Recommender System:

Individual Assignment 2 

*Hans Alberto Franke,* [*h.a.franke@students.uu.nl*](mailto:h.a.franke@students.uu.nl)

Personalization for Public Media

Applied Data Science - Utrecht University - 2021

**Main Objective**

Build a recommender system and interface taking in account user’s values like control, diversity, transparency and acceptance

# Introduction

Recommender systems have been researched extensively over the past decades. Whereas several algorithms have been developed and deployed in various application domains, recent research efforts are increasingly oriented towards the user experience of recommender systems [1]. This research goes beyond accuracy of recommendation algorithms and focuses on various human factors that affect acceptance of recommendations, such as user satisfaction, trust, transparency and sense of control.

Many different stakeholders can use or affect a recommender system. These stakeholder have different values and desired outcomes. Which leads to the question of: How this can be connected and addressed in a single recommender system? How the interface can be used to answer for human values like **control**, **diversity**, **transparency** and **acceptance** and increase audience metrics like **usage** and **engagement**?

In this work, we present an interface that bridges the recommendation with visualization techniques to support human-recommender interaction.

# Methodology

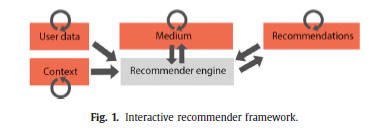
Recommender algorithms are categorized in three areas: *collaborative filtering* recognizes commonalities between users or between items on the basis of explicit (ratings, tags, etc.) or implicit (actions like reading, downloading.) [2]. It can be user-based (overlapping interaction) or item-based (similarities/distances. [3]. *Content-based filtering* matches descriptions of items to descriptions of users [4]. They base their predictions on information about individual users and items, and ignore contributions from other users (metadata) [5]. *Hybrid recommender systems* combine recommendation techniques, to gain better performance with fewer drawbacks [3].

New approaches makes the user more relevant to provide inputs and context to a recommender, for example, providing feedback that is used to better predicting in the future. This is given, allowing user exert control over the algorithm, choosing your preferences or values for example. A teenager may be interested in more trend topics similar to his friend, but an adult on the other side may be interested in old well rated movies of western genre.

Some well know values in literature are*,* ***Transparency***deals with the “black-box” nature of current recommender systems by explaining the inner logic of the system to end users. J***ustification***helps users understand why they get certain recommendations, but it may not relate to the inner logic of the recommendation techniques. ***Controllability***strengthens user involvement by incorporating input and feedback from the end user into the recommendation process. It can occurs in any step, such as providing ratings, adjusting preference data, and revising or exploring recommendations.

***Diversity****,* itrefers to providing recommendations with a relatively large coverage of the recommendation space [6]. For instance, it is important to recommend items that the user would prefer, but that are different from those which she has already purchased or experienced. The inability to make recommendation to new comers is called the ***cold start***problem. Acquiring ***contextual***information and incorporating it into recommendation processes in a flexible and fluid manner has gained increased interest over the past decades. The goal is to tailor recommendations to the current needs of the target user.

There is many ways to asses these values, and the **Algorithmic Affordance pattern library** is a first attempt to give an overview over the possibilities for designers to give end-users more control over the outcomes of algorithms [7]. These patterns can show many different possibilities to allow, measure and improve recommender system with very well designed interfaces (see table1).



**Figure1**. Workflow of feedback from interaction of a **user** with **recommender.** Adapted from [8]

|  |  |  |
| --- | --- | --- |
| Pattern | Main Idea | How is used in My interface |
| Peek Picker | Users can switch between their own recommendations and those of peers which are further away. | Recommendations based on collaborative filter (selecting friends / genres ) |
| Multiple Profiles | User can select different profiles | Profile Change (So user can compare out what works for them) |
| Incognito | User can go “anonymous” not recording any statistics | Button Incognito on main page |
| Introduce Chance | Giving users the option to activate a chance effect into their recommendations leads to more diverse recommendations. | Movies based on never watched genres, tags, users and random generator. They understand chance plays a big role so they may expect recommendation surprises. |
| Data Toggles | Switches may enable the user to chose which data is used by the algorithm to arrive at a recommendation. | Allow user selection genres, actors, directors from a list in the interface. Users may manipulate this information to build a dynamic model of what information is vital to their recommendation. |
| Social Context | Social recommendation systems can be more transparent by indicating the social group their recommendations are based upon. | Show to user the friends, genres of which recommendation was based. Users are aware the suggestions are context sensitive. |
| Ordered List | By presenting top reccomendations in an ordered list, users can make a choice among multiple items that are recommended for them. | Lists ranked by similarities and ratings, so the user can choose among presented alternatives |
| Liking Items | Users are given a lightweight control to express their opinion about an item, cumulatively resulting in feedback to the algorithm about user preferences. | Allow user rates movies (1-5), these ratings are using as input metrics to algorithm (explicit feedback) |
| Blacklist | Users can blacklist items, giving the algorithm an idea of their dislikes and preventing the algorithm from showing it again. | Users rates movies, so intuitively the algorithm will recommend less similar items. A user can filter a genre, so it will blacklist that genre from results. |
| Cold Star | Algorithms need basic information about users to be able to deliver their first recommendation. | User can define during register page settings like: favorite genres, news or rated movies. |

