Alex Look

DSC 324
Homework Module 2

1/23/22

1a) Copy of dataset with select columns

```
   mtcars.cyl      mtcars.disp      mtcars.hp    
 Min.   :4.000   Min.   : 71.1   Min.   : 52.0  
 1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5  
 Median :6.000   Median :196.3   Median :123.0  
 Mean   :6.188   Mean   :230.7   Mean   :146.7  
 3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0  
 Max.   :8.000   Max.   :472.0   Max.   :335.0  
   mtcars.wt      mtcars.carb   
 Min.   :1.513   Min.   :1.000  
 1st Qu.:2.581   1st Qu.:2.000  
 Median :3.325   Median :2.000  
 Mean   :3.217   Mean   :2.812  
 3rd Qu.:3.610   3rd Qu.:4.000  
 Max.   :5.424   Max.   :8.000  
```

1b) Column of 1's named "count"

Description: df [32 x 6]

count <dbl>	mtcars.cyl <dbl>	mtcars.disp <dbl>	mtcars.hp <dbl>	mtcars.wt <dbl>	mtcars.carb <dbl>
1	6	160.0	110	2.620	4
1	6	160.0	110	2.875	4
1	4	108.0	93	2.320	1
1	6	258.0	110	3.215	1
1	8	360.0	175	3.440	2
1	6	225.0	105	3.460	1
1	8	360.0	245	3.570	4
1	4	146.7	62	3.190	2
1	4	140.8	95	3.150	2
1	6	167.6	123	3.440	4
1	6	167.6	123	3.440	4
1	8	275.8	180	4.070	3
1	8	275.8	180	3.730	3

1-13 of 32 rows   Previous 1 2 3 Next

1c) Convert dataset into matrix

	count	mtcars.cyl	mtcars.disp	mtcars.hp	mtcars.wt
[1,]	1	6	160.0	110	2.620
[2,]	1	6	160.0	110	2.875
[3,]	1	4	108.0	93	2.320
[4,]	1	6	258.0	110	3.215
[5,]	1	8	360.0	175	3.440
[6,]	1	6	225.0	105	3.460
[7,]	1	8	360.0	245	3.570
[8,]	1	4	146.7	62	3.190
[9,]	1	4	140.8	95	3.150
[10,]	1	6	167.6	123	3.440
[11,]	1	6	167.6	123	3.440
[12,]	1	8	275.8	180	4.070
[13,]	1	8	275.8	180	3.730
[14,]	1	8	275.8	180	3.780
[15,]	1	8	472.0	205	5.250
[16,]	1	8	460.0	215	5.424
[17,]	1	8	440.0	230	5.345
[18,]	1	4	78.7	66	2.200
[19,]	1	4	75.7	52	1.615
[20,]	1	4	71.1	65	1.835
[21,]	1	4	120.1	97	2.465
[22,]	1	8	318.0	150	3.520
[23,]	1	8	304.0	150	3.435
[24,]	1	8	350.0	245	3.840
[25,]	1	8	400.0	175	3.845
[26,]	1	4	79.0	66	1.935
[27,]	1	4	120.3	91	2.140
[28,]	1	4	95.1	113	1.513
[29,]	1	8	351.0	264	3.170
[30,]	1	6	145.0	175	2.770
[31,]	1	8	301.0	335	3.570
[32,]	1	4	121.0	109	2.780

```
      mtcars.carb
 [1,]           4
 [2,]           4
 [3,]           1
 [4,]           1
 [5,]           2
 [6,]           1
 [7,]           4
 [8,]           2
 [9,]           2
[10,]           4
[11,]           4
[12,]           3
[13,]           3
[14,]           3
[15,]           4
[16,]           4
[17,]           4
[18,]           1
[19,]           2
[20,]           1
[21,]           1
[22,]           2
[23,]           2
[24,]           4
[25,]           2
[26,]           1
[27,]           2
[28,]           2
[29,]           4
[30,]           6
[31,]           8
[32,]           2
```

```
      mtcars.mpg
 [1,]       21.0
 [2,]       21.0
 [3,]       22.8
 [4,]       21.4
 [5,]       18.7
 [6,]       18.1
 [7,]       14.3
 [8,]       24.4
 [9,]       22.8
[10,]       19.2
[11,]       17.8
[12,]       16.4
[13,]       17.3
[14,]       15.2
[15,]       10.4
[16,]       10.4
[17,]       14.7
[18,]       32.4
[19,]       30.4
[20,]       33.9
[21,]       21.5
[22,]       15.5
[23,]       15.2
[24,]       13.3
[25,]       19.2
[26,]       27.3
[27,]       26.0
[28,]       30.4
[29,]       15.8
[30,]       19.7
[31,]       15.0
[32,]       21.4
```

