机器学习纳米学位毕业项目

项目背景：

Rossmann 在欧洲经营着 7 国 3000 个药店。目前，Rossmann 商店的经理被要求预测他们未来六周的日销售情况，商店销售受很多因素的影响，包括促销，竞争，学校和州政府的假期，气候和地区。由于数千经营者依据他们独特的情况预测销售情况，结果的准确性可能有很大的不同。在这个项目中，将挑战 6 周 1115 家德国境内的 Rossmann 商店的每日销售额。可靠的销售预报可以让商店经营者增加工作效率和积极性来创造更高效的工作人员安排。通过 Rossmann 创建一个强壮的预测模型，你将帮助经营者们关注对他们来说什么是最重要的：他们的客户和他们的团队。

选择该项目的原因是，我今后硕士的主要研究方向是对已有数据进行分析，其次是由于 GPU 算力有限，不想再做一个关于深度学习的项目了，最后，我想为以后在 kaggle 进行数据比赛作为一个试水项目。

问题描述：解决办法所针对的具体问题

通过对以前的销售记录进行学习建模，对六周后的日销售情况进行预测。具体步骤和可能遇到的问题如下：

对数据进行基本的统计，探索，分析销售数据的整体情况

对数据进行清洗，处理异常值，缺失值，并且能解释数据对应的场景意义。

对数据进行相关性分析，提取出有价值的特征值。

针对时间序列对数据进行提取分析，切分不同的训练集，验证集，测试集。

使用机器学习方法进行建模，例如 XGBoost，lightGBM 等，调整参数。

采取一定评价指标作为评价标准，并对结果进行可视化。

评价指标：

运用 RMSPE 来作为验证函数，该值越低代表差异性越小。

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

y_i是真实的销售数据，$\hat{y}_i$是预测数据，这个问题是回归问题。

输入数据：

输入数据集：
train.csv
"Store","DayOfWeek","Date","Sales","Customers","Open","Promo","StateHoliday","School","Holiday" 等字段
test.csv
"Id","Store","DayOfWeek","Date","Open","Promo","StateHoliday","SchoolHoliday" 等字段
sample_submission.csv –
"Id","Sales"等字段。
store.csv - 关于商店的附加信息 包含有
"Store","StoreType","Assortment","CompetitionDistance","CompetitionOpenSinceMonth","CompetitionOpenSinceYear","Promo2","Promo2SinceWeek","Promo2SinceYear","PromoInterval"字段。

数据集特征如下:
Id - 测试集中表示一条记录的编号。
Store - 每个商店的唯一编号。
Sales - 任意一个给定日期的销售营业额。
Customers - 给定那一天的消费者数。
Open - 商店是否开门标志，0 为关，1 为开。
StateHoliday - 表明影响商店关门的节假日，正常来说所有商店，除了极少数，都会在 节假日关门，a=所有的节假日，b=复活节，c=圣诞节，所有学校都会在公共假日和周 末关门。
SchoolHoliday - 表明商店的时间是否受到公共学校放假影响。
StoreType - 四种不同的商店类型 a，b，c 和 d。
Assortment - 描述种类的程度，a = basic, b = extra, c = extended。
CompetitionDistance - 最近的竞争对手的商店的距离。
CompetitionOpenSince[Month/Year] - 最近的竞争者商店大概开业的年和月时间。
Promo - 表明商店该天是否在进行促销。
Promo2 - 指的是持续和连续的促销活动。: 0 = 商店没有参加, 1 = 商店正在参加。
Promo2Since[Year/Week] - 表示参加连续促销开始的年份和周。
PromoInterval - 描述持续促销间隔开始，促销的月份代表新的一轮，月份意味着每一轮的开始在哪几个月。

