STATISTICS FOR DATA SCIENCE

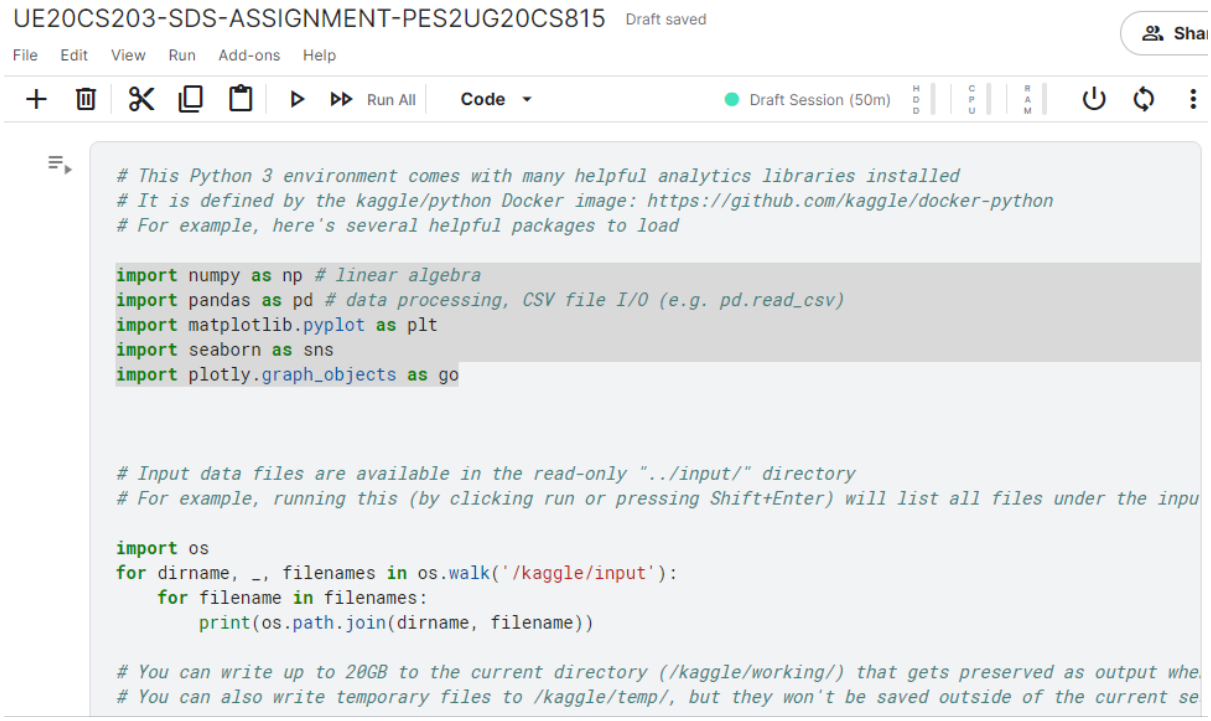
Assignment

Title :- Olympics Dataset (History of Olympics)

NAME: Vijay J	SRN: PES2UG20CS815	SECTION: J

Introductory Questions:-

Import libraries:



1.Clean your Dataset remove any rows with missing data that cannot be substituted and use the mean to fill null values for numeric columns

[93]:

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal
0	86	Jos Manuel Abascal Gmez	M	26.0	182.0	67.0	Spain	ESP	1984 Summer	1984	Summer	Los Angeles	Athletics	Athletics Men's 1,500 metres	Bronze
1	1569	Kriss Kezie Uche Chukwu Duru Akabusi	M	25.0	185.0	81.0	Great Britain	GBR	1984 Summer	1984	Summer	Los Angeles	Athletics	Athletics Men's 4 x 400 metres Relay	Silver
2	1673	John Akii-Bua	M	22.0	188.0	77.0	Uganda	UGA	1972 Summer	1972	Summer	Munich	Athletics	Athletics Men's 400 metres Hurdles	Gold
3	1732	Liudmyla Vasylivna Aksenova (Shapovalova-)	F	29.0	165.0	58.0	Soviet Union	URS	1976 Summer	1976	Summer	Montreal	Athletics	Athletics Women's 4 x 400 metres Relay	Bronze
4	1734	Aleksandr Timofeyevich Aksinin	M	21.0	173.0	67.0	Soviet Union	URS	1976 Summer	1976	Summer	Montreal	Athletics	Athletics Men's 4 x 100 metres Relay	Bronze
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1056	135313	Gyula Zsivtzky	M	27.0	190.0	102.0	Hungary	HUN	1964 Summer	1964	Summer	Tokyo	Athletics	Athletics Men's Hammer Throw	Silver
1057	135313	Gyula Zsivtzky	M	31.0	190.0	102.0	Hungary	HUN	1968 Summer	1968	Summer	Mexico City	Athletics	Athletics Men's Hammer Throw	Gold
1058	135409	Mauro Carlo Zuliani	M	20.0	175.0	62.0	Italy	ITA	1980 Summer	1980	Summer	Moskva	Athletics	Athletics Men's 4 x 400 metres Relay	Bronze
1059	135544	Krzysztof Zwoliski	M	21.0	175.0	70.0	Poland	POL	1980 Summer	1980	Summer	Moskva	Athletics	Athletics Men's 4 x 100 metres Relay	Silver
1060	135553	Galina Ivanovna Zybina (-Fyodorova)	F	33.0	168.0	80.0	Soviet Union	URS	1964 Summer	1964	Summer	Tokyo	Athletics	Athletics Women's Shot Put	Bronze

1061 rows × 15 columns

[94]:
```
data.dtypes
```

[94]:
```
ID          int64
Name       object
Sex        object
Age       float64
Height    float64
Weight    float64
Team       object
NOC        object
Games      object
Year        int64
Season     object
City       object
Sport      object
Event      object
Medal      object
dtype: object
```

+ Code + Markdown

[95]:
```
data.isnull()
```

[95]:

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1056	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1057	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1058	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1059	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1060	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False

1061 rows × 15 columns

[96]:
```
data.isnull().sum()
```

[96]:
```
ID        0
Name      0
Sex       0
Age       7
Height    7
Weight    6
Team      0
NOC       0
Games     0
Year      0
Season    0
City      0
Sport     0
Event     0
Medal     0
dtype: int64
```

