

D214 Data Analytics Capstone Performance Task 2

Sean Simmons

WDU Data Analytics

MSDA D214

March 2023

I have included the code file and an html and pdf of the code, to view at your convenience and preference.

Research Question

A. Summarize the original real-data research question you identified in task 1. Your summary should include justification for the research question you identified in task 1, a description of the context in which the research question exists, and a discussion of your hypothesis.

A hypothetical business wants to gain information about video game sales to be able to predict future global sales. One question from real-world data I would ask to support this goal is, “Can we use video game sales information to create a predictive multiple linear regression model to predict future global sales?” The data used to answer this question is the real public sales information dataset sourced from Kaggle (*Video Game Sales* - Kaggle).

By asking this question, I will be informing the business of a key metric that can then be used to gain actionable insight of global sales and expected global sales based on the characteristics of their video game. The justification of this question is that it informs a key business metric and gives the company a tool to gain actionable insight from.

The context of this question exists in that hypothetical video game business wants to produce a game and know an estimation of what future global sales could look like and what characteristics a game with high global sales looks like. I have two hypotheses associated with this question:

Null hypothesis-. We can not create a model to statistically significantly predict video game global sales.

Alternate Hypothesis- We can create a model to statistically significantly predict video game global sales.

Data Collection

B. Report on your data-collection process by describing the relevant data you collected, discussing one advantage and one disadvantage of the data-gathering methodology you used, and discussing how you overcame any challenges you encountered during the process of collecting your data.

I did not physically measure data points for this project. However, the relevant data I collected was from an open source kaggle dataset that has tracked the sales and characteristics of video games over a long period of time. I knew the hypothetical question I wanted to ask after finding this dataset when I was browsing for topic ideas for my capstone. After examining the data and general question, I was able to narrow down my focus to the specific techniques I wanted to use and could use within the parameters of this dataset. The data can be examined by viewing the first 5 rows included in the preparation summary.

One challenge of collecting this data was that I first needed to create a kaggle account to download it from the website and I was unaware of the quality of data. After


that, it was very easy to use python to upload my data into a data frame in my coding environment and assess the state the data was in.

Data Extraction and Preparation

C. Describe your data-extraction and -preparation process and provide screenshots to illustrate *each* step. Explain the tools and techniques you used for data extraction and data preparation, including how these tools and techniques were used on the data. Justify why you used these particular tools and techniques, including one advantage and one disadvantage when they are used with your data-extraction and -preparation methods.

Data Extraction

In order to extract the data, you need to go to kaggle.com and create an account. Then, I searched for viable datasets and found the video game sales dataset pictured on the next page. I clicked the download button and downloaded the dataset to my device.





GREGORYSMITH ·
UPDATED 6 YEARS AGO

▲ 5102

New Notebook


Download (390 kB)





Video Game Sales

Analyze sales data from more than 16,500 games.



Data Card Code (1242) Discussion (35)

About Dataset

This dataset contains a list of video games with sales greater than 100,000 copies. It was generated by a scrape of vgchartz.com.

Fields include

- Rank - Ranking of overall sales
- Name - The games name
- Platform - Platform of the games release (i.e. PC,PS4, etc.)
- Year - 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)

Usability ⓘ
5.88

License
Unknown

Expected update frequency
Not specified

I used kaggle because it is a widely known and available trusted source of datasets. This specific dataset is also well explained and easy to understand. Smith, the original author, listed no necessary citations for this work, but I have included an APA citation of the dataset, nonetheless. The descriptions of each column can be found in the kaggle site to help in our data exploration. One disadvantage of kaggle is that the authenticity of the data is not 100% known, although it can be trusted, it is not a guarantee. Once I decided to use this data, it was then uploaded to my coding environment using the following code:

```
#Import pandas so we can import our video game file
#Jupyter Lab 3.44, Python 3
#If you do not have the packages available to your machine, please follow the usual procedures for downloading, c
import pandas as pd
df = pd.read_csv(r'C:\Users\e202271009\Documents\D214\vg-sales.csv')
#Your path will be different
#Import Other Packages, these packages allow us to perform statistical analysis and plot visuals for our data set
import numpy as np
import scipy as sp
import scipy.stats as stats
import pylab
from statsmodels.formula.api import ols
import statistics
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sb
# Scikit
import sklearn
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

I am using python version 3 in Jupyter labs version 3.44. To import the data pandas had to be imported. I chose to use this coding environment and pandas for the same reason- they are widely accepted as an easy and straightforward tool to perform a wide range of data analysis tasks. One disadvantage of python and jupyter labs is that it is considered a complicated language when installing the numerous packages needed, whereas other languages like R have many built in capabilities. Despite this, I am more familiar with and wanted to perform this analysis in python (Prasanna 2021).

Data Preparation

Now that the data is uploaded into my coding environment, we need to view the raw data for exploratory and information gathering needs. We will gather the data types, first 5 rows and counts. This is a beginning step in exploratory data analysis (EDA) and one disadvantage is that we do not yet get a full picture or idea of any trends or potential mitigations we need to make:

```
#Let's view the dataset
#View Data Types
print(df.select_dtypes(include="float").info())
print(df.select_dtypes(include="integer").info())
print(df.select_dtypes(include="object").info())

#View example of the information in the dataset
print(df.head(5))
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16598 entries, 0 to 16597
Data columns (total 6 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Year            16327 non-null  float64
1   NA_Sales        16598 non-null  float64
2   EU_Sales        16598 non-null  float64
3   JP_Sales        16598 non-null  float64
4   Other_Sales     16598 non-null  float64
5   Global_Sales    16598 non-null  float64
dtypes: float64(6)
memory usage: 778.2 KB
None
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16598 entries, 0 to 16597
Data columns (total 1 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Rank            16598 non-null  int64
dtypes: int64(1)
memory usage: 129.8 KB
None
```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16598 entries, 0 to 16597
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Name        16598 non-null  object
1   Platform    16598 non-null  object
2   Genre       16598 non-null  object
3   Publisher   16540 non-null  object
dtypes: object(4)
memory usage: 518.8+ KB
None

```

	Rank	Name	Platform	Year	Genre	Publisher
0	1	Wii Sports	Wii	2006.0	Sports	Nintendo
1	2	Super Mario Bros.	NES	1985.0	Platform	Nintendo
2	3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo
3	4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo
4	5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo

```


```

	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	41.49	29.02	3.77	8.46	82.74
1	29.08	3.58	6.81	0.77	40.24
2	15.85	12.88	3.79	3.31	35.82
3	15.75	11.01	3.28	2.96	33.00
4	11.27	8.89	10.22	1.00	31.37

