NoSQL avec MongoDB

Cours 7 - Analyse des données

Table des matières

Utilité de l'analyse des données

L'analyse des données est une étape préalable, ou du moins fortement recommandée de faire avant toute chose, pour avoir un aperçu de plusieurs facteurs importants pouvant impacter l'entraînement d'un modèle d'Intelligence Artificielle (AI).

Par la suite, les décisions importantes et le nettoyage des données sont **beaucoup plus faciles** à accomplir.

Utilité de l'analyse des données

Voici une liste simple* des opérations d'analyse que nous allons voir :

- Description du Dataframe.
- Groupement des données par attribut.
- Visualisation graphique avec Matplotlib.

*il existe beaucoup plus d'opérations.

Différence entre l'analyse et le nettoyage des données

Avantages de Python :

- Facile à apprendre
- Simple et rapide de programmer sur des "deadlines" courts
- Open-source
- Compatible avec plusieurs domaines (Web, AI, etc.)

Description du Dataframe

La description du Dataframe Pandas permet d'avoir un tableau complet de la disparité statistique des données pour toutes les colonnes (attributs) de l'objet en question.

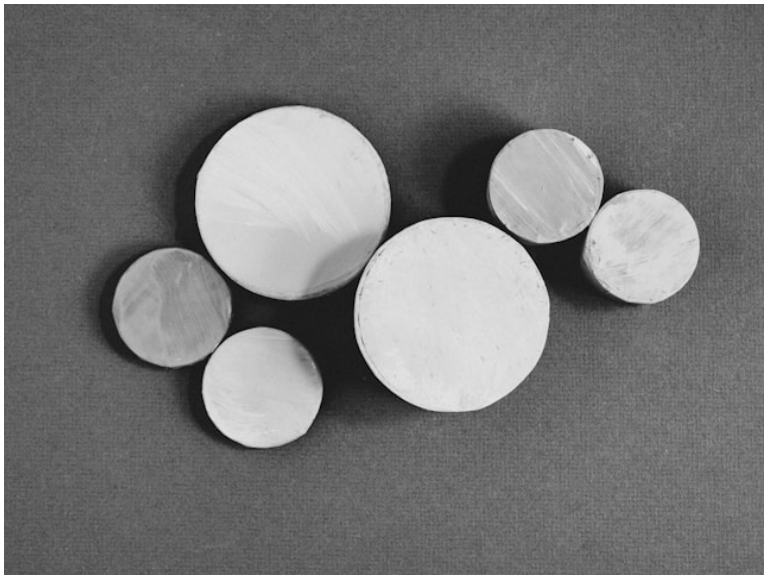
Très utile pour voir les **débalancements des classes**, les erreurs dans le nettoyage des données, et voir la moyenne et la déviation standard (à quel point la valeur est proche de la moyenne).

	#	Total	HP	...	Defense
count	800.000000	800.00000	800.000000	...	800.000000
mean	362.813750	435.10250	69.258750	...	73.842500
std	208.343798	119.96304	25.534669	...	31.183501
min	1.000000	180.00000	1.000000	...	5.000000
25%	184.750000	330.00000	50.000000	...	50.000000
50%	364.500000	450.00000	65.000000	...	70.000000
75%	539.250000	515.00000	80.000000	...	90.000000
max	721.000000	780.00000	255.000000	...	230.000000

Description du Dataframe

```
1  # Steve Levesque, All rights reserved where applicable
2  import pandas as pd
3  from pymongo import MongoClient
4
5  client = MongoClient()
6  db = client.admin
7  collection = db.Vegetables
8  documents = collection.find()
9
10 df = pd.DataFrame(documents)
11
12 # Descriptive Statistics: Show stats of all columns of the dataframe.
13 print(df.describe())
```

Groupement des données par attribut



Les groupements permettent d'avoir les données rassemblées par grappe ("clusters") vis-à-vis une caractéristique commune.

Ensuite, les grappes peuvent avoir des opérations statistiques appliquées de manière isolée en silo. Relativement utile pour avoir de l'information pure sur un sous-ensemble de données.

Groupement des données par attribut

Nous allons voir comment les données sont regroupées par un attribut avec valeurs semblables entre les entrées.

Par exemple, il est possible de grouper les données par valeur identique de “calcium (mg)”.

On pourrait aussi rajouter une catégorie plus distinctive, comme la couleur, pour grouper les légumes par couleur. Par contre, ceci demande qu’on fasse une modification de la base de données. Essayez-le de votre côté.

