Data Analysis

Ariane Stark

4/28/2021

```
dat <- read_csv("case_and_demographics.csv") %>%
  subset(select = -c(1)) %>%
  subset(select = c(1,3:4,2,5:6,43,56:61 )) %>%
  #mutate(Cum_Case_Median = cut_number(cum_case,2, labels= c(0,1))) %>%
  mutate(Pop_Pov = Total_Households_Below_Poverty/(
    Total_Households_Below_Poverty + Total_Households_Above_Poverty)) %>%
  na.omit()  #removes Rio Arriba County NM which has NA in the pov. dem.
```

Cross Validation of Best Subset, Forward and Backward Stepwise

```
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
}
```

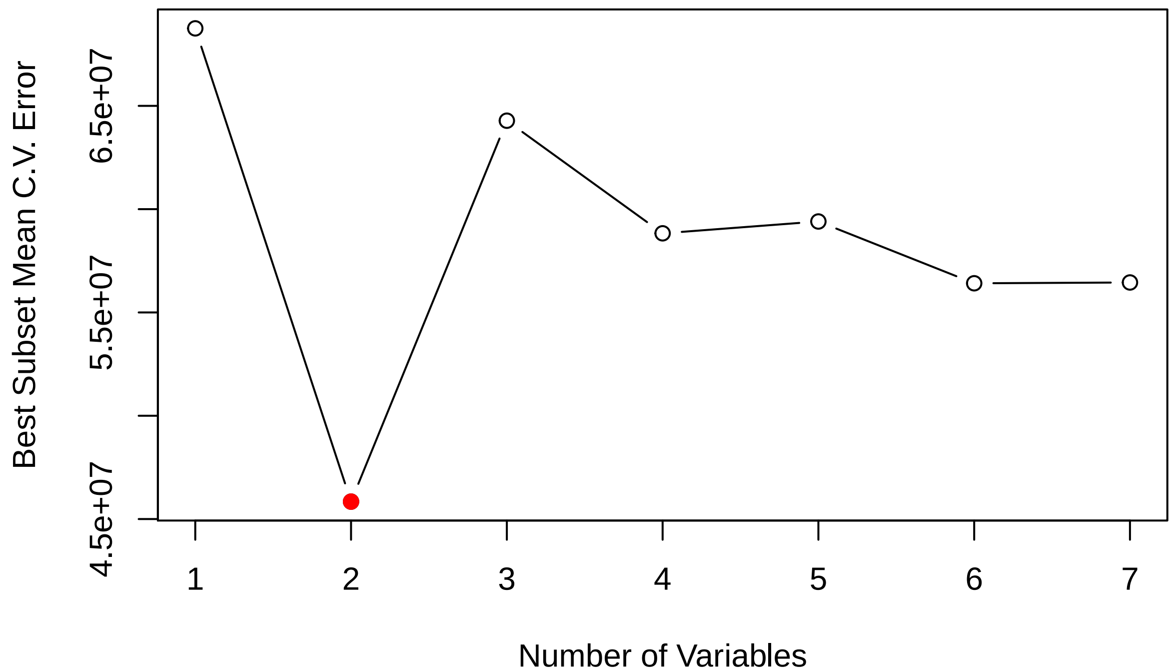
```
set.seed(100)

k<-10

folds<-sample(1:k,nrow(dat[,c(4:10,14)]),replace=TRUE)
cv.errors.best_subset <- matrix(NA,k,7, dimnames=list(NULL, paste(1:7)))
cv.errors.forward <- matrix(NA,k,7, dimnames=list(NULL, paste(1:7)))
cv.errors.backward <- matrix(NA,k,7, dimnames=list(NULL, paste(1:7)))


for(j in 1:k){
  # best subset select
  best_subset.fit <- regsubsets(cum_case~.,data = dat[folds!=j,c(4:10,14)],
                                method = "exhaustive", nvmax = 7)
  # forward stepwise select
  forward.fit <- regsubsets(cum_case~.,data = dat[folds!=j,c(4:10,14)],
                                method = "forward", nvmax = 7)
  # backward stepwise select
  backward.fit <- regsubsets(cum_case~.,data = dat[folds!=j,c(4:10,14)],
                                method = "backward", nvmax = 7)
```

```r
for(i in 1:7){
  # best subset error
  pred.best <-
    predict.regsubsets(best_subset.fit, dat[folds==j,c(4:10,14)], id=i)
  cv.errors.best_subset[j,i] <-
    mean((dat$cum_case[folds==j]-pred.best)^2)

  # forward stepwise error
  pred.forward <-
    predict.regsubsets(forward.fit, dat[folds==j,c(4:10,14)], id=i)
  cv.errors.forward[j,i] <-
    mean((dat$cum_case[folds==j]-pred.forward)^2)

  # backward stepwise error
  pred.backward <-
    predict.regsubsets(backward.fit, dat[folds==j,c(4:10,14)], id=i)
  cv.errors.backward[j,i] <-
    mean((dat$cum_case[folds==j]-pred.backward)^2)


} }
```

```r
#best subset CV model selection
mean.cv.errors.best <- apply(cv.errors.best_subset,2,mean)
mean.cv.errors.best
```

```
##        1        2        3        4        5        6        7
## 68750976 45843477 64281278 58830255 59403224 56411908 56450593
```

