课　　程：2017 软件工程综合实训

项　　目：　　数据挖掘比赛

院　　系：数据科学与计算机学院

专　　业：　　软件工程

学生姓名：　　蔡岳

学　　号：　　14331012

授课教师：　　郑子彬，曾海标

2017 年 07 月 01 日

目录

一、比赛简介

Housing costs demand a significant investment from both consumers and developers. And when it comes to planning a budget—whether personal or corporate—the last thing anyone needs is uncertainty about one of their biggets expenses. Sberbank, Russia's oldest and largest bank, helps their customers by making predictions about realty prices so renters, developers, and lenders are more confident when they sign a lease or purchase a building.

Although the housing market is relatively stable in Russia, the country's volatile economy makes forecasting prices as a function of apartment characteristics a unique challenge. Complex interactions between housing features such as number of bedrooms and location are enough to make pricing predictions complicated. Adding an unstable economy to the mix means Sberbank and their customers need more than simple regression models in their arsenal.

In this competition, Sberbank is challenging Kagglers to develop algorithms which use a broad spectrum of features to predict realty prices. Competitors will rely on a rich dataset that includes housing data and macroeconomic patterns. An accurate forecasting model will allow Sberbank to provide more certainty to their customers in an uncertain economy.

二、实验环境

操作系统：Ubuntu16.04
编程工具：Jupyter Notebook
Python 版本：Python3

三、数据观察

在进行数据处理和训练之前，我认为第一步比较关键的操作是对数据的观察，先要了解数据的分布特征以及重要程度等特点才能进行之后的操作。在本次实验中，我觉得对数据的观察有以下几个作用：

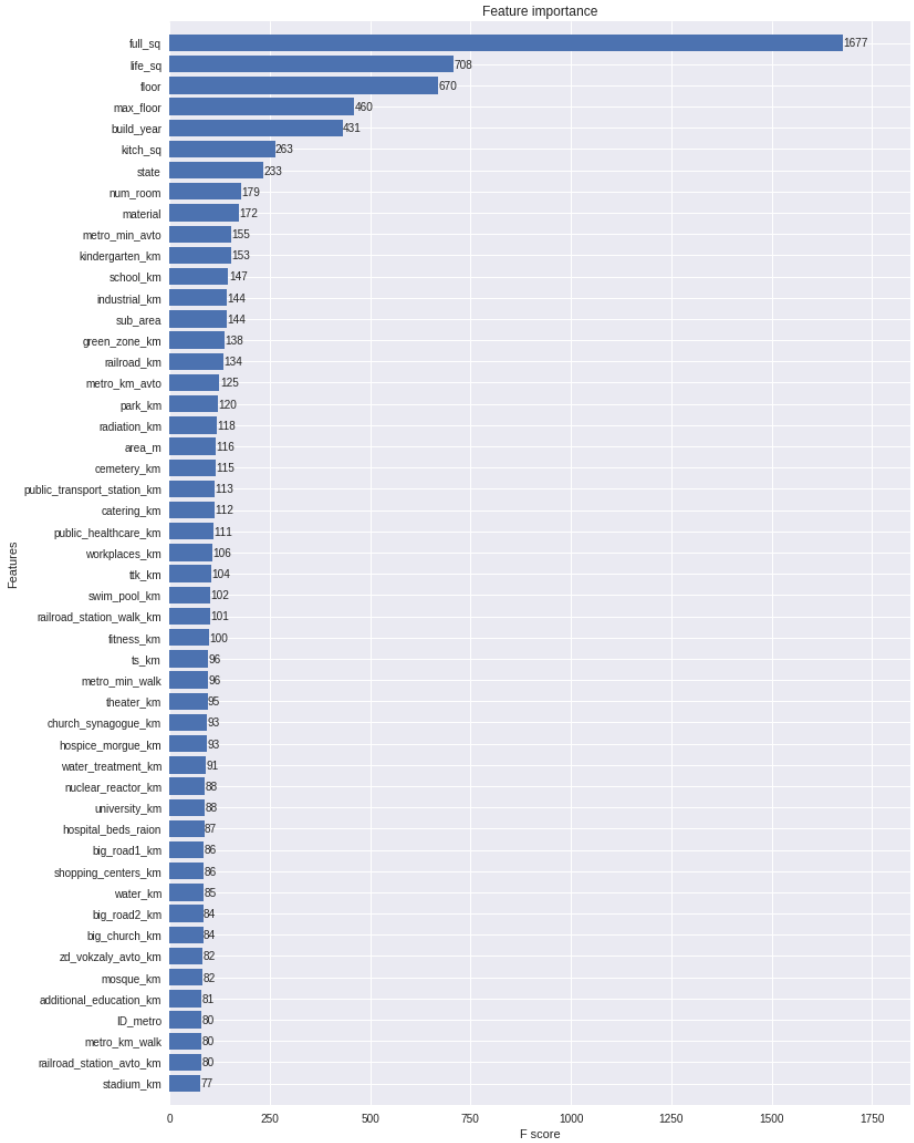
1.对特征量的重要程度进行排序，可以对更重要的特征进行操作，稍微忽略一些与房间关联程度很小的特征量的处理；

