Variables Selection

Background

Selected molecular descriptors from the Dragon chemoinformatics application were used to predict bioconcentration factors for 779 chemicals in order to evaluate QSAR (Quantitative Structure Activity Relationship). This dataset was obtained from the UCI machine learning repository.

The dataset consists of 779 observations of 10 attributes. Below is a brief description of each feature and the response variable (logBCF) in our dataset:

1. *nHM* - number of heavy atoms (integer)
2. *piPC09* - molecular multiple path count (numeric)
3. *PCD* - difference between multiple path count and path count (numeric)
4. *X2Av* - average valence connectivity (numeric)
5. *MLOGP* - Moriguchi octanol-water partition coefficient (numeric)
6. *ON1V* - overall modified Zagreb index by valence vertex degrees (numeric)
7. *N.072* - Frequency of RCO-N< / >N-X=X fragments (integer)
8. *B02[C-N]* - Presence/Absence of C-N atom pairs (binary)
9. *F04[C-O]* - Frequency of C-O atom pairs (integer)
10. *logBCF* - Bioconcentration Factor in log units (numeric)

Note that all predictors with the exception of B02[C-N] are quantitative. For the purpose of this assignment, DO NOT CONVERT B02[C-N] to factor. Leave the data in its original format - numeric in R.

Please load the dataset “Bio_pred” and then split the dataset into a train and test set in a 80:20 ratio. Use the training set to build the models in Questions 1-6. Use the test set to help evaluate model performance in Question 7. Please make sure that you are using R version 3.6.X.

Read Data

```
# Clear variables in memory
rm(list=ls())

# Import the libraries
library(CombMSC)
library(boot)
library(leaps)
library(MASS)
library(glmnet)

# Ensure that the sampling type is correct
RNGkind(sample.kind="Rejection")

# Set a seed for reproducibility
set.seed(100)

# Read data
fullData = read.csv("Bio_pred.csv",header=TRUE)

# Split data for traIning and testing
testRows = sample(nrow(fullData),0.2*nrow(fullData))
testData = fullData[testRows, ]
trainData = fullData[-testRows, ]

head(trainData)
```

```
##   nHM piPC09  PCD X2Av MLOGP ON1V N.072 B02.C.N. F04.C.O. logBCF
## 2   0  0.000 1.47 0.14  1.70 0.88     0        1        5   0.93
## 3   0  0.000 1.20 0.25  4.14 2.06     0        0        0   3.24
## 4   0  0.000 1.69 0.13  1.89 0.79     0        1        8  -0.40
## 5   0  0.000 0.52 0.25  2.65 1.31     0        0        0   2.24
## 6   0  0.000 1.40 0.18  2.85 0.86     0        0        0   1.13
## 8   1  3.446 1.23 0.24  2.01 1.12     0        0        7   2.76
```

Question 1: Full Model

a. Fit a standard linear regression with the variable *logBCF* as the response and the other variables as predictors. Call it *model1*. Display the model summary.

```
model1 = lm(logBCF ~ ., data=trainData)
summary(model1)
```

```
##
## Call:
## lm(formula = logBCF ~ ., data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2577 -0.5180  0.0448  0.5117  4.0423
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.001422   0.138057   0.010  0.99179
## nHM          0.137022   0.022462   6.100 1.88e-09 ***
## piPC09       0.031158   0.020874   1.493  0.13603
## PCD          0.055655   0.063874   0.871  0.38391
## X2Av        -0.031890   0.253574  -0.126  0.89996
## MLOGP        0.506088   0.034211  14.793  < 2e-16 ***
## ON1V         0.140595   0.066810   2.104  0.03575 *
## N.072       -0.073334   0.070993  -1.033  0.30202
## B02.C.N.    -0.158231   0.080143  -1.974  0.04879 *
## F04.C.O.    -0.030763   0.009667  -3.182  0.00154 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7957 on 614 degrees of freedom
## Multiple R-squared:  0.6672, Adjusted R-squared:  0.6623
## F-statistic: 136.8 on 9 and 614 DF,  p-value: < 2.2e-16
```

Answer

See above model1’s summary.

b. Which regression coefficients are significant at the 95% confidence level? At the 99% confidence level?

```
# Significant coefficients at the 95% confidence level
print("Significant coefficients at the 95% confidence level:")
```

```
## [1] "Significant coefficients at the 95% confidence level:"
```

```
which(summary(model1)$coeff[,4]<0.05)
```

```
##      nHM    MLOGP     ON1V B02.C.N. F04.C.O.
##        2        6        7        9       10
```

```
# Significant coefficients at the 99% confidence level
print("Significant coefficients at the 99% confidence level:")
```

```
## [1] "Significant coefficients at the 99% confidence level:"
```

```
which(summary(model1)$coeff[,4]<0.01)
```

```
##      nHM    MLOGP F04.C.O.
##        2        6       10
```

Answer

The significant coefficients at the 95% confidence level are nHM, MLOGP, ON1V, B02.C.N., F04.C.O..

