Weekly Assignment 5a

Loan Pham and Brandan Owens

Q.1 We'll We'll be working with the 120 years of Olympic History dataset. Download the dataset "athlete_events.csv" and perform the following:

In [162…

```
#import dataset and tools
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

olympic_data = pd.read_csv("../dataFiles/athlete_events.csv")
olympic_data
```

Out[162…

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	1992 Summer	1992	Summer
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	2012 Summer	2012	Summer
2	3	Gunnar Nielsen Aaby	M	24.0	NaN	NaN	Denmark	DEN	1920 Summer	1920	Summer
3	4	Edgar Lindenau Aabye	M	34.0	NaN	NaN	Denmark/Sweden	DEN	1900 Summer	1900	Summer
4	5	Christine Jacoba Aaftink	F	21.0	185.0	82.0	Netherlands	NED	1988 Winter	1988	Winter
...	...	...	...	...	...	...	...	...	...	...	...
271111	135569	Andrzej ya	M	29.0	179.0	89.0	Poland-1	POL	1976 Winter	1976	Winter
271112	135570	Piotr ya	M	27.0	176.0	59.0	Poland	POL	2014 Winter	2014	Winter
271113	135570	Piotr ya	M	27.0	176.0	59.0	Poland	POL	2014 Winter	2014	Winter

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season
271114	135571	Tomasz Ireneusz ya	M	30.0	185.0	96.0	Poland	POL	1998 Winter	1998	Winter
271115	135571	Tomasz Ireneusz ya	M	34.0	185.0	96.0	Poland	POL	2002 Winter	2002	Winter

271116 rows × 15 columns

In [162…

```
#Q.1.a Filter the DataFrame to only include the rows corresponding to medal winners fro
olympic_filtered = olympic_data[olympic_data['Year'] >= 2016]
olympic_filtered = olympic_filtered[olympic_filtered['Medal'].notna()]
olympic_filtered
```

Out[162…

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	
158	62	Giovanni Abagnale	M	21.0	198.0	90.0	Italy	ITA	2016 Summer	2016	Summer	I Ja
161	65	Patimat Abakarova	F	21.0	165.0	49.0	Azerbaijan	AZE	2016 Summer	2016	Summer	I Ja
175	73	Luc Abalo	M	31.0	182.0	86.0	France	FRA	2016 Summer	2016	Summer	I Ja
450	250	Saeid Morad Abdevali	M	26.0	170.0	80.0	Iran	IRI	2016 Summer	2016	Summer	I Ja
794	455	Denis Mikhaylovich Ablyazin	M	24.0	161.0	62.0	Russia	RUS	2016 Summer	2016	Summer	I Ja
…	…	…	…	…	…	…	…	…	…	…	…	…
269511	134857	Zhu Ting	F	21.0	198.0	78.0	China	CHN	2016 Summer	2016	Summer	I Ja
270111	135132	Bojana ivkovi	F	28.0	186.0	72.0	Serbia	SRB	2016 Summer	2016	Summer	I Ja
270281	135205	Shakhobiddin Shokirovich Zoirov	M	23.0	169.0	52.0	Uzbekistan	UZB	2016 Summer	2016	Summer	I Ja
270370	135245	Milenko Zori	M	27.0	179.0	73.0	Serbia	SRB	2016 Summer	2016	Summer	I Ja

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	
271019	135525	Martin Zwicker	M	29.0	175.0	64.0	Germany	GER	2016 Summer	2016	Summer	I Ja

2023 rows × 15 columns

In [162…

```
#Q.1.b Find out the number of medals awarded in 2016 for each sport.
awards_per_sport = olympic_filtered[['Sport','Year','Medal']]
awards_per_sport.groupby(['Sport','Year']).count()
```

Out[162…

Sport	Year	Medal
Archery	**2016**	24
Athletics	**2016**	192
Badminton	**2016**	24
Basketball	**2016**	72
Beach Volleyball	**2016**	12
Boxing	**2016**	51
Canoeing	**2016**	82
Cycling	**2016**	84
Diving	**2016**	36
Equestrianism	**2016**	45
Fencing	**2016**	65
Football	**2016**	106
Golf	**2016**	6
Gymnastics	**2016**	66
Handball	**2016**	89
Hockey	**2016**	99
Judo	**2016**	56
Modern Pentathlon	**2016**	6
Rhythmic Gymnastics	**2016**	18
Rowing	**2016**	144
Rugby Sevens	**2016**	74
Sailing	**2016**	45
Shooting	**2016**	45
Swimming	**2016**	191

