Package fanalysis

Analyse des Correspondances Multiples

Ce tutoriel a pour objectif de présenter rapidement les principales fonctionnalités offertes par le package *fanalysis* pour réaliser une Analyse des Correspondances Multiples.
Il suppose connu les soubassements théoriques de cette méthode.
Il ne s'attarde pas non plus sur l'interprétation du jeu de données, qui n'a pour but que de présenter les fonctionnalités du package.
2 approches sont présentées :

- Une approche "datamining" : l'ACM vise à décrire un jeu de données
- Une approche "machine learning" : l'ACM est utilisée comme méthode de réduction des données, le résultat servant d'entrée pour un modèle prédictif (nous ferons ici l'usage d'outils de scikit-learn).

I. Approche Datamining

L'ACM a ici pour but de décrire un fichier de données.
Celui-ci est extrait du site de Ricco Rakotomalala (Université Lyon 2) :
http://eric.univ-lyon2.fr/%7Ericco/tanagra/fichiers/races_canines_acm.xls

Nous partons d'un fichier texte intitulé "mca_data.txt".
On importe la librairie pandas pour charger les données, ainsi que la classe MCA du package *fanalysis*.
Les données sont transformées en matrice de type numpy.ndarray.

Dans la matrice de données X, les catégories peuvent être codées :

- soit par des entiers
- soit par des chaines de caractères

Dans cet exemple, les catégories sont codées par des chaines de caractère.

In [1]:
```
import pandas as pd
from fanalysis.mca import MCA
%matplotlib inline
```

In [2]:
```
df = pd.read_table("mca_data.txt", header=0, index_col=0, delimiter="\t", encoding="utf-8")
```

In [3]:
```
print(df)
```

```
                 Taille    Poids Velocite Intelligence Affection Agressivite  \
Chien
Beauceron       Taille++   Poids+  Veloc++      Intell+    Affec+     Agress+
Basset           Taille-   Poids-   Veloc-      Intell-    Affec-     Agress+
Berger-All      Taille++   Poids+  Veloc++     Intell++    Affec+     Agress+
Boxer            Taille+   Poids+   Veloc+      Intell+    Affec+     Agress+
Bull-Dog         Taille-   Poids-   Veloc-      Intell+    Affec+     Agress-
Bull-Mastif     Taille++  Poids++   Veloc-     Intell++    Affec-     Agress+
Caniche          Taille-   Poids-   Veloc+     Intell++    Affec+     Agress-
Chihuahua        Taille-   Poids-   Veloc-      Intell-    Affec+     Agress-
Cocker           Taille+   Poids-   Veloc-      Intell+    Affec+     Agress+
Colley          Taille++   Poids+  Veloc++      Intell+    Affec+     Agress-
Dalmatien        Taille+   Poids+   Veloc+      Intell+    Affec+     Agress-
Doberman        Taille++   Poids+  Veloc++     Intell++    Affec-     Agress+
Dogue-All       Taille++  Poids++  Veloc++      Intell-    Affec-     Agress+
Epag.-Breton     Taille+   Poids+   Veloc+     Intell++    Affec+     Agress-
Epag.-Français  Taille++   Poids+   Veloc+      Intell+    Affec-     Agress-
Fox-Hound       Taille++   Poids+  Veloc++      Intell-    Affec-     Agress+
Fox-Terrier      Taille-   Poids-   Veloc+      Intell+    Affec+     Agress+
Gd-Bleu-Gasc    Taille++   Poids+   Veloc+      Intell-    Affec-     Agress+
Labrador         Taille+   Poids+   Veloc+      Intell+    Affec+     Agress-
Levrier         Taille++   Poids+  Veloc++      Intell-    Affec-     Agress-
Mastiff         Taille++  Poids++   Veloc-      Intell-    Affec-     Agress+
Pekinois         Taille-   Poids-   Veloc-      Intell-    Affec+     Agress-
Pointer         Taille++   Poids+  Veloc++     Intell++    Affec-     Agress-
St-Bernard      Taille++  Poids++   Veloc-      Intell+    Affec-     Agress+
Setter          Taille++   Poids+  Veloc++      Intell+    Affec-     Agress-
Teckel           Taille-   Poids-   Veloc-      Intell+    Affec+     Agress-
Terre-Neuve     Taille++  Poids++   Veloc-      Intell+    Affec-     Agress-

                 Fonction
Chien
Beauceron         utilite
Basset             chasse
Berger-All        utilite
Boxer           compagnie
Bull-Dog        compagnie
Bull-Mastif       utilite
Caniche         compagnie
Chihuahua       compagnie
Cocker          compagnie
Colley          compagnie
Dalmatien       compagnie
Doberman          utilite
Dogue-All         utilite
Epag.-Breton       chasse
Epag.-Français     chasse
Fox-Hound          chasse
Fox-Terrier     compagnie
Gd-Bleu-Gasc       chasse
Labrador           chasse
Levrier            chasse
Mastiff           utilite
Pekinois        compagnie
Pointer            chasse
St-Bernard        utilite
Setter             chasse
Teckel          compagnie
Terre-Neuve       utilite
```

L'analyse va porter sur les 6 premières variables.

```
In [4]: X = df.iloc[:, 0:6].as_matrix()
```

On crée une instance de la classe MCA, en lui passant ici des étiquettes pour les lignes et les variables. Ces paramètres sont facultatifs ; en leur absence, le programme détermine automatiquement des étiquettes.

```
In [5]: my_mca = MCA(row_labels=df.index.values, var_labels=df.columns.values[0:6])
```

On estime le modèle en appliquant la méthode *fit* de la classe MCA sur le jeu de données.

```
In [6]: my_mca.fit(X)
```

```
Out[6]: MCA(n_components=None,
          row_labels=array(['Beauceron', 'Basset', 'Berger-All', 'Boxer', 'Bull-Dog',
               'Bull-Mastif', 'Caniche', 'Chihuahua', 'Cocker', 'Colley',
               'Dalmatien', 'Doberman', 'Dogue-All', 'Epag.-Breton',
               'Epag.-Français', 'Fox-Hound', 'Fox-Terrier', 'Gd-Bleu-Gasc',
               'Labrador', 'Levrier', 'Mastiff', 'Pekinois', 'Pointer',
               'St-Bernard', 'Setter', 'Teckel', 'Terre-Neuve'], dtype=object),
          stats=True,
          var_labels=array(['Taille', 'Poids', 'Velocite', 'Intelligence', 'Affection',
               'Agressivite'], dtype=object))
```

L'exécution de la méthode *my_mca.fit(X)* provoque a minima le calcul des attributs :

- *my_pca.eig_* : valeurs propres
- *my_pca.row_coord_* : coordonnées des points lignes
- *my_pca.col_coord_* : coordonnées des points colonnes

I.1. Analyse des valeurs propres

L'attribut *my_mca.eig_* contient :

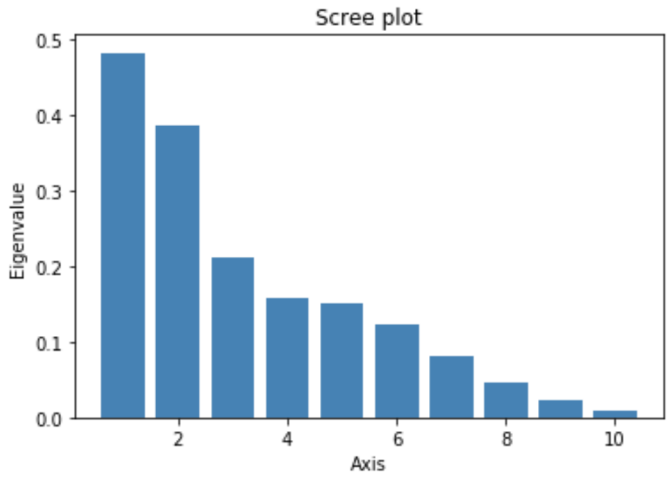
- en 1ère ligne : les valeurs propres en valeur absolue
- en 2ème ligne : les valeurs propres en pourcentage de la variance totale
- en 3ème ligne : les valeurs propres en pourcentage cumulé de la variance totale

```
In [7]: print(my_mca.eig_)
```

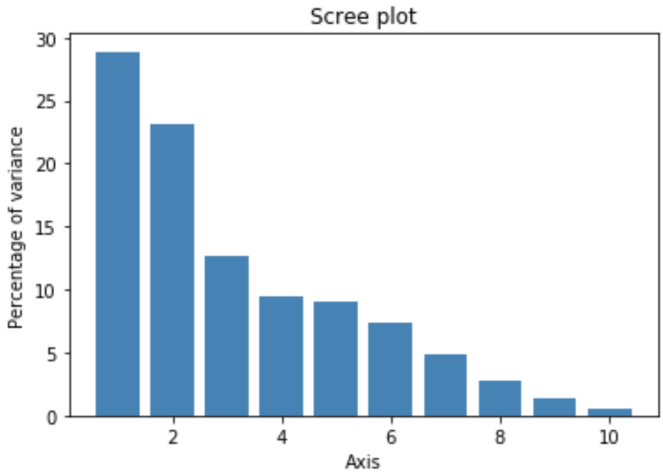
```
[[  4.81606165e-01   3.84737288e-01   2.10954049e-01   1.57554025e-01
    1.50132670e-01   1.23295308e-01   8.14624601e-02   4.56697566e-02
    2.35419107e-02   7.71303416e-03]
 [  2.88963699e+01   2.30842373e+01   1.26572430e+01   9.45324152e+00
    9.00796020e+00   7.39771849e+00   4.88774761e+00   2.74018539e+00
    1.41251464e+00   4.62782050e-01]
 [  2.88963699e+01   5.19806071e+01   6.46378501e+01   7.40910916e+01
    8.30990518e+01   9.04967703e+01   9.53845179e+01   9.81247033e+01
    9.95372180e+01   1.00000000e+02]]
```

Les valeurs propres peuvent être représentées graphiquement (par défaut : représentation en valeur absolue).

