# Recommender System:
## Individual Assignment 2

*Hans Alberto Franke, 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

## 1. Introduction

While 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 as well as affect a recommender system. These stakeholder have different values and desired outcomes, which leads to the following questions: How can these differences be connected and addressed in a single recommender system? How can the interface be used to implement 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.

## 2. Methodology

Some well know values in literature [2] are, **Transparency** that deals with the "black-box" nature of current recommender systems by explaining the inner logic of the system to end users. **Justification** 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. This can occur in any step, such as providing ratings, adjusting preference data, and revising or exploring recommendations.

**Diversity**, refers to providing recommendations with a relatively large coverage of the recommendation space [3]. For instance, it is important to recommend items that the user would prefer, but that are different from those which she/he 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.

**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 [4]. These patterns can show many different possibilities to allow, measure and improve recommender system with well-designed interfaces (see table1).
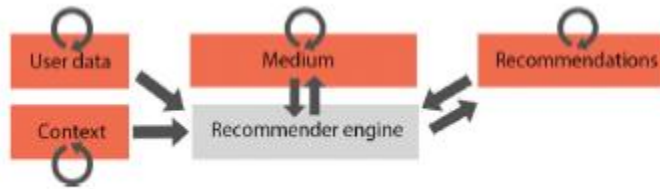


Fig. 1. Interactive recommender framework.

**Figure1**. Workflow of feedback from interaction of a **user** with **recommender.** Adapted from [5]

| Pattern | Main Idea | How it 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 choose 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 recommendations 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 on how they are implemented on the interface

# 3. Recommender System and User's Interface

## 3.1 How the interface allows users values to be fulfilled:

Users can and should provide feedback through the interface. The interface was designed with focus the main values: **Controllability**, **Transparency**, **justification** and **Diversity**. It was designed following the patterns suggested in [4].

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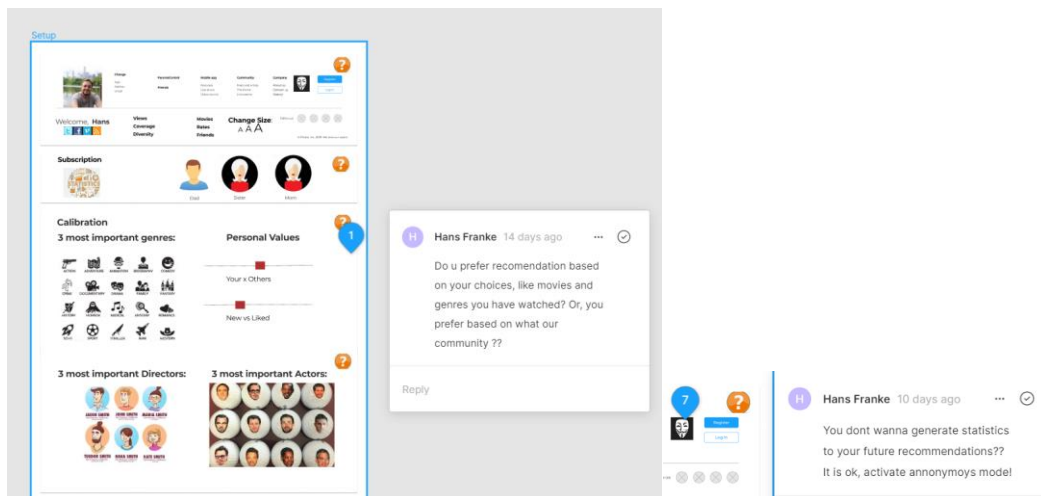


