Olympic Analysis

March 9, 2025

Olympic Data Analysis

0.1 Introduction

This report explores a dataset containing details of Olympic athletes, including demographic information, physical attributes, participation, and medal achievements. The objective is to uncover patterns, trends, and insights while identifying missing data or inconsistencies for further analysis. Key areas of focus include demographic trends, physical characteristics, participation over time, and medal distributions. This analysis sets the foundation for more detailed investigations and actionable insights.

0.1.1 Importing the Libraries

[2]:
```
import pandas as pd
import numpy as np
from sqlalchemy import create_engine
from urllib.parse import quote_plus
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import gridspec

import warnings
warnings.filterwarnings("ignore")
```

0.1.2 Data Feature Description

For the dataset considered in the analysis following is the detailed overview for the same, representing important info for each feature.

[3]:
```
df = pd.read_csv('olympics.csv')
```

[4]:
```
df.shape
```

[4]:
```
(271116, 15)
```

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271116 entries, 0 to 271115
Data columns (total 15 columns):
```

```
 #   Column   Non-Null Count   Dtype
---  ------   --------------   -----
 0   id       271116 non-null  int64
 1   name     271116 non-null  object
 2   sex      271116 non-null  object
 3   age      261642 non-null  float64
 4   height   210945 non-null  float64
 5   weight   208241 non-null  float64
 6   team     271116 non-null  object
 7   noc      271116 non-null  object
 8   games    271116 non-null  object
 9   year     271116 non-null  int64
 10  season   271116 non-null  object
 11  city     271116 non-null  object
 12  sport    271116 non-null  object
 13  event    271116 non-null  object
 14  medal    39783 non-null   object
dtypes: float64(3), int64(2), object(10)
memory usage: 31.0+ MB
```

[6]:
```
df.describe()
```

[6]:
```
                  id            age         height         weight  \
count  271116.000000  261642.000000  210945.000000  208241.000000
mean    68248.954396      25.556898     175.338970      70.702393
std     39022.286345       6.393561      10.518462      14.348020
min         1.000000      10.000000     127.000000      25.000000
25%     34643.000000      21.000000     168.000000      60.000000
50%     68205.000000      24.000000     175.000000      70.000000
75%    102097.250000      28.000000     183.000000      79.000000
max    135571.000000      97.000000     226.000000     214.000000

                year
count  271116.000000
mean     1978.378480
std        29.877632
min      1896.000000
25%      1960.000000
50%      1988.000000
75%      2002.000000
max      2016.000000
```

[7]:
```
df.isnull().sum()
```

[7]:
```
id            0
name          0
sex           0
age        9474
```

```
height     60171
weight     62875
team           0
noc            0
games          0
year           0
season         0
city           0
sport          0
event          0
medal     231333
dtype: int64
```

0.1.3 Handling missing values

Medal Column Null Values Handling

[8]:
```
# Replace 'NA' in the medal column with 'None'
df['medal'] = df['medal'].fillna('None')

# Verify the results
print(df.isnull().sum())
```

```
id            0
name          0
sex           0
age        9474
height    60171
weight    62875
team          0
noc           0
games         0
year          0
season        0
city          0
sport         0
event         0
medal         0
dtype: int64
```

[9]:
```
df.head()
```

[9]:
```
   id                      name sex   age  height  weight            team  \
0   1                 A Dijiang   M  24.0   180.0    80.0           China
1   2                  A Lamusi   M  23.0   170.0    60.0           China
2   3       Gunnar Nielsen Aaby   M  24.0     NaN     NaN         Denmark
3   4      Edgar Lindenau Aabye   M  34.0     NaN     NaN  Denmark/Sweden
4   5  Christine Jacoba Aaftink   F  21.0   185.0    82.0     Netherlands
```

```
   noc         games  year  season       city          sport  \
0  CHN   1992 Summer  1992  Summer  Barcelona     Basketball
1  CHN   2012 Summer  2012  Summer     London           Judo
2  DEN   1920 Summer  1920  Summer  Antwerpen       Football
3  DEN   1900 Summer  1900  Summer      Paris     Tug-Of-War
4  NED   1988 Winter  1988  Winter    Calgary  Speed Skating

                                event medal
0       Basketball Men's Basketball  None
1      Judo Men's Extra-Lightweight  None
2           Football Men's Football  None
3       Tug-Of-War Men's Tug-Of-War  Gold
4  Speed Skating Women's 500 metres  None
```

Age Null Values Handling

[10]:
```python
dfFillAge = df[~df['age'].isnull()].
    ↪groupby(['year','season','sex','sport','event'])[['age']].mean().
    ↪reset_index()
dfFillAge['age'] = dfFillAge['age'].astype(int)
df = pd.merge(df,dfFillAge,on=['year','season','sex','sport','event'],
    how='left',
    suffixes=('', '_mean'))
df['age'] = df['age'].fillna(df['age_mean'])
df.dropna(subset=['age'],inplace=True)
print(df.isnull().sum())
```

```
id              0
name            0
sex             0
age             0
height      60022
weight      62728
team            0
noc             0
games           0
year            0
season          0
city            0
sport           0
event           0
medal           0
age_mean        0
dtype: int64
```

Height Null Values Handling

[11]:

```
dfFillheight = df[~df['height'].isnull()].
 ↪groupby(['year','season','sex','sport','event'])[['height']].mean().
 ↪reset_index()
dfFillheight['height'] = dfFillheight['height'].astype(int)
df = pd.merge(df,dfFillheight,on=['year','season','sex','sport','event'],
    how='left',
    suffixes=('', '_mean'))
df['height'] = df['height'].fillna(df['height_mean'])
df.dropna(subset=['height'],inplace=True)
print(df.isnull().sum())
```

```
id                 0
name               0
sex                0
age                0
height             0
weight         56826
team               0
noc                0
games              0
year               0
season             0
city               0
sport              0
event              0
medal              0
age_mean           0
height_mean        0
dtype: int64
```

Weight Null Values Handling

[12]:
```
dfFillweight = df[~df['weight'].isnull()].
 ↪groupby(['year','season','sex','sport','event'])[['weight']].mean().
 ↪reset_index()
dfFillweight['weight'] = dfFillweight['weight'].astype(int)
df = pd.merge(df,dfFillweight,on=['year','season','sex','sport','event'],
    how='left',
    suffixes=('', '_mean'))
df['weight'] = df['weight'].fillna(df['weight_mean'])
df.dropna(subset=['weight'],inplace=True)
print(df.isnull().sum())
```

```
id             0
name           0
sex            0
age            0
height         0
weight         0
```

```
team           0
noc            0
games          0
year           0
season         0
city           0
sport          0
event          0
medal          0
age_mean       0
height_mean    0
weight_mean    0
dtype: int64
```

[13]:
```
df.shape
```

[13]:
```
(257646, 18)
```

[14]:
```
df.drop(['age_mean','height_mean','weight_mean'],axis=1,inplace=True)
```

[186]:
```
def plotTwoCharts(df, chartParams):
    # print(df.columns)
    """
    Function to plot two charts side by side with different chart types (line,␣
 ↪scatter, bar, pie, histogram).
    Parameters:
    df (DataFrame): The dataframe containing the data
    chartParams (dict): Dictionary containing chart details
    """
    totalCharts = len(chartParams['chartData'])
    rows = (totalCharts + 1) // 2  # Calculate rows for the fixed 2-column␣
 ↪layout

