Assignment1_rspake1

Part A

Question 1

“What is the main purpose of regularization when training predictive models?”

- Regularization is used to combat overfitting in a model (when the model can’t make a generalization). This is done by shrinking parameters in the cost function.

Question 2

“What is the role of a loss function in a predictive model? And name two common loss functions for regression models and two common loss functions for classification models.”

- The role of the loss function is to penalize the generalization from the model for deviating from the actual. The larger the deviation, the larger the output of the loss function.
- Two common regression loss functions are MSE (Mean Square Error)(L2 Loss) and MAE (Mean Absolute Error)(L1 Loss).
- Two common classification loss functions are Binary Cross-Entropy(Log Loss) and Hinge Loss (Hinge also penalizes correct predictions that are not confident).

Question 3

" Consider the following scenario. You are building a classification model with many hyper parameters on a relatively small dataset. You will see that the training error is extremely small. Can you fully trust this model? Discuss the reason."

- I would not trust this model readily because with small datasets with many features (parameters) , we run a greater risk of overfitting as it is easier for the model to learn these specific data points in the training set than to make any generalizations.

Question 4

“What is the role of the lambda parameter in regularized linear models such as Lasso or Ridge regression models?”

- lambda is the regularization parameter that controls the two goals of fitting the trainign set well and keeping the parameters small.

Part B

Here we will load the Carseats dataset and prepare it to answer the questions.

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 4.1.2
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.1.2
```

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

```
## Loaded glmnet 4.1-3
```

```
library(caret)
```

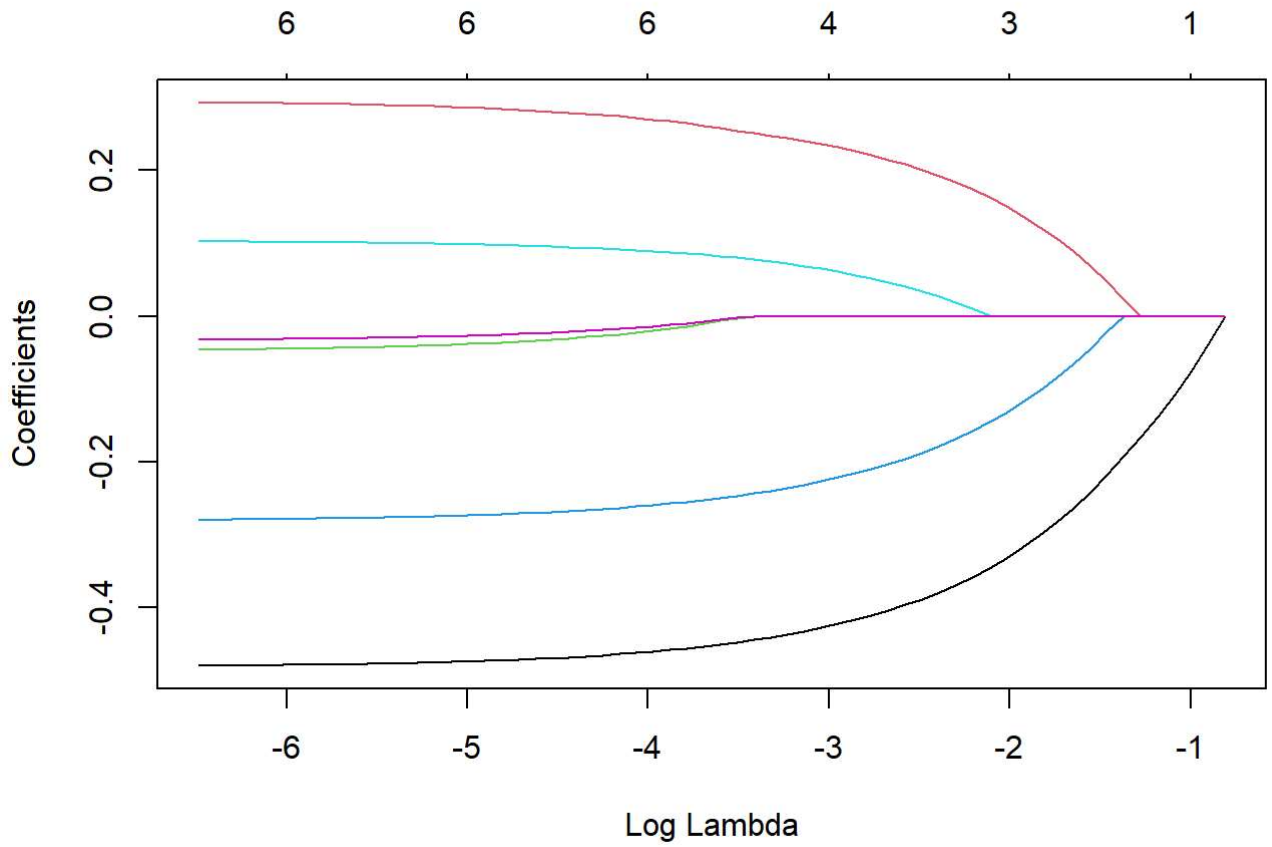
```
## Warning: package 'caret' was built under R version 4.1.2
```