对数据的初步统计分析如下：
train.csv 一共有 1017209 行 8 列数据，其中前 5 行数据如下图

Date	Store	DayOfWeek	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
2015-07-31	1	5	5263	555	1	1	0	1
2015-07-31	2	5	6064	625	1	1	0	1
2015-07-31	3	5	8314	821	1	1	0	1
2015-07-31	4	5	13995	1498	1	1	0	1
2015-07-31	5	5	4822	559	1	1	0	1

test.csv 一共有 41088 行 7 列的数据，其中前 5 行数据如下图：

Date	Id	Store	DayOfWeek	Open	Promo	StateHoliday	SchoolHoliday
2015-09-17	1	1	4	1.0	1	0	0
2015-09-17	2	3	4	1.0	1	0	0
2015-09-17	3	7	4	1.0	1	0	0
2015-09-17	4	8	4	1.0	1	0	0
2015-09-17	5	9	4	1.0	1	0	0

Store.csv 一共有 1115 行 10 列数据，其中前 5 行数据如下图：

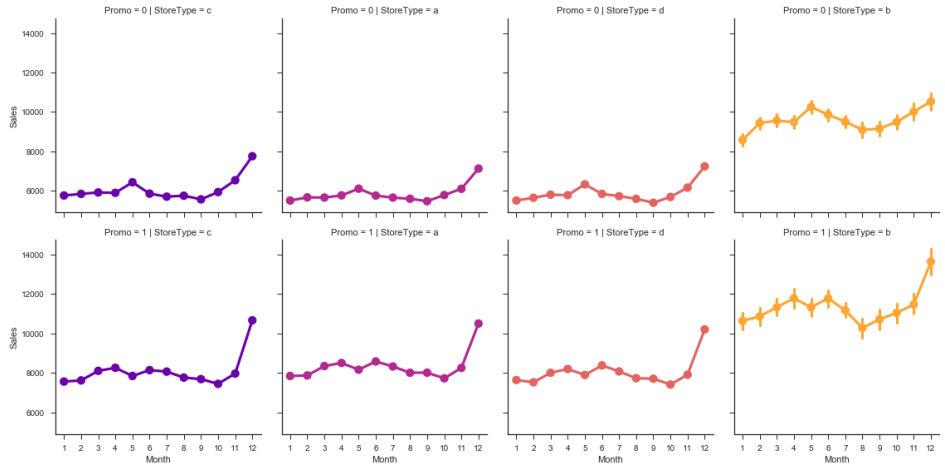
	Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear	F
0	1	c	a	1270.0	9.0	2008.0	0	NaN	NaN	
1	2	a	a	570.0	11.0	2007.0	1	13.0	2010.0	J
2	3	a	a	14130.0	12.0	2006.0	1	14.0	2011.0	J
3	4	c	c	620.0	9.0	2009.0	0	NaN	NaN	
4	5	a	a	29910.0	4.0	2015.0	0	NaN	NaN	

针对 test.csv 里的 11 条缺失数据，根据缺失的 state,经过判断，应该补为 1。
针对 store.csv 里 3 条缺失数据，其缺失 competition 相关的数据，在这里判断为没有竞争对手，则将其距离设置为一个较大值，且竞争时间设置远超其他时间之后，其余缺失数据没有竞争时间的，随机生成一个之前的竞争时间。对于 promo2 相关的缺失数据采取补 0。
针对 train 里的数据，无确实情况，则不作处理。

