乘用车销量预测比赛复赛第六名解题思路

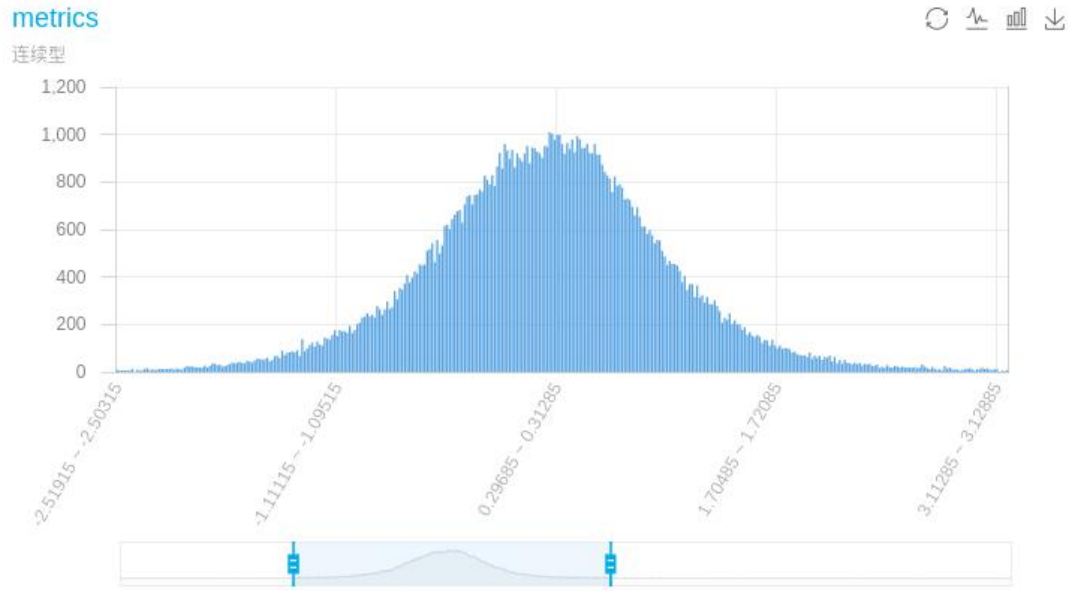
数加平台开放的模型和功能有限，因此重点放在数据处理、数据分析、特征提取、数据分类、结果后处理方面，模型超参数上花的时间不多。本来打算多做些 ensemble 的，但是由于平台资源问题不愿意排队等资源决定放弃。
其中在数据分类上，可以采用多种特征对数据进行分类（如车型配置、宏观经济指标等），但在本方案中由于对 sql 语句和平台不够熟练，只利用了模型预测残差对数据进行分类。
另外，在复赛阶段的评价指标本质上是对于真实值和预测值的比例求 log，这种指标在同样的误差量下，小的销量计算出的评价指标大于大的销量。因此，如何将小的销量预测的更准显得尤为重要。

算法思路：

建立一个能够解决问题的比较粗糙的模型如下：
Model=m(theta, features)×alpha(month,city,class)；
其中 Model 为最终模型，m 为机器学习模型，需要特征 features 训练出参数 theta，alpha 为和月份、城市、车型相关的修正系数（本方案将 alpha 简化为和月份相关）。
具体实施的 pipeline 如下：
1. 补值：保证所有不同的（sale_date, city_id, class_id）组合都有对应的 sale_quantity 值，为下一步特征提取做准备；
2. 提取特征，主要包括
    a) 日期特征：年、月、节假日信息（假期天数、重要性、是否春节、五一、国庆、工作天数）
    b) 销量特征：上 N 个月销量（N 是[1,12]范围内的整数）、前两月销量的均值、前三月销量均值、前六个月销量均值、前十二个月销量均值、前三月销量中位数、前五月销量中位数
    c) 产量特征：当月产量；
    PS.从模型重要性输出来看，有用的主要还是销量特征，日期特征基本没用。

序号	colname	feature_importance
23	last_2m_avg	0.5378941438271632
4	last_month_sale	0.1933890042125439
7	last_3m_avg	0.13744574410721222
20	produce_quantity	0.02552387624397879
29	median_3month_sale	0.02060510398852421
19	last_12m_avg	0.013414277079058939
18	last_6m_avg	0.010666195273564149

3. 划分 train, val, test 数据集，其中 label 是 log（sale_quantity），以让被预测量的变化相对比较平稳；
4. 建立 GBDT 模型，分别查看不同特征、参数在训练集和验证集上的结果；
5. 在验证集上观察 metrics 的直方图，将 201712 月的（city_id, class_id）分为三类 (a.metrics 为负且小于-0.5； b.metrics 为正且大于 1； c.其他)。并对 a 和 b 重新构建模型训练（在这里为图方便，使用同样的模型超参数）；



6. 在测试集进行预测，并根据历史同期计算修正参数；
7. 后处理：对小于 1 的预测置为 1；
注 1 :在复赛预测 201802 时，使用了 201801 的预测结果作为真实值，加入预测 201802 的特征提取步骤。
注 2：在复赛预测 201802 时，笔者尝试搭建隔月预测模型，但由于 validation 结果差于注 1 中的方案，最终放弃，故不再赘述。

在平台上完整可运行代码：

1. 拉数据表：

Create table if not exists yc_passenger_car_sales as select * from odps_tc_257100_f673506e024.yc_passenger_car_sales;
Create table if not exists yc_passenger_car_yields as select * from odps_tc_257100_f673506e024.yc_passenger_car_yields;
Create table if not exists yc_macro_econ as select * from odps_tc_257100_f673506e024.yc_macro_econ;
Create table if not exists yc_result_sample_a as select * from odps_tc_257100_f673506e024.yc_result_sample_a;

