基于 XGBoost 预测 Rossmann 未来的销售额
项目报告

张承博

2019 年 1 月 20 日

目录

I. 问题的定义

1.1 项目概述

Rossmann 公司在 7 个欧洲国家经营超过 3000 家连锁药店。目前，Rossmann 的商店经理需要提前预测未来 6 周的日销售情况。商店的销售情况受多个因素的影响，如促销、竞争、节假日、季节性和地方性。由于数千个商店经理基于其独特的环境预测销售情况，预测结果的准确性可能会有很大差异。可靠的销售预测能够使商店经理创建有效的员工时间表，从而提高其生产率和动力 [1]。Rossmann 于 2015 年 9 月在 Kaggle 开启该项目竞赛，即开发一个稳健模型以预测其在德国境内 1115 家商店未来 6 个月的日销售情况。该竞赛已提供 1115 家商店历史销售情况的数据集，该数据集包含特征（例如：商店类型、最近竞争者距离）和标签（即销售额）。本项目即该竞赛解决方案的一种具体实现。

机器学习使用计算方法直接从数据中“学习”信息，对未见数据做出预测，而不依赖于预定方程模型 [2]。当可用于学习的样本数量增加时，这些算法可自适应提高性能。机器学习采用两种技术：监督式学习和无监督学习。监督式学习根据已知的输入和输出训练模型，让模型能够预测未来输出；无监督学习从输入数据中找出隐藏模式或内在结构。

如今，基于机器学习技术对销售预测的研究逐渐成熟，较传统数据分析方法其预测能力更加强大，同时具有更高可靠性。因此，本项目使用机器学习作为解决途径。

1.2 问题陈述

本项目需依所提供的数据集构建稳定的销售额预测模型。数据集样本基于连续时序统计，包含多特征和标签。因此，该问题是一个基于多特征的监督式回归预测问题，是可被量化、可被测量且可被重现的。

该问题需进行特征工程，将原始数据转换为合适的特征向量。经初步探索可知数据集包含大量分类数据和时序数据，这些数据需要被转换为相应的数值数据。

机器学习（监督式学习）中的回归方法，具体的 XGBoost 算法，可能是潜在解决方案。该问题的最终模型需生成测试预测结果，此预测结果需被提交至 Kaggle 进行评估。

1.3 评估指标

根据竞赛发起者发布的规则，提交的预测模型均采用 RMSPE（Root Mean Square Percentage Error）评估。RMSPE 计算方法如下：

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

其中 y_i 表示单个商店单日实际销售额，$\hat{y}_i$ 表示对应预测值。RMSPE 相较于 RMSE（Root Mean Square Error），对数值的绝对大小不敏感，因此更适合大尺度变化的序列的评估。

II. 分析

2.1 探索数据

这一部分将通过数据分析方法探索数据集，将讨论数据集的不同数据类型，研究数据集的基本信息和统计量，观察数据集的缺失值和异常值。

2.1.1 原始数据

所提供的原始数据集文件均为 CSV 格式，文件包括：

- *train.csv* - 包含销售额（标签）的历史数据，即训练集
- *test.csv* - 不包含销售额（标签）的历史数据，即测试集
- *sample_submission.csv* - 正确格式的提交文件样本
- *store.csv* - 关于商店的补充信息

训练集包含从 2013-01-01 至 2015-07-31 共计 1017209 条记录，测试集包含 41088 条记录。单条记录对应某商店某日的销售情况。

数据集字段包括：

- *Sales* - 给定日期当日销售额，即标签
- *Customers* - 给定日期当日顾客量
- *Open* - 表明商店是否开门营业：0 = 关门，1 = 开门
- *StateHoliday* - 法定假日类别：a = 公众假日，b = 复活节，c = 圣诞节，0 = 非假日
- *SchoolHoliday* - 表明是否受到学校关闭影响
- *StoreType* - 商店类型：a，b，c，d
- *Assortment* - 零售商品品类级别：a = 基础，b = 附加，c = 扩展
- *CompetitionDistance* - 最近竞争者距离
- *CompetitionOpenSince[Month/Year]* - 最近竞争者开始经营的大概时间（月/年）
- *Promo* - 表明给定日期当日是否有促销活动
- *Promo2* - 表明商店是否参与连续促销活动：0 = 未参与，1 = 参与
- *Promo2Since[Year/Week]* - 开始参与连续促销活动的时间（年/周）
- *PromoInterval* - 连续促销活动开始的月份

2.1.2 导入数据

分别导入数据集文件至 `data_train`、`data_test` 和 `data_store`，即训练集、测试集和商店补充数据集。为方便随后分析，通过 *Store* 字段联合合并 `data_train/data_test` 与

`data_store` 数据集到同一数据集中。合并后的 `data_train/data_test` 包含原有字段以及对应的商店信息。

2.1.3 检查数据

问题要求预测销售额，故确定 *Sales* 为标签（label）。现实中 *Customers* 信息同样未知，故 *Customers* 不作为特征。确定特征（feature）如下：*Store*, *DayOfWeek*、*Date*、*Open*、*Promo*、*StateHoliday*、*SchoolHoliday*、*StoreType*、*Assortment*、*CompetitionDistance*、*CompetitionOpenSinceMonth*、*CompetitionOpenSinceYear*、*Promo2*、*Promo2SinceWeek*、*Promo2SinceYear* 和 *PromoInterval*；其中，*Open*、*StateHoliday*、*SchoolHoliday*、*StoreType*、*Assortment*、*Promo*、*Promo2* 为离散分类数据，*Sales*、*CompetitionDistance* 为连续数值数据，*DayOfWeek*、*Date*、*CompetitionOpenSinceMonth*、*CompetitionOpenSinceYear*、*Promo2SinceWeek*、*Promo2SinceYear*、*PromoInterval* 为时序数据。

