Reconstructing the History of Music Recommendation: Content-Based, Collaborative and Neural Approaches

2025

Andra Mihaela Andruță
andra-mihaela.andruta@s.unibuc.ro

Ioana Alexandra Tunaru
ioana-alexandra.tunaru@s.unibuc.ro

Abstract

This project reconstructs the historical evolution of music recommendation systems by implementing three major paradigms: **Content-Based Filtering, Collaborative Filtering and Neural Contextual Models**.

Using the “Kaggle 30,000 Spotify Songs" dataset, we reproduce the core ideas that defined each technological era: similarity-based recommendations using acoustic features, matrix-factorization models based on collective user behavior and modern embedding-based approaches inspired by Word2Vec.

Our goal is to understand how each model operates, to contrast the strengths and limitations of symbolic, statistical and neural methods, also to illustrate how the recommendations for the same song vary across paradigms. The project offers a unified, hands-on reconstruction of the historical development of intelligent music recommendation systems.

1 Introduction

Recommender systems have become a fundamental component of modern digital platforms, shaping how users discover music, movies, products and information.

In the context of music streaming, recommendation technology evolved dramatically—from **early content-based approaches** relying on acoustic features, to **statistical models** using collective user behavior, and finally to **neural methods** that learn contextual representations from large-scale playlists.

The goal of our project is to reconstruct this historical trajectory by implementing representative models from each major era and analyzing how their underlying assumptions influence the recommendations they generate.

The central problem we aim to explore is how different paradigms interpret musical similarity and user preference. By applying the same dataset to three distinct methodologies— *Content-Based Filtering, Collaborative Filtering, and Neural Item2Vec embeddings*—we show how the meaning of “similar music” changes depending on the algorithmic logic used.

Contributions

Our individual contributions to the project are summarized as follows:

- **Andra Mihaela Andruță:** Dataset preprocessing, exploratory data analysis (EDA), implementation of the Content-Based Filtering model, PCA visualizations and Word Cloud analysis, Word2Vec embedding training and semantic playlist transition analysis.

- **Ioana Alexandra Tunaru:** Simulation of user–song interactions + generation of playlist corpora for the Item2Vec model, implementation of the CF stage (user–item matrix, centering and SVD-based latent factor analysis), personalized top-N recommendations using latent factors, Venn study and comparative study: CBF vs. BF.

Summary of the Approach

Our methodology mirrors the historical evolution of music recommendation systems. First, we rebuilt an early content-based system that computes cosine similarity between acoustic features such as *danceability*, *energy*, *valence*, *tempo* and more.

Next, we simulated user behavior and used matrix factorization (SVD) to approximate the statistical recommender systems popularized by the Netflix Prize.

Finally, we trained a Word2Vec-based model on synthetic playlists, reproducing the principles of the Item2Vec architecture developed by Spotify Research.

Motivation

We chose this project because it allowed us to explore the interdisciplinary nature of recommender systems—combining machine learning and neural representation learning. Rebuilding these models from scratch helped us understand not only how they work, but also why platforms such as Spotify or YouTube make certain recommendations.

Another strong motivation was the freedom offered by this topic: it provided a wide analytical and creative space in which we could experiment, interpret results and draw our own conclusions. Because each paradigm produced different patterns and behaviors, the project encouraged us to analyze, question, and visualize the outcomes in a clear and intuitive way. We particularly enjoyed taking the historical evolution of recommendation systems and transforming it into a hands-on, engaging, and visually appealing exploration. This made the learning process both meaningful and genuinely enjoyable.

Related Work

Our work draws on three major recommendation paradigms. **Early content-based systems** relied on item attributes and similarity measures. Pazzani and Billsus (2007) provided a comprehensive overview of this approach, while Billsus and Pazzani (1998) showed how classifiers such as decision trees or k-NN could be trained to recommend items based on content alone. **Collaborative filtering** became dominant after matrix factorization methods outperformed neighbor-based techniques. Koren et al. (2009) introduced latent factor models that captured complex user–item interactions and temporal dynamics. **Neural approaches** later introduced vector embeddings to model item similarity. Mikolov et al. (2013) proposed the skip-gram model, which inspired Spotify's Item2Vec—learning song relationships from listening patterns.

