1 问题的定义

1.1 项目概述

该项目源自2015年Kaggle的比赛Rossmann Store Sales。Rossmann是欧洲的一家大型连锁药店，Rossmann的经理希望能对各个门店未来六周的销售额进行预测，因为可靠的销售额预测能让经理提前进行合理的资源分配，从而提高资源的使用效率。门店的销售额会受很多因素的影响，该项目的目标就是要基于Rossmann各个门店的信息（比如促销、竞争、节假日、季节性和位置等）以及过往的销售情况来建模预测未来六周的销售额。项目使用到的数据集为1115个Rossmann门店的历史销售记录和这些门店的相关信息。

1.2 问题陈述

项目要求预测的是门店未来六周的销售额，所以按照机器学习对问题的分类方法，该项目属于回归问题，那么项目要完成的任务就是从所给的数据中提取出可能对销售额有影响的特征，建立有效的回归模型进行预测。具体来说，主要可以分为以下几步：

- 通过探索性数据分析（Exploratory Data Analysis）来尝试了解数据的一些基本情况，像数据的分布情况、缺失情况等，为后面的特征工程（Feature Engineering）做准备和提供一些参考；
- 通过数据预处理（Preprocess the data）过程来处理诸如类别信息、缺省值、时间序列信息等，便于后续的特征提取；
- 通过特征工程（Feature Engineering）来最大限度地提取出特征供后续的模型使用；
- 利用提取到的特征建立回归模型，在建模的过程中可以尝试通过特征选择和模型融合（feature selection and model ensemble）来争取达到比较好的预测效果。

最终期望达成的结果是，训练好的模型根据门店的相关信息（比如促销，竞争对手，位置等）和预测日当天以及前后一段时期的节假日等信息，能相对准确地对预测日的销售额进行预测。

1.3 评价指标

评价指标选用的是百分比均方根误差（the Root Mean Square Percentage Error(RMSPE)），计算方式如下：

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

其中y_i表示单个门店单天的销售额，$\hat{y}_i$表示对应门店对应单天的销售额预测值，对于单个门店单天的销售额为0的情况不予评价。采用百分比误差可以有效降低大尺度的数据对最终误差的影响，对于门店的销售额数据来说，某些节假日或者某些特殊日子的销售额肯定比平常日要高出不少，如果采用均方根误差，那些很大的销售额数据就会对误差评估产生较大的影响，从而可能对模型好坏的评估产生误判。

2 分析

2.1 数据的探索

项目提供的数据集中包含三个部分：train、test和store，train数据集中包含了各个门店的历史销售额数据，store数据集中包含了各个门店的一些补充信息，test数据集被用于测试和评估模型。下面对各个数据集做一个详细介绍。

2.1.1 数据域的介绍

train数据集中一共包含1017209条数据记录，包含的数据域如下：

- Store: 门店的数字标识
- DayOfWeek: 周一到周日的数字标识
- Date: 日期，从2013-01-01到2015-07-31
- Sales: 单个门店在某个日期的销售额数据
- Customers: 单个门店在某个日期的顾客数
- Open: 指示门店是否开放（0表示关闭，1表示开放）
- Promo: 表示门店是否在进行促销（0表示否，1表示是）
- StateHoliday: 表示对应日期当天是否是法定假日（a表示是公共假日，b表示复活节，c表示圣诞节，0表示不是法定节日）
- SchoolHoliday: 表示对应日期当天是否是学校假日（0表示不是，1表示是）

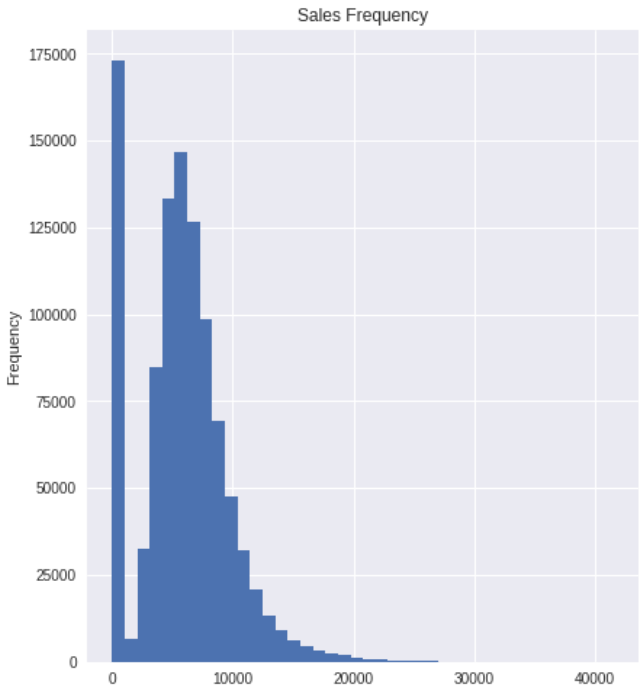
store数据集中一共包含1115条数据记录，包含的数据域如下：

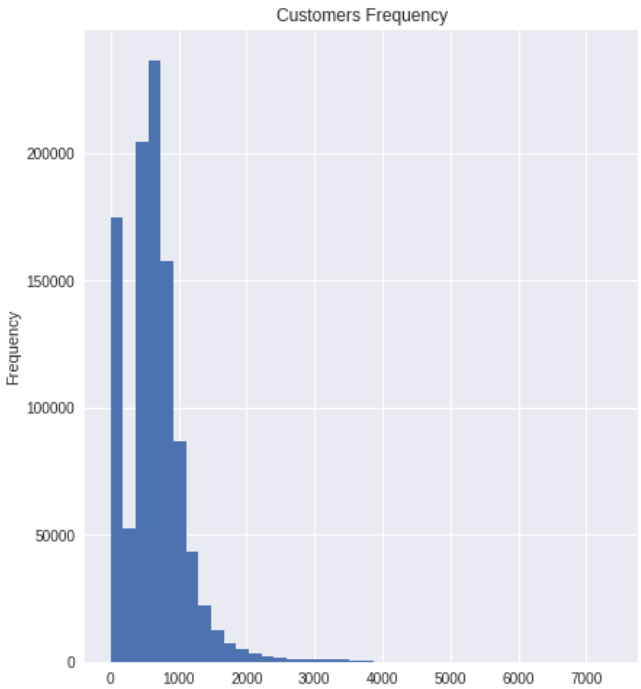
- Store: 门店的数字标识
- StoreType: 一共四种不同类型的门店（a,b,c,d）
- Assortment: 描述门店上架的商品类型（a表示基本，b表示额外，c表示扩展）
- CompetitionDistance: 表示与最近的竞争对手门店的距离
- CompetitionOpenSinceMonth: 表示最近的竞争对手门店开业的月份
- CompetitionOpenSinceYear: 表示最近的竞争对手门店开业的年份
- Promo2: 表示一个门店是否进行了持续的促销（0表示否，1表示是）
- Promo2SinceWeek: 表示一个门店进行持续促销的开始的周数
- Promo2SinceYear: 表示一个门店进行持续促销的开始的年份
- PromoInterval: 表示门店开始促销的月份

test数据集包含41088条数据记录，数据域和train数据集一样，只是没给销售额数据和顾客数，额外的Id用于Kaggle上数据的提交。

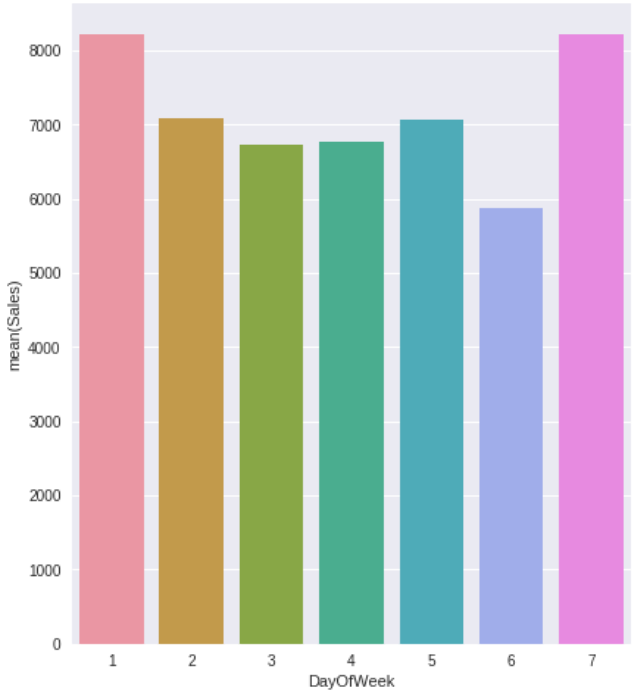
2.2 探索性可视化

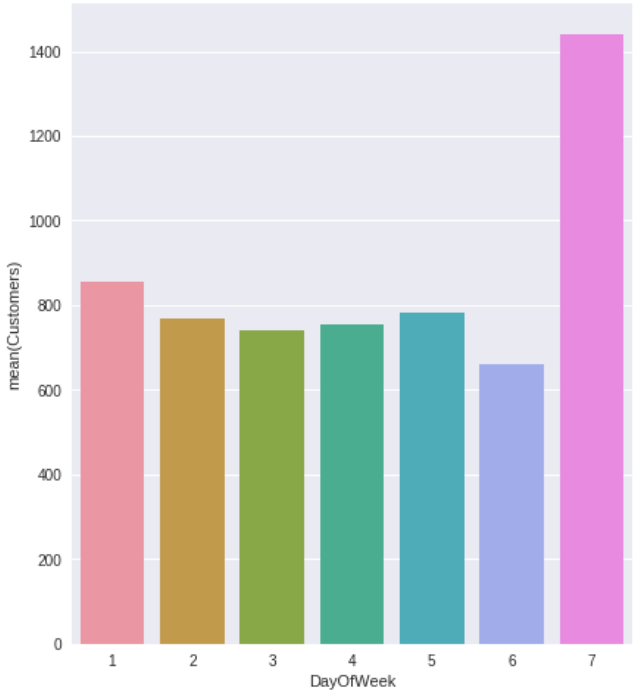
下面的两张图分别是销售额数据和顾客数据的频率分布图，可以看出销售额大部分在4000~8000之间，顾客数大部分在500~1000之间。此外，有一定数量的记录显示销售额和顾客数为0。