检查 *data_train* 中数值数据的**统计量**：

	Sales	***CompetitionDistance***
平均值	5.773819e+03	5.430086e+03
标准差	3.849926e+03	7.715324e+03
最小值	0	2.000000e+01
最大值	4.155100e+04	7.586000e+04

从特征数据缺失值统计图（图 1）可发现若干特征数据中存在**缺失值**：*CompetitionDistance*, *CompetitionOpenSinceMonth*, *CompetitionOpenSinceYear*, *Promo2SinceWeek*, *Promo2SinceYear* 和 *PromoInterval*。其中 *CompetitionDistance* 为连续数值变量，可通过均值补全缺失数据。由于本项目将采用 XGBoost 构建模型，默认情况下 XGBoost 支持缺失值（树算法在训练期间学习缺失值的分叉方向），可令 XGBoost 自动处理其它分类变量的缺失数据 [3]。

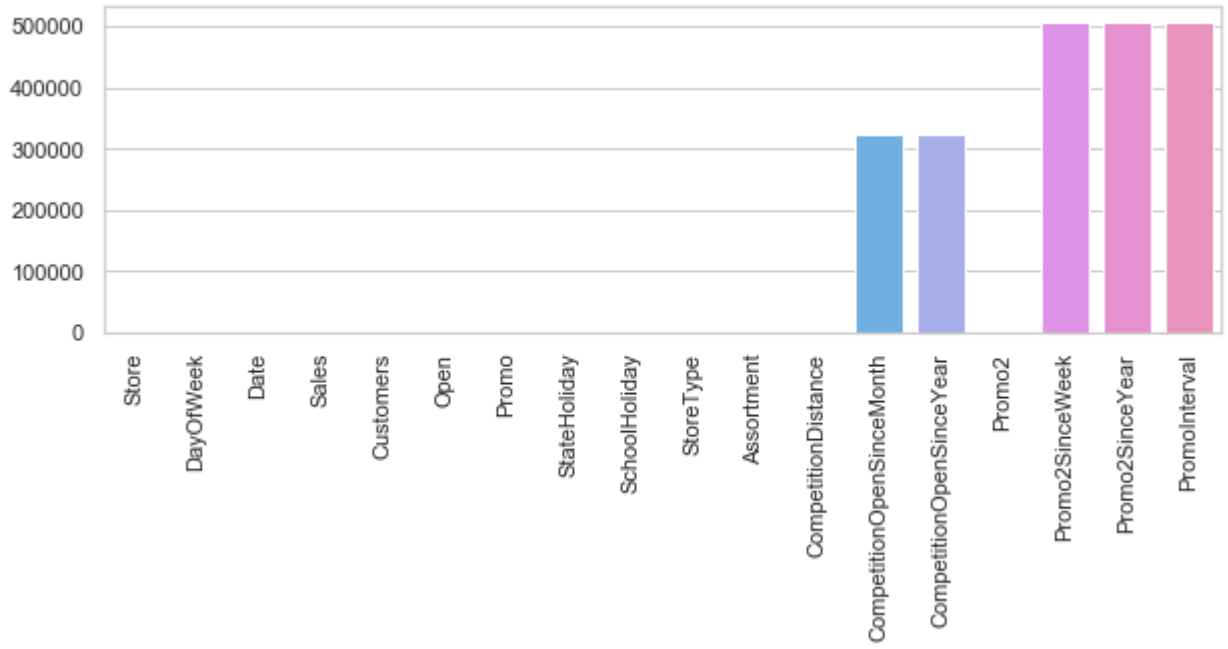

图 1：特征数据缺失值统计图。

从图 2 可发现 *Sales* 和 *CompetitionDistance* 两个特征数据中均存在**异常值**，呈长尾分布。因为这些异常值可能会影响最终模型的准确性，所以将在数据预处理过程中被删除。

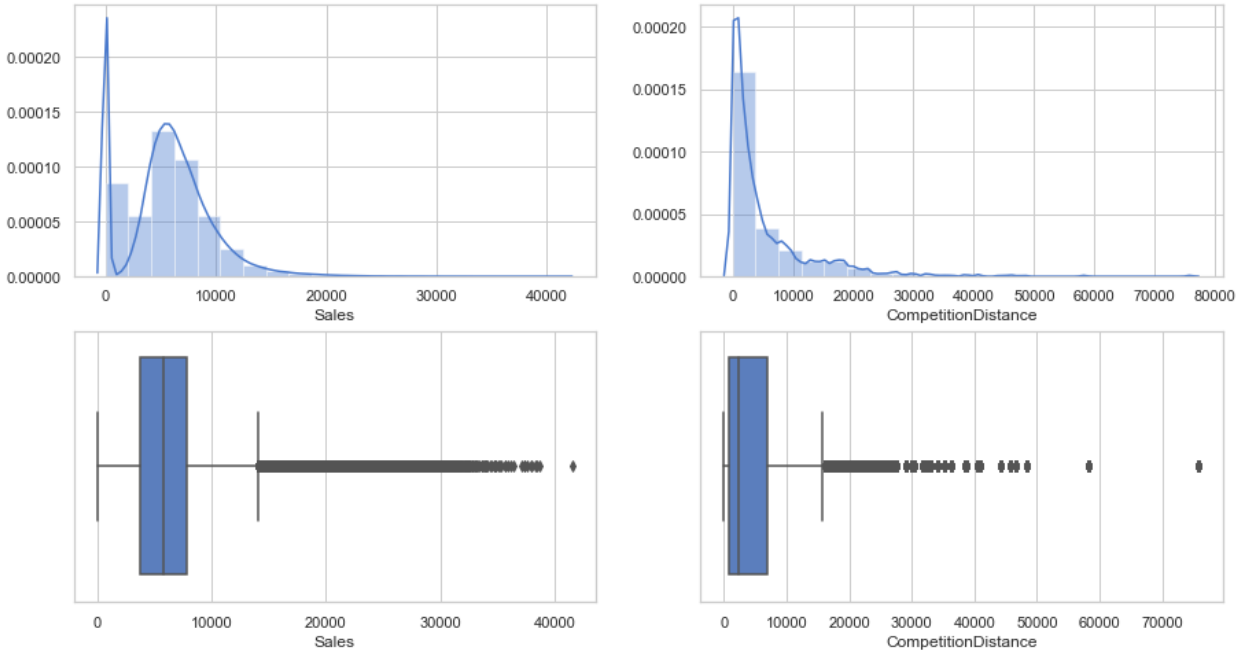

图 2：*Sales* 和 *CompetitionDistance* 的频率直方图和箱线图。

2.2 可视化数据

2.2.1 销售额与客户量

图 3 反映出从 2013 年 1 月至 2015 年 7 月每 7 天平均日销售额的变化情况。从图 3 可观察到销售额呈现波动性，其变化具有一定周期性，且在每年大约 12 月底出现高峰。

