Self-organizing map

Monika Osiak, Anna Pręgowska, Patrycja Szczepaniak, Rafał Szulejko

08 05 2020

Wstęp

Zadaniem było wykonanie analizy wybranego zbioru danych z wykorzystaniem samoorganizującej się mapy (SOM). Ze względu na brak informacji na temat tego, co miał zawierać projekt, zdecydowaliśmy się zrealizować tutorial dot. tego tematu, aby pisząc kod jednocześnie wzgłębić się w teorię zagadnienia. Link: https://clarkdatalabs.github.io/soms/SOM_NBA

Samoorganizujące się mapy (Self Organizing Maps, SOM) lub sieci Kohonena to sieci neuronów, z którymi są stowarzyszone współrzędne na prostej, płaszczyźnie lub w dowolnej n-wymiarowej przestrzeni. Uczenie tego rodzaju sieci polega na zmianach współrzędnych neuronów, tak, by dążyły one do wzorca zgodnego ze strukturą analizowanych danych. Sieci zatem „rozpinają się" wokół zbiorów danych, dopasowując do nich swoją strukturę.

Sieci te klasyfikują wielowymiarowe dane wejściowe w taki sposób, by możliwa była ich reprezentacji w mniejszej ilości wymiarów - przeważnie dwóch - przy jednoczesnym jak najwierniejszym odwzorowaniu struktury wewnętrznej wektora wejściowego.

Nasze dane dotyczą statystyk poszczególnych zawodników NBA w sezonie 2015/16. Uwzględniamy statystyki graczy podczas rozegranych 36 minut. Wczytane dane są już oczyszczone i przygotowane do dalszej pracy.

```
library(RCurl)
NBA <- read.csv(text = getURL("https://raw.githubusercontent.com/clarkdatalabs/soms/master/NBA_2016_pla
    sep = ",", header = T, check.names = FALSE)
```

```
head(NBA)
```

```
##                 Player            Pos Age  Tm  G GS   MP  FG  FGA   FG%  3P 3PA
## 1  1        Quincy Acy  Power Forward  25 SAC 59 29  876 4.9  8.8 0.556 0.8 2.0
## 2  2      Jordan Adams Shooting Guard  21 MEM  2  0   15 4.8 14.4 0.333 0.0 2.4
## 3  4     Arron Afflalo Shooting Guard  30 NYK 71 57 2371 5.4 12.1 0.443 1.4 3.6
## 4  5     Alexis Ajinca         Center  27 NOP 59 17  861 6.3 13.2 0.476 0.0 0.0
## 5  7 LaMarcus Aldridge  Power Forward  30 SAS 74 74 2261 8.5 16.6 0.513 0.0 0.3
## 6 10        Tony Allen Shooting Guard  34 MEM 64 57 1620 4.8 10.4 0.458 0.3 0.9
##     3P%  2P  2PA   2P%  FT  FTA   FT% ORB DRB  TRB AST STL BLK TOV  PF  PTS
## 1 0.388 4.1  6.8 0.606 2.1  2.8 0.735 2.7 5.1  7.7 1.1 1.2 1.0 1.1 4.2 12.6
## 2 0.000 4.8 12.0 0.400 7.2 12.0 0.600 0.0 4.8  4.8 7.2 7.2 0.0 4.8 4.8 16.8
## 3 0.382 4.0  8.5 0.469 1.7  2.0 0.840 0.3 3.7  4.0 2.2 0.4 0.2 1.2 2.2 13.8
## 4 0.000 6.3 13.1 0.478 2.2  2.6 0.839 3.1 8.1 11.2 1.3 0.8 1.5 2.3 5.6 14.7
## 5 0.000 8.5 16.4 0.521 4.1  4.8 0.858 2.8 7.3 10.1 1.8 0.6 1.3 1.6 2.4 21.2
## 6 0.357 4.4  9.5 0.468 2.0  3.1 0.652 2.3 4.3  6.6 1.6 2.4 0.4 1.7 3.9 11.9
```

```
colnames(NBA)
```

