Assignment 3

Deepak Karunakaran

October 16, 2015

Introduction

First part of the problem was to generate data acording to this criteria:

Responses should be linear functions of the rst 5 columns of X with coe?cients c(0.3,0.25,0.2,0.15,0.1) with an intercept equal to 0.5. y1 should be based on standard normal errors whereas y2 should be based on Laplace errors with parameters 0 and 1 for location and scale, respectively.

We follow the basic linear model to generate the linear function:

y=bx+c+e

```
    **where y is the response**
    **x  created using rnorm,**
    **b is given-c(0.5,0.3,0.25,0.2,0.15,0.1,0,0,0,0,0)- Note that b(0) is 0.5
is the intercept**
    **epsilon is the error term,**
    **based on a) normal and b) laplace.**
    **c is the intercept term**
```

for finding the laplace term we could use **rlaplace()** function from **Vgam** package or **rnl()** from **NormalLaplace** library in both the case we need to set the parameters **0 and 1 for location and scale**, respectively.

We have 3 different cases having **25,50,100** data points. All having **10** columns.Then we replicate the process three times to create three different x and y.

In the program:

x1-matrix of random variable numbers with dimensions(n x 10) where n=25,50,100.

y1-Response 1 having standard errors with dimensions(n x 3) where n=25,50,100. 3 different responses

y2-Response 2 having Laplace errors with dimensions(n x 3) where n=25,50,100. 3 different responses

Best subset regression

Where all possible potential regression models can be fitted. 2^p models were tested and the best one selected based on the BIC criterion given in the JASA paper was used on the value of k. Here we evaluvate all possible subsets regression.

rq was used and tau was set to 0.5 since it was best subset median regression the absolute deviation was found using fit$residual ###regsubsets() was used for the least squares based regression model.

```
library(NormalLaplace)
```

```
## Loading required package: DistributionUtils
## Loading required package: RUnit
## Loading required package: GeneralizedHyperbolic
```

```
library(leaps)
library(quantreg)
```

```
## Loading required package: SparseM
##
## Attaching package: 'SparseM'
##
## The following object is masked from 'package:base':
##
##     backsolve
```

```
M=10; #number of predictors
N=c(25,50,100); # number of rows
intercept=0.5;
beta=c(0.3,0.25,0.2,0.15,0.1,0,0,0,0,0)


x1=matrix(rnorm(N[1]*M,mean=0,sd=1), N[1], M)
y1=matrix(0, nrow = 25, ncol = 3)
y2=matrix(0, nrow = 25, ncol = 3)
```

We create the function L1bestsub that performs median regression

```
L1bestsub=function(y1,x1,k)
{
library(quantreg)
y1=y1[,1]
df1<-data.frame(y1,x1)
regfit = rq(y1~.,data=df1)
max.num.pred=9;
s=1;
I=1
absolute_deviation=rep(0,10);
k1_BIC=rep(0,9);
k2_BIC=rep(0,9);
penalty=ρτ=0
position=NULL;
posit=matrix(NA, nrow = 10, ncol = 10)

BIC=penalty=0 # stores the smallest error for each number of predictors
N=length(y1);
residual=matrix(NA, nrow = 480, ncol = 12)
min_residual=rep(0,10)

###Coefficents and Confidence intervals obtained for n=100 and standard normal
errors
print("Coefficents and Confidence intervals obtained")
print(summary(regfit))


for(s in 1:max.num.pred)
{
  BICsmall=1000;     #intial big BIC value to easily find samller BIC values
  regfits=rep(0,11)  # initial predictor matrix
  pick=NULL;
  pick=combn(regfit$coefficients[2:11],s) # all combinations of 's' prdectors
  count=1;
  y=y1;
  while(count!=dim(pick)[2])   # Loop to find minimum BIC value for all combina
tions of 's' predictors
  {
    loc=regfit$coefficients%in%pick[,count] #stores location of current predict
ors used
    x=x1[,loc[2:10]]
    residual[count,s]=sum(rq(y1~x)$residuals)
    regfits[loc]=regfit$coefficients[loc]   #current predictor under testing
    regfits=na.omit(regfits)
    u=0;  # loss function
    #find sum of loss function over all the data points
```

```
     count=count+1;
  }
  min=which.min(abs(residual[,s]))
  absolute_deviation[s]=abs(min(residual[,s],na.rm=TRUE))
  loc=regfit$coefficients%in%pick[,min]
  u= abs(min(residual[,s],na.rm=TRUE))
  if(k==1) {
    penalty=s*log(N)/(2*N);    #penalty term for loss function
     τ=0.5;#for median regression
    ρτ=abs(u*(2*τ-2*I))
    k1_BIC[s]=(log((ρτ))+penalty); #calculate BIC
    }

  if (k==2)
  { penalty=s*log(N);
    k2_BIC[s]=abs((ρτ))+penalty; #calculate BIC
    τ=0.5;#for median regression
    ρτ=u*(2*τ-2*I)
    }
position=list(position,t(as.matrix(which(loc))))
}
 position=unlist(position)

     #storing the position of the smallest value

  #k1_BIC=k2_BIC=penalty=ρτ=0
  if(k==1)
  {
print("K1 values of BIC")

print(k1_BIC)  #for n=100 and standard normal errors
print("Best performing BIC has predictors:")
    print(which.min(k1_BIC)) #for n=100 and standard normal errors
    plot(k1_BIC[1:9],ylab="BIC value",xlab="number of predictors",main = "BIC v
alues vs predictors",type='l')



    }
 else
    {
      ###K1 values of BIC

 print("K2 values of BIC")
print(k2_BIC)  #for n=100 and standard normal errors
print("Best performing BIC has predictors:")
    print(which.min(k2_BIC)) #for n=100 and standard normal errors
```

```
      plot(k2_BIC[1:9],ylab="BIC value",xlab="number of predictors",main = "BIC
 values vs predictors",type='l')

 }
print("Best performing predictors for different subsets from p=1:9")
print(position[1])
print(position[2:3])
print(position[4:6])
print(position[7:10])
print(position[11:15])
print(position[16:21])
print(position[22:28])
print(position[29:36])
print(position[37:45])
}
```

25 rows- Laplace and standard normal errors and three responses each

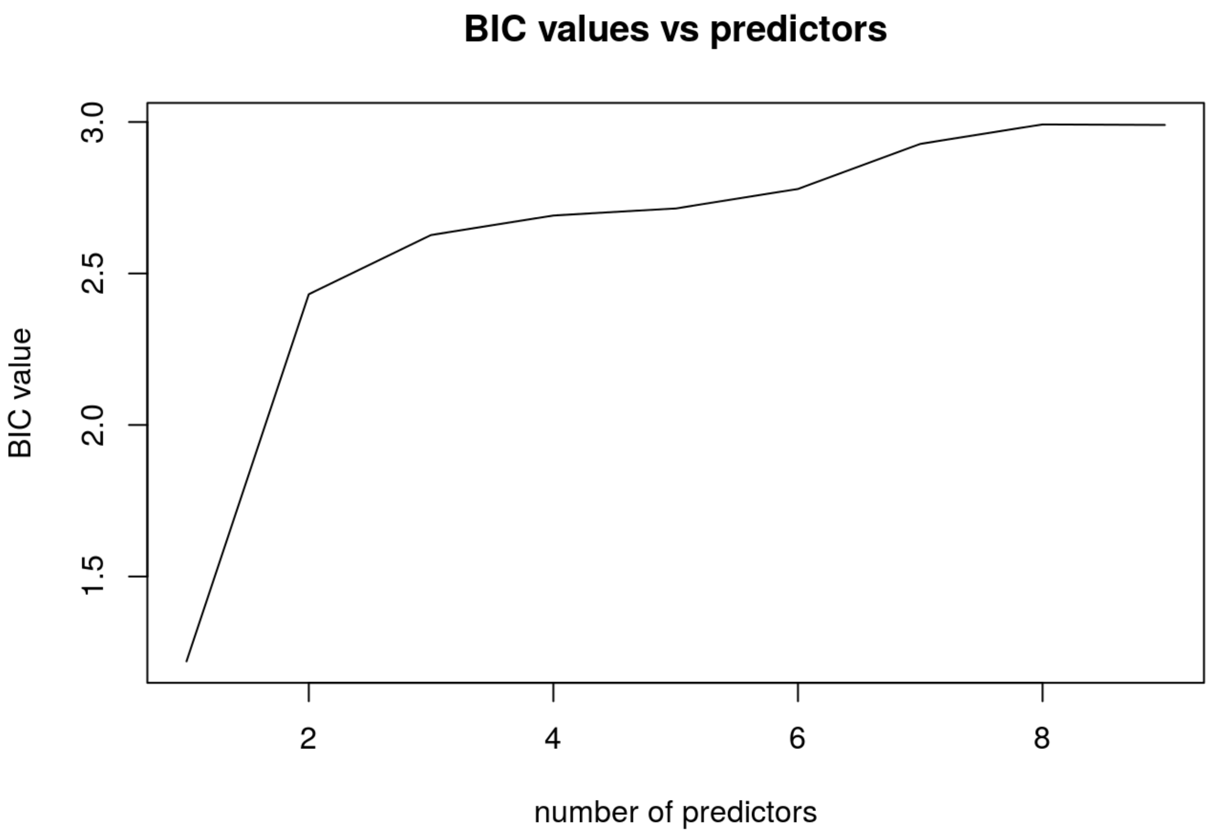
```
df1=data.frame(x1,y1)
df2=data.frame(x1,y1)
for(j in 1:3)
{
  x1=matrix(rnorm(N[1]*M,mean=0,sd=1), N[1], M)
  for(i in 1:25)
  {
beta=c(0.3,0.25,0.2,0.15,0.1,0,0,0,0,0)
error.rnorm=rnorm(1)
error.lap=rnl(1,0,1)
y1[i,j]=intercept+beta%*%x1[i,]+error.rnorm
y2[i,j]=intercept+beta%*%x1[i,]+error.lap
}
  regfit_25.rnorm=regsubsets(y1[,j]~x1[,1]+x1[,2]+x1[,3]+x1[,4]+x1[,5]+x1[,6]+x
1[,7]+x1[,8]+x1[,9]+x1[,10],df1,nvmax=10);
  regfit_25.laplace=regsubsets(y2[,j]~x1[,1]+x1[,2]+x1[,3]+x1[,4]+x1[,5]+x1[,6]
+x1[,7]+x1[,8]+x1[,9]+x1[,10],df2,nvmax=10);
}
```

Comparing results of median regression and least squares for laplace and normal errors(N=25)

some results for n=25 for median regression

```
L1bestsub(y2,x1,1)# laplace 25 rows k=1
```

