PCA_Steps

Sachith M Gunawardane

2023-06-04

```
library(FactoMineR)
```

```
## Warning: package 'FactoMineR' was built under R version 4.2.3
```

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.2.3

## Loading required package: ggplot2

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
#prcomp() # base R
#princomp() #base R
#?PCA #base FactoMineR / factoextra
```

```
#?decathlon
data(decathlon)
head(decathlon)
```

```
##          100m Long.jump Shot.put High.jump  400m 110m.hurdle Discus Pole.vault
## SEBRLE  11.04      7.58    14.83      2.07 49.81       14.69  43.75       5.02
## CLAY    10.76      7.40    14.26      1.86 49.37       14.05  50.72       4.92
## KARPOV  11.02      7.30    14.77      2.04 48.37       14.09  48.95       4.92
## BERNARD 11.02      7.23    14.25      1.92 48.93       14.99  40.87       5.32
## YURKOV  11.34      7.09    15.19      2.10 50.42       15.31  46.26       4.72
## WARNERS 11.11      7.60    14.31      1.98 48.68       14.23  41.10       4.92
##         Javeline 1500m Rank Points Competition
## SEBRLE     63.19 291.7    1   8217    Decastar
## CLAY       60.15 301.5    2   8122    Decastar
## KARPOV     50.31 300.2    3   8099    Decastar
## BERNARD    62.77 280.1    4   8067    Decastar
## YURKOV     63.44 276.4    5   8036    Decastar
## WARNERS    51.77 278.1    6   8030    Decastar
```

```
# Check for Event
unique(decathlon$Competition)
```

```
## [1] Decastar OlympicG
## Levels: Decastar OlympicG
```

```
dim(decathlon)
```

```
## [1] 41 13
```

Rank and Points are response variables (prediction)

```
#dropping last 3 variables which are predictions
decathlon1 <- decathlon[,1:10]
#get correlation between first 10 variables
cor(decathlon1)
```

```
##                    100m   Long.jump    Shot.put   High.jump         400m
## 100m         1.00000000 -0.59867767 -0.35648227 -0.24625292  0.520298155
## Long.jump   -0.59867767  1.00000000  0.18330436  0.29464444 -0.602062618
## Shot.put    -0.35648227  0.18330436  1.00000000  0.48921153 -0.138432919
## High.jump   -0.24625292  0.29464444  0.48921153  1.00000000 -0.187956928
## 400m         0.52029815 -0.60206262 -0.13843292 -0.18795693  1.000000000
## 110m.hurdle  0.57988893 -0.50541009 -0.25161571 -0.28328909  0.547987756
## Discus      -0.22170757  0.19431009  0.61576810  0.36921834 -0.117879365
## Pole.vault  -0.08253683  0.20401411  0.06118185 -0.15618074 -0.079292469
## Javeline    -0.15774645  0.11975893  0.37495551  0.17188009  0.004232096
## 1500m       -0.06054645 -0.03368613  0.11580306 -0.04490252  0.408106432
##              110m.hurdle     Discus   Pole.vault     Javeline       1500m
## 100m         0.579888931 -0.2217076 -0.082536834 -0.157746452 -0.06054645
## Long.jump   -0.505410086  0.1943101  0.204014112  0.119758933 -0.03368613
## Shot.put    -0.251615714  0.6157681  0.061181853  0.374955509  0.11580306
## High.jump   -0.283289090  0.3692183 -0.156180742  0.171880092 -0.04490252
## 400m         0.547987756 -0.1178794 -0.079292469  0.004232096  0.40810643
## 110m.hurdle  1.000000000 -0.3262010 -0.002703885  0.008743251  0.03754024
## Discus      -0.326200961  1.0000000 -0.150072400  0.157889799  0.25817510
## Pole.vault  -0.002703885 -0.1500724  1.000000000 -0.030000603  0.24744778
## Javeline     0.008743251  0.1578898 -0.030000603  1.000000000 -0.18039313
## 1500m        0.037540240  0.2581751  0.247447780 -0.180393128  1.00000000
```

Most of the values are correlated shot.put and discus is having 0.6157681 correlation ship. therefore it may be possible to represent this trend by 1 of it. And reduce the dimensionality of the data set

```
cov(decathlon1)
```

```
##                     100m  Long.jump    Shot.put    High.jump        400m
## 100m         0.069181098 -0.0498225 -0.07730085 -0.005761341  0.15785018
## Long.jump   -0.049822500  0.1001100  0.04781500  0.008292500 -0.21972500
## Shot.put    -0.077300854  0.0478150  0.67968122  0.035875488 -0.13164098
## High.jump   -0.005761341  0.0082925  0.03587549  0.007912195 -0.01928439
## 400m         0.157850183 -0.2197250 -0.13164098 -0.019284390  1.33044878
## 110m.hurdle  0.071959207 -0.0754450 -0.09786744 -0.011888476  0.29820695
## Discus      -0.196976280  0.2076700  1.71478433  0.110935732 -0.45927896
## Pole.vault  -0.006035122  0.0179450  0.01402232 -0.003862073 -0.02542585
## Javeline    -0.200269329  0.1828975  1.49208476  0.073796402  0.02356220
## 1500m       -0.185897744 -0.1244175  1.11445963 -0.046624146  5.49495579
##               110m.hurdle     Discus    Pole.vault     Javeline        1500m
## 100m         0.0719592073 -0.1969763 -0.0060351220  -0.20026933  -0.18589774
## Long.jump   -0.0754450000  0.2076700  0.0179450000   0.18289750  -0.12441750
## Shot.put    -0.0978674390  1.7147843  0.0140223171   1.49208476   1.11445963
## High.jump   -0.0118884756  0.1109357 -0.0038620732   0.07379640  -0.04662415
## 400m         0.2982069512 -0.4592790 -0.0254258537   0.02356220   5.49495579
## 110m.hurdle  0.2225848780 -0.5198437 -0.0003546341   0.01991049   0.20674573
## Discus      -0.5198436585 11.4098352 -0.1409240244   2.57427463  10.17995195
## Pole.vault  -0.0003546341 -0.1409240  0.0772839024  -0.04025646   0.80300780
## Javeline     0.0199104878  2.5742746 -0.0402564634  23.29819305 -10.16419043
## 1500m        0.2067457317 10.1799520  0.8030078049 -10.16419043 136.26470061
```