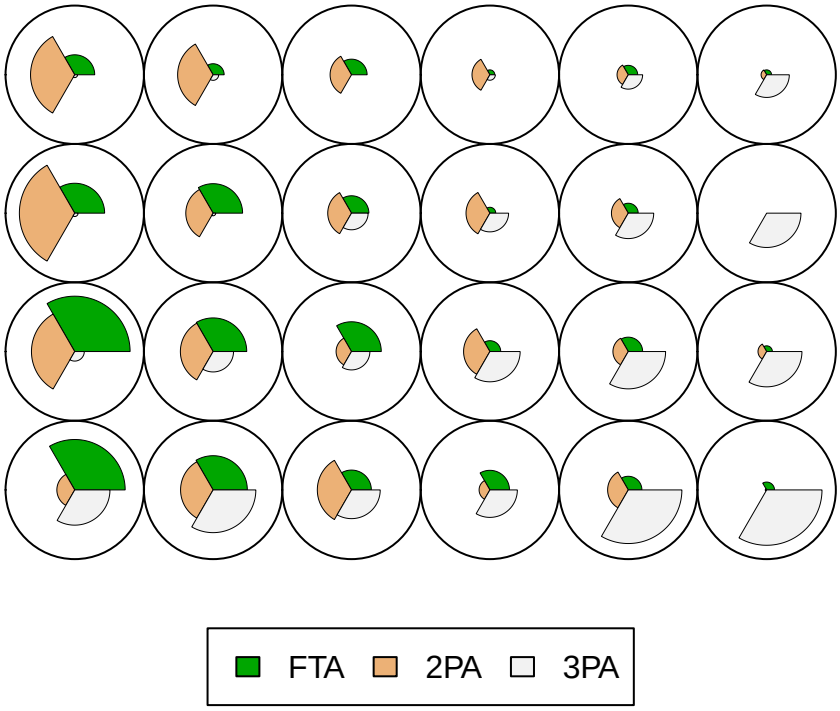
```
##  [1] ""       "Player" "Pos"    "Age"    "Tm"     "G"      "GS"     "MP"
##  [9] "FG"     "FGA"    "FG%"    "3P"     "3PA"    "3P%"    "2P"     "2PA"
## [17] "2P%"    "FT"     "FTA"    "FT%"    "ORB"    "DRB"    "TRB"    "AST"
```

```
## [25] "STL"    "BLK"    "TOV"    "PF"     "PTS"
```

Podstawowa sieć - próby rzutów

Weźmy pod uwagę wszystkie próby rzutów wykonane przez zawodników (rzuty wolne, za 2 oraz 3 punkty).

```
NBA.measures1 <- c("FTA", "2PA", "3PA")
NBA.SOM1 <- som(scale(NBA[NBA.measures1]), grid = somgrid(6, 4, "rectangular"))
plot(NBA.SOM1)
```



