Udacity 机器学习（进阶）

毕业项目报告

Rossmann 销售额预测（From Kaggle）

1. 定义

1.1. 项目概览

在众多的机器学习研究中，根据数据集中是否给出样本标签，我们可以将机器学习简单的分为监督学习和非监督学习两大类。一般的，监督学习中的数据集是由输入信息（特征）和预期输出（分类标签或连续数值标签等）组成的，机器学习算法在观察一些样本之后，对输入和输出之间的映射关系建立模型，并对任意的输入信息进行预测，非监督学习则是对仅包含特征信息的数据集进行聚类分析。

基于监督学习的学习模式和数据组成，监督学习主要可以解决两类问题：回归问题和分类问题。从机器学习广泛应用之初，监督学习在诸如数据分类、回归预测等领域中的成果十分显著。受数据数量和质量的影响，早期的监督学习应用大多集中在金融、教育领域，而随着大数据在诸多行业中逐步受到重视，监督学习在更多行业中的实用价值也越来越高。

在面临大量**特征-标签**模式的数据时，人类的经验判断法也并不是不能得到任何结果，只是这一经验模式很难被完整的量化并应对新的输入信息，从而无法得到理想的准确度和效率，而机器学习则可以将这一问题简化为类似于$AX + b = Y$（当然不止于此）这样的数学问题并进行海量的训练。因此机器学习在销售、水文地质等具有大数据背景的行业中可以极大的提高数据分析、挖掘效率，快速学习大量数据模式并进行实时响应，以此来实现指导、优化分配的任务。

在毕业项目中，我选择了 Rossmann 销售预测作为毕业题目。Rossmann 来自德国，是一家在欧洲拥有超过 3600 家分店的连锁药店。其庞大的体量在带来可观利润的同时也对销售管理、配货管理以及市场预测提出了挑战。在已经积累的历史数据中，门店销售额的影响因素众多，例如促销活动安排、商业竞争、学校假期和法定假期等等因素都会影响门店的销售额。Rossmann 需要提前六周对门店的日销售额进行分析和预测，Rossman 的数千名独立经理人根据自身经验实现的预测结果则良莠不齐，很难依据经理人的判断来指导市场调控。在本项目中，我们将使用机器学习的方法对这一问题进行研究，并形成预测。

1.2. 问题说明

在本项目中，我们需要根据 1115 家门店的共 1017209 条历史数据进行研究，并建立一个鲁棒性强的预测模型，并在提前六周的预测任务中得到一个令人可以接受的结果。这是一个非常典型的监督式回归问题。回归问题在统计学中出现的较早，人们最开始使用了一些简单的低次函数对一些现有的数据进行拟合，从而对未出现的数据点进行预测，随着统计学理论的发展，回归分析数学模型也在不断地丰富，但是回归模型仍通过人们的分析和研究而建立。而在机器学习中，基于现有的统计学理论基础，人们建立了更为抽象的高维度模型（如随机森林、SVM 等）来进行回归问题的分析，随之产生的大型特征空间的计算则借助不断发展的计算机算力实现。现在，有许多方法都可以用来解决回归问题，如决策树回归算法、SVM 回归算法、随机梯度下降回归算法等。同时，基于这些单一回归算法，越来越多的集成型和增强型算法也被提出，而且表现出了优越的性能，比如基于决策树模型的随机森林和 GBDT 算法，以及近年来在 Kaggle 中被广泛应用的 XGboost 算法。

在本项目中，我们的实施将按照以下步骤进行：

- 将首先对现有数据进行处理和整合，形成便于机器学习算法研究的数据集；
- 之后借助随机森林算法和 XGboost 算法来建立回归模型并对其进行优化设计；
- 最终预测出 41088 条未来样本的销售额信息。

1.3. 指标

在机器学习研究中，学习算法的选择以及算法参数的调试对结果的影响是极大的，但这并不意味着我们只需要完成算法相关的工作。在研究中，我们会选择数据集整体的一部分作为训练集，另一部分作为测试集。我们在训练集上应用我们的算法，而测试集则作为”期末考试”来对研究成果进行测试和评价。那么算法运行到什么程度以及我们最终结果是否理想都需要一个量化指标来体现，这也就是评价指标。不同的机器学习任务有着不同的性能评价指标。例如，在垃圾邮件检测系统中，它本身是一个二分类问题（垃圾邮件 vs 正常邮件），可以使用准确率(Accuracy)、对数损失函数(log-loss)、AUC 等评价方法。又如在股票预测中，它本身是一个实数序列数据预测问题，可以使用平方根误差(root mean square error，RMSE)等指标；又如在搜索引擎中进行与查询相关的项目排序中，可以使用精确率－召回率(precision-recall)、NDCG(normalized discounted cumulative gain)4。

在本项目中，我们使用 RMSPE 也就是 Root Mean Square Percentage Error 作为模型的评价指标。RMSPE 的计算方式如下：

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

可以看出，通过计算预测误差和实际值的比值，我们可以对预测结果有一个量化的表示，同时也避免了数值大小本身带来的误判，这也是 RMSPE 指标的优点，对数值绝对大小不敏感，更加适合于多尺度规模的序列评测。

2. 分析

2.1. 数据研究

通过 Kaggle 上提供的信息，我们掌握了一些可以用于训练模型的内容。这些信息以 csv 格式文件提供，根据监督学习的方法，我们可以将这些信息分为特征和标签两类。特征包括门店的规模、促销互动信息、节假日信息等可以于销售额对应起来又比较便于观测得到的信息，而标签则是训练数据中已有的我们期望得到的输出内容。以下是数据集中各项信息的说明：

- **Id** - 一例特定日期和特定门店的样本。
- **Store** - 各门店唯一的编号
- **Sales** - 销售额（本项目的预测内容）。
- **Customers** - 日客流量。
- **Open** - 用来表征商店开张或闭店的数据，0 表示闭店，1 表示开张。
- **StateHoliday** - 用来表征法定假期。除了少数例外，通常所有门店都会在节假日关闭。值得注意的是，所有学校在法定假期以及周末都会关闭。数据 a 表示公共假期，b 表示复活节，c 表示圣诞节，0 则意味着不是假期。
- **SchoolHoliday** - 用来表征当前样本是否被学校的关闭所影响，也可以理解为学校放假。
- **StoreType** - 使用 a,b,c,d 四个值来表征四种不同类型的商店
- **Assortment** - 表征所售商品品类的等级，a 为基础型，b 为大型，c 为特大型。
- **CompetitionDistance** - 距离最近竞争商家的距离（m）。
- **CompetitionOpenSince[Month/Year]** - 距离最近竞争商家的开业时间。
- **Promo** - 表征某天是否有促销活动。
- **Promo2** - 表征门店是否在持续推出促销活动
- **Promo2Since[Year/Week]** - 以年和年中周数表征该门店参与持续促销的时间。
- **PromoInterval** - 周期性推出促销活动的月份，例如 "Feb,May,Aug,Nov" 表示该门店在每年的 2 月 5 月 8 月和 11 月会周期性的推出促销活动。

如果想要顺利的建立机器学习模型，那么在进行计算分析之前对数据形成概览是极为重要的，我们应当清楚各个特征的取值范围和大致分布规律。在下方，我对已有的信息中的数值量进行了初步的统计整理。

训练数据特征统计数据表

	DAY OF WEEK	SALES	CUSTOMERS	OPEN	PROMO	SCHOOL HOLIDAY
COUNT	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	1.017209e+06	**1.017209E+06**
MEAN	3.998341e+00	5.773819e+03	6.331459e+02	8.301067e-01	3.815145e-01	**1.786467E-01**
STD	1.997391e+00	3.849926e+03	4.644117e+02	3.755392e-01	4.857586e-01	**3.830564E-01**
MIN	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	**0.000000E+00**
25%	2.000000e+00	3.727000e+03	4.050000e+02	1.000000e+00	0.000000e+00	**0.000000E+00**
50%	4.000000e+00	5.744000e+03	6.090000e+02	1.000000e+00	0.000000e+00	**0.000000E+00**
75%	6.000000e+00	7.856000e+03	8.370000e+02	1.000000e+00	1.000000e+00	**0.000000E+00**
MAX	**7.000000E+00**	**4.155100E+04**	**7.388000E+03**	**1.000000E+00**	**1.000000E+00**	**1.000000E+00**

测试数据特征统计数据表

	DayOfWeek	Open	Promo	SchoolHoliday
count	41088.000000	41077.000000	41088.000000	41088.000000
mean	3.979167	0.854322	0.395833	0.443487
std	2.015481	0.352787	0.489035	0.496802
min	1.000000	0.000000	0.000000	0.000000
25%	2.000000	1.000000	0.000000	0.000000
50%	4.000000	1.000000	0.000000	0.000000
75%	6.000000	1.000000	1.000000	1.000000
max	7.000000	1.000000	1.000000	1.000000

