ISYE6501 HW5

June 20, 2018

Question 11.1

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

1. Stepwise regression - I have omitted the output here since it's not particularly interesting. I ran the three different stepwise regression techniques as in Andrew's recitation video, but I will show the models found by each technique:

```
#forwardStep <- lm(formula = Crime ~ Po1 + Ineq + Ed + M + Prob + U2, data = crime)
#backwardStep <- lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob, data=crime)
#bothStep <- lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob, data=crime)
```

2. Lasso

Standardizing the relevant data and converting to a matrix:

```
#Removing columns 2 and 16:
Crime <- crime[,16]
So <- crime[,2]
crime<-crime[,-16]
crime<-crime[,-2]

#Scaling with relevant variables removed
scaledCrime <- scale(crime)

#Adding the variables back:
scaledCrime <- cbind(scaledCrime, So, Crime)

#Convert to a matrix
matrixCrime <- as.matrix(scaledCrime)
```

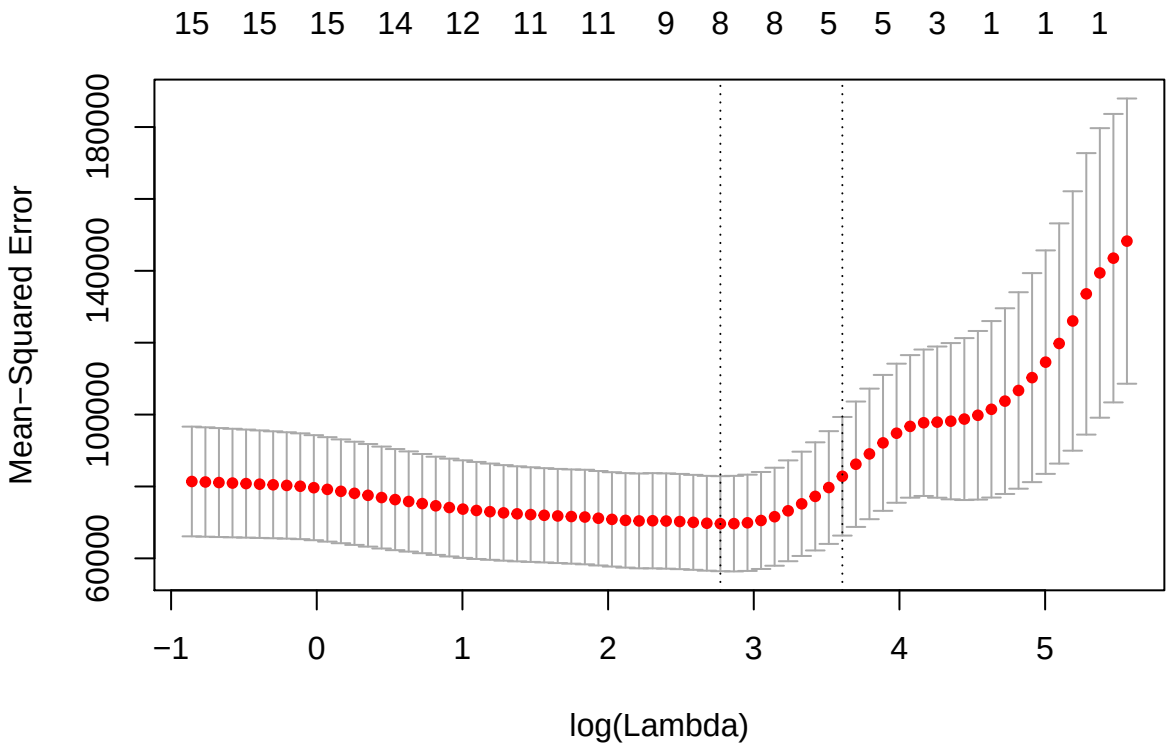
Implementing Lasso:

```
set.seed(42)
#Run cv.glmnet with alpha=1 to obtain optimal lambda
cvlasso <- cv.glmnet(x=matrixCrime[,1:15],y=matrixCrime[,16],
                     type.measure="mse", standardize=FALSE, family="gaussian",
                     alpha=1)

#Run lasso
lasso<-glmnet(matrixCrime[,1:15],matrixCrime[,16],
              lambda = cvlasso$lambda.1se, standardize=FALSE, family="gaussian",
              alpha=1)

#Extract coefficients of this model
coef(lasso)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                     s0
## (Intercept) 905.08511
## M            38.16771
## Ed            .
```

```
## Po1          266.05157
## Po2            .
## LF             .
## M.F           45.66491
## Pop            .
## NW             .
## U1             .
## U2             .
## Wealth         .
## Ineq          64.02517
## Prob         -42.95943
## Time           .
## So             .
```

```
#Plot mse vs lambda
plot(cvlasso)
```




```
#Minimum lambda found
cvlasso$lambda.min
```

```
## [1] 15.97063
```

