SIMON FRASER UNIVERSITY

Analyzing Housing Prices in King County Using MLR, Ridge Regression and LASSO Regression

Statistics 350 - Linear Regression
Professor Harsha Perera

Authors

Samantha Yu
301 275 511, sya111

Vasena Jayamanna
301 227 954, vjayaman

Alexander Lo
301 284 737, ala148

December 3, 2018

1 Introduction

Housing prices in the Seattle area, also known as King County, are very similar to the housing prices in Vancouver. In the hopes of learning about Vancouver's housing prices, we investigated a dataset containing 21.6k records of house sales in King County from May 2014 to May 2015.

1.1 Our Variables

Variable name	Description
id	an identification variable for a house; we don't include this in our analysis
date	the date the house was sold
price	the value we want to predict, based on most of the other features
bedrooms	number of bedrooms in a given house
bathrooms	number of bathrooms per bedroom
sqft_living	square footage of the house
sqft_lot	square footage of the lot
floors	number of floors in the house
waterfront	binary value referring to whether or not the house has a waterfront view
view	number of times the house has been viewed
condition	how good is the overall condition of the house
grade	a grade given to the house based on the King County grading system
sqft_above	square footage of the house apart from the basement
sqft_basement	square footage of the basement alone
yr_built	the year the house was built
yr_renovated	the year the house was renovated
zipcode	the ZIP code; we don't include this in our analysis
lat	the latitude coordinate
long	the longitude coordinate
sqft_living15	the area of the living room as of 2015
sqft_lot15	the lot area as of 2015

The date that the house was sold is a string of the form "YYYYMMDDT000000", so we created three new features by extracting "YYYY" as the year, "MM" as the month, and "DD" as the day. We removed ID from the dataset because it does not provide any relevant information about the house. We also remove the ZIP code from the dataset because its information is also given by the latitude and longitude.

```
df <- read.csv("kc_house_data.csv")
years_sold <- laply(df$date, function(r)
                substring(r, 1, 4) %>% as.integer)
months_sold <- laply(df$date, function(r)
                substring(r, 5, 6) %>% as.integer)
days_sold <- laply(df$date, function(r)
                substring(r, 7, 8) %>% as.integer)
df_clean <- data.frame(df[,-c(1,2,17)], year_sold = years_sold,
                month_sold = months_sold, day_sold = days_sold)
```

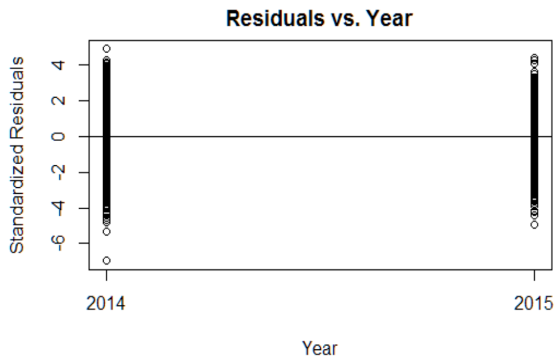
2 Original Model

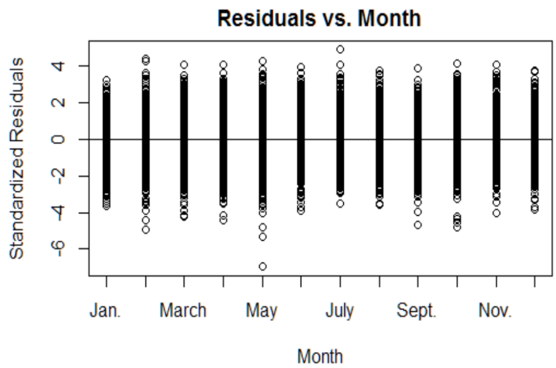
Initially, we created our model by using all 20 variables as regressors:

price ~ bedrooms + bathrooms + sqft_living + sqft_lot + floors + waterfront + view + condition + grade + sqft_above + sqft_basement + yr_built + yr_renovated + lat + long + sqft_living15 + sqft_lot15 + year_sold + month_sold + day_sold

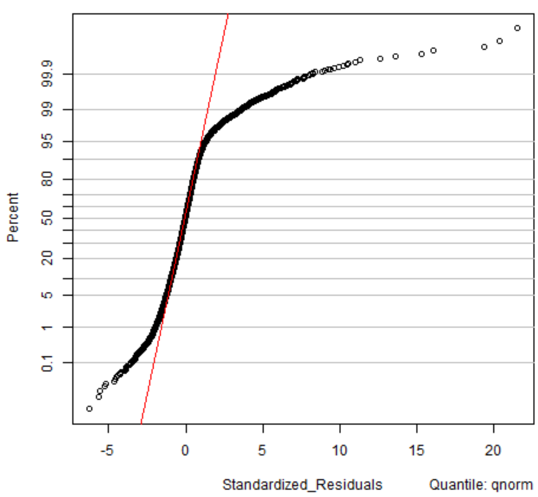
From this model's summary, our model gets an adjusted R-squared of 0.6967. However, we must check that this model does not violate our assumptions about the error terms and the relationship between the response and the regressors.

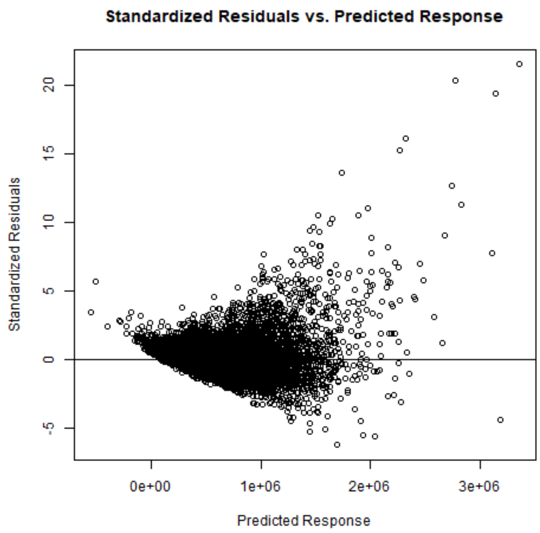
Based on the residual plots below, the residuals do not appear to vary greatly over time, so they do not violate our assumptions about constant variance with respect to time.





