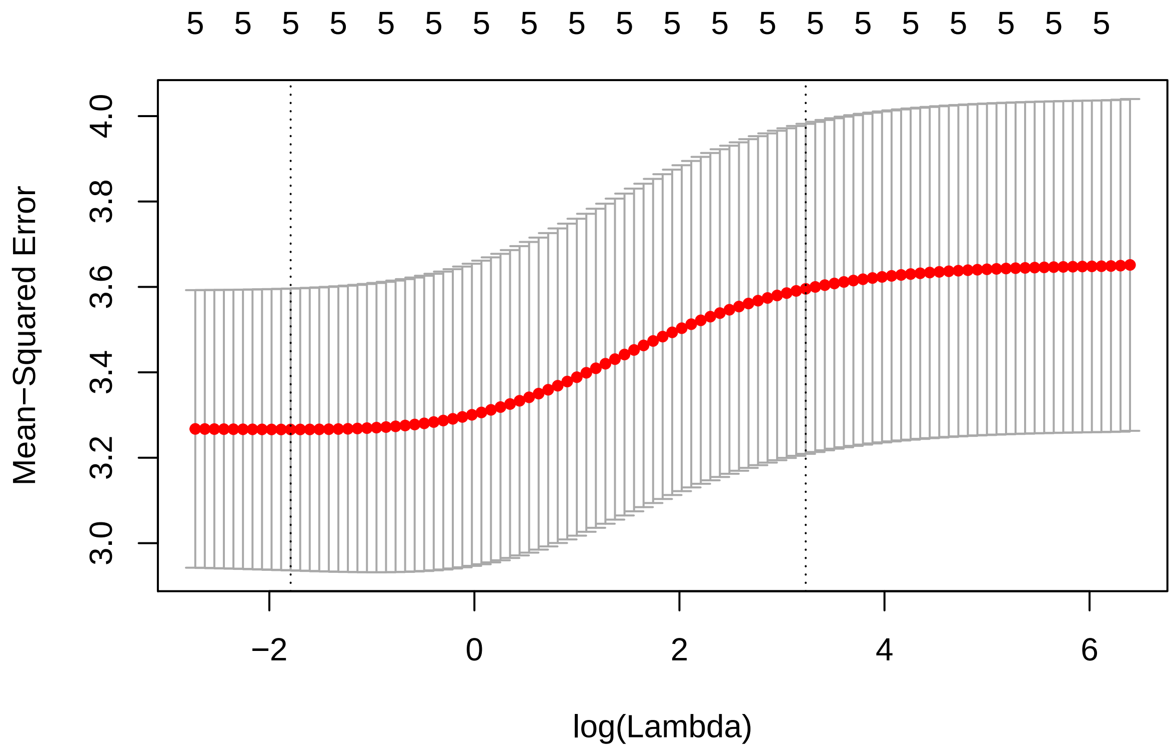
Part 2, Ridge, LASSO, and KRLS

```r
#Prepare the data
xTrain<-model.matrix(art~.,data=dataTrain)[,-1]
yTrain<-dataTrain$art
yTrain2 <- log(yTrain+1)

#"Note also that the results of cv.glmnet are random, since the folds are selected at random. Users can
#?glmnet, standardize: "Logical flag for x variable standardization, prior to fitting the model sequenc
```

```r
#Ridge

#Run cross validation
cvRidge<-cv.glmnet(xTrain,yTrain,alpha=0)
plot(cvRidge)
```




```r
#See which lambda minimizes the MSE and the corresponding coefficient estimates
lambdaOptRidge<-cvRidge$lambda.min #Or, use lambda.1se
print(lambdaOptRidge)
```

```
## [1] 0.1666041
```

```r
coef(cvRidge,s=lambdaOptRidge) #Alternatively, use s="lambda.min"
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  1.07680275
## femWomen    -0.33287310
## marMarried   0.35779733
## kid5        -0.32113915
## phd          0.05120413
```

```
## ment          0.05425488
```

