Question 11.1

Using the crime data set `uscrime.txt` from Questions 8.2, 9.1, and 10.1, build a regression model using:

- **Stepwise regression**
- Lasso
- Elastic net

For Parts 2 and 3, remember to scale the data first – otherwise, the regression coefficients will be on different scales and the constraint won't have the desired effect.
For Parts 2 and 3, use the `glmnet` function in R.

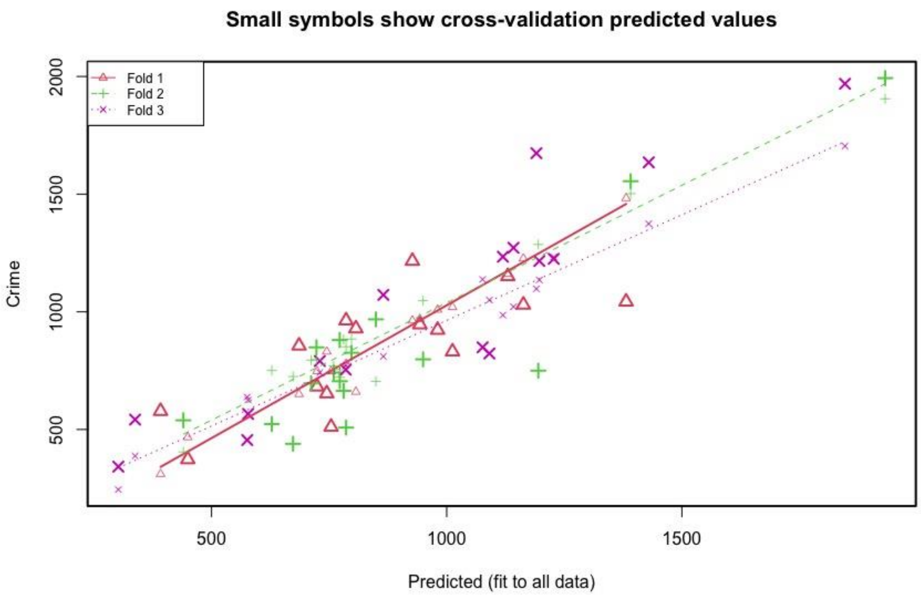
Notes on R:

- For the elastic net model, what we called λ in the videos, `glmnet` calls "alpha"; you can get a range of results by varying alpha from 1 (lasso) to 0 (ridge regression) [and, of course, other values of alpha in between].
- In a function call like `glmnet(x,y,family="mgaussian",alpha=1)` the predictors `x` need to be in R's matrix format, rather than data frame format. You can convert a data frame to a matrix using `as.matrix` – for example, `x <- as.matrix(data[,1:n-1])`
- Rather than specifying a value of T, `glmnet` returns models for a variety of values of T.

```
# -------------------- Code Answer for Question 11.1 -----------------------------
# Clear environment

rm(list = ls())

# Setting the random number generator seed so that our results are reproducible
# (Your solution doesn't need this, but it's usually good practice to do)

set.seed(100)



# ---------------------------- Data manipulation -------------------------------------

# First, read in the data
#

c_data <- read.table("uscrime.txt", stringsAsFactors = FALSE, header = TRUE)

#
# Optional check
#

head(c_data)
```

```
# ---------------------------- Start Stepwise Regression -------------------------------------

# Stepwise Regression using original variables and Cross Validation
# In backward stepwise regression.
#Scaling the data except the response variable and categorical

s_Data = as.data.frame(scale(c_data[,c(1,3,4,5,6,7,8,9,10,11,12,13,14,15)]))
s_Data <- cbind(c_data[,2],s_Data,c_data[,16]) # Add column 2 back in
colnames(s_Data)[1] <- "So"
colnames(s_Data)[16] <- "Crime"

library(caret)

# Perform 5 fold CV

ct_data <- trainControl(method = "repeatedcv", number = 5, repeats = 5)
lmFit_Step <- train(Crime ~ ., data = s_Data, "lmStepAIC", scope =
          list(lower = Crime~1, upper = Crime~.), direction = "backward",trControl=ct_data)

##Step:  AIC=503.93
##.outcome ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob
#
#Fitting a new model with the above 8 variables
#

mod_data = lm(Crime ~ M.F+U1+Prob+U2+M+Ed+Ineq+Po1, data = s_Data)
summary(mod_data)
```

cv.lm(c_data,form.lm = formula(Crime ~ M.F+U1+Prob+U2+M+Ed+Ineq+Po1),m=3,dots=FALSE,plotit = TRUE,printit = TRUE)