2.可视化比较重要的特征量，可以观察到这些特征量的异常数据，对接下来的数据清洗更方便；

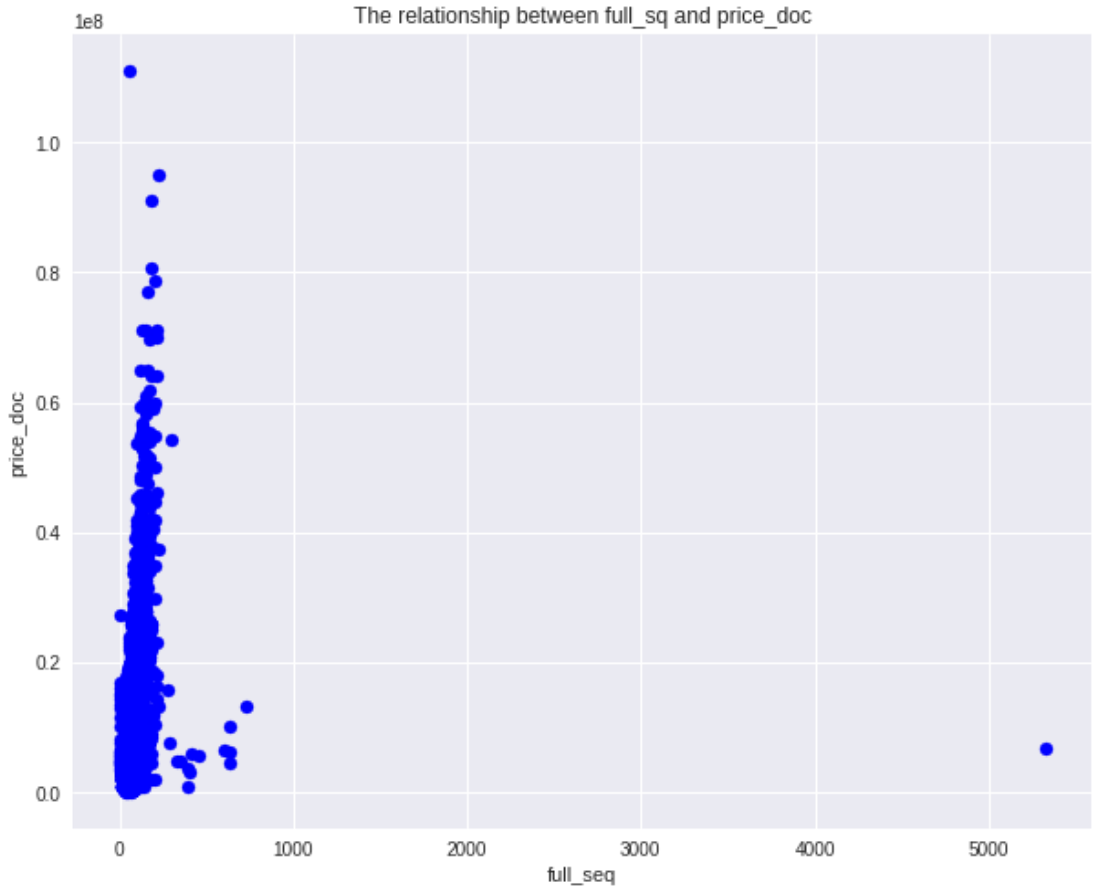
3.通过尝试可视化某些由旧特征组合出的新特征，可以确定该新特征是否与房价有较大的关联，有助于新特征对模型的训练。

