esiea
INGÉNIEUR.E.S NUMÉRIQUES UTILES

TD Nº3 : Analyse des Correspondances Multiples
Analyse exploratoire

Auteurs : Manelle Nouar, Eliott Bernardou, Sarra Madad, Mohamed Abdelkader, Alfio Chadoin
Enseignants : Myriam Bertrand & Jordan Gonzalez

30 septembre 2022

1 | Pokemon

1.1 Questions/Réponses

1. **Téléchargez la bibliothèque ade4 et la bibliothèque adegraphics.**

```
library(ade4)
library(adegraphics)
library(PCAmixdata)
library(vcd)
library(corrplot)
library(factoextra)
library(FactoMineR)
```

2. **Chargez les données du fichier pokemon.csv disponible à l'adresse https : tinyurl.com/y4y6a86m.**

```
pokemon = read.csv2("/cloud/project/pokemon.csv", sep = ",")
```

3. **Associez ce jeu de données à un jeu de données de type data.frame que vous appelerez poke.**

```
poke = as.data.frame(pokemon)

View(poke)
str(poke)
names(poke)
attach(poke)
summary(poke)
dim(poke)
colnames(poke)
row.names(poke)
```

X.	Name	Type.1	Type.2	Total	HP	Attack	Defense	Sp..Atk	Sp..Def	Speed	Generation	Legendary
1	Bulbasaur	Grass	Poison	318	45	49	49	65	65	45	1	False
2	Ivysaur	Grass	Poison	405	60	62	63	80	80	60	1	False
3	Venusaur	Grass	Poison	525	80	82	83	100	100	80	1	False
3	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120	80	1	False
4	Charmander	Fire		309	39	52	43	60	50	65	1	False
5	Charmeleon	Fire		405	58	64	58	80	65	80	1	False
6	Charizard	Fire	Flying	534	78	84	78	109	85	100	1	False
6	CharizardMega Charizard X	Fire	Dragon	634	78	130	111	130	85	100	1	False

```
'data.frame':   800 obs. of  13 variables:
 $ X.         : int  1 2 3 3 4 5 6 6 6 7 ...
 $ Name       : chr  "Bulbasaur" "Ivysaur" "Venusaur" "VenusaurMega Venusaur" ...
 $ Type.1     : chr  "Grass" "Grass" "Grass" "Grass" ...
 $ Type.2     : chr  "Poison" "Poison" "Poison" "Poison" ...
 $ Total      : int  318 405 525 625 309 405 534 634 634 314 ...
 $ HP         : int  45 60 80 80 39 58 78 78 78 44 ...
 $ Attack     : int  49 62 82 100 52 64 84 130 104 48 ...
 $ Defense    : int  49 63 83 123 43 58 78 111 78 65 ...
 $ Sp..Atk    : int  65 80 100 122 60 80 109 130 159 50 ...
 $ Sp..Def    : int  65 80 100 120 50 65 85 85 115 64 ...
 $ Speed      : int  45 60 80 80 65 80 100 100 100 43 ...
 $ Generation: int  1 1 1 1 1 1 1 1 1 1 ...
 $ Legendary : chr  "False" "False" "False" "False" ...
 [1] "X."         "Name"       "Type.1"     "Type.2"     "Total"      "HP"         "Attack"
 [8] "Defense"    "Sp..Atk"    "Sp..Def"    "Speed"      "Generation" "Legendary"
```

En regardant le type de certaines variables, nous allons devoir les transformer en facteur pour pouvoir les utiliser. Le type facteur, facilite la manipulation de données qualitatives (qu'elles soient numériques ou caractères). En effet, en plus de stocker les différents éléments comme un vecteur classique, il stocke également l'ensemble des différentes modalités possibles dans un attribut.

4. **Transformez la variable Generation en type factor.**

```
poke$Generation = as.factor(poke$Generation)
summary(poke)
```

```
       X.           Name              Type.1             Type.2              Total       
 Min.   :  1.0   Length:800         Length:800         Length:800         Min.   :180.0  
 1st Qu.:184.8   Class :character   Class :character   Class :character   1st Qu.:330.0  
 Median :364.5   Mode  :character   Mode  :character   Mode  :character   Median :450.0  
 Mean   :362.8                                                           Mean   :435.1  
 3rd Qu.:539.2                                                           3rd Qu.:515.0  
 Max.   :721.0                                                           Max.   :780.0  
       HP            Attack          Defense          Sp..Atk          Sp..Def      
 Min.   :  1.00   Min.   :  5   Min.   :  5.00   Min.   : 10.00   Min.   : 20.0  
 1st Qu.: 50.00   1st Qu.: 55   1st Qu.: 50.00   1st Qu.: 49.75   1st Qu.: 50.0  
 Median : 65.00   Median : 75   Median : 70.00   Median : 65.00   Median : 70.0  
 Mean   : 69.26   Mean   : 79   Mean   : 73.84   Mean   : 72.82   Mean   : 71.9  
 3rd Qu.: 80.00   3rd Qu.:100   3rd Qu.: 90.00   3rd Qu.: 95.00   3rd Qu.: 90.0  
 Max.   :255.00   Max.   :190   Max.   :230.00   Max.   :194.00   Max.   :230.0  
     Speed         Generation  Legendary      
 Min.   :  5.00   1:166      Length:800        
 1st Qu.: 45.00   2:106      Class :character  
 Median : 65.00   3:160      Mode  :character  
 Mean   : 68.28   4:121                        
 3rd Qu.: 90.00   5:165                        
 Max.   :180.00   6: 82                        
```

Nous pouvons maintenant voir que la variable Generation est bien affiché.