**Figure2**. Example on interface of user exerting control: he/she can select genres, directors and actors. Another form to exert control is rating a movie (explicitly), or for example simply watching a movie (implicitly). On right: description to enter in **anonymous** mode.
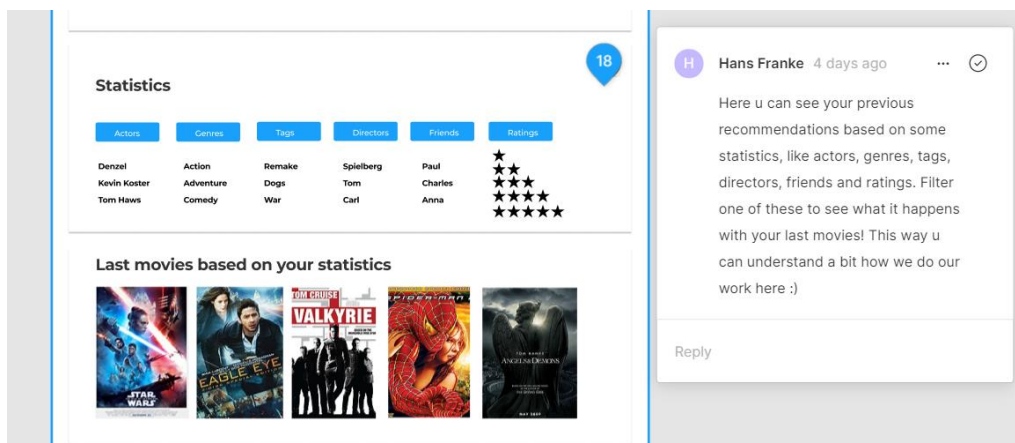


**Figure3**. User's statistics to improve acceptance through transparent explanations, showing the last movies recommended based on these statistics.

**Transparency** and **justification** *(*see figure 3, 4 and 5*)* were assessed in an indirect way by adding text showing 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 addressed in the login process by letting the user choose 3 preferred genres (see figure2), but as our system is static no approach to a new movies was assessed.
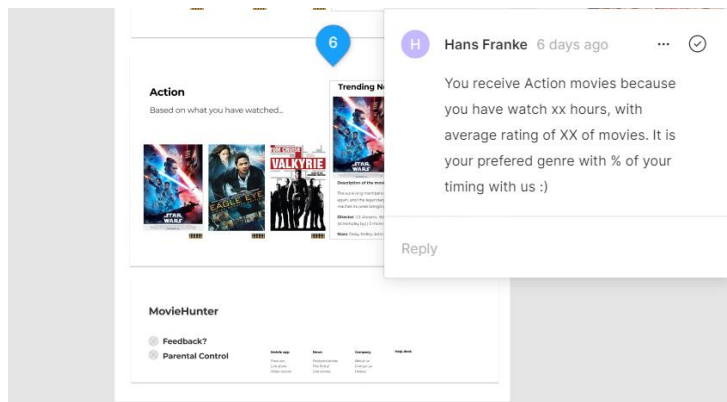


**Figure4**. Example of popup message to increase transparency and acceptance from the user.
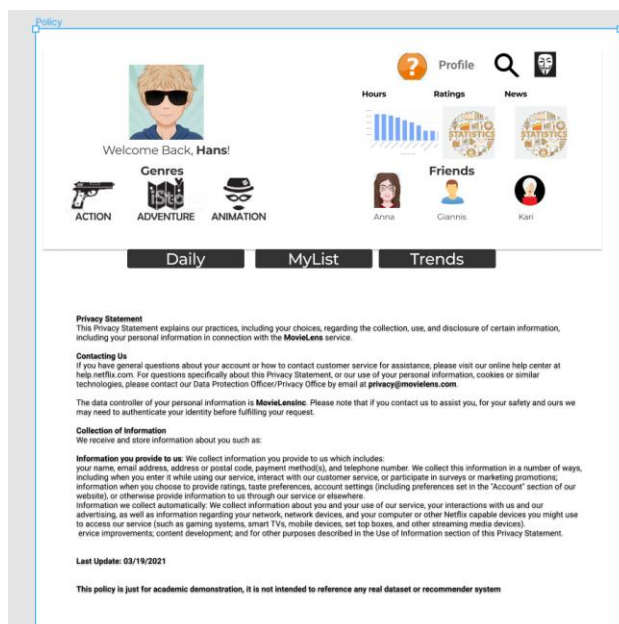