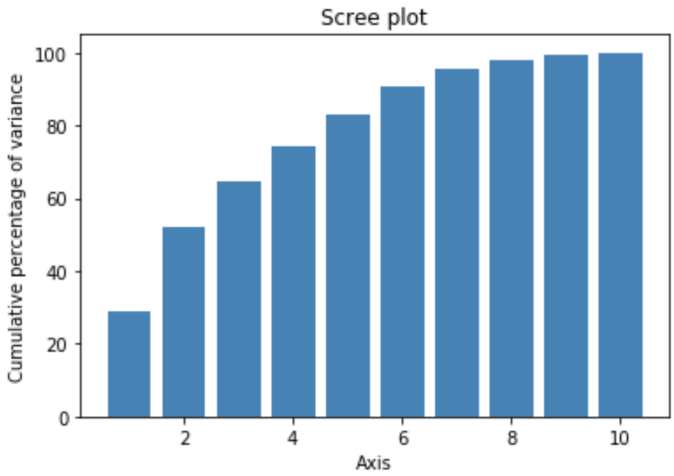
In [8]:
```
my_mca.plot_eigenvalues()
```



In [9]:
```
my_mca.plot_eigenvalues(type="percentage")
```



In [10]:
```
my_mca.plot_eigenvalues(type="cumulative")
```



Quand l'objet *my_mca* a été instancié, son paramètre *stats* a reçu la valeur True par défaut.
En conséquence, lors de l'exécution de la méthode *my_mca.fit(X)*, les statistiques suivantes ont été calculées :

- *my_mca.row_contrib_* : contributions des points lignes à la variance de l'axe
- *my_mca.col_contrib_* : contributions des points colonnes à la variance de l'axe
- *my_mca.row_cos2_* : cosinus carrés des points lignes
- *my_mca.col_cos2_* : cosinus carrés des points colonnes

Si l'on avait souhaité éviter le calcul de ces statistiques pour gagner du temps et économiser des ressources mémoire, il aurait fallu instancier :
my_mca = MCA(stats=False)

Par défaut, les coordonnées des points lignes et colonnes, leurs contributions et cosinus carrés sont calculés sur l'ensemble des axes extraits de l'analyse.
On aurait toutefois pu spécifier le nombre d'axes à retenir via le paramètre *n_components* avec par exemple :
my_mca = MCA(n_components=3)

I.2. Extraction des statistiques sur les points lignes

Export de la totalité des données lignes vers une DataFrame pandas

On peut simplement envoyer vers une Dataframe : les coordonnées, les contributions et les cos2 de chacun des points **lignes**, pour tous les axes factoriels (identifiés par les suffixes dim1, dim2, etc.).

In [11]:
```
df_rows = my_mca.row_topandas()
```

In [12]:
```
print(df_rows)
```

```
                row_coord_dim1  row_coord_dim2  row_coord_dim3  \
Beauceron             -0.317200       -0.417701       -0.101468
Basset                 0.254110        1.101227       -0.190701
Berger-All            -0.486396       -0.464450       -0.498134
Boxer                  0.447365       -0.881778        0.692016
Bull-Dog               1.013352        0.549879       -0.163423
Bull-Mastif           -0.752574        0.546912        0.497573
Caniche                0.912301       -0.016188       -0.576570
Chihuahua              0.840799        0.843852       -0.469947
Cocker                 0.733295        0.079073        0.662230
Colley                -0.117325       -0.526108       -0.334894
Dalmatien              0.647240       -0.990184        0.458590
Doberman              -0.873210       -0.315481       -0.452314
Dogue-All             -1.047017        0.506958        0.165035
Epag.-Breton           0.478044       -1.036933        0.061924
Epag.-Français        -0.144910       -0.515783        0.117127
Fox-Hound             -0.876568        0.025240       -0.362171
Fox-Terrier            0.881622        0.138967        0.053522
Gd-Bleu-Gasc          -0.517338       -0.113404        0.044029
Labrador               0.647240       -0.990184        0.458590
Levrier               -0.676693       -0.083167       -0.595598
Mastiff               -0.755932        0.887633        0.587715
Pekinois               0.840799        0.843852       -0.469947
Pointer               -0.673335       -0.423887       -0.685740
St-Bernard            -0.583379        0.593660        0.894239
Setter                -0.504140       -0.377139       -0.289074
Teckel                 1.013352        0.549879       -0.163423
Terre-Neuve           -0.383504        0.485254        0.660813

                row_coord_dim4  row_coord_dim5  row_coord_dim6  \
Beauceron             -0.211436       -0.118510       -0.844917
Basset                 0.292637       -0.524009        0.039895
Berger-All             0.577425        0.275902       -0.567765
Boxer                  0.260002       -0.455590       -0.213746
Bull-Dog              -0.349919        0.330786       -0.201414
Bull-Mastif            0.655153        0.721946        0.117926
Caniche                0.628133        0.434017        0.386066
Chihuahua             -0.086343       -0.177346        0.197088
Cocker                 0.189743       -0.104627       -0.626726
Colley                -0.657755        0.192130       -0.374168
Dalmatien             -0.186316       -0.144950        0.257003
Doberman               0.510087        0.239880       -0.254209
Dogue-All              0.062888       -0.316522       -0.001702
Epag.-Breton           0.602545        0.249461        0.534156
Epag.-Français        -0.468922        0.000850        0.490693
Fox-Hound             -0.015198       -0.662665       -0.132859
Fox-Terrier            0.285590       -0.271035       -0.361835
Gd-Bleu-Gasc           0.240972       -0.817923        0.418446
Labrador              -0.186316       -0.144950        0.257003
Levrier               -0.461516       -0.352025        0.337890
Mastiff                0.129868       -0.180598        0.239276
Pekinois              -0.086343       -0.177346        0.197088
Pointer                0.063769        0.550519        0.216540
St-Bernard            -0.133709        0.327535       -0.159227
Setter                -0.725093        0.156108       -0.060612
Teckel                -0.349919        0.330786       -0.201414
Terre-Neuve           -0.580027        0.638174        0.311523

                row_coord_dim7  row_coord_dim8  row_coord_dim9  \
Beauceron              0.089050        0.201986       -0.167019
Basset                 0.052833       -0.447363        0.100738
Berger-All            -0.129097        0.187330       -0.234185
Boxer                 -0.003008       -0.019819       -0.002446
Bull-Dog               0.063544       -0.079036       -0.035602
```

Statistiques pour les points lignes

In [13]:
```
# Coordonnées des points lignes
print(my_mca.row_coord_)
```

```
[[ -3.17200051e-01  -4.17701298e-01  -1.01467714e-01  -2.11436281e-01
   -1.18509544e-01  -8.44917274e-01   8.90501548e-02   2.01986411e-01
   -1.67018837e-01  -2.28067149e-02]
 [  2.54109843e-01   1.10122699e+00  -1.90700968e-01   2.92637269e-01
   -5.24008519e-01   3.98946811e-02   5.28331582e-02  -4.47362922e-01
    1.00738366e-01  -1.47102197e-01]
 [ -4.86395534e-01  -4.64449578e-01  -4.98133882e-01   5.77425252e-01
    2.75902052e-01  -5.67764838e-01  -1.29096839e-01   1.87330321e-01
   -2.34185002e-01   8.91993408e-03]
 [  4.47364924e-01  -8.81777941e-01   6.92015804e-01   2.60001835e-01
   -4.55589836e-01  -2.13745841e-01  -3.00768021e-03  -1.98192681e-02
   -2.44628915e-03  -1.40900948e-01]
 [  1.01335218e+00   5.49879489e-01  -1.63423202e-01  -3.49919273e-01
    3.30786485e-01  -2.01414177e-01   6.35440823e-02  -7.90356807e-02
   -3.56024365e-02  -6.65432469e-02]
 [ -7.52574495e-01   5.46911834e-01   4.97573073e-01   6.55152661e-01
    7.21946355e-01   1.17925813e-01  -1.85943335e-02  -3.73250962e-02
   -1.12102625e-01  -2.45732947e-02]
 [  9.12301484e-01  -1.61876744e-02  -5.76569722e-01   6.28133396e-01
    4.34016531e-01   3.86066396e-01   4.01809663e-01   2.49346973e-01
    1.65086085e-01   1.13481695e-01]
 [  8.40799374e-01   8.43852157e-01  -4.69947142e-01  -8.63428687e-02
   -1.77346455e-01   1.97088251e-01  -3.27291133e-01   1.31410453e-01
   -1.36707423e-01   2.48406642e-03]
 [  7.33295293e-01   7.90731735e-02   6.62230424e-01   1.89743186e-01
   -1.04627236e-01  -6.26726024e-01  -4.74951235e-01  -4.06017775e-01
    1.43663776e-01   3.04810338e-01]
 [ -1.17325200e-01  -5.26107651e-01  -3.34893734e-01  -6.57754538e-01
    1.92130196e-01  -3.74168009e-01  -8.41976661e-02   2.76173951e-01
   -1.10667059e-01   5.06561417e-02]
 [  6.47239775e-01  -9.90184294e-01   4.58589784e-01  -1.86316421e-01
   -1.44950097e-01   2.57003424e-01  -1.76255501e-01   5.43682719e-02
    5.39054897e-02  -6.74380911e-02]
 [ -8.73210214e-01  -3.15481101e-01  -4.52313727e-01   5.10087132e-01
    2.39879728e-01  -2.54209144e-01   7.77796319e-02  -3.17255514e-01
    5.96125655e-02  -6.72034723e-02]
 [ -1.04701683e+00   5.06957678e-01   1.65034759e-01   6.28882045e-02
   -3.16521571e-01  -1.70166797e-03  -1.12827456e-01   5.06165196e-01
    4.48174302e-01   1.95886373e-02]
 [  4.78044292e-01  -1.03693257e+00   6.19236160e-02   6.02545112e-01
    2.49461500e-01   5.34155860e-01  -3.94402495e-01   3.97121822e-02
   -1.32606750e-02  -3.57114421e-02]
 [ -1.44910083e-01  -5.15782953e-01   1.17126612e-01  -4.68922193e-01
    8.49711247e-04   4.90693297e-01   6.00636280e-01  -2.03761210e-01
   -1.43230553e-01   1.15969760e-01]
 [ -8.76567539e-01   2.52398470e-02  -3.62171499e-01  -1.51979963e-02
   -6.62664808e-01  -1.32859151e-01  -9.49085902e-02  -9.21532905e-02
    2.56737440e-02  -2.99028079e-02]
 [  8.81622116e-01   1.38966958e-01   5.35224668e-02   2.85590120e-01
   -2.71034805e-01  -3.61835304e-01   7.93204478e-01   1.89815523e-01
    1.75900471e-01   8.29218930e-03]
 [ -5.17337740e-01  -1.13403932e-01   4.40286924e-02   2.40972468e-01
   -8.17922968e-01   4.18446461e-01   3.83048885e-01  -6.75026163e-02
   -3.00687318e-01   1.11534217e-01]
 [  6.47239775e-01  -9.90184294e-01   4.58589784e-01  -1.86316421e-01
   -1.44950097e-01   2.57003424e-01  -1.76255501e-01   5.43682719e-02
    5.39054897e-02  -6.74380911e-02]
 [ -6.76692688e-01  -8.31665059e-02  -5.95597519e-01  -4.61516253e-01
   -3.52025068e-01   3.37890114e-01  -2.68156411e-01  -1.79657504e-02
    8.20255228e-02   4.35600486e-02]
 [ -7.55931819e-01   8.87632782e-01   5.87715302e-01   1.29867533e-01
   -1.80598180e-01   2.39275806e-01  -1.91282556e-01   1.87777127e-01
   -1.46041447e-01   1.27273696e-02]
 [  8.40799374e-01   8.43852157e-01  -4.69947142e-01  -8.63428687e-02
```

