预测 Rossmann 未来销售额

王姜萍

I. 问题的定义

1.1 项目概述

Rossmann 在 7 个欧洲国家经营着 3,000 多家药店。目前，Rossmann 商店经理的任务是预先提前六周预测他们的日常销售。成千上万的个体经理根据他们独特的情况预测销售情况，结果的准确性可能会有很大差异。本项目通过帮助 Rossmann 创建一个强大的预测模型使商店经理更准确的预测日常销售。

本项目将应用特征工程、数据挖掘和机器学习算法帮助 Rossmann 更清晰准确的分析销售数据和预测销售数据。

特征工程是通过工程活动，将原始数据中特征规律使用更高效的编码方式(特征)表示，使用特征表示的信息损失较少，规律依然保留，尽量减少原始数据中的不确定因素，如白噪声、异常数据、数据缺失等[1]。

数据挖掘确切地讲是一种决策支持过程，它主要基于人工智能、机器学习、统计学等技术，高度自动化地分析企业原有数据，做出归纳性的推理，从中挖掘出潜在的模式，预测客户的行为，帮助企业的决策者调整市场策略，减少风险，做出正确决策。

机器学习(Machine Learning, ML)是一门多领域交叉学科，涉及概率论、统计学、逼近论、凸分析、算法复杂度理论等多门学科[2]。机器学习模型，如随机森林、增强学习(Boosting)、支持向量机（SVM）和神经网络，比简单的可解释模型如决策树，朴素贝叶斯、决策规则具有更好的预测性能[3]。

本项目数据来源：

(1)训练集 train.csv:时间范围为 2013 年 01 月 01 日到 2015 年 07 月 31 日，共 942 天，1017209 条数据。

(2)测试集 test.csv:时间范围为 2015 年 08 月 01 日到 2015 年 09 月 17 日，共 48 天，41088 条数据。

(3)商店基本信息数据集 store.csv:1115 条数据，共 1115 家商店的信息。

1.2 问题陈述

本项目通过 2013/1/1-2015/7/31Rossmann1115 家商店历史销售数据及其相关因素预测未来某天销售额。商店销售受许多因素的影响：DayofWeek、CompetitionOpenSinceWeek、Store、Promo、CompetitionDistance、SchoolHoliday、etc。 分析这些影响因素及历史销售数据选择算法建立模型，用于对未来销售预测，预测过程注意数据集时间序列的特性。本项目选择 XGBoost 算法建模预测。

XGBoos 极端梯度提升树算法，是梯度提升算法的扩展。Boosting 分类器属于集成学习模型，其基本思想是把成百上千个分类准确率较低的树模型组合成一个准确率较高的模型。XGBoost 是 Gradient Boosting Machine 的实现，能自动利用 CPU 的多线程进行并行、并对算法加以改进以提高精度。XGBoost 的基学习器既有树(gbtree)又有线性分类器(gblinear)，从而得到带 L1+L2 惩罚的线性回归或逻辑回归，其损失函数采用二阶泰勒展开，具有高准确度、不易过拟合、可扩展性等特点，能分布式处理高维稀疏特征，因此在同等情况下，XGBoost 算法比同类算法快 10 倍以上[4]。

初步应用 XGBoost 模型后，依据训练结果再次对模型进行校正。校正模型后最后应用模型融合 Bagging 算法再次预测销售数据。

预测模型的评估主要由泛化能力和拟合程度两个指标衡量。尽量使得预测的结果靠近真实值。

1.3 评价指标

1.使用回归评价指标[8]

（1）RMSE

RMSE（root mean square error,平方根误差），定义为：

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

其中，y_i是真实值，$\hat{y}_i$是预测值，n 是样本数量，使用了欧式距离。

缺点：平均值是非鲁棒的，对于异常点敏感，如果某个异常点误差大，整个 RMSE 就会比较大。

（2）均方差（mean squared error）

$$\mathrm{MAE}(y, \hat{y}) = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

其中，y_i是真实值，$\hat{y}_i$是预测值，n 是样本数量。

（3）平均绝对误差（mean absolute error）

$$\mathrm{MAE}(y,\hat{y}) = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n}$$

其中，y_i是真实值，$\hat{y}_i$是预测值，n 是样本数量。

（4）中值绝对误差（Median absolute error）

$$\mathrm{MedianAE} = \mathrm{median}(|y_i - \hat{y}_i|)$$

其中，y_i是真实值，$\hat{y}_i$是预测值。

（5）R2 决定系数（r2 score）

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y_i})^2}$$

其中，y_i是真实值，$\hat{y}_i$是预测值，$\overline{y_i}$ 是平均值，n 是样本数量。

2.sklearn 回归评价指标

Scoring(Regression)	Function
Neg_mean_absolute_error	Metrics.mean_absolute_error
Neg_mean_squared_error	Metrics.mean_squared_error
Neg_median_absolute_error	Metrics.median_absolute_error
R2	Metrics.r2_score

3.自定义评价函数

由于 XGBoost 函数库没有模型评估函数，所以本项目采用自定义函数用于项目内每次运算评估，1-2 上面两项指标用于模型间评估。

XGBoost 的 best_iteration 和 best_score 均是基于评价函数得出。XGBoost 中对于评价函数调用时同样会传入 preds 和 dvalid，即为验证集和验证集上的预测值，返回值为一个字符串标识自定义评价函数的类型和一个 flaot 类型的 fevalerror 值表示评价值的大小，其是以 error 的形式定义，即当此值越大是认为模型效果越差。XGBoost 官方库中不支持以 F1 Score 来作为评价函数，用户可同过自定义 feval 实现。

自定义 feval 用 rmspe 来表示：$\mathrm{RMSPE} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}/y-1)^2}{n}}$

其中，y_i是真实值，$\hat{y}_i$是预测值，n 是样本数量。

II. 分析

2.1 数据的探索

1.数据来源：

(1)训练集 train.csv:时间范围为 2013 年 01 月 01 日到 2015 年 07 月 31 日，共 942 天，1017209 条数据。

(2)测试集 test.csv:时间范围为 2015 年 08 月 01 日到 2015 年 09 月 17 日,共 48 天,41088 条数据。

(3)商店基本信息数据集 store.csv:1115 条数据，共 1115 家商店的信息。

2.数据描述

（1）store 数据集简要描述

#Store：Rossmann 店铺的编码，没有空值，数值型，范围【1-1115】；

#StoreType：有四种不同的店铺类型：类别型 a,b,c,d，没有空值；

#Assortment：描述店铺分类水平：a=basic,b=extra,c=extended，类别型，没有空值；

#CompetitionDistance：离最近竞争对手店铺距离，范围【20-75860】，数值型，有空值；

#CompetitionOpenSinceMonth:最近竞争对手店铺开业时间,761 家店铺有竞品:月份【1-12】，数值型，有空值；

#CompetitionOpenSinceYear：最近竞争对手店铺开业时间：年份【1900-2015】，数值型，有空值；

#Promo2：店铺的 Promo2 促销活动，0 表示店铺没有参加促销，1 表示店铺参加促销有 571 家，没有空值；

#Promo2SinceWeek：表示店铺开始参加 Promo2 促销活动的当年第几周，不参加促销店铺为空，数值型，范围【1-50】；

#Promo2SinceYear：表示店铺开始参加 Promo2 促销活动的年份，不参加促销店铺为空，数值型，范围【2009-2015】；

#PromoInterval：表示 Promo2 促销活动每年的启动月份间隔，例如 Jan,Apr,Jul,Oct 表示当年 1,4,7,10 月份开始每一轮 Promo2 促销活动，字符串，不参加为空。

（2）train、test 数据集简要描述

#Store：店铺编号，范围【1-1115】，与 store 数据集 store 对应，数值型，没有空值；

#DayOfWeek：一周的第几天，范围【1-7】，数值型，没有空值；

#Date：销售日期，格式 xxxx-xx-xx，日期数据需要转化数据特征，没有空值；

#Sales：营业额，目标变量，范围【0-41551】，数值型，没有空值；

#Customers：对应客户数量，范围【0-7388】，数值型，没有空值；

#Open：表示是否正在营业中，0 表示关门，1 表示营业，数值型，没有空值；

#Promo：表示营业当天是否有促销活动，0 表示没有促销，1 表示有促销，数值型，没有空值；

#StateHoliday：表示州假期，一般情况州假期商店关门，少数店铺开门，类别型；

#SchoolHoliday：表示学校假期，0 表示学校假期关门，1 表示学校假期照常营业；

#Id：在测试集中象征具体一个店铺和具体销售日期双重特征，数值型。

3.数据清洗

（1)缺失值 NaN 用 0 填充,包括 store 数据集中 PromoSinceYear 和 PromoSinceWeek 字段，竞争店铺距离 CompetitionDistance 用中位数填充；test 数据集中 Open 字段补为 1，表示营业状态；

（2）分别合并训练集（train）和店铺集（store），测试集（test）和店铺集（store）；

（3）数值化特征值。原始数据集中 StateHoliday 分类特征取值为 0、a、b、c，无法代人模型计算，重新编码为 0、1、2、3；StoreType 分类特征值为 a、b、c、d，Assortment 分类特征值为 a、b、c，采用整数型重新编码；

