**Game Recommendation System**

**C964 – Computer Science Capstone**

**Western Governors University**

**By Diomy Gabasan Jr**

10/28/2025

LETTER OF TRANSMITTAL

Jeffery Schniel

Spigot Inc.

1234 Rainbow Rd

Ventura, California

Dear Mr. Schniel,

Since the start of our journey as a digital distribution platform, our company has committed to serving our customers at the highest capacity. In pursuit of this excellence, our company has offered digital goods at competitive prices, but this is not enough. Our platform is currently suffering from a growing challenge, an ever-expanding catalogue and limited game discovery for our users. For our company to take the next step forward, we need to implement a game recommendation system. The goal of this project is to enhance user engagement, increase conversion rates, and strengthen our competitive position within the digital gaming marketplace using machine learning in a game recommendation system.

As our catalog continues to expand, customers are increasingly challenged to discover games that align with their interests, play styles, and budgets. The absence of a personalized recommendation engine limits our ability to surface relevant titles and minimizes the value of our existing library. Competitors such as Steam, Epic, and GOG have demonstrated measurable success by leveraging machine learning–driven recommendation systems that tailor suggestions to individual user preferences and behavior.

The proposed system would analyze a combination of user activity data (purchases, playtime, and ratings) and game attributes (genre, price, popularity, and sentiment) to generate targeted recommendations. By guiding players toward games they are more likely to enjoy, we can, increase user retention through personalized engagement, sales and cross-title discovery by connecting players with underexposed titles, enhance user experience and platform stickiness, and provide valuable insights into player preferences for future content acquisition and marketing. Additionally, this recommendation system will provide Spigot Inc. with actionable insights into user preferences and game performance. This will lead to increased discoverability of games as well as higher conversion rates and valuable analytics that will lead to future content acquisition decisions.

The data used for this project will draw on publicly available sources derived from Steam’s game database. This dataset includes game name, genres, average playtime, price, and positive ratings. After cleaning and preprocessing the dataset, categorical data like genres are encoded and combined with numerical features like average playtime, price, and positive ratings to form a data set suited for clustering and similarity modeling.

This project will proceed in five key phases. Data preparation, where the dataset is cleaned, encoded, and scaled. Model development where k-Means clustering and cosine similarity analysis is used to identify groupings and generate recommendations. Interface design where an interactive JupyterLab prototype that features search, selection, and recommendation widgets. Evaluation and optimization where silhouette score and accuracy of the model is evaluated. Finally, reporting and presentation where the project is wrapped up with delivery of a final report as well as integration into Spigot Inc’s production environment.

The funding needed to create this system approaches approximately $35,000 with an additional $2,000-$3,000 of upkeep per year. The developer chosen for this project has multiple years of experience in developing data applications with machine learning. This developer also has a bachelor’s degree in computer science. With this funding and experience, I believe that the developer will create this project in a timely and efficient manner.

Thank you for your consideration. I look forward to your feedback and to the opportunity to advance this project in alignment with our strategic objectives.

Sincerely,

[Diomy Gabasan Jr, Project Lead]

# Part B: Project Proposal Plan

## Project Summary

As the company’s catalogue grows, so too does the players struggle to discover games that match their interest. With thousands of titles available, users often face decision fatigue, leading to lower engagement, shorter browsing situations and missed sales opportunities. This project aims to address these problems using machine learning. The recommendation engine will use content-based and clustering approaches to identify relationships between games through shared attributes such as genre, ratings, playtime, and pricing. By analyzing these relationships, the system will be able to recommend games that align with the user’s interest. The goal of the game recommendation system is to enhance Spigot Inc.’s distribution platform by serving personalized game suggestions to users based on preference, play style and purchasing patterns.

This project will produce many key deliverables. First of which is a cleaned and processed game dataset that includes genre, rating, playing, and pricing information. This dataset can be used for other uses but that is out of scope of current project. A machine learning model, K-Means clustering and a similarity-based recommendation engine will be developed and trained according to the dataset to deliver personalized game suggestions. Additionally, this project will produce an evaluation report detailing model performance metrics such as silhouette scores and interpretability findings. The finished application is an interactive JupyterLab prototype that allows users to search for titles and receive tailored recommendations. Also, this project will include a user guide that will direct the client to the correct way to access the web application and how to use it.

