m4

Aniket Tikotkar & Jeroen Hak

13-7-2020

Price is right: Home edition

m1

Description

The basic definition of the problem is to predict the prices of the houses based on the features given. The project revolves around predicting the price of the house depending on the features given. A dataset has been given which has the features of houses and its prices for the year 2014-2015. The dataset has 21 columns, which could all be used as variables. All possible variables will be evaluated in order to decide if they are contributing to the algorithm. For example, the number of square meters/feet will probably have a more direct relation to the price as opposed to the amount of trees in the yard.

Function

The algorithm will predict the price for a house based on one or more predictors like date, zip code, square footage, number of bathrooms. It will be supervised and as a response the prices will be used. The end goal of the algorithm is that someone can put in several features of a house and the algorithm will estimate for how much it probably will be sold. Since the output is quantitative, a regression method will be used to make the prediction.

Necessity

This model is necessary because it can help reduce the time to decide whether to buy a house or not. This system can be used by real estate salespeople, brokers and customers alike. Depending on their needs customers can find a house suitable for them and likewise brokers can refer houses to them. This will improve efficiency. The model developed can be used to cater to other markets as well, and can be modified as is suitable.

Data and candidate algorithms

The dataset that is going to be used for this project is obtained from Kaggle. It contains the house prices of King County in Washington and includes Seattle. The data is publicly available and contains 21 columns and 21613 rows. This dataset was chosen because of the amount of observations and the large amount of variables. The data ranges between May 2014 and May 2015. A training and test set will be produced from the data in order to train and test the algorithm. Since the problem is a regression problem, several regression methods will be considered. These are: Linear Regression, Polynomial Regression, Ridge Regression, Stepwise Regression and Lasso Regression. To find the best algorithm for the problem, these will be tested and their performance will be evaluated. The best one will be chosen thereafter.

m2 Data summary/visualization

Data summary

Summary

Table 1: Table continues below

id	date	price	bedrooms
Min. :1.000e+06	20140623T000000: 142	Min. : 75000	Min. : 0.000
1st Qu.:2.123e+09	20140625T000000: 131	1st Qu.: 321950	1st Qu.: 3.000
Median :3.905e+09	20140626T000000: 131	Median : 450000	Median : 3.000
Mean :4.580e+09	20140708T000000: 127	Mean : 540088	Mean : 3.371
3rd Qu.:7.309e+09	20150427T000000: 126	3rd Qu.: 645000	3rd Qu.: 4.000
Max. :9.900e+09	20150325T000000: 123	Max. :7700000	Max. :33.000
NA	(Other) :20833	NA	NA

Table 2: Table continues below

bathrooms	sqft_living	sqft_lot	floors
Min. :0.000	Min. : 290	Min. : 520	Min. :1.000
1st Qu.:1.750	1st Qu.: 1427	1st Qu.: 5040	1st Qu.:1.000
Median :2.250	Median : 1910	Median : 7618	Median :1.500
Mean :2.115	Mean : 2080	Mean : 15107	Mean :1.494
3rd Qu.:2.500	3rd Qu.: 2550	3rd Qu.: 10688	3rd Qu.:2.000
Max. :8.000	Max. :13540	Max. :1651359	Max. :3.500
NA	NA	NA	NA

Table 3: Table continues below

waterfront	view	condition	grade
Min. :0.000000	Min. :0.0000	Min. :1.000	Min. : 1.000
1st Qu.:0.000000	1st Qu.:0.0000	1st Qu.:3.000	1st Qu.: 7.000
Median :0.000000	Median :0.0000	Median :3.000	Median : 7.000
Mean :0.007542	Mean :0.2343	Mean :3.409	Mean : 7.657
3rd Qu.:0.000000	3rd Qu.:0.0000	3rd Qu.:4.000	3rd Qu.: 8.000
Max. :1.000000	Max. :4.0000	Max. :5.000	Max. :13.000
NA	NA	NA	NA

Table 4: Table continues below

sqft_above	sqft_basement	yr_built	yr_renovated	zipcode
Min. : 290	Min. : 0.0	Min. :1900	Min. : 0.0	Min. :98001
1st Qu.:1190	1st Qu.: 0.0	1st Qu.:1951	1st Qu.: 0.0	1st Qu.:98033
Median :1560	Median : 0.0	Median :1975	Median : 0.0	Median :98065
Mean :1788	Mean : 291.5	Mean :1971	Mean : 84.4	Mean :98078
3rd Qu.:2210	3rd Qu.: 560.0	3rd Qu.:1997	3rd Qu.: 0.0	3rd Qu.:98118
Max. :9410	Max. :4820.0	Max. :2015	Max. :2015.0	Max. :98199

sqft_above	sqft_basement	yr_built	yr_renovated	zipcode
NA	NA	NA	NA	NA

lat	long	sqft_living15	sqft_lot15
Min. :47.16	Min. :-122.5	Min. : 399	Min. : 651
1st Qu.:47.47	1st Qu.:-122.3	1st Qu.:1490	1st Qu.: 5100
Median :47.57	Median :-122.2	Median :1840	Median : 7620
Mean :47.56	Mean :-122.2	Mean :1987	Mean : 12768
3rd Qu.:47.68	3rd Qu.:-122.1	3rd Qu.:2360	3rd Qu.: 10083
Max. :47.78	Max. :-121.3	Max. :6210	Max. :871200
NA	NA	NA	NA

The above data shows the properties of each variable, such as the mean, median, minimum and maximum value, 1st and 3rd Quantile. The price variable has a big range with the minimum value being 75000 and the maximum value being 7700000. In the rooms variable, the maximum number of rooms a house has is 33. The dataset consists of houses built between 1900 and 2015. It can be noted that people opt for a house which has three or four bedrooms. For the condition variable the median is three, from which we can notice that people prefer buying an average house, which may not have a lot of amenities but is decent enough.

Data table example

	1	2	3
id	7129300520	6414100192	5631500400
date	20141013T000000	20141209T000000	20150225T000000
price	221900	538000	180000
bedrooms	3	3	2
bathrooms	1.00	2.25	1.00
sqft_living	1180	2570	770
sqft_lot	5650	7242	10000
floors	1	2	1
waterfront	0	0	0
view	0	0	0
condition	3	3	3
grade	7	7	6
sqft_above	1180	2170	770
sqft_basement	0	400	0
yr_built	1955	1951	1933
yr_renovated	0	1991	0
zipcode	98178	98125	98028
lat	47.5112	47.7210	47.7379
long	-122.257	-122.319	-122.233
sqft_living15	1340	1690	2720
sqft_lot15	5650	7639	8062

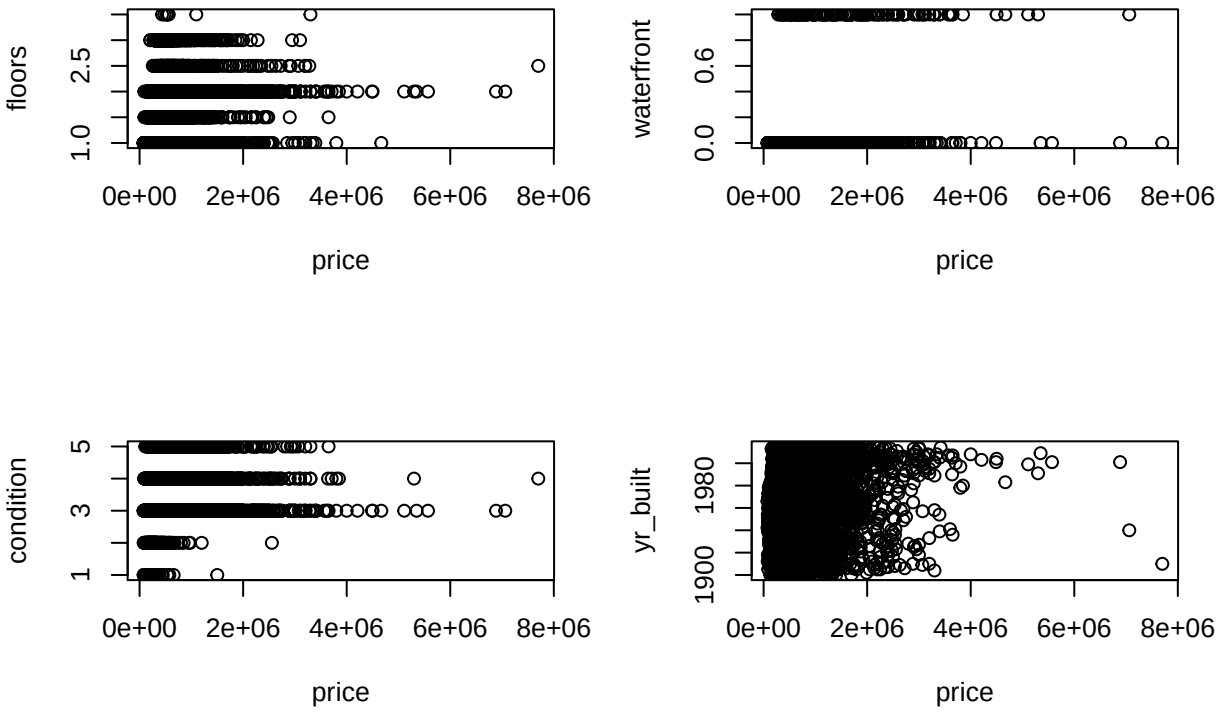
The table above shows a transposed version of the data. The first three rows are shown and the reasons the table was transposed was because of it being more readable with 20 columns. The rows in this example are the columns and give information about a specific property.