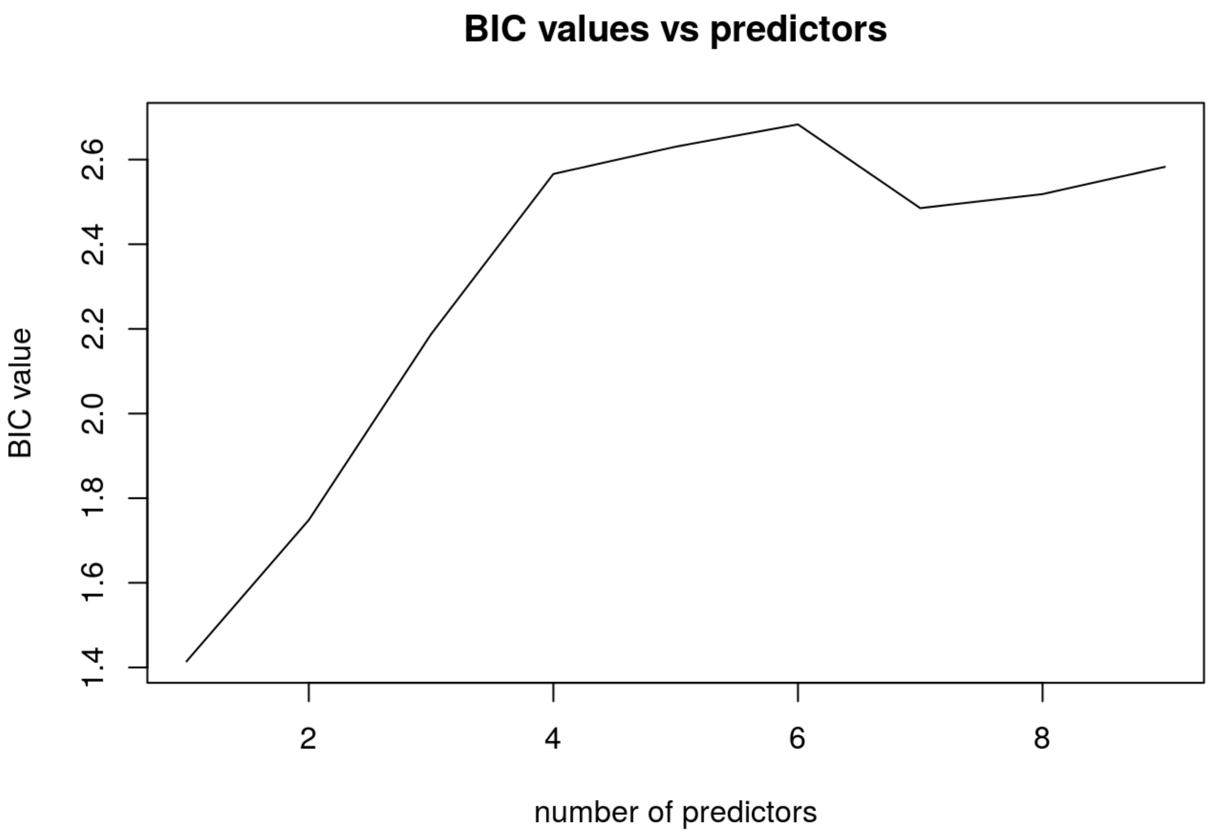
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.09020     -0.18932  1.15570
## X1          -0.66669     -1.06039  0.49538
## X2           0.50579     -0.64241  0.64508
## X3           0.25494     -0.66430  1.46388
## X4          -0.50386     -0.85048  0.18527
## X5           1.13064     -0.44496  1.67908
## X6           0.11573     -0.75511  0.66606
## X7          -0.47138     -0.52060  0.55986
## X8           0.18823     -0.13352  1.43126
## X9          -1.12804     -1.66450  0.50561
## X10         -0.57927     -0.88934  0.81509
## [1] "K1 values of BIC"
## [1] 1.219801 2.431533 2.626935 2.691313 2.714532 2.778909 2.927588 2.991965
## [9] 2.990225
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 6
## [1] 4 9
## [1]  2  4 10
## [1]  2  3  9 10
## [1]  2  3  9 10 11
## [1] 2 4 6 7 8 9
## [1]  2  4  6  7  8  9 11
## [1]  3  4  5  6  7  8  9 10
## [1]  2  3  4  5  6  7  8  9 10
```

```
L1bestsub(y1,x1,1)# normal errors 25 rows k=1
```

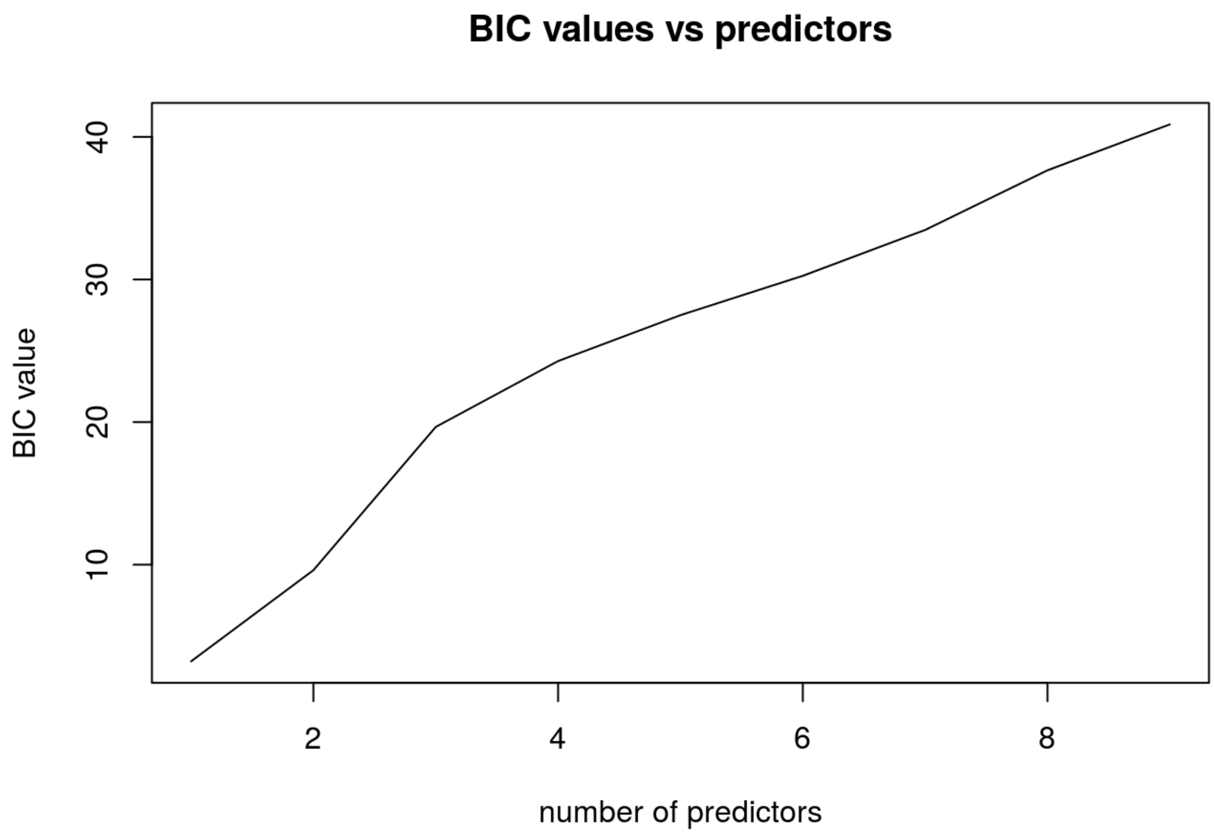
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  1.08783     -0.05971  1.61751
## X1           0.15393     -0.17180  0.45779
## X2          -0.29793     -0.75698  0.58745
## X3          -0.20090     -0.77249  0.55978
## X4           0.42127     -0.30453  0.75304
## X5          -0.89448     -0.93011 -0.47540
## X6          -0.36129     -0.54799  0.15385
## X7           0.24878     -0.53383  0.82742
## X8          -0.21471     -0.93380 -0.15243
## X9           0.07362     -0.63508  0.32598
## X10         -0.01898     -0.60066  0.75087
## [1] "K1 values of BIC"
## [1] 1.414460 1.748461 2.187584 2.566130 2.630508 2.683133 2.485303 2.518509
## [9] 2.582887
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 3
## [1]  3 11
## [1] 5 6 7
## [1] 2 3 7 8
## [1] 2 3 4 5 8
## [1]  2  3  4  5  8 11
## [1]  3  4  5  7  9 10 11
## [1]  2  4  5  6  7  8  9 11
## [1]  2  3  4  5  6  7  8  9 10
```

```
L1bestsub(y2,x1,2)# laplace 25 rows k=2
```

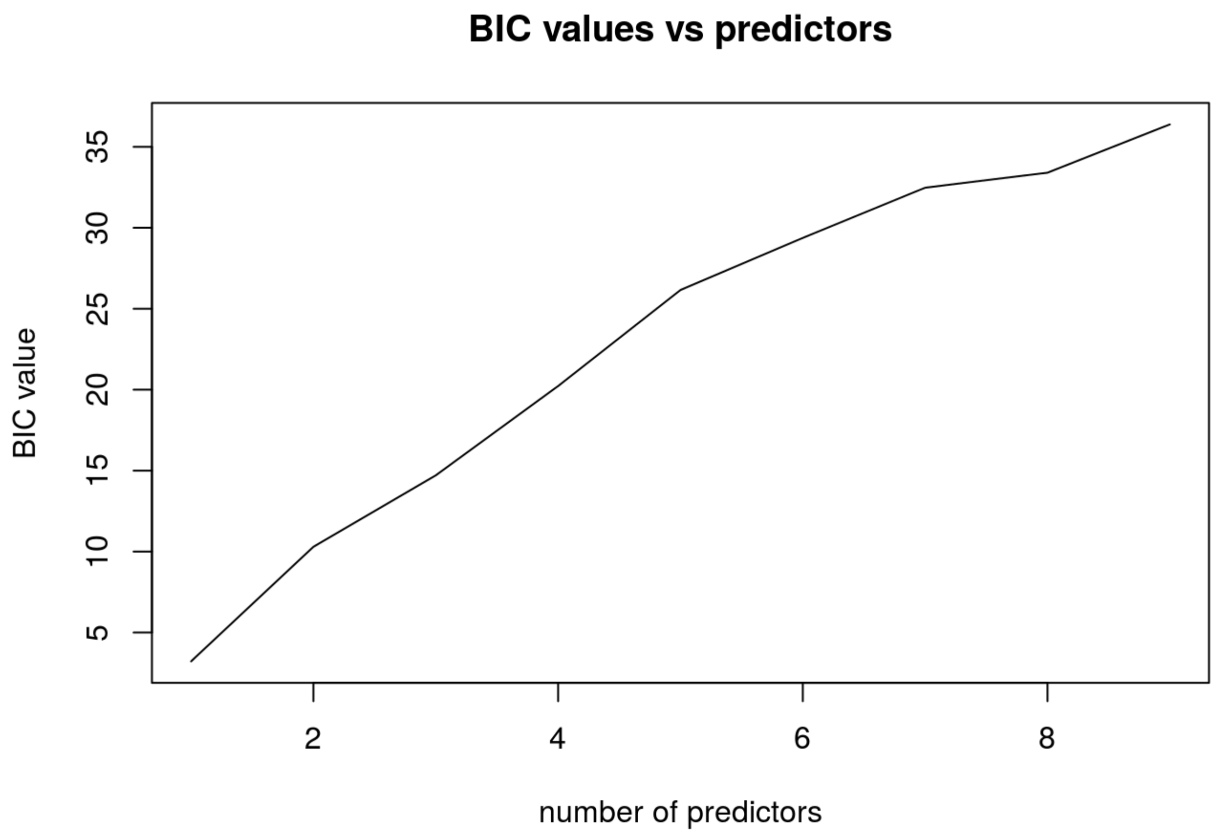
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.09020     -0.18932  1.15570
## X1          -0.66669     -1.06039  0.49538
## X2           0.50579     -0.64241  0.64508
## X3           0.25494     -0.66430  1.46388
## X4          -0.50386     -0.85048  0.18527
## X5           1.13064     -0.44496  1.67908
## X6           0.11573     -0.75511  0.66606
## X7          -0.47138     -0.52060  0.55986
## X8           0.18823     -0.13352  1.43126
## X9          -1.12804     -1.66450  0.50561
## X10         -0.57927     -0.88934  0.81509
## [1] "K2 values of BIC"
## [1]  3.218876  9.613119 19.658555 24.277664 27.496540 30.255643 33.474519
## [8] 37.655848 40.874724
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 6
## [1] 4 9
## [1]  2  4 10
## [1]  2  3  9 10
## [1]  2  3  9 10 11
## [1] 2 4 6 7 8 9
## [1]  2  4  6  7  8  9 11
## [1]  3  4  5  6  7  8  9 10
## [1]  2  3  4  5  6  7  8  9 10
```

