Olympic Games Performance and Impact of Population

DATA1902 Project Stage 1

SIDs

490379581
490421488

Introduction

The Olympic Games have been a universal symbol of competition and collaboration since their inception in 1896. In this project, data for Olympic athletes and their results was merged with population data from the country they represented during the year of those Olympic games. Analysis was performed on a various number of attributes and aggregate summaries were provided for country's of medal counts and best performing athletes.

Data Gathering

Three data sets downloaded from kaggle.com were used. The Olympic athlete results and National Olympic Committee 3-letter codes data sets were compiled from www.sports-reference.com using R code and then uploaded to kaggle as two separate files, both in csv format. The uploader published both data sets under the licence CC0: Public Domain, meaning there is no copyright reserved on the data.

The Olympic athlete results csv file has 271116 rows and 15 columns, with each row corresponding to one Olympic athlete's result in one event. The columns are:

1. ID - A unique identifying number for each athlete
2. Name - The athlete's name in English
3. Sex - M or F
4. Age - An integer or NA
5. Height - Height of athlete in centimeters or NA
6. Weight - Weight of athlete in kilograms or NA
7. Team - Name of the team the athlete competed for
8. NOC - National Olympic Committee 3-letter code for the athlete's country
9. Games - The year and season
10. Year - Year of the games
11. Season - Summer or Winter
12. City - Host city
13. Sport - General sport
14. Event - Specific event
15. Medal - Gold, Silver, Bronze or NA meaning the athlete did not win a medal in that event for those games

Looking at the code used to wrangle the data, the publisher has already taken many steps to account for quality issues: replacing outlying "" values for Medal with NA, converting the Age column to integer type and fixing text encoding for cities and events with different spellings. This was an encouraging sign that the data would be of a high enough quality to work with. The vast size of the data set provided ample information for statistical analysis regarding country, number of medals and which countries were best at which sports. This also means, however, that manually checking for highly obscure quality issues is impractical and could subsequently go undetected.

The National Olympic Committee 3-letter codes csv file has 230 rows and 3 columns, with each row corresponding to the unique 3-letter code for each country. The columns are:

1. NOC - 3-letter code
2. region - Name of country or NA
3. notes - Possible other name for region or other information

This data set has NA values for several regions, for example, the NOC 3-letter code of ROT, which was the Refugee Olympic Team as displayed in the notes column. This was an issue when merging data by country code, but was solved through inspection of missing values.

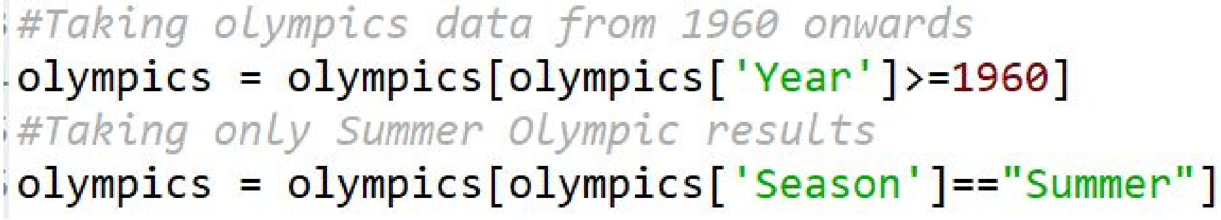
The countries population csv file has 217 rows and 62 columns. Each row represents a country in alphabetical order and the columns are:

1. Country - Name of country
2. Country code - NOC 3-letter code
3. Indicator name
4. Indicator code
5. 1960-2016 - population of country in corresponding year

For the years it accounts for, this data set is expansive and clean. This is reassuring as world/country population is tracked by several reliable parties and organisations. The completeness of this data suggests it comes from a reliable source. However, since our report only demands for the years in which the Summer Olympics were held, this creates a great amount of null values, making the analysis process of the data more mistake-prone.

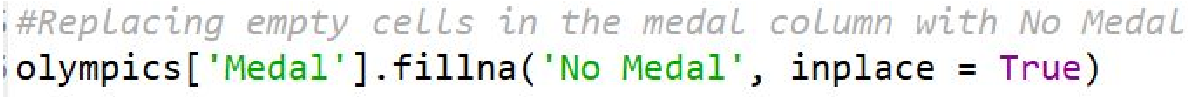
Transforming/Cleaning

Firstly, since the population data only dates back to 1960, only Olympics data from 1960 onwards was used. Only data for the Summer Olympics games was used.



