explore

June 19, 2023

Analyse exploratoire

1 Intro

[1]:
```
# Import des bibliothèques
import os
os.chdir("..")
import pandas as pd
import numpy as np
import re
from utils.helper import parse_name, give_short_name

# Import des données
from data.cleaning import athlet, summer, winter
region = pd.read_csv("regions.csv", index_col=0)
```

[2]:
```
print("'athlet':")
print(f"{athlet.shape[0]} lignes")
print(f"{athlet.shape[1]} colonnes")
```

```
'athlet':
286237 lignes
12 colonnes
```

[3]:
```
athlet.head(3)
```

[3]:
```
                        Name Sex   Age     Team  NOC        Games  Year  \
index
S000001            A Dijiang   M  24.0    China  CHN  1992 Summer  1992
S000002             A Lamusi   M  23.0    China  CHN  2012 Summer  2012
S000003  Gunnar Nielsen Aaby   M  24.0  Denmark  DEN  1920 Summer  1920

         Season       City       Sport                          Event Medal
index
S000001  Summer  Barcelona  Basketball    Basketball Men's Basketball   NaN
S000002  Summer     London        Judo   Judo Men's Extra-Lightweight   NaN
S000003  Summer  Antwerpen    Football        Football Men's Football   NaN
```

2 Athlètes : Nettoyage des données

2.1 Extraire les noms

[4]:
```
person = (pd.DataFrame(athlet
                        .groupby("Name").Event.agg('count')
                        .sort_values(ascending=False)
                        .reset_index()))
person.head()
```

[4]:
```
                        Name  Event
0        Robert Tait McKenzie     58
1   Heikki Ilmari Savolainen     39
2      Joseph "Josy" Stoffel     38
3        Ioannis Theofilakis     36
4               Takashi Ono     33
```

[5]:
```
person[person.Name.str.contains("Phelps")]
```

[5]:
```
                                        Name  Event
15                  Michael Fred Phelps, II     30
4169             Richard Lawson Phelps, Jr.      7
5084              Robert Lawson Phelps, Sr.      6
7151     Jaycie Lynn Phelps (-McClure, -Marus)      6
9352     Monica Kathleen Rutherford (-Phelps)      5
31508                      Mason Elliott Phelps      2
32809                     Harlow Phelps Rothert      2
39453                  John Phelps "Jack" George      2
58195                         Brian Eric Phelps      2
72932                    Richard Charles Phelps      1
74357                      Robert Edward Phelps      1
91583                              Peter Phelps      1
138023                 Harold Roy "Pete" Phelps      1
```

[6]:
```
person["ShortName"] = person.Name.apply(give_short_name)
person["FirstName"] = person.Name.apply(lambda s: parse_name(s)["first"])
person["LastName"] = person.Name.apply(lambda s: parse_name(s)["last"].upper())
person = person[["Name", "FirstName", "LastName", "ShortName", "Event"]]
person.head()
```

[6]:
```
                        Name FirstName     LastName           ShortName  Event
0        Robert Tait McKenzie    robert     MCKENZIE      Robert MCKENZIE     58
1   Heikki Ilmari Savolainen    heikki   SAVOLAINEN   Heikki SAVOLAINEN     39
2      Joseph "Josy" Stoffel    joseph      STOFFEL      Joseph STOFFEL     38
3        Ioannis Theofilakis   ioannis  THEOFILAKIS  Ioannis THEOFILAKIS     36
4               Takashi Ono   takashi          ONO         Takashi ONO     33
```

[7]:
```
short_names = (person
 .groupby("ShortName")
 .sum("Event")
 .sort_values(by="Event", ascending=False))
short_names.to_csv("../../draft/athlete_short_names.csv")
short_names.head(3)
```

[7]:
```
                   Event
ShortName
Robert MCKENZIE       58
Gustaf CARLBERG       49
Heikki SAVOLAINEN     39
```

