**华东理工大学 2019 至 2020 学年（第 1 学期）**

**研究生《模式识别原理与应用》课程论文 2019 年 11 月**

开课学院： 信息学院 任课教师： 赵海涛

考生姓名： 余芙丽 学号：\_Y30190701\_ 成绩：

|  |
| --- |
| 论文题目： 房价预测比赛 |
| 论文要求：   1. 方法部分要针对问题对使用的方法进行较为详细的描述 2. 实验部分要先给出实验数据的描述，实验结果有相关的图表和分析 3. 给出自己的结论，对使用方法的优缺点进行总结 |
| 教师评语：  教师签字：  年 月 日 |

**房价预测比赛**

余芙丽

**一、房价预测比赛简介**

房价一直是人们心中关心的话题。要求购房者描述他们的梦想中的房子，他们可能不会从地下室天花板的高度或与铁路的距离开始。但是，这场比赛的数据集证明，除了卧室或白色栅栏的数量，影响价格谈判的因素很多。

本项目中79个特征变量（几乎）描述了爱荷华州埃姆斯的住宅的方方面面，通过这些因素，我们来对房价进行预测。

**二、整体解决方案**

房价预测问题是一个回归问题，本文选用XGBoost模型来解决问题。首先通过特征选择来选出对房价影响较大的几个因素，然后进行数据清洗，最后用这几个因素对模型进行训练，然后用测试集数据来测试，回归出预测房价。

**2.1 数据导入**

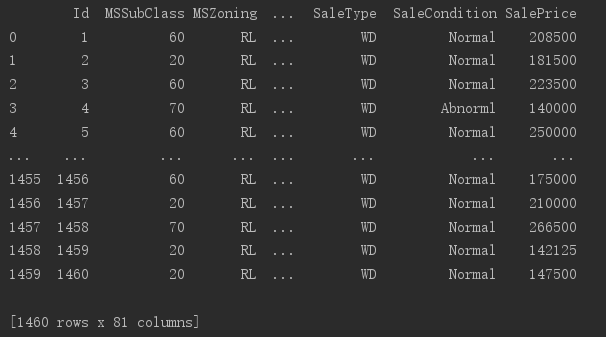
首先导入程序必要的包，

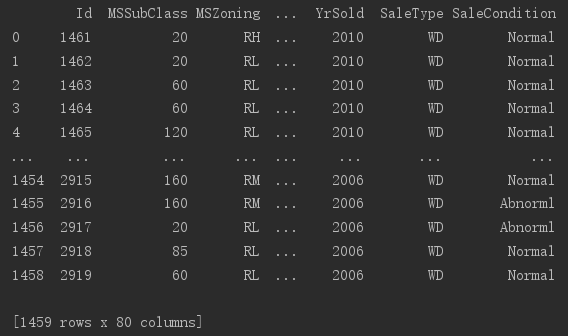
|  |
| --- |
| import pandas as pd  import matplotlib.pyplot as plt  import numpy as np  from xgboost import XGBRegressor  import seaborn as sns  from sklearn.preprocessing import Imputer  import warnings  warnings.filterwarnings('ignore') |

数据从Kaggle下载。数据包括训练数据train.csv，和测试数据test.csv。

同时查看训练数据和测试数据的结构。

|  |
| --- |
| train = pd.read\_csv("train.csv") test = pd.read\_csv("test.csv") print(train) print(test) |