Reviewing this literature helped us understand how each paradigm defines similarity. Content-based models rely on explicit features. Collaborative filtering uncovers latent user-item patterns. Neural models learn dense embeddings from listening sequences. Some concepts, such as **latent factors in matrix factorization**, were initially difficult to interpret. These numerical vectors captured user–item relationships, but lacked direct meaning or labels. Re-implementing the model helped us understand how these abstract dimensions influence recommendation behavior.

Learning Outcomes

Each member reflected on personal learning outcomes:

- **Andra:** learned how preprocessing choices influence similarity metrics, how PCA reveals structure in acoustic spaces, and how Word2Vec embeddings capture contextual meaning in playlist sequences.
- **Alexandra:** learned how to build and evaluate collaborative filtering models (SVD + co-listening), how personalized recommendations differ from popularity-based ones, and how CF compares to Content-Based methods.

Beyond our individual tasks, we regularly reviewed and improved each other's contributions, enabling us to fully understand every part of the project.

2 Approach

The goal of this section is to describe in detail the methodology we used to reconstruct three historical paradigms of music recommendation systems. Our approach was fully implemented in Google Colab, using open-source tools and reproducible workflows. We structured the pipeline in a way that mirrors the chronological evolution of recommender systems: from early symbolic models, to statistical methods, to modern neural approaches.

Below we describe each step of our workflow, following the structure required in the template.

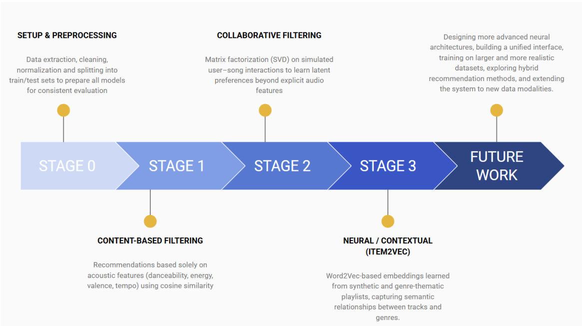

Figure 1: High-level pipeline illustrating the four historical stages of our recommender system reconstruction.

2.1 Code and Data Repository

All code, datasets, intermediate artifacts and generated visualizations are stored in a public GitHub repository:

`Access the full project repository here`

The repository contains:

- Google Colab notebooks for each stage (CBF, CF, Item2Vec, transitions, visualizations)
- the Spotify dataset used in all experiments
- exported PCA, t-SNE and Word2Vec embeddings

2.2 Software Tools Used

We implemented the entire project in **Google Colab**, which provided sufficient CPU/GPU resources for all three paradigms. The main libraries and tools used were:

- **Python 3.10**
- **NumPy, Pandas** for preprocessing and data manipulation
- **scikit-learn** for normalization, cosine similarity, PCA
- **Surprise** for Collaborative Filtering (SVD)
- **gensim** for Word2Vec / Item2Vec embeddings
- **matplotlib, seaborn** for all charts and visualizations

2.3 Training and Processing Time

Despite the diversity of methods, all experiments ran efficiently in Colab:

- Content-Based similarity (5000×5000 cosine matrix): a few seconds on CPU
- Collaborative Filtering (SVD on 200 users × 500 items): 2–3 seconds
- Word2Vec training (7 synthetic playlists × 20 epochs): under 10 seconds on CPU, faster on GPU

The project required no specialized hardware; all training was done using standard Colab resources.

2.4 Machine Learning Models and Architectures

Each historical stage corresponds to a distinct class of models:

- **Content-Based Filtering**: cosine similarity over normalized acoustic features (danceability, energy, valence, tempo)
- **Collaborative Filtering (SVD)**: low-rank matrix factorization using user–item interactions
- **Neural Item2Vec**: Word2Vec skip-gram architecture applied to synthetic playlists (vector size = 128, window = 5, epochs = 20–30)

These models allowed us to observe how the concept of "similarity" evolves from symbolic to statistical to neural representations.

