PYTHON PROJECT

SUMMER OLYMPICS

JOVAN TRAJCESKI

SUMMARY

DATA SETS (1896 TO 2008)

1. Olympic Editions
2. IOC Country Codes
3. Medalist

QUESTION TO ANSWER

Does a host country win more medals?

TOOLS USED




DATAFRAME SUMMARY (1896 TO 2008)

	Edition	Grand Total	City	Country
0	1896	151	Athens	Greece
1	1900	512	Paris	France
2	1904	470	St. Louis	United States
3	1908	804	London	United Kingdom
4	1912	885	Stockholm	Sweden
5	1920	1298	Antwerp	Belgium
6	1924	884	Paris	France
7	1928	710	Amsterdam	Netherlands
8	1932	615	Los Angeles	United States
9	1936	875	Berlin	Germany

OLYMPIC EDITIONS

	Country	NOC
0	Afghanistan	AFG
1	Albania	ALB
2	Algeria	ALG
3	American Samoa*	ASA
4	Andorra	AND
5	Angola	ANG
6	Antigua and Barbuda	ANT
7	Argentina	ARG
8	Armenia	ARM
9	Aruba*	ARU

IOC COUNTRY CODES

	Athlete	NOC	Medal	Edition
0	HAJOS, Alfred	HUN	Gold	1896
1	HERSCHMANN, Otto	AUT	Silver	1896
2	DRIVAS, Dimitrios	GRE	Bronze	1896
3	MALOKINIS, Ioannis	GRE	Gold	1896
4	CHASAPIS, Spiridon	GRE	Silver	1896
5	CHOROPHAS, Efstathios	GRE	Bronze	1896
6	HAJOS, Alfred	HUN	Gold	1896
7	ANDREOU, Joannis	GRE	Silver	1896
8	CHOROPHAS, Efstathios	GRE	Bronze	1896
9	NEUMANN, Paul	AUT	Gold	1896

ALL MEDALIST

Quantifying performance

- Counting medals by country/edition in a pivot table
- Computing fraction of medals per Olympic edition
- Computing percentage change in fraction of medals won

COUNTING MEDALS BY COUNTRY/EDITION IN A PIVOT TABLE

```
# Construct the pivot_table: medal_counts
medal_counts = medals.pivot_table(index='Edition', values='Athlete', columns='NOC', aggfunc='count')

# Print the first & last 5 rows of medal_counts
medal_counts.head()
```

NOC	AFG	AHO	ALG	ANZ	ARG	ARM	AUS	AUT	AZE	BAH
Edition										
1992	NaN	NaN	2.0	NaN	2.0	NaN	57.0	6.0	NaN	1.0
1996	NaN	NaN	3.0	NaN	20.0	2.0	132.0	3.0	1.0	5.0
2000	NaN	NaN	5.0	NaN	20.0	1.0	183.0	4.0	3.0	6.0
2004	NaN	NaN	NaN	NaN	47.0	NaN	157.0	8.0	5.0	2.0
2008	1.0	NaN	2.0	NaN	51.0	6.0	149.0	3.0	7.0	5.0

5 rows × 138 columns

As you can see, the pivot table DataFrame has mostly NaN entries
(because most countries do not win any medals in a given Olympic edition).

COMPUTING FRACTION OF MEDALS PER OLYMPIC EDITION

```
# Extract the 'Grand Total' column from totals and assign the result back to totals.
totals = totals['Grand Total']


# Divide the DataFrame medal_counts by totals along each row.
fractions = medal_counts.divide(totals, axis='rows')

# Print first 5 rows of fractions
fractions.head()
```

NOC	AFG	AHO	ALG	ANZ	ARG	ARM	AUS	AUT
Edition								
1896	NaN	NaN	NaN	NaN	NaN	NaN	0.013245	0.033113
1900	NaN	NaN	NaN	NaN	NaN	NaN	0.009766	0.011719
1904	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.002128
1908	NaN	NaN	NaN	0.023632	NaN	NaN	NaN	0.001244
1912	NaN	NaN	NaN	0.011299	NaN	NaN	NaN	0.015819

This gives you a normalized indication of each country's performance in each edition.