+ Code + Markdown

[97]:
```
data['Weight'].mean()
```

[97]:
```
71.93364928909952
```

[98]:
```
data['Age'].mean()
```

[98]:
```
24.740037950664135
```

[99]:
```
data['Height'].mean()
```

[99]:
```
177.89278937381405
```

[100]:
```
data['Weight'].fillna(data['Weight'].mean(),inplace=True)
```

[101]:
```
data['Age'].fillna(data['Age'].mean(),inplace=True)
```

[102]:
```
data['Height'].fillna(data['Height'].mean(),inplace=True)
```

[103]:
```
data.isnull().sum()
```

[103…
```
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
dtype: int64
```

2.Visualize the distribution of age for Silver medalists

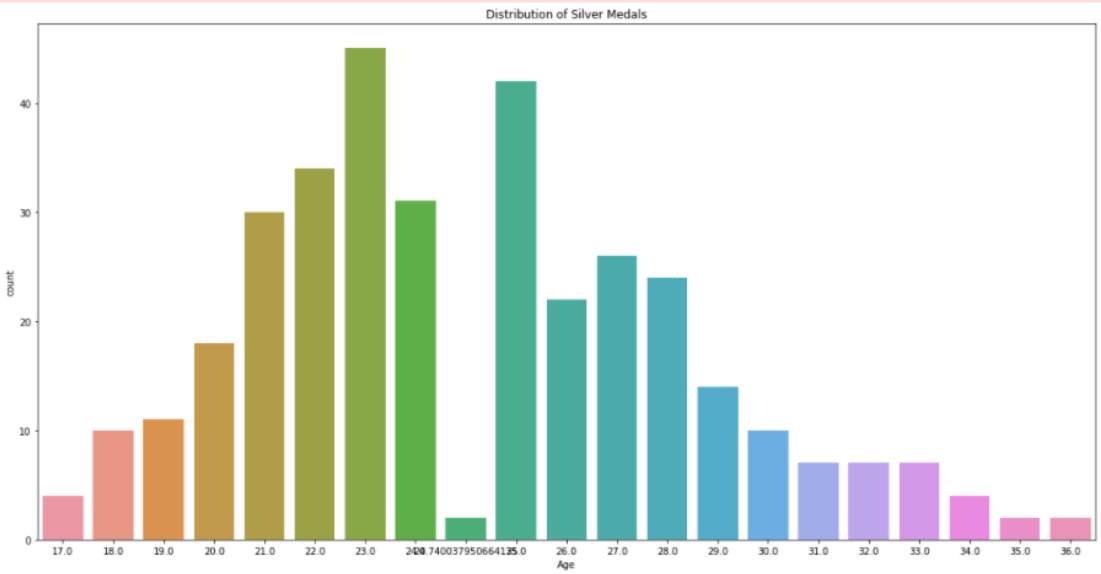
Visualization The distribution of age for Silver Medalist

+ Code + Markdown

[104]:
```
SilverMedals = data[(data.Medal == 'Silver')]
print(SilverMedals.head())
```

```
      ID                                      Name Sex   Age  Height  \
1   1569          Kriss Kezie Uche Chukwu Duru Akabusi   M  25.0   185.0
8   3788                      Grete Andersen-Waitz   F  30.0   172.0
10  4390  Tatyana Mikhaylovna Anisimova (Poluboyarova-)   F  26.0   172.0
12  4391        Vera Vasilyevna Anisimova (Mikheyeva-)   F  28.0   167.0
13  4487                          Mrta Antal-Rudas   F  27.0   164.0

    Weight           Team  NOC        Games  Year  Season         City  \
1     81.0  Great Britain  GBR  1984 Summer  1984  Summer  Los Angeles
8     53.0         Norway  NOR  1984 Summer  1984  Summer  Los Angeles
10    65.0   Soviet Union  URS  1976 Summer  1976  Summer     Montreal
12    52.0   Soviet Union  URS  1980 Summer  1980  Summer       Moskva
13    66.0        Hungary  HUN  1964 Summer  1964  Summer        Tokyo

        Sport                                  Event   Medal
1   Athletics    Athletics Men's 4 x 400 metres Relay  Silver
8   Athletics             Athletics Women's Marathon  Silver
10  Athletics    Athletics Women's 100 metres Hurdles  Silver
12  Athletics  Athletics Women's 4 x 100 metres Relay  Silver
13  Athletics         Athletics Women's Javelin Throw  Silver
```

[105]:
```
plt.figure(figsize=(20,10))
plt.title('Distribution of Silver Medals')
sns.countplot(SilverMedals['Age'])
plt.show()
```



