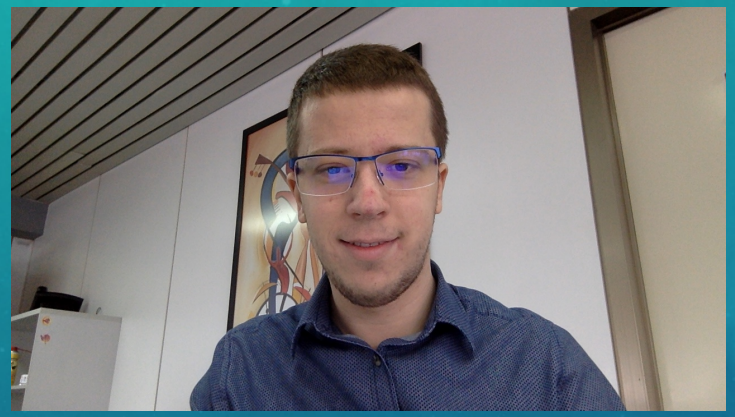
RECOMMENDER SYSTEMS

WHAT, WHY, HOW

Gorjan Zajkovski

JavaSkop ‘17, 11.03.2017, Skopje

Who am I



- Software Engineer working in the field of Data Science
- **3 years** of experience in working on data platforms, and applications that take advantage of machine learning techniques

What I’ll talk about

- **What** is a Recommender System (Engine)
- Brief history
- **Why** do we need one
- **How** do we build one
- How do we know we’ve build a **good** one

What is a Recommender System (Engine) ?

What is a Recommender System (Engine) ?

- A **piece of software** whose job is to:

What is a Recommender System (Engine) ?

- A **piece of software** whose job is to:
  - **Analyze data** and discover **patterns**

What is a Recommender System (Engine) ?

- A **piece of software** whose job is to:
  - **Analyze data** and discover **patterns**
  - **Build models** that **describe** user behavior

What is a Recommender System (Engine) ?

- A **piece of software** whose job is to:
  - **Analyze data** and discover **patterns**
  - **Build models** that **describe** user behavior
  - Use these models to **improve** and **enrich the user experience**

What is a Recommender System (Engine) ?

- A **piece of software** whose job is to:
  - **Analyze data** and discover **patterns**
  - **Build models** that **describe** user behavior
  - Use these models to **improve** and **enrich the user experience**
  - Create a **personalized environment** for the user, on the system

What is a Recommender System (Engine) ?

- A **piece of software** whose job is to:
  - **Analyze data** and discover **patterns**
  - **Build models** that **describe** user behavior
  - Use these models to **improve** and **enrich the user experience**
  - Create a **personalized environment** for the user, on the system
  - **Adjust that model** by analyzing the user-system interactions and by "**learning",** by means of incorporating **user feedback**

amazon.com

Help | Close window

Recommended for You

High Performance Web Sites: Essential Knowledge for Front-End Engineers
by Steve Souders (Author)
Our Price: $19.79
Used & new from $16.24

I own it
Not interested

Add to Cart | Add to Wish List

Because you purchased...

Programming Collective Intelligence: Building Smart Web 2.0 Applications (Paperback)
by Toby Segaran (Author)

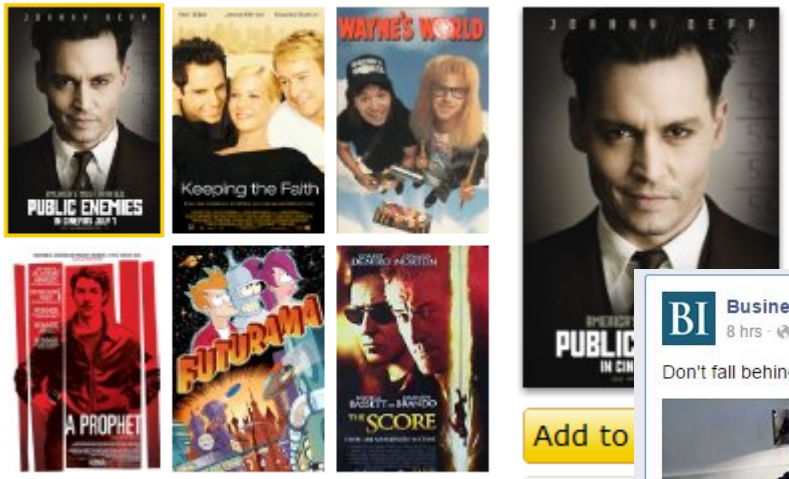
Frequently Bought Together

Price Fo
Add al
Show avai

This item: Data Mining: Practical Machine Learni
Ian H. Witten Paperback $42.09
Data Mining: Concepts and Techniques, Third Edi
Mining the Social Web: Analyzing Data from Face

