

## Notebook 1 - Exploring the Video Games Datasets

In this notebook I will investigate and try to find answers to some specific questions a video games developer/publisher company might ask.

The questions are as follows:

- How many games were sold on new consoles around their launch date?
- What type of game has the potential to make a developer/publisher the most money?
- Do games that sell more appeal to the player?
- What do players have to say about specific video game titles?
- What type of game will cost a developer/publisher the most money to create?

I feel that my findings in relation to these questions and the information I can draw from these findings will be of high value to a video games developer/publisher company.

I start off the notebook by importing some useful libraries.

```
In [1]: import pandas as pd
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
import pylab as py
import numpy as np
import json
from textblob import TextBlob
from collections import Counter
from sklearn.feature_extraction.text import CountVectorizer
from sklearn import feature_extraction
%matplotlib inline
from subprocess import check_output
```

I check out the file directory I am going to be working with to see which important files/folders I need to use for data analysis further on.

```
In [2]: print(check_output(["ls", "/home/vagrant/Desktop/VideoGames"]).decode("utf8"))
20-years-of-games.zip
ign.csv
over-13000-steam-games.zip
platforms.csv
platforms.csv~
steam-data-master.zip
steam_reviews-master
video-game-sales-with-ratings.zip
videogamesales.zip
videogames.csv
Video Games Sales as at 22 Dec 2016.csv
Video Games Sales as at 22 Dec 2016.json~
```

I read in the CSVs and assign them to new data frames for easy manipulation further on.

```
In [3]: vgd = pd.read_csv('/home/vagrant/Desktop/VideoGames/Video Games Sales as at 22 Dec 2016.csv',)
ignndf = pd.read_csv('/home/vagrant/Desktop/VideoGames/ign.csv')
vgcdf = pd.read_csv('/home/vagrant/Desktop/VideoGames/videogames.csv')
vgpdf = pd.read_csv('/home/vagrant/Desktop/VideoGames/platforms.csv')
```

I have added a list of where I sourced the CSVs from. Two of the CSVs were sourced externally. I created one using Microsoft Excel and the second was sourced from GitHub.

**Video\_Games\_Sales\_as\_at\_22\_Dec\_2016.csv:**

[https://github.com/otacke/udacity-machine-learning-engineer/blob/master/submissions/capstone\\_project/data/Video\\_Games\\_Sales\\_as\\_at\\_22\\_Dec\\_2016.csv](https://github.com/otacke/udacity-machine-learning-engineer/blob/master/submissions/capstone_project/data/Video_Games_Sales_as_at_22_Dec_2016.csv)

**ign.csv:**

<https://github.com/graknlabs/sample-projects/blob/master/example-csv-migration-games/ign.csv>

**video games.csv:**

<https://github.com/ali-ce/datasets/blob/master/Most-Expensive-Things/Videogames.csv>

**platforms.csv:**

Created by me using Microsoft Excel.

**steam\_reviews master:**

[https://github.com/mulhod/steam\\_reviews](https://github.com/mulhod/steam_reviews)

I check out the first five rows of the video game sales data frame. I have a look at the first five rows of the data frame to get a better understanding of the data. By looking at the column names and the data in the rows I can get an idea of what the data represents.

In [4]: `vgdf.head(5)`

Out [4]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24	NaN	NaN	
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0	
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0	
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN	

The column **"Year\_Of\_Release"** represents the year the game was released in. The column **"Genre"** represents the video game genre. The column **"Publisher"** represents the company which published the video game. The column **"NA\_Sales"** represents North American sales. The column **"EU\_Sales"** represents European sales. The column **"JP\_Sales"** represents Japanese sales. The column **"Other\_Sales"** represents sales in other regions. The column **"Global\_Sales"** represents sales globally. The column **"Critic\_Score"** represents a video game critic score. The column

"Critic\_Count" represents the number of video game critics. The column "User\_Score" represents a score given by a person who played the video game. The column "User\_Count" represents the number of users who gave a user score. The column "Developer" represents the company who developed the video game. The column "Rating" represents the age rating given to the video game.

I check out the video game sales data frame's column names so that I have the names to use in data analysis further on.

```
In [5]: vgdf.columns
Out[5]: Index([u'Name', u'Platform', u'Year of Release', u'Genre', u'Publisher',
              u'NA_Sales', u'EU_Sales', u'JP_Sales', u'Other_Sales', u'Global_Sales',
              u'Critic_Score', u'Critic_Count', u'User_Score', u'User_Count',
              u'Developer', u'Rating'],
              dtype='object')
```

I run correlation on the video game sales data frame.

```
In [6]: pd.set_option('display.width', 100)
pd.set_option('precision', 3)
correlations = vgdf.corr(method='pearson')
print(correlations)
```

	Year_of_Release	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	\
Year_of_Release	1.000	-0.093	0.004	-0.168	0.038	-0.076	
NA_Sales	-0.093	1.000	0.765	0.450	0.639	0.941	
EU_Sales	0.004	0.765	1.000	0.435	0.723	0.901	
JP_Sales	-0.168	0.450	0.435	1.000	0.291	0.612	
Other_Sales	0.038	0.639	0.723	0.291	1.000	0.749	
Global_Sales	-0.076	0.941	0.901	0.612	0.749	1.000	
Critic_Score	0.011	0.241	0.221	0.153	0.199	0.245	
Critic_Count	0.223	0.295	0.278	0.180	0.252	0.304	
User_Count	0.175	0.246	0.283	0.076	0.239	0.265	

	Critic_Score	Critic_Count	User_Count
Year_of_Release	0.011	0.223	0.175
NA_Sales	0.241	0.295	0.246
EU_Sales	0.221	0.278	0.283
JP_Sales	0.153	0.180	0.076
Other_Sales	0.199	0.252	0.239
Global_Sales	0.245	0.304	0.265
Critic_Score	1.000	0.426	0.264
Critic_Count	0.426	1.000	0.362
User_Count	0.264	0.362	1.000

The first important correlation that I notice is happening between the global sales and the rest of the sales from all the other regions. Other region's sales have their highest value correlating with the global sales value. I can assume that this is because a large part of the other region's sales makes up part of the total global sales.

The final important correlation that I notice happens between critic count and user count. I notice that the user count has a lower value when correlated to the critic count this may show that a lower number of users reviewed games compared to a higher number of critics. From this I could say that because of the lower user count the information around user scores may be somewhat bias towards a smaller group of individuals.

I check out the video game platforms data frame's column names. so that I have the names to use in data analysis further on.

```
In [7]: vgpdf.columns
Out[7]: Index([u'Platform', u'Year_Of_Release'], dtype='object')
```

I check out the first five rows of the video game platform data frame.

```
In [8]: vgpdf.head(5)
Out[8]:
```

	Platform	Year_Of_Release
0	Xbox 360	2005.0
1	Xbox One	2013.0
2	PS3	2007.0
3	PS4	2013.0
4	Wii U	2012.0

I have a look at the first five rows of the data frame to get a better understanding of the data. By looking at the column names and the data in the rows I can get an idea of what the data represents.

