日本房屋价格预测 - RFLR 模型

郭牧豪

2020 年 11 月

1 摘要

“居者有其屋”是人们的基本生活需求，房屋作为人们的主要财产，其价格一直以来是人们关注的焦点，尤其是人口密集型国家和地区，像中国，日本，美国等，人们对房屋价格的价格尤为关注。本报告根据 2005 年到 2019 年日本国土交通省（MLIT）调查的日本房地产交易价格的记录，建立房屋价格预测模型，并根据地区的房屋价格对地区的发展提出可行性建议。本模型由数据预处理，数据降维分析，分类预测，精准预测等模块组成。模型以 LDA 分类，逻辑回归，随机森林，支持向量机分类器和多种回归算法为基础，采用 PCA 或 LDA 进行降维分析。优化出一种双训练双预测的机器学习模型，即 RFlR 模型。

2 介绍

2.1 题目分析

提供的数据包括 2005 年到 2019 年日本国土交通省（MLIT）调查的日本房地产交易价格的记录。数据分为县级代码和房价数据两部分。房价数据有 47 个文件，分别代表 47 个县级地区的房价记录，每个文件中，有 38 列数据，包括 1 个序号列，一个房价标签数据列，和 36 个“特征”列。

2.2 模型概论

对每个县的数据进行单独分析。分别得出每个县市地区的房屋价格预测。分为数据预处理阶段和模型分类预测和模型精准预测 3 个阶段。

2.2.1 数据预处理

- 数据读取：以县级地区为单位，读取每一个数据文件
- 特征编码：对于字符串特征，进行直接编码转化为浮点型特征，对于特殊性浮点型特征如“TimeToNearestStation”，由于其内部有些数据由字符串表示，但其本质是浮点型特征，因此进行人工赋值处理。
- 标签分类：根据每个数据集的 25% 分位数、50% 分位数、75% 分位数，将数据集分为 4 类并赋予对应的标签。初步转化为分类问题，避免了数据不平衡问题。与此同时，根据房屋价格由低到高，将数据分为 4 部分，每一部分代表一个价格区间的数据，以便之后精准预测使用。

- 训练集测试集分离：将所有数据分为测试集和训练集。另外，上一步中的用于精准预测的四部分小数据集作为第二阶段的训练集，根据不同的策略选取不同的小训练集进行训练。
- 降维分析：包括 PCA 主成分分析和 LDA 线性判别分析，得到新的特征或选择最好的特征，并且提高计算速度。

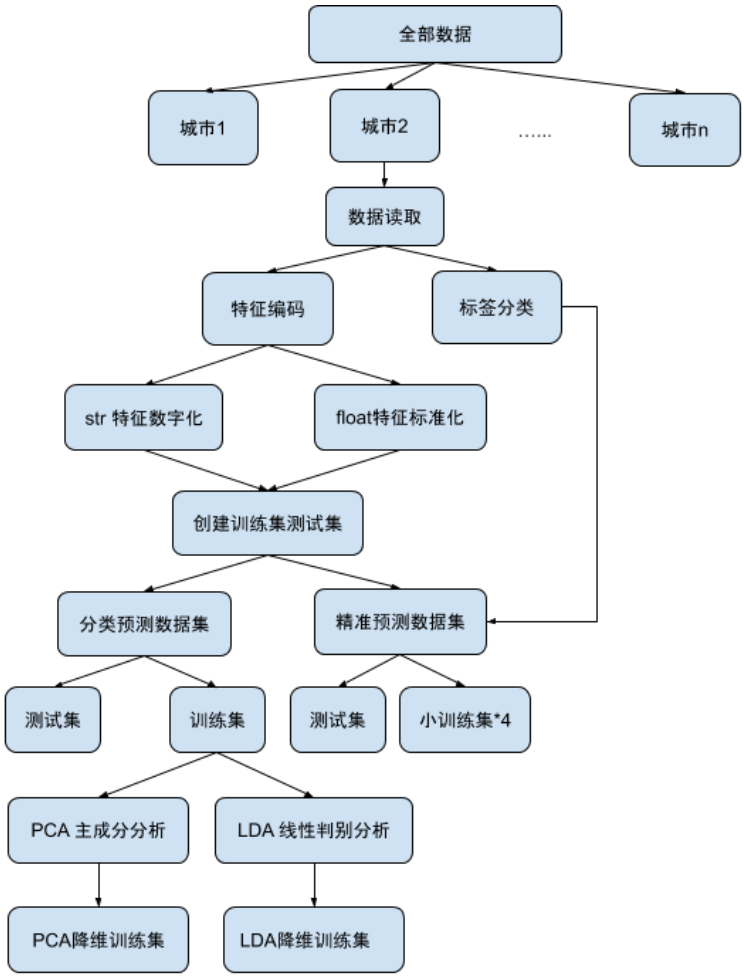

图 1: 数据预处理

2.2.2 模型分类预测

比较不同的分类器，选取预测成功率最高的分类器。以该分类器最为精准预测的前驱分类器。使用的分类器如下：

- LDA
- Logistic Regression with L1 penalty
- Logistic Regression with L2 penalty