门店特征计数据表

	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear
count	1112.000000	761.000000	761.000000	1115.000000	571.000000	571.000000
mean	5404.901079	7.224704	2008.668857	0.512108	23.595447	2011.763573
std	7663.174720	3.212348	6.195983	0.500078	14.141984	1.674935
min	20.000000	1.000000	1900.000000	0.000000	1.000000	2009.000000
25%	717.500000	4.000000	2006.000000	0.000000	13.000000	2011.000000
50%	2325.000000	8.000000	2010.000000	1.000000	22.000000	2012.000000
75%	6882.500000	10.000000	2013.000000	1.000000	37.000000	2013.000000
max	75860.000000	12.000000	2015.000000	1.000000	50.000000	2015.000000

从上述表中，我们可以得到以下信息：

- 在训练数据中我们有 1017209 个样本，测试数据中有 41088 个样本。其中训练数据的数值数据没有出现缺失（非数值特征将在数据预处理部分进行进一步分析），测试数据中的门店开门情况有少量缺失，但是考虑到门店开门情况与当前日期、节假期情况密切相关，所以这一缺失对本项目的进一步分析影响不大。

- 从星期几（DayOfWeek）来看，训练数据和测试数据的分布都比较均匀，可以大致说明样本日期的分布是均匀的。

- 训练数据和测试数据的门店开门状态和学校假期分布都比较符合常识。

- 从训练数据来看，销售额和客流量的分布很相近，这是合理的。这两个数据的分布较宽，最大值离均值较远，最小值为 0 则是因为门店休假或关闭，从分布情况来看，大部分样本中的销售额和客流量在均值上下 50%左右的范围内，小部分样本较大，在数据预处理部分中应当考虑是否有异常值出现。

- 在门店特征中，我们共有 1115 条信息对应 1115 家门店。可以看出，竞争商家的距离和开业时间信息以及门店自身的周年促销活动信息均有相当一部分的

缺失，而这些内容显然对最终销售额的预测有很大的影响，在数据预处理部分，我们应当考虑这些缺失值的影响，并根据算法需求进行补充。

- 门店特征中的竞争对手距离的分布较宽，最大值和最小值均距离均值较远，在数据预处理中应当考虑进行适当的放缩。
- 竞争对手开业时间的最小值和均值的距离比较大，在数据预处理中应当考虑考虑进行适当的放缩。
- 周年性促销活动的分布相对合理，可以看到约有一半的商家参与了周年性促销活动，而且开启的时间在 2009~2015 之间，分布也相对均匀，这有利于后续的分析。

除了上表中出现的内容，数据中还有一些重要特征以非数值形式表现，这些特征主要为时间性数据或类别型数据，有：

- 样本日期：year-month-day 字符串，需要进行拆分编码，明确样本的年、月、日。
- 门店类型：a,b,c,d 表征四种类型的门店，需要进行 OHE 处理。
- 品类规模：使用 a,b,c 分别表示基础型、大型和扩充型的店铺
- 周年性促销活动的间隔点：对于推出了周年性促销活动（Promo2）的店铺，有四种间隔周期，在数据与处理中需要进行 OHE 处理。

在对数据集的整体进行概览之后，我们可以选择性的抽取一些样本进行对比和观察，这里选择了三个样本，样本信息如下：

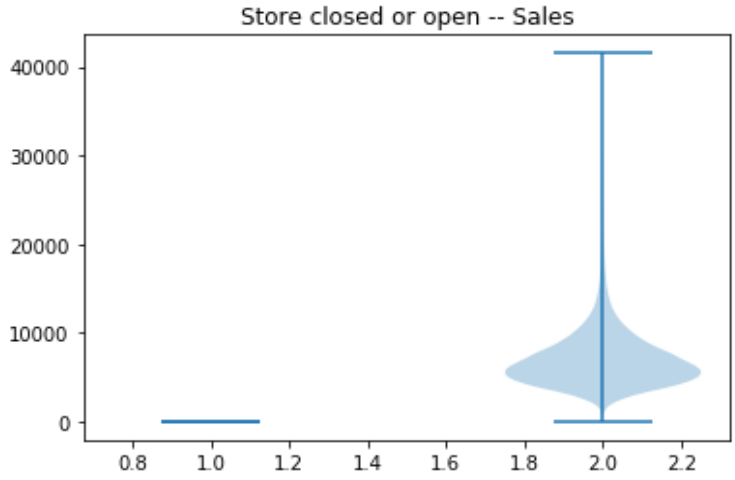
Store	**DayOfWeek**	**Date**	**Sales**	**Customers**	**Open**	**Promo**	**StateHoliday**	**SchoolHoliday**
101	5	2015-07-31	11075	915	1	1	0	1
101	5	2015-06-19	8779	829	1	1	0	0
101	5	2015-03-27	7039	723	1	0	0	0

从选择的 101 号店铺的三个样本来看，同样是星期五，没有促销活动，同时也不是法定假日和学校假日的时候销量最低，有促销活动时，销量有了一定的提高，同时拥有促销活动并且是学校假期时的销量最高。

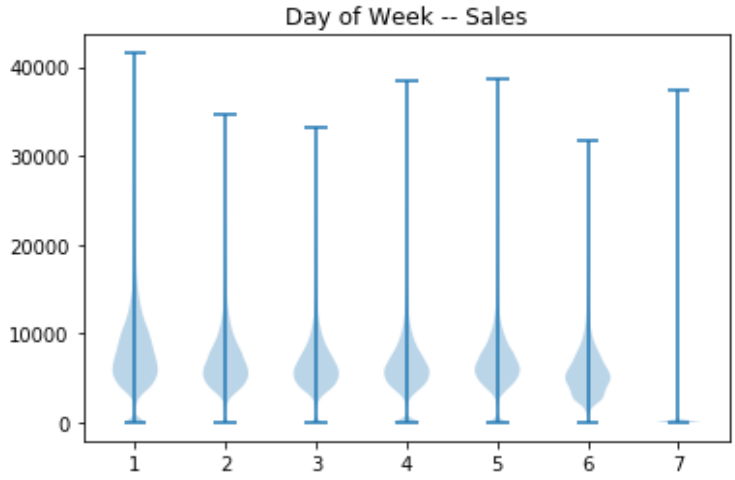
2.2. 探索性可视化

为了对数据建立一个更加直观的认识，我们需要对数据的各项特征进行探索性可视化。在这一部分，我将尝试寻找训练数据特征与销售额之间的相关性，并对其分布和关联性进行可视化分析。

门店开门与否对销售额应当有很大的影响，下图是训练数据中样本关门（左侧）或开门（右侧）情况下销售额的提琴图，可以看出，门店关门时没有产生销售额。因此，在后续讨论中，可以考虑剔除店铺关门的样本。



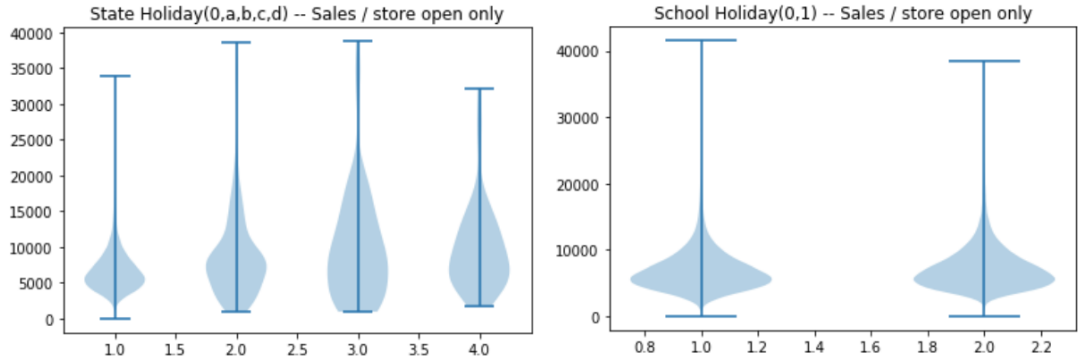
在数据集中，样本位于每周的第几天也被采集下来，下图是每周的七天对应的所有销售额的提琴图。



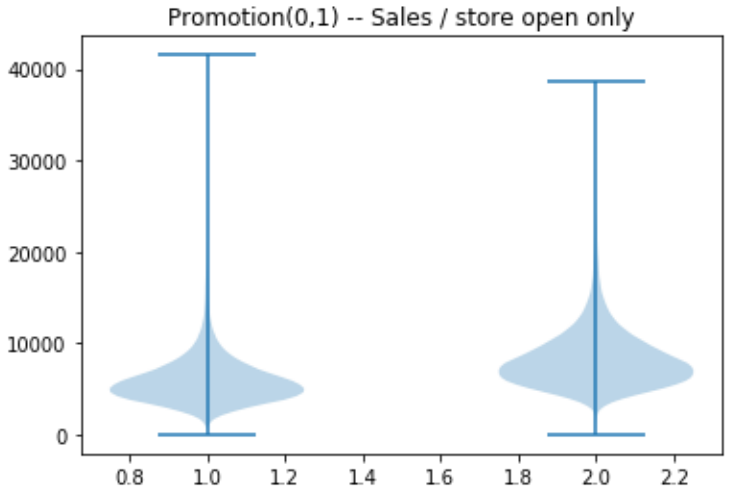
可以看出，在不考虑店铺开门与否的统计中，周一至周六的销售额分布规律相近，大量的数据集中在 10000 以下的部位，而周日的样本数据则大量聚集在 0 附近，但是最大值却处于一个较高的水平这是因为门店在周日会发生两个极端情况，