The column **"Platform"** represents the platform or console. The column **"Year\_Of\_Release"** represents the year a console was released.

I query the video game sales data frame for games released and sold around each console's launch date and get the length of the data frame returned.

```
In [9]: X360 = vgdf.query('Year_of_Release == 2005 & Platform == "X360"')
lengthX360 = len(X360.index)
```

```
In [10]: XOne = vgdf.query('Year_of_Release == 2013 & Platform == "XOne"')
lengthXOne = len(XOne.index)
```

```
In [11]: PS3 = vgdf.query('Year_of_Release == 2007 & Platform == "PS3"')
lengthPS3 = len(PS3.index)
```

```
In [12]: PS4 = vgdf.query('Year_of_Release == 2013 & Platform == "PS4"')
lengthPS4 = len(PS4.index)
```

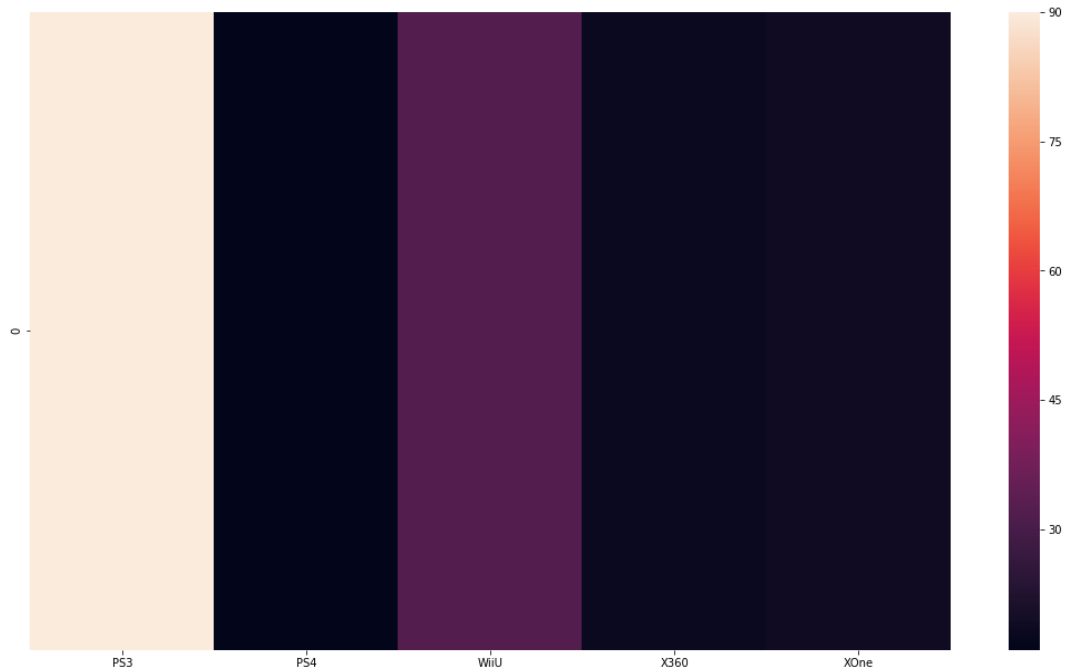
```
In [13]: WiiU = vgdf.query('Year_of_Release == 2012 & Platform == "WiiU"')
lengthWiiU = len(WiiU.index)
```

I append the length variables to a new data frame.

```
In [14]: dflengths = pd.DataFrame()
df = dflengths.append({'X360': lengthX360, 'XOne': lengthXOne, 'PS3': lengthPS3, 'PS4': lengthPS4, 'WiiU': lengthWiiU}, ig
```

I create a heatmap from the lengths data frame.

```
In [15]: fig, ax = plt.subplots(figsize=(18.5, 10.5))
ax = sns.heatmap(df)
```



From the heatmap I can tell that the PS3 had the most games sold around launch compared to other platforms. The PS4 and Xbox 360 had the lowest amount of games sold around launch. I can assume that the PS3 did better at launch compared with other consoles due to Sony making better decisions when purchasing titles around launch date and having a better marketing plan in place.

I create a data frame video game sales data frame using its sales columns and describe it to find the max sales figures for each region.

```
In [16]: vgdfi = vgdf[['NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales', 'Global_Sales']]
vgdfi.describe()
```

Out[16]:

	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
count	16719.000	16719.000	16719.000	16719.000	16719.000
mean	0.263	0.145	0.078	0.047	0.534
std	0.814	0.503	0.309	0.187	1.548
min	0.000	0.000	0.000	0.000	0.010
25%	0.000	0.000	0.000	0.000	0.060
50%	0.080	0.020	0.000	0.010	0.170
75%	0.240	0.110	0.040	0.030	0.470
max	41.360	28.960	10.220	10.570	82.530

I query the video game sales data frame to find data about the highest selling game(s) Globally and in each region. I also perform cleaning, so I get no results which return a “NaN” field.

```
In [17]: na = vgdf.query('NA_Sales == 41.36 & NA_Sales != "NaN"')
In [18]: eu = vgdf.query('EU_Sales == 28.96 & EU_Sales != "NaN"')
In [19]: jp = vgdf.query('JP_Sales == 10.22 & JP_Sales != "NaN"')
In [20]: os = vgdf.query('Other_Sales == 10.57 & Other_Sales != "NaN"')
In [21]: gs = vgdf.query('Global_Sales == 82.53 & Global_Sales != "NaN"')
```

I concatenate all the highest selling games data frames into one data frame and display it.

```
In [22]: highsales = pd.concat([na, eu, jp, os, gs], ignore_index=True)
         highsales
```

Out[22]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	
1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	
2	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN	
3	Grand Theft Auto: San Andreas	PS2	2004.0	Action	Take-Two Interactive	9.43	0.40	0.41	10.57	20.81	95.0	80.0	
4	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	

I notice that Wii Sports sold more globally, in Europe and in north America. Due to it selling globally and in larger regions such as Europe and north America I can assume that games like Wii Sports have the potential to make a developer/publisher the most money.

Looking at the sales in Japan I could assume that the difference in cultures could have affected the type of top selling game. I know from research that in Japan Anime has become a part of their culture with the series Pokémon being among one of the top Anime shows in the country. Therefore, due to Pokémon’s popularity within Japan and outside of Japan, having Pokémon as the highest selling game is not a surprise here.

As for other region's sales I failed to find any data on why Grand Theft Auto: San Andreas sold so much it could be just down to the fact the game was well made and ticked all the right boxes for gamers.