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16598 entries, 0 to 16597
Data columns (total 11 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Rank        16598 non-null  int64
1   Name        16598 non-null  object
2   Platform    16598 non-null  object
3   Year        16327 non-null  float64
4   Genre       16598 non-null  object
5   Publisher   16540 non-null  object
6   NA_Sales    16598 non-null  float64
7   EU_Sales    16598 non-null  float64
8   JP_Sales    16598 non-null  float64
9   Other_Sales 16598 non-null  float64
10  Global_Sales 16598 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB

```


Now, we must explore the data for missing information. Based on the nature of the data a 0 is an acceptable value and we should not look to mitigate outliers, since we want all of the sales information represented in our multiple linear regression analysis. Later we will see any outliers and continue our decision of leaving them alone in our histograms, but if we find any null values we must fill them and then check for nulls again to verify. One disadvantage of this stage of EDA is that the outliers may indeed present issues with our modeling in the future, which is a risk I am taking here for my choice of analysis (Prasanna 2021):

```
#Check for any missing values  
df.isna().sum()
```

```
Rank          0  
Name          0  
Platform      0  
Year         271  
Genre         0  
Publisher     58  
NA_Sales      0  
EU_Sales      0  
JP_Sales      0  
Other_Sales   0  
Global_Sales  0  
dtype: int64
```

```
# Remove nulls from where it says "True" above this cell  
df = df.dropna(subset=['Year', 'Publisher'])
```

```
#Check for any missing values  
df.isna().sum()
```

```
Rank          0  
Name          0  
Platform      0  
Year          0  
Genre         0  
Publisher     0  
NA_Sales      0  
EU_Sales      0  
JP_Sales      0  
Other_Sales   0  
Global_Sales  0  
dtype: int64
```

Now I want to drop the only column that will not aid us in any way, the name column. This will allow us to focus our analysis and one disadvantage is that we are losing a small fraction of integrity from the original dataset.

```
#Drop irrelevant columns from the dataset  
df = df.drop(['Name'], axis=1)
```

Afterwards, we now need to change our data types. From the initial EDA I know we have a lot of string data. For our hypothesis and multiple linear regression (MLR) we need numerical data. So we will view the unique values of each column and then encode each of our string columns into numeric ones so we can use it in our model. One disadvantage of this is that we lose the identifying labels at a quick glance, and the organization will need to view the original columns and our label list to see what the numeric values represent:

```
[74]: for col in df:
      print(df[col].unique())
```

```
[ 1    2    3 ... 16598 16599 16600]
['Wii' 'NES' 'GB' 'DS' 'X360' 'PS3' 'PS2' 'SNES' 'GBA' '3DS' 'PS4' 'N64'
 'PS' 'XB' 'PC' '2600' 'PSP' 'XOne' 'GC' 'WiiU' 'GEN' 'DC' 'PSV' 'SAT'
 'SCD' 'WS' 'NG' 'TG16' '3DO' 'GG' 'PCFX']
[2006. 1985. 2008. 2009. 1996. 1989. 1984. 2005. 1999. 2007. 2010. 2013.
 2004. 1990. 1988. 2002. 2001. 2011. 1998. 2015. 2012. 2014. 1992. 1997.
 1993. 1994. 1982. 2003. 1986. 2000. 1995. 2016. 1991. 1981. 1987. 1980.
 1983. 2020. 2017.]
['Sports' 'Platform' 'Racing' 'Role-Playing' 'Puzzle' 'Misc' 'Shooter'
 'Simulation' 'Action' 'Fighting' 'Adventure' 'Strategy']
['Nintendo' 'Microsoft Game Studios' 'Take-Two Interactive'
 'Sony Computer Entertainment' 'Activision' 'Ubisoft' 'Bethesda Softworks'
 'Electronic Arts' 'Sega' 'SquareSoft' 'Atari' '505 Games' 'Capcom'
 'GT Interactive' 'Konami Digital Entertainment'
 'Sony Computer Entertainment Europe' 'Square Enix' 'LucasArts'
 'Virgin Interactive' 'Warner Bros. Interactive Entertainment'
 'Universal Interactive' 'Eidos Interactive' 'RedOctane' 'Vivendi Games'
 'Enix Corporation' 'Namco Bandai Games' 'Palcom' 'Hasbro Interactive'
 'THQ' 'Fox Interactive' 'Acclaim Entertainment' 'MTV Games'
 'Disney Interactive Studios' 'Majesco Entertainment' 'Codemasters'
 'Red Orb' 'Level 5' 'Arena Entertainment' 'Midway Games' 'JVC'
 'Deep Silver' '989 Studios' 'NCSOFT' 'UEP Systems' 'Parker Bros.' 'Maxis'
 'Imagic' 'Tecmo Koei' 'Valve Software' 'ASCII Entertainment' 'Mindscape'
 'Infogrames' 'Unknown' 'Square' 'Valve' 'Activision Value' 'Banpresto'
 'D3Publisher' 'Oxygen Interactive' 'Red Storm Entertainment'
 'Video System' 'Hello Games' 'Global Star' 'Gotham Games'
 'Westwood Studios' 'GungHo' 'Crave Entertainment' 'Hudson Soft' 'Coleco'
 'Rising Star Games' 'Atlus' 'TDK Mediactive' 'ASC Games' 'Zoo Games'
 'Accolade' 'Sony Online Entertainment' '3DO' 'RTL' 'Natsume'
 'Focus Home Interactive' 'Alchemist' 'Black Label Games'
 'SouthPeak Games' 'Mastertronic' 'Ocean' 'Zoo Digital Publishing'
 'Psygnosis' 'City Interactive' 'Empire Interactive' 'Success' 'Compile'
 'Russel' 'Taito' 'Agetec' 'GSP' 'Microprose' 'Play It'
 'Slightly Mad Studios' 'Tomy Corporation' 'Sammy Corporation'
 'Koch Media' 'Game Factory' 'Titus' 'Marvelous Entertainment' 'Genki']
```

```
#Now we need to convert our categorical data into numeric data using the LabelEncoder
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
label = le.fit_transform(df['Platform'])
print(label)
df.drop("Platform", axis=1, inplace=True)
df["Platform"] = label
df
```

[26 11 26 ... 16 4 6]

	Rank	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_S
0	1	2006.0	Sports	Nintendo	41.49	29.02	3.77	
1	2	1985.0	Platform	Nintendo	29.08	3.58	6.81	
2	3	2008.0	Racing	Nintendo	15.85	12.88	3.79	
3	4	2009.0	Sports	Nintendo	15.75	11.01	3.28	
4	5	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	
...
16593	16596	2002.0	Platform	Kemco	0.01	0.00	0.00	
16594	16597	2003.0	Shooter	Infogrames	0.01	0.00	0.00	
16595	16598	2008.0	Racing	Activision	0.00	0.00	0.00	
16596	16599	2010.0	Puzzle	7G//AMES	0.00	0.01	0.00	
16597	16600	2003.0	Platform	Wanadoo	0.01	0.00	0.00	

16291 rows × 10 columns