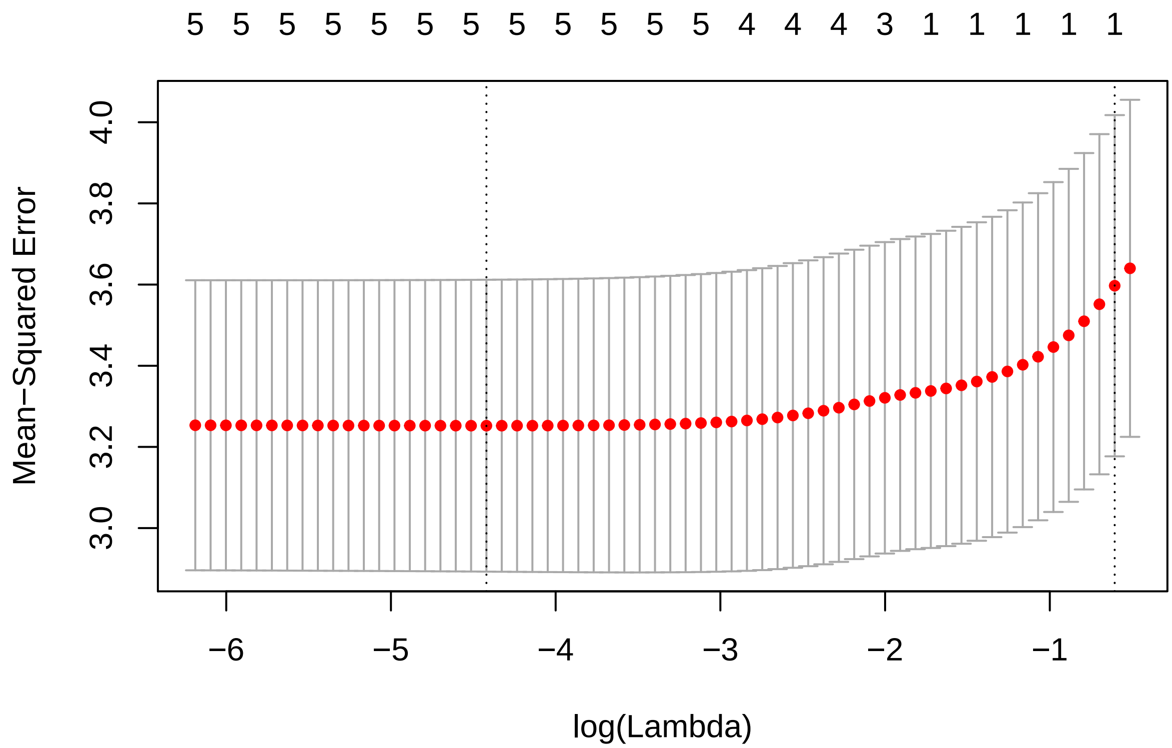
```r
#Get y hat for training data
modRidge<-cvRidge$glmnet.fit
yHatRidge<-predict(modRidge,s=lambdaOptRidge,newx=xTrain)

#Get MSE on training data
MSERidge<-mean((yTrain-yHatRidge)^2) #3.194847
print(MSERidge)
```

```
## [1] 3.194847
```

```r
#LASSO

#Run cross validation
cvLASSO<-cv.glmnet(xTrain,yTrain,alpha=1)
plot(cvLASSO)
```




```r
#See which lambda minimizes the MSE and the corresponding coefficient estimates
lambdaOptLASSO<-cvLASSO$lambda.min
print(lambdaOptLASSO)
```

```
## [1] 0.01203016
```

```r
coef(cvLASSO,s=lambdaOptLASSO)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  1.09871179
## femWomen    -0.33978264
## marMarried   0.37154609
## kid5        -0.34149612
## phd          0.03422224
```

```
## ment         0.05823863
```

```
#Get y hat for training data
modLASSO<-cvLASSO$glmnet.fit
yHatLASSO<-predict(modLASSO,s=lambdaOptLASSO,newx=xTrain)

#Get MSE on training data
MSELASSO<-mean((yTrain-yHatLASSO)^2) #3.191976
print(MSELASSO)
```

```
## [1] 3.191976
```

```
#Kernel Regularized Least Squares
modKRLS<-krls(X=xTrain,y=yTrain)
```

```
## Warning in Eigenobject$values + lambda: Recycling array of length 1 in vector-array arithmetic is dep
##   Use c() or as.vector() instead.

## Warning in Eigenobject$values + lambda: Recycling array of length 1 in vector-array arithmetic is dep
##   Use c() or as.vector() instead.

##
##  Average Marginal Effects:
##
##    femWomen  marMarried        kid5         phd        ment
## -0.27754699  0.18064139 -0.14878876  0.05947991  0.04484820
##
##  Quartiles of Marginal Effects:
##
##       femWomen marMarried        kid5         phd       ment
## 25% -0.4135114  0.0721593 -0.23964663 -0.07814581 0.03264739
## 50% -0.2877684  0.1659567 -0.08054785  0.08173500 0.04805456
## 75% -0.1150186  0.2832102 -0.02059481  0.20216908 0.06093803
```

```
summary(modKRLS)
```

```
## * *********************** *
## Model Summary:
##
## R2: 0.1633309
##
## Average Marginal Effects:
##                     Est  Std. Error   t value     Pr(>|t|)
## femWomen*   -0.27754699 0.175733340 -1.579364 1.147709e-01
## marMarried*  0.18064139 0.170258653  1.060982 2.891179e-01
## kid5        -0.14878876 0.065360527 -2.276431 2.316583e-02
## phd          0.05947991 0.053124289  1.119637 2.633092e-01
## ment         0.04484820 0.007611001  5.892549 6.291628e-09
##
## (*) average dy/dx is for discrete change of dummy variable from min to max (i.e. usually 0 to 1))
##
##
## Quartiles of Marginal Effects:
##                     25%         50%         75%
## femWomen*   -0.41351140 -0.28776839 -0.11501859
## marMarried*  0.07215930  0.16595674  0.28321020
## kid5        -0.23964663 -0.08054785 -0.02059481
```

```
## phd         -0.07814581  0.08173500  0.20216908
## ment         0.03264739  0.04805456  0.06093803
##
## (*) quantiles of dy/dx is for discrete change of dummy variable from min to max (i.e. usually 0 to 1)
```

```
yHatKRLS<-predict(modKRLS,newdata=xTrain)

#Get MSE on training data
MSEKRLS<-mean((yTrain-yHatKRLS$newdataK)^2) #3.191976
print(MSEKRLS)
```

```
## [1] 5.647153
```

We can see that being female and having additional kids under the age of 5 have a negative average marginal effect on the PhD student's number of publications, which is what we had expected.

