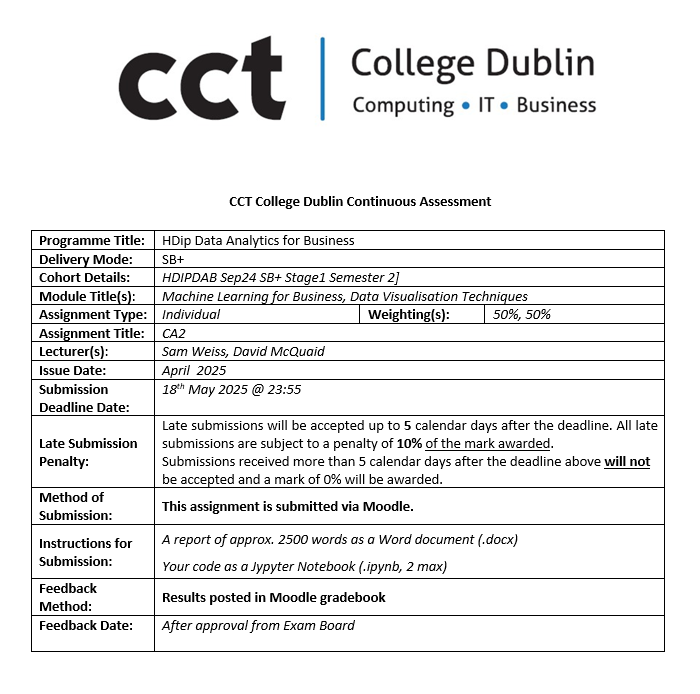
****

Contents

[Business Understanding 3](#_TOC_250008)

[Dataset Understanding 3](#_TOC_250005)

[Dataset Preparation 4](#_TOC_250004)

[Modeling 5](#_TOC_250006)

[Evaluation 7](#_TOC_250005)

[Clustering Evaluation 7](#_TOC_250005)

[Recommendation Evaluation 10](#_TOC_250005)

[Deployment 11](#_TOC_250005)

[Conclusion 12](#_TOC_250005)

[References 13](#_TOC_250000)

[GitHub Link 16](#_TOC_250000)

[Dashboard Link 16](#_TOC_250000)

**1. Business Understanding**

Nowadays, the entertainment industry in online streaming services and digital media platforms is being transformed by this next thing: personalization. Everybody has access to the vast majority of content, businesses must deliver tailored experiences to gain user interest and engagement, and as a result, the revenue will increase. The core of this project is leveraging machine learning (ML) to have a clustering of people and personalize recommendations with the help of collaborative filtering. With these two, businesses will be able to understand their audience and offer them the product that they are willing to buy.

Clustering makes groups based on patterns, behavioral patterns, such as average ratings or preferred genres. These data now send to the marketing team can help to create a specialized campaign for a specific demographic in mind. E.g a cluster detected by the algorithm, can let the team know if a Michael Myers movie will be accepted by them, avoiding unnecessary resources on unsuccessful campaigns. A business that has an understanding of its customers' clusters could design the perfect price for its product to sell.

Adding to this, Recommender systems are the key to retaining that user and becoming more satisfied. If a content is well recommended people, will be more likely to enjoy using this platform, increasing watching time, and the human brain tends to want to belong to what its constantly watching, if a person sees his favour star buying a soda, because of that sense of belonginess, he or she will most likely to try that soda and well. For that reason, collaborative filtering was employed. not only do we recommend things based on what the person does but capitalizing on the similarity between other users to generate tailored suggestions is what a business needs to do if wants to stay up with the recommending system game. This new user can help the former one with discovering hidden affinities, it's important to get the most data from the person to provide he or she the best thing one can consume to purely enjoy.

To reach real world usability, Thanks to Python a Streamlit dashboard was developed to a demographic of young adults between 18-35. This demographic is the core of every era, it’s the workforce and now in the era of the digital engagement this young adults generation not only will the ones using the platform but are the ones who are going to teach them their parents and their children, broading even more the target audience if done well. With this in mind the dashboard contains real time interaction, multimedia (Sound and video for every movie trailer), full of emoticons, minimalistic design, the always desired dark mode for this generation and lets bring an snippet, at the bottom of the page there is individual movie filterer that given a prompted movie prediction rating, the machine will generate and specific movie for each individual. This feedback received instantly, specific for each need, will make the user feel the dashboard alive, talking to him, caring for what he feels, and recommending what is best.

The project aims to take advantage of the new trend of machine learning businesses that make smarter decisions. Making sense of data, which is what data science and machine model do, could be as reliable as predicting what day is going to be tomorrow based on what day it's today.

