ISYE 6501 Intro Analytics Modeling – HW5

Question 8.1 Describe a situation or problem from your job, everyday life, current events, etc., for which a linear regression model would be appropriate. List some (up to 5) predictors that you might use.

Linear Regression can be used by credit card companies to predict the credit card balance of a cardholder and measure the risk of overdue payment. Predictors I might use includes:

- Income – Monthly income in $1,000
- History Balance - Average monthly credit card balance in $1,000
- Rating - Credit ratings
- Age - Age in years
- Education – Total number of years of education after high school

Question 8.2 Using crime data (file `uscrime.txt`), use regression (a useful R function is `lm` or `glm`) to predict the observed crime rate in a city with the following data:

M = 14.0	*LF = 0.640*	*U2 = 3.6*
So = 0	*M.F = 94.0*	*Wealth = 3200*
Ed = 10.0	*Pop = 150*	*Ineq = 20.1*
Po1 = 12.0	*NW = 1.1*	*Prob = 0.04*
Po2 = 15.5	*U1 = 0.120*	*Time = 39.0*

Show your model (factors used and their coefficients), the software output, and the quality of fit.

The first step is to read in the dataset and do some simple data summary. From the code, I get to know this dataset contains only **47 data points** and **16 columns** (15 predictors and 1 independent variable). Before building the linear regression model, I make scatter plots of each predictor and the independent variable and linear regression line, R and P-value. From the graph, we can see Po1 and Po2 have the greatest linear relationship with Crime. (Fig.1)

