Rossmanm 销售预测

I. 问题的定义

项目概述

Rossmann 在 7 个欧洲国家经营着 3000 多家药店。目前，Rossmann 门店经理的任务是提前 6 周预测他们的日常销售。商店的销售受到许多因素的影响，包括促销、竞争、学校和国家假日、季节性和地区。由于成千上万的个人经理根据自己的特殊情况预测销售情况，结果的准确性可能会非常不同。

在他们的第一个 Kaggle 竞赛中，Rossmann 挑战通过在德国各地的 1,115 家商店的每日销售额来预测未来 6 周的销售额。可靠的销售预测使商店经理能够制定有效的员工时间表，提高生产力和积极性。通过帮助 Rossmann 创建一个健壮的预测模型，将会帮助商店经理专注于他们最重要的事情:他们的客户和他们的团队。

问题陈述

Rossmann 是欧洲的一家连锁药店。在这个源自 Kaggle 比赛 Rossmann Store Sales 中，我们需要根据 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。在 kaggle 中选手 Paso[4]同对数据中竞争对手距离、所开店的年月进行特征工程，通过 xgboost 进行建模来解决问题。选手 Elena Petrona[5]通过 Facebook 提供的时间序列算法 Prophet 对数据建模，通过在时间序列的维度上对问题进行了分析。

评价指标

提交的内容是根据均方根百分比误差(RMSPE)计算的。RMSPE 被计算为

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

其中 yi 表示单个商店在一天内的销售额，yhat_i 表示相应的预测。任何一天和商店 0 的销售都被忽视得分。

II. 分析

数据的探索与可视化

输入的数据有两部分：

1. Rossmann 商店信息数据：
    1. Store（商店 ID）
    2. DayOfWeek（日期所在的周几）
    3. Date（销售时间）
    4. Sales（销售额）
    5. Customers（消费者数量）
    6. Open（商店当日是否开放）
    7. Promo（商店当日是否促销）
    8. StateHoliday（国家节假日，a、b、c 是国家节假日，0 表示非节假日）
    9. SchoolHoliday（国家公立学校节假日，1：节日，0 非节日）
2. 商店销售数据。
    1. Store（商店 ID）
    2. StoreType（商店类型）
    3. Assortment（商店级别）
    4. CompetitionDistance（竞争对手距离）
    5. CompetitionOpenSinceMonth（竞争对手开店月份，部分缺失）
    6. CompetitionOpenSinceYear（竞争对手开店年份，部分缺失）
    7. Promo2（是否持续促销）
    8. Promo2SinceWeek（持续促销的周）
    9. Promo2SinceYear（持续促销的年）
    10. PromoInterval（促销的月份）
3. 缺失数据分析

```
<class 'pandas.core.frame.DataFrame'>
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
dtypes: float64(5), int64(2), object(3)
memory usage: 87.2+ KB
```

图 1

如图 1 所示 CompetitionDistance 有三百家数据缺失，Promo2 促销日有 571 天，竞争对手开店时间大量缺失,缺失 206 条数据，如图 2 所示分析竞争对手开店时间与促销关系。

