PRISE EN MAIN
JUPYTER

Jean-Christophe RANGON

2024

email: jc.rangon.formateur@gmail.com

linkedIn: www.linkedin.com/in/jean-christophe-rangon-dev-web

pour :

Digital Campus

Table des matières

I. Création d'un notebook

Dans le dossier racine de Jupyter, créer un dossier nommé « kaggle »

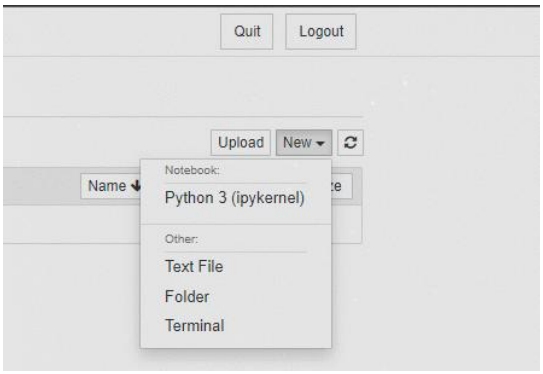
Ouvrez PowerShell, déplacez-vous dans le dossier racine jupyter et lancez le server jupyter.

$> **jupyter notebook**

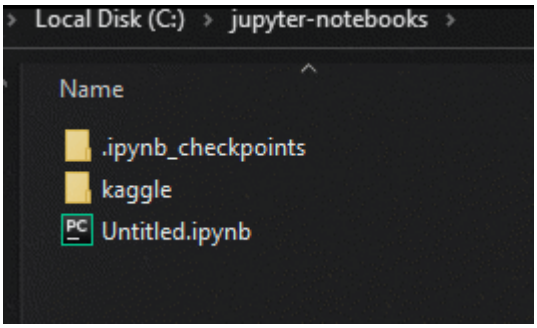
```
PS C:\> cd jupyter-notebooks
PS C:\jupyter-notebooks> jupyter notebook
[I 2023-01-15 17:59:27.675 LabApp] JupyterLab extension loaded from C:\Users\jcran\.pyenv\pyenv-win\versions\3
site-packages\jupyterlab
[I 2023-01-15 17:59:27.675 LabApp] JupyterLab application directory is C:\Users\jcran\.pyenv\pyenv-win\version
hare\jupyter\lab
[I 17:59:27.675 NotebookApp] Serving notebooks from local directory: C:\jupyter-notebooks
[I 17:59:27.675 NotebookApp] Jupyter Notebook 6.5.2 is running at:
[I 17:59:27.675 NotebookApp] http://localhost:8888/?token=8daefb827b6c5fa44c68bca3eaca161beecfd2f91637e633
[I 17:59:27.675 NotebookApp]  or http://127.0.0.1:8888/?token=8daefb827b6c5fa44c68bca3eaca161beecfd2f91637e633
[I 17:59:27.675 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confir
[C 17:59:27.744 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/jcran/AppData/Roaming/jupyter/runtime/nbserver-59496-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=8daefb827b6c5fa44c68bca3eaca161beecfd2f91637e633
     or http://127.0.0.1:8888/?token=8daefb827b6c5fa44c68bca3eaca161beecfd2f91637e633
0.00s - Debugger warning: It seems that frozen modules are being used, which may
0.00s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.00s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validation.
```

Aller sur http://localhost:8888

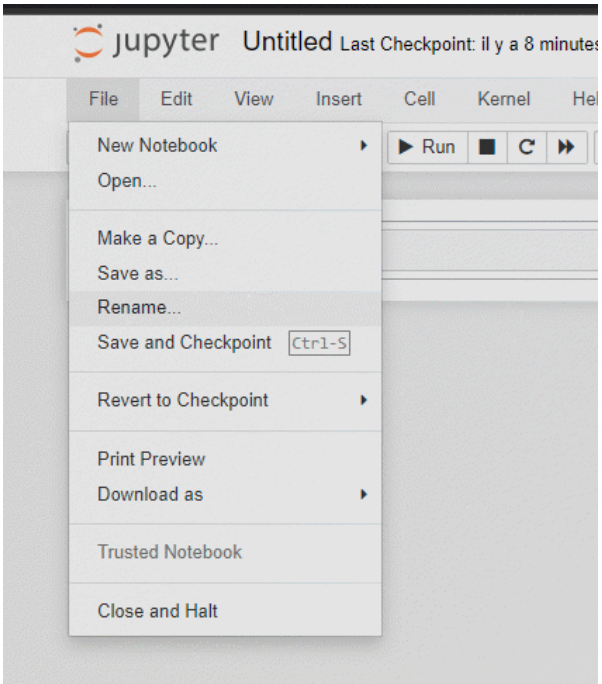
Cliquez sur le bouton ‘’new’ et choisir ‘’python 3’’

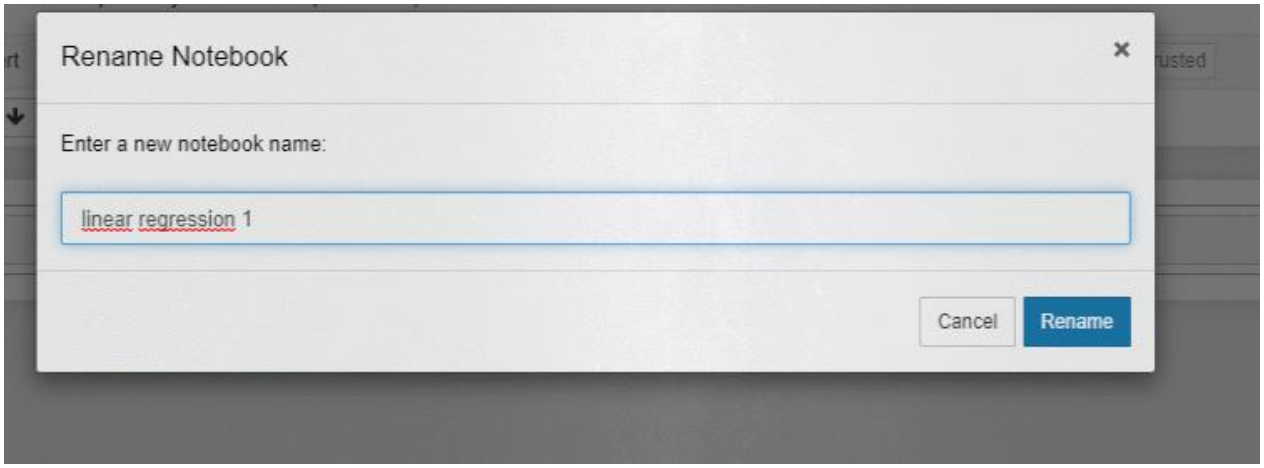


Dans votre dossier un nouveau notebook apparait avec son extension .ipynb:

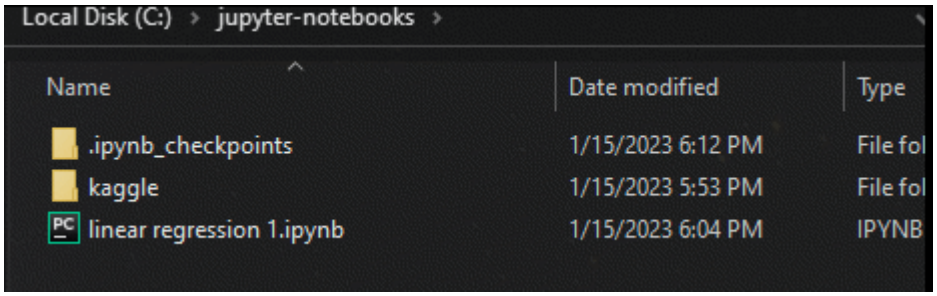


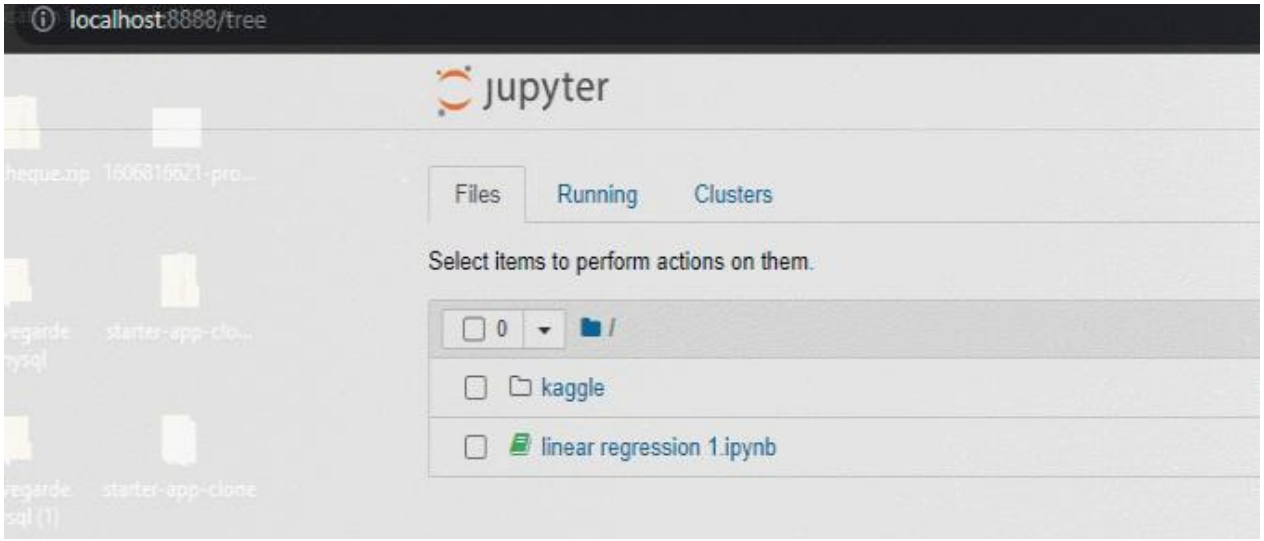
Donnons un nom à notre notebook en allant sur **File > rename**





Résultat :



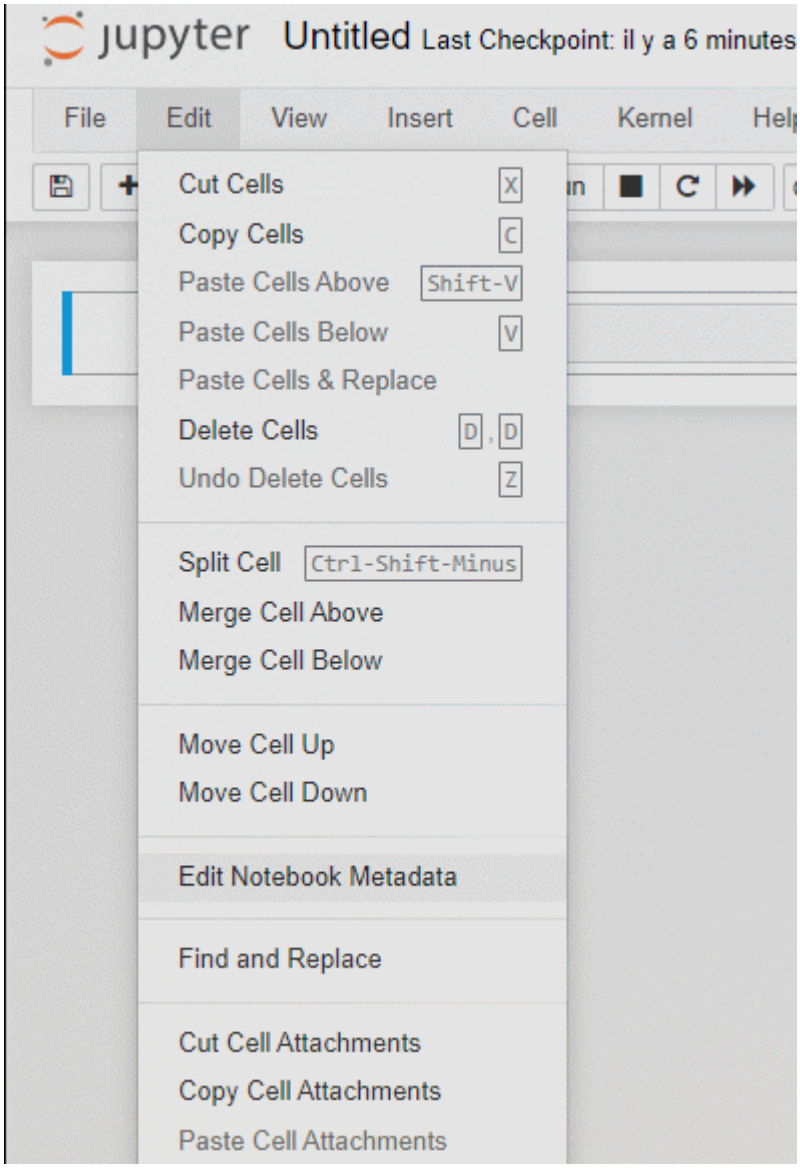


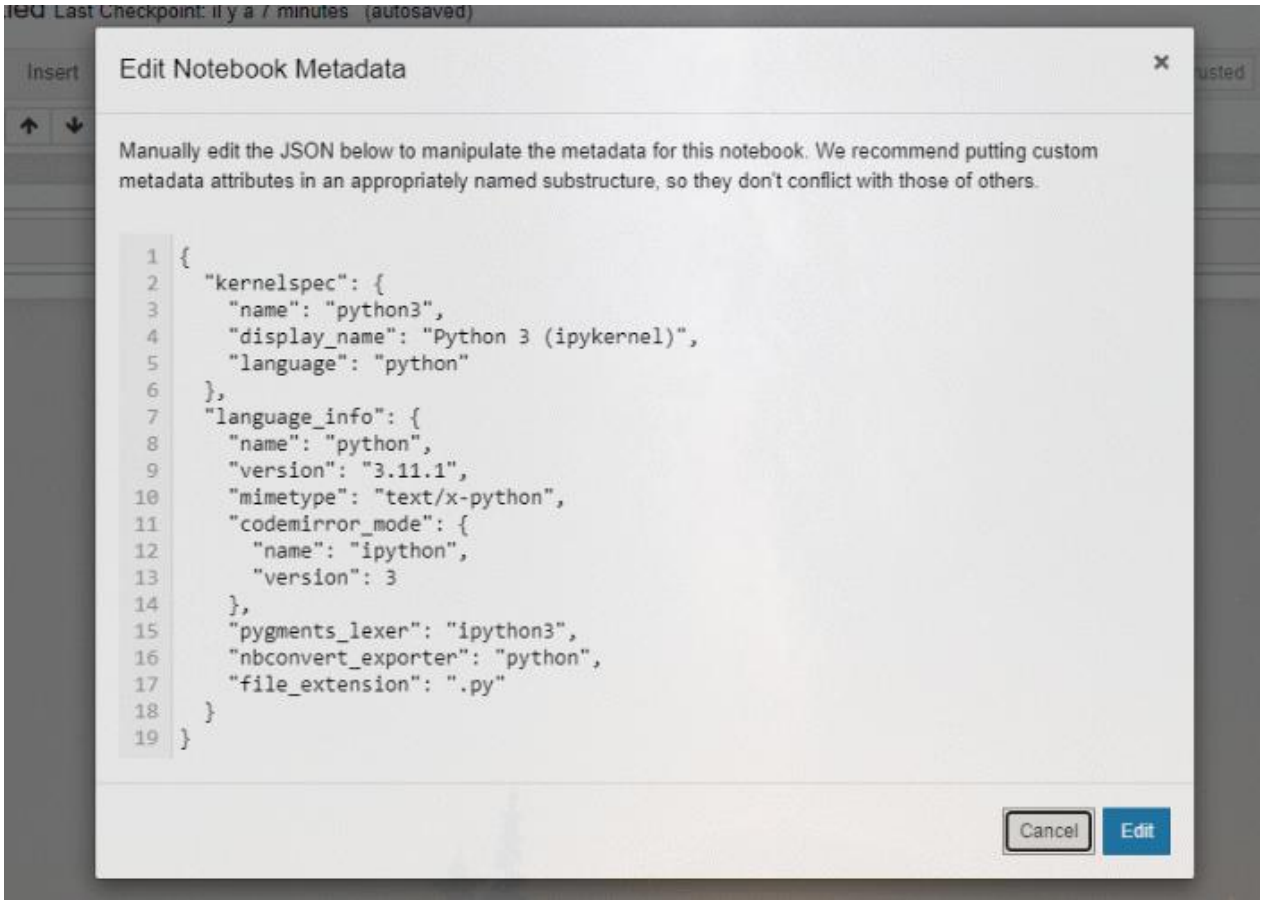
Vous pouvez aussi cliquer directement sur le nom du notebook sur la page de votre navigateur pour activer la fonctionnalité ‘’rename’’