To do:

Ridge and LASSO seem to give very similar results to each other. However, we will need to think about how to compare ridge, LASSO, and KRLS to our GLM methods from Part 1.

TESTING RESULTS:

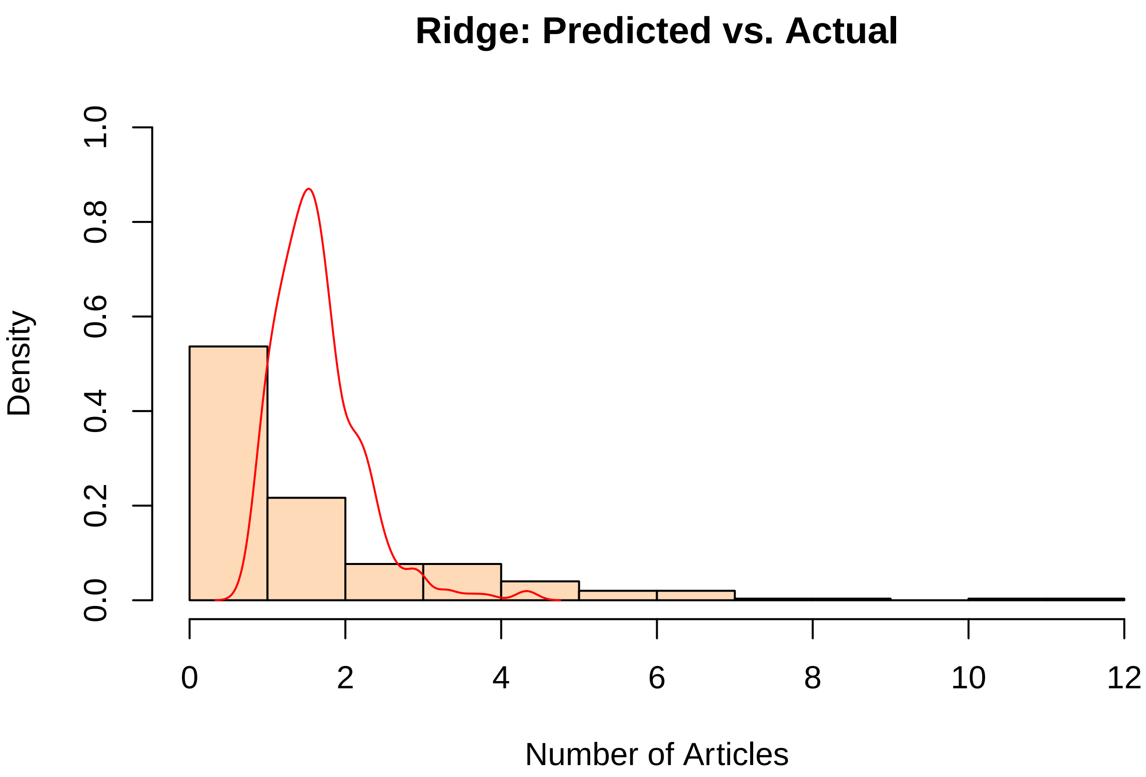
```
yTest <- dataTest[,1]
xTest2 <-model.matrix(art~.,data=dataTest)[,-1]
```

RIDGE TEST MLE

```
yHatTestRidge<-predict(modRidge,s=lambdaOptRidge,newx=xTest2)
MSETestRidge<-mean((yTest-yHatTestRidge)^2)
print(sqrt(MSETestRidge))
```

```
## [1] 1.881557
```

```
hist(yTest,freq = F, ylim = c(0,1),col="peachpuff", main = "Ridge: Predicted vs. Actual", xlab = "Number
lines(density(yHatTestRidge), col = "red")
```