**Figure5**. Another example of transparency, on the policy page describe how the user data is collected.

**Diversity**, was assessed allowing user receive random recommendations that he never watched before(see figure 6). This could be a new genre or a new movie unrelated to any similarity with movies or users. In the login screen the user can chose if the previous experiences are valued more than other users experience and if new content is preferred over liked content (i.e similar that what he has watched) (see figure2).

```
[29]: #show genres never watched!
      genres

[29]: ['(no genres listed)', 'Sci-Fi', 'Film-Noir', 'Western', 'Horror', 'Fantasy']

[30]: #show list of movies of that genres!
      movies_list.head()
```

| | index | movieId | title | genres | userId | rating | timestamp | tag |
|---|---|---|---|---|---|---|---|---|
| 0 | 102973 | 913 | Maltese Falcon, The (1941) | Film-Noir | 16 | 4.0 | 1377477016 | statue |
| 1 | 108049 | 930 | Notorious (1946) | Film-Noir | 4 | 5.0 | 964538571 | assassination |
| 2 | 108310 | 968 | Night of the Living Dead (1968) | Horror | 64 | 3.5 | 1161536420 | zombies |
| 3 | 119586 | 1212 | Third Man, The (1949) | Film-Noir | 18 | 4.5 | 1455059742 | ferris wheel |
| 4 | 119784 | 1214 | Alien (1979) | Horror | 1 | 4.0 | 964981855 | aliens |

```
[31]: #predict scores for that movies and sort_by that
      predictions2 = []
      for i in range(0,len(movies_list)):
          movie_id = movies_list.loc[i]["movieId"]
          pred = svd.predict(user, movie_id).est #chose only the value of prediction!
          predictions2.append(pred)

      #create a new column with the predicitons
      movies_list['pred'] = predictions2
      movie_list = movies_list[['movieId', "title", "genres", "tag", "pred"]]

      #order and return only 5 values!
      movie_list.sort_values("pred", ascending=False)[0:5]
```

| | movieId | title | genres | tag | pred |
|---|---|---|---|---|---|
| 5 | 1258 | Shining, The (1980) | Horror | atmospheric | 4.008160 |
| 0 | 913 | Maltese Falcon, The (1941) | Film-Noir | statue | 4.006685 |
| 1 | 930 | Notorious (1946) | Film-Noir | assassination | 3.994357 |
| 3 | 1212 | Third Man, The (1949) | Film-Noir | ferris wheel | 3.953664 |
| 47 | 109487 | Interstellar (2014) | Sci-Fi | black hole | 3.928913 |
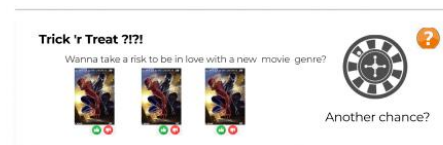
**Figure6**. Example of diversity, based on genres never watched by a user, and a print of the interface part of the "Lucky wheel". The prediction score was based on SVD (Single Value Decomposition).

The goal of the recommender is to increase the audience metrics. Crafting content that connects with its preferred audience requires an understanding of the audience objectives as well as 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 needs some metrics to measure the success of the recommender not only based on users perspective but also from a business standpoint. For example, **users metrics**: diversity, engagement and satisfaction.

