heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Machine Learning

Practical work 05 – Unsupervised Learning

BURGBACHER LIONEL & JAQUET DAVID

Table des matières

1 Introduction

Le but de ce laboratoire est de tester une technique de machine learning non-supervisée. Dans une première partie, nous allons analyser les résultats par cet algorithme sur la classification de vin. Dans une seconde étape, il s'agit d'utiliser le même algorithme sur des images avec trois méthodes d'extractions différentes.

2 Clustering of wine data

Les *notebooks* contiennent les réponses aux différentes questions, les codes nécessaires à la réalisation des différents résultats présentés ci-dessous ainsi que certaines explications sur les sorties générées.

Pour calculer *l'accuracy*, nous avons d'abord calculé le nombre de bouteilles par classe. Ensuite, nous appliquons l'algorithme et nous utilisons le fait que nous savons que la matrice de base est correcte.

```
Classes
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

De ce fait, il suffit de calculer les résultats obtenus dans cet intervalle pour chaque classe de vin. On prend donc le nombre qui apparaît le plus de fois sur cette longueur et on calcule le pourcentage de vin trouvé correctement. On applique cela pour les 3 classes. On le fait 10 fois pour avoir un résultat cohérent et ne pas nous retrouver avec des valeurs particulièrement bonnes ou, au contraire, particulièrement mauvaises.

Dans le premier cas appliqué, nous n'avons pas de normalisation et toutes les *features* sont utilisées.

```
Last execution:
 [1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 2 2 1 1 2 1 1 1 1 1 1 2 2
 1 1 2 2 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 0 2 0 2 0 0 2 0 0 2 2 2 0 0 1
 2 0 0 0 2 0 0 2 2 0 0 0 0 0 2 2 0 0 0 0 0 2 2 0 2 0 2 0 0 0 2 0 0 0 0 2 0
 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 2 0 0 2 2 2 2 0 0 0 2 2 0 0 2 2 0 2
 2 0 0 0 0 2 2 2 0 2 2 2 0 2 0 2 2 0 2 2 2 2 0 0 2 2 2 2 2 0]

Accuracy is 69.60%
```

On obtient un résultat de 69.60% ce qui n'est pas très bon.

Dans le deuxième, on normalise les données ce qui minimise la variance. On constate une nette amélioration sur le résultat qui est de 95.79%.

```
Last execution:
 [2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 1 1 0 0 0 0 0 0 1 0 1 0 0 2
 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

Accuracy is 95.79%
```

Dans cette partie, nous avons enlevé quelques *features*. On constate que le choix ici n'a pas d'impact, on pourrait donc imaginer que les 10 premiers critères n'ont pas vraiment d'influence sur la classification.

```
Last execution:
 [1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 2 2 1 1 2 1 1 1 1 1 1 2 2
 1 1 2 2 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 2 0 2 0 0 2 0 0 2 2 2 0 0 1
 2 0 0 0 2 0 0 2 2 0 0 0 0 0 2 2 0 0 0 0 0 2 2 0 2 0 2 0 0 0 2 0 0 0 0 2 0
 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 2 0 0 2 2 2 2 0 0 0 2 2 0 0 2 2 0 2
 2 0 0 0 0 2 2 2 0 2 2 2 0 2 0 2 2 0 2 2 2 2 0 0 2 2 2 2 2 0]

Accuracy is 69.60%
```

Ici, nous appliquons la suppression de *features* et la normalisation. Nous arrivons à un résultat de 93.60%, on a donc que la normalisation augmente à nouveau considérablement la justesse des données, mais que la suppression de *features* ne fait que peu baisser le résultat final.

```
Last execution:
 [1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 2 2 2 2 2 2 0 2 0 2 2 1
 1 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 0 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

Accuracy is 93.60%
```

Dans le dernier cas, nous avons testé la suppression d'une *feature* et constaté que le résultat était le moins bon. On en déduit donc que cette *feature* permet une bonne séparation des données.

```
Last execution:
 [1 2 2 2 1 2 2 1 2 2 2 2 0 0 2 2 1 1 2 1 1 2 2 2 2 1 0 0 2 2 2 2 2 1 2 2 2
 2 2 1 1 0 2 2 2 2 2 2 2 2 2 0 0 2 1 1 1 1 2 2 0 2 2 0 0 2 2 0 0 2 1 2 0 0 1
 2 2 0 2 1 2 0 0 0 0 0 2 0 0 0 0 0 0 0 0 2 1 1 0 0 0 2 0 2 0 0 0 0 0 0 0 2
 0 2 0 0 0 0 2 0 0 2 1 2 0 0 0 0 0 0 0 1 2 2 2 0 0 0 2 0 2 2 0 2 0 2 2 0 0
 0 2 1 2 1 2 2 0 0 2 2 0 0 2 2 2 0 0 2 0 2 2 2 0 0 2 2 1 1 2]