（4）分解特征。将原始数据集中 Date（年-月-日）特征分解为 Day、Month、Year、 Date（年-月）4 个特征；

（5）增加字段 PromoMonth 和 CompetitionMonth。将 CompetitionOpenSinceYear 和 CompetitionOpenSinceMonth 字段转化为整数型并以月为基准；同理字段 Promo2SinceYear 和 Promo2SinceWeek。

2.2 探索性可视化

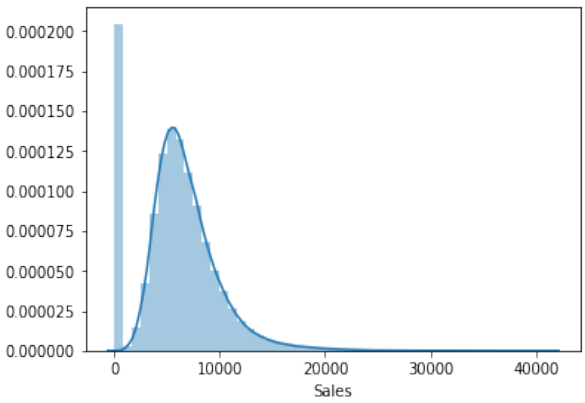
应用 Numpy,Pandas 和 seaborn 模块进行原始数据分析和可视化。分析其他特征和 Sales 的关系并绘制可视化图表，例如 DayOfWeek、Month、Promo、StateHoliday、SchoolHoliday、StoreType、Assortment、CompetitionDistance 等。

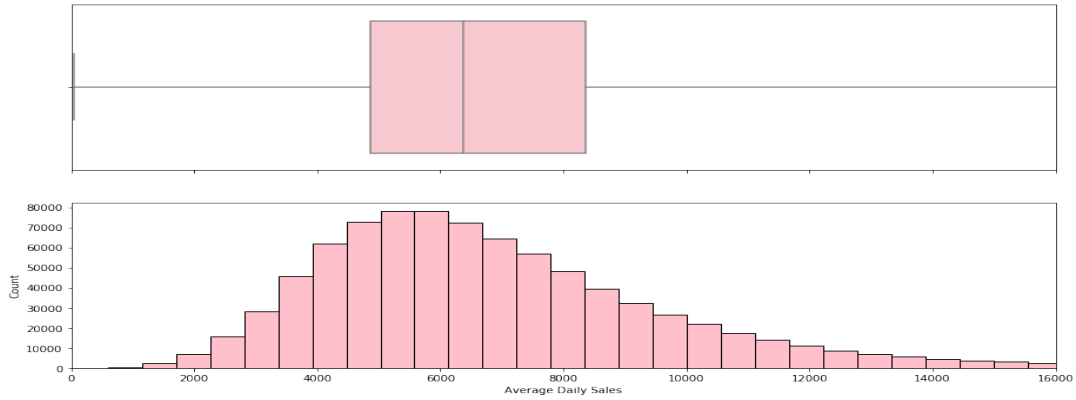
1.分析单变量 Sales

Sales 是我们去预测的值。

（1）描述统计结论：均值、最值、中值、分位数、样本数如图所示。最小值为 0，最值在可接受范围内，每日销售额集中在 4000-8000。

```
train['Sales'].describe()
count    1.017209e+06
mean     5.773819e+03
std      3.849926e+03
min      0.000000e+00
25%      3.727000e+03
50%      5.744000e+03
75%      7.856000e+03
max      4.155100e+04
Name: Sales, dtype: float64
```





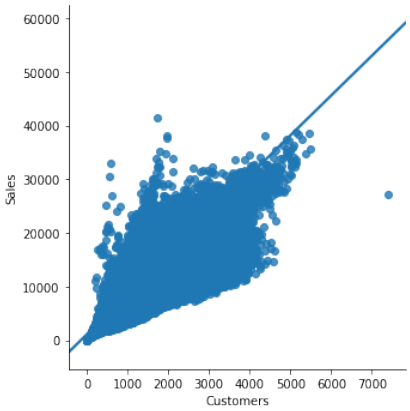
（2）Sales 分布

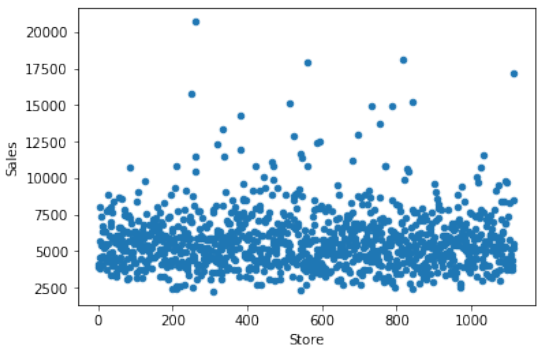
偏离正态分布，数据正偏，有峰值。

2.数据间关系

（1）Sales 与 Customers/Store 关系

Sales 与 Customers 成线性相关，销售额越高的店铺客户数量也越大；Sales 在各个 Store 间分布相对均匀，特殊销售额店铺在可以接受范围。

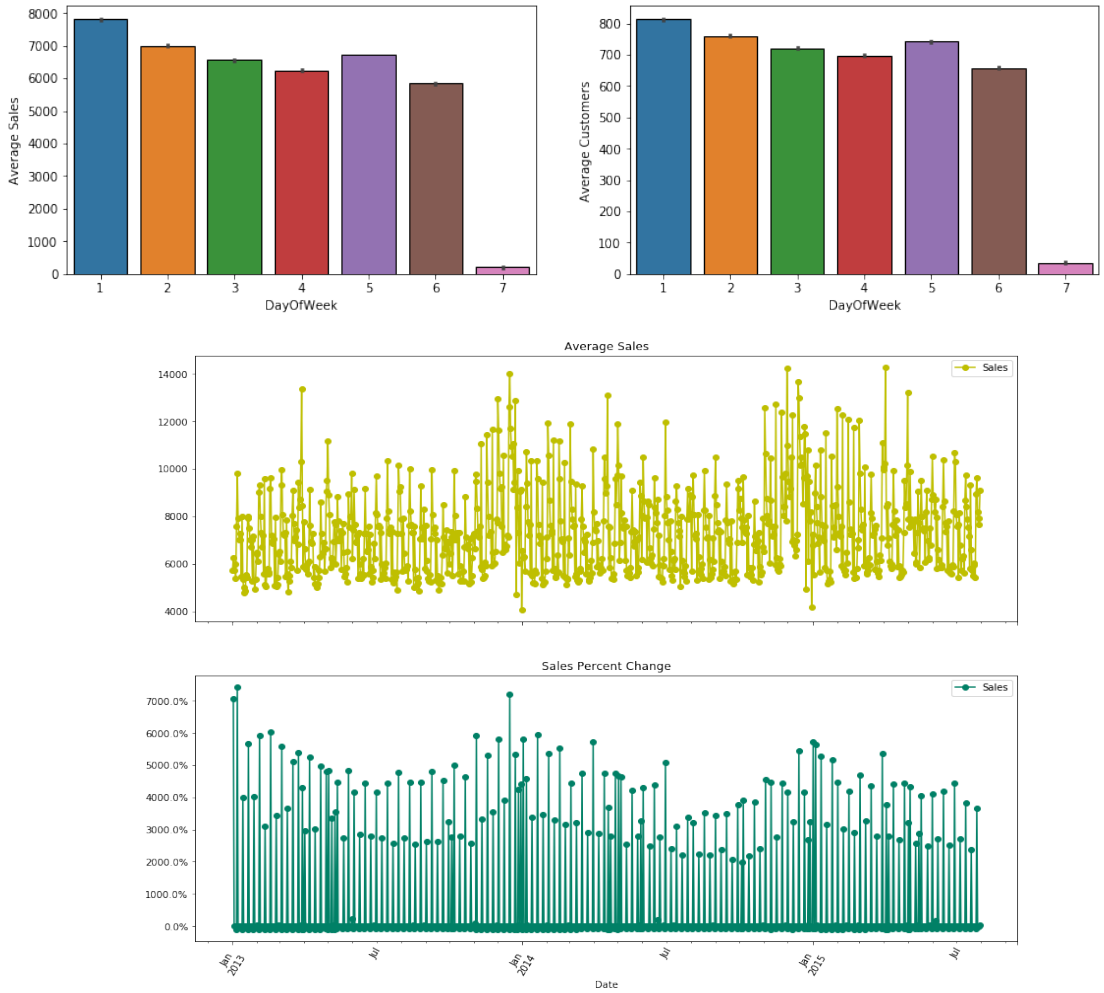




（2）Sales 时间特点

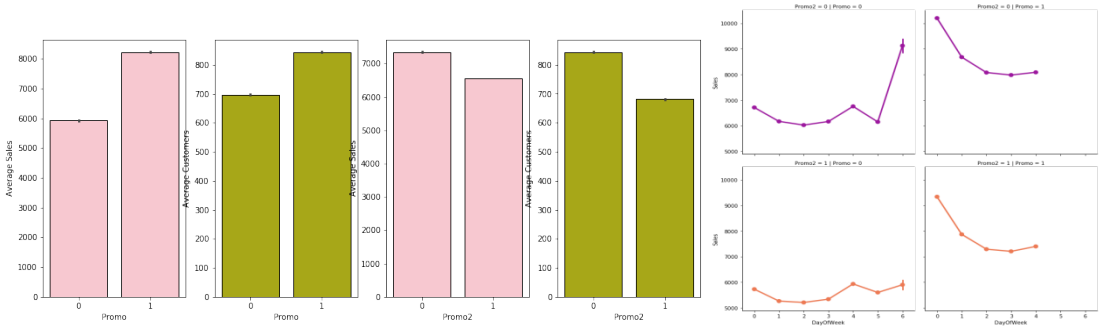
由于是时序数据集，所以可视化分析以时间为维度分析数据。销售额在星期一最好，星期二、星期五相对好，客户数量与销售额一致。11，12，1 月份销售最大，可能与圣诞节和新年有