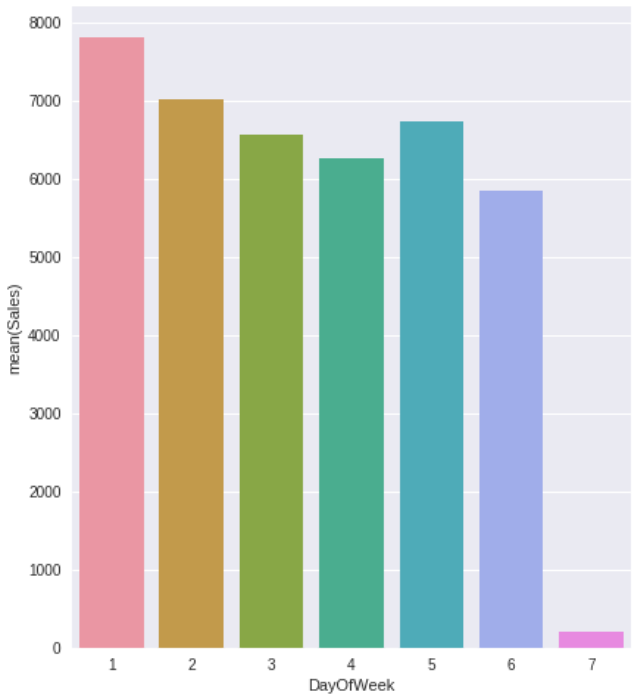


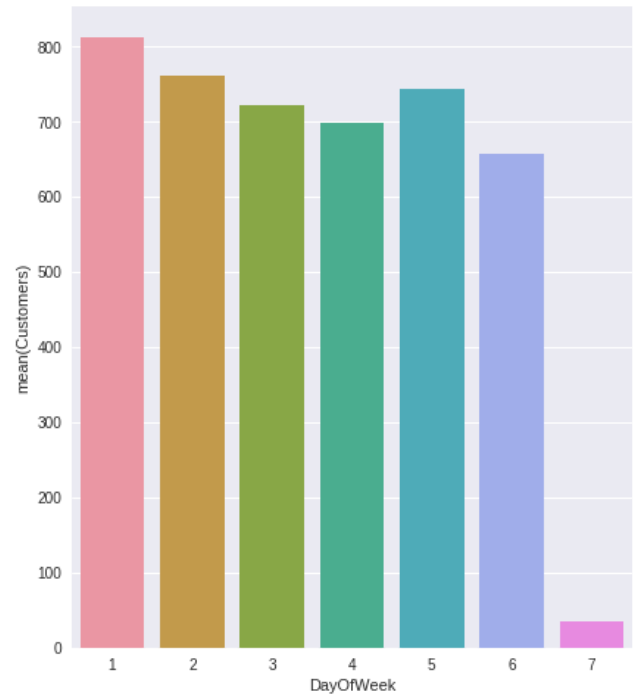
下面的两张图分别是所有开放的门店周一到周日的平均销售额和平均顾客数。可以看出周一、周五和周日的平均销售额和平均顾客数都相对较高，其中周日的平均销售额和平均顾客数最大，可能是因为周日大部分店是关门的，只有少数店是开门的，所以顾客相对比较集中。



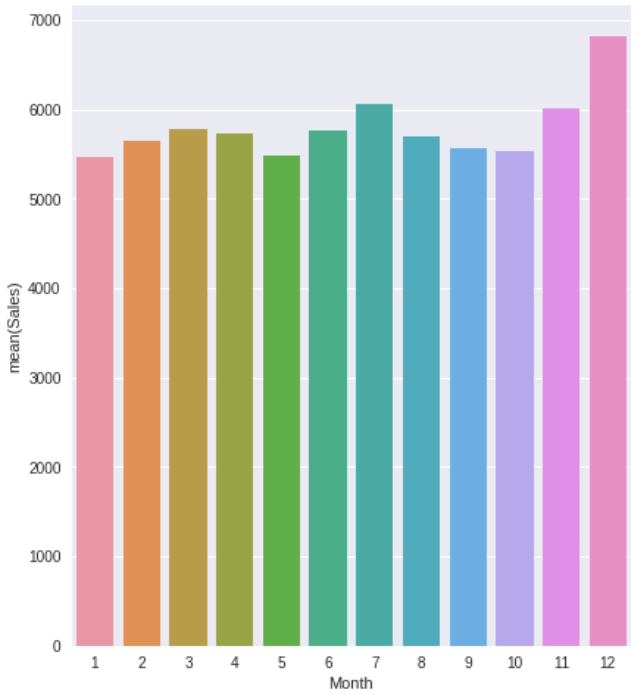


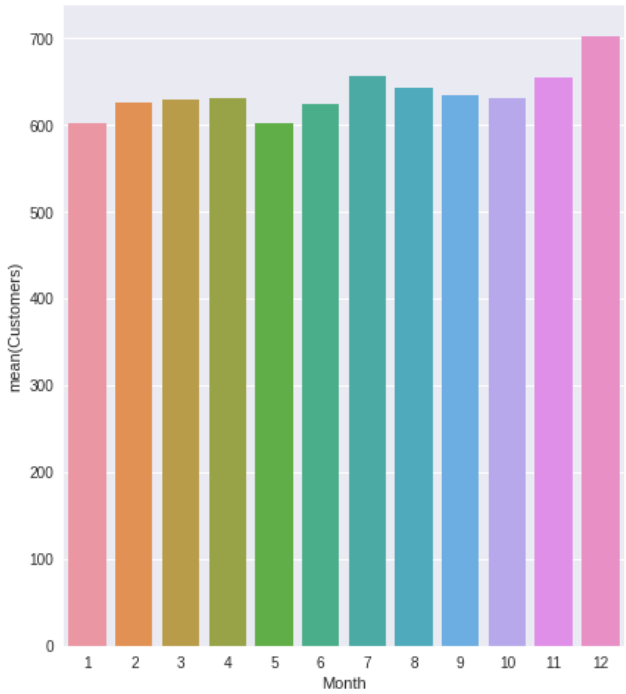
下面的两张图是所有门店（包括开放的和关闭的）周一到周日的平均销售额和平均顾客数，对比上面的两张图，可以分析出周日大部分门店是关闭的，关闭的门店销售额会是0。



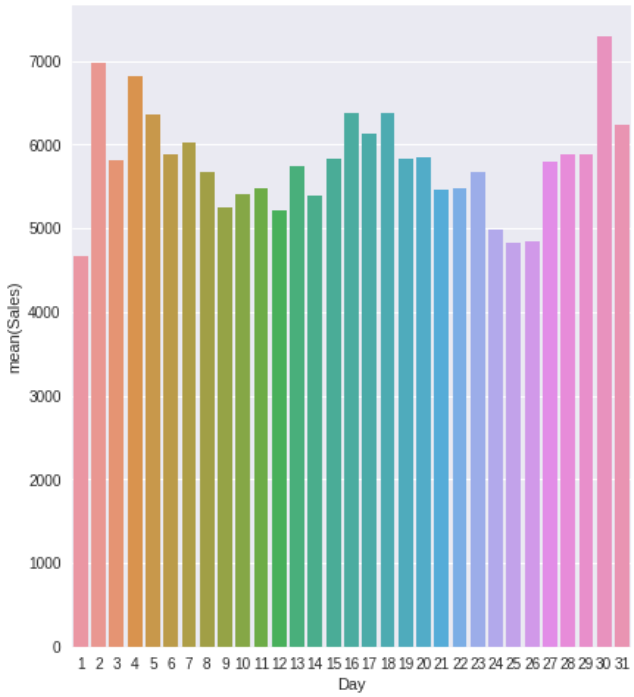


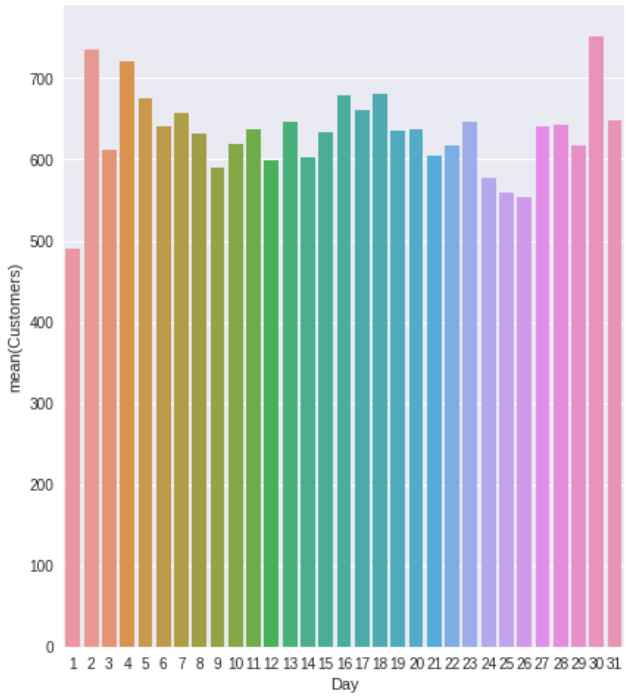
下面的两张图是所有门店月度平均销售额和平均顾客数，可以看出在6,7月份和11,12月份，平均销售额和顾客数有明显的增加，可能是由于暑假和圣诞节等假期的影响。