In [14]:
```
# Contributions des points lignes
print(my_mca.row_contrib_)
```

```
[[  7.73767876e-01   1.67959126e+00   1.80760743e-01   1.05091056e+00
    3.46471633e-01   2.14446056e+01   3.60535532e-01   3.30865774e+00
    4.38859435e+00   2.49767800e-01]
 [  4.96577656e-01   1.16741602e+01   6.38490092e-01   2.01310302e+00
    6.77388350e+00   4.78101032e-02   1.26908835e-01   1.62303361e+01
    1.59655665e+00   1.03908074e+01]
 [  1.81937974e+00   2.07658208e+00   4.35652825e+00   7.83787399e+00
    1.87789320e+00   9.68337324e+00   7.57722059e-01   2.84592660e+00
    8.62805061e+00   3.82062375e-02]
 [  1.53910433e+00   7.48497608e+00   8.40776174e+00   1.58913049e+00
    5.12046123e+00   1.37241366e+00   4.11284205e-04   3.18553782e-02
    9.41477848e-04   9.53320231e+00]
 [  7.89705226e+00   2.91076402e+00   4.68895218e-01   2.87834243e+00
    2.69933348e+00   1.21862409e+00   1.83581510e-01   5.06586878e-01
    1.99413231e-01   2.12627267e+00]
 [  4.35555187e+00   2.87943048e+00   4.34672445e+00   1.00900136e+01
    1.28579249e+01   4.17741330e-01   1.57195441e-02   1.12982040e-01
    1.97708842e+00   2.89959779e-01]
 [  6.40060403e+00   2.52255635e-03   5.83649387e+00   9.27492702e+00
    4.64700961e+00   4.47727416e+00   7.34039564e+00   5.04214975e+00
    4.28761188e+00   6.18390213e+00]
 [  5.43661975e+00   6.85495624e+00   3.87745169e+00   1.75250667e-01
    7.75900532e-01   1.16683960e+00   4.87020447e+00   1.40044921e+00
    2.94021250e+00   2.96303915e-03]
 [  4.13525212e+00   6.01908248e-02   7.69957277e+00   8.46328783e-01
    2.70053951e-01   1.17989984e+01   1.02559704e+01   1.33689603e+01
    3.24705079e+00   4.46139180e+01]
 [  1.05858761e-01   2.66453355e+00   1.96907570e+00   1.01703158e+01
    9.10651649e-01   4.20555184e+00   3.22313741e-01   6.18547372e+00
    1.92677615e+00   1.23218295e+00]
 [  3.22162209e+00   9.43852268e+00   3.69230120e+00   8.16035401e-01
    5.18320094e-01   1.98411803e+00   1.41242071e+00   2.39716870e-01
    4.57151890e-01   2.18384353e+00]
 [  5.86383597e+00   9.58117236e-01   3.59193035e+00   6.11639421e+00
    1.41954159e+00   1.94120777e+00   2.75049260e-01   8.16255080e+00
    5.59075103e-01   2.16867467e+00]
 [  8.43046492e+00   2.47408927e+00   4.78188600e-01   9.29704908e-02
    2.47154005e+00   8.69839921e-05   5.78773059e-01   2.07774430e+01
    3.16000979e+01   1.84255063e-01]
 [  1.75743986e+00   1.03507779e+01   6.73226167e-02   8.53465298e+00
    1.53521105e+00   8.57088522e+00   7.07225679e+00   1.27895435e-01
    2.76646889e-02   6.12386712e-01]
 [  1.61488428e-01   2.56097846e+00   2.40857143e-01   5.16902112e+00
    1.78116339e-05   7.23285797e+00   1.64021899e+01   3.36705771e+00
    3.22749712e+00   6.45803658e+00]
 [  5.90901328e+00   6.13261066e-03   2.30290971e+00   5.42974456e-03
    1.08330025e+01   5.30240175e-01   4.09533807e-01   6.88698652e-01
    1.03698611e-01   4.29373190e-01]
 [  5.97735639e+00   1.85906708e-01   5.02945136e-02   1.91731332e+00
    1.81222100e+00   3.93288786e+00   2.86054661e+01   2.92193795e+00
    4.86775300e+00   3.30178961e-02]
 [  2.05822345e+00   1.23802149e-01   3.40345447e-02   1.36502808e+00
    1.65038716e+01   5.25980306e+00   6.67094131e+00   3.69529191e-01
    1.42240985e+01   5.97347744e+00]
 [  3.22162209e+00   9.43852268e+00   3.69230120e+00   8.16035401e-01
    5.18320094e-01   1.98411803e+00   1.41242071e+00   2.39716870e-01
    4.57151890e-01   2.18384353e+00]
 [  3.52149573e+00   6.65838446e-02   6.22807925e+00   5.00703621e+00
    3.05708990e+00   3.42957778e+00   3.26930233e+00   2.61756975e-02
    1.05850409e+00   9.11145413e-01]
 [  4.39449982e+00   7.58470376e+00   6.06432357e+00   3.96467792e-01
    8.04614339e-01   1.71983690e+00   1.66352543e+00   2.85951856e+00
    3.35541916e+00   7.77836458e-02]
 [  5.43661975e+00   6.85495624e+00   3.87745169e+00   1.75250667e-01
```