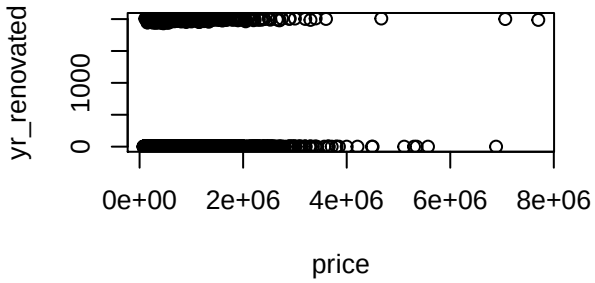
Column_names	Column_names_description
id	House id
date	Date the house was sold
price	Selling price
bedrooms	Number of bedrooms
bathrooms	Number of bathrooms
sqft_living	Square footage living area
sqft_lot	square footage entire lot
floors	Number of floors
waterfront	Waterfront view, 1=Yes, 0=No
view	Number of times the house has been viewed
condition	Indication of condition, 1=worn out, 5=excellent
grade	Overall grade, 1=poor, 13=excellent
sqft_above	square footage minus basement
sqft_basement	square footage of the basement
yr_built	Year the house was built
yr_renovated	Year the house was renovated
zipcode	Zipcode
lat	Latitude
long	Longitude
sqft_living15	Square footage living area in 2015
sqft_lot15	Square footage entire lot in 2015

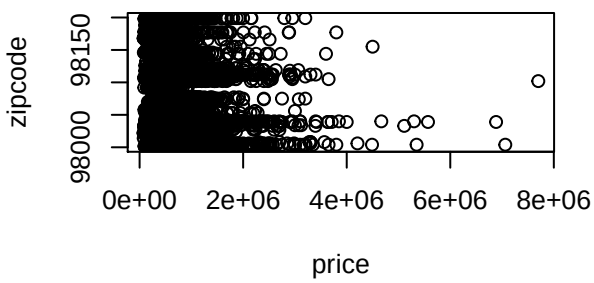
In the table above a description of each column name is given.

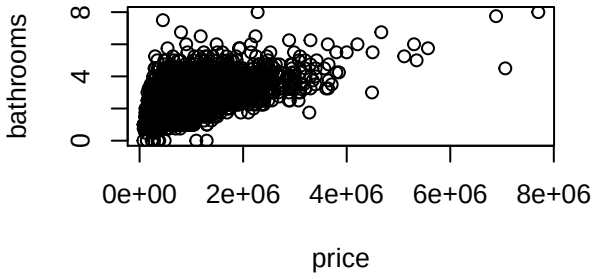
Plots

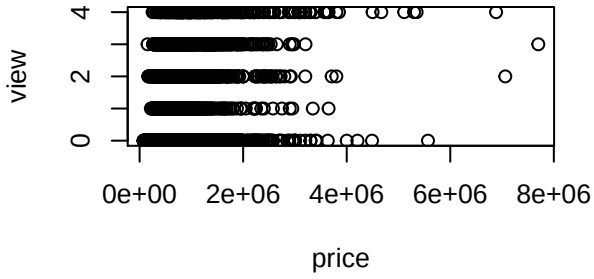
Variable plots

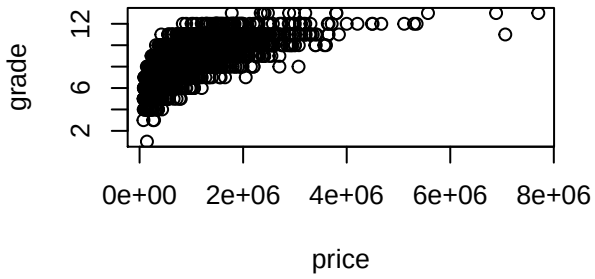


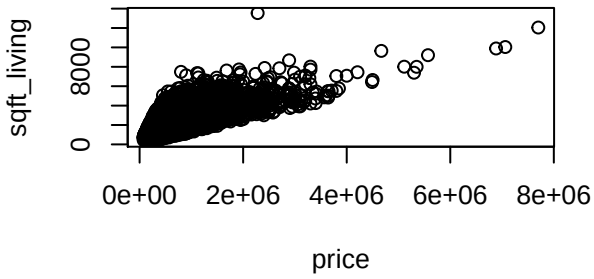

yr_renovated
0
1000
0e+00
2e+06
4e+06
6e+06
8e+06
price

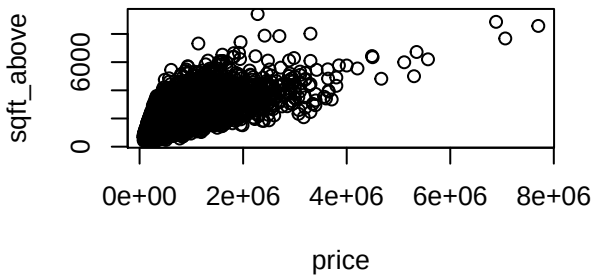

zipcode
98000
98150
0e+00
2e+06
4e+06
6e+06
8e+06
price

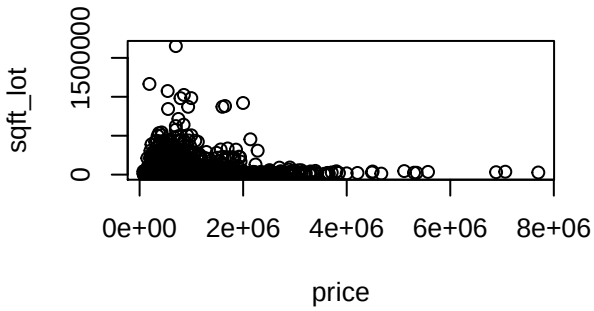

bathrooms
0
4
8
0e+00
2e+06
4e+06
6e+06
8e+06
price

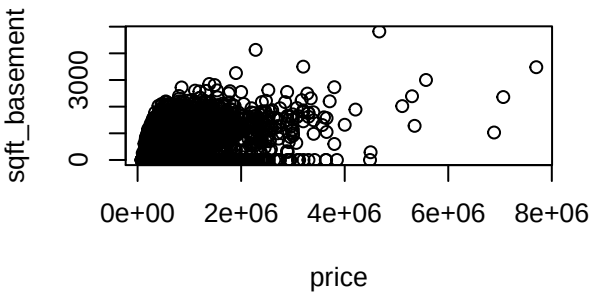

view
0
2
4
0e+00
2e+06
4e+06
6e+06
8e+06
price

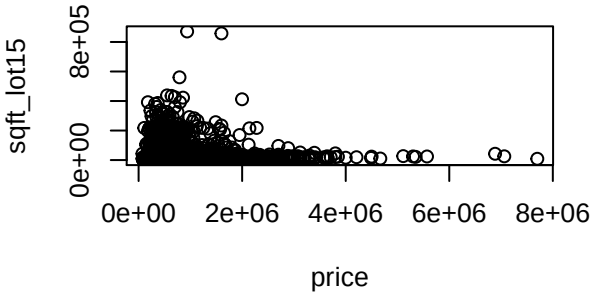

grade
2
6
12
0e+00
2e+06
4e+06
6e+06
8e+06
price

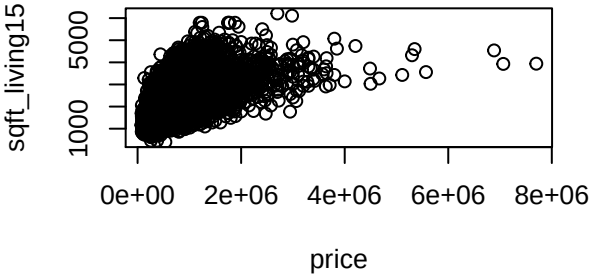

sqft_living
0
8000
0e+00
2e+06
4e+06
6e+06
8e+06
price


sqft_above
0
6000
0e+00
2e+06
4e+06
6e+06
8e+06
price


sqft_lot
0
1500000
0e+00
2e+06
4e+06
6e+06
8e+06
price







In the plots above are all the columns plotted against the selling price. These plots are created to pick out the candidate variables. The candidate variables that show a possible connection are bathrooms, sqft_living, grade, sqft_above, sqft_living15, bedrooms, floors and condition. These candidates will all be used for the model testing. Further analysis on the candidate variables will be performed later to check how well the variable can be used to predict the price.

