Linear Models and Extensions

Ruben Solis

Due: 15 April 2020

Contents

1 Data Preparation

We begin our venture by bringing in the data into R.

```
dat <- read.csv("crime.csv", header = TRUE, sep = ",")
```

1.1 Preparing the data

We next remove the first five columns of the data, as they are not predictive

```
dat <- dat[, -c(1:5)]
```

Next we inspect the variables to identify which ones are missing more than 60% of their values.

```
vnames <- colnames(dat)
n <- nrow(dat)
out <- NULL
for (j in 1:ncol(dat)){
  vname <- colnames(dat)[j]
  x <- as.vector(dat[,j])
  n1 <- sum(is.na(x), na.rm = TRUE)
  n2 <- sum(x=="NA", na.rm = TRUE)
  n3 <- sum(x=="", na.rm = TRUE)
  nmiss <- n1 + n2 + n3
  miss.perc <- nmiss/n
  if (miss.perc >= 0.60){
    out <- rbind(out, c(vname, miss.perc))
  }
}
out <- as.data.frame(out)
row.names(out) <- NULL
colnames(out) <- c("Variable", "Missing Proportion")
out
```

```
##                Variable Missing Proportion
## 1          LemasSwornFT  0.840020060180542
## 2       LemasSwFTPerPop  0.840020060180542
## 3     LemasSwFTFieldOps  0.840020060180542
## 4  LemasSwFTFieldPerPop  0.840020060180542
## 5         LemasTotalReq  0.840020060180542
## 6     LemasTotReqPerPop  0.840020060180542
## 7      PolicReqPerOffic  0.840020060180542
## 8           PolicPerPop  0.840020060180542
## 9    RacialMatchCommPol  0.840020060180542
## 10        PctPolicWhite  0.840020060180542
## 11        PctPolicBlack  0.840020060180542
## 12         PctPolicHisp  0.840020060180542
## 13        PctPolicAsian  0.840020060180542
## 14        PctPolicMinor  0.840020060180542
## 15  OfficAssgnDrugUnits  0.840020060180542
## 16    NumKindsDrugsSeiz  0.840020060180542
## 17     PolicAveOTWorked  0.840020060180542
## 18            PolicCars  0.840020060180542
## 19        PolicOperBudg  0.840020060180542
```

```
## 20  LemasPctPolicOnPatr  0.840020060180542
## 21  LemasGangUnitDeploy  0.840020060180542
## 22      PolicBudgPerPop  0.840020060180542
```

Now we need to find the column numbers of these variables to make their deletion easier. We accomplish this task with the following code.

```
delete.index <- NULL
for(k in 1:length(colnames(dat))){
  if (colnames(dat)[k] %in% out$Variable) {
    delete.index <- c(delete.index, k)
  }
}
delete.index
```

```
##  [1]  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 117 118
## [20] 119 120 122
```

Next, we delete those columns from the data set.

```
dat <- dat[, -c(delete.index)]
```

Before we impute any missing data, we need to make sure that our target variable `ViolentCrimePerPop` does not contain any missing values.

```
sum(is.na(dat$ViolentCrimesPerPop))
```

```
## [1] 0
```

We see that `ViolentCrimesPerPop` does not contain any missing values, so we proceed with the imputation.

```
library(mice, quietly = TRUE)
```

```
##
## Attaching package: 'mice'
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
fit.mice <- mice(dat, m=1, maxit = 50, method = 'pmm', seed=500, printFlag = FALSE)
```

```
## Warning: Number of logged events: 50
```

```
dat.imp <- complete(fit.mice, 1)
```

1.2 Exploratory Data Analysis

We next perform some minor exploratory data analysis. We plot the response variable `Violent CrimesPerPop` in the following histogram.

```
hist(dat.imp$ViolentCrimesPerPop, main="ViolentCrimesPerPop", xlab="Value")
box()
```