The significant coefficients at the 99% confidence level are nHM, MLOGP, F04.C.O.

c. What are the 10-fold and leave one out cross-validation scores for this model?

```
set.seed(100)

n = nrow(trainData)
## CV: 10-fold and leave one out
glm_model1 = glm(logBCF ~ ., family="gaussian", data=trainData)
c(cv.glm(trainData, glm_model1, K=10)$delta[1], cv.glm(trainData, glm_model1, K=n)$delta[1])
```

```
## [1] 0.6512928 0.6529872
```

Answer

See above the estimated 10-fold cross-validation prediction error is 0.6512928 and the Leave One Out CV is 0.6529872.

d. What are the Mallow’s Cp, AIC, and BIC criterion values for this model?

```
set.seed(100)

sigma = summary(model1)$sigma

c(Cp(model1, S2=sigma^2),
  AIC(model1, k=2), AIC(model1,k=log(n)))
```

```
## [1]   10.000 1497.477 1546.274
```

Answer

See above Mallow’s Cp value is 10, the AIC is 1497.477 and the BIC is 1546.274.

e. Build a new model on the training data with only the variables which coefficients were found to be statistically significant at the 99% confident level. Call it *model2*. Perform an ANOVA test to compare this new model with the full model. Which one would you prefer? Is it good practice to select variables based on statistical significance of individual coefficients? Explain.

```
set.seed(100)

model2 = lm(logBCF ~ nHM + MLOGP + F04.C.O., data=trainData)

anova(model2, model1)
```

```
## Analysis of Variance Table
##
## Model 1: logBCF ~ nHM + MLOGP + F04.C.O.
## Model 2: logBCF ~ nHM + piPC09 + PCD + X2Av + MLOGP + ON1V + N.072 + B02.C.N. +
##     F04.C.O.
##   Res.Df    RSS Df Sum of Sq     F  Pr(>F)
## 1    620 400.51
## 2    614 388.70  6    11.809 3.109 0.00523 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Answer

Here the null hypothesis is that the coefficients not included in model2 are all 0 while the alternative is that at least one is not 0. With a p-value of 0.00523 we reject the null and conclude that at least one of the coefficients not included in model2 is not 0 and hence have predicting power. It is ***NOT*** a good practice to select variables based on the statistical significance of individual coefficients so it is hard to conclude which model is better. We would have to go through variables selection methods to determine which coefficients we should include in our “best” model. With that being said, I would suspect that some of the coefficients not included in model2 would be included in such (“best”) model.

Question 2: Full Model Search

a. Compare all possible models using Mallow’s Cp. What is the total number of possible models with the full set of variables? Display a table indicating the variables included in the best model of each size and the corresponding Mallow’s Cp value.

Hint: You can use nbest parameter.

```
set.seed(100)

num_predictors = ncol(trainData)-1
num_models = 2^num_predictors

out = leaps(trainData[,-c(10)], trainData$logBCF, method = "Cp", nbest=1)
cbind(as.matrix(out$which),out$Cp)
```

```
##   1 2 3 4 5 6 7 8 9
## 1 0 0 0 0 1 0 0 0 0 58.596851
## 2 1 0 0 0 1 0 0 0 0 17.737801
## 3 1 1 0 0 1 0 0 0 0 15.184626
## 4 1 1 0 0 1 0 0 0 1  9.495041
## 5 1 1 0 0 1 0 0 1 1  7.240754
## 6 1 1 0 0 1 1 0 1 1  6.116174
## 7 1 1 0 0 1 1 1 1 1  6.831852
## 8 1 1 1 0 1 1 1 1 1  8.015816
## 9 1 1 1 1 1 1 1 1 1 10.000000
```

```
best.model = which(out$Cp==min(out$Cp))

print("Best model:")
```

```
## [1] "Best model:"
```

```
cbind(as.matrix(out$which),out$Cp)[best.model,]
```

```
##        1        2        3        4        5        6        7        8 
## 1.000000 1.000000 0.000000 0.000000 1.000000 1.000000 0.000000 1.000000 
##        9          
## 1.000000 6.116174
```

```
print(paste("The total number of possible models is: ", num_models))
```

```
## [1] "The total number of possible models is:  512"
```

Answer

The total number of possible model is 512. See above the table indicating the variables included in the best model of each size, indicted by each row, and where each column corresponds to the variables nHM, piPC09, PCD, X2Av, MLOGP, ON1V, N.072, B02.C.N., F04.C.O..

b. How many variables are in the model with the lowest Mallow's Cp value? Which variables are they? Fit this model and call it *model3*. Display the model summary.

