**机器学习纳米学位**

##毕业项目 李榕 优达学城

2019年9月11日

**I. 问题的定义**

**项目概述**

本项目旨在为零售商Rossmann（劳诗曼）解决其销售额预测的问题，即利用过去已经发生的店铺的销量信息、店铺的类别、打折、 对手相关信息，对店铺近期未来的数据进行准确预测。

准确预测出销售额分析影响因素，对于零售商整体店铺布局，门店物品准备，改变打折策略以达到提升最终销售额的目的起着至关重要的作用。

为解决此问题，分析现有的一些数据，与每个店铺相关的信息包括：

|  |  |
| --- | --- |
| **列名** | **列含义** |
| store | 商店ID |
| storetype | 商店类型 |
| Assortment | 商店分类 |
| CompetitionDistance | 竞争者的距离 |
| CompetitionOpenSinceMonth | 竞争者开启的月份 |
| CompetitionOpenSinceYear | 竞争者开启的年份 |
| Promo2 | 是否有广告投放 |
| Promo2SinceWeek | 广告从那一周开始投放 |
| Promo2SinceYear | 广告从哪一年开始投放 |
| PromoInterval | 广告投放月份周期 |

每天生成的销量数据包括如下信息：

|  |  |
| --- | --- |
| **列名** | **列含义** |
| store | 商店ID |
| dayofweek | 周几 |
| date | 日期 |
| sales | 销量 |
| customer | 顾客量 |
| open | 店铺是否开门 |
| promo | 是否有促销 |
| StateHoliday | 日期状态 |
| SchoolHoliday | 是否是假日 |

本项目将根据如上已经发生的历史信息，来预测未来近期，每日每个店铺的销量（sales）的数据

**问题陈述**

本课课程需要解决依据Rossmann各个商店13年2月到15年7月之间的销售额数据，结合商店类型、商店的竞争者位置、是否是节假日、当天是星期几等综合因素，来对Rossmann15年8月到15年9月之间的销售额进行预测。据此，问题可以被拆解为如下几点：

1、店铺的销售额与店铺的品类是不是有关系，有怎样的影响？

2、店铺周围是否有竞争者，对销售额是否有影响，有怎样的影响？

3、店铺竞争者的位置远近，对销售额是否有影响，有怎样的影响？

4、当天的日期是正常工作日还是星期日，对销售额是否有影响，有怎样影响

5、当天是否是公共假期，对销售额是否有影响，有怎样影响

6、当天日期是否是学校假日，对销售额是否有影响，有怎样影响

7、当天是是否发生促销，对销售额是否有影响

8、当天是否发生了季节性促销，对销售额的影响怎样

9、明确了对销售额的影响因素之后，进行特征工程的构建，对空值进行删除补充

10、考虑采用回归模型来进行不同店铺销量的预测。

11、评估预测的效果如何

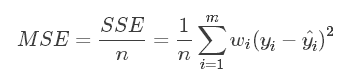
期望得到结果：

通过对train.csv数据进行训练，得到合适模型，模型好坏通过对预测值与真实值之间的差距来衡量；将模型应用于测试数据中（test.csv文件）得到对应商店每一日的销量数据

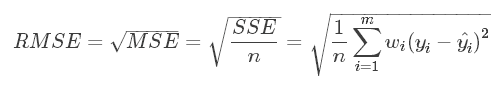
**评价指标**

本文处理的问题为回归问题，回归问题通过分析预测值与真实值之间的差距，来评估模型的好坏，常用的评价标准：

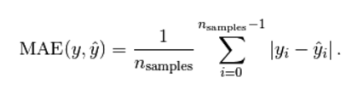
1）mse：均方误差——越接近于0越好



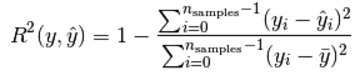
2）rmse：均方根误差——越接近于0越好



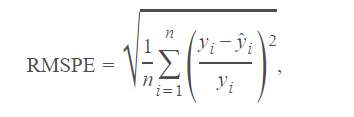
3）mae：平均绝对误差——越接近于0越好



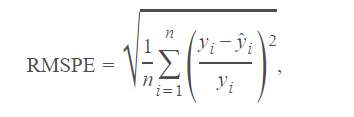
4）R2：决定系数——越接近1越好



**5）RMSPE——均方根百分比误差，为项目要求使用的评估方法。**



了解常用方法之后，本方案选择：**RMSPE**——均方根误差百分比。



yi 代表某店铺某天真实销量

i 代表对应店铺对应日期的预测销量

n代表总的记录数

指标整体衡量了预测值与真实值偏差的百分比均值。

与rmse（均方根误差）进行比较，均方根误差是预测值与真实值偏差的平方和和观测次数n比值的平方根，反映了预测数据偏离真实值的程度，即反应预测精密度，即该值越小，模型精密度更高。

Rmspe与rmse不同在于，求取的是预测值与真实值之间的变化百分比平方和，整个评价函数思想是使得预测值与真实值得变化率最小化，而非绝对值最小化。更加适用于本个项目的原因是，本个项目是对于所有的店铺进行总体的预测，店铺中的销量有的基数很大，有的基数则很小，对于基数很大的店铺，其实真实值和预测值之间的差距可以接受的差值的容忍度要高于那些基数很小的店铺，使用相对的误差可以从一定程度上实现不同问题区别对待，以达到整体的最优。

本项目需要得到的最终评估标准是以RMSPE来衡量，因而指标评估方案选择为RMSPE。

**II. 分析**

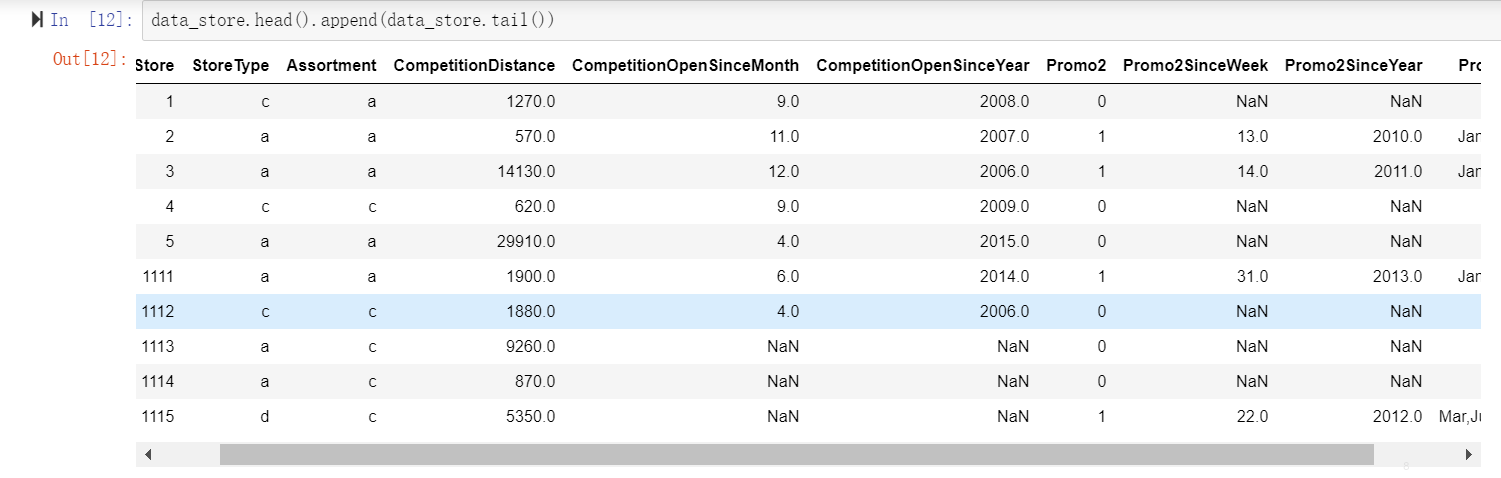
**数据的探索**

#### 1 项目给定表内容分析

本项目共包含三个表文件：

1. store.csv 文件内容为每个店铺的说明情况：

数据情况如下：



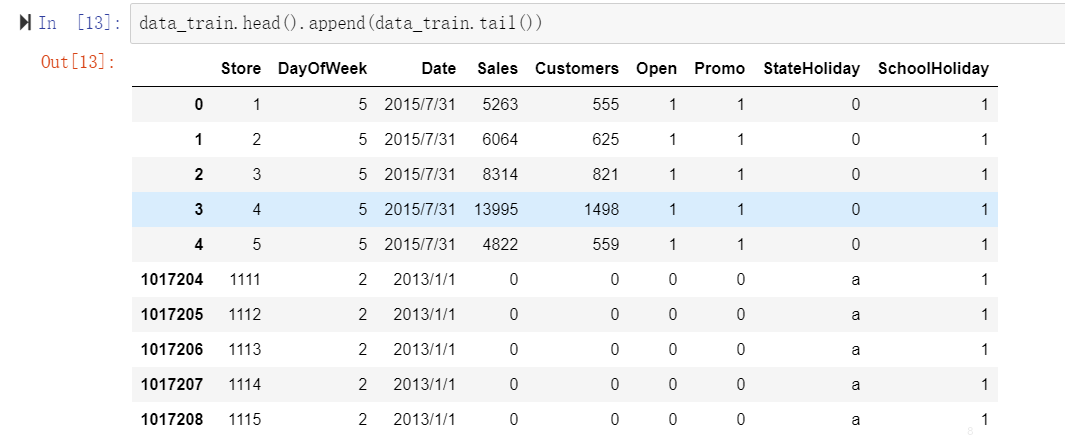
行情况：共1115行，代表1115个店铺，主键为Store

列情况：共10列，分析代表含义如下：

|  |  |
| --- | --- |
| **列名** | **列含义** |
| store | 商店ID |
| storetype | 商店类型 |
| Assortment | 商店分类 |
| CompetitionDistance | 竞争者的距离 |
| CompetitionOpenSinceMonth | 竞争者开启的月份 |
| CompetitionOpenSinceYear | 竞争者开启的年份 |
| Promo2 | 是否有广告投放 |
| Promo2SinceWeek | 广告从那一周开始投放 |
| Promo2SinceYear | 广告从哪一年开始投放 |
| PromoInterval | 广告投放月份周期 |