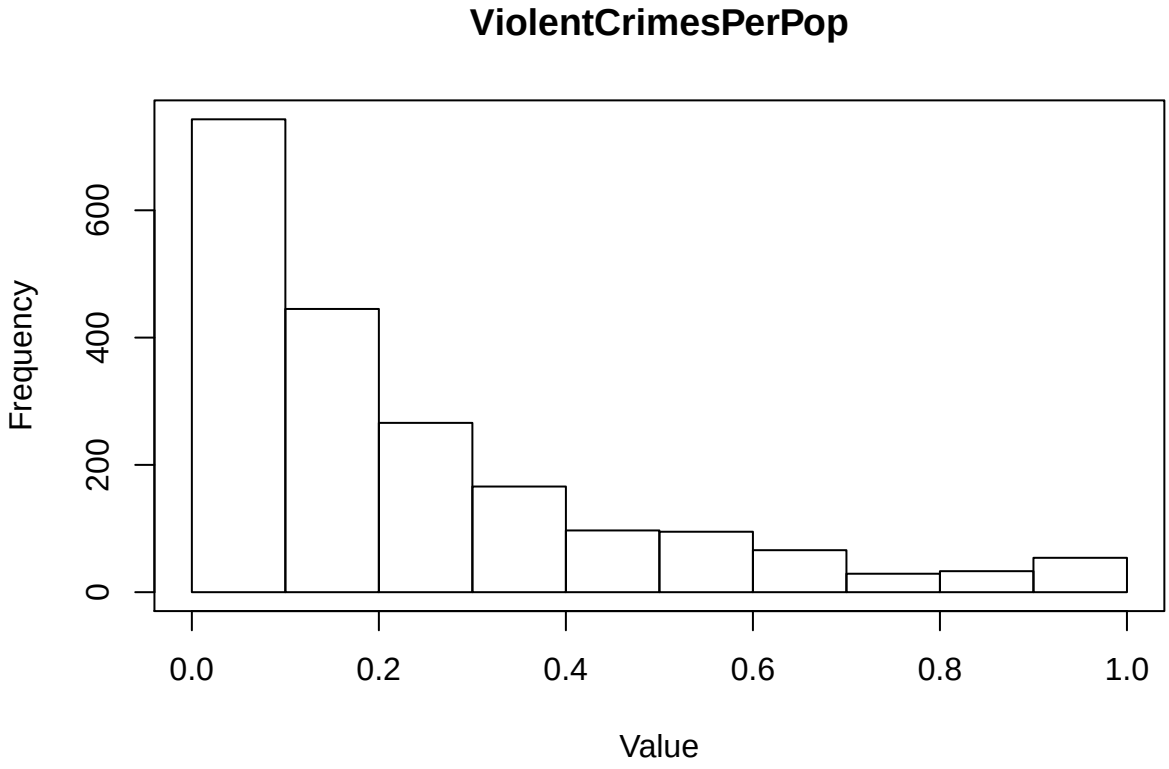

Figure 1: Histogram of our target variable ViolentCrimesPerPop

We see in Figure 1 that the response appears to be approximately exponentially distributed, but we do not perform a transformation on it.

```
qqnorm(dat.imp$ViolentCrimesPerPop)
qqline(dat.imp$ViolentCrimesPerPop)
```

[1]

[1]There were issues regarding the transformation. The response column did have a 0 which would automatically cause domain error if we used a log transformation. If we use say $\log(u) := \log(x+1)$ then we