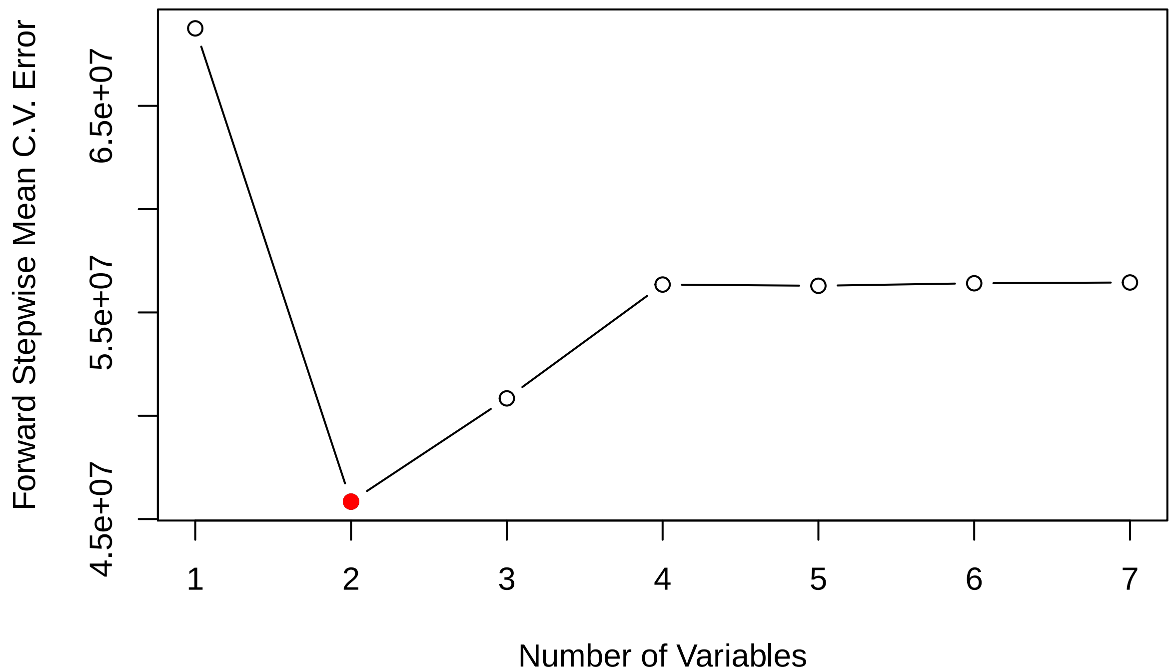
```r
plot(mean.cv.errors.best,type="b",
     xlab="Number of Variables",
     ylab="Best Subset Mean C.V. Error")
points(which.min(mean.cv.errors.best),
       mean.cv.errors.best[which.min(mean.cv.errors.best)],
       col="red", cex=1.5,pch=20)
```




```
#forward stepwise CV model selection
mean.cv.errors.forward <- apply(cv.errors.forward,2,mean)
mean.cv.errors.forward
```

```
##        1        2        3        4        5        6        7
## 68750976 45843477 50841013 56350133 56291409 56411908 56450593
```

