机器学习工程师毕业项目

2020 年 8 月 4 日

王泓霏

1 定义

1.1 项目概述

这是一个 Kaggle 竞赛项目，目标是训练一个模型，来预测 Rossmann 零售店的销售额情况，是一个典型的回归问题。
Rossmann 成立于 1972 年，是德国最大的日化用品超市，在 7 个欧洲国家有 3000 多家药店。商店不定时会举办短期的促销活动以及连续的促销活动以此来提高销售额。除此之外，商店的销售还受到许多因素的影响，包括促销、竞争、学校和国家假日、季节性和周期性。
我选择这个问题，一是因为它具有挑战性，可以提升自己的专业能力，二是我本人对于数据挖掘部分很感兴趣，渴望在这部分得到更多的锻炼，为接下来的专业深造做铺垫。
其中，药妆店的信息，在 kaggle 项目“Forecast Rossmann Store Sales”中提供。
本项目使用的模型是 xgboost。
项目选择的数据集是 Kaggle 竞赛提供的数据，训练集包括 1017209 个样本，共 9 个特征，测试集包括 41088 个样本。对于，模型需要预测出是未来一段时间内，零售店的销售额情况。

1.2 问题陈述

Kaggle 竞赛提供的数据是 Rossmann 零售店过去实际的销售额情况，因此数据有大量的缺失值，这些都增加了问题难度。
这是数据挖掘中的回归问题，首先对于数据进行大致探索，关于此药妆店的开店情况，竞争对手的情况，进行可视化的展示。
目标是根据 Rossmann 药妆店的信息（比如促销，竞争对手，节假日）以及在过去的销售情况，来预测 Rossmann 未来的销售额。理想情况是，能够通过现有的数据集中药妆店的特征，较准确的地分析出未来的销售额的情况。

1.3 评价指标

y_i 是一个店在一天内的销售额，而 y 均是预测的平均值。
我选择与本次 kaggle 项目“Forecast Rossmann Store Sales”相同的的评价指标，即均方误差，即 Root Mean Square Percentage Error (RMSPE). 由于 RMSPE 对一组预测中的特大或特小误差反映非常敏感，因而它能够很好地反映出模型预测的精密度。对于需要预测的销售量，用 RMSPE 能够比较不错地表示模型的效果。
得出来的值越小，越接近真实值，模型的效果也就越好。最小值是 0，也就是预测结果完全与真实值重合。

$$\text{预测值: } \hat{\mathbf{y}} = \{\hat{y_1}, \hat{y_2}, \ldots, \hat{y_n}\}$$
$$\text{真实值: } \mathbf{y} = \{y_1, y_2, \ldots, y_n\}$$