This project will provide Spigot Inc. with a powerful data driven tool to enhance user engagement, improve discoverability across a growing catalogue of digital goods, and drive measurable increases in sales and retention. By serving users personalized recommendations, it reduces decision fatigue and helps players find games that they are more likely to enjoy and play.

## 

## Data Summary

**Source and Collection of Data**

The source of the dataset that will be used for this project originates from Steam’s publicly available game catalog and internal data curated by Spigot Inc. The primary data file, steam.csv, contains game related information such as titles, genres, user ratings, average playtime and price. This data was sourced from open access repositories that compile data from Steam’s API and public listings. When the application launches, the data will be collected from Spigot Inc’s own internal systems for use in the machine learning algorithms.

**Data Processing and Management**

Data will be managed through the application development life cycle to ensure consistency, accuracy, and maintainability. Through the design phase, key data attributes will be identified and defined. These variables influence game similarity and user preferences. In the Development phase, the data undergoes cleaning and transformation. Genre strings will be split into lists and will be encoded using multi-label binarizer. Next, numerical data will be standardized using a robust scaler and then combined with the categorical data to create one dataset. Through the testing and evaluation phase, data integrity will be validated before model training, this makes sure that the data passed into the machine learning model will work. Then clusters and recommendation outputs will be cross checked to ensure logical groupings. In the maintenance phase a version-controlled data pipeline will be implemented. This allows for periodic refreshes, real time updates, and scalable data management.

**Justification and Data Quality**

The chosen dataset effectively meets the project’s needs because it includes both content-based features (genres, price) and behavioral indicators (ratings and playtime), which are essential in identifying player interests and generating relevant recommendations.

Data anomalies such as playtime outliers, unrealistic price values or missing genres will be detected and managed through statistical methods and logical filters. Outliers will be transformed using scaled data, missing data will be removed, and inconsistent genre tags will be standardized through preprocessing. This careful preparation of data will ensure that the resulting data supports accurate clustering, valid similarity scoring and interpretable results.

**Ethical and Legal Considerations**

The data set used does not pose any significant ethical or legal concerns. The dataset is publicly available data and does not contain any personal or identifiable user information. It only contains metadata such as titles, and aggregated review counts. For further integration with Spigot’s internal data such as user behavior logs or purchase history, appropriate privacy, and data safeguards according to GDPR and CCPA standards will be implemented to maintain user confidentiality.

## Implementation

This project will implement the CRISP-DM (Cross Industry Standard Process for Data Mining) methodology, a widely recognized framework in the industry for structuring data-driven projects. This methodology provides a systematic iterative approach that encompasses six key phases: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation and Deployment. This methodology ensures that the project aligns with business objectives, uses high quality data, and delivers actionable results.

**Business Understanding**

In this stage of implementation, project objectives such as improving user engagement, increasing game discoverability, and enhancing decision making for marketing and catalog management are defined. Success metrics such as user satisfaction, conversion rates and recommendation accuracy are established.

**Data Understanding**

In this stage of implementation, data from Steam and internal Spigot sources for game catalog data is acquired. Additionally exploratory data analysis is conducted here to identify trends, distributions and potential data quality issues such as missing values or outliers.

**Data Preparation**

The data in this stage will be cleaned and preprocessed by handling missing values and removing any inconsistencies. Categorical features such as genre will be encoded using multi-label binarization and numerical features such as price and playtime are scaled through a robust scaler. This scaling ensures that the numerical categories are weighted and contribute equitably to similarity computations.

**Modeling**

In this stage, the application of unsupervised learning methods will be used to structure and interpret game relationships. K-Means clustering algorithm groups games with similar features and creates clusters that form the basis of recommendation logic. Next, similarity scores will be computed within clusters using cosine similarity to identify the most relevant games for each user.

**Evaluation**

In this stage of implementation, cluster coherence and recommendation relevance will be evaluated using quantitative measures like silhouette scores. User testing will be conducted, and performance will be assessed.

**Deployment**

The recommendation system will be implemented using a JupyterLab based prototype. An interactive search and recommendation widget will allow users to search and select games to receive tailored suggestions. A pipeline for ongoing data updates, model retraining and performance monitoring will be established to maintain data and accuracy over time.