2. 构建数据，补全所有不同的（sale_date, city_id, class_id）组合
--提取关键的 sale_quantity 数据

```
Create table ts as
    select sale_date, province_id, city_id, class_id, sum(sale_quantity) as sale_quantity from yc_passenger_car_sales group by sale_date, province_id, city_id, class_id;

--建立一张新表，包含了所有不同的（sale_date, city_id, class_id）组合

drop table if exists data_full;
create table data_full as
select a.sale_date, c.city_id, d.class_id
from (select distinct sale_date, 1 as raoguo from ts) a
join (select distinct city_id, 1 as raoguo from ts) c
join (select distinct class_id, 1 as raoguo from ts) d
on a.raoguo = c.raoguo and a.raoguo=d.raoguo ;

--建立一张新表，包含了 province_id 和 city_id 的对应关系
create table p2c as select distinct ts.province_id, ts.city_id from ts;

--建立一张新表，包含了所有的（sale_date, city_id, class_id）组合，并对空值补零
drop table if exists haha;
create table haha as
    select data_full.sale_date, p2c.province_id, data_full.city_id, data_full.class_id, (case when ts.sale_quantity is NULL then 0 else ts.sale_quantity end) as sale_quantity
    from ts right JOIN data_full
    ON ts.city_id=data_full.city_id AND ts.class_id=data_full.class_id AND ts.sale_date=data_full.sale_date,
    p2c where p2c.city_id=data_full.city_id;

--同理，对产量表进行补全：建立一张新表，包括了所有的（sale_date, class_id）组合，对缺失的 produce_quantity 补零
create table class_full as
select a.sale_date, d.class_id
from (select distinct sale_date, 1 as raoguo from ts) a
join (select distinct class_id, 1 as raoguo from ts) d
on a.raoguo=d.raoguo ;

drop table if exists produce_full;
create table produce_full as
    select class_full.sale_date as produce_date, class_full.class_id, (case when y.produce_quantity is NULL then 0 else y.produce_quantity end) as produce_quantity
    from yc_passenger_car_yields y right join class_full
    on y.class_id =class_full.class_id and y.produce_date=class_full.sale_date;
```

```
--假期数据
--HOLIDAY DATA
drop table if exists holiday;

create table if not exists holiday(
holiday_date STRING ,
holidays bigint ,
important bigint,
spring_month bigint,
nd_month bigint,
labor_month bigint,
work_days bigint
);

insert into table holiday values
(   201201 , 7, 1, 1, 0, 0, 16),
( 201202 , 0, 0, 0, 0, 0, 21),
(   201203 , 0, 0, 0, 0, 0, 22),
(   201204 , 4, 1, 0, 0, 0, 21),
(   201205 , 1, 1, 0, 0, 1, 19),
(   201206 , 1, 0, 0, 0, 0, 21),
(   201207 , 0, 0, 0, 0, 0, 22),
(   201208 , 0, 0, 0, 0, 0, 23),
(   201209 , 0, 0, 0, 0, 0, 20),
(   201210 , 5, 1, 0, 1, 0, 20),
(   201211 , 0, 0, 0, 0, 0, 22),
(   201212 , 0, 0, 0, 0, 0, 21),
(   201301 , 3, 1, 0, 0, 0, 22),
(   201302 , 5, 1, 1, 0, 0, 17),
(   201303 , 0, 0, 0, 0, 0, 21),
(   201304 , 4, 0, 0, 0, 0, 21),
(   201305 , 1, 1, 0, 0, 1, 22),
(   201306 , 3, 0, 0, 0, 0, 19),
(   201307 , 0, 0, 0, 0, 0, 23),
(   201308 , 0, 0, 0, 0, 0, 22),
(   201309 , 2, 0, 0, 0, 0, 21),
(   201310 , 5, 1, 0, 1, 0, 19),
(   201311 , 0, 0, 0, 0, 0, 21),
(   201312 , 0, 0, 0, 0, 0, 22),
(   201401 , 2, 1, 0, 0, 0, 22),
(   201402 , 4, 1, 1, 0, 0, 17),
(   201403 , 0, 0, 0, 0, 0, 21),
(   201404 , 1, 0, 0, 0, 0, 21),
```

(201405 , 2, 1, 0, 0, 1, 21),
(201406 , 1, 0, 0, 0, 0, 20),
(201407 , 0, 0, 0, 0, 0, 23),
(201408 , 0, 0, 0, 0, 0, 21),
(201409 , 1, 1, 0, 0, 0, 22),
(201410 , 5, 1, 0, 1, 0, 19),
(201411 , 0, 0, 0, 0, 0, 20),
(201412 , 0, 0, 0, 0, 0, 23),
(201501 , 2, 1, 0, 0, 0, 21),
(201502 , 5, 1, 1, 0, 0, 17),
(201503 , 0, 0, 0, 0, 0, 22),
(201504 , 1, 0, 0, 0, 0, 21),
(201505 , 1, 1, 0, 0, 1, 20),
(201506 , 1, 0, 0, 0, 0, 21),
(201507 , 0, 0, 0, 0, 0, 23),
(201508 , 0, 0, 0, 0, 0, 21),
(201509 , 2, 0, 0, 0, 0, 21),
(201510 , 5, 1, 0, 1, 0, 18),
(201511 , 0, 0, 0, 0, 0, 21),
(201512 , 0, 0, 0, 0, 0, 22),
(201601 , 1, 0, 0, 0, 0, 20),
(201602 , 5, 1, 1, 0, 0, 18),
(201603 , 0, 0, 0, 0, 0, 23),
(201604 , 1, 0, 0, 0, 0, 20),
(201605 , 1, 1, 0, 0, 1, 21),
(201606 , 2, 0, 0, 0, 0, 21),
(201607 , 0, 0, 0, 0, 0, 21),
(201608 , 0, 0, 0, 0, 0, 23),
(201609 , 2, 1, 0, 0, 0, 21),
(201610 , 5, 1, 0, 1, 0, 18),
(201611 , 0, 0, 0, 0, 0, 22),
(201612 , 0, 0, 0, 0, 0, 22),
(201701 , 4, 1, 1, 0, 0, 19),
(201702 , 2, 1, 0, 0, 0, 19),
(201703 , 0, 0, 0, 0, 0, 23),
(201704 , 2, 0, 0, 0, 0, 19),
(201705 , 3, 1, 0, 0, 1, 21),
(201706 , 0, 0, 0, 0, 0, 22),
(201707 , 0, 0, 0, 0, 0, 21),
(201708 , 0, 0, 0, 0, 0, 23),
(201709 , 0, 0, 0, 0, 0, 22),
(201710 , 5, 1, 0, 1, 0, 17),
(201711 , 0, 0, 0, 0, 0, 22),
(201712 , 0, 0, 0, 0, 0, 21),

