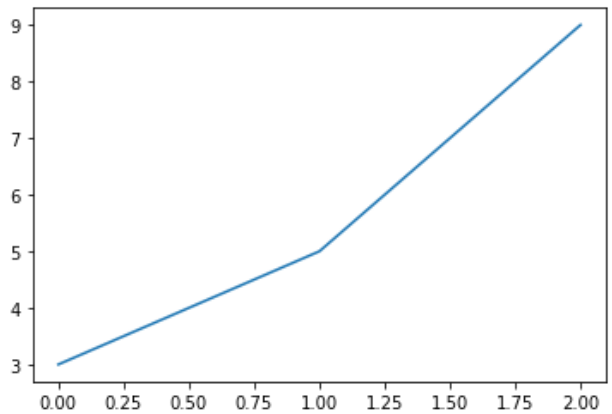
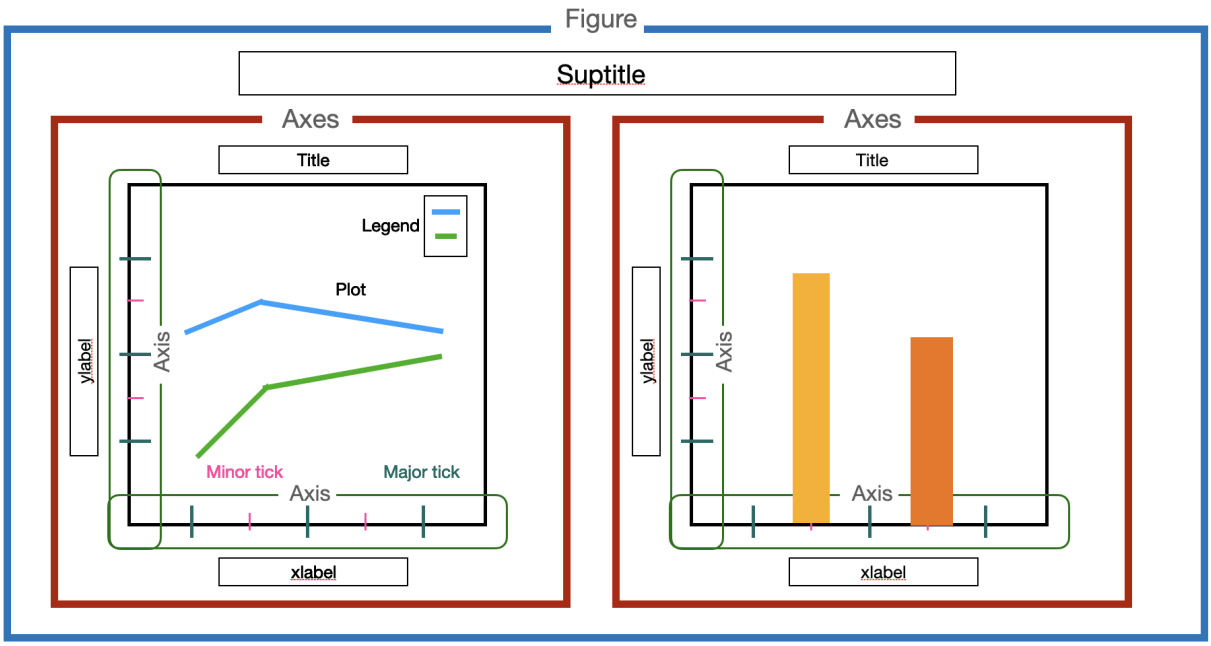
Link: https://colab.research.google.com/drive/1BOyZUnmNjbDYcILb4h5hxXNilCfWAWgy?usp=sharing

```
import pandas as pd
import numpy as np


# Where all do you think Data Voz. is helpful or needed?
# Exploratory - see some difficult patterns, EDA
# Explanatory - I want to create a stort out of it and I want to stake holders


# Art and Science Data Visualisation

# Science
# Understanding the anatomy of a plot
# How to choose which plot is right to answer my question?'

# Art
# Choosing thje right scale, axis, ticks and labels
# Identify and removing clutters
# Ways to highlight the information


# Intro to Matplotlib


import matplotlib.pyplot as plt
import seaborn as sns


# (0,3) (1,5) (2,9)


x_val = [0,1,2]
y_val = [3,5,9]


plt.plot(x_val, y_val)
```

```
[<matplotlib.lines.Line2D at 0x7f5a550bf400>]
```






```
# how to choose the plot


# Data
# Rows - Records, Samples, Volume, Data-points
# Columns - Features, Attributes, Characteristics


# Columns - Continous, Categorical
# Categorical
# Oridinal - inherant ordering
# Nominal - no ordering


# Thumb for deciding the right plot?

# Question
# 1. How many variables/features are involved in answering my question?
  # Univatiate
  # Bivariate
  # Multi-variate
# 2. What are data-types of different variables involved?
  # Numerical
  # Categorical


# Univariate Data Visualisation
  # N - histogram, boxplot, swarm plot, density plot, violin
  # C - pie-chart, donut, bar chart
# Bivariate Data Visualisation
  # N, N - scatter, ...
  # N, C -
  # C, C -
# Multi-variate Data Visualisatiob (3 variables)
  # N, N, N -
  # N, N, C -
  # C, C, N -
  # C, C, C -
```

```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/021/299/original/final_vg1_-
```

```
--2023-02-11 15:34:20--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/021/
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 65.8.55.79, 65.8.
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|65.8.55.79|:443.
HTTP request sent, awaiting response... 200 OK
Length: 2041483 (1.9M) [text/plain]
Saving to: 'final_vg.csv'

final_vg.csv        100%[===================>]   1.95M  --.-KB/s    in 0.07s

2023-02-11 15:34:20 (29.9 MB/s) - 'final_vg.csv' saved [2041483/2041483]
```

```
import pandas as pd
import numpy as np
data = pd.read_csv('final_vg.csv')
data.head()
```

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales
0	2061	1942	NES	1985.0	Shooter	Capcom	4.569217
1	9137	¡Shin Chan Flipa en colores!	DS	2007.0	Platform	505 Games	2.076955
2	14279	.hack: Sekai no Mukou ni + Versus	PS3	2012.0	Action	Namco Bandai Games	1.145709
3	8359	.hack//G.U. Vol.1//Rebirth	PS2	2006.0	Role-Playing	Namco Bandai Games	2.031986
4	7109	.hack//G.U. Vol.2//Reminisce	PS2	2006.0	Role-Playing	Namco Bandai Games	2.792725

