Bonus1.R

```
# Below is the function for clusterwise regression:
# Data: i have used the Auto data from the ISLR package and car.test.frame fr
om rpart
# K= is the no of clusters
# t is no of iterations
# q is the seed for ramdom sampling.


clustreg = function(data,k,t,q) {

  rsq = c()       # array of R-Squared values
  data0 = data[,-1]     # data set without response variable
  n = dim(data)[1]   # number of observations
  p = dim(data)[2]   # number of columns
  ybar = mean(data[,1])
  xnames = names(data)

 # Regression formula for the multi regression

  formula = paste(xnames[1],'~.')

  # Initializing variables
  # initial size of each cluster
  # remainder
  # set seed for reproducible results
  # randomized indices

  s = floor(n/k)
  r = n%%k
  set.seed(q)
  ss = sample(1:n,n,replace=FALSE)


  # Setting up initial cluster indices
  # Initial clusters each have n/k observations from the random sample array
  # The last cluster starts with n/k+r observations if n/k is not an integer
  c = list()
  for(i in 1:k) {
    if(i==k) {
      c[[i]] = ss[((i-1)*s+1):(n)]
    } else {
      c[[i]] = ss[((i-1)*s+1):(i*s)]
    }
  }
```

```
  # Setting up initial clusters
  cc = list()         # data in k clusters
  ccc = list()        # data in k clusters without response variable
  for(i in 1:k) {
    cc[[i]] = data[c[[i]],]
    ccc[[i]] = data0[c[[i]],]
  }

  for(ii in 1:t) {    # runs loop t times

    # Storing multiple regression models in a list
    fit = list()
    for(i in 1:k) {
      fit[[i]]=lm(formula,data=cc[[i]])
    }

    # Calculating Squared Residual Error (SRE) for all observations
    pred = matrix(data = NA, n, k)
    res = matrix(data = NA, n, k)
    actual = matrix(data = NA, n, k)
    sre = matrix(data = NA, n, k)
    for(i in 1:k) {
      pred[,i] = predict(fit[[i]], data0)
      actual[,i] = data[,1]
      res[,i] = actual[,i]-pred[,i]
      sre[,i] = res[,i]^2
    }

    # Reclassifying observations to the cluster with the minimum SRE
    # Computing overall R-Squared value for all cluster regression models com
bined
    a = c()
    rsqnum = c()
    rsqden = c()
    for(i in 1:n) {
      for(j in 1:k) {
        if(min(sre[i,])==sre[i,j]) {
          a[i] = j
          rsqnum[i] = (pred[i,j]-ybar)^2
          rsqden[i] = (actual[i,j]-ybar)^2
        }
      }
    }
    rsq[ii] = sum(rsqnum)/sum(rsqden)

    # Reforming new clusters for next iteration
    # Refitting k linear regression models
    c = list()
    fit = list()
    crsq = c()
```

```
    for(i in 1:k) {
      c[[i]] = which(a==i)
      cc[[i]] = data[c[[i]],]
      ccc[[i]] = data0[c[[i]],]
      fit[[i]] = lm(formula,data=cc[[i]])
      crsq[i] = summary(fit[[i]])$r.squared
    }

    # If the overall R-Squared value is the same as in last iteration,
    # the procedure has converged and we will break out of the loop
    if(ii>1) {
      if(rsq[ii]==rsq[ii-1]) break
    }

    # End of iteration loop
  }

  #output = list()
  #output[[1]] = fit  # return final regression models for each cluster
  #output[[2]] = rsq  # return overall R-Squared value for each iteration (co
mbined from all clusters)
  #output[[3]] = crsq # return R-Squared value for the final regression model
for each cluster
  #output[[4]] = c    # return cluster list of observation assignments
  #output[[5]] = a    # return observation list of cluster classifications
  #output[[6]] = ii   # total number of iterations
  output = list("fit" = fit,
                "rsq" = rsq,
                "crsq" = crsq,
                "clusters" = c,
                "obs" = a,
                "i" = ii)
  return(output)
}
```

applying the function on the data####