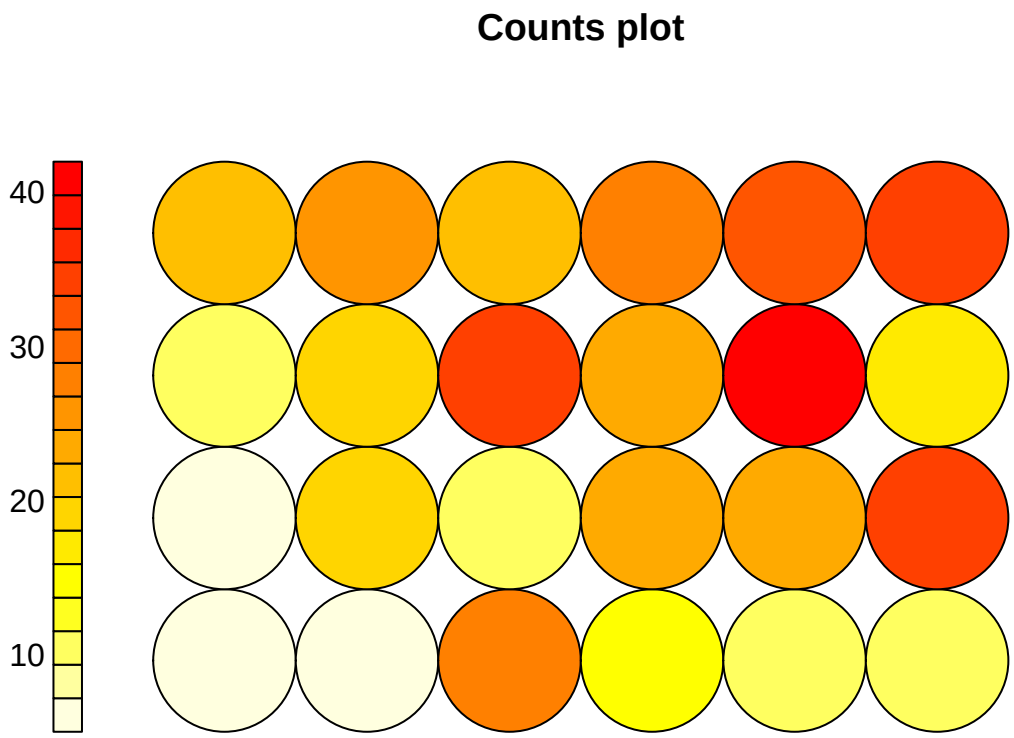
Nasze dane zostały przeskalowane i wyśrodkowane. Ustaliliśmy również rozmiar i układ siatki. Standardowy wykres SOM Kohonena tworzy wykresy kołowe dla reprezentowanych wektorów. Promień każdego klina odpowiada wielkości w danym wymiarze. Można tu dostrzec pierwsze wzorce - gracze są generalnie pogrupowani wg. tego, ile rzutów danego rodzaju wykonują.

```
plot(NBA.SOM1, type="changes")
```

Zliczanie zawodników

Powyższy wykres to jedynie mapa danych, w której każda komórka wyświetla swój reprezentatywny wektor. Teraz, za pomoca mapy ciepła, chcemy przypisać każdego gracza do komórki. Poniższa mapa termiczna została stworzona na podstawie liczby graczy zaliczonych do każdej komórki.

```
plot(NBA.SOM1, type = "counts", palette.name = colors, heatkey = TRUE)
```