**2. Data Understanding**

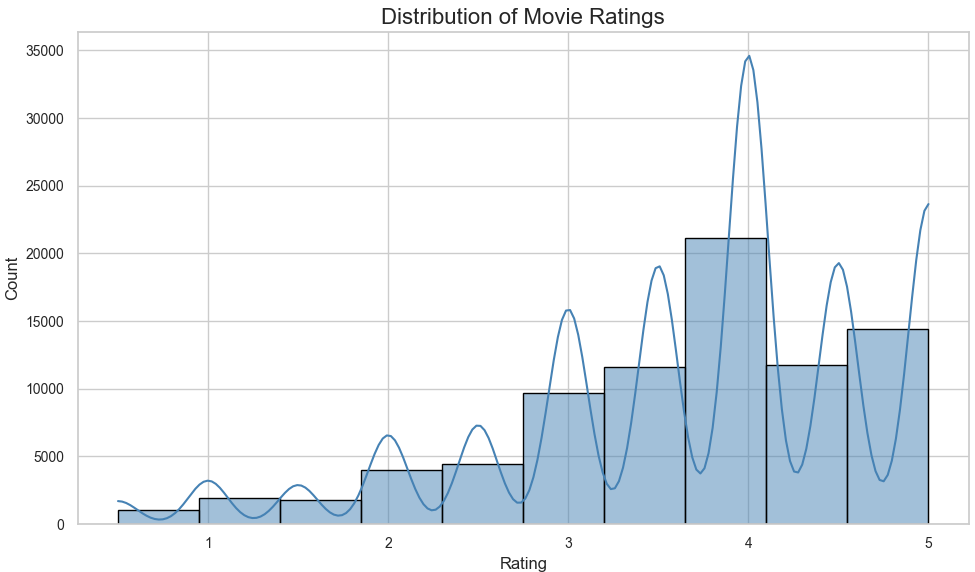
The base of this project lies on the following three datasets: movies.csv, rating.csv and tags.csv. The movies.csv contains data about films, (e.g: movieId, title, and genres). The rating.csv includes user-generated data like the ratings. From the beginning, these datasets were merged to facilitate analysis and the further modelling process

Inside this new merged dataset, it was found that the column genre has not only one genre per row but, normally, a variety of them; this latter was dealt with by splitting those entities and maintaining one

Understanding the structure of the tables is vital for the next steps. The starting analysis revealed over 80000 ratings provided by thousands of users on several thousand of movies. Ratings were typically centered around a mean, most likely above it than below it, 4, indicating favourable feedback. Later on was found that some users actually rated some movies either very high or very low.

Histograms and boxplots were more than needed in identifying these patterns. For example, the histogram of average ratings showed a normal data distribution, and the box plot showed the presence of some outliers in the data.

*Figure 1: Distribution of Movie ratings Chart*

**

One-hot encoding was used to help us categorize movie genres. In this case, we see that Genres like Thriller, Comedy, and Drama dominated the dataset.

**3. Data Preparation**

Data preparation involved cleaning, transforming, and engineering features

Thanks to this new merged data, the a basis for unsupervised learning (clustering) and collaborative filtering (recommendation). Then with the feature engineering process, the process began. The columns ‘movieId’, ‘avg\_rating’, and ‘rating\_count’ are the dataset now in order to align this study with the technical objectives to get the business insights. NaN were dropped because they did not alter the classification steps, and it was done to improve GPU resources, as mentioned before, we are dealing with an over 80000 entries dataset

Later on, I did some scaling and normalization. K-Means, Hierarchical Clustering, and Euclidean distance algorithms deal with data distance, meaning scaling helps tremendously in the process of getting the right results.

PCA was performed to the data, to create a profile, this was taken into consideration:

* AVG Rating
* Rating Count
* Movie genre

This combination worked perfectly for this task, as we will see later in this document, the features selected make sense to answer the business question.

Distribution charts supported this. Scatterplots of rating mean versus count revealed now user engagement clusters; clusters generated by the machine learning seemed natural and reliable.

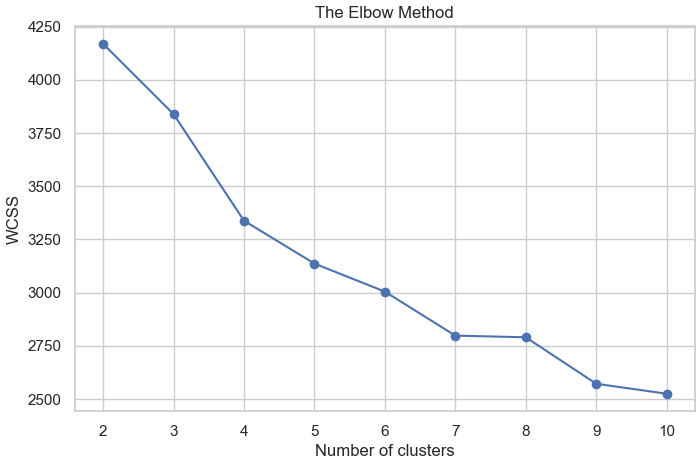
Finally, for the dashboard integration, Data was added to an app.py, and Nan values were dropped, leaving space for the required study to be done without any noise. This well-structured data helped both model accuracy and visualization clarity

**4. Modeling**

The modeling phase was focused on managing the multiple clusters altogether to compare which one would fit the data the best. For clustering, K-Means, Hierarchical Clustering, Fuzzy C-Means, and lastly, the DBSCAN with OPTICS were tested. The process was easy, , all of them were plotted and then compared visually, if the scatter plot did not make any sense, the clustering technique would be left out. The second round was then analyzing in the clusters in themselves, and if they aligned with our business question, we moved forward with them.