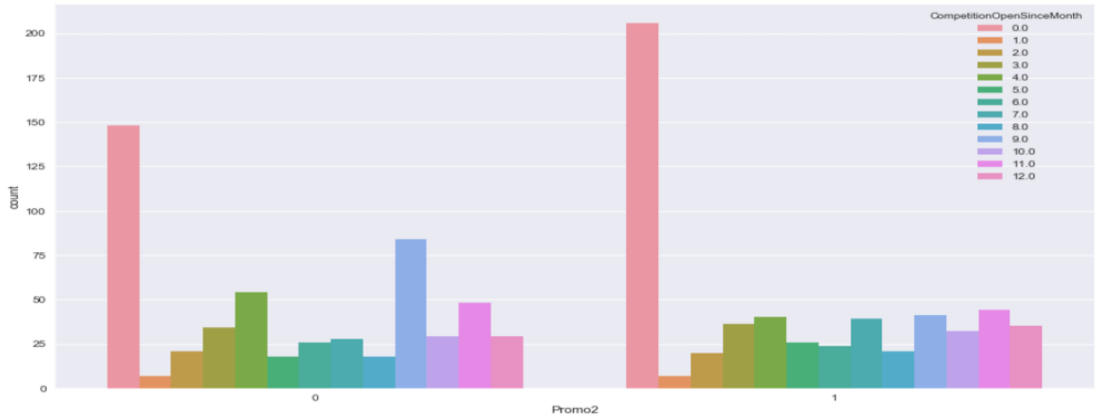


图 2

如图 3 所示通过对一周数据探索发现第七天绝大部分店铺都没有开门，而很少的店铺开门，说明这第七天开门的数据存在有一些其它因素干扰。

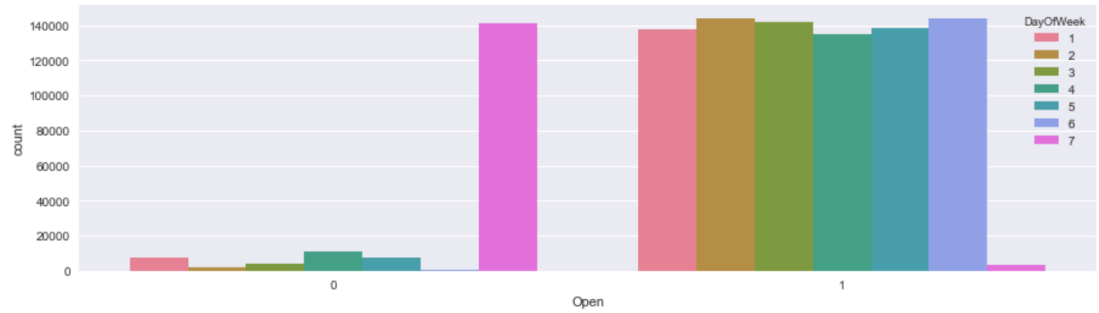


图 3

查看每月平均销售和销售百分比发现到 3、12 月份平均销售是个高峰，而这个时候一般是节日比较多。

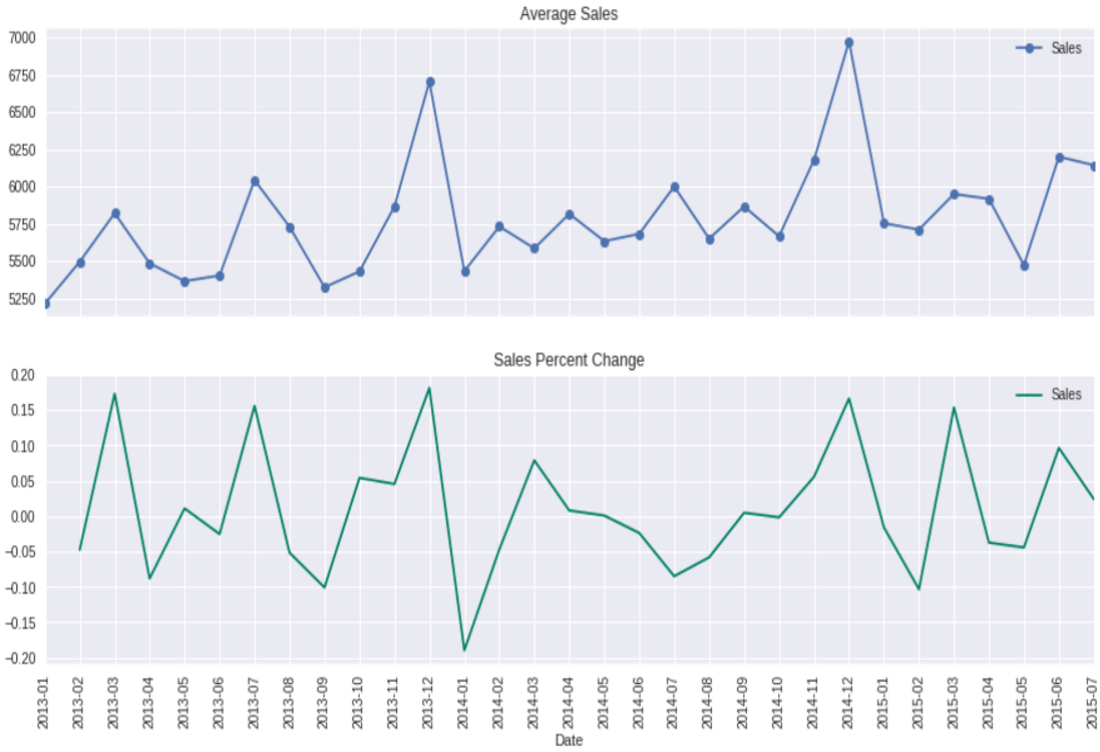


图 4

根据图 5 中可以看出销售额和用户数在促销和非促销的对比情况，促销状况下销售额和用户数都比较多。

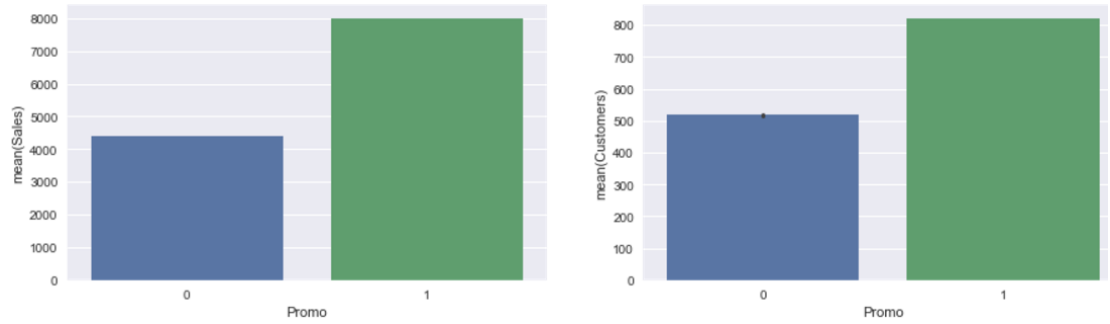
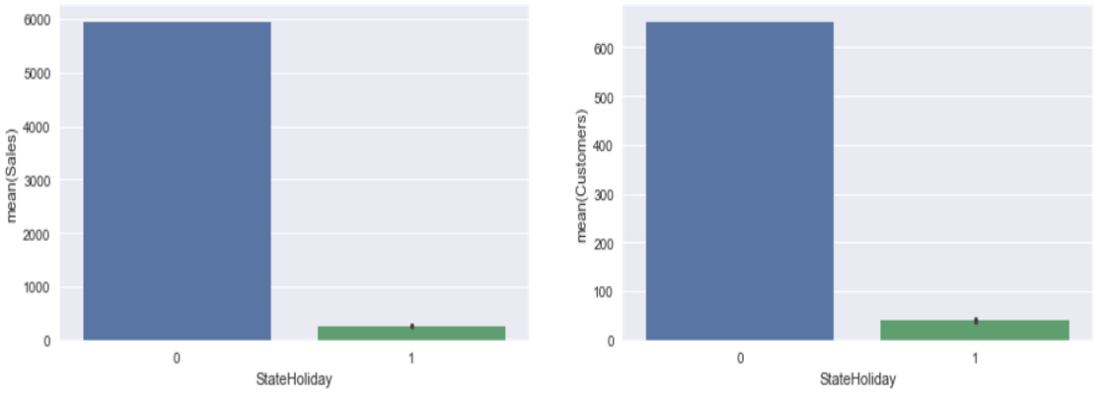