Accuracy is 59.50%
```

3 Expériences

3.1 Définition des méthodes

Nous avons analysé chacune des méthodes utilisées pour générer les histogrammes. Le code fourni dispose donc de trois méthodes comme décrites ci-dessous. Ces méthodes permettront de générer des histogrammes contenant différentes features qui seront utilisés pour classifier les images. Chacune de ses méthodes appartient à la classe `ImageFeatureExtractor`.

Les méthodes sont donc les suivantes :

- Niveau de gris
  - Description : Cette méthode va transformer l'image en niveau de gris et crée ensuite un histogramme de 10 *features* de l'image. Les valeurs des *features* sont comprises entre 0 et 1.
  - Nom de la méthode : `extract_histogram`
- Hue
  - Description : Cette méthode va transformer en un premier temps l'image en *HSV (Hue, Saturation, Value)* et crée ensuite un histogramme de 10 *features* de l'image. Les valeurs des *features* sont comprises entre 0 et 1.
  - Nom de la méthode : `extract_hue_histogram`
- Couleurs :
  - Description : Cette méthode ne va pas transformer l'image, mais va directement récupérer l'intensité des couleurs *RGB (Red, Green, Blue)* de l'image et crée ensuite un histogramme de 30 *features* de l'image. Les valeurs des *features* sont comprises entre 0 et 255.