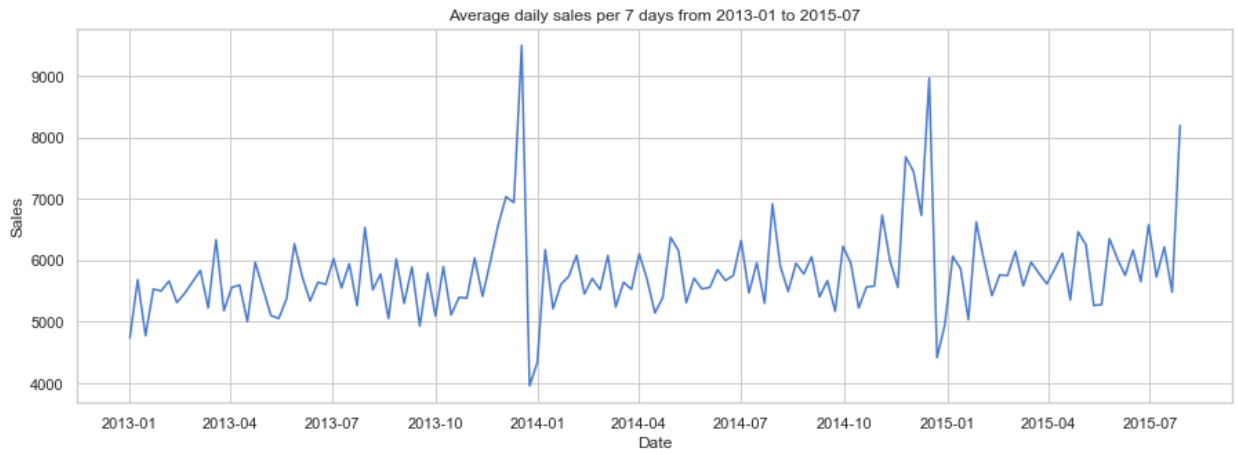

图 3

图 4 直观地反映出 7 天平均日客户量与 7 天平均日销售额具有相似的波动趋势，结合 *Sales* 与 *Customers* 的散点关系图（图 5），可发现两者具有相关性。计算 *Sales* 和 *Customers* 两列数据的皮尔逊相关系数得 0.823597，说明两者具有显著的相关性。

