# Index

# 1. Introduction

Digital distribution platforms have profoundly transformed the video game industry, reshaping not only how games are produced and distributed, but also how they are consumed, evaluated, and sustained over time. Among these platforms, Steam occupies a central position in the PC gaming ecosystem, acting simultaneously as a marketplace, a distribution infrastructure, and a social environment in which players interact, review products, and generate large volumes of behavioral data. With tens of thousands of available titles and millions of active users worldwide, Steam represents one of the most complex and dense digital marketplaces in contemporary cultural industries.

The scale and heterogeneity of the Steam catalogue introduce significant analytical challenges. While a small number of blockbuster titles attract massive engagement and sustained visibility, the vast majority of games remain marginal, sporadically consumed, or never actively played. This highly skewed structure raises fundamental questions about how success, engagement, and visibility emerge on the platform, and whether they can be understood as variations along a single performance dimension or as the result of multiple and distinct strategic approaches.

Recent academic research has increasingly leveraged Steam data to study popularity prediction, recommendation systems, and the drivers of commercial success using machine learning techniques. These contributions have identified several key indicators as strong predictors of short-term performance. At the same time, descriptive analyses have documented the extreme heterogeneity of the catalogue and the presence of long-tail dynamics. However, much of this literature remains fragmented: predictive approaches often focus on isolated outcomes, while descriptive studies lack a systematic representation of the catalogue's internal structure.

Against this background, the present thesis adopts a different perspective: rather than trying to predict the success of individual games, it seeks to provide a structural and interpretable representation of the Steam catalogue as a whole. The central premise of the thesis is that video games on Steam can be meaningfully grouped into a limited number of coherent segments, defined by recurring combinations of economic, technical, and engagement-related characteristics. These segments reflect different modes of

production, distribution, and consumption, and correspond to diverse forms of success rather than to a single dominant model.

To investigate this premise, the thesis employs a large-scale quantitative analysis of the Steam Games Dataset, integrating unsupervised and supervised machine learning techniques. A clustering analysis is first used to uncover latent structures within the catalogue, organizing games into interpretable groups based on numerical indicators such as price, technical accessibility, internationalization, and user engagement. Subsequently, a Random Forest classifier combined with SHAP-based explanations is applied to validate the robustness of the segmentation and to clarify the role of individual variables in defining each cluster. This interpretative layer allows the analysis to move beyond purely descriptive segmentation and to provide transparent explanations of how different clusters are formed.

In addition to numerical analysis, the thesis explores the textual descriptions associated with games on Steam as a complementary source of information. While textual embeddings are shown to have limited explanatory power for clustering, a descriptive analysis of linguistic patterns contributes to enriching the interpretation of the identified segments. By integrating these insights, the study derives a set of consumer profiles that translate quantitative structures into meaningful narratives of video game consumption. These profiles do not represent empirically observed users, but analytical constructions that summarize recurring modes of engagement inferred from the data.

As a consequence, the objective of the thesis is to understand the structural logic of the Steam catalogue and to show how heterogeneous consumption models coexist within a single digital distribution platform.

# 2. Literature Review

## 2.1 Digital distribution and the transformation of the video game industry

The digital distribution of software content has been technically possible for decades, however, in the video game sector its consolidation is closely tied to 1990s PC culture. During this period, practices such as shareware (software distributed through informal networks and often free at least in an initial phase) and modding (user-driven game modification) became largely popular. In the early 2000s, some publishers began experimenting with more structured forms of sales based on downloads, and by 2004 industry analysts were already predicting that this distribution model would become a central component of the market within roughly a decade (Gaudiosi, 2004).

A major turning point in this transition was Valve's entry into online distribution. Already established as a developer through titles such as Half-Life (1998) and Counter-Strike (2000), Valve launched the Steam platform for PC in 2003, anticipating many of the dynamics that would later shape the market (Walker, 2003). At launch, Valve reported more than one million users, around 75% of whom had broadband access (Walker, 2003). Steam then expanded rapidly: by 2007 it had surpassed 13 million users and offered more than 250 downloadable games ("Valve's Steam…," 2007). In the following years, the growth of both the catalogue and the user base contributed to positioning Valve as a leading global distributor of PC softwares (Bergen & Palmeri, 2018). By 2019, Steam offered more than 30,000 titles and, for many developers, represented the predominant revenue source, in some cases accounting for more than 75% of total sales (Bailey, 2019). In this sense, Valve's trajectory illustrates how the shift from game development to operating a distribution platform can generate long-term stability, control over market access, and durable revenue streams (Nichols, 2023).

A second key factor in the consolidation of the contemporary video game industry was the rise of smartphones. The launch of the iPhone in 2007 and the opening of the App Store in 2008 are often described as a structural shift in the sector. By 2009, tens of thousands of independent developers were submitting apps for Apple's review process, with a catalogue that quickly exceeded 100,000 products and surpassed two billion total downloads (Wortham, 2009). In parallel, Google launched Android Market in 2008,

which expanded from a few dozen initial apps to more than 2.6 million applications by 2018 (Pan, 2018). Together, these developments contributed to the emergence of a full-fledged "app economy," with annual revenues estimated above 150 billion dollars (Bergen & Palmeri, 2018).

The relevance of these changes goes beyond purely technological innovation. Unlike Valve, which built its market power starting from internally developed games, Apple and Google entered the video game sector as external actors, leveraging capabilities and dominance acquired in adjacent markets. This process marked a profound change in industrial structure, reshaping power relations along the value chain (Nichols, 2023). Importantly, the spread of online distribution did not eliminate the traditional organization of the industry; rather, it introduced a strategic choice for developers. On the one hand, a "mainstream" pathway remains dominated by large publishers and established hardware producers such as Nintendo, Sony, and Microsoft. On the other hand, a platform-centered pathway has emerged, driven by digital storefronts and typically associated with newer entrants and a stronger role for online intermediaries.

From an economic perspective, one of the most significant effects of this transition concerns costs and profitability. The shift toward digital distribution reduces or eliminates several cost components associated with physical production, packaging, and logistics, altering the cost structure of game publishing and potentially improving profit margins for developers and publishers. However, these advantages are accompanied by intensified competition, as lower barriers to entry dramatically increase the number of titles available on digital storefronts. Depending on platform policies and distribution agreements, developers can retain a variable share of revenues, often shaped by the bargaining power of intermediaries. In contrast, building a proprietary platform, as in the cases of Valve or Epic, represents a rare but potentially transformative strategy, allowing firms to move from content production toward a central infrastructural role within the market (Nichols, 2023).

From the consumer side, online distribution introduces both new opportunities and new challenges: digital purchases typically grant a license to use the software rather than legal ownership, with the risk of losing access if content is removed from a store or has not been installed locally. Additional concerns relate to security and content authenticity: between 2018 and 2019, estimates pointed to hundreds of thousands of malware

downloads disguised as video game releases, particularly concentrated around highly anticipated titles (Bizcommunity, 2019). Finally, the large-scale collection of user data raises questions of privacy and regulation. As noted in critical literature, platform-based ecosystems can transform leisure time and playful interaction into systematic sources of economic value (deWinter et al., 2014; Nichols, 2023).

## 2.2 Platforms and ecosystems in the video game industry

To better contextualize these dynamics, it is useful to examine how the academic literature defines the complexity of the Video Game Industry (VGI). Goh, Al-Tabbaa, and Khan (2023) provide a comprehensive account of the VGI within a business and management framework. They describe the industry as emerging in the 1970s and 1980s, initially associated with arcade distribution, and later becoming the largest and most dynamic segment of global entertainment. This expansion has been accompanied by continuous technological transformations—from the diffusion of home consoles to the rise of mobile gaming—making the VGI a particularly fast-changing, interactive, and immersive context with significant cultural and social impact (Goh et al., 2023).

A central contribution of this work lies in its emphasis on the fragmentation of existing research. Studies on the VGI span multiple domains, including consumer psychology, innovation, monetization models, and corporate strategy, often without systematic integration. Through a systematic review of 84 empirical articles published between 2007 and 2022, the authors propose an evidence-based framework that connects key themes such as product ecosystems, pricing strategies, user behavior, innovation networks, and business model sustainability. From this perspective, the video game industry emerges as a complex system in which technological innovation, user experience, and institutional settings interact simultaneously, providing a solid theoretical foundation for large-scale quantitative analyses and data-driven approaches (Goh et al., 2023).

From a macroeconomic perspective, the video game industry is a rapidly expanding global sector and is highly heterogeneous across devices, regions, and demographics. Ozalp (2024) analyzes the industry's evolution through four national hubs, showing that growth does not follow a single trajectory but depends on historically contingent and institutionally embedded development paths. According to the author, successful hubs

emerge from combinations of technological capabilities, human capital, creative infrastructures, and supportive public policies.

In terms of market structure, Ozalp (2024) distinguishes the industry's traditional actors—developers, publishers, and platform owners—and highlights how digitalization has reshaped the PC segment. While PC gaming historically lacked licensing and royalty systems comparable to those of consoles, the rise of digital storefronts such as Steam has progressively integrated the PC into a platform-like model in which online intermediaries retain commissions on sales. At the same time, new actors—such as middleware providers and mobile SDK ecosystems—have emerged, further increasing interdependence and complexity. This context reinforces the view of an industry in which technological and organizational innovation is tightly intertwined with the structure and governance of digital platforms (Ozalp, 2024).

## 2.3 Steam as a digital platform and data-rich environment

Steam represents one of the earliest and most influential examples of a digital platform for video game distribution operating at global scale. Introduced by Valve in 2003, Steam did not merely adopt online delivery early on, but progressively consolidated a platform-based distribution model that would become central to the PC gaming ecosystem (Visser & Fontugne, 2024). Following the introduction of the first digital game on the store in 2004, the platform experienced rapid and sustained growth. By 2022 alone, users downloaded approximately 44.7 exabytes of data from Steam servers, highlighting the magnitude of the infrastructure underpinning contemporary game distribution (Visser & Fontugne, 2024). This expansion has been driven not only by growth in the user base, but also by changes in the nature of video games themselves.

Modern titles increasingly feature large file sizes and frequent updates, particularly in the case of multiplayer and live-service games. As shown by Visser and Fontugne (2024), the release of new games or major updates can generate substantial traffic peaks across broadband networks and Content Delivery Networks, distinguishing Steam's traffic patterns from those of many other digital content providers. As a result, Steam operates as a large-scale technical infrastructure with distribution requirements that are structurally different from those of streaming or software update platforms.

A key factor differentiating Steam from services such as Netflix or operating system update mechanisms lies in the mode of content consumption. While video streaming allows for immediate and progressive access, video games generally require the complete download of the product before use. This introduces a strong urgency component at release, as users tend to demand immediate access, while the resulting network load cannot be smoothed over time in the same way as streaming traffic (Visser & Fontugne, 2024). Moreover, major game releases often occur simultaneously at the global level, producing distinctive and highly synchronized traffic patterns.

The scale of the platform is further amplified by the variety of its catalogue. While video streaming services typically offer a few thousand titles, Steam hosts more than 150,000 products in its global store (Visser & Fontugne, 2024). The combination of catalogue magnitude, large file sizes, and simultaneous global releases positions Steam not only as a marketplace, but as a complex and highly informative digital environment, particularly well suited to large-scale quantitative analysis of products, users, and consumption dynamics.

## 2.4 Consumer behaviour, discoverability, and recommendation on Steam

The extreme variety of titles available on Steam gives rise to a central issue of discoverability: users may struggle to identify games that align with their preferences within an exceptionally broad and heterogeneous offering. As highlighted by Cheuque, Guzmán, and Parra (2019), Steam functions not merely as a digital storefront, but as an integrated platform that enables users to purchase games, write reviews, share opinions, and play both online and offline. However, the coexistence of millions of users and tens of thousands of games substantially complicates the decision-making process, particularly for newly released titles.

A particularly relevant insight reported in the literature is that approximately 37% of games purchased on Steam are never actually played by the users who buy them (Cheuque et al., 2019). This discrepancy between purchase and effective consumption underscores the difficulty users face in navigating the catalogue and reinforces the need for systems capable of guiding discovery. In this context, recommender systems play a

central role by providing personalized suggestions and balancing the visibility of popular games with the promotion of less-known titles.

An additional characteristic that differentiates video game consumption from other forms of media is its recurrent nature. Unlike film viewing, which is typically episodic, gaming tends to involve repeated engagement over time and is therefore closer to music listening (Cheuque et al., 2019). This behavioral pattern creates a dual challenge for platforms: on the one hand, encouraging continued engagement with games users already enjoy; on the other, facilitating the discovery of new titles that may become part of long-term consumption habits. From this perspective, the analysis of game attributes, user interactions, and expressed preferences becomes crucial for understanding how consumption dynamics emerge within the Steam catalogue.

## 2.5 Predicting game popularity and machine learning approaches

A relevant strand of the literature has focused on predicting the popularity of video games on Steam using machine learning techniques. De Luisa et al. (2021) develop and evaluate predictive models aimed at forecasting game popularity in the early stages following release, operationalizing popularity through the number of active players over time. In particular, the authors adopt the median player count in the second month after release as their primary prediction target, showing that model performance tends to decline as the temporal horizon of the prediction is extended (De Luisa et al., 2021).

An important contribution of this work lies in its emphasis on understanding the influence of individual features on prediction outcomes, rather than focusing exclusively on overall model accuracy. Nevertheless, the analysis remains centered on a specific definition of popularity and a relatively short time window. It does not extend to a broader structural analysis of the Steam catalogue or to a segmentation of games based on their characteristics. As a result, while the approach provides valuable insights into early-stage performance, it leaves open questions regarding how different types of games are positioned within the platform and how heterogeneous consumption patterns may arise.

More recent work has expanded the scope of analysis by examining the drivers of commercial success across multiple platforms. Ma (2025) proposes an integrated study combining data from Steam, Twitch, and Metacritic to identify the relative influence of different platforms on video game revenue. Using machine learning models, the study shows that Steam-related metrics are the strongest predictors of revenue, followed by indicators from Twitch and, to a lesser extent, aggregated Metacritic scores (Ma, 2025).

Specifically, variables such as the number of followers on Steam, Peak Concurrent Users (Peak CCU), and total reviews emerge as the most influential factors in revenue prediction. The analysis further indicates that certain game attributes, such as the early access tag, are positively associated with revenue, whereas categories such as indie or free-to-play exhibit negative correlations (Ma, 2025). While these findings provide actionable insights for developers and publishers, the study also acknowledges key limitations, including potential selection bias toward commercially successful titles and the absence of a structured representation of the full Steam catalogue.

## 2.6 Critical synthesis and research gap

In parallel, several studies have focused on descriptive analyses of the Steam catalogue and the information available at the platform level. Research based on user interactions, game tags, and purchase histories consistently shows that the catalogue is highly heterogeneous and strongly skewed, with a small number of extremely popular titles and a long tail of games characterized by limited visibility and consumption Salkanović (2024).

While these contributions demonstrate the richness of Steam data and enable exploration across multiple dimensions of the video game product, they largely remain confined to exploratory or descriptive perspectives. As such, they do not provide a systematic segmentation of games nor establish a direct link between game characteristics and patterns of consumption or success. This highlights the need for more structured approaches that integrate descriptive analysis with statistical and machine learning methods.

Overall, the reviewed literature positions Steam as a unique digital distribution platform in terms of scale, complexity, and informational richness (Visser & Fontugne, 2024). Existing research has addressed early-stage popularity prediction (De Luisa et al., 2021), discoverability and recommendation challenges in large catalogues (Cheuque et al., 2019), descriptive analyses of game attributes Salkanović (2024), and the identification of success drivers through interpretable machine learning models (Ma, 2025).

However, a substantial gap remains. These strands of research tend to treat performance prediction, feature analysis, and recommendation as separate problems, without integrating a structural segmentation of the Steam catalogue with interpretable predictive modeling and a perspective explicitly oriented toward consumption profiles. In particular, the literature lacks an approach that jointly combines large-scale descriptive analysis, clustering of games based on their characteristics, model interpretability, and the construction of gamer profiles derived from emerging patterns. Addressing this gap motivates the present thesis, which aims to analyze the Steam catalogue systematically by integrating statistical and machine learning techniques to provide a structured and interpretable account of video game success and consumption dynamics on the platform.

## 2.7 Research Objectives, Research Gap and Scope of the Thesis

### 2.7.1 Research motivation and identification of the gap

Building on the literature reviewed in the previous sections, this thesis is motivated by the need for a more structured and integrative understanding of the Steam catalogue as a complex digital ecosystem. Existing research has convincingly demonstrated the scale, heterogeneity, and informational richness of Steam, as well as its central role in contemporary video game distribution. At the same time, prior studies have tended to approach the platform through partially disconnected analytical perspectives, focusing either on popularity prediction, recommendation systems, or descriptive analyses of specific features.

Despite the increasing availability of large-scale platform data and advanced machine learning techniques, the literature remains fragmented in the way it conceptualizes video game success and consumption on Steam. Predictive studies typically aim to forecast

short-term popularity or revenue using selected indicators, while descriptive contributions document skewed distributions and long-tail dynamics without providing a coherent structural representation of the catalogue. As a result, success is often treated as a single outcome variable, instead of being treated as the manifestation of different consumption approaches.

This reveals a central gap in the existing literature. While it is well established that the Steam catalogue is highly uneven and dominated by a small number of blockbuster titles, there is limited understanding of how different types of games coexist on the platform, which structural dimensions distinguish them, and whether multiple, qualitatively distinct forms of success emerge beyond simple popularity rankings. In particular, existing approaches rarely investigate whether engagement, pricing strategies, technical accessibility, and internationalization combine into recurrent configurations that define stable segments within the catalogue.

Moreover, although recent work has emphasized the importance of model interpretability and feature importance in understanding game performance, these insights are rarely embedded within a broader segmentation framework. Consequently, the literature lacks an approach that jointly combines large-scale descriptive analysis, systematic clustering of games, and interpretable modeling to explain the structural organization of the Steam ecosystem as a whole.

### 2.7.2 Research objectives and research questions

In response to the limitations identified above, the primary objective of this thesis is to provide a structured and interpretable segmentation of the Steam catalogue based on economic, technical, and engagement-related characteristics. Rather than focusing on the prediction of individual outcomes, the study aims to uncover latent groupings of games and to understand the mechanisms that differentiate them.

More specifically, the thesis addresses the following research questions:

- **RQ1**: Can the Steam catalogue be meaningfully segmented into a limited number of coherent and interpretable clusters?
- **RQ2**: Which variables play a dominant role in defining these clusters?

- **RQ3**: Do the identified clusters correspond to distinct patterns of consumption and engagement?
- **RQ4**: Can machine learning techniques improve the understanding of cluster structure?

By addressing these questions, the thesis moves beyond basic measures of performance to uncover the specific patterns that differentiate various groups of games. Rather than focusing on individual sales or the immediate impact of advertising, this research explores how the Steam catalogue is organized and how diverse models of consumption coexist within the same space. To map this structure, the next chapter introduces the methodology, explaining how the data was gathered and analyzed to create the clusters that form the core of this study.

# 3. Dataset Description and Methodology

## 3.1 Dataset Overview

This study relies on the Steam Games Dataset, a large-scale collection of game-level information, designed to capture commercial performances and community engagement outcomes on the Steam platform. Steam is the largest digital distribution platform for PC gaming worldwide, and the dataset is constructed by combining data retrieved through the official Steam Web API with complementary information obtained from Steam Spy. The Steam API provides structured and regularly updated metadata directly from the platform, including details on pricing, release dates, supported systems, user activity, and reviews. Steam Spy is an independent analytical service that estimates ownership figures and player engagement by leveraging publicly available Steam profile data and statistical sampling techniques. While Steam does not disclose exact sales numbers, Steam Spy offers widely used ownership estimates that enrich API data and allow for a more comprehensive analysis of market performance. Importantly, all data refer to a global scope, without geographic restrictions.

After downloading the dataset, it initially contained 111.452 observations and 39 variables. Each row corresponds to a single video game uniquely identified by a Steam application identifier (*AppID*). The dataset includes information related to release timing, pricing strategies, supported operating systems, user engagement, and reception, as well as several metadata fields describing the creative and commercial aspects of each title. Specifically, the variables are as follows: *AppID* uniquely identifies each game on Steam, while *Name* reports the game title and *Release.date* indicates its launch date. Commercial reach is approximated through *Estimated.owners*, expressed as a range (e.g., "0–20,000"), and *PeakCCU*, which captures the maximum number of concurrent players. Regulatory and access features are represented by *Required_age* and *Price*, with free-to-play titles assigned a value of zero. Content extensibility is measured through *Discount.DLC.count*, while textual and descriptive information is provided by *About.the.game*. Linguistic accessibility is captured by *Supported.languages* and *Full.audio.languages*. User feedback is summarized through *Reviews*, *Positive*, *Negative*, *User.score*, *scoreRank*, and *Recommendations*, while external critical

reception is reflected in *Metacritic.score* and *Metacritic.URL*. Technical compatibility is described by the binary variables *Windows*, *Mac*, and *Linux*. Engagement depth is further measured through *Achievements* and detailed playtime metrics, including *Average.playtime.forever*, *Average.playtime.two.weeks*, *Median.playtime.forever*, and *Median.playtime.two.weeks*. Additional metadata include *Developers*, *Publishers*, *Categories*, *Genres*, and *Tags*, which collectively describe the creative identity and market positioning of each game. Finally, the dataset contains several auxiliary variables related to store presence and communication, such as *Header.Image*, *Website*, *Support.URL*, *Support.email*, *Screenshots*, *Movies*, and *Notes*.

The temporal coverage of the dataset begins on April 24, 2022, with monthly updates thereafter. The dataset used in this study was downloaded in July 2025, meaning that all variables are updated through June 2025.

## 3.2 Data Quality Assessment and Missing Values

A preliminary inspection of the dataset revealed the presence of a substantial number of missing values across several variables. To examine the level of data completeness, the proportion of missing values (NA) was calculated for each variable by dividing the number of missing observations by the total number of rows. The resulting indicator was mapped onto a continuous scale ranging from 0 to 1, where 0 denotes the absence of missing values and 1 indicates that the variable is entirely missing (100% NA). This approach allowed for a consistent and comparable evaluation of data quality across all features.

To reduce potential distortions caused by variables with limited data availability, a filtering rule was applied: all variables exhibiting a missing-value proportion greater than 0.3 (NA share > 0.30) were excluded from the analyses.

Based on this condition, the following variables were excluded from the dataset:

- **Tags** (33.58% missing values)
- **Support.url** (54.41% missing values)
- **Website** (58.32% missing values)

- **Notes** (83.45% missing values)
- **Reviews** (90.47% missing values)
- **Metacritic.url** (96.41% missing values)
- **Score.rank** (100% missing values)

After completing the data cleaning and preprocessing steps, the final dataset was composed of 73.864 observations.

## 3.3 Data Type Standardization and Feature Engineering

After the initial data cleaning phase, several variables were reformatted and transformed to improve readability and suitability for subsequent analyses. First, the variable *Estimated.owners*, originally reported as a range of values, was decomposed into two separate variables representing the lower and upper bounds (*owners_min* and *owners_max*). These were then used to compute a single continuous indicator (*estimated.owners.avg*), allowing ownership estimates to be expressed as a numerical value suitable for standard quantitative analyses.

### 3.3.1 Variables Transformation

The *Release.date* variable was converted into a proper date format to enable time-based analyses. Binary variables indicating operating system compatibility (*Windows*, *Mac*, and *Linux*) were converted into dummy variables using a 0–1 encoding. Several new derived variables were also constructed to capture relevant aspects of game characteristics. Specifically, a synthetic indicator named *device_support* was created by summing the three operating system dummies, resulting in a variable ranging from 1 to 3 that reflects the number of supported platforms.

Similarly, the variable *n_languages* was generated to represent the number of supported languages per game, and an additional variable capturing the number of supported categories (e.g., single-player, multiplayer) was derived from the corresponding categorical field and named *n_categories.*

User perception was also examined by constructing a *Review_Ratio* variable, defined as the proportion of positive reviews over the total number of reviews. Values close to one indicate predominantly positive reception, while lower values reflect more negative or polarized user evaluations.

### 3.3.2 Preliminary Descriptive Analysis

Following these transformations, a set of preliminary descriptive analyses was conducted to explore the main characteristics of the dataset. Summary statistics were computed for all meaningful quantitative variables, excluding identifiers or variables for which measures such as mean, minimum, or maximum were not conceptually relevant (e.g., game identifiers). When reporting minimum values, only variables with strictly positive lower bounds were considered, as many features naturally include zero values by construction.

**Summary Statistics (Minimum > 0 when applicable)**

| Variable | Min | Max | Mean | Median |
|---|---|---|---|---|
| Achievements | 1.00 | 9821 | 22.84 | 6.00 |
| Average.playtime.forever | 1.00 | 145727 | 121.61 | 0.00 |
| Average.playtime.two.weeks | 1.00 | 19159 | 13.67 | 0.00 |
| DiscountDLC.count | 1.00 | 2366 | 0.61 | 0.00 |
| Estimated.owners_avg | 10000.00 | 150000000 | 101408.33 | 10000.00 |
| Median.playtime.forever | 1.00 | 208473 | 108.49 | 0.00 |
| Median.playtime.two.weeks | 1.00 | 19159 | 14.73 | 0.00 |
| Metacritic.score | 8.00 | 97 | 3.92 | 0.00 |
| Negative | 1.00 | 895978 | 189.91 | 4.00 |
| Peak.CCU | 1.00 | 1311366 | 266.40 | 0.00 |
| Positive | 1.00 | 5764420 | 1138.23 | 14.00 |
| Price | 0.35 | 999 | 8.20 | 4.99 |
| Recommendations | 101.00 | 3441592 | 919.81 | 0.00 |
| Required.age | 1.00 | 21 | 0.33 | 0.00 |
| Review_Ratio | 0.02 | 1 | 0.76 | 0.83 |
| Total_Reviews | 1.00 | 6531097 | 1328.13 | 19.00 |
| User.score | 46.00 | 100 | 0.05 | 0.00 |
| Year | 1997.00 | 2025 | 2019.78 | 2020.00 |
| cluster_k7 | 1.00 | 7 | 3.77 | 3.00 |
| device_support | 1.00 | 3 | 1.35 | 1.00 |
| n_categories | 1.00 | 22 | 3.49 | 3.00 |

*Image 1: Summary statystics of the numerical variables of the dataset whith the exclusion of 0 as a minimum.*

Pie charts were used to illustrate the distribution of operating system support across games:

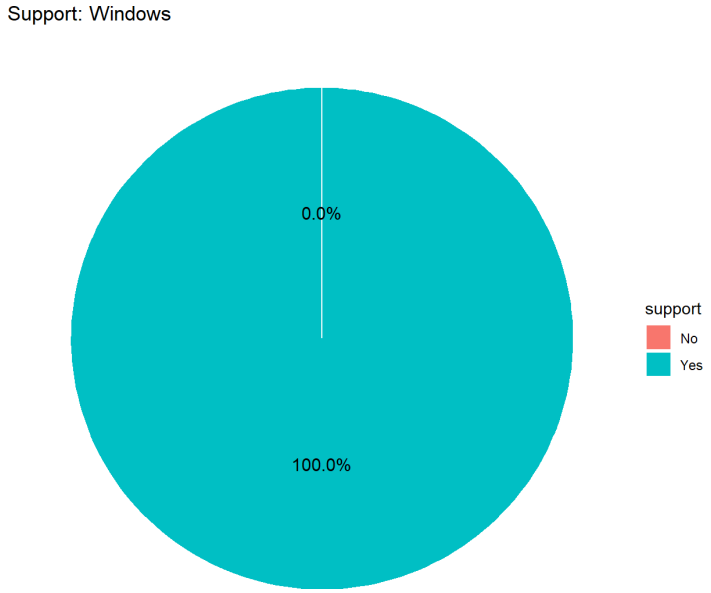


*Image 2: Graph showing the compatibility of Windows with the games in the dataset.*

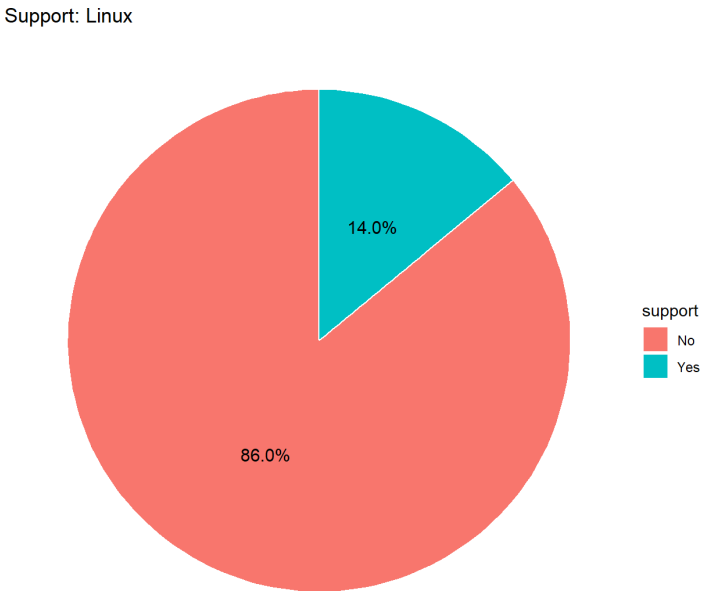


*Image 3: Graph showing the compatibility of Mac with the games in the dataset*

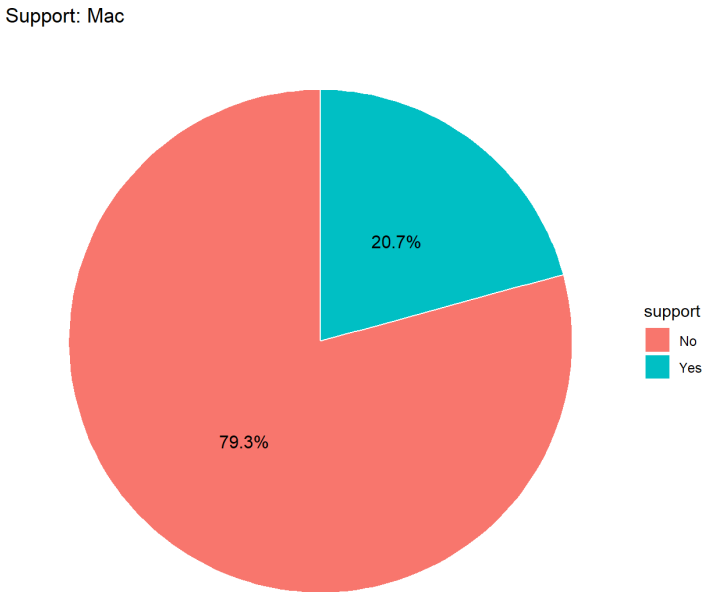


*Image 4: Graph showing the compatibility of Linux with the games in the dataset.*

The results showed that Windows compatibility is universal across the dataset, with 100% of games supporting the Windows platform. In contrast, support for macOS and Linux is substantially lower: only about 20.7% of games are compatible with macOS, and approximately 14% with Linux. This highlights a strong platform asymmetry in the PC gaming ecosystem, where Windows clearly represents the dominant target environment for developers. The limited support for macOS and Linux suggests higher development and maintenance costs relative to their smaller user bases, reinforcing Windows as the default and most economically viable platform for game distribution on Steam.

A time-based analysis was conducted by visualizing the average price of games by release year, highlighting general pricing trends over time:

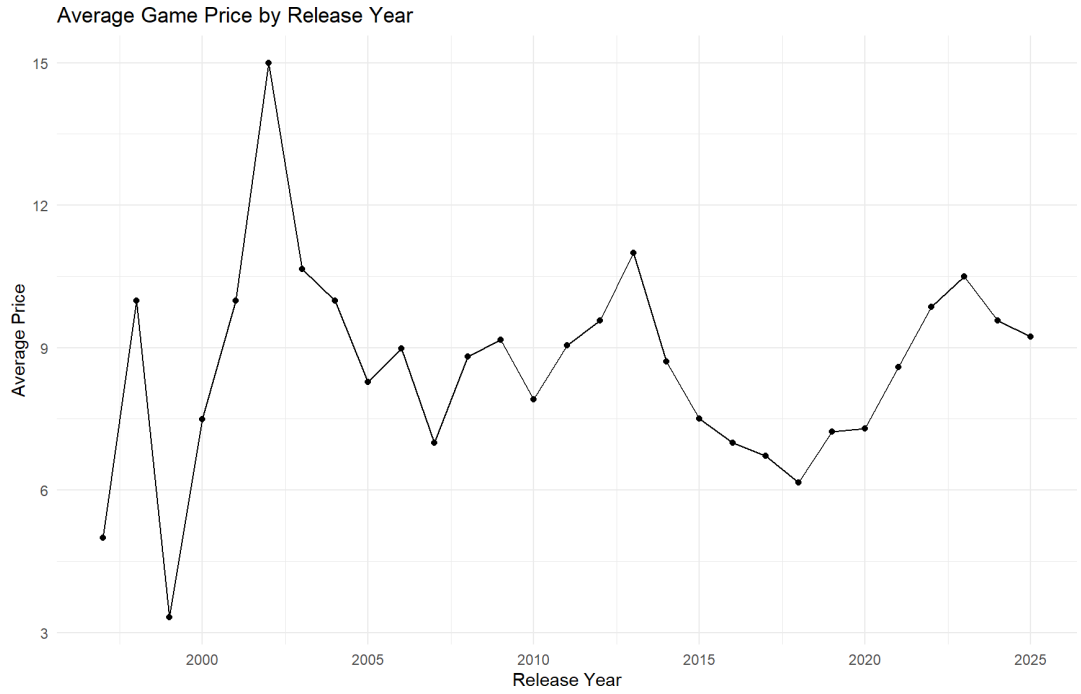


*Image 5: Average price distribution trhoughout games' release years.*

This graph shows that the average price during the years has been mostly between $6 and $10, with a significant spike in 2002 where the average price was about 15$.

The number of available achievements per game was grouped into six discrete classes (0, 1–10, 11–25, 26–50, 51–100, and more than 100) and visualized to assess how achievement design is distributed across titles.