1. train.csv 文件内容为用于模型训练的数据

数据情况如下：

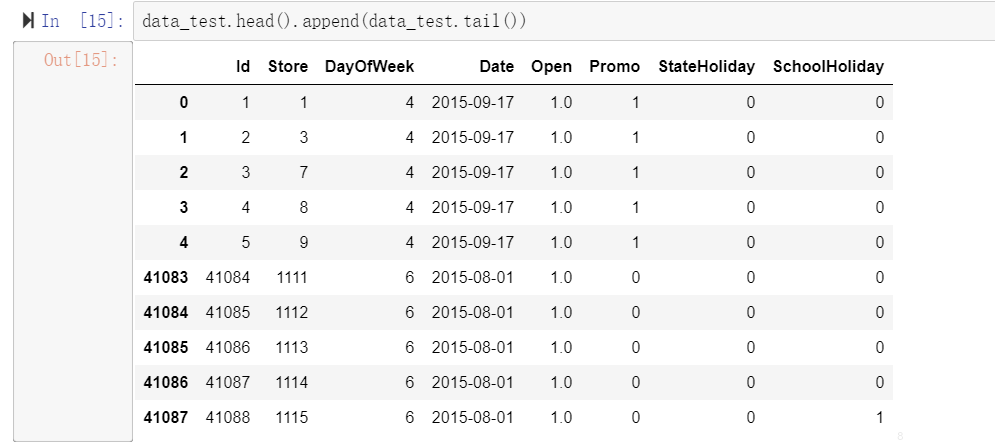


行情况：共计1017209行，代表从13年1月1日到15年7月31日的1115家店铺，每日的数据情况

列情况：共计列，分析各列代表含义如下：

|  |  |
| --- | --- |
| **列名** | **列含义** |
| store | 商店ID |
| dayofweek | 周几 |
| date | 日期 |
| sales | 销量 |
| customer | 顾客量 |
| open | 店铺是否开门 |
| promo | 是否有促销 |
| StateHoliday | 日期状态 |
| SchoolHoliday | 是否是假日 |

1. test.csv 文件内容为用于预测的结果数据



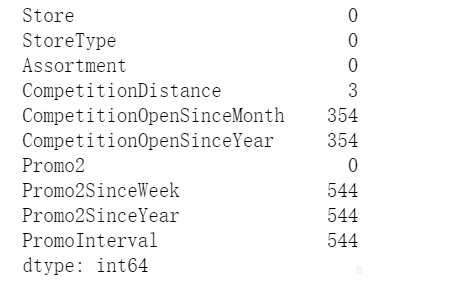
行情况：共计41088行，代表从15年8月1日到15年9月17日的店铺ID、促销信息、节假日信息等情况

列情况：

|  |  |
| --- | --- |
| **列名** | **列含义** |
| store | 商店ID |
| dayofweek | 周几 |
| date | 日期 |
| open | 店铺是否开门 |
| promo | 是否有促销 |
| StateHoliday | 日期状态 |
| SchoolHoliday | 是否是假日 |

#### 2 表内容缺省值分析

1）store.csv缺省值分析



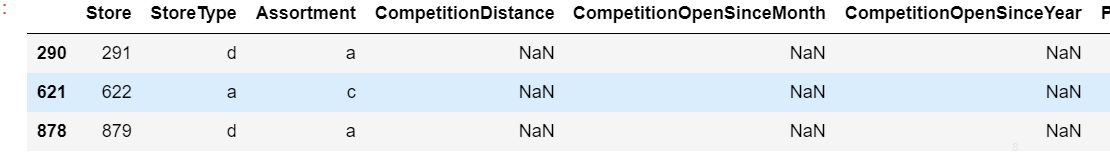
与竞争者相关的字段：

CompetitionDistance

CompetitionOpenSinceMonth

CompetitionOpenSinceYear

字段有空值，进一步分析

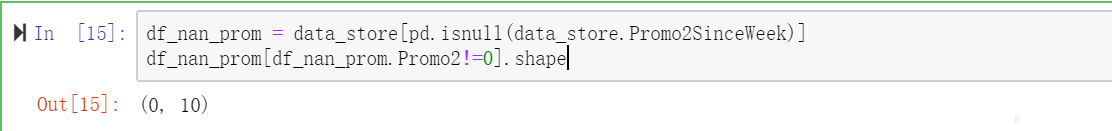


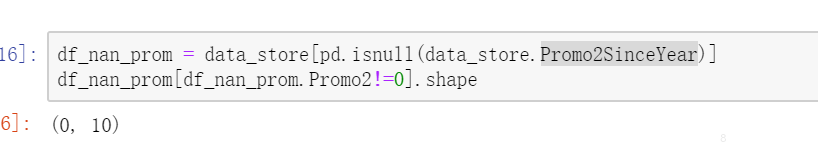
CompetitionDistance字段为空时，对应的CompetitionOpenSinceMonth，CompetitionOpenSinceYear字段同时为空，认定为店铺可能不存在竞争者，这里考虑把对应的值写为0

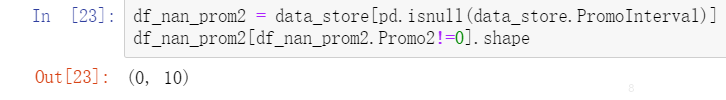
进一步观察CompetitionOpenSinceMonth，CompetitionOpenSinceYear，两者为空的列是一致的，认定为竞争者开启的年份和月份是未知的，考虑将对应的两列写为0

