机器学习纳米学位

刘伟

2019年6月27日

I. 问题的定义

项目概述

Rossmann是欧洲的一家连锁药店，在7个欧洲国家中用超过3000家连锁药店。Rossmann各个药店经理需要对未来6周内每天销售额进行预测，这样可以提前进行合理的资源分配。影响药店销售额的因素有很多，包括促销，竞争对手，节假日，季节以及地理位置，这个项目的目标是基于各个门店过往的销售数据，以及门店本身的信息，对未来6周的销售额进行预测，项目提供的数据集为1115个Rossmann门店的历史销售记录和这些门店的相关信息。

问题陈述

项目要求预测各个门店，未来6周内每天的销售额，这个问题属于一个回归问题，预测需要考虑的特征包含各个门店的相关信息，比如地理位置，附件竞争对手，促销活动，历史销售额等。具体来说，大致步骤可分为：

- 数据探索性分析：包括对项目提供的门店历史销售数据，以及门店的相关数据进行探索了解，例如数据丰富度，是否存有大量缺失数据，各个特征的数据分布情况等。
- 特征预处理：包括空值填充，类别特征处理等。
- 特征工程：包括特征组合，特征筛选等。
- 构建模型：选择合适的模型进行训练，可选用包括树模型（xgboost，randomForest），RNN深度模型等。
- 模型评估：定义模型评价指标：rmspe。
- 模型分析及优化：结合模型预测的结果，分析优化方向，例如提取更丰富的特征，选择不同的模型等。

通过以上各个步骤，不断对模型以及数据特征处理，特征工程等方法的优化，在验证集上达到比较高的准确度，体现在对各个门店的未来6周的销售额预测和实际销售额的准确率。

评价指标

首先这个问题是销售预测问题，属于回归模型，销售额的准确性是评估模型的指标，采用的评价指标是RMSPE：

RMSPE=\sqrt { \frac { 1 }{ n } \sum *{ i=1 }^{ n } (\frac { y*{ i }-\hat { y*{ i } } }{ { y }*{ i } })^{ 2 } }

II. 分析

数据的探索

项目一共有三份数据：用于训练的train.csv和store.csv，以及用于测试的test.csv，其中train.csv包含了各个门店历史每天的销售情况：

- Store：各个门店的ID标识，一共有1115家门店
- Date：日期，从2013-01-01到2015-07-31一共942天的数据
- DayOfWeek：1-7分别表示周一到周日
- Sales：某门店某天的销售额，最小0，最大41551
- Customers：某门店某天的顾客数，最小0，最大7388
- Open：某门店某天是否营业，0表示没有营业，1表示营业
- Promo：某门店某天是否有进行促销，0表示没有促销，1表示有促销
- StateHoliday：当天是否是节假日，以及是哪个节假日，0表示非节假日，a表示公共假日，b表示复活节，c表示圣诞节
- SchoolHoliday：当天是否是学校假日，0表示不是，1表示是

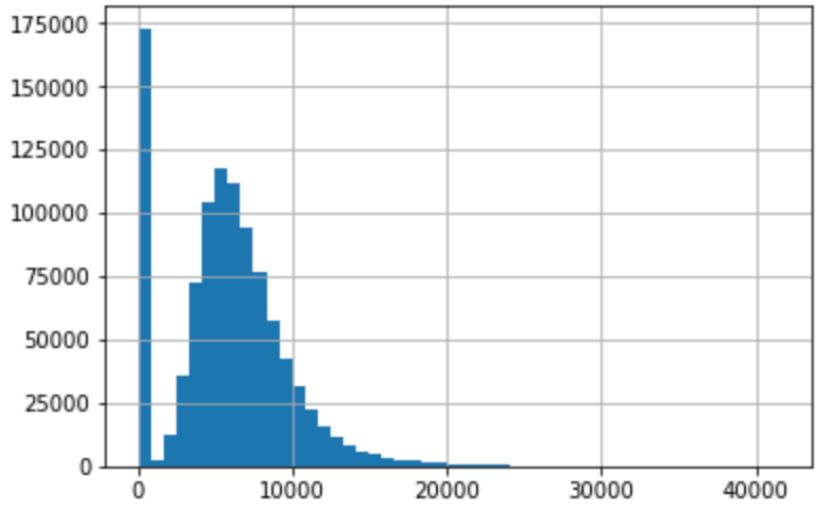
store.csv包含了各个门店的相关信息：

- Store：各个门店的ID标识，和train.csv的Store字段对应
- StoreType：门店的类型，一共有4种门店，分别用a，b，c，d表示，四种门店类型中，b类型的门店数量最少，只有17家
- Assortment：门店的商品类型，一共有3种，分别用a，b，c表示，其中b类型商品的门店数量最少，只有9家
- CompetitionDistance：离门店最近的竞争对手门店距离
- CompetitionOpenSinceMonth：离门店最近的竞争对手门店开业的月份
- CompetitionOpenSinceYear：离门店最近的竞争对手门店开业的年份
- Promo2：表示门店是否有参与进行持续的促销活动，0表示没有参与，1表示有参与
- Promo2SinceWeek：表示门店参与持续促销的周数，NaN表示门店未参与持续促销
- Promo2SinceYear：表示门店参与持续促销的开始年份，NaN表示门店未参与持续促销
- PromoInterval：表示门店每次开始持续促销的月份，NaN表示门店未参与持续促销

探索性可视化

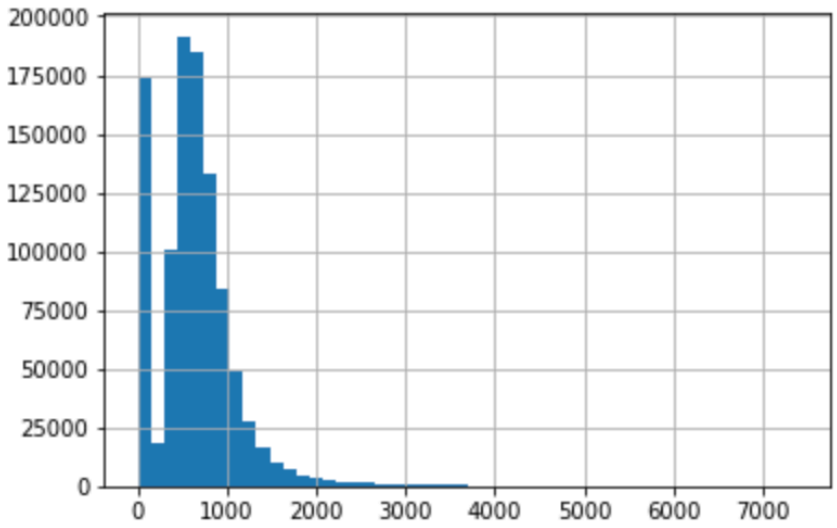
- 训练数据中的Sales，主要集中在4000-8000的范围内，有一部分数据的Sales是0，数值的分布如下图：

```
In [5]: train_df['Sales'].hist(bins=50)
Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x11193d550>
```



