clustering

February 7, 2024

- Tanguy Malandain
- Hugo Deplagne
- Pierre Litoux
- Param Dave

1 Clustering

We will first form clusters based on questions of type `usage` and then `attitude`. To achieve this, we need to determine the optimal number of clusters through various methods.

1.0.1 Import modules

[1]:
```
import pandas as pd
```

[2]:
```
from utils import plot_cluster_metrics
from utils import make_clusters
from utils import feature_importance
from utils import features_per_cluster
from utils import compare_cluster_describe
from utils import compare_cluster_answers
```

1.0.2 Load dataframe

[3]:
```
codes = 'data/fic_epita_kantar_codes.csv'
labels = 'data/fic_epita_kantar_labels.csv'

df_codes = pd.read_csv(codes, sep=';')
```

[4]:
```
df_codes.head()
```

[4]:
```
   cle    Respondent_ID    weight  A11  A12  A13  A14  A4   A5  A5bis  …  \
0    1  MET20_999999996  2.501255    1    0    0    0   1  2.0    NaN  …
1    2   MET20_98888888  0.722914    1    0    0    0   1  5.0    NaN  …
2    3    MET20_1978307  1.039611    1    0    0    0   1  2.0    NaN  …
3    4    MET20_1302078  0.976590    1    1    1    0   1  1.0    NaN  …
4    5    MET20_1869308  0.812315    0    1    0    0   2  NaN    1.0  …

   RS193  RS102RECAP  rs11recap2  RS11recap  RS193bis  RS2Recap  RS56Recap  \
```

```
0      2           4           1           2       NaN         1           1
1      2           1           1           2       NaN         4           1
2      2           3           2           1       NaN         3           2
3      2           2           1           2       NaN         5           3
4      2           3           2           1       NaN         3           1

   RS2  RS11  RS102
0   24     0      4
1   50     0      1
2   37     1      3
3   63     0      2
4   44     1      3

[5 rows x 133 columns]
```

1.0.3 Fill NaN values

[5]:
```
columns_to_fill = ['A5', 'A5bis']
df_codes[columns_to_fill] = df_codes[columns_to_fill].fillna(0)

df_codes.iloc[:, 4:] = df_codes.iloc[:, 4:].fillna(df_codes.iloc[:, 4:].
 ↪median())
```

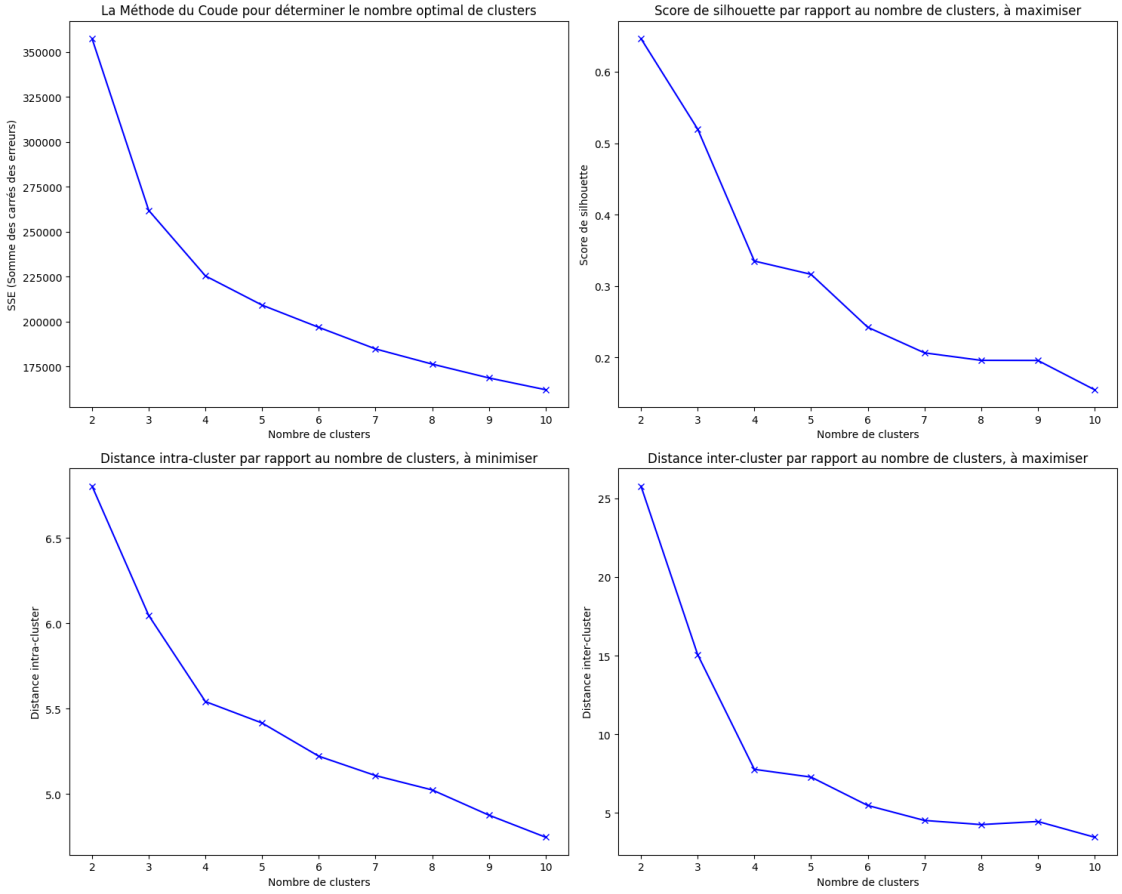
1.0.4 Separate usage and attitude questions

[6]:
```
df_usage = df_codes.iloc[:, 0:30].copy()
df_attitude = df_codes.iloc[:, list(range(3)) + list(range(30, 67))].copy()
```

1.1 Get optimal number of clusters

For usage questions

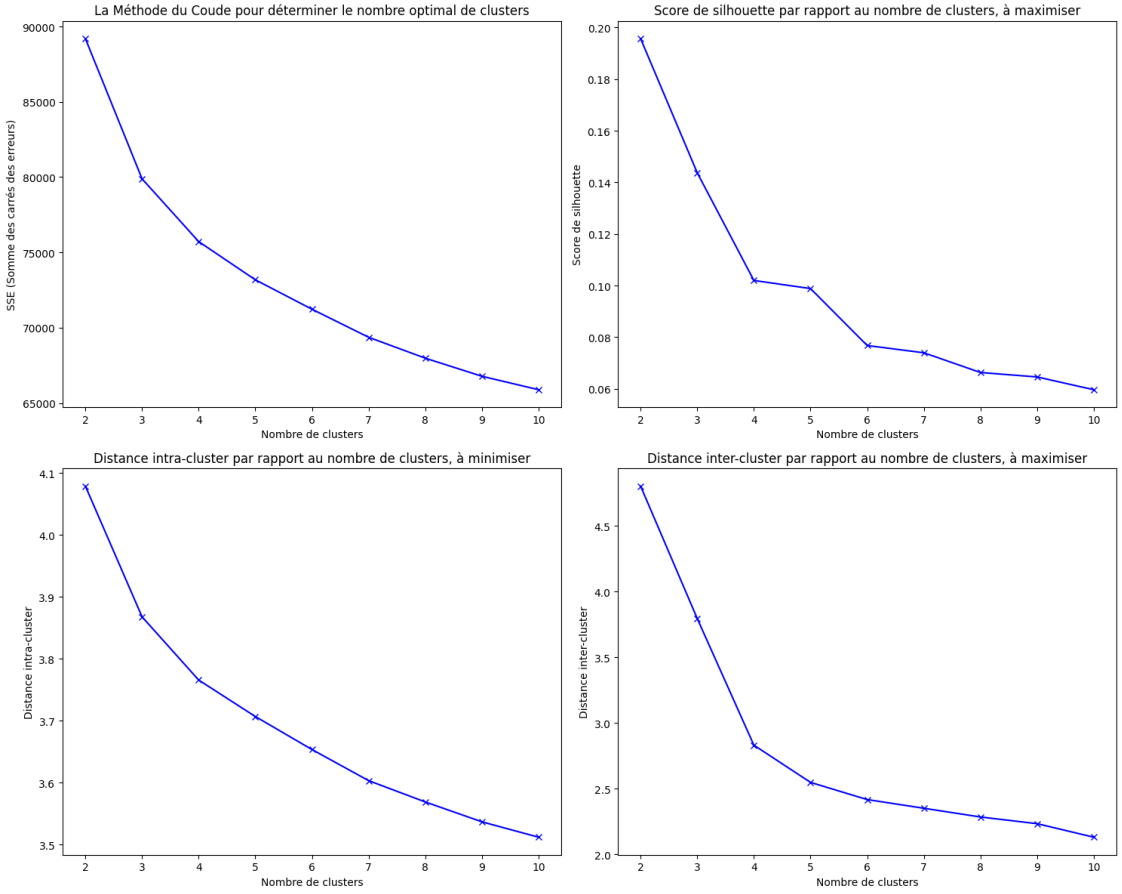
[7]:
```
plot_cluster_metrics(df_usage.iloc[:, 3:])
```



Le nombre de cluster optimal peut être estime à 4.

For attitude questions

[8]:
```
plot_cluster_metrics(df_attitude.iloc[:, 3:])
```



Le nombre de cluster optimal peut être estime à 4 aussi.

1.2 Usage Clustering

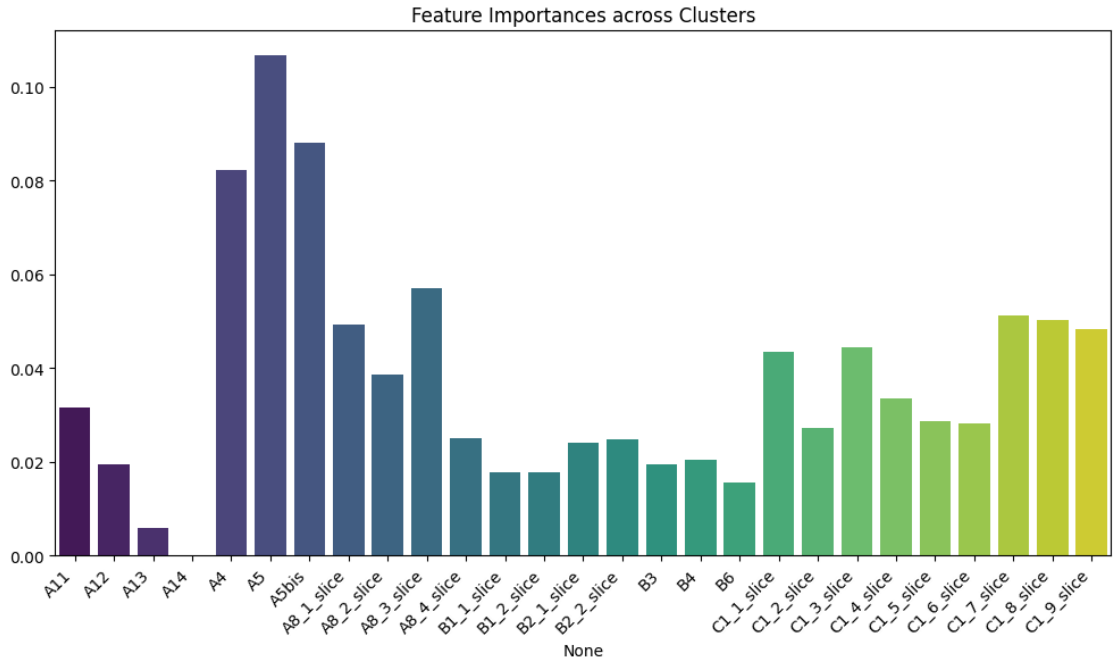
[9]:
```
kmeans_usage = make_clusters(df_usage, 5)
```

1.3 Explication des clusters

-

1.4 Importance des features global pour la decision

[10]:
```
feature_importance(df_usage)
```



[10]:
```
array([0.03167527, 0.01944901, 0.00583454, 0.        , 0.08231449,
       0.10667379, 0.08812257, 0.049358  , 0.03855753, 0.05697959,
       0.02504394, 0.0178533 , 0.01782854, 0.02412305, 0.02479063,
       0.0195966 , 0.02054612, 0.01569058, 0.04353875, 0.0273302 ,
       0.0443834 , 0.03364591, 0.02862541, 0.02832406, 0.05120617,
       0.05024574, 0.04826281])
```

-

1.5 Importance des features par cluster

[11]:
```
features_per_cluster(df_usage.iloc[:, 3:-1], kmeans_usage, 5)
```

```
Most Important Features by Cluster:
Cluster 0: ['A4', 'A5bis', 'A11', 'C1_1_slice', 'C1_9_slice']
Cluster 1: ['C1_6_slice', 'C1_4_slice', 'C1_7_slice', 'C1_3_slice',
'C1_5_slice']
Cluster 2: ['A8_3_slice', 'A8_1_slice', 'A8_2_slice', 'A8_4_slice', 'A5']
Cluster 3: ['A4', 'A11', 'A5bis', 'A5', 'A12']
Cluster 4: ['C1_1_slice', 'B2_2_slice', 'B2_1_slice', 'C1_9_slice',
'C1_2_slice']
```

-

1.6 Analyse des features

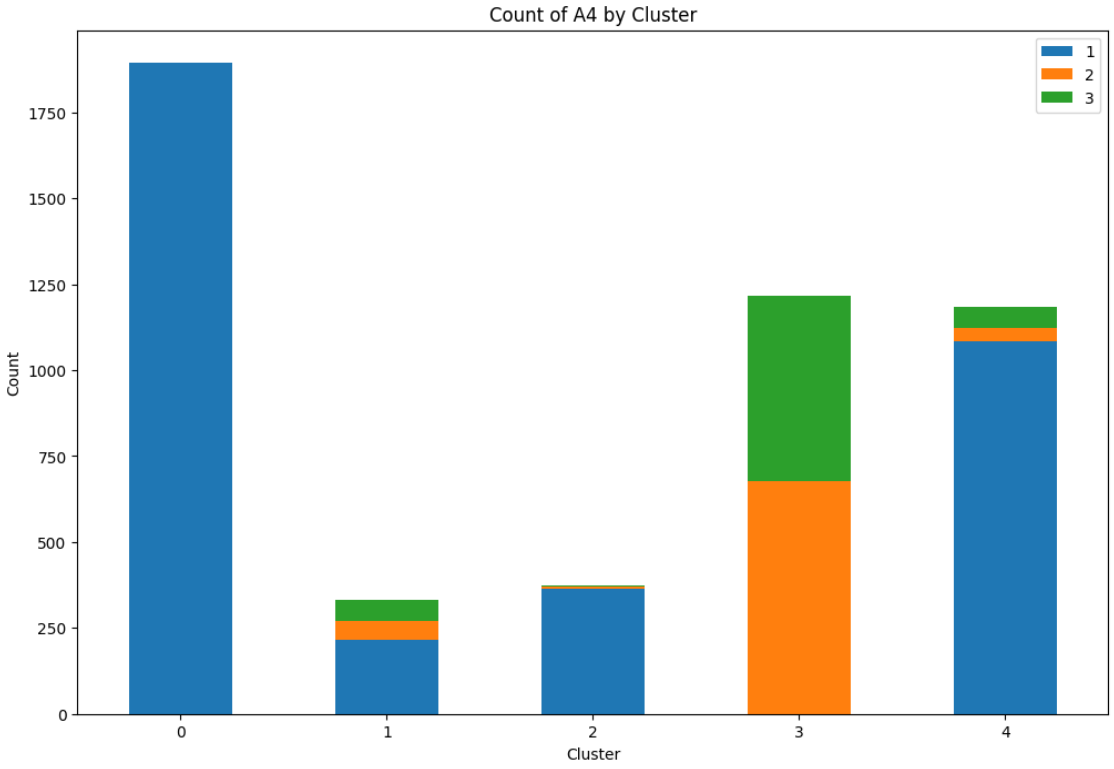
-

1.6.1 Jardin, terasse et balcon

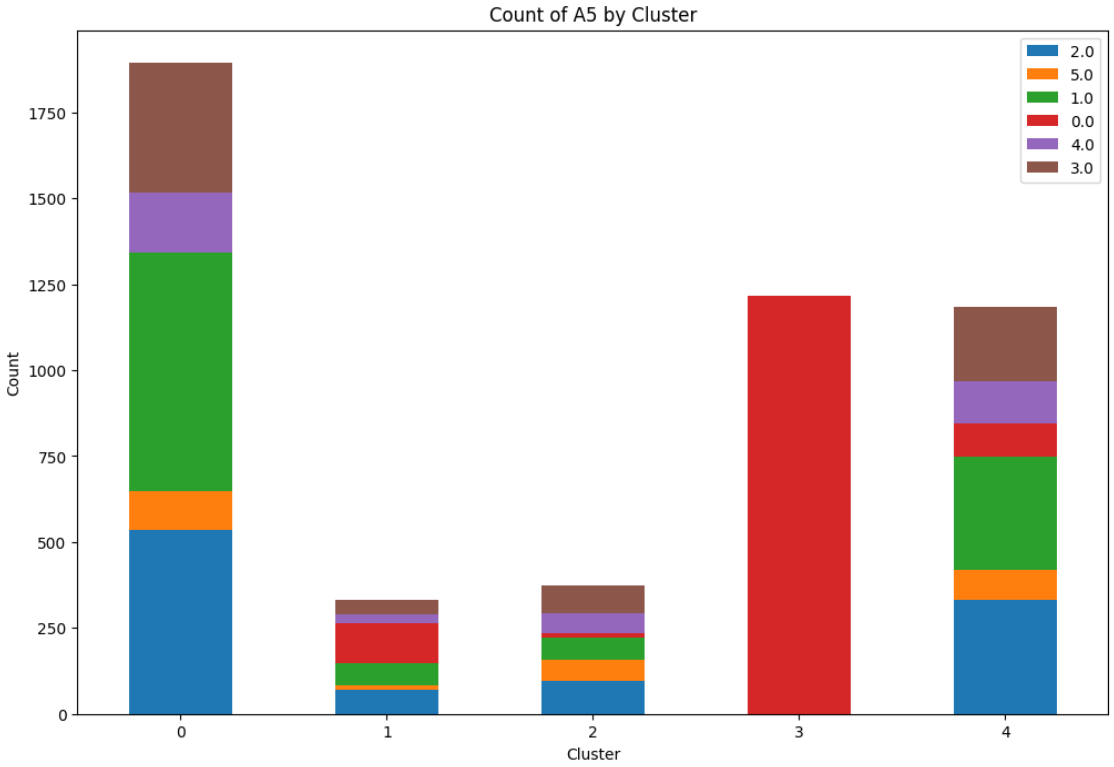
Nous remarquons l'importance des questions (features) `A4`, `A5` et `A5bis`. Ces questions sont: - `A4`: **Type d'espace** (reponses possibles: Jardin, Balcon, Terrasse) - `A5`: **Taille du jardin** (reponses possibles: Tranches de taille croissante) - `A5bis`: **Taille de la terrasse/balcon** (reponses possibles: Tranches de taille croissante)

On peut facilement remarquer que **le cluster 3 consiste d'individu n'ayant pas de jardin**. Nous pouvons aussi note que le cluster 0 et 2 ne comporte quasiment personne ayant une terasse ou un balcon.