```
round(diag(cov(decathlon1)),4) # Variances for each variable
```

```
##        100m   Long.jump    Shot.put   High.jump        400m 110m.hurdle
##      0.0692      0.1001      0.6797      0.0079      1.3304      0.2226
##      Discus  Pole.vault    Javeline       1500m
##     11.4098      0.0773     23.2982    136.2647
```

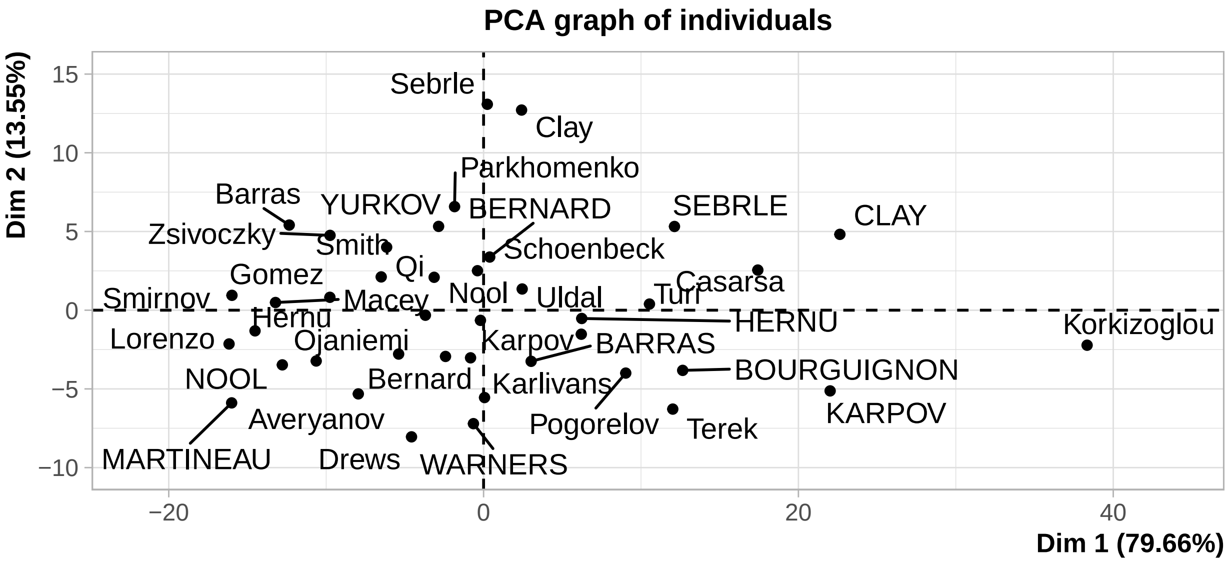
Variance of 1500m is 136.2647 which is high compared to others. Hence it is best to use correlation matrix to extract P.C. to avoid bias

Total variance is 173.4599 (sum(round(diag(cov(decathlon1)),4))) and Javeline and 1500m dominate biggest percentage

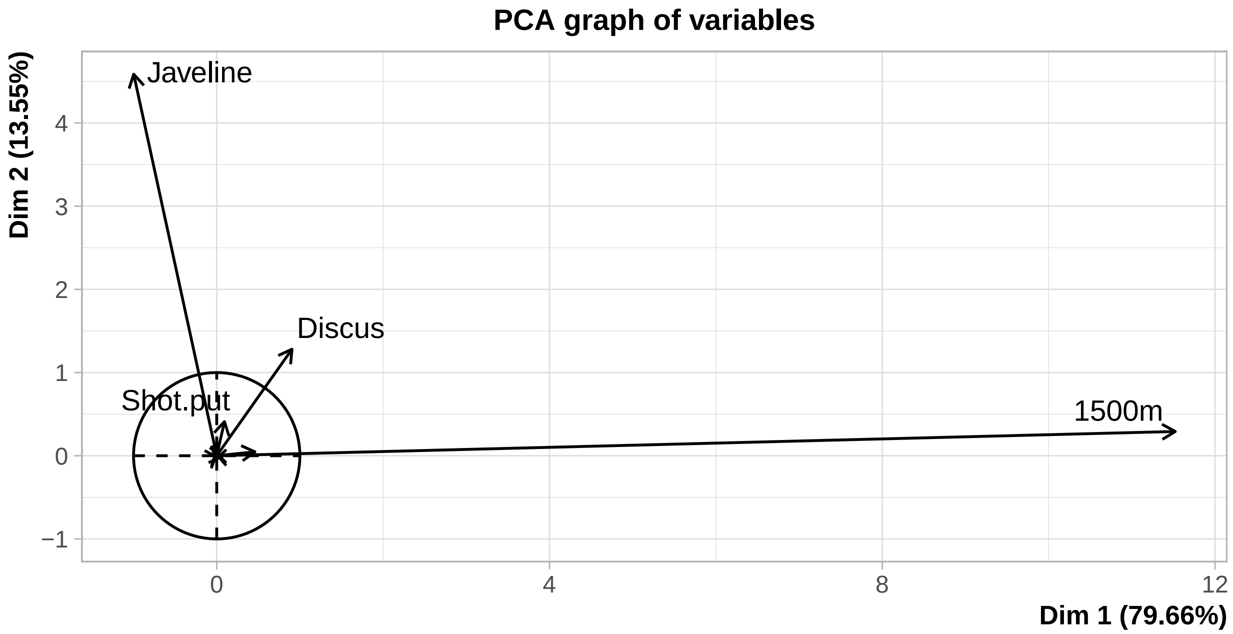
PCA using covariance Matrix

```
pca.out <- PCA(decathlon1, scale.unit = F)
```

```
## Warning: ggrepel: 4 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
## Warning: ggrepel: 6 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
pca.out
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 41 individuals, described by 10 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

```
pca.out$eig
```

```
##            eigenvalue percentage of variance cumulative percentage of variance
## comp 1  1.348073e+02            79.659589641                          79.65959
## comp 2  2.293556e+01            13.552956464                          93.21255
## comp 3  9.747263e+00             5.759799777                          98.97235
## comp 4  1.117215e+00             0.660178830                          99.63252
## comp 5  3.477705e-01             0.205502637                          99.83803
## comp 6  1.326819e-01             0.078403653                          99.91643
## comp 7  6.208630e-02             0.036687700                          99.95312
## comp 8  4.938498e-02             0.029182305                          99.98230
## comp 9  2.504308e-02             0.014798320                          99.99710
## comp 10 4.908785e-03             0.002900673                         100.00000
```

1st Column - Eigen Values 2nd Column - Percentage of Variance explain by each component 3rd Column - cumulative percentage

First 3 component captures 98.97 of the variance

#Check whether all eigen values get accumulated to total of variance Tr(A)

```
sum(pca.out$eig[,1])
```

```
## [1] 169.2292
```

Total variability tr(A) is 173.4599. Based on above we can see small percentage is missing from eigen values. This can be due to PCA function and number of components that it extract. But as per theory max eigen values are 10 because max parameters are 10???????????

```
# values for coefficients from eigen vector
pca.out$var
```

```
## $coord
##                    Dim.1        Dim.2       Dim.3       Dim.4        Dim.5
## 100m        -0.014870972 -0.053468594 -0.04631315  0.16622463 -0.036121423
## Long.jump   -0.011037014  0.046233271  0.06127291 -0.20847829 -0.023916452
## Shot.put     0.093404798  0.410300564  0.38056693 -0.09658081  0.575519949
## High.jump   -0.003737963  0.020497437  0.03024197 -0.01078509  0.024973640
## 400m         0.459398847  0.046987108 -0.35742457  0.96824537  0.070937807
## 110m.hurdle  0.014641014 -0.023521895 -0.17995415  0.26365801  0.005228275
## Discus       0.904507505  1.280402979  2.94097520  0.13773090 -0.074224336
## Pole.vault   0.066291824 -0.005348426 -0.06043331 -0.08429291  0.049697097
## Javeline    -0.997918761  4.586347263 -0.83519699 -0.03113659 -0.031397327
## 1500m       11.522524623  0.291537560 -0.29144635 -0.05116776 -0.004740230
##
## $cor
##                   Dim.1       Dim.2       Dim.3        Dim.4        Dim.5
## 100m        -0.05724105 -0.20581024 -0.17826764  0.639828513 -0.139037857
## Long.jump   -0.03531627  0.14793734  0.19606122 -0.667089398 -0.076527927
## Shot.put     0.11470398  0.50386177  0.46734796 -0.118604217  0.706756286
## High.jump   -0.04254499  0.23329907  0.34421005 -0.122754493  0.284246601
## 400m         0.40322992  0.04124218 -0.31372365  0.849861744  0.062264515
## 110m.hurdle  0.03141849 -0.05047617 -0.38616772  0.565789758  0.011219474
## Discus       0.27110306  0.38376814  0.88148232  0.041281327 -0.022246852
## Pole.vault   0.24142230 -0.01947796 -0.22008670 -0.306978863  0.180987439
## Javeline    -0.20931290  0.96198378 -0.17518210 -0.006530883 -0.006585572
## 1500m        0.99935064  0.02528511 -0.02527719 -0.004437789 -0.000411121
##
## $cos2
##                    Dim.1        Dim.2        Dim.3        Dim.4        Dim.5
## 100m        0.0032765375 0.0423578537 0.0317793523 4.093805e-01 1.933153e-02
## Long.jump   0.0012472388 0.0218854580 0.0384400011 4.450083e-01 5.856524e-03
## Shot.put    0.0131570029 0.2538766873 0.2184141136 1.406696e-02 4.995044e-01
## High.jump   0.0018100763 0.0544284562 0.1184805564 1.506867e-02 8.079613e-02
## 400m        0.1625943716 0.0017009170 0.0984225271 7.222650e-01 3.876870e-03
## 110m.hurdle 0.0009871213 0.0025478439 0.1491255059 3.201181e-01 1.258766e-04
## Discus      0.0734968695 0.1472779884 0.7770110866 1.704148e-03 4.949224e-04
## Pole.vault  0.0582847253 0.0003793908 0.0484381560 9.423602e-02 3.275645e-02
## Javeline    0.0438118912 0.9254127864 0.0306887691 4.265243e-05 4.336976e-05
## 1500m       0.9987016992 0.0006393365 0.0006389365 1.969397e-05 1.690205e-07
##
## $contrib
##                    Dim.1        Dim.2        Dim.3       Dim.4        Dim.5
## 100m        1.640459e-04 1.246488e-02  0.022005231  2.47316940  0.375177694
## Long.jump   9.036284e-05 9.319656e-03  0.038517163  3.89031507  0.164475337
## Shot.put    6.471799e-03 7.339980e-01  1.485865204  0.83491982 95.241902111
## High.jump   1.036470e-05 1.831850e-03  0.009382909  0.01041145  0.179337446
## 400m        1.565548e-01 9.626049e-03  1.310648141 83.91390988  1.446980949
## 110m.hurdle 1.590117e-04 2.412322e-03  0.332231671  6.22221562  0.007860027
## Discus      6.068914e-01 7.147991e+00 88.736037789  1.69795392  1.584163281
## Pole.vault  3.259917e-03 1.247219e-04  0.037468826  0.63598259  0.710181485
## Javeline    7.387152e-01 9.171165e+01  7.156408994  0.08677713  0.283460573
## 1500m       9.848768e+01 3.705780e-01  0.871434072  0.23434512  0.006461096
```