I query the video game sales data frame for user scores and describe the returned data frame as type float to get the max user score. I found difficulty describing the returned data frame normally so to fix this I described as type float. I also perform cleaning so when I query I don't get any fields returned with "tbd" or "NaN".

```
In [23]: userscores = vgdf.query('User_Score != "tbd" & User_Score != "NaN"')
userscoreinfo = userscores[['User_Score']]
print(userscoreinfo.astype(float).describe())
```

	User_Score
count	7590.000
mean	7.125
std	1.500
min	0.000
25%	6.400
50%	7.500
75%	8.200
max	9.700

I query the video games sales data frame for max user score video games. I also perform cleaning so that I get no fields returned with "NaN".

```
In [24]: highuserscores = vgdf.query('User_Score == "9.7" & User_Score != "tbd" & User_Score != "NaN"')
```

I query the video games sales data frame for critic scores and describe the returned data frame as type float to get the max critic score. I found difficulty describing the returned data frame normally so to fix this I described as type float. I also perform cleaning to get no fields returned with "NaN".

```
In [25]: criticscores = vgdf.query('Critic_Score != "tbd" & Critic_Score != "NaN"')
criticscoreinfo = criticscores[['Critic_Score']]
print(criticscoreinfo.astype(float).describe())
```

	Critic_Score
count	8137.000
mean	68.968
std	13.938
min	13.000
25%	60.000
50%	71.000
75%	79.000
max	98.000

I query the video games data frame for max critic score video games. I also perform cleaning to make sure no fields return with the value of "NaN".

```
In [26]: highcriticscores = vgdf.query('Critic_Score == "98.0" & Critic_Score != "tbd" & Critic_Score != "NaN"')
```

I check out the first five rows of the IGN data frame.

```
In [27]: igndf.head(5)
```

```
Out[27]:
```

	Unnamed: 0	score_phrase	title	url	platform	score	genre	editors_choice	release_year	release_month	release_day
0	0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	Y	2012	9	12
1	1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	Y	2012	9	12
2	2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	N	2012	9	12
3	3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	N	2012	9	11
4	4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	N	2012	9	11

I have a look at the first five rows of the data frame to get a better understanding of the data. By looking at the column names and the data in the rows I can get an idea of what the data represents.

The column "Unnamed:" represents the row number. The column "score\_phrase" represents the review phrase given to the video game. The column "title" represents the name of the video game. The column "url" represents the ign.com url for the video game. The column "platform" represents the console or platform of the video game. The column "score" represents the IGN score given to the video game. The column "genre" represents the genre of the video game. The column "editors\_choice" represents whether the game was editor's choice or not. The column "release\_year" represents the release year of the video game. The column "release\_month" represents the release month of the game. The column "release\_day" represents the release day of the game.

I check out the IGN data frame's column names so that I have the names to use in data analysis further on.

```
In [28]: igndf.columns
```

```
Out[28]: Index([u'Unnamed: 0', u'score_phrase', u'title', u'url', u'platform', u'score', u'genre',  
u'editors_choice', u'release_year', u'release_month', u'release_day'],  
dtype='object')
```



I query the IGN data frame for IGN scores and describe the returned data frame as type float to get max IGN score. I found difficulty describing the returned data frame normally so to fix this I described as type float. I also perform cleaning, so the fields returned do not contain "tbd" or "NaN".

```
In [29]: ignscores = igndf.query('score != "tbd" & score != "NaN"')
ignscoreinfo = ignscores[['score']]
print(ignscoreinfo.astype(float).describe())
```

```

score
count 18625.000
mean   6.950
std    1.712
min     0.500
25%    6.000
50%    7.300
75%    8.200
max    10.000
```

I check out the first five rows of the IGN high scores data frame.

```
In [30]: highignscores = igndf.query('score == 10')
highignscores.head(5)
```

Out[30]:

	Unnamed: 0	score_phrase	title	url	platform	score	genre	editors_choice	release_year	release_month	release_day
1058	1058	Masterpiece	The Legend of Zelda: Ocarina of Time	/games/the-legend-of-zelda-ocarina-of-time/n64...	Nintendo 64	10.0	Action, Adventure	Y	1998	11	25
1287	1287	Masterpiece	Pokemon Blue Version	/games/pokemon-blue-version/gb-16708	Game Boy	10.0	RPG	Y	1999	6	23
1289	1289	Masterpiece	Pokemon Red Version	/games/pokemon-red-version/gb-9846	Game Boy	10.0	RPG	Y	1999	6	23
1354	1354	Masterpiece	Joust	/games/defenderjoust/lynx-4146	Lynx	10.0	Action	N	1999	7	6
1363	1363	Masterpiece	Shanghai	/games/shanghai-809520/lynx-5876	Lynx	10.0	Puzzle	N	1999	7	6

I notice there are a lot of games getting returned with a score of 10.

I concatenate all high user score(s), critic score(s) & highest selling game's user score(s). I prepare the high IGN score(s) for further concatenation by only taking the first row from the IGN high scores data frame because there were so many results we only need one for visualization purposes. I also rename some column names so that the data will group together. I found difficulty using the group by and aggregate methods so opted to instead rename the column names to group the data. I am at this point starting to get a final data frame ready for visualization.

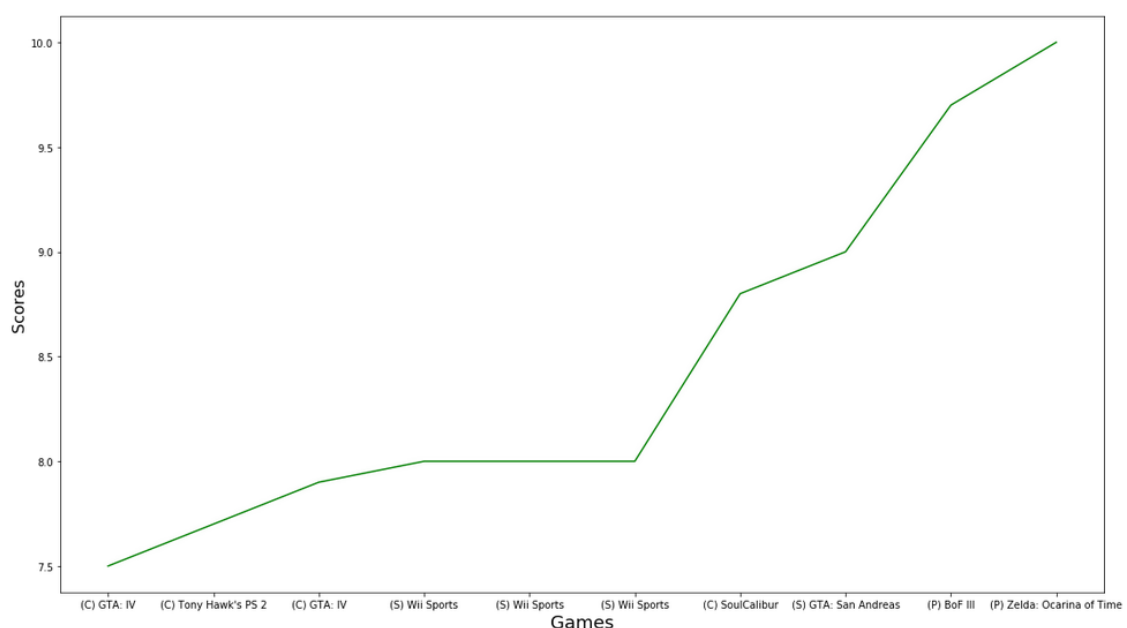
```
In [31]: ignhs = highignscores[["title", "score"]].head(1)
nahs = na[["Name", "User_Score"]]
euhs = eu[["Name", "User_Score"]]
jphs = jp[["Name", "User_Score"]]
oshs = os[["Name", "User_Score"]]
gshs = gs[["Name", "User_Score"]]
us = highuserscores[["Name", "User_Score"]]
cs = highcriticscores[["Name", "User_Score"]]
finaluserscores = pd.concat([nahs, euhs, jphs, oshs, gshs, us, cs])
finaluserscores.rename(columns={"User_Score": "score", "Name": "title"}, inplace=True)
```