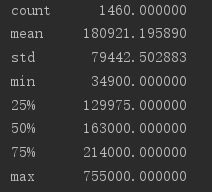




其中，训练数据有1460组，测试数据有1459组，有79个影响变量。

同时对房价进行分析：

|  |
| --- |
| print(train['SalePrice'].describe()) |

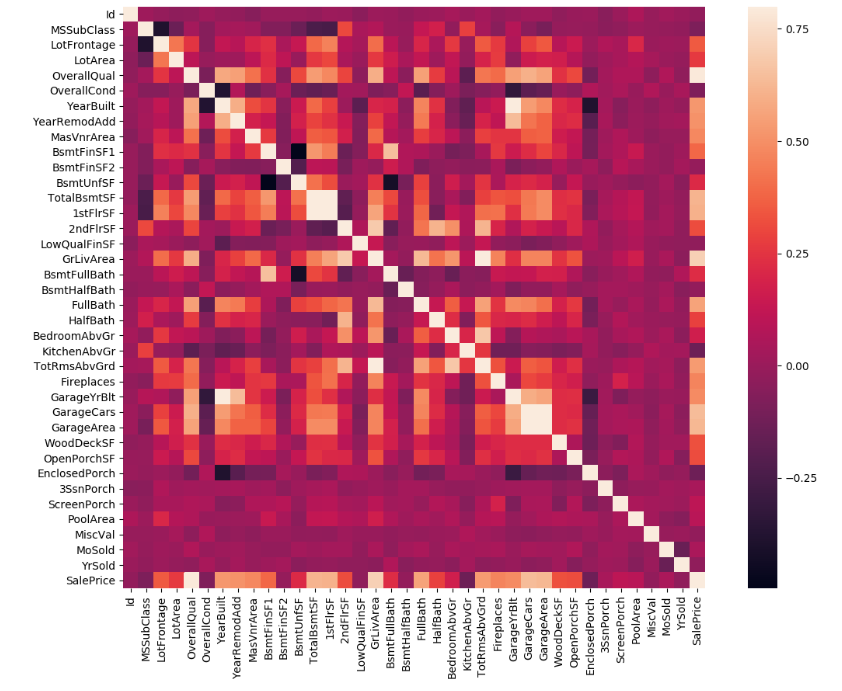


从图中可以看出房价没有缺失值，可以得到房价的平均值，最大最小等数值。

**2.2 特征选择**

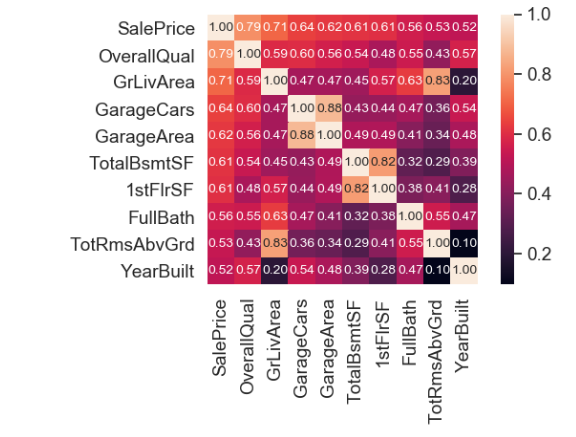
画出相关系数矩阵，查看各个特征之间的关系。

|  |
| --- |
| corrmat = train.corr() f, ax = plt.subplots(figsize=(12, 9)) sns.heatmap(corrmat, vmax=.8, square=True); plt.show() |



从图中可以看出，颜色越浅，两者的相关性越大，选出与SalePrice相关性最大的几个特征，画出热力图分布。

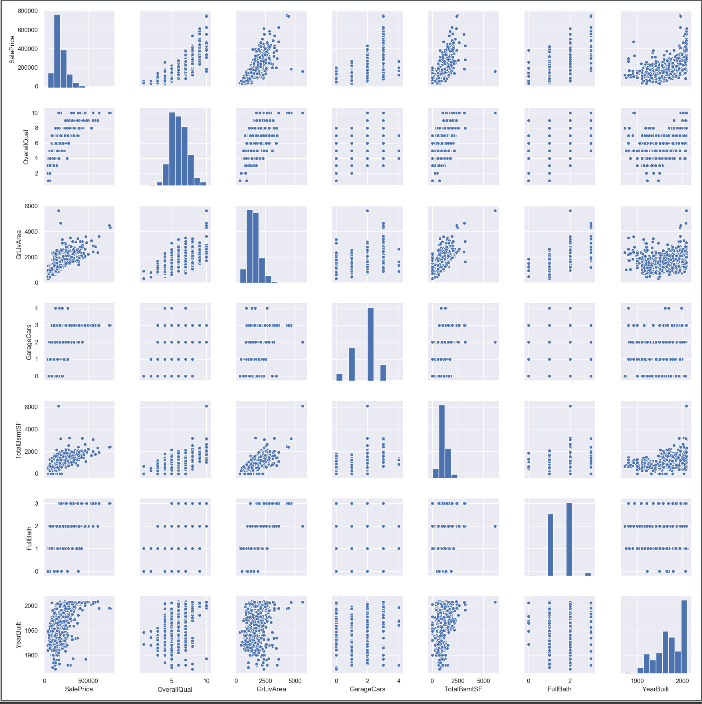
|  |
| --- |
| k = 10 #number of variables for heatmap  cols1 = corrmat.nlargest(k, 'SalePrice')['SalePrice'].index  cm = np.corrcoef(train[cols1].values.T)  sns.set(font\_scale=1.25)  hm = sns.heatmap(cm, cbar=True, annot=True, square=True, fmt='.2f', annot\_kws={'size': 10}, yticklabels=cols1.values, xticklabels=cols1.values)  plt.show() |



