# TP 1 - Introduction aux bibliothèques Python

## 1 Analyse descriptive des données

### 1.1 Combien y a-t-il d'attributs ? Que représentent-ils ? Quel est leur type ?

Il y a 7 attributs: 'fruit_label', 'fruit_name', 'fruit_subtype', 'mass', 'width', 'height', 'color_score'.

Ils représentent différentes caractéristiques des fruits, telles que le label, le nom, le sous-type, la masse, la largeur, la hauteur et le score de couleur.

Le type des attributs peut varier, mais généralement, 'fruit_label' est catégorique, 'fruit_name' et 'fruit_subtype' sont des chaînes, et 'mass', 'width', 'height', 'color_score' sont numériques.

**Quelle est la classe à prédire ?**

La classe à prédire semble être 'fruit_label'.

**Combien y a-t-il d'instances dans le fichier ?**

Il y a 59 instances.

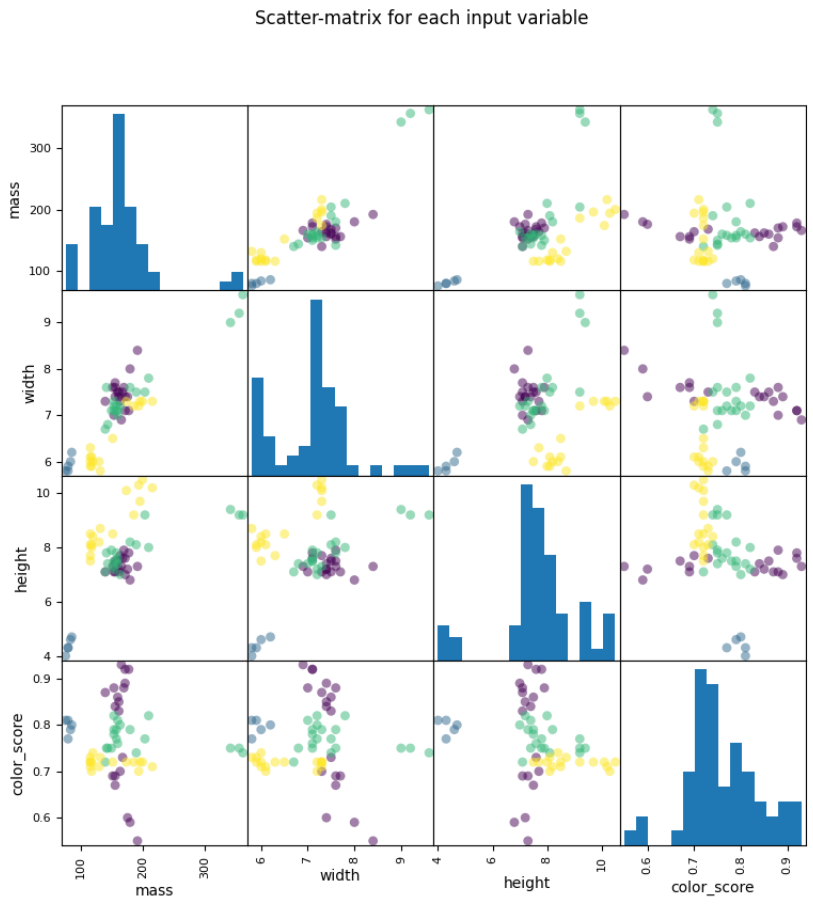
**Les classes sont-elles équilibrées quant au nombre d'instances ?**

```
Classe 'apple': 19
Classe 'mandarin': 5
Classe 'orange': 19
Classe 'lemon': 16
```

Non les classes ne sont pas équilibrées car mandarin a bien moins d’instances.

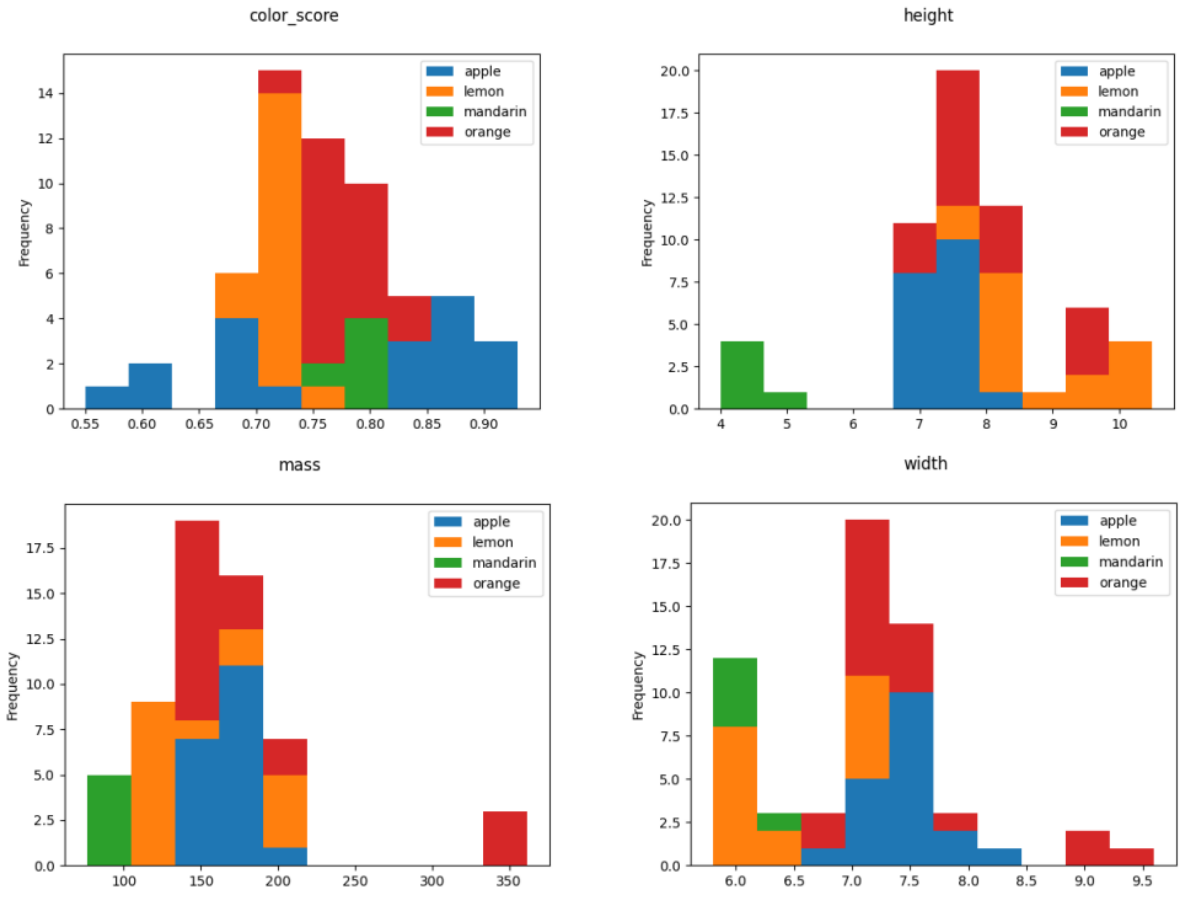
### 1.2 Examinez les données de manière graphique à l'aide d'un diagramme de dispersions (fonction plotting.scatter_matrix()

## de pandas) et d'histogrammes montrant la répartition selon un attribut et la classe.




**Scatter Matrix:**

- La première partie du code crée une matrice de dispersion (scatter matrix) pour les attributs 'mass', 'width', 'height' et 'color_score'.
- Chaque point dans la matrice représente une paire d'attributs, et les points sont colorés en fonction de l'étiquette de classe du fruit correspondant.
- Cela permet de visualiser la distribution des données et de détecter des tendances ou des regroupements en fonction des classes.



