PES UNIVERSITY

COMPUTER SCIENCE ENGINEERING

STATISTICS FOR DATA SCIENCE

Datathon-A Hackathon on Data Analysis-Statistics for Data Science

Exploratory Data Analysis

NAME: CHANDAN KUMAR.S

SRN: PES2UG20CS804

SECTION: J

DATASET: 6.CSV, SET-106 (OLYMPICS DATASET)

What is Exploratory Data Analysis (EDA)?

Exploratory Data Analysis (EDA) helps to answer all these questions, ensuring the best outcomes for the project. It is an approach for summarizing, visualizing, and becoming intimately familiar with the important characteristics of a data set. Exploratory Data Analysis (EDA) is the first step in your data analysis process. Here, you make sense of the data you have and then figure out what questions you want to ask and how to frame them, as well as how best to manipulate your available data sources to get the answers you need.

1. Olympic Dataset (History of Olympics)

This dataset contains information about Olympic medalists throughout history (upto 2016). The athletes, the events they participated in and all other relevant data is present.

Data Dictionary

Column	Description
ID	Unique number for each athlete
Name	Athlete's name
Sex	Male or Female
Age	Integer
Height	In centimeters
Weight	In kilograms
Team	Team Name
NOC	National Olympic Committee 3-letter code
Games	Year and season
Year	Integer
Season	Summer or Winter
City	Host city
Sport	Sport
Event	Event
Medal	Gold, Silver or Bronze

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Share

Run All Code Draft Session (10m)

[70]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.graph_objects as go
```

```
data=pd.read_csv("../input/olympic-dataset/6.csv")
data
```

[71]:

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal
0	67	Mariya Vasilyevna Abakumova (-Tarabina)	F	23.0	176.0	74.0	Qatar	QAT	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Javelin Throw	Silver
1	411	Gezahgne Abera	M	23.0	163.0	52.0	Turkey	TUR	2000 Summer	2000	Summer	Sydney	Athletics	Athletics Men's Marathon	Bronze
2	428	Elvan Abeylegesse	F	26.0	156.0	40.0	Germany	GER	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's 5,000 metres	Silver
3	846	Valerie Kasanita Adams-Vili (-Price)	F	24.0	190.0	114.0	Sudan	SUD	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Shot Put	Bronze
4	846	Valerie Kasanita Adams-Vili (-Price)	F	28.0	190.0	114.0	Sudan	SUD	2012 Summer	2012	Summer	London	Athletics	Athletics Women's Shot Put	Bronze

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Share

Run All Code Draft Session (14m)

+ Code + Markdown

[72]:
```
data.head()
```

[72]:

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal
0	67	Mariya Vasilyevna Abakumova (-Tarabina)	F	23.0	176.0	74.0	Qatar	QAT	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Javelin Throw	Silver
1	411	Gezahgne Abera	M	23.0	163.0	52.0	Turkey	TUR	2000 Summer	2000	Summer	Sydney	Athletics	Athletics Men's Marathon	Bronze
2	428	Elvan Abeylegesse	F	26.0	156.0	40.0	Germany	GER	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's 5,000 metres	Silver
3	846	Valerie Kasanita Adams-Vili (-Price)	F	24.0	190.0	114.0	Sudan	SUD	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Shot Put	Bronze
4	846	Valerie Kasanita Adams-Vili (-Price)	F	28.0	190.0	114.0	Sudan	SUD	2012 Summer	2012	Summer	London	Athletics	Athletics Women's Shot Put	Bronze

+ Code + Markdown

[73]:
```
data.dtypes
```

[73]:
```
ID          int64
Name       object
Sex        object
```

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Share

Run All Code Draft Session (14m)

[74]:
```
data.isnull()
```

[74]:

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1108	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1109	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1110	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1111	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1112	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False

1113 rows × 15 columns

+ Code + Markdown

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (15m)

1113 rows × 15 columns

[75]:
```
data.isnull().sum()
```

[75]:
```
ID        0
Name      0
Sex       0
Age       5
Height    2
Weight    7
Team      0
NOC       0
Games     0
Year      0
Season    0
City      0
Sport     0
Event     0
Medal     0
dtype: int64
```

+ Code + Markdown

[76]:
```
data.shape
```

[76]: (1113, 15)

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Share

Run All Code Draft Session (15m)