However, the normal probability plot of the residuals appear to have a light-tailed distribution so our model violates our normality assumptions. Similarly, the residual plot below has a funnel shape so the variance is not constant with respect to the predicted response.





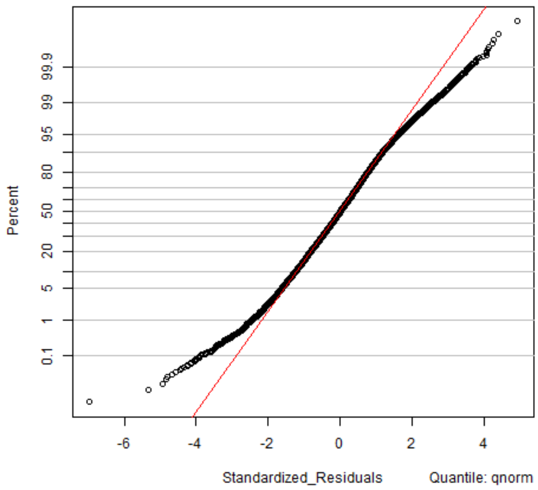
3 Log(Price) Model

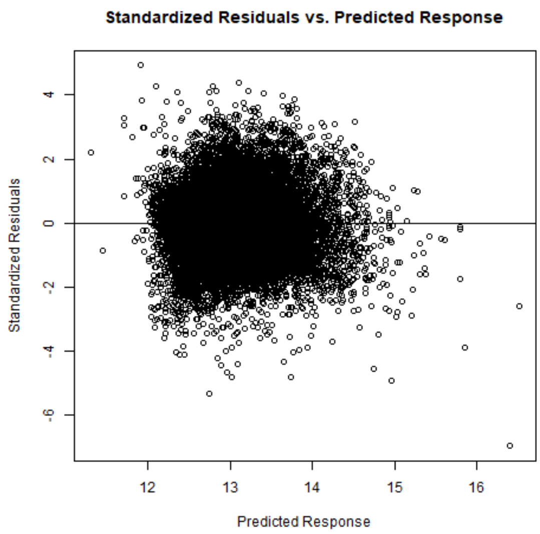
Because the range of the residuals increases over time, $\sigma^2 \propto E(y)$ so we will use the transformation: $y' = log(y)$ to get the model:

log(price) ~ bedrooms + bathrooms + sqft_living + sqft_lot + floors + waterfront + view + condition + grade + sqft_above + sqft_basement + yr_built + yr_renovated + lat + long + sqft_living15 + sqft_lot15 + year_sold + month_sold + day_sold

After transforming the response, we improved the R^2_{Adj} from 0.6967 to 0.7698 (see this model's summary). Again, we must check the adequacy of this model.

The normal probability plot of the residuals is still a light-tailed distribution but it appears to be more ideal than before. The residual plot below does not appear to be violating the constant variance assumption with respect to the predicted response.

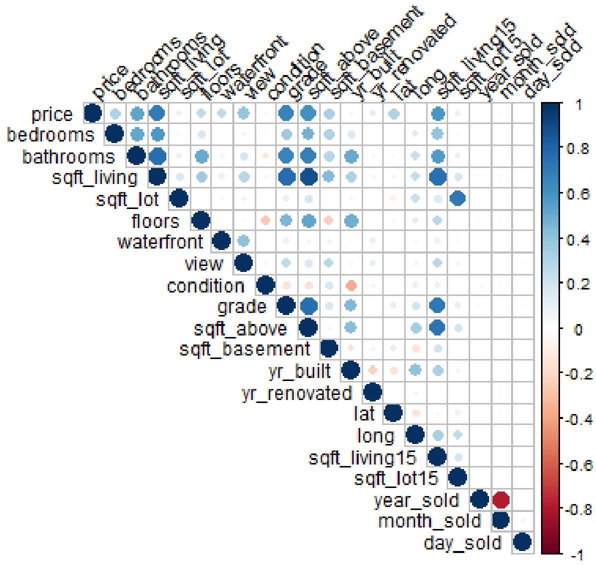




4 Multicollinearity

In the correlation plot of our dataset, we see that `sqft_living` is strongly correlated with `sqft_above` and `sqft_living15`. This result is expected given that the area specified by `sqft_living` contains `sqft_above`, and `sqft_living` and `sqft_living15` would have often have the same values. `sqft_living` is also positively correlated to `grade`, probably because houses with more land area would tend to be given a better grade. Since the dataset is only from May 2014 to May 2015, the reason why `year_sold` and `month_sold` are negatively correlated with each other could be that `year_sold` is basically specifying whether the house was sold either in the months January to May, or May to December.

```
corrplot(cor(df_clean), type="upper", tl.col="black", tl.srt=45)
```



In the summary of the full linear model with all regressors included, there is a row of NAs for `sqft_basement` which indicates multicollinearity, as supported by the correlation plot.

We used the `alias()` function to check for linear dependence, and found that `sqft_above` and `sqft_living` were highly correlated with `sqft_basement`.

We tried different configurations of removing each of these three regressors from the model, and found the fewest large variance inflation factors if we removed `sqft_basement` from the model:

log(price) ∼ bedrooms + bathrooms + sqft_living + sqft_lot + floors + waterfront + view + condition + grade + sqft_above + ~~sqft_basement~~ + yr_built + yr_renovated + lat + long + sqft_living15 + sqft_lot15 + year_sold + month_sold + day_sold

The summary for this model without `sqft_basement` no longer includes the rows of NAs and we have the same adjusted R-squared as before (0.7698). However, there are still VIFs greater than 5 for `sqft_living` and `sqft_above` indicating that there remains some multicollinearity.

```
lm_no_basement <- lm(price ~ . - sqft_basement, data=df_clean)
vif(lm_no_basement)
```

```
  bedrooms   bathrooms  sqft_living     sqft_lot       floors   waterfront         view    condition        grade
  1.648487    3.350149     8.641323     2.102658     1.995130     1.203662     1.423844     1.238287     3.414443
sqft_above    yr_built yr_renovated          lat         long sqft_living15   sqft_lot15    year_sold   month_sold
  6.955766    2.390964     1.150676     1.124462     1.501798     2.953509     2.135714     2.613956     2.613214
  day_sold
  1.012506
```

5 Variable Selection

Forwards selection, backwards elimination, and stepwise selection on our dataset all resulted in the exact same model:

log(price) ~ bedrooms + bathrooms + sqft_living + sqft_lot + floors + waterfront + view + condition + grade + sqft_above + ~~sqft_basement~~ + yr_built + yr_renovated + lat + long + sqft_living15 + sqft_lot15 + year_sold + month_sold + day_sold

Namely, these variable selection techniques removed `sqft_basement` as we suspected would be beneficial in the previous Multicollinearity section.