5. **Créez un sous jeu de données composé exclusivement des variables suivantes : Type$_1$, Generation et Legendary.**

```
poke.x = poke[,c(3,12,13)]
```

	Type.1	Generation	Legendary
1	Grass	1	False
2	Grass	1	False
3	Grass	1	False
4	Grass	1	False
5	Fire	1	False
6	Fire	1	False
7	Fire	1	False
8	Fire	1	False
9	Fire	1	False
10	Water	1	False

Nous créons notre jeu de données et on sélectionne bien le rang des variables Type1 (rang 3), Generation (rang 12) et Legendary (rang 13).

6. **Appliquez la fonction summary() au jeu de données poke, de type data.frame.**

```
summary(poke)
summary(poke.x)
```

```
> summary(poke)
       X.            Name              Type.1             Type.2              Total            HP              Attack          Defense
 Min.   :  1.0   Length:800         Length:800         Length:800         Min.   :180.0   Min.   :  1.00   Min.   :  5   Min.   :  5.00
 1st Qu.:184.8   Class :character   Class :character   Class :character   1st Qu.:330.0   1st Qu.: 50.00   1st Qu.: 55   1st Qu.: 50.00
 Median :364.5   Mode  :character   Mode  :character   Mode  :character   Median :450.0   Median : 65.00   Median : 75   Median : 70.00
 Mean   :362.8                                                           Mean   :435.1   Mean   : 69.26   Mean   : 79   Mean   : 73.84
 3rd Qu.:539.2                                                           3rd Qu.:515.0   3rd Qu.: 80.00   3rd Qu.:100   3rd Qu.: 90.00
 Max.   :721.0                                                           Max.   :780.0   Max.   :255.00   Max.   :190   Max.   :230.00
    Sp..Atk          Sp..Def          Speed        Generation  Legendary
 Min.   : 10.00   Min.   : 20.0   Min.   :  5.00   1:166      Length:800
 1st Qu.: 49.75   1st Qu.: 50.0   1st Qu.: 45.00   2:106      Class :character
 Median : 65.00   Median : 70.0   Median : 65.00   3:160      Mode  :character
 Mean   : 72.82   Mean   : 71.9   Mean   : 68.28   4:121
 3rd Qu.: 95.00   3rd Qu.: 90.0   3rd Qu.: 90.00   5:165
 Max.   :194.00   Max.   :230.0   Max.   :180.00   6: 82
> summary(poke.x)
    Type.1          Generation  Legendary
 Length:800         1:166      Length:800
 Class :character   2:106      Class :character
 Mode  :character   3:160      Mode  :character
                    4:121
                    5:165
                    6: 82
```

Nos variables sont maintenant bien formatés et prêtes à être utilisés.

7. **À l'aide de la bibliothèque ade4 et de la bibliothèque adegraphics, appliquez la fonction dudi.acm() au jeu de données poke.x.**

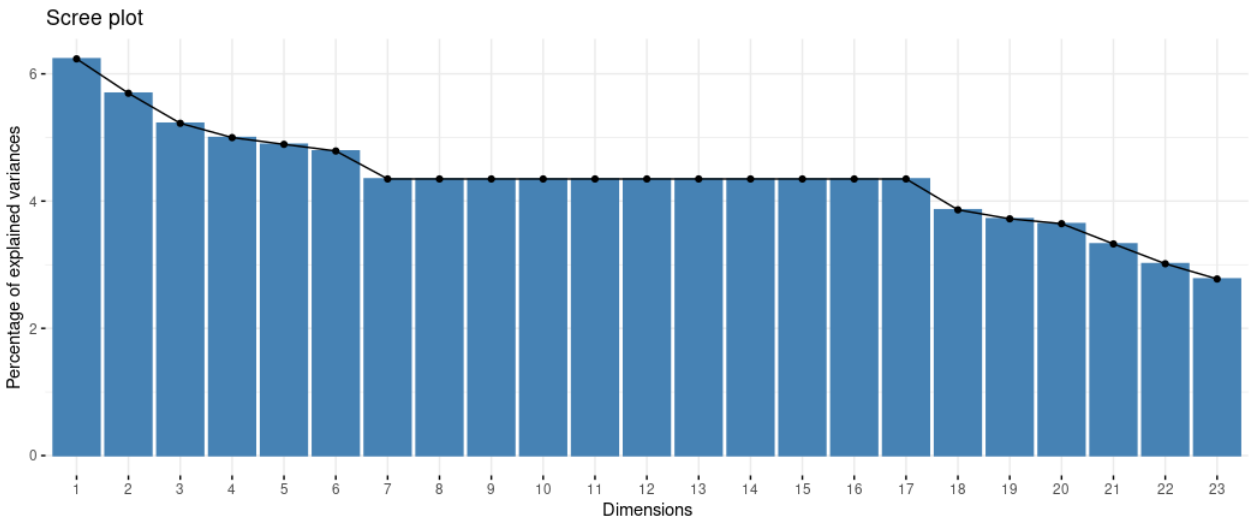
```
poke.x$Type.1 = as.factor(poke$Type.1)
poke.x$Legendary = as.factor(poke$Legendary)
summary(poke.x)
res.acm.poke = dudi.acm(poke.x,scannf=FALSE)
res.acm.poke
```

```
> res.acm.poke
Duality diagramm
class: acm dudi
$call: dudi.acm(df = poke.x, scannf = FALSE)

$nf: 2 axis-components saved
$rank: 23
eigen values: 0.478 0.4366 0.4004 0.3832 0.375 ...
  vector length mode    content
1 $cw    26     numeric column weights
2 $lw    800    numeric row weights
3 $eig   23     numeric eigen values

  data.frame nrow ncol content
1 $tab       800  26   modified array
2 $li        800  2    row coordinates
3 $l1        800  2    row normed scores
4 $co        26   2    column coordinates
5 $c1        26   2    column normed scores
other elements: cr
```

Pour utiliser la fonction dudi.acm(), nous avons dû transformer toutes les variables en facteur. Cette fonction effectue l'analyse des correspondances multiples d'une table de facteurs.