```
## Loading required package: ggplot2
```

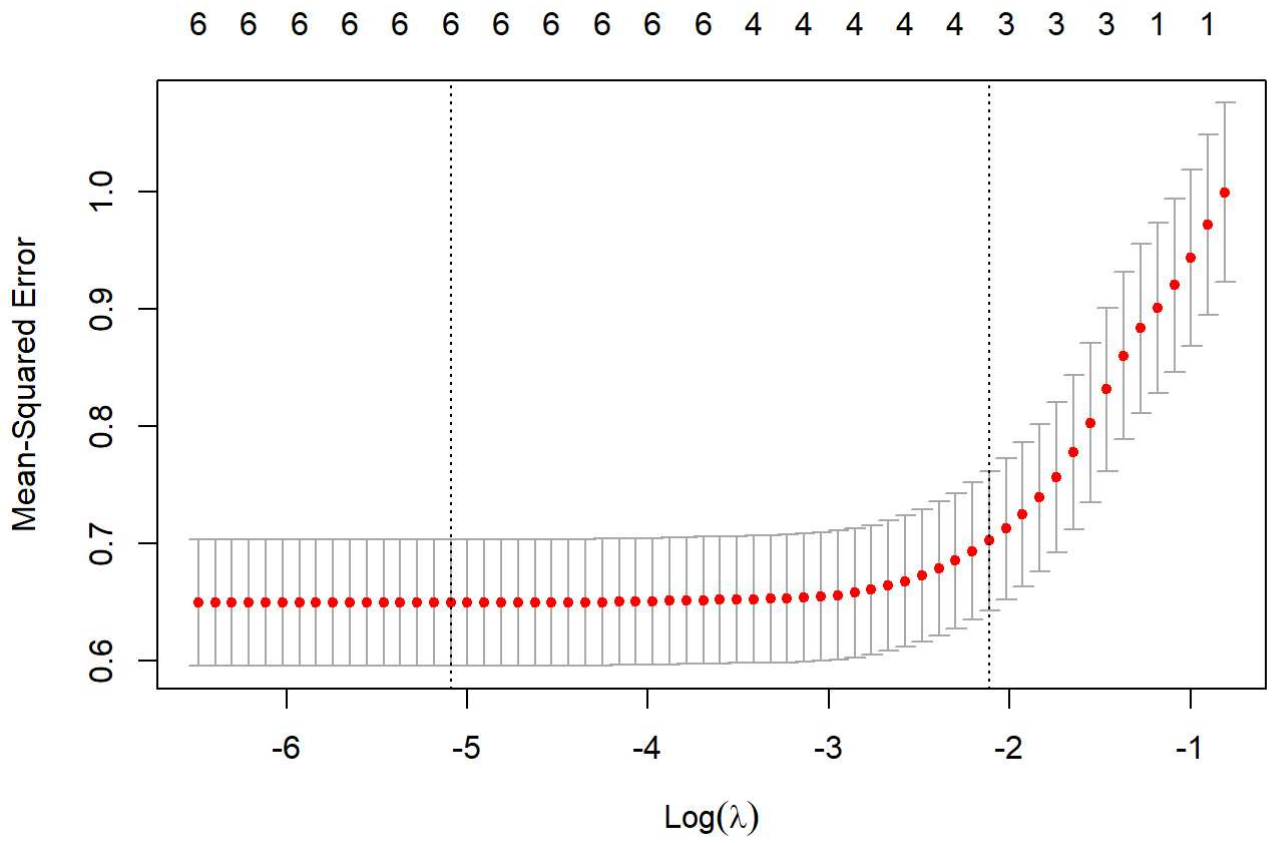
```
## Warning: package 'ggplot2' was built under R version 4.1.2
```