- Random Forest Tree
- Support Vector Classifier

2.2.3 模型精准预测

根据模型分类预测的结果，得到预测结果类别，根据此类别，运用对应的精准预测训练集作为精准预测模型的训练集。对于每个小训练集进行数据标准化处理以及 PCA 降维，最后运用不同的回归函数进行精确分类。

- Linear Regression
- Polynomial Regression Degree2
- Polynomial Regression Degree3
- Polynomial Regression Degree4

2.2.4 模型整体流程图

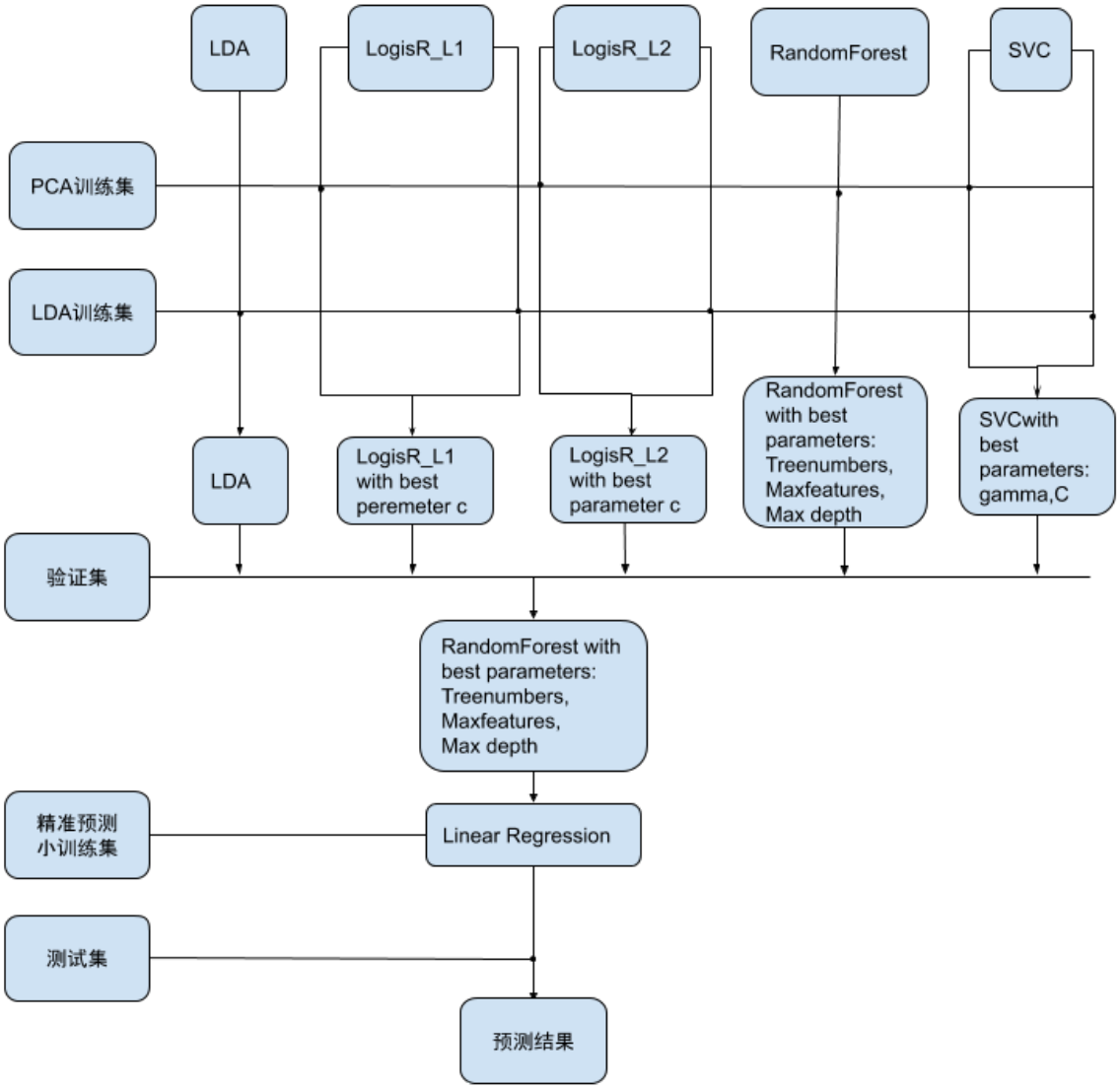

图 2: 模型选择、预测流程图

3 实验执行

3.1 数据预处理

3.1.1 数据集特征选取

原始数据共 36 列，将 TradePrice 作为标签列，No 是序号列，去掉重复列后，为最大程度的保留原始数据，共选择 30 列作为特征列，其中 10 列为 Float 型数据，3 列为 Boolean 型数据，17 列为 String 型数据。选择的特征列如表1 所示。

