

机器学习：Kaggle 项目实践

马天祥

I. 问题的定义

项目概述

Rossmann 是欧洲的一家连锁药店，在欧洲 7 个国家经验上千家店铺。本项目源自 Kaggle 比赛 Rossmann Store Sales，我们需要根据 Rossmann 的 1115 家药妆店的信息（比如促销，竞争对手，节假日）以及在过去销售情况，来预测其未来 6 周的销售额。

商店销售额受到很多因素的影响，包括促销活动，竞争对手，法定节假日，等等。由于数千药妆店管理者依据他们独特的情况预测销售情况，结果的准确性必定差别很大。可靠的销售预报可以让管理者创建更高效的安排工作人员日程，并增加他们的工作效率和积极性。通过帮助管理者们创建一个健壮的预测模型，也将帮助他们节省时间，以聚焦并关注最为重要的：客户和团队。

我选择此项目，因为他涵盖了数据分析、算法模型、机器学习等多个领域的知识，并且回归预测也是现今商业分析中很常见的情形，完整的过一遍该领域的具体问题及解决流程，对自己的技能提升有帮助。

在本项目中，我会采用 XGBoost 算法来建模。XGBoost 是基于梯度提升框架进行优化的算法，在 Kaggle 比赛的回归预测类题目中被普遍使用并表现亮眼。

本项目使用的数据集都来自 Kaggle 项目，没有过多寻找外部数据；store.csv 是以店铺维度的详细信息，train.csv 和 test.csv 是以店铺编号和日期为维度的相关信息，分别代表训练集和测试集。

问题陈述

本项目需要预测各药妆店未来一段时间的销售额，而在已给出的历史统计数据中已经存在销售额字段，故本项目需解决的问题属于机器学习中监督学习的范围，且为一个回归问题。因此，具体方案会包括如下几个步骤：

- 数据清洗：缺失值，异常值，以及其他干扰数据的处理；
- 特征工程：数据类型、分布、互相关度等维度的不同，需要结合目标仔细处理；
- 模型构建：模型选择（xgboost，或多模型融合）；
- 评估及调参：根据统一评价标准，训练模型并寻找最佳参数，防止过拟合；

在 Kaggle 的 Rossmann 比赛中，分 Private 和 Public 两个排行榜，Public Leaderboard 由大概 39% 的测试数据预测值计算得到，而 Private Leaderboard

由另外 61% 的测试数据预测值计算得到；两个板块的分数会有所不同，Kaggle 官方以 Private 排行榜为最终评选依据。在已揭晓的榜单中，前 10% 的分数为 0.11773 (Private)；在本项目中，我也会以此做为基准，在至少达到此分数的基础上，通过衍生变量，XGBoost 参数优化，模型融合等方法，来实现更高的 Private 分数，希望可以进入前 5% 到 3%。

评价指标

使用 Kaggle 建议的 RMSPE 来做为验证函数，该值越低代表差异性越小。它是指模型的预测值和实际观察值之间的差异的一种衡量方式。

$$RMPSE = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{y_i} \right)^2}$$

在公式中，y 为真实销量数据，y_hat 为预测数据，比赛中最终真实值并不公布，但任何销量为 0 的数据极其预测值，不会被加入到评估中。

II. 分析

数据的探索

项目所用到数据集可以在 Kaggle 官网下载，数据文件如下：

- **train.csv**: 训练数据集，共 1017209 条记录，包含各门店的所有历史统计数据，包含 9 个属性：

"Store", "DayOfWeek", "Date", "Sales", "Customers", "Open", "Promo", "StateHoliday", "SchoolHoliday"

- **test.csv**: 测试数据集，共 41088 条数据，记录各门店最近两个月历史数据（不含销量数据），包含 8 个特征：

"Id", "Store", "DayOfWeek", "Date", "Open", "Promo", "StateHoliday", "SchoolHoliday"

- **sample_submission.csv**: 预测数据格式样本，包含特征：

"Id", "Sales"

- **store.csv**: 关于商店的附加信息，共 1115 条店铺数据，包含 10 个特征：

"Store", "StoreType", "Assortment", "CompetitionDistance", "CompetitionOpenSinceMonth", "CompetitionOpenSinceYear", "Promo2", "Promo2SinceWeek", "Promo2SinceYear", "PromoInterval"

数据集特征及含义：

- Id - 单条数据唯一编号。
- Store - 药妆店唯一编号。

- Sales - 当天的销售营业额。
- Customers - 当天的消费者数。
- Open - 商店是否开门，0 为关，1 为开。
- StateHoliday - 法定（公共）节假日，a=所有的节假日，b=复活节，c=圣诞节
- SchoolHoliday - 学校放假日。
- StoreType - 商店类型：a, b, c, d。
- Assortment - 描述种类的程度，a = basic, b = extra, c = extended。
- CompetitionDistance - 最近的竞争对手的商店的距离。
- CompetitionOpenSince[Month/Year] - 最近的竞争者商店大概开业的年和月时间。
- Promo - 表明商店该天是否在进行促销。
- Promo2 - 指的是持续和连续的促销活动。：0：商店没有参加，1：商店正在参加。
- Promo2Since[Year/Week] - 表示参加连续促销开始的年份和周。
- PromoInterval - 描述持续促销间隔开始月份。

