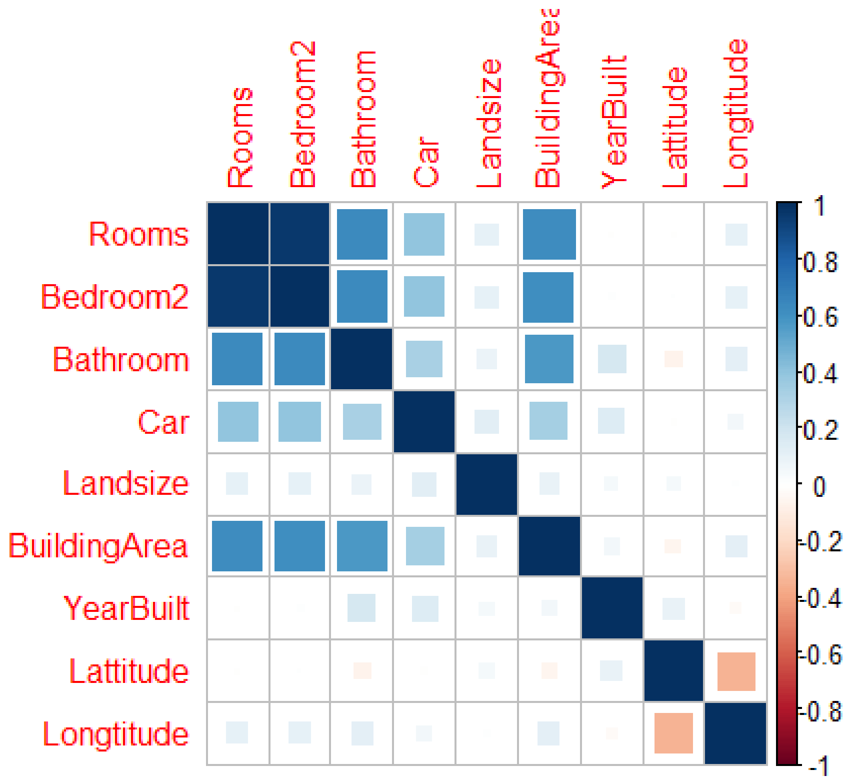
STAT542 HW6

Xiruo Li (xiruoli2)

Q1

Variable Selection:

Correlation Matrix of numerical variable in original data:



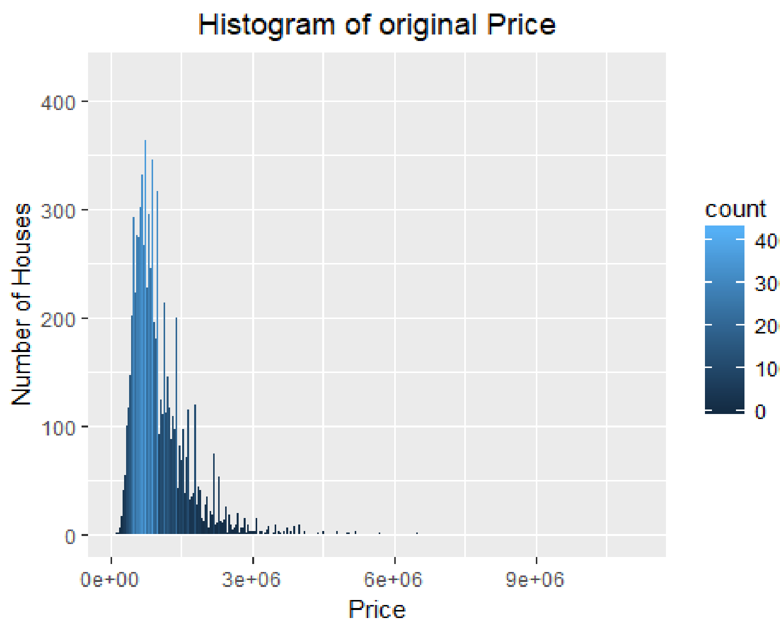
It's clear that **Bedroom2** and **Bathroom** have high correlation with Rooms. So, I drop these two variables. Also, for factor variables, **Suburb, Address, Postcode, CoucilArea** represent location information, which is same as Longitude and Latitude. So, I also drop these four factor variables. Last, I drop **SellerG** due to its useless information.

