The Way To Pandas

Prepared by

Eng . Mahmoud Soliman

جدول المحتويات

الدرس الأول : مقدمة عن مكتبة Pandas..... ٣
(Introduction to Pandas) ٣
الدرس الثاني : تنظيف البيانات..... ٢٢
(Data Cleaning)..... ٢٢
الدرس الثالث : العلاقة بين البيانات..... ٤٢
(Data Correlations)..... ٤٢
الدرس الرابع : الرسومات البيانية أو المُخططات..... ٤٤
(Plotting)..... ٤٤
الخاتمة..... ٤٩

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

الطريق إلى مكتبة Pandas
The Way to Pandas

الدرس الأول : مقدمة عن مكتبة Pandas
(Introduction to Pandas)

📖 Pandas Tutorial :

في هذا البرنامج التعليمي ، ستتعرف على أساسيات مكتبة داخل لغة البايثون وهى مكتبة Pandas. وسيتبع هذا الكتاب نفس تسلسل الدروس الخاصة بموقع W3Schools ، ويمكنكم الوصول لهذه السلسلة عن طريق https://www.w3schools.com/python/numpy_intro.asp.

➢ Pandas : مكتبة بايثون لتحليل البيانات analyze data

Pandas هي مكتبة Python تُستخدم للعمل مع مجموعات البيانات data sets وتحتوي هذه المكتبة على مجموعة من الدوال functions لتقوم بتحليل البيانات analyzing data وتنظيف البيانات cleaning data واستكشاف البيانات exploring data ومعالجة البيانات manipulating data .

تم إنشاء مكتبة pandas بواسطة WES McKinney في عام ٢٠٠٨ ، ويُعبر اسم "Pandas" عن كل من بيانات اللوحة "Panel Data" وتحليل بيانات البايثون " Python Data Analysis" .

➢ لماذا تستخدم Pandas ؟

تسمح لنا مكتبة Pandas بتحليل البيانات الكبيرة analyze big data وإجراء استنتاجات بناء على النظريات الإحصائية statistical theories ، ويمكن أن تقوم Pandas بتنظيف مجموعات البيانات غير المرتبة messy data sets ، وجعلها بيانات قابلة للقراءة readable وذات صلة relevant .

- **Relevant data : البيانات ذات الصلة أو البيانات المرتبطة**

البيانات ذات الصلة مهمة جدا في علوم البيانات data science .

- **Data Science : علم البيانات**

هو فرع مهم جداً من فروع علوم الكمبيوتر computer science والذي يدرس كيفية تخزين البيانات واستخدامها وتحليلها لاشتقاق المعلومات منها .

- **What Can Pandas Do ? : ماذا يمكننا أن نفعل بمكتبة Pandas**

تمنحك مكتبة Pandas إجابات مهمة جداً حول البيانات data على سبيل المثال :

- هل هناك علاقة correlation بين عمودين أو أكثر؟
- ما هو القيمة المتوسطة average value ؟
- قيمة الحد الأقصى Max value ؟
- قيمة الحد الأدنى Min value ؟

هذه المكتبة تقوم أيضاً بتنظيف البيانات cleaning the data مثل حذف الصفوف delete rows التي ليس لها صلة not relevant ، أو التي تحتوي على قيم خاطئة wrong values مثل القيم الفارغة empty values أو الفارغة NULL .

- **ما هو كود المصدر لمكتبة Pandas ؟**

Pandas هي مكتبة داخل لغة Python وهى مكتبة مفتوحة المصدر open source والكود المصدري لـ pandas موجود في مستودع repository على موقع GitHub يمكن الدخول عليه من خلال هذا الرابط https://github.com/pandas-dev/pandas.

GitHub يسمح للعديد من الأشخاص من العمل على نفس التعليمات البرمجية codebase.

➢ طريقة تثبيت مكتبة Pandas .

إذا كان لديك Python و PIP مثبتين بالفعل على نظام التشغيل ، فإن تثبيت Pandas سهل للغاية. قم بتثبيته باستخدام هذا الأمر داخل cmd : *pip install pandas* ، وبعد كتابة هذا الكود ستظهر تلك النتيجة التي تدل على نجاح العملية كما في الصورة :

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Hoda_PC>pip install pandas
Collecting pandas
  Downloading pandas-1.2.3-cp39-cp39-win_amd64.whl (9.3 MB)
     |[redacted]| 9.3 MB 258 kB/s
Requirement already satisfied: numpy>=1.16.5 in c:\users\hoda_pc\appdata\local\p
rograms\python\python39\lib\site-packages (from pandas) (1.20.1)
Collecting python-dateutil>=2.7.3
  Downloading python_dateutil-2.8.1-py2.py3-none-any.whl (227 kB)
     |[redacted]| 227 kB 1.7 MB/s
Collecting pytz>=2017.3
  Downloading pytz-2021.1-py2.py3-none-any.whl (510 kB)
     |[redacted]| 510 kB 726 kB/s
Requirement already satisfied: six>=1.5 in c:\users\hoda_pc\appdata\roaming\pyth
on\python39\site-packages (from python-dateutil>=2.7.3->pandas) (1.15.0)
Installing collected packages: python-dateutil, pytz, pandas
Successfully installed pandas-1.2.3 python-dateutil-2.8.1 pytz-2021.1
WARNING: You are using pip version 20.2.3; however, version 21.0.1 is available.

You should consider upgrading via the 'c:\users\hoda_pc\appdata\local\programs\p
ython\python39\python.exe -m pip install --upgrade pip' command.

C:\Users\Hoda_PC>
```

إذا فشل هذا الأمر فاستخدم أي توزيعة distribution من توزيعات python المثبت عليها بالفعل NumPy مثل Anaconda أو Spyder وما إلى ذلك .

الآن قمنا بتثبيت مكتبة pandas على نظام التشغيل الخاص بنا ، يمكننا الآن بسهولة عمل استيراد لتلك المكتبة داخل البرنامج الخاص بنا كما تعلمنا في درس استيراد الوحدات import modules .

```
import pandas
```

☒ في المثال السابق قمنا باستيراد مكتبة pandas وبالتالي يمكن استخدامها بسهولة .

✓ مثال ١ إنشاء جدول من البيانات :

```
import pandas

mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pandas.DataFrame(mydataset)
```

Output

```
    cars  passings
0    BMW         3
1  Volvo         7
2   Ford         2
```

☒ في المثال السابق قمنا بإنشاء جدول table من البيانات عن طريق إطار بيانات DataFrame.

♥ ملاحظات مهمة :

لا يمكن عمل جداول tables مباشراً من داخل pandas وإلا سيحدث خطأ.

✓ مثال ٢ إنشاء جدول Creating Table بطريقة ثانية :

```
import pandas as pd

mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pd.DataFrame(mydataset)
```

Output

```
    cars  passings
0    BMW         3
1  Volvo         7
2   Ford         2
```

☒ في المثال السابق قمنا باستيراد مكتبة pandas تحت الاسم المستعار Alias يسمى pd ، وبالتالي يمكن الإشارة إلى حزمة pandas على أنها pd بدلاً من pandas .

➢ Alias : الاسم المستعار

هو اسم بديل للإشارة إلى نفس الشيء.

✓ مثال ٣ لمعرفة آخر اصدار Pandas Version :

```
import pandas as pd

print(pd.__version__)
```

Output

```
1.2.3
```

☒ يمكن معرفة آخر اصدار version عن طريق خاصية __version__ ، حيث تحتوي هذه الخاصية على سلسلة الإصدار version string ويختلف الاصدار من وقت لآخر.

✓ مثال ٤ :

```
import pandas as pd

mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pd.DataFrame(mydataset)
print(myvar)
print(type(myvar))
```

Output

```
    cars  passings
0    BMW         3
1  Volvo         7
2   Ford         2
<class 'pandas.core.frame.DataFrame'>
```

☒ type() :هى دالة مضمنة داخل Python حيث تقوم بإرجاع نوع الكائن الذي تم تمريره إليه. كما هو الحال في الكود السابق فإنه يوضح أن pd هو نوع pandas.core.frame.DataFrame .

سلسلات بانداس Pandas Series

السلسلة Series هي بمثابة عمود column في جدول table ، وهي تمثل مصفوفة أحادية البعد 1-D تحتوي على بيانات من أي نوع .

➢ Series() function : دالة تقوم بتخصيص القيم كعمود داخل جدول .

✓ مثال :

```
import pandas as pd

a = [1, 7, 2]
myvar = pd.Series(a)
print(myvar)
```

Output

```
0    1
1    7
2    2
dtype: int64
```

Access items : الوصول إلى العناصر

يمكن الوصول إلى العناصر أو القيم بطريقتين :

- الطريق الأولى : عن طريق الفهرسة index .
- الطريقة الثانية : عن طريق التسمية label .

➢ index : الفهرس أو المؤشر

يمكن استخدام هذه الفهرس index للوصول إلى قيمة محددة.

✓ مثال الوصول للعناصر عن طريق index :

```
import pandas as pd

a = [1, 7, 2]
myvar = pd.Series(a)
print (myvar)
print("\n the value of index is : ",myvar[0])
```

Output

```
0    1
1    7
2    2
dtype: int64

 the value of index is :  1
```

☒ كما هو معروف يتم تسمية القيم برقم الفهرس الخاص بها ، فالقيمة الأولى لها مؤشر 0 ، والقيمة الثانية لها فهرس 1 وما إلى ذلك.

➢ Labels : الملصقات أو المسميات

يمكن استخدام التسمية label للوصول إلى قيمة محددة .

لكي نستخدم label يجب أولاً أن نقوم بإنشاء labels باستخدام معامل الفهرس index argument .

✓ مثال إنشاء مسميات Create Labels :

```
import pandas as pd

a = [1, 7, 2]
myvar = pd.Series(a, index = ["x", "y", "z"])
print(myvar)
```

Output

```
x    1
y    7
z    2
dtype: int64
```

✓ مثال الوصول للعناصر عن طريق label :

```
import pandas as pd

a = [1, 7, 2]
myvar = pd.Series(a, index = ["x", "y", "z"])
print(myvar)
print("\nthe value of lable is : ", myvar["y"])
```

Output

```
x    1
y    7
z    2
dtype: int64

the value of lable is :  7
```

☒ لاحظ أن هذه الطريقة myvar["y"] تساوى هذه الطريقة myvar.y

✓ مثال يمكن إنشاء series عن طريق القاموس dictionary :

```
import pandas as pd

calories = {"day1": 420, "day2": 380, "day3": 390}
myvar = pd.Series(calories)
print(myvar)
```

Output

```
day1    420
day2    380
day3    390
dtype: int64
```

☒ لاحظ أن keys تمثل التسميات labels .

♦ يمكن استخدم معامل argument index لتحديد بعض العناصر الموجودة داخل dictionary ، حيث نحدد فقط العناصر التي تريد تضمينها في المتسلسلة Series .