```
set.seed(100)

model3 = lm(logBCF ~ nHM+piPC09+MLOGP+ON1V+B02.C.N.+F04.C.O., data=trainData)
summary(model3)
```

```
##
## Call:
## lm(formula = logBCF ~ nHM + piPC09 + MLOGP + ON1V + B02.C.N. +
##     F04.C.O., data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2364 -0.5234  0.0421  0.5196  4.1159
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.035785   0.099454   0.360  0.71911
## nHM          0.124086   0.019083   6.502 1.63e-10 ***
## piPC09       0.042167   0.014135   2.983  0.00297 **
## MLOGP        0.528522   0.029434  17.956  < 2e-16 ***
## ON1V         0.098099   0.055457   1.769  0.07740 .
## B02.C.N.    -0.160204   0.073225  -2.188  0.02906 *
## F04.C.O.    -0.028644   0.009415  -3.042  0.00245 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7951 on 617 degrees of freedom
## Multiple R-squared:  0.666,  Adjusted R-squared:  0.6628
## F-statistic: 205.1 on 6 and 617 DF,  p-value: < 2.2e-16
```

Answer

The model with the lowest Mallow's Cp value (6.116174) includes 6 variables which are nHM, piPC09, MLOGP, ON1V, B02.C.N. and F04.C.O.. See above the summary of this model.

Question 3: Stepwise Regression

a. Perform backward stepwise regression using BIC. Allow the minimum model to be the model with only an intercept, and the full model to be *model1*. Display the model summary of your final model. Call it *model4*

```
set.seed(100)

# Backward
#Define intercept-only model
minimum = lm(logBCF ~ 1, data=trainData)
full = model1

# Backward. AIC->k=2; BIC->k = log(n)
model4 = (step(full, scope=list(lower=minimum, upper=full) ,k = log(nrow(trainData)), direction=
"backward"))
```

```
## Start:  AIC=-231
## logBCF ~ nHM + piPC09 + PCD + X2Av + MLOGP + ON1V + N.072 + B02.C.N. +
##     F04.C.O.
##
##            Df Sum of Sq    RSS      AIC
## - X2Av      1     0.010 388.71 -237.417
## - PCD       1     0.481 389.18 -236.662
## - N.072     1     0.676 389.38 -236.350
## - piPC09    1     1.411 390.11 -235.173
## - B02.C.N.  1     2.468 391.17 -233.484
## - ON1V      1     2.804 391.51 -232.949
## <none>                  388.70 -230.997
## - F04.C.O.  1     6.410 395.11 -227.226
## - nHM       1    23.557 412.26 -200.718
## - MLOGP     1   138.539 527.24  -47.211
##
## Step:  AIC=-237.42
## logBCF ~ nHM + piPC09 + PCD + MLOGP + ON1V + N.072 + B02.C.N. +
##     F04.C.O.
##
##            Df Sum of Sq    RSS      AIC
## - PCD       1     0.517 389.23 -243.025
## - N.072     1     0.667 389.38 -242.783
## - piPC09    1     1.423 390.14 -241.574
## - B02.C.N.  1     2.510 391.22 -239.838
## - ON1V      1     2.915 391.63 -239.192
## <none>                  388.71 -237.417
## - F04.C.O.  1     6.491 395.21 -233.520
## - nHM       1    25.431 414.15 -204.309
## - MLOGP     1   146.081 534.80  -44.772
##
## Step:  AIC=-243.02
## logBCF ~ nHM + piPC09 + MLOGP + ON1V + N.072 + B02.C.N. + F04.C.O.
##
##            Df Sum of Sq    RSS      AIC
## - N.072     1     0.813 390.04 -248.159
## - B02.C.N.  1     2.099 391.33 -246.105
## - ON1V      1     2.412 391.64 -245.606
## <none>                  389.23 -243.025
## - F04.C.O.  1     6.088 395.32 -239.776
## - piPC09    1     6.203 395.43 -239.594
## - nHM       1    27.541 416.77 -206.800
## - MLOGP     1   181.833 571.06  -10.264
##
## Step:  AIC=-248.16
## logBCF ~ nHM + piPC09 + MLOGP + ON1V + B02.C.N. + F04.C.O.
##
##            Df Sum of Sq    RSS      AIC
## - ON1V      1     1.978 392.02 -251.438
## - B02.C.N.  1     3.026 393.07 -249.773
## <none>                  390.04 -248.159
## - piPC09    1     5.626 395.67 -245.659
## - F04.C.O.  1     5.851 395.89 -245.304
## - nHM       1    26.728 416.77 -213.236
## - MLOGP     1   203.819 593.86    7.728
##
## Step:  AIC=-251.44
## logBCF ~ nHM + piPC09 + MLOGP + B02.C.N. + F04.C.O.
##
##            Df Sum of Sq    RSS      AIC
## - B02.C.N.  1     2.693 394.72 -253.602
## - F04.C.O.  1     3.902 395.92 -251.695
## <none>                  392.02 -251.438
## - piPC09    1     7.252 399.27 -246.437
## - nHM       1    25.197 417.22 -219.003
## - MLOGP     1   247.006 639.03   47.031
##
## Step:  AIC=-253.6
## logBCF ~ nHM + piPC09 + MLOGP + F04.C.O.
##
##            Df Sum of Sq    RSS      AIC
## <none>                  394.72 -253.602
## - F04.C.O.  1     4.868 399.58 -252.390
## - piPC09    1     5.798 400.51 -250.939
## - nHM       1    26.847 421.56 -218.977
## - MLOGP     1   302.931 697.65   95.359
```

