**Date: 10th December, 2020**

**XMAS INDIVIDUAL PROJECT**

**The project is about hypothesis: "The winners of the best running races in the over the world has been won by African athletes".**

**It's going to show with two datasets: first one is about the winners in 120 years of Olympic Games in the Sport's History, and the second one is about the six best world marathon majors.**

**legend file:**

- **blue box markdown: alert info about the file's content.**
- **green text: Comments about results.**

**ALERT INFO (STEPS)**

**First of all, It worked with 2 files about tests and tests2 in notebook/ folder where there are different operations, correct and fail code. After, when the code returned correctly, it was transferred and copied in each module appropriate with its function in utils/ folders. In Addition, sometimes it used excel (not much because it has limits) to confirm the results. Finally, they were imported every function with the operations in main.ipynb file in src/ folder.**

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns #visualisation
import matplotlib.pyplot as plt #visualisation
```

In [2]:

```
import os.path
print(os.path)
#/Users/ariadnapuigventos/Documents/CURSOS/BRIDGE/DS_Ejercicios_Python/BootCamp_TheBridge
/Proyecto_Navidad_Ariadna/src/utils/folders_tb.py
```

```
<module 'posixpath' from '/Library/Frameworks/Python.framework/Versions/3.9/lib/python3.9
/posixpath.py'>
```

**Explain the code organization of this file:**

**It's going to tell about one of two datasets and show the collecting data to understand our hypothesis. Below all these lines, it will show the second datasets with the best insights of Olympic Games Athletes. Finally, It's going to create a new dataframe to try show some similarities to confirm or not hypothesis.**

In [3]:

```
from utils.folders_tb import readcsv
#This is one of two dataframes about Best Marathon Majors in all Sport History.
readcsv()
```

```
     year            winner  gender        country      time marathon
0    2014    Dennis Kimetto    Male          Kenya  02:02:57   Berlin
1    2011    Geoffrey Mutai    Male          Kenya  02:03:02   Boston
2    2016  Kenenisa Bekele     Male       Ethiopia  02:03:03   Berlin
3    2016    Eliud Kipchoge    Male          Kenya  02:03:05   London
4    2013    Wilson Kipsang    Male          Kenya  02:03:23   Berlin
..    ...               ...     ...            ...       ...      ...
531  1966        Bobbi Gibb  Female  United States  03:21:40   Boston
532  1974   Jutta von Haase  Female        Germany  03:22:01   Berlin
533  1969  Sara Mae Berman  Female  United States  03:22:46   Boston
534  1967        Bobbi Gibb  Female  United States  03:27:17   Boston
535  1968        Bobbi Gibb  Female  United States  03:30:00   Boston

[536 rows x 6 columns]
```

In [4]:

```
from utils.mining_data_tb import topandtail, dimention

topandtail()
```

```
   year               winner gender   country      time marathon
0  2014       Dennis Kimetto   Male     Kenya  02:02:57   Berlin
1  2011       Geoffrey Mutai   Male     Kenya  02:03:02   Boston
2  2016      Kenenisa Bekele   Male  Ethiopia  02:03:03   Berlin
3  2016       Eliud Kipchoge   Male     Kenya  02:03:05   London
4  2013       Wilson Kipsang   Male     Kenya  02:03:23   Berlin
5  2017       Eliud Kipchoge   Male     Kenya  02:03:32   Berlin
6  2011      Patrick Musyoki   Male     Kenya  02:03:38   Berlin
7  2013       Dennis Kimetto   Male     Kenya  02:03:45  Chicago
8  2017       Wilson Kipsang   Male     Kenya  02:03:58    Tokyo
9  2008   Haile Gebrselassie   Male  Ethiopia  02:03:59   Berlin
......
     year             winner  gender        country      time marathon
531  1966         Bobbi Gibb  Female  United States  03:21:40   Boston
532  1974    Jutta von Haase  Female        Germany  03:22:01   Berlin
533  1969   Sara Mae Berman  Female  United States  03:22:46   Boston
534  1967         Bobbi Gibb  Female  United States  03:27:17   Boston
535  1968         Bobbi Gibb  Female  United States  03:30:00   Boston
```

In [5]:

```
dimention()
```

```
(536, 6)
number of duplicate rows:  Empty DataFrame
Columns: [year, winner, gender, country, time, marathon]
Index: []
     year             winner  gender        country      time marathon
0    2014     Dennis Kimetto    Male          Kenya  02:02:57   Berlin
1    2011     Geoffrey Mutai    Male          Kenya  02:03:02   Boston
2    2016    Kenenisa Bekele    Male       Ethiopia  02:03:03   Berlin
3    2016     Eliud Kipchoge    Male          Kenya  02:03:05   London
4    2013     Wilson Kipsang    Male          Kenya  02:03:23   Berlin
..    ...                ...     ...            ...       ...      ...
531  1966         Bobbi Gibb  Female  United States  03:21:40   Boston
532  1974    Jutta von Haase  Female        Germany  03:22:01   Berlin
533  1969   Sara Mae Berman  Female  United States  03:22:46   Boston
534  1967         Bobbi Gibb  Female  United States  03:27:17   Boston
535  1968         Bobbi Gibb  Female  United States  03:30:00   Boston

[536 rows x 6 columns]
(536, 6)
```

**ALERT INFO (STEPS)**

**The Dataframe has not any duplicates but there are some values equality. It needs to check what it means because it's possible some majors who has already won more than one marathons, that's why it's going to show using the method values_counts by country and winner.**

In [6]:

```
from utils.mining_data_tb import repite_pais, repetidores

# Effectively, the most country that's repeat is Kenya on the top, below United States an
d Ethiopia.
# It's considering that Kenya started to compete in 1960 and United States since 1896.


repite_pais()
```

```
Kenya             136
United States     104
Ethiopia           51
Germany            36
United Kingdom     35
```

```
Japan             22
Norway            20
Canada            17
Portugal          11
Finland           10
Mexico            10
Russia             8
Poland             8
Brazil             7
Italy              6
Name: country, dtype: int64
```

In [7]:

```
from utils.visualization_tb import piechart_repitepais

#Thanks to this pie chart graphic it's showing that Kenya is the country winner with 136 marathons, it's 25,4% of the total of the competition. In addition, the third country is Ethiopia with aprox 10%, so if it's talking about African Athletes are winners of the com petition for a aprox. 35% of the total pie chart. For curiosity, only there was 1 Spanish athlete who won 2 World Marathons: Berlín 1996 and London 1998 with the best time 2:09:15 and 2:07:57, respectively.

piechart_repitepais()
```

```
0      0 days 02:02:57
1      0 days 02:03:02
2      0 days 02:03:03
3      0 days 02:03:05
4      0 days 02:03:23
             ...
531    0 days 03:21:40
532    0 days 03:22:01
533    0 days 03:22:46
534    0 days 03:27:17
535    0 days 03:30:00
Name: time, Length: 536, dtype: timedelta64[ns]
[136, 104, 51, 36, 35, 22, 20, 17, 11, 10, 10, 8, 8, 7, 6, 5, 5, 5, 5, 4, 3, 3, 3, 2, 2, 2
, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1]
```

In [8]:

```
repetidores()
```

```
Grete Waitz            11
Bill Rodgers            8
Ingrid Kristiansen      8
Uta Pippig              7
Clarence DeMar          7
Paula Radcliffe         7
Eliud Kipchoge          6
Catherine Ndereba       6
Rosa Mota               6
Mary Keitany            6
Khalid Khannouchi       5
Wilson Kipsang          5
Joyce Chepchumba        5
Martin Lel              5
Katrin Dörre-Heinig     4
Name: winner, dtype: int64
```

In [9]:

```
#AQUÍ VA GENDER DATA CLASIFICATION!!!!
```

In [10]:

```
from utils.mining_data_tb import checkingdata
#It wants to know how are the values because it has seen that there is one about time.
```

```
checkingdata()
```

```
year          int64
winner       object
gender       object
country      object
time         object
marathon     object
dtype: object
```

**ALERT INFO (STEPS)**

**It needs to change some data rows after to see time column in dataframe is an object. It will be necessary to change from a object to pd.to_timedelta and after from timedelta to float64 with method "timedelta64[s]" for detecting some outliers and for doing to histogram bins=5.**

In [11]:

```
from utils.mining_data_tb import changetype

#With this fuction it changed from object time column with seconds to use it in boxplot f
or detecting outliers.

