**西雅图King County地区房价预测模型**

利用2014年5月1日至2015年5月27日的西雅图King County地区房屋购买交易数据构建该地区的房价预测模型。原数据含有从2014年5月1日至2015年5月27日的西雅图King County地区总共共21613条房屋购买交易数据，其中有21个特征。

以下描述了用机器学习的方法预测房价的过程和思路

**1、数据清洗：**

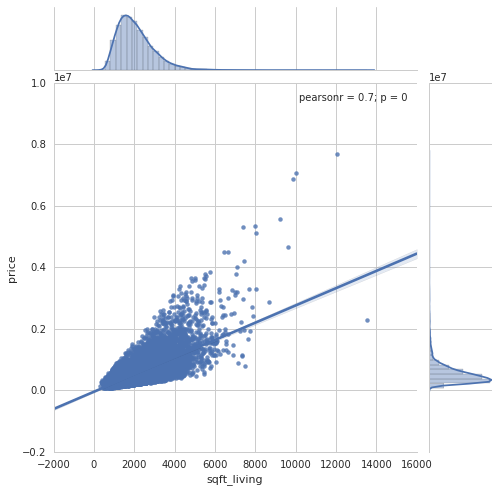
删除掉'id', 'date'两列不需要的特征。

该模型暂时没有考虑季节可能对房价造成的影响，而且数据是短期内的数据，所以删除了购房时间

**2、特征分析：**

**连续特征：**

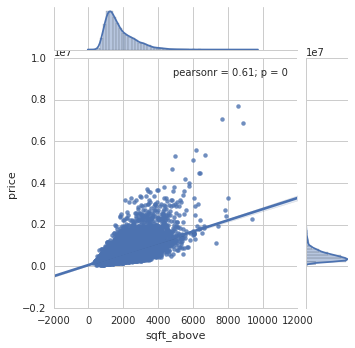
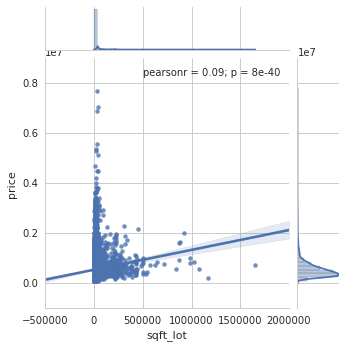
用散点图和相关系数展示特征之间的潜在关系

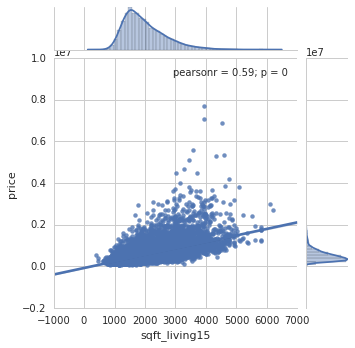
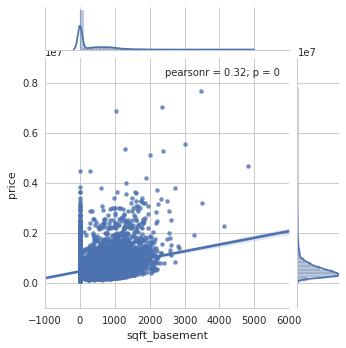
****

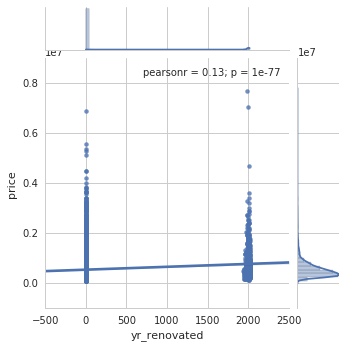
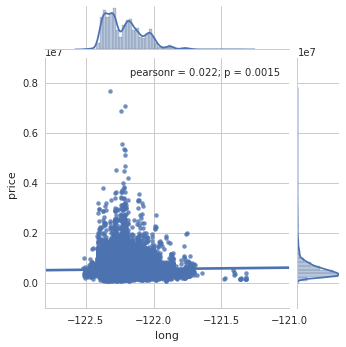
可以发现它们之间有很大的正相关性，所以sqft\_living是一个很好的预测房价的特征。

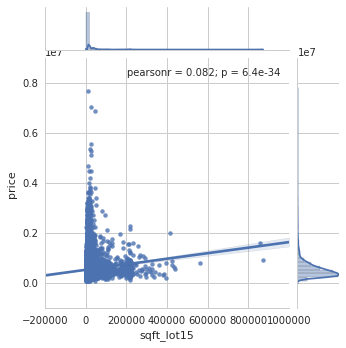
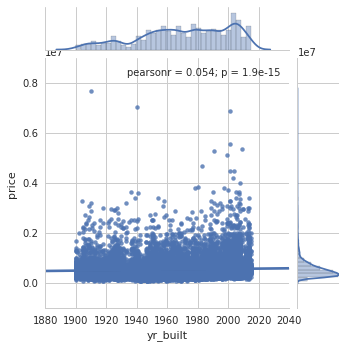
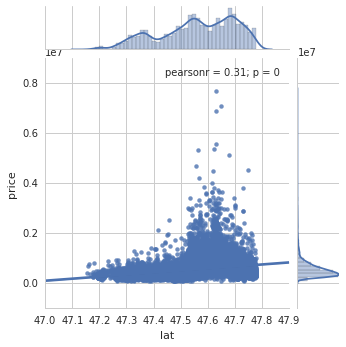
sqft\_above ( sqft\_above = sqft\_living - sqft\_basement)、sqft\_basement、

sqft\_living15（the average house square footage of the 15 closest neighbours）、sqft\_lot15（the average lot square footage of the 15 closest neighbours）、yr\_built、yr\_renovated、lat、long