图 5

节假日和非节假日对比可以看出非节假日的平均销售额多，并且三种节日类型的销售额均值相差不大如图 6 所示。



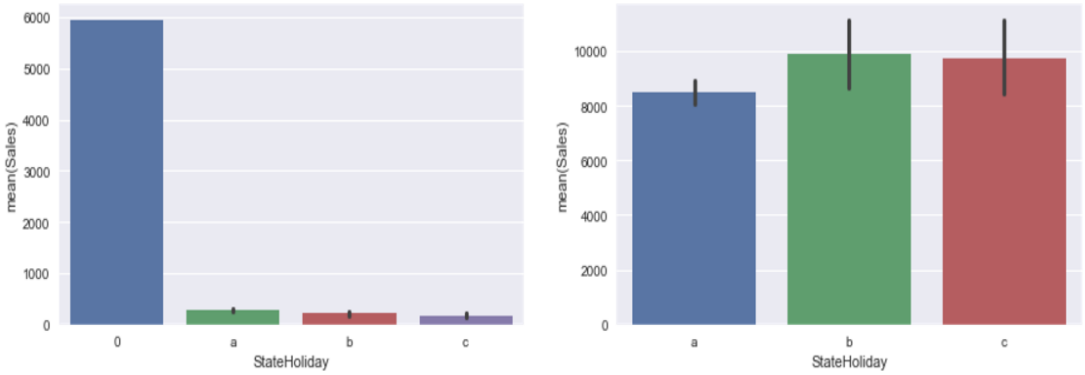


图 6

根据商店类型的统计量和各个商店类型的平均销售额和平均用户数，发现 b 类型商店数量少，但是销售额和用户数多如图 7 所示。

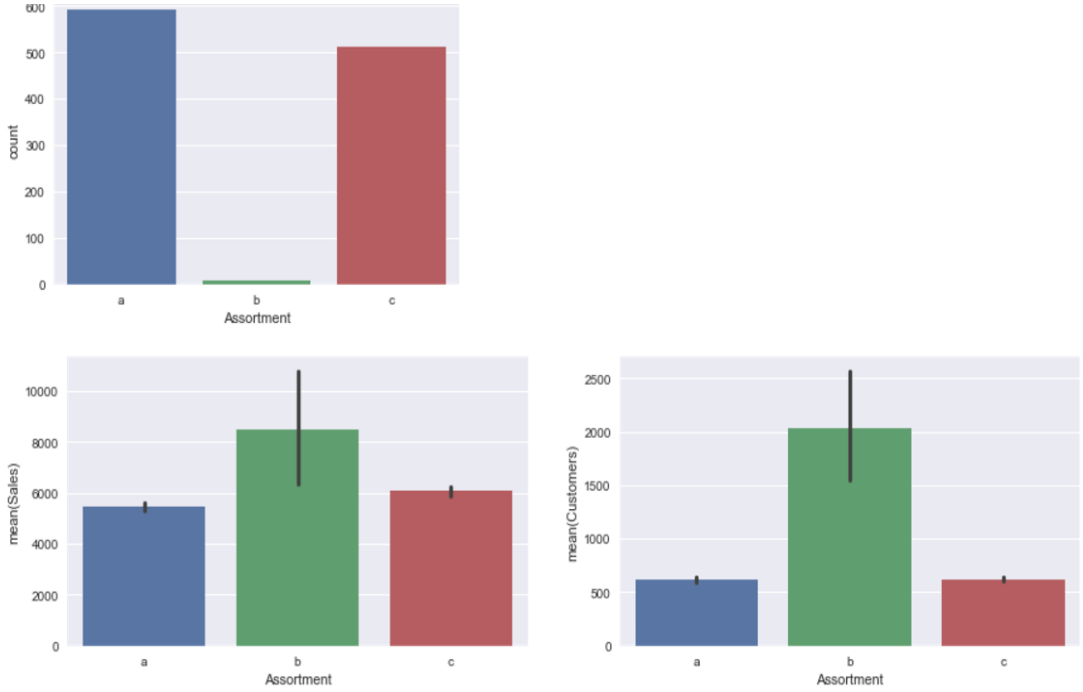


图 7

从竞争对手的距离可以可以发现销售额越高，竞争对手越近，并且竞争对手大部分集中在 2000~8000 距离之间如图 8 所示：

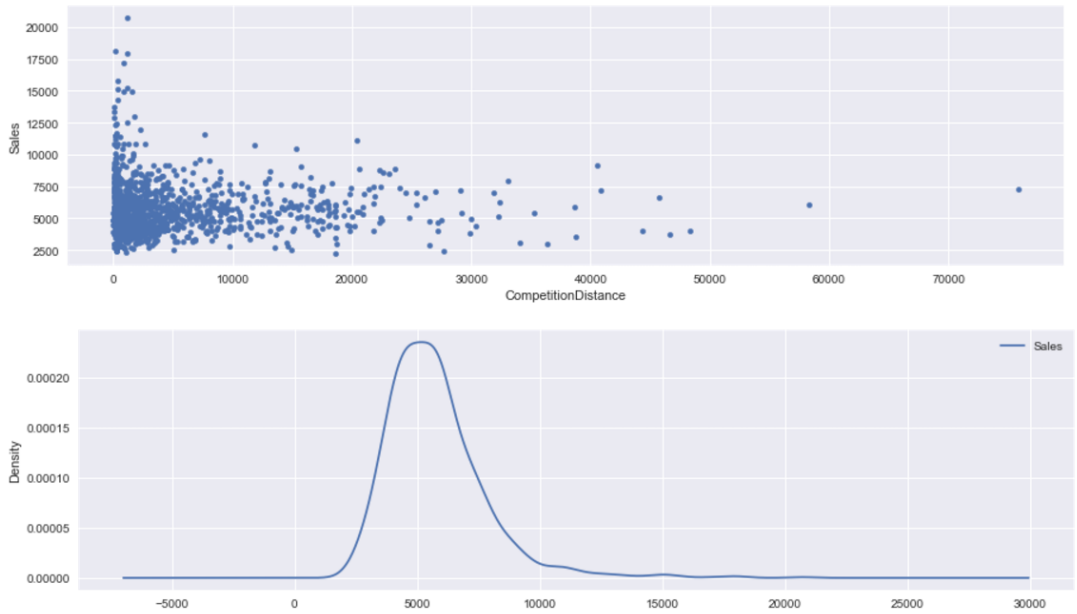


图 8

通过对前 5 个商店进行相关性分析如图 9 所示发现商店之间的相关性很大。

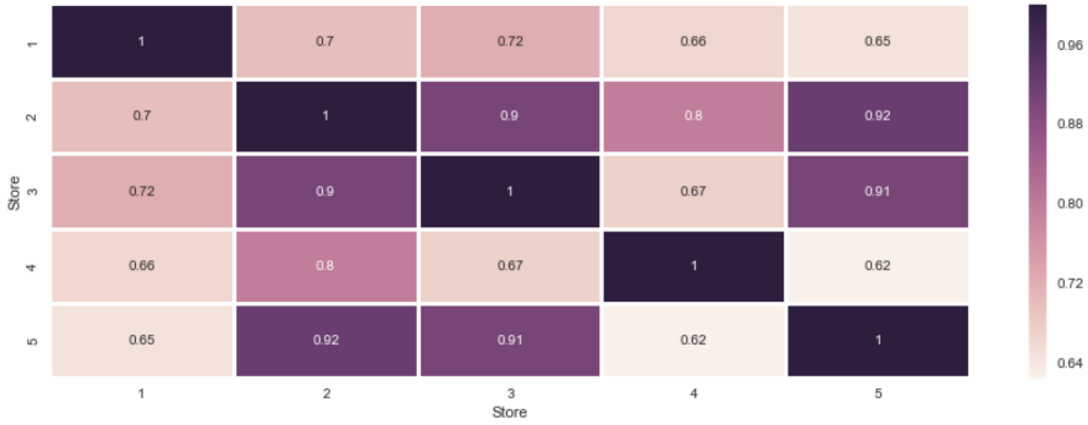


图 9

对于单个销售额的特征通过展示发现结果值存在大量的异常值如图 10 所示；

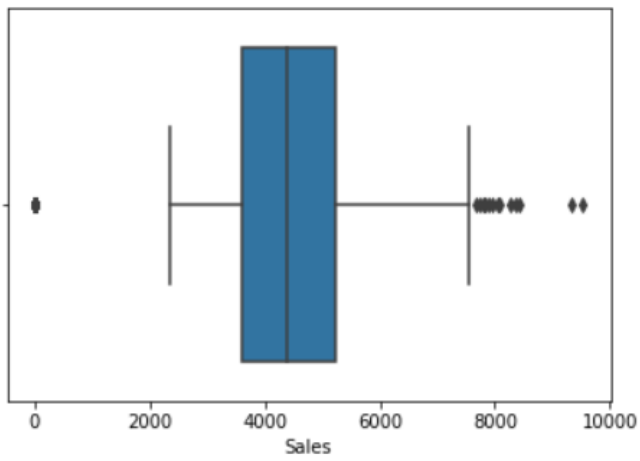


图 10

根据图 10 的箱图发现的异常值，再通过对单个店面的正态分布分析发现异常值的数据的数量较少如图 11 所示：

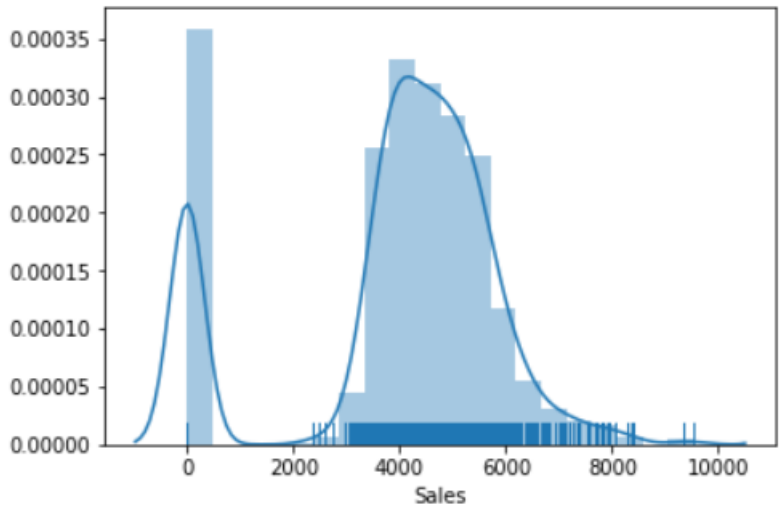


图 11

算法和技术

为了对数据建模来实现对商店销售额的准确的预测需要考虑以下三个方面因素：

1. 理论模型
2. 实际数据
3. 系统的实现

1) 以在理论模型的角度来分析

一个机器学习模型想要取得好的效果，这个模型需要满足以下两个条件：