关；3-7 月份为销售旺季；2 和 9 月份销售额最少。每年的销售额变化与月份销量变化一致。



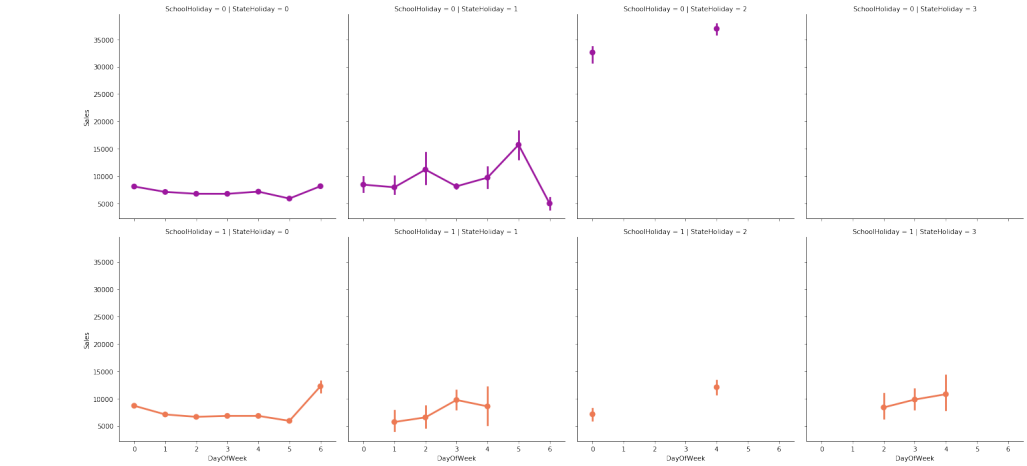
（3）Promo 与 Sales、Customers 关系

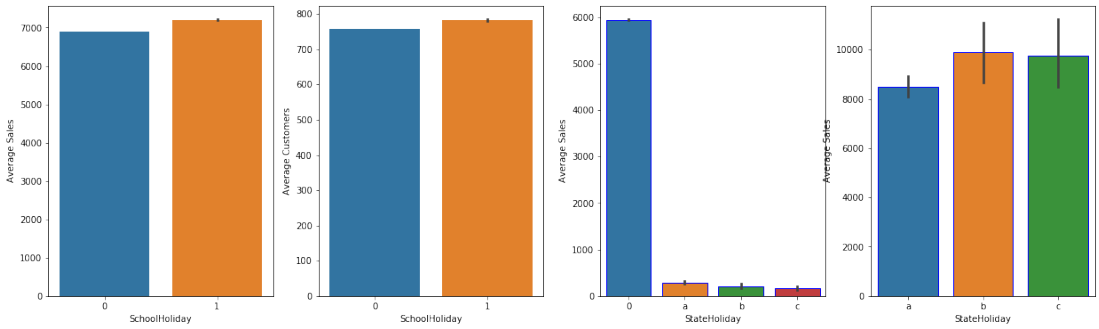
实行 Promo 促销，不实行 Promo2 促销星期一的销售额最高；实行 Promo2 促销，不实行 Promo 促销的销售额最低。做促销比不促销的平均销售额多 2000 以上，客户多 100。



（4）Sales 与 StateHoliday/SchoolHoliday 关系

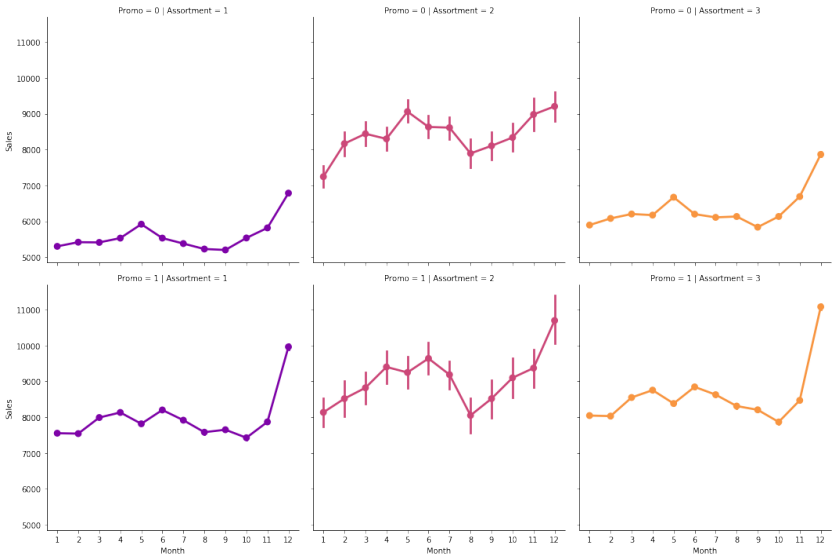
在 a\b\c 类州假期平均销售额不不在州假期高，b 类州假期平均销售额最高，接近 10000。在学校假期平均销售额和平均客户数量相对多一些。





