HW2.5_Mary_Futey

Mary Futey

5/19/2020

```
library(ggplot2)
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.6.2
```

```
library(cluster)
library(fpc)
```

load data and explore

```
boston <- Boston
```

kmeans

```
set.seed(4)
# the result printed is the best result of 20 tries
# clustering vector: information on each observation (what cluster it is assigned)
# sum of squares: measure of variance inside the cluster
km_res <- kmeans(boston, centers = 2, nstart = 20)
km_res
```

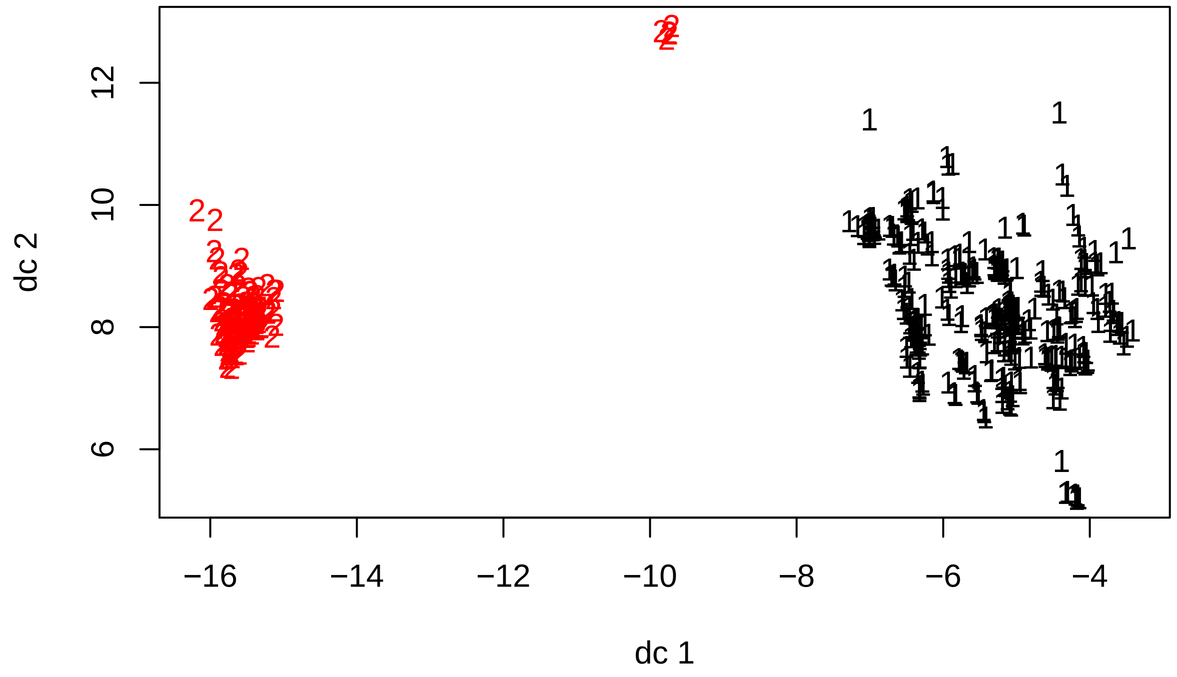
```
## K-means clustering with 2 clusters of sizes 369, 137
##
## Cluster means:
##          crim       zn     indus       chas       nox       rm      age      dis
## 1  0.3887744 15.58266  8.420894 0.07317073 0.5118474 6.388005 60.63225 4.441272
## 2 12.2991617  0.00000 18.451825 0.05839416 0.6701022 6.006212 89.96788 2.054470
##          rad      tax  ptratio    black    lstat     medv
## 1  4.455285 311.9268 17.80921 381.0426 10.41745 24.85718
## 2 23.270073 667.6423 20.19635 291.0391 18.67453 16.27226
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   2   2   2
## 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500
##   2   2   2   2   2   2   2   2   2   2   2   2   2   1   1   1   1   1   1   1
## 501 502 503 504 505 506
##   1   1   1   1   1   1
##
## Within cluster sum of squares by cluster:
## [1] 2868770 2896224
##  (between_SS / total_SS =  70.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# withinss: sum of squares within each cluster
# tot.withins: what we are trying to minimize
```

```r
str(km_res)
```

```
## List of 9
##  $ cluster     : Named int [1:506] 1 1 1 1 1 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:506] "1" "2" "3" "4" ...
##  $ centers     : num [1:2, 1:14] 0.389 12.299 15.583 0 8.421 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "1" "2"
##   .. ..$ : chr [1:14] "crim" "zn" "indus" "chas" ...
##  $ totss       : num 19401064
##  $ withinss    : num [1:2] 2868770 2896224
##  $ tot.withinss: num 5764994
##  $ betweenss   : num 13636070
##  $ size        : int [1:2] 369 137
##  $ iter        : int 1
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```r
# vizualize results of clustering by kmeans
plotcluster(boston, km_res$cluster)
```



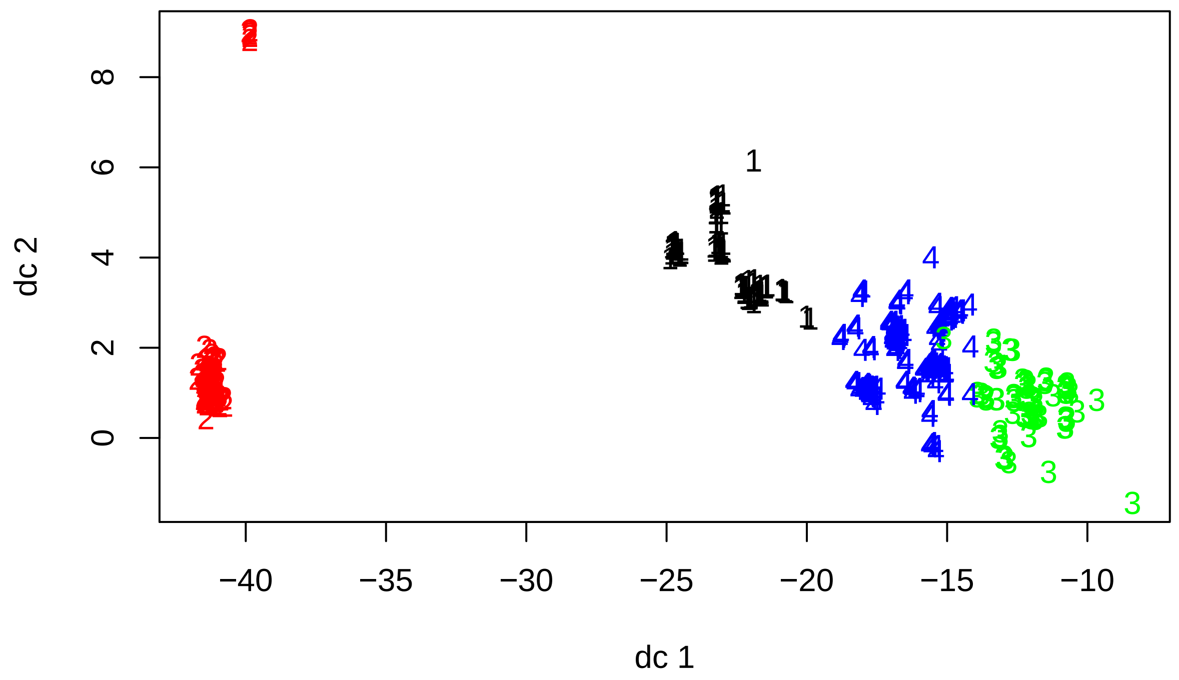
try with different cluster sizes

```r
set.seed(4)

# 4 cluster sizes produces good results based on plot and tot.withinss,
# but has outliers in 2nd cluster
km_res3 <- kmeans(boston, 4, nstart = 3)
km_res3
```

```
## K-means clustering with 4 clusters of sizes 96, 137, 81, 192
##
```

```
## Cluster means:
##          crim       zn     indus       chas       nox       rm      age
## 1  0.81874729  7.96875 13.589896 0.07291667 0.5962187 6.169427 75.68750
## 2 12.29916168  0.00000 18.451825 0.05839416 0.6701022 6.006212 89.96788
## 3  0.08724728 24.11728  6.198642 0.08641975 0.4659494 6.588148 49.26790
## 4  0.30099479 15.78906  6.773906 0.06770833 0.4890250 6.412859 57.89896
##        dis       rad      tax  ptratio    black     lstat     medv
## 1 3.054214  4.864583 408.4167 17.70417 362.3918 13.009062 22.18125
## 2 2.054470 23.270073 667.6423 20.19635 291.0391 18.674526 16.27226
## 3 4.916821  3.567901 225.9259 17.83210 391.3574  8.404444 28.91358
## 4 4.934178  4.625000 299.9635 17.85208 386.0164  9.970885 24.48385
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   4   3   3   3   3   3   4   4   4   4   4   4   4   4   4   4   4   4   4   4
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   3
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   1   3   4   3   4   4
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   4   4   4   4   3   4   4   4   4   4   4   4   4   4   1   1   1   1   1   1
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   4   4   4   4   3   3   3   3   4   4   4   4   4   4   4   4   4   4   4   4
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   3   3   3   3   3   3   3   1   1   1   1   1   1   1   1   1   1   1   1   1
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   1   1   1   1   1   1   1   1   1   1   1   1   4   4   4   4   4   4   4   3
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
##   3   3   3   3   3   3   3   1   1   1   1   1   1   3   3   3   4   4   4   1
## 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220
##   1   4   4   3   3   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
## 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240
##   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
## 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260
##   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   3   4   4   4
## 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280
##   4   4   4   4   4   4   4   4   3   3   3   3   3   3   3   3   3   3   3   3
## 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300
##   3   3   3   3   4   4   3   4   4   4   3   3   3   4   4   4   4   4   4   4
## 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320
##   4   4   4   4   3   3   3   3   4   4   4   4   4   4   4   4   4   4   4   4
## 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340
##   4   4   4   4   4   4   4   4   1   1   1   4   4   3   3   3   3   3   3   3
## 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
##   3   4   1   1   1   4   4   4   4   4   4   1   1   3   4   4   2   2   2   2
## 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500
##   2   2   2   2   2   2   2   2   2   2   2   2   2   1   1   1   1   1   1   1
## 501 502 503 504 505 506
##   1   4   4   4   4   4
##
## Within cluster sum of squares by cluster:
## [1]  604434.9 2896224.4  186507.3  458666.1
##  (between_SS / total_SS =  78.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
plotcluster(boston, km_res3$cluster)
```

