Feature 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)

Fitting a Full Model

A multiple linear regression with the variable *logBCF* as the response and the other variables as predictors was fit. The summaryt for this model can be seen below:

```
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
```

Using a 95% confidence level, it can be seen that the regression coefficients associated with the variables *nHM*, *MLOGP*, *ON1V*, *B02.C.N.*, and *F04.C.O.* are significant. Using a 99% confidence level, it can be seen that the regression coefficients associated with the variables *nHM*, *MLOGP*, and *F04.C.O.* are significant.

For this model, Mallow's Cp is 10, AIC is 1497.477, and BIC is 1546.274.

```
set.seed(100)
Cp(model1,S2=summary(model1)$sigma^2)[1]
```

```
## [1] 10
```

```
AIC(model1, k=2)
```

```
## [1] 1497.477
```

```
AIC(model1, k=log(nrow(trainData)))
```

```
## [1] 1546.274
```

A new model with only the variables which coefficients were found to be statistically significant at the 99% confident level was fit.

```
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
```

As it can be seen above, the p-value associated with the Partial F-test comparing the reduced model (*model2*) and the full model (*model1*) is 0.00523, which is significant at a 99% confidence level and therefore it can be concluded that some additional predictors in *model1* are significantly associated with the response. Given this understanding, retaining the full model (*model1*) would be preferred. It should be noted however, that selecting variables based on the statistical significance of individual coefficients is bad practice as it ignores the possible presence of multicollinearity between the predictor variables.

Model Selection

Mallow's Cp

A model was fit with the lowest Mallow's Cp value. The model summary can be seen below:

```
set.seed(100)
out = leaps(trainData[1:9], trainData$logBCF , method='Cp')
nrow(out$which)
```

```
## [1] 79
```

```
out = leaps(trainData[1:9], 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))
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
```

```
names(trainData[c(as.matrix(out$which)[best.model,])][-7])
```

```
## [1] "nHM"      "piPC09"   "MLOGP"    "ON1V"     "B02.C.N." "F04.C.O."
```

```
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
```

There are 6 variables included in the model with the lowest Mallow's Cp value, which are *nHM*, *piPC09*, *MLOGP*, *ON1V"*, *B02.C.N.*, and *F04.C.O.*. The summary for the model using these variables can be seen above.

Backwards Stepwise Regression

Backward stepwise regression was performed using BIC. The minimum model was a model with only an intercept, and the full model to be *model1*. The model summary can be seen below:

```r
set.seed(100)
reduced = lm(logBCF~1, data=trainData)
model4 = step(model1, scope=list(lower=reduced,
                                upper=model1),
              direction='backward',
              k=log(nrow(trainData)))
```

```
## 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
```

In *model4*, there are four variables, all of which are significant at the 99% confidence level.

Forwards Stepwise Regression

Forward stepwise selection was performed with AIC. The minimum model to be the model with only an intercept, and the full model to be *model1*. The model summary can be seen below:

```
set.seed(100)
model5 = step(reduced, scope=list(lower=reduced,
                                  upper=model1),
              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
```

model5 has 6 variables while *model4* only has 4 variables, with the variables *B02.C.N.* and *ON1V* being the difference between these two models.