8. **Sauriez-vous dire le nombre total de valeurs propres ? Affichez les valeurs propres avec la fonction fvizscreeplot() du package ade4.**

```
fviz_screeplot(res.acm.poke, ncp=23)
get_eig(res.acm.poke)
```



Sur cette figure, nous pouvons voir qu'il y a une dimension de 23, donc 23 valeurs propres. Par défaut, la figure affiche 10 dimension. Nous avons dû ajouter un paramètre (ncp) pour pouvoir afficher toutes les dimensions.

Si on met ncp = 50 on a encore que 23 valeurs propres qui s'affichent, je pense, c'est plus logique que ncp=23 vu que techniquement le 23 on ne le connait pas encore à cette étape.

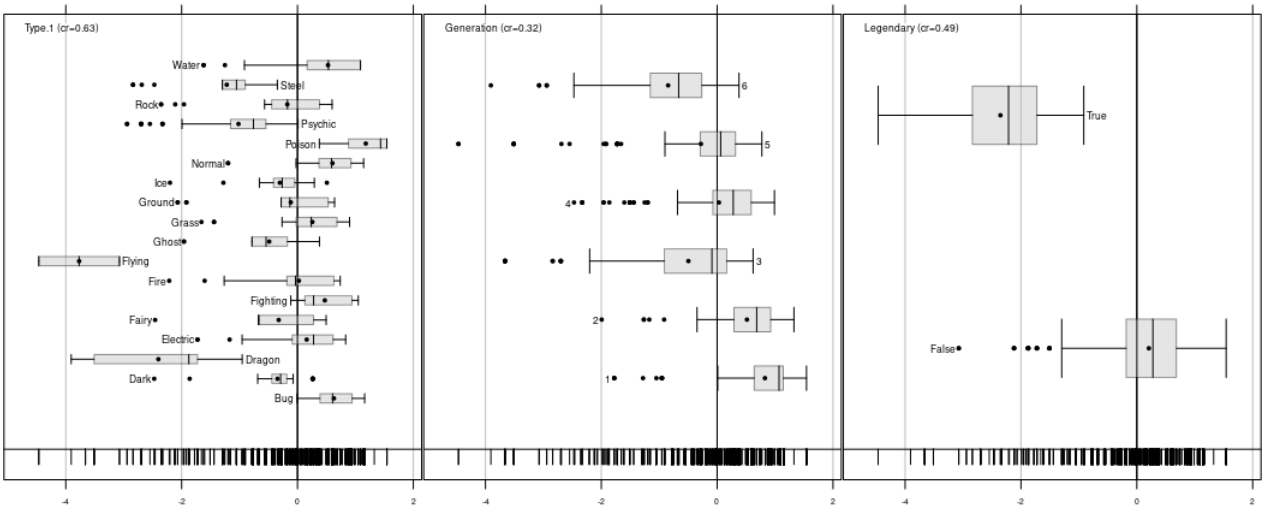
	eigenvalue	variance.percent	cumulative.variance.percent
Dim.1	0.4780316	6.235195	6.235195
Dim.2	0.4365931	5.694692	11.929888
Dim.3	0.4003682	5.222194	17.152081
Dim.4	0.3831707	4.997879	22.149961
Dim.5	0.3750273	4.891660	27.041621
Dim.6	0.3670084	4.787066	31.828686
Dim.7	0.3333333	4.347826	36.176512
Dim.8	0.3333333	4.347826	40.524338
Dim.9	0.3333333	4.347826	44.872165
Dim.10	0.3333333	4.347826	49.219991
Dim.11	0.3333333	4.347826	53.567817
Dim.12	0.3333333	4.347826	57.915643
Dim.13	0.3333333	4.347826	62.263469
Dim.14	0.3333333	4.347826	66.611295
Dim.15	0.3333333	4.347826	70.959121
Dim.16	0.3333333	4.347826	75.306947
Dim.17	0.3333333	4.347826	79.654773
Dim.18	0.2961106	3.862313	83.517086
Dim.19	0.2853067	3.721391	87.238477
Dim.20	0.2794082	3.644455	90.882932
Dim.21	0.2550264	3.326432	94.209364
Dim.22	0.2311516	3.015021	97.224384
Dim.23	0.2127972	2.775616	100.000000

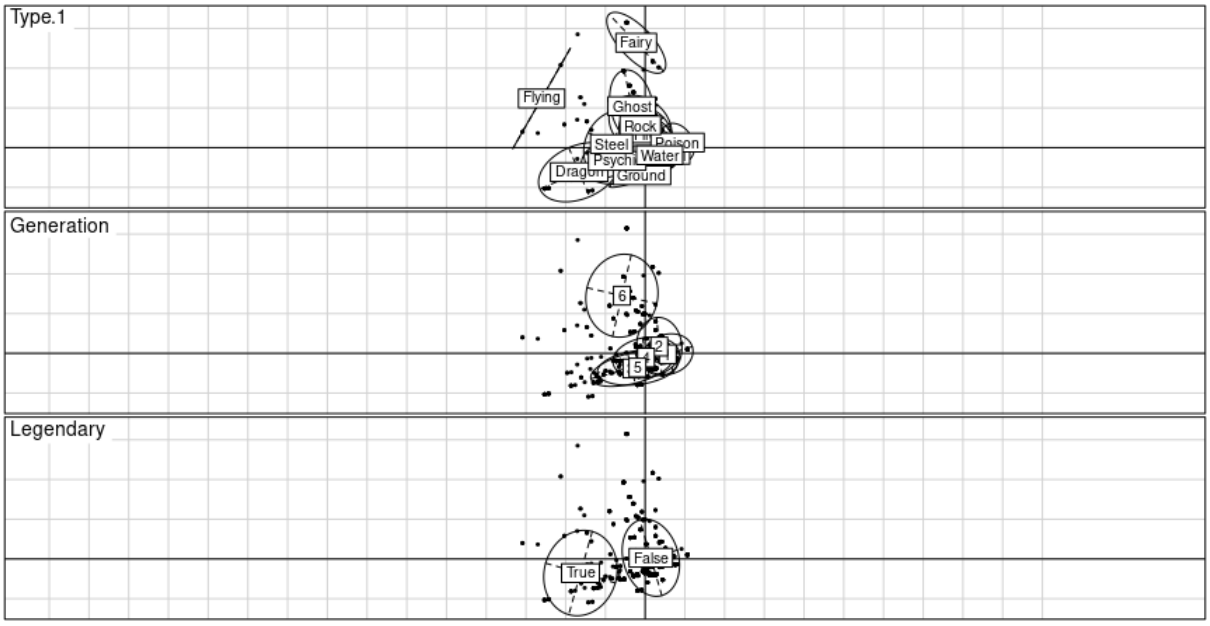
Nous pouvons affirmer, qu'il y a bien 23 valeurs propres.