```
#Next variable
from sklearn.preprocessing import LabelEncoder
le2 = LabelEncoder()
label2 = le2.fit_transform(df['Genre'])
print(label2)
df.drop("Genre", axis=1, inplace=True)
df["Genre"] = label2
df
```

```
[10  4  6 ...  6  5  4]
```

	Rank	Year	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Glo
0	1	2006.0	Nintendo	41.49	29.02	3.77	8.46	
1	2	1985.0	Nintendo	29.08	3.58	6.81	0.77	
2	3	2008.0	Nintendo	15.85	12.88	3.79	3.31	
3	4	2009.0	Nintendo	15.75	11.01	3.28	2.96	
4	5	1996.0	Nintendo	11.27	8.89	10.22	1.00	
...
16593	16596	2002.0	Kemco	0.01	0.00	0.00	0.00	
16594	16597	2003.0	Infogrames	0.01	0.00	0.00	0.00	
16595	16598	2008.0	Activision	0.00	0.00	0.00	0.00	
16596	16599	2010.0	7G//AMES	0.00	0.01	0.00	0.00	
16597	16600	2003.0	Wanadoo	0.01	0.00	0.00	0.00	

16291 rows × 10 columns

```
#Now we need to convert our categorical data into numeric data using the LabelEncoder
from sklearn.preprocessing import LabelEncoder
le3 = LabelEncoder()
label3 = le3.fit_transform(df['Genre'])
print(label3)
df.drop("Publisher", axis=1, inplace=True)
df["Publisher"] = label3
df
```

[10 4 6 ... 6 5 4]

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Pl
0	1	2006.0	41.49	29.02	3.77	8.46	82.74	
1	2	1985.0	29.08	3.58	6.81	0.77	40.24	
2	3	2008.0	15.85	12.88	3.79	3.31	35.82	
3	4	2009.0	15.75	11.01	3.28	2.96	33.00	
4	5	1996.0	11.27	8.89	10.22	1.00	31.37	
...
16593	16596	2002.0	0.01	0.00	0.00	0.00	0.01	
16594	16597	2003.0	0.01	0.00	0.00	0.00	0.01	
16595	16598	2008.0	0.00	0.00	0.00	0.00	0.01	
16596	16599	2010.0	0.00	0.01	0.00	0.00	0.01	
16597	16600	2003.0	0.01	0.00	0.00	0.00	0.01	

16291 rows × 10 columns

Okay, now I want to perform the last of our EDA. The first part is to pull the means and medians of our dataset. Next, we can view histograms and boxplots to gain an understanding of our individual variables. Lastly, we will visualize scatter plots between our target variable, global sales, and the other variables to visualize any trends. One disadvantage of this is that it can be too much information and it can mislead our ideas of what the final MLR model should look like. The trends and relationships identified here may not present itself in the model in the same way we see it. For example, the boxplots show several outliers for many variables, but we have to

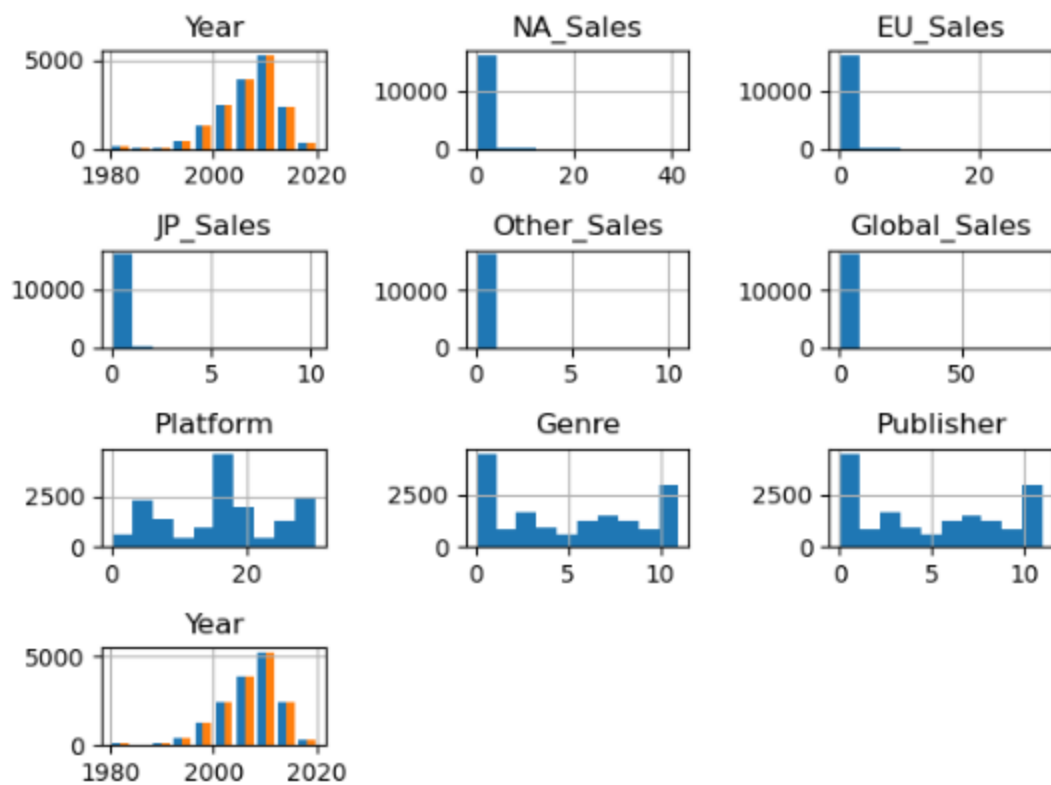
remember we are choosing to keep all data points in our dataset to preserve the integrity of the data. Although, this is a very small mental disadvantage and can be ignored by remaining unbiased:

```
#Mean Values in the Distribution  
print (df.mean())  
#Median Values in the Distribution  
print (df.median())
```

```
Rank          8290.190228  
Year          2006.405561  
NA_Sales      0.265647  
EU_Sales      0.147731  
JP_Sales      0.078833  
Other_Sales   0.048426  
Global_Sales  0.540910  
Platform      15.812841  
Genre         4.928611  
Publisher     4.928611  
dtype: float64  
Rank          8292.00  
Year          2007.00  
NA_Sales      0.08  
EU_Sales      0.02  
JP_Sales      0.00  
Other_Sales   0.01  
Global_Sales  0.17  
Platform      16.00  
Genre         5.00  
Publisher     5.00  
dtype: float64
```



```
#Create Histograms to view our variables
df[['Year' ,
    'NA_Sales' ,
    'EU_Sales' ,
    'JP_Sales' ,
    'Other_Sales' ,
    'Global_Sales' ,
    'Platform' , 'Genre' , 'Publisher']].hist()
mpl.savefig('churn_hists.jpg')
mpl.tight_layout()
```



