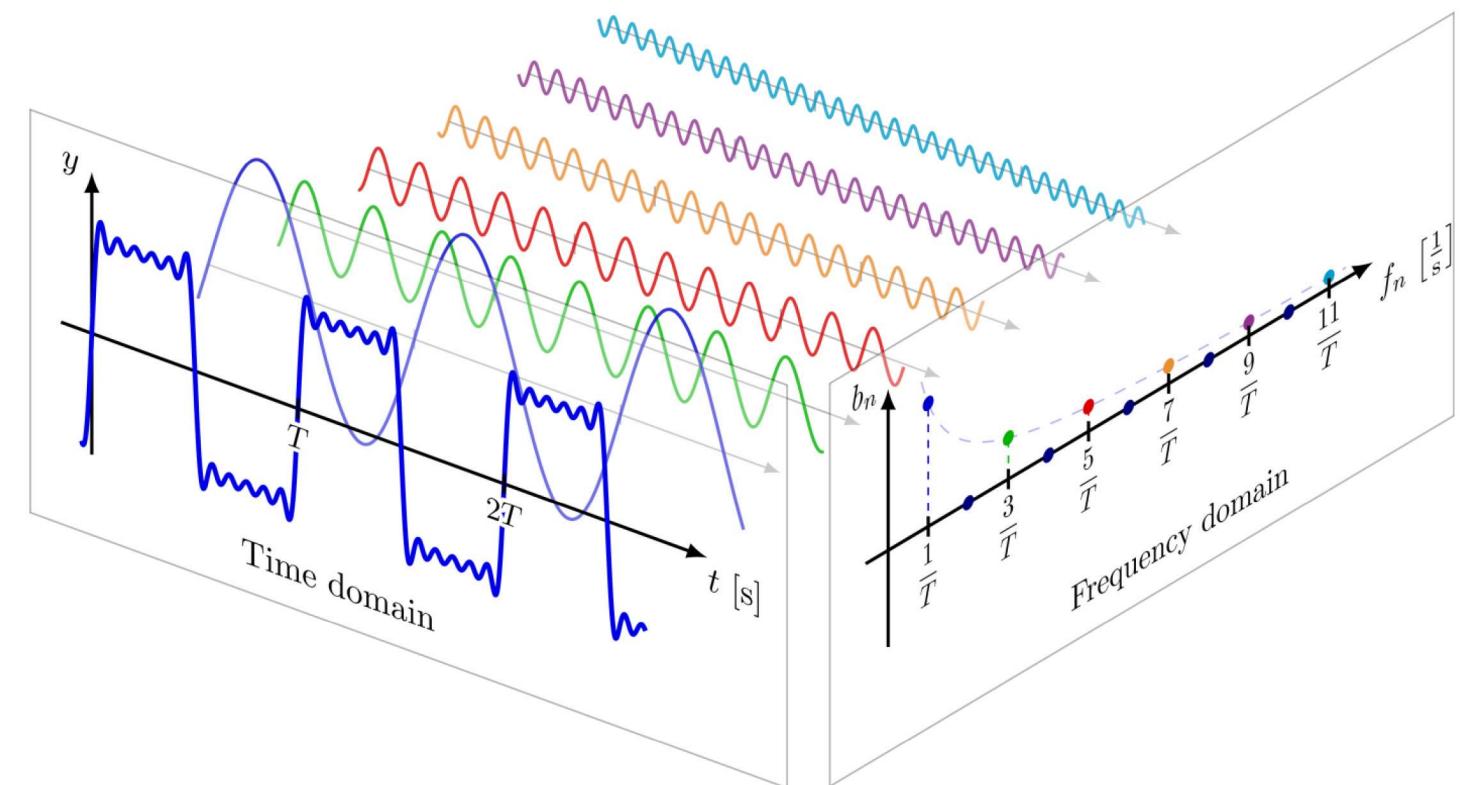


CURS 4 PD

Introducere în limbajul Python

Librăria Pandas



Introducere

Serii de date

DataFrame

1. Introducere

2. Serii de date

3. DataFrame

Introducere

DataFrame

Definiție

Pandas (prescurtare de la **Python Data Analysis**) este o librărie **Python** utilizată pentru manipularea și analiza datelor, ea oferind structuri de date și funcții pentru manipularea tabelelor și a serilor de date, oferind posibilitatea de a analiza, explora, curăța și manipula cantități mari de date.