(201801 , 1, 1, 0, 0, 0, 22),
(201802 , 4, 1, 1, 0, 0, 16);

先对 201801 进行预测

3. 数据处理完后，可以进行特征提取并划分数据集，其中 201712 之前数据作为训练，201712 数据作为验证，201801 作为测试集

----------FEATURES TRAIN SET
drop table if exists temp0311_1;
create table temp0311_1 as
select z.sale_date, z.province_id, z.city_id, z.class_id, bigint (substr(z.sale_date, 1, 4)) as year, bigint(substr(z.sale_date, 5, 6)) as month, sin(bigint(substr(z.sale_date, 5, 6))) as sin_month,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale,
    sum(yc_passenger_car_sales.sale_quantity) as current_sale,
    log(sum(yc_passenger_car_sales.sale_quantity)) as log_current_sale
from yc_passenger_car_sales,
    (select sale_date, province_id, city_id, class_id, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_month_sale,
    lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_2month_sale,
    lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_3month_sale,
    lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_4month_sale,
    lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_5month_sale,
    lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_6month_sale,
    lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_7month_sale,
    lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_8month_sale,
    lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_9month_sale,
    lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_10month_sale,

```
lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_11month_sale,
lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_year_sale,
(lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/2 as last_2m_avg,
(lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/3 as last_3m_avg,
(lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
+lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/6 as last_6m_avg,
(lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
+lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
+lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
+lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/12 as last_12m_avg,
ordinal(2, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)) as median_3month_sale,
ordinal(3, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id,
```

```
province_id  ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    , lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)) as median_5month_sale
    from haha) z
    where z.sale_date<'201712'
    and z.sale_date=yc_passenger_car_sales.sale_date
    and z.province_id=yc_passenger_car_sales.province_id
    and z.city_id=yc_passenger_car_sales.city_id
    and z.class_id=yc_passenger_car_sales.class_id
group by z.sale_date, z.province_id, z.city_id, z.class_id,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale;


drop table if exists temp0311_2;
create table temp0311_2 as
select t.sale_date, t.province_id, t.city_id, t.class_id, t.year, t.month, t.sin_month,
    t.last_month_sale, t.last_2month_sale, t.last_3month_sale, t.last_4month_sale, t.last_5month_sale, t.last_6month_sale, t.last_7month_sale, t.last_8month_sale, t.last_9month_sale, t.last_10month_sale, t.last_11month_sale,
    t.last_year_sale, t.last_2m_avg, t.last_3m_avg, t.last_6m_avg, t.last_12m_avg,
    t.median_3month_sale, t.median_5month_sale,
    produce_full.produce_quantity,
    holiday.holidays, holiday.important, holiday.spring_month, holiday.nd_month, holiday.labor_month, holiday.work_days,
    t.current_sale,
    t.log_current_sale
from produce_full, holiday,
    temp0311_1 t
    where t.sale_date<'201712'
    and t.sale_date=produce_full.produce_date
    and t.sale_date=holiday.holiday_date
    and t.class_id=produce_full.class_id;
```

```
drop table if exists features_train0311;
ALTER table temp0311_2    rename to features_train0311;


--FEATURES VAL SET
drop table if exists features_val0311;
--
create table features_val0311 as
select z.sale_date, z.province_id, z.city_id, z.class_id, bigint (substr(z.sale_date, 1, 4)) as year, bigint(substr(z.sale_date, 5, 6)) as month, sin(bigint(substr(z.sale_date, 5, 6))) as sin_month,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale,
    produce_full.produce_quantity,
    holiday.holidays, holiday.important, holiday.spring_month, holiday.nd_month, holiday.labor_month, holiday.work_days,
    sum(yc_passenger_car_sales.sale_quantity) as current_sale,
    log(sum(yc_passenger_car_sales.sale_quantity)) as log_current_sale
from yc_passenger_car_sales, produce_full, holiday,
    (select sale_date, province_id, city_id, class_id, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) as last_month_sale,
    lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_2month_sale,
    lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_3month_sale,
    lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_4month_sale,
    lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_5month_sale,
    lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_6month_sale,
    lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_7month_sale,
    lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_8month_sale,
    lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id
```

```
ORDER BY sale_date) as last_9month_sale,
    lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_10month_sale,
    lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_11month_sale,
    lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_year_sale,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/2 as last_2m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/3 as last_3m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/6 as last_6m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/12 as last_12m_avg,
    ordinal(2, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 3, 0)
```

