ENSIAS AI CLUB

DATA PREPERATION

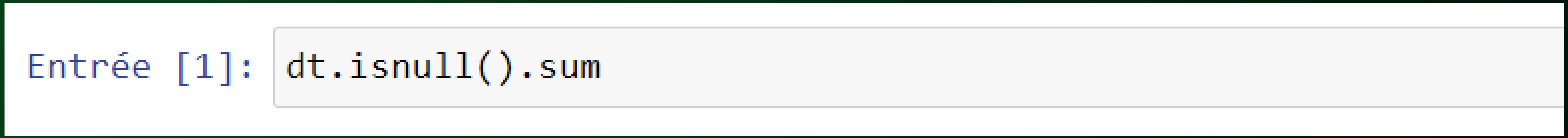
BY : EL KAMEL ISMAIL

1. Data Cleaning
2. Categorical Data
3. Normalisation/Standarisation
4. Feature Selection

PLAN

DATA CLEANING

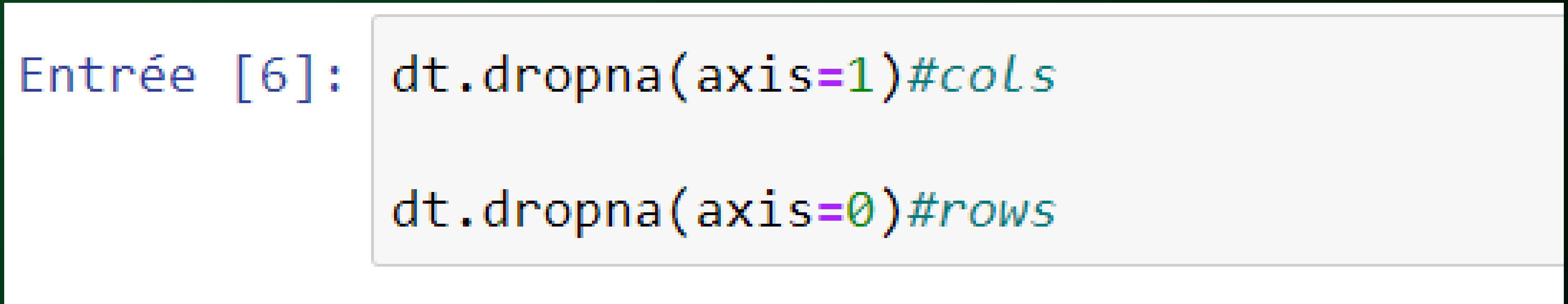
- MISSING VALUES



```
dt.isnull().sum
```

DATA CLEANING

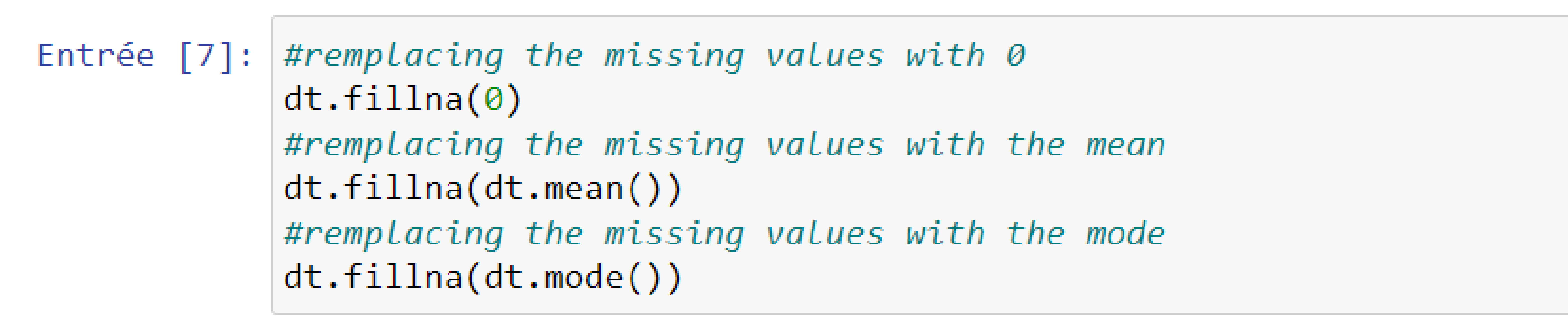
- DELETING THE MISSING VALUES



```
dt.dropna(axis=1)#cols

dt.dropna(axis=0)#rows
```

DATA CLEANING

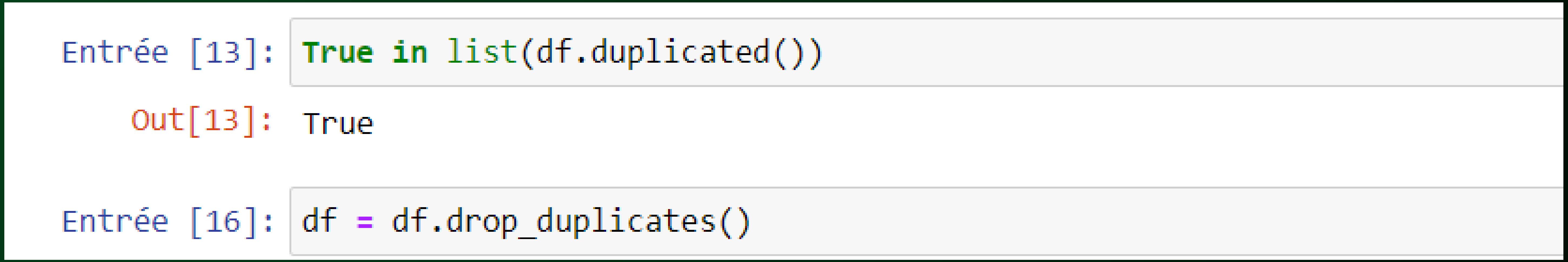
- REMPLACING THE MISSING VALUES



```
#remplacing the missing values with 0
dt.fillna(0)
#remplacing the missing values with the mean
dt.fillna(dt.mean())
#remplacing the missing values with the mode
dt.fillna(dt.mode())
```