```
: #Create Boxplots for our continuous variables
```

```
sb.boxplot('Year' , data = df)
```

```
mpl.show()
```

```
sb.boxplot('NA_Sales' , data = df)
```

```
mpl.show()
```

```
sb.boxplot('EU_Sales' , data = df)
```

```
mpl.show()
```

```
sb.boxplot('JP_Sales' , data = df)
```

```
mpl.show()
```

```
sb.boxplot('Other_Sales' , data = df)
```

```
mpl.show()
```

```
sb.boxplot('Global_Sales' , data = df)
```

```
mpl.show()
```

```
sb.boxplot('Platform' , data = df)
```

```
mpl.show()
```

```
sb.boxplot('Genre' , data = df)
```

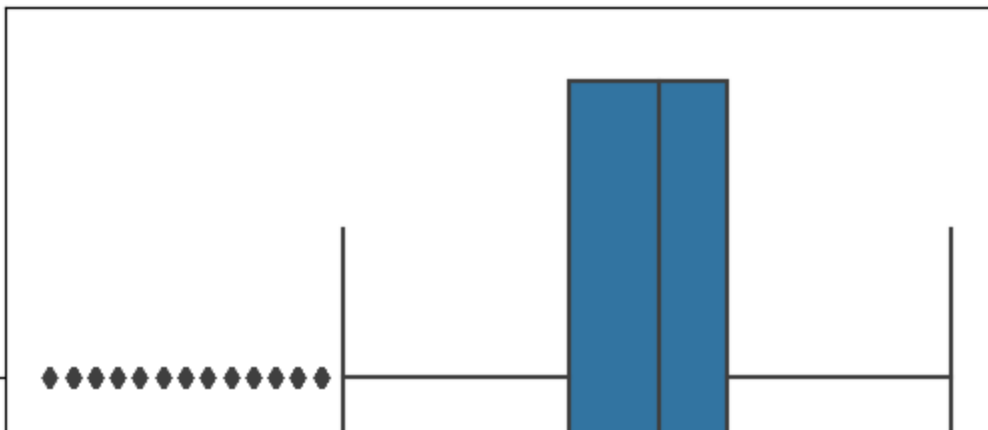
```
mpl.show()
```

```
sb.boxplot('Publisher' , data = df)
```

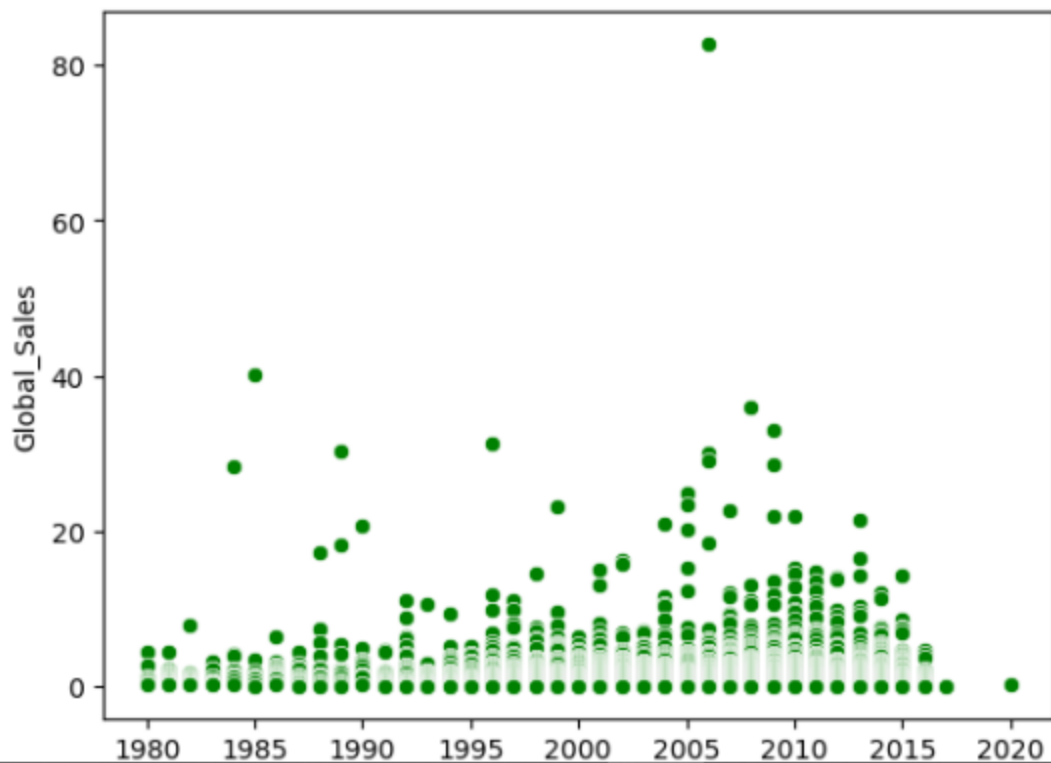
```
mpl.show()
```

C:\Users\se202271009\Anaconda3\lib\site-packages\seaborn_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

```
warnings.warn(
```




```
#Scatterplots to show direct or inverse relationships between the target &  
sb.scatterplot(x=df['Year'], y=df['Global_Sales'],  
               color='green')  
mpl.show()  
sb.scatterplot(x=df['NA_Sales'], y=df['Global_Sales'],  
               color='green')  
mpl.show()  
sb.scatterplot(x=df['JP_Sales'], y=df['Global_Sales'],  
               color='green')  
mpl.show()  
sb.scatterplot(x=df['EU_Sales'], y=df['Global_Sales'],  
               color='green')  
mpl.show()  
sb.scatterplot(x=df['Other_Sales'], y=df['Global_Sales'],  
               color='green')  
mpl.show()  
sb.scatterplot(x=df['Platform'], y=df['Global_Sales'],  
               color='green')  
mpl.show()  
sb.scatterplot(x=df['Genre'], y=df['Global_Sales'],  
               color='green')  
mpl.show()  
sb.scatterplot(x=df['Publisher'], y=df['Global_Sales'],  
               color='green')  
mpl.show()
```



The data is now fully prepared and extracted as “Sales_Cleaned.csv” The next step is to start working on our MLR model. The rest of the visualizations can be seen in the code file, html, or pdf provided.

Analysis

D. Report on your data-analysis process by describing the analysis technique(s) you used to appropriately analyze the data. Include the calculations you performed and their outputs. Justify how you selected the analysis technique(s) you used, including one advantage and one disadvantage of these technique(s).

The main idea of this project is to see if we can create a multiple linear regression model to predict future global sales and see what characteristics are most important in determining global sales.. In order to answer this question, we are going to create a MLR using global sales as our target variable and the other variables in the dataset as the independent variables. MLR can show us linear relationships in our dataset and be used to predict global sales but does not show causality (Yadav 2021). One disadvantage of MLR is that it can only describe linear relationships between variables, limiting what we can see. The actual modeling itself will be done by packages in our coding environment, specifically sklearn and its subpackages, which were chosen because of their ability to perform a MLR and my familiarity with them. One disadvantage of sklearn is that it can not visualize very well and we need to import and use more packages to visualize our model in graph form.

To analyze the model, we will look at p values with stepwise backwards reduction, variance inflation factor reduction (VIF), residual error, and r square. Stepwise backwards reduction works by removing all variables with a p value greater than .05, which is an output of our model and a disadvantage of it is that it quickly eliminates variables from one metric and allows for bias to be introduced into the model. VIF measures how much a variable is influenced by another variable i.e. the higher the number the greater the need to eliminate the variable. We used a cutoff of 3 for a strict elimination and therefore shows the disadvantage of possibly allowing bias and definitely allowing for an arbitrary reduction method (Yadav 2021).

Beyond the reduction metrics, I chose to use residual error to measure our model because it is the difference between the actual and predicted values. This will allow us to measure how well our prediction method works and should show small residual error and a tightly plotted line. One disadvantage of this is that any inconsistencies with residuals will show up with an inconsistent model. Finally, the R squared value shows strength of prediction and this can be looked at to show percentage of strength of prediction, to measure the accuracy of the model. One disadvantage of R squared is that it does not decrease as our model changes.

It is now time to show the code for each of these analyses and techniques in the order they were presented in this report. As a note, I ran several models to fully analyze this data and the results show a different story than our hypotheses suggest, which will be further explored in the reporting and summarizing section of this report.