以下是我对 train 数据集进行可视化的情况：

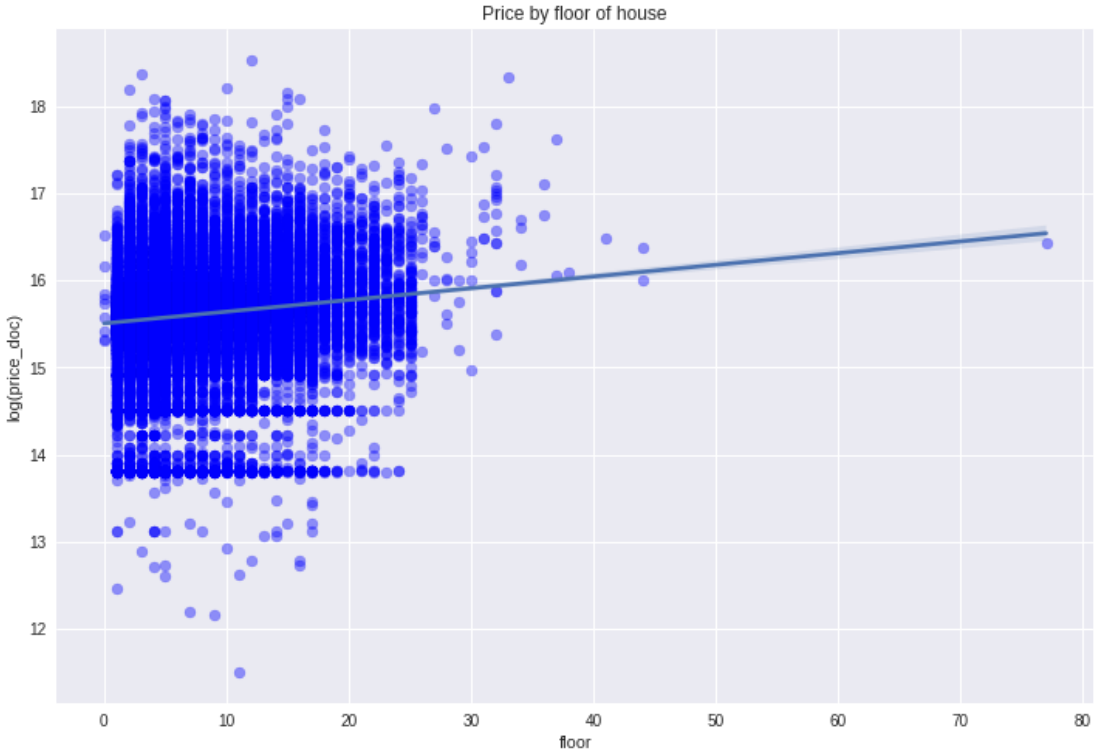
1.列举出对房价影响程度从大到校的部分特征量，这一步可以引导我着重关注影响程度大的特征量，且对这些特征量进行处理：



2.可视化比较重要的特征与房价之间的关系，观察其分布有利于下一步对数据的清洗：



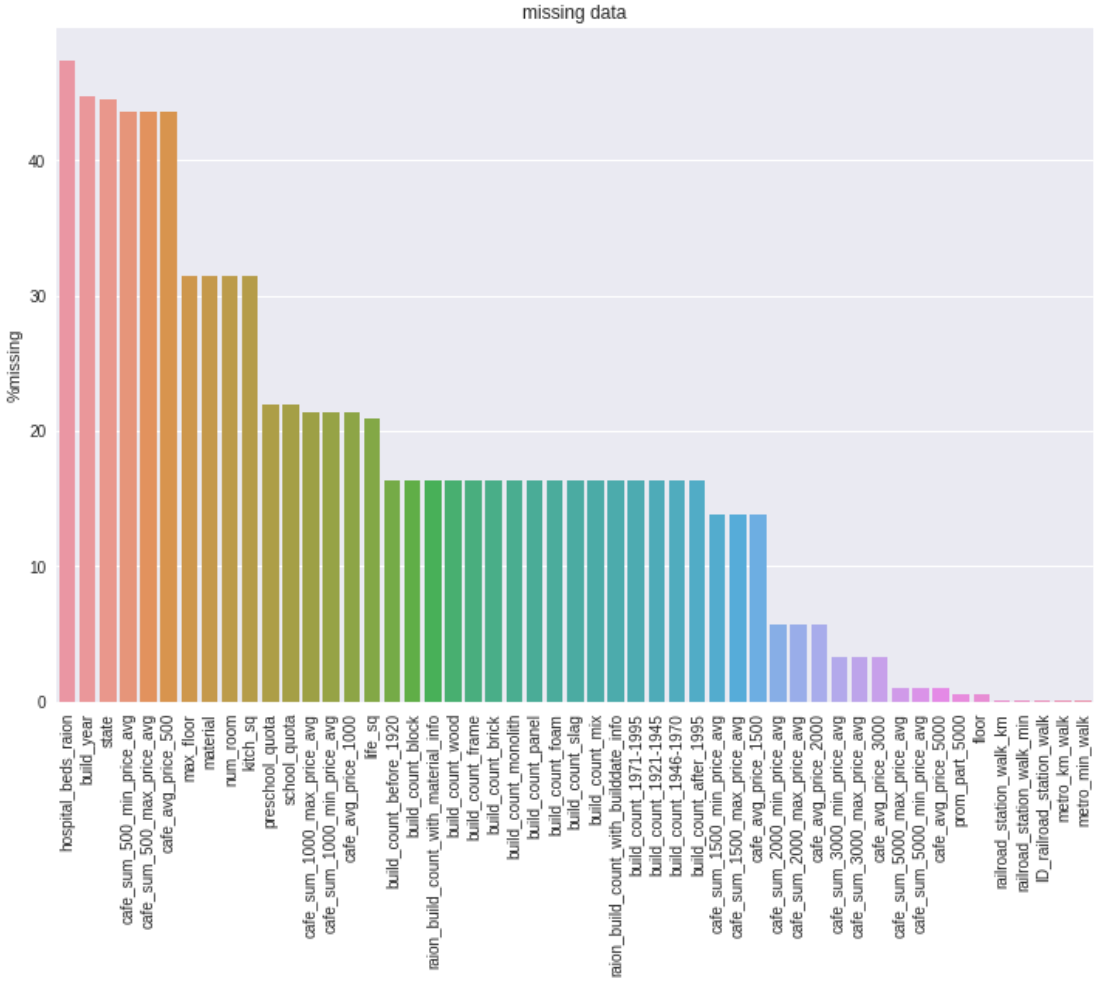
这是对特征 **full_sq** 进行的可视化处理，显然可以看出在左上角和右下角都有过分异常的数据，违背了生活常识（一般情况下住宅的总面积不可能大到超过 4000m² ），有可能是人工输入导致的错误。