普通的周日，大多数门店会休假，而在节假日前后，部分门店会开店。因此，我又绘制了只考虑门店开门状态下的七天销售额提琴图如下，可以看到，在只考虑开门样本时，周日的销售额分布要比其余 6 天更宽。从这里也可以看出，有必要在后续分析中剔除未开门的样本，否则容易形成误导。

从直观印象来说，节假日对销售额也会产生影响，下面两张图分别是门店**开门情况下**法定假日、学校假日和销售额之间的提琴图。可以看出，法定节假日对销售额具有一定的促进作用，而学校假期则和销售额相关性不大。



对于一般商家，提高销量最常用的办法就是开展促销活动，下图表示的是是否开展促销活动和销售额的关系。可以看出，有促销活动时，销售额的统计密度曲线整体上移，说明促销活动对销售额的刺激作用还是较为明显的。



总的来看，上述各图中的统计数据分布均能找到合理的解释，但是可以看出，所有提琴图中最大值均较突出，应当在数据处理中考虑异常值的筛选和剔除。

2.3. 算法与方法

针对本项目的回归问题，我选定随机森林法和 XGboost 法这两种非常常用又高效的算法来开展预测。在数据研究和探索性可视化部分，我们已经注意到现有数据的特征需要进行编码，数据集中也可能存在一些异常值需要剔除。同时，一些重要的特征出现了空值需要填充。在本项目中，我主要使用 OHE（独立热编码）来对大部分非数值特征进行数值化，一些数值特征则需要进一步的放缩操作。而对于空值填充的部分，则需要建立必要的填充策略。本项目所运用的算法和方法细节整理如下。

2.3.1. 随机森林算法[1]

在机器学习中，随机森林是一个包含多个决策树的分类器，并且其输出的类别是由个别树输出的类别的众数而定。 Leo Breiman 和 Adele Cutler 发展出推论出随机森林的算法。而"Random Forests"是他们的商标。这个术语是 1995 年由贝尔实验室的 Tin Kam Ho 所提出的随机决策森林（random decision forests）而来的。这个方法则是结合 Breimans 的"Bootstrap aggregating"想法和 Ho 的"random subspace method" 以建造决策树的集合。算法的实现步骤如下：

1. 用 N 来表示训练用例（样本）的个数，M 表示特征数目。
2. 输入特征数目 m，用于确定决策树上一个节点的决策结果；其中 m 应远小于 M。
3. 从 N 个训练用例（样本）中以有放回抽样的方式，取样 N 次，形成一个训练集（即 bootstrap 取样），并用未抽到的用例（样本）作预测，评估其误差。

4. 对于每一个节点，随机选择 m 个特征，决策树上每个节点的决定都是基于这些特征确定的。根据这 m 个特征，计算其最佳的分裂方式。

5. 每棵树都会完整成长而不会剪枝（Pruning，这有可能在建完一棵正常树状分类器后会被采用）。

对于随机森林模型，实施过程中有多种参数可以进行调整，主要参数和其对模型的影响意义如下[2][3][4]：

1. n_estimators：随机森林中的决策树数目，默认为 10。

2. max_features：最大特征数，可以用整数形式考虑确定数目的特征，用浮点数形式来考虑确定比例的特征，也可以使用“log2”，“sqrt”来指定一种计算考虑特征数目的方法。

3. max_depth：决策树最大深度，可以指定一个整数来限制决策树建立过程中子数的深度。

4. min_samples_split：内部节点再划分所需最小样本数，这个值限制了子树继续划分的条件，如果某节点的样本数少于 min_samples_split，则不会继续再尝试选择最优特征来进行划分。

5. min_samples_leaf：叶子节点最少样本数:，这个值限制了叶子节点最少的样本数，如果某叶子节点数目小于样本数，则会和兄弟节点一起被剪枝。 默认是 1, 可以输入最少的样本数的整数，或者最少样本数占样本总数的百分比。

6. min_weight_fraction_leaf：叶子节点最小的样本权重和，这个值限制了叶子节点所有样本权重和的最小值，如果小于这个值，则会和兄弟节点一起被剪枝。默认是 0，就是不考虑权重问题。

7. max_leaf_nodes：最大叶子节点数，通过限制最大叶子节点数，可以防止过拟合，默认是"None”，即不限制最大的叶子节点数。如果加了限制，算法会建立在最大叶子节点数内最优的决策树。

8. min_impurity_split：节点划分最小不纯度: 这个值限制了决策树的增长，如果某节点的不纯度(基于基尼系数，均方差)小于这个阈值，则该节点不再生成子节点。即为叶子节点 。

在机器学习应用中，调优过程中的重要参数包括树的数目 n_estimators，最大特征数 max_features， 最大深度 max_depth， 内部节点再划分所需最小样本数 min_samples_split 和叶子节点最少样本数 min_samples_leaf。

2.3.2. XGboost 算法

XGBoost 的全称为 eXtreme Gradient Boosting，是 GBDT 的一种高效实现，XGBoost 中的基学习器除了可以是 CART（gbtree）也可以是线性分类器（gblinear）。XGboost 的前身 GBDT(Gradient Boosting Decision Tree) 又叫 MART

（Multiple Additive Regression Tree)，是一种迭代的决策树算法，该算法由多棵决策树组成，所有树的结论累加起来做最终答案。它在被提出之初就和 SVM 一起被认为是泛化能力（generalization)较强的算法。GBDT 的核心在于，每一棵树学的是之前所有树结论和的残差，这个残差就是一个加预测值后能得真实值的累加量。与随机森林不同，随机森林采用多数投票输出结果；而 GBDT 则是将所有结果累加起来，或者加权累加起来[5]。

在 xgboost 模型运行过程中，需要设置三种类型的参数：常规参数, 提升器参数和任务参数[6]。其中常规参数用来指定模型使用的 booster 以及模型的计算资源分配等，提升器参数的探索在相似研究中占据较大的比重，这部分用来指定 tree、dart 以及 linear 三种 booster 的参数，对于本项目中的问题，tree booster 较为适用。任务参数则是用来适应不同的具体问题，如指定 regression 或 classification，指定不同的评价方法等。

对于本项目，通过阅读类似问题的 demo[7]我选取了以下几项参数作为主要优化目标。

1. eta: 学习率，我理解为步长控制参数，常见案例的取值范围约为 0.01 到 0.3 之间；
2. max_depth：树最大深度，这是一个经典的决策树模型参数，在随机森林算法中也是极为重要的一个参数。默认值为 6，我计划首先使用默认值进行计算，然后对 4~11 范围内的值进行优化搜索。

对于其他参数，由于对 xgboost 模型的了解并不深入，因此直接取用同类问题中的参考值或根据参数意义进行假定。

1. min_child_weight、gama、subsample 这三个值可以调整模型的保守和激进取向，为了避免模型出现严重的过拟合，在优化过程中，我将分别取 50、2、和 0.8 来让模型稍微偏向保守。
2. alpha 参数可以对模型进行一定的加速，默认值为 1，我计划取用 2 来提高计算效率。
3. tree_method：在较新版本的 xgboost 中提供了 gpu 加速的功能，在这里我选用了快速的 gpu 计算方式“gpu_hist”来保证模型的效率。

2.3.3. Tukey 异常截断法

Tukey 异常截断法是在统计学中广泛应用的一种及其简单的异常值剔除方法[8]，也是统计图表中箱型图的绘制基础[9]。首先，对数据依据大小进行排列，那么第一四分位数、中位数和第三四分位数可以通过 25%、50%、75%位置的数值来得到，我们称第一四分位数为Q_1第三四分位数为Q_3，二者之间的差值则称为 IQR，通过这三个参数，我们便可以指定异常值的分割区域如下式，$Limit_{upper}$和$Limit_{lower}$之间的区域为内限（正常数据），其外成为外限（异常数据）。

$$Limit_{upper} = Q_3 + k \times IQR$$

$$Limit_{lower} = Q_1 - k \times IQR$$

式中，k 为控制内限宽度的参数，k 值越大，异常值的截断规则越宽松，反之，k 值越小，内外限的截断规则越严格。一般情况下，k 取 1.5 可以平衡数据的完整性和合理性。

2.3.4. OHE 编码方法

One Hot Encoding（OHE），又称独立热编码或者一位有效编码，是一种处理非连续特征的有效办法。设有数据集 A，对于 n 个取值的非连续特征（例如分类特征），传统的方法是将其转换为一维序列的数值特征，OHE 则是使用多位的状态寄存器来对不同状态进行编码，并且保证任意样本的特征的有且只有一位有效。一个示例数据集非连续特征的序列化和 OHE 处理结果如下表所示。

