Medals by the Numbers: Uncovering the Secrets of Olympic Performance

Group 5: Thaissa Champagne, Thomas Gartley, Adam Loux, and Noah Stevens

In our project, we sought to find interesting trends, unexpected outliers and unique data stories for countries using historical data for the 126 Years of Olympic competition. We focused on three main questions: Host country performance, country performance by population, and trends among men and women. Through this analysis, we hoped to identify countries who have notable performances.

Table of Contents:


```
# Import Dependencies
import matplotlib.pyplot as plt
import scipy.stats as sc
import pandas as pd
import numpy as np
import os
```

Show code

Data Cleanup

As is the first step in any data analysis, we first needed to assess the quality of our data. We found that the dataset was excellent, requiring only that we adjust the names of some countries that have gone by different names over the many years that the Olympics have been held. In the cases of East and West Germany or South Vietnam it is simple enough to assign them to their parent countries of Germany and Vietnam. Other cases, such as Korea Town, are more complicated. During one Olympic Games, North and South Korean soccer players combined onto one team which was labeled as Korea Town. Since the majority of athletes on the team were South Koreans, we decided to combine it with South Korea. Finally, some countries' names have changed over the years. The Soviet Union, Russian Olympic Commitee, and Russian Federation are considered seqarate teams by the Olympics officials but for the purpose of our analysis they should be combined under Russia. In the case of Yugoslavia, there isn't a modern country it could be merged into because it split into several countries that still exist today.

Next, to reduce the size of the data, we extracted Gender from the Athlete Event Details csv instead of using the Athlete Biographies. This lowered the required data from 98 MB to 44 MB. We also dropped most of the columns from our population dataset as they did not have relevant information or the data was too incomplete to use.

Once we began analyzing the data, we quickly realized another cleanup task that needed to be done: Team Medals. In the data, each athlete on the team is listed as a separate winner, so when the United States won the gold in Mens Basketball those 12 gold medals were being counted as though they were the same as a runner winning gold in an individual event. To get around this, we merged the rows in our dataframe based on a few conditions. First, we only considered rows where isTeamSport was True. Then, we grouped rows that shared the same Edition, Country Code, Sport, and Event. Finally, we combined them with the rows of the original dataframe where isTeamSport was False. This ensured that all medals were counted accurately and we did not lose any athlete's performance in the process.

```
# Rename Country Profiles country_noc column
df_Country_Profiles.rename(columns={'noc': 'country_noc'}, inplace=True)
```

```
#Condense countries by replacing names with contemporary country or actual country name
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Soviet Union': 'Russian Federation'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Russian Olympic Committee': 'Russian Federation'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'ROC': 'Russian Federation'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Unified Team': 'Russian Federation'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'East Germany': 'Germany'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'West Germany': 'Germany'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Saar': 'Germany'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Bohemia': 'Czechia'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Australasia': 'Australia'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Rhodesia': 'Zimbabwe'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'South Yemen': 'Yemen'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'North Yemen': 'Yemen'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'Serbia and Montenegro': 'Serbia'})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({"Democratic People's Republic of Korea": "Republic of K
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({"Korea Town": "Republic of Korea"})
df_Country_Profiles['country'] = df_Country_Profiles['country'].replace({'South Vietnam': 'Vietnam'})

#Create New CSV for condensed countries
df_Country_Profiles.to_csv("Country_Profiles_Clean.csv", index = False)
```

```
#Clean up the athlete event details tab to collapse all of the teams sports into one athlete
# Filter out individual rows for team sports
non_team_sports = df_Athlete_Event_Details[~df_Athlete_Event_Details['isTeamSport']]
team_sports_grouped = (
    df_Athlete_Event_Details[df_Athlete_Event_Details['isTeamSport']]
    .groupby(['edition', 'country_noc', 'sport', 'event'], as_index=False)
    .agg({
        'athlete': lambda x: ', '.join(x),  # Combine athlete names into one string
        'pos': 'first',                     # Keep position (or change aggregation logic as needed)
        'medal': 'first',                   # Assume medals are the same for team members
        'isTeamSport': 'first'              # Keep the isTeamSport value
    })
)

# Combine the grouped team sports with non-team sports
athlete_results_clean = pd.concat([team_sports_grouped, non_team_sports], ignore_index=True).reset_index(drop = True)
```

Show code

Show code

Show code

Research Questions

$\quad\quad$We had three research topics that we focused on in our data analysis: host, population, and gender. For host, we asked which nations benefited the most from hosting the Olympics and if there were any countries that did not benefit. For population, we explored the relationship between country population and their medal count to see if there were any outliers among the data. Finally, for gender we examined how the ratio between the number of male and female events has changed over the years and whether there were any countries that were outliers in terms of their female success.

Question 1: Host Nation Medal Performance

The process for selecting a host for the Olympics begins about a decade in advance with a vote by the IOC and its stakeholders. Cities will need to construct stadiums, build facilities for athletes to stay in, and prepare their city for the millions of spectators they will need to accomodate. Hosting also comes with some benefits, however, such as not needing to qualify for events and being involved in selecting judges for events like gymnastics and diving. How much do these benefits matter, and do countries perform better when they are hosting?

```
# @title
#make a new dataframe with just sport, country, and results
df_country_results = pd.DataFrame({
    'country_noc':athlete_results_clean['country_noc'],
    'sport':athlete_results_clean['sport'],
    'weighted_results':athlete_results_clean['Weighted Medals'],
    'gold':athlete_results_clean["Gold_ct"],
    'silver':athlete_results_clean["Silver_ct"],
    'bronze':athlete_results_clean["Bronze_ct"],
})

df_country_results = pd.merge(df_country_results, df_Country_Profiles, on='country_noc', how='left')
```

```
#group the results by country and sport and then drop any results where the country has not won much.
df_sport_country_sum = df_country_results.groupby(['country', 'sport'], as_index=False)['weighted_results'].sum()
df_sport_country_sum = df_sport_country_sum[df_sport_country_sum['weighted_results'] > 2].dropna().reset_index(drop = True)
```

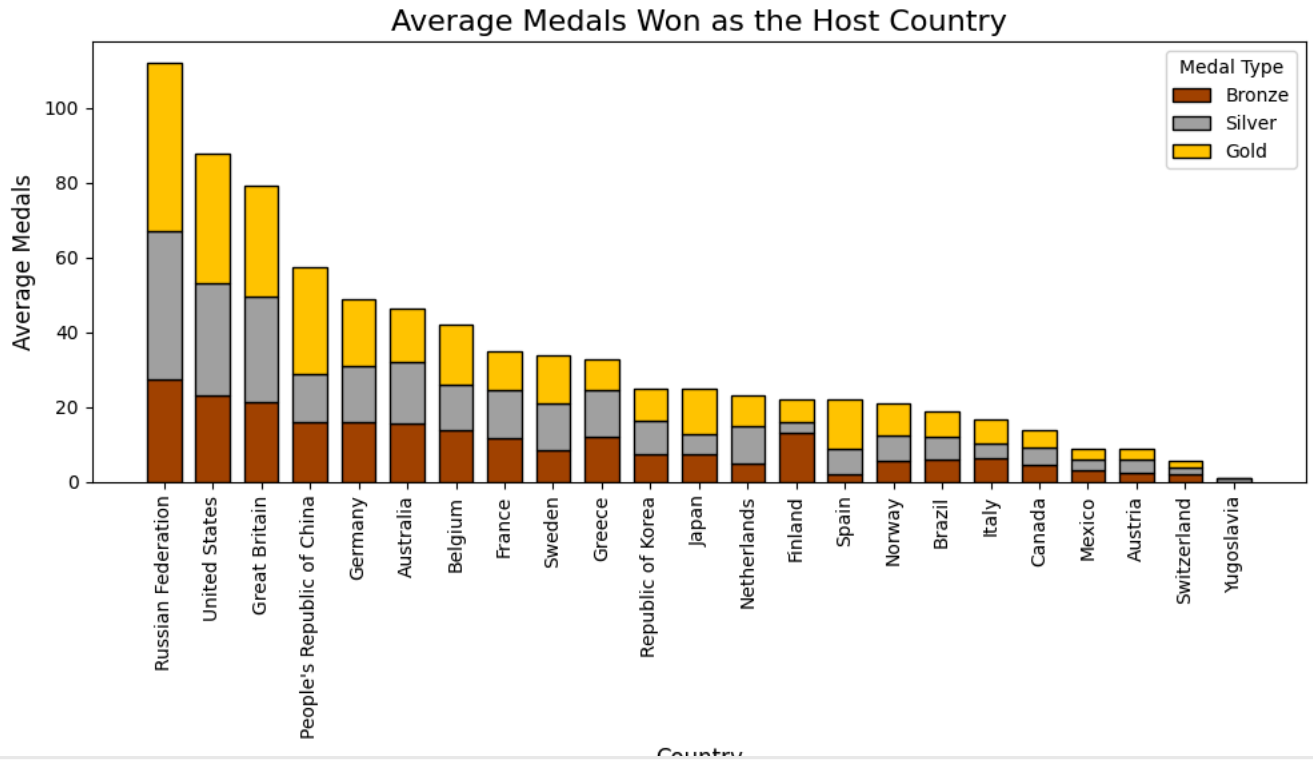
```
#Merging the medal and host tally data
df_host_and_medal_temp = pd.merge(
    df_Medal_Tally_History,
    df_Games_Summary,
    on=["edition", "edition_id", "year", "country_noc"],
    suffixes=("_medals", "_games")
)

df_host_and_medal = pd.merge(df_host_and_medal_temp, df_Country_Profiles, on='country_noc', how='left')
df_host_and_medal.rename(columns={'country_y': 'country'}, inplace=True)
```

```
# Count the number of times each country has hosted
host_count = df_host_and_medal["country"].value_counts().reset_index()
host_count.columns = ["country", "host_count"]

# Calculate total medals for each country as the host
df_medals_by_type = df_host_and_medal.groupby("country")[["gold", "silver", "bronze", "total"]].sum().reset_index()

# Merge medal totals with host count
df_home_medals = pd.merge(df_medals_by_type, host_count, on="country")


# Calculate average medals per hosting for each type
df_home_medals["average_gold_h"] = df_home_medals["gold"] / df_home_medals["host_count"]
df_home_medals["average_silver_h"] = df_home_medals["silver"] / df_home_medals["host_count"]
```

```
df_home_medals["average_bronze_h"] = df_home_medals["bronze"] / df_home_medals["host_count"]
df_home_medals["average_total_h"] = df_home_medals["total"] / df_home_medals["host_count"]
```