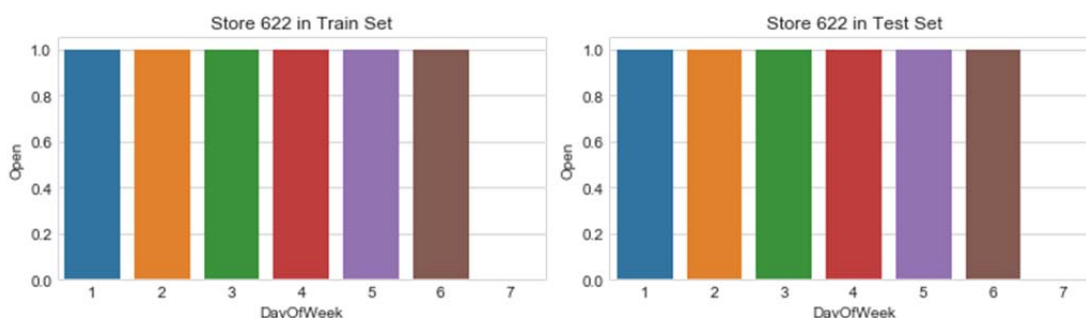
数据完整性初探：

- 变量名称：训练集数据包含标签 ‘Sales’ 和变量 ‘Customers’ 而测试集数据不含有这两列数据；
- 数据类型及数据缺失：部分数据的类型需要转换，比如类型转数值（StateHoliday），对数转换（Sales），等；测试集（test）在变量 ‘Open’ 有 11 条数据缺失：

	DayOfWeek	Open	Promo	SchoolHoliday	StateHoliday
train	[5, 4, 3, 2, 1, 7, 6]	[1, 0]	[1, 0]	[1, 0]	[0, a, b, c, 0]
test	[4, 3, 2, 1, 7, 6, 5]	[1.0, nan, 0.0]	[1, 0]	[0, 1]	[0, a]

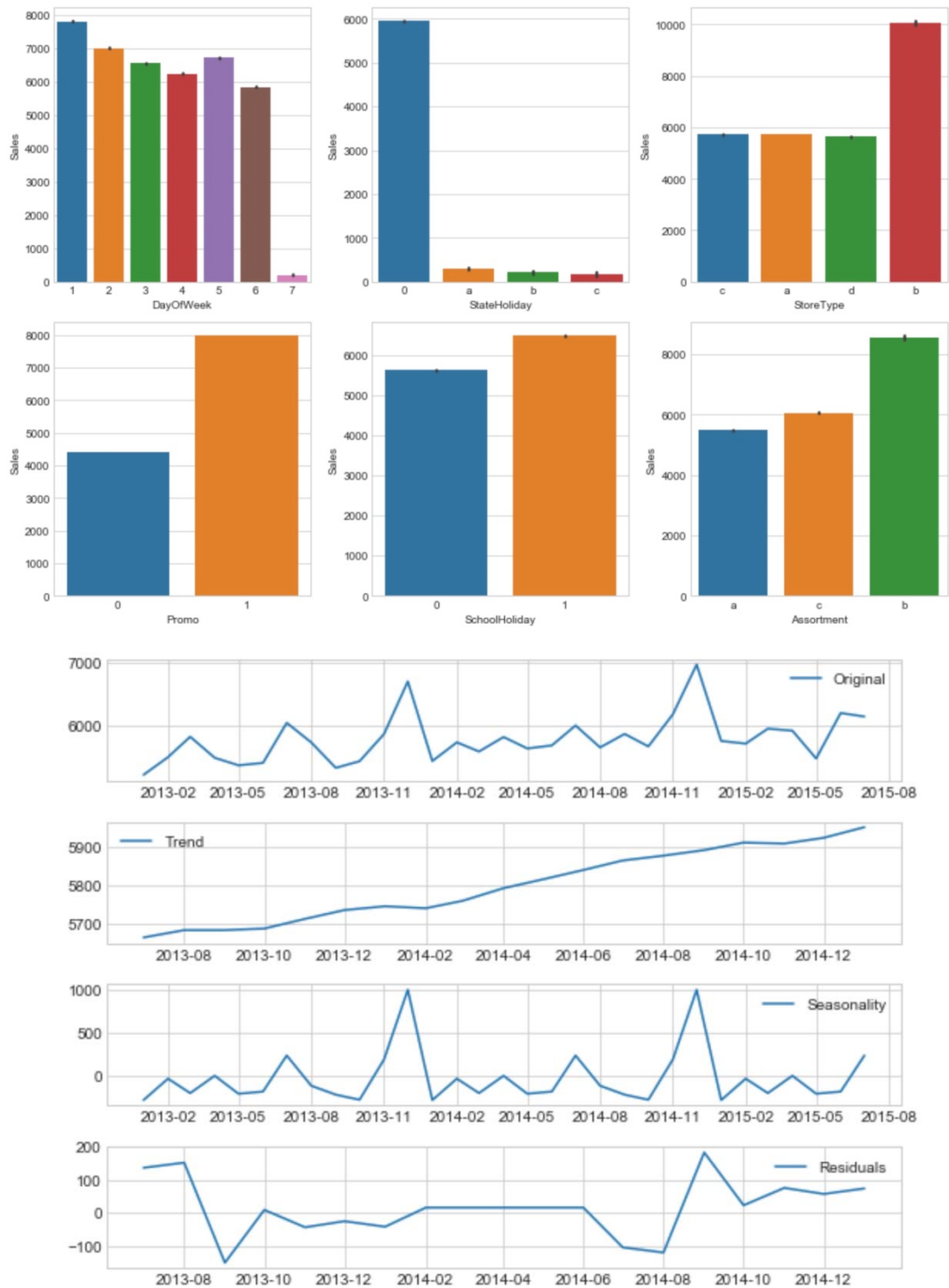
由上图对部分变量取值范围的观察：

- 1、训练集 StateHoliday 变量，在数值 0 和字符串 ‘0’ 都有取值，应属于数据异常，由于原始变量属于类型值，后续统一转为字符串 ‘0’；
- 2、测试集有 11 条数据在 Open 变量为缺失值，经提取，均为 622 号店铺，且都在周一至周六，又由下图统计得，622 号店铺在所有的周一至周六均开业，而在周日全休，故上述缺失值后续都补值 1。



