DAR F23 Project Status Notebook

Hockey Analytics

Liebin Zhang

2023-10-02

Contents

Weekly Work Summary

NOTE: Follow an outline format; use bullets to express individual points.

- RCS ID: zhangl38
- Project Name: Hockey Analytics
- Summary of work since last week
  - I dropped the feature of goal and then I build k-means elbow and visualizes means. After this I build Heatmap and visualize the pca graph without goal.
- NEW: Summary of github issues added and worked
  - Issues that you've submitted
  - Issues that you've self-assigned and addressed
- Summary of github commits
  - include branch name(s)
  - include browsable links to all external files on github
  - Include links to shared Shiny apps
- List of presentations, papers, or other outputs
  - Include browsable links
- List of references (if necessary)
- Indicate any use of group shared code base
- Indicate which parts of your described work were done by you or as part of joint efforts
- **Required:** Provide illustrating figures and/or tables

Personal Contribution

- Clearly defined, unique contribution(s) done by you: code, ideas, writing…
- Include github issues you've addressed

Besides the function codes are contained in the file `AnalysisCodeFunc.R`. in my data preparation step that imports as a helper script that defines various functions used through this notebook for data processing and analysis, all other ideas, codes, and writings are all done myself.

Analysis: Question 1

How to build k-means elbow and visualizes means and build Heatmap without goal?

Data Preparation

Setup: Define functions The major functions of this notebook are contained in the file `AnalysisCodeFunc.R`. This code chunk *sources* (imports) a helper script that defines various functions used through this notebook for data processing and analysis.

```
# All user-defined functions are contained in the following helper script file.
source("../../AnalysisCodeFunc.R")
```

Setting program parameters This section sets the dimensions of the data structures used in this notebook, based on the captured video.

```
# Size of rink image and of all plots
xsize <- 2000
ysize <- 850

# FPS of the video
fps <- 29.97

# Coordinates to the goal pipes
pipes_x <- 1890
lpipe_y <- 395
rpipe_y <- 455
```

Data preparation for predictive modelling Based on our settings, this section reads in the captured image data.

The code is highly dependent upon the following directory structure:

- The file path determined by the `filepath` variable contains folders named with the game number, followed by 'p', followed by the period number
- Each `period` folder contains a folder named `Sequences`.
- Each `Sequences` folder contains `sequence_folders` that contain all the relevant sequence data.

```
# Set filepaths and read the rink
# Results suppressed because this can be messy...

# This file path should contain the hockey rink images and all the sequences
filepath <- '../../FinalGoalShots/'

# See above for explanation of file path syntax
games <- c(24, 27, 33, 34)
# Only take the first and third periods. These are when the opposing team shoots on our goal. Our shots
periods <- map(games, ~ str_c(., 'p', c(1, 3))) %>% unlist

# Get the 'Sequences' folder for every period
period_folders <- map(periods, ~ {
  str_c(filepath, ., '/Sequences')
})
```

```r
# Get every folder inside each 'Sequences' folder
sequence_folders <- period_folders %>%
  map(~ str_c(., '/', list.files(.))) %>%
  unlist

# Read the rink images and format them to a raster used for graphing
rink_raster <- makeRaster(filepath, 'Rink_Template.jpeg')
half_rink_raster <- makeRaster(filepath, 'Half_Rink_Template.jpeg')

# As every folder is run through the `combinePasses` function, the info.csv file in each sequence folde
info <- matrix(0, nrow = 0, ncol = 4) %>%
  data.frame %>%
  set_names(c('possessionFrame', 'shotFrame', 'outcome', 'rightHanded'))

# Read in all the sequences
# NOTE: This step takes a long time (minutes)
sequences = sequence_folders %>% map(combinePasses)

# Change outcomes to more verbose names
info$outcome %<>% fct_recode(Goal = 'G', Save = 'GB', 'Defender Block' = 'DB', Miss = 'M')
```

Shot statistics retrieval This section constructs the dataframe used to predict if shots are successful.

```r
# Get stats for the shot in every sequence
shots_stats.df <- seq_along(sequences) %>%
  map_dfr(goalShotStats) %>%
  # Some models can't use logical data
  mutate_if(is.logical, as.factor)
```

We first combine the shots data with the outcomes vector;

```r
# Split data into training and validation sets
outcomes.goal <- (info$outcome == 'Goal') %>% as.numeric %>% as.factor

# Append to shots_stats.df
shots_stats_goal.df <- cbind(shots_stats.df, outcomes.goal)

# Save this dataframe on the file system in case we want to simply load it later (to save time)
saveRDS(shots_stats_goal.df, "shots_stats_goal.df.Rds")
```

Now let's do some basic classification analysis on the `shots_stats_goal.df` dataset!

