STA442 Assignment 2

Alexander Tran

3/10/2023

```
library(tidyverse)
library(glmnet)
```

Question 1

a)

```
set.seed(1006314089)

# Training Set
X_train <- matrix(nrow = 100, ncol = 50)
for (i in 1:50) {
  X_train[,i] <- rnorm(100, 0, 1)
}

epsilon_train <- rnorm(100, 0, 1)

B <- matrix(nrow = 50, ncol = 1)
for (i in 1:20) {
  B[i] <- runif(1, 0.5, 1.5)
}
for (i in 21:50) {
  B[i] <- runif(1, 0.2, 0.4)
}

Y_train <- matrix(nrow = 100, ncol = 1)
for (i in 1:100) {
  Y_train[i] <- sum(B*X_train[i,]) + epsilon_train[i]
}

# Test Set
X_test <- matrix(nrow = 1000, ncol = 50)
for (i in 1:50) {
  X_test[,i] <- rnorm(1000, 0, 1)
}

epsilon_test <- rnorm(1000, 0, 1)

Y_test <- matrix(nrow = 1000, ncol = 1)
for (i in 1:1000) {
  Y_test[i] <- sum(B*X_test[i,]) + epsilon_test[i]
}
```

b)

```
Y_train_df <- as.data.frame(Y_train) %>% rename(Y = V1)
X_train_df <- as.data.frame(X_train)
Y_test_df <- as.data.frame(Y_test)
X_test_df <- as.data.frame(X_test)

train <- cbind(Y_train_df, X_train_df)
lin_reg <- lm(Y ~ ., data = train)

error_lin <- sum((Y_test - predict(lin_reg, newdata = as.data.frame(X_test_df)))**2)/1000
```

The prediction error, calculated as $\frac{1}{1000}\sum_{i=1}^{1000}(y_i - \hat{y}_i)^2$ is equal to 2.428.

c)

```
set.seed(1006314089)

ridge_cv <- cv.glmnet(x = X_train, y = Y_train, alpha = 0)
ridge_reg <- glmnet(x = X_train, y = Y_train, alpha = 0, lambda = ridge_cv$lambda.1se)

error_ridge <- sum((Y_test - predict(ridge_reg, newx = X_test))**2)/1000
```

The prediction error, calculated as $\frac{1}{1000}\sum_{i=1}^{1000}(y_i - \hat{y}_i)^2$ is equal to 2.401.

d)

```
set.seed(1006314089)

lasso_cv <- cv.glmnet(x = X_train, y = Y_train, alpha = 1)
lasso_reg <- glmnet(x = X_train, y = Y_train, alpha = 1, lambda = lasso_cv$lambda.1se)

error_lasso <- sum((Y_test - predict(lasso_reg, newx = X_test))**2)/1000
```

The prediction error, calculated as $\frac{1}{1000}\sum_{i=1}^{1000}(y_i - \hat{y}_i)^2$ is equal to 2.337.

e) The LASSO method used in (d) provides the lowest prediction error on the test set. This is because LASSO optimizes driving small weight estimates to 0 in addition to driving down large weights. This results in a model with less flexibility and greater generalizability, compared to simple linear or ridge regression, given that we already expect overfitting and large variance due to a small training set size relative to the number of predictors.

f)

```r
set.seed(1006314089)

# Training Set
X_train2 <- matrix(nrow = 10000, ncol = 50)
for (i in 1:50) {
  X_train2[,i] <- rnorm(10000, 0, 1)
}

epsilon_train2 <- rnorm(10000, 0, 1)

B <- matrix(nrow = 50, ncol = 1)
for (i in 1:20) {
  B[i] <- runif(1, 0.5, 1.5)
}
for (i in 21:50) {
  B[i] <- runif(1, 0.2, 0.4)
}

Y_train2 <- matrix(nrow = 10000, ncol = 1)
for (i in 1:10000) {
  Y_train2[i] <- sum(B*X_train2[i,]) + epsilon_train2[i]
}

# Test Set
X_test <- matrix(nrow = 1000, ncol = 50)
for (i in 1:50) {
  X_test[,i] <- rnorm(1000, 0, 1)
}

epsilon_test <- rnorm(1000, 0, 1)

Y_test <- matrix(nrow = 1000, ncol = 1)
for (i in 1:1000) {
  Y_test[i] <- sum(B*X_test[i,]) + epsilon_test[i]
}

# Linear Regression
Y_train2_df <- as.data.frame(Y_train2) %>% rename(Y = V1)
X_train2_df <- as.data.frame(X_train2)
Y_test_df <- as.data.frame(Y_test)
X_test_df <- as.data.frame(X_test)

train2 <- cbind(Y_train2_df, X_train2_df)
lin_reg2 <- lm(Y ~ ., data = train2)
error_lin2 <- sum((Y_test - predict(lin_reg2, newdata = as.data.frame(X_test_df)))**2)/1000

# Ridge Regression
ridge_cv2 <- cv.glmnet(x = X_train2, y = Y_train2, alpha = 0)
ridge_reg2 <- glmnet(x = X_train2, y = Y_train2, alpha = 0, lambda = ridge_cv2$lambda.1se)
error_ridge2 <- sum((Y_test - predict(ridge_reg2, newx = X_test))**2)/1000

# LASSO Regression
```

```
lasso_cv2 <- cv.glmnet(x = X_train2, y = Y_train2, alpha = 1)
lasso_reg2 <- glmnet(x = X_train2, y = Y_train2, alpha = 1, lambda = lasso_cv2$lambda.1se)
error_lasso2 <- sum((Y_test - predict(lasso_reg2, newx = X_test))**2)/1000
```