探索性可视化

- 1、变量不同取值条件下的销量均值统计：由下图可明显看出，上述六个类型变量（包含节假日类型、店铺类型、是否促销、星期几，等），都对销量有较明显的影响。

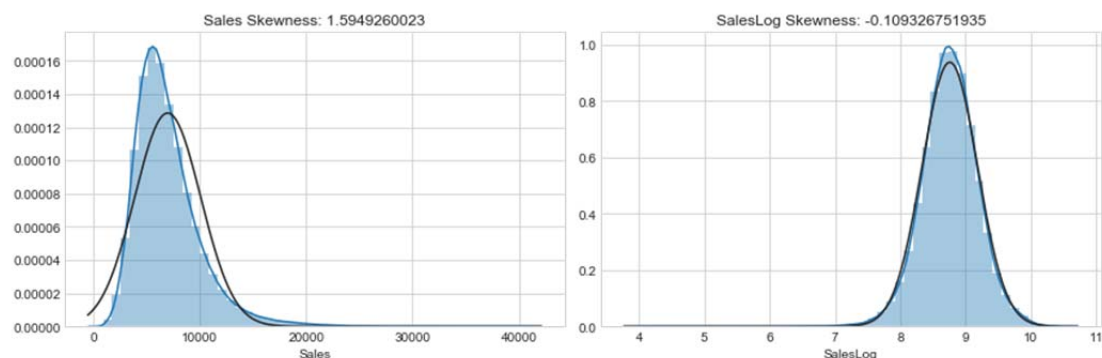


2、时间维度下销量总体趋势：由上图，日期也对销量有一定影响。通过取包含所有店铺的月均销量，按时间顺序排列并分解后，可明显看出随时间推进，销量可分为三个因素：

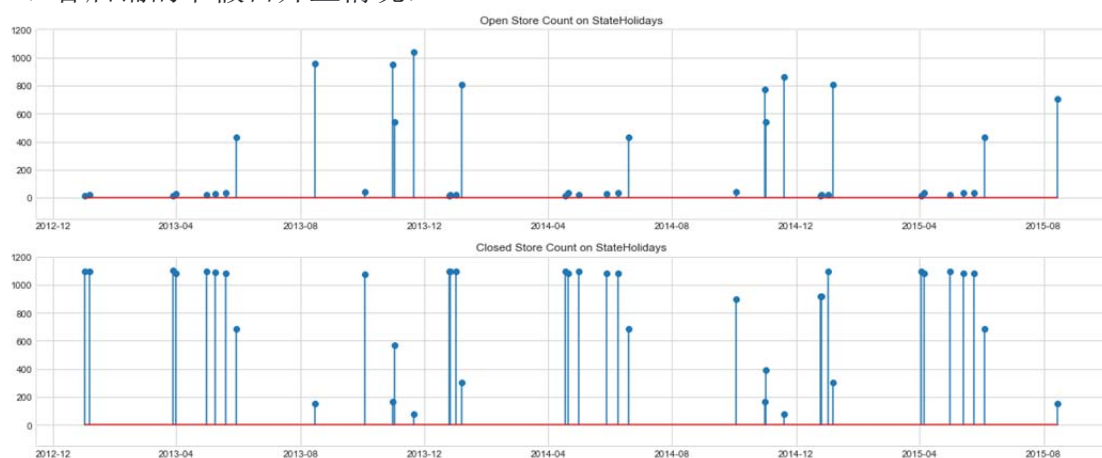
- (1) 趋势因素：近似线性增长；
- (2) 季节性波动：以一年为周期，7、11-12月增长显著；
- (3) 残差：理论上与其他因素（竞争对手、节假日、促销活动，等）有关。

3、销量标签值分布，以及对数转换：由下图，取销量大于 0 的数据分布（左），

可见其在右侧有相当长的长尾，相比正态分布，中心也有一定程度的左偏；后续将会对其做对数转换（右），转换后，整体分布更加趋向正态分布；



4、各店铺的节假日开业情况：



由上图，各店铺对于节假日的安排并不一致。部分节假日，所有店铺都休息；但也有部分日期，有的店铺并不认为是节假日，照常营业；这可能和店铺所在的国家地区不同或店铺自身的管理制度等决定。基于此，后续在时间维度上生成更多衍生变量时，比如对节日的相对天数，则需要按店铺单独计算。

算法和技术

预测销量是机器学习中的一类监督学习回归问题。经典的回归算法有很多，如线性回归、决策树、SVM，也有集成模型如随机森林（bagging），GBM（boosting），等。