##

##Residual standard error: 195.5 on 38 degrees of freedom

##Multiple R-squared: 0.7888,   Adjusted R-squared: 0.7444

##F-statistic: 17.74 on 8 and 38 DF, p-value: 1.159e-10

#We got an Adjusted R-SQuared value = 0.7444 using the selected 8 variables using

Backward StepWise regression and Cross Validation

Now let's use cross-validation twith only 47 data points, to see how good is the model

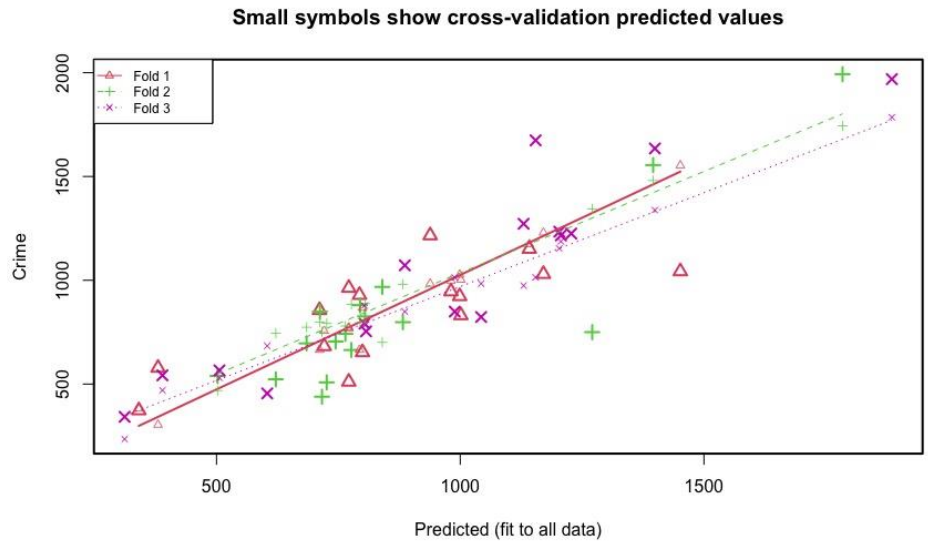
47-fold cross-validation

#

SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)

totsse <- 0

```
for(i in 1:nrow(s_Data)) {
  mod_data_i = lm(Crime ~ M.F+U1+Prob+U2+M+Ed+Ineq+Po1, data = s_Data[-i,])
  pred_i <- predict(mod_data_i,newdata=s_Data[i,])
  totsse <- totsse + ((pred_i - c_data[i,16])^2)
}
R2_mod <- 1 - totsse/SStot
R2_mod

## 0.6676

# Notice that in the model above, the p-value for M.F is above 0.1.
# We might keep it in the model, because it's close to 0.1 and
# might be important.  That's what we tested above.
# Or, we might remove it, and re-run the model without it.
# Let's see what happens if we do:

mod_data1 = lm(Crime ~ U1+Prob+U2+M+Ed+Ineq+Po1, data = s_Data)
summary(mod_data1)
cv.lm(s_Data,form.lm = formula(Crime ~ U1+Prob+U2+M+Ed+Ineq+Po1),m=3,dots=FALSE,plotit =
TRUE,printit = TRUE)
```