1d) Compute beta coefficients

```
              [,1]   [,2]   [,3]    [,4]   [,5]   [,6]
count         1.00  1.000   1.00   1.000   1.00   1.00
mtcars.cyl    6.00  6.000   4.00   6.000   8.00   6.00
mtcars.disp 160.00 160.000 108.00 258.000 360.00 225.00
mtcars.hp   110.00 110.000  93.00 110.000 175.00 105.00
mtcars.wt     2.62  2.875   2.32   3.215   3.44   3.46
mtcars.carb   4.00  4.000   1.00   1.000   2.00   1.00
              [,7]   [,8]   [,9]  [,10]  [,11]  [,12]
count         1.00   1.00   1.00   1.00   1.00   1.00
mtcars.cyl    8.00   4.00   4.00   6.00   6.00   8.00
mtcars.disp 360.00 146.70 140.80 167.60 167.60 275.80
mtcars.hp   245.00  62.00  95.00 123.00 123.00 180.00
mtcars.wt     3.57   3.19   3.15   3.44   3.44   4.07
mtcars.carb   4.00   2.00   2.00   4.00   4.00   3.00
             [,13]  [,14]  [,15]   [,16]   [,17] [,18]
count         1.00   1.00   1.00   1.000   1.000   1.0
mtcars.cyl    8.00   8.00   8.00   8.000   8.000   4.0
mtcars.disp 275.80 275.80 472.00 460.000 440.000  78.7
mtcars.hp   180.00 180.00 205.00 215.000 230.000  66.0
mtcars.wt     3.73   3.78   5.25   5.424   5.345   2.2
mtcars.carb   3.00   3.00   4.00   4.000   4.000   1.0
             [,19]  [,20]   [,21]  [,22]   [,23]  [,24]
count        1.000  1.000   1.000   1.00   1.000   1.00
mtcars.cyl   4.000  4.000   4.000   8.00   8.000   8.00
mtcars.disp 75.700 71.100 120.100 318.00 304.000 350.00
mtcars.hp   52.000 65.000  97.000 150.00 150.000 245.00
mtcars.wt    1.615  1.835   2.465   3.52   3.435   3.84
mtcars.carb  2.000  1.000   1.000   2.00   2.000   4.00
              [,25]  [,26]  [,27]   [,28]  [,29]  [,30]
count         1.000  1.000   1.00   1.000   1.00   1.00
mtcars.cyl    8.000  4.000   4.00   4.000   8.00   6.00
mtcars.disp 400.000 79.000 120.30  95.100 351.00 145.00
mtcars.hp   175.000 66.000  91.00 113.000 264.00 175.00
mtcars.wt     3.845  1.935   2.14   1.513   3.17   2.77
mtcars.carb   2.000  1.000   2.00   2.000   4.00   6.00
             [,31]  [,32]
count         1.00   1.00
mtcars.cyl    8.00   4.00
mtcars.disp 301.00 121.00
mtcars.hp   335.00 109.00
mtcars.wt     3.57   2.78
mtcars.carb   8.00   2.00
```