```
import pandas as pd

calories = {"day1": 420, "day2": 380, "day3": 390}
myvar = pd.Series(calories, index = ["day1", "day2"])
print(myvar)
```

Output

```
day1    420
day2    380
dtype: int64
```

☒ لاحظ في المثال السابق تم إنشاء series باستخدام البيانات فقط من "day1" و "day2" .

DataFrames : إطارات البيانات

هى عبارة عن مجموعات البيانات Data sets الموجودة داخل جداول متعددة الأبعاد أو ما يسمى بـ multi-dimensional tables في Pandas.

أو هى عبارة عن هيكلة بيانات data structure ثنائية الأبعاد 2-D مثل مصفوفة 2-D ، أو جدول به صفوف وأعمدة .

المتسلسلة Series تمثل العمود column وإطار البيانات DataFrame يمثل الجدول table بأكلمه

✓ مثال يمكن إنشاء DataFrame من متسلسلتين 2 series :

```
import pandas as pd

# Create a simple Data
data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

#load data into a DataFrame object:
df = pd.DataFrame(data)

print(df)
```

Output

```
   calories  duration
0       420        50
1       380        40
2       390        45
```

➢ loc attribute : خاصية تقوم بإرجاع صف row أو أكثر من الصفوف المحددة وهى اختصار لجملة حدد موقع الصف locate row .

✓ مثال :

```python
import pandas as pd

# Create a simple Data
data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

#load data into a DataFrame object:
df = pd.DataFrame(data)

print("return whole Pandas DataFrame")
print(df)

print("\nreturn Pandas Series")
print(df.loc[0])

print("\nreturn Pandas DataFrame")
print(df.loc[[0,1]])
```

Output

```
return whole Pandas DataFrame
   calories  duration
0       420        50
1       380        40
2       390        45

return Pandas Series
calories    420
duration     50
Name: 0, dtype: int64

return Pandas DataFrame
   calories  duration
0       420        50
1       380        40
```

لاحظ المثال السابق :

☒ في الجزء الأول : لم نستخدم index أو label ولذلك سترجع الجدول بالكامل .

☒ في الجزء الثاني : استخدمنا index مباشرة ولذلك سترجع متسلسلة Series .

☒ في الجزء الثالث : استخدمنا [ ] ولذلك سترجع DataFrame .

✓ مثال :

```python
import pandas as pd

# Create a simple Data
data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

#load data into a DataFrame object and Add a list of names to give each row a name:
df = pd.DataFrame(data, index = ["day1", "day2", "day3"])

print("return whole Pandas DataFrame")
print(df)

print("\nreturn Pandas Series")
print(df.loc["day2"])

print("\nreturn Pandas DataFrame")
print(df.loc[["day1","day2"]])
```

Output

```
return whole Pandas DataFrame
      calories  duration
day1       420        50
day2       380        40
day3       390        45

return Pandas Series
calories    380
duration     40
Name: day2, dtype: int64

return Pandas DataFrame
      calories  duration
day1       420        50
day2       380        40
```

☒ في المثال السابق قمنا بالدخول على العناصر (الصفوف) عن طريق label (named index) .

Load Files Into a DataFrame : تحميل الملفات إلى DataFrame

إذا تم تخزين مجموعات البيانات data sets الخاصة بك في ملف file ، فيمكن لـ Pandas تحميلها في DataFrame والقراءة منها . وسنتعرف على بعض أنواع الملفات التي يمكن القراءة منها .

◊ Read CSV Files : القراءة من ملفات بصيغة CSV

◊ Read JSON Files : القراءة من ملفات بصيغة JSON

أولاً : Read CSV Files : القراءة من ملفات بصيغة CSV

ملفات CSV هى نوع من الملفات الذي يسمح بتخزين مجموعات البيانات الضخمة وهى اختصار لعبارة (قيم مفصولة بفواصل Comma-Separated Values) .

تحتوي ملفات CSV على نص عادي plain text وهي تنسيق معروف جيدًا يمكن قراءته بواسطة الجميع بما في ذلك Pandas.

في أمثلتنا سنستخدم ملف CSV يسمى "data.csv" ويمكن تحميله من موقع W3Schools عن طريق هذا الرابط https://www.w3schools.com/python/pandas/data.csv.txt .

♥ شكل ملف CSV على الجهاز الخاص بنا :

	A	B	C	D	E	F	G	H
1	Duration	Pulse	Maxpulse	Calories				
2	60	110	130	409.1				
3	60	117	145	479				
4	60	103	135	340				
5	45	109	175	282.4				
6	45	117	148	406				
7	60	102	127	300				
8	60	110	136	374				
9	45	104	134	253.3				
10	30	109	133	195.1				
11	60	98	124	269				
12	60	103	147	329.3				

✓ مثال كيفية قراءة ملف بصيغة CSV (قراءة أول ٥ صفوف وآخر ٥ صفوف) :

```
import pandas as pd

df = pd.read_csv(r'E:\Way To Programming\Data Analysis\Python\data.csv')

print(df)
```

Output

```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
..        ...    ...       ...       ...
164        60    105       140     290.8
165        60    110       145     300.0
166        60    115       145     310.2
167        75    120       150     320.4
168        75    125       150     330.4

[169 rows x 4 columns]
```

➢ function read_csv(*path*) : دالة تقوم بقراءة مجموعة البيانات Data set .

☒ بشكل افتراضي عند طباعة DataFrame ، ستحصل فقط على أول ٥ صفوف ، وآخر ٥ صفوف.

✓ مثال كيفية قراءة ملف بصيغة CSV (قراءة الملف كاملاً) :

```
import pandas as pd

df = pd.read_csv(r'E:\Way To Programming\Data Analysis\Python\data.csv')

print(df.to_string())
```

Output

```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
5          60    102       127     300.0
6          60    110       136     374.0
7          45    104       134     253.3
8          30    109       133     195.1
```

➢ function to_string() : دالة تقوم بطباعة DataFrame بالكامل .

ثانياً : Read JSON Files : القراءة من ملفات بصيغة JSON

ملفات JSON هو تنسيق تبادل بيانات خفيف الوزن وهذا النوع من الملفات الذي يسمح غالبًا ما يتم تخزين مجموعات البيانات الضخمة أو استخراجها على هيئة JSON.وهى اختصار لعبارة (ترميز الكائنات باستعمال جافا سكريبت JavaScript Object Notation) .

ملفات JSON هى نص عادي plain text ، ولكن له تنسيق كائن Object ، وهو معروف جيدًا في عالم البرمجة ، بما في ذلك Pandas. وهي صيغة بسيطة وقابلة للقراءة بسهولة من قبل الإنسان وتستخدم لتمثيل البيانات بهدف سهولة تبادلها بين الأنظمة البرمجية المختلفة.

في أمثلتنا سنستخدم ملف JSON يسمى "data.json" ويمكن فتحه من موقع W3Schools عن طريق هذا الرابط https://www.w3schools.com/python/pandas/data.js .

◊ يتم تمثيل البيانات في JSON عن طريق جزءان أساسيان هما المفاتيح Keys والقيم Values.

- المفتاح Key : يمثل اسم فريد لقيمة البيانات ويتم وضعه عادة بين علامات التنصيص .
- القيمة Value : تمثل البيانات ويمكن أن تمثل اكثر من نوع بيانات .

يشكل Key / Value معًا سطر في صيغة JSON حيث يتم استخدام علامة (،) كفاصل بين السطور.

♥ شكل ملف JSON على الجهاز الخاص بنا :

```
{
  "Duration":{
    "0":60,
    "1":60,
    "2":60,
    "3":45,
    "4":45,
    "5":60,
    "6":60,
    "7":45,
    "8":30,
```

✓ مثال كيفية قراءة ملف بصيغة JSON (قراءة أول ٥ صفوف وآخر ٥ صفوف):

```
import pandas as pd

df = pd.read_json(r'E:\Way To Programming\Data Analysis\Python\data.json')

print(df)
```

Output

```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
..        ...    ...       ...       ...
164        60    105       140     290.8
165        60    110       145     300.4
166        60    115       145     310.2
167        75    120       150     320.4
168        75    125       150     330.4

[169 rows x 4 columns]
```

✓ مثال كيفية قراءة ملف بصيغة JSON (قراءة أول ٥ صفوف وآخر ٥ صفوف):

```
import pandas as pd

df = pd.read_json(r'E:\Way To Programming\Data Analysis\Python\data.json')

print(df.to_string())
```

Output

```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
5          60    102       127     300.0
6          60    110       136     374.0
7          45    104       134     253.3
8          30    109       133     195.1
```

☒ مما سبق يتضح لنا أن ملفات JSON هى عبارة عن قاموس Dictionary ، فكائنات JSON لها نفس التنسيق مثل قواميس Python وبالتالي إذا لم يكن كود JSON الخاص بك في ملف ، ولكنه موجود في قاموس Python ، فيمكنك تحميله في DataFrame مباشرة :

✓ مثال كيفية تحميل قاموس Python إلى DataFrame :

```
import pandas as pd

data = {
  "Duration":{
    "0":60,
    "1":60,
    "2":60,
    "3":45,
    "4":45,
    "5":60
  },
  "Pulse":{
    "0":110,
    "1":117,
    "2":103,
    "3":109,
    "4":117,
    "5":102
  },
  "Maxpulse":{
    "0":130,
    "1":145,
    "2":135,
    "3":175,
    "4":148,
    "5":127
  },
  "Calories":{
    "0":409,
    "1":479,
    "2":340,
    "3":282,
    "4":406,
    "5":300
  }
}

df = pd.DataFrame(data)

print(df)
```

Output

```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
5          60    102       127     300.0
```

Analyzing DataFrames : تحليل إطارات البيانات أو تحليل الجداول

يتم تحليل DataFrames عن طريق مجموعة من الدوال حيث يمكن الحصول على نظرة عامة سريعة على DataFrame من خلال الدوال الآتية :

- function head(*rows*) : دالة تقوم بعرض الرؤوس headers وعدد مُحدد من الصفوف rows بدءا من الأعلى وإذا لم نحدد عدد الصفوف فسترجع أعلى ٥ صفوف.

- function tail(*rows*) : دالة تقوم بعرض الرؤوس headers وعدد مُحدد من الصفوف rows ، بدءا من الأسفل ، وإذا لم يتم تحديد عدد الصفوف، فسترجع آخر ٥ صفوف .

✓ مثال :