#Residual standard error: 199.8 on 39 degrees of freedom

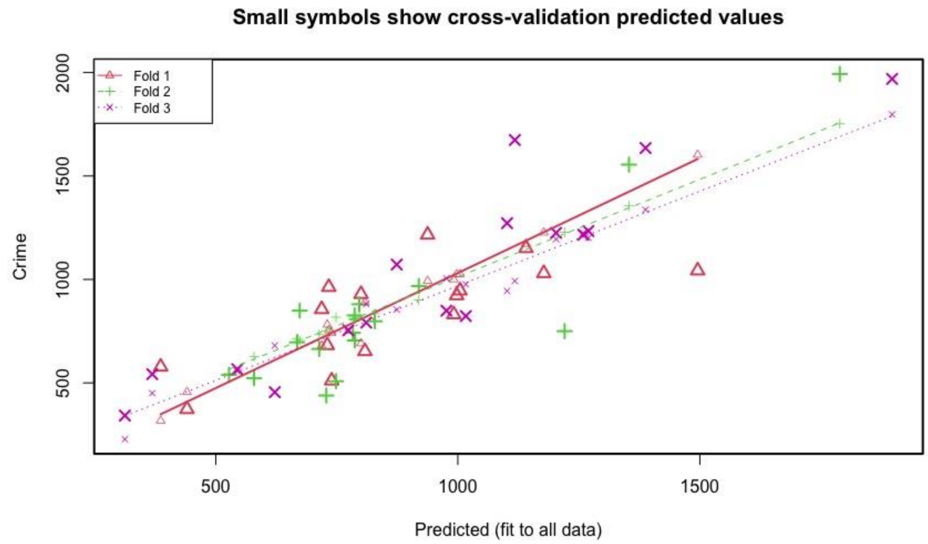
#Multiple R-squared: 0.7738,   Adjusted R-squared: 0.7332

#F-statistic: 19.06 on 7 and 39 DF, p-value: 8.805e-11

Now notice that U1 doesn't look significant... so we can take

it out too, and re-run the model.

```
mod_data2 = lm(Crime ~ Prob+U2+M+Ed+Ineq+Po1, data = s_Data)

summary(mod_data2)

cv.lm(s_Data,form.lm = formula(Crime ~ Prob+U2+M+Ed+Ineq+Po1),m=3,dots=FALSE,plotit =
TRUE,printit = TRUE)
```



##

##Residual standard error: 200.7 on 40 degrees of freedom

##Multiple R-squared: 0.7659,   Adjusted R-squared: 0.7307

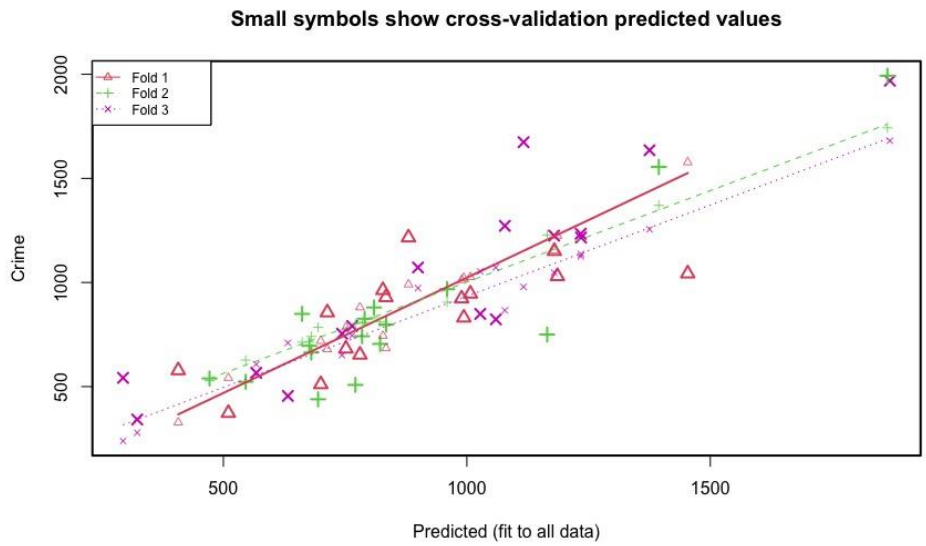
##F-statistic: 21.8 on 6 and 40 DF, p-value: 3.42e-11

