Analysis

Libraries and imports

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(foreign)
library(NbClust)
library(fpc)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```
library(ggrepel)
library(ggfortify)
library(rcompanion)
library(mclust)
```

```
## Package 'mclust' version 5.4.5
## Type 'citation("mclust")' for citing this R package in publications.
```

```
##
## Attaching package: 'mclust'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```
set.seed(44)

NBA.df <- read_csv("Data/season_stats_clean.csv")
```

```
## Parsed with column specification:
```

```
## cols(
##   .default = col_double(),
##   Player = col_character(),
##   Tm = col_character(),
##   Pos = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
names(NBA.df)[3] <- "Team"
```

Data cleanup

Why did we decide on these columns?

```
# set the columns to keep
cols.to.keep <- c("Player", "Pos", "Age", "Team", "Salary", "RPM",
    grep("_pg", names(NBA.df), value = TRUE))
NBA.df <- NBA.df[, cols.to.keep]
```

EDA

```
# set the columns to explore
cols.to.explore <- grep("_pg", names(NBA.df), value = TRUE)

# range of the data
diff(apply(NBA.df[, cols.to.explore], MARGIN = 2, FUN = range))
```

```
##         MP_pg    FG_pg   FGA_pg    3P_pg   3PA_pg    2P_pg  2PA_pg    FT_pg
## [1,] 36.75676 10.26667 23.96296 4.101266 9.987342 9.733333 19.2027 9.209877
##        FTA_pg   TRB_pg   AST_pg   STL_pg BLK_pg   PTS_pg
## [1,] 10.87654 14.12987 11.18519 2.026316      3 31.58025
```

```
# biavariate plot ggpairs(data = NBA.df, columns =
# cols.to.explore)

# bivariate plot of log values NBA.df %>%
# select(cols.to.explore) %>% mutate_all(log) %>% ggpairs()
```

Standardize and take the log of the data

```
# use Tukey Ladder of Powers to normalize the data
NBA.stdz <- lapply(NBA.df[, cols.to.explore], transformTukey,
    plotit = FALSE, quiet = TRUE) %>% as_tibble()

# scale the data
NBA.stdz <- scale(NBA.stdz) %>% as_tibble()

# add back in the orginal attributes
NBA.stdz <- bind_cols(NBA.df[, c("Player", "Pos", "Age", "Team",
    "Salary", "RPM")], NBA.stdz)
```

Log transform may not be neccessary

```
# NBA.stdz <- NBA.df # add some noise to remove 0s which will
# avoid -Inf after log transformation NBA.stdz[,
# cols.to.keep] <- apply(NBA.stdz[, cols.to.keep], MARGIN =
# 2, jitter, amount = 0.001) # apply log and scale NBA.stdz[,
# cols.to.keep] <- log(NBA.stdz[, cols.to.keep]) NBA.stdz[,
# cols.to.keep] <- scale(NBA.stdz[, cols.to.keep]) # remove
# NaNs from the cols.to.keep columns only then join back to
# original # data using a new index column NBA.stdz <-
# NBA.stdz %>% mutate(Index = row_number()) %>%
# select(c(cols.to.keep, 'Index')) %>% na.omit() %>%
# left_join(y = mutate(NBA.stdz[, c('Player', 'Pos', 'Age',
# 'Team')], Index = row_number()), by = 'Index') %>%
# select(-Index) ggpairs(data = NBA.stdz, columns =
# cols.to.keep)
```

Principle Component Analysis

```
# principle component analysis
NBA.pca <- princomp(NBA.stdz[, cols.to.explore])

