Problem 10.19

Homayra Alam A04985568

5/4/2020

```r
a<-(table.b12[,2])
b<- (table.b12[,3])
x1<-(cbind(a*b))

c<-(table.b12[,4])
d<-(table.b12[,5])
x2<-(cbind(c*d))

table.b12.new<-as.data.frame(cbind(table.b12,x1,x2))
table.b12.new
```

```
##    temp soaktime soakpct difftime diffpct pitch      x1     x2
## 1  1650     0.58    1.10     0.25    0.90 0.013  0.6380 0.2250
## 2  1650     0.66    1.10     0.33    0.90 0.016  0.7260 0.2970
## 3  1650     0.66    1.10     0.33    0.90 0.015  0.7260 0.2970
## 4  1650     0.66    1.10     0.33    0.95 0.016  0.7260 0.3135
## 5  1600     0.66    1.15     0.33    1.00 0.015  0.7590 0.3300
## 6  1600     0.66    1.15     0.33    1.00 0.016  0.7590 0.3300
## 7  1650     0.66    1.10     0.50    0.80 0.014  0.7260 0.4000
## 8  1650     1.00    1.10     0.58    0.80 0.021  1.1000 0.4640
## 9  1650     1.17    1.10     0.58    0.80 0.018  1.2870 0.4640
## 10 1650     1.17    1.10     0.58    0.80 0.019  1.2870 0.4640
## 11 1650     1.17    1.10     0.58    0.90 0.021  1.2870 0.5220
## 12 1650     1.17    1.10     0.58    0.90 0.019  1.2870 0.5220
## 13 1650     1.17    1.15     0.58    0.90 0.021  1.3455 0.5220
## 14 1650     1.20    1.15     1.10    0.80 0.025  1.3800 0.8800
## 15 1650     2.00    1.15     1.00    0.80 0.025  2.3000 0.8000
## 16 1650     2.00    1.10     1.10    0.80 0.026  2.2000 0.8800
## 17 1650     2.20    1.10     1.10    0.80 0.024  2.4200 0.8800
## 18 1650     2.20    1.10     1.10    0.80 0.025  2.4200 0.8800
## 19 1650     2.20    1.15     1.10    0.80 0.024  2.5300 0.8800
## 20 1650     2.20    1.10     1.10    0.90 0.025  2.4200 0.9900
## 21 1650     2.20    1.10     1.10    0.90 0.027  2.4200 0.9900
## 22 1650     2.20    1.10     1.50    0.90 0.026  2.4200 1.3500
## 23 1650     3.00    1.15     1.50    0.80 0.029  3.4500 1.2000
## 24 1650     3.00    1.10     1.50    0.70 0.030  3.3000 1.0500
## 25 1650     3.00    1.10     1.50    0.75 0.028  3.3000 1.1250
## 26 1650     3.00    1.15     1.66    0.85 0.032  3.4500 1.4110
## 27 1650     3.33    1.10     1.50    0.80 0.033  3.6630 1.2000
## 28 1700     4.00    1.10     1.50    0.70 0.039  4.4000 1.0500
## 29 1650     4.00    1.10     1.50    0.70 0.040  4.4000 1.0500
## 30 1650     4.00    1.15     1.50    0.85 0.035  4.6000 1.2750
## 31 1700    12.50    1.00     1.50    0.70 0.056 12.5000 1.0500
```

