Modelling - Ridge & Lasso Regression

```
library(tidyverse) #loading in packages
library(glmnet)
std_data = read.csv("standardised_data_460.csv")
```

Ridge Regression

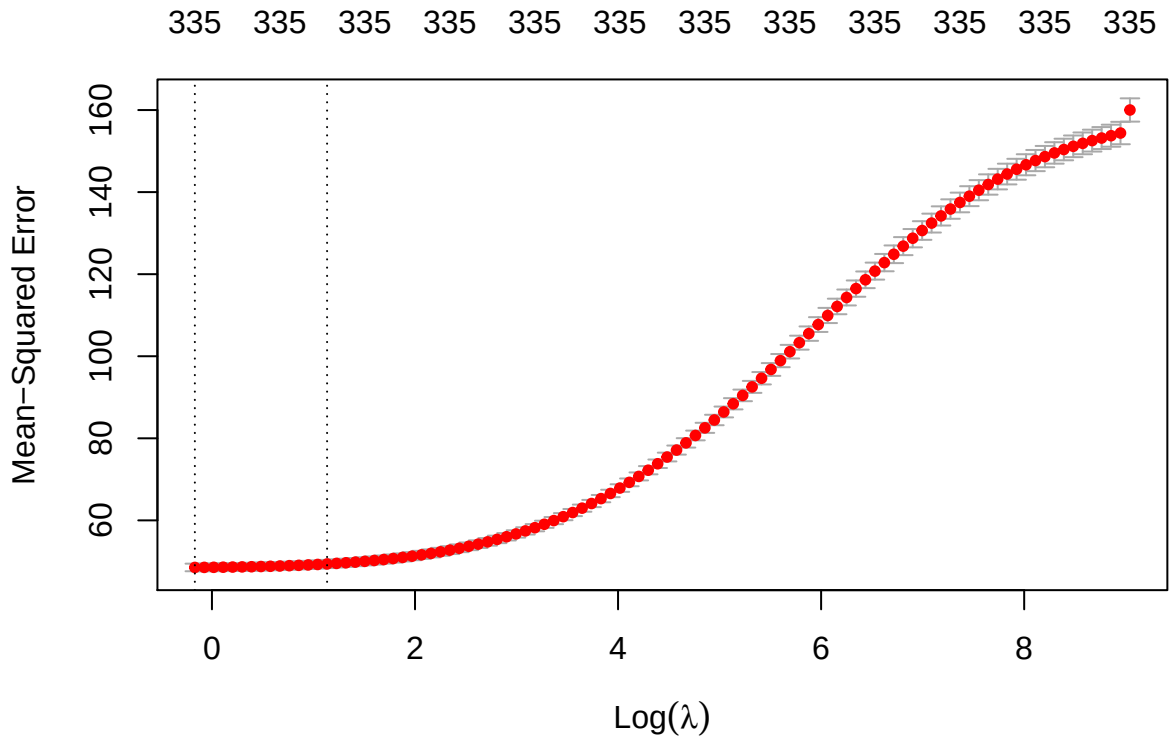
Making std_data_predictors store all attributes that could influence response.

```
std_data_predictors = data.matrix(select(std_data, -c("Response")))
```

Finding the lambda value that creates the lowest mean squared error using k fold cross validation.

```
kfolds_cv_model <-
  cv.glmnet(std_data_predictors, std_data$Response, alpha = 0)

plot(kfolds_cv_model)
```



```
bestLambdaVal = kfolds_cv_model$lambda.min
```

Using the best lambda value found, we create a ridge regression model utilising all predictor variables.

```
finalModel <- glmnet(std_data_predictors, std_data$Response, alpha = 0, lambda = bestLambdaVal)
```

Predict the response variable using the model for each set of attribute values, then calculate the R^2 value to see the percentage of the variance which is explained by the model.

```
predictedResponse <- predict(finalModel, s = bestLambdaVal, newx = std_data_predictors)

sse <- sum((predictedResponse - std_data$Response)^2)
sst <- sum((std_data$Response - mean(std_data$Response))^2)


rsq <- 1 - sse/sst
rsq
```

```
## [1] 0.7126104
```