**Table1**. Summary of patterns used in this work with connection how they are assessed on the interface

# Recommender System and User’s Interface

## 3.1 How the interface allow personas/users values be assessed / Control:

Users can and should provide feedback through interface control. The interface was design focusing on assessing the main values: ***Controllability****,* ***Transparency***, ***justify*** *and* ***Diversity****.* This was design following the patterns suggested in [7].

This work try to asses *Controllability* allowing user to explicitly and implicitly interact with recommender. The choices of genres, friends, tags, by search button, using anonymous mode are mechanisms to allow **control** in a **transparency** way, with users understand what is happening and with **acceptance** of the suggestions, once he participated in the recommendation through **explicit** (i.e rate, like/dislike, de-select of genre or movie) and **implicit** (i.e watching the movie, adding to his list, recommend to a friend. ) feedback.

*Transparency* and *justify* were assessed but in an indirect way, with text showing to the user how the recommender was provided (i.e. based on movie you had watched or because of similar users watched), but not with high level detail (i.e what are the main similarities between users/movies). *Cold start* problem was answered in a login step with a user choosing 3 preferred genres, but as our system is static no approach to a new movies was assessed.

*Diversity,* was assessed allowing user receive random recommendations that he never watched, it can be a new genre or a new movie. In the login screen the user can chose if they value most his previous experiences compare to others and if he likes most new content or liked content (i.e similar that what he has watched)

On the other side, a recommender must increase audience metrics. Crafting content that connects with its preferred audience requires understanding audience objectives and how these objectives match enterprise goals. Starting by aligning with your audience personas, use trend data to trace the lifecycle of a customer and their content needs.

## 3.2 Metrics:

On the other side of the interface there is a company that need some metrics to measure the success of the recommender, not only based on users perspective but in business as well. For example, **users** **metrics**: diversity, engagement and satisfaction.

So to address **users values** the metrics would be, % of selection of filters (i.e genres, tags, friends) this will assess how much **control** a user has over the recommender**. Diversity,** can be measure with how many %random chances the user is trying in the lucky wheel. **Engagement**, can measure by how my suggestion to friends a user made, or how many days/week a user join the website. **Satisfaction** can be measure by how many positive or negative feedbacks a user made, or number of friends invited.

**Business metrics**: hours of views, % of correct prediction, coverage (how many genres a user has watched or % long tail movies), number of subscriptions. These metrics is not only for make more money, but they are important to measure if the user has his own metrics/values fulfilled, like a user that is increasing his average number of views probably has his values addressed.

## 3.3 Explanation of algorithms:

### 3.3.a. Simpler Recommender

One simple way to recommend a movie is based on ratings. One can user weighted ratings based on rates and number of votes (i.e. score). It can be built on “best” movies, genres, tags, and many different filters. Who never go to google and type: “best movies of all time?”, and the return is a list from **imbd** showing their famous TOP250 [ <https://www.imdb.com/chart/top/> ].

### 3.3.b. Content Based Recommender

The recommender we built in the previous section suffers some severe limitations. For one, it gives the same recommendation to everyone, regardless of the user's personal taste. If a person who loves romantic movies (and hates action) were to look at our Top 15 Chart, he wouldn't probably like most of the movies. If he were to go one step further and look at our charts by genre, he wouldn't still be getting the best recommendations.

To personalize our recommendations more, an engine that computes similarity between movies based on certain metrics (i.e tags, genres, users views) and suggests movies that are most similar to a particular movie that a user liked. Since we will be using movie metadata (or content) to build this engine, this also known as **Content Based Filtering. The user can provide input to this part choosing in the interface his preferences.**

### 3.3.c. Collaborative Filtering

Content based engine do not close all the gap, for it is only capable of suggesting movies which are close to a certain movie. That is, it is not capable of capturing **tastes** (i.e users preferences) and providing recommendations across genres. Also, the engine that we built is not really personal in that it doesn't capture the personal tastes and biases of a user. Anyone querying our engine for recommendations based on a movie will receive the same recommendations for that movie, regardless of who he is. Therefore, a technique called **Collaborative Filtering** is based on the idea that users similar to me can be used to predict how much one will like a particular product or service those users have used/experienced but one have not.