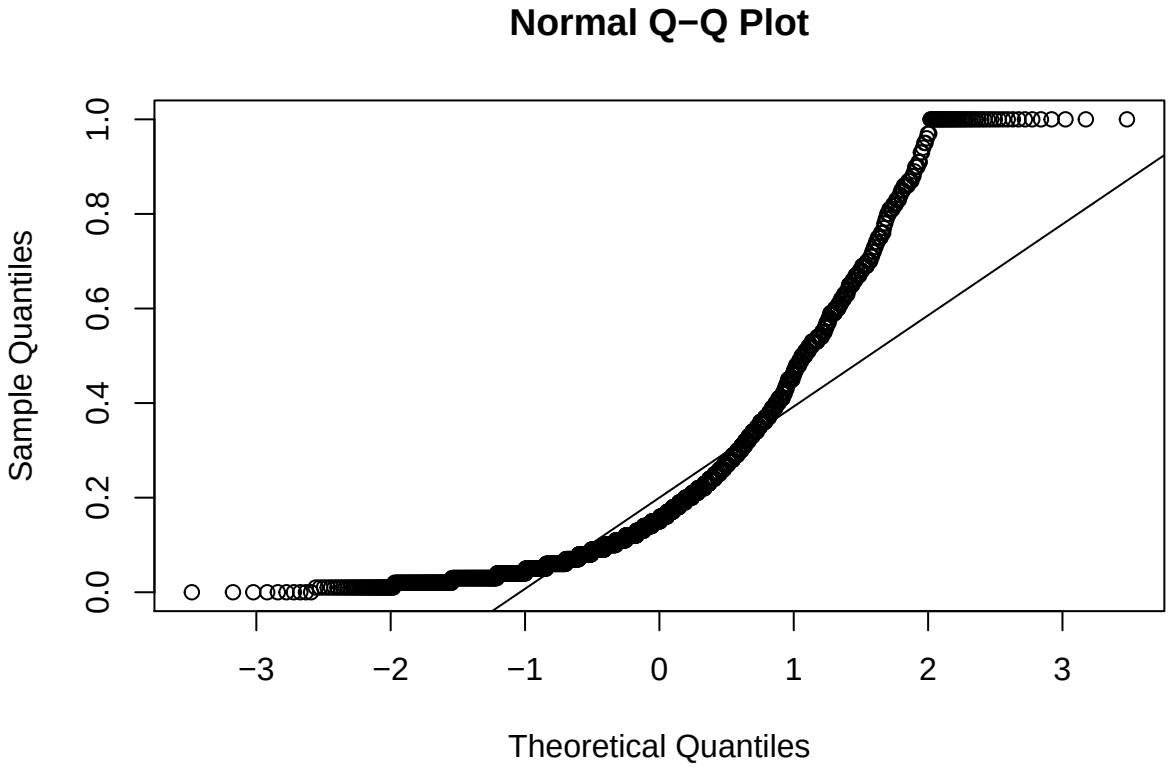

Figure 2: Q-Q plot of our target variable ViolentCrimesPerPop

1.3 Partitioning the data

We first start by partitioning the data into two sets: the training set $\mathcal{D}_1$ and the test set $\mathcal{D}_2$ with a ratio 2:1.

```
set.seed(5474)
training.data.index <- sample(1:nrow(dat.imp), 0.667*nrow(dat.imp))
training <- dat.imp[training.data.index, ]
test <- dat.imp[-training.data.index, ]
```

We will use the training data to fit the models on the different variable selection methods, and then use the testing data to arrive at a "best" model.

2 Linear Regression

2.1 Variable selection methods

We will deploy three different variable selection methods to our data: backward elimination, forward addition, and the least absolute shrinkage and selection operator (LASSO). We hope to find a "best model" and then see how well this "best model" performs given new data. We will compare their peformance with mean square error of prediction (MSEP).

2.1.1 Backward elimination

Our first venture into variable selection is to employ backward elimination. We start by first fitting the full model that contain all the predictors.

```
fit.full <- lm(ViolentCrimesPerPop ~., data = training)
```

With this full model, we proceed then proceed by deleting one predictor at-a-time until a optimal (minimum) AIC is computed. The model with this minimum will be considered the best model from backward elimination.

```
fit.backward <- step(fit.full, direction="backward", k=log(NROW(dat)), trace=FALSE)
summary(fit.backward)
```

```
##
## Call:
## lm(formula = ViolentCrimesPerPop ~ racepctblack + numbUrban +
##     pctUrban + pctWWage + pctWRetire + PctPopUnderPov + MalePctDivorce +
##     PctKids2Par + PctWorkMom + PctIlleg + PctPersDenseHous +
##     HousVacant + RentLowQ + MedRent + MedRentPctHousInc + MedOwnCostPctIncNoMtg +
```