The prediction error of the linear, ridge, and LASSO regressions are now 0.900, 0.955, and 0.925, respectively. It can be seen that increasing the training set size from 100 to 10000 significantly decreased prediction error for all methods, which implies that there was significant overfitting when fitting on the smaller training set. Of the three, the prediction error of the linear regression is now the lowest, having the greatest decrease after increasing the training set size. This is likely due to the fact that the training data was all simulated from $N(0, 1)$, and after increasing the training set size, the sample distribution of X_i became closer to $N(0, 1)$ and the least square estimates were unbiases with low variance. In addition, it became less likely that there were any significantly small or large weight estimates, such that the penalization from LASSO and ridge regression were no longer necessary and only introduced more bias to increase prediction error.

Question 2

```
## If the package is not installed then use ##
## install.packages('NHANES') And install.packages('tidyverse')
library(tidyverse)
library(NHANES)
small.nhanes <- na.omit(NHANES[NHANES$SurveyYr=="2011_12"
& NHANES$Age > 17,c(1,3,4,8:11,13,25,61)])
small.nhanes <- small.nhanes %>%
group_by(ID) %>% filter(row_number()==1)
```

a)

```
set.seed(1006314089)

sample <- small.nhanes[sample(nrow(small.nhanes), 500),]

logit_mod <- glm(SmokeNow ~ . - ID, family = binomial, data = sample)
summary(logit_mod)
```

```
##
## Call:
## glm(formula = SmokeNow ~ . - ID, family = binomial, data = sample)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0517  -0.9172  -0.5576   0.9972   2.1789
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 1.967351   1.297390   1.516   0.1294
## Gendermale                  0.313115   0.226002   1.385   0.1659
## Age                        -0.035379   0.008135  -4.349 1.37e-05 ***
## Race3Black                  0.485549   0.572987   0.847   0.3968
## Race3Hispanic              -0.129257   0.640748  -0.202   0.8401
## Race3Mexican               -0.176552   0.632681  -0.279   0.7802
## Race3White                 -0.015883   0.488082  -0.033   0.9740
## Race3Other                  1.044106   0.748109   1.396   0.1628
## Education9 - 11th Grade     0.820900   0.467735   1.755   0.0793 .
## EducationHigh School        0.531622   0.438926   1.211   0.2258
## EducationSome College       0.414193   0.440059   0.941   0.3466
## EducationCollege Grad      -0.306669   0.487213  -0.629   0.5291
## MaritalStatusLivePartner    0.731067   0.435643   1.678   0.0933 .
## MaritalStatusMarried       -0.166515   0.337609  -0.493   0.6219
## MaritalStatusNeverMarried   0.053260   0.402631   0.132   0.8948
## MaritalStatusSeparated      1.146655   0.721937   1.588   0.1122
## MaritalStatusWidowed        0.108939   0.501493   0.217   0.8280
## HHIncome 5000-9999          0.162164   0.817472   0.198   0.8428
## HHIncome10000-14999        -0.160948   0.704329  -0.229   0.8192
## HHIncome15000-19999        -0.405803   0.727556  -0.558   0.5770
## HHIncome20000-24999        -0.158961   0.728112  -0.218   0.8272
## HHIncome25000-34999        -0.728362   0.716537  -1.017   0.3094
## HHIncome35000-44999        -0.346336   0.747481  -0.463   0.6431
## HHIncome45000-54999        -0.662395   0.791743  -0.837   0.4028
## HHIncome55000-64999         0.405484   0.863994   0.469   0.6388
```

```
## HHIncome65000-74999       -0.714962   0.885037  -0.808   0.4192
## HHIncome75000-99999       -0.873012   0.898731  -0.971   0.3314
## HHIncomemore 99999        -0.245078   0.891151  -0.275   0.7833
## Poverty                   -0.071541   0.141471  -0.506   0.6131
## BPSysAve                  -0.004401   0.007205  -0.611   0.5414
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 685.44  on 499  degrees of freedom
## Residual deviance: 576.16  on 470  degrees of freedom
## AIC: 636.16
##
## Number of Fisher Scoring iterations: 4
```

Based on the output above, most predictors are not statistically significant. Only variables *Age*, *Education = 9 - 11th Grade*, and *MaritalStatus = LivePartner* significantly explain the outcome variable, with the latter two only being significant at the 10% level. With this in mind, we should consider removing some predictors from the model.