In [15]:
```
# Cos2 des points lignes
print(my_mca.row_cos2_)
```

```
[[  8.86354732e-02   1.53699594e-01   9.06978140e-03   3.93822110e-02
    1.23722226e-02   6.28882412e-01   6.98570800e-03   3.59406044e-02
    2.45737810e-02   4.58212084e-04]
 [  3.38043142e-02   6.34867136e-01   1.90385973e-02   4.48320320e-02
    1.43749337e-01   8.33218971e-04   1.46130981e-03   1.04772939e-01
    5.31274484e-03   1.13283712e-02]
 [  1.53722499e-01   1.40163659e-01   1.61231704e-01   2.16645575e-01
    4.94615977e-02   2.09457186e-01   1.08290285e-02   2.28020784e-02
    3.56349731e-02   5.16989317e-05]
 [  1.11330751e-01   4.32523528e-01   2.66393303e-01   3.76048711e-02
    1.15462068e-01   2.54147956e-02   5.03216176e-06   2.18507578e-04
    3.32894680e-06   1.10438146e-02]
 [  6.24484641e-01   1.83880633e-01   1.62415840e-02   7.44623393e-02
    6.65420938e-02   2.46706743e-02   2.45556352e-03   3.79880801e-03
    7.70833162e-04   2.69282992e-03]
 [  2.70690765e-01   1.42958206e-01   1.18328181e-01   2.05144306e-01
    2.49106067e-01   6.64648773e-03   1.65247798e-04   6.65849872e-04
    6.00628608e-03   2.88603256e-04]
 [  3.85193916e-01   1.21275082e-04   1.53853124e-01   1.82602376e-01
    8.71796657e-02   6.89805511e-02   7.47211265e-02   2.87747035e-02
    1.26131510e-02   5.96011012e-03]
 [  3.79931295e-01   3.82695220e-01   1.18691149e-01   4.00657480e-03
    1.69030599e-02   2.08757353e-02   5.75690136e-02   9.28068713e-03
    1.00439490e-02   3.31624583e-06]
 [  2.79156818e-01   3.24600200e-03   2.27671518e-01   1.86905820e-02
    5.68303006e-03   2.03913464e-01   1.17108519e-01   8.55816269e-02
    1.07148229e-02   4.82336169e-02]
 [  1.23961745e-02   2.49260987e-01   1.00999475e-01   3.89612409e-01
    3.32427027e-02   1.26077730e-01   6.38418085e-03   6.86863599e-02
    1.10291440e-02   2.31083688e-03]
 [  2.36285170e-01   5.53016560e-01   1.18619154e-01   1.95798039e-02
    1.18506750e-02   3.72549417e-02   1.75223129e-02   1.66723619e-03
    1.63897399e-03   2.56517269e-03]
 [  4.87611688e-01   6.36477694e-02   1.30832617e-01   1.66389236e-01
    3.67979467e-02   4.13255677e-02   3.86872855e-03   6.43657523e-02
    2.27254304e-03   2.88815115e-03]
 [  5.60793909e-01   1.31473847e-01   1.39330697e-02   2.02317920e-03
    5.12510285e-02   1.48130883e-06   6.51216736e-03   1.31063126e-01
    1.02751900e-01   1.96292818e-04]
 [  1.04983391e-01   4.93952692e-01   1.76155797e-03   1.66787484e-01
    2.85885007e-02   1.31075135e-01   7.14601014e-02   7.24489049e-04
    8.07821830e-05   5.85867084e-04]
 [  1.75332292e-02   2.22125629e-01   1.14544928e-02   1.83597294e-01
    6.02847459e-07   2.01041147e-01   3.01222788e-01   3.46663185e-02
    1.71291589e-02   1.12293397e-02]
 [  5.58313038e-01   4.62892815e-04   9.53093579e-02   1.67833893e-04
    3.19076498e-01   1.28259619e-02   6.54512653e-03   6.17061850e-03
    4.78944747e-04   6.49727046e-04]
 [  4.36271010e-01   1.08396312e-02   1.60791707e-03   4.57802078e-02
    4.12326772e-02   7.34874674e-02   3.53152039e-01   2.02234317e-02
    1.73670242e-02   3.85948911e-05]
 [  1.86023209e-01   8.93871388e-03   1.34738090e-03   4.03602327e-02
    4.64990004e-01   1.21702250e-01   1.01982866e-01   3.16708720e-03
    6.28418601e-02   8.64639470e-03]
 [  2.36285170e-01   5.53016560e-01   1.18619154e-01   1.95798039e-02
    1.18506750e-02   3.72549417e-02   1.75223129e-02   1.66723619e-03
    1.63897399e-03   2.56517269e-03]
 [  3.38815590e-01   5.11772953e-03   2.62473933e-01   1.57599349e-01
    9.16911884e-02   8.44756203e-02   5.32055317e-02   2.38820247e-04
    4.97826984e-03   1.40396775e-03]
 [  2.99995069e-01   4.13633334e-01   1.81335511e-01   8.85421459e-03
    1.71228323e-02   3.00570559e-02   1.92087717e-02   1.85111860e-02
    1.11969855e-02   8.50405730e-05]
 [  3.79931295e-01   3.82695220e-01   1.18691149e-01   4.00657480e-03
```

I.3. Extraction des statistiques sur les points colonnes

Export de la totalité des données colonnes vers une DataFrame pandas

On peut envoyer vers une Dataframe : les coordonnées, les contributions et les cos2 de chacun des points **colonnes**, pour tous les axes factoriels (identifiés par les suffixes dim1, dim2, etc.).

In [16]:
```
df_cols = my_mca.col_topandas()
```

In [17]:
```
print(df_cols)
```

```
                      col_coord_dim1  col_coord_dim2  col_coord_dim3  \
Taille_Taille+              0.851088       -1.231720        1.016052
Taille_Taille++            -0.836675       -0.020578       -0.051217
Taille_Taille-              1.184956        0.923897       -0.616000
Poids_Poids+               -0.305405       -0.818876       -0.231272
Poids_Poids++              -1.015134        0.973901        1.221595
Poids_Poids-                1.168918        0.824345       -0.358770
Velocite_Veloc+             0.603687       -0.887814        0.356312
Velocite_Veloc++           -0.892100       -0.371832       -0.763088
Velocite_Veloc-             0.319941        1.044900        0.401729
Intelligence_Intell+        0.369443       -0.285503        0.493203
Intelligence_Intell++      -0.335066       -0.459483       -0.599924
Intelligence_Intell-       -0.349045        0.808555       -0.351511
Affection_Affec+            0.775496       -0.266936       -0.060797
Affection_Affec-           -0.835150        0.287470        0.065474
Agressivite_Agress+        -0.431539        0.209196        0.333548
Agressivite_Agress-         0.400714       -0.194253       -0.309723

                      col_coord_dim4  col_coord_dim5  col_coord_dim6  \
Taille_Taille+              0.342456       -0.310040        0.118297
Taille_Taille++            -0.170222        0.112663       -0.049965
Taille_Taille-              0.120149       -0.019963        0.022569
Poids_Poids+               -0.118364       -0.190201        0.012908
Poids_Poids++               0.067605        0.614518        0.289232
Poids_Poids-                0.164884       -0.051221       -0.203360
Velocite_Veloc+             0.370243       -0.371036        0.629313
Velocite_Veloc++           -0.239848       -0.010089       -0.532181
Velocite_Veloc-            -0.080331        0.305908       -0.024488
Intelligence_Intell+       -0.603492        0.146256       -0.378518
Intelligence_Intell++       1.275249        1.063191        0.205389
Intelligence_Intell-        0.024238       -1.035060        0.461050
Affection_Affec+            0.077216        0.040322       -0.318067
Affection_Affec-           -0.083156       -0.043424        0.342534
Agressivite_Agress+         0.551156       -0.374464       -0.514255
Agressivite_Agress-        -0.511788        0.347717        0.477522

                      col_coord_dim7  col_coord_dim8  col_coord_dim9  \
Taille_Taille+             -0.858306       -0.259599        0.307322
Taille_Taille++             0.117844        0.056414       -0.144633
Taille_Taille-              0.360553        0.064541        0.090411
Poids_Poids+               -0.037178       -0.125673       -0.184946
Poids_Poids++              -0.067864        0.641508        0.204009
Poids_Poids-                0.107475       -0.181014        0.196151
Velocite_Veloc+             0.625743        0.173445       -0.008821
Velocite_Veloc++           -0.192758        0.141837        0.291628
Velocite_Veloc-            -0.327112       -0.266409       -0.255408
Intelligence_Intell+        0.281329       -0.075777        0.041319
Intelligence_Intell++      -0.092247       -0.094569       -0.020514
Intelligence_Intell-       -0.387975        0.194064       -0.051758
Affection_Affec+           -0.170577        0.311516       -0.130227
Affection_Affec-            0.183698       -0.335479        0.140244
Agressivite_Agress+         0.153837       -0.049324       -0.026899
Agressivite_Agress-        -0.142849        0.045801        0.024978

                     col_coord_dim10       ...        col_cos2_dim1  \
Taille_Taille+             -0.015208       ...             0.164625
Taille_Taille++             0.121550       ...             0.875032
Taille_Taille-             -0.249601       ...             0.491442
Poids_Poids+               -0.097573       ...             0.100447
Poids_Poids++              -0.071494       ...             0.234204
Poids_Poids-                0.215436       ...             0.575313
Velocite_Veloc+             0.053786       ...             0.153447
Velocite_Veloc++           -0.020744       ...             0.397921
Velocite_Veloc-            -0.024359       ...             0.060213
```

Statistiques pour les points colonnes

In [18]:
```
# Coordonnées des points colonnes
print(my_mca.col_coord_)
```

```
[[ 0.85108805 -1.23171972  1.01605178  0.34245635 -0.31004022  0.11829709
  -0.85830588 -0.25959948  0.30732198 -0.01520824]
 [-0.83667535 -0.02057846 -0.05121744 -0.17022176  0.11266304 -0.04996469
   0.11784406  0.05641401 -0.14463269  0.12155001]
 [ 1.18495571  0.9238965  -0.61599962  0.12014924 -0.0199635   0.02256927
   0.36055265  0.06454103  0.09041149 -0.24960128]
 [-0.30540525 -0.81887572 -0.23127208 -0.11836395 -0.19020146  0.0129084
  -0.03717755 -0.12567337 -0.18494632 -0.09757299]
 [-1.01513413  0.97390062  1.22159452  0.06760494  0.61451838  0.28923175
  -0.06786357  0.64150795  0.20400871 -0.07149381]
 [ 1.16891802  0.82434462 -0.35877044  0.16488382 -0.05122143 -0.20335954
   0.10747545 -0.18101407  0.19615062  0.21543637]
 [ 0.60368666 -0.88781355  0.35631249  0.37024339 -0.37103561  0.62931321
   0.62574307  0.17344491 -0.00882083  0.05378561]
 [-0.89209989 -0.37183247 -0.76308752 -0.23984823 -0.01008873 -0.5321807
  -0.19275804  0.1418371   0.29162795 -0.02074368]
 [ 0.31994057  1.04490006  0.40172878 -0.0803313   0.30590834 -0.02448794
  -0.32711222 -0.26640932 -0.25540849 -0.02435918]
 [ 0.3694426  -0.28550314  0.49320252 -0.6034918   0.14625633 -0.37851765
   0.28132937 -0.07577653  0.04131926 -0.01449247]
 [-0.33506557 -0.45948302 -0.59992378  1.27524863  1.06319128  0.20538874
  -0.09224687 -0.09456899 -0.02051408  0.00222567]
 [-0.34904504  0.80855486 -0.35151126  0.02423769 -1.03505999  0.46104962
  -0.38797506  0.19406361 -0.05175824  0.02188101]
 [ 0.77549643 -0.26693613 -0.06079688  0.07721587  0.04032182 -0.31806714
  -0.17057709  0.31151592 -0.13022666  0.01931353]
 [-0.83515001  0.28746968  0.06547357 -0.08315555 -0.0434235   0.34253384
   0.1836984  -0.33547868  0.14024409 -0.02079919]
 [-0.43153864  0.20919553  0.33354829  0.55115649 -0.37446406 -0.51425491
   0.15383745 -0.04932429 -0.02689949 -0.02007225]
 [ 0.40071446 -0.19425299 -0.30972341 -0.51178817  0.34771663  0.47752242
  -0.14284906  0.04580113  0.0249781   0.01863851]]
```