- 训练数据中的Customers，主要集中在300-1000，有一部分数据的Customers是0：

```
train_df['Customers'].hist(bins=50)
Out[3]: <matplotlib.axes._subplots.AxesSubplot at 0x112c5e400>
```

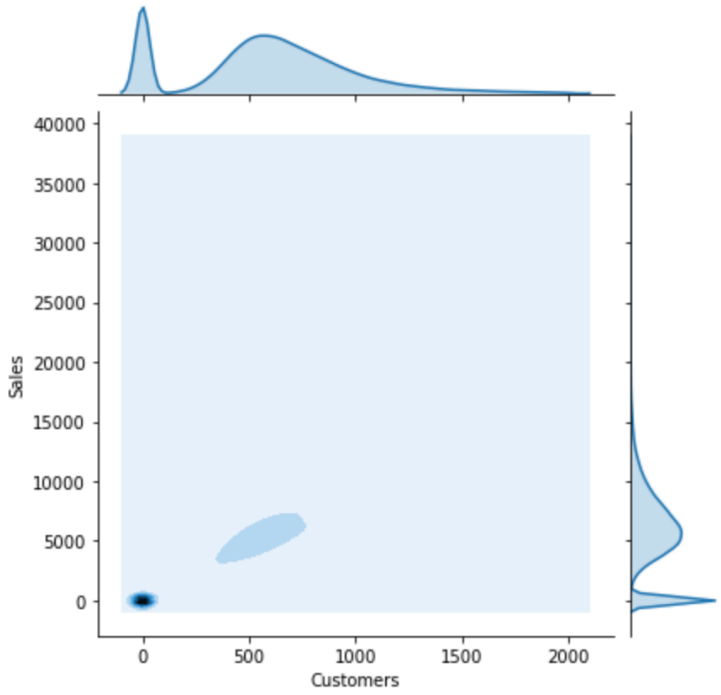


- 训练数据中，Customer人数和Sales销售额成正比关系：

```
In [7]: sample_df = train_df.loc[train_df['Customers'] < 2000].sample(100000)

        sns.jointplot(sample_df['Customers'], sample_df['Sales'], kind="kde")
```

```
Out[7]: <seaborn.axisgrid.JointGrid at 0x111b53518>
```

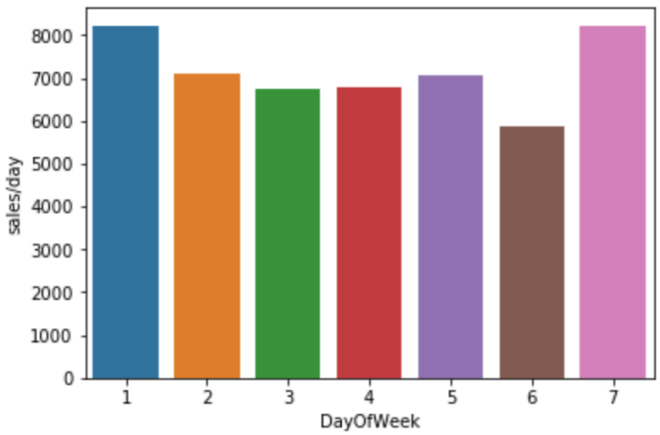


- DayOfWeek的每一天里销售额的分布也有一定规律，首先看到每天对应的销售额分布，星期一的销售额最高，周日的销售额最低，但发现周日里药店开业的数量也是最低的，因此重新对比周一到周日每天有营业的平均销售额分布，如下图：

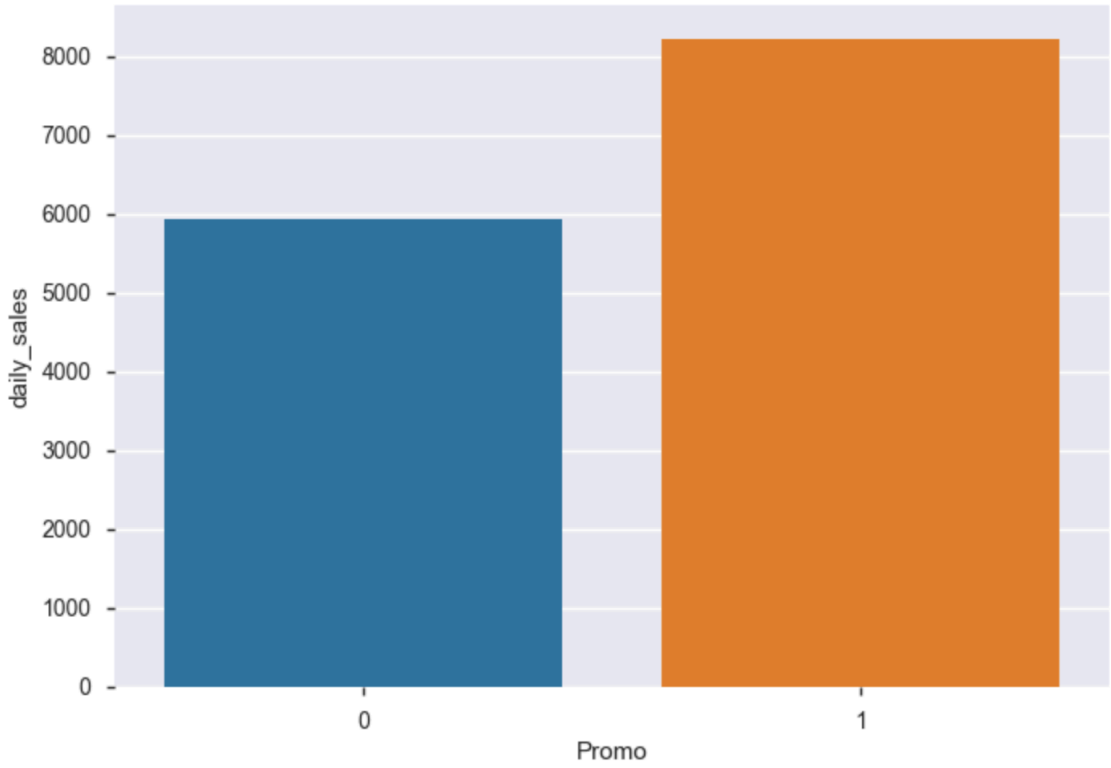
```
dayAvgSales = dayOfWeek_Sales.merge(open_days, how='inner', on = ['DayOfWeek'])
dayAvgSales['sales/day'] = dayAvgSales['Sales'] / dayAvgSales['Open']

sns.barplot(x=dayAvgSales['DayOfWeek'], y=dayAvgSales['sales/day'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x12410a4a8>
```