Recommended for you

Learn more



Public Enemies (2009)
R Biography | Crime | Drama | His...
7.0/10
The Feds try to take down notorious American gangsters John Dillinger, Baby Face Nelson and Pretty Boy Floyd during a booming crime wave in the 1930s.

Add to
Not

◄ Back Next 6 ►

Google to Launch +1 Recommendations On Sites

MOMENTS AGO BY BAS VAN DEN BELD NO COMMENTS

Tweet 0 Like +1 0 Submit Buffer 0 Share Pin it

Google is about to roll out a
sites. Until now the button w
could share through it as we

The recommendations will b
usage of the +1 button witho

The feature will show other
on the behavior of your frien
that will show up.

Publicly recommend this as Bas van den Beld.

RECOMMENDED ON WWW.STATEOFSEARCH.COM

The Ultimate Cookie Legislation Guide
Everything you need to know about the...
+15 people recommend this

Awesome Marketing and Business Boo...
Looking for some reading material for t...
+2 people recommend this

Google Supporting Gay Marriage With ...
Google shows a 'gay bar' below the sea...

BI Business Insider
8 hrs

Don't fall behind.

6 tech skills you need to stay relevant in 2015
How to stay relevant.
WWW.BUSINESSINSIDER.COM

Unlike · Comment · Share · 1,173 30 738

You and 1,172 others like this. Top Comments

RELATED

The Telegraph
Ivana Nikoloska and 4 other friends like this.
Like

India Today
5,658,403 people like this.
Like

The Atlantic
Bojana Besova and 3 other friends like this.
Like

Bloomberg Business
Anastasija Reskova and 18 other friends
Like

Why do **we need** Recommender Systems?

- In the eyes of the **service provider**
  - ⇒ **Helps in deciding what kind of offerings should be made to the user**
  - ⇒ **Helps in guiding user attention**
  - ⇒ **Helps in presenting the user with the desired customization**