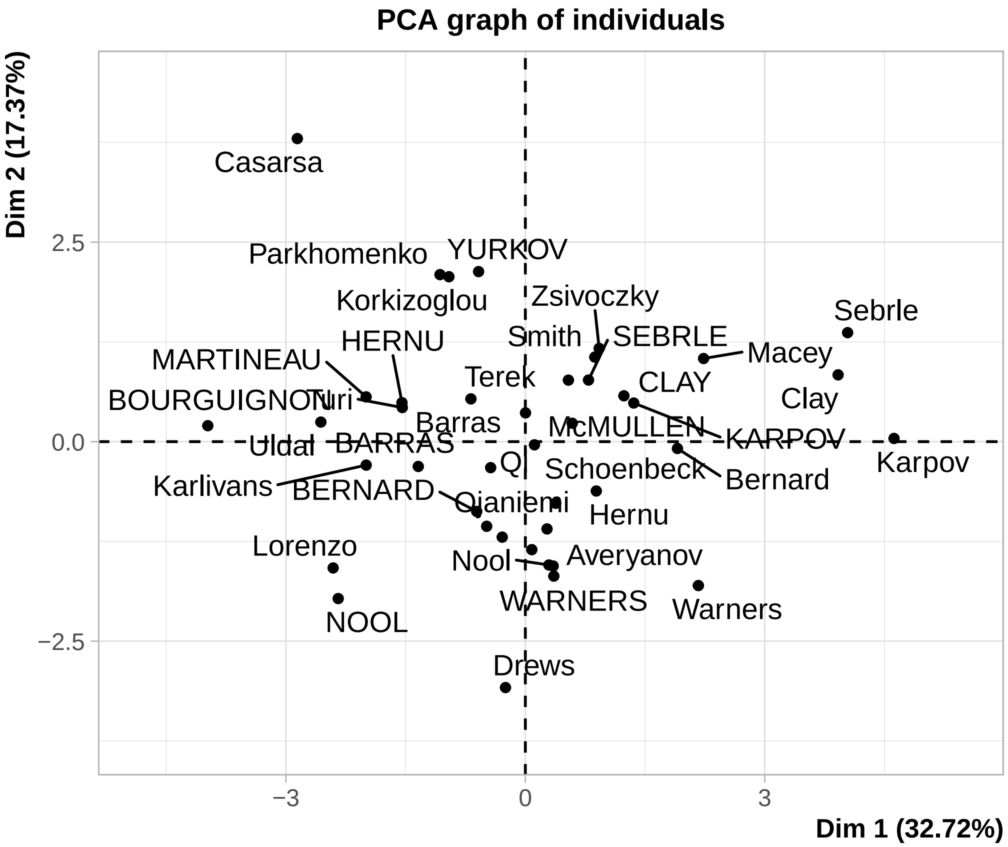
Dim.1 under the $coord is eigen vector for comp1 (Y1). As you can see it is dominated by 1500m with 11.522524623 and very significantly different from others.

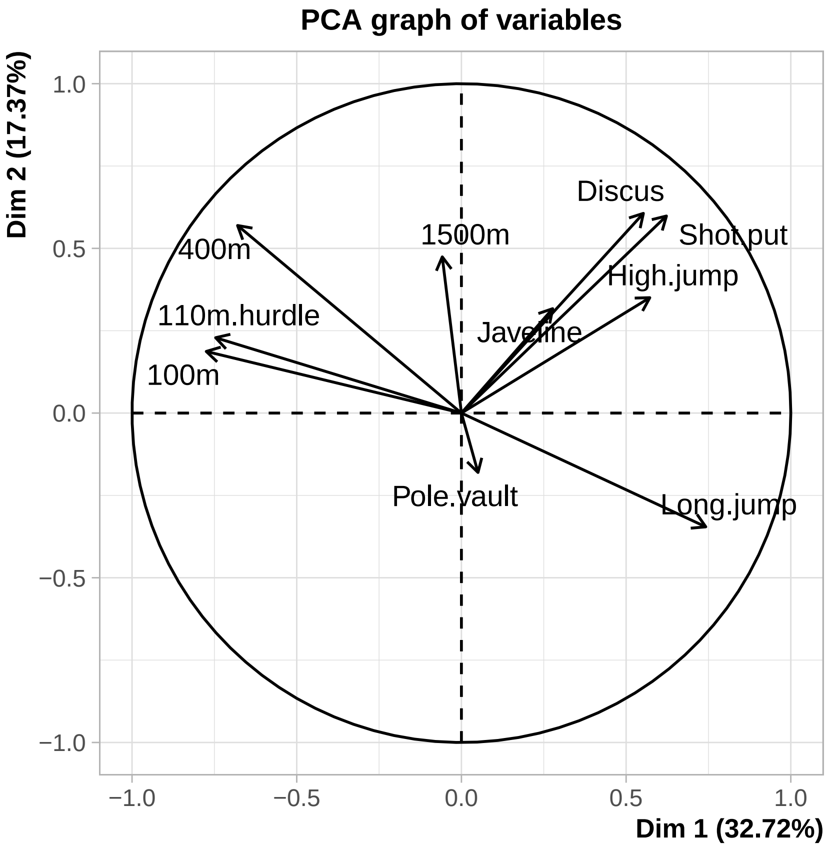
This is an issue we have with the use of covariance matrix when we have different scales (Comp1/PC1 is dominated by 1500m variable due to its high variability). Therefore always recommended to use correlation matrix for PCA

#PCA with corelation matrix

```
pca.out2 <- PCA(decathlon1, ncp = 10)
```

```
## Warning: ggrepel: 5 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```


PCA graph of individuals
Dim 2 (17.37%)
Dim 1 (32.72%)
2.5
0.0
−2.5
−3
0
3
Casarsa
Parkhomenko
YURKOV
Korkizoglou
Zsivoczky
Sebrle
Smith
SEBRLE
HERNU
MARTINEAU
Macey
Terek
CLAY
Clay
BOURGUIGNON
Turi
Barras
McMULLEN
KARPOV
Uldal
BARRAS
Karpov
QI
Schoenbeck
Bernard
Karlivans
BERNARD
Ojaniemi
Hernu
Lorenzo
Nool
Averyanov
WARNERS
Warners
NOOL
Drews



```
pca.out2$eig
```

```
##          eigenvalue percentage of variance cumulative percentage of variance
## comp 1   3.2719055              32.719055                          32.71906
## comp 2   1.7371310              17.371310                          50.09037
## comp 3   1.4049167              14.049167                          64.13953
## comp 4   1.0568504              10.568504                          74.70804
## comp 5   0.6847735               6.847735                          81.55577
## comp 6   0.5992687               5.992687                          87.54846
## comp 7   0.4512353               4.512353                          92.06081
## comp 8   0.3968766               3.968766                          96.02958
## comp 9   0.2148149               2.148149                          98.17773
## comp 10  0.1822275               1.822275                         100.00000
```

In this case sum of eigen values should be equal to Tr(A); A is correlation matric. In this case since we have 10 parameters it should be 10