We'll use `tidymodels`, part of the **tidyverse**, to split the data into an 80% train/20% test split.

```r
#Create training set
set.seed(100)

# Type ?initial_split , ?training , or ?testing in the R console to see how these work!
hockey_split <- initial_split(shots_stats_goal.df, prop = 0.8)
hockeyTrain <- training(hockey_split)
hockeyTest <- testing(hockey_split)

# Check how many observations for each split we have
nrow(hockeyTrain)
```

```
## [1] 84
```

```
nrow(hockeyTest)
```

```
## [1] 21
```

```
# How many features are there
ncol(hockeyTrain)
```

```
## [1] 12
```

Analysis: Methods and results

Cluster the unsuccessful shots Step 1: I use new matrix named shotsNum to take off the goals(outcomes) and change the True/False feature to factor Step 2: Using shotsNum to cluster the unsuccessful shots as HockeyTrain.m and prepare to build k-means elobow

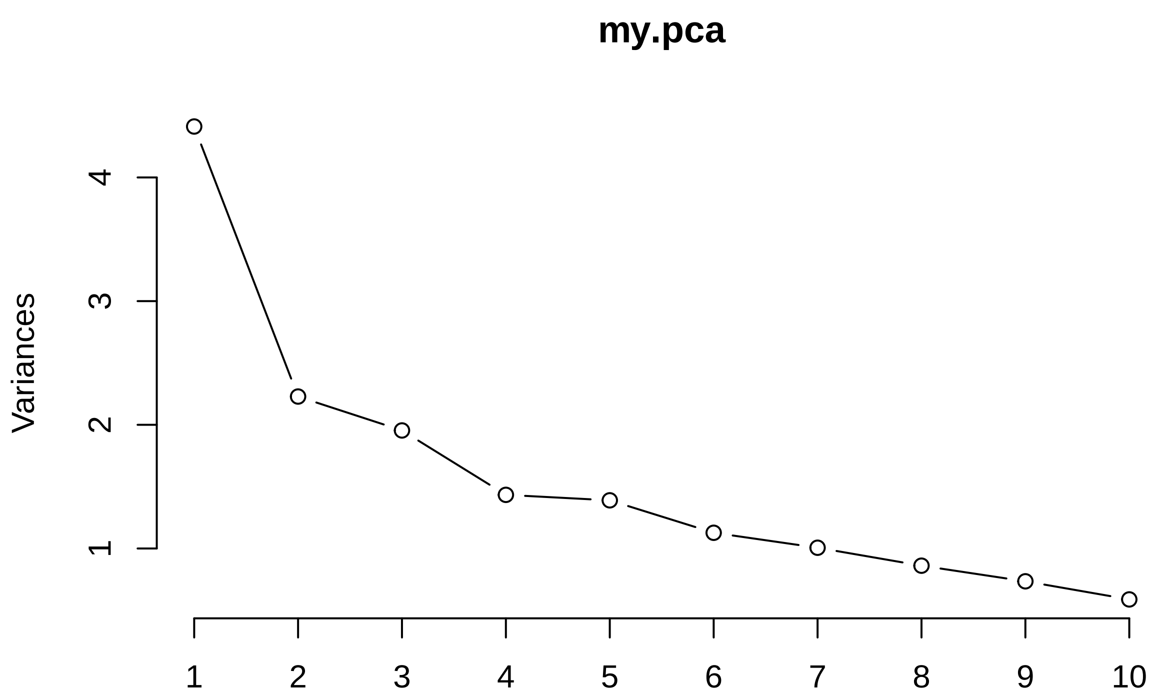
```
shots <- readRDS("shots_stats_goal.df.Rds")#Need to make sure this is grabb
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose

## The following objects are masked from 'package:reshape2':
##
##     dcast, melt
```

```
library(mltools)
```