[12]:
```
compare_cluster_answers(df_usage, 'cluster', 'A4')
```

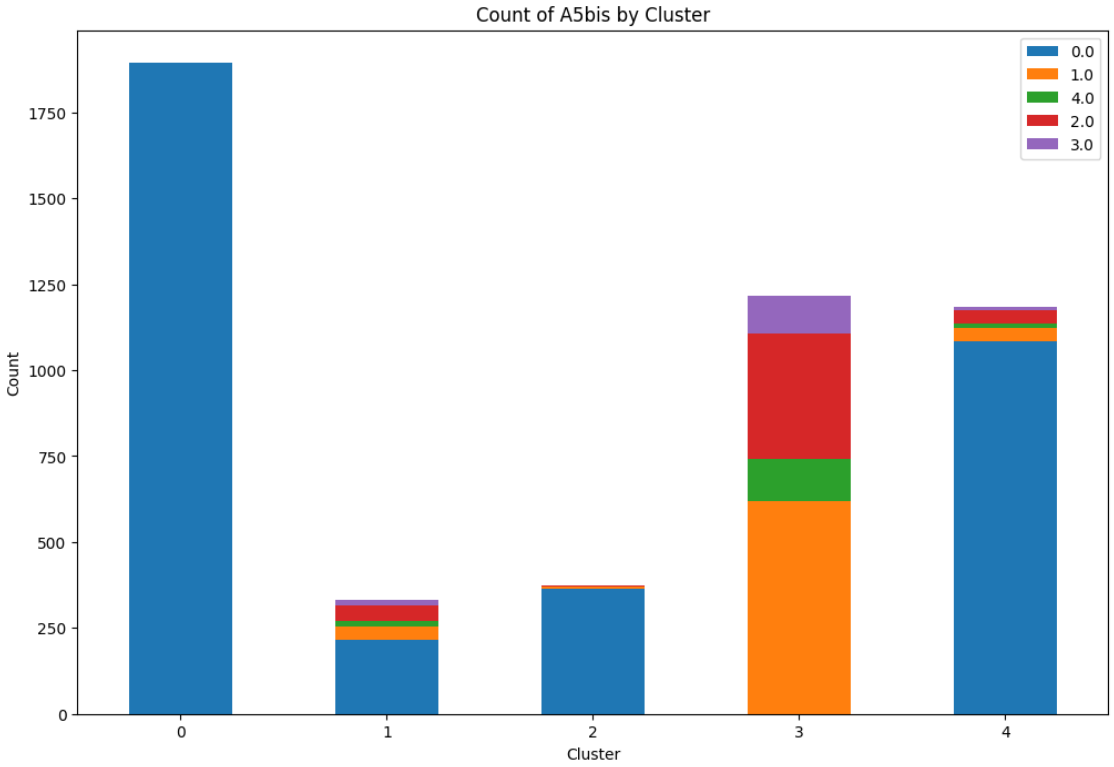


[13]:
```
compare_cluster_answers(df_usage, 'cluster', 'A5')
```



Ceci est confirme par la moyenne de la taille des jardin pour chaque cluster, le cluster 3 ayant une moyenne de 0.
Cependant cela ne donne pas davantage d'information sur les autres clusters.
Ainsi, à part pour le cluster 3, **la taille du jardin n'est pas impactant dans le choix clusters**.

[14]:
```
compare_cluster_answers(df_usage, 'cluster', 'A5bis')
```



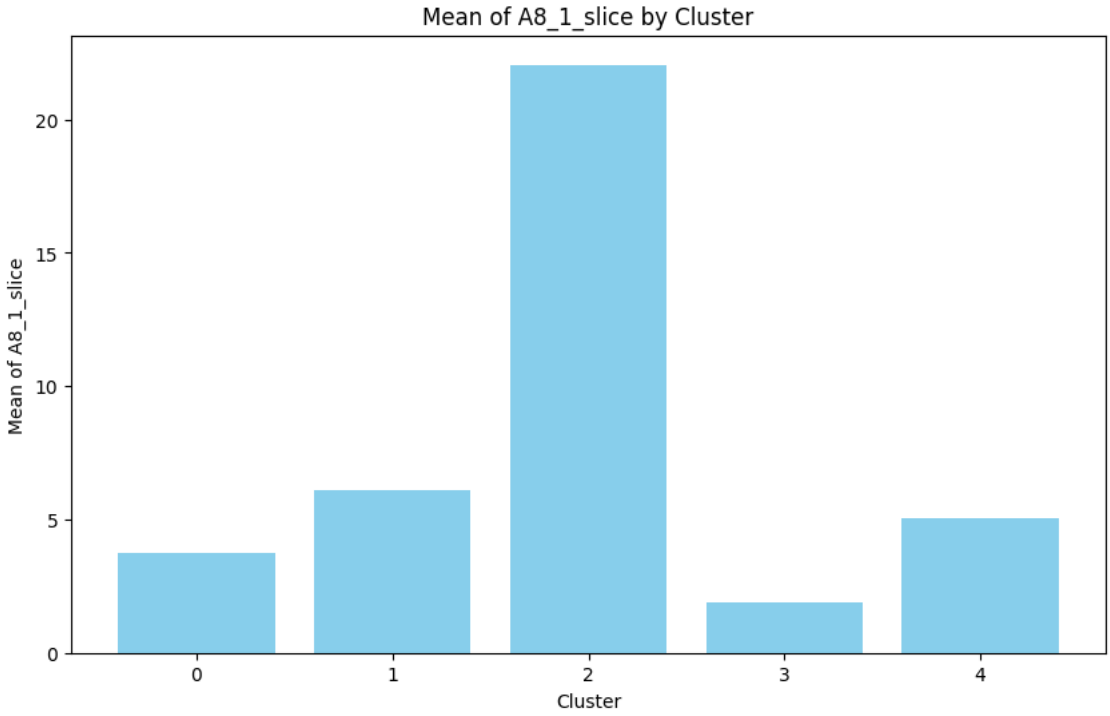
Le complementaire. La majorite des terasses ou balcons ont une taille entre 11 et 30 m2.

1.6.2 Temps passe à l'entretien de l'espace exterieur

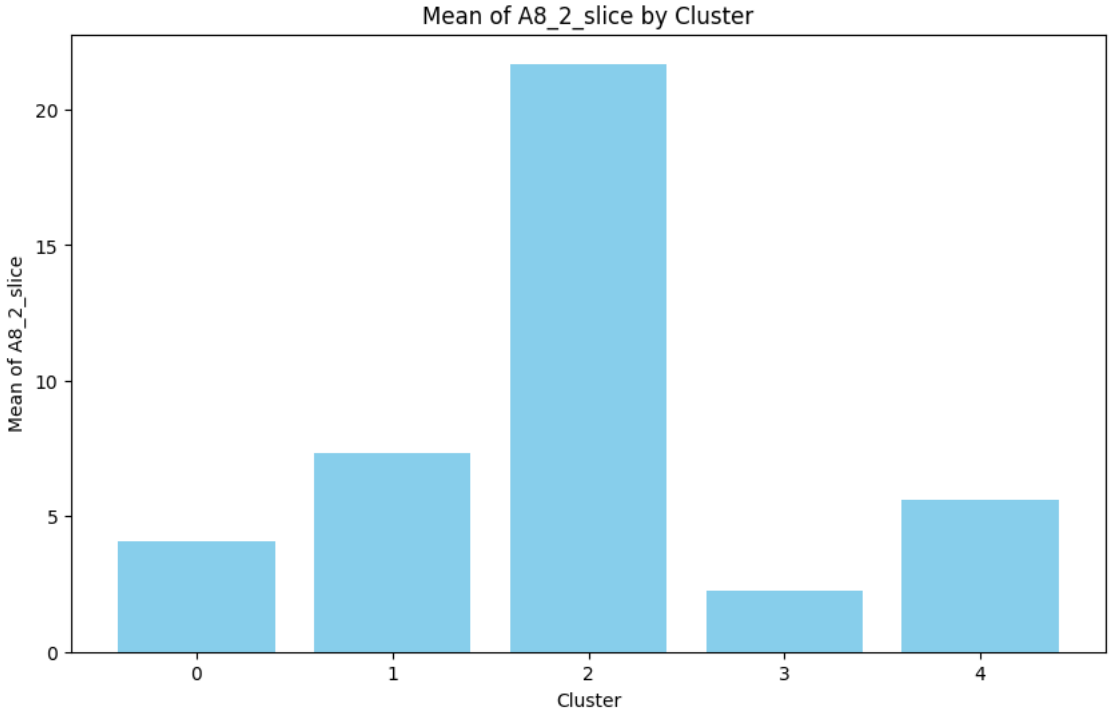
Nous avons une suite de question a propos du temps passe à l'entretien de l'espace exterieur en heure: - `A8_1_slice`: en printemps - `A8_2_slice`: en ete - `A8_3_slice`: en automne - `A8_4_slice`: en hiver

Ces questions font parties des 5 features les plus importantes du cluster 2.

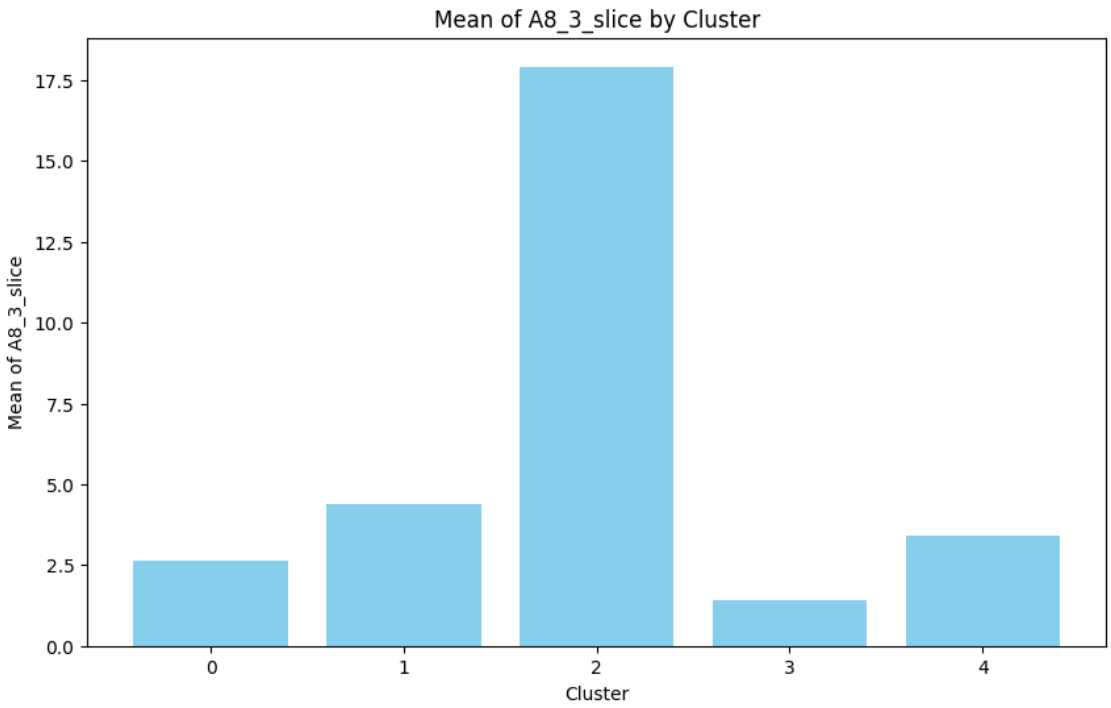
[15]:
```
compare_cluster_describe(df_usage, 'cluster', 'A8_1_slice', 'mean')
```



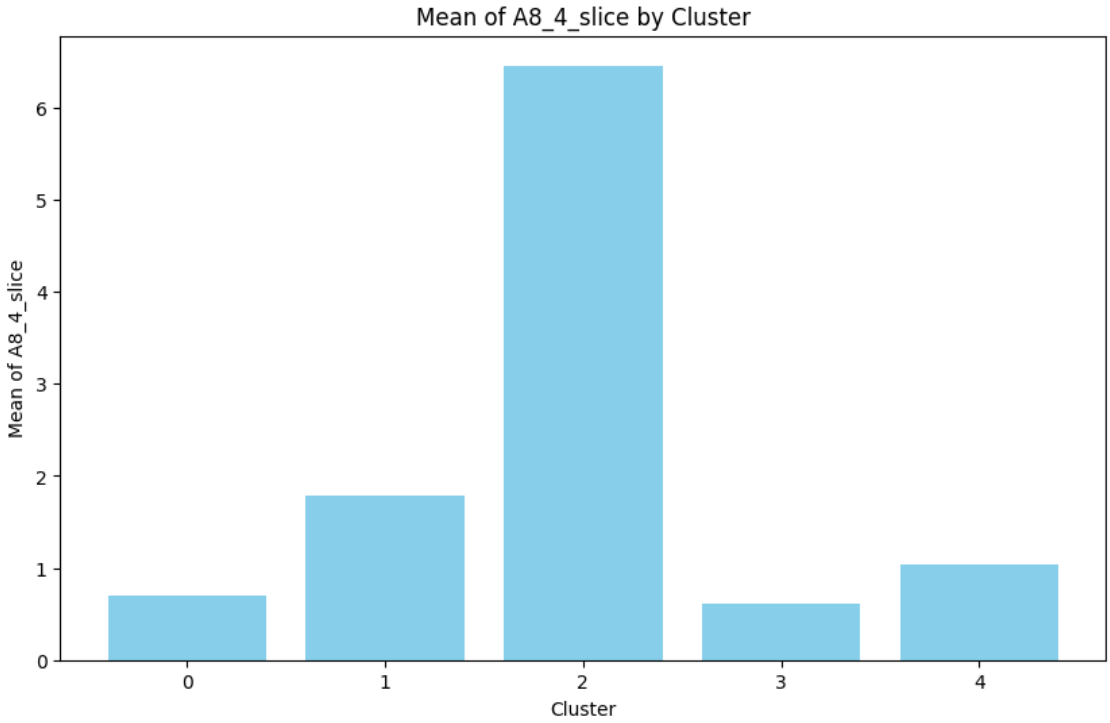
[16]:
```
compare_cluster_describe(df_usage, 'cluster', 'A8_2_slice', 'mean')
```



[17]:
```
compare_cluster_describe(df_usage, 'cluster', 'A8_3_slice', 'mean')
```



[18]:
```
compare_cluster_describe(df_usage, 'cluster', 'A8_4_slice', 'mean')
```