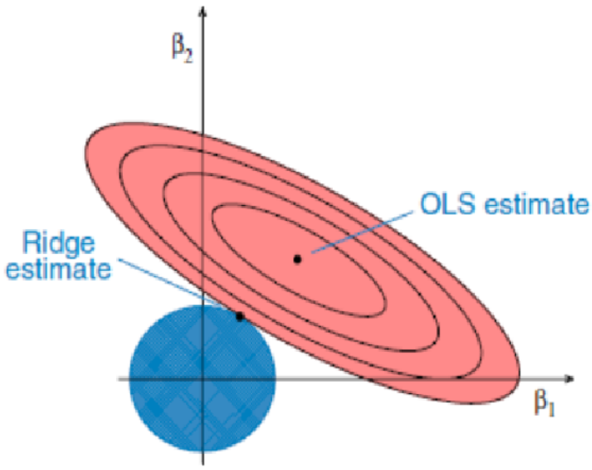
6 Ridge Regression

The general OLS method in linear regression involves minimizing the residual sum of squares. Ridge regression is a modification of this, where we instead try to minimize the residual sum of squares plus a penalty term λ times the sum of squared coefficients,

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2, \tag{1}$$

which limits both the residuals and the magnitude of the coefficients, the latter to curb the variance.

Geometric interpretation of Ridge Regression[1]:



This can be visualized by a figure where the ellipses represent the residual sum of squares, and the circle represents the sum of the squared coefficients (multiplied by a tuning parameter λ) being less than some value c. With $0 < \lambda < \infty$, larger λ values penalize larger coefficients, and so we have restrictions on how large the coefficients can be, while still attempting to minimize the residual sum of squares. λ can be found using cross validation.

We adjust λ to help shrink the coefficients term (which does not include the intercept). Note that $\sum_{j=1}^{p} \beta_j^2$ is the square of the L2 norm of the β vector.

6.1 Input preparation

We begin the input preparation for finding a ridge regression model by using the log-transformed response variable and all of the regressors we are considering.

```
x = model.matrix(log(price) ~ ., df_clean)[,-1]
y = log(df_clean$price)
grid = 10^seq(10,-2,length=100)
```

`model.matrix()` gets the design matrix, and `grid` is a sequence of λ values to draw from.

6.2 Training and Test Sets

We split the housing data into a training set for model fitting, and a test or validation set for estimating the error.

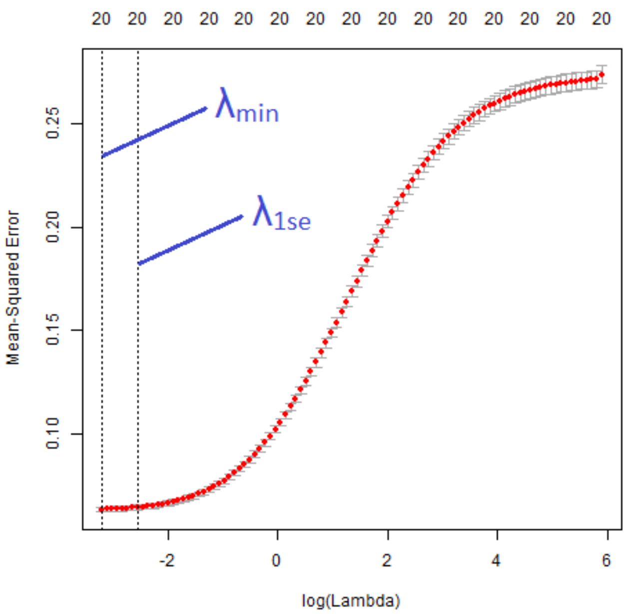
```
set.seed(222)
train = sample(1:nrow(x), nrow(x)/2)
test = (-train)
y.train = y[train]
y.test = y[test]
```

The `train` variable is a random sample of indices of our x-values, with total length being half the size of x.

6.3 Selecting the best lambda in ridge, via cross-validation

We then run 10-fold cross-validation on the training set to find optimal values for λ.

```
cv.out=cv.glmnet(x[train,], y[train], alpha=0, nfolds=10)
plot(cv.out)
```



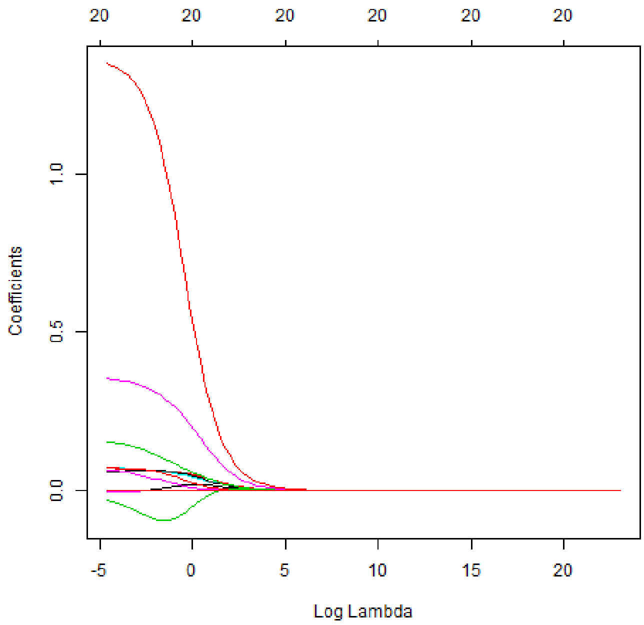
We have a plot of the cross-validation curve in red, with error bars of the standard deviation of the mean-squared error estimates. The dotted lines indicate the positions of key λ values. $\lambda_{min} = 0.04026129$ gives the minimum mean-squared error and $\lambda_{1se} = 0.07721754$ is the largest λ where the error is within one standard error of the minimum error. We select λ_{min} for prediction, as for ridge regression the main difference between the two is the mean squared error, not the number of variables.

```
ridge.mod=glmnet(x[train,],y[train],alpha=0,lambda=grid)
ridge.pred=predict(ridge.mod,s=bestlam,newx=x[test,])
(mse.ridge = mean((ridge.pred-y.test)^2))
```

Note that **ridge.mod** is the fitted model object, **s** is the penalty parameter λ for which we want predictions, and **newx** is the matrix of x-values (in the test set) for which we want predictions. We used **glmnet** to create the fitted model, and used the test set and the predicted coefficients to calculate the mean squared errors on the test set. We verified with both λ values that although the MSE difference is marginal (a difference of 0.00096999), $\lambda_{min} = 0.04026129$ still gives a better result.

