Modeling for Boston Crime Analysis

Dan Hua Li

2022-10-17

Preamble: The dataset about crime rate document the details surrounding Boston, and capturing the type of factors may affect crime rate. This project focused on building up the models for better prediction of the frequency a crime happen on a certain condition.

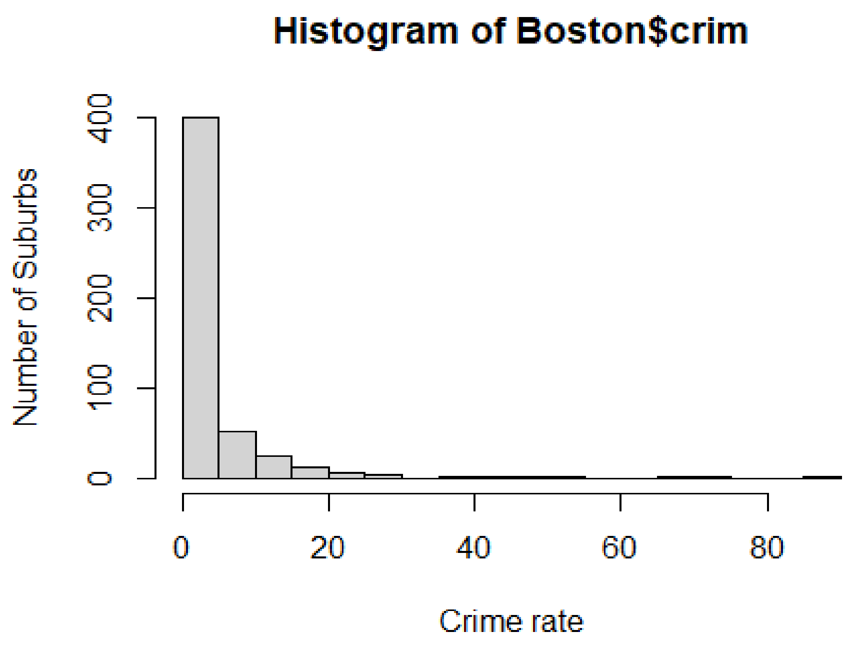
There are 506 neighborhoods of Boston, 12 variables and 1 dependent variables for Y.

```
library(ISLR2)
attach(Boston)
#head(Boston)
Boston <- na.omit(Boston)
dim(Boston)

## [1] 506  13
```

Section 1: Apply pairwise scatterplots, boxplots, and histograms to describe the Info.

1(a) Using histogram for crime rate and Property-tax rate

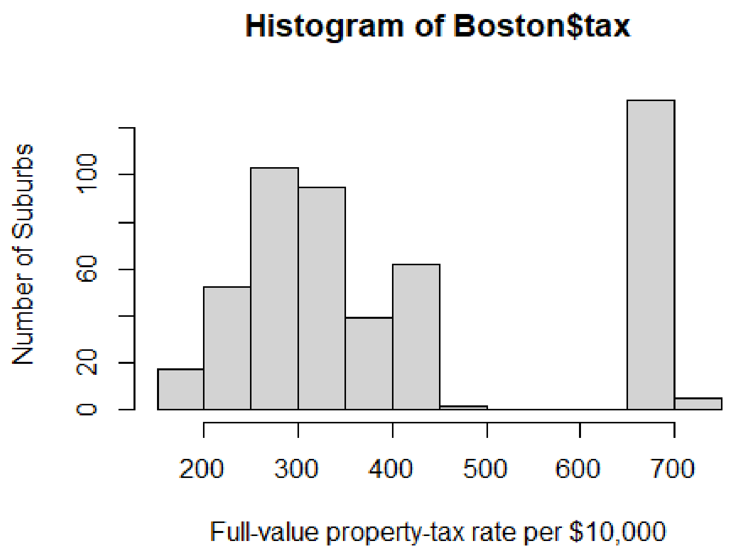
```
hist(Boston$crim, breaks = 20, xlab = "Crime rate", ylab="Number of Suburbs"
)
```



The per capita crime rate by town. it appears to have many outliers as seen in the boxplot above. The majority of towns have close to zero crime rate (of under 5), while some suburbs have a rate as high as 70 -80 (approximate 15%), There are many outlier suburbs marked as outliers indicating most suburbs have a very low crime rate, although there are a significant number of towns in the minority with an outlying crime rate on the higher end (eg. more than 50 towns have crim > 10).

Now histogram of Full value property tax rate per 10,000

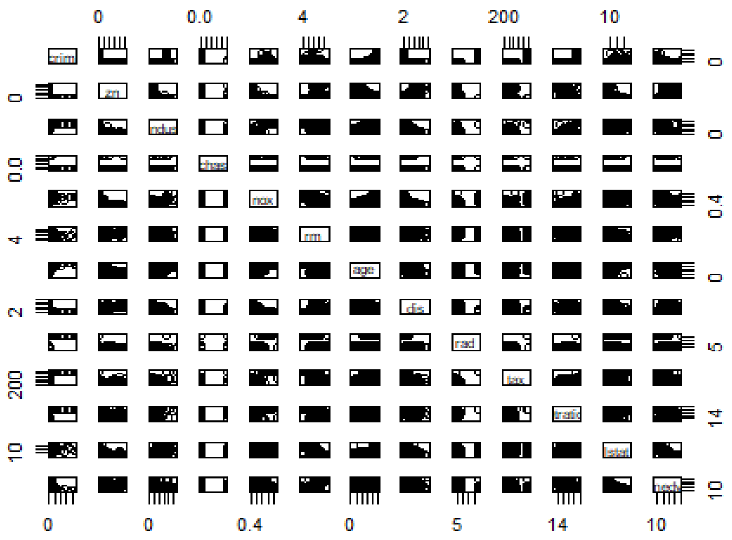
```
hist(Boston$tax, xlab = "Full-value property-tax rate per $10,000",
ylab="Number of Suburbs")
```



The full-value property-tax rate per $10,000. The property tax ranges from approximately 200 to 700, with no extreme outliers. The median lies around 330 dollars.