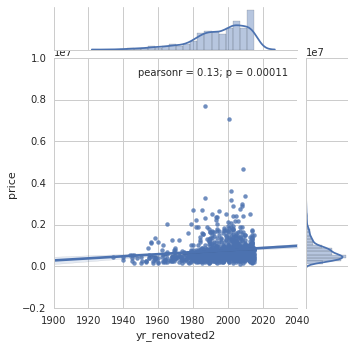
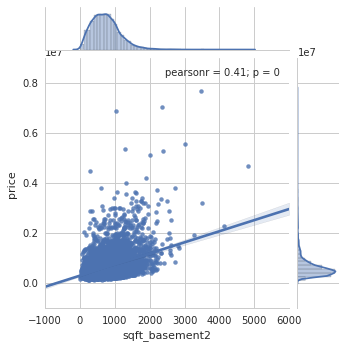
 



通过图示可以看到sqft\_lot、sqft\_lot15、yr\_built和price之间有较差的相关性

另外可以在图中看到sqft\_basement有很多的0值（可能代表这里没有basement）。同样的，在yr\_renovated也有很多的零值。

如果删除sqft\_basement和yr\_renovated特征中的0值，保留其它值，则如下图所示：



可以看到房价与地下室的面积有一定的相关性（如果地下室存在），房价与翻修的年代有一定的弱相关性（如果房子以前翻修过）。

因此可以考虑将sqft\_basement和yr\_renovated转换为二分值特征（即当basement不存在时为0，当basement存在时为1）。

我们将对它进行分类变量分析。

但是首先，我们再仔细分析这两个变量：sqft\_above 和 sqft\_living15，它们好像和price强相关，让我们分析它们的关联性（利用sqft\_living）

分类：

现在我们分析房价和分类变量之间的关系。

**3、数据转换**

将sqft\_basement和yr\_renovated转换为二分值特征并将其作为新特征（即当basement不存在时为0，当basement存在时为1）。原来的两个特征保留，因为根据上面对sqft\_basement和yr\_renovated的分析表明，它们与房价有关联。

**4、转换分类变量**

将分类特征floors、view、condition、grade分别转换成多个对应的二分值特征（比如将floors替换为floors#1.0、floors#1.5、floors#2.0、floors#2.5、floors#3.0、floors#3.5六个特征，存在的标1，不存在的标0）。

zipcode里总共有70个不同的邮政编码。首先算出每个邮政编码对应的平均房价，选取最高的前六个平均房价所对应的zipcode，它们分别是'zipcode#98004','zipcode#98102','zipcode#98109','zipcode#98112','zipcode#98039','zipcode#98040'。并将它们转换成二分值特征。

最终的特征和对应的格式为：

price float64

bedrooms int64

bathrooms float64

sqft\_living int64

sqft\_lot int64

waterfront int64

sqft\_above int64

sqft\_basement int64

yr\_built int64

yr\_renovated int64

lat float64

long float64

sqft\_living15 int64

sqft\_lot15 int64

basement\_present int64

renovated int64

floors#1.0 float64

floors#1.5 float64

floors#2.0 float64

floors#2.5 float64

floors#3.0 float64

floors#3.5 float64

view#0 float64

view#1 float64

view#2 float64

view#3 float64

view#4 float64

condition#1 float64

condition#2 float64

condition#3 float64

condition#4 float64

condition#5 float64

grade#1 float64

grade#3 float64

grade#4 float64

grade#5 float64

grade#6 float64

grade#7 float64

grade#8 float64

grade#9 float64

grade#10 float64

grade#11 float64

grade#12 float64

grade#13 float64

zipcode#98004 float64

zipcode#98102 float64

zipcode#98109 float64

zipcode#98112 float64

zipcode#98039 float64

zipcode#98040 float64

**4、划分数据集：**

按照60%/20%/20%的比例，将数据划分为训练集、验证集和测试集。

**5、回归模型：**

在训练集上训练模型，并通过测试集上RMSE的表现来评价模型的好坏。

5.1 简单回归：

如果训练一个简单的线性回归模型，来评价哪个是最好的估计房价的特征。

|  | **特征** | **RMSE** | **intercept** | **系数** |
| --- | --- | --- | --- | --- |
| **2** | sqft\_living | 268279.643883 | 277.364130 | -3.673818e+04 |
| **5** | sqft\_above | 304131.310592 | 266.306764 | 6.461714e+04 |
| **11** | sqft\_living15 | 320686.541323 | 314.359911 | -8.502590e+04 |
| **1** | bathrooms | 324082.781919 | 246523.891877 | 1.863279e+04 |
| **21** | view#0 | 356019.001320 | -435033.777431 | 9.322014e+05 |
| **6** | sqft\_basement | 357843.745395 | 258.126523 | 4.642966e+05 |
| **40** | grade#11 | 357964.423743 | 965286.415396 | 5.222663e+05 |
| **39** | grade#10 | 360773.700418 | 556992.601325 | 5.107024e+05 |
| **0** | bedrooms | 361295.375626 | 117579.891853 | 1.436815e+05 |
| **9** | lat | 365041.433662 | 814499.981062 | -3.819789e+07 |