```python
import pandas as pd

df = pd.read_csv(r'E:\Way To Programming\Data Analysis\Python\data.csv')

print(df.head(6))              #Print the first 6 rows of the DataFrame:

print("\n", df.head())         #Print the first 5 rows of the DataFrame:

print("\n", df.tail(3))        # printing the last 3 rows of the DataFrame:

print("\n", df.tail())         #Print the last 5 rows of the DataFrame:
```

Output

```
   Duration  Pulse  Maxpulse  Calories
0        60    110       130     409.1
1        60    117       145     479.0
2        60    103       135     340.0
3        45    109       175     282.4
4        45    117       148     406.0
5        60    102       127     300.0

    Duration  Pulse  Maxpulse  Calories
0        60    110       130     409.1
1        60    117       145     479.0
2        60    103       135     340.0
3        45    109       175     282.4
4        45    117       148     406.0

      Duration  Pulse  Maxpulse  Calories
166        60    115       145     310.2
167        75    120       150     320.4
168        75    125       150     330.4

      Duration  Pulse  Maxpulse  Calories
164        60    105       140     290.8
165        60    110       145     300.0
166        60    115       145     310.2
167        75    120       150     320.4
168        75    125       150     330.4
```

➢ function info(): دالة تقوم بعرض مزيد من المعلومات information حول مجموعة البيانات.

✓ مثال :

```
import pandas as pd

df = pd.read_csv(r'E:\Way To Programming\Data Analysis\Python\data.csv')

#Print information about the data:
print(df.info())
```

Output

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 169 entries, 0 to 168
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Duration  169 non-null    int64
 1   Pulse     169 non-null    int64
 2   Maxpulse  169 non-null    int64
 3   Calories  164 non-null    float64
dtypes: float64(1), int64(3)
memory usage: 5.4 KB
None
```

شرح الناتج السابق :

◊ Class : الفئة وهذه النتيجة تخبرنا بنوع البيانات data type لمجموعة البيانات التي تخصنا .

◊ <class 'pandas.core.frame.DataFrame'>

◊ RangeIndex : مؤشر المدى وهذه النتيجة تخبرنا أن هناك ١٦٩ صفوفا و ٤ أعمدة .

◊ RangeIndex: 169 entries, 0 to 168

◊ Data columns (total 4 columns):

◊ تعرض لنا اسم كل عمود ، مع نوع البيانات :

```
#   Column        Non-Null Count      Dtype
---  ------       --------------      -----
 0   Duration     169 non-null        int64
 1   Pulse        169 non-null        int64
 2   Maxpulse     169 non-null        int64
 3   Calories     164 non-null        float64
```

◊ تعرض لنا نوع البيانات :

◊ dtypes: float64(1), int64(3)

◊ تعرض لنا مساحة المستخدمة للملف :

◊ memory usage: 5.4 KB

☒ تخبرنا أيضاً دالة info() على القيم الفارغة Null Values وسنتعرف عليها في الجزء التالي .

Null Values : القيم الفارغة أو القيم الخالية

تخبرنا الدالة info() أيضا عدد القيم غير الفارغة Non-Null values الموجودة في كل عمود، وفي مجموعة بياناتنا، يبدو أن هناك ١٦٤ من ١٦٩ قيما غير فارغة في عمود " Calories" ، وهذا يعني أن هناك ٥ صفوف بدون قيمة Null Values على الإطلاق ، في عمود " Calories " لأي سبب من الأسباب.

القيم الفارغة Empty values أو القيم الخالية Null Values ، يمكن أن تكون سيئة عند تحليل البيانات، ويجب أن تفكر في إزالة الصفوف ذات قيم فارغة. هذه خطوة نحو ما يسمى بتنظيف البيانات cleaning data وسوف تتعلم المزيد عن ذلك في الفصول التالية.

الدرس الثاني : تنظيف البيانات
(Data Cleaning)

➢ Data Cleaning : تنظيف البيانات

تنظيف البيانات Data Cleaning يقصد بها تحديد البيانات السيئة في مجموعة البيانات ، والبيانات السيئة يمكن أن تكون كالآتي :

- خلايا فارغة Empty cells .
- البيانات بتنسيق خاطئ Data in wrong format .
- بيانات خاطئة Wrong data .
- الصفوف التكرارات Duplicated rows .

في هذا الدرس سوف نتعامل مع مجموعة البيانات التالية وسنتعلم كيفية التعامل مع كل نوع منهم .

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

☒ تحتوي مجموعة البيانات على بعض الخلايا الفارغة empty cells مثل ("Date" في الصف ٢٢، و "Calories" في الصف ١٨ والصف ٢٨).

☒ تحتوي مجموعة البيانات على تنسيق خاطئ wrong format مثل ("Date " في الصف ٢٦).

☒ تحتوي مجموعة البيانات على بيانات خاطئة wrong data مثل ("Duration " في الصف ٧).

☒ تحتوي مجموعة البيانات على التكرارات duplicates مثل (الصف ١١ و ١٢).

أولاً : Cleaning Empty Cells : تنظيف الخلايا الفارغة

- Empty Cells : الخلايا الفارغة
  يمكن أن تعطيك الخلايا الفارغة نتيجة خاطئة wrong result عند تحليل البيانات.

- function dropna() : تقوم هذه الدالة تقوم هذه الدالة بإزالة جميع الصفوف ذات القيم الخالية .
  وهذه الدالة تقوم بإرجاع Dataframe جديدة وبالتالي لن تتأثر المصفوفة الأصلية .

- function dropna(*inplace* = True) : تقوم هذه الدالة بإزالة جميع الصفوف ذات القيم الخالية.
  وهذه الدالة تقوم بإرجاع نفس الـ Dataframe الأصلية حيث وبالتالي ستتأثر المصفوفة الأصلية .

◊ **Remove Rows : إزالة الصفوف**

إحدى الطرق للتعامل مع الخلايا الفارغة هي إزالة الصفوف التي تحتوي على خلايا فارغة.

عادة ما يكون إزالة الصفوف موافقاً تماماً لمجموعة بياناتنا، لأن مجموعات البيانات قد تكون كبيرة جدا، وإزالة عدد قليل من الصفوف لن يكون لها تأثير كبير على النتيجة.

◊ Replace Empty Values : استبدال القيم الفارغة

هناك طريقة أخرى للتعامل مع الخلايا الفارغة هي إدراج قيمة جديدة بدلاً من ذلك.

وبهذه الطريقة، لا تضطر إلى حذف الصفوف بأكملها فقط بسبب بعض الخلايا الفارغة.

- function fillna(): تقوم هذه الدالة باستبدال الخلايا الفارغة بقيمة معينة.
  وهذه الدالة تقوم بإرجاع Dataframe جديدة وبالتالي لن تتأثر المصفوفة الأصلية .

- function fillna(*inplace* = True): تقوم هذه الدالة باستبدال الخلايا الفارغة بقيمة معينة.
  وهذه الدالة تقوم بإرجاع نفس الـ Dataframe الأصلية حيث وبالتالي ستتأثر المصفوفة الأصلية .

✓ مثال حذف الصفوف الفارغة مع عدم تأثر جدول البيانات Dataframe الأصلي :

```
import pandas as pd

df = pd.read_csv('data.csv')
new_df = df.dropna()

# Return a new Data Frame with no empty cells:
print(new_df.to_string())

#Notice in the result that some rows have been removed (row 18, 22 and 28).
#These rows had cells with empty values.
```

Output

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	2020/12/26	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

☒ هذه الصفوف كان لديها خلايا مع قيم فارغة.

☒ لاحظ الناتج output للمثال السابق للصفوف الآتية : (الصف ١٨ و ٢٢ و ٢٨) :
ستلاحظ أنه تم إزالة بعض الصفوف من Dataframe .

☒ لاحظ أن مع استخدام دالة fillna() فسنقوم بعمل Dataframe جديدة لأن Dataframe الأصلية لن تتغير لأننا لم نستخدم inplace=True . وبالتالي لو طبعنا قيمة DataFrame الأصلية فإنها ستظهر جميع القيم بدون تغير وستحتوي على القيم الفارغة.

✓ مثال حذف الصفوف الفارغة مع تأثر جدول البيانات Dataframe الأصلي :

```
import pandas as pd

df = pd.read_csv('data.csv')
new_df = df.dropna()

# Return the original Data Frame with no empty cells:
print(new_df.to_string())

#Notice in the result that some rows have been removed (row 18, 22 and 28).
#These rows had cells with empty values.
```

Output

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	2020/12/26	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

☒ هذه الصفوف كان لديها خلايا مع قيم فارغة.

☒ لاحظ الناتج output للمثال السابق للصفوف الآتية : (الصف ١٨ و ٢٢ و ٢٨) :
ستلاحظ أنه تم إزالة بعض الصفوف من Dataframe مثل.

☒ لاحظ أن مع استخدام دالة fillna(*inplace* = True) فسنقوم بإرجاع Dataframe الأصلية مع حذف القيم الفارغة وبالتالي لو طبعنا قيمة DataFrame الأصلية فإنها ستظهر جميع القيم بدون القيم الفارغة.

✓ مثال استبدال الصفوف الفارغة بقيم آخرى :

```
import pandas as pd

df = pd.read_csv(data.csv')
new_df = df.fillna(130)

print(new_df.to_string())
```

```
Duration          Date  Pulse  Maxpulse  Calories
  0         60  '2020/12/01'    110       130     409.1
  1         60  '2020/12/02'    117       145     479.0
  2         60  '2020/12/03'    103       135     340.0
  3         45  '2020/12/04'    109       175     282.4
  4         45  '2020/12/05'    117       148     406.0
  5         60  '2020/12/06'    102       127     300.0
  6         60  '2020/12/07'    110       136     374.0
  7        450  '2020/12/08'    104       134     253.3
  8         30  '2020/12/09'    109       133     195.1
  9         60  '2020/12/10'     98       124     269.0
  10        60  '2020/12/11'    103       147     329.3
  11        60  '2020/12/12'    100       120     250.7
  12        60  '2020/12/12'    100       120     250.7
  13        60  '2020/12/13'    106       128     345.3
  14        60  '2020/12/14'    104       132     379.3
  15        60  '2020/12/15'     98       123     275.0
  16        60  '2020/12/16'     98       120     215.2
  17        60  '2020/12/17'    100       120     300.0
  18        45  '2020/12/18'     90       112       130
  19        60  '2020/12/19'    103       123     323.0
  20        45  '2020/12/20'     97       125     243.0
  21        60  '2020/12/21'    108       131     364.2
  22        45           130    100       119     282.0
  23        60  '2020/12/23'    130       101     300.0
  24        45  '2020/12/24'    105       132     246.0
  25        60  '2020/12/25'    102       126     334.5
  26        60    2020/12/26    100       120     250.0
  27        60  '2020/12/27'     92       118     241.0
  28        60  '2020/12/28'    103       132       130
  29        60  '2020/12/29'    100       132     280.0
  30        60  '2020/12/30'    102       129     380.3
  31        60  '2020/12/31'     92       115     243.0
```

☒ هذه الصفوف كان لديها خلايا مع قيم فارغة.