have introduced some zeroes into the dataset when we evaluate log(1). Is there a way we can circumvent this problem? Perhaps a different transformation?

```
##     NumStreet, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52070 -0.07263 -0.01459  0.04948  0.70440
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)             0.49707    0.08914   5.576 2.98e-08 ***
## racepctblack            0.21657    0.02827   7.662 3.54e-14 ***
## numbUrban              -0.19528    0.07046  -2.771 0.005662 **
## pctUrban                0.04192    0.01055   3.975 7.43e-05 ***
## pctWWage               -0.11941    0.03831  -3.117 0.001866 **
## pctWRetire             -0.13334    0.03274  -4.073 4.93e-05 ***
## PctPopUnderPov         -0.20962    0.04648  -4.509 7.08e-06 ***
## MalePctDivorce          0.12752    0.03718   3.430 0.000623 ***
## PctKids2Par            -0.35759    0.07491  -4.774 2.01e-06 ***
## PctWorkMom             -0.11848    0.02684  -4.415 1.09e-05 ***
## PctIlleg                0.15619    0.04682   3.336 0.000874 ***
## PctPersDenseHous        0.20492    0.02652   7.727 2.18e-14 ***
## HousVacant              0.26815    0.05699   4.705 2.80e-06 ***
## RentLowQ               -0.23169    0.06198  -3.738 0.000193 ***
## MedRent                 0.20217    0.06496   3.112 0.001896 **
## MedRentPctHousInc       0.08967    0.03101   2.891 0.003899 **
## MedOwnCostPctIncNoMtg -0.09290    0.02135  -4.350 1.46e-05 ***
## NumStreet               0.24283    0.05147   4.718 2.64e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.134 on 1311 degrees of freedom
## Multiple R-squared:  0.679,  Adjusted R-squared:  0.6748
## F-statistic: 163.1 on 17 and 1311 DF,  p-value: < 2.2e-16
```

```
yhat.backward <- predict(fit.backward, newdata = test)
yobs <- test[, 101]
MSEP.backward <- mean((yobs - yhat.backward)^2)
```

From the `summary()` output, we see that the best model discovered via backward elimination. That is from the 100 possible predictors, backward elimination selected 21 of them to be included in the model. These predictors and their coefficients are listed above. It is interesting to note that `HousVacant` is significant when determining `ViolentCrimesPerPop`.

2.1.2 Forward addition

The variable selection we will employ is Forward addition. Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all the predictors are in the model. In particular, at each step, the variable that gives the greasest additional improvement to the fit is added to the model.

```
fit.null <- lm(ViolentCrimesPerPop ~ 1, data=training)
fit.forward <- step(fit.null, scope=list(lower=fit.null, upper=fit.full),
    direction="forward", k=log(NROW(dat)), trace=FALSE)
summary(fit.forward)
```

```
##
## Call:
## lm(formula = ViolentCrimesPerPop ~ PctKids2Par + racePctWhite +
##     HousVacant + NumStreet + pctUrban + agePct16t24 + PctPersDenseHous +
##     racepctblack + PctWorkMom + pctWRetire + MedOwnCostPctIncNoMtg +
##     PctPopUnderPov + MedRentPctHousInc + numbUrban + PctIlleg,
##     data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.54316 -0.07199 -0.01395  0.04916  0.73140
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.57613    0.07576   7.605 5.41e-14 ***
## PctKids2Par           -0.53445    0.05127 -10.425  < 2e-16 ***
## racePctWhite           0.03997    0.05101   0.784 0.433466
## HousVacant             0.29995    0.05696   5.266 1.63e-07 ***
## NumStreet              0.25793    0.05172   4.988 6.93e-07 ***
## pctUrban               0.03390    0.01044   3.246 0.001198 **
## agePct16t24           -0.10423    0.02846  -3.663 0.000259 ***
## PctPersDenseHous       0.18052    0.04044   4.463 8.76e-06 ***
## racepctblack           0.21379    0.04523   4.727 2.52e-06 ***
## PctWorkMom            -0.11510    0.02632  -4.373 1.32e-05 ***
## pctWRetire            -0.09406    0.02601  -3.616 0.000310 ***
## MedOwnCostPctIncNoMtg -0.09710    0.02094  -4.637 3.88e-06 ***
## PctPopUnderPov        -0.12873    0.03775  -3.410 0.000669 ***
## MedRentPctHousInc      0.08788    0.02740   3.207 0.001373 **
## numbUrban             -0.22761    0.07066  -3.221 0.001307 **
## PctIlleg               0.12585    0.04489   2.804 0.005125 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.1349 on 1313 degrees of freedom
## Multiple R-squared:  0.6741, Adjusted R-squared:  0.6703
## F-statistic:   181 on 15 and 1313 DF,  p-value: < 2.2e-16
```

```
yhat.forward <- predict(fit.forward, newdata = test)
MSEP.forward <- mean((yobs - yhat.forward)^2)
```