```
sum(pca.out2$eig[,1])
```

```
## [1] 10
```

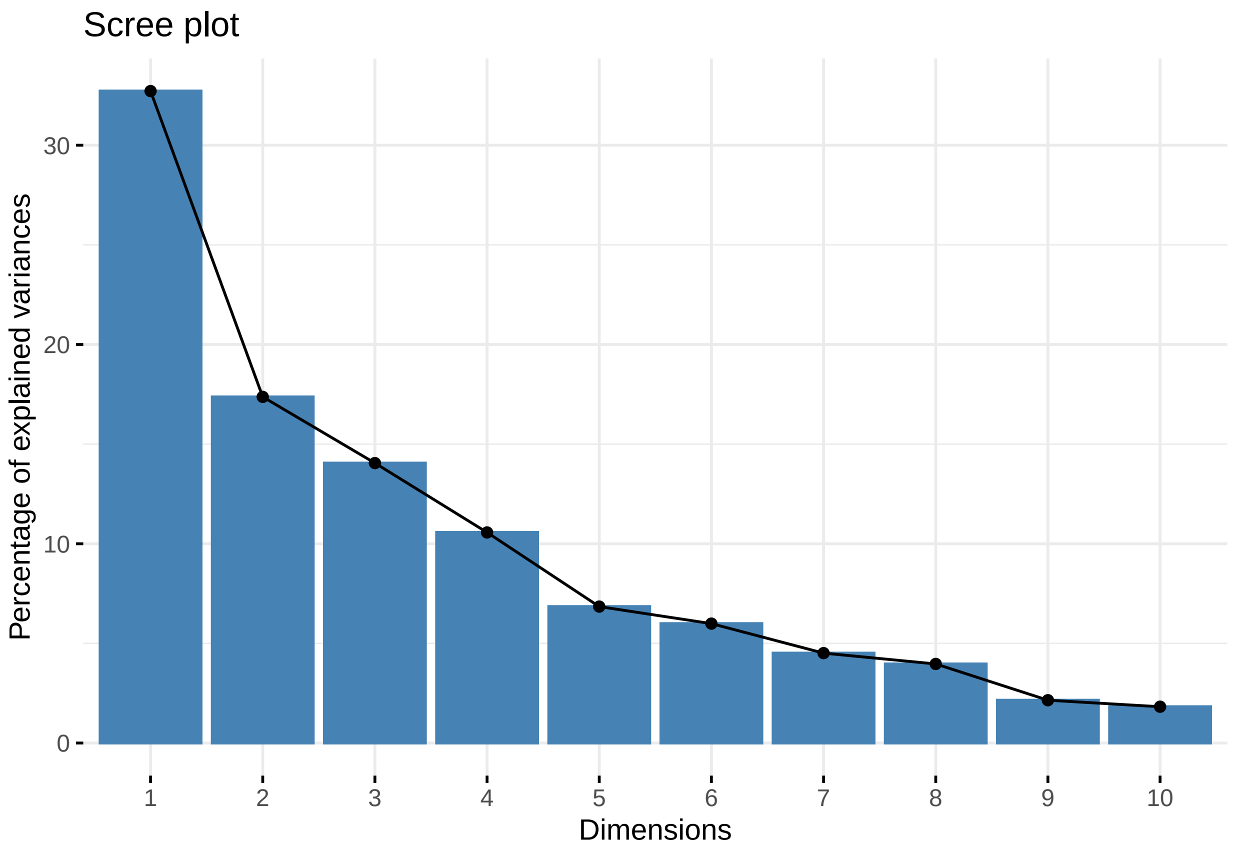
Perfect :)

Important How many PCs should be retain? This can be explain by eigen values or cumulative percentage of variance. Rule of thumb… or generically we prefer the comp with eigen value > 1 OR cumulative percentage of variance at least > 70%

Apart from above there is a another technique to decide number of PCs, which is scree plot

#Scree plot to decide PCs

```
fviz_eig(pca.out2)
```



Along the X axis all the PCs and Y axis is percentage of explained variance. These PCs are Orthogonal to each other

When you are deciding PCs based on scree plot what you have to look at it where the slope of percentage explained is negligible (Elbow Joint). In this case it's from 1st to 5th PCs.

```
pca.out2$var
```

```
## $coord
##                   Dim.1      Dim.2       Dim.3       Dim.4       Dim.5
## 100m        -0.77471983  0.1871420 -0.18440714 -0.03781826  0.30219639
## Long.jump    0.74189974 -0.3454213  0.18221105  0.10178564  0.03667805
## Shot.put     0.62250255  0.5983033 -0.02337844  0.19059161  0.11115082
## High.jump    0.57194530  0.3502936 -0.25951193 -0.13559420  0.55543957
## 400m        -0.67960994  0.5694378  0.13146970  0.02930198 -0.08769157
## 110m.hurdle -0.74624532  0.2287933 -0.09263738  0.29083103  0.16432095
## Discus       0.55246652  0.6063134  0.04295225 -0.25967143 -0.10482712
## Pole.vault   0.05034151 -0.1803569  0.69175665  0.55153397  0.32995932
## Javeline     0.27711085  0.3169891 -0.38965541  0.71227728 -0.30512892
## 1500m       -0.05807706  0.4742238  0.78214280 -0.16108904 -0.15356189
##                  Dim.6       Dim.7        Dim.8       Dim.9      Dim.10
## 100m        -0.22920075  0.25645445  0.290800753  0.04855323  0.18111827
## Long.jump    0.23697868  0.42164691 -0.013236949  0.22370807  0.03459636
## Shot.put    -0.23647411 -0.20805510 -0.197770097  0.19804125  0.16660497
## High.jump    0.36211310 -0.06143068  0.078805424 -0.11293209 -0.04543178
## 400m         0.25741324 -0.08357871  0.134436894  0.25590161 -0.17672678
## 110m.hurdle  0.07713202  0.24003322 -0.447988786 -0.06958442 -0.03878833
## Discus      -0.34787054  0.28877439  0.024184898 -0.07175021 -0.19174040
## Pole.vault  -0.20256095 -0.06580383  0.112160780 -0.03845860 -0.11801187
## Javeline     0.12633919  0.07170506  0.186563995 -0.11463138  0.03746881
## 1500m        0.23089724  0.05617697  0.008641731 -0.14262892  0.18323074
##
## $cor
##                  Dim.1      Dim.2       Dim.3       Dim.4       Dim.5
## 100m        -0.77471983  0.1871420 -0.18440714 -0.03781826  0.30219639
## Long.jump    0.74189974 -0.3454213  0.18221105  0.10178564  0.03667805
## Shot.put     0.62250255  0.5983033 -0.02337844  0.19059161  0.11115082
## High.jump    0.57194530  0.3502936 -0.25951193 -0.13559420  0.55543957
## 400m        -0.67960994  0.5694378  0.13146970  0.02930198 -0.08769157
## 110m.hurdle -0.74624532  0.2287933 -0.09263738  0.29083103  0.16432095
## Discus       0.55246652  0.6063134  0.04295225 -0.25967143 -0.10482712
## Pole.vault   0.05034151 -0.1803569  0.69175665  0.55153397  0.32995932
## Javeline     0.27711085  0.3169891 -0.38965541  0.71227728 -0.30512892
## 1500m       -0.05807706  0.4742238  0.78214280 -0.16108904 -0.15356189
##                  Dim.6       Dim.7        Dim.8       Dim.9      Dim.10
## 100m        -0.22920075  0.25645445  0.290800753  0.04855323  0.18111827
## Long.jump    0.23697868  0.42164691 -0.013236949  0.22370807  0.03459636
## Shot.put    -0.23647411 -0.20805510 -0.197770097  0.19804125  0.16660497
## High.jump    0.36211310 -0.06143068  0.078805424 -0.11293209 -0.04543178
## 400m         0.25741324 -0.08357871  0.134436894  0.25590161 -0.17672678
## 110m.hurdle  0.07713202  0.24003322 -0.447988786 -0.06958442 -0.03878833
## Discus      -0.34787054  0.28877439  0.024184898 -0.07175021 -0.19174040
## Pole.vault  -0.20256095 -0.06580383  0.112160780 -0.03845860 -0.11801187
## Javeline     0.12633919  0.07170506  0.186563995 -0.11463138  0.03746881
## 1500m        0.23089724  0.05617697  0.008641731 -0.14262892  0.18323074
##
## $cos2
##                   Dim.1      Dim.2        Dim.3        Dim.4       Dim.5
## 100m        0.600190812 0.03502213 0.0340059930 0.0014302206 0.091322660
## Long.jump   0.550415232 0.11931587 0.0332008675 0.0103603165 0.001345279
## Shot.put    0.387509426 0.35796686 0.0005465513 0.0363251605 0.012354505
## High.jump   0.327121422 0.12270561 0.0673464410 0.0183857880 0.308513117
## 400m        0.461869674 0.32425938 0.0172842817 0.0008586058 0.007689811
## 110m.hurdle 0.556882084 0.05234639 0.0085816841 0.0845826853 0.027001375
## Discus      0.305219255 0.36761593 0.0018448960 0.0674292539 0.010988725
## Pole.vault  0.002534268 0.03252860 0.4785272696 0.3041897208 0.108873151
## Javeline    0.076790421 0.10048206 0.1518313365 0.5073389244 0.093103658
## 1500m       0.003372945 0.22488818 0.6117473613 0.0259496775 0.023581254
##                   Dim.6       Dim.7        Dim.8       Dim.9      Dim.10
## 100m        0.052532985 0.065768884 8.456508e-02 0.002357417 0.032803826
## Long.jump   0.056158895 0.177786116 1.752168e-04 0.050045300 0.001196908
## Shot.put    0.055920005 0.043286926 3.911301e-02 0.039220335 0.027757216
## High.jump   0.131125895 0.003773728 6.210295e-03 0.012753657 0.002064046
## 400m        0.066261577 0.006985401 1.807328e-02 0.065485634 0.031232355
## 110m.hurdle 0.005949349 0.057615948 2.006940e-01 0.004841992 0.001504535
## Discus      0.121013911 0.083390649 5.849093e-04 0.005148092 0.036764380
## Pole.vault  0.041030940 0.004330144 1.258004e-02 0.001479064 0.013926802
## Javeline    0.015961591 0.005141616 3.480612e-02 0.013140353 0.001403912
## 1500m       0.053313533 0.003155852 7.467951e-05 0.020343009 0.033573506
##
## $contrib
##                   Dim.1     Dim.2       Dim.3       Dim.4     Dim.5      Dim.6
## 100m        18.34376957  2.016090  2.42049891  0.13532858 13.336184  8.7661822
## Long.jump   16.82246707  6.868559  2.36319121  0.98030118  0.196456  9.3712380
## Shot.put    11.84353954 20.606785  0.03890276  3.43711486  1.804174  9.3313745
## High.jump    9.99788710  7.063694  4.79362526  1.73967752 45.053306 21.8809858
## 400m        14.11622887 18.666374  1.23027094  0.08124195  1.122971 11.0570732
## 110m.hurdle 17.02011495  3.013382  0.61083225  8.00327927  3.943110  0.9927683
## Discus       9.32848615 21.162245  0.13131711  6.38020830  1.604724 20.1935985
## Pole.vault   0.07745541  1.872547 34.06090024 28.78266727 15.899147  6.8468354
## Javeline     2.34696326  5.784369 10.80714169 48.00480246 13.596270  2.6635116
## 1500m        0.10308808 12.945954 43.54331962  2.45537861  3.443657  8.8964324
##                  Dim.7       Dim.8      Dim.9     Dim.10
## 100m        14.5752978 21.30765111  1.0974178 18.0015798
## Long.jump   39.3998719  0.04414894 23.2969457  0.6568208
## Shot.put     9.5929838  9.85520753 18.2577389 15.2321787
## High.jump    0.8363105  1.56479244  5.9370460  1.1326757
## 400m         1.5480619  4.55387876 30.4846864 17.1392121
## 110m.hurdle 12.7684941 50.56835299  2.2540304  0.8256354
## Discus      18.4805257  0.14737813  2.3965253 20.1749915
## Pole.vault   0.9596200  3.16976132  0.6885298  7.6425363
## Javeline     1.1394535  8.77001197  6.1170598  0.7704170
## 1500m        0.6993807  0.01881681  9.4700198 18.4239527
```

$coord - PC's coefficients

$cor - PC's correlation with variable

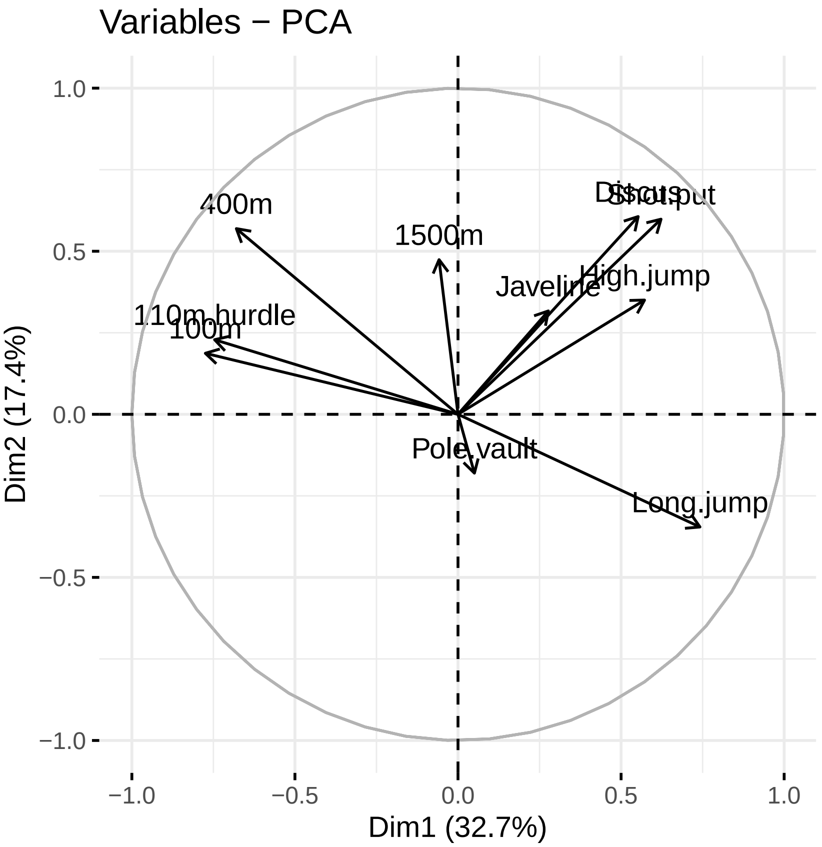
$cos2 -

$contrib -

Dim1 $coord, contributions are fairly distributed.

Visualize PCA

