机器学习纳米学位

焦龙

2019年7月7日

## I. 问题的定义

### 项目概述

本项目是根据商店历史数据进行销售额的预测，该情景非常适合使用监督学习中的回归方法来进行预测。本项目共有train.csv、store.csv、test.csv三种数据集，分别表示训练数据、商店数据、测试数据。通过以上数据的训练，来对未来6周的销售额进行预测。

### 问题陈述

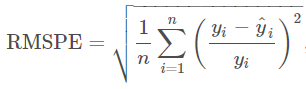
问题定义：使用训练数据、商店数据进行模型训练，通过测试数据预测未来6周的所有商店的销售情况。

任务大纲：1.读取数据，2.数据分析及预处理，3.特征工程，4.模型训练，5.对比基准阈值，6.模型优化，7.测试结果提交Kaggle

希望结果：测试集rmpse小于0.11773

### 评价指标

使用kaggle指定的RMSPE(均方根百分比误差)指标评价方法，公式如下：



该指标表示的是真实值与预测值误差同真实值的百分比，比值越小，说明误差越小，同时该公式避免了出现负数的情况，因此该指标计算方法是合理的。

## II. 分析

### 数据的探索

本项目使用的训练数据集为train.csv、store.csv，其中train字段特征信息如下：

Store - 每家商店的唯一ID。

DayOfWeek - 表示星期几。

Date - 表示日期，由年-月-日组成。

Sales - 任何一天的营业额（这是您预测的）。

Customers- 特定日期的客户数量。

Open - 商店是否营业的指标：0 =关闭，1 =打开。

Promo- 指示商店当天是否正在运营促销。

StateHoliday - 表示州假日。请注意，所有学校在公众假期和周末都关闭。a =公共假日，b =复活节假期，c =圣诞节，0 =无。

SchoolHoliday - 表示学校假期。商店是否受到公立学校关闭的影响。

store字段特征信息如下：

Store- 每家商店的唯一ID。

StoreType - 区分4种不同的商店模型：a，b，c，d。

Assortment- 描述分类级别：a =基本，b =额外，c =扩展。

CompetitionDistance - 距离最近的竞争对手商店的距离（以米为单位）。

CompetitionOpenSince [Month/Year] - 给出最接近的竞争对手开放时间的大致年份和月份。

Promo2 - 商店是否参与连续促销：0 =商店不参与，1 =商店参与。

Promo2Since [年/周] - 描述商店开始参与Promo2的年份和日历周。

PromoInterval - 描述Promo2启动的连续间隔，命名重新开始促销的月份。例如，“2月，5月，8月，11月”意味着每一轮在该商店的任何给定年份的2月，5月，8月和11月开始。

train.csv字段统计信息

RangeIndex: 1017209 entries, 0 to 1017208

Data columns (total 9 columns):

Store 1017209 non-null int64

DayOfWeek 1017209 non-null int64

Date 1017209 non-null datetime64[ns]

Sales 1017209 non-null int64

Customers 1017209 non-null int64

Open 1017209 non-null int64

Promo 1017209 non-null int64

StateHoliday 1017209 non-null object

SchoolHoliday 1017209 non-null int64

store.csv字段统计信息

RangeIndex: 1115 entries, 0 to 1114

Data columns (total 10 columns):

Store 1115 non-null int64

StoreType 1115 non-null object

Assortment 1115 non-null object

CompetitionDistance 1112 non-null float64

CompetitionOpenSinceMonth 761 non-null float64

CompetitionOpenSinceYear 761 non-null float64

Promo2 1115 non-null int64

Promo2SinceWeek 571 non-null float64

Promo2SinceYear 571 non-null float64

PromoInterval 571 non-null object

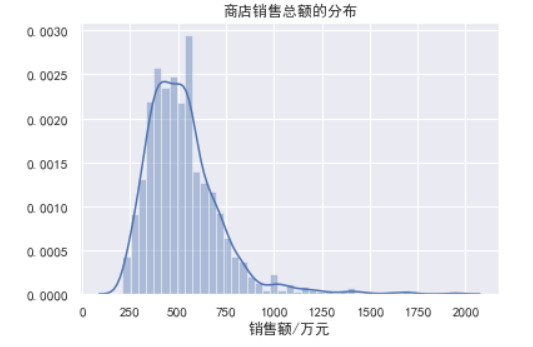
通过数据的对比发现train数据与store数据是通过Store字段进行关联的，store数据是train数据的一个扩充，同时我们发现train数据没有字段为空的数据。