☒ لاحظ الناتج output للمثال السابق للصفوف الآتية (الصف ١٨ و ٢٢ و ٢٨) :
ستلاحظ أنه تم استبدال القيم الفارغة بالقيمة ١٣٠.

✓ مثال استبدال الصفوف الفارغة بقيم آخرى لأعمدة معينة :

```
import pandas as pd

df = pd.read_csv('data.csv')
df["Calories"].fillna(130, inplace = True)
print(df.to_string())

#This operation inserts 130 in empty cells in the "Calories" column (row 18 and 28).
```

```
Duration          Date  Pulse  Maxpulse  Calories
  0         60  '2020/12/01'    110       130     409.1
  1         60  '2020/12/02'    117       145     479.0
  2         60  '2020/12/03'    103       135     340.0
  3         45  '2020/12/04'    109       175     282.4
  4         45  '2020/12/05'    117       148     406.0
  5         60  '2020/12/06'    102       127     300.0
  6         60  '2020/12/07'    110       136     374.0
  7        450  '2020/12/08'    104       134     253.3
  8         30  '2020/12/09'    109       133     195.1
  9         60  '2020/12/10'     98       124     269.0
  10        60  '2020/12/11'    103       147     329.3
  11        60  '2020/12/12'    100       120     250.7
  12        60  '2020/12/12'    100       120     250.7
  13        60  '2020/12/13'    106       128     345.3
  14        60  '2020/12/14'    104       132     379.3
  15        60  '2020/12/15'     98       123     275.0
  16        60  '2020/12/16'     98       120     215.2
  17        60  '2020/12/17'    100       120     300.0
  18        45  '2020/12/18'     90       112       130
  19        60  '2020/12/19'    103       123     323.0
  20        45  '2020/12/20'     97       125     243.0
  21        60  '2020/12/21'    108       131     364.2
  22        45           NaN    100       119     282.0
  23        60  '2020/12/23'    130       101     300.0
  24        45  '2020/12/24'    105       132     246.0
  25        60  '2020/12/25'    102       126     334.5
  26        60    2020/12/26    100       120     250.0
  27        60  '2020/12/27'     92       118     241.0
  28        60  '2020/12/28'    103       132       130
  29        60  '2020/12/29'    100       132     280.0
  30        60  '2020/12/30'    102       129     380.3
  31        60  '2020/12/31'     92       115     243.0
```

☒ لاحظ الناتج output للمثال السابق ١٣٠ للصفوف (الصف ١٨ و ٢٨) :

ستلاحظ أنه تم استبدال القيم الفارغة بالقيمة الموجودة في عمود "Calories".

♦ هناك طريقة شائعة لاستبدال الخلايا الفارغة، وهى عن طريق حساب قيمة المتوسط mean أو الوسيط median أو المنوال mode للأعمدة.

✓ مثال استبدال الصفوف الفارغة بقيمة المتوسط mean :

```
import pandas as pd

df = pd.read_csv('data.csv')

mean_value = df["Calories"].mean()

df["Calories"].fillna(mean_value, inplace=True)

print(df.to_string())
```

```
Duration          Date  Pulse  Maxpulse  Calories
  0        60  '2020/12/01'    110       130     409.1
  1        60  '2020/12/02'    117       145     479.0
  2        60  '2020/12/03'    103       135     340.0
  3        45  '2020/12/04'    109       175     282.4
  4        45  '2020/12/05'    117       148     406.0
  5        60  '2020/12/06'    102       127     300.0
  6        60  '2020/12/07'    110       136     374.0
  7       450  '2020/12/08'    104       134     253.3
  8        30  '2020/12/09'    109       133     195.1
  9        60  '2020/12/10'     98       124     269.0
  10       60  '2020/12/11'    103       147     329.3
  11       60  '2020/12/12'    100       120     250.7
  12       60  '2020/12/12'    100       120     250.7
  13       60  '2020/12/13'    106       128     345.3
  14       60  '2020/12/14'    104       132     379.3
  15       60  '2020/12/15'     98       123     275.0
  16       60  '2020/12/16'     98       120     215.2
  17       60  '2020/12/17'    100       120     300.0
  18       45  '2020/12/18'     90       112     304.68
  19       60  '2020/12/19'    103       123     323.0
  20       45  '2020/12/20'     97       125     243.0
  21       60  '2020/12/21'    108       131     364.2
  22       45           NaN    100       119     282.0
  23       60  '2020/12/23'    130       101     300.0
  24       45  '2020/12/24'    105       132     246.0
  25       60  '2020/12/25'    102       126     334.5
  26       60    2020/12/26    100       120     250.0
  27       60  '2020/12/27'     92       118     241.0
  28       60  '2020/12/28'    103       132     304.68
  29       60  '2020/12/29'    100       132     280.0
  30       60  '2020/12/30'    102       129     380.3
  31       60  '2020/12/31'     92       115     243.0
```

☒ لاحظ الناتج output للمثال السابق للصفوف (الصف ١٨ والصف ٢٨) :ستلاحظ أنه تم استبدال القيم الفارغة بقيمة المتوسط وهى القيمة ٣٠٤.٦٨ لعمود "Calories".

☒ المتوسط Mean يعني متوسط القيمة (وهو مجموع جميع القيم مقسوما على عدد القيم).

✓ مثال استبدال الصفوف الفارغة بقيمة الوسيط median :

```
import pandas as pd

df = pd.read_csv('data.csv')

mean_value = df["Calories"].median()

df["Calories"].fillna(mean_value, inplace=True)

print(df.to_string())
```

```
Duration          Date  Pulse  Maxpulse  Calories
  0        60  '2020/12/01'    110       130     409.1
  1        60  '2020/12/02'    117       145     479.0
  2        60  '2020/12/03'    103       135     340.0
  3        45  '2020/12/04'    109       175     282.4
  4        45  '2020/12/05'    117       148     406.0
  5        60  '2020/12/06'    102       127     300.0
  6        60  '2020/12/07'    110       136     374.0
  7       450  '2020/12/08'    104       134     253.3
  8        30  '2020/12/09'    109       133     195.1
  9        60  '2020/12/10'     98       124     269.0
  10       60  '2020/12/11'    103       147     329.3
  11       60  '2020/12/12'    100       120     250.7
  12       60  '2020/12/12'    100       120     250.7
  13       60  '2020/12/13'    106       128     345.3
  14       60  '2020/12/14'    104       132     379.3
  15       60  '2020/12/15'     98       123     275.0
  16       60  '2020/12/16'     98       120     215.2
  17       60  '2020/12/17'    100       120     300.0
  18       45  '2020/12/18'     90       112     291.2
  19       60  '2020/12/19'    103       123     323.0
  20       45  '2020/12/20'     97       125     243.0
  21       60  '2020/12/21'    108       131     364.2
  22       45           NaN    100       119     282.0
  23       60  '2020/12/23'    130       101     300.0
  24       45  '2020/12/24'    105       132     246.0
  25       60  '2020/12/25'    102       126     334.5
  26       60    2020/12/26    100       120     250.0
  27       60  '2020/12/27'     92       118     241.0
  28       60  '2020/12/28'    103       132     291.2
  29       60  '2020/12/29'    100       132     280.0
  30       60  '2020/12/30'    102       129     380.3
  31       60  '2020/12/31'     92       115     243.0
```

☒ لاحظ الناتج output للمثال السابق للصفوف (الصف ١٨ والصف ٢٨) :ستلاحظ أنه تم استبدال القيم الفارغة بقيمة الوسيط وهى القيمة ٢٩١.٢ لعمود "Calories".

☒ المتوسط Median يعني القيمة التي في المنتصف، بعد فرز جميع القيم تصاعدي ascending.

✓ مثال استبدال الصفوف الفارغة بقيمة المنوال mode :

```
import pandas as pd

df = pd.read_csv('data.csv')

mean_value = df["Calories"].mode()[0]

df["Calories"].fillna(mean_value, inplace=True)

print(df.to_string())
```

```
Duration          Date  Pulse  Maxpulse  Calories
  0        60  '2020/12/01'    110       130     409.1
  1        60  '2020/12/02'    117       145     479.0
  2        60  '2020/12/03'    103       135     340.0
  3        45  '2020/12/04'    109       175     282.4
  4        45  '2020/12/05'    117       148     406.0
  5        60  '2020/12/06'    102       127     300.0
  6        60  '2020/12/07'    110       136     374.0
  7       450  '2020/12/08'    104       134     253.3
  8        30  '2020/12/09'    109       133     195.1
  9        60  '2020/12/10'     98       124     269.0
  10       60  '2020/12/11'    103       147     329.3
  11       60  '2020/12/12'    100       120     250.7
  12       60  '2020/12/12'    100       120     250.7
  13       60  '2020/12/13'    106       128     345.3
  14       60  '2020/12/14'    104       132     379.3
  15       60  '2020/12/15'     98       123     275.0
  16       60  '2020/12/16'     98       120     215.2
  17       60  '2020/12/17'    100       120     300.0
  18       45  '2020/12/18'     90       112     300.0
  19       60  '2020/12/19'    103       123     323.0
  20       45  '2020/12/20'     97       125     243.0
  21       60  '2020/12/21'    108       131     364.2
  22       45           NaN    100       119     282.0
  23       60  '2020/12/23'    130       101     300.0
  24       45  '2020/12/24'    105       132     246.0
  25       60  '2020/12/25'    102       126     334.5
  26       60    2020/12/26    100       120     250.0
  27       60  '2020/12/27'     92       118     241.0
  28       60  '2020/12/28'    103       132     300.0
  29       60  '2020/12/29'    100       132     280.0
  30       60  '2020/12/30'    102       129     380.3
  31       60  '2020/12/31'     92       115     243.0
```

☒ لاحظ الناتج output للمثال السابق للصفوف (الصف ١٨ والصف ٢٨) :ستلاحظ أنه تم استبدال القيم الفارغة بقيمة المنوال وهى القيمة ٣٠٠.٠ لعمود "Calories". وهى القيمة الأكثر تكرارً حيث تكررت في الصف ١٨ و ٢٨ .