  - Nom de la méthode : `extract_color_histogram`

3.2 Expériences

3.2.1 Procédure

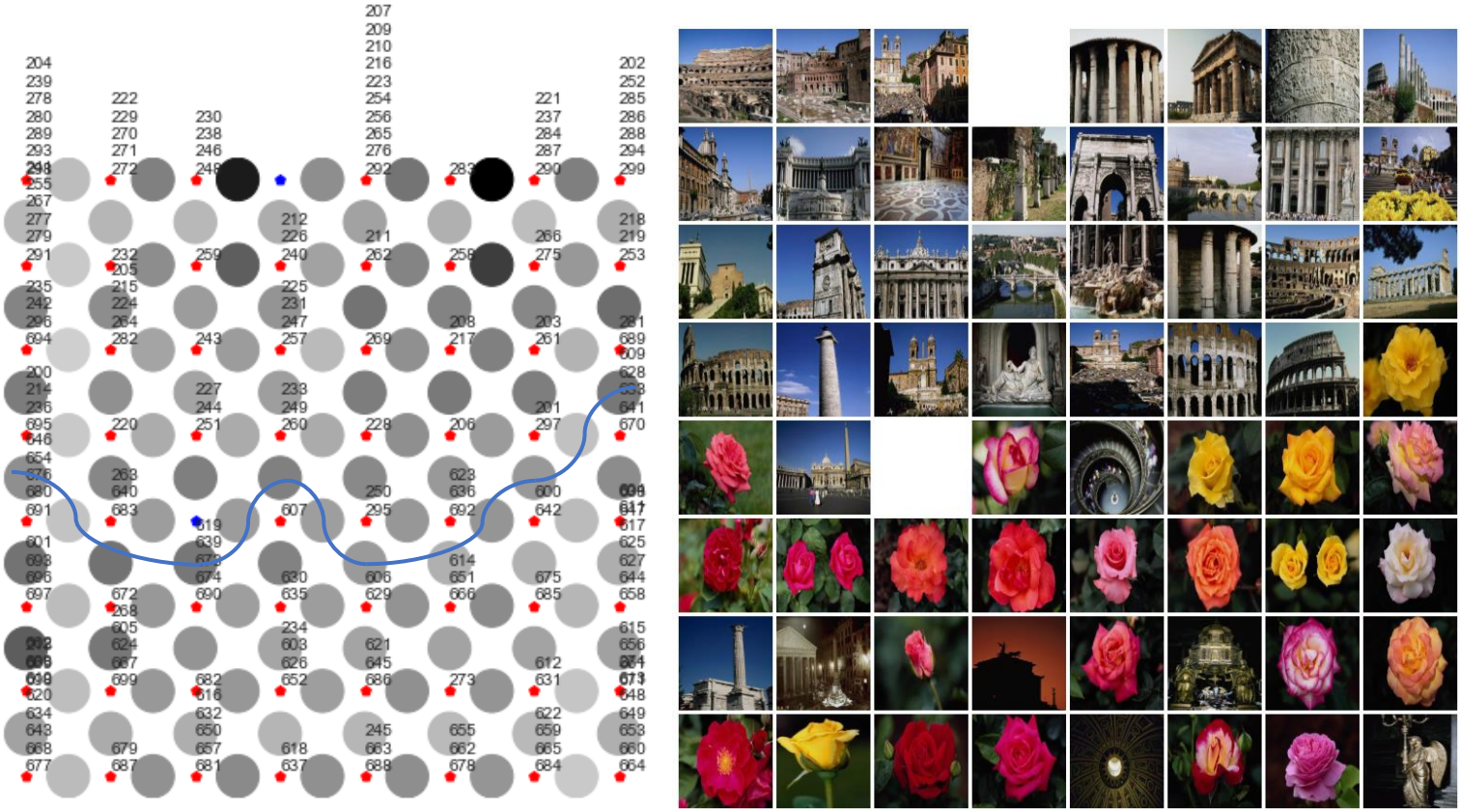
Pour avoir des résultats comparables entre chaque expérience, nous avons choisi de générer des *U-matrix* en faisant appel à chacune des méthodes citées précédemment. De cette manière, nous pouvons avoir quelle est la méthode la plus adéquate pour l'expérience souhaitée.

De plus, chacun des tests a été effectué sur un minimum de 10'000 itérations. Le choix de ce nombre d'itérations a été fait grâce à un message laissé sur Microsoft Teams. Dans ce message, M. Perez-Uribe nous a fait comprendre qu'il ne fallait pas hésiter à itérer entre 10 et 50 fois le nombre d'images au minimum. Nous avons donc multiplié le nombre d'images traité dans l'expérience par 50 pour obtenir notre nombre d'itérations.

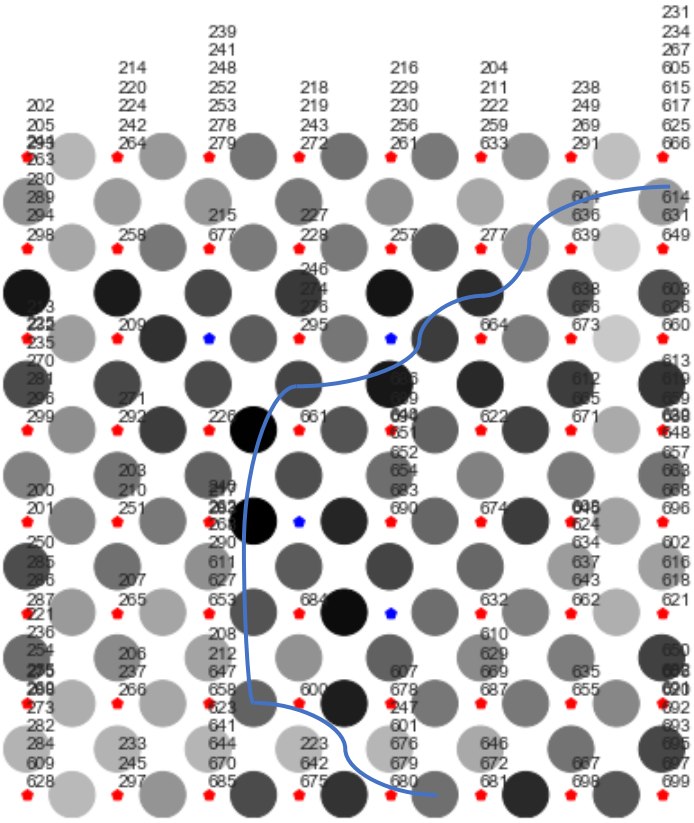
3.2.2 Première expérience : Monuments et fleurs

Pour cette première expérience, nous souhaitions comparer les images contenant des monuments et celles contenant les fleurs. Les images des monuments sont celles comprises dans l'intervalle entre 200 et 299 tandis que les images des fleurs sont comprises dans l'intervalle entre 600 et 699.