Lasso Regression

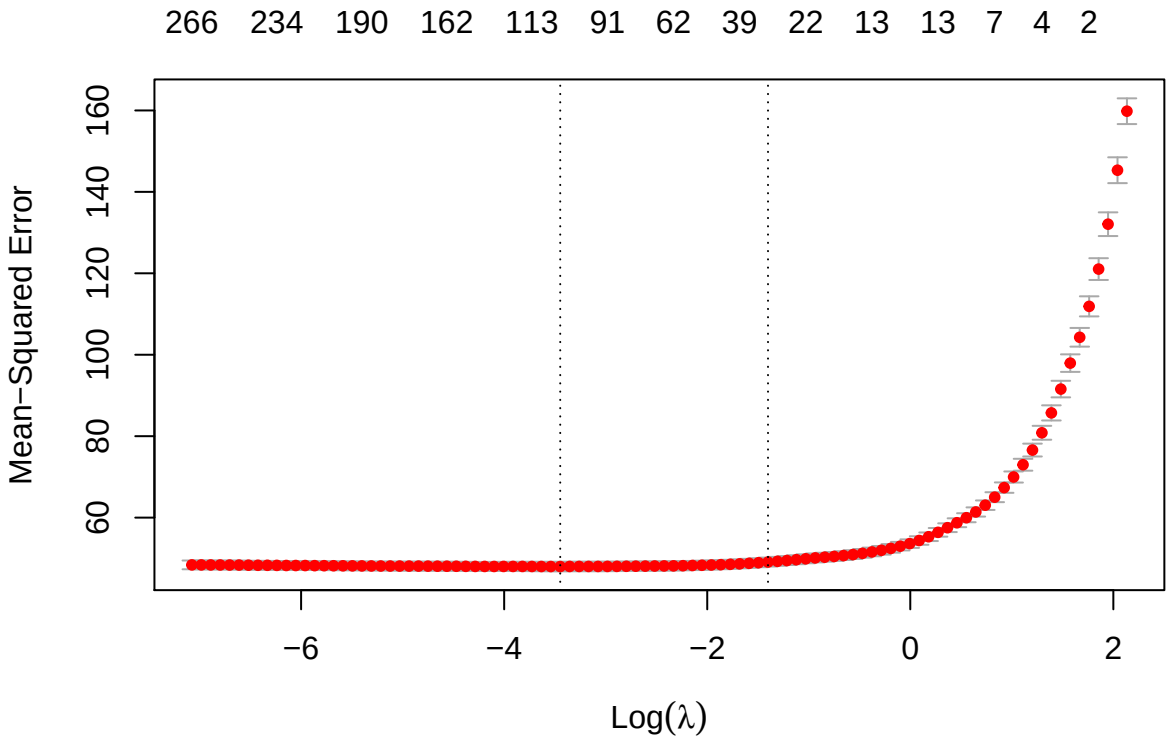
Making std_data_predictors store all attributes that could influence response.

```
std_data_predictors = data.matrix(select(std_data, -c("Response")))
```

Finding the lambda value that creates the lowest mean squared error using k fold cross validation.

```
kfolds_cv_model <-
  cv.glmnet(std_data_predictors, std_data$Response, alpha = 1)

plot(kfolds_cv_model)
```




```
bestLambdaVal = kfolds_cv_model$lambda.min
```

Using the best lambda value found, we create a lasso regression model utilising all predictor variables.

```
finalModel <- glmnet(std_data_predictors, std_data$Response, alpha = 1, lambda = bestLambdaVal)
```

Predict the response variable using the model for each set of attribute values, then calculate the R^2 value to see the percentage of the variance which is explained by the model.

```
predictedResponse <- predict(finalModel, s = bestLambdaVal, newx = std_data_predictors)

sse <- sum((predictedResponse - std_data$Response)^2)
sst <- sum((std_data$Response - mean(std_data$Response))^2)


rsq <- 1 - sse/sst
rsq
```

```
## [1] 0.7115071
```