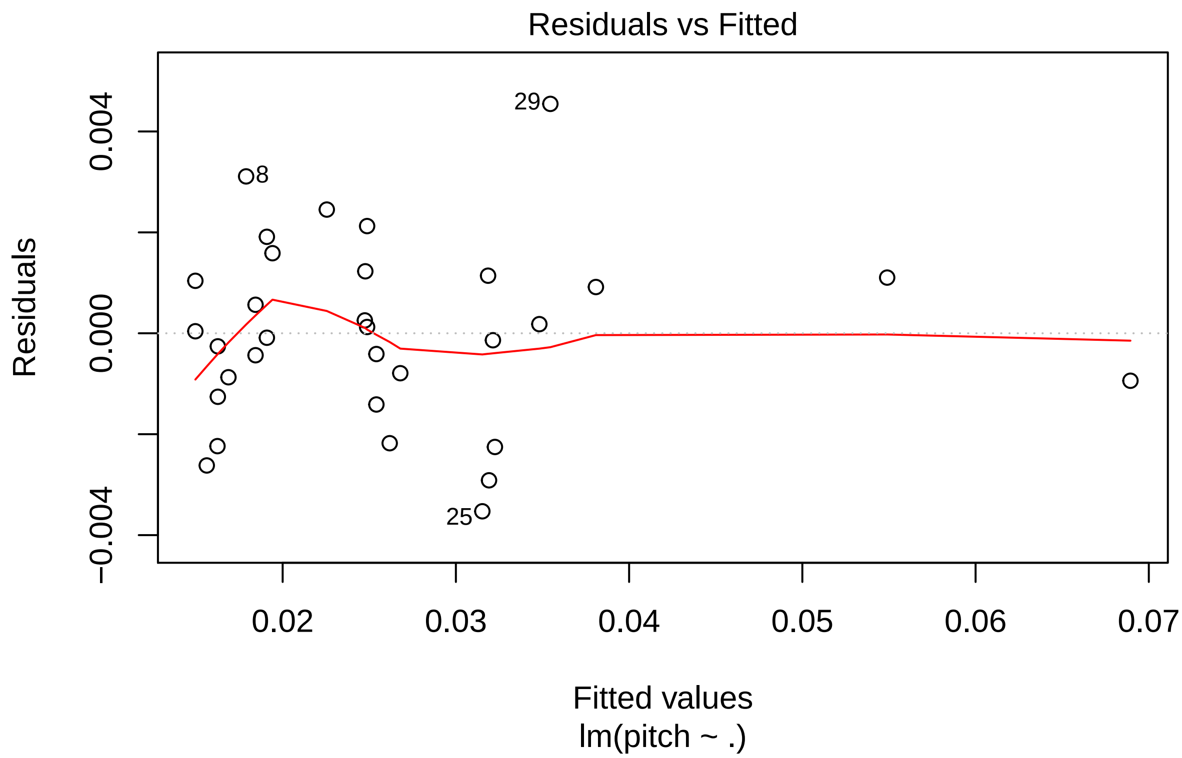
```
## 32 1700    18.50    1.00     1.50    0.70 0.068 18.5000 1.0500
```

Two new regressors, soaktime*soakpct and difftime* diffpct are added in the table. Our task is to analyze this model with all possible regressions and Cp criteria.

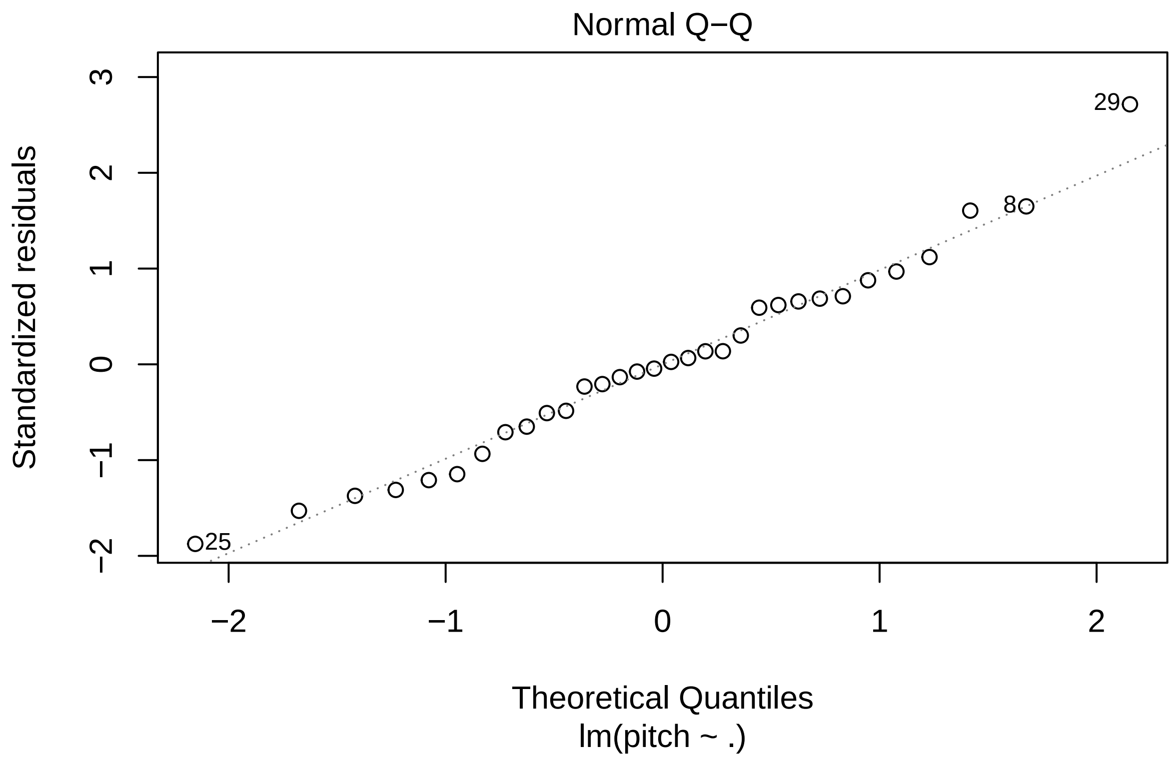
```r
dataset<- (table.b12.new)
sum.s = summary(model.full <- lm(pitch~., data = dataset))
sum.s
```