与促销相关字段

Promo2SinceWeek，Promo2SinceYear，Promo2Interval字段，均是当无促销时出现该情况，因而认为当日无促销

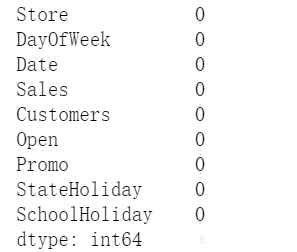






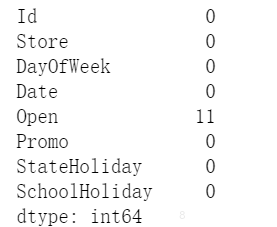
处理时，将对应字段填为0

2）train.csv缺省值分析

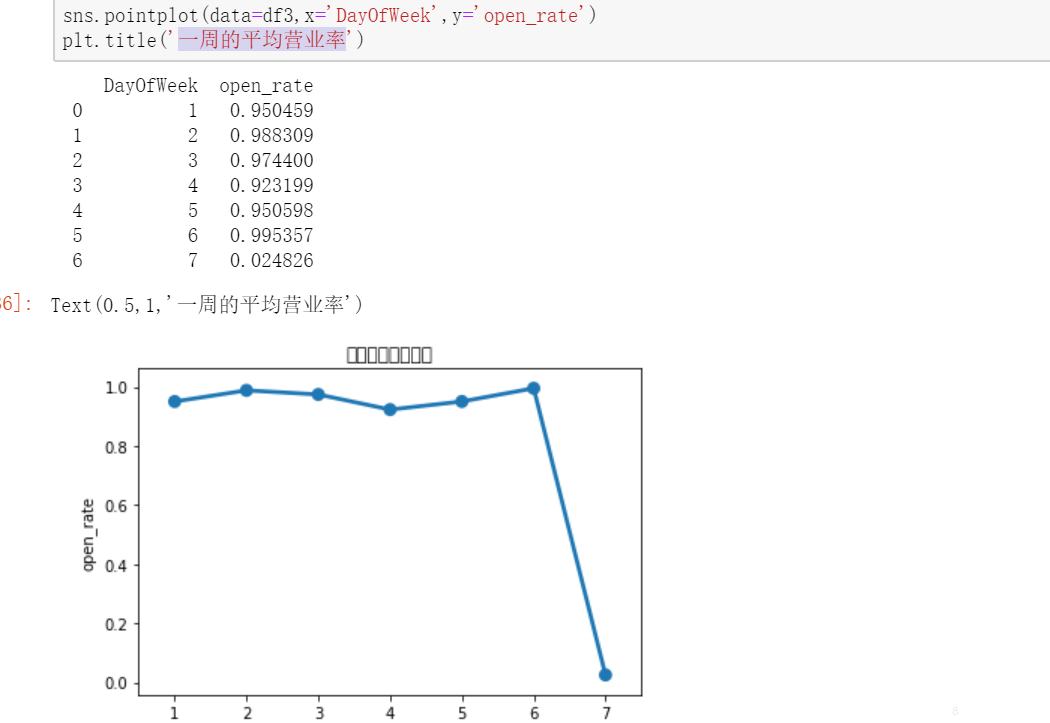


本文件中无缺省值

3）test.csv缺省值分析

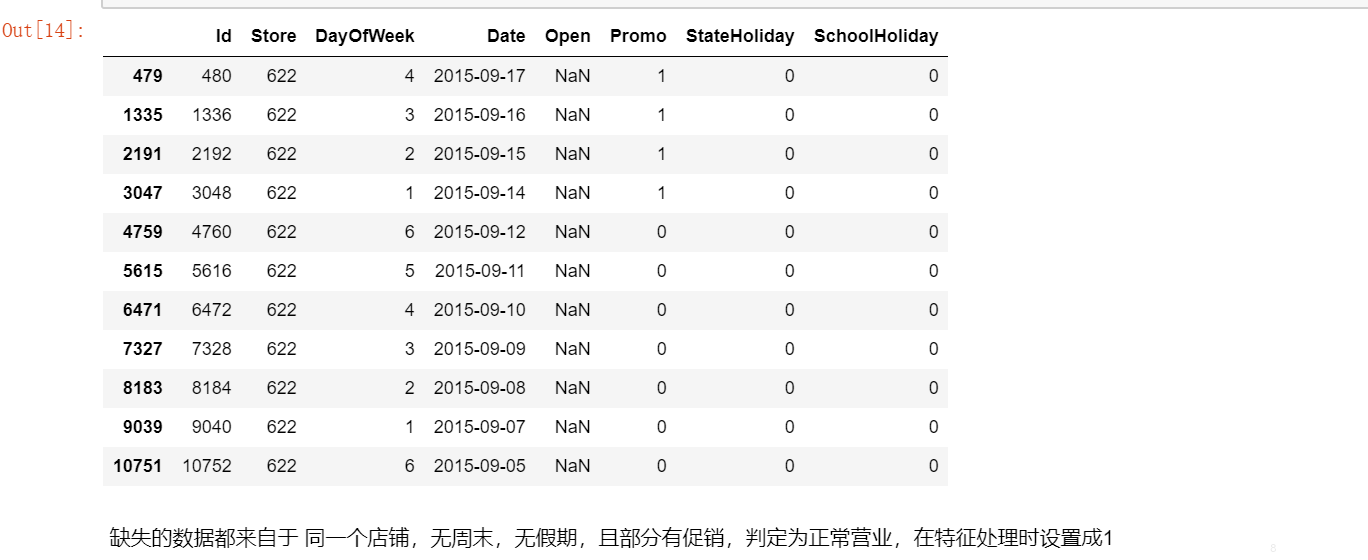


缺省值为open字段，代表商店是否开着。通过常识分析，商店是否开，可能与周几，节假日关系较大，进一步对train.csv文件中的DayofWeek字段的Open分布进行分析，得到下图：



即 星期为周日时，仅2%的店铺是营业状态。

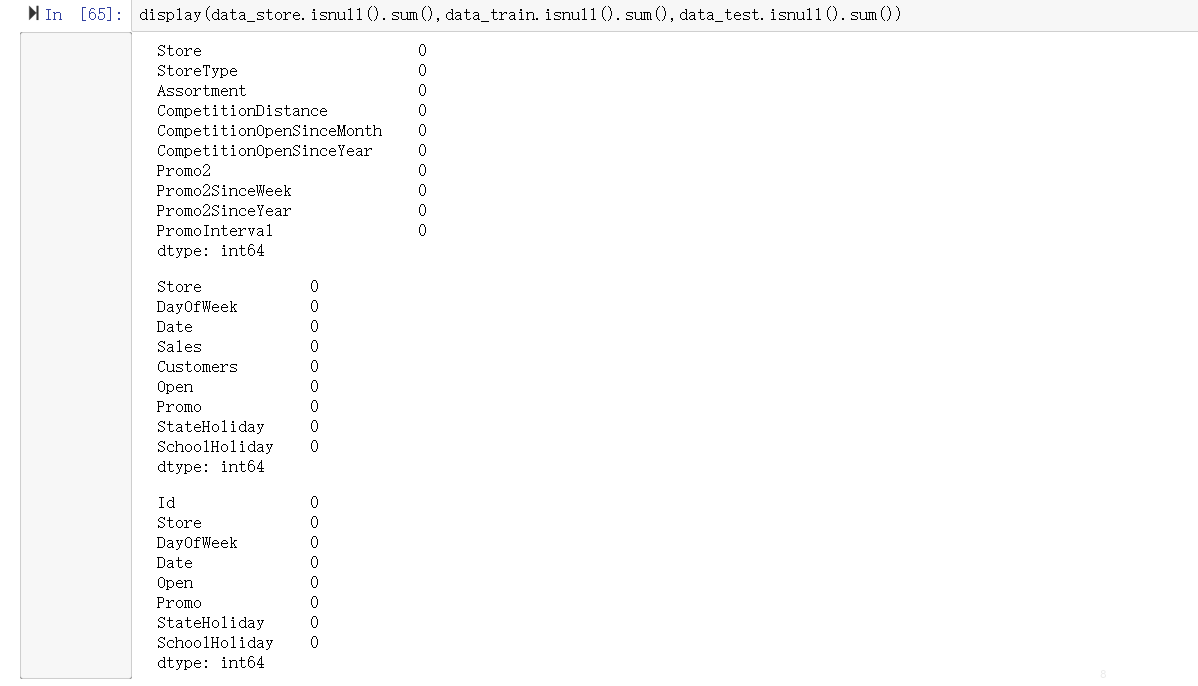
观察test.csv 中open字段缺失所对应的Dayofweek列



未出现周末的情况，因而直接将test.csv 文件中的Open为空的字段写为1

综上，对几个文件的缺省值进行填充，并判定，填充之后的数值是否仍有空值





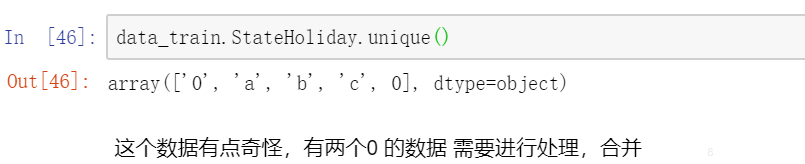
#### 3 表内容异常值分析

1）store.csv

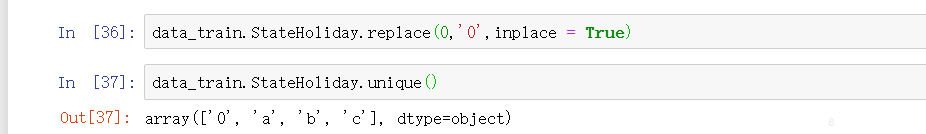
文件中，查看无明显异常值的情况，但是，StoreType, Assortment, PromoInterval为非数值表示，需要在特征处理时，进行数值转化

2）train.csv

明显异常集中在 StateHoliday字段，出现了一个数值0，和一个字符0，需要进行统一的数值化处理



这里将StateHoliday字段中 数值0统一转换为字符’0’



3）test.csv

未发现明显的数值异常情况，

#### 4 数据集整体概览

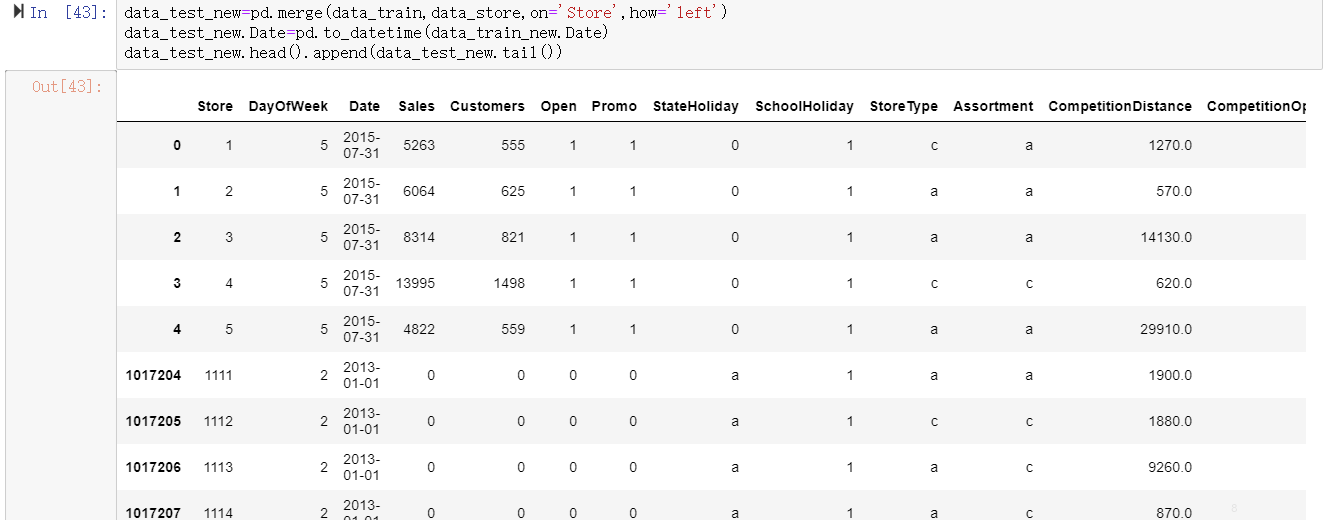
如前所述，train.csv集合中，共计1017209行，代表从13年1月1日到15年7月31日的1115家店铺，每日的数据情况。