# examine the pc
summary(NBA.pca)
```

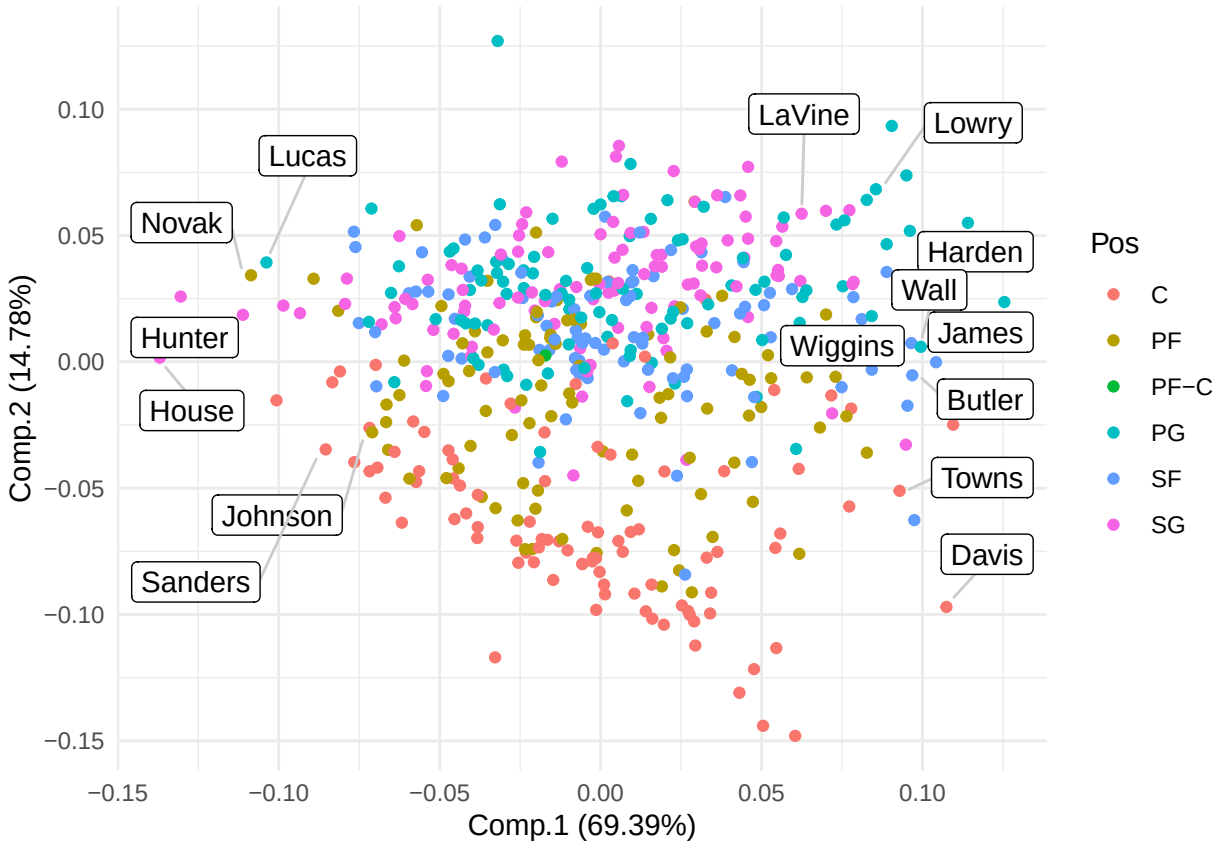
```
## Importance of components:
##                            Comp.1    Comp.2     Comp.3    Comp.4     Comp.5
## Standard deviation     3.1135909 1.4368535 0.83289292 0.7353899 0.60959406
## Proportion of Variance 0.6938883 0.1477718 0.04965292 0.0387081 0.02659794
## Cumulative Proportion  0.6938883 0.8416601 0.89131302 0.9300211 0.95661906
##                            Comp.6     Comp.7      Comp.8      Comp.9
## Standard deviation     0.47117328 0.45429656 0.277992475 0.221632791
## Proportion of Variance 0.01589014 0.01477221 0.005531368 0.003515884
## Cumulative Proportion  0.97250920 0.98728141 0.992812776 0.996328659
##                           Comp.10    Comp.11      Comp.12      Comp.13
## Standard deviation     0.138880602 0.12302549 0.1160643151 0.0479581179
## Proportion of Variance 0.001380542 0.00108332 0.0009641929 0.0001646231
## Cumulative Proportion  0.997709202 0.99879252 0.9997567143 0.9999213374
##                            Comp.14
## Standard deviation     0.0331513252
## Proportion of Variance 0.0000786626
## Cumulative Proportion  1.0000000000
```