☒ المنوال Mode يعني القيمة الاكثر تكراراً .

ثانياً : Cleaning Data of Wrong Format : تنظيف البيانات التي لها تنسيق خاطئ

Wrong Format : التنسيق الخاطئ أو الشكل الخطأ

يمكن أن تعطيك الخلايا التي لها تنسيقات خطأ أو تنسيق غير مناسب نتيجة خاطئة wrong result عند تحليل البيانات.ولإصلاح تلك التنسيقات ، لديك خياران :

١-قم بإزالة الصفوف remove the rows.

٢-تحويل الخلايا في الأعمدة إلى نفس التنسيق Convert Into a Correct Format .

◊ Remove Rows : إزالة الصفوف

إحدى الطرق للتعامل مع التنسيق الغير صحيح هي إزالة الصفوف التي تحتوي على خلايا فارغة.

عادة ما يكون إزالة الصفوف موافقاً تماماً لمجموعة بياناتنا، لأن مجموعات البيانات قد تكون كبيرة جدا، وإزالة عدد قليل من الصفوف لن يكون لها تأثير كبير على النتيجة.

◊ Convert Into a Correct Format : تحويل إلى التنسيق الصحيح

هناك طريقة أخرى للتعامل مع التنسيق الغير صحيح وهي تحويل التنسيق الخطأ إلى التنسيق الصحيح في كل عمود.

على سبيل المثال في Data Frame الخاص بنا ، لدينا خلايا ذات تنسيق خاطئ wrong format. تحقق من الصف ٢٢ والصف ٢٦، يجب أن يكون عمود "Data" سلسلة تمثل تاريخ data.

19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

دعونا نحاول تحويل جميع الخلايا في عمود "Date" إلى تاريخ "datetime" .

➢ function to_datetime(*column*) : تقوم هذه الدالة بتحويل العمود إلى نوع تاريخ datetime.

✓ مثال تحويل جميع الخلايا في عمود "Date" إلى نوع بيانات يسمى تاريخ "datetime"

```
import pandas as pd

df = pd.read_csv('data.csv')

df['Date'] = pd.to_datetime(df['Date'])

print(df.to_string())
```

```
    Duration       Date  Pulse  Maxpulse  Calories
0         60 2020-12-01    110       130     409.1
1         60 2020-12-02    117       145     479.0
2         60 2020-12-03    103       135     340.0
3         45 2020-12-04    109       175     282.4
4         45 2020-12-05    117       148     406.0
5         60 2020-12-06    102       127     300.0
6         60 2020-12-07    110       136     374.0
7        450 2020-12-08    104       134     253.3
8         30 2020-12-09    109       133     195.1
9         60 2020-12-10     98       124     269.0
10        60 2020-12-11    103       147     329.3
11        60 2020-12-12    100       120     250.7
12        60 2020-12-12    100       120     250.7
13        60 2020-12-13    106       128     345.3
14        60 2020-12-14    104       132     379.3
15        60 2020-12-15     98       123     275.0
16        60 2020-12-16     98       120     215.2
17        60 2020-12-17    100       120     300.0
18        45 2020-12-18     90       112       NaN
19        60 2020-12-19    103       123     323.0
20        45 2020-12-20     97       125     243.0
21        60 2020-12-21    108       131     364.2
22        45        NaT    100       119     282.0
23        60 2020-12-23    130       101     300.0
24        45 2020-12-24    105       132     246.0
25        60 2020-12-25    102       126     334.5
26        60 2020-12-26    100       120     250.0
27        60 2020-12-27     92       118     241.0
28        60 2020-12-28    103       132       NaN
29        60 2020-12-29    100       132     280.0
30        60 2020-12-30    102       129     380.3
31        60 2020-12-31     92       115     243.0
```

☒ كما ترون من الناتج السابق ، فإن التاريخ في الصف ٢٦ حيث تم إصلاحه، ولكن التاريخ الفارغ في الصف ٢٢ حصل على قيمة NaT (وليس يعبر عن الوقت)، وبعبارة أخرى قيمة فارغة. طريقة واحدة للتعامل مع القيم الفارغة هي ببساطة إزالة الصف بأكمله.

✓ مثال معالجة التنسيقات الخاطئة :

```python
import pandas as pd

df = pd.read_csv('data.csv')

df['Date'] = pd.to_datetime(df['Date'])

df.dropna(subset=['Date'], inplace = True)

print(df.to_string())
```

```
    Duration       Date  Pulse  Maxpulse  Calories
0         60 2020-12-01    110       130     409.1
1         60 2020-12-02    117       145     479.0
2         60 2020-12-03    103       135     340.0
3         45 2020-12-04    109       175     282.4
4         45 2020-12-05    117       148     406.0
5         60 2020-12-06    102       127     300.0
6         60 2020-12-07    110       136     374.0
7        450 2020-12-08    104       134     253.3
8         30 2020-12-09    109       133     195.1
9         60 2020-12-10     98       124     269.0
10        60 2020-12-11    103       147     329.3
11        60 2020-12-12    100       120     250.7
12        60 2020-12-12    100       120     250.7
13        60 2020-12-13    106       128     345.3
14        60 2020-12-14    104       132     379.3
15        60 2020-12-15     98       123     275.0
16        60 2020-12-16     98       120     215.2
17        60 2020-12-17    100       120     300.0
18        45 2020-12-18     90       112       NaN
19        60 2020-12-19    103       123     323.0
20        45 2020-12-20     97       125     243.0
21        60 2020-12-21    108       131     364.2
23        60 2020-12-23    130       101     300.0
24        45 2020-12-24    105       132     246.0
25        60 2020-12-25    102       126     334.5
26        60 2020-12-26    100       120     250.0
27        60 2020-12-27     92       118     241.0
28        60 2020-12-28    103       132       NaN
29        60 2020-12-29    100       132     280.0
30        60 2020-12-30    102       129     380.3
31        60 2020-12-31     92       115     243.0
```

☒ كما ترون من الناتج السابق ، فإن التاريخ في الصف ٢٦ تم إصلاحه، وأيضاً التاريخ الفارغ في الصف ٢٢ الذي كان يحتوي على قيمة NaT (وليس يعبر عن الوقت) تم إصلاحه أيضاً عن طريق دالة الحذف القيم الخالية dropna() .

ثالثاً : Fixing Wrong Data : إصلاح البيانات الخطأ

Wrong Data : البيانات الخطأ

"البيانات الخاطئة **Data** Wrong " لم تكن فقط "خلايا فارغة cells empty" أو "تنسيق خاطئ wrong format"، ولكن قد يكون الخطأ لو أن شخصا ما سجل "199" بدلا من "1.99".

في بعض الأحيان يمكنك اكتشاف بيانات خاطئة من خلال النظر في مجموعة البيانات، لأن لديك توقعات لما يجب أن يكون عليه. إذا ألقيت نظرة على مجموعة البيانات data set ، فسترى أن في الصف ٧ عمود "Duration" أن قيمته تساوي ٤٥٠، ولكن لجميع الصفوف الأخرى تتراوح المدة بين ٣٠ و ٦٠.

لا يتعين عليه أن تكون مخطئا، ولكن مع الأخذ في الاعتبار أن هذه هي مجموعة البيانات من جلسات تجريب شخص ما، نستنتج مع حقيقة أن هذا الشخص لم ينجح في ٤٥٠ دقيقة.

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

♦ يمكن أن تعطيك الخلايا التي لها بيانات خطأ نتيجة خاطئة wrong result عند تحليل البيانات.ولإصلاح تلك البيانات الخطأ ، لديك خياران :

١- قم بإزالة الصفوف remove the rows.

٢- تحويل الخلايا في الأعمدة إلى نفس التنسيق Convert Into a Correct Format .

◊ Remove Rows : إزالة الصفوف

إحدى الطرق للتعامل مع البيانات الغير صحيحة هي إزالة الصفوف التي تحتوي على خلايا فارغة.

◊ Replacing Values: استبدال القيم

طريقة واحدة لإصلاح القيم الخاطئة وهي عن طريق استبدالها بشيء آخر.

فمثلاً في Data Frame الخاص بنا ، من المرجح أن تكون هناك خطأ مطبعي typo ، ويجب أن تكون القيمة "٤٥" بدلا من "٤٥٠"، ويمكننا فقط إدراج "٤٥" في الصف ٧.

✓ مثال استبدال القيم الغير صحيحة :

```
import pandas as pd

df = pd.read_csv('data.csv')

df.loc[7,'Duration'] = 45

print(df.to_string())
```

```
    Duration          Date  Pulse  Maxpulse  Calories
0         60  '2020/12/01'    110       130     409.1
1         60  '2020/12/02'    117       145     479.0
2         60  '2020/12/03'    103       135     340.0
3         45  '2020/12/04'    109       175     282.4
4         45  '2020/12/05'    117       148     406.0
5         60  '2020/12/06'    102       127     300.0
6         60  '2020/12/07'    110       136     374.0
7         45  '2020/12/08'    104       134     253.3
8         30  '2020/12/09'    109       133     195.1
9         60  '2020/12/10'     98       124     269.0
10        60  '2020/12/11'    103       147     329.3
11        60  '2020/12/12'    100       120     250.7
12        60  '2020/12/12'    100       120     250.7
13        60  '2020/12/13'    106       128     345.3
14        60  '2020/12/14'    104       132     379.3
15        60  '2020/12/15'     98       123     275.0
16        60  '2020/12/16'     98       120     215.2
17        60  '2020/12/17'    100       120     300.0
18        45  '2020/12/18'     90       112       NaN
19        60  '2020/12/19'    103       123     323.0
```

☒ بالنسبة إلى مجموعات البيانات الصغيرة، قد تكون قادرا على استبدال البيانات الخاطئة واحدة تلو الأخرى، ولكن ليس لمجموعات بيانات كبيرة.

- لاستبدال البيانات الخاطئة لمجموعات بيانات أكبر، يمكنك إنشاء بعض القواعد والجمل الشرطية ، على سبيل المثال اضبط بعض الحدود للقيم القانونية، واستبدال أي قيم خارج الحدود.