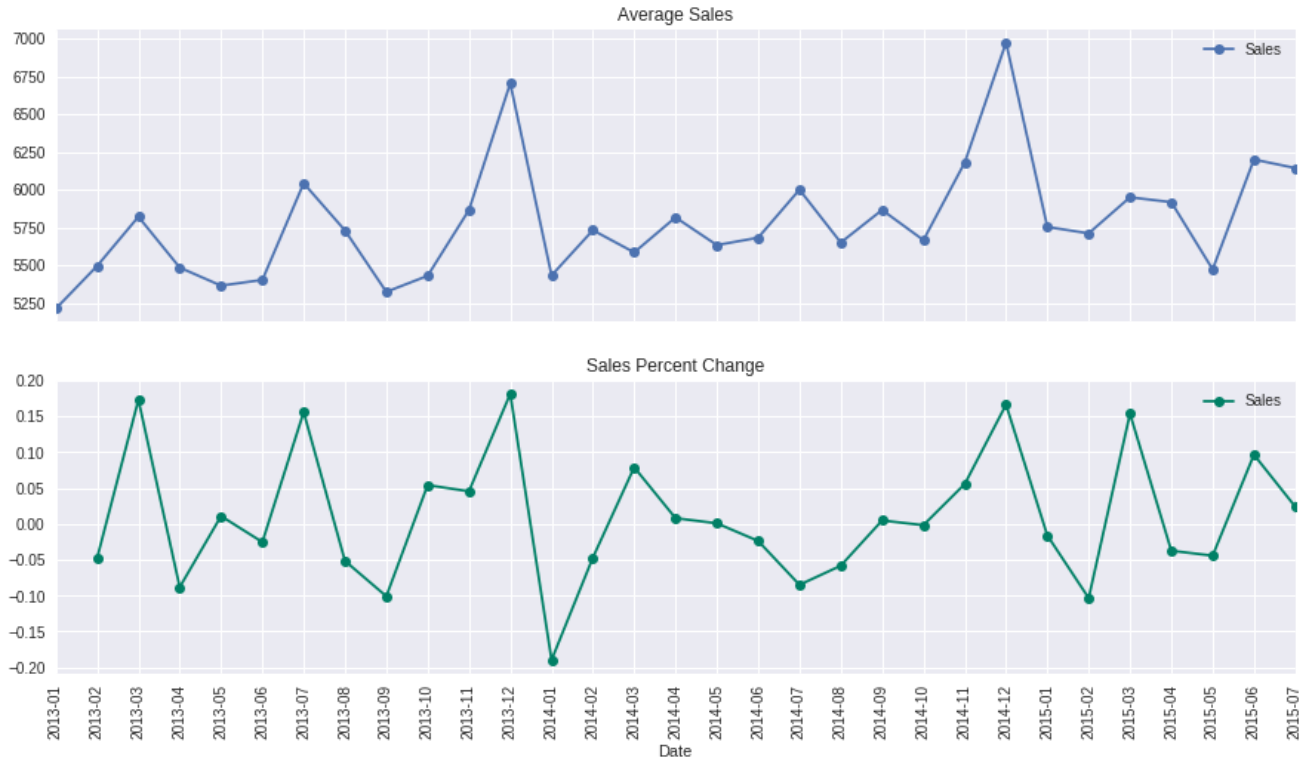


下面的两张图是所有门店按照每月的天数计算的平均销售额和平均顾客数，可以看出月初、月中和月末的平均销售额和平均顾客数要高于平时。

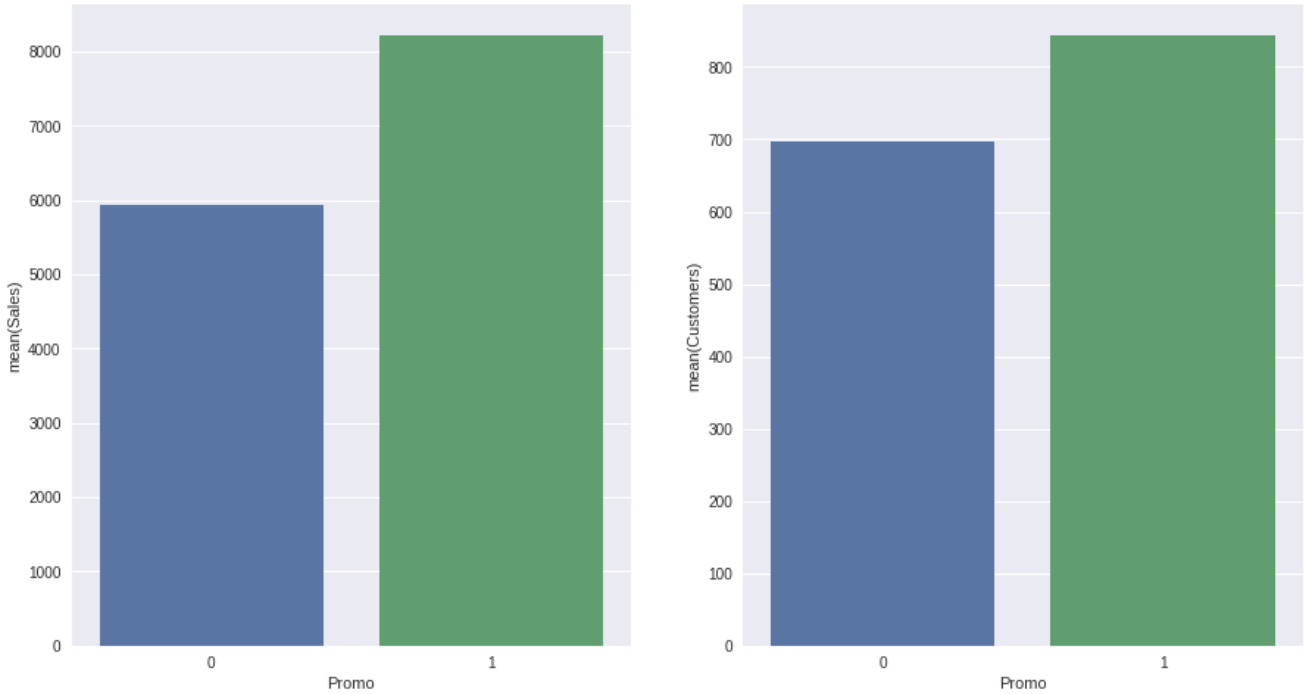




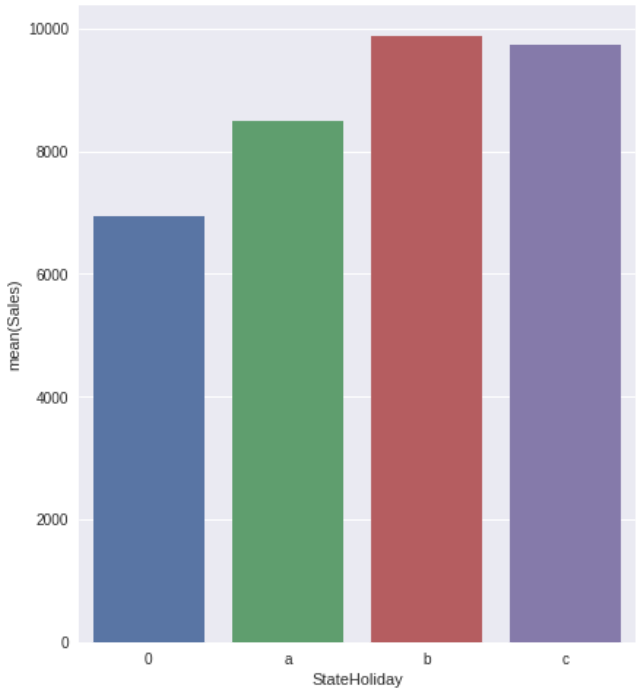
下面的两张图分别展示了2013年1月到2015年7月的月平均销售额的走势和变化率的曲线，可以看出走势和上面分析的结果比较吻合。

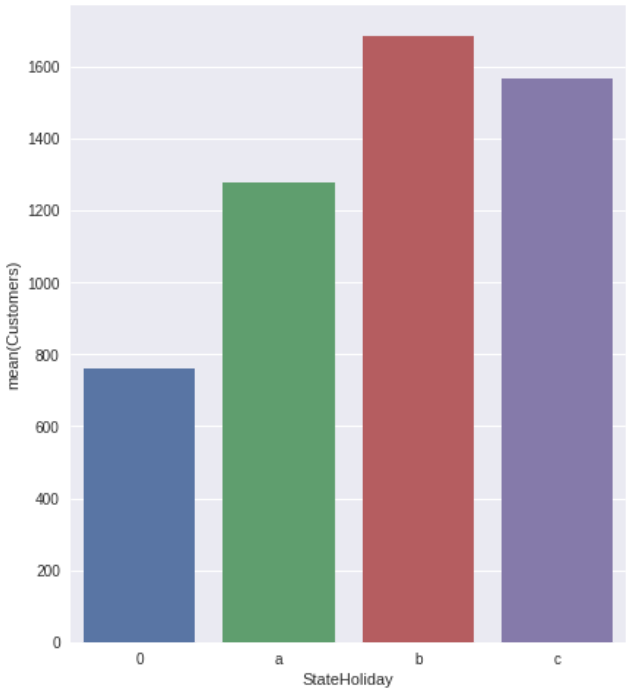


下面的两张图分别展示了所有开放门店有无促销情况下的平均销售额和顾客数，可以看出，有促销（Promo=1）情况下的平均销售额和顾客数均比无促销（Promo=0）情况下的要高，说明促销对销售额的提升有一定作用。