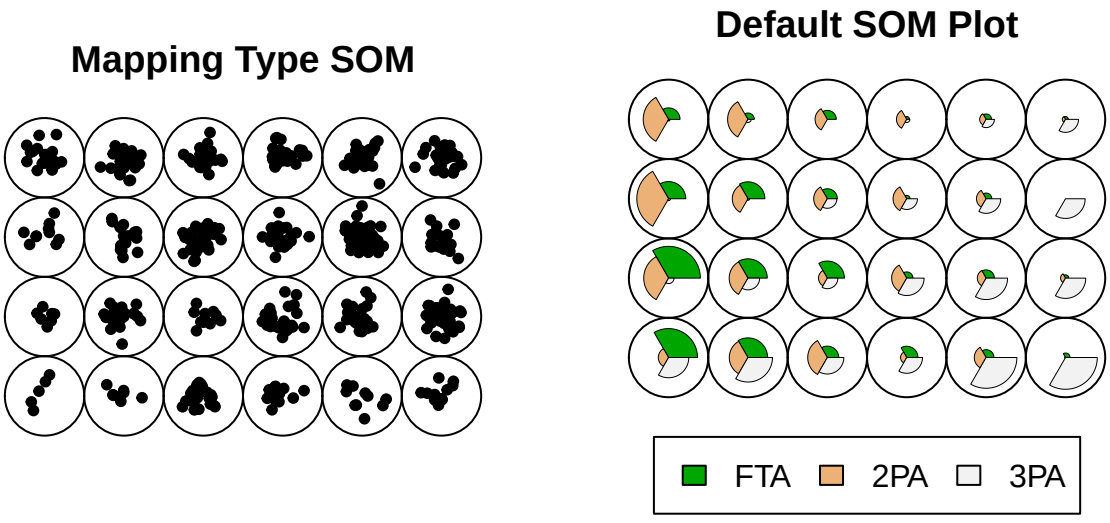


Mapowanie punktów według statystyk

Zamiast typowego wykresu SOM, jak ten zaprezentowany na początku, możemy użyć mappingu.

W tym przypadku każdy z graczy jest tak uwzględniony na mapie, jak blisko reprezentatywnych wektorów odpowiadających komórkom po prawej stronie są ich statystyki.

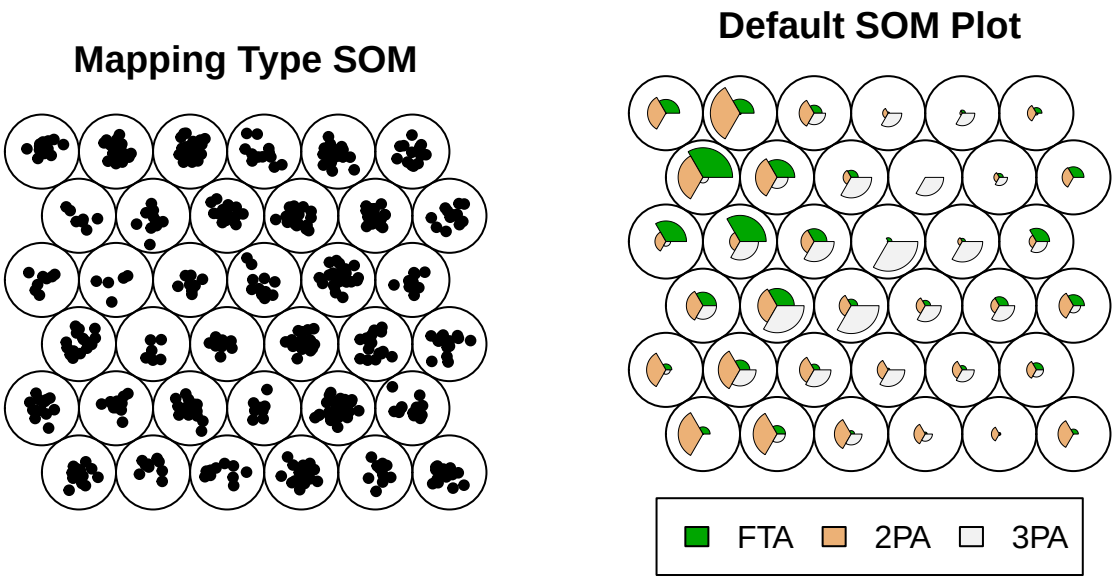
```
par(mfrow = c(1, 2))
plot(NBA.SOM1, type = "mapping", pchs = 20, main = "Mapping Type SOM")
plot(NBA.SOM1, main = "Default SOM Plot")
```