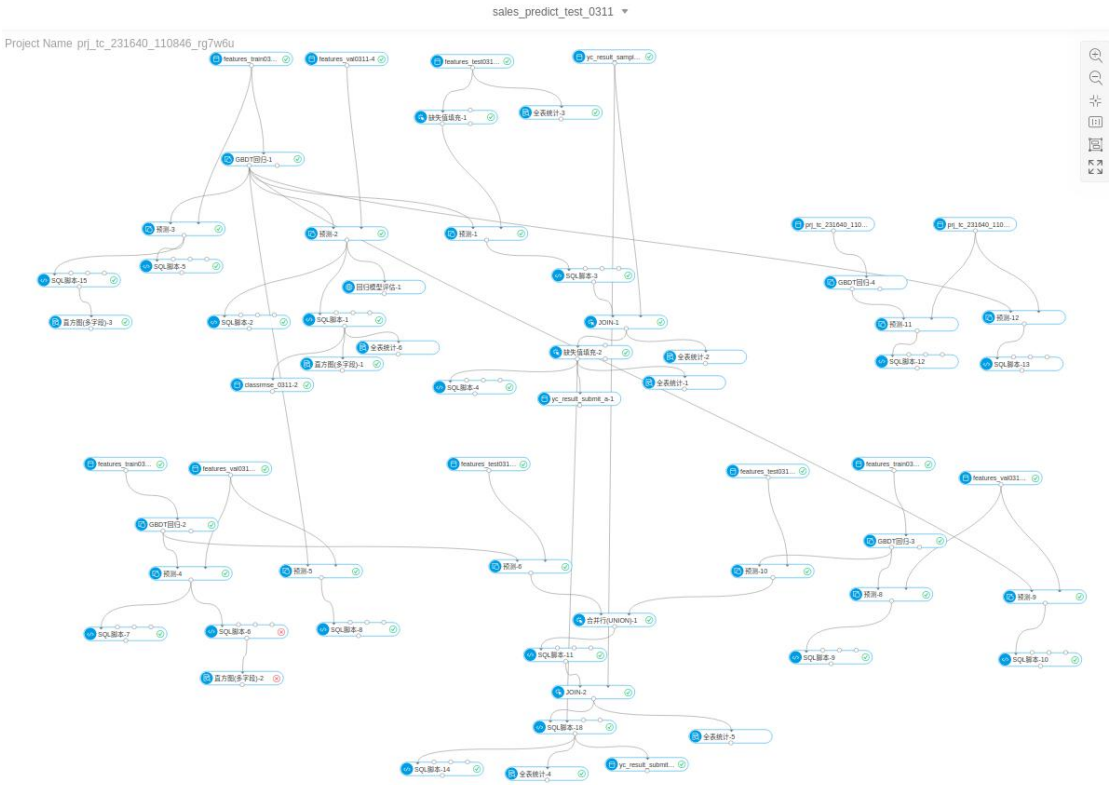
```
OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)) as median_3month_sale,
    ordinal(3, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    , lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)) as median_5month_sale
    from haha) z
    where z.sale_date='201712'
    and z.sale_date=yc_passenger_car_sales.sale_date
    and z.sale_date=produce_full.produce_date
    and z.sale_date=holiday.holiday_date
    and z.class_id=produce_full.class_id
    and z.province_id=yc_passenger_car_sales.province_id
    and z.city_id=yc_passenger_car_sales.city_id
    and z.class_id=yc_passenger_car_sales.class_id
group by z.sale_date, z.province_id, z.city_id, z.class_id,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale,
    produce_full.produce_quantity,
    holiday.holidays, holiday.important, holiday.spring_month, holiday.nd_month, holiday.labor_month, holiday.work_days;


-- FEATURES TEST SET，由于未知 201801 产量数据，这里使用 201712 产量/1.5 作为估计值
--注：这里在测试集的构建上，对 sale_quantity 做 lag 时出现了失误，多 lag 了一行，到复赛 B 榜时才发现。这里为了保证真实性，不做修改，特此说明。
drop table if exists features_test0311 ;

create table features_test0311 as
select yc_result_sample_a.predict_date, z.province_id, z.city_id, z.class_id, bigint (substr(yc_result_sample_a.predict_date, 1, 4)) as year, bigint(substr(yc_result_sample_a.predict_date, 5, 6)) as month, sin(bigint(substr(yc_result_sample_a.predict_date, 5, 6))) as sin_month,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale,
```

```
z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale,
    BIGINT(produce_full.produce_quantity/1.5) as produce_quantity,
    holiday.holidays, holiday.important, holiday.spring_month, holiday.nd_month, holiday.labor_month, holiday.work_days
from yc_result_sample_a, produce_full, holiday,
    (select sale_date, province_id, city_id, class_id, sale_quantity as last_month_sale,
    lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_2month_sale,
    lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_3month_sale,
    lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_4month_sale,
    lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_5month_sale,
    lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_6month_sale,
    lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_7month_sale,
    lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_8month_sale,
    lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_9month_sale,
    lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_10month_sale,
    lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_11month_sale,
    lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_year_sale,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/2 as last_2m_avg,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/3 as last_3m_avg,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id,
```

```
city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date))/6 as last_6m_avg,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) +lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)
    +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)
    +lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)
    +lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date))/12 as last_12m_avg,
    ordinal(2, sale_quantity, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)) as median_3month_sale,
    ordinal(3, sale_quantity, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)
    , lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)) as median_5month_sale
    from haha) z
    where z.sale_date='201712'
    and produce_full.produce_date='201712'
    and z.sale_date=holiday.holiday_date
    and z.class_id=produce_full.class_id
    and z.province_id=yc_result_sample_a.province_id
    and z.city_id=yc_result_sample_a.city_id
    and z.class_id=yc_result_sample_a.class_id;
```

4. 在 PAI 中建立模型：

完整模型如下图所示，注：如果需要复现模型的话，图中一些用于评价结果的 SQL 脚本组件可删除，只保留重要的组件。



其中左上角部分为第一层模型，先运行该部分模型（下图），得到训练好的模型 1。标签列为 log 后的 sale_quantity。由于在平台上无法自动实现超参数寻优，故没有怎么改动 GBDT 中的超参数，基本上是使用默认值。

字段设置 | 参数设置 | 执行调优

regression loss

gbrank与regression loss中的指数底数 [1,10]

1

metric类型

NDCG

树数量 [1,10000]

1000

学习速率 (0-1)

0.05

最大叶子数 [1,1000]

32

一棵树的最大深度 [1,100]

10

叶子节点容纳的最少样本数 [1,1000]

500

样本采样比例 (0,1)

0.6

训练中采集的特征比例 (0,1)

0.6

测试样本数比例 [0,1)

0.1

随机数产生器种子 [0,10]

