Recommendation systems

Agenda

- Why recommendation systems?
- Real world examples and historical trends
- Basic techniques of recommendation
  - Popularity based
  - Classification
  - Content based
  - Collaborative filtering
  - Hybrid approaches
- Evaluation of recommendation
- Python examples
- Advances in recommendation systems

We are overloaded

- Thousands of news articles and news blogs each day
- Millions of movies, books, music tracks online
- Several thousands of ad messages sent to us each day

- But is it new topic?
  - what is new?

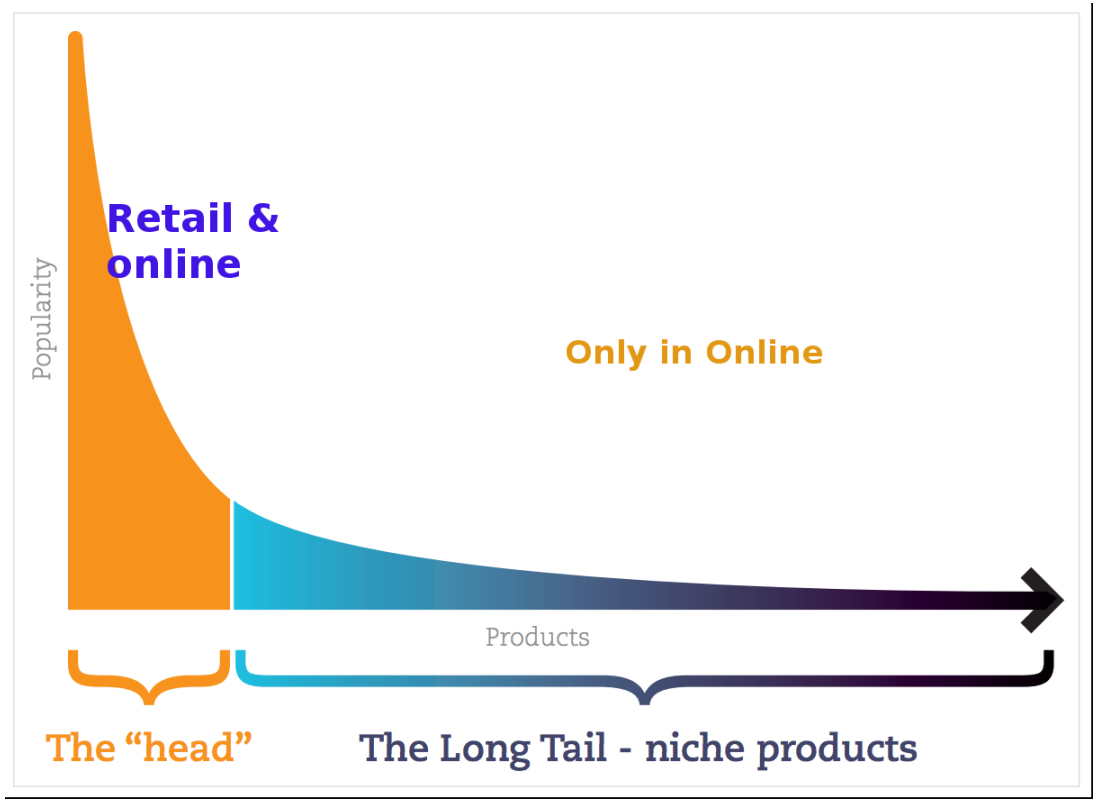
From scarcity to abundance

Limited resources:

- Shelf space
- No of hours for TV shows
- No of theaters for Movies

Web enabled near-zero-cost dissimination of information about products

- Reliance vs Amazon
- **long tail phenomenon**
  - Cut off point
  - Area of the curve



Source: http://dataaspirant.com/2015/05/25/collaborative-filtering-recommendation-engine-implementation-in-python/

Why Recommendation System?

What we can solve?

Help user find item of their interest.

Help item provider deliver their items to right user.

- Identify products most relevant to the user
- Personalized content
- Eg. Top n offers

Help website improve user engagement.

Recommender system creates a matching between users and items and exploits the **similarity between users/items** to make recommendations.

What can be recommended?

Advertising messages

Movies

Books

Music tracks

News articles

Restaurants

Future friends (Social network sites)

Courses in e-learning

Jobs

Research papers

Investment choices

TV programs

Citations

Cloths

Online mates (Dating services)

Supermarket goods

User and matching items

Amazon **Users**: members, **Items**: products, books etc.