m3 Algorithm testing

Ridge regression

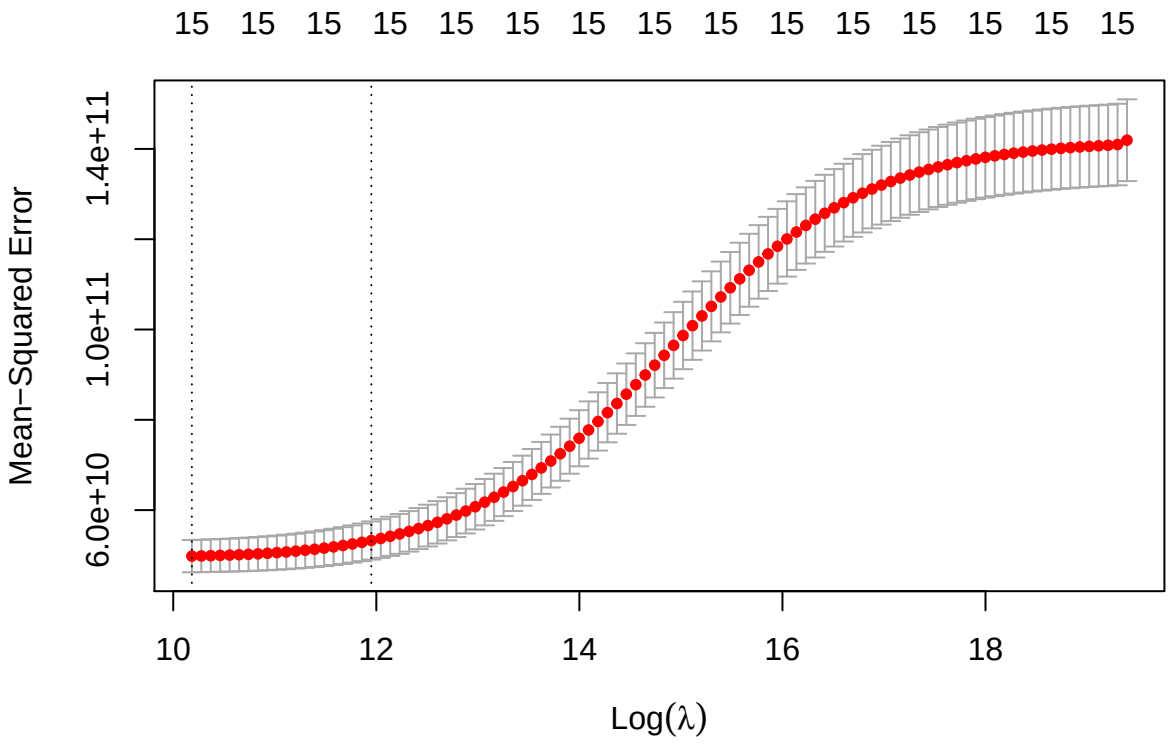
Ridge regression is less sensitive to high variance, than the least squares method because it weighs the variables to make them more stable. This is called L2 regularisation. Ridge regression is also less likely to overfit the data. The parameter lambda is used as a tuning parameter and can weigh variables almost down to zero if this is necessary for stability. Several values for lambda are chosen to figure out which one creates the best model. The best lambda is chosen via cross-validation.

```
## Warning: package 'glmnet' was built under R version 3.6.2
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.0-2
```

```
## The following objects are masked from Houses:
##
##     bathrooms, bedrooms, condition, floors, grade, sqft_above,
##     sqft_basement, sqft_living, sqft_living15, sqft_lot,
##     sqft_lot15, view, waterfront, yr_built, yr_renovated
```




```
## MSE =  45275234661
```

```
##
## Mean test data =  545923.5
```

```
##
## sqrt(MSE) divided by mean =  0.3897612
```

The MSE rises very fast when the log of lambda grows larger than 12. The MSE for the best lambda is 4.528e+10, which is quite large. Since the mean of the testing data is 545923. Taking the square root of the MSE and dividing it by the mean of the test data gives a quick indication that the prediction are ~39% off.

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                            s0
## (Intercept)    5.929302e+06
## bedrooms      -3.752110e+04
## bathrooms      3.956854e+04
## sqft_living    9.121614e+01
## sqft_lot       4.775389e-02
## floors         1.236546e+04
## waterfront     6.351470e+05
## view           4.297237e+04
## condition      2.095410e+04
## grade          9.605509e+04
## sqft_above     8.953433e+01
## sqft_basement  8.104952e+01
## yr_built      -3.371665e+03
## yr_renovated   1.733684e+01
## sqft_living15  4.225997e+01
## sqft_lot15    -5.608722e-01
```

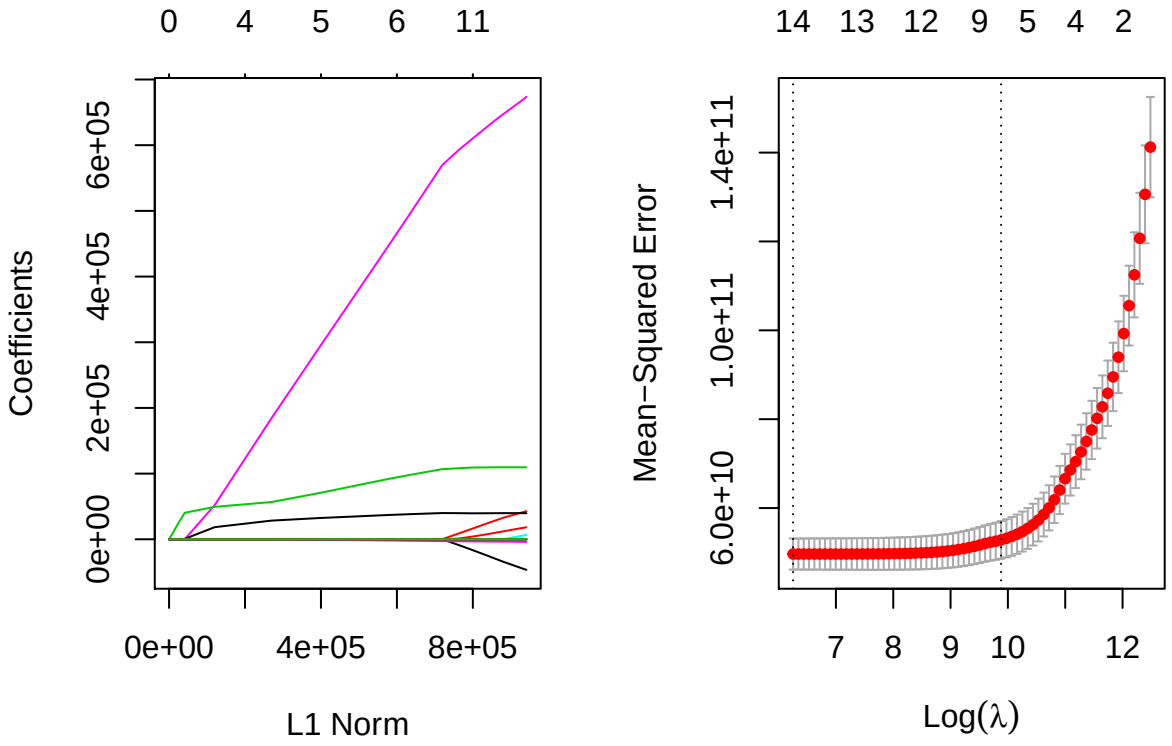
```
## [1] 26456.57
```

The predictors sqft_lot and sqft_lot15 have been (almost) shrunk to zero. The Ridge technique suggest that these predictors should not be used in the model.

Lasso regression

Lasso regression is used when there is a large number of predictor variables. It stands for Least Absolute Shrinkage and Selection Operator (LASSO). The parameter lamba is used as a tuning parameter and if zero, the coefficients are the same as with linear regression. When lambda is large, all the coefficients are close to zero. We need to find the optimal lambda. This value will then be used to train the model.

```
## Warning in regularize.values(x, y, ties, missing(ties)): collapsing to
## unique 'x' values
```




```
## [1] 545923.5
```

```
## [1] 0.3909643
```

```
## [1] 45555184264
```

Most of the coefficients stay zero when the L1 norm is small (left plot). The MSE (right plot) rises very fast when the log of lambda grows larger than 10. The MSE for the best lambda is 45555184264, which is quite large. Since the mean of the testing data is 545923. Taking the square root of the MSE and dividing it by the mean of the test data gives a quick indication that the prediction are ~39% off. This is similar to the outcome of the Ridge regression technique.

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                            s0
## (Intercept)    6.797698e+06
## bedrooms      -4.489982e+04
## bathrooms      4.109114e+04
## sqft_living    1.786835e+02
## sqft_lot       2.975593e-02
```

```
## floors         5.985452e+03
## waterfront     6.699776e+05
## view           3.975868e+04
## condition      1.752077e+04
## grade          1.097041e+05
## sqft_above     1.430672e+01
## sqft_basement  .
## yr_built      -3.840047e+03
## yr_renovated   7.591414e+00
## sqft_living15  2.739160e+01
## sqft_lot15    -5.827642e-01
```

The predictors sqft_basement, sqft_lot and sqft_lot15 have been (almost) shrunk to zero. The Lasso technique suggests that these predictors should not be used in the model.

GAM

Generalized Additive Model is makes use of qualitative as well as quantitative type of response. We???ve made use of GAM for a quantitative type of response, namely price. The predictors have been selected because they have a linear relation to the response, which was found by individually plotting them. The predictors yr_built and grade do not have a large area of confidence as compared to the other predictors. The better the condition of the house the more its price, similar relations can be found if we look at predictors such as sqft_above, sqft_living15, view, grade. The variables yr_built, grade show exceptional show a good fit. The plots related to the GAM can be found in Appendix I.

m4 Core algorithm fine tuning

Table 1

```
## Loading required package: knitr
```

```
## Loading required package: kableExtra
```

Algorithm	MSE	R.squared
Linear Regression	216052428914	0.6295
Polynomial Regression	239025211970	0.7914
Ridge Regression	45330497200	0.6477
Lasso Regression	45555184264	0.6455
GAM	218292338313	

Comparison:

From seeing the above table you can see that out of all the algorithms, ridge regression has the lowest MSE. Polynomial regression has the highest MSE. We have decided to go ahead with ridge regression as our primary algorithm because it has the lowest MSE and its fit is good as well. The models and analysis for linear and polynomial regression can be found in Appendix II and III.