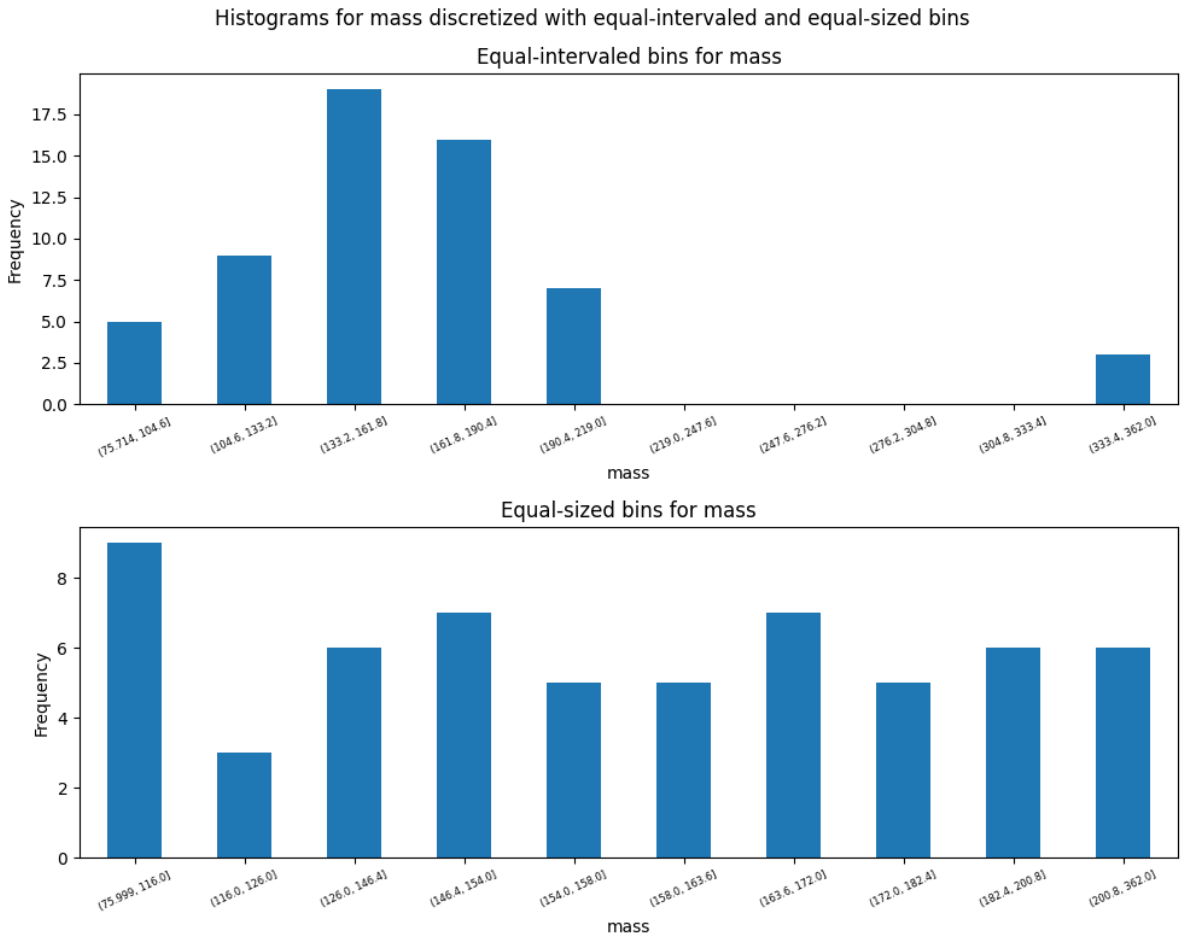

**Histogrammes empilés:**

- La deuxième partie du code génère des histogrammes empilés pour chaque attribut ('mass', 'width', 'height', 'color_score') séparément, en les empilant en fonction des différentes classes de fruits.
- Chaque histogramme représente la distribution de l'attribut spécifique pour chaque classe de fruit.
- Cela permet d'observer la répartition des valeurs d'attributs pour différentes classes, aidant ainsi à identifier des schémas distincts ou des caractéristiques importantes.

# 2 Prétraitement

## 2.1 Appliquez à partir des données originelles deux variantes de discrétisation : une première reposant sur des intervalles de même taille (equal-intervaled avec la fonction cut() de pandas) ou de mêmes effectifs (equal-sized bins avec qcut()). En achant les historiques correspondant à chacune des deux

## congurations à l'aide de plot.bar(), comparez les histogrammes obtenus pour l'attribut  mass




1. **Discrétisation avec des intervalles égaux :**
    - La première sous-figure crée un histogramme qui représente la fréquence des occurrences de valeurs de l'attribut masse groupées dans 10 intervalles de largeur égale.
    - Les intervalles sont déterminés par la fonction **pd.cut(fruits[attr], 10)**, qui divise l'attribut masse en 10 intervalles de largeur égale.
    - Cela permet de visualiser comment les valeurs de la masse sont distribuées lorsqu'elles sont regroupées dans des intervalles de taille égale.
    - On peut voir qu'il y a un trou entre 219 et 333 de masse donc pas de fruits pesant ces masses.

2. **Discrétisation avec des bins de même effectif (équidistants) :**
    - La deuxième sous-figure utilise la fonction **pd.qcut(fruits[attr], 10)**, qui divise l'attribut masse en 10 intervalles de même effectif (fréquence).
    - Cela signifie que chaque intervalle contiendra approximativement le même nombre d'observations.
    - L'histogramme résultant montre comment les valeurs de masse sont distribuées lorsqu'elles sont regroupées en intervalles contenant un nombre similaire d'observations.

## 2.2 Reprenez les données originelles. Quel est l'effet du filtre de de normalisation preprocessing.MinMaxScaler de scikit-learn ?

Le prétraitement avec le MinMaxScaler de scikit-learn permet de mettre à l'échelle les données dans une plage spécifique (par défaut, entre 0 et 1). Cela peut être particulièrement utile pour des algorithmes qui sont sensibles à l'échelle des caractéristiques.

## Dans quel cas d'utilisation est-il préférable d'appeler la fonction transform() de cet objet plutôt que fit_transform() ?

Dans certains cas d'utilisation, vous pouvez préférer utiliser la fonction **transform()** plutôt que **fit_transform()** si vous avez déjà ajusté le scaler à un autre ensemble de données. Par exemple, si vous avez séparé vos données en ensembles d'entraînement et de test, vous ajusterez le scaler sur l'ensemble d'entraînement avec **fit()** et ensuite appliquerez la même transformation à l'ensemble de test avec **transform()**. Cela garantit que la transformation est cohérente entre les ensembles d'entraînement et de test, évitant tout problème de fuite d'information.