样本号	原始特征 F	序列化	OHE
1	蓝色	0	[1,0,0]
2	黄色	1	[0,1,0]
3	红色	2	[0,0,1]

序列化法得到的连续数值可以直观的表现出不同样本之间的特征区别，但是处理后的特征在数值上并不平等，会影响最终的结果。相比于序列化的处理方法，OHE 处理方法拥有以下优势：

（1）通过 OHE 处理，将离散的样本特征扩展到欧式空间，离散特征的某个取值就对应欧式空间的一个点，这样便于机器学习算法的计算；

（2）通过 OHE 处理，样本的特征维度得到的扩充，在进行决策分析以及向量化时有利于增加分析深度；

（3）序列化处理后的特征虽然可以准确的表明各个状态之间的区别，但是序列的数值但是其数值结果会影响同一特征不同状态在算法中的权重，而实际上不同状态是平等的。经过 One Hot Encoding 处理转化为 n 个二元特征，且每个样本只有一个二元特征激活。

2.3.5. 特征缩放方法

Feature scaling（特征缩放）是用来统一数据集中的自变项或者特征范围的方法。在原始数据中，各个特征变量的变化范围可能并不相同，因此，如果不进行特征缩放处理，机器学习算法会弱化变化范围较小的特征变量的影响力，最终结果就容易被变化范围大的特征变量左右。同时，通过对原始数据进行特征缩放操作，可

以优化计算效率。例如对于梯度下降算法，特征缩放可以加速结果的收敛；对于支持向量机方法，特征缩放可以加速寻找支持向量。

常用的特征缩放方法有以下四种：

1. 重新缩放法，将特征变量重新缩放至[0,1]范围内。过程示例如下：
$$x' = [x - \min(x)]/[\max(x) - \min(x)]$$
2. 正态平均法，将特征变量重新缩放至[-1,1]范围内。过程示例如下：
$$x' = [x - \text{mean}(x)]/[\max(x) - \min(x)]$$
3. 标准化法，将特征变量转化为均值等于 0，标准差为 1 的一组数据。即：
$$x' = (x - \bar{x})/\sigma$$
4. 单位向量法，对于向量化的特征变量，一般都会使用单位向量法来消除距离的影响，即：
$$\boldsymbol{x}' = \boldsymbol{x}/\|\boldsymbol{x}\|$$

2.3.6.空值填充方法

对于数据集中出现的空值，常用的填充方法有均值填充和 0 值填充，在数据集较为完善的情况下，也可以根据空值特征与其他特征的关系进行相关性填充，或者根据该特征的统计结果建立相应的填充策略。

2.4. 基准测试

为了检验我的预测模型是否达标，应当建立一个基准模型，通过是否满足基准指标来判断我有没有完成一个较为合理的预测。很方便的是，本项目在 Kaggle 上的竞赛已经完成，Kaggle 通过测试集 RMPSE 的大小对参赛个人和团队进行了排名（Rossmann Store Sales Leaderboard）。从排名表中，我们可以看到，共有 3303 名参赛队伍进行了预测，结果最好的 Gert 得到了 0.10021（越低越好）的最终得分，排除一些较为离谱的结果，最低得分是 1.0 分。同时，Kaggle 根据参赛团队的结果设定值为 0.24213Median DayofWeek Benchmark，我会将这个值作为我的第一个初步评价基准，用来在进行算法比选阶段进行评价。根据 udacity 的要求，我将以 leaderboard 的 top 10%作为基准，也就是对于测试集的评分达到小于等于 0.11773。

3. 方法

3.1. 数据预处理

在机器学习任务中，数据的预处理非常重要，通过前面的工作，我制定了本项目数据预处理的计划流程如下：

1. 训练数据清洗，剔除无用样本和有异常值的样本。

2. 整理无需处理或只需简单编码的特征

3. 时间特征的表示

4. 已有特征的重新整合

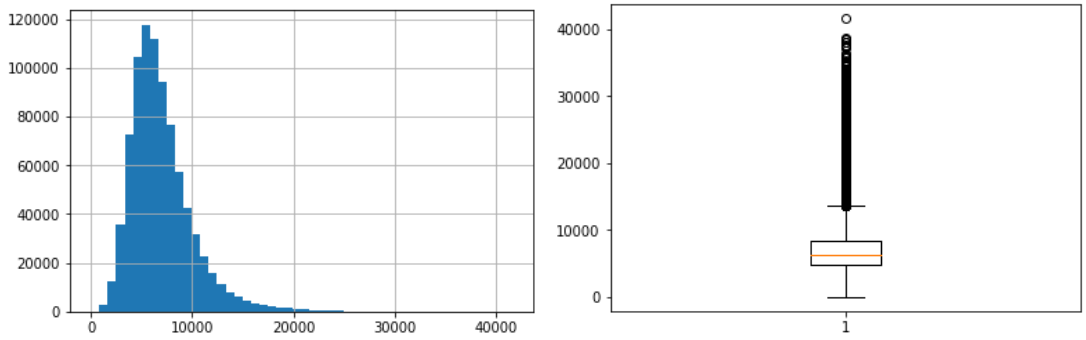
5. 数据缩放

6. 数据填充

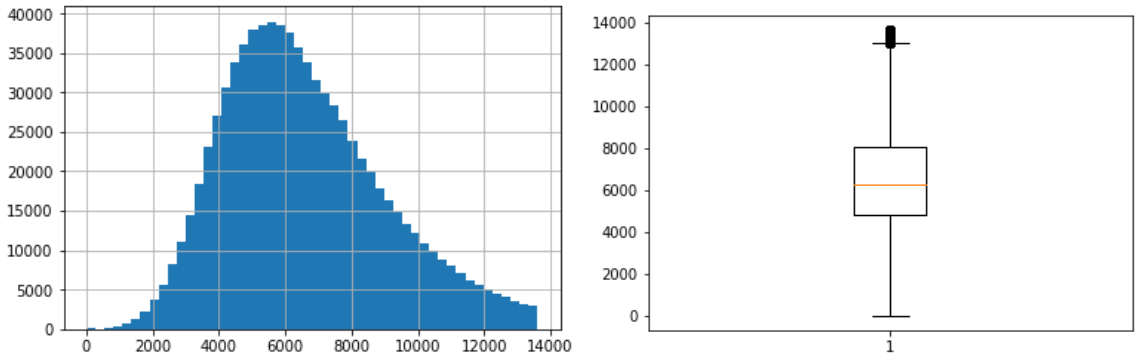
3.1.1. 训练数据清洗

首先，本项目的最终目的是预测销售额，而在前面的部分我们已经验证得出：当店铺未开门时，销售额为 0。因此，未开门的样本是不需要机器学习预测的，因此对未开门样本进行剔除。删除为开门样本后，训练数据大小从 1017209 条样本减小为 844392 条样本，减少了 17%，但是这对我们的预测工作并没有任何影响。

同时，数据集中还有可能存在一些异常点，首先，我们绘制销售额直方图和箱型图如下。



可以看出，销售额的整体分布呈现为左偏态分布，而且右侧数据的量很少。在箱型图中，有部分点处于异常值截断点之外。这里的异常值截断点依照 Tukey method 取得，内外限的控制参数为$k = 1.5$。 将超出阶段点之外的样本剔除后，练集大小从 844392 条样本减小为 813623 条样本，减少了 3.6%。



但是，在我仔细查看各门店样本时发现，按照上述步骤剔除“异常值”后，个别门店的样本数量急剧减少，例如 262 号店铺，仅剩余 2 个样本，这显然是不利于我们的训练的，因此，我认为通过数据放缩的手段来使数据值分布的更为集中应该是更适合的办法。在本项目中，不对异常值进行剔除。

3.1.2. 整理无需处理或只需简单编码的特征

通过对已有信息的观察，我们可以整理出以下无需处理可以直接使用的特征值：“Store, Open, Promo, SchoolHoliday, Customers, Sales, CompetitionDistance, Promo2”。这些特征不需要编码，可以直接放入训练集中。

数据中还有一些需要简单编码的分类特征如：“StateHoliday, StoreType, Assortment, PromoInterval”。这里采用 OHE 方法进行编码，将多分类特征转为二分类（0，1）特征。

3.1.3. 时间特征的表示

经过第上述步骤的处理，我们剩下的都是时间特征，其中包含三种时间参数。

- 样本日期
  - Day of week
  - Year/month/day
- 竞争商家开业日期
  - Month
  - Year
- 周年活动时间信息
  - since week
  - since year
  - Periodic node：{'Feb,May,Aug,Nov', 'Jan,Apr,Jul,Oct', 'Mar,Jun,Sept,Dec', nan}