I concatenate high IGN score(s) with the data frame from before and perform cleaning by removing high score Japanese video game(s) as they contain a user score of "NaN" which is irrelevant in this context. The data is now ready for visualization.

```
In [32]: finalscores = pd.concat([ignhs, finaluserscores])
finalscoresclean = finalscores.query('title != "Pokemon Red/Pokemon Blue"')
```

I manually enter the names for the graph because I found difficulty getting the ordering correct when the final scores data frame was added in ascending order so to fix this I entered the names in manually. I add the final scores data frame in ascending order to the graph. I then create the graph.

(C) Critic, (P) Player & (S) Highest Sales: Game Scores



From the graph we can see the highest critic scores, highest user/IGN scores and the highest selling games user scores. We can make the conclusion from this information that even though Wii Sports sold the most it did not appeal to the player as much as other titles did.

I read in the steam review comments for each game using the json\_lines API. I then put the steam review comments for each game into a bag of words. I found difficulty with this at first but once I could read the steam review comments in using the json\_lines API I was able to append them to a list which would then work with bag of words.

I then run sentiment analysis on the steam review comments list for each game.

By using machine learning I was able to get a count of word occurrences within the steam review comments for each game. I could then find the most common words

from the bag of words that holds the steam review comments for each game by adding the words and word counts to a dictionary. I then call the most common function on this dictionary to return the ten words which occurred most in the bag of words.

### Text analysis for Arma 3:

```
In [34]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Arma_3.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
armalist = [word for line in var for word in line.split()]

vectorizer1 = CountVectorizer()
bagofwords1 = vectorizer1.fit(armalist)
bagofwords1 = vectorizer1.transform(armalist)
```

```
In [35]: armastr = str(armalist)
tb = TextBlob(armastr)
print tb.sentiment

Sentiment(polarity=0.08586322774515183, subjectivity=0.5006874017439243)
```

```
In [36]: armadict = vectorizer1.vocabulary_
dict(Counter(armadict).most_common(10))
```

```
Out[36]: {u'zoom': 16198,
u'zombies': 16199,
u'zombiesies': 16200,
u'zoomed': 16201,
u'zooming': 16202,
u'zrffhw': 16203,
u'zs': 16204,
u'zu': 16205,
u'zubar': 16206,
u'zues': 16207}
```

After researching each of the most common words the word "zues" stood out the most and made more sense out of all the other words. Zeus is an Arma 3 DLC which players seemed to talk about a lot in the steam review comments this could show that they were commenting on how much they enjoyed or disliked this DLC.

## Text analysis for Counter Strike:

```
In [37]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Counter_Strike.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
cslist = [word for line in var for word in line.split()]

vectorizer2 = CountVectorizer()

bagofwords2 = vectorizer2.fit(cslist)

bagofwords2 = vectorizer2.transform(cslist)
```

```
In [38]: csstr = str(cslist)
tb = TextBlob(csstr)
print tb.sentiment

Sentiment(polarity=0.1894113625295528, subjectivity=0.5076360027168362)
```

```
In [39]: csdict = vectorizer2.vocabulary_
dict(Counter(csdict).most_common(10))
```

```
Out[39]: {'u'zneel': 6812,
u'zombie': 6813,
u'zombiemod': 6814,
u'zombies': 6815,
u'zone': 6816,
u'zoomed': 6817,
u'zooming': 6818,
u'zrenjanin': 6819,
u'zvqy7u3ax27dsmxq': 6820,
u'zzzzzz': 6821}
```

After researching each of the most common words the word "zrenjanin" stood out the most and made more sense out of all the other words. Zrenjanin is a city in the province of Vojvodina, Serbia. As this word came up most this could show that most players playing Counter Strike are from Zrenjanin.

## Text analysis for Counter Strike: Global Offensive:

```
In [40]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Counter_Strike_Global_Offensive.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
csgolist = [word for line in var for word in line.split()]

vectorizer3 = CountVectorizer()

bagofwords3 = vectorizer3.fit(csgolist)

bagofwords3 = vectorizer3.transform(csgolist)
```

```
In [41]: csgostr = str(csgolist)
tb = TextBlob(csgostr)
print tb.sentiment

Sentiment(polarity=0.08076105145853191, subjectivity=0.4960738958783354)
```

```
In [42]: csgodict = vectorizer3.vocabulary_
dict(Counter(csgodict).most_common(10))
```

```
Out[42]: {u'zombiemod': 14814,
u'zombies': 14815,
u'zombye': 14816,
u'zone': 14817,
u'zones': 14818,
u'zoom': 14819,
u'zoomed': 14820,
u'zooming': 14821,
u'zozey': 14822,
u'zzz': 14823}
```

After researching each of the most common words the word "zombiemod" stood out the most and made more sense out of all the other words. Zombiemod is a mod for counter strike global offensive. The comments about zombiemod may show that players were commenting how much they liked or disliked zombiemod.