Regarding interpretation of coefficients, we see that the odds of smoking is multiplied by $e^{-0.035379} = 0.965$ per year of *Age* (a decrease per year), while the odds are higher for *Education = 9 - 11th Grade* and *MaritalStatus = LivePartner* by $e^{0.820900} = 2.273$ and $e^{0.731067} = 2.077$ times compared to their respective baseline groups, *Education = 8th Grade* and *MaritalStatus = Divorced*.

b)

```r
set.seed(1006314089)

aic_step_model <- step(logit_mod, trace = 0, k = 2)
aic_selected <- attr(terms(aic_step_model), "term.labels")
aic_selected
```

```
## [1] "Age"       "Education" "Poverty"
```

```r
bic_step_model <- step(logit_mod, trace = 0, k = log(500))
bic_selected <- attr(terms(bic_step_model), "term.labels")
bic_selected
```

```
## [1] "Age"     "Poverty"
```

```r
elastnet_reg <- cv.glmnet(x = model.matrix(logit_mod), y = as.numeric(sample$SmokeNow) - 1,
                          family = "binomial", alpha = 0.5)
elastnet_selected <- coef(elastnet_reg, s = elastnet_reg$lambda.1se)
elastnet_selected
```

```
## 31 x 1 sparse Matrix of class "dgCMatrix"
##                                    s1
## (Intercept)                 1.110586856
## (Intercept)                 .
## Gendermale                  .
## Age                        -0.024068995
## Race3Black                  .
## Race3Hispanic               .
## Race3Mexican                .
## Race3White                 -0.006834978
## Race3Other                  .
## Education9 - 11th Grade     0.123434550
## EducationHigh School        .
## EducationSome College       .
## EducationCollege Grad      -0.264108139
## MaritalStatusLivePartner    0.202853263
## MaritalStatusMarried       -0.041811348
## MaritalStatusNeverMarried   .
## MaritalStatusSeparated      .
## MaritalStatusWidowed        .
## HHIncome 5000-9999          .
## HHIncome10000-14999         .
## HHIncome15000-19999         .
## HHIncome20000-24999         .
## HHIncome25000-34999         .
## HHIncome35000-44999         .
## HHIncome45000-54999         .
## HHIncome55000-64999         .
## HHIncome65000-74999         .
## HHIncome75000-99999         .
## HHIncomemore 99999          .
## Poverty                    -0.046503019
## BPSysAve                    .
```

```r
elastnet_reg2 <- cv.glmnet(x = model.matrix(logit_mod), y = as.numeric(sample$SmokeNow) - 1,
                           family = "binomial", alpha = 1)
```

```r
elastnet_selected2 <- coef(elastnet_reg, s = elastnet_reg2$lambda.1se)
elastnet_selected2
```

```
## 31 x 1 sparse Matrix of class "dgCMatrix"
##                                    s1
## (Intercept)                1.20473497
## (Intercept)                .
## Gendermale                 .
## Age                       -0.02528439
## Race3Black                 .
## Race3Hispanic              .
## Race3Mexican               .
## Race3White                -0.02858768
## Race3Other                 0.03679893
## Education9 - 11th Grade    0.15514222
## EducationHigh School       .
## EducationSome College      .
## EducationCollege Grad     -0.30196600
## MaritalStatusLivePartner   0.23723479
## MaritalStatusMarried      -0.06249439
## MaritalStatusNeverMarried  .
## MaritalStatusSeparated     0.01298444
## MaritalStatusWidowed       .
## HHIncome 5000-9999         .
## HHIncome10000-14999        .
## HHIncome15000-19999        .
## HHIncome20000-24999        .
## HHIncome25000-34999        .
## HHIncome35000-44999        .
## HHIncome45000-54999        .
## HHIncome55000-64999        .
## HHIncome65000-74999        .
## HHIncome75000-99999        .
## HHIncomemore 99999         .
## Poverty                   -0.05143559
## BPSysAve                   .
```

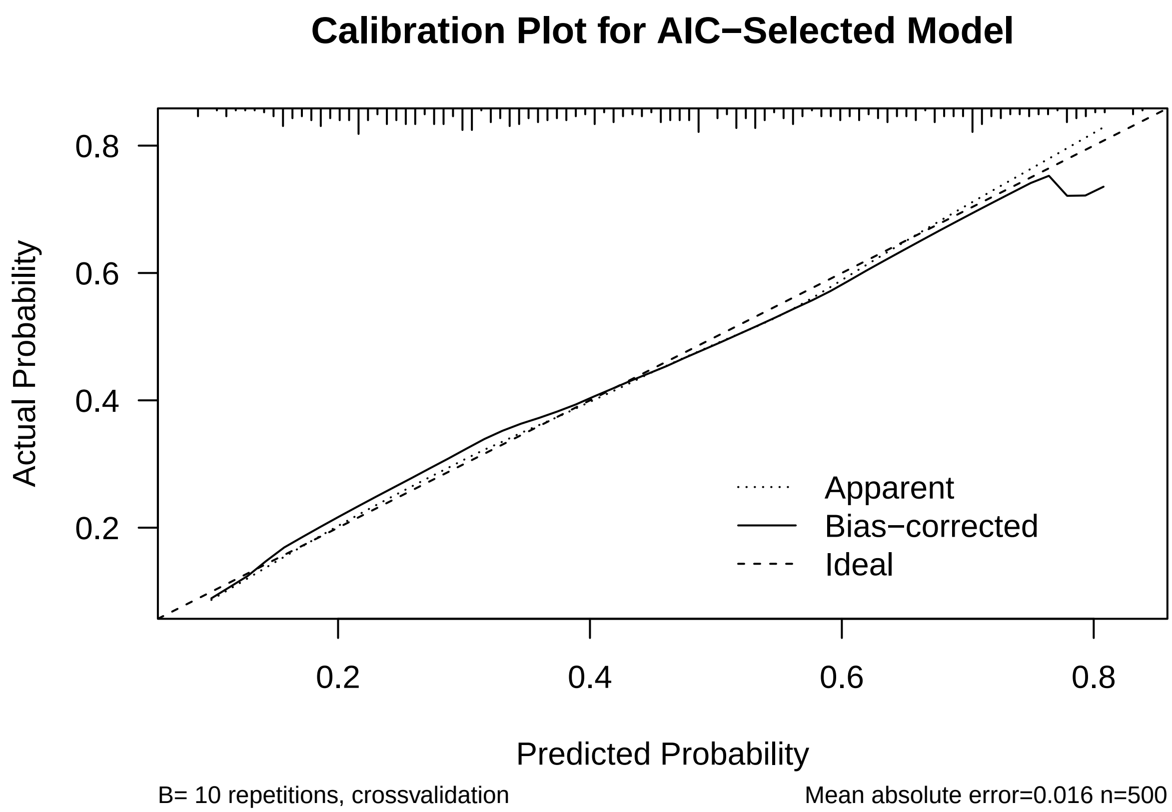
The different selection methods do not all select the same model. While there are some variables selected in all models such as *Age* and *Poverty*, there are others that are only in some models. This is because they all use different criteria to select the best model. AIC and BIC are metrics that penalize a fitted model by a function of the number of parameters and number of observations, while elastic-net penalizes the model in the form a cost function that must be optimized for in regression parameter estimates. While all methods can reduce overfitting, they do so in different ways that may select different models. Also note that elastic net selects the same model for $\alpha = 0.5$ and $\alpha = 1$.