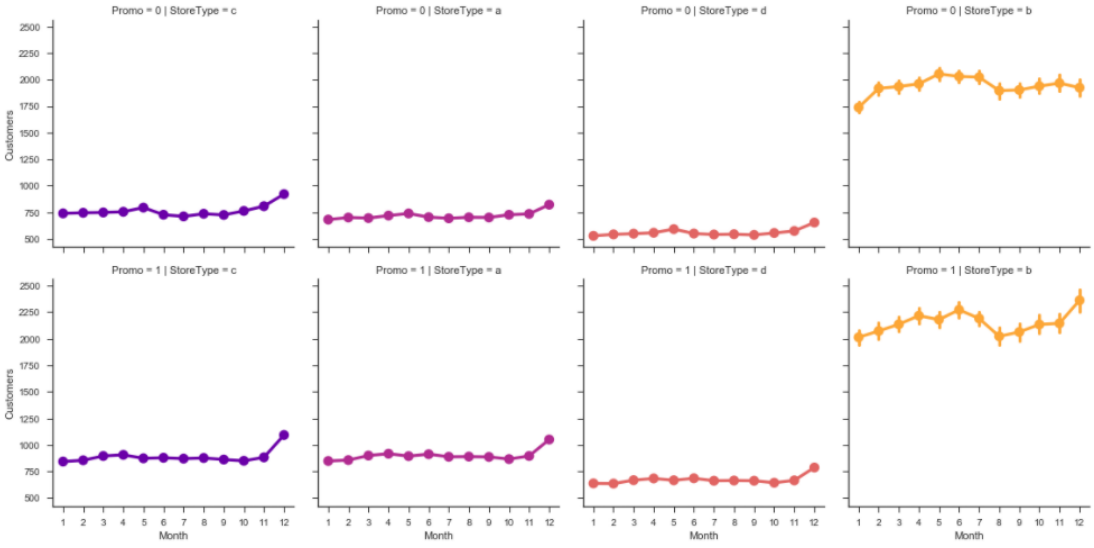
数据分析

在经过数据修补处理以后，对不同月份在不同 StoreType 和 Promo 下进行统计，有：

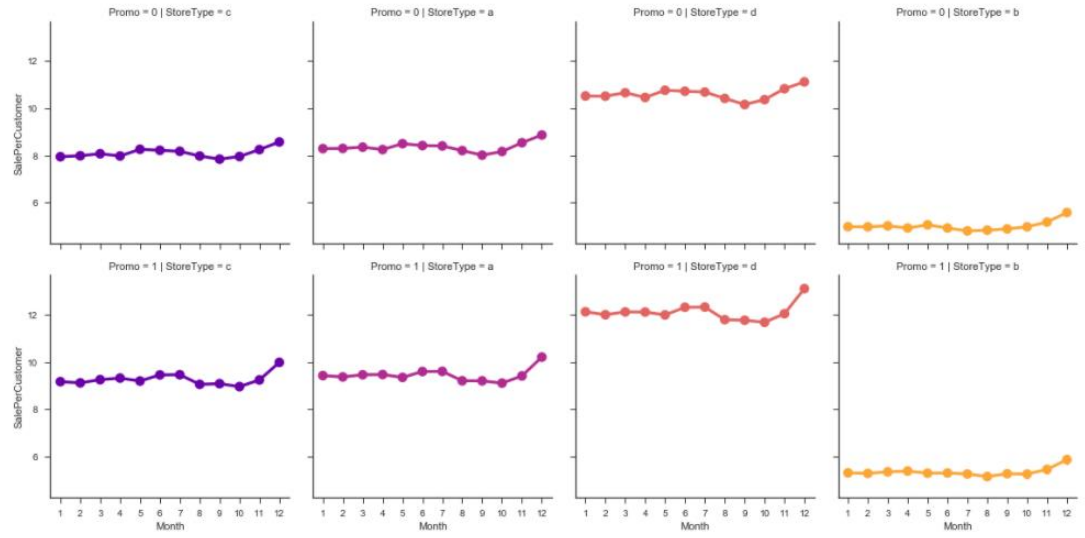
1.在不同月份下，sales 的走势：



2.在不同月份下，customers 的走势：



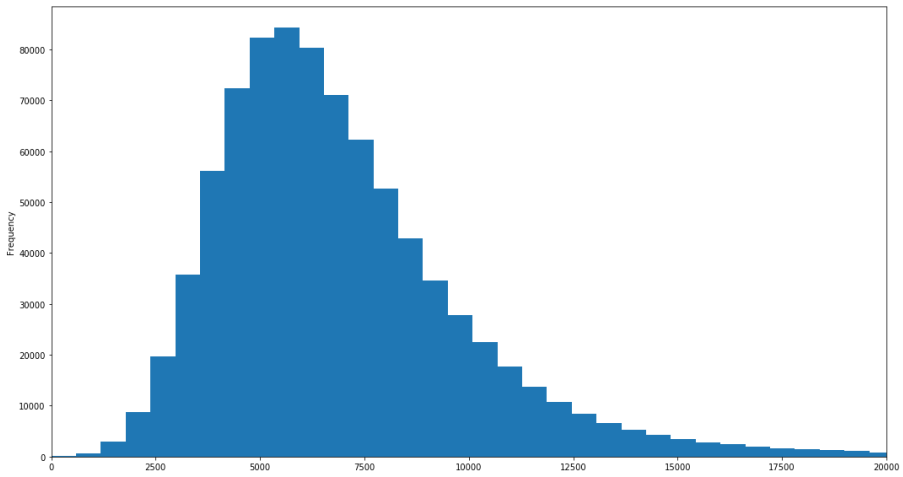
3.在不同月份下，SalePerCostomer 的走势