```
fviz_pca_var(pca.out2)
```



By default fviz_pca_var visualize PCA in 2 dimension environment. (Dim1 and Dim2)

1. Whenever 2 arrows are together, they are positively correlated. (e.g.: 100m and 110m hurdle = 0.579888931)
2. When they are negatively correlated, they go opposite direction (e.g.: 100m and Long jump = -0.59867767)
3. when arrows re perpendicular, that means minimum corrlation (e.g.: 400m to javeline = 0.004232096)
4. distance from Origin to arrow head tells us how much it represented. (Longer arrow well represented and shorter arrows are less)
5. Direction of arrow respect to axis. We can say contributon is negative or positive

```
cor(decathlon1)
```

```
##                    100m   Long.jump    Shot.put   High.jump         400m
## 100m         1.00000000 -0.59867767 -0.35648227 -0.24625292  0.520298155
## Long.jump   -0.59867767  1.00000000  0.18330436  0.29464444 -0.602062618
## Shot.put    -0.35648227  0.18330436  1.00000000  0.48921153 -0.138432919
## High.jump   -0.24625292  0.29464444  0.48921153  1.00000000 -0.187956928
## 400m         0.52029815 -0.60206262 -0.13843292 -0.18795693  1.000000000
## 110m.hurdle  0.57988893 -0.50541009 -0.25161571 -0.28328909  0.547987756
## Discus      -0.22170757  0.19431009  0.61576810  0.36921834 -0.117879365
## Pole.vault  -0.08253683  0.20401411  0.06118185 -0.15618074 -0.079292469
## Javeline    -0.15774645  0.11975893  0.37495551  0.17188009  0.004232096
## 1500m       -0.06054645 -0.03368613  0.11580306 -0.04490252  0.408106432
##               110m.hurdle     Discus   Pole.vault     Javeline        1500m
## 100m          0.579888931 -0.2217076 -0.082536834 -0.157746452 -0.06054645
## Long.jump    -0.505410086  0.1943101  0.204014112  0.119758933 -0.03368613
## Shot.put     -0.251615714  0.6157681  0.061181853  0.374955509  0.11580306
## High.jump    -0.283289090  0.3692183 -0.156180742  0.171880092 -0.04490252
## 400m          0.547987756 -0.1178794 -0.079292469  0.004232096  0.40810643
## 110m.hurdle   1.000000000 -0.3262010 -0.002703885  0.008743251  0.03754024
## Discus       -0.326200961  1.0000000 -0.150072400  0.157889799  0.25817510
## Pole.vault   -0.002703885 -0.1500724  1.000000000 -0.030000603  0.24744778
## Javeline      0.008743251  0.1578898 -0.030000603  1.000000000 -0.18039313
## 1500m         0.037540240  0.2581751  0.247447780 -0.180393128  1.00000000
```

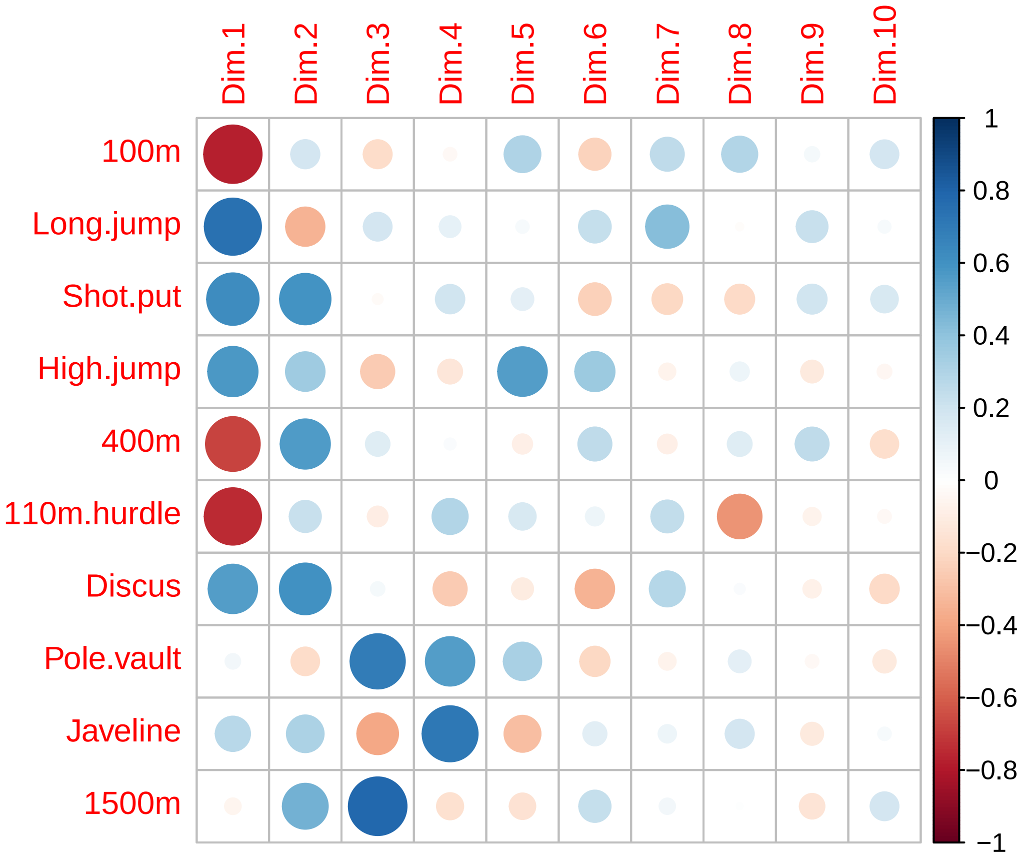
#plot of correlations

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.2.3
```

```
## corrplot 0.92 loaded
```

```
corrplot(pca.out2$var$cor, is.corr = T)
```