c)

```
library(rms)
```

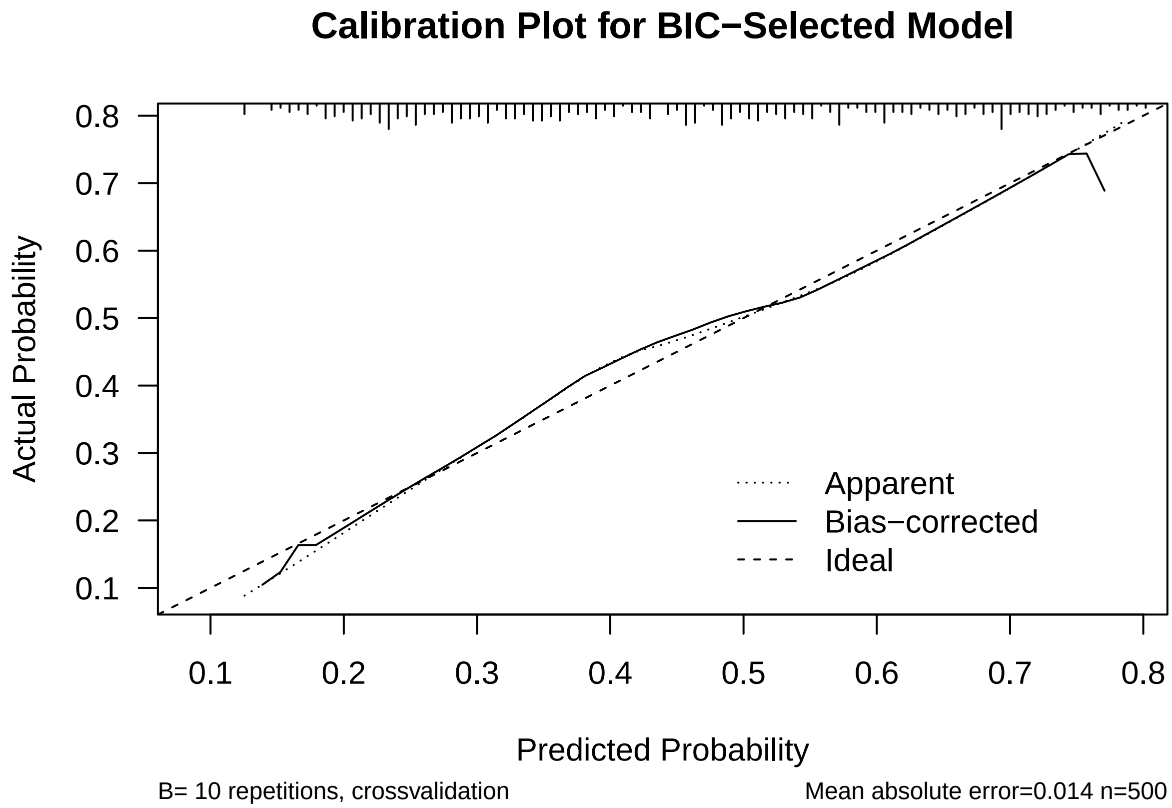
```
## Loading required package: Hmisc

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':
##
##     src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units

## Loading required package: survival

## Loading required package: lattice

## Loading required package: SparseM

##
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':
##
##     backsolve
```

```
set.seed(1006314089)

lrmAIC <- lrm(SmokeNow ~ Age + Education + Poverty, data=sample,
              x =TRUE, y = TRUE, model= T)
cross.calibAIC <- calibrate(lrmAIC, method = "crossvalidation", B = 10)
plot(cross.calibAIC, las=1, xlab = "Predicted Probability", main = "Calibration Plot for AIC-Selected Mo
```