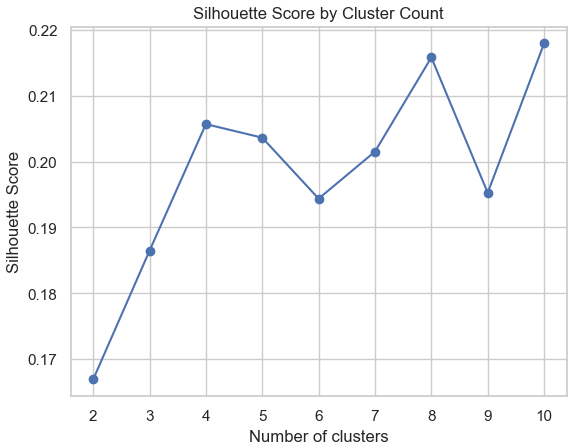
Thanks to the Elbow Method and the Silhouette Score, it was able to determine how many clusters (k) the data would need to keep moving forward

*Figure 2: The Elbow Method Results*

**

The Elbow Method was not very clear, as the pictures show, it was thanks to the Dendogram and the Silhouette Score that the result was given

*Figure 3: Silhouette Score Chart, showing 4 clusters*

**

**Silhouette Score:** The closest dot to 1 would mean, on that particular place, a good fit for a cluster. That said, I'm still picking 4 clusters from this chart.

K-Means clustering was an algorithm selected for fitting, chosen for its simplicity and scalability. I was plotted and compared after deciding how many clusters I would work with in this scenario

Hierarchical clustering was explored, and the dendrogram for this dataset showed clearer divisions supporting our four-cluster model. One advantage of hierarchical clustering is its interpretability; thanks to the dendrograms, a clear visual to select the number of clusters is given, helping tremendously in this business context

The Fuzzy C-Means clustering allows users to belong to different clusters, varying their membership in the group.

DBSCAN, later tuned with OPTICS (Ordering Points to Identify the Clustering Structure), was also tested. This approach does not require a number of clusters to be given; the clustering method will find it by itself, and it's amazing too for finding outliers.

On the contrary, it works with epsilons, similar to the Elbow Method, A graph is plotted to find the most suitable eps to tune the cluster.

The **recommendation modeling**, user-based **collaborative filtering,** was implemented using Euclidean distance as the similarity metric. The user-item rating Pivot Table formed the foundation, with each user's ratings compared to others. Cosine is a powerful tool, but Euclidean distance was chosen due to its superior performance.

The recommendations system works by averaging the rating from its five most similar users; this way of approach helps especially when the users have limited history (data available), helping with the cold-start problem.

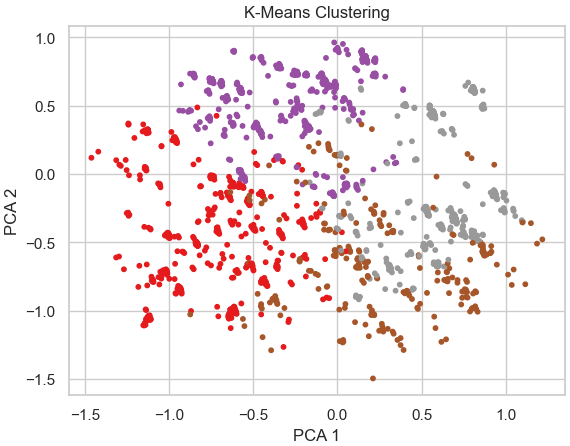
### 5. Evaluation

The evaluation phase assessed both the clustering and recommendation models using a combination of quantitative metrics and qualitative business relevance.

#### Clustering Evaluation

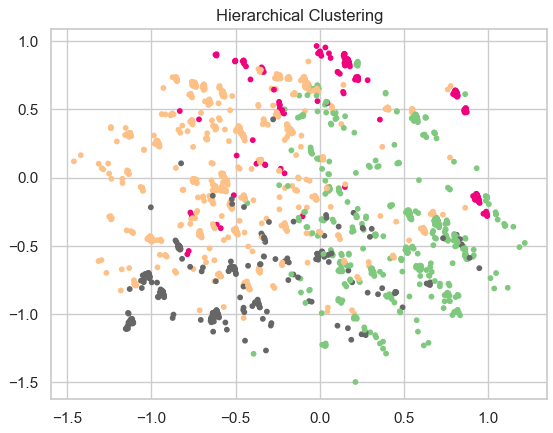
**K-Means Clustering:** A Simple clustering technique is the best-looking one and the one I will be using for this Movie recommendation and classification system. Dots seem to be homogeneous in their place, fitting the 4 clusters, still the Hierarchical falls very close to it. I was the best that held the PCA’d data and made sense out of it. Based on our 4-cluster solution, K-Means 4 clusters looked like the perfect fit.