From the `summary()` output, we see the best model discovered via forward addition. That is from the 100 possible predictors, forward addition selected the 12 aboveto be included in the model. It is interesting to note that `racePctWhite` is not significant when determining `ViolentCrimesPerPop` yet it was included in the model. [2]

2.1.3 LASSO

Our next final model selection method, for this section, is the LASSO. LASSO acts as variable selection method as some of the coefficients will shrink towards zero. In addition to the explanation given in Hastie, Tibshirani, and Friedman (2009), another way to view LASSO is in the Bayesian context where we assume that the β's in the model have a prior distribution of the double exponential.

While normally one would want to standardize the data prior to fitting a LASSO model, we do not need to do that in the following code, as the functions available in the `ncvreg` package do this for us.

```
library(ncvreg)
set.seed(5474)
fit.lasso <- cv.ncvreg(X=training[,-101], y=training[,101],
                       nfolds = 10, family="gaussian",
                       penalty="lasso", lamda.min=.005, nlambda=100, eps=.001,
                       max.iter=1000)
beta.hat <- coef(fit.lasso)
cutoff <- 0
terms <- names(beta.hat)[abs(beta.hat) > cutoff]
formula.LASSO <- as.formula(paste(c("ViolentCrimesPerPop ~ ", terms[-1]),
                                  collapse = "+"))
fit.lasso <- lm(formula.LASSO, data = training)
summary(fit.lasso)
```

```
##
## Call:
## lm(formula = formula.LASSO, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

[2] This is something I intend to explore further.