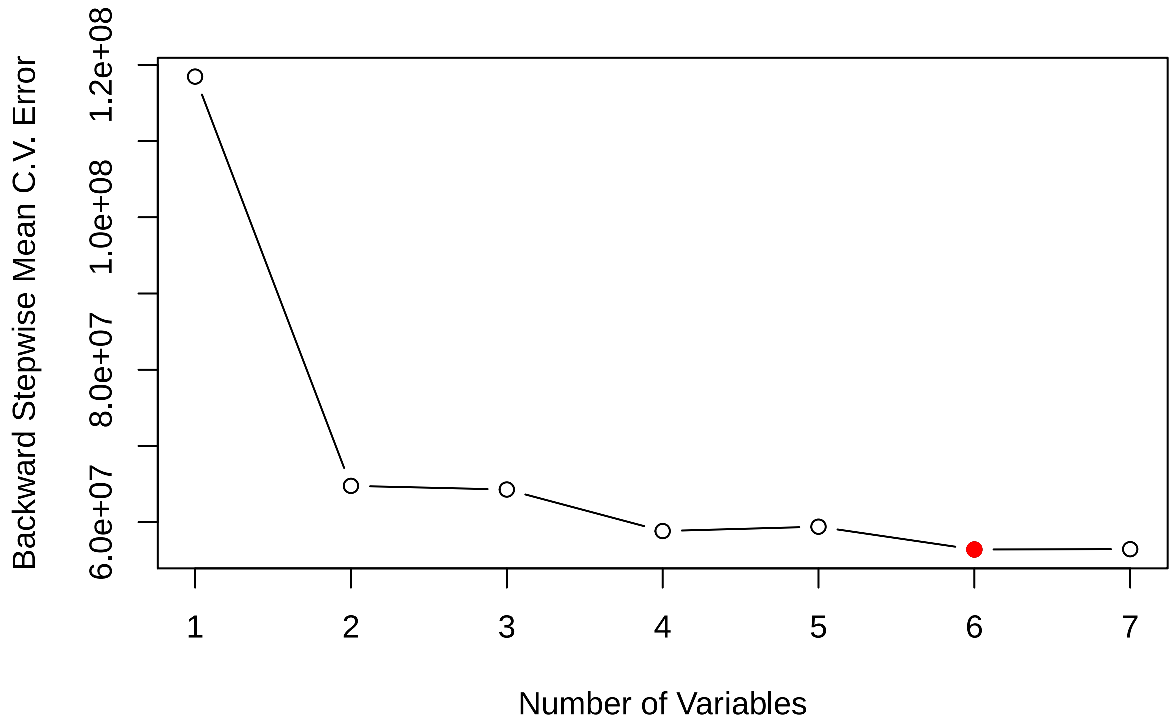
```
plot(mean.cv.errors.forward,type="b",
     xlab="Number of Variables",
     ylab="Forward Stepwise Mean C.V. Error")
points(which.min(mean.cv.errors.forward),
       mean.cv.errors.forward[which.min(mean.cv.errors.forward)],
       col="red", cex=1.5,pch=20)
```




```
#backward stepwise CV model selection
mean.cv.errors.backward <- apply(cv.errors.backward,2,mean)
mean.cv.errors.backward
```

```
##         1          2          3          4          5          6          7
## 118461309   64763873   64281278   58830255   59403224   56411908   56450593
```

```
plot(mean.cv.errors.backward,type="b",
     xlab="Number of Variables",
     ylab="Backward Stepwise Mean C.V. Error")
points(which.min(mean.cv.errors.backward),
       mean.cv.errors.backward[which.min(mean.cv.errors.backward)],
       col="red", cex=1.5,pch=20)
```