After preparing the data, we are ready to create visualizations. First, I compiled the number of medals each country won while hosting and divided it by the number of times it hosted to get its average medal count. The average medals won as the host country can possibly showcase outlier performances a country may have as the host country. The bar graph is sorted in descending order of the average total medal count.

Show code



In first place for average medals won as host is Russia with about 110. Like most of the other countries, their medals are split roughly evenly with gold and silvers being slightly more than bronze. With the exception of Yugoslavia, all of these countries exist today

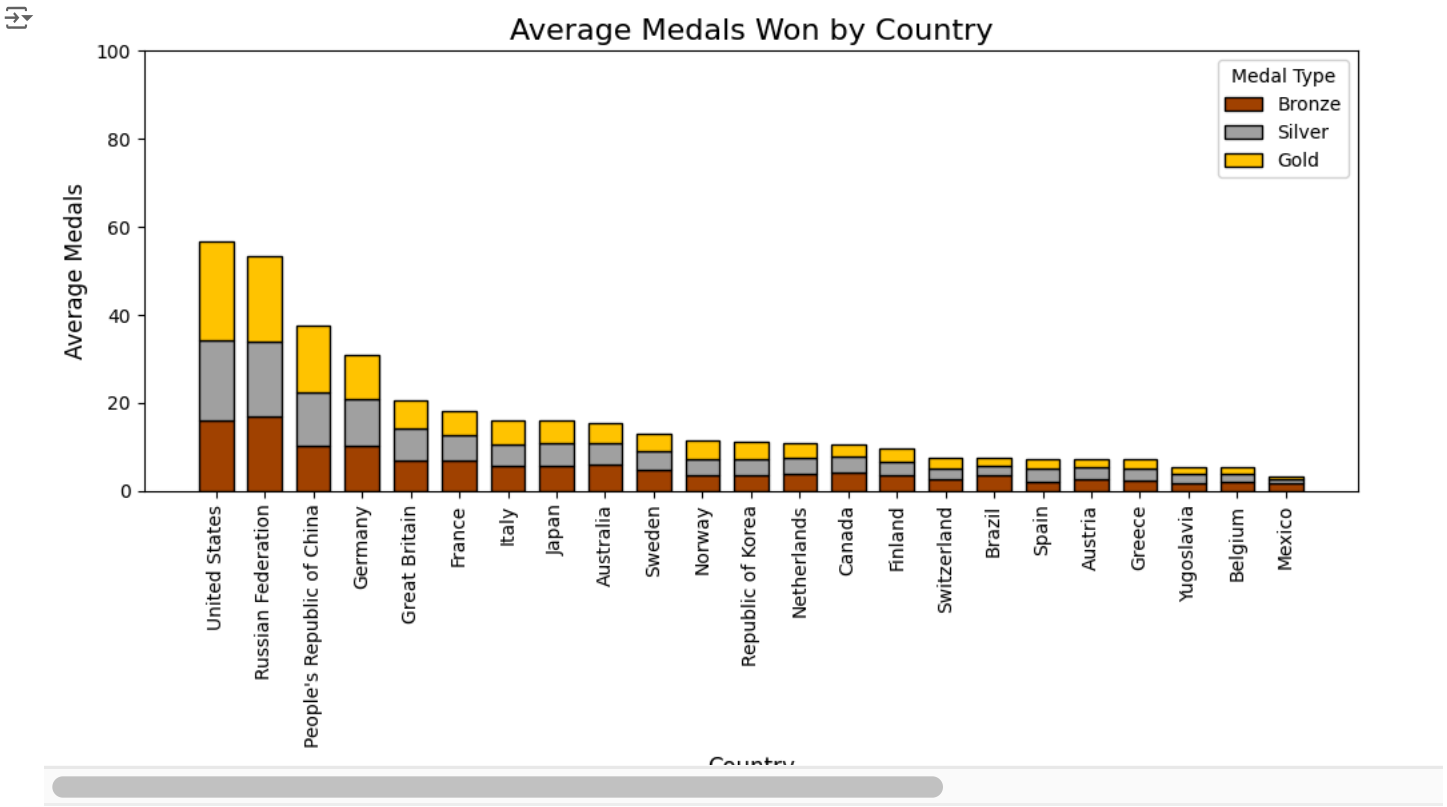
```
# Make a merged df between df_Medal_Tally_History with df_Country_Profiles to input the country data cleaning
df_Medal_Tally = pd.merge(df_Medal_Tally_History, df_Country_Profiles, on='country_noc', how='left')

df_Medal_Tally.rename(columns={'country_y': 'country'}, inplace=True)
```

```
# Use created df to aggregate total
df_country_totals = df_Medal_Tally.groupby("country").agg(
    total_gold=("gold", "sum"),
    total_silver=("silver", "sum"),
    total_bronze=("bronze", "sum"),
    total_medals=("total", "sum"),
    times_competed=("edition", "count")
).reset_index()
```

```
# Calculate average medals per hosting for each type
df_country_totals["average_gold_a"] = df_country_totals["total_gold"] / df_country_totals["times_competed"]
df_country_totals["average_silver_a"] = df_country_totals["total_silver"] / df_country_totals["times_competed"]
df_country_totals["average_bronze_a"] = df_country_totals["total_bronze"] / df_country_totals["times_competed"]
df_country_totals["average_total_a"] = df_country_totals["total_medals"] / df_country_totals["times_competed"]
```

Next, we need to compare it to the average number of medals a country wins. Using the Olympic Medal Tally History data we were able to get the number of times each country has competed and calculate the average medals won.