经过比较，发现如果只使用一元线性回归模型， sqft\_living就是最好的输入特征。

**5.2多元回归：**

人工的新特征：  
为了进一步优化模型的拟合度，这里通过特征的高阶多项式等方式生成几个新的特征。这里新特征为：sqft\_living^3、bedrooms^2、bed\_bath\_rooms、log\_sqft\_living

sqft\_living^3为sqft\_living的3次方

bedrooms^2为bedrooms的2次方

bed\_bath\_rooms为bedrooms乘以bathrooms

log\_sqft\_living为log(sqft\_living)

下面利用贪心算法（[greedy](javascript:void(0);) [algorithm](javascript:void(0);)）forward来选取特征，每次选取一个，下一个特征的选取是在剩余的特征中选取RMSE下降最多的特征。

|  | **feature** | **train\_error** | **validation\_error** |
| --- | --- | --- | --- |
| **0** | sqft\_living\_squared | 249721.440480 | 268747.213950 |
| **1** | lat | 229419.062478 | 252069.707424 |
| **2** | waterfront | 219240.069871 | 241979.268406 |
| **3** | zipcode#98004 | 208411.508254 | 229000.747130 |
| **4** | zipcode#98112 | 202604.911359 | 225124.826562 |
| **5** | zipcode#98039 | 196885.794407 | 217328.215296 |
| **6** | view#0 | 191119.193990 | 209679.561580 |
| **7** | zipcode#98040 | 186849.129730 | 204450.501487 |
| **8** | sqft\_living15 | 185134.594156 | 197733.047694 |
| **9** | grade#13 | 183383.653232 | 197321.800871 |
| **10** | condition#5 | 181891.092067 | 196541.377903 |
| **11** | grade#11 | 180532.105926 | 194534.976260 |
| **12** | grade#10 | 178492.121187 | 190462.363671 |
| **13** | grade#12 | 175704.660910 | 185437.534555 |
| **14** | grade#9 | 172510.818377 | 179597.162787 |
| **15** | grade#8 | 170635.242038 | 176297.524718 |
| **16** | yr\_built | 167977.636383 | 173665.533961 |
| **17** | bathrooms | 166456.154777 | 170353.745230 |
| **18** | view#4 | 165260.808366 | 170390.119346 |
| **19** | grade#7 | 164246.940788 | 168940.611700 |
| **20** | zipcode#98109 | 163458.010133 | 168140.236755 |
| **21** | zipcode#98102 | 162926.763099 | 167535.616923 |
| **22** | view#3 | 162643.227969 | 167797.150367 |
| **23** | floors#1.0 | 162364.749446 | 167570.147704 |
| **24** | condition#4 | 162071.950582 | 167362.587289 |
| **25** | sqft\_living\_cubed | 161840.708282 | 177707.813279 |
| **26** | bed\_bath\_rooms | 161677.933944 | 176365.516340 |
| **27** | long | 161531.531528 | 176130.809976 |
| **28** | yr\_renovated | 161431.315216 | 176382.969484 |
| **29** | renovated | 161079.737982 | 176033.595178 |
| **30** | floors#2.5 | 160994.783556 | 175999.664770 |

通过贪心算法的运行记录表可以看到，虽然train\_error是一直再降低，而validation\_error在贪心算法运行到第25轮时有最小值。说明25轮以后产生了过度拟合。因此选用第25轮之前的25个特征。

通过这25个特征所得到的模型在测试集上的表现：

RMSE = 169561.235885, R2 = 0.803

下面不考虑使用贪心算法，而使用正则化的方法。即选用所有的特征，然后通过正则化项对特征的系数进行压缩。

**5.3岭回归：**

通过调节L2惩罚系数来控制模型的复杂度。

选取一系列的L2惩罚系数，来计算在验证集上的RMSE值。选取其中最小的RMSE值所对应的L2惩罚系数作为模型的系数。

经过不断的筛选，最后L2选为0.07。测试集上的表现:

RMSE = 171566.791859，R2 = 0.799

**5.4 Lasso回归：**

通过调节L1惩罚系数来选取特征。

选取一系列的L1惩罚系数，来计算在验证集上的RMSE值。因为lasso方法倾向于稀释性偏小的方案，因此并不选择最优的L1，而是选取比最优值稍微小一点的值来筛选最终特征。

最后L1选为14.93，非零的特征个数为39个，测试集上的表现：

RMSE = 171369.075713, R2 = 0.799

**结论：**

通过比较使用贪心算法时有最小的值