		Medal
Sport	**Year**	
Synchronized Swimming	**2016**	32
Table Tennis	**2016**	24
Taekwondo	**2016**	32
Tennis	**2016**	24
Trampolining	**2016**	6
Triathlon	**2016**	6
Volleyball	**2016**	72
Water Polo	**2016**	78
Weightlifting	**2016**	45
Wrestling	**2016**	72

In [162…

```
#Q.1.c Filter the DataFrame one more time to only include the records for the top five
awards_count = awards_per_sport.groupby(['Sport','Year']).count()
awards_per_sport_sorted = awards_count.sort_values('Medal', ascending=False)
awards_per_sport_sorted.head(5)
```

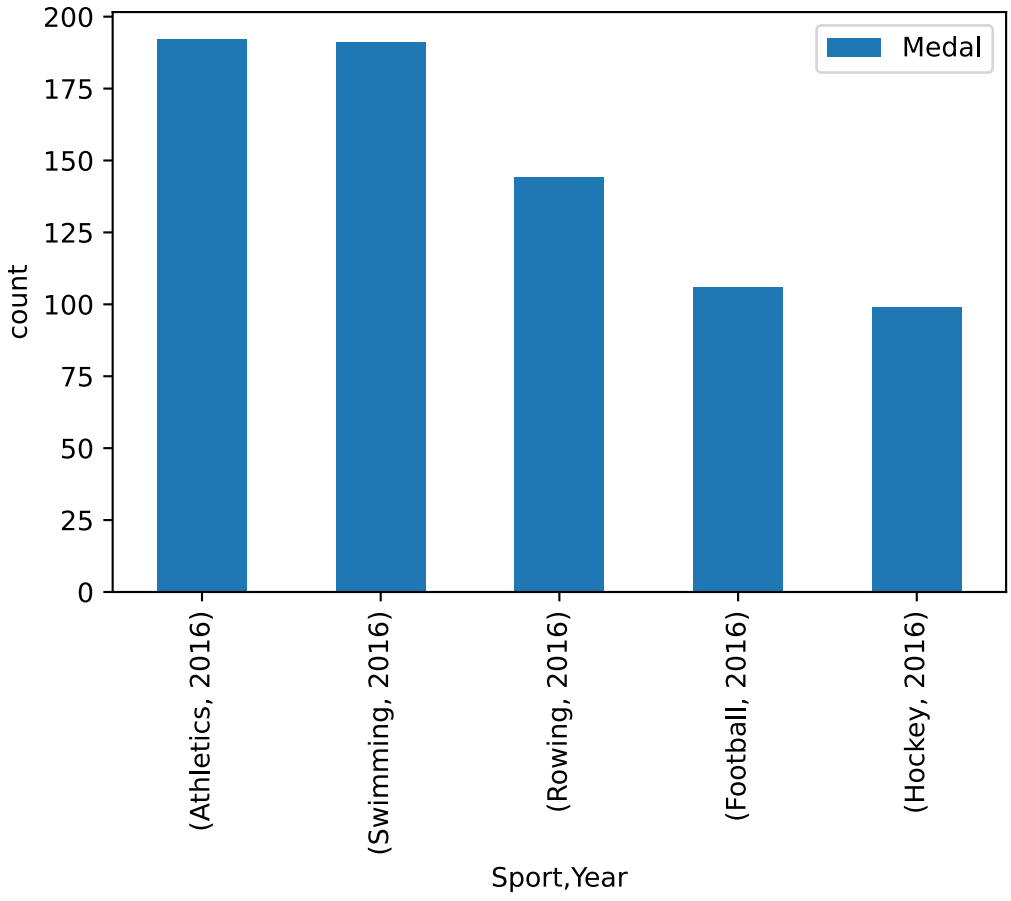
Out[162…

		Medal
Sport	**Year**	
Athletics	**2016**	192
Swimming	**2016**	191
Rowing	**2016**	144
Football	**2016**	106
Hockey	**2016**	99