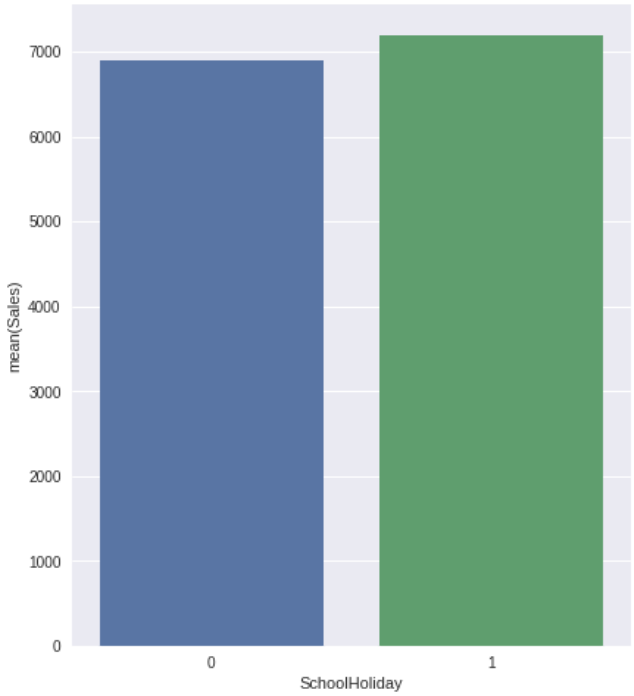


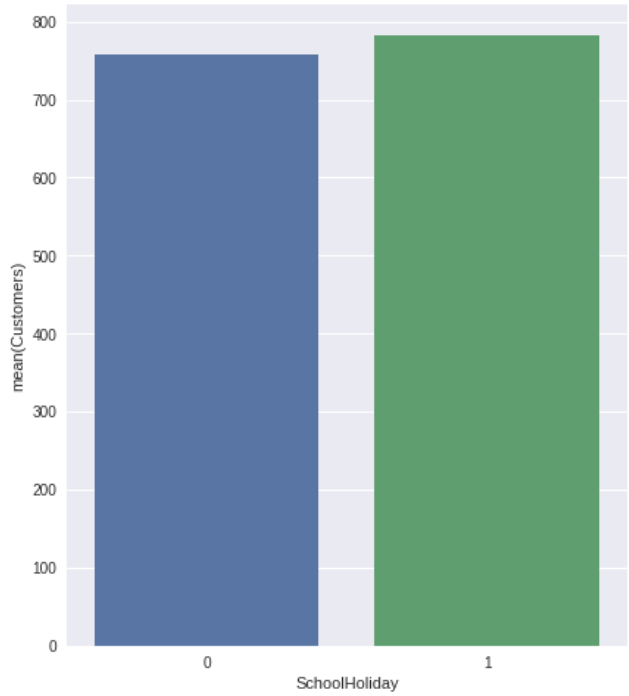
下图展示了所有开放门店不同法定假日情况下的平均销售额和顾客数。



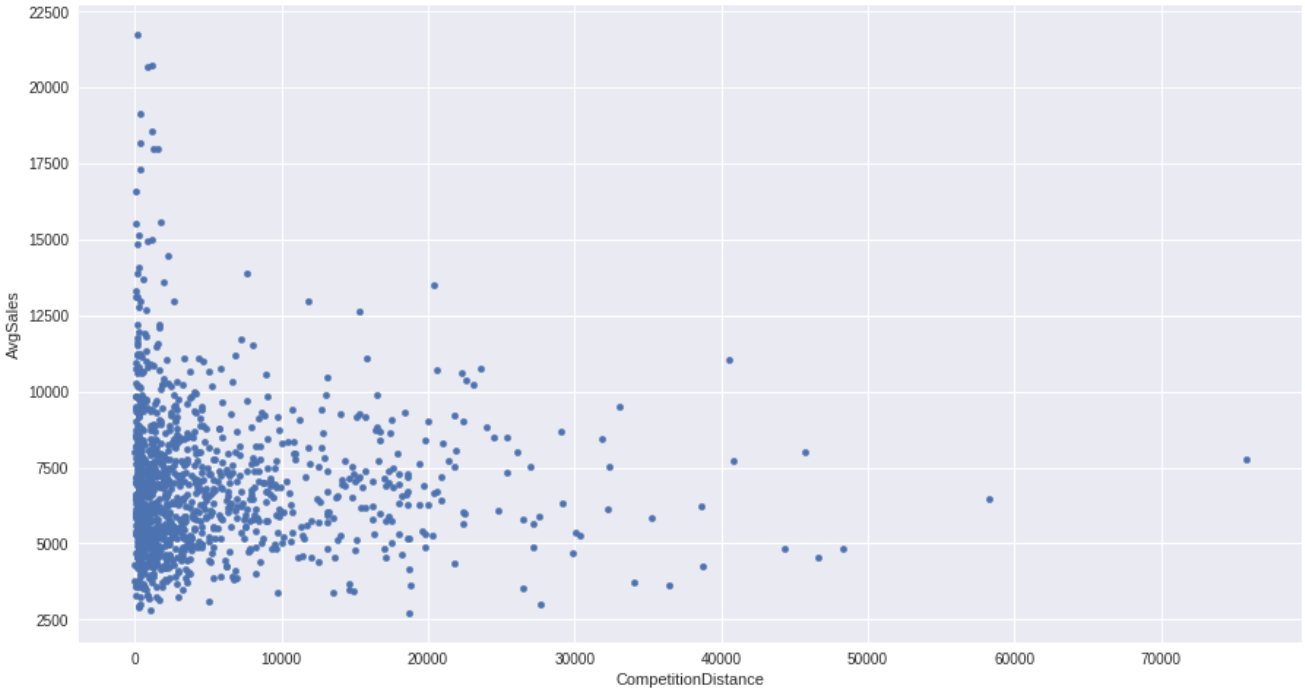


下图展示了所有开放门店是否是学校假日情况下的平均销售额和顾客数。

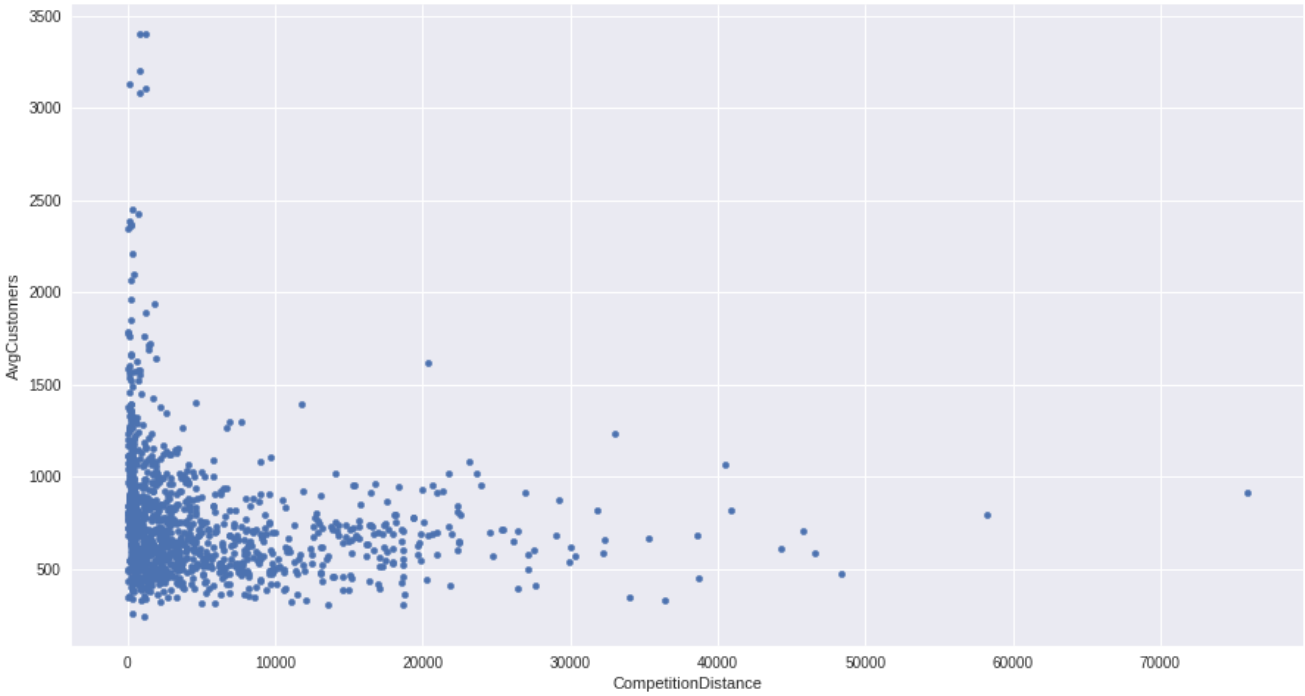




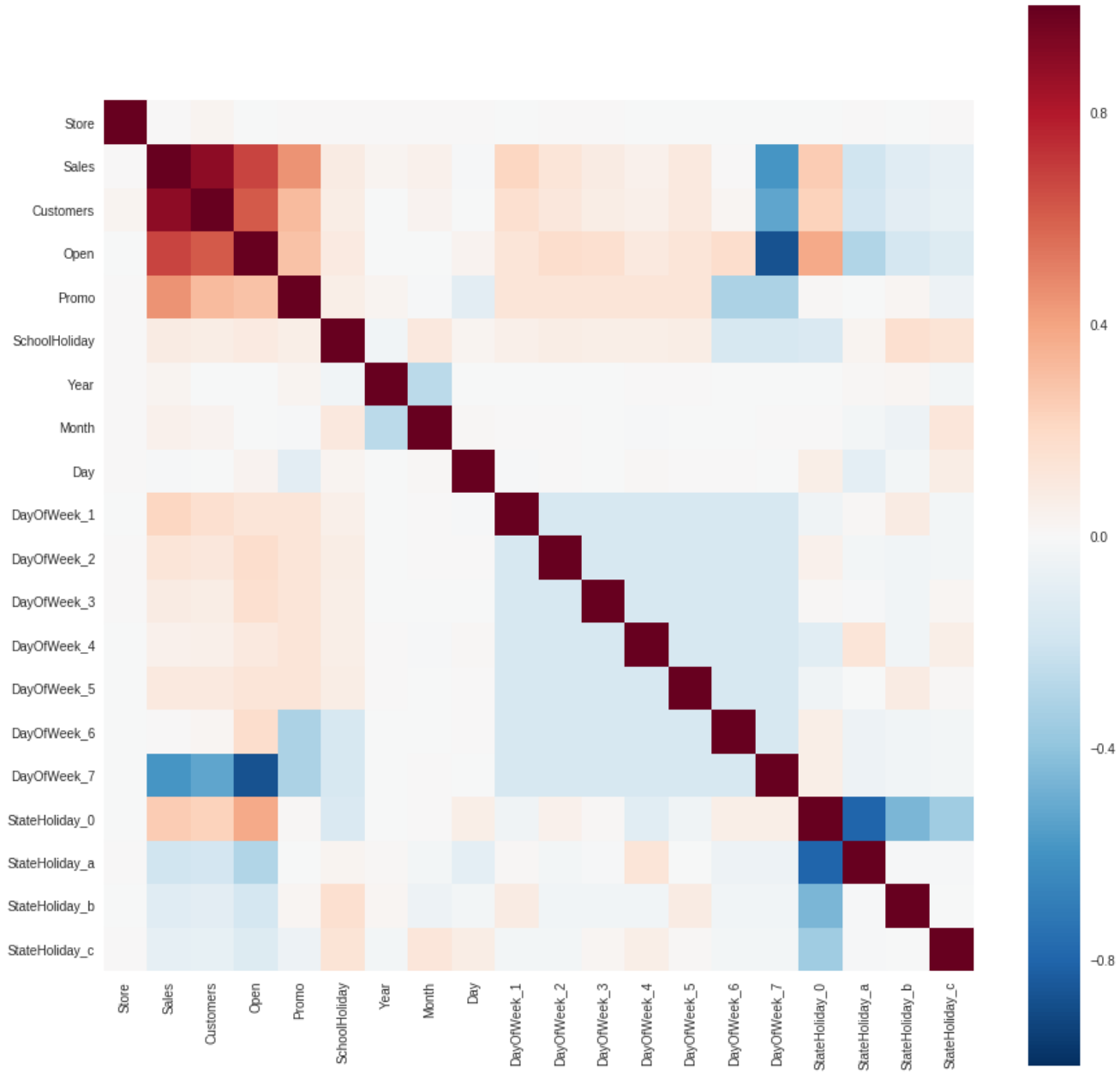
下图是门店的平均销售额与竞争对手与其距离的散点图，可以大致看出竞争对手与其的距离越近，该门店的平均销售额就相对越小。



下图是门店的平均顾客数与竞争对手与其距离的散点图，可以看出和平均销售额有类似的趋势。



下图是一些特征的相关性矩阵图，可以看出，促销（Promo）、是周几（Day of Week）和假日信息等与销售额之间有较强的相关性。



2.3 算法和技术

由前面的分析可知，这是一个回归问题。所以首先想到的是采用简单的线性回归模型进行拟合，观察拟合效果。鉴于给出的训练数据集中包含很多不同类型的特征数据，包括数值型、类别型等，于是尝试采用集成学习的回归模型就成了很自然的想法，常见的集成学习模型有Gradient Tree Boosting、Extreme Gradient Boosting（xgboost）等，而xgboost相比与其他Gradient Boosting的模型，速度更快，模型表现更好，所以该项目主要采用的就是基于xgboost的回归模型。下面对线性回归模型和Gradient Boosting方法做一个简单的介绍。

2.3.1 线性回归

给定数据集$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$，其中$x_i = (x_{i1}; x_{i2}; \ldots; x_{id}), y_i \in R$. 线性回归（linear regression）试图学得一个通过属性的线性组合来进行预测的函数。对于本项目来说，就是要从给定的数据集中提取特征，通过对这些特征的线性组合来预测销售额。线性回归使用最佳的拟合直线在因变量（销售额）和特征（自变量）之间建立一种关系。而最佳拟合直线的求解是通过最小二乘法来完成，在线性回归中，最小二乘法就是试图找到一条直线，使所有样本到直线上的欧氏距离之和最小。但是基于最小二乘的参数估计依赖于不同特征之间的独立性，希望不同特征之间的相关性越小越好，另外线性回归对异常值非常敏感，异常值会严重影响回归线，最终影响预测值。

2.3.2 Gradient Boosting

Gradient Boosting（梯度提升）是一种集成弱学习模型的机器学习方法。机器学习模型主要的目标是得到一个模型F，使得预测值$\hat{y} = F(x)$与真实值y之间的误差尽可能小。Gradient Boosting采取分层学习的方法，通过m个步骤来得到最终模型F，其中第m步学习一个较弱的模型F_m，在第$m+1$步时，不直接优化F_m，而是学习一个基本模型$h(x)$，使得其拟合残差项$y - F_m$，这样就会使$m+1$步的模型预测值$F_m = F_m + h(x)$更接近于真实值y，因而目标变成了如何找到$h(x) = F_{m+1} - F_m$，最终就是要找到某类函数空间中的一组$h(x)$使得

$$F(x) = \sum_{i=1}^{M} \gamma_i h_i(x) + const$$

Gradient Boosting算法的步骤如下：

对于给定的输入：训练数据集$\{(x_i, y_i)\}_i^n$，损失函数$L(y, F(x))$，迭代次数M。

1. 初始化模型为常数值：

$$F_0(x) = argmin_\gamma \sum_{i=1}^{n} L(y_i, \gamma)$$

2. 迭代生成M个基学习器
    1. 计算伪残差

$$r_{im} = -[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}]_{F(x)=F_{m-1}(x)} \ for\ i = 1, \ldots, n.$$