1. Color indicate positive or negative correlation 2. size indicate contribution

#contribution

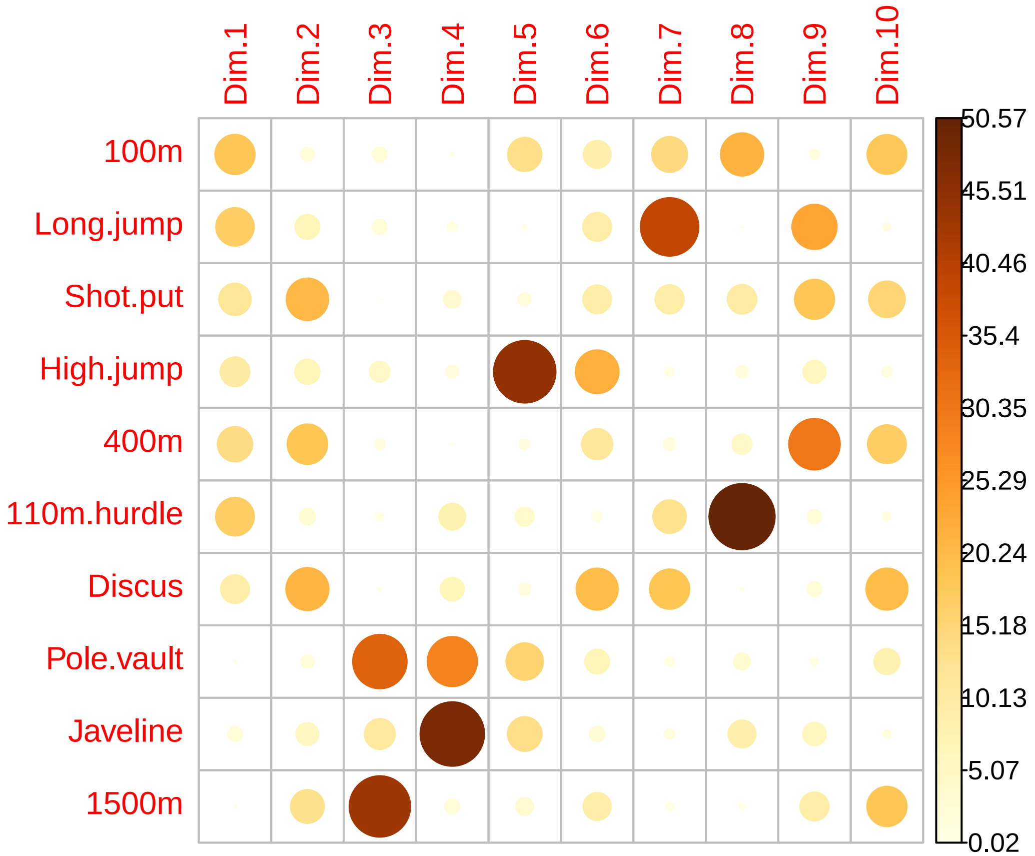
```
pca.out2$var$contrib
```

```
##                   Dim.1     Dim.2       Dim.3       Dim.4     Dim.5      Dim.6
## 100m        18.34376957  2.016090  2.42049891  0.13532858 13.336184  8.7661822
## Long.jump   16.82246707  6.868559  2.36319121  0.98030118  0.196456  9.3712380
## Shot.put    11.84353954 20.606785  0.03890276  3.43711486  1.804174  9.3313745
## High.jump    9.99788710  7.063694  4.79362526  1.73967752 45.053306 21.8809858
## 400m        14.11622887 18.666374  1.23027094  0.08124195  1.122971 11.0570732
## 110m.hurdle 17.02011495  3.013382  0.61083225  8.00327927  3.943110  0.9927683
## Discus       9.32848615 21.162245  0.13131711  6.38020830  1.604724 20.1935985
## Pole.vault   0.07745541  1.872547 34.06090024 28.78266727 15.899147  6.8468354
## Javeline     2.34696326  5.784369 10.80714169 48.00480246 13.596270  2.6635116
## 1500m        0.10308808 12.945954 43.54331962  2.45537861  3.443657  8.8964324
##                  Dim.7       Dim.8      Dim.9     Dim.10
## 100m        14.5752978 21.30765111  1.0974178 18.0015798
## Long.jump   39.3998719  0.04414894 23.2969457  0.6568208
## Shot.put     9.5929838  9.85520753 18.2577389 15.2321787
## High.jump    0.8363105  1.56479244  5.9370460  1.1326757
## 400m         1.5480619  4.55387876 30.4846864 17.1392121
## 110m.hurdle 12.7684941 50.56835299  2.2540304  0.8256354
## Discus      18.4805257  0.14737813  2.3965253 20.1749915
## Pole.vault   0.9596200  3.16976132  0.6885298  7.6425363
## Javeline     1.1394535  8.77001197  6.1170598  0.7704170
## 1500m        0.6993807  0.01881681  9.4700198 18.4239527
```

Comp1, Dim1 PC represent 32.7% variability of data and out of that 18.34% is represented by 100m

#ctribution plot

```
corrplot(pca.out2$var$contrib, is.corr = F)
```



Statistical significant of raw variables on PCs

#Evaluating the significant of the associated raw variable on each dimention

```
#?dimdesc
dimdesc.out <- dimdesc(pca.out2, proba = 0.05)
dimdesc.out$Dim.1
```

```
##
## Link between the variable and the continuous variables (R-square)
## =================================================================================
##              correlation      p.value
## Long.jump      0.7418997 2.849886e-08
## Shot.put       0.6225026 1.388321e-05
## High.jump      0.5719453 9.362285e-05
## Discus         0.5524665 1.802220e-04
## 400m          -0.6796099 1.028175e-06
## 110m.hurdle   -0.7462453 2.136962e-08
## 100m          -0.7747198 2.778467e-09
```

```
dimdesc.out <- dimdesc(pca.out2, proba = 0.1)
dimdesc.out$Dim.1
```