```
NBA.pca$loadings
```

```
##
## Loadings:
##        Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9 Comp.10
## MP_pg   0.303         0.175                0.170  0.198  0.868  0.177
## FG_pg   0.313               -0.129  0.245        -0.153 -0.149  0.397
## FGA_pg  0.311               -0.132  0.226                      -0.457 -0.218
## 3P_pg   0.192  0.512  0.286 -0.221                      -0.176  0.206 -0.248
## 3PA_pg  0.188  0.528  0.241 -0.213 -0.103                      -0.312  0.514
## 2P_pg   0.293 -0.205 -0.145         0.312        -0.228 -0.131  0.268  0.556
## 2PA_pg  0.297 -0.153 -0.202         0.330        -0.148        -0.530 -0.114
## FT_pg   0.283        -0.369 -0.146 -0.522                             -0.281
```

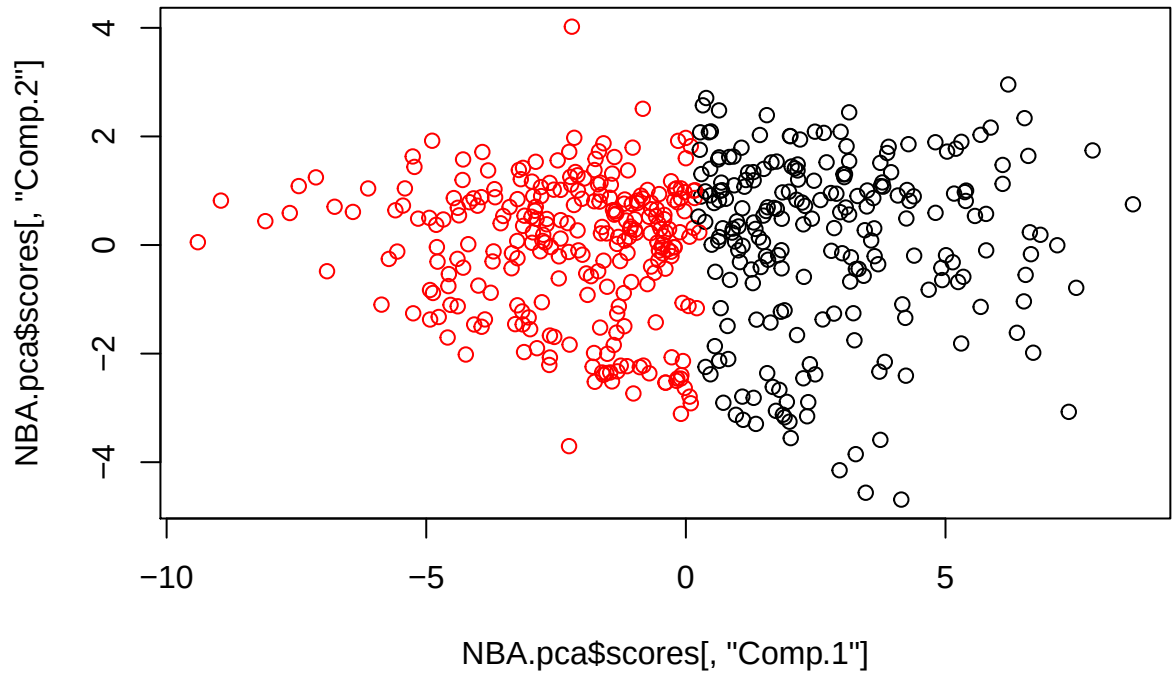
```
## FTA_pg  0.284 -0.121 -0.329 -0.135 -0.498                              0.327
## TRB_pg  0.234 -0.360  0.302                0.239  0.741 -0.328
## AST_pg  0.247  0.199 -0.218  0.564  0.152 -0.628  0.324
## STL_pg  0.251         0.193  0.685 -0.256  0.474 -0.302 -0.197
## BLK_pg  0.164 -0.439  0.588        -0.208 -0.525 -0.318
## PTS_pg  0.316               -0.154                      -0.106  0.302 -0.331
##        Comp.11 Comp.12 Comp.13 Comp.14
## MP_pg
## FG_pg  -0.108  -0.219  -0.216   0.712
## FGA_pg -0.245  -0.334  -0.538  -0.317
## 3P_pg           0.600  -0.209  -0.133
## 3PA_pg  0.122  -0.299   0.272   0.164
## 2P_pg   0.349   0.108  -0.149  -0.376
## 2PA_pg          0.479   0.350   0.258
## FT_pg   0.609  -0.151
## FTA_pg -0.606   0.210
## TRB_pg
## AST_pg
## STL_pg
## BLK_pg
## PTS_pg -0.206  -0.278   0.625  -0.368
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.071  0.071  0.071  0.071  0.071  0.071  0.071  0.071  0.071
## Cumulative Var  0.071  0.143  0.214  0.286  0.357  0.429  0.500  0.571  0.643
##                Comp.10 Comp.11 Comp.12 Comp.13 Comp.14
## SS loadings      1.000   1.000   1.000   1.000   1.000
## Proportion Var   0.071   0.071   0.071   0.071   0.071
## Cumulative Var   0.714   0.786   0.857   0.929   1.000
```