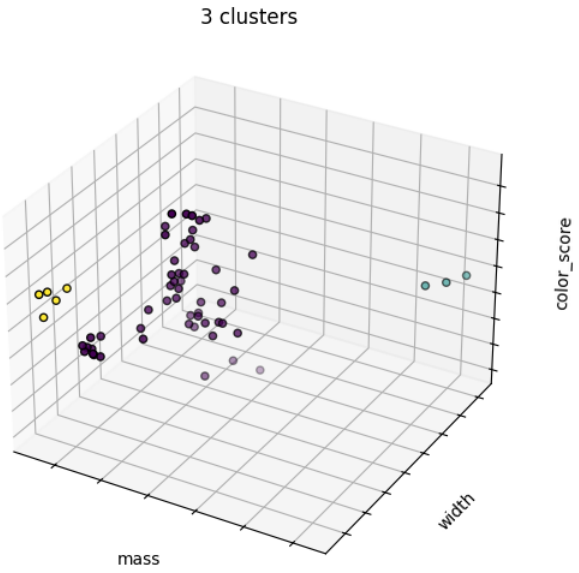
```
Avant normalisation:
              mass       width      height  color_score
count    59.000000   59.000000   59.000000    59.000000
mean    163.118644    7.105085    7.693220     0.762881
std      55.018832    0.816938    1.361017     0.076857
min      76.000000    5.800000    4.000000     0.550000
25%     140.000000    6.600000    7.200000     0.720000
50%     158.000000    7.200000    7.600000     0.750000
75%     177.000000    7.500000    8.200000     0.810000
max     362.000000    9.600000   10.500000     0.930000

Après normalisation:
             mass       width      height color_score
count   59.000000   59.000000   59.000000   59.000000
mean     0.304611    0.343443    0.568188    0.560214
std      0.192374    0.214984    0.209387    0.202257
min      0.000000    0.000000    0.000000    0.000000
25%      0.223776    0.210526    0.492308    0.447368
50%      0.286713    0.368421    0.553846    0.526316
75%      0.353147    0.447368    0.646154    0.684211
max      1.000000    1.000000    1.000000    1.000000
```

Après la normalisation on voit que les données sont maintenant entre 0 et 1.

# 3 Clustering

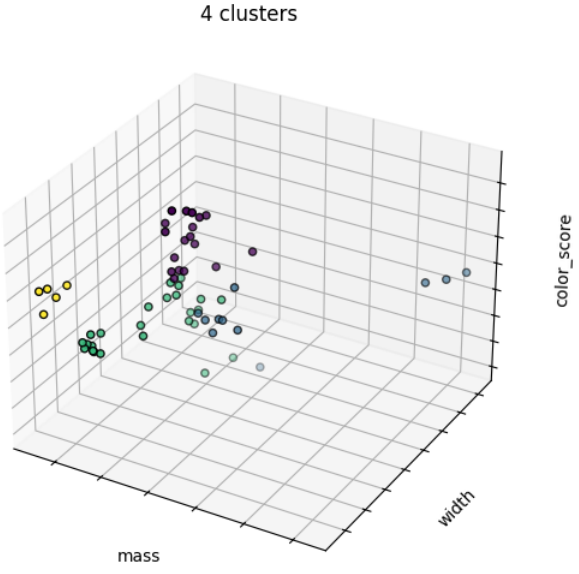
### 3.1 Ouvrez le chier de données en appliquant un ltre de normalisation. Appliquez KMeans sur ces données en xant successivement le nombre de clusters à 3, 4 et 5. Examinez manuellement les résultats du clustering en traçant les instances dans une espace en 3 dimensions correspondant aux attributs `mass` , `width` et `color_score` .

### Quelle(s) classe(s) représente chacun des clusters ?



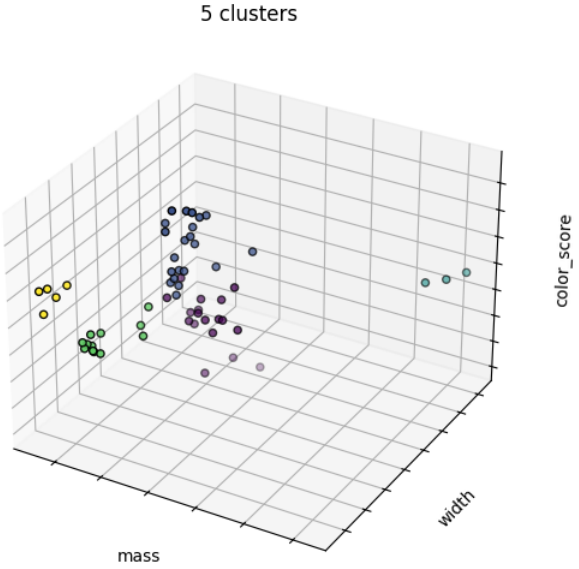

**Cluster 3 :**

- Mandarine pour le cluster jaune
- apple pour le cluster vert
- les autres fruits pour le cluster violet



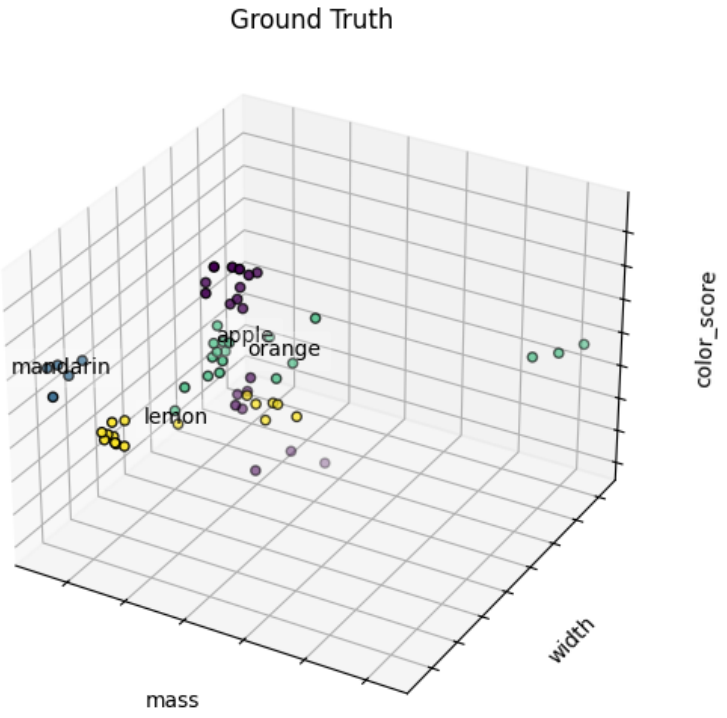
**Cluster 4 :**

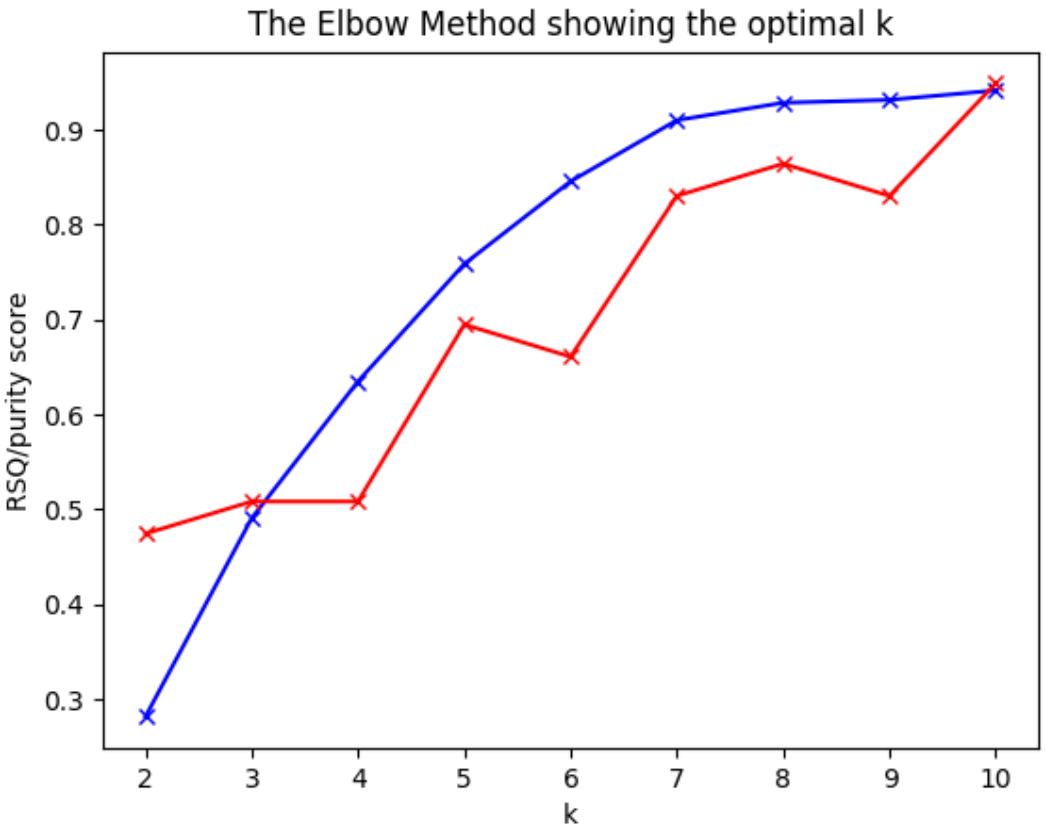
- Mandarine pour le cluster jaune
- lemon et apple pour le cluster vert
- orange et apple pour le cluster violet
- apple pour le cluster bleu