Best Subset Overall

```
best_subset.fit <- regsubsets(cum_case~.,data = dat[,c(4:10,14)],
                              method = "exhaustive", nvmax = 7)
best_subset.summary <- summary(best_subset.fit)
best_subset.summary.frame <- data_frame("Parameters" = seq(1:7),
                                        "R^2"=best_subset.summary$rsq,
                                      "AdjR^2"=best_subset.summary$adjr2,
                                      "CP"=best_subset.summary$cp,
                                      "BIC"=best_subset.summary$bic)
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## Please use `tibble()` instead.
```

```
best_subset.summary.frame %>% kable()
```

Parameters	R^2	AdjR^2	CP	BIC
1	0.9451696	0.9451526	2098.53652	-9330.249
2	0.9638620	0.9638395	289.08327	-10664.172
3	0.9647819	0.9647490	201.93735	-10739.096
4	0.9664109	0.9663690	46.07894	-10883.461
5	0.9666374	0.9665855	26.12260	-10897.170
6	0.9667828	0.9667208	14.03391	-10903.151
7	0.9668657	0.9667935	8.00000	-10903.118

Based on Cross Validation we pick the model with 2 parameters

```
coef(best_subset.fit,2)
```

```
##    (Intercept)      Total_Pop Total_Hispanic
##   315.11931521     0.07300349     0.08304726
```

Forward Stepwise Overall

```
forward.fit <- regsubsets(cum_case~.,data = dat[,c(4:10,14)],
                              method = "forward", nvmax = 7)
forward.summary <- summary(forward.fit)
forward.summary.frame <- data_frame("Parameters" = seq(1:7),
                                        "R^2"=forward.summary$rsq,
                                      "AdjR^2"=forward.summary$adjr2,
                                      "CP"=forward.summary$cp,
                                      "BIC"=forward.summary$bic)
forward.summary.frame %>% kable()
```

Parameters	R^2	AdjR^2	CP	BIC
1	0.9451696	0.9451526	2098.53652	-9330.249
2	0.9638620	0.9638395	289.08327	-10664.172
3	0.9647673	0.9647344	203.35560	-10737.759
4	0.9664109	0.9663690	46.07894	-10883.461
5	0.9666374	0.9665855	26.12260	-10897.170
6	0.9667828	0.9667208	14.03391	-10903.151
7	0.9668657	0.9667935	8.00000	-10903.118

Based on Cross Validation we pick the model with 2 parameters

```
coef(forward.fit,2)
```

```
##    (Intercept)      Total_Pop Total_Hispanic
##   315.11931521     0.07300349     0.08304726
```

Backward Stepwise Overall

```r
backward.fit <- regsubsets(cum_case~.,data = dat[,c(4:10,14)],
                            method = "backward", nvmax = 7)
backward.summary <- summary(backward.fit)
backward.summary.frame <- data_frame("Parameters" = seq(1:7),
                                      "R^2"=backward.summary$rsq,
                                    "AdjR^2"=backward.summary$adjr2,
                                    "CP"=backward.summary$cp,
                                    "BIC"=backward.summary$bic)
backward.summary.frame %>% kable()
```

Parameters	R^2	AdjR^2	CP	BIC
1	0.9025340	0.9025037	6230.29774	-7478.489
2	0.9558885	0.9558611	1061.78164	-10022.384
3	0.9647819	0.9647490	201.93735	-10739.096
4	0.9664109	0.9663690	46.07894	-10883.461
5	0.9666374	0.9665855	26.12260	-10897.170
6	0.9667828	0.9667208	14.03391	-10903.151
7	0.9668657	0.9667935	8.00000	-10903.118

Based on Cross Validation we pick the model with 6 parameters

```r
coef(backward.fit,6)
```