```r
# plot the clusters in PCA space
autoplot(NBA.pca, data = NBA.df, colour = "Pos") + geom_label_repel(aes(label = ifelse(MP_pg >=
    36 | MP_pg <= 3, as.character(sub("^\\S+\\s+", "", Player)),
    "")), box.padding = 0.35, point.padding = 0.5, segment.color = "grey80") +
    theme_minimal()
```

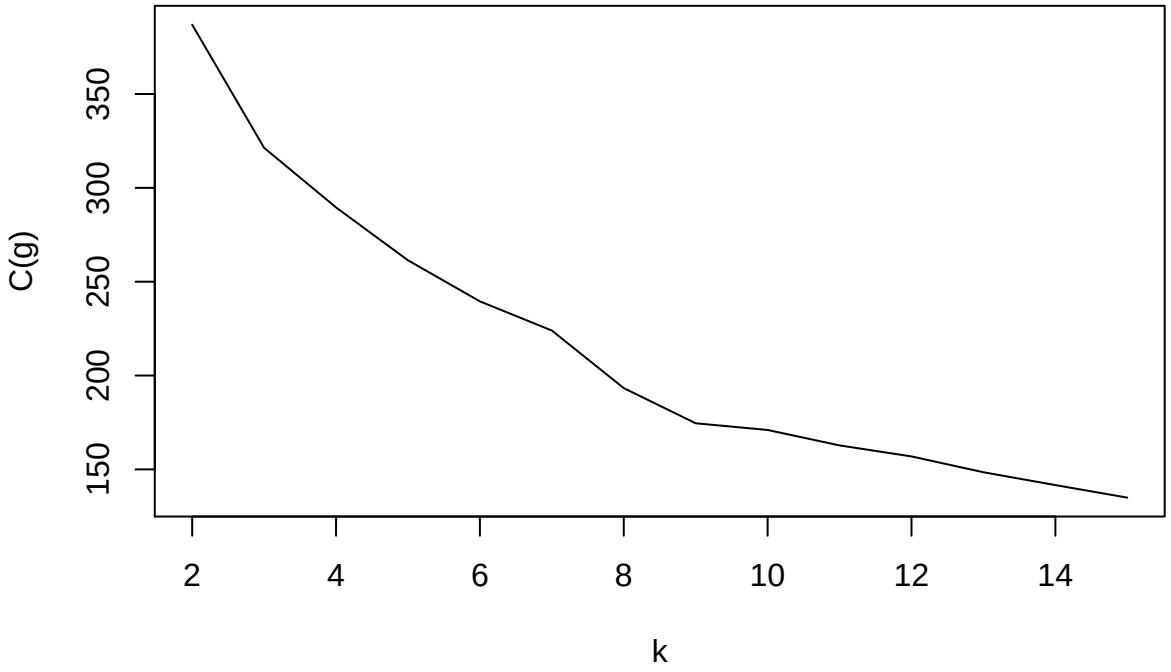


Clustering

```
# NbClust(data = NBA.stdz[, cols.to.keep], method =
# 'centroid', index = 'ch')