**Focus on Machine Learning Implementation**

This machine learning solution will leverage a hybrid approach combining clustering and similarity analysis. Clustering organizes the catalog into meaningful groups based on numerical and categorical data attributes, reducing computational complexity for recommendations. Within the cluster, similarity analysis will measure and identify games most relevant to the user selected title. This approach balances computational efficiency with interpretability and provides a scalable foundation for personalization enhancements.

## 

## Timeline

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Milestone or deliverable** | **Project Dependencies** | **Resources** | **Start and End Date** | **Duration** |
| Project Kickoff | N/A | Project manager, Business Analyst | 1/5/2026 – 1/12/2026 | 1 week |
| Data Acquisition & Exploration | Milestone 1 | Data Engineer, Data Analyst | 1/12/2026 – 1/20/2026 | 1 week |
| Data Cleaning & Preprocessing | Milestone 2 | Data Engineer, Data Analyst | 1/20/2026 – 1/26/2026 | 1 week |
| Feature Engineering & Scaling | Milestone 3 | Data Scientist | 1/26/2026 – 2/2/2026 | 1 week |
| Machine Learning & Model Development | Milestone 4 | Data Scientist | 2/2/2026 – 2/17/2026 | 2 weeks |
| Model Evaluation & Tuning | Milestone 5 | Data Scientist | 2/17/2026 – 2/24/2026 | 1 week |
| Prototype Recommendation System Deployment | Milestone 6 | Data Scientist, Software Engineer | 2/24/2026 – 3/3/2026 | 1 week |
| Interactive Widgets & User Interface | Milestone 7 | Software Engineer, UX Designer | 3/3/2026 – 3/17/2026 | 2 weeks |
| Testing & Quality Assurance | Milestone 8 | QA Analyst, Data Scientist | 3/17/2026 – 3/24/2026 | 1 week |
| Documentation & Final Delivery | Milestone 9 | Project Manager, Data Scientist, Data Engineer | 3/24/2026 – 3/31/2026 | 1 week |

## 

## Evaluation Plan

The evaluation of the Game Recommendation System will be conducted through a combination of verification during development and validation upon project completion to ensure both technical accuracy and business effectiveness. Throughout development, verification will be performed at each stage to confirm that the system functions as intended. For data acquisition and preprocessing, this involves performing integrity checks, as well as assessments of completeness, consistency, and formatting. Missing value analysis will also be conducted to ensure that the raw data is accurately processed. In the feature engineering and scaling stage, statistical validation and unit tests will verify that the one hot encoding of genres and scaling of numerical features have been correctly applied and reflect the underlying characteristics of games.

During the modeling stage of development, algorithmic correctness checks will ensure that clustering and similarity calculations operate within expected values. Silhouette scores for clusters and reasonable ranges for similarity values will be assessed during this phase. The functionality of interactive widgets, and user interface components will be verified through functional testing, ensuring that the search, selection and recommendation features respond accurately to user input.

Upon project completion, validation will assess whether the system meets business objectives and delivers meaningful recommendations. These recommendations will be evaluated using silhouette scores and cosine similarity scores.

## 

## Costs

**Hardware Costs:**

Development workstations with GPU: 2x $2,000

Cloud Computing Servers: $1,000

Total Hardware costs: $5,000

**Software Costs:**

Python & libraries: $0

Pycharm Professional ed. 2025.2.4: $300

Cloud Platform(AWS): $900

ML Tools/Visualization Software Licenses = $500

Total Software costs: $1,800

**Labor Costs(estimated)**

Project Manager: 40 hrs x $75/hr = $3,000

Data Engineer: 80 hrs x $60/hr = $4,800

Machine Learning Engineer: 120 hrs x $80/hr = $9,600

Front-End/UI Developer: 60 hrs x $60/hr = $3,600

Quality Assurance/Testing: 40 hrs x $50/hr = $2,000

DevOps/Deployment Specialist: 40 hrs x $70/hr = $2,800

Total Labor Costs(estimated): $25,800

**Environmment/Deployment/Maintenance costs**

Cloud Hosting: $600

Web App Hosting/SSL Certificates: $200

Monitoring, Logging and Maintenance: $1,200

Total Environment/Deployment/Maintenance Costs: $2,000

**Estimated Grand Total: $34,600**

# Part C: Application

**Submitted files:**

* CapstoneDGab.ipynb
* README.md
* environment.yml
* main.py
* steam.csv

**Submitted links:**

* <https://github.com/diojunior/WGU-Capstone/>
* <https://mybinder.org>
* <https://mybinder.org/v2/gh/diojunior/WGU-Capstone/HEAD?urlpath=%2Fdoc%2Ftree%2FCapstoneDGab.ipynb>