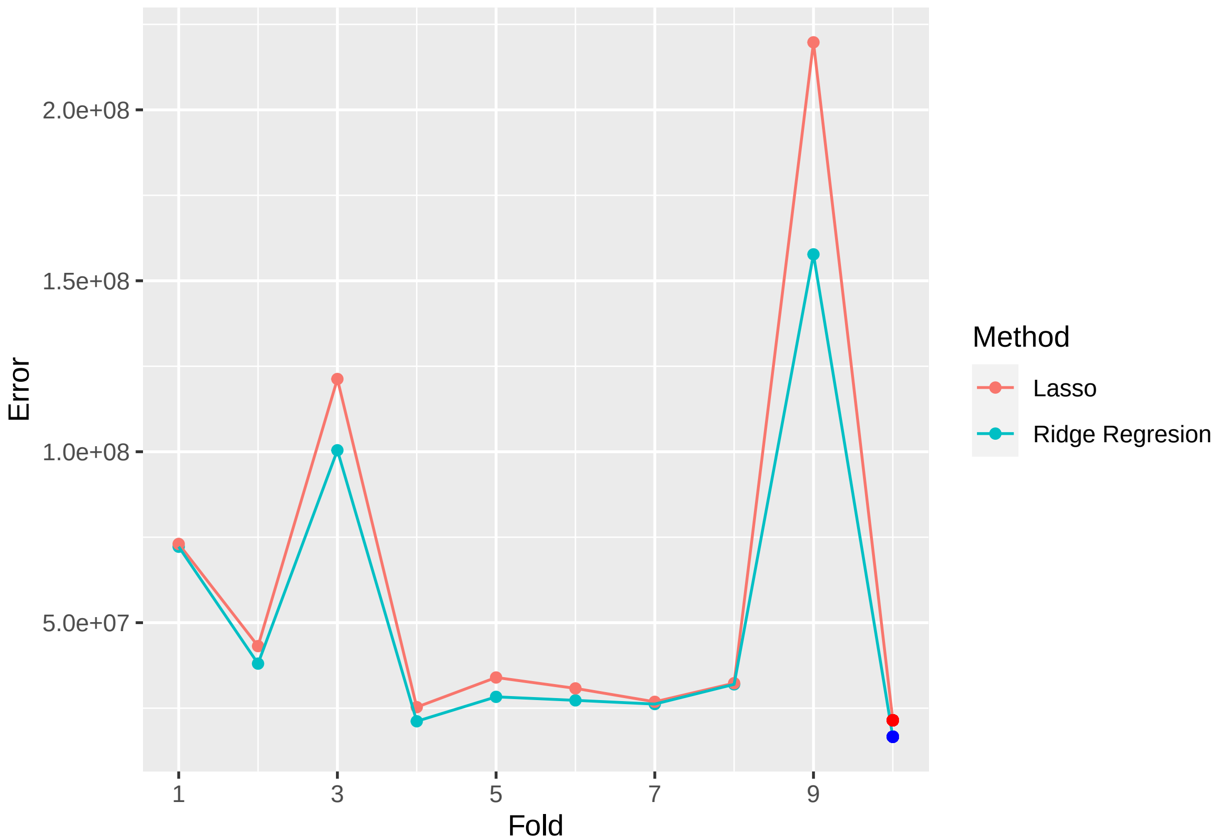
```
##    (Intercept)    POP_DENSITY      Total_Pop    Total_White    Total_Black
##   1.173614e+03  -6.692595e-01   3.446984e-02   4.242023e-02   5.362220e-02
## Total_Hispanic        Pop_Pov
##   1.015292e-01  -6.375931e+03
```