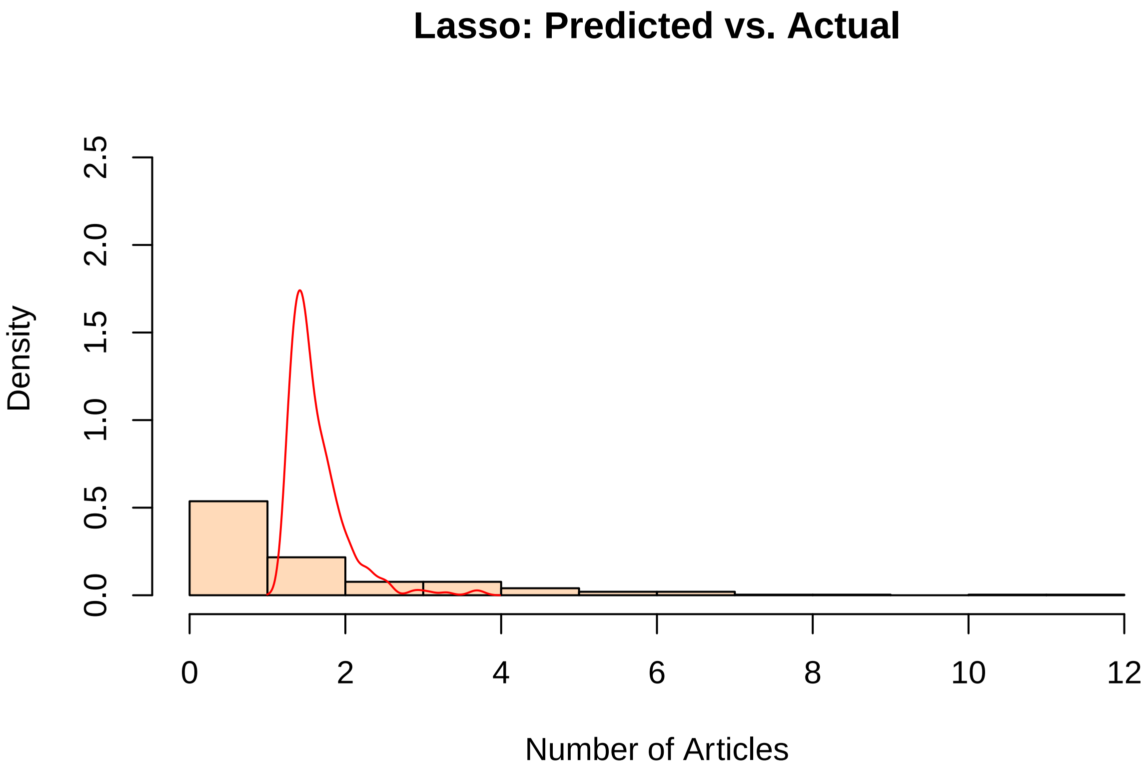


LASSO TEST MLE

```
yHatTestLasso<-predict(modLASSO,s=lambdaOptRidge,newx=xTest2)
MSETestLasso<-mean((yTest-yHatTestLasso)^2)
print(sqrt(MSETestLasso))
```

```
## [1] 1.88683
```

```
hist(yTest,freq = F, ylim = c(0,2.7),col="peachpuff", main = "Lasso: Predicted vs. Actual", xlab = "Numl
lines(density(yHatTestLasso), col = "red")
```

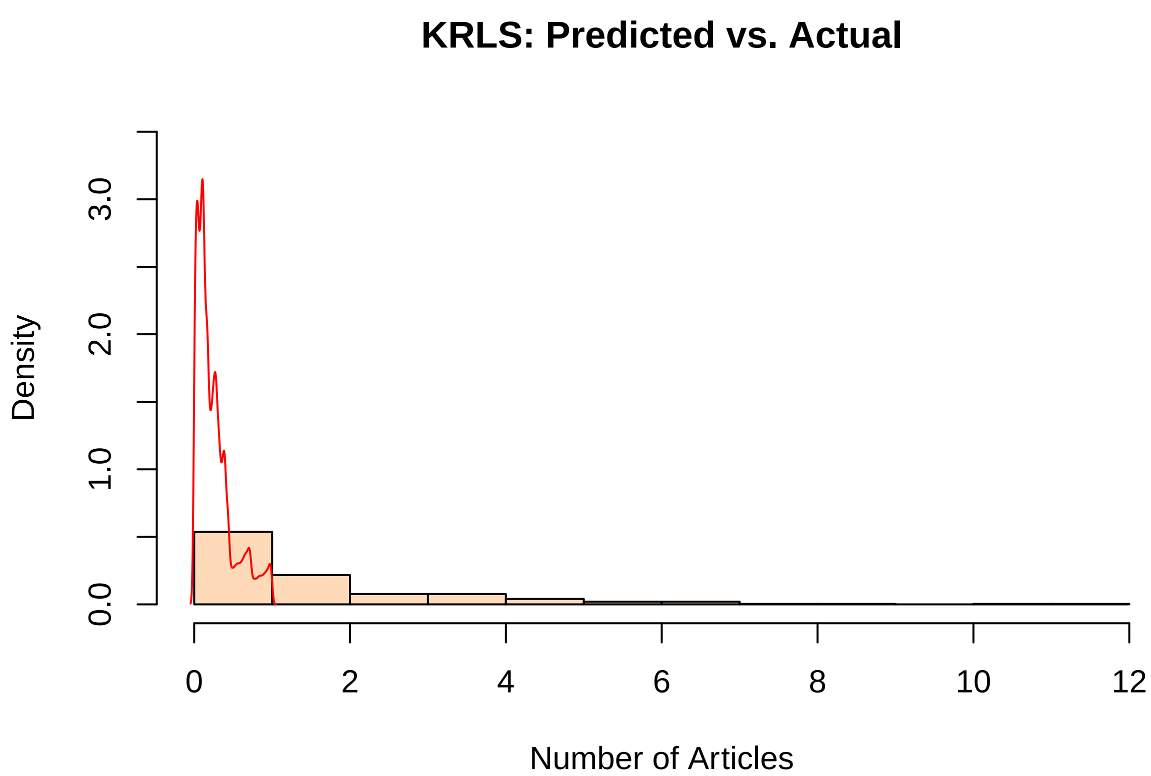


KRLS TEST MLE

```
yHatTestKRLS<-predict(modKRLS,newdata=xTest2)$newdataK
MSETestKRLS<-mean((yTest-yHatTestKRLS)^2)
print(sqrt(MSETestKRLS))
```

```
## [1] 2.516787
```

```
hist(yTest,freq = F, ylim = c(0,3.5),col="peachpuff", main = "KRLS: Predicted vs. Actual", xlab = "Numbe
lines(density(yHatTestKRLS), col = "red")
```