# Part D: Post-implementation Report

## Solution Summary

The core problem that Spigot Inc. faced as a digital distribution platform is that the service had limited ways to help users discover games that align with their interests. With a plethora of games in the catalogue across multiple genres, users have been overwhelmed, which has led to missed opportunities for both user satisfaction and sales. Without a personalized discovery system, users relied on third party top seller lists, or trending games, which did not account for the user’s preferences.

The proposed solution was a hybrid game recommendation system that combined unsupervised machine learning in clustering with similarity-based recommendations. By analyzing both categorical attributes such as genres, and numerical metrics such as positive ratings, play time and price, the system grouped games into meaningful clusters. Within the cluster, similarity calculations allowed the system to recommend games that were most relevant to the user’s selected title. The implementation leveraged scalable data preprocessing, feature encoding, and interactive widgets in a JupyterLab based environment to make recommendations easily accessible to end-users.

This application directly addressed the problem by enhancing the discovery experience for users. Instead of blindly browsing through large catalogs, users have received tailored suggestions according to their selection. This system also supports business objectives by increasing business engagement, retention, and sales. It has also provided Spigot Inc. with valuable customer insights into game popularity, trends within genres, and cluster characteristics.

## Data Summary

The data used for the development of the game recommendation system was sourced from a publicly available dataset of games distributed through the Steam Platform. This dataset contained comprehensive information about each game including title, genre, price, average playtime, and user ratings. This dataset was collected from the Steam storefront and community review records which provide the quantitative metrics and qualitative attributes needed to properly cluster and compare games. Since the dataset originated from a reputable public source, it provided a reliable foundation for building an accurate and representative recommendation model. Additionally, this dataset was combined with Spigot Inc. internal data to better represent the customer base of this platform.

Throughout the application development life cycle, the data was processed and managed to maintain integrity and consistency. During the design phase, relevant features were identified and subsequently extracted to form the basis of the model. In the development phase, missing values were removed, outliers were analyzed and scaled, and genres were split and then transformed using multi-label binarization to enable numerical processing. Numerical attributes such as ratings, playtime and price were scaled using a robust scaler to reduce the overall weight that these values would have in the algorithm since these values were massive in comparison to the transformed categorical numbers. The processed dataset was then combined and integrated into the k-Means clustering algorithm for use in training that machine learning model.

The data management practices used during the development of this project emphasized reproducibility, scalability, and maintainability. Version control was implemented for this dataset and code to ensure that any and all changes to data have been tracked and validated. During the maintenance stage, documentation was created to guide future updates and data refresh cycles. This allowed the system to adapt to new releases and changes to both Steam’s and Spigot Inc.’s catalog. This data set met the needs of the project since it contained a good blend of numerical and categorical features that are essential for clustering and similarity-based recommendations.

## Machine Learning

The machine learning method used in the project was K-Means clustering, an unsupervised learning algorithm designed to group data based on points of similarities in their features. In this context, K-Means was used to segment games into clusters with similar attributes such as genre, user ratings, average playtime, and price. The algorithm iteratively assigned each game to the nearest centroid and recalculated those centroids to minimize the distance between datapoints within the same cluster. This process created well defined groupings of games that shared common characteristics, providing a foundation for the generation of relevant data-driven recommendations. One example of this in the data was Tom Clancy’s Rainbow Six Vegas, this data points to a similarity of 0.999 to the game Tom Clancy’s Splinter Cell Double Agent. These games share a genre as well as a large amount of positive ratings at a price of less than $10. Another example is Prince of Persia: The Two Thrones; this title is most like Rune Classic as well as Prince of Persia: The Forgotten Sands, both of which share a high similarity score as well as sharing the same genres and comparable price points.