[8]:
```
person[person["LastName"] == "SCHMIDT"]
```

[8]:
```
                                    Name  FirstName LastName  \
2662                       Iohan Schmidt      iohan  SCHMIDT
7159            Magdalena Schmidt (-Jakob)  magdalena  SCHMIDT
8132              Oscar Daniel Bezerra Schmidt      oscar  SCHMIDT
10367                    Florian Schmidt    florian  SCHMIDT
15015              Ingrid Schmidt (-Naue)     ingrid  SCHMIDT
…                                      …        …      …
133023                   Herbert Schmidt    herbert  SCHMIDT
133885                  Heinrich Schmidt   heinrich  SCHMIDT
134636  Ingrid Ulrike Schmidt (-Koppers)     ingrid  SCHMIDT
136868                    Gustav Schmidt     gustav  SCHMIDT
141995                     Josef Schmidt      josef  SCHMIDT

                ShortName  Event
2662        Iohan SCHMIDT      8
7159    Magdalena SCHMIDT      6
8132        Oscar SCHMIDT      5
10367     Florian SCHMIDT      4
15015      Ingrid SCHMIDT      4
…                     …    …
133023    Herbert SCHMIDT      1
133885   Heinrich SCHMIDT      1
134636     Ingrid SCHMIDT      1
136868     Gustav SCHMIDT      1
141995      Josef SCHMIDT      1

[89 rows x 5 columns]
```

[9]:
```
names = (person
            .groupby(["ShortName", "FirstName", "LastName", "Name"])
            .sum("Event").reset_index()
            .sort_values(by="Event", ascending=False))
```

```
names.to_csv("../../draft/athlete_names.csv", index=None, chunksize=10000)
names
```

[9]:
```
                  ShortName FirstName     LastName  \
116699      Robert MCKENZIE    robert     MCKENZIE
51535     Heikki SAVOLAINEN    heikki   SAVOLAINEN
69622        Joseph STOFFEL    joseph      STOFFEL
56880   Ioannis THEOFILAKIS   ioannis  THEOFILAKIS
129728         Takashi ONO   takashi          ONO
…                       …       …           …
61070       Janier HERNNDEZ    janier     HERNNDEZ
61069            Janie REED     janie         REED
61067        Janice TROMBLY    janice      TROMBLY
61066       Janice TEIXEIRA    janice     TEIXEIRA
146360       Zzimo CALAZANS     zzimo     CALAZANS

                               Name  Event
116699          Robert Tait McKenzie     58
51535       Heikki Ilmari Savolainen     39
69622          Joseph "Josy" Stoffel     38
56880              Ioannis Theofilakis     36
129728                 Takashi Ono     33
…                              …    …
61070      Janier Concepcin Hernndez      1
61069                    REED Janie      1
61067   Janice Yvonne "Jan" Trombly      1
61066             Janice Gil Teixeira      1
146360          Zzimo Alves Calazans      1

[146361 rows x 5 columns]
```

[10]:
```
print(146361 - 141513)
```

```
4848
```

[11]:
```
person[person["ShortName"] == "Zsuzsanna JAKABOS"]
```

[11]:
```
                          Name  FirstName LastName          ShortName  Event
1476    Zsuzsanna "Zsu" Jakabos  zsuzsanna  JAKABOS  Zsuzsanna JAKABOS     10
144651        JAKABOS Zsuzsanna  zsuzsanna  JAKABOS  Zsuzsanna JAKABOS      1
```

2.2 Age : passer en entier

[12]:
```
# On s'assure que tous les ages sont entiers
# avant de les convertir en "int"
athlet.Age.apply(lambda x: x.is_integer()).value_counts()
```

[12]:
```
Age
True     276763
False      9474
Name: count, dtype: int64
```

[13]:
```
# On s'assure que les ages non entiers
# ne correspondent qu'aux valeurs vides
athlet["intAge"] = athlet.Age.apply(lambda x: x.is_integer())
athlet.Age.isna().sum() == athlet[athlet["intAge"] == False].shape[0]
```

[13]:
```
True
```

[14]:
```
#assert len(athlete[athlete.Age == 0]) == 0
#athlete["Age"].fillna(0, inplace=True)
#athlete["Age"] = athlete.Age.astype("int64")
#athlete.replace(0, np.nan, inplace=True)
athlet["Age"] = athlet.Age.astype("Int64")
athlet.drop(columns=["intAge"], inplace=True)
athlet.head()
```