In [19]:
```
# Contributions des points colonnes
print(my_mca.col_contrib_)
```

```
[[  4.64207270e+00   1.21706703e+01   1.51042375e+01   2.29739639e+00
    1.97613161e+00   3.50313262e-01   2.79114015e+01   4.55442881e+00
    1.23822770e+01   9.25524242e-02]
 [  1.34585463e+01   1.01914919e-02   1.15139433e-01   1.70285200e+00
    7.82823704e-01   1.87480502e-01   1.57846191e+00   6.45240301e-01
    8.22749141e+00   1.77362173e+01]
 [  1.25978150e+01   9.58661729e+00   7.77241612e+00   3.95909184e-01
    1.14704658e-02   1.78513713e-02   6.89545420e+00   3.94117907e-01
    1.50033602e+00   3.49020865e+01]
 [  1.67368602e+00   1.50620723e+01   2.19114722e+00   7.68462024e-01
    2.08240645e+00   1.16791442e-02   1.46628075e-01   2.98862105e+00
    1.25563298e+01   1.06671161e+01]
 [  6.60404174e+00   7.60886705e+00   2.18333953e+01   8.95328362e-02
    7.76335649e+00   2.09411430e+00   1.74490143e-01   2.78118821e+01
    5.45645635e+00   2.04534510e+00]
 [  1.40104164e+01   8.72224556e+00   3.01314769e+00   8.52121591e-01
    8.62984888e-02   1.65637063e+00   7.00222427e-01   3.54299832e+00
    8.07072682e+00   2.97158224e+01]
 [  3.73685373e+00   1.01170578e+01   2.97200269e+00   4.29655213e+00
    4.52825567e+00   1.58621525e+01   2.37361098e+01   3.25289013e+00
    1.63211906e-02   1.85217469e+00]
 [  9.18041736e+00   1.99644722e+00   1.53351608e+01   2.02848142e+00
    3.76638865e-03   1.27614255e+01   2.53393206e+00   2.44725085e+00
    2.00698479e+01   3.09937280e-01]
 [  1.31199315e+00   1.75174229e+01   4.72240258e+00   2.52827940e-01
    3.84762211e+00   3.00222251e-02   8.10813552e+00   9.59301475e+00
    1.71046406e+01   4.74881221e-01]
 [  2.27420826e+00   1.70014447e+00   9.25318084e+00   1.85499001e+01
    1.14335856e+00   9.32511183e+00   7.79652272e+00   1.00894867e+00
    5.81957996e-01   2.18518362e-01]
 [  8.63383636e-01   2.03240794e+00   6.31888605e+00   3.82293106e+01
    2.78858471e+01   1.26719352e+00   3.86885106e-01   7.25278956e-01
    6.62061645e-02   2.37866987e-03]
 [  1.24923997e+00   8.39130827e+00   2.89244826e+00   1.84131478e-02
    3.52395869e+01   8.51380619e+00   9.12483625e+00   4.07225041e+00
    5.61942915e-01   3.06538061e-01]
 [  1.07914697e+01   1.60052866e+00   1.51421587e-01   3.27036829e-01
    9.35875699e-02   7.09094445e+00   3.08671758e+00   1.83630496e+01
    6.22545488e+00   4.17937293e-01]
 [  1.16215827e+01   1.72364624e+00   1.63069402e-01   3.52193508e-01
    1.00786614e-01   7.63640171e+00   3.32415740e+00   1.97755919e+01
    6.70433602e+00   4.50086316e-01]
 [  3.10295649e+00   9.12785746e-01   4.23211941e+00   1.54720794e+01
    7.49503059e+00   1.72122911e+01   2.33128271e+00   4.27485436e-01
    2.46646321e-01   4.19174645e-01]
 [  2.88131675e+00   8.47586764e-01   3.92982517e+00   1.43669309e+01
    6.95967126e+00   1.59828417e+01   2.16476252e+00   3.96950762e-01
    2.29028726e-01   3.89233599e-01]]
```

In [20]:
```
# Cos2 des points colonnes
print(my_mca.col_cos2_)
```

```
[[  1.64625197e-01   3.44803059e-01   2.34627550e-01   2.66537163e-02
    2.18465774e-02   3.18050046e-03   1.67429315e-01   1.53163382e-02
    2.14651811e-02   5.25660553e-05]
 [  8.75032050e-01   5.29341341e-04   3.27903251e-03   3.62193103e-02
    1.58662007e-02   3.12058795e-03   1.73590277e-02   3.97817561e-03
    2.61482672e-02   1.84680068e-02]
 [  4.91442014e-01   2.98754660e-01   1.32809435e-01   5.05254394e-03
    1.39489425e-04   1.78280217e-04   4.54993737e-02   1.45794078e-03
    2.86098291e-03   2.18052798e-02]
 [  1.00447166e-01   7.22138784e-01   5.76011410e-02   1.50877186e-02
    3.89594100e-02   1.79444121e-04   1.48849136e-03   1.70087034e-02
    3.68363070e-02   1.02528343e-02]
 [  2.34203931e-01   2.15564186e-01   3.39157541e-01   1.03873360e-03
    8.58256441e-02   1.90125017e-02   1.04669646e-03   9.35301025e-02
    9.45898915e-03   1.16167377e-03]
 [  5.75313407e-01   2.86123812e-01   5.41963075e-02   1.14470212e-02
    1.10468845e-03   1.74126746e-02   4.86356754e-03   1.37962500e-02
    1.62000281e-02   1.95422436e-02]
 [  1.53447405e-01   3.31879115e-01   5.34562485e-02   5.77179639e-02
    5.79652297e-02   1.66751629e-01   1.64865004e-01   1.26665839e-02
    3.27608348e-05   1.21805981e-03]
 [  3.97921103e-01   6.91296921e-02   2.91151283e-01   2.87635872e-02
    5.08912185e-05   1.41608150e-01   1.85778305e-02   1.00588815e-02
    4.25234310e-02   2.15150115e-04]
 [  6.02129213e-02   6.42244786e-01   9.49329478e-02   3.79595157e-03
    5.50470073e-02   3.52740658e-04   6.29425906e-02   4.17493676e-02
    3.83726466e-02   3.49040920e-04]
 [  1.26738700e-01   7.56897524e-02   2.25873819e-01   3.38187895e-01
    1.98629905e-02   1.33041636e-01   7.34929109e-02   5.33193380e-03
    1.58533237e-03   1.95029438e-04]
 [  3.20768394e-02   6.03213262e-02   1.02831012e-01   4.64645451e-01
    3.22964487e-01   1.20527241e-02   2.43128154e-03   2.55522703e-03
    1.20236370e-04   1.41532164e-06]
 [  5.12978688e-02   2.75267773e-01   5.20253342e-02   2.47354011e-04
    4.51094394e-01   8.95017905e-02   6.33788003e-02   1.58571300e-02
    1.12796422e-03   2.01590970e-04]
 [  6.47655853e-01   7.67360421e-02   3.98058885e-03   6.42092843e-03
    1.75091491e-03   1.08948761e-01   3.13347388e-02   1.04506948e-01
    1.82635190e-02   4.01705745e-04]
 [  6.47655853e-01   7.67360421e-02   3.98058885e-03   6.42092843e-03
    1.75091491e-03   1.08948761e-01   3.13347388e-02   1.04506948e-01
    1.82635190e-02   4.01705745e-04]
 [  1.72923773e-01   4.06368567e-02   1.03307716e-01   2.82075371e-01
    1.30207379e-01   2.45568248e-01   2.19755343e-02   2.25910803e-03
    6.71897969e-04   3.74116824e-04]
 [  1.72923773e-01   4.06368567e-02   1.03307716e-01   2.82075371e-01
    1.30207379e-01   2.45568248e-01   2.19755343e-02   2.25910803e-03
    6.71897969e-04   3.74116824e-04]]
```

I.4. Graphiques

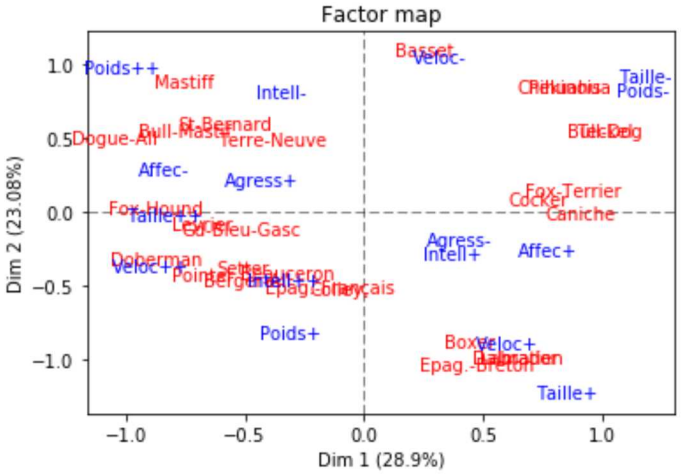
2 types de graphiques peuvent être réalisés :

- Les mapping classiques qui représentent les points lignes et colonnes sur un plan factoriel
- Des graphiques qui permettent d'interpréter rapidement les axes : on choisit un axe factoriel (le 1er axe dans notre exemple) et on observe quels sont les points lignes et colonnes qui présentent les plus fortes contributions et cos2 pour cet axe