Le fichier .ipynb est un fichier texte qui contient la description complète de votre notebook dans le format JSON avec quelques métadonnées.

Vous pouvez voir les métadonnées et même les modifier (si vous savez ce que vous faites) en allant sur

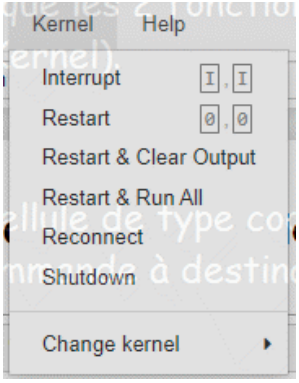
edit > edit notebook metadata





Le menu supérieur offre de nombreuses fonctionnalités ainsi que les 2 fonctionnalités principales d'un notebook. Les cellules (Cells) et les noyaux (Kernel).

Un **Kernel** est un noyau d'environnement d'exécution. Une cellule de type code contient le code qui sera exécuté par le noyaux. Plusieurs commandes à destination du noyau sont disponibles dans le menu supérieur.

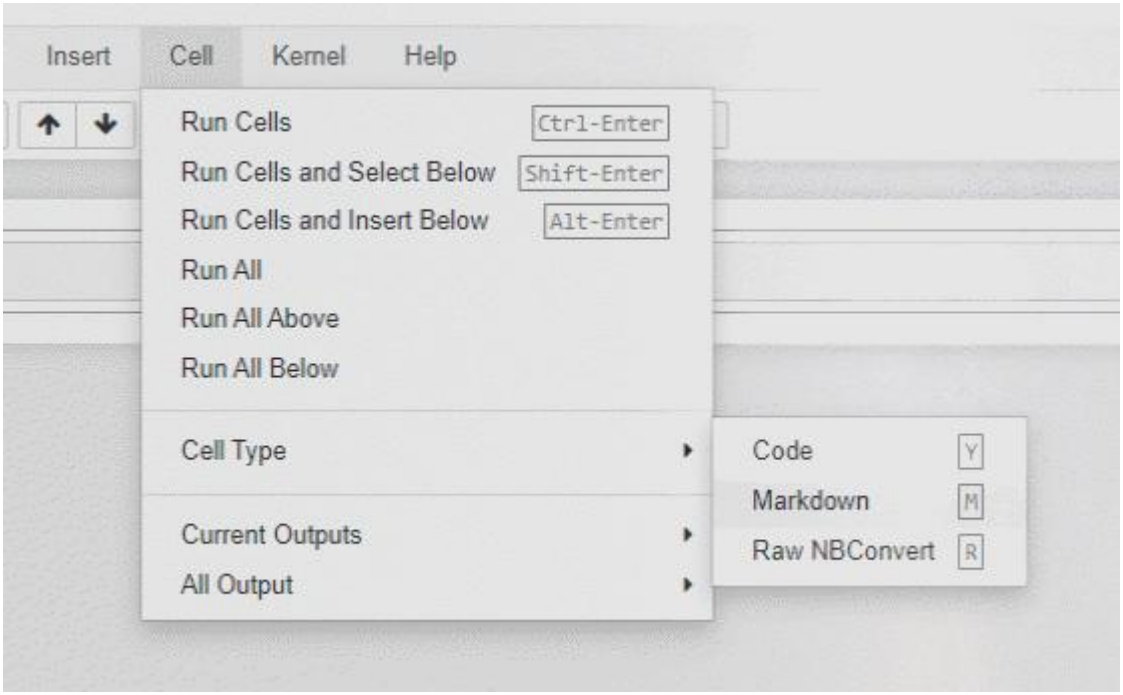


II. Ajouter du contenu au notebook

Une **cellule** est un conteneur de texte ou de code. La cellule de type texte accepte le HTML et un markdown plus spécialisé. Toutes les variantes acceptées dans les cellules de code peut être trouvées ici:

https://www.dataquest.io/blog/jupyter-notebook-tutorial/

Le choix du type de cellule se fait dans le menu supérieur

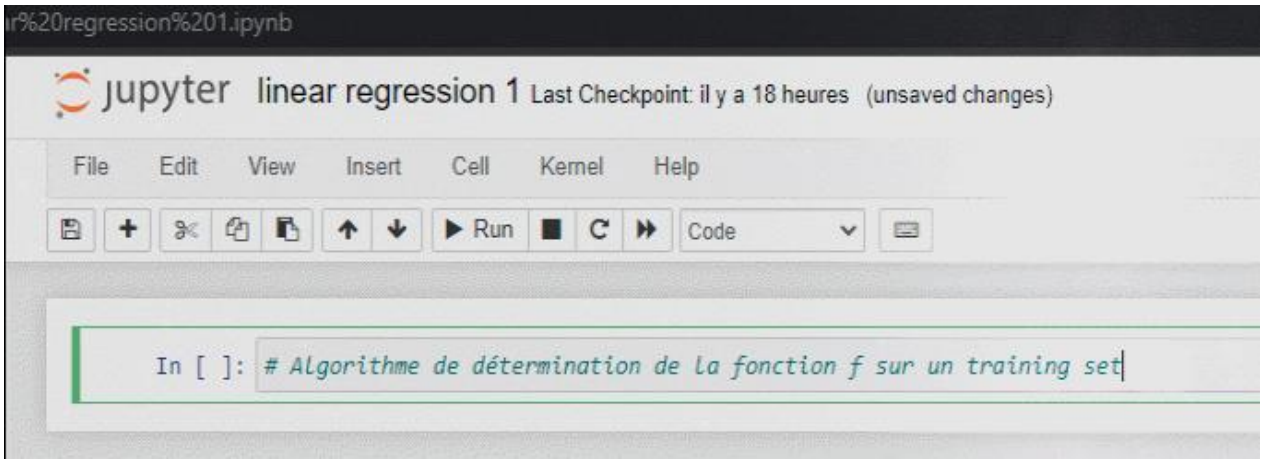


Ou bien en cliquant sur ‘’In[]’’ pour déselectionner la zone de texte de sorte que la bordure latérale gauche devienne bleue.

La bordure verte de la cellule signifie qu’elle est en mode ‘’Edit”. La bordure bleue signifie qu’elle est mode ‘’Command’’.