*Figure 4: K-Means Clustering Results*

**

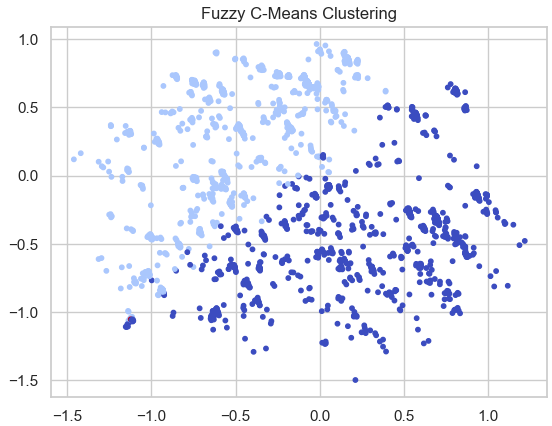
**Hierarchical Clustering:** It shows a good fit for this dataset. The chart looks mostly correct, but it leaves one with a feeling that it could be better. When reviewing the movies in the clusters I see that there is a significant large diversity of genres instead of having an specific one per cluster, that could be supposed as well when seeing the chart, there is the cluster there but sometimes dots seems a bit far away from its core point, making sense of what I just mentioned.

*Figure 5: Hierarchical Clustering Results*

**

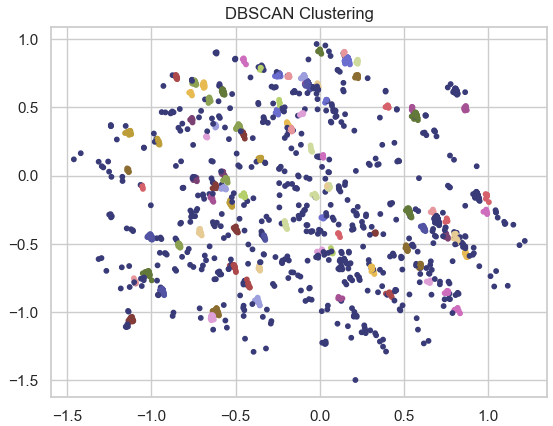
**Fuzzy C-Means Clustering:** Vaguely, it only created 3 clusters, and by looking at the chart, it is not the type of clustering that I am looking for right now. It made the clusters too widely spread and not suitable for this exercise; there were only 2 visible clusters at the time of plotting, being much less complex than the rest.

*Figure 6: Fuzzy C-Means Chart Results*

**

**DBSCAN Clustering:** After taking the time to fine-tune it, it created +90 clusters, and based on the chart, they are too widespread along the dataset, contrary to Fuzzy C-Means, this one found too many clusters, making it not suitable for this particular scenario. An Epsilon of 0.5 was found thanks to the K-distance Graph and the Reachability Plot made with the OPTICS algorithm showed that yes there are a good amount of values compressed at the 0.5 density benchmark but still showed that there is an important amount of data hanging at the 1.00 eps zone going parabolic at the end up to 2.00 eps, showcasing a good amount of unstructured values, basically too many outliers and noise.

*Figure 7: DBSCAN Clustering Results after PCA*

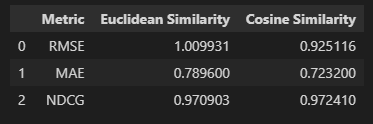
**

For all these reasons, K-Means Clustering is the preferred clustering technique for this study.

#### Recommendation Evaluation

As mentioned, two similarity metrics were evaluated: **Euclidean** and **Cosine**, across three core metrics:

*Figure 8: Recommendation Systems Evaluations*

**

RMSE, MAE, NDCG: For simplicity, let's say these metrics will work as our accuracy scores for the machine learning algorithm, and similar to them, the closer the number is to 1, the better it represents the actual data.