✓ مثال استبدال القيم الغير صحيحة عن طريق بعض الشروط :

```
import pandas as pd

df = pd.read_csv('data.csv')

for x in df.index:
  if df.loc[x, "Duration"] > 120:
    df.loc[x, "Duration"] = 120

print(df.to_string())
```

```
    Duration          Date  Pulse  Maxpulse  Calories
0         60  '2020/12/01'    110       130     409.1
1         60  '2020/12/02'    117       145     479.0
2         60  '2020/12/03'    103       135     340.0
3         45  '2020/12/04'    109       175     282.4
4         45  '2020/12/05'    117       148     406.0
5         60  '2020/12/06'    102       127     300.0
6         60  '2020/12/07'    110       136     374.0
7        120  '2020/12/08'    104       134     253.3
8         30  '2020/12/09'    109       133     195.1
9         60  '2020/12/10'     98       124     269.0
10        60  '2020/12/11'    103       147     329.3
11        60  '2020/12/12'    100       120     250.7
12        60  '2020/12/12'    100       120     250.7
13        60  '2020/12/13'    106       128     345.3
14        60  '2020/12/14'    104       132     379.3
15        60  '2020/12/15'     98       123     275.0
16        60  '2020/12/16'     98       120     215.2
17        60  '2020/12/17'    100       120     300.0
18        45  '2020/12/18'     90       112       NaN
19        60  '2020/12/19'    103       123     323.0
20        45  '2020/12/20'     97       125     243.0
21        60  '2020/12/21'    108       131     364.2
22        45           NaN    100       119     282.0
23        60  '2020/12/23'    130       101     300.0
24        45  '2020/12/24'    105       132     246.0
25        60  '2020/12/25'    102       126     334.5
26        60      20201226    100       120     250.0
27        60  '2020/12/27'     92       118     241.0
28        60  '2020/12/28'    103       132       NaN
29        60  '2020/12/29'    100       132     280.0
30        60  '2020/12/30'    102       129     380.3
31        60  '2020/12/31'     92       115     243.0
```

☒ لاحظ قمنا بعمل حلقة تكرارية loop من خلال جميع القيم في عمود "Duration". إذا كانت القيمة أعلى من ١٢٠، فسنقوم بتغيريها وتعينها إلى ١٢٠ .

✓ مثال حذف الصفوف التي لها قيم غير صحيحة عن طريق بعض الشروط :

```
import pandas as pd

df = pd.read_csv('data.csv')

for x in df.index:
  if df.loc[x, "Duration"] > 120:
    df.drop(x, inplace = True)

#remember to include the 'inplace = True' argument to make the changes in the origina
l DataFrame object instead of returning a copy

print(df.to_string())
```

```
    Duration          Date  Pulse  Maxpulse  Calories
0         60  '2020/12/01'    110       130     409.1
1         60  '2020/12/02'    117       145     479.0
2         60  '2020/12/03'    103       135     340.0
3         45  '2020/12/04'    109       175     282.4
4         45  '2020/12/05'    117       148     406.0
5         60  '2020/12/06'    102       127     300.0
6         60  '2020/12/07'    110       136     374.0
8         30  '2020/12/09'    109       133     195.1
9         60  '2020/12/10'     98       124     269.0
10        60  '2020/12/11'    103       147     329.3
11        60  '2020/12/12'    100       120     250.7
12        60  '2020/12/12'    100       120     250.7
13        60  '2020/12/13'    106       128     345.3
14        60  '2020/12/14'    104       132     379.3
15        60  '2020/12/15'     98       123     275.0
16        60  '2020/12/16'     98       120     215.2
17        60  '2020/12/17'    100       120     300.0
18        45  '2020/12/18'     90       112       NaN
19        60  '2020/12/19'    103       123     323.0
20        45  '2020/12/20'     97       125     243.0
21        60  '2020/12/21'    108       131     364.2
22        45           NaN    100       119     282.0
23        60  '2020/12/23'    130       101     300.0
24        45  '2020/12/24'    105       132     246.0
25        60  '2020/12/25'    102       126     334.5
26        60      20201226    100       120     250.0
27        60  '2020/12/27'     92       118     241.0
28        60  '2020/12/28'    103       132       NaN
29        60  '2020/12/29'    100       132     280.0
30        60  '2020/12/30'    102       129     380.3
31        60  '2020/12/31'     92       115     243.0
```

☒ لاحظ قمنا بعمل حلقة تكرارية loop على جميع القيم في عمود "Duration". إذا كانت القيمة أعلى من ١٢٠، فسنقوم بحذف الصف في الجدول الأصلي.

رابعاً : Removing Duplicates : حذف الصفوف المكررة

- **Removing Duplicates : إزالة التكرار**

لإزالة التكرار من جدول البيانات Dataframe يجب علينا أن نبحث هل يوجد صفوف مكررة أم لا .

- **Discovering Duplicates : اكتشاف التكرار**

الصفوف المكررة Duplicate rows هي صفوف تم تسجيلها أكثر من مرة واحدة.

- function duplicated() : هذه الدالة ترجع قيم منطقية Boolean values لكل صف. حيث تقوم بإرجاع True لكل صف مكررة، وتقوم بإرجاع False للصفوف الغير مكررة.

على سبيل المثال من خلال إلقاء نظرة على مجموعة بيانات الخاصة بنا، يمكننا أن نفترض أن الصف ١١ و ١٢ هما صفان متكرران .

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

✓ مثال لاكتشاف التكرار :

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.duplicated())
```

```
0     False
1     False
2     False
3     False
4     False
5     False
6     False
7     False
8     False
9     False
10    False
11    False
12     True
13    False
14    False
15    False
16    False
17    False
18    False
19    False
20    False
21    False
22    False
23    False
24    False
25    False
26    False
27    False
28    False
29    False
30    False
31    False
dtype: bool
```

☒ الآن وبعد أن اكتشفنا الصفوف المكررة سنقوم بحذف تلك الصفوف .

➢ function drop_duplicates() : هذه الدالة تقوم بإزالة التكرارات .

✓ مثال حذف الصفوف المكررة :

```
import pandas as pd

df = pd.read_csv('data.csv')

df.drop_duplicates(inplace = True)

print(df.to_string())

#Notice that row 12 has been removed from the result
```

```
    Duration          Date  Pulse  Maxpulse  Calories
0         60  '2020/12/01'    110       130     409.1
1         60  '2020/12/02'    117       145     479.0
2         60  '2020/12/03'    103       135     340.0
3         45  '2020/12/04'    109       175     282.4
4         45  '2020/12/05'    117       148     406.0
5         60  '2020/12/06'    102       127     300.0
6         60  '2020/12/07'    110       136     374.0
8         30  '2020/12/09'    109       133     195.1
9         60  '2020/12/10'     98       124     269.0
10        60  '2020/12/11'    103       147     329.3
11        60  '2020/12/12'    100       120     250.7
13        60  '2020/12/13'    106       128     345.3
14        60  '2020/12/14'    104       132     379.3
15        60  '2020/12/15'     98       123     275.0
16        60  '2020/12/16'     98       120     215.2
17        60  '2020/12/17'    100       120     300.0
18        45  '2020/12/18'     90       112       NaN
19        60  '2020/12/19'    103       123     323.0
20        45  '2020/12/20'     97       125     243.0
21        60  '2020/12/21'    108       131     364.2
22        45           NaN    100       119     282.0
23        60  '2020/12/23'    130       101     300.0
24        45  '2020/12/24'    105       132     246.0
25        60  '2020/12/25'    102       126     334.5
26        60      20201226    100       120     250.0
27        60  '2020/12/27'     92       118     241.0
28        60  '2020/12/28'    103       132       NaN
29        60  '2020/12/29'    100       132     280.0
30        60  '2020/12/30'    102       129     380.3
31        60  '2020/12/31'     92       115     243.0
```

☒ لاحظ في المثال السابق أنه تمت إزالة الصف ١٢ من الناتج.

الدرس الثالث : العلاقة بين البيانات
(Data Correlations)

➢ **Data Correlations : ارتباطات البيانات أو العلاقة بين البيانات**

من الجوانب المهمة داخل وحدة pandas هى العثور على العلاقات وحساب العلاقة بين كل عمود في مجموعة البيانات الخاصة بك.

➢ corr() function : هذه الدالة تقوم بحساب العلاقة بين كل عمود في مجموعة البيانات الخاصة بك.

وهذه الدالة تتجاهل الأعمدة الغير رقمية "not numeric ".

✓ مثال حساب العلاقات بين الأعمدة :

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.corr())
```

```
          Duration     Pulse  Maxpulse  Calories
Duration  1.000000 -0.155408  0.009403  0.922717
Pulse    -0.155408  1.000000  0.786535  0.025121
Maxpulse  0.009403  0.786535  1.000000  0.203813
Calories  0.922717  0.025121  0.203813  1.000000
```

❖ شرح المثال السابق :

- نتيجة دالة corr() هي جدول مع الكثير من الأرقام التي تمثل مدى قوة العلاقة بين عمودين.

- تختلف الأرقام التي تعبر عن العلاقة correlation من القيمة 1- إلى القيمة 1 .
- القيمة 1 معناها أن هناك علاقة واحد إلى واحد أي علاقة مثالية بين العمودين أو ارتباط مثالي (perfect correlation) ، وبالنسبة لهذه البيانات، فإن في كل مرة ارتفعت قيمة في العمود الأول، فإن قيمة المقابلة لها في العمود الآخر سوف ترتفع أيضا.

- القيمة 0.9 تعنى أن هناك علاقة أو ارتباط جيد بين العمودين (good correlation) ، وبالنسبة لهذه البيانات، فإن في كل مرة ارتفعت قيمة في العمود الأول، فإن القيمة المقابلة لها في العمود الآخر من المحتمل أن تنخفض (علاقة موجبة positive).

- القيمة 0.9- تعنى أن هناك علاقة أو ارتباط جيد بين العمودين (good correlation) ، وبالنسبة لهذه البيانات، فإن في كل مرة ارتفعت قيمة في العمود الأول، فإن القيمة المقابلة لها في العمود الآخر قد ترتفع أيضا (علاقة سلبية negative).

- القيمة 2 تعنى أن لا علاقة أو ارتباط جيد بين العمودين (not good correlation) مما يعني أن في كل مرة ارتفعت قيمة في العمود الأول، فإن القيمة المقابلة لها في العمود الآخر ليس من الضروري أن تتغير .

➢ **ما هو العلاقة الجيدة good correlation ؟**

ذلك يعتمد على الاستخدام، ولكن يمكن التعبير عن الارتباط الجيد إذا كانت العلاقة 0.6 على الأقل أو إذا كانت العلاقة 0.6- .

على سبيل المثال نجد أن عمود "Duration" وعمود "Calories" يحصلان على قيمة عالية من الارتباط تساوي 0.922721 وهو عبارة عن ارتباط جيد للغاية، ويمكننا التنبؤ بأنك كلما تتحرك وتعمل أكثر ، كلما تزداد السعرات الحرارية التي تحرقها، وبمعنى آخر: إذا أحرقت الكثير من السعرات الحرارية، ربما كان لديك عمل طويل.

