Homework 5 Peer Assessment

Fall Semester 2020

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)
library(tidyverse)
library(caret)

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

```r
# 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, ]
```

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.

```r
model1 <- lm(logBCF~nHM+piPC09+PCD+X2Av+MLOGP+ON1V+N.072+B02.C.N.+F04.C.O.,data=trainData)
summary(model1)
```

```
##
## Call:
## lm(formula = logBCF ~ nHM + piPC09 + PCD + X2Av + MLOGP + ON1V +
##     N.072 + B02.C.N. + F04.C.O., 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
```

(b) Which regression coefficients are significant at the 95% confidence level? At the 99% confidence level?
95% Significance: ON1V, BO2.C.N. 99% Significance: MLOGP, F04.C.O., nHM

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

```r
set.seed(100)
#K-Fold cross validation
loocv_control <- trainControl(method = "LOOCV")
loocv_model <- train(logBCF~nHM+piPC09+PCD+X2Av+MLOGP+ON1V+N.072+B02.C.N.+F04.C.O., data=trainData, metl
print(loocv_model)
```

```
## Linear Regression
##
## 624 samples
##   9 predictor
##
## No pre-processing
## Resampling: Leave-One-Out Cross-Validation
## Summary of sample sizes: 623, 623, 623, 623, 623, 623, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.8080762  0.6512869  0.6257152
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```r
set.seed(100)
#K-Fold cross validation
cv_control <- trainControl(method = "cv", number = 10)
k_fold_model <- train(logBCF~nHM+piPC09+PCD+X2Av+MLOGP+ON1V+N.072+B02.C.N.+F04.C.O., data=trainData, me
print(k_fold_model)
```

```
## Linear Regression
##
## 624 samples
##   9 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 561, 562, 560, 564, 563, 562, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.8046493  0.6621643  0.6257615
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

(d) What are the Mallow's Cp, AIC, and BIC criterion values for this model?

```r
set.seed(100)
n=nrow(trainData)
c(Cp(model1, S2=0.7957^2),
  AIC(model1, k=2), AIC(model1,k=log(n)))
```

```
## [1]    9.932584 1497.476533 1546.274187
```

Mallow's CP = 9.932584 AIC= 1497.476533 BIC= 1546.274187

(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~MLOGP+F04.C.O.+nHM, data=trainData)
anova(model2,model1)
```

```
## Analysis of Variance Table
##
## Model 1: logBCF ~ MLOGP + F04.C.O. + nHM
## 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
```

H0: Reduced Model HA: Full Model

Based on our results, we see a p-value for the partial F-test of 0.00523 which is $<$ alpha level 0.01. This means that we reject the null hypothesis, meaning that the full model includes additional variables which perform better predictions against the target variable logBCF. However, we don't know which additional variables provide more explanatory power, and therefore we will need to use variable selection.

However, it is not good practice to select variables based on statistical significance due to several reasons: 1) Multicollinearity: The significance of the coefficient could be inflated due to being correlated with another significant feature. Therefore, after addressing multicollinearity, the significance of the coefficient could change drastically, and other coefficients which were previously insigificant could now be deemed significant. 2) Controlling Factors: Another reason is that the variable could be a controlling factor, which are features that you do not want to remove as part of the model. So even if a feature that is a controlling is not statistically signifcant, we still want to include it in the 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.

```
library(dplyr)
set.seed(100)
out = leaps(trainData[,-c(10)], trainData[,c(10)], method = "Cp",nbest=1)
matrix <- cbind(as.matrix(out$which),out$Cp)
colnames_list <- c("MLOGP","nHM MLOGP","nHM piPC09 MLOGP","nHM piPC09 MLOGP F04.C.O.","nHM piPC09 MLOGP
matrix <- as.data.frame(matrix)
matrix %>%
    mutate(FeaturesUsed = strsplit(as.character(colnames_list), ",")) %>%
    unnest(FeaturesUsed) %>%
  rename(
```

```
    Mallows_CP = V10
  )
```

```
## # A tibble: 9 x 11
##     `1`   `2`   `3`   `4`   `5`   `6`   `7`   `8`   `9` Mallows_CP FeaturesUsed
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>      <dbl> <chr>
## 1     0     0     0     0     1     0     0     0     0      58.6  MLOGP
## 2     1     0     0     0     1     0     0     0     0      17.7  nHM MLOGP
## 3     1     1     0     0     1     0     0     0     0      15.2  nHM piPC09 M~
## 4     1     1     0     0     1     0     0     0     1       9.50 nHM piPC09 M~
## 5     1     1     0     0     1     0     0     1     1       7.24 nHM piPC09 M~
## 6     1     1     0     0     1     1     0     1     1       6.12 nHM piPC09 M~
## 7     1     1     0     0     1     1     1     1     1       6.83 nHM piPC09 M~
## 8     1     1     1     0     1     1     1     1     1       8.02 All Predicto~
## 9     1     1     1     1     1     1     1     1     1      10    All Predicto~
```

Based on the lectures, there are 2^p (where p is the number of predictors) possible submodels. In our case we have 2^9 = 512 different submodels that can be built. However in our case, we're just looking for the best model out of each combination of predictors, therefore we will set nbest = 1.