    2. 基于$\{(x_i, r_{im})\}_{i=1}^n$生成基学习器$h_m(x)$
    3. 计算最优的γ_m

$$\gamma_m = argmin_\gamma \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$$

    4. 更新模型

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

Gradient Boosting算法中最典型的基学习器是决策树。Gradient Boosting Decision Tree(GBDT)就是一种结合决策树的Gradient Boosting算法实现。这里的决策树是回归树，GBDT中决策树是个弱模型，树的深度和叶子节点的数量一般都较小。

另外一种很常用的Gradient Boosting算法的实现是xgboost，xgboost是Gradient Boosting算法的一种高效实现，以其出众的效率和较高的预测准确度在Kaggle比赛中崭露头角。xgboost中的基学习器除了可以是CART（gbtree），也可以是线性分类器（gblinear）。

2.3.3 技术

项目中将会先使用开源的Python机器学习库scikit-learn中的线性回归模型LinearRegression简单看下效果，之后会主要使用xgboost方法。scikit-learn是一个非常常用的机器学习方面的Python库，提供了大量监督学习和无监督学习算法，调用起来十分方便。xgboost是Gradient Boosting Machine的一个C++实现，最大的特点在于它能够自动利用CPU的多线程进行并行，同时在算法上加以改进提高了精度。为了方便使用，作者陈天奇将xgboost封装成了Python库。

传统GBDT在优化时只用到一阶导数信息，xgboost则对代价函数进行了二阶泰勒展开，同时用到了一阶和二阶导数。xgboost在代价函数中加入了正则项，用于控制模型的复杂度，学习出来的模型更加简单，防止过拟合。xgboost在进行完一次迭代后，会将叶子节点的权重乘上缩减（shrinkage）系数，主要是为了削弱每棵树的影响，让后面有更大的学习空间。

2.4 基准模型

本项目采用具有相同特征【$Store$，$DayOfWeek$，$Promo$】的数据子集的销售额的中位数作为基准模型。因为从前面的分析可以得知，$DayOfWeek$和$Promo$这两个特征和销售额的相关性较高，同时基于每个门店的数据有各自的规律的假设，另外选取中位数是因为其不受极值影响的特点。最终基准模型在Kaggle上提交的得分如下：

Public Score（Rank）	**Private Score（Rank）**
0.14001（2316/3303）	0.14598（2094/3303）

另一方面，鉴于这个项目来自Kaggle上已经结束的一个比赛，所以我们可以利用Kaggle上该比赛的Public Leaderboard和Private Leaderboard上的排名和得分来作为参考。Public Leaderboard上的得分是利用大约39%的测试数据来计算的，Private Leaderboard上的得分是利用剩余的61%的测试数据来计算的。目前Public Leaderboard上的最高得分是0.08933，Private Leaderboard上的最高得分是0.10021。

这个比赛当时参加的队伍总数是3303个，以排名TOP10%，TOP15%，TOP20%来分别划分，在Private Leaderboard上对应的得分分别为0.11774，0.11960，0.12022。基于此，项目设定的基准目标是在Private Leaderboard上进入TOP20%，即RMSPE误差小于0.12022；在基准目标的基础上，争取进入TOP15%，最终目标是进入TOP10%。

3 方法

3.1 数据预处理

数据预处理部分主要包括缺失值的处理、特征值或者类型的转换等。

缺失值的处理：

- 将$test$数据集中的$Open$列中的缺失值用1填充，也就是说将未知开放状态的门店全部当做开放状态处理。这样处理的原因在于，如果门店实际是关闭的，那么它的销售额就为0，不计入最后的模型误差评分，如果门店实际是开放的，那么如果我们当做关闭处理，就会出现很大误差，对模型最终的误差评分产生较大影响。
- 将$store$数据集中的$CompetitionDistance$列的缺失值用0填充。
- 将$store$数据集中的$CompetitionSinceYear$和$CompetitionSinceMonth$分别用1900和1填充，亦即用具有最久历史的竞争对手的开业时间来填充缺失值。

特征的值或类型的转换：

- 将$StateHoliday$列中的数字0转换为字符串类型'0'，原始数据中混杂了数值0和字符串'0'，但是它们代表的是同一种含义，类型的混杂不便于后期类型特征的转换；
- 将$Date$列中的日期信息进行分离，得到不同类别的时间信息，包括$Year$，$Month$，$DayOfMonth$，$WeekOfYear$，$DayOfYear$。
- 将目标预测变量$Sales$进行对数处理，便于模型RMSPE误差的计算。
- 将$train$和$test$数据集中的$StateHoliday$，$DayOfWeek$以及$store$数据集中的$Assortment$和$StoreType$转换成类别信息，并进行类型编码。

3.2 特征提取

提取特征的好坏很大程度上决定了模型最终的表现，所以特征提取是整个项目执行过程中非常重要的一个环节。由于事先并不知道哪些特征会对最终的预测结果有帮助，所以项目中采取的特征提取策略就是尽可能多的提取特征，然后再进行特征选择。最终项目一共提取了28个特征，详细介绍如下：

- 原始数据集中直接存在的特征，包括$Store$，$CompetitionDistance$，$Promo$，$Promo2$，$StoreType$，$Assortment$，$StateHoliday$，$SchoolHoliday$，$Year$，$Month$，$WeekOfYear$，$DayOfWeek$，$DayOfMonth$，$DayOfYear$
- $PromoOpenInMonth$：表示从门店开始进行持续促销的日期开始，到当前日期所经历的时长，以月为单位来进行计算
- $CompetitionOpenInMonth$：表示门店的竞争对手从开业到当前日期所经历的时长，同样以月为单位来进行计算
- $IsPromoMonth$：表示当前月份是否在门店开始进行促销的月份里，如果是，标记该特征的值为1，否则标记为0
- $AvgSales$：表示每个门店在开放的天数里的平均销售额
- $AvgCustomers$：表示每个门店在开放的天数里的平均顾客数
- $AvgSalesPerCustomer$：表示每个门店每位顾客贡献的平均销售额
- $medianCustomers$：表示每个门店顾客数的中位数
- $holidays\_thisweek$，$holidays\_lastweek$，$holidays\_nextweek$：分别表示当前周、上周和下周的学校节假日的天数
- $AvgSalesPerDow$，$medianSalesPerDow$：表示每个门店每周一到每周日的平均销售额和销售额的中位数
- $AvgCustsPerDow$，$medianCustsPerDow$：表示每个门店每周一到每周日的平均顾客数和顾客数的中位数

3.3 建立模型

在本项目中主要尝试了两种模型：线性回归模型和基于xgboost的回归模型。在建立模型的过程中，也尝试了不同的特征组合，看不同的特征子集对模型表现的影响。在使用基于xgboost的回归模型时，使用了模型融合的技术，将几个不同的xgboost模型进行了融合，结果表明，效果相比单一模型有一定的提升。

对于模型训练集和交叉验证集的划分，考虑到模型最终是要预测未来连续六周的销售额，所以采用了$train$数据集中最后六周的训练数据作为交叉验证集，其余的数据用来训练，这样的策略更符合模型的目标：使用历史销售数据来预测未来的销售数据。而不是一般采用的将训练集随机划分为训练集和交叉验证集的策略。

3.3.1 线性回归模型

线性回归模型作为回归问题中最简单的模型，是项目中最先尝试的模型。该项目的最终目标是要预测每个门店未来六周每天的销售额，鉴于此，线性回归模型也可以有两种应用方法。第一种是在整个训练数据集上应用线性回归模型，第二种是对每个门店单独应用线性回归模型，对每个门店的销售额分别进行预测。下面分别对这两种思路进行实验，对比结果。

先来看看在整个数据集上应用线性回归模型的情况。由于提取的特征相对较多，为了防止过拟合，所以模型并没有把提取的全部特征都用在线性回归模型上，而是随机手选了部分特征。手选的特征包括【$PromoOpenInMonth$，$holidays\_thisweek$，$Promo$，$SchoolHoliday$，$Year$，$Month$，$DayOfWeek$，$StateHoliday$，$AvgSales$，$AvgSalesPerCustomer$】，最终结果如下：

Public Score（Rank）	Private Score（Rank）
0.224（2824/3303）	0.249（2829/3303）

可以看出在整个数据集上用手选的特征应用线性回归模型的效果非常差。一方面可能是因为选取的特征不够好的原因；另一方面原因可能是整个数据集上的单个回归模型不能很好地拟合每个门店的情况。

下面来看看分别对每个门店单独应用线性回归模型的情况。为了与前一种回归模型的应用方法进行比较，对每个门店单独应用线性回归模型时，选取的特征和前一种回归模型选取的特征一样。最终结果如下：

Public Score（Rank）	**Private Score（Rank）**
0.183（2675/3303）	0.224（2745/3303）

对比在整个数据集上应用回归模型的表现，可以看出对每个门店单独应用线性回归模型的分数有所提升，但提升不是很明显。当利用每个门店的数据来拟合线性回归模型时，训练数据量相对较小，但是选取的特征相对较多，不利于拟合出很好的线性回归模型。于是尝试去掉一些特征重新对每个门店单独应用线性回归模型。

选取特征【$holidays\_thisweek$，$Promo$，$DayOfWeek$】进行训练，最终得分如下：

Public Score（Rank）	**Private Score（Rank）**
0.163（2566/3303）	0.179（2572/3303）

选取特征【$Promo$，$DayOfWeek$】进行训练，最终得分如下：

Public Score（Rank）	**Private Score（Rank）**
0.166（2570/3303）	0.177（2565/3303）

可以看出，当选取较少数量的特征时，对每个门店应用线性回归模型的分数有一定的提升。

总体来看，线性回归模型在本项目中的表现较差，甚至比基准模型都差不少。不管是在整个数据集上应用一个回归模型，还是对每个门店单独应用线性回归模型。通过比较发现，对每个门店单独应用线性回归模型的效果要稍微优于在整个数据集上应用线性回归模型。

3.3.2 xgboost模型

xgboost是项目最终使用的模型。在使用xgboost模型的过程中，主要精力放在特征选择和模型融合方面，虽然模型调参也是建模过程中很重要的一个环节，但考虑到投入产出比就没有花过多精力在上面，因为模型调参通常需要迭代很多次，消耗大量计算资源，对于xgboost这样参数较多的模型来说更是如此。另外从经验上来说，相对于特征选择和模型融合来说，模型调参对模型最终的表现可能贡献并不会太大。