This model looks good, so now let's see how it cross-validates:

```
SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)

totsse <- 0

for(i in 1:nrow(s_Data)) {

  mod_data2_i = lm(Crime ~ Prob+U2+M+Ed+Ineq+Po1, data = s_Data[-i,])

  pred_i <- predict(mod_data2_i,newdata=s_Data[i,])

  totsse <- totsse + ((pred_i - c_data[i,16])^2)

}

R3_mod <- 1 - totsse/SStot

R3_mod
```

0.666

```
# So, cross-validation shows that it's about the same whether
# we include M.F and U1 (0.668) or not (0.666).  That gives some
# support to the idea that M.F and U1 really aren't significant.
# Since the quality is about the same, we should probably use the
# simpler model.

# --------------------------- Lasso Regression ------------------------------------

library(glmnet)

#building lasso Regression Model

XP=data.matrix(s_Data[,-16])
YP=data.matrix(s_Data$Crime)
lasso=cv.glmnet(x=as.matrix(s_Data[,-16]),y=as.matrix(s_Data$Crime),alpha=1,
        nfolds = 5,type.measure="mse",family="gaussian")

#Output the coefficients of the variables selected by lasso

coef(lasso, s=lasso$lambda.min)

#Fitting a new model with  9 variables

mod_lasso1 = lm(Crime ~So+M+Ed+Po1+M.F+NW+U2+Ineq+Prob, data = s_Data)
summary(mod_lasso1)
cv.lm(s_Data,form.lm = formula(Crime ~So+M+Ed+Po1+M.F+NW+U2+Ineq+Prob),m=3,dots=FALSE,plotit
= TRUE,printit = TRUE)
```



##

##Residual standard error: 204.5 on 37 degrees of freedom

##Multiple R-squared: 0.775,   Adjusted R-squared: 0.72

##F-statistic: 14.2 on 9 and 37 DF, p-value: 1.54e-09

#We obtain a slightly lower Adjusted R-SQuared value = 0.72 using the selected 9 variables using

Lasso regression and Cross Validation

Now let's see how it cross-validates:

```
SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)
totsse <- 0
for(i in 1:nrow(s_Data)) {
  mod_lasso1_i = lm(Crime ~ So+M+Ed+Po1+M.F+NW+U2+Ineq+Prob, data = s_Data[-i,])
  pred_i <- predict(mod_lasso1_i,newdata=s_Data[i,])
  totsse <- totsse + ((pred_i - c_data[i,16])^2)
}
LR3_mod <- 1 - totsse/SStot
```

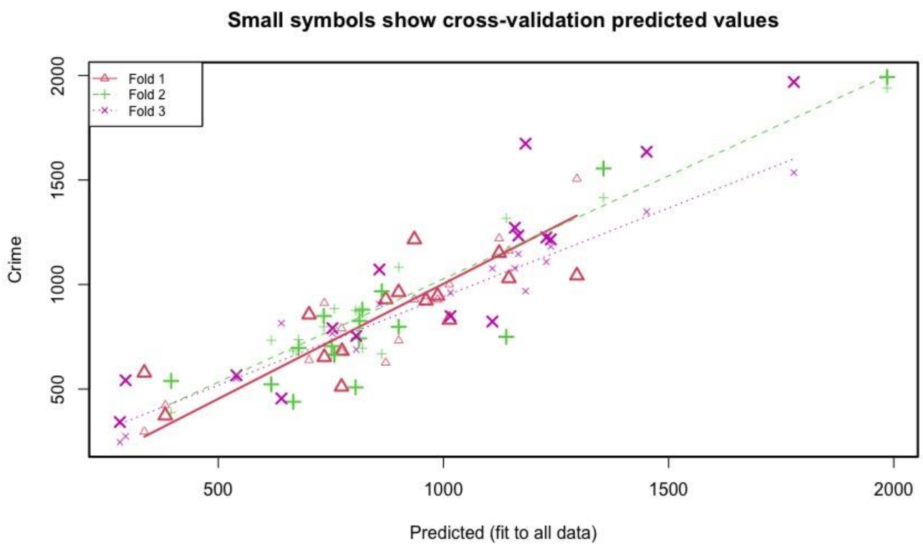
```
LR3_mod

## 0.666

# based on above observation, three of the variables (So, M.F, and NW)

# are not significant.  Let's remove them but

#It's exactly the same model we got above

# stepwis regression model!

# Please refer above R3_mod value




# ---------------------------- Elastic Net  Regression-------------------------------------

#We vary alpha in steps of 0.1 from 0 to 1 and calculate the resultant R-Squared values


R2=c()

for (i in 0:10) {

  mod_elastic = cv.glmnet(x=as.matrix(s_Data[,-16]),y=as.matrix(s_Data$Crime),

                        alpha=i/10,nfolds = 5,type.measure="mse",family="gaussian")



  #The deviance(dev.ratio ) shows the percentage of deviance explained,

  #(equivalent to r squared in case of regression)


  R2 = cbind(R2,mod_elastic$glmnet.fit$dev.ratio[which(mod_elastic$glmnet.fit$lambda == 
mod_elastic$lambda.min)])


}
```

```
R2

##[,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]     [,9]
##[1,] 0.7062797 0.7535141 0.7386241 0.7402678 0.7603113 0.7734391 0.7886831 0.763631
0.7966378
##[,10]    [,11]
##[1,] 0.7569592 0.7726187

#Best value of alpha

alpha_best = (which.max(R2)-1)/10
alpha_best

## 0.8
#Therefore we find that the best value of alpha may not lie somewhether between 0 and 1
 #Lets build the model using this alpha value.

E_net=cv.glmnet(x=as.matrix(s_Data[,-16]),y=as.matrix(s_Data$Crime),alpha=alpha_best,
                 nfolds = 5,type.measure="mse",family="gaussian")

#Output the coefficients of the variables selected by Elastic Net

coef(E_net, s=E_net$lambda.min)

# The Elastic Net selects 13 variables compared to 10 in Lasso and 8 in Step Wise.
# Next we compare how this new model performs
# compared to the Lasso and Step Wise models
```

