Turkan Sevgili

sevgilit@mef.edu.tr

30 December 2017

This document was prepared for term project of BDA 507 course, Fall 2017 MEF University.

Term Project

BDA 507 Introduction to Computer Programming

## Introduction

Data

Description:

My dataset that will used in this project and exploratory data analysis on it contains a list of video games with sales greater than 100,000 copies along with critic and user ratings. It is a combined web scrape from VGChartz and Metacritic along with manually entered year of release values for most games with a missing year of release. The original coding was created by Rush Kirubi and can be found, but it limited the data to only include a subset of video game platforms. Not all of the listed video games have information on Metacritic, so there data set does have missing values.

Content and Data fields:

In this report, I analyze the video games with sales greater than 100,000 copies along with critic and user ratings. Source of this analyze is in [Kaggle Datase](https://www.kaggle.com/saurograndi/airplane-crashes-since-1908)t that is referenced in reference part. Data we analyze contains games names, its publishers, ratings, sales in different locations and other information.

The data set consists of 16,719 observations and 16 variables. Fields include;

* **Name** - The games name
* **Platform** - Platform of the games release (i.e. PC,PS4, etc.)
* **Year\_of\_Release** - Year of the game's release
* **Genre** - Genre of the game
* **Publisher** - Publisher of the game
* **NA\_Sales** - Sales in North America (in millions)
* **EU\_Sales** - Sales in Europe (in millions)
* **JP\_Sales** - Sales in Japan (in millions)
* **Other\_Sales** - Sales in the rest of the world (in millions)
* **Global\_Sales** - Total worldwide sales.
* **Critic\_Score** - Aggregate score compiled by Metacritic staff
* **Critic\_Count** - The number of critics used in coming up with the critic score
* **User\_Score** - Score by Metacritic's subscribers
* **User\_Count** - Number of users who gave the user score
* **Developer** – Developer of video game
* **Rating** - The ESRB ratings

About Video Game Industry:

Video Game is an electronic game that is played on electronic medium devices such as personal computer, TV screen, gaming console or mobile phone. Some time the Video Game industry is called the interactive entertainment industry. The input device used for games, the game controller, varies across platforms. Common controller includes game pad, joysticks, mouse, keyboard, the touchscreens of mobile devices and buttons. Players typically view the game on a video screen or television and there are often game sounds from loudspeakers. Video Game development has a long history since 1970’s and in recent past with the revolution of the smartphones and tablets introduced new categories of video games such as mobile and social games. Developers introduced various technology and methodology in the computing system to popularize and make the video game more interesting and interactive such as “virtual reality”.

## Objectives

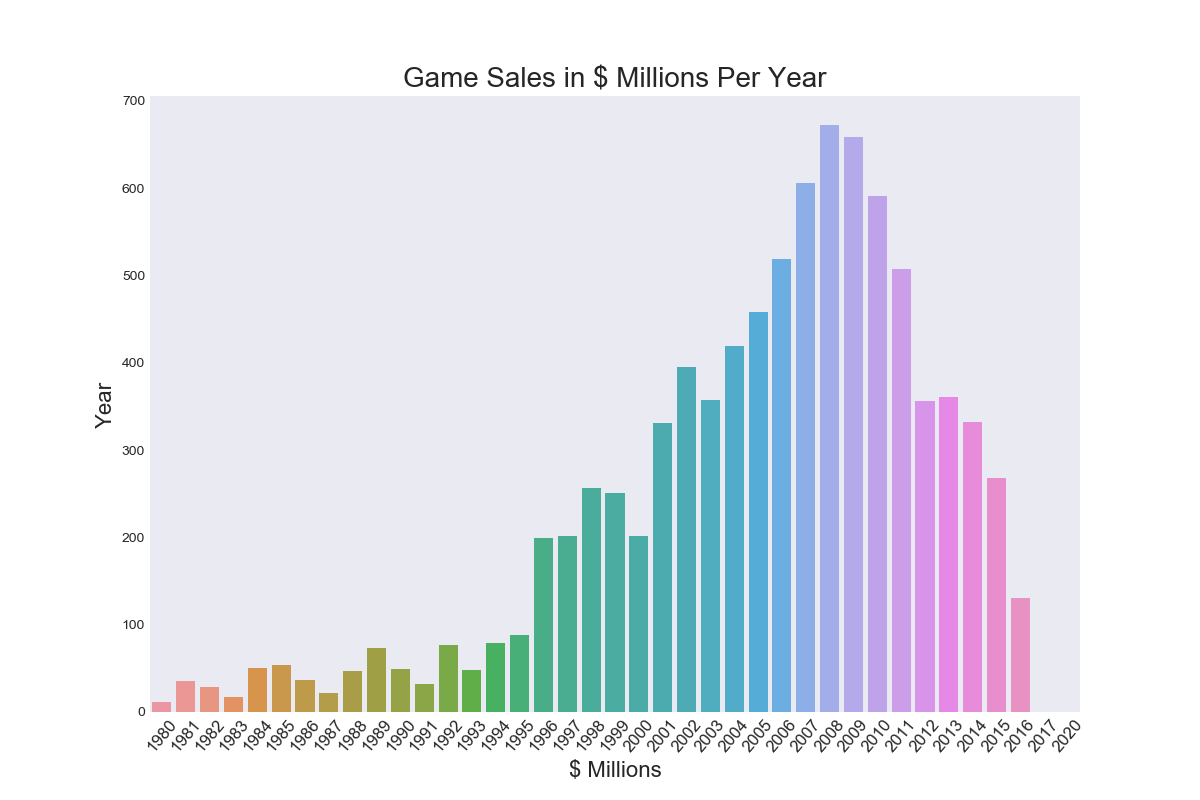
The motivation of this project is to visualize the data set and practice exploratory analysis of data set. For the better understanding I have analyzed the data by some histograms and plot, which will help to know the trend of the industry. By analyzing I’d like answer following questions :

* What are top 20 highest grossing games since 1980?
* How many publishers are in the market and who are the top publishers by number of releases?
* Who are the top publishers by Total Revenue?
* Which genre generated most revenue per release?
* What is the most preferred platform?