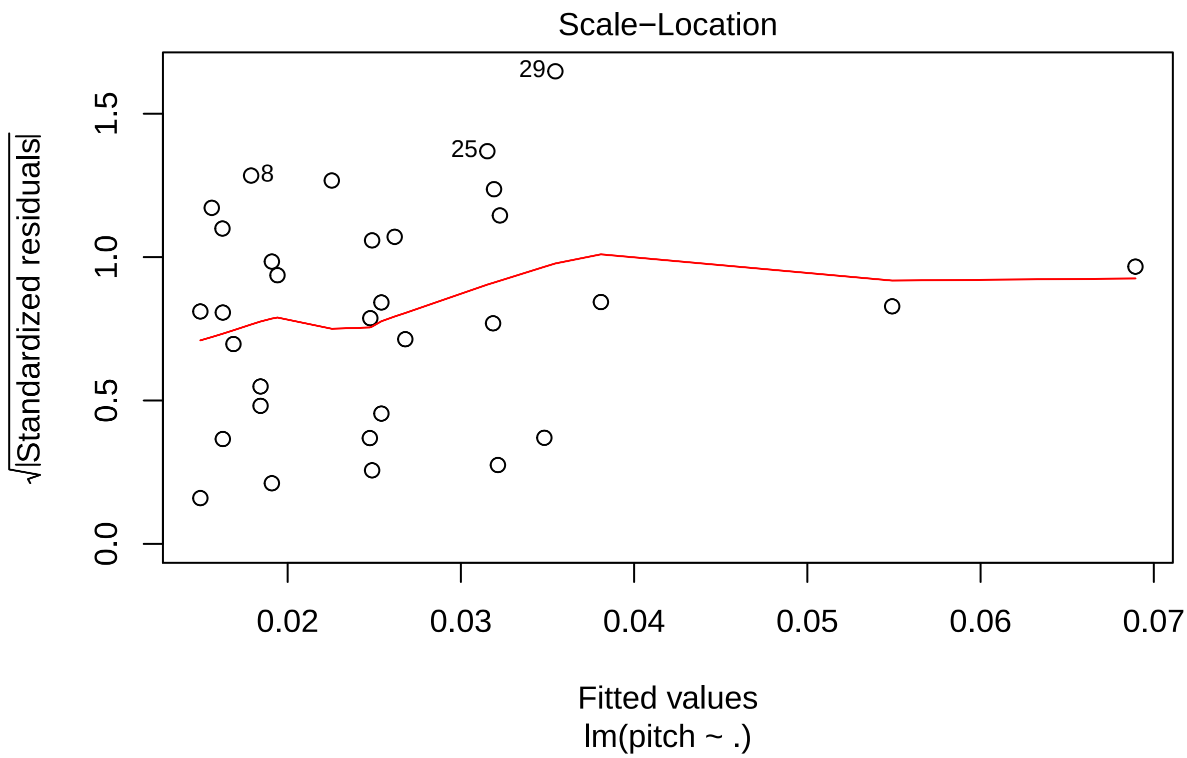
```
##
## Call:
## lm(formula = pitch ~ ., data = dataset)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0035276 -0.0010199 -0.0000239  0.0011133  0.0045464
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.809e-02  8.047e-02  -1.095   0.2845
## temp         5.258e-05  3.566e-05   1.475   0.1533
## soaktime    -6.258e-03  6.870e-03  -0.911   0.3713
## soakpct     -3.570e-03  2.526e-02  -0.141   0.8888
## difftime     2.537e-02  1.046e-02   2.426   0.0232 *
## diffpct      1.980e-02  1.398e-02   1.417   0.1695
## x1           8.599e-03  6.930e-03   1.241   0.2266
## x2          -2.287e-02  1.154e-02  -1.982   0.0591 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002065 on 24 degrees of freedom
## Multiple R-squared:  0.9763, Adjusted R-squared:  0.9694
## F-statistic: 141.5 on 7 and 24 DF,  p-value: < 2.2e-16
```