**Cluster 5 :**

- Mandarine pour le cluster jaune
- lemon pour le cluster vert
- orange et apple pour le cluster bleu
- orange et apple pour le cluster violet
- apple pour le cluster vert pomme


Ground Truth
apple
orange
mandarin
lemon
mass
width
color_score



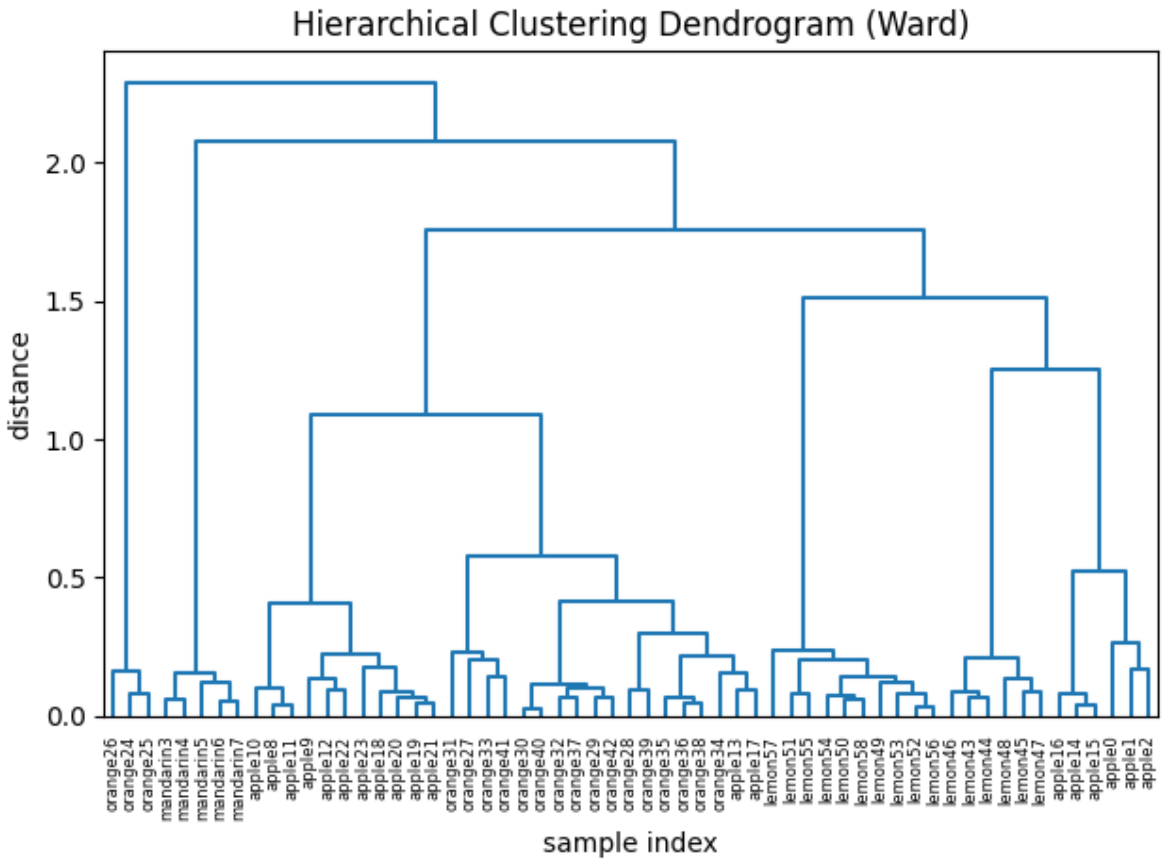

La courbe bleue représente R2 et la courbe rouge le score de pureté.

## 3.2 Comment varie R2 ? Quel est le nombre de clusters que vous retiendriez ?

R2 est croissant et se stabilise à 7 donc je retiendrai 7 clusters.

## 3.3 Quel est le nombre de groupes que vous retiendriez suivant critère du score de pureté ?

Je retiendrai 10 groupes car le nombre de groupes augmente jusqu'au bout.




## 3.4 Commentez l'arbre obtenu et proposez une partition en sélectionnant une hauteur appropriée de coupure.

**Hierarchical Clustering Dendrogram Graphique:**

- Ce graphique représente un dendrogramme pour l'algorithme de clustering hiérarchique.
- Les données normalisées X_norm sont utilisées pour effectuer le clustering hiérarchique.
- La méthode de liaison utilisée est "ward", qui minimise la variance entre les clusters.
- Les étiquettes pour chaque échantillon dans le dendrogramme sont formées en combinant le nom du fruit avec son index dans le dataframe.
- La fonction dendrogram est utilisée pour créer le dendrogramme en utilisant la matrice de liaison.

- La barre horizontale dans le dendrogramme indique la fusion des clusters. Plus la barre est longue, plus la distance entre les clusters fusionnés est grande.
- Les axes x et y représentent respectivement l'index de l'échantillon et la distance.

Je couperai la hauteur à 0.5 pour avoir un cluster qui contient toute les mandarines.
Plus bas la coupure n'ajouterai pas de clusters cohérents.

# 4 Classement

## 4.1 Comparez les performances des méthodes de classement.

```
Accuracy of Dummy classifier on training set: 0.34
Accuracy of Dummy classifier on test set: 0.27
[[4 0 0 0]
 [1 0 0 0]
 [8 0 0 0]
 [2 0 0 0]]
Accuracy of Naive Bayes classifier on training set: 0.86
Accuracy of Naive Bayes classifier on test set: 0.67
[[4 0 0 0]
 [0 1 0 0]
 [4 0 3 1]
 [0 0 0 2]]
Accuracy of Decision tree classifier on training set: 1.00
Accuracy of Decision tree classifier on test set: 0.67
[[4 0 0 0]
 [0 1 0 0]
 [2 0 4 2]
 [0 0 1 1]]
Accuracy of Random Forest classifier on training set: 1.00
Accuracy of Random Forest classifier on test set: 0.80
[[4 0 0 0]
 [0 1 0 0]
 [2 0 5 1]
 [0 0 0 2]]
```

```
Accuracy of Logistic regression classifier on training set: 0.82
Accuracy of Logistic regression classifier on test set: 0.47
[[2 0 2 0]
 [0 1 0 0]
 [6 0 2 0]
 [0 0 0 2]]
```

## Quel est le taux de classification obtenu pour chaque méthode ?

1. **Régression logistique :**
   - Précision sur l'ensemble d'entraînement : 0.82
   - Précision sur l'ensemble de test : 0.47
2. **Classificateur Dummy :**
   - Précision sur l'ensemble d'entraînement : 0.34
   - Précision sur l'ensemble de test : 0.27
3. **Naive Bayes :**
   - Précision sur l'ensemble d'entraînement : 0.86
   - Précision sur l'ensemble de test : 0.67
4. **Arbre de décision :**
   - Précision sur l'ensemble d'entraînement : 1.00
   - Précision sur l'ensemble de test : 0.73
5. **Random Forest :**
   - Précision sur l'ensemble d'entraînement : 1.00
   - Précision sur l'ensemble de test : 0.87

Dans ce cas, le modèle Random Forest a la meilleure précision sur l'ensemble de test parmi les méthodes mentionnées.