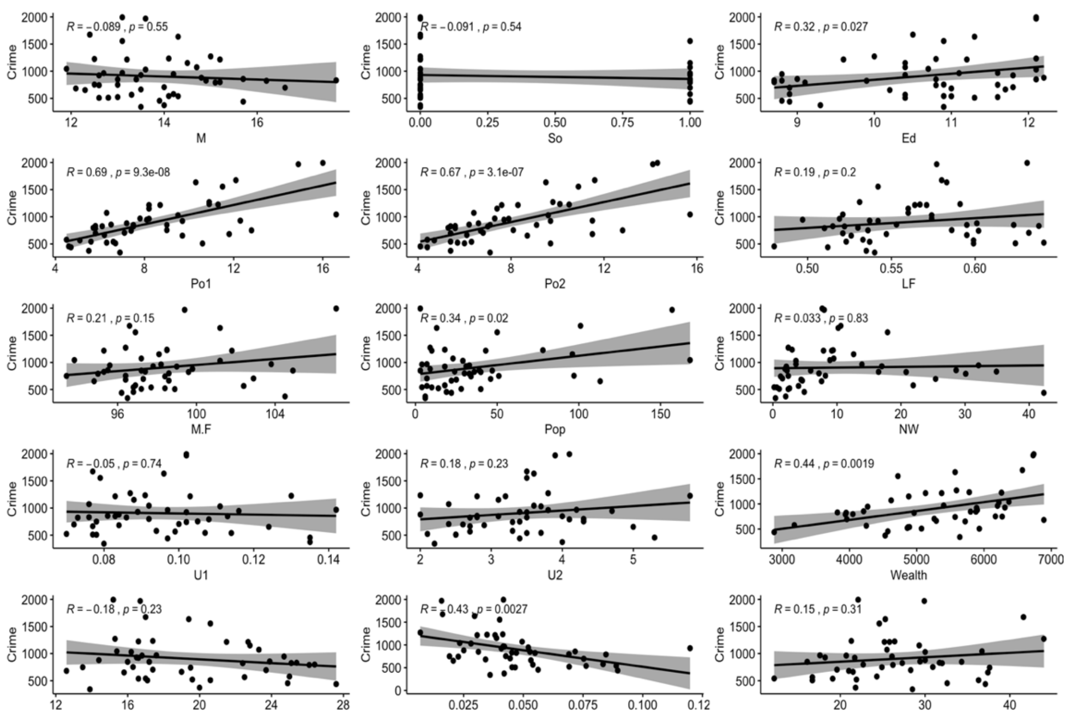

Fig.1

Then I randomly split the data into train set (70%) and test set (30%), and train with Gaussian linear regression, I got Sum of squared error: 554,100 MSE:17,874 for the train set and MSE: 94,588 for the test set. Which is overfitted. Here are the details of the model:

```
Call:  glm(formula = Crime ~ ., family = gaussian, data = train)
Coefficients:
(Intercept)          M          So          Ed         Po1         Po2          LF         M.F
-2.805e+03  8.928e+01  -1.269e+02  1.592e+02  4.750e+02  -4.213e+02  -1.379e+03  8.092e+00
       Pop         NW          U1          U2      Wealth        Ineq        Prob        Time
-5.201e-01  9.869e+00  -6.786e+03  1.457e+02  4.486e-02  5.695e+01  -9.404e+03  -2.580e+01

Degrees of Freedom:  30 Total (i.e. Null);  15 Residual
Null Deviance:       5300000
Residual Deviance:   554100
AIC:                 425.5

95% CI for the coefficients:
                     2.5 %        97.5 %
(Intercept) -6.898835e+03  1288.8598693
M           -1.425656e+01   192.8192314
So          -5.816769e+02   327.9565632
Ed          -2.335514e+00   320.7255817
Po1          1.951647e+02   754.7599161
Po2         -7.352345e+02  -107.3267389
LF          -4.527370e+03  1769.2957841
M.F         -4.239436e+01    58.5788185
Pop         -4.126814e+00     3.0866611
NW          -7.764129e+00    27.5018231
U1          -1.860935e+04  5037.5974299
U2          -3.807691e+01   329.4589436
Wealth      -3.084256e-01     0.3981375
Ineq         6.817819e-01   113.2235481
Prob        -1.658871e+04 -2218.6717731
Time        -4.871470e+01    -2.8896907
```

To get a better model, I tried to using LASSO regression with cross-validation. For the data set with small data points and a comparative big predictor sets like this. LASSO regularization can result in sparse models with few coefficients by adding a penalty equal to the absolute value of the magnitude of coefficients. Larger penalties result in coefficient values closer to zero, which is ideal for reducing noises and producing simpler models.

We get MSE form 61,185 to 75,525 from the 10-fold cross-validation. Which is much smaller than the previous test error. From the coefficients, the model only used 5 variables: M, Po1, M.F, Ineq, Prob. As we can see, Po2 is not selected in this model, which has a very high correlation with Crime from the single variable regression. The reason is that there is collinearity between Po1 and Po2, the R-value between these two variable is 0.99.(Fig.2) This indicates that LASSO can also deal with multicollinearity to reduce model noise.

For this model, we have **Predicted Crime for the new data: 1121.35** (with lambda.min) **and 1180.67** (with lambda.lse)

```
Call:  cv.glmnet(x = as.matrix(df[, c(1:15)]), y = df$Crime, weights = NULL,      offset = NULL, lambda = NULL, type.measure = c("default",          "mse", "deviance", "class", "auc", "mae", "C"), nfolds = 10, foldid = NULL, alignment = c("lambda", "fraction"), grouped = TRUE,      keep = FALSE, parallel = FALSE, gamma = c(0, 0.25, 0.5, 0.75,          1), relax = FALSE, trace.it = 0, family = "gaussian")

Measure: Mean-Squared Error
        Lambda Measure    SE Nonzero
    min   9.24   61185 15958      11
    1se  37.29   75525 20340       5
```

Coefficients- min:

```
        (Intercept) -5.000723e+03
        M            7.108443e+01
        So           4.447322e+01
        Ed           1.234084e+02
        Po1          1.027122e+02
        Po2          .
        LF           .
        M.F          1.871453e+01
        Pop          .
        NW           6.010913e-01
        U1          -2.014423e+03
        U2           8.512460e+01
        Wealth       4.980586e-03
        Ineq         4.809861e+01
        Prob        -3.675090e+03
        Time         .
```

Coefficients-1se:

```
        (Intercept) -2021.92254
        M              30.35597
        So              .
        Ed              .
        Po1            89.51016
        Po2             .
        LF              .
        M.F            15.49501
        Pop             .
        NW              .
        U1              .
        U2              .
        Wealth          .
        Ineq           16.04731
        Prob        -1889.84852
        Time            .
```