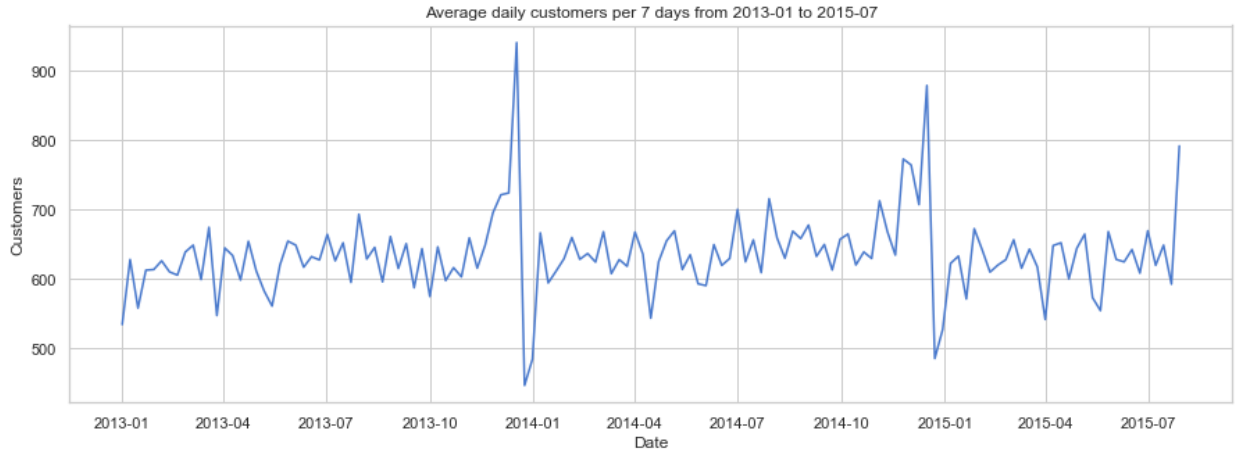

图 4

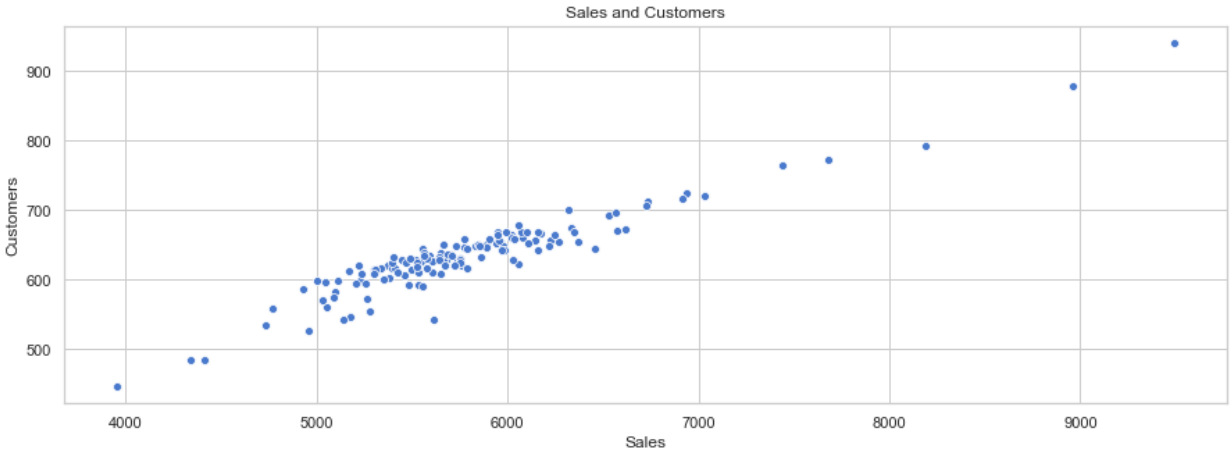

图 5

2.2.2 时间性与季节性因素

自 2013 年 1 月至 2015 年 7 月的星期数据（图 6）显示：一周之内，周一的平均日销售额保持最高水平，周日的平均日销售额保持最低水平，周二至周四持续平均日销售额走低，直到周五有一定回升，周六又有所下降。

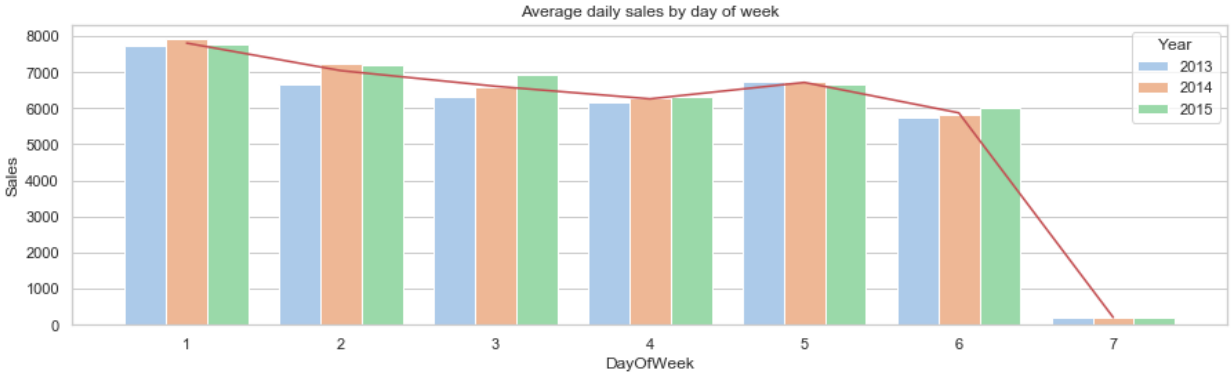

图 6

自 2013 年 1 月至 2015 年 7 月的月数据（图 7）显示：每年 7 月和 12 月分别有两次销售额高峰。另外，尽管缺少 2015 年 7 月以后的数据，但不难发现每年 10 月至 12 月期间规律性地增长趋势，其中 12 月平均日销售额达到全年最高，而 1 月回落到较低水平。

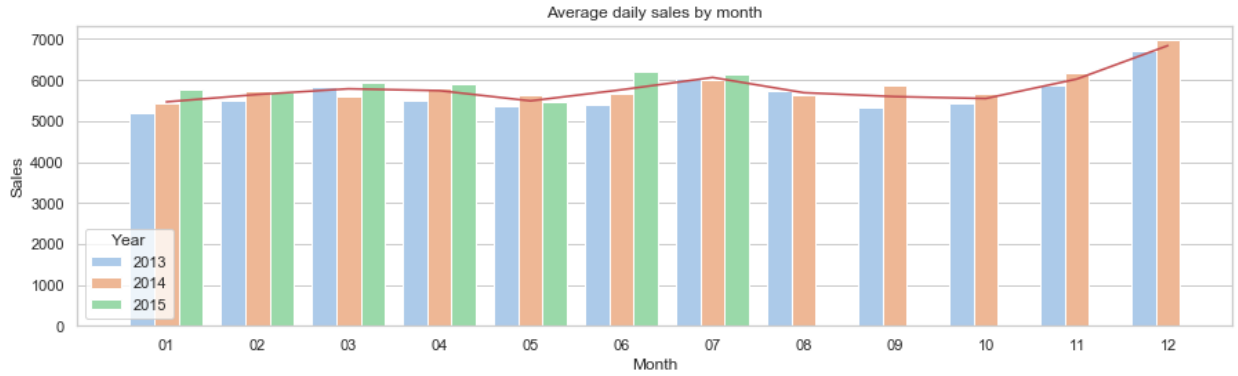

图 7

2.2.3 促销因素

每 30 天平均日销售额与促销情况关系图（图 8）显示：促销因素正向影响销售额。进行促销活动情况下平均日销售额明显增加，其中 12 月的增量尤为突出。*Promo_Promo2* 代表促销活动是否属于连续促销，连续促销的促销活动较非连续促销的促销活动，其平均日销售额更低。