图 1 真实值与预测值

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

图 2 RMSPE 评价指标

2 分析

2.1 数据可视化

2.1.1 店铺销量

竞赛数据集有两个子集，一个训练数据集，一个测试数据集

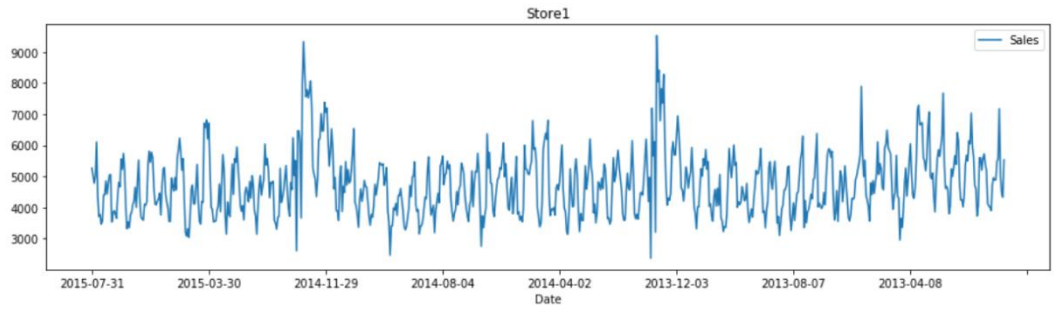
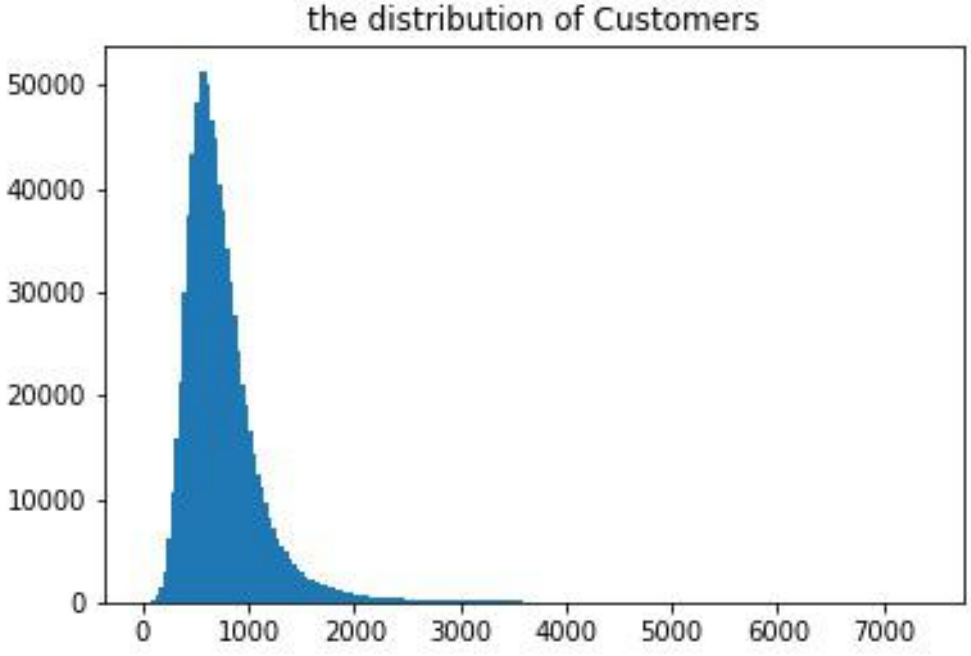
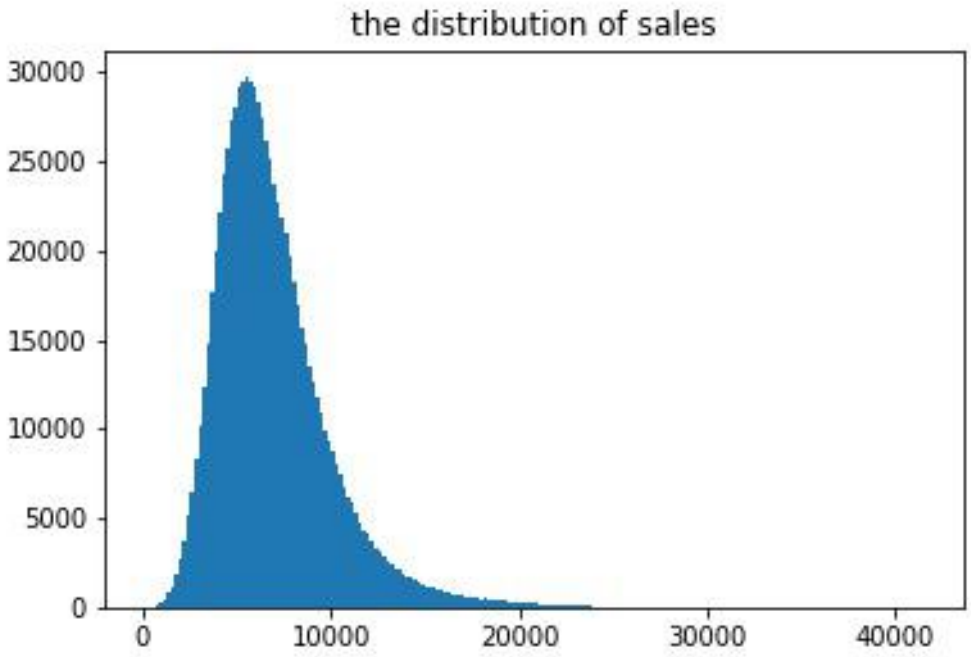