In [163…

```
#Q.1.d Generate a bar plot of record counts corresponding to each of the top five sport
plot = awards_per_sport_sorted.head(5)
plot.plot.bar(ylabel= 'count')
```

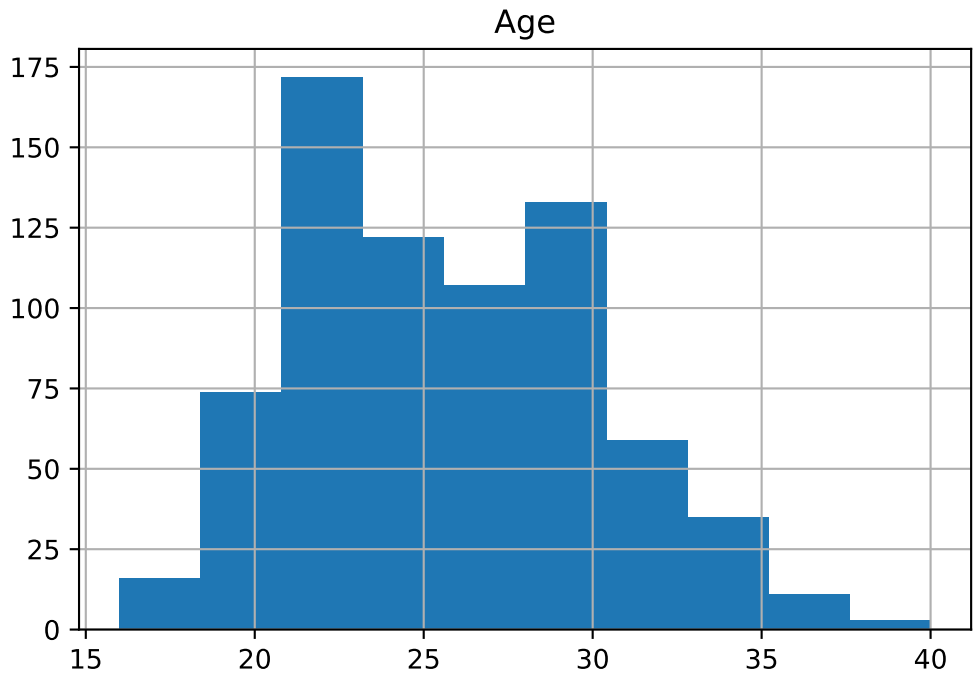
Out[163… <AxesSubplot:xlabel='Sport,Year', ylabel='count'>



In [163…

```
#Q.1.e Generate a histogram for the “Age” of all medal winners in the top five sports (
top_five = olympic_filtered.loc[olympic_filtered['Sport'].isin(['Athletics', 'Swimming'
top_five = top_five[top_five['Year'] >= 2016]
medals_age = top_five[['Sport','Age','Medal']]
medals = ['Gold','Silver','Bronze']
medals_age.hist()
```

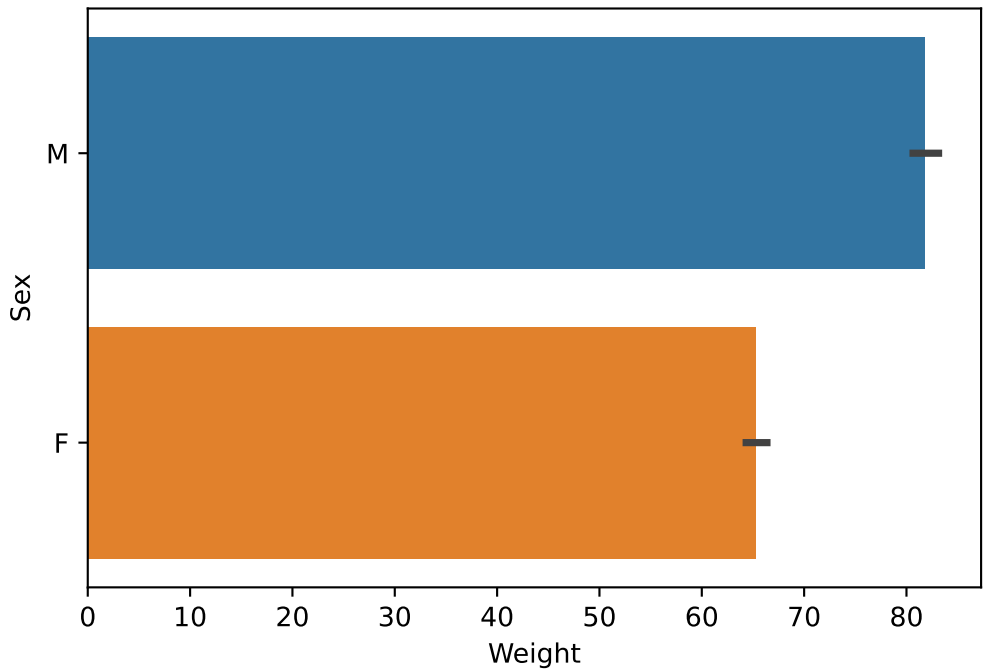
Out[163… array([[<AxesSubplot:title={'center':'Age'}>]], dtype=object)