表 1: 特征变量及类型

特征	类型
Type	String
Region	String
MunicipalityCode	String
Prefecture	String
DistrictName	String
NearestStation	String
TimeToNearestStation	Float
MinTimeToNearestStation	Float
MaxTimeToNearestStation	Float
FloorPlan	String
Area	Float
AreaIsGreaterFlag	Boolean
LandShape	String
Frontage	Float
FrontageIsGreaterFlag	Boolean
TotalFloorArea	Float
TotalFloorAreaIsGreaterFlag	Boolean
BuildingYear	String
PrewarBuilding	String
Structure	String
Use	String
Purpose	String
Direction	String
Classification	String
Breadth	Float
CityPlanning	String
CoverageRatio	Float
FloorAreaRatio	Float
Year	String
Quarter	String

3.1.2 价格标签分析

分别选取 “TradePrice” 列的 25% 分位数 (A)、50% 分位数 (B)、75% 分位数 (C) 作为分类边界，将数据分为 4 类，在原始数据中添加 “TradePrice_class” 列，作为分类标签列，类标签分别为 “0 - A”，”A - B”，”B - C” 和”C - +”。

3.1.3 特征编码

- String 特征编码：使用 LabelEncoder() 函数将 String 特征转化为 Int 特征。
- 特殊 Float 特征：对于"TimeToNearestStation" 特征，取中间值作为数值特征" 30-60minutes "取 45 min，"1H-1H30" 取值 75 min，"1H30-2H" 取值 105 min
- 空值 Float 类型特征：舍弃该样例
- 常规 Float 特征：不变

3.1.4 训练集测试集分离

按照 7：3 的比例，随机选取样本，构成训练集和测试集。训练集用于训练模型。对于总训练集，又将其按照不同的 "TradePrice_class" 类别，分为 4 个小训练集，分别记为 "df_0_A" , "df_A_B" , "df_B_C" 和 "df_C_D" 便于之后的精准预测使用。

3.1.5 PCA 主成分分析

使用 PCA 主成分分析，选取 10 个影响最大的特征组合，作为训练特征。

3.1.6 LDA 线性判别分析

使用 LDA，由于一共分为 4 类，所以共选取 3 个特征，作为训练特征。

3.2 模型分类预测

3.2.1 LDA

LDA 不仅可以用来降维数据，也可以用来作为一个分类器，直接用在降维步骤中训练好的 LDA 模型作为分类器进行预测。

3.2.2 Logistic Regression with L2 penalty

我们希望最小化损失函数，增加 L2 惩戒避免过拟合。使用交叉验证选择最佳参数 C，C 作为正则化系数的倒数，共选取 11 个值，分别为 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^{2}, 10^{3}, 10^{4}, 10^{5}。使用 10 折交叉验证。

3.2.3 Logistic Regression with L1 penalty

和 Logistic Regression with l2 penalty 相似，我们希望最小化损失函数，增加 L1 惩戒避免过拟合。使用交叉验证选择最佳参数 C，C 作为正则化系数的倒数，共选取 11 个值，分别为 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^{2}, 10^{3}, 10^{4}, 10^{5}。使用 10 折交叉验证。

3.2.4 Random Forest

随机森林的预测结果基于森林中每棵树的投票。使用随机森林分类器是因为它可以很好地处理具有特征空间的数据集，特别是多特征的数据。我使用交叉验证来选择森林中的树数，每次迭代的特征数以及树的最大深度。树个数从 100 到 1500，间隔为 50。由于特征的数量应少于特征总数，因此我还使用交叉验证来选择每次迭代的要素数量，即从 6 开始到 10，间隔为 1。对于树的深度，为树的深度设置了三个值：无，50 和 100。使用 10 折交叉验证。

使用交叉验证选择最佳参数 n_estimators”，“max_features”，和” max_depth。使用 10 折交叉验证。最大特征数从 5 到 10，树个数从 100 到 1500，间隔为 50

3.2.5 SVC

假如数据集不是线性可分离的，那么支持向量分类器可以在这种情况下较好地工作。SVC 的目标是拟合提供的数据并返回最合适的超平面，该超平面可以将数据集划分为不同的类。我选择通用内核函数“RBF” 内核，并使用交叉验证来获得最佳参数 “gamma” 和 “C”，它们分别代表 “RBF” 核系数和正则化参数。“C” 的作用是避免过度拟合。

使用交叉验证选择最佳参数 “gamma” 和 “C”。我选择 gamma 值分别为: 0.1, 0.01, 0.001, 0.0001, C 的值分别为: 0.1, 1, 10, 100。并使用 10 折交叉验证。

3.3 模型精准预测

3.3.1 小训练集数据预处理

针对 4 个小训练集，分别进行数据标准化和 PCA 处理，我们之后将陈述不同 PCA 维度对模型的影响。针对整体测试集中的每一个样本，根据该样本的标签 (标签由分类预测得到)，将由小训练集训练好的与其对应的标准化函数和 PCA 函数运用于该样本。生成全部的数据标准化和 PCA 降维后的测试集。

