Advanced Data Visualization Final Project: Investigating What Makes Video Games Popular on Steam

Jonathan Sebastiani

2025-03-14

About

I want to answer the question, "What makes the popular Steam video games popular?"

Using a number of interactive data visualizations, I will be answering this question to the best of my ability. I intend to make visualizations that show differences in the most popular Steam video games to reason out why they are so popular. Answering this question will help video gamers to make better decisions when choosing a game to play, and will help video game developers to target their games to the correct audience and know better where to allocate their efforts.

Motivating question: **Where should game developers be putting their efforts to promote the most growth in the video gaming community?**

Hypothesis: Sustaining popular videos games is extremely hard. Some newer games have spikes in popularity, but gamers fall back to nostalgic games. Video game developers should put more effort into sustaining older games and improving quality of play for those games through efficient software, consistent updates, and overall experience.

Link to the GitHub Data Challenge Repository (https://github.com/UWB-Adv-Data-Vis-2025-Wi-B/data-challenge-steam-game-development-option-2-js)

Limitations

- Many of the genres were in a different language or abbreviated. It would be hard to get all of the data for this.
- This data was collected in October 2024, and not as recent as possible.
- The data sets were not formatted together and were very messy. There is a lot of raw data that would have to be sifted through to get more accurate results, and that would take a lot of time and effort.

Data Background

While the Steam Database (https://steamdb.info/) is great for viewing current and top games, the raw data is not open source which makes it unusable for analysis.

The data that I will be using is from the open source github repository steam-dataset-2024 (https://github.com/win7guru/steam-dataset-2024). Although current data is preferable, the data is collected in October 2024 which is rather close. Although this data looks promising, the github author NewbieIndieGameDev does not site their source but explains that the repo contains “CSV files exported from a SQL database of video game data”.

The data files in the steam-dataset-2024 (https://github.com/win7guru/steam-dataset-2024) are extremely long. I had trouble with uploading the files to my github repository, so I looked at the files in excel to see which parameters would be applicable and helpful for my project. The files contained are as follows.

- games.csv: Main table containing details about the games, such as title, release date, and other metadata.
- genres.csv: Genres assigned to each game. tags.csv: Tags associated with each game, such as “Indie”, “Action”, etc.
- reviews.csv: Review data for the games, including Steam ratings and review counts. steamspy_insights.csv: Insights gathered from SteamSpy, such as estimated sales, playtime, and more.
- categories.csv: Information about the different Steam categories that games belong to (e.g., “Single-player”, “Full controller support”, etc.).
- ~~descriptions.csv~~: Full and summary text descriptions of each game.
- ~~promotional.csv~~: Links and metadata for promotional materials, such as trailers and screenshots.

After reviewing the files manually, I found that both the descriptions.csv and promotional.csv were extremely large and unhelpful for the project so I removed them from the analysis.

Data Cleaning

Lets start by putting the data into a manipulable form.

```
# Unzip and read the csv files into dataframes
categories <- read.csv(unzip("categories.zip"), row.names = NULL)
games <- read.csv(unzip("games.zip"), row.names = NULL)
genres <- read.csv(unzip("genres.zip"), row.names = NULL)
reviews <- read.csv(unzip("reviews.zip"), row.names = NULL)
steamspy_insights <- read.csv(unzip("steamspy_insights.zip"), row.names = NULL)
tags <- read.csv(unzip("tags.zip"), row.names = NULL)

# Set column names for games columns because they get messed up for some reason
colnames(games)[1:7] <- c("app_id", "name", "release_date", "is_free", "price_overview",
"price_overview2", "languages", "type")
```

Here is a look at the column names for these data sets.

```
## $categories
## $categories$Columns
## [1] "app_id"   "category"
##
##
## $games
## $games$Columns
## [1] "app_id"          "name"            "release_date"    "is_free"
## [5] "price_overview"  "price_overview2" "languages"       "type"
##
##
## $genres
## $genres$Columns
## [1] "app_id" "genre"
##
##
## $reviews
## $reviews$Columns
##  [1] "app_id"                   "review_score"
##  [3] "review_score_description" "positive"
##  [5] "negative"                 "total"
##  [7] "metacritic_score"         "reviews"
##  [9] "recommendations"          "steamspy_user_score"
## [11] "steamspy_score_rank"      "steamspy_positive"
## [13] "steamspy_negative"
##
##
## $steamspy_insights
## $steamspy_insights$Columns
##  [1] "app_id"                    "developer"
##  [3] "publisher"                 "owners_range"
##  [5] "concurrent_users_yesterday" "playtime_average_forever"
##  [7] "playtime_average_2weeks"    "playtime_median_forever"
##  [9] "playtime_median_2weeks"     "price"
## [11] "initial_price"             "discount"
## [13] "languages"                 "genres"
##
##
## $tags
## $tags$Columns
## [1] "app_id" "tag"
```