In [163…
```
#Q.1.f Generate a bar plot indicating how many medals were won by each country's team i
medals_by_country = top_five[['NOC','Medal']]
medals_by_country = medals_by_country.head(10)
axs.bar(medals_by_country['NOC'],medals_by_country['Medal'])
```

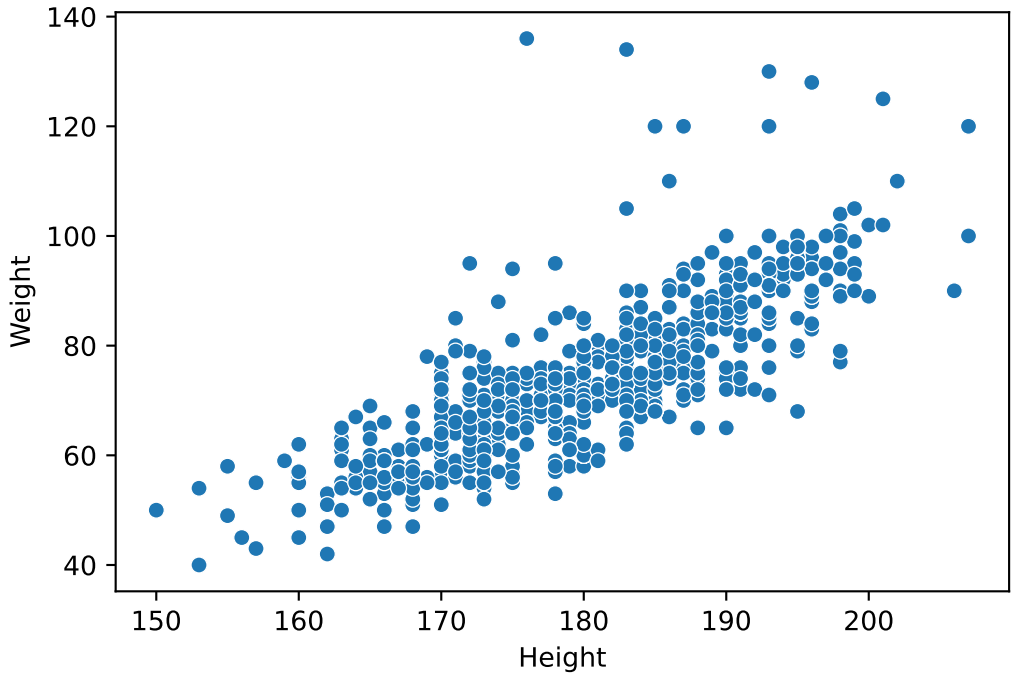
Out[163… <BarContainer object of 10 artists>

In [163…
```
#Q.1.g Generate a bar plot indicating the average weight of players, categorized based
weight_by_gender = top_five[['Weight','Sex']]
ax = sns.barplot(x='Weight', y="Sex", data=weight_by_gender)
```



