GTx: ISYE6501x - Homework 5

Muh Alif Ahsanul Islam

June 18, 2019

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
2. Lasso
3. 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.

Answer
1. Stepwise regression

```
rm(list=ls())
rmse = function(ypred, ytrue){
  return (sqrt(mean((ypred - ytrue)^2)))
}
set.seed(0)
library(caTools)
crime_df = read.table('uscrime.txt', header=TRUE, stringsAsFactors=FALSE)
sample = sample.split(crime_df, SplitRatio=0.7)
train = subset(crime_df, sample == TRUE)
test  = subset(crime_df, sample == FALSE)

library(MASS)
print('Backward elimination')
```

```
## [1] "Backward elimination"
```

```
model_back = step(lm(Crime~., data=train), direction='backward', trace=0)
summary(model_back)
```

```
##
## Call:
## lm(formula = Crime ~ M + So + Po1 + M.F + NW + U1 + U2 + Wealth +
##     Ineq, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -367.89 -114.33    8.02   88.60  446.68
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7955.2369  1435.3321  -5.542 1.23e-05 ***
## M             160.3657    48.1185   3.333  0.00289 **
## So           -520.7358   169.8896  -3.065  0.00548 **
## Po1            65.5290    27.1052   2.418  0.02395 *
## M.F            34.7811    16.2905   2.135  0.04363 *
## NW              9.5304     7.2892   1.307  0.20398
## U1          -8805.2578  4732.9237  -1.860  0.07566 .
```

```
## U2           231.8256    96.2338   2.409  0.02440 *
## Wealth         0.2763     0.1208   2.287  0.03169 *
## Ineq          65.6085    25.2854   2.595  0.01620 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 207.7 on 23 degrees of freedom
## Multiple R-squared:  0.8324, Adjusted R-squared:  0.7668
## F-statistic: 12.69 on 9 and 23 DF,  p-value: 5.493e-07
```

```
print('Forward selection')
```

```
## [1] "Forward selection"
```

```
model_forw = step(lm(Crime~., data=train), direction='forward', trace=0)
summary(model_forw)
```

```
##
## Call:
## lm(formula = Crime ~ M + So + Ed + Po1 + Po2 + LF + M.F + Pop +
##     NW + U1 + U2 + Wealth + Ineq + Prob + Time, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -265.20 -101.78  -15.92   99.74  408.51
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.227e+03  2.228e+03  -3.692  0.00181 **
## M            1.661e+02  7.445e+01   2.231  0.03943 *
## So          -3.136e+02  2.885e+02  -1.087  0.29226
## Ed           6.820e+01  9.559e+01   0.713  0.48523
## Po1          8.415e+01  1.478e+02   0.569  0.57655
## Po2         -9.884e+00  1.703e+02  -0.058  0.95439
## LF           1.459e+03  2.718e+03   0.537  0.59835
## M.F          2.276e+01  3.356e+01   0.678  0.50667
## Pop         -9.738e-03  1.769e+00  -0.006  0.99567
## NW           6.194e+00  9.555e+00   0.648  0.52552
## U1          -6.387e+03  6.591e+03  -0.969  0.34614
## U2           2.297e+02  1.109e+02   2.071  0.05389 .
## Wealth       1.932e-01  1.506e-01   1.282  0.21693
## Ineq         5.658e+01  3.257e+01   1.737  0.10044
## Prob        -1.341e+02  4.230e+03  -0.032  0.97509
## Time         4.707e+00  1.037e+01   0.454  0.65555
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 227.5 on 17 degrees of freedom
## Multiple R-squared:  0.8514, Adjusted R-squared:  0.7204
## F-statistic: 6.496 on 15 and 17 DF,  p-value: 0.0002129
```

```
print('Both direction')
```

```
## [1] "Both direction"
```

```
model_both = step(lm(Crime~., data=train), direction='both', trace=0)
summary(model_both)
```

```
##
## Call:
## lm(formula = Crime ~ M + So + Po1 + U2 + Wealth + Ineq + LF,
##     data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -379.56 -101.09  -33.01   89.72  475.80
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7412.8960  1161.7579  -6.381 1.11e-06 ***
## M             212.9117    43.1895   4.930 4.47e-05 ***
## So           -270.0419   154.8388  -1.744   0.0934 .
## Po1            98.6565    20.6709   4.773 6.71e-05 ***
## U2            137.0578    51.0548   2.685   0.0127 *
## Wealth          0.2115     0.1103   1.917   0.0668 .
## Ineq           61.4948    24.6656   2.493   0.0196 *
## LF           3211.5279  1260.6330   2.548   0.0174 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 199.5 on 25 degrees of freedom
## Multiple R-squared:  0.8319, Adjusted R-squared:  0.7849
## F-statistic: 17.68 on 7 and 25 DF,  p-value: 3.14e-08
```

```
back_train_rmse = rmse(model_back[["fitted.values"]], train$Crime)
back_test_rmse = rmse(predict(model_back, test), test$Crime)
forw_train_rmse = sqrt(mean((model_forw[["fitted.values"]] - train$Crime)^2))
forw_test_rmse = sqrt(mean((predict(model_forw, test) - test$Crime)^2))
both_train_rmse = sqrt(mean((model_both[["fitted.values"]] - train$Crime)^2))
both_test_rmse = sqrt(mean((predict(model_both, test) - test$Crime)^2))
summary_df = data.frame(model=c('backward', 'forward', 'both'),
                        train_error = c(back_train_rmse, forw_train_rmse, both_train_rmse),
                        test_error = c(back_test_rmse, forw_test_rmse, both_test_rmse),
                        stringsAsFactors=FALSE)
summary_df
```

```
##      model train_error test_error
## 1 backward    173.4389   380.6348
## 2  forward    163.2833   342.3818
## 3     both    173.6799   371.7698
```