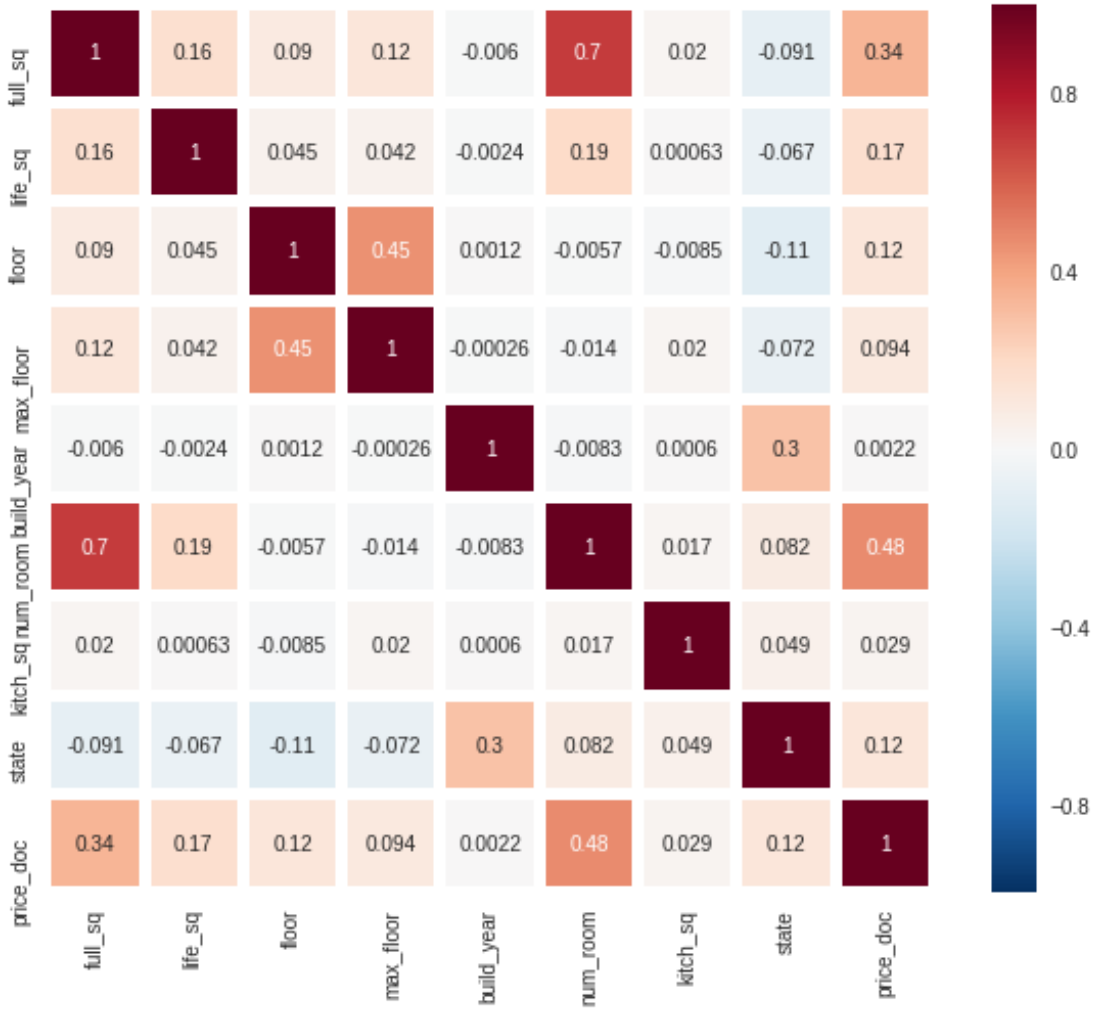
这是对重要程度排在第三位的特征 **floor** 进行的可视化处理，其中纵轴修改为 price_doc 的对数，因为 price_doc 的范围很大，在图中可能不易观察到规律，但是转换为对数形式之后，更容易发现 floor 与房价的关系，可以看出还是有一定的线性关系，同时也能够观察到一些脏数据。