- **Pandas** are la bază **NumPy**, de la care adoptă mare parte a sintaxei.
- Spre deosebire de **NumPy** care este construit astfel încât să lucreze cu matrice cu valori numerice de același tip (deci matrice omogenă), **Pandas** poate lucra cu date eterogene, adică în aceeași structură de date pot fi tipuri de date diferite
- Librăria **Pandas** necesită instalare (așa cum s-a văzut la primul curs).
- Pentru a o putea utiliza, va trebui importată, iar pentru a economisi timp, putem să utilizăm un alias. În mod uzual, pentru librăria **Pandas** se folosește alias-ul **pd**.
- **Pandas** introduce două noi structuri de date:

Structură de date	Descriere
Series	tablou unidimensional, etichetat, ce poate conține date de orice tip
DataFrame	tablou bidimensional (tabel), cu ambele axe (rânduri și coloane) etichetate.

Importare cu alias

```
import pandas as pd
```

1. Introducere

2. Serii de date

3. DataFrame

Introducere

DataFrame

1. Introducere

2. Serii de date

3. DataFrame

Introducere

Serii de date

DataFrame

Definiție

Seriile de date (Series) sunt tablouri unidimensionale etichetate. Acestea pot conține orice tip de date, cu condiția ca toate valorile din serie să fie de același tip.

- Inițializarea seriilor se poate face utilizând funcția `Series` care poate primi unul sau mai mulți parametrii:
 - data** – obiect iterabil (lista, dicționar, matrice NumPy, etc) ce conține valorile propriu zise ale seriei de date; dacă este de tip dicționar, cheile vor reprezenta etichetele valorilor.
 - index** – vector de aceeași dimensiune cu data conținând etichetele; dacă lipsește, atunci seria de date va avea ca etichete valori de la 0 la numărul de elemente.
 - dtype** – tipul de date al seriei specificat în același mod ca la matricile NumPy; dacă lipsește, tipul de date va fi moștenit de la data.

EXEMPLU**inițializare utilizând o listă, fără a specifica etichetele**

main.py	rezultat
<pre>import pandas as pd\n\nA = pd.Series([7, 5, 3])\nprint(A)</pre>	<pre>0 7\n1 5\n2 3\ndtype: int64</pre>

Definiție

Seriile de date (Series) sunt tablouri unidimensionale etichetate. Acestea pot conține orice tip de date, cu condiția ca toate valorile din serie să fie de același tip.

- Inițializarea seriilor se poate face utilizând funcția `Series` care poate primi unul sau mai mulți parametrii:
 - data** – obiect iterabil (lista, dicționar, matrice NumPy, etc) ce conține valorile propriu zise ale seriei de date; dacă este de tip dicționar, cheile vor reprezenta etichetele valorilor.
 - index** – vector de aceeași dimensiune cu data conținând etichetele; dacă lipsește, atunci seria de date va avea ca etichete valori de la 0 la numărul de elemente.
 - dtype** – tipul de date al seriei specificat în același mod ca la matricile NumPy; dacă lipsește, tipul de date va fi moștenit de la data.

EXEMPLU**inițializare utilizând o listă, cu etichete specificate**

main.py	rezultat
<pre>import pandas as pd A = pd.Series([7, 5, 3] , ["a", "b", "c"]) print(A)</pre>	<pre>a 7 b 5 c 3 dtype: int64</pre>

Definiție

Seriile de date (Series) sunt tablouri unidimensionale etichetate. Acestea pot conține orice tip de date, cu condiția ca toate valorile din serie să fie de același tip.

- Inițializarea seriilor se poate face utilizând funcția `Series` care poate primi unul sau mai mulți parametrii:
 - data** – obiect iterabil (lista, dicționar, matrice NumPy, etc) ce conține valorile propriu zise ale seriei de date; dacă este de tip dicționar, cheile vor reprezenta etichetele valorilor.
 - index** – vector de aceeași dimensiune cu data conținând etichetele; dacă lipsește, atunci seria de date va avea ca etichete valori de la 0 la numărul de elemente.
 - dtype** – tipul de date al seriei specificat în același mod ca la matricile NumPy; dacă lipsește, tipul de date va fi moștenit de la data.

EXEMPLU**Inițializare utilizând un dicționar, fără a specifica etichetele**

main.py	rezultat
<pre>import pandas as pd d = {"a": 7, "b": 8, "c": 9} A = pd.Series(d) print(A)</pre>	a 7 b 8 c 9 dtype: int64

Definiție

Seriile de date (Series) sunt tablouri unidimensionale etichetate. Acestea pot conține orice tip de date, cu condiția ca toate valorile din serie să fie de același tip.

- Inițializarea seriilor se poate face utilizând funcția `Series` care poate primi unul sau mai mulți parametrii:
 - data** – obiect iterabil (lista, dicționar, matrice NumPy, etc) ce conține valorile propriu zise ale seriei de date; dacă este de tip dicționar, cheile vor reprezenta etichetele valorilor.
 - index** – vector de aceeași dimensiune cu data conținând etichetele; dacă lipsește, atunci seria de date va avea ca etichete valori de la 0 la numărul de elemente.
 - dtype** – tipul de date al seriei specificat în același mod ca la matricile NumPy; dacă lipsește, tipul de date va fi moștenit de la data.

