Rossman

笔记本：	Udacity		
创建时间：	2019/9/3 21:25	更新时间：	2019/9/9 22:09
作者：	herd_yz@163.com		
URL：	https://csrgxtu.github.io/2015/03/20/Writing-Mathematic-Fomulars-in-Markdown/		

机器学习纳米学位

##毕业项目
Joe 优达学城 </br>

2019年09月03日

Yuan, Zhi

1. 问题的定义

项目是针对rosman的历年销售额，进行分析建模以便可以预测未来的销售额。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。解决该问题涉及回归算法领域，数据集使用的rosman提供的销售数据以及门店信息数据。

1.1 项目概述

本项目来源于Kaggle竞赛Rossmann Store Sales。 Rossmann是德国首家平价日用商品，在欧洲拥有3000多分店。本项目所要解决的问题是，提前6周预测1115家商店的日常销售额。

1.2 研究内容

项目是一个有监督的回归问题，输入的数据由特征数据构成，目标是特征数据的预测销售额。需要对项目的输入数据进行特征处理，获取相关的数据来训练回归模型。通过最后的模型训练，可以有效的对未来的数据进行预测。我们将采用 XGBoost与 LightGBM方法，对

1115家商店未来6周的销售额进行预测。为了达到最终的目标，可以将问题分解以下几个方面：
1). 探索性数据分析，观察数据的特征，记录清洁度以及质量问题，是否存在缺失值和异常，分析特征之间的关系。
2). 特征工程，对特征进行预处理，选择对销售额预测影响比较大的特征，基于原始特征去构建与销售额更为相关的新特征。
3). 建立基准模型及模型优化，建立基准模型并对模型的参数进行调节，提高模型的性能，尝试组合不通的模型进一步提高预测精度和泛化能力

1.3 评价指标

根据Kaggle竞赛所提供的信息，评估标准采用RMSPE均方根百分误差：均方根百分比误差(Root Mean Square Percentage Error)。

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

RMSPE更贴近误差的概念。而相比于 MSE 和 RMSE， RMSPE 计算的是一个误差率，这样就避免了真实值之间大小的不同而对误差产生的影响。

2. 分析

2.1 数据的探索

2.2.1 数据特征

train.csv训练集中包含9个特征，1017209条记录。test.csv测试集中包含了8个特征，41088条记录，少了Sales和Customers 特征，增加了 Id 特征。
train.csv

特征	类型	含义
Store	定量数据	商店编号
DayOfWeek	定性数据	星期几
Date	定量数据	日期
Sales	定量数据	销售额
Customers	定量数据	客户数量
Open	定性数据	是否营业

特征	类型	含义
Promo	定性数据	是否促销
StateHoliday	定性数据	是否假日
SchoolHoliday	定性数据	是否是学校假日

test.csv

特征	类型	含义
Id	定量数据	编号
Store	定量数据	商店编号
DayOfWeek	定性数据	星期几
Date	定量数据	日期
Open	定性数据	是否营业
Promo	定性数据	是否促销
StateHoliday	定性数据	是否假日
SchoolHoliday	定性数据	是否是学校假日

store.csv

特征	类型	含义
Store	定量数据	商店编号
StoreType	定性数据	商店类型
Assortment	定性数据	日期类别
CompetitionDistance	定量数据	与最近竞争商店的距离
CompetitionOpenSinceMonth	定量数据	最近竞争商店开张月份
CompetitionOpenSinceYear	定性数据	最近竞争商店开张年份
Promo2	定性数据	是否有持续性的促销活动
Promo2SinceWeek	定性数据	开始持续性的促销的周
Promo2SinceYear	定性数据	开始持续性的促销的年
PromoInterval	定性数据	持续性的促销月

2.2 探索性可视化

TBD

2.3 算法和技术

2.3.1 算法

根据前面的分析可知，销售额预测本质上是一个回归问题。对于回归的算法有很多，比如线性回归，多项式回归，支持向量机， CART回归树等。这些算法都是机器学习的算法。通过将所需要训练的数据特征转化为特征向量，输出的是预测的最终销售数据。线性回归是求得线性预测的算发，然而对于有这么多特征的数据来说线性划分并不可靠，容易欠拟合。多项式回归能够对于多特征进行有效的拟合预测，可以有效的对特征数据进行分割，模拟。CART是一种二分递归分割的技术，分割方法采用基于最小距离的基尼指数估计函数，将当前的样本集分为两个子样本集，使得生成的的每个非叶子节点都有两个分支。CART算法生成的决策树是结构简洁的二叉树。回归树是针对目标变量是连续性的变量，通过选取最优分割特征的某个值，然后数据根据大于或者小于这个值进行划分进行树分裂最终生成回归树。对于本项目，单一的回归树肯定是不够用的。可以利用集成学习中的boosting框架，对回归树进行改良升级，得到的新模型就是提升树（Boosting Decision Tree），在进一步，可以得到梯度提升树（Gradient Boosting Decision Tree，GBDT），再进一步可以升级到XGBoost。Boosting 是一种可以用来减小监督式学习中偏差的机器学习算法。面对的问题是迈可·肯斯（Michael Kearns）提出的：一组"弱学习者"的集合能否生成一个"强学习者"？弱学习者一般是指一个分类器，它的结果只比随机分类好一点点；强学习者指分类器的结果非常接近真值。大多数提升算法包括由迭代使用弱学习分类器组成，并将其结果加入一个最终的成强学习分类器。加入的过程中，通常根据它们的分类准确率给予不同的权重。加和弱学习者之后，数据通常会被重新加权，来强化对之前分类错误数据点的分类。