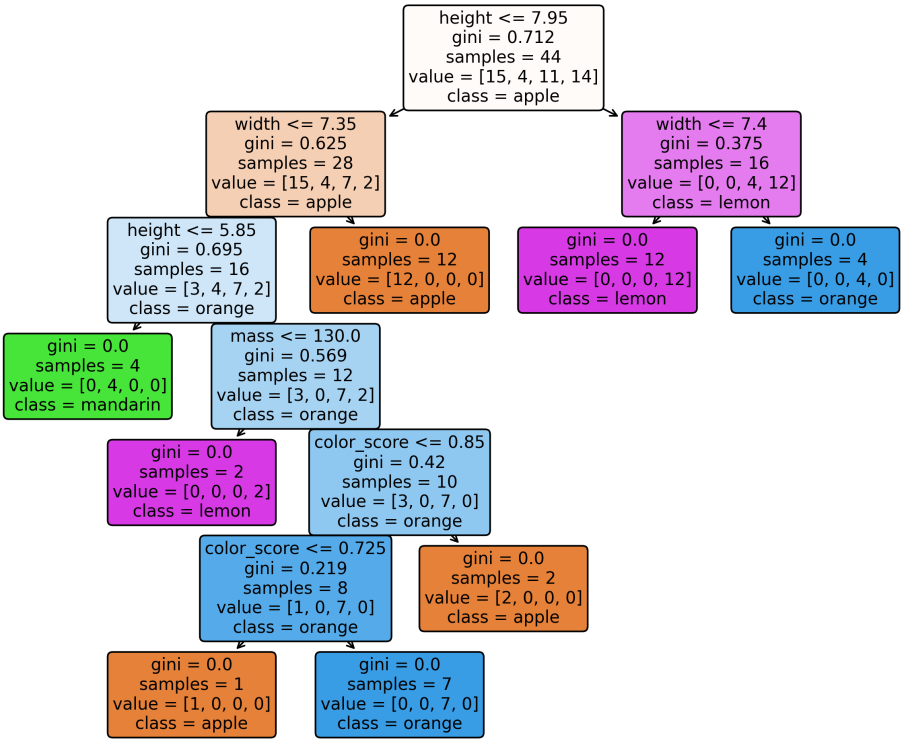
## Quelle est la classe pour laquelle la prédiction fait le plus d'erreurs ?

```
Classe 'apple': 0 + 0 + 0 + 0 + 2 = 2
Classe 'mandarin': 1 + 4 + 0 + 0 + 0 = 5
```

```
Classe 'orange': 8 + 5 + 4 + 3 + 6 = 26
Classe 'lemon': 2 + 0 + 1 + 0 + 0 = 3
```

La classe pour laquelle la prédiction fait le plus d'erreurs est 'orange', avec une somme d'erreurs de 26. Par conséquent, 'orange' est la classe qui est le plus souvent mal prédite par les classificateurs.

## Dans le cas de l'arbre de décision, visualisez avec a fonction tree.plot_tree et commentez l'arbre construit par le modèle.




Observations concernant de l'arbre de décision :

1. **Racine de l'arbre**: Le nœud racine commence par une question sur la hauteur du fruit (si elle est inférieure ou égale à 7,95). Cela suggère que la hauteur est une caractéristique importante pour classer les fruits dans cet ensemble de données.

2. **Impureté de Gini**: Chaque nœud indique un score de Gini, qui est une mesure d'impureté. Un score de Gini de 0 indique que tous les échantillons à ce nœud appartiennent à une seule classe. Par exemple, le nœud qui sépare les oranges des autres fruits a un score de Gini de 0.
3. **Feuilles et Classes**: Les feuilles (les nœuds finaux de chaque branche) indiquent la classification finale et montrent la distribution des classes parmi les échantillons qui ont abouti à cette feuille. Par exemple, il y a des feuilles spécifiques pour les classes telles que apple, orange, lemon, et mandarin.
4. **Découpages et Valeurs**: Chaque nœud interne utilise une valeur de seuil pour diviser les données en deux groupes. Par exemple, la première division après la racine est basée sur la largeur, avec un seuil de 7,35.
5. **Échantillons**: Le nombre d'échantillons à chaque nœud est indiqué, permettant de voir combien d'échantillons de l'ensemble de données ont suivi chaque chemin de l'arbre.

L'arbre de décision semble bien structuré et montre comment différentes caractéristiques des fruits, comme la hauteur, la largeur et le score de couleur, sont utilisées pour les classer. Les nœuds avec un score de Gini de 0 sont particulièrement intéressants car ils indiquent une séparation parfaite des classes à ces points de l'arbre, ce qui est l'idéal dans la classification.

## 4.2 Utilisez maintenant une validation croisée en 5 blocs à l'aide de la fonction cross_val_score.

```
Accuracy of Dummy classifier on cross-validation: 0.30 (+/- 0.07)
Accuracy of Naive Bayes classifier on cross-validation: 0.80 (+/- 0.13)
Accuracy of Decision tree classifier on cross-validation: 0.83 (+/- 0.15)
Accuracy of Random Forest classifier on cross-validation: 0.93 (+/- 0.07)
```

```
Accuracy of Logistic regression classifier on cross-validation: 0.76 (+/- 0.22)
```

## Parmi les cinq méthodes précédemment testées, quelle est la plus performante ?

1. **Régression logistique :**
    - Précision moyenne : 0.76

- Écart-type : 0.22

2. **Classificateur Dummy :**
   - Précision moyenne : 0.30
   - Écart-type : 0.07
3. **Naive Bayes :**
   - Précision moyenne : 0.80
   - Écart-type : 0.13
4. **Decision Tree :**
   - Précision moyenne : 0.83
   - Écart-type : 0.15
5. **Random Forest :**
   - Précision moyenne : 0.93
   - Écart-type : 0.07

En se basant sur la précision moyenne, le modèle Random Forest est le plus performant parmi les cinq méthodes testées, avec une précision moyenne de 0.93.

## Le classement des méthodes est-il le même que lors du protocole d'évaluation précédent ?

**Protocole d'évaluation en utilisant l'ensemble de test:**

1. Random Forest (0.87)
2. Decision Tree (0.73)
3. Naive Bayes (0.67)
4. Logistic Regression (0.47)
5. Dummy Classifier (0.27)

**Résultats de la validation croisée :**

1. Random Forest (0.93)
2. Decision Tree (0.83)
3. Naive Bayes (0.80)

4. Logistic Regression (0.76)
5. Dummy Classifier (0.30)

Les classements sont identiques entre les deux protocoles.

### 4.3 Quel est le meilleur score obtenu parmi l'ensemble des valeurs de paramètres testées ? Quelles sont les meilleurs valeurs constatées sur le jeu de données étudiées ?

```
Best score: 0.815
Best parameters set:
        C: 0.01
        penalty: None
```

Le meilleure score est 0.815 pour C = 0.01.