从图中可以看出：OverallQual，GrLivArea以及TotalBsmtSF与SalePrice有很强的相关性；GarageCars和GarageArea也是相关性比较强的变量，车库中存储的车的数量是由车库的面积决定的，它们就像双胞胎，所以不需要专门区分 GarageCars和GarageAre，所以我们只需要其中的一个变量。这里我们选择了GarageCars，因为它与SalePrice的相关性更高一些；TotalBsmtSF和1stFloor与上述情况相同，我们选择TotalBsmtS；TotRmsAbvGrd 和 GrLivArea 也是变量中的双胞胎。

所以最后选出六个与SalePrice相关性最大的特征：'OverallQual', 'GrLivArea','GarageCars', 'TotalBsmtSF', 'FullBath', 'YearBuilt'。画出SalePrice 和相关变量之间的散点图。

|  |
| --- |
| sns.set()  cols = ['SalePrice', 'OverallQual', 'GrLivArea', 'GarageCars', 'TotalBsmtSF', 'FullBath', 'YearBuilt']  sns.pairplot(train[cols], size = 2.5)  plt.show(); |



尽管我们已经知道了一些主要特征，这一丰富的散点图给了我们一个关于变量关系的合理想法。

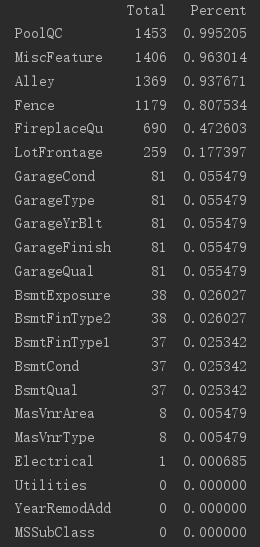
其中，TotalBsmtSF 和 GrLiveArea 之间的散点图是很有意思的。我们可以看出这幅图中，一些点组成了线，就像边界一样。大部分点都分布在那条线下面，这也是可以解释的。地下室面积和地上居住面积可以相等，但是一般情况下不会希望有一个比地上居住面积还大的地下室。

SalePrice 和 YearBuilt 之间的散点图也值得我们思考。在“点云”的底部，我们可以观察到一个几乎呈指数函数的分布。我们也可以看到“点云”的上端也基本呈同样的分布趋势。并且可以注意到，近几年的点有超过这个上端的趋势。

**2.3 缺失数据检测**

缺失数据意味着样本大小的缩减，这会阻止我们的分析进程。除此之外，以实质性的角度来说，我们需要保证对缺失数据的处理不会出现偏离或隐藏任何难以忽视的真相。首先我们分析哪些特征出现了缺失数据。

|  |
| --- |
| total = train.isnull().sum().sort\_values(ascending=False)  percent = (train.isnull().sum()/train.isnull().count()).sort\_values(ascending=False)  missing\_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])  print(missing\_data.head(22)) |