Best Algorithm

The algorithm that performed the best is Ridge regression. As discussed in m3 in the paragraph Ridge regression, the best lambda is choses via cross-validation. The analysis of lambda can also be found in the same paragraph. The value of lambda is used to put weights on variables, which can make the model more stable.

```
##
## Best lambda =  26456.57
```

This value of lambda is used to obtain the best model. This model is less likely to overfit the data as compared to the least squares method.

```
## MSE =  45275234661
```

The predictors sqft_lot and sqft_lot15 will be taken out to try to improve the MSE of the model. In milestone 3 an analysis was performed which shows that both predictors were almost shrunk down to zero.

```
## The following objects are masked from x1 (pos = 5):
##
##     bathrooms, bedrooms, condition, floors, grade, sqft_above,
##     sqft_basement, sqft_living, sqft_living15, sqft_lot,
##     sqft_lot15, view, waterfront, yr_built, yr_renovated
```

```
## The following objects are masked from Houses:
##
##     bathrooms, bedrooms, condition, floors, grade, sqft_above,
##     sqft_basement, sqft_living, sqft_living15, sqft_lot,
##     sqft_lot15, view, waterfront, yr_built, yr_renovated
```

```
## MSE =  45495122746
```

The MSE of the model with sqft_lot and sqft_lot15 is higher than the one with these predictors included. Even though both predictors were almost shrunk down to zero, they still contribute to the model. The final model will therefore include these two predictors.

Appendices

Appendix I

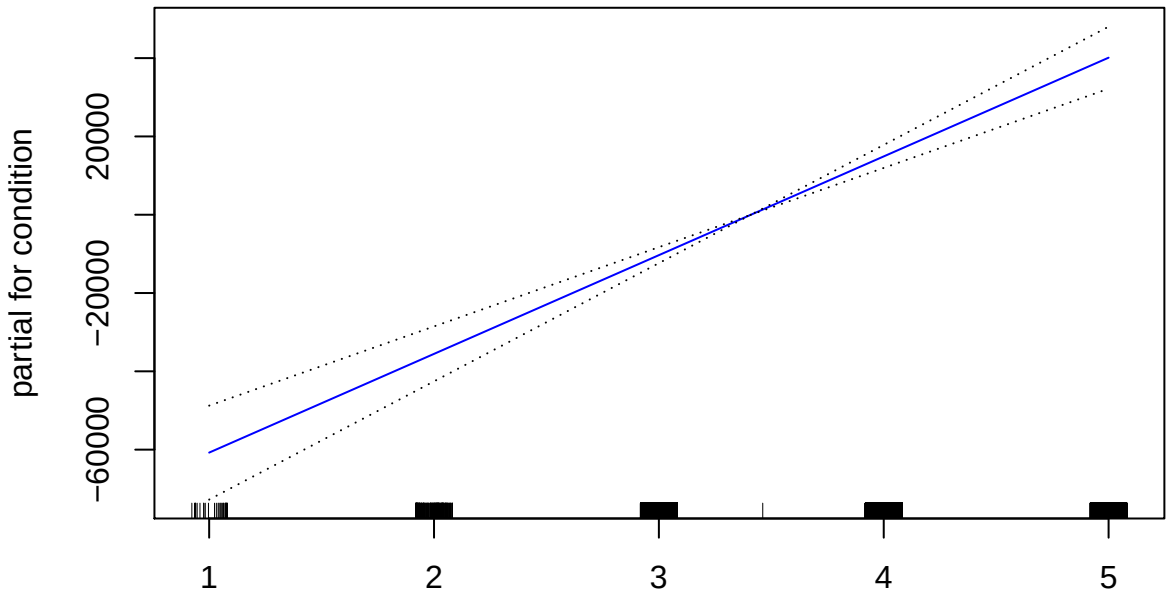
```
## Loading required package: splines
```

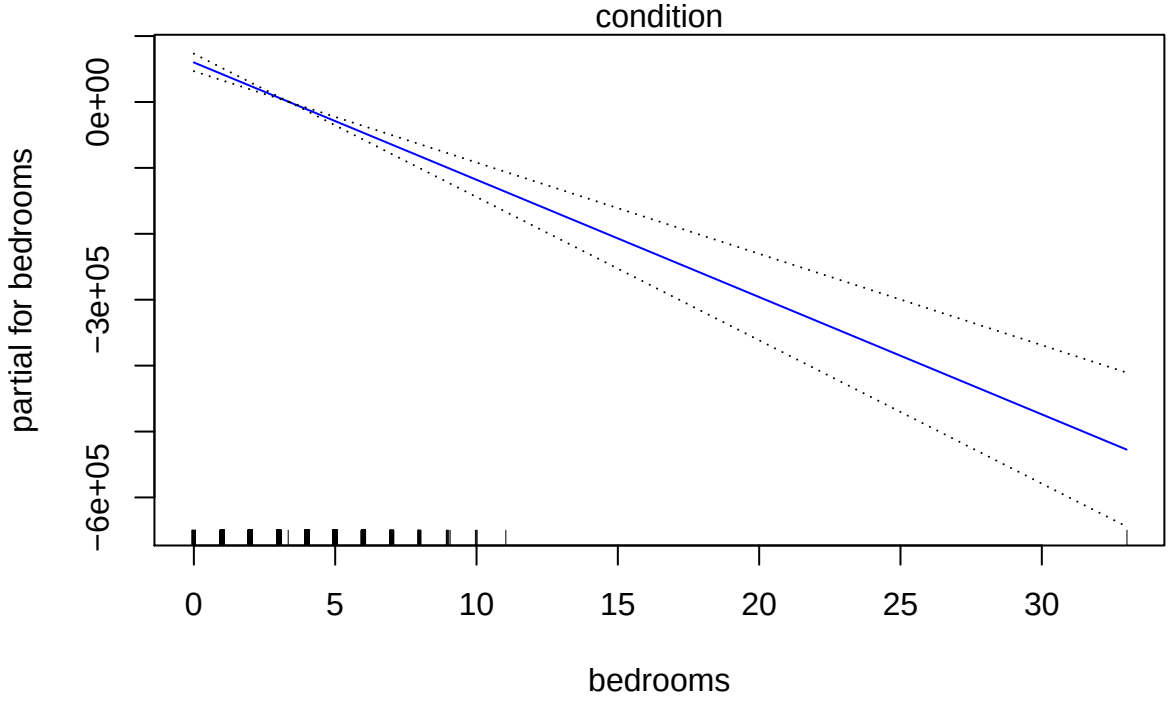
```
## Loading required package: foreach
```

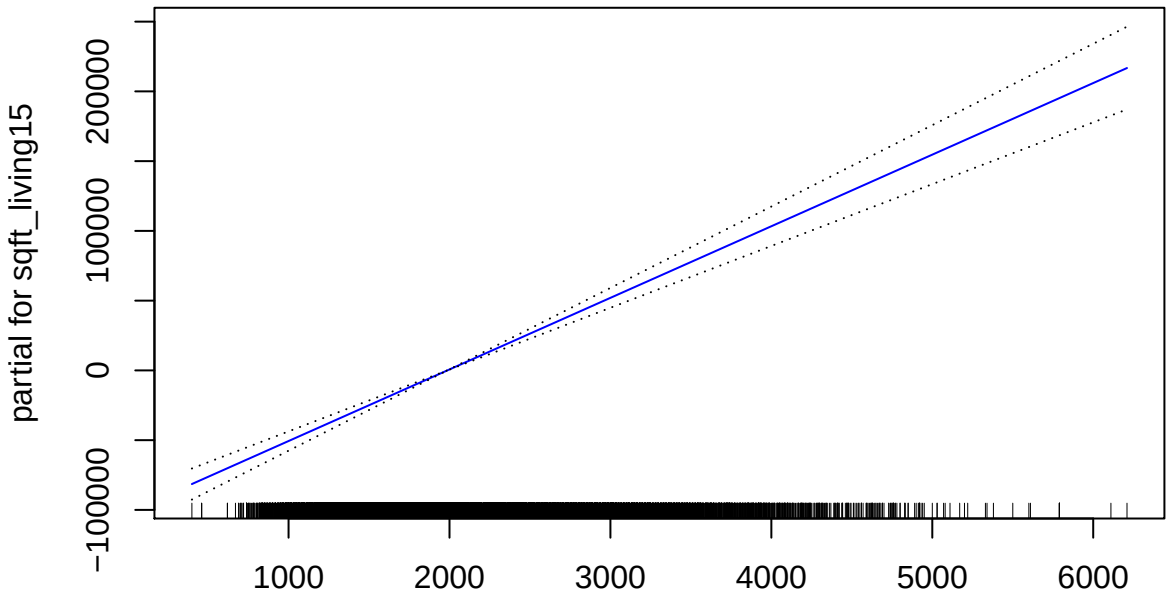
```
## Warning: package 'foreach' was built under R version 3.6.2
```