EXEMPLU**Inițializare utilizând un dicționar, cu etichete specificate**

main.py	rezultat
<pre>import pandas as pd d = {"a": 7, "b": 8, "c": 9} A = pd.Series(d, ["c", "d", "e", "a", "b"]) print(A)</pre>	<pre>c 9.0 d NaN e NaN a 7.0 b 8.0 dtype: float64</pre>

- Indexarea elementelor seriilor de date se face în mod asemănător indexării dicționarelor (adică specificând etichetele între paranteze pătrate)

EXEMPLU	
main.py	rezultat
<pre>import pandas as pd note = pd.Series({"Ion": 7, "Vasile": 8, "George": 9}) note["Ion"] = 2 print(note["Vasile"])</pre>	8

- Seriile de date acționează foarte asemănător cu matricele unidimensionale **NumPy**, un obiect de tip **Series** fiind un argument valid pentru marea majoritate a funcțiilor definite de **NumPy**.
- Seriile de date pot fi convertite în matrice unidimensionale NumPy utilizând metoda `to_numpy()`

EXEMPLU**main.py**

```
import pandas as pd
import numpy as np

date1 = pd.Series([1, 2, 3, 4, 5, 6])
date2 = np.sqrt(date1)

print(date2)
print(date2.to_numpy())
```

rezultat

```
0    1.000000
1    1.414214
2    1.732051
3    2.000000
4    2.236068
5    2.449490
dtype: float64
[1.        1.41421356 1.73205081 2.        2.23606798 2.44948974]
```

- Între două serii de date se pot aplica oricare din operațiile aritmetice de bază (adunare, scădere, înmulțire, împărțire, ridicare la putere), caz în care operațiile se vor realiza element cu element, cu mențiunea că cele două serii vor fi aliniate după etichete

EXEMPLU**main.py**

```
import pandas as pd

note_examen = pd.Series({"Ion":7, "Vasile":8, "George":9, "Alin":2})
note_laborator = pd.Series({"Ion": 2, "George":10, "Vasile":6})

media = (note_examen + note_laborator) / 2

print(media)
```

rezultat

Alin	NaN
George	9.5
Ion	4.5
Vasile	7.0

- **Pandas** definește, pentru obiectele de tip **Series**, o serie de metode pentru manipularea și analiza datelor:

Metode pentru manipularea seriilor de date	
X.count()	returnează numărul total de elemente diferite de NaN/null din serie
X.drop(l)	returnează o copie a seriei cu elementele ale căror etichete sunt specificate de lista l șterse
X.drop_duplicates()	returnează o copie a seriei cu elementele cu valori duplicate eliminate
X.pop(i)	șterge elementul cu eticheta i din serie
X.astype(d)	convertește seria la tipul de date specificat de d
Metode pentru analiza statistică a datelor	
X.min()	returnează valoarea minimă dintr-o serie
X.max()	returnează valoarea maximă dintr-o serie
X.mean()	returnează valoarea media dintr-o serie
X.median()	returnează mediana unei serii
X.mode()	returnează modul unei serii
X.std()	returnează deviația standard a unei serii
X.describe()	generează o statistică descriptivă a datelor dintr-o serie

1. Introducere

2. Serii de date

3. DataFrame

Introducere

Serii de date

DataFrame

1. Introducere

2. Serii de date

3. DataFrame

Introducere

DataFrame

Definiție

DataFrame-ul este o structură de date bidimensională, cu ambele axe (rânduri și coloane) etichetate. Practic, obiectele de tip DataFrame pot fi văzute ca serii de serii. Acestea pot conține orice tip de date, cu condiția ca toate valorile de pe o coloană să fie de același tip.

- Inițializarea **DataFrame-urilor** se poate face utilizând funcția DataFrame care poate primi unul sau mai mulți parametrii:
 - data** – dicționar de iterabile (liste, serii de date etc); cheile dicționarului vor reprezenta etichetele coloanelor, în timp ce valorile vor reprezenta valorile obiectului pe coloana respectivă.
 - index** – vector conținând etichetele rândurilor. Dacă lipsește, atunci liniile vor avea ca etichete valori de la 0 la numărul de elemente.

