DATA FORECASTING PROJECT REPORT
HOUSE PRICES-ADVANCED REGRESSION TECHNIQUES

TAO WANG

1. Project Description

For home buyers, they generally do not buy homes with basements or near railroads, and there are other features of a home that can even much more influence the price of a home than the number of bedrooms. This project provides 79 characteristics of a house that are used to predict the price of a house.

2. Project Evaluation

There are 1459 data in the test set, and the output contains ID numbers and predicted house prices. Submissions are evaluated on Root-Mean-Squared-Error (RMSE) between the logarithm of the predicted value and the logarithm of the observed sales price.

3. Dataset Description

3.1. File Description.

- train.csv - the training set
- test.csv - the test set
- data˙description.txt - full description of each column, originally prepared by Dean De Cock but lightly edited to match the column names used here
- sample˙submission.csv - a benchmark submission from a linear regression on year and month of sale, lot square footage, and number of bedrooms

3.2. **Data fields.** There are 79 variables of inconsistent types, some discrete and some continuous, and after data processing, the variables of type object can be seen as follows:

Date: November 7, 2022.

#	Column	Non-Null Count	Dtype
---	------	--------------	-----
0	MSZoning	1460 non-null	object
1	Street	1460 non-null	object
2	Alley	91 non-null	object
3	LotShape	1460 non-null	object
4	LandContour	1460 non-null	object
5	Utilities	1460 non-null	object
6	LotConfig	1460 non-null	object
7	LandSlope	1460 non-null	object
8	Neighborhood	1460 non-null	object
9	Condition1	1460 non-null	object
10	Condition2	1460 non-null	object
11	BldgType	1460 non-null	object
12	HouseStyle	1460 non-null	object
13	RoofStyle	1460 non-null	object
14	RoofMatl	1460 non-null	object
15	Exterior1st	1460 non-null	object
16	Exterior2nd	1460 non-null	object
17	MasVnrType	1452 non-null	object
18	ExterQual	1460 non-null	object
19	ExterCond	1460 non-null	object
20	Foundation	1460 non-null	object
21	BsmtQual	1423 non-null	object
22	BsmtCond	1423 non-null	object
23	BsmtExposure	1422 non-null	object
24	BsmtFinType1	1423 non-null	object
25	BsmtFinType2	1422 non-null	object
26	Heating	1460 non-null	object
27	HeatingQC	1460 non-null	object
28	CentralAir	1460 non-null	object
29	Electrical	1459 non-null	object
30	KitchenQual	1460 non-null	object
31	Functional	1460 non-null	object
32	FireplaceQu	770 non-null	object
33	GarageType	1379 non-null	object
34	GarageFinish	1379 non-null	object
35	GarageQual	1379 non-null	object
36	GarageCond	1379 non-null	object
37	PavedDrive	1460 non-null	object
38	PoolQC	7 non-null	object
39	Fence	281 non-null	object
40	MiscFeature	54 non-null	object
41	SaleType	1460 non-null	object
42	SaleCondition	1460 non-null	object

dtypes: object(43)

FIGURE 1. the variables of type 'object'

The characteristics of variables of type object are represented by string,such as:

MSZoning: Identifies the general zoning classification of the sale.

A Agriculture
C Commercial
FV Floating Village Residential
I Industrial
RH Residential High Density
RL Residential Low Density
RP Residential Low Density Park
RM Residential Medium Density

FIGURE 2. an example to show the characteristics of type object

The numeric variables are as follows:

#	Column	Non-Null Count	Dtype
---	------	--------------	-----
0	Id	1460 non-null	int64
1	MSSubClass	1460 non-null	int64
2	LotFrontage	1201 non-null	float64
3	LotArea	1460 non-null	int64
4	OverallQual	1460 non-null	int64
5	OverallCond	1460 non-null	int64
6	YearBuilt	1460 non-null	int64
7	YearRemodAdd	1460 non-null	int64
8	MasVnrArea	1452 non-null	float64
9	BsmtFinSF1	1460 non-null	int64
10	BsmtFinSF2	1460 non-null	int64
11	BsmtUnfSF	1460 non-null	int64
12	TotalBsmtSF	1460 non-null	int64
13	1stFlrSF	1460 non-null	int64
14	2ndFlrSF	1460 non-null	int64
15	LowQualFinSF	1460 non-null	int64
16	GrLivArea	1460 non-null	int64
17	BsmtFullBath	1460 non-null	int64
18	BsmtHalfBath	1460 non-null	int64
19	FullBath	1460 non-null	int64
20	HalfBath	1460 non-null	int64
21	BedroomAbvGr	1460 non-null	int64
22	KitchenAbvGr	1460 non-null	int64
23	TotRmsAbvGrd	1460 non-null	int64
24	Fireplaces	1460 non-null	int64
25	GarageYrBlt	1379 non-null	float64
26	GarageCars	1460 non-null	int64
27	GarageArea	1460 non-null	int64
28	WoodDeckSF	1460 non-null	int64
29	OpenPorchSF	1460 non-null	int64
30	EnclosedPorch	1460 non-null	int64
31	3SsnPorch	1460 non-null	int64
32	ScreenPorch	1460 non-null	int64
33	PoolArea	1460 non-null	int64
34	MiscVal	1460 non-null	int64
35	MoSold	1460 non-null	int64
36	YrSold	1460 non-null	int64
37	SalePrice	1460 non-null	int64

dtypes: float64(3), int64(35)

FIGURE 3. the variables of type number

4. Data pre-processing

Both the training set data and the test set data have data loss, so we need to perform data cleaning before training. First we count the missing data in the training and test sets, and the statistics are as follows:

- Number of missing data in the training set:

type	num	percent
PoolQC	1453	0.9952054794520548
MiscFeature	1406	0.963013698630137
Alley	1369	0.9376712328767123
Fence	1179	0.8075342465753425
FireplaceQu	690	0.4726027397260274
LotFrontage	259	0.1773972602739726
GarageType	81	0.05547945205479452
GarageYrBlt	81	0.05547945205479452
GarageFinish	81	0.05547945205479452
GarageQual	81	0.05547945205479452
GarageCond	81	0.05547945205479452
BsmtExposure	38	0.026027397260273973
BsmtFinType2	38	0.026027397260273973
BsmtQual	37	0.025342465753424658
BsmtCond	37	0.025342465753424658
BsmtFinType1	37	0.025342465753424658
MasVnrType	8	0.005479452054794521
MasVnrArea	8	0.005479452054794521
Electrical	1	0.0006849315068493151
Id	0	0.0
MSSubClass	0	0.0
MSZoning	0	0.0
LotArea	0	0.0
Street	0	0.0

Figure 4. the variables of type number

- Number of missing data in the test set:

type	num	percent
PoolQC	1456	0.9972602739726028
MiscFeature	1408	0.9643835616438357
Alley	1352	0.9260273972602739
Fence	1169	0.8006849315068493
FireplaceQu	730	0.5
LotFrontage	227	0.15547945205479452
GarageYrBlt	78	0.05342465753424658
GarageFinish	78	0.05342465753424658
GarageQual	78	0.05342465753424658
GarageCond	78	0.05342465753424658
GarageType	76	0.052054794520547946
BsmtCond	45	0.030821917808219176
BsmtQual	44	0.030136986301369864
BsmtExposure	44	0.030136986301369864
BsmtFinType1	42	0.028767123287671233
BsmtFinType2	42	0.028767123287671233
MasVnrType	16	0.010958904109589041
MasVnrArea	15	0.010273972602739725
MSZoning	4	0.0027397260273972603
Utilities	2	0.0013698630136986301
BsmtFullBath	2	0.0013698630136986301
BsmtHalfBath	2	0.0013698630136986301
Functional	2	0.0013698630136986301
Exterior1st	1	0.0006849315068493151
Exterior2nd	1	0.0006849315068493151
BsmtFinSF1	1	0.0006849315068493151
BsmtFinSF2	1	0.0006849315068493151
BsmtUnfSF	1	0.0006849315068493151
TotalBsmtSF	1	0.0006849315068493151
KitchenQual	1	0.0006849315068493151
GarageCars	1	0.0006849315068493151
GarageArea	1	0.0006849315068493151
SaleType	1	0.0006849315068493151
Id	0	0.0

Figure 5. the variables of type number

- Data cleaning of training set:

  Step1: We choose to remove the feature PoolQC, MiscFeature, Alley, Fence and FireplaceQu because we do not have a suitable method to replenish a large amount of data.