```
L1bestsub(y1,x1,2)# normal errors 25 rows k=2
```

```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  1.08783     -0.05971  1.61751
## X1           0.15393     -0.17180  0.45779
## X2          -0.29793     -0.75698  0.58745
## X3          -0.20090     -0.77249  0.55978
## X4           0.42127     -0.30453  0.75304
## X5          -0.89448     -0.93011 -0.47540
## X6          -0.36129     -0.54799  0.15385
## X7           0.24878     -0.53383  0.82742
## X8          -0.21471     -0.93380 -0.15243
## X9           0.07362     -0.63508  0.32598
## X10         -0.01898     -0.60066  0.75087
## [1] "K2 values of BIC"
## [1]  3.218876 10.295494 14.708232 20.223678 26.154910 29.373786 32.475120
## [8] 33.400661 36.384763
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 3
## [1]  3 11
## [1] 5 6 7
## [1] 2 3 7 8
## [1] 2 3 4 5 8
## [1]  2  3  4  5  8 11
## [1]  3  4  5  7  9 10 11
## [1]  2  4  5  6  7  8  9 11
## [1]  2  3  4  5  6  7  8  9 10
```

Summary of 25 rows Laplace and standard normal errors and three responses for least squares

```
rnorm.25=summary(regfit_25.rnorm) #best models for 25 rows with normal errors
lap.25=summary(regfit_25.laplace)
rnorm.25$outmat #best models for 25 rows with normal errors
```

```
##           x1[, 1] x1[, 2] x1[, 3] x1[, 4] x1[, 5] x1[, 6] x1[, 7] x1[, 8]
## 1  ( 1 )  " "     " "     " "     " "     " "     " "     " "     " "    
## 2  ( 1 )  "*"     " "     " "     " "     " "     " "     " "     " "    
## 3  ( 1 )  "*"     " "     " "     " "     " "     "*"     " "     " "    
## 4  ( 1 )  "*"     " "     " "     "*"     " "     "*"     " "     " "    
## 5  ( 1 )  "*"     " "     " "     "*"     " "     "*"     "*"     " "    
## 6  ( 1 )  "*"     " "     " "     "*"     "*"     "*"     "*"     " "    
## 7  ( 1 )  "*"     "*"     " "     "*"     "*"     "*"     "*"     " "    
## 8  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     "*"     " "    
## 9  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     "*"     "*"    
## 10  ( 1 ) "*"     "*"     "*"     "*"     "*"     "*"     "*"     "*"    
##           x1[, 9] x1[, 10]
## 1  ( 1 )  "*"     " "     
## 2  ( 1 )  "*"     " "     
## 3  ( 1 )  "*"     " "     
## 4  ( 1 )  "*"     " "     
## 5  ( 1 )  "*"     " "     
## 6  ( 1 )  "*"     " "     
## 7  ( 1 )  "*"     " "     
## 8  ( 1 )  "*"     " "     
## 9  ( 1 )  "*"     " "     
## 10  ( 1 ) "*"     "*"     
```

```
lap.25$outmat  #best models for 50 rows with laplace errors
```

```
##           x1[, 1] x1[, 2] x1[, 3] x1[, 4] x1[, 5] x1[, 6] x1[, 7] x1[, 8]
## 1  ( 1 )  " "     "*"     " "     " "     " "     " "     " "     " "
## 2  ( 1 )  " "     "*"     " "     " "     " "     "*"     " "     " "
## 3  ( 1 )  " "     "*"     " "     "*"     " "     "*"     " "     " "
## 4  ( 1 )  " "     "*"     " "     "*"     "*"     "*"     " "     " "
## 5  ( 1 )  "*"     "*"     " "     "*"     "*"     "*"     " "     " "
## 6  ( 1 )  "*"     "*"     " "     "*"     "*"     "*"     " "     " "
## 7  ( 1 )  "*"     "*"     "*"     " "     "*"     "*"     " "     "*"
## 8  ( 1 )  "*"     "*"     "*"     " "     "*"     "*"     "*"     "*"
## 9  ( 1 )  "*"     "*"     "*"     " "     "*"     "*"     "*"     "*"
## 10  ( 1 ) "*"     "*"     "*"     "*"     "*"     "*"     "*"     "*"
##           x1[, 9] x1[, 10]
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     " "
## 4  ( 1 )  " "     " "
## 5  ( 1 )  " "     " "
## 6  ( 1 )  " "     "*"
## 7  ( 1 )  "*"     " "
## 8  ( 1 )  "*"     " "
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
```

```
reg.summary=summary(regfit_25.rnorm)
names(reg.summary)
```

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

```
reg.summary$rsq
```

```
##  [1] 0.1990650 0.3353867 0.4211980 0.4679433 0.5070917 0.5234082 0.5281111
##  [8] 0.5313880 0.5327751 0.5329442
```

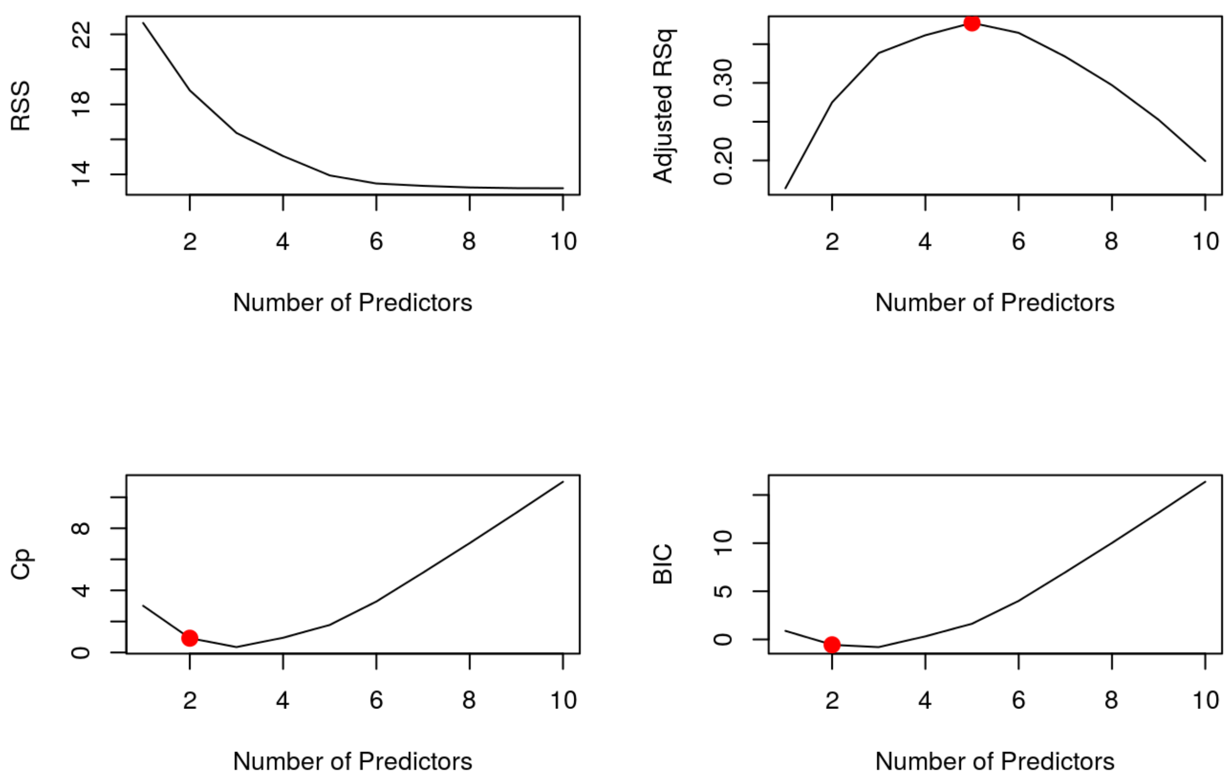
```
par(mfrow=c(2,2))
plot(reg.summary$rss,xlab="Number of Predictors",ylab="RSS",type="l")
plot(reg.summary$adjr2,xlab="Number of Predictors",ylab="Adjusted RSq",type="l"
)
max=which.max(reg.summary$adjr2)
points(5,reg.summary$adjr2[5], col="red",cex=2,pch=20)
plot(reg.summary$cp,xlab="Number of Predictors",ylab="Cp",type='l')
which.min(reg.summary$cp)
```

```
## [1] 3
```

```
points(2,reg.summary$cp[2],col="red",cex=2,pch=20)
which.min(reg.summary$bic)
```

```
## [1] 3
```

```
plot(reg.summary$bic,xlab="Number of Predictors",ylab="BIC",type='l')
points(2,reg.summary$bic[2],col="red",cex=2,pch=20)
```



50 rows Laplace and standard normal errors and three responses

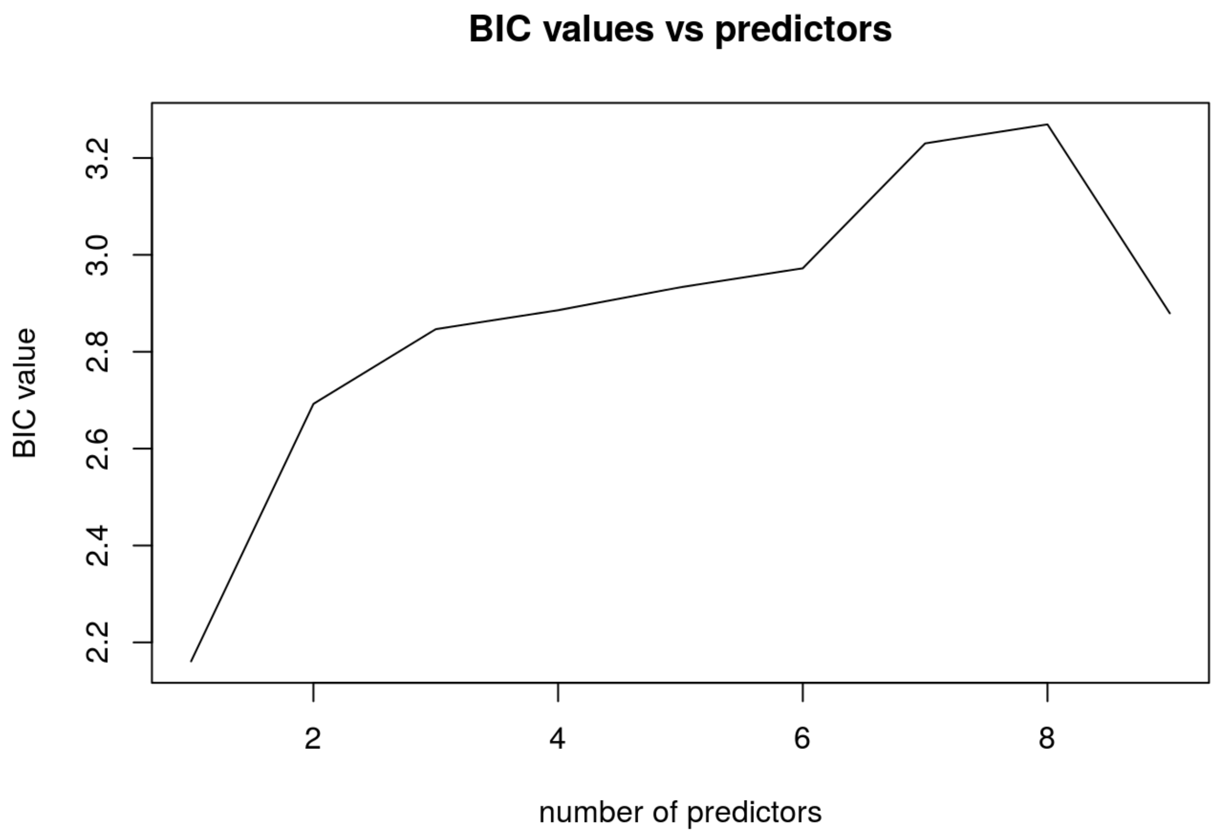
```
x1=matrix(rnorm(N[2]*M,mean=0,sd=1), N[2], M)
y1=matrix(0, nrow = 50, ncol = 3)
y2=matrix(0, nrow = 50, ncol = 3)