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

**Business metrics** can be views per hour, % of correct prediction (accuracy), coverage (how many genres a user has watched or % long tail movies), number of subscriptions. These metrics are not only focusing on adding value to the business, but they are important to measure the user satisfaction. For instance a user that is increasing his average number of views probably has his values addressed.
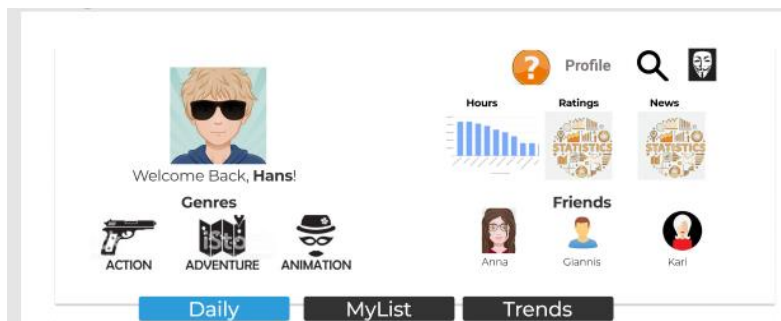
**Figure7**. On the main interface there is a quick overview of views (measured in hours) to show in a transparent way to the user his engagement to the website.

## 3.3 Explanation of algorithms:

### 3.3.a. Simpler Recommender

One simple way to recommend a movie is based on ratings. One can use 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 typed: "best movies of all time?", and the return is a list from **imbd** showing their famous TOP250 [ https://www.imdb.com/chart/top/ ].

Scored based on votes and ratings

```python
#Define a base function to allow weigthed vote, so it changes ratings to score, based on number of votes and ratings!
#You can avg votes by movie or by genre, it would be used later on!
def weight_votes(df, group_col='title', agg_col ="rating", number=10):
    avg_rating_df = df.copy()
    # calculate average with count
    avg_rating_df = df.groupby([group_col])[agg_col].agg(['mean', 'count'])
    avg_rating_df.rename(columns={'mean': 'vote_average', 'count': 'vote_count'}, inplace=True)
    avg_rating_df.sort_values('vote_average', ascending=False).head(10)

    # C is the mean vote or rating across the whole dataframe
    C = avg_rating_df['vote_average'].mean()

    # m is the minimum votes or ratings to be listed | using a low value because the dataset is small
    m = avg_rating_df['vote_count'].quantile(0.50)
    print('Numbers of votes to be in final list:', m)

    # weight function (ranked based on number of votes AND score)
    def weighted_rating(x, m=m, C=C):
        v = x['vote_count']
        R = x['vote_average']
        return (v / (v + m) * R) + (m / (m + v) * C)

    # check the new scores
    avg_rating_df = avg_rating_df.loc[avg_rating_df['vote_count'] >= m]
    avg_rating_df['score'] = avg_rating_df.apply(weighted_rating, axis=1)
    avg_rating_df = avg_rating_df.sort_values('score', ascending=False).head(number)
    return avg_rating_df

avg_rating_df = weight_votes(full_df_genres, "genres")
avg_rating_df
```

Numbers of votes to be in final list: 14241.5

| genres | vote_average | vote_count | score |
|---|---|---|---|
| Crime | 4.123582 | 103102 | 4.101546 |
| Thriller | 4.074407 | 116891 | 4.060029 |
| Drama | 4.053360 | 148389 | 4.043609 |
| Comedy | 3.992477 | 96901 | 3.986011 |
| Action | 3.952206 | 69549 | 3.950475 |
| Mystery | 3.923477 | 24352 | 3.930318 |
| Sci-Fi | 3.901226 | 51977 | 3.909998 |
| Adventure | 3.881026 | 47939 | 3.894995 |
| Romance | 3.801708 | 25344 | 3.852187 |
| Fantasy | 3.737176 | 16220 | 3.832944 |

```python
avg_rating_df = weight_votes(full_df, "title")
avg_rating_df
```

Numbers of votes to be in final list: 19.0

| title | vote_average | vote_count | score |
|---|---|---|---|
| Shawshank Redemption, The (1994) | 4.429022 | 1268 | 4.417556 |
| Guess Who's Coming to Dinner (1967) | 4.545455 | 44 | 4.276109 |
| Fight Club (1999) | 4.272936 | 11772 | 4.271936 |
| Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1964) | 4.268041 | 970 | 4.256213 |
| Departed, The (2006) | 4.252336 | 1070 | 4.241869 |
| Rear Window (1954) | 4.261905 | 504 | 4.239761 |
| Godfather, The (1972) | 4.289062 | 192 | 4.231729 |
| Godfather: Part II, The (1974) | 4.259690 | 387 | 4.231268 |
| Star Wars: Episode IV - A New Hope (1977) | 4.231076 | 6526 | 4.229396 |
| Usual Suspects, The (1995) | 4.237745 | 1224 | 4.228797 |

