的方式的参数如何传给算法？ 因为当前可能是定时任务调度，那么是否意味着需要将对应的参数信息保存到表呢？【需要和韦奇对】
- 如果输出所有price bucket的销量然后让用户选择，那预测结果如何评估？ 如何对比优化的算法？
- 用户上传的其他促销类型的预测结果，是否可以和当前的促销结果兼容【需要和伟奇对】

**设计上考虑的问题：**

- 【internal】train 和 test 的节奏， 每次都retrain 和 做测试？
- 【internal】当前organic下预测是在mt_sku颗粒度， campaign下预测是在mp_sku 颗粒度， 但是未来的方向是预测颗粒度希望都收在mp_sku颗粒度
- 【internal & with dev】pyspark vs pure python?
- 【with dev】预测的触发方式
- 【internal】当前的么个region 的流程有一些不一样， 所以在流程设计中需要兼容不同region
- 【internal】随着biz的扩展，运营的精细化趋势等，可能会在category 级别上预测，需要兼容此方面的发展
- 【with dev】考虑在如果算法报错的情况下， 如何处理？ 需要兜底， 需要再工程设计中给出
- 【internal】 关于数据链路中埋点和check 的逻辑，需要在工程设计中给出
- 【internal】生命周期如何考虑到当前的逻辑中
- 【with dev】 关于历史上不同区域下platform-level campaign 日期的收集
- 【internal】 当前SBD 为了扩充训练集，在date 上根据shop的销量进行campaign标签处理， 此部分逻辑是否有改动的可能和必要
- 【internal】 当前的促销模型的设计如何兼容后续的更新： 参考亚马逊的价格模型（SBD）和 非价格类促销(DD), 架构上可以分层处理
- 【internal】当前organic forecast 中的预测是结合: price band 和 adis 进行选择(注意是mt颗粒度)， 因此也属于部分的价格模型， 后续的平销模块同样需要注意价格的影响
- 【Internal】 关于商品生命周期的引入和处理方式
- 【internal】 日志的方式， 需要再工程设计中给出
- 【internal】 如何设计算法的稳定性，性能，效果...
- 【internal with dev】user case， 给定一些案例作为测试时需要考虑的！
- 【internal】 关于算法， data, dev， qa 等如何合作进行测试设计
- 【Internal】关于特征是在sql中计算还是通过pyspark计算， pyspark计算的话可能会比较慢（有些是数据倾斜的问题）
- 【internal】在设计campaign module时，price 和 non-price的边界是如何的，是否会针对non-price发放一些券? 然后做一些深度的分析？