```
#first i have used the Auto data to run the clusterwise regression function
# The coloumns other than numeric datatype have been excluded.
# Some EDA with the data to identify if any values are missing


library(ISLR)
data("Auto")
data<- Auto
data<- Auto[-9]
head(data)
```

```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1
```

```
#Now use the clusterwise function
data<-data
k=15
t=20
q=1

#call the function
clustreg(data,k,t,q)
```

```
## $fit
## $fit[[1]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##     -5.62948      -1.68931       0.06997      -0.04775      -0.00806
## acceleration          year        origin
##      0.03465       0.67816       0.70492
##
##
## $fit[[2]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
```

```
##  (Intercept)     cylinders  displacement    horsepower        weight
##    48.980603     -3.001665      0.063590     -0.136427     -0.005322
## acceleration          year        origin
##    -0.485898      0.197480      0.155381
##
##
## $fit[[3]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##   -62.325743      1.404285     -0.067138      0.014919     -0.002588
## acceleration          year        origin
##     1.409347      0.992764     -0.613866
##
##
## $fit[[4]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##     6.235388     -3.214100      0.041894     -0.104663     -0.001775
## acceleration          year        origin
##    -0.724680      0.720996      0.156002
##
##
## $fit[[5]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##   -12.010980      0.667435      0.003518     -0.015816     -0.005085
## acceleration          year        origin
##     0.165027      0.504835      4.853250
##
##
## $fit[[6]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##    -0.247843     -2.383661     -0.010686      0.013181     -0.002302
```

```
## acceleration          year        origin
##    -1.283951      0.807206      2.416674
##
##
## $fit[[7]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##   -5.051e+01    -2.968e+00    -1.409e-02    -5.519e-02     7.291e-04
## acceleration          year        origin
##   -1.240e+00     1.522e+00     1.993e+00
##
##
## $fit[[8]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##    24.984850     -1.963878      0.001182      0.057468     -0.006435
## acceleration          year        origin
##    -0.150869      0.281707      1.545351
##
##
## $fit[[9]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##   -21.950096     -1.959454      0.065000     -0.038553     -0.006765
## acceleration          year        origin
##     0.544268      0.716265      2.821344
##
##
## $fit[[10]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##   -40.914020      2.905032     -0.018397     -0.012894     -0.008251
## acceleration          year        origin
##     0.110029      1.033664     -2.162768
```

```
##
##
## $fit[[11]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##   -36.982196     -0.292734      0.002611      0.015975     -0.006441
## acceleration          year        origin
##     0.219340      0.969186      1.269689
##
##
## $fit[[12]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##    18.427165      2.574940     -0.065781     -0.216764      0.003386
## acceleration          year        origin
##    -1.359853      0.461757      0.427428
##
##
## $fit[[13]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##    -4.204525     -1.296037      0.023998      0.017505     -0.005403
## acceleration          year        origin
##     0.418065      0.448533      2.179329
##
##
## $fit[[14]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##    -11.56701       3.79587       0.04170      -0.19139      -0.01495
## acceleration          year        origin
##      0.08392       0.84963       3.37225
##
##
```