1. 模型在我们的训练数据上的表现要不错：也就是 trainning error 要足够小。
2. 模型的 vc-dimension 要低。换句话说，就是模型的自由度不能太大，以防 overfit.(vc dimension 的值越大，参数的个数越多，模型越复杂)

1.对于 LR 模型。如果 underfit，我们可以通过加 feature，或者通过高次的特征转换来使得我们的模型在训练数据上取得足够高的正确率。而对于 tree-enseble 来说，我们解决这一问题的方法是通过训练更多的“弱弱” 的 tree. 所以，这两类模型都可以把 training error 做的足够低，也就是说模型的表达能力都是足够的。

2.在 tree-ensemble 模型中，通过加 tree 的方式，对于模型的 vc-dimension 的改变是比较小的。而在 LR 中，初始的维数设定，或者说特征的高次转换对于 vc-dimension 的影响都是更大的。所以，一不小心我们设定的多项式维数高了，模型就“刹不住车了”。这也就是我们之前说的，tree-ensemble 模型的可控性更好，也即更不容易 overfit。

2) 以在数据的角度来分析

1.kaggle 比赛选择的都是真实世界中的问题。所以数据多多少少都是有噪音的。而基于树的算法通常抗噪能力更强。

2.除了数据噪音之外，feature 的多样性也是 tree-ensemble 模型能够取得更好效果的原因之一。特征的多样性也正是为什么工业界很少去使用 svm 的一个重要原因之一（因为会过拟合 overfitting），因为 svm 本质上是属于一个几何模型，这个模型需要去定义 instance 之间的 kernel 或者 similarity。