3.Create a column called BMI . Calculate BMI for each athlete

Calculate BMI for each Athlete

[106]:
```
data['BMI']=(data['Weight']/(data['Height'])**2)*10000
data
```

[106…

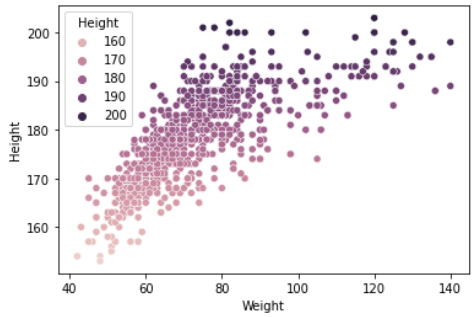
	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	Event	Medal	BMI
0	86	Jos Manuel Abascal Gmez	M	26.0	182.0	67.0	Spain	ESP	1984 Summer	1984	Summer	Los Angeles	Athletics	Athletics Men's 1,500 metres	Bronze	20.227026
1	1569	Kriss Kezie Uche Chukwu Duru Akabusi	M	25.0	185.0	81.0	Great Britain	GBR	1984 Summer	1984	Summer	Los Angeles	Athletics	Athletics Men's 4 x 400 metres Relay	Silver	23.666910
2	1673	John Akii-Bua	M	22.0	188.0	77.0	Uganda	UGA	1972 Summer	1972	Summer	Munich	Athletics	Athletics Men's 400 metres Hurdles	Gold	21.785876
1056	135313	Gyula Zsivtzky	M	27.0	190.0	102.0	Hungary	HUN	1964 Summer	1964	Summer	Tokyo	Athletics	Men's Hammer Throw	Silver	28.254848
1057	135313	Gyula Zsivtzky	M	31.0	190.0	102.0	Hungary	HUN	1968 Summer	1968	Summer	Mexico City	Athletics	Athletics Men's Hammer Throw	Gold	28.254848
1058	135409	Mauro Carlo Zuliani	M	20.0	175.0	62.0	Italy	ITA	1980 Summer	1980	Summer	Moskva	Athletics	Athletics Men's 4 x 400 metres Relay	Bronze	20.244898
1059	135544	Krzysztof Zwoliski	M	21.0	175.0	70.0	Poland	POL	1980 Summer	1980	Summer	Moskva	Athletics	Athletics Men's 4 x 100 metres Relay	Silver	22.857143
1060	135553	Galina Ivanovna Zybina (-Fyodorova)	F	33.0	168.0	80.0	Soviet Union	URS	1964 Summer	1964	Summer	Tokyo	Athletics	Athletics Women's Shot Put	Bronze	28.344671

1061 rows × 16 columns

4.Generate a scatter Plot for the atheletes' height vs weight. State if there is apostive or negative correlation

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

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

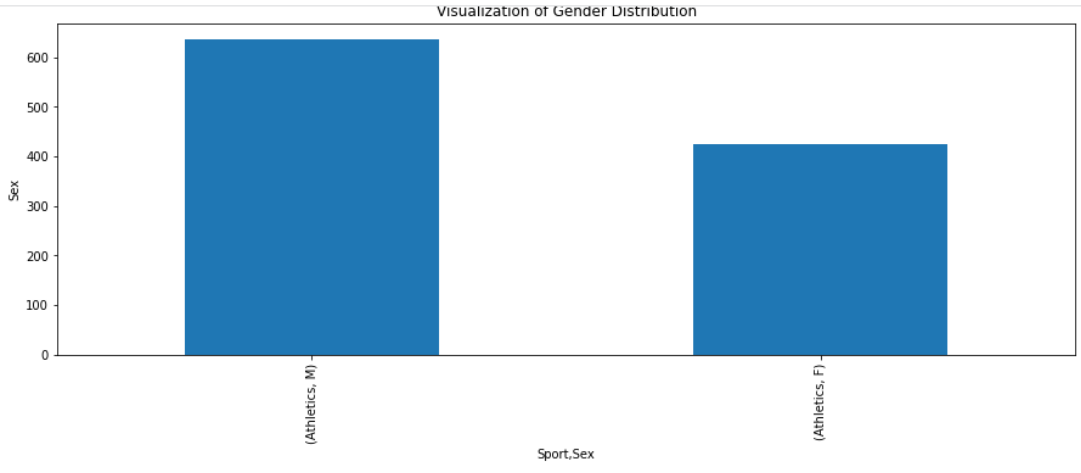


5.Visualize the gender distribution for each sport that you have been assigned over the last 5 years.