图 1 店铺销量随时间的变化

店铺的销售额是有周期性变化的，一年中 11,12 月份销量相对较高，可能是季节因素或者促销等原因 此外从 2014 年 6-9 月份的销量来看，6,7 月份的销售趋势与 8,9 月份类似，而我们需要预测的 6 周在 2015 年 8,9 月份，因此我们可以把 2015 年 6,7 月份最近 6 周的 1115 家店的数据留出作为测试数据，用于模型的优化和验证

2.2.2 顾客和销量之间的相关性





由图可知，顾客数量和销量的之间有很强的相关性，顾客数量越多，销量越高，尤其在顾客数量更多的时候，相关性更加明显。

2.2 算法和技术

2.2.1 xgboost

XGB 也是一种 gradient boosting 的方法，基于决策树的集成机器学习算法，XGBoost 是由 GBDT 发展而来。XGBoost 可以使用 Regression Tree（CART）作为基学习器，也可以使用线性分类器作为基学习器。以 CART 作为基学习器时，其决策规则和决策树是一样的，但 CART 的每一个叶节点具有一个权重，也就是叶节点的得分或者说是叶节点的预测值。
和其他算法相比，XGBoost 算法的不同之处有以下几点：
应用范围广泛：该算法可以解决回归、分类、排序以及用户自定义的预测问题；
可移植性：该算法可以在 Windows、Linux 和 OS X 上流畅地运行；
语言：支持包括 C++、Python、R、Java、Scala 和 Julia 在内的几乎所有主流编程语言；
云集成：支持 AWS、Azure 和 Yarn 集群，也可以很好地配合 Flink、 Spark 等其他生态系统。

在此项目中将应用 xgboost 作为主要模型,来进行数据集的预测。

参数：
params 这是一个字典，里面包含着训练中的参数关键字和对应的值，形式是 params = {‘booster’:’gbtree’,’eta’:0.1}
silent [default=0]:设置成 1 则没有运行信息输出，最好是设置为 0.
nthread [default to maximum number of threads available if not set]：线程数
dtrain 训练的数据

evals 这是一个列表，用于对训练过程中进行评估列表中的元素。形式是 evals = [(dtrain,'train'),(dval,'val')]或者是 evals = [(dtrain,'train')],对于第一种情况，它使得我们可以在训练过程中观察验证集的效果。
obj,自定义目的函数
feval,自定义评估函数
early_stopping_rounds,早期停止次数 ，假设为 100，验证集的误差迭代到一定程度在 100 次内不能再继续降低，就停止迭代。这要求 evals 里至少有 一个元素，如果有多个，按最后一个去执行。返回的是最后的迭代次数（不是最好的）。如果 early_stopping_rounds 存在，则模型会生成三个属性，bst.best_score,bst.best_iteration,和 bst.best_ntree_limit
evals_result 字典，存储在 watchlist 中的元素的评估结果。
verbose_eval (可以输入布尔型或数值型)，也要求 evals 里至少有 一个元素。如果为 True，则对 evals 中元素的评估结果会输出在结果中；如果输入数字，假设为 5，则每隔 5 个迭代输出一次。
xgb_model ,在训练之前用于加载的 xgb model。
scale_pos_weight [默认 1]
在各类别样本十分不平衡时，把这个参数设定为一个正值，可以使算法更快收敛。
max_delta_step[默认 0]
这参数限制每棵树权重改变的最大步长。如果这个参数的值为 0，那就意味着没有约束。如果它被赋予了某个正值，那么它会让这个算法更加保守。
通常，这个参数不需要设置。但是当各类别的样本十分不平衡时，它对逻辑回归是很有帮助的。