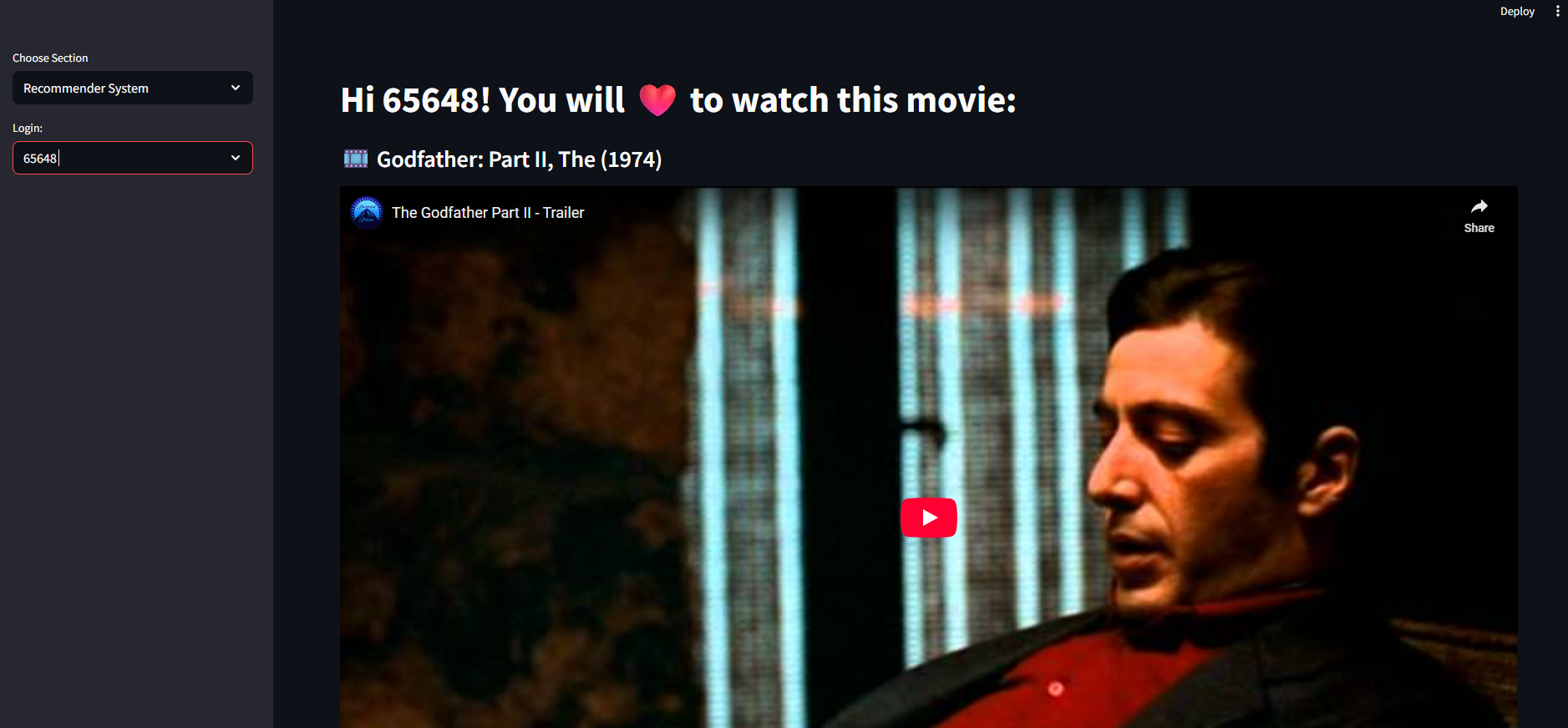
In this case, the winner by far was the Euclidean Similarity, with an RMSE of 1.009931, and not only was one metric above, but out of 2 showed favour to this similarity for this case.

**6. Deployment**

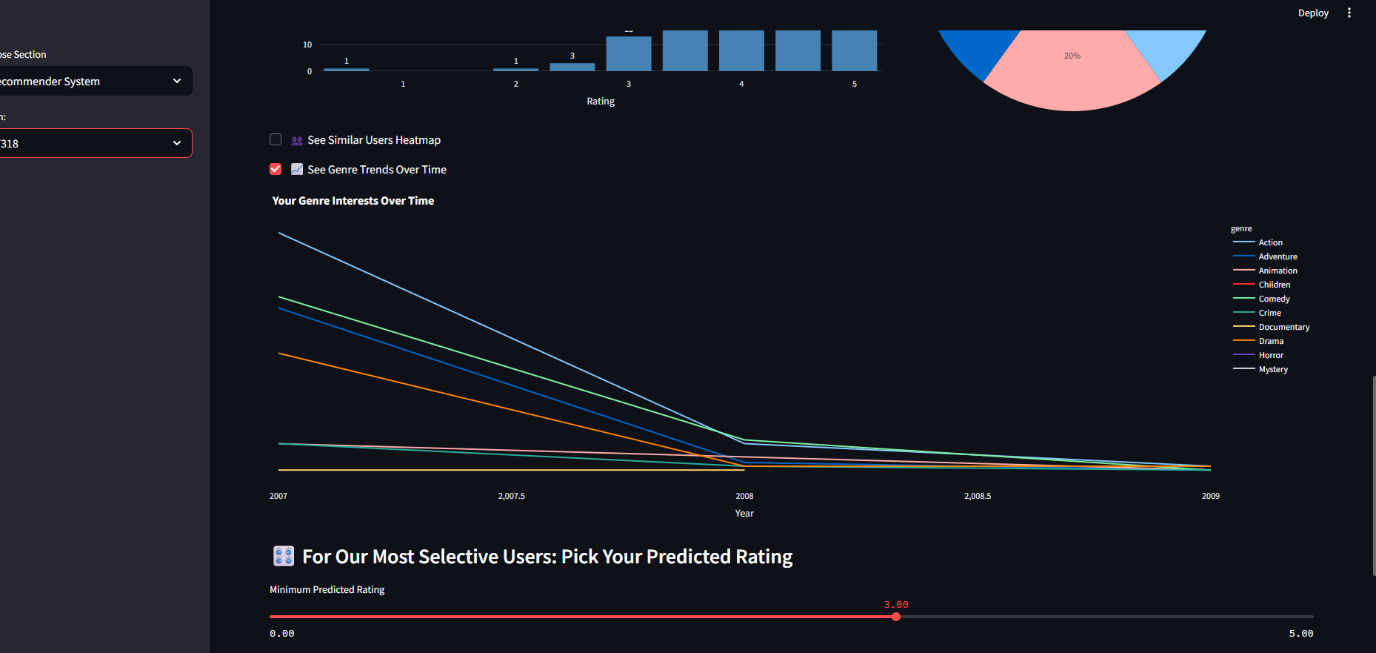
This phase was focused on how to make our Machine learning outputs accessible to the stakeholders and the users, ready to use. For that case, **Streamlit** was used, chosen for its simplicity; it’s a great tool for transforming any Python ML study into a web app or in this case, a dashboard.

