Applied & generative AI

Xander Vandooren

November 25, 2024

Contents

1 Session 7:

1.1 Recommender systems:

Wat zijn recommender systems?

- Traditional media:
  - Broadcasting is time limited

Grafiek: met recommenders proberen ze dit recht te trekken. Voorbeeld van recommender systems zijn spotify,Amazon,Netflix.

1.1.1 Gevaren:

- Je kan vast zitten in je eigen bubbel.
- Op gegeven moment zelfde genre (bvb bij netflix altijd zelfde genre)
- Marketing

1.1.2 Type input data:

- Items data (bvb in films alles van karakteristieken:genre,acteurs,...)
- Gebruiker data (Expliciete feedback bvb duim omhoog, 4/5 sterren en impliciete data bvb kijk je een film uit of hoeveel afleveringen van serie kijk je uit etc. Impliciete data is vaak belangrijker dan expliciet)
- Interractie items (Hoe reageren we op iets.)

Context: bvb dag in de week of feestdag,geslacht,...

1.1.3 Basic concepts:

- Algorithms
  - Non-personalized
  - Personalized

    * Knowledge-based filtering (KBF) (redelijk algemeen bvb gewoon als man die leeftijds groep is zal hij wss deze film graag zien)
    * Content-based filtering (CBF) (items vergelijken met elkaar)
    * Collaborative filtering (CF) (wss bekentste systeem)
        · User based
        · Item based
        · Matrix factorization
    * Context-aware (CARS)
    * Hybrids
        · Deep learning, Factorization machines...

1.1.3.1 Item-content matrix: attributen $\rightarrow$strenght of connection

1.1.3.2 User-rating Matrix (URM):

- Explicit:
    - ratings, thumbs-up...
- Implicit:
    - Did a user finish the movie?
    - Did he watch the second part?
    - DId he fast-forward?

User-item interaction matrix Meestal is er maar 1% van uw matrix ingevuld (groter dan 0). Bij netflix is de URM density=0.02%.

1.1.3.2.1 Explicit ratings: Large rating scales vs small rating scales:

- Large rating scale:
    - large user effort
    - fewer ratings

- Small rating scale:
  - small user effort
  - more ratings

1.1.3.2.2 Explicit ratings: even vs odd ratings:

- even ratings scale:
  - user forced to express opinion
  - few ratings
- odd ratings scale:
  - trend in giving neutral rating
  - more ratings
  - user more comfortable

1.1.3.2.3 Explicit rating: user bias: bvb. enkel negatieve reviews omdat mensen enkel hun tijd nemen om iets te raten als het slecht is.

1.1.3.2.4 Top popular item: Grootste hoeveelheid ratings

1.1.3.2.5 Best rated item: is eigelijk gewoon formule gemiddelde doen.

1.1.3.2.6 Best rated item with shrink term: Items met veel hoge ratings zouden beter moeten zijn dan items met 1 top rating. Hier moet je een Determince C bias toevoegen dit is wel beetje trial en error om uit te zoeken wat juiste C moet zijn.

1.1.4 Preprocessing: global effects formula:

- Item bias: user will think differently about a movie now than in 20 years. Due to the style, genre, how innovative it is/was..
- User bias: users can rate a movie differently based on nostalgia, because they rate higher on average, because it reminds them of a book or a special moment...

1. Averate ratings:
   - Average ratings for all items and users
2. Normalized rating:
   - To be computed for eacht user u and item i, only on non-zero ratings
3. Item bias:
   - N_i: Number of users who have rated item i to be computed for each item i
4. Recompute rating:
   - To be computed for each user u and item i, only on non-zero ratings
5. User bias:
   - N_u: number of items i rated by user u to be computed for each user u

Final formula: Global effects formula to be computed for each user u and item i, only on non-zero ratings

1.1.5 Evaluating recommender systems:

Kwaliteit is afhankelijk van dataset $\rightarrow$ algorithm $\rightarrow$ user interface

1.1.5.1 Quality indicators:

- Relevance
- Coverage
- Novelty
- Diversity
- Consistency
- Confidence
- Serendipity

1.1.5.2 on-line & off-line evaluation:

1.1.5.2.1 on-line: direct user feedback:

Ask for feedback by a rating, questionnaire.

- 2 problems:
  - Not always reliable opinions
  - Amount of feedback should be large enough

1.1.5.2.2 on-line: A/B testing

Monitor the behaviour:

- Set A: users with a recommendation
- Set B: users without recommendation

Will they behave differently? (Buy more products, spend more time on Netflix...)

- But:
  - Difficult to set up
  - Difficult to interpret (if results bad: lack of relevance or lack of diversity?)

1.1.5.2.3 on-line: controlled experiments: Set up a mock-up application for a group of users

- Test application for a while and gather feedback

But:

- Not a real application, not the same as using real users

1.1.5.2.4 Off-line evaluation 4 steps:

- Defining task
  - Rating prediction (Top-N recommendation)
- Dataset
  - URM
- Partitioning
  - Model: look for connections in URM
  - User Profile: what does user like in general?
  - R: recommendation
  - Metric: compare recommendation with true opinion
  - **Three types of data needed:**
    - Model uses data to detect rules -> model = f(X)
    - Estimated ratings come from the model and user profile -> estimated ratings = g(model, Y)
    - Compare estimated ratings with true value -> estimated rating <-> Z
  - Oppassen voor overfitting!
- Metrics:
  - Quality metrics:
    - Error metrics:
      - MAE & MSE

        * Classification metrics :
            · True positive
            · False positive
            · False negative
            · True negative
        * Ranking metrics:
            · ROC curve:Weten dat er een Area under curve is (AUC).
            · mAP (mean average precision)

1.1.6 Content-based filtering:

compare items based on their attributes. A user that expressed a preference for an item is likely to like similar items.

1.1.6.1 Recommendations based on cosine similarity an element of the ICM is 1 if the item has an attribute else 0.

1.1.6.1.1 Dot product: many attributes in common $\rightarrow$ the two items are very similar

$\hookrightarrow$ Cosine similarity is the normalization of the dot product. If two items are similair their cosine will be large.

1.1.6.1.2 Improving on cosine similarity: shrinking Shrinking:

- Reduce similarity to take into account only most similar with large support
- C: Shrink term

1.1.6.1.3 Estimating a rating: Weighted average of previous given ratings:

- r$_u i$ = the rating of user u on item i
- Choose item with the highest r to recommend
- Or if you want an item of the top-N
- Remove the denominator to save calculation

1.1.6.2 K-nearest neighbours (KNN): Similarity matrix problems:

- Very dense - not many 0-values: computational heavy
- A lot of small similarity values:
  - A lot of very similar small values
  - Hard to distinguish between
  - Noise & low quality

Solution: keep K most similar items

- K influences the quality of the recommendations:
- Small K: not enough data for a reliable estimation
- Big K: too much noise in the data

1.1.6.3 TF-IDF

1.1.6.3.1 ICM improvements: non-binary attributes:

1.1.6.3.2 ICM improvements: attribute weights:

1.1.6.3.3 ICM improvements: TF-IDF: Do we have to manually figure out the attribute weights and non-binary attribute values? No! We can use a technique to do this automatically.

- TF-IDF: Term Frequency - Inverse Document Frequency
- TF-IDF = TF * IDF $\rightarrow$ The weight of each attribute depends on its frequency
- Replace Item-Content Matrix by TF-IDF matrix
- TF: Term frequency:
  - Higher if attribute appears more for an item
- IDF: Inverse document frequency:

- Higher if attribute appears more over all items

TF-IDF: higher value to terms that are frequent for an item, but rare in the whole collection of items. One value per item - attribute pair.

1. Apply TF-IDF on the Item-Content matrix to obtain a TF-IDF matrix
2. onstruct the similarity matrix by applying cosine similarity on the TF-IDF matrix. Also use a shrink term if necessary.
3. pply K-Nearest Neighbours on the similarity matrix

1.1.6.3.4 Pros and cons content-based filtering:

- Pros:
  -

1.1.7 Collaborative filtering:

Often better quality recommendations compared to other methods.Large number of different techniques: user-based and item-based are seen here. Ratings that a user has not explicitly given to an item, can be inferred because therating is correlated across various users and items.

1.1.7.1 User-rating matrix: Matrix containing past interactions between users and items

- Explicit ratings: for example rating from 1 to 5 (0 if missing)
- Implicit ratings: for example 1 if there is interaction with item and 0 otherwise

Implicit information is often more interesting than explicit. Most simple algorithms choose between explicit and implicit.

- Search for similar users and recommend items they like:
  - If two users give similar ratings to several items, we assume that they share the same opinion.
  - Better to have a number between 0 and 1

1.1.8 Model based vs memory based systems:

Model based:when a new user is added, the similarity matrix does not change the model significantly (Item based). Memory based: when a new user is added, the similarity matrix changes (User based).