XGB 与 GBDT 对比
正则项：在使用 CART 作为基分类器时，XGBoost 显式地加入了正则项来控制模型的复杂度，有利于防止过拟合，从而提高模型的泛化能力。
二阶导数：GBDT 在模型训练时只使用了代价函数的一阶导数信息，XGBoost 对代价函数进行二阶泰勒展开，可以同时使用一阶和二阶导数。
基分类器：传统的 GBDT 采用 CART 作为基分类器，XGBoost 支持多种类型的基分类器，比如线性分类器。
列采样：传统的 GBDT 在每轮迭代时使用全部的数据，XGBoost 则采用了与随机森林相似的策略，支持对数据进行采样，支持列抽样，不仅能降低过拟合，还能减少计算，这也是 xgboost 异于传统 gbdt 的一个特性。
缺失值处理：传统的 GBDT 没有设计对缺失值进行处理，XGBoost 可以自动学习出它的分裂方向。XGBoost 对于确实值能预先学习一个默认的分裂方向。
Shrinkage（缩减），相当于学习速率（xgboost 中的 eta）。xgboost 在进行完一次迭代后，会将叶子节点的权重乘上该系数，主要是为了削弱每棵树的影响，让后面有更大的学习空间。实际应用中，一般把 eta 设置得小一点，然后迭代次数设置得大一点。（补充：传统 GBDT 的实现也有学习速率）

原理：
1.提升方法是一种非常有效的机器学习方法，在前几篇笔记中介绍了提升树与 GBDT 基本原理，xgboost（eXtreme Gradient Boosting）可以说是提升方法的完全加强版本。xgboost 算法在各大比赛中展现了强大的威力。
2.Regression Tree and Ensemble (What are we Learning，得到学习目标)

（1）.Regression Tree (CART)回归树
（2）.Regression Tree Ensemble 回归树集成
在上面的例子中，我们用两棵树来进行预测。我们对于每个样本的预测结果就是每棵树预测分数的和。
（3）.Objective for Tree Ensemble 得到学习目标函数
这里是构造一个目标函数，然后我们要做的就是去尝试优化这个目标函数。读到这里，感觉与 gbdt 好像没有什么区别，确实如此，不过在后面就能看到他们的不同了（构造（学习）模型参数）。
3. Gradient Boosting

2.3 基准指标

本次竞赛以 RMSPE 作为误差函数来进行评价。
考虑到训练所需要的大量硬件资源，以及需要的大量时间，受限于客观条件，项目设定的目标为 RMSPE 小于 0.11

3 具体方法

3.1 数据预处理

3.1.1 数据分析

```
the number of the store 1115
the feature of the store 10
the samples of the train data 1017209
the features of the train data 9
the samples of the test data 41088
the features of the test data 8
```

从对数据集的分析可知，一共有 1115 个商店。训练集的样本数量超过 100 万个，是一个比较庞大的数据集。

Train	Store	Test
Store	Store	Id
DayOfWeek	StoreType	Store
Sales	Assortment	DayOfWeek
Customers	CompetitionDistance	Date
Open	CompetitionOpenSinceMonth	Open
Promo	Promo2	Promo
StateHoliday	Promo2SinceWeek	StateHoliday
SchoolHoliday	Promo2SinceYear	SchoolHoliday
	PromoInternal	

图 特征分布

由图可知有三个数据及其中的 store 是关于商店类型的，因此在后续数据处理时考虑将 store 分别和 test 和 train 进行合并。

	Amount of unique values		Unique values		NaNs	
Field name	Training set	Test set	Training set	Test set	Training	Test
Store	1115	856			0	0
DayOfWeek	7	7	5 4 3 2 1 7 6	4 3 2 1 7 6 5	0	0
Date	942	48			0	0
Sales	21734	-			0	0
Customers	4086	-			0	0
Open	2	2	1 0	1 nan 0	0	11
Promo	2	2	1 0	1 0	0	0
StateHoliday	5	2	'0' 'a' 'b' 'c' 0L	'0' 'a'	0	0
SchoolHoliday	2	2	1 0	1 0	0	0