    # Create subplots
    fig, axes = plt.subplots(rows, 2, figsize=(19, 5 * rows))
    axes = axes.flatten()  # Flatten to simplify indexing

    for chart in range(totalCharts):
        chartDetails = chartParams['chartData'][chart]
        chartType = chartDetails['type']
        xvalue = chartDetails['xCol']
        yvalues = chartDetails.get('yCol', [])
        lvalue = chartDetails.get('legend', None)  # Use .get to handle␣
 ↪optional keys

        sns.set_style("darkgrid")
        if chartType == 'line':
```

```
            if lvalue:  # If 'legend' is specified, restructure the data for grouped plotting
                plot_df = pd.melt(
                    df,
                    id_vars=[xvalue],
                    value_vars=yvalues,
                    var_name='Group',
                    value_name='Value'
                )
                plot_df['Group'] = plot_df['Group'].replace(dict(zip(yvalues, lvalue)))
                sns.lineplot(
                    data=plot_df,
                    x=xvalue,
                    y='Value',
                    hue='Group',
                    marker='o',
                    ax=axes[chart]
                )
            else:  # Simple line plot
                for col in yvalues:
                    sns.lineplot(
                        data=df,
                        x=xvalue,
                        y=col,
                        marker='o',
                        ax=axes[chart]
                    )
        elif chartType == 'scatter':
            sns.scatterplot(data=df, x=xvalue, y=yvalues[0], hue=lvalue, ax=axes[chart])
        elif chartType == 'bar':
            if len(yvalues) > 1 and lvalue:
                    melted_df = pd.melt(
                        df,
                        id_vars=[xvalue],
                        value_vars=yvalues,
                        var_name='Group',
                        value_name='Value'
                    )
                    melted_df['Group'] = melted_df['Group'].replace(
                        dict(zip(yvalues, lvalue))
                    )
                    sns.barplot(
                        data=melted_df,
                        x=xvalue,
                        y='Value',
```

```
                        hue='Group',
                        palette="Set2",
                        ax=axes[chart]
                    )
            else:
                bars = sns.barplot(
                    data=df,
                    x=xvalue,
                    y=yvalues[0],
                    color="#66b3ff",
                    ax=axes[chart]
                )
                for bar in bars.patches:
                    height = bar.get_height()
                    bars.annotate(
                        f'{height:.1f}',
                        (bar.get_x() + bar.get_width() / 2, height),
                        ha='center',
                        va='bottom',
                        fontsize=9,
                        color='black'
                    )
        elif chartType == 'pie':
            # For pie chart: Use the first column in xCol as categories
            # Enhanced Pie Chart Code
            pie_data = df[xvalue].value_counts()

            # Create the pie chart
            wedges, texts, autotexts = axes[chart].pie(
                pie_data,
                autopct='%1.1f%%',
                startangle=90,
                labels=pie_data.index,
                colors=['#66b3ff', '#ff9999', '#99ff99', '#ffcc99'],  # Custom color palette
                textprops={'fontsize': 10, 'color': 'black'}  # Text properties for better readability
            )

            # Style the percentage labels
            for autotext in autotexts:
                autotext.set_fontsize(12)
                autotext.set_fontweight('bold')

            # Set title with better styling
            axes[chart].set_title(
                "Loan Default Distribution".upper(),
```

```
                fontsize=14,
                fontweight='bold',
                pad=20
            )

            # Remove y-axis label
            axes[chart].set_ylabel("")

            # Add a legend outside the chart
            axes[chart].legend(
                pie_data.index,
                title="Categories",
                loc="upper right",
                bbox_to_anchor=(1.2, 0.9),  # Position outside the chart
                fontsize=10
            )

        elif chartType == 'histogram':
            sns.histplot(data=df, x=xvalue, bins=20, kde=True, ax=axes[chart])

        axes[chart].set_title(chartDetails['chartTitle'].upper(), fontsize=12)
        axes[chart].set_xlabel(chartDetails.get('xlabel', xvalue.upper()),␣
↪fontsize=10)
        axes[chart].set_ylabel(chartDetails.get('ylabel', ', '.join(yvalues)),␣
↪fontsize=10)
        axes[chart].tick_params(axis='both', which='major', labelsize=10)

    # Hide any unused axes
    for ax in axes[totalCharts:]:
        ax.set_visible(False)

    # Adjust layout for better spacing
    plt.tight_layout()
    plt.show()
```

0.1.4 Exploratory Data Analysis

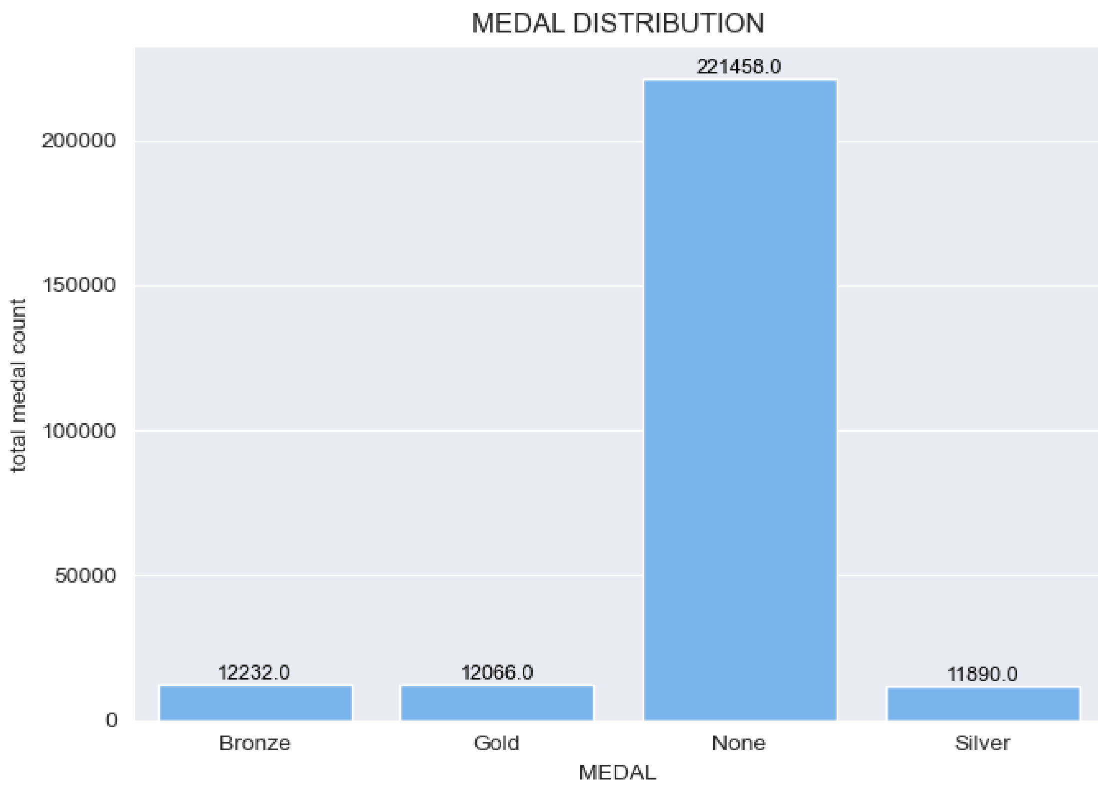
[126]:
```
chartDf = df.groupby('medal').count().iloc[:, 0].reset_index()
chartDf
```

[126]:

	medal	id
0	Bronze	12232
1	Gold	12066
2	None	221458
3	Silver	11890

[129]:

```
chartParams = {
    "chartData": [
       {
            "type": "bar",  # Simple bar chart
            "xCol": "medal",  # States as the x-axis
            "yCol": ["id"],  # Loan defaults as the y-axis
            "chartTitle": "Medal DIstribution",
            "ylabel":"total medal count",
            "legend": None  # Simple bar chart without grouping
        },
    ]
}