6.4 Plotting the estimated coefficients vs log(λ)

```
plot(ridge.mod, xvar = "lambda")
```



Note that while the coefficients are shrunk "towards" zero as log(λ) increases, the number of variables in the model remains the same. With ridge regression, we can shrink coefficients but not remove them entirely. Using our results so far, we refit the ridge regression model using the full dataset,

```
y_predict <- predict(out, type = "response", s = bestlam, newx = x)
sst <- sum((y - mean(y))^2)
sse <- sum((y_predict - y)^2)
rsq <- 1 - sse / sst
rsq_adj <- 1 - ((1-rsq)*(nrow(x)-1))/(nrow(x) - ncol(x) - 1)
```

and then log-transformed the response variable, resulting in an adjusted R-squared value of 0.7680 (see summary).

7 LASSO

LASSO is an acronym for Least Absolute Shrinkage and Selection Operator and it is similar to ridge regression. The main difference is LASSO is able to make the model more interpretable by performing variable selection. This is done by constraining the sum of the absolute value of the regression coefficients to be less than a fixed value, which reduces the coefficients of certain regression variables to 0. Similar to ridge regression, the goal is to minimize the residual sum of squares plus a penalty term.

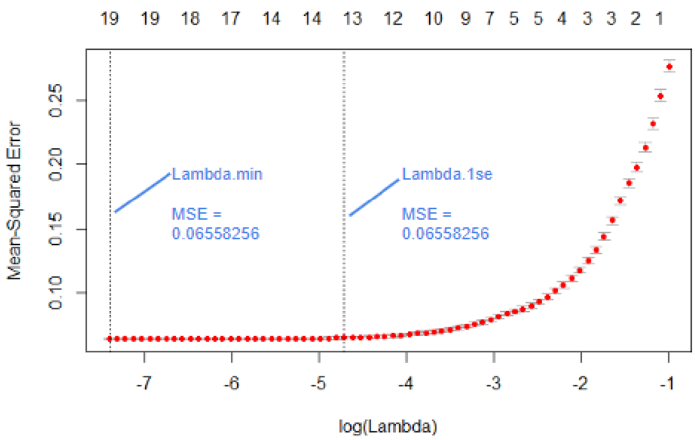
$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j| \quad (2)$$

The tuning parameter λ controls the strength of the penalty term and is always greater or equal to 0. When we increase the value of λ, the bias increases, the variances decreases, and more coefficients will be reduced to exactly 0. However when $\lambda = 0$, the penalty term also reduces to 0 and we have OLS (Ordinary Least Squares) because we are only minimizing the residual sum of squares.

The optimal λ is obtained by using cross validation on the training set. Generally, LASSO is better than ridge at reducing the variance of models which have several insignificant variables.

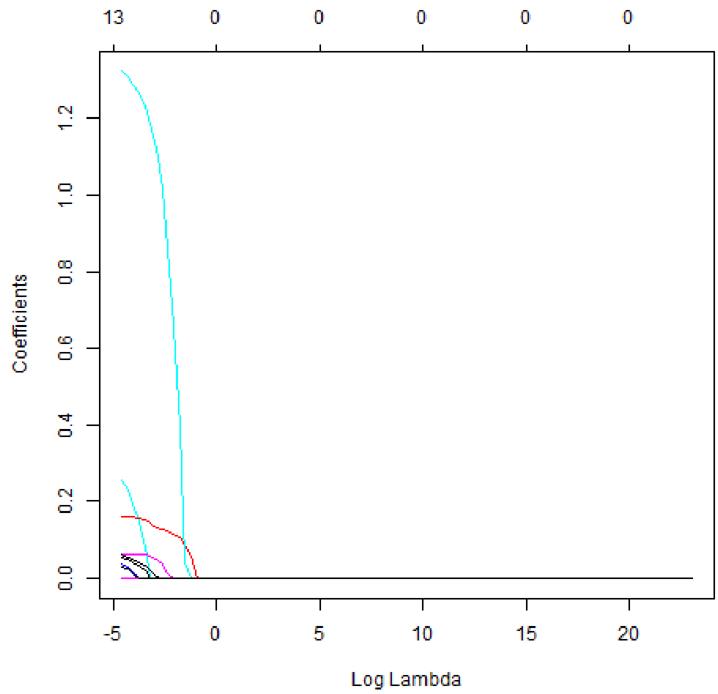
```
x = (model.matrix(log(price) ~., df_clean)[,-1])
y = log(df_clean$price)
grid = 10^seq(10,-2,length=100)
set.seed(1)
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.train = y[train]
y.test = y[test]
x.train = x[train]
x.test = x[test]
cv.lasso=cv.glmnet(x[train,],y[train],alpha=1)
plot(cv.lasso)
```

The data is split into the training set and the test set and cross-validation is performed to find the optimal values of λ.



The cross-validation plot shows MSE versus $\log(\lambda)$. Lambda.min is the value of λ with the smallest MSE and λ_{1se} is the value of λ one standard error away from the minimum. We decided to use Lambda.1se in our prediction because both values of MSE are nearly equal and Lambda.1se leaves us with fewer regressor variables in our model (13) compared to the model using Lambda.min (19).

```
lasso.model=glmnet(x[train,],y[train],alpha=1,lambda=grid)
plot(lasso.model, xvar = "lambda")
```



This graph shows the coefficients of the model reducing to 0 as λ grows large, thus performing variable selection.

