

École Nationale des Sciences Appliquées
UMP Oujda

TP : Machine Learning

Partie 2: Pandas, Matplotlib

A. La bibliothèque Matplotlib :


Matplotlib est une excellente bibliothèque graphique 2D et 3D pour générer des figures scientifiques. Voici quelques-uns des nombreux avantages de cette bibliothèque :

- Facile à utiliser
- Excellent contrôle de chaque élément d'une figure, y compris la taille de la figure et le DPI.
- Sortie de haute qualité dans de nombreux formats, notamment PNG, PDF, SVG,...
- Interface graphique pour explorer de manière interactive les figures et prise en charge de la génération des entêtes de fichiers de figures.

L'une des principales caractéristiques de matplotlib à souligner et qui rend matplotlib parfaitement adapté à la génération de graphes pour des publications scientifiques est que tous les aspects de la figure peuvent être contrôlés par programmation. Ceci est important pour la reproductibilité et pratique lorsque l'on a besoin de régénérer la figure avec des données mises à jour ou de changer son apparence.

Plus d'informations sur la page Web Matplotlib : <http://matplotlib.org/>

Pour commencer à utiliser Matplotlib dans un programme Python, importez le module matplotlib.pyplot :

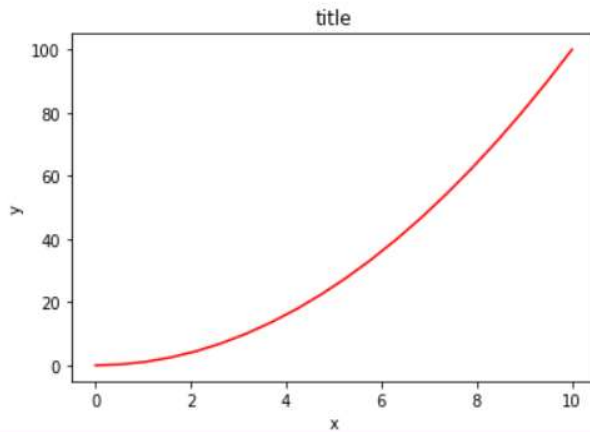


```
import matplotlib.pyplot as plt
```

Voici une simple figure avec matplotlib :

```
x = np.linspace(0, 10, 20) # x = [0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0
#                               , 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 10.0]
y = x ** 2

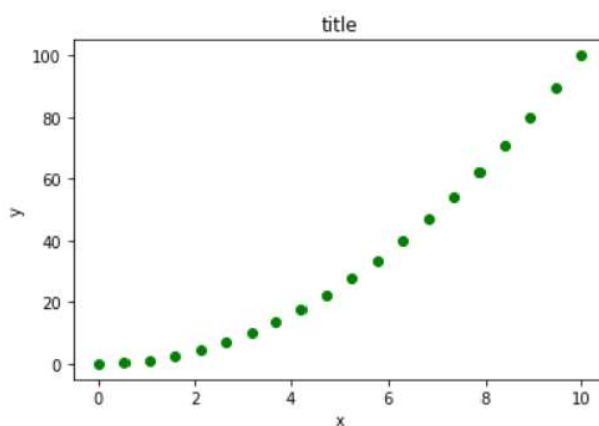
plt.figure()
plt.plot(x, y, 'r') # 'r' pour la couleur du graphe 'red'
plt.xlabel('x')
plt.ylabel('y')
plt.title('title') # le titre du graphe
plt.show()
```



Voici le même exemple, mais dans ce cas av nuage de point :

```
x = np.linspace(0, 10, 20) # x = [0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0
#                               , 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 10.0]
y = x ** 2

plt.figure()
plt.scatter(x, y, c='g') # 'g' pour la couleur du graphe 'green'
plt.xlabel('x')
plt.ylabel('y')
plt.title('title') # le titre du graphe
plt.show()
```