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16652 entries, 0 to 16651
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Rank          16652 non-null  int64
 1   Name          16652 non-null  object
 2   Platform      16652 non-null  object
 3   Year          16381 non-null  float64
 4   Genre         16652 non-null  object
 5   Publisher     16594 non-null  object
 6   NA_Sales      16652 non-null  float64
 7   EU_Sales      16652 non-null  float64
 8   JP_Sales      16652 non-null  float64
 9   Other_Sales   16652 non-null  float64
 10  Global_Sales  16652 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB
```

```
data["Genre"].unique()
```

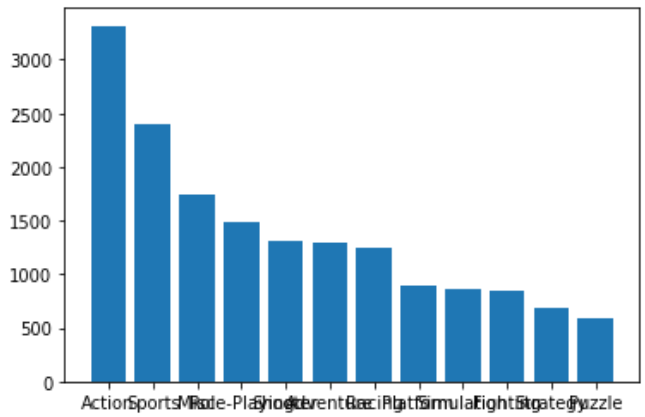
```
array(['Shooter', 'Platform', 'Action', 'Role-Playing', 'Racing', 'Misc',
       'Adventure', 'Sports', 'Puzzle', 'Simulation', 'Strategy',
       'Fighting'], dtype=object)
```

```
cat_counts = data["Genre"].value_counts()
cat_counts
```

```
Action          3316
Sports          2400
Misc            1739
Role-Playing    1488
Shooter         1310
Adventure       1286
Racing          1249
Platform         886
Simulation       867
Fighting         848
Strategy         681
Puzzle           582
Name: Genre, dtype: int64
```

```
x_bar = cat_counts.index
y_bar = cat_counts
plt.bar(x_bar, y_bar)
```

```
<BarContainer object of 12 artists>
```

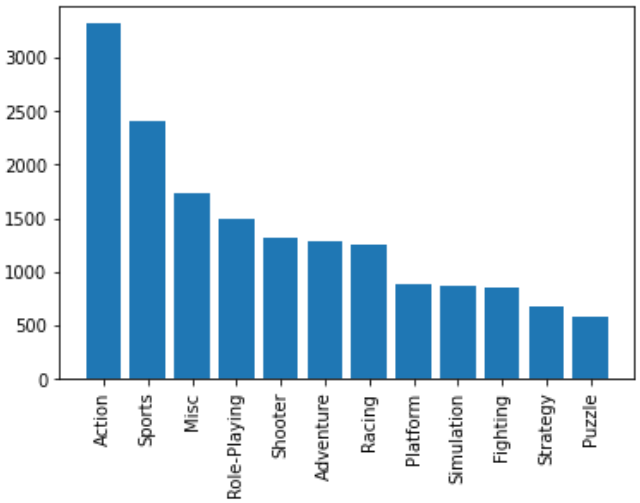



```
plt.figure(figsize=(12,8))
plt.bar(x_bar, y_bar)
```

```
<BarContainer object of 12 artists>
```

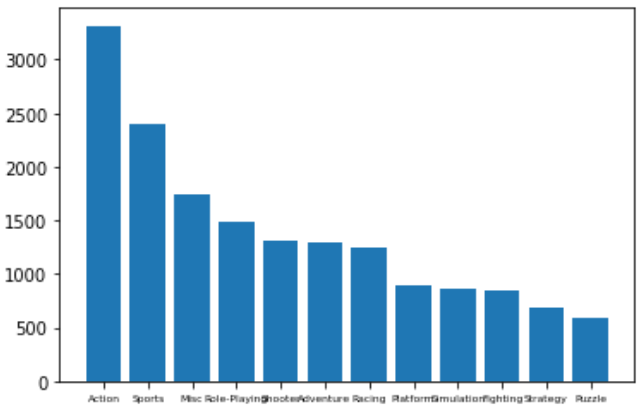

```
plt.bar(x_bar, y_bar)
plt.xticks(rotation=90)
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
 <a list of 12 Text major ticklabel objects>)
```



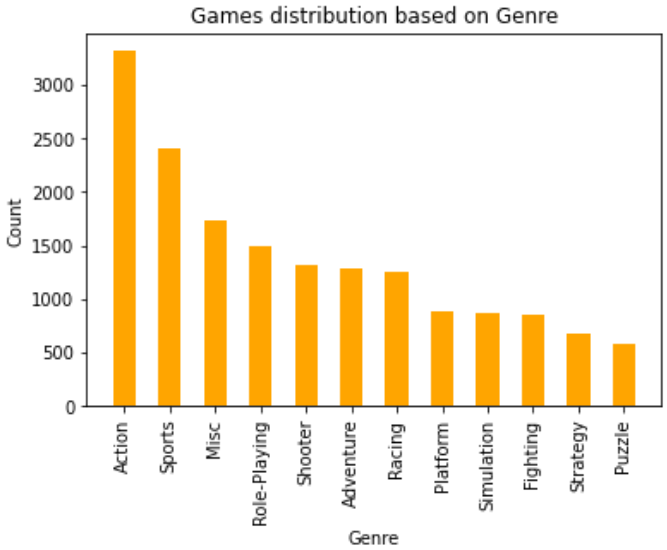

```
plt.bar(x_bar, y_bar)
plt.xticks(fontsize=6)
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
 <a list of 12 Text major ticklabel objects>)
```

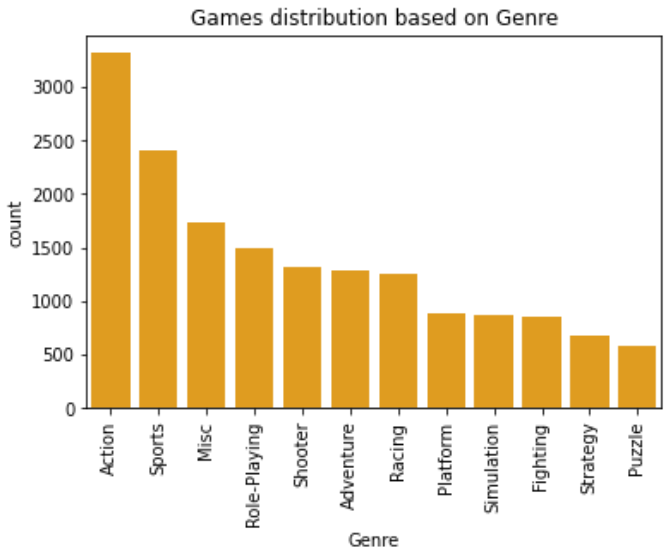


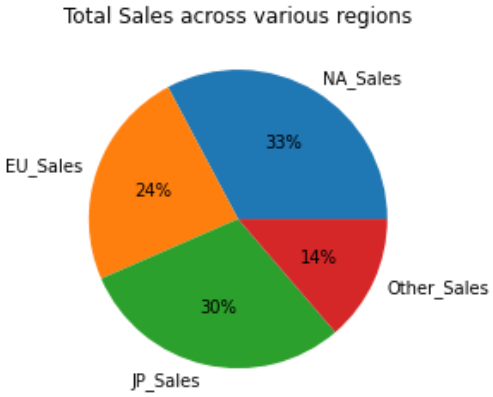