➢ **ما هو العلاقة المثالية Perfect correlation ؟**

يمكن التعبير عن الارتباط المثالي إذا كانت العلاقة تساوي 1 أو إذا كانت تساوي 1- .

على سبيل المثال نجد أن عمود "Duration" وعمود "Duration " يحصلان على أعلى قيمة من الارتباط وهى تساوي 1.000000 مما يجعل أن كل عمود دائما عبارة عن علاقة مثالية مع نفسها.

➢ **ما هو العلاقة السيئة Bad correlation ؟**

ذلك يعتمد على الاستخدام ولكن يمكن التعبير عن الارتباط الجيد إذا كانت العلاقة تساوي 1 أو إذا كانت العلاقة تساوي 1- .

على سبيل المثال نجد أن عمود "Duration" وعمود "MaxPulse" يحصلان على قيمة منخفضة جداً من الارتباط تساوي 0.009403 مما يعني أنه لا يمكننا التنبؤ بأعلى قيمة للنبض pulse بالنظر فقط إلى مدة العمل Duration ، والعكس صحيح.

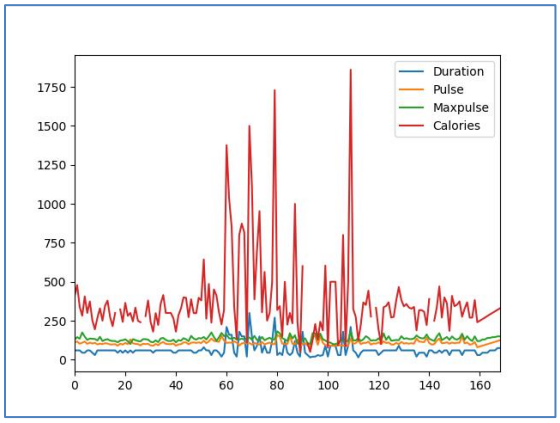
الدرس الرابع : الرسومات البيانية أو المُخططات
(Plotting)

➢ Plotting : الرسومات البيانية أو المُخططات

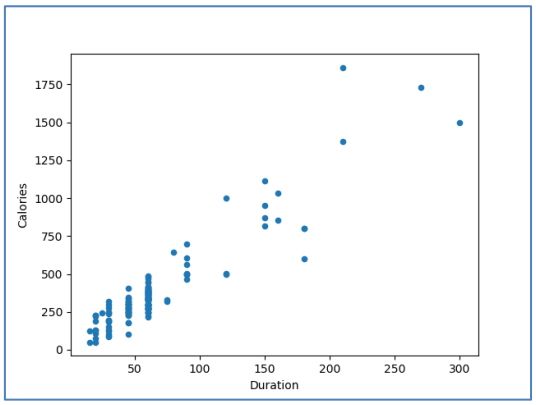
هناك مكتبة Matplotlib تحتوى على وحدة فرعية تسمى Pyplot خاصة بالرسوم البيانية diagrams والرسم التخطيطي plotting على الشاشة ، ويمكن إنشاء مخططات ورسوم بيانية عن طريقة مكتبة pandas من خلال الدالة plot() .

❖ هناك أنواع عديدة من الرسوم البيانية والمخططات plotting منها :

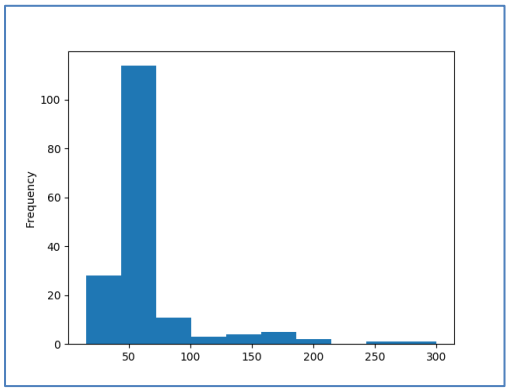
- **plotting** : المؤامرة عبارة عن تقنية رسومية لتمثيل مجموعة بيانات ، عادةً كرسم بياني يوضح العلاقة بين متغيرين أو أكثر. يمكن رسم المؤامرة باليد أو بواسطة الكمبيوتر.



- **scatter plot** : المخطط المبعثر، ويمثل الرسم البياني علاقة بين X-Y ، وأزواج المخطط المخطط مبعثر هى أزواج من البيانات العددية، مع متغير واحد على كل محور، للبحث عن علاقة بينهما. إذا كانت المتغيرات مرتبطة، فسوف تقع النقاط على طول خط أو منحنى. كلما كان الارتباط أفضل، فإن النقاط التي ستحضن الخط.



- **Histogram** : المدرج التكراري هو تمثيل بياني تشبه الرسم البياني للبيانات التي تشبه مجموعة من النتائج إلى أعمدة على طول المحور X . ويمثل المحور Y عدد الأرقام أو النسبة المئوية للحُدوث في البيانات لكل عمود ويمكن استخدامها لتصور توزيع البيانات .



➢ plot() function: هذه الدالة تقوم بعمل مخططات plotting ورسوم بيانية على الشاشة.

يتم استخدام معامل kind لتحديد نوع الرسم البياني المراد رسمه.

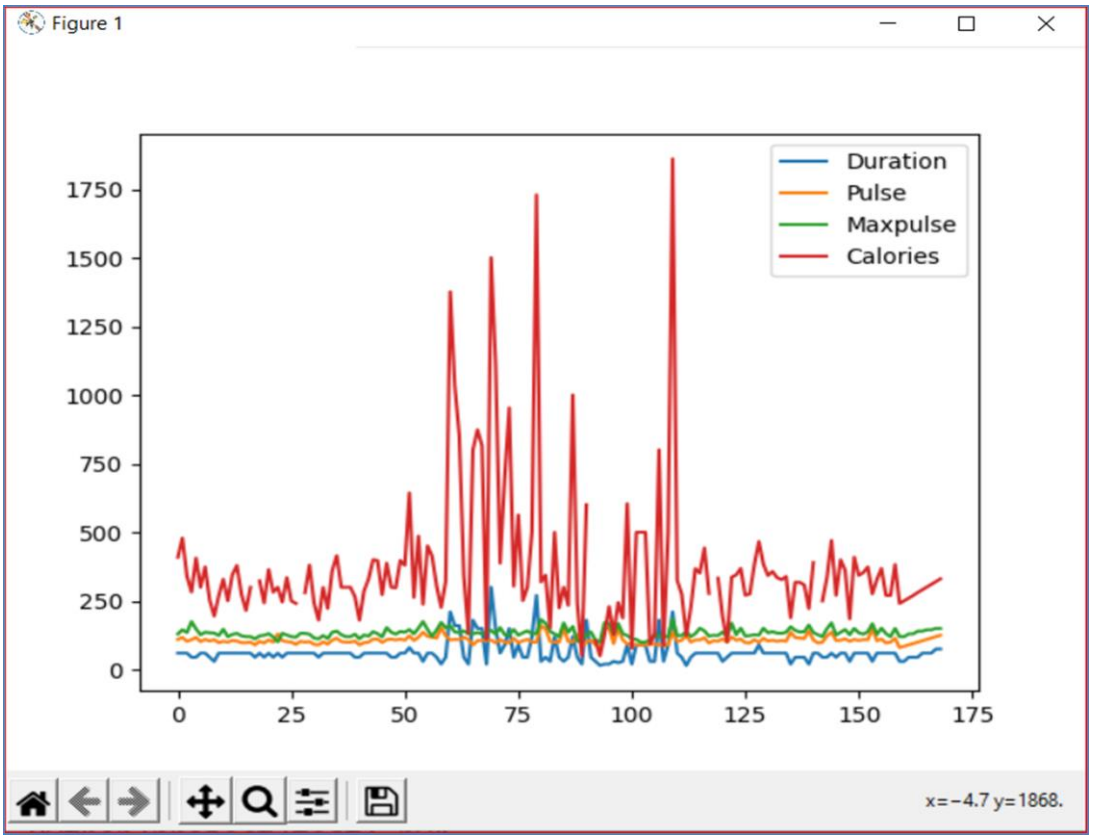
✓ مثال لرسم مخطط plotting :

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv')

df.plot()

plt.show()
```



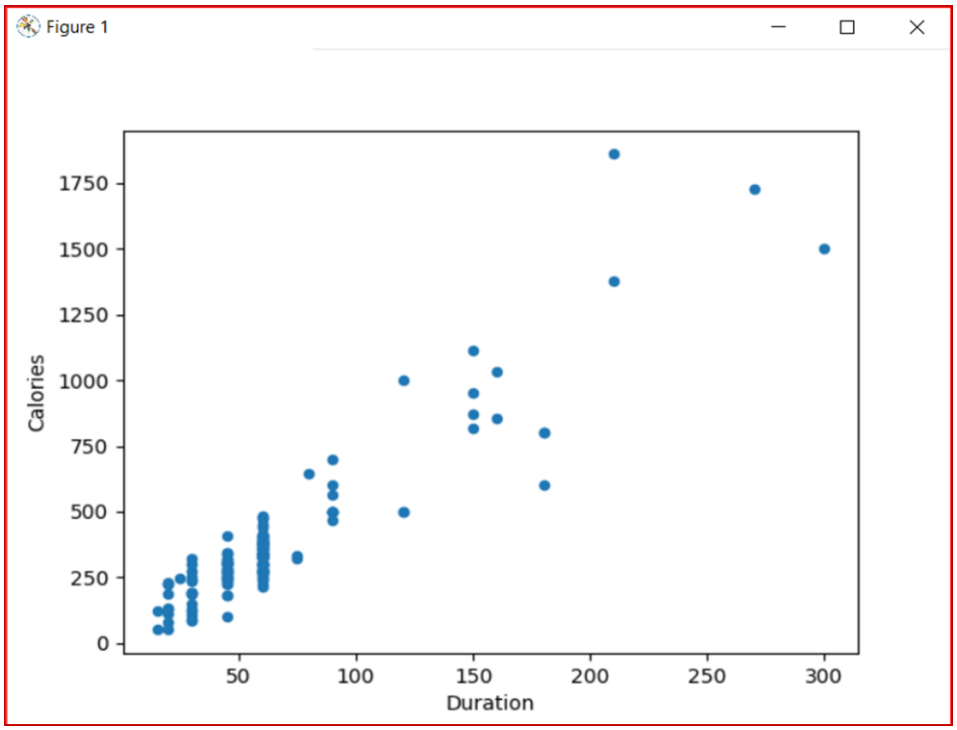
✓ مثال لرسم مخطط مبعثر scatter plot ذات علاقة جيدة بين الأعمدة :

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv')

df.plot(kind = 'scatter', x = 'Duration', y = 'Calories')

plt.show()
```



☒ لرسم **scatter plot** يجب علينا تحديد نوع المخطط المراد رسمه.

```
kind = 'scatter'
```

☒ يحتاج **scatter plot** إلى محور X ومحور Y وفي المثال السابق ، سنستخدم عمود "Duration" ليمثل المحور X وعمود "Calories" ليمثل المحور Y.

```
x = 'Duration', y = 'Calories'
```

☒ تذكر سابقاً رأينا أن الارتباط بين عمود "Duration" وعمود "Calories" كان 0.922721 ، وقد توصلنا إلى أنه كلما كانت المدة أعلى كلما ازدادت السعرات الحرارية المحروقة ومن خلال النظر إلى المخطط المبعثر scatter plot سوف يظهر صحة اتفاقنا على العلاقة القوية بين العمودين.