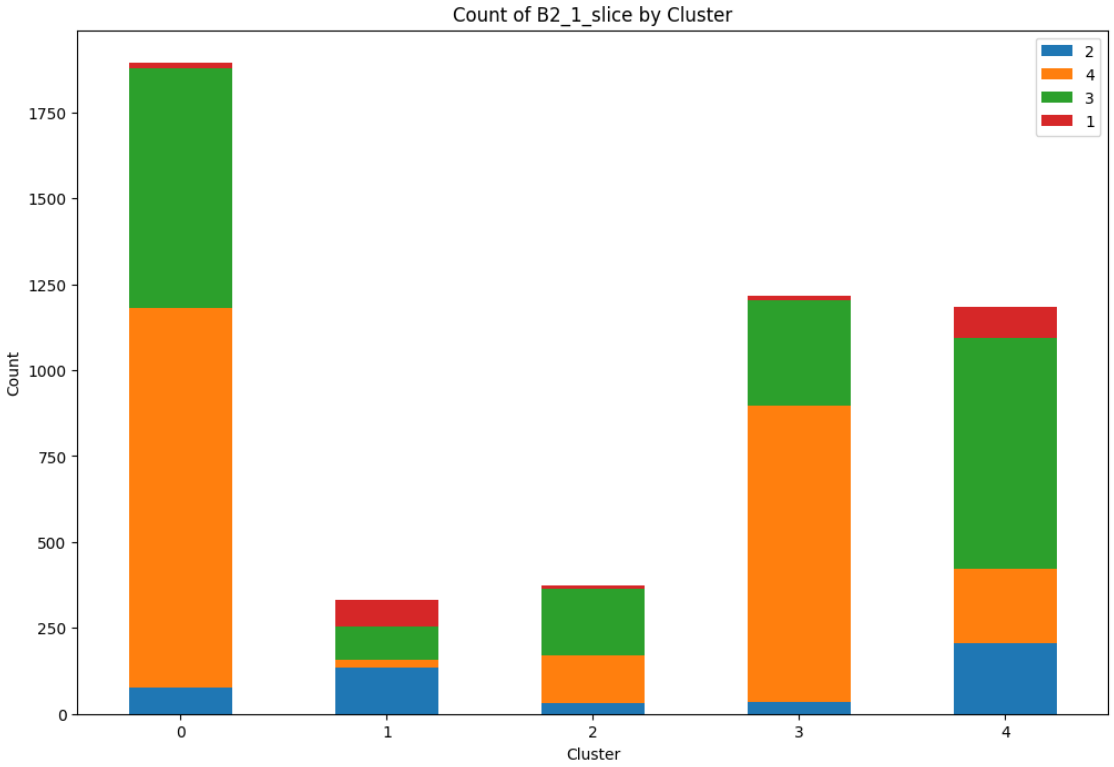


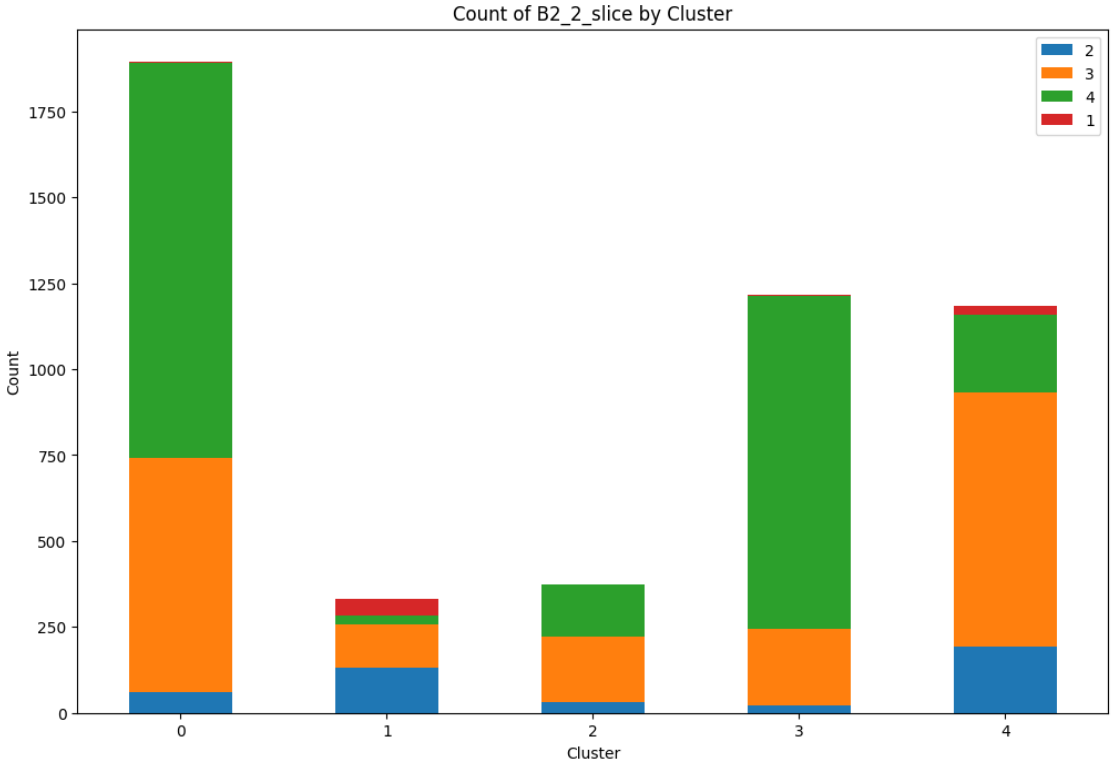
Le cluster 2 regroupe les individus qui consacrent beaucoup de temps à l'entretien de l'espace exterieur.
On peut egalement noter que les gens passent beaucoup moins de temps en hiver pour l'entretien.

-

1.6.3 Pret d'outil

[19]:
```
compare_cluster_answers(df_usage, 'cluster', 'B2_1_slice')
compare_cluster_answers(df_usage, 'cluster', 'B2_2_slice')
```


Count of B2_1_slice by Cluster
2
4
3
1
Count
Cluster
0
1
2
3
4
0
250
500
750
1000
1250
1500
1750


Count of B2_2_slice by Cluster
2
3
4
1
Count
Cluster
0
1
2
3
4
0
250
500
750
1000
1250
1500
1750

Les individus du cluster 1 semble le plus preter leurs outils que les autres clusters.

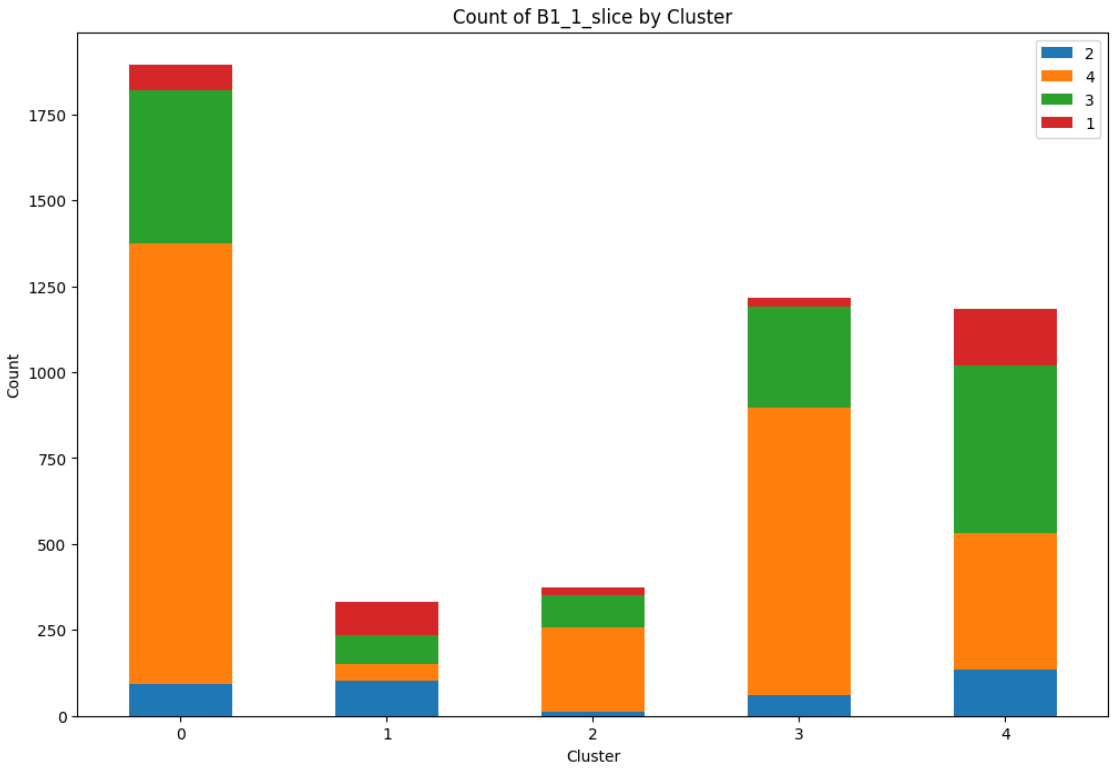
La majorite des individus du cluster 0 et 3 ne prete jamais leurs outils.

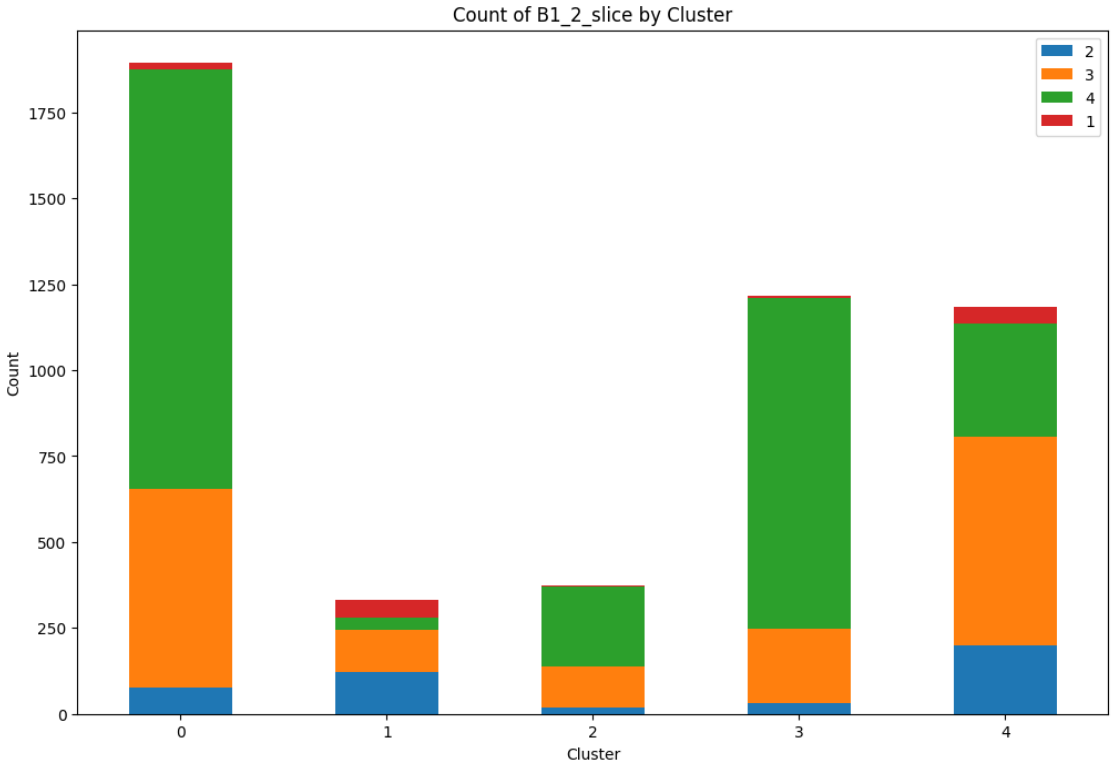
La cluster 2 et 4 prete occasionnelement leurs outils.

•

1.6.4 Emprunt d'outil

[20]:
```
compare_cluster_answers(df_usage, 'cluster', 'B1_1_slice')
compare_cluster_answers(df_usage, 'cluster', 'B1_2_slice')
```





Nous observons à peu pres le meme motif que pour le pret d'outil. Le cluster 4 en pourcentage emprunte plus d'outil que le cluster 2.

Le cluster 2 ayant le plus d'implication dans le pret/emprunt d'outil, il est tout de meme interessant d'bserver **une hausse d'implication chez le cluster 4 par rapport aux restes**.

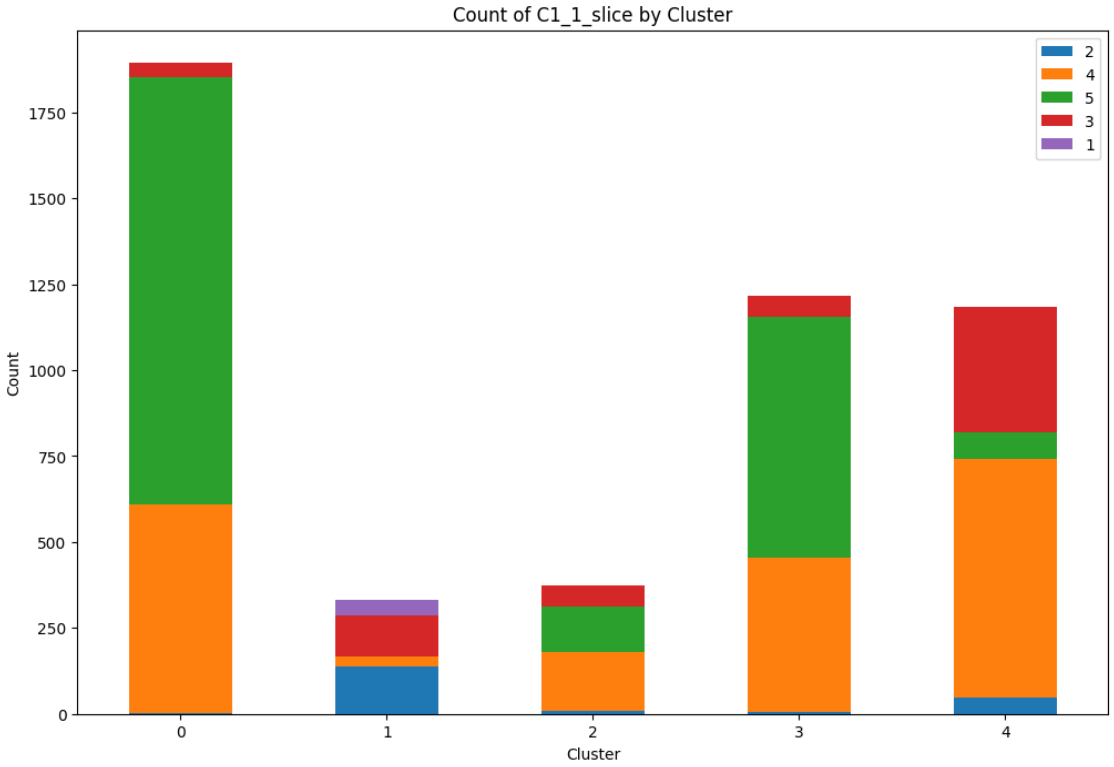
Le cluster 0 et 3 sont ceux qui sont le moins impliques.

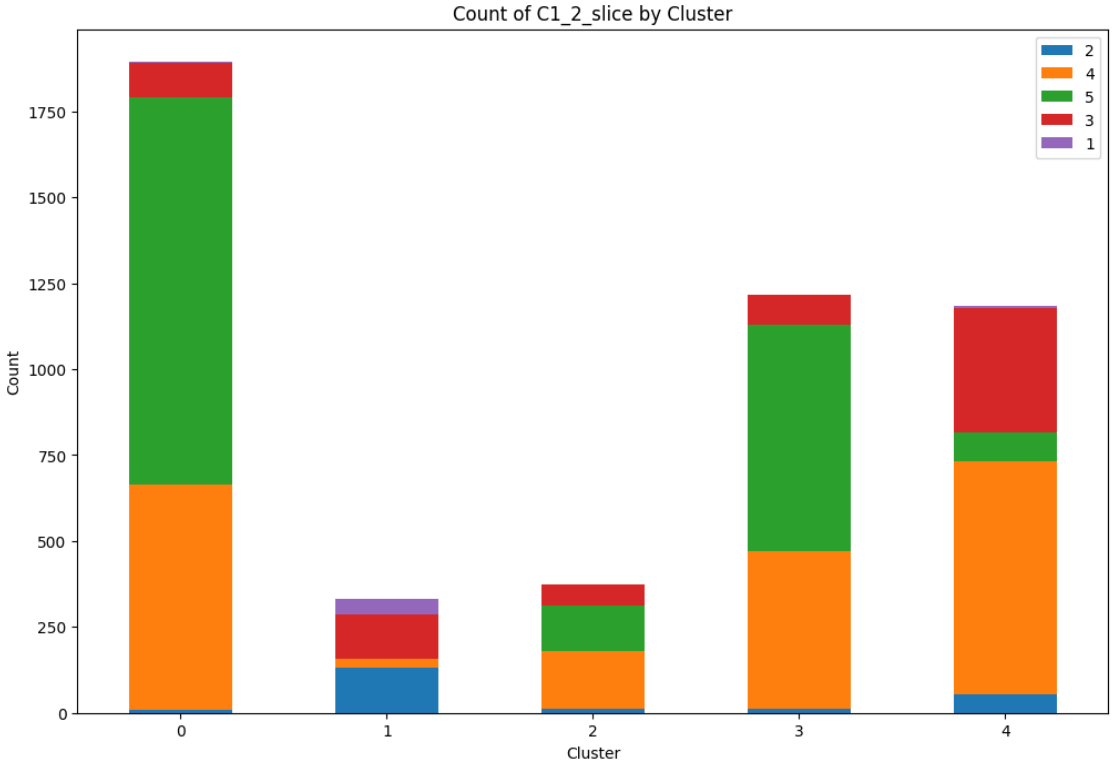
-

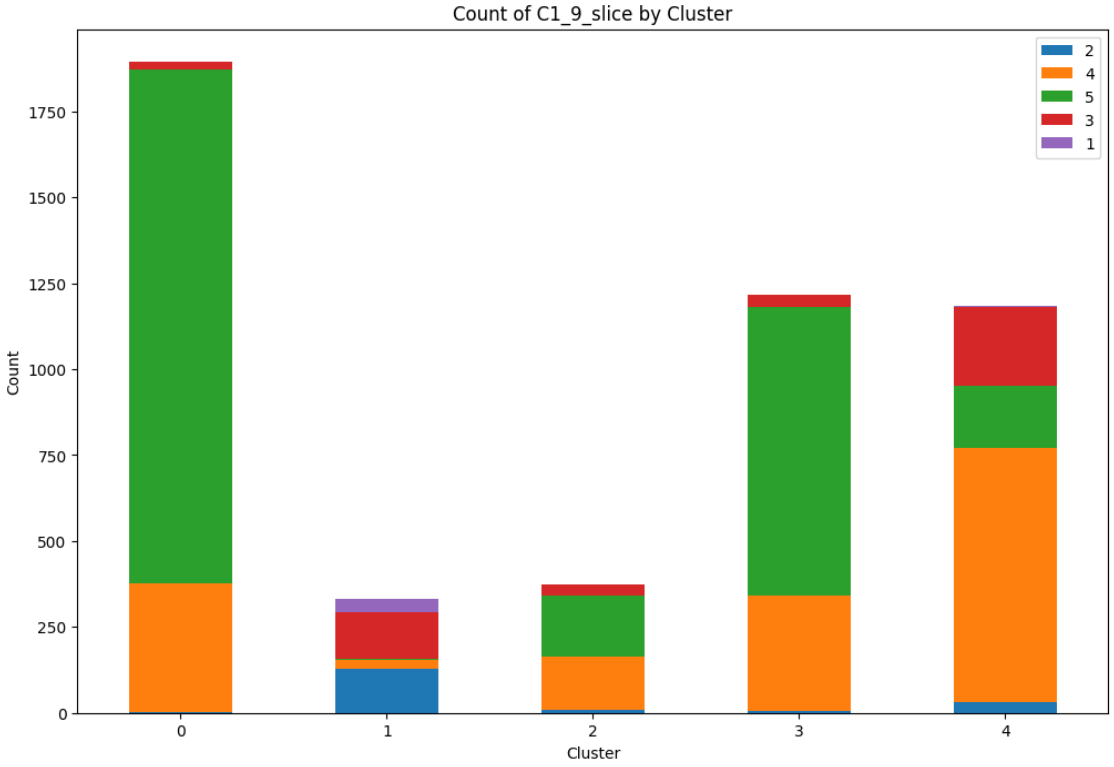
1.6.5 Consultation des sites dedies au jardinage

Les questions sur ce theme ont quasiment toutes une feature importance legerement plus eleves que la moyenne.
On retrouve egalement beaucoup de ces questions dans les features les plus importantes pour les clusters 0, 1 et 4.