**Figure7**. Example of simple recommender based on weighted voting/rating, of genres and movies.

### 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 10 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.
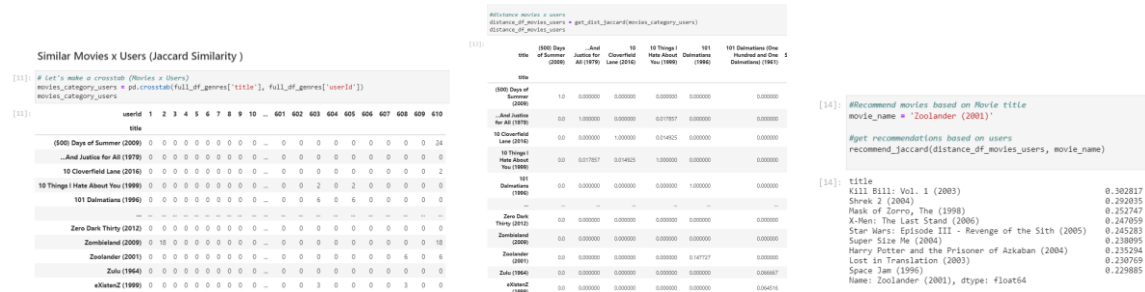


**Figure8**. Example of similarities using jaccard distances, on the left is similatiries from movies to users (which user watch that movies), center is distance from movie to movie based on users rates, and right part is recommend a movies based on "Zoolander", order by the similarities.

### 3.3.c. Collaborative Filtering

A content based engine does not close the gap completely as 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. Another, advantage of collaborative filtering is that it generates models that help users discover new interests.

In this work the **Surprise** library was used 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).

**Figure9**. Example of SVD: left is the training process and evaluating the error metrics (87% accuracy within RMSE), right is the prediction of user 474 rates movieId 2 with 3.57.

### 3.3.d. Hybrid Recommendation

Merging the 3 previous approaches in one single RS is the best scenario to take different criteria into account to fully satisfy a user. For example, we can start by assessing similarities based on others users (collaborative filter), then search the movies within 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 (see figure10**)**.