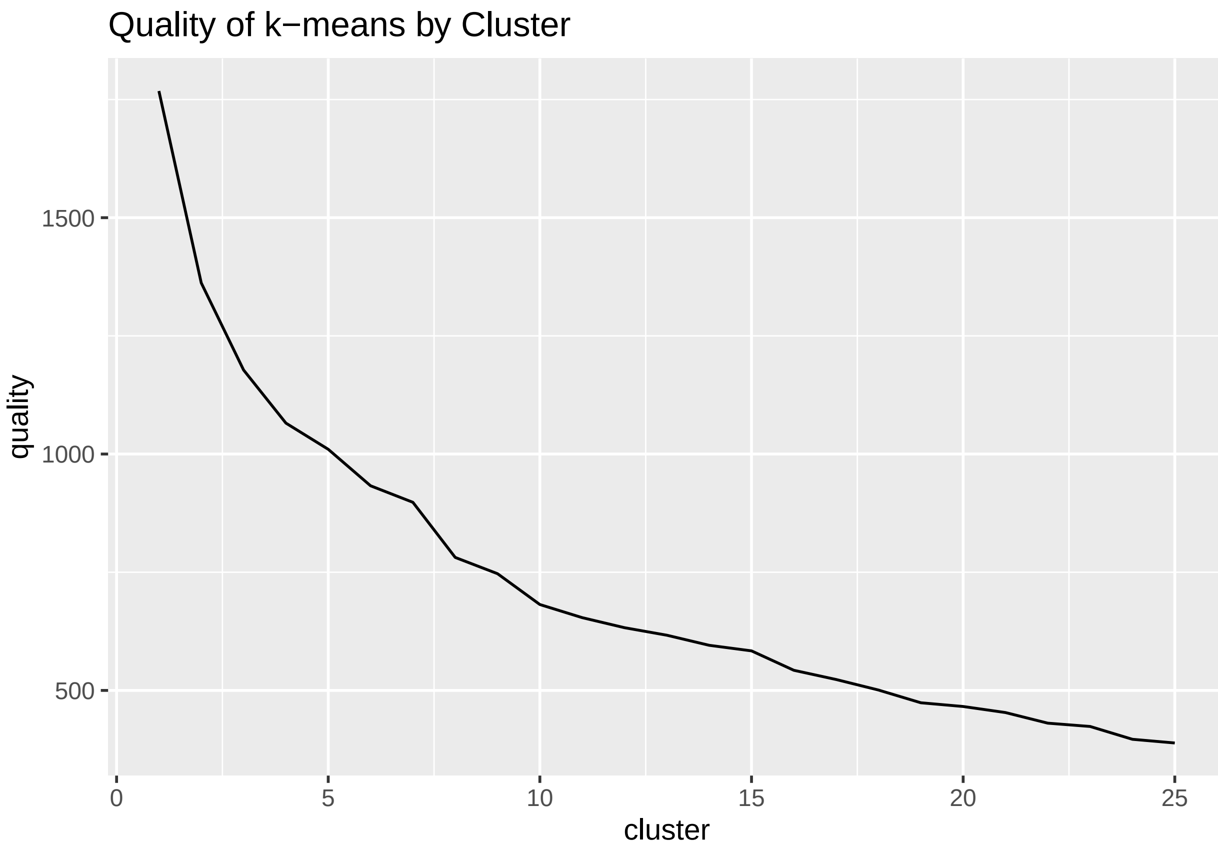
```
##
## Attaching package: 'mltools'

## The following objects are masked from 'package:yardstick':
##
##     mcc, rmse

## The following object is masked from 'package:tidyr':
##
##     replace_na
```

```
shotsNum <- shots
shotsNum$goalieScreened = as.factor(shotsNum$goalieScreened)
shotsNum$oppDefenders = as.factor(shotsNum$oppDefenders)
shotsNum$sameDefenders = as.factor(shotsNum$sameDefenders)
shotsNum <- shotsNum[,1:11]
shotsNum <- one_hot(dt = as.data.table(shotsNum))

h.df <- shotsNum %>%
mutate_all(as.numeric)%>% mutate_all(scale)
hockeyTrain.m <- as.matrix(h.df)

my.pca<-prcomp(hockeyTrain.m,retx=TRUE)
plot(my.pca, type="line")
```