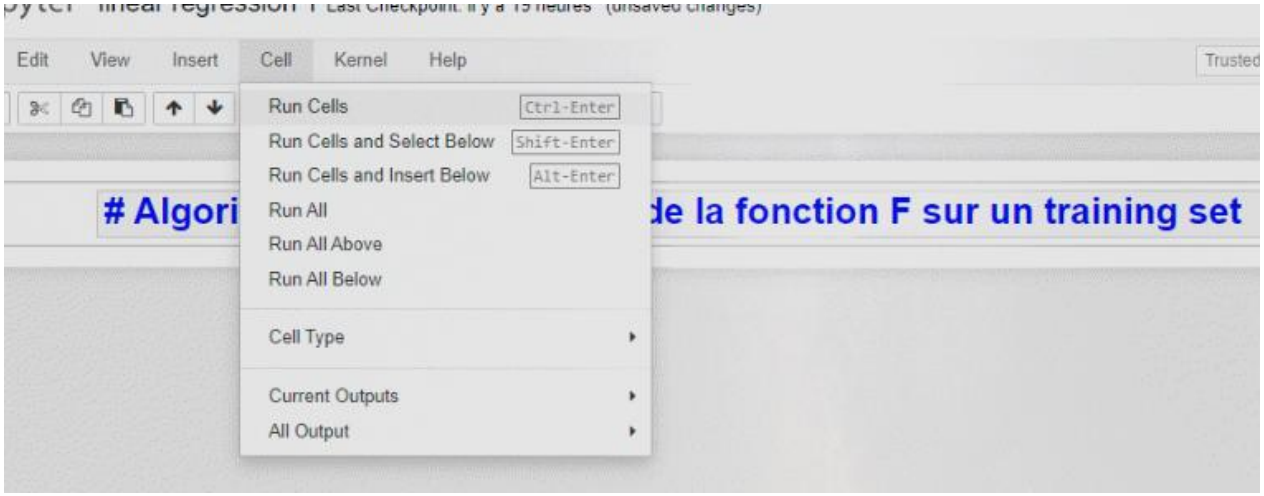
Puis, en tapant ‘’M’’ pour transformer la cellule en une cellule de texte (M arkdown) ou sur ‘’Y’’ pour transformer la cellule une cellule de code.

Testons:

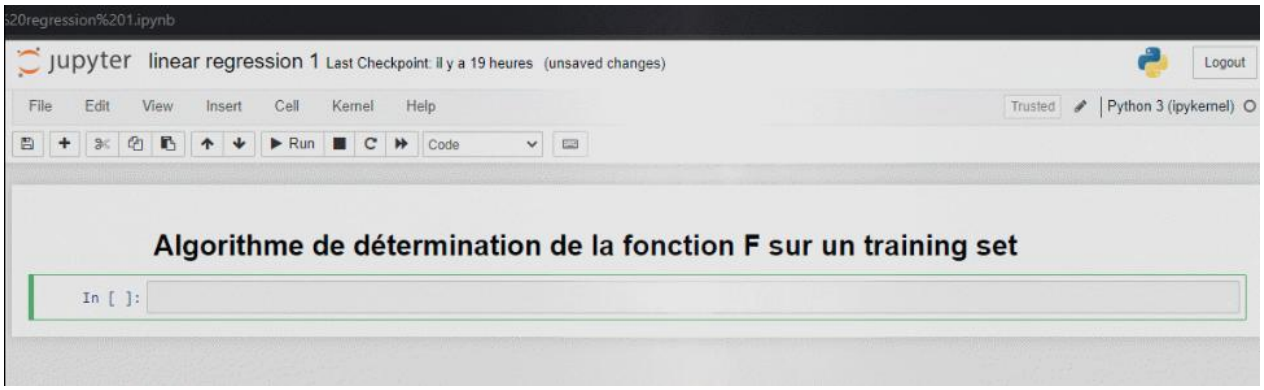


Puis cliquons hors de la cellule de sorte qu'elle passe en mode ''command''.

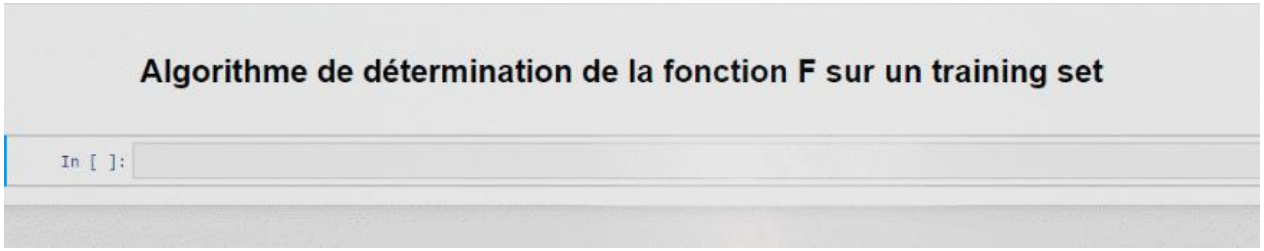
Pour exécuter la cellule, on utilise soit le menu supérieur cells > run cells



On peut aussi utiliser le raccourci clavier ‘’Alt Enter’’



Pour sauter une ligne, formattez la cellule courante en tant que cellule de texte puis tapez deux espaces vides (ou un
). Puis ‘’Alt Enter’’

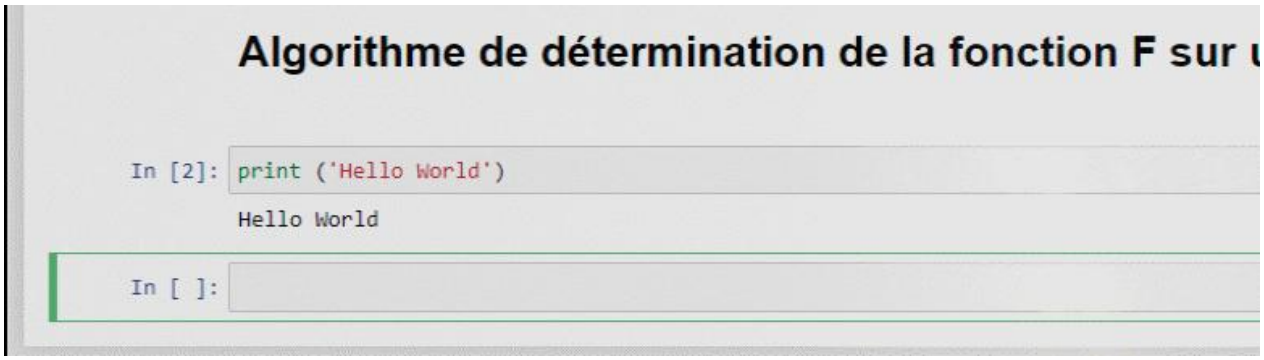


Pour supprimer une cellule, selectionnez la en mode ‘’Commande’’ puis tapez 2 fois sur ‘’D’’.

Testons maintenant les cellules de code:

Une nouvelle cellule est par défaut une cellule de code.
Entrez le code suivant:

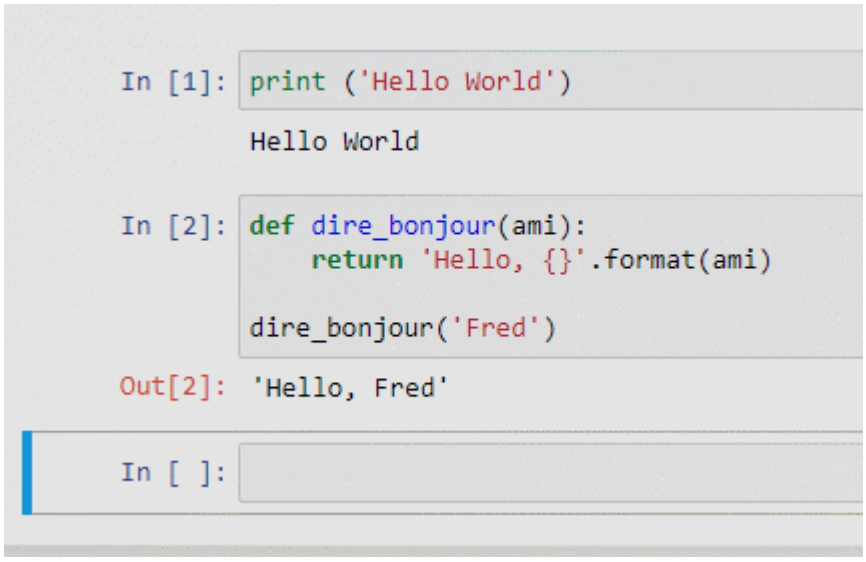
Puis ‘’Alt Enter’’



Le code de la cellule est exécuté par le noyau python.