[21]:
```
compare_cluster_answers(df_usage, 'cluster', 'C1_1_slice')
compare_cluster_answers(df_usage, 'cluster', 'C1_2_slice')
compare_cluster_answers(df_usage, 'cluster', 'C1_9_slice')
```


Count of C1_1_slice by Cluster
2
4
5
3
1
Count
0
250
500
750
1000
1250
1500
1750
0
1
2
3
4
Cluster


Count of C1_2_slice by Cluster
2
4
5
3
1
Count
0
250
500
750
1000
1250
1500
1750
0
1
2
3
4
Cluster



Encore une fois, le cluster 1 semble etre tres actifs, passant le plus de temps sur les reseaux et ensuite c'est le cluster 4 qui passe legerement plus de temps que les autres clusters.

On peut donc en deduire **un engagement fort du cluster 1, modere pour le cluster 4 et tres faible pour les autres clusters**.

1.6.6 Bilan

- Le cluster 0: rassemble les individus ayant juste un jardin et passe tres peu de temps a l'entretien de leur jardin et sur les reseaux. **C'est le groupe le moins impliques**.
- Le cluster 1: rassemble les individus les plus actifs sur les reseaux et meme pour emprunter/preter des outils ou autres. **C'est le groupe le plus interesse par des offres de jardinage**.
- Le cluster 2: rassemble les individus qui passent le plus de temps a entretenir leur espace exterieur. On peut noter que c'est gens passent beaucoup de temps a entretenir leur espaces exterieurs mais n'essaye pas de beneficier des offres de reseaux ou d'echange d'outils. **C'est le groupe d'individu qui ont potentiellement le plus besoin d'offre pour optimiser leur temps d'entretien**.
- Le cluster 3: rassemble les individus n'ayant pas de jardin. **C'est le groupe à cibler pour des offres concernant les balcons et les terasses**.
- Le cluster 4: rassemble les individus moderement impliques sur les reseaux ou pour l'emprunt/pret d'outil. **C'est le groupe potentiellement susceptible d'etre interesses**

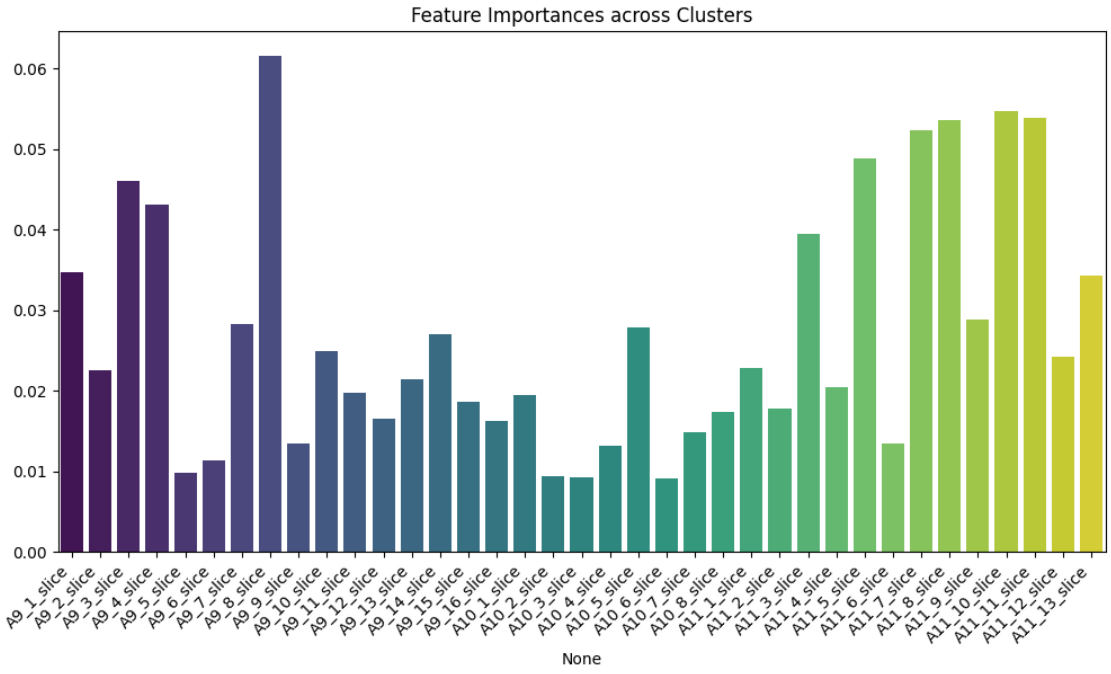
par des offres de jardinage.

1.7 Attitude Clustering

[22]:
```
kmeans_attitude = make_clusters(df_attitude, 4)
```

1.7.1 Importance des features global

[23]:
```
feature_importance(df_attitude)
```



[23]:
```
array([0.03466218, 0.0225467 , 0.04607541, 0.04309371, 0.00983648,
       0.01132001, 0.02821309, 0.06156089, 0.01342006, 0.02485523,
       0.01973481, 0.01647848, 0.02138078, 0.02700149, 0.01864783,
       0.01625619, 0.01944335, 0.00937925, 0.00921679, 0.01317736,
       0.02784215, 0.00912714, 0.014861  , 0.01737009, 0.02275722,
       0.01774229, 0.03940089, 0.02037998, 0.04885382, 0.01347572,
       0.05236134, 0.05365114, 0.02883536, 0.05471404, 0.05382281,
       0.02414846, 0.03435645])
```

1.7.2 Importance des features par cluster

[24]:
```
features_per_cluster(df_attitude.iloc[:, 3:-1], kmeans_attitude, 4)
```

```
Most Important Features by Cluster:
Cluster 0: ['A9_3_slice', 'A9_8_slice', 'A9_4_slice', 'A9_2_slice',
```

```
'A9_7_slice']
Cluster 1: ['A11_8_slice', 'A11_5_slice', 'A11_7_slice', 'A11_3_slice',
'A11_10_slice']
Cluster 2: ['A9_8_slice', 'A9_3_slice', 'A9_4_slice', 'A9_10_slice',
'A9_2_slice']
Cluster 3: ['A11_5_slice', 'A11_7_slice', 'A11_10_slice', 'A11_3_slice',
'A11_1_slice']
```

-

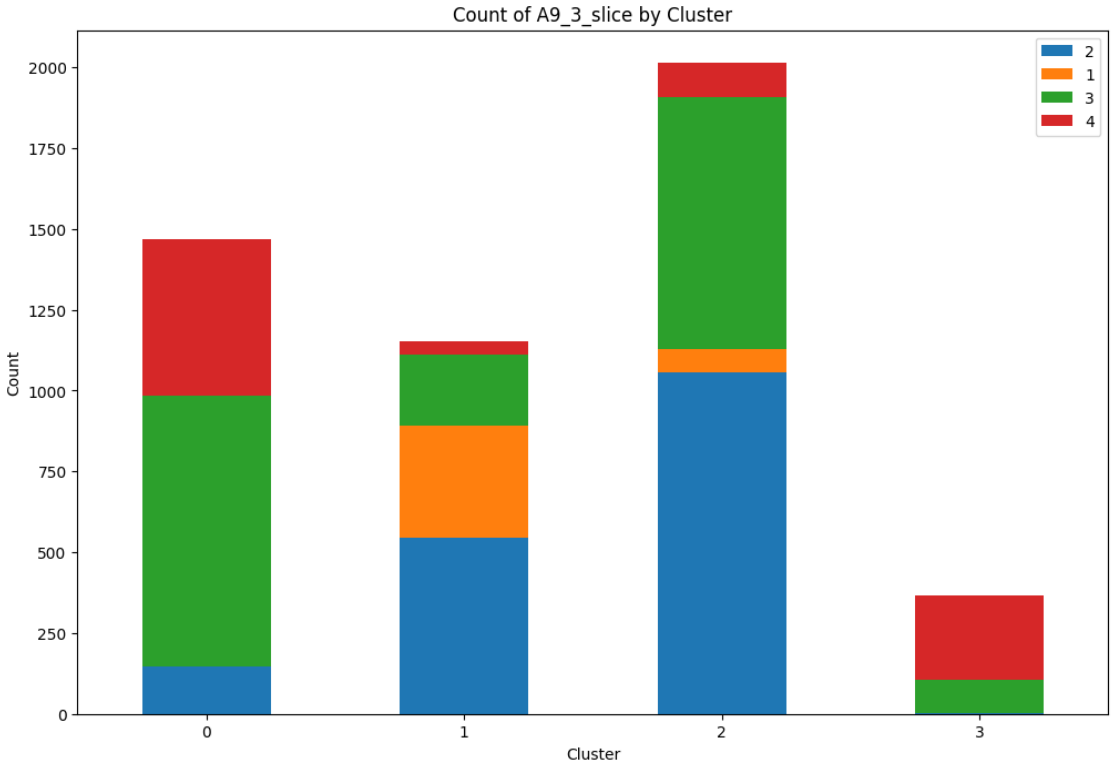
1.8 Analyse des features

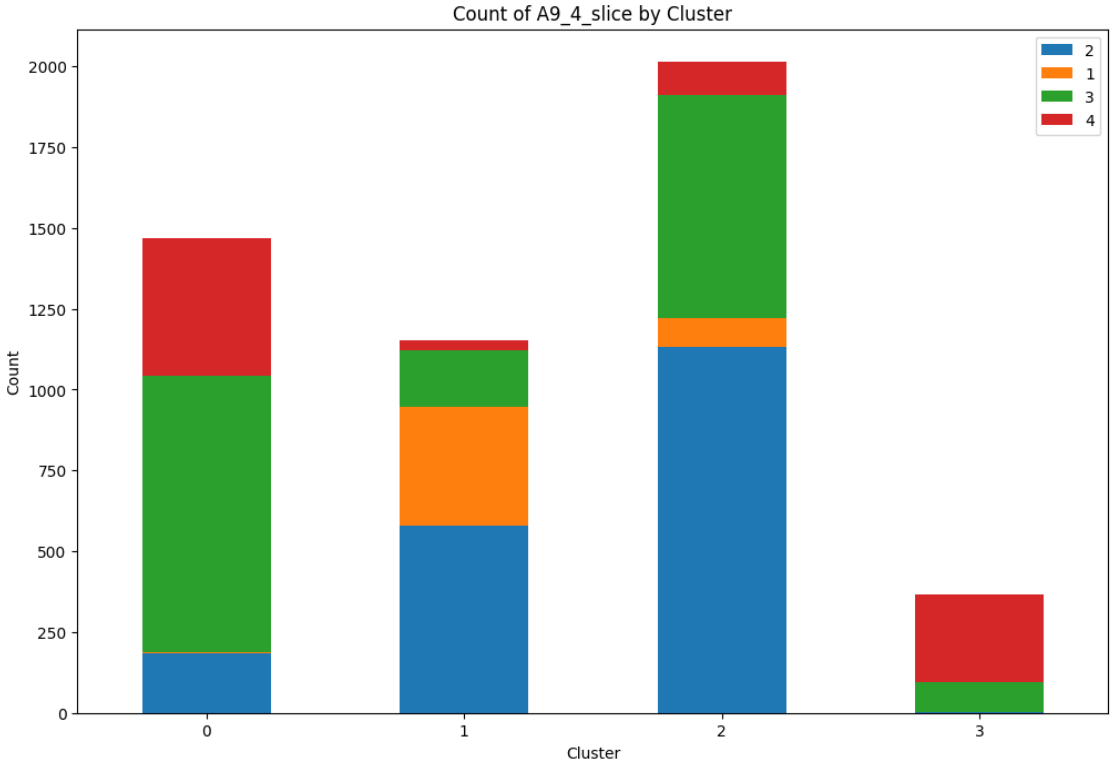
-

1.8.1 Interet envers l'amenagement

Les questions de type `A9` donne une idee de l'interet porter par les individus sur leur espace exterieur et de leur amenagement. Il y'a un total de 16 questions, nous fournissant donc beaucoup d'informations.
Les questions `A9_3_slice`, `A9_4_slice` et `A9_8_slice` sont trois questions ayant la plus grande feature importances parmi toutes les autres features, analysant les de plus pres. - `A9_3_slice`: L'individu recherche souvent des informations sur l'aménagement des espaces extérieurs - `A9_4_slice`: L'individu recherche souvent des informations sur l'entretien des espaces extérieurs - `A9_8_slice`: L'individu privilégie les produits éco-responsables dans l'aménagement et l'entretien des espaces extérieurs

[25]:
```
compare_cluster_answers(df_attitude, 'cluster', 'A9_3_slice')
compare_cluster_answers(df_attitude, 'cluster', 'A9_4_slice')
```


Count of A9_3_slice by Cluster
Count
Cluster
2
1
3
4
0
250
500
750
1000
1250
1500
1750
2000
0
1
2
3

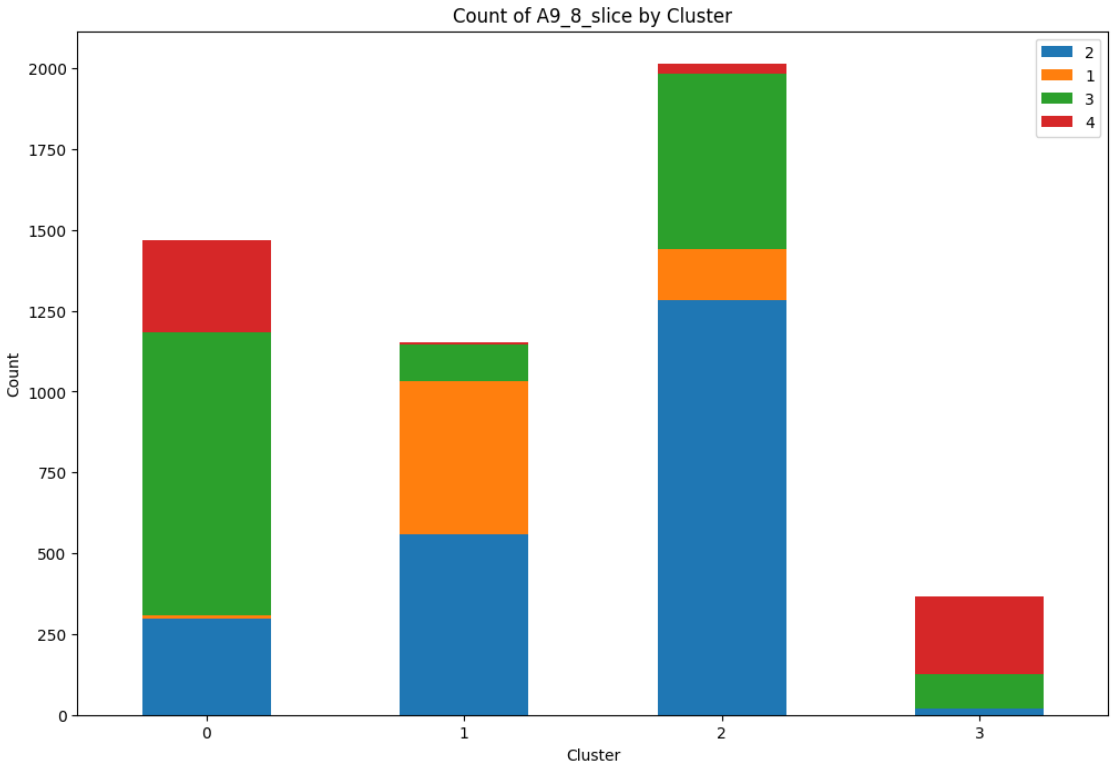

Count of A9_4_slice by Cluster
Count
Cluster
2
1
3
4
0
250
500
750
1000
1250
1500
1750
2000
0
1
2
3

On remarque que le cluster 1 rassemble les gens les plus interesses par l'amenagement exterieur que les autres. En effet, il effectue le plus de recherche sur l'amenagement et l'entretien.

Au contraire, le cluster 3 ne semble pas interesse.

Le cluster 2 semble plus interesse par l'amenagement que le cluster 0.