这些信息虽然也可以通过编码方式处理，但是我认为合理的促销活动对销售额的刺激应该是非常重要的。而且，原始数据中提供的消息虽然很准确，但是并没有很好的把这些信息联系起来。我想要得到更加直观的特征，因此，我认为有必要对这些时间特征进行进一步的转化和整理。

首先是样本日期的拆分，样本日期代表的是当前的时间，不难得知，销售额随着时间体现出了一定的周期性，但是更为重要的是，样本时间所代表的节点，比如现在是竞争商家刚开业一周，或者现在是促销活动的第 2 周等等，这些因素对结果也是有一定影响的。因此我认为拆分后的样本日期应当可以很方便的和其他时间参数进行对比、计算。所以我希望得到样本日期的周数和月份。为此，我借助

datetime 库来将字符串格式的日期（如 2015-07-31）转化为(year, month, day, weekofyear, dayoweek)五个时间特征。

数据集中给出了竞争商家开业的 Year 和 Month，我认为有必要将这一特征转化为与样本日期的差值（以月为单位）。转化后将出现三种情况：

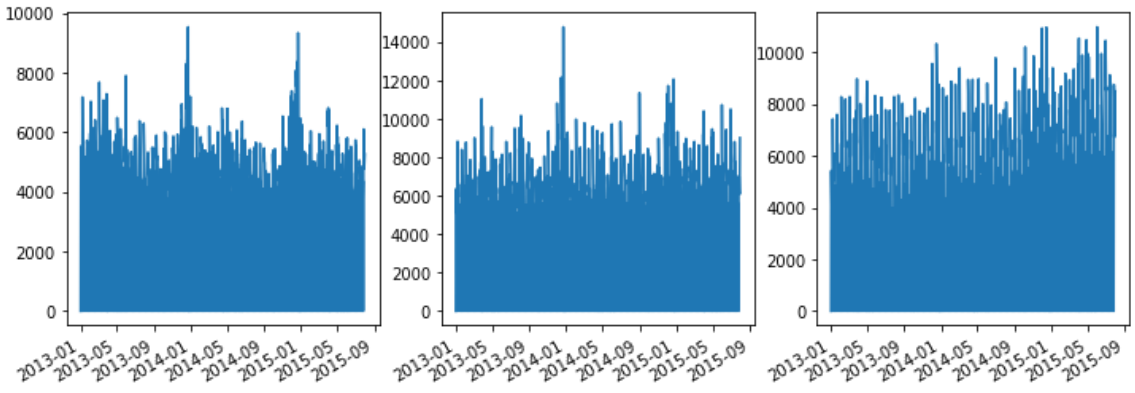
- 样本日期在竞争商家开业之前，差值为负数
- 样本日期在竞争商家开业之后，差值为正数
- 该样本所对应的门店缺失竞争商家数据。

同样的，周年促销信息包含的 year 和 week of year 也可以通过相同的办法转化为样本日期距离周年活动开始的那个时间。

3.1.4. 已有特征的重新整合

经过整理，我们的训练集已经丰富了很多，同样的处理步骤基本上可以套用在测试集上，但是有一个例外。可以发现，测试集并没有客流量属性，所以我们无法在训练集中直接使用这一信息，但是客流量是一个极好的特性，它和销售额的线性相关性极好，因此我决定增加一个购买热度特征（Heat）以表示人均产生销售额，同时再将平均客流量作为店铺特征添加到训练集中。

而通过下图中三家不同店铺 2013 年~2015 年的销售额-时间统计数据，我们可以看到，销售额具有一定的周期性，而这一变化规律并不时所有门店统一的，例如下图中第一家店铺呈现出逐年递减的趋势，第二家店铺则变化不大，而第三家店铺则呈现出逐年递增的趋势。因此，有必要计算三年内的年平均值来作为扩充特征。即 Heat2013、2014、2015 和 sales2013、2014、2015。在整合完成后，删除 Customers 特征。



三家不同店铺 2013 年~2015 年的销售额-时间统计数据

3.1.5. 数据填充

在最初的数据研究和探索性可视化中，我们已经得知数据集在年度促销活动信息等特征上有大量的缺失值，在经过了数据清洗和特征处理后，我们面对的可能室更多的缺失值，因此，有必要研究缺失值补全的策略。

首先，我们通过 pandas 的内建函数‘isnull()’来检查各个数据的缺失情况。

Features	Train	Test
Open	--	11
CompetitionDistance	2140	96
CompetitionTime	263454	15216
is_promo2_a	415042	17232
is_promo2_b	415042	17232
is_promo2_c	415042	17232
Promo2Time	415042	17232

从上面的输出结果可以看出，训练集和测试集均有样本缺失以下特征：

- 竞争商家信息
  - CompetitionDistance（距离）
  - CompetitionTime （时间）
- 周年性促销信息
  - is_promo2_a（活动类型）
  - is_promo2_b（活动类型）
  - is_promo2_c （活动类型）
  - Promo2Time （活动时间）

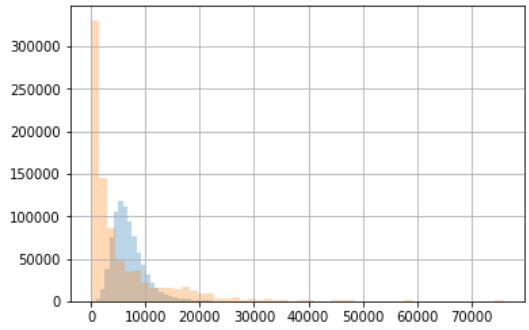
同时，测试集额外缺失 Open 特征。针对这三种缺失，我将分别制定填充策略。

对于竞争商家信息，我计划采用均值填充；对于周年促销活动信息缺失值填充。缺失的特征中，周年促销活动的类型是确定的（无活动），所以类型特征均应为 0 值。如果按照同样的思路将促销活动距今的时间也设为 0，那么将会混淆没有促销和促销刚开始这两种情况，完备样本中 promo2time 这一特征的值为正值（以月为单位），因此应对缺失值，我将使用该特征下所有非正数样本的均值进行填充。

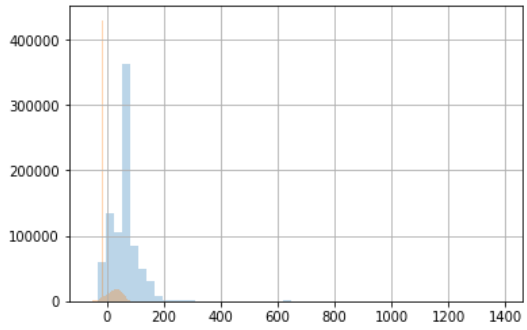
而对于测试集中 Open 信息的缺失，我无法通过均值或者固定值进行填充，但是通过日常生活的常识和对数据集的观察，我认为我计划使用一项小的机器学习任务来补全缺失的数据。我认为'Store', 'Promo', 'is_state_holiday', 'SchoolHoliday', 'DayOfWeek'这些特征和 Open 之间的关联度比较高。所以，我决定使用机器学习而不是人工判断的方法来建立缺失值的判断策略。算法我选择使用高效简单的随机森林算法。通过训练集中的选定特征进行训练和验证，在测试集中的非缺失样本上进行测试，如果效果满意，则使用训练好的模型预测出测试集样本的 Open 特征缺失值。通过试算，模型可以达到 0.997 以上的准确率（最大值为 1），最终的 Open 特征缺失值便由该模型进行确定。

3.1.6. 数据放缩

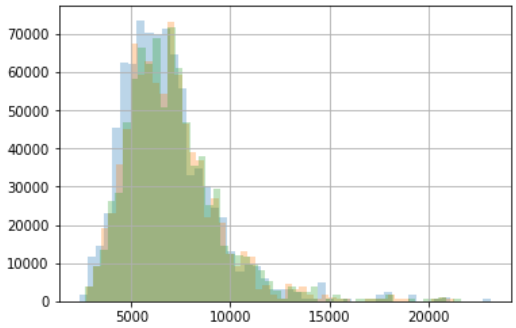
在多个特征中，`Sales，CompetitionDistance，CompetitionTime，Promo2Time，Sales2013/14/15，Heat2013/14/15` 是取值范围比较宽的。我将通过观察他们的分布情况来决定是否需要缩放和使用什么方式来缩放。根据上述特征的类型和取值特点，我将它们分组后绘制了分布直方图如下。



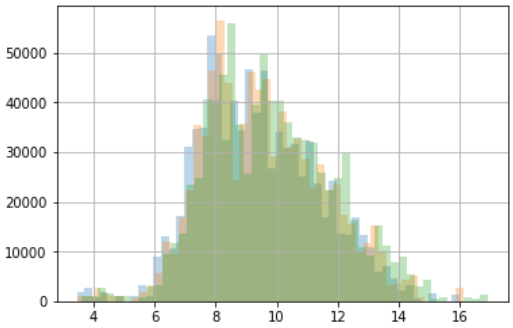
Sales and CompetitionDistance



CompetitionTime and Promo2Time



'Sales2013','Sales2014','Sales2015'



'Heat2013','Heat2014','Heat2015'