3.3.2 回归模型

分别运用线性回归模型，二阶多项式回归，三阶多项式回归，四阶多项式回归。并探讨 PCA 维度对各回归模型的影响。

3.3.3 精准预测流程图

进一步研究精准预测部分，将图 2（模型选择、预测流程图）中的 Linear Regression 部分进行细化和改进。形成图 3 的精准预测流程图

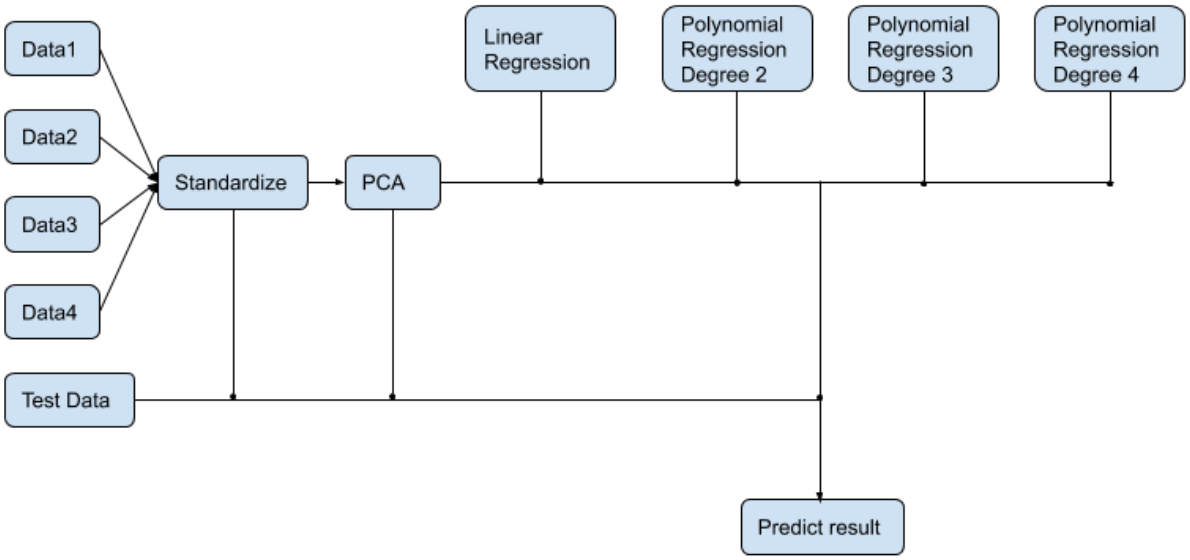

图 3: 精准预测流程图

4 实验结果与结论

4.1 分类模型实验结果

我们以青森🄵（Aomori）为例，不同模型的实验结果如下表 2 所示：

表 2: 实验结果–分类模型预测

算法	预测正确率
LDA	0.679176
Logistic Regression_L2 with LDA	0.677183
Logistic Regression_L2 with PCA	0.661242
Logistic Regression_L1 with LDA	0.542677
Logistic Regression_L1 with PCA	0.538359
Random Forest with PCA	0.715044
SVC with LDA	0.688143
SVC with PCA	0.669212

基于分类预测结果，明显，Random Forest 算法的预测正确最高。我们重点探讨不同的 PCA 维度对 Random Forest 算法的训练时间和预测正确率的影响。结果如表 3 所示。

表 3: Random Forest

Random Forest Algorithms	训练正确率	训练时间 (s)	预测正确率
Random Forest with PCA 10 No crossVal	0.99272	1189.23	0.69645
Random Forest with PCA 10	0.99943	7755.67	0.73364
Random Forest with PCA 20	0.99943	37929.30	0.70641
Random Forest No PCA	0.99957	30529.56	0.90667

可以看出，对于 Random Forest Algorithms, 在 PCA 降维到 10 维且不进行交叉验证的情况下，训练时间最快，为 1189s，但其预测正确率只有 0.69645。在有交叉验证的情况下，可以选出更准确的模型参数。随着 PCA 处理后数据维度的增加，预测正确率没有呈现规律的变化，说明 PCA 处理后的特征维度与预测正确率之间没有显著的关系。

当考虑所有特征时，即不进行 PCA 特征降维的情况下，预测正确率会大幅度提高，可达 0.90667。当训练集为 2 万时，在个人设备上的训练时间达到 8 个多小时。

4.2 精准预测实验结果

基于分类预测模型 Random Forest No PCA 的预测结果, 继续进行精准预测。使用四种回归模型。并探究 PCA 对模型预测结果和模型训练时间的影响。表 4 至表 7 展示出在不同的 PCA 策略下，四种回归模型的 R2 决定系数以及模型训练时间。

表 4: No PCA

Regression Algorithms	时间	R squared 决定系数
Linear Regression	0.1264	0.32935
Polynomial Regression Degree2	10.18656	-1116.62504
Polynomial Regression Degree3	132.13921	-30.57845
Polynomial Regression Degree4	976.26214	-1533.62296

表 5: PCA: 20

Regression Algorithms	时间	R squared 决定系数
Linear Regression	0.11576	0.32073
Polynomial Regression Degree2	1.71544	0.11941
Polynomial Regression Degree3	6.40335	-0.95563
Polynomial Regression Degree4	21.96514	-523.89856

表 6: PCA: 10

Regression Algorithms	时间	R squared 决定系数
Linear Regression	0.12670	0.32067
Polynomial Regression Degree2	1.75862	0.11131
Polynomial Regression Degree3	6.24644	-0.91022
Polynomial Regression Degree4	21.51670	-416.55285