[14]:
```
                                       Name Sex  Age            Team  NOC  \
index
S000001                           A Dijiang   M   24           China  CHN
S000002                            A Lamusi   M   23           China  CHN
S000003                 Gunnar Nielsen Aaby   M   24         Denmark  DEN
S000004                Edgar Lindenau Aabye   M   34  Denmark/Sweden  DEN
S000005  Cornelia "Cor" Aalten (-Strannood)   F   18     Netherlands  NED

               Games  Year  Season         City       Sport  \
index
S000001  1992 Summer  1992  Summer    Barcelona  Basketball
S000002  2012 Summer  2012  Summer       London        Judo
S000003  1920 Summer  1920  Summer    Antwerpen    Football
S000004  1900 Summer  1900  Summer        Paris  Tug-Of-War
S000005  1932 Summer  1932  Summer  Los Angeles   Athletics

                                   Event Medal
index
S000001   Basketball Men's Basketball   NaN
S000002  Judo Men's Extra-Lightweight   NaN
S000003       Football Men's Football   NaN
S000004   Tug-Of-War Men's Tug-Of-War  Gold
S000005  Athletics Women's 100 metres   NaN
```

[15]:
```
athlet[athlet.Age.isna()]
```

```
[15]:                       Name Sex   Age          Team  NOC        Games  \
      index
      S000086  Mohamed Jamshid Abadi   M  <NA>          Iran  IRI  1948 Summer
      S000091       Georgi Abadzhiev   M  <NA>      Bulgaria  BUL  1924 Summer
      S000092       Georgi Abadzhiev   M  <NA>      Bulgaria  BUL  1924 Summer
      S000101        Mohamed Abakkar   M  <NA>         Sudan  SUD  1972 Summer
      S000151      Sayed Fahmy Abaza   M  <NA>         Egypt  EGY  1920 Summer
      …                          …  ..   …           …   …          …
      W047155          Boris Yakimov   M  <NA>  Soviet Union  URS  1956 Winter
      W047156          Boris Yakimov   M  <NA>  Soviet Union  URS  1956 Winter
      W047713       Luciano Zampatti   M  <NA>         Italy  ITA  1928 Winter
      W047757        Ernesto Zardini   M  <NA>         Italy  ITA  1932 Winter
      W047758        Ernesto Zardini   M  <NA>         Italy  ITA  1932 Winter

               Year  Season               City            Sport  \
      index
      S000086  1948  Summer             London           Boxing
      S000091  1924  Summer              Paris          Cycling
      S000092  1924  Summer              Paris          Cycling
      S000101  1972  Summer             Munich           Boxing
      S000151  1920  Summer          Antwerpen         Football
      …         …     …                  …              …
      W047155  1956  Winter  Cortina d'Ampezzo    Speed Skating
      W047156  1956  Winter  Cortina d'Ampezzo    Speed Skating
      W047713  1928  Winter       Sankt Moritz      Ski Jumping
      W047757  1932  Winter        Lake Placid      Ski Jumping
      W047758  1932  Winter        Lake Placid  Nordic Combined

                                                   Event Medal
      index
      S000086                 Boxing Men's Heavyweight   NaN
      S000091         Cycling Men's Road Race, Individual   NaN
      S000092               Cycling Men's Road Race, Team   NaN
      S000101                   Boxing Men's Flyweight   NaN
      S000151                  Football Men's Football   NaN
      …                                              …    …
      W047155            Speed Skating Men's 5,000 metres   NaN
      W047156           Speed Skating Men's 10,000 metres   NaN
      W047713  Ski Jumping Men's Normal Hill, Individual   NaN
      W047757  Ski Jumping Men's Normal Hill, Individual   NaN
      W047758         Nordic Combined Men's Individual   NaN

      [9474 rows x 12 columns]
```

```
[16]: athlet.info()
```

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

```
Index: 286237 entries, S000001 to W048564
Data columns (total 12 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Name    286237 non-null  object
 1   Sex     286237 non-null  object
 2   Age     276763 non-null  Int64
 3   Team    286237 non-null  object
 4   NOC     286237 non-null  object
 5   Games   286237 non-null  object
 6   Year    286237 non-null  int64
 7   Season  286237 non-null  object
 8   City    286237 non-null  object
 9   Sport   286237 non-null  object
 10  Event   286237 non-null  object
 11  Medal   42232 non-null   object
dtypes: Int64(1), int64(1), object(10)
memory usage: 28.7+ MB
```

2.3 Year

[17]:
```
athlet["Year"] = athlet.Year.astype("Int64")
athlet["Year"].sample(n=10)
```

[17]:
```
index
S133381    1972
S187497    2000
S223597    2020
W027020    1968
W010320    1976
S202964    1956
S113226    1980
S096882    1968
S081543    2000
S102519    1996
Name: Year, dtype: Int64
```

