# 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（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：

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**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Competition Distance** | **Competition OpenSinceMonth** | **Competition OpenSinceYear** | **Promo2** | **Promo2 SinceWeek** | **Promo2 SinceYear** |
| **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：

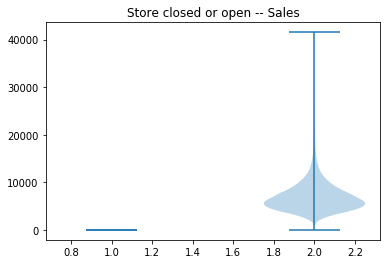
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Store** | **DayOf Week** | **Date** | **Sales** | **Customers** | **Open** | **Promo** | **State Holiday** | **School Holiday** |
| 101 | 5 | 2015-07-31 | 11075 | 915 | 1 | 1 | 0 | 1 |
| 101 | 5 | 2015-06-19 | 8779 | 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。

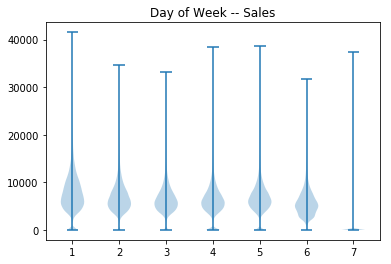
### 2.2. 探索性可视化

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。

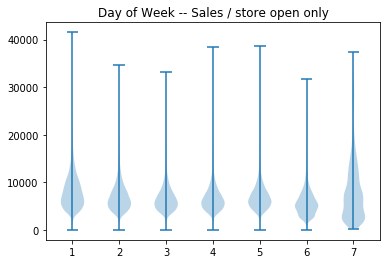
门店开门与否对销售额应当有很大的影响，下图是训练数据中样本关门（左侧）或开门（右侧）情况下销售额的提琴图，可以看出，门店关门时没有产生销售额。因此，在后续讨论中，可以考虑剔除店铺关门的样本。



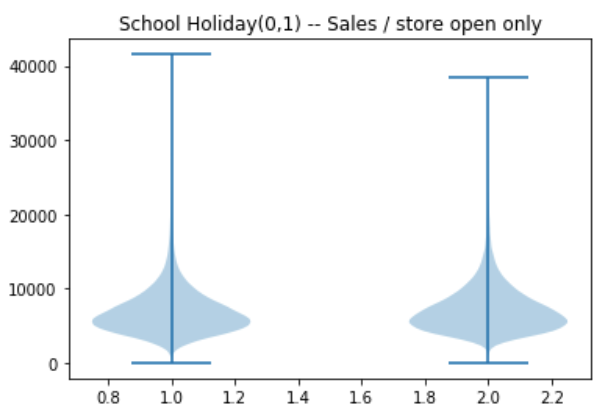
在数据集中，样本位于每周的第几天也被采集下来，下图是每周的七天对应的所有销售额的提琴图。



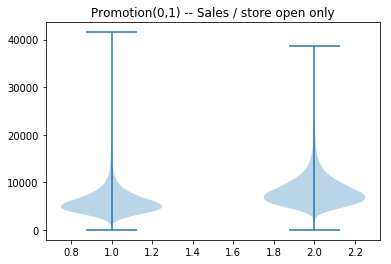
可以看出，在不考虑店铺开门与否的统计中，周一至周六的销售额分布规律相近，大量的数据集中在10000以下的部位，而周日的样本数据则大量聚集在0附近，但是最大值却处于一个较高的水平这是因为门店在周日会发生两个极端情况，普通的周日，大多数门店会休假，而在节假日前后，部分门店会开店。因此，我又绘制了只考虑门店开门状态下的七天销售额提琴图如下，可以看到，在只考虑开门样本时，周日的销售额分布要比其余6天更宽。从这里也可以看出，有必要在后续分析中剔除未开门的样本，否则容易形成误导。



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 reference wiki

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）。

随机森林算法在实施过程中可以调整的参数和其对应的意义如下<http://www.cnblogs.com/pinard/p/6160412.html，http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>。：

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。

#### 2.3.3. Turkey 异常截断法

#### 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。

**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：
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：
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：
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：

**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](https://www.kaggle.com/c/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

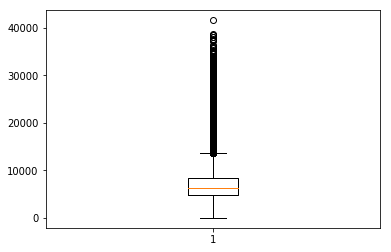
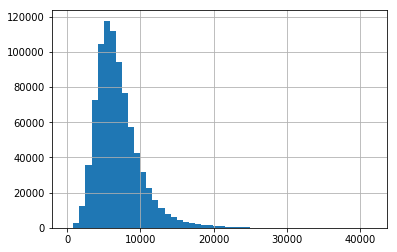
在机器学习任务中，数据的预处理非常重要，通过前面的工作，我制定了本项目数据预处理的计划流程如下：

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

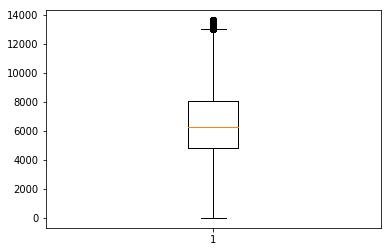
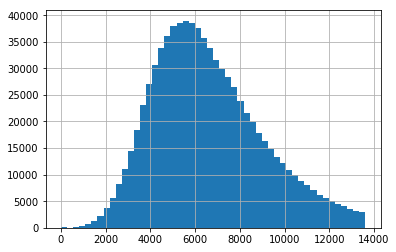
#### 3.1.1. 训练数据清洗

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取得是处。 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：

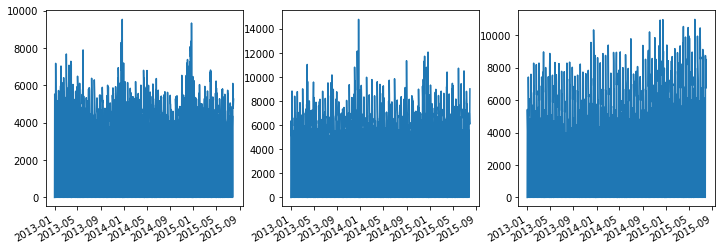
* + 样本日期在竞争商家开业之前，差值为负数
  + 样本日期在竞争商家开业之后，差值为正数
  + 该样本所对应的门店缺失竞争商家数据。

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)。

#### 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. 未优化的随机森林

#### 3.2.2. 未优化的 XGBoost

#### 3.2.3. 随机森林优化过程

#### 3.2.4. XGBoost 优化过程

#### 3.2.5. 算法整合尝试

### 3.3. 改进

## 4. 结果

### 4.1. 模型评估与验证

### 4.2. 理由

## 5. 结论

### 5.1. 自由形态的可视化

### 5.2. 思考

### 5.3. 改进