plotTwoCharts(chartDf, chartParams)
```



- **Majority of athletes (221,458) did not win a medal, showing high competition.**
- **Bronze (12,232) is slightly higher than Gold (12,066) and Silver (11,890), likely due to double Bronze awards in some events.**
- **Only a small percentage of participants achieve podium finishes.**

[130]:

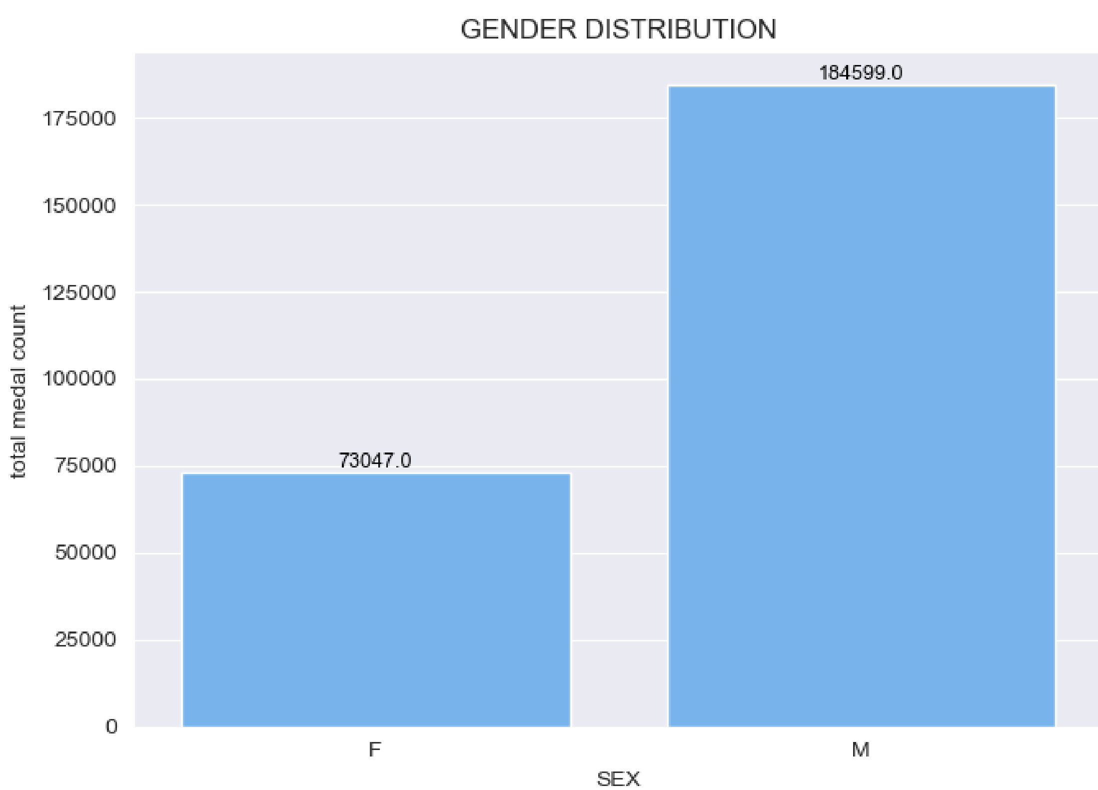
```
chartDf = df.groupby('sex').count().iloc[:, 0].reset_index()
chartDf
```

[130]:

```
  sex      id
0   F   73047
1   M  184599
```

[131]:

```
chartParams = {
    "chartData": [
       {
            "type": "bar",  # Simple bar chart
            "xCol": "sex",  # States as the x-axis
            "yCol": ["id"],  # Loan defaults as the y-axis
            "chartTitle": "Gender DIstribution",
            "ylabel":"total medal count",
            "legend": None  # Simple bar chart without grouping
        },
    ]
}

plotTwoCharts(chartDf, chartParams)
```

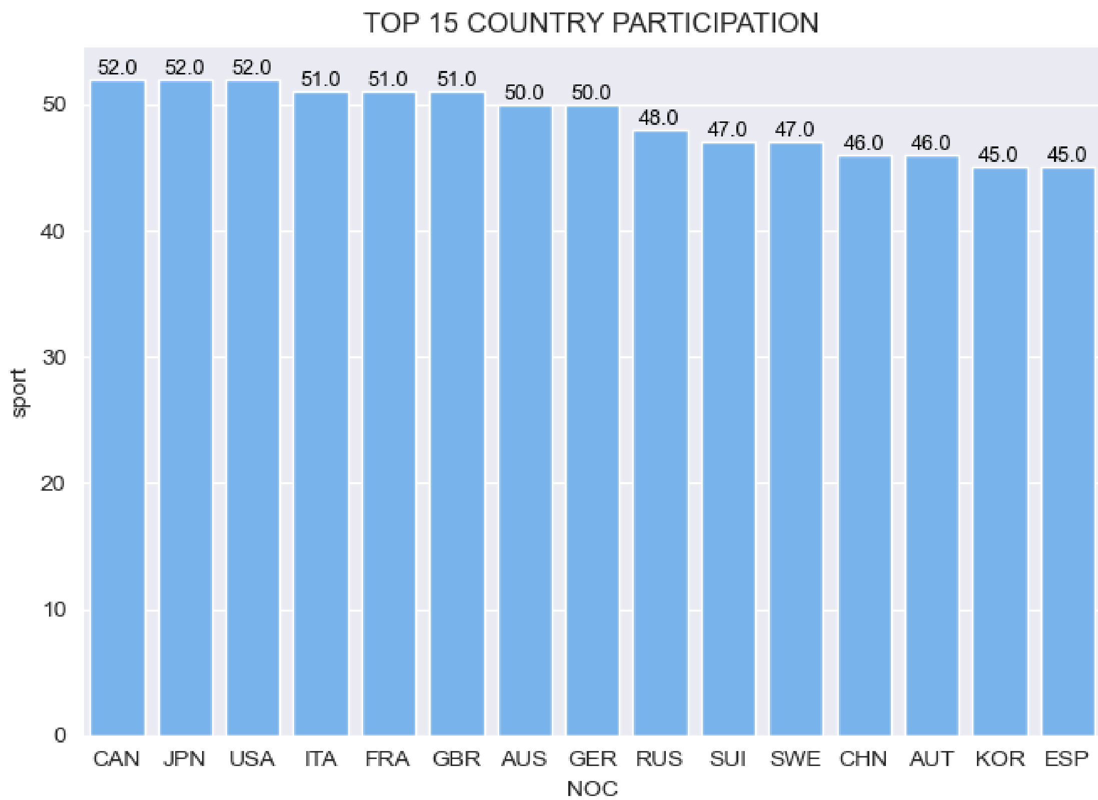


- **Male participants (184,599) significantly outnumber female participants (73,047).**
- **The gender gap indicates historical underrepresentation of female athletes.**
- **The data highlights the need for more inclusivity in sports.**

[132]:
```python
chartDf = df.groupby(['team','noc']).agg({'sport': 'nunique'}).reset_index().
↪sort_values(by='sport', ascending=False)
chartDf
```

[132]:
```
                team  noc  sport
133           Canada  CAN     52
370            Japan  JPN     52
840    United States  USA     52
363            Italy  ITA     51
260           France  FRA     51
..               …   …     …
388      Kannibaltje  FRA      1
389         Kathleen  USA      1
392             Kiel  DEN      1
393       Kingfisher  MYA      1
915             rn-2  FIN      1

[916 rows x 3 columns]
```

[133]:
```python
chartParams = {
    "chartData": [
       {
            "type": "bar",  # Simple bar chart
            "xCol": "noc",  # States as the x-axis
            "yCol": ["sport"],  # Loan defaults as the y-axis
            "chartTitle": "Top 15 Country Participation",
            "legend": None  # Simple bar chart without grouping
        },
    ]
}