changetype()
```

```
0      0 days 02:02:57
1      0 days 02:03:02
2      0 days 02:03:03
3      0 days 02:03:05
4      0 days 02:03:23
             ...
531    0 days 03:21:40
532    0 days 03:22:01
533    0 days 03:22:46
534    0 days 03:27:17
535    0 days 03:30:00
Name: time, Length: 536, dtype: timedelta64[ns]
0       7377.0
1       7382.0
2       7383.0
3       7385.0
4       7403.0
         ...
531    12100.0
532    12121.0
533    12166.0
534    12437.0
535    12600.0
Name: time, Length: 536, dtype: float64
```

In [12]:

```
from utils.visualization_tb import detect_outliers

#2 extrems: the first time was 2:02:57 by Kenian Athlete in Berlin Marathon in 2014; and
the last time was 3:30:00 by United States Athlete in Boston Marathon in 1968. Althought,
25% Marathon majors got a median around 2 hours and 16 minuts and the most majors with 75
% got 2 hours and 46 minuts.

detect_outliers()
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
7783.0
8856.25
1073.25
```

**ALERT INFO (STEPS)**

**It's showing the histogram of each column. In this case, every columns fo the World Marathon Majors Dataframe, except Year, one hand, has been changed by astype "Category" because they were object types;**

**and the other hand, Time Column has been changed from pd.to_timedelta to timedelta64[s] because It needs in seconds to showing in histogram. It shows how the ranges are different between them.**

In [13]:

```
from utils.visualization_tb import histogram_time, histogram_year_time

#There are more participation years later than the begginers of competition when only Can
ada and United States were winners for a long time ago consecutively.

histogram_time()
histogram_year_time()
```

```
[[<AxesSubplot:title={'center':'year'}>
  <AxesSubplot:title={'center':'time'}>]]
```

In [14]:

```
from utils.visualization_tb import histogram_gender

#Only there is a different of 13% (70 World Marathons) more won them by Male athletes wit
h 56,5% (303 WM) than Female with 43,4% (233 WM). It's a great appreciation because it no
tices that the first woman athlete who could compete was in 1967 (71 years after than men
).

histogram_gender()
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

In [15]:

```
from utils.visualization_tb import histogram_country

#It is not showing the real situation with bins=5, below these lines it's changed to a hi
stogram with bins=37 (total countries).

histogram_country()
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
Get better another argument to see almost it
```

In [16]:

```
from utils.visualization_tb import histogram_countryby37bins

#This graphic is showing how there are 2 countries stand out (Kenia and Ethiopia) versus
of the rest countries.

histogram_countryby37bins()
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

**ALERT INFO (STEPS)**

**For showing a correlation Matrix it has needed to tell different steps: 1.- The columns in the dataframe were object, so it needed to change the type to integer to show correlation matrix. 2.- But, it was not possible from object to category or object to integer, so it to do a Encode each column. 3.- They created 3 new columns with encoding values, respectively.**

In [17]:

```python
from sklearn.preprocessing import LabelEncoder
```

In [18]:

```python
from utils.visualization_tb import matrix

#To show the correlation Matrix with columns dataframe 1.

matrix()
```

```
0      0
1      0
2      0
3      0
4      0
      ..
531    1
532    1
533    1
534    1
535    1
Name: gender_1, Length: 536, dtype: category
Categories (2, int64): [1, 0]
0       7377.0
1       7382.0
2       7383.0
3       7385.0
4       7403.0
         ...
531    12100.0
532    12121.0
533    12166.0
534    12437.0
535    12600.0
Name: time, Length: 536, dtype: float64
0      17
1      17
2       8
3      17
4      17
       ..
531    35
532    10
533    35
534    35
535    35
Name: encoded_country, Length: 536, dtype: int64
0      0
1      1
2      0
3      3
4      0
      ..
531    1
532    0
533    1
534    1
535    1
Name: encoded_marathon, Length: 536, dtype: int64
                      year      time  encoded_country  encoded_marathon
year              1.000000 -0.427552        -0.265384          0.276483
time             -0.427552  1.000000         0.204407         -0.148728
encoded_country  -0.265384  0.204407         1.000000          0.098366
encoded_marathon  0.276483 -0.148728         0.098366          1.000000
```

In [19]:

```python
from utils.visualization_tb import matrix_1960
```

```
#To show the correlation Matrix from 1960 when Kenya was the first time compete in BWMM,
almost in 1967 was the first woman who started to compete there as well.