Test.csv共计41088行，代表从15年8月1日到15年9月17日的店铺ID、促销信息、节假日信息等情况

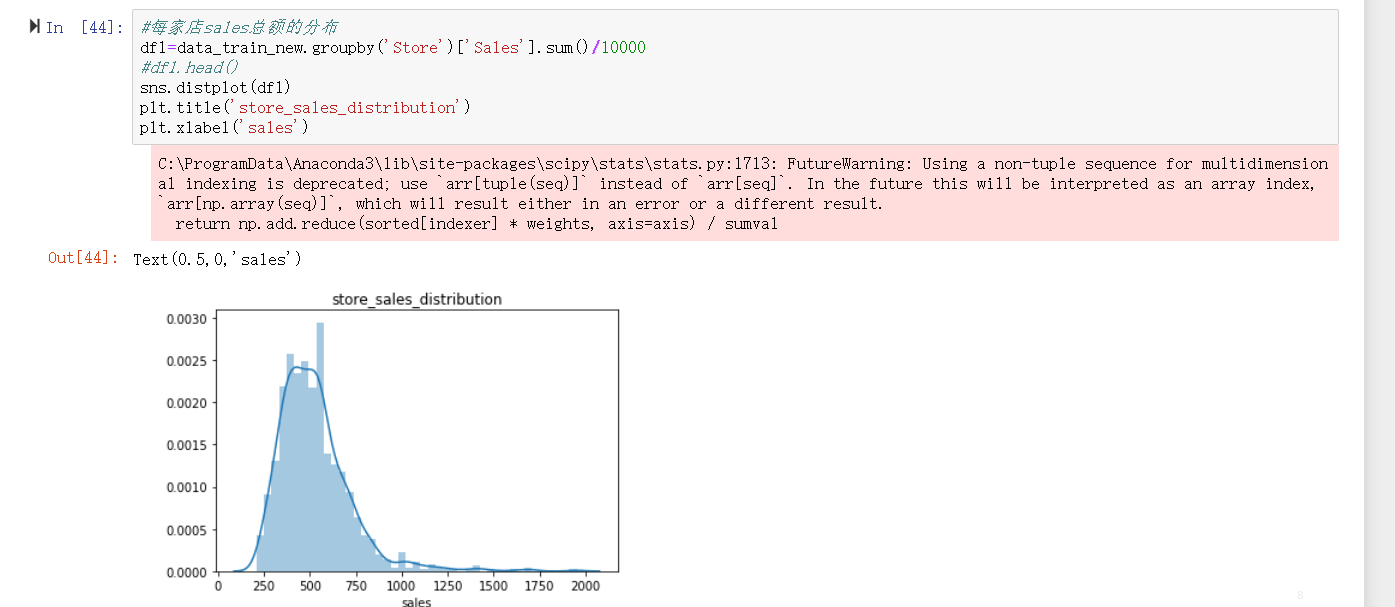
**探索性可视化**

分析的目的是需要将各类的变量对销量的数据产生的影响进行分析，因而分别将train.csv ,test.csv 与store.csv 进行连接，得到新的数据集方便进行下一步分析。



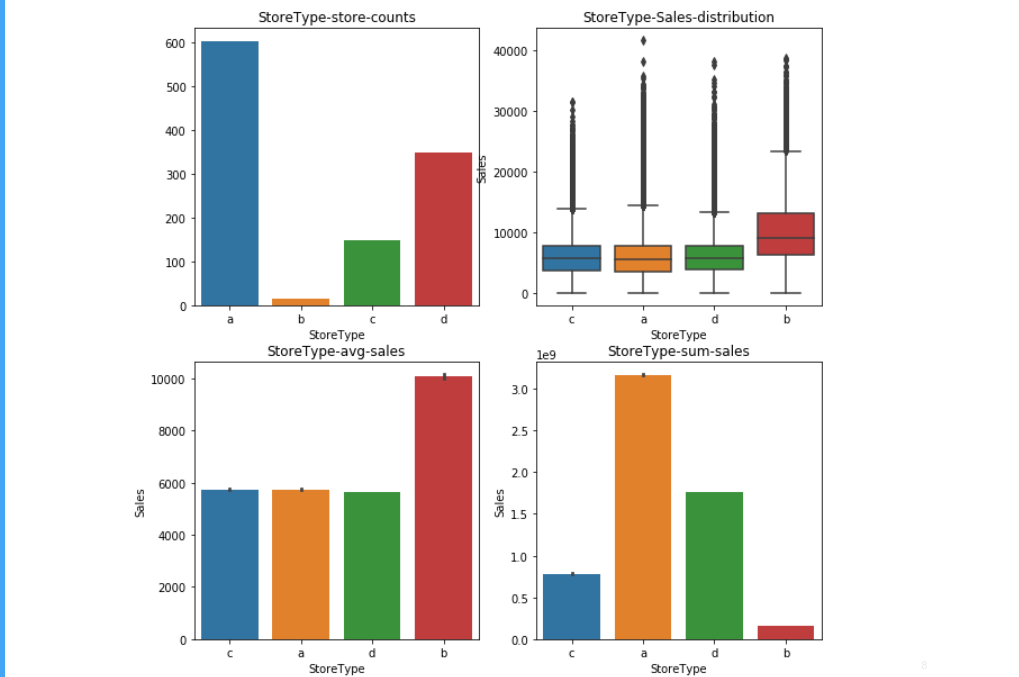


#### 1 店铺整体销量的频率分布情况：



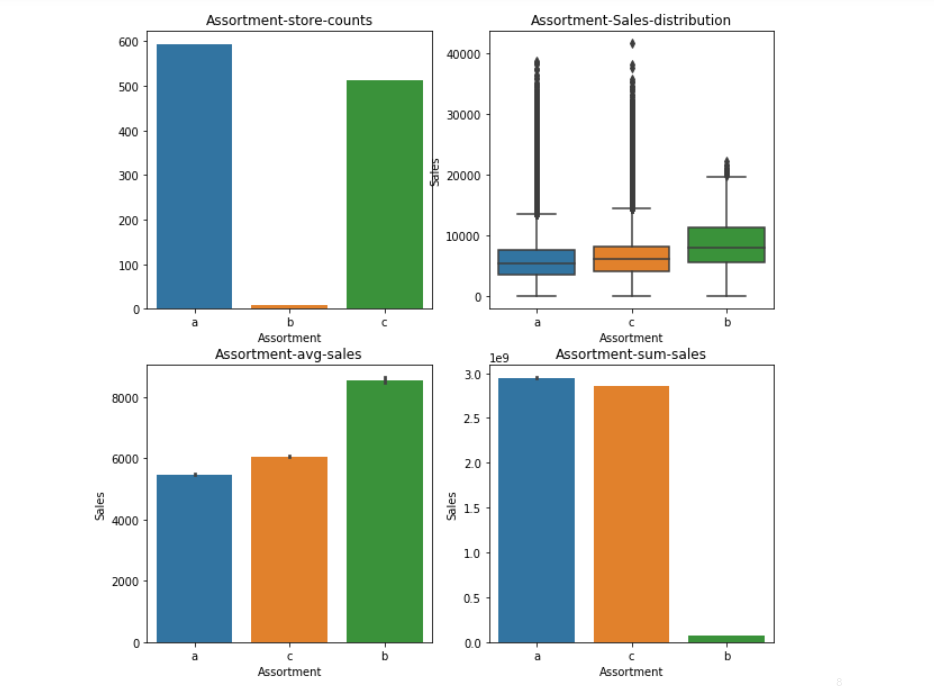
绝大部分的店铺销量销量低于1000W，销量在1000W以上的店铺很少。

#### 2 店铺的不同类型以及对应类型的销量情况



1. 从店铺数量的分布来看，a店铺的数量最多，其次为d类，b类店铺的数量最少
2. 从销量分布来看，a,c,d店铺的销量分布情况相对接近，b类店铺的销量整体分布偏高
3. 从日均销量对比来看，b类店铺的日均销量最高，a,b,c 相对接近
4. 从总销量来对比，与店铺的数量分布类似，a店铺带来的销量最多，其次为d，b销量最少

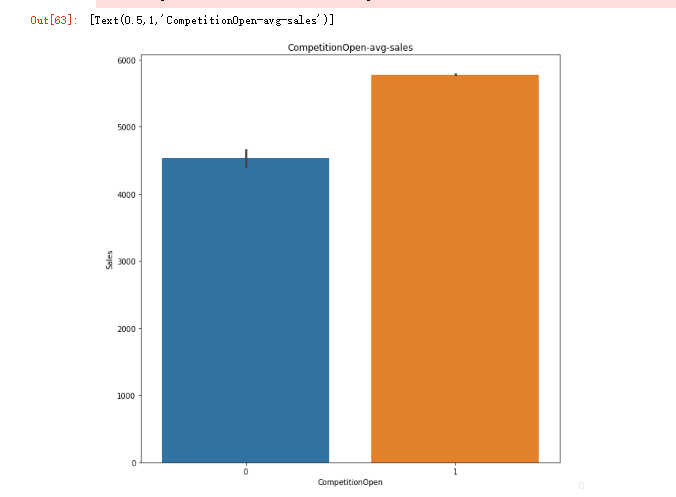
#### 3 不同分类级别（Assortment）情况



1. 店铺数量对比来看，a类店铺最多，c类次之，b类店铺最少，即基本店铺和扩展型店铺较多，额外店铺少
2. 从销量分布来看，b类店铺的销量分布整体比a，c更高
3. 从平均销量来看，b类店铺平均销量最高
4. 店铺的总销量与店铺的总量规律一致

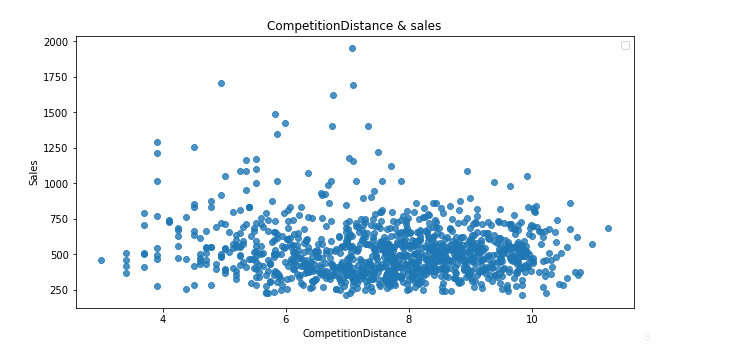
#### 4 与竞争对手的关系

1. 是否有竞争对手



有竞争对手平均销量会稍高些

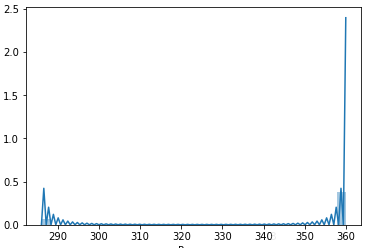
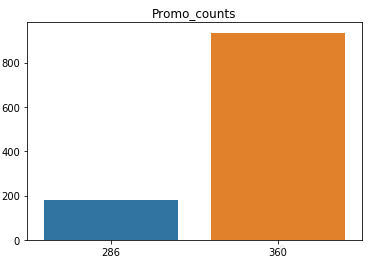
1. 竞争对手距离与平均销量的关系