The dashboard was built with the discussed demographic in mind. The UI was clean, responsive, and intuitive, including features like:

*Figure 9: Dashboard Snippet*

**

*Figure 10: Dashboard Second Image*



* **Login Interface**: This app makes it user-friendly for the users to let them know that everything that is being shown is a tailored response to their needs.
* **Top-5 movie recommendations**: Based on collaborative filtering, this is interactive, and it gets updated in real-time when the user changes the input
* **Movie trailers**: Multimedia elements that enrich the user experience especially with the demands of this demography.
* **Emoticons**: Emoticons, and everywhere
* **Dark Mode and Minimalistic design:** This demographic is too used to Netflix, Spotify, and all these dark theme platforms.
* **Hidden Sections:** not everything has to be shown on the page; this demographic is used to explore web pages and web apps, and they do not fear pressing one button or another. For that reason, there are 2 tabs hidden in plain sight, one of them shows an advanced data analytics chart, which is the heatmap. This heatmap shows the relationship between the user and its closest similar ones, in the future one cool feature would be letting the user to be able to click on the similar user and be able to “follow” them, to create a sense of community, to watch the same movies the other person watched, to be able to see the movies the other person did not like as well. This opens a new window of possibilities.
* **Desired prediction:** This last feature will allow the user to tell the algorithm, “Find me a movie that I would say I will rate with a 5.” And the algorithm will find that movie instantly; this type of instant feedback is the core to maintaining a user base engaged with a platform.
* **Analytics:** All graphs and Charts from the Jupyter notebook were added in a new section called Analytics, where stakeholders can see business-relevant data for this case study. Everything was styled with this demographic in mind, changing the actual background of the graph to black to fit the designed dashboard. The rationale behind these charts' selection can be found in more depth in the Jupyter Notebook in the EDA section. Approx. 300 words.

This dark blue-themed dashboard is hosted locally, but in the future, it can be hosted in any AMZ S3 bucket or something in that style.

**7. Conclusion**

This project showcases how machine learning, when strategically used and aligned with a business goal, can bring a technical masterpiece, which is to be able to predict what a person would like, and a commercial impact in sales revenues, if that prediction is accurate for the end user. In the era of TikTok and the fast-changing digital entertainment industry, a business needs to be able to find the trend they need to be on to maintain business presence and maintain sales. Not only is the data needed, but what to do with the data is key to achieving success. That’s why these types of case studies are so essential for any business. if a machine learning model is wrong, feature engineered, data is unclean, insights are vague, machine learning is wrongly developed, having a Google Data warehouse won't help you with your business problem. If the business does not treat the data well.

Using the CRISP-DM, from data understanding to real-time model deployment solution was made thanks to the principle made known before. “Knowing what to do”. Thanks to this, personalized insights were given to answer this particular business question

With a dual approach, having a clustering segmentation made with AI and then a collaborative filtering for content recommendation (Collaborative was essential, without this, suggestions might have been too generic and not personalized) this dashboard is a high-end tech solution for a young generation that starts to understand the power of AI in their daily lifes and the need of it, these types of solutions can only bring happy customers and happy customers will be willing to keep their happiness at any cost.

**8. References**

Abhinav Ajitsaria (2019). Build a Recommendation Engine With Collaborative Filtering. [online] Realpython.com. Available at: <https://realpython.com/build-recommendation-engine-collaborative-filtering/>.