EXEMPLU																									
main.py	rezultat																								
<pre>import pandas as pd df = pd.DataFrame({ "Nume": ["Ion", "Vasile", "Gheorghe"], "Nota Examen": [10, 4, 7], "Nota Laborator": [9, 8, 6] }) print(df)</pre>	<table> <thead> <tr> <th></th> <th>Nume</th> <th>Nota Examen</th> <th>Nota Laborator</th> <th></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Ion</td> <td>10</td> <td>9</td> <td></td> </tr> <tr> <td>1</td> <td>Vasile</td> <td>4</td> <td>8</td> <td></td> </tr> <tr> <td>2</td> <td>Gheorghe</td> <td>7</td> <td>6</td> <td></td> </tr> </tbody> </table>		Nume	Nota Examen	Nota Laborator		0	Ion	10	9		1	Vasile	4	8		2	Gheorghe	7	6					
	Nume	Nota Examen	Nota Laborator																						
0	Ion	10	9																						
1	Vasile	4	8																						
2	Gheorghe	7	6																						

- Pentru că **Pandas** este dedicat manipulării și analizei datelor de mari dimensiuni, cea mai comună metodă de inițializare a unui **DataFrame** este prin importarea dintr-o sursă externă (de obicei fișier **CSV**)

EXEMPLU

Importarea unui fișier CSV care conține informații despre emisiunile și filmele Netflix (<https://www.kaggle.com/shivamb/netflix-shows>)

main.py	rezultat																																															
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') print(df)</pre>	<p style="text-align: right;">description</p> <table> <thead> <tr> <th>show_id</th> <th>...</th> <th>...</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>s1</td> <td>...</td> <td>As her father nears the end of his life, filmm...</td> </tr> <tr> <td>1</td> <td>s2</td> <td>...</td> <td>After crossing paths at a party, a Cape Town t...</td> </tr> <tr> <td>2</td> <td>s3</td> <td>...</td> <td>To protect his family from a powerful drug lor...</td> </tr> <tr> <td>3</td> <td>s4</td> <td>...</td> <td>Feuds, flirtations and toilet talk go down amo...</td> </tr> <tr> <td>4</td> <td>s5</td> <td>...</td> <td>In a city of coaching centers known to train I...</td> </tr> <tr> <td>...</td> <td>...</td> <td>...</td> <td>...</td> </tr> <tr> <td>8802</td> <td>s8803</td> <td>...</td> <td>A political cartoonist, a crime reporter and a...</td> </tr> <tr> <td>8803</td> <td>s8804</td> <td>...</td> <td>While living alone in a spooky town, a young g...</td> </tr> <tr> <td>8804</td> <td>s8805</td> <td>...</td> <td>Looking to survive in a world taken over by zo...</td> </tr> <tr> <td>8805</td> <td>s8806</td> <td>...</td> <td>Dragged from civilian life, a former superhero...</td> </tr> <tr> <td>8806</td> <td>s8807</td> <td>...</td> <td>A scrappy but poor boy worms his way into a ty...</td> </tr> </tbody> </table> <p>[8807 rows x 12 columns]</p>	show_id	0	s1	...	As her father nears the end of his life, filmm...	1	s2	...	After crossing paths at a party, a Cape Town t...	2	s3	...	To protect his family from a powerful drug lor...	3	s4	...	Feuds, flirtations and toilet talk go down amo...	4	s5	...	In a city of coaching centers known to train I...	8802	s8803	...	A political cartoonist, a crime reporter and a...	8803	s8804	...	While living alone in a spooky town, a young g...	8804	s8805	...	Looking to survive in a world taken over by zo...	8805	s8806	...	Dragged from civilian life, a former superhero...	8806	s8807	...	A scrappy but poor boy worms his way into a ty...
show_id																																														
0	s1	...	As her father nears the end of his life, filmm...																																													
1	s2	...	After crossing paths at a party, a Cape Town t...																																													
2	s3	...	To protect his family from a powerful drug lor...																																													
3	s4	...	Feuds, flirtations and toilet talk go down amo...																																													
4	s5	...	In a city of coaching centers known to train I...																																													
...																																													
8802	s8803	...	A political cartoonist, a crime reporter and a...																																													
8803	s8804	...	While living alone in a spooky town, a young g...																																													
8804	s8805	...	Looking to survive in a world taken over by zo...																																													
8805	s8806	...	Dragged from civilian life, a former superhero...																																													
8806	s8807	...	A scrappy but poor boy worms his way into a ty...																																													

- Se poate observa că tabelul are 8807 rânduri și 12 coloane, dar afișarea se face trunchiat.
- Implicit, prima linie din fișierul CSV acționează ca și etichete pentru coloane
- Rândurile au fost etichetate cu o secvență de numere de la 0 la 8806

- Se poate observa că tabelul are 8807 rânduri și 12 coloane, dar afișarea se face trunchiat.
- Implicit, prima linie din fișierul CSV acționează ca și etichete pentru coloane
- Rândurile au fost etichetate cu o secvență de numere de la 0 la 8806
- Pentru că afișarea se face trunchiat, se poate utiliza metoda `info()`, pentru a vedea ce coloane sunt disponibile și ce tip de date conțin ele