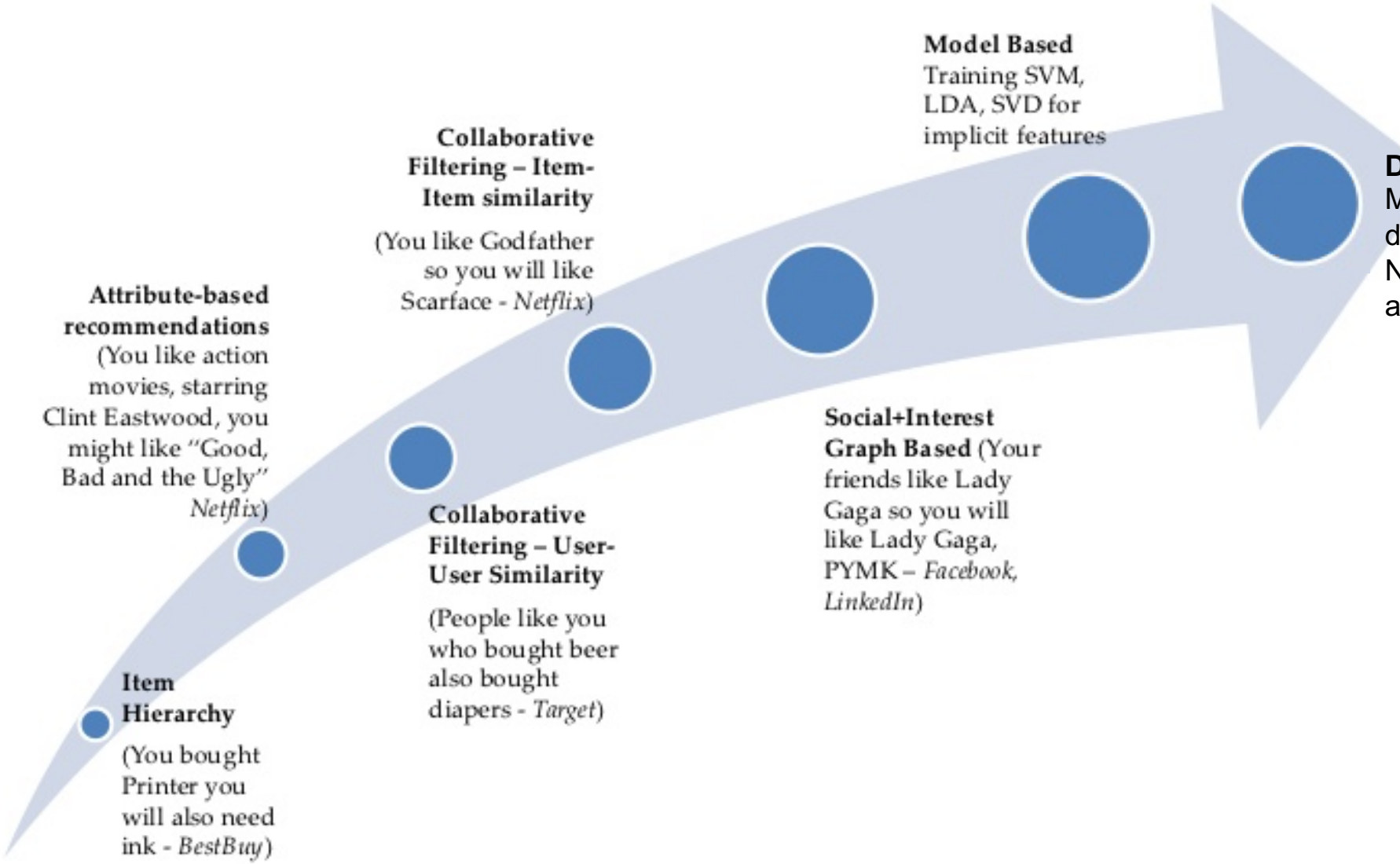
- In the eyes of the **user**
  - ⇒ **Helps in finding a particular item**
  - ⇒ **Helps in comparing / choosing among a large number of items**

Personalized / Non-Personalized

- Is this item important / attractive **in general**?
  - Top N overall, Trending now, Association rules

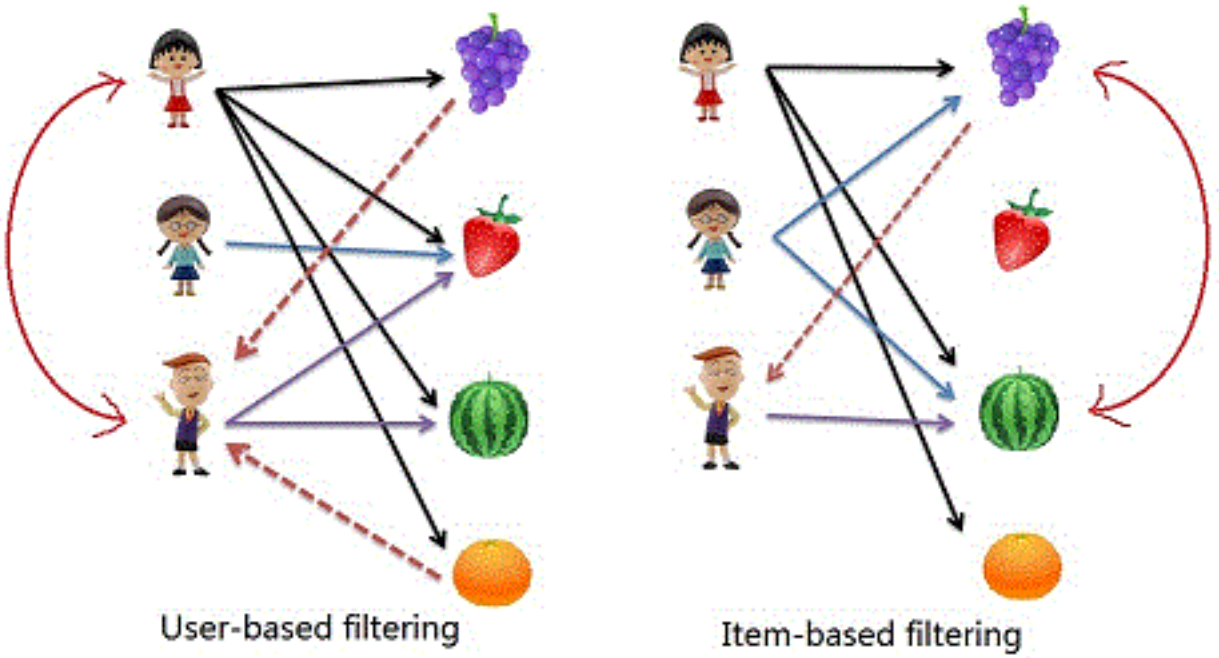
- Is this item important / attractive **for this user**?
  - Because you liked that, you might like this
  - Because your friend liked that, you might like this
  - Because this is similar to something you liked