9. **Affichez et interprétez les rapports de corrélation pour le premier et le deuxième axe (en utilisant la liste de l'objet res.acm.poke).**

```
score(res.acm.poke,xax=1, type="boxplot")
head(inertia.dudi(res.acm.poke)$tot)
scatter(res.acm.poke)
get_mca_var(res.acm.poke)$contrib
res.acm.poke$cr
```

```
6  res.acm.poke$c1
7
8  barplot(res.acm.poke$cr[,1], names.arg=row.names(res.acm.poke$cr),
9          las=2, main="Premiere axe", col = "blue")
10
11 barplot(res.acm.poke$cr[,2], names.arg=row.names(res.acm.poke$cr),
12         las=2, main="Deuxieme axe", col = "blue")
```





Les Pokémon légendaire sont principalement de type Flying, Dragon et steel. Ils sont de génération 5 et 6. Et légendaire qui vaut true.

	inertia <dbl>	cum <dbl>	cum(%) <dbl>
Ax1	0.4780316	0.4780316	6.235195
Ax2	0.4365931	0.9146247	11.929888
Ax3	0.4003682	1.3149929	17.152081
Ax4	0.3831707	1.6981637	22.149961
Ax5	0.3750273	2.0731909	27.041621
Ax6	0.3670084	2.4401993	31.828686

	Dim.1 <dbl>	Dim.2 <dbl>
Type.1.Bug	2.380832716	0.49680626
Type.1.Dark	0.323754130	0.21305281
Type.1.Dragon	16.088309569	2.65955846
Type.1.Electric	0.097106803	0.04140195
Type.1.Fairy	0.155394691	25.88925040
Type.1.Fighting	0.523887121	0.03361691
Type.1.Fire	0.002786429	1.33742074
Type.1.Flying	4.955197310	1.34241065
Type.1.Ghost	0.665179650	7.64453643
Type.1.Grass	0.402523767	0.19149017

1-10 of 26 rows

	RS1 <dbl>	RS2 <dbl>
Type.1	0.6285080	0.65023195
Generation	0.3154106	0.63418729
Legendary	0.4901764	0.02536001

3 rows

Nous remarquons que Type.1 est à l'origine de l'axe 1 et de l'axe 2.
En second lieu, Génération participe à la construction de l'axe 2.
Legendary, participe a la formation de l'axe 1.

	CS1 <dbl>	CS2 <dbl>
Type.1.Bug	0.91000864	-0.41569481
Type.1.Dark	-0.50064794	-0.40613312
Type.1.Dragon	-3.47364825	-1.41232746
Type.1.Electric	0.23014636	-0.15027601
Type.1.Fairy	-0.46838098	6.04562073
Type.1.Fighting	0.68240563	0.17286324
Type.1.Fire	0.03586146	0.78566603
Type.1.Flying	-5.45263091	2.83803874
Type.1.Ghost	-0.70631773	2.39445240
Type.1.Grass	0.37149448	-0.25622992

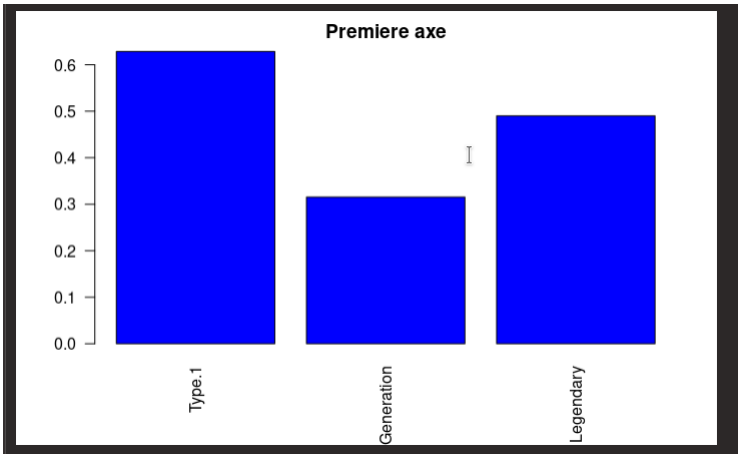
Le type Dragon a une influence négative de 3.47 sur le 1er axe.
Le type Flying a une influence négative de 5.45 sur le 1er axe.