图 唯一值 缺失值

从表格中我们可以看到，在 test 集 open 中有 11 个缺失值，同时在 train 和 test 中，stateholiday 并不一致，test 中只有 0 和 a。

3.1.2 数据集合并

本项目给了三个数据集，有 store，test 和 train。其中 store 是关于每一个商店的信息的，因此将 store 分别和 test 和 train 进行合并。

3.1.3 数据变换

```
<matplotlib.axes._subplots.AxesSubplot at 0x2bc00453c48>
```

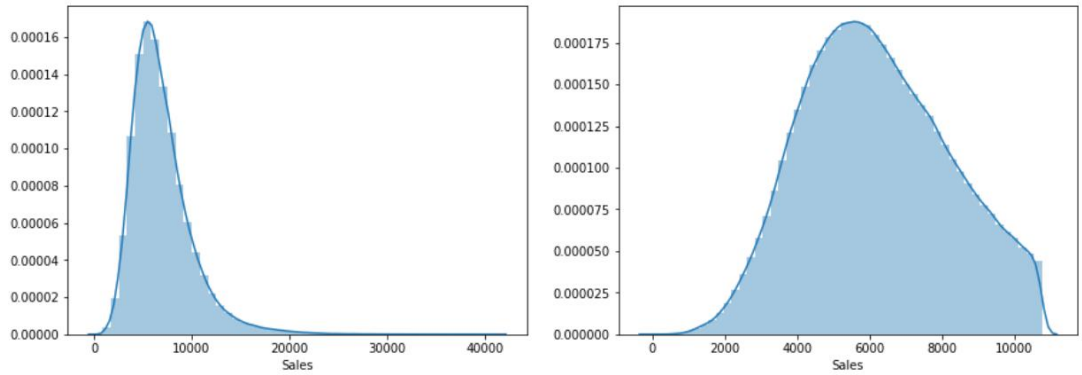


图 长尾分布

这里很明显可以看出来数据的标签（Sales）是长尾分布，并不利于数据的预测，因为很多的模型都假设数据符合正态分布，所以我们需要对长尾分布的数据进行对数变换。

<matplotlib.axes._subplots.AxesSubplot at 0x2bc2b399108>

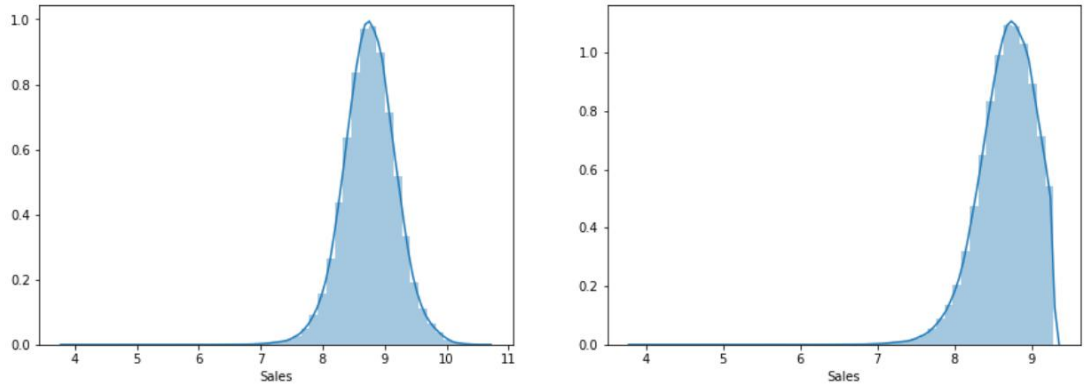


图 正态分布

对数变换后的数据更加符合正态分布，有利于之后训练模型。

3.1.3 缺失值处理

店铺竞争数据缺失，而且缺失的都是对应的。原因不明，而且数量也比较多，如果用中值或均值来填充，有失偏颇。暂且填 0，解释意义就是刚开业
店铺促销信息的缺失是因为没有参加促销活动，所以我们以 0 填充