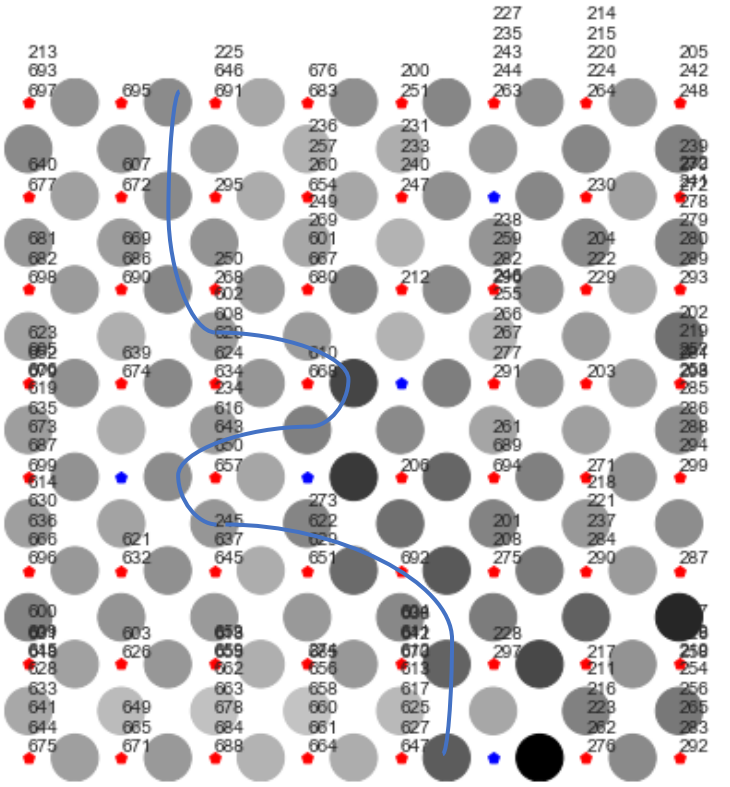
Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les nuances de gris. On peut s'apercevoir sur l'image de gauche que la séparation des clusters n'est pas très voyante. Cependant, en regardant l'image de droite, on remarque que, de manière générale, la séparation est faite. Néanmoins, certains monuments se retrouvent au milieu des fleurs. Les clusters ont été séparés par une ligne bleue.



Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les teintes. On peut s'apercevoir sur l'image de gauche que la séparation des clusters est plus performante que dans la première partie. Cette fois, on constate que la séparation est bien faite excepté pour 2 images, l'algorithme a mieux fonctionné.



Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les couleurs. On constate que, comme pour la première expérience, la séparation des clusters n'est pas très voyante. Nous avons une séparation plus prononcée sur les images, mais on remarque aussi que quelques images ont été mal classées.

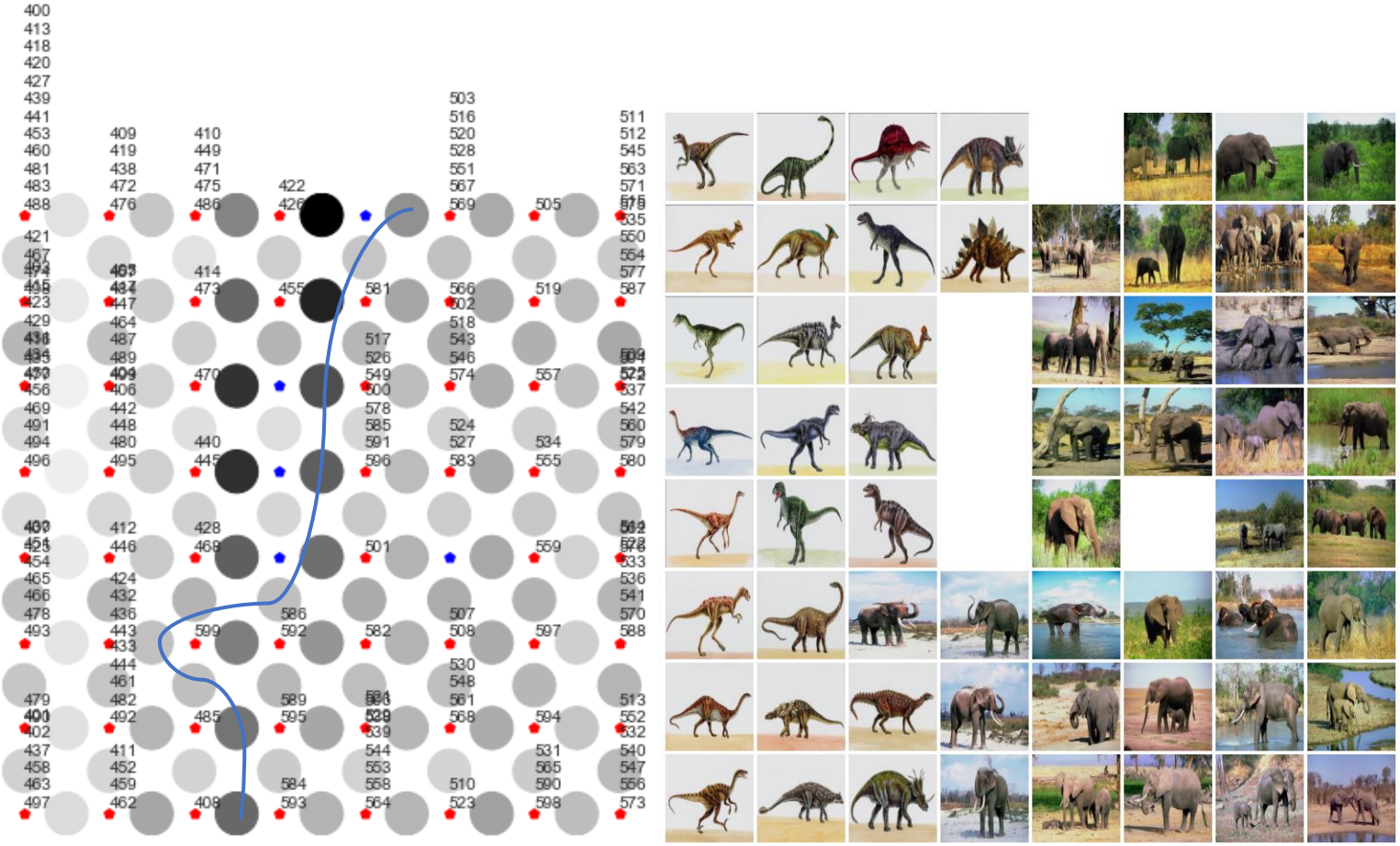


Avec nos différents tests, on s'aperçoit que la méthode la plus adéquate pour cette expérience est celle utilisant les teintes. En effet, la méthode travaillant avec les teintes génère la U-matrix avec la meilleure séparation. C'est également la méthode avec le moins d'erreurs constatées.