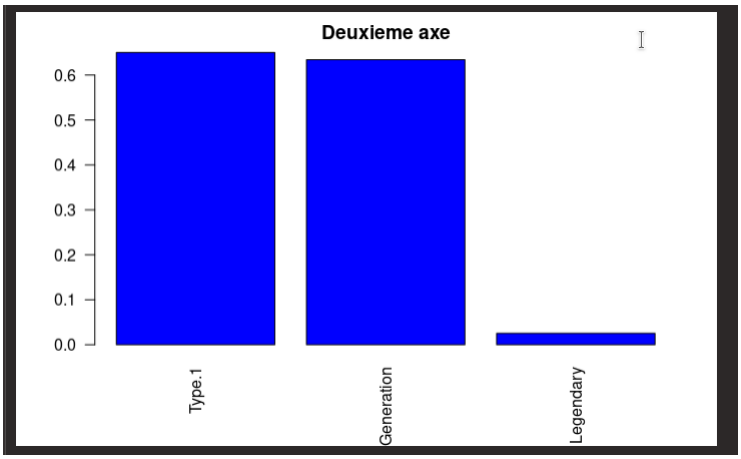
	CS1 <dbl>	CS2 <dbl>
Type.1.Ground	-0.16704856	-1.59504073
Type.1.Ice	-0.44176882	-0.49914287
Type.1.Normal	0.87339456	-0.41579856
Type.1.Poison	1.70554953	0.29361676
Type.1.Psychic	-1.47347453	-0.76194975
Type.1.Rock	-0.25442373	1.28837057
Type.1.Steel	-1.76587650	0.20678178
Type.1.Water	0.76162419	-0.43347227
Generation.1	1.19296323	-0.05482804
Generation.2	0.74382690	0.39157867

Le type Steel a une influence négative de 1.76 sur le 1er axe.

	CS1 <dbl>	CS2 <dbl>
Generation.3	-0.71872470	-0.86887925
Generation.4	0.04488168	-0.24526533
Generation.5	-0.40553347	-0.82494694
Generation.6	-1.22438115	3.32205055
Legendary.False	0.30113475	0.07167197
Legendary.True	-3.40513911	-0.81044453

L'attribut légendaire a une influence négative de 3.40 sur le 1er axe.
La 3e génération a une influence négative de 0.82 sur le 1er axe.
La 5e génération a une influence négative de 0.82 sur le 1er axe.
La 6e génération a une influence positive de 0.82 sur le 2e axe.





10. **Quelles modalités des trois facteurs décrivent le mieux le premier axe ?**

    Nous pouvons en conclure, que les modalités des trois facteurs qui décrivent le mieux l'axe 1 sont :
    - Type.1 = Dragon.
    - Generation = 1.
    - Legendary = True.

11. **À l'aide de la bibliothèque vcd, utilisez la fonction assocstats() sur votre sous jeu de données (attention il faut le transformer en tableau de contingence avant). Commentez.**

```
assocstats(table(poke.x))

t1 = table(poke.x[,1], poke.x[,2])
assocstats(t1)

t2 = table(poke.x[,1], poke.x[,3])
assocstats(t2)
```

```r

t3 = table(poke.x[,2], poke.x[,3])
assocstats(t3)
```

```
$`Legendary:False`
                     X^2 df   P(> X^2)
Likelihood Ratio 170.31 85 1.1698e-07
Pearson          177.69 85 1.6370e-08

Phi-Coefficient   : NA
Contingency Coeff.: 0.441
Cramer's V        : 0.22

$`Legendary:True`
                    X^2 df P(> X^2)
Likelihood Ratio 89.428 85  0.35017
Pearson             NaN 85      NaN

Phi-Coefficient   : NA
Contingency Coeff.: NaN
Cramer's V        : NaN

                     X^2 df   P(> X^2)
Likelihood Ratio 186.13 85 1.5924e-09
Pearson          184.65 85 2.4126e-09

Phi-Coefficient   : NA
Contingency Coeff.: 0.433
Cramer's V        : 0.215
                     X^2 df   P(> X^2)
Likelihood Ratio 76.183 17 1.8115e-09
Pearson          90.420 17 5.1186e-12

Phi-Coefficient   : NA
Contingency Coeff.: 0.319
Cramer's V        : 0.336
                    X^2 df P(> X^2)
Likelihood Ratio 10.9164  5 0.053063
Pearson           9.8768  5 0.078803

Phi-Coefficient   : NA
Contingency Coeff.: 0.11
Cramer's V        : 0.111
```

Règle de décision :

- Si $\rho \leq \alpha$: les variables présentent une association statistiquement significative donc on rejette H_0.
- Si $\rho > \alpha$: on ne possédez pas suffisamment de preuves pour conclure que les variables sont associées, donc on ne pas rejeter H_0.

Notre p valeur est plus petite que 0.05, quand Legendary est à False. Cela veut dire, qu'il n'existe aucun lien entre les Pokémon Légendaire, le type de Pokémon et la génération de Pokémon.

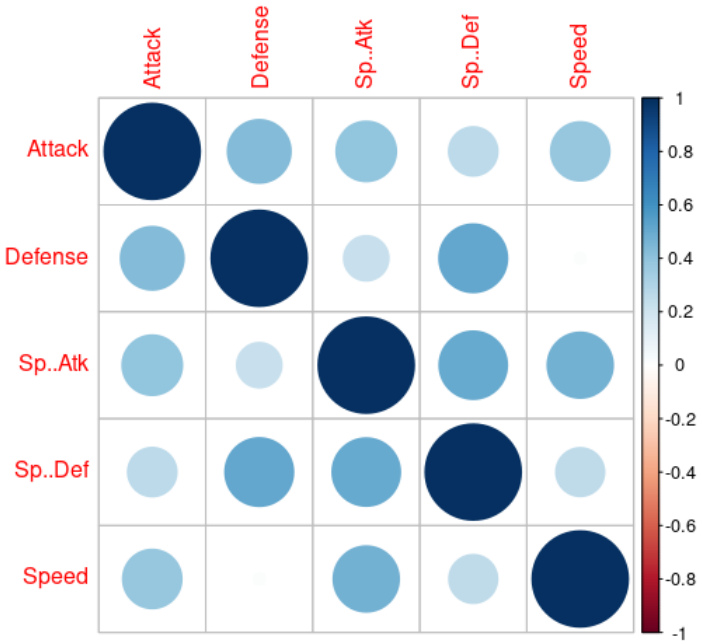
Notre p valeur est plus grande que 0.05, quand Legendary est à True. Cela veut dire, qu'il existe un lien entre les Pokémon Légendaire, le type de Pokémon et la génération de Pokémon.