for(j in 1:3)
{
  x1=matrix(rnorm(N[2]*M,mean=0,sd=1), N[2], M)
  for(i in 1:50)
  {
    error.rnorm=rnorm(1)
    error.lap=rnl(1,0,1)
    y1[i,j]=intercept+beta%*%x1[i,]+error.rnorm
    y2[i,j]=intercept+beta%*%x1[i,]+error.lap
          }

  regfit_50.rnorm=regsubsets(y2[,j]~x1[,1]+x1[,2]+x1[,3]+x1[,4]+x1[,5]+x1[,6]+x
1[,7]+x1[,8]+x1[,9]+x1[,10],df1,nvmax=10);
  regfit_50.laplace=regsubsets(y2[,j]~x1[,1]+x1[,2]+x1[,3]+x1[,4]+x1[,5]+x1[,6]
+x1[,7]+x1[,8]+x1[,9]+x1[,10],df2,nvmax=10);
}
```

Comparing results of median regression and least squares for laplace and normal errors(N=50)

some results for n=50 for median regression

```
L1bestsub(y2,x1,1)# laplace 50 rows k=1
```

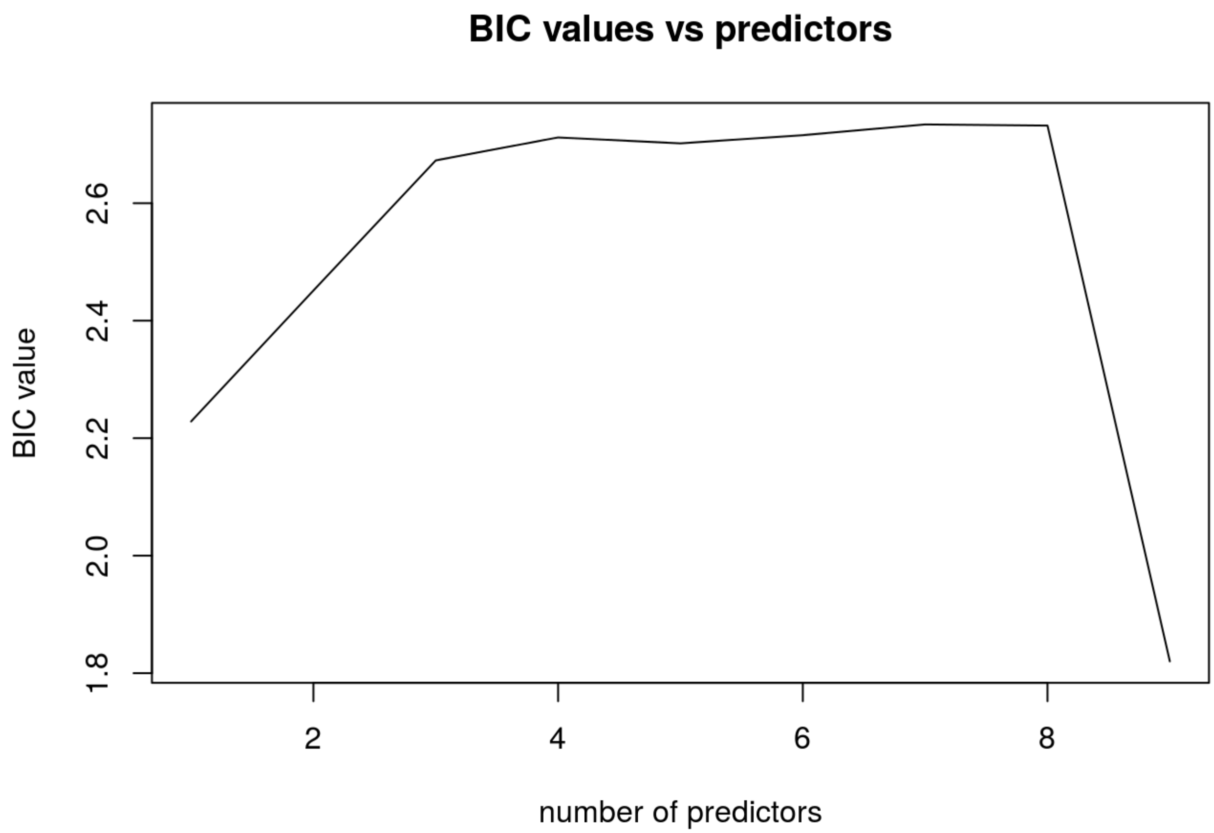
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  1.28462      0.92996  2.09733
## X1          -0.11802     -0.67690  0.09958
## X2          -0.20913     -0.48454 -0.07168
## X3           0.26874     -0.41077  0.58186
## X4           0.51566      0.06651  1.20954
## X5           0.34246     -0.44040  0.72133
## X6           0.58451     -0.17753  0.84569
## X7          -0.28203     -0.46225  1.03512
## X8           0.19465     -0.05476  0.63782
## X9          -0.31915     -1.06425  0.18199
## X10         -0.01625     -0.66888  0.21458
## [1] "K1 values of BIC"
## [1] 2.160868 2.692352 2.846584 2.885704 2.933167 2.972287 3.230141 3.269262
## [9] 2.879314
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 10
## [1]  8 10
## [1] 6 7 9
## [1]  5  7  8 10
## [1]  2  6  7  8 10
## [1]  2  6  7  8 10 11
## [1]  3  4  5  7  8  9 10
## [1]  3  4  5  7  8  9 10 11
## [1]  2  3  4  6  7  8  9 10 11
```

```
L1bestsub(y1,x1,1)# normal errors 50 rows k=1
```

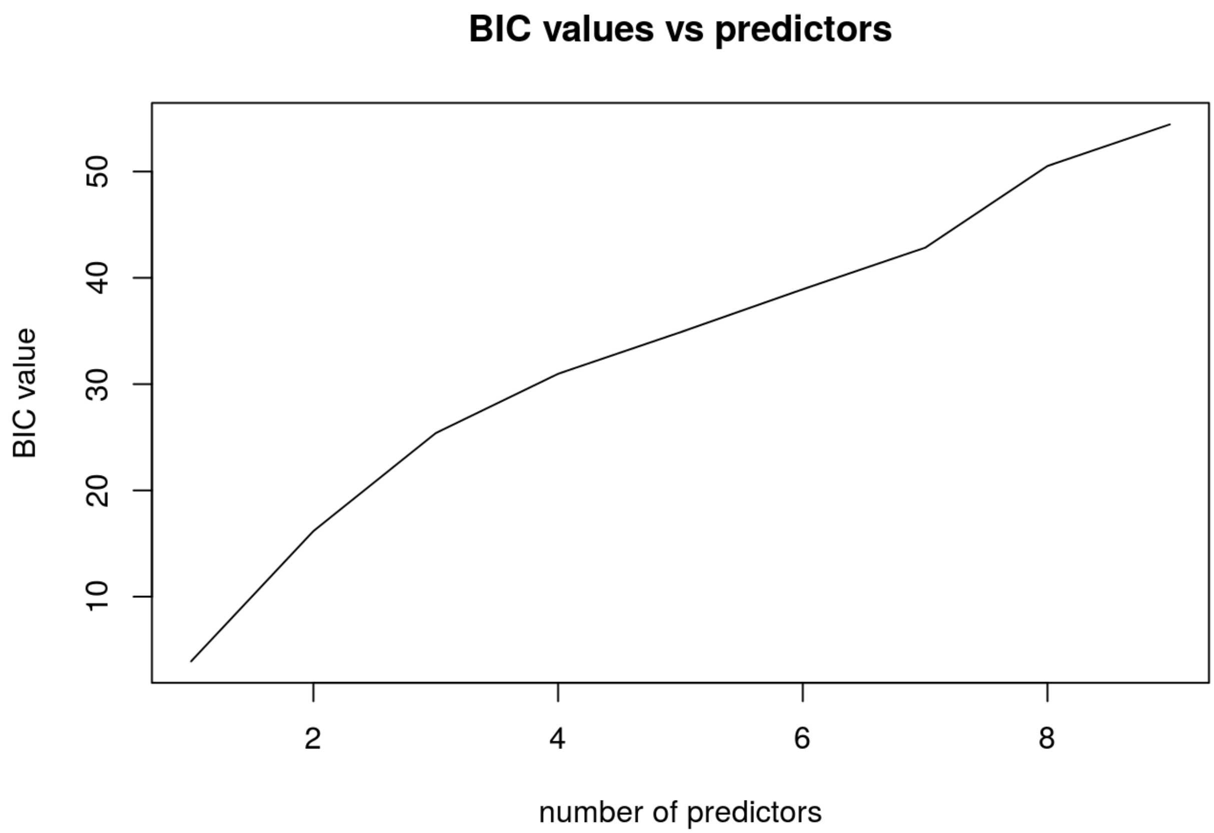
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.66996      0.56497  1.07822
## X1           0.27969     -0.20485  0.59267
## X2           0.00546     -0.26419  0.23541
## X3           0.23760     -0.08983  0.63457
## X4           0.06157     -0.15423  0.33593
## X5          -0.31411     -0.62594  0.23436
## X6          -0.12672     -0.36321  0.22254
## X7           0.25556     -0.57489  0.72132
## X8          -0.18875     -0.31161  0.22302
## X9           0.12437     -0.33802  0.16653
## X10          0.03656     -0.22303  0.20685
## [1] "K1 values of BIC"
## [1] 2.228431 2.451345 2.672834 2.711955 2.701891 2.715814 2.734119 2.732187
## [9] 1.820086
## [1] "Best performing BIC has predictors:"
## [1] 9
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 6
## [1] 2 4
## [1] 2 4 8
## [1] 2 3 4 7
## [1]  2  3  4  7 11
## [1]  4  5  6  9 10 11
## [1]  2  3  6  8  9 10 11
## [1]  2  3  4  6  7  8  9 10
## [1]  2  3  4  6  7  8  9 10 11
```