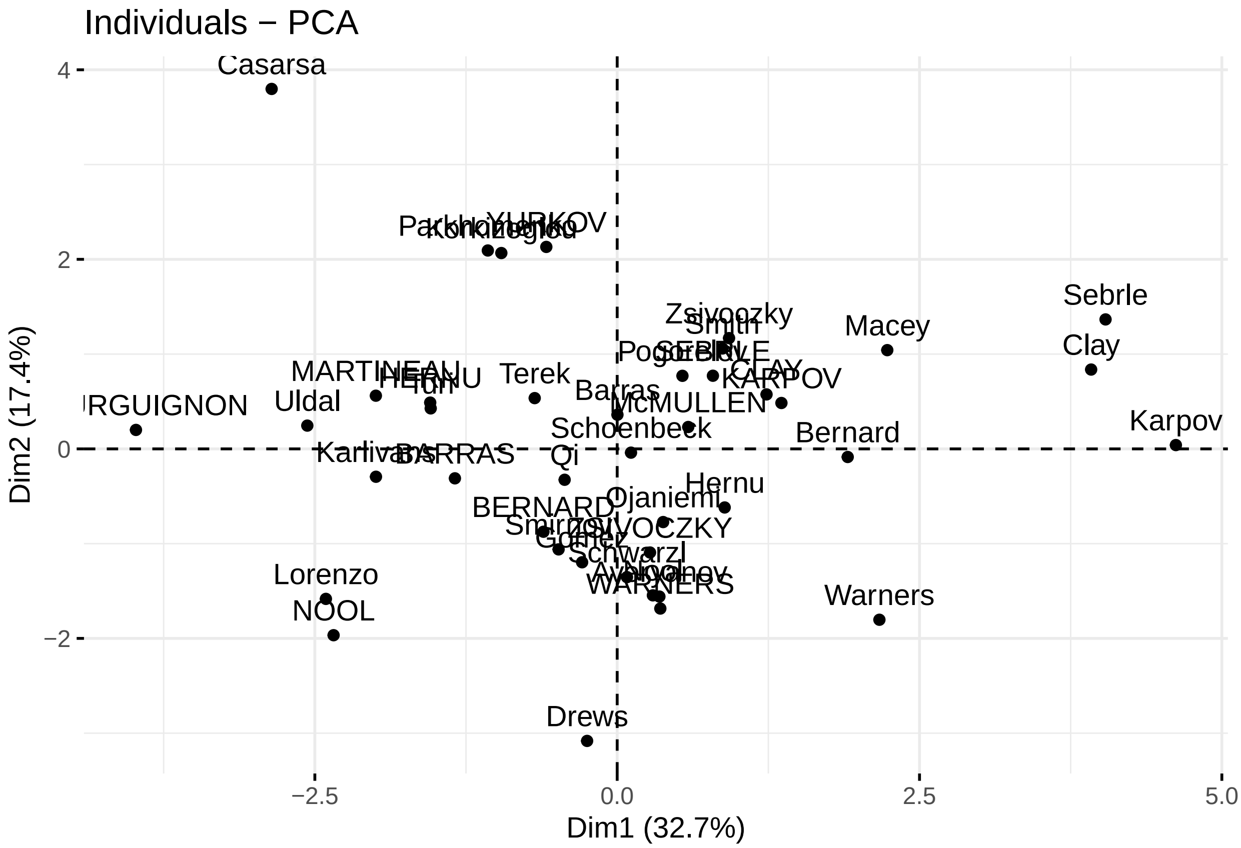
```
##
## Link between the variable and the continuous variables (R-square)
## =================================================================================
##              correlation      p.value
## Long.jump     0.7418997 2.849886e-08
## Shot.put      0.6225026 1.388321e-05
## High.jump     0.5719453 9.362285e-05
## Discus        0.5524665 1.802220e-04
## Javeline      0.2771108 7.942460e-02
## 400m         -0.6796099 1.028175e-06
## 110m.hurdle  -0.7462453 2.136962e-08
## 100m         -0.7747198 2.778467e-09
```

```
dimdesc.out$Dim.3
```

```
##
## Link between the variable and the continuous variables (R-square)
## =================================================================================
##             correlation      p.value
## 1500m         0.7821428 1.554450e-09
## Pole.vault    0.6917567 5.480172e-07
## Javeline     -0.3896554 1.179331e-02
```

#Individual variables

```
fviz_pca_ind(pca.out2)
```



```
pca.out2$ind$coord
```

```
##                    Dim.1       Dim.2         Dim.3       Dim.4       Dim.5
## SEBRLE       0.791627717  0.77161120  0.8268411940  1.17462736  0.70715903
## CLAY         1.234990563  0.57457807  2.1412469664 -0.35484483 -1.97457138
## KARPOV       1.358214936  0.48402090  1.9562579869 -1.85652411  0.79521472
## BERNARD     -0.609515083 -0.87462853  0.8899406619  2.22061245  0.36163619
## YURKOV      -0.585968338  2.13095422 -1.2251567968  0.87357915  1.25136918
## WARNERS      0.356889530 -1.68495667  0.7665531449 -0.58930466  1.00166155
## ZSIVOCZKY    0.271774781 -1.09377558 -1.2827673831 -1.62156457  0.04407327
## McMULLEN     0.587516189  0.23072991 -0.4176329823 -1.52423275  0.25151965
## MARTINEAU   -1.995359298  0.56099598 -0.7299466011 -0.54219127  1.57821348
## HERNU       -1.546076462  0.48838301  0.8407858519  0.33119470 -0.23498029
## BARRAS      -1.341652727 -0.31091157 -0.0003683375 -0.64525336  0.31628748
## NOOL        -2.344973806 -1.96637500 -1.3364815492  0.19530310  0.83022767
## BOURGUIGNON -3.979041865  0.19986019  1.3264851034  0.52435051  0.29001247
## Sebrle       4.038448501  1.36582606 -0.2899565043  1.94113411  0.37695454
## Clay         3.919365157  0.83696136  0.2311753205  1.49397212 -1.03760852
## Karpov       4.619987275  0.03999523 -0.0415857980 -1.31352566  0.18772953
## Macey        2.233460566  1.04176620 -1.8643620154 -0.74321353  0.97727010
## Warners      2.168396445 -1.80320025  0.8510173287 -0.28459958 -0.15139464
## Zsivoczky    0.925132183  1.16865180 -1.4774802908  0.80759472  0.87297257
## Hernu        0.889037852 -0.61842522 -0.8982953480 -0.13478508  0.63498488
## Nool         0.295305667 -1.54561667  1.3552601286  2.19972249 -0.03538887
## Bernard      1.906334368 -0.08580429 -0.7571859709 -1.45097918  0.34164564
## Schwarzl     0.081078659 -1.35345710  0.8224866222  0.39909143  0.28836991
## Pogorelov    0.539677028  0.77075099  1.3476197769 -0.55215692  0.97229498
## Schoenbeck   0.114430985 -0.03985061  0.7404039810  0.92884762 -0.85442565
## Barras       0.002145203  0.36033768 -1.5696934888  0.61238304 -0.67506540
## Smith        0.870310570  1.05932552 -1.6434290616 -1.12132654 -2.01990789
## Averyanov    0.349155138 -1.55864999  0.2825354037 -0.02726334 -0.36098809
## Ojaniemi     0.380113999 -0.77244734 -0.3709431419  0.68727919 -0.49925676
## Smirnov     -0.484514213 -1.06066118 -1.2283378499  0.56603194 -0.40718384
## Qi          -0.434466691 -0.32614690 -1.0697978123 -0.20497404 -0.53559673
## Drews       -0.248684024 -3.08167683  1.0548427375 -0.64577513 -0.17841420
## Parkhomenko -1.069429104  2.09318218 -0.9999839029  1.53455272  0.28180687
## Terek       -0.681953059  0.53561440  2.2091259997  0.10862305  1.04315650
## Gomez       -0.289889208 -1.19671611 -1.3061025895  0.07785116 -1.26143770
## Turi        -1.541813056  0.42716773  0.5140859441 -0.14284738 -0.03871969
## Lorenzo     -2.408509980 -1.58292969 -1.5023461069  0.30103076 -0.68195932
## Karlivans   -1.994368727 -0.29418240 -0.3427836937 -1.27206999  0.37322060
## Korkizoglou -0.957829813  2.06638554  2.5865525263 -1.19146757 -0.80089104
## Uldal       -2.562259591  0.24546871 -0.4191406445 -0.02118842 -1.25954121
## Casarsa     -2.857088268  3.79784505  0.0305611909 -0.73769370 -0.77044963
##                   Dim.6       Dim.7       Dim.8        Dim.9       Dim.10
## SEBRLE       1.03062025  0.55152286  0.43565550 -0.137558873  0.500773776
## CLAY        -0.69012566  0.70797408  0.60341904 -0.649244121 -0.266119255
## KARPOV      -0.73275122  0.18993920  0.25029693 -0.800653566  0.523268830
## BERNARD     -0.27559819 -0.04961070 -0.06745808 -0.723281017  0.188459291
## YURKOV       0.10460569  0.57392548 -0.09460361 -0.202216418  0.056442514
## WARNERS     -0.03235612  0.09659035  0.30044536  0.607465094  0.721284785
## ZSIVOCZKY   -0.18537032  0.54300336  0.73915738 -0.354412930 -0.146059166
## McMULLEN     1.76788298 -0.10495329  0.25748521 -0.538115021 -0.329648746
## MARTINEAU   -2.36187721  0.33238326  0.44812186  0.399108572 -0.584484592
## HERNU       -0.22249804  1.56637865  0.06731710  1.322902163  0.224961868
## BARRAS      -0.40938985 -0.31646694  0.65609217 -0.280275720  0.787396307
## NOOL         0.99433721  0.87390607 -0.07083076 -0.507299066  0.224544692
## BOURGUIGNON  0.04908478  0.18826373 -0.52289350 -0.418431784  0.019430261
## Sebrle      -0.06778623  0.55497694  0.75259621  0.062225128  0.633130750
## Clay         0.81264979  0.86751544  0.30284530 -0.013214514 -0.818729281
## Karpov      -0.74161145  0.45414303 -1.07084207 -0.180314690  0.124574958
## Macey        0.04001698  0.19270849 -0.68974257  0.438424818 -0.166834121
## Warners      0.07938750 -0.06100253 -0.21454500  0.167391548  0.082692117
## Zsivoczky    0.25856515 -0.31352231 -0.54933741 -0.450901316 -0.307612539
## Hernu       -0.64075332 -0.56048799  0.31778062 -0.100140875 -0.301487915
## Nool        -0.40536270  0.19880131 -0.30034913 -0.133023057 -0.365393514
## Bernard      1.08624555 -0.46768687 -0.33134950  0.213024073 -0.052371717
## Schwarzl    -0.08523813 -0.08615387  0.71657047  0.624647042 -0.456745967
## Pogorelov    0.36345544 -0.82134446  0.47857370  0.788644760 -0.262352584
## Schoenbeck  -0.48675267 -0.37063763  0.42065982  0.657717817 -0.299437858
## Barras      -0.90389490 -0.44240244  0.64577089 -0.033610560  0.300976456
## Smith       -1.29179510 -0.92942803  0.10317674 -0.191201887  0.089439874
## Averyanov    0.61650428 -1.35164082 -0.76357457  0.751397746 -0.487116486
## Ojaniemi     0.79491103 -0.22760918 -1.62981377  0.505551532  0.517965031
## Smirnov     -0.15711173 -0.36720227 -0.26809288 -0.455029343 -0.540450528
## Qi           0.42721799  0.94756780  0.17805603 -0.274429259 -0.396922190
## Drews       -0.18998231 -0.46206001  0.54928962 -0.075890729 -0.192719322
## Parkhomenko  0.02343641 -1.72466895  0.21356688 -0.115141938  0.395896413
## Terek       -1.23637214 -0.68973066 -1.32531512 -0.389166469 -0.358281849
## Gomez       -0.49836092 -0.14287075 -0.09667920  0.399815735  1.252146708
## Turi         1.58272797 -1.29143851  1.53696070 -0.147061945 -0.115731380
## Lorenzo     -0.06822736  0.37175217 -0.96612514 -0.312209844 -0.168379968
## Karlivans    0.56539057  0.97187597  0.11879004  0.276702842 -0.138001002
## Korkizoglou  0.80314710 -0.16523591 -0.96695318 -0.240688329  0.444241723
## Uldal        0.19808228  0.49760283  0.04826482  0.003274377  0.001399909
## Casarsa      0.08494663  0.26532308 -0.21238691  0.505220025 -0.334146282
```