竞争对手距离变大或者变小，对销量的影响并不大

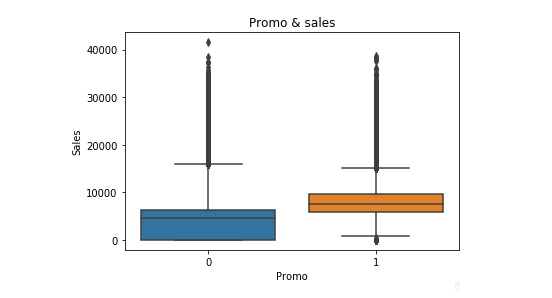
#### 5 与促销的关系

1）促销情况的频率分布

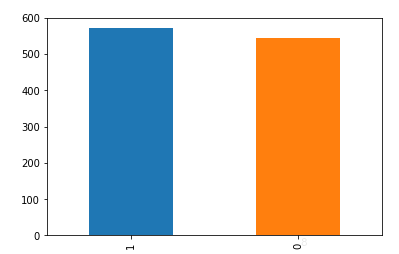
整体促销次数的频率分布图如上。

1. 促销对销量的影响



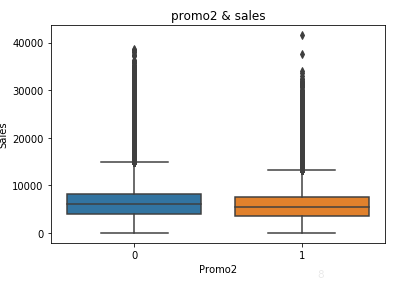
促销之后整体的销量要好于不促销的情况

1. 季节性促销的次数



季节性促销与不促销次数基本接近

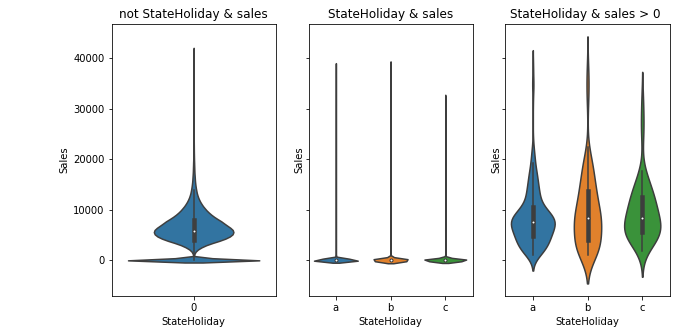
1. 季节性促销与销量的影响



季节性促销与否与销量的影响关系不大

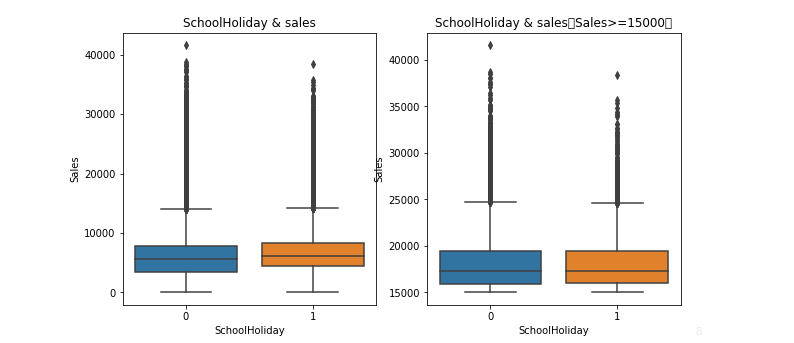
#### 6 探索法定假日、周末、学校假日对销量的影响

1. 法定假日的影响：



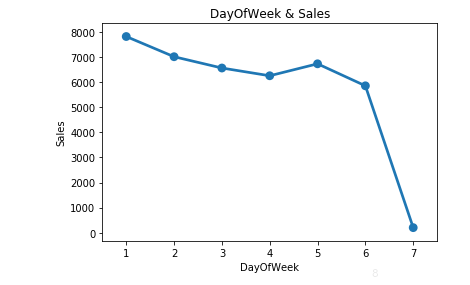
法定假日时，销量很低，说明大部分商店不营业

1. 学校假日对销量的影响

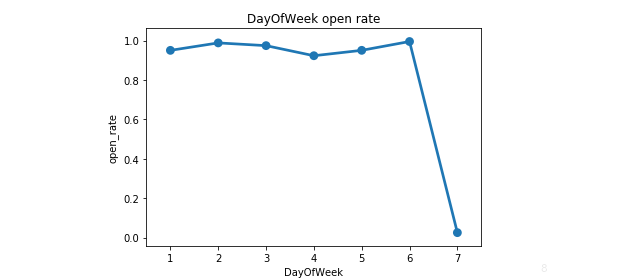


学校的假日基本，整体会比无假日稍高，但高销量数据中差异不明显，说明学校的假日影响到的群体仅有学生这一小部分群体，且学生的消费水平并不高。

1. 星期与销量的关系

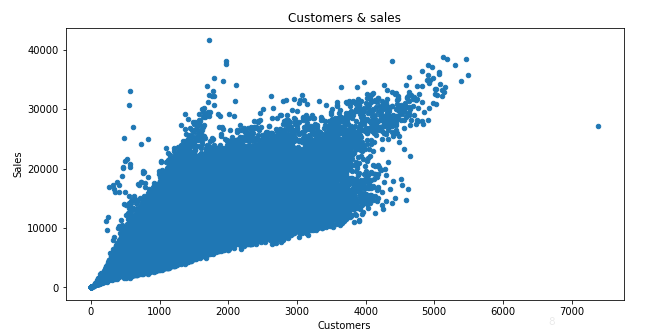


从周一到周日，日均销量逐渐下滑，周日比周一到周六明显降低。



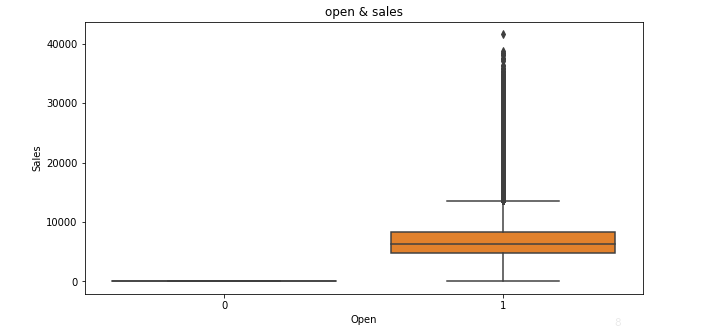
在看店铺的打开率，周日98%店铺不开门，无销量

#### 7 顾客量与销量的关系



顾客量与销量呈现明显得相关性，不难理解，顾客量多自然买东西的人多，但由于在测试的数据中没有顾客量，因而该数据无法作为特征，输入模型训练

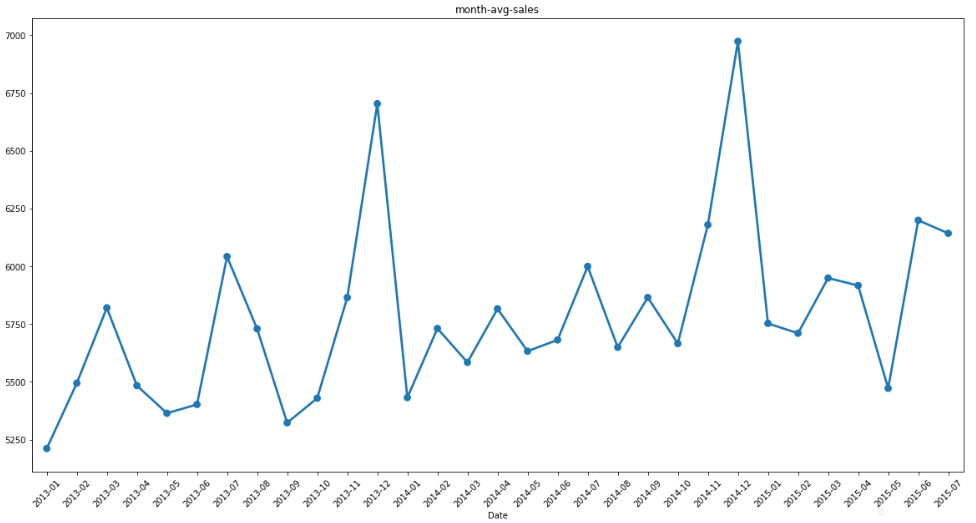
#### 8 店铺是否开启与销量关系



与预想的结果进行相互的验证，确实如果店铺不开那么就是没有销量，符合我们的尝试。

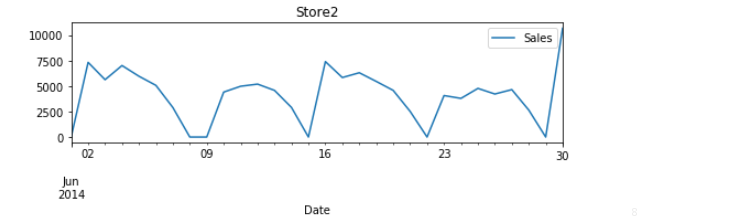
#### 9 探索销量随时间的变化规律

1. 每个月份的平均销量



每年的年底是高峰，之后回落

1. 选取某个月份的情况



可以发现每月的每周规律类似

**算法和技术**

#### 1 相关技术

本问题是一个回归问题，因而考虑采用回归模型来求解：

1）**线性回归，**通过线性回归构造出来的函数一般称之为了线性回归模型。

C:\Users\Administrator\AppData\Local\YNote\data\weixinobU7Vjomn6r2tiNT6zdH5ZvbZEhI\628d8d90f2584e349b8fc70dd3257a8a\p=49&f=jpeg.jpeg