```
## $fit[[15]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
##  (Intercept)     cylinders  displacement    horsepower        weight
##    -50.72594      -1.66046       0.09913      -0.03404      -0.01133
## acceleration          year        origin
##      0.56939       1.21006       0.96662
##
##
##
## $rsq
##  [1] 0.9147780 0.9571820 0.9936954 0.9938871 0.9937385 0.9967324 0.9960643
##  [8] 0.9992900 0.9982274 0.9980148 0.9988928 1.0000771 0.9987599 0.9981354
## [15] 0.9984115 0.9995218 0.9980909 0.9979176 0.9983132 0.9982633
##
## $crsq
##  [1] 0.9995567 0.9980371 0.9996063 0.9993743 0.9994622 0.9972099 0.9988917
##  [8] 0.9985064 0.9993270 0.9937941 0.9996128 0.9989190 0.9988181 0.9914367
## [15] 0.9997725
##
## $clusters
## $clusters[[1]]
##  [1]   4   6  40  55  57  62  73  76  90 104 120 125 164 170 172 183 190
## [18] 202 204 206 238 240 266 290 343 360 364 387 391
##
## $clusters[[2]]
##  [1]  16  30  31  83  96 114 149 166 194 197 209 214 216 226 252 255 263
## [18] 265 273 274 286 289 304 312 372
##
## $clusters[[3]]
##  [1]  27  51  74 100 107 118 123 138 144 165 207 217 267 285 325 326 342
## [18] 346 357
##
## $clusters[[4]]
##  [1]  14  15  25  39  42  43  44  58  59  85  88  89  94  95 106 137 192
## [18] 249 307 318 334 347 349 352 354 358 377 386
##
## $clusters[[5]]
##  [1]  13  18  28  41  54  67 103 135 136 181 187 200 228 230 257 270 279
## [18] 281 292 299 309 311 331 345 369 381
##
## $clusters[[6]]
##  [1]   5  26  38  48  60  69  70  72 105 109 126 127 151 154 158 163 168
## [18] 169 184 201 219 258 264 282 284 300 302 310 315 316 319 329 333
##
## $clusters[[7]]
##  [1]  24  33  34  97  98  99 121 150 155 195 199 211 237 244 248 308 328
```

```
## [18] 341 356
##
## $clusters[[8]]
##  [1]   2   9  21  29  35  53 116 117 128 129 131 133 140 145 153 175 203
## [18] 212 213 215 224 235 239 253 256 277 280 314 317 320 361 362 368
##
## $clusters[[9]]
##  [1]  11  36  79 173 176 177 178 185 218 231 232 261 262 313 327 351 367
## [18] 371 388
##
## $clusters[[10]]
##  [1]   3 101 102 108 111 115 141 146 160 171 182 186 210 236 269 271 272
## [18] 275 332 348 379
##
## $clusters[[11]]
##  [1]  17  20  32  46  78  84  87 132 142 159 174 179 189 193 220 221 223
## [18] 293 295 298 306 322 344 350 353 355 370 375 390 392
##
## $clusters[[12]]
##  [1]  22  45  61  64  66  68  77  80  82  91  92 110 112 143 157 161 191
## [18] 208 225 227 234 250 254 259 260 278 283 287 337 339 383 384
##
## $clusters[[13]]
##  [1]   7  12  19  23  37  49  56  63  65  75  81  93 113 122 130 134 139
## [18] 147 188 222 229 233 245 288 291 330 335 336 338 363
##
## $clusters[[14]]
##  [1]   1  50  52  86 124 152 167 205 241 243 246 251 268 276 294 297 301
## [18] 303 305 321 323 324 359 373 385 389
##
## $clusters[[15]]
##  [1]   8  10  47  71 119 148 156 162 180 196 198 242 247 296 340 365 366
## [18] 374 376 378 380 382
##
##
## $obs
##   [1] 14  8 10  1  6  1 13 15  8 15  9 13  5  4  4  2 11  5 13 11  8 12 13
##  [24]  7  4  6  3  5  8  2  2 11  7  7  8  9 13  6  4  1  5  4  4  4 12 11
##  [47] 15  6 13 14  3 14  8  5  1 13  1  4  4  6 12  1 13 12 13 12  5 12  6
##  [70]  6 15  6  1  3 13  1 12 11  9 12 13 12  2 11  4 14 11  4  4  1 12 12
##  [93] 13  4  4  2  7  7  7  3 10 10  5  1  6  4  3 10  6 12 10 12 13  2 10
## [116]  8  8  3 15  1  7 13  3 14  1  6  