7.0.1 With lambda_1se

```
lasso.prediction=predict(lasso.model,s=lambda_1se,newx=x[test,])
full.refit=glmnet(x,y,alpha=1,lambda=grid)
lasso.coeff=predict(full.refit,type="coefficients",s=lambda_1se)[1:21,]
lasso.coeff[lasso.coeff!=0]
mse.lasso_min = mean((lasso.prediction-y.test)^2)
y_predict <- predict(full.refit, type = "response",
                    s = mse.lasso_1se, newx = x)
sst <- sum((y - mean(y))^2)
sse <- sum((y_predict - y)^2)
rsq <- 1 - sse / sst
rsq_adj <- 1 - ((1-rsq)*(nrow(x)-1))/(nrow(x) - ncol(x) - 1)
```

```
  (Intercept)     bathrooms   sqft_living      sqft_lot        floors    waterfront          view
-9.945557e+01  5.373980e-02  1.439250e-04  6.013460e-08  3.869322e-02  2.963331e-01  5.820002e-02
    condition         grade      yr_built  yr_renovated           lat sqft_living15     year_sold
 5.362552e-02  1.587138e-01 -2.667927e-03  2.639044e-05  1.318415e+00  8.585540e-05  2.631134e-02
```

In the summary, we see that the final model using Lambda.1se has 13 regressor variables (excluding the intercept), a MSE of 0.06558256, and an adjusted R-squared of 0.7654432:

log(price) $\sim$ ~~bedrooms~~ + bathrooms + sqft_living + sqft_lot + floors + waterfront + view + condition + grade + ~~sqft_above~~ + ~~sqft_basement~~ + yr_built + yr_renovated + lat + ~~long~~ + sqft_living15 + ~~sqft_lot15~~ + year_sold + ~~month_sold~~ + ~~day_sold~~

8 Conclusion

Comparing the models from variable selection methods, LASSO regression, and ridge regression, we found the values of the MSEs and adjusted R-squared to be similar.

Models	AIC	Adjusted R-Squared	MSE	# of Variables
Variable Selection	-59443.17	0.7698	0.06378777	19
LASSO	N/A	0.7654432	0.06558256	13
Ridge	N/A	0.7679732	0.0646233	20

We selected the model given by LASSO regression to be our final model because it contains the fewest regressors:

$\log(\widehat{price})$ = -1.503e+02 + 6.966e-02 bathrooms + 1.315e-04 sqft_living + 3.146e-07 sqft_lot + 6.671e-02 floors + 3.798e-01 waterfront + 5.948e-02 view + 6.932e-02 condition + 1.631e-01 grade - 3.332e-03 yr_built + 3.960e-05 yr_renovated + 1.369e+00 lat + 9.839e-05 sqft_living15 + 5.092e-02 year_sold

Surprisingly, `price` decreases by $e^{3.332\times10^{-3}}$ when `yr_built` increases by one year when all other regressors in our final model are held constant. Perhaps the lot size takes precedence over `yr_built`. Because housing prices were cheaper back then, older houses tend to be built on larger plots of land so `yr_built` would be negatively correlated with `price`.

Based on our plot of varImp() results, we noticed that `latitude` and `waterfront` were the two most important features in our model. Using Tableau, we plotted the housing prices on a map of King County. We see that the most expensive houses are by the bodies of water in King County: Lake Washington, Lake Union, Lake Sammamish, and Puget Sound. The cheaper houses are farther away from the waterfront and from Downtown Seattle. Because the bodies of water in King County stretch vertically, LASSO removed `longitude` from the model as `longitude` would be given by `waterfront` in this dataset. If we wanted to verify this theory, we would use a dataset from a region where the bodies of water stretch horizontally and expect that `latitude` is discarded from the model instead.

For future research, we would investigate the effect of outliers on our dataset. For example, data point 15871 has 33 bedrooms, which is most likely an error made while recording the data. Besides splitting the date that the house was sold into year, month, and day, we could also have investigated whether different seasons of the year or different days of the week influenced the price. Because our dataset includes latitude and longitude, we could also have created features based on the distances from the houses to prominent locations, such as Downtown Seattle and the Microsoft headquarters.

9 Appendix

9.1 Summary of the Original Model

```
Call:
lm(formula = price ~ ., data = df_clean)

Residuals:
     Min       1Q   Median       3Q      Max 
-1265615   -99592    -9313    76514  4346722 

Coefficients: (1 not defined because of singularities)
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -1.116e+08  9.712e+06 -11.492  < 2e-16 ***
bedrooms     -3.437e+04  1.898e+03 -18.103  < 2e-16 ***
bathrooms     4.237e+04  3.268e+03  12.963  < 2e-16 ***
sqft_living   1.469e+02  4.402e+00  33.367  < 2e-16 ***
sqft_lot      1.240e-01  4.814e-02   2.577  0.00998 ** 
floors        1.305e+03  3.597e+03   0.363  0.71687    
waterfront    5.884e+05  1.744e+04  33.741  < 2e-16 ***
view          4.914e+04  2.141e+03  22.947  < 2e-16 ***
condition     3.240e+04  2.352e+03  13.779  < 2e-16 ***
grade         9.744e+04  2.162e+03  45.072  < 2e-16 ***
sqft_above    3.277e+01  4.380e+00   7.482 7.58e-14 ***
sqft_basement        NA         NA      NA       NA    
yr_built     -2.455e+03  7.239e+01 -33.912  < 2e-16 ***
yr_renovated  2.262e+01  3.673e+00   6.158 7.49e-10 ***
lat           5.635e+05  1.052e+04  53.544  < 2e-16 ***
long         -1.169e+05  1.197e+04  -9.766  < 2e-16 ***
sqft_living15 2.756e+01  3.448e+00   7.993 1.38e-15 ***
sqft_lot15   -3.918e-01  7.361e-02  -5.322 1.03e-07 ***
year_sold     3.705e+04  4.755e+03   7.792 6.90e-15 ***
month_sold    1.296e+03  7.136e+02   1.816  0.06938 .  
day_sold     -3.195e+02  1.603e+02  -1.994  0.04622 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 202200 on 21593 degrees of freedom
Multiple R-squared:  0.697,	Adjusted R-squared:  0.6967 
F-statistic:  2614 on 19 and 21593 DF,  p-value: < 2.2e-16
```