2.5 Tricks and Practical Considerations

Although the models were relatively lightweight, we applied several practical optimizations:

- feature scaling with MinMaxScaler for stable cosine similarity
- user–song interaction simulation for CF reproducibility
- sampling strategies for playlist generation to avoid duplicates

2.6 Evaluation of the Methods

We evaluated each paradigm using the metrics and visualizations most representative for its era:

- **CBF**: cosine similarity, PCA 2D embeddings, top-N similarity ranking

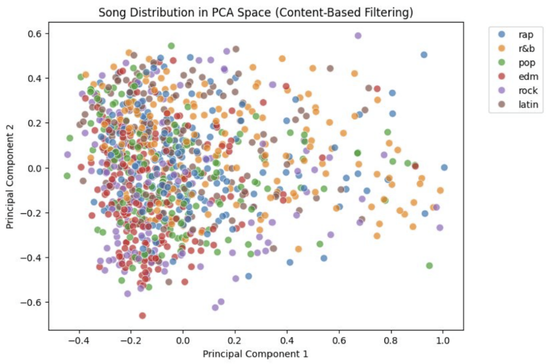

Figure 2: PCA projection of songs based on normalized acoustic features used in the content-based model.

- **CF**: Co-listening recommendations, top-N user recommendations

=== SMART-SVD Recommendations for user_id = 15 ===

Songs already listened to (with rating):

	track_name	track_artist	playlist_genre	rating
2	Only God Can Judge Me	2Pac	rap	2
3	Insta	Bizzey	latin	2
4	Being Alive	Hardwell	edm	4
7	Dance?	Night Tempo	r&b	1
11	Idealne Połączenie	Young Igi	rap	2
...	...	...	...	...
492	In My Mind	The Amazons	rock	5
493	Extraños En El Escaparate - Live	Miguel Rios	rock	4
495	Sparkles	Daxten	latin	2
496	Reflections	Jean Juan	edm	4
497	Cryin' Like A Bitch!!	Godsmack	rock	5

172 rows × 4 columns

SVD Recommendations (new songs):

	track_name	track_artist	playlist_genre	predicted_score
107	Wrong Way	Sublime	rock	4.295300
241	Sofia	Alvaro Soler	latin	4.109802
96	Lovesong	The Cure	rock	3.986341
370	The Dark	John Clark	rock	3.898507
391	Here I Go Again 87 - 2017 Remastered Version	Whitesnake	rock	3.874134
487	White Flag	Dido	rock	3.792521
90	(Don't Fear) The Reaper	Blue Öyster Cult	rock	3.768494
94	Bedside Radio - Remastered	Krokus	rock	3.753841
328	All In My Head	Tori Kelly	r&b	3.708414
344	The Climb	Miley Cyrus	pop	3.705294

Figure 3: Example of SVD-based recommendations generated for a user, showing predicted scores and new suggested tracks.

- **Word2Vec**: similarity in embedding space, t-SNE visualization, genre transition coherence

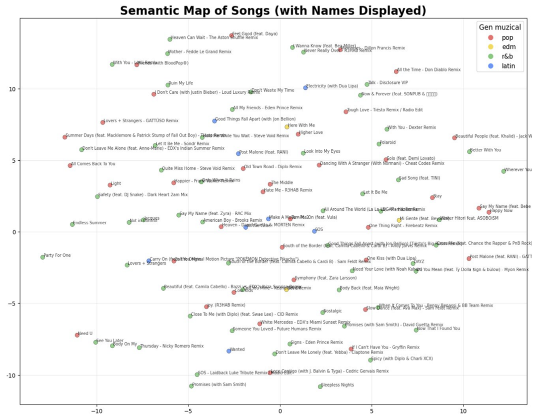

Figure 4: Semantic map of songs obtained with the Item2Vec model, showing genre-consistent clusters in the learned embedding space.

2.7 Tables, Figures and Visual Evidence

Throughout the project we include figures generated directly in our Colab notebooks:

- heatmaps, cluster maps, bar charts
- PCA and t-SNE projections of song embeddings
- comparative tables (CBF vs CF)
- semantic genre-transition diagrams

These visualizations support the validity of each model and highlight the differences between recommendation paradigms.

3 Limitations

Although our project successfully reconstructs the evolution of music recommendation systems, several methodological limitations should be acknowledged.

Dataset Constraints

The "Kaggle 30,000 Spotify Songs" dataset provides only acoustic descriptors and coarse-grained metadata. It does not include real user interaction logs, timestamps, skip-rates, long-term user preferences, or contextual listening information. This limitation particularly affects the Collaborative Filtering component, since we had to *simulate* user–song interactions and construct an artificial user–item matrix instead of relying on real behavioral data.

