Predictive Modeling Exercises Week 5

Amol Gote

08/09/2020

# Problem 9 In this exercise, we will predict the number of applications received using the other variables in the College data set.

# 9a Split the data set into a training set and a test set.

training\_college\_subset <- sample(nrow(College) \* 0.75)  
training\_college\_ds = College[training\_college\_subset, ]  
test\_college\_ds = College[-training\_college\_subset, ]  
nrow(training\_college\_ds)

## [1] 582

nrow(test\_college\_ds)

## [1] 195

test\_college\_mean <- mean(test\_college\_ds[, "Apps"])  
test\_college\_mse <- mean((test\_college\_ds[, "Apps"] - test\_college\_mean)^2)

# 9b Fit a linear model using least squares on the training set, and report the test error obtained.

lm\_fit <- lm(Apps ~ . , data = training\_college\_ds)  
lm\_predictions <- predict(lm\_fit, test\_college\_ds)  
lm\_mse <- round(mean((test\_college\_ds[,"Apps"] - lm\_predictions)^2))  
lm\_rmse <- sqrt(mean((test\_college\_ds[,"Apps"] - lm\_predictions)^2))  
lm\_mse <- round(mean((test\_college\_ds[,"Apps"] - test\_college\_mean)^2))  
cat("MSE: ", lm\_mse, "\n")

## MSE: 17299662

cat("RMSE: ", lm\_rmse, "\n")

## RMSE: 1209.721

# 9c Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

train\_ds\_matrix <- model.matrix(Apps ~ ., data = training\_college\_ds)  
test\_ds\_matrix <- model.matrix(Apps ~ ., data = test\_college\_ds)  
  
grid <- 10 ^ seq(4, -2, length = 100)  
  
ridge\_reg\_model <- cv.glmnet(train\_ds\_matrix, training\_college\_ds[,"Apps"], alpha = 0, lambda = grid, thresh = 1e-12)  
ridge\_reg\_predictions <- predict(ridge\_reg\_model, test\_ds\_matrix, s = ridge\_reg\_model$lambda.min)  
  
ridge\_mse <- round(mean((test\_college\_ds[,"Apps"] - ridge\_reg\_predictions)^2))  
ridge\_rmse <- sqrt(mean((test\_college\_ds[,"Apps"] - ridge\_reg\_predictions)^2))  
cat("MSE: ", ridge\_mse, "\n")

## MSE: 1463367

cat("RMSE: ", ridge\_rmse, "\n")

## RMSE: 1209.697

# 9d Fit a lasso model on the training set, with λ chosen by crossvalidation. Report the test error obtained, along with the number of non-zero coefficient estimates.

lasso\_model <- cv.glmnet(train\_ds\_matrix, training\_college\_ds[,"Apps"], alpha = 1, lambda = grid, thresh = 1e-12)  
  
lasso\_predictions <- predict(lasso\_model, test\_ds\_matrix, s = lasso\_model$lambda.min)  
  
lasso\_mse <- round(mean((test\_college\_ds[,"Apps"] - lasso\_predictions)^2))  
lasso\_rmse <- sqrt(mean((test\_college\_ds[,"Apps"] - lasso\_predictions)^2))  
cat("MSE: ", lasso\_mse, "\n")

## MSE: 1463310

cat("RMSE: ", lasso\_rmse, "\n")

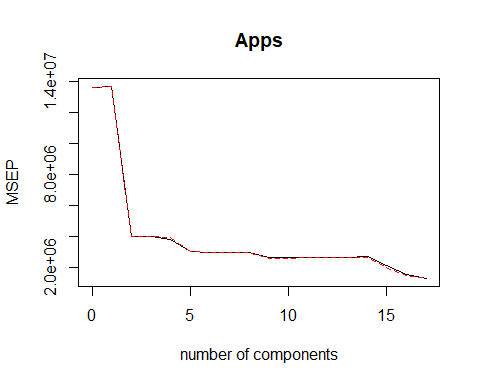
## RMSE: 1209.674

predict(lasso\_model, s = lasso\_model$lambda.min, type = "coefficients")

## 19 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) -558.09214464  
## (Intercept) .   
## PrivateYes -440.71634648  
## Accept 1.74450119  
## Enroll -1.42284478  
## Top10perc 44.66765420  
## Top25perc -13.52700930  
## F.Undergrad 0.09396717  
## P.Undergrad -0.01247536  
## Outstate -0.10714181  
## Room.Board 0.14255438  
## Books 0.06887074  
## Personal 0.03277755  
## PhD -6.34959328  
## Terminal -3.24945429  
## S.F.Ratio 20.84080014  
## perc.alumni 0.49629003  
## Expend 0.10557613  
## Grad.Rate 7.57758378