表 7: PCA: 5

Regression Algorithms	时间	R squared 决定系数
Linear Regression	0.11516	0.32063
Polynomial Regression Degree2	1.54499	0.11276
Polynomial Regression Degree3	5.77468	-0.91628
Polynomial Regression Degree4	20.48434	-443.41047

由上表可以看出，在没有数据标准化和 PCA 降维的情况下，随着多项式模型的阶数增加，训练时间急剧增加。对于数据标准化和 PCA 降维之后的数据，随着模型复杂度增加，训练时间不会增加很快。一般来说，当阶数大于 2，模型 R2 决定系数转为负数，模型失去意义。综上考虑，最理想的模型为 PCA 降维到 5 后的 Linear Regression 模型。

为直观化看出预测结果。取测试集前一百个样本的预测结果与实际值比较并绘图，其中每张图代表不同的特征维度，每张子图代表不同的回归模型算法，横坐标表示样本编号，纵坐标表示房屋价格，红色线表示实际价格，蓝色线表示预测价格。

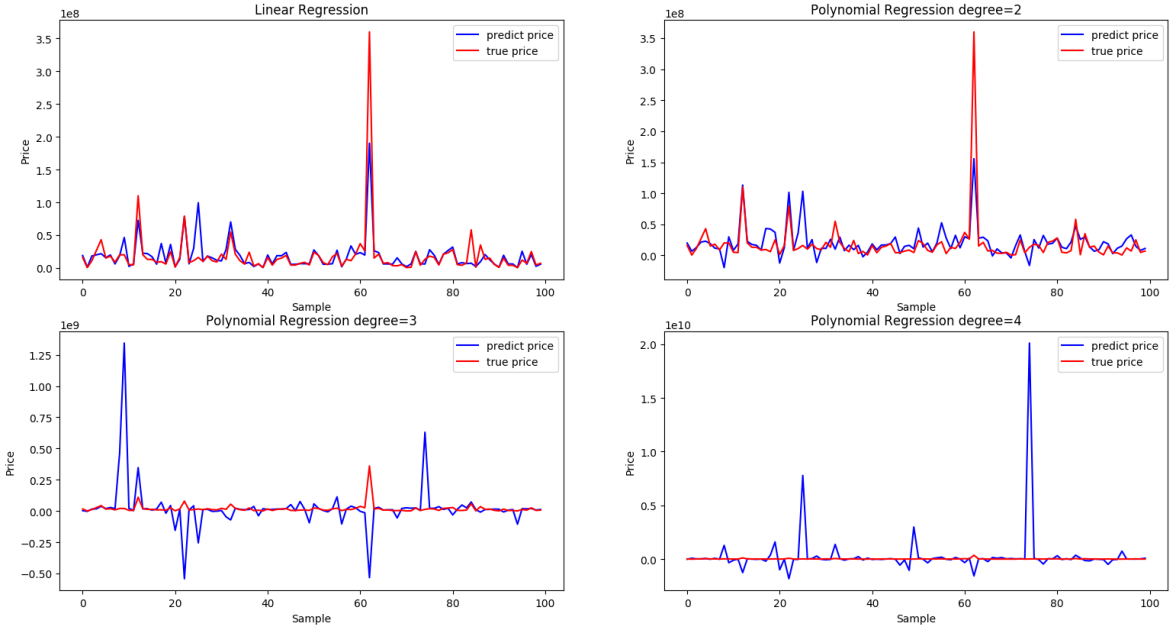

图 4: PCA None

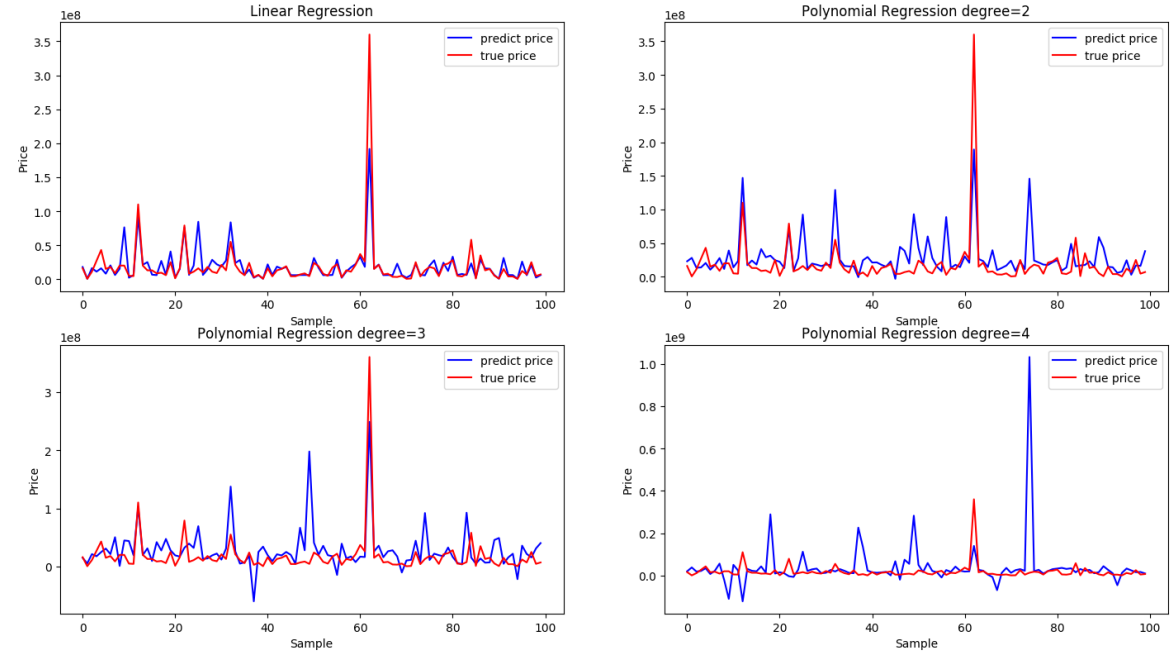

图 5: PCA 20

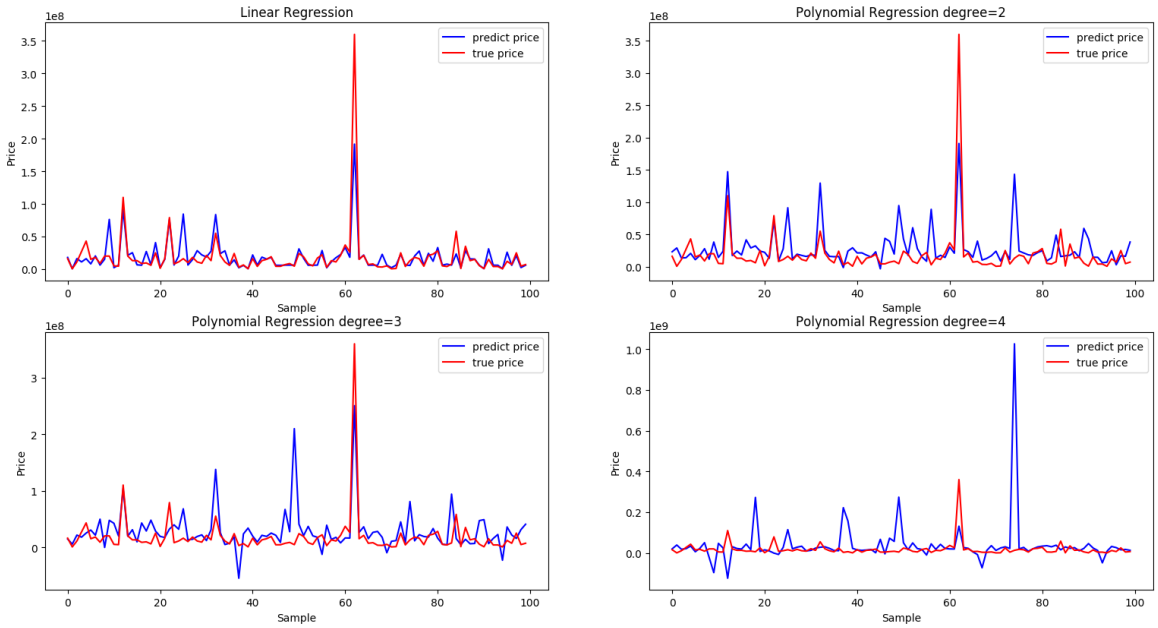

图 6: PCA 10

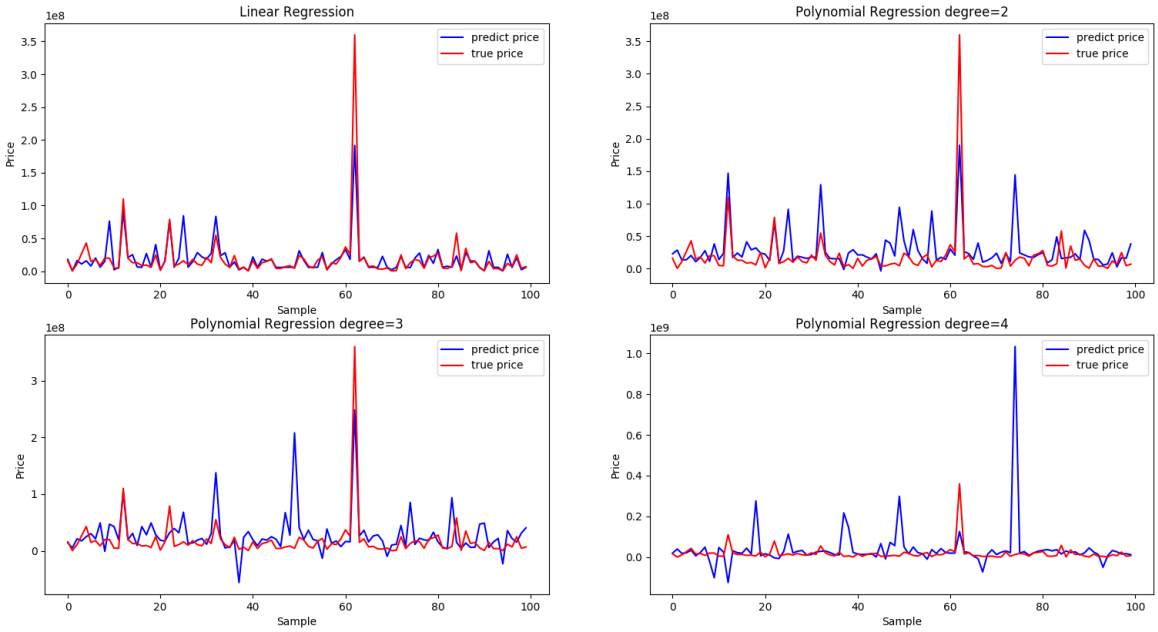

图 7: PCA 5

4.3 结论

本报告提出一种由分类预测和精准预测组成的双训练双预测模型，在分类预测中，探索 PCA 和 LDA 分别运用于 Random Forest、Logistic Regression with L1/L2 Penalty、SVC 模型，在不考虑训练时间因素的情况下，通过交叉验证，得到最优参数"n_estimators"、"max_features" 以及"max_depth", 选出最优模型，即 Random Forest with no PCA. 以青森🄴（Aomori）为例, 预测正确率达到 90% 以上。至此，意味着假如用户希望得到一个房屋价格大致的区间，该模型模型分类预测正确率可达 90% 以上.

在初始对训练数据预处理过程中，已经根据不同的价格区间将训练集分为 4 个小训练集，结合分类预测的结果, 再次对 4 个小训练集进行特定的训练。训练算法选用回归模型。分别尝试了数据标准化 +PCA 与线性回归模型，多项式回归模型的结合，选出最优模型，R2 决定系数大于 0.32, 直观的预测结果可见图 7 中左上角的图。

5 附录

5.1 部分实验截图

图 8-图 10: 筛选分类模型过程

图 11-图 13: Random Forest 对应不同的 PCA 策略的训练过程

图 14-图 16: Random Forest 对应不同的 PCA 策略的测试过程

图 17: 最终模型实验结果。包括 Random Forest 和四种回归模型预测结果

```
LDA : the predict Score is 0.6791763533709732
LDA : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
Logistic Regression_L2 : the predict Score is 0.6771836599136499
Logistic Regression_L2 : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