Best model is forward selection model because it has lowest test error. (In this exercise I am using test dataset to do validation)

2. Lasso model

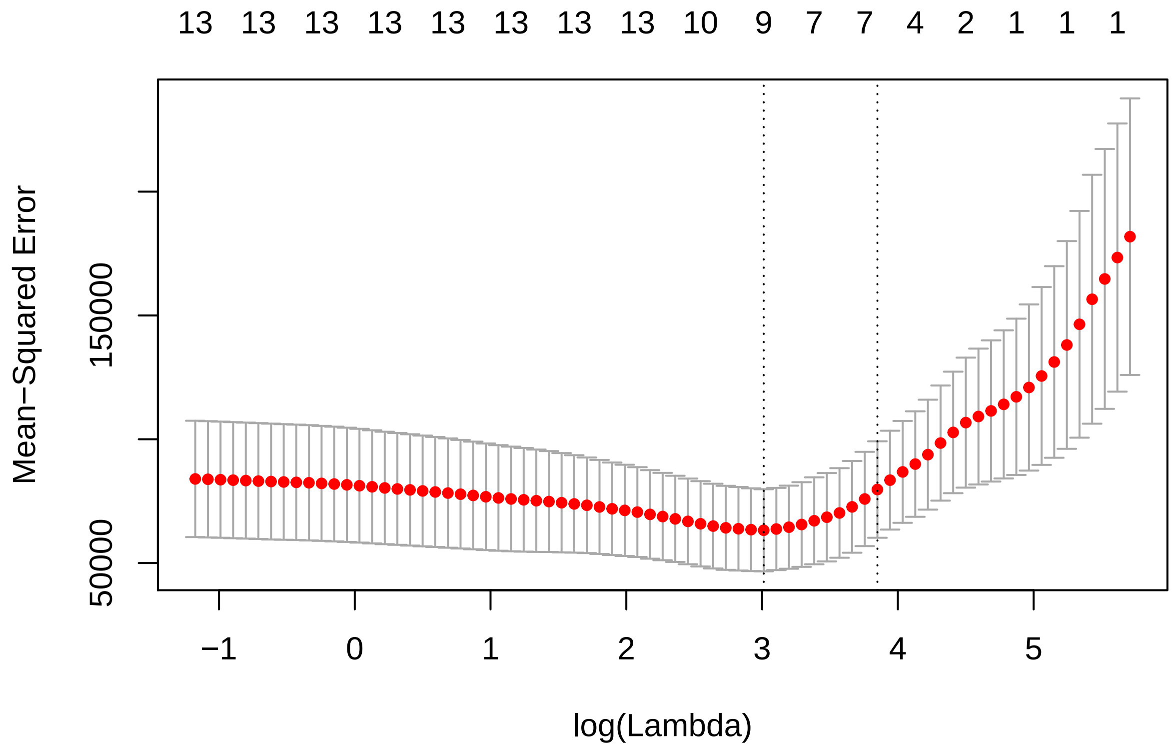
```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-18
```

```r
set.seed(0)
X_train = as.matrix(train[, -16])
y_train = as.matrix(train[, 16])
X_test = as.matrix(test[, -16])
y_test = as.matrix(test[, 16])
lasso_cv = cv.glmnet(x=X_train, y=y_train, type.measure='mse', alpha=1,
                     nfolds=8, family='gaussian', standardize=TRUE)
plot(lasso_cv)
```




```r
model_lasso = glmnet(x=X_train, y=y_train, lambda=lasso_cv$lambda.1se,
                     alpha=1, family='gaussian', standardize=TRUE)
print('attributes used in this model: ')
```

```
## [1] "attributes used in this model: "
```

```r
model_lasso$beta
```

```
## 15 x 1 sparse Matrix of class "dgCMatrix"
##                s0
## M       45.880676
## So       .
## Ed       .
## Po1     82.974338
## Po2      .
## LF     298.269907
## M.F     26.630727
## Pop      .
## NW       .
## U1       .
## U2      15.832260
## Wealth   .
```

```
## Ineq     .
## Prob     .
## Time     5.222667
```

```
lasso_train_rmse = rmse(predict(model_lasso, X_train), y_train)
lasso_test_rmse = rmse(predict(model_lasso, X_test), y_test)
print('Lasso model, train RMSE')
```

```
## [1] "Lasso model, train RMSE"
```

```
print(lasso_train_rmse)
```

```
## [1] 245.1539
```

```
print('Lasso model, test RMSE')
```