EXEMPLU	
main.py	rezultat
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') df.info()</pre>	<pre><class 'pandas.core.frame.DataFrame'> RangeIndex: 8807 entries, 0 to 8806 Data columns (total 12 columns): # Column Non-Null Count Dtype --- -- 0 show_id 8807 non-null object 1 type 8807 non-null object 2 title 8807 non-null object 3 director 6173 non-null object 4 cast 7982 non-null object 5 country 7976 non-null object 6 date_added 8797 non-null object 7 release_year 8807 non-null int64 8 rating 8803 non-null object 9 duration 8804 non-null object 10 listed_in 8807 non-null object 11 description 8807 non-null object dtypes: int64(1), object(11) memory usage: 825.8+ KB</pre>

- Implicit, la importarea unui fișier CSV, etichetele rândurilor sunt generate ca o secvență de numere de la 0 la numărul total de rânduri.
- Putem seta una din coloanele tabelului ca reprezentând etichetele rândurilor utilizând metoda `set_index`.

EXEMPLU																																																																		
main.py	rezultat																																																																	
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') df = df.set_index('show_id') print(df)</pre>	<p style="text-align: center;">EXEMPLU</p> <table border="1"> <thead> <tr> <th></th> <th></th> <th>type</th> <th>...</th> <th>description</th> </tr> </thead> <tbody> <tr> <td></td> <td>show_id</td> <td></td> <td>...</td> <td></td> </tr> <tr> <td>s1</td> <td>Movie</td> <td>...</td> <td>As her father nears the end of his life, filmm...</td> <td></td> </tr> <tr> <td>s2</td> <td>TV Show</td> <td>...</td> <td>After crossing paths at a party, a Cape Town t...</td> <td></td> </tr> <tr> <td>s3</td> <td>TV Show</td> <td>...</td> <td>To protect his family from a powerful drug lor...</td> <td></td> </tr> <tr> <td>s4</td> <td>TV Show</td> <td>...</td> <td>Feuds, flirtations and toilet talk go down amo...</td> <td></td> </tr> <tr> <td>s5</td> <td>TV Show</td> <td>...</td> <td>In a city of coaching centers known to train I...</td> <td></td> </tr> <tr> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> </tr> <tr> <td>s8803</td> <td>Movie</td> <td>...</td> <td>A political cartoonist, a crime reporter and a...</td> <td></td> </tr> <tr> <td>s8804</td> <td>TV Show</td> <td>...</td> <td>While living alone in a spooky town, a young g...</td> <td></td> </tr> <tr> <td>s8805</td> <td>Movie</td> <td>...</td> <td>Looking to survive in a world taken over by zo...</td> <td></td> </tr> <tr> <td>s8806</td> <td>Movie</td> <td>...</td> <td>Dragged from civilian life, a former superhero...</td> <td></td> </tr> <tr> <td>s8807</td> <td>Movie</td> <td>...</td> <td>A scrappy but poor boy worms his way into a ty...</td> <td></td> </tr> </tbody> </table> <p>[8807 rows x 11 columns]</p>			type	...	description		show_id		...		s1	Movie	...	As her father nears the end of his life, filmm...		s2	TV Show	...	After crossing paths at a party, a Cape Town t...		s3	TV Show	...	To protect his family from a powerful drug lor...		s4	TV Show	...	Feuds, flirtations and toilet talk go down amo...		s5	TV Show	...	In a city of coaching centers known to train I...		s8803	Movie	...	A political cartoonist, a crime reporter and a...		s8804	TV Show	...	While living alone in a spooky town, a young g...		s8805	Movie	...	Looking to survive in a world taken over by zo...		s8806	Movie	...	Dragged from civilian life, a former superhero...		s8807	Movie	...	A scrappy but poor boy worms his way into a ty...	
		type	...	description																																																														
	show_id		...																																																															
s1	Movie	...	As her father nears the end of his life, filmm...																																																															
s2	TV Show	...	After crossing paths at a party, a Cape Town t...																																																															
s3	TV Show	...	To protect his family from a powerful drug lor...																																																															
s4	TV Show	...	Feuds, flirtations and toilet talk go down amo...																																																															
s5	TV Show	...	In a city of coaching centers known to train I...																																																															
...																																																														
s8803	Movie	...	A political cartoonist, a crime reporter and a...																																																															
s8804	TV Show	...	While living alone in a spooky town, a young g...																																																															
s8805	Movie	...	Looking to survive in a world taken over by zo...																																																															
s8806	Movie	...	Dragged from civilian life, a former superhero...																																																															
s8807	Movie	...	A scrappy but poor boy worms his way into a ty...																																																															

- Dacă în urma importării unui fișier CSV, tipul de date aferent fiecărei coloane nu a fost stabilit în mod corect, utilizând metoda `astype` căreia îi dăm ca parametru un dicționar ale cărui perechi cheie – valoare reprezintă perechi coloană – tip data pe care să setăm tipurile de date corecte.
- Pe lângă tipurile clasice de date, se pot seta și alte tipuri precum `datetime64[ns]` (care să țină minte o dată) sau `category` (pentru coloanele care specifică categorii).