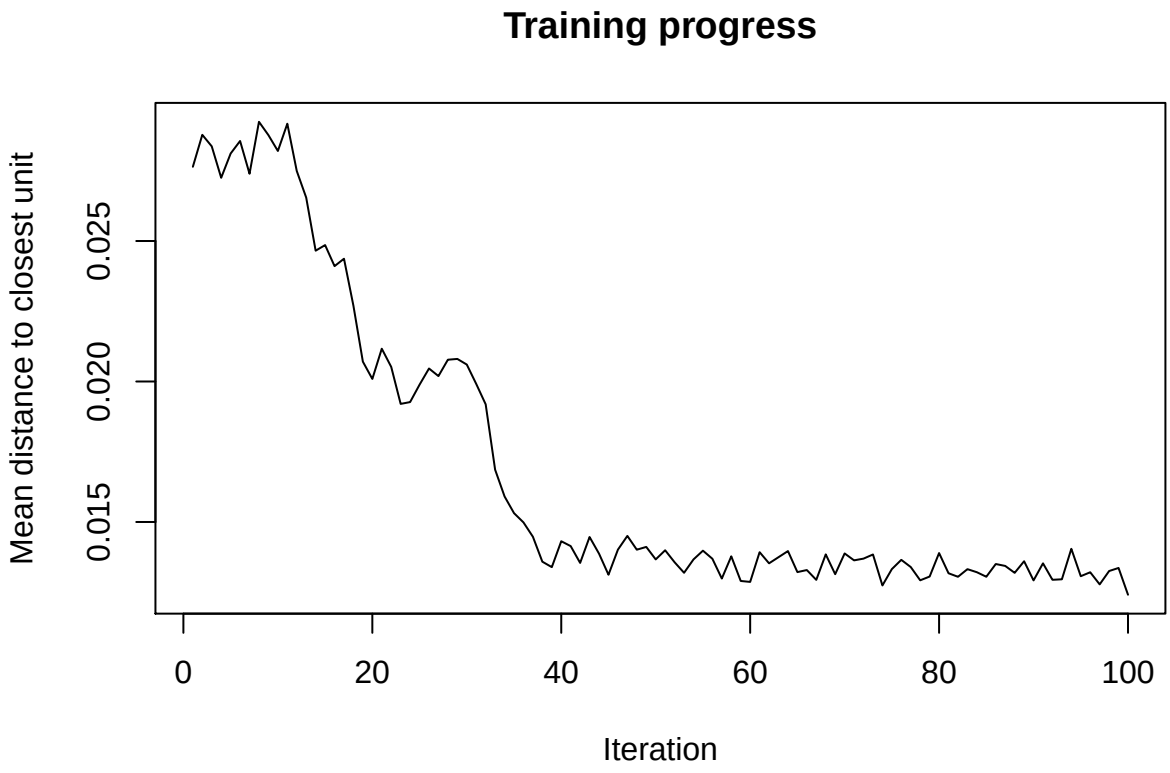
Sieci toroidalne

Możemy zmienić rozmieszczenie komórek na mapie i zamiast prostokątnej siatki użyć torologicznej topologii. W przypadku siatki prostokątnej komórki na krawędziach (zwłaszcza w rogach) mają mniej sąsiadów niż komórki wewnętrzne, a przez to bardziej skrajne wartości są wypychane na krawędzie.

```
NBA.SOM2 <- som(scale(NBA[NBA.measures1]), grid = somgrid(6, 6, "hexagonal"), toroidal = TRUE)
par(mfrow = c(1, 2))
plot(NBA.SOM2, type = "mapping", pchs = 20, main = "Mapping Type SOM")
plot(NBA.SOM2, main = "Default SOM Plot")
```