```
L1bestsub(y2,x1,2)# laplace 50 rows k=2
```

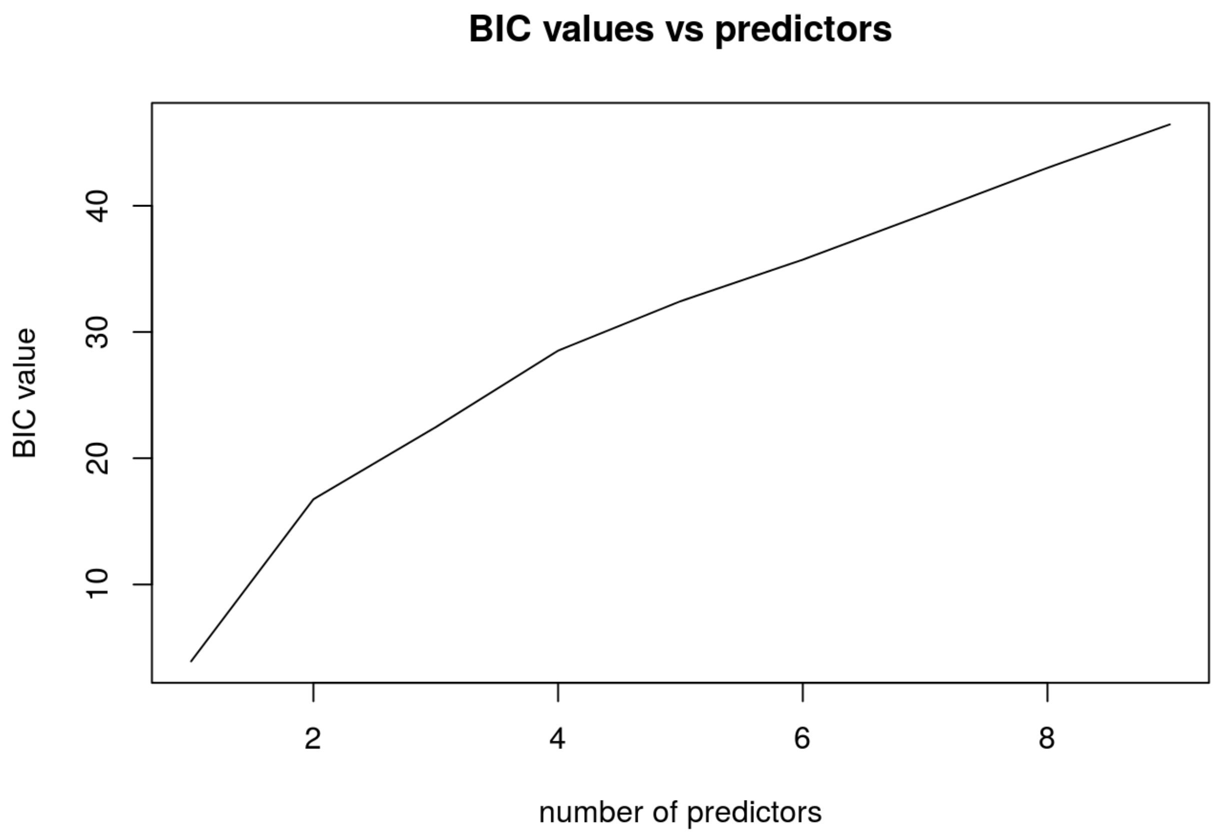
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  1.28462      0.92996  2.09733
## X1          -0.11802     -0.67690  0.09958
## X2          -0.20913     -0.48454 -0.07168
## X3           0.26874     -0.41077  0.58186
## X4           0.51566      0.06651  1.20954
## X5           0.34246     -0.44040  0.72133
## X6           0.58451     -0.17753  0.84569
## X7          -0.28203     -0.46225  1.03512
## X8           0.19465     -0.05476  0.63782
## X9          -0.31915     -1.06425  0.18199
## X10         -0.01625     -0.66888  0.21458
## [1] "K2 values of BIC"
## [1]  3.912023 16.169761 25.391153 30.969079 34.881102 38.921465 42.833488
## [8] 50.522882 54.434905
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 10
## [1]  8 10
## [1] 6 7 9
## [1]  5  7  8 10
## [1]  2  6  7  8 10
## [1]  2  6  7  8 10 11
## [1]  3  4  5  7  8  9 10
## [1]  3  4  5  7  8  9 10 11
## [1]  2  3  4  6  7  8  9 10 11
```

```
L1bestsub(y1,x1,2)# normal errors 50 rows k=2
```

```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.66996      0.56497  1.07822
## X1           0.27969     -0.20485  0.59267
## X2           0.00546     -0.26419  0.23541
## X3           0.23760     -0.08983  0.63457
## X4           0.06157     -0.15423  0.33593
## X5          -0.31411     -0.62594  0.23436
## X6          -0.12672     -0.36321  0.22254
## X7           0.25556     -0.57489  0.72132
## X8          -0.18875     -0.31161  0.22302
## X9           0.12437     -0.33802  0.16653
## X10          0.03656     -0.22303  0.20685
## [1] "K2 values of BIC"
## [1]  3.912023 16.753102 22.466724 28.525492 32.437515 35.731498 39.338481
## [8] 43.004248 46.445356
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 6
## [1] 2 4
## [1] 2 4 8
## [1] 2 3 4 7
## [1]  2  3  4  7 11
## [1]  4  5  6  9 10 11
## [1]  2  3  6  8  9 10 11
## [1]  2  3  4  6  7  8  9 10
## [1]  2  3  4  6  7  8  9 10 11
```

Summary of 50 rows Laplace and standard normal errors using median regression

```
#summary(regfit_50.rnorm)
rnorm.50=summary(regfit_50.laplace)
#summary(regfit_50.laplace)
lap.50=summary(regfit_50.laplace)
rnorm.50$outmat #best models for 50 rows with normal errors
```

```
##           x1[, 1] x1[, 2] x1[, 3] x1[, 4] x1[, 5] x1[, 6] x1[, 7] x1[, 8]
## 1  ( 1 )  "*"     " "     " "     " "     " "     " "     " "     " "    
## 2  ( 1 )  "*"     " "     " "     " "     " "     "*"     " "     " "    
## 3  ( 1 )  "*"     " "     "*"     " "     " "     "*"     " "     " "    
## 4  ( 1 )  "*"     "*"     " "     " "     " "     "*"     " "     "*"    
## 5  ( 1 )  "*"     "*"     "*"     " "     " "     "*"     " "     "*"    
## 6  ( 1 )  "*"     "*"     "*"     "*"     " "     "*"     " "     "*"    
## 7  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"    
## 8  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"    
## 9  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"    
## 10  ( 1 ) "*"     "*"     "*"     "*"     "*"     "*"     "*"     "*"    
##           x1[, 9] x1[, 10]
## 1  ( 1 )  " "     " "     
## 2  ( 1 )  " "     " "     
## 3  ( 1 )  " "     " "     
## 4  ( 1 )  " "     " "     
## 5  ( 1 )  " "     " "     
## 6  ( 1 )  " "     " "     
## 7  ( 1 )  " "     " "     
## 8  ( 1 )  " "     "*"     
## 9  ( 1 )  "*"     "*"     
## 10  ( 1 ) "*"     "*"     
```

```
lap.50$outmat #best models for 50 rows with laplace errors
```

```
##           x1[, 1] x1[, 2] x1[, 3] x1[, 4] x1[, 5] x1[, 6] x1[, 7] x1[, 8]
## 1  ( 1 )  "*"     " "     " "     " "     " "     " "     " "     " "
## 2  ( 1 )  "*"     " "     " "     " "     " "     "*"     " "     " "
## 3  ( 1 )  "*"     " "     "*"     " "     " "     "*"     " "     " "
## 4  ( 1 )  "*"     "*"     " "     " "     " "     "*"     " "     "*"
## 5  ( 1 )  "*"     "*"     "*"     " "     " "     "*"     " "     "*"
## 6  ( 1 )  "*"     "*"     "*"     "*"     " "     "*"     " "     "*"
## 7  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"
## 8  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"
## 9  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"
## 10  ( 1 ) "*"     "*"     "*"     "*"     "*"     "*"     "*"     "*"
##           x1[, 9] x1[, 10]
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     " "
## 4  ( 1 )  " "     " "
## 5  ( 1 )  " "     " "
## 6  ( 1 )  " "     " "
## 7  ( 1 )  " "     " "
## 8  ( 1 )  " "     "*"
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
```

```
reg.summary=summary(regfit_50.rnorm)
names(reg.summary)
```

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

```
reg.summary$rsq
```

```
##  [1] 0.1301734 0.2440023 0.2902968 0.3400185 0.3752347 0.3941519 0.4048322
##  [8] 0.4132012 0.4133354 0.4134052
```

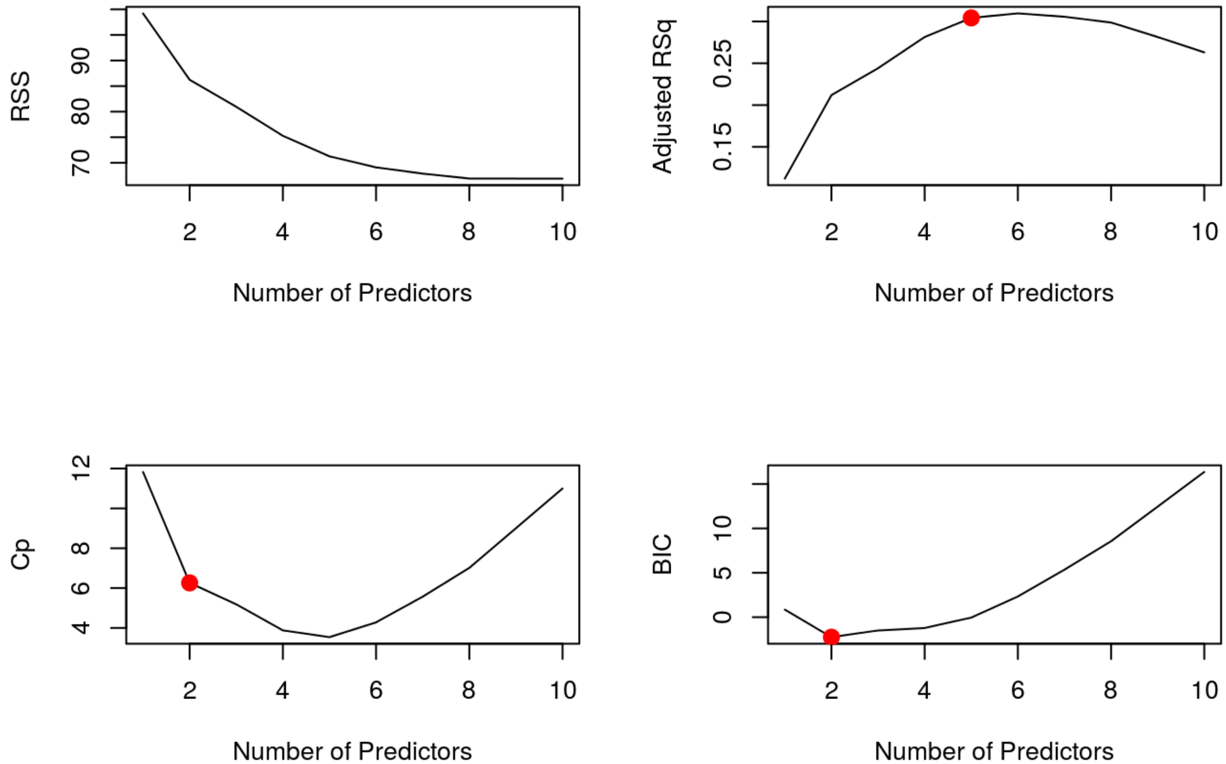
```
par(mfrow=c(2,2))
plot(reg.summary$rss,xlab="Number of Predictors",ylab="RSS",type="l")
plot(reg.summary$adjr2,xlab="Number of Predictors",ylab="Adjusted RSq",type="l"
)
max=which.max(reg.summary$adjr2)
points(5,reg.summary$adjr2[5], col="red",cex=2,pch=20)
plot(reg.summary$cp,xlab="Number of Predictors",ylab="Cp",type='l')
which.min(reg.summary$cp)
```

```
## [1] 5
```