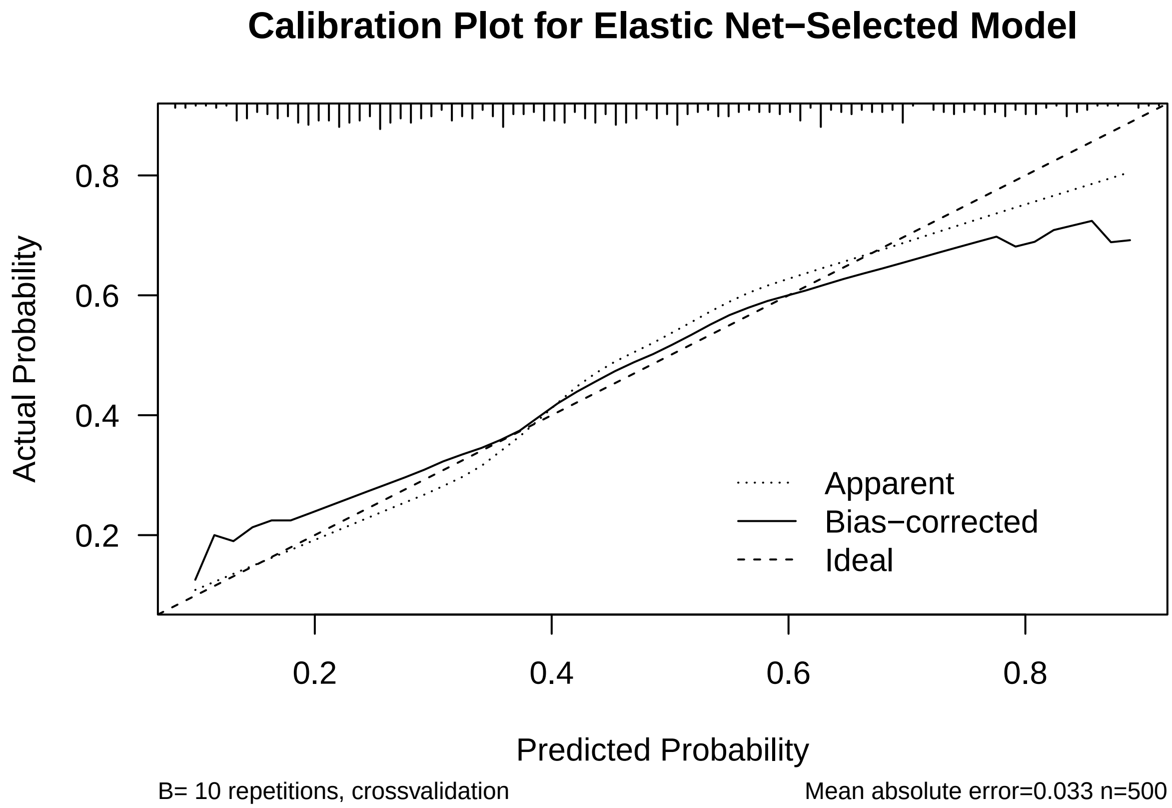

```
##
## n=500   Mean absolute error=0.016   Mean squared error=0.00036
## 0.9 Quantile of absolute error=0.022
```

```
lrmBIC <- lrm(SmokeNow ~ Age + Poverty, data=sample,
              x =TRUE, y = TRUE, model= T)
cross.calibBIC <- calibrate(lrmBIC, method = "crossvalidation", B = 10)
plot(cross.calibBIC, las=1, xlab = "Predicted Probability", main = "Calibration Plot for BIC-Selected Mo
```




```
##
## n=500   Mean absolute error=0.014   Mean squared error=0.00029
## 0.9 Quantile of absolute error=0.03
```

```
lrmEnet <- lrm(SmokeNow ~ Age + Race3 + MaritalStatus + Education + Poverty, data=sample,
              x =TRUE, y = TRUE, model= T)
cross.calibEnet <- calibrate(lrmEnet, method = "crossvalidation", B = 10)
plot(cross.calibEnet, las=1, xlab = "Predicted Probability", main = "Calibration Plot for Elastic Net-S
```




```
##
## n=500   Mean absolute error=0.033   Mean squared error=0.00208
## 0.9 Quantile of absolute error=0.07
```

The calibration plot for AIC-selected model shows that the predicted probabilities are lower than the actual probabilities at lower probabilities (0.10 to 0.40), which indicates that the model is overpredicting cases where people do not smoke. Similarly, the predicted probabilities are higher than the actual probabilities at higher probababilities (0.50 to 0.80), indicating overprediction for those that do smoke. Though, these overpredictions are small and overall, this model seems fairly well calibrated as the bias-corrected line follows the ideal line reasonably closely.

The plot for the BIC-selected model shows that it tends to bias towards predicting smoking at both tails and at middle to higher probabilities (0.55 to 0.75), and biases towards not smoking at lower to middle probabilities (0.25 to 0.50). Overall the plot misses predictions by a larger margin, but on average predicts better than the AIC-selected model, based on the mean absolute error.