噪声问题：学习了天鹅的外形（全局特征）之后，你的例子都是白色的。计算机看见黑天鹅就认为不是天鹅。（黑色只是局部特征）

过拟合：模型对数据学习太彻底了，所有样本都学了，包括局部特征，所以识别新样本没有几个是对的。
解决方法：不需要十分彻底，降低机器学习局部特征和错误特征机率。收集多样化样本，简化模型，交叉验证。
3)以系统实现的角度来分析

除了有合适的模型和数据，一个良好的机器学习系统实现往往也是算法最终能否取得好的效果的关键。一个好的机器学习系统实现应该具备以下特征：

1. 正确高效的实现某种模型。而高效的实现意味着可以快速验证不同的模型和参数。
2. 系统具有灵活、深度的定制功能。
3. 系统简单易用。
4. 系统具有可扩展性，可以从容处理更大的数据。

模型的选取

本项目选取线性回归、决策树、随机森林、xgboost 这四个经典模型来训练模型。通过最后模型效果选择最优模型来做最终模型。

1）线性回归模型

对于线性回归模型做预测是很不错的，它不仅可以预测并求出函数，还可以自己对结果进行残差的检验，检验模型的精度。但是回归模型比较简单，算法相对低级。回归方程假设严格，需要知道引起因变量改变的所有解释变量的因素，否则回出现伪回归等问题，假设检验不过关。

具体优点：

1、回归分析法在分析多因素模型时，更加简单和方便；

2、运用回归模型，只要采用的模型和数据相同，通过标准的统计方法可以计算出唯一的结果，但在图和表的形式中，数据之间关系的解释往往因人而异，不同分析者画出的拟合曲线很可能也是不一样的；

3、回归分析可以准确地计量各个因素之间的相关程度与回归拟合程度的高低，提高预测方程式的效果；在回归分析法时，由于实际一个变量仅受单个因素的影响的情况极少，要注意模式的适合范围，所以一元回归分析法适用确实存在一个对因变量影响作用明显高于其他因素的变量是使用。多元回归分析法比较适用于实际经济问题，受多因素综合影响时使用。

缺点：

1.在回归分析中，选用何种因子和该因子采用何种表达。

2.回归方程式只是一种推测，这影响了因子的多样性和某些因子的不可测性，使得回归分析在某些情况下受到限制。

2)回归决策树模型

决策树是一种树形结构，其中每个内部节点表示一个属性上的测试，每个分支代表一个测试输出，每个叶节点代表一种类别。

决策树(Decision Tree）是在已知各种情况发生概率的基础上，通过构成决策树来求取净现值的期望值大于等于零的概率，评价项目风险，判断其可行性的决策分析方法，是直观运用概率分析的一种图解法。

构建决策树采用贪心算法，只考虑当前纯度差最大的情况作为分割点。

决策树仅有单一输出，若欲有复数输出，可以建立独立的决策树以处理不同输出。

相对于其他数据挖掘算法，决策树在以下几个方面拥有优势：

1、决策树易于理解和实现. 人们在通过解释后都有能力去理解决策树所表达的意义。

2、对于决策树，数据的准备往往是简单或者是不必要的 . 其他的技术往往要求先把数据一般化，比如去掉多余的或者空白的属性。

3、能够同时处理数据型和常规型属性。其他的技术往往要求数据属性的单一。

4、在相对短的时间内能够对大型数据源做出可行且效果良好的结果。

5、对缺失值不敏感

6、可以处理不相关特征数据

7、效率高，决策树只需要一次构建，反复使用，每一次预测的最大计算次数不超过决策树的深度。

决策树的缺点：

1、对有时间顺序的数据，需要很多预处理的工作。

2、当类别太多时，错误可能就会增加的比较快。

3、一般的算法分类的时候，只是根据一个字段来分类。

4、在处理特征关联性比较强的数据时表现得不是太好

3)随机森林模型

Random Forest（随机森林）作为 Bagging 模型的变体，它在以决策树为基学习器构建 Bagging 集成的基础上，进一步在决策树的训练过程中引入了随机特征选择，因此可以概括 RF 包括四个部分：1、随机选择样本（放回抽样）；2、随机选择特征；3、构建决策树；4、随机森林投票（平均）。

随机选择样本和 Bagging 相同，随机选择特征是指在树的构建中，会从样本集的特征集合中随机选择部分特征，然后再从这个子集中选择最优的属性用于划分，这种随机性导致随机森林的偏差会有稍微的增加（相比于单棵不随机树），但是由于随机森林的‘平均’特性，会使得它的方差减小，而且方差的减小补偿了偏差的增大，因此总体而言是更好的模型。在构建决策树的时候，RF 的每棵决策树都最大可能的进行生长而不进行剪枝；在对预测输出进行结合时，RF 通常对分类问题使用简单投票法，回归任务使用简单平均法。

RF 的重要特性是不用对其进行交叉验证或者使用一个独立的测试集获得无偏估计，它可以在内部进行评估，也就是说在生成的过程中可以对误差进行无偏估计，由于每个基学习器只使用了训练集中约 63.2%的样本，剩下约 36.8%的样本可用做验证集来对其泛化性能进行‘包外估计’。

RF 和 Bagging 对比：RF 的起始性能较差，特别当只有一个基学习器时，随着学习器数目增多，随机森林通常会收敛到更低的泛化误差。随机森林的训练效率也会高于 Bagging，因为在单个决策树的构建中，Bagging 使用的是‘确定性’决策树，在选择特征划分结点时，要对所有的特征进行考虑，而随机森林使用的是‘随机性’特征数，只需考虑特征的子集。