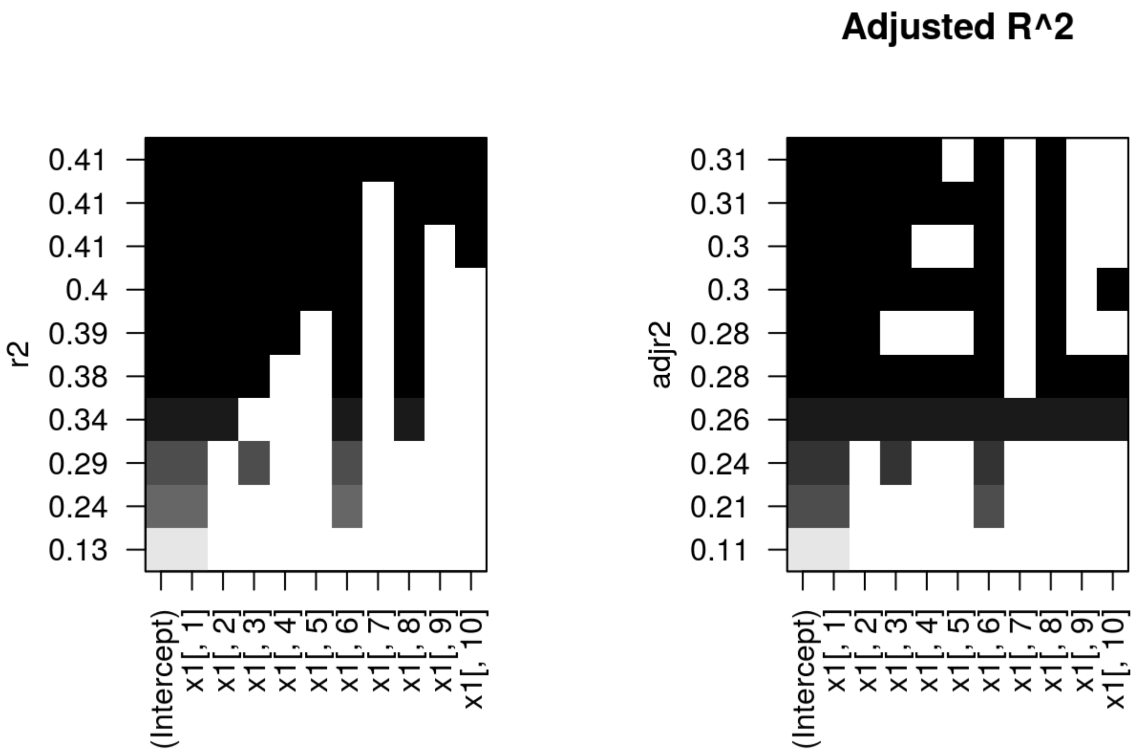
```
points(2,reg.summary$cp[2],col="red",cex=2,pch=20)
which.min(reg.summary$bic)
```

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

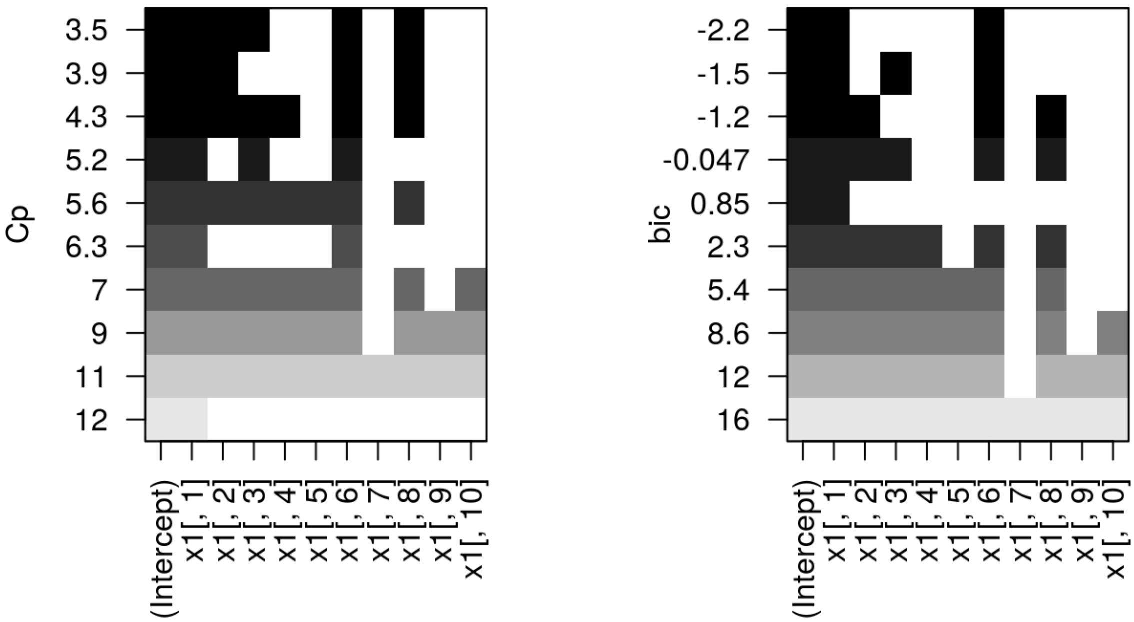
```
plot(reg.summary$bic,xlab="Number of Predictors",ylab="BIC",type='l')
points(2,reg.summary$bic[2],col="red",cex=2,pch=20)
```




```
par(mfrow=c(1,2))
plot(regfit_50.rnorm,scale="r2")
plot(regfit_50.rnorm,scale="adjr2",main = "Adjusted R^2")
```




```
par(mfrow=c(1,2))
plot(regfit_50.rnorm,scale="Cp")
plot(regfit_50.rnorm,scale="bic")
```



100 rows Laplace and standard normal errors and three responses

```
x1=matrix(rnorm(N[3]*M,mean=0,sd=1), N[3], M)
y1=matrix(0, nrow = 100, ncol = 3)
y2=matrix(0, nrow = 100, ncol = 3)

for(j in 1:3)
{
  x1=matrix(rnorm(N[3]*M,mean=0,sd=1), N[3], M)
  for(i in 1:100)
  {
    error.rnorm=rnorm(1)
    error.lap=rnl(1,0,1)
    y1[i,j]=intercept+beta%*%x1[i,]+error.rnorm
    y2[i,j]=intercept+beta%*%x1[i,]+error.lap
  }

  regfit_100.rnorm=regsubsets(y2[,j]~x1[,1]+x1[,2]+x1[,3]+x1[,4]+x1[,5]+x1[,6]+
x1[,7]+x1[,8]+x1[,9]+x1[,10],df1,nvmax=10);
  regfit_100.laplace=regsubsets(y2[,j]~x1[,1]+x1[,2]+x1[,3]+x1[,4]+x1[,5]+x1[,6
]+x1[,7]+x1[,8]+x1[,9]+x1[,10],df2,nvmax=10);
}
```

Comparing results of median regression and least squares for laplace and normal errors(N=50)

some results for n=50 for median regression

```
rnorm.100=summary(regfit_100.rnorm)
lap.100=summary(regfit_100.laplace)
rnorm.100$outmat #best models for 100 rows with normal errors
```

```
##           x1[, 1] x1[, 2] x1[, 3] x1[, 4] x1[, 5] x1[, 6] x1[, 7] x1[, 8]
## 1  ( 1 )  " "     " "     " "     "*"     " "     " "     " "     " "
## 2  ( 1 )  " "     "*"     " "     "*"     " "     " "     " "     " "
## 3  ( 1 )  "*"     "*"     " "     "*"     " "     " "     " "     " "
## 4  ( 1 )  "*"     "*"     " "     "*"     " "     " "     " "     " "
## 5  ( 1 )  "*"     "*"     "*"     "*"     " "     " "     " "     " "
## 6  ( 1 )  "*"     "*"     "*"     "*"     " "     " "     " "     " "
## 7  ( 1 )  "*"     "*"     "*"     "*"     " "     "*"     " "     " "
## 8  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     " "
## 9  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"
## 10  ( 1 ) "*"     "*"     "*"     "*"     "*"     "*"     "*"     "*"
##           x1[, 9] x1[, 10]
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     " "
## 4  ( 1 )  " "     "*"
## 5  ( 1 )  " "     "*"
## 6  ( 1 )  "*"     "*"
## 7  ( 1 )  "*"     "*"
## 8  ( 1 )  "*"     "*"
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
```

```
lap.100$outmat #best models for 50 rows with laplace errors
```

```
##           x1[, 1] x1[, 2] x1[, 3] x1[, 4] x1[, 5] x1[, 6] x1[, 7] x1[, 8]
## 1  ( 1 )  " "     " "     " "     "*"     " "     " "     " "     " "
## 2  ( 1 )  " "     "*"     " "     "*"     " "     " "     " "     " "
## 3  ( 1 )  "*"     "*"     " "     "*"     " "     " "     " "     " "
## 4  ( 1 )  "*"     "*"     " "     "*"     " "     " "     " "     " "
## 5  ( 1 )  "*"     "*"     "*"     "*"     " "     " "     " "     " "
## 6  ( 1 )  "*"     "*"     "*"     "*"     " "     " "     " "     " "
## 7  ( 1 )  "*"     "*"     "*"     "*"     " "     "*"     " "     " "
## 8  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     " "
## 9  ( 1 )  "*"     "*"     "*"     "*"     "*"     "*"     " "     "*"
## 10  ( 1 ) "*"     "*"     "*"     "*"     "*"     "*"     "*"     "*"
##           x1[, 9] x1[, 10]
## 1  ( 1 )  " "     " "
## 2  ( 1 )  " "     " "
## 3  ( 1 )  " "     " "
## 4  ( 1 )  " "     "*"
## 5  ( 1 )  " "     "*"
## 6  ( 1 )  "*"     "*"
## 7  ( 1 )  "*"     "*"
## 8  ( 1 )  "*"     "*"
## 9  ( 1 )  "*"     "*"
## 10  ( 1 ) "*"     "*"
```

```
reg.summary=summary(regfit_100.rnorm)
names(reg.summary)
```

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

```
reg.summary$rsq
```

```
##  [1] 0.03254699 0.04793377 0.07020747 0.07454334 0.07929605 0.08285983
##  [7] 0.08445853 0.08530858 0.08588081 0.08608687
```