The plot for the elastic net selected model shows bias towards not smoking at lower (0.10 to 0.60) probabilities and bias towards smoking at higher (0.6 to 0.9) probabilities, and overall performs the worse out of the three based on the mean absolute error.

d)

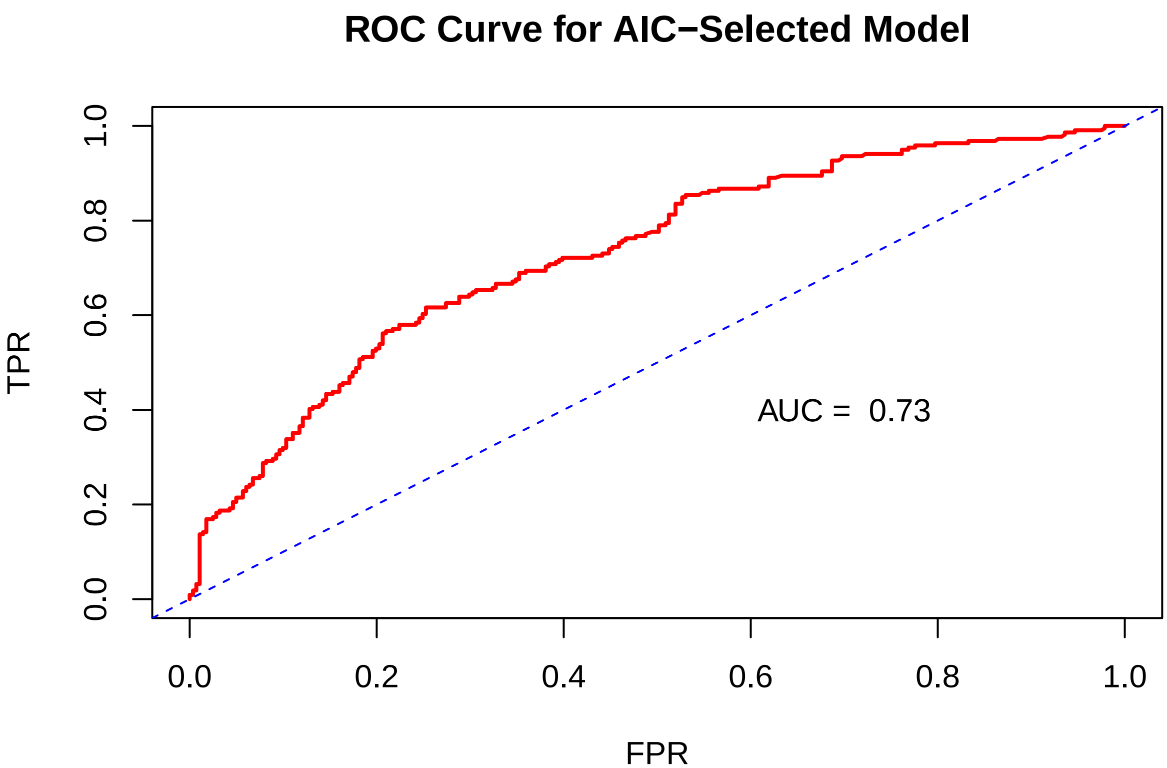
```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
p <- predict(lrmAIC, type = "fitted")
roc_AIC <- roc(sample$SmokeNow ~ p)
```

```
## Setting levels: control = No, case = Yes

## Setting direction: controls < cases
```

```
TPR <- roc_AIC$sensitivities
FPR <- 1 - roc_AIC$specificities
plot(FPR, TPR, xlim = c(0,1), ylim = c(0,1), type = 'l', lty = 1, lwd = 2,col = 'red', main = "ROC Curve
abline(a = 0, b = 1, lty = 2, col = 'blue')
text(0.7,0.4,label = paste("AUC = ", round(auc(roc_AIC),3)))
```

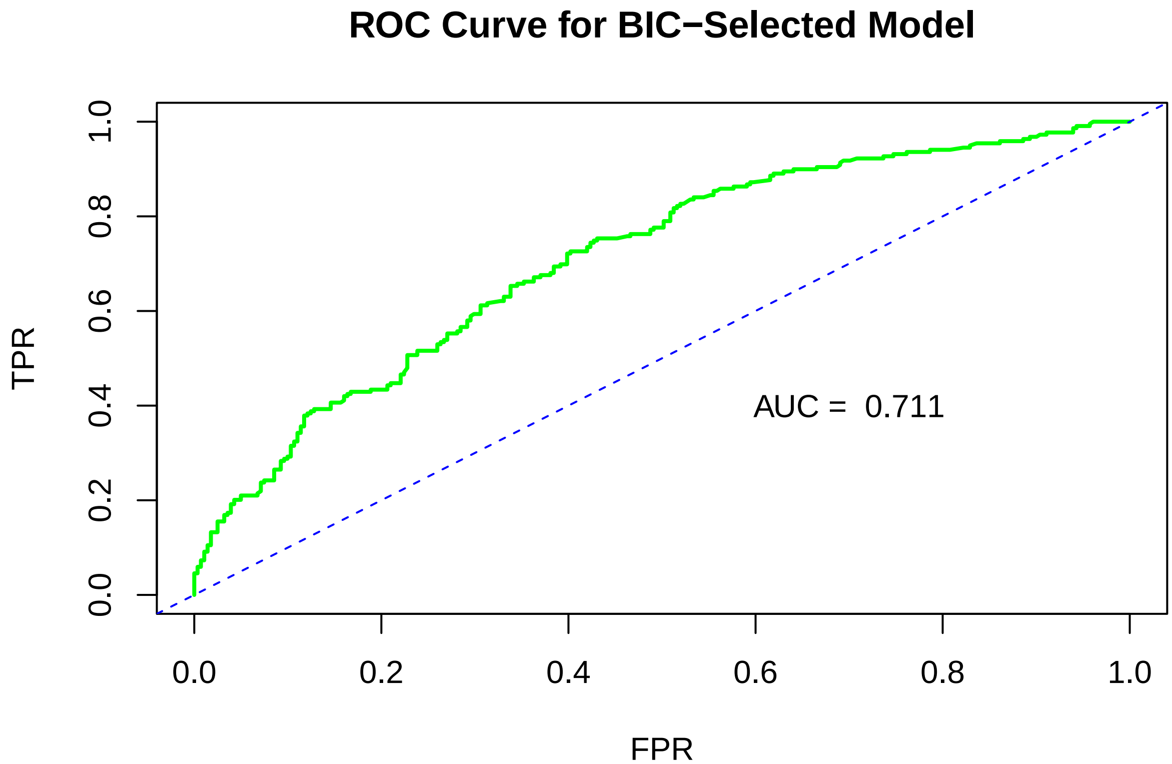



```
p <- predict(lrmBIC, type = "fitted")
roc_BIC <- roc(sample$SmokeNow ~ p)
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
TPR <- roc_BIC$sensitivities
FPR <- 1 - roc_BIC$specificities
```