```
1108    189.0
1109    180.0
1110    180.0
1111    168.0
1112    168.0
Name: Height, Length: 1113, dtype: float64
```

[81]:
```
data['Height'].mean()
```

[81]: 174.35193519351935

+ Code + Markdown

[82]:
```
data['Weight'].mean()
```

[82]: 66.39783001808318

[83]:
```
data['Age'].mean()
```

[83]: 27.12184115523466

2. Clean you dataset. Remove any rows with missing data that cannot be substituted and use the mean to fill null values for numeric columns

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Share

Run All Code Draft Session (18m)

[84]:
```
data['Height'].fillna(data['Height'].mean(),inplace=True)
```

[85]:
```
data['Weight']
```

[85…]
```
0       74.0
1       52.0
2       40.0
3      114.0
4      114.0
       ...
1108   114.0
1109    94.0
1110    94.0
1111    58.0
1112    58.0
Name: Weight, Length: 1113, dtype: float64
```

[86]:
```
data['Weight'].fillna(data['Weight'].mean(),inplace=True)
```

[87]:
```
data.isnull().sum()
```

[87…]
```
ID        0
Name      0
Sex       0
Age       5
Height    0
```

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (19m)

```
data.isnull().sum()
```

[87…]
```
ID        0
Name      0
Sex       0
Age       5
Height    0
Weight    0
Team      0
NOC       0
Games     0
Year      0
Season    0
City      0
Sport     0
Event     0
Medal     0
dtype: int64
```

+ Code + Markdown

[88]:
```
data['Age'].fillna(data['Age'].mean(),inplace=True)
```

[89]:
```
data.isnull().sum()
```

[89…]
```
ID        0
Name      0
Sex       0
Age       0
Height    0
Weight    0
Team      0
NOC       0
Games     0
Year      0
```

2.Visualize the distribution of age for silver medalists

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Share

Run All Code Draft Session (21m)

```
                                Event   Medal
0        Athletics Women's Javelin Throw  Silver
2         Athletics Women's 5,000 metres  Silver
5             Athletics Women's Shot Put  Silver
7  Athletics Women's 4 x 400 metres Relay  Silver
11     Athletics Men's 400 metres Hurdles  Silver
```

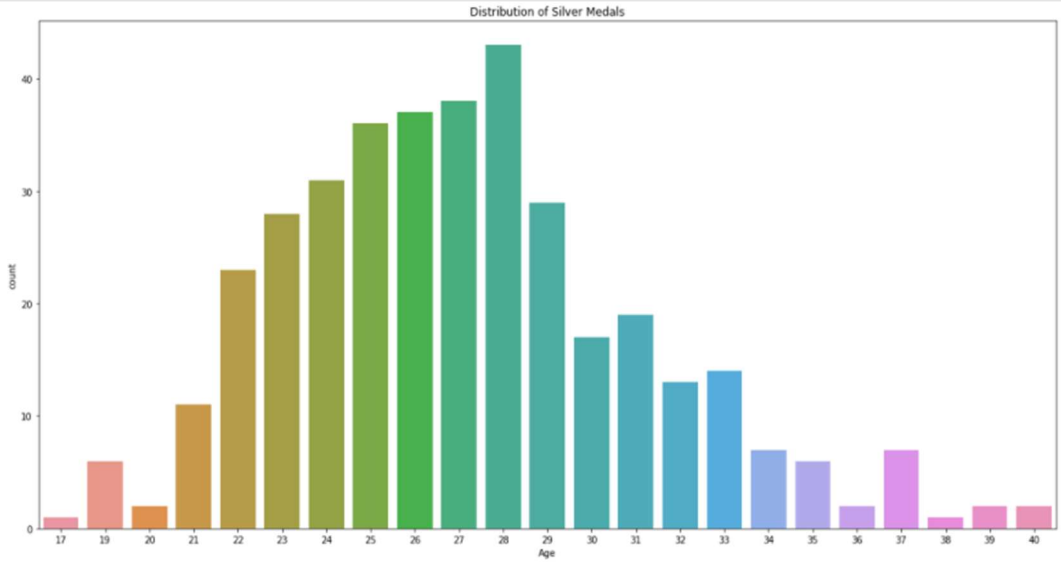
[92]:

```
plt.figure(figsize=(20,10))
plt.title("Distribution of Silver Medals")
sns.countplot(SilverMedals['Age'])
plt.show()
```

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (22m)



+ Code + Markdown

3.Create a column called BMI. Calculate BMI for each athlete.
Hint: Make sure the units match.

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (23m)

+ Code + Markdown

[93]:
```
data['BMI']=(data['Weight']/(data['Height'])**2)*10000
data
```

[93...

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal	BMI
0	67	Mariya Vasilyevna Abakumova (-Tarabina)	F	23	176.0	74.0	Qatar	QAT	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Javelin Throw	Silver	23.889463
1	411	Gezahgne Abera	M	23	163.0	52.0	Turkey	TUR	2000 Summer	2000	Summer	Sydney	Athletics	Athletics Men's Marathon	Bronze	19.571681
2	428	Elvan Abeylegesse	F	26	156.0	40.0	Germany	GER	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's 5,000 metres	Silver	16.436555
3	846	Valerie Kasanita Adams-Villi (-Price)	F	24	190.0	114.0	Sudan	SUD	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Shot Put	Bronze	31.578947
4	846	Valerie Kasanita Adams-Villi (-Price)	F	28	190.0	114.0	Sudan	SUD	2012 Summer	2012	Summer	London	Athletics	Athletics Women's Shot Put	Bronze	31.578947
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1108	135095	Szymon Zikowski	M	25	189.0	114.0	Hungary	HUN	2000 Summer	2000	Summer	Sydney	Athletics	Athletics Men's Hammer Throw	Bronze	31.914000
1109	135501	Ellina Aleksandrovna Zvereva (Kisheyeva-)	F	36	180.0	94.0	Bahrain	BRN	1996 Summer	1996	Summer	Atlanta	Athletics	Athletics Women's Discus Throw	Gold	29.012346
1110	135501	Ellina Aleksandrovna Zvereva (Kisheyeva-)	F	39	180.0	94.0	Bahrain	BRN	2000 Summer	2000	Summer	Sydney	Athletics	Athletics Women's Discus Throw	Bronze	29.012346
1111	135563	Olesya Nikolayevna Zykina	F	20	168.0	58.0	Qatar	QAT	2000 Summer	2000	Summer	Sydney	Athletics	Athletics Women's 4 x 400 metres Relay	Gold	20.549887
1112	135563	Olesya Nikolayevna Zykina	F	24	168.0	58.0	Qatar	QAT	2004 Summer	2004	Summer	Athina	Athletics	Athletics Women's 4 x 400 metres Relay	Silver	20.549887

1113 rows × 16 columns

+ Code + Markdown

4.Generate a scatter plot for the athletes' height vs weight. State if there is a positive or negative correlation.

olympic dataset PES2UG20CS804 Draft saved

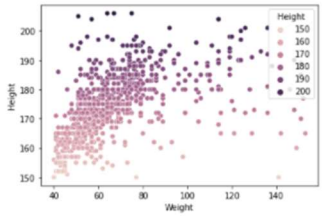
File Edit View Run Add-ons Help

Run All Code Draft Session (25m)

+ Code + Markdown

[94]:
```
print("Scatterplot of height and weight")
sns.scatterplot(data=data,x='Weight', y='Height', hue='Height')
```

[94...
```
Scatterplot of height and weight
<AxesSubplot:xlabel='Weight', ylabel='Height'>