Boosting 属于集成学习模型，它基本思想是把成百上千个分类准确率较低的树模型组合起来，成为一个准确率很高的模型。这个模型会不断地迭代，每次迭代就生成一颗新的树。

本项目我采用 Xgboost 算法来建立预测模型。Xgboost 也是属于 boosting 的一种算法，基于 Gradient-Boosting 算法进行了优化，是 Gradient Boosting 的一种高效系统实现，在业界广泛使用且性价比非常高。Kaggle 的 Rossman 竞赛参赛者 Gert 也是通过 Xgboost 作为基准模型，并使用多模型融合，最终取得了第一名的成绩。

Xgboost 的 loss 由两部分构成，前者优化经验误差，后者控制泛化误差；其目标函数如下：

$$Obj = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k)$$

Training loss Complexity of the Trees

想较于传统的 GBDT，xgboost 正则更加细化，包括传统的 L2 正则以及叶子数目的正则项：

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

Number of leaves L2 norm of leaf scores

在对目标函数的优化上，Xgboost 也和 GBDT 有所不同：GBDT 采用的是数值优化的思维，用的最速下降法求解 Loss function 的最优解，其中用 CART 决策树去拟合负梯度，用牛顿法求步长；而 XGBoost 用的解析的思维，对 Loss Function 展开到二阶近似，求得解析解，并用其作为 Gain 来建立决策树，使得 Loss function 最优。

*** 算法优势：**相对于普通 gbm 的实现，XGBoost 有以下的一些优势：

- 1、正则化：标准 GBM 的实现没有像 XGBoost 这样的正则化步骤。正则化对减少过拟合也是有帮助的，XGboost 显式的将数模型的复杂度作为正则项加载优化目标；
- 2、并行处理：XGBoost 可以实现并行处理，相比 GBM 有了速度的飞跃；同时 XGBoost 也支持 Hadoop 实现。
- 3、高度的灵活性：XGBoost 允许用户定义自定义优化目标和评价标准，在本项目中的评价标准 rmspe，就需要自行编写函数；
- 4、缺失值处理：XGBoost 内置处理缺失值的规则。XGBoost 在不同节点遇到缺失值时采用不同的处理方法，并且会学习未来遇到缺失值时的处理方法。
- 5、剪枝：当分裂时遇到一个负损失时，GBM 会停止分裂。因此 GBM 实际上是一个贪心算法。XGBoost 会一直分裂到指定的最大深度(max_depth)，然后回过头来剪枝。如果某个节点之后不再有正值，它会去除这个分裂。这种做法的优点，当一个负损失（如-2）后面有个正损失（如+10）的时候，就显现出来了。GBM 会在-2 处停下来，因为它遇到了一个负值。但是 XGBoost 会继续分裂，然后发现这两个分裂综合起来会得到+8，因此会保留这两个分裂。
- 6、内置交叉验证：XGBoost 允许在每一轮 boosting 迭代中使用交叉验证。因此，可以方便地获得最优 boosting 迭代次数；而 GBM 使用网格搜索，只能检测有限个值。

*** 模型参数：**XGBoost 模型的主要包含以下参数：

常规参数 (General Parameters)：

1. booster：选择基分类器；gbtree/gblinear

2. silent: 1 为没有运行信息输出, 反之为 0.
3. nthread: 线程数;

模型参数 (Booster Parameters):

1. eta: 学习率;
2. min_child_weight: 这个参数控制叶子节点中二阶导的和的最小值;
3. max_depth: 每颗树的最大深度;
4. max_leaf_nodes: 最大叶结点数;
5. gamma: 后剪枝时, 用于控制是否后剪枝的参数。
6. max_delta_step: 这个参数在更新步骤中起作用, 如果取 0 表示没有约束, 如果取正值则使得更新步骤更加保守。可以防止做太大的更新步子, 使更新更加平缓。
7. subsample: 样本随机采样;
8. colsample_bytree: 列采样, 对每棵树的生成用的特征进行列采样;
9. lambda: 控制模型复杂度的权重值的 L2 正则化项参数;
10. alpha: 控制模型复杂程度的权重值的 L1 正则项参数;
11. scale_pos_weight: 如果取值大于 0 的话, 在类别样本不平衡的情况下有助于快速收敛。

学习任务参数 (Learning Task Parameters):

1. objective: 定义最小化损失函数类型;
2. eval_metric: 误差函数;

基准模型

在 Kaggle 的 Rossmann 比赛现有的 Private 和 Public 两个排行榜中, Private 相对来说更有说服力, 因此本项目中我会以 Private Leaderboard 上前 10% (第 330 名) 的分数来作为基准值, 为 0.11773; 如果模型预测的分数好于这个值, 可以说明预测模型基本达标; 进而继续优化争取更好的分数。

III. 方法

数据预处理

- 1、测试集 ‘Open’ 变量缺失值处理: 由前面数据探索模块, 测试集有 11 条数据 (622 店铺) Open 变量全部补 1;
- 2、训练集 ‘StateHoliday’ 变量, 取值 ‘0’ 和 0 进行统一, 先全转换成字符串 ‘0’ ;
- 3、标签数据对数转换: 对训练集标签 ‘Sales’ 做对数转换, 转换后数据保存为新字段 ‘SalesLog’ ;

4、时间序列分解：从‘Date’变量中提取出‘Year’、‘MonthofYear’、‘WeekofYear’、‘DayofMonth’、‘DayofYear’、‘DateInt’等变量，作为独立特征，以加入到后续模型训练中；