DATA CLEANING

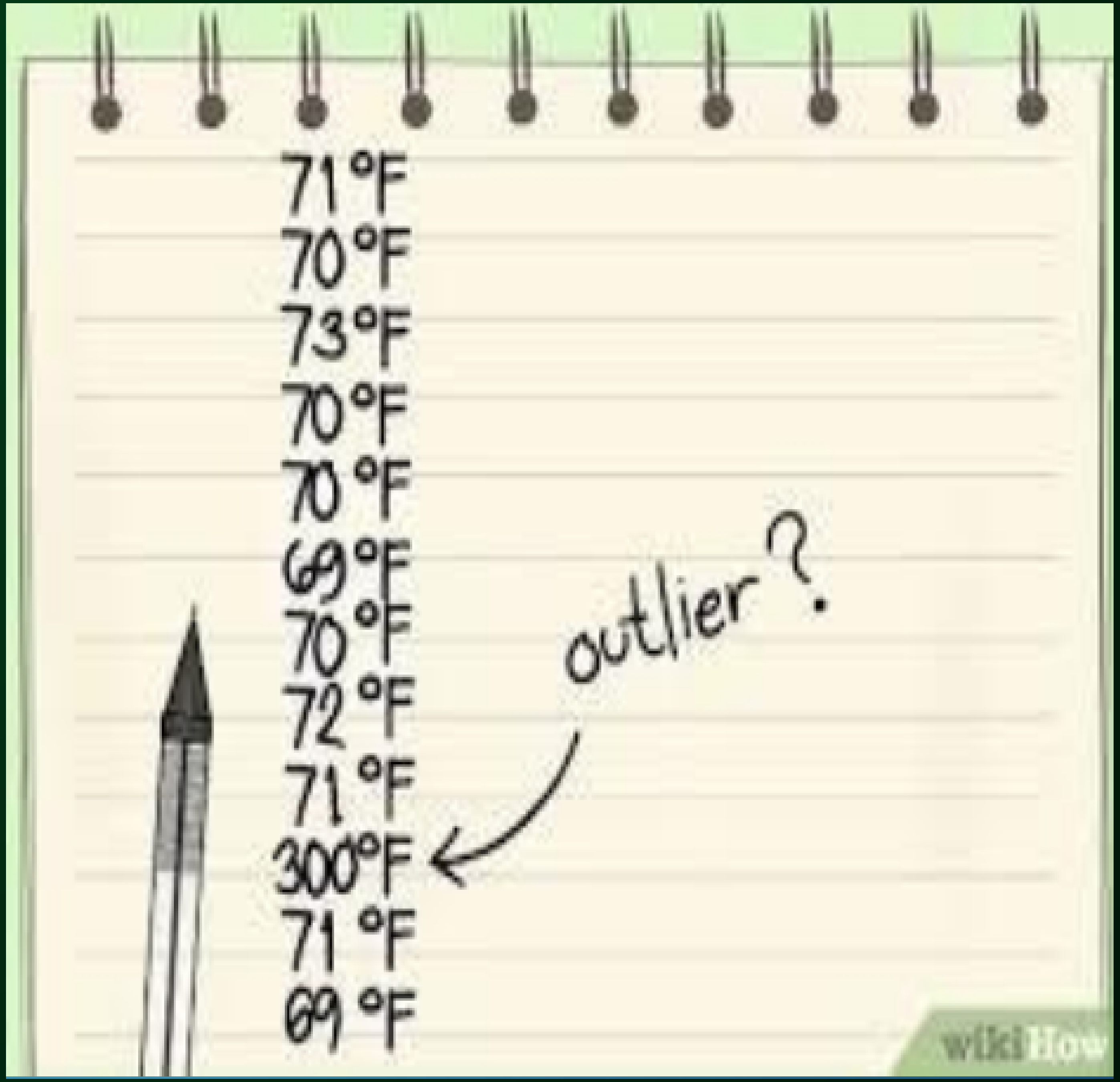
- DUPLICATED ROWS



```
Entrée [13]: True in list(df.duplicated())

    Out[13]: True

Entrée [16]: df = df.drop_duplicates()
```