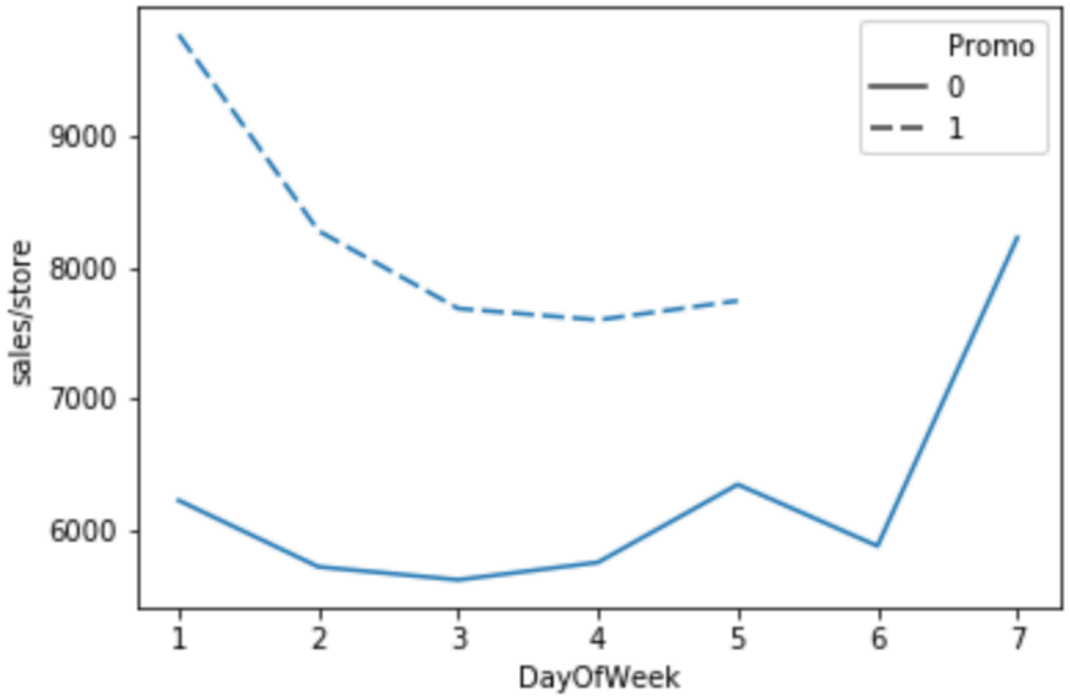


- Promo和销售额的关系，整体平均来看，有促销的销售额比没有促销的销售额要高：

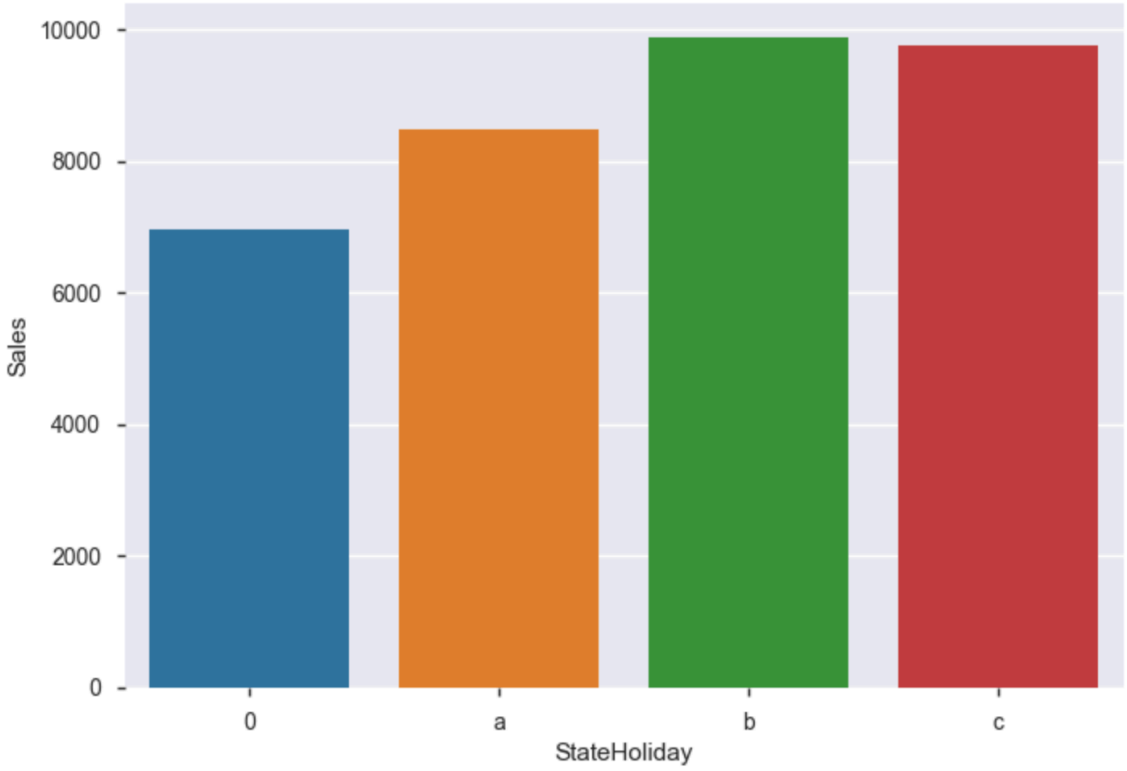


同时，如果把Promo, DayOfWeek结合一起，看有促销和无促销，分别对应每天的平均销售额分布关系，可以发现：

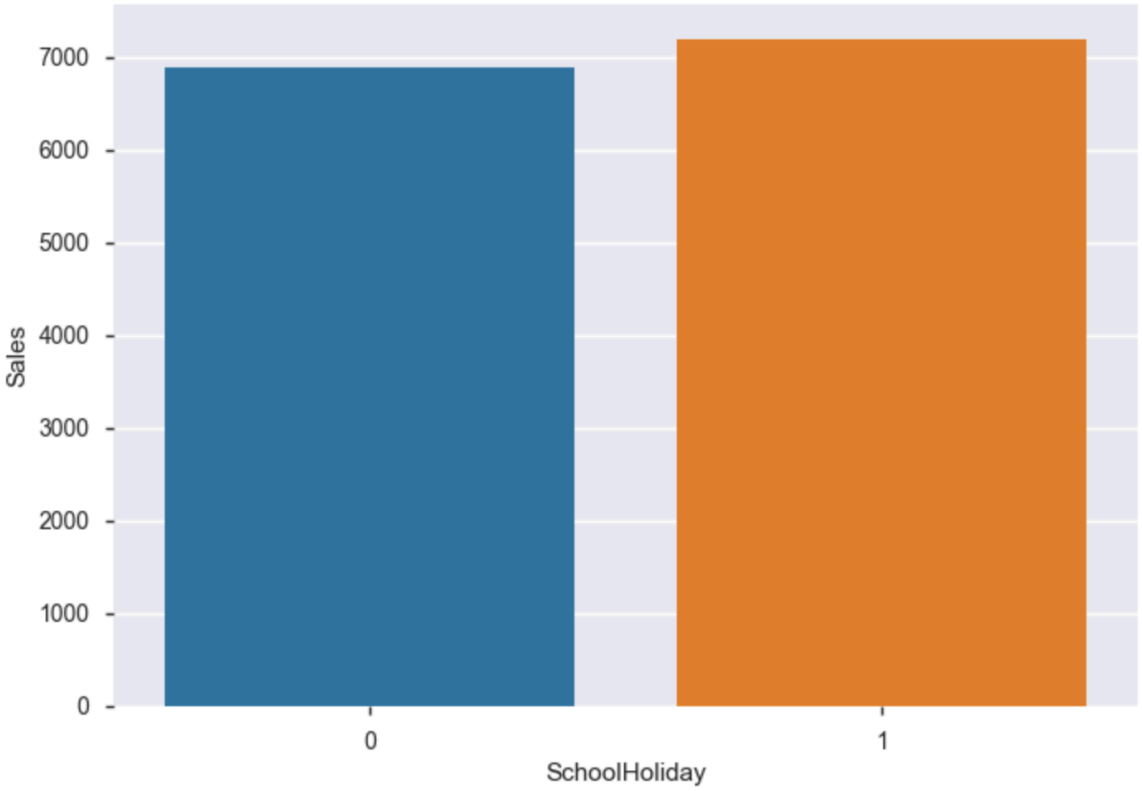