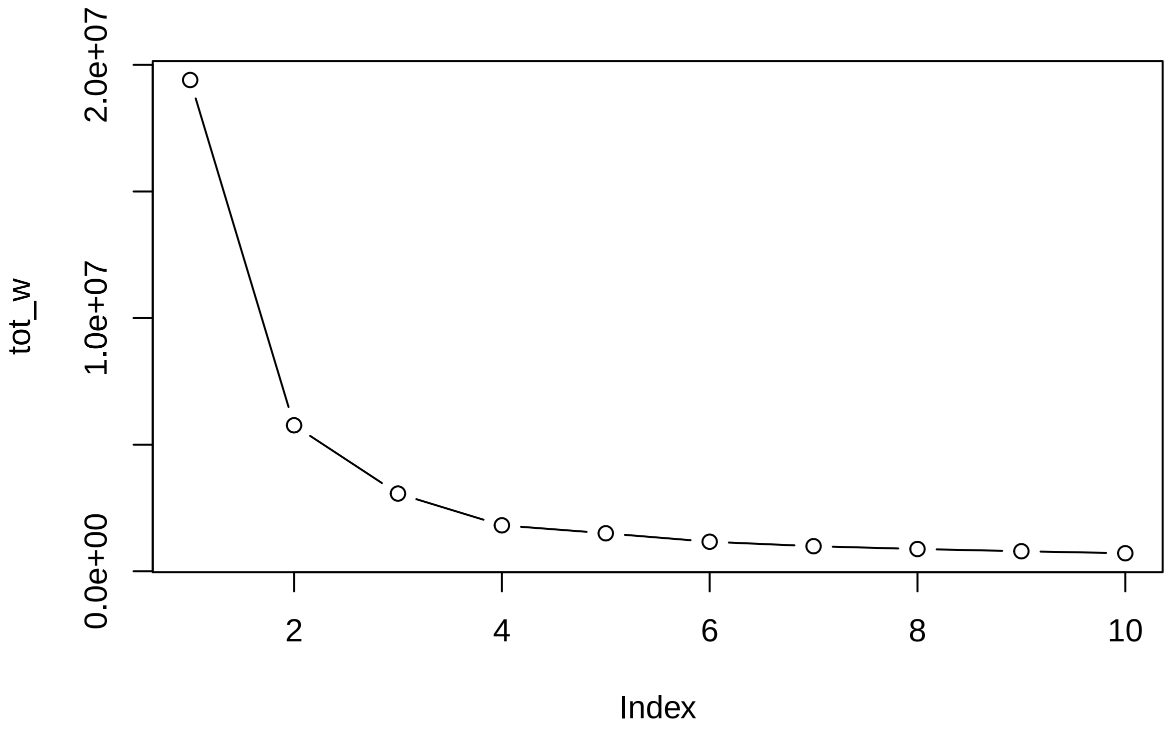


```
# check res
str(km_res3)
```

```
## List of 9
##  $ cluster     : Named int [1:506] 4 3 3 3 3 3 4 4 4 4 ...
##   ..- attr(*, "names")= chr [1:506] "1" "2" "3" "4" ...
##  $ centers     : num [1:4, 1:14] 0.8187 12.2992 0.0872 0.301 7.9688 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:4] "1" "2" "3" "4"
##   .. ..$ : chr [1:14] "crim" "zn" "indus" "chas" ...
##  $ totss       : num 19401064
##  $ withinss    : num [1:4] 604435 2896224 186507 458666
##  $ tot.withinss: num 4145833
##  $ betweenss   : num 15255231
##  $ size        : int [1:4] 96 137 81 192
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

scree plot

```
# scree plot

tot_w <- rep(0,10)

# perform k means with 1 to 10 clusters
# total withinss is measure of quality for clustering
set.seed(8)
for (k in 1:10) {
  tot_w[k] <- kmeans(boston, k, nstart = 20)$tot.withinss
}

# see that 3, maybe 4, is optimal
plot(tot_w, type = "b")
```



```
boston_F <- boston

# create categorical medv (Median house value) to use as class for pca and hc
quant_00 = min(boston$medv)
quant_25 = quantile(boston$medv, 0.25)
quant_50 = quantile(boston$medv, 0.50)
```

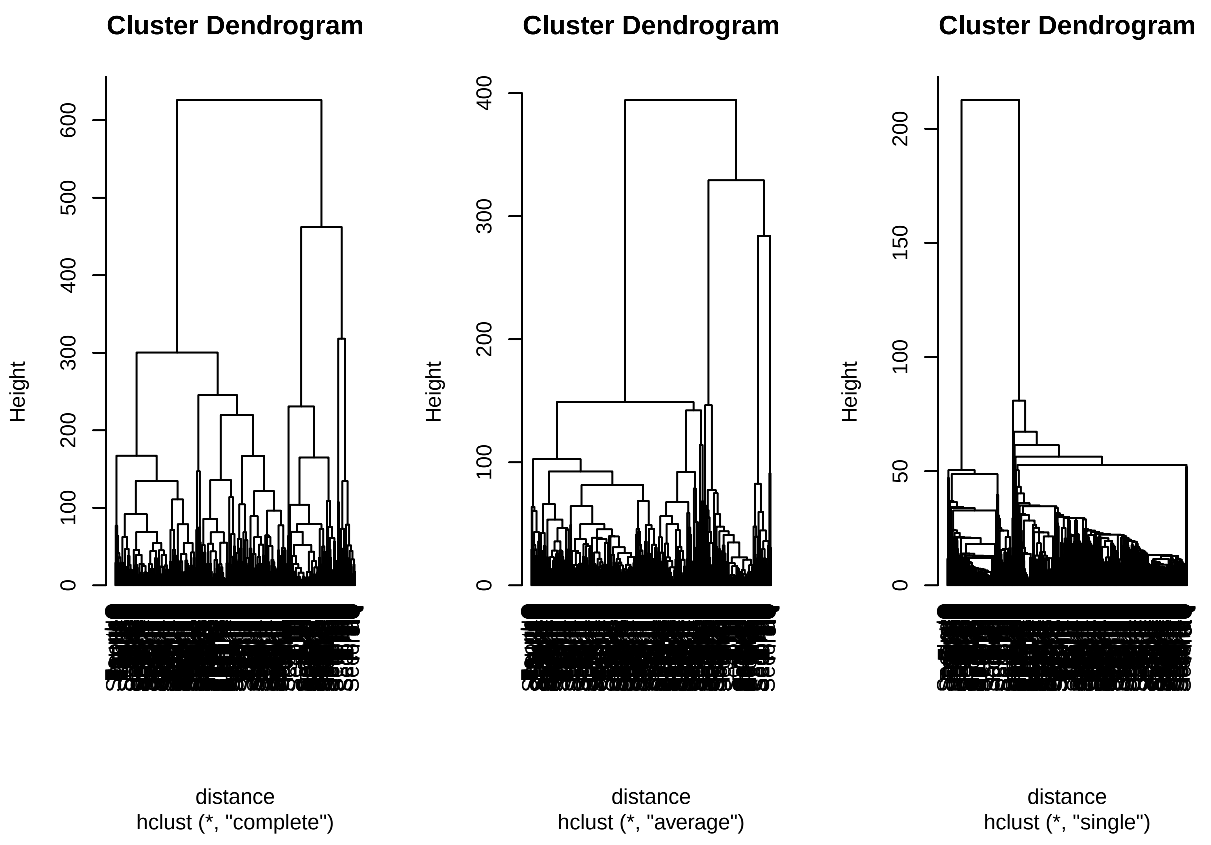
```r
quant_75 = quantile(boston$medv, 0.75)
quant_100 = max(boston$medv)

rb = rbind(quant_00, quant_25, quant_50, quant_75, quant_100)
dimnames(rb)[[2]] = "Value"

boston_F$medv_F[boston$medv >= quant_00 &
                    boston$medv < quant_25] = "FirstQ"
boston_F$medv_F[boston$medv >= quant_25 &
                    boston$medv < quant_50] = "SecondQ"
boston_F$medv_F[boston$medv >= quant_50 &
                    boston$medv <= quant_75] = "ThirdQ"
boston_F$medv_F[boston$medv >= quant_75 &
                    boston$medv <= quant_100] = "FourthQ"
boston_F$medv_F = factor(boston_F$medv_F,
levels=c("FirstQ", "SecondQ", "ThirdQ", "FourthQ"))

# remove median house value from boston housing dataset
boston_pca <- subset(boston, select = -medv)
```

hierarchical clustering

```r
distance <- dist(boston)

