


Module : Analyse et fouille de données	
Responsable du Cours: Bouaziz Souhir, Abbès Amal Enseignants TP: Barhoumi Chawki, Rekik Amal, Njeh Maissa	Auditoire: D-LSI-ADBD A-U: 2023-2024
TP0 : Introduction au processus ECD <i>Manipulation des matrices des données avec Python</i>	

Tutoriel sur la manipulation des matrices des données avec Python

Python exécute programmes ou scripts, programmes qui peuvent être précompilés pour plus d'efficacité. Ce langage s'exécute également à l'aide d'un interprète de commande (*IDLE* ou *IPython*) de manière interactive.

Installation Python 3 : Conda / Anaconda (tous OS)

Outils :

- Jupyter / IPython Notebook : Reproducible Data Analysis in Jupyter (Tutos de Jake Vanderplas : <http://jakevdp.github.io/blog/2017/03/03/reproducible-data-analysis-in-jupyter/>)
- Numpy : pour utiliser vecteurs et tableaux (<https://numpy.org/>)
- Scipy : intègre les principaux algorithmes numériques (<https://www.scipy.org/>)
- Pandas : structure de données et feuilles de calcul (<https://github.com/jorisvandenbossche/pandas-tutorial>)
- scikit-learn : algorithmes d'analyse des données et d'apprentissage statistique (<http://scikit-learn.org/stable/tutorial/index.html>)

Pandas est une librairie Python spécialisée dans la structuration de données et feuilles de calcul et aussi l'analyse des données.

L'installation de la librairie soit:

- Avec *pip*: `pip install pandas`
- Avec *Anaconda*: `conda install pandas`

L'importation de la librairie et raccourci classiquement utilisé: `import pandas as pd`

Vérifier la version installée: `print(pandas.__version__)`

La richesse des fonctionnalités de la librairie pandas est une des raisons, si ce n'est la principale, d'utiliser Python pour extraire, préparer, éventuellement analyser, des données.

- **Objets** : les classes *Series* et *DataFrame* ou table de données.
- Lire, écrire création et exportation de tables de données à partir de fichiers textes (séparateurs, .csv, format fixe, compressés), binaires (HDF5 avec Pytable), HTML, XML, JSON, MongoDB, SQL...
- **Gestion d'une table** : sélection des lignes, colonnes, transformations, réorganisation par niveau d'un facteur, discrétisation de variables quantitatives, exclusion ou imputation élémentaire de données manquantes, permutation et échantillonnage aléatoire, variables indicatrices, chaînes de caractères...
- **Statistiques** élémentaires uni, bi et multivariées, tri à plat (nombre de modalités, de valeurs nulles, de valeurs manquantes...), graphiques associés, statistiques par groupe, détection élémentaire de valeurs atypiques...
- **Manipulation de tables** : concaténations, fusions, jointures, tri, gestion des types et formats...

Les types *Series* et *DataFrame* :

De même que la librairie *Numpy* introduit le type *array* indispensable à la manipulation de matrices en calcul scientifique, celle *pandas* introduit les classes *Series* (séries chronologiques) et *DataFrame* ou table de données indispensables en statistique.

- **Series** : La classe *Series* est l'association de deux *arrays* unidimensionnels. Le premier est un ensemble de valeurs indexées par le 2ème qui est souvent une série temporelle. Ce type est introduit principalement pour des applications en Économétrie et Finance où Python est largement utilisé.
- **DataFrame** : Cette classe est proche de celle du même nom dans le langage R, il s'agit d'associer avec le même index de lignes des colonnes ou variables de types différents (entier, réel, booléen, caractère). Elle correspond à une matrice individus-variables où les lignes correspondent à des observations, les colonnes à des attributs décrivant les individus. Elle peut également être vue comme une liste de *Series* partageant le même index. L'index de colonne (noms des variables) est un objet de type *dict* (dictionnaire). C'est la classe qui sera principalement utilisée dans ce TP.

Exercice :

Considérons la matrice de données suivante :

	Name	score	attempts	qualify
A	Anastasia	12.5	1	yes
b	Dima	9.0	3	no
c	Katherine	16.5	2	yes
d	James	NaN	3	no
e	Emily	9.0	2	no
f	Michael	20.0	3	yes
g	Matthew	14.5	1	yes
h	Laura	NaN	1	no
i	Kevin	8.0	2	no
j	Jonas	19.0	1	yes

- 1) Créer et afficher un *DataFrame* avec les étiquettes de l'index.
- 2) Afficher les informations de base du *DataFrame* ainsi que ses données
- 3) Obtenir les 3 premières lignes du *DataFrame* donné. Ajouter une autre variable à *stat2*, nommée *Nord*, remplie par *True* pour les gouvernorats du nord et *False* sinon. Puis supprimer cette variable.
- 4) Sélectionner les colonnes «name» et «score»
- 5) Sélectionner les colonnes «name» et «score» des lignes 1, 3, 5 et 6
- 6) Sélectionner les lignes où le nombre de tentatives « attempts » d'examen est supérieur à 2

- 7) Compter le nombre de lignes et de colonnes d'un DataFrame
- 8) Sélectionner les lignes où le score est manquant, c'est-à-dire NaN
- 9) Sélectionner les lignes dont le score « score » est compris entre 15 et 20 (inclus)
- 10) Sélectionner les lignes où le nombre de tentatives à l'examen « attempts » est inférieur à 2 et le score « score » supérieur à 15
- 11) Changer le score de la ligne «d» en 11,5
- 12) Calculer la somme des tentatives d'examen « attempts » des élèves
- 13) Calculer le score moyen des élèves
- 14) Ajouter une nouvelle ligne «k» au DataFrame avec des valeurs données pour chaque colonne
- 15) Supprimez la nouvelle ligne et renvoyez le bloc de données d'origine
- 16) Trier le DataFrame d'abord par «name» dans l'ordre croissant
- 17) Trier le DataFrame par «score» dans l'ordre décroissant
- 18) Remplacer la colonne «qualify» contenant les valeurs «yes» et «no» par True et False
- 19) Changer le nom «James» en «Suresh» dans la colonne de nom du DataFrame
- 20) Supprimer la colonne «attempts» du DataFrame
- 21) Insérer une nouvelle colonne dans le DataFrame existant
- 22) Parcourir les lignes d'un DataFrame
- 23) Remplacer toutes les valeurs NaN par des Zéro
- 24) Définir une valeur donnée pour une cellule particulière dans le DataFrame à l'aide de la valeur d'index
- 25) Compter les valeurs NaN dans une ou plusieurs colonnes dans le DataFrame
- 26) Obtenir la liste des en-têtes des colonnes du DataFrame
- 27) Renommer les colonnes d'un DataFrame
- 28) Sélectionner des lignes à partir du DataFrame en fonction des valeurs de certaines colonnes
- 29) Changer l'ordre des colonnes du DataFrame.