2）**回归树**，通过构建决策树来获取到区间均价，分裂点方法与分裂问题不同，分类问题采用的是，信息增益或信息增益率 基尼系数等方法；回归树采用最小化误差平方的方式，从特征变量中，找到一个变量j，找到一个取值，这个取值能够使得按照取值划分之后的 两个集合，对于目标变量的误差平方最小；接着依次遍历所有的变量，得到特征空间的划分。然后依次进行划分，最终得到子向量空间，向量空间的均值就是预测值。

3）**随机森林**

　　1. 假如有N个样本，则有放回的随机选择N个样本(每次随机选择一个样本，然后返回继续选择)。这选择好了的N个样本用来训练一个决策树，作为决策树根节点处的样本。

　　2. 当每个样本有M个属性时，在决策树的每个节点需要分裂时，随机从这M个属性中选取出m个属性，满足条件m << M。然后从这m个属性中采用某种策略（比如说信息增益）来选择1个属性作为该节点的分裂属性。

　　3. 决策树形成过程中每个节点都要按照步骤2来分裂（很容易理解，如果下一次该节点选出来的那一个属性是刚刚其父节点分裂时用过的属性，则该节点已经达到了叶子节点，无须继续分裂了）。一直到不能够再分裂为止。注意整个决策树形成过程中没有进行剪枝。

4. 按照步骤1~3建立大量的决策树，这样就构成了随机森林了。

4）**xgboost**

xgboost由多个相关联的树联合。

xgboost不断对残差进行预测，xgboost每个决策树是逐一被添加进入。下一棵决策树加入的输入，会取决于前一棵决策树产生的残差。

单个树生成方式：

4.1 不断便利特征和特征的取值，然后进行分类，计算loss function最小值，然后再选择一个特征分裂，又得到一个损失函数的最小值，找到分裂效果最好的（即分裂前后损失函数变化最大）的特征进行分裂

4.2 持续4.1的过程，不断进行分裂

4.3 如下条件停止分裂：

a）分裂的增益小于某个阈值

b）分裂达到最大深度时，停止，最大深度理论上应当是是一个超参数

c) 样本权重之和，小于设定阈值时，停止分裂

4.4 计算前一棵树的残差数据，然后将 目标函数定义为 残差，继续输入到模型中，进行重复的模型构建，直到触发 ：增益小于阈值，树的数量小于一定量，样本的权重之和小于设定的阈值，停止整个过程。

#### 2 方案选择详述

本方案选择采用xgboost方法作为算法。

1. 代码实现的xgb相关参数说明

xgboost.train(

params,

dtrain,

num\_boost\_round=10,

evals=(),

obj=None,

feval=None,

maximize=False,

early\_stopping\_rounds=None,

evals\_result=None,

verbose\_eval=True,

learning\_rates=None,

xgb\_model=None)

params 这是一个字典，里面包含着训练中的参数关键字和对应的值。

params = {"objective": "reg:linear",

"booster" : "gbtree",

"eta": 0.03,

"max\_depth": 10,

"subsample": 0.9,

"colsample\_bytree": 0.7,

"silent": 1,

"seed": 10

}

本项目解决的问题是回归问题，针对此，可考虑的参数设置如下：

1. objective [缺省值=reg:linear]

“reg:linear” – 线性回归

“reg:logistic” – 逻辑回归

“binary:logistic” – 二分类逻辑回归，输出为概率

“binary:logitraw” – 二分类逻辑回归，输出的结果为wTx

“count:poisson” – 计数问题的poisson回归，输出结果为poisson分布。在poisson回归中，max\_delta\_step的缺省值为0.7 (used to safeguard optimization)

“multi:softmax” – 设置 XGBoost 使用softmax目标函数做多分类，需要设置参数num\_class（类别个数）

“multi:softprob” – 如同softmax，但是输出结果为ndata\*nclass的向量，其中的值是每个数据分为每个类的概率。

由于本文的问题是回归问题因而使用 reg:linear

1. **seed** [缺省值=0]  
   随机数的种子 设置它可以复现随机数据的结果，也可以用于调整参数
2. **silent** [缺省值=0]  
   设置为0打印运行信息；设置为1静默模式，不打印
3. **booster** [缺省值=gbtree]  
   决定那个使用那个booster，可以是gbtree，gblinear或者dart。 gbtree和dart使用基于树的模型，而gblinear 使用线性函数。

**本项目基于树模型构建，使用缺省值即可，如下内容针对 gbtree进行说明**

1. **eta** [缺省值=0.3，别名：learning\_rate]  
   更新中减少的步长来防止过拟合。在每次boosting之后，可以直接获得新的特征权值，这样可以使得boosting更加鲁棒。  
   范围： [0,1]
2. gamma [缺省值=0，别名: min\_split\_loss]（分裂最小loss）

在节点分裂时，只有分裂后损失函数的值下降了，才会分裂这个节点。Gamma指定了节点分裂所需的最小损失函数下降值。 这个参数的值越大，算法越保守。这个参数的值和损失函数息息相关，所以是需要调整的。

范围: [0,∞]

1. **max\_depth** [缺省值=6]  
   这个值为树的最大深度。 这个值也是用来避免过拟合的。max\_depth越大，模型会学到更具体更局部的样本。设置为0代表没有限制  
   范围: [0,∞]
2. min\_child\_weight [缺省值=1]

决定最小叶子节点样本权重和。XGBoost的这个参数是最小样本权重的和，而GBM参数是最小样本总数。 这个参数用于避免过拟合。当它的值较大时，可以避免模型学习到局部的特殊样本。 但是如果这个值过高，会导致欠拟合。这个参数需要使用CV来调整。.

范围: [0,∞]

1. subsample [缺省值=1]

这个参数控制对于每棵树，随机采样的比例。 减小这个参数的值，算法会更加保守，避免过拟合。但是，如果这个值设置得过小，它可能会导致欠拟合。 典型值：0.5-1，0.5代表平均采样，防止过拟合.

范围: (0,1]

1. colsample\_bytree [缺省值=1]

用来控制每棵随机采样的列数的占比(每一列是一个特征)。 典型值：0.5-1

范围: (0,1]

1. colsample\_bylevel [缺省值=1]

用来控制树的每一级的每一次分裂，对列数的采样的占比。 我个人一般不太用这个参数，因为subsample参数和colsample\_bytree参数可以起到相同的作用。但是如果感兴趣，可以挖掘这个参数更多的用处。

范围: (0,1]

1. lambda [缺省值=1，别名: reg\_lambda]

权重的L2正则化项。(和Ridge regression类似)。 这个参数是用来控制XGBoost的正则化部分的。虽然大部分数据科学家很少用到这个参数，但是这个参数在减少过拟合上还是可以挖掘出更多用处的。.

1. alpha [缺省值=0，别名: reg\_alpha]

权重的L1正则化项。(和Lasso regression类似)。 可以应用在很高维度的情况下，使得算法的速度更快。

scale\_pos\_weight[缺省值=1]

在各类别样本十分不平衡时，把这个参数设定为一个正值，可以使算法更快收敛。通常可以将其设置为负样本的数目与正样本数目的比值。

dtrain 训练的数据

num\_boost\_round 提升迭代次数，即生成多少基模型

evals 这是一个列表，用于对训练过程中进行评估列表中的元素。形式是evals = [(dtrain,’train’),(dval,’val’)]或者是evals = [(dtrain,’train’)],对于第一种情况，它使得我们可以在训练过程中观察验证集的效果。

**obj**：自定义目的函数

**feval**：自定义评估函数，在本文中评估函数是RMSPE，需要自定义一个RMSPE的函数传入此参数中

**maximize**：是否对评估函数进行最大化

early\_stopping\_rounds：早期停止次数 ，假设为100，验证集的误差迭代到一定程度在100次内不能再继续降低，就停止迭代。这要求evals 里至少有 一个元素，如果有多个，按最后一个去执行。返回的是最后的迭代次数（不是最好的）。如果early\_stopping\_rounds存在，则模型会生成三个属性，bst.best\_score，bst.best\_iteration和bst.best\_ntree\_limit

**evals\_result**：字典，存储在watchlist中的元素的评估结果。

**verbose\_eval** ：(可以输入布尔型或数值型)，也要求evals里至少有 一个元素。如果为True,则对evals中元素的评估结果会输出在结果中；如果输入数字，假设为5，则每隔5个迭代输出一次。

**learning\_rates**：每一次提升的学习率的列表，

**xgb\_model**：在训练之前用于加载的xgb model。

对本问题的使用方式如下：

1. params的设置。

Objective的设置无需进行太多考虑，由于本项目处理回归问题，直接设置为：reg:linear

Seeds 随机设置为一个值，保证结果能够复现即可

Booster 本项目采用树模型，这里直接使用默认值：gbtree

**eta** 将尝试多个可能值

gamma 设置为0，因为需要模型在每次节点分裂之后下降的数量等于0 将不再分裂，如果设置更大，算法将会保守

max\_depth 将尝试更多的值，在过拟合与欠拟合之间选择最佳的结果

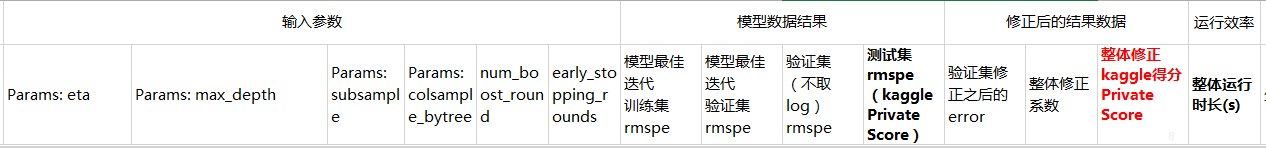
subsample 这里将进行 0.5-1 之间的调整

**colsample\_bytree**  这里将进行0.5-1 之间的调整

其余参数直接使用默认值，不进行调整

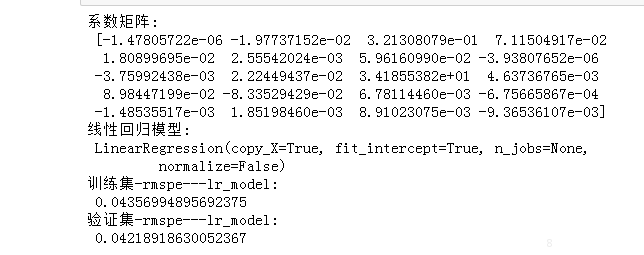
1. dtrain : 将 train.csv 文件的数据分为训练集 与验证集，这里将训练集数据传入
2. num\_boost\_round 这里作为一个可变的数据进行调整
3. evals 将分割的训练数据与验证数据集传入
4. obj 不再自定义目标函数
5. feval 评估函数使用自定义的目标函数，来满足需求
6. early\_stopping\_rounds 这里直接设置为1000
7. 其余参数直接使用默认值