Recommender Approaches



Most common approaches

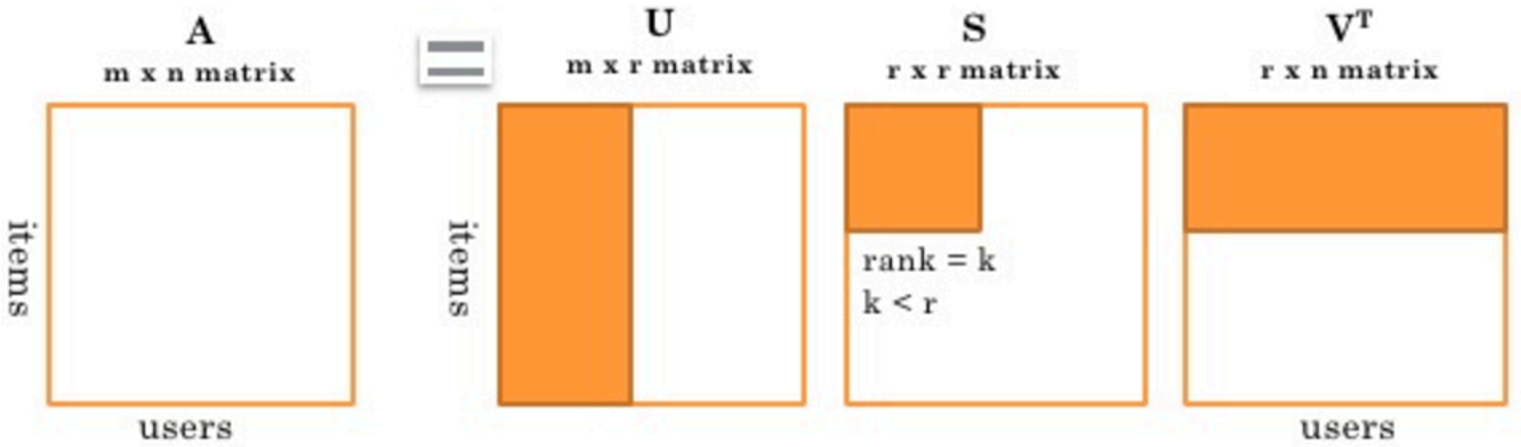
- **Statistical methods**
- **Collaborative filtering techniques**
  - **Item – Item based**
  - **User – User based**
  - **Hybrid** combination
- **Matrix Factorization techniques (SVD, Deep Learning)**



Credits:
http://www.salemmarafi.com/wp-content/uploads/2014/04/collaborativeFiltering-960x540.jpg

Singular Value Decomposition (SVD)

$$A = U \times S \times V^T$$



$$A_k = U_k \times S_k \times V_k^T$$

- Low rank approx. Item profile is $U_k * \sqrt{S_k}$
- Low rank approx. User profile is $\sqrt{S_k} * V_k^T$
- Low rank approx. Item-User matrix is $U_k * \sqrt{S_k} * \sqrt{S_k} * V_k^T$

Credits:
https://image.slidesharecdn.com/4772391/95/building-a-recommendation-engine-an-example-of-a-product-recommendation-engine-21-728.jpg

How do we **build** one?

Data

- Users
- Events
- Actions

Features

- Feature Engineering
- Entity relationships

ML alg.