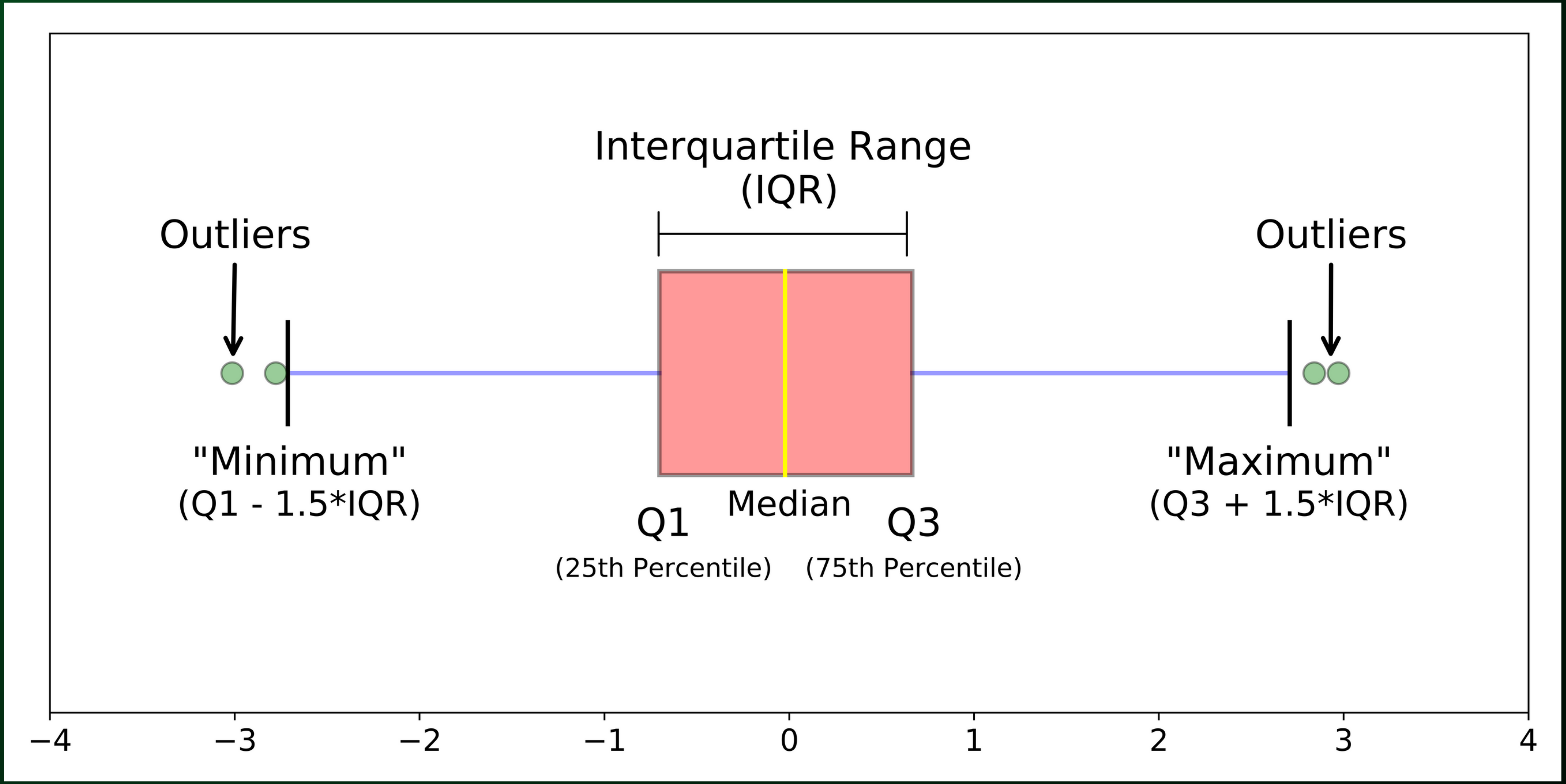
DATA CLEANING

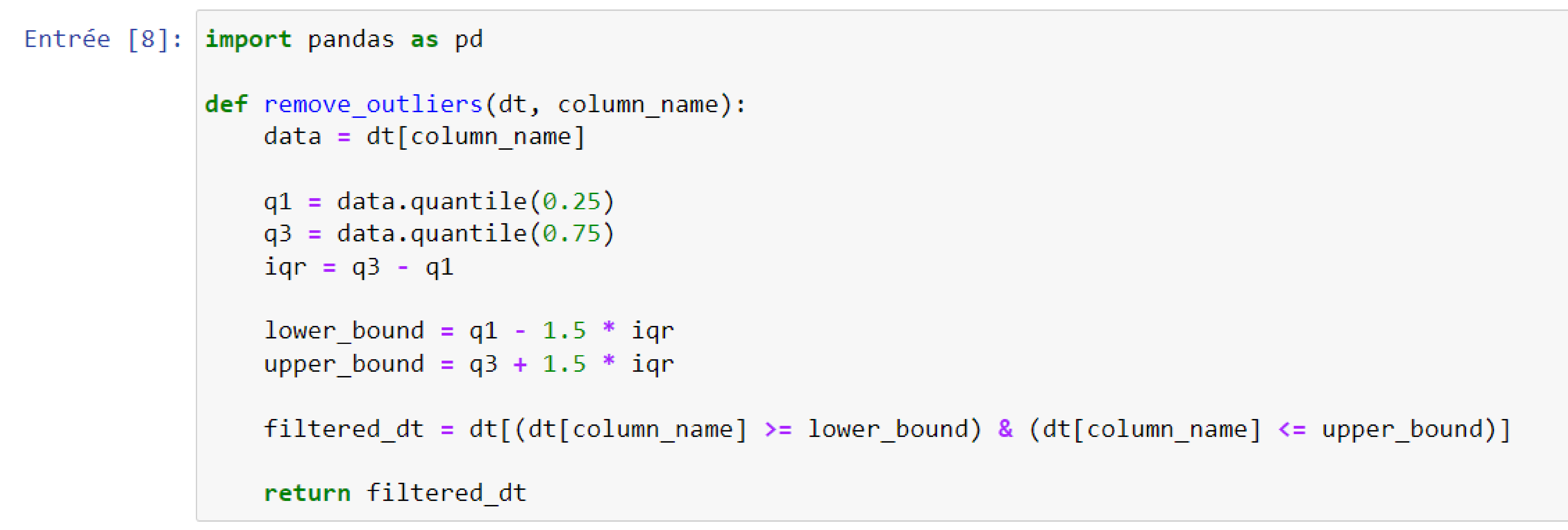
- OUTLIERS



DATA CLEANING

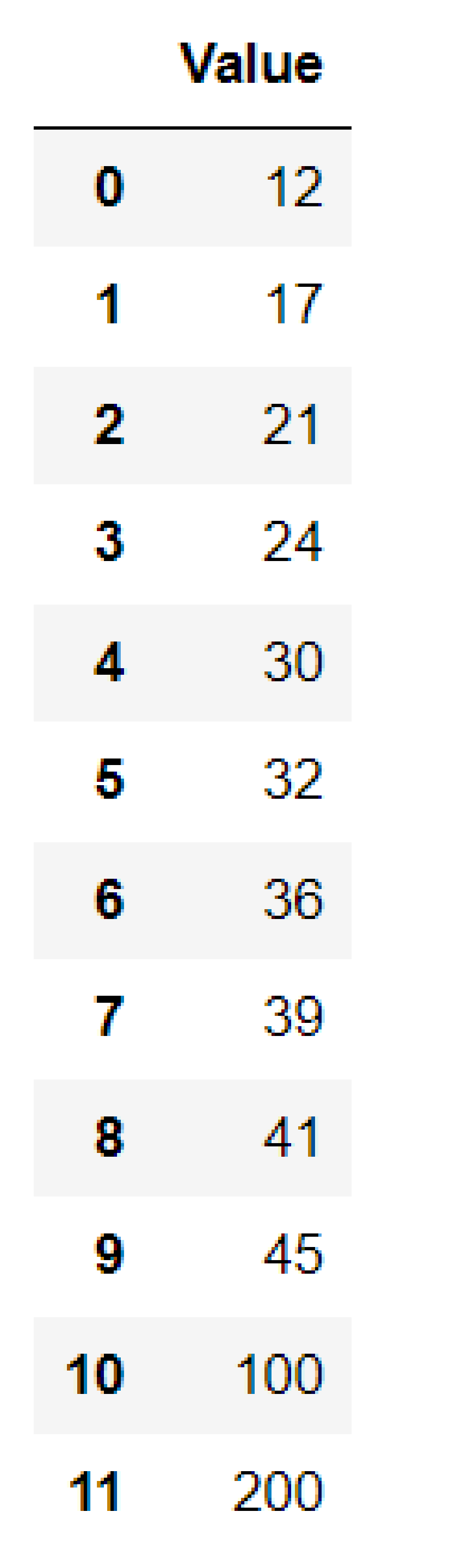
- **DELETING THE OUTLIERS**



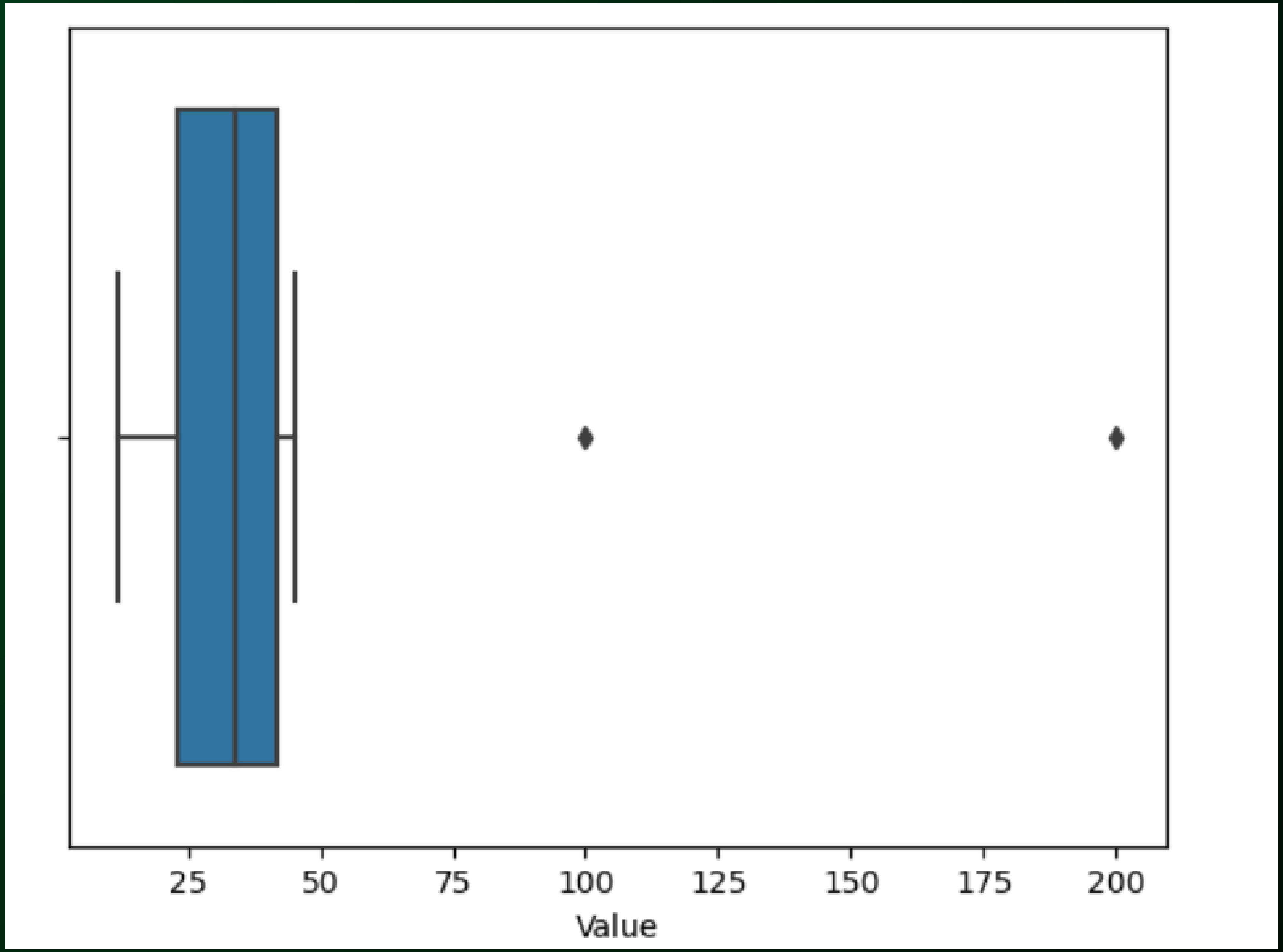


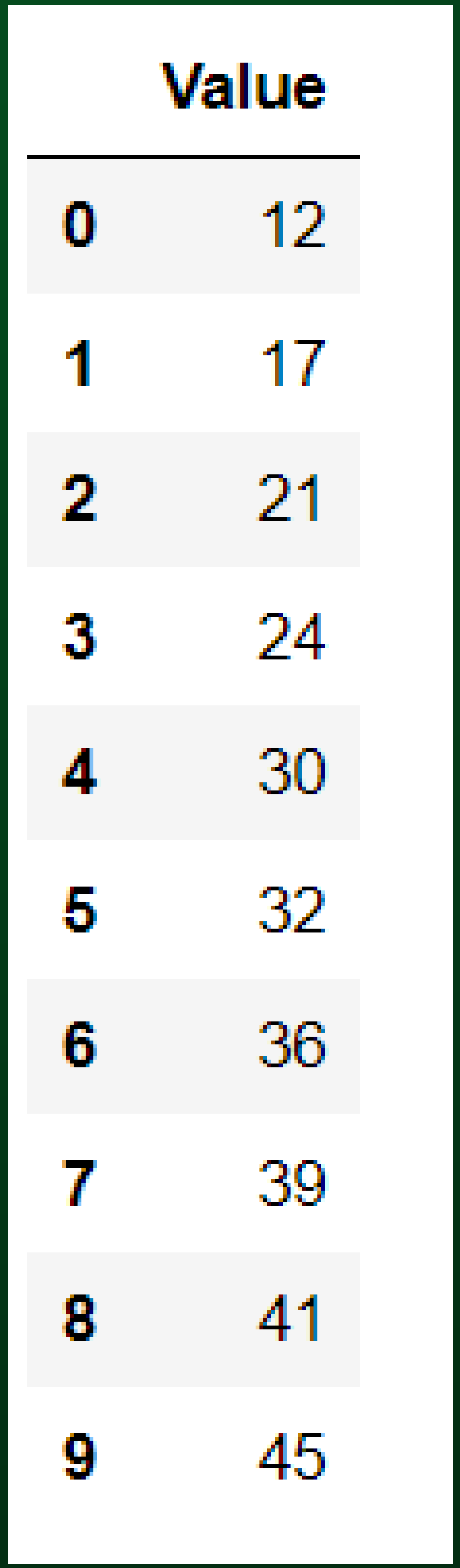
Entrée [8]:

```
import pandas as pd

def remove_outliers(dt, column_name):
    data = dt[column_name]

    q1 = data.quantile(0.25)
    q3 = data.quantile(0.75)
    iqr = q3 - q1

    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr

    filtered_dt = dt[(dt[column_name] >= lower_bound) & (dt[column_name] <= upper_bound)]

    return filtered_dt
```