Some parameters that stood out which would help video game developers identify where they should be putting their time and efforts. By categorizing these we can sum up parameters into…

- GameDetails: app_id, name, genre, is_free, developer, price, release_date, owners_range
- Reviews: total, positive, negative, review_score, recommendations

Next, I'll create a single data frames to consolidate these parameters.

```
# Create GameDetails and GameDetails2 data frames
GameDetails <- games[, c("app_id", "name", "release_date", "is_free")]
GameDetails2 <- steamspy_insights[, c("app_id", "developer", "price", "owners_range")]
Reviews <- reviews[, c("app_id", "total", "positive", "negative", "review_score", "recom
mendations")]

# Merge GameDetails and GameDetails2 by app_id
gameData <- merge(GameDetails, GameDetails2, by = "app_id")

# Making the Genres as a list to join to the data
genres_mutated <- genres %>%
  group_by(app_id) %>%
  summarise(genres = list(unique(genre)), .groups = 'drop')  # Create a list of unique g
enres for each app_id

# Join genres based on app_id
gameData <- merge(gameData, genres_mutated, by = "app_id")

# Merge the Reviews data frame into gameData
gameData <- merge(gameData, Reviews, by = "app_id")

# Convert positive and total columns to numeric
gameData$positive <- as.numeric(gameData$positive)
gameData$total <- as.numeric(gameData$total)

# Calculate the review score
gameData$review_score <- gameData$positive / gameData$total
```

Lets take a look at this final dataset that we will be using.