```
# Create initial estimated regression equation that could be used to predict Global_Sales, given the continuous and categorical variab
LMR = ols("Global_Sales ~ Year + Publisher + Platform + NA_Sales + EU_Sales + JP_Sales + Other_Sales + Genre", data=df).fit()
print(LMR.params)
print(LMR.summary())
```

```
Intercept      6.994607e-03
Year           -3.251709e-06
Publisher       -6.028122e-07
Platform        -9.288370e-06
NA_Sales        9.999536e-01
EU_Sales        9.999913e-01
JP_Sales        9.998387e-01
Other_Sales     9.996273e-01
Genre          -6.028122e-07
dtype: float64
```

OLS Regression Results

Dep. Variable:	Global_Sales	R-squared:	1.000
Model:	OLS	Adj. R-squared:	1.000
Method:	Least Squares	F-statistic:	2.095e+08
Date:	Thu, 09 Mar 2023	Prob (F-statistic):	0.00
Time:	10:17:09	Log-Likelihood:	62490.
No. Observations:	16291	AIC:	-1.250e+05
Df Residuals:	16283	BIC:	-1.249e+05
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
Intercept	0.0070	0.015	0.473	0.636	-0.022	0.036
Year	-3.252e-06	7.36e-06	-0.442	0.659	-1.77e-05	1.12e-05
Publisher	-6.028e-07	5.49e-06	-0.110	0.913	-1.14e-05	1.02e-05
Platform	-9.288e-06	4.99e-06	-1.860	0.063	-1.91e-05	5.02e-07
NA_Sales	1.0000	8.16e-05	1.22e+04	0.000	1.000	1.000
EU_Sales	1.0000	0.000	6829.269	0.000	1.000	1.000
JP_Sales	0.9998	0.000	6580.169	0.000	1.000	1.000
Other_Sales	0.9996	0.000	3125.488	0.000	0.999	1.000
Genre	-6.028e-07	5.49e-06	-0.110	0.913	-1.14e-05	1.02e-05

Omnibus: 213.542 Durbin-Watson: 1.612
Prob(Omnibus): 0.000 Jarque-Bera (JB): 384.363
Skew: 0.048 Prob(JB): 3.44e-84
Kurtosis: 3.746 Cond. No. 7.39e+18

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.2e-27. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.

```
#Try again without the other sales information
```

```
LMR = ols("Global_Sales ~ Year + Publisher + Platform + Genre", data=df).fit()
print(LMR.params)
print(LMR.summary())
```

```
Intercept    43.608796
Year         -0.021537
Publisher      0.002306
Platform      0.007691
Genre         0.002306
dtype: float64
```

OLS Regression Results

```
=====
Dep. Variable:    Global_Sales    R-squared:                0.007
Model:            OLS            Adj. R-squared:            0.007
Method:            Least Squares   F-statistic:                40.34
Date:            Thu, 09 Mar 2023   Prob (F-statistic):        5.80e-26
Time:            10:17:11          Log-Likelihood:            -30376.
No. Observations: 16291          AIC:                        6.076e+04
Df Residuals:     16287          BIC:                        6.079e+04
Df Model:          3
Covariance Type:  nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	43.6088	4.305	10.129	0.000	35.170	52.048
Year	-0.0215	0.002	-10.032	0.000	-0.026	-0.017
Publisher	0.0023	0.002	1.405	0.160	-0.001	0.006
Platform	0.0077	0.001	5.180	0.000	0.005	0.011
Genre	0.0023	0.002	1.405	0.160	-0.001	0.006

```
=====
Omnibus:            33764.736    Durbin-Watson:           0.061
Prob(Omnibus):      0.000        Jarque-Bera (JB):        245097764.986
Skew:               17.360        Prob(JB):                0.00
Kurtosis:           602.895       Cond. No.                7.30e+18
=====
```

Notes:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
- [2] The smallest eigenvalue is 1.23e-27. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.


```
#Stepwise Backwards reduction with a P value cutoff of  $p > .05$  from the regression models and p values in the output  
#After removing all variables with a P value  $> .05$ , we are left with the following variables: Year, Platform, All of
```

```
#Now we will use Variance Inflation Factor (VIF) to reduce our model. For the output, any VIF above 3 should be removed  
from patsy import dmatrices  
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
#find design matrix for linear regression model using 'rating' as response variable  
y, X = dmatrices('Global_Sales ~ Year + Platform + NA_Sales + EU_Sales + JP_Sales + Other_Sales', data = df, return_
```

```
#calculate VIF for each explanatory variable  
VIF = pd.DataFrame()  
VIF['VIF'] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]  
VIF['variable'] = X.columns
```

```
#view VIF for each explanatory variable  
VIF
```

	VIF	variable
0	127983.692443	Intercept
1	1.082921	Year
2	1.040894	Platform
3	2.690268	NA_Sales
4	3.319718	EU_Sales
5	1.340652	JP_Sales
6	2.206489	Other_Sales

```
# Everything above 3 VIF was removed to have a very strict cutoff
```

```
# Create reduced OLS multiple regression using the ols feature. We are compiling our final regression using the variable
LM_Reduced = ols("Global_Sales ~ Year + Platform + NA_Sales + JP_Sales + Other_Sales", data=df).fit()
print(LM_Reduced.params)
print(LM_Reduced.summary())

# Extract Clean dataset
df.to_csv('Capstone_Data.csv')

Residuals = df['Global_Sales'] - LM_Reduced.predict(df[['Global_Sales', 'Year', 'Platform', 'NA_Sales', 'JP_Sales', 'Other_Sales']])
sb.scatterplot(x=df['Global_Sales'], y=Residuals, color='green')
mpl.show()
```

```
Intercept    -9.174632
Year          0.004572
Platform      0.000382
NA_Sales      1.292622
JP_Sales      1.194827
Other_Sales   2.042369
dtype: float64
```