从图中可以看出，我们需要的六个特征没有出现缺失，出现缺失的特征与SalePrice的相关性较小，所以不需要进行缺失值的处理。

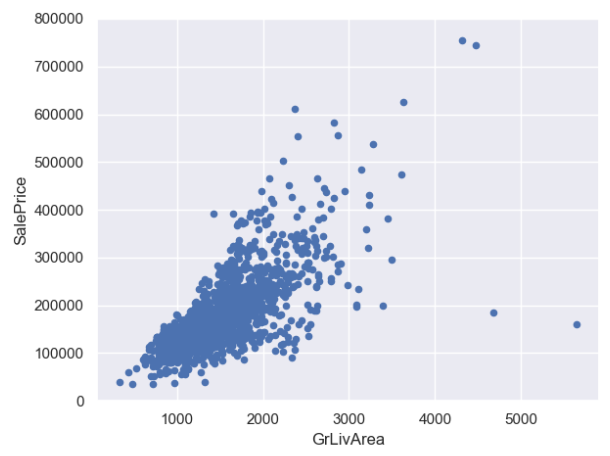
sklearn的preprocessing包中包含了对数据集中缺失值的处理，主要是应用Imputer类进行处理。我们对测试数据进行处理。

|  |
| --- |
| my\_imputer = Imputer()  test\_x = my\_imputer.fit\_transform(test\_x) |

**2.4 数据处理**

1、首先进行GrLivArea 和 SalePrice 双变量分析，画出相关散点图。

|  |
| --- |
| var = 'GrLivArea'  data = pd.concat([train['SalePrice'], train[var]], axis=1)  data.plot.scatter(x=var, y='SalePrice', ylim=(0,800000));  plt.show(); |



从图中可以看出：

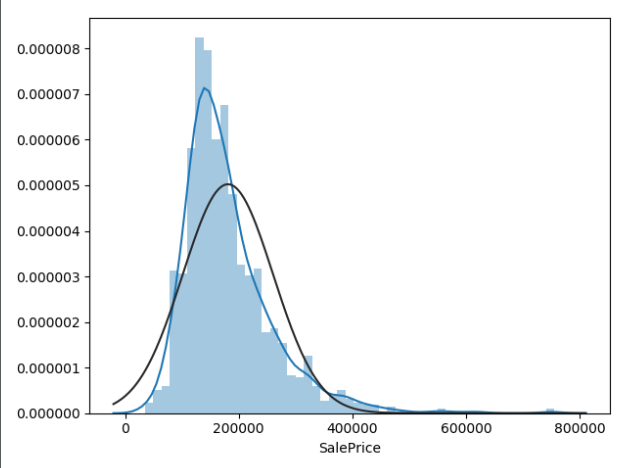
有两个离群的 GrLivArea 值很高的数据，我们可以推测出现这种情况的原因。或许他们代表了农业地区，也就解释了低价。这两个点很明显不能代表典型样例，所以我们将它们定义为异常值并删除。

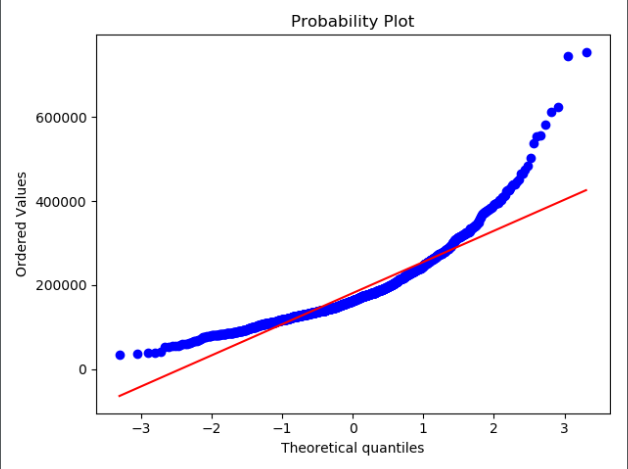
|  |
| --- |
| train = train.drop(train[(train['GrLivArea'] > 4000) & (train['SalePrice'] < 300000)].index) |

图中顶部的两个点是七点几的观测值，他们虽然看起来像特殊情况，但是他们依然符合整体趋势，所以我们将其保留下来。

2、SalePrice是我们需要预测的，因此对这个变量做一些分析。我们一般都期望数据的分布符合正态，但是有时候实际得到的数据不是这样的。关注两点：直方图和正态概率图，直方图考虑峰度和偏度，正态概率图要求数据分布应紧密跟随代表正态分布的对角线。

|  |
| --- |
| sns.distplot(train['SalePrice'], fit=norm);  fig = plt.figure()  res = stats.probplot(train['SalePrice'], plot=plt)  plt.show() |