Store数据中含有空数据的字段如：CompetitionDistance、CompetitionOpenSinceMonth、CompetitionOpenSinceYear，这些需要竞争对手信息可能需要进行填充，最稳妥的办法是添加数据的中值。而store数据以Promo开头的字段无须对空值进行填充，因为这些字段表示的是连续促销的信息。

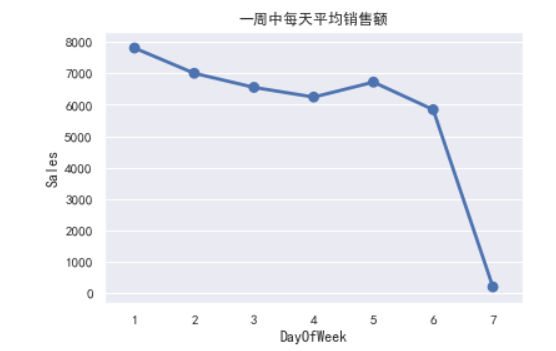
另外还需要检查字段中是否有一个值多种写法情况，如有重复需要对字段进行处理。

### 探索性可视化

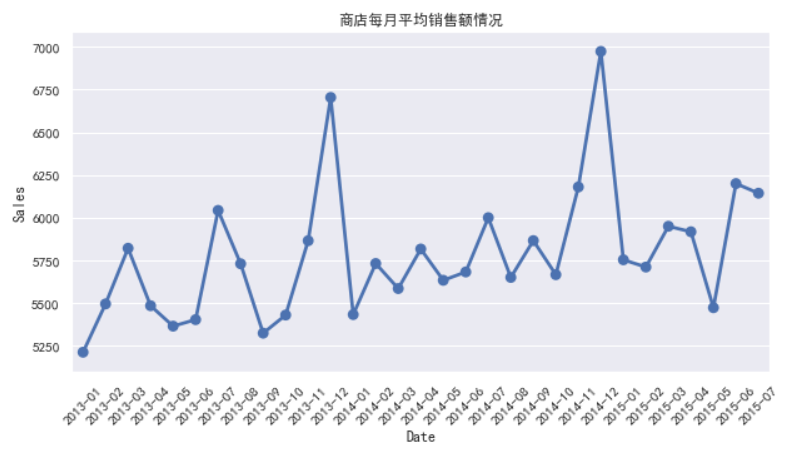
对于数据的探索可按销售额分布、时间、促销、节假日维度进行分析，对于销售额预测问题，先从销售额的分布情况开始分析，如下图：



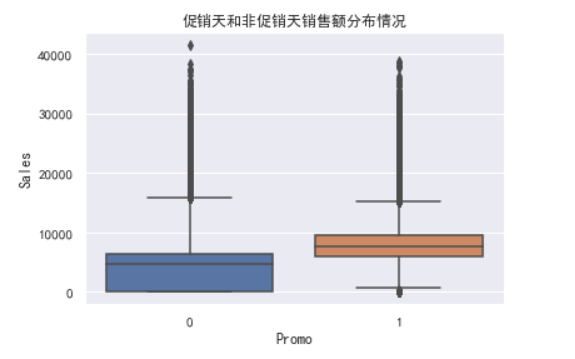
通过观察发现，销售额属于正偏态分布，大部分商店销售额分布在500万附近，再来看一周中每天平均销售情况，统计如下：