The model was developed through a structured workflow consistent with the CRISP-DM methodology. After the dataset was cleaned and prepared, categorical attributes (game genres) were encoded using a MultiLabelBinarizer to convert multi-genre strings into binary vectors, and numerical features (ratings, playtime, and price) were scaled using normalization to ensure that differences in magnitude did not bias the clustering results. The preprocessed dataset was then used to train the K-Means model, where the optimal number of clusters was determined experimentally through the silhouette score, a metric that measures the cohesion and separation of clusters. Once the clustering model was finalized, a cosine similarity approach was applied within each cluster to refine recommendations, ensuring that suggested games closely resembled the user’s selected title in terms of both genre and quantitative attributes.

The selection of K-Means clustering was justified by several factors. First, it is computationally efficient and scalable, making it suitable for large game catalogs such as those managed by Spigot Inc. Second, as an unsupervised method, it does not require labeled data—an important consideration given that user preference labels were not available in this project. Third, its output is interpretable: each cluster represents a segment of similar games, which provides both recommendation utility and business insight into market segmentation. The simplicity and adaptability of K-Means allowed for straightforward integration with the recommendation pipeline and supported real-time responsiveness within the interactive interface.

Overall, the development and use of K-Means clustering provided a practical and effective solution for grouping and recommending games. It not only enhanced the user experience by surfacing personalized suggestions but also enabled Spigot Inc. to gain deeper insights into their catalog structure and player engagement trends, supporting data-driven decision-making across the organization.

## 

## Validation

The machine learning algorithm implemented in the Game Recommendation system, K-Means clustering, belongs to the category of unsupervised learning. Unsupervised learning techniques are used when the data lacks predefined labels or target variables. This algorithm identifies inherent patterns within datasets. In this case, K-Means was used to automatically group games into clusters based on similarities in attributes such as genres, playtime, user ratings and price, without prior knowledge of which games are considered similar. These clusters formed the foundation for the recommendation system by enabling them to suggest games from the same clusters as the user’s selected title.

To evaluate the performance of this unsupervised model, an appropriate validation method used was the silhouette score. This score quantifies the quality of the cluster with values of -1 to +1 where values closer to +1 are closely related to its cluster, values closer to zero suggest overlapping clusters and values in the negative indicate probable misclassification. One example that comes from our data is that the average silhouette score for all eight clusters in the algorithm is 0.881. This indicates that there is strong cluster separation, low overlap, and well-defined feature relationships. Additionally, there is the variance of 0.95 for the strongest cluster and 0.40 for the weakest cluster. Together, this indicates that the overall model performs well but there is an uneven quality to some clusters.

The combination of a high overall silhouette average and moderate variance across clusters demonstrates that the K-Means model effectively captured the dominant structure of the dataset while also highlighting its complexity. Clusters with strong silhouette scores were validated as strong indicators of cohesive game categories, while lower scoring clusters identified ambiguous regions of the data set where certain characteristics blended. This result is valuable for assessing both the model performance and the recommendation logic and in this case, it serves to show that while the results were pretty good, they can be improved upon with further tuning of the algorithm.