mod_Elastic_net = lm(Crime ~So+M+Ed+Po1+Po2+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob, data = s_Data)

summary(mod_Elastic_net)

cv.lm(s_Data,form.lm = formula(Crime ~So+M+Ed+Po1+Po2+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob),m=3,dots=FALSE,plotit = TRUE,printit = TRUE)



##

##Residual standard error: 204 on 33 degrees of freedom

##Multiple R-squared: 0.8005,   Adjusted R-squared: 0.7219

##F-statistic: 10.19 on 13 and 33 DF, p-value: 4.088e-08

The R-SQuared value is similar using Elastic Net and 13 variables. Therefore this method

may not be doing a good job as it selects 3 more variables for a similar RSquared value

Now let's see how it cross-validates:

SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)

totsse <- 0

```
for(i in 1:nrow(s_Data)) {

  mod_Enet_i = lm(Crime ~ So+M+Ed+Po1+Po2+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob, data =
s_Data[-i,])

  pred_i <- predict(mod_Enet_i,newdata=s_Data[i,])

  totsse <- totsse + ((pred_i - c_data[i,16])^2)

}

ER5_mod <- 1 - totsse/SStot

ER5_mod


## 0.574


# That's a much worse cross-validated R-squared estimate. Why?

# As before, look at the p-values.  Most of those variables'

# p-values seem to indicate that they're not significant.

# If we remove them all, we're left with M, Ed, Po1, U2, Ineq,

# and Prob.

# Does that look familiar?  It should -- it's the same set of 6

# variables we were left with after removing insignificant ones

# from the Stepwise and Lasso models above!


# Before we quit, let's go back and use PCA on the variables,

# and then build Stepwise, Lasso, and Elastic Net models using

# the principal components.


# ================================================


# -------Implementing the above 3 models using Principal Component Analysis---------


# Run PCA on matrix of scaled predictors
```

```
pca <- prcomp(c_data[,1:15], scale. = TRUE)

summary(pca)



# PCA PLOT


screeplot(pca, type="lines",col="blue")


var <- pca$sdev^2

propvar <- var/sum(var)


plot(propvar, xlab = "Principal Component", ylab = "Proportion of Variance Explained",ylim = c(0,1), type
= "b")


plot(cumsum(propvar), xlab = "Principal Component", ylab = "Cumulative Proportion of Variance
Explained",

    ylim = c(0,1), type = "b")


# For the purpose of this question, let us use all the PCs instead of the original variables

# and evaluate the performance of the 3 above models



# Creating a dataframe of response variable and PCs

#-----------------


PCcrime <- as.data.frame(cbind(pca$x, c_data[,16]))

colnames(PCcrime)[16] <- "Crime"
```

```
# --------------------------- Stepwise Regression ------------------------------------

# Stepwise Regression using PCs and Cross Validation
# In backward stepwise regression. Our lower model will have only the intercept
# and all variables in our full model.


library(caret)

# Now use the code below to perform 5 fold CV

ctrl <- trainControl(method = "repeatedcv", number = 5, repeats = 5)
set.seed(1)
lmFit_Step_PC <- train(Crime ~ ., data = PCcrime, "lmStepAIC", scope =
                  list(lower = Crime~1, upper = Crime~.), direction = "backward",trControl=ctrl)

##Step:  AIC=507.37
##.outcome ~ PC1 + PC2 + PC4 + PC5 + PC6 + PC7 + PC12 + PC14 +
##PC15


#Fitting a new model with these 9 PCS

mod1_Step_PC = lm(Crime ~ PC15+PC6+PC14+PC7+PC4+PC12+PC2+PC1+PC5, data = PCcrime)
summary(mod1_Step_PC)


##
```