```



+ Code + Markdown

[95]:
```
data.groupby(['Sport'])['Sex'].value_counts().plot.bar(figsize=(15,5))
plt.ylabel('Sex')
plt.title('Visualization of Gender Distribution')
```

[95...
```
Text(0.5, 1.0, 'Visualization of Gender Distribution')
```

Visualization of Gender Distribution

5.Visualize the gender distribution for each sport that you have been assigned over the last 5 years.
Hint: If any sport has under 5 years worth of information, ignore it.

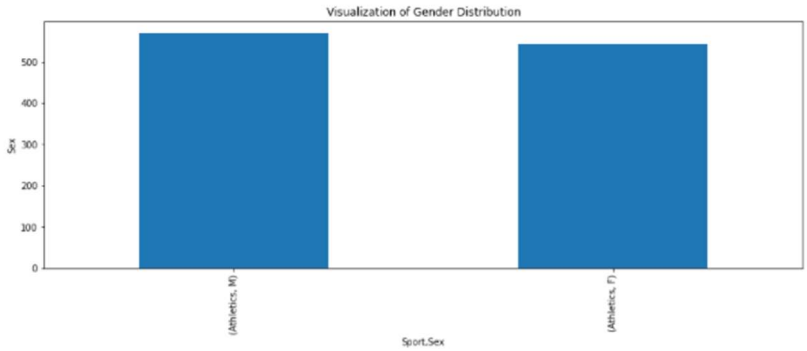
olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (27m)

[95]:
```
data.groupby(['Sport'])['Sex'].value_counts().plot.bar(figsize=(15,5))
plt.ylabel('Sex')
plt.title('Visualization of Gender Distribution')
```

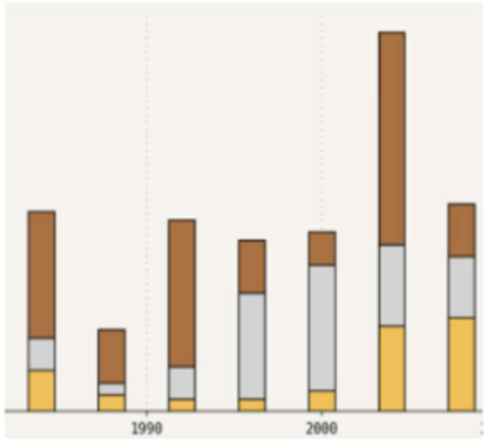
[95… Text(0.5, 1.0, 'Visualization of Gender Distribution')



+ Code + Markdown

[96]:
```
import plotly.graph_objects as go
dt = data[data.Sport=="Athletics"]
dt.head()
```

1. Split your dataset based on the sports you've been assigned. Identify the team that has the maximum years of participation. If there are multiple teams in this category select the team with the highest medal count. Create a stacked bar plot for this Team to count the medals won each year won while differentiating between the different types of medals. Your graph should look similar to:



Hint: Generate a stacked bar plot for each split/sport

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (28m)

+ Code + Markdown

[96]:
```
import plotly.graph_objects as go
dt = data[data.Sport=="Athletics"]
dt.head()
```

[96…

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal	BMI
0	67	Mariya Vasilyevna Abakumova (-Tarabina)	F	23	176.0	74.0	Qatar	QAT	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Javelin Throw	Silver	23.889463
1	411	Gezahgne Abera	M	23	163.0	52.0	Turkey	TUR	2000 Summer	2000	Summer	Sydney	Athletics	Athletics Men's Marathon	Bronze	19.571681
2	428	Elvan Abeylegesse	F	26	156.0	40.0	Germany	GER	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's 5,000 metres	Silver	16.436555
3	846	Valerie Kasanita Adams-Vili (-Price)	F	24	190.0	114.0	Sudan	SUD	2008 Summer	2008	Summer	Beijing	Athletics	Athletics Women's Shot Put	Bronze	31.578947
4	846	Valerie Kasanita Adams-Vili (-Price)	F	28	190.0	114.0	Sudan	SUD	2012 Summer	2012	Summer	London	Athletics	Athletics Women's Shot Put	Bronze	31.578947

+ Code + Markdown

[97]:
```
import plotly.graph_objects as go
#Selecting rows with sport as "Athletics"
dt = data[data.Sport=="Athletics"]
dt.head()
team = dt.Team.unique().tolist()
print(team)
#To generate stacked bar plot
fig = go.Figure(layout=dict(barmode='stack'))