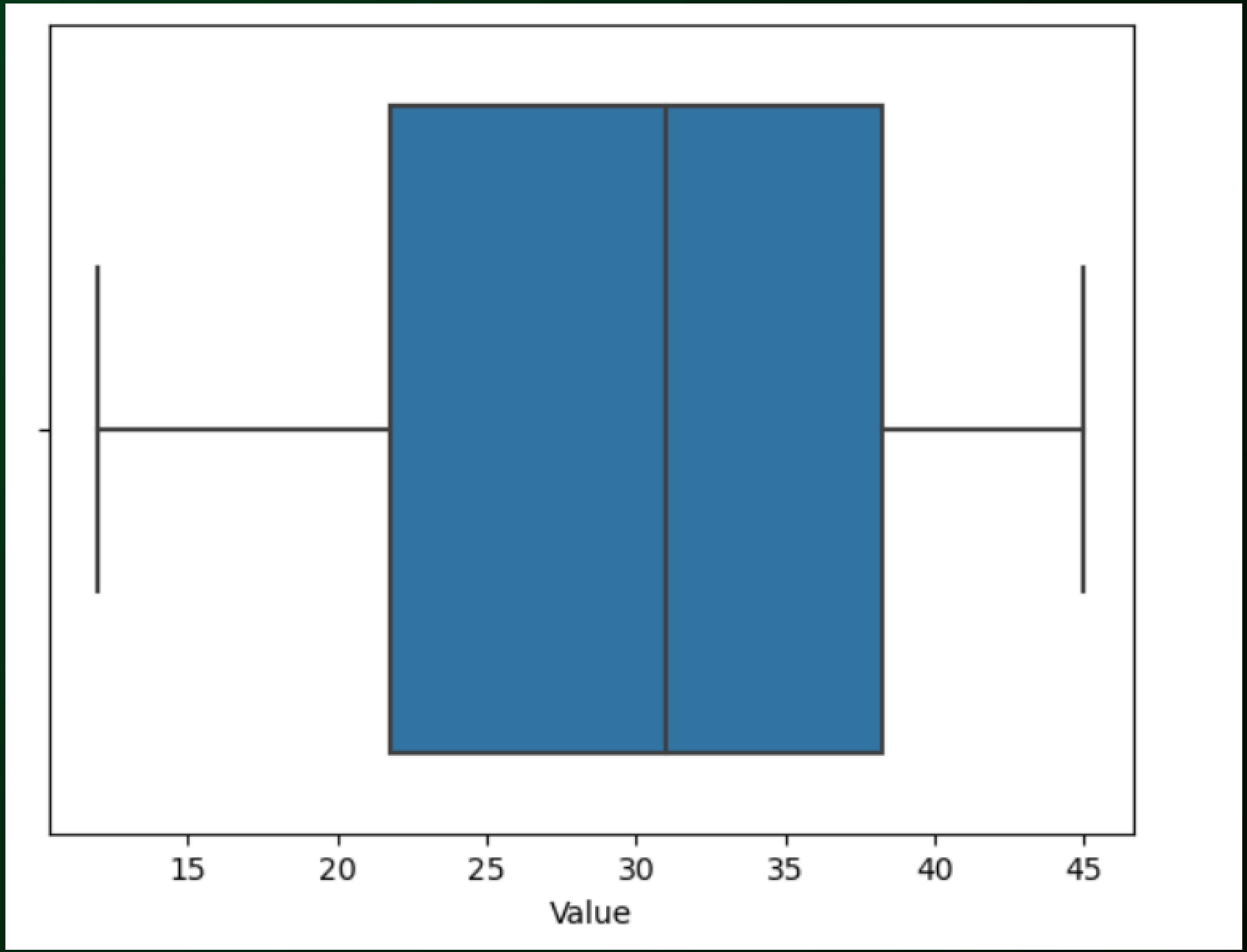


	Value
0	12
1	17
2	21
3	24
4	30
5	32
6	36
7	39
8	41
9	45
10	100
11	200



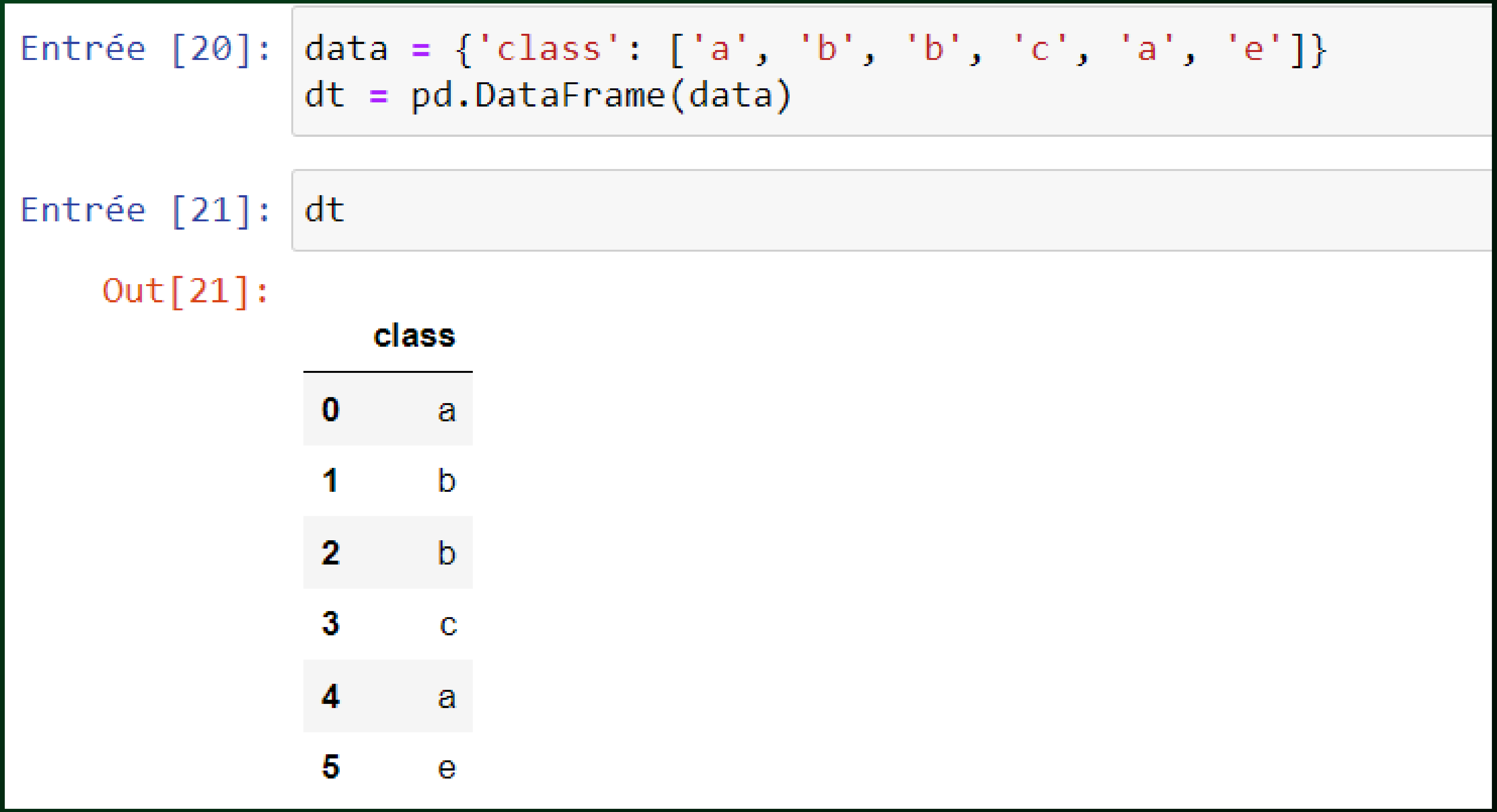


	Value
0	12
1	17
2	21
3	24
4	30
5	32
6	36
7	39
8	41
9	45



CATEGORICAL DATA

- LABEL ENCODER

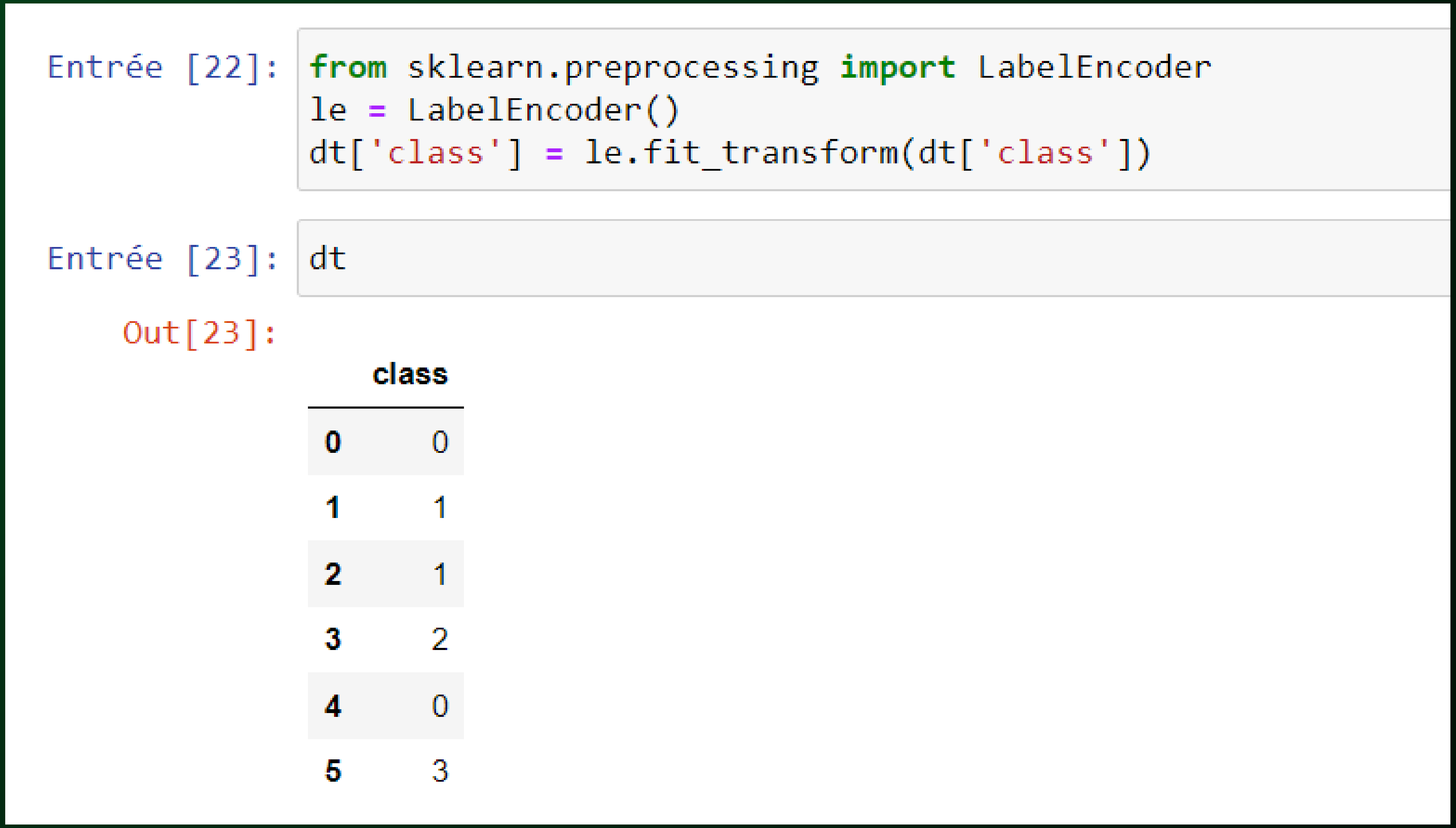


```
Entrée [20]: data = {'class': ['a', 'b', 'b', 'c', 'a', 'e']}
             dt = pd.DataFrame(data)

Entrée [21]: dt

   Out[21]:
```