[26]:
```
compare_cluster_answers(df_attitude, 'cluster', 'A9_8_slice')
```



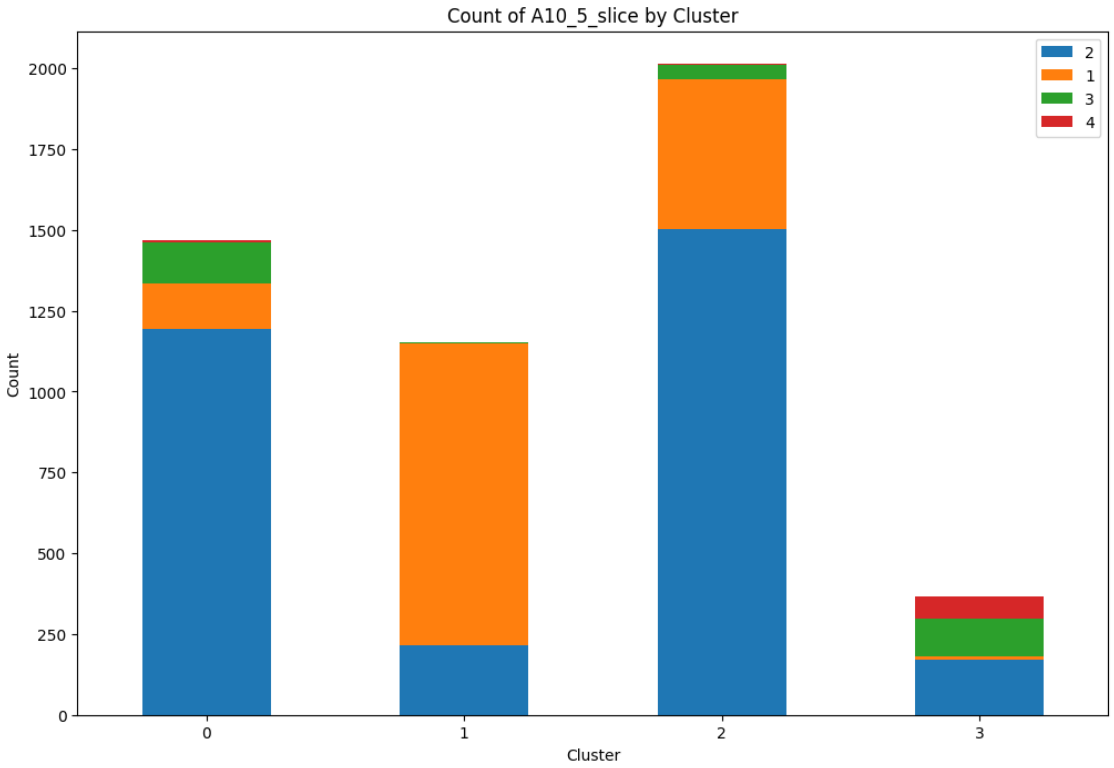
Similairement, nous observons que le meme pattern pour l'investissement dans l'amenagement.

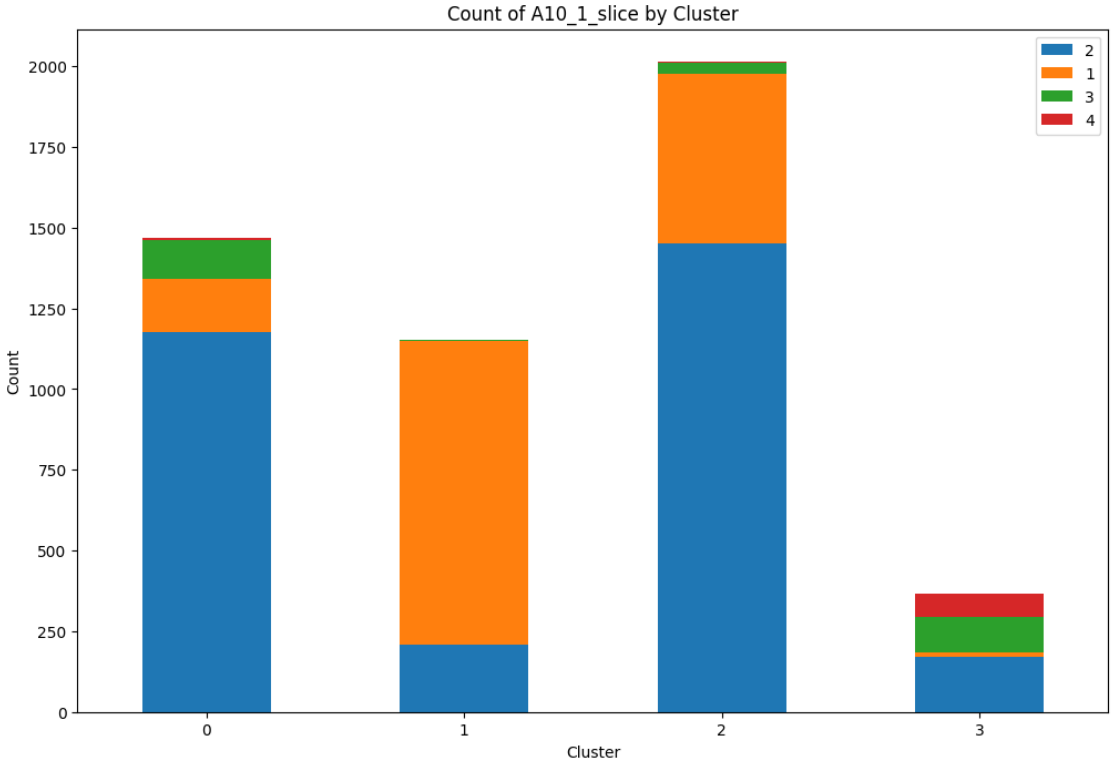
-

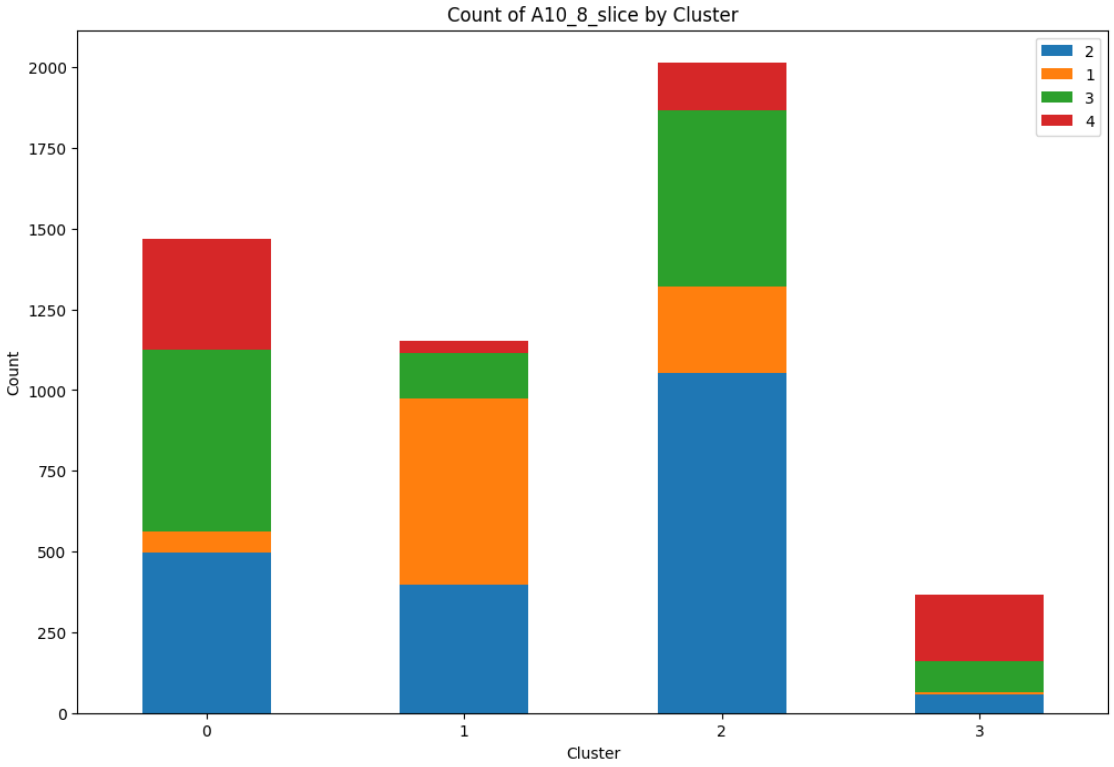
1.8.2 Vision sur l'utilite des espaces exterieurs

Les questions de la catégorie `A10` nous fournissent des informations sur les opinions des individus en ce qui concerne l'utilité de leurs espaces extérieurs. 10 11 8 7 5 Les questions avec le plus d'importance dans cette categorie sont les suivantes: - `A10_5_slice`: Les espaces extérieurs sont des vecteurs de bonheur et de plaisirs - `A10_1_slice`: Les espaces extérieurs permettent de rester en contact avec la nature - `A10_8_slice`: Les espaces extérieurs vous permettent de faire des économies grâce à la culture de fruits et légumes

[27]:
```
compare_cluster_answers(df_attitude, 'cluster', 'A10_5_slice')
compare_cluster_answers(df_attitude, 'cluster', 'A10_1_slice')
compare_cluster_answers(df_attitude, 'cluster', 'A10_8_slice')
```







Nous remarquons également que le cluster 1 et 3 sont les extremes.
Le cluster 2 semble plus se rapprocher du 1 et le cluster 0 avec le cluster 3.

-

1.8.3 Perspective émotionnelle des espaces exterieurs

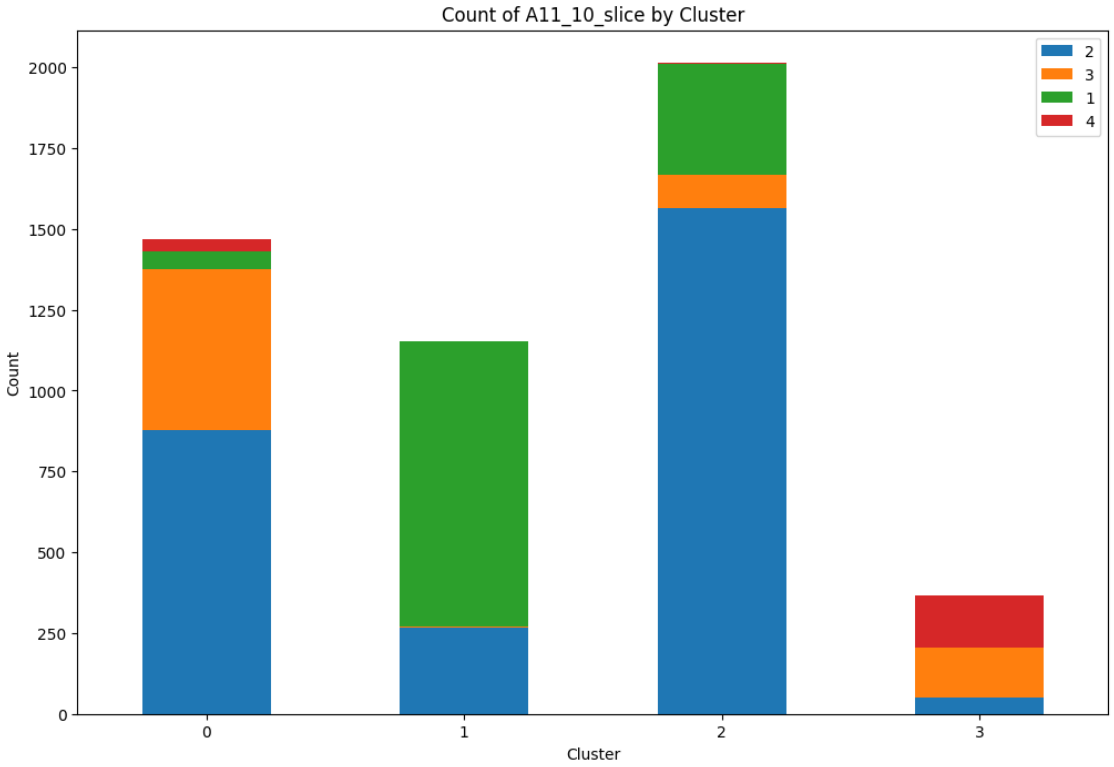
Les questions de la catégorie `A11` nous fournissent des informations sur les opinions des individus en ce qui concerne leur perspaces extérieurs.

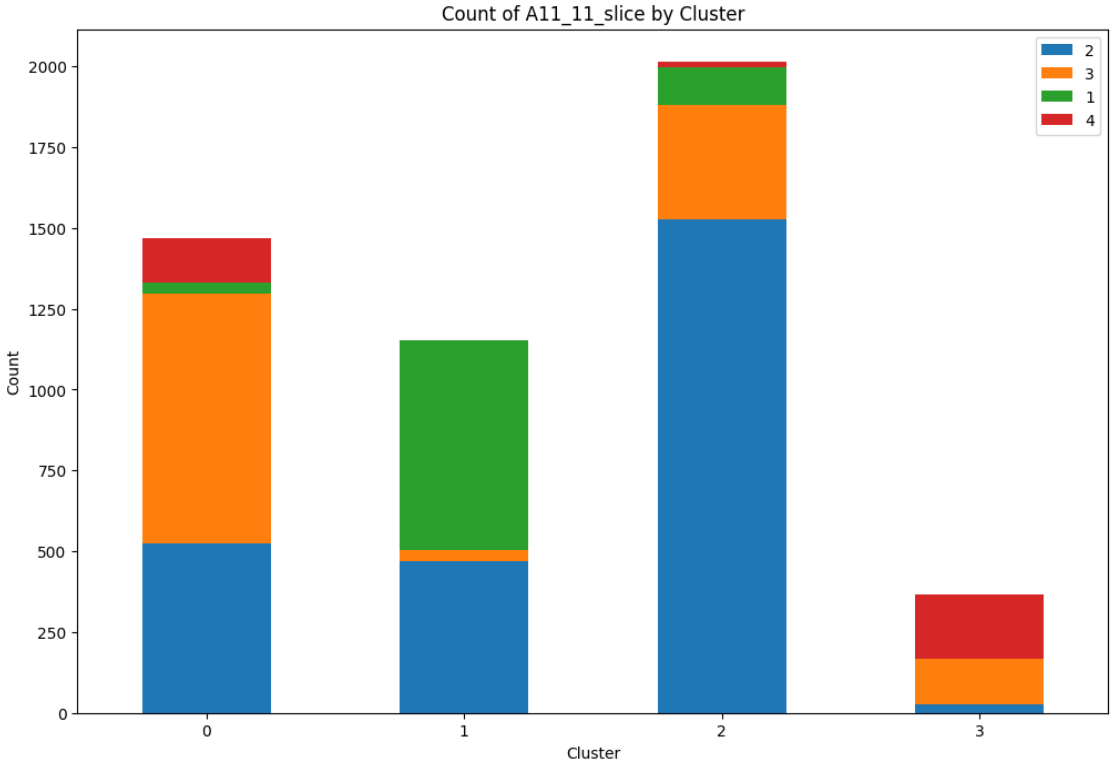
Les questions avec le plus d'importance dans cette categorie sont les suivantes:

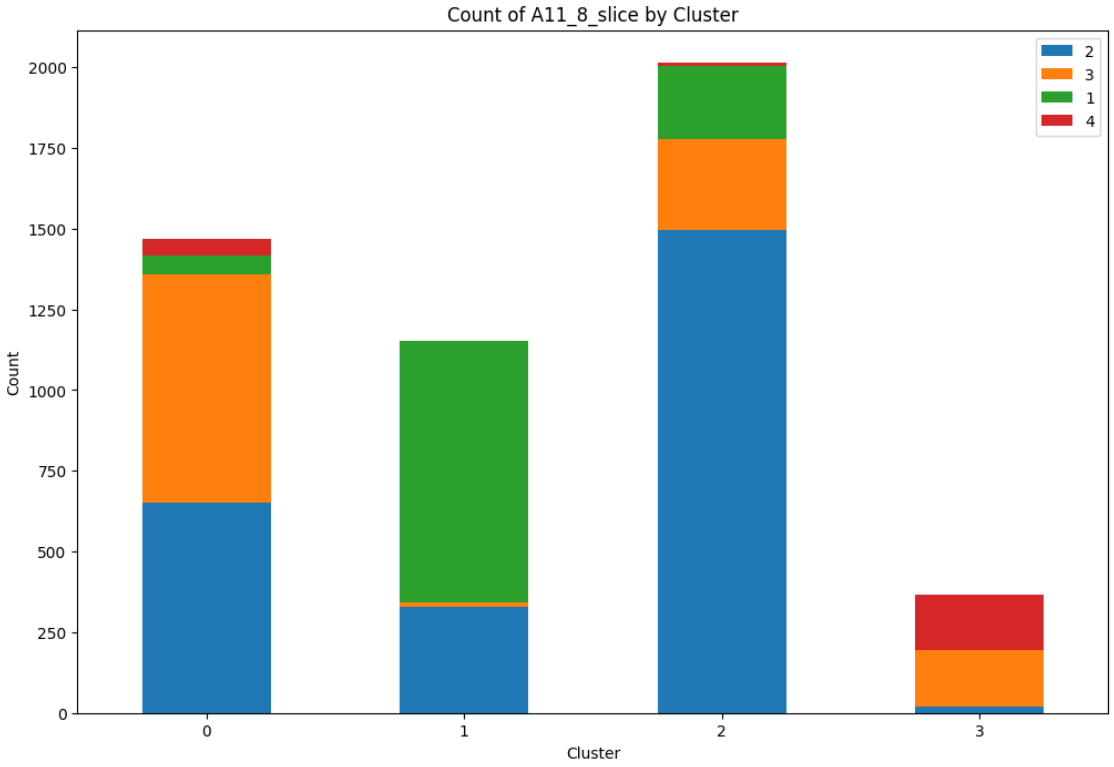
Les questions avec le plus d'importance dans cette categorie sont les suivantes: - `A11_10_slice`: Un moyen de réaliser jusqu'au bout quelque chose de vos propres mains - `A11_11_slice`: Un moyen de transmettre des connaissances, des pratiques qu'il est indispensable de perpétuer - `A11_8_slice`: Votre espace favori de loisirs et de liberté - `A11_7_slice`: Votre contact privilégié avec le vert, la nature - `A11_5_slice`: Un moyen de se resourcer, de refaire le plein d'énergie