COMPUTING % CHANGE IN FRACTION OF MEDALS WON

```
# Apply the expanding mean: mean_fractions
mean_fractions = fractions.expanding().mean()

# Compute the percentage change: fractions_change
fractions_change = mean_fractions.pct_change() * 100

# Reset the index of fractions_change: fractions_change
# This will make 'Edition' an ordinary column
fractions_change = fractions_change.reset_index()
fractions_change.head()
```

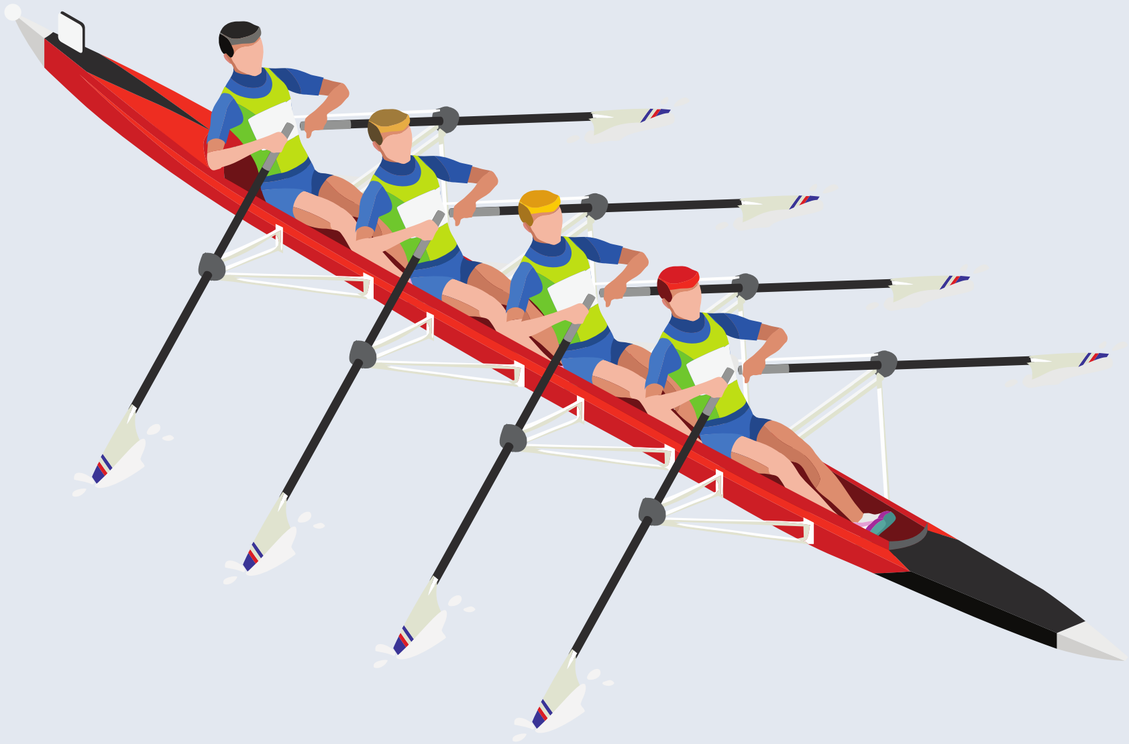
NOC	Edition	AFG	AHO	ALG	ANZ	ARG	ARM	AUS
0	1896	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	1900	NaN	NaN	NaN	NaN	NaN	NaN	-13.134766
2	1904	NaN	NaN	NaN	NaN	NaN	NaN	0.000000
3	1908	NaN	NaN	NaN	NaN	NaN	NaN	0.000000
4	1912	NaN	NaN	NaN	-26.092774	NaN	NaN	0.000000

5 rows × 139 columns

To see if there is a host country advantage, you first want to see how the fraction of medals won changes from edition to edition.The expanding mean provides a way to see this down each column. It is the value of the mean with all the data available up to that point in time.

