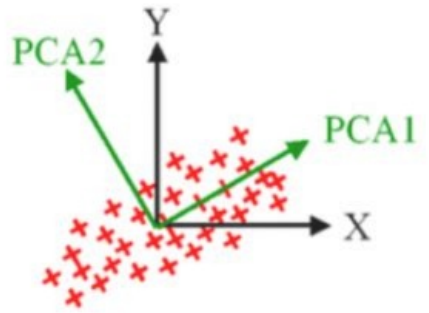
Réduction de dimension

Processus d'analyse de données qui consiste à réduire le nombre de variables (dimensions) d'un ensemble de données tout en préservant autant d'informations que possible. Il simplifie des ensembles de données complexes, améliore la visualisation des données, accélére les algorithmes d'apprentissage automatique, et facilite la compréhension des données.



Import modules

In [155…
```
import os
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from scipy.cluster.hierarchy import dendrogram, linkage
from scipy.stats import kendalltau, spearmanr, chi2_contingency, ttest_ind, bartlett
from pandas.plotting import scatter_matrix
from sklearn.decomposition import FactorAnalysis as FA
from sklearn.decomposition import PCA
from sklearn import metrics
from prince import CA, MCA, FAMD
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture
import seaborn as sns
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 15, 5
```

In [2]:
```
os.getcwd()
```

Out[2]:
```
'C:\\Users\\user\\OneDrive\\Perso\\Cours\\DU Python\\Eco-Python\\TP\\TPs - Corrigé\\TP_1'
```

L'Analyse en Composantes Principales ACP

l'ACP en bref est la creation des nouvelles variables qui sont des combinaison linaires des variables initales et d'avoir differentes axes sur lequelle raproccher les individus tout maximisant la variance (inertie)

Data set Base_freq

Le data set contient les statistiques de détentions par 100k habitantes victimes d'assaut, assassinat et viol dans les 50 départements des Etats Unies en 1973. Il contient aussi le pourcentage de la population qu'habite dans les secteurs urbains.

In [71]:
```
base = pd.read_csv("USAArrests.csv", delimiter = ";")
```

In [72]:
```
base = base.set_index(['Etat']) ; base
```

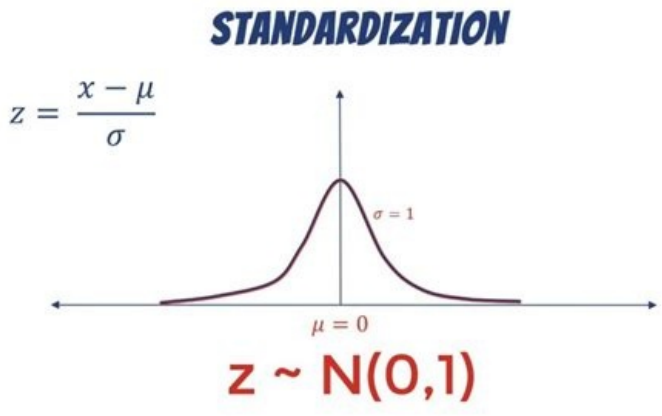
Out[72]:

Etat	Murder	Assault	UrbanPop	Rape
Alabama	13.2	236	58	21.2
Alaska	10.0	263	48	44.5
Arizona	8.1	294	80	31.0
Arkansas	8.8	190	50	19.5
California	9.0	276	91	40.6
Colorado	7.9	204	78	38.7
Connecticut	3.3	110	77	11.1
Delaware	5.9	238	72	15.8
Florida	15.4	335	80	31.9
Georgia	17.4	211	60	25.8
Hawaii	5.3	46	83	20.2
Idaho	2.6	120	54	14.2
Illinois	10.4	249	83	24.0
Indiana	7.2	113	65	21.0
Iowa	2.2	56	57	11.3
Kansas	6.0	115	66	18.0
Kentucky	9.7	109	52	16.3
Louisiana	15.4	249	66	22.2
Maine	2.1	83	51	7.8
Maryland	11.3	300	67	27.8
Massachusetts	4.4	149	85	16.3
Michigan	12.1	255	74	35.1
Minnesota	2.7	72	66	14.9
Mississippi	16.1	259	44	17.1
Missouri	9.0	178	70	28.2
Montana	6.0	109	53	16.4
Nebraska	4.3	102	62	16.5
Nevada	12.2	252	81	46.0
New Hampshire	2.1	57	56	9.5
New Jersey	7.4	159	89	18.8
New Mexico	11.4	285	70	32.1
New York	11.1	254	86	26.1
North Carolina	13.0	337	45	16.1
North Dakota	0.8	45	44	7.3
Ohio	7.3	120	75	21.4
Oklahoma	6.6	151	68	20.0
Oregon	4.9	159	67	29.3
Pennsylvania	6.3	106	72	14.9
Rhode Island	3.4	174	87	8.3
South Carolina	14.4	279	48	22.5
South Dakota	3.8	86	45	12.8
Tennessee	13.2	188	59	26.9
Texas	12.7	201	80	25.5
Utah	3.2	120	80	22.9
Vermont	2.2	48	32	11.2
Virginia	8.5	156	63	20.7
Washington	4.0	145	73	26.2
West Virginia	5.7	81	39	9.3
Wisconsin	2.6	53	66	10.8
Wyoming	6.8	161	60	15.6

¡Attention!

Normalisation de la base

La standardisation des données est la soustraction de la moyenne et en divisant par l'écart type.



La standardisation des données est importante car :

Comparabilité des données : Si les données sont de différentes sources ou avec différentes unités de mesure, la standardisation permet de rendre les données comparables. en les exprimant dans la même échelle.

Amélioration des performances des algorithmes :la convergence des algorithmes tels que la régression linéaire, la régression logistique et les méthodes basées sur les distances (comme les k-moyennes), peut ameliorer.

Réduction de la sensibilité à l'échelle : Certains algorithmes sont sensibles à l'échelle des données. Par exemple, les algorithmes basés sur la distance, comme les k-moyennes, attribueront plus de poids aux caractéristiques avec des échelles plus grandes. La standardisation élimine cette sensibilité en mettant toutes les caractéristiques à la même échelle.

Distribution normale : Peut aider à rendre la distribution des données plus proche d'une distribution normale, ce qui est une hypothèse commune dans de nombreuses méthodes statistiques.

In [75]:
```
sc = StandardScaler()
base_num_cr = sc.fit_transform(base)
```

In [76]:
```
acp = PCA()
res_acp = acp.fit_transform(base_num_cr)
# res_acp représente les points après l'ACP (dans le nouvel espace vectoriel), où chaque nombre est la coordonn
```

1. Analyse de l'ACP

In [77]:
```
acp.explained_variance_ratio_
# Le pourcentage de variance expliqué par chaque composante principale
```

Out[77]:
```
array([0.62006039, 0.24744129, 0.0891408 , 0.04335752])
```

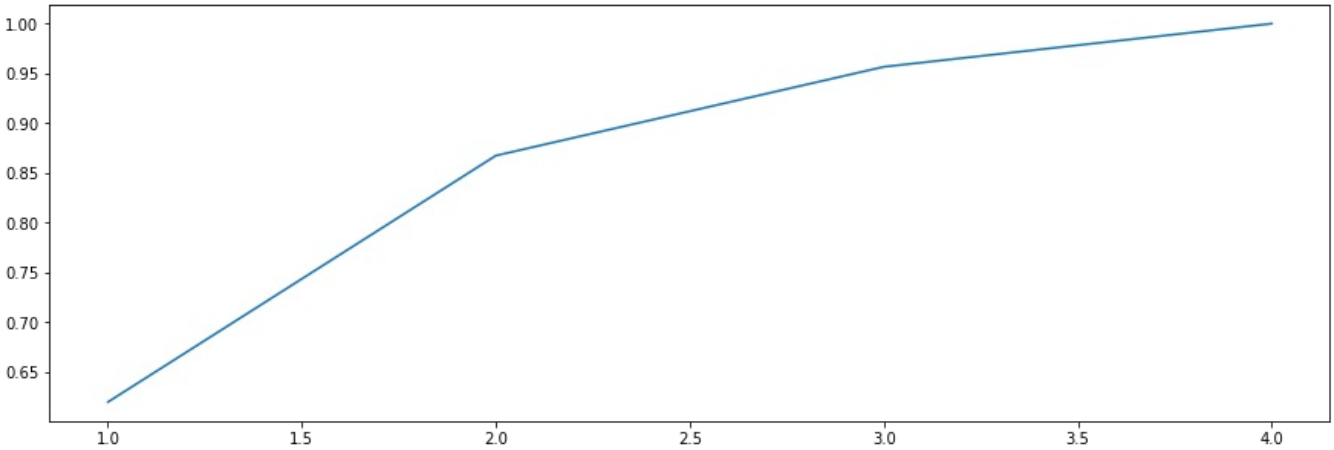
In [78]:
```
np.cumsum(acp.explained_variance_ratio_)
# L'inertie cumulée. Les deux premières variables expliquent 86 % de l'information.
```

Out[78]:
```
array([0.62006039, 0.86750168, 0.95664248, 1.        ])
```

In [79]:
```
p = base_num_cr.shape[1]
n = base_num_cr.shape[0]
```

In [80]:
```
plt.plot(np.arange(1,p+1), np.cumsum(acp.explained_variance_ratio_))
```

Out[80]:
```
[<matplotlib.lines.Line2D at 0x247f1fc5760>]
```

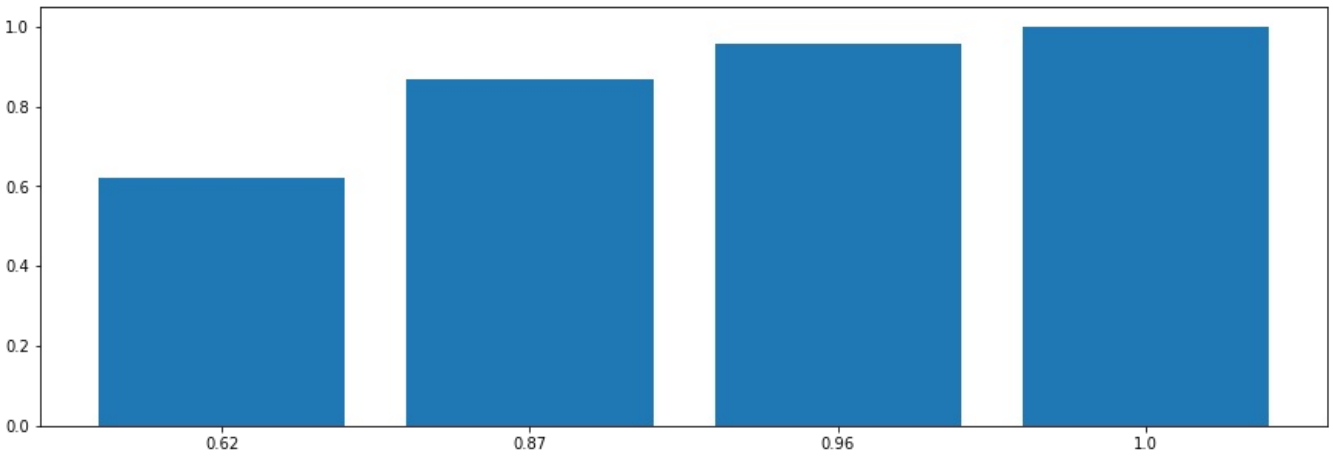


Les 3 lignes représentent l'inertie captée. La première ligne est la somme de la variance des deux premières variables, soit 0.62 + 0.24 = 0.86.

La deuxième ligne est la somme de 3 variables, équivalant à 0.95, et la troisième ligne est la somme de toutes les variables, soit 1, ce qui équivaut à 100 % de l'information.