Groupement des données par attribut

```
1  # Steve Levesque, All rights reserved where applicable
2  import pandas as pd
3  from pymongo import MongoClient
4
5  client = MongoClient()
6  db = client.admin
7  collection = db.Vegetables
8  documents = collection.find()
9
10 df = pd.DataFrame(documents)
11
12 # Grouping and Aggregation :
13 # Vegetable by energy, max of each group's water (g) and protein (g).
14 print(len(df))
15 grouped = df.groupby('calcium (mg)')
16 print(len(grouped))
17 print(grouped['water (g)'].max())
18 print(grouped['protein (g)'].max())
```

Visualisation avec Matplotlib

Il est important de ne pas être trop confiant si un doute occupe notre esprit.

Pour être complètement sûre de la topologie des données, une image vaut 1000 mots !

Les graphiques avec Matplotlib permettent de représenter visuellement les données d'un Dataframe directement à même son objet.

Visualisation avec Matplotlib

Listing: https://stackoverflow.com/questions/15910019/annotate-data-points-while-plotting-from-pandas-dataframe

```
1  # Steve Levesque, All rights reserved where applicable
2  import matplotlib.pyplot as plt, pandas as pd
3  from pymongo import MongoClient
4
5  client = MongoClient()
6  db = client.admin
7  collection = db.Vegetables
8  documents = collection.find()
9
10 df = pd.DataFrame(documents)
11
12 # Data Visualization: All Vegetable scattered by protein (g) and water (g).
13 p_g = 'protein (g)'
14 print(df[p_g].describe())
15 fig, ax = plt.subplots()
16 df.plot(kind='scatter', x='water (g)', y=p_g, color='red', ax=ax)
17 for k, v in df.iterrows():
18     ax.annotate(v['name'] + ", " + str(v[p_g]), (v['water (g)'], v[p_g]))
19 plt.show()
```