[18]:
```
athlet.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 286237 entries, S000001 to W048564
Data columns (total 12 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Name    286237 non-null  object
 1   Sex     286237 non-null  object
 2   Age     276763 non-null  Int64
 3   Team    286237 non-null  object
 4   NOC     286237 non-null  object
 5   Games   286237 non-null  object
 6   Year    286237 non-null  Int64
 7   Season  286237 non-null  object
 8   City    286237 non-null  object
 9   Sport   286237 non-null  object
 10  Event   286237 non-null  object
 11  Medal   42232 non-null   object
dtypes: Int64(2), object(10)
memory usage: 28.9+ MB
```

2.4 Médailles

[19]:
```
compet = (pd.DataFrame(athlet.groupby(["Games", "Event"]).Medal.count())
           .reset_index())
compet.sort_values(by="Medal", ascending=False)
```

[19]:
```
            Games                                              Event  Medal
6228  2020 Summer                                           Men Team    335
6381  2020 Summer                                         Women Team    334
354   1908 Summer                    Gymnastics Men's Team All-Around     94
599   1920 Summer                    Gymnastics Men's Team All-Around     77
466   1912 Summer   Gymnastics Men's Team All-Around, Swedish System     74
…             …                                                  …     …
957   1928 Winter                     Speed Skating Men's 10,000 metres      0
1285  1948 Summer      Art Competitions Mixed Painting, Unknown Event      0
1289  1948 Summer   Art Competitions Mixed Sculpturing, Unknown Event      0
6466  2020 Summer                                      Women's Madison      0
1125  1936 Summer   Art Competitions Mixed Sculpturing, Unknown Event      0

[6498 rows x 3 columns]
```

[20]:
```
compet.to_excel("../../draft/compet.ods")
```

[21]:
```
compet.Medal.describe()
```

[21]:
```
count    6498.000000
mean        6.499231
std        10.449955
min         0.000000
25%         3.000000
50%         3.000000
75%         6.000000
max       335.000000
Name: Medal, dtype: float64
```

[22]:
```
athlet["Medal"].fillna(pd.NA, inplace=True)
```

[23]:
```
athlet["Medal"] = athlet["Medal"].fillna("None")
athlet.Medal.value_counts()
```

[23]:
```
Medal
None      244005
Gold       14172
Bronze     14162
Silver     13898
Name: count, dtype: int64
```

[24]:
```
athlet["Medal"] = athlet.Medal.astype("category")
athlet.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 286237 entries, S000001 to W048564
Data columns (total 12 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Name    286237 non-null  object
 1   Sex     286237 non-null  object
 2   Age     276763 non-null  Int64
 3   Team    286237 non-null  object
 4   NOC     286237 non-null  object
 5   Games   286237 non-null  object
 6   Year    286237 non-null  Int64
 7   Season  286237 non-null  object
 8   City    286237 non-null  object
 9   Sport   286237 non-null  object
 10  Event   286237 non-null  object
 11  Medal   286237 non-null  category
dtypes: Int64(2), category(1), object(9)
memory usage: 27.0+ MB
```

[25]:
```
athlet
```

[25]:
```
                                       Name Sex  Age            Team  NOC  \
index
S000001                           A Dijiang   M   24           China  CHN
S000002                            A Lamusi   M   23           China  CHN
S000003                 Gunnar Nielsen Aaby   M   24         Denmark  DEN
S000004                Edgar Lindenau Aabye   M   34  Denmark/Sweden  DEN
S000005  Cornelia "Cor" Aalten (-Strannood)   F   18     Netherlands  NED
…                                         …  ..  …               …    …
W048560                          Andrzej ya   M   29        Poland-1  POL
W048561                            Piotr ya   M   27          Poland  POL
W048562                            Piotr ya   M   27          Poland  POL
W048563                  Tomasz Ireneusz ya   M   30          Poland  POL
```

```
W048564                  Tomasz Ireneusz ya   M   34         Poland  POL

               Games  Year  Season            City        Sport  \
index
S000001  1992 Summer  1992  Summer       Barcelona   Basketball
S000002  2012 Summer  2012  Summer          London         Judo
S000003  1920 Summer  1920  Summer       Antwerpen     Football
S000004  1900 Summer  1900  Summer           Paris   Tug-Of-War
S000005  1932 Summer  1932  Summer     Los Angeles    Athletics
…                …    …     …             …           …
W048560  1976 Winter  1976  Winter       Innsbruck         Luge
W048561  2014 Winter  2014  Winter           Sochi  Ski Jumping
W048562  2014 Winter  2014  Winter           Sochi  Ski Jumping
W048563  1998 Winter  1998  Winter          Nagano    Bobsleigh
W048564  2002 Winter  2002  Winter  Salt Lake City    Bobsleigh

                                      Event Medal
index
S000001              Basketball Men's Basketball  None
S000002             Judo Men's Extra-Lightweight  None
S000003                  Football Men's Football  None
S000004              Tug-Of-War Men's Tug-Of-War  Gold
S000005             Athletics Women's 100 metres  None
…                                           …    …
W048560               Luge Mixed (Men)'s Doubles  None
W048561  Ski Jumping Men's Large Hill, Individual  None
W048562        Ski Jumping Men's Large Hill, Team  None
W048563                     Bobsleigh Men's Four  None
W048564                     Bobsleigh Men's Four  None

[286237 rows x 12 columns]
```