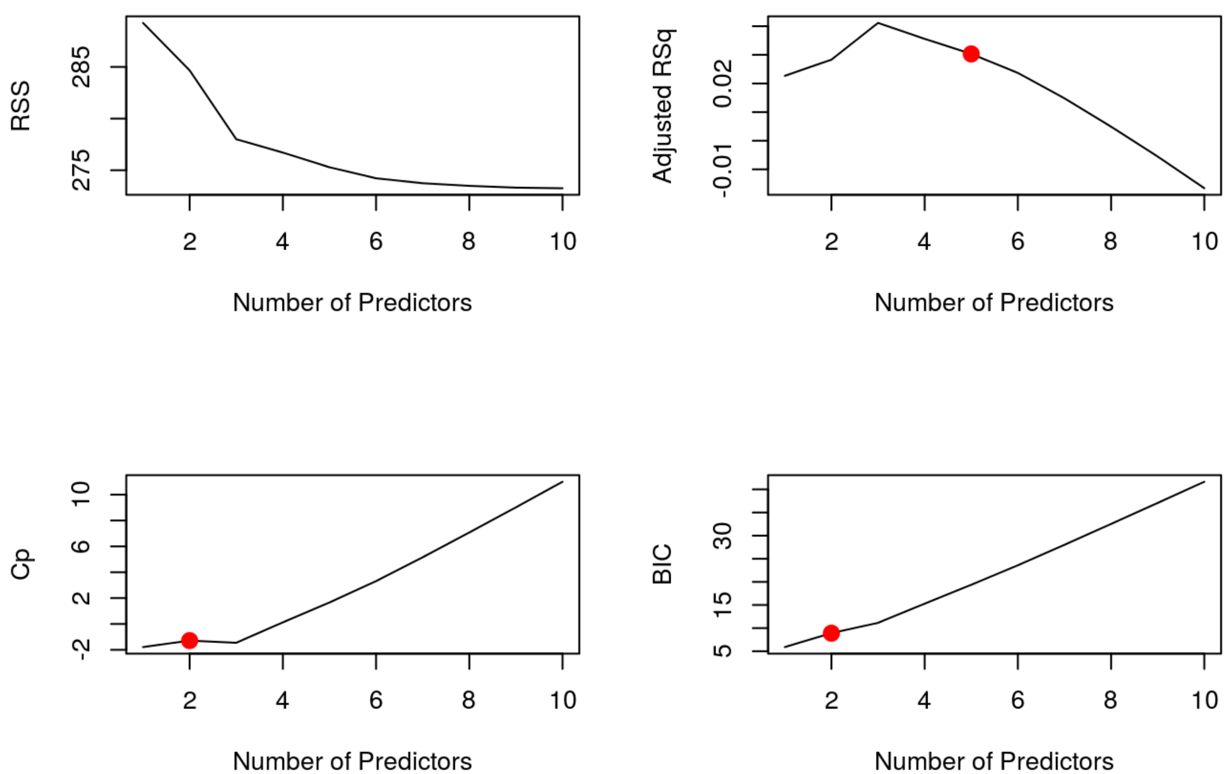
```
par(mfrow=c(2,2))
plot(reg.summary$rss,xlab="Number of Predictors",ylab="RSS",type="l")
plot(reg.summary$adjr2,xlab="Number of Predictors",ylab="Adjusted RSq",type="l"
)
max=which.max(reg.summary$adjr2)
points(5,reg.summary$adjr2[5], col="red",cex=2,pch=20)
plot(reg.summary$cp,xlab="Number of Predictors",ylab="Cp",type='l')
which.min(reg.summary$cp)
```

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

```
points(2,reg.summary$cp[2],col="red",cex=2,pch=20)
which.min(reg.summary$bic)
```

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

```
plot(reg.summary$bic,xlab="Number of Predictors",ylab="BIC",type='l')
points(2,reg.summary$bic[2],col="red",cex=2,pch=20)
```

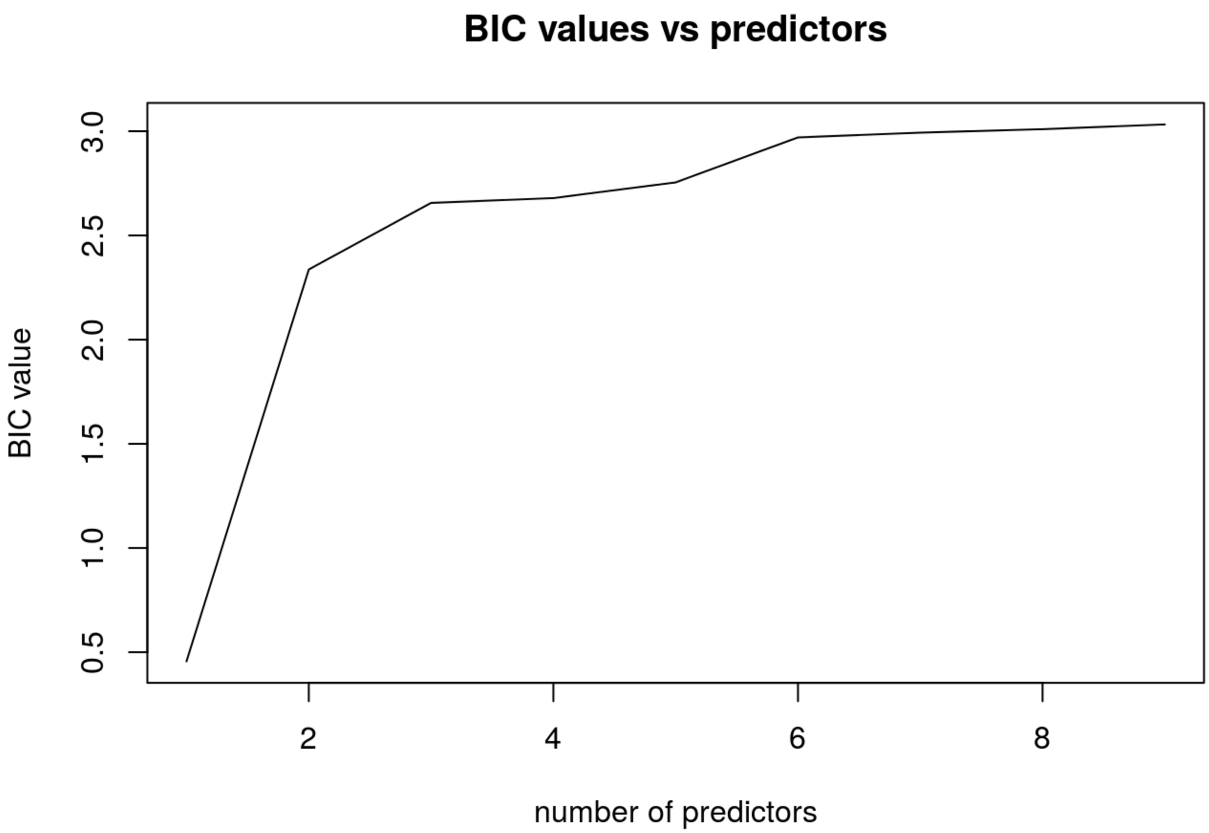



```
library(quantreg)
library(NormalLaplace)
library(leaps)
M=10;
N=c(25,50,100);
intercept=0.5;
beta=c(0.3,0.25,0.2,0.15,0.1,0,0,0,0,0)
```

some results for n=100 Least squares

```
L1bestsub(y2,x1,1)# laplace 100 rows k=1
```

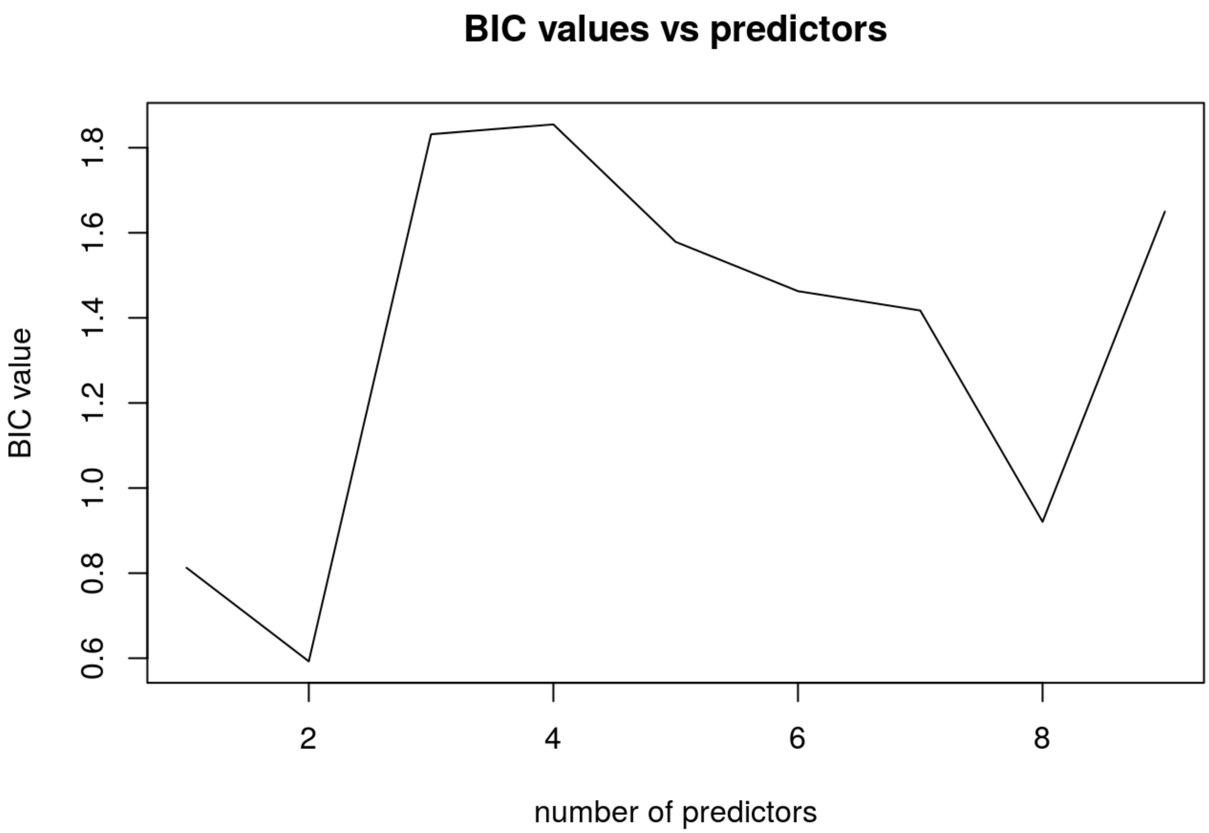
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.31639     -0.06949  0.63908
## X1          -0.04993     -0.37764  0.22612
## X2          -0.11270     -0.57625  0.07895
## X3           0.27017     -0.17306  0.86292
## X4           0.04748     -0.55346  0.57516
## X5          -0.09256     -0.32153  0.20146
## X6           0.26645     -0.47621  0.35564
## X7           0.10551     -0.51875  0.24838
## X8          -0.05399     -0.67355  0.24115
## X9          -0.30778     -0.50647  0.23159
## X10          0.01432     -0.34822  0.46948
## [1] "K1 values of BIC"
## [1] 0.4561071 2.3364910 2.6562152 2.6792410 2.7545018 2.9703004 2.9933263
## [8] 3.0097541 3.0327799
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 6
## [1] 2 8
## [1]  5  7 10
## [1] 3 5 6 9
## [1]  3  5  6  9 11
## [1]  2  3  4  8  9 10
## [1]  2  3  4  8  9 10 11
## [1]  2  3  4  5  6  7  8 10
## [1]  2  3  4  5  6  7  8 10 11
```