- 周六和周日都没有促销活动；
- 在没有促销活动的时候，如果药店营业，周日的销售额最高；
- 在有促销活动时，周一的销售额最高



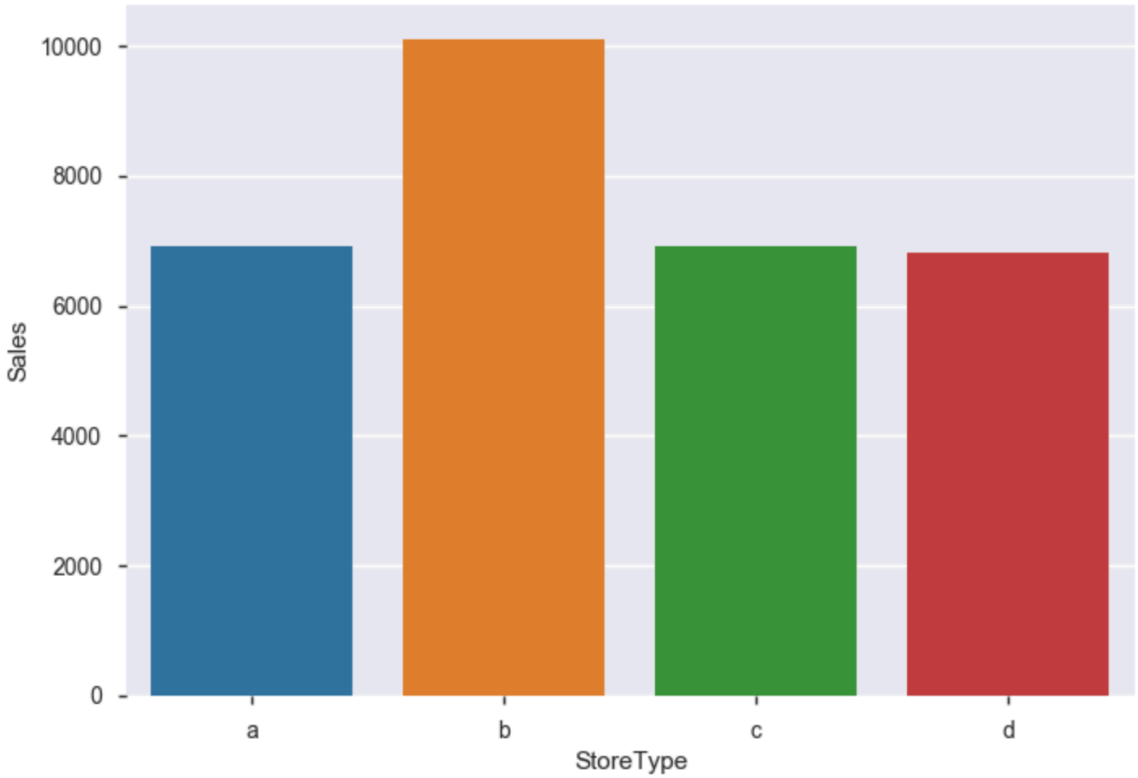
- StateHoliday与销售额的关系，有节假日的药店平均销售额比没有节假日的要高，同时，复活节和圣诞节的平均销售额最高，另外，在训练样本中，StateHoliday存在数字0与字符'0'的数据，需要对他们进行转换，合并为一个类别：