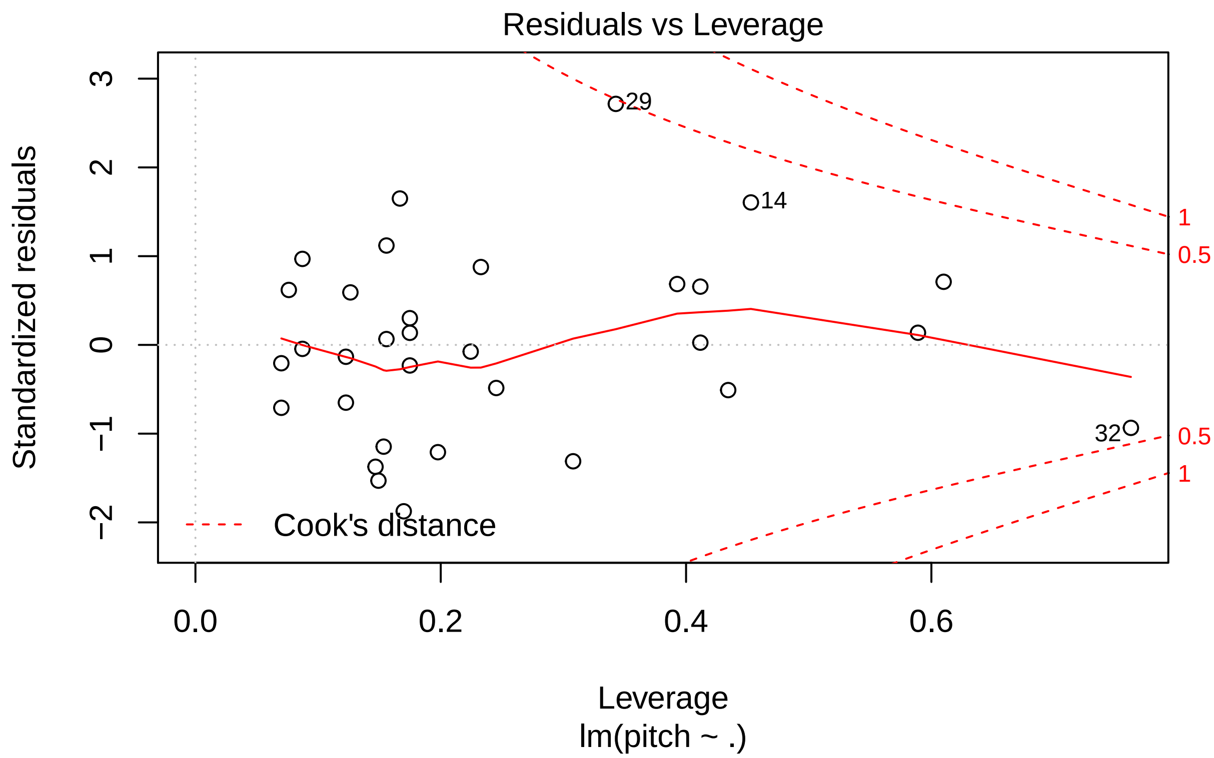
```r
plot(model.full)
```



Residuals vs Fitted
Residuals
0.004
0.000
-0.004
29
8
25
0.02
0.03
0.04
0.05
0.06
0.07
Fitted values
lm(pitch ~ .)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(pitch ~ .)
29
8
25
−2
−1
0
1
2
3


Scale−Location
29
25
8
1.5
1.0
0.5
0.0
√|Standardized residuals|
0.02
0.03
0.04
0.05
0.06
0.07
Fitted values
lm(pitch ~ .)



Question :: Find an appropriate subset regression model for these data using all possible regressions and the Cp criterion.

Selection Methods Table.B14

```
dataset = (table.b12.new)
model.full =lm(pitch~., data = dataset)
pred.names = names(coef(model.full))
result.full <- c(coef(model.full), adjr2 = summary(model.full)$adj.r.squared)
```

Backward Selection

```
library(olsrr)
```

```
## Warning: package 'olsrr' was built under R version 3.6.3
```

```
model.backward = olsrr::ols_step_backward_p(model.full, prem = .10)
step.b = model.backward$steps
tmp = c(0, coef(model.backward$model))
tmp
```

```
##                 (Intercept)     difftime           x1           x2
##  0.000000000  0.011691312  0.016334627  0.002405708 -0.010790924
```

```
beta.backward = tmp[c(2,1,1,1,3,1,4,5)]
names(beta.backward) <- pred.names
beta.backward
```

```
## (Intercept)        temp    soaktime     soakpct    difftime     diffpct
## 0.011691312 0.000000000 0.000000000 0.000000000 0.016334627 0.000000000
##          x1          x2
## 0.002405708 -0.010790924
```

```
adj.r2 = model.backward$adjr[step.b]
result.backward = c(beta.backward, adjr2 =adj.r2)
```

For the backward model,difftime, x1 & x2 are the most significant regressors.