1e) Comparing manual calculation to output of function

```
Call:
lm(formula = mpg ~ cyl + disp + hp + wt + carb, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.0635 -1.4580 -0.4306  1.2927  5.8244 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 40.815359   3.025568  13.490    3e-13 ***
cyl         -1.291899   0.679227  -1.902  0.06830 .  
disp         0.011486   0.015375   0.747  0.46175    
hp          -0.020353   0.020062  -1.015  0.31968    
wt          -3.846949   1.192155  -3.227  0.00337 ** 
carb        -0.006747   0.574269  -0.012  0.99072    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.56 on 26 degrees of freedom
Multiple R-squared:  0.8486,	Adjusted R-squared:  0.8195 
F-statistic: 29.15 on 5 and 26 DF,  p-value: 7.056e-10

       mtcars.mpg
[1,] 40.815359236
[2,] -1.291898563
[3,]  0.011485584
[4,] -0.020352893
[5,] -3.846949031
[6,] -0.006746893
```

When looking at the manual calculations I did compared to the ones that was outputted to the function, the results that were returned were exact same numbers in which confirms that the result I hand calculated was the same as what the function had outputted. I will not consider rounding because that is something very meniscal to the idea of what the problem had asked because the results are vastly similar.

2a) Running regression model

Training MEDV Adj R^2: .7491.

Root MSE	4.66119	R-Square	0.7571
Dependent Mean	22.30737	Adj R-Sq	0.7491
Coeff Var	20.89531		

RSME of both Training and Test sets:

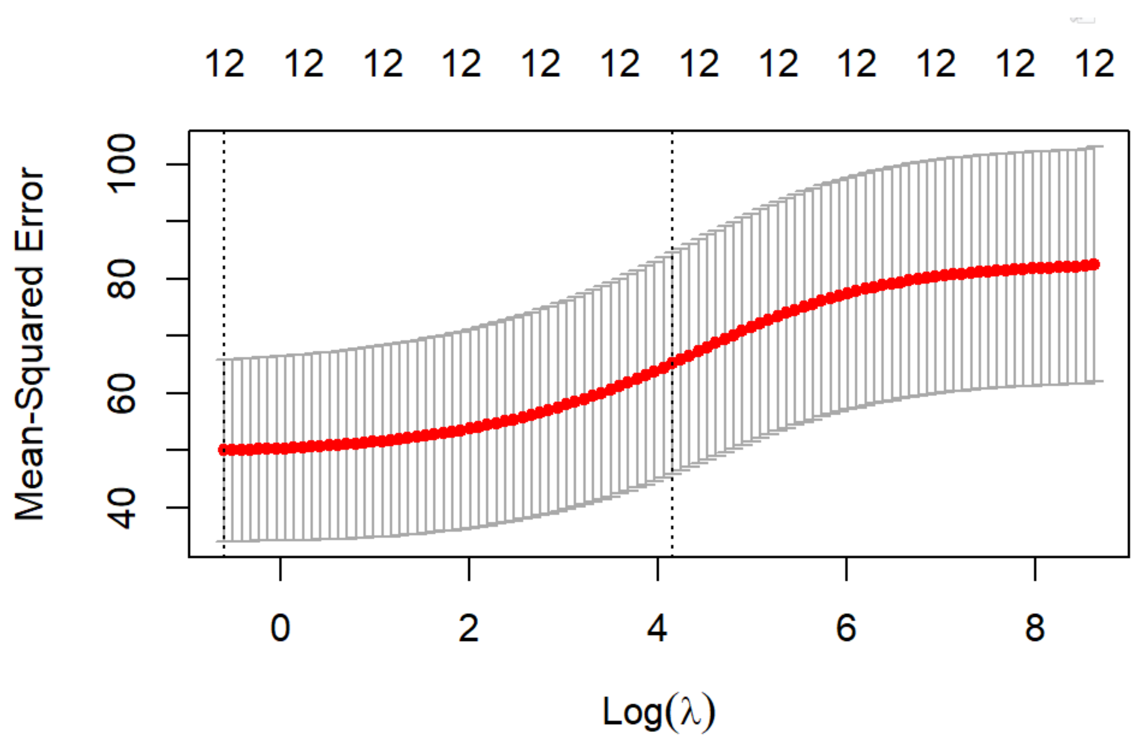
```
Residual standard error: 4.661 on 367 degrees of freedom
Multiple R-squared:  0.7571,    Adjusted R-squared:  0.7491
F-statistic: 95.31 on 12 and 367 DF,  p-value: < 2.2e-16

[1] 4.580769
[1] 5.263608
```

Housing Training RMSE: 4.580769

Housing Testing RMSE: 5.263608

I believe that there is no evidence of overfitting. Between both training and testing RMSE's there is very little difference between the two values.