EXEMPLU	
main.py	rezultat
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') df.set_index("show_id", inplace=True) df = df.astype({ "type": "category", "release_year": "i4", "date_added": "datetime64[ns]" }) df.info()</pre>	<pre><class 'pandas.core.frame.DataFrame'> Index: 8807 entries, s1 to s8807 Data columns (total 11 columns): # Column Non-Null Count Dtype --- 0 type 8807 non-null category 1 title 8807 non-null object 2 director 6173 non-null object 3 cast 7982 non-null object 4 country 7976 non-null object 5 date_added 8797 non-null datetime64[ns] 6 release_year 8807 non-null int32 7 rating 8803 non-null object 8 duration 8804 non-null object 9 listed_in 8807 non-null object 10 description 8807 non-null object dtypes: category(1), datetime64[ns](1), int32(1), object(8) memory usage: 731.2+ KB</pre>

- Selectarea unei singure coloane se poate face scriind, după numele variabilei de tip **DataFrame**, între paranteze pătrate eticheta coloanei respective, rezultând o serie ce conține doar datele de pe acea coloană.
- Dacă între parantezele pătrate se dă o listă de etichete de coloane, va rezulta tot un **DataFrame** conținând doar coloanele specificate.

EXEMPLU

main.py	rezultat
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') titluri = df['title'] print(titluri)</pre>	<pre>0 Dick Johnson Is Dead 1 Blood & Water 2 Ganglands 3 Jailbirds New Orleans 4 Kota Factory ... 8802 Zodiac 8803 Zombie Dumb 8804 Zombieland 8805 Zoom 8806 Zubaan Name: title, Length: 8807, dtype: object</pre>

- Selectarea unei singure coloane se poate face scriind, după numele variabilei de tip **DataFrame**, între paranteze pătrate eticheta coloanei respective, rezultând o serie ce conține doar datele de pe acea coloană.
- Dacă între parantezele pătrate se dă o listă de etichete de coloane, va rezulta tot un **DataFrame** conținând doar coloanele specificate.
- Există mai multe modalități de a extrage elemente, rânduri, coloane sau subseturi, dar cele mai eficiente dintre ele includ utilizarea proprietăților loc (indexare pe baza etichetelor rândurilor și coloanelor) și **iloc** (indexare pe baza poziției rândurilor și coloanelor)

EXEMPLU

main.py	rezultat															
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') df = df.set_index('show_id') print(df.iloc[5, 1]) print(df.iloc[5:7, 1:3]) print(df.loc['s6', 'director'])</pre>	<p>Midnight Mass</p> <table> <thead> <tr> <th></th> <th>title</th> <th>director</th> </tr> </thead> <tbody> <tr> <td>show_id</td> <td></td> <td></td> </tr> <tr> <td>s6</td> <td>Midnight Mass</td> <td>Mike Flanagan</td> </tr> <tr> <td>s7</td> <td>My Little Pony: A New Generation</td> <td>Robert Cullen, José Luis Ucha</td> </tr> <tr> <td></td> <td>Mike Flanagan</td> <td></td> </tr> </tbody> </table>		title	director	show_id			s6	Midnight Mass	Mike Flanagan	s7	My Little Pony: A New Generation	Robert Cullen, José Luis Ucha		Mike Flanagan	
	title	director														
show_id																
s6	Midnight Mass	Mike Flanagan														
s7	My Little Pony: A New Generation	Robert Cullen, José Luis Ucha														
	Mike Flanagan															

- Selectarea unei singure coloane se poate face scriind, după numele variabilei de tip **DataFrame**, între paranteze pătrate eticheta coloanei respective, rezultând o serie ce conține doar datele de pe acea coloană.
- Dacă între parantezele pătrate se dă o listă de etichete de coloane, va rezulta tot un **DataFrame** conținând doar coloanele specificate.
- Există mai multe modalități de a extrage elemente, rânduri, coloane sau subseturi, dar cele mai eficiente dintre ele includ utilizarea proprietăților loc (indexare pe baza etichetelor rândurilor și coloanelor) și **iloc** (indexare pe baza poziției rândurilor și coloanelor)
- Dacă la utilizarea acestor proprietăți se folosește doar un index / o etichetă, atunci el va reprezenta rânduri, deci astfel putem extrage un rând dintr-un **DataFrame**

EXEMPLU

main.py	rezultat
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') print(df.iloc[0])</pre>	show_id s1 type Movie title Dick Johnson Is Dead director Kirsten Johnson cast NaN country United States date_added September 25, 2021 release_year 2020 rating PG-13 duration 90 min listed_in Documentaries description As her father nears the end of his life, filmm... Name: 0, dtype: object

- O sarcină tipică în procesarea datelor este filtrarea acestora, adică extragerea unor anumitori înregistrări.
- Pandas** oferă mai multe metode de a face acest lucru, una din ele fiind utilizarea metodei `query` care primește ca parametru un sir de caractere ce specifică expresia ce descrie filtrarea dorită.