3. Elastic net - please see reference at end of document

```
#Set a range of alpha values to search over
a <- seq(0.001, 0.999, 0.005)

#Elastic net implementation
```

```
search <- foreach(i = a, .combine = rbind) %do% {
  cv <- cv.glmnet(matrixCrime[,1:15], matrixCrime[,16], family = "gaussian",
                  standardize=FALSE, nfold = 10, type.measure = "deviance", alpha = i)
  data.frame(cvm = cv$cvm[cv$lambda == cv$lambda.1se], lambda.1se = cv$lambda.1se, alpha = i)
}
cv3 <- search[search$cvm == min(search$cvm), ]
md3 <- glmnet(matrixCrime[,1:15], matrixCrime[,16], family = "gaussian",
              standardize=FALSE, lambda = cv3$lambda.1se, alpha = cv3$alpha)

#Print optimal lambda and alpha value:
#Lambda
cv3$lambda.1se
```

```
## [1] 22.01978
```

```
#alpha
cv3$alpha
```

```
## [1] 0.796
```

```
#Model coefficients
coef(md3)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## (Intercept) 905.085106
## M            68.579092
## Ed           72.357555
## Po1         295.772314
## Po2           2.122702
## LF            .
## M.F          49.734142
## Pop           .
## NW           14.348356
## U1            .
## U2           20.919000
## Wealth        .
## Ineq        141.849874
## Prob        -70.460643
## Time          .
## So            .
```

We list the final models here and then perform cross validation to determine the best performer. I will note that I made the decision not to whittle down the models further based on p-value because I wanted to see how each technique compared to each other without modification. We use the DAAG package with built-in cross validation feature:

```
forwardStep <- lm(formula = Crime ~ Po1 + Ineq + Ed + M + Prob + U2, data = crime)
backwardStep <- lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob, data=crime)
bothStep <- lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob, data=crime)
lassoModel <- lm(Crime ~ M + Po1 +M.F + Ineq + Prob, data=crime)
eNetModel <- lm(Crime ~  M + Ed + Po1+ Po2 + M.F + NW + U2 + Ineq + Prob, data=crime)
```

Now we calculate adjusted R squared for each model. In the interest of brevity I will simply list the adjusted R squared of the models, as you probably don't want to look at 30 pages of output (please see the R file to verify that it's done correctly though!)

Forward stepwise R^2_{adj}:

```
AdjR2_1
```

```
## [1] 0.644
```

Backwards / both stepwise R^2_{adj}:

```
AdjR2_2
```

```
## [1] 0.628
```

Lasso regression R^2_{adj}:

```
AdjR2_3
```

```
## [1] 0.589
```

Elastic net R^2_{adj}:

```
AdjR2_4
```

```
## [1] 0.576
```

We see that the optimal model is Model 1, found with forward stepwise selection.

Question 12.1

Describe a situation or problem from your job, everyday life, current events, etc., for which a design of experiments approach would be appropriate.