##Residual standard error: 201.2 on 37 degrees of freedom

##Multiple R-squared: 0.7823,   Adjusted R-squared: 0.7293

##F-statistic: 14.77 on 9 and 37 DF, p-value: 8.755e-10

#We obtain an Adjusted R-SQuared value = 0.729 using the selected 9PCs using

Backward StepWise regression and Cross Validation. This is slightly lower

than using the same method on the original variables

Now let's see how it cross-validates:

```
SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)
totsse <- 0
for(i in 1:nrow(PCcrime)) {
  mod_lasso_i = lm(Crime ~ PC15+PC6+PC14+PC7+PC4+PC12+PC2+PC1+PC5, data = PCcrime[-i,])
  pred_i <- predict(mod_lasso_i,newdata=PCcrime[i,])
  totsse <- totsse + ((pred_i - PCcrime[i,16])^2)
}
RR2_mod <- 1 - totsse/SStot
RR2_mod
```

0.6311

Notice that PC15 and PC6 were not significant in the model

above. If we take them out, here's what we get:

```
mod_Step2_PC = lm(Crime ~ PC14+PC7+PC4+PC12+PC2+PC1+PC5, data = PCcrime)
summary(mod_Step2_PC)
```

```
##

##Residual standard error: 207 on 39 degrees of freedom

##Multiple R-squared:  0.757,     Adjusted R-squared:  0.713

##F-statistic: 17.3 on 7 and 39 DF,  p-value: 3.41e-10


# Now let's see how it cross-validates:


SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)

totsse <- 0

for(i in 1:nrow(PCcrime)) {

  mod_lasso_i = lm(Crime ~ PC14+PC7+PC4+PC12+PC2+PC1+PC5, data = PCcrime[-i,])

  pred_i <- predict(mod_lasso_i,newdata=PCcrime[i,])

  totsse <- totsse + ((pred_i - PCcrime[i,16])^2)

}

RR3_mod <- 1 - totsse/SStot

RR3_mod


## 0.627


# About the same as above, so the simpler model might be better to use.


# ---------------------------- Lasso Regression -------------------------------------


library(glmnet)


#building lasso
```

```
XP=data.matrix(PCcrime[,-16])

YP=data.matrix(PCcrime$Crime)

lasso_PC=cv.glmnet(x=as.matrix(PCcrime[,-16]),y=as.matrix(PCcrime$Crime),alpha=1,
          nfolds = 5,type.measure="mse",family="gaussian")

#Output the coefficients of the variables selected by lasso

coef(lasso_PC, s=lasso_PC$lambda.min)

#Fitting a new model with these 12 PCs compared to 10 original variables

mod_l_PC = lm(Crime ~PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC10+PC12+PC13+PC14+PC15, data =
PCcrime)

summary(mod_l_PC)

##

##Residual standard error: 202.1 on 34 degrees of freedom

##Multiple R-squared:  0.7981,   Adjusted R-squared:  0.7269

##F-statistic:  11.2 on 12 and 34 DF,  p-value: 1.408e-08

#We obtain a similar Adjusted R-SQuared value = 0.7269 using the selected 12 PCs instead

# of the 10 variables using Lasso regression

# Now let's see how it cross-validates:

SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)
```

```
totsse <- 0

for(i in 1:nrow(PCcrime)) {

  mod_lasso_i = lm(Crime ~ PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC10+PC12+PC13+PC14+PC15, data =
PCcrime[-i,])

  pred_i <- predict(mod_lasso_i,newdata=PCcrime[i,])

  totsse <- totsse + ((pred_i - PCcrime[i,16])^2)

}

LRR2_mod <- 1 - totsse/SStot

LRR2_mod


## 0.5857


# Looks worse.  But notice that PCs 3, 6, 10, 13, and 15 do not

# appear to be significant.  Let's take them out.

# When we do, we get the same model as when we use only significant

# variables from the stepwise PC model.



# ---------------------------- Elastic Net -------------------------------------


#We vary alpha in steps of 0.1 from 0 to 1 and calculate the resultant R-Squared values


R2_PC=c()

for (i in 0:10) {

  model = cv.glmnet(x=as.matrix(PCcrime[,-16]),y=as.matrix(PCcrime$Crime),

                  alpha=i/10,nfolds = 5,type.measure="mse",family="gaussian")


  #The deviance(dev.ratio ) shows the percentage of deviance explained,

  #(equivalent to r squared in case of regression)
```