# is nbClust to find optimal k value under C(g)
km.clusts <- NbClust(data = NBA.stdz[, cols.to.explore], method = "kmeans",
    index = "ch")
plot(x = NBA.pca$scores[, "Comp.1"], y = NBA.pca$scores[, "Comp.2"],
    col = km.clusts$Best.partition)
```

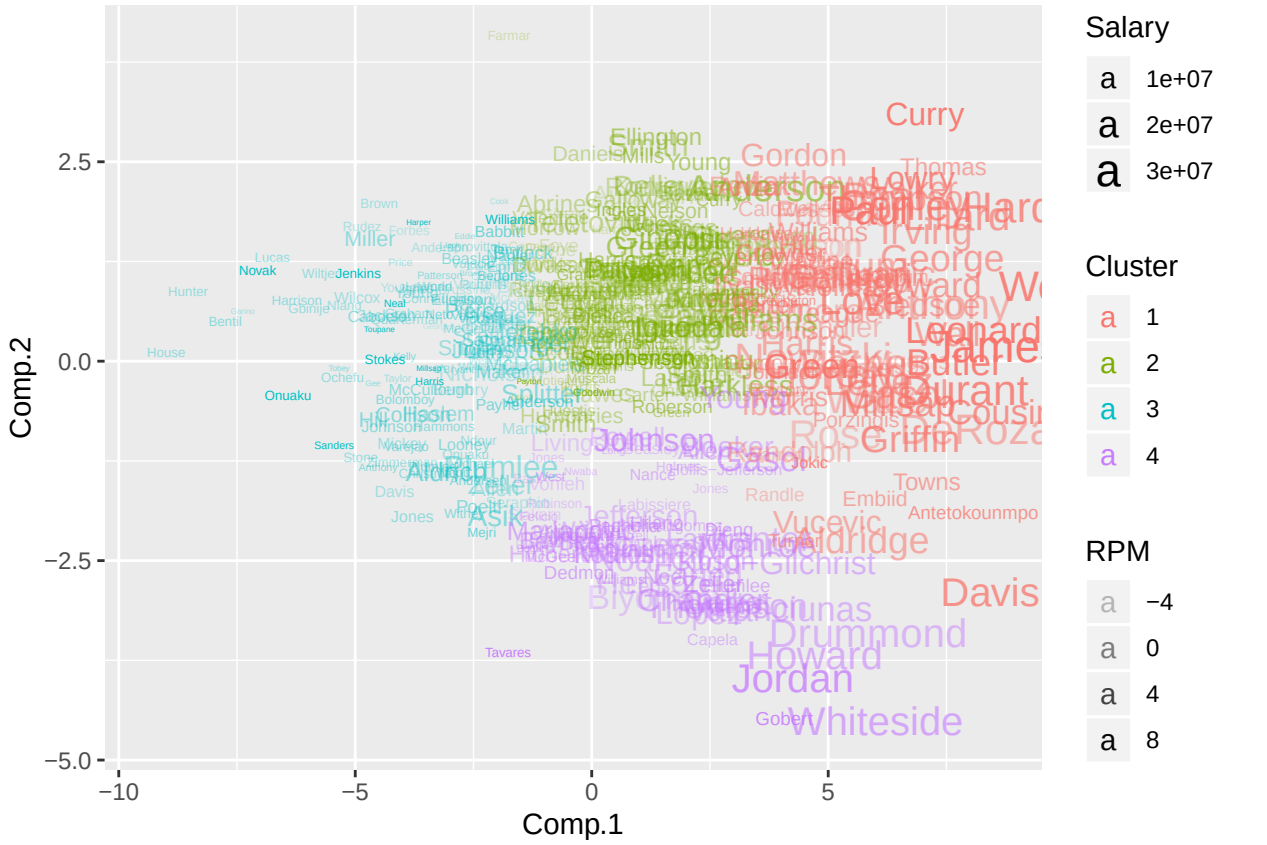