In this work was used the **Surprise** library that used extremely powerful algorithms like **Singular Value Decomposition (SVD)** to minimize RMSE (Root Mean Square Error) and give great recommendations. The algorithm was improved with *gridSearch* (selection of the best parameters and errors metrics).

### 3.3.d. Hybrid Recommendation

Merging the 3 previous approaches in one single recommendation is the best scenario to take in account many different criteria’s to fully satisfy a user. For example, we can start by assessing similarities based on others users (collaborative filter), then search the movies with the 10 most similar users (content based), filtering for the 3 target\_user preferred genre (content based as well), ranking the movies by others users score (basic filter, weighted ratings). The final list of movies is the input for the SVD to predict based on that list which movies will have the most chance to assess the target user preferences.

Using this techniques we are assessing the users values (user can change parameters of the algorithm explained on the interface and Methodology chapter), and assessing the metrics of the recommender like: user engagement, increasing views, and one of the most important: **user** **satisfaction** because we are increasing the chance to predict correctly. For example, one can filter which friend he/she want to compare, select genres that want to include, the level of similarities (own x others), and provide feedback in each iteration (i.e accept or refuse the recommendations).

# Conclusion

In this work, was designed an interface using library patterns to asses **values**, allowing **control** over the recommender. The interface is crucial, for it is the connection point from end-user to the recommender, through her that you are explaining to your users how you are building the recommendations and more important take his own **preferences** into account for future recommendations.

It was built 4 different recommendation engines based on different ideas and algorithms. They are as follows:

1. **Simple Recommender:** This system used overall Vote Count and Vote Averages to build Top Movies Charts, in general and for a specific genre. The weighted Rating System was used to calculate ratings on which the sorting was finally performed.
2. **Content Based Recommender:** Built two content based engines; one that took movie title and users as input and the other which took metadata such as tags and genres to come up with predictions. We also devised a simple filter to give greater preference to movies with more votes and higher ratings.
3. **Collaborative Filtering:** We used the powerful Surprise Library to build a collaborative filter based on single value decomposition. The RMSE obtained was less than 0.9 and the engine gave estimated ratings for a given user and movie.
4. **Hybrid Engine:** We brought together ideas from content and collaborative filtering to build an engine that gave movie suggestions to a particular user based on the estimated ratings that it had internally calculated for that user.

The work assessed metrics of **users** and **business**, trying to taking in account inevitable tensions like: more views or more diversity? Metrics sometimes contrary to each other, but the tension was minimized through transparency improving user acceptance and in a natural way increasing views because simply the algorithm make better predictions, once it used users inputs and control.

# Bibliography

|  |  |
| --- | --- |
| [1] | T. Asikis, J. Klinglmayr, D. Helbing and E. Pournaras, "How value-sensitive desing can empower sustainable consumption," no. https://doi.org/10.1098/rsos.201418, 2021. |
| [2] | R. Burke, "Evaluating the dynamic properties of recommendation algorithms.," *In Proceedings of fourth ACM conference on recommender systems,* no. doi: 10.1145/1864708.1864753, 2010. |
| [3] | R. Burke, "Hybrid recommender systems: Survey and experiments. User Modeling and User Adapted Interaction," 2002. |
| [4] | M. J. Pazzani and D. Billsus, "Content-based recommendation systems," no. http://www.springerlink.com/index/qq35wt68l6774261, 2007. |
| [5] | S. Ternier, K. Verbert, G. Parra, B. Vandeputte, J. Klerkx, E. Duval and X. Ochoa, "The ariadne infrastructure for managing and storing metadata.," *IEEE Internet Computing, 13 ,* no. doi: 10.1109/MIC.2009.90, p. 18–25, 2009. |
| [6] | R. Hu and P. Pu, "Helping users perceive recommendation diversity.," *In Proceedings of the workshop on novelty and diversity in recommender systems, divers.,* 2011. |
| [7] | "Pattern Library," no. https://aapatternlibrary.wordpress.com/. |
| [8] | D. P. ,. V. Chen He, "Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities," 2016. |
| [9] | Z. Fayyaz, M. Ebrahimian, D. Nawara, A. Ibrahim and R. Kashef, "Recommendation Systems: Algorithms, Challenges, Metrics, and Business Opportunities," 2020. |
| [10] | H. Cramer, J. G. Gathright, A. Springer and S. Reddy, "Assessing and addressing algorithmic bias in practice," *https://doi.org/10.1145/3278156,* 2018. |
| [11] | J. Davis and L. Nathan, Value Sensitive Design: Applications, Adaptations, and Critiques, 2015. |
| [12] | R. Dobbe , S. Dean, T. Gilbert and N. Kohli, "A Broader View on Bias in Automated Decision-Making: Reflecting on Epistemology and Dynamics," no. https://doi.org/10.14763/2020.4.1534, 2018. |
| [13] | B. Friedman, Human Values and the Design of Computer Technology, Cambridge University Press, 1997. |
| [14] | S. Kheirandish, M. Funk, S. Wensveen, M. Verkerk and M. Rauterberg, "HuValue: a tool to support design students in considering human values in their design," *International Journal of Technology and Design Education,* no. https://doi.org/10.1007/s10798-019-09527-3, p. 30:101, 2020. |
| [15] | J. Simon, P.-H. Wong and G. Rieder, "Algorithmic bias and the Value Sensitive Design approach," 2020. |
| [16] | StakeholderMap, "https://www.stakeholdermap.com/retail-stakeholders.html". |

