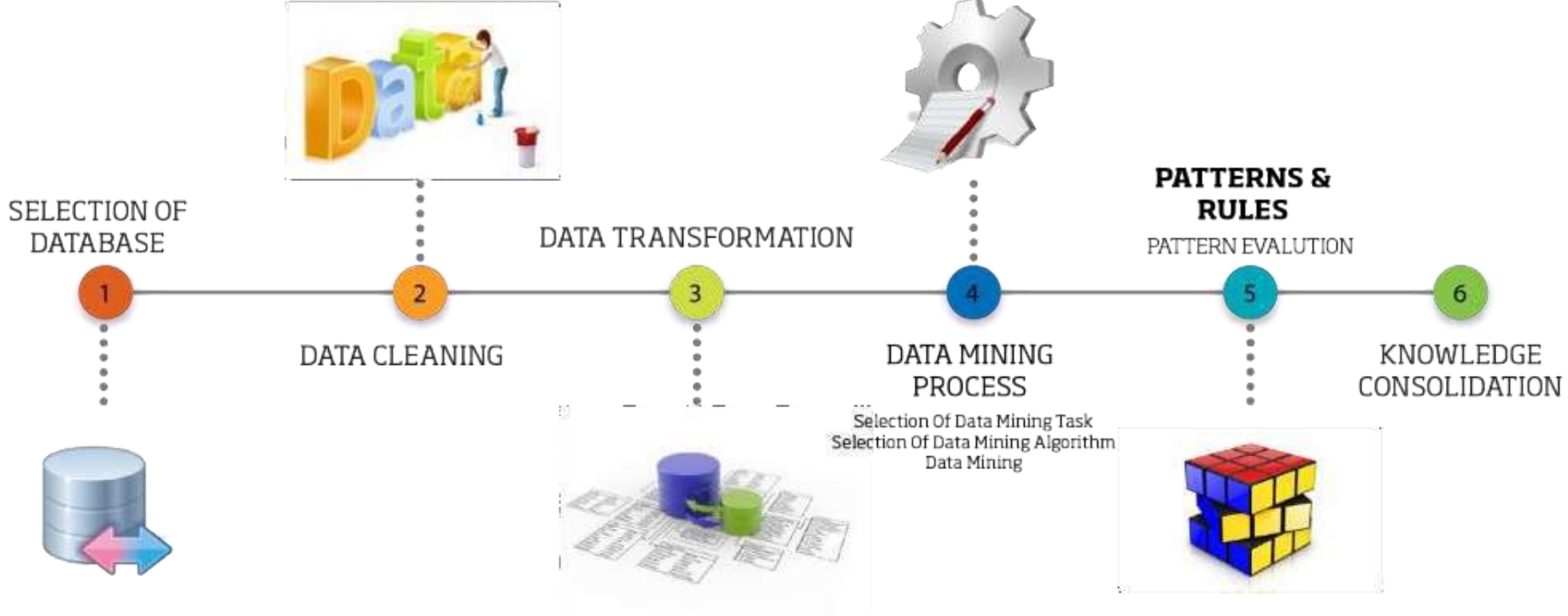


Exploration et nettoyage des données en python

Data cleaning / Data Transformation

La préparation des données

- Nous pouvons définir la préparation des données comme la transformation de données brutes en une forme plus adaptée à la modélisation.

La préparation des données

- La préparation des données permet d'obtenir des données de meilleure qualité pour l'analyse et d'autres tâches liées à la gestion des données. Elle consiste à éliminer les erreurs et à normaliser les données brutes avant leur traitement.