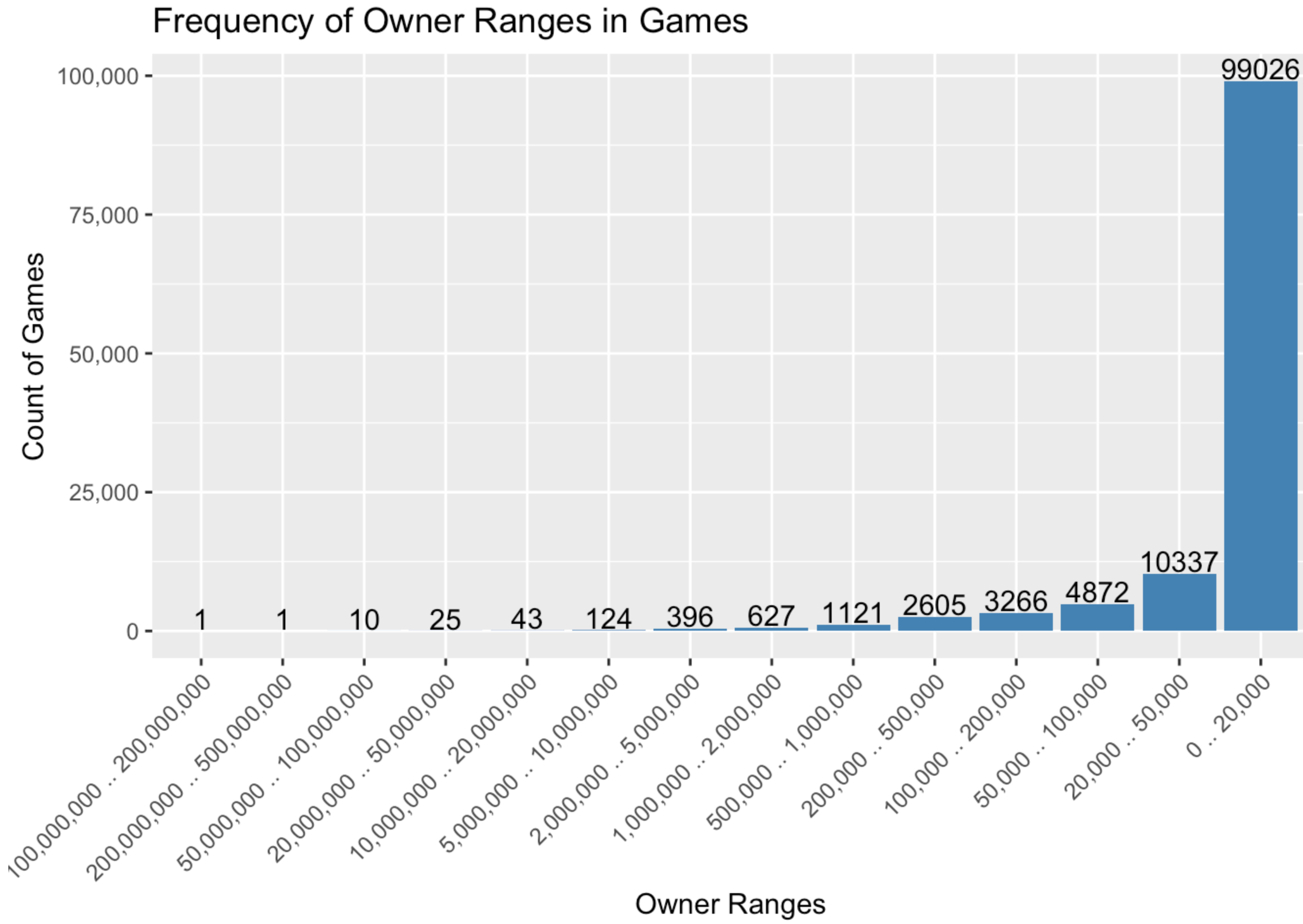
```
##    app_id                        name release_date is_free
## 1      10              Counter-Strike   2000-11-01       0
## 2      10              Counter-Strike   2000-11-01       0
## 3 1000000                   ASCENXION   2021-05-14       0
## 4 1000010                 Crown Trick   2020-10-16       0
## 5 1000020             Organ Corruption          \\N       0
## 6 1000030 Cook, Serve, Delicious! 3?!   2020-10-14       0
##                developer price            owners_range
## 1                  Valve   999 10,000,000 .. 20,000,000
## 2                  Valve   999 10,000,000 .. 20,000,000
## 3 IndigoBlue Game Studio   999             0 .. 20,000
## 4           NEXT Studios  1999     500,000 .. 1,000,000
## 5                    \\N   \\N             0 .. 20,000
## 6    Vertigo Gaming Inc.  1999       100,000 .. 200,000
##                                genres  total positive negative review_score
## 1                              Action 241610   235403     6207    0.9743098
## 2                              Action     NA       NA                    NA
## 3             Action, Adventure, Indie     37       30        7    0.8108108
## 4    Adventure, Indie, RPG, Strategy   5150     4411      739    0.8565049
## 5           Action, Indie, Simulation      0        0        0          NaN
## 6 Action, Indie, Simulation, Strategy   2086     1902      184    0.9117929
##   recommendations
## 1          153259
## 2                
## 3             \\N
## 4            4339
## 5             \\N
## 6            1614
```

Now we have a single data frame that consists of the variables **app_id, name, genre, is_free, developer, price, release_date, owners_range, total, positive, negative, review_score,** and **recommendations**.