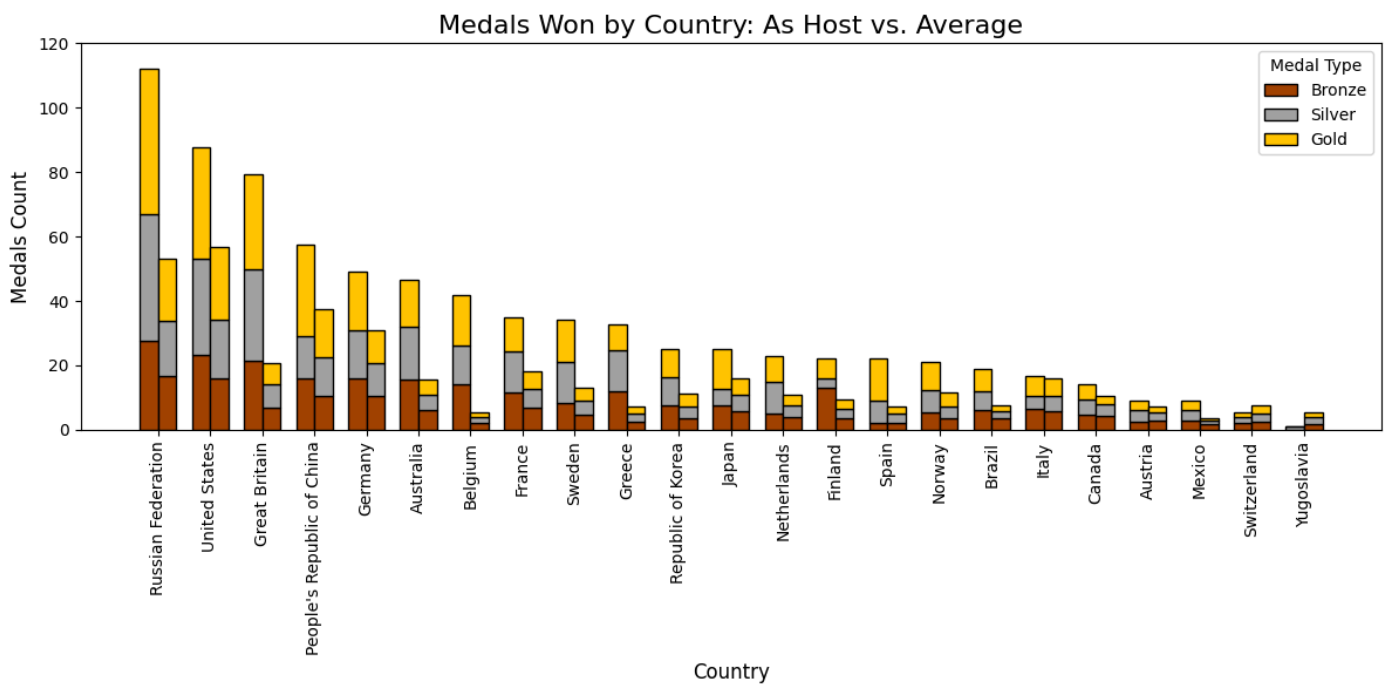
Show code



The United States jumps into first but does not beat Russia by much. Otherwise, this looks pretty similar to the medals won while hosting. Putting the bar graphs next to eachother will help us compare the medal counts.

```
#Merge dataframes so the bars are attached to the country labels on the x-axis
df_country_totals = df_country_totals.sort_values(by="average_total_a", ascending=False)

df_merged = pd.merge(df_country_totals_hosts, df_home_medals, on="country")
df_merged = df_merged.sort_values(by="average_total_h", ascending=False)
```

Show code



Now, looking at the double bar graph above, almost every host country outperforms their average Olympic performance, although some more than others. Russia doubles their medals, winning way more silvers and golds and leaving every other country in the dust. Great Britian and Belgium are even more extreme, more than tripling their average medal counts. Italy, Germany, Japan, and to a lesser extent the United States all benefit much less, not even doubling their medal counts. The only examples of countries performing worse when hosting are Yugoslavia and Switzerland, which have both only hosted a couple of times and win relatively few medals.

Hosting the Olympics is a huge honor, and it requires the host country to begin preparing years in advance. The host country also gets a variety of explicit benefits in the games, including getting to pick a sport to include and automatically qualifying in every event. Additionally, their athletes are able to train in the Olympic facilities. Because of all of this, it makes sense that a country will perform better when they are hosting. Cases where the number of overall medals more than doubles, however, suggest that some countries may be doing more to benefit from hosting than others.

Question 2: Medal Count to Population Performance

By comparing medal counts to population size, we can discover insights into which countries are outperforming and achieving extraordinary outcomes despite their size, and which larger nations might not be reaching their potential. We used another Kaggle Dataset on world population and limited the years to only between 1994 and 2024, giving us a modern time-frame for our analysis. It is important to note that the population data for China was incomplete and needed to be dropped from the dataset, so it does not appear in this section or the regression analysis at the end.

```
#Create column that counts athletes per country
athlete_results_clean["AthletesPerCountry"] = athlete_results_clean.groupby("country_noc")["athlete"].transform("count")
```

```
# Group by 'Country' and sum the 'AthletesPerCountry'
athletes_per_country = athlete_results_clean.groupby("country_noc")["AthletesPerCountry"].sum().reset_index()

# Rename the column for clarity
athletes_per_country.rename(columns={"AthletesPerCountry": "Total_Athletes"}, inplace=True)
```

```
# Sort the DataFrame by the 'Year' column in ascending order
sorted_table = df_world_population.sort_values(by='Rank', ascending=True)
```

```
# Merge df_Medal_Tally_History with df_world_population_ on 'country'
df_medal_tally_pop = pd.merge(df_Medal_Tally_History, df_world_population, on='country', how='inner')

# Filter the merged DataFrame to only include years between 1994 and 2022
filtered_df = df_medal_tally_pop[(df_medal_tally_pop['year'] >= 1994) & (df_medal_tally_pop['year'] <= 2022)]

# Select only the relevant columns: 'country', 'population', 'year', 'gold', 'silver', 'bronze'
df_medal_pop = filtered_df[['country', 'Population', 'year', 'gold', 'silver', 'bronze']]
```

```
# Filter the merged DataFrame to only include years between 1994 and 2022
filtered_df = df_medal_tally_pop[(df_medal_tally_pop['year'] >= 1994) & (df_medal_tally_pop['year'] <= 2022)]

# Sort the DataFrame by 'population' in ascending order and select the first 10 rows
smallest_countries_df = filtered_df.sort_values(by='Population', ascending=True).head(10)

# Select relevant columns to display
result_df = smallest_countries_df[['country', 'Population', 'year', 'gold', 'silver', 'bronze']]
```

```
# Filter the merged DataFrame to only include years between 1994 and 2022
year_mask_s = (df_medal_tally_pop['year'] >= 1994) & (df_medal_tally_pop['year'] <= 2022)
filtered_df = df_medal_tally_pop.loc[year_mask_s, ['country', 'Population', 'gold', 'silver', 'bronze', 'total']]
```

```
#Create an aggregate on a grouped Dataframe to find the mean and sum population based on all year data in addition to the medals
agg_funcs = {'Population': ['mean', 'sum'], 'gold': 'sum', 'silver': 'sum', 'bronze': 'sum', 'total': 'sum'}
recent_medal_sum_df = filtered_df.groupby("country").agg(agg_funcs)
```

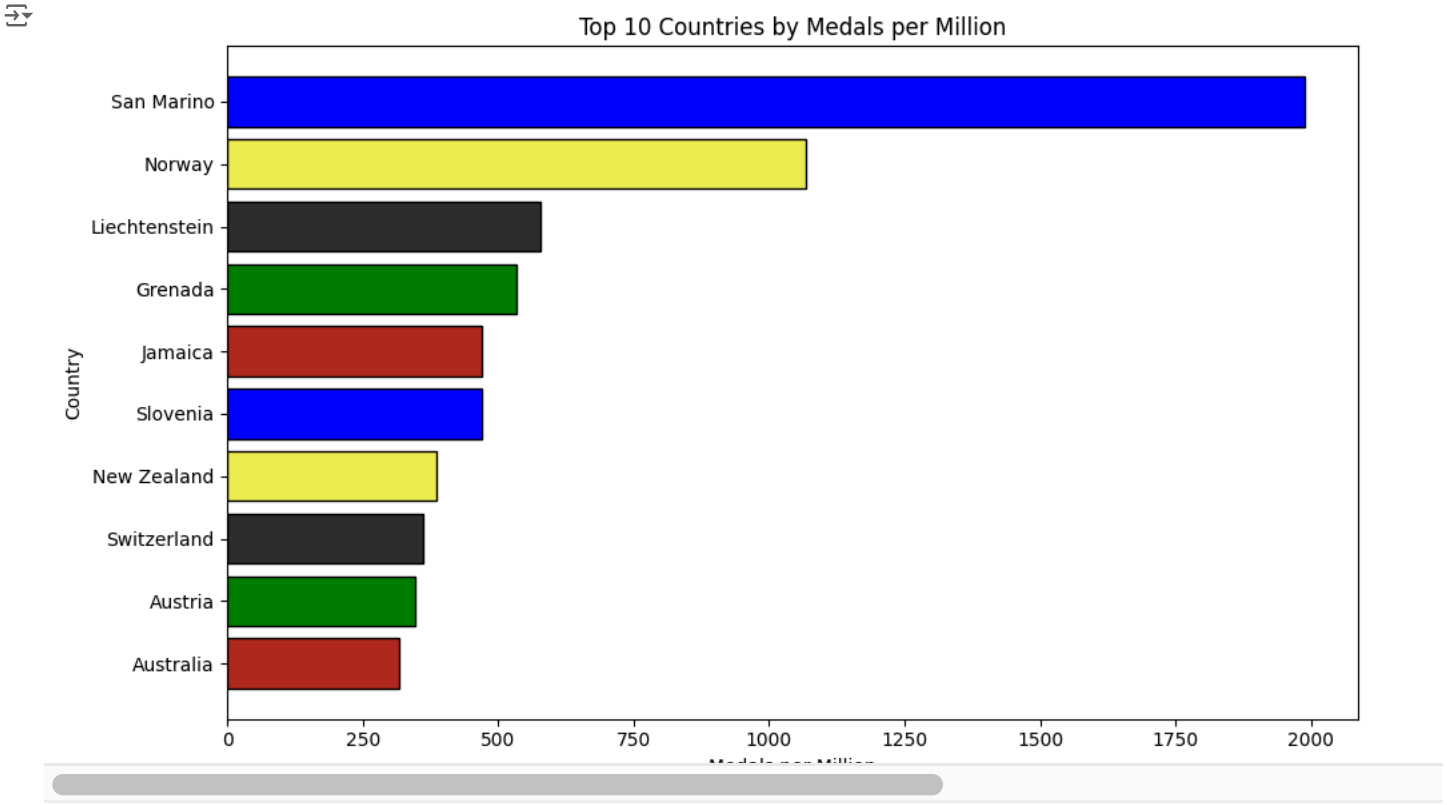
```
#Make column in dataframe that is the result of dividing medals by population then multiplied by a million
recent_medal_sum_df["medals_per_millions"]= 1e6*(recent_medal_sum_df['total']['sum'] / recent_medal_sum_df['Population']['mean']
```

```
# Reset the index if there is a MultiIndex
recent_medal_sum_df_reset = recent_medal_sum_df.reset_index()
```

```
# After resetting the index, proceed with sorting and displaying results
sorted_medals = recent_medal_sum_df_reset.sort_values(by='medals_per_millions', ascending=False)
```