```
summary(model4)
```

```
##
## Call:
## lm(formula = logBCF ~ nHM + piPC09 + MLOGP + F04.C.O., data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2611 -0.5126  0.0517  0.5353  4.3488
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.008695   0.078196  -0.111  0.91150
## nHM          0.114029   0.017574   6.489 1.78e-10 ***
## piPC09       0.041119   0.013636   3.015  0.00267 **
## MLOGP        0.566473   0.025990  21.796  < 2e-16 ***
## F04.C.O.    -0.022104   0.008000  -2.763  0.00590 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7985 on 619 degrees of freedom
## Multiple R-squared:  0.662,  Adjusted R-squared:  0.6599
## F-statistic: 303.1 on 4 and 619 DF,  p-value: < 2.2e-16
```

Answer

See above model4 summary.

b. How many variables are in *model4*? Which regression coefficients are significant at the 99% confidence level?

Answer

There are 4 variables (and an intercept) in model4. Model4’s coefficients that are significant at the 99% level are nHM, piPC09, MLOGP and F04.C.O. which are all of them except the intercept.

c. Perform forward stepwise selection with AIC. Allow the minimum model to be the model with only an intercept, and the full model to be *model1*. Display the model summary of your final model. Call it *model5*. Do the variables included in *model5* differ from the variables in *model4*?

```
set.seed(100)

# Forward
#Define intercept-only model
minimum = lm(logBCF ~ 1, data=trainData)
full = model1

# Forward. AIC->k=2; BIC->k = log(n)
model5 = (step(minimum, scope=list(lower=minimum, upper=full) ,k = 2, direction="forward"))
```

```
## Start:  AIC=393.14
## logBCF ~ 1
## 
##             Df Sum of Sq     RSS     AIC
## + MLOGP      1    738.32  429.60 -228.94
## + nHM        1    255.66  912.25  240.98
## + piPC09     1    220.90  947.02  264.31
## + PCD        1    150.75 1017.17  308.90
## + B02.C.N.   1    139.23 1028.68  315.93
## + N.072      1     43.55 1124.37  371.43
## + ON1V       1     27.76 1140.16  380.13
## + F04.C.O.   1     20.79 1147.13  383.93
## <none>                   1167.92  393.14
## + X2Av       1      2.45 1165.46  393.83
## 
## Step:  AIC=-228.94
## logBCF ~ MLOGP
## 
##             Df Sum of Sq    RSS     AIC
## + nHM        1   27.1327 402.47 -267.65
## + B02.C.N.   1    4.1778 425.42 -233.04
## + F04.C.O.   1    4.1526 425.45 -233.00
## + X2Av       1    3.2819 426.32 -231.72
## + ON1V       1    2.3664 427.23 -230.38
## <none>                   429.60 -228.94
## + piPC09     1    1.0443 428.55 -228.46
## + N.072      1    0.2481 429.35 -227.30
## + PCD        1    0.1198 429.48 -227.11
## 
## Step:  AIC=-267.65
## logBCF ~ MLOGP + nHM
## 
##             Df Sum of Sq    RSS     AIC
## + piPC09     1   2.88247 399.58 -270.13
## + F04.C.O.   1   1.95225 400.51 -268.68
## + B02.C.N.   1   1.93200 400.53 -268.65
## <none>                   402.47 -267.65
## + PCD        1   1.23679 401.23 -267.57
## + N.072      1   0.40989 402.06 -266.29
## + ON1V       1   0.33115 402.13 -266.16
## + X2Av       1   0.11836 402.35 -265.83
## 
## Step:  AIC=-270.13
## logBCF ~ MLOGP + nHM + piPC09
## 
##             Df Sum of Sq    RSS     AIC
## + F04.C.O.   1    4.8680 394.72 -275.78
## + B02.C.N.   1    3.6597 395.92 -273.88
## + N.072      1    1.4631 398.12 -270.42
## <none>                   399.58 -270.13
## + X2Av       1    0.5349 399.05 -268.97
## + ON1V       1    0.0065 399.58 -268.14
## + PCD        1    0.0001 399.58 -268.13
```