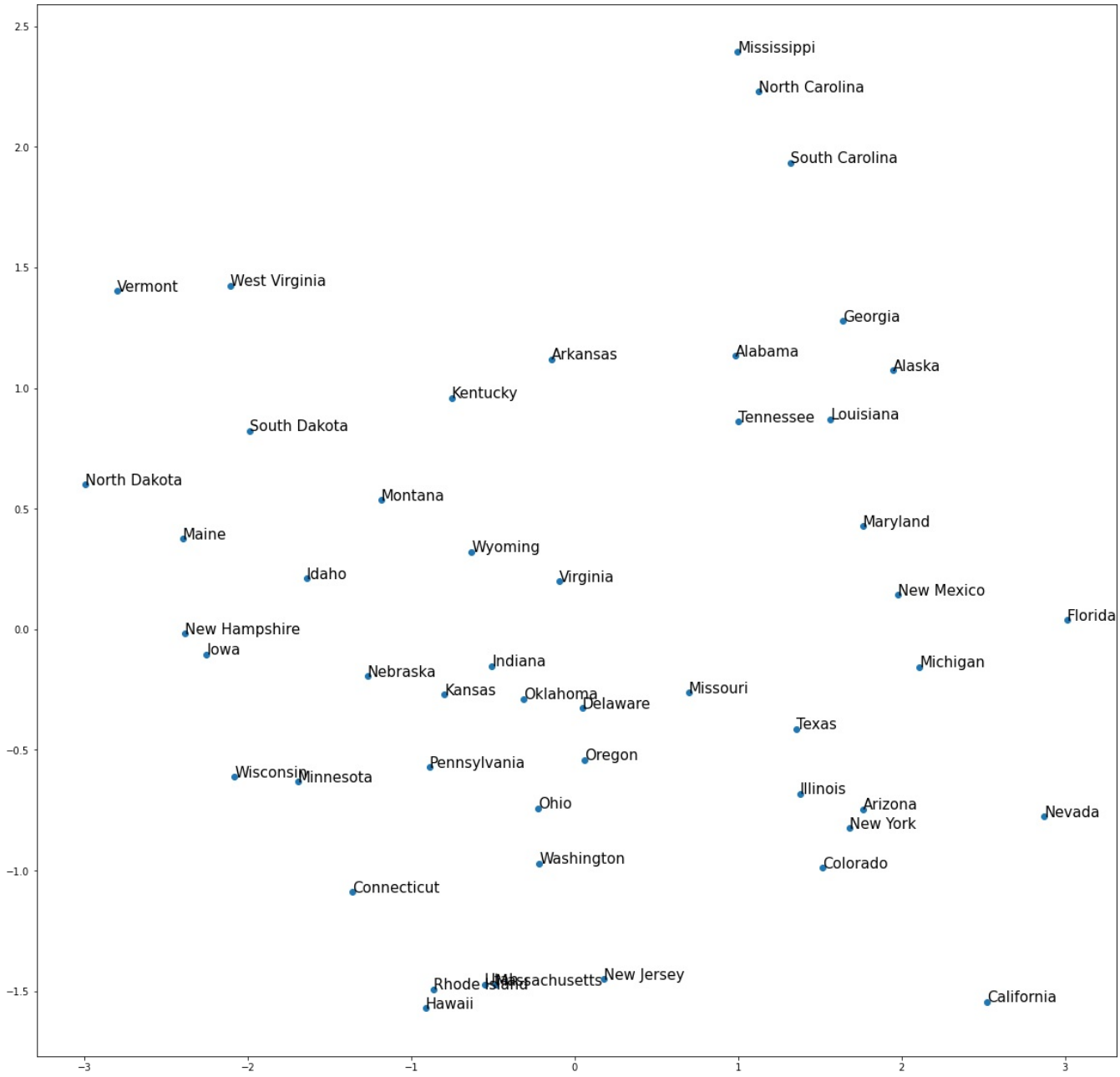
In [81]:
```
plt.bar(np.arange(len(acp.explained_variance_ratio_))+0.5,
        np.cumsum(acp.explained_variance_ratio_),
        tick_label=np.round(np.cumsum(acp.explained_variance_ratio_),2))
```

Out[81]:
```
<BarContainer object of 4 artists>
```



Le principe de ce graphique est le même, à ceci près que l'on peut observer l'inertie de la première variable

In [82]:
```
fig = plt.figure(figsize=(20,20))
plt.scatter(res_acp[:,0], res_acp[:,1]) # nuage de point
for i in range(len(res_acp[:,0])): # etiquettes des points
    plt.text(res_acp[i,0],res_acp[i,1], base.index[i], fontsize=15)
fig.savefig('ACP.pdf')
```



La figure du nouvel espace vectoriel montre la ressemblance entre les États en termes de criminalité.

Par exemple, le Mississippi, la Caroline du Nord et la Caroline du Sud d'un côté, et le Maryland, le Nouveau-Mexique et le Michigan de l'autre, se rapprochent et sont similaires. En revanche, le Mississippi ne présente pas de similitude avec Hawaï.

Toutefois, ce graphique ne permet pas d'évaluer la variation de la criminalité d'un État à un autre. Pour obtenir cette information, il est nécessaire d'obtenir une matrice de variance corrigée et un cercle de corrélations.

Matrice de variance corrigée

In [83]:
```
var_cor = (n-1)/n*acp.explained_variance_  # Variance corrigée
corvar = np.zeros((p,p))
var_cor_rac = np.sqrt(var_cor)
for k in range(p):
 corvar[:,k] = acp.components_[k,:] * var_cor_rac[k] #corr = vect propre. x val propre.
print(corvar)
```

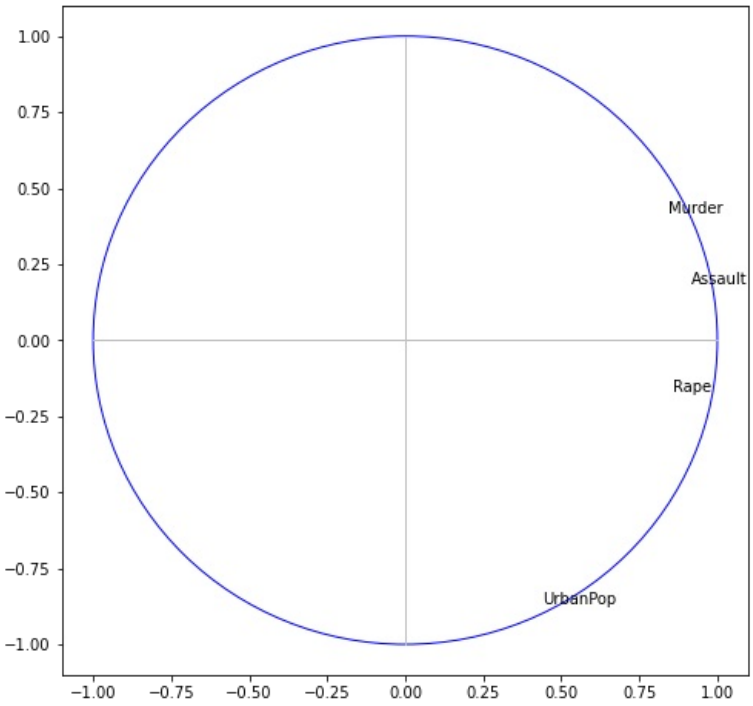
```
[[ 0.84397644  0.41603535 -0.20376     0.27037052]
 [ 0.91844324  0.18702113 -0.16011923 -0.30959159]
 [ 0.43811676 -0.86832819 -0.22572424  0.0557533 ]
 [ 0.85583939 -0.16646019  0.488319    0.03707412]]
```

Chaque colonne représente une composante et chaque ligne une variable (Mort, viol, vol). La première composante est fortement corrélée aux variables (Mort, viol et vol). Une corrélation plus élevée se traduit par une position plus à droite. Ainsi, les États qui se rapprochent de cette zone sont plus fortement liés à la criminalité.

La deuxième composante présente une corrélation négative avec la population. Ainsi, une valeur plus basse (-0,86) signifie une population plus élevée

Cercle des corrélations

In [84]:
```
fig, axes = plt.subplots(figsize=(8,8))
#affichage des étiquettes (noms des variables)
for j in range(p):
 plt.annotate(base.columns[j],(corvar[j,0],corvar[j,1]))
#ajouter les axes
plt.plot([-1,1],[0,0],color='silver',linestyle='-',linewidth=1)
plt.plot([0,0],[-1,1],color='silver',linestyle='-',linewidth=1)
#ajouter un cercle
cercle = plt.Circle((0,0),1,color='blue',fill=False)
axes.add_artist(cercle)
#affichage
plt.show()
```



En utilisant le cercle des corrélations et la matrice de variance corrigée, on peut conclure que la Californie a une population importante et est fortement liée à la criminalité. Elle est positionnée en bas et à droite sur le graphique.

Data set Base_freq

Ce jeu de données contient, entre autres, le nombre de sinistres, l'âge du conducteur et la classe du véhicule (Small, Medium et Large).

In [156…
```
base = pd.read_csv("base_freq.csv", delimiter = ";") ; base.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99997 entries, 0 to 99996
Data columns (total 18 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   PolNum      99997 non-null  int64
 1   CalYear     99997 non-null  int64
 2   Gender      99997 non-null  object
 3   Type        99997 non-null  object
 4   Category    99997 non-null  object
 5   Occupation  99997 non-null  object
 6   Age         99997 non-null  float64
 7   Group1      99997 non-null  int64
 8   Bonus       99997 non-null  int64
 9   Poldur      99997 non-null  float64
 10  Value       99997 non-null  float64
 11  Adind       99997 non-null  int64
 12  Group2      99997 non-null  object
 13  Density     99997 non-null  float64
 14  Value_num   99997 non-null  float64
 15  Expdays     99997 non-null  int64
 16  nb_sin      99997 non-null  float64
 17  chg_sin     99997 non-null  float64
dtypes: float64(7), int64(6), object(5)
memory usage: 13.7+ MB
```

Remplacement des modalités pour une meilleure visibilité sur le graphique

In [157…
```
base.replace({"Male":"H", "Female":"F"}, inplace=True)
```

Agrégation de la base en créant des individus synthétiques selon les variables Age et Sexe. soit les index seron visibles par age et au meme temps par sex. voir tableau ci-dessous

Pour creer ces profiles de risque on prend les variables quantitaives nb_sin;chg_sin, valeur, Pol-dur,Bonus

On va agreger tous les variables choissis. on peut mettre le min,le max, l'ecart,la somme

In [87]:
```
granularite = ['Age','Gender']
agreg_base = pd.pivot_table(base,
                            index = granularite,
                            values = ['chg_sin', 'nb_sin','Value','Poldur','Bonus'],
                            aggfunc = np.mean)
agreg_base
```

Out[87]:

Age	Gender	Bonus	Poldur	Value	chg_sin	nb_sin
18.0	**F**	-0.043478	5.230435	16378.318709	223.673246	0.253623
	H	0.150754	5.334673	16131.023412	477.919618	0.431156
19.0	**F**	2.810734	5.590395	16129.400087	187.163376	0.194915
	H	2.373225	5.305274	15658.153817	382.607677	0.412779
20.0	**F**	4.293059	5.191517	16687.004608	193.610861	0.215938
...	**...**	...	...	...	...	...
73.0	**H**	-27.632653	6.134694	16681.152290	60.804531	0.075510
74.0	**F**	-30.666667	6.225641	17123.531971	69.577231	0.092308
	H	-28.993289	6.635347	17148.486553	58.838076	0.073826
75.0	**F**	-28.750000	6.261905	16636.513361	64.602500	0.107143
	H	-28.685832	6.119097	16991.859611	55.117228	0.059548

116 rows × 5 columns

Standardisation des données

In [88]:
```
sc = StandardScaler()
base_num_cr = sc.fit_transform(agreg_base)
base_num_cr
```