5、在训练集中，去除测试集中未出现的店铺数据：基于不同店铺的销量表现很大概率上不相关，因此为增强模型对测试集的预测准确性，在此剔除可能的干扰数据；

6、其他已有数据转换：‘StateHoliday’、‘StoreType’、‘Assortment’、‘PromoInterval’四个类别型变量转换为数值型；

7、竞争对手衍生变量：

- CompetitionOpenInt：竞争度对手开业时间；
- CompetitionLastMonths：竞争对手已开业月份；
- inCompetition：当前日期店铺是否有已开张竞争对手；

8、持续促销衍生变量：

- Promo2SinceFloat：加盟活动时间；
- inPromo2：当前日期店铺是否已加入连续促销活动；
- Promo2LastMonths：已加入活动月份数；

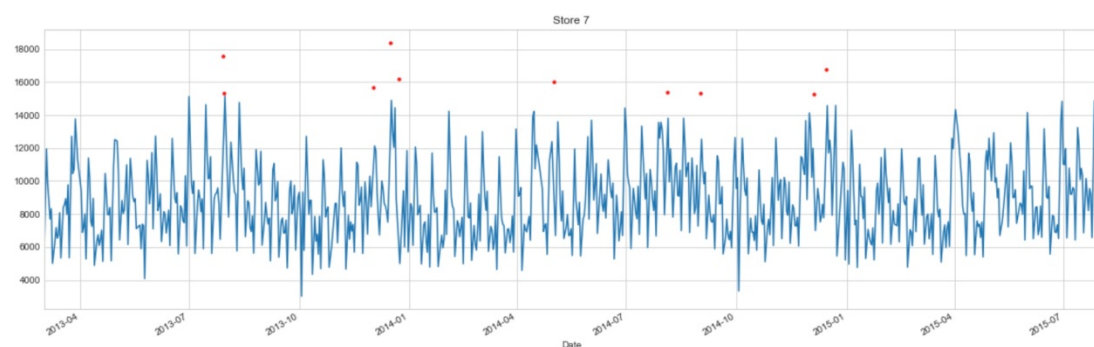
9、店铺维度的销量相关平均值：

- sal_avg：店铺日均销量（同标签）；
- sal_avg_std：店铺日均销量标准差
- sal_avg_percustomer：店铺人均消费；
- sal_avg_schoolhol：店铺学校假期日均销量；
- sal_avg_promo：店铺促销日均销量；
- sal_avg_statehol：店铺节假日日均销量；

10、合并数据集：将店铺数据集与训练集、测试集合并；

11、异常值处理：对于偏离销量均值很多的异常值，这里采用箱型图识别异常值标准，以店铺为维度单独计算：

- Q1 为下四分位数，Q3 为上四分位数，IQR 为 Q1、Q3 之差；
- 销量小于 $Q1 - 1.5 * IQR$ ，大于 $Q3 + 1.5 * IQR$ 的数据则认为是异常值；
- 下图以 7 号店铺为例，被标记为离群点的数据以红点的形式展示：



执行过程

1、环境依赖：该项目主要用到的 python 库有：pandas, numpy, matplotlib, xgboost, 等；因为 GPU 资源有限且 xgboost 在普通 GPU 的加速效果不明显，所以使用的 xgboost 是 CPU 版本，在 AWS 虚拟机的 ubuntu 系统上以多线程($n_thread = 16$) 并发的形式进行运算训练；

2、特征工程：按照上节数据预处理的流程，依次处理异常值、缺失值，转换数据类型，衍生变量等；在缺失值处理时，这里仅处理了教重要的测试集开业状态（Open），由于 xgboost 本身带有缺失值自动处理模块，其他缺失值可交由算法处理；

3、评估函数：由于 Kaggle 在评估时会忽略不营业（销量为 0）的数据，且在数据预处理时我也剔除掉了 0 销量的数据记录，故评估函数直接复制评估公式的逻辑，不用做过多处理：

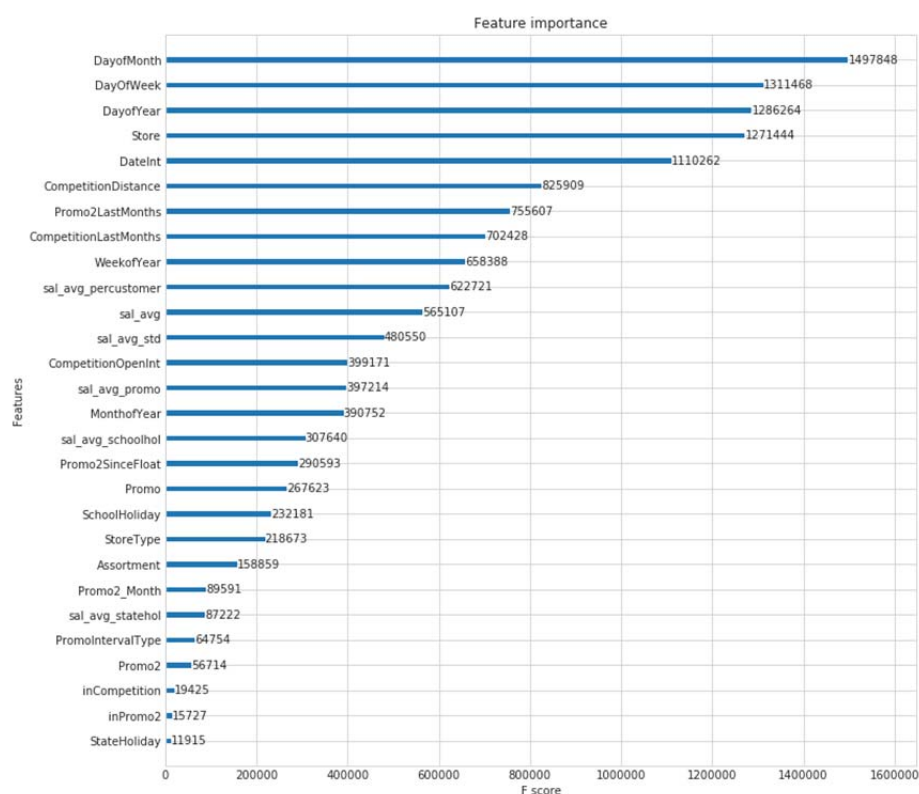
```
def rmspe(y, yhat):  
    return np.sqrt(np.mean((yhat/y-1) ** 2))  
  
def rmspe_xg(yhat, y):  
    y = np.expml(y.get_label())  
    yhat = np.expml(yhat)  
    return "rmspe", rmspe(y,yhat)
```