经过分布图的观察，我认为 `Sales，CompetitionDistance，CompetitionTime，Promo2Time，Sales2013/14/15` 这些特征需要进行值缩小。考虑到训练集和测试集的统一，我认为取对数处理(如下式 1)是较为合理的。但是 Promo2Time 的值中有小于 1 的值，因此对于 Promo2Time，需要首先所有值加上 100 再取对数(如下式 2)。

$$Value_{new} = \lg(Value_{old}) \quad (1)$$

$$Value_{new} = \lg(Value_{old} + 100) \quad (2)$$

3.1.7. 小结

通过数据预处理，我对原始数据的无用样本进行了剔除，考虑了是否进行异常值剔除，对无法直接使用的特征进行了编码处理，对样本时间字符进行了拆分并重新整合了样本时间、竞争商家开业时间以及周年促销开始时间形成了新的时间特征，对现有的销量信息进行了重新整合，得到了可以描述店铺年均销售额和年均销售热度的新特征。最后，我对数据集中的缺失信息制定了不同的填充策略进行了补全，并对一些数值跨度较大，整体分布较宽的信息进行了放缩。

通过本章的多个步骤，我得到了可以适用于选定算法的训练集（cesar_train.csv）和测试集（cesar_test.csv）用于之后的算法实施和优化。

3.2. 实施

在本项目中，算法的实施分为三个步骤：1. 采用未优化的模型（使用默认参数或常用参数）对数据进行训练和预测；2. 探讨模型参数对预测结果准确性的影响，并采用网格搜索法对模型参数进行调优；3. 尝试整合随机森林和 XGBoost 这两种算法，以得到更好的结果。

3.2.1. 随机森林算法应用

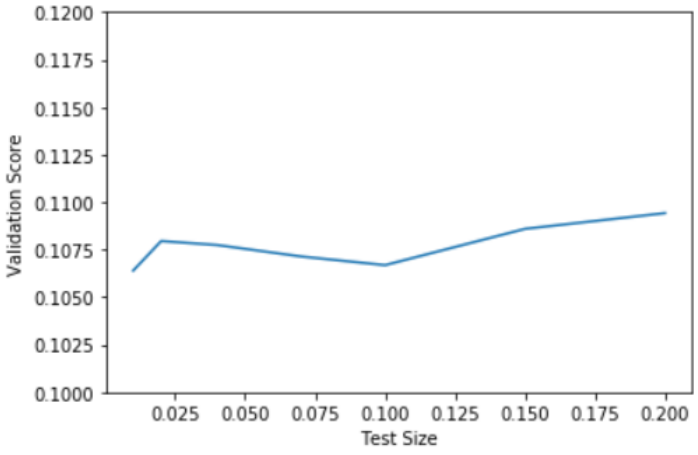
1.**alpha**

Scikit learn 库中的 RandomForest Regressor 在不指定参数的情况下，对各项模型参数赋予了默认值，在这里，不对参数进行任何调整，直接使用默认值对 3.1 章节中得到的数据集进行训练和预测。将此步骤得到的模型称为 alpha。

```
# 读入数据集
cesar_train = pd.read_csv('data/cesar_train.csv',index_col=False)
cesar_test = pd.read_csv('data/cesar_test.csv',index_col=False)
X = cesar_train.drop(['Sales'],axis=1)
y = cesar_train.Sales
# 建立最基础的随机森林回归器
RF_alpha = RandomForestRegressor()
# 将训练集进行再划分
X_train, X_test, y_train, y_test = train_test_split(X, y)
# 进行训练和评价
RF_alpha = RF_alpha.fit(X_train,y_train)
y_pred = RF_alpha.predict(X_test)
print(RMSPE(y_test,y_pred))
```

alpha 在验证集得到的评分为 0.111，当然，这个评分并不能直接描述默认参数下的随机森林模型的性能，但是这可以说明随机森林模型对本项目中要解决的问题有良好的适用性。

考虑到后续计算中必然需要对训练集进行再划分和交叉验证，我简单的测试了一下验证比例对验证集评分的影响，测试的范围为 0.01 至 0.2。结果如下图所示。可以看出，在 0.01 到 0.2 区间内 test_size 对模型评分的影响并没有很大，整体上来说 0.01 到 0.1 之间的 test_size 表现的都比较稳定，后续计算中需要对数据集进行再划分时可以参照这一取值范围。



默认参数下随机森林模型 test_size 对验证集评分的影响

2. beta

随机森林算法中涉及的参数有很多，通过之前项目的学习，我计划对 n_estimators，max_features 以及 max_depth 这三个主要参数进行优化搜索。各个参数的取值范围及取值理由如下：

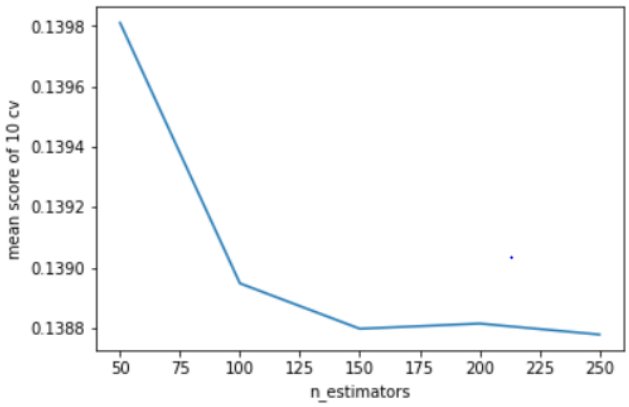
- n_estimators: 初步选取 50,80,100,150,200 进行试算，100 为大部分文献中的取值。
- max_features：初步选取 0.1,0.2,0.4,0.6,0.8,1 进行试算，该值确定的是决策树的特征量，我采用了比例设定的方式覆盖了多数文献中使用的比例范围。
- max_depth: 初步选择 2 到 8 进行试算。该范围参考了课程中波士顿房价项目中使用的范围，在后续计算中这一范围扩大至 2 到 40。

最初准备使用网格搜索法对上述所有参数进行综合搜索。但是经过尝试和思考，认为直接进行网格搜索有两个不足：

- 对于上述三个参数，现在的取值范围都是根据类似研究中的取值来确立的，不能保证在既定范围内存在理想的较优值，也无法获知每一项参数变动对结果准确性的影响；
- 网格搜索的计算量庞大，而暂时没有找到容易上手的随机森林 gpu 加速算法，虽然通过 cpu 多线程（ ×12 ）并行可以对计算过程提速，但是个人计算机的内存不足以支撑多个模型并行计算，只能采用单线程，因此计算的时间成本极高。

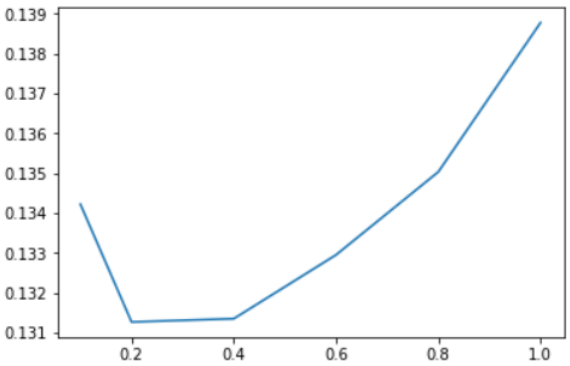
因此，计划首先对各个参数进行分项计算，得出每一项参数的较优值，然后再重新划定合理的范围进行网格搜索优化。每次测试中均采用 cv=10 的交叉验证。

首先对 n_estimators 的取值进行了测试，将不同 n_estimators 值及其对应的 10 次交叉验证评分平均值绘制在下图中，可以看出，随着 n estimators 数量的增加，模型的得分均值也在增加，当 n estimators 大于 100 时，曲线的变化逐渐变缓，为了兼顾计算经济性和准确性，我选择 150 作为一个较优的 n_estimators 值。



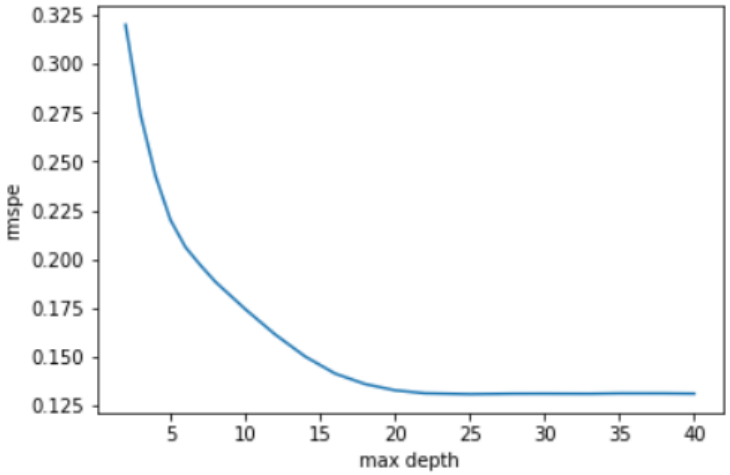
N_estimators 对评分均值的影响

同样的，对 max feature 的变化进行了测试，在测试中继承了上一步的较优 n_estimators 值，测试的结果如下，可以看出，max feature 参数为 0.2 时结果明显优于其他值，因此可以取 0.2 作为一个较优值。