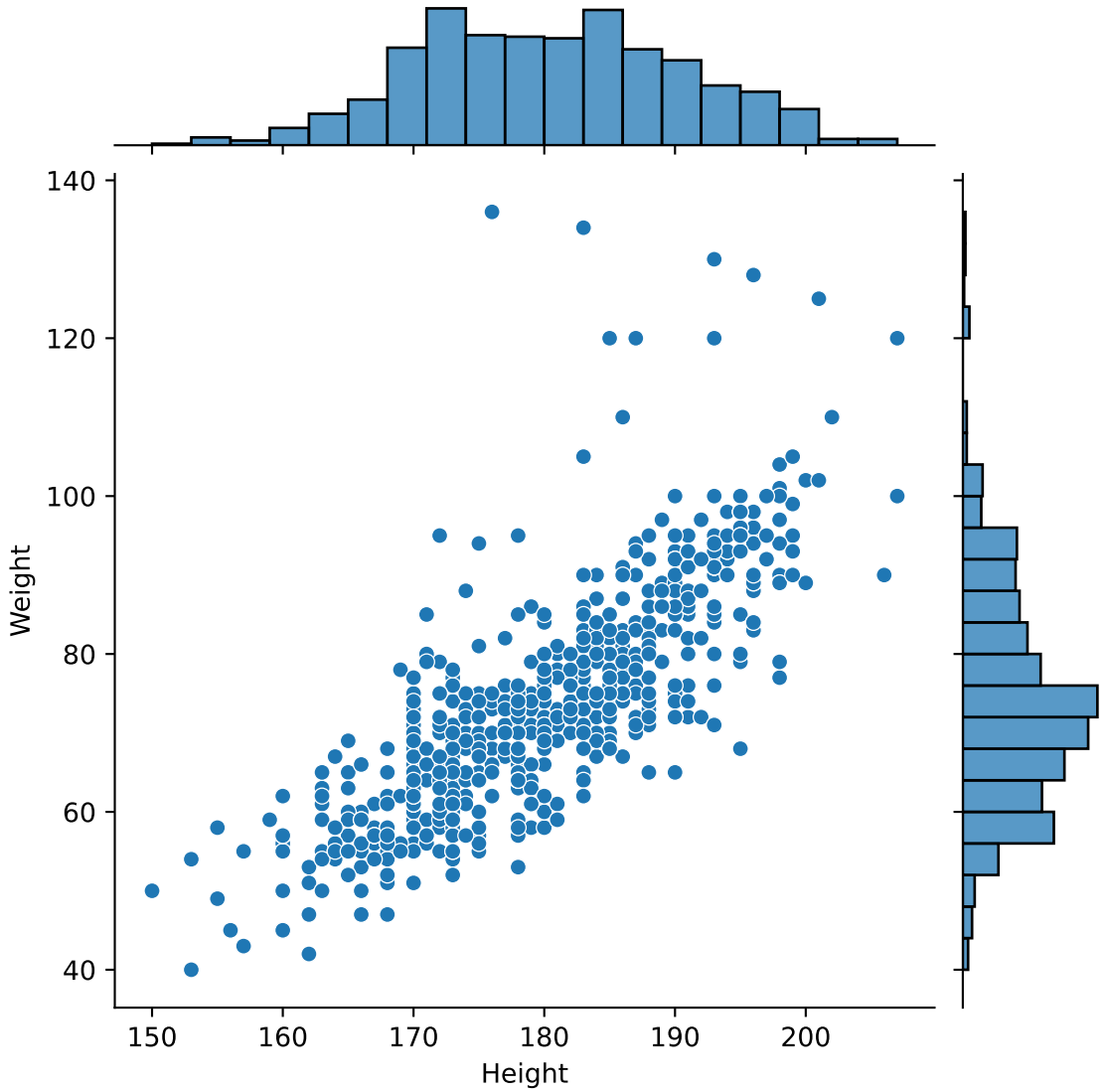
In [163…
```
#Q.1.h Create a scatter plot with x=height and y=weight.
```

```
height_by_weight = top_five[['Weight','Height']]
ax = sns.scatterplot(x='Height', y='Weight', data=height_by_weight)
```