Adithya Prasad Pandelu (2024). Day 10: Evaluation Metrics for Regression — MSE, MAE, RMSE, R2 Score. [online] Medium. Available at: <https://medium.com/@bhatadithya54764118/day-10-evaluation-metrics-for-regression-mse-mae-rmse-r%C2%B2-score-0ffb39e3ea26>.

Anon, (2020). How to Plot K-Means Clusters with Python? - AskPython. [online] Available at: <https://www.askpython.com/python/examples/plot-k-means-clusters-python>.

Arvai, K. (2020). K-Means Clustering in Python: A Practical Guide – Real Python. [online] realpython.com. Available at: <https://realpython.com/k-means-clustering-python/>.

Chugani, V. (2024). Understanding Euclidean Distance: From Theory to Practice. [online] Datacamp.com. Available at: <https://www.datacamp.com/tutorial/euclidean-distance>.

Coralogix. (2025). What is NDCG and How To Use It? [online] Available at: <https://coralogix.com/ai-blog/a-practical-guide-to-normalized-discounted-cumulative-gain-ndcg/>.

Data Science PM (2024). What Is CRISP DM? [online] Data Science Project Management. Available at: <https://www.datascience-pm.com/crisp-dm-2/>.

DataStax (2023). How to Implement Cosine Similarity in Python. [online] Medium. Available at: <https://datastax.medium.com/how-to-implement-cosine-similarity-in-python-505e8ec1d823>.

Dey, D. (2019). DBSCAN Clustering in ML | Density based clustering. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/>.

docs.scipy.org. (n.d.). Hierarchical clustering (scipy.cluster.hierarchy) — SciPy v1.9.1 Manual. [online] Available at: <https://docs.scipy.org/doc/scipy/reference/cluster.hierarchy.html>.

Fanilo Andrianasolo (2022). Crossfiltering in Streamlit • Reacting to Plotly Events. [online] YouTube. Available at: https://www.youtube.com/watch?v=htXgwEXwmNs [Accessed 14 May 2025].

GeeksforGeeks (2020). Matplotlib.colors.LinearSegmentedColormap class in Python. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/matplotlib-colors-linearsegmentedcolormap-class-in-python/>.

GeeksforGeeks. (2021a). SciPy - Cluster Hierarchy Dendrogram. [online] Available at: <https://www.geeksforgeeks.org/scipy-cluster-hierarchy-dendrogram/>.

GeeksforGeeks. (2021b). Using Plotly for Interactive Data Visualization in Python. [online] Available at: <https://www.geeksforgeeks.org/using-plotly-for-interactive-data-visualization-in-python/>.

Github.io. (2019). Time series analysis with pandas. [online] Available at: https://ourcodingclub.github.io/tutorials/pandas-time-series/ [Accessed 13 May 2025].

Gupta, A. (2019). Elbow Method for optimal value of k in KMeans. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/>.

Ibm.com. (2024). IBM Developer. [online] Available at: <https://developer.ibm.com/tutorials/awb-implement-hierarchical-clustering-python/>.

kaggle.com. (n.d.). Recommender Systems in Python 101. [online] Available at: <https://www.kaggle.com/code/gspmoreira/recommender-systems-in-python-101>.

KDnuggets. (n.d.). Implementing DBSCAN in Python. [online] Available at: <https://www.kdnuggets.com/2022/08/implementing-dbscan-python.html>.

Kumar, A. (2020). KMeans Silhouette Score Explained With Python Example - DZone AI. [online] dzone.com. Available at: <https://dzone.com/articles/kmeans-silhouette-score-explained-with-python-exam>.

Kumar, R. (2024). A Guide to the DBSCAN Clustering Algorithm. [online] Datacamp.com. Available at: <https://www.datacamp.com/tutorial/dbscan-clustering-algorithm>.

mhadhbi, N. (2021). Python Tutorial: Streamlit. [online] www.datacamp.com. Available at: <https://www.datacamp.com/tutorial/streamlit>.

Mizan, A. (2023). Learning About Streamlit Slider: How to Make One - Python in Plain English. [online] Medium. Available at: https://python.plainenglish.io/learning-about-streamlit-slider-how-to-make-one-a78314226a9f [Accessed 15 May 2025].

Mohantysandip (2020). A Step by Step approach to Solve DBSCAN Algorithms by tuning its hyper parameters. [online] Medium. Available at: <https://medium.com/@mohantysandip/a-step-by-step-approach-to-solve-dbscan-algorithms-by-tuning-its-hyper-parameters-93e693a91289>.

Plakalovic, A. (2023). Clustering Algorithms: DBSCAN vs. OPTICS. [online] Atlantbh Sarajevo. Available at: <https://www.atlantbh.com/clustering-algorithms-dbscan-vs-optics/>.

plotly (n.d.). Plotly Express. [online] plotly.com. Available at: <https://plotly.com/python/plotly-express/>.