# 9e Fit a PCR model on the training set, with M chosen by crossvalidation. Report the test error obtained, along with the value of M selected by cross-validation.

pcr\_model <- pcr(Apps ~ . , data = training\_college\_ds, scale=T, validation="CV")  
validationplot(pcr\_model, val.type = "MSEP")



pcr\_predictions <- predict(pcr\_model, test\_college\_ds, ncomp = 10)  
pcr\_mse <- round(mean((test\_college\_ds[,"Apps"] - pcr\_predictions)^2))  
pcr\_rmse <- sqrt(mean((test\_college\_ds[,"Apps"] - pcr\_predictions)^2))  
cat("MSE: ", pcr\_mse, "\n")

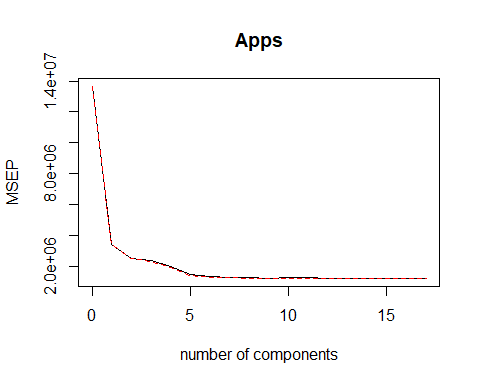
## MSE: 1776089

cat("RMSE: ", pcr\_rmse, "\n")

## RMSE: 1332.7

# 9f Fit a PLS model on the training set, with M chosen by crossvalidation. Report the test error obtained, along with the value of M selected by cross-validation.

pls\_model <- plsr(Apps ~ . , data = training\_college\_ds, scale=T, validation="CV")  
validationplot(pls\_model, val.type = "MSEP")



pls\_predictions <- predict(pls\_model, test\_college\_ds, ncomp = 10)  
pls\_mse <- round(mean((test\_college\_ds[,"Apps"] - pls\_predictions)^2))  
pls\_rmse <- sqrt(mean((test\_college\_ds[,"Apps"] - pls\_predictions)^2))  
cat("MSE: ", pls\_mse, "\n")

## MSE: 1423707

cat("RMSE: ", pls\_rmse, "\n")

## RMSE: 1193.192

# 9g 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 these five approaches?

lm\_test\_r2 <- (1 - (lm\_mse/test\_college\_mse))  
ridge\_test\_r2 <- (1 - (ridge\_mse/test\_college\_mse))  
lasso\_test\_r2 <- (1 - (lasso\_mse/test\_college\_mse))  
pcr\_test\_r2 <- (1 - (pcr\_mse/test\_college\_mse))  
pls\_test\_r2 <- (1 - (pls\_mse/test\_college\_mse))  
  
cat("R square with linear model : ", lm\_test\_r2, "\n")

## R square with linear model : 1.891096e-09

cat("R square with ridge model : ", ridge\_test\_r2, "\n")

## R square with ridge model : 0.9154107

cat("R square with lasso model : ", lasso\_test\_r2, "\n")

## R square with lasso model : 0.915414

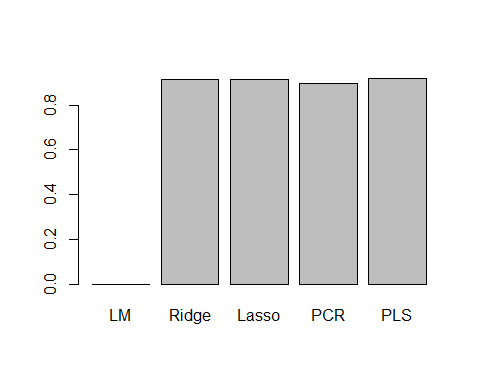
cat("R square with pcr : ", pcr\_test\_r2, "\n")

## R square with pcr : 0.8973339

cat("R square with pls : ", pls\_test\_r2, "\n")

## R square with pls : 0.9177032

barplot(c(lm\_test\_r2, ridge\_test\_r2, lasso\_test\_r2, pcr\_test\_r2, pls\_test\_r2),  
 names.arg = c("LM", "Ridge", "Lasso", "PCR", "PLS"))

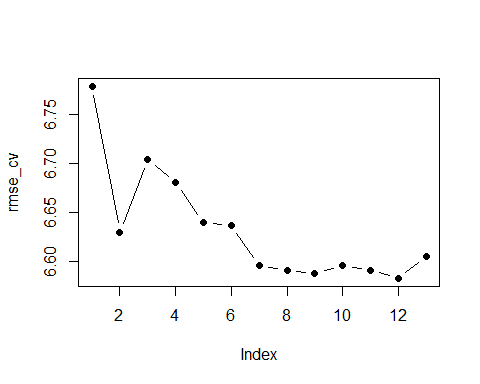


All models have R Square values near to 0.9. All models predict college applications with high accuracy, pcr has lesser accuracy than others.