plotTwoCharts(chartDf[:15], chartParams)
```



- **Canada, Japan, and the USA have the highest participation (52 sports).**
- **Italy, France, and Great Britain follow closely (51 sports).**
- **Spain and South Korea have the lowest among the top 15 (45 sports).**
- **Indicates strong multi-sport engagement in leading nations.**

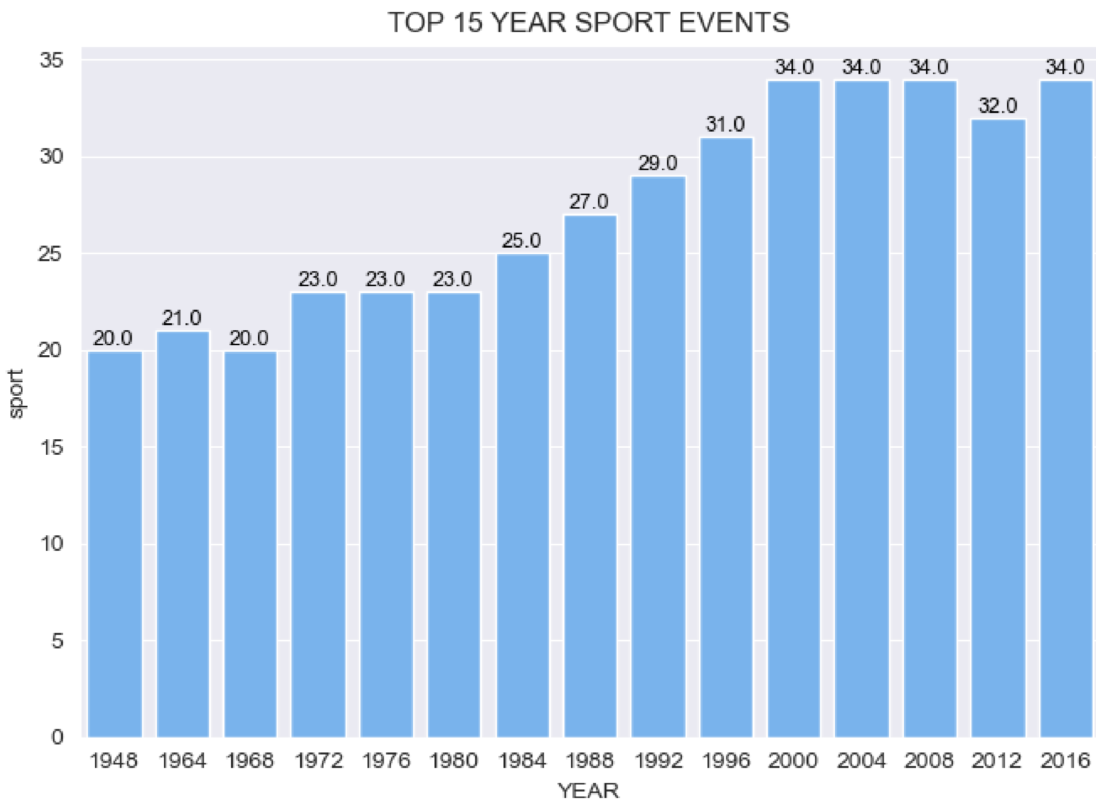
[139]:
```
chartDf = df.groupby(['games','year','city']).agg({'sport': 'nunique'}).
↪reset_index().sort_values(by='sport', ascending=False)
chartDf.head()
```

[139]:

	games	year	city	sport
51	2016 Summer	2016	Rio de Janeiro	34
47	2008 Summer	2008	Beijing	34
45	2004 Summer	2004	Athina	34
43	2000 Summer	2000	Sydney	34
49	2012 Summer	2012	London	32

[135]:
```
chartParams = {
    "chartData": [
      {
          "type": "bar",  # Simple bar chart
          "xCol": "year",  # States as the x-axis
          "yCol": ["sport"],  # Loan defaults as the y-axis
```

```
            "chartTitle": "Top 15 Year Sport Events",
            "legend": None  # Simple bar chart without grouping
        },
    ]
}

plotTwoCharts(chartDf[:15], chartParams)
```

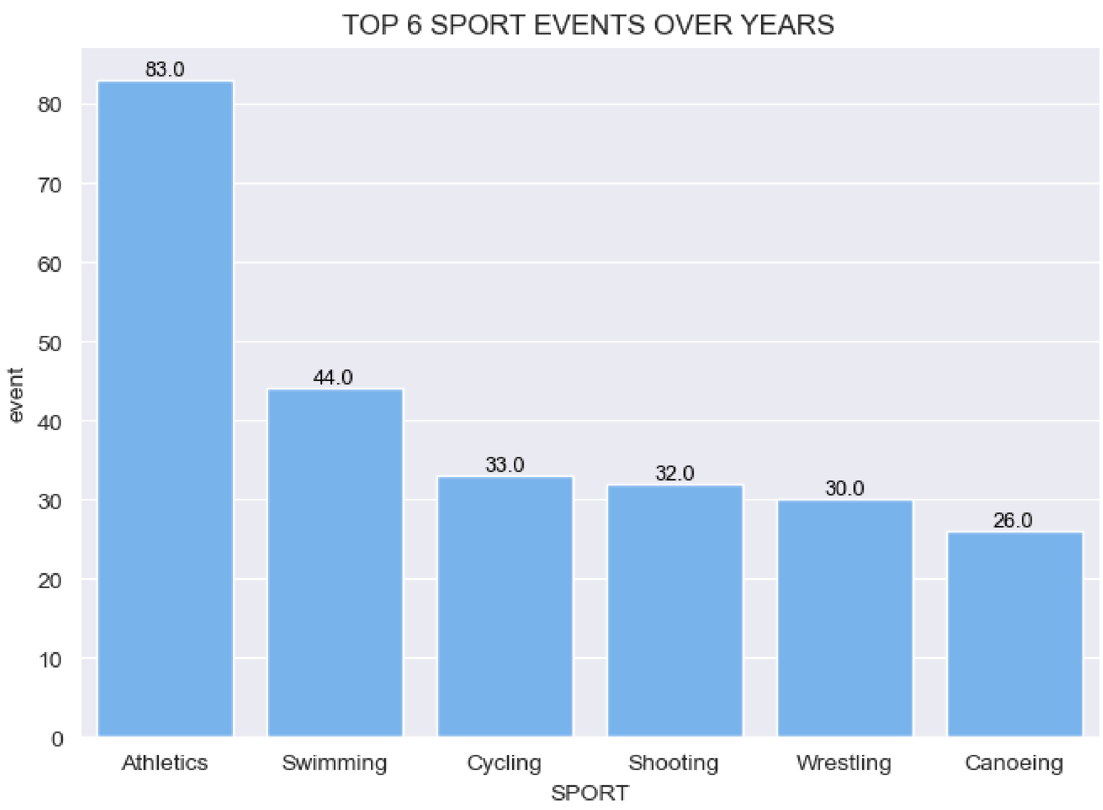


- **The number of sports events has increased over time, indicating the growing diversity of competitions.**
- **The earliest years (1948, 1964, 1968) had around 20-21 sports, while later years (2000, 2004, 2008, 2016) peaked at 34 sports.**
- **There was a steady increase in events from 1972 to 1992, with significant jumps in 1984, 1988, and 1992.**
- **The trend suggests continuous expansion in the variety of sports over the years.**

[136]:
```
chartDf = df.groupby(['sport']).agg({'event': 'nunique','games': 'nunique'}).
  ↪reset_index().sort_values(by='event', ascending=False)
chartDf.head()
```

[136]:
```
          sport  event  games
3     Athletics     83     29
44     Swimming     44     28
14      Cycling     33     25
37     Shooting     32     21
55    Wrestling     30     28
```

[137]:
```python
chartParams = {
    "chartData": [
       {
            "type": "bar",  # Simple bar chart
            "xCol": "sport",  # States as the x-axis
            "yCol": ["event"],  # Loan defaults as the y-axis
            "chartTitle": "Top 6 Sport Events Over Years",
            "legend": None  # Simple bar chart without grouping
        },
    ]
}