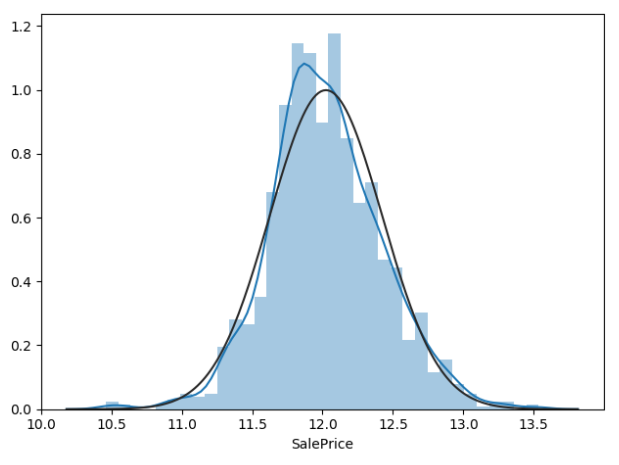


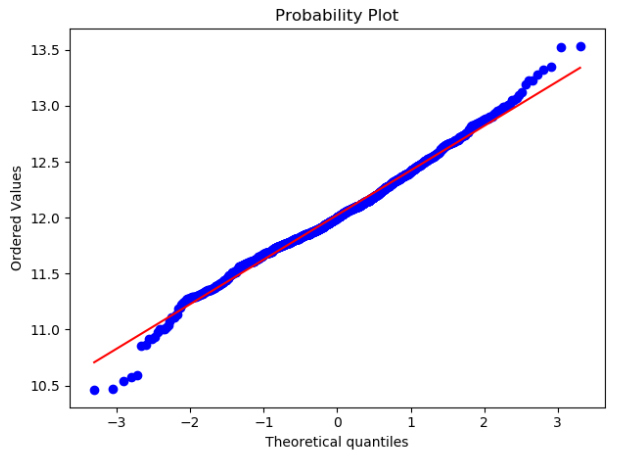


可以看出，房价分布不是正态的，显示了峰值，正偏度，但是并不跟随对角线。可以用对数变换来解决这个问题。

|  |
| --- |
| train['SalePrice']= np.log(train['SalePrice'])  sns.distplot(train['SalePrice'], fit=norm);  fig = plt.figure()  res = stats.probplot(train['SalePrice'], plot=plt)  plt.show() |

变换后的直方图和正态概率图为：





**2.5 One-Hot编码**

使用pandas的dummy方法来进行数据独热编码，并形成最终的训练和测试数据集。

|  |
| --- |
| cols2=['OverallQual', 'GrLivArea', 'GarageCars', 'TotalBsmtSF', 'FullBath', 'YearBuilt']  A=train[cols2]  B=test[cols2]  A = pd.get\_dummies(A)  B = pd.get\_dummies(B) |