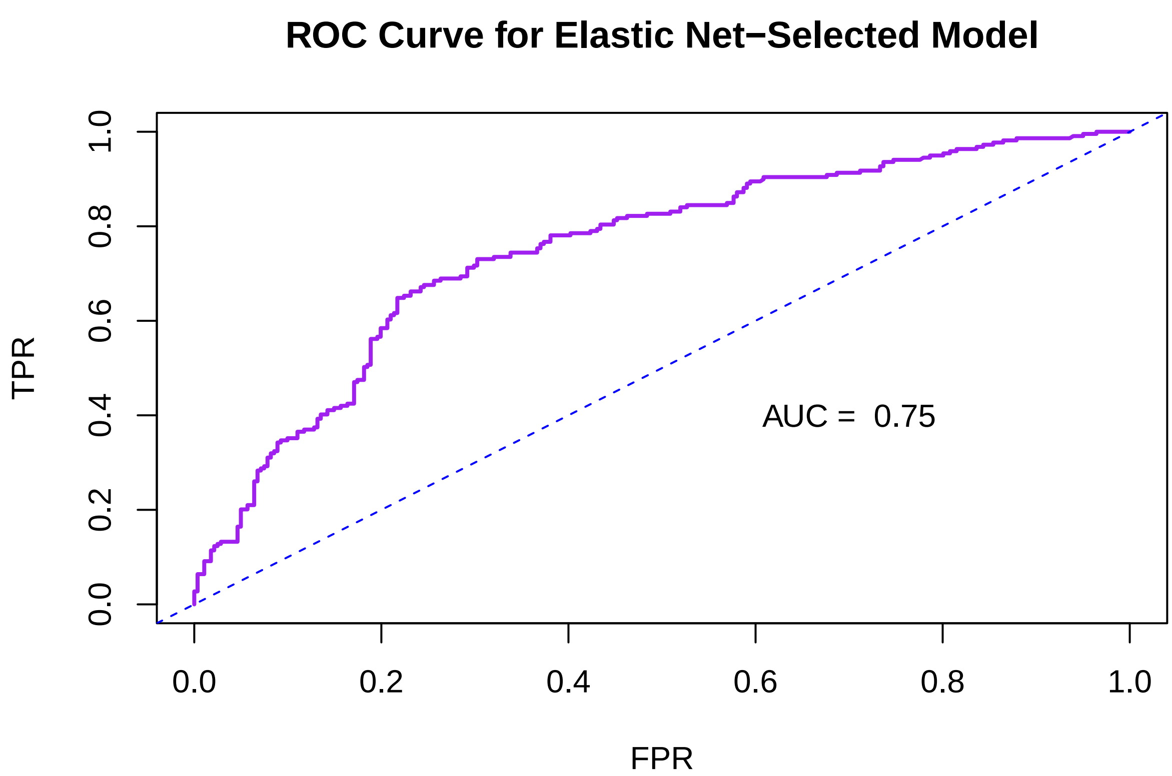
```
plot(FPR, TPR, xlim = c(0,1), ylim = c(0,1), type = 'l', lty = 1, lwd = 2,col = 'green', main = "ROC Cu
abline(a = 0, b = 1, lty = 2, col = 'blue')
text(0.7,0.4,label = paste("AUC = ", round(auc(roc_BIC),3)))
```




```
p <- predict(lrmEnet, type = "fitted")
roc_Enet <- roc(sample$SmokeNow ~ p)
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

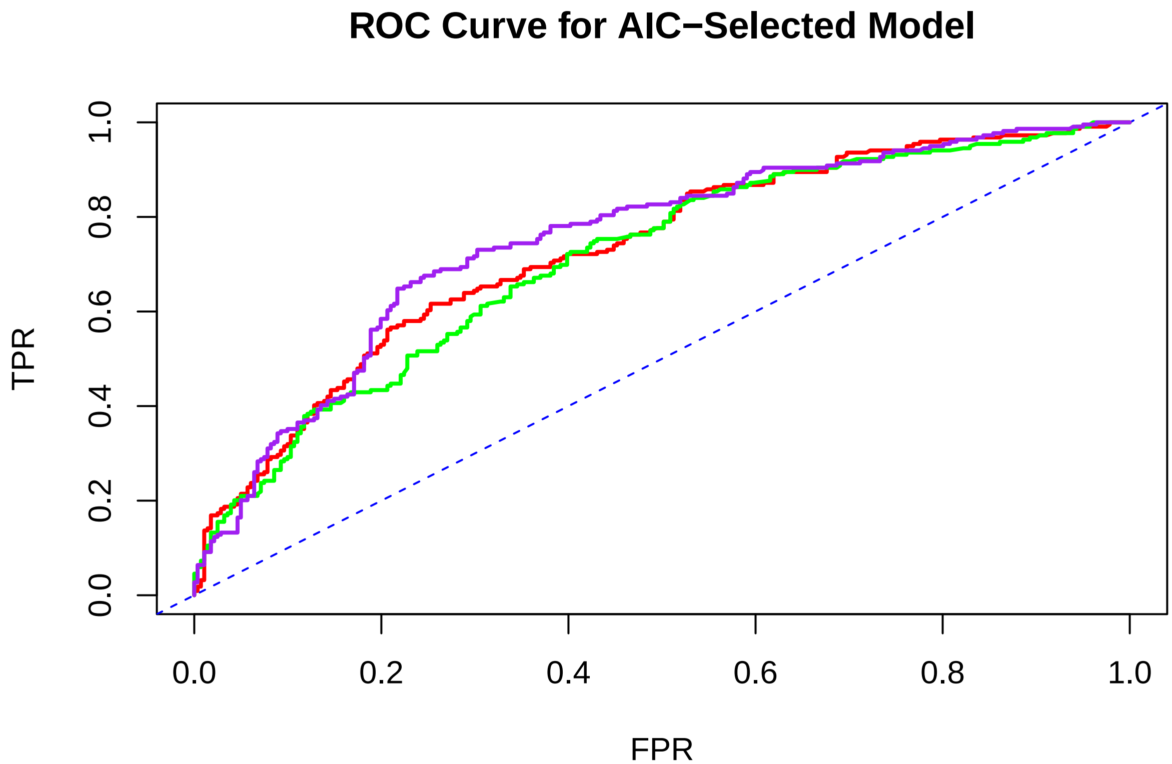
```
TPR <- roc_Enet$sensitivities
FPR <- 1 - roc_Enet$specificities
plot(FPR, TPR, xlim = c(0,1), ylim = c(0,1), type = 'l', lty = 1, lwd = 2,col = 'purple', main = "ROC Cu
abline(a = 0, b = 1, lty = 2, col = 'blue')
text(0.7,0.4,label = paste("AUC = ", round(auc(roc_Enet),3)))
```




