Statistical Modeling Course

Variable Selection Assignment

In this exercise, we will predict the number of applications, `Apps`, received using the other variables in the `College` data set. Use 5-fold cross-validation to calculate the test error using the following methods. You should fit each model five times leaving out one fifth of the data each time and calculating the test error (RMSE) on the data you left out. The total test error will be the average RMSE from the five folds. You should also calculate the standard error of the test errors, `sd(test_error)/sqrt(5)`. For best subset selection, and forward stepwise selection chose the model using BIC. For lasso, ridge, and pcr you will conduct a second level of cross validation for each of the 5 training sets. You can do this with `cv.glmnet`.

Compare your results with the null model. Code for the null model and splits is given below, make sure to compare all of your models on the same splits of the data.

Null Model

```
set.seed(2020)
# Vector of data split
folds <- sample(1:5, nrow(College), replace = TRUE)

# Variable to store error from each fold
error_null <- c()

# Loop through folds and fit models with no predictors
for (i in 1:5){
  test <- College[folds == i, ]
  train <- College[folds != i, ]
  error_null[i] <- sqrt(mean((test$Apps - mean(train$Apps))^2))
}

# Print mean RMSE
mean(error_null)
```

```
## [1] 3812.055
```

```
# Print standard error of RMSE
sd(error_null)/sqrt(5)
```

```
## [1] 363.8367
```

Problem 1

- Least squares linear model
- Best subset selection (chosen using BIC)
- Forward stepwise subset selection (chosen using BIC)

Problem 2

- Ridge-regression with lambda chosen by cross-validation, report the five λ's chosen
- Lasso with lambda chosen by cross-validation, report the five λ's chosen

Problem 3

- Fit a PCR model, with M chosen by cross-validation. Report the test error (MSE) obtained, along with the value of M selected by cross-validation.

Problem 4

Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from the approaches?

```
#Creating predict function for subset selection models
predict.regsubsets <- function(object,newdata,id,...){
  form = as.formula(object$call[[2]])
  mat = model.matrix(form,newdata)
  coefi = coef(object,id=id)
  xvars = names(coefi)
  mat[,xvars]%*%coefi
}

#Lambda choices for ridge and lasso
grid = 10^seq(10, -2, length = 100)

set.seed(2020)
# Vector of data split
folds <- sample(1:5, nrow(College), replace = TRUE)

# Variable to store error from each fold
error_null_lstsq <- c()
error_null_best <- c()
```

```r
error_null_fwd <- c()
error_null_ridge <- c()
error_null_lasso <- c()
error_null_PCR <- c()
lambda.ridge <- c()
lambda.lasso <- c()
comp.PCR <- c()

# Loop through folds and fit models with no predictors
for (i in 1:5){
  test <- College[folds == i, ]
  train <- College[folds != i, ]

  lst_sqrs_mod <- lm(Apps~., data= train)
  regfit.full = regsubsets(Apps~., train, nvmax= 17)
  regfit.forward = regsubsets(Apps~., train, nvmax= 17, method="forward")
  ridge.mod = cv.glmnet(model.matrix(Apps~., train)[,-1], train$Apps,
                        alpha=0, lambda=grid, thresh=1e-12)
  lasso.mod = cv.glmnet(model.matrix(Apps~., train)[,-1], train$Apps,
                        alpha=1, lambda=grid, thresh=1e-12)
  PCR.mod <- pcr(Apps ~ ., data=train, scale=TRUE , validation ="CV")

  #Predict and compute RMSE using test data
  pred_lstsq <- predict(lst_sqrs_mod, newdata = subset(test,select=-c(Apps)))
  ##
  nvar_full = which.min(summary(regfit.full)$bic)
  nvar_forward = which.min(summary(regfit.forward)$bic)
  pred_best <- predict(regfit.full, newdata = test, id = nvar_full)
  pred_fwd <- predict(regfit.forward, newdata = test, id = nvar_forward)
  lambda.ridge[i] <- ridge.mod$lambda.min
  lambda.lasso[i] <- lasso.mod$lambda.min
  comp.PCR[i] <- which.min(RMSEP(PCR.mod)$val[1,,])-1
  ##
  ridge.pred = predict(ridge.mod, s=lambda.ridge[i], newx = model.matrix(Apps~., test)[,-1])
  lasso.pred = predict(lasso.mod, s=lambda.lasso[i], newx = model.matrix(Apps~., test)[,-1])
  PCR.pred = predict(PCR.mod, ncomp=comp.PCR[i], newdata = model.matrix(Apps~., test)[,-1])

  error_null_lstsq[i] <- sqrt(mean((test$Apps - pred_lstsq)^2))
  error_null_best[i] <- sqrt(mean((test$Apps - pred_best)^2))
```

```
  error_null_fwd[i] <- sqrt(mean((test$Apps - pred_fwd)^2))
  error_null_ridge[i] <- sqrt(mean((test$Apps - ridge.pred)^2))
  error_null_lasso[i] <- sqrt(mean((test$Apps - lasso.pred)^2))
  error_null_PCR[i] <- sqrt(mean((test$Apps - PCR.pred)^2))

}
#Lambdas
tibble(lambda = c("lambda1","lambda2","lambda3","lambda4","lambda5"),
       ridge=lambda.ridge, lasso=lambda.lasso)
```

```
## # A tibble: 5 x 3
##   lambda  ridge lasso
##   <chr>   <dbl> <dbl>
## 1 lambda1 10.7  14.2
## 2 lambda2  0.01  0.01
## 3 lambda3  0.01 18.7
## 4 lambda4  0.01  0.01
## 5 lambda5  0.01  0.01
```

```
#PCR components
tibble("PCR components"=comp.PCR)
```

```
## # A tibble: 5 x 1
##   `PCR components`
##              <dbl>
## 1               17
## 2               17
## 3               17
## 4               17
## 5               17
```

```
Method = c("least squares",
           "best selection",
           "forward selection",
           "ridge regression",
           "lasso regression",
           "principal components"
           )
# Print mean RMSE
tibble( "Method" = Method, "Mean RMSE" = c(
                               mean(error_null_lstsq),
```

```
                              mean(error_null_best),
                              mean(error_null_fwd),
                              mean(error_null_ridge),
                              mean(error_null_lasso),
                              mean(error_null_PCR))
  )
```

```
## # A tibble: 6 x 2
##   Method                `Mean RMSE`
##   <chr>                       <dbl>
## 1 least squares               1108.
## 2 best selection              1123.
## 3 forward selection           1121.
## 4 ridge regression            1116.
## 5 lasso regression            1124.
## 6 principal components        1108.
```

```
# Print standard error of RMSE
tibble( "Method" = Method, "Std RMSE" = c(
                              sd(error_null_lstsq)/sqrt(5),
                              sd(error_null_best)/sqrt(5),
                              sd(error_null_fwd)/sqrt(5),
                              sd(error_null_ridge)/sqrt(5),
                              sd(error_null_lasso)/sqrt(5),
                              sd(error_null_PCR)/sqrt(5))
  )
```

```
## # A tibble: 6 x 2
##   Method                `Std RMSE`
##   <chr>                      <dbl>
## 1 least squares               61.9
## 2 best selection              63.2
## 3 forward selection           63.8
## 4 ridge regression            69.7
## 5 lasso regression            75.1
## 6 principal components        61.9
```