# takes max distance
hc_comp <- hclust(distance, method = "complete")
# takes average distance
hc_avg <- hclust(distance, method = "average")
# minimal distance
hc_sing <- hclust(distance, method = "single")
```

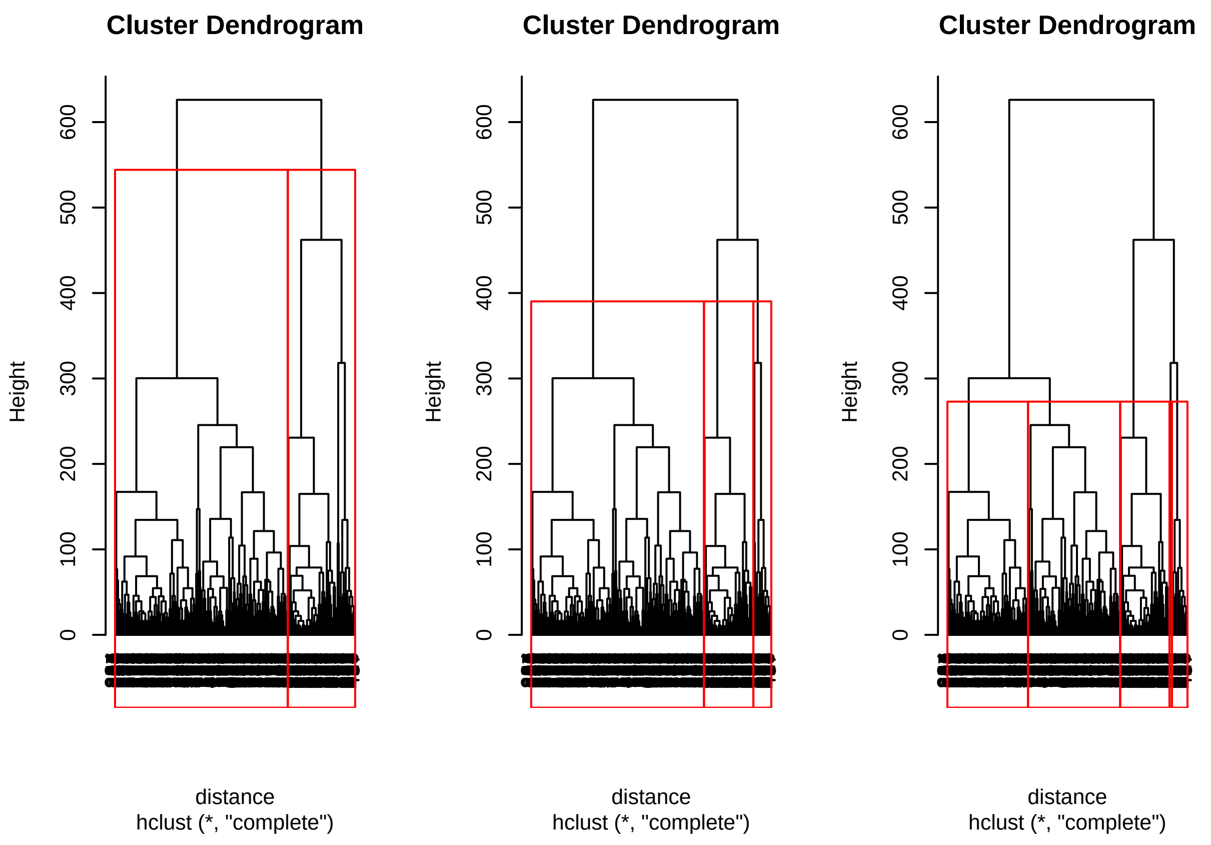
hc dendrograms

```r
# organzine plotting window with 3 cells aligned as columns
par(mfrow = c(1, 3))

# complete and average look similar
plot(hc_comp, hang = -1, labels = boston_F$medv_F)
plot(hc_avg, hang = -1, labels = boston_F$medv_F)
plot(hc_sing, hang = -1, labels = boston_F$medv_F)
```



```
# reset plotting window
par(mfrow = c(1, 1))
```

```
par(mfrow = c(1, 3))

# create hc object with complete and decide how many clusters to choose
hc <- hclust(distance, method = "complete")

# create 3 plots with different number of clusters outlined

# division by 2 or 3 clusters looks best
plot(hc, hang = -1, labels = NULL)
rect.hclust(hc, k = 2)

plot(hc, hang = -1, labels = NULL)
rect.hclust(hc, k = 3)

# after 4 clusters begin to have v. small clusters
plot(hc, hang = -1, labels = NULL)
rect.hclust(hc, k = 5)
```



```r
par(mfrow = c(1, 1))
```

PCA

```r
# check column means
colMeans(boston)
```

```
##         crim           zn        indus         chas          nox           rm
##   3.61352356  11.36363636  11.13677866   0.06916996   0.55469506   6.28463439
##          age          dis          rad          tax      ptratio        black
##  68.57490119   3.79504269   9.54940711 408.23715415  18.45553360 356.67403162
##        lstat         medv
##  12.65306324  22.53280632
```

```r
# check sd
apply(boston, 2, sd)
```

```
##        crim          zn       indus        chas         nox          rm
##   8.6015451  23.3224530   6.8603529   0.2539940   0.1158777   0.7026171
##         age         dis         rad         tax     ptratio       black
##  28.1488614   2.1057101   8.7072594 168.5371161   2.1649455  91.2948644
##       lstat        medv
##   7.1410615   9.1971041
```

```r
# perform PCA with scaled data based on the above checks
pca <- prcomp(x = boston_pca, center = TRUE, scale = TRUE)
pca
```

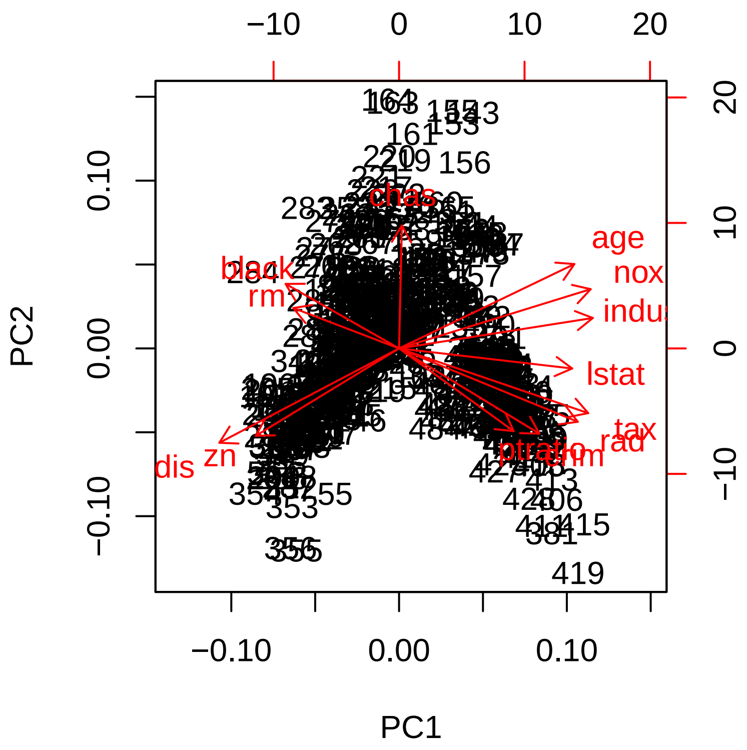
```
## Standard deviations (1, .., p=13):
```

```
##  [1] 2.4752472 1.1971947 1.1147272 0.9260535 0.9136826 0.8108065 0.7316803
##  [8] 0.6293626 0.5262541 0.4692950 0.4312938 0.4114644 0.2520104
##
## Rotation (n x k) = (13 x 13):
##                  PC1         PC2         PC3         PC4          PC5
## crim     0.250951397 -0.31525237  0.24656649 -0.06177071  0.082156919
## zn      -0.256314541 -0.32331290  0.29585782 -0.12871159  0.320616987
## indus    0.346672065  0.11249291 -0.01594592 -0.01714571 -0.007811194
## chas     0.005042434  0.45482914  0.28978082 -0.81594136  0.086530945
## nox      0.342852313  0.21911553  0.12096411  0.12822614  0.136853557
## rm      -0.189242570  0.14933154  0.59396117  0.28059184 -0.423447195
## age      0.313670596  0.31197778 -0.01767481  0.17520603  0.016690847
## dis     -0.321543866 -0.34907000 -0.04973627 -0.21543585  0.098592247
## rad      0.319792768 -0.27152094  0.28725483 -0.13234996 -0.204131621
## tax      0.338469147 -0.23945365  0.22074447 -0.10333509 -0.130460565
## ptratio  0.204942258 -0.30589695 -0.32344627 -0.28262198 -0.584002232
## black   -0.202972612  0.23855944 -0.30014590 -0.16849850 -0.345606947
## lstat    0.309759840 -0.07432203 -0.26700025 -0.06941441  0.394561129
##                 PC6          PC7          PC8         PC9         PC10
## crim    -0.21965961  0.777607207 -0.153350477  0.26039028 -0.019369130
## zn      -0.32338810 -0.274996280  0.402680309  0.35813749 -0.267527234
## indus   -0.07613790 -0.339576454 -0.173931716  0.64441615  0.363532262
## chas     0.16749014  0.074136208  0.024662148 -0.01372777  0.006181836
## nox     -0.15298267 -0.199634840 -0.080120560 -0.01852201 -0.231056455
## rm       0.05926707  0.063939924  0.326752259  0.04789804  0.431420193
## age     -0.07170914  0.116010713  0.600822917 -0.06756218 -0.362778957
## dis      0.02343872 -0.103900440  0.121811982 -0.15329124  0.171213138
## rad     -0.14319401 -0.137942546 -0.080358311 -0.47089067 -0.021909452
## tax     -0.19293428 -0.314886835 -0.082774347 -0.17656339  0.035168348
## ptratio  0.27315330  0.002323869  0.317884202  0.25442836 -0.153430488
## black   -0.80345454  0.070294759  0.004922915 -0.04489802  0.096515117
## lstat   -0.05321583  0.087011169  0.424352926 -0.19522139  0.600711409
##                PC11         PC12         PC13
## crim    -0.10964435 -0.086761070  0.045952304
## zn       0.26275629  0.071425278 -0.080918973
## indus   -0.30316943  0.113199629 -0.251076540
## chas     0.01392667  0.003982683  0.035921715
## nox      0.11131888 -0.804322567  0.043630446
## rm       0.05316154 -0.152872864  0.045567096
## age     -0.45915939  0.211936074 -0.038550683
## dis     -0.69569257 -0.390941129 -0.018298538
## rad      0.03654388  0.107025890 -0.633489720
## tax     -0.10483575  0.215191126  0.720233448
## ptratio  0.17450534 -0.209598826  0.023398052
## black    0.01927490 -0.041723158 -0.004463073
## lstat    0.27138243 -0.055225960  0.024431677
```