1.a. Using pairwise scatterplots to find the correlation of variables,

```
pairs(Boston)
```



Using all features in a pair plot (as seen above) to see correlation among variables. But, it is difficult to figure out much from above figure due to the amount of data on the page(pictures is too small) are not clear, however some pairs seem highly correlated..

1) rad: index of accessibility to radial highways. There appears to be a positive correlation to crime rate

2) medv: Median value of owner occupied homes in %1000 appears to be a negative relation to crime rate. In all cases, many areas have low crime (per capita crime rate In all cases, many areas have low crime (per capita crime rate <20), regardless of the accessibility to highways, tax rate, chas, or lower status of the population.

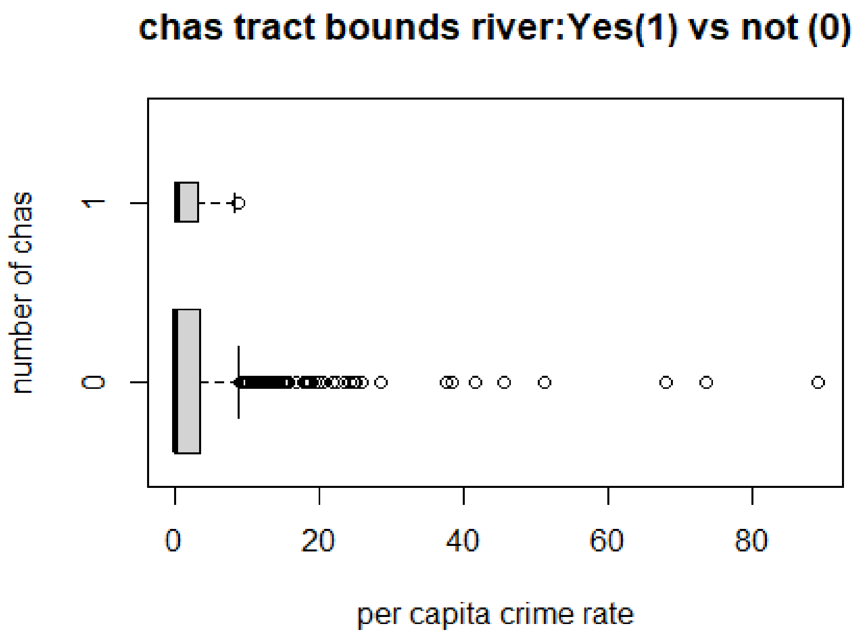
Let's fit a model for all variables to find their coefficients and p-value, and confirm above hypothesis.

```
lm.fit <- lm(crim ~ ., data= Boston)
summary(lm.fit)$coef

##                  Estimate  Std. Error    t value     Pr(>|t|)
## (Intercept) 13.7783937863 7.081825783  1.9455991 5.227089e-02
## zn           0.0457100386 0.018790325  2.4326370 1.534403e-02
## indus       -0.0583501107 0.083635091 -0.6976750 4.857094e-01
## chas        -0.8253775522 1.183396256 -0.6974651 4.858406e-01
## nox         -9.9575865471 5.289824241 -1.8824040 6.036986e-02
## rm           0.6289106622 0.607092391  1.0359390 3.007385e-01
## age         -0.0008482791 0.017948208 -0.0472626 9.623231e-01
## dis         -1.0122467382 0.282467566 -3.5835857 3.725942e-04
## rad          0.6124653115 0.087535764  6.9967438 8.588123e-12
## tax         -0.0037756465 0.005172346 -0.7299678 4.657565e-01
## ptratio     -0.3040727572 0.186359810 -1.6316434 1.033932e-01
## lstat        0.1388005968 0.075721250  1.8330468 6.739844e-02
## medv        -0.2200563590 0.059823956 -3.6783987 2.605302e-04
```

Above hypothesis can almost be accepted. because rad have a positive coefficient, while medv have a negative coefficient and also they are significant in finding the response variable(crim) due to p-value less than .05. In addition to these, dis and zn are significant variables with zn has positive correlation and dis has negative correlation with crim variable and However nox,tax, age, ptratio,rm, indus,lstat and chas have higher p-value which is greater than .05. They are not significantly affect the crime rate and hence they are insignificant in determining the crime rate.

1 (b) Boxplot for chas (Charles River dummy) with crime rates.

```
boxplot(crim ~ chas,data = Boston, main="chas tract bounds river:Yes(1) vs no
t (0)", xlab="per capita crime rate", ylab="number of chas",varwidth=T, horiz
ontal=T)
```