在这个项目中主要运用的是xgboost算法。xgboost内部使用的CART tree，这种结构可以处理分类回归问题。而且xgboost的特点是它能够自动利用CPU的多线程进行并行，同时在算法上加以改进提高了精度。也是一种集成方法。集成方法的有点就是采用很多弱学习器最后合并成强学习器，效果会比较好。Xgboost在学习的过程中， 或不断的通过上一次学习的模型的残差进一步学习，最终将损失缩小

2.3.2 XGBoost 介绍和原理

XGBoost 使用CART 回归树作为基学习器，基于加法模型，采用前向分布算法去学习模型，具体的算法推导如下：/2/

第i个样本在第t轮的模型预测值 $\hat{y}_i$，保留t-1 轮的模型预测值 $\hat{y}_i^{t-1}$后，加入一个新的函数 $f_t(x_i)$，尽可能地让目标函数最大程度地降低。

$$\hat{y}_i^{(0)} = 0$$

$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i)$$

...

$$\hat{y}_i^{(i)} = \sum_{k=1}^{t} f_1(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)$$

(2.1)

目标函数定义为：

$$Obj^{(t)} = L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}) + \sum_{i=1}^{t} \Omega(f_i) = \sum_{i=1}^{n} l(y_i, \hat{y}^{(t-1)} + f_t(x_i)) + \Omega(f_t) + \text{ constant}$$

(2.2)

对上式进行二阶泰勒展开可得：

$$L^{(t)} = \sum_{i=1}^{n} [l(y_i, \hat{y}^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) + \text{constant}$$

(2.3)

上式中

$$g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

$$h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

除去(2.3)中的常量可得：

$$\tilde{L}^{(t)} = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

(2.4)

其中，$f_t(x) = w_{q(x)}, w \in R^T, q : R^d \to \{1, 2, \cdots, T\}$, w为叶子权重，T为叶子数量。可以将树的复杂度定义为：

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

(2.5)

结合(2.4), (2.5) 可得：

$$\begin{aligned} L^{(t)} &\cong \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \\ &= \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \gamma T + \lambda \frac{1}{2} \sum_{j=1}^{T} w_j^2 \\ &= \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T \end{aligned}$$

(2.6)

设 $G_j = \sum_{t \in I_j} g_i$, $H_j = \sum_{t \in I_j} h_i$ 则有

$$L^{(t)} \cong \sum_{j=1}^{T} G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2 + \gamma T$$

(2.7)

假设树得结构q(x)已知, 由上式可以计算出叶子j得最优权重w_j^*为：

$$w_j^* = -\frac{G_j}{H_j + \lambda}$$

(2.8)

由上式可知，最优目标函数值为：

$$L^{(t)}(q) = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T$$

(2.9)

上式称为结构分数，可以对树得结构进行评估，分数越小树的结构越好。我们可以利用这个结构分数来列举所有的树结构，从而找到最优解。我们采用贪心算法来找到当前层次的最有结构：

$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}\right] - \gamma$$

(2.10)

其中，$\frac{G_L^2}{H_L+\lambda}$ 为左子树分数，$\frac{G_R^2}{H_R+\lambda}$ 为右子树分数，$\frac{(GL+GR)^2}{HL+HR+\lambda}$ 为不可分割时的分数。

2.3.4 技术

本项目使用目前十分流行的基于 Python 语言的 Scikit-learn 机器学习框架以及 XGBoost 和 LightGBM 官方提供的算法接口。Scikit-learn 来源于 2007 年的Google Summer of Code 项目，最初由 David Cournapeau 开发。它是一个简洁、高效的算法库，提供一系列的监督学习和无监督学习的算法，以用于数据挖掘和数据分析。Scikit-learn 的基本功能主要被分为六大部分：分类，回归，聚类，数据降维，模型选择和数据预处理。总体上来说，作为专门面向机器学习的 Python 开源框架，Scikit-learn 可以在一定范围内为开发者提供非常好的帮助。它内部实现了各种各样成熟的算法， 容易安装和使用，样例丰富，而且教程和文档也非常详细。

