Approaches to Recommendation in Industry

Xavier Amatriain
xavier@amatriain.net
@xamat
Quora/Xavier-Amatriain

Recsys 2017 Summer School

Bolzano, IT

Outline

1. The Traditional Recommender Problem
2. The Netflix Prize
3. Beyond Rating Prediction
4. Lessons Learned
5. A Recsys Architectural Blueprint
6. Building a state-of-the-art recommender system in practice
7. Hands-on tutorial
8. Future research Directions
9. Conclusions
10. Some references

1. The Recommender Problem

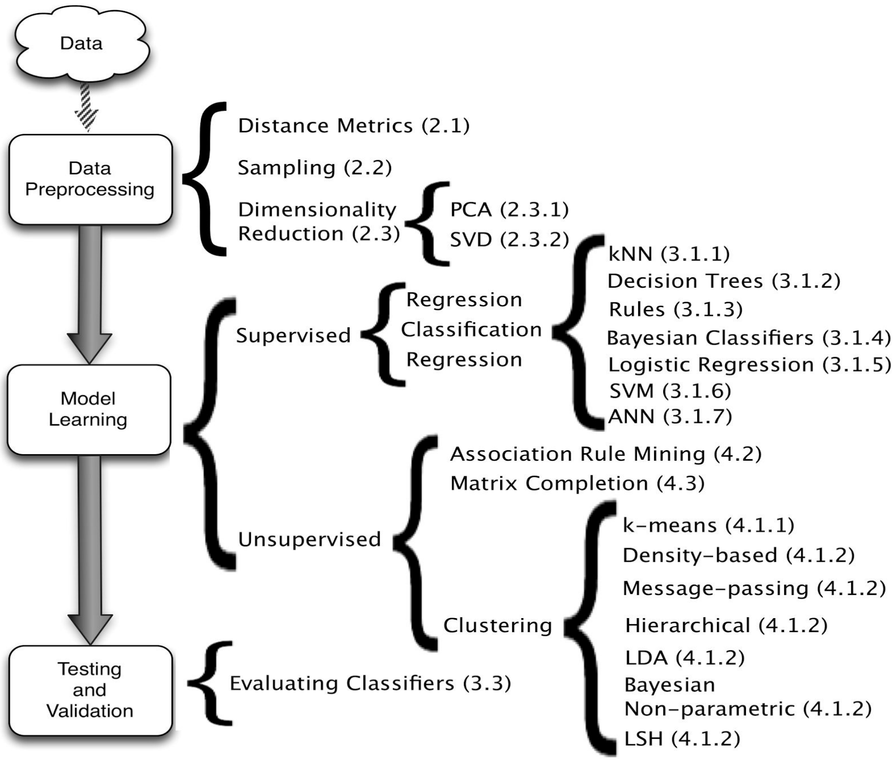
The “Recommender problem”

- Traditional definition: Estimate a utility function that automatically predicts how much a user will like an item.
- Based on:
    - Past behavior
    - Relations to other users
    - Item similarity
    - Context
    - …

Recommendation as data mining

The core of the Recommendation Engine can be assimilated to a general data mining problem

(Amatriain et al. Data Mining Methods for Recommender Systems in Recommender Systems Handbook)



