Recommender Systems

Theo Crookes, Robert Lambert, James Neill, Wanchen Yue

Based on the masterclass presented by Prof. Edward Malthouse, Northwestern University

1. Introduction

Whether you are watching TV, online shopping, or swiping on a dating app, we often receive suggestions about what we would like. But how do these websites/apps know what to propose? They utilise a class of algorithms known as **recommender systems**.

Figure 1. Netflix utilise recommender systems to promote films and TV shows to users. Image copyright Rafael Henrique ©.

What is a recommender system?

Recommender systems are "software, tools, and techniques providing suggestions for **items** to be of use to a **user**" [1]. These items are not restricted to physical objects, instead, they include:

- News and information
- Products, vendors, bundles
- Matchmaking
- Media: movies, music, etc.

How do they work?

These algorithms make suggestions according to some decision-making policy. Depending on the availability of data and the application are, different policies might be used.

2. Naïve Policies

Often used as baseline cases in the comparison of methods, naïve methods are often intuitive but sub-optimal. Such policies include:

1. **Random** - recommended items are selected at random. Completely stochastic.
2. **Popular** - items recommended are the most popular. Either most-viewed, interacted with, bought or liked.
3. **Recent** - newest items are suggested.

More effective policies are those that are adaptive. They learn from their environment and achieve a balance between **exploration** and **exploitation**.

3. User-Based Collaborative Filtering

User-based collaborative filtering (**UBCF**) matches an active user to other individuals who share similar interest in objects. The items recommended to the active user are those that the similar individuals rank highly.

For each user u and item i, we aim to determine r_{ui} – the rating of item i by user u. Our estimate of r_{ui} is denoted $\hat{r}_{ui}$. We also let $\mathcal{I}_{uv}$ be the set of items rated by both users u and v.

User-based collaborative filtering requires us to quantify the similarity between two users; we let $\mathrm{sim}(u,v)$ be the similarity between users u and v. This is determined through **Pearson correlation** (1) or **cosine similarity** (2). Other methods, such as Jaccard similarity, are also available.

$$\mathrm{sim}(u,v) = \frac{\sum_{i\in\mathcal{I}_{uv}}(r_{ui}-\bar{r}_{u\cdot})(r_{vi}-\bar{r}_{v\cdot})}{\sqrt{\sum_{i\in\mathcal{I}_{uv}}(r_{ui}-\bar{r}_{u\cdot})^2\sum_{i\in\mathcal{I}_{uv}}(r_{vi}-\bar{r}_{v\cdot})^2}} \tag{1}$$

$$\mathrm{sim}(u,v) = \frac{\sum_{i\in\mathcal{I}_{uv}} r_{ui}r_{vi}}{\sqrt{\sum_{i\in\mathcal{I}_{uv}} r_{ui}^2}\sqrt{\sum_{i\in\mathcal{I}_{uv}} r_{vi}^2}} \tag{2}$$

For each item i, we determine a set of users $\mathcal{N}_{ui}$ who have rated item i, and are most similar to the active user. Averaging the rating given by the users in this set, we obtain a predicted rating for our active user u. Alternatively, we can use a weighted average of ratings utilising our choice of sim(u,v).

$$\hat{r}_{ui} = \frac{1}{|\mathcal{N}_{ui}|}\sum_{v\in\mathcal{N}_{ui}} r_{vi} \quad \text{or} \quad \hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v\in\mathcal{N}_{ui}}\mathrm{sim}(u,v)\,(r_{vi}-\bar{r}_v)}{\sum_{v\in\mathcal{N}_{ui}}|\,\mathrm{sim}(u,v)|}$$

Items are then recommended in order of their predicted rating.

4. Item-Based Collaborative Filtering

Item-based collaborative filtering (**IBCF**) is an alternative method of estimating ratings; it involves quantifying the similarity between items, and recommending items to the active user that are similar to items they have rated highly before. Again items are recommended in order of their estimated rating.

There are several methods to calculate the similarity between two items. These methods include:

- Pearson correlation
- Cosine similarity
- Jaccard similarity
- k-nearest neighbours

We can then estimate a rating for each user u and item i using a weighted average, where $\mathrm{sim}(i,j)$ is the similarity between items i and j.