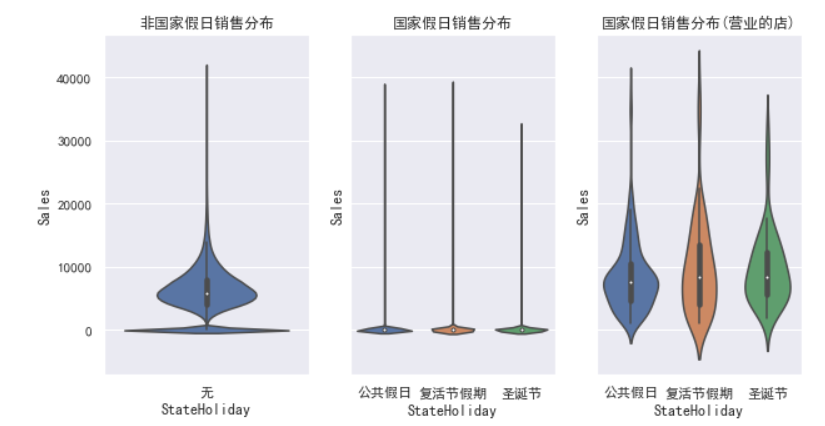
通过观察，发现周日(7)的销售额几乎为0，说明商店在周日是不营业的。再来看一下销售额与月份之间的关系：

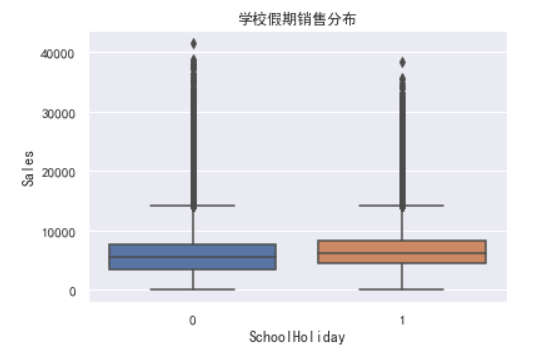


通过观察，明显发现在每年的10月至12月是销售额的最高的时期，说明这段时期是销售旺季，同时也说明这个行业的销售是有周期性的，对我们预测很有帮助。再来看一下促销对销售额的影响：



从上图可知，促销对销售额影响非常有限，没有明显的差别，说明这个因素对预测销售额作用不大，再来看一下节假日对销售额的影响：



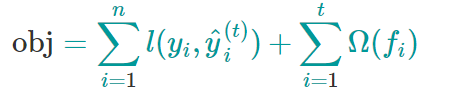
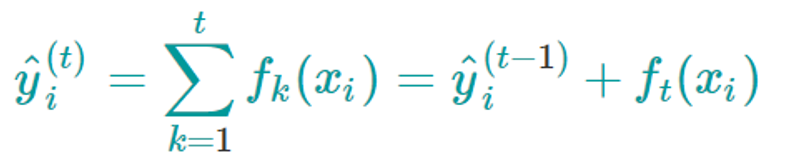


通过分析 国家假日与学校假期，发现这些因素对销售额的预测关系不大，没有必然关联，因此这些因素可以忽略。

### 算法和技术

俗话说“人多力量大”， 其实与XGBoost算法的原理差不多，该算法有多棵树组成。每次添加的新树其实就是拟合上次预测的残差的新函数。当训练完成预测一个样本时，其实只是根据样本的特征，会落到每棵树对应的一个叶子节点，每个叶子节点就对应一个分数，最后只需要将每棵树对应的分数加起来，这该值就是样本的预测值。数学表达式如下；

