QBUS6850 Week 10

Recommendation Systems Introduction

Dr Stephen Tierney

The University of Sydney Business School

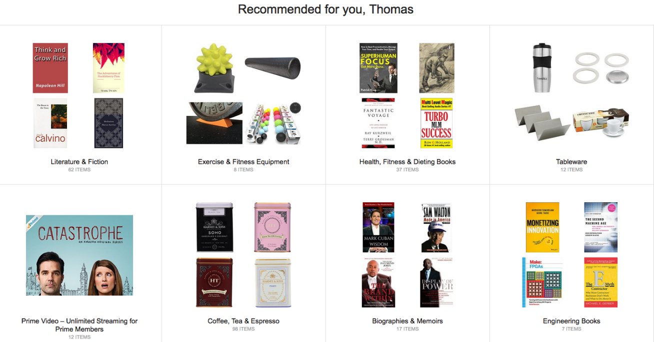
Reading

- A Survey of Collaborative Filtering Techniques
- Empirical Analysis of Predictive Algorithms for Collaborative Filtering

Recommendation Systems

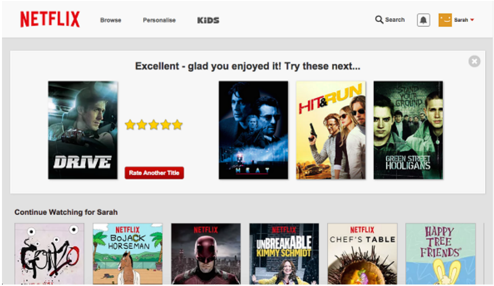
A recommendation system suggests items to users.

Suggestions should be relevant to the user.



Recommendation Systems

The classic example is **Netflix**, who provides a tailored list of recommendations to each customer based on their past viewing behaviour (among other factors).



Accurately predicting customer tastes is so essential to Netflix's business that they ran competitions to try and find the best solution, see "The Netflix Prize".

Recommendation Systems

Classical recommendation systems consist of two stages.

For a target user:

- **Learning**: find users who have similar taste to the target user and estimate outcomes for items the target user hasn't interacted with.
- **Recommending**: identify "best" items to recommend to target user and present them.

Problem Statement

The core idea of a recommendation system is

> *estimate the rating a user* would give *to an item if they were to interact with it*

We can infer this from the user's behaviour and the behaviours of similar users.

Uplift

Users might interact with an item even if we don't recommend it to them!

Therefore only recommendations should be made that increase the probability of interaction.

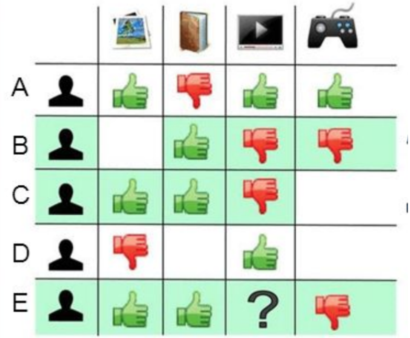
$$\begin{aligned}\text{Uplift} = & \; P(\text{Interact}|\text{Recommended}, \text{User}) \\ & - P(\text{Interact}|\text{Not Recommended}, \text{User})\end{aligned}$$

The recommendations that result in Uplift should then be sorted by domain criteria e.g. expected profit or change in lifetime value from that interaction.

Collaborative Filtering

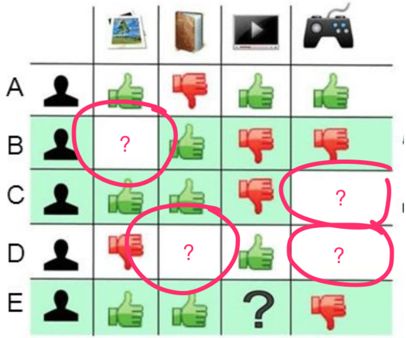
Collaborative Filtering

Collaborative filtering is a broad term to describe many algorithms used in recommendation systems. At the core of collaborative filtering is the **item-user matrix**.



Formalising the Problem

The goal: predict the values which are blank in the user-item matrix.



Baseline Algorithms

Baseline Algorithms

If there are users in our dataset with similar tastes then the simplest approach to predicting a user-rating is to use the ratings given by other users.

Let $r_{u,i}$ be the rating that user u gives to item i and U the set of all users

$$r_{u,i} = \mathrm{aggr}_{u' \in U}(r_{u',i})$$

where the aggregation function could be the weighted or unweighted mean, median or some other measure.

Baseline Algorithms

What happens if we average ratings across all users?

We will end up recommending the same items to every user!

Instead we must use the ratings of users which have **similar preferences** to user u.

KNN

Find users which are similar via **KNN**. Use this subset to predict the rating.

$$r_{u,i} = \mathrm{aggr}_{u' \in N_{u,k}}(r_{u',i})$$

where $N_{u,k}$ is the set of k nearest neighbours to u.

Similarity Weighted

Weight other users by their similarity to u, so that less similar users contribute less

$$r_{u,i} = \mathrm{aggr}_{u' \in U} w_{u,u'}(r_{u',i})$$

where $w_{u,u'}$ is the similarity of user u to u'.

An example similarity score is the cosine similarity. Care needs to be taken since there will be many missing entries in each vector!

KNN and Similarity Weighted

In many cases it is desirable to combine both approaches.

Wrapping Up

User-item Matrix

Analysing the example user-item matrix we can observe a few things:

- the matrix is incomplete and **sparse**, since not every user has an interaction with every item
- potentially some users do not have any interactions (new users), which is known as the **cold start** problem
- the matrix will be **massive** due to a large number of users and items (e.g. Netflix or Amazon)
- a small number of users will have **incoherent tastes** i.e. grey or black sheep, which we can't predict from other users

Disadvantages

The techniques discussed so far are far from perfect:

- user taste is **non-stationary** so the predictions we make will be stale
- hard to keep the models updated since they are not designed to be trained **online**
- often they suffer from **mode collapse** so customers are always recommended the same items or all customers get the same recommendations

Making accurate predictions that are temporal and are novel will likely create more value to the user.