## Descriptive Statistics

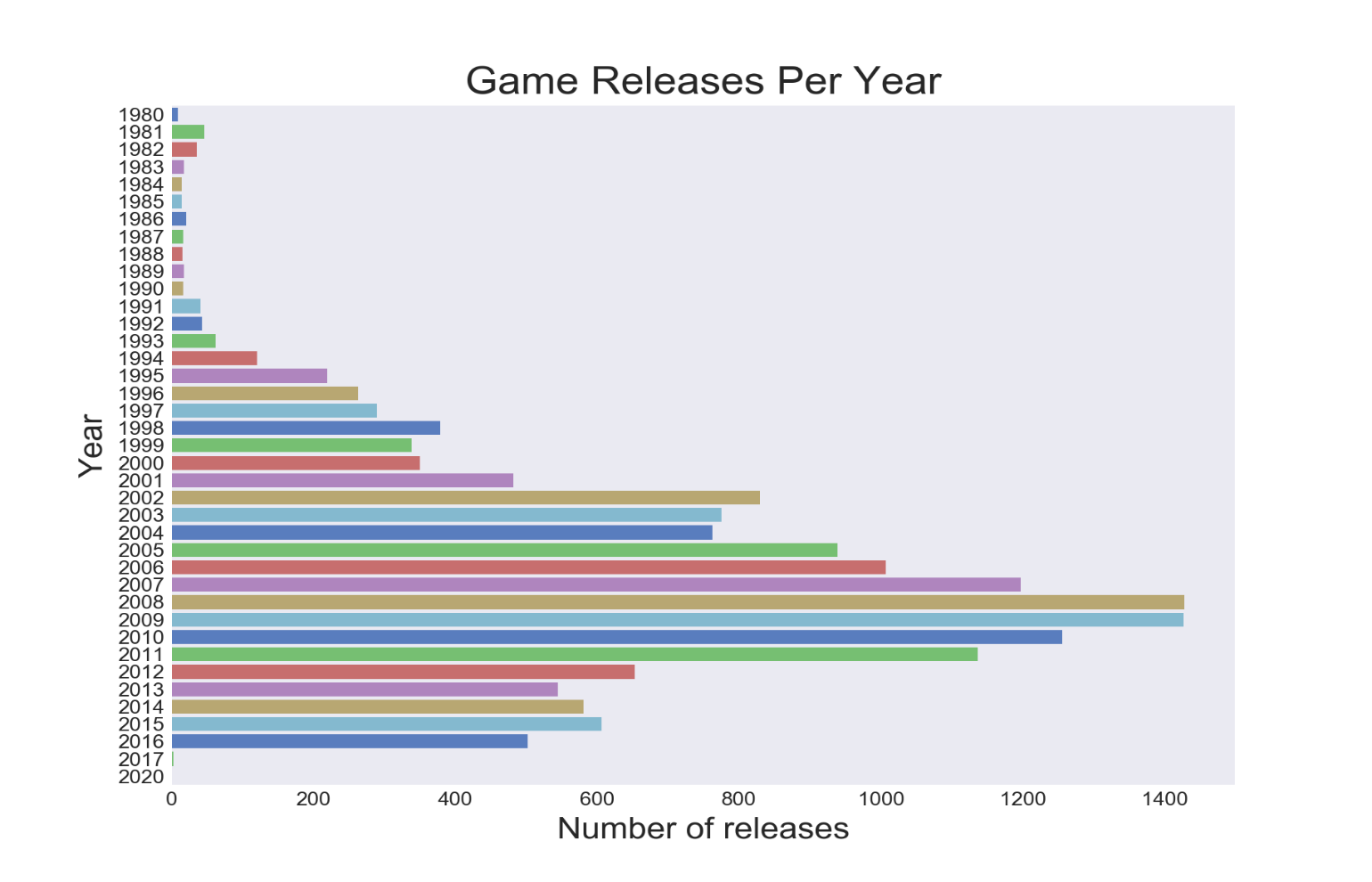
There are 16719 different games with 582 different publishers, 31 different platforms and 13 unique genres.

* **First I checked the number of releases by year and revenue by year**



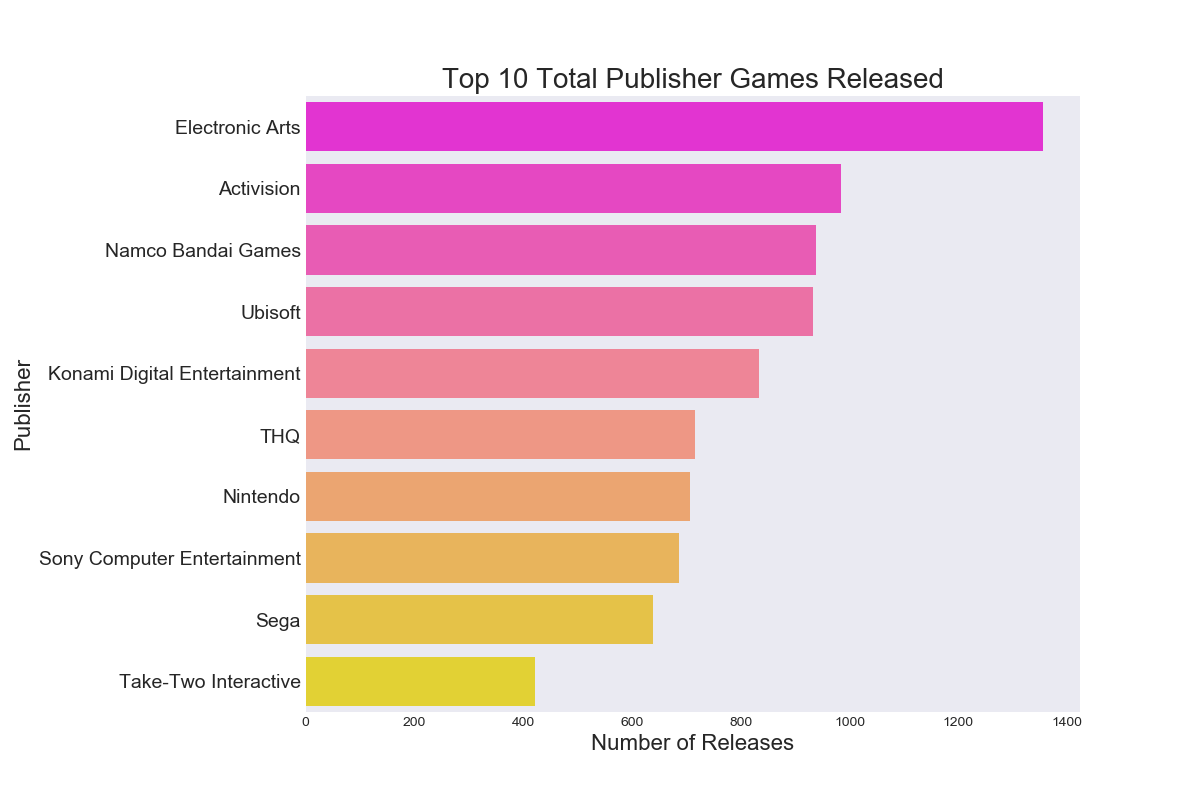
There is huge spike in the number of releases after 2000 and it peaked during 2008 and 2009. It has started declining last few years. This surprising when demand is more options should have been more. Maximum types of game has been found on 2008 and 2009.

* **Following chart shows number of releases and revenue by year just horizontal basis;**