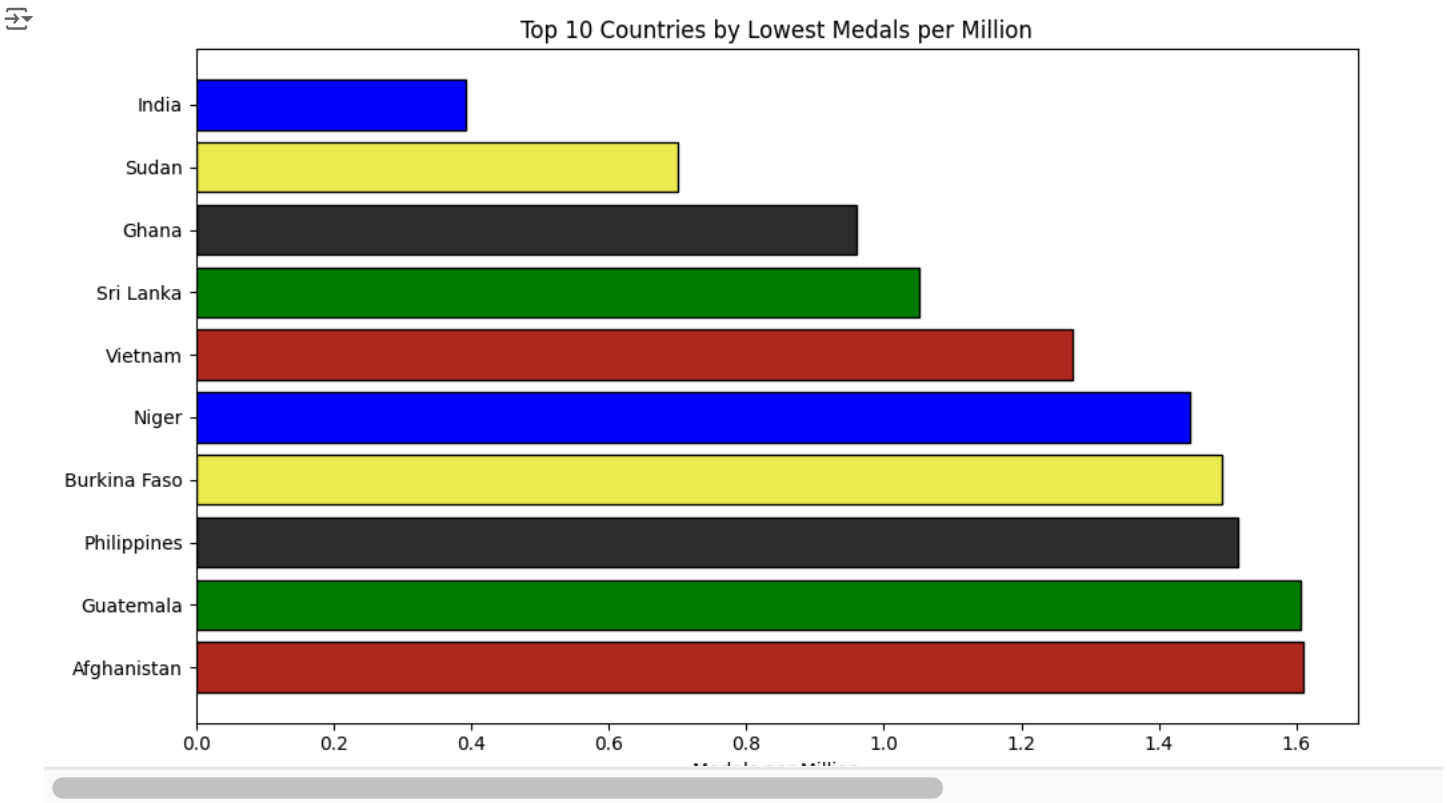
After preparing the data we are ready to create the first visualization examining or the first visualization, we wanted to examine the number of medals a country has won to the average population in millions. This gives us 'Medals per Million', a measure of how many medals the country has won per million people.

Show code



San Marino stands out with nearly double the medals of Norway. A deeper dive into the data reveals that they have only won a couple of medals meaning the results of their analysis will be much less precise. If we consider that San Marino and Liechtenstein both have a population of around 33,000 and have won fewer than 5 medals in the last 20 years, Norway actually stands out as the only country with a relatively large population (at least larger than a music festival) with more than 1000 medals per million people.

Show code



India, Ghana, and Sudan stand out for having less than 1 medal per million people. In our dataset, the population of India was by far the largest, so their performance is skewed negatively. Many of these countries are in Africa or Asia, suggesting that the geographical location may provide additional insights into the country performance.

Question 3: Men vs. Women in Events

The Summer Olympics of 1900 were the first games to allow women to compete, including events like Tug-of-War and Croquet. Many modern events, however, were not made available to women until the 1980s. We wanted to understand how the number of women's medals compare to the number of men's medals and how that ratio has changed over time. Additionally, we wanted to look at whether there were countries whose women perform much better than their men.

The first step, however, was to extract gender from the data. The only data source that had gender as its own column was the Athlete Biography which we did not use because of the size. Instead, we were able to extract the gender from the Event column, usually formatted like "Parallel Bars, Teams, Men". By splitting on the last ", " in the string, we were able to make another column with values "Men, Women, Mixed, Open" to describe the different event types. Mixed events are those that require a man and woman on the team, while open events do not have a gender requirement for participation.

Show code

Show code

Show code

Show code

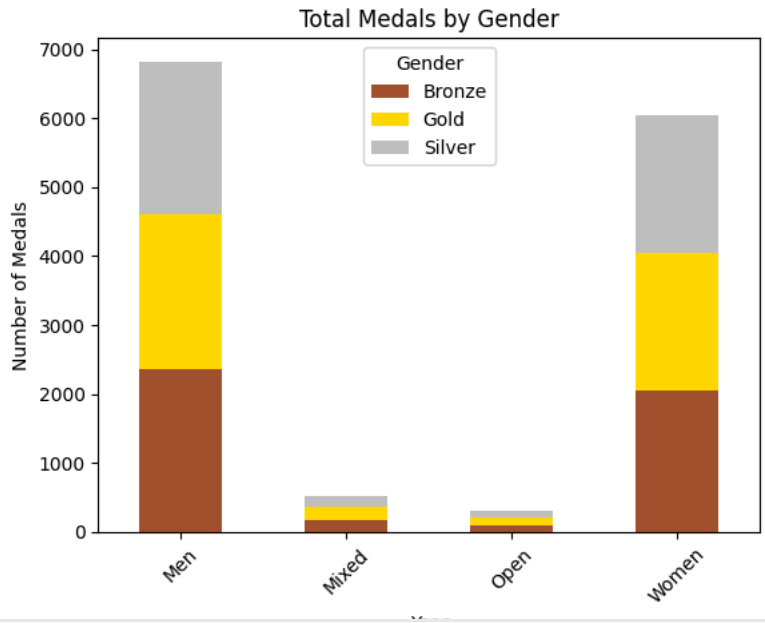
Show code

Show code

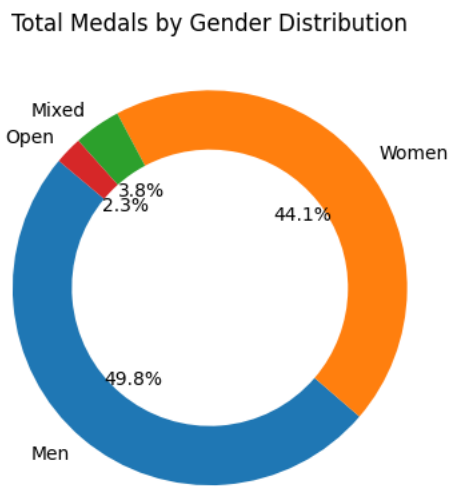
Show code

```
# Split the 'Event' column into 'Event' and 'Gender' columns
country_event_recent_df.sort_values("year")
#look for gold wins: female, last 10 years and top 10 countries
#olympic athlete event to seperate columns
country_event_recent_df = country_event_recent_df.copy()

# Split the 'Event' column into 'Event' and 'Gender' columns
country_event_recent_df[['Event', 'Gender']] = country_event_recent_df['event'].str.extract(r'(.*), (\w+)')
```