```
plt.bar(x_bar, y_bar, width=0.5)
```

```
plt.bar(x_bar, y_bar, color="orange", width=0.5)
plt.xticks(rotation=90)
plt.title("Games distribution based on Genre",fontsize=12)
plt.xlabel("Genre", fontsize=10)
plt.ylabel("Count", fontsize=10)
plt.show()
```

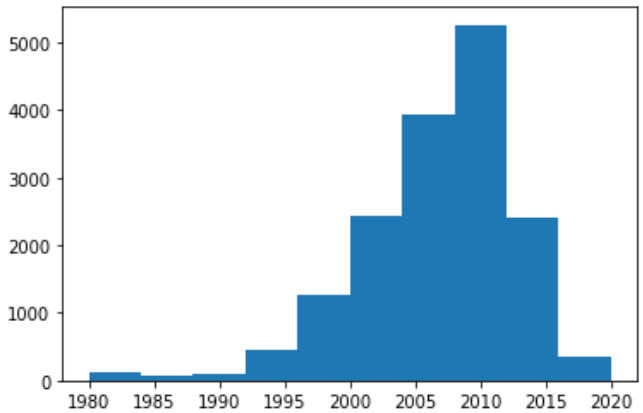



```
sns.countplot(data = data, x = "Genre", color="orange",
              order = data["Genre"].value_counts().index)
plt.xticks(rotation=90)
plt.title("Games distribution based on Genre",fontsize=12)
plt.show()
```

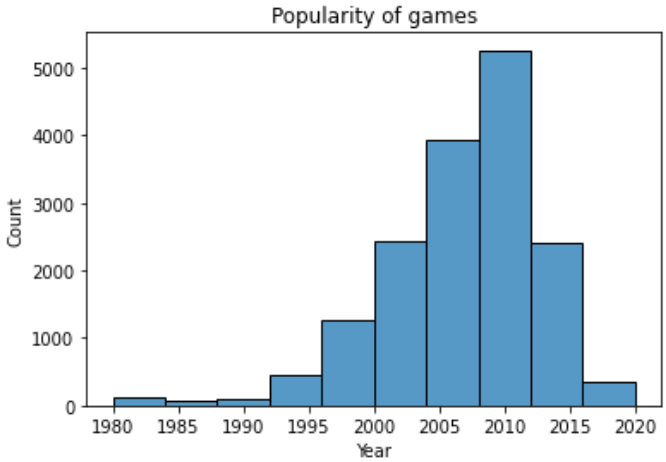




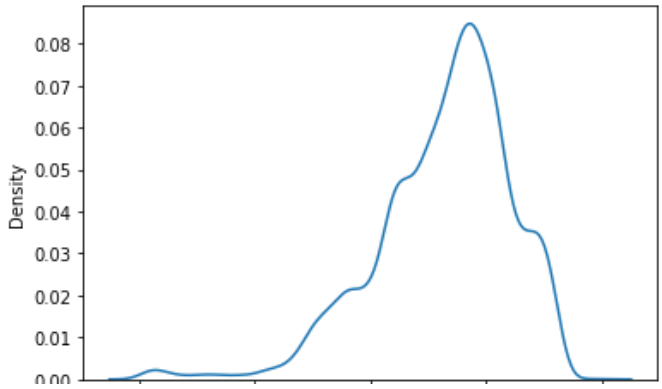

```
# Univariate, Numerical (Cont.)

# Popularity of video games (no. of games in market) year by year?

plt.hist(data["Year"],bins=10)
plt.show()
```




```
sns.histplot(data["Year"],bins=10)
plt.title("Popularity of games")
plt.show()
```

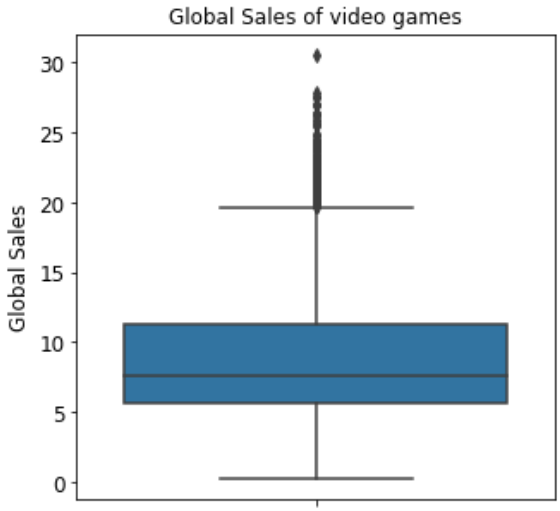



```
sns.kdeplot(data["Year"])
plt.show()
```



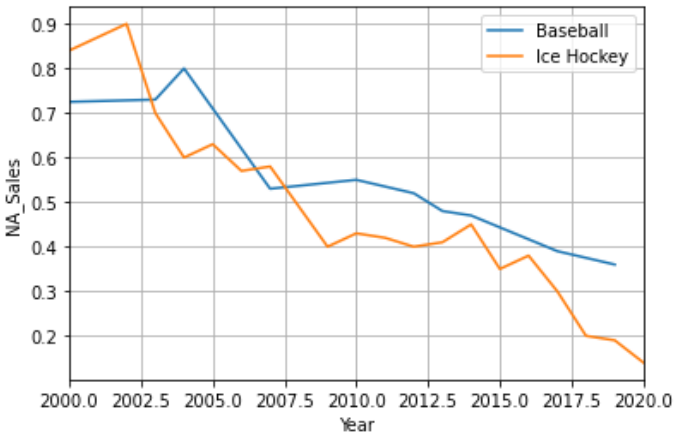