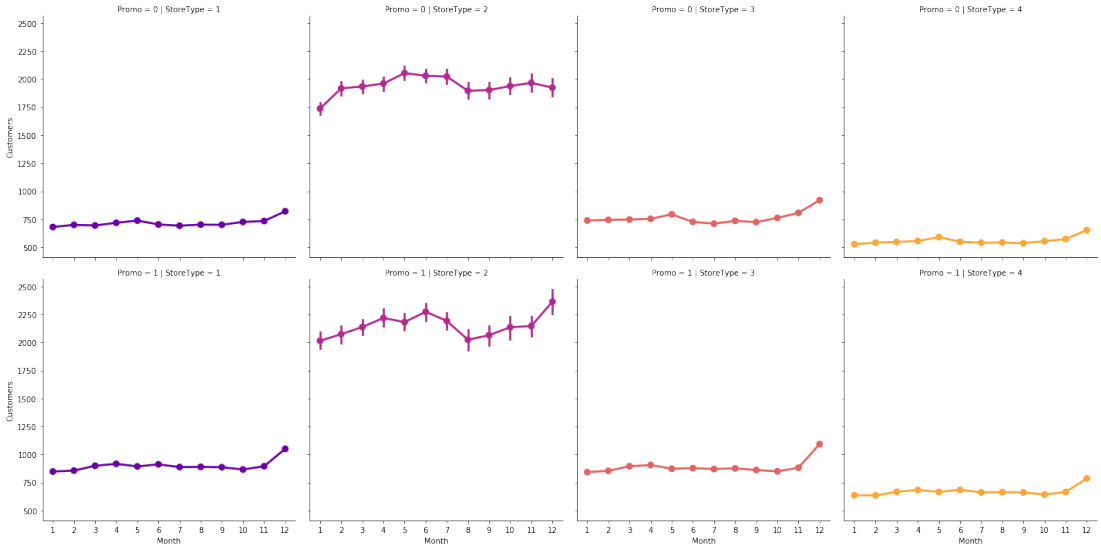
（5）Sales 与 Promo/Assortment 关系

Extra 类型店铺销售额相对较高，促销与否对其影响不大；Basic 和 extended 类型店铺销售额相对较小，促销对销售额有一定促进作用。



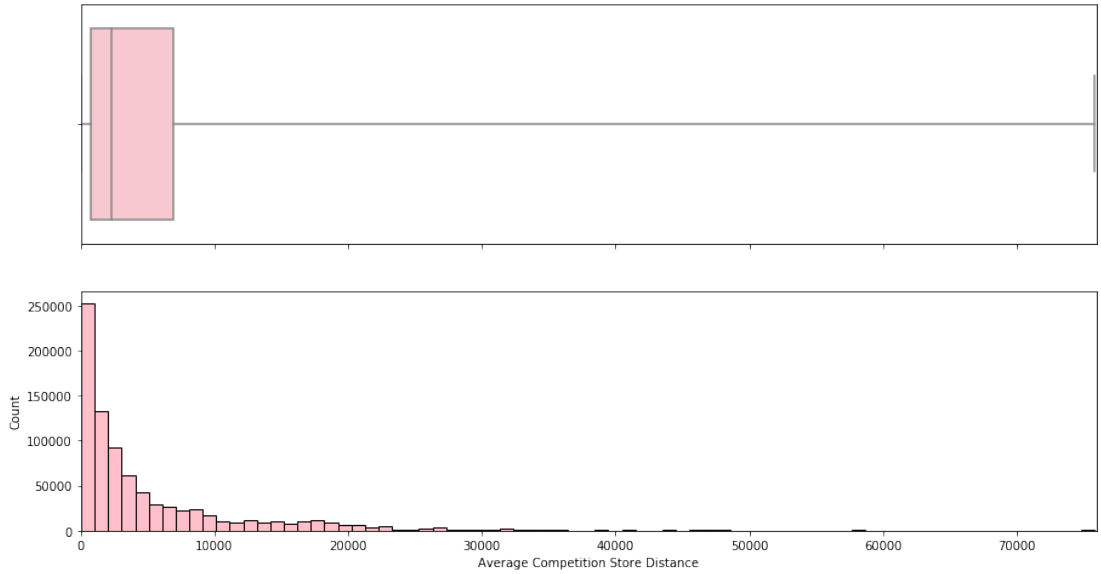
（6）Sales 与 StoreType/Promo 关系

B 类销售额相对高一些，D 类店铺销售额相对低一些，促销对各个类型店铺的销售额影响缓和。



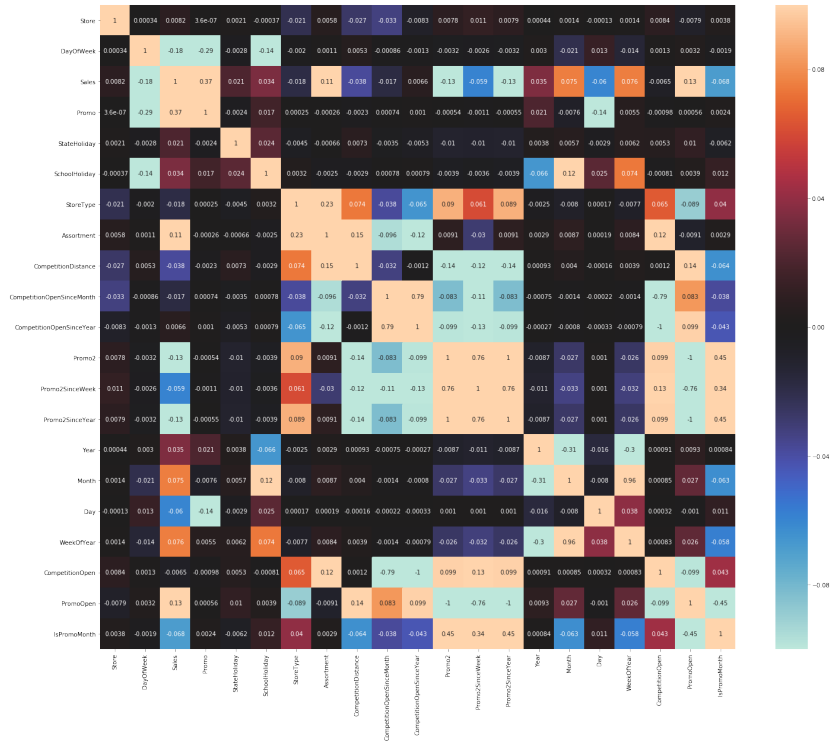
（7）CompetitionDistance 分布

平均竞品店铺距离集中在 10000 米以内。



（8）各个因素间关系

Month、WeekOfYear 对 Sales 影响在 0.07-0.08 之间，最高；Promo、Year、SchoolHoliday 对 Sales 影响在 0.03-0.04 之间，中间；还有负相关 Promo2、Promo2SinceYear、Promo2SinceWeek、CompetitionOpenSinceMonth、CompetitionDistance、StoreType、DayOfWeek 等。



2.3 算法和技术

XGBoost 原理[7]：XGBoost：eXtreme Gradient Boosting

- 可自定义损失函数：损失函数采用二阶近似，二阶泰勒展开；
- 规范化的正则项：叶子节点数目、叶子节点的分数
- 建树与剪枝：先建完全树后剪枝
  - ✓ 支持分裂点近似搜索
  - ✓ 稀疏特征处理
  - ✓ 缺失值处理
- 特征重要性与特征选择
- 并行计算

1.损失函数

由于函数在点 x 的泰勒展开形式为 n>=3:

$$f(x+\Delta x) \simeq f(x)+f'(x)\Delta x+\frac{1}{2!}f''(x)\Delta x^2+o(x^{(n)})$$

- XGBoost:对损失函数的二阶 Taylor 展开近似
- 在第 m 步时，令$g_{m,i}=\left[\frac{\partial L(f(x_i),y_i)}{\partial f(x_i)}\right]_{\mathrm{f=f_{m-1}}}$，$h_{m,i}=\left[\frac{\partial^2 L(f(x_i),y_i)}{\partial^2 f(x_i)}\right]_{\mathrm{f=f_{m-1}}}$

- $L(y_i, f_{m-1}(x_i) + \phi(x_i)) = L(f_{m-1}(x_i), y_i) + g_{m,i}\beta\phi(x_i) + \frac{1}{2}h_{m,i}\phi(x_i)^2$

  $L(f_{m-1}(x_i), y_i)$*与未知量*$\phi(x_i)$*无关*
- 所以$L(f_{m-1}(x_i) + \phi(x_i), y_i) = g_{m,i}\beta\phi(x_i) + \frac{1}{2}h_{m,i}\phi(x_i)^2$
- 对 L2 损失,

$$L(f(x;\theta), y) = \frac{1}{2}(f(x;\theta) - y)^2, \quad \nabla_f L(\theta) = f(x;\theta) - y, \quad \nabla_f^2 L(\theta) = 1$$

- 所以$g_{m,i} = f_{m-1}(x_i) - y_i$ ， $h_{m,i} = 1$

2.回归树概念

分类与回归是一个型号的概念，分类的结果是离散值，回归是连续的，本质是一样的，都是特征（feature）到结果/标签（label）之间的映射。分类树的样本输出（即响应值）是类的形式，如判断蘑菇是有毒还是无毒，周末去看电影还是不去。而回归树的样本输出是数值的形式，比如给某人发放房屋贷款的数额就是具体的数值，可以是 0 到 120 万元之间的任意值[5]。

- 树的定义：把树拆分成结构部分 q 和叶子分数部分 w

$$\phi(\mathrm{x}) = w_{q(x)}, \qquad w \in R^T \ , q \in R^D \ \to 1,2,3,\dots,T$$

- 结构函数 q：把输入映射到叶子的索引号
- 叶子分数函数 w：给出每个索引号对应的叶子的分数
- T 为树中叶子结点的数目，D 为特征维数

3.树的复杂度

树的复杂度定义为：T 为叶子的个数，w 为 L2 的模平方，γ 为 L1 正则的惩罚项，λ 为 L2 正则的惩罚项。

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

4.目标函数[6]

设目标函数为J(θ)

- 令每个叶子 t 上的样本集合为$I_t = \{i|q(x_i) = t\}$
- $\mathrm{J}(\theta) = \sum_{i=1}^{N} L(f(x_i;\theta), y_i) + \Omega(\theta)$
- $\cong \sum_{i=1}^{N} g_{m,i}\phi(x_i) + \frac{1}{2}h_{m,i}\phi(x_i)^2 + \gamma\mathrm{T} + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2$
- $= \sum_{i=1}^{N} g_{m,i}w_q(x_i) + \frac{1}{2}h_{m,i}w_q(x_i)^2 + \gamma\mathrm{T} + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2$
- $= \sum_{t=1}^{T}[\sum_{i\in I_t} g_{m,i}w_t + \frac{1}{2}\sum_{i\in I_t} w_t^2 h_{m,i} + \frac{1}{2}\lambda\sum_{t=1}^{T} w_t^2] + \gamma\mathrm{T}$
- $= \sum_{t=1}^{T}[\sum_{i\in I_t} g_{m,i}w_t + \frac{1}{2}(\sum_{i\in I_t} h_{m,i} + \lambda)w_t^2] + \gamma\mathrm{T}$

这一个目标函数中，包含了 T 个相互独立的单变量二次函数。我们定义为：

$$G_t = \sum_{i \in I_t} g_{m,i}$$

$$H_t = \sum_{i \in I_t} h_{m,i}$$

将其代入上式中，得到简化的代价函数

$$\mathrm{J}(\theta) = \sum_{t=1}^{T} [G_t w_t + \frac{1}{2}(H_t + \lambda) w_t^2] + \gamma \mathrm{T}$$

- 假设我们已经知道树的结构 q,
- $\mathrm{J}(\theta) = \sum_{t=1}^{T} [G_t w_t + \frac{1}{2}(H_t + \lambda) w_t^2] + \gamma \mathrm{T}$
- 则$\frac{\partial \mathrm{J}(\theta)}{\partial w_t} = G_t + (H_t + \lambda) w_t = 0$
- 得到最佳的 w：$w_t = -\frac{G_t}{H_t + \lambda}$
- 以及最佳的 w 对应的目标函数，可视为树的分数，分数越小的树越好：
- $\mathrm{J}(\theta) = -\frac{1}{2} \sum_{t=1}^{T} \left[\frac{G_t^2}{H_t + \lambda} \right] + \gamma \mathrm{T}$