Show code

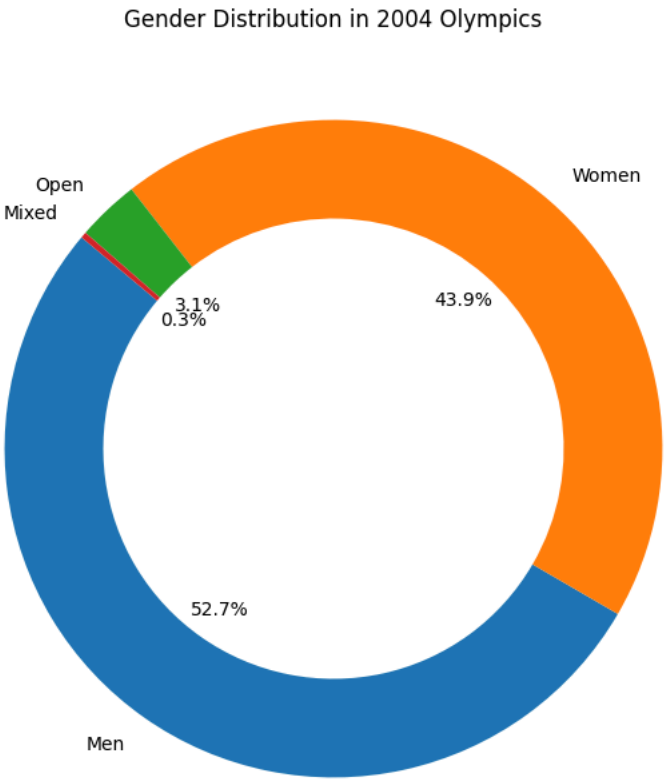


Show code

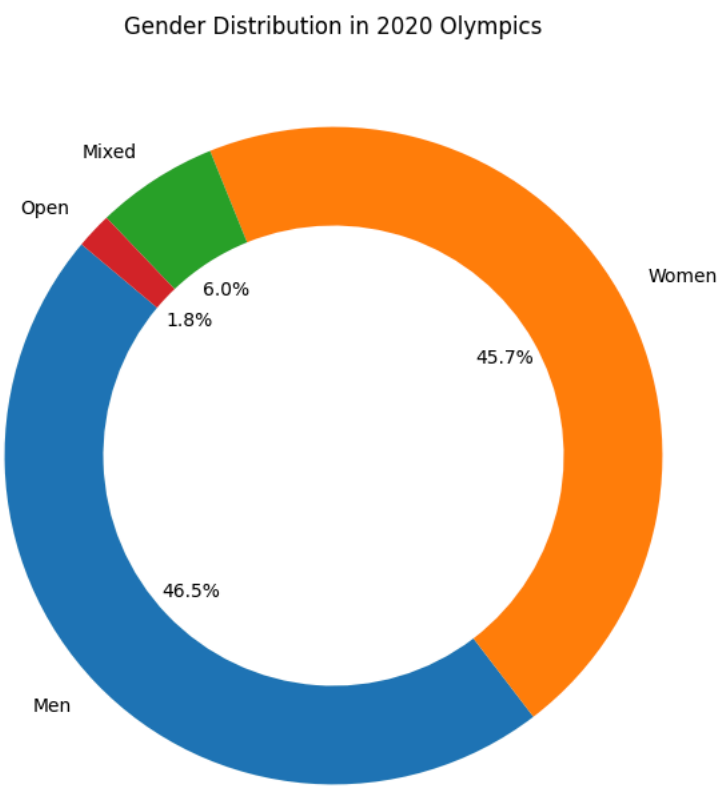


Based on the donut chart, in the last 20 years men have received about 5% more medals than women.

Show code


Gender Distribution in 2004 Olympics
Women
Open
Mixed
43.9%
3.1%
0.3%
52.7%
Men

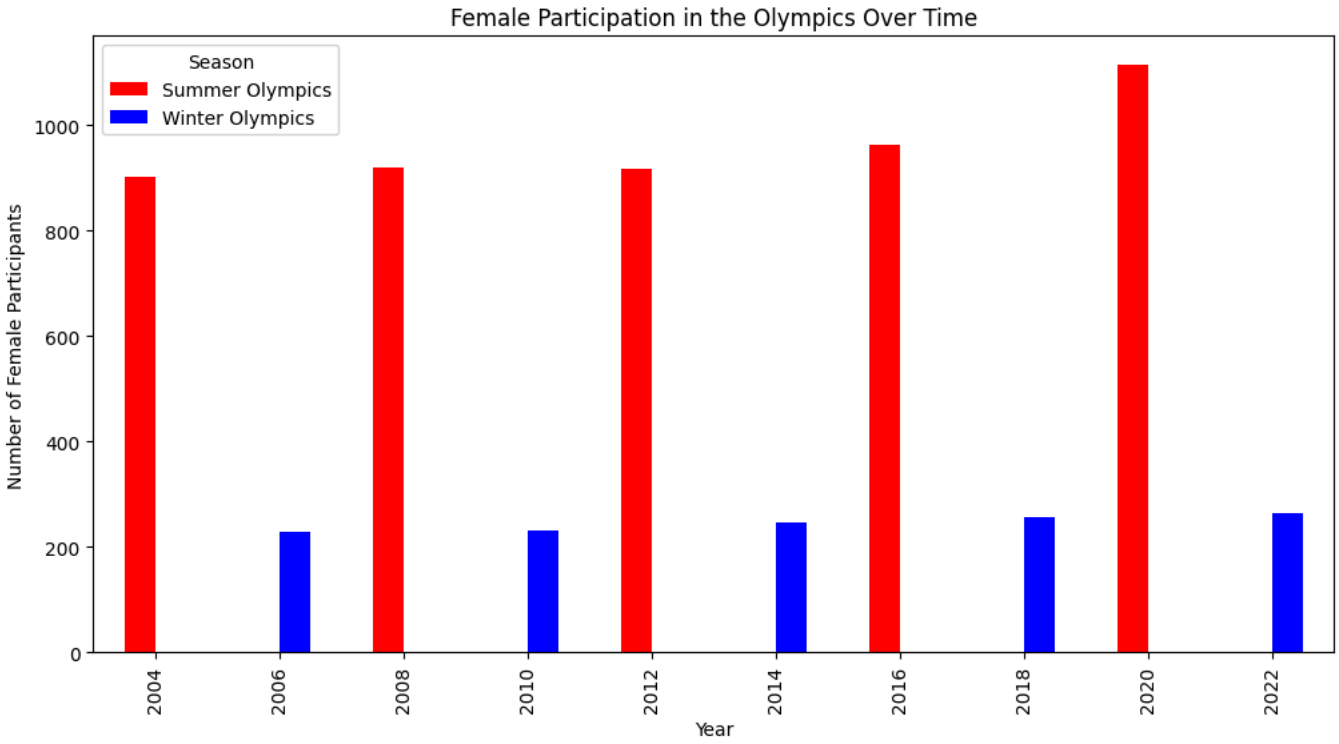
Show code


Gender Distribution in 2020 Olympics
Mixed
Open
Women
6.0%
1.8%
45.7%
46.5%
Men

In the last 20 years, the number of medals that women get have been steadily increasing. In fact, at the 2020 Olympics men and women practically received the same number of medals. Additionally, mixed and open events have become more common, suggesting a shift towards more inclusive competition.