```
# first 4 components explain ~75% of variance
summary(pca)
```

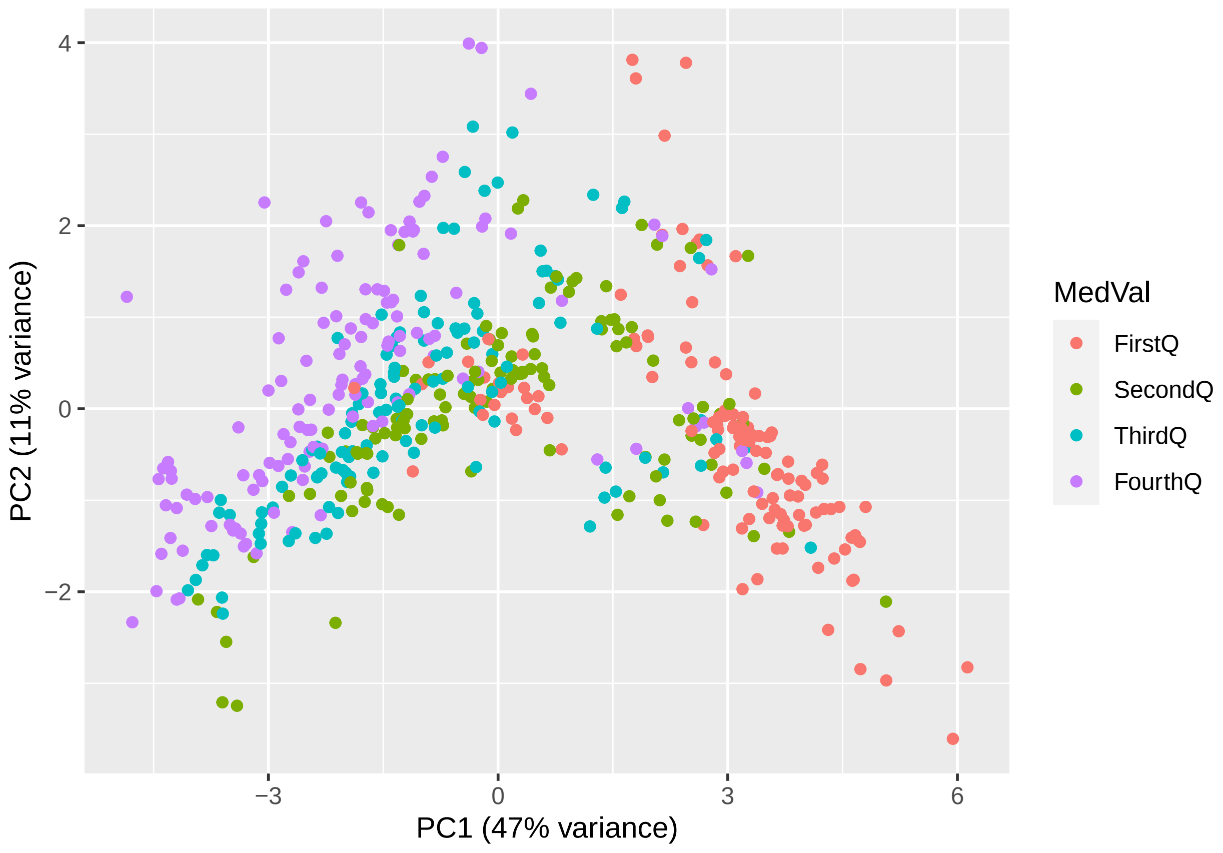
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.4752 1.1972 1.11473 0.92605 0.91368 0.81081 0.73168
## Proportion of Variance 0.4713 0.1103 0.09559 0.06597 0.06422 0.05057 0.04118
## Cumulative Proportion  0.4713 0.5816 0.67713 0.74310 0.80732 0.85789 0.89907
##                             PC8    PC9    PC10    PC11    PC12    PC13
## Standard deviation     0.62936 0.5263 0.46930 0.43129 0.41146 0.25201
## Proportion of Variance 0.03047 0.0213 0.01694 0.01431 0.01302 0.00489
## Cumulative Proportion  0.92954 0.9508 0.96778 0.98209 0.99511 1.00000
```

```
# chas is co-directed with PC1
biplot(pca)
```

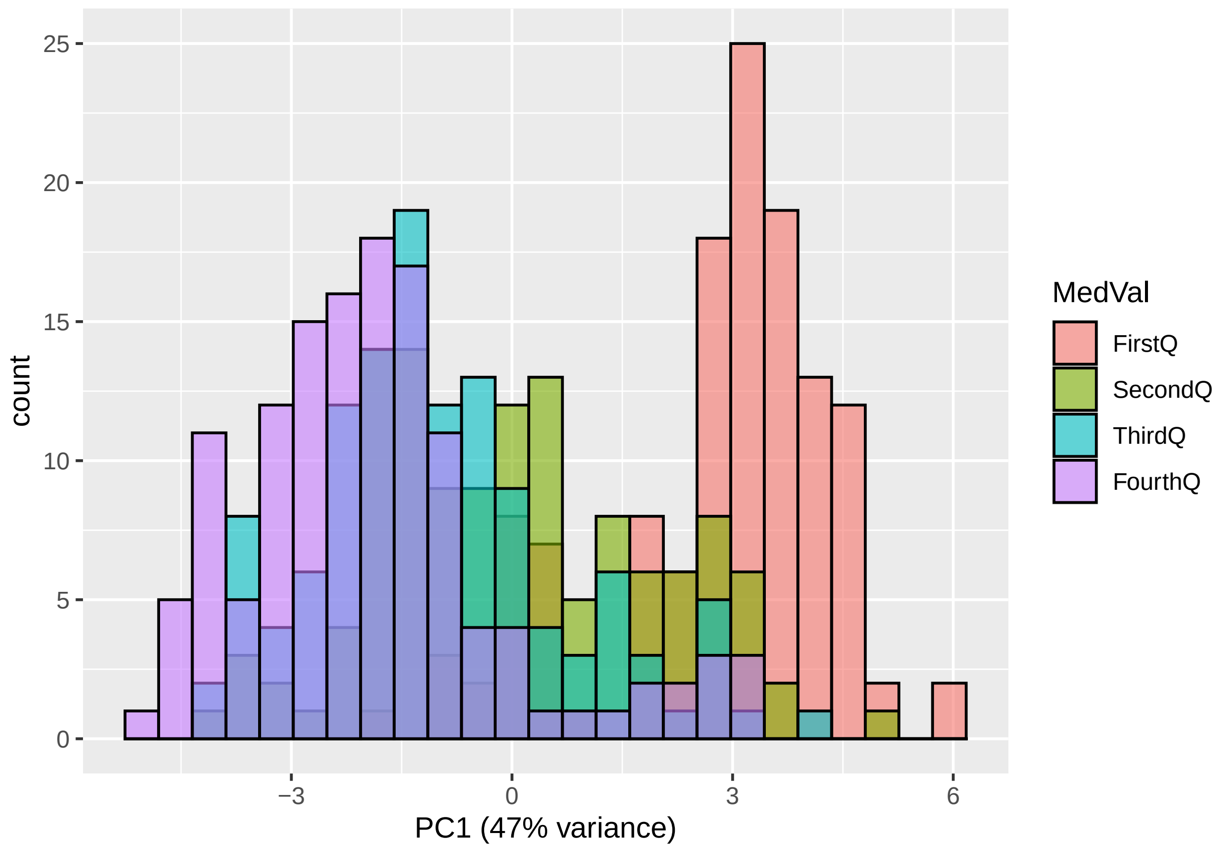


```
# data frame with MedVal
pca_adj <- data.frame(pca$x, MedVal = boston_F$medv_F)
```