```
## -0.52970 -0.07460 -0.01310  0.04879  0.70094
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              0.493078   0.111546   4.420 1.07e-05 ***
## racepctblack             0.199623   0.033907   5.887 5.00e-09 ***
## racePctHisp              0.033418   0.038841   0.860 0.389734
## agePct12t29             -0.089107   0.055513  -1.605 0.108702
## agePct65up               0.080853   0.050988   1.586 0.113047
## pctUrban                 0.029302   0.011211   2.614 0.009064 **
## pctWFarmSelf             0.034517   0.023519   1.468 0.142450
## pctWInvInc              -0.159478   0.054474  -2.928 0.003475 **
## pctWRetire              -0.103108   0.036619  -2.816 0.004941 **
## indianPerCap            -0.047041   0.024511  -1.919 0.055180 .
## AsianPerCap              0.024475   0.023165   1.057 0.290922
## OtherPerCap              0.030482   0.022670   1.345 0.179000
## HispPerCap               0.014078   0.028470   0.494 0.621049
## PctPopUnderPov          -0.158352   0.051762  -3.059 0.002265 **
## PctEmplManu             -0.014945   0.023874  -0.626 0.531433
## MalePctDivorce           0.043795   0.045422   0.964 0.335134
## PctKids2Par             -0.345863   0.081311  -4.254 2.26e-05 ***
## PctWorkMom              -0.107250   0.028829  -3.720 0.000208 ***
## PctIlleg                 0.178093   0.049315   3.611 0.000316 ***
## NumImmig                -0.194807   0.080248  -2.428 0.015337 *
## PctImmigRec10            0.020123   0.029245   0.688 0.491519
## PctRecImmig10            0.004128   0.040134   0.103 0.918088
## PctPersDenseHous         0.101708   0.057914   1.756 0.079291 .
## PctHousLess3BR           0.032138   0.045301   0.709 0.478185
## HousVacant               0.136195   0.042850   3.178 0.001516 **
## PctHousOccup            -0.022211   0.030297  -0.733 0.463609
## PctVacantBoarded         0.019233   0.025121   0.766 0.444037
## PctVacMore6Mos          -0.019892   0.028818  -0.690 0.490158
## RentHighQ                0.016661   0.082245   0.203 0.839493
## MedRent                 -0.011019   0.092204  -0.120 0.904890
## MedRentPctHousInc        0.099142   0.033027   3.002 0.002735 **
## MedOwnCostPctIncNoMtg -0.106002   0.024312  -4.360 1.40e-05 ***
## NumStreet                0.295590   0.057016   5.184 2.51e-07 ***
## PctBornSameState        -0.017119   0.030216  -0.567 0.571119
## PctSameCity85            0.041625   0.034678   1.200 0.230231
## LemasPctOfficDrugUn      0.027797   0.018528   1.500 0.133789
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1341 on 1293 degrees of freedom
## Multiple R-squared:  0.6825, Adjusted R-squared:  0.6739
```

```
## F-statistic: 79.42 on 35 and 1293 DF,  p-value: < 2.2e-16
```

```
yhat.lasso <- predict(fit.forward, newdata = test)
MSEP.lasso <- mean((yobs - yhat.lasso)^2)
```

From the `summary()` output, we see that the best model discovered by LASSO contains far more predictors than the other two methods. Again, we notice that many of them are not significant.

2.1.4 Comparing the methods

Finally we compare the performance of the two models with respect to the prediction mea square error (MSEP). This comparison can be seen below.

```
MSEP <- c(MSEP.backward, MSEP.forward, MSEP.lasso)
MSEP <- as.data.frame(t(t(MSEP)))
MSEP$er <- c("Backward", "Forward", "LASSO")
names(MSEP) <- c("MSEP", "Method")
MSEP
```

```
##         MSEP   Method
## 1 0.01826980 Backward
## 2 0.01884877  Forward
## 3 0.01884877    LASSO
```

From the output me wee that the model that performed the best was backward followed by a tie between lasso and forward. In this case, it would probably be better to go with the more parsimonious model, Forward model, rather than the LASSO model.

3 Extended linear modeling

3.1 Ridge Regression

The first model we explore is ridge regression. We need to select a tuning parameter, and this parameter is chosen with cross validation. The code for this model is below.

```
suppressMessages(library(glmnet, quietly = TRUE))
lambda <- seq(0, 80.0, 0.01)
X <- as.matrix(training[, -101]); y <- training[, 101]
cv.RR <- cv.glmnet(x=X, y=y, alpha = 0, lambda = lambda, nfolds = 10)
lmbd0 <- cv.RR$lambda.min # MINIMUM CV ERROR
fit.RR <- cv.RR$glmnet.fit
y.pred <- predict(fit.RR, s=lmbd0, newx = as.matrix(test[, -101]))
yobs <- test[, 101]
MSEP.RR <- mean((yobs - y.pred)^2)
```

3.2 Principal components regression (PCR)

The second model that we fit is principal components regression. We need to determine the number of components that will yield the best model, and this is accomplished with cross validation. The code for this model is below.