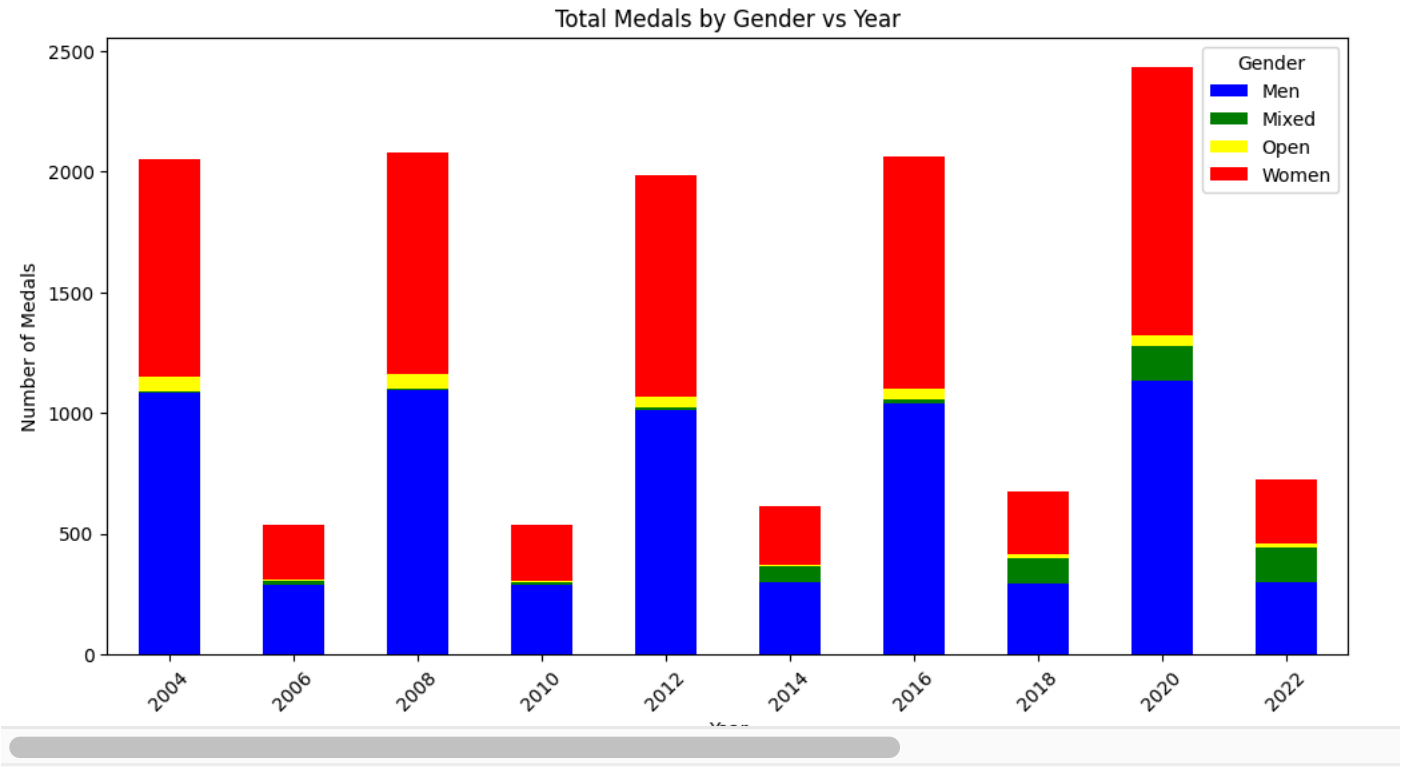
Show code

Show code



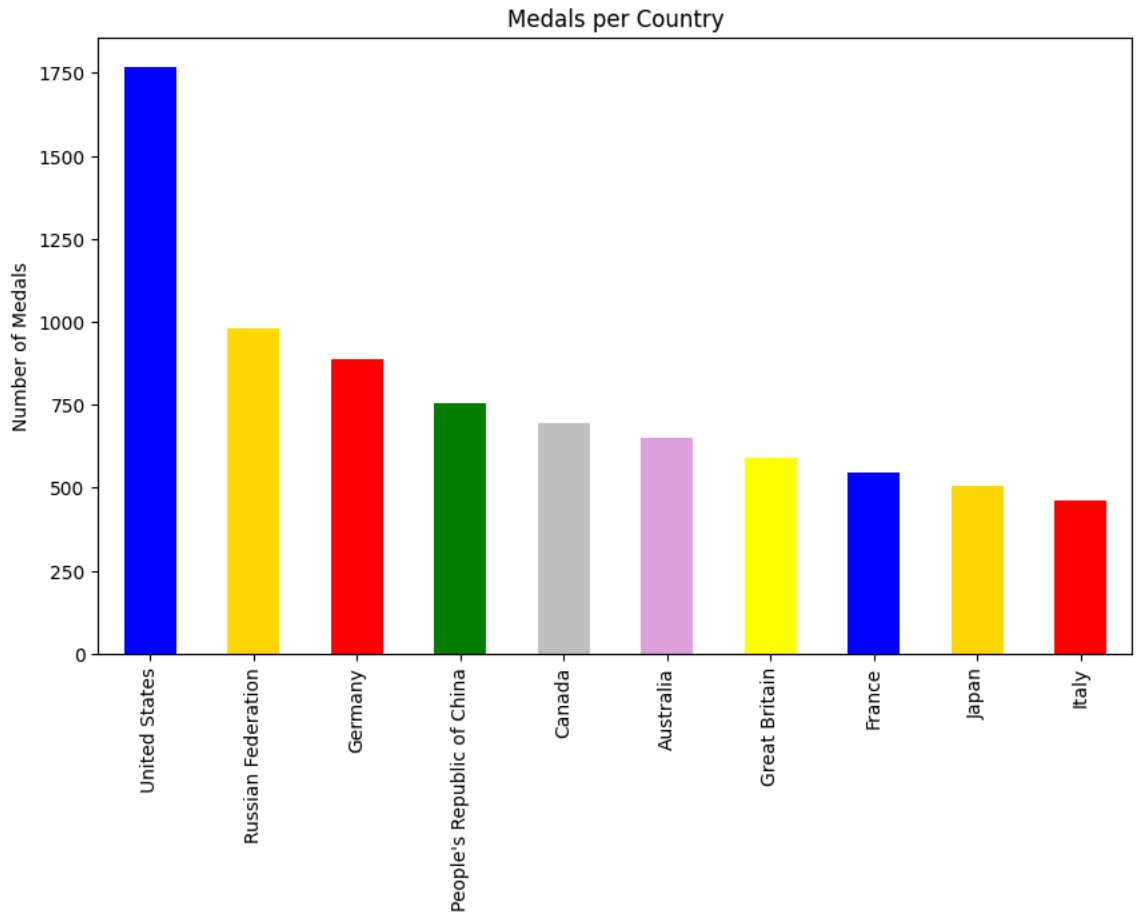
Increasing the number of competitions for women would be impossible without the athletes who compete. Since 2004, female participation in both the Summer and Winter Olympics has steadily increased.

Show code



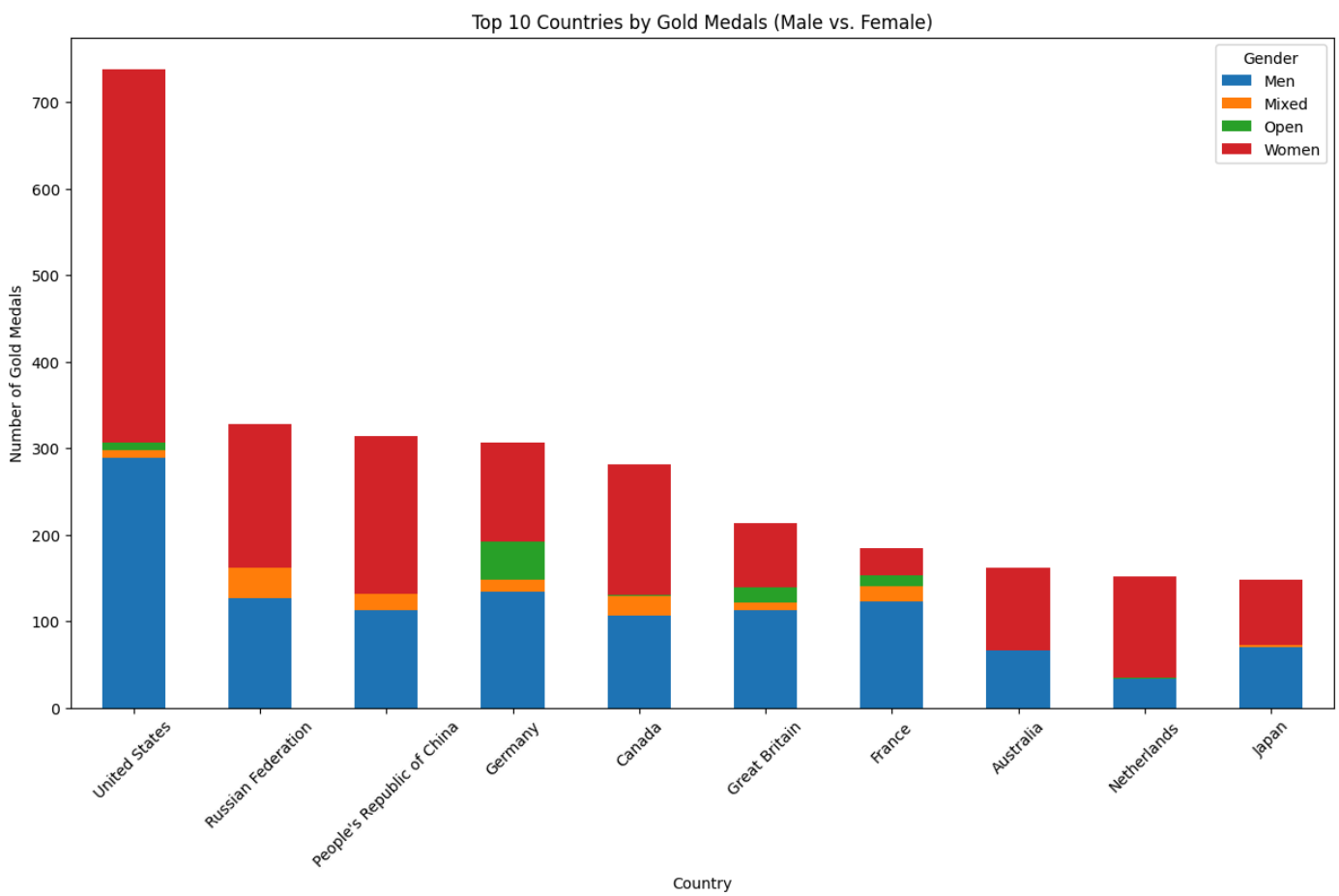
This chart shows how the ratio of events has changed over time. Between the 2010 and 2014 Winter Olympics, a number of mixed events were added, including mixed figure skating and mixed team luge. This trend has continued at the Winter Olympics in 2018 and 2020, with both adding a number of mixed events. The summer events were a little slower to introduce mixed events, with many only beginning in 2020.