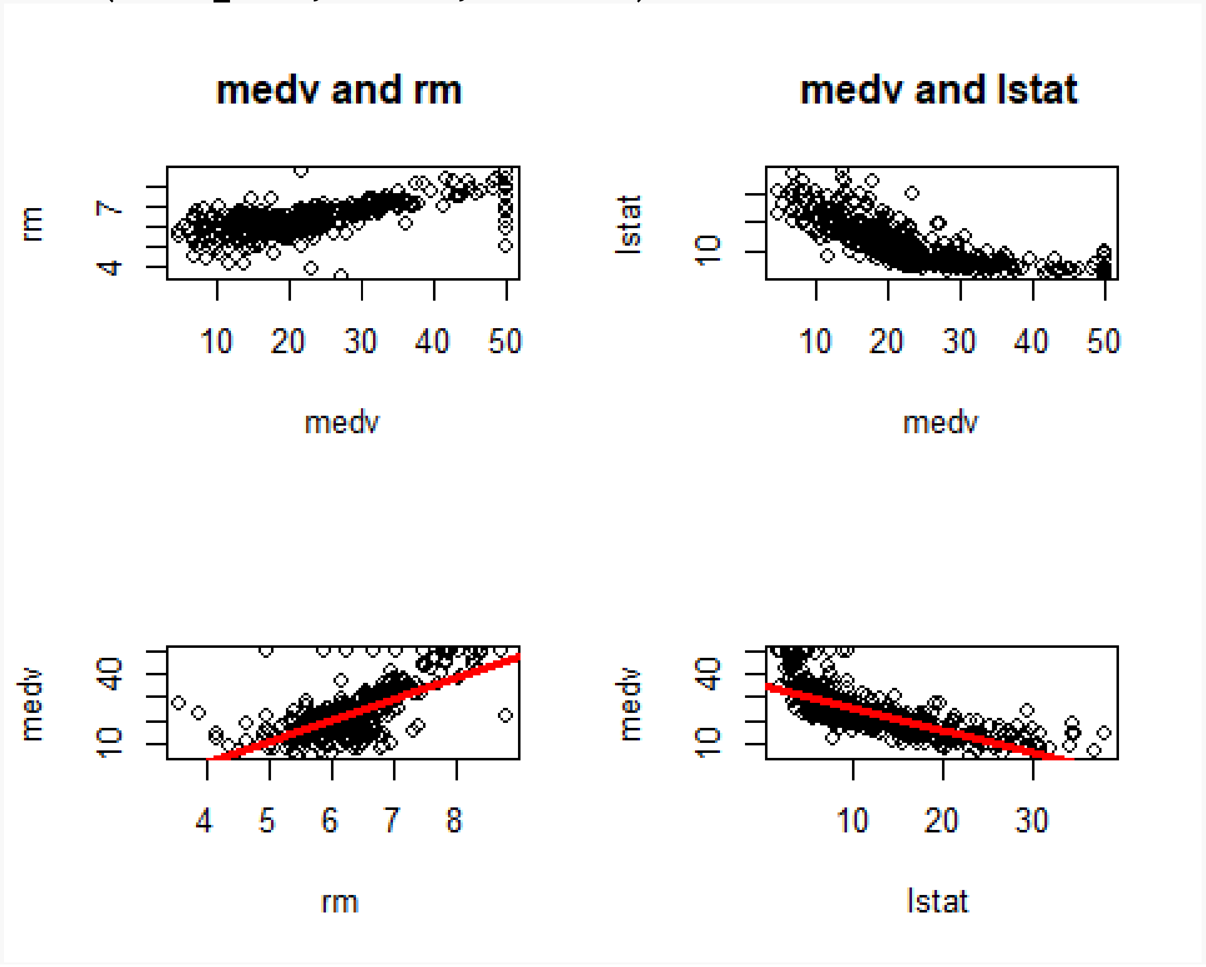


Charles River dummy variable: if tract bounds river(group 0): The box plot is comparatively more tall. This suggests group-0 has much more observations (471/506) and hold quite difference about crime rate. and have many outliers as seen in the boxplot above where crime rate within a wide range. The majority (mean) have close to zero crime rate (as approximately same as group-1) Charles River dummy variable: if it tract bounds river(group 1): the box plot is comparatively short: This suggests group-1 has samll observations (35 among 506) and much less outliers. Both have similar interval with 95% confidence. it appear chas variable is very slightly related to crime rate.

1(c) From the scatter plots find the correlation among the variables

```
par(mfrow = c(2, 2))
plot(medv, rm, main="medv and rm")
plot(medv,lstat,main="medv and lstat")
lm.fit_MandR <-lm(medv ~ rm, data=Boston)
plot(rm,medv)
abline(lm.fit_MandR, lwd = 3, col="red")
lm.fit_MandL <-lm(medv ~ lstat, data=Boston)
plot(lstat,medv)
```

```
abline(lm.fit_MandL, lwd = 3, col="red")
```



We use fit model to obtain the red line which can confirm the correlation. The high correlations between medv & lstat (lower status of the population) and also between medv & rm (average number of rooms per dwelling). lstat is negatively correlated to medv, whereas rm is positively correlated which would be expected as the higher value areas would typically have larger dwellings.

Section 2. validation set method of cross validation.

2 a. Randomly split the data into two groups, 80% for training and 20% for testing.

```
set.seed(1)
train <- sample (506,405)
lm.fitvalid <- lm(crim ~ rad+ zn+ nox+ dis+ ptratio+ medv, data=Boston, subse
t=train)
mean((crim - predict(lm.fitvalid, Boston))[train]^2)

## [1] 35.35255

mean((crim - predict(lm.fitvalid, Boston))[-train]^2)

## [1] 65.35408
```

```
#also use validation set method to identify the degree of a polynomial functi
on.
lm.fit<-lm(crim ~ rad, data=Boston, subset=train)
mean((crim - predict(lm.fit, Boston))[-train]^2)

## [1] 70.95459

lm.fit2 <- lm(crim ~ poly(rad, 2), data= Boston, subset = train)
mean((crim - predict(lm.fit2, Boston))[-train]^2)

## [1] 70.1167

lm.fit3 <- lm(crim ~ poly(rad,3), data= Boston, subset = train)
mean((crim - predict(lm.fit3, Boston))[-train]^2)

## [1] 70.06135

lm.fit4 <- lm(crim ~ poly(rad,4), data= Boston, subset = train)
mean((crim - predict(lm.fit4, Boston))[-train]^2)

## [1] 70.05628

lm.fit5 <- lm(crim ~ poly(rad,5), data= Boston, subset = train)
mean((crim - predict(lm.fit5, Boston))[-train]^2)

## [1] 70.04696

lm.fit6 <- lm(crim ~ poly(rad,6), data= Boston, subset = train)
mean((crim - predict(lm.fit6, Boston))[-train]^2)

## [1] 70.04215
```

It show validation set method has stepwise improve the model in "rad" slightly. the algorithm.d>=6 (rad ^ 6) has smallest MSE in test data.

Section 3. Fit a multiple regression model to the training set to predict per capita crime rate using all of the predictors.

```
lm.fit_full <- lm(crim ~ ., data= Boston, subset=train)
summary(lm.fit_full)

##
## Call:
## lm(formula = crim ~ ., data = Boston, subset = train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.902 -1.991 -0.356  0.978 59.628
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 14.184434   7.384016   1.921  0.05546 .
## zn           0.037243   0.019211   1.939  0.05326 .
## indus       -0.063245   0.084548  -0.748  0.45489
## chas        -0.682564   1.200490  -0.569  0.56997    
## nox         -8.412184   5.437408  -1.547  0.12265    
## rm           0.275890   0.637044   0.433  0.66520    
## age          0.001654   0.018342   0.090  0.92819    
## dis         -0.840973   0.295560  -2.845  0.00467 ** 
## rad          0.590638   0.086858   6.800 3.92e-11 ***
## tax         -0.003590   0.005134  -0.699  0.48483    
## ptratio     -0.336444   0.193509  -1.739  0.08288 .  
## lstat        0.140019   0.077189   1.814  0.07045 .  
## medv        -0.180749   0.060132  -3.006  0.00282 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 5.999 on 392 degrees of freedom
## Multiple R-squared:  0.4602, Adjusted R-squared:  0.4437 
## F-statistic: 27.85 on 12 and 392 DF,  p-value: < 2.2e-16
```

3 (a). Describe your model (i.e., meaning of some β values. Draw a residual plot.)