	class
0	a
1	b
2	b
3	c
4	a
5	e

CATEGORICAL DATA

- LABEL ENCODER

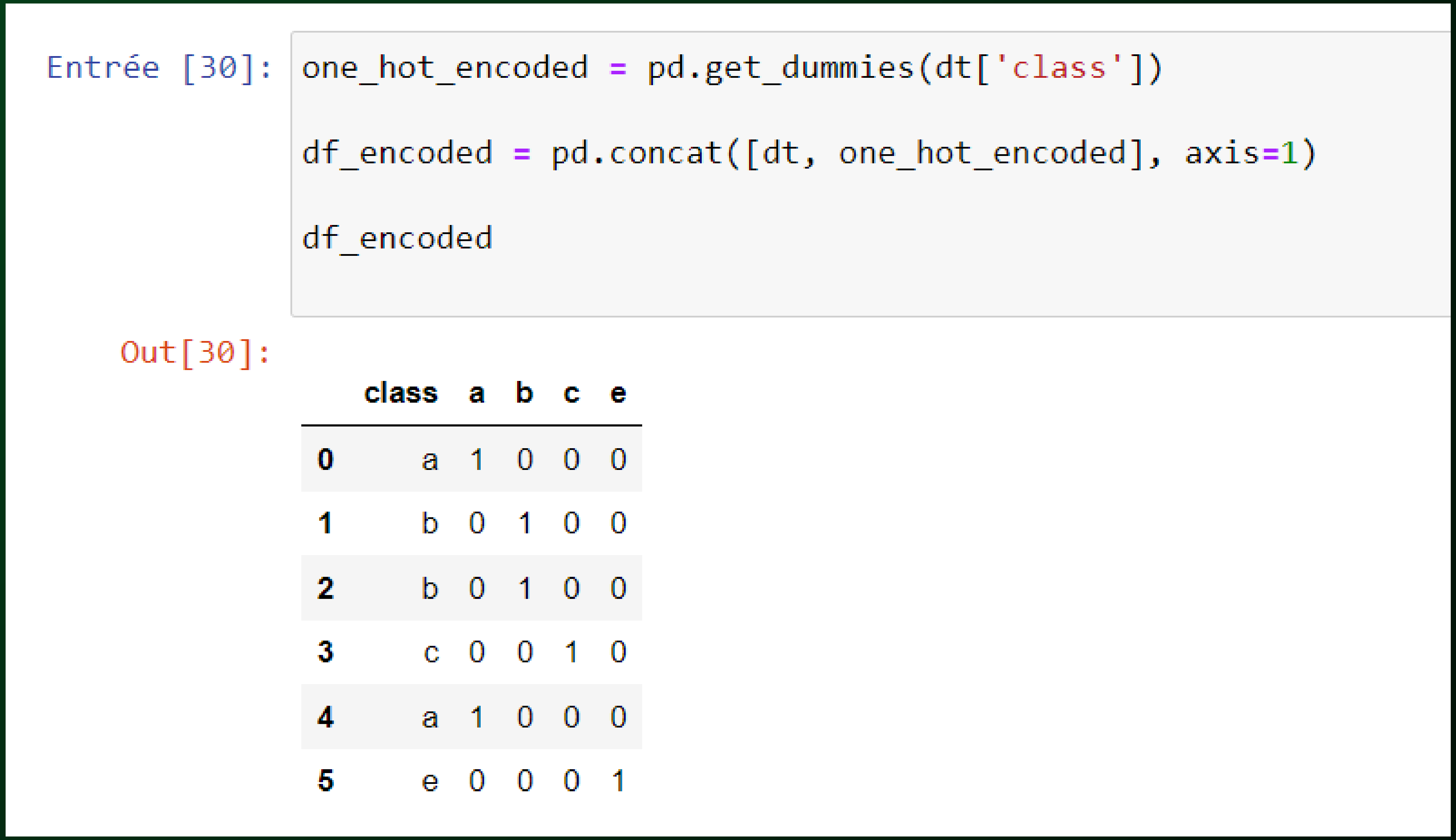


```
Entrée [22]: from sklearn.preprocessing import LabelEncoder
             le = LabelEncoder()
             dt['class'] = le.fit_transform(dt['class'])

Entrée [23]: dt

   Out[23]:
```

	class
0	0
1	1
2	1
3	2
4	0
5	3

CATEGORICAL DATA

- DUMMIES



```
Entrée [30]: one_hot_encoded = pd.get_dummies(dt['class'])

             df_encoded = pd.concat([dt, one_hot_encoded], axis=1)

             df_encoded
```

Out[30]:

	class	a	b	c	e
0	a	1	0	0	0
1	b	0	1	0	0
2	b	0	1	0	0
3	c	0	0	1	0
4	a	1	0	0	0
5	e	0	0	0	1

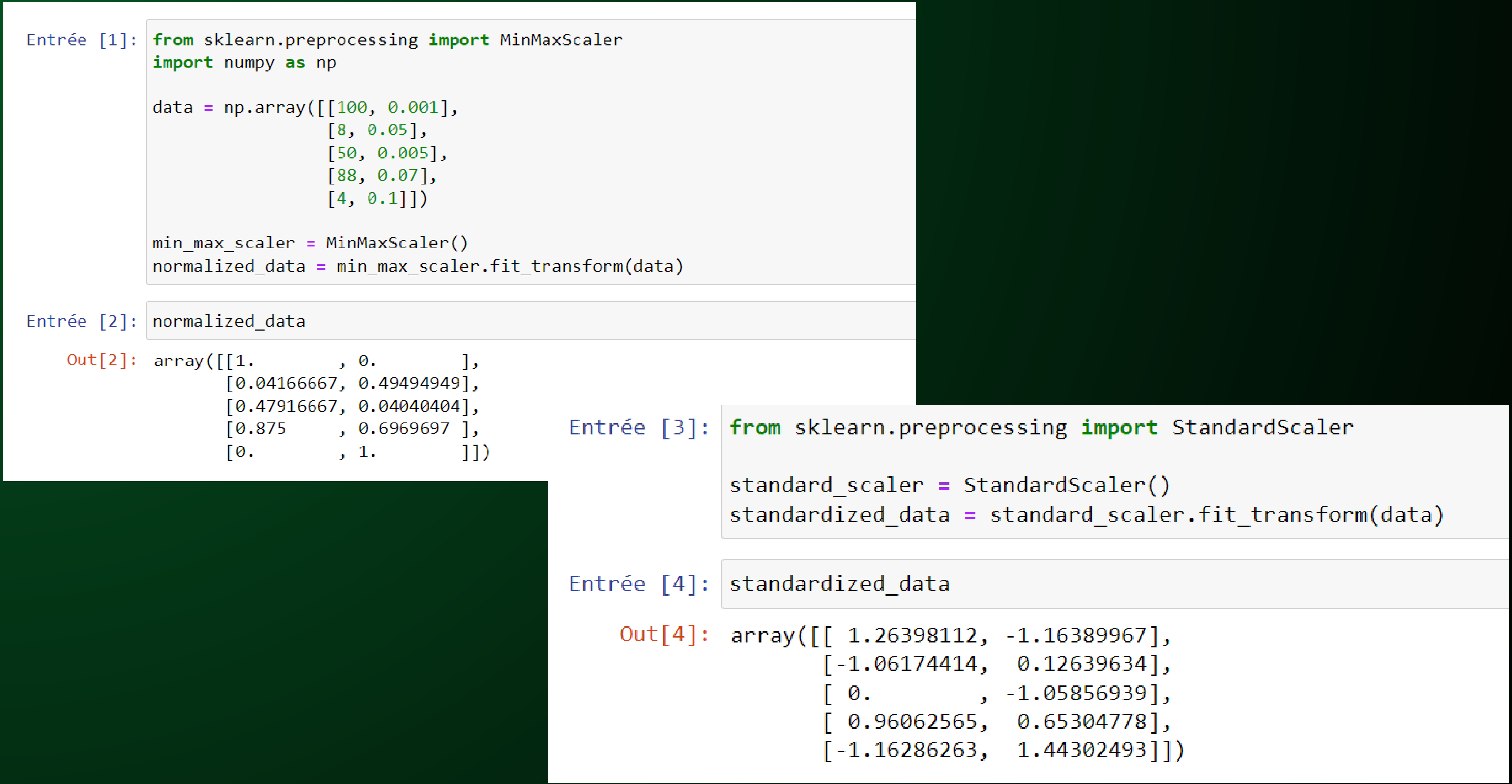
NORMALISATION / STANDARISATION

- NORMALISATION

$$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- STANDARISATION

$$Z = \frac{x - \mu}{\sigma}$$



Entrée [1]:
```
from sklearn.preprocessing import MinMaxScaler
import numpy as np

data = np.array([[100, 0.001],
                 [8, 0.05],
                 [50, 0.005],
                 [88, 0.07],
                 [4, 0.1]])

min_max_scaler = MinMaxScaler()
normalized_data = min_max_scaler.fit_transform(data)
```