2.5 NOC

[26]:
```
print(athlet.NOC.nunique())
```

```
233
```

3 Region

[27]:
```
assert region.shape[0] == region.NOC.nunique()
```

[28]:
```
region
```

[28]:
```
     NOC      region        notes
0    EOR     Refugee          NaN
```

```
1    LBN      Lebanon           NaN
2    SGP    Singapore           NaN
3    ROC      Russia            NaN
4    AFG  Afghanistan           NaN
..   …            …          …
229  YEM        Yemen           NaN
230  YMD        Yemen  South Yemen
231  YUG       Serbia   Yugoslavia
232  ZAM       Zambia           NaN
233  ZIM     Zimbabwe           NaN

[234 rows x 3 columns]
```

[29]:
```
region.loc[region["NOC"] == "ROT", "region"] = "Refugee"
region.loc[region["NOC"] == "TUV", "region"] = "Tuvalu"
region.loc[region["NOC"] == "UNK", "region"] = "Unknown"
```

[30]:
```
region.columns = map(str.lower, region.columns)
region
```

[30]:
```
     noc       region        notes
0    EOR      Refugee           NaN
1    LBN      Lebanon           NaN
2    SGP    Singapore           NaN
3    ROC       Russia           NaN
4    AFG  Afghanistan           NaN
..   …            …          …
229  YEM        Yemen           NaN
230  YMD        Yemen  South Yemen
231  YUG       Serbia   Yugoslavia
232  ZAM       Zambia           NaN
233  ZIM     Zimbabwe           NaN

[234 rows x 3 columns]
```

[31]:
```
region[~region["notes"].isna()]
```

[31]:
```
     noc                          region                          notes
5    AHO                         Curacao           Netherlands Antilles
10   ANT                         Antigua            Antigua and Barbuda
11   ANZ                       Australia                    Australasia
30   BOH                  Czech Republic                        Bohemia
55   CRT                          Greece                          Crete
92   HKG                           China                      Hong Kong
97   IOA  Individual Olympic Athletes    Individual Olympic Athletes
103  ISV               Virgin Islands, US                Virgin Islands
147  NBO                        Malaysia                  North Borneo
```

```
151  NFL                     Canada              Newfoundland
172  ROT                   Refugee    Refugee Olympic Team
179  SCG                    Serbia   Serbia and Montenegro
183  SKN               Saint Kitts  Turks and Caicos Islands
209  TTO                  Trinidad     Trinidad and Tobago
212  TUV                    Tuvalu                  Tuvalu
214  UAR                     Syria   United Arab Republic
217  UNK                   Unknown                 Unknown
227  WIF                  Trinidad  West Indies Federation
228  YAR                     Yemen             North Yemen
230  YMD                     Yemen             South Yemen
231  YUG                    Serbia              Yugoslavia
```

[32]:
```
region[region["region"] == "Tuvalu"]
```

[32]:
```
     noc  region   notes
212  TUV  Tuvalu  Tuvalu
```

[33]:
```
for reg in list(region["region"].unique())[:25]:
    print(reg)
```

```
Refugee
Lebanon
Singapore
Russia
Afghanistan
Curacao
Albania
Algeria
Andorra
Angola
Antigua
Australia
Argentina
Armenia
Aruba
American Samoa
Austria
Azerbaijan
Bahamas
Bangladesh
Barbados
Burundi
Belgium
Benin
Bermuda
```