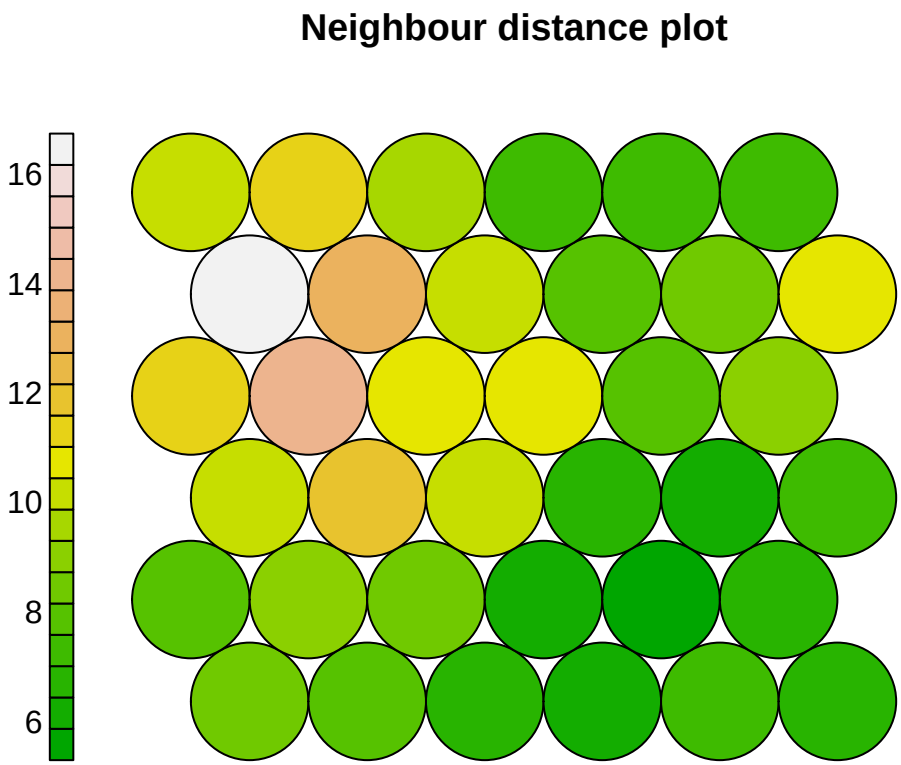

```
plot(NBA.SOM2, type="changes")
```



Mapowanie wzajemnych odległości komórek

W tym przypadku komórki są tak pokolorowane, że możemy stwierdzić ich odległość do najbliższych sąsiadów - dzięki czemu wizualizujemy, jak daleko od siebie znajdują się cechy w przestrzeni o wyższej liczbie wymiarów. Można to trochę porównać do kolorowania mapy geograficznej.

```
plot(NBA.SOM2, type = "dist.neighbours", palette.name = terrain.colors)
```



Sieci nadzorowane

Za pomocą nadzorowanych sieci Kohonena możemy robić klasyfikację. W dalszej części będziemy pracować z danymi, które mają więcej niż trzy wymiary, aby lepiej ukazać działanie SOM.

```
NBA.measures2 <- c("FTA", "FT", "2PA", "2P", "3PA", "3P", "AST", "ORB", "DRB",
    "TRB", "STL", "BLK", "TOV")
```

Klasyfikacja ze względu na pozycję na boisku funkcją xyf()

Za pomocą tej funkcji będziemy chcieli dokonać klasyfikacji zawodników ze względu na ich pozycję na boisku. Losowo dzielimy dane na testowe i treningowe.

```
NBA.SOM3 <- xyf(NBA.training, classvec2classmat(NBA$Pos[training_indices]), grid = somgrid(13, 13, "hex
```

Parametr `xweight` pozwala ustalić wagę, jaką zmienne opisujące mają względem zmiennej opisywanej w algorytmie trenowania.

```
pos.prediction <- predict(NBA.SOM3, newdata = NBA.testing)
table(NBA[-training_indices, "Pos"], pos.prediction$prediction)
```