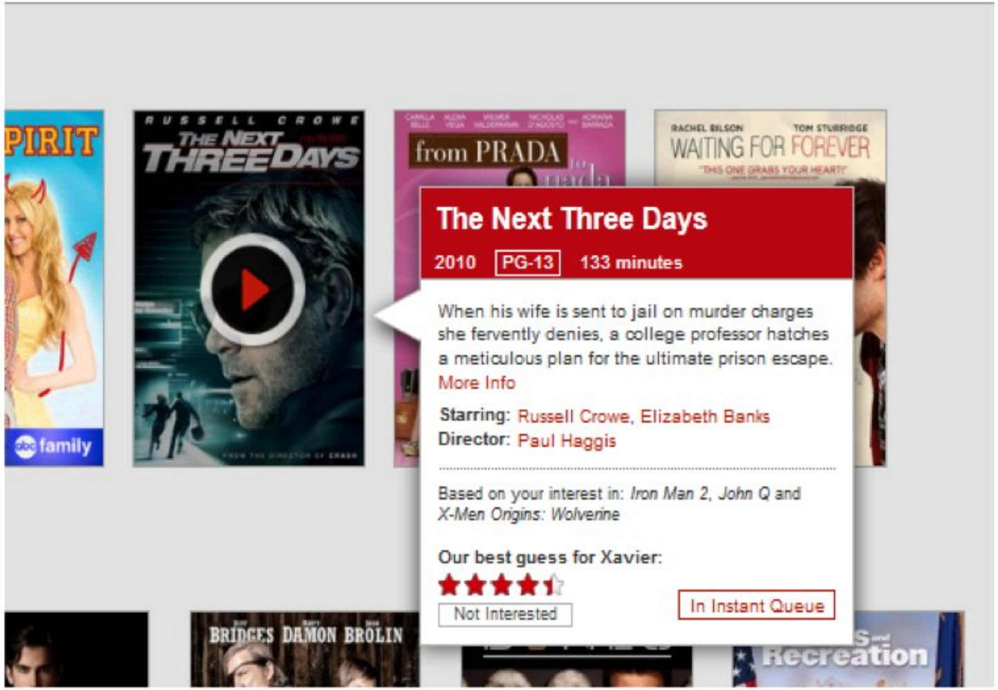
Data Mining + all those other things

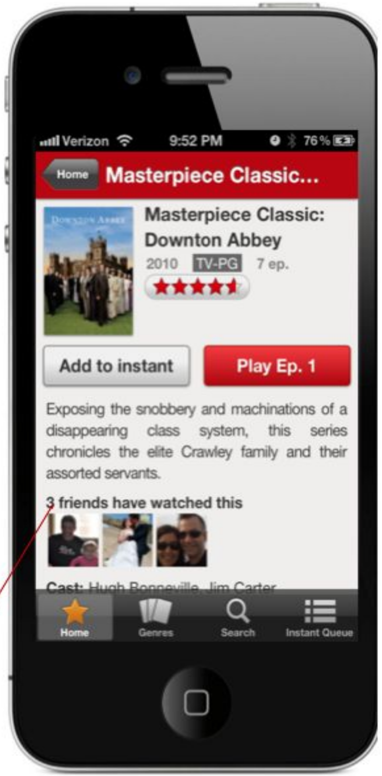
- User Interface
- System requirements (efficiency, scalability, privacy....)
- Serendipity
- Diversity
- Awareness
- Explanations
- ...

Serendipity

- Unsought finding
- Don't recommend items the user already knows or **would have found anyway**.
- Expand the user's taste into neighboring areas by improving the obvious
- Serendipity ~ Explore/exploit tradeoff

Explanation/Support for Recommendations





Social Support

Diversity & Awareness

Personalization awareness

Diversity

What works

- Depends on the **domain** and particular **problem**
- However, in the general case it has been demonstrated that the best isolated approach is CF.
  - Other approaches can be hybridized to improve results in specific cases (cold-start problem...)
- What matters:
  - Data preprocessing: outlier removal, denoising, removal of global effects (e.g. individual user's average)
  - “Smart” dimensionality reduction using MF
  - Combining methods through ensembles

2. The Netflix Prize

Netflix Prize

COMPLETED

What we were interested in:

- High quality *recommendations*

Proxy question:

- Accuracy in predicted rating
- $$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

2007 Progress Prize

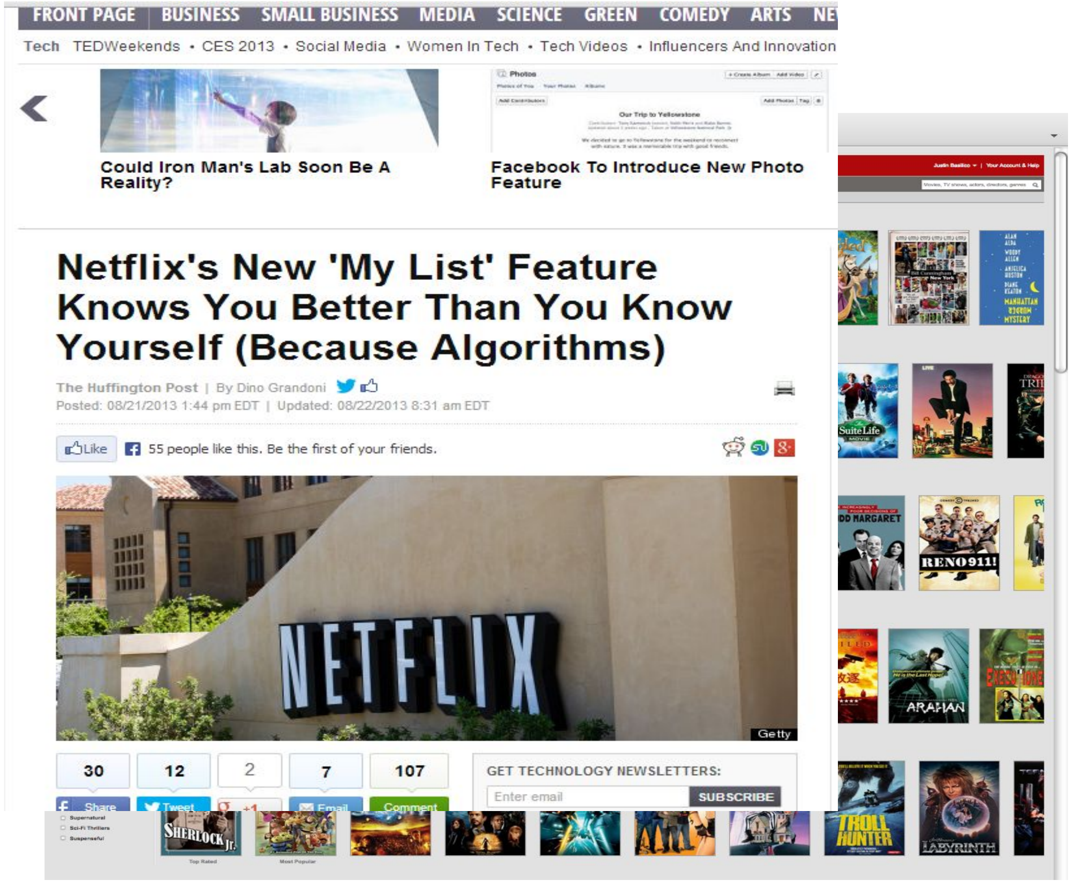
- Top 2 algorithms
  - SVD - Prize RMSE: 0.8914
  - RBM - Prize RMSE: 0.8990
- Linear blend Prize RMSE: 0.88
- Currently in use as part of Netflix' rating prediction component
- Limitations
  - Designed for 100M ratings, not XB ratings
  - Not adaptable as users add ratings
  - Performance issues