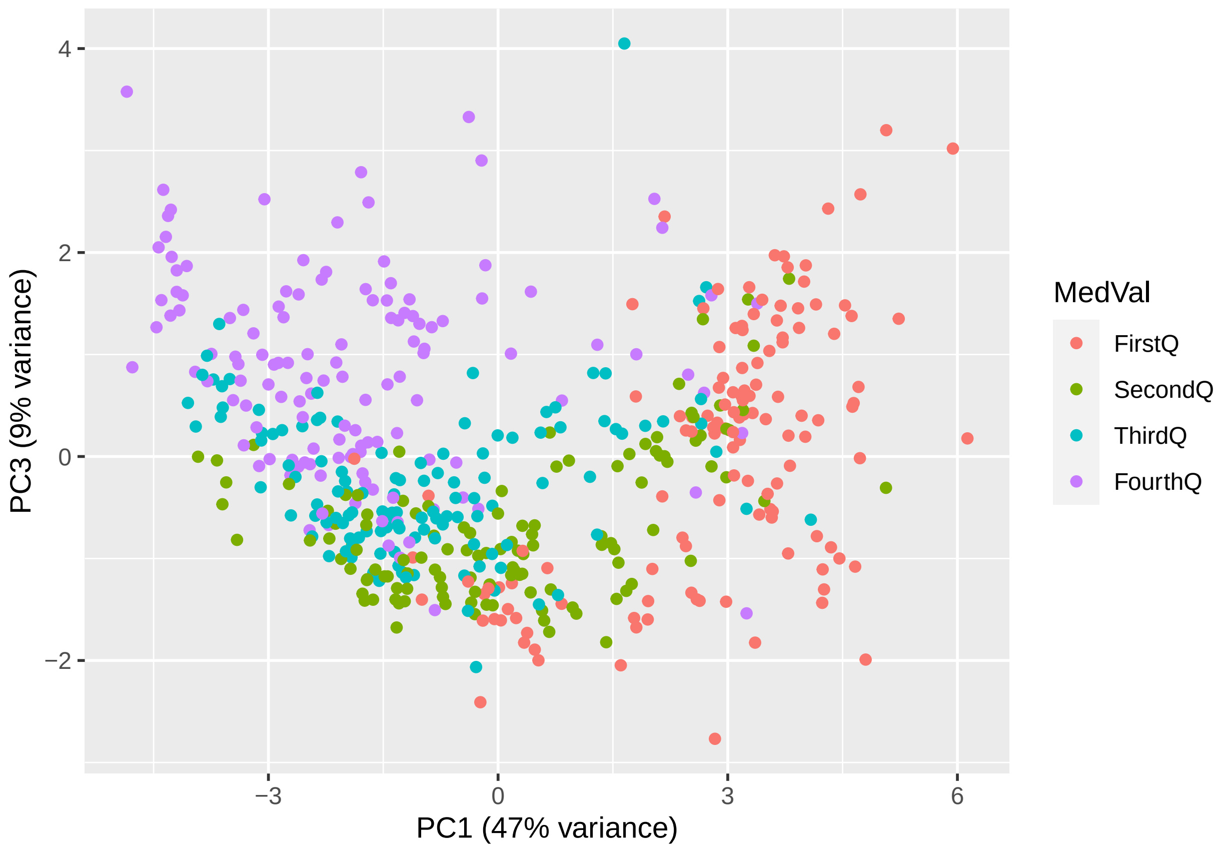
```
# check clustering based on MedVal PC1 and PC2
p1 <- ggplot(data = pca_adj,
       aes(x = PC1,
       y = PC2,
       color = MedVal)) +
  geom_point() +
  labs(x = "PC1 (47% variance)") +
  labs(y = "PC2 (11% variance)" )

p1
```



```
# reduce to one dimension
p2 <- ggplot(data = pca_adj,
       aes(x = PC1,
           fill = MedVal)) +
  geom_histogram(alpha = 0.6,
                 color = "black",
                 bins = 25,
                 position = "identity") +
  labs(x = "PC1 (47% variance)")

p2
```



```
# check clustering based on MedVal fpr PC1 and PC3
p3 <- ggplot(data = pca_adj,
       aes(x = PC1,
       y = PC3,
       color = MedVal)) +
  geom_point() +
  labs(x = "PC1 (47% variance)") +
  labs(y = "PC3 (9% variance)" )

p3
```



```
# get variance: indicator if interesting / informative or not
pca_var <- pca$sdev^2
pca_var
```

```
##  [1] 6.12684883 1.43327512 1.24261667 0.85757511 0.83481594 0.65740718
##  [7] 0.53535609 0.39609731 0.27694333 0.22023782 0.18601437 0.16930298
## [13] 0.06350926
```

```
# pve: proportion of variance explained
pca_pve <- pca_var / sum(pca_var)
# how informaitve each PC is (highest to low)
pca_pve
```

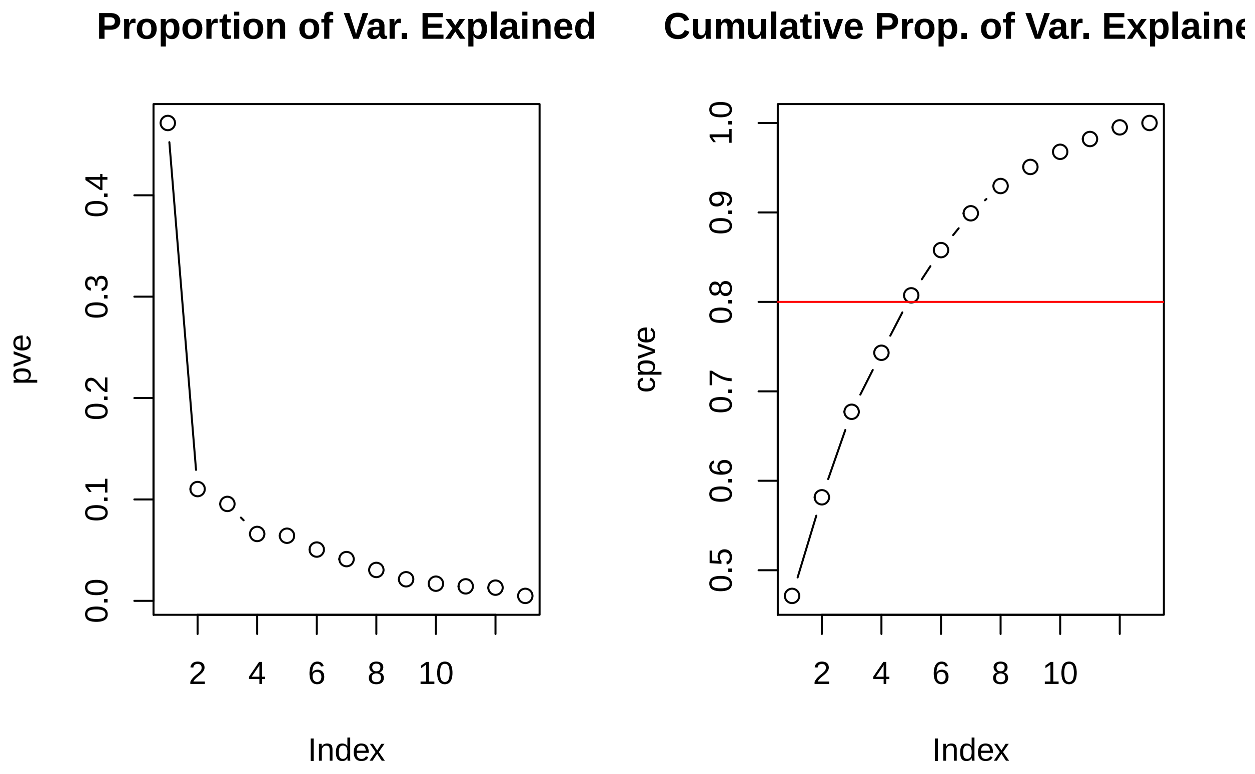
```
##  [1] 0.471296064 0.110251932 0.095585898 0.065967316 0.064216611 0.050569783
##  [7] 0.041181237 0.030469024 0.021303333 0.016941371 0.014308797 0.013023306
## [13] 0.004885328
```

```
 # cummlative: helpful in deciding how many PC to take: cutoff 80% (5 PCs)
cumsum(pca_pve)
```

```
##  [1] 0.4712961 0.5815480 0.6771339 0.7431012 0.8073178 0.8578876 0.8990688
##  [8] 0.9295379 0.9508412 0.9677826 0.9820914 0.9951147 1.0000000
```

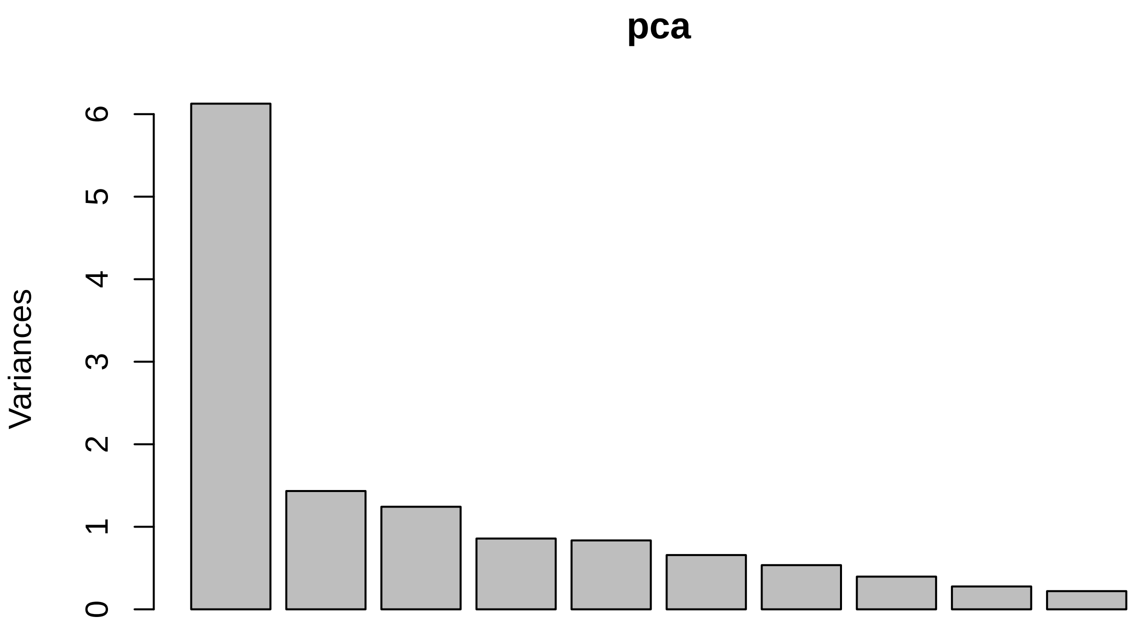
```
par(mfrow = c(1, 2))
#proportion of variance
pve <- summary(pca)$importance[2,]

#cummulative variance
cpve <- summary(pca)$importance[3,]
```

```r
plot(pve, type = "b", main = "Proportion of Var. Explained")

# 5 features explain 80%
plot(cpve, type = "b", main = "Cumulative Prop. of Var. Explained")
abline(h = 0.8, col = "red")
```