```
plot(x = names(km.clusts$All.index), y = km.clusts$All.index,
    xlab = "k", ylab = "C(g)", type = "l")
```

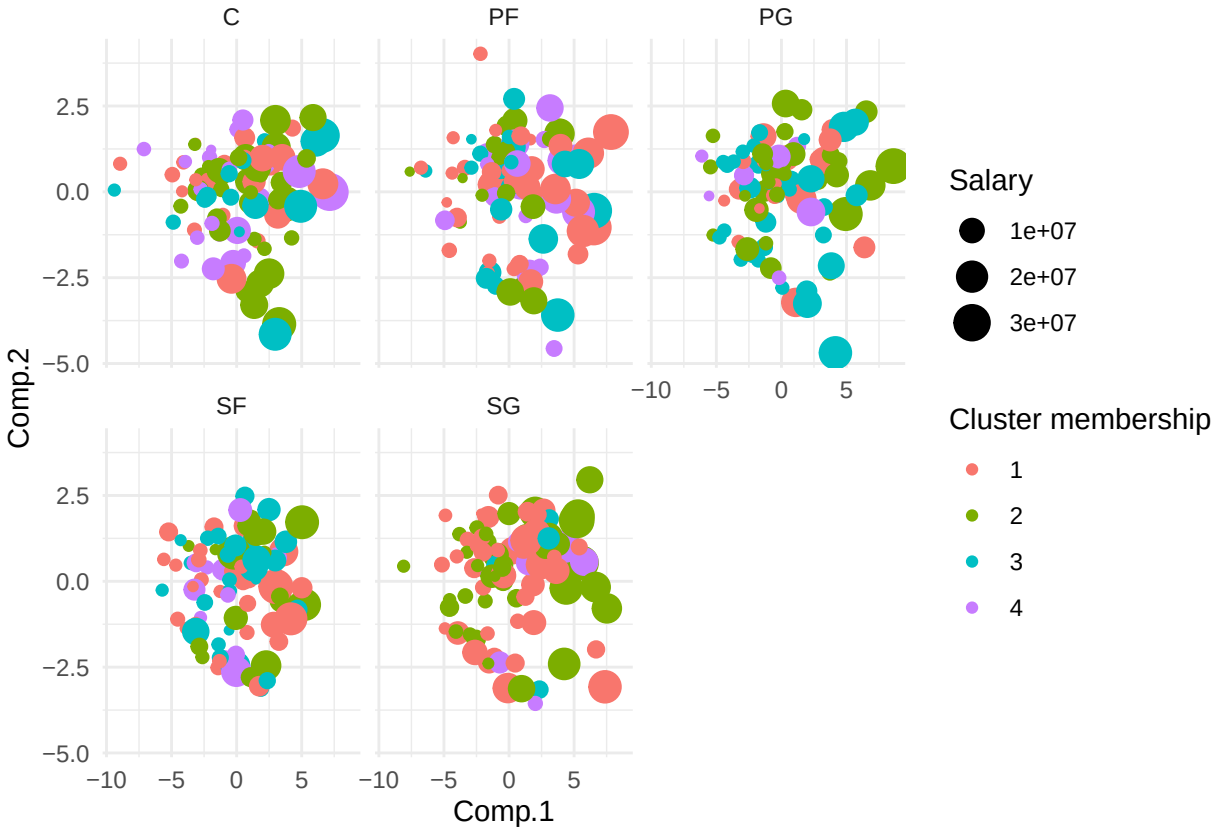



```
# re cluster using kmeans()
km.clusts <- kmeans(x = NBA.stdz[, cols.to.explore], centers = 4,
    nstart = 100)


# plot of clustering in PC space
NBA.stdz %>% cbind(cluster = km.clusts$cluster, NBA.pca$scores[,
    1:2]) %>% mutate(Cluster = as.factor(cluster), Player = sub("^\\S+\\s+",
```