8

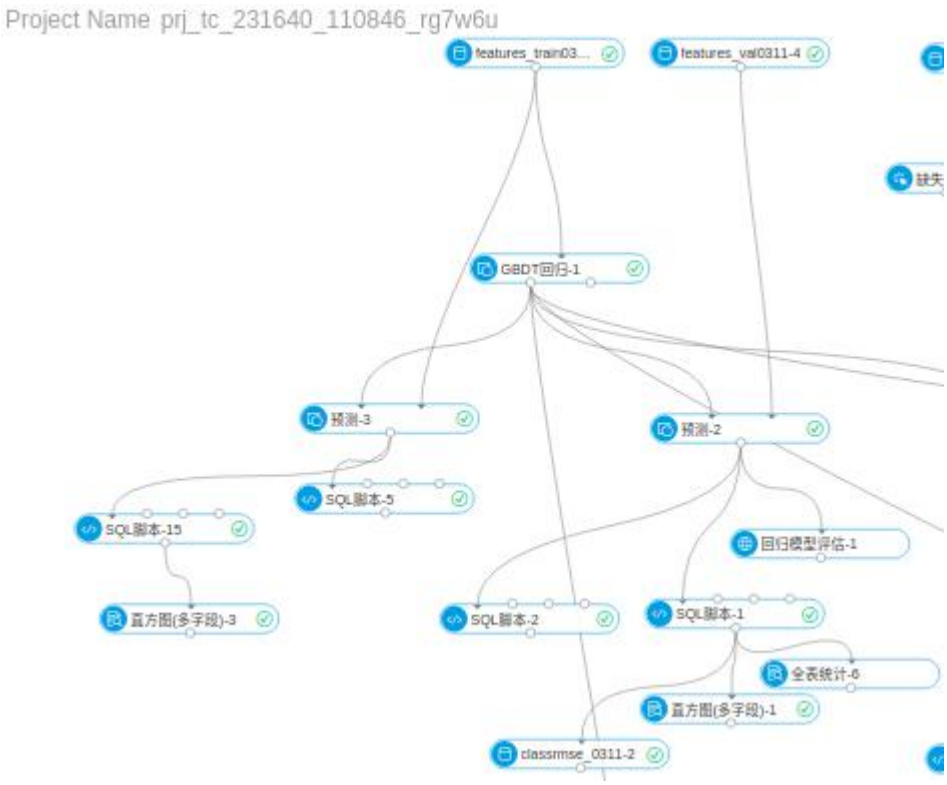
是否使用newton方法来学习

使用

一个特征分裂的最大数量 [1,1000]

500

GBDT回归



其中一个重要输出是右下角的“直方图（多字段）-1”，它的输入为 SQL 脚本-1 的输出，sql 内容如下

```
select class_id as class, city_id as city, log(2, current_sale+1) -log(2, (case when
exp(prediction_result)<0 then 0 else exp(prediction_result)
end)+1)/count(distinct(city_id)) as metrics from ${t1}
group by class_id, city_id, current_sale, prediction_result;
```

即对验证集上的结果进行计算，这里和线上评价函数不同之处在于去掉了平方和开根号，目的是为了保留正负号。结果存入 classrmse_0311 表。

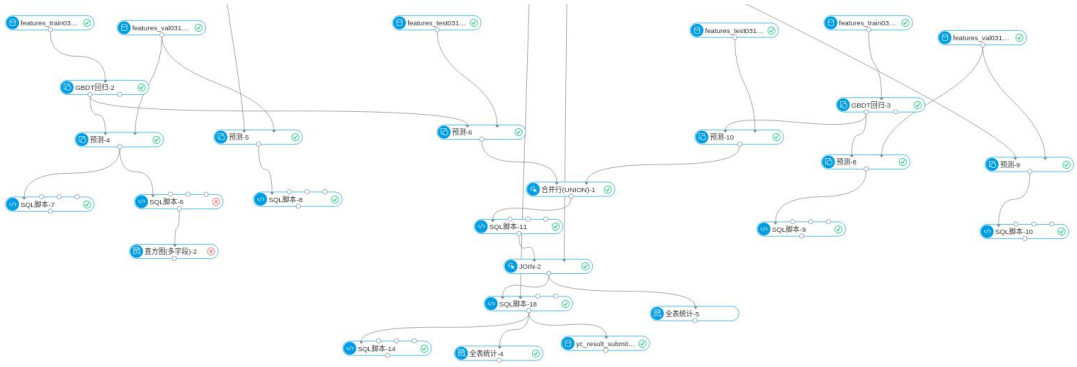
随后根据 classrmse_0311 的结果，对验证集上结果进行分类，具体 sql 代码如下：

```
--metrics 大于 1 的一类
drop table if exists features_train0311_grosser1;
create table features_train0311_grosser1 as
select *
from features_train0311 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics >1;

drop table if exists features_val0311_grosser1;
create table features_val0311_grosser1 as
select *
from features_val0311 f, classrmse_0311 c
```

```
where f.class_id=c.class
and f.city_id=c.city
and c.metrics>1;


drop table if exists features_test0311_grosser1;
create table features_test0311_grosser1 as
select *
from features_test0311 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics>1;

-----metrics 小于-0.5 的一类
drop table if exists features_train0311_kleinerminus05;
create table features_train0311_kleinerminus05 as
select *
from features_train0311 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics <-0.5;

drop table if exists features_val0311_kleinerminus05;
create table features_val0311_kleinerminus05 as
select *
from features_val0311 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics<-0.5;

drop table if exists features_test0311_kleinerminus05;
create table features_test0311_kleinerminus05 as
select *
from features_test0311 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics<-0.5;
```

其他不在这两类的不再次进行单独训练。将数据集进一步切分完成后，继续在 PAI 中训练模型，如下图所示。其中最左部分为分类到“metrics 小于-0.5”的训练和验证数据，模型超参数保持不变，重新训练，得到模型 2。最右部分为分类到“metrics 大于 1”的训练和验证数据，模型超参数保持不变，重新训练，得到模型 3。中间部分分别使用模型 2 预测“metrics 小于-0.5”的测试数据，使用模型 3 预测“metrics 大于 1”的测试数据。再将两个结果和模型 1 预测所有测试数据的结果相结合（异常值处理、UNION、JOIN）。得到 201801 的模

型预测结果（即表 pai_temp_112858_1272110_1）.