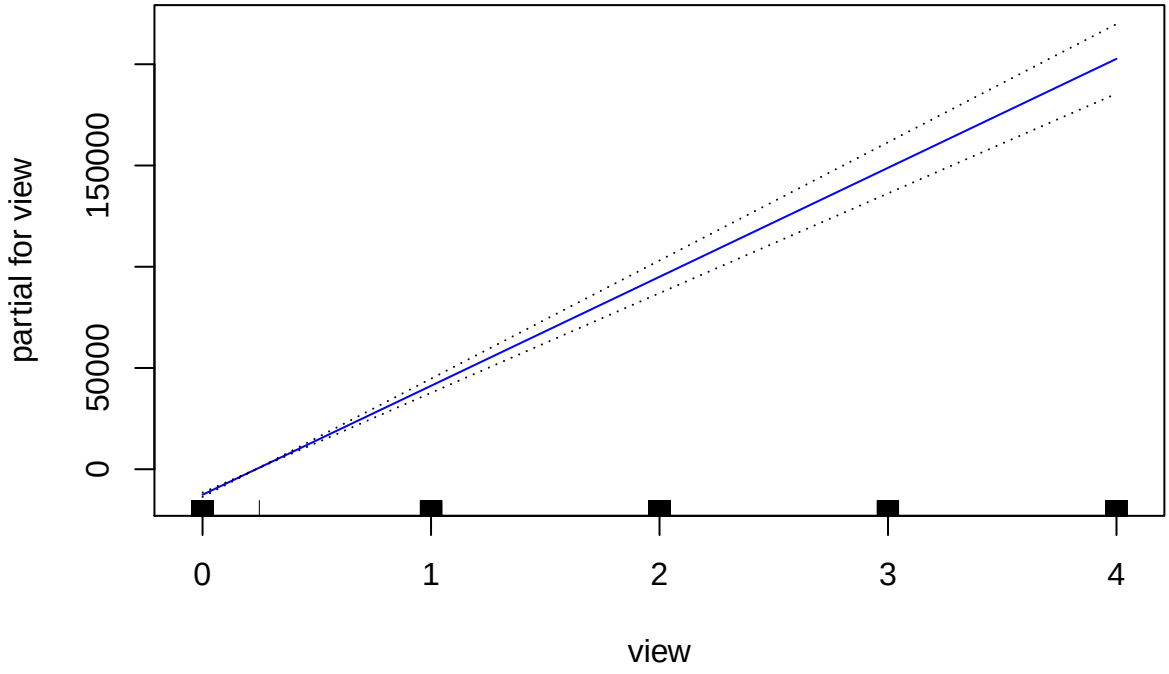
```
## Loaded gam 1.16.1
```

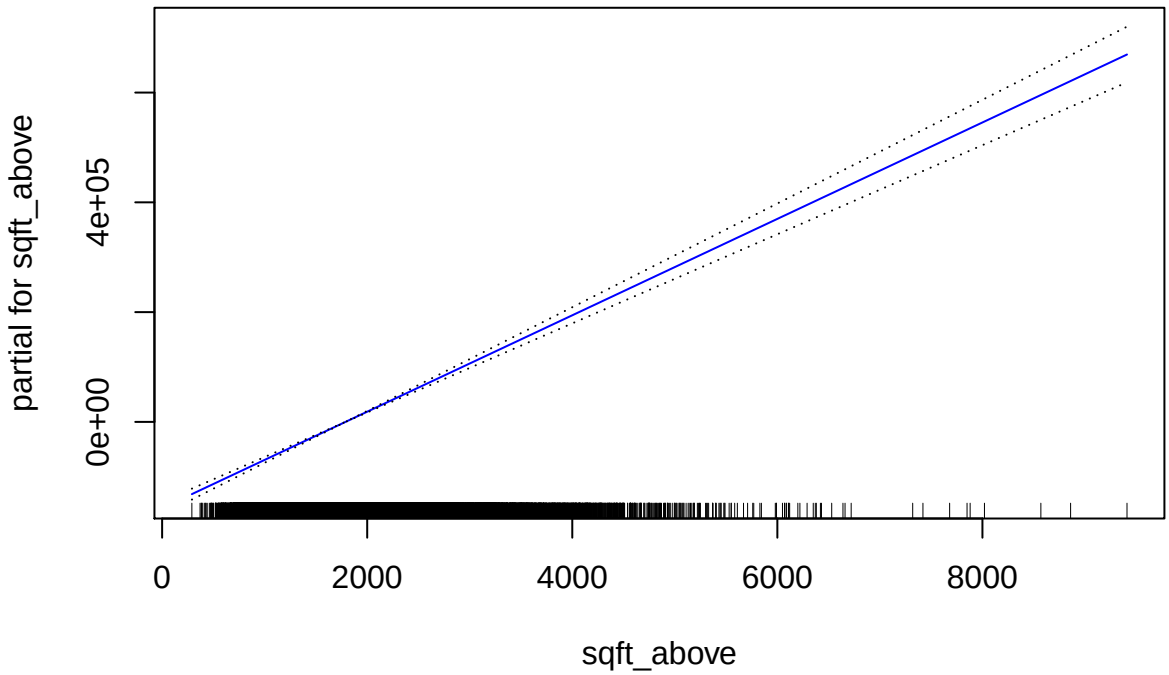
```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

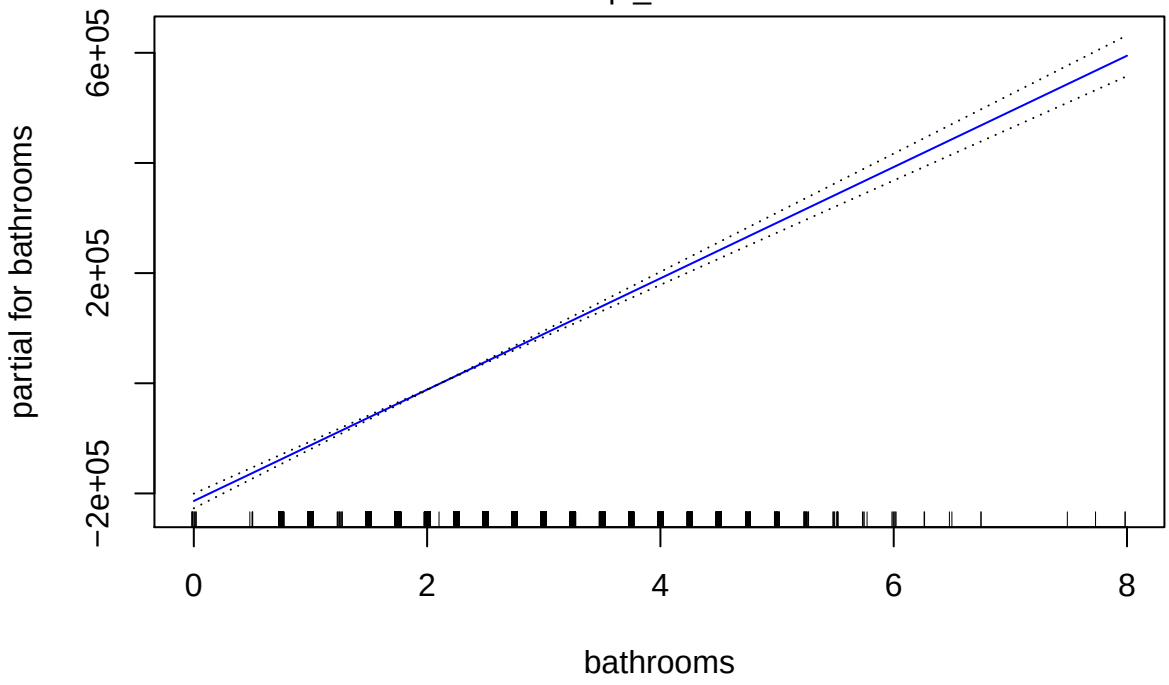

partial for condition
−60000
−20000
20000
1
2
3
4
5
condition

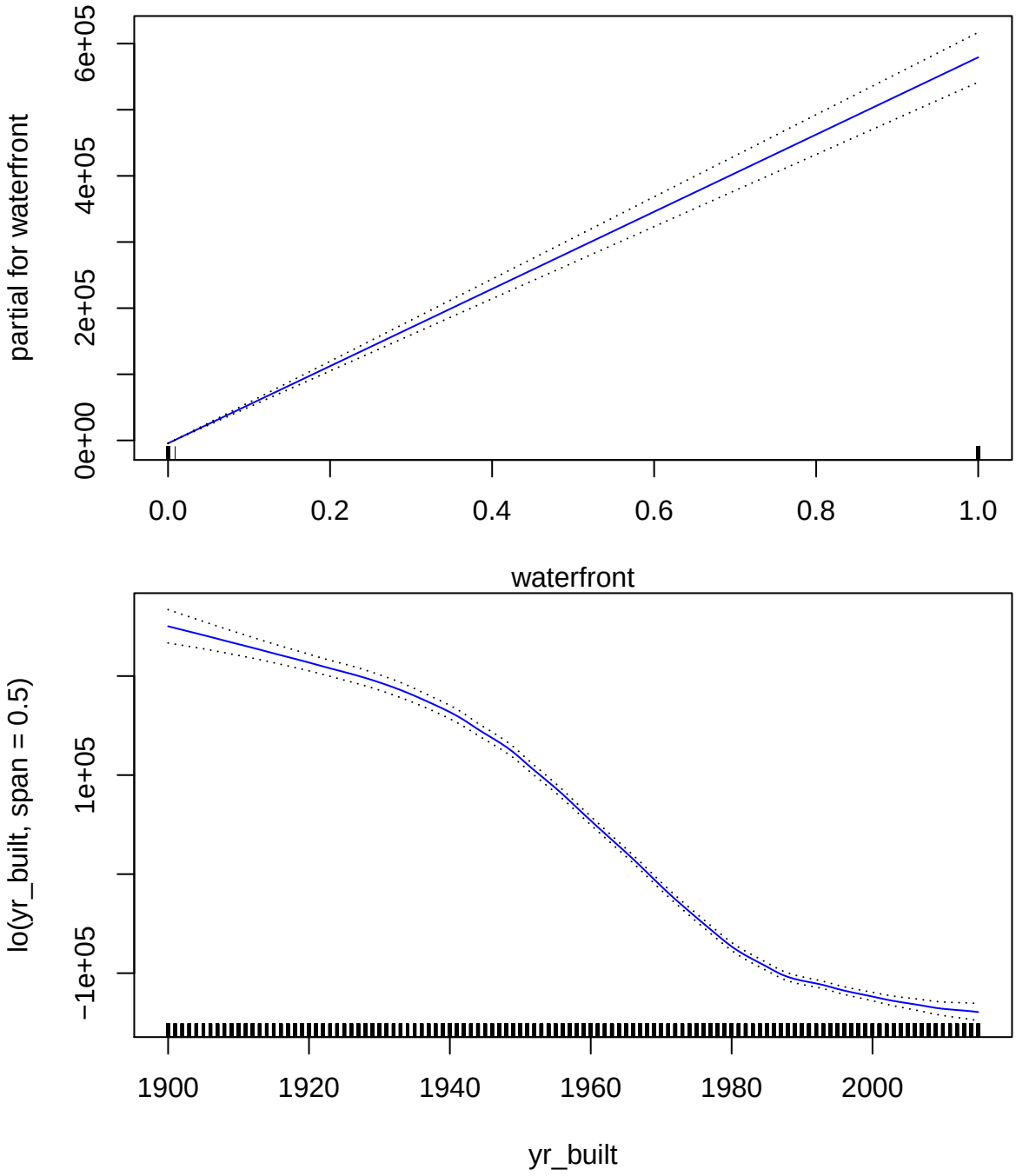

partial for bedrooms
−6e+05
−3e+05
0e+00
0
5
10
15
20
25
30
bedrooms

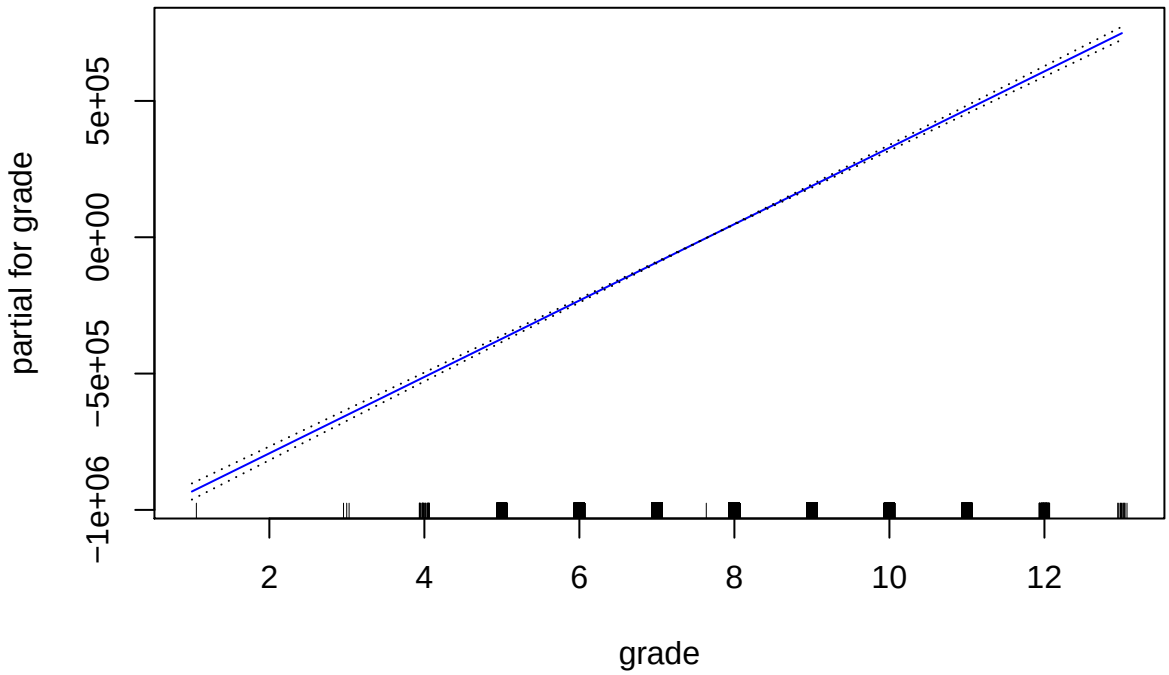

partial for sqft_living15
−100000
0
100000
200000
1000
2000
3000
4000
5000
6000
sqft_living15


partial for view
0
50000
150000
0
1
2
3
4
view


partial for sqft_above
0e+00
4e+05
0
2000
4000
6000
8000
sqft_above


partial for bathrooms
−2e+05
2e+05
6e+05
0
2
4
6
8
bathrooms


partial for waterfront
0e+00
2e+05
4e+05
6e+05
0.0
0.2
0.4
0.6
0.8
1.0
waterfront
lo(yr_built, span = 0.5)
−1e+05
1e+05
1900
1920
1940
1960
1980
2000
yr_built