Entrée [2]:
```
normalized_data
```

Out[2]:
```
array([[1.        , 0.        ],
       [0.04166667, 0.49494949],
       [0.47916667, 0.04040404],
       [0.875     , 0.6969697 ],
       [0.        , 1.        ]])
```

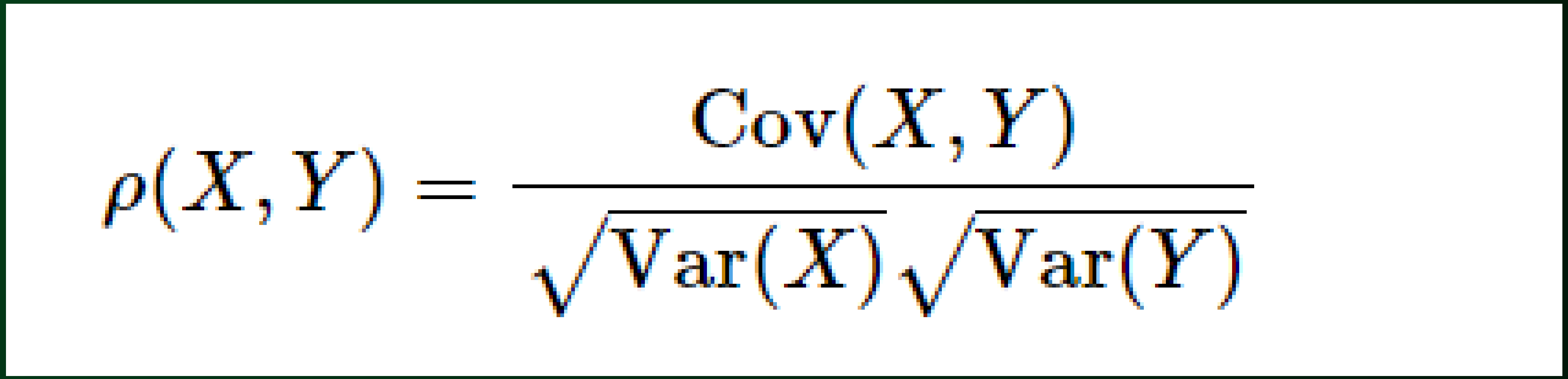
Entrée [3]:
```
from sklearn.preprocessing import StandardScaler

standard_scaler = StandardScaler()
standardized_data = standard_scaler.fit_transform(data)
```

Entrée [4]:
```
standardized_data
```

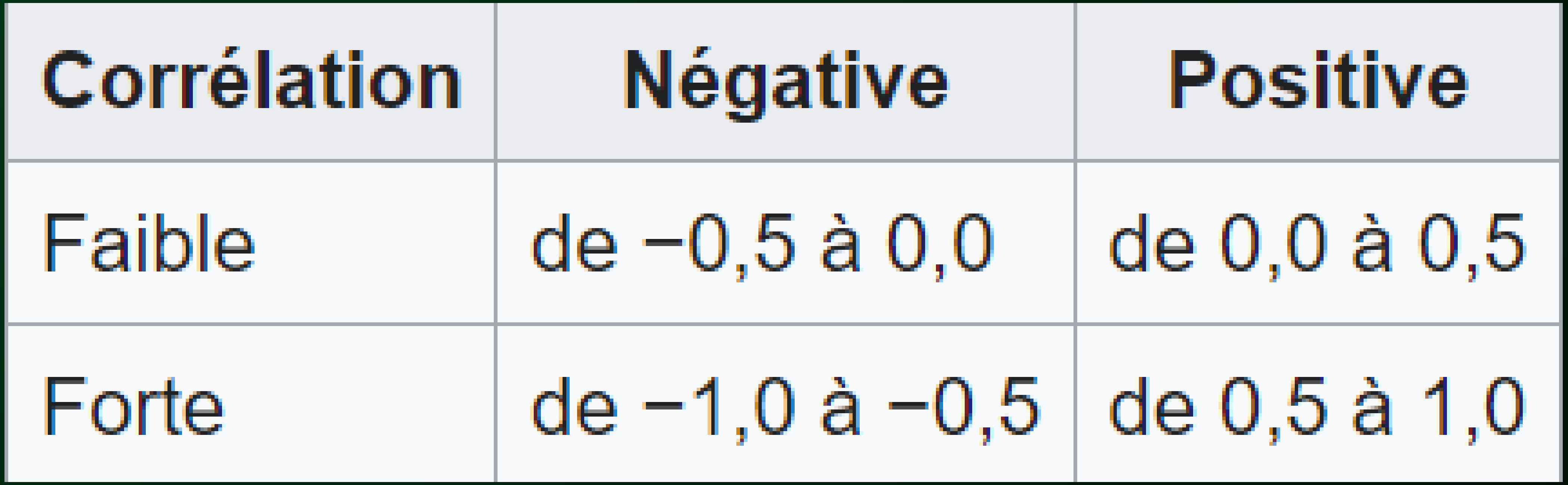
Out[4]:
```
array([[ 1.26398112, -1.16389967],
       [-1.06174414,  0.12639634],
       [ 0.        , -1.05856939],
       [ 0.96062565,  0.65304778],
       [-1.16286263,  1.44302493]])
```

FEATURE SELECTION

- CORRELATION COEFFICIENT



$$\rho(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)}\sqrt{\mathrm{Var}(Y)}}$$



Corrélation	Négative	Positive
Faible	de −0,5 à 0,0	de 0,0 à 0,5
Forte	de −1,0 à −0,5	de 0,5 à 1,0