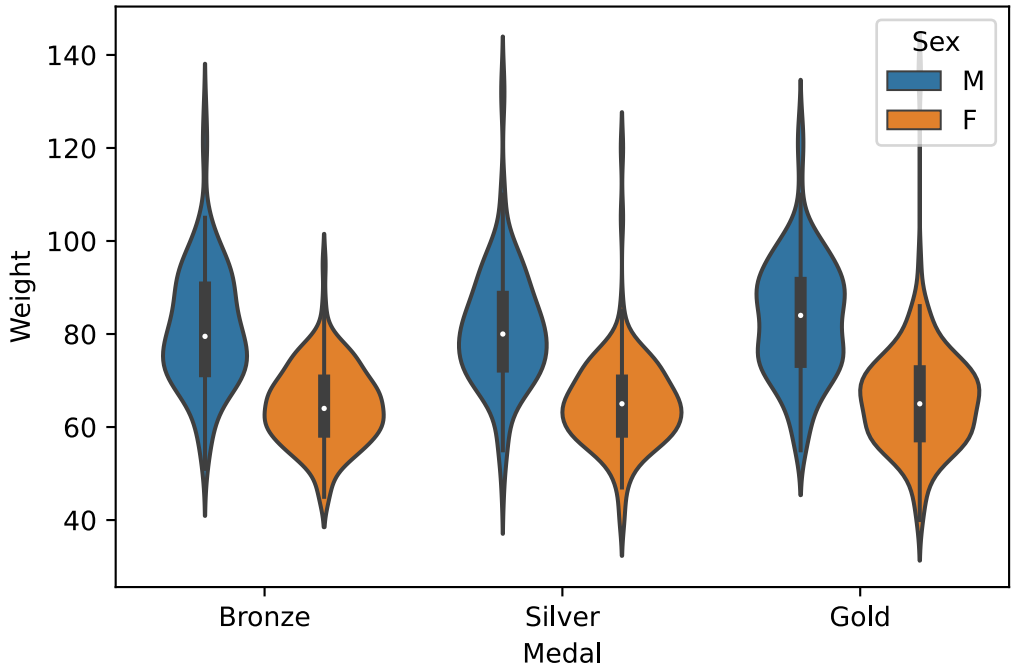
In [163…

```
#Q.1.i Create a joint plot with x=height and y=weight.
ax = sns.jointplot(x='Height', y='Weight', data=height_by_weight)
```



In [163…

```
#Q.1.j Create two violin plots: 1 for distribution of weight by genders & class of meda
ax = sns.violinplot(x=top_five['Medal'], y=top_five['Weight'], hue=top_five['Sex'], dat
```



Q.2 We will work with the HPI dataset. The objective is to draw a bar plot depicting the number of countries in each region and a heatmap indicating the number of countries in various ranges of wellbeing and life-expectancy.

In [163…

```
#Q.2.a Read from the dataset "hpi_data_countries.txt" (it is txt file separated by \t).
hpi_df = pd.read_csv('..\dataFiles\hpi_data_countries.txt', delimiter = "\t")
hpi_data
```

Out[163…

	HPI Rank	Country	Region	Life Expectancy (years)	Wellbeing (0-10)	Inequality of outcomes	Ecological Footprint (gha/capita)	Happy Planet Index
0	1	Costa Rica	Americas	79.1	7.3	15%	2.8	44.7
1	2	Mexico	Americas	76.4	7.3	19%	2.9	40.7
2	3	Colombia	Americas	73.7	6.4	24%	1.9	40.7
3	4	Vanuatu	Asia Pacific	71.3	6.5	22%	1.9	40.6
4	5	Vietnam	Asia Pacific	75.5	5.5	19%	1.7	40.3
…	…	…	…	…	…	…	…	…
135	136	Mongolia	Asia Pacific	68.6	4.9	22%	6.1	14.3
136	137	Benin	Sub Saharan Africa	59.2	3.2	44%	1.4	13.4
137	138	Togo	Sub Saharan Africa	58.6	2.9	43%	1.1	13.2
138	139	Luxembourg	Europe	81.1	7.0	7%	15.8	13.2