Out[88]:
```
array([[ 8.81404856e-01, -8.52623835e-01, -2.49333912e-01,
         1.56280029e+00,  1.38809698e+00],
       [ 8.96912511e-01, -6.25957770e-01, -7.50231980e-01,
         4.60902425e+00,  3.36565581e+00],
       [ 1.10928770e+00, -6.98918419e-02, -7.53520033e-01,
         1.12536142e+00,  7.34141589e-01],
       [ 1.07435657e+00, -6.89886847e-01, -1.70803210e+00,
         3.46705508e+00,  3.16095340e+00],
       [ 1.22763780e+00, -9.37250910e-01,  3.75911180e-01,
         1.20261124e+00,  9.68320094e-01],
       [ 1.15150621e+00, -3.99588395e-01, -1.01066252e+00,
         3.67122117e+00,  3.64686277e+00],
       [ 1.27593355e+00, -8.40588875e-01, -1.24538942e+00,
         8.23893296e-01,  5.94709796e-01],
       [ 1.30319143e+00, -7.32969729e-01, -4.59257401e-01,
         3.19163551e+00,  3.30301085e+00],
       [ 1.36691549e+00, -4.54006190e-01,  1.87404203e-02,
         1.08242013e+00,  7.62933412e-01],
       [ 1.36806903e+00, -4.50017266e-01, -1.59496604e-01,
         2.99276566e+00,  2.88358773e+00],
       [ 1.27529613e+00, -3.51521913e-01, -1.78717628e+00,
         8.95097618e-01,  5.91965449e-01],
       [ 1.42098938e+00, -2.25845950e-01, -2.15485636e+00,
         2.24925556e+00,  2.47772608e+00],
       [ 1.31988366e+00, -9.49032432e-01, -5.60483533e-01,
         7.76134493e-01,  6.48420145e-01],
       [ 1.48352009e+00, -3.89624275e-01, -7.43209326e-01,
         2.37041003e+00,  2.54834915e+00],
       [ 1.16328114e+00, -7.38181628e-01, -9.78460541e-01,
         4.88446856e-01,  7.22313766e-01],
       [ 1.35016203e+00, -7.63647492e-01,  5.49298859e-02,
         1.83594179e+00,  2.25915955e+00],
       [ 1.64047982e+00, -3.34884274e-01, -5.54279125e-01,
         5.29141446e-01,  7.18078285e-01],
       [ 1.43150212e+00, -9.75813233e-01, -7.92996883e-01,
         1.11817101e+00,  1.70026837e+00],
       [ 1.46071968e+00, -5.69538154e-01,  4.81841521e-02,
         5.35684891e-01,  4.97516500e-01],
       [ 1.53727965e+00, -1.21444113e+00, -1.93368627e-01,
         1.29635194e+00,  1.91500838e+00],
       [ 1.26612920e+00, -3.15347343e-01, -1.85389591e-01,
         3.21806632e-01,  3.87673491e-01],
       [ 1.19631983e+00, -6.19155431e-01, -1.03511417e+00,
         9.73493127e-01,  1.45226049e+00],
       [ 1.19493785e+00, -5.04547904e-01, -1.84876665e+00,
         1.96187358e-01,  3.41370765e-01],
```

```
[ 1.28625608e+00, -6.75284546e-01, -2.59351163e-01,
  1.41825959e+00,  1.39326559e+00],
[ 1.15609786e+00, -1.00282905e+00, -2.12159541e+00,
 -2.34418430e-01, -6.53473231e-02],
[ 9.97997768e-01, -9.53333990e-01, -5.35000432e-01,
  4.65279207e-01,  7.20637497e-01],
[ 8.96580620e-01, -5.29444587e-01, -3.93585309e-01,
 -3.23618790e-01, -1.19011683e-01],
[ 9.95687378e-01, -5.31883604e-01, -1.69973864e-01,
  9.71064728e-02,  4.57796532e-01],
[ 9.93833357e-01, -5.73179375e-01, -3.40506103e-02,
 -3.03224917e-01,  6.44332479e-03],
[ 6.86347139e-01, -5.84340915e-01,  6.39262010e-03,
 -1.11602627e-01,  1.42495776e-01],
[ 1.11463411e+00, -4.88822102e-01, -1.07641777e+00,
 -1.71240479e-01,  9.70182832e-02],
[ 8.84876198e-01, -9.74715470e-01, -1.05635017e-01,
  5.99034863e-02,  1.64804875e-01],
[ 7.84309925e-01, -6.93971488e-01,  8.03135462e-01,
 -2.15690092e-01, -1.31582754e-01],
[ 7.22306699e-01, -1.21825198e+00,  3.01995616e-01,
 -1.12247073e-01, -4.55124292e-03],
[ 1.00871103e+00, -7.98985056e-01, -1.68429157e+00,
 -2.67041778e-01, -1.07169141e-01],
[ 8.95845972e-01, -5.26252224e-01,  8.00912263e-02,
 -1.63970365e-01, -6.39195607e-03],
[ 6.69134220e-01, -5.72487678e-01, -4.63104400e-01,
 -5.34902179e-01, -4.10922383e-01],
[ 6.32080532e-01, -6.85901512e-01,  1.49213134e-01,
 -2.56277066e-01, -1.43605282e-01],
[ 3.62176939e-01, -1.96058922e-01, -6.78579303e-01,
 -4.63920130e-01, -3.03669893e-01],
[ 7.60747389e-01, -3.65929140e-01,  4.47931020e-01,
 -1.91680656e-01, -2.11241704e-01],
[ 7.75628909e-01, -1.27753951e-01,  6.28910873e-01,
 -3.57142343e-01, -1.07683672e-01],
[ 6.07190325e-01, -4.96308032e-01,  2.04797043e-01,
 -1.52692111e-01, -9.73106916e-02],
[ 7.36577383e-01, -5.24076823e-01, -6.31870409e-01,
 -3.93337764e-01, -2.73922877e-01],
[ 8.02576364e-01, -3.86046452e-01,  2.73926897e-01,
 -2.10631821e-01, -1.13791836e-01],
[ 6.84872395e-01, -4.72092327e-01,  3.90335493e-01,
 -4.22738483e-01, -3.72439916e-01],
[ 6.02944089e-01, -1.01475290e+00,  1.02118161e+00,
 -3.08679343e-01, -1.28130311e-01],
[ 5.44010664e-01, -1.03819688e+00, -5.16614134e-01,
 -4.41794933e-01, -3.09178908e-01],
[ 3.02186576e-01, -9.41764539e-01,  1.52971451e-01,
 -2.95086790e-01, -1.64001443e-01],
[ 6.11235019e-01, -5.87205675e-01,  6.28365449e-01,
 -4.54502370e-01, -3.13466701e-01],
[ 4.17236783e-01, -9.24452088e-01,  8.03590165e-01,
 -1.49757900e-01, -2.65706024e-02],
[ 3.23347580e-01, -3.97542628e-01, -7.13989559e-01,
 -2.71379941e-01, -1.33529655e-01],
[ 3.10813703e-01, -5.49154080e-01, -2.19793622e-01,
 -4.08503252e-01, -2.11953021e-01],
[ 2.15522655e-01, -6.53353289e-01, -1.77584025e-01,
 -1.11884185e-01,  1.41325194e-02],
[ 3.33248329e-01, -5.42477668e-01,  1.92830693e-01,
 -3.31535786e-01, -1.46856075e-01],
[ 7.27415052e-01, -8.86365348e-01, -3.40319318e-01,
 -2.60391466e-01, -6.46293857e-04],
[ 3.13028179e-01, -5.46796669e-01, -5.20721840e-01,
 -1.63685257e-01,  1.42333415e-02],
[-1.93130958e-01,  6.62850468e-01,  4.48789338e-01,
 -3.51247805e-01, -3.10129013e-01],
[-3.42670274e-01,  1.96023576e-01,  1.33280315e-02,
 -3.13963481e-01, -2.34924334e-01],
[-2.84684740e-01, -6.01445627e-01, -4.30058218e-01,
 -2.89060612e-01, -2.17761729e-01],
[-3.26349336e-01, -3.53466156e-01,  1.35574712e+00,
 -2.04475398e-01, -1.27014965e-01],
[-4.30979541e-01, -3.49181449e-01, -1.52995828e-01,
 -5.95600519e-01, -6.50041179e-01],
[-3.91049760e-01, -4.53733078e-01,  5.37507707e-01,
 -2.42002360e-01, -2.46739124e-01],
[-4.28523491e-01, -1.01721862e-01,  1.63475204e-01,
 -4.59411328e-01, -4.08322938e-01],
[-2.33407864e-01, -4.76461526e-01,  2.86631444e-01,
 -3.79228247e-01, -2.76718843e-01],
[-3.69943797e-01, -4.49357449e-01,  4.25333629e-01,
 -6.55910834e-01, -6.73615461e-01],
[-4.00925428e-01, -2.42621092e-01, -1.49639811e-01,
 -3.08719213e-01, -4.13296866e-01],
[-6.72087304e-01, -1.52485247e-01,  4.81001463e-01,
 -4.43534288e-01, -2.69382338e-01],
[-4.63122151e-01, -6.72311876e-01,  6.04076013e-01,
```

```
 -4.69623916e-01, -5.11963055e-01],
[-2.21352804e-01, -9.34523281e-01,  6.07021321e-02,
 -6.00328378e-01, -4.40083183e-01],
[-4.47884255e-01, -3.75935854e-01, -4.06454286e-01,
 -3.78521980e-01, -3.83082076e-01],
[-2.28971937e-01, -3.38391196e-01,  2.18966729e-01,
 -5.81247649e-01, -5.31595784e-01],
[-6.02463134e-01, -1.86589882e-01,  6.38245834e-01,
 -5.18463480e-01, -4.22413080e-01],
[-3.95190196e-01, -1.66402819e-01, -7.82426077e-01,
 -4.74836024e-01, -4.57530407e-01],
[-3.17369872e-01, -4.09459578e-01, -2.12728103e-01,
 -2.61936752e-01, -4.13623866e-01],
[-5.05539054e-01, -3.95008362e-01, -9.84057477e-01,
 -6.42879028e-01, -6.11097409e-01],
[-7.84905144e-01, -1.18239280e+00,  4.94412204e-01,
 -5.43221586e-01, -5.82888944e-01],
[-7.03356278e-01, -8.29953806e-01,  6.60441831e-01,
 -6.07299547e-01, -5.50705541e-01],
[-5.35174973e-01, -6.78780576e-01,  5.42488527e-01,
 -4.97128278e-01, -3.98651425e-01],
[-1.87930249e-01, -1.00114698e-01, -1.13314743e+00,
 -2.35786542e-01, -4.98492938e-01],
[-3.93688001e-01, -2.60156948e-01, -2.12668354e-01,
 -6.52641760e-01, -5.46430815e-01],
[-3.53757941e-01, -1.98332963e-01, -2.25834296e-01,
 -6.84882999e-01, -3.76175373e-01],
[-5.15673772e-01, -7.56884772e-01,  9.45804639e-01,
 -4.43788235e-01, -6.66078892e-01],
[-4.38513662e-01, -9.48591629e-01,  2.15820406e+00,
 -3.12811698e-01, -2.77354639e-01],
[-5.00689196e-01,  8.74982520e-02,  7.93179147e-01,
 -5.48794290e-01, -4.83500707e-01],
[-7.63151210e-01, -6.88041539e-01, -6.49720655e-01,
 -8.44168803e-01, -8.20008593e-01],
[-4.84291345e-01,  4.35136702e-01, -9.24453916e-01,
 -4.99802369e-01, -4.79657201e-01],
[-1.47606258e+00,  1.20099304e+00,  5.70453685e-01,
 -5.85736004e-01, -8.56170076e-01],
[-1.23243069e+00,  1.81342142e+00, -8.89812213e-01,
 -6.19274761e-01, -8.75836879e-01],
[-1.27314168e+00,  1.06502682e+00,  3.21779706e-01,
 -7.38157038e-01, -8.31657450e-01],
[-1.13505015e+00,  8.37066436e-01,  1.75578508e+00,
 -3.82709198e-01, -6.69979093e-01],
[-1.33195105e+00,  1.55358644e+00, -9.65727765e-01,
 -6.87057227e-01, -9.06610197e-01],
[-1.29799265e+00,  1.76720172e+00, -1.31866226e+00,
 -7.11335921e-01, -1.03303508e+00],
[-1.45025479e+00,  2.02884192e+00, -2.17433427e-01,
 -9.04629455e-01, -1.07771820e+00],
[-1.32510776e+00,  1.80244088e+00,  1.97169714e-01,
 -4.65566401e-01, -7.75909530e-01],
[-1.28271864e+00,  2.37532141e+00,  3.83585043e+00,
 -7.39649590e-01, -1.01043999e+00],
[-1.30187668e+00,  1.12280320e+00,  1.27992516e+00,
 -7.91565022e-01, -9.62600539e-01],
[-1.09074839e+00,  1.28108442e+00, -9.97853612e-01,
 -4.91799281e-01, -9.89853618e-01],
[-1.33747149e+00,  1.55760195e+00, -3.40169322e-01,
 -5.15849297e-01, -8.29456041e-01],
[-1.25227856e+00,  8.49770264e-02,  9.61026157e-01,
 -6.63476984e-01, -5.93171437e-01],
[-1.41632260e+00,  1.44977585e+00, -1.24336949e+00,
 -3.92579300e-01, -7.92632466e-01],
[-1.23641904e+00,  2.14914162e+00,  2.38459872e+00,
 -8.12588026e-01, -9.24365337e-01],
[-1.24569236e+00,  1.10669584e+00,  1.05409448e+00,
 -4.55452526e-01, -6.73808246e-01],
[-1.33954621e+00,  6.04422836e-01, -2.07318225e+00,
 -4.83552672e-01, -6.61185129e-01],
[-1.35220373e+00,  1.50241019e+00, -2.00970510e-01,
 -6.66329429e-01, -8.37245643e-01],
[-1.18699453e+00,  2.19072084e+00,  1.97228788e+00,
 -7.10238767e-01, -8.24392799e-01],
[-1.28892902e+00,  1.80239338e+00,  2.21743595e+00,
 -5.80603096e-01, -9.86068045e-01],
[-1.34703919e+00,  2.40223970e+00,  2.23347484e+00,
 -6.78307883e-01, -7.40849314e-01],
[-1.23361079e+00,  1.20197630e+00, -5.64708095e-01,
 -7.42625817e-01, -8.82307406e-01],
[-1.46660165e+00,  1.70448833e+00,  2.12811464e+00,
 -4.91356455e-01, -5.72363595e-01],
[-1.38392092e+00,  2.12265591e+00,  1.12736832e+00,
 -6.51972637e-01, -7.60481731e-01],
[-1.21916571e+00,  2.56038464e+00, -7.64398368e-01,
 -3.84098986e-01, -1.92083287e-01],
[-1.32133902e+00,  1.11368118e+00,  3.64057276e-01,
 -3.88592696e-01, -5.95926943e-01],
```

```
       [-1.56357733e+00,  1.31144511e+00,  1.26009990e+00,
        -2.83483588e-01, -4.08817516e-01],
       [-1.42997336e+00,  2.20234732e+00,  1.31064555e+00,
        -4.12153548e-01, -6.14693045e-01],
       [-1.41054898e+00,  1.39030035e+00,  2.73640864e-01,
        -3.43087758e-01, -2.43566667e-01],
       [-1.40542572e+00,  1.07976481e+00,  9.93396768e-01,
        -4.56734466e-01, -7.73729194e-01]])
```