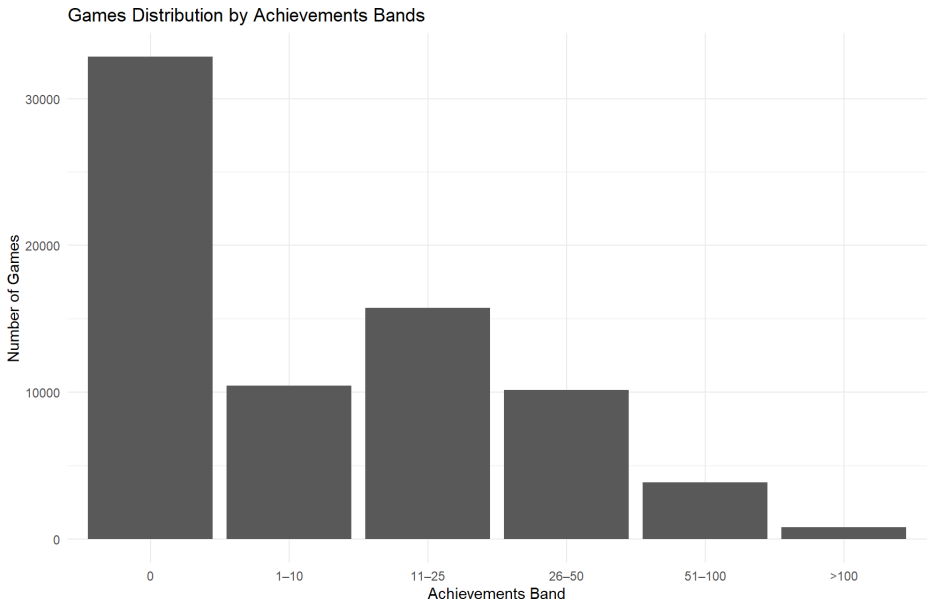


*Image 6: Number of achievements distributed per games.*

Lastly, the relationship between user engagement and commercial reach was explored through a scatter plot depicting the association between *PeakCCU* (peak concurrent users) and *estimated.owners.avg*. A regression line was added to provide an initial indication of the relationship between these two variables

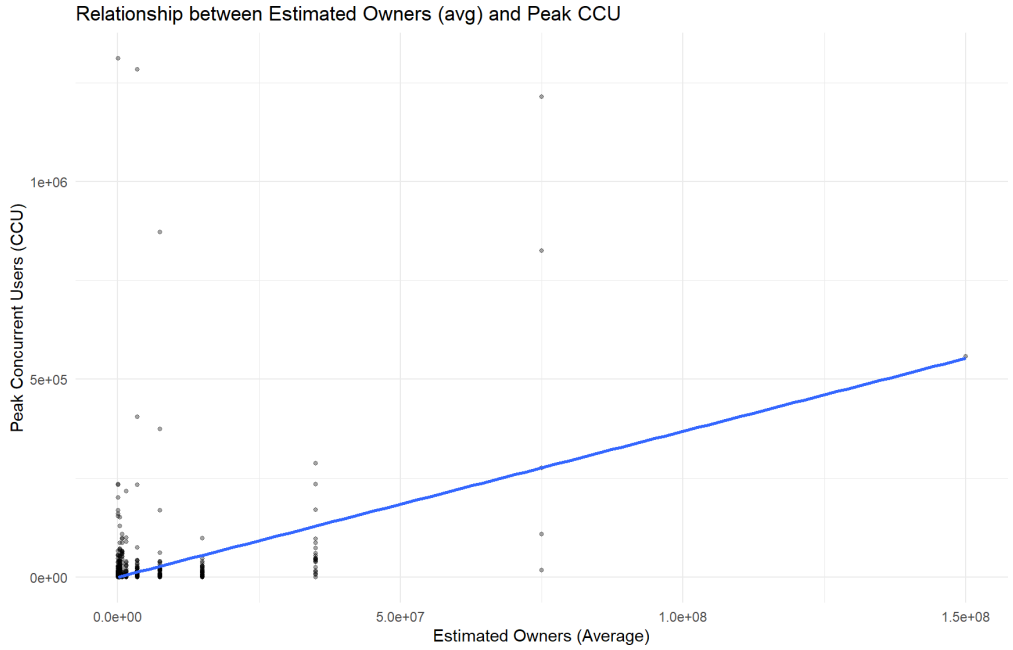


*Image 7 Regression analysis between estimated owners and peak concurrent users.*

The regression line shows only a marginal positive slope, which appears to be mainly driven by a small number of extremely popular titles (outliers), while the majority of games are concentrated in a region of the plot where no clear systematic relationship emerges.

Overall, these analyses are intended as **preliminary descriptive explorations** of the dataset, providing a foundational understanding of variable distributions, relationships, and data structure prior to more advanced analytical and clustering techniques.

# 4. Clustering Analysis

## 4.1 Clustering Motivations

The cluster analysis was performed to uncover latent structures within the dataset and to segment games based on a joint consideration of economic, technical, and structural variables. This approach was adopted to move beyond isolated variable-level analyses and to capture multidimensional patterns that describe how different characteristics combine within the catalogue. Given the size and heterogeneity of the dataset, clustering provided a way to organize the games into a limited number of interpretable groups, which could then be examined in greater detail and used as a foundation for subsequent comparative and interpretative analyses.

## 4.2 Variable Selection

The variables selected for the clustering analysis were: *Total_Reviews, device_support, n_languages, n_categories, Price***.** Some variables presented highly unbalanced distributions, with a small number of very large values, in particular *Total_Reviews, n_languages, and Price*. To mitigate the effect of extreme values and reduce asymmetry, a logarithmic transformation using log1p() was applied. This transformation computes the natural logarithm of (x + 1) and is commonly used when variables contain zeros, as it allows the data to be compressed without discarding zero-valued observations. After the transformation, all variables were standardized using the scale() function (mean equal to 0 and standard deviation equal to 1). This step ensures that all variables are expressed on a comparable scale and prevents variables with larger numerical ranges, such as *Total_Reviews*, from disproportionately influencing the clustering results.

Before performing the clustering, a correlation analysis was conducted among the selected to examine whether clear linear associations exist and to confirm that the variables capture complementary aspects of the data rather than repeated information.

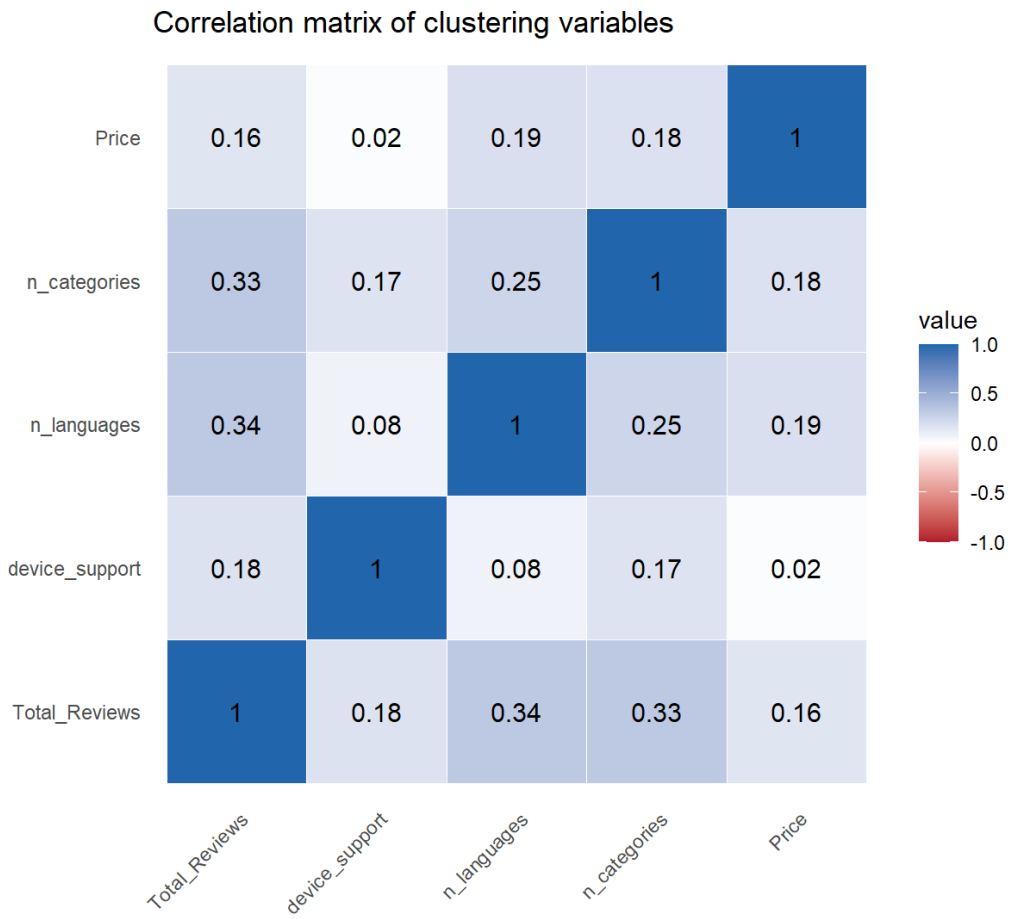


*Image 8: Correlation matrix between the variables used for the clustering analysis.*

## 4.3 Cluster Selection

The clustering analysis was performed using the K-means algorithm, which divides the observations into k distinct groups by minimizing the variance inside each group in the normalized feature space. The algorithm repeatedly assigns each observation to the nearest cluster centroid and updates centroid positions until convergence is reached. To correctly identify the ideal number of clusters, both the elbow method and silhouette analysis were used to evaluate the clustering structure across different values of k. The elbow method examines the relationship between the number of clusters and total variance within each cluster, with the best solution usually marked by a distinct point where adding more clusters yields minimal improvement. However, the resulting curve

displayed a smooth and gradual decrease, without a pronounced elbow, suggesting that no single value of k could be clearly identified as optimal.

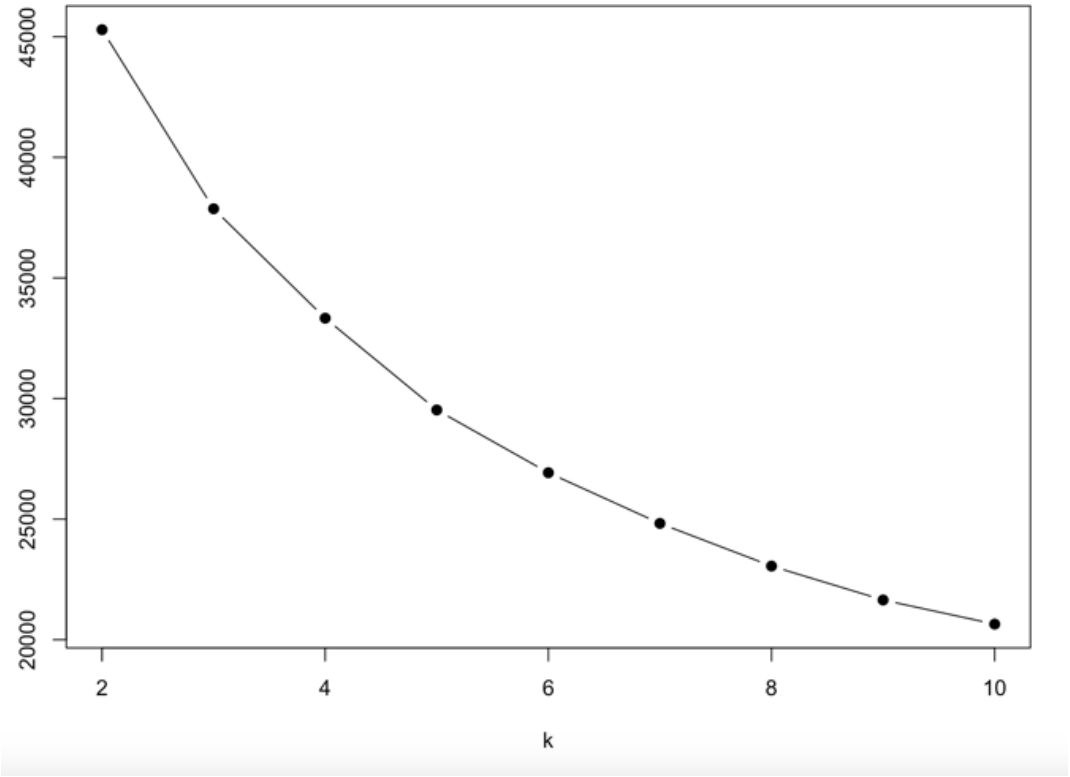


*Image 9: Elbow method.*

Similarly, the silhouette method, which assess the degree of cohesion within clusters and separation between clusters, did not exhibit a clear or stable maximum that could evidently indicate an optimal number of clusters. As shown in the silhouette plot, relatively high values are observed for lower values of *k* (specifically k=2 and k=3). A sharp decrease is then observed at k=4, followed by a modest and irregular increase for higher values of *k*, without any pronounced peak. This pattern suggests that the clustering structure does not naturally converge toward a single optimal solution.

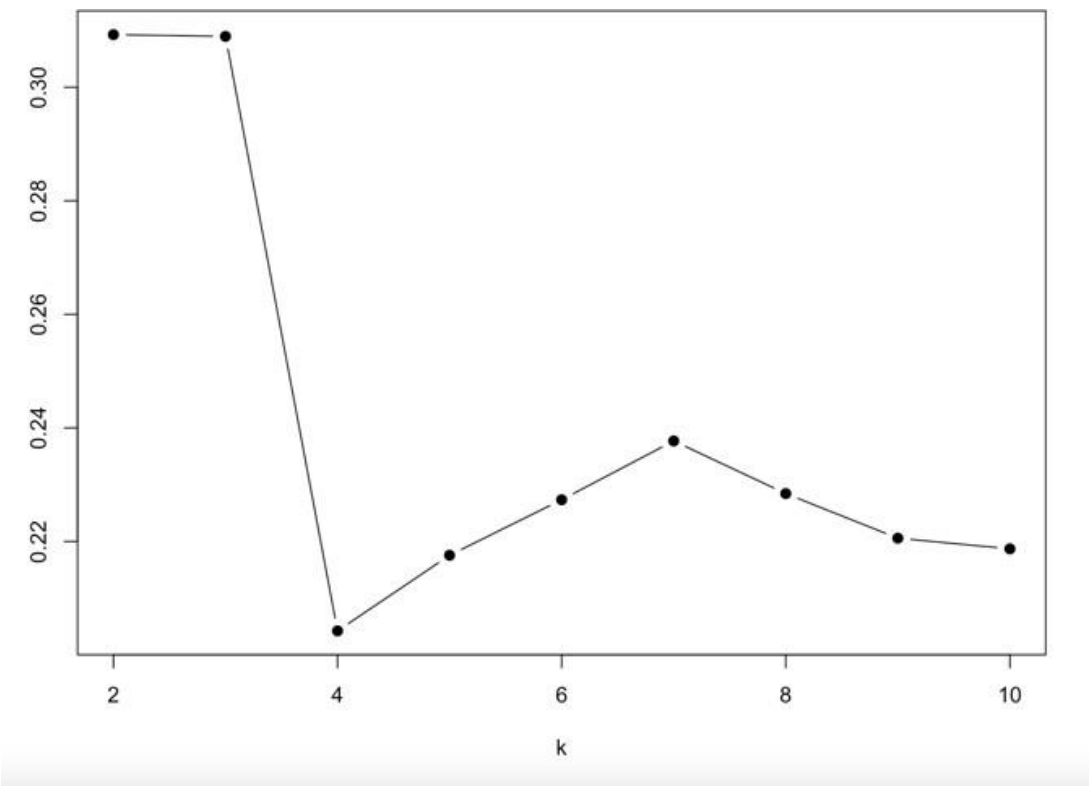


*Image 10: Silhouette method.*

Given the absence of a definite solution from these quantitative measures, a range of clustering solutions from k=3 to k=7 was further examined through qualitative assessment. This interpretative evaluation considered the internal coherence of clusters, their relative size and balance, and their analytical usefulness. Among the tested configurations, the solution with k=7 emerged as the most appropriate, as it produced clusters that were more specific and internally consistent while maintaining a more balanced distribution of observations compared to the other solutions.

Here is a table showing the values for the centroids of each cluster:

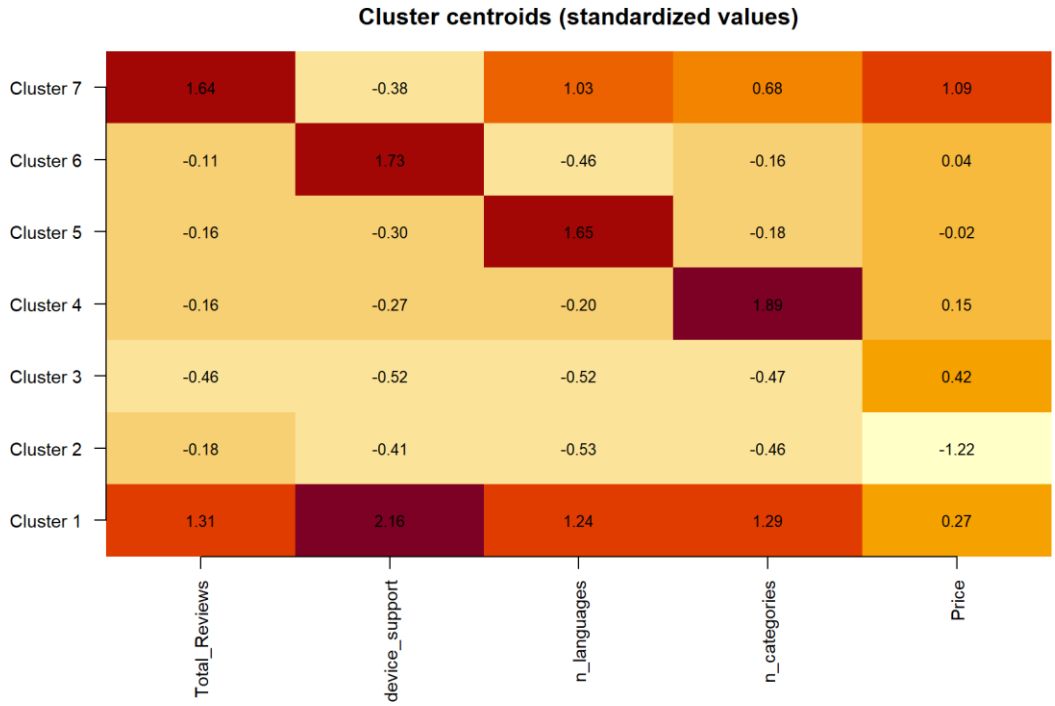


*Image 11: Clusters' centroids graph.*

Following, a small description of each cluster:

1. High values in almost all the variables, except for *Price,* and highest *device_support*
2. Low values in all variables, with the lowest *Price*
3. Low values, even more than k2, but with a higher *Price*
4. Average values with the highest *n_categories*
5. Average values with the highest *n_languages*
6. Average values with very high *device_support* (but not the highest)
7. High values in almost all the variables, except for *device_support*, with highest *Total_Reviews* and *Price*.

## 4.4 Clusters Interpretation

The resulting 7 clusters differ substantially in terms of performance, engagement, pricing, technical features, and distribution strategies. Each cluster can represent a specific market position within the Steam ecosystem, ranging from low-investment, low-engagement titles to premium, high-performing games with highly active communities. The clusters were labeled according to their relative tier (Low, Mid, Top) and their principal structural and strategic traits, as explained below

1. **Top-Tier: Premium (5.2%, 3,839 titles)**
   This cluster includes the highest-performing games in the dataset. Titles in this group exhibit very high values across all engagement metrics, with an average of approximately 600,000 owners, 9,800 reviews, and a *Peak.CCU* value of 830. *Average.playtime.forever* is extremely high (423 hours), while the *device_support* (2.8) indicates excellent compatibility across the three main operating systems. The high number of *n_languages* (9.2) and *n_categories* (7) suggests that these games are designed for international distribution. The concentration of release years between 2016 and 2018 coincides with a major expansion phase of the Steam platform, and current performance levels indicate that these titles continue to sustain a very active user base. Overall, this cluster represents premium, widely adopted games characterized by strong and long-lasting engagement.

2. **Low-Tier: Cheap (21.7%, 16,031 titles)**
   This cluster is composed of very low-priced games, with an average *Price* of $0.98 and frequent price points at $0, $0.99, and $1.99. Engagement indicators are extremely limited, with very low *Peak.CCU* values (around 10), relatively low ownership levels (41,000 *Estimated.owners_avg*), and a small number of *Total_Reviews* (168). The number of *n_categories* and *n_languages* is minimal and highly repetitive, consistent with low-investment productions. Low engagement is further reflected in short *Average.playtime.forever* (70 hours) and weak community response, as evidenced by the lowest *Metacritic.score* in the dataset (0.6). This cluster represents volume-oriented products with minimal development investment and limited player involvement.

3. **Low-Tier: Static (30.8%, 22,728 titles)**

   This cluster displays the lowest usage and engagement levels in the entire dataset. Games in this group have nearly zero *Peak.CCU*, an average of 17,000 *Estimated.owners_avg*, and only 36 *Total_reviews* per title. Despite this, the average *Price* is relatively high ($10) compared to the observed engagement. The most frequent categories indicate a focus on linear, single-player experiences. The strong presence of publishers such as Big Fish Games (approximately 400 titles) suggests that many of these games are puzzles or hidden-object titles originating from long-standing catalogues and later uploaded to Steam in bulk, often after being released on other platforms. These titles are typically republished without significant updates and no longer generate meaningful activity. This cluster represents a large mass of games that expand the catalogue but are not actively played.

4. **Mid-Tier: Niche (8.1%, 5,952 titles)**

   This cluster is characterized by intermediate ownership levels but a very high number of *n_categories* (8.6), indicating feature-rich games with multiple modes and mechanics. *Average.playtime.forever* is moderate to high, and the most common genres include Simulation and Strategy, reflecting a focus on complex systems and layered gameplay. The predominance of release years between 2020 and 2024, along with frequent categories such as Co-op and Multiplayer, suggests more technically advanced developments compared to the dataset average. The relatively low *device_support* (1.2) indicates a strong focus on Windows as the primary operating system. Overall, this cluster targets a specific audience willing to invest substantial time in complex and specialized games.

5. **Mid-Tier: Global (11.3%, 8,344 titles)**

   This cluster achieves the highest average number of *n_languages* in the dataset (11.7), while maintaining moderate values for *Estimated.owners_avg*, *Peak.CCU*, and *Average.playtime.forever*. The average *Price* ($7) and the dominant categories point toward relatively simple and linear gameplay experiences, with a

notably high presence of family sharing compared to other clusters. The strong localization effort suggests products designed for a broad and heterogeneous international audience. The concentration of release years in 2023 and 2024 aligns with current industry trends, where modern productions increasingly prioritize global distribution from launch. This cluster represents games oriented toward wide international reach through accessible and easily consumable content.

6. **Mid-Tier: Hybrid (13.2%, 9,772 titles)**

   Games in this cluster exhibit the second-highest *device_support* (2.5), following Cluster 1 (2.8), indicating strong technical compatibility. *Estimated.owners_avg* and *Average.playtime.forever* values are moderate, while the relatively low *n_categories* suggests more focused and less diversified gameplay experiences. The prominent presence of publishers such as Choice of Games indicates a strong concentration of narrative-driven titles. The limited number of *n_languages* points to less extensive localization strategies. Overall, this cluster consists of solid, stable games with good technical performance and a consistent, though not massive, user base.

7. **Top-Tier: Online (9.7%, 7,198 titles)**

   This cluster contains the highest values for several key performance indicators, including *Peak.CCU* (2,218), *Average.playtime.forever* (600 hours), average *Price* ($21), and extremely high numbers of *Total_reviews* and *Estimated.owners_avg*. The dominant presence of categories such as Online PvP, Co-op, and Multiplayer clearly identifies live-service and highly competitive online games. *Device_support* is relatively low, indicating titles primarily optimized for Windows. Moreover, the strong representation of major publishers such as Square Enix, Ubisoft, Electronic Arts, and SEGA confirms that this cluster is composed of high-budget top-tier productions. This group hosts the most active communities in the dataset and includes some of the most popular and influential titles on the platform.

## 4.5 Clusters Takeaways

Based on the analysis of the individual clusters, several dominant trends emerge that help characterize the overall structure of the Steam catalogue. First, a large marginal segment is evident, dominated by low-activity titles belonging primarily to Clusters 2 and 3. These clusters include, on the one hand, very low-priced games with minimal engagement and limited production investment, and, on the other hand, static legacy titles originating from older catalogues that have been imported into Steam without substantial updates. Together, they account for a significant share of the dataset while contributing marginally to active usage and community participation. Second, a heterogeneous mid-tier segment emerges from Clusters 4, 5, and 6, where strategic differentiation is more pronounced. Within this tier, some games compete through niche complexity and deep gameplay systems, others prioritize broad international reach through extensive localization, while a third group adopts a hybrid strategy focused on technical compatibility and stable, narrative-driven experiences. Finally, the analysis identifies two distinct top-tier segments. Cluster 1 consists of premium titles that combine high engagement with wide international distribution and long-term relevance, whereas Cluster 7 is characterized by highly active online games built around competitive or cooperative live-service models. These two top-tier clusters represent different pathways to success, one driven by broad appeal and sustained quality, and the other by intense community activity and ongoing player interaction.

# 5. Random Forest and SHAP

## 5.1 Random Forest's method

After obtaining a seven-cluster solution through k-means clustering, the analysis moved from a purely descriptive segmentation toward an explicit understanding of which variables are responsible for the separation among clusters and how they interact to define cluster membership. While k-means effectively identifies groups of similar observations, it does not provide information about feature relevance or the direction of influence of individual variables.

To address this limitation, a two-step strategy was adopted. First, a Random Forest classification model was trained using the cluster labels (*cluster_k7*) as the target variable. In this setting, the Random Forest does not serve a predictive goal in the traditional sense, but rather an explanatory one: if the clusters represent coherent and stable segments, it should be possible to accurately predict cluster membership from the observed features. Second, the Random Forest model was complemented with an approach based on SHAP (Shapley Additive Explanations), giving a clearer picture of how each cluster is actually defined.

Before training the model, I created a variable named *cluster_k7*, which assigns to each observation in the dataset an integer value from 1 to 7 identifying the cluster to which the videogame belongs. Subsequently, I selected only the numerical dimensions of the dataset in order to construct a new data frame (df_num). This step was necessary because Random Forest models require predictors that can be evaluated through numerical comparisons when constructing decision tree splits. Non-numeric variables (such as identifiers or textual fields) cannot be directly used in this process.
The new variable *cluster_k7* was then designated as the target variable of the model and converted into a factor. This transformation is required in classification settings in order to explicitly inform the algorithm that the response variable represents a set of finite categories rather than a continuous numerical quantity. Finally, the dataset was partitioned into a training and a test set using a 70%/30% split. This procedure enables

the evaluation of the model on data not used during training, improving the reliability of the performance evaluation.

## 5.2 Model training and configuration

The Random Forest was trained using a relatively large sample to ensure stability of the results. Specifically, the number of trees was set to *ntree = 500*, while the number of candidate variables considered at each split *(mtry)* was set to the square root of the total number of predictors. Variable importance computation was enabled through the *importance = TRUE* option, allowing both Mean Decrease Accuracy and Mean Decrease Gini to be extracted after training.

This configuration was chosen to balance model flexibility and robustness, minimizing overfitting while preserving the ability to capture non-linear interactions between features.

## 5.3 Model performance and validation

The trained Random Forest achieved very strong performance. The Out-of-Bag (OOB) error rate was approximately 2.65%, indicating a high degree of internal consistency. The confusion matrix provides a detailed view of classification performance by reporting the number of correctly and incorrectly classified observations for each class. High values along the main diagonal indicate correct classifications, while off-diagonal entries represent misclassifications and reveal which classes are most frequently confused.

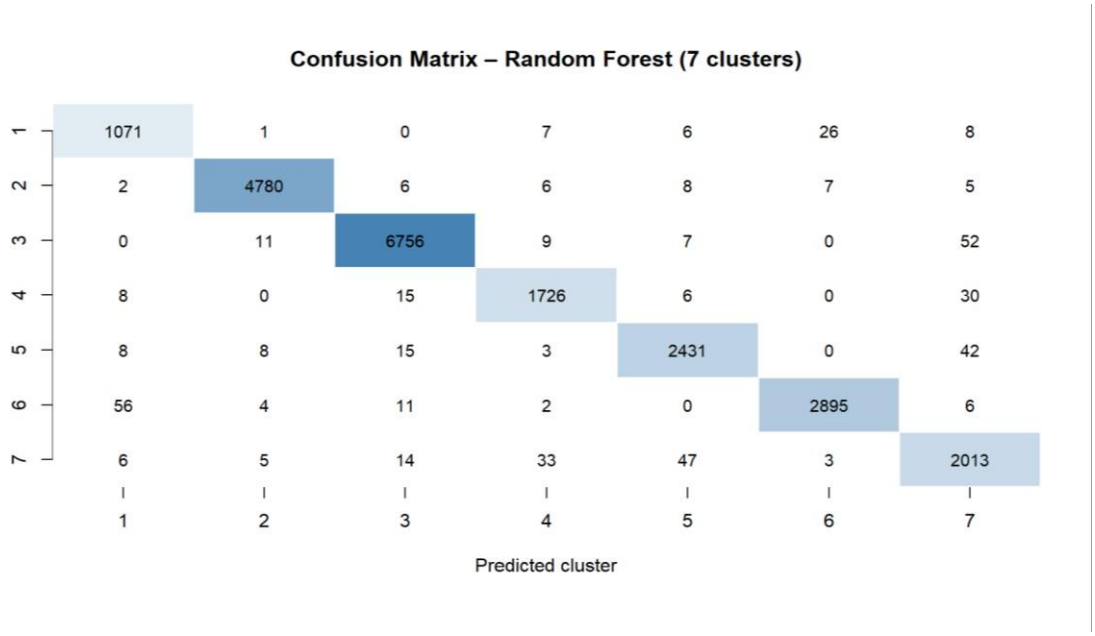


*Image 12: Confusion matrix between 7 clusters.*

Across all seven clusters, the number of misclassifications is relatively small compared to the total number of observations per class. Most errors are confined to off-diagonal cells with low counts, suggesting limited overlap between clusters. In particular, clusters 2 and 3 exhibit the highest diagonal values with minimal confusion, indicating that these segments are especially homogeneous and clearly identifiable based on the available features. Overall, the confusion matrix confirms that the seven clusters are highly separable, with only limited boundary overlap.

## 5.4 Global variable importance

To understand which variables contributed the most to cluster discrimination, variable importance was evaluated using both Mean Decrease Accuracy and Mean Decrease Gini. Both metrics converged on the same set of dominant predictors. In particular: *Price, device_support, n_languages,* and *n_categories* emerged as the strongest discriminative features, followed by engagement variables such as *Total_Reviews*, *Review_Ratio*, and *Positive* and *Negative* review counts.

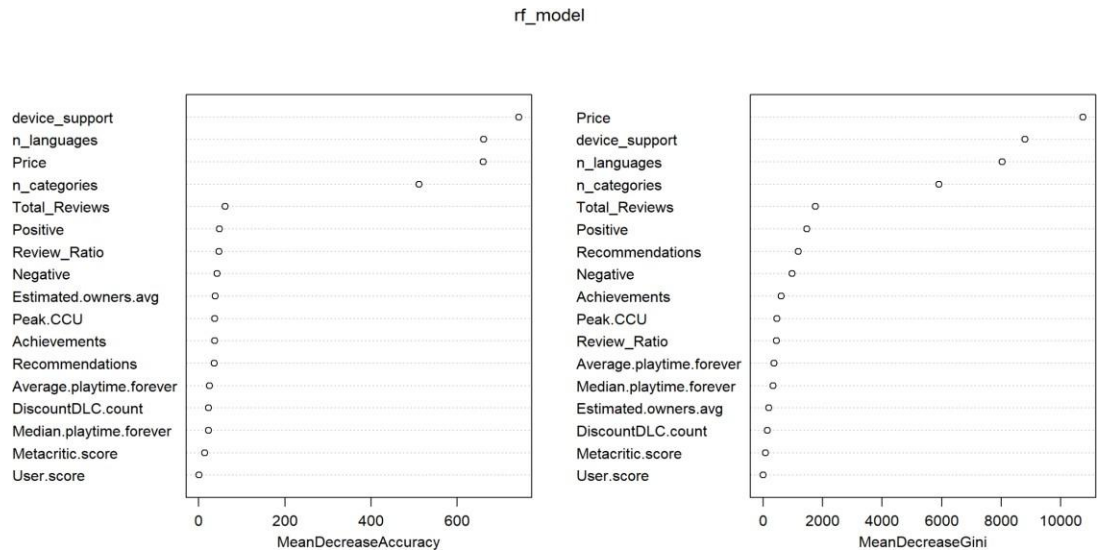


*Image 13: Mean Decrease Accuracy and Mean Decrease Gini.*

Similarly, the global variable importance, measured through Mean Decrease in Accuracy, revealed a coherent pattern. *Device_support*, *Price, n_languages*, and *n_categories* emerged as the most discriminative features, followed by engagement variables such as *Total_Reviews, Review_Ratio, Positive* and *Negative* feedback*, Estimated.owners.avg,* and *Peak.CCU*.

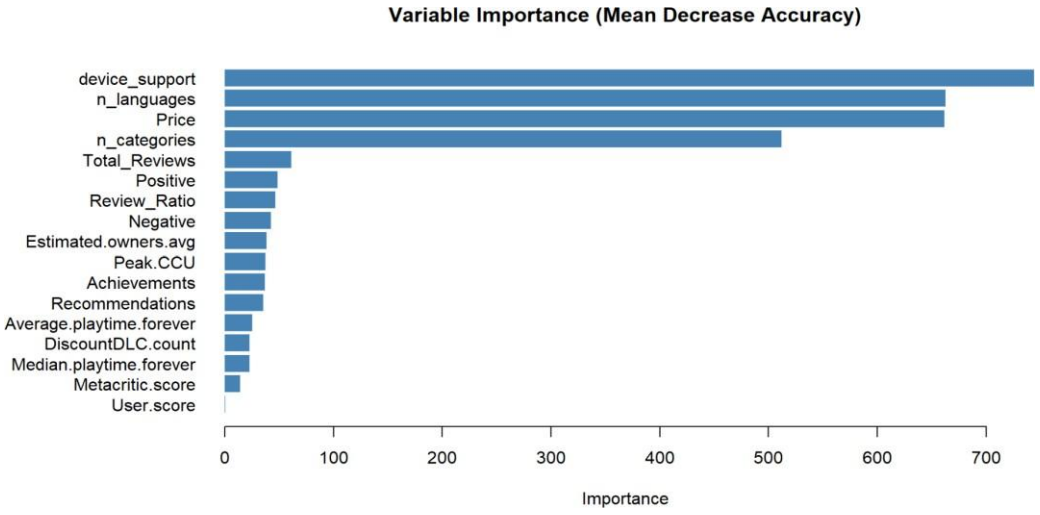


*Image 14: Global Variable Importance's graph.*

However, while this global importance ranking clarifies which variables matter overall, it does not explain how they influence cluster membership at the level of individual observations or specific clusters, and for this reason a SHAP analysis was performed.