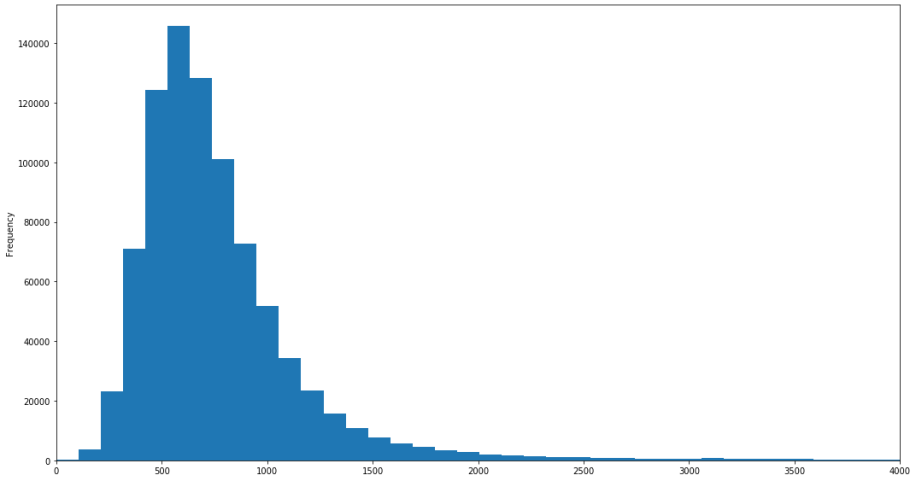
可以发现不同商店类型，以及不同时间，日期是预测 Sales 的关键，同时

Promo 也体现出了分析的重要性.同时分别对 Sales,Customers,Competition Distance 作直方图统计分布情况。

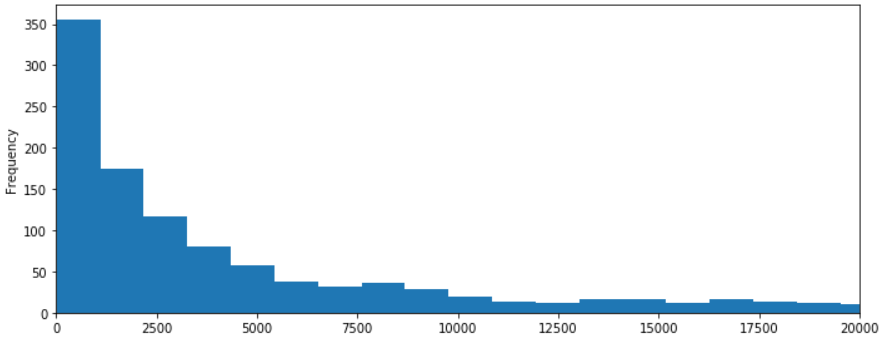
Sales:



Customers:



Competiton Distance:



由此可见 Customer 和 Sales 是基本满足偏左的正态分布的,Competition Distance 距离越长，对应的分布也越少

算法和技术

该问题属于回归问题，在这里出于学习的目的，对异常值处理后，采取xgboost 模型。主要是考虑到数据集并不大，深度学习可能引起过拟合，泛化性能不够好，且深度学习主要针对图像，音频，传统统计学的方法可能更适合，xgboost 在 kaggle 里多次赢得 kaggle 比赛表现稳定良好。[3]

Xgboost 是 GB 算法的高效实现，主要特点在于：

（1） xgboost 在目标函数中显示的加上了正则化项，基学习为 CART 时，正则化项与树的叶子结点的数量 T 和叶子节点的值有关

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

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

（2） GB 中使用 Loss Function 对 f(x)的一阶导数计算出伪残差用于学习生成 fm(x)，xgboost 不仅使用到了一阶导数，还使用二阶导数。

第 t 次的 loss：

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{(t-1)} + f_t(\mathbf{x}_i)) + \Omega(f_t)$$

对上式做二阶泰勒展开：g 为一阶导数，h 为二阶导数：

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n} [l(y_i, \hat{y}^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t)$$

$$\text{where } g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}) \text{ and } h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$$

（3） 上面 Cart 回归树中寻找最佳分割点的衡量标准是最小化均方差，xgboost 寻找分割点的标准是最大化，lamda,gama 与正则化项相关。

$$\mathcal{L}_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma$$

Python API 的核心数据结构是 DMatirx,设置格式为：

class xgboost.DMatrix(data, label=None, missing=None, weight=None, silent=False, feature_names=None, feature_types=None, nthread=None)

其中：data 可以为 string,numpy,array,scicpy.sparse,pd.dataframe ，label 和 feature 需要提前设置好存入。

核心训练函数为：xgboost.train().params 用于设置 booster 的参数，evals 用于展示当前训练成果，feval 代表评估训练结果的函数，early_stopping_rounds 代表只有当误差在每个 stopping_rounds 减小时才停止训练，learning_rate 代表学习率。

Scikit-Learn API 里包括：max_depth 每一个学习器的深度，learning_rate 学习率，subsample 代表下采样的比例，colsample_bytree 当构建每颗树时下采样列的比例

解决办法：

1. 统计各个数据值，得到每个特征值对应的统计特征，了解数据的分布情况，补充缺失值，找到异常值。
2. 找到关联性比较强的特征值，筛选出算法所需要的特征变量。
3. 将数据按照比例随机切割成 train 和 test，test 占整个数据集的 0.01，并整理出对应的训练集，测试集，验证集，从而达到符合算法模型的输入要求。
4. 调整模型参数，使模型预测结果达到最佳。
5. 学习相关的可视化结果的知识来完成最后结果的提交。

基准模型

可以尝试用机器学习一些比较传统的算法与 Xgboost 等集成算法进行比较模型的表现。在这里我指定 XGBOOST 为一个基准模型，在不对数据集做时间序列划分下，划分 train.data 为训练集和测试集，test.data 为测试集，初步训练下，选用[1][2]：

“Store”,”CompetitionDistance”,“CompetitionOpenSinceMonth”,”CompetitionOpenSinceYear”,”Promo”,”Promo2”,”Promo2SinceWeek”,”Promo2SinceYear”,”SchoolHoliday”,”DayOfWeek”,”month”,”day”,”year”,”StoreType”,”Assortment”作为特征。

参数选择：params = {"objective": "reg:linear",
"eta": 0.3,
"max_depth": 15,
"subsample": 0.7,
"colsample_bytree": 0.7,
"silent": 1
}