Pour conclure, il est logique que les Pokémon communs (non légendaires) soient indépendants aux types et générations, car il y en a énormément.

12. **Affichez la matrice de corrélation sur les variables quantitatives suivantes : Attack, Defense, Sp..Atk, Sp..Def et Speed.**

```r
c = cor(poke[c("Attack", "Defense", "Sp..Atk", "Sp..Def", "Speed")])
corrplot(c)
```

	Attack	Defense	Sp..Atk	Sp..Def	Speed
Attack	1.0000000	0.4386871	0.3963618	0.2639896	0.3812397
Defense	0.4386871	1.0000000	0.2235486	0.5107466	0.0152266
Sp..Atk	0.3963618	0.2235486	1.0000000	0.5061214	0.4730179
Sp..Def	0.2639896	0.5107466	0.5061214	1.0000000	0.2591331
Speed	0.3812397	0.0152266	0.4730179	0.2591331	1.0000000



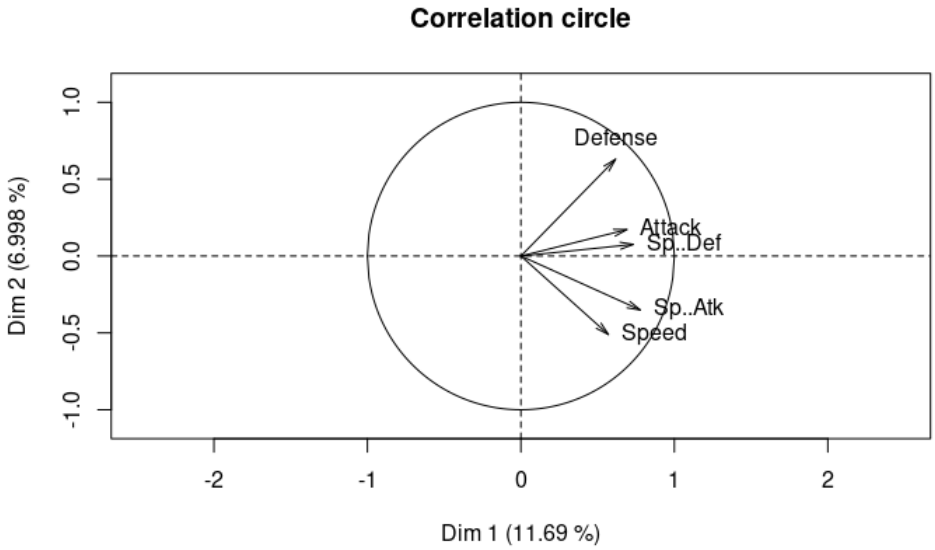
Les couples de variables suivant ont une forte corrélation positive :
- Defense et Attack.
- Sp..Def et Defense.
- Sp..Def et SP Atk.
etc.

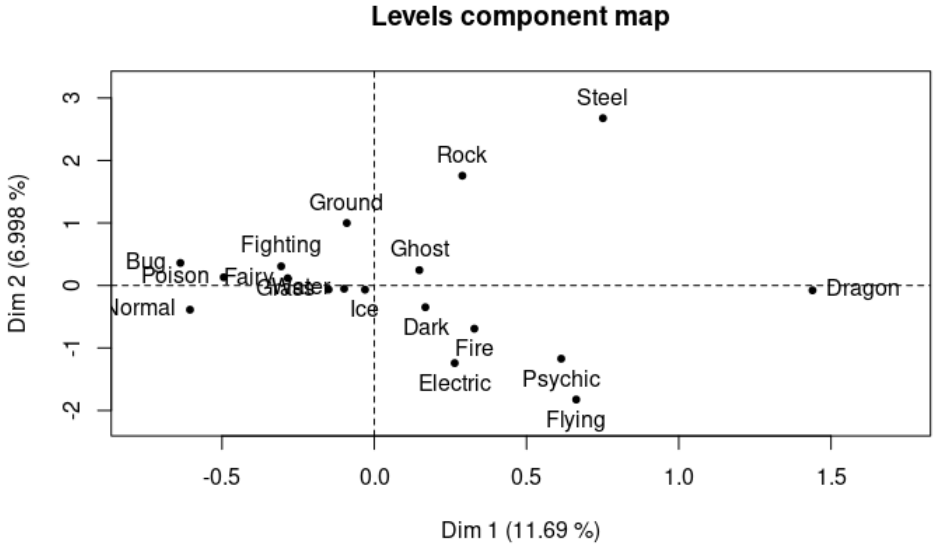
Le couple suivant n'a pas de corrélation :
Speed et Defense.

13. **Téléchargez la bibliothèque PCAmixdata et chargez-là.**

14. **Appliquez la fonction PCAmix() sur les variables quantitatives suivante : Attack,Defense,Sp..Atk,Sp..Def,Speed et la variable qualitative : Type1.**

```r
pcamix.temp <- PCAmix(subset(poke,select=c(7:11)),subset(poke,select=c(3)))
pcamix.poke <- PCAmix(poke[c("Attack","Defense","Sp..Atk","Sp..Def","Speed")],
                      poke[c("Type.1")])
```





Nous effectuons une analyse en composantes principales d'un ensemble de Pokémon décrits par un mélange de variables qualitatives et quantitatives. PCAmix comprend l'analyse en composantes principales ordinaire (PCA) et l'analyse des correspondances multiples (MCA) comme cas particuliers.

Les Pokémon de type Psychic, Fire et Electric ont tendances à être rapide.
Les Pokémon de type Steel et Rock ont tendance à être fort en défense.