RECAP

```
Summary -> so far we have:

1. editions           # 'Summer Olympic medalists 1896 to 2008 - EDITIONS.tsv'
2. ioc_codes          # 'Summer Olympic medalists 1896 to 2008 - IOC COUNTRY CODES.csv'
3. medals             # 'Summer Olympic medalists 1896 to 2008 - ALL MEDALISTS.tsv'
4. medal_counts       # count of medals by country
5. fractions          # computing fraction of medals per Olympic edition
6. mean_fractions     # mean of 'fractions' to see there is a host country advantage,
                      # you first want to see how the fraction of medals won changes from edition to edition.
7. fractions_change   # % change of each country fraction of olympic medals won
```



Reshaping and plotting

- Building hosts DataFrame
- Reshaping for analysis
- Merging to compute influence
- Plotting influence of host country

BUILDING HOSTS DATAFRAME

```
# Left join editions and ioc_codes: hosts
hosts = pd.merge(editions, ioc_codes, how='left')

# Extract relevant columns and set index: hosts
hosts = hosts[['Edition','NOC']].set_index("Edition")

# Fix missing 'NOC' values of hosts
print(hosts.loc[hosts.NOC.isnull()])
hosts.loc[1972, 'NOC'] = 'FRG'
hosts.loc[1980, 'NOC'] = 'URS'
hosts.loc[1988, 'NOC'] = 'KOR'

# Reset Index of hosts: hosts
hosts = hosts.reset_index()
```

```
1 hosts.head()
```

	Edition	NOC
0	1896	GRE
1	1900	FRA
2	1904	USA
3	1908	GBR
4	1912	SWE

> Prepare a DataFrame hosts by left joining editions and ioc_codes.
> Clean up NaN values and reset index.
> You now have a DataFrame consisting of all the host.

RESHAPING FOR ANALYSIS

```
# lets see fraction_change first
fractions_change.head()

# Reshape fractions_change: reshaped
reshaped = pd.melt(fractions_change, id_vars='Edition', value_name='Change')
reshaped.head()

# Print reshaped.shape and fractions_change.shape
print(reshaped.shape, fractions_change.shape)

# Extract rows from reshaped where 'NOC' == 'CHN': chn
chn = reshaped[(reshaped['NOC'])=='CHN']
chn.tail()
```

China fared significantly better in 2008 (i.e., when China was the host country).

	Edition	NOC	Change
567	1992	CHN	4.240630
568	1996	CHN	7.860247
569	2000	CHN	-3.851278
570	2004	CHN	0.128863
571	2008	CHN	13.251332

> Reshape the fractions_change DataFrame for later analysis.
> Initially, fractions_change is a wide DataFrame of 26 rows and 139 columns.
> On reshaping with pd.melt(), the result is a tall DataFrame with 3588 rows and 3 columns that summarizes the fractional change in the expanding mean of the percentage of medals won for each country in blocks.

MERGING TO COMPUTE INFLUENCE

```
# Merge reshaped and hosts: merged
merged = pd.merge(reshaped,hosts,how='inner')
merged.head()

# Set Index of merged and sort it: influence
influence = merged.set_index('Edition').sort_index()
influence.head()
```

Edition	NOC	Change
1896	GRE	NaN
1900	FRA	198.002486
1904	USA	199.651245
1908	GBR	134.489218
1912	SWE	71.896226

> Merge the two DataFrames and tidy the result.

> The end result is a DataFrame summarizing the fractional change in the expanding mean of the percentage of medals won for the host country in each Olympic edition.