实验过程中，我对多个特征进行可视化处理，在报告中仅列举两个比较关键的特征进行展示。

3.可视化数据的缺失率，可以对其中的一些异常数据在下一步进行处理：



4.对一些比较重要的特征进行相关性的分析，很明显地帮助了我在下一阶段特征工程中对于新特征的构造：



四、特征工程

本次实验中，我对数据的处理主要分为**数据清洗**和**新增特征**两个方向。

需要进行清洗的数据主要是在第一步可视化数据中观察到的一些异常数据，大致有以下几种情况：

1.面积问题：full_sq, life_sq 过小或者过大；life_sq 大于 full_sq（不符合基本常识，总面积一定大于等于生活面积）；

2.时间问题：建筑的年限过于久远，或者由于人工输入等错误导致建筑的年限甚至超过现在的时间；

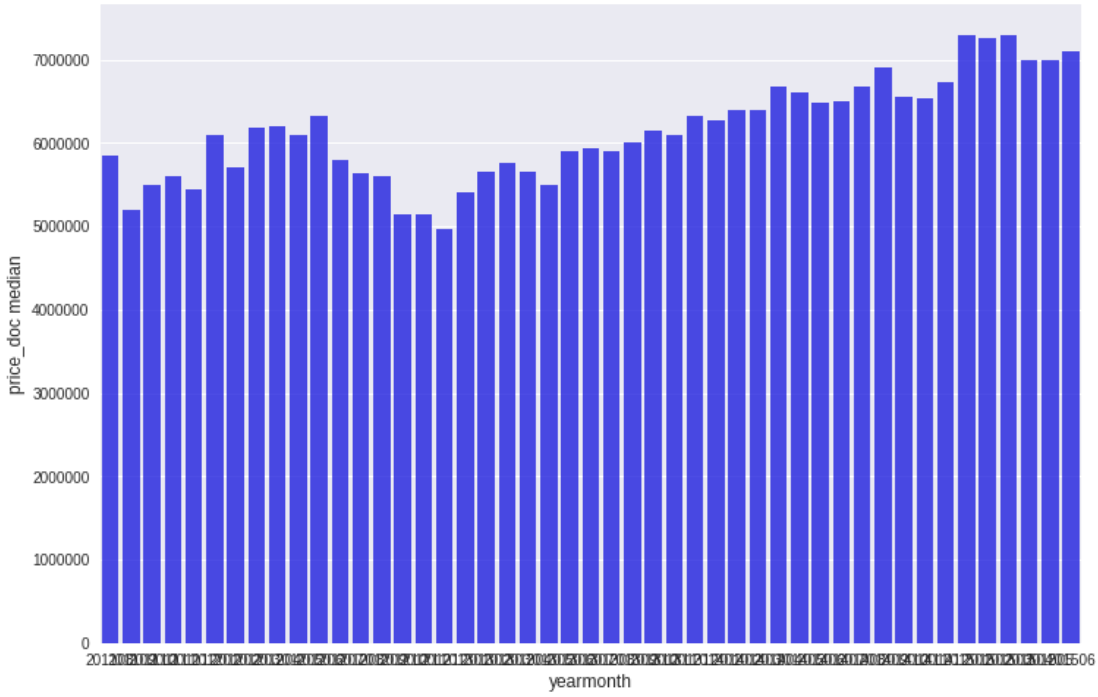
3.楼层问题：max_floor，floor 等于 0；floor>max_floor（不符合现实，住宅的楼层必定小于等于这栋楼的最大楼层 max_floor）；