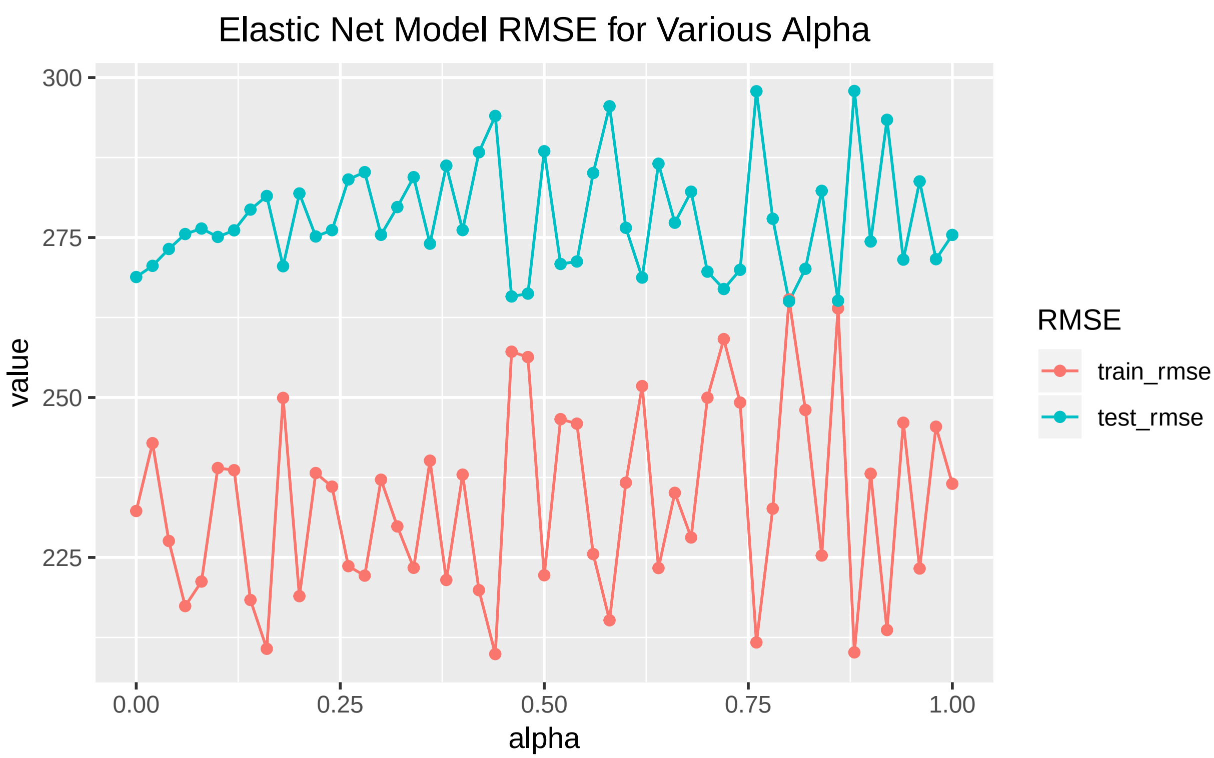
```
## [1] "Lasso model, test RMSE"
```

```
print(lasso_test_rmse)
```

```
## [1] 271.6451
```