应用xgboost的大致思路为：先利用提取的所有特征跑一个基本的xgboost模型，然后再构建100个xgboost模型，每个模型的特征从所有特征中随机选取，模型参数保持不变，再从这100个模型中选出交叉验证集误差最小的模型，对其进行调参，最后从以上获得的模型中选取一些模型进行融合，作为最终结果。详细步骤如下：

1. 利用提取的所有特征训练一个xgboost基准模型；
2. 从xgboost基准模型中，根据特征的重要性排序，选出前16个特征作为下一阶段100个模型的基础特征，剩下的12个特征作为随机选择的特征池；
3. 构建100个xgboost模型。这100个模型的特征由基础特征加上随机选择的特征构成。随机选择的规则为：从12个特征中分别挑选4,5,6,7,8个特征，每类挑选20次，刚好组成100个模型的特征（特征个数为20,21,22,23,24的模型各20个）。
4. 从100个模型中选出交叉验证集误差最小的模型，选取xgboost模型中的几个参数对其进行调参。
5. 从以上步骤得到的模型中选取几个模型进行简单融合，作为最终结果。

下面对xgboost模型构建和训练过程中的细节和结果进行介绍。

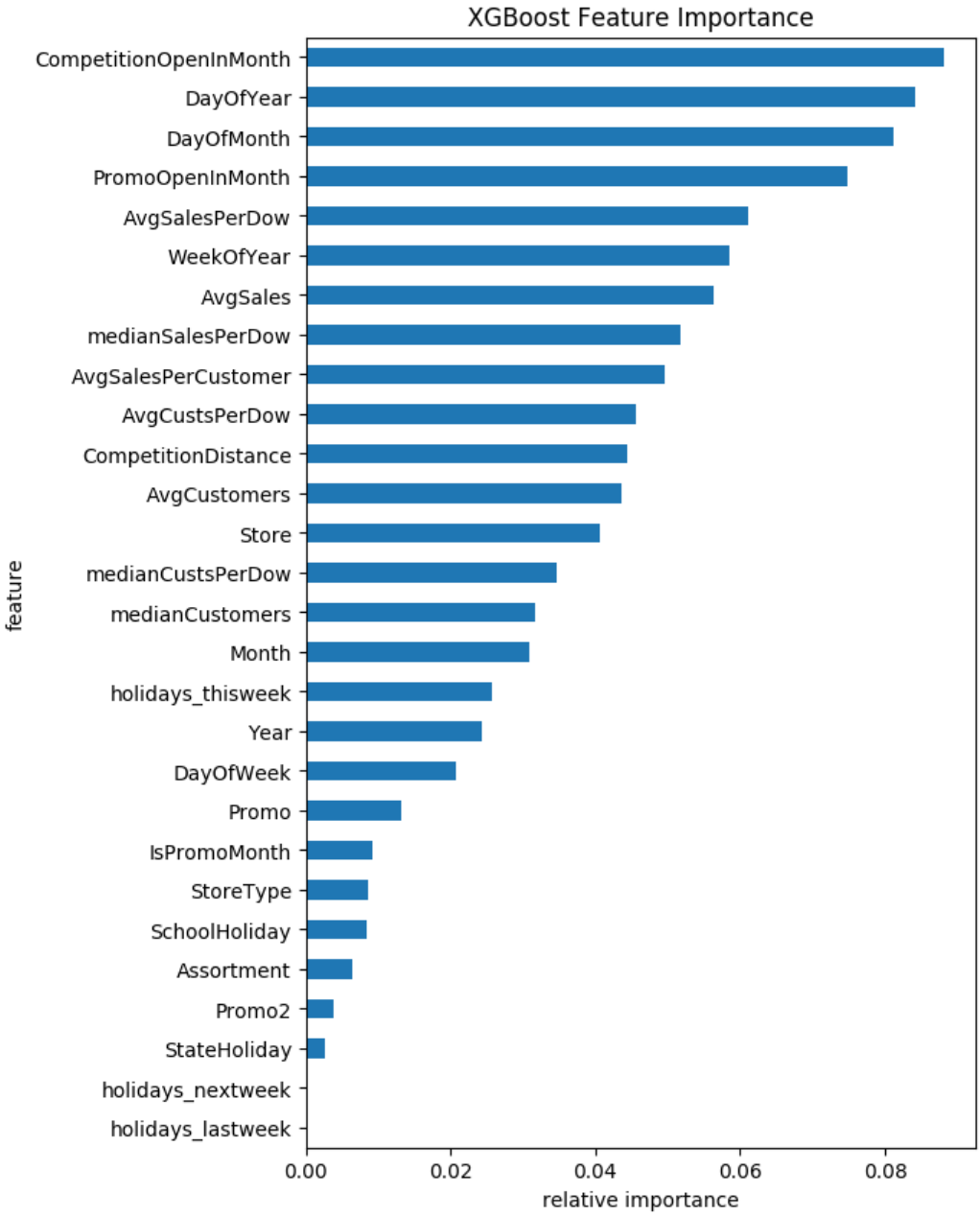
没有调参之前的xgboost模型都使用下面的模型参数：

```
parameters = {
                  'objective': 'reg:linear',
                  'booster': 'gbtree',
                  'eta': 0.03,
                  'max_depth': 10,
                  'subsample': 0.9,
                  'colsample_bytree' : 0.5,
                  'silent': 1,
                  'seed': 1301,
                  'num_round': 3000,
                  'early_stopping_rounds' : 100
              }
```

训练xgboost基准模型时，利用了全部的特征，最终的结果如下：

Train RMSPE	Validation RMSPE	Public Score（Rank）	Private Score（Rank）
0.07656	0.11557	0.11469（1465/3303）	0.12434（1291/3303）

可以看出相比于线性回归模型和基准模型，xgboost模型的表现提升了很多。下面是各个特征的重要性排序。



从特征的重要性排序中，选取了重要性相对较高的前16个特征，即【$CompetitionOpenInMonth$，$DayOfYear$，$DayOfMonth$，$PromoOpenInMonth$，$AvgSalesPerDow$，$WeekOfYear$，$AvgSales$，$medianSalesPerDow$，$AvgSalesPerCustomer$，$AvgCustsPerDow$，$CompetitionDistance$，$AvgCustomers$，$Store$，$medianCustsPerDow$，$medianCustomers$，$Month$】作为后续100个模型的基础特征，其余的特征用来作随机选择。

这100个模型中，交叉验证集误差最小的模型的信息如下，记为模型A：

Features	$CompetitionOpenInMonth$，$DayOfYear$，$DayOfMonth$，$PromoOpenInMonth$，$AvgSalesPerDow$，$WeekOfYear$，$AvgSales$，$medianSalesPerDow$，$AvgSalesPerCustomer$，$AvgCustsPerDow$，$CompetitionDistance$，$AvgCustomers$，$Store$，$medianCustsPerDow$，$medianCustomers$，$Month$，$holidays\_lastweek$，$Promo$，$StoreType$，$StateHoliday$，$DayOfWeek$，$holidays\_thisweek$，$Promo2$，$IsPromoMonth$
Train RMSPE	0.07105
Validation RMSPE	0.1114
Public Score	0.11232（1341/3303）
Private Score	0.13037（1681/3303）

在对模型A进行调参之前，先进行一次模型融合，看看模型融合的效果。从训练过的模型的训练误差和测试误差来看，以上两个模型都出现了严重的过拟合现象。而模型融合就是一种常用的缓解过拟合、提高模型泛化能力的方法。本项目中采用了最简单的平均法，同时对最终结果乘上了一个常数因子，模型融合公式如下：

$$average\_blending = \frac{(y_{pred\_1} + \ldots + y_{pred\_n})}{n} * factor$$

其中y_{pred_n}表示第n个模型的预测值，n为融合的模型个数，$factor$为乘数因子。目前得到两个基于xgboost的模型，一个是模型A，一个是之前的xgboost基准模型，记为模型B。由于模型A和模型B都出现了较为严重的过拟合现象，如果仅仅将它们进行平均数融合，融合后的结果并不理想，甚至比单个xgboost基准模型的效果还差一点。融合后的得分如下：

Public Score（Rank）	**Private Score（Rank）**
0.11306（1370/3303）	0.12475（1304/3303）

为了使融合后的模型具有更好的泛化能力，于是又从之前得到的100个模型中随机选取了两个特征数相对较少、交叉验证误差较大的模型C和模型D。模型C和模型D的特征数目分别为20和21个，交叉验证误差分别为0.1219和0.1262。对这样四个模型进行平均数法融合，乘数因子选为0.995，模型融合后的结果提交到Kaggle上，得到的分数如下：

Public Score（Rank）	**Private Score（Rank）**
0.10961（1243/3303）	0.11984（531/3303）

可以看出这样四个模型融合后的结果相对单一模型来说，泛化能力有较大的提升。下面对模型A进行调参，看调参后的模型能不能提升模型融合的最终效果。

对模型A选取了6个xgboost参数进行调参，分别为：

- max_depth：一棵树的最大深度值，用于控制模型的拟合情况；
- $subsample$：用于训练模型的子样本占整个样本集合的比例
- $colsample\_bytree$：在建树时对特征采样的比例

- eta：更新过程中用到的收缩步长，eta通过缩减特征的权重使提升计算过程更加保守
- num_round：提升迭代计算的次数
- $early\_stopping\_rounds$：提前终止循环的次数

鉴于计算资源的限制，参数值范围的选取都比较粗放，具体的调参步骤如下：

1. 先对max_depth，$subsample$，$colsample_bytree$三个参数进行优化，保持其它参数不变。其中max_depth的取值范围为【6,7,8,9,10】，$subsample$的取值范围为【0.7,0.8,0.9】，$colsample_bytree$的取值范围为【0.5,0.6】；
2. 在第1步中得到的三个参数的最优值的基础上，将eta参数值从原来的0.03减小为0.01，同时增大参数num_round的值为15000，$early_stopping_rounds$参数值为600。

经过上面的调参过程，最终调整后的模型A的参数如下：

```
parameters = {
                    'objective': 'reg:linear',
                    'booster': 'gbtree',
                    'eta': 0.01,
                    'max_depth': 10,
                    'subsample': 0.7,
                    'colsample_bytree' : 0.5,
                    'silent': 1,
                    'seed': 1301,
                    'num_round': 15000,
                    'early_stopping_rounds' : 600
                }
```

调参后模型A的结果如下：

Train RMSPE	Validation RMSPE	Public Score	Private Score
0.069343	0.110677	0.11349（1400/3303）	0.12450（1297/3303）

通过与调参前的模型A的结果进行对比，可以发现，调参后的模型的交叉验证集误差有一定的降低，在测试集上的表现也有所提升。

下面再按照之前的模型融合方式对调参后的模型A和其他三个模型B，C，D进行融合，融合后的得分如下：

Public Score（Rank）	Private Score（Rank）
0.10943（1235/3303）	0.11934（464/3303）

可以看出，调参后模型融合的结果相比于调参前又有了一定的提升，这也是目前得到的最好结果，于是就将其作为最终结果。

4 结果

4.1 模型的评价与验证

最终选取的是几个xgboost模型融合后的结果，在Kaggle上的得分和排名都超过了之前定的基准目标，即进入TOP20%，也达到标准目标，即进入TOP15%，但没能达到终极目标，进入TOP10%。