In [89]:
```
acp = PCA()
res_acp = acp.fit_transform(base_num_cr)
res_acp
```

Out[89]:
```
array([[ 2.31368960e+00,  6.44815854e-01, -2.77327008e-01,
         2.74578597e-01,  1.17968500e-01],
       [ 4.84438504e+00,  3.02906057e+00,  4.75974511e-01,
         1.02079217e+00,  6.46547844e-01],
       [ 1.73310397e+00,  3.17014629e-01,  4.15546435e-01,
        -4.84011207e-01,  3.12258509e-01],
       [ 4.61753375e+00,  1.86449834e+00,  1.13353742e+00,
         6.88183240e-01,  5.64065943e-02],
       [ 1.93681464e+00,  4.05949002e-01, -9.62933951e-01,
        -9.94342070e-02,  2.39971116e-01],
       [ 4.66010402e+00,  2.61206394e+00,  6.64939829e-01,
         5.25978848e-01, -1.62181586e-01],
       [ 2.07023457e+00, -5.75903478e-01,  4.23237296e-01,
        -2.40308763e-01,  2.52302765e-01],
       [ 4.29138693e+00,  2.21467078e+00, -2.97386351e-02,
         4.30114686e-01, -1.90358499e-01],
       [ 1.76438165e+00,  3.49284738e-01, -4.84543501e-01,
        -5.12136110e-01,  3.10681920e-01],
       [ 3.80638162e+00,  2.19824195e+00, -1.88934038e-01,
         1.21369773e-01, -7.53056907e-03],
       [ 2.07837372e+00, -4.85638920e-01,  1.11127817e+00,
        -4.75596363e-01,  2.80991491e-01],
       [ 3.82019751e+00,  9.25435532e-01,  1.54513577e+00,
        -1.61559174e-01, -2.19122015e-01],
       [ 1.92160183e+00, -3.68610264e-01, -2.24603069e-01,
        -2.49463169e-01,  1.96532808e-01],
       [ 3.55992080e+00,  1.49273024e+00,  2.60463270e-01,
        -1.47659472e-01, -1.64322994e-01],
       [ 1.79226744e+00, -5.08513856e-01,  2.44704017e-01,
        -2.81707732e-01, -7.60885081e-02],
       [ 2.98758162e+00,  1.23100564e+00, -5.95262653e-01,
        -1.30805402e-02, -2.96763992e-01],
       [ 1.75031323e+00, -1.93785094e-01, -5.27591586e-02,
        -8.95679139e-01, -5.13236143e-04],
       [ 2.75942312e+00,  1.25352428e-01, -4.16687501e-02,
        -1.33230709e-01, -3.49080060e-01],
       [ 1.45349662e+00, -1.29070934e-01, -6.26350408e-01,
        -6.68724618e-01,  1.61428644e-01],
       [ 2.91082626e+00,  3.99553012e-01, -6.74136918e-01,
        -4.87319336e-02, -3.65903496e-01],
       [ 1.17059352e+00, -1.99142261e-01, -2.79501383e-01,
        -7.05472342e-01,  7.06227045e-02],
       [ 2.38041149e+00,  9.67742138e-02,  3.74218991e-01,
        -1.93658694e-01, -2.99128166e-01],
       [ 1.65954057e+00, -1.05857173e+00,  1.06269288e+00,
        -5.01437767e-01,  1.27822574e-03],
       [ 2.38196567e+00,  5.75509200e-01, -3.01930650e-01,
        -1.75671482e-01,  5.81579476e-02],
       [ 1.51672630e+00, -1.84493245e+00,  1.04863213e+00,
        -3.13114440e-01,  2.67075355e-02],
       [ 1.64262421e+00, -4.24131262e-01, -1.90558261e-01,
        -5.50009816e-02, -9.71321275e-02],
       [ 5.73860310e-01, -8.82412702e-01, -1.53312724e-01,
        -4.75673902e-01, -1.52444004e-02],
       [ 1.04613779e+00, -3.44687034e-01, -3.38976937e-01,
        -4.13125184e-01, -1.52973065e-01],
       [ 5.98692107e-01, -7.17115975e-01, -5.02825510e-01,
        -5.26895715e-01, -7.90284682e-02],
       [ 5.96555272e-01, -4.78322684e-01, -4.50450031e-01,
        -2.24065872e-01, -8.68690335e-02],
       [ 1.06714887e+00, -1.01500064e+00,  4.03734175e-01,
        -5.85766524e-01, -6.00883102e-02],
       [ 9.82192514e-01, -6.87316433e-01, -5.67958577e-01,
        -1.16503802e-01,  4.15676778e-02],
       [ 2.46495124e-01, -4.07135531e-01, -1.21180080e+00,
        -3.24925095e-01,  7.10254811e-02],
       [ 7.00752093e-01, -7.84929188e-01, -9.97207806e-01,
         7.63445719e-02,  4.76139384e-02],
       [ 1.18417923e+00, -1.55318009e+00,  8.02211123e-01,
        -3.41989202e-01,  1.90143077e-02],
       [ 5.53995786e-01, -5.53209805e-01, -5.43816302e-01,
        -4.51667458e-01,  1.14485412e-02],
       [ 2.50734624e-01, -1.12123885e+00, -7.16759159e-02,
        -3.43971822e-01,  3.77825780e-02],
       [ 3.50564493e-01, -6.63586151e-01, -6.14990556e-01,
        -1.90423421e-01,  2.88988910e-02],
```

```
[ 1.02307058e-01, -8.45818552e-01,  3.64435211e-01,
 -2.77570851e-01, -4.22379062e-02],
[ 1.86934716e-01, -3.83800430e-01, -7.53173182e-01,
 -4.77740724e-01,  1.29977137e-01],
[ 1.42955766e-02, -2.23086077e-01, -8.16562992e-01,
 -6.57518899e-01, -5.93178380e-02],
[ 3.16701277e-01, -4.53539457e-01, -5.62936670e-01,
 -2.52562231e-01,  5.51789818e-02],
[ 4.55879744e-01, -1.04446232e+00,  8.76264114e-02,
 -3.72404913e-01,  3.34609226e-02],
[ 3.12056407e-01, -4.44516142e-01, -6.25878417e-01,
 -4.86290999e-01,  4.83160817e-02],
[ 1.83073128e-02, -6.42072619e-01, -7.50000034e-01,
 -4.22183897e-01,  9.14919420e-02],
[ 1.72794037e-01, -5.06167668e-01, -1.50358062e+00,
 -3.09491186e-02, -1.09086832e-03],
[ 4.88934583e-01, -1.28470832e+00, -1.97363028e-01,
  4.96505510e-02,  2.26948128e-02],
[ 2.58469860e-01, -7.62067471e-01, -6.53146386e-01,
  2.03342479e-01, -1.04049247e-02],
[-3.34204736e-02, -5.84815037e-01, -9.88778760e-01,
 -3.10935561e-01,  2.46120232e-02],
[ 2.40130206e-01, -3.75244909e-01, -1.21854062e+00,
  1.19452232e-01,  2.72721065e-03],
[ 3.53461933e-01, -7.90162635e-01,  3.28016416e-01,
 -7.29463988e-02, -4.05445039e-02],
[ 1.45273454e-01, -7.76604927e-01, -1.68956806e-01,
 -3.74502167e-02, -6.41488614e-02],
[ 3.82065417e-01, -5.42463035e-01, -2.05714922e-01,
  1.81980336e-01, -4.35754635e-02],
[ 9.16315405e-02, -5.39935187e-01, -5.17846319e-01,
 -5.38410362e-02, -5.48891326e-02],
[ 7.06218725e-01, -9.39913230e-01, -3.12221731e-01,
 -1.18089386e-01, -6.91973945e-02],
[ 4.72992807e-01, -6.72455548e-01,  1.06493446e-01,
  4.88496741e-02, -7.39872949e-02],
[-8.32121984e-01,  2.81176804e-01, -6.16401800e-02,
 -3.47863667e-01, -3.66219960e-02],
[-5.23676401e-01, -7.51553900e-02,  1.39960656e-01,
  6.76831684e-02, -7.57165968e-02],
[-1.07870767e-02, -6.98412264e-01,  1.46106241e-01,
  4.97202126e-01, -4.74919173e-02],
[-6.16066962e-01,  2.66423889e-01, -1.24869726e+00,
  3.39219605e-01, -5.09949174e-02],
[-6.39723542e-01, -7.57331380e-01,  3.76246690e-02,
  3.48611935e-01,  5.12772727e-02],
[-4.25170645e-01, -1.82009601e-01, -5.82288554e-01,
  4.62813079e-01, -1.26684469e-04],
[-6.51877779e-01, -3.13420234e-01, -1.11413115e-01,
  2.47272518e-01, -4.19140801e-02],
[-3.37419972e-01, -4.18734093e-01, -4.28616822e-01,
  3.29114426e-01, -5.31095060e-02],
[-7.94278019e-01, -6.32404910e-01, -5.20764978e-01,
  3.15628291e-01,  4.51122635e-02],
[-4.11943076e-01, -4.49000189e-01,  9.65984733e-02,
  3.51134593e-01,  6.47694503e-02],
[-7.76795885e-01, -9.01086488e-02, -3.42617557e-01,
  4.71376321e-01, -1.56496550e-01],
[-6.37417963e-01, -4.95874422e-01, -7.36285751e-01,
  5.63576598e-01,  4.60175774e-02],
[-2.62665698e-01, -9.59395716e-01, -4.61834510e-01,
  5.22570283e-01, -6.54119179e-02],
[-3.17522879e-01, -6.43343840e-01,  2.62161592e-01,
  4.62234848e-01, -7.88005369e-03],
[-5.94815957e-01, -5.85754752e-01, -3.24803103e-01,
  1.81419320e-01, -1.18672266e-03],
[-8.91785936e-01, -1.65318946e-01, -5.15192976e-01,
  3.97050513e-01, -8.17748431e-02],
[-3.39107103e-01, -7.74240941e-01,  6.56794232e-01,
  2.90888562e-01, -2.05328295e-02],
[-2.60545847e-01, -5.62606722e-01,  5.76883163e-02,
  3.92005092e-01,  1.09193390e-01],
[-3.95717200e-01, -1.11598623e+00,  7.41107587e-01,
  4.63488290e-01, -2.49944425e-02],
[-6.28903683e-01, -8.25863565e-01, -7.96851835e-01,
  1.08631303e+00,  2.83219780e-02],
[-7.97409493e-01, -5.97980473e-01, -8.05120919e-01,
  8.05455180e-01, -3.72267263e-02],
[-6.06789590e-01, -4.75282630e-01, -6.70660699e-01,
  6.32014884e-01, -6.47520999e-02],
[-5.69112323e-02, -8.16689063e-01,  9.51595694e-01,
  1.53805033e-01,  1.84510350e-01],
[-6.12433628e-01, -7.26400605e-01,  1.14834430e-01,
  2.69455254e-01, -6.24639348e-02],
[-5.42078790e-01, -6.50769779e-01,  1.41260418e-01,
  2.13489340e-01, -2.07777487e-01],
[-8.04789156e-01, -4.43352238e-01, -1.05054673e+00,
  6.27974259e-01,  1.78052263e-01],
[-8.16640534e-01,  1.63595905e-01, -2.18107961e+00,
```

```
         6.88446269e-01,  3.14944582e-03],
       [-1.04630113e+00, -1.22175795e-02, -5.51772674e-01,
         1.42960164e-01, -5.13894028e-02],
       [-7.15330988e-01, -1.28271484e+00,  3.73899249e-01,
         7.52176735e-01, -1.86186589e-02],
       [-6.04023409e-01, -4.96577823e-01,  1.06959524e+00,
         2.15214362e-02, -4.93919866e-02],
       [-2.11942195e+00,  5.62745759e-01,  3.83567605e-01,
         2.51874225e-01,  5.89864685e-02],
       [-1.80358699e+00,  2.28420683e-01,  1.84116335e+00,
        -2.29774909e-01,  4.42586542e-02],
       [-1.94360726e+00,  2.68102662e-01,  4.71721148e-01,
         1.48065919e-01, -3.62129404e-02],
       [-1.98952906e+00,  9.55292503e-01, -8.62429472e-01,
         1.97803130e-01,  1.15837476e-01],
       [-1.77229024e+00,  2.38748418e-02,  1.80888041e+00,
        -1.95814118e-02,  1.81595705e-02],
       [-1.80435529e+00, -7.47393188e-02,  2.19590519e+00,
        -1.68781960e-01,  8.97012772e-02],
       [-2.45279530e+00,  4.35919679e-01,  1.39830174e+00,
        -2.91191711e-01, -1.66038797e-02],
       [-2.06907581e+00,  8.13022021e-01,  9.43871159e-01,
        -1.54848948e-01,  7.35986809e-02],
       [-3.69134907e+00,  2.37230846e+00, -1.93304321e+00,
        -7.54712958e-01,  9.58374460e-02],
       [-2.37942683e+00,  6.15791646e-01, -3.16226256e-01,
         7.12462797e-02,  3.17306821e-02],
       [-1.48061979e+00, -1.27626184e-01,  1.66744093e+00,
        -2.07816300e-02,  2.42864930e-01],
       [-1.85592607e+00,  4.07988792e-01,  1.29129962e+00,
         5.71810630e-05,  7.99726558e-02],
       [-1.58565463e+00,  1.14746214e-01, -5.14992615e-01,
         6.94551907e-01, -1.26336759e-01],
       [-1.48588807e+00,  7.59038785e-02,  2.04232427e+00,
         1.93588947e-01,  1.18246742e-01],
       [-3.10415805e+00,  1.63720234e+00, -8.14253099e-01,
        -6.07400720e-01, -2.07593581e-02],
       [-1.96598372e+00,  8.03623700e-01, -1.19948680e-01,
         1.48206454e-01,  4.28832980e-02],
       [-8.17929961e-01, -7.50248517e-01,  2.34294786e+00,
         6.38162112e-01, -1.57101424e-02],
       [-1.96113929e+00,  3.53650769e-01,  1.14007428e+00,
         2.80192009e-03, -1.19259033e-02],
       [-2.86491175e+00,  1.57763223e+00, -4.48654546e-01,
        -6.19259380e-01, -2.79286661e-02],
       [-2.85808143e+00,  1.48265870e+00, -7.97097560e-01,
        -3.18712777e-01,  1.80453934e-01],
       [-3.05623034e+00,  1.89200282e+00, -5.33439874e-01,
        -6.08111063e-01, -8.97973386e-02],
       [-1.72354178e+00, -5.09471572e-02,  1.27935656e+00,
         7.30096967e-02, -1.10363928e-02],
       [-2.62654414e+00,  1.65377514e+00, -7.13835734e-01,
        -5.88841270e-02, -8.58424769e-02],
       [-2.60587336e+00,  1.29494108e+00,  2.95437904e-01,
        -3.68995612e-01, -6.61487251e-02],
       [-1.67916044e+00,  1.11345279e+00,  2.08448857e+00,
        -5.46738494e-01, -3.27559038e-01],
       [-1.71478073e+00,  6.10222204e-01,  4.95873702e-01,
         2.56891695e-01,  1.28580842e-02],
       [-2.05721064e+00,  1.27848637e+00, -1.10496895e-01,
         3.38925158e-01, -7.92841687e-02],
       [-2.53146138e+00,  1.61511473e+00,  2.04566667e-01,
        -3.18523081e-01, -2.16935618e-02],
       [-1.64157286e+00,  9.22727202e-01,  7.23252271e-01,
         2.20579636e-01, -2.37824641e-01],
       [-2.06706428e+00,  7.56013080e-01, -4.00082996e-02,
         2.82030946e-01,  9.87990378e-02]])
```