4.其他比较琐碎的异常数据，如 state 等。

对于这些异常数据，我了解到一般情况下有两种处理方式，一种是将该异常数据赋值为该列其他所有数据的平均和等有效值的方式；另一种是只赋值为 NaN，由于之后训练选用的 xgboost 可以很好的容纳 NaN，所以本次实验中，我将所有的异常数据都赋值为 NaN。

另一方面，由于第一步尝试性地测试了一些由旧特征组合成的新特征对方昂加的影响，可以挖掘几个有效提升模型质量的特征：

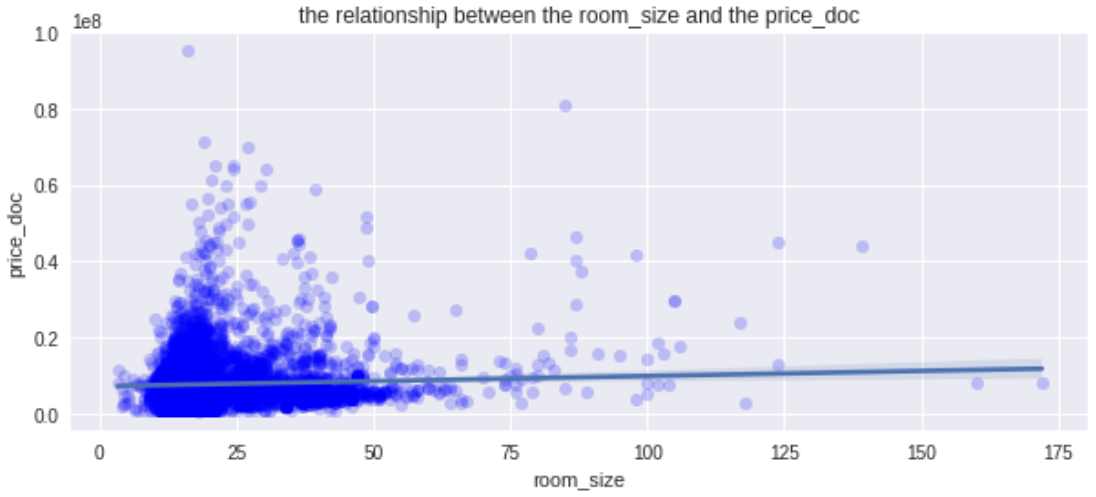
1.年月的序列（如 201101）的新特征：由下图可以看出年月的序列与房价有一个较为可观的线性关系，应该可以提升对 train 的训练效果，所以我将这个特征加入到训练和预测中，实际上按照提交的结果也可以看出十分有效：




```
month_year = (train.timestamp.dt.month*30 + train.timestamp.dt.year * 365)
month_year_cnt_map = month_year.value_counts().to_dict()
train['month_year_cnt'] = month_year.map(month_year_cnt_map)

month_year = (test.timestamp.dt.month*30 + test.timestamp.dt.year * 365)
month_year_cnt_map = month_year.value_counts().to_dict()
test['month_year_cnt'] = month_year.map(month_year_cnt_map)
```

2.房间大小：利用生活面积 life_sq 除以每个住宅拥有的房间数量 num_room 可以得到每个房间的平均大小，这个新恶整在训练模型中也起到了很好的作用：



```
train['room_size'] = train['life_sq'] / train['num_room'].astype(float)
test['room_size'] = test['life_sq'] / test['num_room'].astype(float)
f, ax = plt.subplots(figsize=(10, 4))
sns.regplot(x="room_size", y="price_doc", data=train, scatter=True,
            truncate=True, scatter_kws={"color": "b", "alpha": .2})
ax.set(title="the relationship between the room_size and the price_doc")
plt.show()
```

3.我还添加了许多其他新的特征，包括同学上课展示给出的提示以及 Kernels 给出的提示：

```
train['gender_ratio'] = train['male_f']/train['female_f'].astype(float)
train['kg_park_ratio'] = train['kindergarten_km']/train['park_km'].astype(float) #significance of children?
train['high_ed_extent'] = train['school_km'] / train['kindergarten_km'] #schooling
train['pts_x_state'] = train['public_transport_station_km'] * train['state'].astype(float) #public trans * state of listin
train['lifesq_x_state'] = train['life_sq'] * train['state'].astype(float) #life_sq times the state of the place
train['floor_x_state'] = train['floor'] * train['state'].astype(float) #relative floor * the state of the place
```