Logistic Regression_L1 : the predict Score is 0.5426768515443374
Logistic Regression_L1 : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' '0 ~ A' 'C ~ +']
selected_model_index is15
the best maxfeature for the best model is 5
the best treenumber for the best model is 800
Random Forest : the predict Score is 0.7150448356027898
Random Forest : the predict Class is ['B ~ C' 'C ~ +' 'C ~ +' ... 'C ~ +' 'B ~ C' 'C ~ +']
the parameters for the best model are {'C': 10, 'gamma': 0.1}
the cross validation score for the best model is 0.6865480427046263
SVC : the predict Score is 0.6881434739289273
SVC : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'B ~ C' 'C ~ +']
```

图 8: 基于 LDA 的分类预测模型

```
LDA : the predict Score is 0.6761873131849884
LDA : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
Logistic Regression_L2 : the predict Score is 0.6612421122550648
Logistic Regression_L2 : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
Logistic Regression_L1 : the predict Score is 0.5383593490534706
Logistic Regression_L1 : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
selected_model_index is66
the best maxfeature for the best model is 7
the best treenumber for the best model is 550
Random Forest : the predict Score is 0.6874792427764862
Random Forest : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
the parameters for the best model are {'C': 100, 'gamma': 0.1}
the cross validation score for the best model is 0.6872597864768684
```

图 9: 基于 PCA 的分类预测模型

```
the parameters for the best model are {'C': 1, 'gamma': 0.0001}
the cross validation score for the best model is 0.6511032028469751