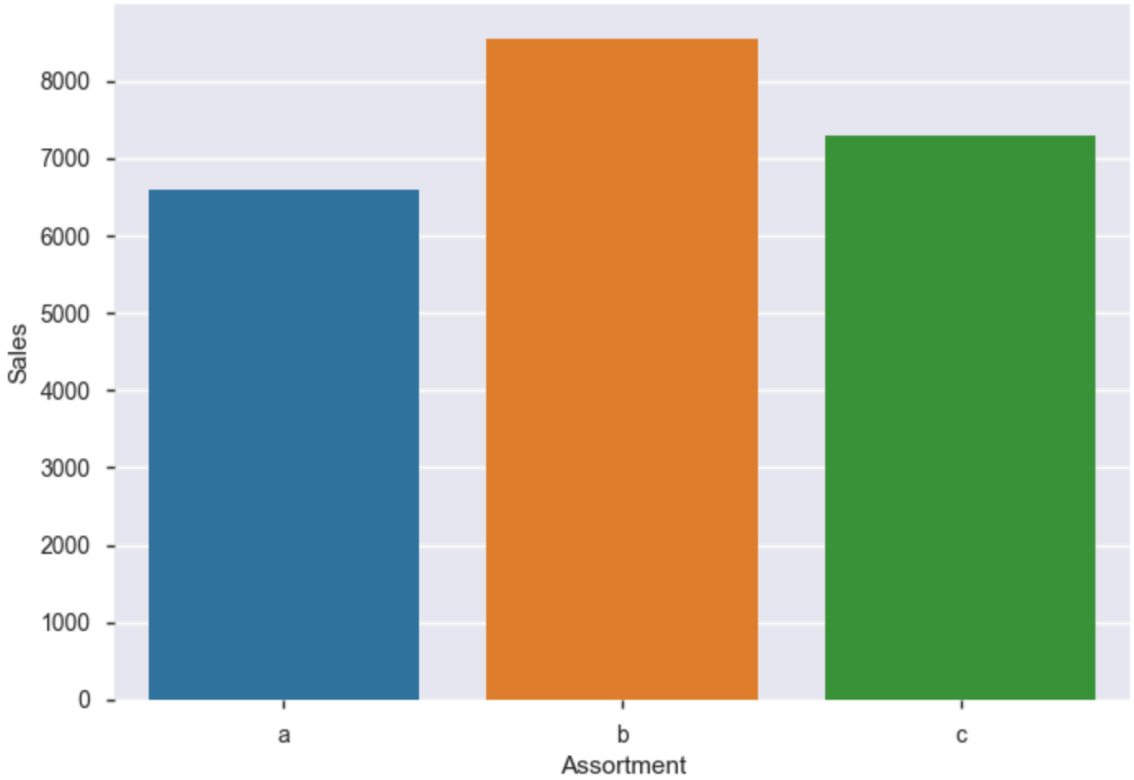
- SchoolHoliday与销售额的关系并不明显，是学校假日的药店平均销售额比非学校假日的高一点：



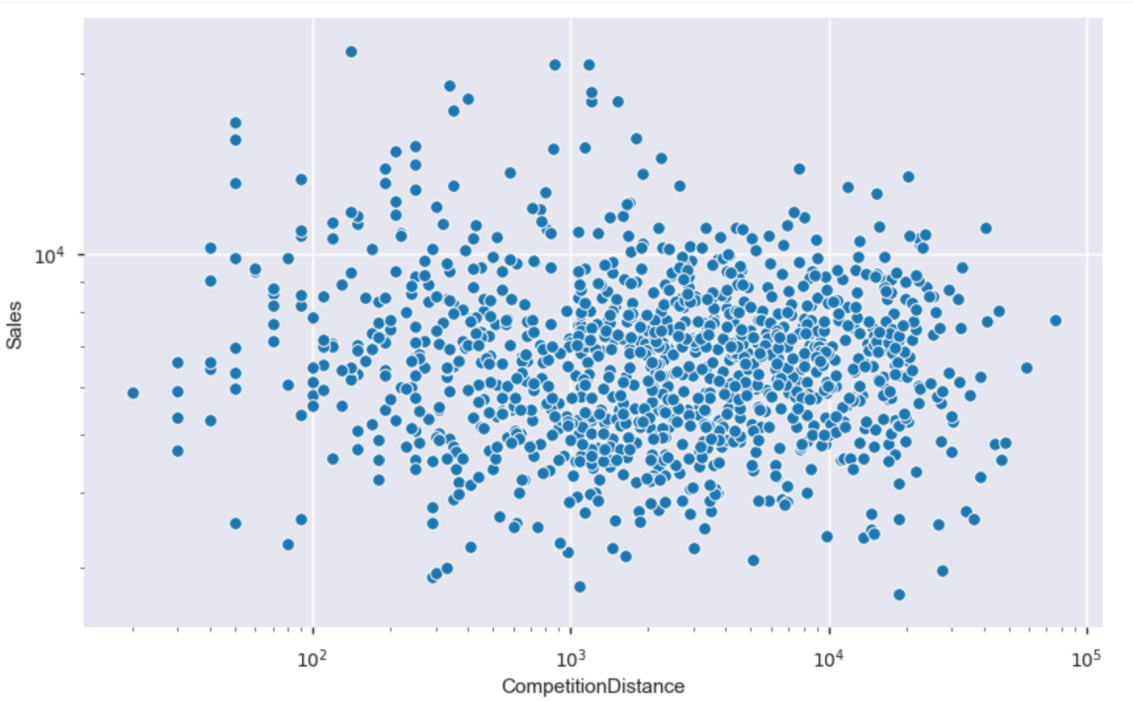
- StoreType一共有四种，统计每种类型的药店的平均日销售额，可发现storeType为b的药店日平均销售额最高：



- Assortment一共有三类，其中b类型对应的日均销售额最高，其次是c：



- 看竞争对手的距离和药店日平均销售额的关系，粗略可看出竞争对手的距离越近，药店的日均销售额较低，竞争对手距离越远，日均销售额高：



算法和技术

这个项目需要对每个药店未来6周的销售额进行预测，属于一个回归问题。鉴于前面的特征分析，训练数据中的特征包括数值特征，同时也包含大量的Categorical特征，部分特征可进行特征组合，因此，可以利用的算法包括线性回归，树模型等算法进行拟合，常见的树模型包括随机森林，GBDT，xgboost等。这里对线性回归模型以及xgboost模型进行说明：