Model Comparison

The model with the best Adjusted R^2 and AIC value is *model3*, while *model4* has the best BIC value. This discrepancy makes sense, however, since the BIC penalizes the most for model complexity and *model4* is the simplest model of those tested. When this complexity is not penalized more, it can be seen that *model3* accounts for the most variability in the response explained by the model. Notably, the Mallow's Cp value for all three models were the same distance from the number of variables in each respective model, which does not provide much information regarding the best model by this criteria.

```
set.seed(100)
paste('Adj R^2:','model1:',
  summary(model1)$adj.r.squared,'model3:',
  summary(model3)$adj.r.squared,'model4:',
  summary(model4)$adj.r.squared)
```

```
## [1] "Adj R^2: model1: 0.662302680014831 model3: 0.662786417109309 model4: 0.659850397065386"
```

```
paste("Mallow's Cp:",'model1:',
  Cp(model1, S2=summary(model1)$sigma^2),'model3:',
  Cp(model3, S2=summary(model3)$sigma^2),'model4:',
  Cp(model4, S2=summary(model4)$sigma^2))
```

```
## [1] "Mallow’s Cp: model1: 10.0000000000001 model3: 7.00000000000011 model4: 5"
```

```
paste("# of Variables:",'model1:',
  length(model1$coefficients)-1,
  length(model3$coefficients)-1,
  length(model4$coefficients)-1)
```

```
## [1] "# of Variables: model1: 9 6 4"
```

```
paste("AIC:",'model1:',
  AIC(model1, k=2),'model3:',
  AIC(model3, k=2),'model4:',
  AIC(model4, k=2))
```

```
## [1] "AIC: model1: 1497.47653274345 model3: 1493.62347413487 model4: 1497.05236356401"
```

```
paste("BIC:",'model1:',
  AIC(model1, k=log(nrow(trainData))),'model3:',
  AIC(model3, k=log(nrow(trainData))),'model4:',
  AIC(model4, k=log(nrow(trainData))))
```

```
## [1] "BIC: model1: 1546.27418679551 model3: 1529.11267708182 model4: 1523.66926577422"
```

The model with the best Adjusted R^2 and AIC value is *model3*, while *model4* has the best BIC value. This discrepancy makes sense, however, since the BIC penalizes the most for model complexity and *model4* is the simplest model of those tested. When this complexity is not penalized more, it can be seen that *model3* accounts for the most variability in the response explained by the model. Notably, the Mallow's Cp value for all three models were the same distance from the number of variables in each respective model, which does not provide much information regarding the best model by this criteria.

Ridge Regression

Ridge regression was performed on the training set while finding the lambda value that minimizes the cross-validation error using 10 fold CV.

```
set.seed(100)
ridge = cv.glmnet(x=as.matrix(trainData[,-10]),
                  y= trainData$logBCF,
                  alpha=0,
                  nfolds = 10)
ridge$lambda.min
```

```
## [1] 0.108775
```

The value of coefficients at the optimum lambda value can be seen below:

```
set.seed(100)
coef(ridge, ridge$lambda.min)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (Intercept)  0.13841426
## nHM          0.14391877
## piPC09       0.03735762
## PCD          0.08235334
## X2Av        -0.06901352
## MLOGP        0.44403655
## ON1V         0.15770114
## N.072       -0.09683534
## B02.C.N.    -0.20919397
## F04.C.O.    -0.03177144
```

There were 9 variables selected, which is all the predictors possible. This is the expected result, however, as Ridge Regression is not used for variable selection as it only regularizes the regression coefficients. Given this, it would not be expected that any variables would not be selected.