## 5.5 SHAP Analysis: Motivation and methodological framework

To provide a transparent and detailed interpretation of the Random Forest classifier and to understand which variables drive the assignment of games to the seven identified clusters, an approach based on SHAP (Shapley Additive Explanations) was adopted.
SHAP is baseded on cooperative game theory and explains a model's prediction through the marginal impact of each feature, clarifying how individual variables increase or decrease the probability of belonging to a given cluster.
All SHAP computations were performed using the iml package and relied on the same numeric dataset (df_num) used to train the Random Forest.

## 5.6 SHAP Procedure

To carry on the analysis, two custom prediction functions were defined, each returning exclusively the predicted probability of belonging to a single cluster, specifically cluster 1 or cluster 7. These two clusters were selected because they represent the premium segments of the dataset. Since the objective of the analysis is to identify the key drivers of popularity and perceived quality in video games, it was initially focused on these premium clusters before being extended to the remaining ones. For each function, a corresponding Predictor object was constructed using the iml package, enabling the computation of SHAP values in a one-vs-rest setting.
A representative game was then randomly selected from the observations assigned to each cluster. For these observations, SHAP values were computed in order to decompose the predicted probability into marginal contributions associated with each feature. This method enables a granular look at the drivers behind each classification while remaining coherent with the Random Forest's internal logic.

## 5.7 SHAP Interpretation

### 5.7.1 Interpretation of SHAP results: Cluster 1 (Top Tier – Premium)

For the representative game in cluster 1, the SHAP values reveal a distinct explanatory pattern, centered on accessibility and international reach. The strongest positive

contributions come from *n_ categories*, *n_languages* and *device_support* indicating a high degree of multi-platform compatibility and game diversity on a global scale.

Engagement variables, including *Total_Reviews* and *Positive* reviews also play an important role, signaling a large and stable user base, although their influence is less pronounced than in cluster 7. Other features such as *Achievements*, *Average.playtime.forever*, *Price*, and *Review_Ratio* contributed positively but with a more marginal impact.

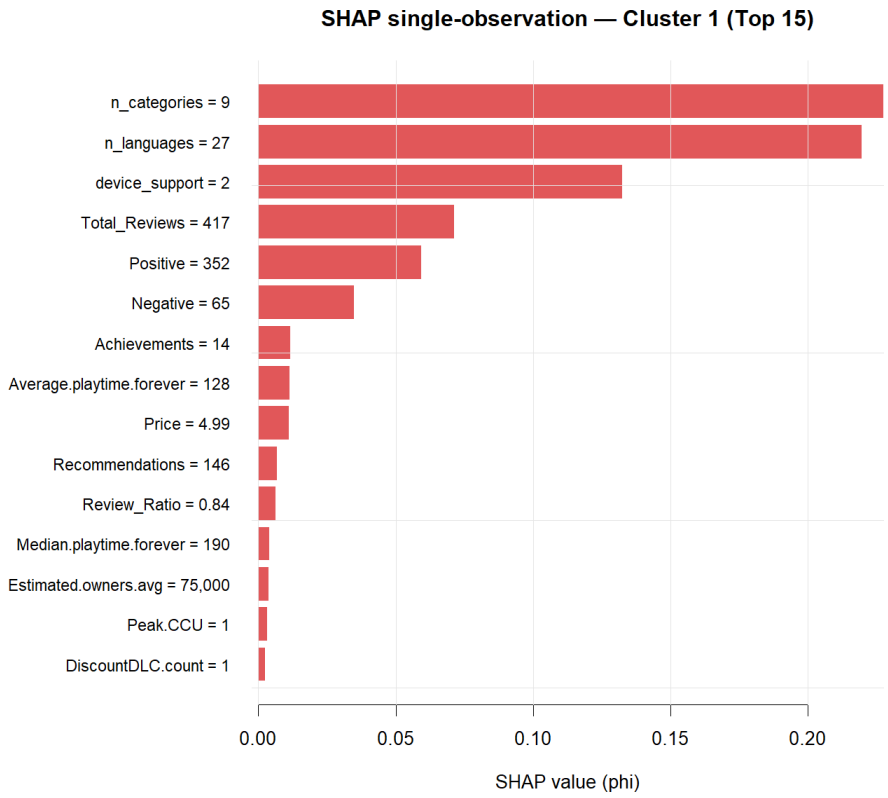


*Image 15: SHAP single observation for K1.*

### 5.7.2 Interpretation of SHAP results: Cluster 7 (Top Tier – Online)

Regarding cluster 7, the SHAP decomposition reveals that this prediction is primarily driven by variables associated with engagement intensity and commercial scale.

The variable *Total_Reviews* emerges as a strong positive contributor, along with *Positive* and *Recommendations*, reflecting the high engagement of these titles, indicating the presence of a very large and highly active community. This evidence is reinforced by the strong contribution of *n_languages*, which reflects the global nature of these titles. The *Price* also contributes positively, highlighting the premium positioning

of these games. Additional variables such as *n_categories* and *Average.playtime.forever* provide further positive support, but with smaller marginal effects.

In contrast, *device_support* contributes negatively to the classification, indicating limited cross-platform compatibility. This pattern is consistent with games optimized primarily for Windows ecosystems, a common characteristic of competitive online or live-service titles. Also *Peak CCU* and *DiscountDLC.count* contribute negatively to the classification, but with marginal effects. Overall, the SHAP explanation identifies cluster 7 as a segment of top-tier online games characterized by extreme engagement intensity and very large communities.

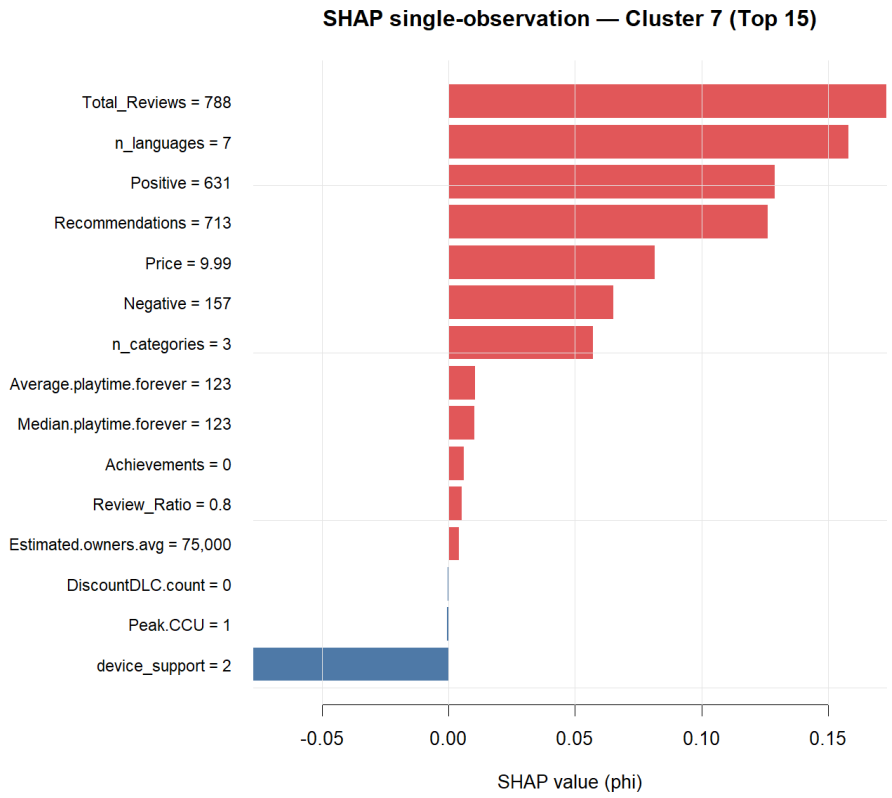


*Image 21: SHAP single observation for K7.*

### 5.7.3 Comparative interpretation of clusters 1 and 7

Comparing the SHAP explanations of clusters 1 and 7 reveals a clear structural distinction between two different forms of top-tier games. While both clusters occupy the highest segment of the market, cluster 1 represents globally distributed premium titles whose success is driven by accessibility, localization, and sustained engagement.

Cluster 7, by contrast, represents premium online or live-service games, where classification is driven by very high prices, remarkable levels of user engagement, massive communities, and complex online-oriented structures, often at the expense of cross-platform compatibility.

### 5.7.4 Multi-observation SHAP

To ensure robustness and avoid excessive weight on individual observations, the SHAP analysis was extended to a multi-observation setting, so the values were computed for a random sample of 50 games from cluster 1 and 50 games from cluster 7, and their average SHAP values were calculated to derive an aggregate measure of feature importance. This extended analysis confirms the initial findings: for cluster 1, accessibility variables such as *device_support* and *n_languages* dominate, followed by engagement metrics, while *Price* plays a relatively minor role.

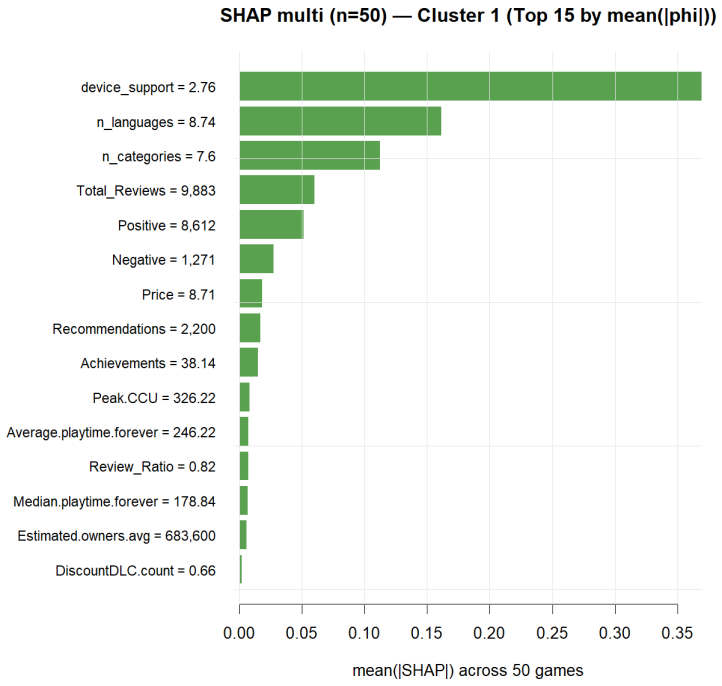


*Image 22: SHAP multiple observation for K1.*

For cluster 7, *Price* and engagement indicators emerge as the dominant drivers, showing a more stable structure compared to the single-observation analysis. While the individual case was primarily characterized by *Total_Reviews*, the aggregate measure highlights *Price* as the most influential factor, shifting *Total_Reviews* as the second

strongest contributor, reinforcing the premium nature of this segment across the entire sample. Right after the first two variables, the strongest contributors are engagement values such as *n_languages*, *Positive* reviews, and *Recommendation.* As for *device_support*, while it was a primary negative driver for the single game, the aggregate view shows that it has positive contributions when considering the cluster as a whole.

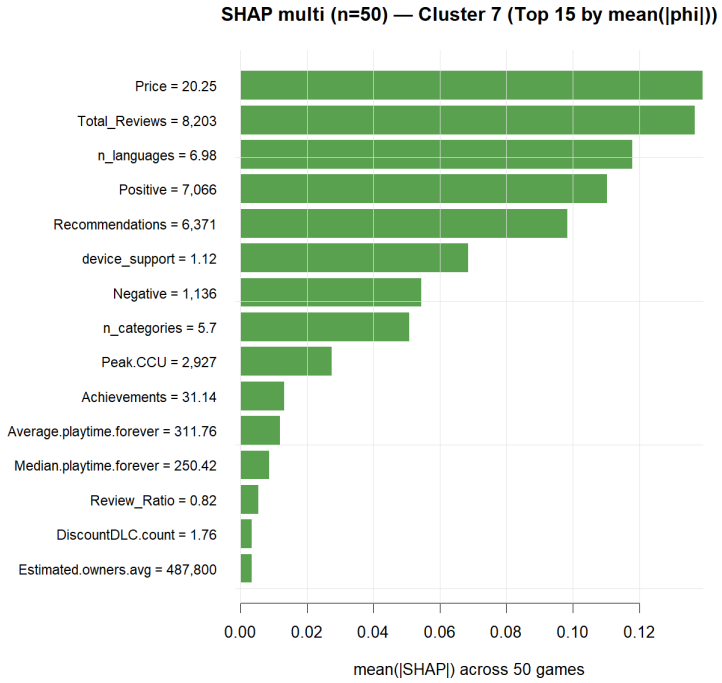


*Image 28: SHAP multiple observation for K7.*

These results further reinforce the interpretation of cluster 1 as a globally accessible premium segment and cluster 7 as a high-intensity, engagement-driven online segment.

### 5.8 Extension of the Analysis to All Seven Clusters

The SHAP framework was extended to the multi-class setting using a one-vs-rest probability formulation. In parallel with the previous SHAP analysis, the study begins by selecting a single representative observation for each cluster, to highlight which variables drive the assignment of a specific data point to its respective cluster most significantly. Some clusters are almost entirely explained by a single dominant variable. Cluster 4, for instance, is mostly driven by *n_categories*, which alone accounts for the vast majority of the predicted probability.

Similarly, cluster 5 is primarily explained by *n_languages*, with small contributions from *device_support*.

Cluster 6 is dominated by *device_support*, showing that technical accessibility plays a defining role in these segments.

Other clusters exhibit more balanced explanations: cluster 2 is largely driven by low *Price*, suggesting a budget-oriented segment with limited localization and structural complexity. It also has small negative contributions from *Total_Reviews*, highlighting the low engagement between these titles.

Cluster 3 shows a combination of moderate *Price*, limited localization explained by *n_languages*, and basic *device_support*, consistent with low or mid-tier identification.

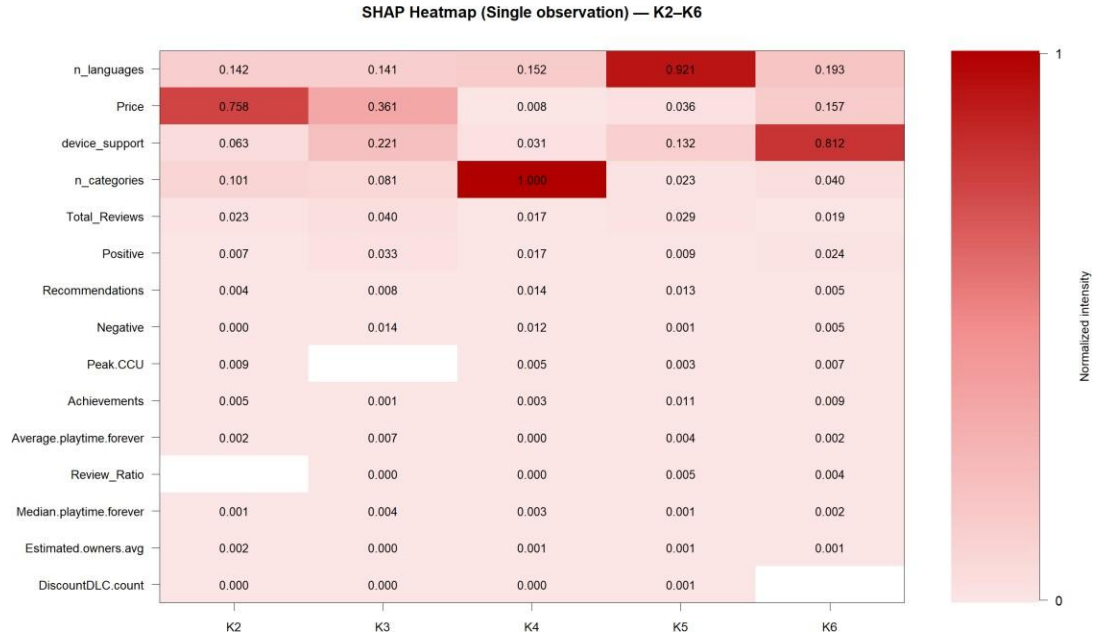


*Image 29: SHAP combined graph of single observation from clusters 2 to 6.*

### 5.9 Multi-observation SHAP to All Seven Clusters

Later on the SHAP values were computed for a random sample of fifty games per cluster. The aggregated results strongly confirm the patterns observed in the single-observation analysis. Clusters 4, 5, and 6 remain dominated by a single structural

variable, respectively *n_categories*, *n_languages*, and *device_support,* indicating that these clusters are defined by clear and simple organizing principles.

Cluster 2 continues to be primarily driven by the *Price*, while cluster 3 shows a more balanced but still relatively simple explanatory structure.

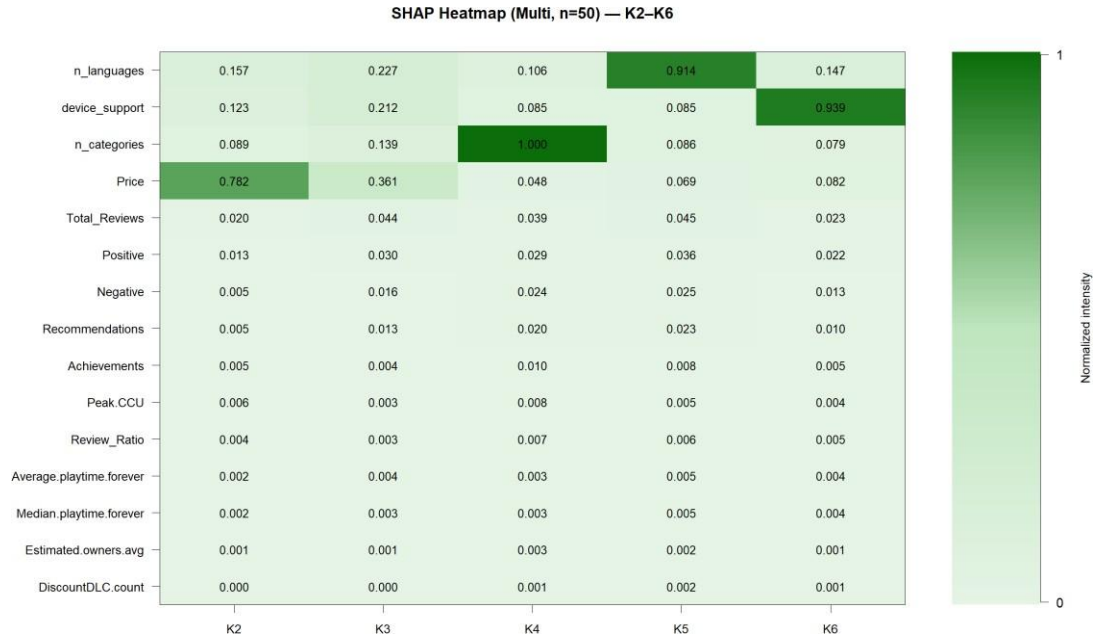


*Image 30: SHAP combined graph of multiple observation from clusters 2 to 6.*

### 5.10 Overall conclusion

Taken together, the combined Random Forest and SHAP analysis confirms that the seven-cluster solution is not only statistically strong but also highly interpretable. The clusters are clearly separable, and their separation is driven by meaningful and consistent dimensions related to technical accessibility, internationalization and community engagement. Importantly, SHAP reveals that different clusters rely on fundamentally different explanatory mechanisms, ranging from single dominant structural features in mid and low tiers, to multidimensional profiles in premium clusters.

# 6. Embeddings

The study then incorporated a specific analysis of textual information: within the dataset, the variable *About.the.game* provides a detailed description of each videogame, typically intended to present its features, themes, and overall experience to potential players. The presence of this additional textual dimension makes it possible to conduct more detailed analyses of videogame descriptions, using linguistic information to enrich the cluster structure and to provide complementary insights for the interpretation of the dataset.

To address this objective, two different types of textual representations were used: the first approach relies on embeddings derived from a bag-of-words framework using TF–IDF weighting. In this context, TF–IDF (Term Frequency–Inverse Document Frequency) represents each word by assigning higher weights to terms that are frequent within a document but relatively rare across the corpus, emphasizing words that are potentially more informative and discriminative. The second approach makes use of neural contextual embeddings based on Sentence-BERT (SBERT), a model designed to convert text into vectors that preserve the actual meaning and context of the language used. By combining these two approaches, the analysis first focuses on enriching the semantic and narrative characterization of the clusters and then extends to the extraction of more general insights about the dataset.

## 6.1 TF-IDF

The variable *About.the.game* contains natural-language descriptions of the videogames included in the dataset. These descriptions are often relatively long and strongly promotional in nature, and they typically include punctuation, special characters, and a large number of highly frequent words such as game, player, or experience. Before any quantitative analysis could be performed, it was therefore necessary to apply a systematic process of text preparation in order to obtain a more homogeneous and analyzable representation of the textual content.

The preprocessing phase was designed to reduce noise and standardize the text while preserving its semantic content. First, the descriptions were normalized by converting all characters to lowercase and by applying a uniform treatment of punctuation and symbols, including the removal of non-alphabetic characters and the standardization of separators, in order to avoid artificial differences between otherwise identical terms.
The text was then cleaned by removing elements that are not informative from a semantic perspective, such as special characters or residual formatting artifacts. Each description was subsequently tokenized, meaning that the text was split into a sequence of individual words, or tokens, which constitute the basic units of analysis in a bag-of-words representation.

In order to focus the analysis on meaningful lexical content, common stopwords were removed: these include highly frequent functional words such as *the*, *and*, *of*, or *to*, which play an important grammatical role but do not carry useful semantic information for distinguishing between documents. In addition, frequency-based filters were applied to further refine the vocabulary: on one hand, extremely rare words that appear in only a very small number of games were excluded, as they are more likely to introduce noise than to reflect systematic patterns. On the other hand, overly common words that appear in almost all descriptions were also removed, since they are not discriminative and tend to reflect the standardized marketing language used across the Steam platform. After this preprocessing phase, the textual data were used to construct classical text embeddings based on a bag-of-words framework. In this first embedding approach, the cleaned descriptions were transformed into a term–document matrix, weighted using TF–IDF in order to emphasize terms that are frequent within specific documents but relatively rare across the corpus. Because this representation is inherently high-dimensional, a dimensionality reduction technique was applied to obtain a compact embedding space. Through this process, each game description was mapped to a numerical vector of reduced dimensionality, on the order of one hundred dimensions, resulting in a set of embedding coordinates (emb_1, emb_2, …,).

These embeddings were then projected into a two-dimensional space using UMAP, with points colored according to their cluster assignment.

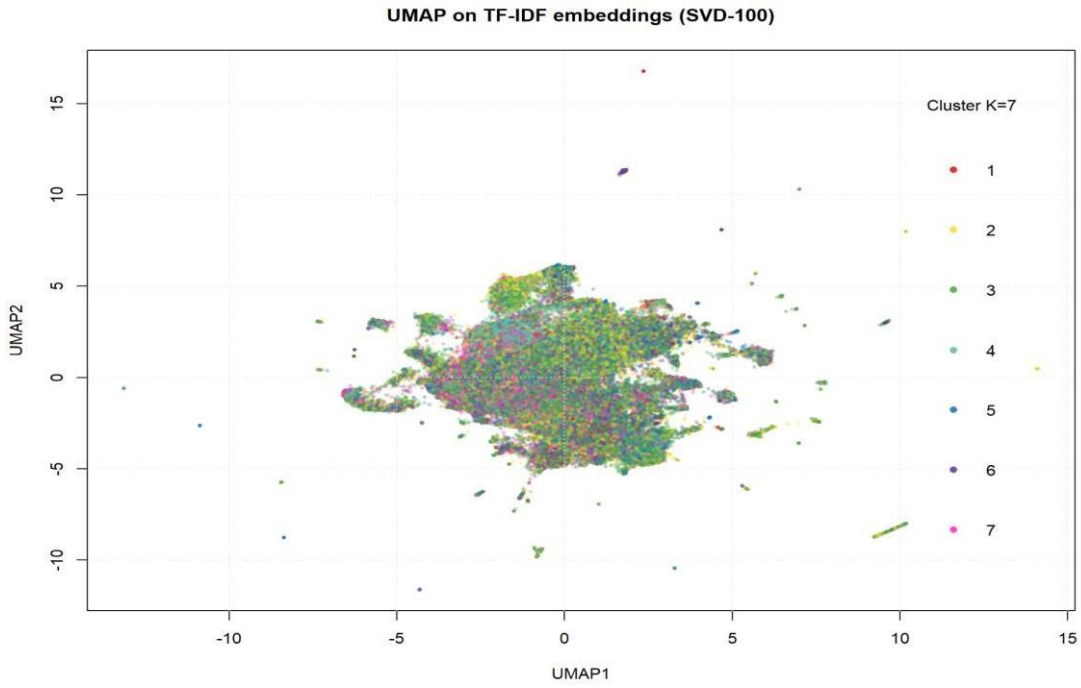


*Image 31: UMAP of embeddings divided by clusters.*

The purpose of this visualization was to assess whether the structure of the embedding space reflected the previously identified cluster structure. The resulting projection showed that the points corresponding to different clusters were strongly mixed, with no clear separation between the seven groups. The embedding space was characterized by a dense central cloud in which clusters largely overlapped, suggesting that the textual embeddings did not naturally organize themselves according to the numerical clustering.

In addition to the visual exploration, the classical embeddings were also evaluated in combination with numerical variables through a supervised model. A hybrid Random Forest classifier was constructed by including both the numerical features and the embedding dimensions as predictors of cluster membership. The results of this model further confirmed the limited contribution of the textual embeddings. The out-of-bag error was similar to that of the numerical model, remaining at approximately five percent. Moreover, the feature importance analysis showed that the most influential variables were still the same numerical indicators identified earlier, such as *Price*, *device_support*, *n_languages*, *n_categories*, *Total_Reviews*, and the balance between

*Positive* and *Negative* reviews. By contrast, the embedding dimensions consistently appeared at the bottom of the importance ranking, with marginal contribution to the model's predictive performance.

Taken together, these results indicate that classical text embeddings derived from TF-IDF and dimensionality reduction do not capture textual differences that are coherent with the cluster structure.

## 6.2 Sentence-BERT

To explore whether more advanced semantic representations could capture information not detected by classical bag-of-words approaches, a second embedding strategy based on neural contextual models was implemented. Specifically, Sentence-BERT (SBERT) was employed to generate sentence-level embeddings from the About the game descriptions. SBERT uses a transformer framework to encode entire texts into vectors, ensuring that both the literal meaning and the context remain intact.

The model used in this analysis was all-MiniLM-L6-v2, which was loaded directly within the R environment through the reticulate interface. For each videogame description, the model produced a vector of 384 numerical dimensions, resulting in a representation of the form *emb_1, emb_2*, …, *emb_384*. These embeddings were then combined with the existing numerical variables to construct a dataset structured as *AppID*, *cluster_k7*, and the corresponding SBERT embedding dimensions.

As in the case of classical embeddings, the SBERT representations were first explored through dimensionality reduction and visualization. The UMAP visualization did not reveal any meaningful improvement compared to the TF–IDF-based embeddings. The clusters remained strongly overlapping, with no clearly separated regions or visually distinct structures that could be associated with specific clusters. Overall, the UMAP projection of SBERT embeddings appeared almost indistinguishable from that obtained using classical embeddings, suggesting that the richer semantic modeling provided by SBERT did not translate into a clearer alignment with the numerical clustering.

In addition to the exploratory visualization, the SBERT embeddings were also evaluated in combination with numerical variables through a hybrid Random Forest model, where cluster membership was predicted using both the numerical features and the SBERT embedding dimensions. The results of this model further reinforced the conclusions drawn from the visualization. Rather than improving predictive performance, the inclusion of SBERT embeddings led to a deterioration in accuracy, with the out-of-bag error increasing to approximately 11%, compared to about 5% for the TF-IDF model.
An inspection of feature importance confirmed that the most influential predictors remained the numerical variables, including *Price*, *device_support*, *n_languages*, *n_categories*, *Total_Reviews*, the balance between *Positive* and *Negative* reviews, *Review_Ratio*, and *Peak.CCU*. By contrast, the SBERT embedding dimensions appeared at the bottom of the importance ranking, contributing with little or no useful information to the model.

Taken together, these results indicate that even a neural embedding model such as SBERT fails to identify semantic patterns that are informative for distinguishing between the clusters. The About the game descriptions on Steam appear to be highly similar across products, largely reflecting standardized marketing language and recurring descriptive formulas. Moreover, the textual content shows limited correlation with key economic and technical variables such as price, ownership, and review variables, which instead play a central role in defining the cluster structure. As a consequence, the neural embeddings do not provide additional information that would meaningfully support cluster separation or prediction in this context.

## 6.3 T-SNE Visualization

Afterwards, I performed a Principal Component Analysis (PCA) on the embedding variables to reduce their dimensionality, using the resulting components to guide the t-SNE visualization. The projection revealed a large, dense central structure surrounded by several isolated points situated far from the rest of the observations.

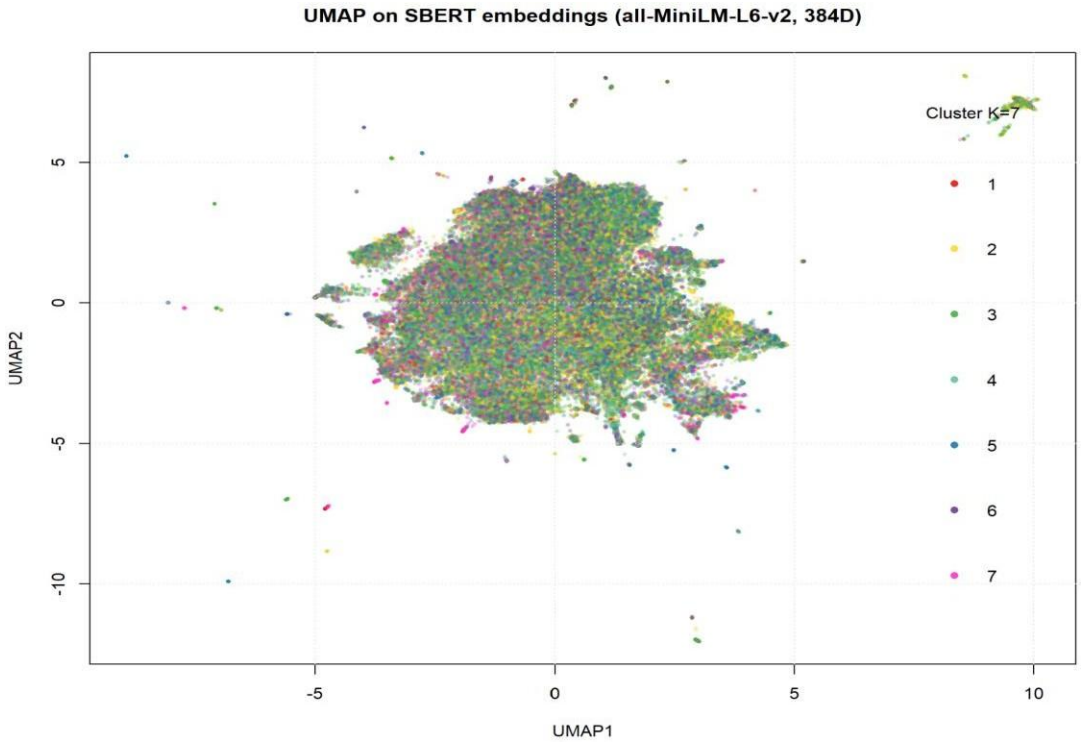


*Image 32: t-SNE visualization of embeddings divided by cluster.*

Given the limited separation in the initial results, the outliers were removed to achieve a more compact visualization that might highlight the semantic distribution and potential distinctness of the clusters more clearly. However, after generating the plot without these extreme observations, it became evident that the overall structure was neither altered nor improved. This lack of change confirms that, from a semantic perspective, the clusters remain largely overlapping.

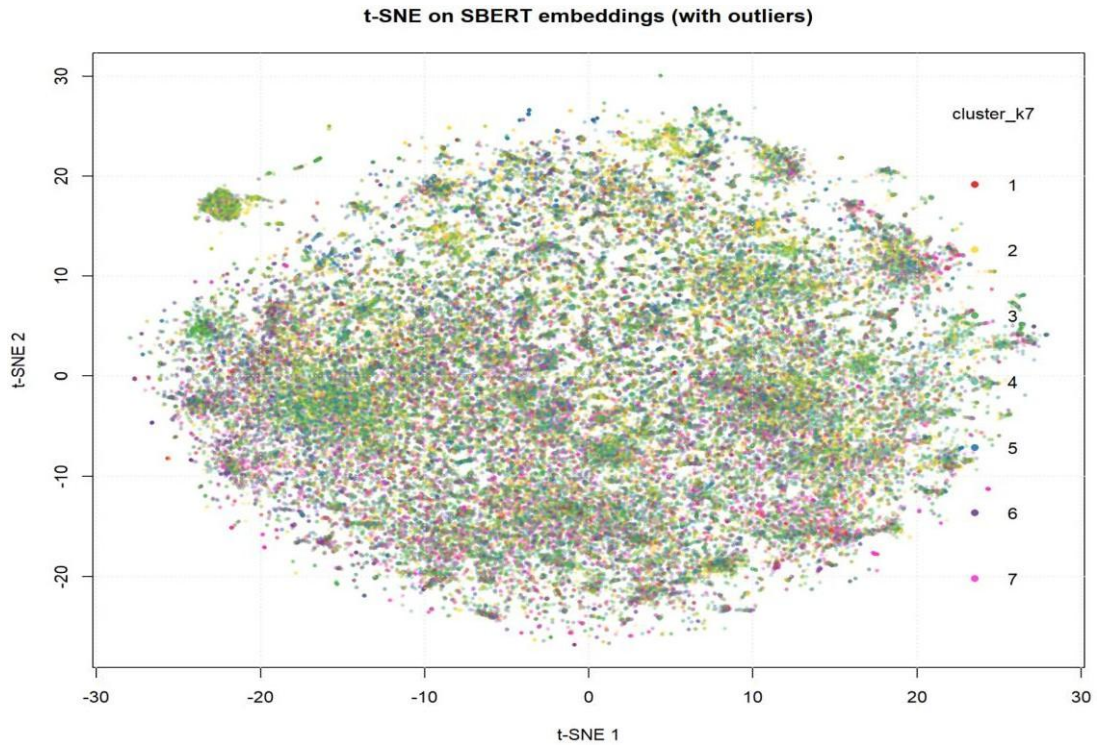


*Image 33: t-SNE visualization of embeddings divided by cluster without outliers.*

## 6.4 Summary of Embeddings Results

The empirical evidence obtained from the embedding analyses consistently points in the same direction across all the approaches that were tested. Both classical textual representations based on TF–IDF and dimensionality reduction, and neural contextual embeddings derived from Sentence-BERT, lead to highly similar outcomes. In all cases, the visual exploration of the embedding spaces through UMAP projections reveals a strong overlap between clusters, with no clearly distinguishable regions corresponding to the seven groups identified in the numerical analysis, and this result is also confirmed by the t-SNE visualization. This suggests that the semantic structure captured by the textual embeddings does not align with the cluster structure derived from numerical variables. Furthermore, the evaluation of textual embeddings in combination with numerical features confirms the limited contribution of the textual component. Even when a modern contextual model such as SBERT is employed, the inclusion of embedding-based representations does not improve the separation or the identification of the clusters. Variables like *Price*, *Total_Reviews*, *Estimated.owners_avg*,

*devices_support*, *n_languages*, and *n_categories* remain the primary drivers of the clustering structure, while the textual embeddings provide little or no additional explanatory power.