What about the final prize ensembles?

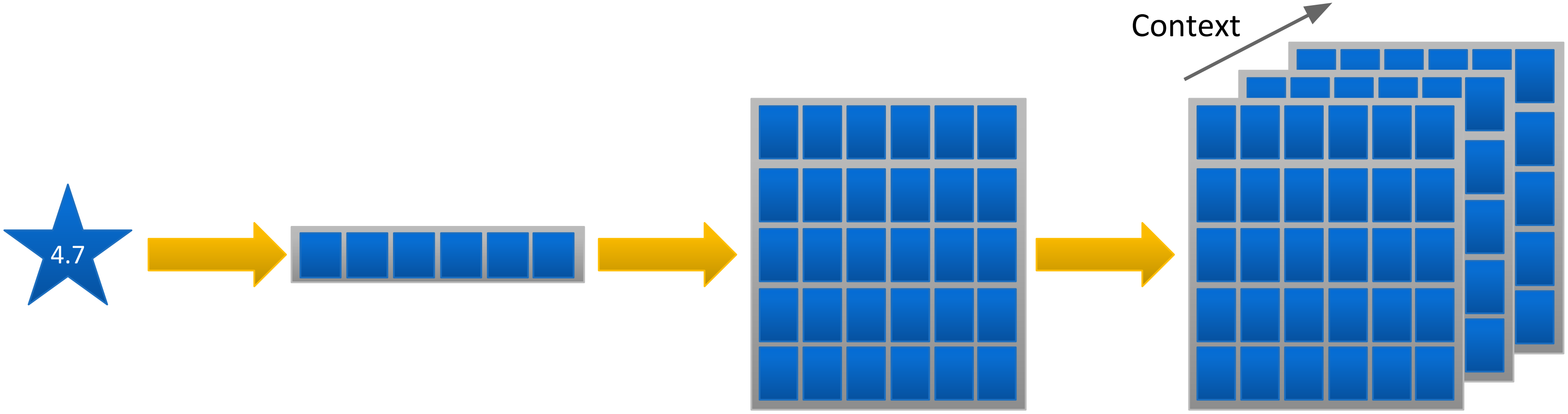
- Offline studies showed they were too computationally intensive to scale
- Expected improvement not worth engineering effort
- Plus.... Focus had already shifted to other issues that had more impact than rating prediction.