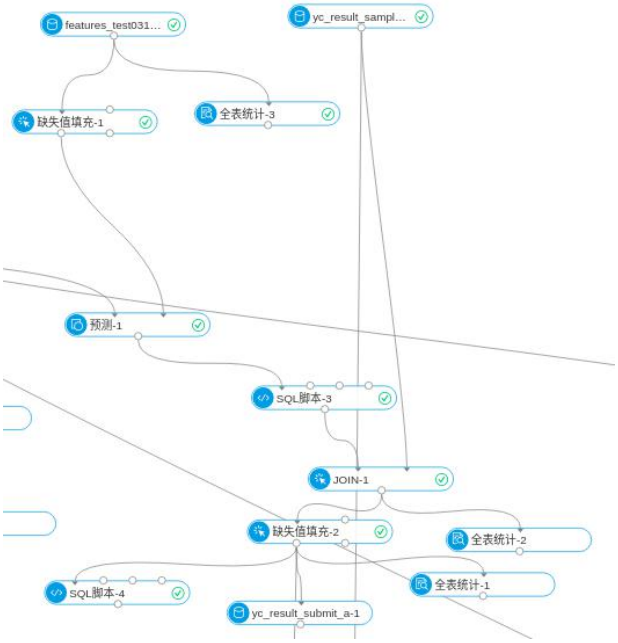
“SQL 脚本-11”中代码如下：

```
select province_id, class_id, city_id, exp(case when prediction_result>10 then 10 else prediction_result end) as predict_quantity from ${t1};
```

“SQL 脚本-18”中代码如下：

```
select t2.predict_date, t2.province_id, t2.class_id, t2.city_id, (case when t1.predict_quantity is Null then t2.predict_quantity else t1.predict_quantity end) as predict_quantity from pai_temp_112858_1271716_1 t1, pai_temp_112858_1271679_1 t2
where t1.province_id=t2.province_id
and t1.city_id=t2.city_id
and t1.class_id=t2.class_id;
```

使用模型 1 在所有数据上进行预测的部分如下图所示：



5. 修正 201801 预测结果：由于模型在 201801 上预测的总量大于历史上的统计值，因此需要对上一步模型的结果进行修正，在这里系数是 1/1.3。

提交后线上成绩 0.82。

create table yc_result_submit_a as select predict_date, province_id, city_id, class_id, predict_quantity/1.3 as predict_quantity from pai_temp_112858_1272110_1;

以下为对复赛 201802 进行预测，步骤和上述类似：

6. 将 201801 预测结果加入数据中：

ALTER table yc_result_submit_a rename to result_0_8159;

----------将预测出的一月销量更新进入训练数据

```
drop table if exists ts_new;
create table ts_new as
  select * from ts union select predict_date as sale_date, province_id, city_id, class_id, ceil(predict_quantity) as sale_quantity from result_0_8159;

drop table if exists data_full;
create table data_full as
  select a.sale_date, c.city_id, d.class_id
  from (select distinct sale_date, 1 as raoguo from ts_new) a
  join (select distinct city_id, 1 as raoguo from ts_new) c
  join (select distinct class_id, 1 as raoguo from ts_new) d
  on a.raoguo = c.raoguo and a.raoguo=d.raoguo ;

drop table if exists p2c;
create table p2c as select distinct province_id, city_id from ts_new;

drop table if exists haha;
create table haha as
    select data_full.sale_date, p2c.province_id, data_full.city_id, data_full.class_id, (case when ts_new.sale_quantity is NULL then 0 else ts_new.sale_quantity end) as sale_quantity
    from ts_new right JOIN data_full
    ON ts_new.city_id=data_full.city_id AND ts_new.class_id=data_full.class_id AND ts_new.sale_date=data_full.sale_date,
    p2c where p2c.city_id=data_full.city_id;


drop table if exists class_full;
create table class_full as
  select a.sale_date, d.class_id
  from (select distinct sale_date, 1 as raoguo from ts_new) a
```

```
  join (select distinct class_id, 1 as raoguo from ts_new) d
  on a.raoguo=d.raoguo ;

drop table if exists produce_full;
create table produce_full as
    select class_full.sale_date as produce_date, class_full.class_id, (case when y.produce_quantity is NULL then 0 else y.produce_quantity end) as produce_quantity
    from yc_passenger_car_yields y right join class_full
    on y.class_id =class_full.class_id and y.produce_date=class_full.sale_date;
```