Finally, I select these 14 variables as my variables for data cleaning.

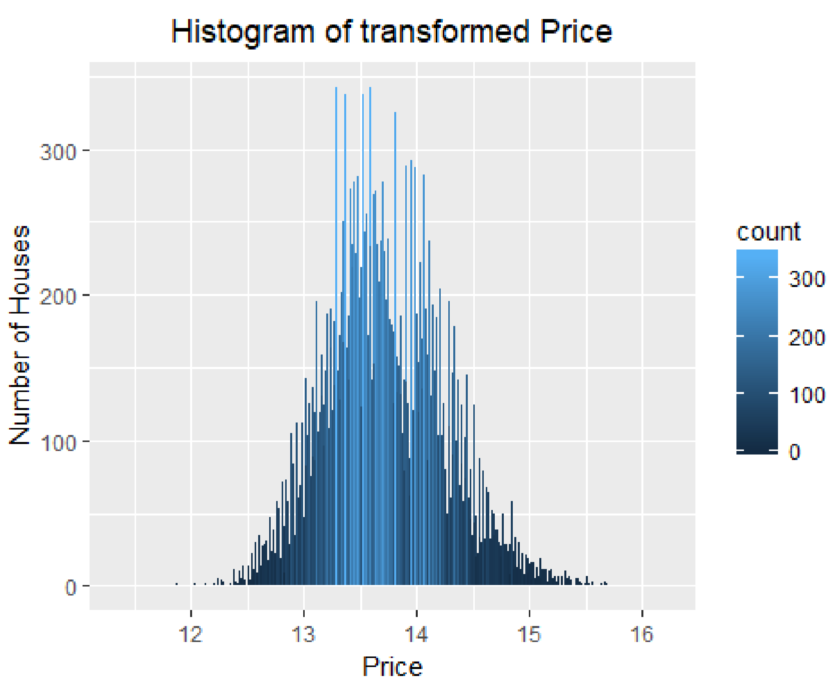
```
#select my variables
my_var=c('Price','Rooms','Car','Landsize','BuildingArea','YearBuilt','L
attitude','Longtitude','Type','Method','Regionname','Date','Distance','
Propertycount')
```

Data Cleaning:

For **Price**, I drop the cases that Price is NA. Then, I found that the histogram of Price is skewed.



Then, I use log transformation for Price. The histogram of Price become normal.