Analyse de l'ACP

In [90]:
```
acp.explained_variance_ratio_ #la 1er variable a la variance plus grand en %
```

Out[90]:
```
array([0.64396912, 0.17552955, 0.14381114, 0.03293477, 0.00375542])
```

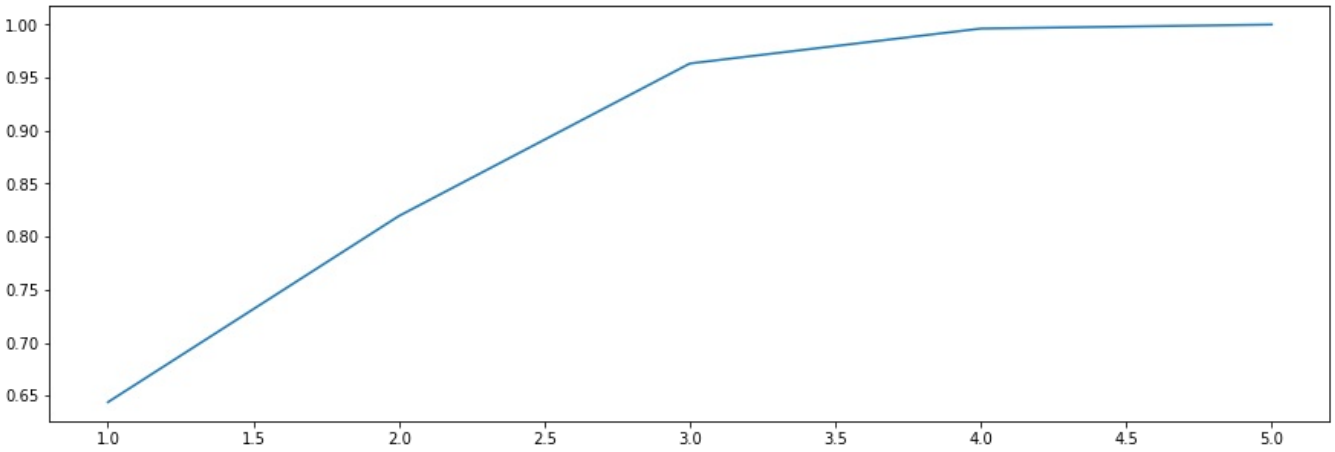
In [91]:
```
np.cumsum(acp.explained_variance_ratio_) #avec les 2 premieres variables on aura le 82% de l'information
```

Out[91]:
```
array([0.64396912, 0.81949867, 0.96330981, 0.99624458, 1.        ])
```

In [92]:
```
p = base_num_cr.shape[1]
n = base_num_cr.shape[0]
```

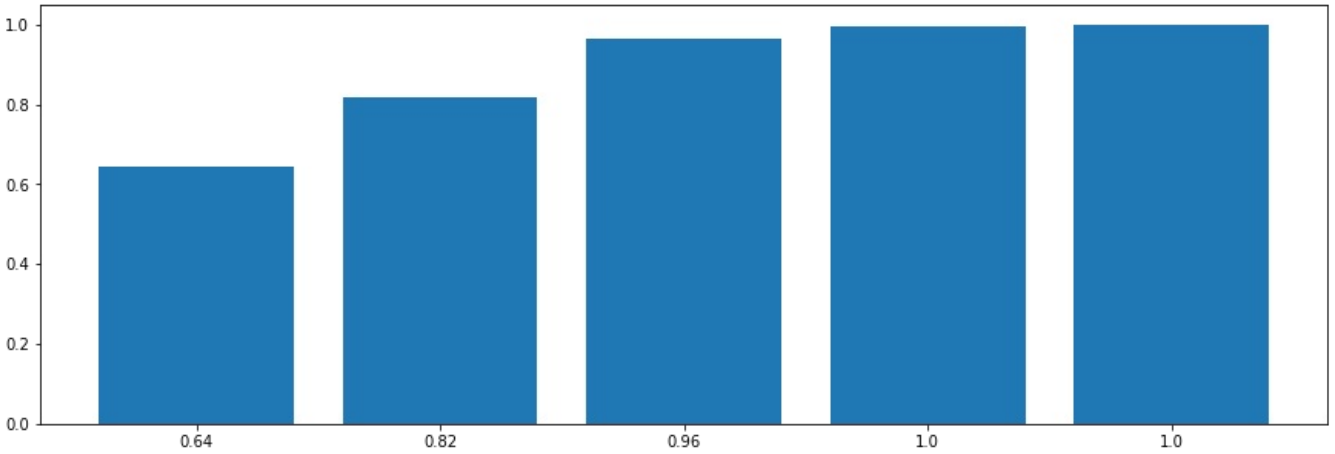
In [93]:
```
plt.plot(np.arange(1,p+1), np.cumsum(acp.explained_variance_ratio_))
```

Out[93]:
```
[<matplotlib.lines.Line2D at 0x247f29229d0>]
```

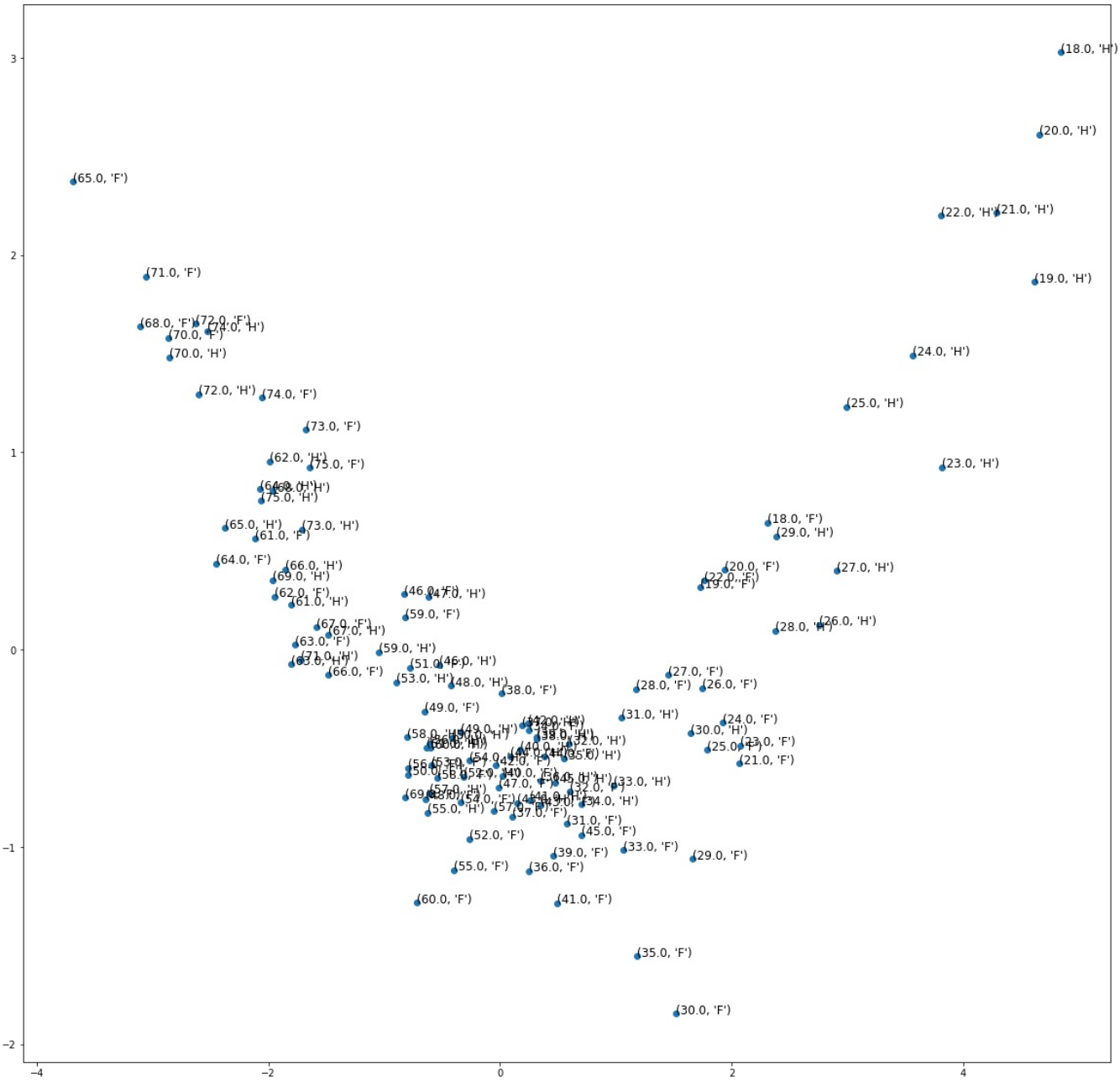


In [94]:
```
plt.bar(np.arange(len(acp.explained_variance_ratio_))+0.5,
        np.cumsum(acp.explained_variance_ratio_),
        tick_label=np.round(np.cumsum(acp.explained_variance_ratio_),2))
```

Out[94]:
```
<BarContainer object of 5 artists>
```



In [95]:
```
fig = plt.figure(figsize=(20,20))
plt.scatter(res_acp[:,0], res_acp[:,1])
for i in range(len(res_acp[:,0])): # etiquettes des points
    plt.text(res_acp[i,0],res_acp[i,1], agreg_base.index[i], fontsize=12)
fig.savefig('ACP.pdf')
```



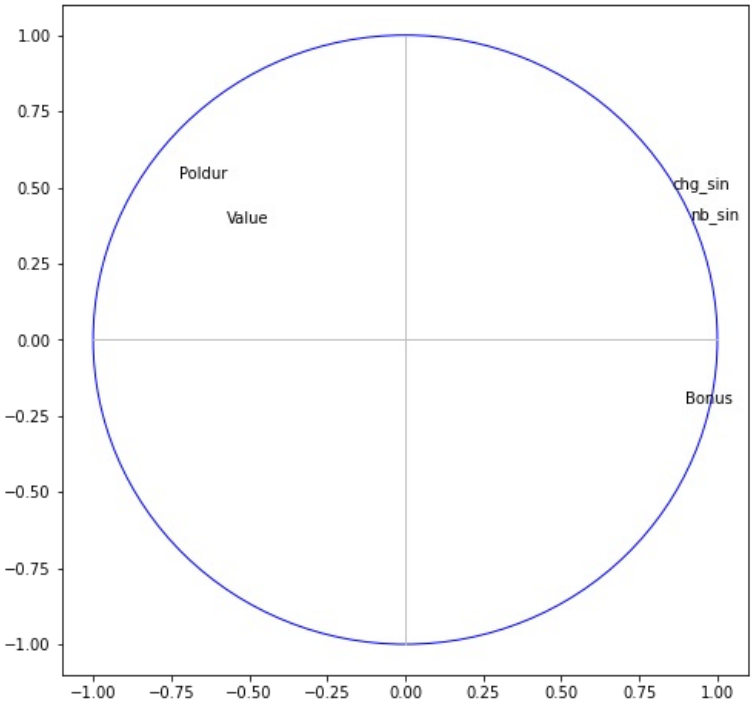
À l'aide du graphique précédent, nous pouvons tirer des conclusions quant aux similitudes dans les profils de risque. Par exemple, on remarque la présence des hommes les plus jeunes à droite du graphique. Au centre, nous constatons que les femmes les plus jeunes présentent un profil de risque similaire à celui des hommes âgés de 30 ans. À gauche se trouvent les personnes les plus âgées. Pour obtenir une idée du niveau de risque, nous devons analyser la matrice et le cercle des corrélations. Encore une fois, nous parlons ici de similitudes entre les individus.

In [96]:
```
var_cor = (n-1)/n*acp.explained_variance_  # Variance corrigée
corvar = np.zeros((p,p))
var_cor_rac = np.sqrt(var_cor)
for k in range(p):
 corvar[:,k] = acp.components_[k,:] * var_cor_rac[k] #corr = vect pr. x val pr.
print(corvar)
```

```
[[ 8.98761078e-01 -2.08579646e-01 -2.13957037e-01 -3.20496024e-01
   1.50910974e-02]
 [-7.23562316e-01  5.27610963e-01  3.82059212e-01 -2.28252534e-01
  -3.97310263e-03]
 [-5.73500463e-01  3.84122525e-01 -7.23341569e-01 -1.79370912e-02
   1.52978978e-03]
 [ 8.54343175e-01  4.99924418e-01  6.39158026e-02  8.90221596e-02
   9.03501084e-02]
 [ 9.10893329e-01  3.97862295e-01 -7.71604309e-04  4.01276567e-02
  -1.01823933e-01]]
```

In [97]:
```
# Cercle des corrélations
fig, axes = plt.subplots(figsize=(8,8))
#affichage des étiquettes (noms des variables)
for j in range(p):
 plt.annotate(agreg_base.columns[j],(corvar[j,0],corvar[j,1]))
#ajouter les axes
plt.plot([-1,1],[0,0],color='silver',linestyle='-',linewidth=1)
plt.plot([0,0],[-1,1],color='silver',linestyle='-',linewidth=1)
#ajouter un cercle
```

```
cercle = plt.Circle((0,0),1,color='blue',fill=False)
axes.add_artist(cercle)
#affichage
plt.show()
```



Les variables chg_sin, nb_sin et Bonus sont des indicateurs du niveau de sinistralité. Ainsi, plus ces variables augmentent, plus elles sont positionnées à droite, ce qui entraîne un profil de risque plus élevé.