```
## Loading required package: lattice
```

```
Carseats_Filtered <- Carseats %>% select("Sales", "Price", "Advertising", "Population", "Age",
"Income", "Education")

Norm_Carseats_Filtered <- scale(Carseats_Filtered)

x <- as.matrix(Norm_Carseats_Filtered[, c('Price','Advertising','Population','Age','Income','Edu
cation')])

y <- Norm_Carseats_Filtered[,'Sales']

fit.lasso <- glmnet(x,y,alpha = 1)

plot(fit.lasso,xvar = "lambda")
```




```
plot(cv.glmnet(x,y,alpha = 1))
```



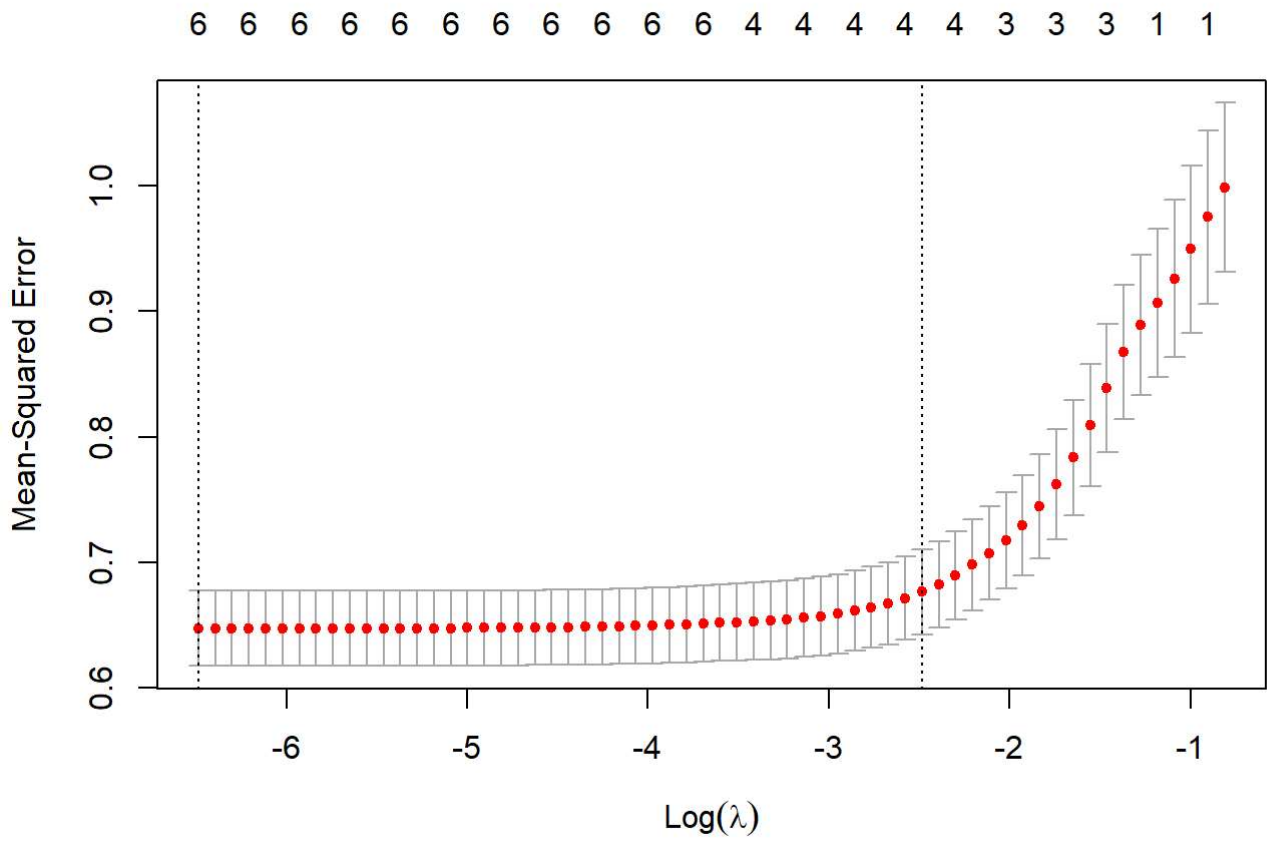
Above, we have called all of our libraries, filtered the dataset into the columns we desire, normalized the filtered data, then established x and y by seperating sales from the rest of the data.

Lastly we created a lasso regression with alpha set to 1 and printed the results.

Question 1

Here we will build a cv.glmnet to discover the best lambda value for this model.