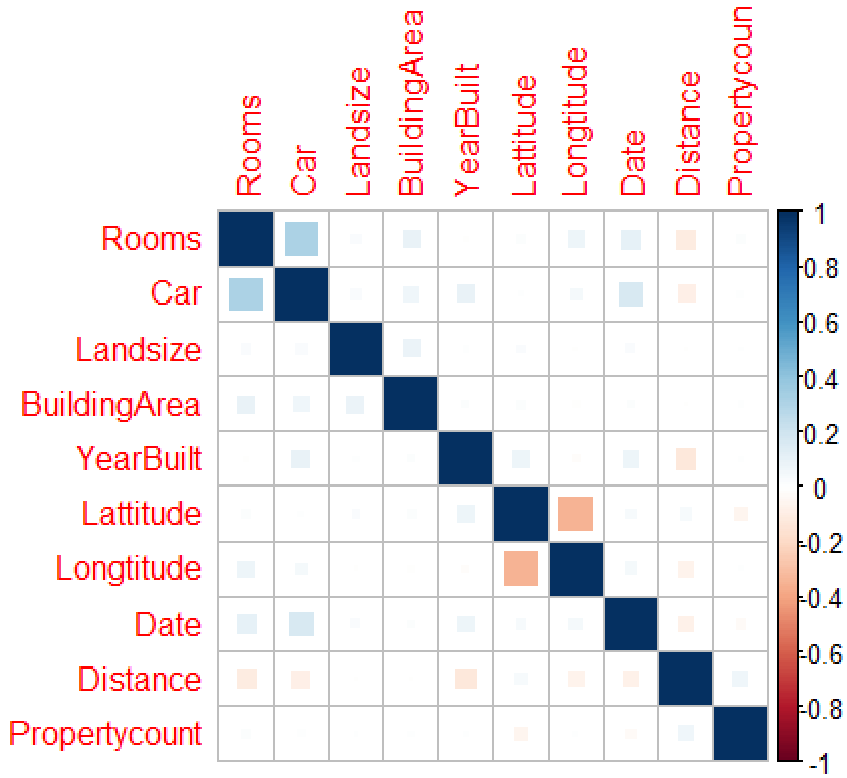


For others numeric variables, I replace NA by mean.

For **Date, Distance, Propertycount**, I convert them into numeric variables. For Date, from old to new, the number become greater. And, I think Distance and Propertycount should be numeric intuitively.

Description of Final Data:

Correlation Matrix of numerical variable in final data:



Summary of final data:

```
##      Price            Rooms             Car            Landsize      
##  Min.   :11.35   Min.   : 1.000   Min.   : 0.000   Min.   :     0.0  
##  1st Qu.:13.36   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.:   351.0  
##  Median :13.68   Median : 3.000   Median : 2.000   Median :   512.0  
##  Mean   :13.72   Mean   : 2.992   Mean   : 1.787   Mean   :   565.8  
##  3rd Qu.:14.07   3rd Qu.: 4.000   3rd Qu.: 2.000   3rd Qu.:   592.0  
##  Max.   :16.23   Max.   :16.000   Max.   :18.000   Max.   :433014.0  
##                                                                      
##   BuildingArea        YearBuilt      Lattitude        Longtitude   
##  Min.   :    0.0   Min.   :1196   Min.   :-38.19   Min.   :144.4  
##  1st Qu.:  133.0   1st Qu.:1970   1st Qu.:-37.84   1st Qu.:145.0  
##  Median :  133.0   Median :1970   Median :-37.80   Median :145.0  
##  Mean   :  142.3   Mean   :1968   Mean   :-37.81   Mean   :145.0  
##  3rd Qu.:  133.0   3rd Qu.:1970   3rd Qu.:-37.77   3rd Qu.:145.0  
##  Max.   :44515.0   Max.   :2019   Max.   :-37.40   Max.   :145.5  
```

```
##
##  Type      Method                          Regionname       Date
##  h:18470   PI: 3255   Southern Metropolitan     :8524   Min.   :16828
##  t: 2866   S :17514   Northern Metropolitan     :7864   1st Qu.:17124
##  u: 5908   SA:  190   Western Metropolitan      :5815   Median :17355
##            SP: 3602   Eastern Metropolitan      :3272   Mean   :17310
##            VB: 2683   South-Eastern Metropolitan:1341   3rd Qu.:17467
##                       Eastern Victoria          : 166   Max.   :17607
##                       (Other)                   : 262
##
##     Distance     Propertycount
##  Min.   :  2.0   Min.   :  2.0
##  1st Qu.: 42.0   1st Qu.: 64.0
##  Median : 90.0   Median :196.0
##  Mean   :107.2   Mean   :176.9
##  3rd Qu.:183.0   3rd Qu.:273.0
##  Max.   :216.0   Max.   :343.0
```

Q2

I use k-prototypes algorithm for clustering. It combines k-means and k-modes algorithm. It's suitable for mix data in this problem (numerical variable+ categorical variables).