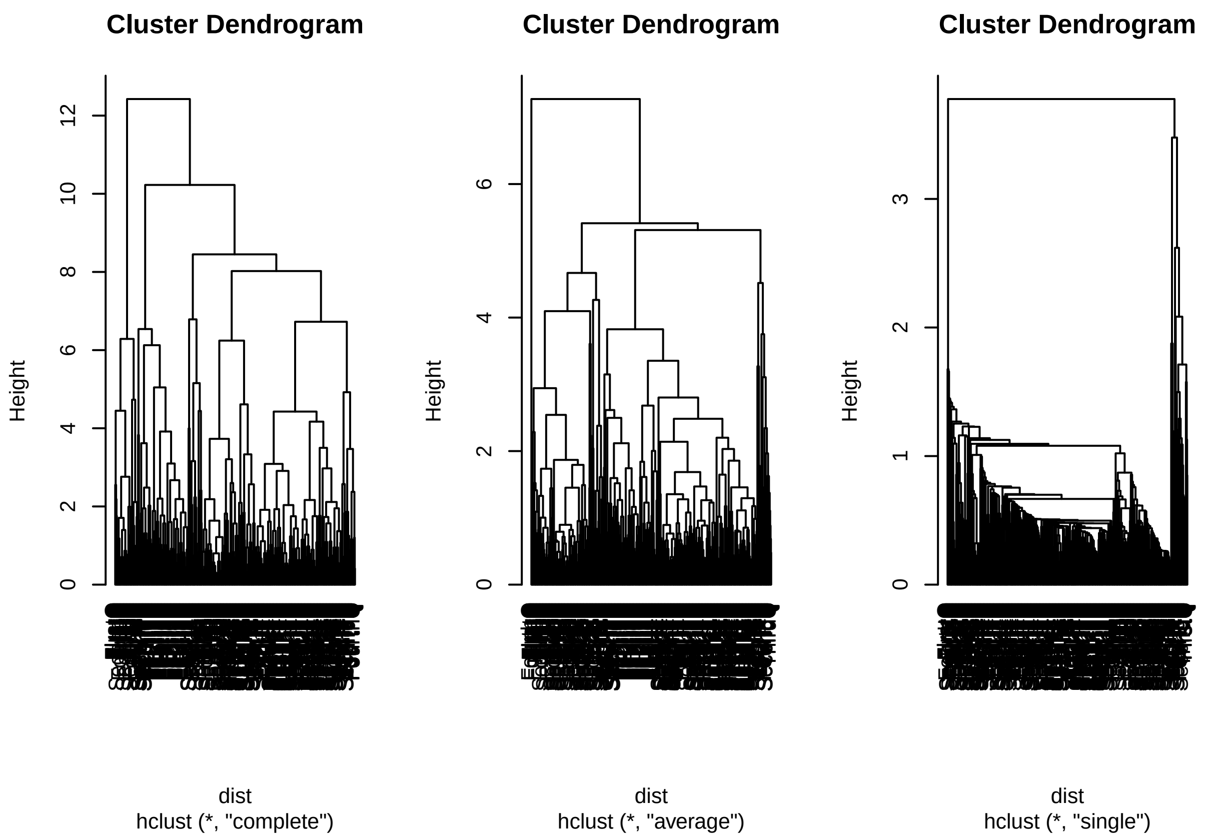


```r
par(mfrow = c(1, 1))
```

```r
plot(pca)
```

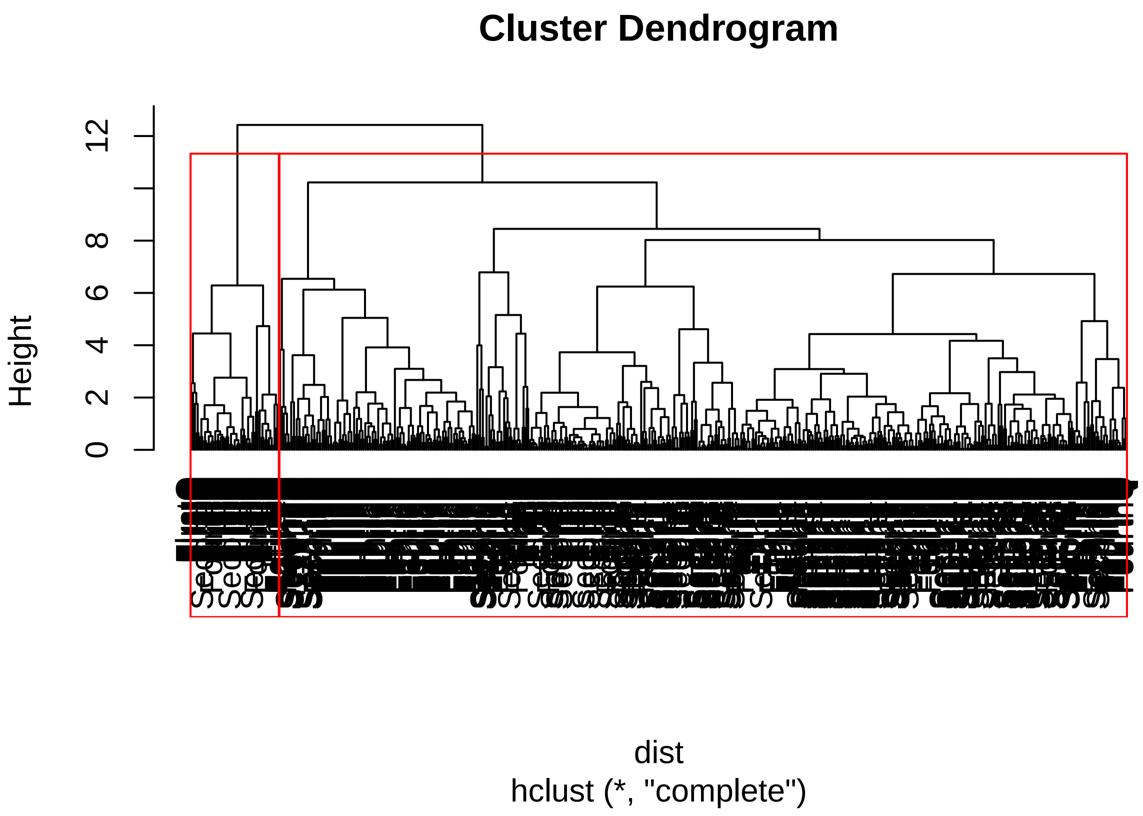


```r
par(mfrow = c(1, 3))

# based on above, decided to use first 5 PCs
# clustering visually looks better than initial

pca_tune <- pca_adj[, 1:5]
dist <- dist(pca_tune)

# complete looks best, single terrible
plot(hclust(dist, method = "complete"), hang = -1, labels = pca_adj$MedVal)
plot(hclust(dist, method = "average"), hang = -1, labels = pca_adj$MedVal)
plot(hclust(dist, method = "single"), hang = -1, labels = pca_adj$MedVal)
```