```
L1bestsub(y1,x1,1)# normal errors 100 rows k=1
```

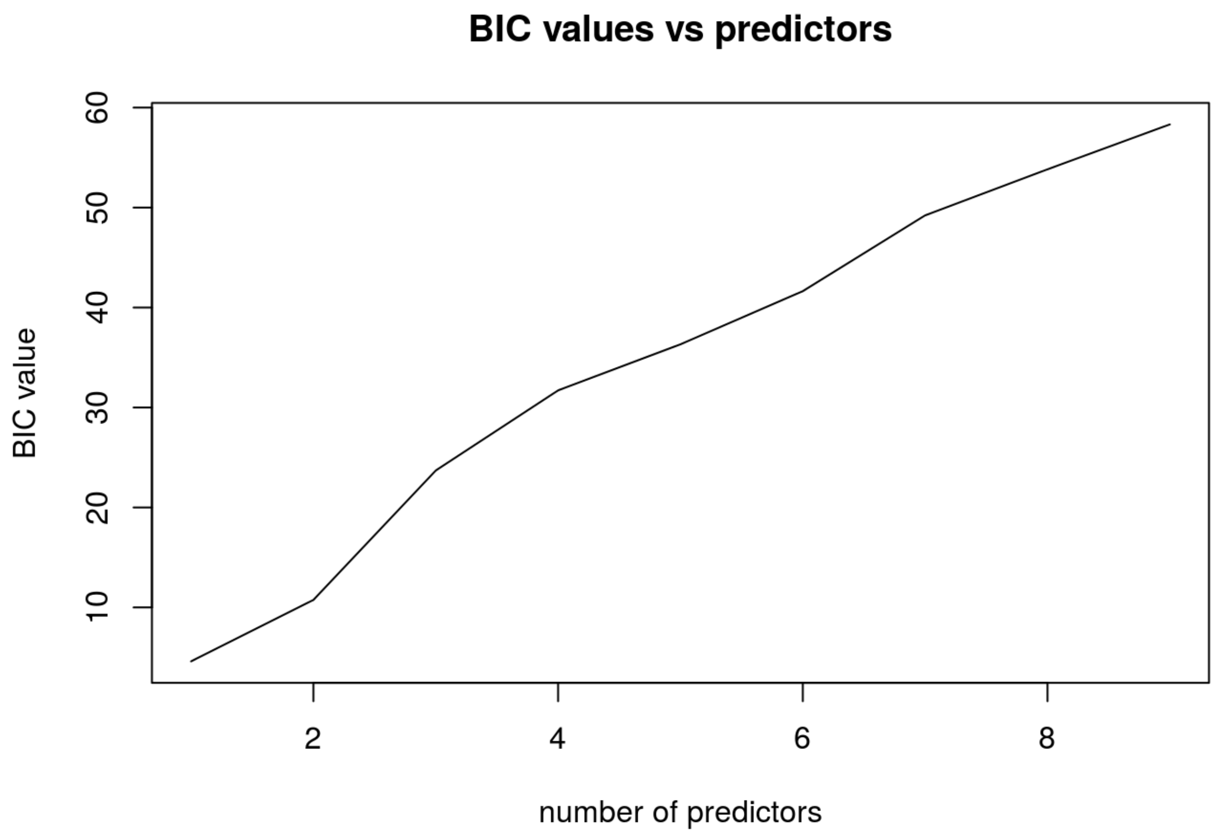
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.44185      0.22530  0.68819
## X1           0.02849     -0.38005  0.44551
## X2          -0.02142     -0.18124  0.16840
## X3           0.25763     -0.07749  0.45360
## X4          -0.18128     -0.30508  0.19499
## X5           0.09800     -0.07771  0.39960
## X6          -0.02428     -0.29336  0.23381
## X7           0.10880     -0.08762  0.37766
## X8          -0.18702     -0.23235  0.03483
## X9          -0.19597     -0.34442  0.03793
## X10         -0.10035     -0.22555  0.03297
## [1] "K1 values of BIC"
## [1] 0.8126844 0.5926724 1.8317070 1.8547328 1.5785792 1.4628350 1.4173714
## [8] 0.9207959 1.6498047
## [1] "Best performing BIC has predictors:"
## [1] 2
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 7
## [1]  4 10
## [1]  4 10 11
## [1] 2 3 5 8
## [1] 2 3 4 5 7
## [1]  2  3  4  5  7 11
## [1]  2  3  5  6  7  9 11
## [1]  2  3  4  5  6  7  8 11
## [1]  2  3  4  5  6  8  9 10 11
```

```
L1bestsub(y2,x1,2)# laplace 100 rows k=2
```

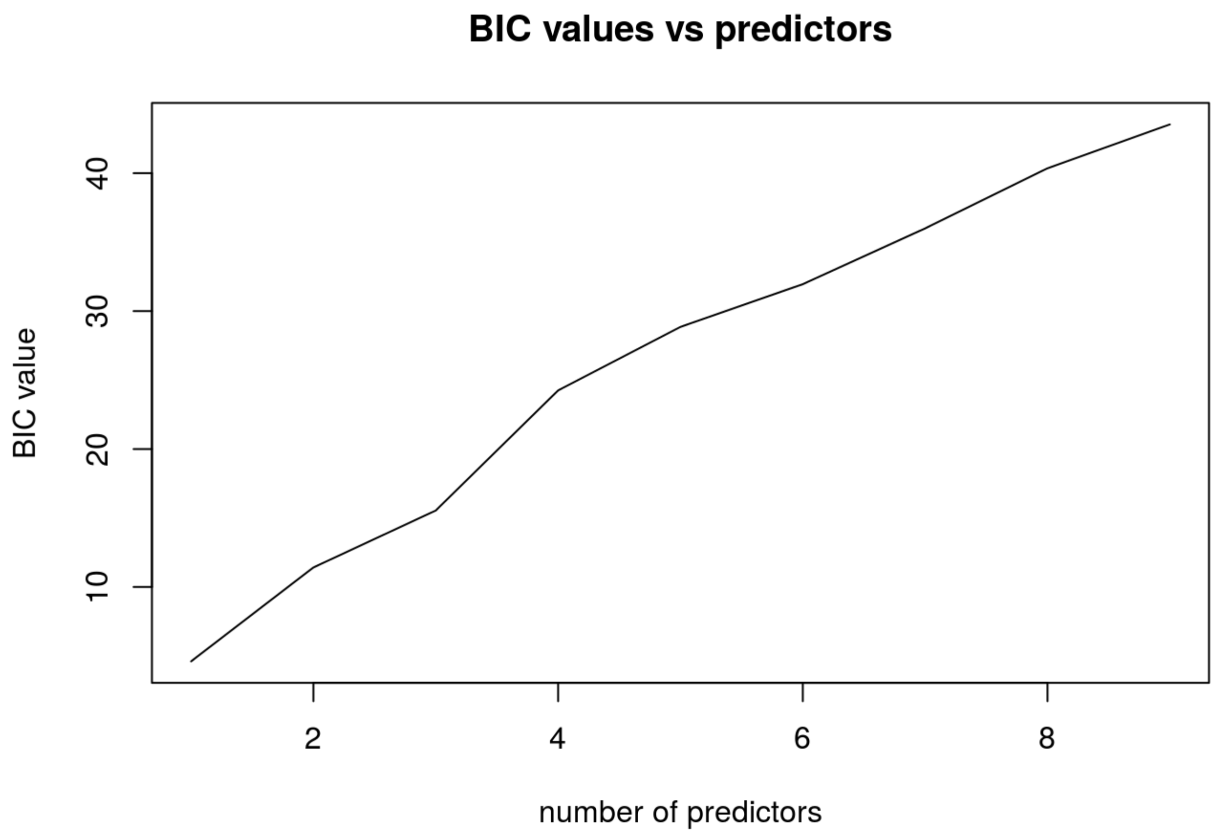
```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.31639     -0.06949  0.63908
## X1          -0.04993     -0.37764  0.22612
## X2          -0.11270     -0.57625  0.07895
## X3           0.27017     -0.17306  0.86292
## X4           0.04748     -0.55346  0.57516
## X5          -0.09256     -0.32153  0.20146
## X6           0.26645     -0.47621  0.35564
## X7           0.10551     -0.51875  0.24838
## X8          -0.05399     -0.67355  0.24115
## X9          -0.30778     -0.50647  0.23159
## X10          0.01432     -0.34822  0.46948
## [1] "K2 values of BIC"
## [1]  4.60517 10.75234 23.69479 31.71235 36.31752 41.63543 49.21804 53.82321
## [9] 58.31671
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 6
## [1] 2 8
## [1]  5  7 10
## [1] 3 5 6 9
## [1]  3  5  6  9 11
## [1]  2  3  4  8  9 10
## [1]  2  3  4  8  9 10 11
## [1]  2  3  4  5  6  7  8 10
## [1]  2  3  4  5  6  7  8 10 11
```

```
L1bestsub(y1,x1,2)# normal errors 100 rows k=2
```

```
## [1] "Coefficents and Confidence intervals obtained"
##
## Call: rq(formula = y1 ~ ., data = df1)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd upper bd
## (Intercept)  0.44185      0.22530  0.68819
## X1           0.02849     -0.38005  0.44551
## X2          -0.02142     -0.18124  0.16840
## X3           0.25763     -0.07749  0.45360
## X4          -0.18128     -0.30508  0.19499
## X5           0.09800     -0.07771  0.39960
## X6          -0.02428     -0.29336  0.23381
## X7           0.10880     -0.08762  0.37766
## X8          -0.18702     -0.23235  0.03483
## X9          -0.19597     -0.34442  0.03793
## X10         -0.10035     -0.22555  0.03297
## [1] "K2 values of BIC"
## [1]  4.60517 11.41298 15.54292 24.24842 28.85359 31.95186 35.99717 40.35338
## [9] 43.53533
## [1] "Best performing BIC has predictors:"
## [1] 1
```




```
## [1] "Best performing predictors for different subsets from p=1:9"
## [1] 7
## [1]  4 10
## [1]  4 10 11
## [1] 2 3 5 8
## [1] 2 3 4 5 7
## [1]  2  3  4  5  7 11
## [1]  2  3  5  6  7  9 11
## [1]  2  3  4  5  6  7  8 11
## [1]  2  3  4  5  6  8  9 10 11
```

Summary:

Attempted all 2^10 = 1024 possible combinations of explanatory variables to choose the best model.

Laplace distribution of errors were expected to show the Least Absolute Deviation and as expected they have lower BIC values. Thes best BIC model was the laplace distribution having 25 data points with a BIC value of 1.545038 for one predictor(This value will vary as the code is generated dynamically) .

K2 values of BIC increased almost every time with addition of predictors. That is almost always the first BIC having one predictor was the best performing.

Linear models will generally perform badly when p > n due to high variance resulting in poor prediction accuracy:

There was not much similarity in the the best predictors obtained between both the median regression techniques and least squares. The intercept played a key role in both the models.

The columns in x1 5 to 10 were insignificant in the least squares model but they had some small but significant coefficents in the median regression.

cross-validation could also be used to comapre the selected from the function to the regsubsets model for the same data.

The BIC criterion used here may have a finite variance and so it is possible to over-fit the selection criterion itself.