线性回归模型，是通过一个多维的线性函数，对一组给定的训练数据，包括多个特征变量，以及一个预测变量进行拟合，在这里是指通过训练数据里的门店自身的特征以及门店的历史销售特征作为特征变量，门店的实际销售额作为待预测的变量，通过各个特征的一个线性函数进行拟合，使得样本中的实际销售额与线性函数预测的销售额尽可能接近，即所有样本的预测销售额与实际销售的均方误差和最小。线性回归模型的优势在于结果非常容易解释，每个特征的权重代表特征对结果的影响大小，权重越高的特征，对结果的影响力最大，因此往往通过一些特征工程的手段，人为制造更多的强特征，使得模型得到优化；而线性回归模型的缺点就是模型只能通过线性方式拟合样本，而往往结果与特征之间并非是线性关系，因此无法用线性回归模型进行有效的拟合。

xgboost模型，是集成模型的一种，通过多个弱的树模型集成而得，每个弱模型的训练是在前一次训练的结果基础上，进一步优化模型预估与实际值之间的残差训练而得，而xgboost的优势在于解决了前面线性回归模型仅限于线性拟合的问题，同时xgboost优化了训练算法，可并行训练，加快模型训练速度。

xgboost的loss function定义为:

Obj=\sum { i=1 }^{ n }{ l({ y }{ i }, } { \hat { y } }{ i })+\sum _{ k=1 }^{ K }{ \Omega (f{ k }) }

前面一部分表示模型的训练误差，后面部分通过增加正则项控制模型的泛化误差，正则化项包含两部分：

\Omega (f)=\gamma T+\frac { 1 }{ 2 } \lambda { \parallel w\parallel }^{ 2 }

T表示叶子结点的个数，w表示叶子节点的分数。γ可以控制叶子结点的个数，λ可以控制叶子节点的分数不会过大，防止过拟合。

模型的训练过程是增量学习的，通过固定前t-1颗数据的学习结果，每次增加一颗树，进行训练。对于树的分裂过程，xgboost使用了和CART回归树一样的想法，利用贪婪算法，遍历所有特征的所有特征划分点，不同的是使用前面这个目标函数值作为评价函数，看分裂后的目标函数值比单子叶子节点的目标函数值的增益，同时为了限制树的深度过大，还加了个阈值，只有当增益大于该阈值才进行分裂。

因此，同样是BoostingTree，相比GBDT，xgboost在loss function上增加了正则化项，防止树的过拟合，同时对loss function进行了优化，使用了泰勒二阶展开，使得xgboost可以支持任何一个二阶可导的函数作为损失函数，也就是自定义损失函数，比如这个项目中采用的RMSPE；另外，在算法实现层面，xgboost实现了并行化训练，使得训练速度更快。

基准模型

从前面的数据探索部分，我们得知与销售额有强相关的特征有：DayOfWeek，Promo，Open三个特征，因此我们可通过对训练样本的每个门店历史销售额，分别计算每个门店对应每个DayOfWeek，Promo，Open值对应的中位数销售额，作为我们的基准模型数据。

将基准模型提交到Kaggle上，public score和private score如下：

Public Score	Private Score
0.14065	0.15205

III. 方法

数据预处理

数据类型转换：StateHoliday的值既有数字类型，也有字符类型，这里统一处理为字符类型。

数据空值填充：对CompetitionDistance采用0填充空值，CompetitionOpenSinceYear和Promo2SinceYear用1990填充，CompetitionOpenSinceMonth和Promo2SinceWeek用1填充，Open用1填充。

特征抽取，将特征分为门店基础特征，门店历史销售特征，以及门店的销售日期的事件特征，事件指StateHoliday，SchoolHoliday，Promo这些：

- 门店Id: Store
- 门店类型：StoreType，将值转化为categorical的数字类型
- 门店售卖商品类型：Assortment，将值转化为categorical的数字类型
- 门店是否有进行促销: Promo
- 该天的节假日类型：StateHoliday，0表示非节假日，将值转化为categorical的数字类型
- 该天是否是学校假日：SchoolHoliday
- 销售日期所在年份：Year
- 销售日期所在月份：Month
- 销售日期所在月份的第几天：DayOfMonth
- 销售日期所在一年的第几周：WeekOfYear
- 销售日期所在一年的低几天：DayOfYear
- 竞争对手出现的年份：CompetitionOpenSinceYear
- 销售日期在出现竞争对手之后的第几个月：MonthAfterCompetition

- 销售日期在出现竞争对手之后的天数：DaysAfterCompetition
- 销售日期在持续促销开始后的天数：DaysAfterPromo2
- 销售日期在持续促销开始后的周数：WeekAfterPromo2
- 销售日期距离上次StateHoliday的间隔天数：daysAfterStateHoliday
- 销售日期距离上次SchoolHoliday的间隔天数：daysAfterSchoolHoliday
- 销售日期距离上次促销的间隔天数：daysAfterPromo
- 销售日期最近7天中StateHoliday的天数：last_7d_state_holiday
- 销售日期最近7天中SchoolHoliday的天数：last_7d_school_holiday
- 销售日期最近7天中Promo的天数：last_7d_promo
- 销售日期所在月份是否是持续促销的月份：isPromoMonth
- 门店平均日营业销售：avg_sales_store
- 门店平均日营业顾客数：avg_customers_store
- 门店平均每个顾客人均消费：avg_sales_customer_store
- 门店平均（中值）日均顾客数：median_customers_store
- 门店一周各天的平均日销售：avg_sales_store_dow
- 门店一周各天的平均（中值）日销售：median_sales_store_dow
- 门店一周各天的平均顾客数：avg_customers_store_dow
- 门店一周各天的平均（中值）顾客数：median_customers_store_dow

执行过程

训练集和校验集的选取，这里校验集选取了训练集中最后的6周数据，保持和测试集的时间长度一致。

这里的特征非常多，有些特征从业务含义上看是存在重复的，比如平均值和中位数，对于特征的选择，一开始通过对全部特征都放入xgboost训练，发现结果并不理想，提交到Kaggle上的public score和private score还不如基准模型。所以这里遇到的问题是特征数量太多，而且存在一些业务意义上重复的特征，需要对这些特征进行有效的筛选。

完善

针对前面遇到的特征筛选的问题，通过对不同类别的特征进行随机筛选组合，通过对校验集上的rmspe得分进行筛选。我将特征分为三组：

- 基础特征：通过前面训练结果的features importance，发现有几个特征权重很高，因此放到基础特征里，需要再随机抽取，这里包含三个：Store, DayOfWeek, CompetitionDistance
- 销售相关特征：avg_sales_store, avg_customers_store, avg_sales_customer_store, median_customers_store, avg_sales_store_dow, median_sales_store_dow, avg_customers_store_dow, median_customers_store_dow, StoreType, Assortment, CompetitionOpenSinceYear，考虑到这其中有一些特征在业务含义上是重复的，因此特征筛选时每次会从这里随机抽取3-6个
- 事件相关特征：Year,Month,DayOfMonth,WeekOfYear,DayOfYear,DaysAfterCompetitionOpen, MonthAfterCompetitionOpen, DaysAfterPromo2, WeekAfterPromo2, daysAfterStateHoliday, daysAfterSchoolHoliday, daysAfterPromo, last_7d_state_holiday, last_7d_school_holiday, last_7d_promo, isPromoMonth，筛选时每次从这些里随机抽取5到12个