Show code



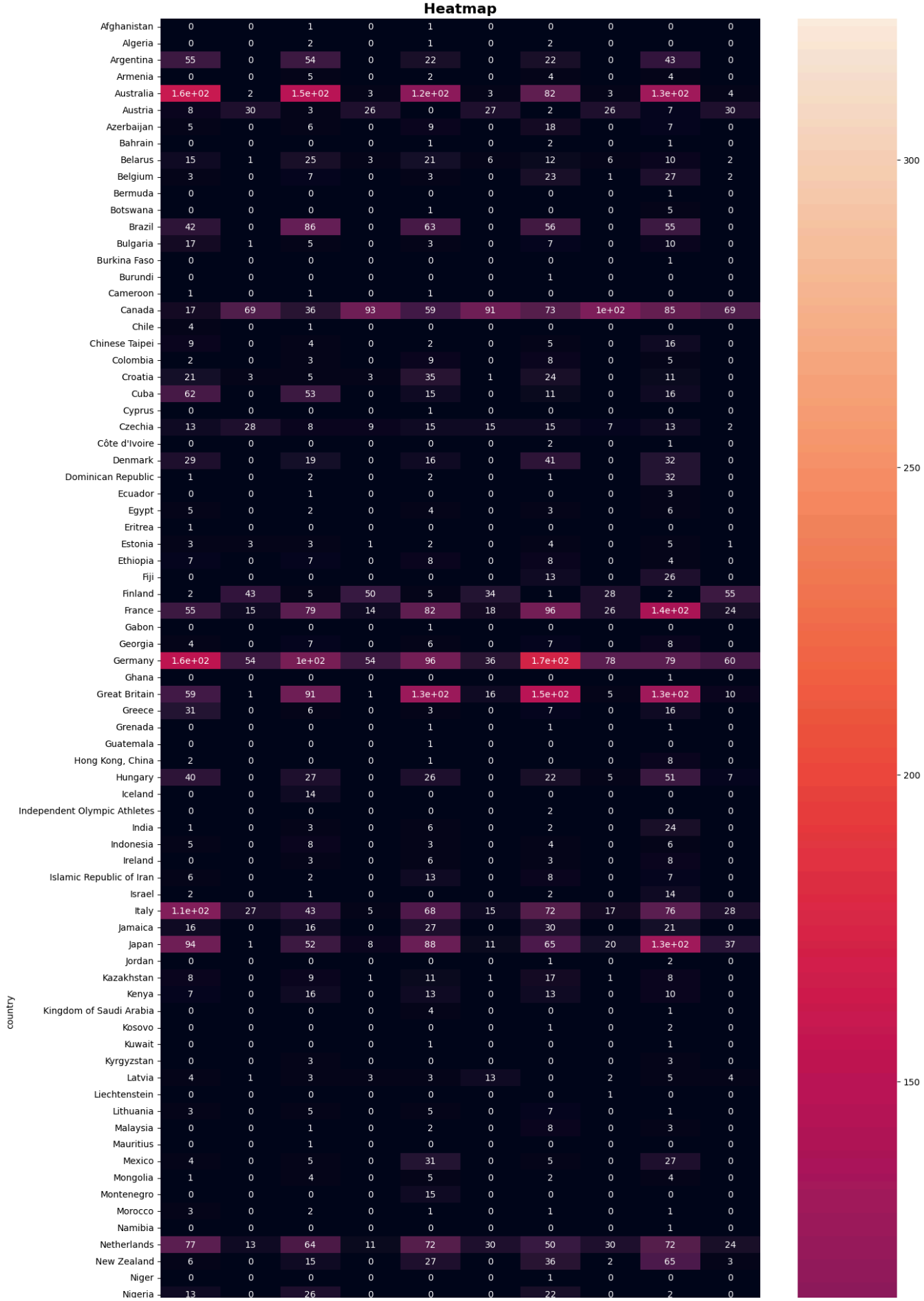
By medal count, the United States stands at the top with around 1750 medals in the last 20 years. Russia is in second, with the following countries having a similar number of medals. How have these countries performed by gender? Has one of them invested heavily in womens athletics or do most countries have a similar composition of medals by gender?

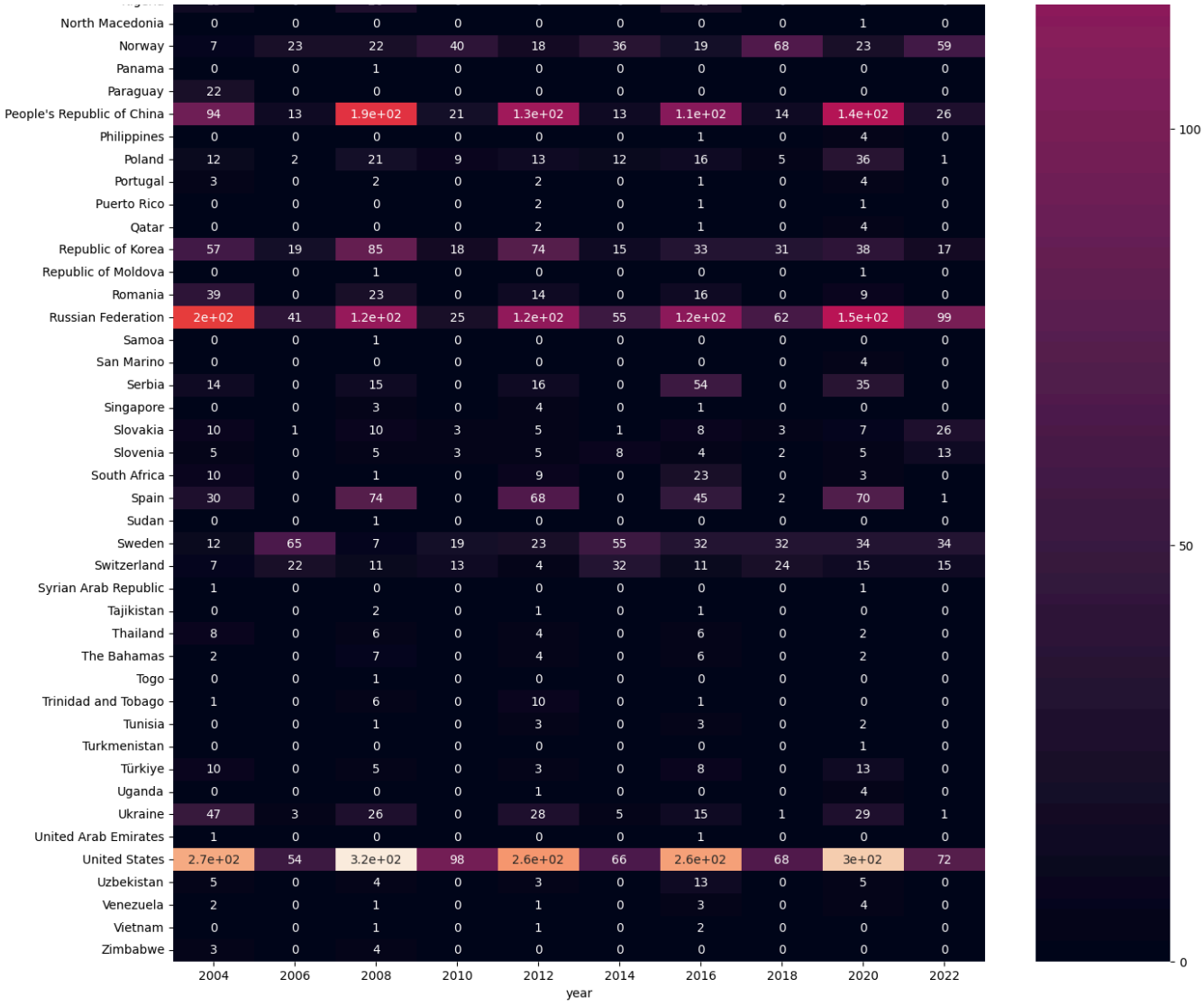
Show code



After filtering for just gold medal winners, the order of the countries has stayed relatively the same. The gender breakdown, however, reveals that the women of the United States have outperformed the men by about 150 gold medals. Similarly, the women of the Netherlands win almost all of the medals for their country. Germany is a bit of an outlier with many more mixed medals than the other countries, and Great Britain and France are the only 2 in the top 10 whose men outperform their women.