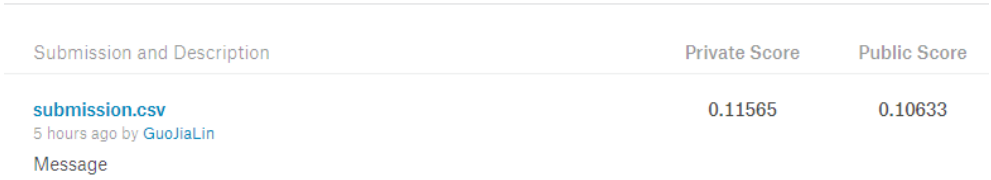
num_trees = 350

训练后提交结果在 private score 上得分 0.15878，public score 得分为 0.1393

在后续实验中，会调整 xgboost 训练的参数，添加有用的特征，删除无用的特征和清洗数据从而提高正确率。

经过试验调整，最终选择加入了 Stateholiday 这一特征，增加了 Promo-Interval，同时参考了 disscussion 上清除异常值的办法，最后成绩达到了：

Private:0.11565 public:0.10633



在调整参数的过程中，发现降低学习率从默认值到 0.04 的过程中明显提升了学习成绩，调整参数的过程中陷入了瓶颈：private score 成绩一直在 0.12 以上，

Max_depth	Learning_rate	private score	Train_loss	Val_loss
8	0.02	0.12312	0.101	0.109
8	0.03	0.12333	0.102	0.106
8	0.04	0.12262	0.103	0.104
9	0.02	0.12301	0.101	0.105
……	……	……	……	……

然后参考了 discussion[5]里方法，清洗超过中值三倍标准差的异常值，达到了最佳效果，max_depth 设置为 10 时达到了毕业要求，在没有引入任何外部数据和单模型的情况下，这个结果比较让人满意。经过去除异常值的数据集进行训练，eval-rmspe 从 0.105 左右可以降到 0.085,达到了更好的预测效果。

评估指标

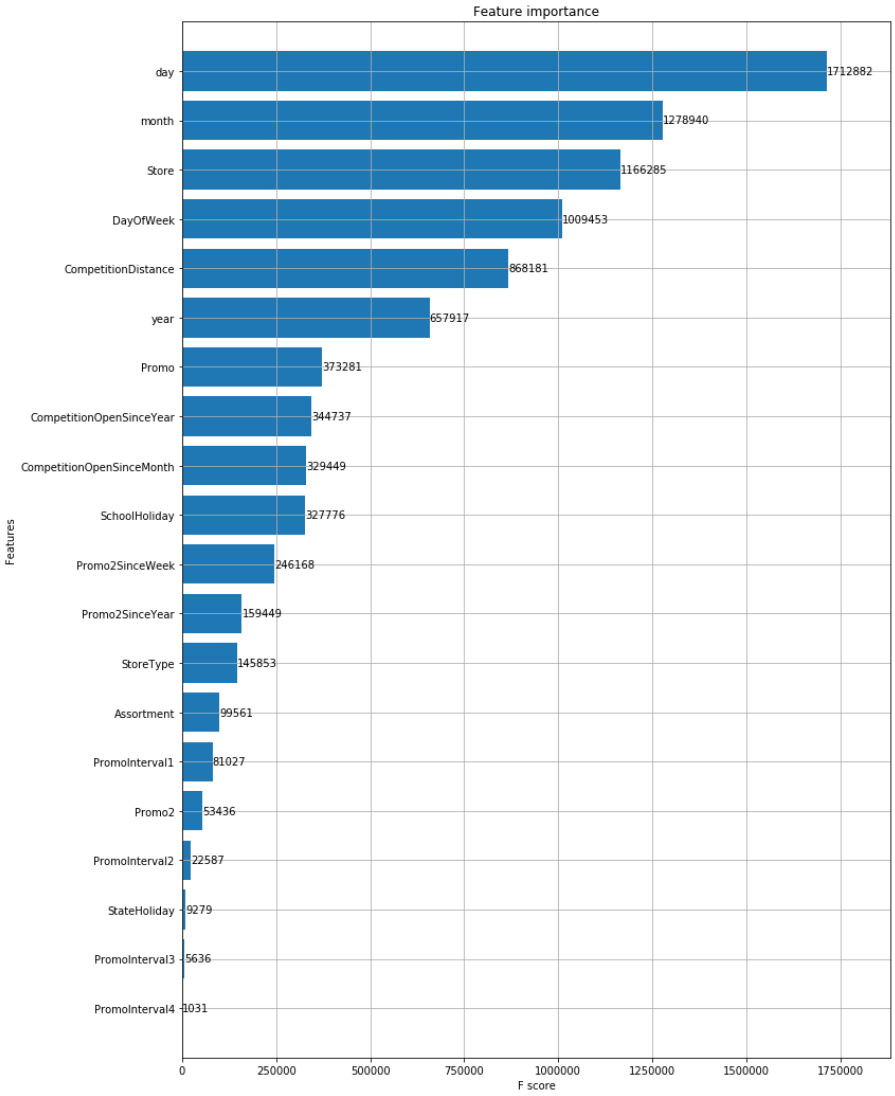
采取 kaggle 提供建议的 RMSPE 来做为验证函数，该值越低代表差异性越小，他是指模型的预测值和实际观察值之间的差异一种衡量方式。