  Step2: Since the properties of numeric types are conveniently complemented by medians or averages, etc., we first examine the properties of numeric types. After studying it, we can find that only LotFrontage, MasVnrArea and GarageYrBlt have missing features for the number types in the training set. For the masvnrarea with only 8 missing numbers, we can use the average to fill in, while for the other two with more missing numbers, we choose to use the plural to fill in.

  Step3: For some positively biased data, we try to use log transformation to reduce their skewness. By using function displot, we can find that the feature LotArea have positive skew like figure6.

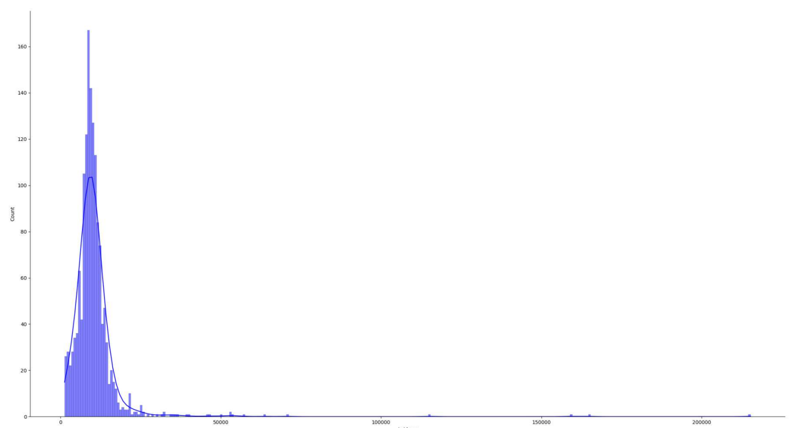

FIGURE 6. the variables of type number

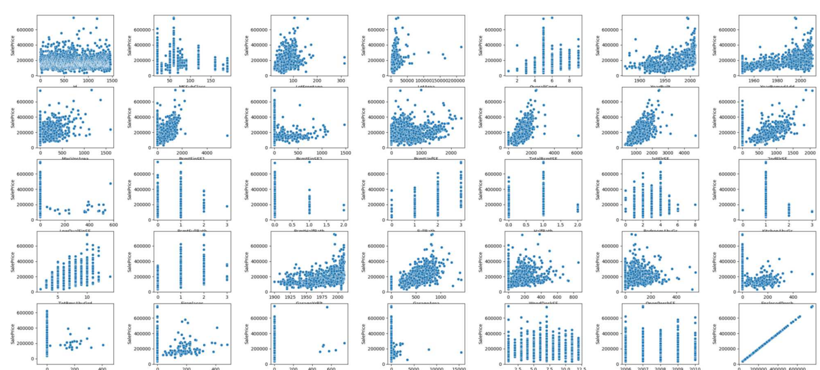

FIGURE 7. the variables of type number

Step4:Some of the data will be unreasonable, may be too large or too small. We need to remove this data before we put it into the training model.

We first use the data visualization to look at it and decide on the removal method in figure7.

From scatter plot we see some columns have outliered data, so drop those rows.

Step5:Now, we analyze the categorical data. For this string type of data, we choose to use the plural to fill in the missing data.