```
# Typical earnings from a video game


plt.figure(figsize=(5,5))
sns.boxplot(y = data["Global_Sales"])
plt.yticks(fontsize=12)
plt.ylabel('Global Sales', fontsize=12)
plt.title('Global Sales of video games', fontsize=12)
```

```
Text(0.5, 1.0, 'Global Sales of video games')
```

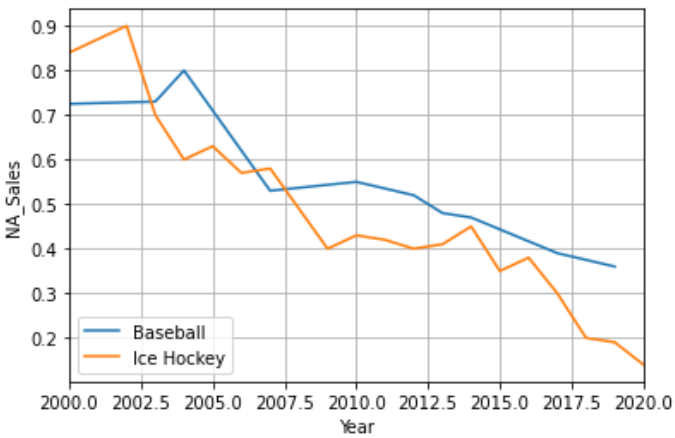



```
# Bivariate Data Visualisation, N N


# Relationship between year and sales of ice hockey
ih = data.loc[data['Name']=='Ice Hockey']
baseball = data.loc[data['Name']=='Baseball']
sns.lineplot(x='Year', y='NA_Sales', data=baseball)
sns.lineplot(data = ih, x="Year", y="NA_Sales")
plt.legend(["Baseball", "Ice Hockey"])
plt.xlim(left=2000, right=2020)
plt.show()
```

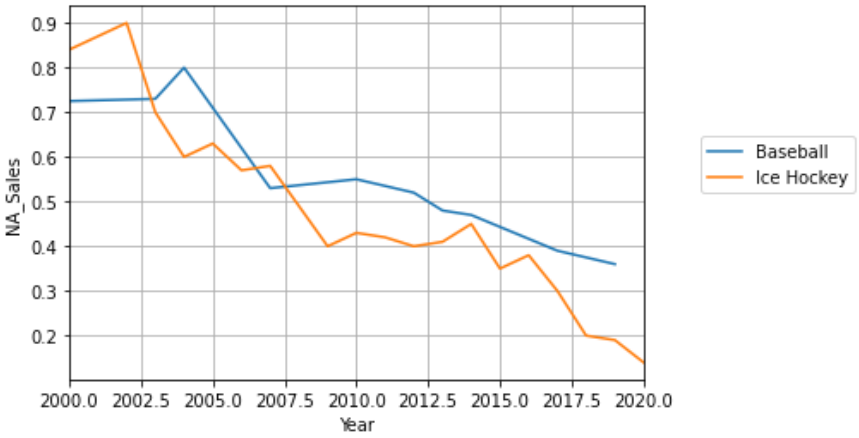

```
# Relationship between year and sales of ice hockey
ih = data.loc[data['Name']=='Ice Hockey']
baseball = data.loc[data['Name']=='Baseball']
sns.lineplot(x='Year', y='NA_Sales', data=baseball)
sns.lineplot(data = ih, x="Year", y="NA_Sales")
plt.legend(["Baseball", "Ice Hockey"])
plt.xlim(left=2000, right=2020)
plt.grid()
plt.show()
```



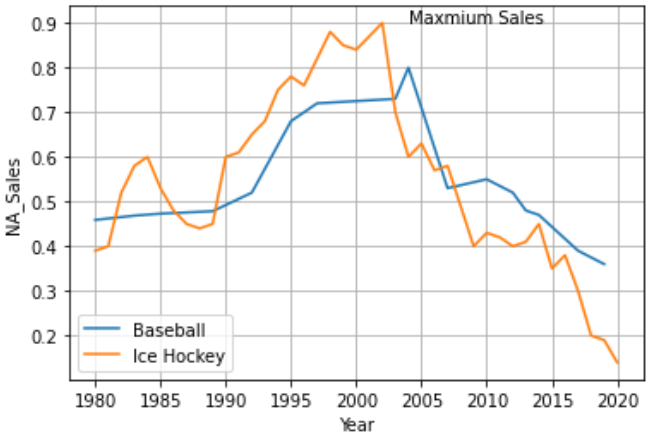

```
# Relationship between year and sales of ice hockey
ih = data.loc[data['Name']=='Ice Hockey']
baseball = data.loc[data['Name']=='Baseball']
sns.lineplot(x='Year', y='NA_Sales', data=baseball)
sns.lineplot(data = ih, x="Year", y="NA_Sales")
plt.legend(["Baseball", "Ice Hockey"], loc="lower left")
plt.xlim(left=2000, right=2020)
plt.grid()
plt.show()
```




```
# Relationship between year and sales of ice hockey
ih = data.loc[data['Name']=='Ice Hockey']
baseball = data.loc[data['Name']=='Baseball']
sns.lineplot(x='Year', y='NA_Sales', data=baseball)
sns.lineplot(data = ih, x="Year", y="NA_Sales")
plt.legend(["Baseball", "Ice Hockey"], loc=(1.1, 0.5))
plt.xlim(left=2000, right=2020)
plt.grid()
plt.show()
```

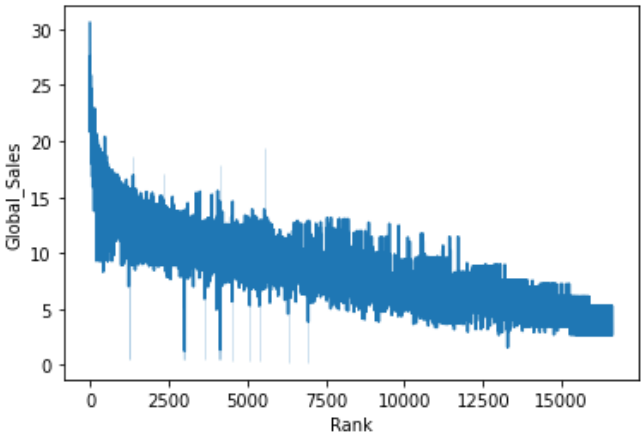