Forward Selection

```
model.forward = olsrr::ols_step_backward_p(model.full)
step.f = model.forward$steps
tmp = c(0, coef(model.forward$model))
tmp
```

```
##                (Intercept)          temp      soaktime      difftime
##  0.000000e+00 -9.566876e-02  5.476591e-05 -5.577312e-03  2.561708e-02
##       diffpct            x1            x2
##  1.984261e-02  7.923475e-03 -2.303599e-02
```

```
beta.forward = tmp[c(2,3,4,1,5,6,7,8)]
names(beta.forward) <- pred.names
beta.forward
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime
## -9.566876e-02  5.476591e-05 -5.577312e-03  0.000000e+00  2.561708e-02
##       diffpct            x1            x2
##  1.984261e-02  7.923475e-03 -2.303599e-02
```

```
adj.r2 = summary(model.full)$adj.r.squared
result.forward = c(beta.forward, adjr2 =adj.r2)
```

For the forward model,apart from soakpct all are the significant regressors.

Stepwise Selection with both forward and backward steps using p-value

```
model.step = olsrr::ols_step_both_p(model.full )
step.step = model.forward$steps
tmp = c(0, coef(model.step$model))
tmp
```

```
##              (Intercept)          x1    difftime           x2
## 0.000000000 0.011691312 0.002405708 0.016334627 -0.010790924
```

```
beta.step = tmp[c(2,1,1,1,4,1,3,5)]
names(beta.step) <- pred.names
beta.step
```

```
## (Intercept)        temp    soaktime     soakpct    difftime     diffpct
## 0.011691312 0.000000000 0.000000000 0.000000000 0.016334627 0.000000000
##          x1          x2
## 0.002405708 -0.010790924
```

```
adj.r2 = summary(model.full)$adj.r.squared
result.step = c(beta.step, adjr2 =adj.r2)
```

```
result.step
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime       diffpct
##   0.011691312  0.000000000  0.000000000  0.000000000  0.016334627  0.000000000
##            x1            x2         adjr2
##   0.002405708 -0.010790924  0.969445805
```

step-wise model has selected difftime, x1 & x2 as significant predictors.

All subsets with leaps package

```
library(leaps)
```

BIC

```
## Warning: package 'leaps' was built under R version 3.6.3
```

```
dataset = table.b12.new

models.leaps.summ = summary(models.leaps <-regsubsets(pitch~., data = dataset,  nbest = 1 ) )
num.models = length(models.leaps.summ$bic)
best.bic = (1:num.models)[models.leaps.summ$bic == min(models.leaps.summ$bic)]
tmp = c(0, coef(models.leaps, best.bic))
tmp
```

```
##                (Intercept)     difftime            x1            x2
##   0.000000000  0.011691312  0.016334627  0.002405708 -0.010790924
```

```
beta.bic = tmp[c(2,1,1,1,3,1,4,5)]
names(beta.bic) <- pred.names
beta.bic
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime       diffpct
##   0.011691312  0.000000000  0.000000000  0.000000000  0.016334627  0.000000000
##            x1            x2
##   0.002405708 -0.010790924
```

```
adj.r2 = models.leaps.summ$adjr2[best.bic]
result.bic = c(beta.bic, adjr2 =adj.r2)
result.bic
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime       diffpct
##   0.011691312  0.000000000  0.000000000  0.000000000  0.016334627  0.000000000
##            x1            x2         adjr2
##   0.002405708 -0.010790924  0.968715225
```

BIC is taking difftime, x1 and x2 as the sigficant regressors just like step wise model.

```
best.adjr2 = (1:num.models)[models.leaps.summ$adjr2 == max(models.leaps.summ$adjr2)]
tmp = c(0, coef(models.leaps, best.adjr2))
tmp
```

Adjusted R-squared

```
##                  (Intercept)          temp      soaktime      difftime
##   0.000000e+00 -9.566876e-02  5.476591e-05 -5.577312e-03  2.561708e-02
```

```
##       diffpct            x1            x2
##  1.984261e-02  7.923475e-03 -2.303599e-02
```

```
beta.adjr2 = tmp[c(2,3,4,1,5,6,7,8)]
names(beta.adjr2) <- pred.names
beta.adjr2
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime
## -9.566876e-02  5.476591e-05 -5.577312e-03  0.000000e+00  2.561708e-02
##       diffpct            x1            x2
##  1.984261e-02  7.923475e-03 -2.303599e-02
```

```
adj.r2 = summary(model.full)$adj.r.squared
result.adjr2 = c(beta.adjr2, adjr2 =adj.r2)
result.adjr2
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime
## -9.566876e-02  5.476591e-05 -5.577312e-03  0.000000e+00  2.561708e-02
##       diffpct            x1            x2         adjr2
##  1.984261e-02  7.923475e-03 -2.303599e-02  9.694458e-01
```