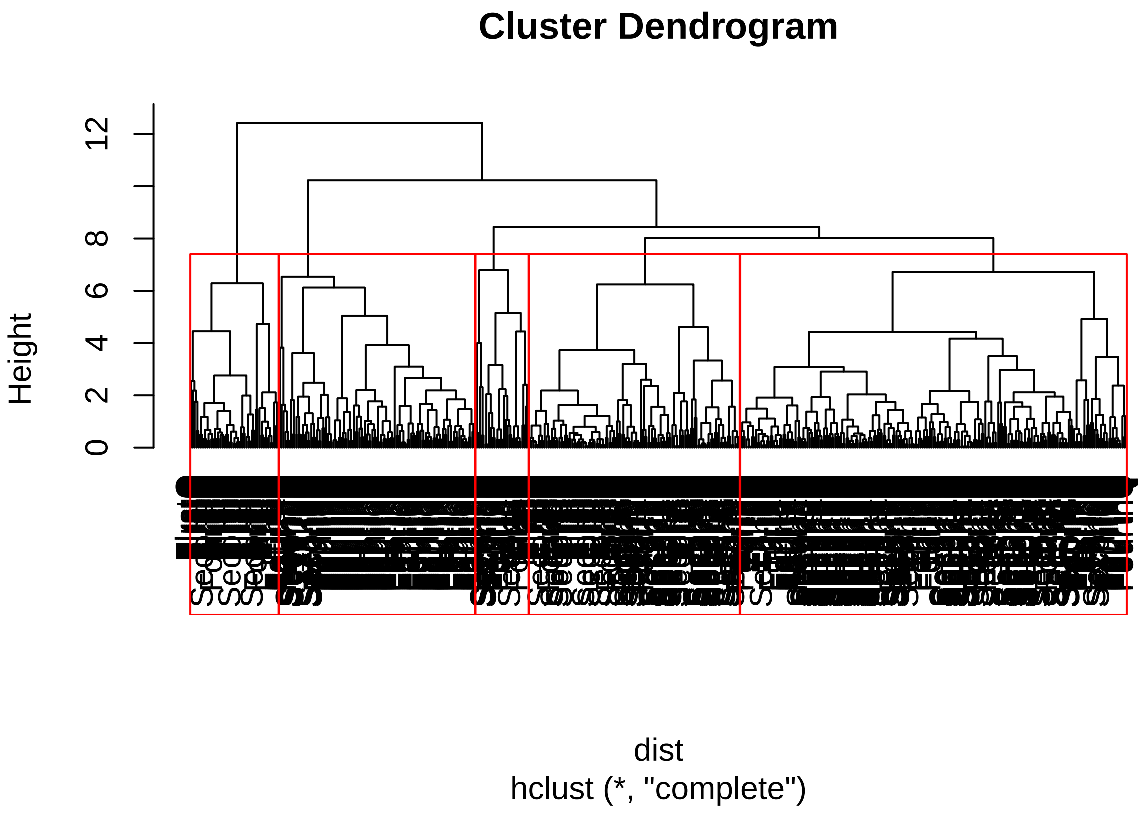


```r
par(mfrow = c(1, 1))
```

```r
# use complete method
hc_tune <- hclust(dist, method = "complete")

# division by two clusters
plot(hc_tune, hang = -1, labels = pca_adj$MedVal)
rect.hclust(hc_tune, k = 2)
```



```r
plot(hc_tune, hang = -1, labels = pca_adj$MedVal)
rect.hclust(hc_tune, k = 5)
```



```
# check intersection of clusters (k = 2-5) and
# compare with original hc without tuning: improved
table(pca_adj$MedVal, cutree(hc_tune, 2))
```

```
##
##             1   2
##   FirstQ   85  42
##   SecondQ 121   3
##   ThirdQ  122   1
##   FourthQ 130   2
```

```
table(pca_adj$MedVal, cutree(hc, 2))
```

```
##
##             1   2
##   FirstQ   36  91
##   SecondQ  98  26
##   ThirdQ  109  14
##   FourthQ 121  11
```

```
table(pca_adj$MedVal, cutree(hc_tune, 3))
```

```
##
##             1   2   3
##   FirstQ    0  85  42
##   SecondQ   7 114   3
##   ThirdQ   29  93   1
##   FourthQ  70  60   2
```

```
table(pca_adj$MedVal, cutree(hc, 3))
```

```
##
##             1   2   3
##   FirstQ   36  33  58
##   SecondQ  98   4  22
##   ThirdQ  109   1  13
##   FourthQ 121   0  11
```

```
table(pca_adj$MedVal, cutree(hc_tune, 4))
```

```
##
##             1   2   3   4
##   FirstQ    0  80   5  42
##   SecondQ   7 110   4   3
##   ThirdQ   29  83  10   1
##   FourthQ  70  50  10   2
```

```
table(pca_adj$MedVal, cutree(hc, 4))
```

```
##
##             1   2   3   4
##   FirstQ   36   4  58  29
##   SecondQ  98   1  22   3
##   ThirdQ  109   0  13   1
##   FourthQ 121   0  11   0
```

```
# difficult to make a call as they are not cleanly clustered,
# but 4 looks best to me (above)
```

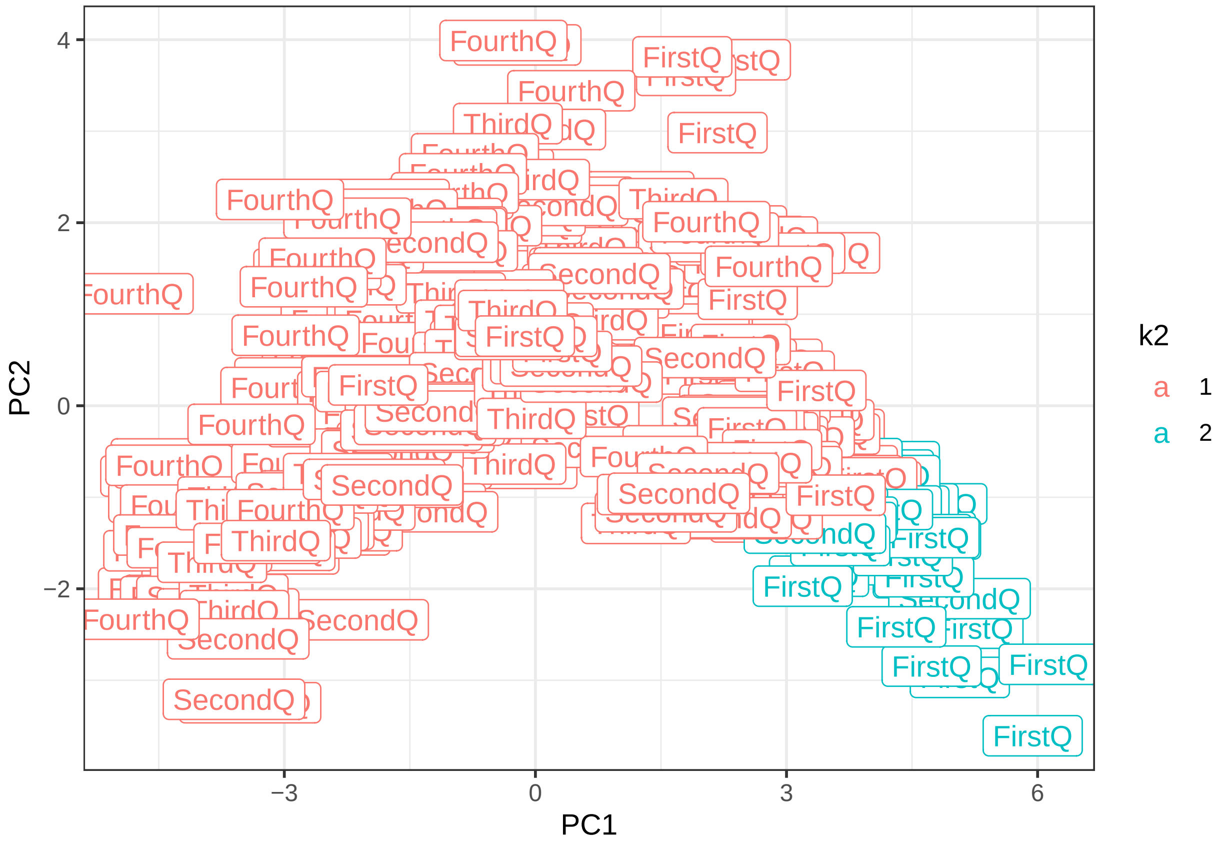
```r
table(pca_adj$MedVal, cutree(hc_tune, 5))
```

```
##
##            1  2  3  4  5
##   FirstQ   0 22 58  5 42
##   SecondQ  7 73 37  4  3
##   ThirdQ  29 70 13 10  1
##   FourthQ 70 44  6 10  2
```

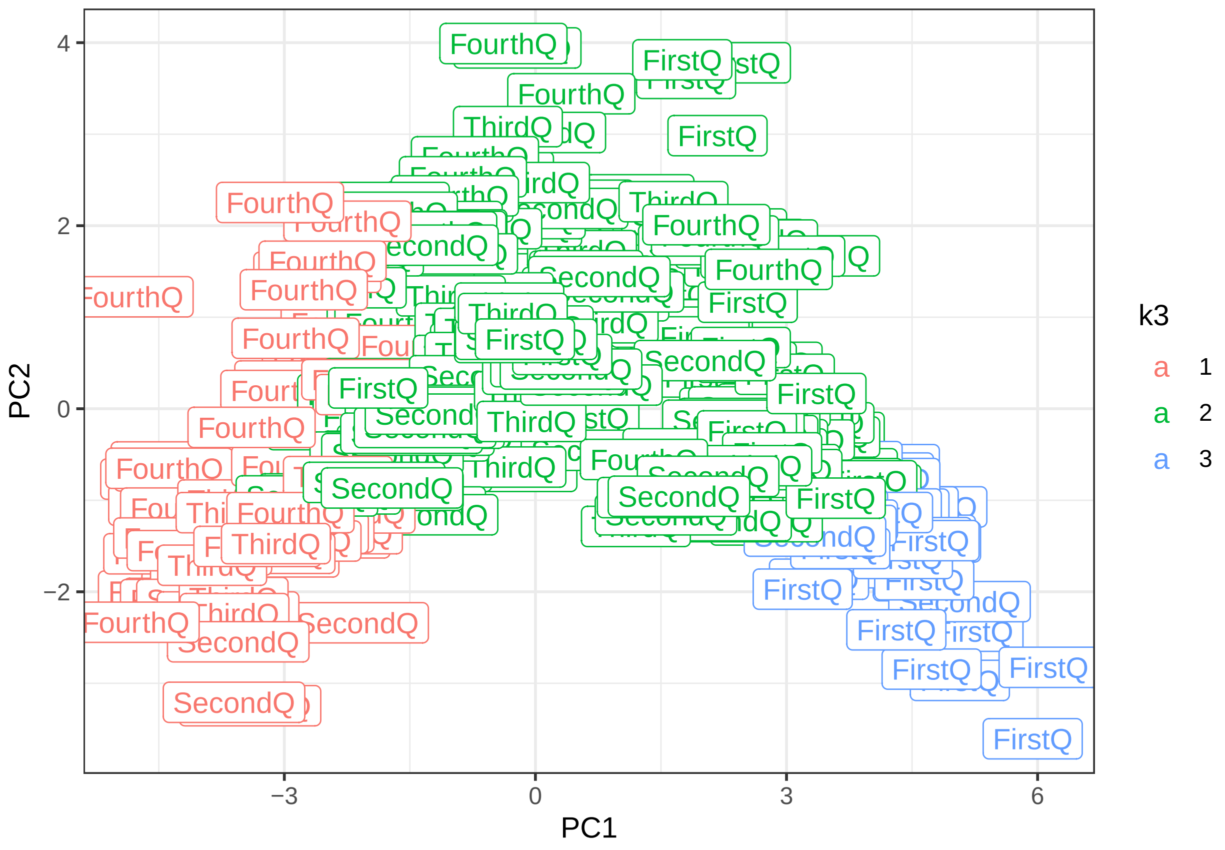
```r
table(pca_adj$MedVal, cutree(hc, 5))
```

```
##
##            1  2  3  4  5
##   FirstQ  20 16  4 58 29
##   SecondQ 56 42  1 22  3
##   ThirdQ  67 42  0 13  1
##   FourthQ 51 70  0 11  0
```