The column “Estimate” is about the coefficient: beta. The first number (intercept) is beta 0, the others orderly : beta-1 for variable “zn”, beta-2 for “indus” …..beta_12 for “medv”. The formula is: (It use linear regression then it should be Least squares line)

hat-y (crim)= beta-0 + beta_1 * zn + beta_2 *indus + …beta_12* medv + E

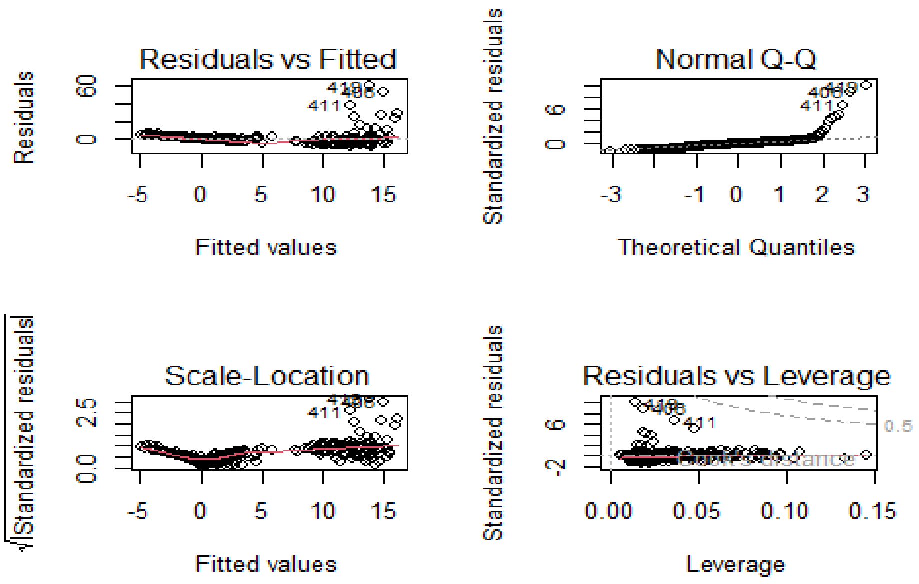
hat-y = 11.249401 + 0.041369 * zn - 0.089904 * indus + … -0.206845* medv +E

Here, the beta_4 for nox is -8, but its p-value reach 0.1 > 0.05, which May strong incorrectly impact the model and should be removed

the information includes the model of multiple linear regession’s RSE (high), R-squared =0.4602 (far from 1, not perfect), F-test result and P-value.

Residual Plot: The residuals exhibit a simlar U-shape, which provides a strong indication of non-linearity in the data.

```
par(mfrow = c(2, 2))
plot(lm.fit_full)
```



```
#plot(predict(lm.fit_full), residuals(lm.fit_full))
```

For this multiple predictors, we plot the residuals versus the predicted (or fitted) values ˆyi. the residual plot shows a little fitted discernible pattern. The presence of a pattern may indicate a problem with some aspect of the linear model.

The red line is not a smooth fit to the residuals, which is displayed in order to make it difficult to identify the trends. The residuals exhibit a similar U-shape, which may provides a indication of non-linearity in the data.

it may need a simple approach which use non-linear transformations of the predictors, such as log X, sqrt X, and X^2, in the regression model.

3 (b). performance on training set: training MSE is 81972.14 (show bellow) Multiple squared-R: 0.4602 (Far from 1,show in section 2)

```
mean((crim-predict(lm.fit_full,Boston))[train]^2)

## [1] 34.83031
```

3 (c). For which predictors can we reject the null hypothesis $H0$: $\beta j = 0$?

predictors: rad, dis, medv…(whose p-value <0.05) can be better use to reject the null hypothesis.

```
summary(lm.fit_full)$coef[ ,4]

##  (Intercept)           zn        indus         chas          nox
 rm
## 5.546116e-02 5.325866e-02 4.548868e-01 5.699730e-01 1.226474e-01 6.651961e
-01
##          age          dis          rad          tax      ptratio         ls
tat
## 9.281900e-01 4.668997e-03 3.916017e-11 4.848332e-01 8.288283e-02 7.044658e
-02
```

```
##         medv
## 2.818861e-03
```

3 (c). The prediction performance on the test set (show bellow) test MSE is: 64.90121 (show bellow)

```
#find the MSE in test data
mean((crim-predict(lm.fit_full, Boston))[-train]^2)

## [1] 64.90121

library(leaps)
library(pls)

##
## Attaching package: 'pls'

## The following object is masked from 'package:stats':
##
##     loadings
```

Section 4. Best subset selection and Lasso

Fit two models (best subset selection and the lasso) to the training set to predict per capita crime rate.