## 5 Classement et discrétisation

### 5.1 Discrétisez les attributs numériques à l'aide d'intervalles de mêmes tailles. Appliquez les mêmes méthodes de classement que précédemment. Obtenez-vous les mêmes performances qu'avant la discrétisation ?

```
Accuracy of Dummy classifier on discretized data with eqsized_bins_: 0.30 (+/- 0.07)
Accuracy of Naive Bayes classifier on discretized data with eqsized_bins_: 0.78 (+/- 0.22)
Accuracy of Decision tree classifier on discretized data with eqsized_bins_: 0.83 (+/- 0.15)
Accuracy of Random Forest classifier on discretized data with eqsized_bins_: 0.85 (+/- 0.27)
Accuracy of Logistic regression classifier on discretized data with eqsized_bins_: 0.87 (+/- 0.17)
```

- **Logistic Regression :** La précision moyenne a augmenté après la discrétisation, passant de 0.47 à 0.87.
- **Random Forest :** La précision moyenne a légèrement diminué après la discrétisation, passant de 0.87 à 0.85.
- **Naive Bayes :** La précision moyenne a augmenté après la discrétisation, passant de 0.67 à 0.78.
- **Decision Tree :** La précision moyenne a augmenté après la discrétisation, passant de 0.73 à 0.83.
- **Dummy Classifier :** La précision moyenne a légèrement augmenté après la discrétisation, passant de 0.27 à 0.30.

En résumé, l'impact de la discrétisation varie en fonction du modèle. Dans ce cas, la Logistic Regression a montré une amélioration significative après la discrétisation, tandis que d'autres modèles ont montré des variations moins prononcées.

### 5.2 Mêmes questions que précédemment en utilisant cette fois-ci une discrétisation à l'aide d'intervalles de mêmes effectifs.

```
Accuracy of Dummy classifier on discretized data with eqintervaled_bins_: 0.30 (+/- 0.07)
Accuracy of Naive Bayes classifier on discretized data with eqintervaled_bins_: 0.83 (+/- 0.15)
Accuracy of Decision tree classifier on discretized data with eqintervaled_bins_: 0.71 (+/- 0.22)
Accuracy of Random Forest classifier on discretized data with eqintervaled_bins_: 0.88 (+/- 0.09)
Accuracy of Logistic regression classifier on discretized data with eqintervaled_bins_: 0.90 (+/- 0.12)
```

- **Logistic Regression :** La précision moyenne a augmenté après la discrétisation, passant de 0.47 à 0.90.
- **Random Forest :** La précision moyenne a légèrement augmenté après la discrétisation, passant de 0.87 à 0.88.
- **Naive Bayes :** La précision moyenne a légèrement augmenté après la discrétisation, passant de 0.67 à 0.71.
- **Decision Tree :** La précision moyenne a augmenté après la discrétisation, passant de 0.73 à 0.83.
- **Dummy Classifier :** La précision moyenne a légèrement augmenté après la discrétisation, passant de 0.27 à 0.30.

Les résultats sont assez similaire que lors de la discrétisation précédente.

## 6 ACP et sélection de variables

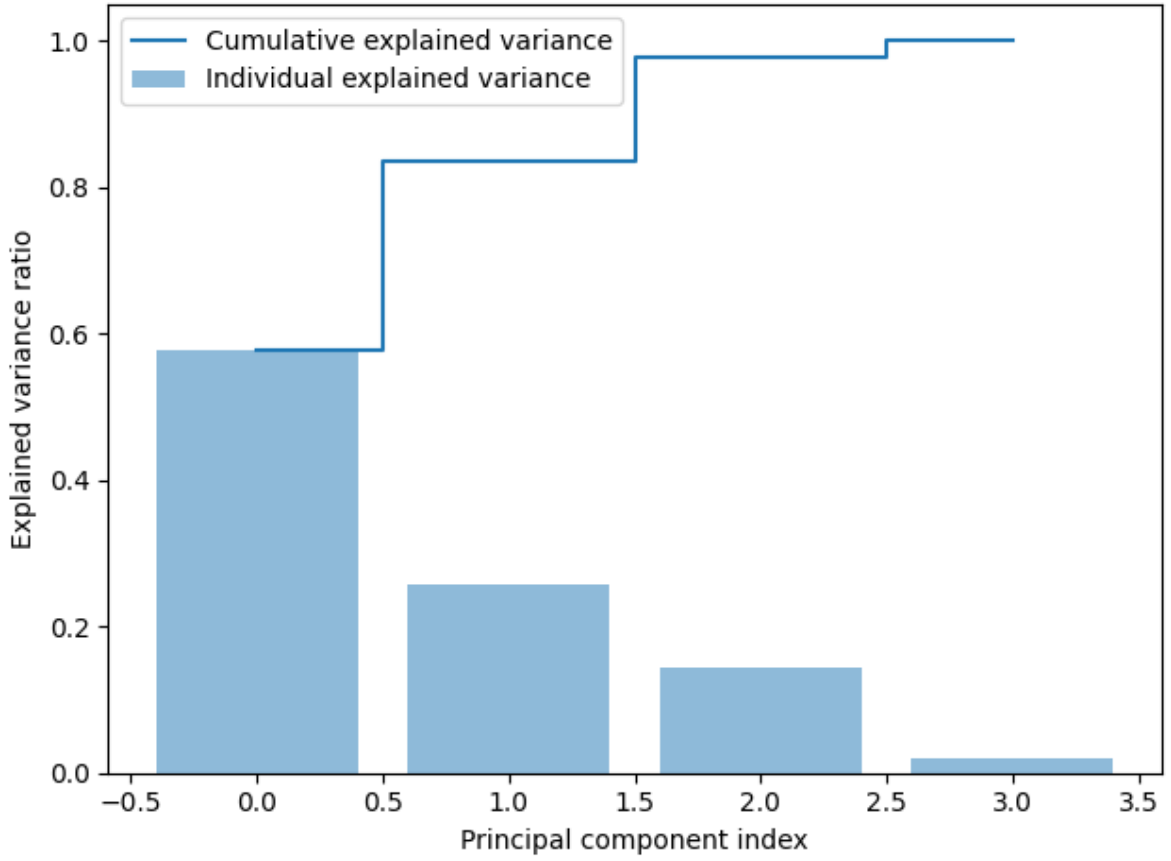
### 6.1 Quelle sont les valeurs propres des facteurs de l'ACP ?

```
acp.n_components_ :  4


exp_var_pca
[0.57845054 0.25650171 0.14402362 0.02102413]


acp.explained_variance_
[2.35369528 1.04369662 0.58602716 0.08554646]
[[ 0.6264132   0.57799922  0.49626726 -0.16505215]
 [ 0.19538115  0.25480203 -0.23856702  0.91650854]
 [ 0.1010637   0.47250483 -0.79781474 -0.36057865]
 [ 0.74780936 -0.61459907 -0.2455529  -0.05246823]]
```