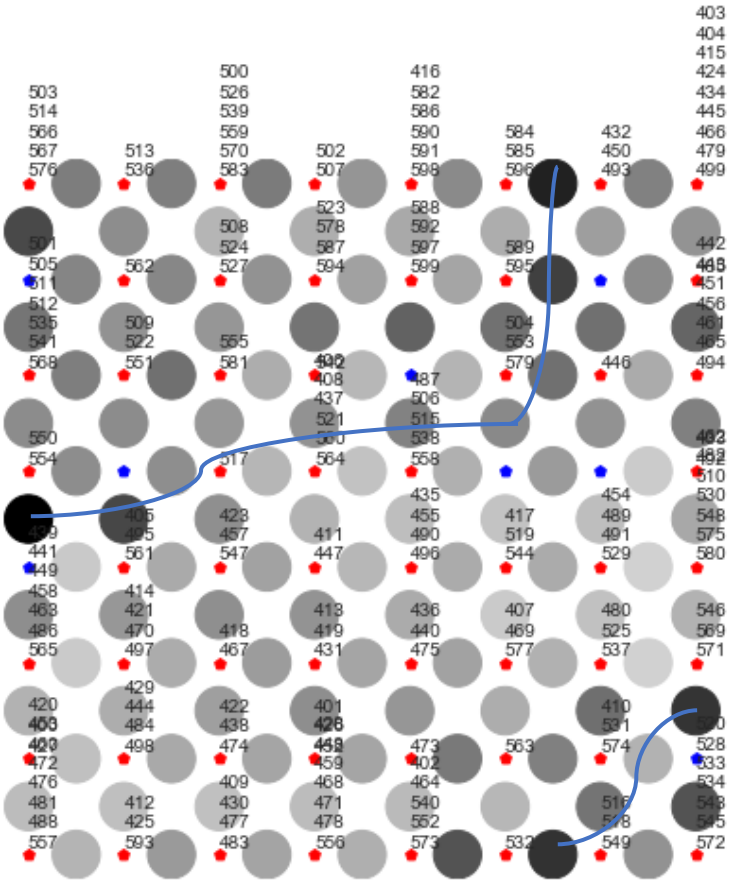
3.2.3 Seconde expérience : Dinosaures et éléphants

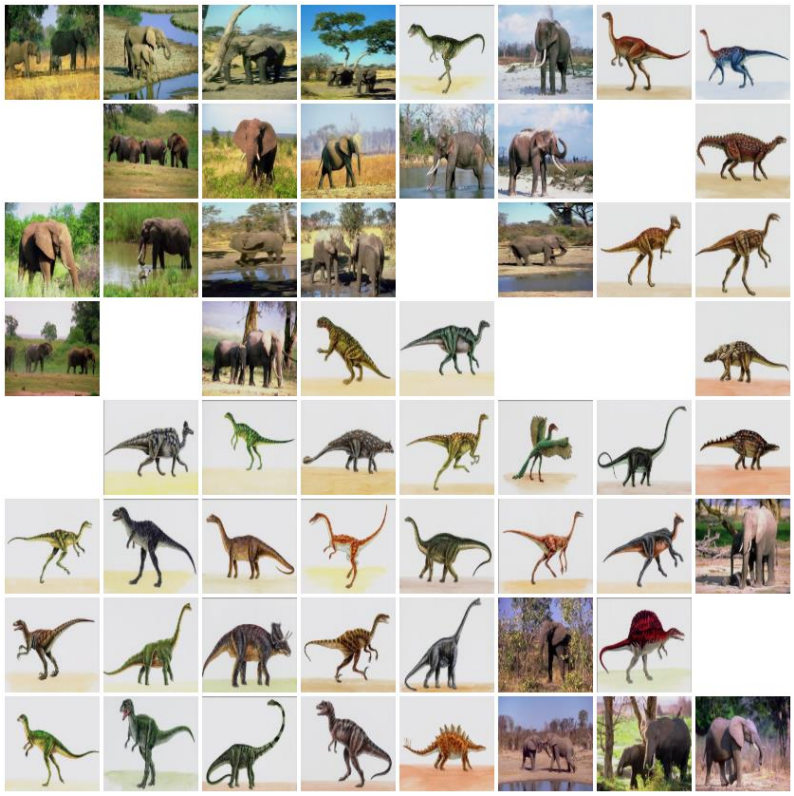
Pour cette seconde expérience, nous souhaitions comparer les images contenant des dinosaures et celles contenant des éléphants. Les images des dinosaures sont celles comprises dans l'intervalle entre 400 et 499 tandis que les images des éléphants sont comprises dans l'intervalle entre 500 et 599. Nous avons choisi ces groupes d'images, car nous avons remarqué qu'elles étaient très différentes. Nous voulions donc nous assurer que l'algorithme était fonctionnel et qu'il nous permettait d'obtenir une séparation claire.

Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les nuances de gris. On peut s'apercevoir sur l'image de gauche que la séparation des clusters est bien voyante. De plus, en regardant l'image de droite, on remarque que la séparation est faite correctement. Les clusters ont été séparés par une ligne bleue.

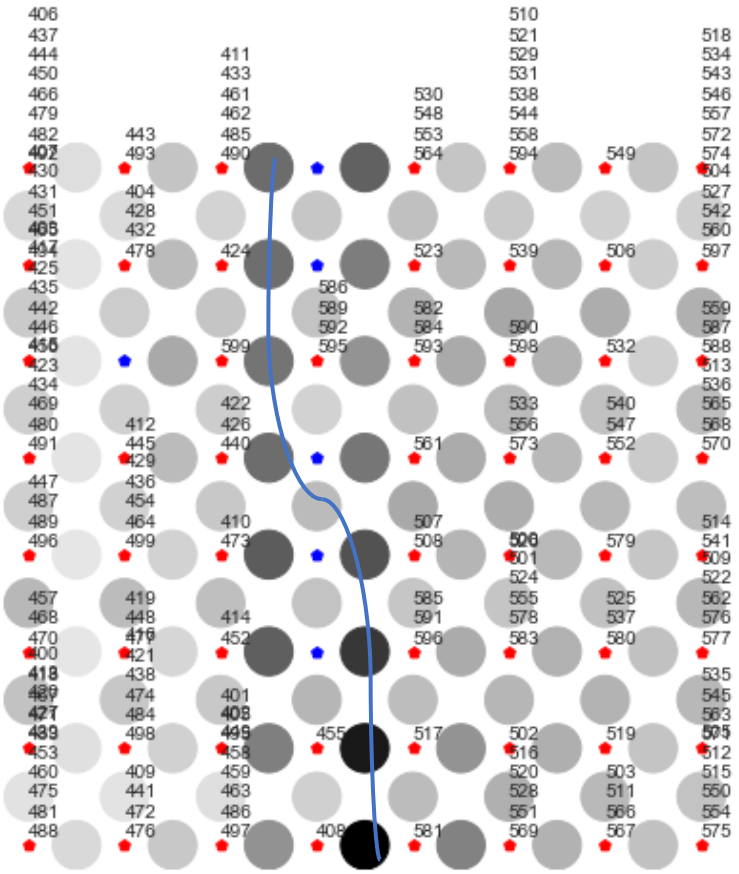


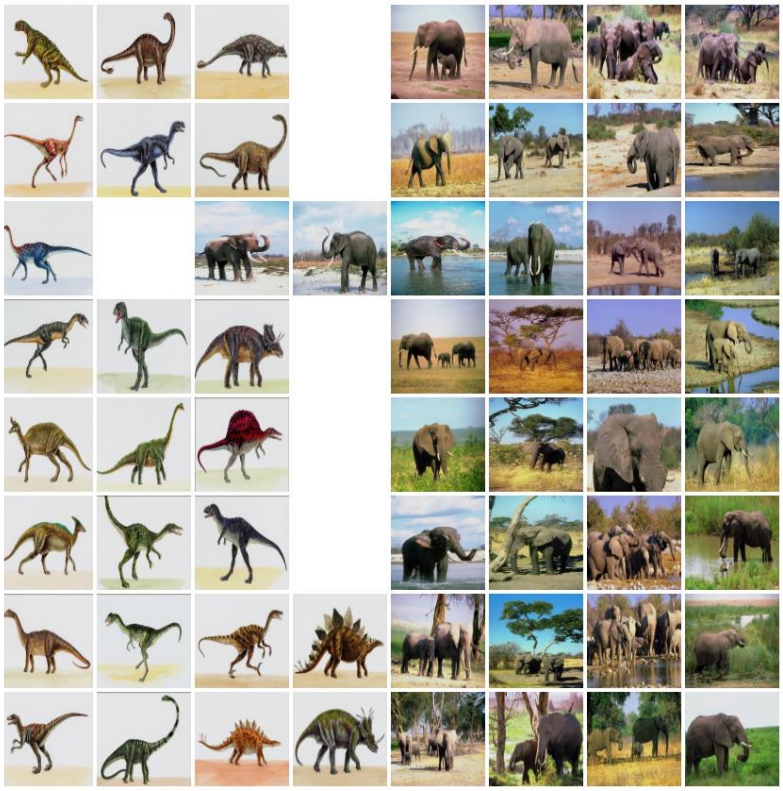
Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les teintes. On remarque qu'il y a 3 clusters différents alors que seulement 2 auraient dû être trouvés. Cela vient du fait qu'une trop grosse différence de teinte existe sur des images de même type.





Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les couleurs. On constate une séparation très claire entre les clusters, autant sur l'image de gauche que sur celle de droite.



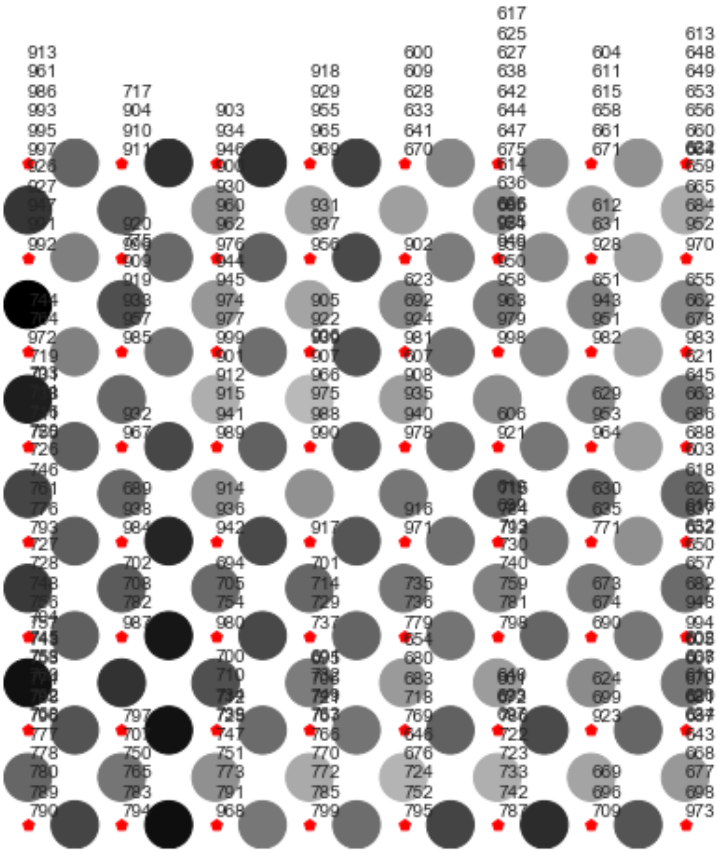


Avec nos différents tests, on s'aperçoit que nous avons deux méthodes adéquates pour cette expérience. Il s'agit de la méthode utilisant les nuances de gris et celle utilisant les couleurs *RGB*. En effet, ces méthodes génèrent des U-matrix avec d'excellentes séparations. Ce sont également les méthodes avec le moins d'erreurs constatées. Notre algorithme fonctionne donc comme espéré excepté pour la méthode utilisant les teintes où nous imaginions une séparation plus flagrante et, par conséquent, un meilleur résultat.

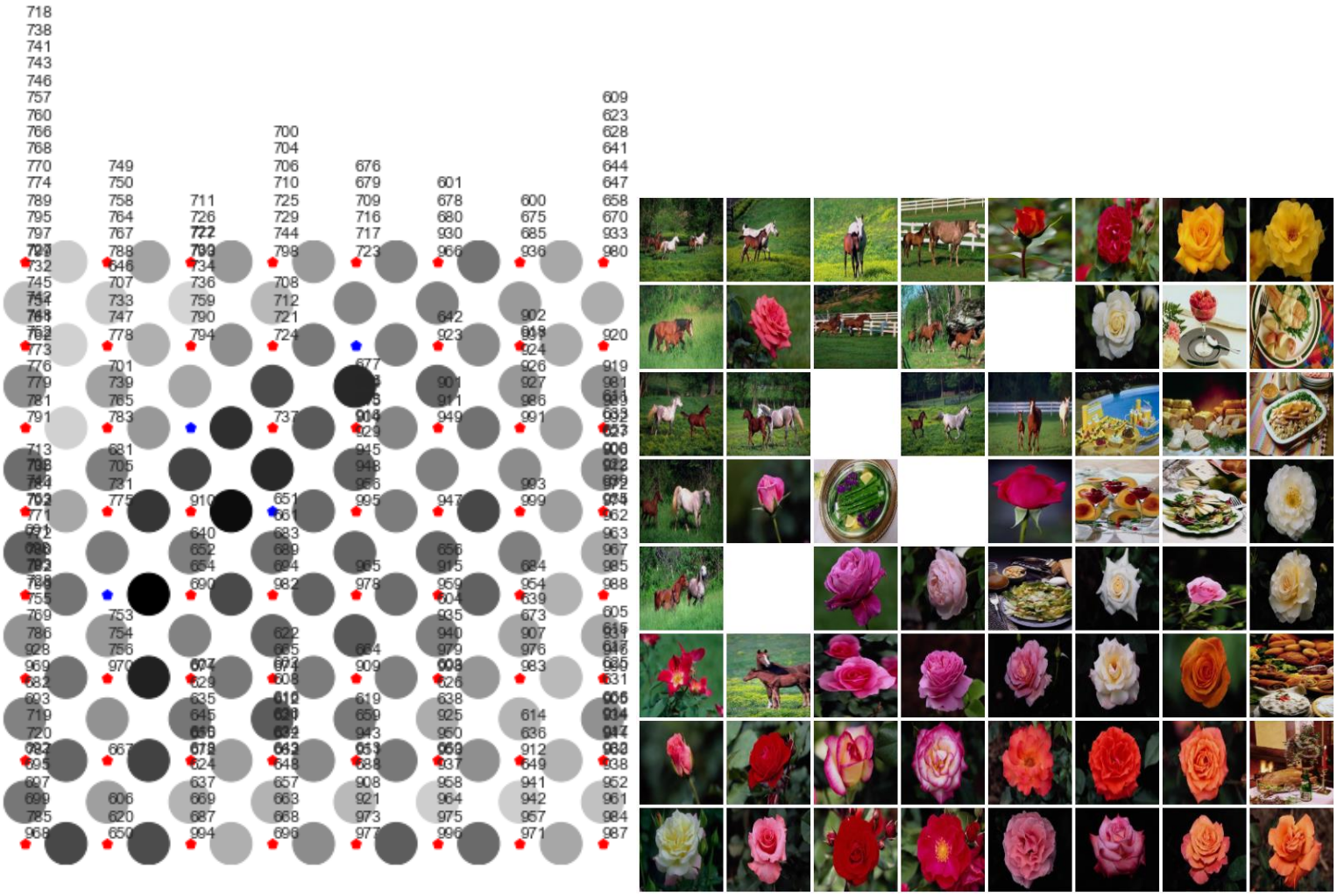
3.2.4 Troisième expérience : Fleurs, chevaux et nourriture

Pour cette troisième et dernière expérience, nous souhaitions comparer les images contenant des fleurs, celles contenant des chevaux et celles contenant de la nourriture. Les images des fleurs sont celles comprises dans