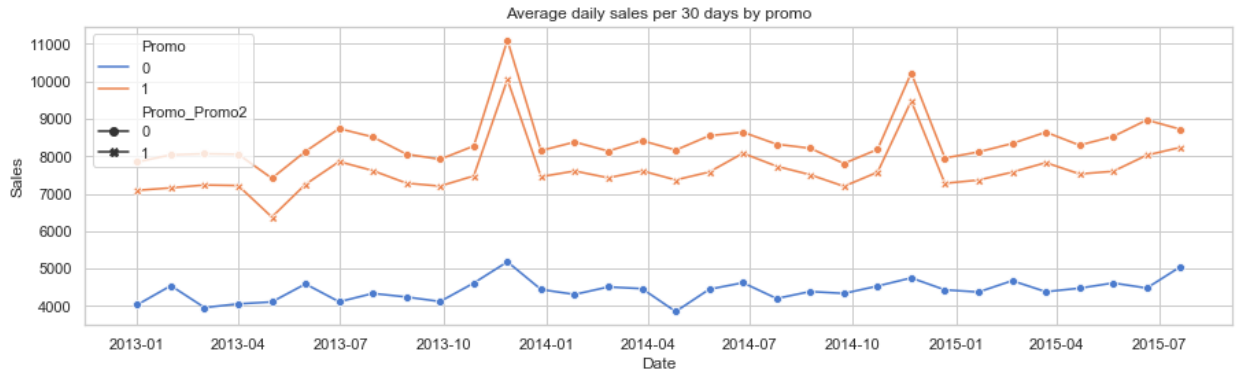

图 8

2.2.4 商店类型与零售商品品类级别因素

销售额与商品类型/零售商品品类级别关系图（图 9）显示：*b* 类商店在所有商品品类级别都具有较高的销售额，尤其在 *c* 类商品品类级别上。*a*、*c* 和 *d* 类商店没有 *b* 类商品品类级别，其销售额在另三种级别上基本保持相同水平。

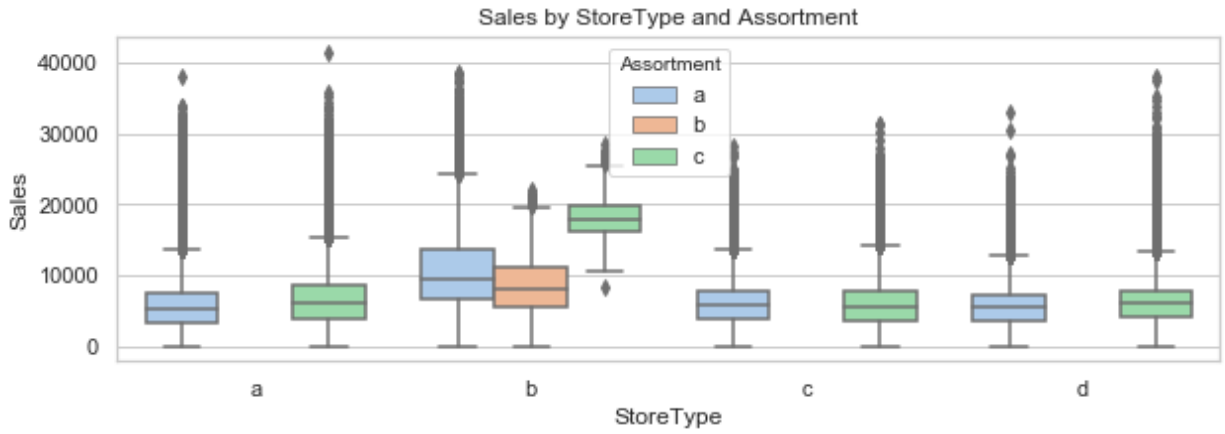

图 9

2.3 算法和技术

XGBoost 是一种为执行速度和模型性能而设计的梯度提升方法（gradient boosting）的实现，可以快速准确地解决许多数据科学问题。考虑到 XGBoost 适合处理大型结构化数据集的回归预测问题，故引入 XGBoost 作为模型构建工具。

XGBoost 选择**树集成**（decision tree ensemble）作为模型，即一组分类回归树（CART）。CART 树与普通决策树有所不同，其叶子节点只包含决策值或一个真实分数，能够提供比分类更为丰富的解释，利于统一的优化途径 [4]。通常，我们利用树集成综合（累加）单独树的预测结果（分数）以提升最终模型的预测能力。

XGBoost 使用**累加训练策略**（additive training）来进行树提升，即：分步构建树；对于每一步，固定已优化的树并添加一棵新树，使新添加的树最优化目标函数。

定义目标函数如下：

$$\begin{aligned}\text{obj}^{(t)} &= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \Omega(f_i) \\ &= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) + \text{constant}\end{aligned}$$

其中，$l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i))$ 为损失函数，$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$ 为正则化项以控制模型复杂度。将损失函数泰勒展开到二阶，得到一个更好形式的目标函数：

$$\sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

$$g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$
$$h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

其中，g_i 和 h_i 分别为损失函数的一阶偏导和二阶偏导。新的目标函数的优点在于其仅依赖 g_i 和 h_i 并支持自定义损失函数。

改进树定义为 $f_t(x) = w_{q(x)}, w \in R^T, q: R^d \to \{1, 2, \cdots, T\}$. 针对某一棵树，每个数据样本通过特定树结构 $q(x)$ 被分配到具体的叶子节点，全部 T 个叶子节点的分数向量 w 代表该树。进一步化简目标函数，得到：

$$\text{obj}^{(t)} = \sum_{j=1}^{T} [G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2] + \gamma T$$

其中，G_j 和 H_j 分别为序号 j 的叶子节点中所有数据样本 g 和 h 的累加值。对于给定的树结构 $q(x)$ 得最优分数向量 w_j^*，代回 w_j^* 到目标函数得到衡量树结构好坏的**结构分数** obj^*（越小即越好）。

$$w_j^* = -\frac{G_j}{H_j + \lambda}$$
$$\text{obj}^* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T$$

对于每次累加训练过程，枚举所有可能的分割方案，通过计算左叶子节点分数、右叶子节点分数及父节点分数，计算信息增益：

$$Gain = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma$$

如果 $Gain > 0$ 则分割节点，否则不分割节点。最终，我们确定了新添加树的结构。

综上，总结 XGBoost 算法步骤如下：

1. 初始化 $f_0(x)$
2. 迭代 T 次：
    a. 计算损失函数在每个训练样本的一阶偏导 $\partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ 和二阶偏导 $\partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$
    b. 递归地运用树分裂算法生成一颗决策树 $f_t(x)$ 使得增益 $Gain$ 最大
    c. 添加新的决策树至模型 $\hat{y_i}^{(t)} = \hat{y_i}^{(t-1)} + f_t(x)$