随机森林的优点较多，简单总结：

1、在数据集上表现良好，相对于其他算法有较大的优势（训练速度、预测准确度）；

2、能够处理很高维的数据，并且不用特征选择，而且在训练完后，给出特征的重要性；

3、容易做成并行化方法。

RF 的缺点：

在噪声较大的分类或者回归问题上回过拟合。

4）xgboost 模型

XGBoost 的性能在 GBDT 上又有一步提升，而其性能也能通过各种比赛管窥一二。坊间对 XGBoost 最大的认知在于其能够自动地运用 CPU 的多线程进行并行计算，同时在算法精度上也进行了精度的提高。由于 GBDT 在合理的参数设置下，往往要生成一定数量的树才能达到令人满意的准确率，在数据集较复杂时，模型可能需要几千次迭代运算。但是 XGBoost 利用并行的 CPU 更好的解决了这个问题。

Xgboost 的优点：

1.xgboost 高效的 c++实现能够通常能够比其它机器学习库更快的完成训练任务。

2. 在灵活性方面，xgboost 可以深度定制每一个子分类器，并且可以灵活的选择 loss function（logistic，linear，softmax 等等）。xgboost 提供了各种语言的封装，使得不同语言的用户都可以使用这个优秀的系统

3.xgboost 提供了分布式训练（底层采用 rabit 接口），并且其分布式版本可以跑在各种平台之上，例如 mpi,yarn,spark 等等。。

4.xgboost 能自动利用 cpu 的多线程，而且适当改进了 gradientboosting，加了剪枝，控制了模型的复杂程度。

XGBoost 的建模思路：就是在每轮迭代中生成一棵新的回归树，并综合所有回归树的结果，使预测值越来越逼近真实值。

3．XGBoost 在代价函数中加入了正则项，用于控制模型的复杂度。从权衡方差偏差来看，它降低了模型的方差，使学习出来的模型更加简单，放置过拟合，这也是 XGBoost 优于传统 GBDT 的一个特性。

5.Shrinkage（缩减），相当于学习速率（XGBoost 中的 eta）。XGBoost 在进行完一次迭代时，会将叶子节点的权值乘上该系数，主要是为了削弱每棵树的影响，让后面有更大的学习空间。（GBDT 也有学习速率）。

6.列抽样。XGBoost 借鉴了随机森林的做法，支持列抽样，不仅防止过拟合，还能减少计算。

7.对缺失值的处理。对于特征的值有缺失的样本，XGBoost 还可以自动学习出它的分裂方向。

8.XGBoost 工具支持并行。XGBoost 是一次迭代完才能进行下一次迭代的（第 t 次迭代的代价函数里包含了前面 t-1 次迭代的预测值）。XGBoost 的并行是在特征粒度上的。决策树的学习最耗时的一个步骤就是对特征的值进行排序（因为要确定最佳分割点），XGBoost 在训练之前，预先对数据进行了排序，然后保存为 block 结构，后面的迭代中重复地使用这个结构，大大减小计算量。这个 block 结构也使得并行成为了可能，在进行节点的分裂时，需要计算每个特征的增益，最终选增益最大的那个特征去做分裂，那么各个特征的增益计算就可以开多线程进行。

基准模型

根据项目背景和问题，为未来 6 周各个门店的销售额进行预测，本项目开始参考 Omar El Gabry 的特征分析方法：

1. 分析不同类型商店的平均销售。
2. 分析一周七天的平均销售额。
3. 分析连续几个月的销售额。
4. 进行促销对比分析。
5. 进行持续促销对比分析。
6. 是否公共节假日对比分析。
7. 通过箱型图查看销售额，用户数分布。
8. 对商店类型进行对比分析。
9. 对商店等级进行对比分析。
10. 查看竞争对手距离分布。
11. 查看用户数分布。

12. 进行商店之间相关性分析。

通过分析发现一周的不同时间、节假日、促销、商店类型等对销售额有影响，通过 one-hot-encoding 编码实现特征工程，在根据 XGboost 算法分类器把成百上千个分类准确率较低的树模型组合起来，成为一个准确率很高的模型；这个模型会不断地迭代，每次迭代就生成一颗新的回归树，在对每个商店单独建模预测数据，最后通过搜索优化提高模型的准确率，最终能够模型的误差低于 0.11773 能够进入排名 top10%。

III. 方法

数据预处理

1. 由于商店是在营业才能有销售额，所以对删除 open=0 的数据，对测试集中 open=0 的数据赋值为 0。

2. 由于销售额与销售日期有关，所以需要把当日时间的年月日拆分为三个年、月、日变量。

3. 对 PromoInterval 特征对应的促销月份转换到对应时间的月份。

4. 把字符型变量进行转换。

5. 对于异常数据通过参考 stackoverflow Joe Kington[4]的数据异常处理对 Sales 数据进行异常值判定，进行数据筛选。

特征工程：

二：根据探索分析的结果来对数据进行特征工程处理：

1. 根据每个商店每日的销售额来计算该商店过去的每周、每月、每日销售额进行均值计算。

2. 由于节假日对商店的销售额影响很大，所以计算各个商店当日到节假日的时间。

3. 通过对商店分析计算各个商店的每个月、季、年、是否有促销的销售额的销售额均值。

执行过程

根据项目要求可知本项目是建立一个回归预测模型，所以选择目前可靠性最好的几个回归模型线性回归、随机森林树回归、决策树回归、xgboost 回归通过参考 Omar El Gabry 对单个店（商店编号为 8）进行建模进行对比分析如表 1：

模型	RMSPE
LinearRegression	0.48695183060797914
RandomForestRegressor	0.2777938516882318
DecisionTreeRegressor	0.35059177563694827
XGboost	0.152735

表 1

模型的优化

根据上几步的数据的处理、模型的比较以及参数的优化，随机选择一个商店的数据进行优化，其中模型中各个参数的权重如图 12 所示，DayOfyear、DayofMonth、CompetitionOpen、WeekOfYear 特征对模型影响很大，模型的误差为 0.43。