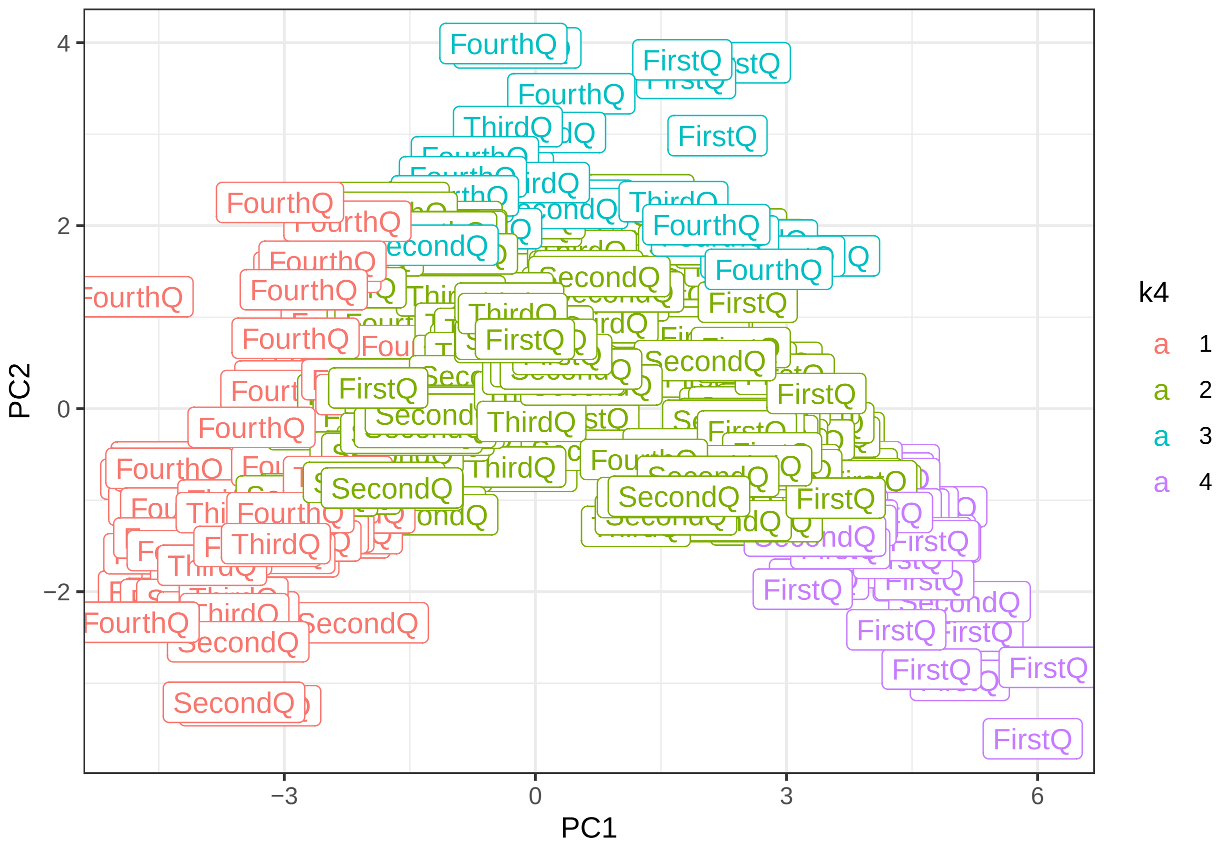
```r
# graphical analysis of pca data with clustering with 5 priniciple components
# create data frame with first 2 PCs
data_pca_clust <- data.frame(pca$x[, 1:2], # use 2 PCs even tho use 5 for clustering
                             # create factored columns with cluster assignment with 5 PCs
                             k2 = factor(cutree(hc_tune, 2)),
                             k3 = factor(cutree(hc_tune, 3)),
                             k4 = factor(cutree(hc_tune, 4)),
                             k5 = factor(cutree(hc_tune, 5)),
                             label = pca_adj$MedVal)

# plot k = 2
p4 <- ggplot(data_pca_clust,
       aes(x = PC1,
           y = PC2,
           color = k2,
           label = pca_adj$MedVal)) +
  geom_point() +
  geom_label() +
  theme_bw()
p4
```



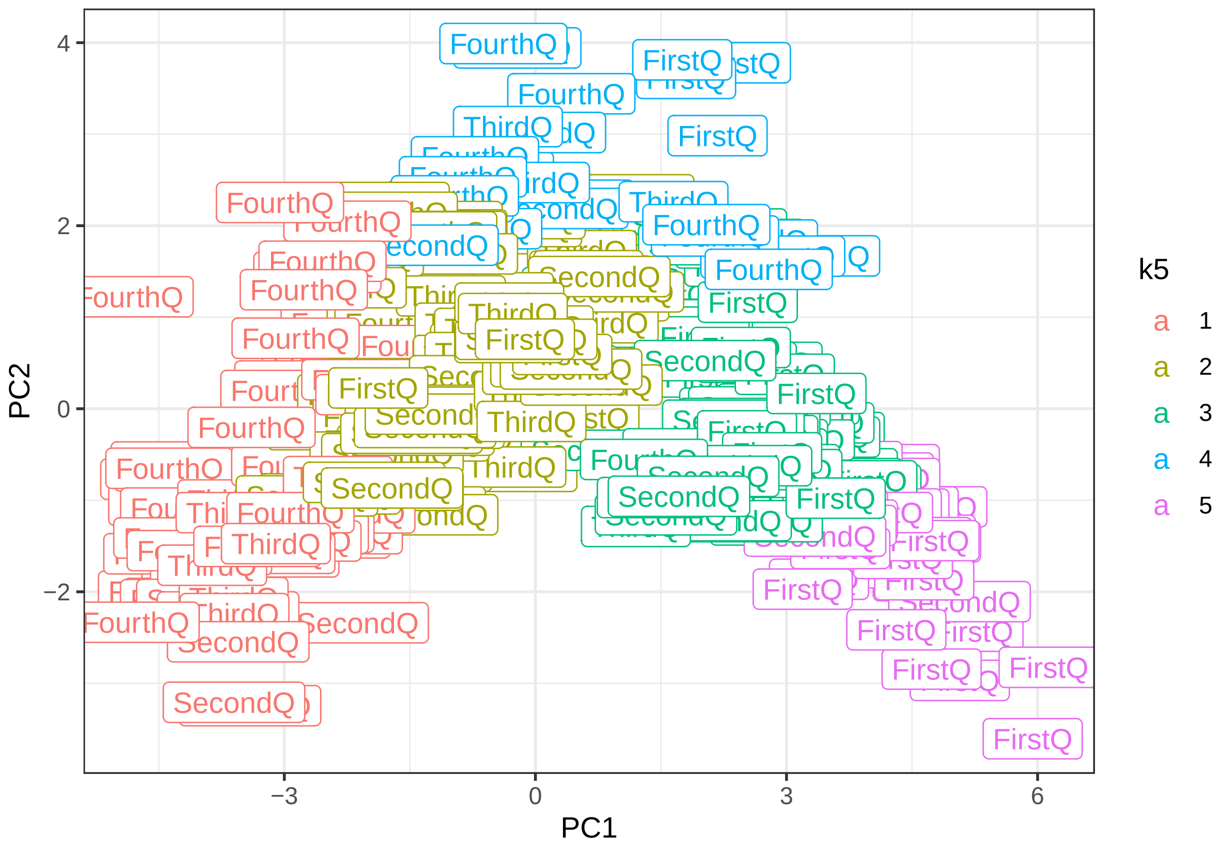
```r
# plot k = 3
p5 <- ggplot(data_pca_clust,
       aes(x = PC1,
           y = PC2,
           color = k3,
           label = pca_adj$MedVal)) +
  geom_point() +
  geom_label() +
  theme_bw()

p5
```



```r
# plot k = 4
p6 <- ggplot(data_pca_clust,
       aes(x = PC1,
           y = PC2,
           color = k4,
           label = pca_adj$MedVal)) +
  geom_point() +
  geom_label() +
  theme_bw()

p6
```



```r
# plot k = 5
p7 <- ggplot(data_pca_clust,
       aes(x = PC1,
           y = PC2,
           color = k5,
           label = pca_adj$MedVal)) +
  geom_point() +
  geom_label() +
  theme_bw()

p7
```


PC1
PC2
−3
0
3
6
4
2
0
−2
k5
a 1
a 2
a 3
a 4
a 5
FourthQ
ThirdQ
SecondQ
FirstQ