Answer: We want to determine the relative value of binary features of desktop computers using 12 binary features such as best processor, best graphics card, best memory etc. Because there are 2^{12} combinations of these features, we need to use a fractional factorial design to help us test a subset of these items while still ensuring we are testing a good selection of features. We could test 50 different computer configurations on consumers using $FrF2$ to get an idea of how the public rates these features.

Question 12.2

To determine the value of 10 different yes/no features to the market value of a house (large yard, solar roof, etc.), a real estate agent plans to survey 50 potential buyers, showing a fictitious house with different combinations of features. To reduce the survey size, the agent wants to show just 16 fictitious houses. Use R's FrF2 function (in the FrF2 package) to find a fractional factorial design for this experiment: what set of features should each of the 16 fictitious houses have? Note: the output of FrF2 is "1" (include) or "-1" (don't include) for each feature.

```
set.seed(42)

#nruns=16 nfactors=10
FrF2(nruns=16, nfactors=10)
```

```
##     A  B  C  D  E  F  G  H  J  K
## 1  -1  1  1  1 -1 -1  1 -1  1 -1
## 2   1  1  1  1  1  1  1  1  1  1
## 3  -1 -1  1 -1  1 -1 -1  1  1 -1
## 4  -1  1 -1  1 -1  1 -1 -1 -1  1
## 5   1  1  1 -1  1  1  1 -1 -1 -1
## 6   1 -1  1 -1 -1  1 -1 -1  1  1
## 7   1  1 -1  1  1 -1 -1  1 -1 -1
```

```
## 8   1 -1 -1 -1 -1 -1  1 -1 -1 -1
## 9  -1 -1  1  1  1 -1 -1 -1 -1  1
## 10  1 -1  1  1 -1  1 -1  1 -1 -1
## 11 -1  1 -1 -1 -1  1 -1  1  1 -1
## 12  1  1 -1 -1  1 -1 -1 -1  1  1
## 13 -1  1  1 -1 -1 -1  1  1 -1  1
## 14 -1 -1 -1 -1  1  1  1  1 -1  1
## 15  1 -1 -1  1 -1 -1  1  1  1  1
## 16 -1 -1 -1  1  1  1  1 -1  1 -1
## class=design, type= FrF2
```

Question 13.1

For each of the following distributions, give an example of data that you would expect to follow this distribution (besides the examples already discussed in class).

a. Binomial: In curriculum design for a school district, we would like to see if the current year's material was well-received by sending out a survey to students and parents. The number of positive responses recorded in this survey would have a Binomial distribution. We could even estimate the success rate ahead of time based on perceived reception of the curriculum.

b. Geometric: The geometric distribution represents the number of failures before you get a success in a series of Bernoulli trials. In baseball, the number of strikeouts of a player before a home run would follow the Geometric distribution.

c. Poisson: An interesting fact that I read on twitter was that one of the first practical applications of the Poisson distribution was made by Ladislaus Bortkiewicz in 1898 when he was given the task of investigating the number of soldiers in the Prussian army killed accidentally by horse kicks. You may read more about this and the connection to the Poisson distribution here: http://blog.minitab.com/blog/quality-data-analysis-and-statistics/no-horsing-around-with-the-poisson-distribution-troops

d. Exponential: The exponential distribution may be viewed as a continuous counterpart of the geometric distribution, which describes the number of Bernoulli trials necessary for a discrete process to change state. In contrast, the exponential distribution describes the time for a continuous process to change state. Continuing our baseball scenario, the amount of time it takes until a player hits a home run could be modeled by an exponential distribution.

e. Weibull: A more general version of the exponential distribution, where the failures can either be decreasing, constant or increasing over time. Continuing our baseball example if a player got "hot" and started hitting home runs with an increased frequency, we could model this with a Weibull distribution with parameter $k < 1$ - failures are decreasing over time.

References

11.1.1: https://www.statmethods.net/stats/regression.html

11.1.3: https://www.r-bloggers.com/variable-selection-with-elastic-net/