Katie Morton May 25th, 2022
Python for Data Analytics
Final Project Summary
Project Link: [x]

The video game industry has been a topic of conversation in my circle for years – mostly as fans and customers. I chose to try an analysis on freely available older sales data to get a visualization for how the market has changed historically as more advances in gaming technology are made. I wanted to show some basic key points with my analysis, including the upturn in popularity of gaming over the years as well as top genres, publishers and platforms. Some interesting information that helped solidify my decision to start with older sales data was finding out that the first computer game to be installed and played on multiple machines was actually developed in 1962 at a university.[1] The data I am working with starts in 1982, a full two decades later toward the period of time leading up to what was known in the industry as the console wars. I felt that showing the progression from that one game to the thriving industry it is today pays homage to its humble start.

I found this particular dataset[2] on Kaggle and noticed it contained truncated data, even at its full length. I chose it for being more complete and in the correct format than many others I was able to locate for this industry. This particular dataset didn't include all possible titles from all possible publishers since 1982, but it still gave a decent picture with the information that was provided. It seems to be gathered from some public reports, but not all companies provide this data in an easy-to-find manner, if at all. Of what can be found, very few are in the .csv file format. I chose to trim the dataset down to the first 200 rows that didn't contain incomplete information.

To begin the project, as suggested I tried to find another analysis and was able to locate one done in a different coding language[3] which came to similar results as what I found. My data, being truncated, did come up with different results overall, but even then the information I gathered from what was seen in that analysis was not unlike my own. The author did code more graphs to answer more specific questions. Without giving away any spoilers from the code itself, it still found the most popular genre of video games and the top selling publisher to be the same as what I was able to figure with just 200 rows of data. My research led me to an additional R project[4] that provided even more direction and helped me to figure out what questions I wanted to ask. I chose not to narrow my focus down to region-specific answers, and instead offered some simple insights from the dataset's global perspective.

My personal favorite part of this project was fixing errors by taking dedicated time to read the documentation and wisdom of other programmers. I learned so much about how truly

[1] https://www.history.com/topics/inventions/history-of-video-games#section_1
[2] https://www.kaggle.com/datasets/gregorut/videogamesales
[3] https://rpubs.com/maalghozi/lbb-game-sales
[4] https://www.kaggle.com/code/upadorprofzs/eda-video-game-sales

organized and efficient the pandas library is and how seaborn makes beautiful graphs without as much fuss. Research and asking the right questions helped me write my code. Once I had the basics of each graph down, I went back to my notes and each library's documentation to figure out how to change things to fit what I wanted each chart to look like. Part of my process was taking coding done during class and applying it to the dataset I was working with[5] before asking myself how it could be optimized and made more organized and visually consistent. I completely rewrote my project using seaborn after researching it[6] thoroughly. By understanding a few key concepts about the pandas library, I was able to reduce the number of lines I needed to write for the rest of the project, optimizing the process. It was an enjoyable process overall after that.

I chose to use bar graphs, a line graph and count plots because my goal was a simple overview of the dataset. I wanted to show popularity and productivity in comparison to sales data. Graph 1 would define the most popular genre of games released. Graph 2 handles global sales per year, showing the exponential growth of the industry over time. Graph 3 shows how many games were released each year to help confirm the global sales data in Graph 2. Graph 4 shows the amount of games released per platform and Graph 5 shows the number of games released per publisher.

I was able to conclude that the highest selling platform for video games in the selection of data I was working with was the Wii, released in 2006.[7] 2006 also happened to be one of the more productive years in gaming based on that data with 10 new releases for the platform. (The most productive years were 2009 – the year the Wii Motion Plus & corresponding Wii Sports Resort was released and 2010 – where Super Mario Galaxy 2, part of a major Nintendo franchise, was released to critical acclaim.[8]) More than just numbers are, as we can see, required to understand the reasons behind Nintendo's success, of course, but the numbers align regardless of if 200 lines of data are being analyzed or 11,000. Still, they show very clearly that a new game development or publishing company would do well to release their product (preferably a game in the action genre, per the data) on Nintendo consoles as the brand recognition and its historical productivity seems to lead to higher global sales overall.

I think my analysis was successful for what I had hoped to achieve. I think a more complete data set, maybe even one specific to each of the top brands, with newer data would be able to show an even clearer picture of the growth of the industry over time. I'd be interested to see how genre, publisher and platform preference shifts post-pandemic, especially with the rise of independent game developers. In the future, I'd love to compare critics' and/or customer reviews and ratings to the sales ranks to see how popular opinion can influence buyers as well.

---

[5] See Graphs 5b and 5c intentionally remaining in the project: https://colab.research.google.com/drive/1zDCEmCfSKhDdAZI4Yq6AFRjYsToPZpjF?usp=sharing
[6] See last page of this document for a list of links to documentation used.
[7] https://www.britannica.com/topic/Nintendo-Wii
[8] https://www.metacritic.com/browse/games/score/metascore/year/wii/all?year_selected=2010

Pandas Documentation Used:

1. .dropna: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html
2. .Groupby: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html
3. Groupby as a way to eliminate for loops for efficiency: https://stackoverflow.com/questions/20108140/how-to-use-groupby-to-avoid-loop-in-python
4. Reset_index: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.reset_index.html
5. .index: https://pandas.pydata.org/docs/reference/api/pandas.Index.html
6. .value_counts: https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
7. .count: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.count.html
8. .sort_values: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.sort_values.html
9. .iloc: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.iloc.html

Other threads and documentation referenced:

1. How to sort a dictionary by value: https://stackoverflow.com/questions/613183/how-do-i-sort-a-dictionary-by-value
2. Seaborn line plots: https://seaborn.pydata.org/generated/seaborn.lineplot.html
3. Seaborn count plots: https://seaborn.pydata.org/generated/seaborn.countplot.html
4. Seaborn bar charts: https://seaborn.pydata.org/generated/seaborn.barplot.html
5. Seaborn color palettes: https://seaborn.pydata.org/tutorial/aesthetics.html and https://seaborn.pydata.org/tutorial/color_palettes.html and https://www.codecademy.com/article/seaborn-design-ii
6. Matplotlib cheat sheets: https://matplotlib.org/cheatsheets/
7. Matplotlib bar_label: https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.bar_label.html
8. Pyplot adjusting xticks: https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.xticks.html
9. Pyplot adding titles: https://www.geeksforgeeks.org/matplotlib-pyplot-title-in-python/
10. Pyplot removing text before inline graph: https://www.youtube.com/watch?v=aXuN0X7hIW8 and https://stackoverflow.com/questions/12056115/disable-the-output-of-matplotlib-pyplot