OLS Regression Results

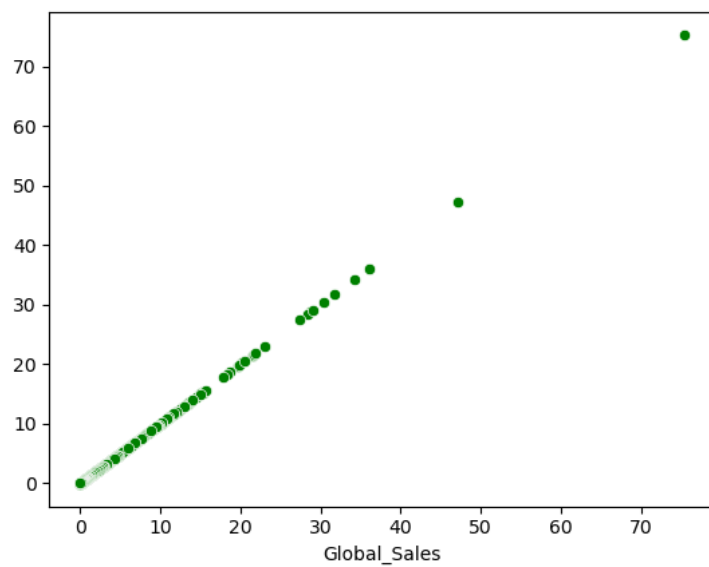
```
=====
Dep. Variable:    Global_Sales    R-squared:                0.968
Model:            OLS            Adj. R-squared:             0.968
Method:           Least Squares   F-statistic:              9.911e+04
Date:             Thu, 09 Mar 2023 Prob (F-statistic):        0.00
Time:             10:17:21        Log-Likelihood:           -2352.8
No. Observations: 16291          AIC:                        4718.
Df Residuals:     16285          BIC:                        4764.
Df Model:         5
Covariance Type:  nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	-9.1746	0.780	-11.756	0.000	-10.704	-7.645
Year	0.0046	0.000	11.746	0.000	0.004	0.005
Platform	0.0004	0.000	1.429	0.153	-0.000	0.001
NA_Sales	1.2926	0.004	347.606	0.000	1.285	1.300
JP_Sales	1.1948	0.008	149.567	0.000	1.179	1.210
Other_Sales	2.0424	0.015	135.762	0.000	2.013	2.072

```
=====
Omnibus:                 23368.617    Durbin-Watson:           2.059
Prob(Omnibus):            0.000      Jarque-Bera (JB):        204487635.835
Skew:                     -7.354      Prob(JB):                 0.00
Kurtosis:                 551.668      Cond. No.                 7.15e+05
=====
```

Notes:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
- [2] The condition number is large, 7.15e+05. This might indicate that there are strong multicollinearity or other numerical problems.



```
In [22]: # Here we are calculating our residual error from the reduced model
print(np.sqrt(LM_Reduced.mse_resid))
0.2796173489745299
```

```
# Create reduced OLS multiple regression using the ols feature without other sales information. We are compiling our
LM_Reduced = ols("Global_Sales ~ Year + Platform", data=df).fit()
print(LM_Reduced.params)
print(LM_Reduced.summary())

# Extract Clean dataset
df.to_csv('Capstone_Data.csv')

Residuals = df['Global_Sales'] - LM_Reduced.predict(df[['Global_Sales', 'Year', 'Platform']])
sb.scatterplot(x=df['Global_Sales'],y=Residuals,color='green')
mpl.show()
```

```
Intercept    44.427467
Year         -0.021935
Platform      0.007792
dtype: float64
```

OLS Regression Results

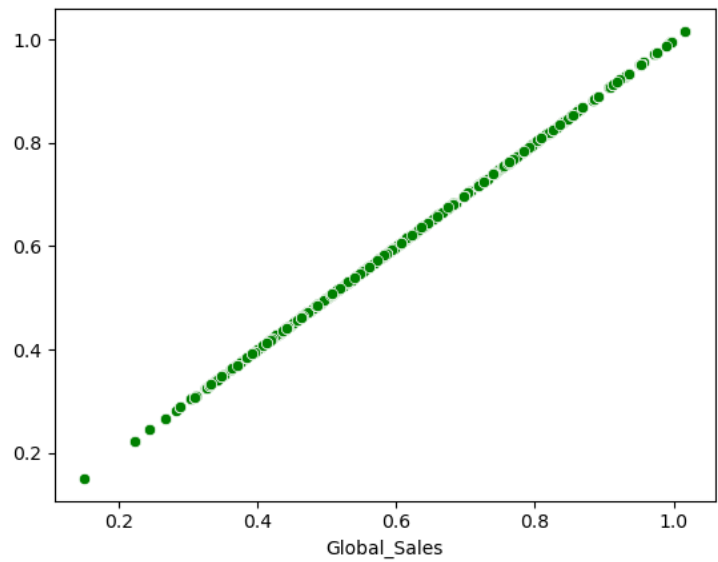
```
=====
Dep. Variable:    Global_Sales    R-squared:                0.007
Model:            OLS            Adj. R-squared:             0.007
Method:           Least Squares   F-statistic:              61.49
Date:             Thu, 09 Mar 2023 Prob (F-statistic):       2.49e-27
Time:             10:17:27        Log-Likelihood:          -30112.
No. Observations: 16291          AIC:                       6.023e+04
Df Residuals:     16288          BIC:                       6.025e+04
Df Model:         2
Covariance Type:  nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	44.4275	4.197	10.585	0.000	36.201	52.654
Year	-0.0219	0.002	-10.476	0.000	-0.026	-0.018
Platform	0.0078	0.001	5.341	0.000	0.005	0.011

```
=====
Omnibus:                 33127.105    Durbin-Watson:           0.102
Prob(Omnibus):            0.000        Jarque-Bera (JB):        182607991.288
Skew:                     16.713        Prob(JB):                 0.00
Kurtosis:                 520.592        Cond. No.                 7.00e+05
=====
```

Notes:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
- [2] The condition number is large, 7e+05. This might indicate that there are strong multicollinearity or other numerical problems.



```
In [24]: # Here we are calculating our residual error from the reduced model
print(np.sqrt(LM_Reduced.mse_resid))
```

1.536513586801373

```
In [25]: df
```

Out[25]:

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Platform	Genre	Publisher
0	1	2006.0	41.49	29.02	3.77	8.46	0.629189	26	10	10
1	2	1985.0	29.08	3.58	6.81	0.77	0.972930	11	4	4
2	3	2008.0	15.85	12.88	3.79	3.31	0.585319	26	6	6
3	4	2009.0	15.75	11.01	3.28	2.96	0.563385	26	10	10
4	5	1996.0	11.27	8.89	10.22	1.00	0.684895	5	7	7
...
16593	16596	2002.0	0.01	0.00	0.00	0.00	0.561079	6	4	4
16594	16597	2003.0	0.01	0.00	0.00	0.00	0.546937	7	8	8
16595	16598	2008.0	0.00	0.00	0.00	0.00	0.507395	16	6	6
16596	16599	2010.0	0.00	0.01	0.00	0.00	0.370017	4	5	5
16597	16600	2003.0	0.01	0.00	0.00	0.00	0.539144	6	4	4

Data Summary and Implications

E. Summarize the implications of your data analysis by discussing the results of your data analysis in the context of the research question, including one limitation of your analysis. Within the context of your research question, recommend a course of action based on your results.