```
# Relationship between year and sales of ice hockey
ih = data.loc[data['Name']=='Ice Hockey']
baseball = data.loc[data['Name']=='Baseball']
sns.lineplot(x='Year', y='NA_Sales', data=baseball)
sns.lineplot(data = ih, x="Year", y="NA_Sales")
plt.legend(["Baseball", "Ice Hockey"], loc="lower left")
plt.text(2004, max(ih["NA_Sales"]), "Maxmium Sales")
plt.grid()
plt.show()
```



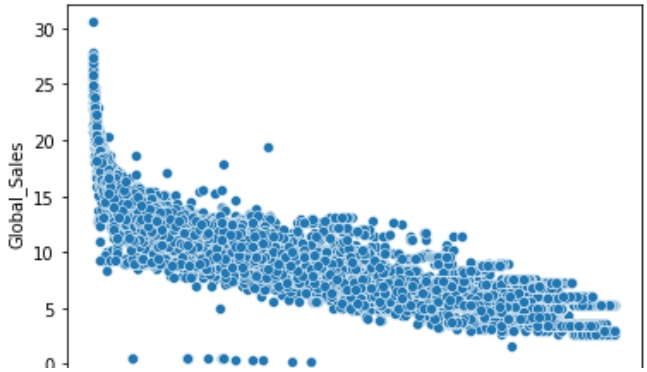

```
# Rank of the video games is associated with sales?
sns.lineplot(data=data, x="Rank", y="Global_Sales")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a5226eeb0>
```




```
sns.scatterplot(data=data, x="Rank", y="Global_Sales")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a52385790>
```

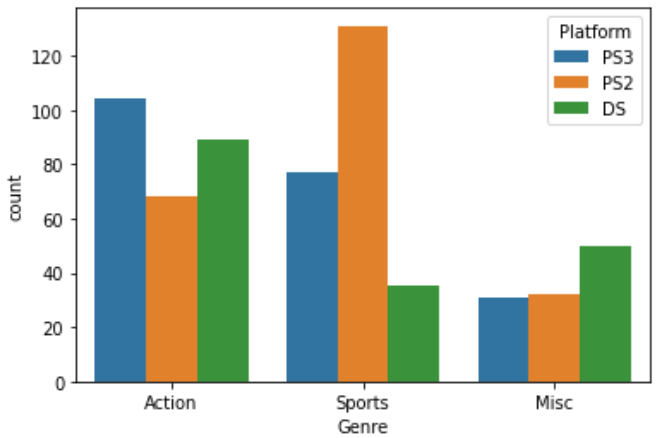


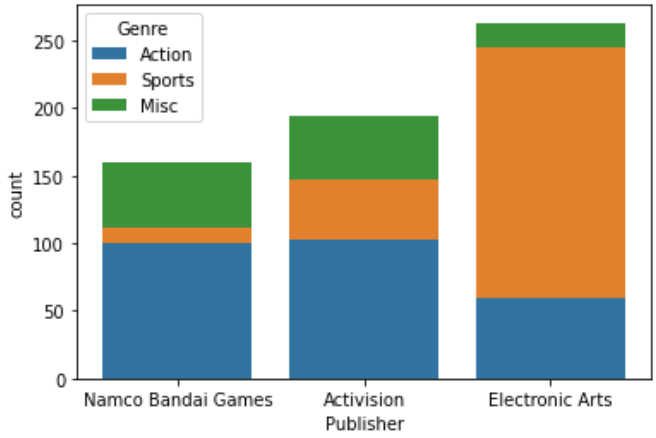

```
# Bivariate, Categorical Categorical


top3_pub = data['Publisher'].value_counts().index[:3]
top3_gen = data['Genre'].value_counts().index[:3]
top3_plat = data['Platform'].value_counts().index[:3]
top3_data = data.loc[(data["Publisher"].isin(top3_pub)) & (data["Platform"].isin(top3_plat)) & (da
top3_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 617 entries, 2 to 16640
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Rank          617 non-null    int64
 1   Name          617 non-null    object
 2   Platform      617 non-null    object
 3   Year          611 non-null    float64
 4   Genre         617 non-null    object
 5   Publisher     617 non-null    object
 6   NA_Sales      617 non-null    float64
 7   EU_Sales      617 non-null    float64
 8   JP_Sales      617 non-null    float64
 9   Other_Sales   617 non-null    float64
 10  Global_Sales  617 non-null    float64
dtypes: float64(6), int64(1), object(4)
memory usage: 57.8+ KB
```

```
sns.countplot(data = top3_data, x="Genre", hue="Platform")
```

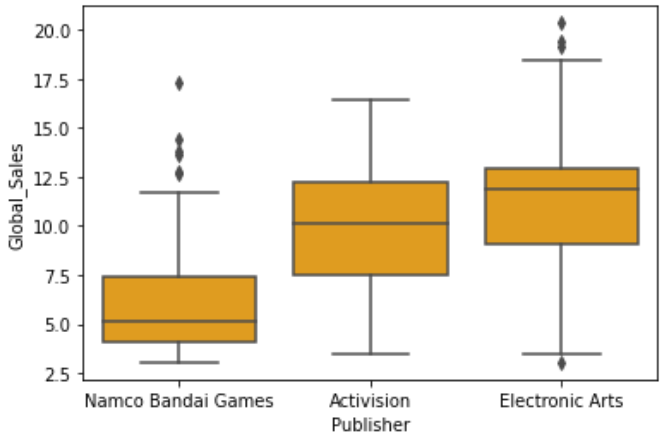
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a521792b0>
```





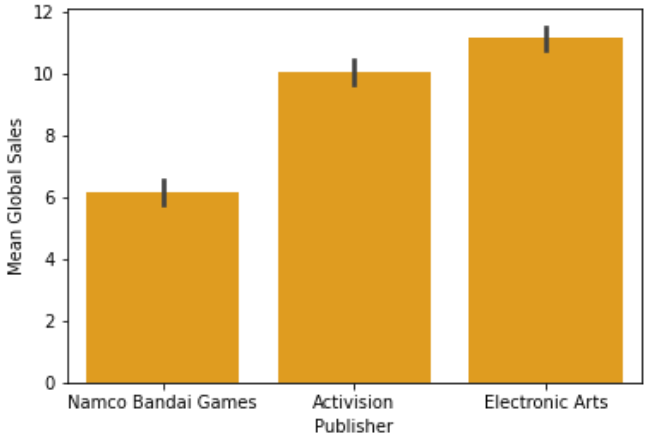

```
# Bivariate - Cont Cat

# Typical sales of top3 publishers seperately?

sns.boxplot(data = top3_data, x="Publisher", y="Global_Sales", color="orange")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a54859400>
```