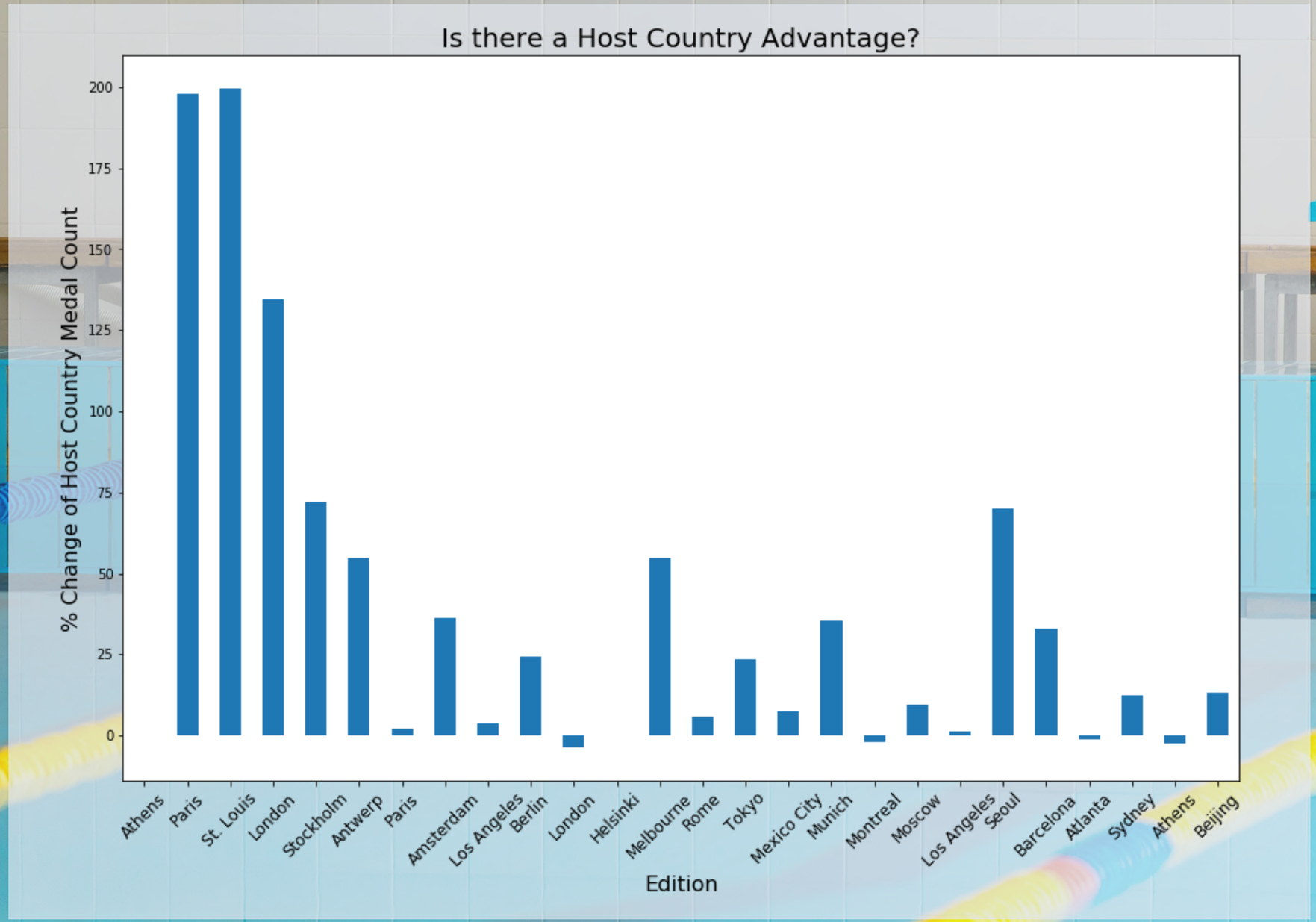
RECAP

```
Summary -> so far we have:

1. editions           # 'Summer Olympic medalists 1896 to 2008 - EDITIONS.tsv'
2. ioc_codes          # 'Summer Olympic medalists 1896 to 2008 - IOC COUNTRY CODES.csv'
3. medals             # 'Summer Olympic medalists 1896 to 2008 - ALL MEDALISTS.tsv'
4. medal_counts       # count of medals by country
5. fractions          # computing fraction of medals per Olympic edition
6. mean_fractions     # mean of 'fractions' to see there is a host country advantage,
                      # you first want to see how the fraction of medals won changes from edition to edition.
7. fractions_change   # % change of each country fraction of olympic medals won per each Olympics (edition)
8. hosts              # DataFrame consisting of all the host
9. reshaped           # reshape the fractions_change DataFrame (#7) for later analysis
10. merged            # merged DataFrame between reshaped and hosts on inner join
11. influence         # indexed 'merged' on 'Edition' column and sorted
```

PLOTTING INFLUENCE OF HOST COUNTRY

```
# Extract influence['Change']: change
change = influence['Change']

# Increase plot size
fig, ax = plt.subplots(figsize=(15, 10))

# Make bar plot of change: ax
ax = change.plot(kind='bar', rot=45 )

# Customize the plot to improve readability
ax.set_ylabel("% Change of Host Country Medal Count", fontsize=16)
ax.set_xlabel("Edition",fontsize=16)
ax.set_title("Is there a Host Country Advantage?", fontsize=20)
ax.set_xticklabels(editions['City'], fontsize=12)

# Display the plot
plt.show()
```

PLOTTING INFLUENCE OF HOST COUNTRY



Jovan Trajceski

https://www.linkedin.com/in/trajceski/

#100DaysofCode

#100DaysofLearning


HAVE A
GREAT DAY !