N_estimators 对评分均值的影响

最后，对 max_depth 参数变化的影响进行了测试，在测试中 n_estimators 和 max feature 的取值分别为 150，0.2。最初计划测试的参数范围是 2 到 8，但是在计算过程中发现模型的评分在这一范围内未达到明显的收敛，逐步增加范围值 40 并将测试结果绘制为下图，可以看出，随着最大深度的增加，交叉验证结果的最大值、均值和最小值趋势一致，均为先快速降低，最大深度大于 22 后则区域稳定，在最大深度大于 25 后出现少量的反弹。因此，综合结果准确性和计算经济性，选择 22 最为较优的最大深度。



Max depth 对评分平均值的影响

3.cherry

通过以上三个步骤的试算，可以初步认为，三项参数的较优值分别为：

- n_estimators:150

•max_features:0.2

•max_depth:22

现在在上面三个值的基础上，给予每个参数一定的变化范围，并进行网格搜索。网格搜索的过程中，由于需要大量的计算（2310 次 fit），个人电脑运行时间过长，内存资源也不足以支撑计算，故使用腾讯云按量计费主机（56vCPU，455GB 内存）进行计算，将网格搜索部分代码整理为单独脚本 gcv-56core.py 将计算结果保存为 gcv.csv 文件。

通过分析计算结果中验证集和训练集的评分，可以看出，max depth 和 max features 在验证集和训练集上的最优值一致，分别为 24 和 0.19。n_estimators 的最优质在测试机和训练集上有差异，但是差异不大（165，175），以测试集结果为准，取 165。

4.danny

通过对三项参数进行单变量分析和网格搜索分析，可以确定随机森林的模型参数为：

•n_estimators:165

•max_features:0.19

•max_depth:24

建立随机森林模型 RF_danny 并以上述数值作为模型参数，将模型 danny 应用到整个训练集 cesar_train 上进行训练，并将预测结果整理为 submission 格式。

提交至 kaggle 后得到的评分如下。

3.2.2. XGBoost 算法应用

首先需要说明 XGBoost 的安装，我的项目主要在 windows 平台上完成，而具有 CUDA 支持的 XGBoost 在 windows 上的编译过程较为繁琐，因此我选择使用编译好的 dll 文件进行安装，这极大的简化了安装过程，只需准备 https://github.com/dmlc/xgboost 的 python 包以及来自 http://www.picnet.com.au/blogs/guido/2016/09/22/xgboost-windows-x64-binaries-for-download/的最新 dll 文件即可完成。

1. Aaron

在准备好了需要的环境后，我首先建立了简单的 xgboost 模型对 99%的训练数据进行训练并使用 1%的数据进行验证。

```
1.  # 建立简单参数的 xgboost 模型
2.
3.  param_aaron = {'objective':'reg:linear',
4.                 'eta':0.1,
```

```
                'booster':'gbtree',
                'predictor':'gpu_predictor',
                'tree_method':'gpu_hist',
               }

rounds = 10000
watchlist = [(dtest, 'eval'), (dtrain, 'train')]

time1 = time.time()
bst_aaron = xgb.train(param_aaron,dtrain,rounds,watchlist,verbose_eval=int(round
s/10),feval=rmspe_xg)
print((time.time()-time1)/60,'minutes')

submit_xgb(bst_aaron,cesar_test,'xgb_arron_submission')
```

通过 10000 轮计算，验证集的评分达到了 0.100759，训练集评分则为 0.082399，首先这说明 xgboost 模型对于该问题的适应性良好，特征工程的处理方式也较为合理，验证集评分已经明显优于随机森林法。

2. Blake

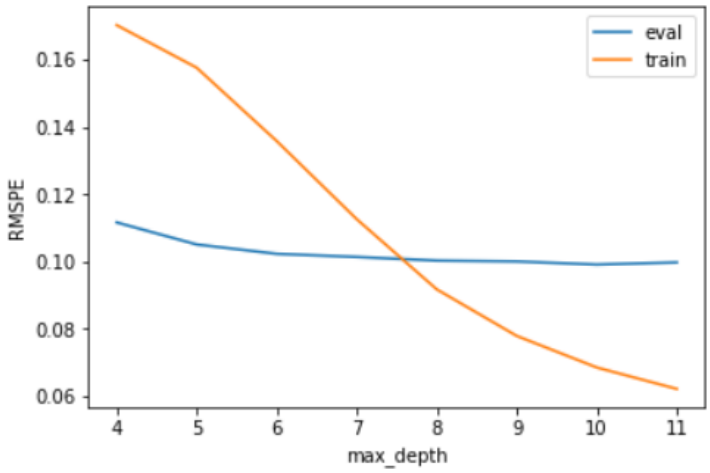
为了避免模型的过拟合现象，我增加了 min_child_weight、gama、subsample 三个参数使模型更加保守，同时也适当的将 alpha 值从默认的 1 设置为 2 以提高模型运算效率。调整过的模型参数如下。

```
param_blake = {'objective':'reg:linear',
                'eta':0.1,
                'booster':'gbtree',
                'predictor':'gpu_predictor',
                'tree_method':'gpu_hist',
                'alpha':2,
                'subsample':0.8,
                'gama':2,
                'min_child_weight':50
               }
```

经过同样 10000 次计算，blake 的验证集评分为 0.102778，训练集评分为 0.134474，模型的过拟合现象有了一定 的改善但是其表现并没有显著提高。

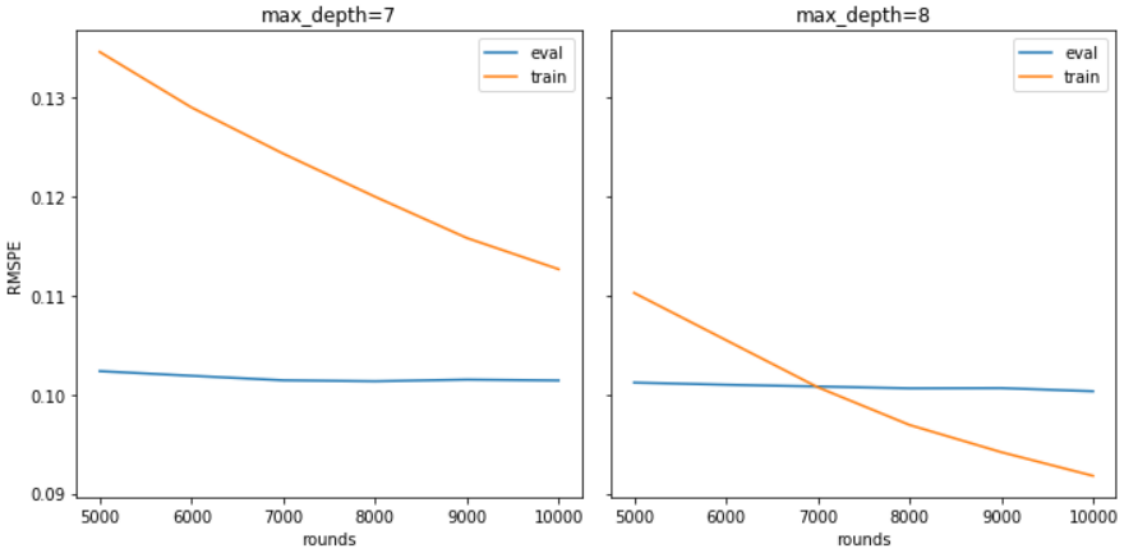
3. Further Blake

为了进一步优化模型，需要对更多的参数进行优化搜索，但是考虑到 Blake 模型的表现已经比较优异，因此仅选取 max_depth 参数进行搜索，搜素范围在 4~11 之间。通过多次尝试，得到最大深度和模型评分之间的关系如下图，可以看出，随着最大深度的不断增加，训练集的评分在 4~8 范围内快速变优，之后稍微变缓，而测试集评分的变化相对要小许多，为了保证模型的鲁棒性，取 max_depth 为 7 和 8 时的结果为较优结果。



最大深度对模型评分的影响

将 max_depth=7 和 8 时的模型评分演变曲线绘制为下图，可以看出 maxd_depth=7 时模型在 10000 次计算后训练集评分和测试集评分均保留有继续变优的趋势而且训练集评分差于验证集评分，因此可以考虑进行更多 round 的计算。而在 max_depth=8 时，模型在 7000 次训练之后训练集评分已经差于验证集评分，因此可以去除 7000 次之后的计算。



Max_depth 取 7 和 8 时模型评分随计算步数的变化规律

4. Carol

通过 Further Blake 的计算，可以看出，其实现阶段的结果已经达到了一个比较好的程度，在 max_depth 为 7 或 8 时的结果都较为优秀。因此不再考虑进行更深入的参数优化，而是尝试将两个模型进行优化结合。在本部分中保持其他参数固定，将 max_depth=7 的模型进行 13000 次计算以优化结果，而 max_depth=8 的模