2.4 基准模型

模型的 Kaggle 竞赛排名到达排行榜前 10%，模型对测试集预测结果的 RMSPE（Root Mean Square Percentage Error）评估得分至少达到 0.11773。RMSPE 计算方法如下：

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

其中，y_i 表示单个商店单日实际销售额，$\hat{y}_i$ 表示对应预测值。

III. 方法

3.1 数据预处理

3.1.1 处理缺失值

因为默认情况下 XGBoost 支持缺失值（树算法在训练期间学习缺失值的分叉方向），所以可不进行额外处理而令 XGBoost 自动处理分类变量的缺失数据 [3]。

3.1.2 处理异常值

XGBoost 算法对异常值不敏感，具有较强的健壮性。权衡数据量与异常值可能带来的干扰，调整 *Sales* 或 *CompetitionDistance* 的异常值阈值为 4.2 个标准差。被删除的异常样本量占总样本量 0.011 是可接受的，预计不会因数据量不足而导致欠拟合。

3.1.3 数据转换

XGBoost 只能处理数值数据，因此所有分类数据需要被转换为相应的数值数据。针对某些时序数据，如 *DayOfWeek*、*CompetitionOpenSinceMonth*、*CompetitionOpenSinceYear*、*Promo2SinceWeek*、*Promo2SinceYear*，我们不做离散化处理（如 one-hot 编码），因为我们需要保证这些时序变量的先后性信息。

将年-月-日格式的 *Date* 字段分割成年、月、日三个独立整型特征：*Year*、*Month* 和 *Day*。该三个特征均为时序变量，具有先后性，因此不需要被离散化。

StateHoliday、*SchoolHoliday*、*StoreType*、*Assortment*、*PromoInterval* 都是分类变量，每个变量都包含几种相互独立无关联的类型值。例如：*StoreType* 值范围是 *a*、*b*、*c*、*d*。我们需对这些变量进行 one-hot 编码，转换为二进制向量。

3.2 执行过程

执行过程所需的 Python 库主要包括：

- scikit-learn
- xgboost
- pandas
- numpy

首先，利用 `sklearn.model_selection.train_test_split` 函数分割训练集为训练集和验证集，设定验证集占比为 0.01。然后，分别创建训练集和验证集的 `xgboost.DMatrix` 数据矩阵；调用 `xgboost.train` 函数，设置输入数据矩阵和参数值，进行模型训练，观察本地验证分数。最后，使用最终模型生成测试集的预测结果并将其提交至 Kaggle，查看评估分数（**Private Score** 和 **Public Score**）。

参数设置如下：

```
num_boost_round=1000,
early_stopping_rounds=20,
params = {
    'objective': 'reg:linear',
    'booster': 'gbtree',
    'eta': 0.3,
    'max_depth': 9,
    'min_child_weight': 1,
    'silent': 1
}
```

设置提升迭代次数 `num_boost_round` 为 1000，早停参数 `early_stopping_rounds` 为 20，以控制模型过拟合的状况。

由于数据集标签（*Sales*）呈右偏态分布（见图 2），所以在训练之前数据集标签需借助 `np.log1p(y)` 进行对数变形。当进行预测时，需再借助 `np.expm1(y_pred)` 将模型输出结果还原得到销售额预测最终结果。

本项目原期望应用网格搜索（GridSearchCV）交叉验证获取最佳模型，但鉴于该运算过程非常耗时，决定参考相关 XGBoost 调参指南，尝试手动调参 [6]。

3.3 完善

初始验证结果：

eval-rmse	train-rmse	eval-rmspe	train-rmspe
1.08704	1.08207	0.165306	0.146012

表 1：初始模型本地验证分数

该分数尚未达到基准模型要求。先尝试仅调整 `max_depth` 和 `min_child_weight` 参数，发现模型性能并未得到显著提升。考虑重新进行特征工程，优化某些特征。

输出并分析特征重要性（图 10）。在特征工程层面作出如下调整：基于 *Date* 数据添加 *DayOfYear* 和 *WeekOfYear* 特征；删除重要性较低的 *StateHoliday* 特征；经过 one-hot 编码后的 *PromoInterval* 特征对模型贡献不大，考虑将其转换为新单一特征 *InPromoMonth*（表示是否处于促销月）以降低过拟合；*SchoolHoliday* 特征已二元化，无必要再次编码，考虑合并冗余特征 *SchoolHoliday_0* 和 *SchoolHoliday_1*；合并 *CompetitionOpenSinceYear* 和 *CompetitionOpenSinceMonth* 至新单一特征 *CompetitionOpen*（表示竞争者开店多久）；合并 *Promo2SinceYear* 和 *Promo2SinceWeek* 至新单一特征 *PromoOpen*（表示连续促销多久）。