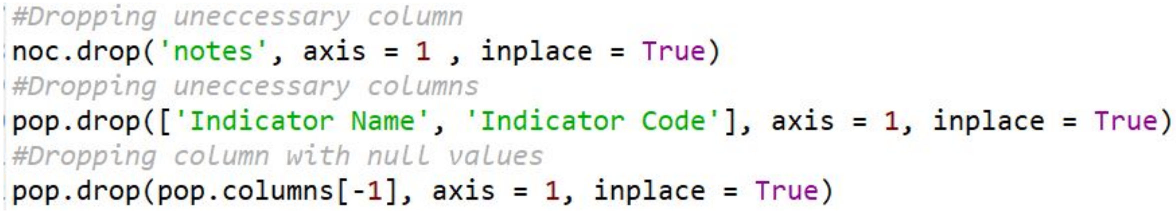
```
#Taking olympics data from 1960 onwards
olympics = olympics[olympics['Year']>=1960]
#Taking only Summer Olympic results
olympics = olympics[olympics['Season']=="Summer"]
```

Then, all NA values in the 'Medal' column was filled with a string of value 'No Medal'. This allows for aggregation and better summaries.



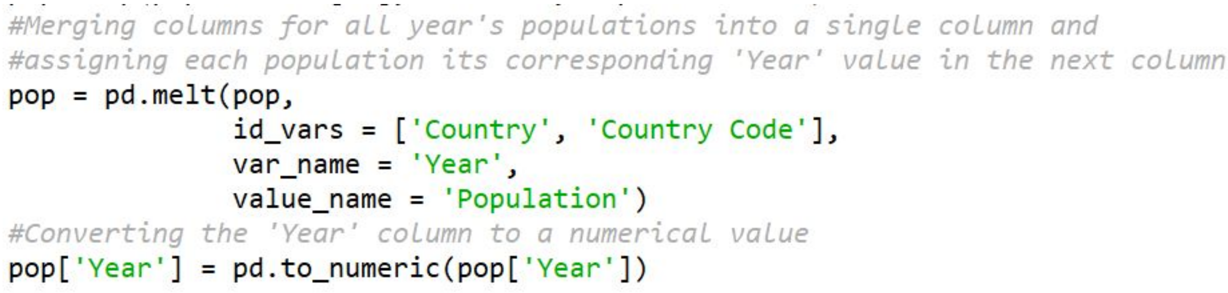
```
#Replacing empty cells in the medal column with No Medal
olympics['Medal'].fillna('No Medal', inplace = True)
```

Other simple formatting that was done prior to cleaning was dropping unnecessary columns, namely, 'notes' in NOC 3-letter codes and 'Indicator Name', 'Indicator Code' and '2017' in the population data.



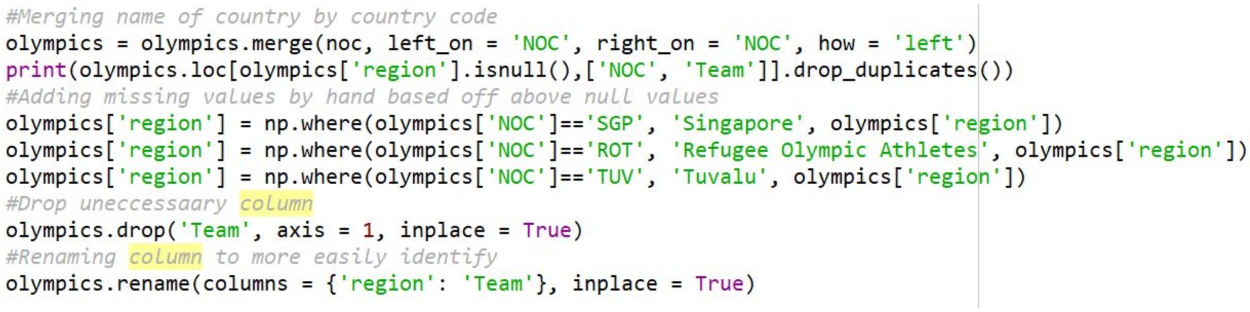
```
#Dropping uneccessary column
noc.drop('notes', axis = 1 , inplace = True)
#Dropping uneccessary columns
pop.drop(['Indicator Name', 'Indicator Code'], axis = 1, inplace = True)
#Dropping column with null values
pop.drop(pop.columns[-1], axis = 1, inplace = True)
```

The population data was also "melted" so that each population value for each year was in a single column with a corresponding 'Year' value in a new column. The 'Year' column was then converted to a number type.