## 6.5 Unigrams VS Bigrams

### 6.5.1 Unigrams Analysis

At this point, having established that the embeddings do not enhance cluster separability, the analysis shifts from a predictive to a purely descriptive approach. The focus is no longer on improving classification, but rather on characterizing each segment through its most frequent linguistic patterns to extract more specific and interpretable insights about the individual groups. The *About.the.game* descriptions were treated as a corpus, and a unigram representation was first constructed after preprocessing and stemming, producing a set of lexical tokens comparable across clusters. In order to reduce the influence of repetitive language that tends to appear across most clusters, a dynamic stoplist was introduced: only terms with a minimum within-cluster frequency of 25 were retained, and then terms appearing in at least 80% of the clusters (in 6/7 or 7/7 clusters) were removed, with the procedure selecting the 400 most cross-cluster pervasive stems (DYN_STOP_K = 400). This filtering step substantially reduced the dominance of standardized vocabulary and allowed the subsequent cluster-level TF–IDF analysis to focus on more distinctive terms.

After applying the dynamic stoplist, TF–IDF was computed at the cluster level to extract representative keywords: each cluster was treated as a document, and TF–IDF emphasized words that were frequent within a given cluster while relatively uncommon across the others. Unique terms, defined as stems occurring in only one cluster, were extracted providing discriminative signals and revealed strong differences in lexical specialization across groups. The top terms obtained through this procedure were often highly specific entities such as proper nouns, franchise references, or specific terms referring to the catalogue. The number of unique unigrams varied substantially from cluster to cluster: cluster 1 contained 17 unique stems, cluster 2 contained 69, cluster 3 contained 1,110, cluster 4 contained 21, cluster 5 contained 27, cluster 6 contained 90,

and cluster 7 contained 171. This result is particularly informative because it indicates that cluster 3 is linguistically much more specialized than the other clusters, which makes sense since cluster 3 is the largest cluster between them all; clusters 7 and 6 also exhibit a strong distinctive vocabulary. In qualitative terms, cluster 7 appeared to be strongly driven by franchise references, while cluster 6 contained mostly terms related to identity (stems connected to terms such as gay, asexual, nonbinary). Cluster 5 showed signals related to software, tools or terms related to the usability and interface of the game (e.g., taskbar or powerdirector), while cluster 4 displayed vocabulary connected to competitive or multiplayer mechanics (e.g., rollback or knockout). Cluster 1, despite having a relatively small number of unique terms, still exhibited highly specific lexical cues, often linked to particular titles or series.

**Unique Unigrams — Top keywords per cluster (df==1)**

| | Cluster 1 | Cluster 2 | Cluster 3 | Cluster 4 | Cluster 5 | Cluster 6 | Cluster 7 |
|---|---|---|---|---|---|---|---|
| Word 1 | deponia | bitcoin | screensav | bros | angela | gay | lego |
| Word 2 | nomine | lawnmow | wukong | dodgebal | hercul | bi | disney |
| Word 3 | chelladin | ppi | nezha | dunk | powerdirector | hiroki | batman |
| Word 4 | rufus | spinner | cao | rollback | arab | asexu | gundam |
| Word 5 | trine | digipen | aldorlea | sumo | taskbar | sokpop | br |
| Word 6 | agf | poke | deriv | knockout | susan | nonbinari | liveri |
| Word 7 | shogun | megapixel | outlin | netcod | ambul | grendel | kof |
| Word 8 | geek | ape | healer | snowmen | equiti | gpl | creed |
| Word 9 | shadowrun | horac | itsuki | darius | panoram | putt | ys |
| Word 10 | xl | whack | tarot | arcana | photodirector | viola | motogp |

*Image 34: Top 10 unique unigrams per cluster.*

Finally, to quantify how similar clusters were from a lexical standpoint, textual overlap was measured using the Jaccard index, computed between the sets of stems associated with each cluster after filtering. When focusing on unigram representations, the overlap between clusters was generally high: the off-diagonal Jaccard similarity values frequently ranged approximately between 0.43 and 0.78.

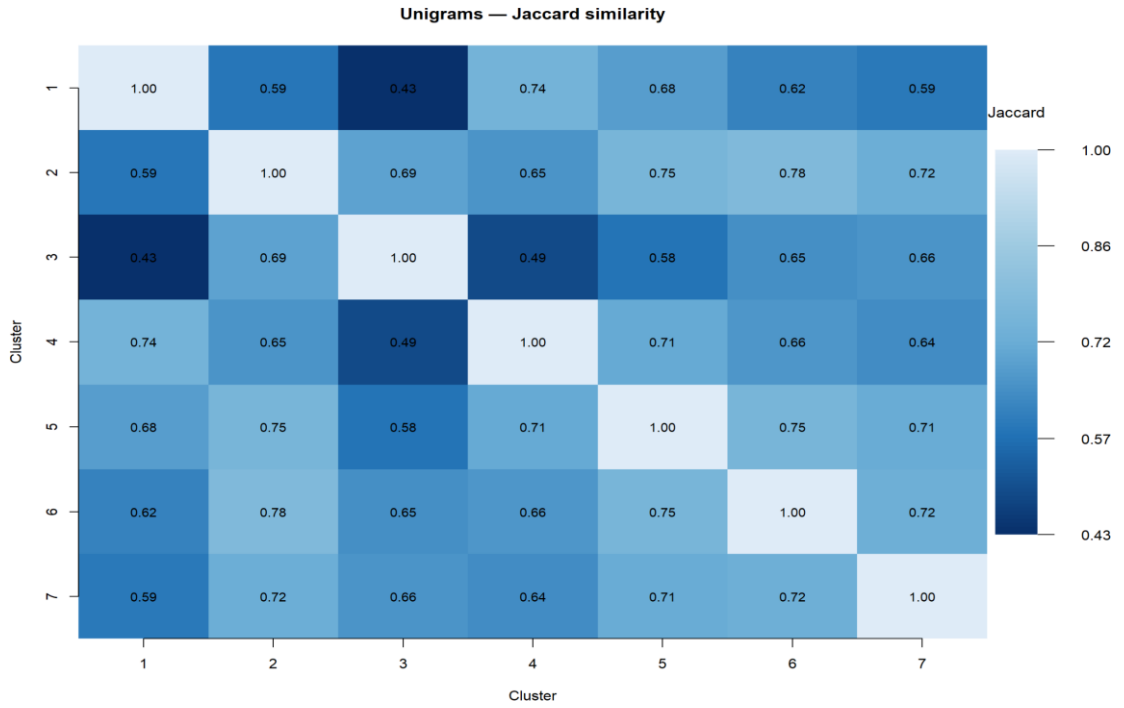


*Image 35: Jaccard's index for unigrams.*

This provides a clear quantitative confirmation that, at the level of individual words, Steam descriptions remain substantially overlapping across clusters, reflecting the platform's standardized descriptive style.

### 6.5.2 Bigrams Analysis

After analyzing the textual content at the level of individual words, the analysis was extended to bigrams in order to capture patterns at the sentence level and recurrent linguistic structures that are not observable when considering single tokens in isolation. While unigram representations primarily reflect the presence or absence of specific terms, bigrams allow the analysis to focus on how words are combined, providing a more narrative and contextual view of how videogames are described within each cluster.

Bigram extraction was performed by tokenizing the *About.the.game* descriptions into recurring word pairings. The extracted bigrams underwent the same refinement procedures previously applied to unigrams, but adapted to operate at the phrase level.

Each bigram was converted to lowercase, non-alphabetic characters, numbers, and underscores were removed, and the two component words were separated and processed individually. Then the stemming was applied separately to each word before reconstructing the final stemmed bigram representation.

As with the unigram analysis, a dynamic stoplist was introduced to eliminate bigrams that were too widely shared across clusters and therefore uninformative. Bigrams were first filtered based on a minimum frequency threshold within clusters, and then ranked according to the number of clusters in which they appeared. Bigrams occurring in at least 80% of the clusters were removed, with the procedure excluding a total of 877 highly pervasive bigrams.

Following dynamic filtering, TF–IDF was computed at the cluster level for the remaining bigrams. As in the unigram case, each cluster was treated as a document composed of all its associated bigrams, and TF–IDF was used to highlight phrase-level patterns that were frequent within a given cluster but relatively rare across the others. The resulting cluster-specific bigram keywords were notably more interpretable and descriptive than their unigram counterparts, as they often corresponded to recognizable narrative formulas, franchise references, or specific game features. For instance, cluster 1 was characterized by bigrams strongly linked to specific series or titles. Cluster 3 displayed bigrams related to editions and product variants, including phrases connected to standard versions, collector editions, or release formats. Cluster 6 exhibited bigrams reflecting themes related interactive narrative structures, while cluster 7 was dominated by expressions associated with established intellectual properties such as star wars, resident evil or metal gear. Overall, the bigram-based TF–IDF analysis provided a much clearer linguistic characterization of the clusters than unigram keywords alone.

The unique bigrams mirrored the unigram distributions and revealed even stronger differences across groups. The number of unique bigrams varied substantially by cluster: cluster 1 contained 17 unique bigrams, cluster 2 contained 245, cluster 3 contained 3829, cluster 4 contained 199, cluster 5 contained 84, cluster 6 contained 105, and cluster 7 contained 208.

**Unique Bigrams — Top 10 per cluster (df==1)**

| | Cluster 1 | Cluster 2 | Cluster 3 | Cluster 4 | Cluster 5 | Cluster 6 | Cluster 7 |
|---|---|---|---|---|---|---|---|
| Word 1 | tomb raider | hd qualiti | releas full | fun friend | pretti girl | entir text | star war |
| Word 2 | shovel knight | slide puzzl | standard version | multiplay friend | hidden cat | power imagin | call duti |
| Word 3 | lara s | student project | edit releas | screen multiplay | soccer manag | unstopp power | fan favorit |
| Word 4 | raider definit | o o | special collector | mode friend | complet town | vast unstopp | mega man |
| Word 5 | shadow tomb | classic card | exclus extra | support local | french german | without graphic | resid evil |
| Word 6 | phone tablet | graphic simpl | find standard | mode multiplay | lost land | fuel vast | mobil suit |
| Word 7 | perfect team | level relax | full exclus | local coop | video edit | effect fuel | metal gear |
| Word 8 | rusti lake | wasd movement | extra won | fight friend | footbal manag | s entir | time ever |
| Word 9 | english german | develop develop | version collector | competit mode | swipe swipe | base without | assassin s |
| Word 10 | object charact | develop student | specif genr | friend use | emili s | word interact | ninja gaiden |

*Image 36: Top 10 unique bigrams per cluster.*

Finally, textual overlap between clusters was quantified using the Jaccard index computed on the sets of filtered bigrams associated with each cluster. The resulting heatmap revealed a marked reduction in overlap compared to the unigram analysis. The Jaccard values outside the diagonal were generally low, typically ranging between approximately 0.01 and 0.32 at most. For example, overlap values between cluster 1 and clusters 2 or 3 were close to 0.01, while higher but still moderate overlap was observed between clusters such as cluster 2 and cluster 3 or cluster 2 and cluster 6, with values around 0.31–0.32. Many other cluster pairs exhibited overlap values in the range of 0.03 to 0.27. This results show that while clusters share a substantial amount of vocabulary at the unigram level, they become much more clearly differentiated when language is analyzed in terms of phrase-level patterns.

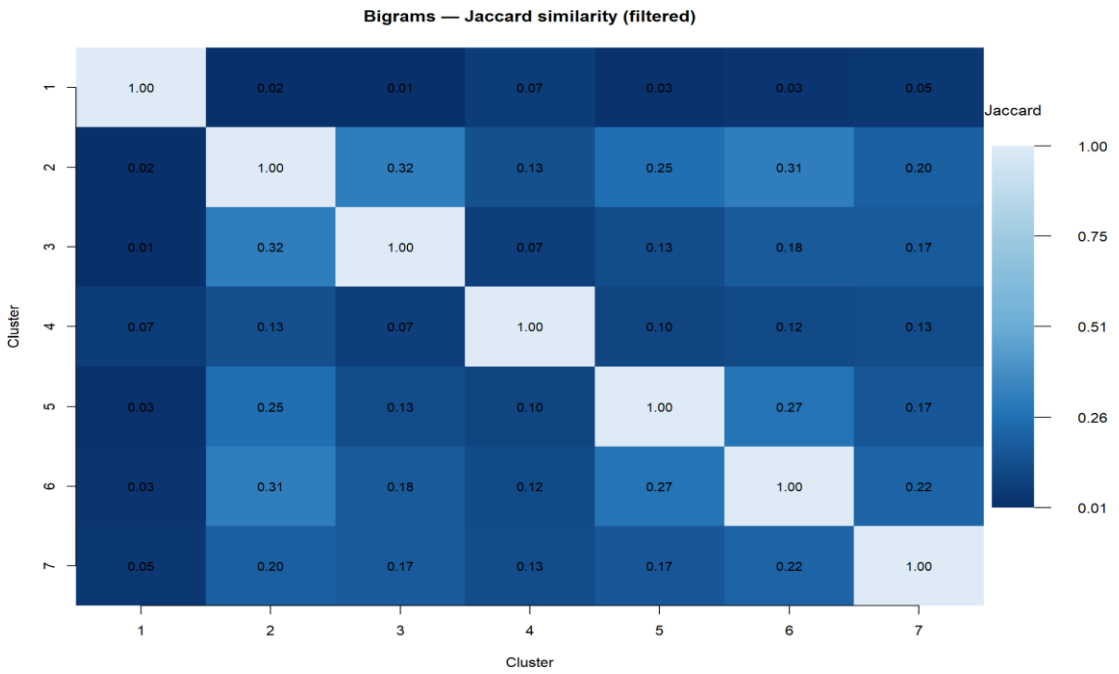


*Image 37: Jaccard's index for bigrams.*

Taken together, the bigram-based analysis confirms and refines the insights obtained from unigrams. At the level of single words, the language of Steam descriptions remains largely standardized and promotional, leading to substantial lexical overlap across clusters. At the level of bigrams, the clusters exhibit markedly lower overlap and more distinctive linguistic signatures, indicating that differences between groups are better captured by recurring phrase structures and narrative patterns than by isolated terms.

## 6.6 Consumer Profiles Derived from Game Clusters

Once the textual analysis was concluded with the aim of characterizing the previously formed clusters, a set of consumer profiles was derived by integrating the general insights from the cluster division with the specific findings of the bigram analysis. The resulting consumer profiles do not correspond to empirically observed user segments, but represent interpretative representations derived from the previously conducted analysis. Each profile emerges from the structural and economic characteristics shared by the games belonging to a given cluster and reflects a plausible mode of consumption inferred from these properties. Following there's a description of the seven consumer profiles:

1. **Collector Gamer**

   The textual patterns associated with this cluster are dominated by the titles of the games themselves, indicating that product identity functions as a primary semantic indicator. Rather than relying on generic genre descriptors, these games are presented as distinctive and self-contained experiences. The frequent occurrence of labels such as “Complete”, “Definitive”, and “Extended” points to a strong emphasis on completeness and editorial care. Notably, references to competitive mechanics, rankings, or multiplayer dynamics are largely absent, reinforcing the idea of a consumption model oriented toward immersion and narrative depth.

   This linguistic structure aligns closely with the numerical characteristics of the cluster. Games in this group tend to have very high *Price* and levels of engagement, as reflected by *Estimated.owners_avg* , *Total_Reviews* and *Peak.CCU*. At the same time, they are characterized by vast technical support, including high values of *n_languages* coverage and *device_support,* which facilitates international diffusion and sustained use over time.

   The resulting profile is that of highly engaged consumers, attracted to complete and polished games with a strong narrative imprint and recognizable franchises. They invest time and resources in premium experiences, favor immersion and longevity, and seek distinctive, high-quality cultural products.

2. **Snack Gamer**

   The language characterizing this cluster is frequently technical and functional, suggesting a conception of games as experimental object rather than as narrative or identity-driven products. Many descriptions include references to resolution, image size, and visual parameters (“ppi”, “megapixels”, “hd quality”, “graphic simple” indicates a primary focus on the technological and instrumental aspects of the experience. This points to titles that often function as prototypes, exercises, or showcases of technical skills rather than fully developed entertainment experiences. Furthermore, the emphasis on terms such as “level relax,” “classic card,” and “slide puzzle” suggests a low barrier to entry, with products designed for immediate and exploratory consumption

From a quantitative perspective, this cluster is associated with extremely low levels of engagement. Metrics such as *Average.playtime.forever*, *Peak.CCU*, and *Total_Reviews* remain minimal, and *Price* is typically very low or entirely absent. These patterns indicate products that are easily accessible, quickly consumable, and easily replaceable.

The Snack Gamer therefore emerges as an opportunistic and low-involvement consumer, driven primarily by ease of access and free or cheap titles. The value of the product resides in immediate availability and the consumer doesn't look for identity ties, taking an experimental and transitory approach to gaming.

**3. Shelf Gamer**

The linguistic profile of this cluster is strongly oriented toward storytelling, symbolic content, and editorial enhancement of the gaming experience. On one hand, the recurring presence of references to characters, myths, and cultural works, often of Asian or Eastern origin ("wukong", "nezha", "cao", or other words in Chinese and Japanese) suggests games built around a specific and recognizable imagery, in which value lies in the setting and mythology of the video game. On the other hand, the emergence of numerous terms related to special and curated editions ("full exclusive", "special collector", "exclusive extras", "standard version") indicates a strong emphasis on the completeness of the product and the enrichment of the experience through additional content, extra materials, and differentiated editorial versions.

These textual signals are consistent with the numerical profile of the cluster, which exhibits very low levels of active engagement despite relatively higher prices. *Average.playtime.forever*, *Peak.CCU*, and *Total_Reviews* remain marginal, indicating that games are often purchased but only sporadically used. The presence of publishers specializing in older titles further supports the interpretation of a consumption model oriented toward archival or retrospective use.

The Shelf Gamer represents a consumer who engages with videogames in a passive and discontinuous way. Titles are acquired as complete, independent

cultural products, often for their familiarity or recognizability, but rarely lead to sustained engagement.

4. **Party Gamer**

   The vocabulary associated with this cluster is centered on socializing, competition, and multiplayer interaction. Recurring bigrams make explicit reference to shared game modes ("multiplayer screen", "local co-op", "multiplay friend", "competitive mode"), suggesting experiences designed to be enjoyed in person, with multiple players sharing the same physical space or screen.

   Alongside these elements, technical terms such as "competitive mode" and "online features" indicates the presence of online competition elements. As for unigrams, there are words such as "dodgeball", "dunk", "sumo" or "knockout", which refer to arcade sports and stylized combat, and keep pointing toward a shared play experience.

   Numerically, this cluster is characterized by relatively high values of *Average.playtime.forever* and *n_categories*, reflecting feature-rich products with multiple modes and mechanics. *Device_support* tends to be low and more focused on Windows, suggesting an audience with a stable technical setup.

   The Party Gamer can therefore be described as a consumer who values social engagement and interactive complexity. Games are chosen for their ability to facilitate shared experiences and competition, rather than for narrative immersion or solitary exploration.

5. **Comfort Gamer**

   The textual patterns observed in this cluster are emphasized by the presence of proper names and references, such as characters, explicit titles, and names related to commonly used interfaces or software. Recurring references to characters ("Angela", "Hercules", "Susan"), narrative titles ("Lost Lands"), and puzzle or casual games ( "Hidden Cat") indicate products built around guided, often linear experiences designed for a wide audience. Moreover, bigrams like "French German" highlight the presence of multiple languages available for the

titles, which is confirmed by the variable *n_languages* that displays the highest value in this cluster.

*Price* tends to be contained, and features such as the category family sharing reinforce a model of relaxed and inclusive consumption. As a consequence, the Comfort Gamer represents a user who prioritizes familiarity, clarity, and ease of use over complexity or shared experiences. Videogames are consumed as accessible and reassuring experiences, designed to be immediately understandable.

**6. Choice Gamer**

The language defining this cluster revolves around textual interaction, narrative choice, and personalization of the experience, with an explicit emphasis on the centrality of the player in the construction of branching narrative paths.

Recurring bigrams (“entire text”, “word interact”, “power imagine”) indicate games in which the user's decisions directly influence the development of the story. Terms related to identity and representation (“bi”, “gay”, “asexual”, “nonbinary”) further highlight the importance of personalization and expressive freedom within these titles.

From a quantitative standpoint, games in this cluster exhibit a limited *n_categories* and moderate engagement levels as can be seen by variables *Total_Reviews*. *Device_support* is very high, ensuring stability across platforms, while localization remains more selective due to low level of *n_languages* supported, reflecting a focus on specific audiences or narrative communities.

The Choice Gamer is therefore characterized by consumers who are oriented towards interactive narrative and choice, who value freedom of decision and identification with the character. They favor experiences with emotional involvement, without focusing on global mass distribution or the technical features of the games.

**7. Grind Gamer**

The linguistic identity of this cluster is dominated by globally recognized franchises and established intellectual properties. The most recurring unigrams

(“lego”, “disney”, “batman”, “gundam”, “creed”, “motogp”) and corresponding bigrams (“star wars”, “call duty”, “ninja gaiden”, “metal gear”) refer almost exclusively to established narrative universes, known to the public well before the specific gaming experience. The recurrence of terms such as “time ever” and “fan favorite” indicates a strong emphasis on the fanbase of these titles.

This narrative structure aligns with a numerical profile characterized by extremely high levels of engagement. Games in this cluster exhibit the highest values of *Average.playtime.forever* and *Peak.CCU*, indicating sustained participation and long-term involvement. *Price* is typically higher, reflecting a willingness to invest in products that offer persistent content, regular updates, and strong social ecosystems.

The Grind Gamer emerges as a deeply committed consumer oriented towards global franchises and premium versions. They favor competitive, persistent, and social experiences, investing significant time, attention, and financial resources, focusing on growth, performance, and community belonging rather than casual experiences.

# 7. Managerial Implications

## 7.1 Managerial implications for platform stakeholders

The findings of this thesis provide strategic insights for stakeholders operating within the Steam ecosystem, including developers, publishers, and platform operators. By uncovering a structured and interpretable segmentation of the catalogue, the analysis demonstrates that video game success on Steam cannot be reduced to a single dominant model, but rather emerges from multiple, qualitatively distinct configurations of game characteristics and consumption patterns.

From a managerial point of view, this implies that strategic decisions should be evaluated within the context of comparable segments. Games primarily compete with titles sharing similar economic, technical, and engagement profiles, and their performance should therefore be evaluated against targeted reference groups rather than against the platform as a whole. This perspective reduces the risk of misunderstanding performance signals and supports more accurate positioning.

## 7.2 Implications of consumer profiles for strategic decision-making

A key contribution of this thesis is the construction of consumer profiles derived from the integration of numerical clustering results and textual analysis of game descriptions. These profiles translate abstract statistical segments into interpretable consumption habits, making the results more accessible to general audiences.

From a managerial perspective, these consumer profiles can help developers match their game's features and communication strategies with the unique habits of each market segment. For example, clusters characterized by intense engagement and community-driven dynamics correspond to consumers who value persistence, competition, and continuous content updates. In contrast, segments associated with low or intermittent engagement reflect consumption patterns oriented towards experimentation or completeness, rather than sustained play.

## 7.3 Managerial insights from textual analysis

The extensive textual analyses conducted in this study play a crucial role in contextualizing and interpreting the numerical segmentation. While both classical and

neural textual embeddings were shown to have limited explanatory power in predicting cluster membership, the subsequent unigram and bigram analyses revealed meaningful differences in narrative structures and linguistic patterns across clusters.

From a managerial perspective, these findings highlight the limits of standardized marketing language in differentiating products within a crowded digital marketplace. The strong lexical overlap observed at the unigram level suggests that generic descriptive strategies provide little informational value for positioning. In contrast, the bigram analysis reveals that distinctive patterns within the phrasing are more effective in characterizing different segments, reflecting variations in themes, mechanics, identity representation, and franchise affiliation.

These results suggest that communication strategies aligned with the linguistic patterns of the segment may improve coherence between a game's positioning and its target audience. Rather than increasing the volume of descriptive content, developers and publishers may benefit from focusing on narrative structures and linguistic framing that resonates with the consumption profile associated with their segment.

## 7.4 Implications for platform governance and discoverability

From the perspective of platform operators, the coexistence of heterogeneous consumer profiles and linguistic identities has direct implications for discoverability and recommendation systems. Treating the catalogue as a homogeneous tends to prioritize the most active player bases, while overlooking other ways of playing that remain consistent and valuable to the ecosystem.

Integrating structural segments into recommendation systems can help balance visibility across the whole catalogue. Because current descriptions are often too similar, tools that promote more specific and unique content could help players navigate the catalogue with more clarity.

## 7.5 Discussion of research questions

The empirical findings of this study provide clear answers to the research questions formulated in Chapter 2:

- **RQ1: Can the Steam catalogue be meaningfully segmented into a limited number of coherent and interpretable clusters?**

  The results provide a positive answer to this question. The clustering analysis identifies a solution based on seven clusters that is statistically robust and analytically interpretable, demonstrating a consistent way to categorize the diversity of the Steam catalogue.
- **RQ2: Which variables play a dominant role in defining these clusters?**

  The Random Forest and SHAP analyses reveal that *Price*, *device_support*, *n_languages*, and *n_categories* are the primary drivers of cluster differentiation. Engagement metrics, such as *Total_Reviews*, *Peak.CCU*, *Average.playtime.forever*, and *Review_Ratio*, provide additional discrimination among the most active clusters
- **RQ3: Do the identified clusters correspond to distinct patterns of consumption and engagement?**

  Yes. These consumer profiles show that each cluster follows a specific style of consumption. This ranges from casual play with minimal involvement at one end, to long-term engagement fueled by social dynamics or a deep focus on the story at the other. The integration of textual analysis reinforces these distinctions by highlighting coherent narrative and thematic patterns aligned with each profile.
- **RQ4: Can machine learning techniques improve the understanding of cluster structure?**

  The application of SHAP confirms that interpretable machine learning techniques are essential for explaining cluster membership and translating complex models into insights that offer both academic depth and real value for industry stakeholders.

## 7.6 Concluding managerial insights

Overall, the managerial relevance of this thesis lies in its ability to connect structural segmentation, interpretability, and consumption narratives within a unified framework. By integrating numerical analysis with textual interpretation and consumer profiling, the study provides a detailed understanding of how different types of games are positioned and consumed on Steam.

Rather than promoting universal optimization strategies, the findings encourage stakeholders to recognize the many ways to succeed on the platform and to align strategic decisions with the structural and narrative traits of their target segment.

# 8. Limitations and Directions for Future Research

## 8.1 Limitations of the study

Despite the robustness of the analytical framework and the scale of the dataset employed, this thesis is subject to several limitations that should be acknowledged when interpreting the results.

A first limitation concerns the nature of the available data: the analysis relies exclusively on game information aggregated at the platform level. While this approach is well suited for analyzing the structure and the overall catalogue, it does not allow for direct observation of individual user behavior. Consequently, the consumer profiles derived in 65this study represent interpretative constructs inferred from game characteristics and engagement patterns, rather than empirically observed user segments.

A second limitation relates to the use of estimated ownership figures. Steam does not publicly disclose exact sales data, and ownership estimates provided by Steam Spy are based on sampling and inference techniques. While these estimates are widely used in academic research and provide valuable approximations of market reach, they inevitably involve some level of approximation. However, the strong link between these estimates and actual player activity shows that the structural patterns identified in the study are still accurate.

An additional limitation arises from the restricted role of textual data. While extensive textual analyses were conducted, both classical and neural embeddings failed to align with the numerical cluster structure. This outcome reflects the standardized and promotional nature of Steam descriptions rather than a limitation of the analysis.

Finally, the analysis adopts a static perspective on the Steam catalogue. All variables are observed at a specific point in time, capturing accumulated engagement and structural characteristics but not their evolution. As a result, the study does not address how games transition between segments over time as their popularity and engagement change over the years.

## 8.2 Directions for future research

The limitations identified above point to several promising directions for future research. A natural extension of this work would involve the integration of user-level behavioral data, where available. Access to anonymized play histories, session durations, or

purchasing sequences would allow for the empirical validation of the consumer profiles proposed in this thesis and enable the study of how different user types interact with structurally distinct segments of the catalogue.

Future research could analyze the progression of games over time, tracking games across time to analyze transitions between clusters. Such an approach would provide insights into lifecycle dynamics, including how updates, pricing changes or localization strategies can influence a game's structural position within the platform.

Textual analysis also offers opportunities for further development. While this thesis focused on the promotional language used by developers and publishers in their games' descriptions, future work could explore the perspective of the community. By analyzing user reviews or forum discussions, researchers could align the way players talk about games with the way distributors market them. Such an alignment would allow developers to direct their messaging more effectively, ensuring the game's value proposition resonates with the right community.

# 9. Conclusion

This thesis is oriented to investigate the structural organization of the Steam catalogue, with the aim of moving beyond metrics of success and toward an interpretable representation of how heterogeneous video games coexist within a large digital distribution platform. By combining large-scale descriptive analysis, clustering techniques, interpretable machine learning models, and textual analysis, the study provides a comprehensive account of the multiple pathways through which games achieve relevance, engagement, and sustainability on Steam.

The empirical results demonstrate that the Steam catalogue is not an unstructured continuum dominated only by a small number of blockbuster titles, but rather a system composed of distinct and coherent segments**.** The seven-cluster solution identifies stable configurations of economic, technical, and engagement characteristics, highlighting the coexistence of low, mid, and top-tier segments driven by different strategic logics. Moreover, the analysis shows that success on Steam cannot be reduced to a single dimension, but instead emerges from the interaction of pricing strategies, technical accessibility, internationalization, and patterns of user engagement.

The integration of Random Forest classification and SHAP model plays a crucial role in clarifying the mechanisms underlying this segmentation. Rather than treating the clusters as purely descriptive outcomes, the interpretability analysis reveals how a limited set of variables drives cluster differentiation, reinforcing the robustness and explanatory value of the proposed framework. In parallel, the extensive textual analyses show that while standardized marketing language dominates individual word usage, meaningful differentiation emerges at the level of phrase structures and narrative patterns. These linguistic signals contribute to the interpretation of clusters and support the construction of consumer profiles that translate quantitative structures into coherent modes of consumption.

By deriving consumer profiles from the combined analysis of numerical and textual data, the thesis fills the gap between statistical segmentation and qualitative interpretation. These profiles do not aim to represent individual users, but rather to summarize recurring patterns of interaction between games and players, offering an interpretative framework through which structural differences across segments can be understood.

Overall, this thesis contributes to the literature on digital platforms and the video game industry by proposing a structural, interpretable, and data-driven perspective on the Steam ecosystem.