# Appendix:

## Code

The is provided in the jupyter notebook named: Media\_RecommenderBackEnd\_HansFranke.ipynb

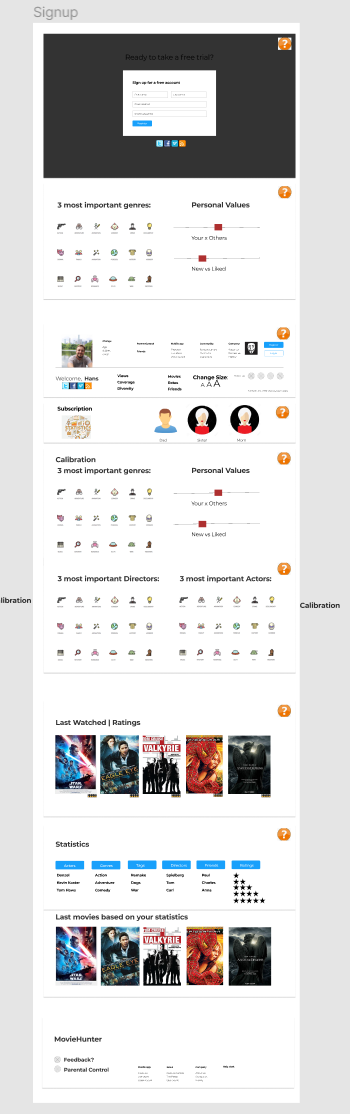
Or you can open the html generated:



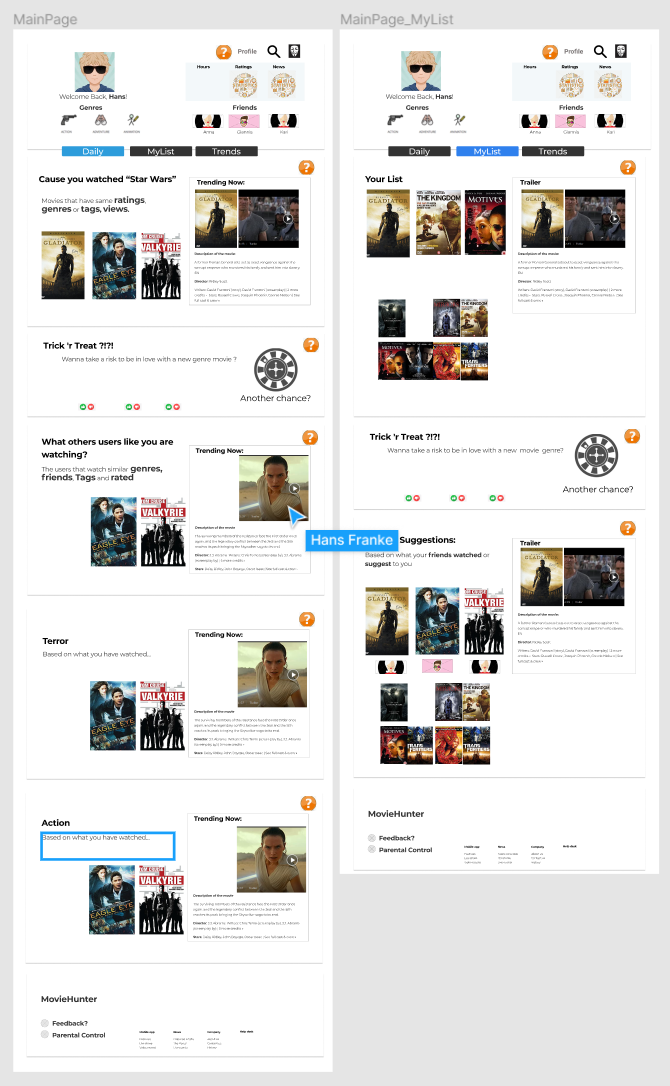
## Interface

The interface is public available at: <https://www.figma.com/file/ItyHCF6CAdeDkbazTYkyAI/Media?node-id=23%3A83>