在这一部分，你需要讨论你解决问题时用到的算法和技术。你需要根据问题的特性和所属领域来论述使用这些方法的合理性。你需要考虑：

- *你所使用的算法，包括用到的变量/参数都清晰地说明了吗？*
- *你是否已经详尽地描述并讨论了使用这些技术的合理性？*
- *你是否清晰地描述了这些算法和技术具体会如何处理这些数据？*

2.4 基准模型

本项目采用的基准模型是 Boosting（提升法）族算法为基础，包含如下三个要素：

1. 函数模型：Boosting 族的函数模型是叠加型的，即

$$\mathrm{F(x)} = \sum_{i=1}^{k} f_i(x; \theta_i)$$

2. 目标函数：选定某种损失函数作为优化目标

$$\mathrm{E}\{F(x)\} = E\left\{ \sum_{i=1}^{k} f_i(x; \theta_i) \right\}$$

3. 基准模型特点

（1） Boosting 族是一种框架算法，拥有系列算法，如 AdaBoosting，GradientBoosting，XGBoosting 等算法。

（2） Boosting 族可以提高任意给定学习算法准确度。

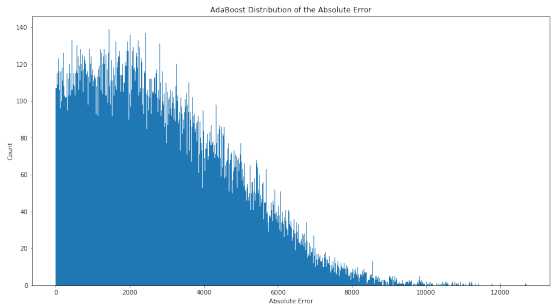
（3） 训练过程为阶梯状，弱分类器按次序一一进行训练，弱分类器的训练集按照某种策略每次都进行一定的转化，最后以一定的方式将弱分类器组合成一个强分类器。

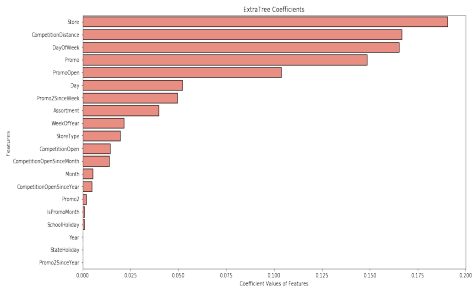
（4） Boosting 中所有弱分类器可以是不同类的分类器。

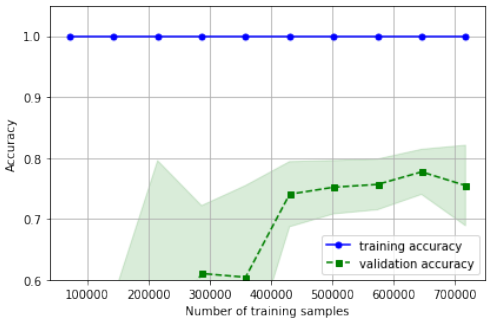
4. Boosting 族——AdaBoosting 算法结果

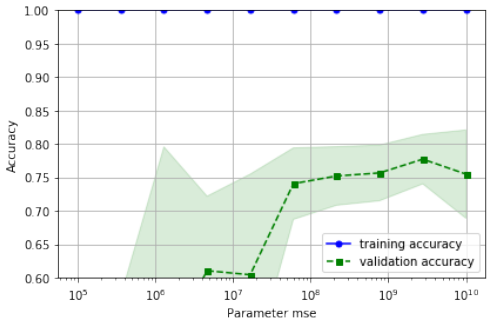
模型得分为负数，拟合度欠佳，需要使用得分更好的模型，经过多次尝试，AdaBoosting 算法结果如下。

Mean Squared Error	12390227.869364351
R2 Error	-0.29700975749572933
Median Absolute Error	2621.8064737582004
Mean	2917.703738140398
Variance(Sigma2)	3877232.7658052826
Sigma	1969.0690099144017
Training Score	-0.297010
Valid Score	-0.297010









对特征工程后的所有特征用 XGBoost 模型进行训练，XGBoost 模型有 3 种类型的参数：通用参数、辅助参数和任务参数。通用参数确定上升过程中上升模型类型，常用树或线性模型；辅助参数取决于所选的上升模型；任务参数定义学习任务和相应的学习目标。

XGBoost 模型中，常用参数说明如下：

- Booster：设置需要使用上升模型，可选 gbtree(树)或 gblinear(线性函数)，默认为 gbtree。
- Nthread：XGBoost 运行时 CPU 并行线程数，默认系统可以获得的最大可用线程数。
- Eta：收缩步长，即学习速率，取值范围是[0,1]， 默认为 0.3。在更新叶子节点的时候，权重乘以 eta，以避免在更新过程中的过拟合。
- Max_depth：每棵树的最大深度，默认为 6。树越深，越容易过拟合。
- Subsample：训练的实例样本占整体实例样本的比例，取值范围是(0，1]。值为 0.5 时意味着 XGBoost 随机抽取一半的数据实例来生成树模型，这样能防止过拟合。
- Colsample_bytree：在构建每棵树时，列(特征)采样，参数值的范围是(0，1]。
- Objective：默认为 reg：linear。
- Seed：随机数种子，为确保数据的可重现性。
- Lambda：控制模型复杂度的权重值得 L2 正则化参数，参数越大，模型越不容易过拟合
- Gamma：用于控制是否后剪枝的参数，越大越保守。

III. 方法

3.1 数据预处理

1.定义特征函数

定义特征选取函数,转化和处理数据特征。首先将其他字符表示分类的特征转化为数字；其次将时间特征进行拆分和转化，并加入 WeekOfYear 特征；再次新增 CompetitionOpen 和 PromoOpen 特征，计算某天某店铺的竞争对手的已开业时间和店铺已促销时间；最后将 PromoInterval 转化为 IsPromoMonth 特征，表示某天某店铺是否处于促销月，1 表示是，0 表示否。

2.数据集特征转化

根据定义的函数运行转化训练集、验证集和测试集进行特征转化。之后删除掉训练集和验证集中不需要的特征：Id、Date、Customers、Open、PromoInterval、monthStr。

3.拆分特征与标签

将需要处理的 Sales 从 x_train,x_valid 中拆分出来,放入 y_train、y_valid 标签中，并将标签取对数处理，对数处理使得数据集更稳定，有利于预测结果。

4.划分数据集

由于数据集是时间序列，所以按照时间排序，并且分出训练集和验证集。

```
train = train.sort_values(['Date'],ascending = False)
valid = train[:6*7*1115]
train = train[6*7*1115:]
```

3.2 执行过程

1.初始 XGBoost 模型——参数设置

参数	值
nthread	7
seed	1000
objective	Reg:linear
nrounds	20000
Early.stop.round	250
eta	0.3
Max_depth	12
subsample	0.7
Colsample_bytree	0.7
Booster	Gbtree
gamma	0.1
lambda	2
Min_child_weight	3

初始训练结果：RMSPE: 0.081856

Mean Squared Error: 0.005528550403082787

R2 Error: 0.9695010611621038

Median Absolute Error: 0.04349681132121841

Mean: 0.05469884634102282

Variance(sigma2): 0.0025365866120440174

Sigma: 0.05036453724640005

3.3 完善

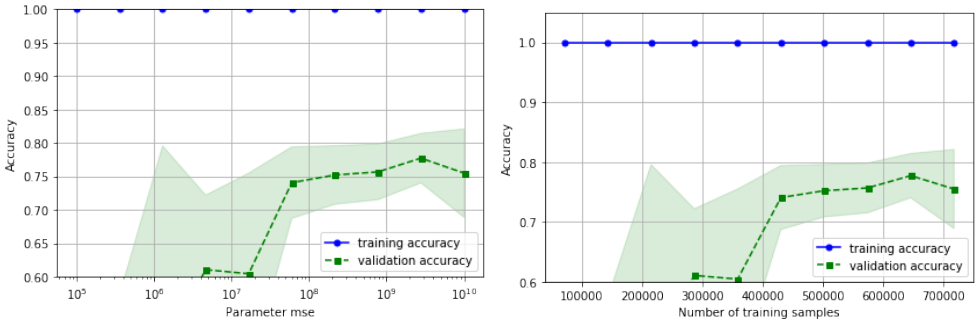
1.参数优化

（1）交叉验证

基本思想就是把训练数据分成几份，分别为训练集和测试集，用训练集和验证集作为模型选择。最典型的是 k 折交叉验证。由于本数据集是时间序列数据集，所以使用交叉验证不但没有使得结果提升，反而不利于预测。

（2）学习曲线和验证曲线

通过绘制模型训练和验证准确性关于训练集大小的函数，我们能很容易地诊断出模型是高方差还是高偏差。学习曲线是训练集数量与准确性之间的函数。而验证曲线是不同的模型参数与准确性之间的函数。尝试后发现在训练集范围内不能很好地拟合，不能很好地指导调节算法的各项参数。