## Text analysis for Dota 2:

```
In [43]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Dota_2.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
dotalist = [word for line in var for word in line.split()]

vectorizer4 = CountVectorizer()

bagofwords4 = vectorizer4.fit(dotalist)

bagofwords4 = vectorizer4.transform(dotalist)
```

```
In [44]: dotastr = str(dotalist)
tb = TextBlob(dotastr)
print tb.sentiment

Sentiment(polarity=0.0840700293721373, subjectivity=0.5069623201378455)
```

```
In [45]: dotadict = vectorizer4.vocabulary_
dict(Counter(dotadict).most_common(10))
```

```
Out[45]: {u'zoo': 15806,
u'zskavania': 15807,
u'zues': 15808,
u'zvuky': 15809,
u'zvykol': 15810,
u'zygamantis': 15811,
u'zynga': 15812,
u'zyzz': 15813,
u'zzz': 15814,
u'zzzz': 15815}
```

After researching each of the most common words the word "zues" stood out the most and made more sense out of all the other words. Zeus is a hero the player can play in dota 2. The comments about zeus could show that players liked playing that hero in dota 2 or that they dislike that hero.

## Text analysis for Football Manager 2015:

```
In [46]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Football_Manager_2015.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
fblist = [word for line in var for word in line.split()]

vectorizer5 = CountVectorizer()

bagofwords5 = vectorizer5.fit(fblist)

bagofwords5 = vectorizer5.transform(fblist)
```

```
In [47]: fbstr = str(fblist)
tb = TextBlob(fbstr)
print tb.sentiment

Sentiment(polarity=0.0496541006407484, subjectivity=0.4922505289180941)
```

```
In [48]: fbdict = vectorizer5.vocabulary_
dict(Counter(fbdict).most_common(10))
```

```
Out[48]: {u'youve': 7345,
u'youwant': 7346,
u'yrs': 7347,
u'system': 7348,
u'yt': 7349,
u'yup': 7350,
u'zenith': 7351,
u'zero': 7352,
u'zone': 7353,
u'zx': 7354}
```

After researching each of the most common words the word "zx" stood out the most and made more sense out of all the other words. In the comments the players are talking about ZX which relates to the ZX spectrum console which the 1988 Football Manager 2 came out on. These comments may be talking about how the original version of the game was better or worse than the new edition of Football Manager.

## Text analysis for Garry's Mod:

```
In [49]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Garrys_Mod.jsonlines', 'rb') as f:
    for item in json lines.reader(f):
        var.append(item['review'])
gmlist = [word for line in var for word in line.split()]

vectorizer6 = CountVectorizer()

bagofwords6 = vectorizer6.fit(gmlist)

bagofwords6 = vectorizer6.transform(gmlist)
```

```
In [50]: gmstr = str(gmlist)
tb = TextBlob(gmstr)
print tb.sentiment

Sentiment(polarity=0.10539860461356092, subjectivity=0.49983190177965936)
```

```
In [51]: gmdict = vectorizer6.vocabulary_
dict(Counter(gmdict).most_common(10))
```

```
Out[51]: {u'zoidbergs': 11839,
u'zombie': 11840,
u'zombierp': 11841,
u'zombies': 11842,
u'zomg': 11843,
u'zone': 11844,
u'zoom': 11845,
u'zs': 11846,
u'zuera': 11847,
u'zworld': 11848}
```

After researching each of the most common words the word "Zuera" stood out the most and made more sense out of all the other words. Zuera is a municipality located in the province of Zaragoza, Aragon, Spain. The comments may show that players of this game were mainly located in this municipality.



## Text analysis for Grand Theft Auto V:

```
In [52]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Grand_Theft_Auto_V.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
gtalist = [word for line in var for word in line.split()]

vectorizer7 = CountVectorizer()

bagofwords7 = vectorizer7.fit(gtalist)

bagofwords7 = vectorizer7.transform(gtalist)
```

```
In [53]: gtastr = str(gtalist)
tb = TextBlob(gtastr)
print tb.sentiment

Sentiment(polarity=0.07026684754741393, subjectivity=0.5061428760901148)
```

```
In [54]: gtadict = vectorizer7.vocabulary_
dict(Counter(gtadict).most_common(10))
```

```
Out[54]: {u'zonah': 21019,
u'zone': 21020,
u'zoned': 21021,
u'zoom': 21022,
u'zoomed': 21023,
u'zooms': 21024,
u'zrobien': 21025,
u'zyba': 21026,
u'zzle': 21027,
u'zzzzz': 21028}
```

After researching each of the most common words the word "zonah" stood out the most and made more sense out of all the other words. In the game GTA V, Zonah is a medical centre where a player goes when they die. The players in the comments could have been talking about this medical centre.

## Text analysis for Sid Meiers Civilization 5:

```
In [55]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Sid Meiers Civilization 5.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
smlist = [word for line in var for word in line.split()]

vectorizer8 = CountVectorizer()

bagofwords8 = vectorizer8.fit(smlist)

bagofwords8 = vectorizer8.transform(smlist)
```

```
In [56]: smstr = str(smlist)
tb = TextBlob(smstr)
print tb.sentiment

Sentiment(polarity=0.11377876245451478, subjectivity=0.5035939936344109)
```

```
In [57]: smdict = vectorizer8.vocabulary_
dict(Counter(smdict).most_common(10))
```

```
Out[57]: {u'ziti': 13180,
u'zombies': 13181,
u'zone': 13182,
u'zones': 13183,
u'zoo': 13184,
u'zooming': 13185,
u'zoroastrianism': 13186,
u'zulu': 13187,
u'zulunation': 13188,
u'zulus': 13189}
```

After researching each of the most common words the word "zoroastrianism" stood out the most and made more sense out of all the other words. Zoroastrianism is a religion in the game which a player can unlock. The players in the comments might have been talking about unlocking this religion in the game.

## Text analysis for Team Fortress 2:

```
In [58]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Team_Fortress_2.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
tflist = [word for line in var for word in line.split()]

vectorizer9 = CountVectorizer()
bagofwords9 = vectorizer9.fit(tflist)
bagofwords9 = vectorizer9.transform(tflist)
```

```
In [59]: tfstr = str(tflist)
tb = TextBlob(tfstr)
print tb.sentiment

Sentiment(polarity=0.1153535844071964, subjectivity=0.5114566384049448)
```

```
In [60]: tfdict = vectorizer9.vocabulary_
dict(Counter(tfdict).most_common(10))
```

```
Out[60]: {u'zombie': 13105,
u'zombies': 13106,
u'zombiewave': 13107,
u'zone': 13108,
u'zoomed': 13109,
u'zooseexualist': 13110,
u'zskavania': 13111,
u'zx': 13112,
u'zxc': 13113,
u'zyaxitron': 13114}
```

After researching each of the most common words the word "zombie" stood out the most and made more sense out of all the other words. Zombie fortress is a mod for team fortress 2 so the players in the comments may have been discussing how much they liked this mod or how much they disliked it.