# 10. References


- **Gaudiosi, J. (2004). Digital distribution poised to dominate game sales.**
- **Walker, J. (2003). Valve launches Steam platform. Valve Software.**
- https://www.gamesindustry.biz/steam-surpasses-13-million-users
- **Bergen, M., & Palmeri, C. (2018). How Apple and Google built a $100 billion app economy. Bloomberg.** https://www.bloomberg.com/news/features/2018-01-18/how-apple-and-google-built-a-100-billion-app-economy
- **Bailey, K. (2019). Steam now has over 30,000 games. PC Gamer.** https://www.pcgamer.com/steam-now-has-over-30000-games/
- **Nichols, R. (2023). Disruption through distribution: Impacts and limits in the global video game industry. In Digital disruption and media transformation: How technological innovation shapes the future of communication (pp. 269–281). Springer.** https://doi.org/10.1007/978-3-031-32079-1_15
- **Wortham, J. (2009). Apple's App Store downloads top 2 billion. The New York Times.** https://www.nytimes.com/2009/09/29/technology/companies/29apple.html
- **Pan (2018) — Android app store scale** https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/
- https://www.bizcommunity.com/Article/196/661/192547.html 2019
- **deWinter, J., Kocurek, C. A., & Nichols, R. (2014). Taylorism 2.0: Gamification, scientific management and the capitalist appropriation of play. Journal of Gaming & Virtual Worlds, 6(2), 109–127.** https://intellectdiscover.com/content/journals/10.1386/jgvw.6.2.109_1
- **Goh, E., Al-Tabbaa, O., & Khan, Z. (2023). Unravelling the complexity of the video game industry: An integrative framework and future research directions. Telematics and Informatics Reports, 12, 100100.** https://www.sciencedirect.com/science/article/pii/S2772503023000609

- **Ozalp, H. (2024). Heterogeneous development paths to growth and innovation: The evolution of the video game industry across four hubs (WIPO Economic Research Working Paper No. 84). World Intellectual Property Organization.**
  https://www.wipo.int/publications/en/details.jsp?id=4756

- **Visser, C., & Fontugne, R. (2024). Inside the engine room: Investigating Steam's content delivery platform infrastructure in the era of 100GB games. In Passive and Active Measurement (pp. 32–60). Springer.**
  https://www.iijlab.net/en/members/romain/pdf/chris_pam2024.pdf

- **Cheuque, G., Guzmán, J., & Parra, D. (2019). Recommender systems for online video game platforms: The case of Steam. In Companion Proceedings of the 2019 World Wide Web Conference (pp. 763–771). Association for Computing Machinery.**
  https://dl.acm.org/doi/10.1145/3308560.3316457

- **De Luisa, A., Hartman, J., Nabergoj, D., Pahor, S., Rus, M., Stevanoski, B., & Štrumbelj, E. (2021). Predicting the popularity of games on Steam. arXiv preprint.**
  https://arxiv.org/pdf/2110.02896

- **Ma, J. (2025). Unveiling success drivers in gaming: A machine learning study across Steam, Twitch, and Metacritic. International Journal of Computer Games Technology, 2025(1), 5776632.**
  https://onlinelibrary.wiley.com/doi/10.1155/ijcg/5776632

- **Salkanović (2024). A Data Analysis of Steam's Game Catalog and Diverse Recommendation Strategies**
  https://www.ijcaonline.org/archives/volume186/number54/salkanovic-2024-ijca-924261.pdf

# 11. Appendix

**Summary Statistics (Minimum > 0 when applicable)**

| Variable | Min | Max | Mean | Median |
|---|---|---|---|---|
| Achievements | 1.00 | 9821 | 22.84 | 6.00 |
| Average.playtime.forever | 1.00 | 145727 | 121.61 | 0.00 |
| Average.playtime.two.weeks | 1.00 | 19159 | 13.67 | 0.00 |
| DiscountDLC.count | 1.00 | 2366 | 0.61 | 0.00 |
| Estimated.owners_avg | 10000.00 | 150000000 | 101408.33 | 10000.00 |
| Median.playtime.forever | 1.00 | 208473 | 108.49 | 0.00 |
| Median.playtime.two.weeks | 1.00 | 19159 | 14.73 | 0.00 |
| Metacritic.score | 8.00 | 97 | 3.92 | 0.00 |
| Negative | 1.00 | 895978 | 189.91 | 4.00 |
| Peak.CCU | 1.00 | 1311366 | 266.40 | 0.00 |
| Positive | 1.00 | 5764420 | 1138.23 | 14.00 |
| Price | 0.35 | 999 | 8.20 | 4.99 |
| Recommendations | 101.00 | 3441592 | 919.81 | 0.00 |
| Required.age | 1.00 | 21 | 0.33 | 0.00 |
| Review_Ratio | 0.02 | 1 | 0.76 | 0.83 |
| Total_Reviews | 1.00 | 6531097 | 1328.13 | 19.00 |
| User.score | 46.00 | 100 | 0.05 | 0.00 |
| Year | 1997.00 | 2025 | 2019.78 | 2020.00 |
| cluster_k7 | 1.00 | 7 | 3.77 | 3.00 |
| device_support | 1.00 | 3 | 1.35 | 1.00 |
| n_categories | 1.00 | 22 | 3.49 | 3.00 |

*Image 1: Summary statystics of the numerical variables of the dataset whith the exclusion of 0 as a minimum.*

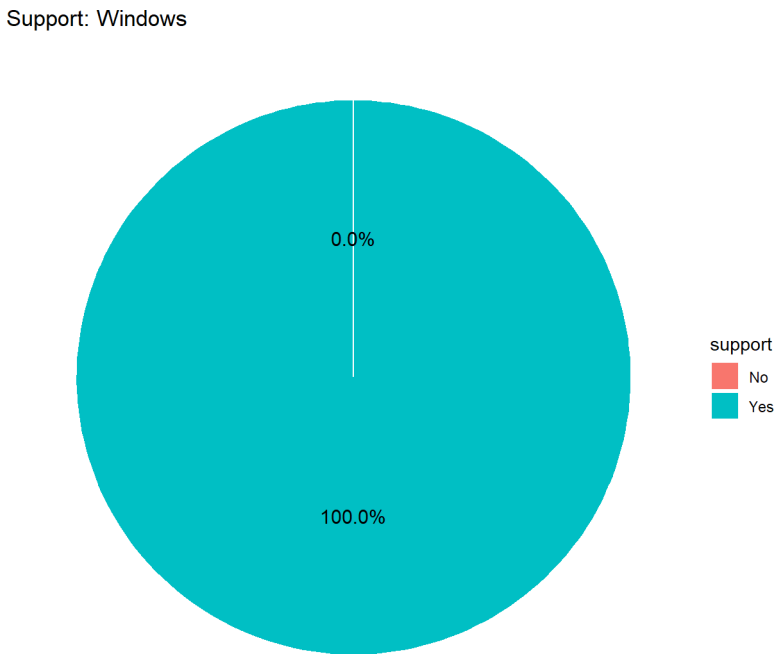


*Image 2: Graph showing the compatibility of Windows with the games in the dataset.*

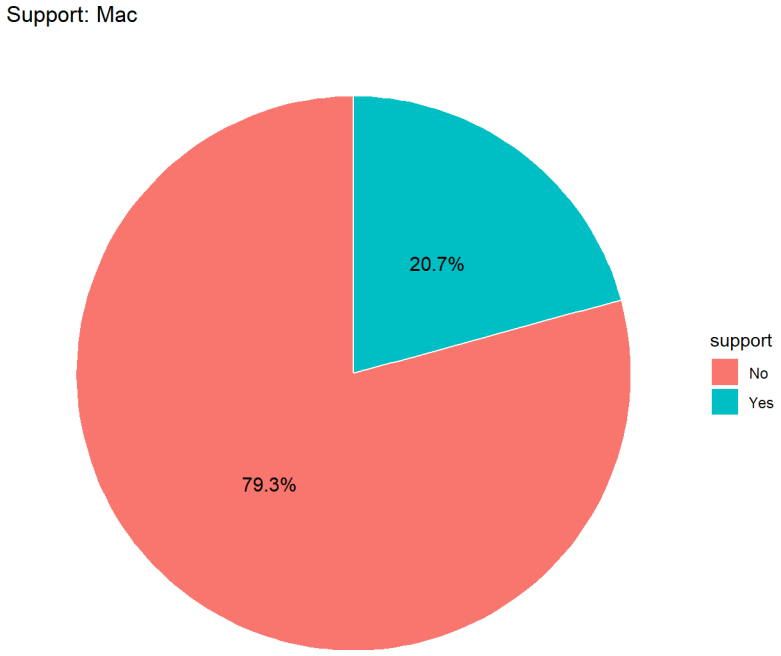


*Image 3: Graph showing the compatibility of Mac with the games in the dataset.*

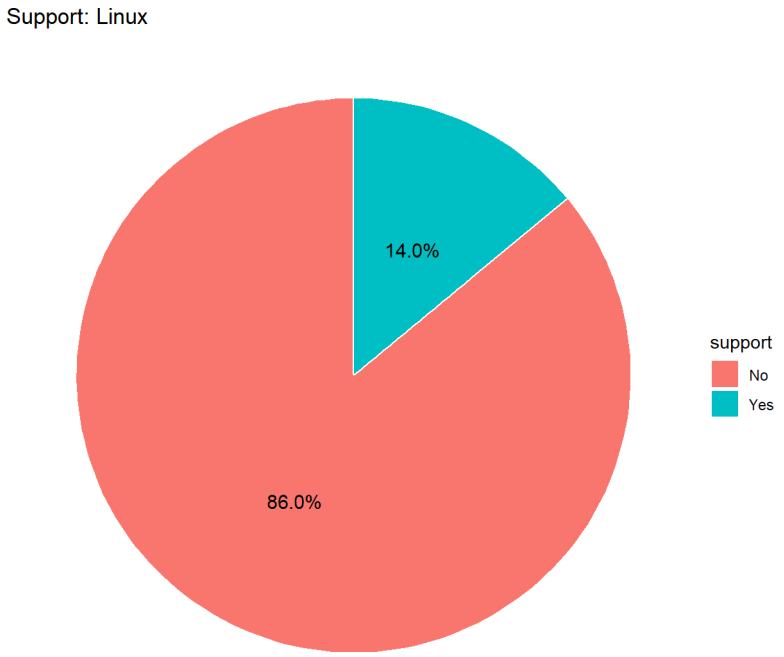


*Image 4: Graph showing the compatibility of Linux with the games in the dataset.*

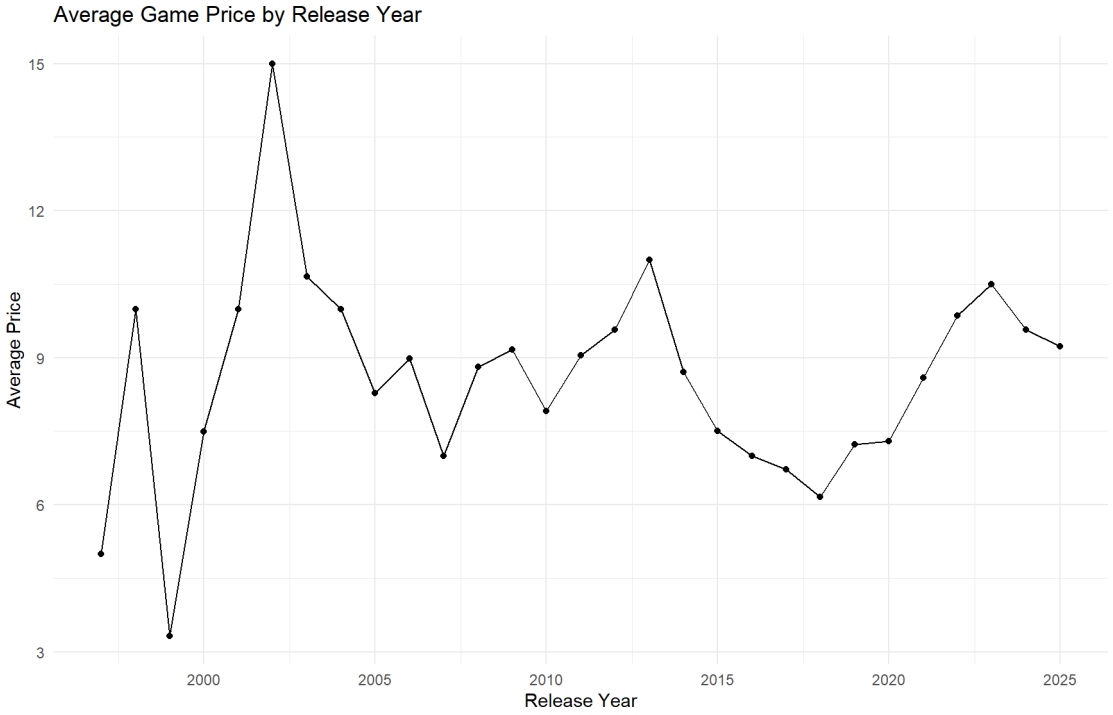


*Image 5: Average price distribution trhoughout games' release years.*

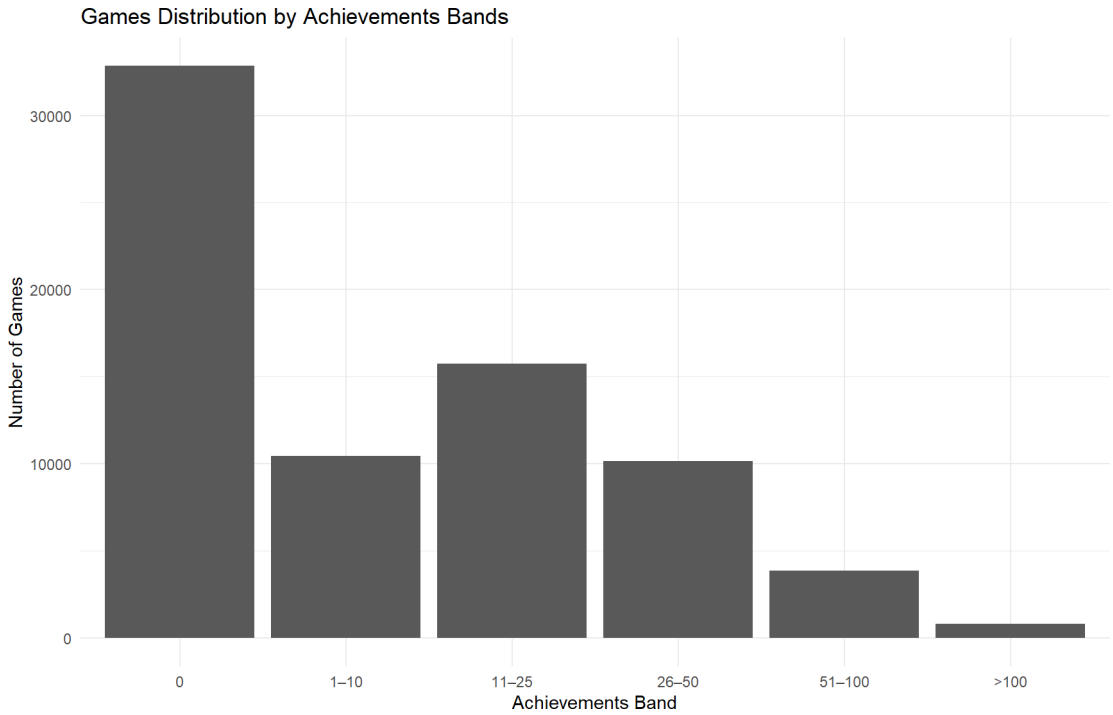


*Image 6: Number of achievements distributed per games.*

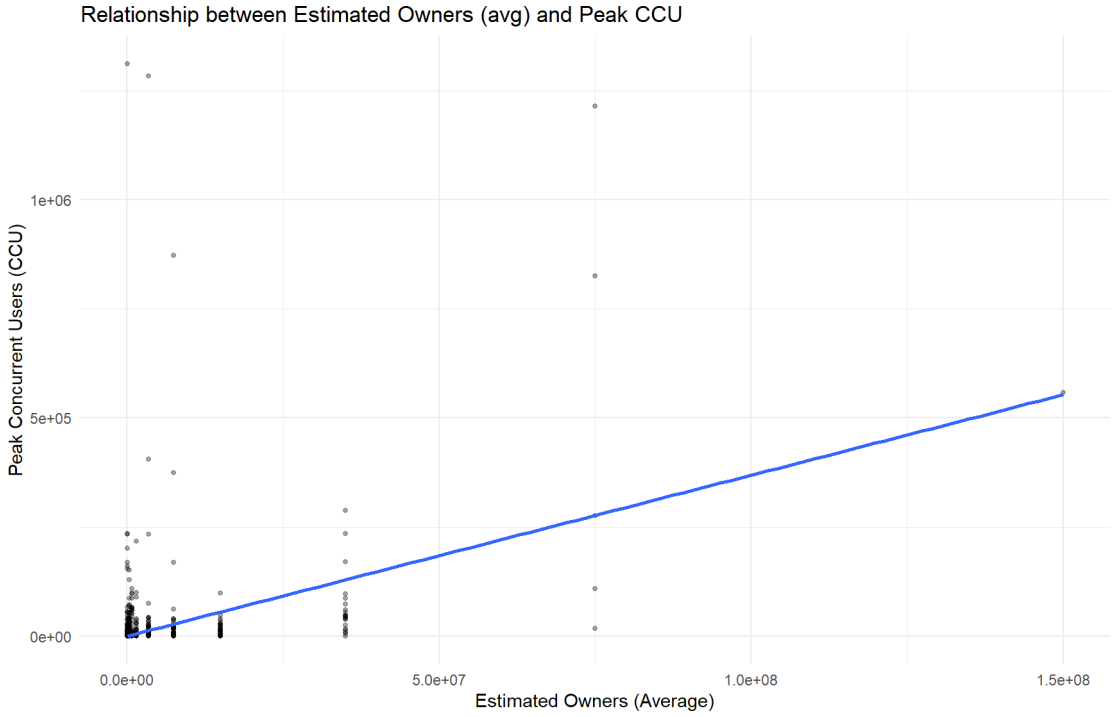


*Image 7 Regression analysis between estimated owners and peak concurrent users.*

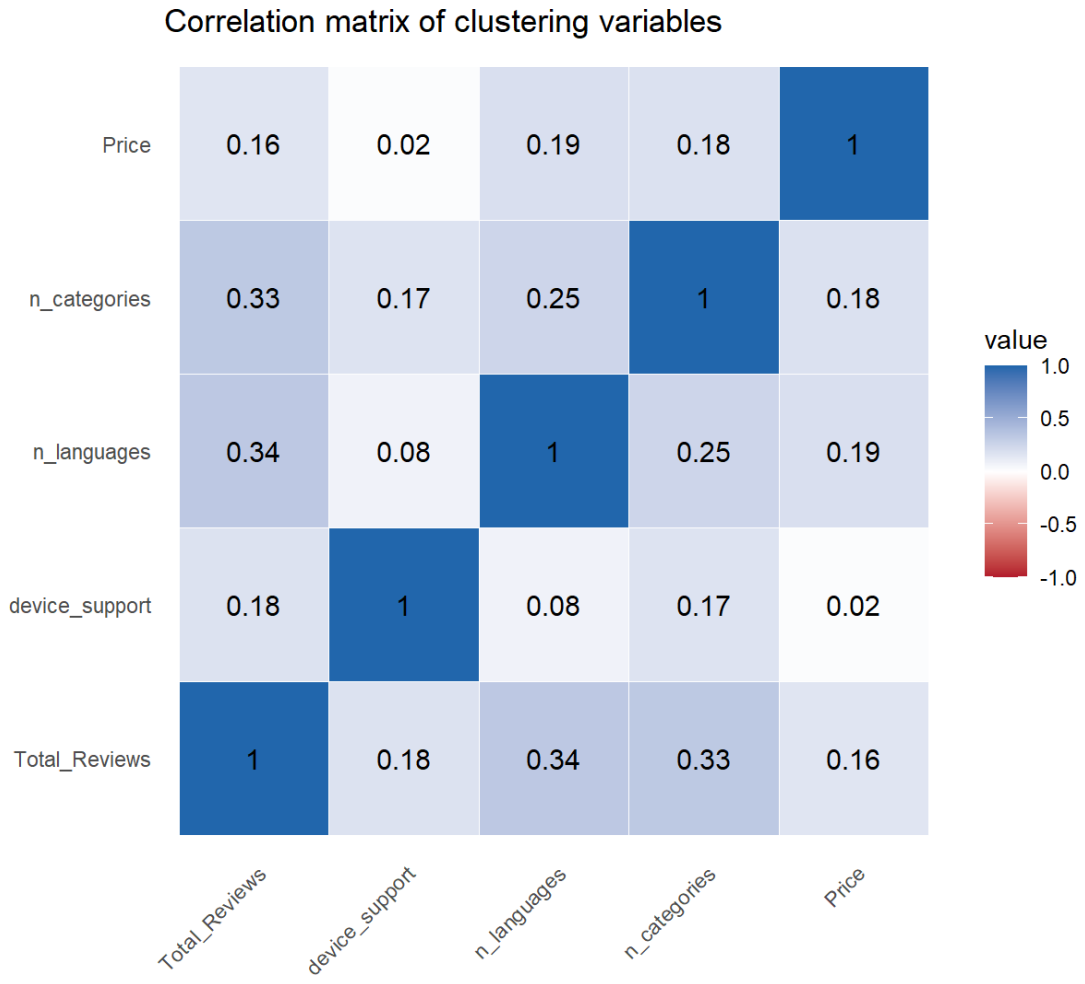


*Image 8: Correlation matrix between the variables used for the clustering analysis.*