SVC : the predict Score is 0.6692128860843574
SVC : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
```

图 10: 基于 PCA 的 SVC 预测模型

```
 1 /usr/local/bin/python3.7 /Users/muhaoguo/Documents/study/神州数码/日本房价/--Japanese-house-price-prediction/
   main.py
 2 -------------TRAIN------------------
 3 Random Forest algorithm is training ...
 4 the parameters for the best model are {'max_features': 9, 'n_estimators': 1450}
 5 the training score for the best model is 0.9995729537366548
 6
 7 Random Forest algorithm training has finished
 8 NOW, RUN THE TRAIN.py AGAIN
 9 time cost 30529.562619924545 s
10
11 Process finished with exit code 0
12
```

图 11: RF no PCA 训练

```
/usr/local/bin/python3.7 /Users/muhaoguo/Documents/study/神州数码/日本房价/--Japanese-house-price-prediction/
main.py
PCA Processing is running ...
PCA Processing has Done !

we need train the model first
-------------TRAIN------------------
Random Forest algorithm is training ...
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages/sklearn/model_selection/
_split.py:2053: FutureWarning: You should specify a value for 'cv' instead of relying on the default value
. The default value will change from 3 to 5 in version 0.22.
  warnings.warn(CV_WARNING, FutureWarning)
the parameters for the best model are {'max_features': 5, 'n_estimators': 1200}
the training score for the best model is 0.9994306049822064
Random Forest : the predict Score is 0.704417137163733
Random Forest : the predict Class is ['C ~ +' 'C ~ +' 'C ~ +' ... 'C ~ +' 'C ~ +' 'C ~ +']
Random Forest algorithm training has finished
NOW, RUN THE main.py AGAIN
time cost 7755.67250418663 s
```