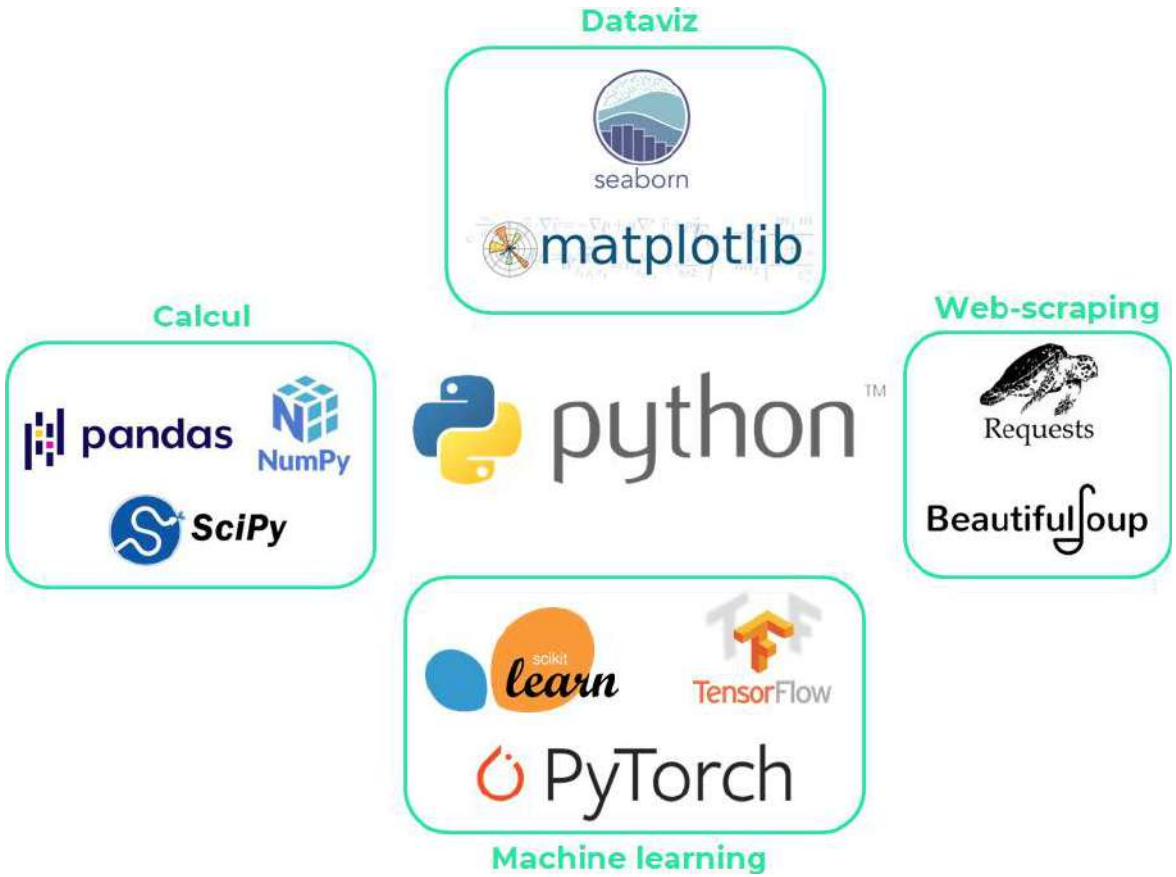
- Les types de préparation des données effectués dépendent du type des données

Les tâches de préparation de données

- **Nettoyage des données**: Identifier et corriger les erreurs ou les erreurs dans les données.
- **Feature selection**: Identification des variables d'entrée les plus pertinentes pour la tâche.
- **Feature engineering**: Dérivation de nouvelles variables à partir des données disponibles.
- **Transformations de données**: Modification de l'échelle ou de la distribution des variables.
-
- **Réduction de la dimensionnalité**: Création de projections compactes des données.

Roadmap for Data Preparation

Bibliothèques Python



Nettoyage des données
Data Cleaning

Les types des variables

- Les variables quantitatives ou numériques
  - Continue : le nombre de valeurs possibles est très grand
  - Discrète : le nombre de valeur possible est très petit

- Les variables Qualitatives
  - Ordinal : si les valeurs peuvent être ordonnées comme les mentions attribuées à un examen (moyen, bien, très bien)
  - Nominal : qualitatives sans ordre comme le code alphanumérique.
  - Binaire : Ce sont des variables qualitatives qui ne prennent que 2 modalités (0/1, oui/non, true/false).

Le nettoyage du jeu de données Data Cleaning

- Pourquoi?
  - Les données réelles sont des fois incomplètes dû à un dysfonctionnement lors de la récupération de la valeur à travers un dispositif.
- Impact?
  - Le résultat de la fouille va être erroné
- Comment nettoyer les données
  - Traiter les données manquantes,
  - Traiter les doublons
  - Traiter les données erronées ou les outliers
  - Traiter les dates

Nettoyage des données manquantes

- Variable non importante pour l'analyse et le nombre de valeurs manquantes est important
  - Éliminer cette variable
- La fonction dropna permet de supprimer des axes (colonnes ou lignes) d'un DataFrame

```
import numpy as np
import pandas as pd

df=pd.DataFrame([[np.nan, 2, 0, np.nan, 9],[np.nan, 0,np.nan, 1,5],[np.nan, 4, 4, np.nan, 3],
                 [np.nan, np.nan, np.nan, 5,8]], columns=list('ABCDE'))

#supprimer les colonnes avec toutes les valeurs NaN
df.dropna(axis='columns', how='all', inplace=True)

#supprimer les colonnes avec au moins une valeur NaN
df.dropna(axis='columns', how='any', inplace=True)

#supprimer les lignes avec toutes les valeurs NaN
df.dropna(axis='index', how='all', inplace=True)

#supprimer les lignes avec au moins une valeur NaN
df.dropna(axis='index', how='any', inplace=True)

print(df)
```

Nettoyage des données manquantes

- **Supprimer des axes**

- La fonction drop permet de supprimer des axes (colonnes oulignes) d'un DataFrame

```
import pandas

df = pandas.DataFrame({'A': [1.1, 2.7, 5.3], 'B': [2, 10, 9], 'C': [3.3, 5.4, 1.5], 'D': [4, 7, 15]}, index = ['a1', 'a2', 'a3']

print(df.shape)

df.drop(columns=["A"], inplace=True)
```

```
(3, 4)
(3, 3)
```

Nettoyage des données manquantes

- Variable importante pour l'analyse et le nombre de valeurs manquantes est important

  - Créer un sous échantillon en enlevant les individus ayant cette variable manquante. Le problème c'est que nous pouvons perdre beaucoup de données ou de point d'entrée dans le dataset.