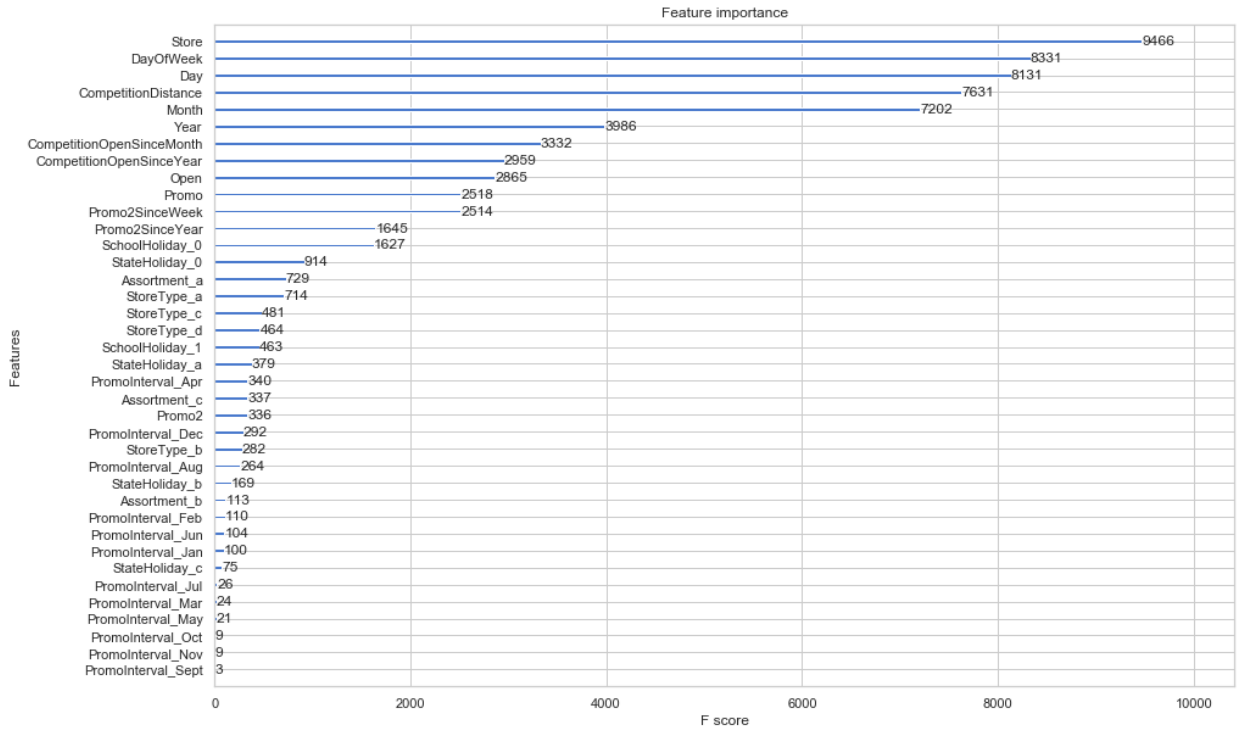

图 10：初始模型特征重要性

数据样本具有时序性，为在训练过程中保持样本间时序关系信息，考虑在进行数据集分割时设置 `sklearn.model_selection.train_test_split` 的 `shuffle` 参数为 `False`，从而禁用其随机打乱样本顺序的操作。

另外，在参数层面作出如下调整：设置树最大深度 `max_depth` 为 10，适当加强拟合程度；降低学习率 `eta` 至 0.03，以提高模型的鲁棒性；设置单棵树随机采样比例 `subsample` 为 0.9，每棵

树随机采样列数比例 `colsample_bytree` 为 0.7，以防止过拟合；设置提升迭代次数 `num_boost_round` 为 4500，以令模型更细致地学习。调整后参数设置如下：

```
num_boost_round=4500,
early_stopping_rounds=100,
params = {
    'eta': 0.03,
    'max_depth': 10,
    'min_child_weight': 2,
    'subsample': 0.9,
    'colsample_bytree': 0.7,
    'silent': 1,
    'seed': 10
}
```

最终验证结果：

eval-rmse	train-rmse	eval-rmspe	train-rmspe
0.100495	0.061135	0.011709	0.007186

表 2：最终模型本地验证分数

IV. 结果

4.1 模型的验证与评估

经过多次优化过程，包括特征工程和调参，获得的最终模型表现最为出色。最终模型通过本地验证和 Kaggle 在线评估：经过 XGBoost 多轮迭代训练，实时验证评分显示模型已基本收敛并达到最优（表 2）；使用最终模型生成测试集的预测结果并将其提交至 Kaggle 进行在线评估，评估分数（表 4）达到预期目标。根据预测结果所示，最终模型的预测偏差较小，具有较强的鲁棒性。

Private Score	Public Score
0.17079	0.16674

表 3：初始模型在线评估分数

Private Score	Public Score
0.12466	0.11608

表 4：最终模型在线评估分数

4.2 合理性分析

最终模型的性能评估是在测试数据上实施的，Kaggle 在线评估机制利用测试集非公开的实际销售额对预测结果进行评估计算。最终模型的在线评估分数 0.11608 略优于 2.4 部分所提出的基准模型标准 0.11773。因此，我们解决了在时序数据上构建稳定销售额预测模型的问题。

V. 项目结论

5.1 结果可视化

根据测试集数据的时间跨度，预测结果相应的时间范围是 从 2015-08-01 到 2015-09-17。对比 2014 年和 2015 年同时间段内平均日销售额波动趋势（图 10，图 11），发现存在明显的基于时间的变化规律。再观察最终模型特征重要性（图 12），可发现 *DayOfYear*、*Day*、*DayOfWeek* 等时序特征重要性更高，这印证了上述情况。