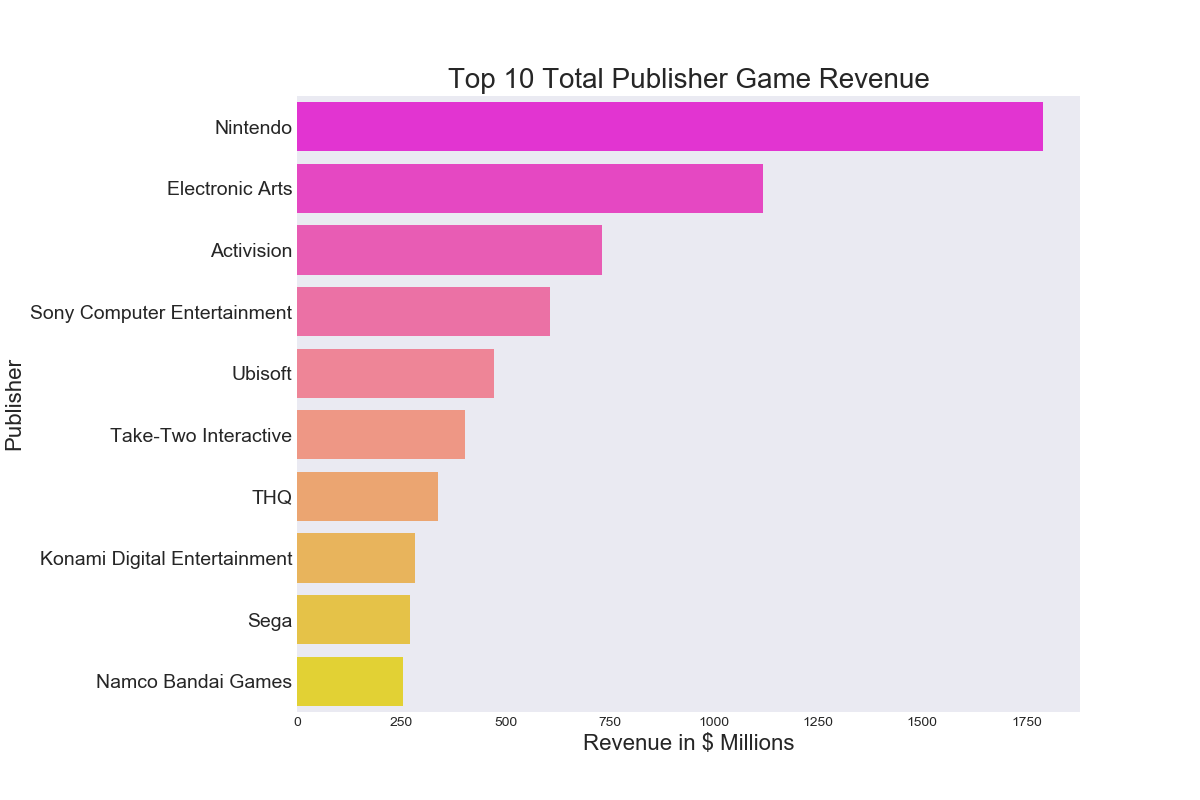
Total revenue was increasing till 2008 after that it is in declined. Maximum revenue was in 2008 and 2009.

* **How many publishers are in the market and who are the top publishers by number of releases?**



There are 582 different publishers. EA is the top publisher with Activision coming in second.

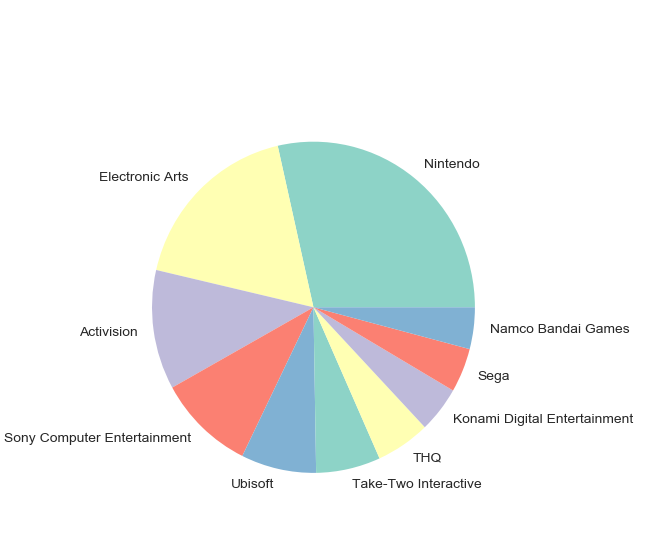
* **Who are the top publishers by Total Revenue?**



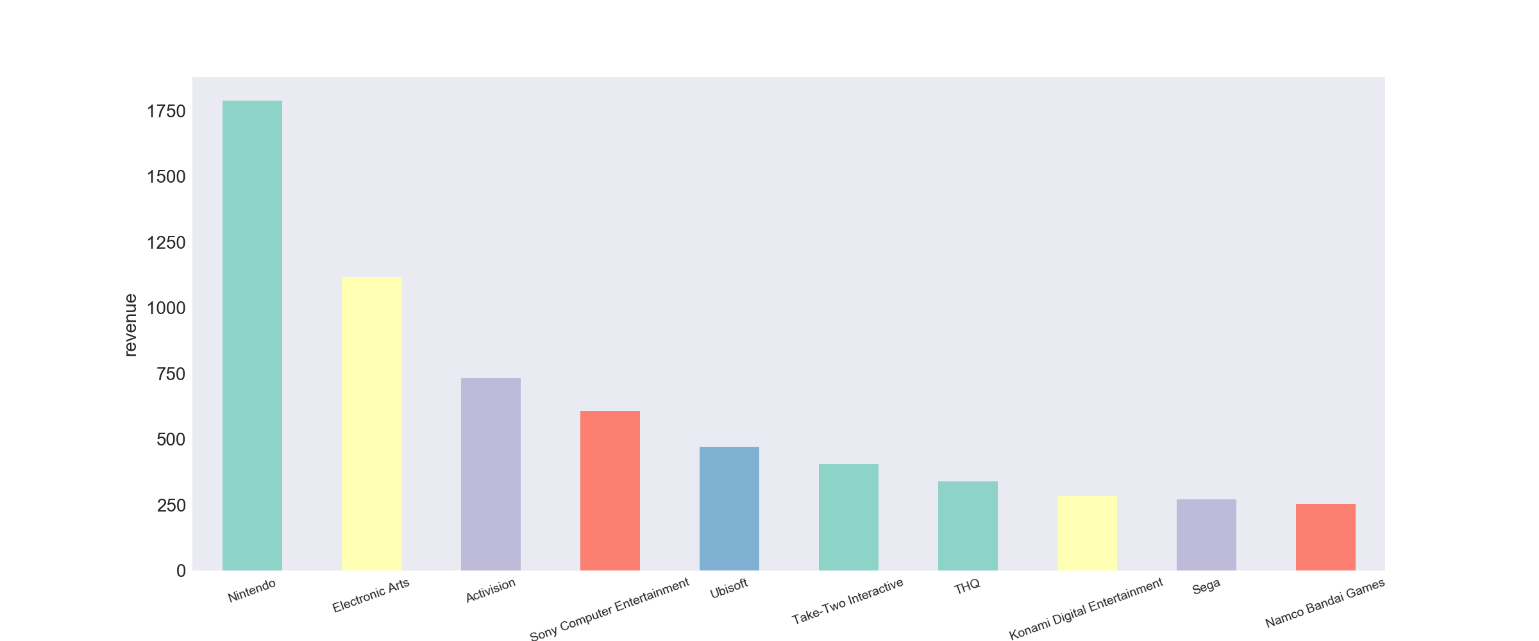
There is change in the positions from the list by number of releases.

Nintendo is Top 1 and EA being Top 2 with nearly half the revenue of the Nintendo.