  - L'imputation ou le remplacement des valeurs manquantes par la moyenne, le mode ou la médiane si la variable est numérique ou le mode si elle est catégorique

Nettoyage des données manquantes

- **Remplacer les valeurs NaN : fonction fillna**
- Remplacer les NaN par d'autres valeurs

```
import numpy as np
import pandas as pd

df=pd.DataFrame([[np.nan, 2, 0, np.nan, 9],[np.nan, 0,np.nan, 1,5],[np.nan, 4, 4, np.nan, 3],
                 [np.nan, np.nan, np.nan, 5,8]], columns=list('ABCDE'))

#Remplacer toutes les valeurs NaN par 4
df.fillna(4, inplace=True)

df.fillna(50, axis='columns', limit=2)


print(df)
```

```
    A    B    C    D  E
0 NaN  2.0  0.0  NaN  9
1 NaN  0.0  NaN  1.0  5
2 NaN  4.0  4.0  NaN  3
3 NaN  NaN  NaN  5.0  8
```

Les valeurs Outliers

- Ce sont les valeurs extrêmes ou erronées
- Comment détecter ces valeurs
  - Utiliser un **box plot** : un graphique sous forme de rectangle où sont décrites les statistiques de la variable (les quartiles (Q1, médiane, Q3).
  - Les bornes du graphique délimitent les valeurs selon la distribution de la variable. Au-delà de ces extrémités, ces valeurs sont considérées comme des valeurs aberrantes.

Les données catégoriques

- Les données Ordinales
- Les données nominales
- Le type de données en python c'est « object »

```
object_columns_df = df.select_dtypes(include=['object'])
numerical_columns_df =df.select_dtypes(exclude=['object'])
```

Les données catégoriques

One-hot encoding pour les variables nominales

- C’est le codage des variables catégoriques en bits
- Une variable qui peut avoir n valeurs peut être codée enn bits avec un seul bit qui prend la valeur 1
- Pourquoi parce que certains algorithmes ne prennent en considération que des valeurs numériques

Couleur	code
Noir	100
Blanc	010
rouge	001

Exemple

- Codage one hot de la colonne color

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.4
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.3
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.3
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.6
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.7

Exemple codage one hot d'une couleur

- Appeler la méthode dummies de pandas

	D	E	F	G	H	I	J
0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0
2	0	1	0	0	0	0	0
3	0	0	0	0	0	1	0
4	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...
53935	1	0	0	0	0	0	0
53936	1	0	0	0	0	0	0
53937	1	0	0	0	0	0	0
53938	0	0	0	0	1	0	0
53939	1	0	0	0	0	0	0

```
dummies = pd.get_dummies(df['color'])
```

Exemple codage one hot d'une couleur

- Concaténer le résultat dummies avec le dataset de base

cut	color	clarity	depth	table	price	x	y	z	D	E	F	G	H	I	J
Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43	0	1	0	0	0	0	0
Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31	0	1	0	0	0	0	0
Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31	0	1	0	0	0	0	0
Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63	0	0	0	0	0	1	0
Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

```
df = pd.concat([df, dummies], axis = 1)
```

Exemple codage one hot d'une couleur

- Supprimer la colonne color qui est devenue inutile

cut	color	clarity	depth	table	price	x	y	z	D	E	F	G	H	I	J
Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43	0	1	0	0	0	0	0
Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31	0	1	0	0	0	0	0
Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31	0	1	0	0	0	0	0
Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63	0	0	0	0	0	1	0
Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

```
df.drop(['color'], axis = 1, inplace = True)
```

Label encoding Variables Ordinales

- Donnez un entier représentant l'ordre de la variable
- Nous utilisons dans ce cas « Label Encoding » de sklearn

Original Encoding	Ordinal Encoding
Poor	1
Good	2
Very Good	3
Excellent	4

Exemple

- Variable non importante pour l'analyse et le nombre de valeurs manquantes est important

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	1	3	61.5	55.0	326	3.95	3.98	2
1	2	0.21	Premium	1	2	59.8	61.0	326	3.89	3.84	2
2	3	0.23	Good	1	4	56.9	65.0	327	4.05	4.07	2
3	4	0.29	Premium	5	5	62.4	58.0	334	4.20	4.23	2
4	5	0.31	Good	6	3	63.3	58.0	335	4.34	4.35	2
...	...	...	...	...	...	...	...	...	...	...	...
53935	53936	0.72	Ideal	0	2	60.8	57.0	2757	5.75	5.76	3
53936	53937	0.72	Good	0	2	63.1	55.0	2757	5.69	5.75	3
53937	53938	0.70	Very Good	0	2	62.8	60.0	2757	5.66	5.68	3

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['color'] = le.fit_transform(df['color'])
df['clarity'] = le.fit_transform(df['clarity'])
```

Installer Scikit learn

- Installing scikit-learn — scikit-learn 0.17.1 documentation


Great
Job!