Dans la cellule suivante tapons le code suivant :

puis ‘’Alt Enter’’



III. Les raccourcis claviers

Basculez entre le mode édition et le mode commande avec **Esc** et **Enter**, respectivement.
Une fois en mode commande :

- Faites défiler vos cellules de haut en bas avec les touches Up et Down
- Appuyez sur **A** ou **B** pour insérer une nouvelle cellule au-dessus ou en dessous de la cellule active.
- **M** transformera la cellule active en une cellule Markdown.
- **Y** définira la cellule active sur une cellule de code.
- **D + D (D deux fois)** supprimera la cellule active.
- **Z** annulera la suppression de la cellule.
- Maintenez enfoncé la touche **Shift** et appuyez sur **Up** ou **Down** pour sélectionner plusieurs cellules à la fois. Avec plusieurs cellules sélectionnées, **Shift + M** fusionnera votre sélection.
- **Ctrl + Shift + -**, en mode édition, divisera la cellule active au niveau du curseur.
- Vous pouvez également cliquer sur **Shift + Click** dans la marge à gauche de vos cellules pour les sélectionner.

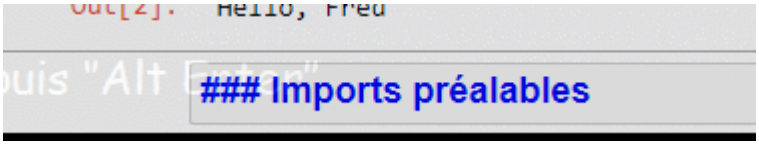
IV. L’exécution de code: Le noyau (kernel) Python

Derrière chaque notebook s'exécute un noyau. Lorsque vous exécutez une cellule de code, ce code est exécuté dans le noyau. Toute sortie est renvoyée à la cellule à afficher. L'état du noyau persiste dans le temps et entre les cellules - il concerne le document dans son ensemble et non des cellules individuelles.

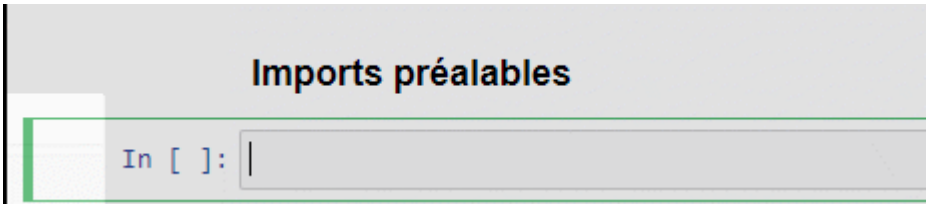
Par exemple, si vous importez des bibliothèques ou déclarez des variables dans une cellule, elles seront disponibles dans une autre. Essayons ceci pour nous faire une idée. Tout d'abord, nous allons importer des packages Python et définir une fonction :

Transformez la cellule courante avec ‘’m’’.

Puis tapez ### Imports préalables :



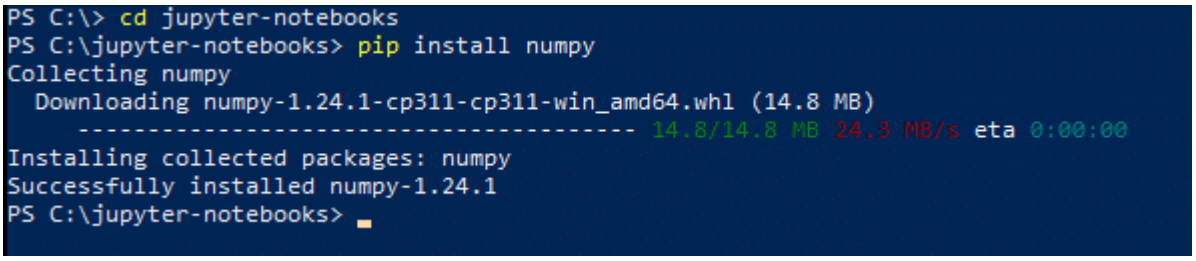
Puis ‘’Alt Enter’’



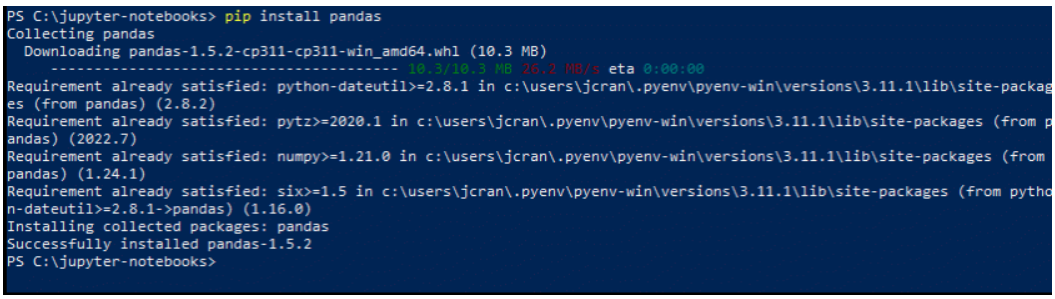
Installons maintenant les differents packages dont nous aurons besoin et certains packages que nous n’utiliserons pas tout de suite, mais dont vous pourriez avoir besoin plus tard.

Aller dans votre terminal puis

$> **pip install numpy**



$> **pip install pandas**



$> **pip install matplotlib**

```
PS C:\jupyter-notebooks> pip install matplotlib
Collecting matplotlib
  Downloading matplotlib-3.6.3-cp311-cp311-win_amd64.whl (7.2 MB)
     ---------------------------------------- 7.2/7.2 MB 19.2 MB/s eta 0:00:00
Collecting contourpy>=1.0.1
  Downloading contourpy-1.0.7-cp311-cp311-win_amd64.whl (162 kB)
     ---------------------------------------- 163.0/163.0 kB ? eta 0:00:00
Collecting cycler>=0.10
  Downloading cycler-0.11.0-py3-none-any.whl (6.4 kB)
Collecting fonttools>=4.22.0
  Downloading fonttools-4.38.0-py3-none-any.whl (965 kB)
     ---------------------------------------- 965.4/965.4 kB 30.8 MB/s eta 0:00:00
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.4.4-cp311-cp311-win_amd64.whl (55 kB)
     ---------------------------------------- 55.4/55.4 kB ? eta 0:00:00
Requirement already satisfied: numpy>=1.19 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (from ma
tplotlib) (1.24.1)
Requirement already satisfied: packaging>=20.0 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (fro
m matplotlib) (22.0)
Collecting pillow>=6.2.0
  Downloading Pillow-9.4.0-cp311-cp311-win_amd64.whl (2.5 MB)
     ---------------------------------------- 2.5/2.5 MB 26.5 MB/s eta 0:00:00
Collecting pyparsing>=2.2.1
  Using cached pyparsing-3.0.9-py3-none-any.whl (98 kB)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages
 (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (from pytho
n-dateutil>=2.7->matplotlib) (1.16.0)
Installing collected packages: pyparsing, pillow, kiwisolver, fonttools, cycler, contourpy, matplotlib
Successfully installed contourpy-1.0.7 cycler-0.11.0 fonttools-4.38.0 kiwisolver-1.4.4 matplotlib-3.6.3 pillow-9.4.0 pyp
arsing-3.0.9
PS C:\jupyter-notebooks>
```