```
    "", Player)) %>% ggplot(aes(x = Comp.1, y = Comp.2)) + geom_text(aes(label = Player,
    col = Cluster, size = Salary, alpha = RPM), hjust = 0, vjust = 0)
```



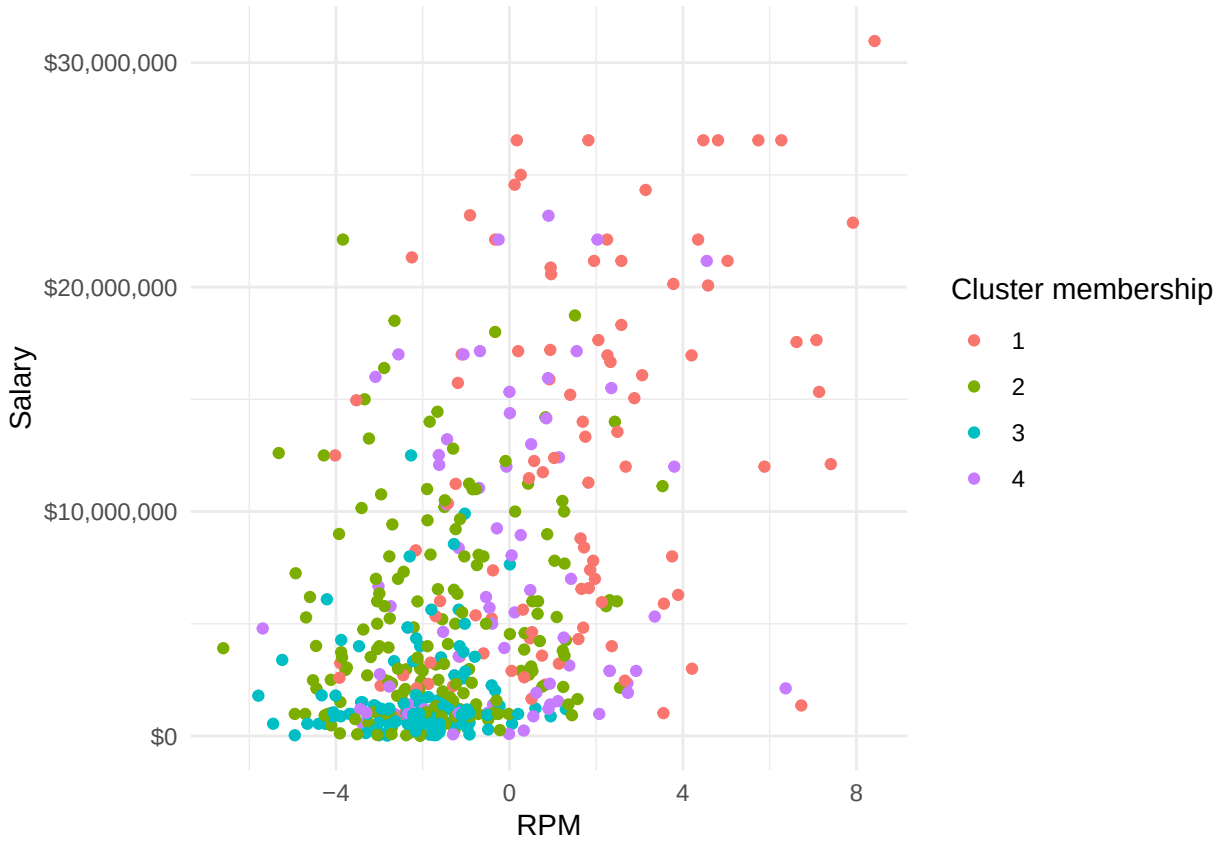

```
# model based clustering
mc.clusts <- NBA.stdz %>% na.omit() %>% mutate(Pos = if_else(Pos ==
    "PF-C", "PF", Pos)) %>% select(c("Pos", cols.to.explore)) %>%
    group_by(Pos) %>% nest() %>% mutate(Cluster = lapply(data,
    function(df) {
        Mclust(data = df, G = 4)$classification
    })) %>% unnest(cols = c(data, Cluster)) %>% ungroup()

# plot it
mc.clusts %>% # merge with Player and PCA scores but remove the NA rows
bind_cols(bind_cols(NBA.stdz, as_tibble(NBA.pca$scores[, 1:2]))) %>%
    na.omit() %>% select(Player, Salary, RPM, Comp.1, Comp.2)) %>%
    mutate(Cluster = as.factor(Cluster)) %>% ggplot(aes(x = Comp.1,
    y = Comp.2, color = Cluster, size = Salary, text = paste("Player:",
        Player))) + geom_point() + labs(color = "Cluster membership") +
    facet_wrap(~Pos) + theme_minimal()
```




```
# interactive plot plotly::ggplotly()
```

Post-hoc analyses

```
# scatterplot of RPM vs. Salary
NBA.df %>% mutate(Cluster = as.factor(km.clusts$cluster)) %>%
    na.omit() %>% ggplot(aes(x = RPM, y = Salary, color = Cluster,
    text = paste("Player:", Player))) + geom_point() + scale_y_continuous(labels = scales::dollar) +
    labs(col = "Cluster membership") + theme_minimal()
```




```
# interactive plot plotly::ggplotly()
```