- Choose wisely for the use-cases

Predictions

- Generate predictions (ranking…)

Validate

- Test the system in the wild
- Incorporate user feedback


data
mining
Data
information
analysis
software
warehousing
relationships
knowledge
different
determine
advances
organizations
databases
companies
patterns
Information
beer
grocery
example
Thursdays
supermarket
used
sales
enables
analyze
transactional
retail
Knowledge
computer
power
storage
diapers
customer
many
bought
process
promotional
Saturdays
increase
processing
among
discovered
provide
buying
factors
revenue
various
analyzing
economic
cost
chain
consumer
Dramatic
relatively
one
years
new
retailer
term
items
summary
price
access
users
vision
display
efforts
However
meta
transmission
equally
converted
historical
Continuous
shoppers
inventory
integrate
freely
detail
capture
light
purchased
dozens
processed
shopping
repository
drill
positioning
maximize
showed
summarize
available
indicators
Example
Centralization
men
formats
make
competition
fields
use
selling
however
typically
although
accumulating
external
payroll
technological
text
upcoming
ways
facts
future
vast
large
trends
driving
defined
categorize
weekly
view
sift
Oracle
financial
sale
increasing
Warehouses
volumes
summarizing
discovery
Finally
primarily
Midwest
number
today
reports
centralized
nonoperational
needed
communication
demographics
behavior
supports
numbers
dramatically
accounting
identified
forecast

First, we get the **data**!

In order to recommend an item to a user or a group of users, you need to have:

- an offering of items, and
- users to which to recommend those items

Step 1:

Aggregate, structure and analyze the data that the system holds. Some visualizations might point to particular trends, and the size and volume of data might give clue to what algorithm is most appropriate.

Define the features

- **Decide on:**
  - What is known for the user entity?
  - What is known for the item entity?
  - What actions does the system track, that might offer useful information?
  - What is known for the relationship between the entities?

Step 2:

Define the features of the entities and the features of the relationship between the users, items and actions in the system. Make the features as descriptive as possible. Sometimes a mixture of attributes can be a feature also.

Choose an Algorithm

- **Research, experiment and decide:**
  - Whether the recommender system will be personalized / non personalized
  - Whether it will use collaborative filtering, or factorization techniques
  - Which machine algorithm will be leveraged (supervised / unsupervised learning) ?

Step 3:

Build on the shoulders of giants. See what has been used before, to solve a similar problem. Research and experiment with different algorithms, and techniques.

Build models and generate predictions

- **Use the features / algorithm decision and implement a model**
- **Use the data to train the model**
- **Use the model to generate predictions**
- **Check the performance of the model**

Step 4:

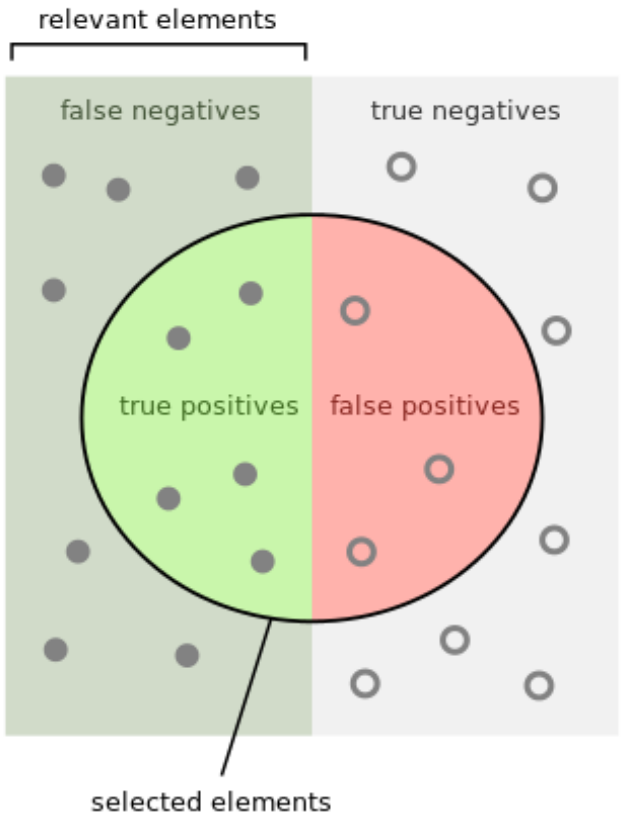
Build and train a model. Build more than one model. Compare the predictions and evaluate the models.