```
cvfit <- cv.glmnet(x,y, alpha = 1)
plot(cvfit)
```




```
best_lambda <- cvfit$lambda.min
best_lambda
```

```
## [1] 0.001524481
```

As our cross-validation glmnet model shows, the optimal lambda for our lasso model is ~0.0015.

Question 2

```
coef(fit.lasso, s = best_lambda)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)  9.866665e-17
## Price       -4.793834e-01
## Advertising  2.932098e-01
## Population  -4.624934e-02
## Age         -2.792202e-01
## Income       1.024459e-01
## Education   -3.223128e-02
```

Here we are using the “best_lambda” that was established in the first question to find the coefficients of all of the attributes in our original fit.lasso model when this “best_lambda” is applied.

For “Price” (which has been normalized), we see that the coefficient is -4.79x10-1.

Question 3

```
coef(fit.lasso, s = 0.01)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)  9.798009e-17
## Price       -4.696889e-01
## Advertising  2.815718e-01
## Population  -3.323443e-02
## Age         -2.693300e-01
## Income       9.585212e-02
## Education   -2.330455e-02
```

```
coef(fit.lasso, s = 0.1)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)  9.803050e-17
## Price       -3.691394e-01
## Advertising  1.839178e-01
## Population   .
## Age         -1.684796e-01
## Income       1.925921e-02
## Education    .
```

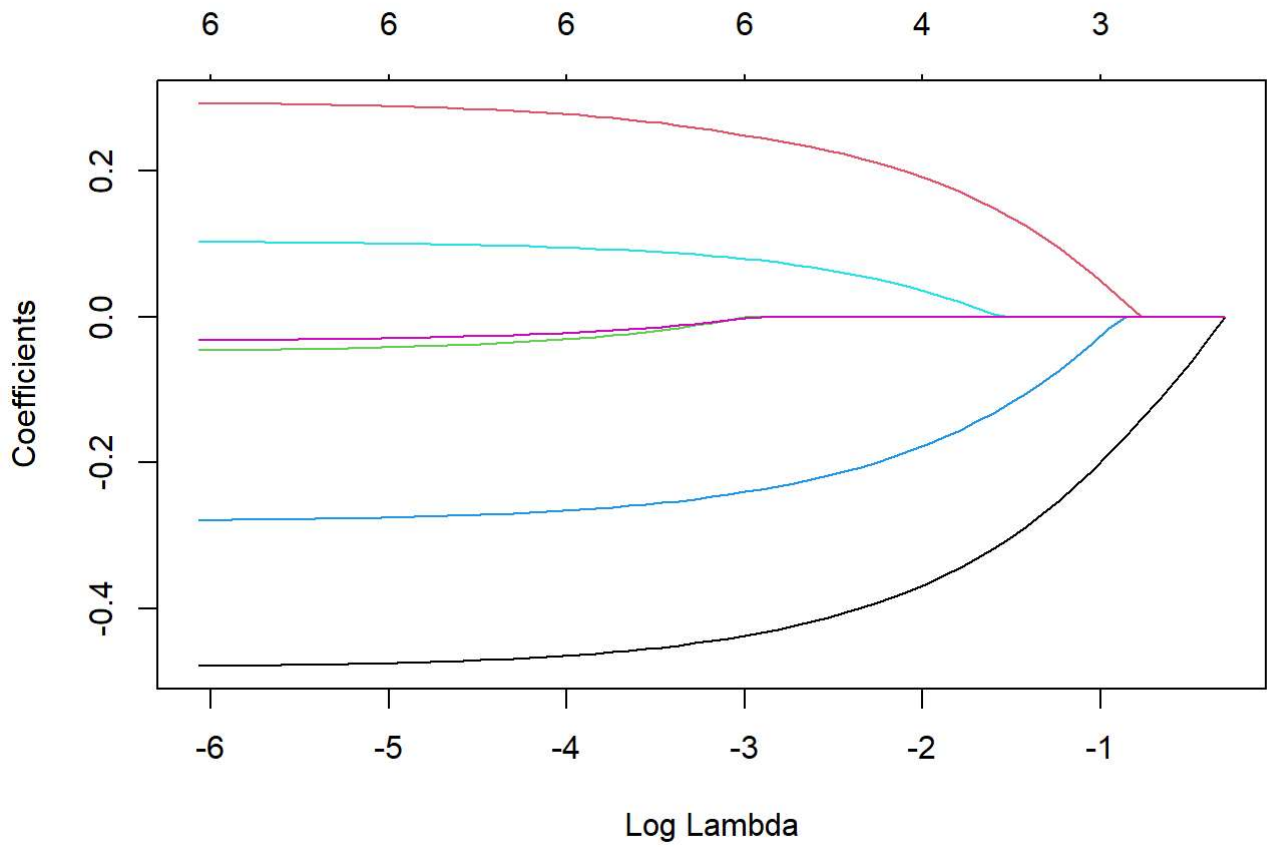
When we replace the lambda value in our model from best_lambda(~.0015) to 0.01 we do not lose any attributes along the way.