En tenant compte de la répartition des individus dans le graphique précédent et en se basant sur le cercle des corrélations, nous pouvons conclure que les hommes les plus jeunes présentent un profil de risque plus élevé, car ils sont positionnés à droite.

De plus, on observe que plus une personne est âgée, plus elle se rapproche de la gauche du graphique, où se trouve la variable "Durée du contrat".

In [98]:
```
df = pd.DataFrame({'comp1' : res_acp[:, 0],'comp2' : res_acp[:, 1],
                   'Bonus': agreg_base.Bonus });df
```

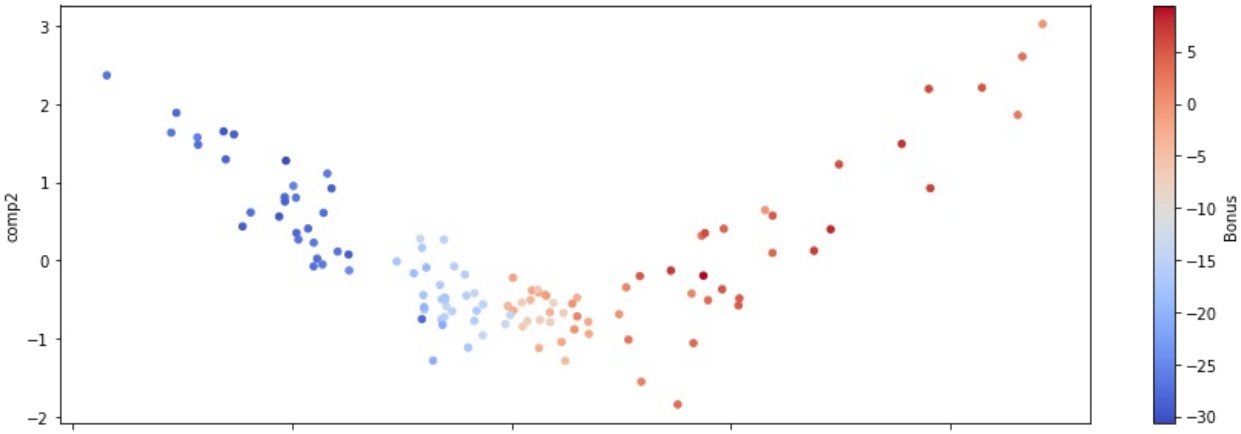
Out[98]:

		comp1	comp2	Bonus
Age	**Gender**			
18.0	**F**	2.313690	0.644816	-0.043478
	H	4.844385	3.029061	0.150754
19.0	**F**	1.733104	0.317015	2.810734
	H	4.617534	1.864498	2.373225
20.0	**F**	1.936815	0.405949	4.293059
...	**...**	...	...	...
73.0	**H**	-1.714781	0.610222	-27.632653
74.0	**F**	-2.057211	1.278486	-30.666667
	H	-2.531461	1.615115	-28.993289
75.0	**F**	-1.641573	0.922727	-28.750000
	H	-2.067064	0.756013	-28.685832

116 rows × 3 columns

In [99]:
```
df.plot.scatter(x='comp1',y='comp2', c='Bonus',cmap='coolwarm')
```

Out[99]:
```
<AxesSubplot:xlabel='comp1', ylabel='comp2'>
```



Ce nuage de points nous permet de visualiser la répartition des individus ainsi que le niveau de Bonus.

Analyse Factorielle des Correspondances (AFC)

Cette analyse est particulièrement utile lorsque on a un tableau de données avec plusieurs variables catégorielles (Droit,Sciences,Medecine et IUT) et que vous souhaitez **identifier des associations ou des tendances entre ces variables**.

L'AFC consiste ensuite à transformer un tableau de contingence en une représentation graphique ou en une série de vecteurs, en utilisant des méthodes de calcul des valeurs propres et des vecteurs propres. Ces vecteurs sont appelés "axes factoriels" et nous aident à visualiser les associations et les tendances dans les données.

Data set "CSP_Etudes"

"CSP_Etudes recopile les statistiques qui représentent la répartition des étudiants dans l'enseignement supérieur en France selon la catégorie socioprofessionnelle (CSP) des parents et la filière d'études. On observe ainsi que plus d'un tiers des étudiants français avaient des parents cadres ou occupant des professions intellectuelles supérieures. Cette proportion est bien plus élevée que dans toutes les autres CSP".

Source: fr.statista.com

In [100…
```
base_illu = pd.read_csv("CSP_Etudes.csv", delimiter = ";",index_col=0) ; base_illu.head()
```

Out[100]:

CSP.Filiere	Droit	Sciences	Medecine	IUT
Exp.agri	80	99	65	58
Patron	168	137	208	62
Cadre.sup	470	400	876	79
Employe	145	133	135	54
Ouvrier	166	193	127	129

Le tableau, qui résume les fréquences d'apparition conjointes des catégories pour deux variables données. Cela donne une vue d'ensemble des associations entre les variables.

In [102…
```
profil_lig = base_illu.copy()
```

In [ ]:
```
base_illu.shape[0] #base_illu.shape[0] pour voir le numb des lignes.
#base_illu.shape[1] pour voir nb colonnes.
#base_illu.shape pour voir colonnes et lignes
#len(base_illu) pour le nb colonnes
```

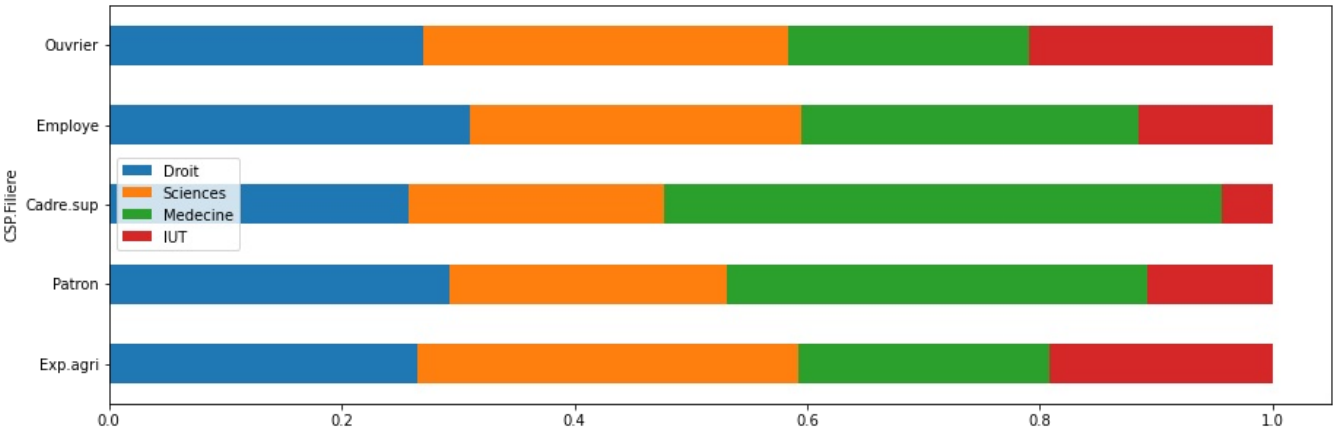
In [103…
```
for i in range(profil_lig.shape[0]):
    profil_lig.iloc[i] = profil_lig.iloc[i] / sum(profil_lig.iloc[i]) #chaque valeur d'une ligne divisée par la
profil_lig
```

Out[103]:

CSP.Filiere	Droit	Sciences	Medecine	IUT
Exp.agri	0.264901	0.327815	0.215232	0.192053
Patron	0.292174	0.238261	0.361739	0.107826
Cadre.sup	0.257534	0.219178	0.480000	0.043288
Employe	0.310493	0.284797	0.289079	0.115632
Ouvrier	0.269919	0.313821	0.206504	0.209756

In [104…
```
profil_lig.plot.barh(stacked = True)
```

Out[104]:
```
<AxesSubplot:ylabel='CSP.Filiere'>
```



26,9 % des enfants d'ouvriers poursuivent leurs études en droit, 31,3 % en sciences, etc.

L'idée, une fois de plus, est de mettre en évidence les similitudes des profils. Nous centrons donc notre attention sur les ouvriers et les exploitants agricoles, qui présentent une répartition très similaire.

In [105…
```
profil_col = base_illu.transpose().copy()
```

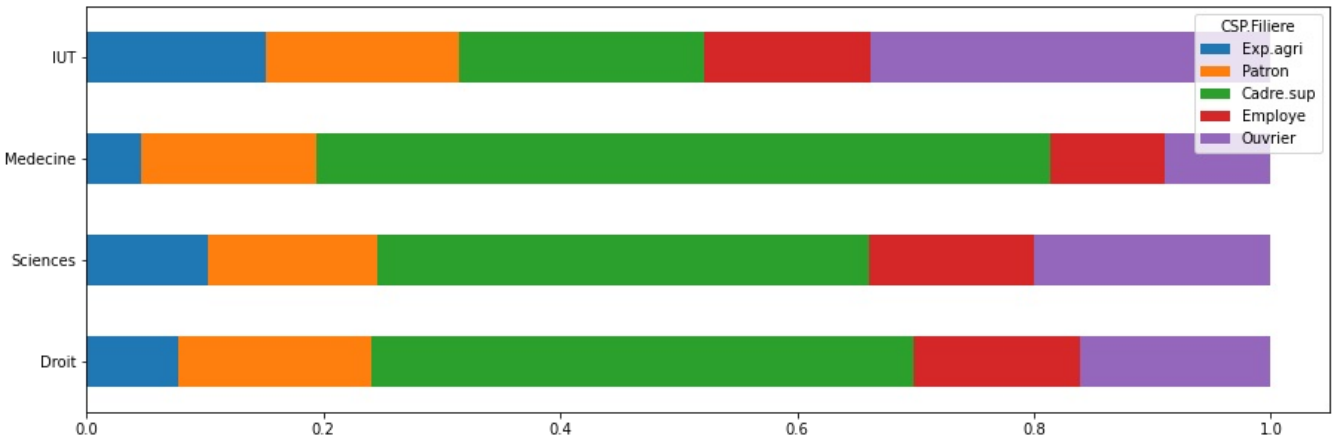
In [106…
```
for i in range(profil_col.shape[0]):
    profil_col.iloc[i] = profil_col.iloc[i] / sum(profil_col.iloc[i])
profil_col
```

Out[106]:

CSP.Filiere	Exp.agri	Patron	Cadre.sup	Employe	Ouvrier
Droit	0.077745	0.163265	0.456754	0.140914	0.161322
Sciences	0.102911	0.142412	0.415800	0.138254	0.200624
Medecine	0.046067	0.147413	0.620836	0.095677	0.090007
IUT	0.151832	0.162304	0.206806	0.141361	0.337696

In [107…
```
profil_col.plot.barh(stacked = True)
```

Out[107]:
```
<AxesSubplot:>
```

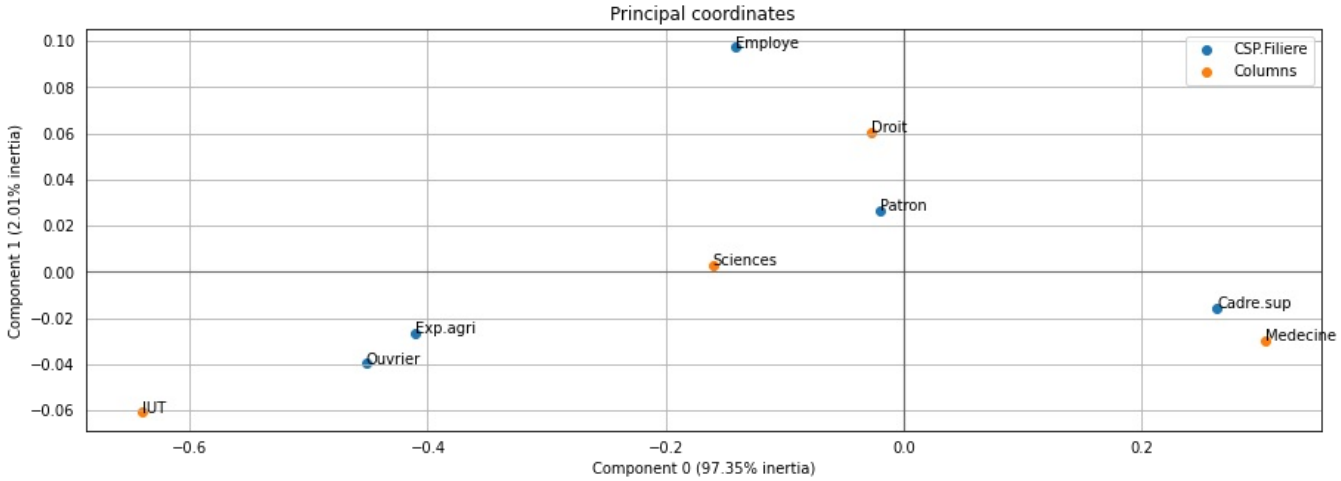


In [101…
```
ca = CA()
ca.fit(base_illu)
ca.explained_inertia_
```

Out[101]:
```
[0.9734955224825108, 0.02012655838500913]
```

In [108…
```
ax = ca.plot_coordinates(X=base_illu, show_row_labels=True, show_col_labels=True, figsize=(15,5))
```

```
ax.get_figure().savefig('AFC_illu.pdf')
```



Les ouvriers et les exploitants agricoles ont un profil très similaire. Il est donc possible que leurs enfants choisissent les mêmes études. Dans ce cas, ils se dirigent souvent vers une formation en IUT.