Then propose two directions or approaches for future study of the data set.

The implications of this analysis are that we have a model that can be used to predict future global sales. However, in our model exists the possibility of strong multicollinearity and it does not include many of the original variables in the dataset. I built several models to show how we can group the data of sales broken down by region, all of the variables without sales, and then our final model. The final model without sales had a 0.7% prediction, indicating this model has no strength of prediction and is not a viable means to predict future global sales. The implications and results are that the business can not use these variables and the regression equation to predict future sales. However, we also have a better model.

There is a model that includes the sales of other regions to show which regions the business should focus their efforts and includes year, platform, NA, JP, and other sales to predict future global sales. This model has a residual error of 0.28, indicating that there is a small amount of error and then a strength of prediction of 96.8% . Overall with this final model we reject the null hypothesis and conclude we can create a model to predict video game sales. The interpretation of the coefficients and regression equation from this successful model are listed

below. The strong multicollinearity in our data and the limitations of the variables in the final models represent issues within this analysis and should be explored in a future investigation.

First model after reduction that is statistically insignificant:

```
# Create reduced OLS multiple regression using the ols feature without other sales information. We are compiling our
LM_Reduced = ols("Global_Sales ~ Year + Platform", data=df).fit()
print(LM_Reduced.params)
print(LM_Reduced.summary())

# Extract Clean dataset
df.to_csv('Capstone_Data.csv')

Residuals = df['Global_Sales'] - LM_Reduced.predict(df[['Global_Sales', 'Year', 'Platform']])
sb.scatterplot(x=df['Global_Sales'], y=Residuals, color='green')
mpl.show()
```

```
Intercept    44.427467
Year         -0.021935
Platform      0.007792
dtype: float64
```

OLS Regression Results

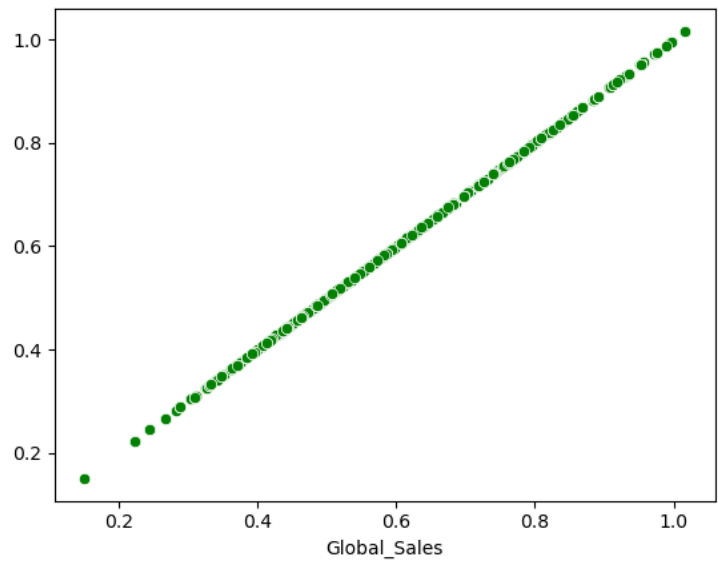
```
=====
Dep. Variable:    Global_Sales    R-squared:                0.007
Model:            OLS            Adj. R-squared:             0.007
Method:            Least Squares   F-statistic:                61.49
Date:              Thu, 09 Mar 2023   Prob (F-statistic):        2.49e-27
Time:              10:17:27          Log-Likelihood:            -30112.
No. Observations: 16291            AIC:                        6.023e+04
Df Residuals:      16288            BIC:                        6.025e+04
Df Model:           2
Covariance Type:   nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	44.4275	4.197	10.585	0.000	36.201	52.654
Year	-0.0219	0.002	-10.476	0.000	-0.026	-0.018
Platform	0.0078	0.001	5.341	0.000	0.005	0.011

```
=====
Omnibus:            33127.105    Durbin-Watson:           0.102
Prob(Omnibus):      0.000        Jarque-Bera (JB):        182607991.288
Skew:               16.713        Prob(JB):                0.00
Kurtosis:           520.592        Cond. No.                7.00e+05
=====
```

Notes:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
- [2] The condition number is large, 7e+05. This might indicate that there are strong multicollinearity or other numerical problems.



```
In [24]: # Here we are calculating our residual error from the reduced model
print(np.sqrt(LM_Reduced.mse_resid))

1.536513586801373
```

```
In [25]: df
```

Out[25]:

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Platform	Genre	Publisher
0	1	2006.0	41.49	29.02	3.77	8.46	0.629189	26	10	10
1	2	1985.0	29.08	3.58	6.81	0.77	0.972930	11	4	4
2	3	2008.0	15.85	12.88	3.79	3.31	0.585319	26	6	6
3	4	2009.0	15.75	11.01	3.28	2.96	0.563385	26	10	10
4	5	1996.0	11.27	8.89	10.22	1.00	0.684895	5	7	7
...
16593	16596	2002.0	0.01	0.00	0.00	0.00	0.561079	6	4	4
16594	16597	2003.0	0.01	0.00	0.00	0.00	0.546937	7	8	8
16595	16598	2008.0	0.00	0.00	0.00	0.00	0.507395	16	6	6
16596	16599	2010.0	0.00	0.01	0.00	0.00	0.370017	4	5	5
16597	16600	2003.0	0.01	0.00	0.00	0.00	0.539144	6	4	4

Final model that is statistically significant and the resulting equation:


```
# Create reduced OLS multiple regression using the ols feature without other sales information. We are compiling our
LM_Reduced = ols("Global_Sales ~ Year + Platform", data=df).fit()
print(LM_Reduced.params)
print(LM_Reduced.summary())

# Extract Clean dataset
df.to_csv('Capstone_Data.csv')

Residuals = df['Global_Sales'] - LM_Reduced.predict(df[['Global_Sales', 'Year', 'Platform']])
sb.scatterplot(x=df['Global_Sales'],y=Residuals,color='green')
mpl.show()
```