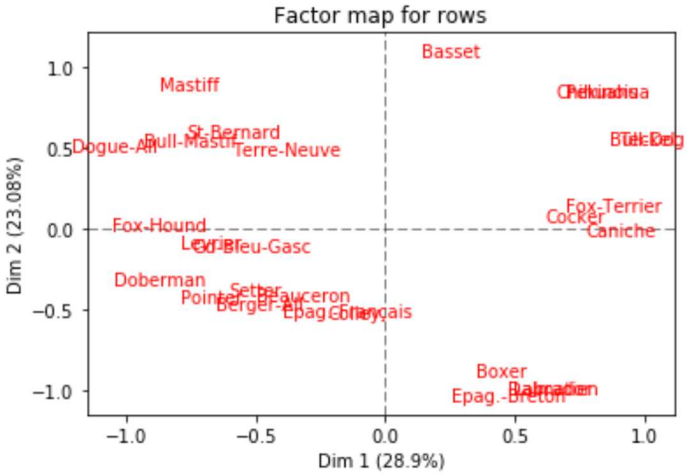
</p>

Graphiques factoriels

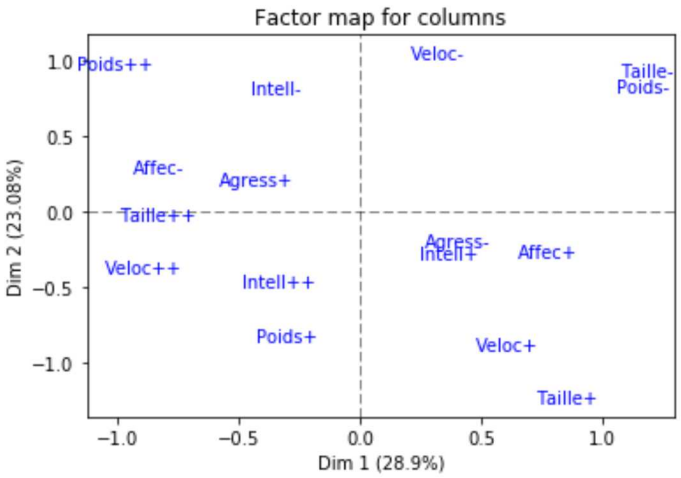
In [21]:
```
# Mapping simultané des points lignes et colonnes
# Les paramètres de la méthode mapping indiquent que ce sont les axes 1 et 2 qui so
nt ici représentés
my_mca.mapping(num_x_axis=1, num_y_axis=2)
```



In [22]:
```
# Mapping des points lignes
my_mca.mapping_row(num_x_axis=1, num_y_axis=2)
```

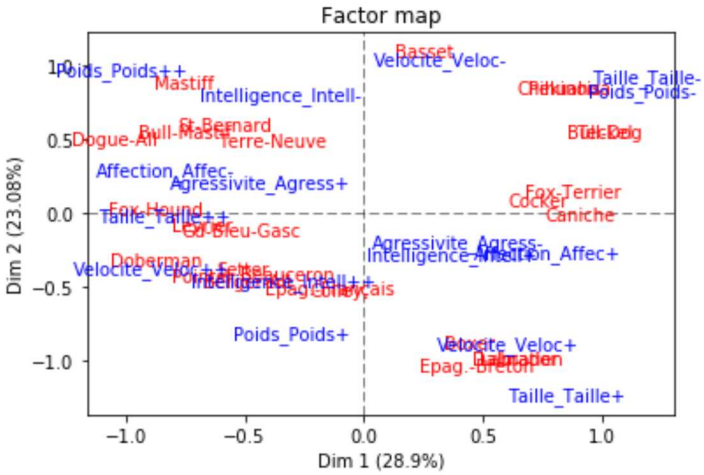


In [23]:
```
# Mapping des points colonnes
my_mca.mapping_col(num_x_axis=1, num_y_axis=2)
```



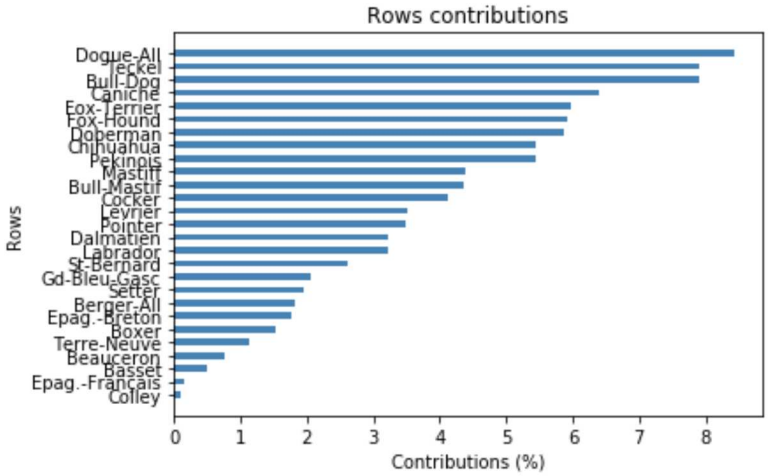
Sur les 3 graphiques factoriels précédents, les catégories ne sont pas préfixées par les noms des variables auxquelles elles appartiennent (c'est le choix fait par défaut par le package).
Ce choix est pertinent dans cet exemple car l'omission des préfixes permet d'alléger les graphiques, et il n'y a pas de risque de confusion entre les catégories.
Mais quand l'ensemble des catégories présente des doublons, il est préférable de les préfixer par les noms de variables pour lever toute ambiguïté, ce qui alourdit la présentation.
Dans ce cas, on fixe le paramètre *short_labels* à *False*

In [24]:
```
my_mca.mapping(num_x_axis=1, num_y_axis=2, short_labels=False)
```

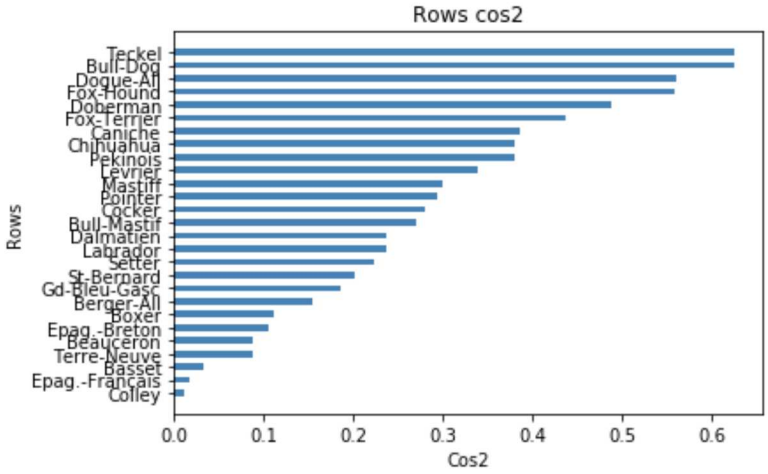


Analyse du 1er axe - Points lignes

In [25]:
```
# Classement des points lignes en fonction de leur contribution au 1er axe
# Le paramètre de la méthode plot_row_contrib indique que c'est pour l'axe numéro 1
que les contributions sont ici
# représentées
my_mca.plot_row_contrib(num_axis=1)
```

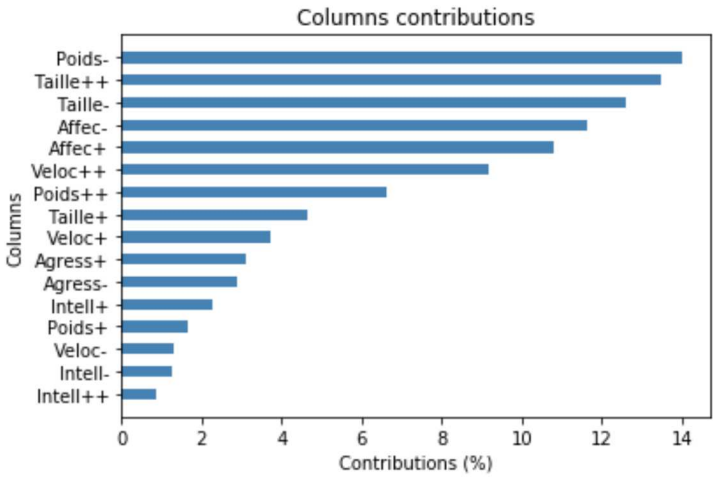


In [26]:
```
# Classement des points lignes en fonction de leur cos2 sur le 1er axe
my_mca.plot_row_cos2(num_axis=1)
```

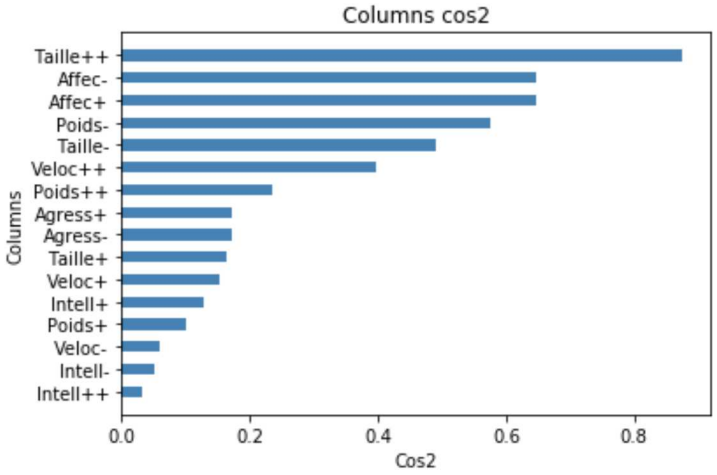


Analyse du 1er axe - Points colonnes

In [27]:
```
# Classement des points colonnes en fonction de leur contribution au 1er axe
my_mca.plot_col_contrib(num_axis=1)
```