```
suppressMessages(library(pls, quietly = TRUE))
fit.PCR.best <- pcr(training[, 101] ~ ., ncomp=16, data=training, method = pls.options()
    validation = "CV", segments = 10,  segment.type ="random", scale=TRUE)
CV <- fit.PCR.best$validation
ncomp.best <- which.min(CV$PRESS)
# PREDICTION
yhat.PCR <- predict(fit.PCR.best, newdata = test, comps = 1:ncomp.best)
yobs <- test[,101]
# MEAN SQUARE ERROR FOR PREDICTION
MSEP.PCR <- mean((yobs-yhat.PCR)^2)
```

3.3 Partial least squares regression (PLSR)

The third model that we fit is partial least squares regression. We need to determine the number of components that will yield the best model, and this is accomplished with cross validation. The code for this model is below.

```
suppressMessages(library(pls, quietly = TRUE))
fit.PLS <- plsr(ViolentCrimesPerPop ~., ncomp=100, data=training, method="simpls",
                validation = "CV", segments = 10, segment.type = "random", scale=TRUE)
CV <- fit.PLS$validation
ncomp.best <- which.min(CV$PRESS)
# MAKE PREDICTION
yhat.PLSR <- predict(fit.PLS, newdata = test, comps = 1:ncomp.best)
yobs <- test[,101]
# MAKE SQUARE ERROR FOR PREDICTION
MSEP.PLSR <- mean((yobs - yhat.PLSR)^2)
```

3.4 Total least squares regression (TSLR)

The fourth model that we fit is total least squares regression. The code for this model is below.

```
suppressMessages(library(pracma, quietly = TRUE))
X <- as.matrix(training[, -101]); y <- training[, 101]
fit.TLS <- odregress(x=X, y=y)
beta <- fit.TLS$coeff
Xnew <- as.matrix(test[, -101])
```

```
yhat.TLS <- cbind(Xnew, 1) %*% beta
# MEAN SQUARE ERROR FOR PREDICTION
MSEP.TLS <- mean((yobs - yhat.TLS)^2);
```

3.5 Least angle regression

The fifth, and final, model that we fit is least angle regression. The code for this model is below.

```
suppressMessages(library(lars, quietly = TRUE))
X <- as.matrix(training[, -101]); y <- training[, 101]
fit.lar <- lars(X, y, type = "lar", trace = FALSE, normalize = TRUE, intercept = TRUE)
# TO FIND THE L1-NORM
BETA <- as.matrix(fit.lar$beta)
L1.norm <- function(x) sum(abs(x))
norm.L1 <- apply(BETA, 1, L1.norm)
norm.L1 <- norm.L1/max(norm.L1)
df <- as.vector(fit.lar$df)
Cp <- as.vector(fit.lar$Cp)
RSS <- as.vector(fit.lar$RSS)
lambda <- c(as.vector(fit.lar$lambda), 0)
b.lambda <- lambda[Cp==min(Cp)]
b.L1norm <- norm.L1[Cp==min(Cp)]
b.max.steps <- 5
yhat.lar <- predict(fit.lar, s=b.max.steps, newx=as.matrix(test[, -101]))$fit
MSEP.lar <- mean((yobs-yhat.lar)^2);
```

4 Results

Finally, we can view the performance of the models as measured with mean square error of prediction (MSEP). These results are below.

```
MSEP <- c(MSEP.RR, MSEP.PCR, MSEP.PLSR, MSEP.TLS, MSEP.lar)
MSEP <- as.data.frame(t(t(MSEP)))
MSEP$er <- c("RR", "PCR", "PLSR", "TLS", "LAR")
names(MSEP) <- c("MSEP", "Method")
MSEP
```

```
##          MSEP Method
## 1  0.01787837     RR
## 2  0.01604643    PCR
## 3  0.01856683   PLSR
## 4 20.87924935    TLS
```

```
## 5  0.02238127    LAR
```

We see that the performance of all of the methods are fairly similar except for total least squares regression that has a MSEP of 18.32. It appears that PCR performed the best out of these models.

5 Discussion

One interesting finding with this data set deals with the demographics of law enforcement officers. Some of the variables removed from the data set prior to fitting the model were the ethnicity of the police. It would be interesting to see if such data has been recorded in recent years and what impact, if any, it has had on violent crime rate because it has been suggested in some political circles that a police for ethnically representative of the are it serves can decrease overall crime.