```
##
## Step:  AIC=-275.78
## logBCF ~ MLOGP + nHM + piPC09 + F04.C.O.
##
##            Df Sum of Sq    RSS     AIC
## + B02.C.N.  1   2.69326 392.02 -278.06
## + ON1V      1   1.64544 393.07 -276.39
## <none>                  394.72 -275.78
## + N.072     1   1.06163 393.65 -275.46
## + X2Av      1   0.51804 394.20 -274.60
## + PCD       1   0.07778 394.64 -273.91
##
## Step:  AIC=-278.06
## logBCF ~ MLOGP + nHM + piPC09 + F04.C.O. + B02.C.N.
##
##         Df Sum of Sq    RSS     AIC
## + ON1V   1   1.97807 390.04 -279.21
## <none>               392.02 -278.06
## + N.072  1   0.37905 391.64 -276.66
## + X2Av   1   0.12543 391.90 -276.25
## + PCD    1   0.00000 392.02 -276.06
##
## Step:  AIC=-279.21
## logBCF ~ MLOGP + nHM + piPC09 + F04.C.O. + B02.C.N. + ON1V
##
##         Df Sum of Sq    RSS     AIC
## <none>               390.04 -279.21
## + N.072  1   0.81306 389.23 -278.51
## + PCD    1   0.66238 389.38 -278.27
## + X2Av   1   0.02794 390.02 -277.26
```

```
summary(model5)
```

```
##
## Call:
## lm(formula = logBCF ~ MLOGP + nHM + piPC09 + F04.C.O. + B02.C.N. +
##     ON1V, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2364 -0.5234  0.0421  0.5196  4.1159
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.035785   0.099454   0.360  0.71911
## MLOGP        0.528522   0.029434  17.956  < 2e-16 ***
## nHM          0.124086   0.019083   6.502 1.63e-10 ***
## piPC09       0.042167   0.014135   2.983  0.00297 **
## F04.C.O.    -0.028644   0.009415  -3.042  0.00245 **
## B02.C.N.    -0.160204   0.073225  -2.188  0.02906 *
## ON1V         0.098099   0.055457   1.769  0.07740 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7951 on 617 degrees of freedom
## Multiple R-squared:  0.666,  Adjusted R-squared:  0.6628
## F-statistic: 205.1 on 6 and 617 DF,  p-value: < 2.2e-16
```

Answer

See above model5 summary output. Model5's coefficients differ from model4's coefficients. Model5 includes the variables B02.C.N. and ON1V which are not included in model4.

d. Compare the adjusted R^2, Mallow's Cp, AICs and BICs of the full model(*model1*), the model found in Question 2 (*model3*), and the model found using backward selection with BIC (*model4*). Which model is preferred based on these criteria and why?

```
set.seed(100)

# Adjusted R^2
model1_Adjr = summary(model1)$adj.r.squared
model3_Adjr = summary(model3)$adj.r.squared
model4_Adjr = summary(model4)$adj.r.squared

# Calculate sigma for Mallow's Cp. Always use the Sigma of the FULL model!!!
sigma1 = summary(model1)$sigma

n = nrow(trainData)

print("Model1's R^2, Mallow's Cp, AIC and BIC, respectively:")
```

```
## [1] "Model1's R^2, Mallow's Cp, AIC and BIC, respectively:"
```

```
c(model1_Adjr, Cp(model1, S2=sigma1^2), AIC(model1, k=2), AIC(model1,k=log(n)))
```

```
## [1]    0.6623027   10.0000000 1497.4765327 1546.2741868
```

```
print("Model3's R^2, Mallow's Cp, AIC and BIC, respectively:")
```

```
## [1] "Model3's R^2, Mallow's Cp, AIC and BIC, respectively:"
```

```
c(model3_Adjr, Cp(model3, S2=sigma1^2), AIC(model3, k=2), AIC(model3,k=log(n)))
```

```
## [1]    0.6627864    6.1161737 1493.6234741 1529.1126771
```

```
print("Model4's R^2, Mallow's Cp, AIC and BIC, respectively:")
```

```
## [1] "Model4's R^2, Mallow's Cp, AIC and BIC, respectively:"
```

```
c(model4_Adjr, Cp(model4, S2=sigma1^2), AIC(model4, k=2), AIC(model4,k=log(n)))
```

```
## [1]    0.6598504    9.4950405 1497.0523636 1523.6692658
```

Answer

Based on these criteria, it seems like there is no clear and definitive answer. Meaning, while the R^2 are almost equal, model1 and model3 have a slightly better R^2 than model4, but the difference is insignificant. Based on the Mallow's Cp criteria, model3 is best with the lowest value of 6.116. Based on the AIC criteria, model3 is best with a lowest value of 1493.62, while model4 is best based on the BIC criteria with a lowest value of 1523.669.

Question 4: Ridge Regression

a. Perform ridge regression on the training set. Use cv.glmnet() to find the lambda value that minimizes the cross-validation error using 10 fold CV.