生成的模型，使用在验证集中，计算评估函数的得分，记录在如下表中。



**基准模型**

针对本项目的问题，计划采用线性回归和决策树的方式作为基准模型，同样使用rmspe来对结果进行评估，同时衡量计算的时间，基于此基础来对结果进行不断提升。



生成的结果得到的private score



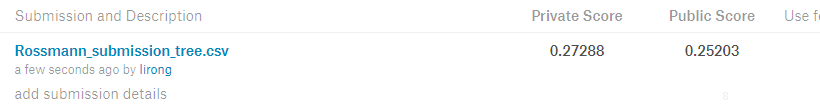
效果并不理想。

再使用回归数模型进行建模



可以看到训练集上rmspe要远远低于在验证集上的结果，说明 模型放到验证集之后 效果打折扣很多。

决策树结果在kaggle的private score



相比较线性回归有所升高，但是距离要求还很远。

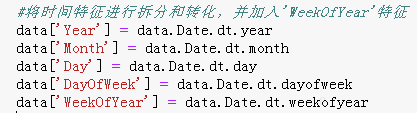
综上，基模型使用了简单的弱模型，泛化能力不足，同时对生成的结果并不进行任何的改进

**III. 方法**

**数据预处理**

构建模型前，需要对数据进行处理，进行特征工程的构建，由于在前期的分析之中，已经对空值和异常值进行了分析与处理，故本部分重点对非数值特征进行处理，同时衍生出新的便于模型处理的特征。

1. StoreType取值为字符，这里进行数值化：
2. Assortment取值为字符，这里同样进行数值化：
3. StateHoliday取值为字符，这里同样进行数值化：
4. 将店铺当天的销售日期是 年-月-日 形式，为了便于处理，同时能够与竞争对手年份、第几周这种变量进行计算，得到时间差距，新增 年份、月份、日期、星期、一年中的第几周的数值特征，



1. 对竞争对手开启的年份和开启的月份进行处理

新增特征：竞争对手已经开始营业的的时间，相对于当前时间月份时间差。

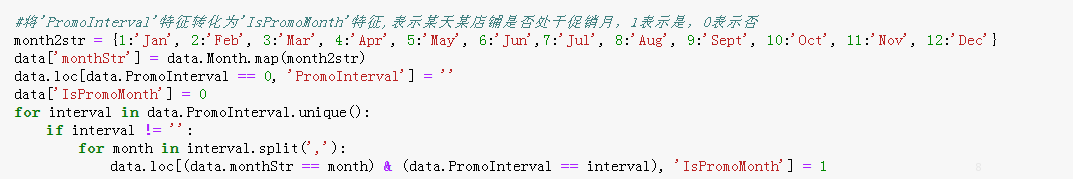


1. 对促销开始的时间进行处理：

新增特征：促销开始的时间相对于当前时间的月份时间差

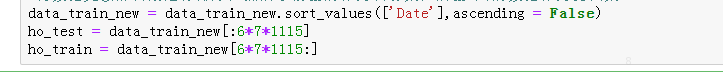


Promointerval进行处理，将Promointerval月份进行拆解，得到分别的月份，构建新的特征（IsPromoMonth）：当前的时间对应的打折月份时，对应值赋为1，否则为0



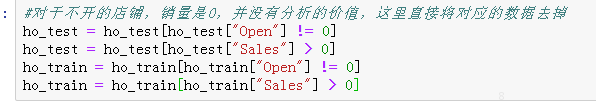
**执行过程**

1. 将data\_train\_new中的数据进行日期升序排列之后，分割为训练集与验证集合：

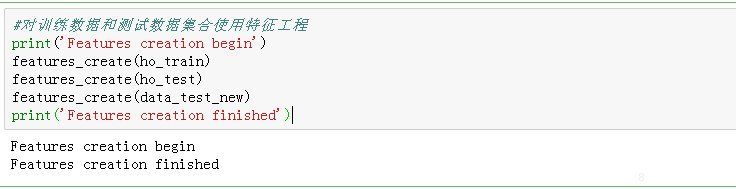


这里没有的测试数据与验证数据进行K折的验证，而是直接选择后6周的数据作为验证集合，前面所有周的数据作为训练集。由于xgboost的运行效率较低

1. 对于不开的店铺以及销量为0的店铺对于数据的分析是无意义的，且基本上周日的时间店铺基本不开，因而这些数据不输入模型：



1. 对分割之后数据进行特征的构建：



1. 删除掉无用的特征

Date特征由于已经转化为 年份，月份，每年几周的特征，因而在这里使用的意义不大

Customers特征由于和Sales强相关，且在需要预测的数据集中不存在对应的特征，因而即便输入模型中未来在进行预测时也无法使用，因而删除

Open特征由于在数据入模型前，已经限定了是开启的店铺，因而Open都为1，对模型无价值

PromoInterval特征由于已经转换为，对应的月份是否有促销的特征，因而可以删除

monthSer特征为中间处理过程中使用，可以删除

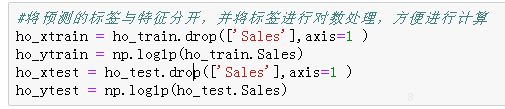


1. 观察特征之间的相关性情况：

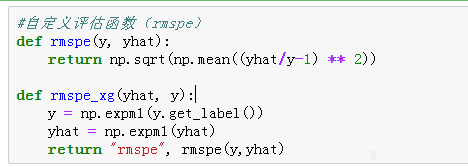


与销量相关性最强的特征为 打折 Promo，以及开启的月份

1. 将预测的标签与特征分开，并将标签进行对数处理



1. 自定义评估函数



1. 定义参数，对不同参数进行输入，然后组合为不同的params列表
2. 参数输入模型，拟合出一个模型列表，然后分别观察模型的结果值
3. 将模型应用于验证集中，得到rmspe结果

参数以及对应的结果记录参见表格excel内容：

《调参结果.xlsx》

**完善**

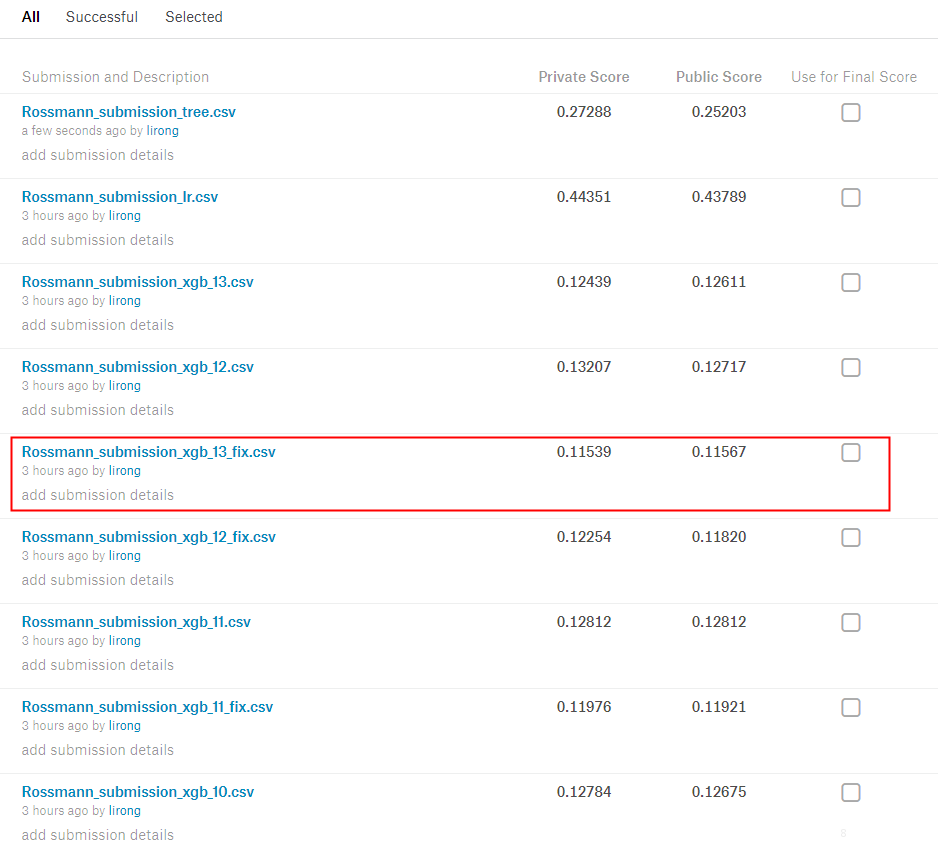
基于模型预测的结果可以考虑对结果进行修正，将基于模型预测之后的值，乘以系数，来对结果进行修正

* 1. 为了能够尽可能地逼近真实值，系数从 0.9900 ~ 1.0 每隔0.5个步长进行修正。
  2. 对预测值进行修正之后y值再次和真实值进行对比，计算rmspe
  3. 得到所有的系数中rmspe最小的值