EXEMPLU

se dorește selectarea doar a filmelor/emisiunilor lansate după anul 2020

main.py	rezultat																		
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') filme_noi = df.query("release_year > 2020") print(filme_noi["title"])</pre>	<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;">1</td> <td>Blood & Water</td> </tr> <tr> <td>2</td> <td>Ganglands</td> </tr> <tr> <td>3</td> <td>Jailbirds New Orleans</td> </tr> <tr> <td>4</td> <td>Kota Factory</td> </tr> <tr> <td></td> <td>...</td> </tr> <tr> <td>1696</td> <td>Polly Pocket</td> </tr> <tr> <td>2920</td> <td>Love Is Blind</td> </tr> <tr> <td>8437</td> <td>The Netflix Afterparty</td> </tr> <tr> <td></td> <td>Name: title, Length: 592, dtype: object</td> </tr> </table>	1	Blood & Water	2	Ganglands	3	Jailbirds New Orleans	4	Kota Factory		...	1696	Polly Pocket	2920	Love Is Blind	8437	The Netflix Afterparty		Name: title, Length: 592, dtype: object
1	Blood & Water																		
2	Ganglands																		
3	Jailbirds New Orleans																		
4	Kota Factory																		
	...																		
1696	Polly Pocket																		
2920	Love Is Blind																		
8437	The Netflix Afterparty																		
	Name: title, Length: 592, dtype: object																		

- Se pot scrie expresii de filtrare mult mai complexe. Exemple:

Expresie	Descriere
release_year >= 2020	filmele/emisiunile lansate începând cu anul 2020
release_year == 2020 and type == 'Movie'	filmele lansate în anul 2020
release_year in (1999, 2007)	filmele/emisiunile lansate fie în anul 1999 fie în anul 2007
listed_in.str.contains('Documentaries')	filmele/emisiunile care sunt de tip documentar

- Dacă se dorește analiza statistică a datele dintr-o coloană, se vor putea utiliza metodele prezentate anterior, specifice seriilor de date, după ce extragem coloana respectivă.
- ACEste metode se pot aplica și obiectelor de tip DataFrame, caz în care parametrul statistic dorit se va calcula pentru fiecare coloană ce conține un tip de date asupra căruia se poate aplica metoda (de exemplu maximul se poate aplica pentru date numerice dar și pentru siruri de caractere, în timp ce media sau abaterea standard poate fi calculată doar pentru date numerice).
- O metodă utilă la analiza statistică a datelor este metoda groupby care primește ca parametru o etichetă de coloană și care permite gruparea datelor și aplicarea metodelor de analiză statistică fiecărui grup

EXEMPLU

se dorește să se afle câte emisiuni/filme au fost lansate în fiecare an

main.py	rezultat																
<pre>import pandas as pd df = pd.read_csv('netflix_titles.csv') # extragem coloanele release_year și title # grupăm după coloana release_year # aplicam metoda count print(df[["release_year", "title"]].groupby("release_year").count())</pre>	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">title</th> <th style="text-align: right;">release_year</th> </tr> </thead> <tbody> <tr> <td style="text-align: right;">1925</td> <td style="text-align: right;">1</td> </tr> <tr> <td style="text-align: right;">1942</td> <td style="text-align: right;">2</td> </tr> <tr> <td style="text-align: right;">1943</td> <td style="text-align: right;">3</td> </tr> <tr> <td style="text-align: right;">...</td> <td style="text-align: right;">...</td> </tr> <tr> <td style="text-align: right;">2020</td> <td style="text-align: right;">953</td> </tr> <tr> <td style="text-align: right;">2021</td> <td style="text-align: right;">592</td> </tr> <tr> <td colspan="2" style="text-align: center;">[74 rows x 1 columns]</td></tr> </tbody> </table>	title	release_year	1925	1	1942	2	1943	3	2020	953	2021	592	[74 rows x 1 columns]	
title	release_year																
1925	1																
1942	2																
1943	3																
...	...																
2020	953																
2021	592																
[74 rows x 1 columns]																	