```
set.seed(100)

# Ridge regression -> alpha=0
#find optimal lambda

# Remove response variable
X = trainData[,-10]

cv.ridge = cv.glmnet(x=as.matrix(X), y=trainData$logBCF, alpha = 0, family = "gaussian", type.me
asure = c("mse"), nfolds=10)
opt_lambda = cv.ridge$lambda.min

print(paste("Optimal lambda is: ", opt_lambda))
```

```
## [1] "Optimal lambda is:  0.108775012396901"
```

Answer

See above the lambda value that minimizes the cross-validation error using 10 fold CV is 0.108775.

b. List the value of coefficients at the optimum lambda value.

```
set.seed(100)

# ## Another method to fit the model and extract coefficients
# ## Fit ridge model with best lambda
# ridge.lm = glmnet(x=as.matrix(X), y=trainData$logBCF, alpha=0, lambda=cv.ridge$lambda.min)
# coef(ridge.lm)

# Extract the coefficients at the optimal lambda
ridge.lm = glmnet(x=as.matrix(X), y=trainData$logBCF, alpha=0, nlambda=100)
coef(ridge.lm, s=cv.ridge$lambda.min)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.13841426
## nHM          0.14391877
## piPC09       0.03735762
## PCD          0.08235334
## X2Av        -0.06901352
## MLOGP        0.44403654
## ON1V         0.15770114
## N.072       -0.09683534
## B02.C.N.    -0.20919397
## F04.C.O.    -0.03177144
```

Answer

See above the value of coefficients at the optimal ridge regression’s lambda value.

c. How many variables were selected? Give an explanation for this number.

Answer

All 9 variables were selected, as expected in ridge regression. Ridge regression is not variables selection since the coefficients are "shrinking" and non of the coefficients will ever be 0. Ridge regression is a parsimonious model that performs L2 regularization. The L2 regularization adds a penalty equivalent to the square of the magnitude of regression coefficients and tries to minimize them (but they will not be 0). Hence, we should expect all coefficients to be included in the model, but the value of the coefficients is smaller than when we fit a "standard" regression.

Question 5: Lasso Regression

a. Perform lasso regression on the training set.Use cv.glmnet() to find the lambda value that minimizes the cross-validation error using 10 fold CV.

```
set.seed(100)

# LASSO regression -> alpha=1
#find optimal lambda

# Remove response variable
X = trainData[,-10]

cv.lasso = cv.glmnet(x=as.matrix(X), y=trainData$logBCF, alpha = 1, family = "gaussian", type.me
asure = c("mse"), nfolds=10)
opt_lambda = cv.lasso$lambda.min