I tried several k and I found that, when k=5, Price in each cluster have largest difference. The result is in this table:

	1 N=5687	2 N=6880	3 N=6211	4 N=4310	5 N=4156	p.overall
Rooms	3.27 (0.87)	3.01 (0.83)	3.35 (0.84)	3.04 (0.99)	2.00 (0.67)	0.000
Car	1.87 (0.83)	1.69 (0.94)	1.97 (0.90)	1.86 (0.86)	1.48 (0.61)	<0.001
Landsize	511 (214)	404 (216)	617 (279)	477 (229)	923 (7785)	<0.001
BuildingArea	151 (76.1)	137 (46.8)	148 (102)	143 (108)	129 (692)	<0.001
YearBuilt	1962 (27.1)	1968 (28.6)	1972 (19.1)	1969 (24.9)	1973 (17.3)	<0.001
Lattitude	-37.84 (0.06)	-37.78 (0.06)	-37.79 (0.11)	-37.81 (0.09)	-37.82 (0.06)	0.000
Longtitude	145 (0.04)	145 (0.09)	145 (0.12)	145 (0.10)	145 (0.05)	0.000
Type:						0.000
h	4802 (84.4%)	5363 (78.0%)	5327 (85.8%)	2929 (68.0%)	49 (1.18%)	
t	885 (15.6%)	840 (12.2%)	520 (8.37%)	577 (13.4%)	44 (1.06%)	
u	0 (0.00%)	677 (9.84%)	364 (5.86%)	804 (18.7%)	4063 (97.8%)	
Method:						0.000
PI	591 (10.4%)	786 (11.4%)	606 (9.76%)	694 (16.1%)	578 (13.9%)	
S	4699 (82.6%)	4992 (72.6%)	4941 (79.6%)	0 (0.00%)	2882 (69.3%)	
SA	20 (0.35%)	38 (0.55%)	52 (0.84%)	53 (1.23%)	27 (0.65%)	
SP	0 (0.00%)	505 (7.34%)	166 (2.67%)	2837 (65.8%)	94 (2.26%)	
VB	377 (6.63%)	559 (8.12%)	446 (7.18%)	726 (16.8%)	575 (13.8%)	
Regionname:						0.000
Eastern Metropolitan	0 (0.00%)	0 (0.00%)	3063 (49.3%)	191 (4.43%)	18 (0.43%)	
Eastern Victoria	0 (0.00%)	7 (0.10%)	100 (1.61%)	50 (1.16%)	9 (0.22%)	
Northern Metropolitan	1267 (22.3%)	1593 (23.2%)	2129 (34.3%)	1462 (33.9%)	1413 (34.0%)	
Northern Victoria	0 (0.00%)	13 (0.19%)	95 (1.53%)	45 (1.04%)	13 (0.31%)	
South-Eastern Metropolitan	49 (0.86%)	104 (1.51%)	753 (12.1%)	283 (6.57%)	152 (3.66%)	
Southern Metropolitan	4371 (76.9%)	0 (0.00%)	0 (0.00%)	1664 (38.6%)	2489 (59.9%)	
Western Metropolitan	0 (0.00%)	5160 (75.0%)	0 (0.00%)	593 (13.8%)	62 (1.49%)	
Western Victoria	0 (0.00%)	3 (0.04%)	71 (1.14%)	22 (0.51%)	0 (0.00%)	
Date	17193 (199)	17289 (200)	17411 (150)	17390 (173)	17270 (203)	0.000
Distance	104 (78.1)	121 (72.9)	86.3 (56.5)	112 (69.8)	116 (70.5)	<0.001
Propertycount	175 (115)	190 (97.2)	172 (105)	171 (111)	170 (115)	<0.001
Price	1543514 (810828)	891324 (410458)	993170 (453398)	1111799 (737929)	659581 (333573)	0.000

Conclusion from cluster:

If Room, Car, Landsize, BuildingArea increase, Price increase;

If Latitude, YearBuilt decrease, Price increase;

For Type:h, Method: S, Regionname: Southern Metropolitan, Price is usually high.