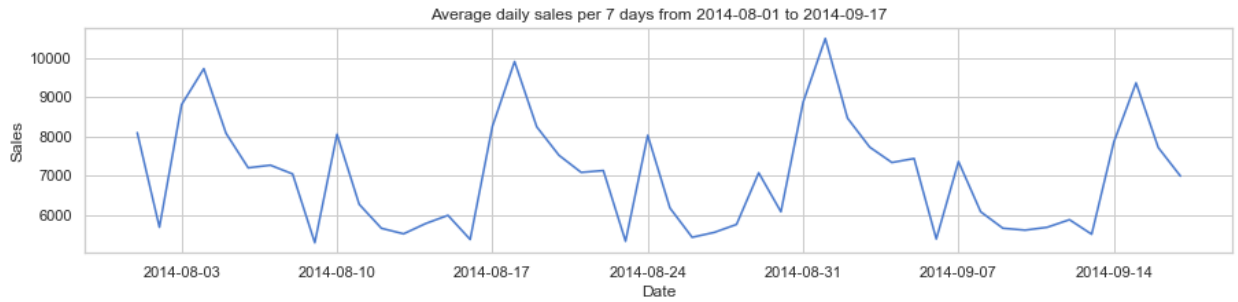

图 10：2014-08-03 至 2014-09-14 平均日销售额波动趋势（实际）

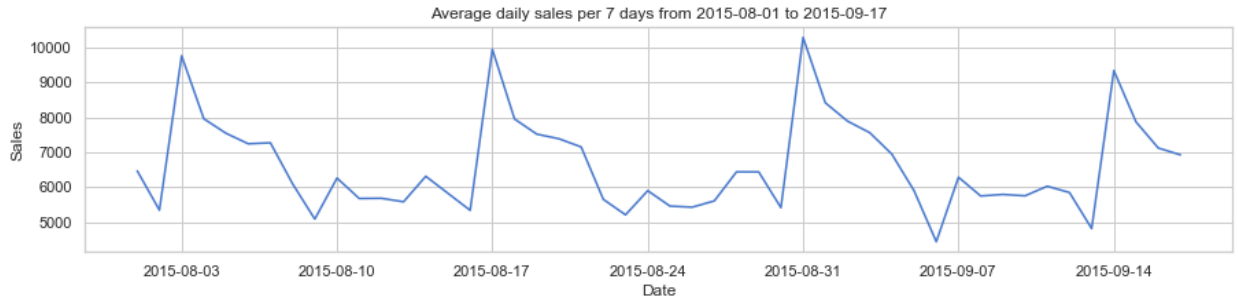

图 11：2015-08-03 至 2015-09-14 平均日销售额波动趋势（预测）

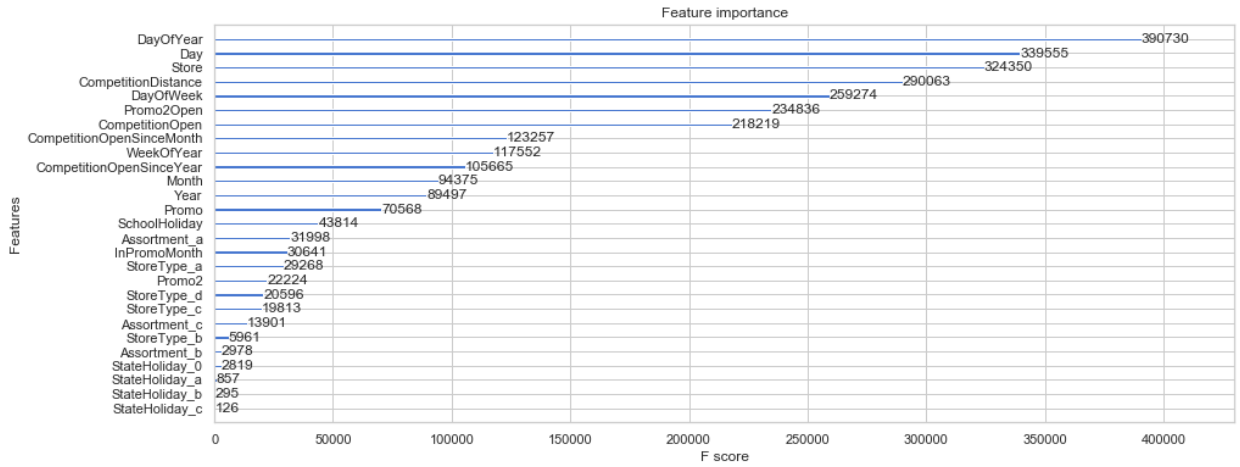

图 12：最终模型特征重要性

5.2 对项目的思考

本项目遵循被推荐的监督式学习研究步骤 [7]。所有问题随工作流程分步解决，最终构建出符合预期的稳健的销售额预测模型。期间，根据模型评估结果多次迭代特征工程和调参以优化模型性能。

项目流程如下：

1. 探索数据
   通过数据分析方法探索数据集，包括如下方面：
   a. 导入数据
      利用 Pandas 库导入数据集文件至数据帧（DataFrame），联合多个数据源以令其合理化到同一数据集中。
   b. 检查数据
      检查统计量、缺失值和异常值。
   c. 可视化数据
      利用 Seaborn 和 Matplotlib 库可视化数据并观察数据特点和趋势。
2. 预处理数据
   清洗数据，处理缺失值和异常值，转换某些特征数据。
3. 训练和验证模型
   利用 XGBoost 库训练模型。调整超参，选择最优化模型。输出并分析特征重要性，进行特征选择。因数据集具有时序性，需通过时序性（而非随机）方式分割数据集。该阶段应用 XGBoost 与 scikit-learn 机器学习库。
4. 测试和评估模型
   使用最终模型生成测试集的预测结果并将其提交至 Kaggle，查看分数。分析结果，进一步调整优化模型，提高性能。

在上述过程中，特征工程是需反复考量的复杂任务，例如，如何将 *Promo2SinceYear* 和 *Promo2SinceWeek* 信息合并转换为时间距离数值型特征；另外调参也具有一定困难，例如，如何使参数保证偏差-方差平衡。

5.3 需要作出的改进

可考虑今后进行更为细致的特征筛选或尝试构造新的特征，从而优化模型的性能。由于本地运算能力的限制，本项目未能进行网格搜索交叉验证而使得调参效率较低，可考虑今后部署分布式并行计算系统来优化这一过程。

引用

1. *Rossmann Store Sales*, www.kaggle.com/c/rossmann-store-sales.
2. *What Is Machine Learning?*, https://www.mathworks.com/discovery/machine-learning.html.
3. *How to deal with Missing Value*, https://xgboost.readthedocs.io/en/latest/faq.html#how-to-deal-with-missing-value.
4. *Introduction to Boosted Trees*, https://xgboost.readthedocs.io/en/latest/tutorials/model.html.
5. Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *KDD*.
6. *Complete Guide to Parameter Tuning in XGBoost (with codes in Python)*, https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/.
7. *Machine Learning Workflow*, https://cloud.google.com/ml-engine/docs/tensorflow/ml-solutions-overview