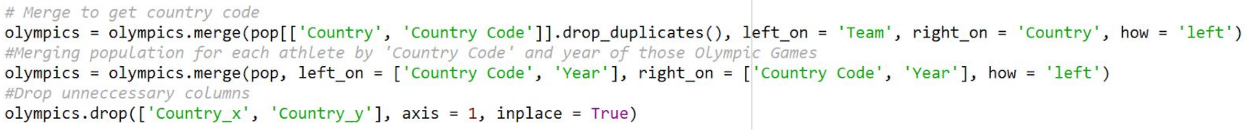
```
#Merging columns for all year's populations into a single column and
#assigning each population its corresponding 'Year' value in the next column
pop = pd.melt(pop,
              id_vars = ['Country', 'Country Code'],
              var_name = 'Year',
              value_name = 'Population')
#Converting the 'Year' column to a numerical value
pop['Year'] = pd.to_numeric(pop['Year'])
```

NOC 3-letter codes and corresponding ‘region’ values were merged in preparation to merge population data by the ‘region’ values. Leftover null values were manually inserted and unnecessary columns were again removed. The ‘region’ column was also renamed to ‘Team’.



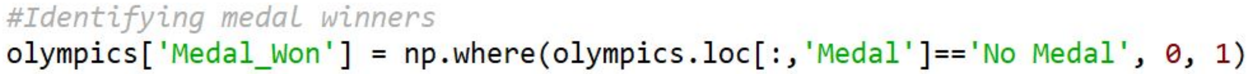
```
#Merging name of country by country code
olympics = olympics.merge(noc, left_on = 'NOC', right_on = 'NOC', how = 'left')
print(olympics.loc[olympics['region'].isnull(),['NOC', 'Team']].drop_duplicates())
#Adding missing values by hand based off above null values
olympics['region'] = np.where(olympics['NOC']=='SGP', 'Singapore', olympics['region'])
olympics['region'] = np.where(olympics['NOC']=='ROT', 'Refugee Olympic Athletes', olympics['region'])
olympics['region'] = np.where(olympics['NOC']=='TUV', 'Tuvalu', olympics['region'])
#Drop uneccessaary column
olympics.drop('Team', axis = 1, inplace = True)
#Renaming column to more easily identify
olympics.rename(columns = {'region': 'Team'}, inplace = True)
```

Population data was merged by ‘Country Code’ and ‘Year’ and then unnecessary columns were removed.



```
# Merge to get country code
olympics = olympics.merge(pop[['Country', 'Country Code']].drop_duplicates(), left_on = 'Team', right_on = 'Country', how = 'left')
#Merging population for each athlete by 'Country Code' and year of those Olympic Games
olympics = olympics.merge(pop, left_on = ['Country Code', 'Year'], right_on = ['Country Code', 'Year'], how = 'left')
#Drop unneccessary columns
olympics.drop(['Country_x', 'Country_y'], axis = 1, inplace = True)
```

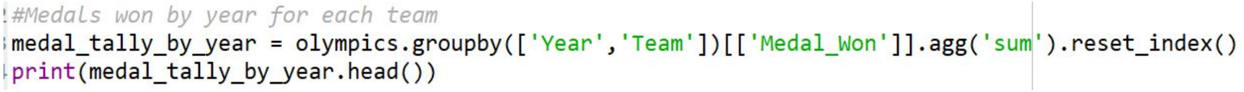
Finally, another column was added which indicated whether the athlete won a medal, represented by a 1, or not, represented by a 0. This will allow for easier code when performing analysis on total medals.



```
#Identifying medal winners
olympics['Medal_Won'] = np.where(olympics.loc[:,'Medal']=='No Medal', 0, 1)
```

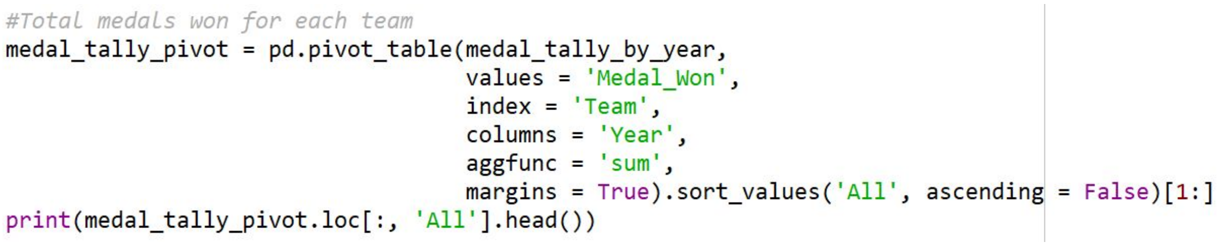
Analysis

A medal tally for each team and year was done first. This is a basic statistic and is a good indicator of how well country did comparatively to other countries for that year.