Lasso Regression

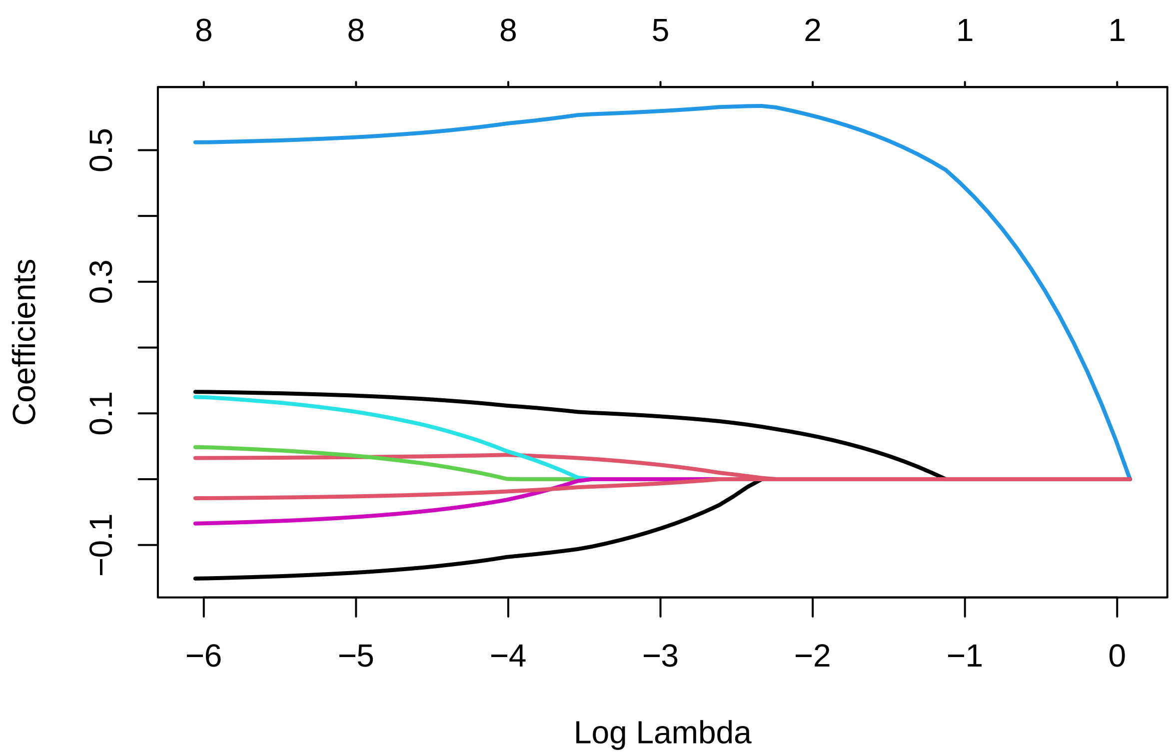
Lasso regression was performed on the training set while finding the lambda value that minimizes the cross-validation error using 10 fold CV.

```
set.seed(100)
lasso.cv = cv.glmnet(x=as.matrix(trainData[,-10]),
                     y= trainData$logBCF,
                     alpha=1,
                     nfolds = 10)
lasso.cv$lambda.min
```

```
## [1] 0.007854436
```

A plot the regression coefficient path can be seen below:

```r
set.seed(100)
lasso = glmnet(x=as.matrix(trainData[,-10]),
               y= trainData$logBCF,
               alpha=1,
               nlambda = 100)
plot(lasso,xvar='lambda',lwd=2)
```



There were 8 variables selected, which were *NHM*, *piPC09*, *PCD*, *MLOGP*, *ON1V*, *N.072*, *B02.C.N.*, and *F04.C.O.*.

```r
set.seed(100)
coef(lasso.cv,lasso.cv$lambda.min)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (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
```

Elastic Net

Elastic Net regression was performed on the training set while finding the lambda value that minimizes the cross-validation error using 10 fold CV.

```
set.seed(100)
elastic = cv.glmnet(x=as.matrix(trainData[,-10]),
                    y= trainData$logBCF,
                    alpha=0.5,
                    nfolds = 10)
elastic$lambda.min
```

```
## [1] 0.0207662
```

There were 8 variables selected again, which were the same as from the LASSO model.

```
set.seed(100)
coef(elastic,elastic$lambda.min)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (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
```

Model comparison

Predictions were made on the test data set and the performance of each model was compared using mean squared error. The results can be seen below:

```
set.seed(100)
mean((testData[[10]]-pred.1)^2)
```

```
## [1] 0.5839296
```

```
mean((testData[[10]]-pred.4)^2)
```

```
## [1] 0.5742198
```

```
mean((testData[[10]]-pred.ridge[,1])^2)
```

```
## [1] 0.5877835
```

```r
mean((testData[[10]]-pred.lasso[,1])^2)
```

```
## [1] 0.5790832
```

```r
mean((testData[[10]]-pred.elastic[,1])^2)
```

```
## [1] 0.578275
```

According to the MSPE calcualted on the test data set, model using backward stepwise regression with BIC (*model4*) performed the best.