plotTwoCharts(chartDf[:6], chartParams)
```



- **Athletics has the highest number of events (83), significantly more than any other**

sport.

- **Swimming follows with 44 events, making it the second most frequent sport.**
- **Cycling, Shooting, Wrestling, and Canoeing have event counts ranging from 26 to 33.**
- **Athletics and Swimming dominate the list, indicating their importance in major sports events.**
- **The distribution suggests a mix of endurance, precision, and strength-based sports among the top six.**

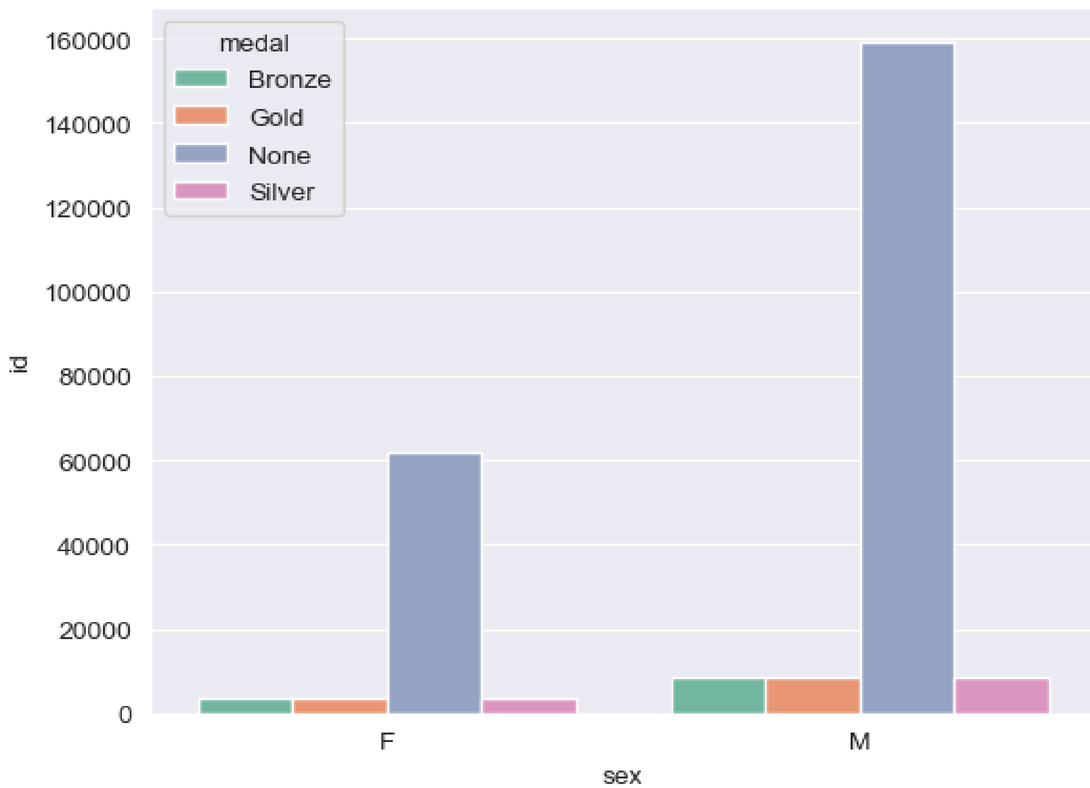
[141]:
```
chartDf = df.groupby(['sex','medal']).count().iloc[:,0].reset_index()
chartDf
```

[141]:

	sex	medal	id
0	F	Bronze	3668
1	F	Gold	3643
2	F	None	62104
3	F	Silver	3632
4	M	Bronze	8564
5	M	Gold	8423
6	M	None	159354
7	M	Silver	8258

[147]:
```
sns.barplot(x='sex', y='id', hue='medal', data=chartDf, palette='Set2',␣
↪estimator=sum)
```

[147]: <Axes: xlabel='sex', ylabel='id'>



- **Male participants are significantly higher than female participants.**
- **Most participants did not win a medal, with males dominating this category.**
- **Medal distribution among winners is relatively balanced across genders.**
- **Possible factors: participation rate, competition structure, selection criteria.**

[166]:
```
chartDf = df.groupby(['medal']).agg({
    'age': ['min', 'max', 'mean',
            lambda x: x.quantile(0.25),  # 25th percentile
            lambda x: x.quantile(0.50),  # 50th percentile (median)
            lambda x: x.quantile(0.75)],  # 75th percentile,
    'height': ['min', 'max', 'mean',
            lambda x: x.quantile(0.25),  # 25th percentile
            lambda x: x.quantile(0.50),  # 50th percentile (median)
            lambda x: x.quantile(0.75)],  # 75th percentile,
    'weight': ['min', 'max', 'mean',
            lambda x: x.quantile(0.25),  # 25th percentile
            lambda x: x.quantile(0.50),  # 50th percentile (median)
            lambda x: x.quantile(0.75)]  # 75th percentile
}).reset_index()
chartDf
```

[166]:
```
    medal   age                                                     height  \
            min   max       mean <lambda_0> <lambda_1> <lambda_2>    min
0  Bronze  10.0  63.0  25.573741       22.0       25.0       28.0  136.0
1    Gold  13.0  63.0  25.519973       22.0       25.0       28.0  136.0
2    None  11.0  96.0  25.261770       21.0       24.0       28.0  127.0
3  Silver  13.0  66.0  25.597477       22.0       25.0       28.0  136.0

                                                     weight         \
     max        mean <lambda_0> <lambda_1> <lambda_2>    min    max
0  223.0  176.987901      170.0      177.0      183.0   28.0  182.0
1  223.0  177.461959      170.0      177.0      184.0   28.0  170.0
2  226.0  174.732785      168.0      175.0      181.0   25.0  214.0
3  223.0  177.104121      170.0      177.0      184.0   30.0  167.0


        mean <lambda_0> <lambda_1> <lambda_2>
0  73.445866       64.0      72.25       82.0
1  73.956033       64.0      73.00       82.0
2  70.339304       61.0      70.00       78.0
3  73.514340       64.0      73.00       82.0
```

[167]:
```python
chartDf.columns = ['medal',
              'age_min', 'age_max', 'age_mean', 'age_25th', 'age_median',␣
 ↪'age_75th',
              'height_min', 'height_max', 'height_mean', 'height_25th',␣
 ↪'height_median', 'height_75th',
              'weight_min', 'weight_max', 'weight_mean', 'weight_25th',␣
 ↪'weight_median', 'weight_75th']
chartDf.columns
```

[167]:
```
Index(['medal', 'age_min', 'age_max', 'age_mean', 'age_25th', 'age_median',
       'age_75th', 'height_min', 'height_max', 'height_mean', 'height_25th',
       'height_median', 'height_75th', 'weight_min', 'weight_max',
       'weight_mean', 'weight_25th', 'weight_median', 'weight_75th'],
      dtype='object')
```

[168]:
```python
chartDf
```

[168]:
```
    medal  age_min  age_max   age_mean  age_25th  age_median  age_75th  \
0  Bronze     10.0     63.0  25.573741      22.0        25.0      28.0
1    Gold     13.0     63.0  25.519973      22.0        25.0      28.0
2    None     11.0     96.0  25.261770      21.0        24.0      28.0
3  Silver     13.0     66.0  25.597477      22.0        25.0      28.0

   height_min  height_max  height_mean  height_25th  height_median  \
0       136.0       223.0   176.987901        170.0          177.0
1       136.0       223.0   177.461959        170.0          177.0
```

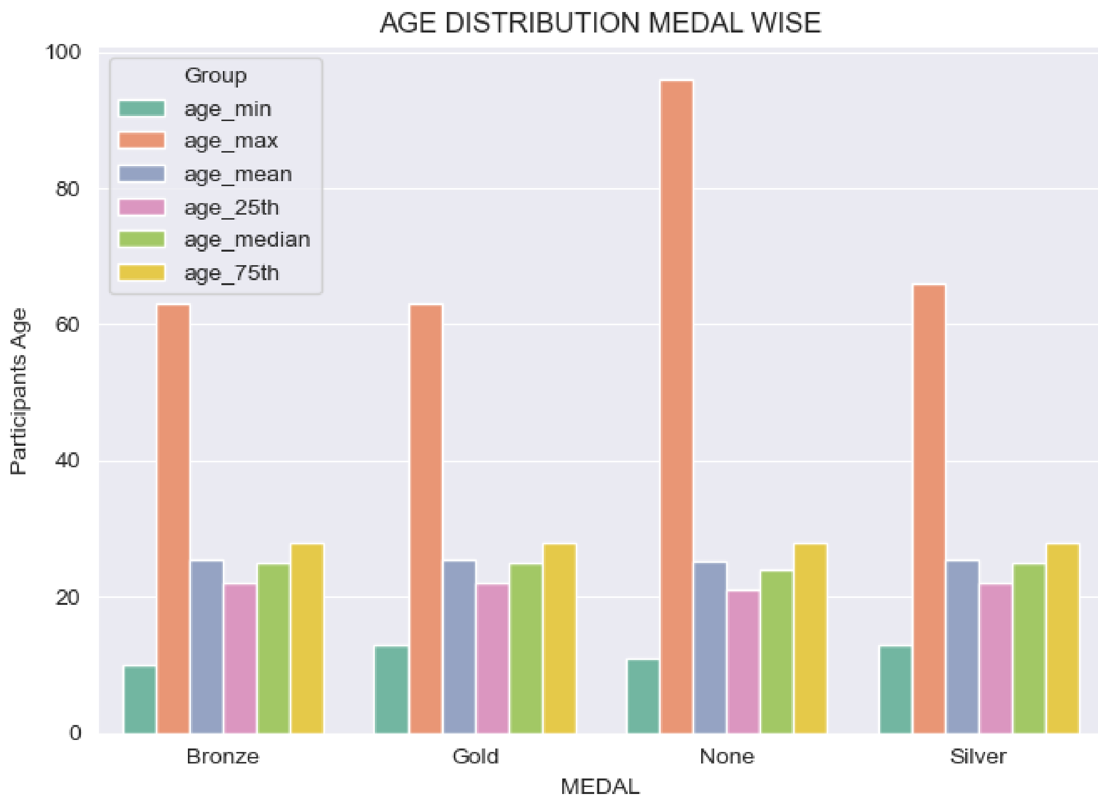
```
2       127.0       226.0   174.732785       168.0          175.0
3       136.0       223.0   177.104121       170.0          177.0

   height_75th  weight_min  weight_max  weight_mean  weight_25th  \
0        183.0        28.0       182.0    73.445866         64.0
1        184.0        28.0       170.0    73.956033         64.0
2        181.0        25.0       214.0    70.339304         61.0
3        184.0        30.0       167.0    73.514340         64.0

   weight_median  weight_75th
0          72.25         82.0
1          73.00         82.0
2          70.00         78.0
3          73.00         82.0
```

[ ]:

```

```

[169]:

```
chartParams = {
    "chartData": [
      {
            "type": "bar",  # Simple bar chart
            "xCol": "medal",  # States as the x-axis
            "yCol":␣
 ↪["age_min","age_max","age_mean","age_25th","age_median","age_75th"],  # Loan␣
 ↪defaults as the y-axis
            "chartTitle": "Age Distribution Medal Wise",
            "ylabel":"Participants Age",
            "legend":␣
 ↪["age_min","age_max","age_mean","age_25th","age_median","age_75th"]  #␣
 ↪Simple bar chart without grouping
        },
    ]
}

plotTwoCharts(chartDf, chartParams)
```



- **Age distribution is similar across medal categories, with minor variations.**
- **The maximum age of participants is significantly higher for those without a medal.**
- **Median, mean, and quartiles remain consistent, suggesting experience plays a role.**
- **Younger participants are present in all medal categories, but older participants are more frequent in non-medal groups.**

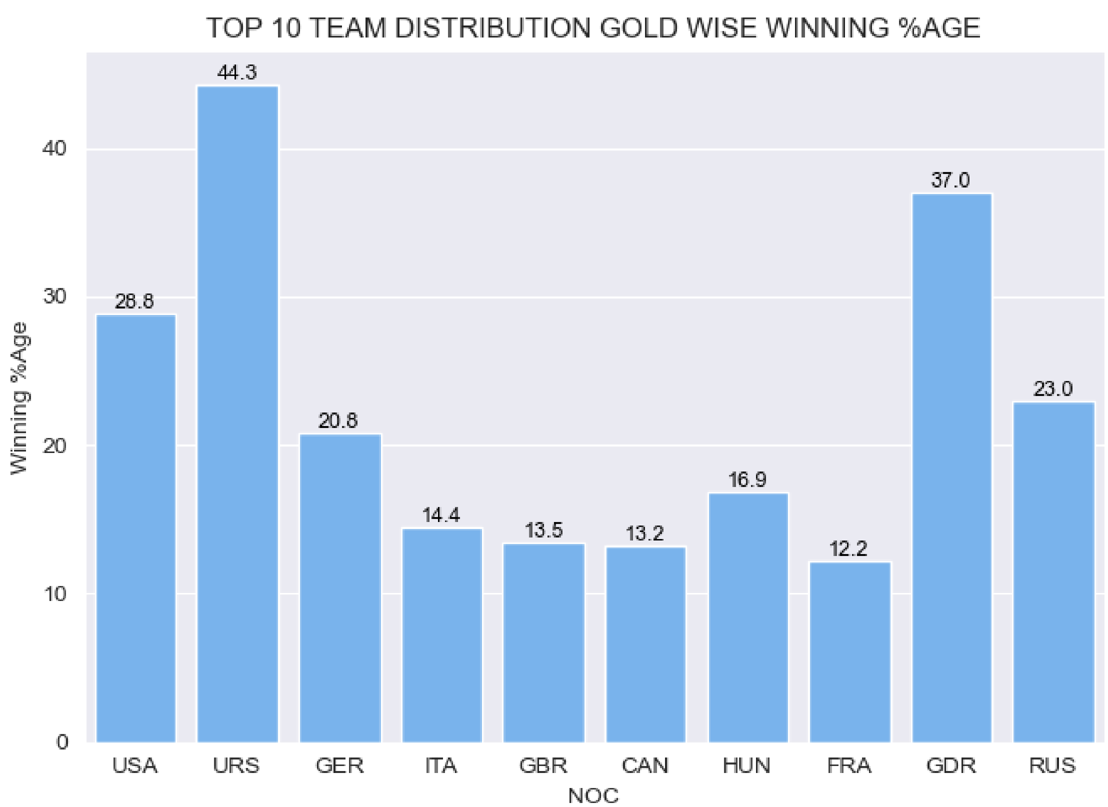
[26]:
```
dfmedal = pd.get_dummies(df[df['medal'].notnull()], columns=['medal'])
```

[27]:
```
dfmedal.loc[dfmedal['medal_Bronze'] == False,'medal_Bronze'] = 0
dfmedal.loc[dfmedal['medal_Bronze'] == True,'medal_Bronze'] = 1
dfmedal.loc[dfmedal['medal_Gold'] == False,'medal_Gold'] = 0
dfmedal.loc[dfmedal['medal_Gold'] == True,'medal_Gold'] = 1
dfmedal.loc[dfmedal['medal_None'] == False,'medal_None'] = 0
dfmedal.loc[dfmedal['medal_None'] == True,'medal_None'] = 1
dfmedal.loc[dfmedal['medal_Silver'] == False,'medal_Silver'] = 0
dfmedal.loc[dfmedal['medal_Silver'] == True,'medal_Silver'] = 1
```

[162]:
```
chartDf = dfmedal.groupby(['team', 'noc'])[['medal_Bronze', 'medal_Silver',␣
↪'medal_Gold', 'medal_None']].sum().reset_index().
↪sort_values(by='medal_Gold',ascending=False)
```

```
chartDf['win_percent'] =␣
↪((chartDf['medal_Bronze']+chartDf['medal_Silver']+chartDf['medal_Gold'])/
↪(chartDf['medal_Bronze']+chartDf['medal_Silver']+chartDf['medal_Gold']+chartDf['medal_None'
↪* 100
chartDf = chartDf.sort_values(by=['medal_Gold','win_percent'],ascending=False).
↪head(10)
```

[164]:
```
chartParams = {
    "chartData": [
       {
            "type": "bar",  # Simple bar chart
            "xCol": "noc",  # States as the x-axis
            "yCol": ["win_percent"],  # Loan defaults as the y-axis
            "chartTitle": "Top 10 Team Distribution Gold Wise Winning %age",
            "ylabel":"Winning %Age",
        },
    ]
}

