C3_M3 SportsStats Beyond Desc Stats

May 28, 2025

1 Beyond Descriptive Statistics

1.1 Step 1: Perform Initial Statistics

1.1.1 Hypotheses

1. Athletes have an advantage when their host country is also their home country.
2. Athletes who compete in multiple events at the same Games are more likely to medal.
3. There a correlation between physical attributes and winning medals.

[1]:
```
# Import all necessary libraries library

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Import the SQL library

from pandasql import sqldf
pysqldf = lambda q: sqldf(q, globals())
```

[2]:
```
# Import the datasets

events = pd.read_csv('athlete_events.csv')
regions = pd.read_csv('noc_regions.csv')
```

1.2 Exploring Hypothesis 1

[5]:
```
# Match host City to its corresponding NOC

host_city_noc_map_data = {
    'Host_City': ['London', 'Athina', 'Sydney', 'Atlanta', 'Rio de Janeiro',␣
 ↪'Beijing', 'Barcelona', 'Los Angeles',
                  'Seoul', 'Munich', 'Montreal', 'Mexico City', 'Helsinki',␣
 ↪'Roma', 'Tokyo', 'Moskva', 'Paris',
```

```
                  'Berlin', 'Amsterdam', 'Sochi', 'Melbourne', 'Vancouver',␣
 ↪'Torino', 'Stockholm', 'Antwerpen',
                  'Salt Lake City', 'Innsbruck', 'Nagano', 'Albertville',␣
 ↪'Lillehammer', 'Calgary', 'Sarajevo',
                  'Lake Placid', 'Grenoble', 'Sankt Moritz', 'Sapporo',␣
 ↪"Cortina d'Ampezzo", 'St. Louis',
                  'Squaw Valley', 'Oslo', 'Garmisch-Partenkirchen', 'Chamonix'],
    'Host_NOC': ['GBR', 'GRE', 'ANZ', 'USA', 'BRA', 'CHN', 'ESP', 'USA',
                 'KOR', 'GER', 'CAN', 'MEX', 'FIN', 'ITA', 'JPN', 'RUS', 'FRA',
                 'GER', 'NED', 'RUS', 'ANZ', 'CAN', 'ITA', 'SWE', 'BEL',
                 'USA', 'AUT', 'JPN', 'FRA', 'NOR', 'CAN', 'BIH',
                 'USA', 'FRA', 'SWZ', 'JPN', 'ITA', 'USA',
                 'USA', 'NOR', 'GER', 'FRA']
}
host_city_noc_map = pd.DataFrame(host_city_noc_map_data)


# Return records where the host City is in an athlete's NOC

athletes_in_host_country = pysqldf('SELECT e.*, h.Host_NOC FROM events AS e␣
 ↪JOIN host_city_noc_map AS h ON e.City = h.Host_City WHERE e.NOC = h.
 ↪Host_NOC')

print("\nAthletes whose Home NOC matches the Host City's NOC:")
athletes_in_host_country
```

```
Athletes whose Home NOC matches the Host City's NOC:
```

[5]:

	ID	Name	Sex	Age	Height	Weight	\
0	10	Einar Ferdinand "Einari" Aalto	M	26.0	NaN	NaN	
1	17	Paavo Johannes Aaltonen	M	32.0	175.0	64.0	
2	17	Paavo Johannes Aaltonen	M	32.0	175.0	64.0	
3	17	Paavo Johannes Aaltonen	M	32.0	175.0	64.0	
4	17	Paavo Johannes Aaltonen	M	32.0	175.0	64.0	
…	…	…	..	…	…	…	
18511	135485	Stepan Olegovich Zuyev	M	25.0	189.0	90.0	
18512	135485	Stepan Olegovich Zuyev	M	25.0	189.0	90.0	
18513	135485	Stepan Olegovich Zuyev	M	25.0	189.0	90.0	
18514	135539	Marius Edmund Zwiller	M	18.0	NaN	NaN	
18515	135560	Stavroula Zygouri	F	36.0	171.0	63.0	