```
Intercept    44.427467
Year         -0.021935
Platform      0.007792
dtype: float64
```

OLS Regression Results

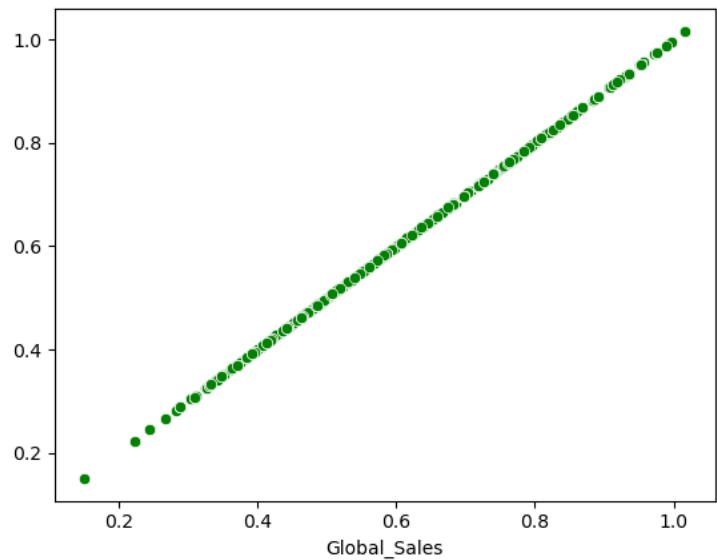
```
=====
Dep. Variable:    Global_Sales    R-squared:                0.007
Model:            OLS            Adj. R-squared:            0.007
Method:           Least Squares   F-statistic:              61.49
Date:             Thu, 09 Mar 2023 Prob (F-statistic):       2.49e-27
Time:             10:17:27        Log-Likelihood:          -30112.
No. Observations: 16291          AIC:                       6.023e+04
Df Residuals:     16288          BIC:                       6.025e+04
Df Model:         2
Covariance Type:  nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	44.4275	4.197	10.585	0.000	36.201	52.654
Year	-0.0219	0.002	-10.476	0.000	-0.026	-0.018
Platform	0.0078	0.001	5.341	0.000	0.005	0.011

```
=====
Omnibus:                 33127.105    Durbin-Watson:           0.102
Prob(Omnibus):            0.000      Jarque-Bera (JB):        182607991.288
Skew:                     16.713      Prob(JB):                 0.00
Kurtosis:                 520.592      Cond. No.                 7.00e+05
=====
```

Notes:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
- [2] The condition number is large, 7e+05. This might indicate that there are strong multicollinearity or other numerical problems.



```
In [24]: # Here we are calculating our residual error from the reduced model
print(np.sqrt(LM_Reduced.mse_resid))

1.536513586801373
```

```
In [25]: df
```

Out[25]:

	Rank	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Platform	Genre	Publisher
0	1	2006.0	41.49	29.02	3.77	8.46	0.629189	26	10	10
1	2	1985.0	29.08	3.58	6.81	0.77	0.972930	11	4	4
2	3	2008.0	15.85	12.88	3.79	3.31	0.585319	26	6	6
3	4	2009.0	15.75	11.01	3.28	2.96	0.563385	26	10	10
4	5	1996.0	11.27	8.89	10.22	1.00	0.684895	5	7	7
...
16593	16596	2002.0	0.01	0.00	0.00	0.00	0.561079	6	4	4
16594	16597	2003.0	0.01	0.00	0.00	0.00	0.546937	7	8	8
16595	16598	2008.0	0.00	0.00	0.00	0.00	0.507395	16	6	6
16596	16599	2010.0	0.00	0.01	0.00	0.00	0.370017	4	5	5
16597	16600	2003.0	0.01	0.00	0.00	0.00	0.539144	6	4	4

$$y = -9.174632 + (\text{Year} * 0.004572) + (\text{Platform} * 0.000382) + (\text{NA_Sales} * 1.292622) +$$

$$(\text{JP_Sales} * 1.194827) + (\text{Other_Sales} * 2.042369)$$

Residual error: .28

R-squared: .968, 96.8% strength of prediction

The coefficients suggest that per each unit of the following, Global Sales will increase or decrease by the following per one unit of the variable:

Year, Global Sales will increase by 0.0046

Platform, Global Sales will increase by .0004

NA_Sales, Global Sales will increase by 1.12926

JP_Sales, Global Sales will increase by 1.1948

Other_Sales, Global Sales will increase by 2.0424

The condition number is large, indicating we have multicollinearity or other numerical problems with this model. However, we can reject the null hypothesis and this is a valid, statistically significant model to predict future global sales. The Year variable is included here in case the hypothetical business is looking for information regarding a previous game's performance and can be investigated further, described below.

Hypothesis: Null hypothesis- We can not create a model to predict video game sales.

Alternate Hypothesis- We can create a model to predict video game sales.

The main limitation of this analysis is that it can only identify linear relationships and can't identify causality among those linear relationships. This is a major limitation for all MLR models and does not take away from the analysis itself, but does help to explain the shortcomings of the results. A way to mitigate this limitation would be to perform more investigations into this dataset.

A course of action I recommend to help answer the question of building a model to predict future global sales is directly related to the major limitation of this analysis. Now that we

have looked for linear relationships, I recommend the hypothetical business to open up the analysis to non-linear relationships and to go back and greatly increase the size of the dataset and the characteristics listed within the dataset. Doing both of these will build on this investigation.

One direction for future study to build a model to predict future global sales would be to gather a larger dataset with more games and characteristics of games to run another model with the same questions. Further information gathering could be where the game was released, which time of year, if other major titles released during the same time, and the overall inflation/economy at the time since the sales span many years. I believe gathering a much larger dataset and introducing more possible characteristics would present this hypothetical business problem with a clearer picture.

A second direction for future study to build a model to predict future global sales would be to dive into the relationships found here, despite the strong multicollinearity and non-conclusive results. This would mean diving into the relationship between year and sales and see if there's an inflation factor or if there's a pattern we can investigate for the year and sales information. Seeing the overall performance of platforms would also be beneficial in this path and opening up our analysis to non-linear relationships will provide more actionable insight.

F. Acknowledge sources, using in-text citations and references, for content that is quoted.

Sources

Video Game Sales. (n.d.). www.kaggle.com.

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

Prasanna, M. (2021, October 21). *How to Efficiently Handle Large Datasets for Machine*

Learning and Data Analysis Using Python. Medium.

<https://python.plainenglish.io/working-with-large-datasets-for-machine-learning-d8da0dd802fb>

Yadav, H. (2021, May 8). *Multiple Linear Regression Implementation in Python*. Machine

Learning with Python.

<https://medium.com/machine-learning-with-python/multiple-linear-regression-implementation-in-python-2de9b303fc0c>

G. Demonstrate professional communication in the content and presentation of your submission.

This aspect cannot be summarized, however, I hope it has shown through in all aspects of this report. Thank you for reading this.