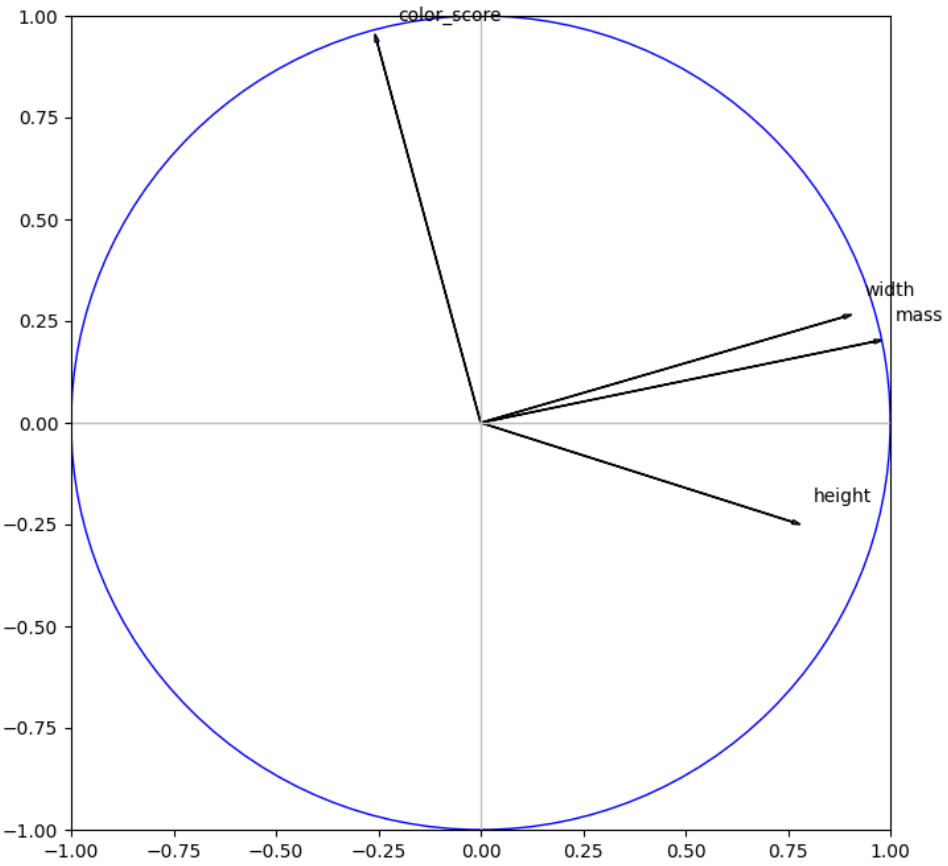
Les valeurs propres (explained_variance_) pour chaque composante principale sont les suivantes :

1. Première composante principale : 2.35369528
2. Deuxième composante principale : 1.04369662
3. Troisième composante principale : 0.58602716
4. Quatrième composante principale : 0.08554646

## Dessinez le graphe montrant l'évolution des valeurs propres en fonction de leur rang.

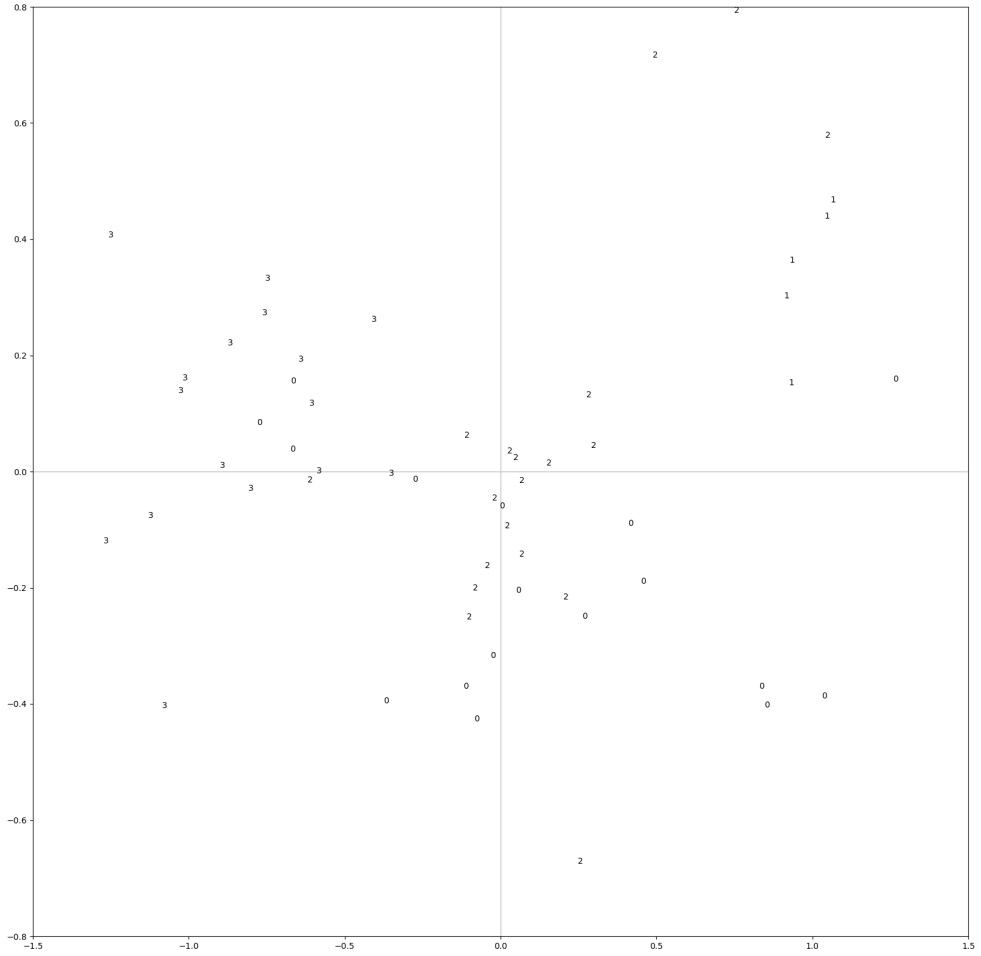


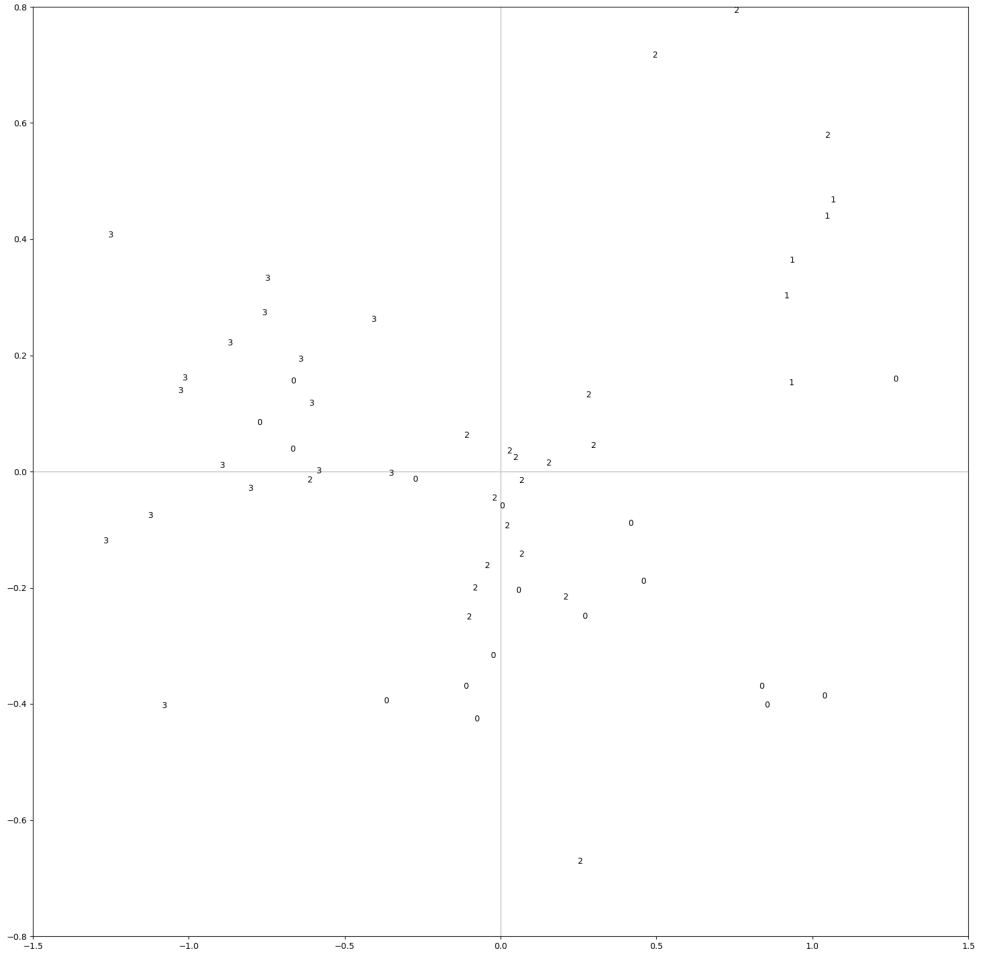

## Quelles variables originelles sont corrélées avec chacun des nouveaux facteurs ?




- mass est très corrélée au premier facteur.
- height et width sont corrélées au premier facteur mais moins que mass.
- color_score par contre est fortement corrélée au deuxième facteur.