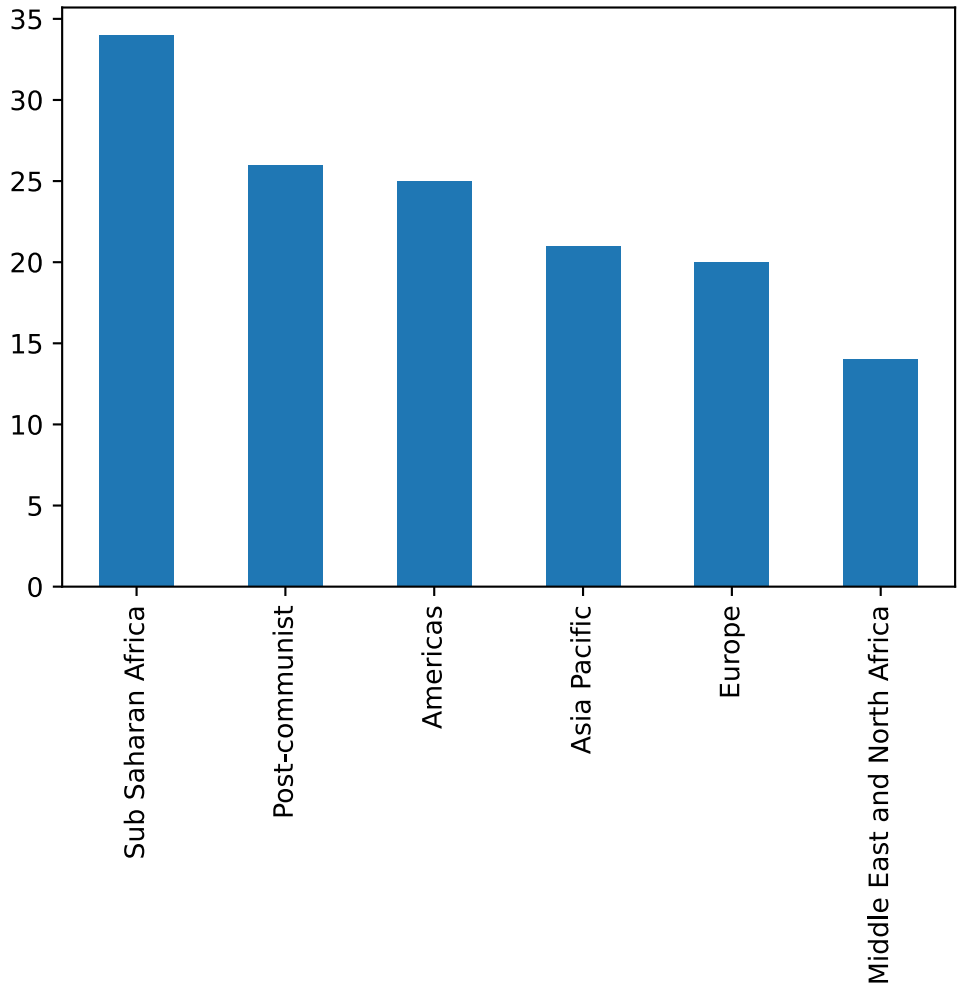
	HPI Rank	Country	Region	Life Expectancy (years)	Wellbeing (0-10)	Inequality of outcomes	Ecological Footprint (gha/capita)	Happy Planet Index
139	140	Chad	Sub Saharan Africa	50.8	4.0	51%	1.5	12.8

140 rows × 8 columns

In [163…

```
#Q.2.b Count the number of rows for each region to generate a bar chart.
region_count = hpi_df['Region'].value_counts()
region_count.plot(kind='bar')
```

Out[163… <AxesSubplot:>



In [163…

```
#Q.2.c Use pd.cut to bin the values into different categories.
#Then generate a heatmap using cmap="Greens"
hpi_df["Wellbeing (0-10)(binned)"] = pd.cut(hpi_df["Wellbeing (0-10)"], bins=np.linspac
hpi_df["Life Expectancy (years)(binned)"] = pd.cut(hpi_df["Life Expectancy (years)"], b
heat_df = hpi_df.groupby(["Wellbeing (0-10)(binned)", "Life Expectancy (years)(binned)"
heat_df = heat_df.sort_values(ascending=False, by="Life Expectancy (years)(binned)")

sns.heatmap(heat_df, cmap="Greens")
```

Out[163… <AxesSubplot:xlabel='Wellbeing (0-10)(binned)', ylabel='Life Expectancy (years)(binned)'>