Best subset selection: Model_1

```
library(leaps)
set.seed(1)
train <-sample (506,405)

regfit.full <-regsubsets(crim ~., data = Boston, subset=train, nvmax=12)
reg.summary <- summary(regfit.full)
names(reg.summary)

## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

To find the optimal model,now we need to identify the location of the maximum point of a vector,using this point we can indicate a model with the largest adjusted R2 statistic and the minimum Cp, BIC as the red dots. Similarly which.min( ) is used to find a pt representing model with the minimum Cp, BIC.

adjust R^2 : 8 Cp: 7 BIC: 2 (show bellow)

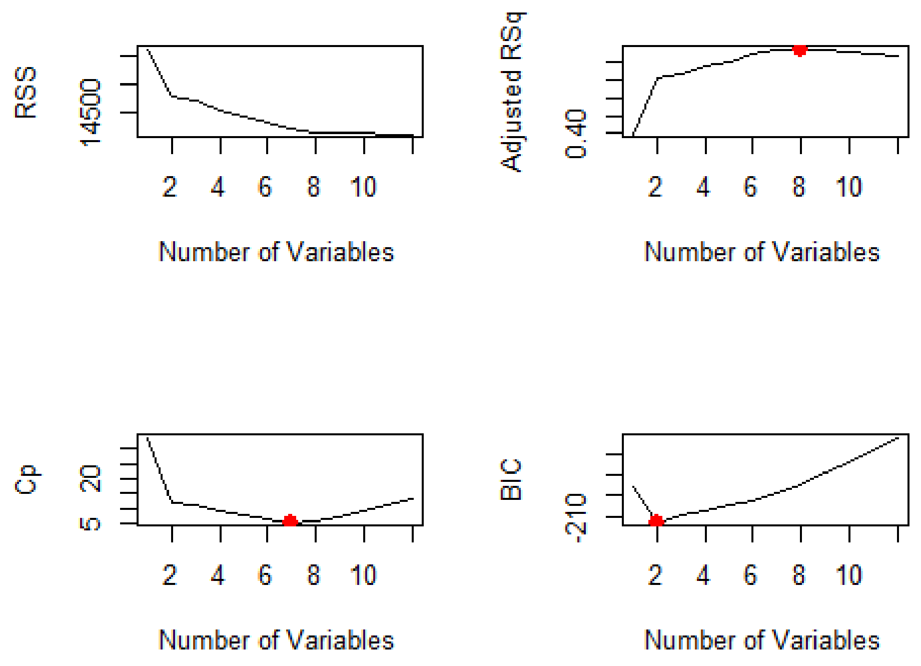
```
which.max(reg.summary$adjr2)

## [1] 8

which.min(reg.summary$cp)

## [1] 7

which.min(reg.summary$bic)
```

```
## [1] 2
```

Combine the result. Select the model size: 2 ?,   7 ? ,   8 ?

We can figure out the best model size by by adjust R^2, Cp, and BIC or viewing their plots . Here, the best subset selection already chose the model whose RSS and R^2 are best. View its RSS and find its tain MSE = reg.summary$rss/nrow(x_train)).

```
reg.summary$rss

##  [1] 15643.51 14780.64 14695.64 14539.77 14427.32 14297.08 14198.79 14141.
76
##  [9] 14126.58 14114.00 14106.57 14106.28

reg.summary$cp

##  [1] 33.718235 11.739864 11.377790  9.046293  7.921359  6.302330  5.570768
##  [8]  5.986138  7.564140  9.214486 11.008132 13.000000

reg.summary$bic

##  [1] -195.8309 -212.8059 -209.1378 -207.4525 -204.5931 -202.2617 -199.0519
##  [8] -194.6778 -189.1090 -183.4660 -177.6753 -171.6798

reg.summary$adjr2

##  [1] 0.3999267 0.4316155 0.4334748 0.4380825 0.4410310 0.4446849 0.4471137
##  [8] 0.4479435 0.4471402 0.4462307 0.4451137 0.4437097

par(mfrow = c(2, 2))
plot(reg.summary$rss, xlab = "Number of Variables",
    ylab = "RSS", type = "l")
plot(reg.summary$adjr2, xlab = "Number of Variables",
    ylab = "Adjusted RSq", type = "l")
points(8, reg.summary$adjr2[8], col = "red", cex = 2,
    pch = 20)
plot(reg.summary$cp, xlab = "Number of Variables",
    ylab = "Cp", type = "l")
points(7, reg.summary$cp[7], col = "red", cex = 2,
    pch = 20)
plot(reg.summary$bic, xlab = "Number of Variables",
    ylab = "BIC", type = "l")
points(2, reg.summary$bic[2], col = "red", cex = 2,
    pch = 20)
```



```
#tain MSE = reg.summary$rss/nrow(x_train)) where the 6th RSS
train_MSE <- 14297.08/405
train_MSE

## [1] 35.30143
```

Based on above results, I would choose model #6 because among models#2, #7, and #8, the Cp, BIC and Adjusted are not too far apart. Here is the code to obtain the coefficients:

```
#find the coefficients for this methods
coef(regfit.full,6)

## (Intercept)          zn       indus         dis         rad       lstat 
##  3.77366111  0.04370455 -0.13876552 -0.71116602  0.49457639  0.13734937 
##        medv 
## -0.13247257
```

That is y = 3.773 +0.0437*zn - 0.139* indus - 0.711*dis + 0.495* rad +0.137*lstat -0.132*medv

Use the validation set of cross-validation approaches to determine which model of a given size is best.(using *only the training observations*) for comparing.

```
set.seed(1)
train <- sample(c(TRUE, FALSE), nrow(Boston), replace = TRUE)
test <- (!train)
```

Apply `regsubsets()` to the training set in order to perform best subset selection. compute the validation set error for the best model of each model size. First make a matrix from the test data.Run a loop for each size `i`, we extract the coefficients from `regfit.best` for the best model of that size, multiply them into the appropriate columns of the test model matrix to form the predictions, and compute the test MSE and train MSE.

```
regfit.best <- regsubsets(crim ~ ., data = Boston[train, ], nvmax = 12)
test.mat <- model.matrix(crim ~ ., data = Boston[test, ])

val.errors <- rep (NA,12)
for (i in 1:12) {
  coefi <- coef(regfit.best, id=i)
  pred <- test.mat [, names(coefi)] %*% coefi
  val.errors[i] <- mean ((crim[test]-pred^2))
}
#test MSE: the one is the value of 6th.
val.errors

##  [1] -30.98041 -34.18930 -34.68316 -35.04712 -35.59389 -36.02203 -34.60602
##  [8] -35.27404 -35.23611 -35.20437 -35.19477 -35.24378
```

Obviously, the #6 model is the best with smallest test MSE.36.02203

Now use the full data set obtain more accurate coefficient estimates and select the best 6-variable model

```
which.min(val.errors)

## [1] 6

# Now use the full data set obtain more accurate coefficient estimates and se
lect the best 6-variable model.
regfit.best <-regsubsets(crim ~ ., data= Boston, nvmax = 12)
coef(regfit.best, 6)