## Text analysis for The Elder Scrolls V:

```
In [61]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/The_Elder_Scrolls_V.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
skylist = [word for line in var for word in line.split()]

vectorizer11 = CountVectorizer()

bagofwords11 = vectorizer11.fit(skylist)

bagofwords11 = vectorizer11.transform(skylist)
```

```
In [62]: skystr = str(skylist)
tb = TextBlob(skystr)
print tb.sentiment

Sentiment(polarity=0.08634567983483905, subjectivity=0.5204243922387338)
```

```
In [63]: skydict = vectorizer11.vocabulary_
dict(Counter(skydict).most_common(10))
```

```
Out[63]: {u'zmkov': 15524,
u'zo': 15525,
u'zoidberg': 15526,
u'zombie': 15527,
u'zombies': 15528,
u'zone': 15529,
u'zones': 15530,
u'zor': 15531,
u'zun': 15532,
u'zzzzzzzz': 15533}
```

After researching each of the most common words the word "zombies" stood out the most and made more sense out of all the other words. In the Elder Scrolls V there are many undead enemies so the players in the comments could have been discussing these enemies.

## Text analysis for Warframe:

```
In [64]: var = []
with open('/home/vagrant/Desktop/VideoGames/steam_reviews-master/data/Warframe.jsonlines', 'rb') as f:
    for item in json_lines.reader(f):
        var.append(item['review'])
wflist = [word for line in var for word in line.split()]

vectorizer22 = CountVectorizer()
bagofwords22 = vectorizer22.fit(wflist)
bagofwords22 = vectorizer22.transform(wflist)
```

```
In [65]: wfstr = str(wflist)
tb = TextBlob(wfstr)
print tb.sentiment

Sentiment(polarity=0.0948656742014625, subjectivity=0.5071836010347777)
```

```
In [66]: wfdict = vectorizer22.vocabulary_
dict(Counter(wfdict).most_common(10))
```

```
Out[66]: {u'zombie': 15563,
u'zombies': 15564,
u'zone': 15565,
u'zones': 15566,
u'zoom': 15567,
u'zooming': 15568,
u'zorencoptering': 15569,
u'zorens': 15570,
u'zyqw7sb': 15571,
u'zyrohunter': 15572}
```

After researching each of the most common words the word "zorencoptering" stood out the most and made more sense out of all the other words. In Warframe zorencoptering is an ability the player can use with their weapons to get around the map quickly. Players in the comments may have been discussing this ability.

## Sentiment analysis for all games:

Out of all the polarities Team Fortress 2 received the highest positive polarity so I can assume that players found this game most appealing out of all the games.

Football Manager 2015 received the lowest polarity of all the games, so I can assume that players found this game less appealing out of all the other games.

I check out the first five rows of the video games costs data frame.

```
In [67]: vgcdf.head(5)
```

```
Out[67]:
```

	Videogame	Year	Developer	Platform(s)	Publisher	Estimated Development Costs	Estimated Marketing Costs.text	Estimated total costs.text	Wikipedia Profile	Image	D
0	APB: All Points Bulletin	2010	Realtime Worlds	PC	Realtime Worlds	\$50,000,000	NaN	\$50,000,000	<a href="http://en.wikipedia.org/wiki/APB:_All_Points_B...">http://en.wikipedia.org /wiki/APB:_All_Points_B...</a>	<a href="https://upload.wikimedia.org/wikipedia/en/f/f1...">https://upload.wikimedia.org /wikipedia/en/f/f1...</a>	APB de
1	Call of Duty: Elite	2011	Beachhead Studios	PC	Activision	\$50,000,000	NaN	\$50,000,000	<a href="http://en.wikipedia.org/wiki/Call_of_Duty:_Elite">http://en.wikipedia.org /wiki/Call_of_Duty:_Elite</a>	<a href="https://upload.wikimedia.org/wikipedia/en/b/b1...">https://upload.wikimedia.org /wikipedia/en/b/b1...</a>	Call of wa: ser
2	Crysis 3	2013	Crytek Frankfurt	PS3, Xbox 360, PC	Electronic Arts	\$66,000,000	NaN	\$66,000,000	<a href="http://en.wikipedia.org/wiki/Crysis_3">http://en.wikipedia.org /wiki/Crysis_3</a>	<a href="https://upload.wikimedia.org/wikipedia/en/2/20...">https://upload.wikimedia.org /wikipedia/en/2/20...</a>	Crysis sh
3	DC Universe Online	2012	Sony Online Entertainment	PC, PS3, PS4	NaN	\$50,000,000	NaN	\$50,000,000	<a href="http://en.wikipedia.org/wiki/DC_Universe_Online">http://en.wikipedia.org /wiki/DC_Universe_Online</a>	<a href="https://upload.wikimedia.org/wikipedia/en/c/c9...">https://upload.wikimedia.org /wikipedia/en/c/c9...</a>	DC ml
4	Deadpool	2013	High Moon Studios	PS3, Xbox 360, PC	Activision	NaN	NaN	\$100,000,000	<a href="http://en.wikipedia.org/wiki/Deadpool_(video_g...">http://en.wikipedia.org /wiki/Deadpool_(video_g...</a>	<a href="https://upload.wikimedia.org/wikipedia/en/4/41...">https://upload.wikimedia.org /wikipedia/en/4/41...</a>	Dea actionc 'em up

I have a look at the first five rows of the data frame to get a better understanding of the data. By looking at the column names and the data in the rows I can get an idea of what the data represents.

The column "**Videogame**" represents the video game's name. The column "**Year**" represents the year the video game was created. The column "**Developer**" represents the developer who developed the video game. The column "**Platform(s)**" represents the console or platform of the video game. The column "**Publisher**" represents the publisher of the video game. The column "**Estimated Development Costs**" represents the development costs when creating the videogame. The column "**Estimated Marketing Costs.text**" represents the estimated marketing costs of the video game. The column "**Estimated total costs.text**" represents the estimated total costs of creating the video game and marketing it. The column "**Wikipedia Profile**" represents a link to the Wikipedia page of the video game. The column "**Image**" represents a link to an image of the video game's cover art. The column "**Description**" represents a short description of the video game.

I checkout the video game costs data frame column names so that I have the names to use in data analysis further on.

```
In [68]: vgcdf.columns
Out[68]: Index([u'Videogame', u'Year', u'Developer', u'Platform(s)', u'Publisher',
               u'Estimated Development Costs', u'Estimated Marketing Costs.text',
               u'Estimated total costs.text', u'Wikipedia Profile', u'Image', u'Description'],
              dtype='object')
```

I add the year and estimated total costs from the video games costs data frames to their own data frames. I then concatenate these data frames.