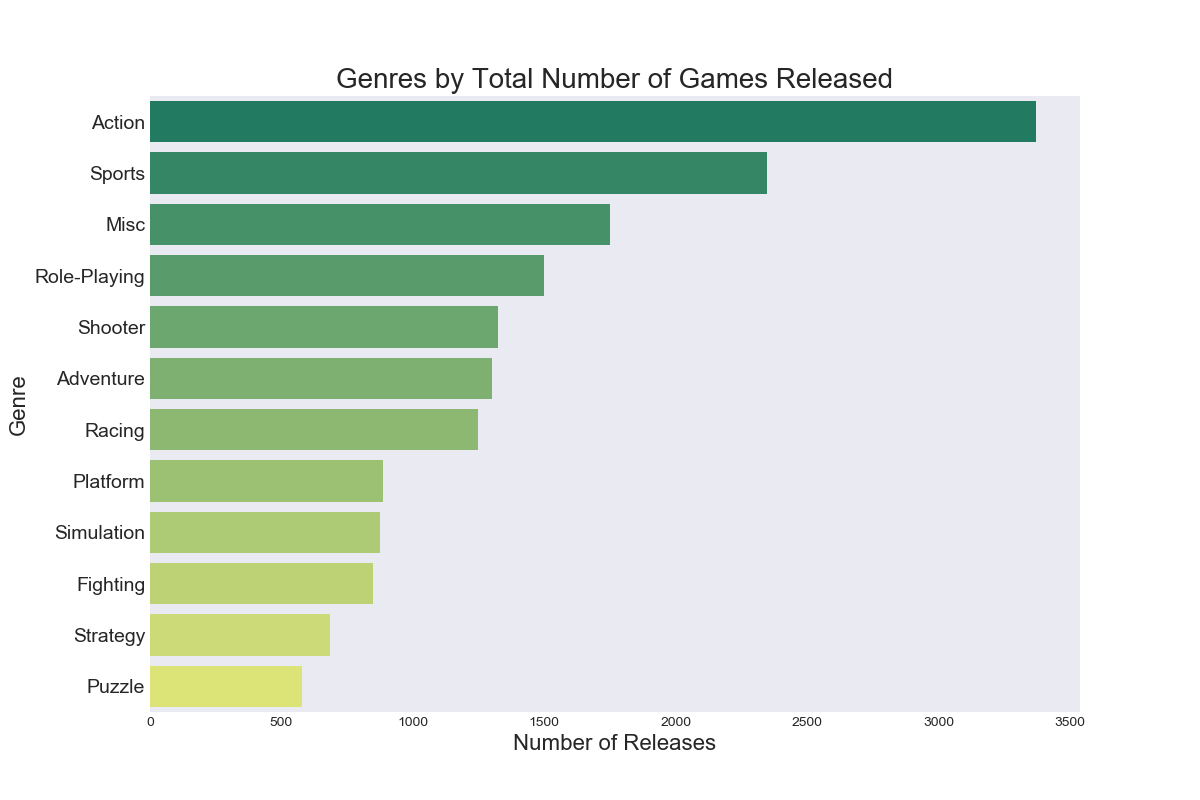
* **We also see this information in pie chart as below:**



* **We also see this information in bar chart as below:**



* **Genres by Number of Releases**

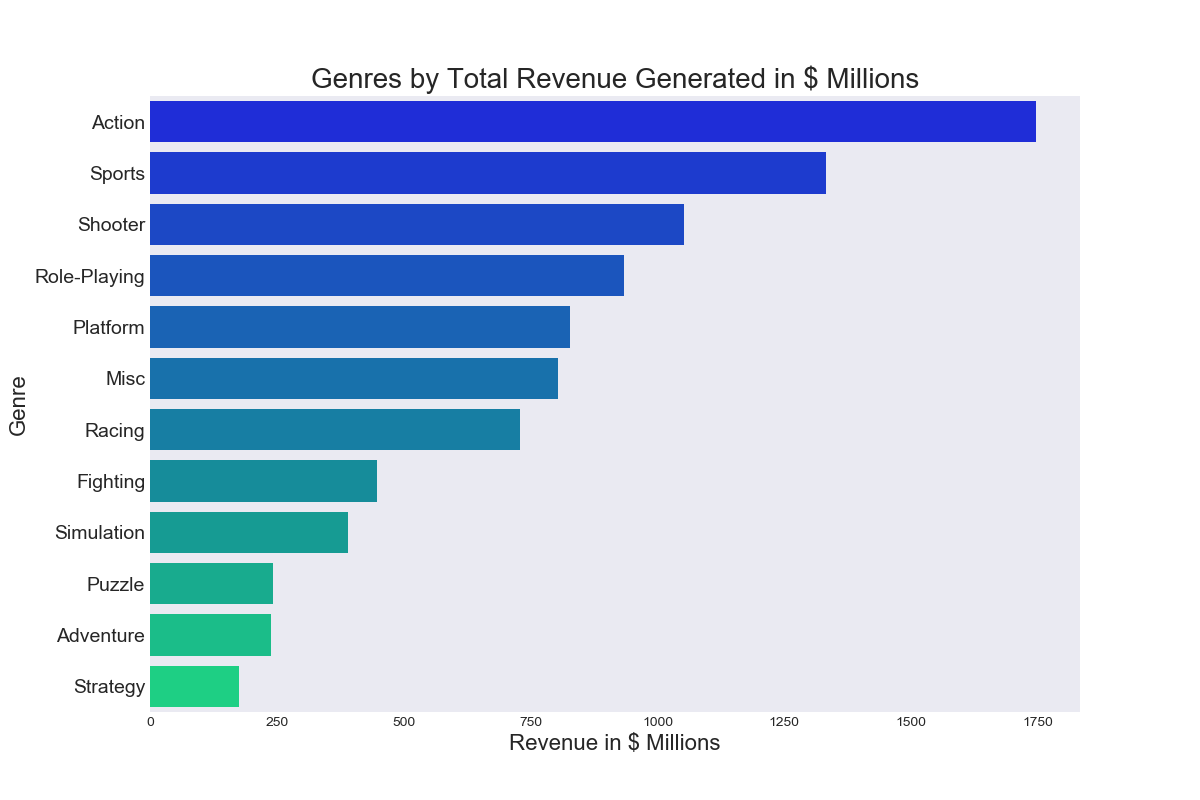


Action Games contribute almost 20% of the games released.

Sports games coming in second.

Overall all top 5 genres contribute to 60% of the games released.