##  (Intercept)           zn          nox          dis          rad       ptra
tio
##  20.57369642   0.04666807 -11.85795204  -1.04623129   0.57113805  -0.36940
158
##         medv
##  -0.24759359
```

Model_best: Before cross-validation. best subset selection model with 6 variables is: y = 3.773 +0.0437*zn - 0.139* indus - 0.711*dis + 0.495* rad +0.137*lstat -0.132*medv

```
  After cross-validation. model_best is:
```

Y(crim)=20.574 +0.047*zn -11.858*nox -1.046*dis +0.571*rad -0.369*ptratio -0.248*medv

```
train MSE: 35.30143  (it shows different results when run the code)
```

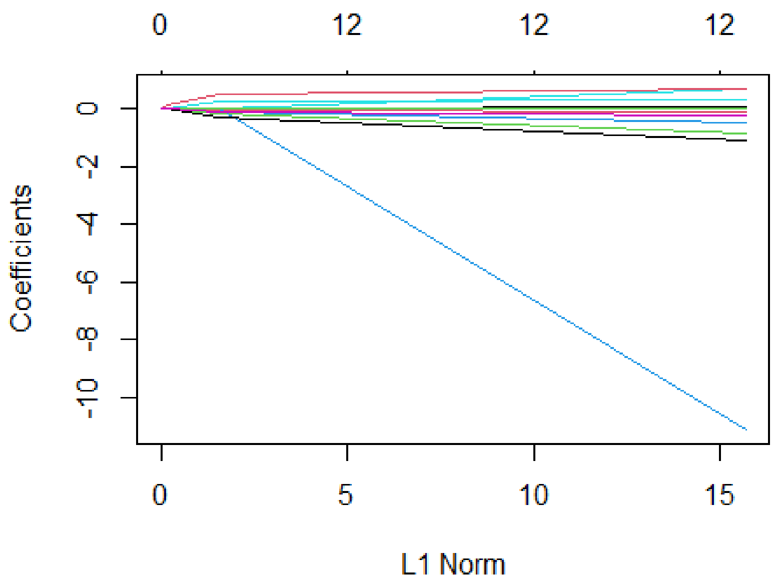
After cross-validation, Model_best test MSE= train MSE : 36.022031 (train on full data)

Their result in coefficients are different but the 6-variable are similar.

Lasso (model_lasso)

Now , fit a lasso model to the simulated data, again using 12 predictors. Use cross-validation to select the optimal value of lambda. here we chosen to implement the function over a grid of values ranging from λ=10^10 to λ=10e−2. Associated with each value of λ is a vector of lasso regression coefficients, stored in a matrix

```
x <- model.matrix(crim ~., Boston)[, -1]
y <- Boston$crim

library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-4

grid <- 10^seq(10, -2, length=10)
set.seed(1)
train <- sample(1: nrow(x), nrow(x) / 2)
test <- (-train)
y.test <- y[test]
y.train <- y[train]

lasso.mod <- glmnet(x[train, ], y[train], alpha = 1, lambda = grid)
plot(lasso.mod)

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



From the coefficient plot that depending on the choice of tuning parameter, some of the coefficients will be exactly equal to zero. We now perform cross-validation turning the parameter lambda and compute the associated test error. Determine the best lambda by using cross-validation.

```r
#Create a 13X10 matrix, with 13 rows (for 12 predictors,plus one intercept),
10 columns (one for each value of lambda)

set.seed(1)
cv.out <- cv.glmnet(x[train, ], y[train], alpha = 1)
#plot(cv.out)
#find best lambda to predict
bestlam <- cv.out$lambda.min
bestlam

## [1] 0.01850161

lasso.pred <- predict(lasso.mod, s = bestlam, newx = x[test, ])
lasso.pred_train <- predict (lasso.mod, s = bestlam, newx = x[train, ])
#find the train MSE
mean((lasso.pred_train - y.train)^2)

## [1] 42.51117

#find the test MSE
mean((lasso.pred - y.test)^2)

## [1] 40.91307
```

The best lambda= 0,01850161 , the train MSE= 42.51117, the test MSE= 40.91307

(The Train MSE > Test MSE, which is unexpected. The code for train MSE I create and it is not from textbook. Maybe this is the reason as we already discuss this with you in chat.)

Finally, we refit Lasso regression model on the full data set, using the value of λ chosen by cross-validation, and examine the coefficient estimates.

```r
out <- glmnet(x, y, alpha =1,lambda=grid)

#predict(out, type="coefficients", s= bestlam )[1:13,]
lasso.coef <- predict(out, type="coefficients", s = bestlam)[1:13, ]

#lasso.coef <- predict(out, type="coefficients", s = bestlam, newx = x)
lasso.coef[lasso.coef !=0]

##  (Intercept)           zn        indus         chas          nox
 rm
## 12.389316431  0.042907277 -0.058524343 -0.772862768 -9.014759377  0.558638
998
##          dis          rad          tax      ptratio        lstat           m
edv
## -0.939296467  0.590919477 -0.002712841 -0.276998850  0.136825012 -0.206739
248
```

Unfortunately, Lasso select the model with 11 variables. Since this RStudio is not stable, which will appear different vale of coefficients. so that I use beta_number to represent the above coefficients for the model: Intercept is 12.39, zn's coef is 0.043 .....

Y (crim)= intercept+ beta_1*zn + beta_2*indus + beta_3*chas + beta_4*nox + ...beta_11*medv

Here the performance of Best subset selection method with lower test MSE is better than Lasso.