3.2 实现

#3.2 章节可以总结一下你的算法实施过程。

3.2.1 构建特征

首先将数据集中不需要处理的特征，加入到选择的特征集中，有'Store', 'CompetitionDistance','Promo', 'Promo2', 'SchoolHoliday'。
然后对'StoreType', 'Assortment', 'StateHoliday'这几个分类变量进行 0-4 编码。
在处理时间序列，因为所给的时间序列是带年月日的完整日期，所以需要将年月日分开来形成三个单独的特征，并获取每天的星期以及在当年的周数，从而形成 5 个特征。'DayOfWeek', 'Month', 'Day', 'Year', 'WeekOfYear'
最后再对'CompetitionOpen','PromoOpen'这两个变量进行处理，将所给的日期转化成所持续的时间数据。
最终选择的特征是'Store', 'CompetitionDistance', 'Promo', 'Promo2', 'SchoolHoliday','StoreType','Assortment','StateHoliday','DayOfWeek','Month', 'Day', 'Year', 'WeekOfYear', 'CompetitionOpen', 'PromoOpen'

3.2.2 模型的训练

使用 xgb 模型进行训练，程序设置了 Early Stop，监视 val_loss。当 val_loss 连续 100 个 epoch 没有改进时停 止训练。

3.2.3 结果的生成

Validating RMSPE: 0.102849

3.3 改进

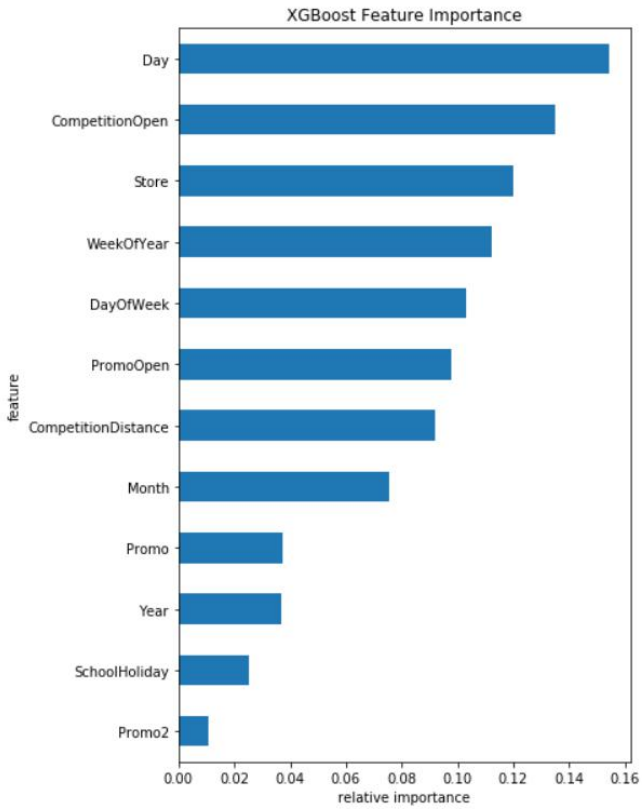
增加其他模型，如时间序列线性模型，随机森林。
进行多模型融合，以达到比单个模型本身更好的效果。

4 结果

4.1 模型评价与验证

4.1.1 特征分析

图 模型特征重要性排序



从图上可以知道，在模型中，最重要的特征有天数，竞争对手开店时间，商店种类。由 fscore 可知分别占了 0.154189，0.134849，0.120229

4.2 结果分析

Validating　RMSPE: 0.102849，达到预期。

5 结论

挑战：处理海量数据（10,17,210 个样本，13 个特征）
180 个商店关闭了 6 个月，无法填补这些商店销量的空白。
对 1115 左右个商店进行预测，且商店门类还有很大不同。

The Rossmann Store Sales problem 此次的预测销售额竞赛事项，关于数据挖掘的任务，其中，我认为特征选择比模型选择更为重要，我花了大部分的时间在于挖掘数据之间的联系，以构建更好的特征。

参考文献

[1] Kaggle 官网
[2]https://xgboost.readthedocs.org/en/latest/
[3] https://www.kaggle.com/wiki/RandomForests