**2.6 训练和测试**

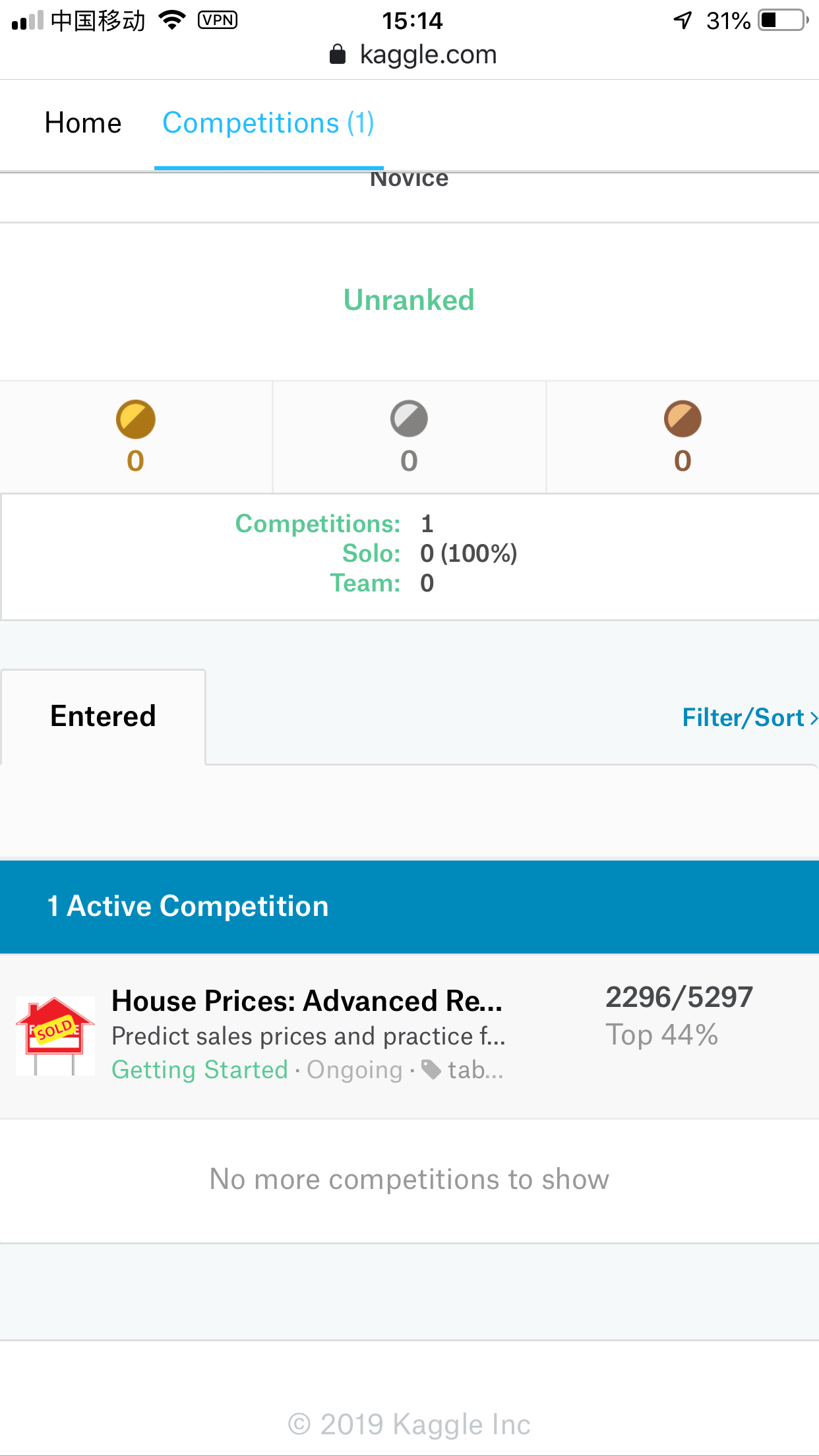
运用XGBoost回归模型对选出的六个特征数据进训练，之后用测试数据回归出我们的预测房价。

|  |
| --- |
| xgb\_model = XGBRegressor(n\_estimators=1000,learning\_rate = 0.01, early\_stopping\_rounds =5,n\_jobs=4)  xgb\_model.fit(train\_x, train\_y, verbose=False)  predictions = xgb\_model.predict(test\_x)  pre = np.round(predictions) |

生成最后的文件mysubmission.csv上传至kaggle。

|  |
| --- |
| subm = pd.DataFrame({'Id':test.Id,'SalePrice':pre})  subm.to\_csv('mysubmission.csv', index = False) |

**3 实验总结**

****

由于第一次接触大数据的处理，阅读了kaggle上的许多笔记，最后合成了自己的一个简单的程序，最后的效果不是特别理想，在44%，可能算法比较简单，没有更好得处理全部数据。

这次的大作业让我接触到了用模式识别解决实际问题，比较有趣味性，也感谢赵老师在这一学期的认真教学。

**附：文件说明**

本次附件一共包含有：

1. 大作业报告；
2. 最终的python程序源码：demo.py；
3. 训练数据：train.csv；
4. 测试数据;test.csv；
5. 最终预测数据：mysubmission.csv