2.4 基准模型

本项目所采用的基准指标是在 Kaggle 上 Private LeaderBoard 的得分，具体来说，对 test.csv 中各商店的销售额进行预测，然后将预测结果提交到 Kaggle上， 利用 Kaggle 竞赛 Rossmann Store Sales 的 Private LeaderBoard 的得分作为基准， 目标是最终的得分要达到 0.117 及以下。

3. 方法

(大概 3-5 页）

3.1 数据预处理

在这一部分， 你需要清晰记录你所有必要的数据预处理步骤。在前一个部分所描述的数据的异常或特性在这一部分需要被更正和处理。需要考虑的问题有：

- *如果你选择的算法需要进行特征选取或特征变换，你对此进行记录和描述了吗？*
- ***数据的探索****这一部分中提及的异常和特性是否被更正了，对此进行记录和描述了吗？*
- *如果你认为不需要进行预处理，你解释个中原因了吗？*

3.2 执行过程

在这一部分， 你需要描述你所建立的模型在给定数据上执行过程。模型的执行过程，以及过程中遇到的困难的描述应该清晰明了地记录和描述。需要考虑的问题：

- *你所用到的算法和技术执行的方式是否清晰记录了？*
- *在运用上面所提及的技术及指标的执行过程中是否遇到了困难，是否需要作出改动来得到想要的结果？*
- *是否有需要记录解释的代码片段(例如复杂的函数）？*

3.3 完善

在这一部分，你需要描述你对原有的算法和技术完善的过程。例如调整模型的参数以达到更好的结果的过程应该有所记录。你需要记录最初和最终的模型，以及过程中有代表性意义的结果。你需要考虑的问题：

- *初始结果是否清晰记录了？*
- *完善的过程是否清晰记录了，其中使用了什么技术？*
- *完善过程中的结果以及最终结果是否清晰记录了？*

4. 结果

（大概 2-3 页）

4.1 模型的评价与验证

在这一部分，你需要对你得出的最终模型的各种技术质量进行详尽的评价。最终模型是怎么得出来的，为什么它会被选为最佳需要清晰地描述。你也需要对模型和结果可靠性作出验证分析，譬如对输入数据或环境的一些操控是否会对结果产生影响（敏感性分析 sensitivity analysis）。一些需要考虑的问题：

- *最终的模型是否合理，跟期待的结果是否一致？最后的各种参数是否合理？*
- *模型是否对于这个问题是否足够稳健可靠？训练数据或输入的一些微小的改变是否会极大影响结果？（鲁棒性）*
- *这个模型得出的结果是否可信？*

4.2 合理性分析

在这个部分，你需要利用一些统计分析，把你的最终模型得到的结果与你的前面设定的基准模型进行对比。你也分析你的最终模型和结果是否确确实实解决了你在这个项目里设定的问题。你需要考虑：

- *最终结果对比你的基准模型表现得更好还是有所逊色？*
- *你是否详尽地分析和讨论了最终结果？*
- *最终结果是不是确确实实解决了问题？*

5. 项目结论

（大概 1-2 页）

5.1 结果可视化

在这一部分，你需要用可视化的方式展示项目中需要强调的重要技术特性。至于什么形式，你可以自由把握，但需要表达出一个关于这个项目重要的结论和特点，并对此作出讨论。一些需要考虑的：

- *你是否对一个与问题，数据集，输入数据，或结果相关的，重要的技术特性进行了可视化？*

- *可视化结果是否详尽的分析讨论了？*
- *绘图的坐标轴，标题，基准面是不是清晰定义了？*

5.2 对项目的思考

在这一部分，你需要从头到尾总结一下整个问题的解决方案，讨论其中你认为有趣或困难的地方。从整体来反思一下整个项目，确保自己对整个流程是明确掌握的。需要考虑：

- *你是否详尽总结了项目的整个流程？*
- *项目里有哪些比较有意思的地方？*
- *项目里有哪些比较困难的地方？*
- *最终模型和结果是否符合你对这个问题的期望？它可以在通用的场景下解决这些类型的问题吗？*

5.3 需要作出的改进

在这一部分，你需要讨论你可以怎么样去完善你执行流程中的某一方面。例如考虑一下你的操作的方法是否可以进一步推广，泛化，有没有需要作出变更的地方。你并不需要确实作出这些改进，不过你应能够讨论这些改进可能对结果的影响，并与现有结果进行比较。一些需要考虑的问题：

- *是否可以有算法和技术层面的进一步的完善？*
- *是否有一些你了解到，但是你还没能够实践的算法和技术？*
- *如果将你最终模型作为新的基准，你认为还能有更好的解决方案吗？*

REF

/1/ https://blog.csdn.net/yinyu19950811/article/details/81079192
/2/ Tianqi Chen,Tong He. Higgs Boson Discovery with Boosted Trees[C]. JMLR: Workshop and Conference Proceedings, 2015 (42): 69-80.