Playlist Availability for Neural Models

For the neural Item2Vec component, the dataset did not contain real playlists, which are essential for learning meaningful sequential embeddings. Originally, the only grouping available was the musical *genre*, which is much too broad to approximate real listening behavior. To address this, we generated synthetic contextual playlists using controlled ranges of *danceability*, *energy* , and *valence*. Although these playlists allowed us to train a Word2Vec model, they cannot fully replicate the richness and unpredictability of real user-created playlists. As a consequence, the learned embeddings may exhibit weaker semantic structure compared to embeddings trained on real Spotify playlist corpora.

Model Simplifications

Each of the three paradigms implements a simplified version of the industrial systems used by platforms such as Spotify or YouTube Music:

- **The Content-Based** model relies only on ten acoustic features, whereas real-world engines use hundreds of descriptors (spectral, temporal, timbral, emotional, etc.).
- **The Collaborative Filtering** model uses standard SVD instead of more advanced methods such as ALS (Alternating Least Squares), Bayesian MF, or neural collaborative filtering.
- **The Item2Vec** model is trained on a small synthetic corpus, unlike industrial versions trained on millions of playlists.

Scalability and Computational Constraints

All experiments were run in Google Colab on CPU resources. Larger datasets or more complex neural architectures were not feasible due to runtime limitations. As a result, some hyperparameters (embedding dimensionality, negative sampling rate, training epochs) had to be kept small, which may reduce the expressiveness of the learned representations.

Evaluation Limitations

Evaluating recommendation systems requires:

- user-level feedback metrics,
- A/B testing,
- qualitative playlist coherence evaluation.

Since we did not have real users or behavioral logs, the evaluation was limited to internal inspection, cosine similarity consistency and visualization via PCA and t-SNE. This makes the assessment valid for experimental reconstruction, but not for deployment-level quality.

Interpretability and Reproducibility

While we documented the full implementation, some intermediate behaviors (e.g., latent factor interpretability in SVD or cluster meaning in t-SNE) remain only partially explained due to the inherent opacity of the models. Additionally, the generation of synthetic playlists introduces randomness, making the exact reproducibility dependent on random seeds.

4 Conclusions and Future Work

Completion of this project allowed us to better understand not only how different recommendation paradigms work, but also how they reflect the technological evolution of the last two decades. Rebuilding classical methods, statistical models and neural embeddings from scratch helped us bridge the gap between theory and practical implementation.

Conclusions

Looking back, there are several aspects that we would approach differently with the experience we gained throughout the project:

- We would collect or generate a richer dataset, ideally one that includes real playlists or user interaction signals, which would make the neural Item2Vec embeddings more coherent and meaningful.
- We would allocate more time to hyperparameter tuning and deeper evaluation, especially for the Collaborative Filtering model.
- We would design a more systematic experiment protocol from the beginning (clear baselines, well-defined metrics, reproducibility rules), which would have made comparisons between models even more consistent.

Overall, we genuinely enjoyed this project because it combined creativity, machine learning, data analysis and visualization. The process of visually interpreting the results (PCA, t-SNE, semantic transitions) made the entire learning experience more intuitive and rewarding.

Future Work

There are several directions in which this project could be extended in meaningful ways:

- **Designing a fully custom neural architecture:** instead of using Word2Vec, we would like to build our own sequence-based model to learn deeper musical semantics and transitions.
- **Building a unified graphical interface:** integrating all three recommendation paradigms into a small web or desktop application, where the user can select a song and visually compare recommendations generated by each model. This would hide the code and make the exploration more interactive and accessible.
- **Training on larger, more realistic data:** real playlists (from Last.fm, Million Playlist Dataset, Deezer, etc.) would significantly improve the contextual embeddings and reduce the synthetic bias.
- **Exploring hybrid recommenders:** combining content-based and collaborative features, or mixing latent factors with embedding distances, similar to modern industrial systems.
- **Extending the analysis to new modalities:** incorporating lyrics embeddings or emotional analysis to understand how multimodal recommendation systems behave.

In future editions of this course, it would be interesting to explore projects in which students design interactive recommendation tools or in which multiple models can be compared visually. We believe that such extensions would make the learning experience even more engaging and enjoyable.

References

Daniel Billsus and Michael J. Pazzani. 1998. Learning collaborative information filters. *Proceedings of ICML*, pages 46–54.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of ICLR*.

Michael J. Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The Adaptive Web*, pages 325–341. Springer.