（3）网格搜索 GridSearch

网格搜索的思路其实很简单，就是列举出所有你想要调节的参数，然后穷举出所有参数组合，最后得出一个使模型性能最好的参数组合[10]。有两种类型的参数：一个是从训练集中学得的参数，例如逻辑回归的权重;另一个是为了使学习算法达到最优化可调节的参数，例如逻辑回归中的正则化参数或决策树中的深度参数。这种可调节的参数称为超参数（hyperparameters）。上面我们用验证曲线调节超参数中的一个参数来优化模型。现在，我们要用网格搜索这个更加强大的超参数优化工具来找到超参数值的最优组合从而进一步改善模型的性能。本方法使用后运行调参速度慢，在实际使用中耗时太久不能很好地提升效率。

（4）贝叶斯优化

常用的调参方式有 grid search 和 random search ，grid search 是全空间扫描，所以比较慢，random search 虽然快，但可能错失空间上的一些重要的点，精度不够，于是，贝叶斯优化出现了。

hyperopt 是一种通过贝叶斯优化来调整参数的工具，对于像 XGBoost 这种参数比较多的算法，可以用它来获取比较好的参数值[9]。xgboost 具有很多的参数，把 xgboost 的代码写成一个函数，然后传入 fmin 中进行参数优化，将交叉验证的 auc 作为优化目标。auc 越大越好，由于 fmin 是求最小值，因此求-auc 的最小值。所用的数据集是 202 列的数据集，第一列样本 id，最后一列是 label,中间 200 列是属性。

Hyperopt 提供了一个优化接口，这个接口接受一个评估函数和参数空间，能计算出参数空间内的一个点的损失函数值。用户还要指定空间内参数的分布情况。

Hyheropt 四个重要的因素：指定需要最小化的函数，搜索的空间，采样的数据集(trails database)（可选），搜索的算法（可选）。

本方法运行速度快，运行结果如下：0.18347888920656855

max_depth:13

n_estimator:50

learning_rate:0.09

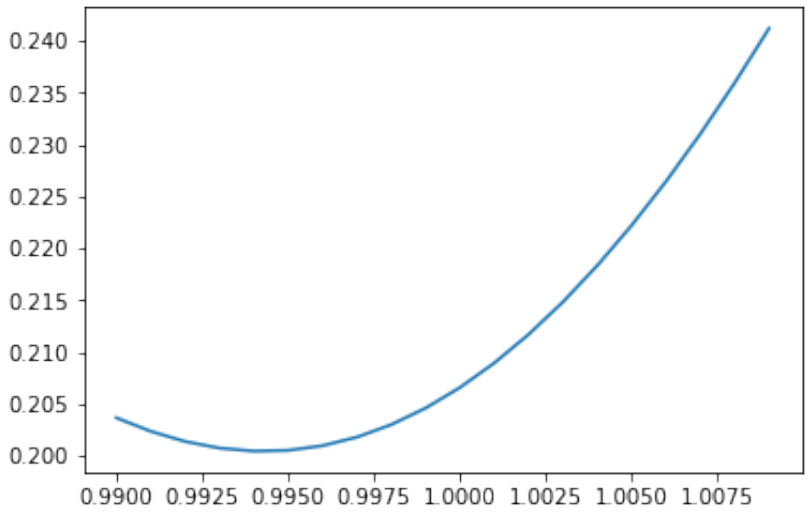
subsample:0.7

min_child_weight:3

2. 偏差分析优化

构建保留数据及预测结果，分析保留数据集中任意三个店铺的预测结果，分析偏差最大的 10 个预测结果，从分析结果看，初始模型已经可以比较好的预测验证集的销售趋势，但是相对真实值，我们的模型的整体偏差偏高一些。

当校正系数为 0.995 时，RMSPE 得分最低，为 0.1189，相对于初始模型 0.1254 得分有所提升。因为每个店铺都有自己的特点，我们设计的模型对不同店铺的偏差并不完全相同，所以我们需要做细致矫正。以不同的店铺分组进行细致矫正，每个店铺分别计算取得的最佳 RMSPE 得分的校正系数。



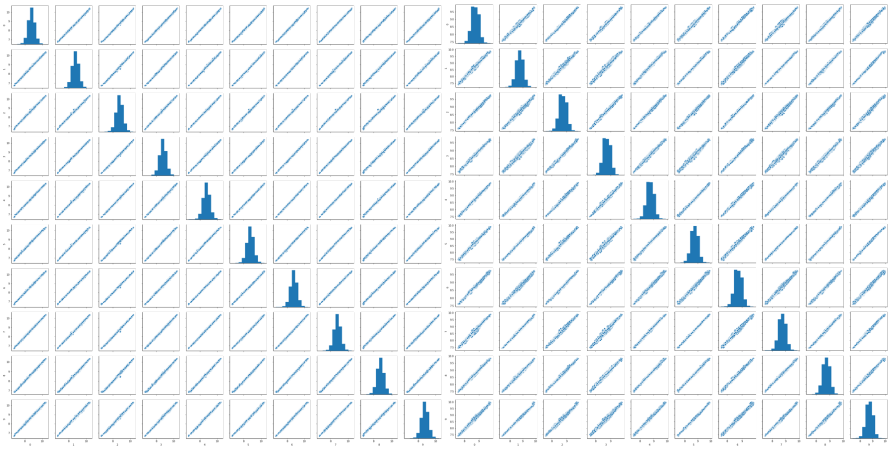
3. Bagging 模型融合优化

给定一个大小为 n 的训练集 D，Bagging 算法从中均匀、有放回地选出 m 个大小为 n’的子集 Di，作为新的训练集。在这 m 个训练集上使用分类、回归等算法，则可得到 m 个模型，再通过取平均值、取多数票等方法综合产生预测结果，即可得到 Bagging 的结果[11]。

用不同 seed 训练 10 个模型，每个模型单独进行偏差校正然后进行融合。分析模型的相关性，模型融合可以采用平均或者加权重方法，从图上看，模型相关性较高，分别进行简单平均融合和加权融合。权重模型较均值模型有比较好的得分。分别用平均融合和加权融合进行预测。

Bagging 阶段分别采用 XGBoost 算法和 LightBoost 算法进行十轮的 bagging 运算。XGBoost 验证结果平均 RMSPE=0.114272，加权融合 RMSPE=0.113477，加权融合有所提升。模型融合相关性如左图所示，XGBoost 模型相关性高。

LightGBM 十轮模型融合 RMSPE=0.112562，比 XGBoost 有所提升。模型相关性如右图所示，模型相关性高。



IV. 结果

4.1 模型的评价与验证

1.模型评价

由于 XGBoosting 模型没有自带的 score 评价结果函数，所以本项目使用自定义评估函数结合 kaggle 网站给与的评估模型相结合的方式评价训练结果。

由于 XGBoost 函数库没有模型评估函数,所以本项目采用自定义函数用于项目内每次运算评估，1-2 上面两项指标用于模型间评估。

（1）RMSE

RMSE（root mean square error,平方根误差），定义为：

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

其中，y_i是真实值，$\hat{y}_i$是预测值，n 是样本数量，使用了欧式距离。

缺点：平均值是非鲁棒的，对于异常点敏感，如果某个异常点误差大，整个 RMSE 就会比较大。

（2）RMSPE

XGBoost 的 best_iteration 和 best_score 均是基于评价函数得出。XGBoost 中对于评价函数调用时同样会传入 preds 和 dvalid，即为验证集和验证集上的预测值，返回值为一个字符串标识自定义评价函数的类型和一个 flaot 类型的 fevalerror 值表示评价值的大小，其是以 error 的形式定义，即当此值越大是认为模型效果越差。XGBoost 官方库中不支持以 F1 Score 来作为评价函数，用户可同过自定义 feval 实现。

自定义 feval 用 rmspe 来表示：$\mathrm{RMSPE} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}/y-1)^2}{n}}$

其中，y_i是真实值，$\hat{y}_i$是预测值，n 是样本数量。

2.模型验证

（1）Hyperopt 最优参数训练

参数	值
nthread	7
seed	10
objective	Reg:linear
nrounds	4000
Early.stop.round	100
eta	0.09
Max_depth	13
subsample	0.7
Colsample_bytree	0.7
Booster	Gbtree
gamma	0.1
lambda	2

Min_child_weight	3

结果：train-rmse:0.074167 train-rmspe:0.075866

2.学习率调节

参数	值
nthread	7
seed	10
objective	Reg:linear
nrounds	6000
Early.stop.round	100
eta	0.03
Max_depth	13
subsample	0.7
Colsample_bytree	0.7
Booster	Gbtree
gamma	0.1
lambda	2
Min_child_weight	3

结果：train-rmse:0.079958 train-rmspe:0.085944. 收敛慢，运行增加 2000 效果不及前面。

（3）max_depth 和 subsample 调节

参数	值
nthread	7
seed	10
objective	Reg:linear
nrounds	3000
Early.stop.round	100
eta	0.03
Max_depth	10
subsample	0.9
Colsample_bytree	0.7
Booster	Gbtree
gamma	0.1
lambda	2
Min_child_weight	3