构建100个模型（都是xgboost模型，超参数也是一致的，区别仅在于不同模型采用的特征不同）进行训练，通过在校验集上选取表现得分最高的特征组合：Store, DayOfWeek, CompetitionDistance, median_customers_store, avg_sales_customer_store, avg_customers_store_dow, median_sales_store_dow, avg_sales_store, CompetitionOpenSinceYear, last_7d_promo, WeekAfterPromo2, DaysAfterCompetitionOpen, Year, last_7d_school_holiday, DayOfYear, daysAfterPromo, WeekOfYear，在校验集的得分（rmspe）为0.11305，同时将这个模型对测试集的预测结果提交到Kaggle，得分如下：

Public Score	Private Score
0.11409	0.12012

然后在这个特征组合的基础上，再通过尝试多种不同的xgboost超参数组合，对模型进行调优，主要调整的超参数有：

- max_depth：尝试10，11，12三种不同深度，深度越深，拟合能力越强，但容易过拟合，所以同时还要对下面两个参数进行调整
- subsample：尝试0.7，0.8，0.9，采样越少，对过拟合的抵抗越强
- colsample_bytree：尝试0.5，0.6，0.7，同样，也是采样越少，对过拟合的抵抗越强

最终通过不同超参数的交叉组合训练，选取校验集得分最高的超参数组合为最终模型，校验集得分为0.11096，对应的超参数为：

```
{
    'objective': 'reg:linear',
    'eta': 0.03,
    'max_depth': 12,
    'colsample_bytree': 0.6,
    'subsample': 0.7
}
```

最终以这个模型对测试集进行预测，结果提交到Kaggle的得分如下：

Public Score	Private Score
0.10984	0.11687

达到项目要求的0.11773的要求。

IV. 结果

模型的评价与验证

从整个过程来看，最初是直接将大量的特征作为训练特征，放入xgboost训练，发现模型过于复杂，导致最后的结果并不理想，后续采用构建100个模型，对不同的特征随机抽取组合，通过验证集对各种特征组合模型进行筛选，在验证集的结果上看是进步了很多的，训练集得分0.09463和验证集的得分0.11304差距并不大，同时提交到Kaggle上也比基准模型的得分好一些，从而进一步在筛选出最优的特征后，对模型的超参数进行调参，使得模型在训练集和验证集上进一步提升，优化后的训练集得

分0.08308和验证集得分0.11252，相比调参前的结果又有一定提升，提交到Kaggle后，得分也比之前的要更好。因此，最终训练出来的模型通过训练集，验证集，测试集都能得到验证，效果是最好的，说明模型对不同数据的泛化能力是很好的。

合理性分析

最终的模型和基准模型效果对比如下：

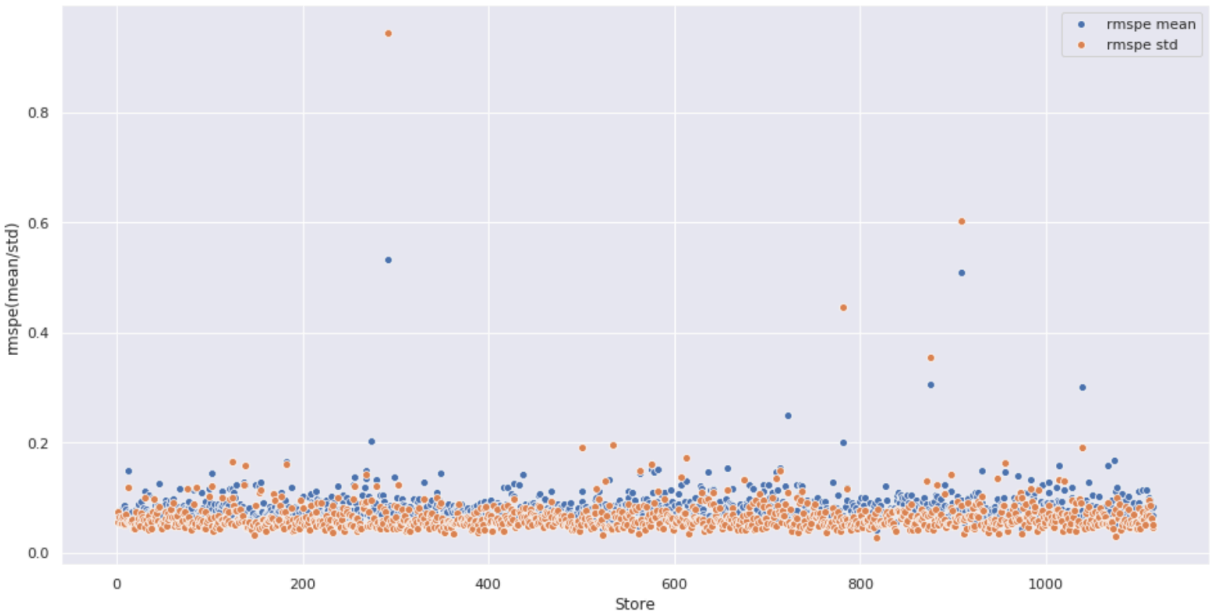
	Public Score	Private Score
基准模型	0.14065	0.15205
最终模型	0.10984	0.11687

从得分上看，最终模型整体得分都优于基准模型得分，说明最终模型确实是比基准模型效果更好的，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。