R2_PC = cbind(R2_PC,model$glmnet.fit$dev.ratio[which(model$glmnet.fit$lambda == model$lambda.min)])

}

R2_PC

[,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]

##[1,] 0.7695465 0.7517182 0.7787271 0.8014505 0.749221 0.7857614 0.7590517 0.7981891 0.7635869 0.7937638 0.7940698

#Best value of alpha

alpha_best_PC = (which.max(R2_PC)-1)/10

alpha_best_PC

0.3

An interesting observation after we use PCs instead of original variables. We observe that the best

alpha value=0.3 which is slightly closer to a Lasso model. The R-Squared values are

slightly higher here. Lets build the model using this alpha value.

Elastic_net_PC=cv.glmnet(x=as.matrix(PCcrime[,-16]),y=as.matrix(PCcrime$Crime),alpha=alpha_best,

nfolds = 5,type.measure="mse",family="gaussian")

#Output the coefficients of the variables selected by Elastic Net

coef(Elastic_net_PC, s=Elastic_net_PC$lambda.min)

The Elastic Net selects only 10 PCs compared to 12 in Lasso. Next we compare how this new model performs

compared to the Lasso model

```
mod_Elastic_net_PC = lm(Crime ~PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC12+PC14+PC15, data = PCcrime)

summary(mod_Elastic_net_PC)


##

##Residual standard error: 200 on 36 degrees of freedom

##Multiple R-squared:  0.7908,   Adjusted R-squared:  0.7327

##F-statistic: 13.61 on 10 and 36 DF,  p-value: 1.785e-09


# The R-SQuared value is slightly higher using Elastic Net and only 10 PCS compared to 12 PCs which

# was returned by Lasso. Elastic Net performs relatively better compared to Stepwise and Lasso


# Now let's see how it cross-validates:


SStot <- sum((c_data$Crime - mean(c_data$Crime))^2)

totsse <- 0

for(i in 1:nrow(PCcrime)) {

  mod_lasso_i = lm(Crime ~ PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC12+PC14+PC15, data = PCcrime[-i,])

  pred_i <- predict(mod_lasso_i,newdata=PCcrime[i,])

  totsse <- totsse + ((pred_i - PCcrime[i,16])^2)

}

ER_PC_mod <- 1 - totsse/SStot

ER_PC_mod
```

```
## 0.6267

# If we take out the seemingly-insignificant variables PC3,
# PC6, and PC15, we're left with the same model we had before
# after taking insignificant variables out of a PC model.
```