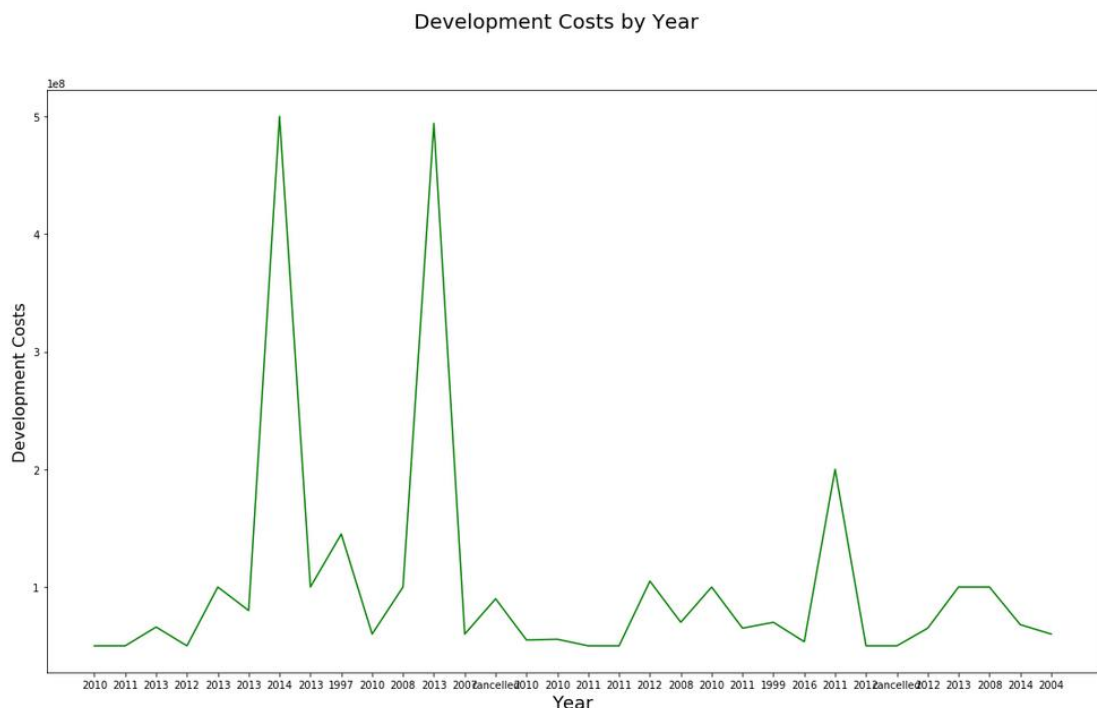
```
In [69]: vgcdfy = vgcdf['Year']
vgcdfc = vgcdf['Estimated total costs.text'].str.replace(',', ' ').str.replace('$', '')
vgcdfg = pd.concat([vgcdfy, vgcdfc], axis=1)
```

I replace full stops and spaces with null so later functions such as queries will work on the columns.

```
In [70]: vgcdfg.columns = vgcdfg.columns.str.replace('.', '').str.replace(' ', '')
```

Add the year from concatenated data frame to the graph and add the estimated total costs from concatenated data frame to the graph. I then create the graph.

```
In [71]: names = vgcdfg['Year']
counts = vgcdfg['Estimatedtotalcoststext']
py.figure(1)
fig = matplotlib.pyplot.gcf()
fig.set_size_inches(18.5, 10.5)
fig.suptitle('Development Costs by Year', fontsize=20)
plt.xlabel('Year', fontsize=18)
plt.ylabel('Development Costs', fontsize=16)
x = range(32)
py.xticks(x, names)
py.plot(x, counts, "g")
py.show()
```



From the graph I can see the years where it costed the most to create and market video games. In this case the year 2014 cost the most to create and market a video game.

I describe the estimated total costs from the concatenated data frame to find the highest development/marketing cost.

```
In [72]: vgcdfgt = vgcdfg['Estimatedtotalcoststext']
         vgcdfgt.describe()
```

I double check the highest cost returned from describe function was correct.

```
In [73]: a = vgcdfgt.tolist()
         for x in a:
             if(x >= 500000000):
                 break;
         print x
50000000
```

I replace all full stops and spaces with null for all column names in the video games costs data frame so later functions such as queries will work.

```
In [74]: vgcdfnew = vgcdf
         vgcdfnew.columns = vgcdfnew.columns.str.replace('.', '').str.replace(' ', '')
```

I query to find the highest costing game and display the result.

```
In [75]: highdevcost = vgcdfnew.query('Estimatedtotalcoststext == "$500,000,000"')
         highdevcost
```

```
Out[75]:
```

	Videogame	Year	Developer	Platform(s)	Publisher	EstimatedDevelopmentCosts	EstimatedMarketingCoststext	Estimatedtotalcoststext	Wikipedia
6	Destiny	2014	Bungie	PS3, PS4, Xbox 360, Xbox One	Activision	NaN	NaN	\$500,000,000	<a href="http://en.wikipedia/wiki/Destiny_(video_">http://en.wikiper /wiki/Destiny_(video_</a>

From the table I can see that the game "**Destiny**" costed the most to develop and market. I can assume that games like destiny cost the most to create.



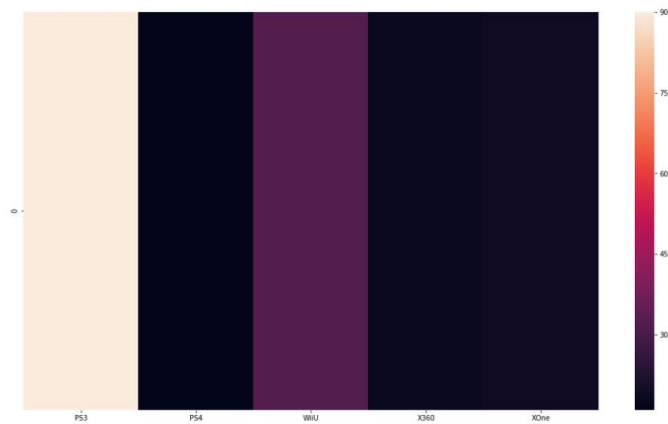
## Notebook 2 - Taking Action

### Notebook 2 - Taking Action

Who is my target audience?

*A video games developer/publisher company who has an interest in my findings.*

How many games were sold on new consoles around their launch date?



From the heatmap I can tell that the PS3 had the most games sold around launch compared to other platforms. The PS4 and Xbox 360 had the lowest amount of games sold around launch. I can assume that the PS3 did better at launch compared with other consoles due to Sony making better decisions when purchasing titles around launch date and having a better marketing plan in place.

If a new console was to be released by a video games developer/publisher company I would recommend based on my findings they purchase many games around launch date to encourage successful profits and to build a following amongst players.