Plotly.com. (2025). Dendrograms. [online] Available at: https://plotly.com/python/dendrogram/ [Accessed 110 May 2025].

prateekk94 (2020). Fuzzy C-Means Clustering on Iris Dataset. [online] Kaggle.com. Available at: https://www.kaggle.com/code/prateekk94/fuzzy-c-means-clustering-on-iris-dataset [Accessed 18 May 2025].

Pydata.org. (2019). How to handle time series data with ease — pandas 3.0.0.dev0+1662.g084b1999cf documentation. [online] Available at: <https://pandas.pydata.org/docs/dev/getting_started/intro_tutorials/09_timeseries.html>.

PyPI. (2022). youtube-search-python. [online] Available at: https://pypi.org/project/youtube-search-python/ [Accessed 15 May 2025].

PyPI. (2024). fuzzy-c-means. [online] Available at: https://pypi.org/project/fuzzy-c-means/ [Accessed 12 May 2025].

pythonhosted.org. (n.d.). Fuzzy c-means clustering — skfuzzy v0.2 docs. [online] Available at: <https://pythonhosted.org/scikit-fuzzy/auto_examples/plot_cmeans.html>.

scikit-learn. (2025). Demo of OPTICS clustering algorithm. [online] Available at: https://scikit-learn.org/stable/auto\_examples/cluster/plot\_optics.html [Accessed 12 May 2025].

Shizuya, Y. (2024). Detailed explanation about Collaborative filtering with Python examples. [online] Medium. Available at: <https://medium.com/@ichigo.v.gen12/detailed-explanation-about-collaborative-filtering-eab116e3b28b>.

Shrimanta Satpati (2023). Clustering by DBSCAN (Density-Based Spatial Clustering of Applications with Noise) Clearly Explained with Coding in Python. [online] Medium. Available at: <https://medium.com/@satpatishrimanta/clustering-by-dbscan-density-based-spatial-clustering-of-applications-with-noise-clearly-f93c5c72f706>.

Singh, Y. (2022). 3 Regression Metrics You Must Know: MAE, MSE, and RMSE. [online] Proclus Academy. Available at: <https://proclusacademy.com/blog/explainer/regression-metrics-you-must-know/>.

Streamlit (2021). How to create a horizontal bar chart with streamlit? [online] Streamlit. Available at: https://discuss.streamlit.io/t/how-to-create-a-horizontal-bar-chart-with-streamlit/18162 [Accessed 15 May 2025].

Streamlit (2022a). How to filter datafarme with python and streamlit. [online] Streamlit. Available at: https://discuss.streamlit.io/t/how-to-filter-datafarme-with-python-and-streamlit/30031/3 [Accessed 16 May 2025].

Streamlit (2022b). How to select a function with a checkbox and run it using a button in Streamlit? [online] Streamlit. Available at: https://discuss.streamlit.io/t/how-to-select-a-function-with-a-checkbox-and-run-it-using-a-button-in-streamlit/28786 [Accessed 15 May 2025].

Streamlit. (2024). Building a dashboard in Python using Streamlit. [online] Available at: <https://blog.streamlit.io/crafting-a-dashboard-app-in-python-using-streamlit/>.

Streamlit.io. (2024). Create an app - Streamlit Docs. [online] Available at: <https://docs.streamlit.io/get-started/tutorials/create-an-app>.

Streamlit.io. (2025a). st.checkbox - Streamlit Docs. [online] Available at: <https://docs.streamlit.io/develop/api-reference/widgets/st.checkbox>.

Streamlit.io. (2025b). Theming - Streamlit Docs. [online] Available at: https://docs.streamlit.io/develop/concepts/configuration/theming [Accessed 16 May 2025].

W3Schools (n.d.). Python Machine Learning - K-means. [online] www.w3schools.com. Available at: <https://www.w3schools.com/python/python_ml_k-means.asp>.

www.w3schools.com. (n.d.). Python Machine Learning - Hierarchical Clustering. [online] Available at: <https://www.w3schools.com/python/python_ml_hierarchial_clustering.asp>.

Matplotlib (n.d.). Choosing Colormaps in Matplotlib — Matplotlib 3.8.0 documentation. [online] matplotlib.org. Available at: <https://matplotlib.org/stable/users/explain/colors/colormaps.html>.

matplotlib.org. (n.d.). matplotlib.colors.LinearSegmentedColormap — Matplotlib 3.8.2 documentation. [online] Available at: <https://matplotlib.org/stable/api/_as_gen/matplotlib.colors.LinearSegmentedColormap.html>.

GeeksforGeeks. (2020). Python | How to make a terminal progress bar using tqdm. [online] Available at: <https://www.geeksforgeeks.org/python-how-to-make-a-terminal-progress-bar-using-tqdm/>.

Search, Y. (2020). Stack Overflow. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/62779030/youtube-search-on-python-3-8 [Accessed 20 May 2025].

**9. GitHub Link**

<https://github.com/CCT-Dublin/machine-learning-for-business-ca2-Gabriel-studies>

**10. Dashboard Link**

<https://dashboard-ca3.streamlit.app/>