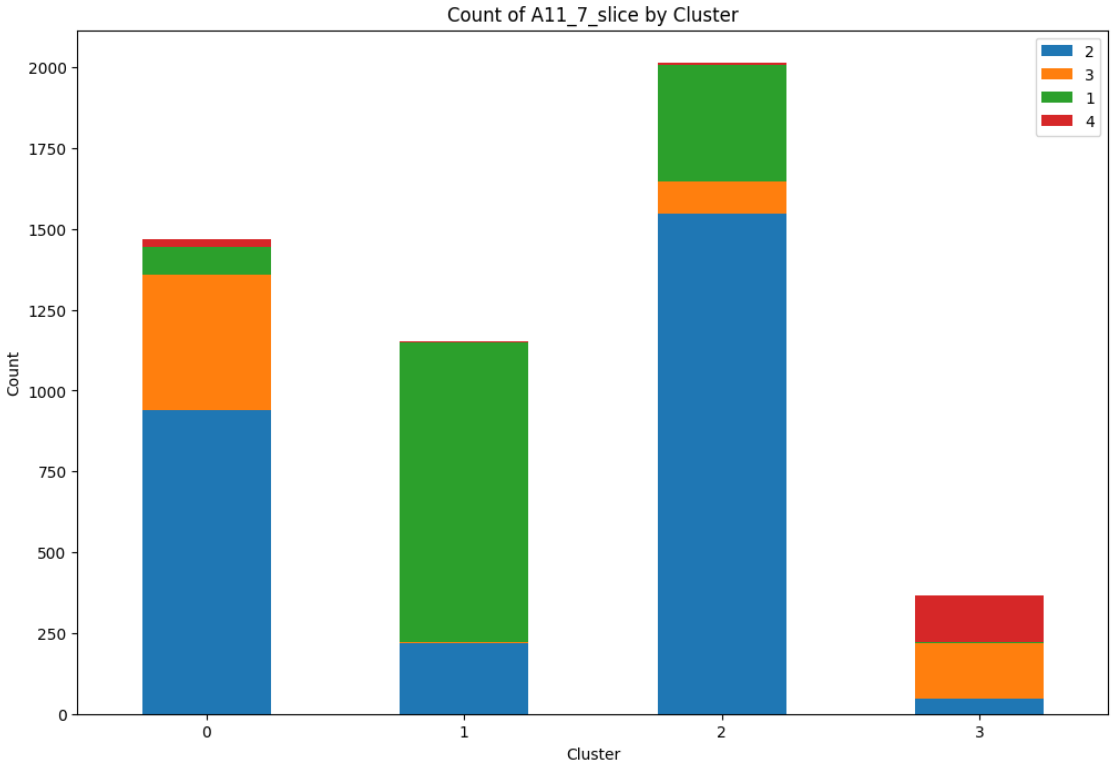
Ces questions ont tous une importance elevee.

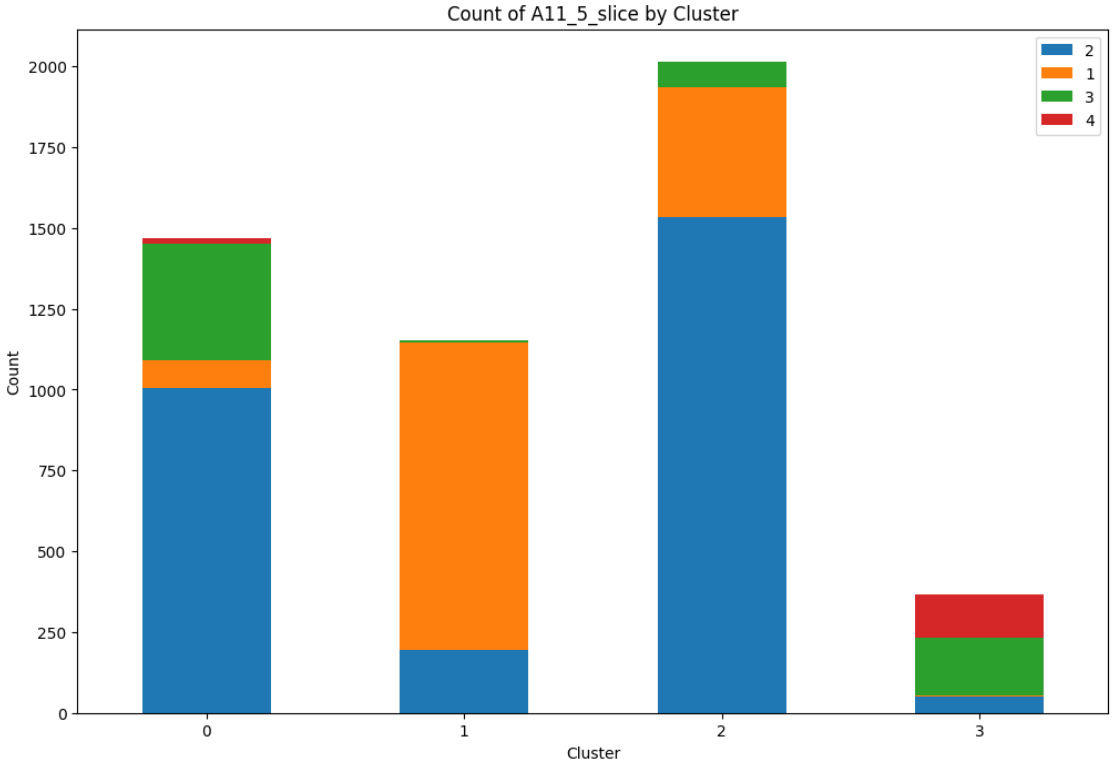
[28]:
```
compare_cluster_answers(df_attitude, 'cluster', 'A11_10_slice')
compare_cluster_answers(df_attitude, 'cluster', 'A11_11_slice')
compare_cluster_answers(df_attitude, 'cluster', 'A11_8_slice')
compare_cluster_answers(df_attitude, 'cluster', 'A11_7_slice')
compare_cluster_answers(df_attitude, 'cluster', 'A11_5_slice')
```






Count of A11_8_slice by Cluster
2
3
1
4
2000
1750
1500
1250
1000
750
500
250
0
Count
0
1
2
3
Cluster


Count of A11_7_slice by Cluster
Count
Cluster
0
1
2
3
0
250
500
750
1000
1250
1500
1750
2000
2
3
1
4


Count of A11_5_slice by Cluster
Count
Cluster
0
1
2
3
0
250
500
750
1000
1250
1500
1750
2000
2
1
3
4

Encore une fois, nous observons que le cluster 1 voit tres positivement leur espaces exterieurs et le cluster 3 pas du tout.

Le cluster 2 se rapproche du cluster 1 mais est plus modere dans ses reponses et le cluster 3 est neutre.

1.8.4 Bilan

- Le cluster 0: rassemble les individus ayant une perspective legerement positive voir neutre sur l'espace exterieur et son amenagement. **C'est le groupe sans opinions prononcees**.
- Le cluster 1: rassemble les individus qui portent un grand interet a leur espace exterieur, voit beaucoup d'utilite a avoir un espace exterieur et y sont tres attaches. **C'est le groupe le plus engage**.
- Le cluster 2: rassemble les individus qui portent un leger interet a leur espace exterieur. **C'est le groupe qui legerement engage**.
- Le cluster3: rassemble les individues qui ne porte aucun interet a leur espace exterieur. **C;est le groupe le moins impliques**.

1.9 Save the clusters

[29]:
```
df_codes["cluster_usage"] = df_usage["cluster"]
df_codes["cluster_attitude"] = df_attitude["cluster"]
```

[30]:
```
df_codes.head()
```

[30]:

	cle	Respondent_ID	weight	A11	A12	A13	A14	A4	A5	A5bis	…	\
0	1	MET20_999999996	2.501255	1	0	0	0	1	2.0	0.0	…	
1	2	MET20_98888888	0.722914	1	0	0	0	1	5.0	0.0	…	
2	3	MET20_1978307	1.039611	1	0	0	0	1	2.0	0.0	…	
3	4	MET20_1302078	0.976590	1	1	1	0	1	1.0	0.0	…	
4	5	MET20_1869308	0.812315	0	1	0	0	2	0.0	1.0	…	

	rs11recap2	RS11recap	RS193bis	RS2Recap	RS56Recap	RS2	RS11	RS102	\
0	1	2	1.0	1	1	24	0	4	
1	1	2	1.0	4	1	50	0	1	
2	2	1	1.0	3	2	37	1	3	
3	1	2	1.0	5	3	63	0	2	
4	2	1	1.0	3	1	44	1	3	

	cluster_usage	cluster_attitude
0	1	2
1	4	1
2	4	2
3	0	2
4	3	2

[5 rows x 135 columns]

[31]:
```
df_codes.to_csv("data/clusters.csv", index=False)
```

reaffectation-active

February 7, 2024

- Tanguy Malandain
- Hugo Deplagne
- Pierre Litoux
- Param Dave

1 Reaffectation active

1.0.1 Import modules

[1]:
```
import pandas as pd

from utils import feature_importance
from utils import select_top_k_features
from utils import train_and_evaluate
from utils import plot_accuracy
```

1.0.2 Load dataframe

[2]:
```
df = pd.read_csv("data/clusters.csv")
df.head()
```

[2]:

	cle	Respondent_ID	weight	A11	A12	A13	A14	A4	A5	A5bis	…	\
0	1	MET20_999999996	2.501255	1	0	0	0	1	2.0	0.0	…	
1	2	MET20_98888888	0.722914	1	0	0	0	1	5.0	0.0	…	
2	3	MET20_1978307	1.039611	1	0	0	0	1	2.0	0.0	…	
3	4	MET20_1302078	0.976590	1	1	1	0	1	1.0	0.0	…	
4	5	MET20_1869308	0.812315	0	1	0	0	2	0.0	1.0	…	

	rs11recap2	RS11recap	RS193bis	RS2Recap	RS56Recap	RS2	RS11	RS102	\
0	1	2	1.0	1	1	24	0	4	
1	1	2	1.0	4	1	50	0	1	
2	2	1	1.0	3	2	37	1	3	
3	1	2	1.0	5	3	63	0	2	
4	2	1	1.0	3	1	44	1	3	

	cluster_usage	cluster_attitude
0	1	2
1	4	1

```
2              4                2
3              0                2
4              3                2

[5 rows x 135 columns]
```

[3]:
```
df_usage = df.iloc[:, 0:30].copy()
df_usage["cluster"] = df["cluster_usage"]
```

[4]:
```
df_usage.head()
```

[4]:
```
   cle    Respondent_ID    weight  A11  A12  A13  A14  A4   A5  A5bis  …  \
0    1  MET20_999999996  2.501255    1    0    0    0   1  2.0    0.0  …
1    2   MET20_98888888  0.722914    1    0    0    0   1  5.0    0.0  …
2    3    MET20_1978307  1.039611    1    0    0    0   1  2.0    0.0  …
3    4    MET20_1302078  0.976590    1    1    1    0   1  1.0    0.0  …
4    5    MET20_1869308  0.812315    0    1    0    0   2  0.0    1.0  …

   C1_1_slice  C1_2_slice  C1_3_slice  C1_4_slice  C1_5_slice  C1_6_slice  \
0           2           2           2           2           2           2
1           4           4           4           4           4           4
2           4           4           4           4           4           4
3           4           4           5           5           5           5
4           4           4           4           4           4           4

   C1_7_slice  C1_8_slice  C1_9_slice  cluster
0           2           2           2        1
1           4           4           4        4
2           4           4           4        4
3           5           4           4        0
4           4           4           4        3

[5 rows x 31 columns]
```

[5]:
```
df_attitude = df.iloc[:, list(range(3)) + list(range(30, 67))].copy()
df_attitude["cluster"] = df["cluster_attitude"]
```

[6]:
```
df_attitude.head()
```

[6]:
```
   cle    Respondent_ID    weight  A9_1_slice  A9_2_slice  A9_3_slice  \
0    1  MET20_999999996  2.501255           2           2           2
1    2   MET20_98888888  0.722914           1           1           1
2    3    MET20_1978307  1.039611           3           2           2
3    4    MET20_1302078  0.976590           1           2           2
4    5    MET20_1869308  0.812315           2           2           2

   A9_4_slice  A9_5_slice  A9_6_slice  A9_7_slice  …  A11_5_slice  \
```

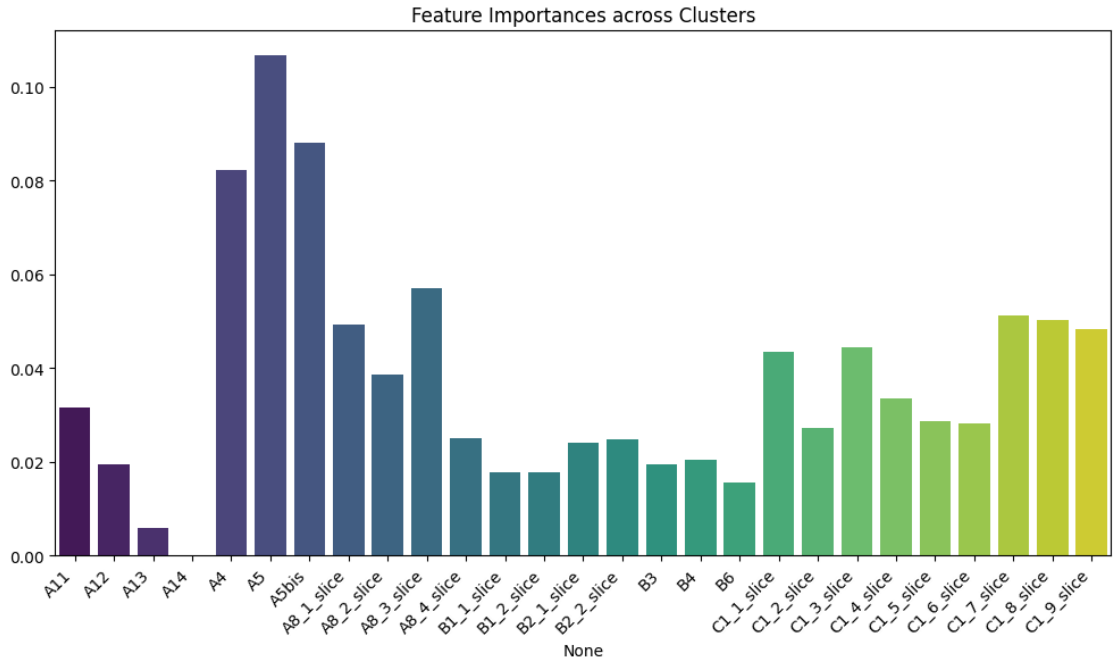
```
0           2           2           2           2  …           2
1           1           1           1           1  …           2
2           2           3           2           2  …           2
3           2           3           2           3  …           1
4           2           2           2           2  …           2

   A11_6_slice  A11_7_slice  A11_8_slice  A11_9_slice  A11_10_slice  \
0            2            2            2            2             2
1            2            2            2            2             2
2            2            2            2            2             2
3            2            3            3            3             2
4            2            2            2            2             2

   A11_11_slice  A11_12_slice  A11_13_slice  cluster
0             2             2             2        2
1             2             2             2        1
2             2             2             2        2
3             2             1             2        2
4             2             2             2        2

[5 rows x 41 columns]
```

1.1 Feature selection

1.1.1 For usage

Get feature importance with a random forest.

[7]:
```
importances = feature_importance(df_usage)
```



Select the top features.