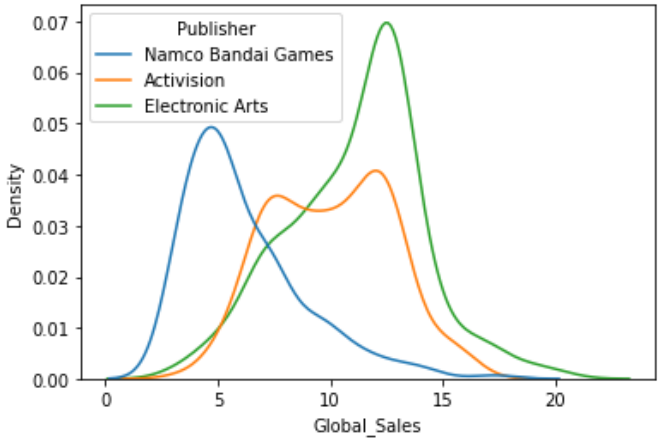

```
# Mean earnings of top3 publishers seperately?
sns.barplot(data = top3_data, x="Publisher", y="Global_Sales", color="orange", estimator=np.mean)
plt.ylabel("Mean Global Sales")
```

```
Text(0, 0.5, 'Mean Global Sales')
```

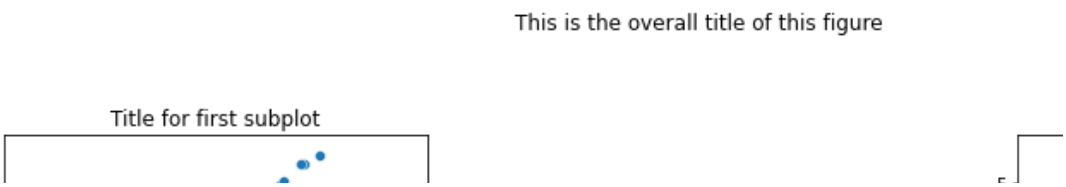



```
# Distribution of sales of top3 publishers seperately?
sns.kdeplot(data = top3_data, hue="Publisher", x="Global_Sales")
```

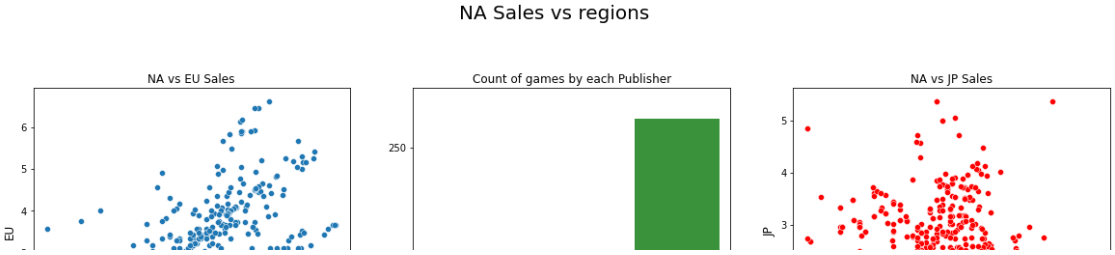
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a51230b20>
```



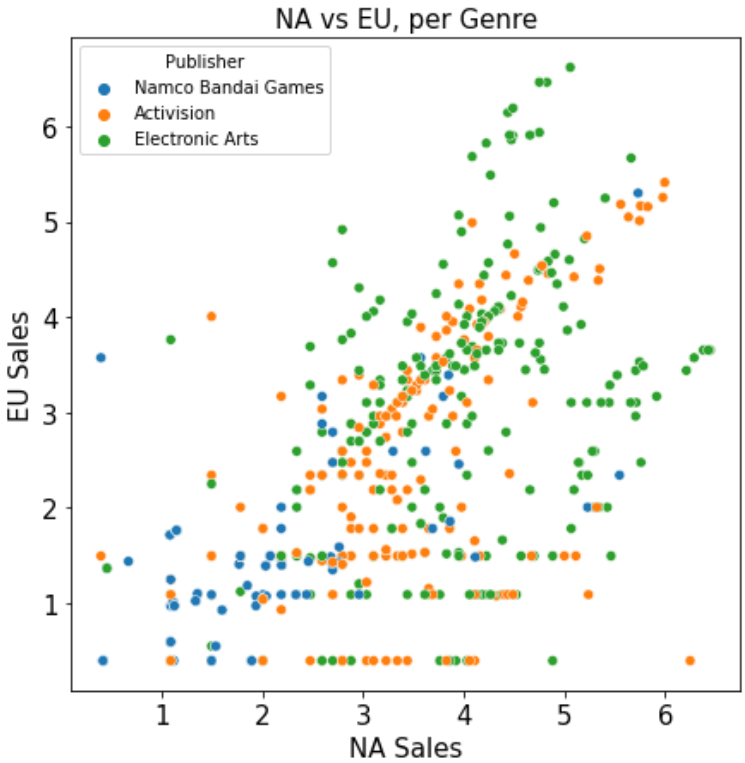

```
fig = plt.figure(figsize=(15,10))

plt.subplot(2, 3, 1)
sns.scatterplot(x='NA_Sales', y='EU_Sales', data=top3_data)
plt.title("Title for first subplot")

plt.subplot(2, 3, 3)
sns.scatterplot(x='NA_Sales', y='JP_Sales', data=top3_data, color='red')
plt.title("Title for third subplot")

plt.subplot(2, 3, 5)
sns.scatterplot(x='NA_Sales', y='Other_Sales', data=top3_data, color='red')
plt.title("Title for fifth subplot")

fig.suptitle("This is the overall title of this figure")