Ridge and Lasso

```r
set.seed(100)
k <- 10
cv.error.lasso <- rep(NA,k)
cv.error.ridge <- rep(NA,k)
cv.lam.lasso <- rep(NA,k)
cv.lam.ridge <- rep(NA,k)
set.seed(1)
folds<-sample(1:k,nrow(dat[,c(4:10,14)]),replace=TRUE)
for (i in 1:k){
  train <- dat[folds!=i,c(4:10,14)] #Set the training set
  test <- dat[folds==i,c(4:10,14)] #Set testing set
  x.train <- model.matrix(cum_case~., data=train)[,-1]
  y.train <- train$cum_case
  x.test <- model.matrix(cum_case~., data=test)[,-1]
  y.test <- test$cum_case

  #Lasso
  lasso.mod <- glmnet(x.train,y.train,alpha=1)
  cv.out <- cv.glmnet(x.train,y.train,alpha=1)
  bestlam <- cv.out$lambda.min
```

```r
  cv.lam.lasso[i] <- bestlam
  lasso.pred <- predict(lasso.mod,s=bestlam ,newx=x.test)
  cv.error.lasso[i] <- mean((lasso.pred-y.test)^2)


  #Ridge
  ridge.mod <- glmnet(x.train,y.train,alpha=0)
  cv.out <- cv.glmnet(x.train,y.train,alpha=0)
  bestlam <- cv.out$lambda.min
  cv.lam.ridge[i] <- bestlam
  ridge.pred <- predict(ridge.mod,s=bestlam ,newx=x.test)
  cv.error.ridge[i] <- mean((ridge.pred-y.test)^2)

}
```

```r
data.frame("Fold" = rep(1:10,2),
           "Error" = c(cv.error.ridge,cv.error.lasso),
           "Method" = c(rep("Ridge Regresion",10),rep("Lasso",10))) %>%
  ggplot(aes(x=Fold,y=Error,color=Method))+
  geom_point()+
  geom_line() +
  geom_point(aes(x=which.min(cv.error.ridge),
                 y=cv.error.ridge[which.min(cv.error.ridge)]),
             color="blue")+
  geom_point(aes(x=which.min(cv.error.lasso),
                 y=cv.error.lasso[which.min(cv.error.lasso)]),
             color="red") +
  scale_x_continuous(breaks = seq(1,10,2))
```




```
# Lasso CV Errors and Mean
cv.error.lasso[which.min(cv.error.lasso)]
```

```
## [1] 21450422
```

```
mean(cv.error.lasso)
```

```
## [1] 62785019
```

```
# Ridge CV Errors and Mean
cv.error.ridge[which.min(cv.error.ridge)]
```

```
## [1] 16642166
```

```
mean(cv.error.ridge)
```

```
## [1] 51993169
```

Ridge Regression performs better when the response is a function of many predictors of roughly equal size

Ridge With Best CV Error Lambda

```r
x<-model.matrix(cum_case~., data=dat[,c(4:10,14)])[,-1]
y<-dat$cum_case

ridge.fit<-glmnet(x,y,alpha=0)
ridge.coef<-predict(ridge.fit,type="coefficients",
                    s=cv.lam.ridge[which.min(cv.error.ridge)])[1:7,]
ridge.coef[ridge.coef!=0]
```

```
##    (Intercept)    POP_DENSITY      Total_Pop     Median_Age    Total_White
##  4313.64863130    -0.38262860     0.02948192   -70.41443827     0.04633555
##    Total_Black Total_Hispanic
##     0.06786887     0.09367194
```

