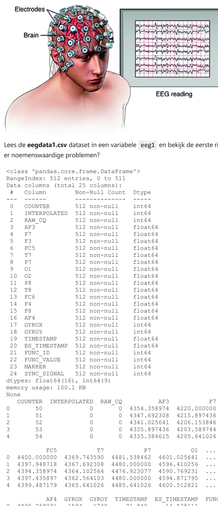
Elektro-encefalografie

Elektro-encefalografie is een methode om de **elektrische activiteit** van de hersenen te meten. Het is een non-invasieve ingreep, waarbij de elektroden doorgaans op de hoofdhuid worden geplakt. EEG meet potentiaalverschillen die ontstaan door de ionenstroom in de zenuwcellen van de hersenen. Via een speciale headset kan de hersenactiviteit in 14 punten gemeten worden.



Lees de **eegdata1.csv** dataset in een variabele `eeg1` en bekijk de eerste rijen en de datatypes van de kolommen. Zijn er noemenswaardige problemen?

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 512 entries, 0 to 511
Data columns (total 25 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   COUNTER      512 non-null    int64
 1   INTERPOLATED  512 non-null    int64
 2   RAW_CQ       512 non-null    int64
 3   AF3          512 non-null    float64
 4   F7           512 non-null    float64
 5   F3           512 non-null    float64
 6   FC5          512 non-null    float64
 7   T7           512 non-null    float64
 8   P7           512 non-null    float64
 9   O1           512 non-null    float64
 10  O2           512 non-null    float64
 11  P8           512 non-null    float64
 12  T8           512 non-null    float64
 13  FC6          512 non-null    float64
 14  F4           512 non-null    float64
 15  F8           512 non-null    float64
 16  AF4          512 non-null    float64
 17  GYROX        512 non-null    int64
 18  GYROY        512 non-null    int64
 19  TIMESTAMP    512 non-null    float64
 20  ES_TIMESTAMP  512 non-null    int64
 21  FUNC_ID      512 non-null    int64
 22  FUNC_VALUE   512 non-null    int64
 23  MARKER       512 non-null    int64
 24  SYNC_SIGNAL  512 non-null    int64
dtypes: float64(16), int64(9)
memory usage: 100.1 KB
None
   COUNTER  INTERPOLATED  RAW_CQ         AF3          F7          F3  \
0        0             0       0  4354.358974  4220.000000  4356.410256
1        1             0       0  4347.692308  4215.897436  4348.717949
2        2             0       0  4341.025641  4206.153846  4348.205128
3        3             0       0  4335.897436  4203.589744  4349.743590
4        4             0       0  4335.384615  4205.641026  4345.128205

         FC5          T7          P7          O1  ...          P8  \
0  4400.000000  4369.743590  4481.538462  4601.025641  ...  4188.717949
1  4398.461538  4367.692308  4480.000000  4596.410256  ...  4183.589744
2  4396.358974  4364.102564  4476.923077  4590.769231  ...  4173.333333
3  4397.435897  4362.564103  4480.000000  4594.871795  ...  4170.769231
4  4399.487179  4365.641026  4485.641026  4600.512821  ...  4170.256410

          AF4  GYROX  GYROY  TIMESTAMP  ES_TIMESTAMP  FUNC_ID  FUNC_VALUE  \
0  4390.769231   1594   1740    71.840     14.528111        0           0
1  4380.512821   1592   1739    71.848     14.528111        0           0
2  4372.307692   1594   1740    71.856     14.528111        0           0
3  4369.230769   1593   1737    71.864     14.528111        0           0
4  4364.102564   1593   1737    71.872     14.528111        0           0

   MARKER  SYNC_SIGNAL
0       0            0
1       0            0
2       0            0
3       0            0
4       0            0

[5 rows x 25 columns]
```

Print de kolomnamen op van de dataset en ook de index.

```
Index(['COUNTER', 'INTERPOLATED', 'RAW_CQ', 'AF3', 'F7', 'F3', 'FC5', 'T7',
       'P7', 'O1', 'O2', 'P8', 'T8', 'FC6', 'F4', 'F8', 'AF4', 'GYROX',
       'GYROY', 'TIMESTAMP', 'ES_TIMESTAMP', 'FUNC_ID', 'FUNC_VALUE', 'MARKER',
       'SYNC_SIGNAL'],
      dtype='object')
RangeIndex(start=0, stop=512, step=1)
```

Print de waarden van meetpunt AF3 uit de `eeg1` dataset.

```
0      4354.358974
1      4347.692308
2      4341.025641
3      4335.897436
4      4335.384615
          ...
507    4292.307692
508    4292.307692
509    4298.974359
510    4303.076923
511    4297.948718
Name: AF3, Length: 512, dtype: float64
```

	AF3
0	4354.358974
1	4347.692308
2	4341.025641
3	4335.897436
4	4335.384615
...	...
507	4292.307692
508	4292.307692
509	4298.974359
510	4303.076923
511	4297.948718

512 rows × 1 columns

Print nu ook het **gemiddelde, de minimum en maximum waarden** van meetpunt AF3.

```
4335.919471162109
4269.179487
4469.230769
```

Doe hetzelfde voor de meetpunten F7 en F3.

```
F7    4180.959535
F3    4322.464258
dtype: float64
F7    4143.076923
F3    4262.564103
dtype: float64
F7    4250.769231
F3    4392.820513
dtype: float64
```

Print de inhoud van de cel op positie [10, 10], print dan enkel de 4de kolom en print totslot enkel de 4de rij uit.