2b)



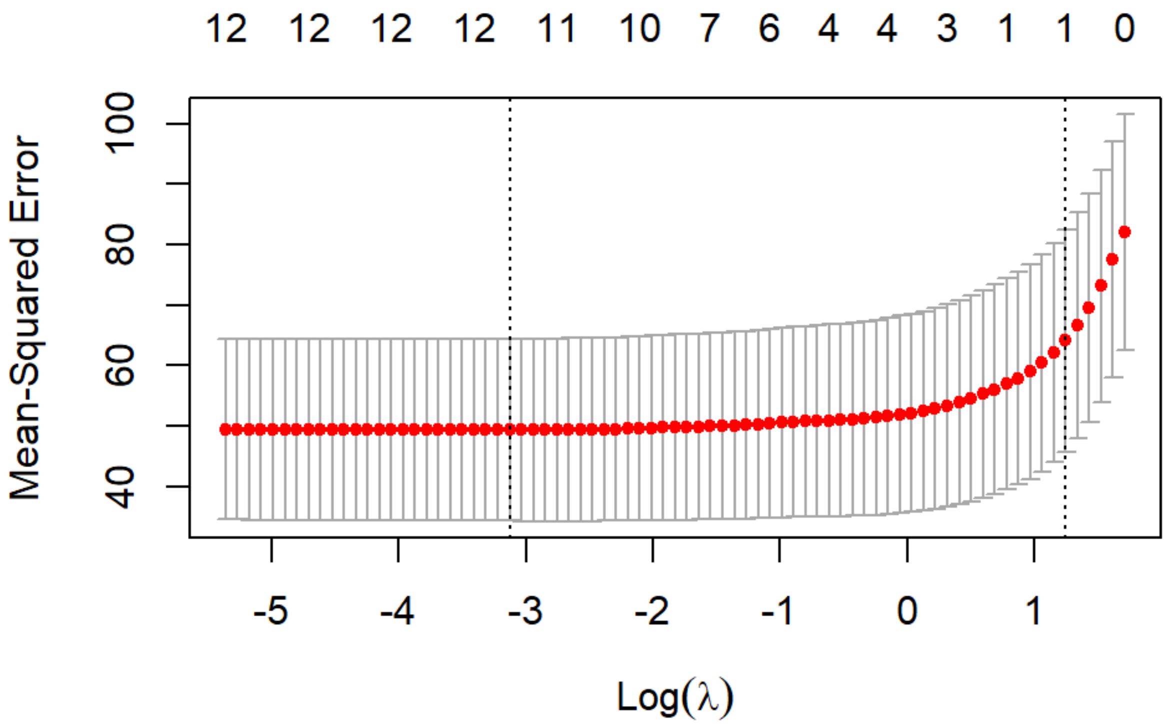
When looking at the graph there is no obvious dip in this graph so finding a minimum lambda might not exist.

2c)

```
[1] 63.53669
```

The dev value used to be 75 and now is 20.61% that would mean I am losing coefficients and the R-value was lost so it may not have regularized correctly.

2d)
```
  Df  %Dev Lambda
1 12 20.61  69.73
```



The Lasso is doing better cause of the dev percentage difference is less than the ridge.

2e)
```
  Df  %Dev Lambda
1  1 19.48  3.809
```

There is one variable selected in the lambda.1se under Df. The variance after the OLS went down even more than the Lasso or Ridge to 19.48 from the initial 75%.