4、模型训练：在预处理完成的基础数据上，通过划分训练、验证集数据，并构建 xgboost 模型进行训练；xgboost 模型参数主要分为三类：通用参数、booster 参数、学习目标参数；我们需要不断调整优化的是第二类，booster 参数；

（1）划分数据集：在划分训练、验证集时，基于本项目为时序预测的特性，我们把训练数据的最后六周做为验证集；确保在训练时不出现数据泄露问题；同时也可以检验训练后的模型对未来六周销量预测的准确程度；

（2）参数调优：实际中，我首先用到了 sklearn 的 GridSearchCV 模块，在较大的学习率（0.1）和较小的 booster 数量（100）条件下，对几个重要参数做了网格搜索以得到最优解，包括：max_depth、min_child_weight、gamma、subsample、colsample_bytree、reg_alpha、reg_lambda；

（3）然后，再以一个小的学习率（0.01），以及一个较大的 booster 数量（500）来做正式训练；训练最终得到的特征重要性图如下：



(3) 训练后得到的特征重要性排行见上图，可以看到各种时间类特征几乎都排在了前面；模型对验证集的 rmspe 达到 0.1247；将模型对测试集的预测数据提交 Kaggle, 得到初版预测的分数分别为: 0.12795(Private), 0.12256(Public); 由基准模型可以看到分数未进入 Private Top 10%, 可见模型还有很大提升空间。

完善

基于前述的算法和训练过程，我认为主要存在两个优化方向：

- (1) 在现有模型基础上继续优化调参，并训练多个模型，用模型融合的方式取均值作为最终预测值；
- (2) 寻找和引入更多效果显著的新变量，增加模型的预测能力；

从 Kaggle 该竞赛第一名获得者 Gert 的经验分享中，可以看到他衍生出了很多新变量，主要分为 Recent Data, Temporal Information, Current Trends, 都和 时间维度有关；同时从初版模型的特征重要性图也可充分说明时间维度的关键性，因此我决定从挖掘更多新衍生变量入手。

**** 更多衍生变量：**在时间维度上，对每个店铺按天的历史销量进行衍生，这里包含了我理解的三类变量：recent data、temporal information 和 current trends：

* Temporal Information:

- daysFromLastStateHol: 店铺距离上次节假日天数;
- daysToNextStateHol: 店铺距离下一个节假日天数;
- daysFromLastSchHol: 店铺距离上一个学校假期天数;
- daysToNextSchHol: 店铺距离下一个学校假期天数;

* Current Trends:

- avgsal_last3m: 店铺近 3 个月（到昨天为止）日均销量;
- avgsal_last6m: 店铺近 6 个月（到昨天为止）日均销量;
- avgsal_curryear: 店铺今年以来（到昨天为止）日均销量;
- avgsal: 店铺历史（从 13 年到昨天为止）的日均销量;

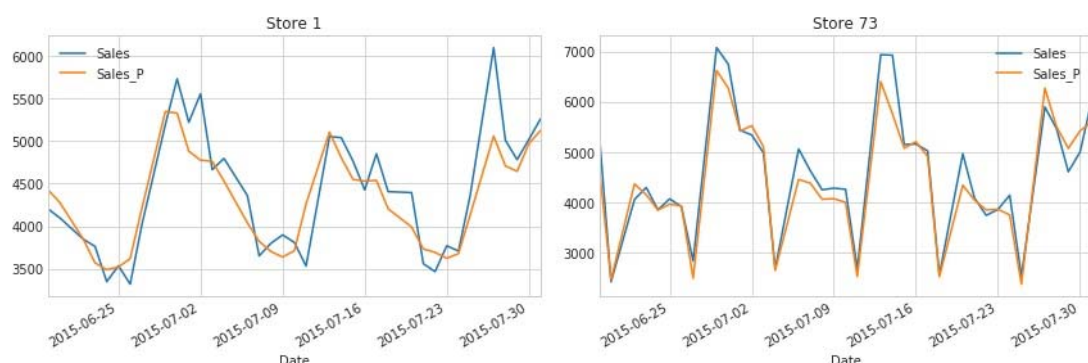
* Recent Data:

- avgsal_mblm: 店铺当月往前推第二个月月均销量; • avgsal_std: 店铺历史（从 13 年到昨天为止）的日均销量标准差;
- avgsal_percustomer: 店铺历史（从 13 年到昨天为止）的人均消费;
- avgsal_dayofyear: 店铺历史上每年同天的日均销量;
- avgsal_dayofweek: 店铺近 6 个月（到昨天为止）星期同天的日均销量;
- avgsal_dayofweek_res: 店铺近 6 个月（到昨天为止）星期同天日均销量与近 6 月日均销量之差;
- avgsal_dayofmonth: 店铺近 6 个月（到昨天为止）每月同天的日均销量;
- avgsal_dayofmonth_res: 店铺近 6 个月（到昨天为止）每月同天日均销量与近 6 月日均销量之差;

加入上述新变量后，由于其中一些需要计算当前日期前一段时间的销量均值，因此训练集数据剔除了 2013 年前三个月数据，使得衍生变量值更合理；

（1）再次网格搜索调整 Booster 参数后，此时模型在验证集数据上的 rmspe 达到了 0.1124；由下图可以看出，模型对验证集最后六周的各店铺的预测基本能反映出销量的实际量级和变化趋势；

提交测试集预测数据到 Kaggle，成绩也有了较大提高，Private 分数达到 0.11656，已经高于基准分数水平（0.11773），进入到 Top 7%。




（2）由于前述模型训练时，分离了最近 6 周有标签的数据作为验证集，而实际上这六周是离预测集时间最近的一段，理论上有更强的相关性，所以我最后尝试将这六周也加入到训练中（此时使用了随机划分），在迭代 1 万多颗树后，最终

得到的 Kaggle 结果分数再度提升：0.10488（Public），0.11413（Private）；Private 分数，已经可以排到 106 名，约前 3%，这也已经达到了最初的目标分数。Kaggle 成绩截图见下：

3/18/2018 Rossmann Store Sales | Kaggle

[kaggle](#) [Competitions](#) [Datasets](#) [Kernels](#) [Discussion](#) [Jobs](#)

**Rossmann Store Sales**

Forecast sales using store, promotion, and competitor data
\$35,000 · 3,303 teams · 2 years ago

[Overview](#) [Data](#) [Kernels](#) [Discussion](#) [Leaderboard](#) [Rules](#) [Team](#) [My Submissions](#) [Late Submission](#)

Your most recent submission

Name	Submitted	Wait time	Execution time	Score
submission.csv	8 hours ago	6 seconds	0 seconds	0.10753

Complete

[Jump to your position on the leaderboard](#)

You can select up to 2 submissions to be used to calculate your final leaderboard score. If 2 submissions are not selected, they will be chosen based on your best submission scores on the public leaderboard.

Your final score may not be based on the same exact subset of data as the public leaderboard, but rather a different private data subset of your full submission — your public score is only a rough indication of what your final score is.

You should thus choose submissions that will most likely be best overall, and not necessarily on the public subset.

```
kaggle competitions submit -c rossmann-store-sales -f submission.csv -m "Message"
```

48 submissions for [Tianxiang Ma](#) Sort by Most recent

[All](#) [Successful](#) [Selected](#)

Submission and Description	Private Score	Public Score	Use for Final Score
submission.csv a minute ago by Tianxiang Ma 0318SC097-d12-09-07	0.11413	0.10488	<input type="checkbox"/>
submission.csv 2 minutes ago by Tianxiang Ma MessageNSC	0.12103	0.10871	<input type="checkbox"/>

IV. 结果

模型的评价与验证

实际训练中，在上模块新增店铺+时间维度的新衍生变量后，代入 xgboost 模型最终收敛在第 11856 轮（在多核 CPU 的 Linux 虚拟机上，以 n_thread=16，训练时间约 3 到 4 个小时）。

由前述模块，因为本项目原为 Kaggle 竞赛，所以对模型的评价可以通过提交预测数据到 Kaggle 官网，自动运算得到。

提交 Kaggle 的最终分数：0.10488 (Public)，0.11413 (Private)；从更为客观的 Private Leaderboard（包含 61%测试集数据）来看，这个分数可以排在第 106 名，约属于 Top 3%，已经达到预订目标；且由于时间原因，后续没有继续做模型融合（需要以更多参数组合多次训练并求平均）。

合理性分析

在模型选取上，Xgboost 本身是一种非常精致的算法，可以处理各种不规则的数据，这使得其预测结果很大程度上优于传统的线性模型；

对比基准模型，最终模型做了很多的改进，效果最显著的是加入了多个时间序列上的衍生变量，这些时间趋势上的相关性，进一步增加了预测的精准度；

在 Kaggle 分数上，最终模型相比基准模型取得了很大的改善，最终分数也达到了预期值。但相比 Kaggle 排行榜前几仍然有差距，我思考的主要原因有以下几点：需要更仔细的打磨变量组合，可以引入外部数据如天气、等，以及模型融合提高平均分数，防止过拟合；