Visualize of Gender Distribution

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

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



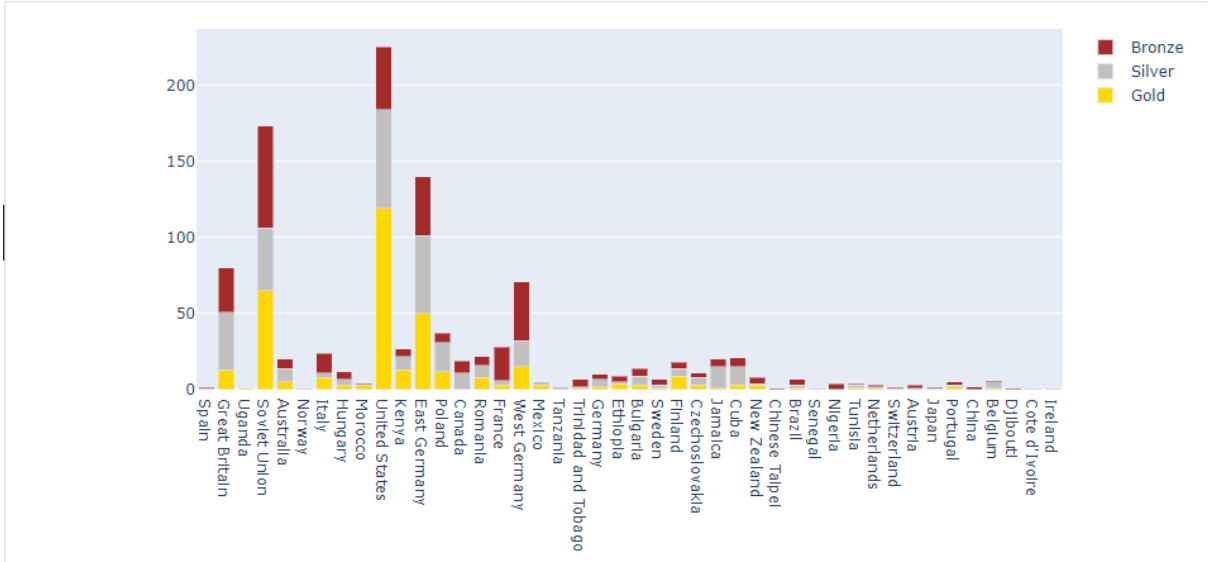
Task Questions:-

1.Create a Stacked bar plot for this team to count the medals won each year won while differentiating between the different types of medals

TASK-1 A stacked Bar Plot for the teams to count medals won while differenting between the different types of medals

[110]:
```
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_

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

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

fig.show()
```

```
['Spain', 'Great Britain', 'Uganda', 'Soviet Union', 'Australia', 'Norway', 'Italy', 'Hungary', 'Morocco', 'United States',
'Kenya', 'East Germany', 'Poland', 'Canada', 'Romania', 'France', 'West Germany', 'Mexico', 'Tanzania', 'Trinidad and Tobag
o', 'Germany', 'Ethiopia', 'Bulgaria', 'Sweden', 'Finland', 'Czechoslovakia', 'Jamaica', 'Cuba', 'New Zealand', 'Chinese Tai
```



2.Generate a new dataset for all the athletes in your original dataset.

Task-2 Generating New Data Set for All Athelitcs

+ Code + Markdown

[113]:

```
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("NEWDATE.csv")
```

```
[1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 2, 4, 1, 3, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1,
1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 3, 1, 1, 2, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 5, 2, 1, 1,
3, 1, 4, 1, 1, 1, 4, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 2,
1, 1, 1, 1, 3, 1, 1, 1, 1, 2, 3, 2, 2, 2, 1, 1, 1, 4, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 3, 3, 2, 2, 1, 1, 1,
2, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 1, 1, 2, 1, 1, 1, 1, 1, 1, 4, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1,
2, 2, 1, 2, 1, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 2, 2, 2, 4, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1,
2, 1, 1, 1, 2, 5, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 2, 2, 3, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 3, 1,
1, 1, 3, 1, 1, 2, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1,
```