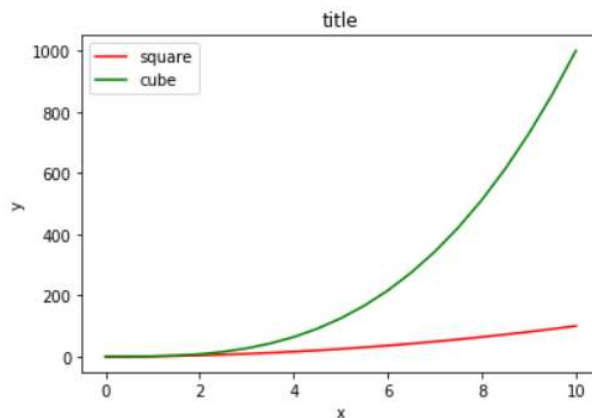
Et voici comment afficher plusieurs courbes dans une seule figure :

```

x = np.linspace(0, 10, 21) # x = [0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0,
#                               4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 10.0]
y1 = x ** 2
y2 = x ** 3

plt.figure()
plt.plot(x, y1, c='r', label = 'square')
plt.plot(x, y2, c='g', label = 'cube')
plt.xlabel('x')
plt.ylabel('y')
plt.title('title') # le titre du graphe
plt.legend()
plt.show()

```

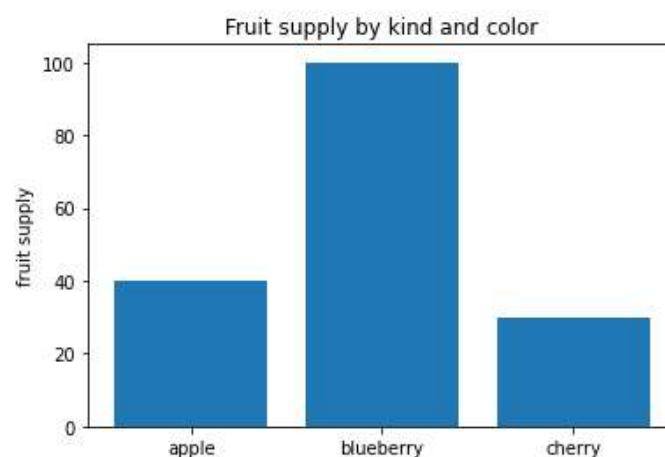


Graphe à bars :

```

1 import matplotlib.pyplot as plt
2
3 fig, ax = plt.subplots()
4
5 fruits = ['apple', 'blueberry', 'cherry']
6 counts = [40, 100, 30]
7
8 ax.bar(fruits, counts)
9
10 ax.set_ylabel('fruit supply')
11 ax.set_title('Fruit supply by kind and color')
12
13 plt.show()

```

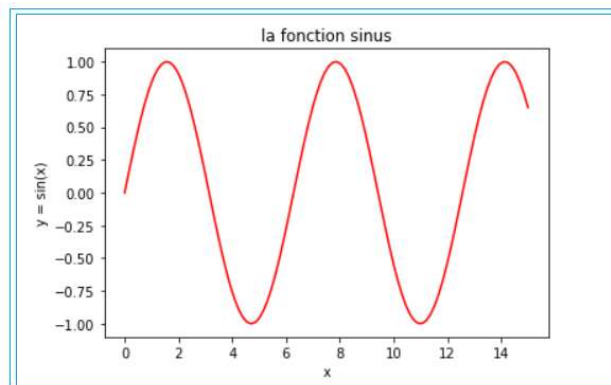


Afin de sauvegarder une figure sous forme d'une image on utilise la fonction :

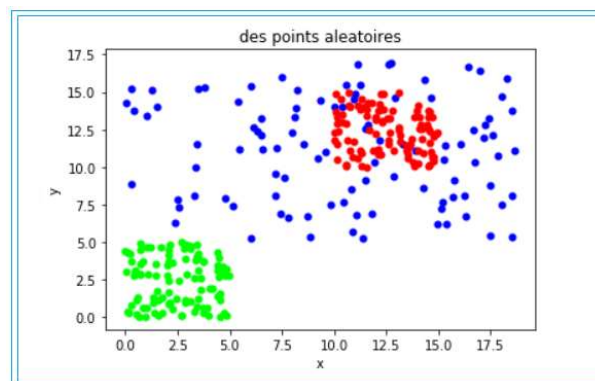
```
plt.savefig('plt_test.png')
```

TP 1 :

1. Écrivez un script matplotlib qui affiche la courbe de la fonction sinus :



2. Écrivez un script matplotlib qui affiche la figure suivante :



B. La bibliothèque Pandas :

Lorsque vous travaillez avec des données tabulaires, telles que des données stockées dans des feuilles de calcul ou des bases de données, pandas est l'outil qu'il vous faut.