plt.show()
```



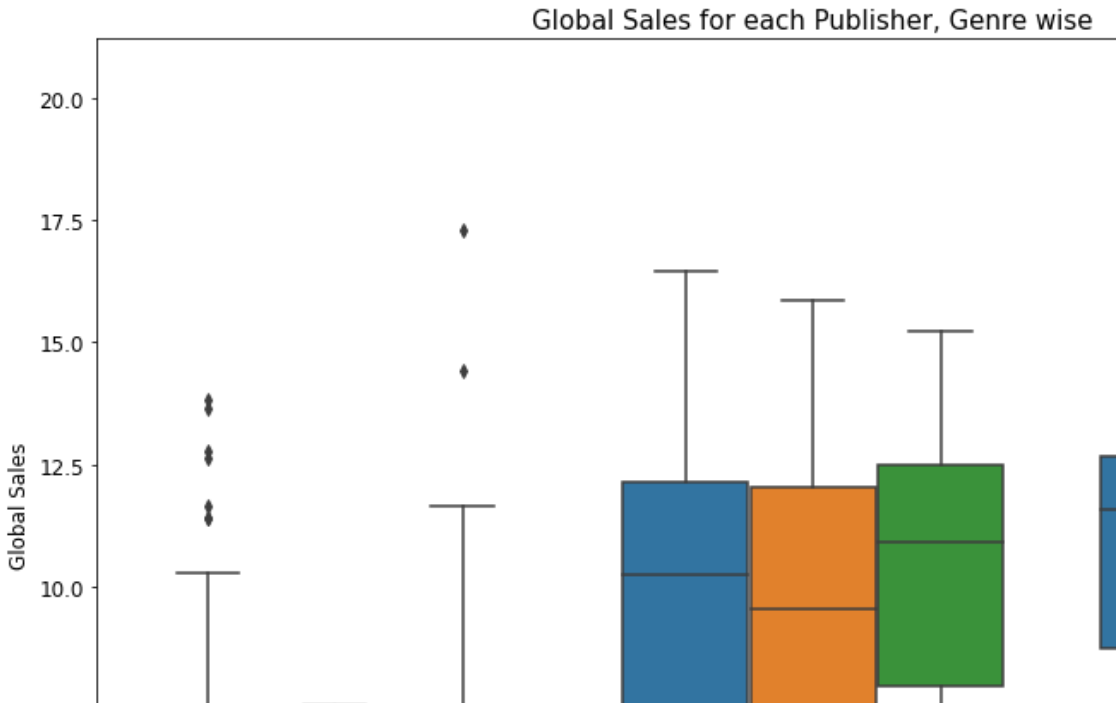

```
plt.figure(figsize=(20,12)).suptitle("NA Sales vs regions",fontsize=20)
# Using a 2x3 subplot
plt.subplot(2, 3, 1)
sns.scatterplot(x='NA_Sales', y='EU_Sales', data=top3_data)
plt.title('NA vs EU Sales', fontsize=12)
plt.xlabel('NA', fontsize=12)
plt.ylabel('EU', fontsize=12)

plt.subplot(2, 3, 3)
sns.scatterplot(x='NA_Sales', y='JP_Sales', data=top3_data, color='red')
plt.title('NA vs JP Sales', fontsize=12)
plt.xlabel('NA', fontsize=12)
plt.ylabel('JP', fontsize=12)


plt.subplot(2, 3, 4)
sns.scatterplot(x='NA_Sales', y='Other_Sales', data=top3_data, color='green')
plt.title('NA vs Other Region Sales', fontsize=12)
plt.xlabel('NA', fontsize=12)
plt.ylabel('Other', fontsize=12)

plt.subplot(2, 3, 6)
sns.scatterplot(x='NA_Sales', y='Global_Sales', data=top3_data, color='orange')
plt.title('NA vs Global Sales', fontsize=12)
plt.xlabel('NA', fontsize=12)
plt.ylabel('Global', fontsize=12)

plt.subplot(1,3,2)
sns.countplot(x='Publisher', data=top3_data)
plt.title('Count of games by each Publisher', fontsize=12)
plt.xlabel('Publisher', fontsize=12)
plt.ylabel('Count of games', fontsize=12)

plt.show()
```



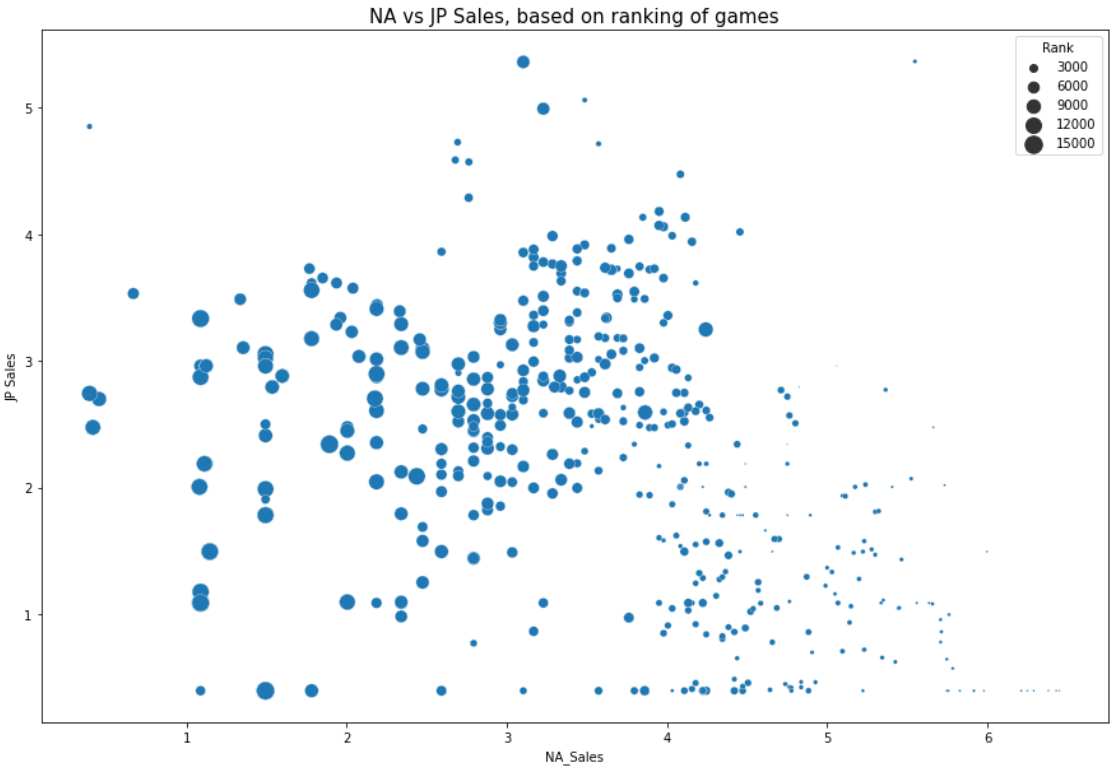

```
plt.figure(figsize=(7,7))
sns.scatterplot(x='NA_Sales', y='EU_Sales',hue='Publisher',data=top3_data)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.xlabel('NA Sales',fontsize=15)
plt.ylabel('EU Sales',fontsize=15)
plt.title('NA vs EU, per Genre', fontsize=15)
plt.show()
```