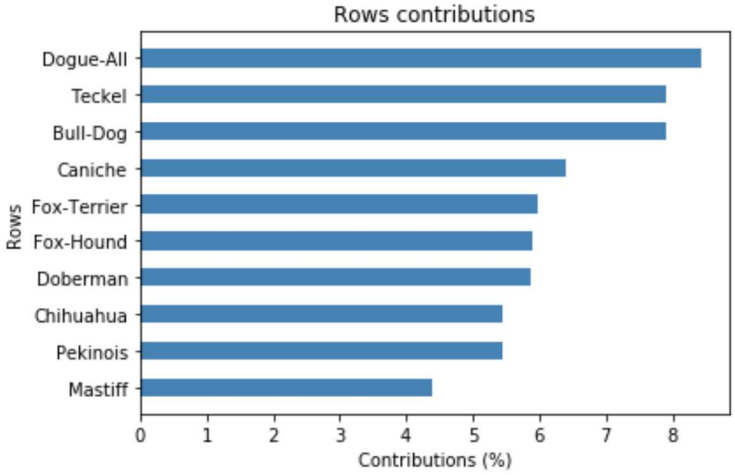


In [28]:
```
# Classement des points colonnes en fonction de leur cos2 sur le 1er axe
my_mca.plot_col_cos2(num_axis=1)
```



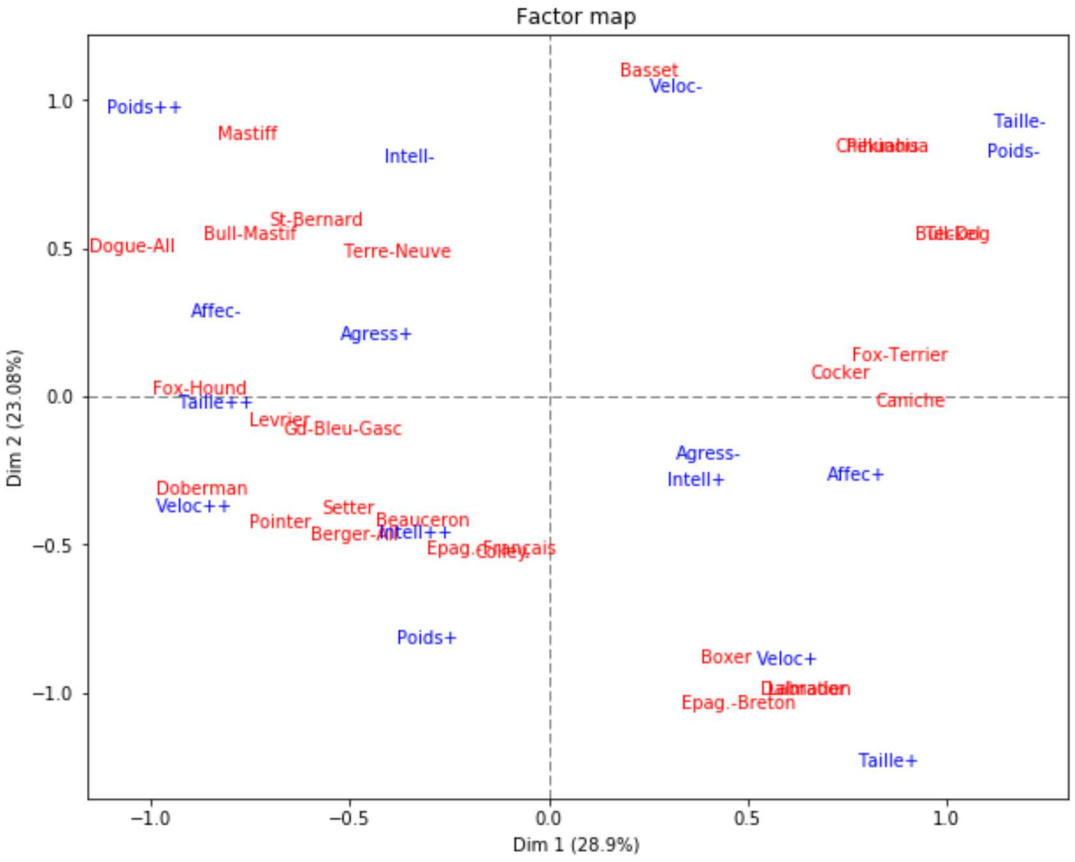
Pour ces graphiques produits par les méthodes *plot_row_contrib*, *plot_row_cos2*, *plot_col_contrib*, *plot_col_cos2*, on peut se limiter à visualiser les x valeurs les plus grandes via le paramètre nb_values.

In [29]:
```
my_mca.plot_row_contrib(num_axis=1, nb_values=10)
```



Pour tous les graphiques présentés plus haut, il est possible de définir un taille particulière via le paramètre *figsize*.

In [30]:
```
my_mca.mapping(1, 2, figsize=(10, 8))
```



II. Approche Machine Learning

Ici, l'objectif est d'utiliser l'Analyse des Correspondance Multiples en tant que méthode de prétraitement.

La classe MCA implémente les méthodes *fit*, *transform* et *fit_transform* bien connues des utilisateurs de scikit-learn.

Il est ici judicieux de fixer le paramètre *stats* à *False* pour gagner en temps de traitement et en ressources mémoire.

In [31]:
```
my_mca = MCA(stats=False)
```

In [32]:
```
my_mca.fit(X)
```

Out[32]:
```
MCA(n_components=None, row_labels=None, stats=False, var_labels=None)
```

In [33]:
```
my_mca.transform(X)
```

```
Out[33]: array([[ -3.17200051e-01,  -4.17701298e-01,  -1.01467714e-01,
                  -2.11436281e-01,  -1.18509544e-01,  -8.44917274e-01,
                   8.90501548e-02,   2.01986411e-01,  -1.67018837e-01,
                  -2.28067149e-02],
                 [  2.54109843e-01,   1.10122699e+00,  -1.90700968e-01,
                   2.92637269e-01,  -5.24008519e-01,   3.98946811e-02,
                   5.28331582e-02,  -4.47362922e-01,   1.00738366e-01,
                  -1.47102197e-01],
                 [ -4.86395534e-01,  -4.64449578e-01,  -4.98133882e-01,
                   5.77425252e-01,   2.75902052e-01,  -5.67764838e-01,
                  -1.29096839e-01,   1.87330321e-01,  -2.34185002e-01,
                   8.91993408e-03],
                 [  4.47364924e-01,  -8.81777941e-01,   6.92015804e-01,
                   2.60001835e-01,  -4.55589836e-01,  -2.13745841e-01,
                  -3.00768021e-03,  -1.98192681e-02,  -2.44628915e-03,
                  -1.40900948e-01],
                 [  1.01335218e+00,   5.49879489e-01,  -1.63423202e-01,
                  -3.49919273e-01,   3.30786485e-01,  -2.01414177e-01,
                   6.35440823e-02,  -7.90356807e-02,  -3.56024365e-02,
                  -6.65432469e-02],
                 [ -7.52574495e-01,   5.46911834e-01,   4.97573073e-01,
                   6.55152661e-01,   7.21946355e-01,   1.17925813e-01,
                  -1.85943335e-02,  -3.73250962e-02,  -1.12102625e-01,
                  -2.45732947e-02],
                 [  9.12301484e-01,  -1.61876744e-02,  -5.76569722e-01,
                   6.28133396e-01,   4.34016531e-01,   3.86066396e-01,
                   4.01809663e-01,   2.49346973e-01,   1.65086085e-01,
                   1.13481695e-01],
                 [  8.40799374e-01,   8.43852157e-01,  -4.69947142e-01,
                  -8.63428687e-02,  -1.77346455e-01,   1.97088251e-01,
                  -3.27291133e-01,   1.31410453e-01,  -1.36707423e-01,
                   2.48406642e-03],
                 [  7.33295293e-01,   7.90731735e-02,   6.62230424e-01,
                   1.89743186e-01,  -1.04627236e-01,  -6.26726024e-01,
                  -4.74951235e-01,  -4.06017775e-01,   1.43663776e-01,
                   3.04810338e-01],
                 [ -1.17325200e-01,  -5.26107651e-01,  -3.34893734e-01,
                  -6.57754538e-01,   1.92130196e-01,  -3.74168009e-01,
                  -8.41976661e-02,   2.76173951e-01,  -1.10667059e-01,
                   5.06561417e-02],
                 [  6.47239775e-01,  -9.90184294e-01,   4.58589784e-01,
                  -1.86316421e-01,  -1.44950097e-01,   2.57003424e-01,
                  -1.76255501e-01,   5.43682719e-02,   5.39054897e-02,
                  -6.74380911e-02],
                 [ -8.73210214e-01,  -3.15481101e-01,  -4.52313727e-01,
                   5.10087132e-01,   2.39879728e-01,  -2.54209144e-01,
                   7.77796319e-02,  -3.17255514e-01,   5.96125655e-02,
                  -6.72034723e-02],
                 [ -1.04701683e+00,   5.06957678e-01,   1.65034759e-01,
                   6.28882045e-02,  -3.16521571e-01,  -1.70166797e-03,
                  -1.12827456e-01,   5.06165196e-01,   4.48174302e-01,
                   1.95886373e-02],
                 [  4.78044292e-01,  -1.03693257e+00,   6.19236160e-02,
                   6.02545112e-01,   2.49461500e-01,   5.34155860e-01,
                  -3.94402495e-01,   3.97121822e-02,  -1.32606750e-02,
                  -3.57114421e-02],
                 [ -1.44910083e-01,  -5.15782953e-01,   1.17126612e-01,
                  -4.68922193e-01,   8.49711247e-04,   4.90693297e-01,
                   6.00636280e-01,  -2.03761210e-01,  -1.43230553e-01,
                   1.15969760e-01],
                 [ -8.76567539e-01,   2.52398470e-02,  -3.62171499e-01,
                  -1.51979963e-02,  -6.62664808e-01,  -1.32859151e-01,
                  -9.49085902e-02,  -9.21532905e-02,   2.56737440e-02,
                  -2.99028079e-02],
```