7. 更新特征并划分训练、验证、测试（注：最初笔者划分 2017 一整年作为验证集检验模型，但其实也没有调超参，只是看了下会不会过拟合。在最终模型训练时用的训练集是 201801 之前的数据）

```
----------更新特征
--训练集特征
drop table if exists temp0312_1;
create table temp0312_1 as
select z.sale_date, z.province_id, z.city_id, z.class_id, bigint (substr(z.sale_date, 1, 4)) as year, bigint(substr(z.sale_date, 5, 6)) as month, sin(bigint(substr(z.sale_date, 5, 6))) as sin_month,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale,
    sum(yc_passenger_car_sales.sale_quantity) as current_sale,
    log(sum(yc_passenger_car_sales.sale_quantity)) as log_current_sale
from yc_passenger_car_sales,
    (select sale_date, province_id, city_id, class_id, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) as last_month_sale,
    lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_2month_sale,
    lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_3month_sale,
    lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_4month_sale,
    lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_5month_sale,
    lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id
```

```
ORDER BY sale_date) as last_6month_sale,
    lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_7month_sale,
    lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_8month_sale,
    lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_9month_sale,
    lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_10month_sale,
    lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_11month_sale,
    lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_year_sale,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/2 as last_2m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/3 as last_3m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/6 as last_6m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id,
```

```
city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date))/12 as last_12m_avg,
    ordinal(2, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)) as median_3month_sale,
    ordinal(3, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)
    , lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)) as median_5month_sale
    from haha) z
    where z.sale_date<'201801'
    and z.sale_date=yc_passenger_car_sales.sale_date
    and z.province_id=yc_passenger_car_sales.province_id
    and z.city_id=yc_passenger_car_sales.city_id
    and z.class_id=yc_passenger_car_sales.class_id
group by z.sale_date, z.province_id, z.city_id, z.class_id,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale;


drop table if exists temp0312_2;
create table temp0312_2 as
select t.sale_date, t.province_id, t.city_id, t.class_id, t.year, t.month, t.sin_month,
    t.last_month_sale, t.last_2month_sale, t.last_3month_sale, t.last_4month_sale, t.last_5month_sale, t.last_6month_sale, t.last_7month_sale, t.last_8month_sale, t.last_9month_sale, t.last_10month_sale, t.last_11month_sale,
    t.last_year_sale, t.last_2m_avg, t.last_3m_avg, t.last_6m_avg, t.last_12m_avg,
    t.median_3month_sale, t.median_5month_sale,
    produce_full.produce_quantity,
    holiday.holidays, holiday.important, holiday.spring_month,
```

```
holiday.nd_month, holiday.labor_month, holiday.work_days,
    t.current_sale,
    t.log_current_sale
from produce_full, holiday,
    temp0312_1 t
    where t.sale_date<'201801'
    and t.sale_date=produce_full.produce_date
    and t.sale_date=holiday.holiday_date
    and t.class_id=produce_full.class_id;

drop table if exists features_train0312;
ALTER table temp0312_2   rename to features_train0312;

--验证集特征
--FEATURES VAL SET：以下为使用 2017 一整年作为 validation，若使用，则需要在构建训练集时修改时间范围。若只是为复现模型结果，可不执行下列代码构建该验证集。
drop table if exists features_val0312_2;
--
drop table if exists temp0312_1_2;
create table temp0312_1_2 as
select z.sale_date, z.province_id, z.city_id, z.class_id, bigint (substr(z.sale_date, 1, 4)) as year, bigint(substr(z.sale_date, 5, 6)) as month, sin(bigint(substr(z.sale_date, 5, 6))) as sin_month,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale,
    sum(yc_passenger_car_sales.sale_quantity) as current_sale,
    log(sum(yc_passenger_car_sales.sale_quantity)) as log_current_sale
from yc_passenger_car_sales,
    (select sale_date, province_id, city_id, class_id, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) as last_month_sale,
    lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_2month_sale,
    lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_3month_sale,
    lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_4month_sale,
    lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_5month_sale,
```

```
    lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_6month_sale,
    lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_7month_sale,
    lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_8month_sale,
    lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_9month_sale,
    lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_10month_sale,
    lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_11month_sale,
    lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_year_sale,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/2 as last_2m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/3 as last_3m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/6 as last_6m_avg,
    (lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) + lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id
```

```
ORDER BY sale_date)+lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 12, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date))/12 as last_12m_avg,
    ordinal(2, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)) as median_3month_sale,
    ordinal(3, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)
    , lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)) as median_5month_sale
    from haha) z
    where z.sale_date>'201612'
    and z.sale_date=yc_passenger_car_sales.sale_date
    and z.province_id=yc_passenger_car_sales.province_id
    and z.city_id=yc_passenger_car_sales.city_id
    and z.class_id=yc_passenger_car_sales.class_id
group by z.sale_date, z.province_id, z.city_id, z.class_id,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale;


drop table if exists temp0312_2_2;
create table temp0312_2_2 as
select t.sale_date, t.province_id, t.city_id, t.class_id, t.year, t.month, t.sin_month,
    t.last_month_sale, t.last_2month_sale, t.last_3month_sale, t.last_4month_sale, t.last_5month_sale, t.last_6month_sale, t.last_7month_sale, t.last_8month_sale, t.last_9month_sale, t.last_10month_sale, t.last_11month_sale,
    t.last_year_sale, t.last_2m_avg, t.last_3m_avg, t.last_6m_avg, t.last_12m_avg,
    t.median_3month_sale, t.median_5month_sale,
    produce_full.produce_quantity,
```

```
    holiday.holidays, holiday.important, holiday.spring_month, holiday.nd_month, holiday.labor_month, holiday.work_days,
    t.current_sale,
    t.log_current_sale
from produce_full, holiday,
    temp0312_1_2 t
    where t.sale_date>'201612'
    and t.sale_date=produce_full.produce_date
    and t.sale_date=holiday.holiday_date
    and t.class_id=produce_full.class_id;

drop table if exists features_val0312_2;
ALTER table temp0312_2_2  rename to features_val0312_2;

--测试集特征
-- FEATURES TEST SET
drop table if exists features_test0312 ;

create table features_test0312 as
select b.predict_date, z.province_id, z.city_id, z.class_id, bigint(substr(b.predict_date, 1, 4)) as year, bigint(substr(b.predict_date, 5, 6)) as month, sin(bigint(substr(b.predict_date, 5, 6))) as sin_month,
    z.last_month_sale, z.last_2month_sale, z.last_3month_sale, z.last_4month_sale, z.last_5month_sale, z.last_6month_sale, z.last_7month_sale, z.last_8month_sale, z.last_9month_sale, z.last_10month_sale, z.last_11month_sale,
    z.last_year_sale, z.last_2m_avg, z.last_3m_avg, z.last_6m_avg, z.last_12m_avg,
    z.median_3month_sale, z.median_5month_sale,
    BIGINT(produce_full.produce_quantity/2) as produce_quantity,
    holiday.holidays, holiday.important, holiday.spring_month, holiday.nd_month, holiday.labor_month, holiday.work_days
from yc_result_sample_b b, produce_full, holiday,
    (select sale_date, province_id, city_id, class_id, sale_quantity as last_month_sale,
    lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_2month_sale,
    lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_3month_sale,
    lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_4month_sale,
    lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_5month_sale,
    lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id
```

```
ORDER BY sale_date) as last_6month_sale,
    lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_7month_sale,
    lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_8month_sale,
    lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_9month_sale,
    lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_10month_sale,
    lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_11month_sale,
    lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date) as last_year_sale,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/2 as last_2m_avg,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/3 as last_3m_avg,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/6 as last_6m_avg,
    (sale_quantity + lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date) +lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 5, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 6, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 7, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 8, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    +lag(sale_quantity, 9, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date)+lag(sale_quantity, 10, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)+lag(sale_quantity, 11, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date))/12 as last_12m_avg,
    ordinal(2, sale_quantity, lag(sale_quantity, 1, 0) OVER(PARTITION BY
```