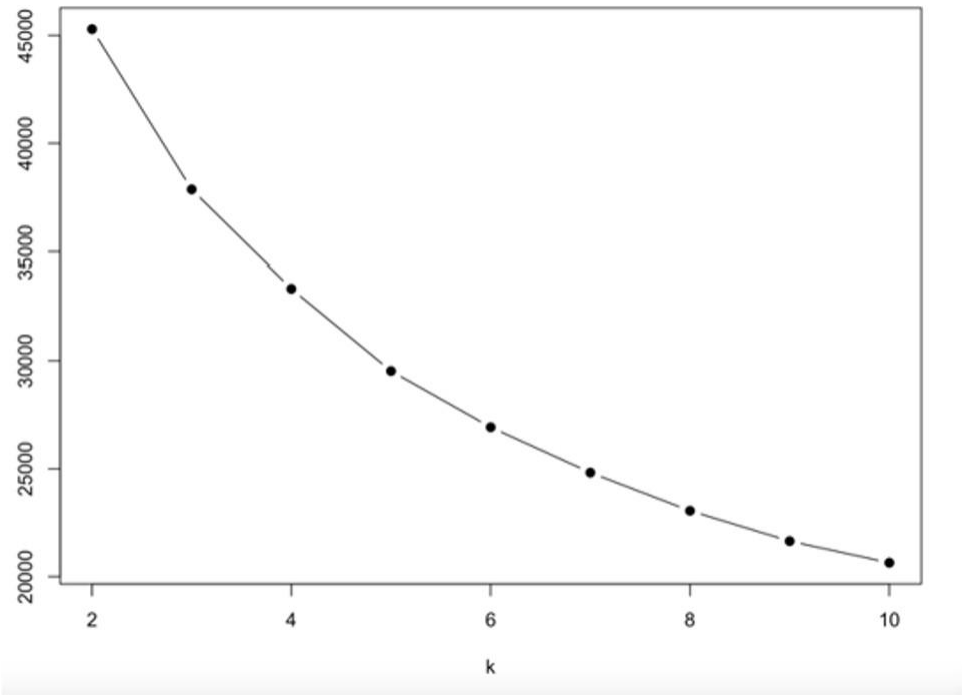


*Image 9: Elbow method.*

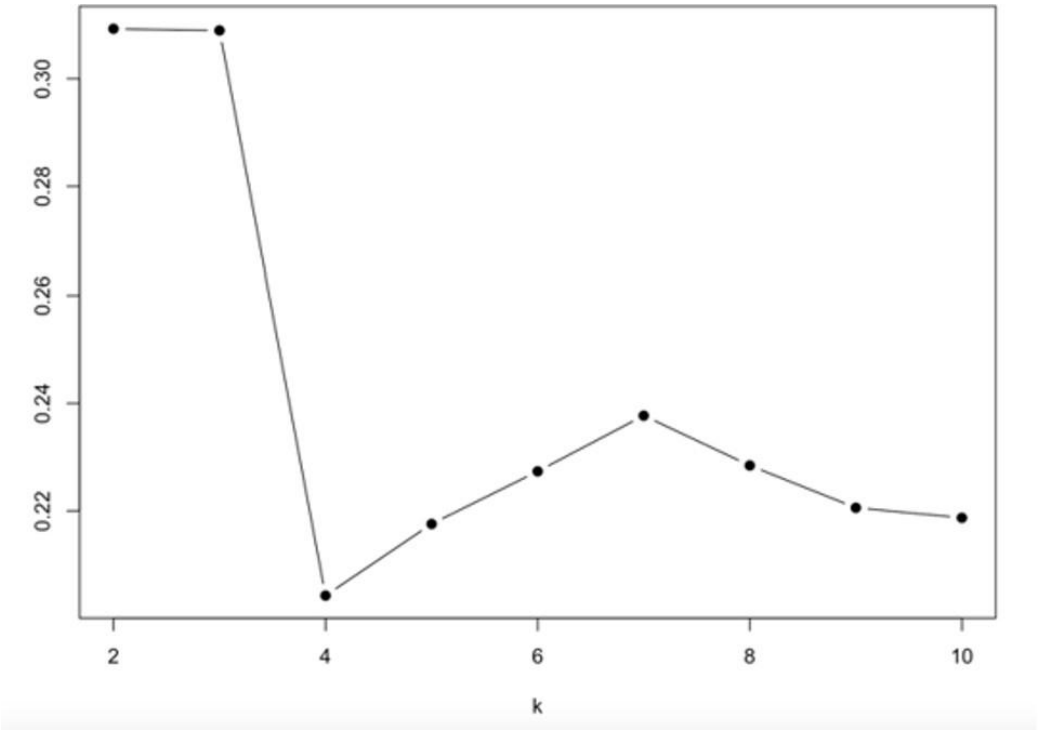


*Image 10: Silhouette method.*

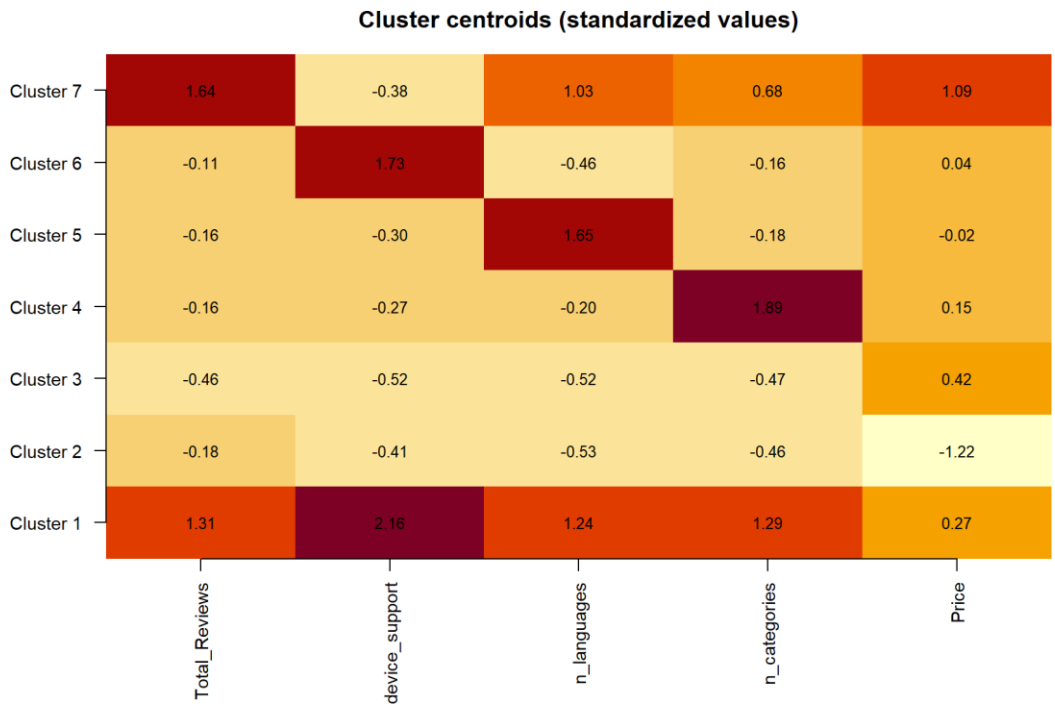


*Image 11: Clusters' centroids graph.*

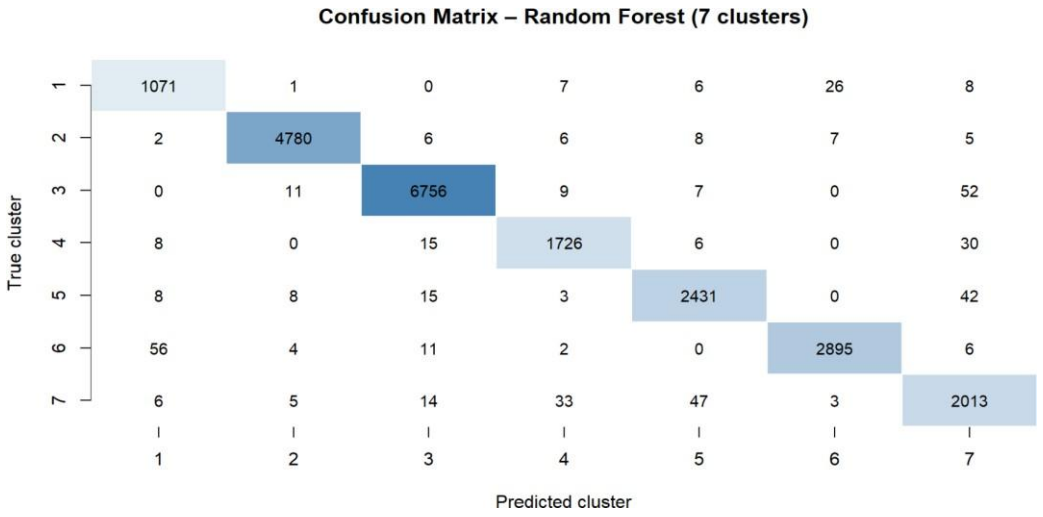


*Image 12: Confusion matrix between 7 clusters.*

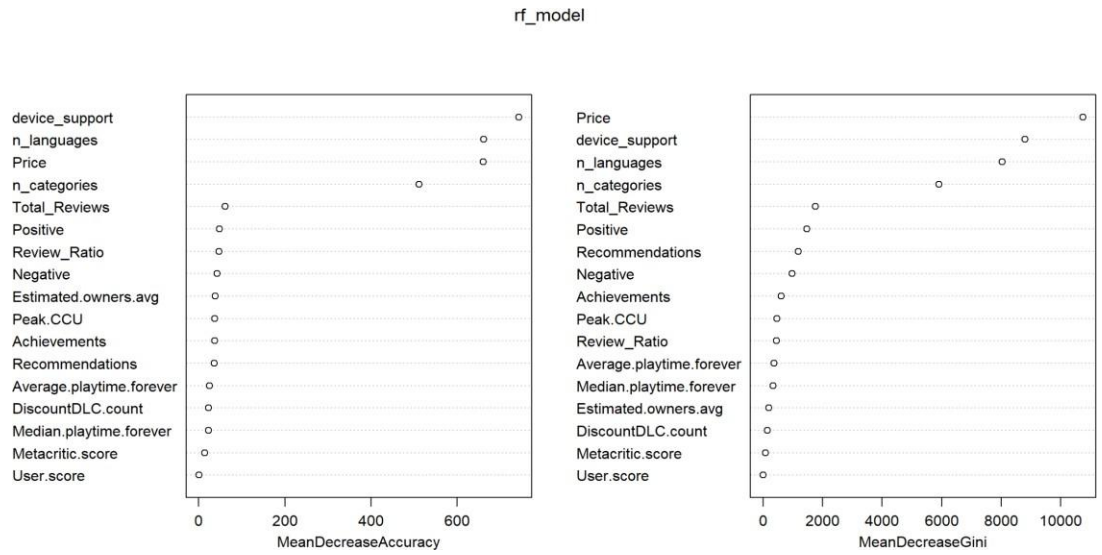


*Image 13: Mean Decrease Accuracy and Mean Decrease Gini.*

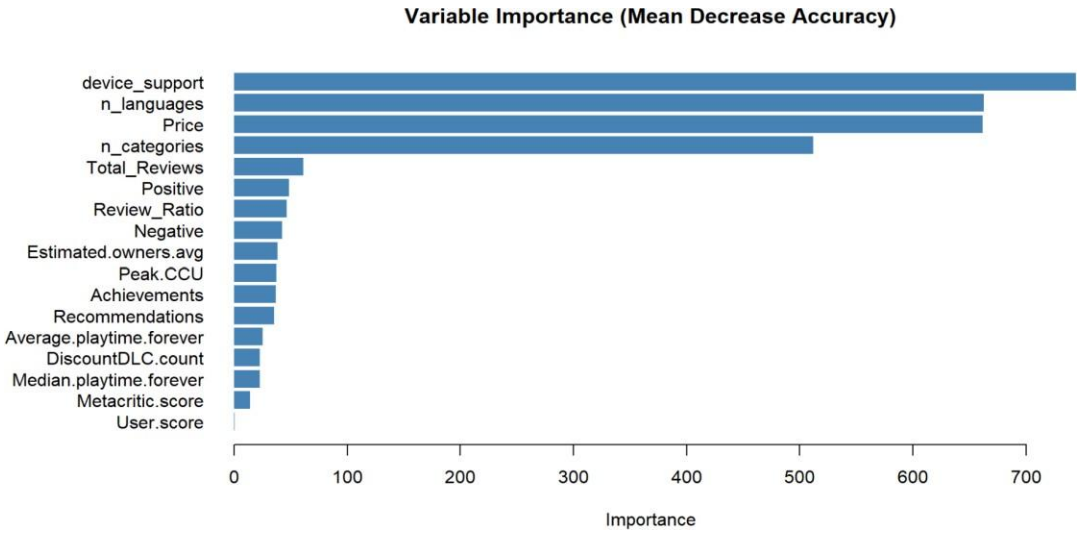


*Image 14: Global Variable Importance's graph.*

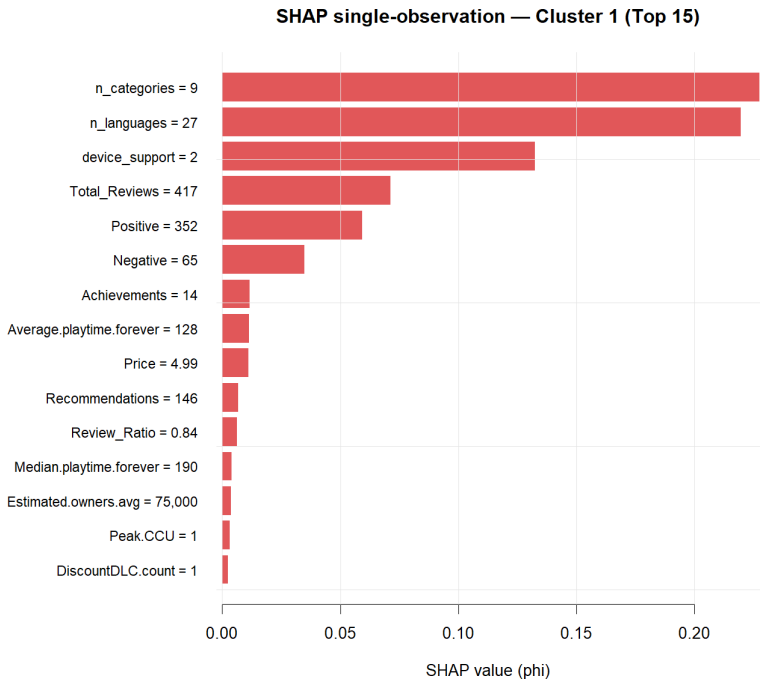


*Image 15: SHAP single observation for K1.*

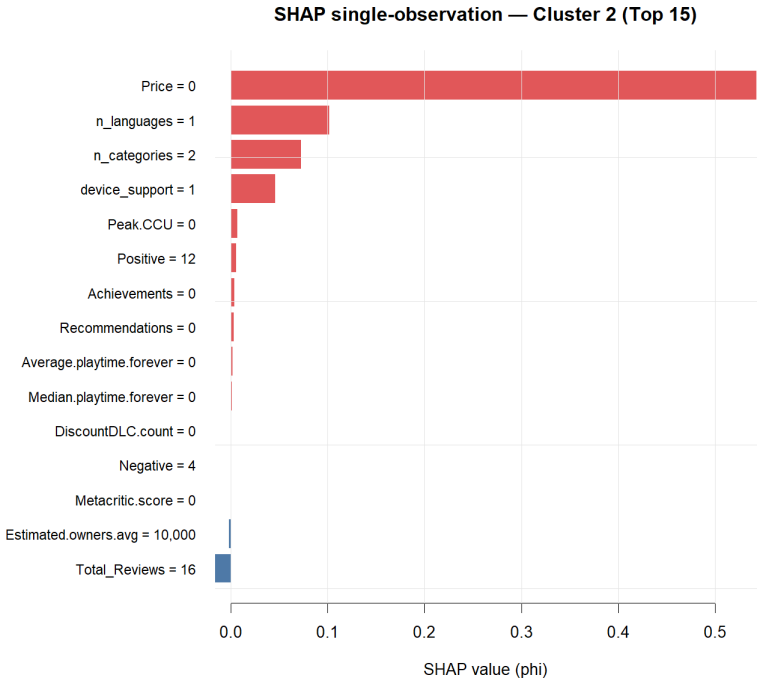


*Image 16: SHAP single observation for K2.*

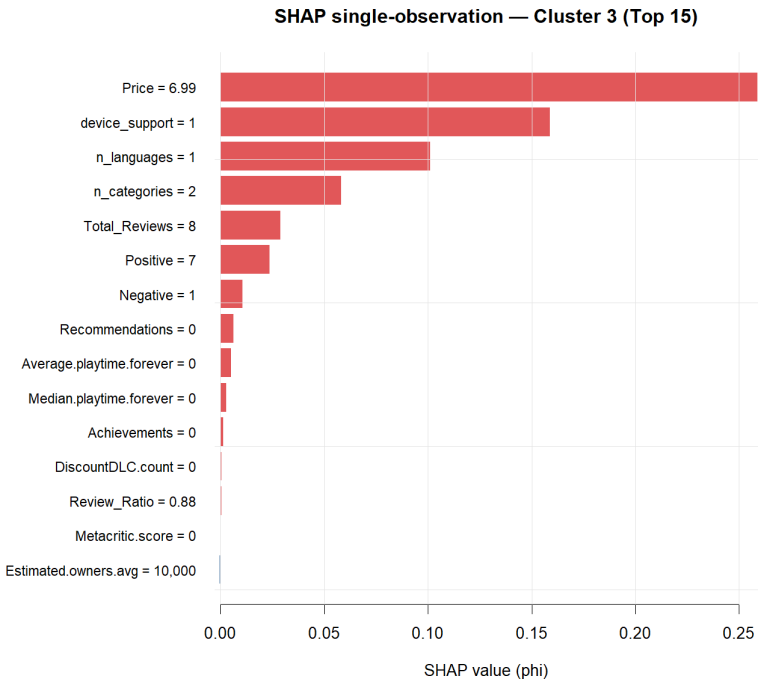


*Image 17: SHAP single observation for K3.*

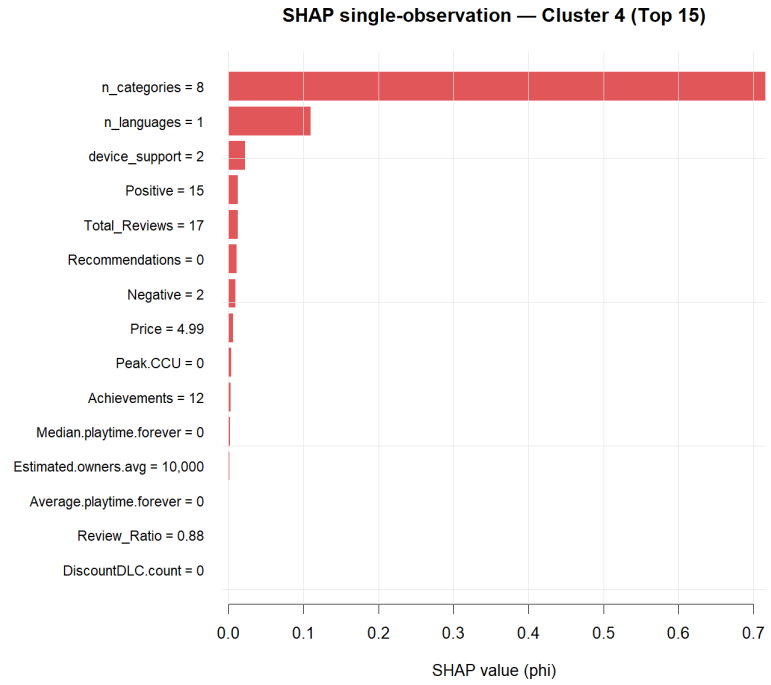


*Image 18: SHAP single observation for K4.*

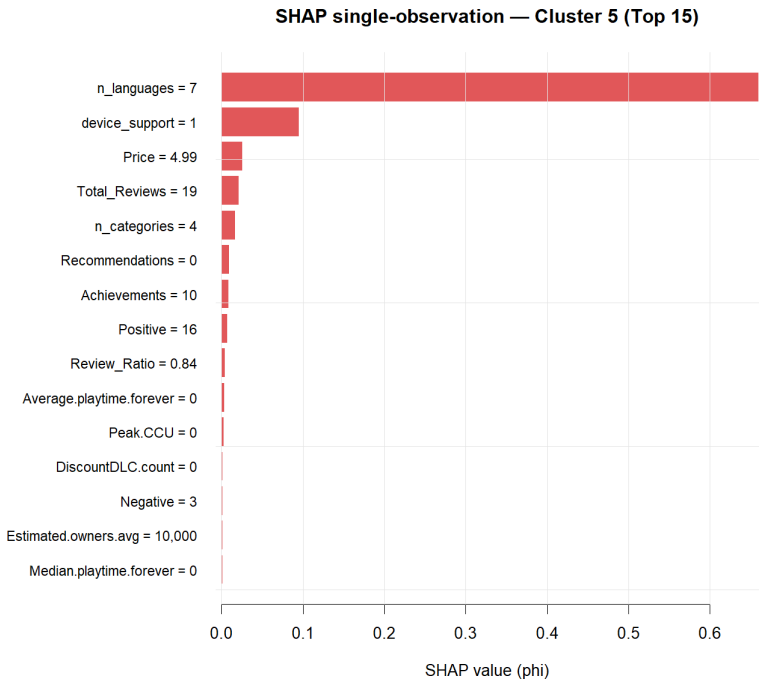


*Image 19: SHAP single observation for K5.*

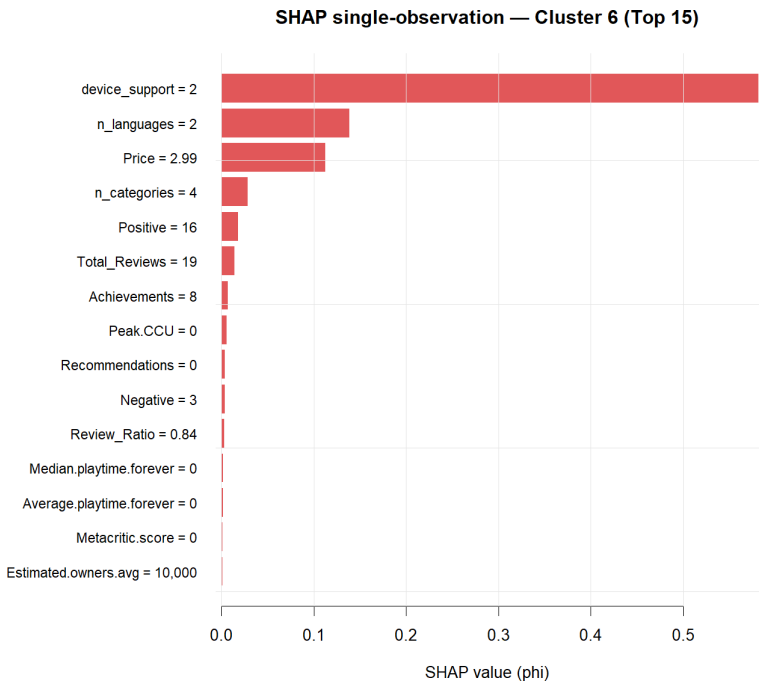


*Image 20: SHAP single observation for K6.*

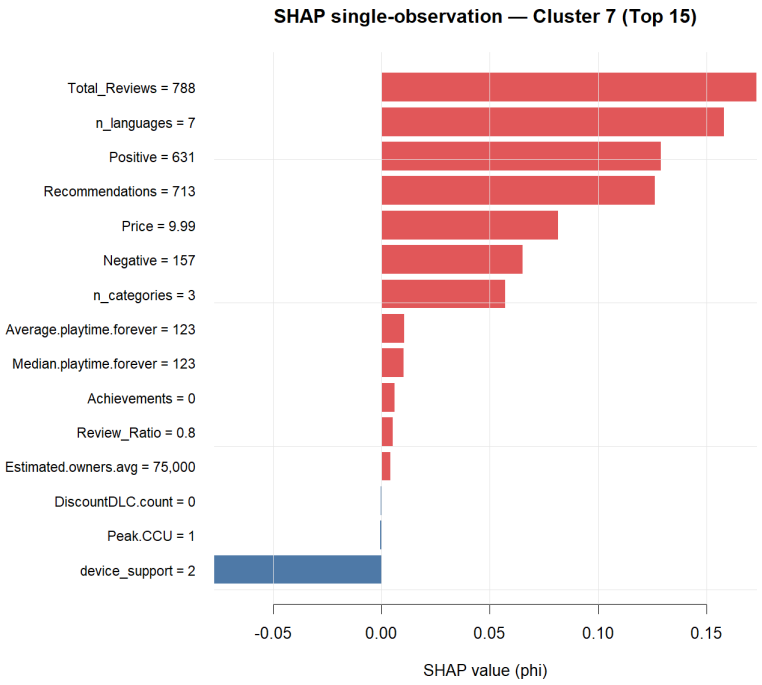


*Image 21: SHAP single observation for K7.*

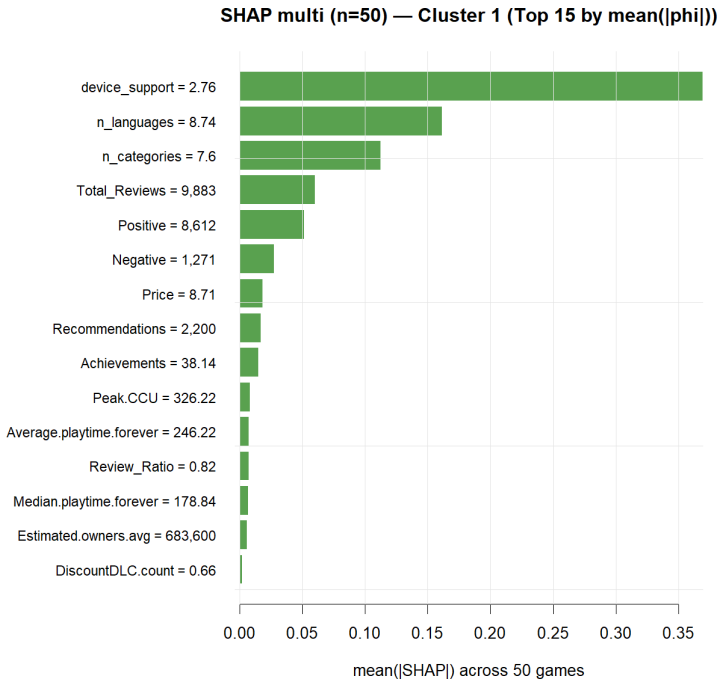


*Image 22: SHAP multiple observation for K1.*

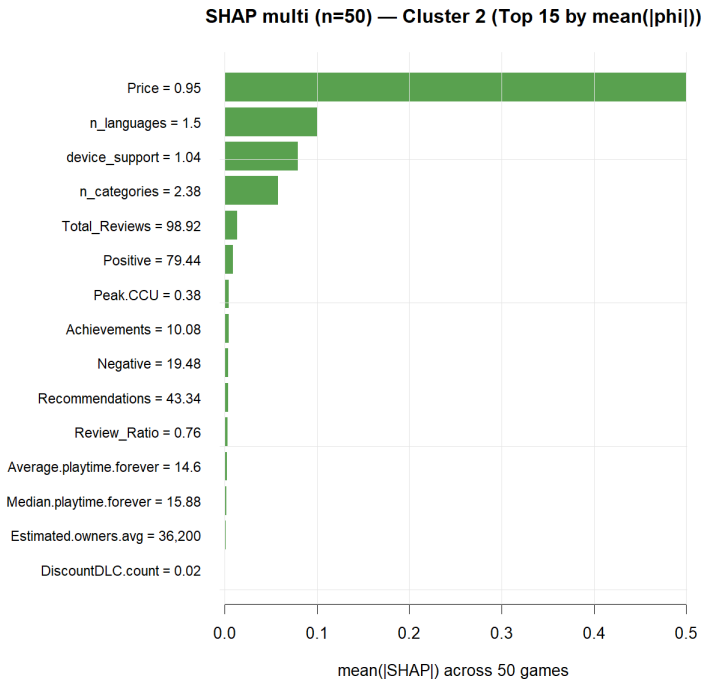


*Image 23: SHAP multiple observation for K2.*

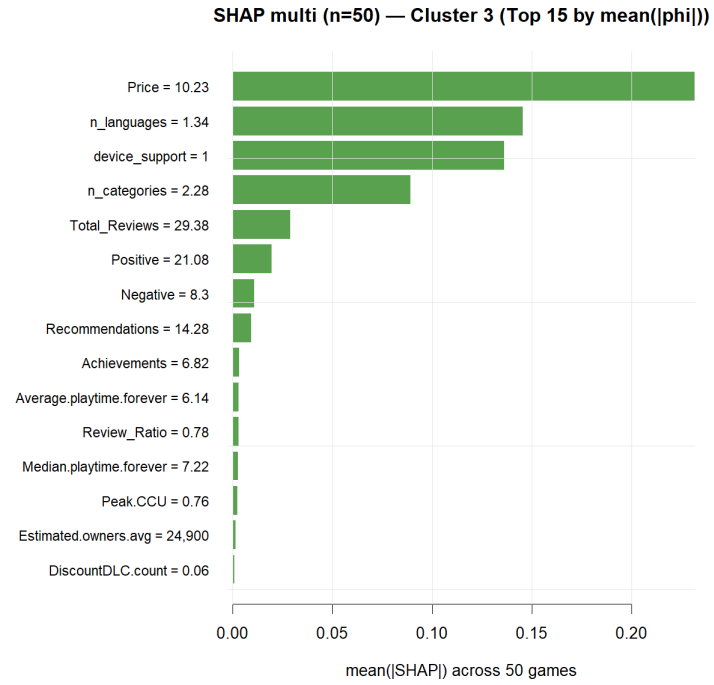


*Image 24: SHAP multiple observation for K3.*

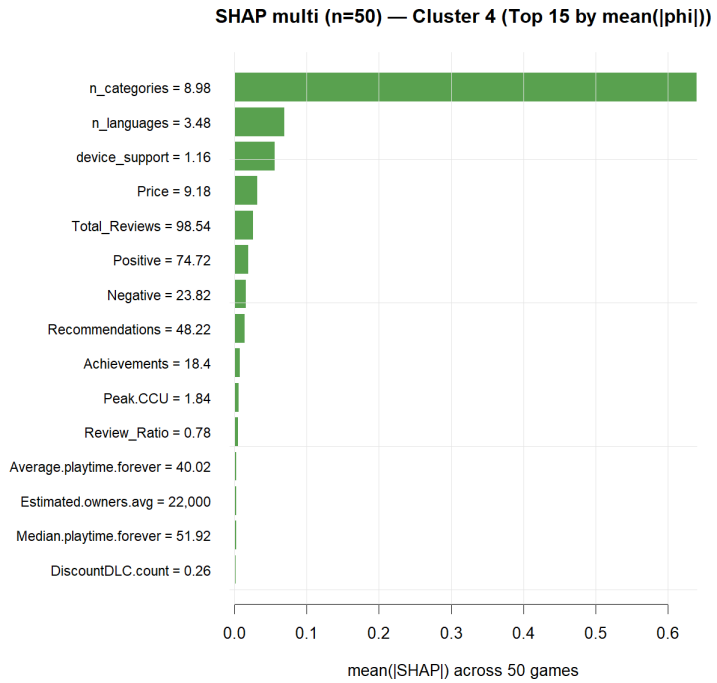


*Image 25: SHAP multiple observation for K4.*

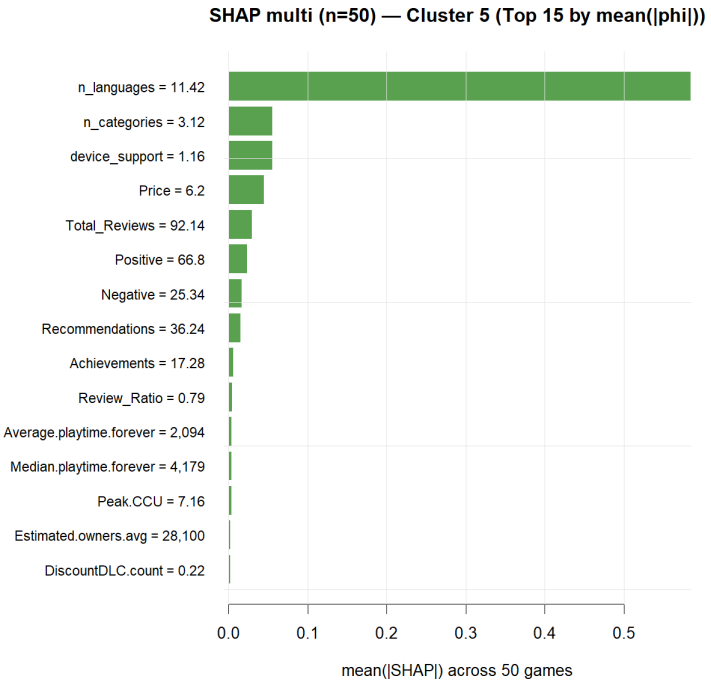


*Image 26: SHAP multiple observation for K5.*

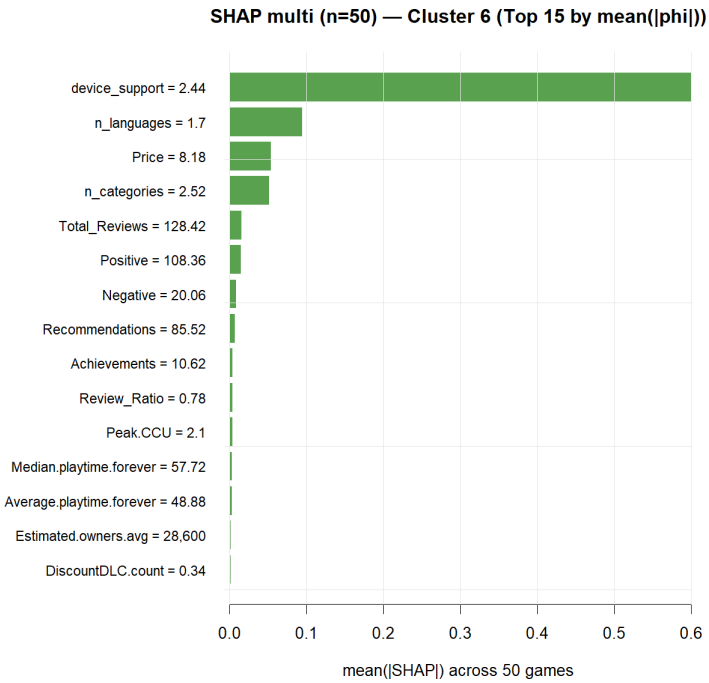


*Image 27: SHAP multiple observation for K6.*

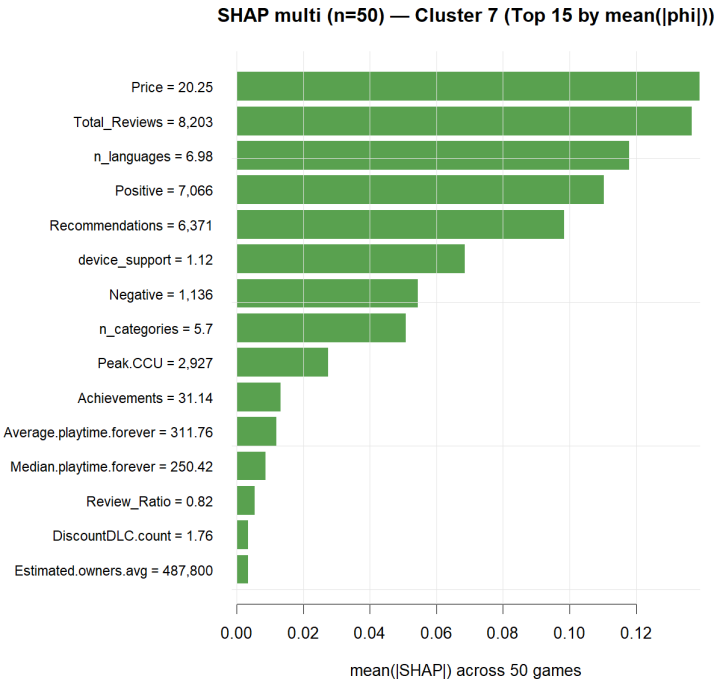


*Image 28: SHAP multiple observation for K7.*

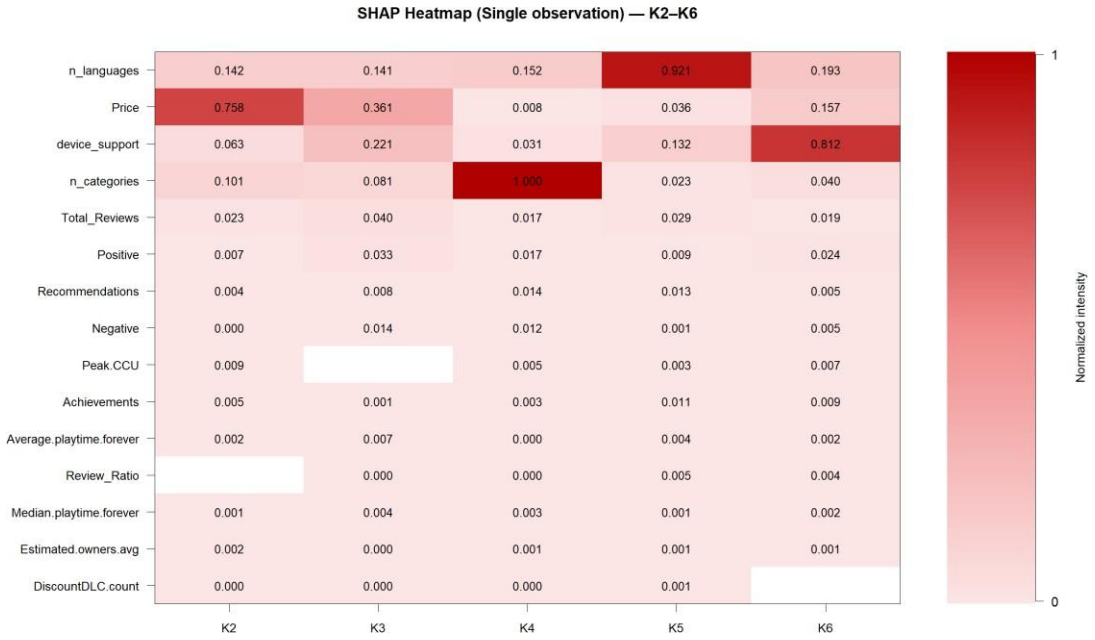


*Image 29: SHAP combined graph of single observation from clusters 2 to 6.*

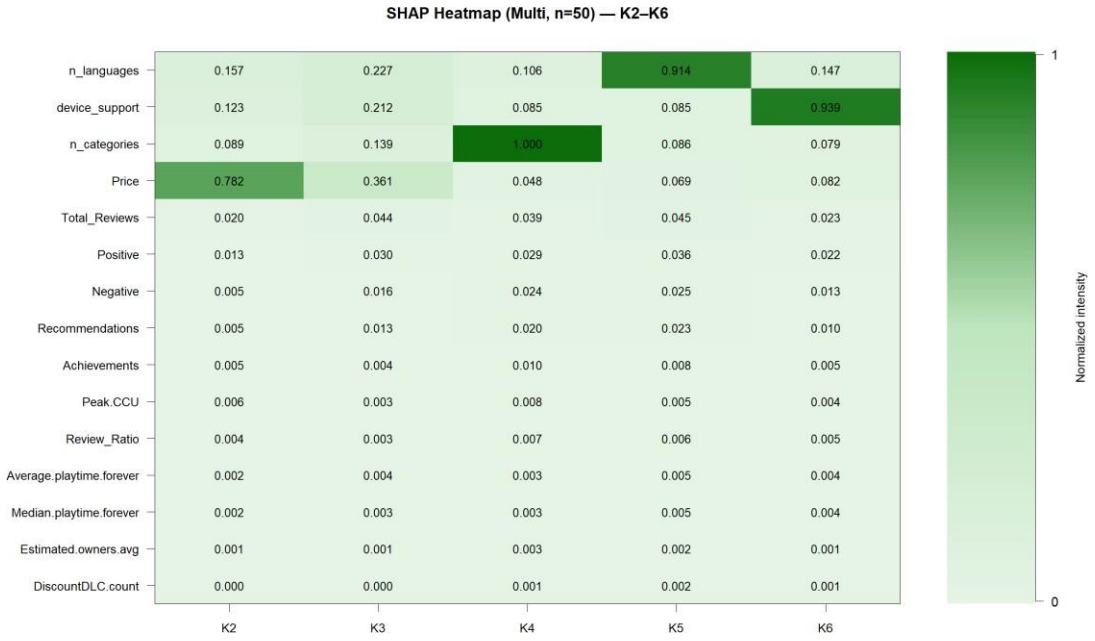


*Image 30: SHAP combined graph of multiple observation from clusters 2 to 6.*

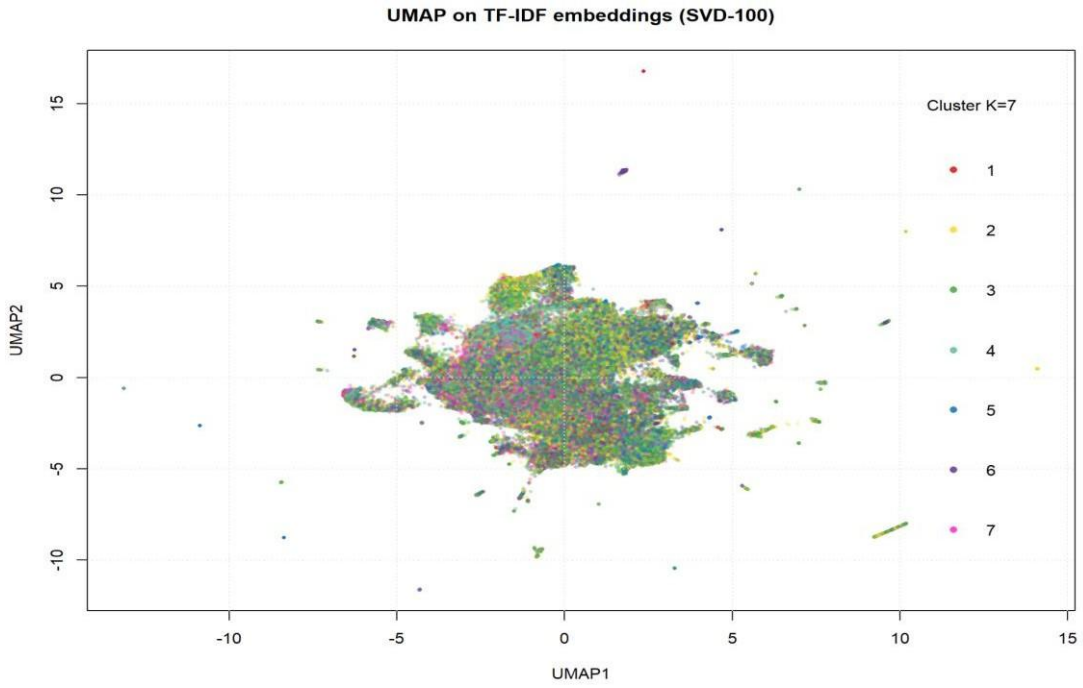


*Image 31: UMAP of embeddings divided by clusters.*

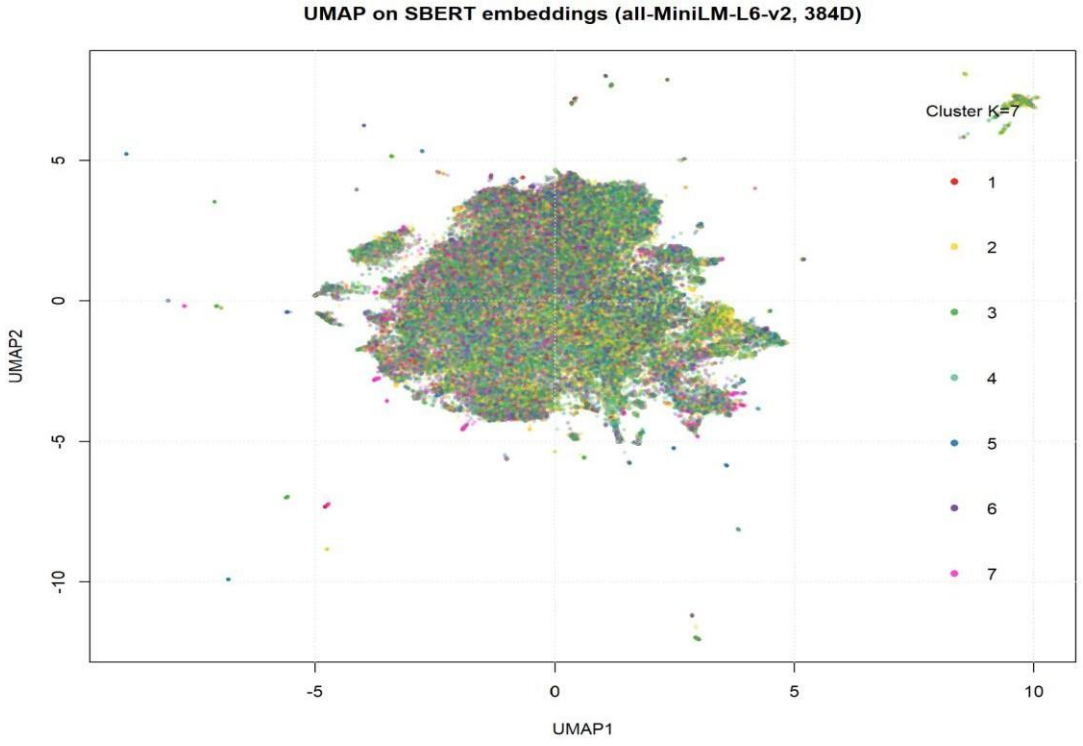


*Image 32: t-SNE visualization of embeddings divided by cluster.*

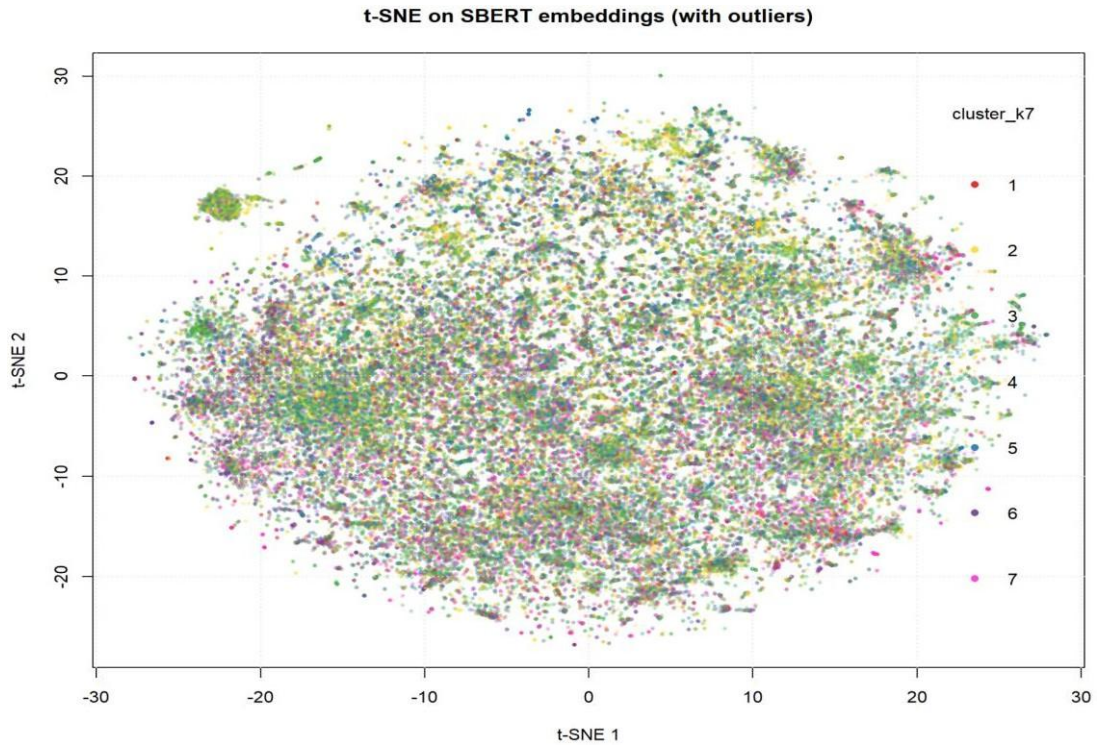


*Image 33: t-SNE visualization of embeddings divided by cluster without outliers.*

**Unique Unigrams — Top keywords per cluster (df==1)**

| | Cluster 1 | Cluster 2 | Cluster 3 | Cluster 4 | Cluster 5 | Cluster 6 | Cluster 7 |
|---|---|---|---|---|---|---|---|
| Word 1 | deponia | bitcoin | screensav | bros | angela | gay | lego |
| Word 2 | nomine | lawnmow | wukong | dodgebal | hercul | bi | disney |
| Word 3 | chelladin | ppi | nezha | dunk | powerdirector | hiroki | batman |
| Word 4 | rufus | spinner | cao | rollback | arab | asexu | gundam |
| Word 5 | trine | digipen | aldorlea | sumo | taskbar | sokpop | br |
| Word 6 | agf | poke | deriv | knockout | susan | nonbinari | liveri |
| Word 7 | shogun | megapixel | outlin | netcod | ambul | grendel | kof |
| Word 8 | geek | ape | healer | snowmen | equiti | gpl | creed |
| Word 9 | shadowrun | horac | itsuki | darius | panoram | putt | ys |
| Word 10 | xl | whack | tarot | arcana | photodirector | viola | motogp |