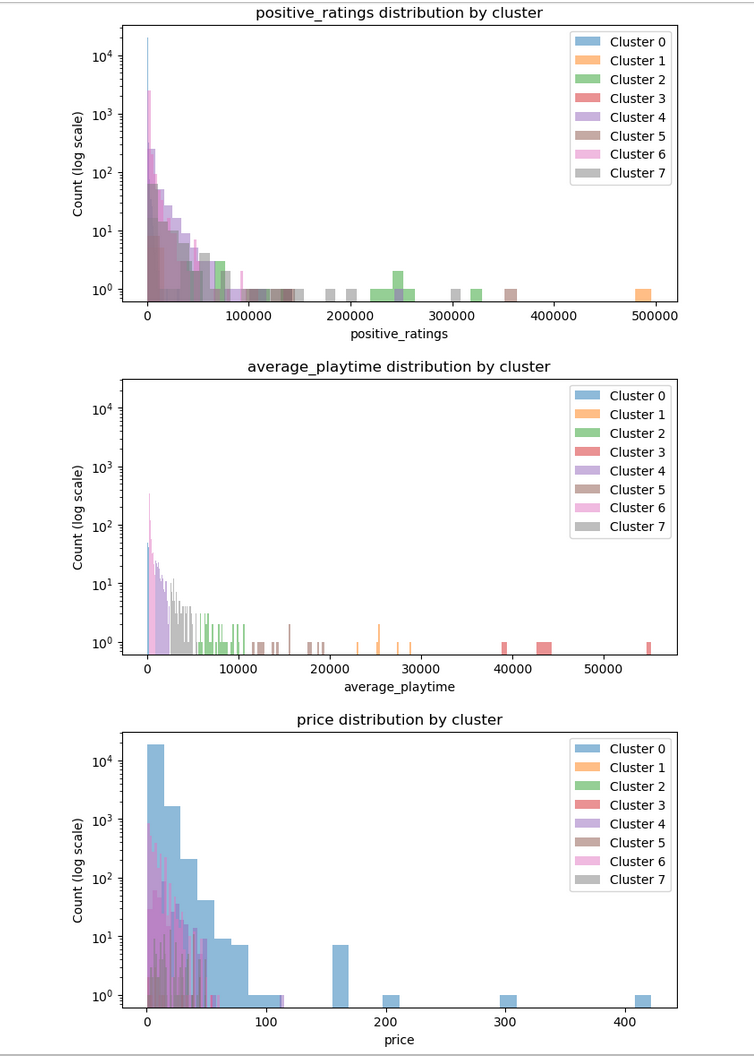
A screenshot of a computer program

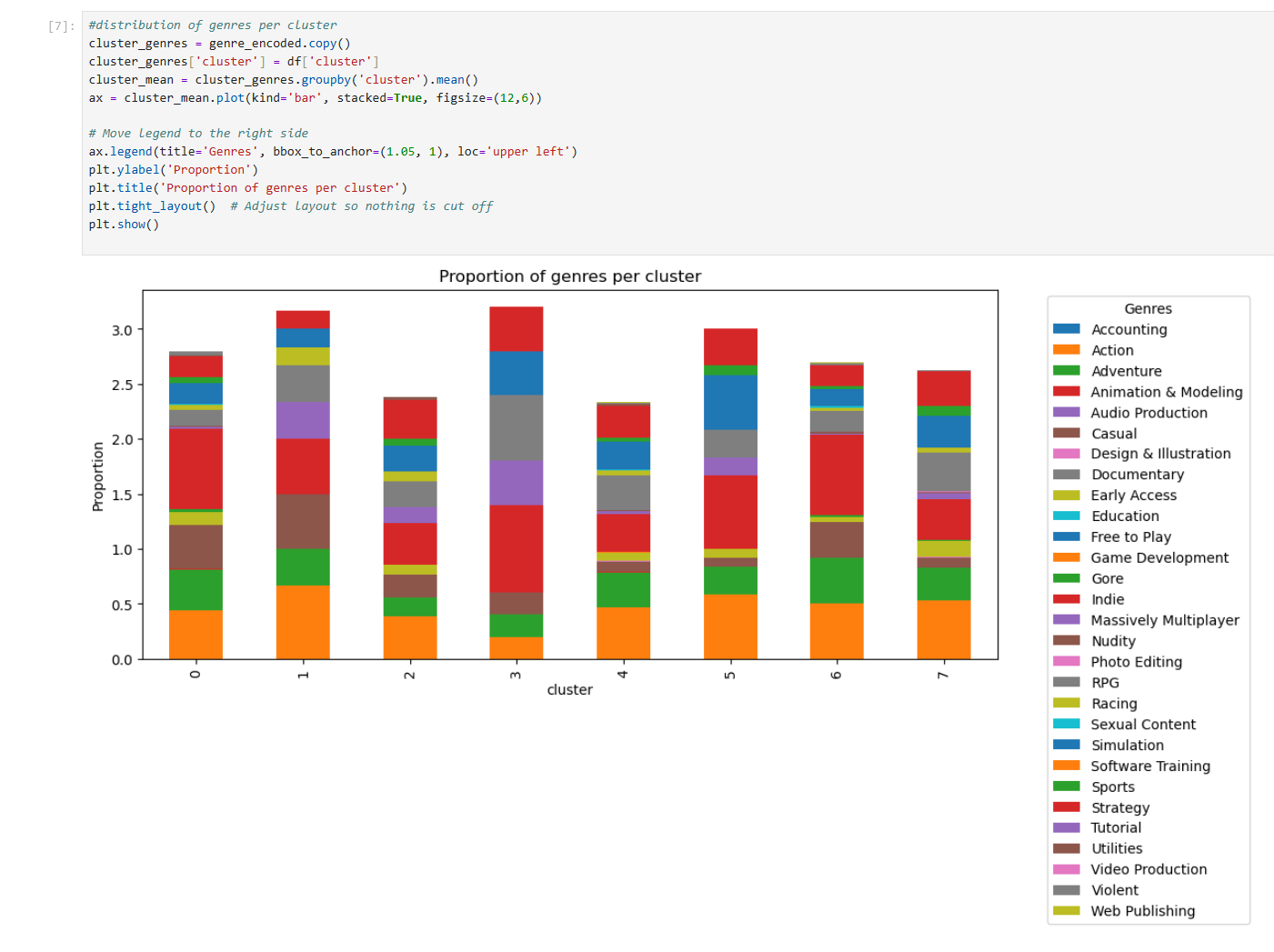
AI-generated content may be incorrect.