print(paste("Optimal lambda is: ", opt_lambda))
```

```
## [1] "Optimal lambda is:  0.00785443587360036"
```

Answer

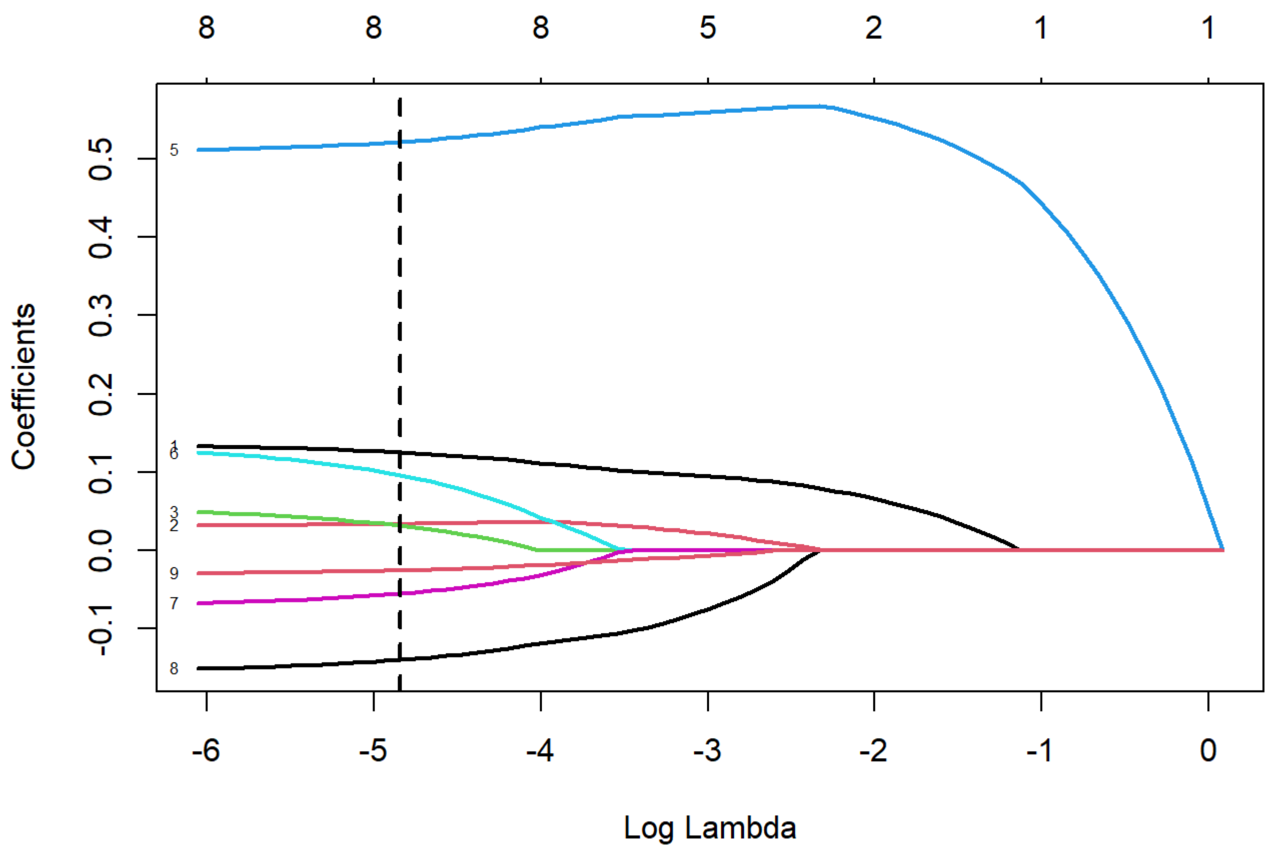
See above the lambda value that minimizes the cross-validation error using 10 fold CV is 0.0078545.

b. Plot the regression coefficient path.

```
set.seed(100)


## Fit lasso models with 100 values for lambda
lasso.lm = glmnet(x=as.matrix(X), y=trainData$logBCF, alpha=1, nlambda=100)

# Coefficient path plots
plot(lasso.lm, xvar="lambda", label=TRUE, lwd=2)
abline(v=log(cv.lasso$lambda.min),col='black',lty = 2,lwd=2)
```



Answer

See above the regression coefficient path.

c. How many variables were selected? Which are they?

```
set.seed(100)

#Extract coefficients at optimal lambda
coef(lasso.lm, s=cv.lasso$lambda.min)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.02722838
## nHM          0.12543866
## piPC09       0.03387665
## PCD          0.03194878
## X2Av         .
## MLOGP        0.52174346
## ON1V         0.09633951
## N.072       -0.05487196
## B02.C.N.    -0.13961811
## F04.C.O.    -0.02535576
```

Answer

8 variables are selected (and an intercept). The coefficients selected are nHM, piPC09, PCD, MLOGP, ON1V, N.072, B02.C.N., F04.C.O. and their values are indicated above.

Question 6: Elastic Net

a. Perform elastic net regression on the training set. Use cv.glmnet() to find the lambda value that minimizes the cross-validation error using 10 fold CV. Give equal weight to both penalties.

```
set.seed(100)

# Elastic Net -> Equal weight -> alpha=0.5
#find optimal lambda

# Remove response variable
X = trainData[,-10]

cv.elastic = cv.glmnet(x=as.matrix(X), y=trainData$logBCF, alpha = 0.5, family = "gaussian", typ
e.measure = c("mse"), nfolds=10)
opt_lambda = cv.elastic$lambda.min

print(paste("Optimal lambda is: ", opt_lambda))
```

```
## [1] "Optimal lambda is:  0.0207662039532022"
```

Answer

See above the lambda value that minimizes the cross-validation error using 10 fold CV is 0.020766.

b. List the coefficient values at the optimal lambda. How many variables were selected? How do these variables compare to those from Lasso in Question 5?

```
set.seed(100)

# Extract the coefficients at the optimal lambda
elastic.lm = glmnet(x=as.matrix(X), y=trainData$logBCF, alpha=0.5, nlambda=100)
coef(elastic.lm, s=cv.elastic$lambda.min)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  0.04903516
## nHM          0.12397290
## piPC09       0.03470891
## PCD          0.03060034
## X2Av         .
## MLOGP        0.51776470
## ON1V         0.08901088
## N.072       -0.05236840
## B02.C.N.    -0.14155538
## F04.C.O.    -0.02420217
```

```
# elastic.lm = glmnet(x=as.matrix(X), y=trainData$logBCF, alpha=0.5, lambda=cv.elastic$lambda.mi
n)
# coef(elastic.lm)
```

Answer

See the coefficients values above. As indicated above, the optimal lambda is 0.020766. 8 variables were selected using the elastic net (and an intercept) and they are the same variables selected by LASSO. Looking at the coefficients values, the are very close to the values given by LASSO.

Question 7: Model comparison

a. Predict *logBCF* for each of the rows in the test data using the full model, and the models found using backward stepwise regression with BIC, ridge regression, lasso regression, and elastic net.

```
set.seed(100)

# Full model (model1)
pred_full = predict(model1, newdata=testData)

# Backward stepwise (model4)
pred_backward = predict(model4, newdata=testData)

# Ridge regression
pred_ridge = predict(ridge.lm, newx=as.matrix(testData[,-10]), s=cv.ridge$lambda.min)

# LASSO regression
# Retrain LASSO model with selected variables
lasso_model = lm(logBCF ~ nHM+piPC09+PCD+MLOGP+ON1V+N.072+B02.C.N.+F04.C.O., data=trainData)
pred_lasso = predict(lasso_model, newdata=testData)
#pred_lasso = predict(lasso.lm, newx=as.matrix(testData[,-10]), s=cv.lasso$lambda.min)

# Elastic Net
pred_elastic = predict(elastic.lm, newx=as.matrix(testData[,-10]), s=cv.elastic$lambda.min)


print("First 10 predicted values:")
```

```
## [1] "First 10 predicted values:"
```

```
data.frame("Full"=pred_full[1:10], "Backward"=pred_backward[1:10], "Ridge"=pred_ridge[1:10], "LA
SSO"=pred_lasso[1:10], "Elastic_Net"=pred_elastic[1:10])
```

```
##          Full  Backward     Ridge     LASSO Elastic_Net
## 714 2.4464790 2.4249160 2.4548777 2.4463922   2.4415065
## 503 4.3337590 4.3531671 4.2344245 4.3334153   4.2964508
## 358 3.2668917 3.2741920 3.2231662 3.2672536   3.2526382
## 624 1.6647698 1.2971754 1.7340940 1.6584462   1.6067741
## 718 1.9553625 2.0013985 1.9939667 1.9567983   1.9394645
## 470 4.3332777 4.3554698 4.2332224 4.3328883   4.2971705
## 516 1.1733488 1.2318810 1.1688682 1.1727659   1.1799112
## 98  0.7920842 0.9745210 0.7777016 0.7937128   0.8925199
## 7   0.9760207 1.0449449 0.9882619 0.9752417   0.9897446
## 183 0.8965352 0.7565987 0.9633564 0.8940798   0.8913939
```

b. Compare the predictions using mean squared prediction error. Which model performed the best?

```
set.seed(100)

full_MSPE = mean((pred_full-testData$logBCF)^2)
backward_MSPE = mean((pred_backward-testData$logBCF)^2)
ridge_MSPE = mean((pred_ridge-testData$logBCF)^2)
lasso_MSPE = mean((pred_lasso-testData$logBCF)^2)
elastic_MSPE = mean((pred_elastic-testData$logBCF)^2)

print(paste("full model MSPE: ", full_MSPE))
```

```
## [1] "full model MSPE:  0.583929593178489"
```

```
print(paste("backward model MSPE: ", backward_MSPE))
```

```
## [1] "backward model MSPE:  0.57421978017202"
```

```
print(paste("ridge model MSPE: ", ridge_MSPE))
```

```
## [1] "ridge model MSPE:  0.587783468737245"
```

```
print(paste("lasso model MSPE: ", lasso_MSPE))
```

```
## [1] "lasso model MSPE:  0.583547038087542"
```

```
print(paste("elastic model MSPE: ", elastic_MSPE))
```

```
## [1] "elastic model MSPE:  0.57827500474914"
```

Answer

See above the models MSPE. The backward stepwise model performs best with the lowest MSPE of 0.5742.

c. Provide a table listing each method described in Question 7a and the variables selected by each method (see Lesson 5.8 for an example). Which variables were selected consistently?

Answer

	Backward Stepwise	**Ridge**	**Lasso**	**Elastic Net**
nHM	x	x	x	x
piPC09	x	x	x	x
PCD		x	x	x
X2AV		x		
MLOGP	x	x	x	x
ON1V		x	x	x
N.072		x	x	x
B02.C.N.		x	x	x
F04.C.O.	x	x	x	x

The variables that were selected consistently are nHM, piPC90, MLOGP and F04.C.O..

Thank You!