$> **pip install seaborn**

```
PS C:\jupyter-notebooks> pip install seaborn
Collecting seaborn
  Downloading seaborn-0.12.2-py3-none-any.whl (293 kB)
     ---------------------------------------- 293.3/293.3 kB 9.1 MB/s eta 0:00:00
Requirement already satisfied: numpy!=1.24.0,>=1.17 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages
 (from seaborn) (1.24.1)
Requirement already satisfied: pandas>=0.25 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (from s
eaborn) (1.5.2)
Requirement already satisfied: matplotlib!=3.6.1,>=3.1 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packa
ges (from seaborn) (3.6.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (fr
om matplotlib!=3.6.1,>=3.1->seaborn) (1.0.7)
Requirement already satisfied: cycler>=0.10 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (from m
atplotlib!=3.6.1,>=3.1->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (f
rom matplotlib!=3.6.1,>=3.1->seaborn) (4.38.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (f
rom matplotlib!=3.6.1,>=3.1->seaborn) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (fro
m matplotlib!=3.6.1,>=3.1->seaborn) (22.0)
Requirement already satisfied: pillow>=6.2.0 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (from
matplotlib!=3.6.1,>=3.1->seaborn) (9.4.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (fr
om matplotlib!=3.6.1,>=3.1->seaborn) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages
 (from matplotlib!=3.6.1,>=3.1->seaborn) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (from p
andas>=0.25->seaborn) (2022.7)
Requirement already satisfied: six>=1.5 in c:\users\jcran\.pyenv\pyenv-win\versions\3.11.1\lib\site-packages (from pytho
n-dateutil>=2.7->matplotlib!=3.6.1,>=3.1->seaborn) (1.16.0)
Installing collected packages: seaborn
Successfully installed seaborn-0.12.2
PS C:\jupyter-notebooks>
```

$> **pip install Theano**

$> **pip install keras**

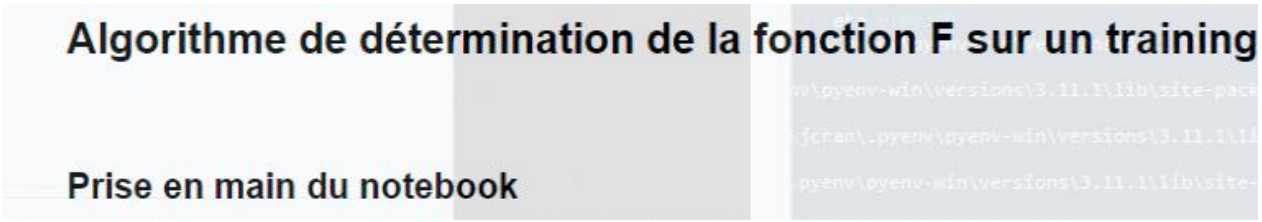
$> **pip install lab-utils**

$> **pip install scikit-learn**

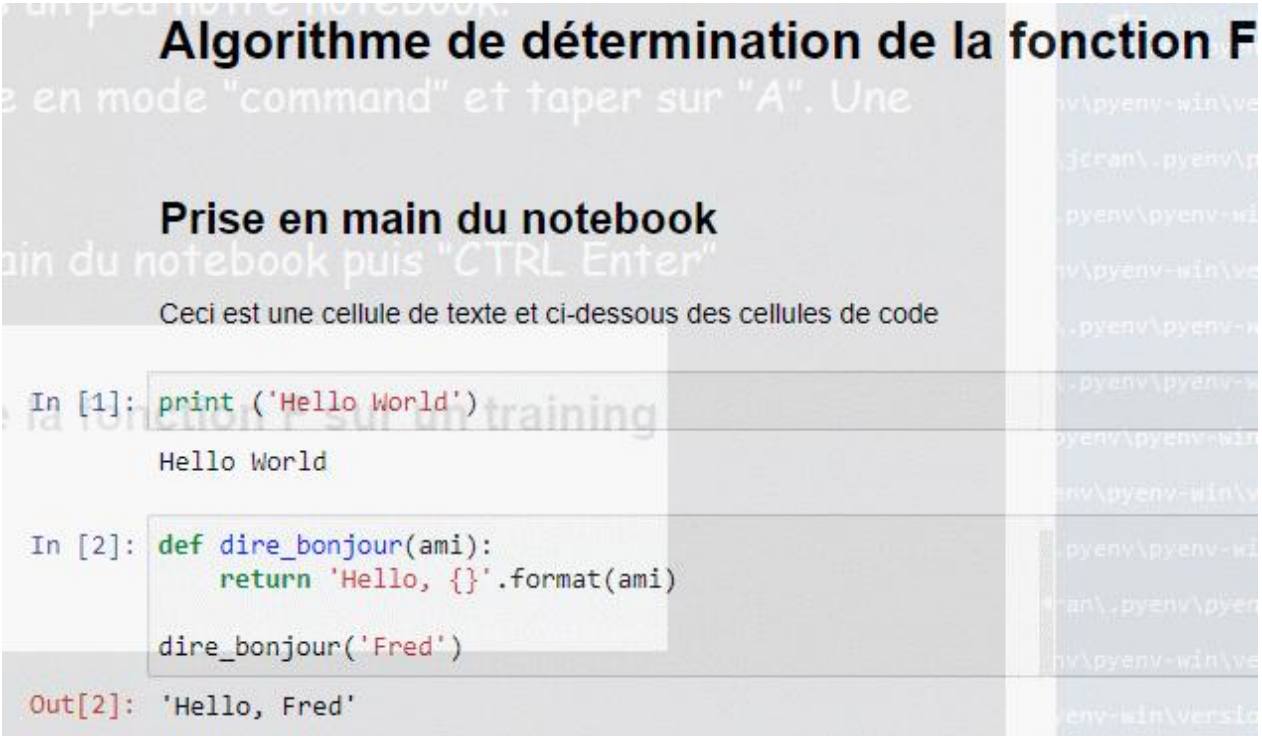
```
$> pip install ipympl
```

Dans la cellule courante taper le code suivant, puis ‘’Alt Enter’’:

Maintenant que cela est fait arrangeons un peu notre notebook. Sélectionnez la premiere cellule de code en mode ‘’command’’ et taper sur ‘’A’’. Une cellule vide apparait juste au dessus. Dans cette cellule taper ## Prise en main du notebook puis ‘’CTRL Enter’’

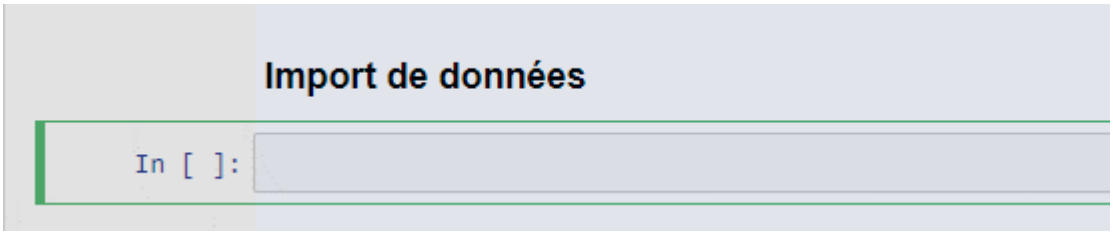


Re-selectionner la cellule contenant notre sous-titre et taper sur ‘’B’’. Une nouvelle cellule vierge apparait en dessous. Sélectionnez cette cellule en mode ‘’command’’ puis taper sur ‘’M’’ pour la transformer en cellule de texte. Entrez le texte : ‘’Ceci est une cellule de texte et ci-dessous, deux cellules de code’’.