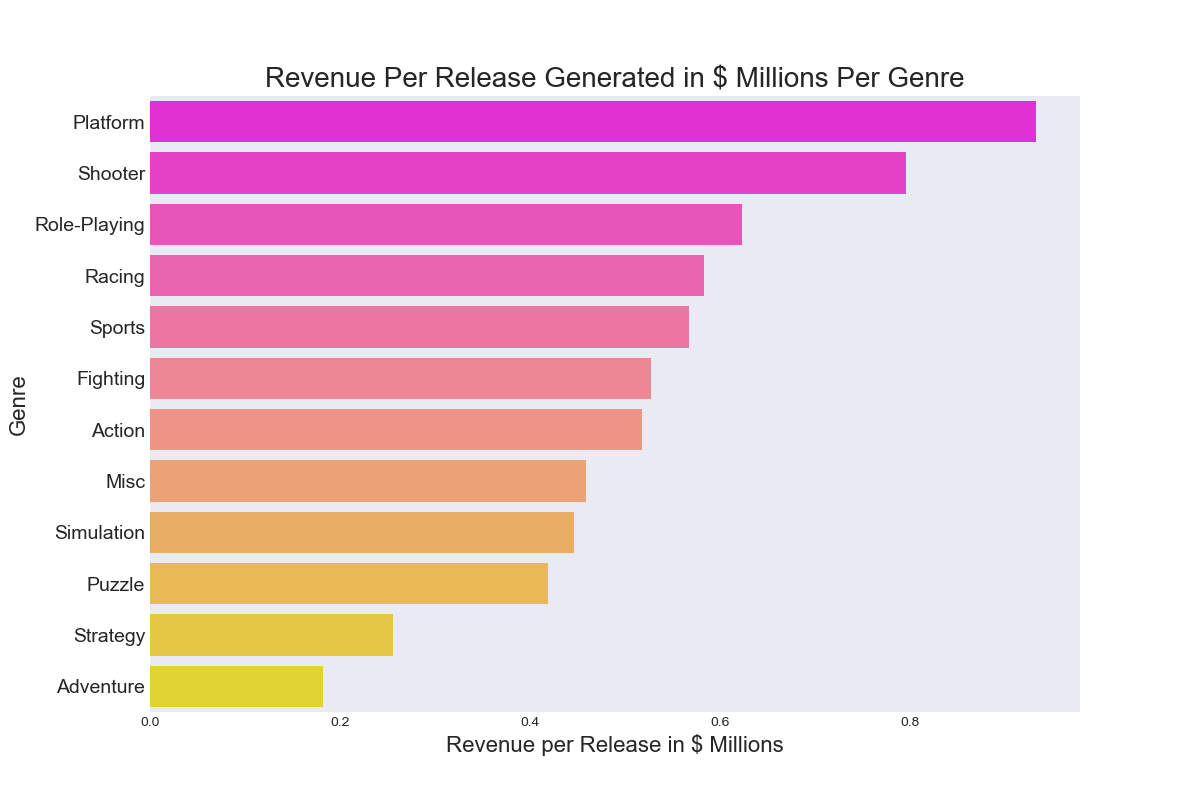
* **Genres by Revenue**



Action Games contribute almost 20% of the revenue.

Sports games coming in second with 15% of the revenue.

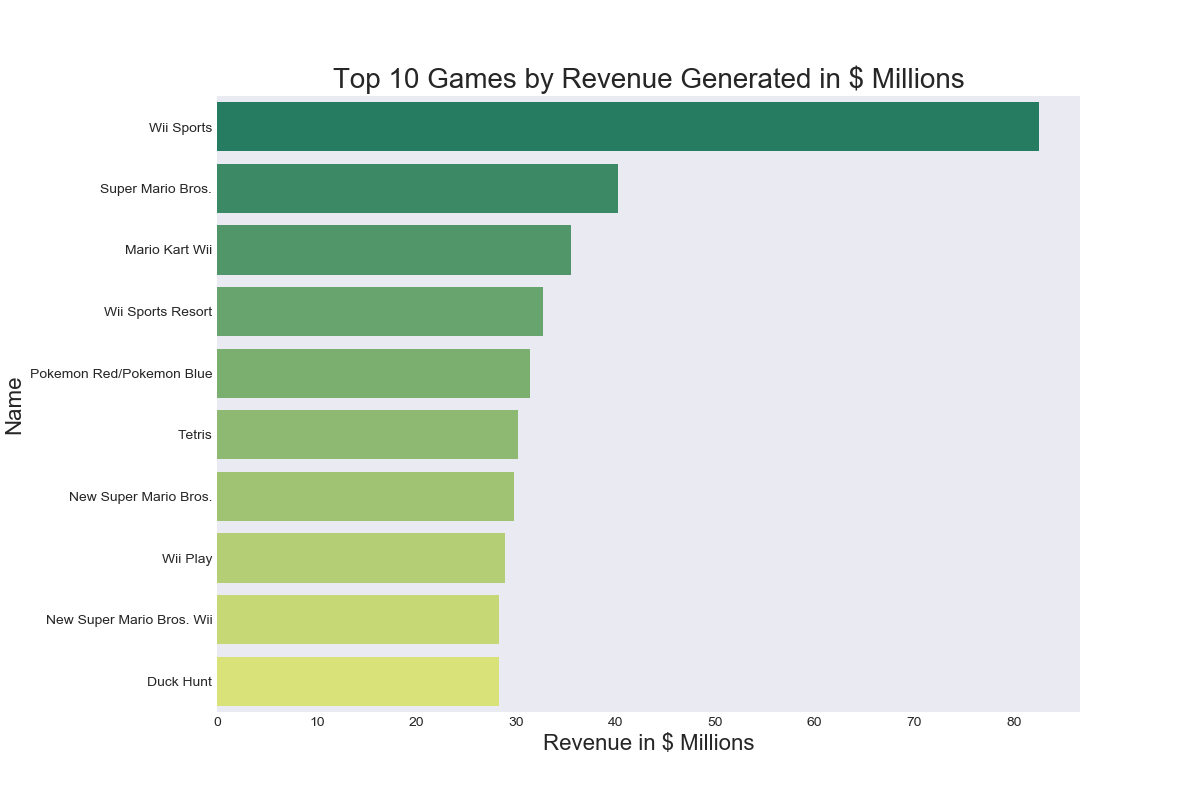
* **Which genre generated most revenue per release?**



Platform is the best genre by revenue per release.

Shooter and Role Playing with second and third best.

* **Top 10 Games by Revenue**



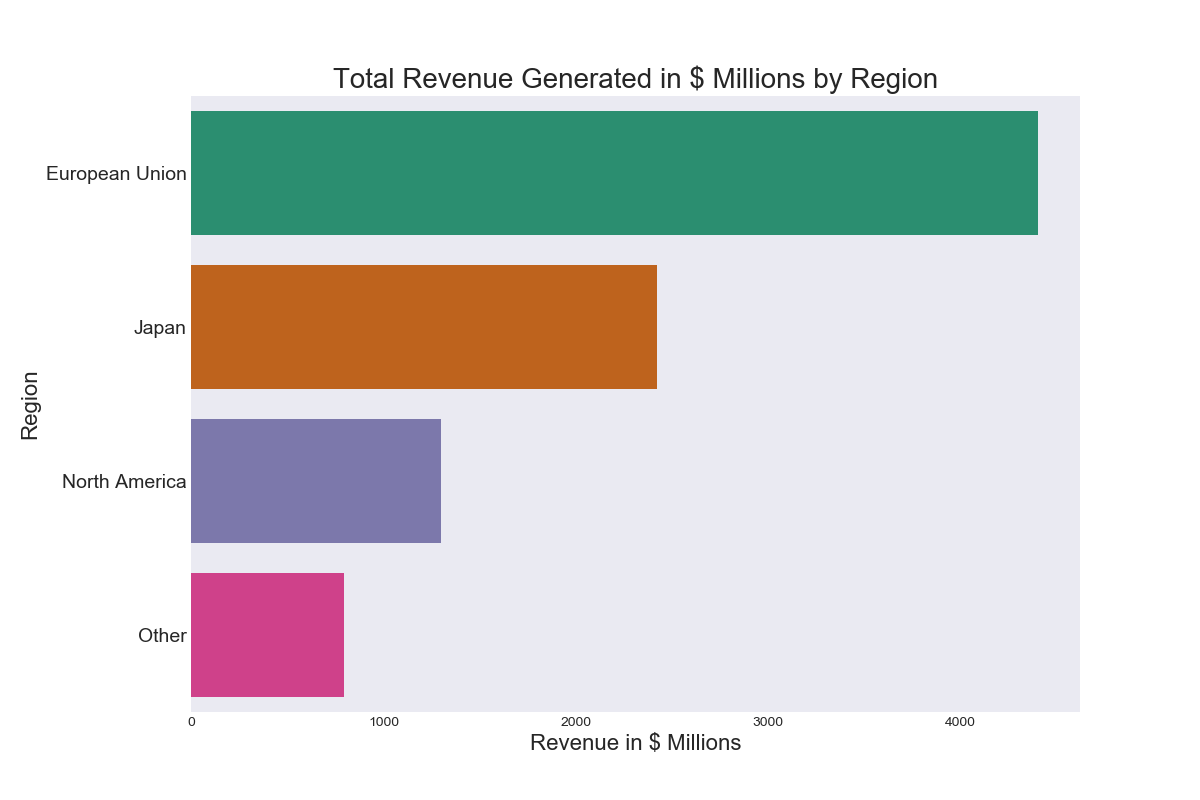
Wii Sports game has 1% of the total revenue with 82.74 million

Super Mario Bros. coming second with 40.24 million.