2f) OLS, Ridge, Lasso

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.965e+01  5.610e+00   7.069 7.94e-12 ***
CRIM        -1.299e-01  3.412e-02  -3.807 0.000165 ***
ZN           4.341e-02  1.570e-02   2.764 0.005994 ** 
INDUS        6.302e-03  6.958e-02   0.091 0.927884    
CHAS         3.594e+00  9.454e-01   3.802 0.000168 ***
NOX         -2.197e+01  4.377e+00  -5.021 8.05e-07 ***
RM           4.229e+00  4.898e-01   8.634  < 2e-16 ***
AGE         -1.268e-04  1.511e-02  -0.008 0.993307    
DIS         -1.529e+00  2.318e-01  -6.598 1.46e-10 ***
RAD          2.665e-01  7.341e-02   3.630 0.000324 ***
TAX         -1.134e-02  4.130e-03  -2.746 0.006338 ** 
PTRATIO     -9.828e-01  1.506e-01  -6.526 2.24e-10 ***
LSTAT       -4.665e-01  6.094e-02  -7.655 1.73e-13 ***
---
Signif. codes:  
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.661 on 367 degrees of freedom
Multiple R-squared:  0.7571,    Adjusted R-squared:  0.7491 
F-statistic: 95.31 on 12 and 367 DF,  p-value: < 2.2e-16
```

```
13 x 1 sparse Matrix of class "dgCMatrix"
                      s1
(Intercept) -0.122160867
ZN          -0.002991104
INDUS        0.033262853
CHAS        -0.195382591
NOX          2.282376267
RM          -0.120336124
AGE          0.006961588
DIS         -0.121059676
RAD          0.056792060
TAX          0.002498431
PTRATIO      0.083324303
LSTAT        0.036690422
MEDV        -0.027024183
13 x 1 sparse Matrix of class "dgCMatrix"
                   s1
(Intercept) 1.8617434
ZN          .
INDUS       .
CHAS        .
NOX         .
RM          .
AGE         .
DIS         .
RAD         0.1964528
TAX         .
PTRATIO     .
LSTAT       .
MEDV        .
```

When looking at all the different datasets OLS and Ridge are considerably close in coefficients. Lasso only has 1 coefficient and that value compared to the other two are not very close.

2g) If I had to pick a model to use out of these, I would use the OLS model because the values would already be checked for overfitting. Each model after Ridge and Lasso we are losing coefficients after each run and to me I would rather have the whole dataset and it already be checked for overfitting.

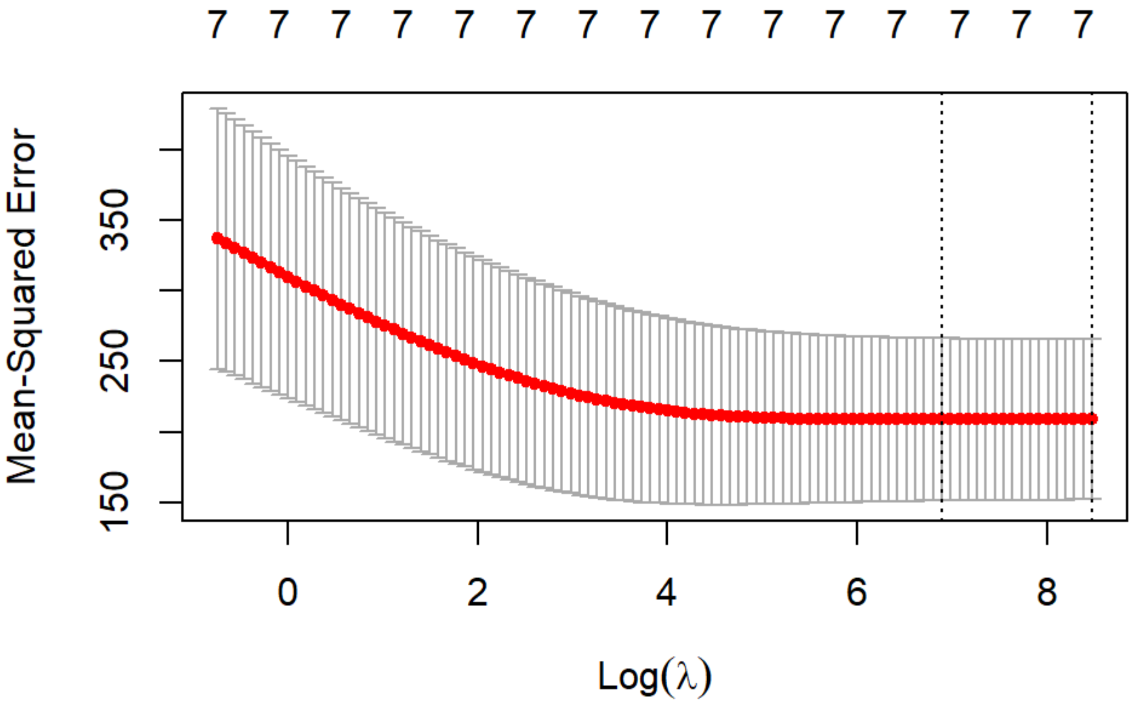
3a) Predicting the test set