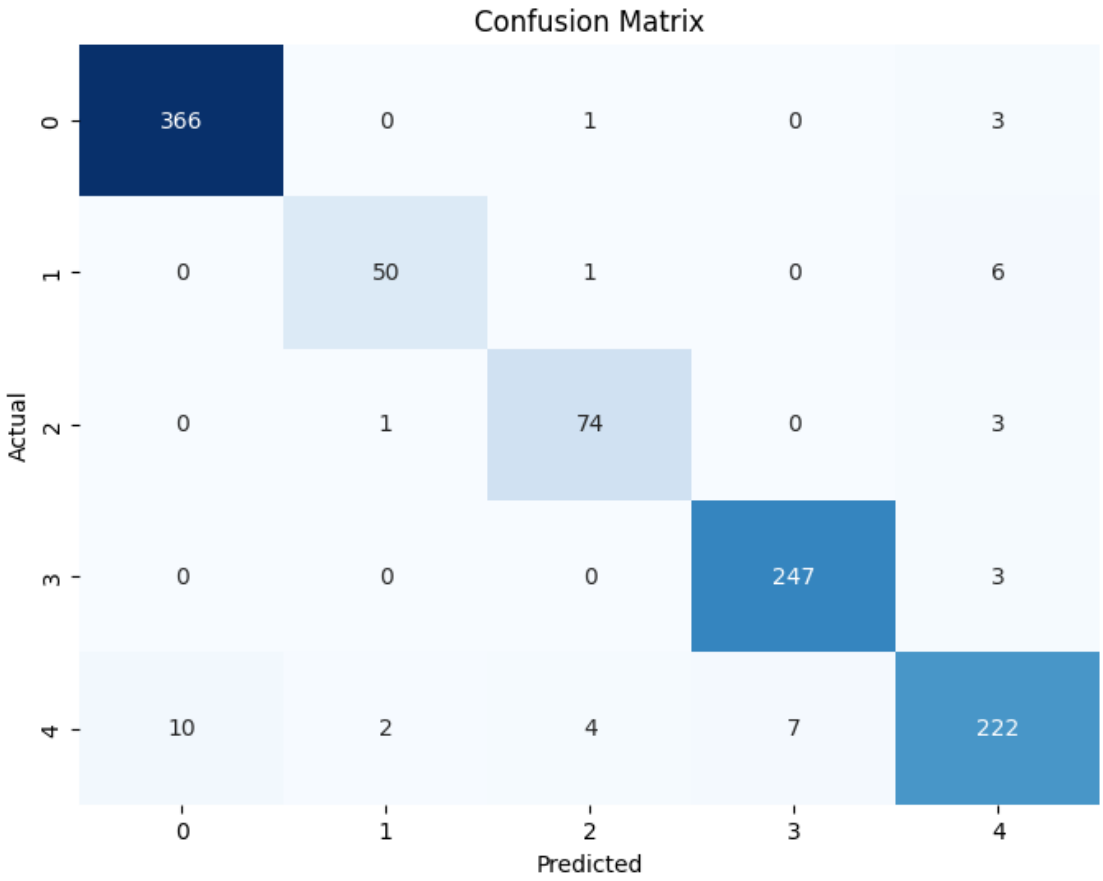
[8]:
```
# Example Usage:
# Assuming importances is a list of floats and feature_names is a list of␣
 ↪feature names
# Replace 5 with the desired number of top features to select
top_k_features = select_top_k_features(importances, df_usage.iloc[:, 3:-1].
 ↪columns, k=10)
print("Top Features:", top_k_features)
```

```
Top Features: ['A5', 'A5bis', 'A4', 'A8_3_slice', 'C1_7_slice', 'C1_8_slice',
'A8_1_slice', 'C1_9_slice', 'C1_3_slice', 'C1_1_slice']
```

Implement XGB classifier.

[9]:
```
accuracy = train_and_evaluate(df_usage, df_usage.iloc[:, 3:-1].columns)
```

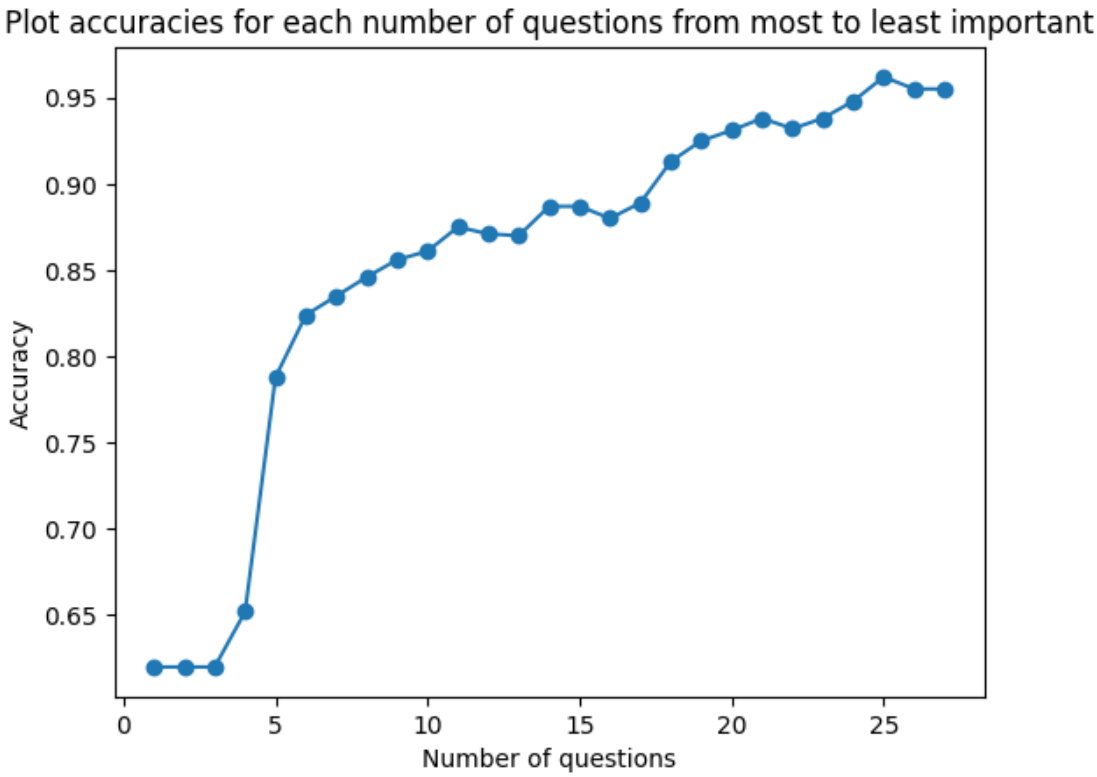
```
Accuracy Score: 0.959
```



Le modele de classification a une precision d'environ 96%.

Nous allons maintenant entrainer avec seulement la meilleure feature et iterativement rajouter la suite des meilleures features.

[10]:
```
plot_accuracy(df_usage, importances, df_usage.iloc[:, 3:-1].columns)
```



Nous observons une forte croissance de notre precision a partir de la 5eme question.

Arbitrairement, on peut etablir que les 7 meilleurs questions sont suffisant pour avoir une bonne precision.

Ces 7 questions sont les **golden questions** parmi les 27 questions d'usage au total.

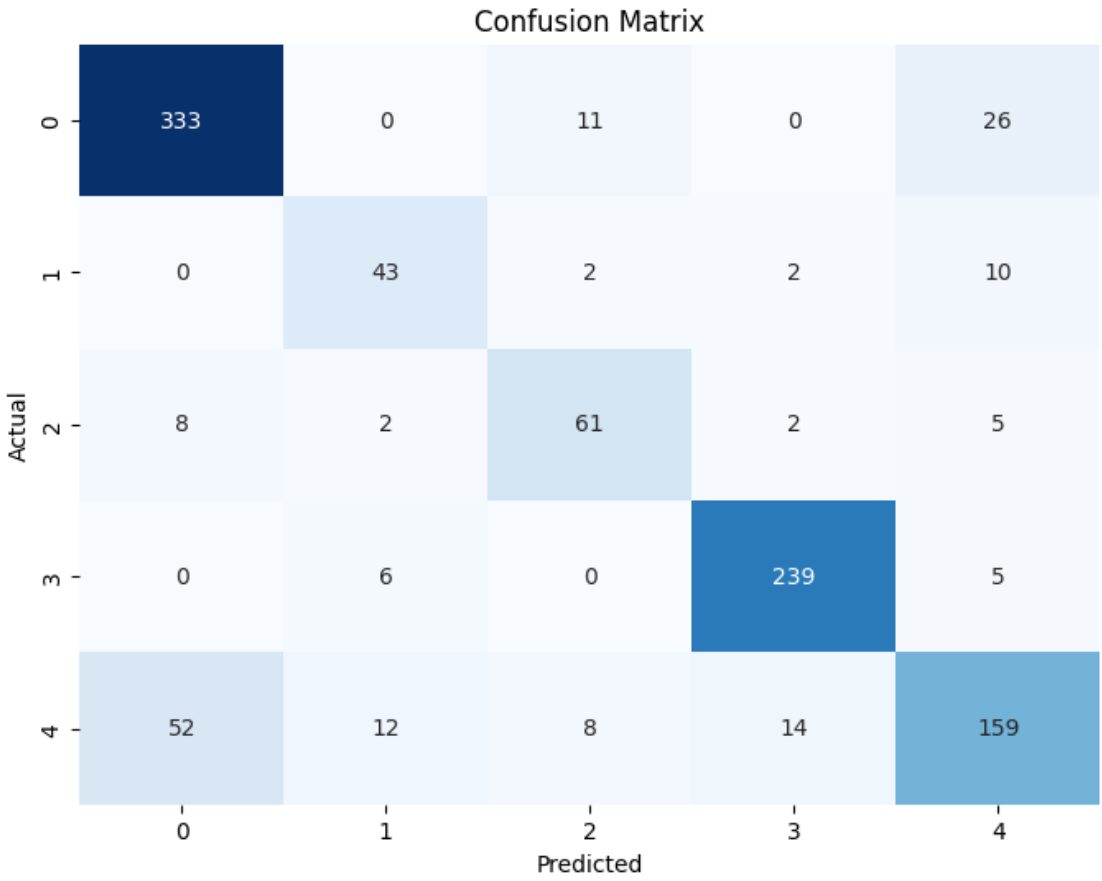
[11]:
```
select_top_k_features(importances, df_usage.iloc[:, 3:-1].columns, 7)
```

[11]:
```
['A5', 'A5bis', 'A4', 'A8_3_slice', 'C1_7_slice', 'C1_8_slice', 'A8_1_slice']
```

Voici les 7 **golden questions**. - `A5`: Taille du jardin - `A5bis`: Taille de la terrasse/balcon - `A4`: TYPE D'ESPACE - `A8_3_slice`: Temps passé à l'entretien de l'espace extérieur En automne - `C1_7_slice`: Fréquence consultation sites dédiés au jardinage Des forums de discussion sur Internet, centrés sur le jardinage, la décoration ou l'entretien des espaces extérieurs - `C1_8_slice`: Fréquence consultation sites dédiés au jardinage Des sites Internet de fabricants ou de marques de produits pour d'aménagement ou d'entretien des espaces extérieurs - `A8_1_slice`: Temps passé à l'entretien de l'espace extérieur Au printemps

[12]:
```
train_and_evaluate(df_usage, ['A5', 'A5bis', 'A4', 'A8_3_slice', 'C1_7_slice',␣
↪'C1_8_slice', 'A8_1_slice'])
```

```
Accuracy Score: 0.835
```



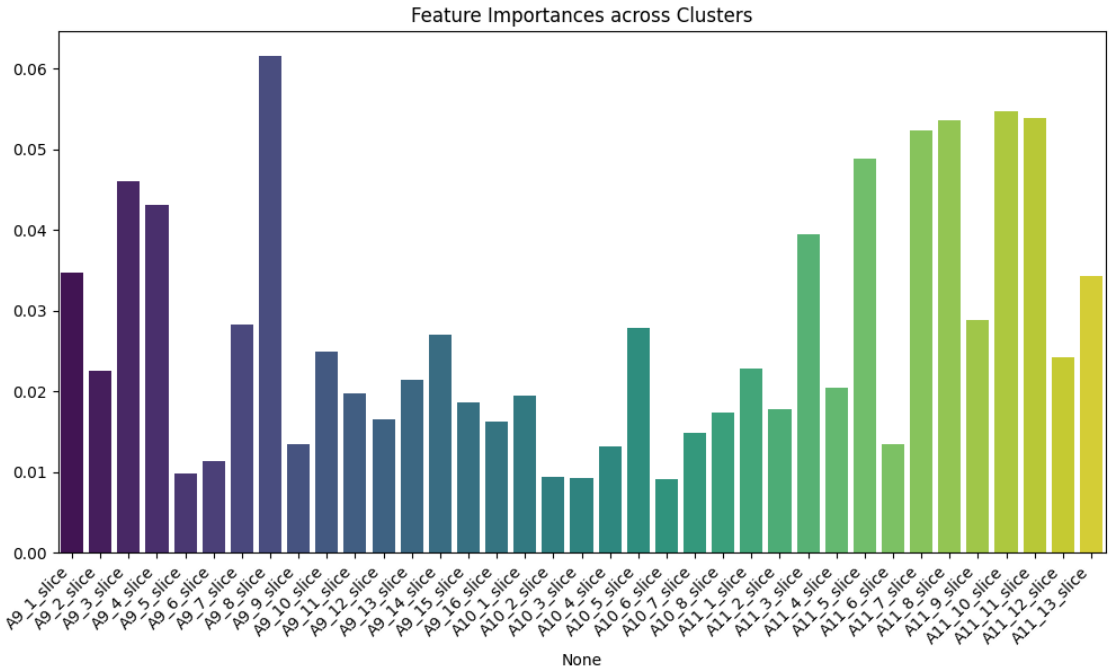
[12]: 0.835

Avec seulement les **golden questions**, nous avons une precision de 83.5%. Ceci nous montre que ces questions sont largement suffisantes pour la classification. Ainsi, ces questions seront assez pour la reaffectation.

1.1.2 For attitude

Regardons les features les plus importantes a partir du random forest.

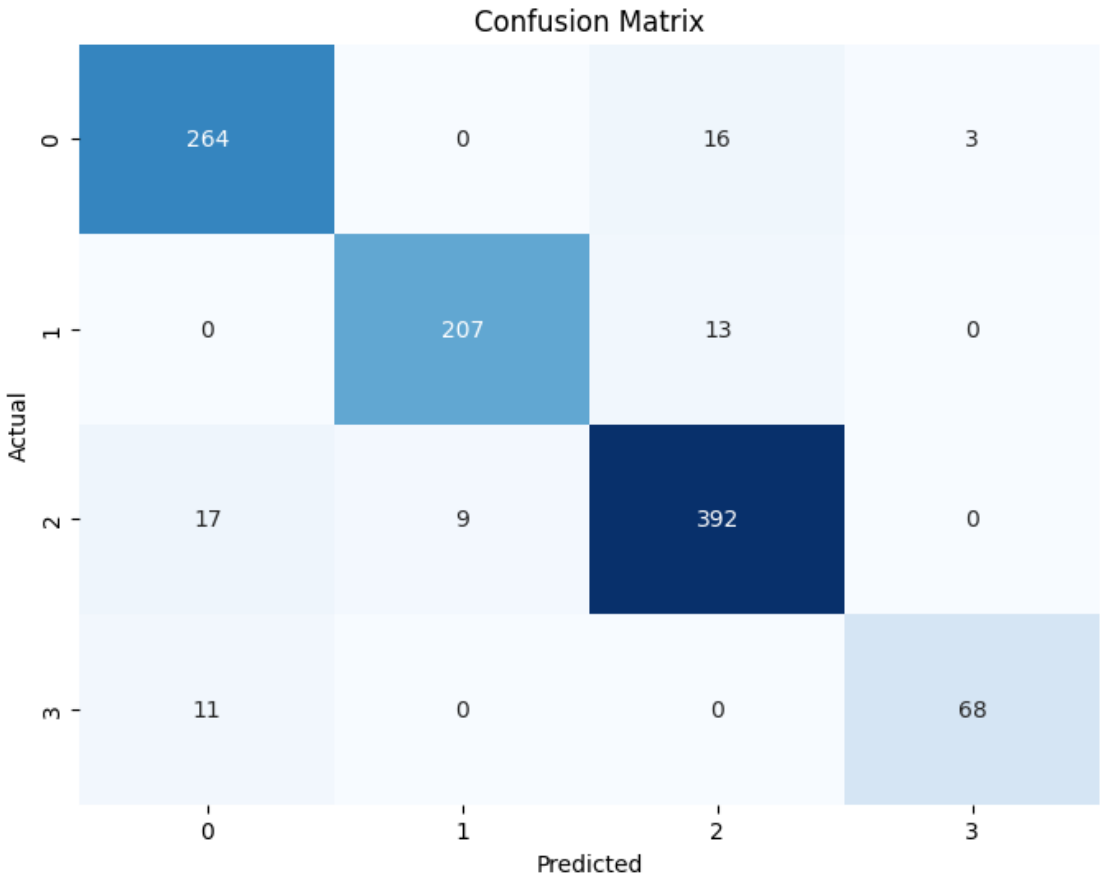
[13]:
```
importances = feature_importance(df_attitude)
```



Entrainons notre model de classification sur toutes les features.

[14]:
```
accuracy = train_and_evaluate(df_attitude, df_attitude.iloc[:, 3:-1].columns)
```

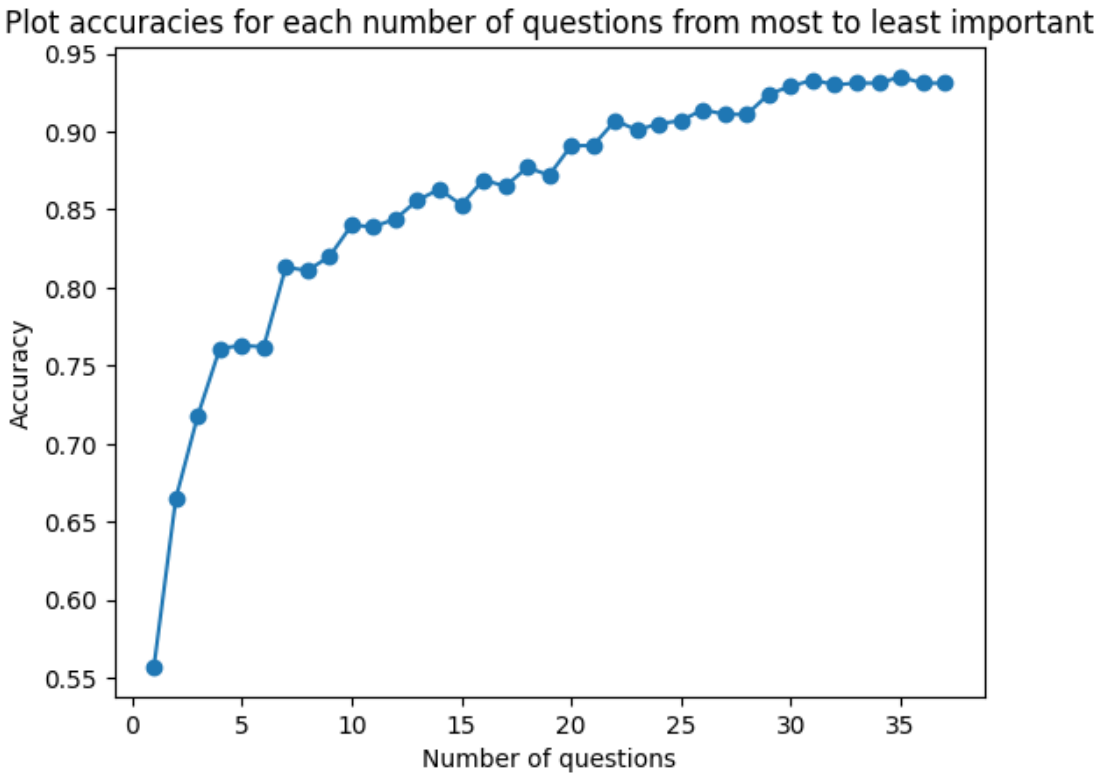
```
Accuracy Score: 0.931
```



Nous obtenons une precision de 93.1%.