型将进行 7000 次计算以缓解过拟合现象，在计算完成之后，平均两个模型的结果作为最终结果，训练过程中均采用完整的训练集。

3.2.3. 更多尝试

在 3.2.2.的末尾，Carol 模型其实是两个 xgboost 模型的整合，这已经和之前使用到的单个模型有一定的区别，那么更进一步的，在这里我将尝试代入一些随机森林的思路。

在随机森林模型中，每个数只使用一部分的样本进行训练，每个树也只是用一部分的特征进行学习，所以在本项目中，不妨以 Carol 为基础，将训练集打乱并拆分成四份，对每一份训练集应用 carol 模型参数进行训练，训练完成后分别使用这四个模型对测试集进行预测，并将最终的结果求平均值得到结果。具体实施代码在 Predict.ipynb 中的 3.Fusion 部分。

4. 结果

本项目的题目来源于 Kaggle 数据竞赛，因此模型的评价方法和优劣验证都很明确。在第三章中我总共得到了三种主要的模型，分别为基于随机森林算法的模型 danny，基于 xgboost 的模型 carol 以及融合模型 fusion 通过训练过程中的模型训练集、验证集评价。可以看出 xgboost 模型整体上相比随机森林模型具有较明显的优势。将三种模型的预测结果提交至 kaggle 验证得到以下结果：

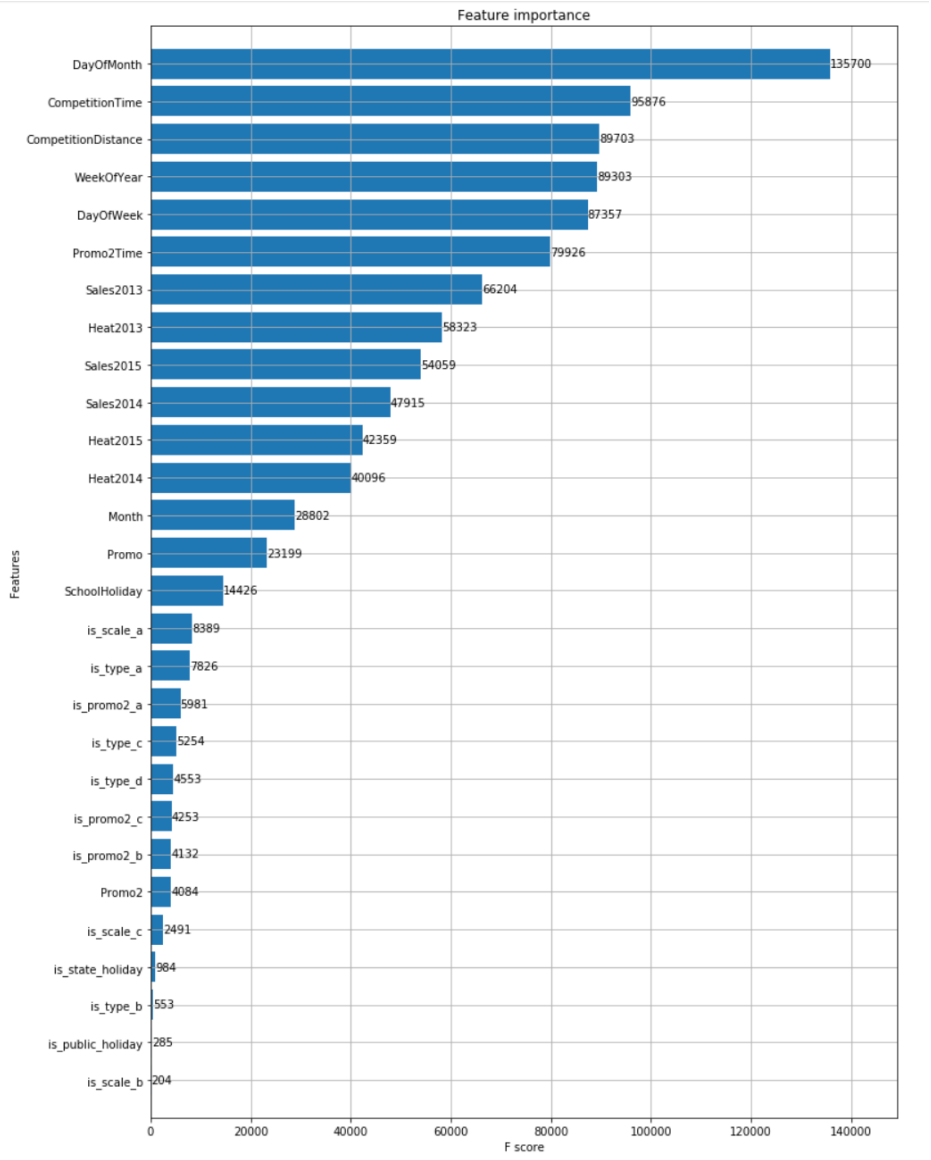
模型	文件名	Private score	Public score
Danny（随机森林）	RF_danny_submission.csv	0.12421	0.11103
Carol（xgboost）	xgb_carol.csv	0.11452	0.11258
Fusion	fusion.csv	0.11560	0.11007

可以看出，和研究过程中的模型表现基本一致，carol 和 fusion 模型表现出了更优异的预测能力。fusion 相对于 carol 模型的 private score 稍微降低，但是 public score 的得分更优，整体上要微微优于 carol 模型。对比项目原本计划的达到的 0.1173 的 private score 目标，基于 xgboost 的 carol 和 fusion 都很好的完成了任务，但是基于随机森林的 danny 在 private score 上距离目标分数还有一段距离，而其 public score 则较为优异，这说明模型还是存在一定的问题，或是过拟合或是泛化能力差，导致在 private 和 public 数据集上的表现差距明显。本项目最终以基于 xgboost 的 carol 模型作为最终结论。

5. 结论

5.1. 自由形态的可视化

对于本项目最终模型 carol，我绘制了 max_depth=7 时的特征重要性图，可以看出，时间特征、竞争商家特征周期促销特征以及销售热度（自建特征）对结果的影响都很大，这也是和预期保持一致的。但是有趣的时在时间特征中，DOM，WOY 以及 DOW 三个特征的位置很靠前，比是否节假日、学校放假等时间特征要更为重要，这说明销售数据受时间周期性影响非常大。



5.2. 思考

在本项目实施过程中，我再次巩固了 udacity 课程中学习到的技巧，也在过程中学习了很多课程中未能接触的知识，例如 xgboost 模型。通过这次完整的机器

学习任务，从特征工程到算法比选以及最终结论，对机器学习任务的常规流程和方式有了深入的体会。

在特征工程阶段，我其实也进行了多次计算，使用最初原始数据、增加了热度数据以及放缩后的数据得到的模型差距较为明显，报告中使用的最终 cesar 数据集的表现要明显较优，在随机森林和 xgboost 模型的默认参数下也可以取得能够接受的结果。通过这一过程的体验，我领会到了特征工程对机器学习的重要性，也；理解了“特征工程决定上限，模型优化逼近上限”这句话。

5.3. 改进

在本项目实施过程中，受制于时间和知识面，有很多地方都有巨大的进步空间，我计划在毕业之后继续完善的问题有以下几个：

1. 参数优化的优化步骤：在本项目的随机森林模型部分，调参数花费了巨大的时间，而结果也不甚理想。我在调参过程中只是使用了暴力穷举的办法，需要深入研究如何科学的调参。

2. Xgboost 参数的理解：在本项目中 xgboost 模型使用了很少的参数，调整过程也很简单，但是对于使用的参数理解程度还不够。

3. 特征工程：本项目中的特征工程全部来自于原始数据的组合和拆分，看到 kaggle 上有人使用外界的数据例如天气数据来帮助分析，在后续也可以进行类似的尝试。

4. 更多的模型：本项目采用的随机森林模型和 xgboost 模型都是通用性较好的模型，但是在 xgboost 模型的特征重要性图中可以看到，数据的周期性影响巨大，可以考虑寻找更适合时间周期性的机器学习模型。

参考文献：

[1] https://en.wikipedia.org/wiki/Random_forest

[2] http://www.cnblogs.com/pinard/p/6160412.html

[3] http://www.cnblogs.com/pinard/p/6160412.html

[4] http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[5] 小巧玲珑：机器学习届快刀 XGBoost 的介绍和使用 https://zhuanlan.zhihu.com/p/29432901

[6] http://xgboost.apachecn.org/cn/latest/parameter.html

[7] xgboost demo https://github.com/apachecn/xgboost-doc-zh/tree/v0.60/demo/en

[8] https://en.wikipedia.org/wiki/Tukey%27s_range_test

[9] https://zh.wikipedia.org/wiki/%E7%AE%B1%E5%BD%A2%E5%9C%96