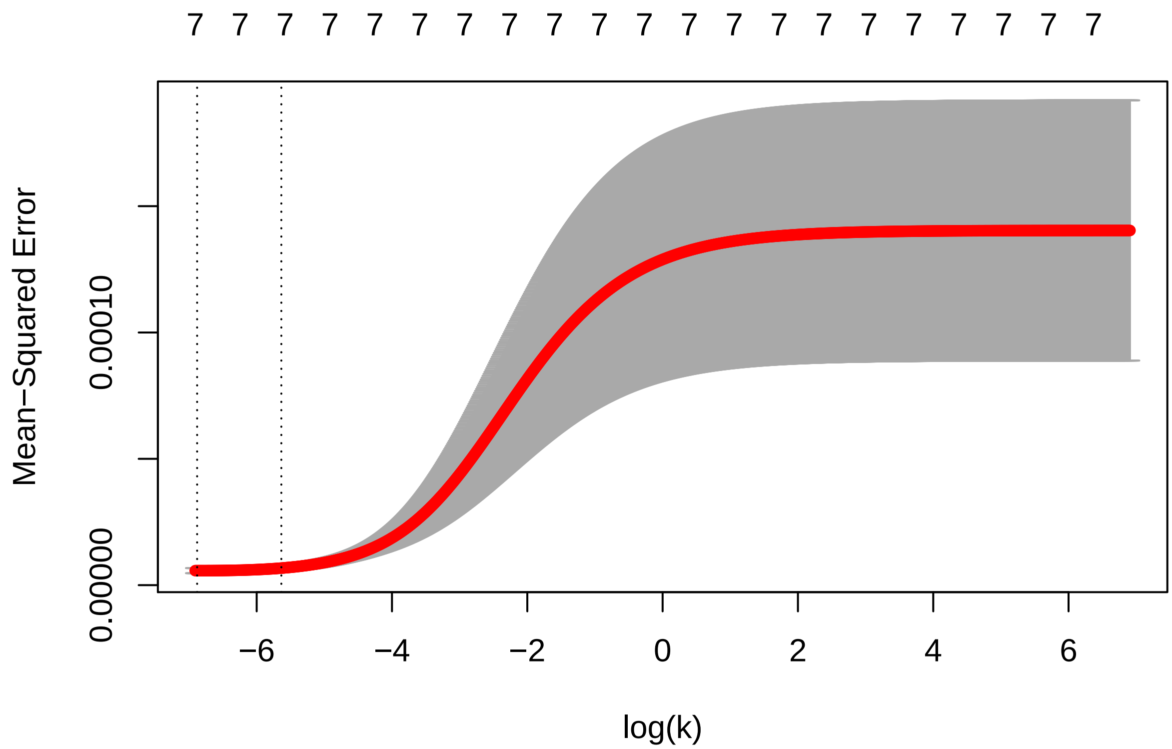
Adjusted R square model is eliminating only soackpct and trending the model's adjusted r square to clsoer to one which makes it more fit.

Ridge Regression

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.6.3

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 3.0-2
```

```
X = model.matrix(model.full)
ks = 10^seq(-3,3, length.out = 1000)
ridge.cv = glmnet::cv.glmnet(X, dataset$pitch, alpha = 0, lambda = ks, nfolds = 5, standardize = TRUE )
plot(ridge.cv, xlab = "log(k)")
```




```
beta.ridge = coef(ridge.cv, s = "lambda.min")[-2]
names(beta.ridge) <- pred.names
beta.ridge
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime
## -5.029881e-02  4.193432e-05  1.101963e-03  6.501591e-04  4.632375e-03
##       diffpct            x1            x2
## -8.402698e-03  1.143395e-03  3.367596e-03
```

The mean squared error gets stablized at log(k)=2 or k=e^2.

```
X <- model.matrix(model.full)
p.ridge = length(beta.ridge)
y = dataset$pitch
yhat.ridge = X%*%beta.ridge
n = length(y)
adjr2 = 1- (sum(((y) - yhat.ridge)^2)/ (n-p.ridge))/var((y))
result.ridge= c(beta.ridge, adjr2 =adj.r2)
result.ridge
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime
## -5.029881e-02  4.193432e-05  1.101963e-03  6.501591e-04  4.632375e-03
##       diffpct            x1            x2         adjr2
## -8.402698e-03  1.143395e-03  3.367596e-03  9.694458e-01
```

We can observe that, ridge regression has made all the regressors less than zero.So, with the optimized value of k , this model will face less penalty for having smaller beta's.

PCA

```
PCA = princomp(X[, -1], cor = TRUE, scores = TRUE)

eigen.values = (PCA$sdev)^2

eigen.values
```

```
##       Comp.1       Comp.2       Comp.3       Comp.4       Comp.5       Comp.6
## 4.4817628755 1.4707999143 0.5834464214 0.2550280204 0.2054503843 0.0034004307
##       Comp.7
## 0.0001119534
```