回顾整个建模过程，从数据预处理、特征提取、构建基准模型，之后从简单线性回归模型开始，发现线性回归模型无论是应用在整个数据集上，还是应用在单个门店上，拟合效果都不理想，比基准模型效果还差。于是决定采用xgboost模型。在应用xgboost模型的过程中，首先利用所有特征训练一个基准xgboost模型，效果比之前所有的模型都要好，但是出现了较严重的过拟合现象，由于本项目最终是要使用模型融合技术来减轻单个模型的过拟合现象，所以该基准模型作为被融合模型之一暂时被放在一边，接下来再构建几个xgboost模型。xgboost基准模型出现过拟合现象的原因之一可能是特征数太多，于是采取从所有特征中随机选取不同数量的特征子集来训练xgboost模型的策略，构建了100个模型，然后从这100个模型中选出交叉验证集误差最小的模型进行调参，得到另一个将要被融合的模型。之后又从100个模型中手选了两个模型，最终将这四个模型进行了简单的平均数融合。整个过程中使用了模型选择、模型调参、模型融合等技术，融合后的模型在测试集上的Public Score和Private Score分别为0.10943和0.11934，说明融合后的模型有了较好的泛化性能。虽然仍然存在一定的过拟合，但相对于单个模型，融合后的模型鲁棒性更好。模型的参数也在常见的参数范围内。

4.2 合理性分析

最终结果与基准模型在测试集上的Public Score和Private Score的对比如下表所示：

	Public Score	**Private Score**
基准模型	0.14001（2316/3303）	0.14598（2094/3303）
最终模型	0.10943（1325/3303）	0.11934（464/3303）

可以看出最终模型在测试集上的Public Score和Private Score均小于基准模型，并且Score的值也相对较小，结合项目的评价标准RMSPE的定义，说明最终模型确实具有较好的泛化性能，具体来说，最终模型对门店未来六周的销售额预测误差较小，准确度较高，具有很高的参考价值。

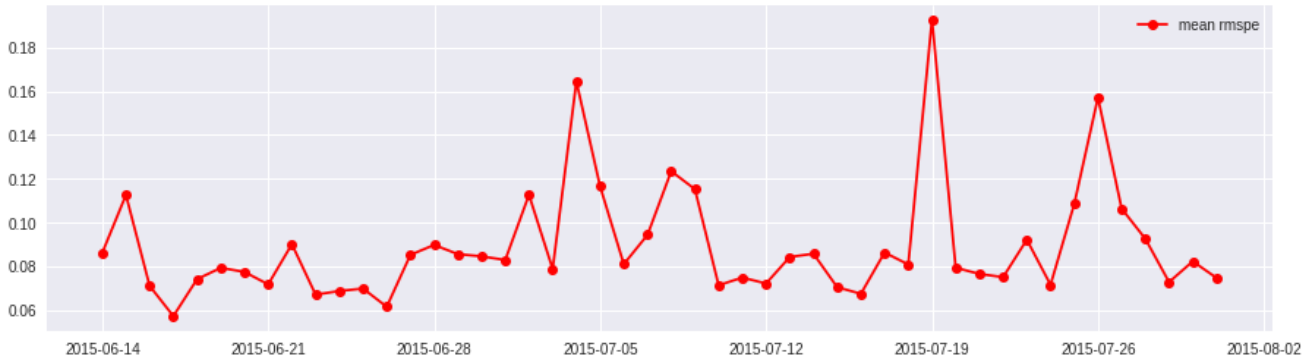
5 项目结论

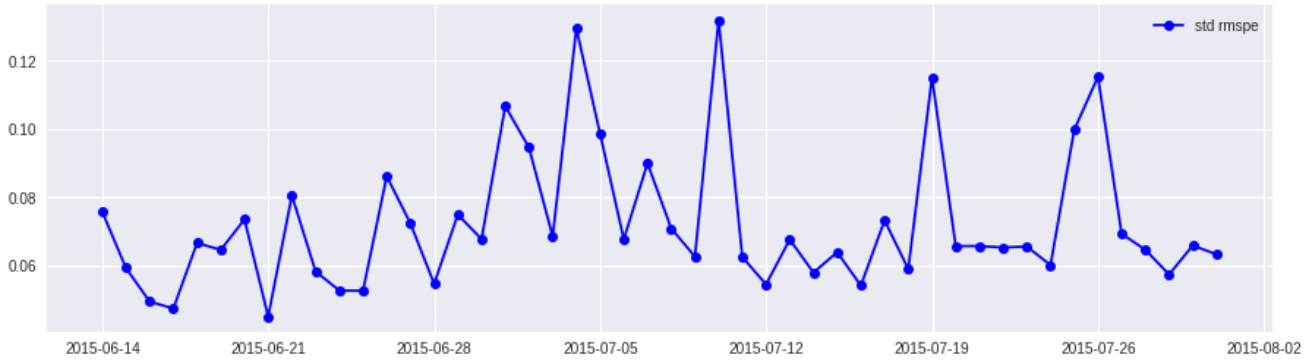
5.1 结果可视化

结果的可视化在交叉验证集上进行。交叉验证集中包含了不同门店六周时间内的销售额数据，结果可视化从以下三个方面来进行：

- 对每个日期内的所有门店的实际销售额和预测销售额的RMSPE绝对值的均值和标准差进行可视化
- 对每个门店所有日期内的实际销售额和预测销售额的RMSPE绝对值的均值和标准差进行可视化
- 由于门店数量太多，故选取几个门店，画出其六周内的实际销售额走势和预测销售额走势图，看看它们之间的吻合情况

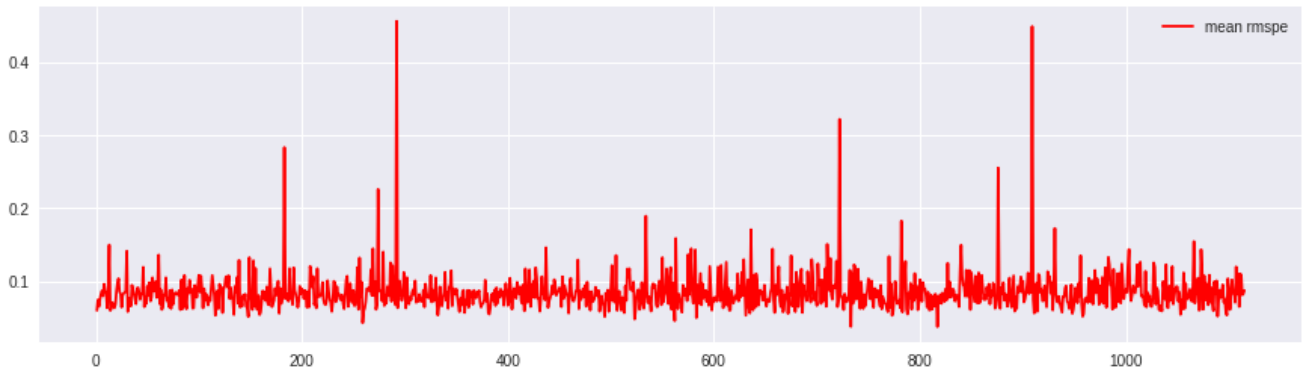
下图是按照日期进行可视化的结果：

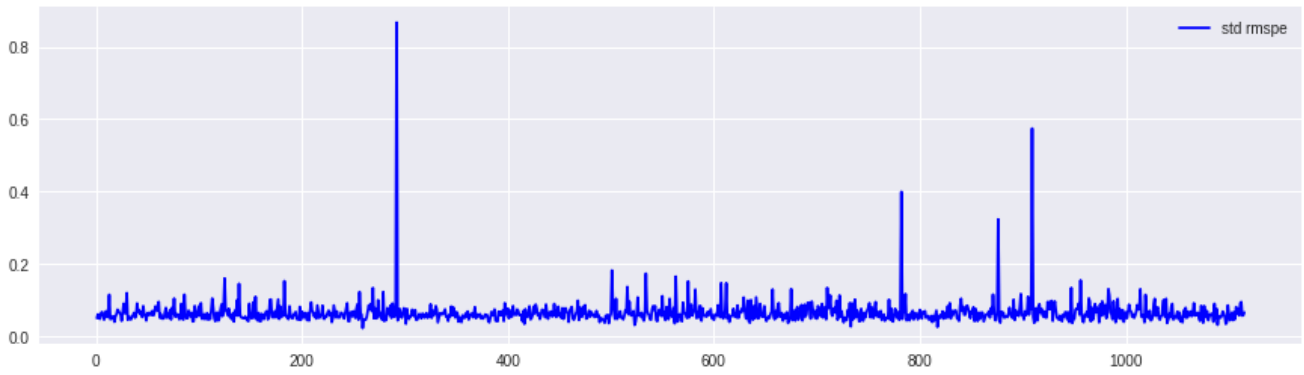




从上图可以看出，大部分日期内的RMSPE均值均在0.06和0.1之间波动，有少数日期的RMSPE均值超过0.1，比如2015-07-04、2015-07-08前后、2015-07-19和2015-07-26前后等，大部分日期内的RMSPE标准差在0.05和0.1之间波动，有少数日期的RMSPE标准差超过0.1。同时对比RMSPE均值和标准差的走势图可以发现，均值较小的时间点其对应的标准差也相对较小，说明模型的预测结果相对比较稳定可靠。