Test / Validate the models

- **Use the built models, to show predictions (recommendations) to real users**
- **Monitor the response / actions of the users**
- **Make sure to provide as much options as possible for the users to provide feedback**

Step 5:

Test the built models with real users. Start off with something small like a “what you might like” section. Incorporate user feedback in order to improve model performance.



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

$$F_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

The cold start problem…

- What if there is no sufficient data for users / items in the system?

The cold start problem…

- What if there is no sufficient data for users / items in the system?

⇒ Start with general recommendations, non-personalized, the same for every user.

The cold start problem…

- What if there is no sufficient data for users / items in the system?

⇒ Start with general recommendations, non-personalized, the same for every user.

⇒ Link the user account with some social network, and ask for data

The cold start problem…

- What if there is no sufficient data for users / items in the system?

⇒ Start with general recommendations, non-personalized, the same for every user.

⇒ Link the user account with some social network, and ask for data

⇒ Provide quick and easy ways for users to let you know more about them

Offer ways for the user to give **feedback**

- Like / dislike
- Rate
- Swipe to remove / hide (don’t show me items like this)
- Was this helpful?

- **Track implicit feedback, as well as explicit…**

=> If the user was presented with 5 recommendations, and the user choose the third one, maybe the first two deserve a bad mark...

Gotchas…

- Uneven rankings from different people (one mans' 3 can mean the same as other mans' 5, on a 1-5 scale)
- Temporal data (what is relevant today, may not be as relevant tomorrow)
- What scale best captures your items nature 0-1 (like, no like), 1-5, 1-10, 0-100?
- Are user ratings reliable? Would the same user rate the same item with the same grade today, as it did 1 week/month/year ago?

Gotchas…

- Uneven rankings from different people (one mans' 3 can mean the same as other mans' 5, on a 1-5 scale)

=> Normalize by subtracting the average rating from that users' ratings

- Temporal data (what is relevant today, may not be as relevant tomorrow)

- What scale best captures your items nature 0-1 (like, no like), 1-5, 1-10, 0-100?

- Are user ratings reliable? Would the same user rate the same item with the same grade today, as it did 1 week/month/year ago?

Gotchas…

- Uneven rankings from different people (one mans' 3 can mean the same as other mans' 5, on a 1-5 scale)

=> Normalize by subtracting the average rating from that users' ratings

- Temporal data (what is relevant today, may not be as relevant tomorrow)

=> Order the recommendations by date, then take from the top

- What scale best captures your items nature 0-1 (like, no like), 1-5, 1-10, 0-100?

- Are user ratings reliable? Would the same user rate the same item with the same grade today, as it did 1 week/month/year ago?

Gotchas…

- Uneven rankings from different people (one mans' 3 can mean the same as other mans' 5, on a 1-5 scale)

=> Normalize by subtracting the average rating from that users' ratings

- Temporal data (what is relevant today, may not be as relevant tomorrow)

=> Order the recommendations by date, then take from the top

- What scale best captures your items nature 0-1 (like, no like), 1-5, 1-10, 0-100?

=> Very item specific, experiment with the most common few, settle on one

- Are user ratings reliable? Would the same user rate the same item with the same grade today, as it did 1 week/month/year ago?

Gotchas…

- Uneven rankings from different people (one mans' 3 can mean the same as other mans' 5, on a 1-5 scale)

=> Normalize by subtracting the average rating from that users' ratings

- Temporal data (what is relevant today, may not be as relevant tomorrow)

=> Order the recommendations by date, then take from the top

- What scale best captures your items nature 0-1 (like, no like), 1-5, 1-10, 0-100?

=> Very item specific, experiment with the most common few, settle on one

- Are user ratings reliable? Would the same user rate the same item with the same grade today, as it did 1 week/month/year ago?

=> Implement a weight cost function which puts accent on newest actions

It’s the little things…

- “We recommend” vs “what you might like”
- This product has a 4.3 star rating. Out of HOW MANY ratings?
- We recommend this item… Why?