Build k-means elbow Step 1: using wssplot and cluster set(HockeyTrain.m) to build k-means by cluster Step2: Analysis k-means elbow which is 10

```
wssplot <- function(data, nc=25, seed=20){
wss <- data.frame(cluster=1:nc, quality=c(0))
for (i in 1:nc){
set.seed(seed)
wss[i,2] <- kmeans(data, centers=i)$tot.withinss}
ggplot(data=wss,aes(x=cluster,y=quality)) +
geom_line() +
ggtitle("Quality of k-means by Cluster")
}
wssplot(hockeyTrain.m, nc=25)
```



Visualizes means and build Heatmap without goal Step 1: Visualizes means in 7 clusters which clusters of sizes are 4, 15, 15, 3, 32, 14, 22 Step 2: Using ggplot to visualize my cluster means. Step 3: Analysis Heatmap: The high relavent in heatmap is pcudistance in cluster 2 and cluster 5, puckAngle has high respective in cluster 1 and cluster 1 and 3. Result: I plan to analysis puck distance influence in Hockey game in the future analysis.

```r
library("gplots")
```

```
##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

```r
library("ggbiplot")
```

```
## Loading required package: plyr

## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
```

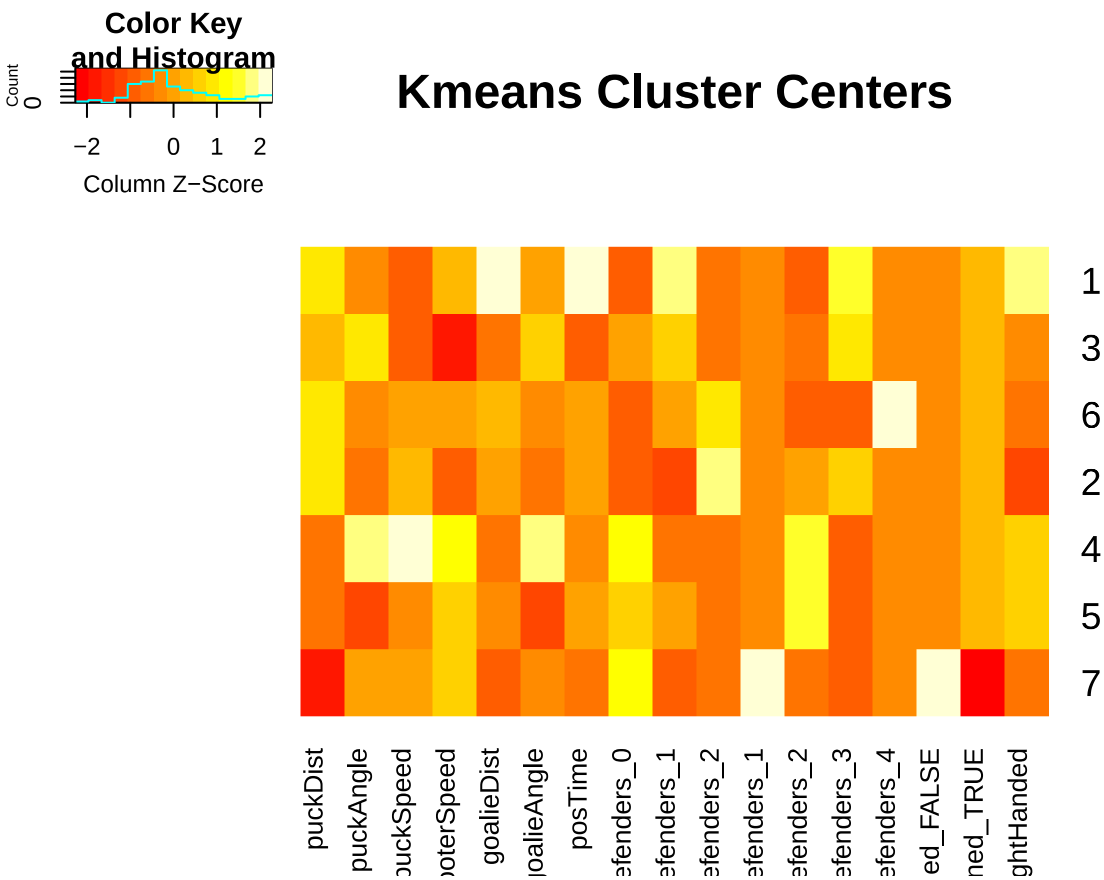
```
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
## The following object is masked from 'package:purrr':
##
##     compact
```