In [34]:
```
my_mca.fit_transform(X)
```

```
Out[34]: array([[ -3.17200051e-01,  -4.17701298e-01,  -1.01467714e-01,
                  -2.11436281e-01,  -1.18509544e-01,  -8.44917274e-01,
                   8.90501548e-02,   2.01986411e-01,  -1.67018837e-01,
                  -2.28067149e-02],
                 [  2.54109843e-01,   1.10122699e+00,  -1.90700968e-01,
                   2.92637269e-01,  -5.24008519e-01,   3.98946811e-02,
                   5.28331582e-02,  -4.47362922e-01,   1.00738366e-01,
                  -1.47102197e-01],
                 [ -4.86395534e-01,  -4.64449578e-01,  -4.98133882e-01,
                   5.77425252e-01,   2.75902052e-01,  -5.67764838e-01,
                  -1.29096839e-01,   1.87330321e-01,  -2.34185002e-01,
                   8.91993408e-03],
                 [  4.47364924e-01,  -8.81777941e-01,   6.92015804e-01,
                   2.60001835e-01,  -4.55589836e-01,  -2.13745841e-01,
                  -3.00768021e-03,  -1.98192681e-02,  -2.44628915e-03,
                  -1.40900948e-01],
                 [  1.01335218e+00,   5.49879489e-01,  -1.63423202e-01,
                  -3.49919273e-01,   3.30786485e-01,  -2.01414177e-01,
                   6.35440823e-02,  -7.90356807e-02,  -3.56024365e-02,
                  -6.65432469e-02],
                 [ -7.52574495e-01,   5.46911834e-01,   4.97573073e-01,
                   6.55152661e-01,   7.21946355e-01,   1.17925813e-01,
                  -1.85943335e-02,  -3.73250962e-02,  -1.12102625e-01,
                  -2.45732947e-02],
                 [  9.12301484e-01,  -1.61876744e-02,  -5.76569722e-01,
                   6.28133396e-01,   4.34016531e-01,   3.86066396e-01,
                   4.01809663e-01,   2.49346973e-01,   1.65086085e-01,
                   1.13481695e-01],
                 [  8.40799374e-01,   8.43852157e-01,  -4.69947142e-01,
                  -8.63428687e-02,  -1.77346455e-01,   1.97088251e-01,
                  -3.27291133e-01,   1.31410453e-01,  -1.36707423e-01,
                   2.48406642e-03],
                 [  7.33295293e-01,   7.90731735e-02,   6.62230424e-01,
                   1.89743186e-01,  -1.04627236e-01,  -6.26726024e-01,
                  -4.74951235e-01,  -4.06017775e-01,   1.43663776e-01,
                   3.04810338e-01],
                 [ -1.17325200e-01,  -5.26107651e-01,  -3.34893734e-01,
                  -6.57754538e-01,   1.92130196e-01,  -3.74168009e-01,
                  -8.41976661e-02,   2.76173951e-01,  -1.10667059e-01,
                   5.06561417e-02],
                 [  6.47239775e-01,  -9.90184294e-01,   4.58589784e-01,
                  -1.86316421e-01,  -1.44950097e-01,   2.57003424e-01,
                  -1.76255501e-01,   5.43682719e-02,   5.39054897e-02,
                  -6.74380911e-02],
                 [ -8.73210214e-01,  -3.15481101e-01,  -4.52313727e-01,
                   5.10087132e-01,   2.39879728e-01,  -2.54209144e-01,
                   7.77796319e-02,  -3.17255514e-01,   5.96125655e-02,
                  -6.72034723e-02],
                 [ -1.04701683e+00,   5.06957678e-01,   1.65034759e-01,
                   6.28882045e-02,  -3.16521571e-01,  -1.70166797e-03,
                  -1.12827456e-01,   5.06165196e-01,   4.48174302e-01,
                   1.95886373e-02],
                 [  4.78044292e-01,  -1.03693257e+00,   6.19236160e-02,
                   6.02545112e-01,   2.49461500e-01,   5.34155860e-01,
                  -3.94402495e-01,   3.97121822e-02,  -1.32606750e-02,
                  -3.57114421e-02],
                 [ -1.44910083e-01,  -5.15782953e-01,   1.17126612e-01,
                  -4.68922193e-01,   8.49711247e-04,   4.90693297e-01,
                   6.00636280e-01,  -2.03761210e-01,  -1.43230553e-01,
                   1.15969760e-01],
                 [ -8.76567539e-01,   2.52398470e-02,  -3.62171499e-01,
                  -1.51979963e-02,  -6.62664808e-01,  -1.32859151e-01,
                  -9.49085902e-02,  -9.21532905e-02,   2.56737440e-02,
                  -2.99028079e-02],
```

Intégration dans une Pipeline de scikit-learn

La class MCA peut être intégrée dans une Pipeline de scikit-learn.
Dans le cadre de notre exemple, nous cherchons à prédire la 7ème variable (variable *Fonction*) à partir des 6 premières variables du jeu de données.
Fonction est une variable nominale comprenant 3 catégories : "chasse", "compagnie" et "utilite".
Pour la prédire, nous allons utiliser un modèle de régression logistique, qui prendra en input des axes issus d'une Analyse des Correspondances Multiples pratiquée sur les données brutes.
Dans un premier temps, et de façon tout à fait arbitraire, nous fixons le nombre de composantes extraites à 4.

In [35]:
```
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
```

In [36]:
```
df = pd.read_table("mca_data.txt", header=0, index_col=0, delimiter="\t", encoding=
"utf-8")
```

In [37]:
```
# X = features
X = df.iloc[:, 0:6].as_matrix()
# y = labels
y = df.iloc[:, 6].as_matrix()
```

In [38]:
```
# Construction de la Pipeline
# On enchaine une Analyse des Correspondances Multiples (4 axes retenus) puis une r
égression logistique multinomiale
pipe = Pipeline([("mca", MCA(n_components=4, stats=False)),
                 ("logit", LogisticRegression(multi_class="multinomial", solver="lb
fgs"))])
```

In [39]:
```
# Estimation du modèle
pipe.fit(X, y)
```

Out[39]:
```
Pipeline(steps=[('mca', MCA(n_components=4, row_labels=None, stats=False, var_la
bels=None)), ('logit', LogisticRegression(C=1.0, class_weight=None, dual=False,
fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='multinomial',
          n_jobs=1, penalty='l2', random_state=None, solver='lbfgs',
          tol=0.0001, verbose=0, warm_start=False))])
```

In [40]:
```
# Prédiction sur l'échantillon de test
print(pipe.predict(X))
```

```
['chasse' 'compagnie' 'chasse' 'compagnie' 'compagnie' 'utilite'
 'compagnie' 'compagnie' 'compagnie' 'chasse' 'compagnie' 'utilite'
 'utilite' 'compagnie' 'chasse' 'utilite' 'compagnie' 'utilite' 'compagnie'
 'chasse' 'utilite' 'compagnie' 'chasse' 'utilite' 'chasse' 'compagnie'
 'utilite']
```

Le paramètre *n_components* peut faire l'objet d'une optimisation via GridSearchCV de scikit-learn.
Nous reconstruisons donc une Pipeline, sans spécifier de valeur a priori pour *n_components*.

In [41]:
```
# Reconstruction d'une Pipeline, sans spécifier de valeur a priori pour n_component
s
pipe2 = Pipeline([("mca", MCA(stats=False)), ("logit", LogisticRegression(multi_cla
ss="multinomial", solver="lbfgs"))])
```

In [42]:
```
# Paramétrage de la grille de paramètres
# Attention à l'étendue des valeurs possibles pour mca__n_components !!!
param = [{"mca__n_components": [x + 1 for x in range(10)]}]
```

In [43]:
```
# Construction de l'obet GridSearchCV
grid_search = GridSearchCV(pipe2, param_grid=param, scoring="accuracy")
```

In [44]:
```
# Estimation du modèle
grid_search.fit(X, y)
```

Out[44]:
```
GridSearchCV(cv=None, error_score='raise',
       estimator=Pipeline(steps=[('mca', MCA(n_components=None, row_labels=None,
stats=False, var_labels=None)), ('logit', LogisticRegression(C=1.0, class_weight
=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='multinomial',
          n_jobs=1, penalty='l2', random_state=None, solver='lbfgs',
          tol=0.0001, verbose=0, warm_start=False))]),
       fit_params={}, iid=True, n_jobs=1,
       param_grid=[{'mca__n_components': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]}],
       pre_dispatch='2*n_jobs', refit=True, return_train_score=True,
       scoring='accuracy', verbose=0)
```

In [45]:
```
# Affichage du score optimal
grid_search.best_score_
```

Out[45]:
```
0.7777777777777779
```

In [46]:
```
# Affichage du paramètre optimal
grid_search.best_params_
```

Out[46]:
```
{'mca__n_components': 5}
```

In [47]:
```
# Prédiction sur l'échantillon de test
grid_search.predict(X)
```

Out[47]:
```
array(['chasse', 'compagnie', 'chasse', 'compagnie', 'compagnie',
       'utilite', 'compagnie', 'compagnie', 'compagnie', 'chasse',
       'compagnie', 'utilite', 'utilite', 'compagnie', 'chasse', 'chasse',
       'compagnie', 'chasse', 'compagnie', 'chasse', 'utilite',
       'compagnie', 'chasse', 'utilite', 'chasse', 'compagnie', 'utilite'], dtyp
e=object)
```