但是，与此同时，我也做出了一些失败的尝试，找到了一些没有对预测结果起到提升作用的特征：

1.比如一开始同学在展示的时候提到了一个利用 max_floor 减去 floor 之后得到的相对楼层的特征，将这个特征加入训练集之后，我的结果有了提升，但之后再加入其他特征以及利用 magic_number 进行调整之后，一直会有一个瓶颈，尝试着删除了相对楼层这个特征之后，得到的结果反而有了提升，所以最后我把这个特征去掉了：

```
#failure test
#train['floor_sub'] = train['max_floor'] - train['floor']
#test['floor_sub'] = test['max_floor'] - test['floor']
#bad_index = train[train.floor_sub < 0].index
#train.loc[bad.index, 'floor_sub'] = np.NaN
#bad_index = test[test.floor_sub < 0].index
#test.loc[bad.index, 'floor_sub'] = np.NaN
```

2.还有在 Kernels 中有人提到了利用住宅的经纬度两个特征来训练模型，我也尝试了加入经度和维度的两列数据，但是得到的结果特别差，提交到 Kaggle 上的预测结果比原来没加经纬度的结果的效果差了接近 0.01。

五、模型调参

本次实验一开始，我自己利用了一次 xgboost 来训练模型，经过了数据清洗以及新增特征两个步骤之后进行训练，但是得到的结果在 leaderboard 上会有一个极限，大概在 0.31xxxx，一直无法有突破。

后来我在 Kernels 上参考了另外一种模型，利用了三次 xgboost 进行训练，而且每次都有不同的处理方式以及参数调整。最后得到的三个预测结果再以不同的权重叠加得到最终的结果。

但利用三个模型之后，不管我新增什么特征或者删去什么特征，最终也会有一个瓶颈，后来我尝试调整三个预测结果的权重，得到了第二次的大幅提升，对于 35%的预测数据最终能够得到一个 0.300000 的成绩，下图是最终调整得到的相对较好的权重比：

```
first_result = output.merge(df_sub, on="id", suffixes=['_louis','_bruno'])
first_result["price_doc"] = np.exp( .78*np.log(first_result.price_doc_louis) +
                                    .22*np.log(first_result.price_doc_bruno) )
result = first_result.merge(gunja_output, on="id", suffixes=['_follow','_gunja'])

result["price_doc"] = np.exp( .78*np.log(result.price_doc_follow) +
                              .22*np.log(result.price_doc_gunja) )
```

六、Magic Number

此外，本次实验有一个很特殊而且在 Kernels 上也引发大家热烈讨论以及尝试的一个话题：Magic Number。在一开始没有加入 Magic Number 的时候，**新增特征**和**模型调参**是两个重要的能够优化预测结果的手段，但是在 Magic Numbe 的帮助下，对于 35%的预测数据有了更好的拟合效果。

我个人对 Magic Number 的看法有好有坏。

在实验过程中，有一些 Magic Number 其实是有一定的逻辑和现实依据支撑。比如对于下图所显示的 Magic Number，其实质是对过往的每年的每个季度的房价进行一定的调整，这其实是对比赛给出的 macro.csv 的一种体现，以每个季度作为单位，Magic Number 的作用可能在于弱化宏观经济的影响，比如物价可能带来的对房价的影响，所以我个人认为在此处利用 Magic Number，有一定的现实依据支撑：

```
rate_2015_q2 = 1
rate_2015_q1 = rate_2015_q2 / 0.9932
rate_2014_q4 = rate_2015_q1 / 1.0112
rate_2014_q3 = rate_2014_q4 / 1.0169
rate_2014_q2 = rate_2014_q3 / 1.0086
rate_2014_q1 = rate_2014_q2 / 1.0126
rate_2013_q4 = rate_2014_q1 / 0.9902
rate_2013_q3 = rate_2013_q4 / 1.0041
rate_2013_q2 = rate_2013_q3 / 1.0044
rate_2013_q1 = rate_2013_q2 / 1.0104  # This is 1.0
rate_2012_q4 = rate_2013_q1 / 0.9832  #     maybe u
rate_2012_q3 = rate_2012_q4 / 1.0277
rate_2012_q2 = rate_2012_q3 / 1.0279
rate_2012_q1 = rate_2012_q2 / 1.0279
rate_2011_q4 = rate_2012_q1 / 1.076
rate_2011_q3 = rate_2011_q4 / 1.0236
rate_2011_q2 = rate_2011_q3 / 1
rate_2011_q1 = rate_2011_q2 / 1.011
```

但是，在 Kernels 上，也有人给出了一些我个人认为没有什么逻辑的特征并对此加以一个 Magic Number 作为系数，我尝试把这个特征加入到训练中，提交到 Kaggle 上确实会对预测有一定的提升作用。但是我当时认为那个特征是用两个现实生活中毫无关系的特征组合起来并且乘上 Magic Number，所以我个人无法苟同。我认为 Kernels 上有部分提出的 Magic Number 只是为了拟合那 35%的数据而有意调整出来的（即可能对 35%的数据有故意拟合的现象），可能对另外 65%的数据以及其他数据的预测并不能起到很好的作用，所以我放弃了那些我个人认为的不太合理的 Magic Number。