	Team	NOC	Games	Year	Season	City	Sport	\
0	Finland	FIN	1952 Summer	1952	Summer	Helsinki	Swimming	
1	Finland	FIN	1952 Summer	1952	Summer	Helsinki	Gymnastics	
2	Finland	FIN	1952 Summer	1952	Summer	Helsinki	Gymnastics	
3	Finland	FIN	1952 Summer	1952	Summer	Helsinki	Gymnastics	

```
4       Finland  FIN  1952 Summer  1952  Summer  Helsinki     Gymnastics
…         …  …          …   …      …       …           …
18511    Russia  RUS  2014 Winter  2014  Winter     Sochi  Alpine Skiing
18512    Russia  RUS  2014 Winter  2014  Winter     Sochi  Alpine Skiing
18513    Russia  RUS  2014 Winter  2014  Winter     Sochi  Alpine Skiing
18514    France  FRA  1924 Summer  1924  Summer     Paris       Swimming
18515    Greece  GRE  2004 Summer  2004  Summer    Athina      Wrestling

                                          Event   Medal Host_NOC
0            Swimming Men's 400 metres Freestyle    None      FIN
1         Gymnastics Men's Individual All-Around    None      FIN
2               Gymnastics Men's Team All-Around  Bronze      FIN
3                Gymnastics Men's Floor Exercise    None      FIN
4                   Gymnastics Men's Horse Vault    None      FIN
…                                             …     …       …
18511                Alpine Skiing Men's Super G    None      RUS
18512           Alpine Skiing Men's Giant Slalom    None      RUS
18513                 Alpine Skiing Men's Slalom    None      RUS
18514    Swimming Men's 200 metres Breaststroke    None      FRA
18515  Wrestling Women's Middleweight, Freestyle    None      GRE

[18516 rows x 16 columns]
```

[6]:
```
# How many medals have been won in total?

pysqldf('SELECT COUNT(Medal) AS medals FROM events WHERE Medal IS NOT NULL')
```

[6]:
```
   medals
0   39783
```

[7]:
```
home_medal_pct = (18516/39783)*100
home_medal_pct
```

[7]:
```
46.542493024658775
```

We see that 18,516 of the 39,783 medals were won by athletes in their home country, or 46.5%. This is a significant correlation given the number of countries that participate in the Olympic Games.

However, this result could be skewed due to the number of times a specific country has hosted an Olympic Games.

[5]:
```
# Match host City to its corresponding NOC

host_city_noc_map_data = {
    'Host_City': ['London', 'Athina', 'Sydney', 'Atlanta', 'Rio de Janeiro',␣
 ↪'Beijing', 'Barcelona', 'Los Angeles',
                  'Seoul', 'Munich', 'Montreal', 'Mexico City', 'Helsinki',␣
 ↪'Roma', 'Tokyo', 'Moskva', 'Paris',
```

```
                  'Berlin', 'Amsterdam', 'Sochi', 'Melbourne', 'Vancouver',␣
 ↪'Torino', 'Stockholm', 'Antwerpen',
                  'Salt Lake City', 'Innsbruck', 'Nagano', 'Albertville',␣
 ↪'Lillehammer', 'Calgary', 'Sarajevo',
                  'Lake Placid', 'Grenoble', 'Sankt Moritz', 'Sapporo',␣
 ↪"Cortina d'Ampezzo", 'St. Louis',
                  'Squaw Valley', 'Oslo', 'Garmisch-Partenkirchen', 'Chamonix'],
    'Host_NOC': ['GBR', 'GRE', 'ANZ', 'USA', 'BRA', 'CHN', 'ESP', 'USA',
                 'KOR', 'GER', 'CAN', 'MEX', 'FIN', 'ITA', 'JPN', 'RUS', 'FRA',
                 'GER', 'NED', 'RUS', 'ANZ', 'CAN', 'ITA', 'SWE', 'BEL',
                 'USA', 'AUT', 'JPN', 'FRA', 'NOR', 'CAN', 'BIH',
                 'USA', 'FRA', 'SWZ', 'JPN', 'ITA', 'USA',
                 'USA', 'NOR', 'GER', 'FRA']
}
host_city_noc_map = pd.DataFrame(host_city_noc_map_data)


# Return how many times an NOC corresponds to a distinct City

host_country = pysqldf('SELECT h.Host_NOC, COUNT(DISTINCT e.City) FROM events␣
 ↪AS e JOIN host_city_noc_map AS h ON e.City = h.Host_City WHERE e.NOC = h.
 ↪Host_NOC GROUP BY h.Host_NOC')

print("\nTimes an NOC Hosted :")
host_country
```

```
Times an NOC Hosted :
```