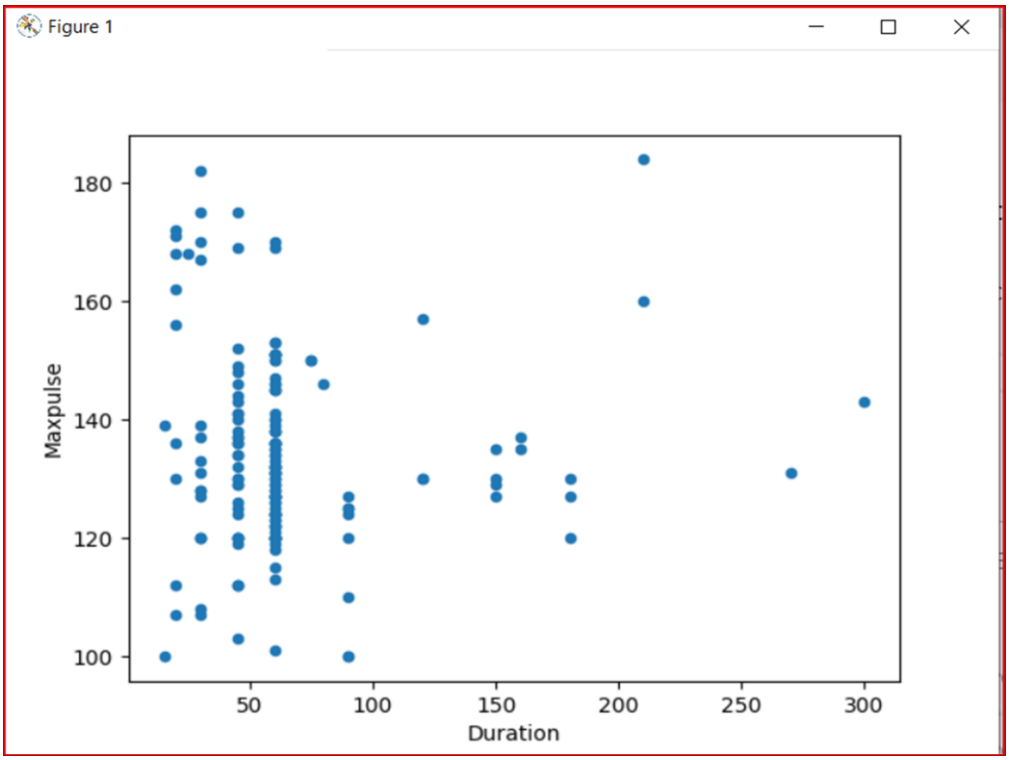
✓ مثال لرسم مخطط مبعثر scatter plot ذات علاقة سيئة بين الأعمدة :

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv')

df.plot(kind = 'scatter', x = 'Duration', y = 'Maxpulse')

plt.show()
```



☒ لرسم **scatter plot** يجب علينا تحديد نوع المخطط المراد رسمه.

`kind = 'scatter'`

☒ يحتاج **scatter plot** إلى محور X ومحور Y وفي المثال السابق ، سنستخدم عمود "Duration" ليمثل المحور X وعمود "Maxpluse" ليمثل المحور Y.

`x = 'Duration', y = 'Maxpluse'`

☒ تذكر سابقاً رأينا أن الارتباط بين عمود "Maxpluse" وعمود "Calories" كان 0.009403 ، وقد توصلنا إلى أنه ليس هناك أي علاقة بين السعرات الحرارية Calories التي تحرقها وبين أعلى قيمة للنبض Maxpluse ومن خلال النظر إلى المخطط المبعثر scatter plot سوف يظهر صحة اتفاقنا على عدوم وجود علاقة بين العمودين.

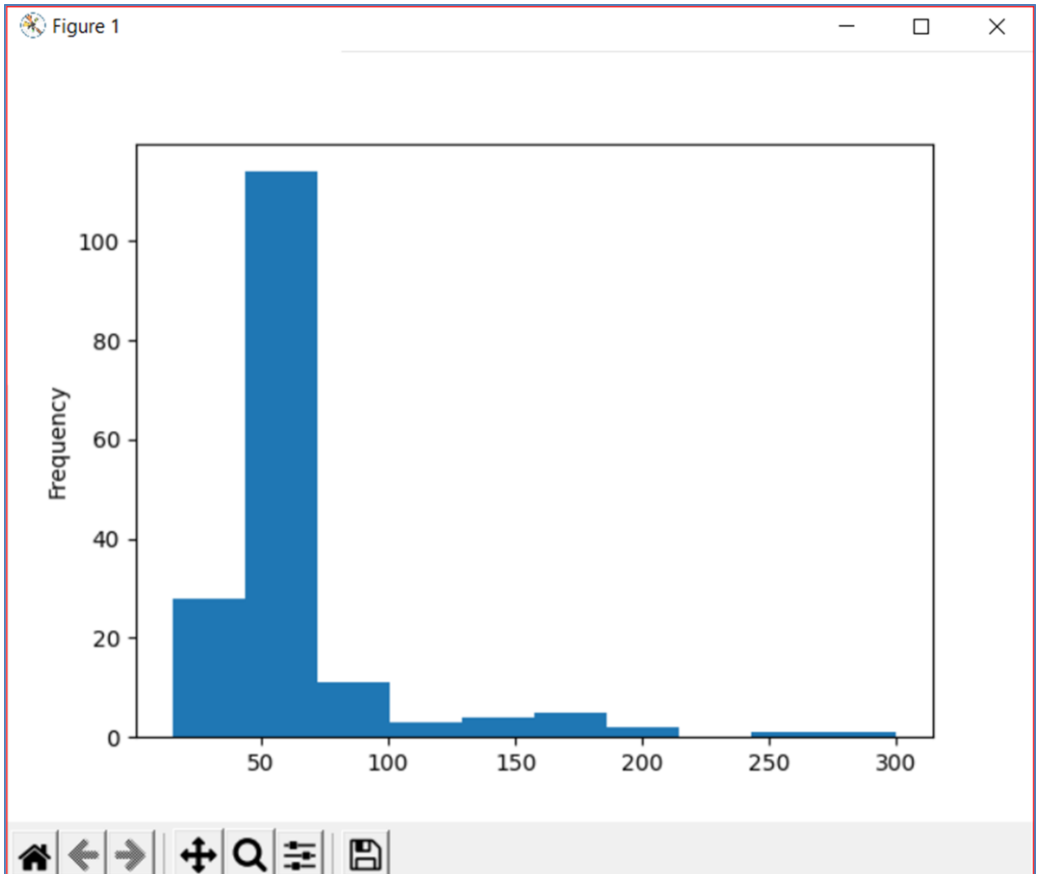
✓ مثال لرسم المخطط التكراري Histogram :

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv')

df["Duration"].plot(kind = 'hist')

plt.show()
```



☒ لرسم **scatter plot** يجب علينا تحديد نوع المخطط المراد رسمه.

```
kind = 'hist'
```

☒ يحتاج **Histogram** إلى عمود واحد فقط وفي المثال السابق ، سنستخدم عمود "Duration" ليظهر لنا عدد مرات التكرار frequency لكل فترة interval فعلى سبيل المثال كم عدد التدريبات 'workouts' التي استمرت ما بين ما بين الفترة من ٥٠ إلى ٦٠ دقيقة ؟ .

```
df["Duration"].hist()
```

☒ يخبرنا الرسم البياني أن هناك أكثر من ١٠٠ تمرين 'workouts' استمرت ما بين ٥٠ و ٦٠ دقيقة.

الخاتمة

إلى هنا نكون قد انتهينا من شرح أساسيات مكتبة pandas ، وفي الدروس القادمة سوف نبدأ سلسلة جديدة بعنوان **Way to Matplotlib** وهى مكتبة تسمح لنا بالرسوم البيانية diagrams ورسم المخططات plotting .

من باب الأمانة العلمية فهذا الكتاب هو حصيلة وتجميعة معلومات من عدة مواقع كثيرة مثل موقع www.w3schools.com وبعض الواقع الآخرى التي ربما نسيتها سهواً .

هذا الكتاب هو صدقة جارية عن نفسي وأهل بيتي وأحبابي ، وأسأل الله أن يتقبل هذا العمل ويجلعه خالصاً لوجهه الكريم . وكما وضعته بين أيدكم لتنالوا منه علماً ولو كان يسيراً ، فيمكنكم أيضاً نشره على أي موقع وتعديل محتواه بما يخدم الناس والاقتباس منه أيضاً كما تشاءون بدون استئذان.

ومن يرغب في نسخة Word من هذا الكتاب فيمكنه أن يراسلنى على الإيميلات .

رحم الله رجلاً أهدى لي عيوبي ، أرجو من كل شخص يقرأ هذا الكتاب إذا وقف عند خطأ علمى أو خطأ إملائي أو معلومة قد يساء فهمها أن يراسلني على الإيميلات الآتية لكى أقوم بإصلاحه في الطبعات الآخرى ، كما أرجو ألا تنسونا من صالح دعواتكم .

E-mail: mmasoliman19962020@gmail.com
E-FaceBook: Mahmoud Soliman
Youtube: https://www.youtube.com/channel/UCfZvWjnYnfxI55v4LNhBekA

لتحميل جميع الكتب الخاصة بنا يمكن متابعة قناتي على التيليجرام من خلال هذا الرابط :
Telegram Channel: https://t.me/joinchat/-aC4Ps0pna5lY2M0

لتحميل جميع الكتب الخاصة بنا يمكن متابعة صفحتي على الفيس بوك من خلال هذا الرابط :
Public Page: Help-me-page

Mahmoud Soliman