(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
```

They are 6 variables in the model with the lowest Mallow's CP and they are: nHM piPC09 MLOGP ON1V B02.C.N. F04.C.O.

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 Stepwise Regression
minimum = lm(logBCF ~ 1, data=trainData)
step(model1, scope=list(lower=minimum, upper=model1), direction="backward",k=log(n))
```

```
## 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

##
## Call:
## lm(formula = logBCF ~ nHM + piPC09 + MLOGP + F04.C.O., data = trainData)
##
## Coefficients:
## (Intercept)          nHM       piPC09        MLOGP     F04.C.O.
##   -0.008695     0.114029     0.041119     0.566473    -0.022104
```

We can see from the output of the backward stepwise regression that the final model chosen with the lowest AIC of -253.6 contains the 4 variables: nHM + piPC09 + MLOGP + F04.C.O. We use k = log(n) to perform BIC instead of AIC in our case.

Now we're going to use this result to create model4:

```
model4 <- lm(formula = logBCF ~ nHM + piPC09 + MLOGP + F04.C.O., data = trainData)
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
```

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

There are 4 variables in model4. In our case, all coefficients are significant at the 99% level.

(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)
step(lm(logBCF~1, data=trainData), scope=list(lower=lm(logBCF~1, data=trainData), upper=lm(logBCF~., da
```

```
## 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

##
## Call:
## lm(formula = logBCF ~ MLOGP + nHM + piPC09 + F04.C.O. + B02.C.N. +
##     ON1V, data = trainData)
##
## Coefficients:
## (Intercept)        MLOGP          nHM       piPC09     F04.C.O.     B02.C.N.
##     0.03578      0.52852      0.12409      0.04217     -0.02864     -0.16020
##        ON1V
##     0.09810
```

```
model5 <- lm(formula = logBCF ~ MLOGP + nHM + piPC09 + F04.C.O. + B02.C.N. +
    ON1V, data = trainData)
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
```

```
c(Cp(model1, S2=0.7957^2),
  AIC(model1, k=2), AIC(model1,k=log(n)))
```

```
## [1]    9.932584 1497.476533 1546.274187
```

```
c(Cp(model3, S2=0.7951^2),
  AIC(model3, k=2), AIC(model3,k=log(n)))
```

```
## [1]    6.978644 1493.623474 1529.112677
```

```
c(Cp(model4, S2=0.7985^2),
  AIC(model4, k=2), AIC(model4,k=log(n)))
```

```
## [1]    5.062064 1497.052364 1523.669266
```

(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?

	Model1	Model3	Model4
R2	0.6672	0.666	0.662
CP	9.932584	6.978644	5.062064
AIC	1497.476533	1493.623474	1497.052364
BIC	1546.274187	1529.112677	1523.669266

Based on our results we can see that model1 contains the best R^2 value by a fraction, but the R2 values for the other two models are only smaller by a fraction, and therefore is not a determining factor when selecting the preferred model.

Based on the remaining 3 criteria, we can see that model4 has the lowest CP and BIC value, while model3 has the lowest AIC value. From this we can determine that model4 is the preferred model since it contains lower values for 2 of the 3 remaining metrics compared to model4.

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)
#Find the lambda value that minimizes the CV error
ridge.cv = cv.glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=0, nfolds=10)

## Fit ridge model with 100 values for lambda
ridge = glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=0, nlambda=100)
ridge_final = glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=0, lambda=ridge.cv$lambda.m
```

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

```
set.seed(100)
## Extract coefficients at optimal lambda
coef(ridge, s=ridge.cv$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
```

(c) How many variables were selected? Give an explanation for this number. Ridge regression will utilize all variables but will not remove any predictors compared to Lasso regression. This is why it makes sense that there's still 9 coefficients in our output. What this model will do is penalize the terms and provide a value close to 0 but not equal to 0 for the coefficients that are not significant.

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.