设计大纲

先对数据做统计，清洗，整理，处理缺失值，统一数据类型。
做数据相关性分析，找出关联度高的特征，剔除部分特征。
融合数据，剔除和拆分部分特征，去除部分异常值。
划分训练集，验证集，训练集，为训练做准备。
配置模型参数进行训练，得到初步训练的结果。
对初步训练作总结，对特征进一步优化，对参数进一步优化，在进行训练，寻找出最佳超参，保存模型。
对结果进行 predict,并保存提交。

训练结果与分析

private score 最终到了 0.11565,相比于 github 以及 kaggle 上使用 xgboost 的分享，该模型需要的生成树小，训练时间短，一个训练周期 3 分钟内就可以完成（cpu:i7-8750H,16GB）,其他 xgboost 模型需要大量的生成树，特征数量多，且容易过拟合，达不到最好效果，下图对特征重要性可视化：[4]



由此分析可以得，之前的数据分析与训练结果发现：日期，Competion-Distance，节假日，促销情况起到了大部分决定销售结果的作用，这与人的常识相同。

对项目的思考

项目应该分成数据分析，模型选择，评估和优化，通过对数据的分析，我们可以大概了解到数据的分布特征，并且根据实际情况补充缺失值，明确问题为回归问题。模型选择 xgboost,在网上有充足的资源，充分参考他们对特征模型的选择以后，选择特征和训练参数，反复实验。第一名的分享中，他对特征做了很深度的加工处理，并且结合外部数据，取得了 0.10 左右的成绩，令人叹服。如果要改进，我觉得可以继续参考 kaggle 上对特征的选择，重新选优秀的特征进行训练，并且尝试引入外部数据，提高最终正确率。

采取了 early-stopping，这样可以比较好的控制过拟合，随机划分验证集可

能会出现训练集包含了测试集以后的情况，而我们的任务是预测最后六周的销量，所以这不太适合，但是按照日期划分测试验证集，模型又无法学习到离预测六周销量最近一段时间的销量特征，所以各有利弊，在这里，我采取的还是随机划分训练和验证集的方式。在这里，我对部分类别变量进行了哑变量的处理，但也可以转化为数字，进行区分。

参考文献

[1]Elena Petrova. Time Series Analysis and Forecasts with Prophet
https://www.kaggle.com/elenapetrova/time-series-analysis-and-forecasts-with-prophet
Published:2015.9
[2]Paso. XGBoost in python with RMSPE
https://www.kaggle.com/paso84/xgboost-in-python-with-rmspe
[3]wxquare wxquare 的学习笔记
https://www.cnblogs.com/wxquare/p/5541414.html
[4]xueweiyema xgboost predicting
https://github.com/xueweiyema
[5] https://www.kaggle.com/c/rossmann-store-sales/leaderboard