- Dacă se dorește analiza statistică a datele dintr-o coloană, se vor putea utiliza metodele prezentate anterior, specifice seriilor de date, după ce extragem coloana respectivă.
- ACEste metode se pot aplica și obiectelor de tip DataFrame, caz în care parametrul statistic dorit se va calcula pentru fiecare coloană ce conține un tip de date asupra căruia se poate aplica metoda (de exemplu maximul se poate aplica pentru date numerice dar și pentru siruri de caractere, în timp ce media sau abaterea standard poate fi calculată doar pentru date numerice).
- O metodă utilă la analiza statistică a datelor este metoda groupby care primește ca parametru o etichetă de coloană și care permite gruparea datelor și aplicarea metodelor de analiză statistică fiecărui grup
- O altă metodă utilă pentru analiza datelor o reprezintă metoda corr care va calcula corelația fiecărei perechi de coloane. Această metodă este utilă pentru a cunoaște gradul de similaritate/dependență între oricare două coloane

EXEMPLU

`main.py`

```
import pandas as pd
import numpy as np

df = pd.DataFrame({
    "x": np.arange(5, 10, 1),
    "y": np.sqrt(np.arange(5, 10, 1)),
    "z": [3, 2, 8, 1, 9]
})

print(df.corr())
```

`rezultat`

	x	y	z
x	1.000000	0.999067	0.491935
y	0.999067	1.000000	0.458418
z	0.491935	0.458418	1.000000

- **Pandas** pune la dispoziție o serie de metode pentru manipularea **DataFrame-urilor**. Acestea se aplică obiectelor de tip **DataFrame**, dar **NU** le modifică, ci returnează un alt **DataFrame** cu modificările dorite.

Metoda	Descriere
X.drop(labels, axis)	șterge rândurile/coloanele etichetate cu etichetele specificate de parametrul <code>labels</code> ; parametrul <code>axis</code> specifică dacă e vorba de rânduri (0) sau coloane (1)
X.dropna(axis)	șterge rândurile sau coloanele din care lipsesc date; parametrul <code>axis</code> specifică dacă e vorba de rânduri (0) sau coloane (1)
X.drop_duplicates()	șterge rândurile duplicate
X.merge(Y, on)	unește coloanele DataFrame -ului X cu coloanele DataFrame -ului Y aliniindu-le după coloana specificată de parametrul <code>on</code> ; Dacă parametrul <code>on</code> lipsește, alinierarea se va face după etichetele liniilor
X.rename(map, axis)	redenumește etichetele rândurilor/coloanelor conform dicționarului <code>map</code> . Parametrul <code>axis</code> specifică dacă e vorba de rânduri sau coloane.

EXEMPLU																																
main.py																																
<pre>df1 = pd.DataFrame({ "sensor_id": [1, 2, 4, 3, 2, 1, 1], "value": [22, 0.8, 28, 7, 0.6, 21, 20] }) df2 = pd.DataFrame({ "id_sensor": [1, 2, 3, 4], "sensor_name": ["Temp1", "Pres", "Temp2", "pH"] }) df2 = df2.rename({"id_sensor": "sensor_id"}, axis=1) print(df1.merge(df2, on="sensor_id"))</pre>																																
rezultat																																
<table border="1"> <thead> <tr> <th></th> <th>sensor_id</th> <th>value</th> <th>sensor_name</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> <td>22.0</td> <td>Temp1</td> </tr> <tr> <td>1</td> <td>1</td> <td>21.0</td> <td>Temp1</td> </tr> <tr> <td>2</td> <td>1</td> <td>20.0</td> <td>Temp1</td> </tr> <tr> <td>3</td> <td>2</td> <td>0.8</td> <td>Pres</td> </tr> <tr> <td>4</td> <td>2</td> <td>0.6</td> <td>Pres</td> </tr> <tr> <td>5</td> <td>4</td> <td>28.0</td> <td>pH</td> </tr> <tr> <td>6</td> <td>3</td> <td>7.0</td> <td>Temp2</td> </tr> </tbody> </table>		sensor_id	value	sensor_name	0	1	22.0	Temp1	1	1	21.0	Temp1	2	1	20.0	Temp1	3	2	0.8	Pres	4	2	0.6	Pres	5	4	28.0	pH	6	3	7.0	Temp2
	sensor_id	value	sensor_name																													
0	1	22.0	Temp1																													
1	1	21.0	Temp1																													
2	1	20.0	Temp1																													
3	2	0.8	Pres																													
4	2	0.6	Pres																													
5	4	28.0	pH																													
6	3	7.0	Temp2																													

1. Introducere

2. Serii de date

3. DataFrame

Introducere

DataFrame

1. Introducere

2. Serii de date

3. DataFrame

Introducere

Serii de date

DataFrame