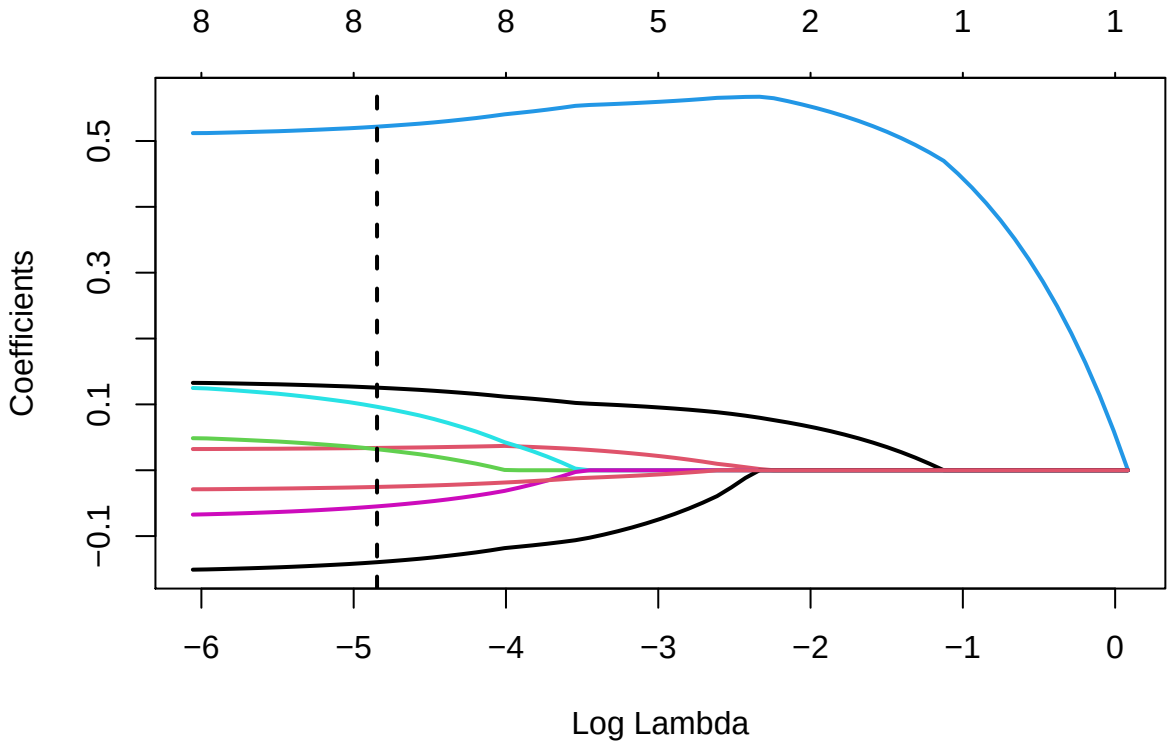
```
set.seed(100)
## LASSO
## alpha=1 for lasso
## Find the optimal lambda using 10-fold CV
lasso.cv = cv.glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=1, nfolds=10)

## Fit lasso model with 100 values for lambda
lasso = glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=1, nlambda=100)
lasso_final = glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=1, lambda=lasso.cv$lambda.m
## Extract coefficients at optimal lambda
coef(lasso, s=lasso.cv$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
```

(b) Plot the regression coefficient path.

```r
set.seed(100)
## Plot coefficient paths
plot(lasso, xvar="lambda", lwd=2)
abline(v=log(lasso.cv$lambda.min), col='black', lty=2, lwd=2)
```



(c) How many variables were selected? Which are they?

From our result the following predictors were selected: "nHM"
"piPC09"
"PCD"
"X2Av"
"MLOGP"
"ON1V"
"N.072"
"B02.C.N." "F04.C.O."

The lasso model selected all variables except for the X2Av variable.

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.

```r
set.seed(100)
## Elastic Net
## alpha=0.5 anywhere between 0 and 1
## Find the optimal lambda using 10-fold CV
enet.cv = cv.glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=0.5, nfolds=10)

## Fit enet model with 100 values for lambda
enet = glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=0.5, nlambda=100)
enet_final = glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=1, lambda=enet.cv$lambda.min
```

(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?

```r
set.seed(100)
## Extract coefficients at optimal lambda
coef(enet, s=enet.cv$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
```

```r
enet_final = glmnet(as.matrix(trainData[,-c(10)]), trainData$logBCF, alpha=0.5, lambda=enet.cv$lambda.m
```

We can see the same result at the previous Lasso model except the coefficient values vary slightly.

Same as Lasso, the X2Av variable was not selected by the model.

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.

```r
set.seed(100)
pred_model1 <- predict(model1, testData)
pred_model_backwardstep <- predict(model4, testData)
x <- testData %>% select(nHM,piPC09, PCD, X2Av, MLOGP, ON1V, N.072, B02.C.N., F04.C.O.) %>% data.matrix
pred_model_ridge <- predict(ridge_final, s = ridge.cv$lambda.min, newx = x)
pred_model_lasso <- predict(lasso_final, s = lasso.cv$lambda.min, newx = x)
pred_model_enet <- predict(enet_final, s = enet.cv$lambda.min, newx = x)
```

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

```r
set.seed(100)
mean((testData$logBCF - pred_model1^2) ^ 2)
```

```
## [1] 26.54817
```

```r
mean((testData$logBCF - pred_model_backwardstep^2) ^ 2)
```

```
## [1] 26.15979
```

```r
mean((testData$logBCF - pred_model_ridge^2) ^ 2)
```

```
## [1] 24.24939
```

```r
mean((testData$logBCF - pred_model_lasso^2) ^ 2)
```

```
## [1] 25.7561
```

```r
mean((testData$logBCF - pred_model_enet^2) ^ 2)
```

```
## [1] 25.27896
```

Based on our response we can see that the ridge regression performed the best with the lowest MSPE value of 24.24939

(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?

	Backward Step	Ridge	Lasso	ENet
nHM	x	x	x	x
piPC09		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

We can see when looking at the results that the nHM, MLOGP and F04.C.O. variables were selected consistenly by all models.