七、个人总结

这是我第一次数据挖掘的实战接触，之前在 Kaggle 上做过两次作业，是数据挖掘的基础作业，个人实现线性回归，逻辑回归等算法而已，并没有真正到进行比赛的地步。

第一次进行比赛，可以说一开始是一头雾水，不知道比赛的思路是什么，上网查看一些 Kaggle 比赛的流程作为自己比赛的一个依据，参照着 Kernels 和 Discussion 上的一些建议，最终完成了这次比赛，也积累了不少数据挖掘比赛的经验。

我个人觉得这次实训设置得很合理，合理之处在于一开始让我们先体验了一个 Two Sigma 比赛，让我们首先了解了 xgboost 的使用，这对于之后这个比赛的使用有所帮助。

在比赛的过程中，其实我作为一个新手，更多的是在进行几个工作：可视化数据；寻找新的特征；调整模型参数等。

在可视化数据的时候，我个人觉得可视化数据对我有一定的启示作用。

其一，一般数据挖掘比赛给出的数据文件都很庞大，而且都是文本型，我们很难用肉眼去判断规律，但是通过一些库提供的函数，我们可以将特征与标签，特征与特征之间的关系利用图表来表示，有助于我们直观地感受数据中的规律。

其二，可视化数据可以让我发现一些重要特征的脏数据，这点是什么重要的，因为如果一个特征有一些与大多数数据大相径庭的脏数据的话，会对训练产生副作用，副作用可大可小，但是通过可视化数据，我们能够尽量的降低这种副作用。

其三，可视化数据，有时候需要自己有一定的思考也要有一定的耐心，通过组合一些现实中我们认为可能会互相影响的特征，将其与标签的关系可视化出来，有助于我们挖掘新的特征，就比如这次实验过程中用到的年月的序列，房间平均大小等新特征。

寻找新特征的过程中，其实是一个不断试错来找到对的特征的过程，我在 Kernels 找到了不少人们提出的特征，也在同学们的讨论和展示中找到一些特征，对于这些特征，我都进行训练，观察模型最后的效果，有好有坏，但是试错这个阶段不可避免，而且也只有这个阶段才能提升自己的预测效果。

还有模型调参部分，在比赛的最后一天的时候，我的成绩一直停留在一个瓶颈，没办法通过特征的增删来改变，也不知道怎么办，本来就就此打住，但后来想想不如试试看调整最后三个预测结果的加权值来看看能不能提升，然后尝试调整了第一个和第二个模型的加权，就得到了很大的提升；之后调整第三个模型的加权，反而是提交的结果变差了。所以最后只调整了第一个第二个模型的加权。

本次实验过程中，还有一个重点是 Magic Number，在 Magic Number 的调整过程中，我觉得有一些不符合现实的 Magic Number，盲目地为了拟合 35%的预测数据而强行调整出来的 Magic Number，我直接抛弃了这部分 Magic Number。但是有的我不确定我还是选择使用了，结果显而易见，在最终预测另外 65%的数据的时候，我的排名下降了 100 名，我觉得其中很大一部分在于 Magic Number 并不具备普适性导致的。所以在比赛过程中我觉得还是要更主要新特征的挖掘和模型参数的调整，这才是具备普遍性的方式，而不能去迎合预测数据，往往会带来一个更坏的结果。

总而言之，这次比赛我经历了很多失败的尝试，比如，增加了无效的变量（经纬度等）；开始只用单个 xgboost 模型；过分依赖于 Magci Number；对 xgboost 模型参数的调整。这些尝试都使我的预测结果有不同程度的下降，但并不妨碍我对整个流程的学习，其实反而能让我学到更多调试的方法。

我也通过比赛了解了数据挖掘的基本流程，也在 Kernels 学到了很多有用的技巧，其实也是参考了很多个参赛者的代码实现，毕竟是一个完全的新手，我决定在这之后自己再多参加一些比赛，积累数据挖掘比赛的经验，希望以后能够继续往这个方向深造。

八、参考文献及 Github 链接

参考文献：

https://zhuanlan.zhihu.com/p/27424282?utm_source=wechat_session&utm_medium=social

https://www.kaggle.com/agzamovr/a-very-extensive-exploratory-analysis-in-python

https://www.kaggle.com/captcalculator/a-very-extensive-sberbank-exploratory-analysis

https://www.kaggle.com/sudalairajkumar/simple-exploration-notebook-sberbank

Github 链接：

https://github.com/OtherwiseCY/Sberbank-Russian-Housing-Market.git