15. **Affichez les valeurs propres.**

```
print(round(pcamix.temp$eig))
print(round(pcamix.poke$eig))
```

	Eigenvalue	Proportion	Cumulative
dim 1	3	12	12
dim 2	2	7	19
dim 3	1	6	24
dim 4	1	5	29
dim 5	1	5	34
dim 6	1	5	38
dim 7	1	5	43
dim 8	1	5	47
dim 9	1	5	52
dim 10	1	5	56
dim 11	1	5	61
dim 12	1	5	65
dim 13	1	5	70
dim 14	1	5	75
dim 15	1	5	79
dim 16	1	5	84
dim 17	1	5	88
dim 18	1	4	92
dim 19	1	3	95
dim 20	0	2	97
dim 21	0	2	99
dim 22	0	1	100

Nous affichons dans cette question les valeurs propres.

16. **Affichez les corrélations des variables quantitatives suivantes : Attack,Defense,Sp..Atk,Sp..Def,Speed.**

```
print(round(pcamix.temp$quanti.cor))
print(round(pcamix.poke$quanti.cor))
```

	dim 1	dim 2	dim 3	dim 4	dim 5
Attack	1	0	0	0	0
Defense	1	1	0	0	0
Sp..Atk	1	0	0	0	0
Sp..Def	1	0	0	0	0
Speed	1	-1	0	0	0

Nous affichons dans cette question les corrélations. Elles vont de -1 à 1.

0 signifie qu'il y a une absence de corrélation avec la dimension.
1 qu'il y a une corrélation positive forte avec la dimension.
-1 une corrélation négative forte avec la dimension.

17. **Affichez les coordonnées des modalités de la variable qualitative : Type1. Si vous n'arrivez pas à répondre à ces cinq questions, voici les commandes qui vous permettraient d'y répondre.**

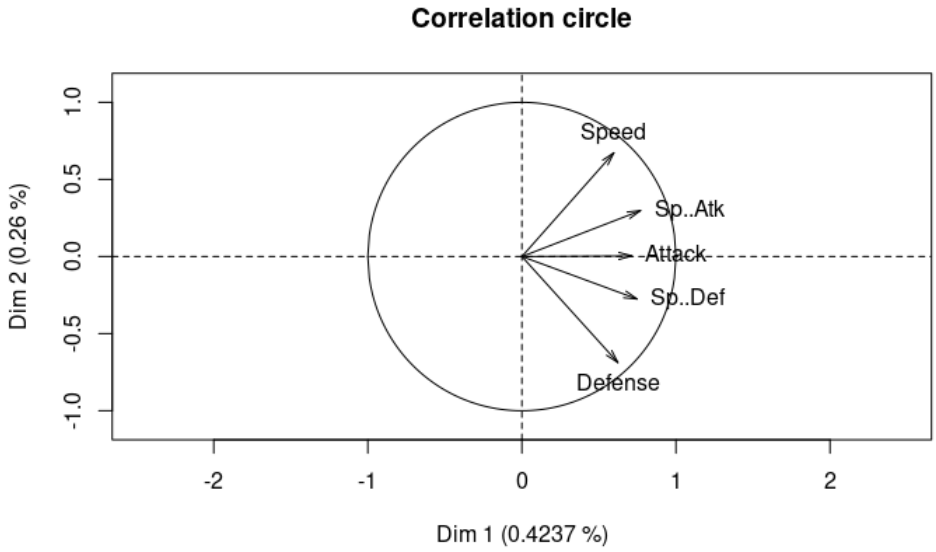
```
print(round(pcamix.temp$categ.coord))
print(round(pcamix.poke$categ.coord))
```

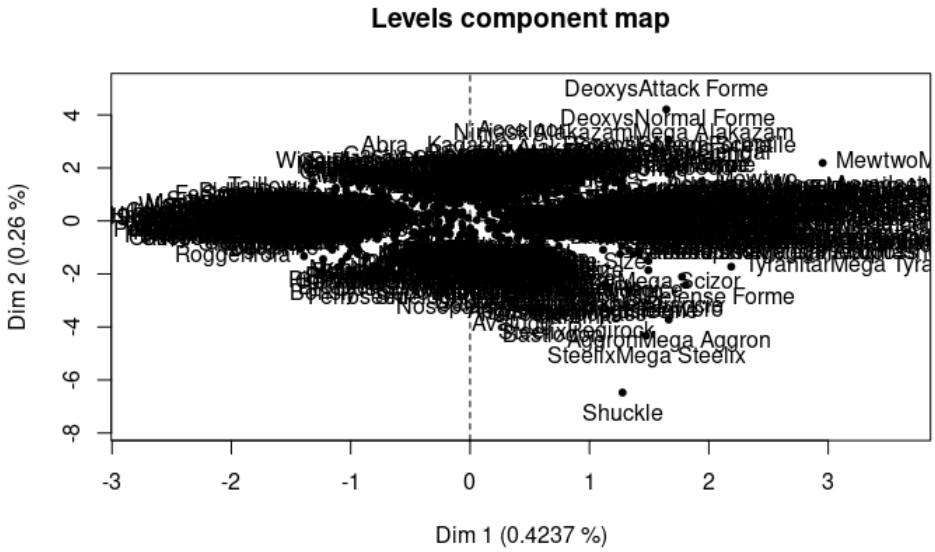
	dim 1	dim 2	dim 3	dim 4	dim 5
Bug	-1	0	0	-1	1
Dark	0	0	1	0	0
Dragon	1	0	1	1	1
Electric	0	-1	0	-2	-1
Fairy	0	0	-3	2	3
Fighting	0	0	2	2	1
Fire	0	-1	0	1	-1
Flying	1	-2	1	-3	-1
Ghost	0	0	-1	0	0
Grass	0	0	-1	1	-1
Ground	0	1	2	0	-1
Ice	0	0	-1	1	1
Normal	-1	0	1	-1	1
Poison	0	0	0	0	0
Psychic	1	-1	-1	-1	1
Rock	0	2	0	0	0
Steel	1	3	-1	-2	-1
Water	0	0	0	0	-1

Nous affichons dans cette question les modalités de la variable qualitative Type1.