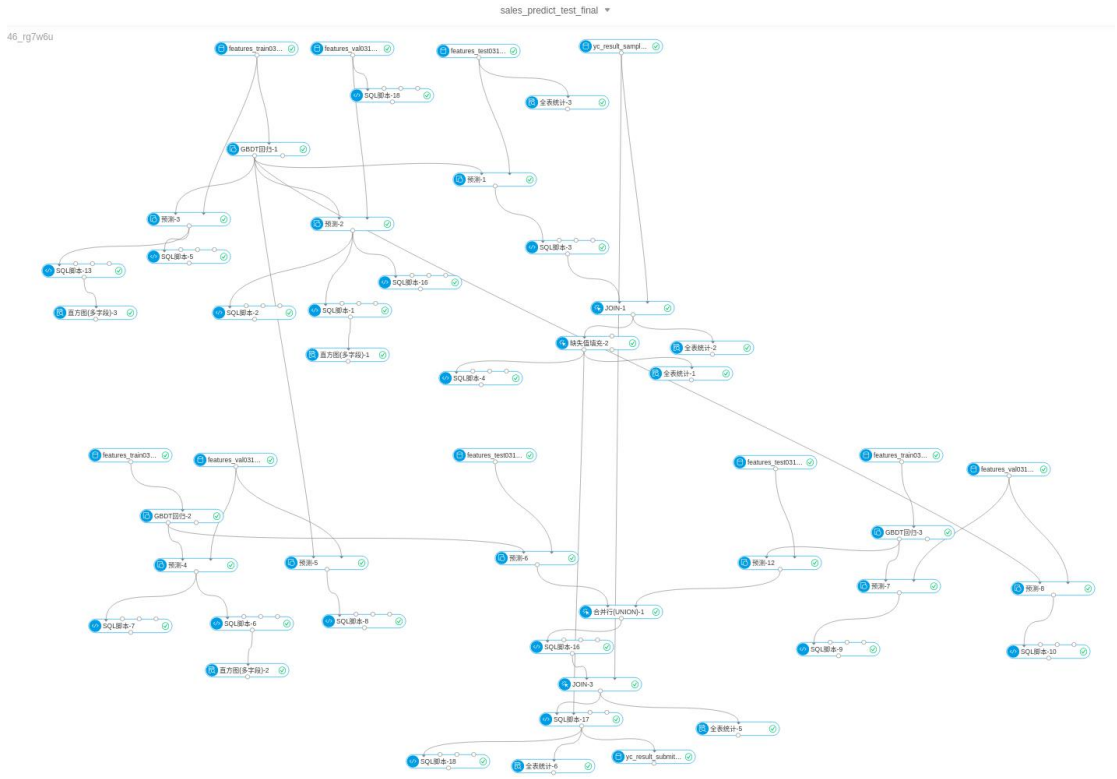
```
class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)) as median_3month_sale,
    ordinal(3, sale_quantity, lag(sale_quantity, 1, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date), lag(sale_quantity, 2, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)
    , lag(sale_quantity, 3, 0) OVER(PARTITION BY class_id, city_id, province_id ORDER BY sale_date), lag(sale_quantity, 4, 0) OVER(PARTITION BY class_id, city_id, province_id  ORDER BY sale_date)) as median_5month_sale
    from haha) z
    where z.sale_date='201801'
    and produce_full.produce_date='201712'
    and z.sale_date=holiday.holiday_date
    and z.class_id=produce_full.class_id
    and z.province_id=b.province_id
    and z.city_id=b.city_id
    and z.class_id=b.class_id;
```

8. 使用复赛 A 榜得出的分类结果进一步划分数据：

```
--
drop table if exists features_train0312_grosser1;
create table features_train0312_grosser1 as
select *
from features_train0312 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics >1;

drop table if exists features_val0312_grosser1;
create table features_val0312_grosser1 as
select *
from features_val0312 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics>1;


drop table if exists features_test0312_grosser1;
create table features_test0312_grosser1 as
select *
from features_test0312 f, classrmse_0311 c
where f.class_id=c.class
```

```
and f.city_id=c.city
and c.metrics>1;

-----
drop table if exists features_train0312_kleinerminus05;
create table features_train0312_kleinerminus05 as
select *
from features_train0312 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics <-0.5;

drop table if exists features_val0312_kleinerminus05;
create table features_val0312_kleinerminus05 as
select *
from features_val0312 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics<-0.5;

drop table if exists features_test0312_kleinerminus05;
create table features_test0312_kleinerminus05 as
select *
from features_test0312 f, classrmse_0311 c
where f.class_id=c.class
and f.city_id=c.city
and c.metrics<-0.5;
```

9. 构建模型并预测（模型和复赛 A 榜预测 201801 相同，只更新了输入数据），全图如下：



10. 对模型预测结果进行修正，由于没有更新 201801 月的信息，故仍使用 A 榜计算出的模型修正系数。得到最终预测结果：提交后线上成绩 0.93

```
create table yc_result_submit_b as
    select predict_date, province_id, city_id, class_id, (case when predict_quantity/1.3<1 then 1 else predict_quantity/1.3 end) as predict_quantity from pai_temp_113052_1273449_1;
```

写在最后：

复赛阶段大家都只能用 sql 和 PAI，其实限制了复杂的算法和特征，因此我想只要按照机器学习的 pipeline 做，大家的结果都不会差别特别大。比较勤快的大神应该能手动提取很多复杂的特征、做更多数据挖掘以对数据进行分类训练、做多模型融合。可能还有很多我没有想到的方法。总之，希望以后的平台赛能开放更多资源和更强大的模型。