结果：train-rmse:0.068427 train-rmspe:0.074751.效果优于最大深度 max_depth=13。

（5） 再次调节学习率

参数	值
nthread	7
seed	10

objective	Reg:linear
nrounds	3000
Early.stop.round	100
eta	0.09
Max_depth	10
subsample	0.9
Colsample_bytree	0.7
Booster	Gbtree
gamma	0.1
lambda	2
Min_child_weight	3

结果：train-rmse:0.076433 train-rmspe:0.07977，效果较学习率 0.03 差。

（6）确定参数

参数	值
nthread	7
seed	10
objective	Reg:linear
nrounds	3000
Early.stop.round	100
eta	0.03
Max_depth	10
subsample	0.9
Colsample_bytree	0.7
Booster	Gbtree
gamma	0.1
lambda	2
Min_child_weight	3

结果：train-rmse:0.067051 train-rmspe:0.07124，目前效果最优参数。

4.2 合理性分析

1.设置迭代次数和早起停止

XGBoost 模型设置初始模型迭代 4000 次，early_stopping_rounds=100,也就是 RMSPE 在 100 次内没有提升，停止迭代，在 3567 次停止，最佳 Best Iteration 在 3467 次，[3467] train-rmse:0.081949 valid-rmse:0.118754 train-rmspe:0.092733 valid-rmspe:0.128244。

LightGMB 模型设置初始迭代 4000 次，early_stopping_rounds=5,也就是 RMSPE 在 5 次内没有提升，停止迭代，每轮均在 1300 次内停止迭代，产生最优值，例如：Early stopping, best iteration is:[1155] valid_0's rmse: 0.236086 valid_0's l2: 0.0557364

round 10 end

RMSPE: 0.112562

2.验证集验证

XGBoost 读入验证集，计算 Ratio、Error、Weight 三个值合理。

Sales	Prediction	Ratio	Error	Weight
8.568456	8.611975	1.005079	0.005079	0.994947
8.521185	8.508659	0.998530	0.001470	1.001472
8.472614	8.422850	0.994126	0.005874	1.005908
8.519391	8.450933	0.991965	0.008035	1.008101
8.716372	8.561447	0.982226	0.017774	1.018096

LightGMB 选取误差最大十个店铺，如图所示，最大误差也在可以接受范围。

Sales	Prediction_lgb	Ratio_lgb	Error_lgb	Weight_lgb
6.919684	8.615549	1.245078	0.245078	0.803162
6.555357	7.899639	1.205066	0.205066	0.829830
10.634677	8.751360	0.822908	0.177092	1.215203
6.343880	7.454574	1.175081	0.175081	0.851005
7.259820	8.464497	1.165938	0.165938	0.857679
6.806829	7.885318	1.158442	0.158442	0.863228
7.300473	8.438605	1.155898	0.155898	0.865128
10.310219	8.712185	0.845005	0.154995	1.183425
7.050123	8.114865	1.151025	0.151025	0.868791
7.406103	8.515004	1.149728	0.149728	0.869771

V. 项目结论

5.1 结果可视化

1. 模型验证可视化

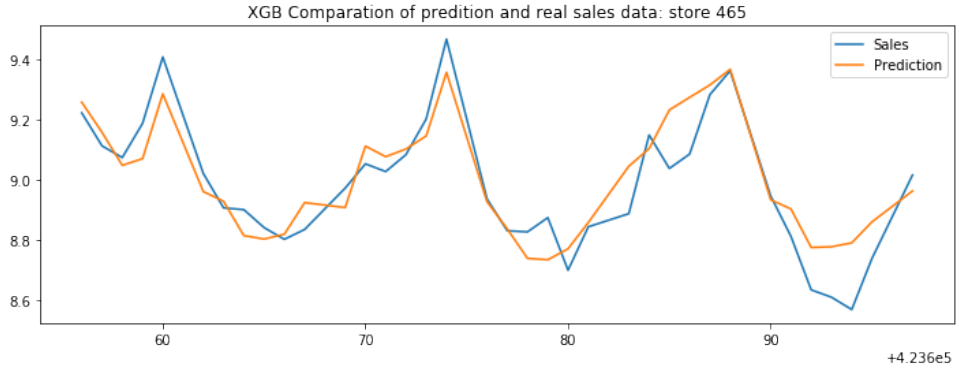
XGBoost 模型训练后在验证集中任意选取三个店铺预测结果，选取其中之一如下图所示：

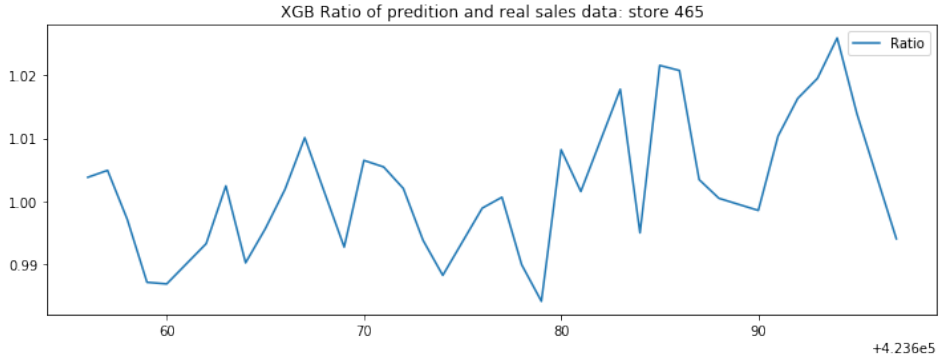
XGB Mean Ratio of predition and real sales data is 1.0024277948900546: store all

XGB Mean Ratio of predition and real sales data is 1.0023449196553966: store 465

XGB Mean Ratio of predition and real sales data is 1.0016105088423435: store 33

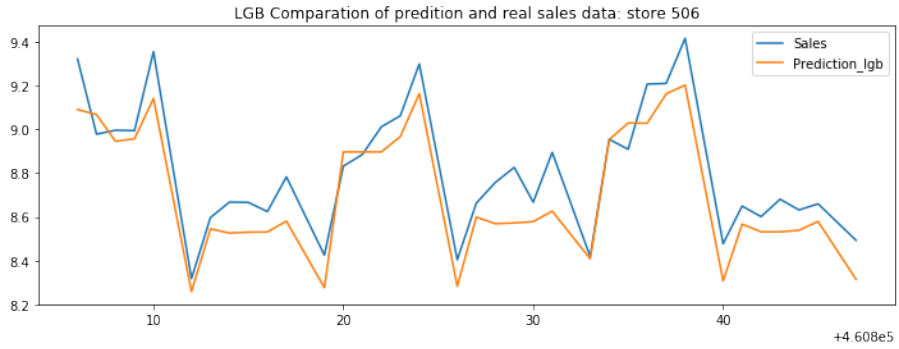
XGB Mean Ratio of predition and real sales data is 0.9992126084278942: store 500

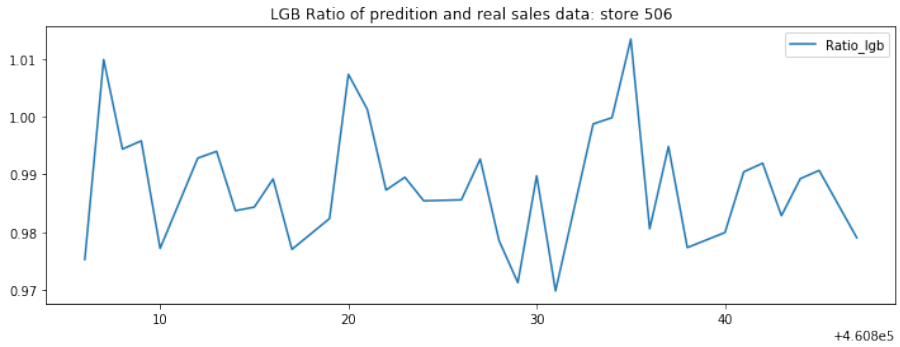




预测结果与真实值拟合度较好。

LightGMB 模型训练后同样在验证集中选取三家验证，如图所示，拟合度较好，模型泛化能力较强。



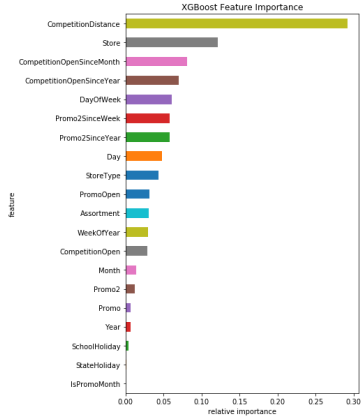


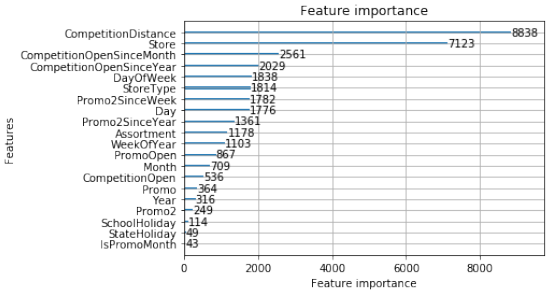
2.模型特征重要性

从左图 XGBoost 模型特征重要性分析，比较重要的特征包括：DayOfWeek，CompetitionDistance，WeekOfYear，Store；可见店铺的销售额与时间是息息相关的，尤其是周期较短的时间特征；2.店铺差异'Store'和'StoreTyp'特征，不同店铺的销售额存在特异性；3.短期促销（Promo)情况:'PromoOpen'和'Promo'特征，促销时间的长短与营业额相关性比较大；4.竞争对手相关特征包括：'CompetitionOpen'，‘CompetitionDistance'，'CompetitionOpenSinceMoth'以及'CompetitionOpenScinceyear'，竞争者的距离与营业年限对销售额有影响。