[5]:

	Host_NOC	COUNT(DISTINCT e.City)
0	AUT	1
1	BEL	1
2	BRA	1
3	CAN	3
4	CHN	1
5	ESP	1
6	FIN	1
7	FRA	4
8	GBR	1
9	GER	2
10	GRE	1
11	ITA	3
12	JPN	3
13	KOR	1
14	MEX	1
15	NED	1
16	NOR	2

17	RUS	1
18	SWE	1
19	USA	6

Add to this the number of athletes each NOC sent to the Games each time it hosted.

[10]:
```
pysqldf('SELECT h.Host_NOC, COUNT(DISTINCT e.City) AS times_hosted,␣
 ↪COUNT(DISTINCT CASE WHEN e.NOC = h.Host_NOC THEN e.Name ELSE NULL END) AS␣
 ↪home_athletes, COUNT(DISTINCT e.Name) AS total_athletes, CAST(COUNT(DISTINCT␣
 ↪CASE WHEN e.NOC = h.Host_NOC THEN e.Name ELSE NULL END) AS REAL) * 100.0 /␣
 ↪CASE WHEN COUNT(DISTINCT e.Name) = 0 THEN 1 ELSE COUNT(DISTINCT e.Name) END␣
 ↪AS percent_home FROM events AS e JOIN host_city_noc_map AS h ON e.City = h.
 ↪Host_City GROUP BY h.Host_NOC ORDER BY h.Host_NOC')
```

[10]:

	Host_NOC	times_hosted	home_athletes	total_athletes	percent_home
0	ANZ	2	0	13827	0.000000
1	AUT	1	158	2208	7.155797
2	BEL	1	336	2675	12.560748
3	BIH	1	0	1272	0.000000
4	BRA	1	462	11174	4.134598
5	CAN	3	696	10020	6.946108
6	CHN	1	583	10880	5.358456
7	ESP	1	422	9380	4.498934
8	FIN	1	258	4931	5.232204
9	FRA	4	1342	7738	17.342983
10	GBR	1	1672	16924	9.879461
11	GER	3	488	12239	3.987254
12	GRE	1	828	11536	7.177531
13	ITA	3	524	8662	6.049411
14	JPN	3	568	8317	6.829386
15	KOR	1	399	8443	4.725808
16	MEX	1	274	5552	4.935159
17	NED	1	266	3246	8.194701
18	NOR	2	160	2432	6.578947
19	RUS	2	213	7993	2.664832
20	SWE	1	453	2567	17.647059
21	SWZ	1	0	1126	0.000000
22	USA	6	2594	23792	10.902824

1.3 Exploring Hypothesis 2

[9]:
```
# Find athletes who were in multiple events in the same year

mea = pysqldf('SELECT Name, Year, COUNT(*) AS event_count FROM events GROUP BY␣
 ↪Name, Year HAVING COUNT(*) > 1')
mea
```

[9]:

	Name	Year	event_count
0	Eleonora Margarida Josephina Scmitt	1948	2
1	Luis ngel Fernando de los Santos Grossi	1952	4
2	Th Ngn Thng	2008	5
3	Th Ngn Thng	2012	2
4	A. Abdul Razzak	1960	2
…	…	…	…
47537	yvind Berg	1994	3
47538	yvind Tveter	1980	2
47539	zcan Ediz	1992	2
47540	zdemir Akbal	2000	2
47541	zer Atei	1968	3