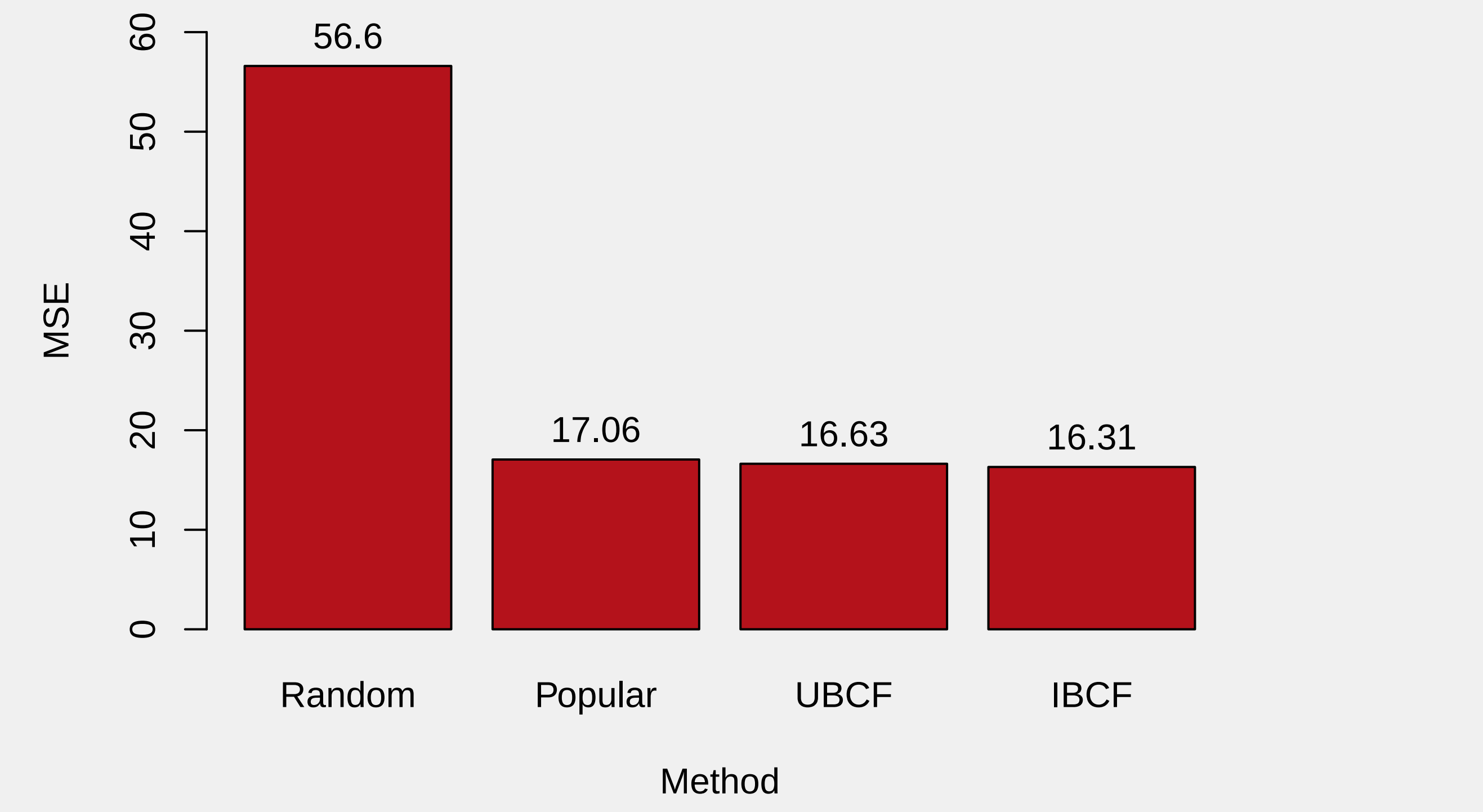
$$\hat{r}_{ui} = \frac{\sum_{j\in\mathcal{N}_{ui}}\mathrm{sim}(i,j)r_{uj}}{\sum_{j\in\mathcal{N}_{ui}}|\,\mathrm{sim}(i,j)|} \quad \text{or} \quad \hat{r}_{ui} = \bar{r}_i + \frac{\sum_{j\in\mathcal{N}_{ui}}\mathrm{sim}(i,j)\,(r_{uj}-\bar{r}_j)}{\sum_{j\in\mathcal{N}_{ui}}|\,\mathrm{sim}(i,j)|}$$

5. Comparison

Mean Squared Error (**MSE**) is a metric for quantifying the error incurred when comparing a prediction to known, true values. We calculate MSE for each method with the following formula (where $\mathcal{K}$ is the set of all user-item pairs with observed ratings).

$$\mathrm{MSE} = \frac{1}{|\mathcal{K}|}\sum_{(u,i)\in\mathcal{K}}(r_{ui}-\hat{r}_{ui})^2$$

Below, we plot the MSE for the different algorithms, applied to the joke recommendation dataset from the masterclass.



As expected, random policies perform poorly due to their inability to use information. UBCF, IBCF and popular give **significantly lower MSE**, however, the adaptive methods slightly outperform the popular approach.

6. Conclusion

Recommender systems are used across a number of domains to present users with suggestions. In addition to determining which items are **presented**, they also determine the **order** in which these items are shown.

Random policies perform poorly, whilst giving popular recommendations is effective, hence its prevalence in media and e-commerce. Adaptive methods balance **exploration** and **exploitation** to present optimal recommendations.

Limitations

- **Cold start -** Initially, recommender systems have no information about the users. The predictions they make initially often are not appropriate, this is known as cold start.
- **Availability of data -** In settings where data is sparse (either about users or items) recommender systems struggle to make effective recommendations.

References

[1] Ricci, F., Rokach, L. Shapira, B. (2015), 'Recommender systems: introduction and challenges', *Recommender systems handbook*, pp. 1–34.