Q3

To make a regression, I transfer level of factors into dummy variables firstly.

```
#X and y, transfer each level of factors into dummy variables
num_data = as.matrix(subset(my_data,select=-c(Type,Method,Regionname,Pr
ice)))
factor_data= model.matrix(~ Type+Method+Regionname+0, my_data)
X=cbind(num_data,factor_data)
y=as.vector(my_data[,1])
reg_data=as.data.frame(cbind(y,X))
```

Then I splict data in train(0.7) and test(0.3). For each algorithm, I use 10 fold cross validation to select the best model. For prediction, Price is exponetionalized back to the original scale. I use R square and RMSE as the evaluation method. Also, I plot actual Price vs predict Price.

Penalized linear regression:

Lasso:

Best model:

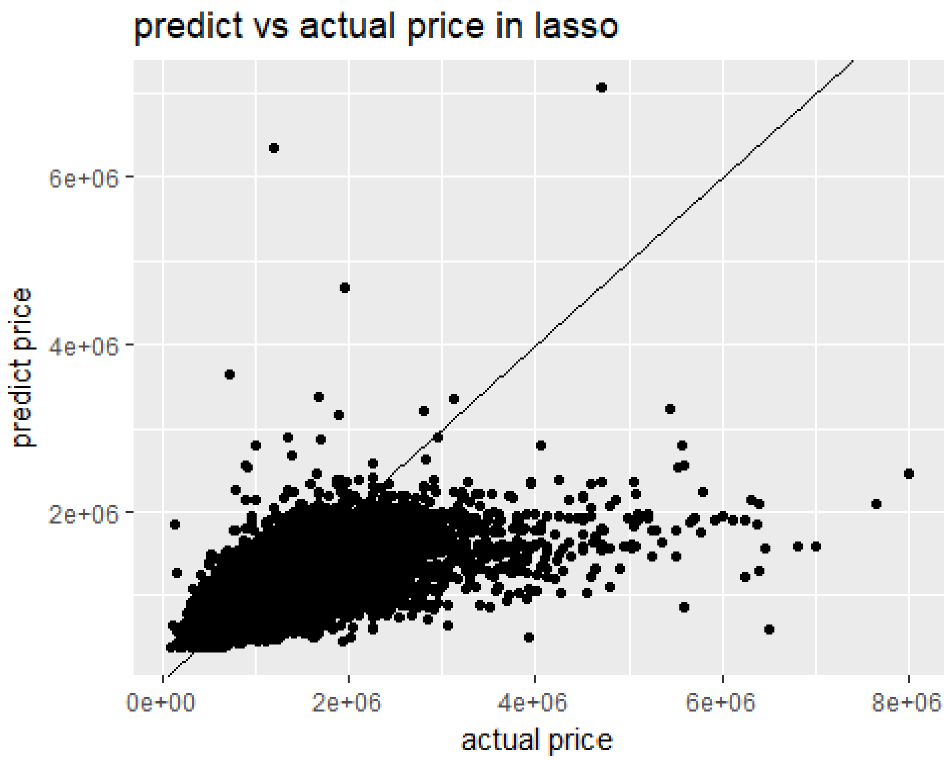
```
## [1] "Best lambda for lasso is:  0.00874469178862405"
```

Coefficients of the best model, which selects 12 variables from original variables:

```
## (Intercept)                             -27.867060409
## Rooms                                     0.186592934
## Car                                       .
## Landsize                                  .
## BuildingArea                              .
## YearBuilt                                -0.002513245
## Lattitude                                -0.114529297
## Longtitude                                0.286184507
## Date                                      .
## Distance                                  0.000443242
## Propertycount                             .
## Typeh                                     0.087980351
## Typet                                     .
## Typeu                                    -0.263537407
## MethodS                                   .
## MethodSA                                  .
## MethodSP                                  .
## MethodVB                                  .
## RegionnameEastern Victoria               -0.049852087
## RegionnameNorthern Metropolitan           .
```

```
## RegionnameNorthern Victoria               -0.056335439
## RegionnameSouth-Eastern Metropolitan    .
## RegionnameSouthern Metropolitan          0.349679136
## RegionnameWestern Metropolitan          -0.022624198
## RegionnameWestern Victoria              -0.156390207
```