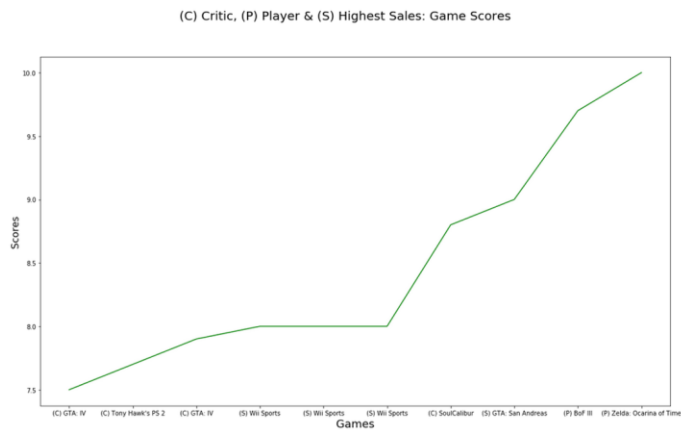
What type of game has the potential to make a developer/publisher the most money?

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	
1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	
2	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN	
3	Grand Theft Auto: San Andreas	PS2	2004.0	Action	Take-Two Interactive	9.43	0.40	0.41	10.57	20.81	95.0	80.0	
4	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	

*I notice that Wii Sports sold more globally, in Europe and in north America. Due to it selling globally and in larger regions such as Europe and north America I can assume that games like Wii Sports have the potential to make a developer/publisher the most money.*

From this information I would suggest if a develop/publisher company wants to make the most money when developing their games they create a Sports game like Wii Sports and target to a particular console or platform using the hardware that console or platform has to offer to it's fullest extent like Wii Sports did.

Do games that sell more appeal to the player?

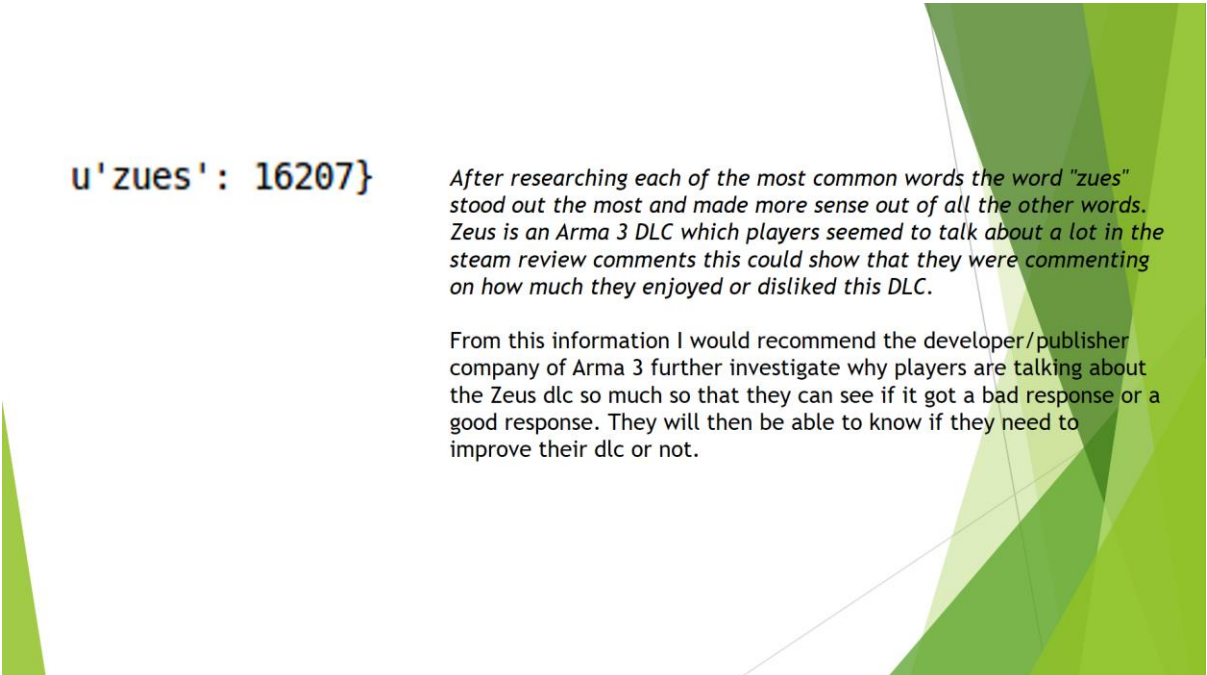


*From the graph we can see the highest critic scores, highest user/IGN scores and the highest selling games user scores. We can make the conclusion from this information that even though Wii Sports sold the most it did not appeal to the player as much as other titles did.*

From this information I would recommend a develop/publisher take into account why they are creating a game and what purpose this game will serve. Is it for the hardcore player or for a broader audience?



What do players have to say about specific video game titles? - Arma 3



u'zues': 16207}

*After researching each of the most common words the word "zues" stood out the most and made more sense out of all the other words. Zeus is an Arma 3 DLC which players seemed to talk about a lot in the steam review comments this could show that they were commenting on how much they enjoyed or disliked this DLC.*

From this information I would recommend the developer/publisher company of Arma 3 further investigate why players are talking about the Zeus dlc so much so that they can see if it got a bad response or a good response. They will then be able to know if they need to improve their dlc or not.



What do players have to say about specific video game titles? - **Counter Strike**

u'zrenjanin': 6819,

*After researching each of the most common words the word "zrenjanin" stood out the most and made more sense out of all the other words. Zrenjanin is a city in the province of Vojvodina, Serbia. As this word came up most this could show that most players playing Counter Strike are from Zrenjanin.*

From this information the developer/publisher company of Counter Strike could increase their profits by advertising the game Counter Strike within Zrenjanin more and running Counter Strike events in the city of Zrenjanin.

What do players have to say about specific video game titles? - **Garry's Mod**

u'zuera': 11847,

*After researching each of the most common words the word "Zuera" stood out the most and made more sense out of all the other words. Zuera is a municipality located in the province of Zaragoza, Aragon, Spain. The comments may show that players of this game were mainly located in this municipality.*

From this information the developer/publisher company of Garry's Mod could increase their profits by advertising the game Garry's Mod within Zuera more and running Garry's Mod events in the city of Zuera.

What type of game will cost a developer/publisher the most money to create?

	Videogame	Year	Developer	Platform(s)	Publisher	EstimatedDevelopmentCosts	EstimatedMarketingCoststext	Estimatedtotalcoststext	Wikipedia
6	Destiny	2014	Bungie	PS3, PS4, Xbox 360, Xbox One	Activision	NaN	NaN	\$500,000,000	<a href="http://en.wikipedia/wiki/Destiny_(video_game)">http://en.wikipedia/wiki/Destiny_(video_game)</a>

*From the table I can see that the game "Destiny" costed the most to develop and market. I can assume that games like destiny cost the most to create.*

I would recommend that a video games develop/publisher company looking to create a game does not make a game like Destiny unless they are willing to spend a lot of money developing and marketing the game.