matrix_1960()
```

```
0      0
1      0
2      0
3      0
4      0
      ..
531    1
532    1
533    1
534    1
535    1
Name: gender_1, Length: 536, dtype: category
Categories (2, int64): [1, 0]
0       7377.0
1       7382.0
2       7383.0
3       7385.0
4       7403.0
        ...
531    12100.0
532    12121.0
533    12166.0
534    12437.0
535    12600.0
Name: time, Length: 536, dtype: float64
0      17
1      17
2       8
3      17
4      17
       ..
531    35
532    10
533    35
534    35
535    35
Name: encoded_country, Length: 536, dtype: int64
0      0
1      1
2      0
3      3
4      0
      ..
531    1
532    0
533    1
534    1
535    1
Name: encoded_marathon, Length: 536, dtype: int64
     year             winner  gender        country     time marathon gender_1  \
63   2005         Martin Lel    Male          Kenya   7655.0   London        0
64   2006   Robert Cheruiyot    Male          Kenya   7655.0  Chicago        0
65   2011    Hailu Mekonnen    Male       Ethiopia   7655.0    Tokyo        0
66   2012   Michael Kipyego    Male          Kenya   7657.0    Tokyo        0
67   1997      Elijah Lagat    Male          Kenya   7661.0   Berlin        0
..    ...               ...     ...            ...      ...      ...      ...
531  1966         Bobbi Gibb  Female  United States  12100.0   Boston        1
532  1974   Jutta von Haase  Female        Germany  12121.0   Berlin        1
533  1969   Sara Mae Berman  Female  United States  12166.0   Boston        1
534  1967         Bobbi Gibb  Female  United States  12437.0   Boston        1
535  1968         Bobbi Gibb  Female  United States  12600.0   Boston        1

     encoded_country  encoded_marathon
63                17                 3
64                17                 2
65                 8                 5
```

```
66              17              5
67              17              0
..             ...            ...
531             35              1
532             10              0
533             35              1
534             35              1
535             35              1

[473 rows x 9 columns]
                      year      time  encoded_country  encoded_marathon
year              1.000000 -0.357453        -0.244556          0.299996
time             -0.357453  1.000000         0.177196         -0.177691
encoded_country  -0.244556  0.177196         1.000000          0.092922
encoded_marathon  0.299996 -0.177691         0.092922          1.000000
```

In [20]:

```
from utils.folders_tb import readbdd

readbdd(url="/Users/ariadnapuigventos/Documents/CURSOS/BRIDGE/DS_Ejercicios_Python/BootCamp_TheBridge/Proyecto_Navidad_Ariadna/documentation/altitud_countries.csv")
```

Out[20]:

| | country | latitude | longitude | name | Unnamed: 4 | Unnamed: 5 | Unnamed: 6 |
|---|---|---|---|---|---|---|---|
| **0** | AD | 42.546245 | 1.601554 | Andorra | NaN | NaN | NaN |
| **1** | AE | 23.424076 | 53.847818 | United Arab Emirates | NaN | NaN | NaN |
| **2** | AF | 33.939110 | 67.709953 | Afghanistan | NaN | NaN | NaN |
| **3** | AG | 17.060816 | -61.796428 | Antigua and Barbuda | NaN | NaN | NaN |
| **4** | AI | 18.220554 | -63.068615 | Anguilla | NaN | NaN | NaN |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **240** | YE | 15.552727 | 48.516388 | Yemen | NaN | NaN | NaN |
| **241** | YT | -12.827500 | 45.166244 | Mayotte | NaN | NaN | NaN |
| **242** | ZA | -30.559482 | 22.937506 | South Africa | NaN | NaN | NaN |
| **243** | ZM | -13.133897 | 27.849332 | Zambia | NaN | NaN | NaN |
| **244** | ZW | -19.015438 | 29.154857 | Zimbabwe | NaN | NaN | NaN |

**245 rows × 7 columns**

In [21]:

```
from utils.mining_data_tb import droppingcolumns

droppingcolumns()
```

```
      latitude  longitude               country
0    42.546245   1.601554               Andorra
1    23.424076  53.847818  United Arab Emirates
2    33.939110  67.709953           Afghanistan
3    17.060816 -61.796428   Antigua and Barbuda
4    18.220554 -63.068615              Anguilla
..         ...        ...                   ...
240  15.552727  48.516388                 Yemen
241 -12.827500  45.166244               Mayotte
242 -30.559482  22.937506          South Africa
243 -13.133897  27.849332                Zambia
244 -19.015438  29.154857              Zimbabwe

[245 rows x 3 columns]
```

In [ ]:

```

```