plotTwoCharts(chartDf, chartParams)
```



- The USSR (URS) leads with the highest gold-winning percentage (44.3%).
- The USA follows with a significant 28.8% win rate.
- East Germany (GDR) also has a strong presence with 37.0%.
- Other countries show competitive but lower winning percentages, with Germany (GER) at 20.8% and Russia (RUS) at 23.0%.
- The distribution indicates historical dominance by a few nations in Olympic gold medal achievements.

[ ]:
```
df.columns
```

[172]:
```
chartDf = dfmedal.groupby(['sport'])[['medal_Bronze', 'medal_Silver',␣
↪'medal_Gold', 'medal_None']].sum().reset_index().
↪sort_values(by='medal_Gold',ascending=False)
chartDf['eng_percen'] =␣
↪((chartDf['medal_Bronze']+chartDf['medal_Silver']+chartDf['medal_Gold']+chartDf['medal_None
↪(chartDf['medal_Bronze'].sum()+chartDf['medal_Silver'].
↪sum()+chartDf['medal_Gold'].sum()+chartDf['medal_None'].sum()))*100
```

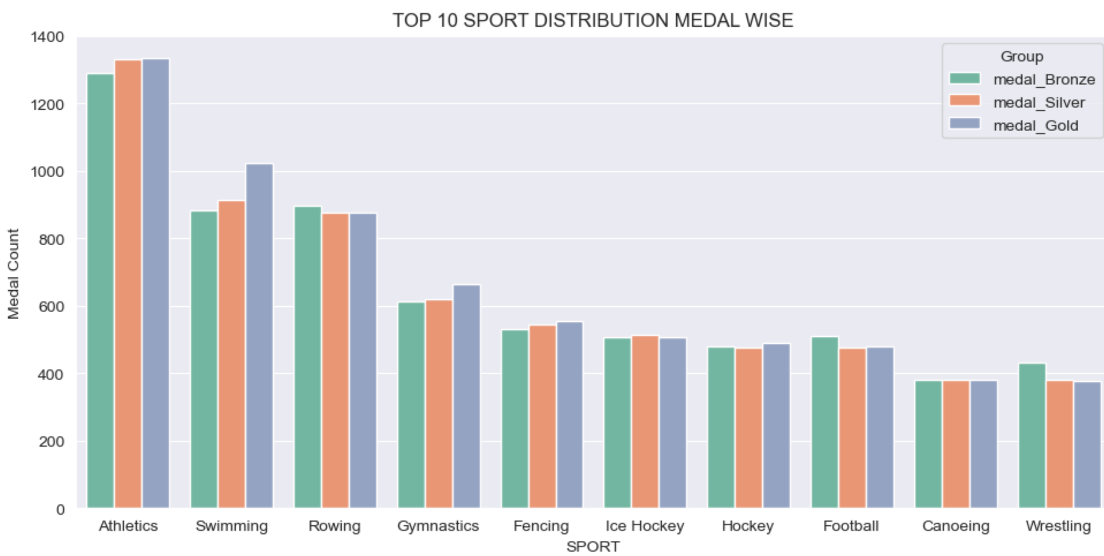
[183]:
```
chartDf = chartDf.sort_values(by=['medal_Gold'],ascending=False).head(10)
```

[184]:
```
chartDf
```

[184]:
```
          sport  medal_Bronze  medal_Silver  medal_Gold  medal_None  eng_percen  \
3    Athletics          1291          1330        1335       34639   14.979856
44    Swimming           882           913        1022       19376    8.613757
33      Rowing           897           878         877        7361     3.88634
22  Gymnastics           612           621         665       23710    9.939219
17     Fencing           532           544         555        8454    3.914285
25  Ice Hockey           507           515         508        3986    2.140922
24      Hockey           479           475         490        3785    2.029529
19    Football           509           478         479        5163     2.57291
11    Canoeing           380           379         379        4948    2.362156
55   Wrestling           432           379         377        5526    2.605901

   total_medals medal_efficiency
3          3956           0.1025
44         2817         0.126932
33         2652         0.264856
22         1898         0.074117
17         1631         0.161725
25         1530         0.277375
24         1444         0.276152
19         1466         0.221149
11         1138         0.186987
55         1188         0.176944
```

[192]:

```
chartParams = {
    "chartData": [
        {
            "type": "bar",  # Simple bar chart
            "xCol": "sport",  # States as the x-axis
            "yCol": ["medal_Bronze","medal_Silver","medal_Gold"],  # Loan␣
↪defaults as the y-axis
            "chartTitle": "Top 10 Sport Distribution Medal Wise",
            "ylabel":"Medal Count",
            "legend" : ["medal_Bronze","medal_Silver","medal_Gold"]
        },
    ]
}

plotTwoCharts(chartDf, chartParams)
```



- **Athletics dominates the Olympic medal count with approximately 1,300 medals across each category (bronze, silver, gold).**
- **These sports are quite traditional in their participation, so maximum players enroll in this.**
- **Swimming and Rowing complete the top three sports, with Swimming showing notably more gold medals than other types.**
- **There's a significant drop in medal counts between the top 3 sports and the remaining 7 (Gymnastics, Fencing, Ice Hockey, Hockey, Football, Canoeing, and Wrestling).**
- **The medal distribution becomes more balanced between bronze, silver, and gold in lower-ranked sports.**

[321]:
```
dfsport = df.groupby(['sport']).agg({'year':'nunique'}).reset_index()
dfsport = dfsport.sort_values(by=['year'], ascending=[False]).
 ↪reset_index(drop=True)
dfsport.head(10)
```

[321]:

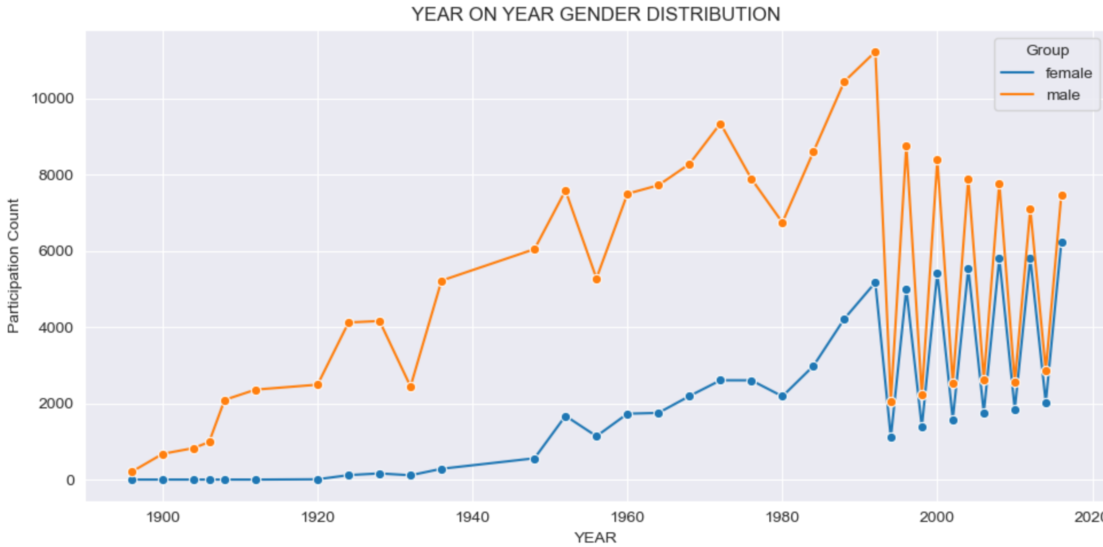
	sport	year
0	Athletics	29
1	Wrestling	28
2	Swimming	28
3	Gymnastics	28
4	Fencing	27
5	Weightlifting	26
6	Rowing	25
7	Water Polo	25
8	Boxing	25
9	Cycling	25

[328]:
```
chartdf = df.groupby('year')['sex'].value_counts().unstack().fillna(0).
 ↪reset_index()
chartdf.columns = ['year','female','male']
chartdf.tail(5)
```

[328]:

	year	female	male
30	2008	5816.0	7786.0
31	2010	1847.0	2555.0
32	2012	5815.0	7105.0
33	2014	2023.0	2868.0
34	2016	6223.0	7465.0

[330]:
```
chartParams = {
    "chartData": [
       {
            "type": "line",  # Simple bar chart
            "xCol": "year",  # States as the x-axis
            "yCol": ["female","male"],  # Loan defaults as the y-axis
            "chartTitle": "Year on Year Gender Distribution",
            "ylabel":"Participation Count",
            "legend" : ["female","male",]
        },
    ]
}