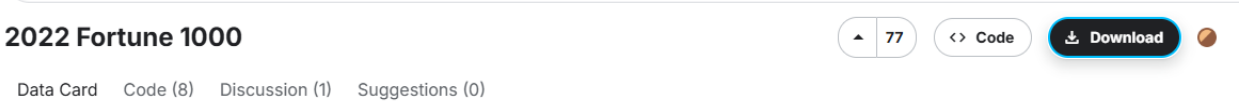
Revenons maintenant à la dernière cellule de notre notebook. Transformez-la en cellule de texte et taper dedans :

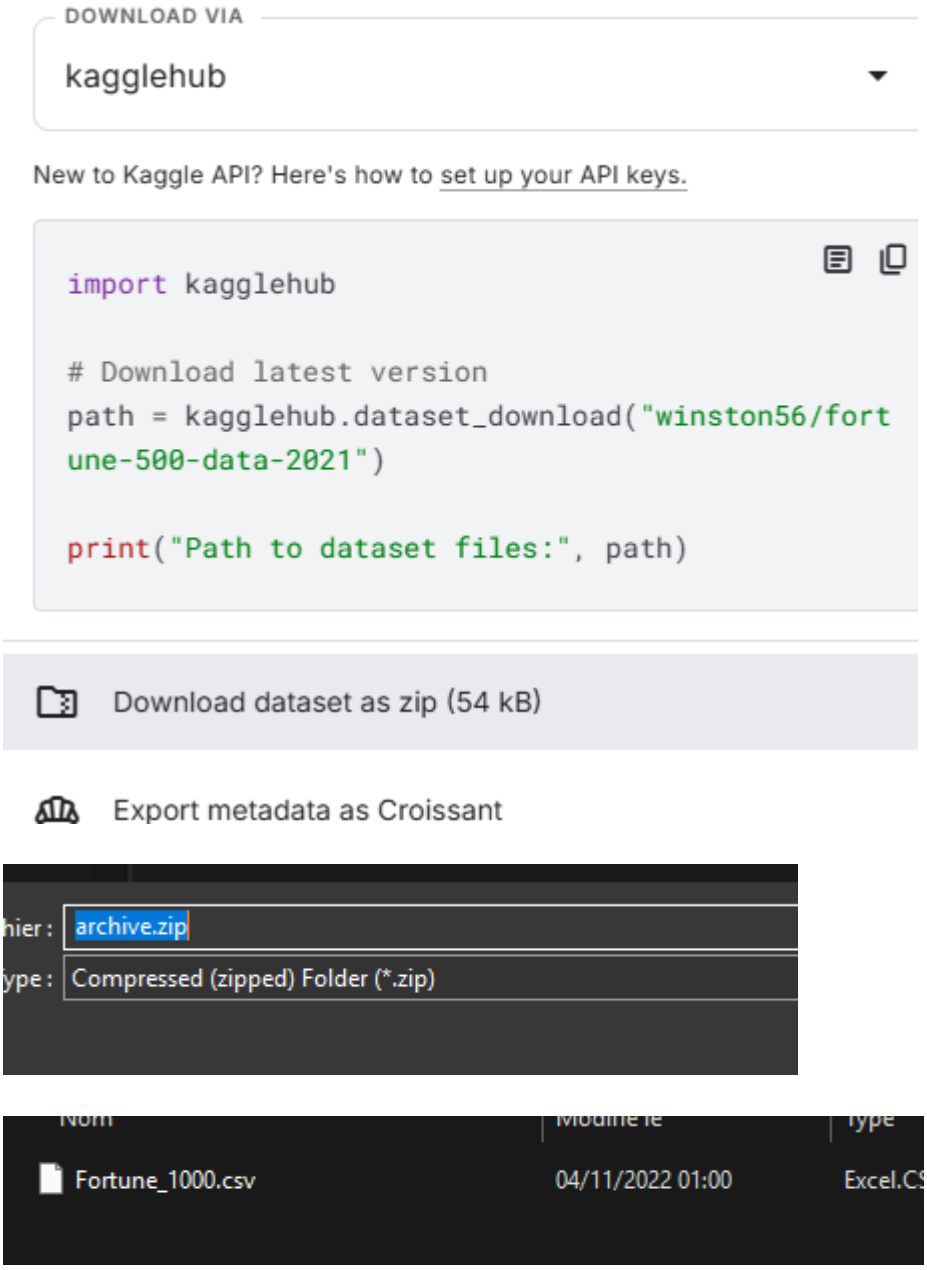
« ### Import de données » puis Alt Enter



Aller sur https://www.kaggle.com/datasets/winston56/fortune-500-data-2021

Télécharger en tant que zip:





Couper ce fichier et collez le dans le dossier « kaggle » créé au début de ce tuto.

Dans la cellule suivante taper le code suivant

puis ‘’Alt Enter’’

	company	rank	rank_change	revenue	profit	num. of employees	sector	city	state	newcomer	ceo_founder	ceo_woman	profitable	prev_rank
0	Walmart	1	0.0	572754.0	13673.0	2300000.0	Retailing	Bentonville	AR	no	no	no	yes	1.0
1	Amazon	2	0.0	469822.0	33364.0	1608000.0	Retailing	Seattle	WA	no	no	no	yes	2.0
2	Apple	3	0.0	365817.0	94680.0	154000.0	Technology	Cupertino	CA	no	no	no	yes	3.0
3	CVS Health	4	0.0	292111.0	7910.0	258000.0	Health Care	Woonsocket	RI	no	no	yes	yes	4.0
4	UnitedHealth Group	5	0.0	287597.0	17285.0	350000.0	Health Care	Minnetonka	MN	no	no	no	yes	5.0

Maintenant, nous roulons vraiment ! Notre bloc-notes est enregistré en toute sécurité et nous avons chargé notre ensemble de données « df » dans la structure de données pandas la plus utilisée, qui s'appelle une DataFrame et qui ressemble essentiellement à une table.

Dans la cellule suivante taper le code:

Puis ‘’Alt Enter’’

```
In [8]: df.tail()
```

Out[8]:

	company	rank	rank_change	revenue	profit	num. of employees	sector	city	state	newcomer	ceo_founder	ceo_woman	profitable	prev_ran
995	Vizio Holding	996	0.0	2124.0	-39.4	800.0	Industrials	Irvine	CA	no	yes	no	no	
996	1-800-Flowers.com	997	0.0	2122.2	118.7	4800.0	Retailing	Jericho	NY	no	no	no	yes	
997	Cowen	998	0.0	2112.8	295.6	1534.0	Financials	New York	NY	no	no	no	yes	
998	Ashland	999	0.0	2111.0	220.0	4100.0	Chemicals	Wilmington	DE	no	no	no	yes	
999	DocuSign	1000	0.0	2107.2	-70.0	7461.0	Technology	San Francisco	CA	no	no	no	no	

Notre dataset a 1000 entrées. Affichons le:

```
In [6]: len(df)
Out[6]: 1000

In [ ]:
```

Vérifions le type de chaque colonne :

```
In [9]: df.dtypes
Out[9]: company             object
        rank                 int64
        rank_change        float64
        revenue            float64
        profit             float64
        num. of employees  float64
        sector              object
        city                object
        state               object
        newcomer            object
        ceo_founder         object
        ceo_woman           object
        profitable          object
        prev_rank           object
        CEO                 object
        Website             object
        Ticker              object
        Market Cap          object
        dtype: object
```

Ajouter dans la nouvelle cellule code suivant:

```
In [10]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   company           1000 non-null   object
 1   rank              1000 non-null   int64
 2   rank_change       1000 non-null   float64
 3   revenue           1000 non-null   float64
 4   profit            997 non-null    float64
 5   num. of employees 999 non-null    float64
 6   sector            1000 non-null   object
 7   city              1000 non-null   object
 8   state             1000 non-null   object
 9   newcomer          1000 non-null   object
 10  ceo_founder       1000 non-null   object
 11  ceo_woman         1000 non-null   object
 12  profitable        1000 non-null   object
 13  prev_rank         1000 non-null   object
 14  CEO               1000 non-null   object
 15  Website           1000 non-null   object
 16  Ticker            951 non-null    object
 17  Market Cap        969 non-null    object
dtypes: float64(4), int64(1), object(13)
memory usage: 140.8+ KB
```