Top 10 games which have most revenue information is shown below table.

|  |  |  |
| --- | --- | --- |
|  | Name | Global\_Sales |
| 0 | Wii Sports | 82.53 |
| 1 | Super Mario Bros. | 40.24 |
| 2 | Mario Kart Wii | 35.52 |
| 3 | Wii Sports Resort | 32.77 |
| 4 | Pokemon Red/Pokemon Blue | 31.37 |
| 5 | Tetris | 30.26 |
| 6 | New Super Mario Bros. | 29.80 |
| 7 | Wii Play | 28.92 |
| 8 | New Super Mario Bros. Wii | 28.32 |
| 9 | Duck Hunt | 28.31 |

* **Sales Revenue by regions**



North America market contributes 49% of the revenue.

Europe contributes 27% of the revenue.

## Discussion & Conclusion

From the graphs above, I can see that the industry was booming in the late 90's and beyond. It seems as though the industry had become very efficient in production, and was able to produce more volume, which resulted in higher revenue.

Key take-aways from the analysis:

* Over the last few years there is decrease in the games which have crossed 100K copies sales.
* In fact last 5-6 years has seen decrease in the revenue.
* Nintendo is the top most publisher by revenue.
* Action is the top most genre by revenue.
* Action is ruling the market from 2001.
* Play Station is the most popular platform.
* EA is the top most publisher by number of releases.
* Platform Genre generated most revenue per release.
* Wii Sports is the top most game by revenue.
* North America and Europe contribute 75% of the market share by revenue.

## References

<https://www.kaggle.com/kendallgillies/video-game-sales-and-ratings>

<https://www.kaggle.com/gregorut/videogamesales>

<http://seaborn.pydata.org/examples/>

<http://www.wikizero.org/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvVkdfQ2hhcnR6>

<https://www.labri.fr/perso/nrougier/teaching/numpy.100/index.html>

<https://plot.ly/matplotlib/scatter/>

<https://plot.ly/python/line-charts/>

<https://medium.com/python-pandemonium/data-visualization-in-python-line-graph-in-matplotlib-9dfd0016d180>

<https://www.kaggle.com/neilslab/seaborn-visualization>

<https://www.kaggle.com/jiehu15/eda-by-plotly-on-game-sales-dataset>

<https://www.kaggle.com/ohmygod/data-visualization>

<https://www.kaggle.com/kostyabahshetsyan/video-game-sales-visualization>

## Appendix

*# Import related packages*

**import** **pandas** **as** **pd**

**import** **os**

*# Set working directory*

os.chdir('D:/Users/tkartalkaya/PycharmProjects/Project')

*# Load dataset*

vgs = pd.read\_csv("Video\_Games\_Sales\_as\_at\_22\_Dec\_2016.csv")

*# Get general information*

vgs.info()

*"""*

*RangeIndex: 16719 entries, 0 to 16718*

*Data columns (total 16 columns):*

*Name 16717 non-null object*

*Platform 16719 non-null object*

*Year\_of\_Release 16450 non-null float64*

*Genre 16717 non-null object*

*Publisher 16665 non-null object*

*NA\_Sales 16719 non-null float64*

*EU\_Sales 16719 non-null float64*

*JP\_Sales 16719 non-null float64*

*Other\_Sales 16719 non-null float64*

*Global\_Sales 16719 non-null float64*

*Critic\_Score 8137 non-null float64*

*Critic\_Count 8137 non-null float64*

*User\_Score 10015 non-null object*

*User\_Count 7590 non-null float64*

*Developer 10096 non-null object*

*Rating 9950 non-null object*

*dtypes: float64(9), object(7)*

*memory usage: 2.0+ MB*

*"""*

*# Check dimensions*

vgs.shape

*"""(16719, 16)"""*

vgs.dtypes

*"""Output*

*# Name object*

*# Platform object*

*# Year\_of\_Release float64*

*# Genre object*

*# Publisher object*

*# NA\_Sales float64*

*# EU\_Sales float64*

*# JP\_Sales float64*

*# Other\_Sales float64*

*# Global\_Sales float64*

*# Critic\_Score float64*

*# Critic\_Count float64*

*# User\_Score object*

*# User\_Count float64*

*# Developer object*

*# Rating object*

*# dtype: object*

*"""*

*# Check the first 5 rows*

print(vgs.head(5))

*"""*

*Name Platform Year\_of\_Release Genre Publisher \*

*0 Wii Sports Wii 2006.0 Sports Nintendo*

*1 Super Mario Bros. NES 1985.0 Platform Nintendo*

*2 Mario Kart Wii Wii 2008.0 Racing Nintendo*

*3 Wii Sports Resort Wii 2009.0 Sports Nintendo*

*4 Pokemon Red/Pokemon Blue GB 1996.0 Role-Playing Nintendo*

*NA\_Sales EU\_Sales JP\_Sales Other\_Sales Global\_Sales Critic\_Score \*

*0 41.36 28.96 3.77 8.45 82.53 76.0*

*1 29.08 3.58 6.81 0.77 40.24 NaN*

*2 15.68 12.76 3.79 3.29 35.52 82.0*

*3 15.61 10.93 3.28 2.95 32.77 80.0*

*4 11.27 8.89 10.22 1.00 31.37 NaN*

*Critic\_Count User\_Score User\_Count Developer Rating*

*0 51.0 8 322.0 Nintendo E*

*1 NaN NaN NaN NaN NaN*

*2 73.0 8.3 709.0 Nintendo E*

*3 73.0 8 192.0 Nintendo E*

*4 NaN NaN NaN NaN NaN*

*"""*

print( vgs.describe() )

*""""*

*Year\_of\_Release NA\_Sales EU\_Sales JP\_Sales \*

*count 16450.000000 16719.000000 16719.000000 16719.000000*

*mean 2006.487356 0.263330 0.145025 0.077602*

*std 5.878995 0.813514 0.503283 0.308818*

*min 1980.000000 0.000000 0.000000 0.000000*

*25% 2003.000000 0.000000 0.000000 0.000000*

*50% 2007.000000 0.080000 0.020000 0.000000*

*75% 2010.000000 0.240000 0.110000 0.040000*

*max 2020.000000 41.360000 28.960000 10.220000*

*Other\_Sales Global\_Sales Critic\_Score Critic\_Count User\_Count*

*count 16719.000000 16719.000000 8137.000000 8137.000000 7590.000000*

*mean 0.047332 0.533543 68.967679 26.360821 162.229908*

*std 0.186710 1.547935 13.938165 18.980495 561.282326*

*min 0.000000 0.010000 13.000000 3.000000 4.000000*

*25% 0.000000 0.060000 60.000000 12.000000 10.000000*

*50% 0.010000 0.170000 71.000000 21.000000 24.000000*

*75% 0.030000 0.470000 79.000000 36.000000 81.000000*

*max 10.570000 82.530000 98.000000 113.000000 10665.000000 """*

categorical = vgs.dtypes[vgs.dtypes == "object"].index

print(categorical)

*"""*

*Index(['Name', 'Platform', 'Genre', 'Publisher', 'User\_Score', 'Developer',Rating'],dtype='object')*