[47542 rows x 3 columns]

[10]:
```
# Which ones medaled?

mea_medals = pysqldf('SELECT Name, Year, COUNT(*) AS event_count, Medal FROM␣
↪events WHERE Medal IS NOT NULL GROUP BY Name, Year, Medal HAVING COUNT(*) >␣
↪1')
mea_medals
```

[10]:

	Name	Year	event_count	Medal
0	Aagje "Ada" Kok (-van der Linden)	1964	2	Silver
1	Aaron Wells Peirsol	2004	3	Gold
2	Aaron Wells Peirsol	2008	2	Gold
3	Abelardo Olivier	1920	2	Gold
4	Adam Henryk Maysz	2010	2	Silver
…	…	…	…	…
1856	scar Cristi Gallo	1952	2	Silver
1857	sten stensen	1920	2	Bronze
1858	sten stensen	1920	2	Silver
1859	tienne Nol Henri Vandernotte	1936	2	Bronze
1860	va Grard-Novk	1952	2	Silver

[1861 rows x 4 columns]

[12]:
```
# What percentage of them medaled?

mea_medals_pct = (1861/47542)*100
mea_medals_pct
```

[12]: 3.9144335534895465

[18]:
```
# Compare that to the percentage of all athletes who medaled

athletes = pysqldf('SELECT COUNT(*) AS athletes FROM events')
```

```
medals = pysqldf('SELECT COUNT(Medal) AS medals FROM events WHERE Medal IS NOT␣
↪NULL')
percent = (39783/271116)*100

print('Total Athletes: ', athletes)
print('Medals: ', medals)
print('Medal %: ', percent)
```

```
Total Athletes:     athletes
0    271116
Medals:     medals
0   39783
Medal %:  14.673792767671404
```

Athletes that compete in multiple events account for just 3.9% of medals, whereas the overall percentage of athletes that account for all medals is 14.7%. There seems to be a negative correlation between competing in multiple events and winning medals.

1.4 Exploring Hypothesis 3

To determine whether a correlation exists between physical attributes and winning medals, it is logical to look at specific Sports and Events rather than overall averages because the range of physical attributes is wide when considering athletes of all types together.

[14]:
```
# Athletics as an example

# Find the average age, height, and weight for all Athletics athletes

athletics_averages = pysqldf('SELECT AVG(Age), AVG(Height), AVG(Weight) FROM␣
↪events WHERE Sport = "Athletics" ')
print('Athlete Averages: ', athletics_averages)

# Compare that to the average age, height, and weight for all Athletics␣
↪athletes who medaled

medalist_athletics_averages = pysqldf('SELECT AVG(Age), AVG(Height),␣
↪AVG(Weight) FROM events WHERE Medal IS NOT NULL AND Sport = "Athletics" ')
print('Medalist Average: ', medalist_athletics_averages)
```

```
Athlete Averages:      AVG(Age)  AVG(Height)  AVG(Weight)
0  25.161223   176.256268    69.249287
Medalist Average:      AVG(Age)  AVG(Height)  AVG(Weight)
0  25.020532   177.623978    71.506294
```

There are very slight differences between the overall Athletics averages and the medalist averages. While the medalist average age is slightly younger than the overall average, the average height and weight are both higher than the overall. Further study is needed to know if this is signficant.

[15]:
```
# Swimming as an example

# Find the average age, height, and weight for all Swimming athletes

swimming_averages = pysqldf('SELECT AVG(Age), AVG(Height), AVG(Weight) FROM␣
 ↪events WHERE Sport = "Swimming" ')
print('Swimming Averages: ', swimming_averages)

# Compare that to the average age, height, and weight for all Swimming athletes␣
 ↪who medaled

medalist_swimming_averages = pysqldf('SELECT AVG(Age), AVG(Height), AVG(Weight)␣
 ↪FROM events WHERE Medal IS NOT NULL AND Sport = "Swimming" ')
print('Medalist Average: ', medalist_swimming_averages)
```

```
Swimming Averages:      AVG(Age)  AVG(Height)  AVG(Weight)
0  20.566803   178.562454    70.588492
Medalist Average:      AVG(Age)  AVG(Height)  AVG(Weight)
0  20.923356   180.923197    73.251005
```

There are more noticeable differences between the overall Swimming averages and the medalist averages. Medalist are slightly older, but are markedly taller and heavier.