*Image 34: Top 10 unique unigrams per cluster.*

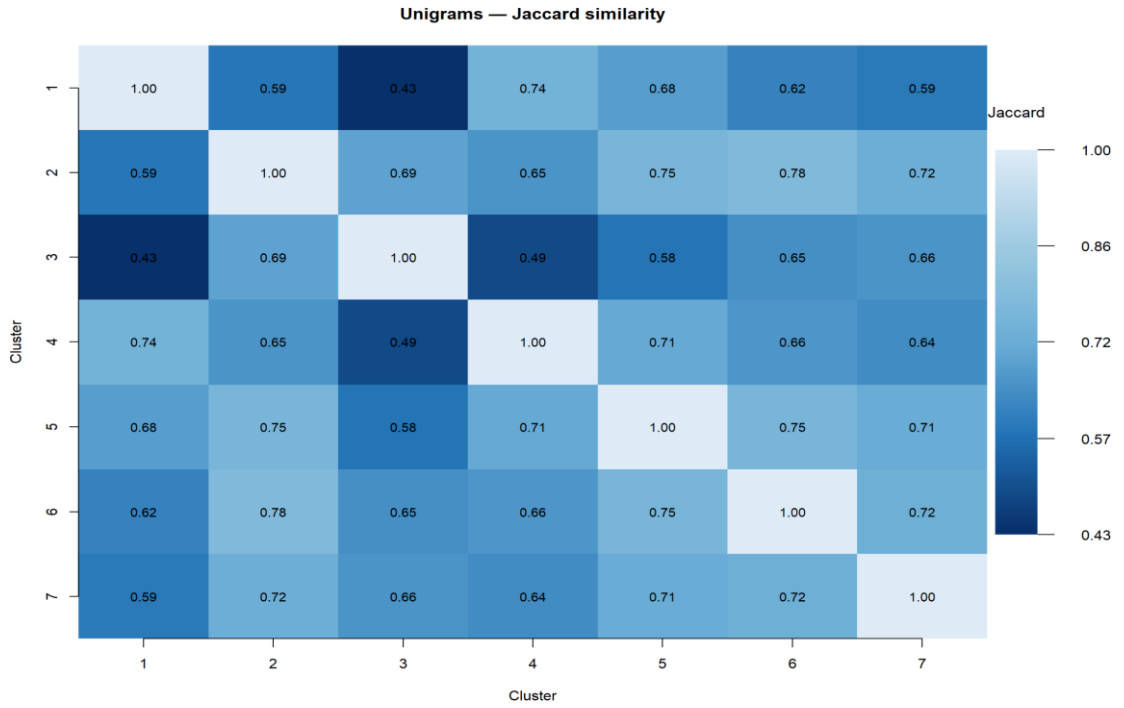


*Image 35: Jaccard's index for unigrams.*

**Unique Bigrams — Top 10 per cluster (df==1)**

| | Cluster 1 | Cluster 2 | Cluster 3 | Cluster 4 | Cluster 5 | Cluster 6 | Cluster 7 |
|---|---|---|---|---|---|---|---|
| Word 1 | tomb raider | hd qualiti | releas full | fun friend | pretti girl | entir text | star war |
| Word 2 | shovel knight | slide puzzl | standard version | multiplay friend | hidden cat | power imagin | call duti |
| Word 3 | lara s | student project | edit releas | screen multiplay | soccer manag | unstopp power | fan favorit |
| Word 4 | raider definit | o o | special collector | mode friend | complet town | vast unstopp | mega man |
| Word 5 | shadow tomb | classic card | exclus extra | support local | french german | without graphic | resid evil |
| Word 6 | phone tablet | graphic simpl | find standard | mode multiplay | lost land | fuel vast | mobil suit |
| Word 7 | perfect team | level relax | full exclus | local coop | video edit | effect fuel | metal gear |
| Word 8 | rusti lake | wasd movement | extra won | fight friend | footbal manag | s entir | time ever |
| Word 9 | english german | develop develop | version collector | competit mode | swipe swipe | base without | assassin s |
| Word 10 | object charact | develop student | specif genr | friend use | emili s | word interact | ninja gaiden |

*Image 36: Top 10 unique bigrams per cluster.*

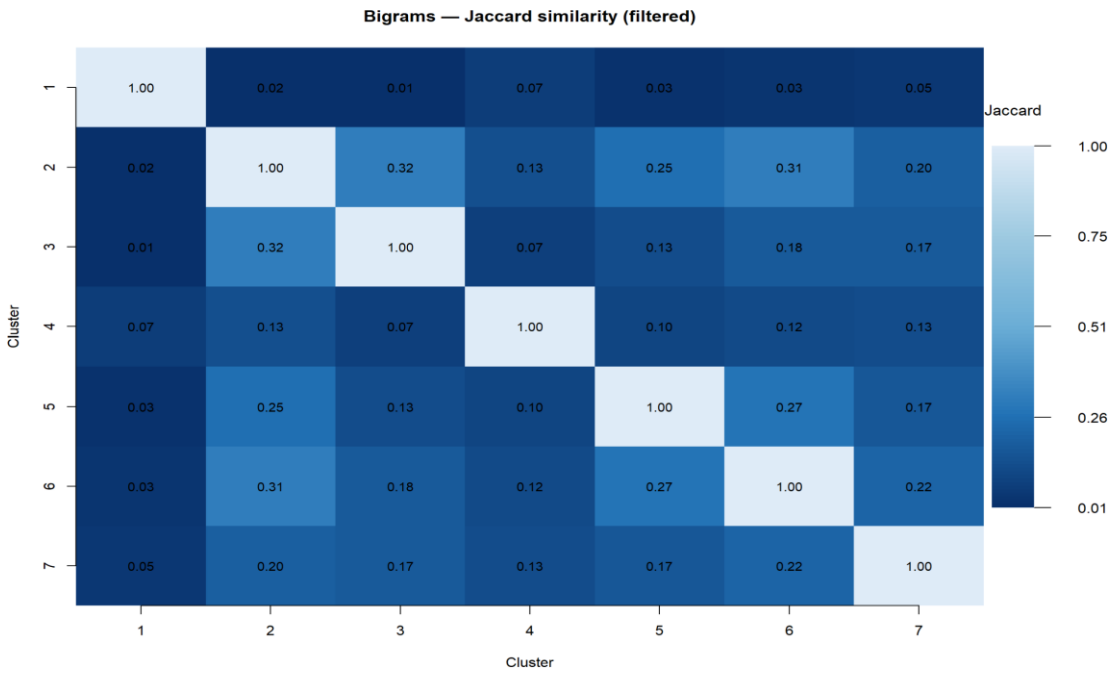


*Image 37: Jaccard's index for bigrams.*

# 12. R Code

```
---
output:
  pdf_document: default
  html_document: default
---
Reading File from desktop
```{r}
library(readxl)
df <- read_excel("C:/Users/fsitt/OneDrive/Desktop/steam.df.xlsx")
```
Getting rid of NA columns
```{r}
df <- as.data.frame(df)
# Deleting empty columns
df <- df[, colSums(!is.na(df)) > 0]
df <- df[, !grepl("^\\.\\.\\.", names(df))]
```
```{r}
# Fix shift columns from H
start_col <- 8
if (ncol(df) > start_col) {
  # Shifting column values
  df[, start_col:(ncol(df) - 1)] <- df[, (start_col + 1):ncol(df)]
  df[, ncol(df)] <- NA
  # Deleting last column if NA
  if (all(is.na(df[, ncol(df)]))) {
  df <- df[, -ncol(df)]
  }
}
```
Numerical Variables
```

```{r}
# Converting numerical variables
num_vars <- c(
  "Peak.CCU",
  "Required.age",
  "Price",
  "DiscountDLC.count",
  "Metacritic.score",
  "User.score",
  "Positive",
  "Negative",
  "Achievements",
  "Recommendations",
  "Average.playtime.forever",
  "Average.playtime.two.weeks",
  "Median.playtime.forever",
  "Median.playtime.two.weeks"
)
num_vars <- intersect(num_vars, names(df))
for (v in num_vars) {
  df[[v]] <- as.numeric(df[[v]])
}
```

Filter NA >30%

```{r}
# Calculate NA share (0-1) and remove columns with NA > 0.30
na_count <- colSums(is.na(df))
na_share <- na_count / nrow(df)
na_table <- data.frame(
  variable = names(na_share),
  na_share = as.numeric(na_share),
  stringsAsFactors = FALSE
```

```
)
THRESH_NA <- 0.30
keep_cols <- na_table$variable[na_table$na_share <= THRESH_NA]
drop_cols <- na_table$variable[na_table$na_share > THRESH_NA]
df <- df[, keep_cols, drop = FALSE]
# 2) Deduplication of AppID
if ("AppID" %in% names(df)) {
  df <- df[!duplicated(df$AppID), ]
}
df <- df[rowSums(!is.na(df)) > 0, ]
```

Estimated.owners.avg

```{r}
# Estimated owners -> owners_min / owners_max / estimated.owners.avg
x <- as.character(df[["Estimated owners"]])
x <- gsub(" ", "", x)
x <- gsub("-", "-", x)
x <- gsub("\\s+", "", x)
owners_min <- rep(NA_real_, length(x))
owners_max <- rep(NA_real_, length(x))
for (i in seq_along(x)) {
  if (!is.na(x[i]) && grepl("^[0-9]+-[0-9]+$", x[i])) {
    parts <- strsplit(x[i], "-", fixed = TRUE)[[1]]
    owners_min[i] <- as.numeric(parts[1])
    owners_max[i] <- as.numeric(parts[2])
  }
}
df$owners_min <- owners_min
df$owners_max <- owners_max
df$Estimated.owners.avg <- (df$owners_min + df$owners_max) / 2
```

Windows, Mac, Linux

```{r}
to01 <- function(x) {
  if (is.logical(x)) {
  return(as.integer(x))
  }
  if (is.numeric(x)) {
    return(as.integer(x != 0))
  }
  x <- tolower(trimws(as.character(x)))
  out <- rep(0, length(x))
  out[x %in% c("true", "t", "1", "yes", "y")] <- 1
  return(out)
}
if ("Windows" %in% names(df)) {
  df$Windows <- to01(df$Windows)
}
if ("Mac" %in% names(df)) {
  df$Mac <- to01(df$Mac)
}
if ("Linux" %in% names(df)) {
  df$Linux <- to01(df$Linux)
}
```
Device_support
```{r}
if (all(c("Windows", "Mac", "Linux") %in% names(df))) {
  df$device_support <-
    df$Windows +
    df$Mac +
    df$Linux
}
```

n_languages and n_categories

```{r}
count_items <- function(x) {
  x <- as.character(x)
  x[is.na(x)] <- ""
  counts <- rep(0, length(x))
  for (i in 1:length(x)) {
    if (x[i] != "") {
      parts <- strsplit(x[i], ",")[[1]]
      parts <- trimws(parts)
      parts <- parts[parts != ""]
      counts[i] <- length(parts)
    }
  }
  return(counts)
}
if ("Supported languages" %in% names(df)) {
  df$n_languages <- count_items(df[["Supported languages"]])
}
if ("Categories" %in% names(df)) {
  df$n_categories <- count_items(df$Categories)
}
```

Total_Reviews

```{r}
if (all(c("Positive", "Negative") %in% names(df))) {
  df$Total_Reviews <- df$Positive + df$Negative
  df$Review_Ratio <- NA_real_
  for (i in 1:nrow(df)) {
    if (!is.na(df$Total_Reviews[i]) &&
      df$Total_Reviews[i] > 0) {
      df$Review_Ratio[i] <-
```

```
      df$Positive[i] / df$Total_Reviews[i]
    }
  }
}
```