However, when we change to 0.1 we start to lose attributes such as Population and Education.

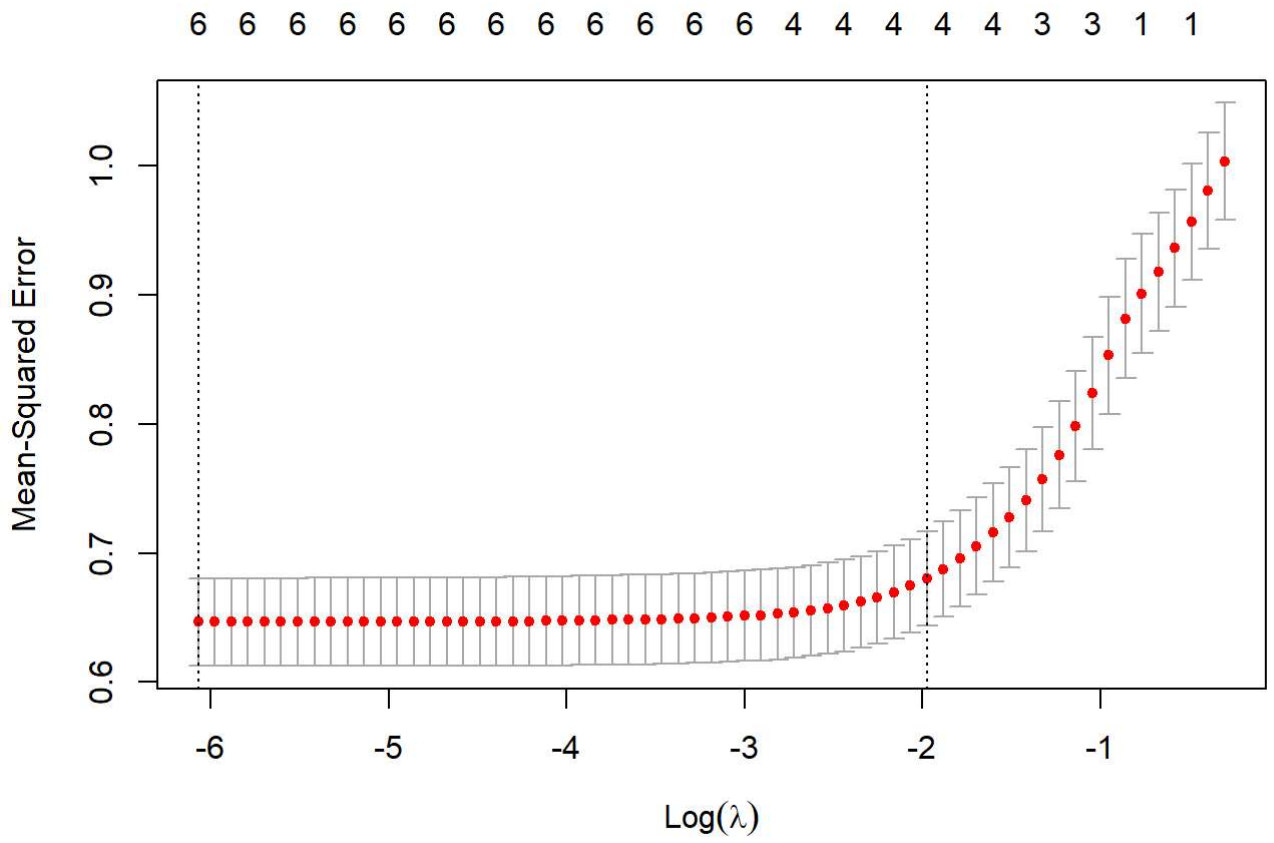
As we increase lambda, we are going to lose more attributes as the model drops them out just like we saw between 0.01 and 0.1.

Question 4

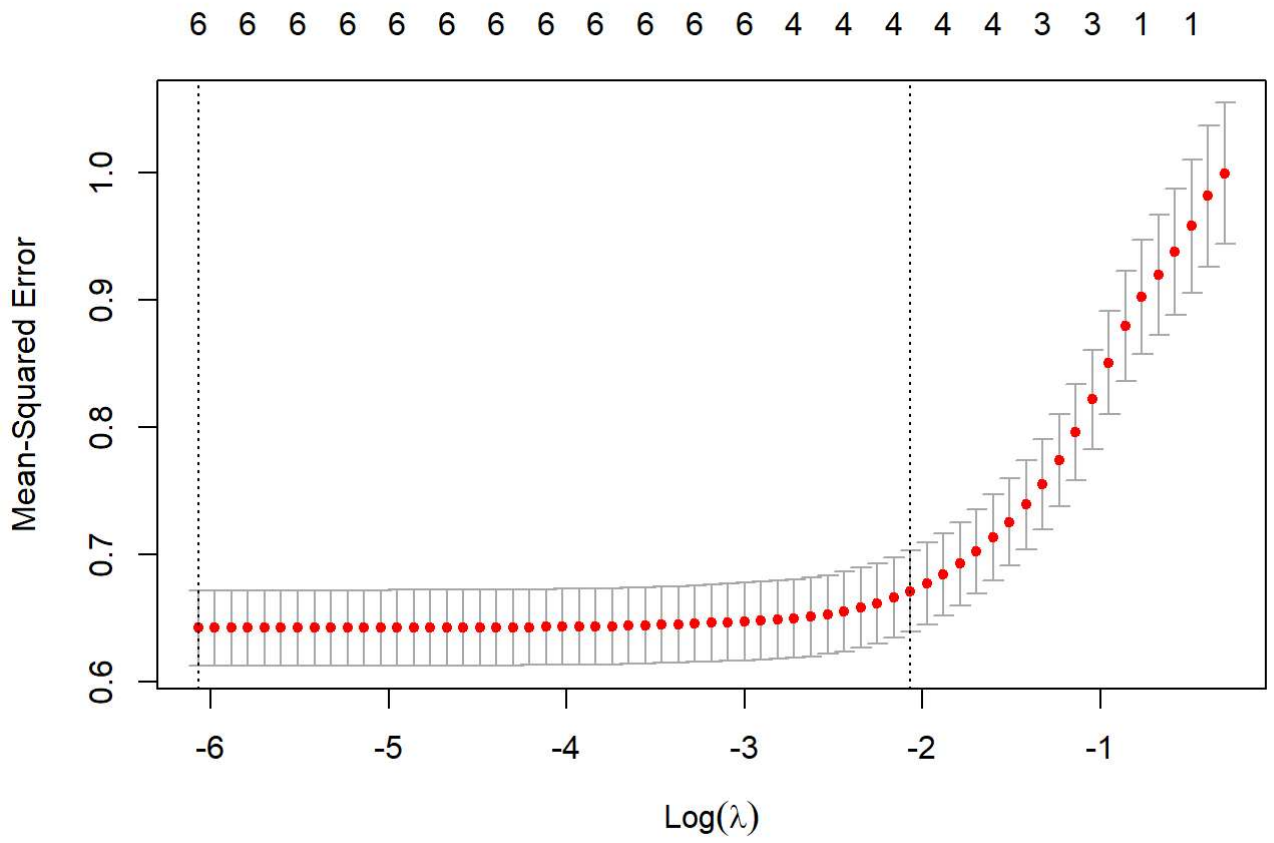
```
fit.elastic <- glmnet(x,y,alpha = 0.6)

plot(fit.elastic,xvar = "lambda")
```




```
plot(cv.glmnet(x,y,alpha = 0.6))
```




```
cvfit_elastic <- cv.glmnet(x,y, alpha = 0.6)
plot(cvfit_elastic)
```




```
best_lambda_elastic <- cvfit_elastic$lambda.min
best_lambda_elastic
```

```
## [1] 0.002315083
```

The best Lambda for an elastic model with alpha set to 0.6 is ~0.024.

(Graphs are supplied to compare between this and the original lasso model if you would like to see the difference.).