Math 537 Final Exam

Shamir Naran

Due June 26, 2020

```
library(tidyverse) # mutate
library(glmnet) # glmnet, cv.glmnet
library(pls) # pcr
```

```
college <- read.csv("College.csv", header = TRUE)
attach(college)
```

In this exam we will be exploring a dataset called college.csv. In this dataset, we have the following variables:

ID:

- X: Name of college (Neither a predictor nor a response)

Potential Responses:

- Apps: Number of applications this last year (Potential Response)
- Accept: Number of accepted applications (Potential Response)
- Enroll: Number of enrollees (Potential Response)

Potential Predictors:

- Private: Whether or not the University is private
- Top10perc: % of applications that are top 10% students
- Top25perc: % of applications that are top 25% students
- F.Undergrad: Full time undergrads enrolled
- P.Undergrad: Part time undergrads enrolled
- Outstate: Out of stats tuition fee
- Room.Board: Estimate for room and board
- Books: Cost of books annually
- Personal: Annual personal expenses
- PhD: Number of PhDs granted last year
- S.F. Ratio: Student to Faculty Ratio
- Perc.alumni: Percentage of alumni who donate
- Expend: Cost of education per year on average
- Grad.Rate: Graduation Rate

Our goal to model **Exclusivity** of a University using the potential predictor variables. First, we create the new variable **Exclusivity** and add it to our dataset.

```
# Add new variable with mutate()
# Exclusivity which is % application denied + % of accepted applications attending

college <- mutate(college,
                  Exclusivity = 100 * (((Apps - Accept) / (Apps)) + ((Enroll / Accept)))
)
```

Problem 1

We'd like to determine which of the following six models is best to model **Exclusivity** of a University. To do this, we will perform k-fold cross validation and use residual sum of squares (RSS) as our objective measure of fit.

1) A simple linear model using least squares and all of the variables
2) A ridge regression model with lambda = best.lam.ridge
3) A lasso regression model with lambda = best.lam.lasso
4) An elastic.net regression model with lambda = best.lam.elasticnet
5) PCR. Be sure to document and or discuss the number of components you selected and why
6) PLSR. Be sure to document and or discuss the number of components you selected and why

Before modeling, we change **Private** into a numerical variable and standardize all of our variables. Note, glmnet() can only take numerical, quantitative values.

```
college$Private <- ifelse(college$Private == "Yes",1,0) # numerical
```

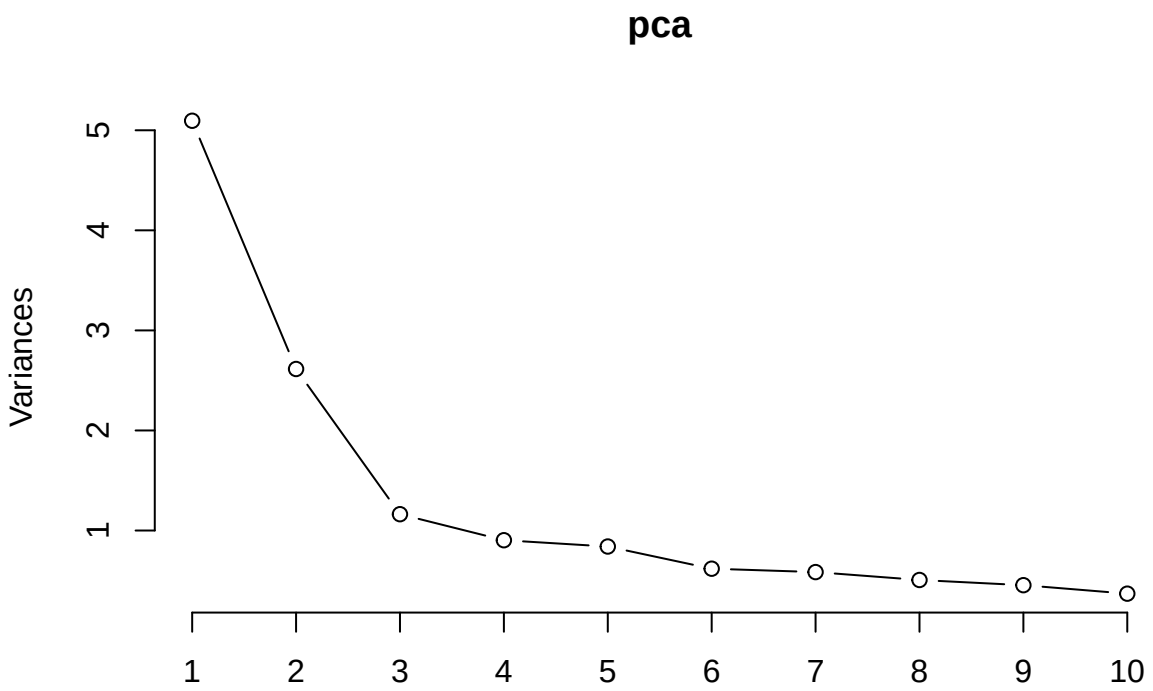
```
# Standardize

college$Private <- scale(college$Private)
college$Top10perc <- scale(college$Top10perc)
college$Top25perc <- scale(college$Top25perc)
college$F.Undergrad <- scale(college$F.Undergrad)
college$P.Undergrad <- scale(college$P.Undergrad)

college$Outstate <- scale(college$Outstate)
college$Room.Board <- scale(college$Room.Board)
college$Books <- scale(college$Books)
college$Personal <- scale(college$Personal)
college$PhD <- scale(college$PhD)

college$S.F.Ratio <- scale(college$S.F.Ratio)
college$perc.alumni <- scale(college$perc.alumni)
college$Expend <- scale(college$Expend)
college$Grad.Rate <- scale(college$Grad.Rate)
college$Exclusivity <- scale(college$Exclusivity)
```