Section 5 Smoothing spline and GAM

Apply smoothing spline to one or more predictors. Fit a model to the training set to predict per capita crime rate using gam(). From Best subset selection find out the most effective predictors are: zn, nox, dis, rad, ptratio, medv.(when run the code, it may has different coefficients)

Some of those predictors may have their own polynomial function. A smoothing splines method will find the best value of knot for basis function of each predictor and fit a smooth curve to a set of data. known that rad: degree=6, find d (degree of polynomials) for zn.

```
#find best d of zn polynomial regression using cross-validation
set.seed(1)
cv.error.10=rep(0,10)
library(boot)
for (i in 1:10){
 glm.fit=glm(crim~poly(zn,i),data=Boston)
 cv.error.10[i]=cv.glm(Boston,glm.fit,K=10)$delta[1]
 }
 cv.error.10

##  [1] 71.25384 70.11898 70.02375 69.78716 70.01350 69.72330 69.75505 69.999
13
##  [9] 70.08599 69.75448
```

d=5 (zn ^5) is the best with the smallest cross-validation error

```
#find best degree for nox polynomial function by select the smallest MSE
noxlm.fit<-lm(crim ~ nox, data=Boston, subset=train)
mean((crim - predict(noxlm.fit, Boston))[-train]^2)

## [1] 55.2246

noxlm.fit2 <- lm(crim ~ poly(nox, 2), data= Boston, subset = train)
mean((crim - predict(noxlm.fit2, Boston))[-train]^2)

## [1] 53.80184

noxlm.fit3 <- lm(crim ~ poly(nox,3), data= Boston, subset = train)
mean((crim - predict(noxlm.fit3, Boston))[-train]^2)

## [1] 49.37537

noxlm.fit4 <- lm(crim ~ poly(nox,4), data= Boston, subset = train)
mean((crim - predict(noxlm.fit4, Boston))[-train]^2)

## [1] 49.37246
```

degree =2 (nox ^2) is the best with the smallest MSE

```
#find best d for dis polynomial regression
set.seed(1)
cv.error.10=rep(0,10)
library(boot)
for (i in 1:10){
 glm.fit=glm(crim~poly(dis,i),data=Boston)
 cv.error.10[i]=cv.glm(Boston,glm.fit,K=10)$delta[1]
 }
 cv.error.10

##  [1]   63.84038   57.61490   54.90961   53.09247   53.20065   53.32629
##  [7]  134.72136 1074.79423 1566.74268 3921.45148

#find best d for ptratio polynomial regression
set.seed(1)
cv.error.10=rep(0,10)
library(boot)
for (i in 1:10){
 glm.fit=glm(crim~poly(ptratio,i),data=Boston)
 cv.error.10[i]=cv.glm(Boston,glm.fit,K=10)$delta[1]
 }
 cv.error.10

##  [1] 68.17299 67.11087 66.36092 59.59994 64.97866 59.01387 55.97813 51.937
43
##  [9] 51.30971 69.12126
```

in this case I select smaller degree: d(ptratio)=4

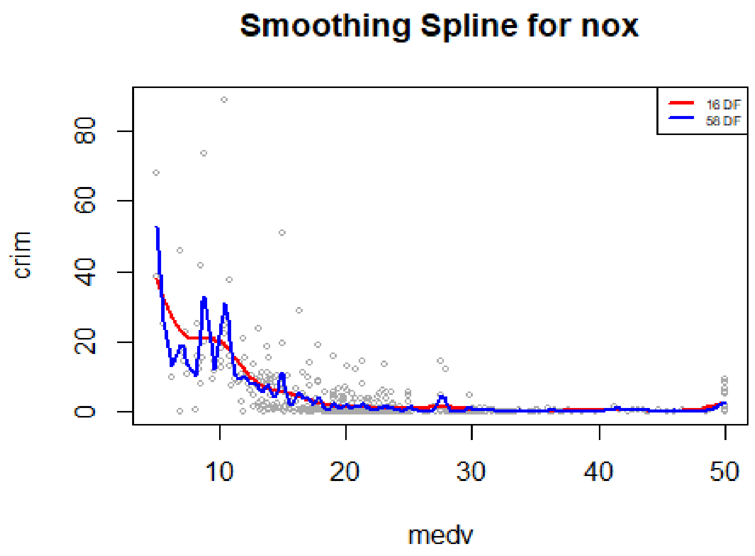
```
#find best d for medv polynomial regression
set.seed(1)
cv.error.10=rep(0,10)
library(boot)
for (i in 1:10){
 glm.fit=glm(crim~poly(medv,i),data=Boston)
 cv.error.10[i]=cv.glm(Boston,glm.fit,K=10)$delta[1]
 }
 cv.error.10

##  [1] 63.56089 48.38985 45.24890 42.67781 45.22372 43.85592 45.01620 44.323
82
##  [9] 45.64407 44.22817
```

d(medv)=4

Now try to fit the smoothing splines for nox

```
library(splines)
medvlims <- range(medv)
plot(medv,crim,xlim=medvlims, cex=.5, col="darkgrey")
title("Smoothing Spline for nox")
fit<-smooth.spline(medv, crim, df=16)
fit2<-smooth.spline(medv,crim,cv=TRUE)
```

```
## Warning in smooth.spline(medv, crim, cv = TRUE): cross-validation with non
-
## unique 'x' values seems doubtful

fit2$df

## [1] 65.19787

#Notice: The best df in here is 65.19787, it was different of 58 (once run th
e code, so I use 58 as best df)

lines(fit, col = "red", lwd = 2)
lines(fit2, col = "blue", lwd = 2)
legend("topright", legend = c("16 DF", "58 DF"),
    col = c("red", "blue"), lty = 1, lwd = 2, cex = .5)
```