6  8  8 13  8 11  8 13  5  5  4  3
## [139] 13  8 10 11 12  3  8 10 13 15  2  7  6 14  8  6  7 15 12  6 11 10 12
## [162] 15  6  1  3  2 14  6  6  1 10  1  9 11  8  9  9  9 11 15  5 10  1  6
## [185]  9 10  5 13 11  1 12  4 11  2  7 15  2 15  7  5  6  1  8  1 14  1  3
## [208] 12  2 10  7  8  8  2  8  2  3  9  6 11 11 13 11  8 12  2 12  5 13  5
## [231]  9  9 13 12  8 10  7  1  8  1 14 15 14  7 13 14 15  7  4 12 14  2  8
## [254] 12  2  8  5  6 12 12  9  9  2  6  2  1  3 14 10  5 10 10  2  2 10 14
## [277]  8 12  5  8  5  6 12  6  3  2 12 13  2  1 13  5 11 14 11 15 14 11  5
## [300]  6 14  6 14  2 14 11  4  7  5  6  5  2  9  8  6  6  8  4  6  8 14 11
```

```
## [323] 14 14  3  3  9  7  6 13  5 10  6  4 13 13 12 13 12 15  7  3  1 11  5
## [346]  3  4 10  4 11  9  4 11  4 11  7  3  4 14  1  8  8 13  1 15 15  9  8
## [369]  5 11  9  2 14 15 11 15  4 15 10 15  5 15 12 12 14  4  1  9 14 11  1
## [392] 11
##
## $i
## [1] 20
```

Model 2

###Run the function on the car.test.frame data from rpart

```
library(rpart)
data("car.test.frame")
data1<-car.test.frame
rownames(data1) <- NULL
data1<-data1[-2]
data1<-data1[-4]
head(data1)
```

```
##   Price Reliability Mileage Weight Disp.  HP
## 1  8895           4      33   2560    97 113
## 2  7402           2      33   2345   114  90
## 3  6319           4      37   1845    81  63
## 4  6635           5      32   2260    91  92
## 5  6599           5      32   2440   113 103
## 6  8672           4      26   2285    97  82
```

```
#removed rows with NA values in them
data1<-na.exclude(data1)

data<-data1
k=5
t=15
q=1

#call the function:

clustreg(data,k,t,q)
```

```
## Warning in predict.lm(fit[[i]], data0): prediction from a rank-deficient
## fit may be misleading
```

```
## $fit
## $fit[[1]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
## (Intercept)  Reliability      Mileage       Weight        Disp.
##    -299.571     -127.946     -379.150        6.611      -68.798
##          HP
##     116.885
##
##
## $fit[[2]]
##
```

```
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
## (Intercept)  Reliability      Mileage       Weight        Disp.
##    35080.66     -3292.52      -533.07        10.21      -267.18
##          HP
##      111.14
##
##
## $fit[[3]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
## (Intercept)  Reliability      Mileage       Weight        Disp.
##   34299.228     7915.133    -1875.484       -6.288       83.870
##          HP
##     -29.229
##
##
## $fit[[4]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
## (Intercept)  Reliability      Mileage       Weight        Disp.
##   -1487.330    -1161.471       48.260        5.032      -55.783
##          HP
##      80.721
##
##
## $fit[[5]]
##
## Call:
## lm(formula = formula, data = cc[[i]])
##
## Coefficients:
## (Intercept)  Reliability      Mileage       Weight        Disp.
##   -18842.46      -360.64        56.48        12.09       -26.85
##          HP
##       -7.03
##
##
##
## $rsq
## [1] 0.9162624 1.0252097 0.9958277 0.9956375 0.9956375
##
```

```
## $crsq
## [1] 0.9763287 0.9997833 0.9995029 0.9963291 0.9904294
##
## $clusters
## $clusters[[1]]
##  [1]  4 14 20 23 25 26 29 34 41 42 44
##
## $clusters[[2]]
## [1]  3  9 17 18 32 43 49
##
## $clusters[[3]]
## [1]  6 22 27 31 33 40 46 48
##
## $clusters[[4]]
##  [1]  7 10 11 12 13 16 19 28 30 37 38 39 45 47
##
## $clusters[[5]]
## [1]  1  2  5  8 15 21 24 35 36
##
##
## $obs
##  [1] 5 5 2 1 5 3 4 5 2 4 4 4 4 1 5 4 2 2 4 1 5 3 1 5 1 1 3 4 1 4 3 2 3 1 5
## [36] 5 4 4 4 3 1 1 2 1 4 3 4 3 2
##
## $i
## [1] 5
```