Final Cleaning

```{r}
df_new <- df[!is.na(df$Review_Ratio), ]
cat("Final rows:", nrow(df_new), "\n")
```

```{r}
# Pie charts: OS support (Windows / Mac / Linux)

pie_os <- function(x01, title_txt) {
  x01 <- as.integer(x01)
  x01[is.na(x01)] <- 0L
  yes <- sum(x01 == 1L)
  no  <- sum(x01 == 0L)
  vals <- c(yes, no)
  labs <- c(
    paste0("Yes (", round(100 * yes / sum(vals), 1), "%)"),
    paste0("No (",  round(100 * no  / sum(vals), 1), "%)")
  )
  pie(vals, labels = labs, main = title_txt)
}
if ("Windows" %in% names(df_new)) pie_os(df_new$Windows, "Windows Support Share")
if ("Mac" %in% names(df_new))     pie_os(df_new$Mac,     "Mac Support Share")
```

```
if ("Linux" %in% names(df_new))   pie_os(df_new$Linux,   "Linux Support Share")
```

```{r}
# Bar chart: games by achievements groups (0, 1-10, 11-25, 26-50, 51-100, >100)
if ("Achievements" %in% names(df_new)) {
  ach <- as.numeric(df_new$Achievements)
  ach[is.na(ach)] <- 0
  ach[ach < 0] <- 0
  ach_group <- cut(
  ach,
    breaks = c(-Inf, 0, 10, 25, 50, 100, Inf),
    labels = c("0", "1-10", "11-25", "26-50", "51-100", ">100"),
    right = TRUE
  )
  ach_counts <- table(ach_group)
  barplot(
    ach_counts,
    main = "Distribution of Games by Achievements Groups",
    xlab = "Achievements Groups",
    ylab = "Number of Games"
  )
}
```

```{r}
# Scatter plot: Peak CCU vs Estimated owners average + regression line
if (all(c("Peak CCU", "Estimated.owners.avg") %in% names(df_new))) {
x <- as.numeric(df_new$Estimated.owners.avg)
  y <- as.numeric(df_new[["Peak CCU"]])
  ok <- is.finite(x) & is.finite(y)
  x <- x[ok]
  y <- y[ok]
  plot(
```

```
  x, y,
  xlab = "Estimated Owners (Average)",
  ylab = "Peak Concurrent Users (Peak CCU)",
  main = "Peak CCU vs Estimated Owners (Average)"
 )
 abline(lm(y ~ x), col = "red", lwd = 2)
}
```

```{r}
## Clustering preparation
suppressPackageStartupMessages({
library(dplyr)
})
# Selecting clustering variables
vars_clust <- c("Total_Reviews", "device_support", "n_languages", "n_categories",
"Price")
stopifnot(exists("df_new"))
stopifnot(all(vars_clust %in% colnames(df_new)))
df_clust_vars <- df_new %>%
select(all_of(vars_clust)) %>%
 mutate(across(everything(), ~ ifelse(is.infinite(.x), NA, .x))) %>%
 na.omit()
# log1p
df_clust_transformed <- df_clust_vars %>%
 mutate(
  Total_Reviews = log1p(Total_Reviews),
  n_languages  = log1p(n_languages),
  Price        = log1p(Price)
 )
```

```
# scaling
df_clust_scaled <- as.data.frame(scale(df_clust_transformed))

# correlation matrix
cor_mat <- cor(df_clust_scaled, use = "pairwise.complete.obs", method = "pearson")
# Heatmap
x_order <- c("Total_Reviews", "device_support", "n_languages", "n_categories",
"Price")
y_order <- rev(x_order)
cor_plot <- cor_mat[y_order, x_order, drop = FALSE]
pal <- colorRampPalette(c("red", "white", "blue"))(200)
val_to_col <- function(v, pal) {
  v <- pmax(-1, pmin(1, v))
  idx <- round(((v + 1) / 2) * (length(pal) - 1)) + 1
  pal[idx]
}
op <- par(no.readonly = TRUE)
par(mar = c(7, 9, 4, 8) + 0.1)
image(
  x = 1:ncol(cor_plot),
  y = 1:nrow(cor_plot),
  z = t(cor_plot),
  axes = FALSE,
  col = pal,
  zlim = c(-1, 1),
  xlab = "",
  ylab = "",
  main = "Correlation matrix of clustering variables"
)
axis(1, at = 1:ncol(cor_plot), labels = colnames(cor_plot), las = 2)
axis(2, at = 1:nrow(cor_plot), labels = rownames(cor_plot), las = 1)
for (i in 0:ncol(cor_plot)) abline(v = i + 0.5, col = "white", lwd = 2)
```

```
for (i in 0:nrow(cor_plot)) abline(h = i + 0.5, col = "white", lwd = 2)
for (i in 1:nrow(cor_plot)) {
  for (j in 1:ncol(cor_plot)) {
    text(
      x = j,
      y = i,
      labels = sprintf("%.2f", cor_plot[i, j]),
      cex = 1
    )
  }
}

ticks <- c(-1, -0.5, 0, 0.5, 1)
legend(
  "right",
  inset = c(-0.20, 0),
  legend = ticks,
  fill = val_to_col(ticks, pal),
  title = "value",
  xpd = TRUE,
  bty = "n"
)
par(op)
```

```{r}
## K-means clustering
suppressPackageStartupMessages({
library(dplyr)
  library(cluster)
})
```

```
x_order <- c("Total_Reviews", "device_support", "n_languages", "n_categories", "Price")
df_clust_scaled <- df_clust_scaled[, x_order]
# Elbow method
set.seed(123)
k_vals <- 2:10
wss <- numeric(length(k_vals))
for (i in seq_along(k_vals)) {
  k <- k_vals[i]
  km <- kmeans(df_clust_scaled, centers = k, nstart = 25, iter.max = 100)
  wss[i] <- km$tot.withinss
}
op <- par(no.readonly = TRUE)
par(mar = c(5, 5, 3, 2) + 0.1)
plot(
  k_vals, wss,
  type = "o", pch = 16,
  xlab = "k", ylab = "",
  main = ""
)
par(op)
# Silhouette method on a sample of df
suppressPackageStartupMessages({
library(cluster)
})
stopifnot(exists("df_clust_scaled"))
stopifnot(is.data.frame(df_clust_scaled))
set.seed(123)
k_vals <- 2:10
N_SIL <- 5000
N_SIL <- min(N_SIL, nrow(df_clust_scaled))
idx_sil <- sample(seq_len(nrow(df_clust_scaled)), size = N_SIL, replace = FALSE)
```

```
X_sil <- df_clust_scaled[idx_sil, , drop = FALSE]
D_sil <- dist(X_sil)
sil_avg <- numeric(length(k_vals))

for (i in seq_along(k_vals)) {
  k <- k_vals[i]
  km <- kmeans(X_sil, centers = k, nstart = 25, iter.max = 100)
  sil <- silhouette(km$cluster, D_sil)
  sil_avg[i] <- mean(sil[, 3])
}
op <- par(no.readonly = TRUE)
par(mar = c(5, 5, 3, 2) + 0.1)
plot(
  k_vals, sil_avg,
  type = "o", pch = 16,
  xlab = "k", ylab = "",
  main = ""
)
par(op)
# K-means k=7
set.seed(123)
km7 <- kmeans(df_clust_scaled, centers = 7, nstart = 50, iter.max = 200)
vars_clust <- x_order
idx_used <- complete.cases(
  df_new[, vars_clust] %>%
    mutate(across(everything(), ~ ifelse(is.infinite(.x), NA, .x)))
)

df_clustered <- df_new[idx_used, ]
df_clustered$cluster_k7 <- km7$cluster
```

```{r}
# Centroids heatmap
centroids <- km7$centers[, x_order, drop = FALSE]
rn <- as.integer(rownames(centroids))
pick_max <- function(col, pool) pool[which.max(centroids[as.character(pool), col])]
pick_min <- function(col, pool) pool[which.min(centroids[as.character(pool), col])]
pool <- rn
cl1 <- pick_max("device_support", pool); pool <- setdiff(pool, cl1)
cl4 <- pick_max("n_categories", pool); pool <- setdiff(pool, cl4)
cl5 <- pick_max("n_languages", pool); pool <- setdiff(pool, cl5)
cl2 <- pick_min("Price", pool); pool <- setdiff(pool, cl2)
cl6 <- pick_min("device_support", pool); pool <- setdiff(pool, cl6)
cl7 <- pick_max("Total_Reviews", pool); pool <- setdiff(pool, cl7)
cl3 <- pool[1]
centroids_tesi <- centroids[as.character(c(cl1, cl2, cl3, cl4, cl5, cl6, cl7)), , drop = FALSE]
rownames(centroids_tesi) <- paste0("Cluster ", 1:7)

centroids_plot <- centroids_tesi[7:1, , drop = FALSE]
rownames(centroids_plot) <- paste0("Cluster ", 7:1)
pal_cent <- colorRampPalette(c("#fff7bc", "#fec44f", "#fe9929", "#ec7014", "#cc4c02", "#8c2d04"))(200)
zmax <- max(abs(centroids_plot))
val_to_col <- function(v, pal, zmax) {
v <- pmax(-zmax, pmin(zmax, v))
  idx <- round(((v + zmax) / (2 * zmax)) * (length(pal) - 1)) + 1
  pal[idx]
}
nr <- nrow(centroids_plot)
nc <- ncol(centroids_plot)
op <- par(no.readonly = TRUE)
par(mar = c(8, 7, 4, 2) + 0.1)
```

```
plot(
  NA,
  xlim = c(0.5, nc + 0.5),
  ylim = c(0.5, nr + 0.5),
  xaxt = "n", yaxt = "n",
  xlab = "", ylab = "",
  main = "Cluster centroids (standardized values)"
)

for (i in 1:nr) {
  for (j in 1:nc) {
    v <- centroids_plot[i, j]
    y <- nr - i + 1
    rect(
      xleft = j - 0.5, ybottom = y - 0.5,
      xright = j + 0.5, ytop = y + 0.5,
      col = val_to_col(v, pal_cent, zmax),
      border = NA
    )
    text(j, y, sprintf("%.2f", v), cex = 1)
  }
}
axis(1, at = 1:nc, labels = colnames(centroids_plot), las = 2)
axis(2, at = 1:nr, labels = rev(rownames(centroids_plot)), las = 1)
par(op)
```
```{r}
## df_clustered
suppressPackageStartupMessages({
library(dplyr)
})
stopifnot(exists("df_new"))
```

```
stopifnot(exists("km7"))
stopifnot(exists("x_order"))
vars_clust <- x_order
idx_used <- complete.cases(
df_new[, vars_clust] %>%
   mutate(across(everything(), ~ ifelse(is.infinite(.x), NA, .x)))
)
df_clustered <- df_new[idx_used, drop = FALSE]
stopifnot(nrow(df_clustered) == length(km7$cluster))
df_clustered$cluster_k7 <- as.character(km7$cluster)
centroids <- km7$centers[, vars_clust, drop = FALSE]
rn <- as.integer(rownames(centroids))
pick_max <- function(col, pool) pool[which.max(centroids[as.character(pool), col])]
pick_min <- function(col, pool) pool[which.min(centroids[as.character(pool), col])]
pool <- rn
cl1 <- pick_max("device_support", pool); pool <- setdiff(pool, cl1)
cl4 <- pick_max("n_categories", pool); pool <- setdiff(pool, cl4)
cl5 <- pick_max("n_languages", pool); pool <- setdiff(pool, cl5)
cl2 <- pick_min("Price", pool); pool <- setdiff(pool, cl2)
cl6 <- pick_max("device_support", pool); pool <- setdiff(pool, cl6)
cl7 <- pick_max("Total_Reviews", pool); pool <- setdiff(pool, cl7)
cl3 <- pool[1]

map_old_to_new <- setNames(
  object = as.character(1:7),
  nm     = as.character(c(cl1, cl2, cl3, cl4, cl5, cl6, cl7))
)
df_clustered$cluster_k7 <- map_old_to_new[df_clustered$cluster_k7]
stopifnot(!any(is.na(df_clustered$cluster_k7)))
df_clustered$cluster_k7 <- factor(df_clustered$cluster_k7, levels = as.character(1:7))
```

```{r}
## df_num and renaming variables
vars_rf <- c(
  "Total_Reviews",
  "device_support",
  "n_languages",
  "n_categories",
  "Price",
  "Positive",
  "Negative",
  "Review_Ratio",
  "Recommendations",
  "Peak.CCU",
  "Achievements",
  "Average.playtime.forever",
  "Median.playtime.forever",
  "Estimated.owners.avg",
  "DiscountDLC.count",
  "Metacritic.score",
  "User score"
)
df_num <- df_clustered[, c(vars_rf, "cluster_k7")]
df_num$cluster_k7 <- as.factor(df_num$cluster_k7)
names(df_num) <- make.names(names(df_num), unique = TRUE)
```

```{r}
## Train/Test split 70/30
suppressPackageStartupMessages({
library(caret)
})
set.seed(123)
```

```
train_index <- createDataPartition(df_num$cluster_k7, p = 0.7, list = FALSE)
train_data <- df_num[train_index, , drop = FALSE]
test_data  <- df_num[-train_index, , drop = FALSE]
```

```{r}
## Random Forest training (ntree=500)
suppressPackageStartupMessages({
library(randomForest)
})
set.seed(123)
p <- ncol(train_data) - 1
mtry_val <- floor(sqrt(p))
rf_model <- randomForest(
cluster_k7 ~ .,
  data = train_data,
  ntree = 500,
  mtry = mtry_val,
  importance = TRUE
)
```

```{r}
## OOB
suppressPackageStartupMessages({
  library(caret)
})
oob_err <- tail(rf_model$err.rate[, "OOB"], 1)
cat("OOB error:", round(oob_err * 100, 2), "%\n")
```

```{r}
```

```
## rf_pred and conf_matrix
suppressPackageStartupMessages(library(caret))
rf_pred <- predict(rf_model, newdata = test_data, type = "class")
conf_matrix <- confusionMatrix(rf_pred, test_data$cluster_k7)
print(conf_matrix)
```

```{r}
## Confusion Matrix plot
cm <- conf_matrix$table
op <- par(no.readonly = TRUE)
par(mar = c(6, 6, 4, 2) + 0.1)
pal <- colorRampPalette(c("white", "steelblue"))(100)
image(
  x = 1:ncol(cm),
  y = 1:nrow(cm),
  z = t(cm)[, nrow(cm):1],
  axes = FALSE,
  col = pal,
  xlab = "Predicted cluster",
  ylab = "True cluster",
  main = "Confusion Matrix – Random Forest (7 clusters)"
)
axis(1, at = 1:ncol(cm), labels = colnames(cm))
axis(2, at = 1:nrow(cm), labels = rev(rownames(cm)))
for (i in 0:ncol(cm)) abline(v = i + 0.5, col = "white")
for (i in 0:nrow(cm)) abline(h = i + 0.5, col = "white")
for (i in 1:nrow(cm)) {
  for (j in 1:ncol(cm)) {
    text(
      x = j,
      y = nrow(cm) - i + 1,
```

```
      labels = cm[i, j],
      cex = 0.9
    )
  }
}
par(op)
```

```{r}
## MeanDecreaseAccuracy + MeanDecreaseGini
varImpPlot(
  rf_model,
  main = "rf_model",
  n.var = min(20, nrow(importance(rf_model)))
)
```

```{r}
## Variable Importance
var_imp <- importance(rf_model, type = 1)
var_imp_df <- data.frame(
  Variable = rownames(var_imp),
  Importance = var_imp[, 1],
  row.names = NULL
)
TOP_N <- 30
var_imp_df <- var_imp_df[order(var_imp_df$Importance), ]
if (nrow(var_imp_df) > TOP_N) var_imp_df <- tail(var_imp_df, TOP_N)
op <- par(no.readonly = TRUE)
par(mar = c(5, 10, 4, 2) + 0.1)
barplot(
  var_imp_df$Importance,
```

```
  names.arg = var_imp_df$Variable,
  horiz = TRUE,
  las = 1,
  col = "steelblue",
  border = NA,
  main = "Variable Importance (Mean Decrease Accuracy)",
  xlab = "Importance"
)
par(op)
```

```{r}
# Libraries
pkgs <- c("dplyr", "tidyr", "iml")
to_install <- pkgs[!sapply(pkgs, requireNamespace, quietly = TRUE)]
if (length(to_install) > 0) install.packages(to_install)
suppressPackageStartupMessages({
  library(dplyr)
  library(tidyr)
  library(iml)
})
# setup df
df_work <- df_num %>%
  mutate(cluster_k7 = as.factor(cluster_k7)) %>%
  select(cluster_k7, where(is.numeric)) %>%
  tidyr::drop_na()
X_df <- df_work %>% select(-cluster_k7)
y <- df_work$cluster_k7
clusters <- levels(y)
cat("Rows used:", nrow(df_work), "\n")
cat("Predictors:", ncol(X_df), "\n")
```

```
cat("Clusters:", paste(clusters, collapse = ", "), "\n\n")
# setup folder
win_desktop <- function() {
  od <- Sys.getenv("OneDrive")
  if (!is.na(od) && od != "" && dir.exists(file.path(od, "Desktop"))) return(file.path(od,
"Desktop"))
  up <- Sys.getenv("USERPROFILE")
  if (!is.na(up) && up != "" && dir.exists(file.path(up, "Desktop"))) return(file.path(up,
"Desktop"))
  return(path.expand("~"))
}
base_dir <- win_desktop()
OUTDIR <- file.path(base_dir, "grafici rossoverdi")
dir.create(OUTDIR, showWarnings = FALSE, recursive = TRUE)
# test
probe <- file.path(OUTDIR, "__write_test__.txt")
ok_write <- FALSE
try({
  writeLines("ok", probe)
  ok_write <- file.exists(probe)
  if (ok_write) file.remove(probe)
}, silent = TRUE)
if (!ok_write) {
  up <- Sys.getenv("USERPROFILE")
  fallback <- if (!is.na(up) && up != "" && dir.exists(file.path(up, "Documents")))
    file.path(up, "Documents") else path.expand("~")
  OUTDIR <- file.path(fallback, "grafici rossoverdi")
  dir.create(OUTDIR, showWarnings = FALSE, recursive = TRUE)
  probe <- file.path(OUTDIR, "__write_test__.txt")
  writeLines("ok", probe)
  ok_write <- file.exists(probe)
  if (ok_write) file.remove(probe)
```

```
}
stopifnot(ok_write)

OUTDIR <- normalizePath(OUTDIR, winslash = "/", mustWork = TRUE)
```

```{r}
# Predictor one-vs-rest
make_predictor_class <- function(class_k, model, X_df, y_factor) {
  class_k <- as.character(class_k)
  pred_fun <- function(mod, newdata) {
    p <- predict(mod, newdata = newdata, type = "prob")
    if (!(class_k %in% colnames(p))) stop("Class ", class_k, " not in predict prob cols.")
    out <- as.numeric(p[, class_k])
    out[is.na(out)] <- 0
    data.frame(.prediction = out)
  }
  iml::Predictor$new(
    model = model,
    data = X_df,
    y = y_factor,
    predict.fun = pred_fun
  )
}
```

```{r}
# representative selection
pick_representative_median_like <- function(class_k, X_df, y_factor) {
  class_k <- as.character(class_k)
  idx <- which(as.character(y_factor) == class_k)
  Xk <- X_df[idx, , drop = FALSE]
```

```
  med <- apply(Xk, 2, median, na.rm = TRUE)
  d <- rowSums((sweep(Xk, 2, med, "-"))^2)
  idx[which.min(d)]
}
```

```{r}
# Extracting SHAP
extract_shap_results <- function(shap_obj) {
  r <- shap_obj$results
  r %>%
    transmute(feature = as.character(feature),
              phi = as.numeric(phi))
}
```

```{r}
# single plots
plot_shap_single_base <- function(res_df, x_row, top_n = 15, main_title = "") {
  res_df <- res_df %>%
    mutate(abs_phi = abs(phi)) %>%
    arrange(desc(abs_phi)) %>%
    head(top_n)
  x_vals <- as.list(x_row[1, , drop = TRUE])
  labels <- sapply(res_df$feature, function(v) {
  vv <- x_vals[[v]]
    if (is.null(vv)) return(v)
    if (is.numeric(vv)) {
      vv_fmt <- if (abs(vv) >= 1000)
        format(round(vv, 0), big.mark = ",", scientific = FALSE)
      else
        format(round(vv, 2), scientific = FALSE, trim = TRUE)
```

```
    } else {
      vv_fmt <- as.character(vv)
    }
    paste0(v, " = ", vv_fmt)
  })
  ord <- order(res_df$phi)
  res_df <- res_df[ord, , drop = FALSE]
  labels <- labels[ord]
  cols <- ifelse(res_df$phi >= 0, "#E15759", "#4E79A7")
  par(mar = c(5, 30, 4, 2), bg = "white")
  barplot(
    res_df$phi,
    names.arg = labels,
    horiz = TRUE,
    las = 1,
    col = cols,
    border = NA,
    cex.names = 0.85,
    main = main_title,
    xlab = "SHAP value (phi)"
  )
  abline(v = 0, col = "grey50", lwd = 1)
  grid(nx = NULL, ny = NULL, lty = 1, col = "grey90")
}
```

```{r}
# multi plots
plot_shap_multi_base <- function(agg_df, X_sample_df, top_n = 15, main_title = "") {
  agg_df <- agg_df %>%
    arrange(desc(mean_abs_phi)) %>%
    head(top_n)
```

```
  labels <- sapply(agg_df$feature, function(v) {
    if (!(v %in% colnames(X_sample_df))) return(v)
    vv <- mean(X_sample_df[[v]], na.rm = TRUE)
    vv_fmt <- if (abs(vv) >= 1000)
      format(round(vv, 0), big.mark = ",", scientific = FALSE)
    else
      format(round(vv, 2), scientific = FALSE, trim = TRUE)
    paste0(v, " = ", vv_fmt)
  })
  ord <- order(agg_df$mean_abs_phi)
  agg_df <- agg_df[ord, , drop = FALSE]
  labels <- labels[ord]
  par(mar = c(5, 34, 4, 2), bg = "white")
  barplot(
    agg_df$mean_abs_phi,
    names.arg = labels,
    horiz = TRUE,
    las = 1,
    col = "#59A14F",
    border = NA,
    cex.names = 0.85,
    main = main_title,
    xlab = "mean(|SHAP|) across 50 games"
  )
  grid(nx = NULL, ny = NULL, lty = 1, col = "grey90")
}
```

```{r}
# SHAP parameters
TOP_N <- 15
SAMPLE_SIZE_SINGLE <- 100
```

```
SAMPLE_SIZE_MULTI <- 100
N_PER_CLUSTER <- 50
set.seed(1)
single_top15_all <- list()
multi_top15_all  <- list()
```

```{r}
# SHAP analysis
for (k in clusters) {
  cat("====================================\n")
  cat("Cluster:", k, "\n")
  pred_k <- make_predictor_class(k, rf_model, X_df, y)
  # single SHAP
  rep_idx <- pick_representative_median_like(k, X_df, y)
  X_rep <- X_df[rep_idx, , drop = FALSE]
  cat(" SINGLE: representative row index =", rep_idx, "\n")
  shap_single <- iml::Shapley$new(
    pred_k,
    x.interest = X_rep,
    sample.size = SAMPLE_SIZE_SINGLE
  )
  single_df <- extract_shap_results(shap_single) %>%
    mutate(cluster = k, type = "single")
  f_single <- file.path(OUTDIR, paste0("shap_single_cluster_", k, ".png"))
  png(f_single, width = 1900, height = 1200, res = 160)
  plot_shap_single_base(
    res_df = single_df %>% select(feature, phi),
    x_row = X_rep,
    top_n = TOP_N,
    main_title = paste0("SHAP single-observation — Cluster ", k,
                        " (Top ", TOP_N, ")")
```

```
)
dev.off()
single_top15_all[[as.character(k)]] <-
single_df %>%
  mutate(abs_phi = abs(phi)) %>%
  arrange(desc(abs_phi)) %>%
  head(TOP_N) %>%
  mutate(rank = row_number())
# multi SHAP
idx_k <- which(as.character(y) == as.character(k))
sample_idx <- if (length(idx_k) < N_PER_CLUSTER)
  idx_k else sample(idx_k, N_PER_CLUSTER, replace = FALSE)
cat(" MULTI: sampling", length(sample_idx), "games\n")
X_sample_df <- X_df[sample_idx, , drop = FALSE]
phi_long <- vector("list", length(sample_idx))

for (i in seq_along(sample_idx)) {
  row_id <- sample_idx[i]
  x_i <- X_df[row_id, , drop = FALSE]
  shap_i <- iml::Shapley$new(
    pred_k,
    x.interest = x_i,
    sample.size = SAMPLE_SIZE_MULTI
  )
  df_i <- extract_shap_results(shap_i) %>%
  mutate(cluster = k, obs_row = row_id)
  phi_long[[i]] <- df_i
  if (i %% 10 == 0)
    cat("   computed", i, "/", length(sample_idx), "\n")
}
phi_long_df <- bind_rows(phi_long)
agg <- phi_long_df %>%
```

```
  group_by(cluster, feature) %>%
  summarise(
    mean_abs_phi = mean(abs(phi), na.rm = TRUE),
    mean_phi     = mean(phi, na.rm = TRUE),
    .groups = "drop"
  )
 f_multi <- file.path(OUTDIR, paste0("shap_multi50_cluster_", k, ".png"))
 png(f_multi, width = 1900, height = 1200, res = 160)
 plot_shap_multi_base(
  agg_df = agg,
  X_sample_df = X_sample_df,
  top_n = TOP_N,
  main_title = paste0("SHAP multi (n=50) — Cluster ", k,
                      " (Top ", TOP_N, " by mean(|phi|))")
 )
 dev.off()
 multi_top15_all[[as.character(k)]] <-
 agg %>%
 arrange(desc(mean_abs_phi)) %>%
 head(TOP_N) %>%
  mutate(rank = row_number())
 cat(" Saved:\n")
 cat("  -", f_single, "\n")
 cat("  -", f_multi, "\n")
}
```

```{r}
# export of files
single_out <- bind_rows(single_top15_all)
multi_out  <- bind_rows(multi_top15_all)
write.csv(
```

```
  single_out,
  file.path(OUTDIR, "shap_single_top15_all_clusters.csv"),
  row.names = FALSE
)
write.csv(
  multi_out,
  file.path(OUTDIR, "shap_multi_meanAbs_top15_all_clusters.csv"),
  row.names = FALSE
)
```

```{r}
# combined graph
while (dev.cur() > 1) dev.off()
win_desktop <- function() {
  od <- Sys.getenv("OneDrive")
  if (!is.na(od) && od != "" && dir.exists(file.path(od, "Desktop"))) return(file.path(od,
"Desktop"))
  up <- Sys.getenv("USERPROFILE")
  if (!is.na(up) && up != "" && dir.exists(file.path(up, "Desktop"))) return(file.path(up,
"Desktop"))
  path.expand("~")
}
DESKTOP <- win_desktop()
IN_DIR  <- normalizePath(file.path(DESKTOP, "grafici rossoverdi"), winslash = "/",
mustWork = TRUE)
single_csv <- file.path(IN_DIR, "shap_single_top15_all_clusters.csv")
multi_csv  <- file.path(IN_DIR, "shap_multi_meanAbs_top15_all_clusters.csv")
stopifnot(file.exists(single_csv), file.exists(multi_csv))
single_df <- read.csv(single_csv, stringsAsFactors = FALSE)
multi_df  <- read.csv(multi_csv,  stringsAsFactors = FALSE)
stopifnot(all(c("cluster","feature","phi") %in% names(single_df)))
```

```
stopifnot(all(c("cluster","feature","mean_abs_phi") %in% names(multi_df)))
single_df$cluster <- as.character(single_df$cluster)
multi_df$cluster  <- as.character(multi_df$cluster)
keep_k <- c("2","3","4","5","6")
single_k <- single_df[single_df$cluster %in% keep_k, , drop = FALSE]
multi_k  <- multi_df[multi_df$cluster  %in% keep_k, , drop = FALSE]
stopifnot(nrow(single_k) > 0, nrow(multi_k) > 0)

# top 15 multi and single
TOP_N <- 15
top_feat_single <- aggregate(abs(single_k$phi),
                             by = list(feature = single_k$feature),
                             FUN = mean, na.rm = TRUE)
top_feat_single <- top_feat_single[order(top_feat_single$x, decreasing = TRUE), ]
feat_single_top <- head(top_feat_single$feature, TOP_N)
top_feat_multi <- aggregate(multi_k$mean_abs_phi,
                            by = list(feature = multi_k$feature),
                            FUN = mean, na.rm = TRUE)
top_feat_multi <- top_feat_multi[order(top_feat_multi$x, decreasing = TRUE), ]
feat_multi_top <- head(top_feat_multi$feature, TOP_N)
# building matrixes
make_matrix <- function(df, value_col, features, keep_k) {
  M <- matrix(NA_real_,
              nrow = length(features),
              ncol = length(keep_k),
              dimnames = list(features, paste0("K", keep_k)))
  for (k in keep_k) {
    d_k <- df[df$cluster == k, , drop = FALSE]
    if (nrow(d_k) == 0) next
    agg <- tapply(d_k[[value_col]], d_k$feature,
                  function(v) v[which.max(abs(v))])
    common <- intersect(names(agg), features)
```

```
    M[common, paste0("K", k)] <- as.numeric(agg[common])
  }
  M
}

single_k$abs_phi <- abs(single_k$phi)
M_single <- make_matrix(single_k, "abs_phi", feat_single_top, keep_k)
M_multi  <- make_matrix(multi_k,  "mean_abs_phi", feat_multi_top, keep_k)
# normalizing
normalize_global_01 <- function(M) {
  v <- as.numeric(M)
  v <- v[is.finite(v)]
  if (length(v) == 0) return(M * 0)
  rng <- range(v, na.rm = TRUE)
  if (!is.finite(rng[1]) || !is.finite(rng[2]) || rng[1] == rng[2]) {
    return(M * 0)
  }
  (M - rng[1]) / (rng[2] - rng[1])
}
M_single_01 <- normalize_global_01(M_single)
M_multi_01  <- normalize_global_01(M_multi)
# Plot settings
plot_heatmap_base <- function(M01, main, palette_colors, legend_title, digits = 3) {
  pal <- grDevices::colorRampPalette(palette_colors)(100)
  layout(matrix(c(1,2), nrow = 1), widths = c(5.2,1.0))
  par(mar = c(7, 12, 5, 1), bg = "white")
  M2 <- M01
  M2[is.na(M2)] <- -0.01
  pal2 <- c(pal[1], pal)
  image(
    x = 1:ncol(M2),
    y = 1:nrow(M2),
```

```
  z = t(M2[nrow(M2):1, , drop = FALSE]),
  col = pal2,
  zlim = c(0,1),
  axes = FALSE,
  xlab = "",
  ylab = "",
  main = main
)
axis(1, at = 1:ncol(M2), labels = colnames(M2), las = 1, cex.axis = 0.95)
axis(2, at = 1:nrow(M2), labels = rev(rownames(M2)), las = 1, cex.axis = 0.95)
box()
# cell digits
for (i in 1:nrow(M01)) {
  for (j in 1:ncol(M01)) {
  val <- M01[i, j]
    if (is.finite(val)) {
      text(
        x = j,
        y = nrow(M01) - i + 1,
        labels = format(round(val, digits), nsmall = digits),
        cex = 0.9
      )
    }
  }
}
# legend
par(mar = c(7,2,5,4), bg = "white")
zleg <- matrix(seq(0,1,length.out = 100), nrow = 1, ncol = 100)
image(
  x = 1,
  y = seq(0,1,length.out = 100),
  z = zleg,
```

```
    col = pal,
    axes = FALSE,
    xlab = "",
    ylab = ""
  )
  axis(4, at = c(0,1), labels = c("0","1"), las = 1, cex.axis = 1.0)
  mtext(legend_title, side = 4, line = 2.3, cex = 0.95)
  layout(1)
}
# Saving
OUTDIR <- file.path(IN_DIR, "SHAP_HEATMAPS_FINAL")
dir.create(OUTDIR, showWarnings = FALSE, recursive = TRUE)
OUTDIR <- normalizePath(OUTDIR, winslash = "/", mustWork = TRUE)
f1 <- file.path(OUTDIR, "heatmap_single_K2-K6.png")
f2 <- file.path(OUTDIR, "heatmap_multi50_K2-K6.png")
png(f1, width = 2600, height = 1700, res = 180)
plot_heatmap_base(
  M01 = M_single_01,
  main = "SHAP Heatmap (Single observation) — K2–K6",
  palette_colors = c("#FCE5E5", "#F28E8E", "#B10000"),
  digits = 3
)
dev.off()
png(f2, width = 2600, height = 1700, res = 180)
plot_heatmap_base(
  M01 = M_multi_01,
  main = "SHAP Heatmap (Multi, n=50) — K2–K6",
  palette_colors = c("#E6F4E6", "#8FE6BF", "#0B6E0B"),
  digits = 3
)
dev.off()
```

```{r}
# TF-IDF and UMAP
# Libraries
pkgs <- c("dplyr", "text2vec", "irlba", "uwot", "ranger")
to_install <- pkgs[!sapply(pkgs, requireNamespace, quietly = TRUE)]
if (length(to_install) > 0) install.packages(to_install)
suppressPackageStartupMessages({
  library(dplyr)
  library(text2vec)
  library(irlba)
  library(uwot)
  library(ranger)
})
# text column
text_col <- if ("About the game" %in% names(df_clustered)) "About the game"
df_txt <- df_clustered %>%
  mutate(cluster_k7 = as.factor(cluster_k7))
df_txt$About_text <- as.character(df_txt[[text_col]])
df_txt$About_text[is.na(df_txt$About_text)] <- ""
df_txt <- df_txt[df_txt$About_text != "", drop = FALSE]
stopifnot(nrow(df_txt) > 1000)


# cleaning text
clean_text <- function(x) {
  x <- tolower(x)
  x <- gsub("[^a-z\\s]", " ", x)
  x <- gsub("\\s+", " ", x)
  x <- gsub("^\\s+|\\s+$", "", x)
  trimws(x)
```

```
}
df_txt$txt_clean <- clean_text(df_txt$About_text)
it <- itoken(df_txt$txt_clean, tokenizer = word_tokenizer, progressbar = TRUE)
stop_words_en <- c(

"a","about","above","after","again","against","all","am","an","and","any","are","as","at"
,
  "be","because","been","before","being","below","between","both","but","by","can","coul
  d","did","do","does","doing","down","during","each","few","for","from","fu rther",

"had","has","have","having","he","her","here","hers","herself","him","himself","his","h
ow",
  "i","if","in","into","is","it","its","itself","just","me","more","most","my","myself",

"no","nor","not","now","of","off","on","once","only","or","other","our","ours","ourselv
es",

"out","over","own","same","she","should","so","some","such","than","that","the","their
",

"theirs","them","themselves","then","there","these","they","this","those","through","to",

"too","under","until","up","very","was","we","were","what","when","where","which","
while",
"who","whom","why","with","would","you","your","yours","yourself","yourselves"
)
steam_common                                                    <-
c("game","games","player","players","experience","play","playing","steam")
stop_all <- unique(c(stop_words_en, steam_common))
vocab <- create_vocabulary(it, stopwords = stop_all)
```

```
vocab <- prune_vocabulary(
  vocab,
  term_count_min = 100,
  doc_proportion_max = 0.55,
  doc_proportion_min = 0.003
)
dtm <- create_dtm(it, vocab_vectorizer(vocab))
tfidf <- TfIdf$new(sublinear_tf = TRUE, smooth_idf = TRUE, norm = "l2")
dtm_tfidf <- tfidf$fit_transform(dtm)
```

```{r}
cat("DTM TF-IDF dims:", nrow(dtm_tfidf), "x", ncol(dtm_tfidf), "\n")
# embeddings 100 dimensions
set.seed(123)
K_EMB <- 100
svd_fit <- irlba::irlba(dtm_tfidf, nv = K_EMB, nu = K_EMB)
emb <- svd_fit$u %*% diag(svd_fit$d)
colnames(emb) <- paste0("emb_", seq_len(ncol(emb)))
# Umap plot
set.seed(123)
um <- uwot::umap(
  emb,
  n_neighbors = 15,
  min_dist = 0.10,
  metric = "cosine",
  verbose = TRUE
)
um1 <- um[,1]
um2 <- um[,2]
cl <- as.factor(df_txt$cluster_k7)
cols7 <- c("1"="#E41A1C","2"="#FFD92F","3"="#4DAF4A",
```

```
        "4"="#66C2A5","5"="#1F78B4",
        "6"="#6A3D9A","7"="#FF33CC")
col_vec <- cols7[as.character(cl)]
col_vec <- adjustcolor(col_vec, alpha.f = 0.35)
par(mar=c(5,5,4,2), bg="white")
plot(
  um1, um2,
  col = col_vec,
  pch = 16,
  cex = 0.55,
  xlab = "UMAP1",
  ylab = "UMAP2",
  main = "UMAP on TF-IDF embeddings (SVD=100)"
)
grid(col="grey90")
legend(
  "topright",
  legend = levels(cl),
  col = adjustcolor(cols7[levels(cl)], alpha.f=0.9),
  pch = 16,
  pt.cex = 1,
  bty = "n",
  title = "Cluster K=7"
)
# emb_df
emb_df <- as.data.frame(emb)
emb_df$cluster_k7 <- df_txt$cluster_k7
if ("AppID" %in% names(df_txt)) emb_df$AppID <- df_txt$AppID else emb_df$AppID
<- seq_len(nrow(emb_df))
emb_df <- emb_df[, c("AppID","cluster_k7", grep("^emb_", names(emb_df), value =
TRUE)), drop = FALSE]
# Hybrid RF
```

```
vars_num <- c(
  "Total_Reviews","device_support","n_languages","n_categories","Price",
  "Positive","Negative","Review_Ratio","Recommendations","Peak.CCU",
  "Achievements","Average.playtime.forever","Median.playtime.forever",
  "Estimated.owners.avg","DiscountDLC.count","Metacritic.score","User.score",
  "Year"
)
vars_num <- intersect(vars_num, names(df_txt))
df_num <- df_txt[, c(vars_num, "cluster_k7"), drop = FALSE] %>%
  mutate(across(all_of(vars_num), ~as.numeric(.x)))
keep <- complete.cases(df_num[, vars_num, drop = FALSE])
df_num <- df_num[keep, , drop = FALSE]
emb_keep <- emb[keep, , drop = FALSE]
df_hybrid2 <- cbind(df_num, as.data.frame(emb_keep))
set.seed(123)
NMAX <- 20000
if (nrow(df_hybrid2) > NMAX) {
  idx <- sample.int(nrow(df_hybrid2), NMAX)
  df_fast <- df_hybrid2[idx, , drop = FALSE]
} else {
  df_fast <- df_hybrid2
}
cat("\nHybrid fast rows:", nrow(df_fast), " | predictors:", ncol(df_fast)-1, "\n")
set.seed(123)
rf_fast <- ranger(
  cluster_k7 ~ .,
  data = df_fast,
  num.trees = 200,
  mtry = floor(sqrt(ncol(df_fast)-1)),
  importance = "impurity",
  oob.error = TRUE,
  classification = TRUE
```

```
)
cat("Hybrid FAST OOB error:", round(rf_fast$prediction.error * 100, 2), "%\n")
```

```{r}
# Libraries
pkgs <- c("dplyr", "uwot", "ranger", "Rtsne", "reticulate")
to_install <- pkgs[!sapply(pkgs, requireNamespace, quietly = TRUE)]
if (length(to_install) > 0) install.packages(to_install)
suppressPackageStartupMessages({
  library(dplyr)
  library(uwot)
  library(ranger)
  library(Rtsne)
  library(reticulate)
})
# text column
stopifnot(exists("df_clustered"))
stopifnot("cluster_k7" %in% names(df_clustered))
text_col <- if ("About the game" %in% names(df_clustered)) "About the game"
df_txt <- df_clustered %>%
  mutate(cluster_k7 = as.factor(cluster_k7))
df_txt$About_text <- as.character(df_txt[[text_col]])
df_txt$About_text[is.na(df_txt$About_text)] <- ""
df_txt <- df_txt[df_txt$About_text != "", drop = FALSE]
stopifnot(nrow(df_txt) > 1000)
if (!("AppID" %in% names(df_txt))) df_txt$AppID <- seq_len(nrow(df_txt))
# Python setup
ENV <- "r-sbert"
conda_ok <- TRUE
tryCatch({
reticulate::conda_binary()
```

```
}, error = function(e) {
  conda_ok <<- FALSE
})
if (!conda_ok) {
  reticulate::install_miniconda()
}
if (!reticulate::condaenv_exists(ENV)) {
  reticulate::conda_create(ENV)
}
reticulate::use_condaenv(ENV, required = TRUE)
reticulate::conda_install(
  ENV,
  packages = c("numpy", "torch", "sentence-transformers"),
  pip = TRUE
)
# SBERT
st <- reticulate::import("sentence_transformers", delay_load = FALSE)
model_name <- "all-MiniLM-L6-v2"
model <- st$SentenceTransformer(model_name)

# 4) SBERT embeddings
batch_size <- 64
texts <- df_txt$About_text
cat("\nEncoding SBERT... N:", length(texts), " | batch_size:", batch_size, "\n")
emb_sbert <- model$encode(
  texts,
  batch_size = as.integer(batch_size),
  show_progress_bar = TRUE
)
emb_sbert <- as.matrix(emb_sbert)
cat("SBERT embedding matrix:", nrow(emb_sbert), "x", ncol(emb_sbert), "\n")
stopifnot(ncol(emb_sbert) == 384)
```

```
colnames(emb_sbert) <- paste0("emb_", seq_len(ncol(emb_sbert)))
emb_df_sbert <- as.data.frame(emb_sbert)
emb_df_sbert$AppID <- df_txt$AppID
emb_df_sbert$cluster_k7 <- df_txt$cluster_k7
emb_cols <- grep("^emb_", names(emb_df_sbert), value = TRUE)
emb_df_sbert <- emb_df_sbert[, c("AppID", "cluster_k7", emb_cols), drop = FALSE]
# SBERT Umap
set.seed(123)
um_sbert <- uwot::umap(
  emb_sbert,
  n_neighbors = 15,
  min_dist = 0.10,
  metric = "cosine",
  verbose = TRUE
)
um1 <- um_sbert[,1]
um2 <- um_sbert[,2]
cl <- as.factor(df_txt$cluster_k7)
cols7 <- c("1"="#E41A1C","2"="#FFD92F","3"="#4DAF4A",
           "4"="#66C2A5","5"="#1F78B4",
           "6"="#6A3D9A","7"="#FF33CC")
col_vec <- adjustcolor(cols7[as.character(cl)], alpha.f = 0.35)
par(mar=c(5,5,4,2), bg="white")
plot(
  um1, um2,
  pch=16, cex=0.55,
  col=col_vec,
  xlab="UMAP1", ylab="UMAP2",
  main="UMAP on SBERT embeddings (all-MiniLM-L6-v2, 384D)"
)
grid(col="grey90")
legend(
```

```
  "topright",
  legend=levels(cl),
  col=adjustcolor(cols7[levels(cl)], alpha.f=0.9),
  pch=16, pt.cex=1, bty="n",
  title="Cluster K=7"
)
# Hybrid RF
vars_num <- c(
"Total_Reviews","device_support","n_languages","n_categories","Price",
"Positive","Negative","Review_Ratio","Recommendations","Peak.CCU",
"Achievements","Average.playtime.forever","Median.playtime.forever",
"Estimated.owners.avg","DiscountDLC.count","Metacritic.score","User.score",
"Year"
)
vars_num <- intersect(vars_num, names(df_txt))
df_num <- df_txt[, c(vars_num, "cluster_k7"), drop = FALSE] %>%
  mutate(across(all_of(vars_num), ~as.numeric(.x)))
keep <- complete.cases(df_num[, vars_num, drop = FALSE])
df_num <- df_num[keep, , drop = FALSE]
emb_keep <- emb_sbert[keep, , drop = FALSE]
df_hybrid_sbert <- cbind(df_num, as.data.frame(emb_keep))
set.seed(123)
NMAX <- 20000
if (nrow(df_hybrid_sbert) > NMAX) {
  idx <- sample.int(nrow(df_hybrid_sbert), NMAX)
  df_fast <- df_hybrid_sbert[idx, , drop = FALSE]
} else {
  df_fast <- df_hybrid_sbert
}
set.seed(123)
rf_sbert_fast <- ranger(
cluster_k7 ~ .,
```

```
  data = df_fast,
  num.trees = 500,
  mtry = floor(sqrt(ncol(df_fast)-1)),
  importance = "impurity",
  oob.error = TRUE,
  classification = TRUE
)
cat("Hybrid SBERT FAST OOB error:", round(rf_sbert_fast$prediction.error * 100, 2),
"%\n\n")
# t-SNE
X <- emb_sbert
X_scaled <- scale(X)
pca_init <- prcomp(X_scaled, rank. = 2)$x
perpl <- min(30, floor(nrow(X_scaled)/3) - 1)
if (perpl < 5) perpl <- 5
cat("t-SNE perplexity:", perpl, "\n")
set.seed(1)
ts <- Rtsne(
  X_scaled,
  dims = 2,
  perplexity = perpl,
  theta = 0.5,
  max_iter = 1000,
  pca = FALSE,
  Y_init = pca_init,
  check_duplicates = FALSE,
  verbose = TRUE
)
ts_df <- data.frame(
  tSNE1 = ts$Y[,1],
  tSNE2 = ts$Y[,2],
  cluster_k7 = cl
```

```
)
# Plot with outliers
par(mfrow=c(1,1), mar=c(5,5,4,2), bg="white")
plot(
  ts_df$tSNE1, ts_df$tSNE2,
  col=adjustcolor(cols7[as.character(ts_df$cluster_k7)], alpha.f = 0.35),
  pch=16, cex=0.55,
  xlab="t-SNE 1", ylab="t-SNE 2",
  main="t-SNE on SBERT embeddings (with outliers)"
)
grid(col="grey90")
legend(
  "topright",
  legend=levels(cl),
  col=adjustcolor(cols7[levels(cl)], alpha.f=0.9),
  pch=16, pt.cex=1, bty="n",
  title="cluster_k7"
)
# Outlier removal
q1_x <- quantile(ts_df$tSNE1, 0.25); q3_x <- quantile(ts_df$tSNE1, 0.75); iqr_x <- q3_x
- q1_x
q1_y <- quantile(ts_df$tSNE2, 0.25); q3_y <- quantile(ts_df$tSNE2, 0.75); iqr_y <- q3_y
- q1_y
keep_out <- ts_df$tSNE1 >= (q1_x - 3*iqr_x) &
          ts_df$tSNE1 <= (q3_x + 3*iqr_x) &
          ts_df$tSNE2 >= (q1_y - 3*iqr_y) &
          ts_df$tSNE2 <= (q3_y + 3*iqr_y)
# Plot without outliers
plot(
  ts_df$tSNE1[keep_out], ts_df$tSNE2[keep_out],
  col=adjustcolor(cols7[as.character(ts_df$cluster_k7)][keep_out], alpha.f = 0.35),
  pch=16, cex=0.55,
```

```
  xlab="t-SNE 1", ylab="t-SNE 2",
  main="t-SNE on SBERT embeddings (outliers removed)"
)
grid(col="grey90")
legend(
  "topright",
  legend=levels(cl),
  col=adjustcolor(cols7[levels(cl)], alpha.f=0.9),
  pch=16, pt.cex=1, bty="n",
  title="cluster_k7"
)
```

```{r text_setup}
# Text analysis setup (packages + helpers)
suppressPackageStartupMessages({
library(dplyr)
  library(SnowballC)
})
if (!requireNamespace("stopwords", quietly = TRUE)) install.packages("stopwords")
if (!requireNamespace("openxlsx", quietly = TRUE)) install.packages("openxlsx")
suppressPackageStartupMessages({
  library(openxlsx)
})
TOP_N <- 10
OVERLAP_PCT <- 0.80
MIN_FREQ_BI <- 25
jaccard_matrix <- function(sets, levels) {
  K <- length(levels)
  M <- matrix(0, nrow = K, ncol = K)
  for (i in seq_len(K)) {
    for (j in seq_len(K)) {
```

```
      a <- sets[[levels[i]]]
      b <- sets[[levels[j]]]
      inter <- length(intersect(a, b))
      uni <- length(union(a, b))
      M[i, j] <- ifelse(uni == 0, 0, inter / uni)
    }
  }
  rownames(M) <- levels
  colnames(M) <- levels
  M
}
plot_heatmap_base <- function(M, main, legend_title = "Jaccard") {
  op <- par(no.readonly = TRUE)
  on.exit(par(op), add = TRUE)
  par(mar = c(5, 5, 4, 7), bg = "white")
  cols <- colorRampPalette(c("#08306B", "#2171B5", "#6BAED6", "#9ECAE1",
"#DEEBF7"))(100)
  image(
    x = 1:ncol(M), y = 1:nrow(M),
    z = t(M[nrow(M):1, ]),
    col = cols, axes = FALSE,
    xlab = "Cluster", ylab = "Cluster",
    main = main
  )
  axis(1, at = 1:ncol(M), labels = colnames(M))
  axis(2, at = 1:nrow(M), labels = rev(rownames(M)))
  for (i in seq_len(nrow(M))) {
    for (j in seq_len(ncol(M))) {
      text(j, nrow(M) - i + 1, labels = sprintf("%.2f", M[i, j]), cex = 0.9)
    }
  }
  zlim <- range(M, na.rm = TRUE)
```

```
  yseq <- seq(0, 1, length.out = 5)
  par(new = TRUE)
  plot(0, 0, type = "n", axes = FALSE, xlab = "", ylab = "", xlim = c(0, 1), ylim = c(0, 1))
  x0 <- 1.05; y0 <- 0.15; w <- 0.05; h <- 0.70
  for (k in 1:100) rect(x0, y0 + (k-1)*h/100, x0 + w, y0 + k*h/100, col = cols[k], border =
NA)
  text(x0 + w/2, y0 + h + 0.07, legend_title, cex = 1.1, xpd = NA)
  for (tck in yseq) {
    yy <- y0 + tck*h
    segments(x0 + w, yy, x0 + w + 0.02, yy, xpd = NA)
    text(x0 + w + 0.07, yy, sprintf("%.2f", zlim[1] + tck*(zlim[2]-zlim[1])), xpd = NA)
  }
}
```

```{r}
plot_table_base <- function(mat, title = "") {
  op <- par(no.readonly = TRUE)
  on.exit(par(op), add = TRUE)
  par(mar = c(1, 1, 3, 1), bg = "white")
  plot.new()
  title(main = title, cex.main = 1.4, font.main = 2)
  tbl <- cbind(rownames(mat), mat)
  colnames(tbl)[1] <- ""
  nr <- nrow(tbl) + 1
  nc <- ncol(tbl)
  x_left <- 0.02; x_right <- 0.98
  y_bot <- 0.05; y_top <- 0.92
  xbreaks <- seq(x_left, x_right, length.out = nc + 1)
  ybreaks <- seq(y_bot, y_top, length.out = nr + 1)
  for (i in 1:nr) {
    for (j in 1:nc) {
```

```
    rect(xbreaks[j], ybreaks[nr - i + 1], xbreaks[j+1], ybreaks[nr - i + 2],
        border = "black", lwd = 1)
    lab <- if (i == 1) colnames(tbl)[j] else tbl[i-1, j]
    font <- if (i == 1) 2 else 1
    text((xbreaks[j] + xbreaks[j+1]) / 2,
        (ybreaks[nr - i + 1] + ybreaks[nr - i + 2]) / 2,
        labels = lab, cex = 1.0, font = font)
   }
  }
}
```

```{r unigrams_unique_and_jaccard}
# Unigrams
cluster_levels <- sort(unique(as.character(uni_filt$cluster)))
K <- length(cluster_levels)
unique_uni_terms <- df_uni %>% filter(df == 1) %>% pull(term)
unique_unigram_top10 <- uni_filt %>%
  filter(term %in% unique_uni_terms) %>%
  group_by(cluster) %>%
  slice_max(order_by = n, n = TOP_N, with_ties = FALSE) %>%
  ungroup() %>%
  arrange(as.integer(as.character(cluster)), desc(n))
uni_unique_mat <- matrix("", nrow = TOP_N, ncol = K)
colnames(uni_unique_mat) <- paste0("Cluster ", cluster_levels)
rownames(uni_unique_mat) <- paste0("Word ", seq_len(TOP_N))
for (j in seq_along(cluster_levels)) {
  cl <- cluster_levels[j]
  w <- unique_unigram_top10 %>% filter(as.character(cluster) == cl) %>% pull(term)
  if (length(w) > 0)
    uni_unique_mat[seq_len(min(TOP_N, length(w))), j] <- w[seq_len(min(TOP_N,
length(w)))]
```

```
}
plot_table_base(uni_unique_mat, title = "Unique Unigrams — Top 10 per cluster (df=1)")
unique_unigram_counts <- uni_filt %>%
  filter(term %in% unique_uni_terms) %>%
  group_by(cluster) %>%
  summarise(unique_unigrams = n_distinct(term), .groups = "drop") %>%
  arrange(as.integer(as.character(cluster)))
print(unique_unigram_counts)
uni_sets <- lapply(cluster_levels, function(cl)
unique(uni_filt$term[as.character(uni_filt$cluster) == cl]))
names(uni_sets) <- cluster_levels
M_uni <- jaccard_matrix(uni_sets, cluster_levels)
plot_heatmap_base(M_uni, main = "Unigrams — Jaccard similarity")
```

```{r bigrams_pipeline_unique_jaccard_and_excel}
# Bigrams
stopifnot(exists("df_clustered"))
stopifnot("cluster_k7" %in% names(df_clustered))
text_col <- if ("About the game" %in% names(df_clustered)) "About the game"
sw <- stopwords::stopwords("en")
sw_extra <- c("game","games","player","players","play","steam")
stop_set <- unique(c(sw, sw_extra))
clean_text_bi <- function(x){
  x <- tolower(x)
  x <- gsub("[^a-z\\s]", " ", x)
  x <- gsub("\\s+", " ", x)
  trimws(x)
}
make_bigrams <- function(tok){
  tok <- tok[tok != ""]
  if (length(tok) < 2) return(character(0))
```

```
  paste(tok[-length(tok)], tok[-1])
}
stem_bigram <- function(bi_vec){
  if (length(bi_vec) == 0) return(character(0))
  w1 <- sub(" .*", "", bi_vec)
  w2 <- sub(".* ", "", bi_vec)
  s1 <- SnowballC::wordStem(w1, language = "en")
  s2 <- SnowballC::wordStem(w2, language = "en")
  paste(s1, s2)
}
dfb <- df_clustered %>%
  transmute(cluster = as.factor(cluster_k7),
            text = as.character(.data[[text_col]]))
dfb$text[is.na(dfb$text)] <- ""
dfb <- dfb %>% filter(nchar(text) > 0)
K_bi <- length(unique(dfb$cluster))
presence_thr <- ceiling(OVERLAP_PCT * K_bi)
all_bigrams <- vector("list", nrow(dfb))
for (i in seq_len(nrow(dfb))) {
  tx <- clean_text_bi(dfb$text[i])
  tok <- unlist(strsplit(tx, "\\s+"))
  tok <- tok[!(tok %in% stop_set)]
  bi <- make_bigrams(tok)
  all_bigrams[[i]] <- stem_bigram(bi)
}
bi_long <- data.frame(
  cluster = rep(dfb$cluster, times = sapply(all_bigrams, length)),
  bigram_stem = unlist(all_bigrams, use.names = FALSE),
  stringsAsFactors = FALSE
) %>%
  filter(!is.na(bigram_stem), bigram_stem != "")
bigram_counts <- bi_long %>% count(cluster, bigram_stem, name = "n")
```

```
bigram_counts_min <- bigram_counts %>% filter(n >= MIN_FREQ_BI)
df_bi <- bigram_counts_min %>%
  distinct(cluster, bigram_stem) %>%
count(bigram_stem, name = "df")
bi_counts_f <- bigram_counts_min %>%
  inner_join(df_bi, by = "bigram_stem") %>%
  filter(df < presence_thr) %>%
  select(cluster, bigram_stem, n)
unique_bi_terms <- df_bi %>% filter(df == 1) %>% pull(bigram_stem)
unique_bigram_top10 <- bi_counts_f %>%
  filter(bigram_stem %in% unique_bi_terms) %>%
  group_by(cluster) %>%
  slice_max(order_by = n, n = TOP_N, with_ties = FALSE) %>%
  ungroup() %>%
  arrange(as.integer(as.character(cluster)), desc(n))
cluster_levels_bi <- sort(unique(as.character(bi_counts_f$cluster)))
K_show <- length(cluster_levels_bi)
bi_unique_mat <- matrix("", nrow = TOP_N, ncol = K_show)
colnames(bi_unique_mat) <- paste0("Cluster ", cluster_levels_bi)
rownames(bi_unique_mat) <- paste0("Word ", seq_len(TOP_N))
for (j in seq_along(cluster_levels_bi)) {
  cl <- cluster_levels_bi[j]
  w <- unique_bigram_top10 %>%
    filter(as.character(cluster) == cl) %>%
    pull(bigram_stem)
  if (length(w) > 0)
    bi_unique_mat[seq_len(min(TOP_N,    length(w))),    j]    <-    w[seq_len(min(TOP_N,
length(w)))]
}
plot_table_base(bi_unique_mat, title = "Unique Bigrams — Top 10 per cluster (df==1)")
unique_bigram_counts <- bi_counts_f %>%
  filter(bigram_stem %in% unique_bi_terms) %>%
```

```
  group_by(cluster) %>%
  summarise(unique_bigrams = n_distinct(bigram_stem), .groups = "drop") %>%
  arrange(as.integer(as.character(cluster)))
print(unique_bigram_counts)
bi_sets <- lapply(cluster_levels_bi, function(cl)
unique(bi_counts_f$bigram_stem[as.character(bi_counts_f$cluster) == cl]))
names(bi_sets) <- cluster_levels_bi
M_bi <- jaccard_matrix(bi_sets, cluster_levels_bi)
plot_heatmap_base(M_bi, main = "Bigrams — Jaccard similarity (filtered)")
# Excel export
out_xlsx <- file.path(getwd(), "Steam_Text_TopWords_UNIQUE.xlsx")
wb <- createWorkbook()
addWorksheet(wb, "Top10_Unique_Unigram")
writeData(wb, "Top10_Unique_Unigram", as.data.frame(uni_unique_mat))
addWorksheet(wb, "Top10_Unique_Bigram")
writeData(wb, "Top10_Unique_Bigram", as.data.frame(bi_unique_mat))
addWorksheet(wb, "Unique_Unigram_Long")
writeData(wb, "Unique_Unigram_Long", unique_unigram_top10)
addWorksheet(wb, "Unique_Bigram_Long")
writeData(wb, "Unique_Bigram_Long", unique_bigram_top10)
addWorksheet(wb, "Unique_Counts")
writeData(wb, "Unique_Counts",
          list(unigram_counts = unique_unigram_counts,
               bigram_counts  = unique_bigram_counts))
for (sh in names(wb)) {
  freezePane(wb, sh, firstRow = TRUE)
  setColWidths(wb, sh, cols = 1:50, widths = "auto")
}
saveWorkbook(wb, out_xlsx, overwrite = TRUE)
cat("Excel saved:", out_xlsx, "\n")
```