目标函数 新树预测值

**L**表示损失函数，这里其实就是新树与上次拟合的残差进行预测。

XGBoost优点：

XGBoost是boosting迭代型树类算法，适用于分类、回归问题。好处是速度快、效果好、能处理大规模数据、支持自定义损失函数等。

GBDT和XGBoost的区别为：

**1.分类器的选择：**GBDT以CART作为基分类器，而XGBoost可指定损失函数(如最小平方差、或定义损失函数)，可增强模型的泛化能力。

**2.二阶泰勒展开：**GBDT优化时只用到一阶导数，而XGBoost则对损失函数进行二阶泰勒展开，也就是对损失函数进行二阶导数。

**3.列抽样：**XGBoost借鉴了随机森林的做法，支持列抽样，不仅能降低过拟合，还能减少计算。

**4.复杂度控制：**XGBoost在目标函数里加入了正则项，用于控制模型的复杂度。正则项里包含了树的叶子节点个数、每个叶子节点上输出分数的L2模的平方和。可以有效防止模型的过拟合。

**5.支持并行化：**虽然树与树之间是串行关系，但可以在“特征”粒度上实现并行。在训练之前，已先对数据进行了排序，然后将特征保存为 block(块)结构。当在进行节点的分裂时，可并行计算各个特征的增益，并最终选择增益最大的特征去做分裂。

模型超参数：

params = {"objective": "reg:linear",

"booster": "gbtree",

"eta": 0.02,

"max\_depth": 10,

"subsample": 0.9,

"colsample\_bytree": 0.7,

"silent": 1,

"seed": 78

}

num\_boost\_round = 5000

1、objective

选在最小化的损失函数：reg:linear

2、booster

选择迭代的模型，gbtree：基于树的模型

3、eta

类似于learning rate，通过减少每一步的权重，可以提高模型的鲁棒性。

典型值为0.01-0.2。

4、max\_depth

树的最大深度，用来避免过拟合的。max\_depth越大，模型会学到更具体的更局部样本。

典型值：3-10

5、subsample

控制对于每棵树随机采样的比例。减小参数的值，算法会更加保守，避免过拟合。如果值设置过小，可能会导致欠拟合。

典型值：0.5-1

6、colsample\_bytree

控制每棵随机采样的列数的占比(每一列是一个特征)。

典型值：0.5-1

7、num\_boost\_round

设置为5000，表示每次的训练次数。在训练中发现在3841次时模型达到最优。

8、early\_stopping\_rounds

设置为100，表示在 100次模型都没有提升的情况下，会终止模型，从而避免模型过拟合。

### 基准模型

项目要求的基准阈值RMSPE是小于0.11773。

## III. 方法

### 数据预处理

Test数据

发现只有Open字段出现空值，而且只有编号为622这一家店，且周一至周六应该是营业的，而且也不是节假日，因此将Open字段填充为1。

Store数据：

1.竞争对手信息出现空值，由于缺失数量较多，在加上缺失原因不明，因此用0来填充比较可靠。

2.店铺促销信息的缺失是因为没有参加促销活动，所以用0填充

3. StateHoliday字段中的‘0’字符转换为0

Store分别与 train、test进行数据合并，同时将Date字段转换为日期格式。

### 执行过程

模型执行过程如下：

1.模型构建：指定模型参数、迭代次数以及将训练数据转换DMatrix格式。

2.模型训练：通过train方法进 行模型训练，返回一个Booster对象。

3.使用 Booster的 predict进行验证集上的预测，并通过 RMSPE方法计算分数。

4.模型优化：

5.提交 kaggle。

主要困难来自对xgboost的不熟悉，通过查看官方文档可以基本满足使用要求。另外就是对模型的优化，由于每次训练时间很长，因此调优非常耗时间，不过这也加深了对模型的了解。

914商店销售额预测



Sales：实际销售额

Pred\_store是模型最终预测的销售额。

### 完善

通过系数校正方法，提高模型的准去率。实现过程是对每家商店进行误差计算，在其中选取最小的误差权重，从而提升没交商店的预测准确率。

## IV. 结果

### 模型的评价与验证

通过使用XGBoost模型对数据训练，最终模型的RMSPE为0.10722,达到了基准模型RMSPE:0.11773的要求，因此模型的选择合理的，与期待结果一致。

训练数据中训练集和验证集是按时间前后划分，即取最后6周为验证集数据，由于销售存在周期性，既利用周期性特点对时间上的前后关系训练数据、验证数据、测试数据进行很好的预测。因此模型预测是合理可靠的。

### 合理性分析

项目要求的基准模型0.11773，而模型优化后最佳情况得分：0.10722，通过对比，我们训练的模型比基准模型表现的更好。

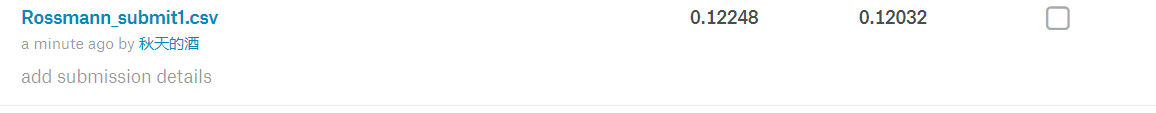
事实证明我们的模型相比较基准阈值确实有很大提升，因此我们的模型确实解决了问题，或者说一定程度的解决了问题。