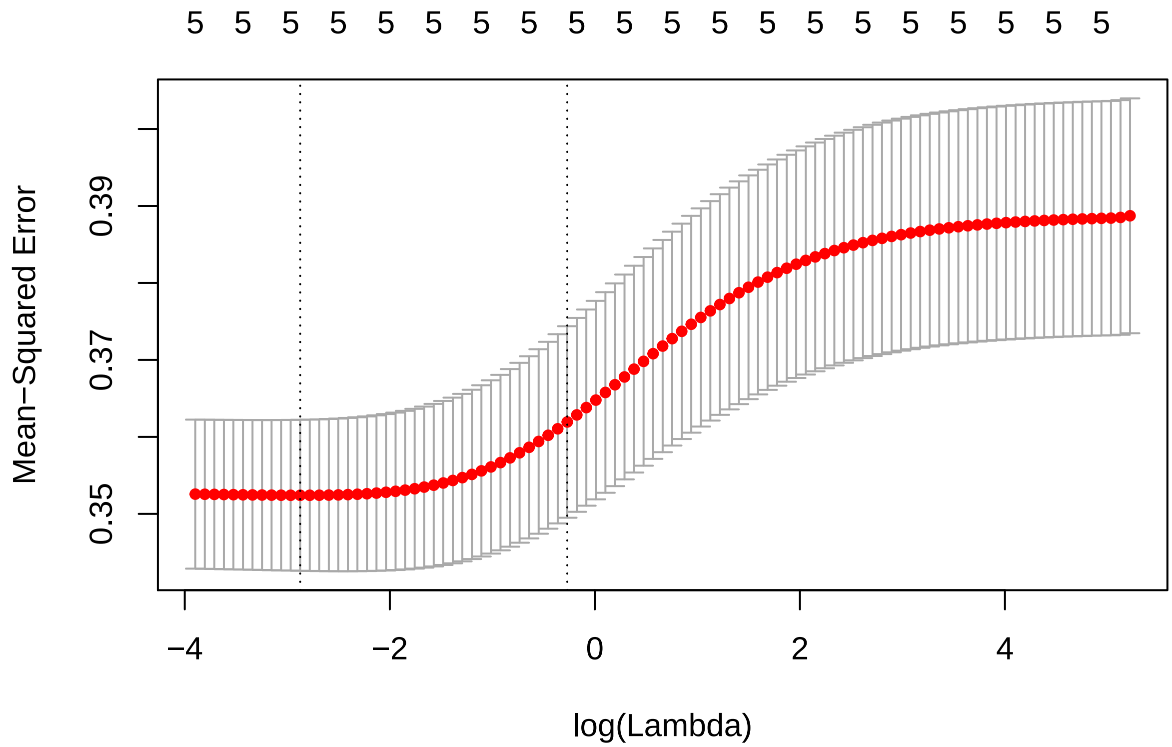


LLOOOGGG

LLOOOGG!!!

```
#Ridge

#Run cross validation
cvRidge2<-cv.glmnet(xTrain,yTrain2,alpha=0)
plot(cvRidge2)
```




```
#See which lambda minimizes the MSE and the corresponding coefficient estimates
lambdaOptRidge2<-cvRidge2$lambda.min #Or, use lambda.1se
print(lambdaOptRidge2)
```

```
## [1] 0.05647041
```

```
coef(cvRidge2,s=lambdaOptRidge2) #Alternatively, use s="lambda.min"
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.50719601
## femWomen    -0.08704214
## marMarried   0.12436030
## kid5        -0.09709482
## phd          0.04024237
## ment         0.01617343
```