- Data cleaning of testing set All the previous steps are the same, but with the previous steps, we can find that the missing data in the test set is not the same as the training set, so we just need to simultaneously use the same analysis idea for these new missing data.
- Data normalization and removal of weakly correlated data After we clean the training and test set data, we need to consider if all the characteristics are related to the sales price. So we calculate the correlation coefficient of each characteristic and the sales price, we consider the correlation coefficient below 0.3 as almost irrelevant and remove these characteristics

5. Training process

After cleaning the data, I chose to remove the variables with a correlation coefficient of less than 0.3 with the sales price and put the remaining ones into the linear regression model for training. Now that we have finished pre-processing the data, we need to select the right model for training. I have selected a total of six models to submit.

- Linear Regression

  Linear regression models use regression analysis to allow price forecasting by generating a function of saleprice versus variables.After putting the data into the training model, the result given is 0.7313. The submitted data is evaluated based on the root mean square error (RMSE) between the logarithm of the predicted value and the logarithm of the observed sales price. Then 0.7313 is actually a relatively large error, indicating that the linear model is not suitable for house price prediction.
- Random Forest

  Algorithm flow:

  (1) If there are N samples in the over training sample, then the samples obtained from these N samples are put back for sampling N times and the samples are used for tree building

  (2) Let M be the number of features in the input samples, for each node splitting, we first select m (m¡¡M) features from these M features, and then select the best splitting point for splitting among these m features

  (3) Each tree is grown as much as possible without pruning

  The larger the value of m, the higher the correlation in 1 above and the stronger the classification ability in 2, so m is a very important parameter in RF.

  I chose to build 20 decision trees and after putting the training data in, the final training result was 0.16099, which is much better compared to the linear regression model.
- K-neighbor

  The specific method of the K-nearest neighbor model: with the distance and K values defined in advance, for any new sample, it is classified as the one with the most categories among the K samples with the closest distance to that sample.After putting the data into the model, the prediction result is 0.22111. The k-nearest neighbor method is also okay in predicting house prices, but the accuracy is still not quite enough to be accurate to a range.

- Adaboost Regressor

  Algorithm flow:

  (1) First obtain the first weak classifier by learning from N training samples.

  (2) Form a new training sample of N by taking the misclassified sample together with other new data, and obtain the second weak classifier by learning from this sample.

  (3) The misclassified samples of 1 and 2 together with other new samples to form a new training sample of N. The third weak classifier is obtained by learning from this sample.

  (4) The final boosted strong classifier. That is, the classification of a data into which category is determined by the classifier weights.

  I chose to use 50 decision trees as parameters. After putting the data into the model, the prediction result is 0.22124.The implementation of this algorithm requires a large training sample set in order to achieve higher detection accuracy, and during each iteration, a weak classifier is trained for each sample in that sample set, each sample having many features. However, our data itself is not large and many features are removed, so the final result is mediocre.

- Xgboost

  Similar to the GBDT routine, XGBoost also requires accumulating the scores of multiple trees to obtain the final prediction score (at each iteration, an additional tree is added to the existing tree to fit the residuals between the prediction results of the previous tree and the true value). xgboost is considered to be an easy-to-use and easily scalable algorithm. After I put the training data in, the training result was rather mediocre and not quite what I thought beforehand, the result was surprisingly 0.60098. I removed the previous operation of removing variables with low correlation coefficients, retrained, and added the number of trees to 400, and then the result became 0.17429. So it was found that in random forest related algorithms, perhaps too many variables should not be removed.

- Gradient Boosting Regressor

  The gradient boosting algorithm, GBDT mentioned above, was the one that gave me the smallest error in the results during these training sessions. Although xgboost is optimized based on the GBDT algorithm, in this experiment, I was able to achieve 0.15922 after putting in the data, and 0.14173 after keeping the variables with low correlation coefficients as well.

6. Conclusion

The difference between the house price prediction experiment and the usual data classification is that the data classification only needs to give the data a specific prediction result and then analyze the correct rate. This time, however, the house price prediction is to give the prediction result to a more detailed point, and then to analyze the error. I think this kind of prediction is more difficult to go up to a better result, and the number of data sets itself is not large. I think the training effect can be improved by increasing the dataset through data expansion.

In conclusion, I learned a lot about the new training model in this experiment. For the xgboost model, more adjustments are needed to increase the number of parameters and the depth of the tree to prevent underfitting.