Show code


Heatmap
country
Afghanistan 0 0 1 0 1 0 0 0 0 0
Algeria 0 0 2 0 1 0 2 0 0 0
Argentina 55 0 54 0 22 0 22 0 43 0
Armenia 0 0 5 0 2 0 4 0 4 0
Australia 1.6e+02 2 1.5e+02 3 1.2e+02 3 82 3 1.3e+02 4
Austria 8 30 3 26 0 27 2 26 7 30
Azerbaijan 5 0 6 0 9 0 18 0 7 0
Bahrain 0 0 0 0 1 0 2 0 1 0
Belarus 15 1 25 3 21 6 12 6 10 2
Belgium 3 0 7 0 3 0 23 1 27 2
Bermuda 0 0 0 0 0 0 0 0 1 0
Botswana 0 0 0 0 1 0 0 0 5 0
Brazil 42 0 86 0 63 0 56 0 55 0
Bulgaria 17 1 5 0 3 0 7 0 10 0
Burkina Faso 0 0 0 0 0 0 0 0 1 0
Burundi 0 0 0 0 0 0 1 0 0 0
Cameroon 1 0 1 0 1 0 0 0 0 0
Canada 17 69 36 93 59 91 73 1e+02 85 69
Chile 4 0 1 0 0 0 0 0 0 0
Chinese Taipei 9 0 4 0 2 0 5 0 16 0
Colombia 2 0 3 0 9 0 8 0 5 0
Croatia 21 3 5 3 35 1 24 0 11 0
Cuba 62 0 53 0 15 0 11 0 16 0
Cyprus 0 0 0 0 1 0 0 0 0 0
Czechia 13 28 8 9 15 15 15 7 13 2
Côte d'Ivoire 0 0 0 0 0 0 2 0 1 0
Denmark 29 0 19 0 16 0 41 0 32 0
Dominican Republic 1 0 2 0 2 0 1 0 32 0
Ecuador 0 0 1 0 0 0 0 0 3 0
Egypt 5 0 2 0 4 0 3 0 6 0
Eritrea 1 0 0 0 0 0 0 0 0 0
Estonia 3 3 3 1 2 0 4 0 5 1
Ethiopia 7 0 7 0 8 0 8 0 4 0
Fiji 0 0 0 0 0 0 13 0 26 0
Finland 2 43 5 50 5 34 1 28 2 55
France 55 15 79 14 82 18 96 26 1.4e+02 24
Gabon 0 0 0 0 1 0 0 0 0 0
Georgia 4 0 7 0 6 0 7 0 8 0
Germany 1.6e+02 54 1e+02 54 96 36 1.7e+02 78 79 60
Ghana 0 0 0 0 0 0 0 0 1 0
Great Britain 59 1 91 1 1.3e+02 16 1.5e+02 5 1.3e+02 10
Greece 31 0 6 0 3 0 7 0 16 0
Grenada 0 0 0 0 1 0 1 0 1 0
Guatemala 0 0 0 0 1 0 0 0 0 0
Hong Kong, China 2 0 0 0 1 0 0 0 8 0
Hungary 40 0 27 0 26 0 22 5 51 7
Iceland 0 0 14 0 0 0 0 0 0 0
Independent Olympic Athletes 0 0 0 0 0 0 2 0 0 0
India 1 0 3 0 6 0 2 0 24 0
Indonesia 5 0 8 0 3 0 4 0 6 0
Ireland 0 0 3 0 6 0 3 0 8 0
Islamic Republic of Iran 6 0 2 0 13 0 8 0 7 0
Israel 2 0 1 0 0 0 2 0 14 0
Italy 1.1e+02 27 43 5 68 15 72 17 76 28
Jamaica 16 0 16 0 27 0 30 0 21 0
Japan 94 1 52 8 88 11 65 20 1.3e+02 37
Jordan 0 0 0 0 0 0 1 0 2 0
Kazakhstan 8 0 9 1 11 1 17 1 8 0
Kenya 7 0 16 0 13 0 13 0 10 0
Kingdom of Saudi Arabia 0 0 0 0 4 0 0 0 1 0
Kosovo 0 0 0 0 0 0 1 0 2 0
Kuwait 0 0 0 0 1 0 0 0 1 0
Kyrgyzstan 0 0 3 0 0 0 0 0 3 0
Latvia 4 1 3 3 3 13 0 2 5 4
Liechtenstein 0 0 0 0 0 0 0 1 0 0
Lithuania 3 0 5 0 5 0 7 0 1 0
Malaysia 0 0 1 0 2 0 8 0 3 0
Mauritius 0 0 1 0 0 0 0 0 0 0
Mexico 4 0 5 0 31 0 5 0 27 0
Mongolia 1 0 4 0 5 0 2 0 4 0
Montenegro 0 0 0 0 15 0 0 0 0 0
Morocco 3 0 2 0 1 0 1 0 1 0
Namibia 0 0 0 0 0 0 0 0 1 0
Netherlands 77 13 64 11 72 30 50 30 72 24
New Zealand 6 0 15 0 27 0 36 2 65 3
Niger 0 0 0 0 0 0 1 0 0 0
Nigeria 13 0 26 0 0 0 22 0 2 0
300
250
200
150


North Macedonia 0 0 0 0 0 0 0 0 1 0
Norway 7 23 22 40 18 36 19 68 23 59
Panama 0 0 1 0 0 0 0 0 0 0
Paraguay 22 0 0 0 0 0 0 0 0 0
People's Republic of China 94 13 1.9e+02 21 1.3e+02 13 1.1e+02 14 1.4e+02 26
Philippines 0 0 0 0 0 0 1 0 4 0
Poland 12 2 21 9 13 12 16 5 36 1
Portugal 3 0 2 0 2 0 1 0 4 0
Puerto Rico 0 0 0 0 2 0 1 0 1 0
Qatar 0 0 0 0 2 0 1 0 4 0
Republic of Korea 57 19 85 18 74 15 33 31 38 17
Republic of Moldova 0 0 1 0 0 0 0 0 1 0
Romania 39 0 23 0 14 0 16 0 9 0
Russian Federation 2e+02 41 1.2e+02 25 1.2e+02 55 1.2e+02 62 1.5e+02 99
Samoa 0 0 1 0 0 0 0 0 0 0
San Marino 0 0 0 0 0 0 0 0 4 0
Serbia 14 0 15 0 16 0 54 0 35 0
Singapore 0 0 3 0 4 0 1 0 0 0
Slovakia 10 1 10 3 5 1 8 3 7 26
Slovenia 5 0 5 3 5 8 4 2 5 13
South Africa 10 0 1 0 9 0 23 0 3 0
Spain 30 0 74 0 68 0 45 2 70 1
Sudan 0 0 1 0 0 0 0 0 0 0
Sweden 12 65 7 19 23 55 32 32 34 34
Switzerland 7 22 11 13 4 32 11 24 15 15
Syrian Arab Republic 1 0 0 0 0 0 0 0 1 0
Tajikistan 0 0 2 0 1 0 1 0 0 0
Thailand 8 0 6 0 4 0 6 0 2 0
The Bahamas 2 0 7 0 4 0 6 0 2 0
Togo 0 0 1 0 0 0 0 0 0 0
Trinidad and Tobago 1 0 6 0 10 0 1 0 0 0
Tunisia 0 0 1 0 3 0 3 0 2 0
Turkmenistan 0 0 0 0 0 0 0 0 1 0
Türkiye 10 0 5 0 3 0 8 0 13 0
Uganda 0 0 0 0 1 0 0 0 4 0
Ukraine 47 3 26 0 28 5 15 1 29 1
United Arab Emirates 1 0 0 0 0 0 1 0 0 0
United States 2.7e+02 54 3.2e+02 98 2.6e+02 66 2.6e+02 68 3e+02 72
Uzbekistan 5 0 4 0 3 0 13 0 5 0
Venezuela 2 0 1 0 1 0 3 0 4 0
Vietnam 0 0 1 0 1 0 2 0 0 0
Zimbabwe 3 0 4 0 0 0 0 0 0 0
2004 2006 2008 2010 2012 2014 2016 2018 2020 2022
year
100
50
0