```
factors <- PCA$scores[, eigen.values>1]
factors
```

```
##        Comp.1     Comp.2
## 1  -1.8854636  1.6494503
## 2  -1.7380670  1.4531522
## 3  -1.7380670  1.4531522
## 4  -1.9501831  1.4912530
## 5  -3.6142184  0.3214797
## 6  -3.6142184  0.3214797
## 7  -1.0591030  0.9675448
## 8  -0.8529899  0.8153188
## 9  -0.8098703  0.8355366
## 10 -0.8098703  0.8355366
## 11 -1.2115052  0.8706417
## 12 -1.2115052  0.8706417
## 13 -1.6579450  0.1437929
## 14 -0.4693383 -1.1399648
## 15 -0.4114033 -0.8018858
## 16  0.1798352 -0.3180528
## 17  0.2305642 -0.2942673
## 18  0.2305642 -0.2942673
## 19 -0.2096166 -1.0183996
## 20 -0.1240687 -0.3446431
## 21 -0.1240687 -0.3446431
## 22  0.5114561 -1.3737050
## 23  0.5975300 -1.8844547
## 24  1.3513269 -1.0463018
## 25  1.1920882 -1.1043671
## 26  0.6852701 -2.3409938
## 27  1.1165523 -1.1231861
## 28  2.6073522 -0.4476937
## 29  1.6049718 -0.9273740
## 30  0.6980129 -1.8209547
## 31  5.5185147  1.9571279
## 32  6.9674633  2.6390466
```

Now we run the model with selected factors

```r
PCR.data = data.frame(y, factors)
model.PCA.full <- lm(y~., data = PCR.data)
model.PCA = olsrr::ols_step_backward_p(model.PCA.full, prem =.10)
coef(model.PCA$model)
```

```
## (Intercept)      Comp.1
## 0.026281250 0.005238696
```

Now we relate to the original variables

```r
eigen.vectors = PCA$loadings[,c(1,2)]
beta.PCReg = c(coef(model.PCA$model)[1],eigen.vectors%*%coef(model.PCA$model)[c(2,3)] )
names(beta.PCReg) <- pred.names
beta.PCReg
```

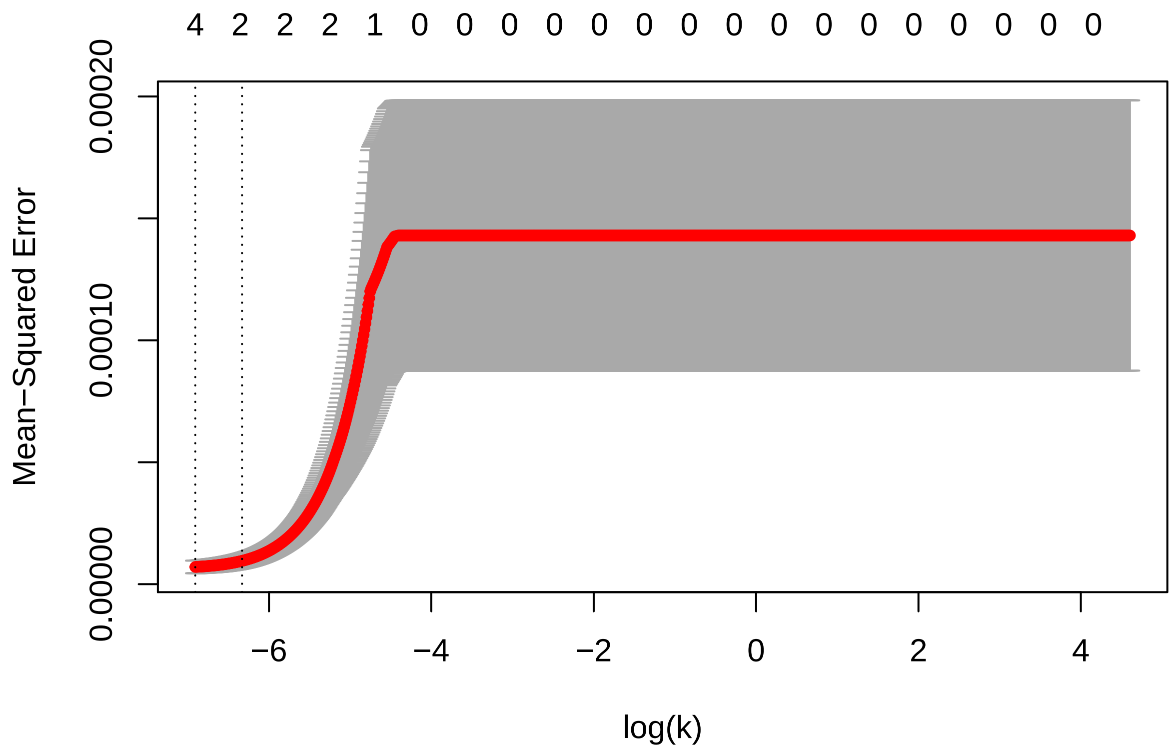
```
## (Intercept)        temp    soaktime     soakpct    difftime     diffpct
##  0.02628125          NA          NA          NA          NA          NA
##          x1          x2
##          NA          NA
```