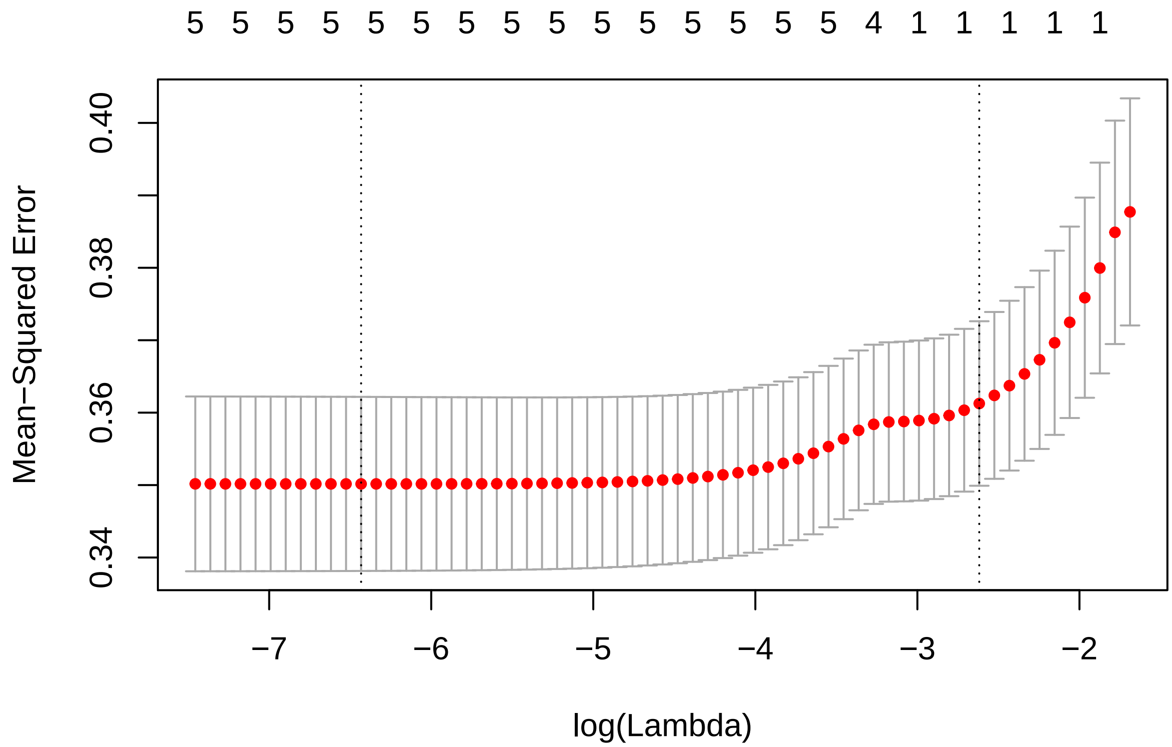
```
#Get y hat for training data
modRidge2<-cvRidge2$glmnet.fit
yHatRidge2<-predict(modRidge2,s=lambdaOptRidge2,newx=xTrain)

#Get MSE on training data
MSERidge2<-mean((yTrain2-yHatRidge2)^2) #3.194847
print(MSERidge2)
```

```
## [1] 0.3435092
```

```
#LASSO

#Run cross validation
cvLASSO2<-cv.glmnet(xTrain,yTrain2,alpha=1)
plot(cvLASSO2)
```




```r
#See which lambda minimizes the MSE and the corresponding coefficient estimates
lambdaOptLASSO2<-cvLASSO2$lambda.min
print(lambdaOptLASSO2)
```

```
## [1] 0.001608297
```

```r
coef(cvLASSO2,s=lambdaOptLASSO2)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.49574484
## femWomen    -0.09220022
## marMarried   0.14002743
## kid5        -0.10992285
## phd          0.03965661
## ment         0.01751149
```

```r
#Get y hat for training data
modLASSO2<-cvLASSO2$glmnet.fit
yHatLASSO2<-predict(modLASSO2,s=lambdaOptLASSO2,newx=xTrain)

#Get MSE on training data
MSELASSO2<-mean((yTrain2-yHatLASSO2)^2) #3.191976
print(MSELASSO2)
```

```
## [1] 0.3431454
```

```r
#Kernel Regularized Least Squares
modKRLS2<-krls(X=xTrain,y=yTrain2)
```

```
## Warning in Eigenobject$values + lambda: Recycling array of length 1 in vector-array arithmetic is dep
##   Use c() or as.vector() instead.
```

```
## Warning in Eigenobject$values + lambda: Recycling array of length 1 in vector-array arithmetic is dep
##   Use c() or as.vector() instead.

##
##  Average Marginal Effects:
##
##    femWomen  marMarried        kid5         phd        ment
## -0.08417597  0.07551640 -0.04632589  0.02629924  0.01835693
##
##  Quartiles of Marginal Effects:
##
##        femWomen marMarried         kid5         phd       ment
## 25% -0.13713224 0.02635293 -0.093112335 -0.03429341 0.01432425
## 50% -0.08877888 0.06874733 -0.030572545  0.03027724 0.01967991
## 75% -0.03408611 0.11594498  0.007904866  0.07877286 0.02338940
```

```r
summary(modKRLS2)
```

```
## * *********************** *
## Model Summary:
##
## R2: 0.1537408
##
## Average Marginal Effects:
##                    Est  Std. Error   t value     Pr(>|t|)
## femWomen*   -0.08417597 0.060781164 -1.384902 1.665886e-01
## marMarried*  0.07551640 0.058863671  1.282903 2.000132e-01
## kid5        -0.04632589 0.024288827 -1.907292 5.695214e-02
## phd          0.02629924 0.018952794  1.387618 1.657600e-01
## ment         0.01835693 0.002878281  6.377741 3.552048e-10
##
## (*) average dy/dx is for discrete change of dummy variable from min to max (i.e. usually 0 to 1))
##
##
## Quartiles of Marginal Effects:
##                    25%         50%          75%
## femWomen*   -0.13713224 -0.08877888 -0.034086112
## marMarried*  0.02635293  0.06874733  0.115944981
## kid5        -0.09311234 -0.03057255  0.007904866
## phd         -0.03429341  0.03027724  0.078772859
## ment         0.01432425  0.01967991  0.023389402
##
## (*) quantiles of dy/dx is for discrete change of dummy variable from min to max (i.e. usually 0 to 1)
```

```r
yHatKRLS2<-predict(modKRLS2,newdata=xTrain)

#Get MSE on training data
MSEKRLS2<-mean((yTrain2-yHatKRLS2$newdataK)^2) #3.191976
print(MSEKRLS2)
```

```
## [1] 0.7170357
```

TESTING RESULTS:

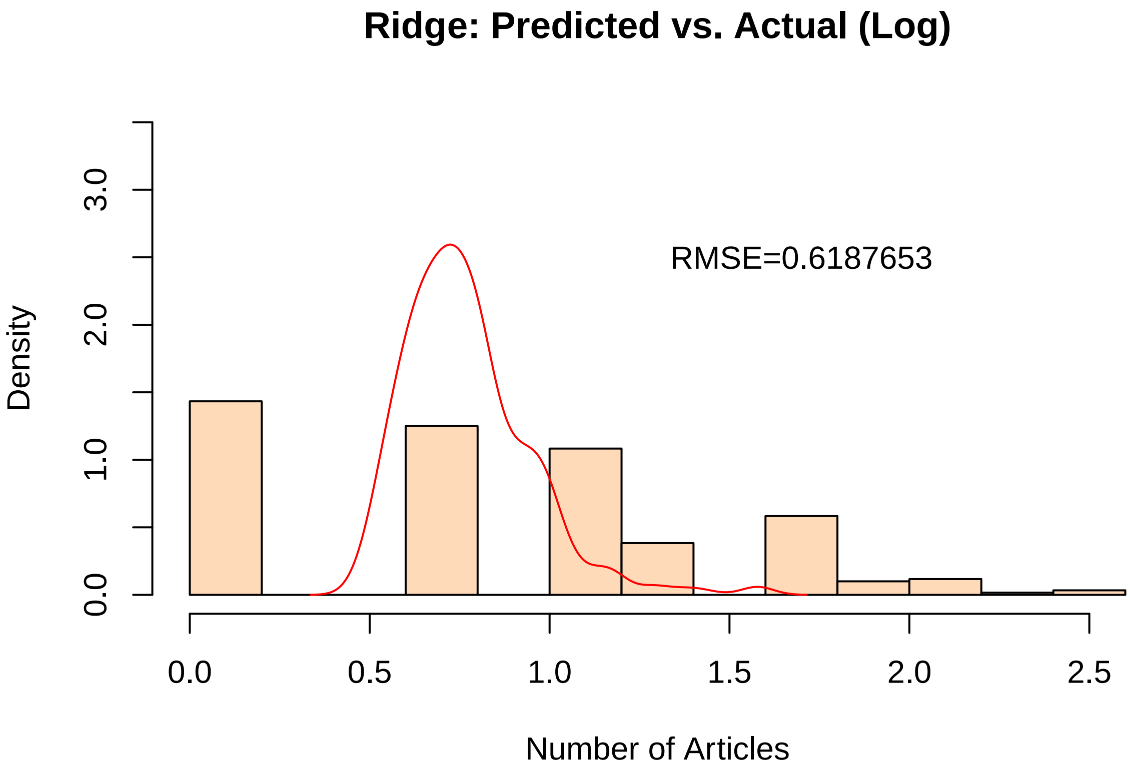
```
yTest2 <- log(dataTest[,1]+1)
xTest2 <-model.matrix(art~.,data=dataTest2)[,-1]
```

RIDGE TEST MLE

```
yHatTestRidge2<-predict(modRidge2,s=lambdaOptRidge2,newx=xTest2)
MSETestRidge2<-mean((yTest2-yHatTestRidge2)^2)
print(sqrt(MSETestRidge2))
```

```
## [1] 0.6187791
```

```
hist(yTest2,freq = F, ylim = c(0,3.5),col="peachpuff", main = "Ridge: Predicted vs. Actual (Log)", xlab
lines(density(yHatTestRidge2), col = "red")
text(1.7, y = 2.5, labels = "RMSE=0.6187653")
```