综上，得益于优秀的 xgboost 算法，和时间序列上的衍生变量，本项目的最终结果是比较合理的。

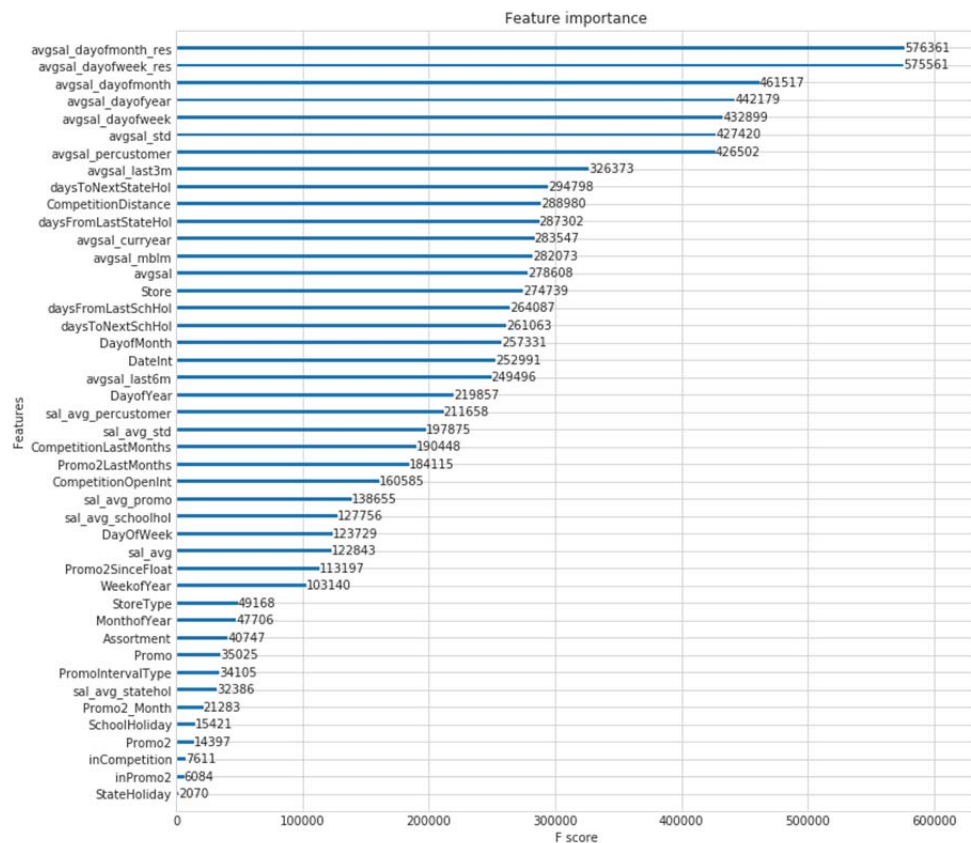
V. 项目结论

结果可视化

最终模型的特征重要性排行见下图，可以看出新增的衍生变量对预测产生了非常重要的影响。

排名靠前的变量，包括历史相似日期的平均销量、残差、人均消费、距离节假日的天数，等；

相反的，店铺类型，是否加入连续促销活动，等变量在此模型中作用十分有限。



对项目的思考

本项目的处理流程，主要包括：

- 1、数据清洗、预处理、探索性分析；
- 2、xgboost 建模；
- 3、变量衍生、调参优化；
- 4、结果分析；

这个项目属于一个基于真实商业数据的回归问题，有效结合了数据科学、机器学习等多个模块，可以带给我一个完整的方法论和思维框架，这也是我选择一边学习一边做的原因。

总体上，我感觉数据、模型、调参三者相辅相成，共同为最终预测结果提供坚实的基础。

数据特征的好坏，直接决定了标签数据是否可预测，是否能更精准的预测，因此对数据预处理和特征工程，是最需要关注的，特别在本项目所属的回归问题中；

模型选取相对来说没那么纠结，xgboost 已经是业界广泛使用的前沿模型，上手也方便；而调参却是一个慢工出细活的模块，不仅需要对模型参数有很好的把握，在变量特征的衍生和取舍，也对最终预测结果影响很大。

需要作出的改进

在我做的这个项目上，肯定还有很大的提升空间，结合上模块所说的数据、模型、调参，我觉得主要以如下几方面为主：

特征工程：衍生变量可以从更多角度思考，如外部天气数据、等；特征组合和取舍，可以做到更精细，一些被证明基本无关的变量，可以剔除在训练之外；

算法：可以尝试业界的其他前沿算法，如 lightgbm 等；

模型融合：本项目中因时间原因未使用，但实际流程处理中，应该加入这一环，以进一步增加预测准确性，防止过拟合；多模型融合，及用多个模型预测结果的平均值作为最终预测结果；多个模型预测结果，可以用相同模型的不同参数，不同变量训练得到，也可用不同模型训练得到。

VI. 参考资料

1. XGBoost 模型：<https://github.com/dmlc/xgboost>
2. Kaggle 竞赛：<https://www.kaggle.com/c/rossmann-store-sales>
3. rossmann 官网：<https://www.rossmann.de/einkaufsportal/>
4. Gert 方案：<https://www.kaggle.com/c/rossmann-store-sales/discussion/18024>
5. 可视化工具：<http://seaborn.pydata.org/>
6. 时间序列分析库：
http://www.statsmodels.org/stable/generated/statsmodels.tsa.seasonal.seasonal_decompose.html
7. xgboost 调参指导：http://blog.csdn.net/han_xiaoyang/article/details/52665396
8. xgboost 参数文档：<http://xgboost.readthedocs.io/en/latest/parameter.html>
9. Kaggle 官方排行：<https://www.kaggle.com/c/rossmann-store-sales/leaderboard>
10. xgboost 概念解析：<https://www.zhihu.com/question/41354392>