Predicted Price vs Actual Price



Evaluation:

```
## [1] "RMSE for lasso is:  482758.326446391"

## [1] "R square for lasso is:  0.488580009267118"
```

Ridge:

Best model:

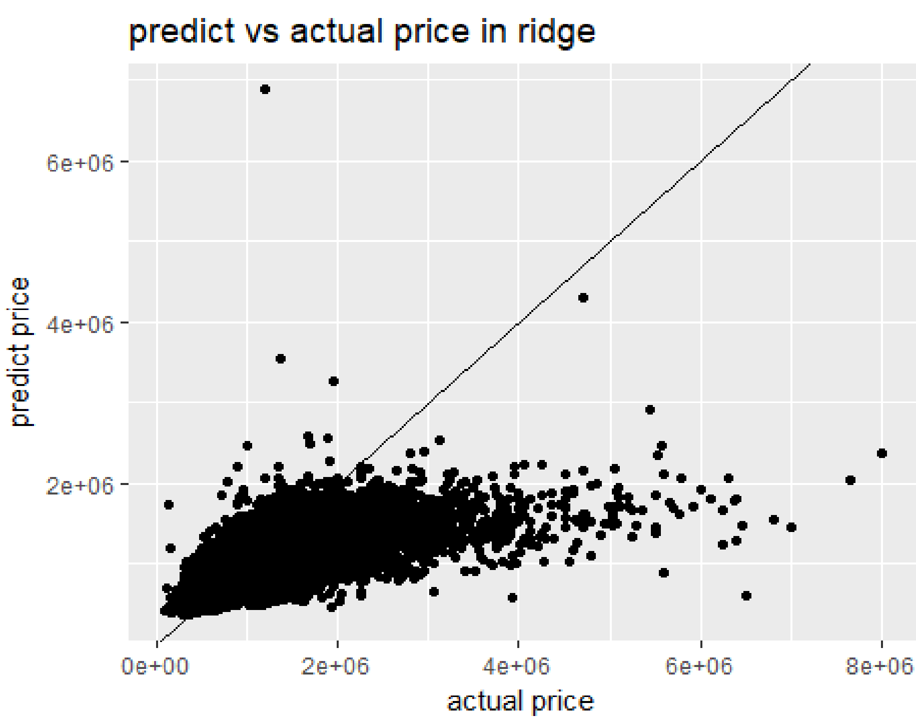
```
## [1] "Best lambda for ridge is:  0.0299983967577159"
```

Coefficients of the best model, which keep all of variables from original variables:

```
## (Intercept)                             -4.603862e+01
## Rooms                                    1.362329e-01
## Car                                      2.551575e-02
## Landsize                                 4.285315e-06
## BuildingArea                             2.934021e-05
```

```
## YearBuilt                              -2.372549e-03
## Lattitude                              -5.264858e-01
## Longtitude                              3.025344e-01
## Date                                    2.528025e-06
## Distance                                6.191546e-04
## Propertycount                           1.054090e-04
## Typeh                                   1.555586e-01
## Typet                                   2.517561e-02
## Typeu                                  -2.147351e-01
## MethodS                                 1.994455e-02
## MethodSA                               -5.169017e-02
## MethodSP                               -4.032529e-02
## MethodVB                                2.855724e-02
## RegionnameEastern Victoria             -3.831485e-01
## RegionnameNorthern Metropolitan        -6.704434e-02
## RegionnameNorthern Victoria            -3.304567e-01
## RegionnameSouth-Eastern Metropolitan -1.885779e-01
## RegionnameSouthern Metropolitan         1.993563e-01
## RegionnameWestern Metropolitan         -1.224020e-01
## RegionnameWestern Victoria             -4.852976e-01
```

Predict Price vs Actual Price:



Evaluation:

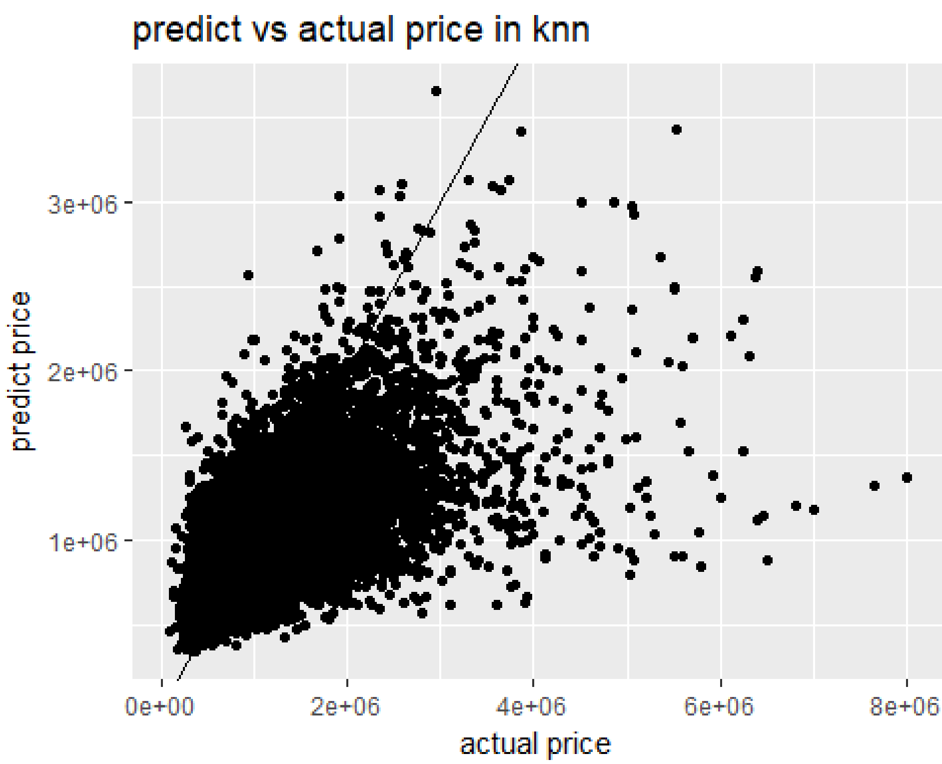
```
## [1] "RMSE for ridge is:  488322.134392713"

## [1] "R square for ridge is:  0.503447805697087"
```

Nonparametric Model:

KNN:

```
##    k   RMSE       Rsquared   MAE
##    5  0.4253217  0.3390658  0.3216509
##   10  0.4231966  0.3353094  0.3233900
##   15  0.4246441  0.3306464  0.3255979
##   20  0.4277984  0.3218259  0.3288412
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 10.
```

Predict Price vs Actual Price:



Evaluation:

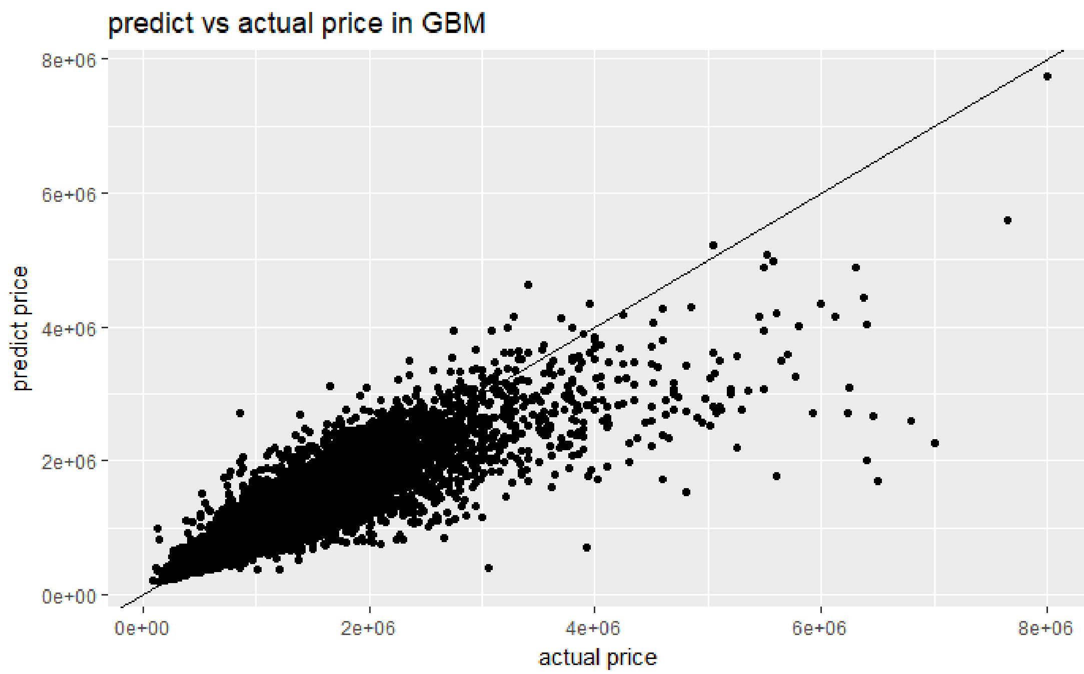
```
## [1] "RMSE for knn is:  508485.634410324"

## [1] "R square for knn is:  0.42005573419629"
```

Gradient Boosting:

n.trees	RMSE	Rsquared	MAE
100	0.2409833	0.7857059	0.1789633
200	0.2244178	0.8132786	0.1658705
300	0.2163357	0.8263308	0.1591855
400	0.2119541	0.8332091	0.1555831
500	0.2087059	0.8381915	0.1528804

```
[1] "The final values used for the GBM model were n.trees = 500,
interaction.depth = 6, shrinkage = 0.1 and n.minobsinnode = 10."
```

Predicted Price vs Actual Price



Evaluation:

```
[1] "RMSE for gbm is:  280775.183493644"

[1] "R square for gbm is:  0.815875042367176"
```

Conclusion

Comparison of each algorithm:

	Penalized Linear Regression		Nonparametric model	
	Lasso	Ridge	KNN	GBM
R^2	0.4885	0.5034	0.4200	0.8158
RMSE	482758	488322	508485	280775

KNN has the worst result, since it's hard to find a suitable distance function for this mixed data (numeric and categorical). Also, the computation cost is high.

Lasso and Ridge have the medium performance. They shrink the coefficient and reduce the variance, so that the effects of irrelevant features can be minimized. Thus, the overfitting will be suppressed. To be specific, Lasso can shrink the coefficient into zeros, so it can select the variable implicitly. In this data, Lasso select 12 variables from original data. For ridge, it can shrink the coefficient nearly to zero and cannot select variables.

Gradient Boosting has the best performance. In each iteration, it fit regression tree to negative gradient (residual) and update the model using gradient descent, so that the accuracy can increase iteratively.

Q4

Since GBM is prone to overfitting, I use Xgboost to overcome this problem. Xgboost used a more regularized model formalization to control over-fitting, which gives it better performance.

I keep the parameters (nrounds, max_depth, eta, min_child_weight) as previous best GBM model. And I use cross validation tune the gamma (penalty term). The final model is:

```
  gamma  RMSE       Rsquared   MAE
  0.1    0.2011754  0.8496227  0.1462586
  1.0    0.2269880  0.8102202  0.1671654
```

```
[1] "The final values used for the Xgboost model were nrounds = 500,
max_depth = 6, eta = 0.1, gamma = 0.1, colsample_bytree = 1,
min_child_weight = 10 and subsample= 1."
```

Predict Price vs Actual Price:

Evaluation:

```
[1] "RMSE for xgboost is:  257737.11518774"

[1] "R square for xgboost is:  0.846168325604234"
```

Compared to GBM, Xgboost increase the accuracy of the model, since it controls the overfitting.