```
Residual standard error: 1.303 on 17 degrees of freedom
Multiple R-squared:  0.9272,    Adjusted R-squared:  0.9058
F-statistic: 43.32 on 5 and 17 DF,  p-value: 4.387e-09

[1] 1.120282
[1] 4.260169
```

I feel there is overfitting because of the huge gap in RMSE from the training and test sets.

3b)

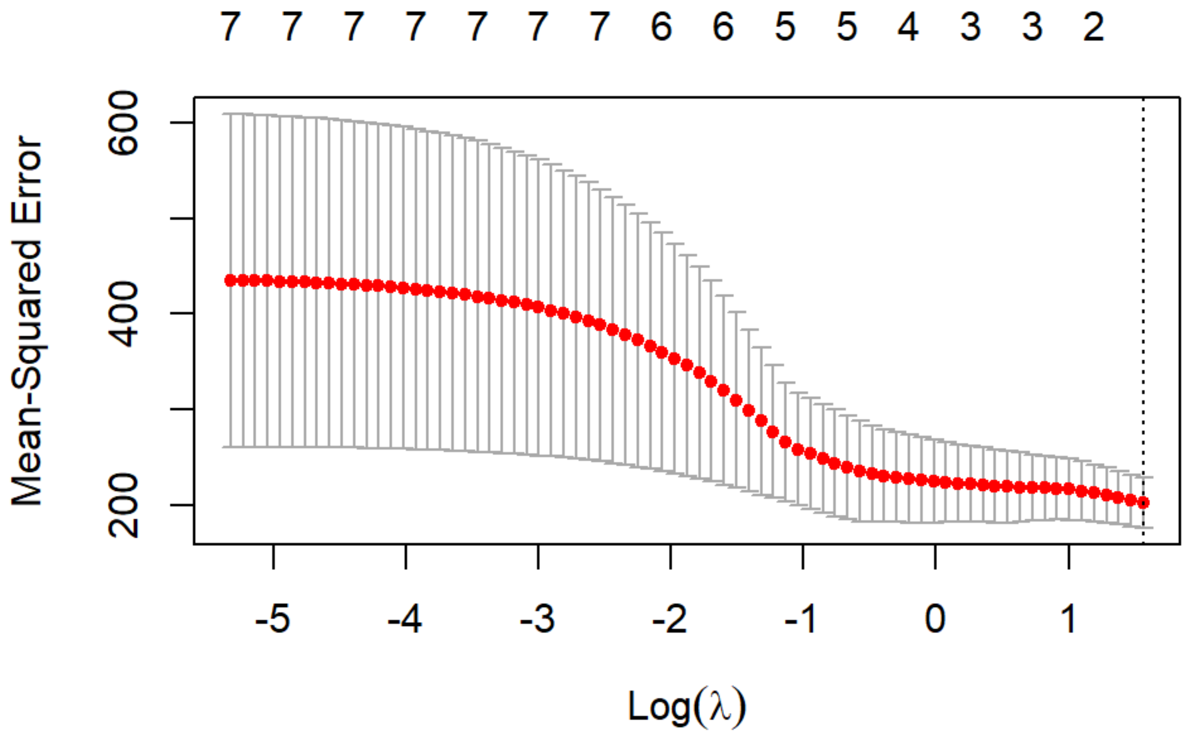
```
[1] 15.37505
```



That 15.37505 is the RMSE.

3c)

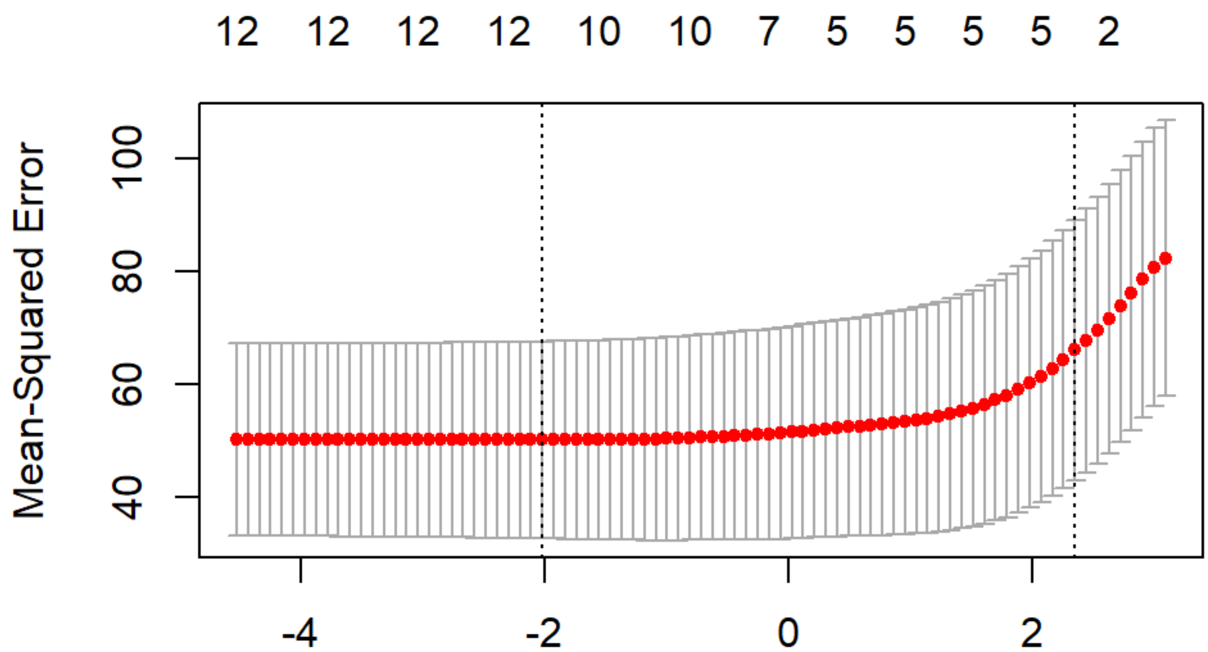
[1] 15.37505

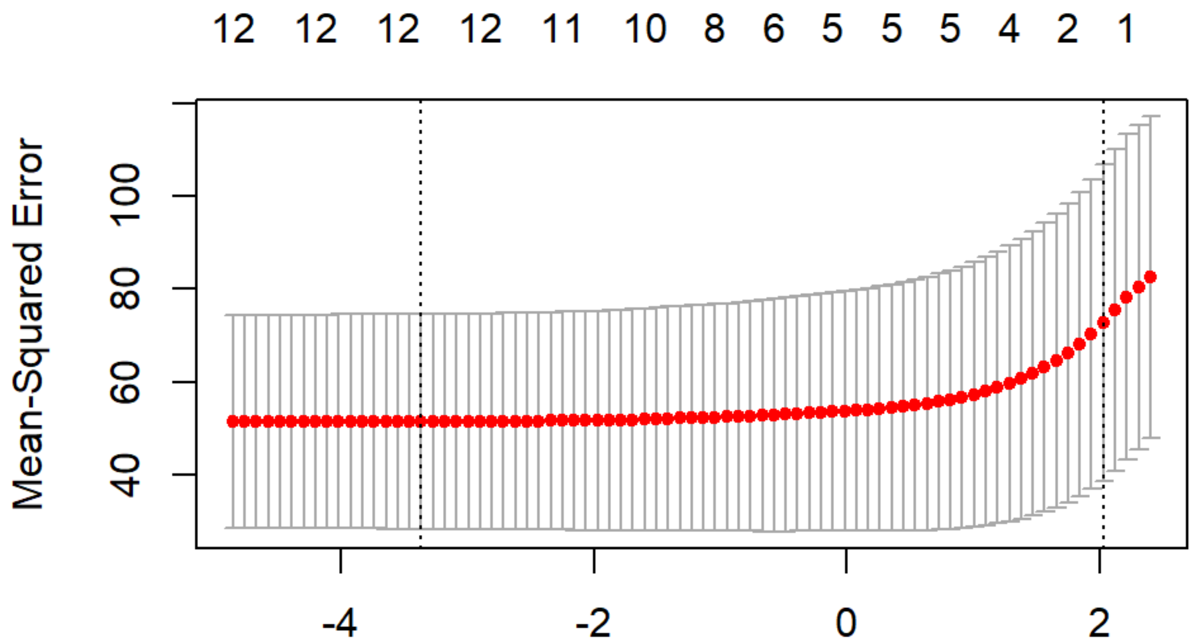


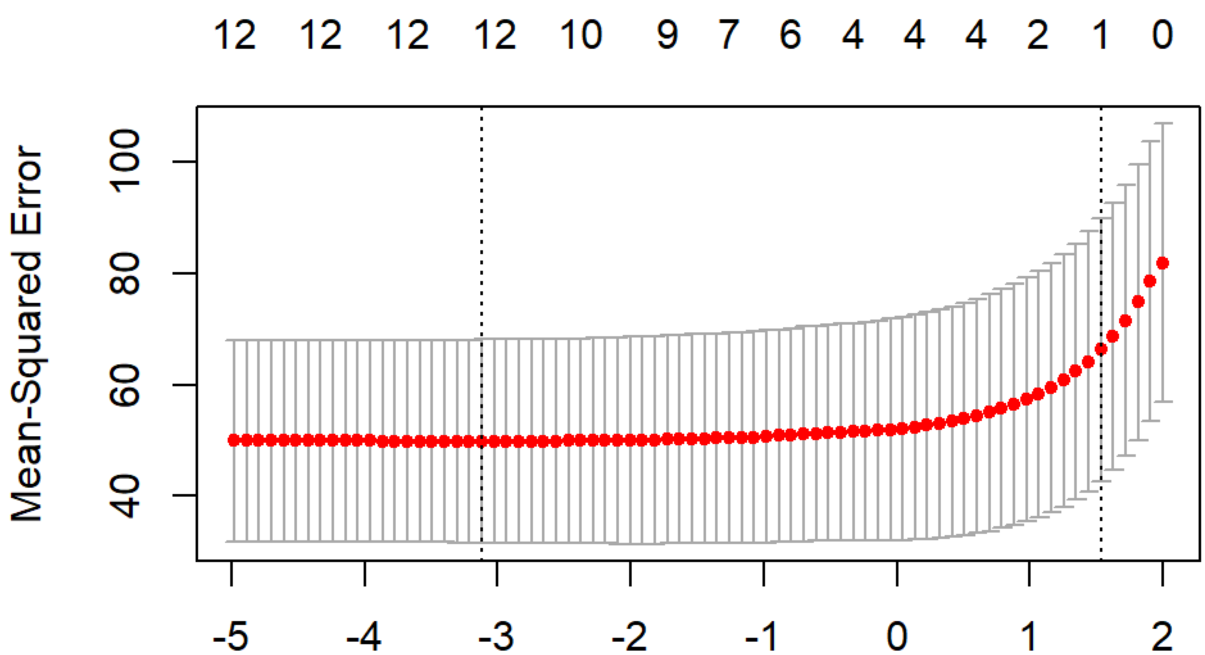
It is doing better because there is less lambda but the RMSE is the same value between both.

3d)

3e) The .5 RMSE was the closest RMSE to the test RMSE. All three were better than the ridge and lasso.







3f) When mixing both ridge and lasso its better than the extremes to have regularized seeing as though .5 was much closer to the test RMSE.