*"""*

vgs[categorical].describe()

*""""*

*Name Platform Genre Publisher \*

*count 16717 16719 16717 16665*

*unique 11562 31 12 581*

*top Need for Speed: Most Wanted PS2 Action Electronic Arts*

*freq 12 2161 3370 1356*

*User\_Score Developer Rating*

*count 10015 10096 9950*

*unique 96 1696 8*

*top tbd Ubisoft E*

*freq 2425 204 3991*

*"""*

*# Check the first 10 sorted years*

sorted(vgs["Year\_of\_Release"])[0:10]

*""""[1982.0, 1983.0, 1984.0, 1984.0, 1984.0, 1985.0, 1985.0, 1986.0, 1986.0, 1987.0]"""*

vgs["Name"].describe()

*""""*

*count 16717*

*unique 11562*

*top Need for Speed: Most Wanted*

*freq 12*

*Name: Name, dtype: object*

*"""*

*# Check the first 10 names*

vgs["Name"][0:10]

*"""*

*0 Wii Sports*

*1 Super Mario Bros.*

*2 Mario Kart Wii*

*3 Wii Sports Resort*

*4 Pokemon Red/Pokemon Blue*

*5 Tetris*

*6 New Super Mario Bros.*

*7 Wii Play*

*8 New Super Mario Bros. Wii*

*9 Duck Hunt*

*"""*

*# Check unique Platform*

vgs["Platform"].unique()

*"""*

*array(['Wii', 'NES', 'GB', 'DS', 'X360', 'PS3', 'PS2', 'SNES', 'GBA',*

*'PS4', '3DS', 'N64', 'PS', 'XB', 'PC', '2600', 'PSP', 'XOne',*

*'WiiU', 'GC', 'GEN', 'DC', 'PSV', 'SAT', 'SCD', 'WS', 'NG', 'TG16',*

*'3DO', 'GG', 'PCFX'], dtype=object)*

*"""*

*# Platform*

vgs.Platform=vgs.Platform.astype('category')

vgs.Platform.describe()

*"""*

*count 16719*

*unique 31*

*top PS2*

*freq 2161*

*Name: Platform, dtype: object*

*"""*

*# Checking the maximum year value we see that it is 2020 which is an impossible release date.*

year\_data=vgs['Year\_of\_Release']

print("Max Year Value: ",year\_data.max())

*# Looking at the name of the entry with the erroneous year we can simpl search for the games's release date online and replace the current value with the correct release date.*

max\_entry=year\_data.idxmax()

max\_entry=vgs.iloc[max\_entry]

pd.DataFrame(max\_entry).T

*"""*

*Name Platform Year\_of\_Release Genre Publisher \*

*5936 Imagine: Makeup Artist DS 2020 Simulation Ubisoft*

*NA\_Sales EU\_Sales JP\_Sales Other\_Sales Global\_Sales Critic\_Score \*

*5936 0.27 0 0 0.02 0.29 NaN*

*Critic\_Count User\_Score User\_Count Developer Rating*

*5936 NaN tbd NaN Ubisoft E*

*"""*

*# Below I check the number of games(rows) and the number of unique publishers, platforms and genres to get an idea of how our the games in the dataset are distributed categorically.*

print("Number of games: ", len(vgs))

publishers = vgs['Publisher'].unique()

print("Number of publishers: ", len(publishers))

plotforms = vgs['Platform'].unique()

print("Number of plotforms: ", len(plotforms))

genres = vgs['Genre'].unique()

print("Number of genres: ", len(genres))

*# What are top 20 highest grossing games since 1980?*

vgs.groupby(['Name','Year\_of\_Release'])['Global\_Sales'].sum().sort\_values(ascending=**False**)[:20]

*"""  
Name Year\_of\_Release  
Wii Sports 2006.0 82.53  
Super Mario Bros. 1985.0 40.24  
Grand Theft Auto V 2013.0 37.31  
Mario Kart Wii 2008.0 35.52  
Wii Sports Resort 2009.0 32.77  
Pokemon Red/Pokemon Blue 1996.0 31.37  
Call of Duty: Modern Warfare 3 2011.0 30.59  
Tetris 1989.0 30.26  
New Super Mario Bros. 2006.0 29.80  
Call of Duty: Black Ops II 2012.0 29.40  
Call of Duty: Black Ops 2010.0 29.19  
Wii Play 2006.0 28.92  
New Super Mario Bros. Wii 2009.0 28.32  
Duck Hunt 1984.0 28.31  
Call of Duty: Ghosts 2013.0 27.40  
Call of Duty: Black Ops 3 2015.0 25.68  
Call of Duty: Modern Warfare 2 2009.0 24.95  
Nintendogs 2005.0 24.67  
Mario Kart DS 2005.0 23.21  
Pokemon Gold/Pokemon Silver 1999.0 23.10  
Name: Global\_Sales, dtype: float64  
"""*

*# I do a simple null value check.*

print(vgs.isnull().sum())

*"""*

*Name 2*

*Platform 0*

*Year\_of\_Release 269*

*Genre 2*

*Publisher 54*

*NA\_Sales 0*

*EU\_Sales 0*

*JP\_Sales 0*

*Other\_Sales 0*

*Global\_Sales 0*

*Critic\_Score 8582*

*Critic\_Count 8582*

*User\_Score 6704*

*User\_Count 9129*

*Developer 6623*

*Rating 6769*

*dtype: int64*

*"""*

*##VISUALIZATIONS*

*# Below I create a simple column chart to represent the total 'Global\_Sales' of video games per year. I get our data by data our dataframe of all video games sales, grouping it by 'Year' and then calling .sum() to get totals for each year. This creates a dataframe with our years as the index or row names and the entries our total sales for that year.*

*# In the dataset, the index representing years are floating point numbers eg "2006.0" not "2006". I get our x-axis entries by taking these values as integers.*

*# Once data is ready I simply pass our x and y variables to our Seaborn barplot function. I also set our x-label names, the title and I also rotate our xticklabels and change their fontsize.*

**import** **matplotlib.pyplot** **as** **plt**

**import** **seaborn** **as** **sns**

sns.set\_style("dark")

sns.despine()

y = vgs.groupby(['Year\_of\_Release']).sum()

y = y['Global\_Sales']

x = y.index.astype(int)

plt.figure(figsize=(12,8))

ax = sns.barplot(y = y, x = x)

ax.set\_xlabel(xlabel='$ Millions', fontsize=16)

ax.set\_xticklabels(labels = x, fontsize=12, rotation=50)

ax.set\_ylabel(ylabel='Year', fontsize=16)

ax.set\_title(label='Game Sales in $ Millions Per Year', fontsize=20)

plt.show()

*# Below I create a simple column chart to represent the total total number of releases of video games per year, but with a twist, it is oriented horizontally, which means our Year entries, which usually would be our x-axis, are now on the y-axis and the count of “Global\_Sales” entries, which would usually be on the y-axis, are now on the x-axis.*

*# The below chart that represents the number of games released each year.*

x = vgs.groupby(['Year\_of\_Release']).count()

x = x['Global\_Sales']

y = x.index.astype(int)

plt.figure(figsize=(12,8))

colors = sns.color\_palette("muted")

ax = sns.barplot(y = y, x = x, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Number of releases', fontsize=16)

ax.set\_ylabel(ylabel='Year', fontsize=16)

ax.set\_title(label='Game Releases Per Year', fontsize=20)

plt.show()