```
# Combined Plot
p <- predict(lrmAIC, type = "fitted")
roc_AIC1 <- roc(sample$SmokeNow ~ p)
```

```
## Setting levels: control = No, case = Yes
## Setting direction: controls < cases
```

```
TPR <- roc_AIC$sensitivities
FPR <- 1 - roc_AIC$specificities
plot(FPR, TPR, xlim = c(0,1), ylim = c(0,1), type = 'l', lty = 1, lwd = 2,col = 'red', main = "ROC Curve
abline(a = 0, b = 1, lty = 2, col = 'blue')
lines(1 - roc_BIC$specificities, roc_BIC$sensitivities, type = 'l', lty = 1, lwd = 2,col = 'green')
lines(1 - roc_Enet$specificities, roc_Enet$sensitivities, type = 'l', lty = 1, lwd = 2,col = 'purple')
```



The AUC (area under curve) represents the model's overall performance under various thresholds. Based on the AUCs of the three models, we say that the elastic net-selected model performs the best overall. Note that while its overall performance is the best, the other two models perform noticeably better at low thresholds, as well as some other thresholds.

e)

```
set.seed(1006314089)

sample_remainder <- small.nhanes[-sample(nrow(small.nhanes), 500),]

pAIC <- predict(lrmAIC, newdata = sample_remainder, type = "fitted")
quantile(pAIC, probs = seq(.1, .9, by = .1))
val.prob(pAIC, sample_remainder$SmokeNow, logit = 'p')


pBIC <- predict(lrmBIC, newdata = sample_remainder, type = "fitted")
quantile(pBIC, probs = seq(.1, .9, by = .1))
val.prob(pBIC, sample_remainder$SmokeNow, logit = 'p')

pEnet <- predict(lrmEnet, newdata = sample_remainder, type = "fitted")
quantile(pEnet, probs = seq(.1, .9, by = .1))
val.prob(pEnet, sample_remainder$SmokeNow, logit = 'p')
```

While I do not have the actual probabilities, and this question would be straightforward if I did, I would expect the observed and the predicted probabilities to differ for the deciles, more so than in (c) as we did not cross-validate and correct for bias here as we did in (c).

f

```
library(lme4)

small.nhanes <- na.omit(NHANES[NHANES$SurveyYr=="2011_12" & NHANES$Age > 17,c(1,3,4,8:11,13,25,61)])

mixed_log <- glmer(SmokeNow ~ Gender + Age + Race3 + Education + MaritalStatus +
                   HHIncome + Poverty + BPSysAve + (1 | ID),
                   family = "binomial", data = small.nhanes)
```

There is a convergence issue. It is difficult to proceed.