9.2 Summary of the Log Model

```
Call:
lm(formula = log(price) ~ . - sqft_basement, data = df_clean)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.76109 -0.15889  0.00298  0.15727  1.24460 

Coefficients:
                Estimate Std. Error t value Pr(>|t|)    
(Intercept)   -1.798e+02  1.214e+01 -14.814  < 2e-16 ***
bedrooms      -1.078e-02  2.373e-03  -4.542 5.60e-06 ***
bathrooms      7.059e-02  4.085e-03  17.281  < 2e-16 ***
sqft_living    1.478e-04  5.501e-06  26.866  < 2e-16 ***
sqft_lot       4.641e-07  6.017e-08   7.713 1.28e-14 ***
floors         6.951e-02  4.496e-03  15.461  < 2e-16 ***
waterfront     3.775e-01  2.180e-02  17.320  < 2e-16 ***
view           5.609e-02  2.676e-03  20.956  < 2e-16 ***
condition      7.011e-02  2.939e-03  23.854  < 2e-16 ***
grade          1.608e-01  2.702e-03  59.510  < 2e-16 ***
sqft_above    -1.355e-05  5.474e-06  -2.475 0.013332 *  
yr_built      -3.227e-03  9.048e-05 -35.662  < 2e-16 ***
yr_renovated   4.032e-05  4.590e-06   8.784  < 2e-16 ***
lat            1.358e+00  1.315e-02 103.239  < 2e-16 ***
long          -5.050e-02  1.496e-02  -3.376 0.000736 ***
sqft_living15  1.052e-04  4.310e-06  24.403  < 2e-16 ***
sqft_lot15    -2.701e-07  9.200e-08  -2.936 0.003327 ** 
year_sold      6.268e-02  5.943e-03  10.547  < 2e-16 ***
month_sold     2.229e-03  8.919e-04   2.499 0.012464 *  
day_sold      -4.615e-04  2.003e-04  -2.304 0.021221 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2527 on 21593 degrees of freedom
Multiple R-squared:   0.77,	Adjusted R-squared:  0.7698 
F-statistic:  3806 on 19 and 21593 DF,  p-value: < 2.2e-16
```

9.3 Summary of the Model with No sqft_basement

```
Call:
lm(formula = price ~ . - sqft_basement, data = df_clean)

Residuals:
     Min       1Q   Median       3Q      Max 
-1265615   -99592    -9313    76514  4346722 

Coefficients:
                Estimate Std. Error t value Pr(>|t|)    
(Intercept)   -1.116e+08  9.712e+06 -11.492  < 2e-16 ***
bedrooms      -3.437e+04  1.898e+03 -18.103  < 2e-16 ***
bathrooms      4.237e+04  3.268e+03  12.963  < 2e-16 ***
sqft_living    1.469e+02  4.402e+00  33.367  < 2e-16 ***
sqft_lot       1.240e-01  4.814e-02   2.577  0.00998 ** 
floors         1.305e+03  3.597e+03   0.363  0.71687    
waterfront     5.884e+05  1.744e+04  33.741  < 2e-16 ***
view           4.914e+04  2.141e+03  22.947  < 2e-16 ***
condition      3.240e+04  2.352e+03  13.779  < 2e-16 ***
grade          9.744e+04  2.162e+03  45.072  < 2e-16 ***
sqft_above     3.277e+01  4.380e+00   7.482 7.58e-14 ***
yr_built      -2.455e+03  7.239e+01 -33.912  < 2e-16 ***
yr_renovated   2.262e+01  3.673e+00   6.158 7.49e-10 ***
lat            5.635e+05  1.052e+04  53.544  < 2e-16 ***
long          -1.169e+05  1.197e+04  -9.766  < 2e-16 ***
sqft_living15  2.756e+01  3.448e+00   7.993 1.38e-15 ***
sqft_lot15    -3.918e-01  7.361e-02  -5.322 1.03e-07 ***
year_sold      3.705e+04  4.755e+03   7.792 6.90e-15 ***
month_sold     1.296e+03  7.136e+02   1.816  0.06938 .  
day_sold      -3.195e+02  1.603e+02  -1.994  0.04622 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 202200 on 21593 degrees of freedom
Multiple R-squared:  0.697,	Adjusted R-squared:  0.6967 
F-statistic:  2614 on 19 and 21593 DF,  p-value: < 2.2e-16
```

9.4 Summary of the Model Produced by Variable Selection

```
Call:
lm(formula = log(price) ~ grade + lat + sqft_living + yr_built +
    view + bathrooms + sqft_living15 + condition + waterfront +
    floors + year_sold + yr_renovated + sqft_lot + bedrooms +
    long + sqft_lot15 + month_sold + sqft_above + day_sold, data = df_clean)

Residuals:
     Min       1Q   Median       3Q      Max
-1.76109 -0.15889  0.00298  0.15727  1.24460

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)   -1.798e+02  1.214e+01 -14.814  < 2e-16 ***
grade          1.608e-01  2.702e-03  59.510  < 2e-16 ***
lat            1.358e+00  1.315e-02 103.239  < 2e-16 ***
sqft_living    1.478e-04  5.501e-06  26.866  < 2e-16 ***
yr_built      -3.227e-03  9.048e-05 -35.662  < 2e-16 ***
view           5.609e-02  2.676e-03  20.956  < 2e-16 ***
bathrooms      7.059e-02  4.085e-03  17.281  < 2e-16 ***
sqft_living15  1.052e-04  4.310e-06  24.403  < 2e-16 ***
condition      7.011e-02  2.939e-03  23.854  < 2e-16 ***
waterfront     3.775e-01  2.180e-02  17.320  < 2e-16 ***
floors         6.951e-02  4.496e-03  15.461  < 2e-16 ***
year_sold      6.268e-02  5.943e-03  10.547  < 2e-16 ***
yr_renovated   4.032e-05  4.590e-06   8.784  < 2e-16 ***
sqft_lot       4.641e-07  6.017e-08   7.713 1.28e-14 ***
bedrooms      -1.078e-02  2.373e-03  -4.542 5.60e-06 ***
long          -5.050e-02  1.496e-02  -3.376 0.000736 ***
sqft_lot15    -2.701e-07  9.200e-08  -2.936 0.003327 **
month_sold     2.229e-03  8.919e-04   2.499 0.012464 *
sqft_above    -1.355e-05  5.474e-06  -2.475 0.013332 *
day_sold      -4.615e-04  2.003e-04  -2.304 0.021221 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2527 on 21593 degrees of freedom
Multiple R-squared:   0.77,     Adjusted R-squared:  0.7698
F-statistic:  3806 on 19 and 21593 DF,  p-value: < 2.2e-16
```