## V. 项目结论

### 结果可视化

基准模型的：0.11773。

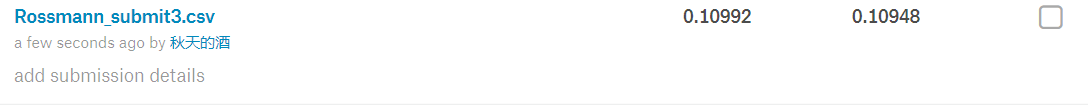
模型训练后表现



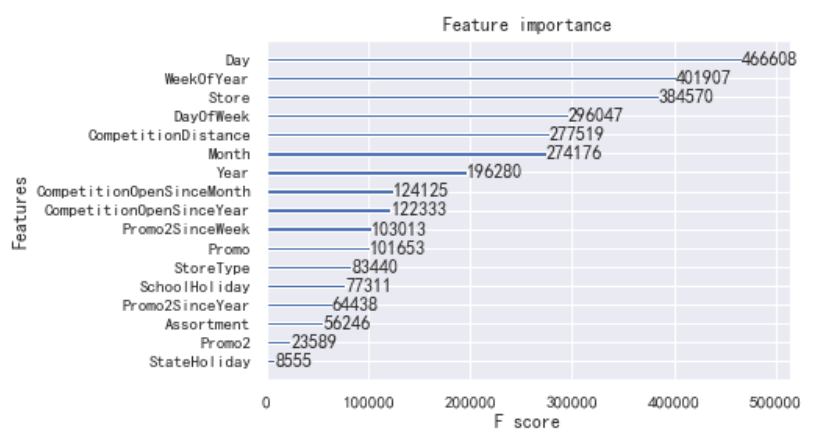
统一偏差校验后的分数



分组偏差校验后的分数



模型重要特征如下



### 对项目的思考

项目的整体流程：

1. 读取kaggle指定的数据集。

2. 对数据中的异常，无用数据进行清洗。

3. 将trian和test分别与store数据合并

4. 特征工程：枚举字段映射为统一形式数字，对日期字段提取新的特征。

6. 指定基准阈值。

7. XGBoost模型构建、训练、评分。

8. 校正系数。

9. 将测试数据预测结果上传至kaggle。

在数据分析中发生了一些有意思的发现，比如从主观上感觉节假日是对销售有很大贡献的，但从分析上来看，是没有多大作用的。另外促销也是一样对销售额没有多大的提升。反而发现销售是存在周期性，每到年底都是销售的高峰，这颠覆了我对销售的预测的一些认知。

困难来自对xgboost模型的学习，库的使用，和调优，由于模型训练时间长，因此调优非常困难。

从训练结果来看，我是很满意的，毕竟达到了要求，还是很高兴的。

### 需要作出的改进

1. 模型方面还可以尝试lightgbm，这也是一个非常优秀的boosting类模型,也非常适合进行预测。
2. 由于数据中大部分都与时间有关，因此可以数据中加入额外的信息，例如如药店在各州的天气、降雨量、气温数据，这些数据也可以帮助我们进行预测。
3. 可尝试使用Entity Embedding神经网络模型进行销售预测。
4. 在数据特征上，还可以在进一步进行挖掘。

### 参考文献

【1】XGBoost官方文档：https://xgboost.readthedocs.io/en/latest/parameter.html

【2】Pandas官方文档：http://pandas.pydata.org/pandas-docs/stable/api.html

【3】Kaggle项目：<https://www.kaggle.com/c/rossmann-store-sales>

【4】XGBoost介绍：<http://blog.itpub.net/31542119/viewspace-2199549/>

【5】XGBoost算法：<https://blog.csdn.net/qq_20412595/article/details/82621744>