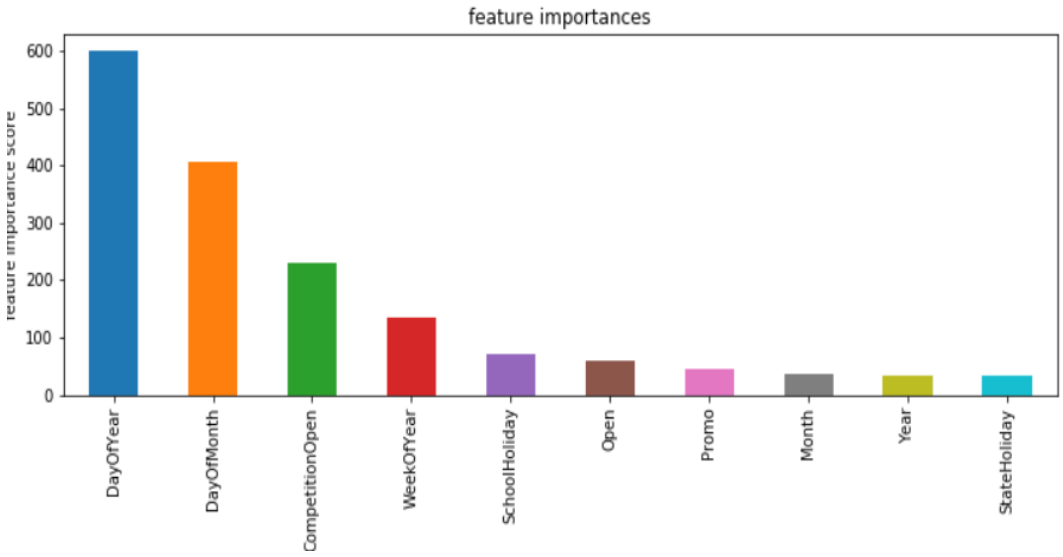


图 12

通过 scikit_learn 中的 GridSearchCV 函数对 xgboost 算法中 max_depth 和 min_child_weight 进行参数搜索，并通过 6 折交叉验证来减少误差。最终模型可靠性为 0.93，搜索 max_depth 为 8，min_child_weight 为 12。各个特征的权重如下图所示：

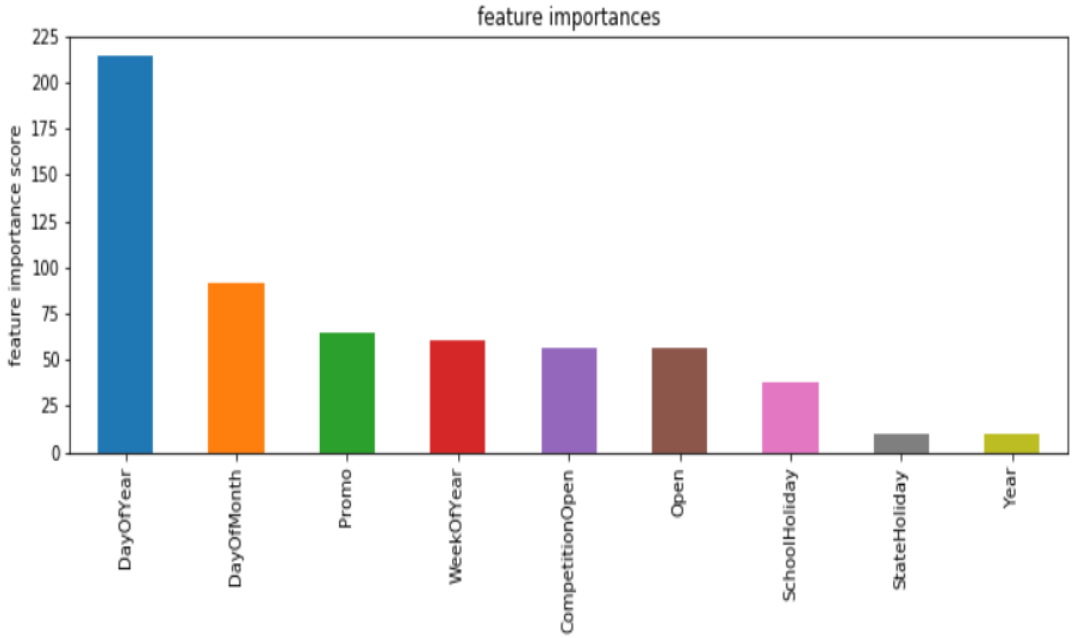


图 13

对参数 gamma、subsample、colsample_bytree 进行调优得到 gamma 等于 0、subsample 等于 0.8，colsample_bytree 为 0.6 时模型误差减小到 0.4135。

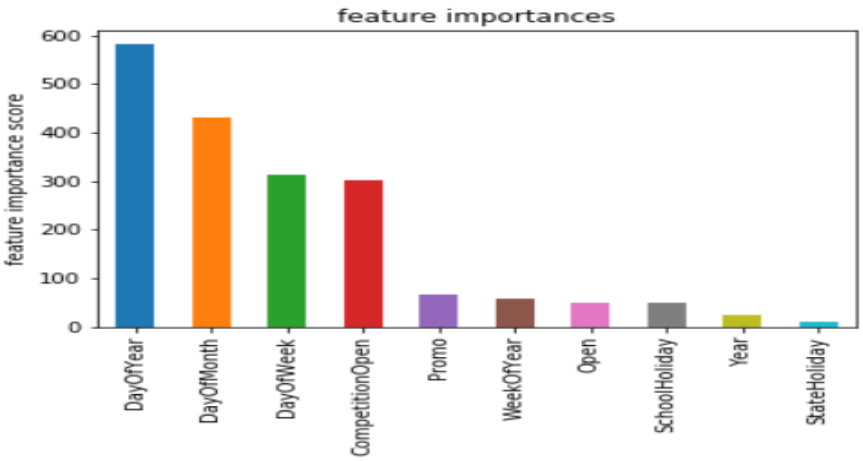


图 14

合理性分析

综合各个参数的优化，发现 max_depth 为 8、min_child_weight 为 12、gamma 为 0、subsample 为 0.8、ntrees 为 8000 时模型最优。

根据 gridsearch 搜索的函数在 xgboost.train 函数下进行训练，为了能够达到最好的准确率，通过手动调整模型参数，最终选择 max_depth: 9,subsample: 0.7,alpha: 2, gamma: 2, colsample_bytree: 0.8,silent': 1 为最优参数，并且模型的各个参数权重如图 15 所示：