[16]:
```
# Gymnastics as an example

# Find the average age, height, and weight for all Gymnastics athletes

gymnastics_averages = pysqldf('SELECT AVG(Age), AVG(Height), AVG(Weight) FROM␣
 ↪events WHERE Sport = "Gymnastics" ')
print('Gymnastics Averages: ', gymnastics_averages)

# Compare that to the average age, height, and weight for all Gymnastics␣
 ↪athletes who medaled

medalist_gymnastics_averages = pysqldf('SELECT AVG(Age), AVG(Height),␣
 ↪AVG(Weight) FROM events WHERE Medal IS NOT NULL AND Sport = "Gymnastics" ')
print('Medalist Average: ', medalist_gymnastics_averages)
```

```
Gymnastics Averages:      AVG(Age)  AVG(Height)  AVG(Weight)
0  22.733038    162.93602    56.916553
Medalist Average:      AVG(Age)  AVG(Height)  AVG(Weight)
0  23.406493    161.57686    55.083763
```

Again, there are slight differences between the overall Gymnastics averages and the medalist averages. This time, however, medlists are slightly older, but are both shorter and lighter.

Since results based on averages were inconclusive regarding their statistical significance, a new metric that tracks how these averages have changed over time could be useful.

[17]:
```
# Create a new metric: Medalist age, height, and weight averages over time. Use␣
↪Gymnastics as an example.

medalist_gymnastics_avg_time = pysqldf('SELECT Year, AVG(Age), AVG(Height),␣
↪AVG(Weight) FROM events WHERE Medal IS NOT NULL AND Sport = "Gymnastics"␣
↪GROUP BY Year')
medalist_gymnastics_avg_time
```

[17]:

	Year	AVG(Age)	AVG(Height)	AVG(Weight)
0	1896	25.322581	161.000000	65.000000
1	1900	25.000000	NaN	NaN
2	1904	26.400000	171.882353	71.000000
3	1906	24.547619	169.000000	NaN
4	1908	22.608247	171.285714	69.400000
5	1912	24.525253	172.000000	73.000000
6	1920	26.335329	172.500000	64.666667
7	1924	28.272727	165.000000	NaN
8	1928	23.685714	169.500000	64.000000
9	1932	24.022222	168.105263	62.000000
10	1936	25.571429	169.500000	61.000000
11	1948	28.027778	173.666667	63.000000
12	1952	26.400000	165.600000	61.066667
13	1956	25.575758	162.913793	58.689655
14	1960	25.534247	163.126761	59.718310
15	1964	25.369863	162.684932	58.616438
16	1968	22.472222	163.791667	57.090278
17	1972	21.875000	163.125000	55.041667
18	1976	21.081081	162.608108	54.621622
19	1980	20.333333	162.640000	53.613333
20	1984	20.786667	158.266667	52.053333
21	1988	19.413333	158.693333	52.146667
22	1992	18.701299	159.350000	54.566667
23	1996	20.087500	159.521739	53.434783
24	2000	20.694444	160.414286	53.642857
25	2004	21.652778	159.680556	53.277778
26	2008	21.083333	157.777778	51.070423
27	2012	20.848485	161.121212	54.935484
28	2016	21.787879	159.439394	53.727273

Looking at the data, it is clear average Age, Height, and Weight have all changed significantly in the 120 years we are analyzing.

Average age: Peaked in 1924 at 28 years old. Under 22 years old since 1972.

Average height: Peaked in 1948 at approximately 174cm. Under 164cm since 1956.

Average weight: Peaked in 1912 at 73kg. Under 55kg since 1976.