## Tracez les instances dans le plan principal constitué des deux premiers facteurs.

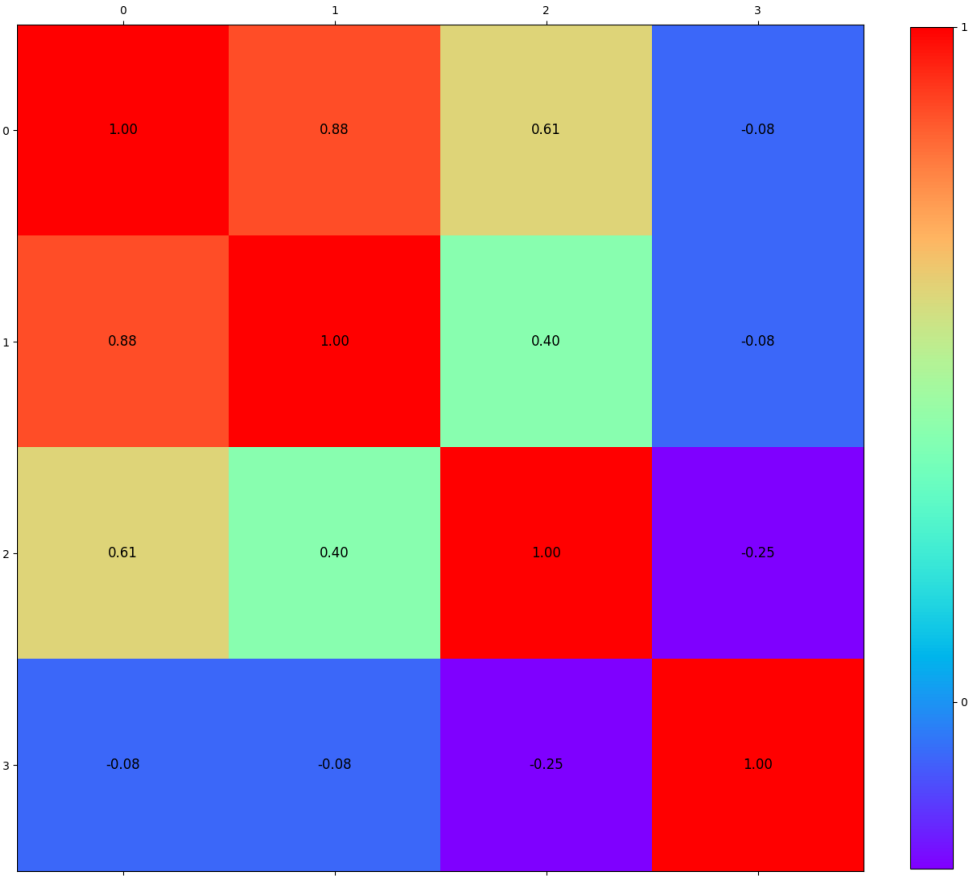

0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5



## Les classes des fruits sont-ils bien séparés dans ce nouvel espace ?

Dans les graphes ci-dessus chaque nombre correspond à une classe de fruits.

Les classes ne sont pas bien séparés dans ce nouvel espace, elles sont mélangé comme on peut le voir sur les schémas.




## 6.2 Calculez les performances en validation croisée des cinq méthodes de classement utilisées dans la section 4 sur les données, en ne considérant non plus les quatre attributs initiaux mais les deux facteurs principaux de l'ACP. Obtenez-vous les mêmes performances qu'en utilisant deux attributs du fichier original normalisé (par exemple les deux premiers attributs)?

```
*** Results for first 2 factors of ACP ***
Accuracy of Dummy classifier on ACP (2 factors): 0.30 (+/- 0.07)
Accuracy of Naive Bayes classifier on ACP (2 factors): 0.71 (+/- 0.40)
Accuracy of Decision tree classifier on ACP (2 factors): 0.80 (+/- 0.17)
Accuracy of Random Forest classifier on ACP (2 factors): 0.82 (+/- 0.12)
Accuracy of Logistic regression classifier on ACP (2 factors): 0.53 (+/- 0.35)
*** Results for first 2 original variables ACP ***
Accuracy of Dummy classifier on original features: 0.30 (+/- 0.07)
Accuracy of Naive Bayes classifier on original features: 0.80 (+/- 0.13)
Accuracy of Decision tree classifier on original features: 0.90 (+/- 0.16)
Accuracy of Random Forest classifier on original features: 0.95 (+/- 0.08)
Accuracy of Logistic regression classifier on original features: 0.75 (+/- 0.18)
```

Comparons :

- Dummy : ACP (0,30) vs. Original (0,30)
- Naive Bayes : ACP (0,71) vs. Original (0,80)
- Decision Tree : ACP (0,80) vs. Original (0,90)
- Random Forest : ACP (0,82) vs. Original (0,95)
- Logistic Regression : ACP (0,53) vs. Original (0,75)

Les performances ne sont pas les mêmes. Les performances des classificateurs varient généralement entre les deux ensembles de caractéristiques. Naive Bayes montre une légère diminution des performances lors de l'utilisation des facteurs de l'ACP, ainsi que les classificateurs Decision Tree et Random Forest connaissent également une diminution. Logistic Regression, en revanche, connaît une chute significative des performances lors de l'utilisation des facteurs de l'ACP.

## 6.3 Dans la question précédente, les deux attributs ont été choisis au hasard. scikit-learn met à disposition plusieurs méthodes de sélection de variables. Utilisez le sélectionneur SelectKBest avec un critère basé sur l'information mutuelle, pour ne retenir que les deux entrées. Quels sont les deux attributs jugés les plus importants par cette méthode ?

```
Selected features:
['height', 'color_score']
```

Les attributs sont height et color_score.

## Calculez les performances en validation croisée des quatre classieurs utilisés sur les deux meilleures variables retenues et

## comparez les résultats précédents calculés sur les deux premiers facteurs de l'ACP.

```
*** Results for the 2 attributes selected with mutual_info_classif ***
Accuracy of Dummy classifier on selected features: 0.30 (+/- 0.07)
Accuracy of Naive Bayes classifier on selected features: 0.83 (+/- 0.11)
Accuracy of Decision tree classifier on selected features: 0.83 (+/- 0.21)
Accuracy of Random Forest classifier on selected features: 0.78 (+/- 0.34)
Accuracy of Logistic regression classifier on selected features: 0.44 (+/- 0.15)
```

**Comparaison avec les résultats sur les deux premiers facteurs de l'ACP :**

- Dummy (Information mutuelle: 0.30 vs. ACP: 0.30)
- Naive Bayes (Information mutuelle: 0.83 vs. ACP: 0.71)
- Decision Tree (Information mutuelle: 0.83 vs. ACP: 0.80)
- Random Forest (Information mutuelle: 0.78 vs. ACP: 0.82)
- Logistic Regression (Information mutuelle: 0.44 vs. ACP: 0.53)

En comparant les résultats, on peut observer que les performances des classificateurs sur les deux attributs sélectionnés par SelectKBest avec information mutuelle sont généralement meilleures ou équivalentes par rapport aux performances sur les deux premiers facteurs de l'ACP.