Pandas est une librairie Python qui vous aidera à explorer, nettoyer et traiter vos données. Dans pandas, une table de données s'appelle un DataFrame.

Pour utiliser Pandas, vous devez au préalable importer le package *pandas* avec l'instruction suivante :

```
import pandas
```

ou

```
import pandas as pd
```

Voici quelques fonctions et méthodes fournies par pandas pour la manipulation des DataFrames :

❖ **Pour créer un dataframe à partir des fichiers de données :**

- ✓ **pandas.read_excel()** ou **pd.read_excel()** : pour les fichiers (.xls).
- ✓ **pd.read_csv()** : pour les fichiers (.csv).
- ✓ **pd.read_json()** : pour les fichiers (.json).
- ✓ **pd.read_html(), pd.read_sql(), pd.read_....()**
- ✓ C'est possible aussi de créer un DataFrame à partir d'un tableau : **df = pd.Series([1, 2, 3, 4, ...])**

❖ **La sélection :**

- ✓ **df['column_name']**
- ✓ **df[num_row_start : num_row_end]**
- ✓ **df.loc[num_row, ['column_name', 'column_name'], ...]**

❖ **La modification :**

- ✓ **df.at[num_row, "column_name"] = value**

❖ **Filtrer les données :**

- ✓ **df[df['column_name'] == 'value']**, vous pouvez utiliser **>**, **>=**, **<**, **<=**, et **!=**, utiliser le caractère **&** avec les parenthèses pour plusieurs conditions, exemple : **df[(df['column_name'] == 'value') & (df['column_name'] == 'value')]**

❖ **La création de nouvelles colonnes à partir de colonnes existantes :**

- ✓ **df['column_name'] = df['column_name'] opération df['column_name']**, l'opération peut être n'importe quelle opération arithmétique

❖ **Manipulation des données :**

- ✓ **df.head()** : Afficher le début du DataFrame.
- ✓ **df.tail()** : Afficher la fin du DataFrame
- ✓ **df.columns** : Afficher les noms des colonnes
- ✓ **df.describe()** : Statistiques rapides
- ✓ **df.drop(['column', 'column', ...])** : Eliminer certaines colonnes
- ✓ **df['column'].value_counts()** : Compter les répétitions
- ✓ **df.groupby(['column'])** : Analyse par groupe
- ✓ **data.fillna(valeur_par_defaut)** : compléter les données manquantes par une valeur par défaut

- ✓ **data.dropna()** : Eliminer les lignes dans lesquelles il manque des données.
- ✓ **Le trie** : `df.sort_values(by="column_name")`
- ✓ **Applique une fonction sur une colonne** : `df['column_name'].apply(lambda x : opération)`, exemple : `df['column_name'].apply(lambda x : x * 2)`

TP 2 :

1. Écrivez un script qui permet de charger les données à partir du fichier 'titanic.xls'
2. Afficher les cinq premières lignes
3. Afficher les 5 dernières lignes
4. Afficher les lignes de 6 à 10
5. Afficher la valeur de la colonne 'name' de la ligne 5
6. Afficher les passagers ayants plus de 30 ans
7. Afficher les passagers de sexes féminins âgés de moins de 25 ans
8. Afficher le nombre de passagers de sexe masculins
9. Afficher le nombre de passagers ayants survécus
10. Écrivez un script Matplotlib qui affiche le nombre de passagers par sexe dans un graphe de type bar.
11. Écrivez un script Matplotlib qui affiche le nombre de passagers décédés et survécus dans un graphe de type bar.
12. Écrivez un script Matplotlib qui affiche le nombre de passagers par tranche d'âge (moins de 25 ans, entre 25 et 49 ans, entre 50 et 74 ans, entre 75 et plus) un graphe de type bar.
13. Écrivez un script Matplotlib qui affiche le nombre de passagers par type d'embarquement dans un graphe de type bar.

TP 3 :

1. Écrivez un script qui permet de charger les données à partir du fichier 'titanic.xls'
2. Nettoyez le jeu de données avec pandas. Soit avec la suppression des lignes avec des valeurs manquantes soit avec la correction de ces données.
3. Sauvegardez les données propres dans un nouveau fichier.