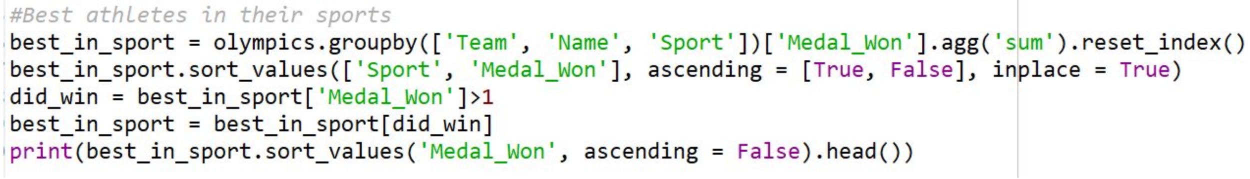
```
#Medals won by year for each team
medal_tally_by_year = olympics.groupby(['Year','Team'])[['Medal_Won']].agg('sum').reset_index()
print(medal_tally_by_year.head())
```

The total medals for each country was then tallied, which is a better statistic for how strong a country performs over a long period of time and how consistent they are.



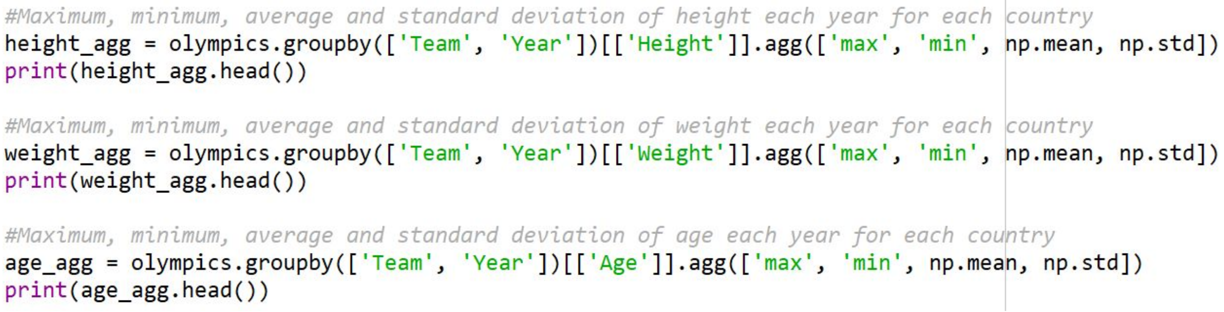
```
#Total medals won for each team
medal_tally_pivot = pd.pivot_table(medal_tally_by_year,
                                   values = 'Medal_Won',
                                   index = 'Team',
                                   columns = 'Year',
                                   aggfunc = 'sum',
                                   margins = True).sort_values('All', ascending = False)[1:]
print(medal_tally_pivot.loc[:, 'All'].head())
```

Athletes who won more than one medal in a sport were then counted and printed in descending order from most medals won to least. Michael Phelps was first, with a total of 28 medals.



```
#Best athletes in their sports
best_in_sport = olympics.groupby(['Team', 'Name', 'Sport'])['Medal_Won'].agg('sum').reset_index()
best_in_sport.sort_values(['Sport', 'Medal_Won'], ascending = [True, False], inplace = True)
did_win = best_in_sport['Medal_Won']>1
best_in_sport = best_in_sport[did_win]
print(best_in_sport.sort_values('Medal_Won', ascending = False).head())
```

The height, weight and age of each country was then analysed, calculating the maximum, minimum, average and standard deviation for each of these attributes for each country and each year.



```
#Maximum, minimum, average and standard deviation of height each year for each country
height_agg = olympics.groupby(['Team', 'Year'])[['Height']].agg(['max', 'min', np.mean, np.std])
print(height_agg.head())

#Maximum, minimum, average and standard deviation of weight each year for each country
weight_agg = olympics.groupby(['Team', 'Year'])[['Weight']].agg(['max', 'min', np.mean, np.std])
print(weight_agg.head())

#Maximum, minimum, average and standard deviation of age each year for each country
age_agg = olympics.groupby(['Team', 'Year'])[['Age']].agg(['max', 'min', np.mean, np.std])
print(age_agg.head())
```

References

Athlete Data/NOC Data
https://www.kaggle.com/heesoo37/120-years-of-olympic-history-athletes-and-results

Population Data
https://www.kaggle.com/centurion1986/countries-population