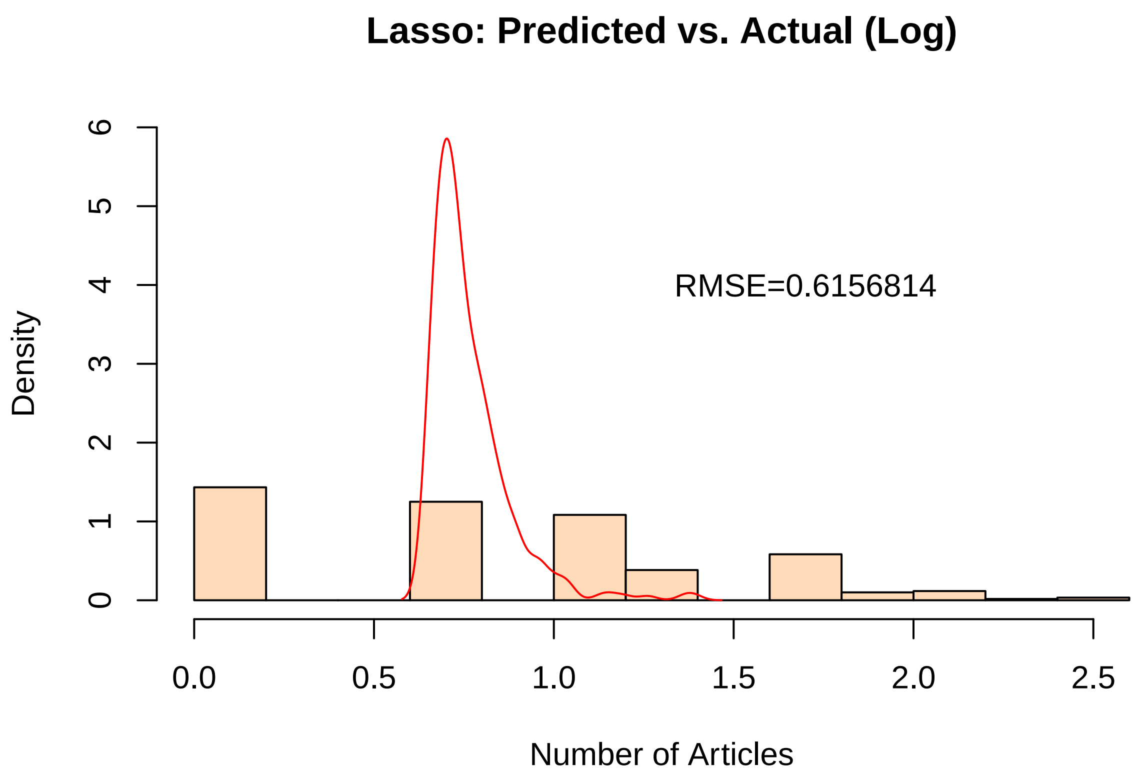


LASSO TEST MLE

```
yHatTestLasso2<-predict(modLASSO2,s=lambdaOptRidge2,newx=xTest2)
MSETestLasso2<-mean((yTest2-yHatTestLasso2)^2)
print(sqrt(MSETestLasso2))
```

```
## [1] 0.6214921
```

```
hist(yTest2,freq = F, ylim = c(0,6),col="peachpuff", main = "Lasso: Predicted vs. Actual (Log)", xlab =
lines(density(yHatTestLasso2), col = "red")
text(1.7, y = 4, labels = "RMSE=0.6156814")
```

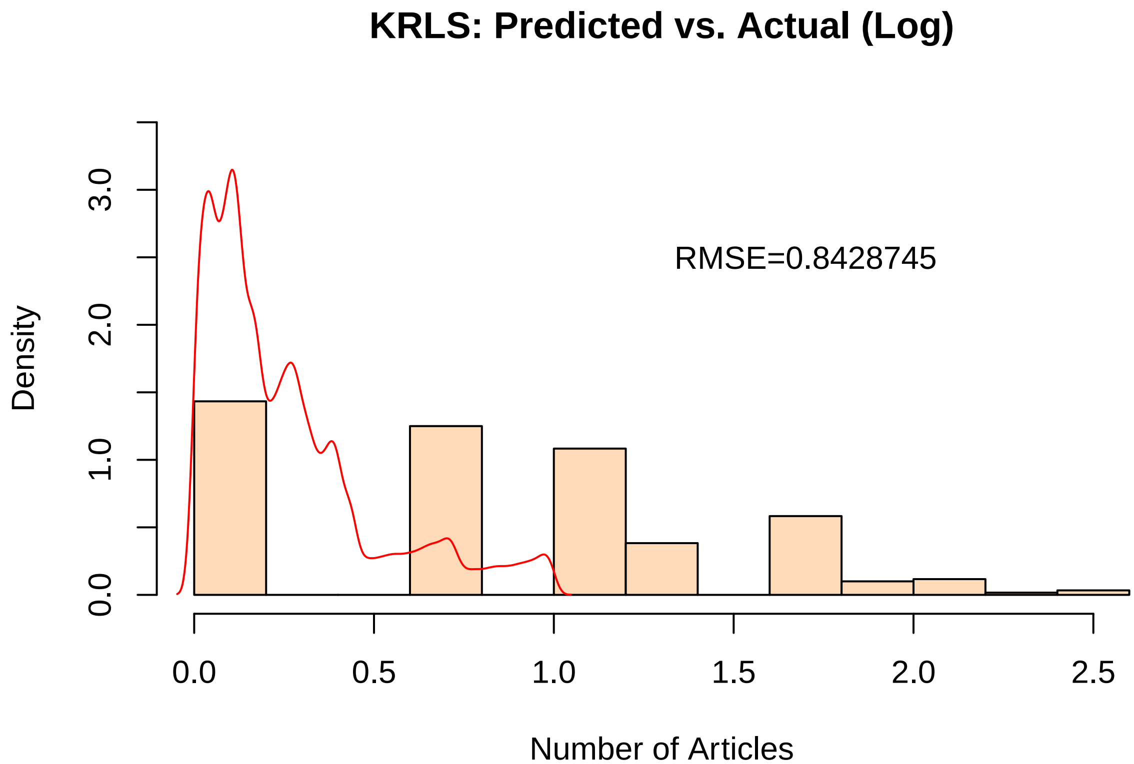


KRLS TEST MLE

```
yHatTestKRLS2<-predict(modKRLS2,newdata=xTest2)$newdataK
MSETestKRLS2<-mean((yTest2-yHatTestKRLS2)^2)
print(sqrt(MSETestKRLS2))
```

```
## [1] 0.8985234
```

```
hist(yTest2,freq = F, ylim = c(0,3.5),col="peachpuff", main = "KRLS: Predicted vs. Actual (Log)", xlab =
lines(density(yHatTestKRLS2), col = "red")
text(1.7, y = 2.5, labels = "RMSE=0.8428745")
```



KRLS: Predicted vs. Actual (Log)
RMSE=0.8428745
Density
0.0
1.0
2.0
3.0
0.0
0.5
1.0
1.5
2.0
2.5
Number of Articles