3. Elastic net
   For elastic net I will vary alpha from 0 to 1 and find the best model. Note that when alpha=1, the model is lasso and when alpha=0, the model is ridge..

```
alpha_list = seq(0, 1.0, 0.02)
result_df = NULL
set.seed(0)
for (alpha in alpha_list){
  model = cv.glmnet(x=X_train, y=y_train, type.measure='mse',
                    alpha=alpha, nfolds=8, family='gaussian', standardize=TRUE)
  train_rmse = rmse(predict(model, X_train), y_train)
  test_rmse = rmse(predict(model, X_test), y_test)
  result_df = rbind(result_df,
                    data.frame(alpha, train_rmse, test_rmse))
}
library(reshape2)
plot_result_df = melt(result_df, id.vars='alpha', variable.name='RMSE')
library(ggplot2)
theme_update(plot.title = element_text(hjust = 0.5))
ggplot(plot_result_df, aes(alpha,value)) +
geom_line(aes(colour = RMSE)) + geom_point(aes(colour = RMSE)) +
ggtitle("Elastic Net Model RMSE for Various Alpha")
```



Graph above shows that there are no clear trend or relation between RMSE and alpha. Moreover, it is hard to say which model is the best because the RMSE values are very close to each other.

Best model is the one with lowest test RMSE.

```
best_res = result_df[which.min(result_df$test_rmse),]
best_res
```

```
##    alpha train_rmse test_rmse
## 41   0.8   265.3355  265.0306
```

Compare the error from Question 11.1.1, 11.1.2, 11.1.3

```
temp_df = data.frame('lasso', lasso_train_rmse, lasso_test_rmse)
names(temp_df) = names(summary_df)
summary_df = rbind(summary_df, temp_df)

temp_df = data.frame('elastic net', best_res$train_rmse, best_res$test_rmse)
names(temp_df) = names(summary_df)
summary_df = rbind(summary_df, temp_df)

summary_df
```

```
##         model train_error test_error
## 1    backward    173.4389   380.6348
## 2     forward    163.2833   342.3818
## 3        both    173.6799   371.7698
## 4       lasso    245.1539   271.6451
## 5 elastic net    265.3355   265.0306
```

From table above we can see that Elastic net model is the best model with test RMSE 240.

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.

Problem: Suppose I want to manufacture and then sell cars with price as high as possible. So I try to gauge customer's willingness to pay by using market testing.
I have many combinations of feature of the car:
1. Engine: Internal combustion engine, hybrid engine
2. Transmission: manual, automatic
3. Body color: red, blue
Because I have 2ˆ3=8 combinations, it will be very costly for me to create all 8 prototypes. I will use design experiment so that I can create only 4 combinations of car and still get good amount information of the market condition.

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.

```
library(FrF2)
FrF2(nruns=16, nfactors=10, default.levels=c('no', 'yes'))
```

```
##      A   B   C   D   E   F   G   H   J   K
## 1   no  no  no  no yes yes yes yes  no yes
## 2   no  no yes yes yes  no  no  no  no yes
## 3  yes  no yes yes  no yes  no yes  no  no
## 4  yes yes  no yes yes  no  no yes  no  no
## 5   no  no yes  no yes  no  no yes yes  no
## 6   no yes  no yes  no yes  no  no  no yes
## 7  yes yes  no  no yes  no  no  no yes yes
## 8   no  no  no yes yes yes yes  no yes  no
## 9   no yes yes yes  no  no yes  no yes  no
## 10 yes yes yes  no yes yes yes  no  no  no
## 11  no yes yes  no  no  no yes yes  no yes
## 12 yes yes yes yes yes yes yes yes yes yes
## 13  no yes  no  no  no yes  no yes yes  no
## 14 yes  no yes  no  no yes  no  no yes yes
## 15 yes  no  no yes  no  no yes yes yes yes
## 16 yes  no  no  no  no  no yes  no  no  no
## class=design, type= FrF2
```

The table above shows 16 fictitious houses for survey. Row indicates house number, and column indicates features of the house, 'yes' means the feature is included in the house and 'no' means the feature is not included in the house.

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).

Binomial distribution: probability of number of students that passed the midterm
Geometric distribution: probability of number of drug research needs to be done until the drug is successfully manufactured
Poisson distribution: probability of a shop sells n items in a day (the shop usually sells x items/day)
Exponential distribution: number of hits a website receives in an hour
Weibull distribution: lifetimes of medical and dental implants

---