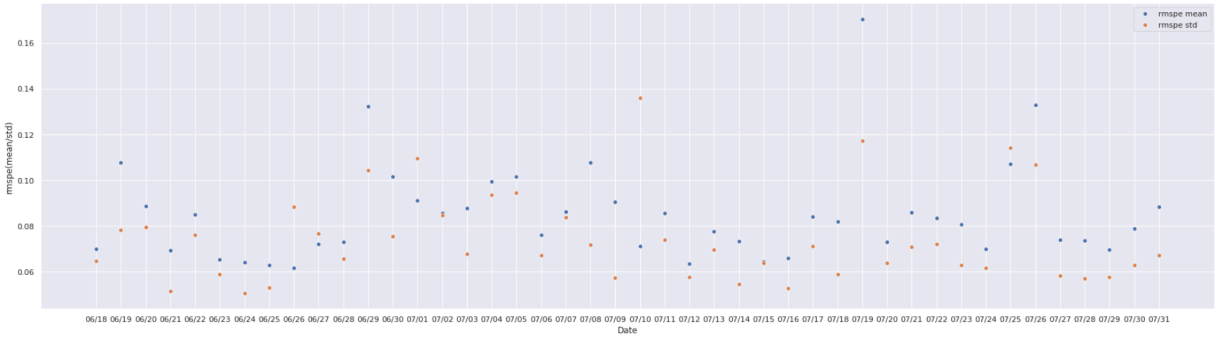
V. 项目结论

结果可视化

校验集的数据包含了6周内的所有门店销售数据，因此可以通过校验集的预测销售额和实际销售额进行模型结果的可视化验证。分别从门店维度和日期维度对比实际销售额和预测销售额的rmspe，分别取均值及方差：

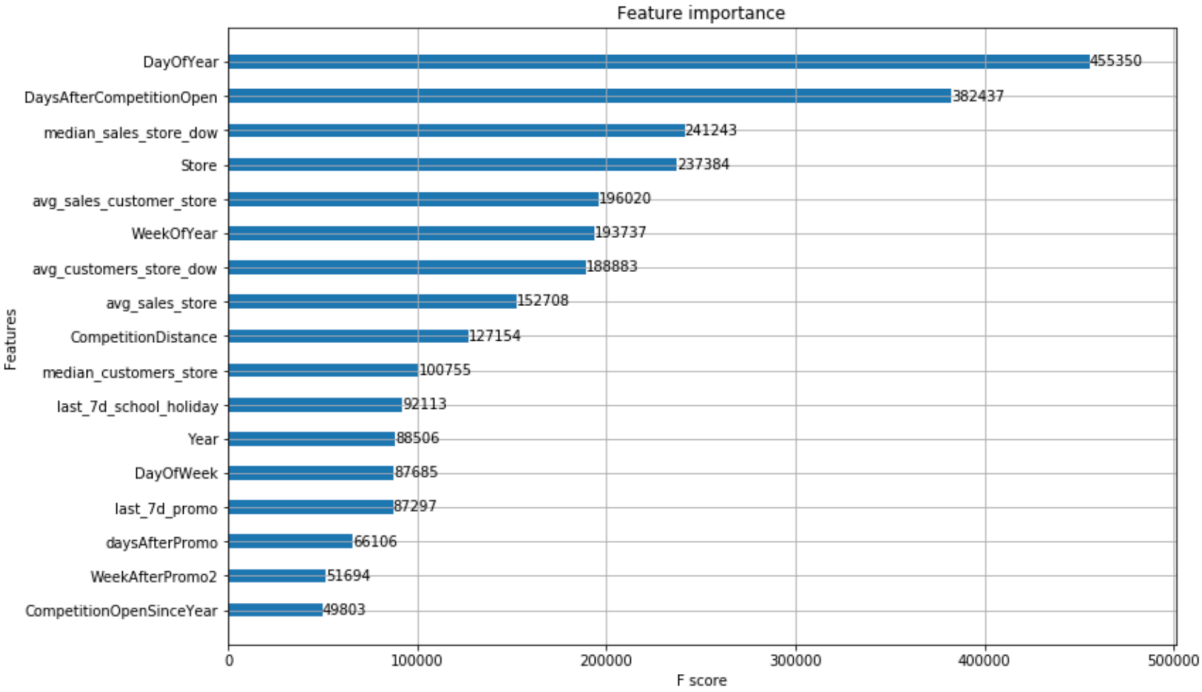


上图是门店维度看各个门店的平均rmspe和方差，可以发现绝大部分门店的rmspe平均值和方差都非常小，说明这些门店的预估销售额是非常准确的，但也发现少数几个门店的rmspe平均值在0.2以上，而且方差也比较大，说明模型对这几个少数门店的销售预测不是很准确，也有一些门店是rmspe平均值在0.1附近，但方差比较高，说明模型对这些门店的某几天的销售预测不准确。



上图是从日期维度对每天所有门店的rmspe取平均值和方差，能看出来大部分时间的rmspe均值和方差都很低，说明这些日期的销售预测数据是比较准确的，而有少数几个时间，比如07-19这天的数据，rmspe均值和方差都比较高，说明模型在这天的销售预测不太准确。

另外，对最终模型选取的特征，每个特征的重要度如下图所示：



从上图可以看出，销售日期，竞争对手和门店的历史销售特征对销售预测都非常重要，而其中时间维度的特征对销售预测非常重要。从前面一张图，通过日期维度对每天所有门店的销售预估的误差数据可看出，不同日期的误差确实是有些不同的，说明对时间维度可以增加更多的特征维度（例如季节性特征等），让模型能更好的拟合各个不同时间点的销售预测。

对项目的思考

项目首先通过数据探索，发现和销售强相关的几个重要特征，由此构造基准模型。之后对特征的业务理解，构造历史销售相关的特征，以及事件相关的特征，有了这些特征后，遇到一个棘手的问题就是如何有效的筛选特征，一开始直接将全部特征放入xgboost模型中训练，发现效果并不好，原因可能是特征太多，模型也变得复杂，后面通过工程化的方式进行特征筛选，随机筛选一些特征，每组都作为一个模型，去训练，通过校验集统一校验的方式，选出得分最高的特征筛选作为改进后的模型，在验证集和测试集上都得以验证，确实效果有提升。因此，采用同样的方式，在此改进的模型基础上，进一步对模型的超参数进行调优，也是通过验证集的校验，选取得分最高的超参数组合，最终得到一个满意的结果。

需要作出的改进

目前看到可以有两个方面进一步改进：

- 抽取更多的特征：门店最近1个月，3个月，6个月的历史销售特征等，看看对模型的效果是否有提升；
- 尝试深度模型，通过构建深度模型对特征进行有效的提取，降低特征工程的复杂性，也是可以尝试的方向，比如第三名的选手提出embedding的思路构建深度模型。

参考

- Kaggle比赛第一名的经验分享
- 一文读懂机器学习大杀器XGBoost原理
- Introduction to Boosted Trees
- XGBoost原理解析
- xgboost是用二阶泰勒展开的优势在哪