l'intervalle entre 600 et 699, les images des chevaux sont comprises dans l'intervalle entre 700 et 799 et les images de nourriture sont comprises dans l'intervalle entre 900 et 999. Nous avons choisi ces groupes d'images, car nous souhaitions tester l'algorithme avec trois groupes d'images différents.

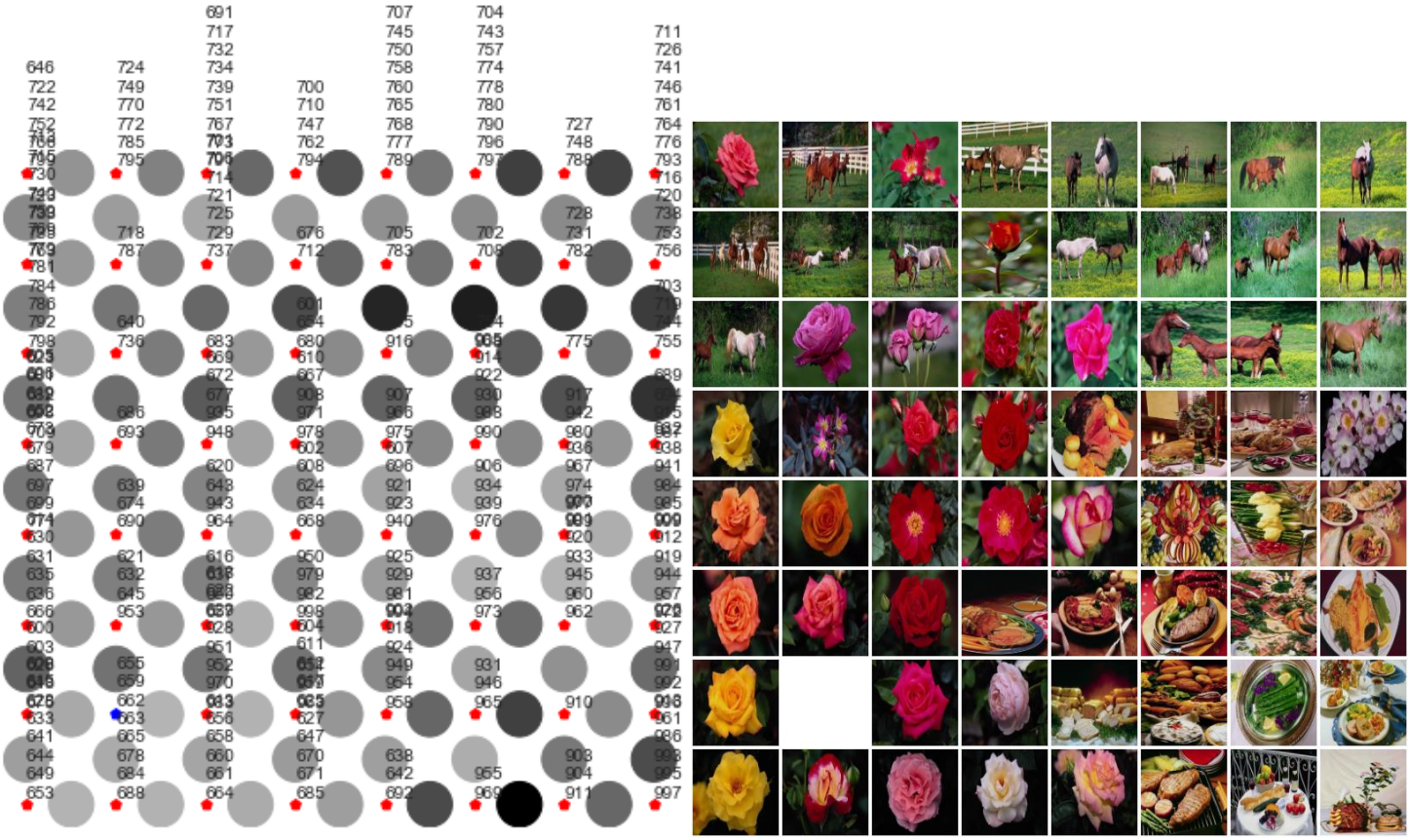
Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les nuances de gris. On peut s'apercevoir sur l'image de gauche que la séparation des clusters est très peu voyante. De plus, en regardant l'image de droite, on remarque que les fleurs ont été placées sur la droite, et ce malgré des fleurs mal classées. Cependant, on remarque beaucoup d'erreurs entre les chevaux et la nourriture.



Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les teintes. On peut s'apercevoir sur l'image de gauche que la séparation des clusters est voyante. Cependant, nous avons l'impression d'obtenir seulement deux clusters. De plus, en regardant l'image de droite, on remarque qu'une partie des fleurs a été placée sur le bas de la matrice, et ce malgré des fleurs mal classées. Cependant, comme avec la méthode précédente, on remarque beaucoup d'erreurs entre les chevaux et la nourriture.



Les images ci-dessous montrent le résultat en utilisant l'histogramme généré par la méthode travaillant avec les couleurs *RGB*. On peut s'apercevoir sur l'image de gauche que la séparation des clusters n'est pas très démarquée. Cependant, en regardant l'image de droite, on remarque que nous avons une séparation correcte. En effet, nous avons un groupe de fleurs en bas à gauche, un groupe d'images de nourriture se situe en bas à droite tandis que les chevaux occupent la partie supérieure de la matrice.



En analysant nos résultats avec les différentes méthodes, on remarque que nous obtenons beaucoup d'erreurs. Cependant, la méthode nous offrant le meilleur résultat reste la celle qui utilise l'histogramme *RGB*.

4 Difficultés

Nous n'avons pas rencontré de difficulté particulière durant ce laboratoire. Le code fourni était assez clair et permettait une bonne compréhension de ce qui était appliqué ici. La difficulté se trouvait plus dans la compréhension du laboratoire. Nous avons pu trouver réponse à beaucoup de nos questions sur Microsoft Teams.

5 Conclusion

Ce laboratoire a permis de bien mettre en application les principes vus en cours. Il a permis de clarifier certaines questions et de montrer les limites de cet algorithme d'apprentissage non-supervisé.