Netflix **Users**: members, **Items**: movies, TV shows

LinkedIn **Users**: members, **Items**: members

Facebook **Users**: members, **Items**: jobs

Power of Recommendations*:

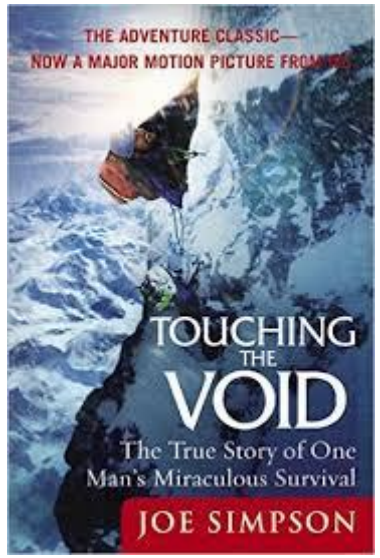
Netflix: – 2/3 rented movies are from recommendation

Google News – 38% more click-through are due to recommendation

Amazon – 35% sales are from recommendation

*(Celma & Lamere, ISMIR 2007)

Recommendation historical trends



Published in 1988

Published in 1996

In 1988, Mr. Joe Simpson, an English mountain climber, wrote a book called Touching the Void. It got good reviews but was considered as modest success and it was soon forgotten.

However, a decade later, an interesting thing happened. Mr. Jon Krakauer wrote a book called Into Thin Air, a thriller about a mountain-climbing tragedy, which became a publishing sensation. Suddenly, Touching the Void started to sell again.

Real world examples

- GroupLens(https://grouplens.org/)
  - Helped in development of initial recommender systems by pioneering initial collaborative filtering models
  - Provided various datasets –MovieLensand BookLens
- Amazon –Did lot of work on implementing commercial recommender systems
  - They also implemented lot of computational improvements
- Netflix Prize
  - Pioneered Latent Factor/Matrix Factorization models
- Google –You Tube
  - Hybrid recommendation systems
  - Deep Learning Based Systems
- Social Network Recommendations

Perfect matching item may not exist

I want a laptop with 500GB HD, 8GB memory and i5 processor for $400

Product #1
- HD: 500 GB
- Memory: 8 GB
- Processor: i7
- Price: $550

Product #2
- HD: 250 GB
- Memory: 4 GB
- Processor: i3
- Price: $350

Product #3
- HD: 250 GB
- Memory: 6 GB
- Processor: i5
- Price: $450

Types of recommendation systems

- Popularity based recommendations
- Classification model based
- Content based recommendations
- Nearest neighbour collaborative filtering:
  - User based
  - Item based
- Hybrid approches
- Association rule mining

Fundamental concepts of similarity

Euclidean Distance

Cosine Similarity:

Cosine Similarity is the cosine of the angle between the 2 vectors A and B

Closer the vectors, smaller will be the angle and hence larger their cosine value.

Pearson Similarity:

Pearson similarity is also another measure of finding similarity between two vectors.

Pearson correlation and cosine similarity are invariant to scaling.

Cosine similarity is NOT invariant to shifts. Pearson correlation is invariant to shifts.

$$Inner(x, y) = \sum_i x_i y_i = \langle x, y \rangle$$

$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} = \frac{\langle x, y \rangle}{||x||\,||y||}$$

$$\begin{aligned} Corr(x, y) &= \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2}\sqrt{\sum (y_i - \bar{y})^2}} \\ &= \frac{\langle x - \bar{x},\ y - \bar{y} \rangle}{||x - \bar{x}||\,||y - \bar{y}||} \\ &= CosSim(x - \bar{x}, y - \bar{y}) \end{aligned}$$

Correlation is the cosine similarity between centered versions of x and y & between -1 to 1

Fundamentals

Jaccard Similarity:

Similarity is defined as the number of users which have rated item A and B divided by the number of users who have rated either A or B

Intersection over Union

It is used for comparing both similarity and diversity of two sets

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

If x,y are two vectors with all real xi, yi then jaccardian similarity coefficient:

$$J(\mathbf{x},\mathbf{y}) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)},$$

The Jaccard distance, which measures dissimilarity between sample sets, is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1

Popularity based Recommender System

Popularity based recommendation system works by recommending items viewed/purchased by most people and rated high