```
plt.figure(figsize=(15,10))
sns.boxplot(x='Publisher',y='Global_Sales',hue='Genre',data=top3_data)
plt.xlabel('Genre', fontsize=12)
plt.ylabel('Global Sales', fontsize=12)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.title('Global Sales for each Publisher, Genre wise', fontsize=15)
plt.show()
```

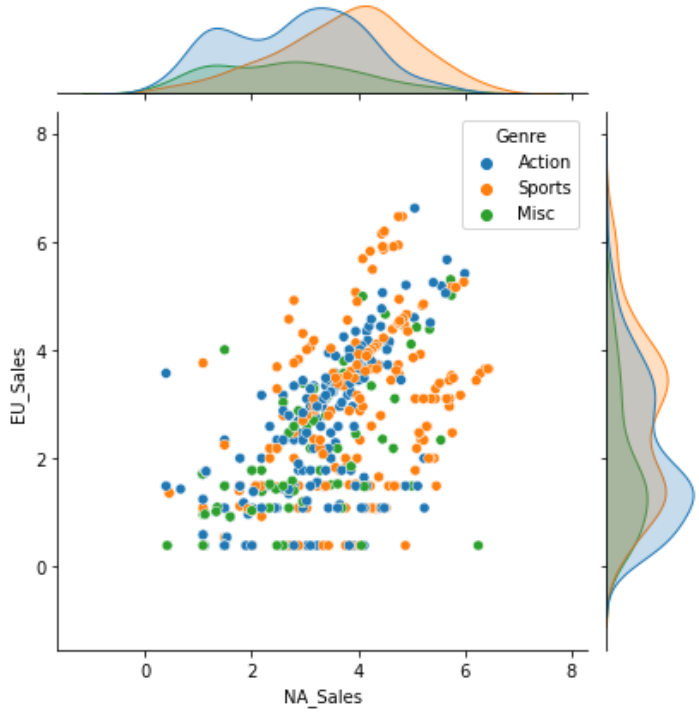



```
plt.figure(figsize=(15,10))
# sns.scatterplot(x=data['NA_Sales'], y=data['JP_Sales'],data=top3_data, size=data['Rank'], sizes=
sns.scatterplot(x='NA_Sales', y='JP_Sales', size='Rank', sizes=(1, 200), data=top3_data)

plt.xlabel('NA_Sales',fontsize=10)
plt.ylabel('JP Sales', fontsize=10)
plt.title('NA vs JP Sales, based on ranking of games', fontsize=15)
plt.show()
```



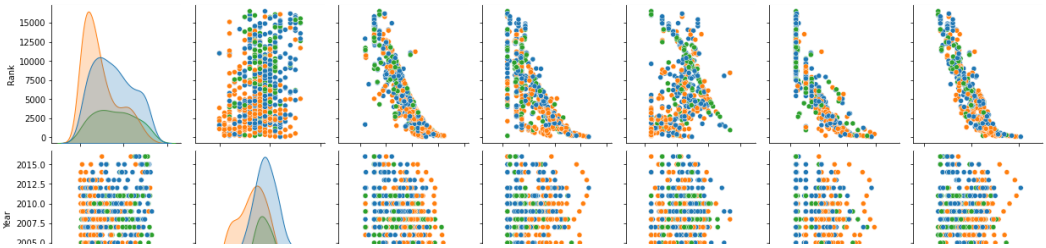
```
sns.jointplot(x='NA_Sales', y='EU_Sales', data=top3_data, hue='Genre')
```

```
<seaborn.axisgrid.JointGrid at 0x7f5a4ee3ddf0>
```




```
# Pairwise relation between every two possible numerical variable
sns.pairplot(data=top3_data, hue="Genre")
```

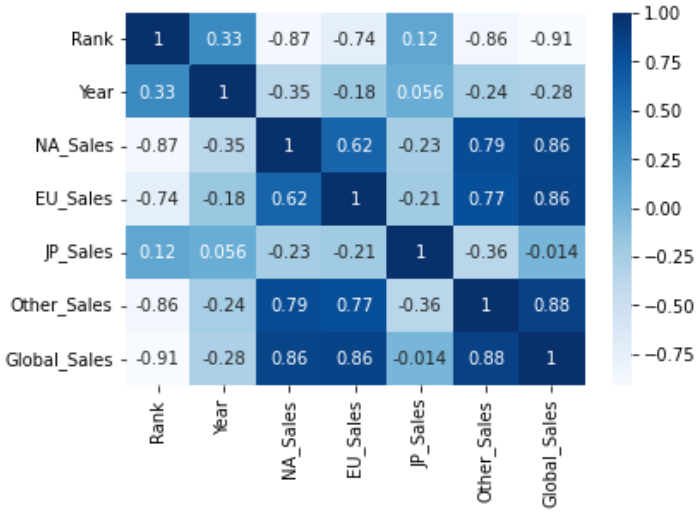
```
<seaborn.axisgrid.PairGrid at 0x7f5a4e327250>
```



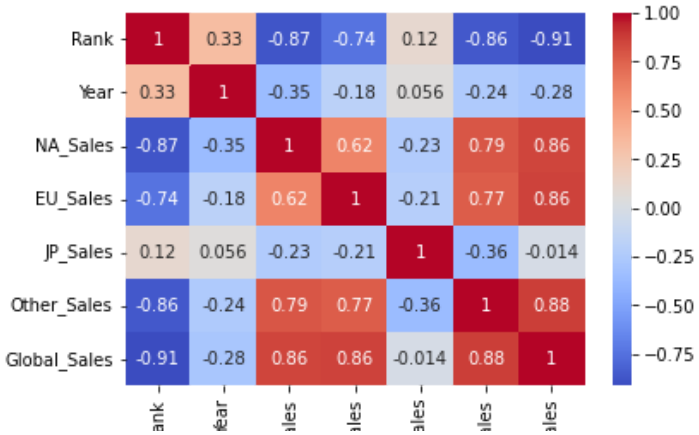
```
top3_data.corr() # -1 to 1
```

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Globa
Rank	1.000000	0.328705	-0.873726	-0.735711	0.115459	-0.857567	
Year	0.328705	1.000000	-0.354256	-0.178026	0.055864	-0.239876	
NA_Sales	-0.873726	-0.354256	1.000000	0.617483	-0.233315	0.794353	
EU_Sales	-0.735711	-0.178026	0.617483	1.000000	-0.208249	0.771105	
JP_Sales	0.115459	0.055864	-0.233315	-0.208249	1.000000	-0.355825	
Other_Sales	-0.857567	-0.239876	0.794353	0.771105	-0.355825	1.000000	
Global_Sales	-0.911721	-0.280351	0.856300	0.864147	-0.014193	0.878816	

```
sns.heatmap(top3_data.corr(), cmap= "Blues", annot=True)
plt.show()
```




```
sns.heatmap(top3_data.corr(), cmap= "coolwarm", annot=True)
plt.show()
```



✓ 0s completed at 22:57