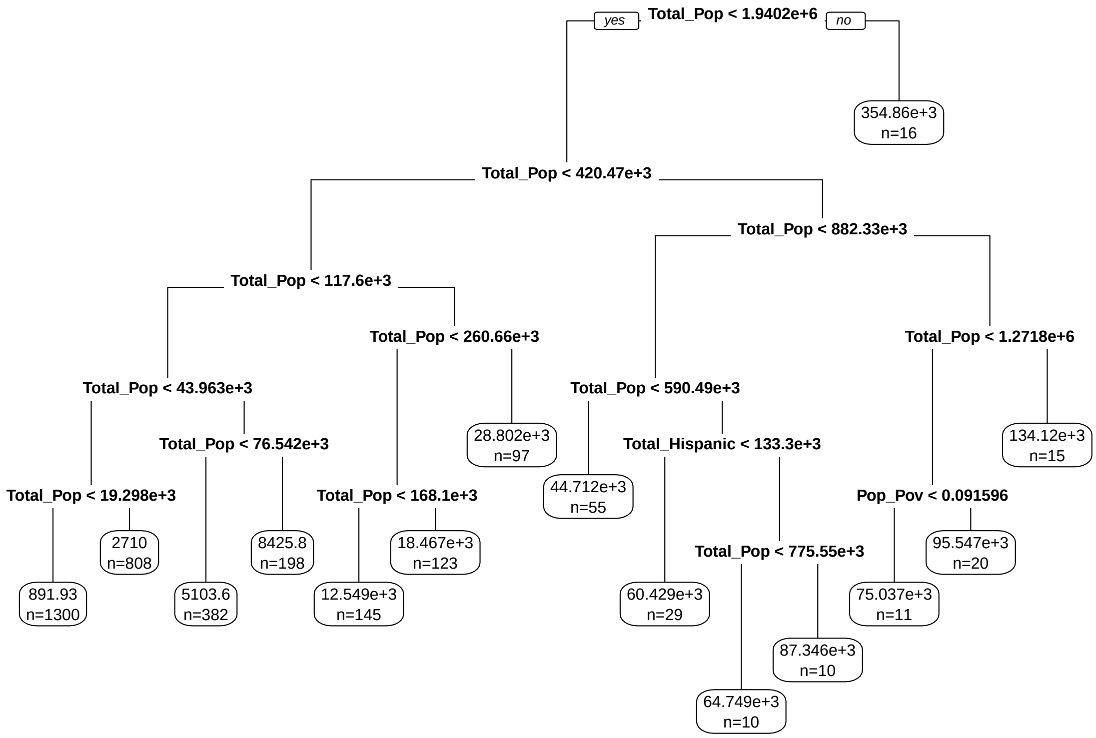
Lasso With Best CV Error Lambda

```r
lasso.fit<-glmnet(x,y,alpha=1)
lasso.coef<-predict(lasso.fit,type="coefficients",
                    s=cv.lam.lasso[which.min(cv.error.lasso)])[1:7,]
lasso.coef[lasso.coef!=0]
```

```
##    (Intercept)      Total_Pop    Total_White    Total_Black Total_Hispanic
##   363.98561597     0.04640675     0.03016595     0.02550912     0.08893831
```

Regression Trees

```r
tree_dat <- dat %>%
  subset(select = c(4:10,14))

tree <- rpart(cum_case ~.,
              data=tree_dat,control=rpart.control(cp=.0001))

#printcp(tree)

best <- tree$cptable[which.min(tree$cptable[,"xerror"]),"CP"]
pruned_tree <- prune(tree, cp=best)
prp(pruned_tree,
    faclen=0, #use full names for factor labels
    extra=1, #display number of obs. for each terminal node
    roundint=F, #don't round to integers in output
    branch = 1,
    varlen = 0,
    digits=5) #display 5 decimal places in output
```


Total_Pop < 1.9402e+6
yes
no
354.86e+3
n=16
Total_Pop < 420.47e+3
Total_Pop < 882.33e+3
Total_Pop < 117.6e+3
Total_Pop < 260.66e+3
Total_Pop < 1.2718e+6
Total_Pop < 43.963e+3
Total_Pop < 590.49e+3
Total_Pop < 76.542e+3
28.802e+3
n=97
Total_Hispanic < 133.3e+3
134.12e+3
n=15
Total_Pop < 19.298e+3
Total_Pop < 168.1e+3
44.712e+3
n=55
Pop_Pov < 0.091596
2710
n=808
8425.8
n=198
18.467e+3
n=123
Total_Pop < 775.55e+3
95.547e+3
n=20
891.93
n=1300
5103.6
n=382
12.549e+3
n=145
60.429e+3
n=29
75.037e+3
n=11
87.346e+3
n=10
64.749e+3
n=10