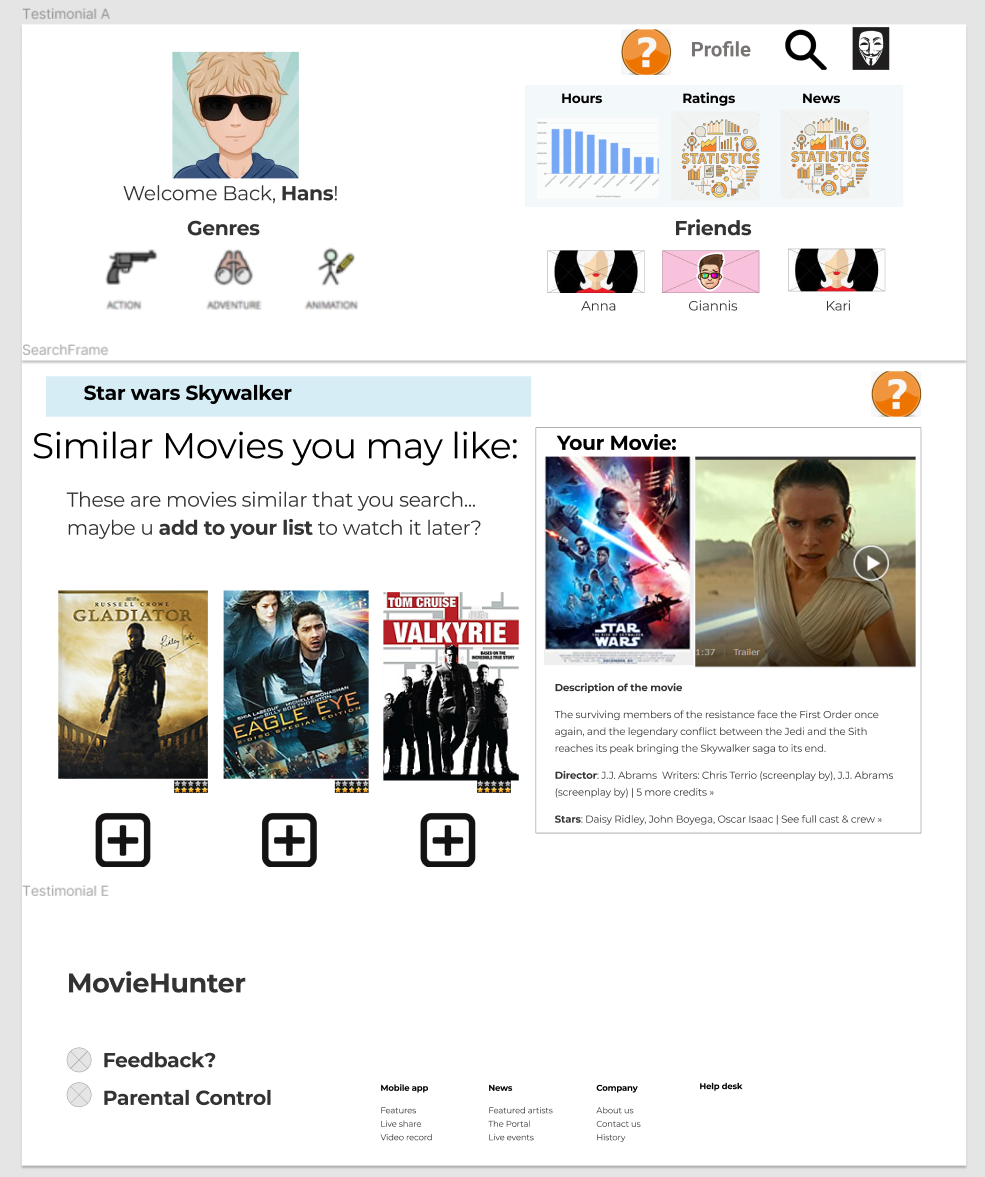
Sign and login pages:



Main Page and MainPage/MyList:



SearchPage:



Policy Page:

