TP 1 : La manipulation des données avec les bibliothèques Pandas, Numpy, et Matplotlib

I. Introduction à Pandas

Pandas est une bibliothèque Python open-source offrant des outils de manipulation et d'analyse de données à haute performance. Le nom Pandas est dérivé de l'expression Panel Data – un terme en économétrie qui désigne des données multidimensionnelles. Grâce à Pandas, nous pouvons accomplir les cinq étapes typiques du traitement et de l'analyse des données, quelle que soit leur origine : charger, préparer, manipuler, modéliser et analyser.

II. Fonctionnalités clés de Pandas

- Objet DataFrame rapide et efficace avec un indexage par défaut et personnalisé.

- Outils pour charger des données dans des objets en mémoire à partir de différents formats de fichiers.

- Alignement des données et gestion intégrée des données manquantes.

- Remodelage et pivotement des ensembles de données.

- Découpage, indexation et sous-ensemble de grands ensembles de données basés sur des étiquettes.

- Suppression ou insertion de colonnes dans une structure de données.

- Groupement de données pour l'agrégation et les transformations.

- Fusion et jointure de données à haute performance.

- Fonctionnalité pour les séries temporelles.

III. Structures de données dans Pandas

- **Series** : une dimension

- **DataFrame** : deux dimensions

- **Panel** : trois dimensions et plus

Exercice 1

1. Créer un DataFrame, où la clé est le **nom de l'entreprise** et les valeurs sont **les séries**. Associer les valeurs à la liste d'index lors de la création des séries.

```
index_list=['Company A','Company B','Company C','Company D','Company
E','Company F']
company_dir = {'Closing price': pd.Series([346.15,0.59,459,0.52,589.8,158.88],
                                index=index_list),

            'EPS': pd.Series([1133.43,36.05,145.02, 4.5, 31.44,380.64],
                        index=index_list),

            'Beta': pd.Series([1,2,3,4,5,6],index=index_list),
            'P/E': pd.Series([10,20,30,40,60,50], index=index_list),
            'Market Cap(B)': pd.Series([1254.05, 43.2, 2300, 5.6, 773.8, 521.56],
index=index_list)

           }
```

2. Convertir la structure de données « company_dir » en dataframe et nommer le « companydf » puis l' afficher.
3. **Slicing** est une façon d'extraire un sous-ensemble d'un DataFrame qui permet d'extraire les lignes et colonnes souhaitées en utilisant les indices appropriés pour le slicing. Il existe deux méthodes, loc()et iloc(), qui peuvent être utilisées pour créer des sous-ensembles de données.
    a) Extraire les lignes de 1 à 4 de début et de fin . Le premier indice dans le slicing est inclus, et le dernier indice ne l'est pas.
    b) Quelle est le résultat de la commande suivante companydf[-3:] ?
    c) Extraire seulement la colonne 'EPS' .
    d) Quelle est le rôle de la fonction **loc** dans ce cas companydf.loc['Company B': 'Company D'][['EPS', 'Beta']]
    e) Extraire les lignes d'index 1 et 2 et les colonnes d'index 0 et 1.
    f) Extraire toutes les lignes des colonnes aux indices 1, 4 et 3.
4. Exporter le DataFrame (companydf) vers un fichier CSV.

Exercice 2

Dans cet exercice , vous allez travailler avec le dataset "POS_Data.csv".

1. Importer les packages nécessaires
2. Charger le data set (le fichier de données)
3. Afficher les premières lignes
4. Afficher les 8 premières lignes
5. Afficher les 5 dernières lignes
6. Afficher le nombres de lignes et de colonnes dans dataset
7. Afficher les attributs (features) du dataset
8. Afficher le type de chaque caractéristique
9. Convertir le type de la colonne « Page_traffic » en type int64

10. Trouver les valeurs manquantes dans chaque colonne
11. Supprimer la colonne « Unit_price » et afficher le dataset après la suppression
12. Utiliser dropna() pour supprimer les lignes qui contiennent des données manquantes.

Exercice 3

Dans cet exercice , vous allez travailler avec le dataset " POS_CleanData.csv".

1. Générer les statistiques descriptives de toutes les colonnes du dataset
2. Filtrer les lignes du DataFrame où la colonne Brand est égale à 'Close-up'.
3. Retourner le nombre de lignes du DataFrame
4. Créer un nouveau DataFrame df contenant uniquement les lignes de pos_data où la colonne Revenue($) est comprise entre 10 000 et 15 000
5. Retourner les produits ayant Revenue($) supérieur à 25 000 USD et Page_traffic inférieur à 1000.
6. Convertir la colonne Date du DataFrame pos_data en un format de date-heure

Exercice 4

1. Charger le dataset FuelConsumption CO2
2. Tracer les valeurs CO2EMISSIONS par rapport à ENGINESIZE
3. Définir la variable indépendantes (Engine Size) et dépendantes (CO2EMISSIONS)
4. Diviser les données en données entrainement 80% et test 20%
5. Créer le modèle de régression linéaire et entraîner le modèle
6. Afficher les coefficients (pente et ordonnée à l'origine(intercept))
7. Faites des prédictions sur l'ensemble de test
8. Tracer les valeurs prédites et réelles sur un graphique de dispersion
9. Evaluer le modèle en utilisant des métriques comme la moyenne des erreurs quadratiques (MSE) et le coefficient de détermination (R^2)
10. Tracer les résidus pour voir la répartition des erreurs de prédiction