Regardons maintenant la precision en prenant un nombre de feature variable.

[15]:
```
plot_accuracy(df_attitude, importances, df_attitude.iloc[:, 3:-1].columns)
```



Nous observons une forte augmentation de la precision jusqu'a la question 4 et un autre pic à la question 7.
Nous pouvons donc egalement choisir arbitrairement 7 **golden questions** parmi les 37 questions sur l'attitude.

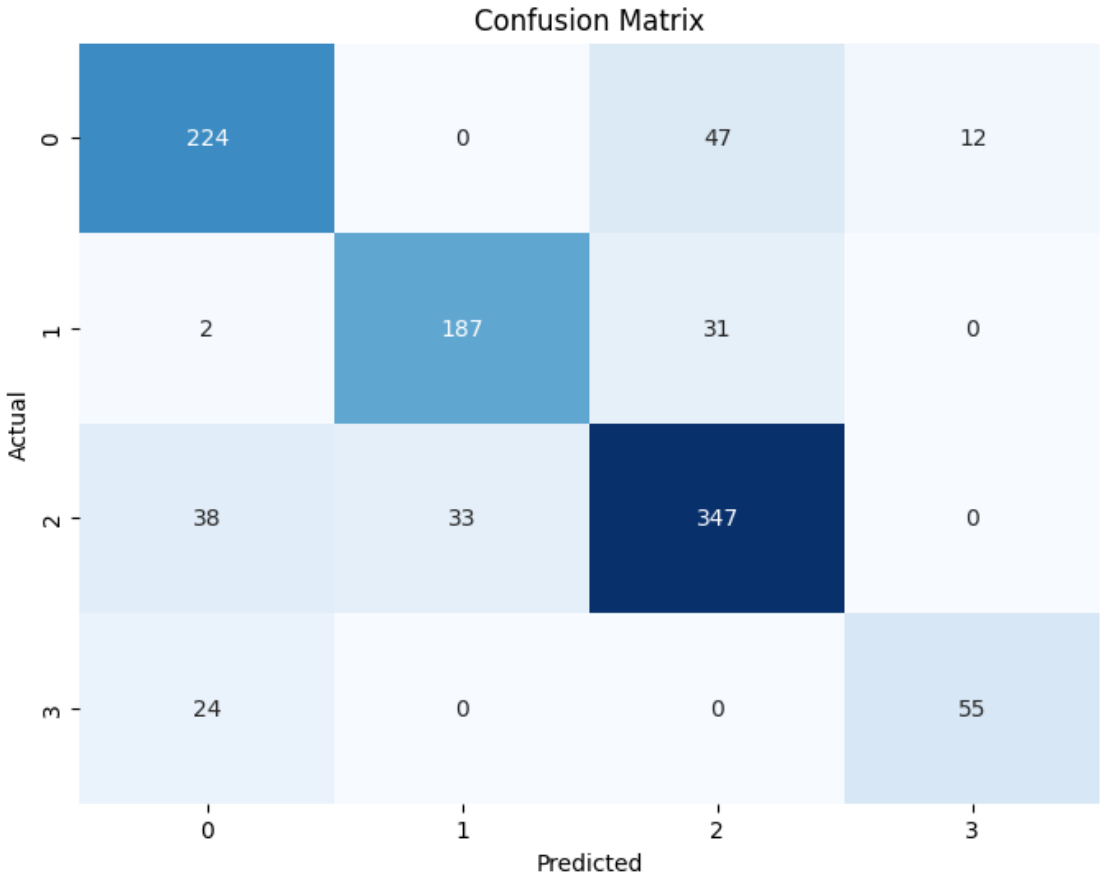
[16]:
```
select_top_k_features(importances, df_attitude.iloc[:, 3:-1].columns, 7)
```

[16]:
```
['A9_8_slice',
 'A11_10_slice',
 'A11_11_slice',
 'A11_8_slice',
 'A11_7_slice',
 'A11_5_slice',
 'A9_3_slice']
```

Voici les 7 **golden questions**: - `A9_8_slice`: Je m'investis beaucoup dans l'aménagement et l'entretien des espaces extérieurs - `A11_10_slice`: Un moyen de réaliser jusqu'au bout quelque chose de vos propres mains - `A11_11_slice`: Un moyen de transmettre des connaissances, des pratiques qu'il est indispensable de perpétuer - `A11_8_slice`: Votre espace favori de loisirs et de liberté - `A11_7_slice`: Votre contact privilégié avec le vert, la nature - `A11_5_slice`: Un moyen de se resourcer, de refaire le plein d'énergie - `A9_3_slice`: Je recherche souvent des informations sur l'aménagement des espaces extérieurs

[17]:
```
train_and_evaluate(df_attitude, ['A9_8_slice', 'A11_10_slice', 'A11_11_slice',␣
↪'A11_8_slice', 'A11_7_slice', 'A11_5_slice', 'A9_3_slice'])
```

```
Accuracy Score: 0.813
```



[17]:
```
0.813
```

Avec seulement les **golden questions**, nous avons une precision de 81.3%. Ceci nous montre que ces questions sont largement suffisantes pour la classification. Ainsi, ces questions seront assez pour la reaffectation.

reaffectation-illustrative

February 7, 2024

- Tanguy Malandain
- Hugo Deplagne
- Pierre Litoux
- Param Dave

1 Reaffectation illustrative

1.0.1 Import modules

[1]:
```
import numpy as np
import pandas as pd
```

1.0.2 Load dataframe

[2]:
```
df = pd.read_csv("data/clusters.csv")
df.head()
```

[2]:

	cle	Respondent_ID	weight	A11	A12	A13	A14	A4	A5	A5bis	…	\
0	1	MET20_999999996	2.501255	1	0	0	0	1	2.0	0.0	…	
1	2	MET20_98888888	0.722914	1	0	0	0	1	5.0	0.0	…	
2	3	MET20_1978307	1.039611	1	0	0	0	1	2.0	0.0	…	
3	4	MET20_1302078	0.976590	1	1	1	0	1	1.0	0.0	…	
4	5	MET20_1869308	0.812315	0	1	0	0	2	0.0	1.0	…	

	rs11recap2	RS11recap	RS193bis	RS2Recap	RS56Recap	RS2	RS11	RS102	\
0	1	2	1.0	1	1	24	0	4	
1	1	2	1.0	4	1	50	0	1	
2	2	1	1.0	3	2	37	1	3	
3	1	2	1.0	5	3	63	0	2	
4	2	1	1.0	3	1	44	1	3	

	cluster_usage	cluster_attitude
0	1	2
1	4	1
2	4	2
3	0	2
4	3	2

```
[5 rows x 135 columns]
```

Pour les clusters sur les question d'usage Prendre les questions d'attitude et les questions mentionnees.

[3]:
```
selected_columns = list(range(3)) + list(range(30, 67)) + list(range(df.
 ↪shape[1]-18, df.shape[1]))

# Create a new DataFrame with the selected columns
df_cluster = df.iloc[:, selected_columns].copy()

df_usage = df_cluster.drop("cluster_attitude", axis=1)
df_usage.rename(columns={'cluster_usage': 'cluster'}, inplace=True)
df_usage.columns
```

[3]:
```
Index(['cle', 'Respondent_ID', 'weight', 'A9_1_slice', 'A9_2_slice',
       'A9_3_slice', 'A9_4_slice', 'A9_5_slice', 'A9_6_slice', 'A9_7_slice',
       'A9_8_slice', 'A9_9_slice', 'A9_10_slice', 'A9_11_slice', 'A9_12_slice',
       'A9_13_slice', 'A9_14_slice', 'A9_15_slice', 'A9_16_slice',
       'A10_1_slice', 'A10_2_slice', 'A10_3_slice', 'A10_4_slice',
       'A10_5_slice', 'A10_6_slice', 'A10_7_slice', 'A10_8_slice',
       'A11_1_slice', 'A11_2_slice', 'A11_3_slice', 'A11_4_slice',
       'A11_5_slice', 'A11_6_slice', 'A11_7_slice', 'A11_8_slice',
       'A11_9_slice', 'A11_10_slice', 'A11_11_slice', 'A11_12_slice',
       'A11_13_slice', 'rs3', 'rs5', 'rs6', 'RS1', 'RS191', 'RS192', 'RS193',
       'RS102RECAP', 'rs11recap2', 'RS11recap', 'RS193bis', 'RS2Recap',
       'RS56Recap', 'RS2', 'RS11', 'RS102', 'cluster'],
      dtype='object')
```

Pour les clusters sur les question d'attitude Prendre les questions d'usage et les questions mentionnees.

[4]:
```
df_attitude = df.iloc[:, list(range(30)) + list(range(-18, -2))].copy()
df_attitude["cluster"] = df["cluster_attitude"]
df_attitude.columns
```

[4]:
```
Index(['cle', 'Respondent_ID', 'weight', 'A11', 'A12', 'A13', 'A14', 'A4',
       'A5', 'A5bis', 'A8_1_slice', 'A8_2_slice', 'A8_3_slice', 'A8_4_slice',
       'B1_1_slice', 'B1_2_slice', 'B2_1_slice', 'B2_2_slice', 'B3', 'B4',
       'B6', 'C1_1_slice', 'C1_2_slice', 'C1_3_slice', 'C1_4_slice',
       'C1_5_slice', 'C1_6_slice', 'C1_7_slice', 'C1_8_slice', 'C1_9_slice',
       'rs3', 'rs5', 'rs6', 'RS1', 'RS191', 'RS192', 'RS193', 'RS102RECAP',
       'rs11recap2', 'RS11recap', 'RS193bis', 'RS2Recap', 'RS56Recap', 'RS2',
       'RS11', 'RS102', 'cluster'],
      dtype='object')
```

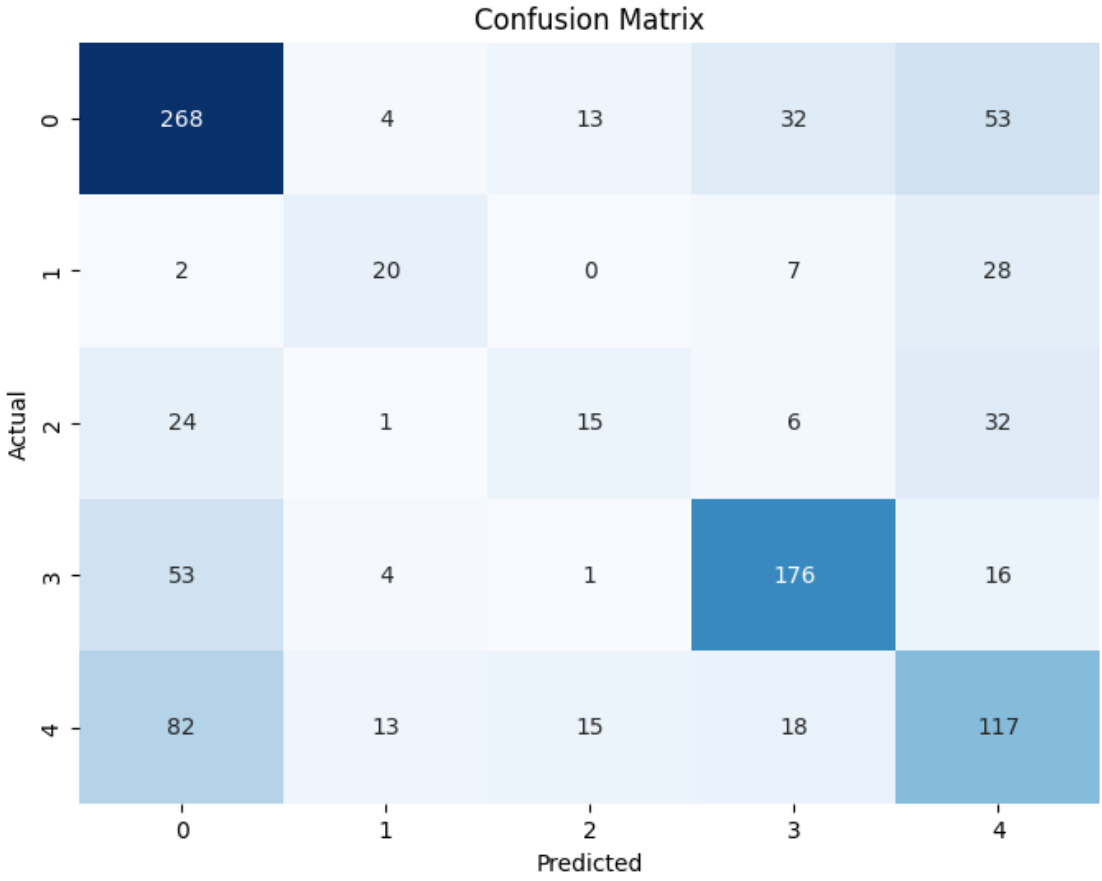
1.0.3 Implement reaffectation

[5]:
```
from utils import train_and_evaluate
```

Pour la segmentation variable verte.

[6]:
```
train_and_evaluate(df_usage, df_usage.iloc[:, 3:-1].columns)
```

```
Accuracy Score: 0.596
```



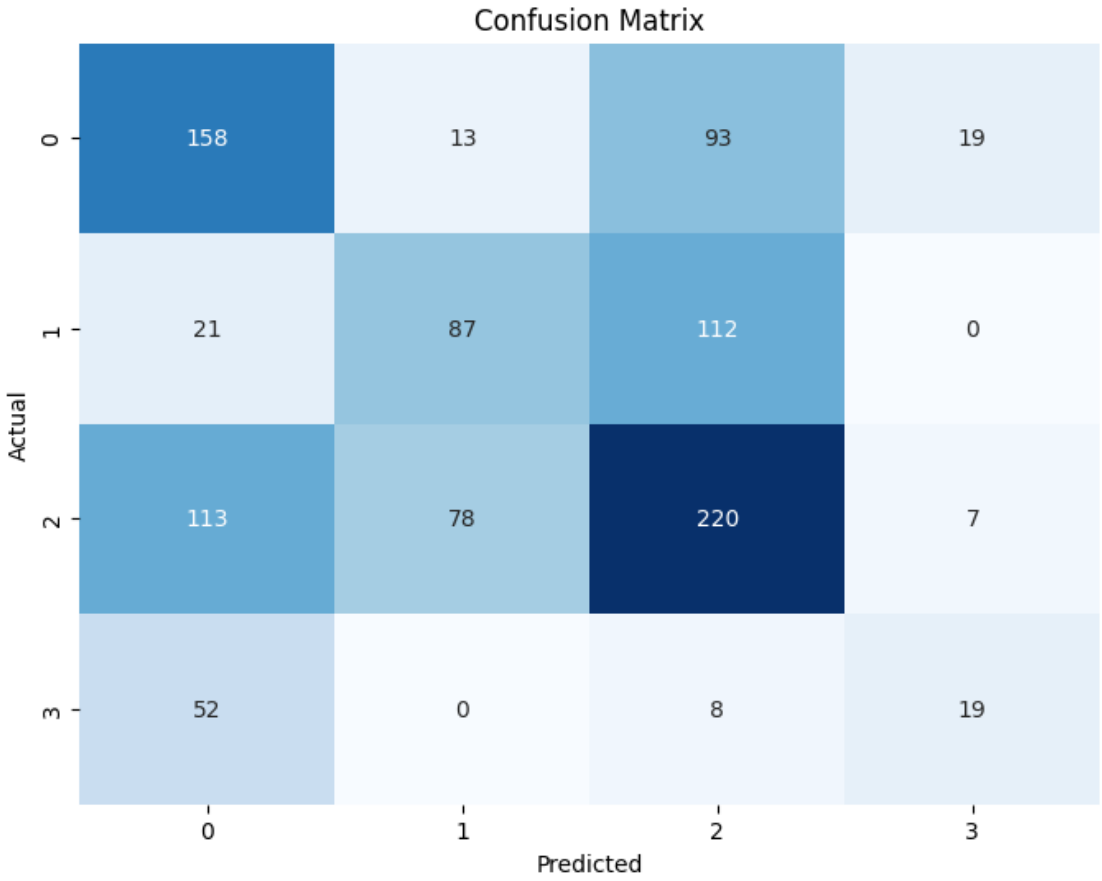
[6]:
```
0.596
```

Nous obtenons une precision de 59.6% pour la segmentation variable verte. Un resultat peu satisfaisant. Ceci est du au fait que les clusters n'ont pas ete forme à partir de ces questions.

Pour la segmentation variable orange

[7]:
```
train_and_evaluate(df_attitude, df_attitude.iloc[:, 3:-1].columns)
```

```
Accuracy Score: 0.484
```



[7]: 0.484

Nous obtenons une precision de 48.4% pour la segmentation variable orange. Un resultat peu satisfaisant. Ceci est du au fait que les clusters n'ont pas ete forme à partir de ces questions.