作用不大的特征主要两类包括：1.假期特征：'SchoolHoliday'和'StateHoliday'，假期对销售额影响不大，有可能是假期店铺大多不营业，对模型预测没有太大帮助。2.持续促销(Promo2)相关的特征：'Promo2'，'Prom2SinceYear'以及'Prom2SinceWeek'等特征，有可能持续的促销活动对短期的销售额影响有限。

右图 LightGMB 基本相同。

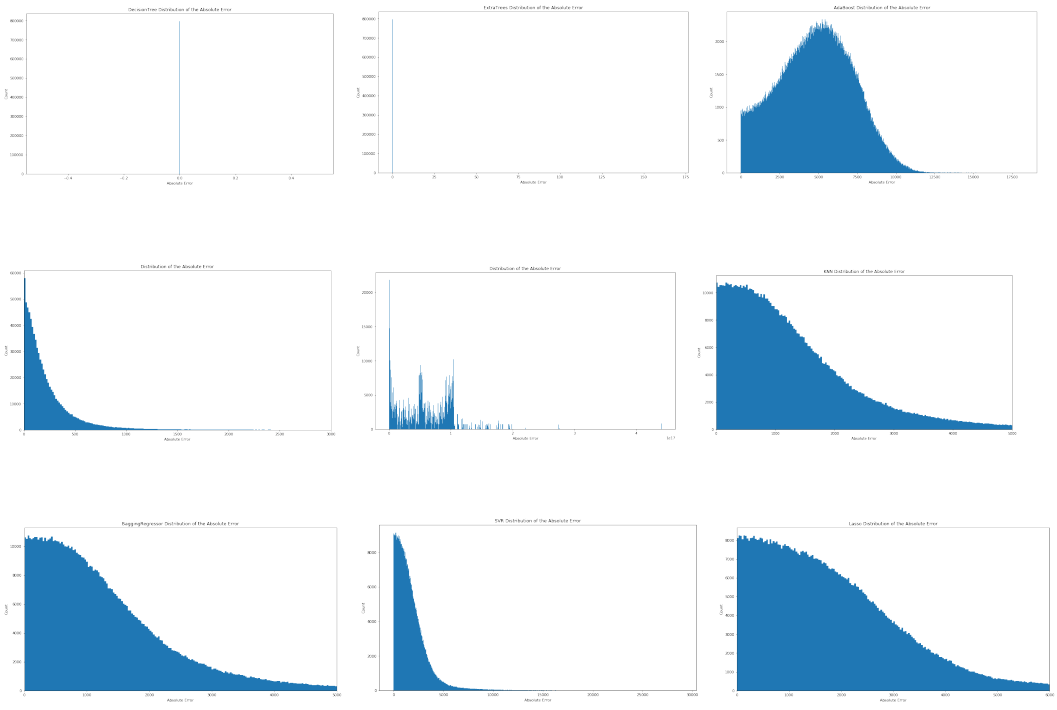


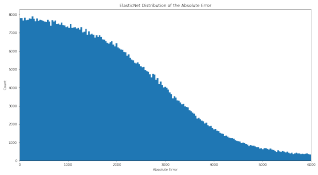
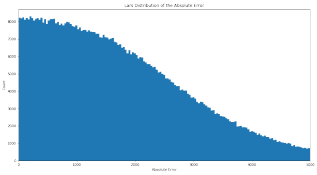


5.2 对项目的思考

1. 难点——模型选取

项目难度在于选取模型和选取模型后的调节参数，以及后续模型优化，每一次优化都仅仅有很小的进步，有没有可用的工具记录计量评估这种微小的差异，使得工作效率不高。下图是确定模型前的比较工作：



2.难点——模型参数调优

最终选择 XGBoost 算法，但是 XGBoost 算法参数众多，各个参数有都会对结果造成影响，选取参数耗费时间，并且使用 GridSearch、RandomSearch、Hyperopt 等工具耗时巨大。如下来自官方翻译过来的参数，供参考。

Booster 参数：

（1）eta[默认是 0.3] 和 GBM 中的 learning rate 参数类似。通过减少每一步的权重，可以提高模型的鲁棒性。典型值 0.01-0.2

（2） min_child_weight[默认是 1] 决定最小叶子节点样本权重和。当它的值较大时，可以避免模型学习到局部的特殊样本。但如果这个值过高，会导致欠拟合。这个参数需要用 cv 来调整

（3） max_depth [默认是 6] 树的最大深度，这个值也是用来避免过拟合的 3-10

（4） max_leaf_nodes 树上最大的节点或叶子的数量，可以代替 max_depth 的作用，应为如果生成的是二叉树，一个深度为 n 的树最多生成 2n 个叶子,如果定义了这个参数 max_depth 会被忽略

（5）gamma[默认是 0] 在节点分裂时，只有在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。这个参数值越大，算法越保守。

（6）max_delta_step[默认是 0] 这参数限制每颗树权重改变的最大步长。如果是 0 意味着没有约束。如果是正值那么这个算法会更保守，通常不需要设置。

（7）subsample[默认是 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。但是这个值设置的过小，它可能会导致欠拟合。典型值：0.5-1

（8）colsample_bytree[默认是 1] 用来控制每颗树随机采样的列数的占比每一列是一个特征 0.5-1

（9）colsample_bylevel[默认是 1] 用来控制的每一级的每一次分裂，对列数的采样的占比。

(10)lambda[默认是 1] 权重的 L2 正则化项

(11)alpha[默认是 1] 权重的 L1 正则化项

(12)scale_pos_weight[默认是 1] 各类样本十分不平衡时，把这个参数设置为一个正数，可以使算法更快收敛。

通用参数：

(1)booster[默认是 gbtree]

选择每次迭代的模型，有两种选择：gbtree 基于树的模型、gbliner 线性模型

(2)silent[默认是 0]

当这个参数值为 1 的时候，静默模式开启，不会输出任何信息。一般这个参数保持默认的 0，这样可以帮我们更好的理解模型。

(3)nthread[默认值为最大可能的线程数]

这个参数用来进行多线程控制，应当输入系统的核数，如果你希望使用 cpu 全部的核，就不要输入这个参数，算法会自动检测。

学习目标参数：

(1)objective[默认是 reg：linear]

这个参数定义需要被最小化的损失函数。最常用的值有：binary：logistic 二分类的逻辑回归，返回预测的概率非类别。multi:softmax 使用 softmax 的多分类器，返回预测的类别。在这种情况下，你还要多设置一个参数：num_class 类别数目。

(2)eval_metric[默认值取决于 objective 参数的取之]

对于有效数据的度量方法。对于回归问题，默认值是 rmse，对于分类问题，默认是 error。典型值有：rmse 均方根误差；mae 平均绝对误差；logloss 负对数似然函数值；error 二分类错误率；merror 多分类错误率；mlogloss 多分类损失函数；auc 曲线下面积。

(3)seed[默认是 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【12】。

3.模型优化

即使模型参数最优后预测的结果与真实值之间仍会有差距，这个时候就需要模型优化，例如 bagging,stacking,pipeline 等技术，但是消耗时间都较大。

5.3 需要作出的改进

需要改进的地方还有很多，本项目只是在模型的参数和后续预测时给与了一定的优化，但是本项目没有对比各个模型组合之间的差别，也许尝试更多模型的不同组合会有不同的结果，有些模型组合可能会更高效更节约时间。

参考文献：

【1】https://www.cnblogs.com/peizhe123/p/7412364.html

【2】https://baike.baidu.com/item/机器学习/217599?fr=aladdin

【3】https://blog.csdn.net/qunnie_yi/article/details/80129857

【4】叶倩怡　饶泓　姬名书基于Xgboost的商业销售预测[J].南昌大学学报（理科班），2017，3：275.Doi:10.3969/j.issn.1006-0464.2017.03.015

【5】https://blog.csdn.net/github_38414650/article/details/76061893

【6】https://blog.csdn.net/g11d111/article/details/73409811

【7】https://xgboost.readthedocs.io/en/latest/tutorials/index.html

【8】https://blog.csdn.net/shine19930820/article/details/78335550?utm_source=blogxgwz4

【9】https://blog.csdn.net/u014084065/article/details/78543523

【10】https://blog.csdn.net/xlinsist/article/details/51344449

【11】https://blog.csdn.net/weixin_38569817/article/details/80534785?utm_source=blogxgwz0

【12】https://blog.csdn.net/u013963380/article/details/72731477