修正的整体结果参见如下 excel文件：

《调参结果.xlsx》

Kaggle上的结果截图如下：



**IV. 结果**

**模型的评价与验证**

对模型的评估，从几个方面着手：

1. 模型在测试集的得分
2. 模型在训练集的表现
3. 模型的运行效率
4. 模型运行时占用的机器资源情况。

根据模型的结果表现，可以观察如下情况：

1. 本试验中，eta分别设置了0.03,0.005伴随此，分别将**n**um\_boost\_round与early\_stopping\_rounds进行了数量的提升，观察到在两种情况下的测试集和验证集的数据分布情况：

模型在训练集的得分 随eta的变小，得分升高，但是通过对验证集的观察，得知对验证集还是有较好的效果。

学习的速度变慢，更有可能找到全局的最优解，尽管最优解并不一定在训练集上表现好。

训练时长的对比情况

随着步长的变小，训练的时间有了很明显的提升。训练速度非常慢。

1. max\_depth的设置，本实验中，max\_depth分别设置了10,15两个值，但是明显看到的是选择15 模型出现明显得过拟合现象。

训练集的表现随着深度的增加而变好

应用于验证集之后，效果明显变差

树模型的深度增加有利于去学习到更多的细化特征，但是这些特征的覆盖的数据量是很少的一部分数据，因而导致的问题是模型泛化能力不强，出现过拟合的情况。

而对于数据计算效率的影响

由图可见，随着深度的增加，模型学习到的更多的是细化的特征，这将导致模型对于训练集的数据能够较快的达到一个最优解，因而时间也随之降低。但带来的问题就是模型泛化能力差

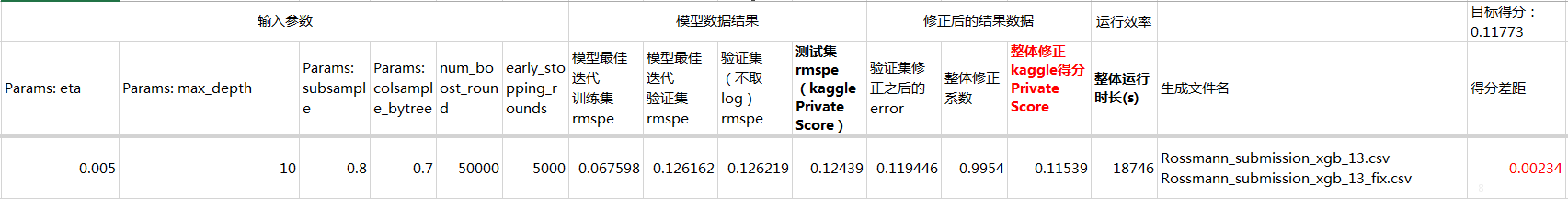
1. 不同subsample对于结果的表现。

Subsample用于控制每棵树随机采样的比例

Subsample设置到0.9之后，每棵树随机采样样本的比例提升，即每棵树的构建都使用了大部分样本，不利于对部分数据进行深入运算，导致模型效果不佳。

计算时间上，无明显规律

1. colsample\_bytree的设置情况，colsample\_bytree用来控制对每一列的采样比例，即特征的采样比例，特征的采样比例越高，出现的过拟合的现象会相对更加严重。
2. 综上，在计算的时间和资源条件允许的情况下，选择了在验证上表现最优的模型。



**合理性分析**

与基准模型相比，xgboost在训练集以及验证集上的表现并不优异，但是由于强模型的泛化能力以及对结果的多次提升能力更好，因而最终的表现更加优秀

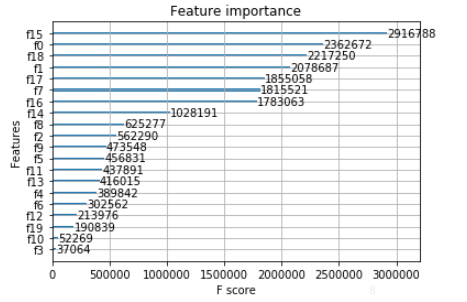
|  |  |  |  |
| --- | --- | --- | --- |
| 模型 | 训练集 rmspe | 验证集 rmspe | kaggle得分 |
| 线性回归 | 0.0435 | 0.0421 | 0.44351 |
| 决策树 | 2.57E-06 | 0.02249 | 0.27288 |
| xgboost | 0.067598 | 0.126219 | 0.11539 |

**V. 项目结论**

**结果可视化**

与销量相关的重要因素如下：





|  |  |  |  |
| --- | --- | --- | --- |
| 特征ID | 特征含义 | 特征权重 | 特征重要度排序 |
| f15 | Day | 2916788 | 1 |
| f0 | Store | 2362672 | 2 |
| f18 | PromoOpen | 2217250 | 3 |
| f1 | DayOfWeek | 2078687 | 4 |
| f17 | CompetitionOpen | 1855058 | 5 |
| f7 | CompetitionDistance | 1815521 | 6 |
| f16 | WeekOfYear | 1783063 | 7 |
| f14 | Month | 1028191 | 8 |
| f8 | CompetitionOpenSinceMonth | 625277 | 9 |
| f2 | Promo | 562290 | 10 |
| f9 | CompetitionOpenSinceYear | 473548 | 11 |
| f5 | StoreType | 456831 | 12 |
| f11 | Promo2SinceWeek | 437891 | 13 |
| f13 | Year | 416015 | 14 |
| f4 | SchoolHoliday | 389842 | 15 |
| f6 | Assortment | 302562 | 16 |
| f12 | Promo2SinceYear | 213976 | 17 |
| f19 | IsPromoMonth | 190839 | 18 |
| f10 | Promo2 | 52269 | 19 |
| f3 | StateHoliday | 37064 | 20 |

重点对TOP8的特征进行阐述：

* + 1. Day，前期的数据分析中可见，销量其实和具体的日期具有较大的关联性，一方面，当某日为日常情况时，会随着当天是周几有明显的规律变化，另一方面如果当天有着促销等活动时也会随着带来明显的影响
    2. Store，每个店铺有自己不同的特性，其实就是促销、竞争对手等等信息在store上的具体表现
    3. PromoOpen，显然与分析结论相一致，有促销时，销量是变多的，绝大部分的人都是对价格敏感的
    4. DayOfWeek，也与分析结果类似，在没有进行打折或者假日的时期，销量每一周的规律是非常一致的。
    5. CompetitionOpen，竞争者虽然在前期的数据分析中的表现并不明显，但是从最终的特征重要程度上来看，竞争对手对销量还是具有明显的影响
    6. CompetitionDistance，这与CompetitionOpen的结论应当是一致的
    7. WeekOfYear，该特征也是综合假日打折等相关因素的综合体现
    8. Month，通常这个因素会结合打折和年底假日等情况表现出来

**对项目的思考**

本项目解决商铺销量预测的回归问题，通过将已知的train.csv 数据进行分析，分析之后进行特征的转换，将转换之后的特征输入于模型中，建模并对结果进行预测。

有趣的地方在于对特征的组合探索，尝试深入的理解数据以及尝试多种特征的组合对目标结果产生的影响。以及最终通过模型的特征重要性反向去验证分析的一些结论。

困难的地方在于对模型参数的调整，xgboost受限于其模型的运行效率，在进行参数的调整尝试中，耗费较大的时间。

通过本项目进行总结梳理，基本尝试了多种可用于回归的常用做法，并且了解了结果的优化处理方法，该流程与方法论可以沉淀到各类预测的场景中，但前提要对商店或者预测商品具有足够的特征信息，对于仅有时间单一维度的预测信息，可能就需要单独考虑时间序列等方式来进行。

但是最终的结果仍然可以考虑更加科学和柔性的方式来进行结果的提升，详细想法在“需要作出的改进”部分进行阐述

**需要作出的改进**

考虑到可以进行提升的方面：

1. 数据层面：

可以考虑变换组合更多的特征进行

1. 模型算法层面：

假设使用多个不同的随机种子，构建多个xgboost模型，然后将每个结果输入到一个融合模型或许能够将模型提升更好的效果

1. 计算能力层面：我们使用的xgboost模型对于计算资源的开销是较大的，在未来应用于更多数据的场景时，需要考虑进行分布式的运算能力。

4、 结果处理上：

本方案中仅考虑了针对整体预测之后的销量进行一个修正，更加柔性的方式是对不同类型的店铺，甚至对不同的店铺进行结果的修正

以上问题，可以在后续的实际环境中进行深入的分析和使用

**VI 参考资料**

线性回归参考来源：<https://baijiahao.baidu.com/s?id=1622335010099956389&wfr=spider&for=pc>

回归树参考来源：

<https://blog.csdn.net/weixin_40604987/article/details/79296427>

随机森林参考来源：

<https://blog.csdn.net/qq_34106574/article/details/82016442>

<https://www.cnblogs.com/emanlee/p/4851555.html>

xgboost参考资料：

<https://blog.csdn.net/huacha__/article/details/81029680>

<https://blog.csdn.net/github_38414650/article/details/76061893>

<https://www.cnblogs.com/Allen-rg/p/10563362.html>

<https://www.cnblogs.com/baby-lily/p/10663328.html>

<https://blog.csdn.net/m_buddy/article/details/79337492>

回归模型评价指标：

<https://blog.csdn.net/qq_37279279/article/details/81041470>

<https://www.cnblogs.com/qwj-sysu/p/8489323.html>

<https://blog.csdn.net/xiezhen_zheng/article/details/81143135>

可视化及算法实现参考来源：

<https://zhuanlan.zhihu.com/p/67731440?utm_source=wechat_session&utm_medium=social&utm_oi=611637534023880704>

<https://zhuanlan.zhihu.com/p/51516385>

<https://blog.csdn.net/aicanghai_smile/article/details/80987666>