ISYE 6501 Homework 5

Due: June 24, 2021

1 Variable Selection in Regression

1.1 Question 11.1

Limiting the number of factors in a factor-based model can be beneficial for several reasons, namely, it avoids overfitting and allows for simplicity for easier interpretation and less of a chance of including insignificant factors. Using the same *uscrime* dataset from previous works, I aimed to create new, and ideally, better regression models through the use of three different variable selection techniques: stepwise regression, Lasso, and Elastic net.

To start, I completed a train/test (70%/30%) split on the data and ran normal linear regression on the data using all variables as a benchmark for accuracy. The results are summarized in Figure 1 but most notably, the sum of squared errors = 2269711 and the Akaike information criterion (AIC) $\approx$ 453.

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-385.90  -116.94    15.07    88.10   296.57

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.848e+03  2.409e+03  -0.767   0.4536
M            8.064e+01  4.471e+01   1.803   0.0891 .
So          -6.289e+01  1.778e+02  -0.354   0.7279
Ed           1.556e+02  7.383e+01   2.108   0.0502 .
Po1         -5.098e+01  1.619e+02  -0.315   0.7567
Po2          2.017e+02  1.811e+02   1.113   0.2810
LF          -1.426e+03  1.423e+03  -1.002   0.3303
M.F         -9.627e+00  2.502e+01  -0.385   0.7052
Pop         -3.492e-01  1.651e+00  -0.211   0.8350
NW          -1.665e+00  7.136e+00  -0.233   0.8183
U1          -5.634e+03  4.428e+03  -1.272   0.2204
U2           1.253e+02  9.210e+01   1.361   0.1914
Wealth      -1.954e-02  1.452e-01  -0.135   0.8945
Ineq         6.259e+01  2.794e+01   2.240   0.0388 *
Prob        -5.179e+03  2.426e+03  -2.135   0.0477 *
Time        -4.277e+00  7.302e+00  -0.586   0.5658
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 36839.76)

    Null deviance: 3418528  on 32  degrees of freedom
Residual deviance:  626276  on 17  degrees of freedom
AIC: 452.73

Number of Fisher Scoring iterations: 2
```

Figure 1: Summary of a linear regression model using all factors.

Next, I took a stepwise regression approach to the problem which is a blend of forward selection and backwards elimination. I started with all factors as shown above and eliminate those whose p-value > 0.15. I found that only a few factors were found to be kept: M, Ed, Ineq, and Prob. I then ran the linear regression

model once again using solely these factors and find that only M and Prob have $p < 0.15$. Because two factors is my threshold for the question, "do we have enough factors?". I decided to stop here and once again run the linear regression model on the training data using these two predictors. Figure 2 summarizes the results.

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-527.12  -173.73   -49.27   112.33   640.20

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   740.65     549.68   1.347  0.18793
M              33.09      40.33   0.820  0.41843
Prob        -7234.06    2258.35  -3.203  0.00321 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 84904.64)

    Null deviance: 3418528  on 32  degrees of freedom
Residual deviance: 2547139  on 30  degrees of freedom
AIC: 473.03

Number of Fisher Scoring iterations: 2
```

Figure 2: Summary of a linear regression model using M and Ed factors.

Interestingly enough, this process actually increases both the AIC (473) and the sum of squared error (3163176) which led me to believe this process was taken one step too far and that the threshold for number of factors is too low. For this reason I reverted back to using the 4 factors and still found that the AIC and squared error was worse than using all 15 factors. This tells me that the threshold for initial inclusion ($p < 0.15$) might be too low. The reason I cannot truly claim that either of these models are "worse" than that when using all factors is because the first model may simply be overfit.

Next, I aimed to further improve the model via lasso regression where the sum of the model coefficients are capped at some arbitrary value. By using *glmnet* in R, we can automatically find this value that minimizes the mean squared error (MSE). I followed [1] for an useful example of this. See Appendix A for code. Figure 3 below shows the values for lambda that minimizes the shrinkage penalty and hence MSE after completing 10 fold cross validation on the training data. Note that data was scaled before training.

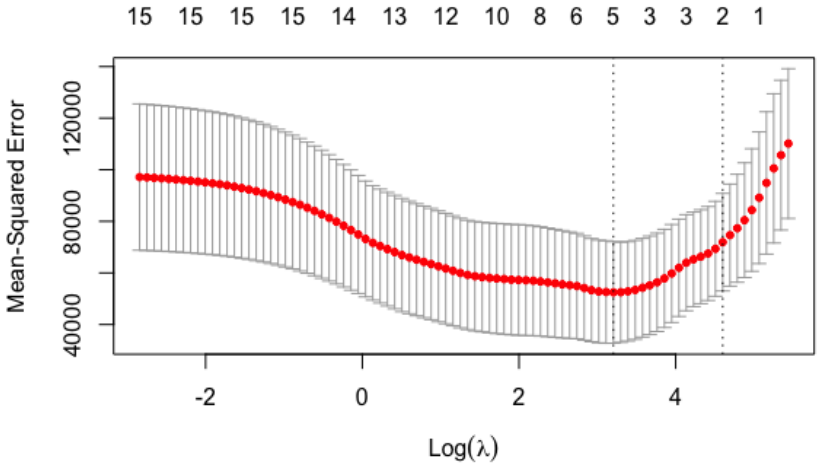

Figure 3: Optimizing lambda parameter for lasso regression.