9.5 Ridge Regression Final Model

The final ridge regression model has an adjusted R-squared value of 0.7680:

$\log(\widehat{price})$ = -153.5478 - 0.007386 bedrooms + 0.06525 bathrooms + 7.7068e-5 sqft_living + 3.9985e-07 sqft_lot + 0.06661 floors + 0.3570 waterfront + 0.05787 view + 0.06689 condition + 0.1390 grade + 6.9248e-05 sqft_above + 7.9394e-05 sqft_basement - 0.002698 yr_built + 4.5048e-05 yr_renovated + 1.2882 lat - 0.0843 long + 0.0001087 sqft_living15 - 1.8322e-07 sqft_lot15 + 0.04879 year_sold + 0.0006612 month_sold - 0.0005026 day_sold

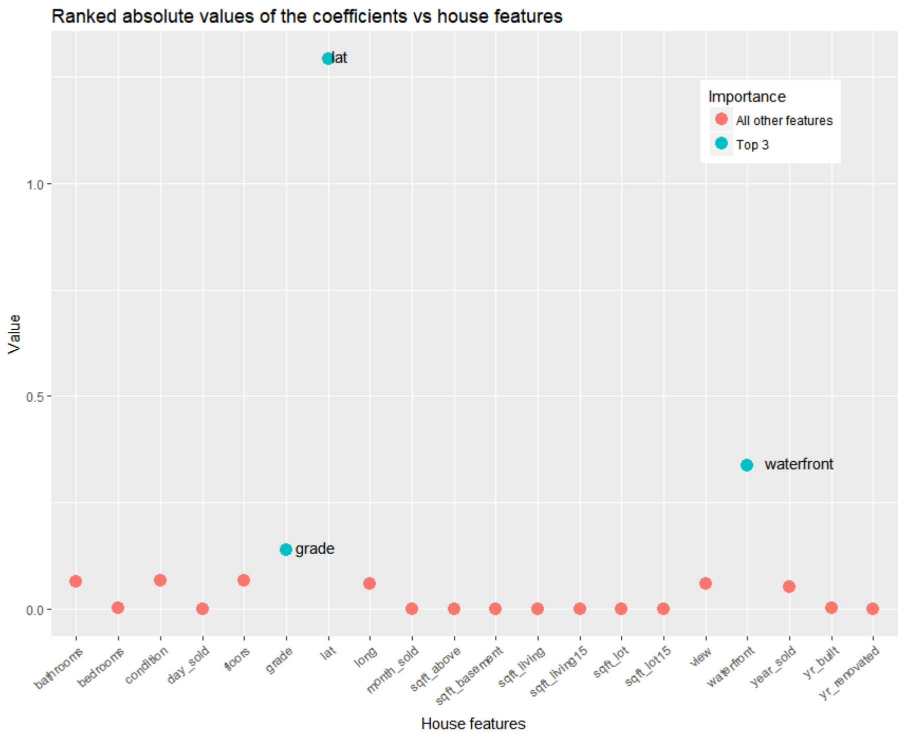
9.6 LASSO Regression Final Model

```
Call:
lm(formula = log(price) ~ bathrooms + sqft_living + sqft_lot +
    floors + waterfront + view + condition + grade + yr_built +
    yr_renovated + lat + sqft_living15 + year_sold, data = df_clean)

Residuals:
     Min       1Q   Median       3Q      Max
-1.73436 -0.15960  0.00252  0.15724  1.25187

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)   -1.503e+02  7.482e+00 -20.086  < 2e-16 ***
bathrooms      6.966e-02  3.951e-03  17.632  < 2e-16 ***
sqft_living    1.315e-04  3.939e-06  33.389  < 2e-16 ***
sqft_lot       3.146e-07  4.257e-08   7.389 1.54e-13 ***
floors         6.671e-02  4.025e-03  16.572  < 2e-16 ***
waterfront     3.798e-01  2.179e-02  17.435  < 2e-16 ***
view           5.948e-02  2.611e-03  22.777  < 2e-16 ***
condition      6.932e-02  2.927e-03  23.679  < 2e-16 ***
grade          1.631e-01  2.628e-03  62.041  < 2e-16 ***
yr_built      -3.332e-03  8.522e-05 -39.104  < 2e-16 ***
yr_renovated   3.960e-05  4.586e-06   8.635  < 2e-16 ***
lat            1.369e+00  1.303e-02 105.056  < 2e-16 ***
sqft_living15  9.839e-05  4.121e-06  23.875  < 2e-16 ***
year_sold      5.092e-02  3.691e-03  13.797  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.253 on 21599 degrees of freedom
Multiple R-squared:  0.7693,    Adjusted R-squared:  0.7692
F-statistic:  5542 on 13 and 21599 DF,  p-value: < 2.2e-16
```