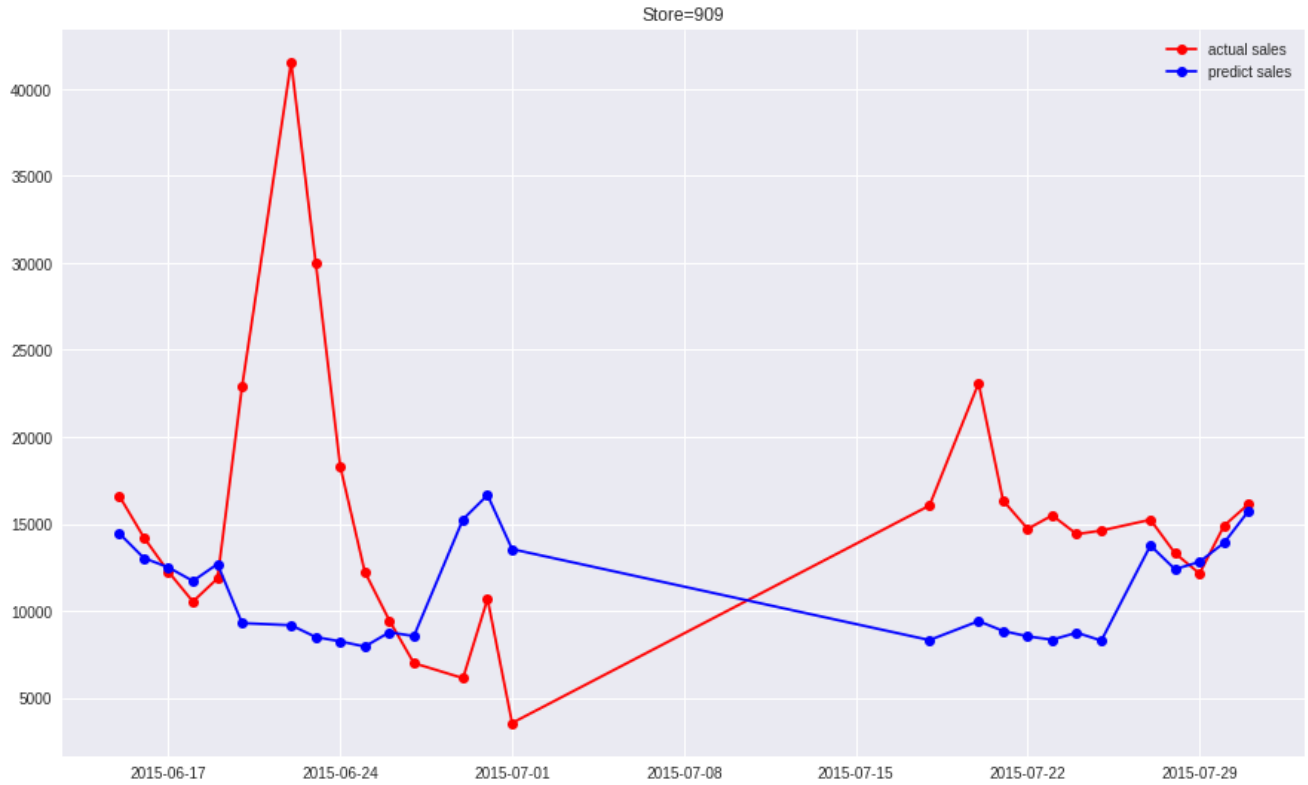
下图是按照门店进行可视化的结果：

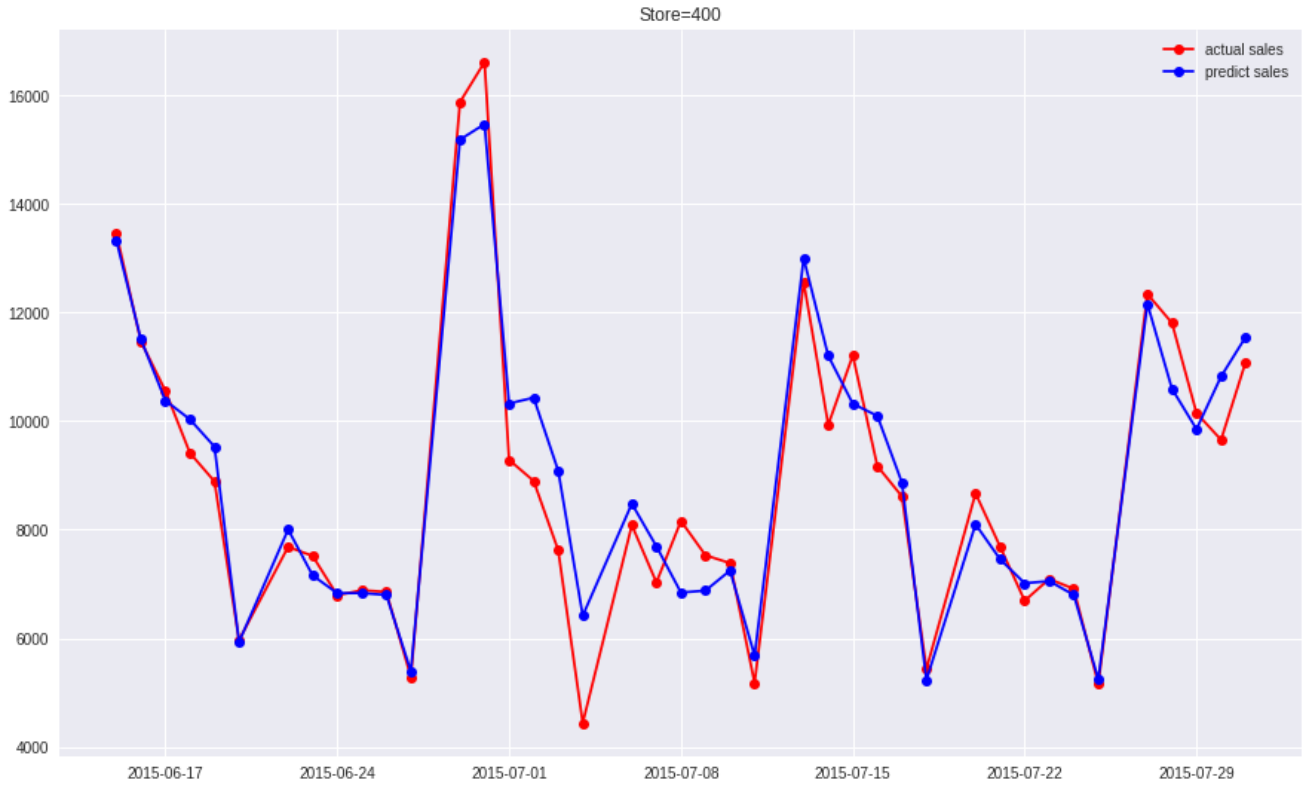


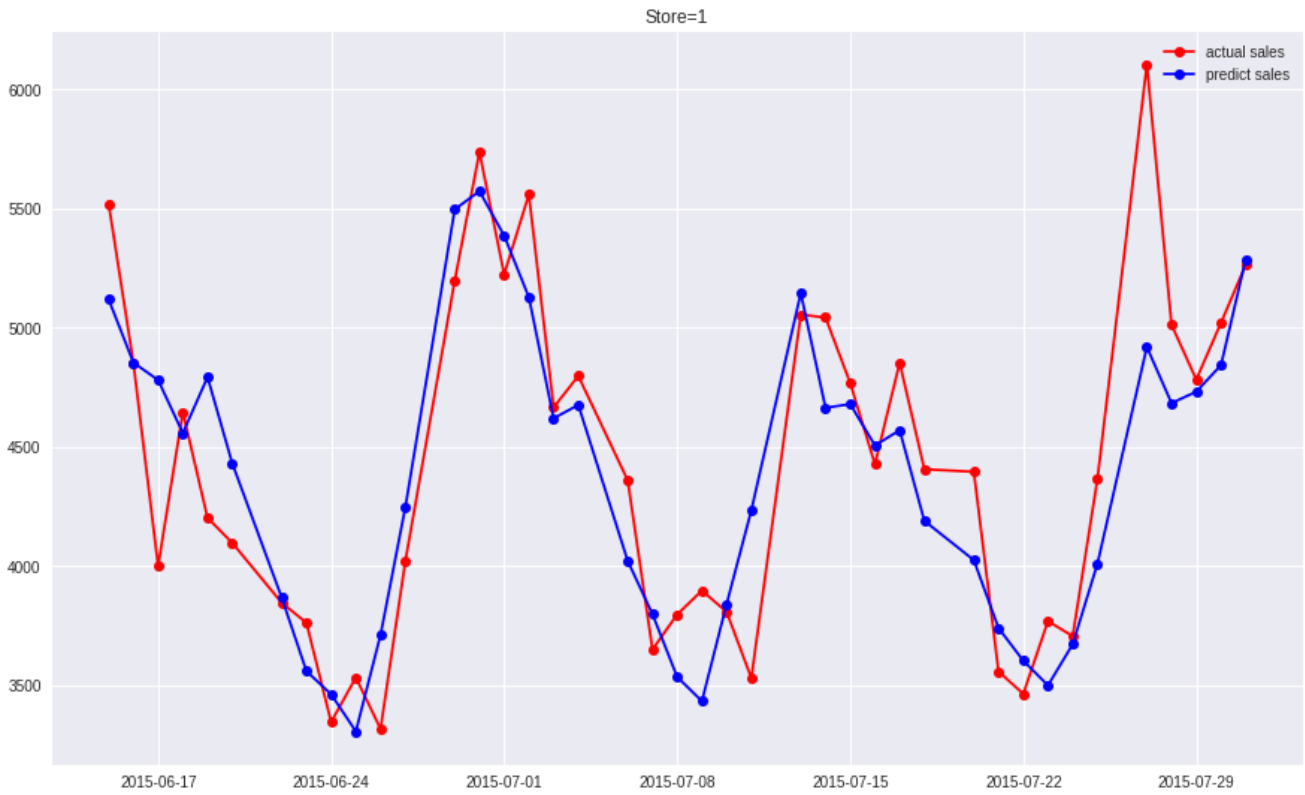


从上图可以看出，大部分门店在六周内的RMSPE均值和标准差在0.1附近波动，有少数门店六周内的RMSPE均值和标准差远远大于0.1，说明模型对这些门店的预测效果不太理想。但这些门店的数量相对较少，所以总体而言，模型的预测结果比较稳定可靠。

下面几幅图是选取的几个门店的六周内的实际销售额和预测销售额的走势图：







可以看出，对于Store 1和Store 400来说，六周内的销售额预测走势和实际值比较吻合，对于Store 292和Store 909来说，六周内的销售额预测走势和实际值相差就比较大，对应到之前的按照门店进行可视化的图中RMSPE均值大于0.4的门店的情况。

5.2 对项目的思考

本项目的整个流程主要包括以下几个步骤：

- 数据的探索和可视化
- 基准模型的确立
- 数据预处理
- 特征提取
- 线性回归模型的两种尝试
- xgboost模型的确立（模型选择、模型调参、模型融合）
- 结果分析与提交

在整个项目过程中，特征提取花了比较多的精力，因为特征的好坏对模型最终的表现具有很大的影响，好的特征能大大提升模型最终的效果。在模型方面，当确定使用xgboost模型后，得到的第一个xgboost模型的表现相比于基准模型就有很大提升，但也伴随着较为严重的过拟合问题，于是想到用模型调参和模型融合的技术来降低过拟合的影响，模型融合的过程中就涉及到选择哪些模型进行融合的问题，需要进行大量的尝试，这是项目比较困难的地方。但另一方面，每次尝试之后，如果看到模型表现有了一定的提升，即使提升不是很显著，也是很有意思的一件事。模型最终的表现也比较符合预期，本项目中使用到的技术和方案对于回归类预测的问题具有一定的通用性。

5.3 需要作出的改进

本项目可以完善和尝试的地方有很多，下面举出一些：

- 在模型调参和模型融合部分，会发现这些技术对模型的提升效果不是太明显，所以可以返回到特征提取部分，尝试再提取一些特征，看看对模型的提升效果如何，或许这样的尝试投入产出比更高；
- 在模型方面，除了xgboost模型，还可以尝试使用其他的技术，比如深度学习，这个比赛的第三名就是使用的深度学习方法。

6 参考

1. 周志华《机器学习》第三章线性模型
2. wikipedia: gradient boosting
3. 一步一步理解GB、GBDT、xgboost
4. xgboost: 速度快效果好的Boosting模型
5. xgboost浅入浅出
6. Model documentation 1st place
7. Github: SuyashLakhotia/RossmannStoreSales
8. Arshay Jain: Complete Guide to Parameter Tuning in XGBoost (with codes in Python)