```
summary(fit1)
```

```
##
## Call: gam(formula = price ~ condition + bedrooms + sqft_living15 +
##     view + sqft_above + bathrooms + waterfront + lo(yr_built,
##     span = 0.5) + grade, data = Houses)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1319912  -113351   -11271    89046  4776223
##
## (Dispersion Parameter for gaussian family taken to be 48504678256)
##
##     Null Deviance: 2.912917e+15 on 21612 degrees of freedom
## Residual Deviance: 1.047736e+15 on 21600.72 degrees of freedom
## AIC: 593135.6
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                              Df     Sum Sq    Mean Sq   F value    Pr(>F)
## condition                     1 4.8440e+12 4.8440e+12    99.867 < 2.2e-16
## bedrooms                      1 2.8524e+14 2.8524e+14  5880.759 < 2.2e-16
## sqft_living15                 1 7.8125e+14 7.8125e+14 16106.638 < 2.2e-16
## view                          1 1.6337e+14 1.6337e+14  3368.129 < 2.2e-16
## sqft_above                    1 2.1243e+14 2.1243e+14  4379.646 < 2.2e-16
## bathrooms                     1 4.0623e+13 4.0623e+13   837.510 < 2.2e-16
## waterfront                    1 4.2816e+13 4.2816e+13   882.712 < 2.2e-16
## lo(yr_built, span = 0.5)      1 1.5127e+14 1.5127e+14  3118.766 < 2.2e-16
## grade                         1 1.9226e+14 1.9226e+14  3963.814 < 2.2e-16
## Residuals                 21601 1.0477e+15 4.8505e+10
##
## condition                 ***
## bedrooms                  ***
```

```
## sqft_living15            ***
## view                     ***
## sqft_above               ***
## bathrooms                ***
## waterfront               ***
## lo(yr_built, span = 0.5) ***
## grade                    ***
## Residuals
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                          Npar Df Npar F     Pr(F)
## (Intercept)
## condition
## bedrooms
## sqft_living15
## view
## sqft_above
## bathrooms
## waterfront
## lo(yr_built, span = 0.5)     2.3 223.08 < 2.2e-16 ***
## grade
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fit1
```

```
## Call:
## gam(formula = price ~ condition + bedrooms + sqft_living15 +
##     view + sqft_above + bathrooms + waterfront + lo(yr_built,
##     span = 0.5) + grade, data = Houses)
##
## Degrees of Freedom: 21612 total; 21600.72 Residual
## Residual Deviance: 1.047736e+15
```

Appendix II Linear regression

```
## Warning in price[test] - pred_linear: longer object length is not a
## multiple of shorter object length
```

```
## [1] 215764453593
```

```
##
## Call:
## lm(formula = price ~ condition + bedrooms + sqft_living15 + view +
##     sqft_above + bathrooms + waterfront + yr_built + grade, data = Houses[train,
##     ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1461377  -113035   -10731    91645  4663529
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)    8.039e+06  2.006e+05  40.075  < 2e-16 ***
## condition      2.410e+04  3.656e+03   6.591 4.59e-11 ***
## bedrooms      -2.191e+04  3.077e+03  -7.121 1.15e-12 ***
## sqft_living15  5.559e+01  5.556e+00  10.006  < 2e-16 ***
## view           5.498e+04  3.496e+03  15.727  < 2e-16 ***
## sqft_above     1.124e+02  5.165e+00  21.762  < 2e-16 ***
## bathrooms      9.673e+04  4.836e+03  20.003  < 2e-16 ***
## waterfront     6.740e+05  2.846e+04  23.684  < 2e-16 ***
## yr_built      -4.578e+03  1.022e+02 -44.802  < 2e-16 ***
## grade          1.274e+05  3.481e+03  36.599  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 229500 on 9990 degrees of freedom
## Multiple R-squared:  0.6294, Adjusted R-squared:  0.629
## F-statistic:  1885 on 9 and 9990 DF,  p-value: < 2.2e-16
```

Appendix III Polynomial regression