Exclusivity is a function of **Apps**, **Accept**, and **Enroll**. We can remove these variables, along with **Terminal** which we don't use.

```
college <- college[,-c(1,3:5,15)] # delete unused variables
```

For the PCR model (model5), we need to determine how many principal components to use. Using the prcomp() function, we run PCA on the predictor variables.

```
X <- college[,-15] # only predictors
pca <- prcomp(X, center = TRUE) # standardize
plot(pca, type = "l")
```



From the scree plot, we see that the first 4 principal components contain most of the information for our predictors. We have reduced the predictor space from 14 variables to 4. The first principal component is the normalized linear combination of the predictors that has the largest variance. The second principal component is the linear combination of the predictors that has maximal variance out of all linear combinations that are uncorrelated with the first principal component. And so on.

For the PLSR model (model6), we also need to determine how many components to use.

```
pls.fit <- plsr(Exclusivity~., data = college, validation = "CV")
summary(pls.fit)
```

```
## Data:    X dimension: 777 14
##  Y dimension: 777 1
## Fit method: kernelpls
## Number of components considered: 14
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.001   0.9301   0.8893   0.8903   0.8813   0.8795   0.8797
## adjCV        1.001   0.9292   0.8856   0.8897   0.8800   0.8782   0.8785
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.8791   0.8785   0.8784    0.8783    0.8783    0.8783    0.8783
## adjCV   0.8777   0.8772   0.8771    0.8771    0.8771    0.8771    0.8771
##        14 comps
## CV       0.8783
## adjCV    0.8771
##
```

```
## TRAINING: % variance explained
##               1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X               17.88    27.37    58.94    63.66    67.46    73.25    76.84
## Exclusivity     15.65    22.57    23.80    26.16    26.60    26.73    26.84
##               8 comps  9 comps  10 comps  11 comps  12 comps  13 comps  14 comps
## X               80.00    82.39     87.35     90.46     94.03     97.01    100.00
## Exclusivity     26.87    26.88     26.88     26.88     26.88     26.88     26.88
```

Looking at the output, we see that the cross-validation error lowers all the way to the 6th component. Other components have lower cross-validation, but only slightly. For this reason, we'll use 6 as the components.