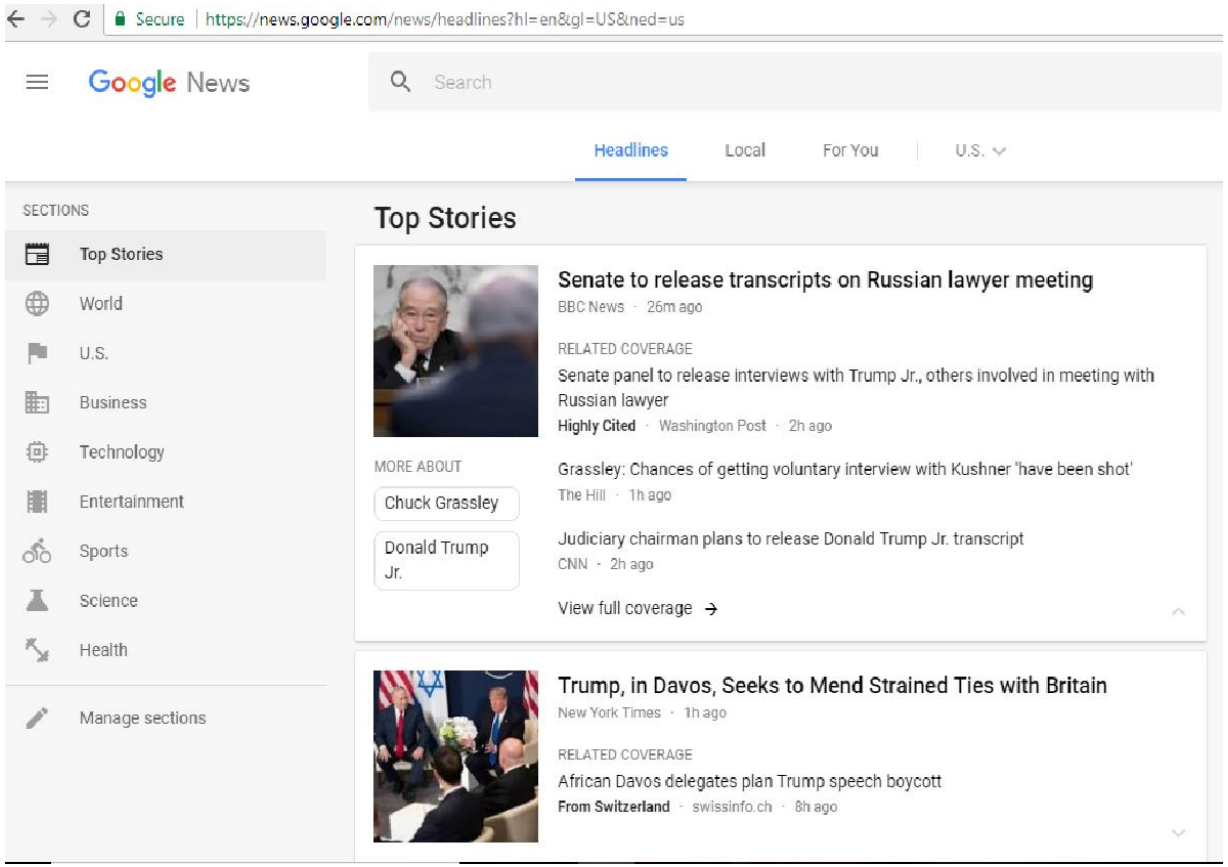
Recommendations: **Ranked list of items by their purchase count / viewed count**

“Popular News”

It uses:

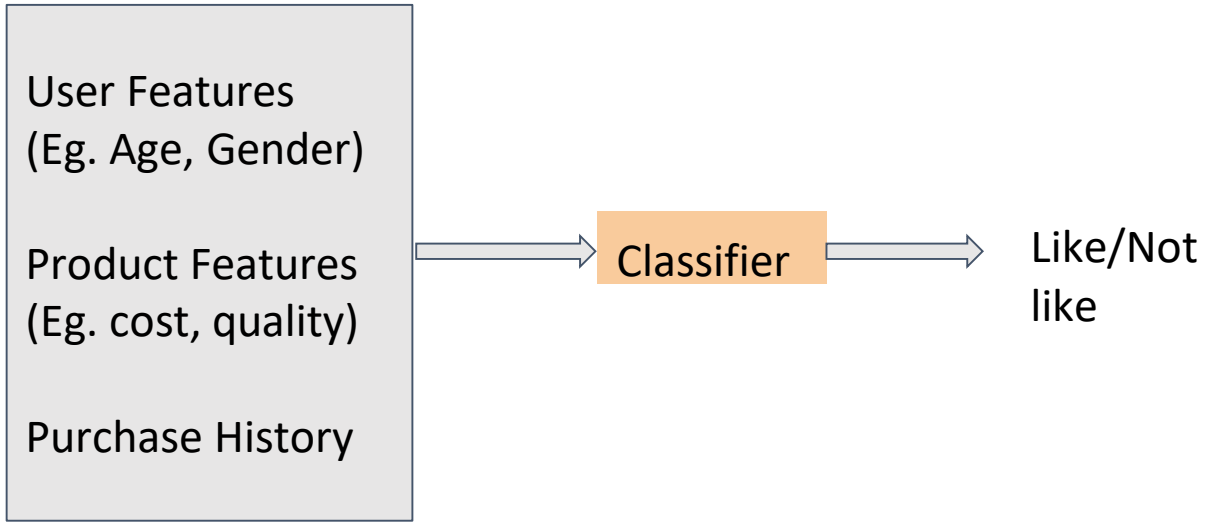
- Can use context
- Purchase history
- User and item features
- Scalale