Methods / Interactive Visualizations

For our first visualization, lets look at the spread of number of downloads range to the number of games.

```
# Create a bar chart for owners_range
gameData %>%
  count(owners_range) %>%
  ggplot(aes(x = reorder(owners_range, n), y = n)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  geom_text(aes(label = n), vjust = -0.1, color = "black") +  # Adds the count above eac
h bar
  labs(title = "Frequency of Owner Ranges in Games",
       x = "Owner Ranges",
       y = "Count of Games") +
  scale_y_continuous(labels = label_comma()) +  # Formats y-axis to display raw numbers
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

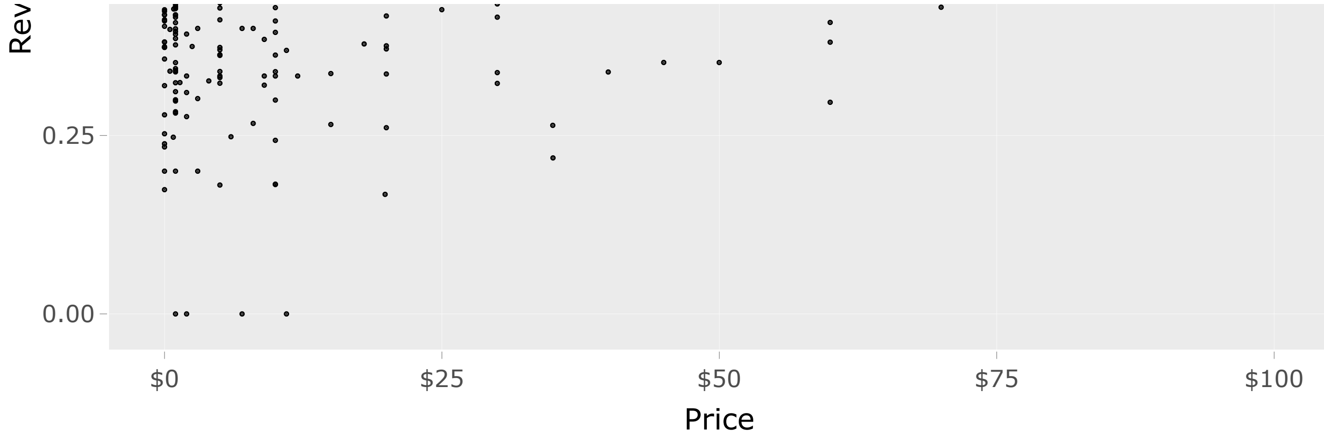


As we can see, there are a very few amount of games that have a large impact on the gaming community. To look into which games are more attractive to video game players, lets reduce our dataset of the games in the “200,000 .. 500,000” range and above (4,953 games) and study them more closely.

Plot our first Visualization

By using only the top ~5,000 games, we have a better judge of what makes these games become in this category. Lets solidify these by putting them in a data frame of their own.

```
mostDownloadedGames <- gameData[gameData$owners_range %in% c(
  "200,000 .. 500,000",
  "500,000 .. 1,000,000",
  "1,000,000 .. 2,000,000",
  "2,000,000 .. 5,000,000",
  "5,000,000 .. 10,000,000",
  "10,000,000 .. 20,000,000",
  "20,000,000 .. 50,000,000",
  "50,000,000 .. 100,000,000",
  "100,000,000 .. 200,000,000",
  "200,000,000 .. 500,000,000"
), ]

# Convert 'price' to numeric (remove any commas and convert it to a number)
mostDownloadedGames$price <- as.numeric(gsub(",", "", mostDownloadedGames$price))

# Fit the linear model
lm_model <- lm(review_score ~ price, data = mostDownloadedGames)

# Get the R-squared value
r_squared <- summary(lm_model)$r.squared

# Scatter plot with regression line
p <- mostDownloadedGames %>%
  ggplot(aes(x = price, y = review_score)) +
  geom_point(size = 0.2) +
  geom_smooth(method = "lm", se = FALSE, color = "red") +  # Add regression line
  labs(title = "Correlation between Price and Review Score",
       x = "Price",
       y = "Review Score") +
  scale_x_continuous(labels = scales::label_dollar(scale = 0.01))  # Formatting price