9.7 varImp() Results

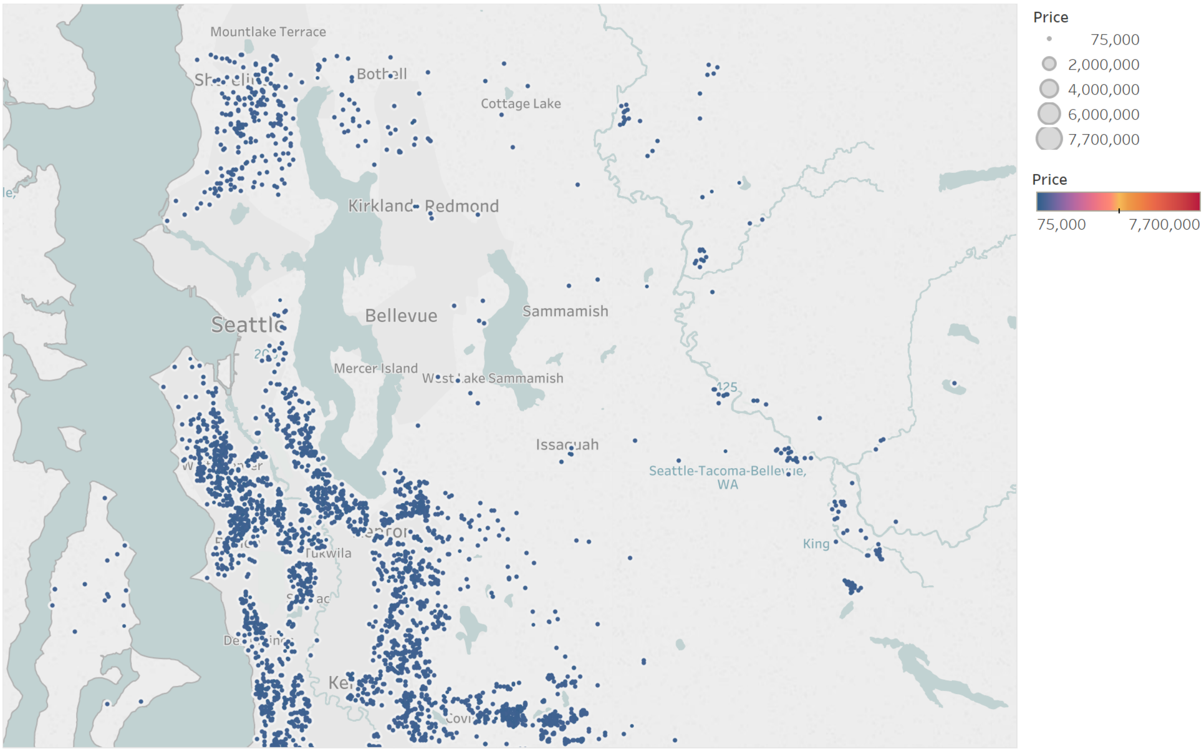
Important: Note that the following steps use `varImp()` with scaled coefficients, but they are not standardized. When we standardized the coefficients, we found that features known to be highly correlated like `sqft_basement` and `sqft_living` rose in "importance". For the purposes of understanding the model, we decided to visualize the ranked absolute values of the coefficients scaled to 0 and 100. This visual is not for supporting statistical significance, as we are comparing regressor coefficients without standardizing them, but for assisting with our understanding of the final ridge regression model.

```
library(caret)
a <- data.frame(varImp(ridge.mod, lambda = bestlam, scale = TRUE))
a <- data.frame("House features" = rownames(a),
       Importance = a$Overall, stringsAsFactors = FALSE)
a <- a[order(-a$Importance),]
b <- data.frame(a, Value = c(rep("Top 3", 3),
       rep("All other features", dim(a)[1]-3)),
       stringsAsFactors = FALSE)

ggplot(b, aes(x=b$House.features, y=b$Importance, color=b$Value)) +
  theme(axis.text.x = element_text(angle = 40, hjust = 1),
     legend.position = c(0.85,0.85)) + geom_point(size = 4) +
  geom_text(aes(label= ifelse(Value == "Top 3",
     as.character(b$House.features), ""),
     hjust=-0.25,vjust=0.25), color = "black") +
  labs(x = "House features", y = "Value", color = "Importance",
     title = paste0("Ranked absolute values of the",
       " coefficients vs house features"))
```

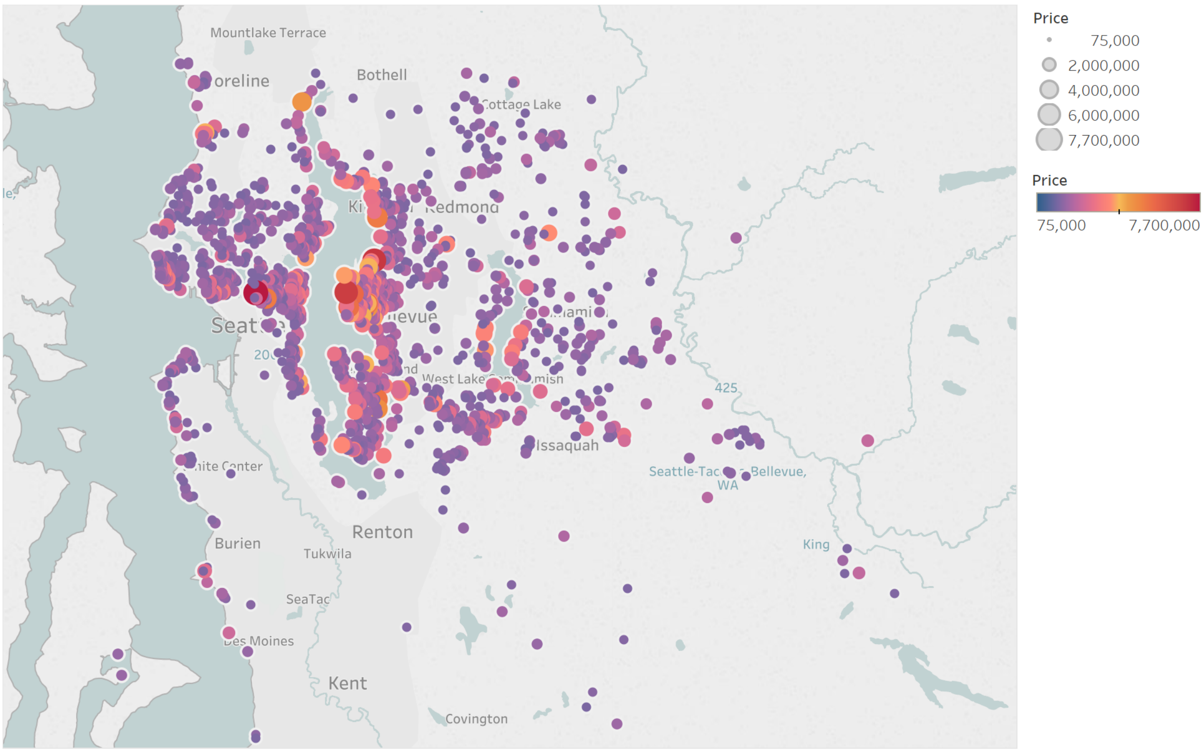


9.8 Maps of Housing Prices in King County

Houses Sold for <$300,000



Houses Sold for >$1,000,000



9.9 Required packages

MASS, stats4, magrittr, car, corrplot, e1071, carets, olsrr, glmnet, plyr.

References

[1] *Geometric Interpretation of Ridge Regression.* 5.1 - Ridge Regression. 1.5 - The Coefficient of Determination, r-Squared. — STAT 501, onlinecourses.science.psu.edu/stat857/node/155/.