## Visualizations

## The visualizations are within the JupyterLab. I’ve also attached them here. The three graphs I chose to visualize the data in order are cluster visualization using PCA, a histogram featuring the numerical distributions by cluster, and a stacked bar graph that shows the proportions of genres per cluster.

## A screen shot of a computer screen AI-generated content may be incorrect..





## User Guide

The application needs 3 files to function. These files are CapstoneDGab.ipynb, environment.yml and steam.csv. The CapstoneDGab.ipynb is the main file that will be interacted with while the steam.csv is the data used in the main file. The environment.yml is used to initialize the main file using mybinder.org. There are no additional libraries needed since the service mybinder.org will set up the environment with the yml file.

Additionally included with the package is a README.md and a main.py. The README.md offers a shortcut to opening the web application using a clickable link while the main.py file contains the source code for CapstoneDGab.ipynb. There are two ways to start the application.

***Recommended way***

1. Open a web browser and go to <https://github.com/diojunior/WGU-Capstone>
2. Click on the README.md
3. Click on the “launch binder” icon and wait, this may take up to 2-3 minutes.
   1. If the webpage times out, you can refresh the webpage, and it should start the process again.
4. You will be redirected to a JupyterLab interface with the file CapstoneDGab.ipynb initialized.
5. Once initialized, click on the Run tab in the top left and use the option Run All Cells (this may take up to 5 minutes to run all cells at once)
   1. Alternatively, you can use shift + enter or use the option Run Selected Cell if you want to run a cell at a time.
6. Once all code has been run, there are visualizations to look at as well as a text box at the end.
   1. There is a circle in the top right corner that shows the status of the kernel. If it is filled in, it is busy, and the application is not ready for user input.
7. Please start typing in letters of a game in the text box, then select it from the drop box and hit the recommend button. There will be a list of recommendations based on the game selected in the drop box.
8. For example, type in cou in the search bar, Select Counter-Strike and click recommend.
9. To close the application, click Kernel in the top left, then Shut Down Kernel and then close the tab or browser.

***Alternative way***

1. Open a web browser and go to <https://mybinder.org>
2. In the first text field next to the dropdown menu that says “GitHub” paste in the link <https://github.com/diojunior/WGU-Capstone>
3. In the text field underneath “File to open (in JupyterLab)” paste in CapstoneDGab.ipynb
   1. If “File to open (in JupyterLab)” is not there, then select “URL” from the dropdown menu and switch it to “File”.
4. Click Launch and wait for the JupyterLab interface to load. The loading may take 2-3 minutes.
5. Once initialized, click on the Run tab and use the option Run All Cells (running the code may take up to 5 minutes to run all cells at once)
   1. Alternatively, you can shift + enter or use the option Run Selected Cell if you want to run a cell at a time.
6. Once all code has been run, there are visualizations to look at as well as a text box at the end.
   1. There is a circle in the top right corner that shows the status of the kernel. If it is filled in, it is busy, and the application is not ready for user input.
7. Please start typing in letters of a game in the text box, then select it from the drop box and hit the recommend button. There will be a list of recommendations based on the game selected in the drop box.
8. For example, type in grand, select Grand Theft Auto III and hit the recommend button.
9. To close the application, you should click Kernel, then Shut Down Kernel and then close the tab or browser.

# Reference Page

No references or sources were used in the making of this document