Now we run our 6 models using k-fold cross validation, with k = 5. This approach involves randomly dividing the set of observations into k groups. The first fold is treated as a test set and the model is fit on the remaining k-1 groups. Our objective measure of fit, RSS, is computed on the observations in the held out group. This procedure is repeated k times; each time, a different group of observations is treated as a test set.

```r
n <- nrow(college)
k <- 5 # 5 fold cross validation
l <- 100 # 100 of each model


modelRes <- matrix(0,6,l) # initialize RSS matrix

for (j in 1:l) {

  set.seed(j)
  sample.rows <- sample(1:n)
  cols <- floor(n/k)
  k.folds <- matrix(sample.rows[1:(cols*k)], nrow = k)

  for (i in 1:k) {

    test.index <- as.numeric(k.folds[i,])
    train.index <- as.numeric(k.folds[-i,])

    test <- college[test.index,]
    train <- college[train.index,]

    x.train <- as.matrix(train[,1:14])
    y.train <- train[,15]

    x.test <- as.matrix(test[,1:14])
    y.test <- test[,15]


    # Model 1: Simple linear model using least squares and all of the variables

    model1 <- lm(Exclusivity ~., data = train)
    pred <- predict(model1, newdata = test, type = "response")
    modelRes[1,j] <- modelRes[1,j] + sum((y.test - pred)^2) # Calculate RSS


    # Model 2: A ridge regression model with lambda = best.lam.ridge
```

```
    model2 <- glmnet(x.train, y.train, alpha = 0)
    cv.out.ridge <- cv.glmnet(x.train, y.train, alpha = 0)
    best.lam.ridge <- cv.out.ridge$lambda.min
    pred <- predict(model2, s = best.lam.ridge, newx = x.test)
    modelRes[2,j] <- modelRes[2,j] + sum((y.test - pred)^2) # Calculate RSS


    # Model 3: A lasso regression model with lambda = best.lam.lasso

    model3 <- glmnet(x.train, y.train, alpha = 1)
    cv.out.lasso <- cv.glmnet(x.train, y.train, alpha = 1)
    best.lam.lasso <- cv.out.lasso$lambda.min
    pred <- predict(model3, s = best.lam.lasso, newx = x.test)
    modelRes[3,j] <- modelRes[3,j] + sum((y.test - pred)^2) # Calculate RSS


    # Model 4: An elastic.net regression model with lambda = best.lam.elasticnet

    model4 <- glmnet(x.train, y.train, alpha = 0.5)
    cv.out.elastic <- cv.glmnet(x.train, y.train, alpha = 0.5)
    best.lam.elastic <- cv.out.elastic$lambda.min
    pred <- predict(model3, s = best.lam.elastic, newx = x.test)
    modelRes[4,j] <- modelRes[4,j] + sum((y.test - pred)^2) # Calculate RSS


    # Model 5: Principal Component Regression

    model5 <- pcr(Exclusivity~., data = train, ncomp = 4)
    pred <- as.matrix(predict(model5, newdata = x.test, ncomp = 4))
    modelRes[5,j] <- modelRes[5,j] + sum((y.test - pred)^2) # Calculate RSS


    # Model 6: Partial Least Squares Regression

    model6 <- plsr(Exclusivity~., data = train)
    pred <- as.matrix(predict(model6, newdata = x.test, ncomp = 4))
    modelRes[6,j] <- modelRes[6,j] + sum((y.test - pred)^2) # Calculate RSS

  }

}
```

In addition, we repeated this process 100 times and take an average of our SSR. This will give us a good idea of which model is performing best.

```
apply(modelRes,1,mean)
```

```
## [1] 601.4046 600.4872 601.0078 602.7493 679.8500 606.8362
```

```
which.min(apply(modelRes, 1, mean)) # which model is best
```

```
## [1] 2
```

Problem 2

Which model is best and why? The lowest RSS belongs to model2, Ridge Regression. We see that there is not much difference between our simple linear model, ridge regression, lasso regression, elastic regression, and PLSR. The fact that a simple linear model using least squares is competitive with the other models tells us that it is not over fitting the model by much. The ridge model is slightly better because we are able to shrink the coefficient estimates resulting in a more flexible model. Of note is that the PCR model with 4 principal components is by far the least competitive. This tells us that there is not much benefit in dimension reduction for this particular dataset. From the scree plot above, we chose 4 principal components but there we still information left on the table in the latter components.