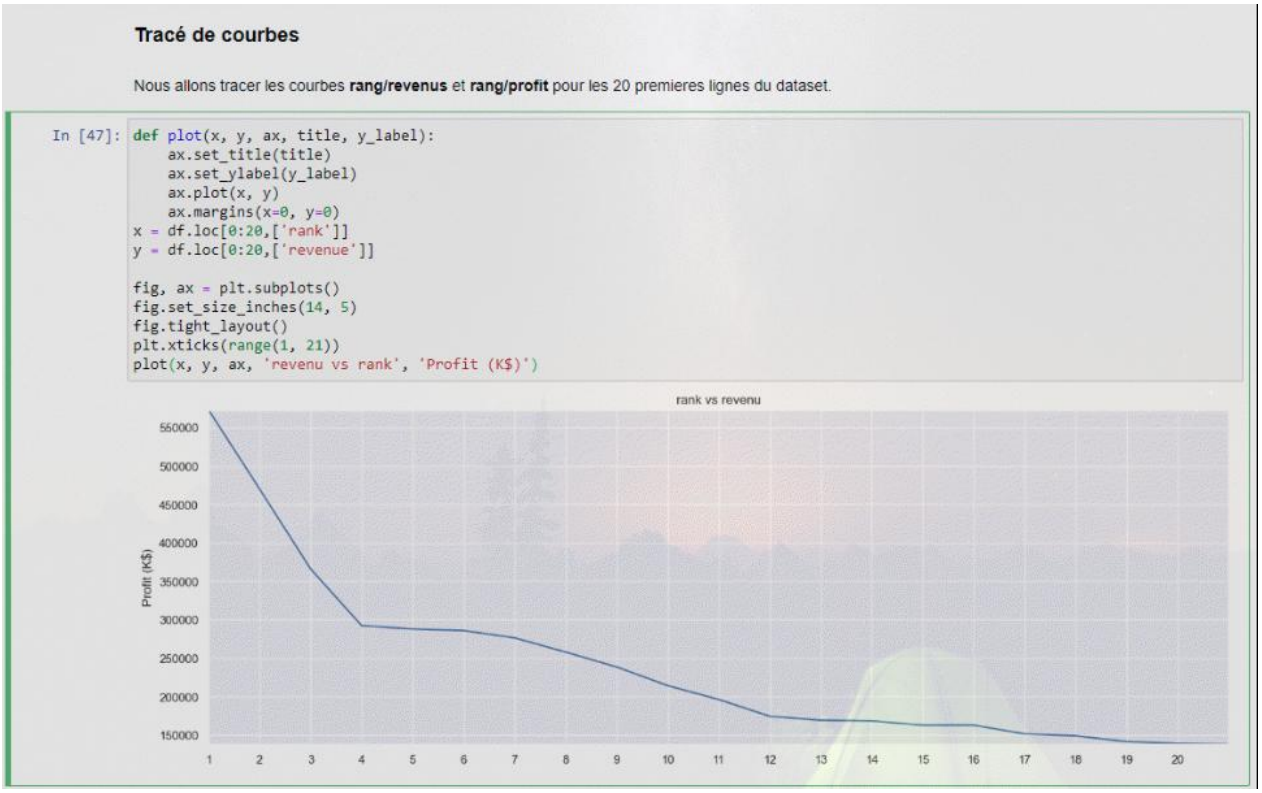
Ajouter une cellule texte contenant: ### Tracé de courbes
Puis une cellule de texte contenant:

« Nous allons tracer les courbes rang/revenus et rang/profit pour les 20 premieres lignes du dataset. »

Puis créer une cellule de code contenant le code suivant:

```
def plot(x, y, ax, title, y_label):
ax.set_title(title)
ax.set_ylabel(y_label)
ax.plot(x, y)
ax.margins(x=0, y=0)
x = df.loc[0:20,['rank']]
y = df.loc[0:20,['revenue']]

fig, ax = plt.subplots()
fig.set_size_inches(14, 5)
fig.tight_layout()
plt.xticks(range(1, 21))
plot(x, y, ax, 'revenu vs rank', 'Profit (K$)')
```

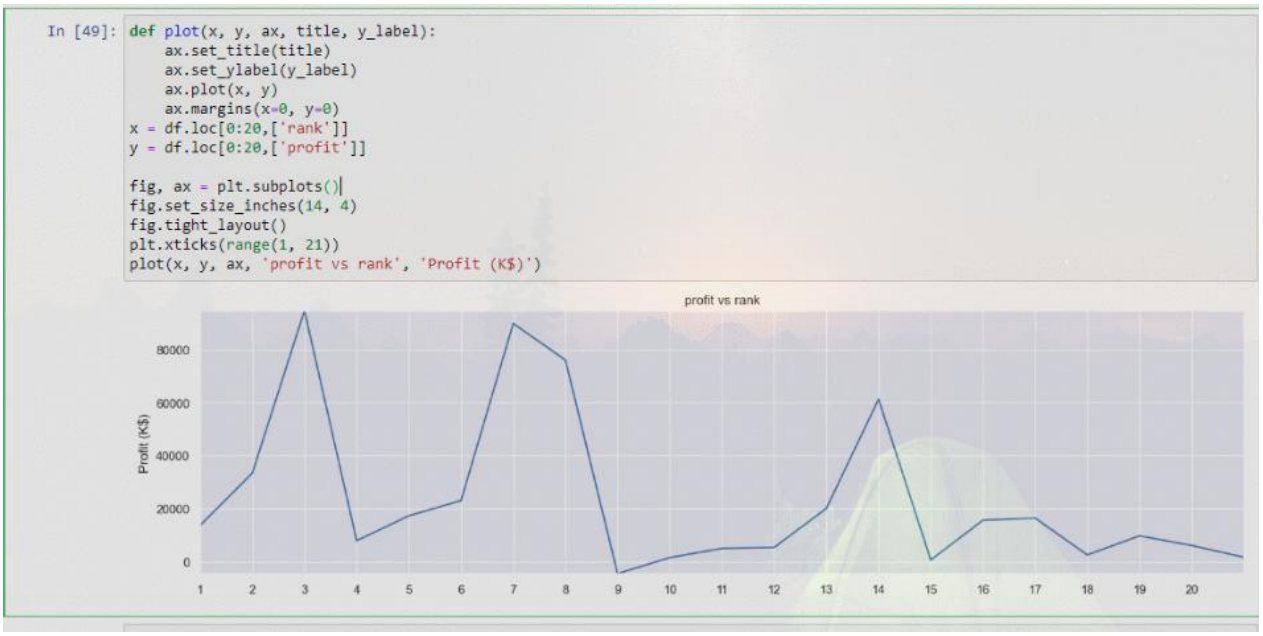
puis ‘’Alt Enter’’



Dans la cellule suivante taper le code suivant:

```
def plot(x, y, ax, title, y_label):
ax.set_title(title)
ax.set_ylabel(y_label)
ax.plot(x, y)
ax.margins(x=0, y=0)
x = df.loc[0:20,['rank']]
y = df.loc[0:20,['profit']]

fig, ax = plt.subplots()
fig.set_size_inches(14, 4)
fig.tight_layout()
plt.xticks(range(1, 21))
plot(x, y, ax, 'profit vs rank', 'Profit (K$)')
```

Puis ‘’Alt enter’’



V. Exportation d’un notebook

Jupyter a un support intégré pour l'exportation vers HTML et PDF ainsi que plusieurs autres formats, que vous pouvez trouver dans le menu sous "Fichier > Télécharger sous".
Si vous souhaitez partager vos cahiers avec un petit groupe privé, cette fonctionnalité pourrait bien être tout ce dont vous avez besoin.

En effet, comme de nombreux chercheurs dans les établissements universitaires disposent d'un espace Web public ou interne, et parce que vous pouvez exporter un bloc-notes vers un fichier HTML, Jupyter Notebooks peut être un moyen particulièrement pratique pour les chercheurs de partager leurs résultats avec leurs pairs.

Mais si le partage de fichiers exportés ne vous convient pas, il existe également des méthodes extrêmement populaires pour partager des fichiers .ipynb plus directement sur le Web.

Github

Suivre la procédure suivante:
https://reproducible-science-curriculum.github.io/sharing-RR-Jupyter/01-sharing-github/

NbViewer
voir ici: https://nbviewer.org/