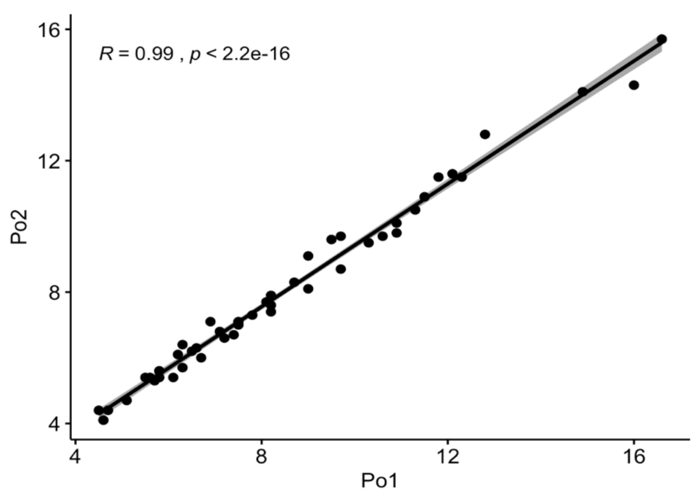

Fig.2

```r
# Code

# ISYE 6501 Intro Analytics Modeling - HW5
# IP uscrime.txt

# Loading and examining data
df<-read.delim("uscrime.txt", header = TRUE, sep = "\t")
#### Display Head Lines ####
head(df,2)
#### Show summary, number of row of last column##
summary(df$Crime)
nrow(df)
ncol(df)
for (i in attributes(df)$names ){
  print(paste0('Name:',i,'   Class:',class(df[[i]]),'   nlevels:',length(uniq
ue(df[[i]]))))
  print(summary(df[[i]]))
}

library("ggplot2")
library("ggpubr")

for (i in c(1:15)){
  assign(paste0("g", i),
         ggscatter(df, x = colnames(df)[i], y = "Crime",
         add = "reg.line", conf.int = TRUE,
         cor.coef = TRUE, cor.method = "pearson",
         xlab = colnames(df)[i], ylab = "Crime")
          )
}

library(gridExtra)
grid.arrange(g1, g2, g3, g4, g5, g6, g7, g8, g9, g10, g11, g12, g13, g14, g15
,  nrow = 5)

#splite train and validation set
set.seed(666)
g <- sample(1:2,size=nrow(df),replace=TRUE,prob=c(0.7,0.3))
train <- df[g==1,]
test <- df[g==2,]

# Fit glm model: gaussian model
glm_model1<-glm(Crime~.,family = gaussian,train)
glm_model1
MSE_train<-mean(glm_model1$residuals^2) #MSE Train
confint(glm_model1) # 95% CI for the coefficients

p1_test<-predict(glm_model1,test,type="response")
p1_residials<-p1_test-test$Crime
```

```r
MSE_test<-mean(p1_residials^2) #MSE Train


# Fit glm model: gaussian model leave one out cross validation
library(glmnet)
lasso_glm<-cv.glmnet(as.matrix(df[,c(1:15)]), df$Crime, family = "gaussian",
                           weights = NULL, offset = NULL, lambda = NULL,
                           type.measure = c("default", "mse", "deviance", "cl
ass", "auc", "mae","C"),
                           nfolds = 10, foldid = NULL, alignment = c("lambda"
,  "fraction"),
                           grouped = TRUE, keep = FALSE, parallel = FALSE,
                           gamma = c(0, 0.25, 0.5, 0.75, 1), relax = FALSE, t
race.it = 0)
lasso_glm
plot(lasso_glm)
coef(lasso_glm, s = "lambda.min")
coef(lasso_glm, s = "lambda.1se")

       ggscatter(df, x = "Po1", y = "Po2",
                 add = "reg.line", conf.int = TRUE,
                 cor.coef = TRUE, cor.method = "pearson",
                 xlab = "Po1", ylab = "Po2")


new_data<- matrix(1:15, nrow = 1, dimnames = list(1, colnames(df)[c(1:15)]))
new_data[1,]<-c(14.0, 0, 10.0, 12.0, 15.5, 0.640, 94.0, 150, 1.1, 0.120, 3.6,
3200, 20.1, 0.04, 39.0)

predict(lasso_glm,new_data, s = "lambda.min")
predict(lasso_glm,new_data, s = "lambda.1se")
```