18. **Réalisez cette A.C.P. mixte en remplaçant le facteur Type1 par le nom des pokemons.**

```
acp = PCAmix(subset(poke, select=7:11),subset(poke, select=2))
```





19. **Observez et commentez les coordonnées du pokemon Pikachu sur l'A.C.P. mixte.**

```
acp[["levels"]][["coord"]]["Pikachu",]
acp$quanti.cor
```

dim 1	dim 2	dim 3	dim 4	dim 5
-0.783122996	1.151944208	0.008491054	1.016673653	0.152459138

	dim 1	dim 2	dim 3	dim 4
Attack	0.7137138	0.004122036	0.63951379	-0.1517522
Defense	0.6230973	-0.687486286	0.14686705	0.1428119
Sp..Atk	0.7701656	0.297991012	-0.28278644	-0.4548817
Sp..Def	0.7475578	-0.275418689	-0.49617062	0.1630355
Speed	0.5970917	0.672958390	0.06827486	0.4149743

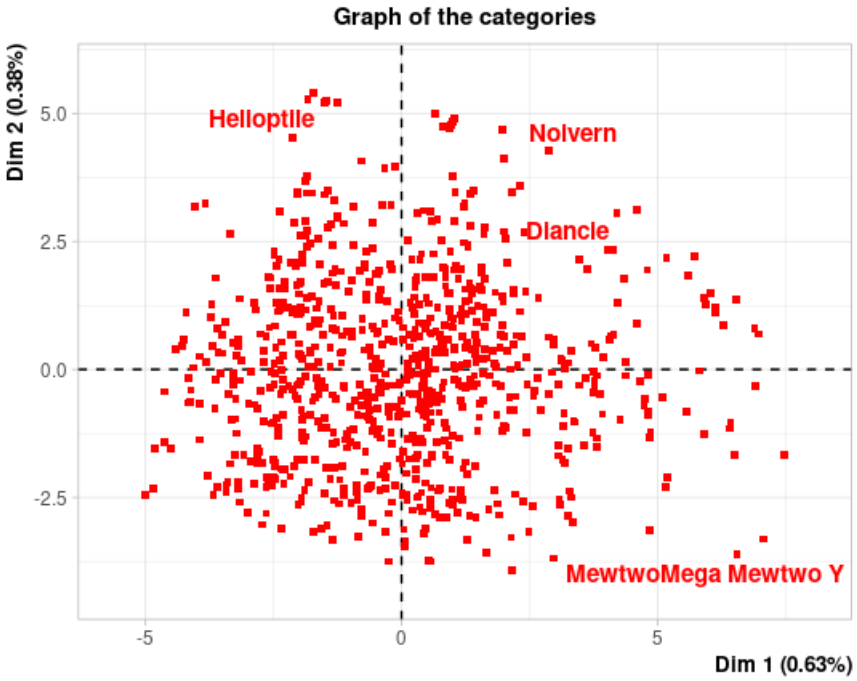
	dim 5
Attack	-0.2420517
Defense	0.3116844
Sp..Atk	0.1765239
Sp..Def	-0.3041973
Speed	0.1172321

Pikachu est fortement corrélé de manière positive dans l'axe 2 et l'axe 4. La commande acp$quanti.cor montre que pikachu n'a pas une bonne defense mais une bonne vitesse (en dim 2), la dim 4 montre qu'il n'a pas une bonne attaque spé mais une bonne vitesse.

20. **Appliquez la fonction FAMD() au jeu de données poke. Observez le résultat avec la fonction summary().**

```
fd = FAMD(poke, graph = TRUE, axes = c(1,2), ncp = 23)
summary(fd)
```

```
Eigenvalues
                       Dim.1 Dim.2 Dim.3 Dim.4 Dim.5 Dim.6
Variance               5.328 3.252 2.871 2.554 2.522 2.441
% of var.              0.628 0.383 0.339 0.301 0.297 0.288
Cumulative % of var.   0.628 1.012 1.350 1.652 1.949 2.237
                       Dim.7 Dim.8 Dim.9 Dim.10 Dim.11
Variance               2.338 2.325 2.302  2.288  2.250
% of var.              0.276 0.274 0.271  0.270  0.265
Cumulative % of var.   2.513 2.787 3.058  3.328  3.593
                      Dim.12 Dim.13 Dim.14 Dim.15 Dim.16
Variance               2.235  2.200  2.190  2.168  2.155
% of var.              0.264  0.259  0.258  0.256  0.254
Cumulative % of var.   3.857  4.116  4.375  4.630  4.884
                      Dim.17 Dim.18 Dim.19 Dim.20 Dim.21
Variance               2.125  2.119  2.079  2.068  2.046
% of var.              0.251  0.250  0.245  0.244  0.241
Cumulative % of var.   5.135  5.385  5.630  5.874  6.115
                      Dim.22 Dim.23
Variance               2.005  1.983
% of var.              0.236  0.234
Cumulative % of var.   6.351  6.585

Individuals (the 10 first)
                         Dist    Dim.1    ctr
1                     | 29.034 | -2.728  0.175
2                     | 28.977 | -1.336  0.042
```



Eigenvalues : Tableau avec les valeurs propres, le pourcentage de variance et le pourcentage cumulée de variance.
Individuals : matrice avec les coordonnées, le carré du cosinus (cos2) et la contribution de chaque dimension des valeurs propres.
Categories : matrice avec les coordonnées, le carré du cosinus (cos2), la contribution et le V de cramer de chaque pokemon.

21. **Expliquez ce qu'est la fonction FAMD() et son utilité.**

Le Multiple Factor Analysis for Mixed Data (AFDM), nous permet d'effectuer une analyse factorielle

multiple avec des données quantitatives et qualitatives. Nous pouvons donc utiliser l'AFDM sur des variables quantitatives et qualitatives, contrairement à une AFC ou ACP.