En revanche, les cadres supérieurs sont opposés aux ouvriers et sont plutôt enclins à suivre des études en médecine.

Data set Base_freq

Ce jeu de données contient, entre autres, le nombre de sinistres, l'âge du conducteur et la classe du véhicule (Small, Medium et Large).

Nous allons appliquer l'AFC sur le tableau croisé (de contingence) de base_freq : Age x Catégorie et compter le nombre de sinistres pour établir des profils de risque.

In [109…
```
df = pd.crosstab(base["Age"],base["Category"], values = base["nb_sin"],
            aggfunc = pd.Series.count);df.head()
```

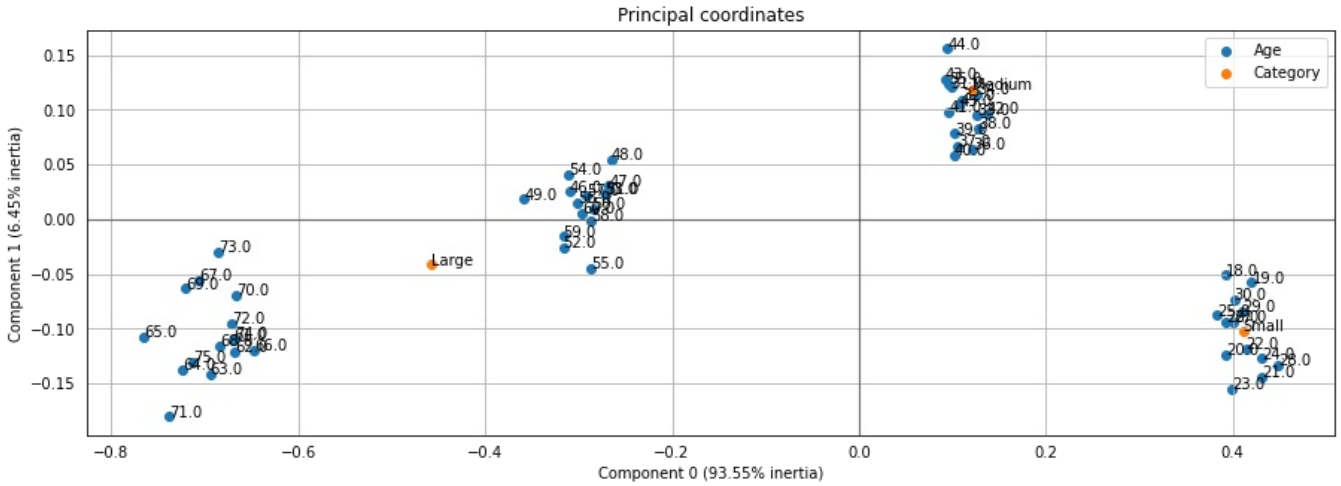
Out[109]:

Category	Large	Medium	Small
Age			
18.0	307	660	718
19.0	289	664	741
20.0	359	664	833
21.0	343	675	894
22.0	357	710	888

In [110…
```
ca = CA()
ca.fit(df)
ca.explained_inertia_
```

Out[110]:
```
[0.9355301890382188, 0.06446981096178098]
```

In [111…
```
ax = ca.plot_coordinates(X=df, show_row_labels=True, show_col_labels=True, figsize=(15,5))
ax.get_figure().savefig('AFC.pdf')
```



Les clusters permettent d'identifier que les plus jeunes sont propriétaires de véhicules de la catégorie "Small". Les personnes âgées de 30 à 45 ans possèdent des véhicules "Medium".

De leur côté, la catégorie "Large" se trouve au milieu des plus âgés et des personnes de plus de 45 ans.

Analyse des correspondances multiples ACM

L'ACM a pour objectif principal d'analyser la structure des données catégorielles en mettant l'accent sur la **relation entre les lignes et les colonnes dans un tableau de contingence**.

¡Attention! La difference entre L'ACM et L'AFC est que la L'AFC cherche à expliquer la **variance et la covariance des variables catégorielles**. Elle vise à réduire la dimensionnalité des données en dégageant des facteurs (analogues aux composantes principales dans l'AFC) qui expliquent la variation dans les données catégorielles.**Elle ne se concentre pas spécifiquement sur les relations entre les catégories de variables**

Application sur base_freq

In [112…
```
df = base.copy().iloc[0:20,]
```

In [113…
```
df.dtypes
```

Out[113]:
```
PolNum          int64
CalYear         int64
Gender         object
Type           object
Category       object
Occupation     object
Age           float64
Group1          int64
Bonus           int64
Poldur        float64
Value         float64
Adind           int64
Group2         object
Density       float64
Value_num     float64
Expdays         int64
nb_sin        float64
chg_sin       float64
dtype: object
```

In [114…
```
df = df[["Gender","Category","Occupation","Type"]] ; df.head()
# df composé uniquement de colonnes catégoriques
```

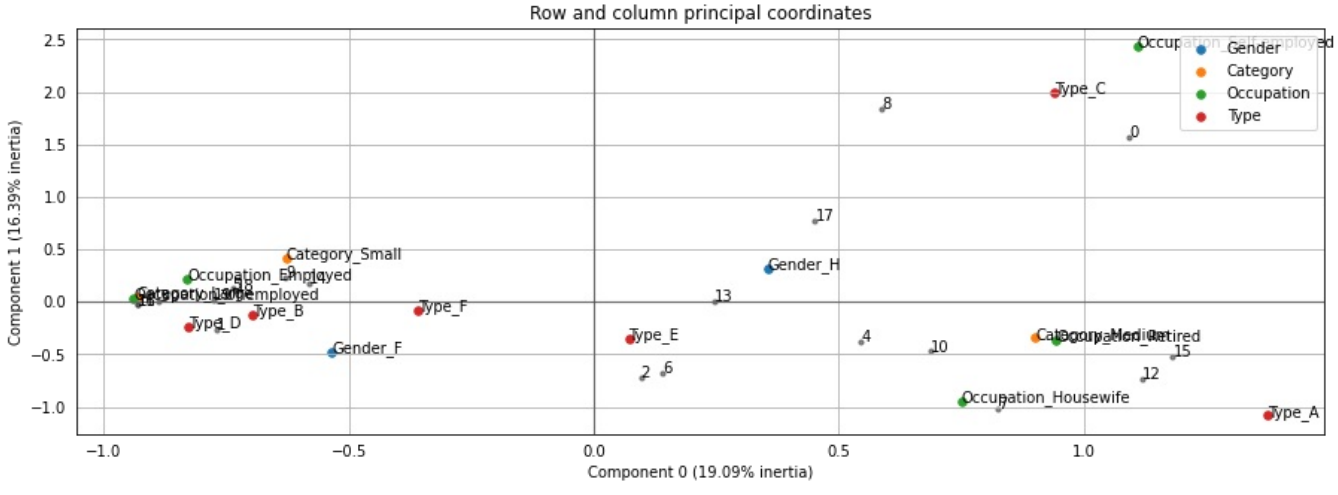
Out[114]:

	Gender	Category	Occupation	Type
0	H	Medium	Self-employed	C
1	F	Large	Unemployed	E
2	F	Medium	Housewife	D
3	F	Small	Employed	B
4	H	Medium	Housewife	F

In [115…
```
mca = MCA()
mca = mca.fit(df)
mca.explained_inertia_
```

Out[115]:
```
[0.19086690165004072, 0.16388508388393352]
```

In [116…
```
ax = mca.plot_coordinates(X=df,ax=None,figsize=(15, 5),show_row_points=True, row_points_size=10,
                          show_row_labels=True,show_column_points=True,
                          column_points_size=30,show_column_labels=True,legend_n_cols=1)
```



À gauche du graphique, on peut clairement observer un regroupement d'individus de genre féminin occupant des postes d'employées. Ils semblent particulièrement enclins à choisir des véhicules des catégories Small, tels que B, D ou F.

En revanche, les hommes ne semblent pas présenter de regroupement évident par rapport à ces critères.

réaliser l'ACM sur un tableau de contingence : granularité : Sexe x Age sur les variables de sinistralité : nb_sin, chg_sin, bonus

In [117…
```
df = base.copy()
df.replace({"Male":"H", "Female":"F"}, inplace=True)
granularite = ['Gender','Age']
agreg_base = pd.pivot_table(df,
                            index = granularite,
                            values = ['chg_sin', 'nb_sin','Bonus'],
                            aggfunc = np.mean)
agreg_base.head()
#on prend les variables de sinistralité pour voir le profile de risque
#le resultat montre des variables quanti, alors on peut discretiser avec .cut
```

Out[117]:

Gender	Age	Bonus	chg_sin	nb_sin
F	18.0	-0.043478	223.673246	0.253623
	19.0	2.810734	187.163376	0.194915
	20.0	4.293059	193.610861	0.215938
	21.0	4.897959	162.002003	0.182398
	22.0	6.037500	183.579375	0.197500

In [118…
```
# Pour améliorer la clarté de l'information, nous avons segmenté (discretiser)les variables en cinq parts égale
seg_bonus = 5
agreg_base["BM"] = pd.cut(agreg_base.Bonus, seg_bonus, labels=["B1","B2","B3","B4","B5"])
seg_nbsin = 5
agreg_base["FQ"] = pd.cut(agreg_base.nb_sin, seg_nbsin, labels=["S1","S2","S3","S4","S5"])
seg_chsin = 5
agreg_base["CS"] = pd.cut(agreg_base.chg_sin, seg_chsin, labels=["C1","C2","C3","C4","C5"])
```

In [119…
```
print(agreg_base.BM.value_counts())
print(agreg_base.FQ.value_counts())
print(agreg_base.CS.value_counts())
```

```
B1    30
B2    25
B4    25
B5    25
B3    11
Name: BM, dtype: int64
S1    71
S2    31
S3     5
S5     5
S4     4
Name: FQ, dtype: int64
C1    92
C2    14
C3     5
C4     3
C5     2
Name: CS, dtype: int64
```

In [120…
```
agreg_base = agreg_base[["BM","FQ","CS"]];agreg_base
```

Out[120]:

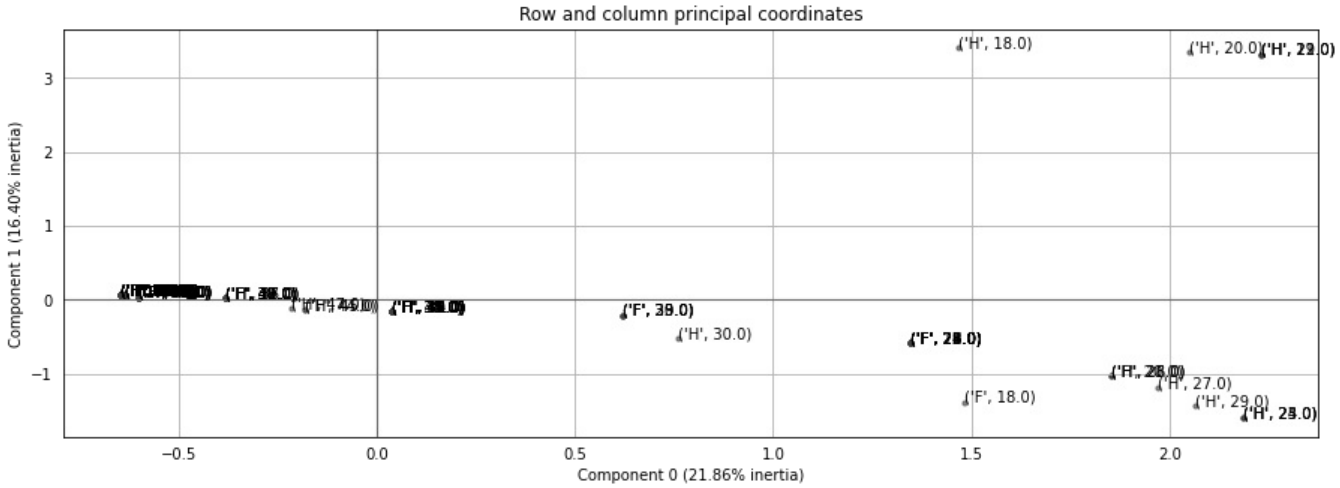
		BM	FQ	CS
Gender	**Age**			
F	**18.0**	B4	S3	C3
	19.0	B5	S2	C2
	20.0	B5	S3	C2
	21.0	B5	S2	C2
	22.0	B5	S2	C2
...	**...**	...	...	...
H	**71.0**	B1	S1	C1
	72.0	B1	S1	C1
	73.0	B1	S1	C1
	74.0	B1	S1	C1
	75.0	B1	S1	C1

116 rows × 3 columns

In [121…
```
mca = MCA()
mca = mca.fit(agreg_base)
mca.explained_inertia_
```

Out[121]:
```
[0.21860807432091753, 0.164023568861189]
```

In [122…
```
ax = mca.plot_coordinates(X=agreg_base,ax=None,figsize=(15, 5),show_row_points=True, row_points_size=10,
                          show_row_labels=True,show_column_points=False,
                          column_points_size=30,show_column_labels=False,legend_n_cols=1)
```