# Make the ggplot interactive with plotly
interactive_plot <- ggplotly(p)

# Display the interactive plot
interactive_plot
```




```
# Print the R-squared value
print(paste("R-squared: ", round(r_squared, 3)))
```

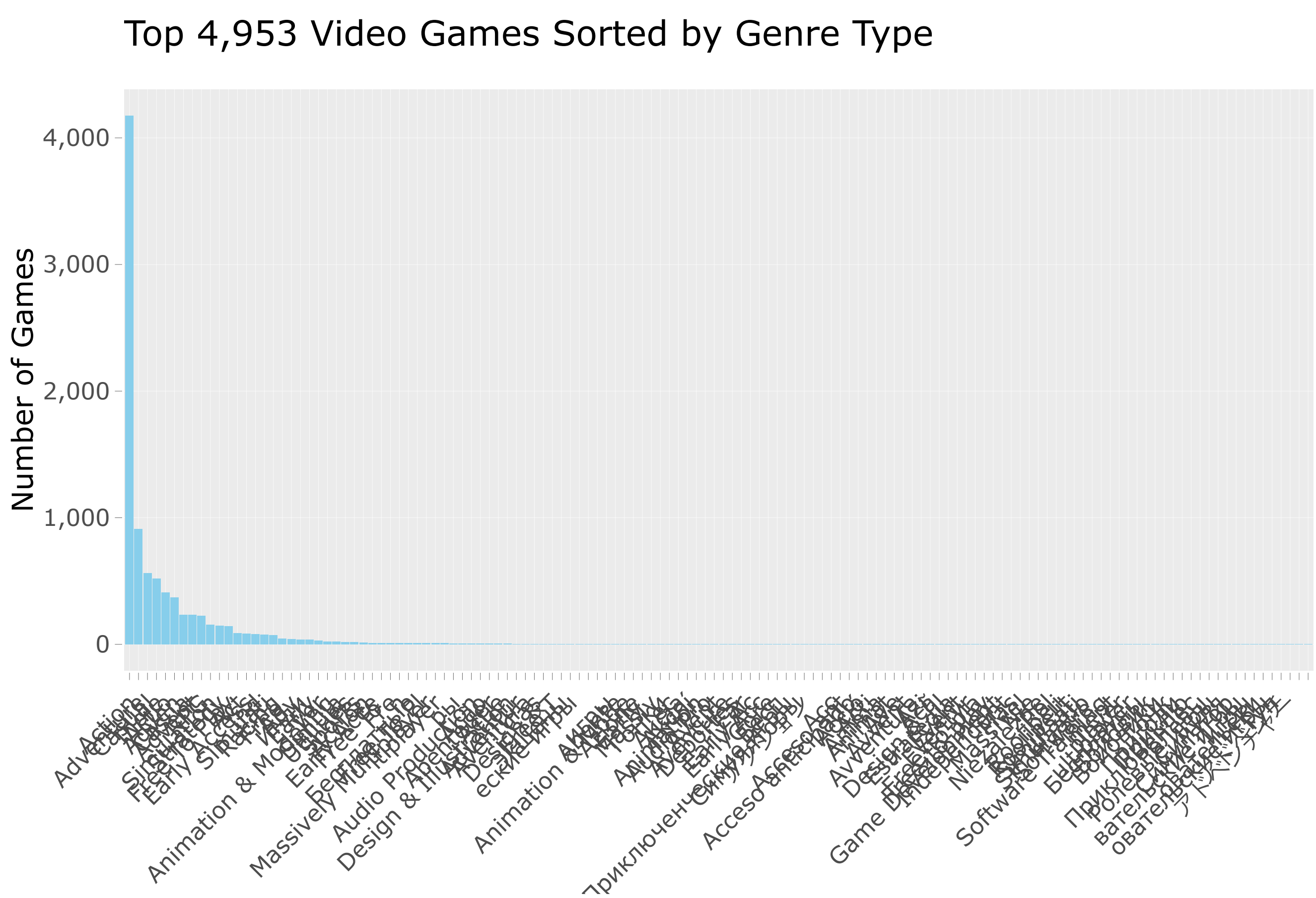
```
## [1] "R-squared:  0.014"
```

Since the R-squared value is 0.014, there is a very weak correlation between the review score and the price.

My Second Visualization

Genre is a great way to get an essence of what a game is. Next, lets investigate how genre effects the rating.

```r
# Create the genre count dataset, excluding empty or NA genres
genre_count <- mostDownloadedGames %>%
  separate_rows(genres, sep = ", ") %>%  # Split genres by comma and create a new row fo
r each genre
  filter(genres != "" & !is.na(genres)) %>%  # Remove empty or NA genres
  count(genres, sort = TRUE) %>%  # Count how many times each genre appears
  arrange(desc(n))  # Explicitly sort by count in descending order

# Convert genres to an ordered factor based on the count
genre_count$genres <- factor(genre_count$genres, levels = genre_count$genres)

# Plot the frequency of genres as a bar chart
p <- genre_count %>%
  ggplot(aes(x = genres, y = n)) +
  geom_bar(stat = "identity", fill = "skyblue") +  # Use bars to represent the counts
  labs(title = "Top 4,953 Video Games Sorted by Genre Type",
       x = "Genres",
       y = "Number of Games") +
  scale_y_continuous(labels = label_comma()) +  # Formats y-axis to display raw numbers
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis labels for b
etter readability

# Make the ggplot interactive with plotly
interactive_plot <- ggplotly(p)

# Display the interactive plot
interactive_plot
```



Genres

There are a few top genres. This graph and the one before it are interactive graphs. Click and draw to create a box to zoom in. Double click to zoom out.

Results

As we can see, there is not a correlation between price and rating, suggesting that games that are more graphically intensive or have more effort put into them do not attract the most customers. We can also see that there are prominent genres of video games that appeal to people the most. Among these being Action, Adventure, Indie, and Casual. Video game developers should market their efforts towards these genres to receive the highest feedback from video gamers.