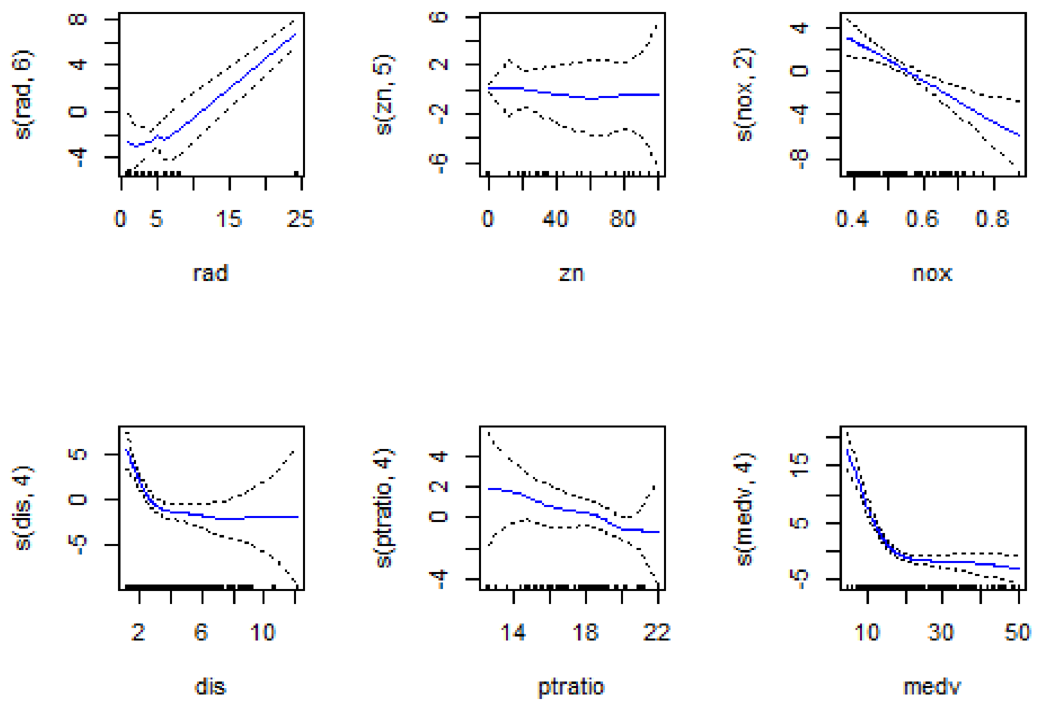


With an automatedly selected degree of freedom 58, medv polynomial function becomes more flexible, which may derive overfitting. Medv has a significant nonlinear feature, which should drop the smoothing spline fit.

Now fit train data and predict crim using GAM functions for 6 predictors.(all use smoothing spline) known that d(rad)=6, (show in section 2) d(zn)=5,d(nox)=2, d(dis)=4, d(ptratio)=4, d(medv)=4

```
set.seed(1)
train <-sample(506,405)
library(gam)

## Loading required package: foreach

## Loaded gam 1.20.2

gam <- gam(crim ~ s(rad,6) + s(zn,5) + s(nox,2) +s(dis,4)+s(ptratio,4) +s(med
v,4),
           data = Boston, subset=train)
```

```r
par(mfrow=c(2,3))
plot(gam, se=TRUE, col="blue")
```



```r
#find the MSE of train data
mean((crim - predict(gam, Boston))[train]^2)

## [1] 25.11552

#find the MSE of test data
mean((crim - predict(gam, Boston))[-train]^2)

## [1] 57.12355

summary(gam)

##
## Call: gam(formula = crim ~ s(rad, 6) + s(zn, 5) + s(nox, 2) + s(dis,
##     4) + s(ptratio, 4) + s(medv, 4), data = Boston, subset = train)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -15.3543  -1.0453  -0.0842   0.7517  53.6933
##
## (Dispersion Parameter for gaussian family taken to be 26.8389)
##
##     Null Deviance: 26134.02 on 404 degrees of freedom
## Residual Deviance: 10171.93 on 378.9999 degrees of freedom
## AIC: 2508.858
##
## Number of Local Scoring Iterations: NA
```

```
##
## Anova for Parametric Effects
##                 Df  Sum Sq Mean Sq  F value    Pr(>F)
## s(rad, 6)        1  6436.2  6436.2 239.8100 < 2.2e-16 ***
## s(zn, 5)         1   164.3   164.3   6.1222   0.01379 *
## s(nox, 2)        1    53.3    53.3   1.9858   0.15960
## s(dis, 4)        1    67.6    67.6   2.5184   0.11336
## s(ptratio, 4)    1    21.0    21.0   0.7815   0.37724
## s(medv, 4)       1  1241.3  1241.3  46.2514 4.064e-11 ***
## Residuals      379 10171.9    26.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                Npar Df Npar F     Pr(F)
## (Intercept)
## s(rad, 6)            5  0.567    0.7257
## s(zn, 5)             4  0.031    0.9981
## s(nox, 2)            1  0.293    0.5885
## s(dis, 4)            3  7.847 4.301e-05 ***
## s(ptratio, 4)        3  0.259    0.8549
## s(medv, 4)           3 36.126 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

look at the parametric table and assess the significance of the linear effect. Rad, zn, and medv has significant effect and can turn the term into a smooth s(x) -> x . Other predictors are insignificant you might consider dropping the term from the model entirely. in the nonparametric table, dis and medv sure can leave as smooth nonlinear effect.

```
coef(gam)

##   (Intercept)     s(rad, 6)      s(zn, 5)     s(nox, 2)     s(dis, 4)
##  24.269633988   0.457888666  -0.008231832 -18.487212056  -0.742924488
## s(ptratio, 4)    s(medv, 4)
##  -0.356443296  -0.240012557
```

the model by GAM fitting is:

Y(crim)=24.270 + 0.458*rad -0.008*zn -18.487*nox-0.743*dis -0.356*ptratio -0.240*medv

the model has a MSE in train data: 25.11552 , test MSE:57.12355

Conclusion:

The project's data provide a clear clue that leads us to statistical analysis. Start with estimating the data structure of each variable via histograms and boxplots. Par() and scaterplots() help to find the significant features affect the crime rate and correlation between variables. Then start to fit the model via simple multiple regression, best subset selection, Lasso and smoothing spline, improve the performance through comparing their test MSE. Even though the Rstudio performs unstable, still some results can prove that:

1. The model directly using validation set method with 6 predictors has a highest test MSE: 65.35408 and lower train MSE: 35.35255
2. a multiple regression model with all variables have train MSE: 34.83031. higher test MSE: 64.90121 .
3. Fit best subset selection data obtain 6 variables model with train MSE 35.30143 (full data), after cross-validation reduce the test MSE to 36.022031
4. When using Lasso obtain the model with 11 predictors and its train MSE= 42.51117, the test MSE= 40.91307
5. When apply smoothing spline to all 6 predictors, the model perform very well.it only has a train MSE 29.51791, and test MSE 33.49104

(Notice: above value of MSE will show in different when run the code.)

Obviously, in this case, smoothing spline model have the best performance with lowest test MSE, but we still need best subset selection method to determine the significant variables. In general, there is huge different in data structure that derive different results among above models.