*# Below I create 2 plots: the top 10 publishers by the number of games they have released and the total sales generated, the only difference being the we apply count() in the first instance and sum() in the second.*

*# I groupby “Publisher”, choose a single column, sort in descending order and then choose the top 10 values. We also apply the sequential colour maps "spring" to the plots.*

*# Top 10 Total Publisher Games Released*

data = vgs.groupby(['Publisher']).count().iloc[:,0]

data = pd.DataFrame(data.sort\_values(ascending=**False**))[0:10]

publishers = data.index

data.columns = ['Releases']

colors = sns.color\_palette("spring", len(data))

plt.figure(figsize=(12,8))

ax = sns.barplot(y = publishers , x = 'Releases', data=data, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Number of Releases', fontsize=16)

ax.set\_ylabel(ylabel='Publisher', fontsize=16)

ax.set\_title(label='Top 10 Total Publisher Games Released', fontsize=20)

ax.set\_yticklabels(labels = publishers, fontsize=14)

plt.show()

*# Top 10 Total Publisher Game Revenue*

data = vgs.groupby(['Publisher']).sum()['Global\_Sales']

data = pd.DataFrame(data.sort\_values(ascending=**False**))[0:10]

publishers = data.index

data.columns = ['Global Sales']

colors = sns.color\_palette("spring", len(data))

plt.figure(figsize=(12,8))

ax = sns.barplot(y = publishers , x = 'Global Sales', data=data, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Revenue in $ Millions', fontsize=16)

ax.set\_ylabel(ylabel='Publisher', fontsize=16)

ax.set\_title(label='Top 10 Total Publisher Game Revenue', fontsize=20)

ax.set\_yticklabels(labels = publishers, fontsize=14)

plt.show()

*## PIE CHART*

*# I wonder which companys earn most. I prepare a pie for publisher releases. The first is Nintendo, the second is EA and the third is Activision.*

**import** **seaborn** **as** **sns**

**import** **matplotlib.pyplot** **as** **plt**

sns.set\_palette("Set3",5)

sns.palplot(sns.color\_palette())

sns.set\_context("dark")

f,ax=plt.subplots(1,1,figsize=(4,4))

vgs.groupby("Publisher")["Global\_Sales"].sum().sort\_values(ascending=**False**)[:10].plot.pie()

ax.set\_ylabel("")

plt.tight\_layout()

*#The global game-sales revenue is divided by Nintendo, EA, Activisions.*

f,ax=plt.subplots(1,1,figsize=(12,5))

vgs.groupby("Publisher")["Global\_Sales"].sum().sort\_values(ascending=**False**)[:10].plot(kind="bar")

ax.set\_ylabel("revenue")

ticks=plt.setp(ax.get\_xticklabels(), rotation=20, fontsize=7)

*# Below I create three plots for genres: total number of games released, total revenue generated and revenue per game released. I compute revenue per game by concatenating our total revenue generated and games released data frames and then dividing the former column by the latter to get Revenue/Release. For our genres, "rel" is the count of number of games released and "rev" is the total revenue generated.*

*# I groupby, apply sum() or count() and choose the first column using .iloc() when I just want to count entries and ['Global\_Sales'] when I want to sum sales. Finally I sort our values and then plot our data, testing a couple of different colour palettes.*

*# Genres by Total Number of Games Released*

rel = vgs.groupby(['Genre']).count().iloc[:,0]

rel = pd.DataFrame(rel.sort\_values(ascending=**False**))

genres = rel.index

rel.columns = ['Releases']

colors = sns.color\_palette("summer", len(rel))

plt.figure(figsize=(12,8))

ax = sns.barplot(y = genres , x = 'Releases', data=rel, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Number of Releases', fontsize=16)

ax.set\_ylabel(ylabel='Genre', fontsize=16)

ax.set\_title(label='Genres by Total Number of Games Released', fontsize=20)

ax.set\_yticklabels(labels = genres, fontsize=14)

plt.show()

*# Genres by Total Revenue Generated in $ Millions*

rev = vgs.groupby(['Genre']).sum()['Global\_Sales']

rev = pd.DataFrame(rev.sort\_values(ascending=**False**))

genres = rev.index

rev.columns = ['Revenue']

colors = sns.color\_palette('winter', len(rev))

plt.figure(figsize=(12,8))

ax = sns.barplot(y = genres , x = 'Revenue', data=rev, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Revenue in $ Millions', fontsize=16)

ax.set\_ylabel(ylabel='Genre', fontsize=16)

ax.set\_title(label='Genres by Total Revenue Generated in $ Millions', fontsize=20)

ax.set\_yticklabels(labels = genres, fontsize=14)

plt.show()

*# Revenue Per Release Generated in $ Millions Per Genre*

data = pd.concat([rev, rel], axis=1)

data = pd.DataFrame(data['Revenue'] / data['Releases'])

data.columns = ['Revenue Per Release']

data = data.sort\_values(by='Revenue Per Release',ascending=**False**)

genres = data.index

colors = sns.color\_palette("spring", len(data))

plt.figure(figsize=(12,8))

ax = sns.barplot(y = genres , x = 'Revenue Per Release', data=data, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Revenue per Release in $ Millions', fontsize=16)

ax.set\_ylabel(ylabel='Genre', fontsize=16)

ax.set\_title(label='Revenue Per Release Generated in $ Millions Per Genre', fontsize=20)

ax.set\_yticklabels(labels = genres, fontsize=14)

plt.show()

*# Below I plot the top 10 games by the sales that they have generated since inception. I create a pivot table with the entries that I need and then change it into a data frame. I also sort the values in descending order and select the first ten values.*

*# Top 10 Games by Revenue Generated in $ Millions*

data = pd.concat([vgs['Name'][0:10], vgs['Global\_Sales'][0:10]], axis=1)

plt.figure(figsize=(12,8))

colors = sns.color\_palette("summer", len(data))

ax = sns.barplot(y = 'Name' , x = 'Global\_Sales', data=data, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Revenue in $ Millions', fontsize=16)

ax.set\_ylabel(ylabel='Name', fontsize=16)

ax.set\_title(label='Top 10 Games by Revenue Generated in $ Millions', fontsize=20)

ax.set\_yticklabels(labels = games, fontsize=14)

plt.show()

*# Below I plot the total revenue generated by the four regions that the "Global\_Sales" column is made up of: namely, the European Union, Japan, North America and other.*

*# First I sum all the columns in our original data frame. Then I create a dataframe which has regions by selecting columns from our summed data variable. I also create our own regions index because we want the full names for the regions in our plots, not "EU\_Sales" etc.*

*# Total Revenue Generated in $ Millions by Region*

data = vgs.sum()

data = pd.DataFrame([data['EU\_Sales'], data['JP\_Sales'], data['NA\_Sales'], data['Other\_Sales']])

regions = ['European Union', 'Japan', 'North America', 'Other']

data.index = regions

data.columns = ['Revenue']

data = data.sort\_values(by='Revenue', ascending=**False**)

plt.figure(figsize=(12,8))

colors = sns.color\_palette("Dark2", len(data))

ax = sns.barplot(y = regions , x = 'Revenue', data=data, orient='h', palette=colors)

ax.set\_xlabel(xlabel='Revenue in $ Millions', fontsize=16)

ax.set\_ylabel(ylabel='Region', fontsize=16)

ax.set\_title(label='Total Revenue Generated in $ Millions by Region', fontsize=20)

ax.set\_yticklabels(labels = regions, fontsize=14)

plt.show()