plotTwoCharts(chartdf, chartParams)
```



- **Male Dominance Historically:** Male participation in sports has consistently been higher than female participation over the years.
- **Gradual Increase in Female Participation:** Female participation started rising significantly after the 1950s, indicating improved gender inclusivity in sports.
- **Peak and Fluctuations:** Male participation saw a steady rise but fluctuated in recent years, possibly due to event-specific changes or data inconsistencies.
- **Significant Growth in Modern Era:** From the 1980s onward, female participation increased at a much faster rate, narrowing the gender gap.
- **Sharp Decline and Variations in Recent Years:** The last few data points show sharp fluctuations, which might be due to changes in event formats or incomplete data collection.
- **Gender Gap Still Exists:** While female participation has grown, it still lags behind male participation, indicating room - **for further improvement in equality.
- **Implications:** The trends highlight the impact of policy changes, societal shifts, and global efforts in promoting gender diversity in sports.

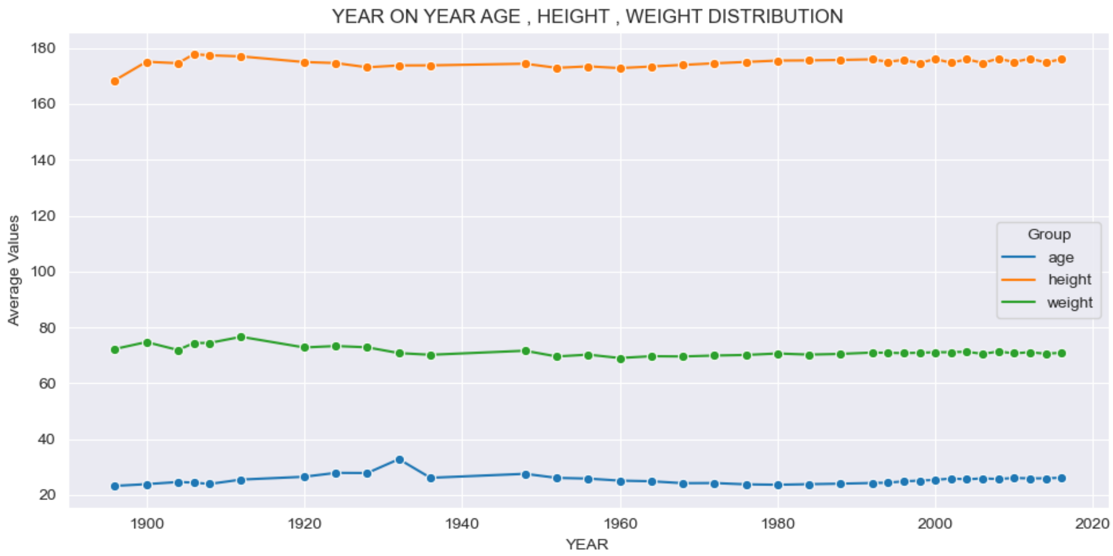
[331]:
```
df.columns
```

[331]:
```
Index(['id', 'name', 'sex', 'age', 'height', 'weight', 'team', 'noc', 'games',
       'year', 'season', 'city', 'sport', 'event', 'medal'],
      dtype='object')
```

[334]:
```
chartdf = df.groupby('year')[['age','height','weight']].mean().reset_index()
```

[336]:
```
chartParams = {
    "chartData": [
      {
          "type": "line",  # Simple bar chart
          "xCol": "year",  # States as the x-axis
          "yCol": ["age","height","weight"],  # Loan defaults as the y-axis
```

```
            "chartTitle": "Year on Year Age , Height , Weight Distribution",
            "ylabel":"Average Values",
            "legend" : ["age","height","weight"]
        },
    ]
}

plotTwoCharts(chartdf, chartParams)
```



- **Height Stability:** The average height has remained relatively stable over the years, with minor fluctuations around 170-180 cm.
- **Weight Variation:** The average weight has shown some fluctuations but has largely remained within the 60-80 kg range.
- **Age Increase and Decline:** The average age saw a gradual rise until the early 20th century, followed by a noticeable decline after 1940.
- **Steady Trends in the Late 20th Century:** After the mid-20th century, height and weight stabilized, while the age trend remained lower.
- **Potential Factors for Changes:** The variations could be due to shifts in athletic requirements, selection criteria, or improvements in training and nutrition.
- **Height and Weight Correlation:** Despite minor fluctuations, height and weight have remained within a proportional range, suggesting a balance in physical requirements.
- **Consistency in Recent Years:** From 1980 onward, all three metrics have remained relatively stable, indicating a well-established selection trend.

[338]:
```
chartdf = df.groupby('year').agg({'team':'nunique'}).reset_index()
```

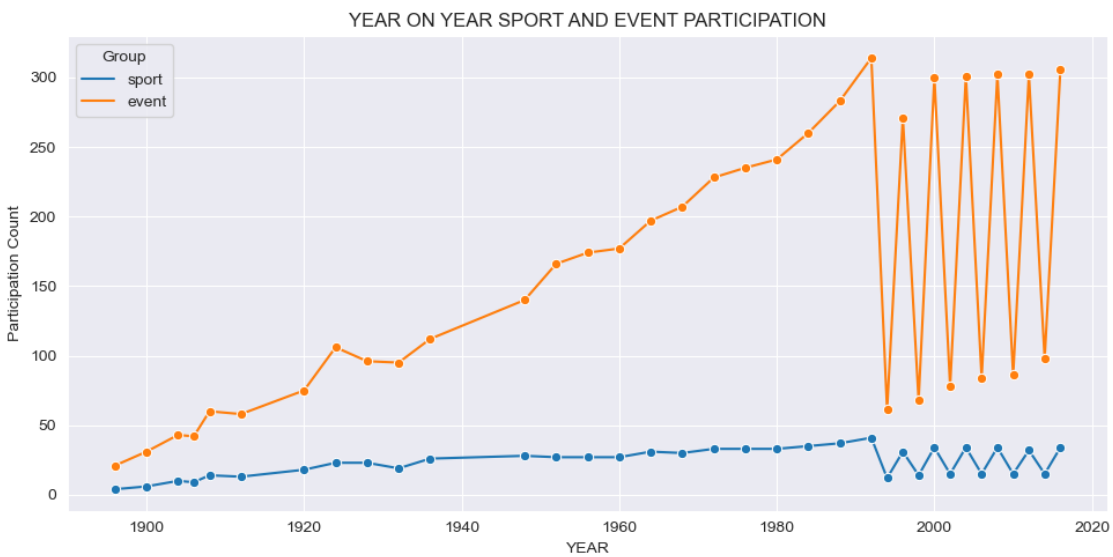
[339]:
```
chartParams = {
    "chartData": [
```

```
        {
            "type": "line",   # Simple bar chart
            "xCol": "year",   # States as the x-axis
            "yCol": ["team"],  # Loan defaults as the y-axis
            "chartTitle": "Year on Year Age Total Team Participation",
            "ylabel":"Total Teams",
         },
    ]
}

plotTwoCharts(chartdf, chartParams)
```

- **Early Growth (Pre-1940s):** The number of participating teams gradually increased, with some fluctuations, showing early adoption and expansion.
- **Post-War Stability (1940s-1960s):** There was a steady increase in participation, with a notable spike around the 1960s, indicating increased global engagement.
- **Fluctuations in the 1970s-1980s:** The number of teams fluctuated but maintained an overall upward trend, possibly due to geopolitical or economic factors.
- **Rapid Growth in the 1990s:** The participation count increased significantly, reaching over 200 teams.
- **Extreme Variability Post-2000:** The sharp up-and-down pattern suggests irregular participation cycles, possibly due to rule changes, qualification criteria, or external influences.
- **Peaks and Dips Post-2000:** The alternating high and low participation levels indicate inconsistencies, possibly due to qualification processes or event-based fluctuations.

[342]:
```
chartdf = df.groupby('year').agg({'sport':'nunique','event':'nunique'}).
↪reset_index()
```

[348]:
```
chartParams = {
    "chartData": [
       {
            "type": "line",  # Simple bar chart
            "xCol": "year",  # States as the x-axis
            "yCol": ["sport","event"],  # Loan defaults as the y-axis
            "chartTitle": "Year on Year Sport and Event Participation",
            "ylabel":"Participation Count",
            "legend" : ["sport","event"]
        },
    ]
}

plotTwoCharts(chartdf, chartParams)
```



- **Overall Growth Trend** – The participation in both sports and events has shown a steady increase from the early 1900s to the 1990s, indicating a growing interest and inclusion in competitive activities.
- **Introduction of Winter Olympics (1990s)** – The sharp fluctuations from the 1990s onward are due to the inclusion of Winter Olympic Games, which alternate with the Summer Olympics every two years, leading to varying participation counts.
- **Event Participation vs. Sport Participation** – The number of events has grown significantly compared to the number of sports, highlighting an expansion in the variety of competitions rather than entirely new sports being introduced.
- **Peak in the Late 20th Century** – A notable rise in participation occurred from the 1950s to 1980s, likely due to global expansion, increased accessibility, and inclusion of more nations in the Olympics.
- **Stability in Sports Participation** – Unlike event participation, the number of sports has remained relatively stable, suggesting that while event formats may change, core sports

remain consistent.

- **Sharp Decline and Recovery Post-1990s** – The fluctuations observed post-1990 reflect the alternating nature of the Summer and Winter Olympics, causing participation counts to drop and spike every two years.
- **Impact of Modern Olympics Expansion** – Post-2000, the Olympics have seen restructuring in event formats, leading to more dynamic participation trends, particularly with the inclusion of new disciplines within existing sports.

[ ]:

```

```