The value for lambda that minimized the MSE was $\lambda = 24.65$. This optimization led to the inclusion of only factors Po2 and Prob which interestingly is different that the stepwise regression approach. Lastly, there is a drastic improvement on predictions of the test set. The sum of squared errors drops all the way to 21716.

Finally, I test to see if I can improve the model even more using elastic net regression. I follow [2] to test 50 combinations of lambda and alpha to find the coefficients that once again minimize the MSE but this time taking the squared coefficients into consideration when capping T. Note that data was once again scaled before training. After cross validation, the optimal parameters were found to be $\alpha = 0.32$ and $\lambda = 12.4$. It was fascinating to see that the only predictors NOT used in the optimized model were: So, M.F, and Wealth. The results after predictions using the new coefficients and test set data conclude and sum of squared error of 1577899. Therefore my conclusion in that the lasso regression performs best by far this is likely because using elastic net could underestimate coefficients of very predictive variables.

2 Design of Experiments

2.1 Question 12.1

I think a very natural example of a design of experiments approach and specifically the multi-armed bandit problem would be targeted advertisements on my social media feeds. Assuming that when I first joined, the media company had no apriori about my interests, there was likely some exploration vs. exploitation. Although I can't specifically remember, they probably showed me advertisements about clothing, food, technology, and other services etc. They want to maximize their expected payout which in this case is getting me to physically click on the ad. So with k alternatives (a very high number) they kept showing me new random ads until I finally clicked one, say, a clothing company. They then likely updated their probabilities and repeated the process but maybe using a subset with only clothing companies and a few other categories sprinkled in there on a probabilistic basis. This way they got more value out of the process by making it more likely to choose the option with the highest expected payout.

2.2 Question 12.2

To solve the factorial design problem of a real estate agent surveying 50 potential buyers but only wanting to show 16 fictitious homes (each with 10 yes/no features), the R package, $FrF2$ can complete this remarkably easy. See Appendix B for code. In theory, there are $2^{10} = 1024$ unique combinations which is not feasible to do in this real world instance so the goal is to test each choice as well as each pair of choices as best as possible. Figure 4 shows the results of the experimental design.

	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

Figure 4: Columns A-K are the 10 features and rows 1-16 are the unique choices for homes. 1 means to include the feature and -1 means to omit it.

3 Probability Distributions

3.1 Question 13.1

3.1.1 Binomial Distribution

The binomial distribution can be thought of as the probability of getting x successes out of n independent and identically distributed (i.i.d) Bernoulli(p). An example of this could be the number of people in a store of any kind that choose either purchase an item or not.

3.1.2 Geometric Distribution

The geometric distribution can be thought of as the probability of having Bernoulli(p) failures until the first success. An example of this could be the number of items a store sells before a customer returns something.

3.1.3 Poisson Distribution

The Poisson distribution can be thought of as the probability of x things arriving given the average number of arrival per time period. An example of this could be the number of people arriving at said store given the average arrival data per hour of the day.

3.1.4 Exponential Distribution

The exponential distribution can be thought of as the time between successive arrivals of Poisson(λ) arrivals. An example of this could be the time between customer arrivals to a store in the last example.

3.1.5 Weibull Distribution

The Weibull distribution can be thought of as the time between failures of something. An example of this could be the time between customers leaving a store and choosing not to purchase anything.

References

[1] https://www.statology.org/lasso-regression-in-r/

[2] http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/153-penalized-regression-essentials-ridge-lasso-elastic-net/#elastic-net

Appendix A Variable selection code

```
data <- read.table("uscrime.txt", header=TRUE)
print(head(data))

set.seed(123)
n <- nrow(data)
train_idx <- sample(1:n, size=round(n*0.7), replace = FALSE)
train <- data[train_idx,]
test <- data[-train_idx,]

model1 <- glm(Crime~., data=train)
summary(model1)
pred <- predict(model1, test)
print(pred)

#print(test[,16] - pred)
squared_err = sum((test[,16] - pred)^2)

keep <- c(1, 3, 14, 15, 16)

model2 <- glm(Crime~., data=train[,keep])
summary(model2)
pred2 <- predict(model2, test[,keep])
print(pred2)
squared_err = sum((test[,16] - pred2)^2)

# Lasso
library(glmnet)
lasso <- cv.glmnet(as.matrix(train[,1:15]), as.matrix(train[,16]),
                   alpha=1, standardize = TRUE)

print(lasso$lambda.min)
plot(lasso)
best_lambda <- lasso$lambda.min
best_lasso <- glmnet(as.matrix(train), as.matrix(train[,16]),
                   alpha=1, standardize = TRUE, lambda = best_lambda)
coef(lasso)
pred3 <- predict(best_lasso, as.matrix(test))
print(pred3)
squared_err = sum((test[,16] - pred3)^2)
```

```
#Elastic net
library(tidyverse)
library(caret)
elastic <- train(Crime ~., data = train, method = "glmnet",
  trControl = trainControl("cv", number = 10), tuneLength = 50)
print(elastic$bestTune)
coef(elastic$finalModel, elastic$bestTune$lambda)
print(elastic)
pred4 <- elastic%>%predict(test)
squared_err = sum((test[,16] - pred4)^2)
```

Appendix B Design of experiments code

```
library(FrF2)

design <- FrF2(16, 10)
print(design)
```