# Problem 11 We will now try to predict per capita crime rate in the Boston data set.

# 11a Try out some of the regression methods explored in this chapter, such as best subset selection, the lasso, ridge regression, and PCR. Present and discuss results for the approaches that you consider.

#nrow(Boston)  
#head(Boston)  
#Boston <- subset(Boston, select = -c(resp))  
  
predict.regsubsets <- function(object, newdata, id, ...) {  
 form <- as.formula(object$call[[2]])  
 model\_matrix <- model.matrix(form, newdata)  
 coefficient <- coef(object, id = id)  
 model\_matrix[, names(coefficient)] %\*% coefficient  
}  
  
k <- 10  
p <- ncol(Boston)-1  
folds <- sample(rep(1:k, length = nrow(Boston)))  
cv.errors <- matrix(NA, k, p)  
  
for (i in 1:k) {  
 best.fit <- regsubsets(crim ~ . , data = Boston[folds!=i,], nvmax = p)  
 for (j in 1:p) {  
 pred <- predict(best.fit, Boston[folds==i,], id = j)  
 cv.errors[i,j] <- mean((Boston$crim[folds==i] - pred)^2)  
 }  
}  
  
rmse\_cv <- sqrt(apply(cv.errors, 2, mean))  
plot(rmse\_cv, pch = 19, type = "b")



best\_subset\_index\_min\_rmse <- which.min(rmse\_cv)  
best\_subset\_min\_rmse <- rmse\_cv[which.min(rmse\_cv)]  
cat("Best subset minumum rmse Index : ", best\_subset\_index\_min\_rmse, "\n")

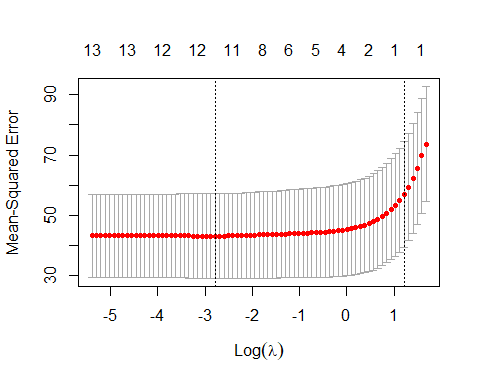
## Best subset minumum rmse Index : 12

cat("Best subset minumum rmse : ", best\_subset\_min\_rmse, "\n")

## Best subset minumum rmse : 6.582437

best\_fit\_rmse <- best\_subset\_min\_rmse

model\_matrix <- model.matrix(crim ~ ., data = Boston)  
lasso\_cv <- cv.glmnet(model\_matrix, Boston$crim, type.measure = "mse", alpha = 1)  
plot(lasso\_cv)



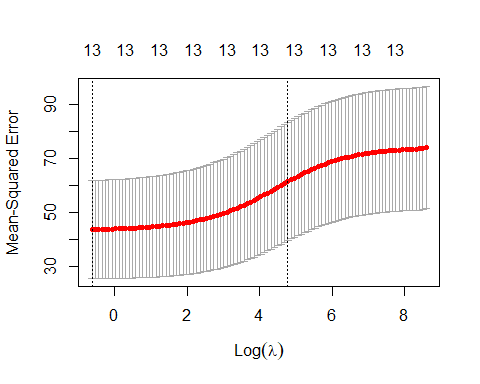
coef(lasso\_cv)

## 15 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 1.4186415  
## (Intercept) .   
## zn .   
## indus .   
## chas .   
## nox .   
## rm .   
## age .   
## dis .   
## rad 0.2298449  
## tax .   
## ptratio .   
## black .   
## lstat .   
## medv .

lasso\_rmse <- sqrt(lasso\_cv$cvm[lasso\_cv$lambda == lasso\_cv$lambda.1se])  
lasso\_rmse

## [1] 7.552794

model\_matrix <- model.matrix(crim ~ . -1, data = Boston)  
ridge\_cv <- cv.glmnet(model\_matrix, Boston$crim, type.measure = "mse", alpha = 0)  
plot(ridge\_cv)



coef(ridge\_cv)