fig.add_bar(name="Gold", x=team, y=data[data.Medal == "Gold"].Team
            .value_counts().reindex(team), marker_color="gold")

fig.add_bar(name="Silver", x=team, y=data[data.Medal == "Silver"].Team
```

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (29m)

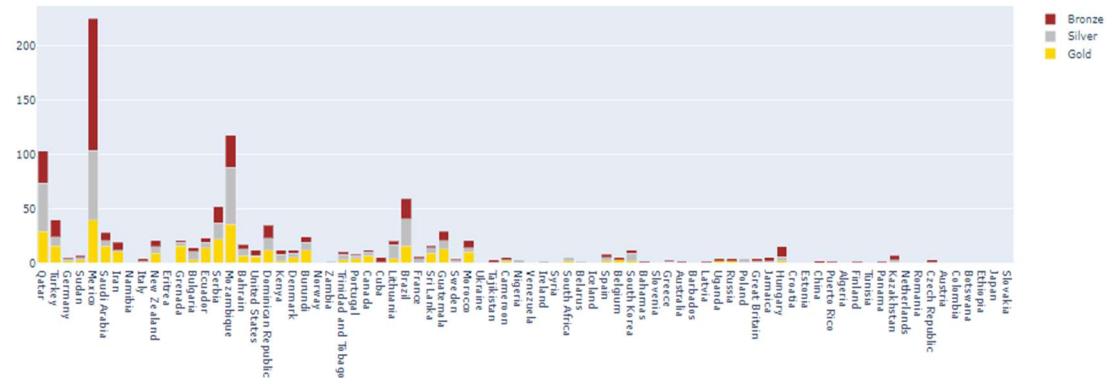
```
import plotly.graph_objects as go
#Selecting rows with sport as "Athletics"
dt = data[data.Sport=="Athletics"]
dt.head()
team = dt.Team.unique().tolist()
print(team)
#To generate stacked bar plot
fig = go.Figure(layout=dict(barmode='stack'))

fig.add_bar(name="Gold", x=team, y=data[data.Medal == "Gold"].Team
            .value_counts().reindex(team), marker_color="gold")

fig.add_bar(name="Silver", x=team, y=data[data.Medal == "Silver"].Team
            .value_counts().reindex(team), marker_color="silver")

fig.add_bar(name="Bronze", x=team, y=data[data.Medal == "Bronze"].Team
            .value_counts().reindex(team), marker_color="brown")

fig.show()
```

```
['Qatar', 'Turkey', 'Germany', 'Sudan', 'Mexico', 'Saudi Arabia', 'Iran', 'Namibia', 'Italy', 'New Zealand', 'Eritrea', 'Grenada', 'Bulgaria', 'Ecuador', 'Serbia', 'Mozambique', 'Bahrain',
'United States', 'Dominican Republic', 'Kenya', 'Denmark', 'Burundi', 'Norway', 'Zambia', 'Trinidad and Tobago', 'Portugal', 'Canada', 'Cuba', 'Lithuania', 'Brazil', 'France', 'Sri Lanka',
'Guatemala', 'Sweden', 'Morocco', 'Ukraine', 'Tajikistan', 'Cameroon', 'Nigeria', 'Venezuela', 'Ireland', 'Syria', 'South Africa', 'Belarus', 'Iceland', 'Spain', 'Belgium', 'South Korea',
'Bahamas', 'Slovenia', 'Greece', 'Australia', 'Barbados', 'Latvia', 'Uganda', 'Russia', 'Poland', 'Great Britain', 'Jamaica', 'Hungary', 'Croatia', 'Estonia', 'China', 'Puerto Rico', 'Alge
ria', 'Finland', 'Tunisia', 'Panama', 'Kazakhstan', 'Netherlands', 'Romania', 'Czech Republic', 'Austria', 'Colombia', 'Botswana', 'Ethiopia', 'Japan', 'Slovakia']
```

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (29m)



2.Generate a new dataset for all the athletes in your original dataset. Create a new attribute called medal frequency and find the most successful athlete in your dataset.
Hint: Success is calculated by (medals won)/(years of participation).

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (31m)

+ Code + Markdown

[98]:
```
team = dt.Team.unique().tolist()
print(team)
```