```r
result.PCReg = c(beta.PCReg, adjr2 =summary(model.PCA.full)$adj.r.squared)
result.PCReg
```

```
## (Intercept)        temp    soaktime     soakpct    difftime     diffpct
##  0.02628125          NA          NA          NA          NA          NA
##          x1          x2       adjr2
##          NA          NA  0.90381706
```

Lasso Regression

```r
model.full = lm(pitch~., data = dataset)

X = model.matrix(model.full)
ks = 10^seq(-3,2, length.out = 1000)

lasso.cv = glmnet::cv.glmnet(X, dataset$pitch, alpha = 1, lambda = ks, nfolds = 5, standardize = TRUE )
plot(lasso.cv, xlab = "log(k)")
```




```r
coef(lasso.cv, s = "lambda.min")
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -6.280515e-03
## (Intercept)  .
## temp         1.213004e-05
## soaktime     .
## soakpct      .
## difftime     6.409648e-03
## diffpct     -6.617839e-04
## x1           2.304213e-03
## x2           .
```

```r
tmp = c(0, coef(lasso.cv, s = "lambda.min")@x)
tmp
```

```
## [1]  0.000000e+00 -6.280515e-03  1.213004e-05  6.409648e-03 -6.617839e-04
## [6]  2.304213e-03
```

```r
beta.lasso = tmp[c(2,3,1,1,4,1,5,6)]
names(beta.lasso) <- pred.names
beta.lasso
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime
## -6.280515e-03  1.213004e-05  0.000000e+00  0.000000e+00  6.409648e-03
##       diffpct            x1            x2
##  0.000000e+00 -6.617839e-04  2.304213e-03
```

```r
X <- model.matrix(model.full)
p.lasso = 6
y = dataset$pitch
yhat.lasso = X%*%beta.lasso
n = length(y)
adjr2 = 1- (sum((y - yhat.lasso)^2)/ (n-p.lasso))/var(y)
result.lasso= c(beta.lasso, adjr2=adj.r2)
result.lasso
```

```
##   (Intercept)          temp      soaktime       soakpct      difftime 
## -6.280515e-03  1.213004e-05  0.000000e+00  0.000000e+00  6.409648e-03 
##       diffpct            x1            x2         adjr2 
##  0.000000e+00 -6.617839e-04  2.304213e-03  9.694458e-01
```

Compare the Results

```r
results.b2 = data.frame(result.full,result.backward, result.forward, result.step, result.bic, result.adj
round(results.b2,3)
```

```
##             result.full result.backward result.forward result.step result.bic
## (Intercept)      -0.088           0.012         -0.096       0.012      0.012
## temp              0.000           0.000          0.000       0.000      0.000
## soaktime         -0.006           0.000         -0.006       0.000      0.000
## soakpct          -0.004           0.000          0.000       0.000      0.000
## difftime          0.025           0.016          0.026       0.016      0.016
## diffpct           0.020           0.000          0.020       0.000      0.000
## x1                0.009           0.002          0.008       0.002      0.002
## x2               -0.023          -0.011         -0.023      -0.011     -0.011
## adjr2             0.969           0.969          0.969       0.969      0.969
##             result.adjr2 result.ridge result.PCReg result.lasso
## (Intercept)       -0.096       -0.050        0.026       -0.006
## temp               0.000        0.000           NA        0.000
## soaktime          -0.006        0.001           NA        0.000
## soakpct            0.000        0.001           NA        0.000
## difftime           0.026        0.005           NA        0.006
## diffpct            0.020       -0.008           NA        0.000
## x1                 0.008        0.001           NA       -0.001
## x2                -0.023        0.003           NA        0.002
## adjr2              0.969        0.969        0.904        0.969
```

From the above comparison we can observe that, almost all the models shows the adjusted r square value as like as the full model.Most of the models marked three to five regressors as significant.ridge regression,forward and adjuster r square procedures selects highest number of significant regressors.It can also be mentioned that the new regressors x1 and x2 is considered as significant for most of the models.For the final model selection , we go for the highest adjusted r square with lowest numberof regressors.In perspective of selecting number of regressors , we choose backward, step,bic and in the sense of optimizing MSE we choose lasso.