## 14 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 1.984601825  
## zn -0.002712896  
## indus 0.023751842  
## chas -0.120843958  
## nox 1.499819906  
## rm -0.118580407  
## age 0.005022922  
## dis -0.075115440  
## rad 0.034570984  
## tax 0.001596306  
## ptratio 0.056535699  
## black -0.001981669  
## lstat 0.027880296  
## medv -0.018358122

ridge\_rmse <- sqrt(ridge\_cv$cvm[ridge\_cv$lambda == ridge\_cv$lambda.1se])  
ridge\_rmse

## [1] 7.832279

pcr\_model <- pcr(crim ~ . , data = Boston, scale = TRUE, validation = "CV")  
summary(pcr\_model)

## Data: X dimension: 506 13   
## Y dimension: 506 1  
## Fit method: svdpc  
## Number of components considered: 13  
##   
## VALIDATION: RMSEP  
## Cross-validated using 10 random segments.  
## (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps  
## CV 8.61 7.173 7.174 6.736 6.743 6.742 6.769  
## adjCV 8.61 7.172 7.173 6.734 6.736 6.739 6.764  
## 7 comps 8 comps 9 comps 10 comps 11 comps 12 comps 13 comps  
## CV 6.758 6.651 6.667 6.658 6.656 6.618 6.553  
## adjCV 6.752 6.643 6.660 6.649 6.648 6.608 6.542  
##   
## TRAINING: % variance explained  
## 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps 8 comps  
## X 47.70 60.36 69.67 76.45 82.99 88.00 91.14 93.45  
## crim 30.69 30.87 39.27 39.61 39.61 39.86 40.14 42.47  
## 9 comps 10 comps 11 comps 12 comps 13 comps  
## X 95.40 97.04 98.46 99.52 100.0  
## crim 42.55 42.78 43.04 44.13 45.4

cverr <- RMSEP(pcr\_model)$val[1,,]  
pcr\_num\_components <- which.min(cverr) - 1

13 component PCR fit has lowest RMSE value.

pls\_model <- plsr(crim ~ . , data = Boston, scale = TRUE, validation = "CV")  
summary(pls\_model)

## Data: X dimension: 506 13   
## Y dimension: 506 1  
## Fit method: kernelpls  
## Number of components considered: 13  
##   
## VALIDATION: RMSEP  
## Cross-validated using 10 random segments.  
## (Intercept) 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps  
## CV 8.61 7.019 6.685 6.656 6.633 6.593 6.600  
## adjCV 8.61 7.017 6.680 6.643 6.621 6.582 6.587  
## 7 comps 8 comps 9 comps 10 comps 11 comps 12 comps 13 comps  
## CV 6.592 6.581 6.582 6.580 6.581 6.580 6.580  
## adjCV 6.579 6.569 6.569 6.568 6.568 6.568 6.568  
##   
## TRAINING: % variance explained  
## 1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps 8 comps  
## X 47.27 56.79 61.38 71.13 76.41 79.78 83.99 86.27  
## crim 34.32 41.81 44.03 44.58 44.94 45.24 45.33 45.38  
## 9 comps 10 comps 11 comps 12 comps 13 comps  
## X 88.5 91.32 96.56 98.26 100.0  
## crim 45.4 45.40 45.40 45.40 45.4

cat("RMSE with Best Fit:", best\_fit\_rmse, "\n")

## RMSE with Best Fit: 6.582437

cat("RMSE with lasso: ", lasso\_rmse, "\n")

## RMSE with lasso: 7.552794

cat("RMSE with ridge: ", ridge\_rmse, "\n")

## RMSE with ridge: 7.832279

cat("RMSE with pcr: ", "6.517", "\n")

## RMSE with pcr: 6.517

# 11b Propose a model (or set of models) that seem to perform well on this data set, and justify your answer. Make sure that you are evaluating model performance using validation set error, crossvalidation, or some other reasonable alternative, as opposed to using training error.

Models indicate that best fit and PCR with 14 components give the best results, with PCR performing marginally better. RMSE values for the best fit PCR and very close.

# 11c Does your chosen model involve all of the features in the data set? Why or why not?

Based on the RMSE value chosen models would be

1. PCR with 13 components   
  
2. 12 parameter best subset model

PCR model with 13 components comprises of all the predictor variables. Weightage of each predictor variable for every component is given by scores. 12 parameter best subset model also contains all the predictor variables. This confrims that all the predictor variables are contributing in predicting the response variable