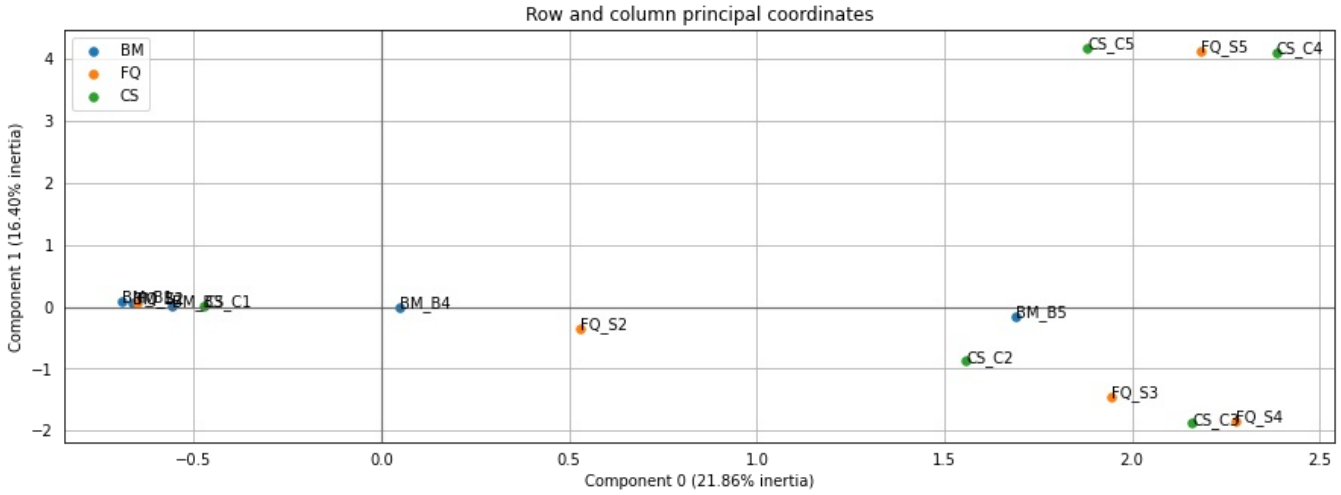


In [123…
```
coord = mca.row_coordinates(agreg_base)
coord[coord.iloc[:,1]>3]
```

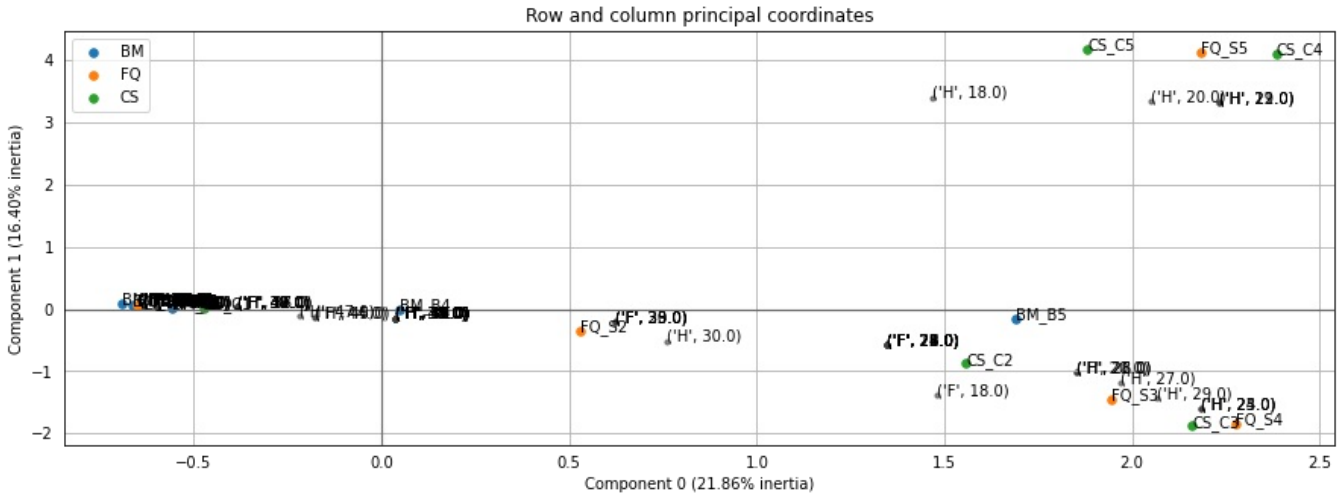
Out[123]:

	0	1
(H, 18.0)	1.466673	3.404393
(H, 19.0)	2.230556	3.317825
(H, 20.0)	2.050674	3.347304
(H, 21.0)	2.230556	3.317825
(H, 22.0)	2.230556	3.317825

In [124…
```
ax = mca.plot_coordinates(X=agreg_base,ax=None,figsize=(15, 5),show_row_points=False, row_points_size=10,
                          show_row_labels=False,show_column_points=True,
                          column_points_size=30,show_column_labels=True,legend_n_cols=1)
```



In [125…
```
ax = mca.plot_coordinates(X=agreg_base,ax=None,figsize=(15, 5),show_row_points=True, row_points_size=10,
                          show_row_labels=True,show_column_points=True,
                          column_points_size=30,show_column_labels=True,legend_n_cols=1)
```



L'analyse se base sur l'identification de clusters d'individus qui representent leur ressemblance en fonction de leur profil de risque. Pour rappel,les variables prises en compte sont le Bonus, ch_sin et Nb_sin. Dans la partie supériuere du graphique, nous pouvons observer que les hommes plus jeunes presentent une proximité avec la segmentation CS_5 FQ_5, ce qui se traduit par un profil de risque plus élevé avec un nombre plus important de sinistres.

Analyse Factorielle de Données Mixte

L'AFDM analyse un ensemble de données qui comprend à la fois des variables quantitatives (continues) et des variables catégorielles (nominales ou ordinales).

Elle applique des techniques d'Analyse Factorielle à partir des données quantitatives et de l'ACM pour les données catégorielles pour réduire la dimensionnalité et explorer la structure de ces données.

In [126…
```
granularite = ['Age','Gender']
agreg_base = pd.pivot_table(base,
                            index = granularite,
                            values = ['chg_sin', 'nb_sin','Value','Poldur','Bonus'],
                            aggfunc = np.mean)
agreg_base
```

Out[126]:

Age	Gender	Bonus	Poldur	Value	chg_sin	nb_sin
18.0	F	-0.043478	5.230435	16378.318709	223.673246	0.253623
	H	0.150754	5.334673	16131.023412	477.919618	0.431156
19.0	F	2.810734	5.590395	16129.400087	187.163376	0.194915
	H	2.373225	5.305274	15658.153817	382.607677	0.412779
20.0	F	4.293059	5.191517	16687.004608	193.610861	0.215938
...	...	...	...	...	...	...
73.0	H	-27.632653	6.134694	16681.152290	60.804531	0.075510
74.0	F	-30.666667	6.225641	17123.531971	69.577231	0.092308
	H	-28.993289	6.635347	17148.486553	58.838076	0.073826
75.0	F	-28.750000	6.261905	16636.513361	64.602500	0.107143
	H	-28.685832	6.119097	16991.859611	55.117228	0.059548

116 rows × 5 columns

In [127…
```
seg_bonus = 5
agreg_base["Bonus"] = pd.cut(agreg_base.Bonus, seg_bonus, labels=["B1","B2","B3","B4","B5"])
seg_poldur = 5
agreg_base["Poldur"] = pd.cut(agreg_base.Poldur, seg_poldur, labels=["P1","P2","P3","P4","P5"])
agreg_base.Bonus = agreg_base.Bonus.astype(str)
agreg_base.Poldur = agreg_base.Poldur.astype(str)
agreg_base.head()
```

Out[127]:

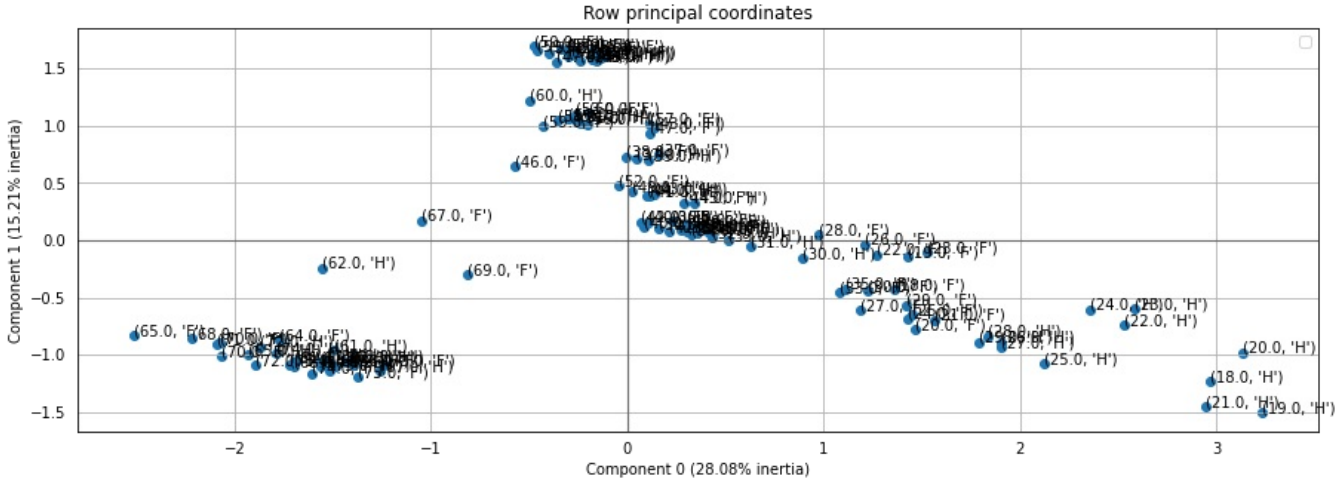
Age	Gender	Bonus	Poldur	Value	chg_sin	nb_sin
18.0	F	B4	P1	16378.318709	223.673246	0.253623
	H	B4	P1	16131.023412	477.919618	0.431156
19.0	F	B5	P2	16129.400087	187.163376	0.194915
	H	B5	P1	15658.153817	382.607677	0.412779
20.0	F	B5	P1	16687.004608	193.610861	0.215938

In [128…
```
afdm = FAMD()
afdm = afdm.fit(agreg_base)
afdm.explained_inertia_
```

Out[128]:
```
array([0.28079242, 0.15207438])
```

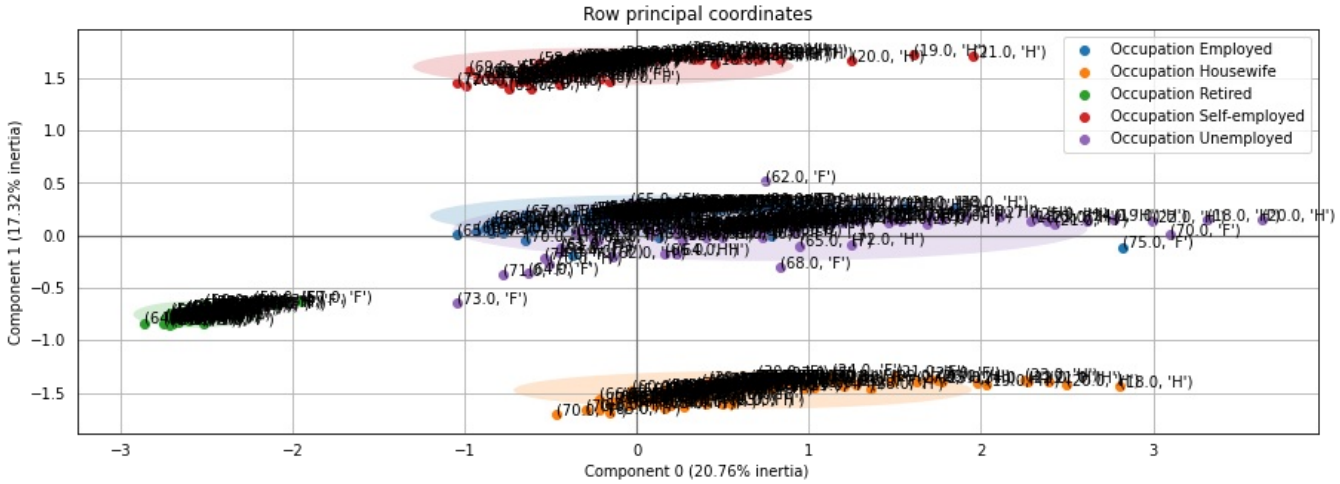
In [129…
```
ax = afdm.plot_row_coordinates(agreg_base, figsize=(15, 5), labels=agreg_base.index,
                               show_points=True)
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are igno
red when legend() is called with no argument.
```



In [130…
```
granularité = ['Age','Gender', 'Occupation']
agreg_base = pd.pivot_table(base,
                            values=['chg_sin','nb_sin','Value','Poldur','Bonus'],
                            index=granularité,
                            aggfunc=np.mean)
agreg_base = agreg_base.reset_index()
agreg_base = agreg_base.set_index(['Age','Gender'])
afdm = FAMD()
afdm = afdm.fit(agreg_base)
```

In [131…
```
ax = afdm.plot_row_coordinates(agreg_base, figsize=(15, 5), labels=agreg_base.index,
                               color_labels=['Occupation {}'.format(t) for t in agreg_base['Occupation']],
                               ellipse_outline=False,ellipse_fill=True,show_points=True)
```



In [132…
```
afdm.column_correlations(agreg_base)
```

Out[132]:

	0	1
Bonus	0.260407	0.210747
Occupation_Employed	0.041340	0.042572
Occupation_Housewife	0.019543	0.014103
Occupation_Retired	-0.114535	-0.084830
Occupation_Self-employed	0.006425	0.027609
Occupation_Unemployed	0.020613	-0.019272
Poldur	0.076491	0.069740
Value	-0.899522	-0.966077
chg_sin	0.515149	0.327028
nb_sin	0.516099	0.354120