Visualisation avec Matplotlib

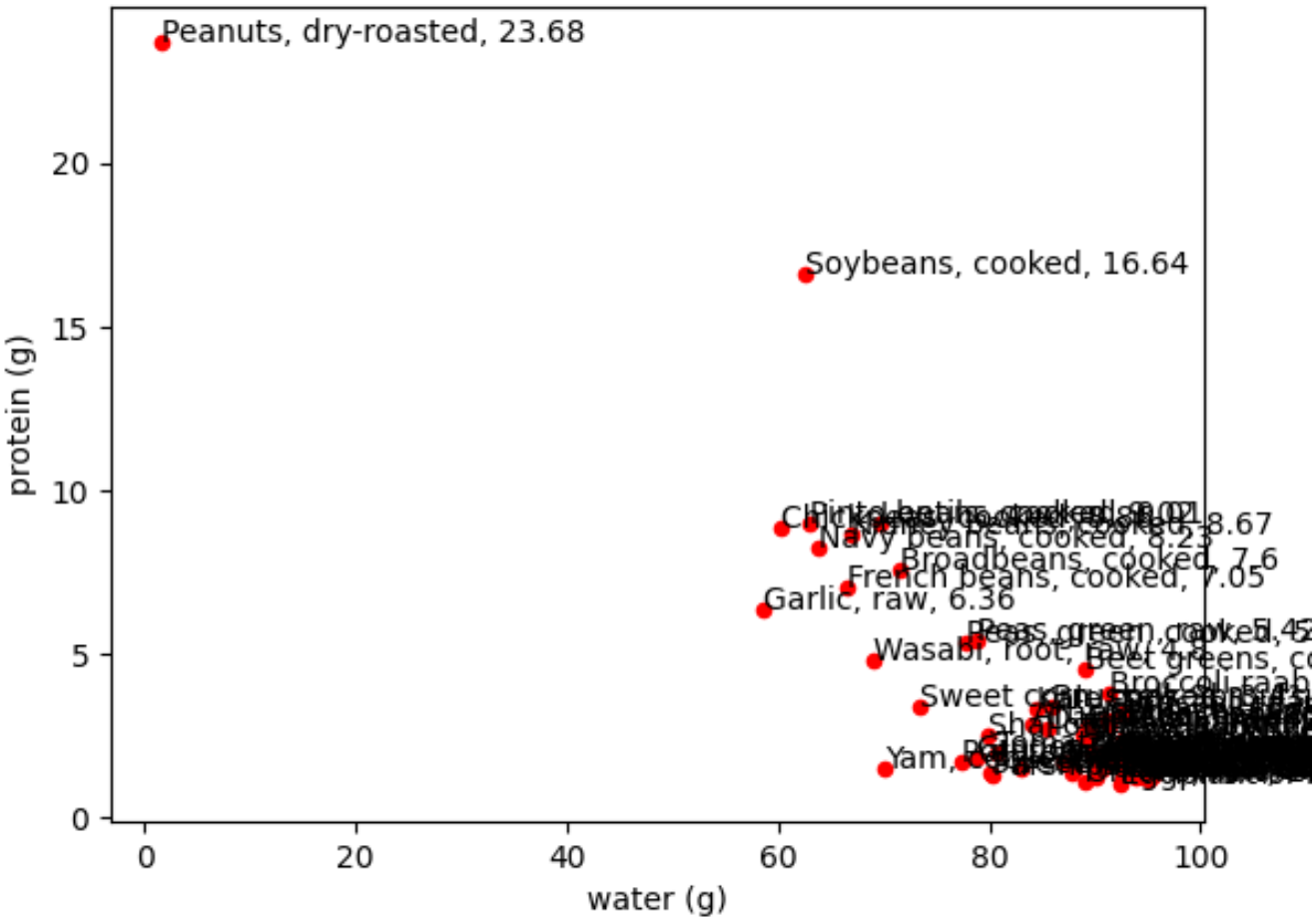


Figure: Visualisation de tous les Légumes par “scatter plot” en fonction de leur “protein (g)” et “water (g)”.

Visualisation avec Matplotlib

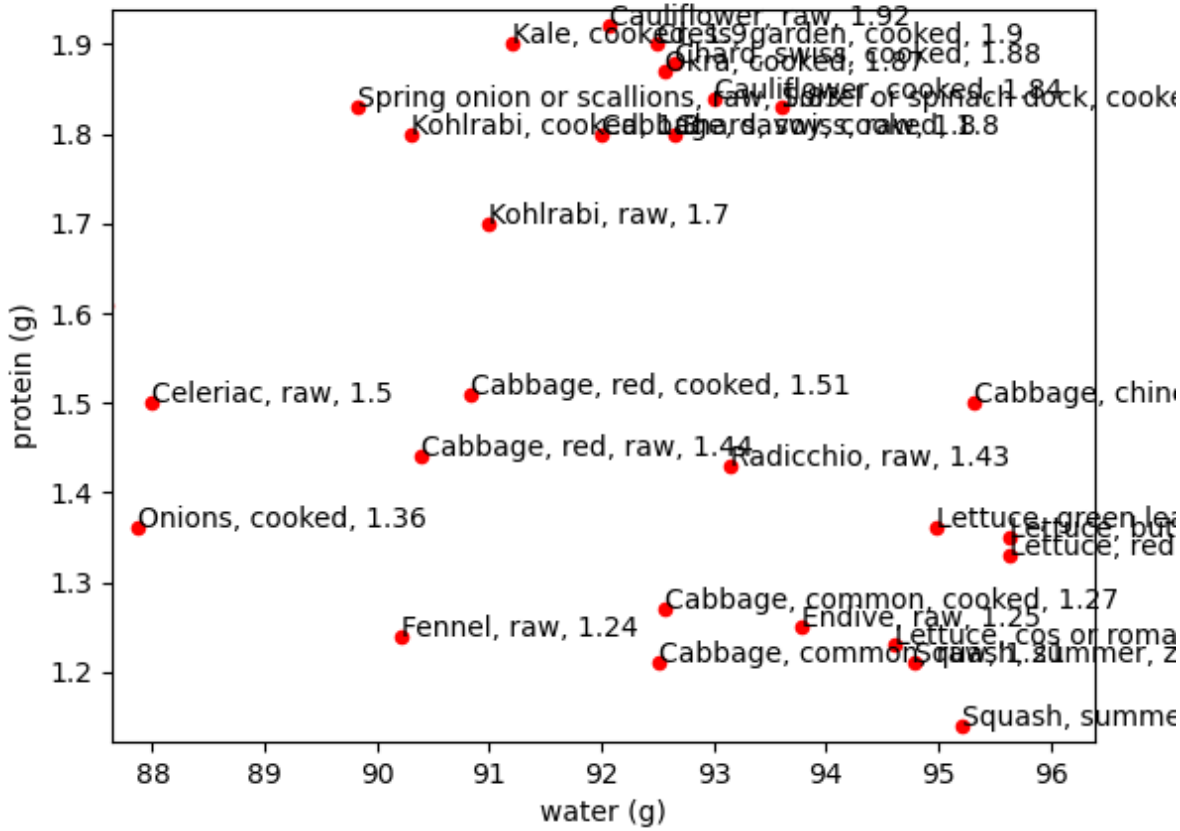


Figure: Il est possible de faire un Zoom avec l'interface interactif.

Bibliographie

- https://medium.com/@stevencibambo/loading-data-from-mongodb-with-pymongo-for-analysis-c0
- https://stackoverflow.com/questions/15910019/annotate-data-points-while-plotting-from-pandas-datafr