```
km <- kmeans(hockeyTrain.m,7)
km
```

```
## K-means clustering with 7 clusters of sizes 5, 8, 16, 14, 23, 7, 32
##
## Cluster means:
##      puckDist  puckAngle   puckSpeed shooterSpeed goalieDist goalieAngle
## 1  0.65571693 -0.24690762 -0.63196744   0.09731089  1.7207458 -0.04520701
## 2  0.66195505 -0.36349023  0.07699793  -0.42900536  0.2382595 -0.40360431
## 3  0.47611274  0.54487300 -0.53062681  -0.82596040 -0.1414694  0.45355726
## 4 -0.01427667  1.04652972  1.16360646   0.44081510 -0.2482893  1.09878749
## 5 -0.05530758 -0.73814532 -0.12336714   0.20077936  0.1541647 -0.63049476
## 6  0.69792871 -0.24926116 -0.04197131  -0.11755071  0.5126830 -0.15074069
## 7 -0.61267444 -0.01577355 -0.06641764   0.19357416 -0.3720254 -0.11339085
##       posTime sameDefenders_0 sameDefenders_1 sameDefenders_2 oppDefenders_1
## 1  1.12176689     -1.19504329       1.5376484       -0.357496     -0.6441500
## 2  0.13075140     -1.19504329      -0.6441500        2.770594     -0.6441500
## 3 -0.27352173     -0.43609443       0.7194740       -0.357496     -0.5077876
## 4 -0.05577616      0.53969697      -0.3324645       -0.357496     -0.6441500
## 5  0.13858983      0.03687371       0.2095972       -0.357496     -0.6441500
## 6  0.20645506     -1.19504329       0.2909064        1.429984     -0.6441500
## 7 -0.19157448      0.70232887      -0.5077876       -0.357496      1.4012860
##   oppDefenders_2 oppDefenders_3 oppDefenders_4 goalieScreened_FALSE
## 1    -0.84524662      1.8791655     -0.2659855           -0.6737717
## 2    -0.08884695      0.9768223     -0.2659855           -0.6737717
## 3    -0.59311339      1.4279939     -0.2659855           -0.6737717
## 4     1.17181917     -0.5270830     -0.2659855           -0.6737717
## 5     1.17181917     -0.5270830     -0.2659855           -0.5805621
## 6    -0.84524662     -0.5270830      3.7237973           -0.6737717
## 7    -0.71918000     -0.5270830     -0.2659855            1.4700473
##   goalieScreened_TRUE rightHanded
## 1           0.6737717  1.06394210
## 2           0.6737717 -0.68158791
## 3           0.6737717 -0.05818433
## 4           0.6737717  0.35148087
## 5           0.5805621  0.37006682
## 6           0.6737717 -0.36098035
## 7          -1.4700473 -0.30754576
##
## Clustering vector:
##   [1] 5 5 7 4 5 2 7 2 6 5 5 7 3 3 5 6 7 5 2 7 7 4 3 7 6 2 7 7 7 2 5 3 4 2 5 7 7
##  [38] 7 4 6 7 3 5 3 3 2 6 5 5 5 7 5 7 4 2 7 7 5 7 4 5 5 4 3 1 7 4 7 3 7 4 6 5 7
##  [75] 3 3 3 5 3 7 3 7 3 7 4 4 4 3 5 7 1 1 7 4 6 7 5 1 7 5 7 5 7 4 1
##
## Within cluster sum of squares by cluster:
## [1]  26.15541  40.63992 136.97536 103.88666 191.65345  55.64951 300.12181
##  (between_SS / total_SS =  51.6 %)
```

```
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
par(mar = c(1, 1, 1, 1))
heatmap.2(km$centers,
scale = "column",
dendrogram = "column",
Colv=FALSE,
cexCol=1.0,
main = "Kmeans Cluster Centers", trace ="none")
```



Discussion of results

Based on the analysis that I performed, according to the quality of k mean by cluster graph, we can easily see that the elbow was found at 7. So I choose to divide the data set into 7 different clusters, and then visualize them in the form of heatmap. By taking a look into the heatmap, I got the conclusion that the high relavent in heatmap is pcudistance in cluster 2 and cluster 5, puckAngle has high respective in cluster 1 and cluster 1 and 3.

Analysis: Question 2

How to visualize clusters with PCA without goal?

Data Preparation

I use the same data set as Analysis Question 1.

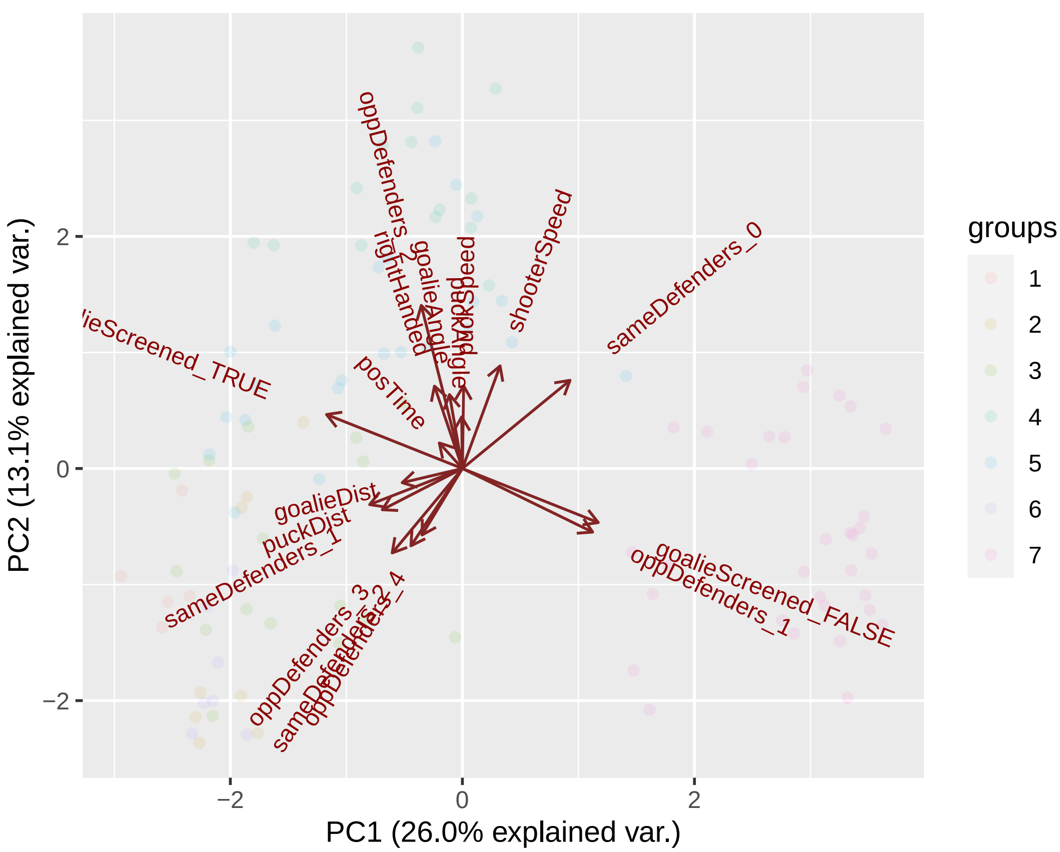
Analysis: Methods and Results

```
# Create a biplot for PC1 and PC2 colored by cluster
t<-1.2*max(abs(my.pca$x[,1:2]))

p <- ggbiplot(my.pca,
choices=c(1,2),
alpha=.1,
scale = 0,
groups=as.factor(km$cluster))
p <- p + scale_colour_hue()
ggtitle('Biplot of PC1 and PC2')+
xlim(-t,t) + ylim(-t,t)
```

Visualize clusters with PCA without goal

```
## NULL
```

```
p
```



Discussion of results

According to the pca graph, we can see that there are a lot of arrows points toward the point sets. For example, “goalie_screen_FALSE” and “oppDefender_1” are pointing into the group 3 meaning that these two features explain the most behaviors in that group. And also, the features of “puckedAngle”, “rightHanded”, and “goalieAngle” are pointing into the group 6 meaning that these features explain the most behaviors inside group 6.

Summary and next steps

For the next step, I am going to use the given function in analysis code to determine the distance of each player and how they reach the goal.