```
4012.307692
0      4220.000000
1      4215.897436
2      4206.153846
3      4203.589744
4      4205.641026
          ...
507    4155.897436
508    4153.333333
509    4157.435897
510    4159.487179
511    4156.923077
Name: F7, Length: 512, dtype: float64
COUNTER                 54.000000
INTERPOLATED             0.000000
RAW_CQ                   0.000000
AF3                   4335.384615
F7                    4205.641026
F3                    4345.128205
FC5                   4399.487179
T7                    4365.641026
P7                    4485.641026
O1                    4600.512821
O2                    4014.358974
P8                    4431.282051
T8                    4363.589744
FC6                   4571.794872
F4                    4202.564103
F8                    4170.256410
AF4                   4364.102564
GYROX                 1593.000000
GYROY                 1737.000000
TIMESTAMP               71.872000
ES_TIMESTAMP            14.528111
FUNC_ID                  0.000000
FUNC_VALUE               0.000000
MARKER                   0.000000
SYNC_SIGNAL              0.000000
Name: 4, dtype: float64
```

Vraag de waarde van cel [1,1] op, verander ze daarna naar 120 en controleer of de waarde veranderd is.

```
0
120
```

Print de waarden van kolommen TIMESTAMP kolom uit als een series-object.

```
0      71.840
1      71.848
2      71.856
3      71.864
4      71.872
        ...
507    75.804
508    75.812
509    75.820
510    75.827
511    75.835
Name: TIMESTAMP, Length: 512, dtype: float64
```

Print de waarden van kolommen TIMESTAMP kolom uit als een dataframe-object. Gebruik een list van strings om te indexeren i.p.v. alleen een string.

	TIMESTAMP
0	71.840
1	71.848
2	71.856
3	71.864
4	71.872
...	...
507	75.804
508	75.812
509	75.820
510	75.827
511	75.835

512 rows × 1 columns

Lees dataset eegdata4.csv in een dataframe genaamd `eeg4` en bekijk de NA-waarden van deze dataset. Hoeveel vind je er per kolom? Hoeveel vind je er in totaal?

```
COUNTER          0
INTERPOLATED     0
RAW_CQ           0
AF3              1
F7               0
F3               0
FC5              0
T7               0
P7               0
O1               0
O2               1
P8               0
T8               0
FC6              0
F4               0
F8               0
AF4              0
GYROX            0
GYROY            0
TIMESTAMP        0
ES_TIMESTAMP     0
FUNC_ID          0
FUNC_VALUE       0
MARKER           0
SYNC_SIGNAL      0
dtype: int64
2
```

Verander de NA-waarden naar de waarde -1, maar sla het resultaat niet op in het originele dataframe (dus voer de veranderingen niet door). Controleer of er dan effectief geen NA-waarden meer aanwezig zijn in het resulterende dataframe.

```
0
```

Verwijder nu alle rijen met een NA waarde met één commando, maar sla het resultaat ditmaal op in een nieuw dataframe genaamd *eeg4_zonder_na*.

Maak een nieuw dataframe, enkel de kolommen 4 tm 17.

	AF3	F7	F3	FC5	T7	P7	O1	O2	P8	
0	4354.358974	4220.000000	4356.410256	4400.000000	4369.743590	4481.538462	4601.025641	4024.102564	4436.410256	4
1	4347.692308	4215.897436	4348.717949	4397.948718	4367.692308	4480.000000	4596.410256	4024.102564	4434.871795	4
2	4341.025641	4206.153846	4348.205128	4394.358974	4364.102564	4476.923077	4590.769231	4016.410256	4432.820513	4
3	4335.897436	4203.589744	4349.743590	4397.435897	4362.564103	4480.000000	4594.871795	4010.256410	4430.256410	4
4	4335.384615	4205.641026	4345.128205	4399.487179	4365.641026	4485.641026	4600.512821	4014.358974	4431.282051	4

Bereken per kolom het gemiddelde. Gebruik de .mean()-methode

```
AF3    4359.014845
F7     4206.277383
F3     4347.994740
FC5    4405.499673
T7     4360.946746
P7     4469.757085
O1     4589.125575
O2     4011.686910
P8     4427.245233
T8     4333.556813
FC6    4547.613412
F4     4234.477318
F8     4182.287968
AF4    4982.614705
dtype: float64
```

Bereken per rij het gemiddelde. Gebruik eveneens de .mean()-methode data.mean(axis=1)

Wijzig het data type van timestamp (een float) naar een aantal microseconden sinds de start. De EEG headset doet namelijk 512 metingen elke 4ms.

Ga als volgt te werk:

1. Gebruik een reguliere expressie om de milliseconden te extraheren uit de timestamp. Dit doe je met een matching groep.
2. Cast dat resultaat naar een int en zet die gegevens om naar `timedelta` -object met de `to_timedelta` -functie waarbij je de `unit` gelijkstelt aan `ms`. Stop het resultaat in een series genaamd `milliseconden`.
3. Doe gelijkaardige stappen voor de seconden.
4. Trek van alle seconden het minimum seconden af.
5. Tel de seconden op bij de milliseconden.

```
0     0 days 00:00:00.840000
1     0 days 00:00:00.848000
2     0 days 00:00:00.856000
3     0 days 00:00:00.864000
4     0 days 00:00:00.872000
              ...
507   0 days 00:00:04.804000
508   0 days 00:00:04.812000
509   0 days 00:00:04.820000
510   0 days 00:00:04.827000
511   0 days 00:00:04.835000
Name: TIMESTAMP, Length: 512, dtype: timedelta64[ns]
```