Problem 5

Generate a data set with $p = 100$ features, $n = 300$ observations, and an associated quantitative response vector generated according to the model

$$p_i = \frac{e^{x_i^T \beta}}{1 + e^{x_i^T \beta}}$$

$$y_i \sim Bin(p_i, n = 1)$$

Simulate some of the features as categorical and some as numeric. Set most of the values of $\beta_p = 0$ for most but not all p.

- Using your simulated dataset split your dataset into at training set and a test set containing using an 80/20 split.
- Perform lasso and ridge regression on the training set.
- Which model has a lower test error (MSE)? Comment on your results.

```
#Simulate dataset: 100 features, 300 observations
x <- data.frame(replicate(n = 30,
          expr = rnorm(n = 300, mean = 0, sd = 1) ),
          replicate(n = 35,
          expr = rpois(n = 300, lambda = 3) ),
          replicate(n = 35,
          expr = runif(n = 300, min = 0, max=35) )
        )
x <- data.matrix(x)
b <- matrix(0,nrow=100,ncol=1)
b[sample(seq(100),20,replace=FALSE),1] <-rnorm(20)
p <- exp(x%*%b)/(1+exp(x%*%b))
y <- rep(NA,nrow=100,ncol=1)
for (i in 1:300){
  y[i]<-rbinom(1,n=1,prob=p[i])
}
train = sample(c(TRUE,FALSE), 300, prob=c(0.8,0.2), replace=TRUE)

grid = 10^seq(10, -2, length = 100)

ridge.mod = cv.glmnet(x[train,], y[train], alpha=0, lambda=grid,
                      thresh=1e-12,family="binomial")
lasso.mod = cv.glmnet(x[train,], y[train], alpha=1, lambda=grid,
                      thresh=1e-12, family="binomial")

ridge.pred = predict(ridge.mod, s=ridge.mod$lambda.min, newx = x[!train,])
lasso.pred = predict(lasso.mod, s=lasso.mod$lambda.min, newx = x[!train,])
```

```r
error_null_ridge <- sqrt(mean((y[!train] - ridge.pred)^2))
error_null_lasso <- sqrt(mean((y[!train] - lasso.pred)^2))

print(error_null_ridge)
```

```
## [1] 4.616636
```

```r
print(error_null_lasso)
```

```
## [1] 4.724932
```

Ridge has a lower test MSE.