```
##
## Call:
## lm(formula = price ~ poly(bedrooms, degree = d, raw = TRUE) +
##     poly(bathrooms, degree = d, raw = TRUE) + poly(sqft_living,
##     degree = d, raw = TRUE) + poly(sqft_lot, degree = d, raw = TRUE) +
##     poly(floors, degree = d, raw = TRUE) + poly(waterfront, degree = d,
##     raw = TRUE) + poly(view, degree = d, raw = TRUE) + poly(condition,
##     degree = d, raw = TRUE) + poly(grade, degree = d, raw = TRUE) +
##     poly(sqft_above, degree = d, raw = TRUE) + poly(sqft_basement,
##     degree = d, raw = TRUE) + poly(yr_built, degree = d, raw = TRUE) +
##     poly(yr_renovated, degree = d, raw = TRUE) + poly(zipcode,
##     degree = d, raw = TRUE) + poly(lat, degree = d, raw = TRUE) +
##     poly(long, degree = d, raw = TRUE) + poly(sqft_living15,
##     degree = d, raw = TRUE) + poly(sqft_lot15, degree = d, raw = TRUE),
##     data = Houses[train, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2104873   -84532   -12634    66091  2161201
##
## Coefficients: (2 not defined because of singularities)
##                                                Estimate Std. Error t value
## (Intercept)                                   2.865e+10  6.536e+09   4.384
## poly(bedrooms, degree = d, raw = TRUE)1       4.944e+04  9.168e+03   5.392
## poly(bedrooms, degree = d, raw = TRUE)2      -7.060e+03  1.194e+03  -5.911
## poly(bathrooms, degree = d, raw = TRUE)1      2.401e+04  1.228e+04   1.955
## poly(bathrooms, degree = d, raw = TRUE)2      1.085e+03  2.456e+03   0.442
## poly(sqft_living, degree = d, raw = TRUE)1   -1.443e+02  1.174e+01 -12.292
## poly(sqft_living, degree = d, raw = TRUE)2    7.556e-02  2.519e-03  30.000
## poly(sqft_lot, degree = d, raw = TRUE)1       4.431e-02  1.037e-01   0.427
## poly(sqft_lot, degree = d, raw = TRUE)2       1.785e-07  9.533e-08   1.873
## poly(floors, degree = d, raw = TRUE)1        -3.027e+04  2.543e+04  -1.190
## poly(floors, degree = d, raw = TRUE)2         1.108e+04  7.431e+03   1.491
## poly(waterfront, degree = d, raw = TRUE)1     5.675e+05  2.421e+04  23.441
```

```
## poly(waterfront, degree = d, raw = TRUE)2           NA         NA      NA
## poly(view, degree = d, raw = TRUE)1           1.222e+04  8.659e+03   1.412
## poly(view, degree = d, raw = TRUE)2           1.041e+04  2.833e+03   3.675
## poly(condition, degree = d, raw = TRUE)1      3.527e+04  2.601e+04   1.356
## poly(condition, degree = d, raw = TRUE)2      7.226e+02  3.446e+03   0.210
## poly(grade, degree = d, raw = TRUE)1         -6.369e+04  1.684e+04  -3.781
## poly(grade, degree = d, raw = TRUE)2          9.660e+03  1.055e+03   9.154
## poly(sqft_above, degree = d, raw = TRUE)1     2.183e+01  1.392e+01   1.569
## poly(sqft_above, degree = d, raw = TRUE)2    -3.052e-02  3.341e-03  -9.133
## poly(sqft_basement, degree = d, raw = TRUE)1         NA         NA      NA
## poly(sqft_basement, degree = d, raw = TRUE)2 -1.097e-01  8.922e-03 -12.300
## poly(yr_built, degree = d, raw = TRUE)1      -6.480e+04  1.043e+04  -6.215
## poly(yr_built, degree = d, raw = TRUE)2       1.603e+01  2.666e+00   6.014
## poly(yr_renovated, degree = d, raw = TRUE)1  -3.775e+03  5.319e+02  -7.096
## poly(yr_renovated, degree = d, raw = TRUE)2   1.906e+00  2.664e-01   7.154
## poly(zipcode, degree = d, raw = TRUE)1       -6.123e+05  1.324e+05  -4.624
## poly(zipcode, degree = d, raw = TRUE)2        3.117e+00  6.749e-01   4.618
## poly(lat, degree = d, raw = TRUE)1            1.876e+08  8.850e+06  21.202
## poly(lat, degree = d, raw = TRUE)2           -1.967e+06  9.309e+04 -21.134
## poly(long, degree = d, raw = TRUE)1           4.927e+07  1.660e+07   2.968
## poly(long, degree = d, raw = TRUE)2           2.026e+05  6.796e+04   2.981
## poly(sqft_living15, degree = d, raw = TRUE)1  1.299e+02  1.506e+01   8.626
## poly(sqft_living15, degree = d, raw = TRUE)2 -1.546e-02  3.100e-03  -4.986
## poly(sqft_lot15, degree = d, raw = TRUE)1    -1.101e-01  1.555e-01  -0.708
## poly(sqft_lot15, degree = d, raw = TRUE)2    -7.576e-07  3.262e-07  -2.322
##                                             Pr(>|t|)
## (Intercept)                                 1.18e-05 ***
## poly(bedrooms, degree = d, raw = TRUE)1     7.12e-08 ***
## poly(bedrooms, degree = d, raw = TRUE)2     3.51e-09 ***
## poly(bathrooms, degree = d, raw = TRUE)1    0.050601 .
## poly(bathrooms, degree = d, raw = TRUE)2    0.658512
## poly(sqft_living, degree = d, raw = TRUE)1   < 2e-16 ***
## poly(sqft_living, degree = d, raw = TRUE)2   < 2e-16 ***
## poly(sqft_lot, degree = d, raw = TRUE)1     0.669218
## poly(sqft_lot, degree = d, raw = TRUE)2     0.061088 .
## poly(floors, degree = d, raw = TRUE)1       0.233897
## poly(floors, degree = d, raw = TRUE)2       0.136037
## poly(waterfront, degree = d, raw = TRUE)1    < 2e-16 ***
## poly(waterfront, degree = d, raw = TRUE)2         NA
## poly(view, degree = d, raw = TRUE)1         0.158113
## poly(view, degree = d, raw = TRUE)2         0.000239 ***
## poly(condition, degree = d, raw = TRUE)1    0.175107
## poly(condition, degree = d, raw = TRUE)2    0.833915
## poly(grade, degree = d, raw = TRUE)1        0.000157 ***
## poly(grade, degree = d, raw = TRUE)2         < 2e-16 ***
## poly(sqft_above, degree = d, raw = TRUE)1   0.116792
## poly(sqft_above, degree = d, raw = TRUE)2    < 2e-16 ***
## poly(sqft_basement, degree = d, raw = TRUE)1       NA
## poly(sqft_basement, degree = d, raw = TRUE)2  < 2e-16 ***
## poly(yr_built, degree = d, raw = TRUE)1     5.34e-10 ***
## poly(yr_built, degree = d, raw = TRUE)2     1.87e-09 ***
## poly(yr_renovated, degree = d, raw = TRUE)1 1.37e-12 ***
## poly(yr_renovated, degree = d, raw = TRUE)2 8.99e-13 ***
## poly(zipcode, degree = d, raw = TRUE)1      3.81e-06 ***
```

```
## poly(zipcode, degree = d, raw = TRUE)2       3.91e-06 ***
## poly(lat, degree = d, raw = TRUE)1            < 2e-16 ***
## poly(lat, degree = d, raw = TRUE)2            < 2e-16 ***
## poly(long, degree = d, raw = TRUE)1          0.003001 **
## poly(long, degree = d, raw = TRUE)2          0.002876 **
## poly(sqft_living15, degree = d, raw = TRUE)1  < 2e-16 ***
## poly(sqft_living15, degree = d, raw = TRUE)2 6.26e-07 ***
## poly(sqft_lot15, degree = d, raw = TRUE)1    0.478859
## poly(sqft_lot15, degree = d, raw = TRUE)2    0.020240 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 176300 on 9965 degrees of freedom
## Multiple R-squared:  0.7818, Adjusted R-squared:  0.7811
## F-statistic:  1050 on 34 and 9965 DF,  p-value: < 2.2e-16
```

```
##
## Call:
## lm(formula = price ~ poly(bedrooms, degree = d, raw = TRUE) +
##     poly(bathrooms, degree = d, raw = TRUE) + poly(sqft_living,
##     degree = d, raw = TRUE) + poly(sqft_lot, degree = d, raw = TRUE) +
##     poly(floors, degree = d, raw = TRUE) + poly(waterfront, degree = d,
##     raw = TRUE) + poly(view, degree = d, raw = TRUE) + poly(condition,
##     degree = d, raw = TRUE) + poly(grade, degree = d, raw = TRUE) +
##     poly(sqft_above, degree = d, raw = TRUE) + poly(sqft_basement,
##     degree = d, raw = TRUE) + poly(yr_built, degree = d, raw = TRUE) +
##     poly(yr_renovated, degree = d, raw = TRUE) + poly(zipcode,
##     degree = d, raw = TRUE) + poly(lat, degree = d, raw = TRUE) +
##     poly(long, degree = d, raw = TRUE) + poly(sqft_living15,
##     degree = d, raw = TRUE) + poly(sqft_lot15, degree = d, raw = TRUE),
##     data = Houses[train, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1680258   -84587   -12148    65503  1913725
##
## Coefficients: (6 not defined because of singularities)
##                                              Estimate Std. Error t value
## (Intercept)                                 1.923e+10  6.464e+09   2.975
## poly(bedrooms, degree = d, raw = TRUE)1      3.963e+04  2.019e+04   1.963
## poly(bedrooms, degree = d, raw = TRUE)2     -8.779e+03  4.875e+03  -1.801
## poly(bedrooms, degree = d, raw = TRUE)3      4.169e+02  3.750e+02   1.112
## poly(bathrooms, degree = d, raw = TRUE)1    -5.268e+04  2.616e+04  -2.014
## poly(bathrooms, degree = d, raw = TRUE)2     3.378e+04  9.699e+03   3.482
## poly(bathrooms, degree = d, raw = TRUE)3    -4.099e+03  1.094e+03  -3.747
## poly(sqft_living, degree = d, raw = TRUE)1  -4.780e+01  2.170e+01  -2.203
## poly(sqft_living, degree = d, raw = TRUE)2   4.567e-02  6.583e-03   6.937
## poly(sqft_living, degree = d, raw = TRUE)3   1.347e-06  5.158e-07   2.611
## poly(sqft_lot, degree = d, raw = TRUE)1      3.023e-01  1.678e-01   1.802
## poly(sqft_lot, degree = d, raw = TRUE)2     -6.017e-08  3.679e-07  -0.164
## poly(sqft_lot, degree = d, raw = TRUE)3     -6.306e-15  1.892e-13  -0.033
## poly(floors, degree = d, raw = TRUE)1       -2.150e+05  1.863e+05  -1.154
## poly(floors, degree = d, raw = TRUE)2        1.058e+05  1.034e+05   1.024
## poly(floors, degree = d, raw = TRUE)3       -1.550e+04  1.764e+04  -0.879
```

