**Recommendation Engines**

THEY ASSUME THERE IS SIMILARITY, DEPENDENCIES AND PATTERNS BETWEEN HUMANS.AND ALSO BETWEEN CONTENTS

FILTERING ACTIVE: Use patterns in user history to make predictions

* COLLABORATIVE: patterns between users on the basis of their ratings or history and uses this to generate new recommendations (MEMORY BASED – MODEL BASED)
* CONTENT BASED: learns a profile of new user’s interests based on these features
* HYBRID

TYPES OF SIMILARITIES:

-inner product, cosine, pearson correlation

COLLABORATIVE

* Memory based / neighbourhood methods
  + *Item-item filtering-* *first compute the similarity between different items (user ratings) and then make item recommendations based on the items any given user has liked or consumed (USE THE SIMILARITIES TO CALCULATE ratings)*
  + *User-Item filtering* similarity between users. Using that predict rating
* Model based / latent factor methods
  + Cluster based algorithms
  + *Matrix factorisation*
  + Deep learning

IN PRACTICE:

* iF i HAVE RATINGS: <https://github.com/the-mean-square/collaborativeFiltering_fastai/blob/main/Collaborative_filtering_fastai.ipynb>
* IF I DON’T I could measure it in how many views or how many purchases or how many clicks
* [benfred/implicit: Fast Python Collaborative Filtering for Implicit Feedback Datasets (github.com)](https://github.com/benfred/implicit)
* DO THIS CHALLENGE!! [SVD ReRanking: Implicit to Explicit Feedback | Kaggle](https://www.kaggle.com/code/iib2020030/svd-reranking-implicit-to-explicit-feedback)

COURSE: Implicit vs explicit feedback

* implicit uses the user action to measure the preferences
* spotify uses the historical songs that a user has listened to as a measure to suggest new songs
* non personalised recommendations: for example suggest the highest ranked items between the most rated items
* non personalised suggestions: we can pair up items that are read by the same person to make sure that you give a use reading the book the suggestion that they may like the next book
* CONTENT BASED RECOMMENDATIONS: recommend based on past items that the user has liked or interacted with. FIRST STEP is to create a dataset with binaries for each attribute. SECOND create similarities, THIRD FIND USER PROFILES
* COLLABORATIVE FILTERING: find most similar users based on their ratings
* MATRIX FACTOR : mf allow you to deal with real world sparsity
* VALIDATE YOUR PREDICTIONS