Within these techniques we are assessing the users values (user can change parameters of the algorithm explained on the interface and Methodology chapter), and 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).
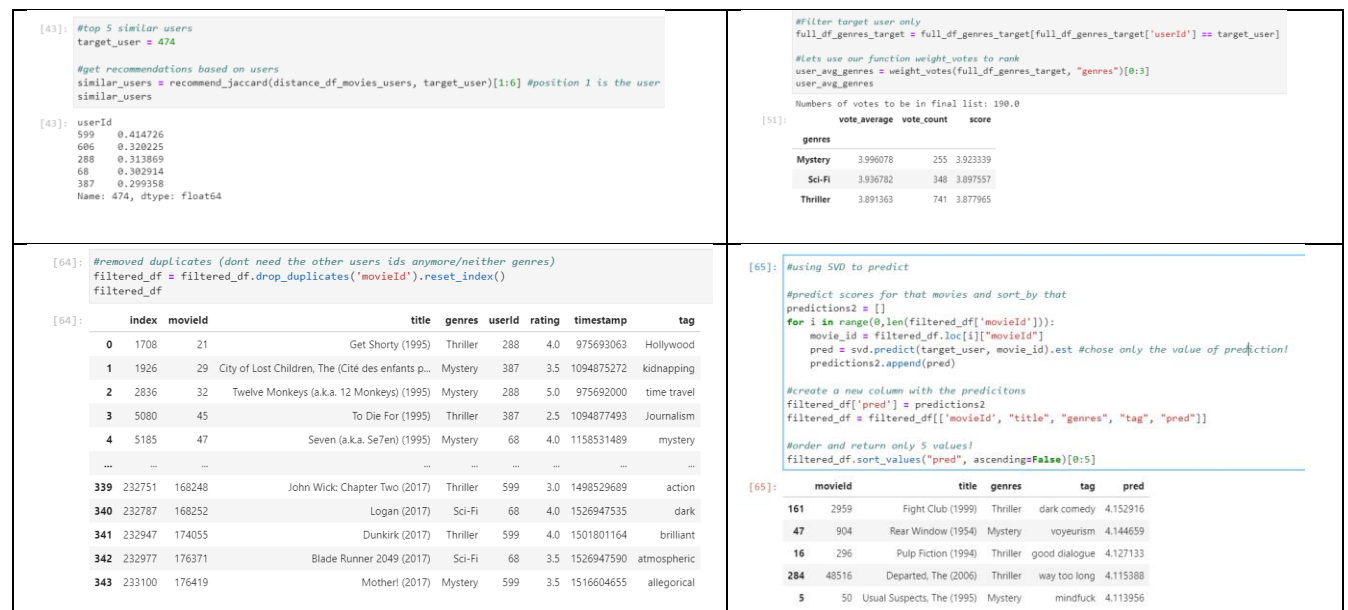


**Figure10**. Example of the **hybrid recommender**: First, select similar users based on distances to movies (jaccard), then filter the users top 3 genres (content based, based on voting score), followed by  filtering the df within similar users and preferred genres and finally use SVD to predict all movies and sort the list based on prediction scores.

## 4. Conclusion

In this work, was designed an interface using library patterns to asses **values**, allowing user **control** over the recommender system. The interface is crucial, for it is the connection point from the end-user to the recommender. Through the interface one can explain to the user how the recommendations are build (transparency and acceptance) and take the users own **preferences** into account for future recommendations (predicting power).

This strategy makes the user more relevant providing inputs and context to a recommender, for example, providing feedback that will be used to better predicting in the future. This is given, allowing user exert control over the algorithm, choosing his preferences (genres, actors, tags). 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. Using the interface the user can select parameters and visually notice how it changes his recommendations.

The work assessed metrics of **users** and **business**, trying to taking in account inevitable tensions like: more views versus more diversity? Metrics are sometimes opposed to each other, but the tension can be minimized through transparency and by improving the user acceptance which in a natural way increases the views as the algorithm makes better predictions, once it includes the users inputs and control.

## 5. 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. Hu and P. Pu, "Helping users perceive recommendation diversity.," *In Proceedings of the workshop on novelty and diversity in recommender systems, divers.,* 2011.

[4]   "Pattern Library," no. https://aapatternlibrary.wordpress.com/.

[5]   D. P. ,. V. Chen He, "Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities," 2016.

[6]   Z. Fayyaz, M. Ebrahimian, D. Nawara, A. Ibrahim and R. Kashef, "Recommendation Systems: Algorithms, Challenges, Metrics, and Business Opportunities," 2020.

[7]   H. Cramer, J. G. Gathright, A. Springer and S. Reddy, "Assessing and addressing algorithmic bias in practice," *https://doi.org/10.1145/3278156,* 2018.

[8]   J. Davis and L. Nathan, Value Sensitive Design: Applications, Adaptations, and Critiques, 2015.

[9]   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.

[10] B. Friedman, Human Values and the Design of Computer Technology, Cambridge University Press, 1997.

[11] 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.

[12] J. Simon, P.-H. Wong and G. Rieder, "Algorithmic bias and the Value Sensitive Design approach," 2020.

[13] StakeholderMap, "https://www.stakeholdermap.com/retail-stakeholders.html".