Not a personalized recommendation



Classification model

Uses features of both products as well as users in order to predict whether a user will like a product or not.



Limitations.

1. It is not easy to collect **quality information** about products and users.
2. Even if we are able to collect good information, they may not be sufficient to make a good classifier
3. Scalability issue

- The outcome can be 1 if the user likes it else 0
- Incorporates personalization

Content based recommendations

Recommendations are based on information on the content of items rather than on other users' opinions.

Main idea:

If a user likes an item then he/she will also like a "similar" item

Recommend items based on their similarity:

- Recommend items to customer x similar to previous items rated highly by x

Uses a machine learning algorithm to induce a profile of the users preferences from examples based on a featural description of content.

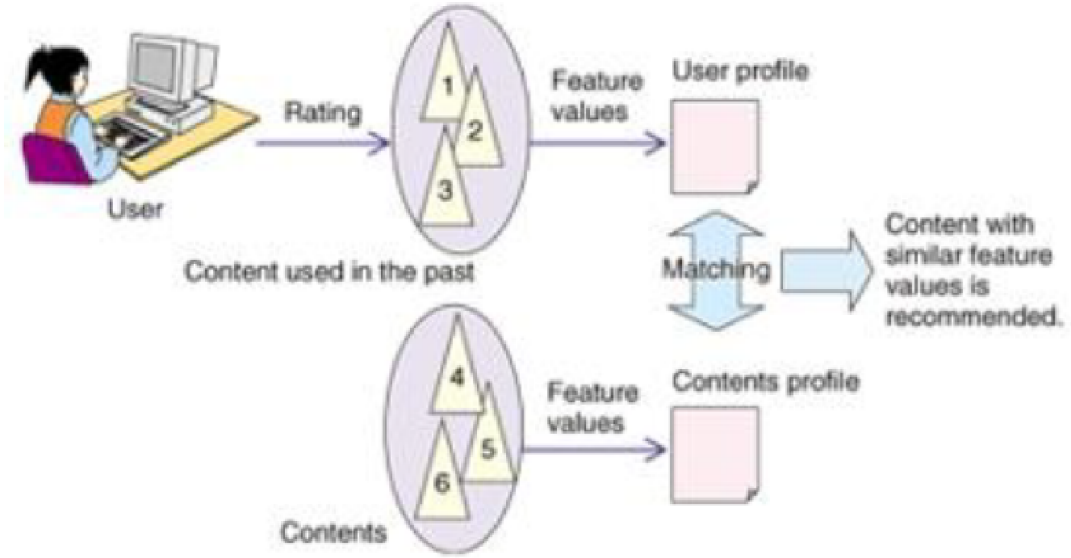
No need for data on other users.

- No cold-start or sparsity problems.

Able to recommend to users with unique tastes.

Techniques that can be used: Cosine similarity

Content based recommendation system



Item Profiles:

- set of features
- vector

User profiles:

- Weighted average of rated item profiles
- Normalize rating using average rating of user

Source: cs.uky.edu

Content based recommendation system

Advantages

- Content-based recommender systems **don’t require a lot of user data**.
- We just **need item data** and you’re able to start giving recommendations to users.
- Recommendation engine does not depend on lots of user data, so it is possible to give recommendations to even your first customer as long as you have adequate data to build his user profile. **Does not suffer from cold start**
- **Less expensive to build and maintain**

Challenges

- Your item data **needs to be well distributed**
- **Availability of features** which explain Items and user preferences
- Recommendations will likely be **direct substitutes**, and not complements, of the item the user interacted with. This is one of the key reasons why collaborative filtering provides better recommendations
- **Less dynamic**