```
['Qatar', 'Turkey', 'Germany', 'Sudan', 'Mexico', 'Saudi Arabia', 'Iran', 'Namibia', 'Italy', 'New Zealand', 'Eritrea', 'Grenada', 'Bulgaria', 'Ecuador', 'Serbia', 'Mozambique', 'Bahrain',
'United States', 'Dominican Republic', 'Kenya', 'Denmark', 'Burundi', 'Norway', 'Zambia', 'Trinidad and Tobago', 'Portugal', 'Canada', 'Cuba', 'Lithuania', 'Brazil', 'France', 'Sri Lanka',
'Guatemala', 'Sweden', 'Morocco', 'Ukraine', 'Tajikistan', 'Cameroon', 'Nigeria', 'Venezuela', 'Ireland', 'Syria', 'South Africa', 'Belarus', 'Iceland', 'Spain', 'Belgium', 'South Korea',
'Bahamas', 'Slovenia', 'Greece', 'Australia', 'Barbados', 'Latvia', 'Uganda', 'Russia', 'Poland', 'Great Britain', 'Jamaica', 'Hungary', 'Croatia', 'Estonia', 'China', 'Puerto Rico', 'Alge
ria', 'Finland', 'Tunisia', 'Panama', 'Kazakhstan', 'Netherlands', 'Romania', 'Czech Republic', 'Austria', 'Colombia', 'Botswana', 'Ethiopia', 'Japan', 'Slovakia']
```

```
athletes=data['Name'].value_counts()
athletesmdf=pd.DataFrame(athletes)
athletesmdf.iloc[0:]
```

[99…

	Name
Allyson Michelle Felix	9
Veronica Angella Campbell-Brown	8
Usain St. Leo Bolt	8
Shelly-Ann Fraser-Pryce	6
Tirunesh Dibaba Keneni	6
...	...
Bershawn D. Jackson	1
Evan Reese Jager	1
Maryam Yusuf Jamal	1
Godday James	1
Nancy Jebet Langat	1

olympic dataset PES2UG20CS804 Draft saved

File Edit View Run Add-ons Help

Run All Code Draft Session (31m)

Nancy Jebet Langat	1

811 rows × 1 columns

```
newData=pd.DataFrame()

newData["Name"]=data['Name'].unique()
medals=[]
for i in data['Name'].unique():
    player_data=data.loc[data['Name']==i]
    medals.append(player_data.shape[0])
print(medals)
newData["Medals"]=medals


newData.to_csv("NEWDATA.csv")
```

```
[1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 4,
1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 4, 2, 1, 4, 8, 1, 1, 1, 1, 1, 1, 3, 1, 1, 2, 1, 1, 4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 2, 8, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 4,
2, 1, 1, 1, 1, 1, 2, 2, 3, 3, 1, 1, 1, 1, 3, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 3, 1, 2, 1, 1, 3, 1, 2, 1, 1, 3, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 6, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1,
2, 2, 1, 1, 3, 1, 1, 2, 1, 1, 1, 1, 1, 4, 1, 1, 9, 1, 2, 3, 1, 3, 1, 1, 1, 1, 1, 1, 1, 6, 1, 1, 2, 2, 1, 2, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 5, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1,
1, 1, 2, 1, 4, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 2, 2, 2, 1, 1, 1, 2, 3, 1, 1, 1, 1, 1, 1, 1, 3, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 2,
1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 3, 1, 1, 1, 1, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 2, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 4, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 3, 5, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 3, 1, 2, 2, 3, 1, 1, 1,
2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 3, 1, 2, 1, 1, 1, 2, 2, 1, 2, 3, 2, 1, 2, 1, 1, 2, 1, 1, 2, 3, 2, 1, 1, 1, 4, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 2,
1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 3, 2, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 1, 2, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 5, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1,
2, 1, 1, 2, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1,
3, 1, 2, 1, 2, 2, 1, 2, 1, 1, 1, 1, 3, 1, 1, 1, 1, 4, 1, 1, 1, 1, 1, 1, 4, 2, 1, 2, 1, 1, 1, 1, 3, 1, 3, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 3, 1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1,
1, 1, 1, 2, 1, 1, 2, 2, 1, 1, 3, 2, 1, 1, 2, 1, 4, 1, 1, 1, 1, 1, 2, 1, 1, 1, 2, 3, 2, 1, 4, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 2, 2]
```

+ Code + Markdown