```
## poly(waterfront, degree = d, raw = TRUE)1     5.573e+05  2.443e+04  22.814
## poly(waterfront, degree = d, raw = TRUE)2            NA         NA      NA
## poly(waterfront, degree = d, raw = TRUE)3            NA         NA      NA
## poly(view, degree = d, raw = TRUE)1           1.216e+05  2.555e+04   4.760
## poly(view, degree = d, raw = TRUE)2          -7.397e+04  1.859e+04  -3.979
## poly(view, degree = d, raw = TRUE)3           1.462e+04  3.219e+03   4.541
## poly(condition, degree = d, raw = TRUE)1      5.242e+04  9.459e+04   0.554
## poly(condition, degree = d, raw = TRUE)2     -3.805e+03  2.651e+04  -0.144
## poly(condition, degree = d, raw = TRUE)3      3.725e+02  2.444e+03   0.152
## poly(grade, degree = d, raw = TRUE)1          9.886e+04  7.160e+04   1.381
## poly(grade, degree = d, raw = TRUE)2         -1.498e+04  8.962e+03  -1.671
## poly(grade, degree = d, raw = TRUE)3          1.193e+03  3.685e+02   3.237
## poly(sqft_above, degree = d, raw = TRUE)1     2.795e+02  2.680e+01  10.431
## poly(sqft_above, degree = d, raw = TRUE)2    -1.258e-01  1.022e-02 -12.308
## poly(sqft_above, degree = d, raw = TRUE)3     1.094e-05  1.087e-06  10.067
## poly(sqft_basement, degree = d, raw = TRUE)1         NA         NA      NA
## poly(sqft_basement, degree = d, raw = TRUE)2 -9.397e-02  1.841e-02  -5.105
## poly(sqft_basement, degree = d, raw = TRUE)3  1.813e-06  4.357e-06   0.416
## poly(yr_built, degree = d, raw = TRUE)1       9.242e+06  9.469e+05   9.760
## poly(yr_built, degree = d, raw = TRUE)2      -4.737e+03  4.836e+02  -9.795
## poly(yr_built, degree = d, raw = TRUE)3       8.091e-01  8.233e-02   9.828
## poly(yr_renovated, degree = d, raw = TRUE)1   5.982e+04  4.825e+04   1.240
## poly(yr_renovated, degree = d, raw = TRUE)2  -6.225e+01  4.854e+01  -1.282
## poly(yr_renovated, degree = d, raw = TRUE)3   1.618e-02  1.221e-02   1.325
## poly(zipcode, degree = d, raw = TRUE)1       -5.475e+05  1.300e+05  -4.212
## poly(zipcode, degree = d, raw = TRUE)2        2.787e+00  6.626e-01   4.206
## poly(zipcode, degree = d, raw = TRUE)3               NA         NA      NA
## poly(lat, degree = d, raw = TRUE)1            1.806e+08  8.727e+06  20.693
## poly(lat, degree = d, raw = TRUE)2           -1.893e+06  9.179e+04 -20.626
## poly(lat, degree = d, raw = TRUE)3                   NA         NA      NA
## poly(long, degree = d, raw = TRUE)1           4.355e+07  1.633e+07   2.666
## poly(long, degree = d, raw = TRUE)2           1.791e+05  6.687e+04   2.678
## poly(long, degree = d, raw = TRUE)3                  NA         NA      NA
## poly(sqft_living15, degree = d, raw = TRUE)1 -8.685e+01  4.239e+01  -2.049
## poly(sqft_living15, degree = d, raw = TRUE)2  6.940e-02  1.692e-02   4.101
## poly(sqft_living15, degree = d, raw = TRUE)3 -9.601e-06  2.096e-06  -4.580
## poly(sqft_lot15, degree = d, raw = TRUE)1    -1.315e+00  2.979e-01  -4.415
## poly(sqft_lot15, degree = d, raw = TRUE)2     6.226e-06  1.363e-06   4.569
## poly(sqft_lot15, degree = d, raw = TRUE)3    -7.032e-12  1.295e-12  -5.431
##                                             Pr(>|t|)
## (Intercept)                                 0.002939 **
## poly(bedrooms, degree = d, raw = TRUE)1     0.049655 *
## poly(bedrooms, degree = d, raw = TRUE)2     0.071744 .
## poly(bedrooms, degree = d, raw = TRUE)3     0.266226
## poly(bathrooms, degree = d, raw = TRUE)1    0.044072 *
## poly(bathrooms, degree = d, raw = TRUE)2    0.000499 ***
## poly(bathrooms, degree = d, raw = TRUE)3    0.000180 ***
## poly(sqft_living, degree = d, raw = TRUE)1  0.027641 *
## poly(sqft_living, degree = d, raw = TRUE)2  4.26e-12 ***
## poly(sqft_living, degree = d, raw = TRUE)3  0.009050 **
## poly(sqft_lot, degree = d, raw = TRUE)1     0.071547 .
## poly(sqft_lot, degree = d, raw = TRUE)2     0.870082
## poly(sqft_lot, degree = d, raw = TRUE)3     0.973420
## poly(floors, degree = d, raw = TRUE)1       0.248588
```

```
## poly(floors, degree = d, raw = TRUE)2        0.305929
## poly(floors, degree = d, raw = TRUE)3        0.379541
## poly(waterfront, degree = d, raw = TRUE)1     < 2e-16 ***
## poly(waterfront, degree = d, raw = TRUE)2          NA
## poly(waterfront, degree = d, raw = TRUE)3          NA
## poly(view, degree = d, raw = TRUE)1          1.96e-06 ***
## poly(view, degree = d, raw = TRUE)2          6.98e-05 ***
## poly(view, degree = d, raw = TRUE)3          5.67e-06 ***
## poly(condition, degree = d, raw = TRUE)1     0.579473
## poly(condition, degree = d, raw = TRUE)2     0.885877
## poly(condition, degree = d, raw = TRUE)3     0.878841
## poly(grade, degree = d, raw = TRUE)1         0.167374
## poly(grade, degree = d, raw = TRUE)2         0.094769 .
## poly(grade, degree = d, raw = TRUE)3         0.001211 **
## poly(sqft_above, degree = d, raw = TRUE)1     < 2e-16 ***
## poly(sqft_above, degree = d, raw = TRUE)2     < 2e-16 ***
## poly(sqft_above, degree = d, raw = TRUE)3     < 2e-16 ***
## poly(sqft_basement, degree = d, raw = TRUE)1       NA
## poly(sqft_basement, degree = d, raw = TRUE)2 3.36e-07 ***
## poly(sqft_basement, degree = d, raw = TRUE)3 0.677280
## poly(yr_built, degree = d, raw = TRUE)1       < 2e-16 ***
## poly(yr_built, degree = d, raw = TRUE)2       < 2e-16 ***
## poly(yr_built, degree = d, raw = TRUE)3       < 2e-16 ***
## poly(yr_renovated, degree = d, raw = TRUE)1  0.215037
## poly(yr_renovated, degree = d, raw = TRUE)2  0.199706
## poly(yr_renovated, degree = d, raw = TRUE)3  0.185147
## poly(zipcode, degree = d, raw = TRUE)1       2.55e-05 ***
## poly(zipcode, degree = d, raw = TRUE)2       2.62e-05 ***
## poly(zipcode, degree = d, raw = TRUE)3             NA
## poly(lat, degree = d, raw = TRUE)1            < 2e-16 ***
## poly(lat, degree = d, raw = TRUE)2            < 2e-16 ***
## poly(lat, degree = d, raw = TRUE)3                 NA
## poly(long, degree = d, raw = TRUE)1          0.007679 **
## poly(long, degree = d, raw = TRUE)2          0.007415 **
## poly(long, degree = d, raw = TRUE)3                NA
## poly(sqft_living15, degree = d, raw = TRUE)1 0.040500 *
## poly(sqft_living15, degree = d, raw = TRUE)2 4.15e-05 ***
## poly(sqft_living15, degree = d, raw = TRUE)3 4.71e-06 ***
## poly(sqft_lot15, degree = d, raw = TRUE)1    1.02e-05 ***
## poly(sqft_lot15, degree = d, raw = TRUE)2    4.96e-06 ***
## poly(sqft_lot15, degree = d, raw = TRUE)3    5.74e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 172500 on 9951 degrees of freedom
## Multiple R-squared:  0.7914, Adjusted R-squared:  0.7904
## F-statistic: 786.4 on 48 and 9951 DF,  p-value: < 2.2e-16
```

```
## Warning in price[test] - pred_poly2: longer object length is not a multiple
## of shorter object length
```

```
## [1] 237277346994
```

```
## Warning in price[test] - pred_poly3: longer object length is not a multiple
## of shorter object length
```

```
## [1] 2.38721e+11
```