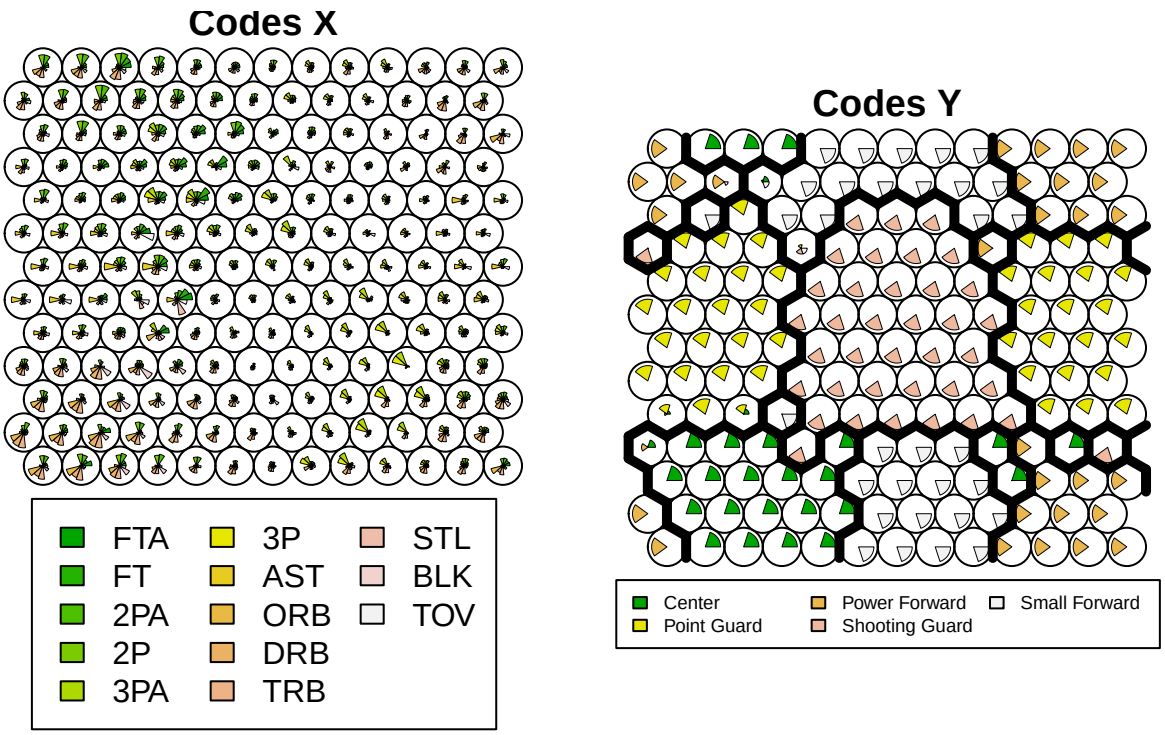
```
##
```

```
##                  Center Point Guard Power Forward Shooting Guard Small Forward
##    Center            16           0            24              0             1
##    Point Guard        0          61             0              8             7
##    Power Forward      9           2            37              0             9
##    Shooting Guard     0          10             4             30            32
##    Small Forward      3           5            12              7            26
```

Wizualizacja przewidywań

Teraz użyjemy do klasyfikacji wszystkich rekordów, a nie tylko zbioru treningowego. Zwiększamy istotność zmiennych opisujących.

```
NBA.SOM4 <- xyf(scale(NBA[, NBA.measures2]), classvec2classmat(NBA[, "Pos"]), grid = somgrid(13, 13, "h

par(mfrow = c(1, 2))

plot(NBA.SOM4, type = "codes", main = c("Codes X", "Codes Y"))
NBA.SOM4.hc <- cutree(hclust(dist(NBA.SOM4$codes$Y)), 5)
add.cluster.boundaries(NBA.SOM4, NBA.SOM4.hc)
```