图 12: RF 10 PCA 训练

```
/usr/local/bin/python3.7 /Users/muhaoguo/Documents/study/神州数码/日本房价/--Japanese-house-price-prediction/
main.py
PCA Processing is running ...
PCA Processing has Done !

-------------TRAIN------------------
Random Forest algorithm is training ...
the parameters for the best model are {'max_features': 9, 'n_estimators': 700}
the training score for the best model is 0.9994306049822064

Random Forest algorithm training has finished
NOW, RUN THE TRAIN.py AGAIN
time cost 37929.29609298706 s

Process finished with exit code 0

```

图 13: RF 20 PCA 训练

```
/usr/local/bin/python3.7 /Users/muhaoguo/Documents/study/神州数码/日本房价/--Japanese-house-price-prediction/
main.py
There is a .sav file, that's the trained model, we don't need train again.
------------Class Predict-------------------
Random Forest algorithm is running ...
Random Forest algorithm TEST score is 0.9066755230820326
Random Forest algorithm has finished.

result is loading to a file ...
result has writen into a file.

time cost 2.3979787826538086 s
------------Precise Predict-------------------
Precise predict algorithm is running ...
['C ~ +' '0 ~ A' 'C ~ +' ... 'B ~ C' 'C ~ +' 'C ~ +']
linear regression score is 0.3293494683122302
Precise predict algorithm has finished.

time cost 0.11872696876525879 s

Process finished with exit code 0

```

图 14: RF no PCA 测试

```
/usr/local/bin/python3.7 /Users/muhaoguo/Documents/study/神州数码/日本房价/--Japanese-house-price-prediction/main.py
There is a .sav file, that's the trained model, we don't need train again.
------------Class Predict-------------------
Random Forest algorithm is running ...
Random Forest algorithm TEST score is 0.7064098306210561
Random Forest algorithm has finished.

result is loading to a file ...
result has writen into a file.

time cost 0.9149298667907715 s
------------Precise Predict-------------------
Precise predict algorithm is running ...
['C ~ +' 'B ~ C' 'B ~ C' ... 'C ~ +' 'C ~ +' 'C ~ +']
linear regression score is 0.2635319795279373
Precise predict algorithm has finished.

time cost 0.12325501441955566 s

Process finished with exit code 0

```

图 15: RF 20 PCA 测试

```
/usr/local/bin/python3.7 /Users/muhaoguo/Documents/study/神州数码/日本房价/--Japanese-house-price-prediction/main.py
There is a .sav file, that's the trained model, we don't need train again.
------------Class Predict-------------------
Random Forest algorithm is running ...
Random Forest algorithm TEST score is 0.7336433078711392
Random Forest algorithm has finished.

result is loading to a file ...
result has writen into a file.

time cost 0.6878659725189209 s
------------Precise Predict-------------------
Precise predict algorithm is running ...
['C ~ +' 'C ~ +' 'C ~ +' ... 'B ~ C' 'C ~ +' 'A ~ B']
linear regression score is 0.37330464105636674
Precise predict algorithm has finished.

time cost 0.12044405937194824 s

Process finished with exit code 0

```

图 16: RF 10 PCA 测试

```
/usr/local/bin/python3.7 /Users/muhaoguo/Documents/study/神州数码/日本房价/--Japanese-house-price-prediction/
main.py
There is a .sav file, that's the trained model, we don't need train again.
------------Class Predict-------------------
Random Forest algorithm is running ...
Random Forest algorithm TEST score is 0.9066755230820326
Random Forest algorithm has finished.

result is loading to a file ...
result has writen into a file.

time cost 2.239809036254883 s
------------Precise Predict-------------------
### Linear_Regression algorithm is running ... ###
linear regression score is 0.3205760389506338
time cost 0.11604690551757812 s
Linear_Regression algorithm has finished.

### Polynomial_Regression_Degree2 algorithm is running ... ###
Polynomial Regression score is 0.11090783640043733
time cost 1.5804858207702637 s
Polynomial_Regression_Degree2 algorithm has finished.

### Polynomial_Regression_Degree3 algorithm is running ... ###
Polynomial Regression score is -0.9091463299746727
time cost 6.0142388343811035 s
Polynomial_Regression_Degree3 algorithm has finished.

### Polynomial_Regression_Degree4 algorithm is running ... ###
Polynomial Regression score is -506.5500570175022
time cost 20.819649934768677 s
Polynomial_Regression_Degree4 algorithm has finished.


Process finished with exit code 0

```

图 17: 最终模型结果

5.2 代码

Github: https://github.com/MuhaoGuo/–Japanese-house-price-prediction

Github-Japanese-house-price-prediction