[14] R. Burke, "Hybrid recommender systems: Survey and experiments. User Modeling and User Adapted Interaction," 2002.

[15] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," no. http://www.springerlink.com/index/qq35wt68l6774261, 2007.

[16] 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. Appendix:

### a. Code

The is provided in a jupyter notebook named: Media_RecommenderBackEnd_HansFranke.ipynb

Open in Github:

https://github.com/hansfranke1985/Public-Media/blob/main/Assigment_2/Media_RecommenderBackEnd_HansFranke.ipynb

Or you can open the html generated:

Media_Recommende
rBackEnd_HansFranke.

## b. Interface

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

**Video:** https://www.youtube.com/watch?v=5_fiefXknjA&ab_channel=HansFranke

**Sign and Setup pages:**

**Main Page and MainPage/MyList:**

**Search Page:**

Profile

**Hours**

**Ratings**

**News**

Welcome Back, **Hans**!

**Genres**

ACTION    ADVENTURE    ANIMATION

**Friends**

Anna    Giannis    Kari

**Star wars Skywalker**

# Similar Movies you may like:

These are movies similar that you search...
maybe u **add to your list** to watch it later?

**Your Movie:**

1:37    Trailer

**Description of the movie**

The surviving members of the resistance face the First Order once
again, and the legendary conflict between the Jedi and the Sith
reaches its peak bringing the Skywalker saga to its end.

**Director**: J.J. Abrams  Writers: Chris Terrio (screenplay by), J.J. Abrams
(screenplay by) | 5 more credits »

**Stars**: Daisy Ridley, John Boyega, Oscar Isaac | See full cast & crew »

# MovieHunter

⊗ **Feedback?**

⊗ **Parental Control**

| Mobile app | News | Company | Help desk |
| --- | --- | --- | --- |
| Features | Featured artists | About us | |
| Live share | The Portal | Contact us | |
| Video record | Live events | History | |

**Policy Page:**
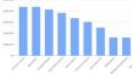
Welcome Back, **Hans**!

Profile

Hours
Ratings
News

**Genres**

ACTION
ADVENTURE
ANIMATION

**Friends**

Anna
Giannis
Kari

Daily
MyList
Trends

**Privacy Statement**
This Privacy Statement explains our practices, including your choices, regarding the collection, use, and disclosure of certain information, including your personal information in connection with the **MovieLens** service.

**Contacting Us**
If you have general questions about your account or how to contact customer service for assistance, please visit our online help center at help.netflix.com. For questions specifically about this Privacy Statement, or our use of your personal information, cookies or similar technologies, please contact our Data Protection Officer/Privacy Office by email at **privacy@movielens.com**.

The data controller of your personal information is **MovieLensInc**. Please note that if you contact us to assist you, for your safety and ours we may need to authenticate your identity before fulfilling your request.

**Collection of Information**
We receive and store information about you such as:

**Information you provide to us**: We collect information you provide to us which includes:
your name, email address, address or postal code, payment method(s), and telephone number. We collect this information in a number of ways, including when you enter it while using our service, interact with our customer service, or participate in surveys or marketing promotions; information when you choose to provide ratings, taste preferences, account settings (including preferences set in the "Account" section of our website), or otherwise provide information to us through our service or elsewhere.
Information we collect automatically: We collect information about you and your use of our service, your interactions with us and our advertising, as well as information regarding your network, network devices, and your computer or other Netflix capable devices you might use to access our service (such as gaming systems, smart TVs, mobile devices, set top boxes, and other streaming media devices).
 ervice improvements; content development; and for other purposes described in the Use of Information section of this Privacy Statement.

**Last Update: 03/19/2021**

**This policy is just for academic demonstration, it is not intended to reference any real dataset or recommender system**

# MovieHunter

⊗ **Feedback?**
⊗ **Parental Control**

**Mobile app**
Features
Live share
Video record

**News**
Featured artists
The Portal
Live events

**Company**
About us
Contact us
History

**Help desk**