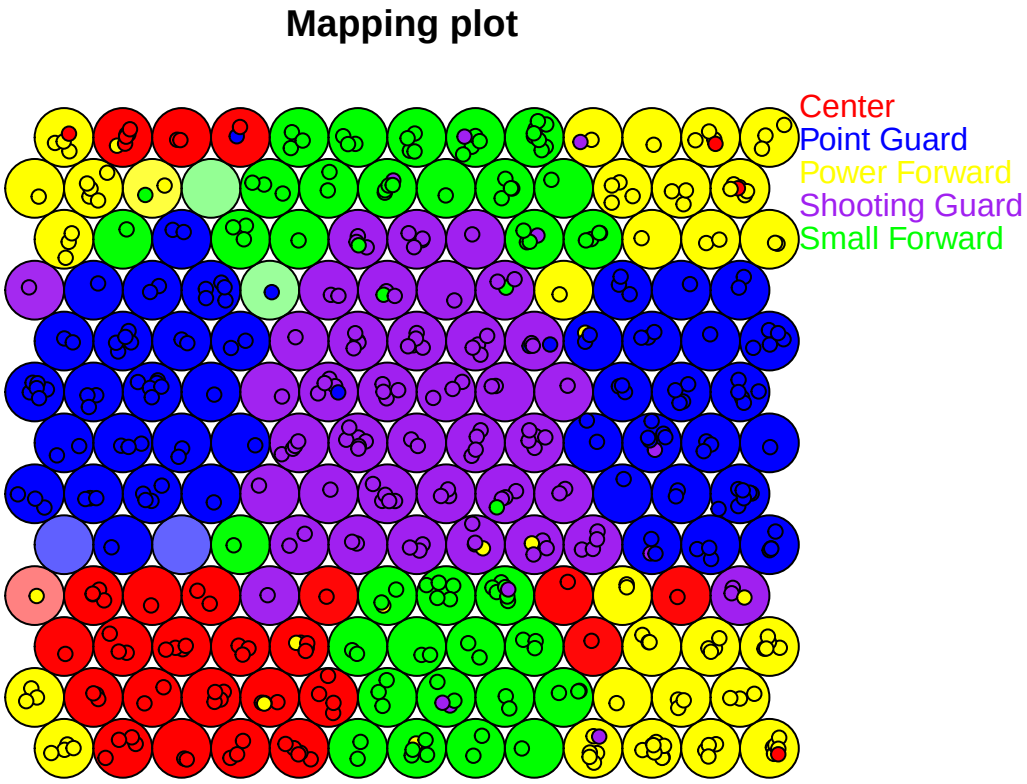


Po lewej stronie mamy wizualizację statystyk graczy, a po prawej - predykcję ich pozycji na boisku. Nadal jednak nie mamy pojęcia o tym, jak wypadło grupowanie.

```
bg.pallet <- c("red", "blue", "yellow", "purple", "green")

position.predictions <- classmat2classvec(predict(NBA.SOM4)$unit.predictions)
base.color.vector <- bg.pallet[match(position.predictions, levels(NBA$Pos))]

bgcols <- c()
max.conf <- apply(NBA.SOM4$codes$Y, 1, max)
for (i in 1:length(base.color.vector)) {
```

```
    bgcols[i] <- adjustcolor(base.color.vector[i], max.conf[i])
}
```

```
par(mar = c(0, 0, 0, 4), xpd = TRUE)
plot(NBA.SOM4, type = "mapping", pchs = 21, col = "black", bg = bg.pallet[match(NBA$Pos,
    levels(NBA$Pos))], bgcol = bgcols)

legend("topright", legend = levels(NBA$Pos), text.col = bg.pallet, bty = "n",
    inset = c(-0.03, 0))
```



Każdy z pięciu kolorów reprezentuje jedną pozycję. Przezroczystość komórek mówi o tym, z jaką pewnością sieć sklasyfikowała tę komórkę. Kolory pojedynczych graczy (te małe kółka) reprezentują ich rzeczywistą pozycję.

Podsumowanie i wnioski

- Sieci Kohonena są w stanie dobrze dopasować się do skomplikowanych i wielowymiarowych danych, natomiast ich analiza może być bardziej wymagająca niż przy innych metodach służących do redukcji wymiarowości. Segmentacja danych wejściowych jest jednak wystarczajaco intuicyjna.
- Algorytm nadaje się tylko do zbioróW danych, które rzeczywiście da się przedstawić w postaci dwuwymiarowej, i jednocześnie zawierających wyłącznie czyste dane liczbowe.
- Zastosowanie sieci toroidalnych nieznacznie zmniejszyło średnie odległości neuronów (zmiana wewnątrz tego samego rzędu wielkości) przy niemal identycznej ilości epok. Być może różnica byłaby bardziej widoczna przy większym zbiorze danych.