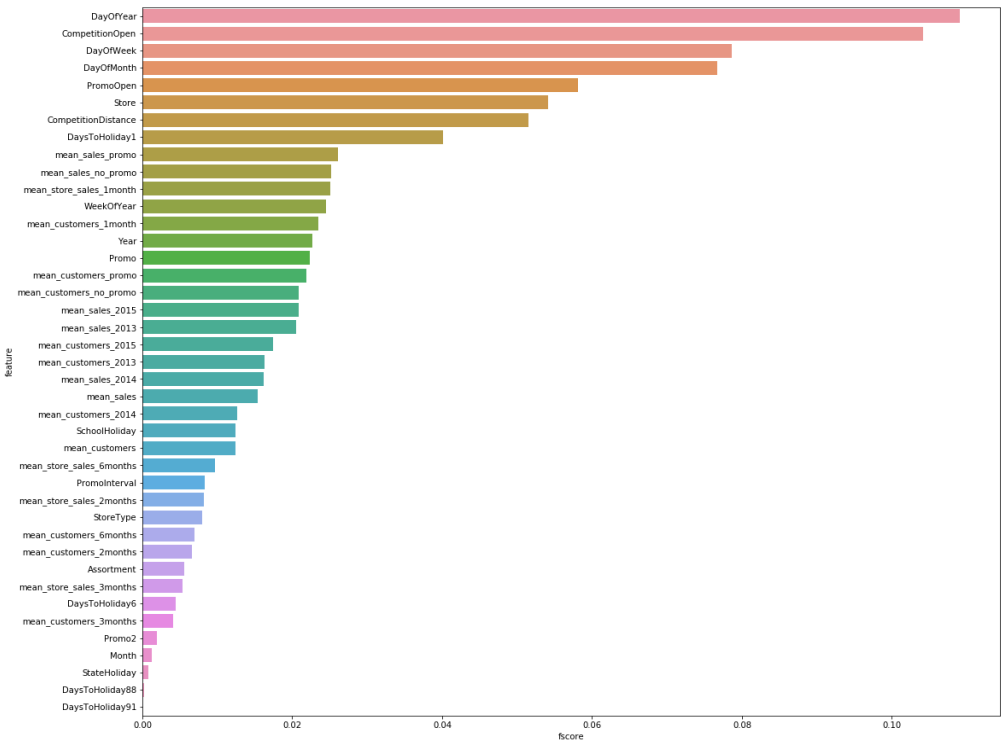


图 15

项目结果

最后计算结果为 0.11613,低于前 10%选手的最低得分 0.11773，满足项目期望。

Private Score	Public Score
0.11533	0.11049

V. 项目结论

根据最后模型输出的结果如图 16 所示，前十个重要特征可得知模型最容易受到 DayofYear 影响，其次受到 CompetitionOpen 、DayofWeek 对模型影响最高，说明零售商店对每一天所在的时间维度非常重要，对模型的改进可以通过与其它的日期进行特征工程，例如目前日期所在 DayofQuant,与国际节假日的时间距离，尽可能的与影响人们活动的日期进行联系构造特征。

竞争者的商店与所在位置商店的距离影响也很大，说明很符合现实的实际情况。

结果可视化

最后对模型的误差平方均值和标准差进行可视化如图 16 所示，预测值与实际值误差相差不大。

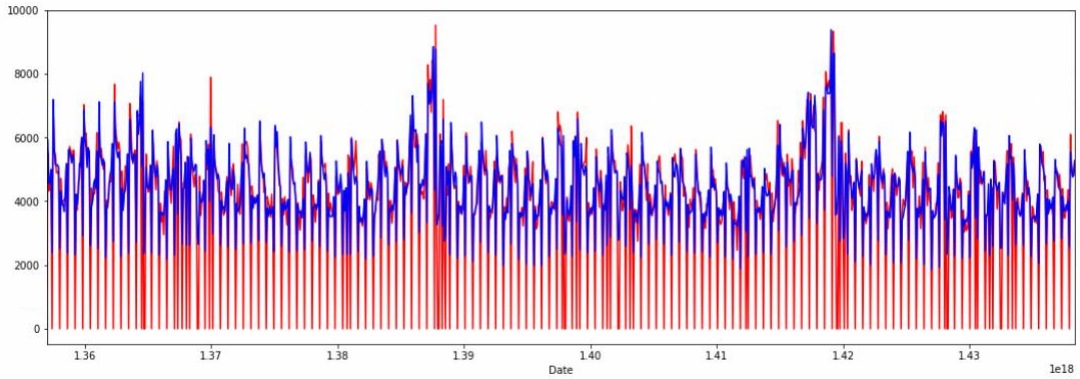


图 16

对项目的思考

对于该项目属于线下实体店的零售情况，实体店的零售的销售额情况与每个实体店所在的位置情况息息相关，如与竞争者距离的特征权重对模型影响很高，如果每个商店有周围人群的密度分布情况、人群流动情况、天气情况等这些信息就会大大提高模型的准确度。由于主办方提供 100 万的数据量，所以模型训练非常费时间，本人开始时期使用非常普通的电脑，训练一遍模型需要花费 3、4 个小时，后期选择 1080ti 的 GPU 来训练模型，一次需要 6 分钟，大大减少任务训练时间，所以在计算资源丰富的情况下可以对当日所在日期与其它所有日期

建立特征以及当前日期以前所有日期销售额的均值进行特征组合。从而能够大大降低模型的误差。

参考文献

[1] https://www.jianshu.com/p/0f96f97c3d5a
[2]https://www.jiqizhixin.com/articles/2017-12-20-2
[3]https://tech.meituan.com/machinelearning-data-feature-process.html
[4]https://stackoverflow.com/questions/22354094/pythonic-way-of-detecting-outliers-in-one-dimensional-observation-data/22357811#22357811
[5]https://www.kaggle.com/elenapetrova/time-series-analysis-and-forecasts-with-prophet
[6]https://www.kaggle.com/paso84/xgboost-in-python-with-rmspe