Introduksjon

Hva er recommender systems?

- Sånn som på Youtube og Spotify f.eks. Du får rekommandert videoer og sanger basert på hva du har sett på og hørt på.
- Recommender systems handler mye om hva som gjør oss unike, men også hva som gjør oss like. Kan bruke andres "likings" for å finne andre sine "likings". F.eks person A og B kjøpte produkt C og D, men A kjøpte også E og F. Da kan vi rekommandere E og F til person B.

Hvorfor recommender systems?

- Det er ofte veldig vanskelig å finne hva du ser etter fordi det er så mye info der ute.
- Netflix: 2/3 av sette serier/filmer er fra recommendations. Goolge news - 38%, Amazon - 35% sales fra recommendations.
- Det hjelper brukere til å finne relevante ting.

To fundamentale approaches

- Collaborative filtering - Machine learning
  - To syklere liker samme ting. F.eks Person A og B fra tidligere.
  - "Deep and slow processing"
- Content-based filtering - Information retrieval
  - Her er det mer similarity i content. F.eks hvis en person kjøper noe fra Nintendo, kan vi rekommandere andre ting fra Nintendo. Trenger ikke være samme type ting, men fra samme provider.
  - "Shallow and fast processing". Unstructured, large-scale, dynamic data.
- Har jo såklart også en Hybrid.

Et suksessfult recommender system

- Vanskelig å si. Mange faktorer som spiller inn.

Paradigms of recommender systems

- Personalized recommentations: Ta bruker inn i likningen. F.eks posisjonen til brukeren (for f.eks lokale nyheter)
- Collaborative
- Content-based
- Knowledge-based: "Tell me what fits based on my needs".
- Hybrid: Merge all of the above.

Collaborative filtering

Each user has expressed an opinion for som items.

Explicit opinion: Rating score Implicit: puchase records or listen to tracks

Hvordan kjenner vi brukeren?

Enten ved at h*n er logget inn, eller headers i HTTP requests. Kanskje også cookies.

Neighbourhood based collaborative filtering

> **Main idea**: Similar users display similar patterns of rating behaviuor and similar items recieve similar ratings

There are two main algorithms; **User-based** (assume similar people like similar items) and **Item-based** (have a target item. E.g one friut. Which user should we recommend this friut to? Basically user-based backwards).

We can use **User-item Rating Matrix**. The rows shows users, the columns shows items. Every user "rates" an item and this is placed in the matrix. The matrix are incomplete, not completely filled in. This is called the sparsity problem. The

In Neighbourhood-based collaborative filtering, we have to approaches to this matrix:

1. Predicting the rating a given user will give an item
2. Determining the top-k items (or top-k users)

Recommendation vs Classification

The recommendation problem

> R: rating matrix, **Test data**: Unspecified entries, **Traning data**: Specified entries.

There are different types of ratings. We can have continuous ratings ("slider"), *Interval based ratings* (e.g values between 1 and 5), *Ordinal ratings* ("strongly disagree", "disagree", ... "Strongly agree"), *Binary rating* ("Like", "Dislike") and *Unary ratings* ("like", can be derived from customer action or by **implicit feedback**, e.g watching a movie to the end).

The long-tail property

> "In a real-world setting, only a small fraction of the items are rated frequently"

Most of the profit come by recommending items in the long tail (the least popular items), however, many recommendation algorithms tend to suggest popular items.

It is a challenge recommending items from the long tail

Predicting rates with Neighbourhood-based methods

Main idea: use user-user or item-item similarity to make recommendations.

User-based models: Similar users have similar ratings on the same items.

Item-based methods: Similar items are rated in a similar way by the same user.

User-based Neighbourhood models

Rating matrix R, targer user U and similarity between user U and V.

$\text{Sim}(u, v) = \text{Perason}(u, v)$

= \fraq{\sum*{k \in I*{u} \cap I*{v}}{(r*{uk} - \mu*{u}) \dot (r*{vk} - \mu*{v})}}}{\sqrt{\sum*{k \in I*{u} \cap I*{v}}{(r*{uk} - \mu*{u})^2}}} \dot \sqrt{\sum*{k \in I*{u} \