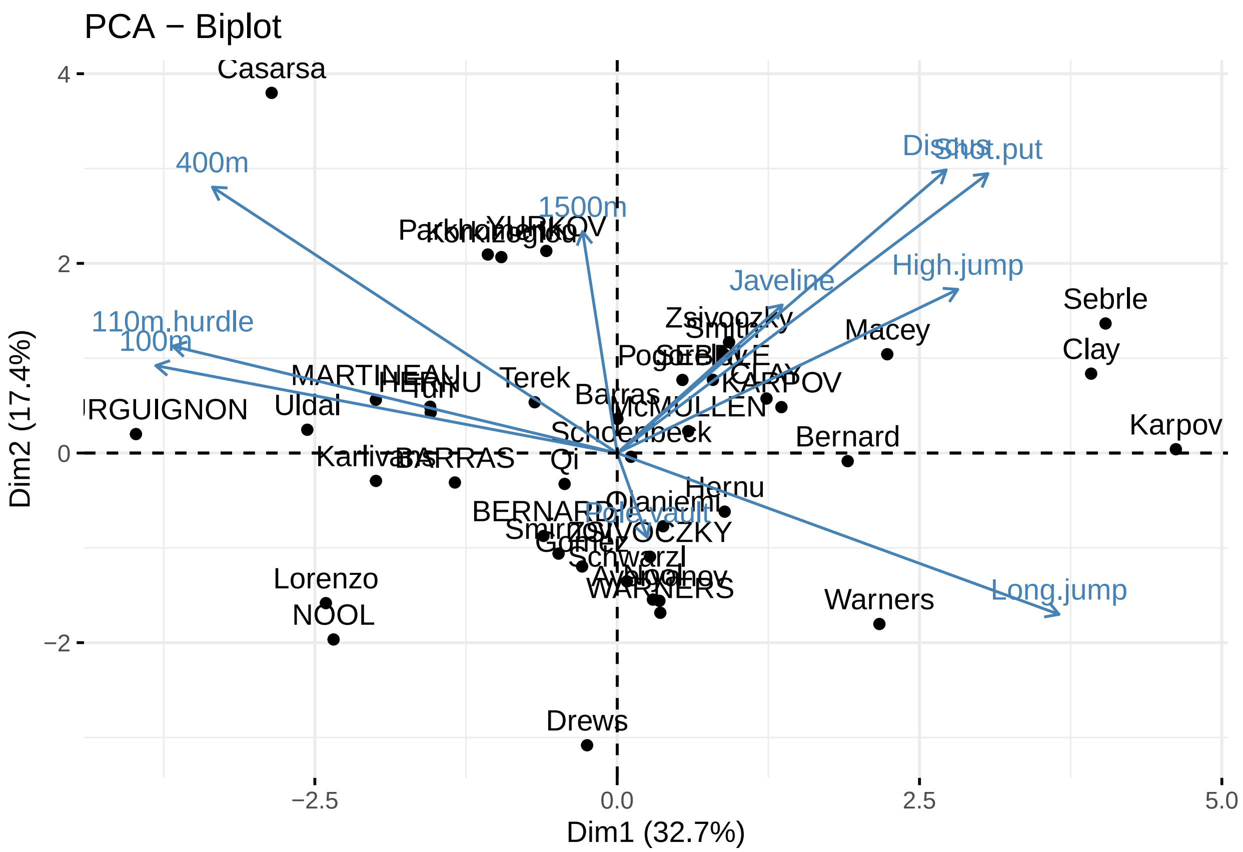
```
#pca.out2$ind$coord[pca.out2$ind$coord[,0] == "NOOL" ,]
```

Karpov 4.619987275 0.03999523

Karpov’s performance more explain by Dim1 which means that variables contributed to Dim1 define his performance

#Biplot

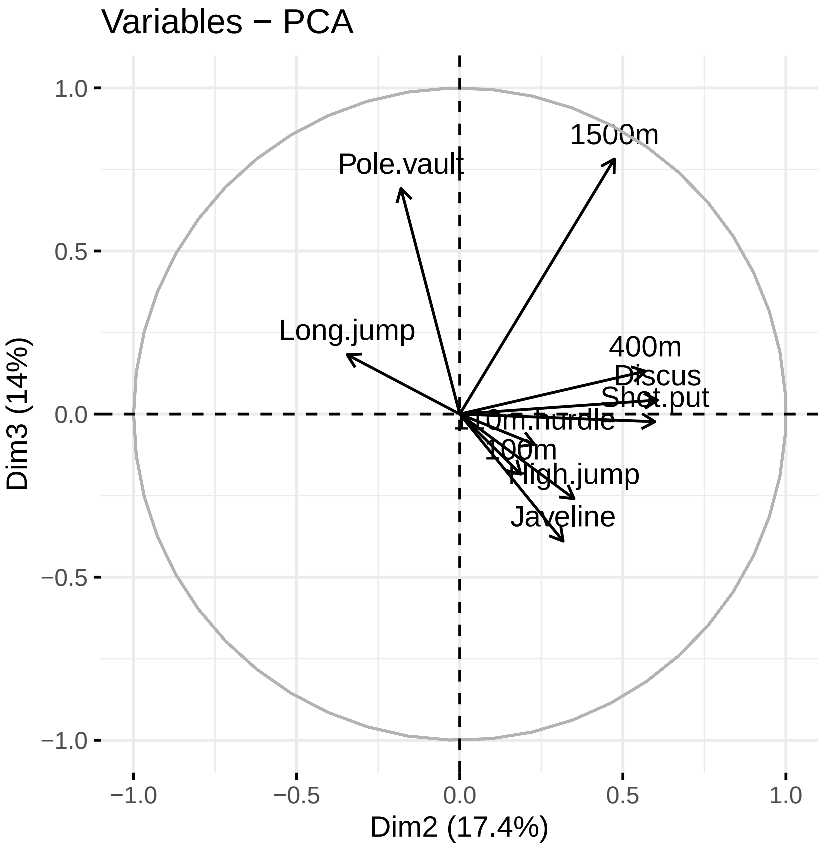
```
fviz_pca_biplot(pca.out2)
```



If athletic close to an arrow he is explain more using that specific arrow

#if you want to visualise other dimensions

```
fviz_pca_var(pca.out2, axes = c(2,3))
```


Variables − PCA
1500m
Pole.vault
Long.jump
400m
Discus
Shot.put
110m.hurdle
100m
High.jump
Javeline
1.0
0.5
0.0
−0.5
−1.0
Dim3 (14%)
−1.0
−0.5
0.0
0.5
1.0
Dim2 (17.4%)