FEATURE SELECTION

- **CORRELATION MATRIX**

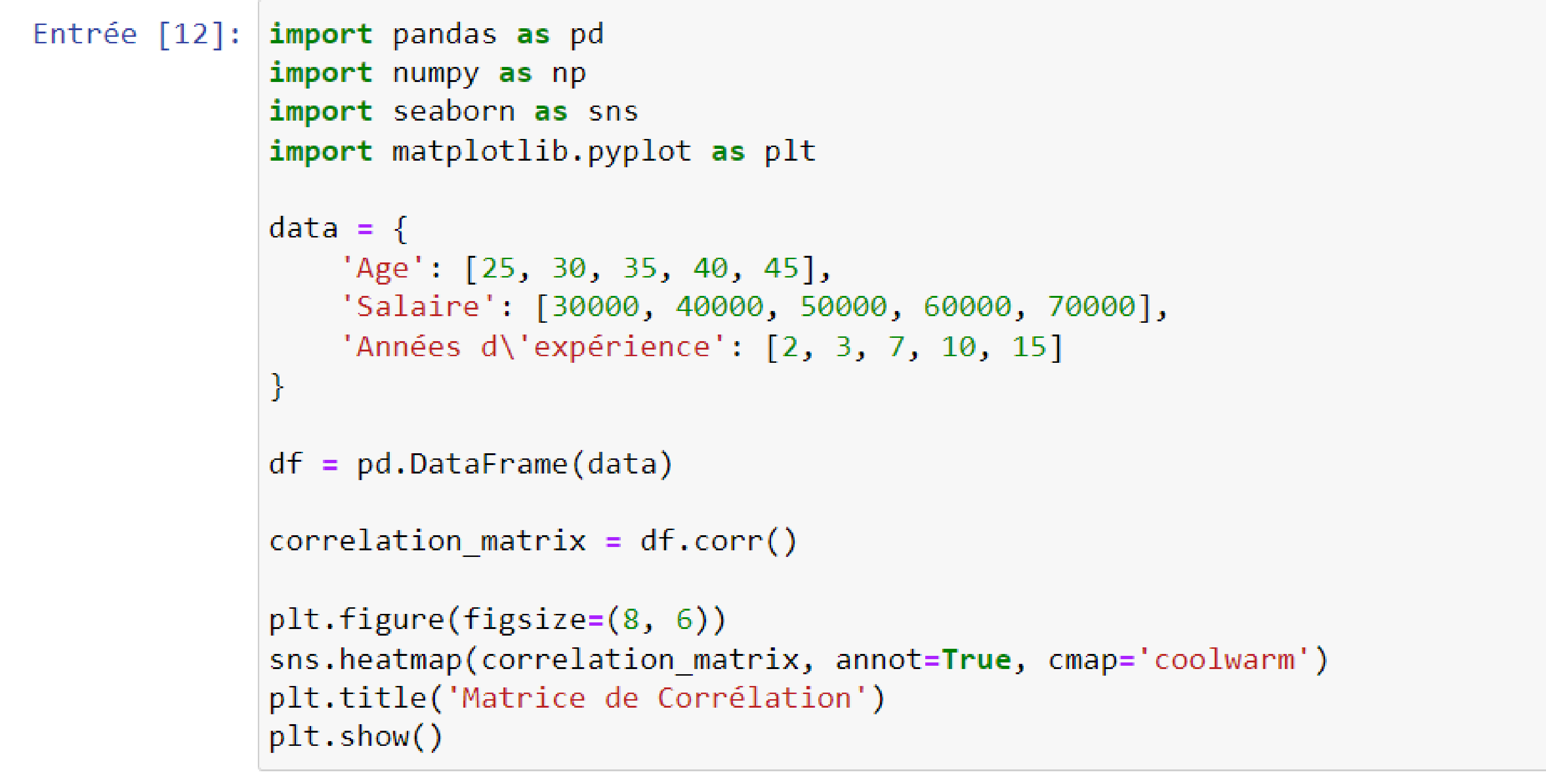
SOIT XI ($\forall$ I $\in$ {1,....,N}) DES VARIABLES, LA MATRICE DE CORRÉLATION EST LA MATRICE R DÉFINIT PAR:

$$\forall\ i,j \in \{1,....,n\} \qquad R_{ij} = \mathrm{Corr}(X_i,X_j)$$

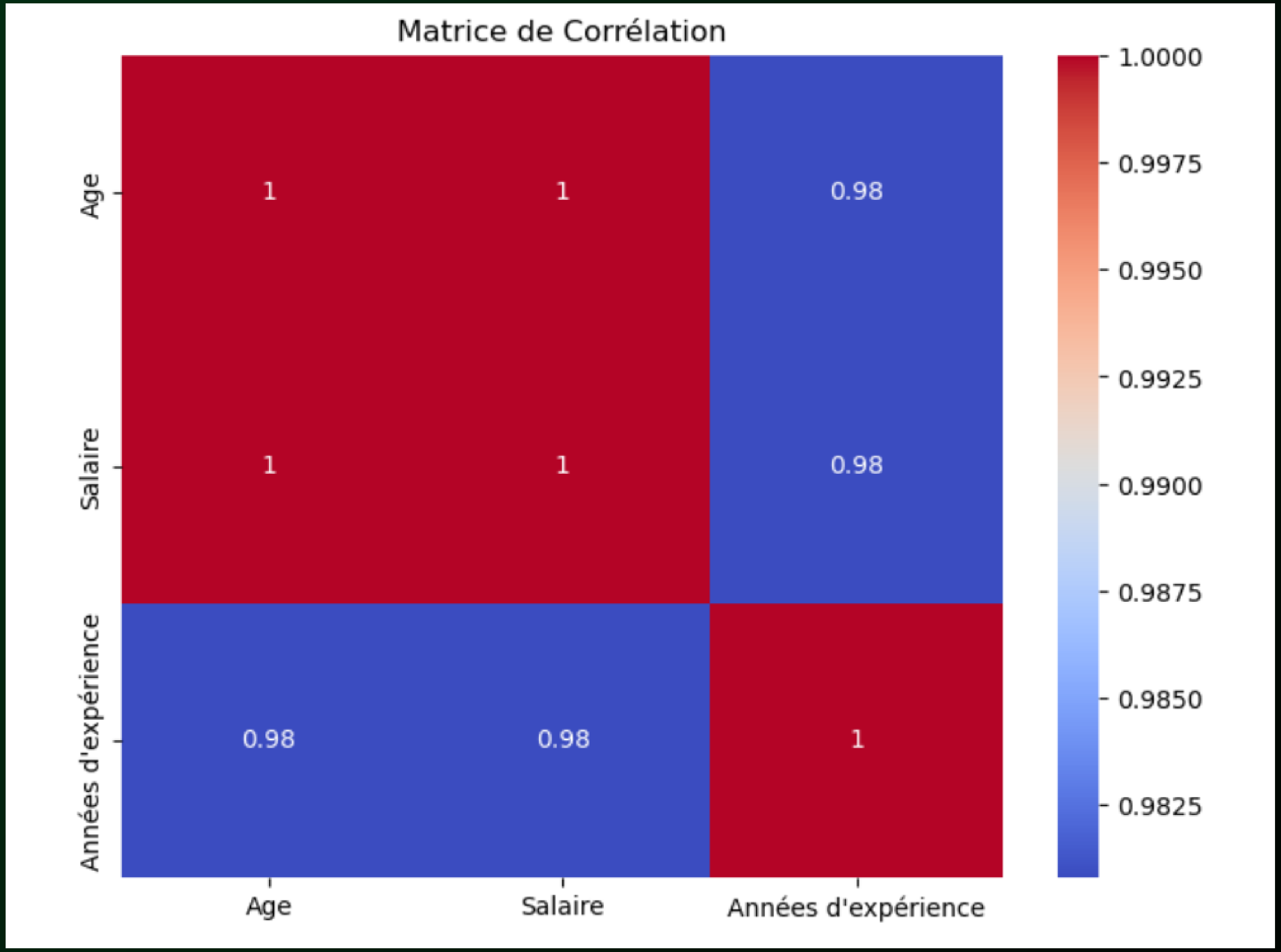
TRANSFORMATION DES DONNÉES

• SÉLECTION DES CARACTÉRISTIQUES

• MATRICE DE CORRÉLATION



```
Entrée [12]: import pandas as pd
             import numpy as np
             import seaborn as sns
             import matplotlib.pyplot as plt

             data = {
                 'Age': [25, 30, 35, 40, 45],
                 'Salaire': [30000, 40000, 50000, 60000, 70000],
                 'Années d\'expérience': [2, 3, 7, 10, 15]
             }

             df = pd.DataFrame(data)

             correlation_matrix = df.corr()

             plt.figure(figsize=(8, 6))
             sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
             plt.title('Matrice de Corrélation')
             plt.show()
```



YALAH JBDO

DB DOK PCS