3. Beyond Rating Prediction

Everything is a recommendation



Evolution of the Recommender Problem



Rating

Ranking

Page Optimization

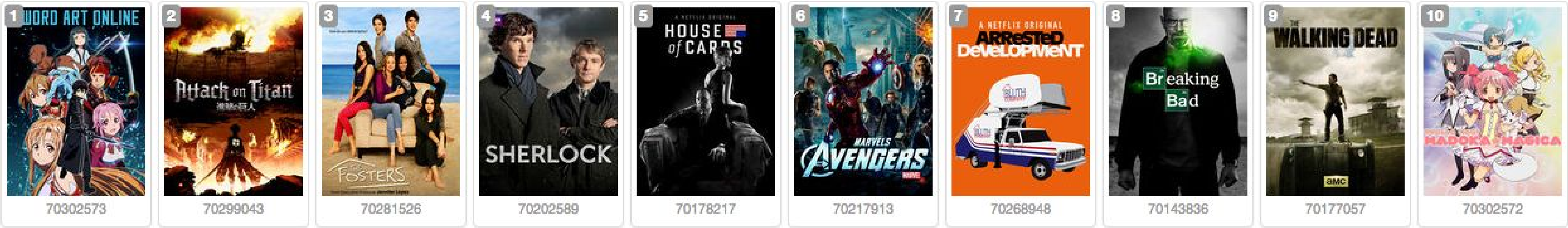
Context-aware Recommendations

3.1 Ranking

Ranking

- Most recommendations are presented in a sorted list
- Recommendation can be understood as a ranking problem
- Popularity is the obvious baseline
- What about rating predictions?

Ranking by ratings



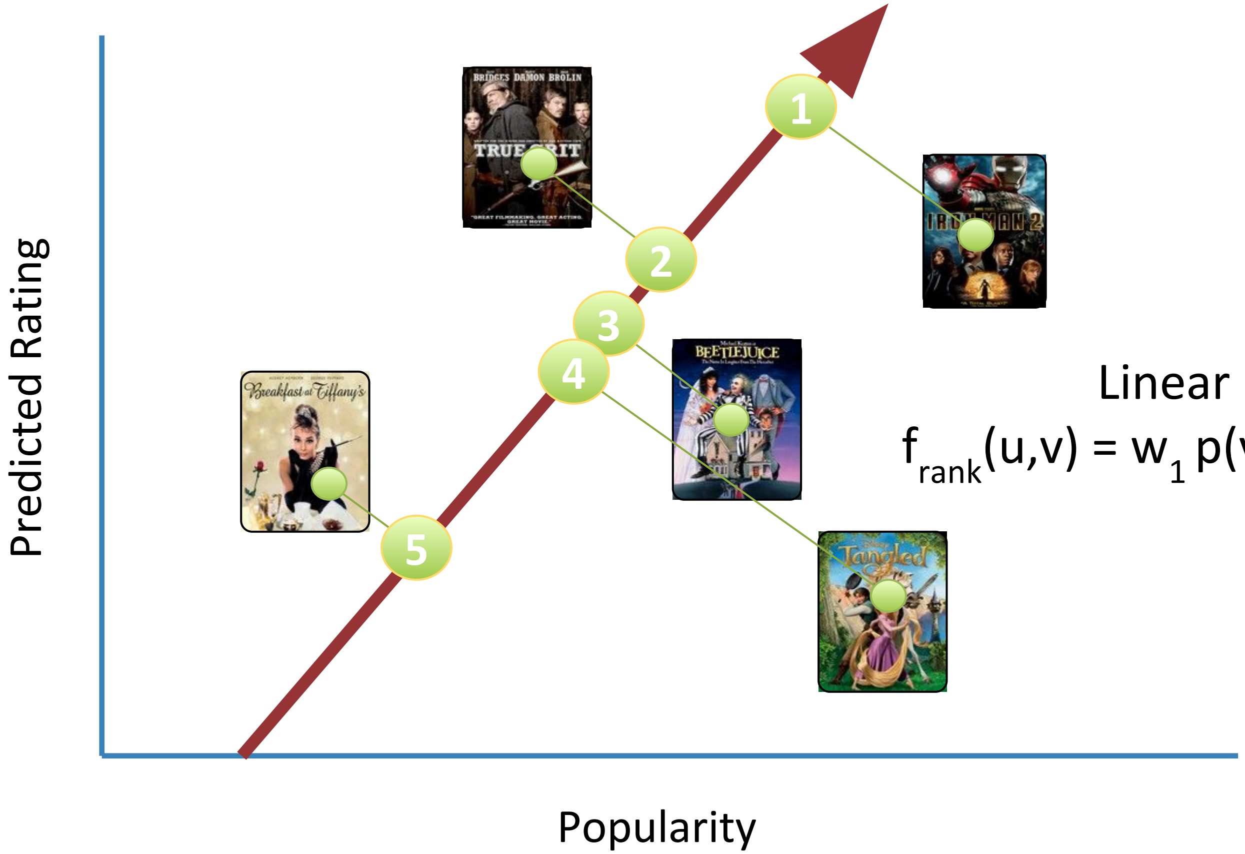
4.7 4.6 4.5 4.5 4.5 4.5 4.5 4.5 4.5 4.5

Niche titles

High average ratings... by those who would watch it


RMSE

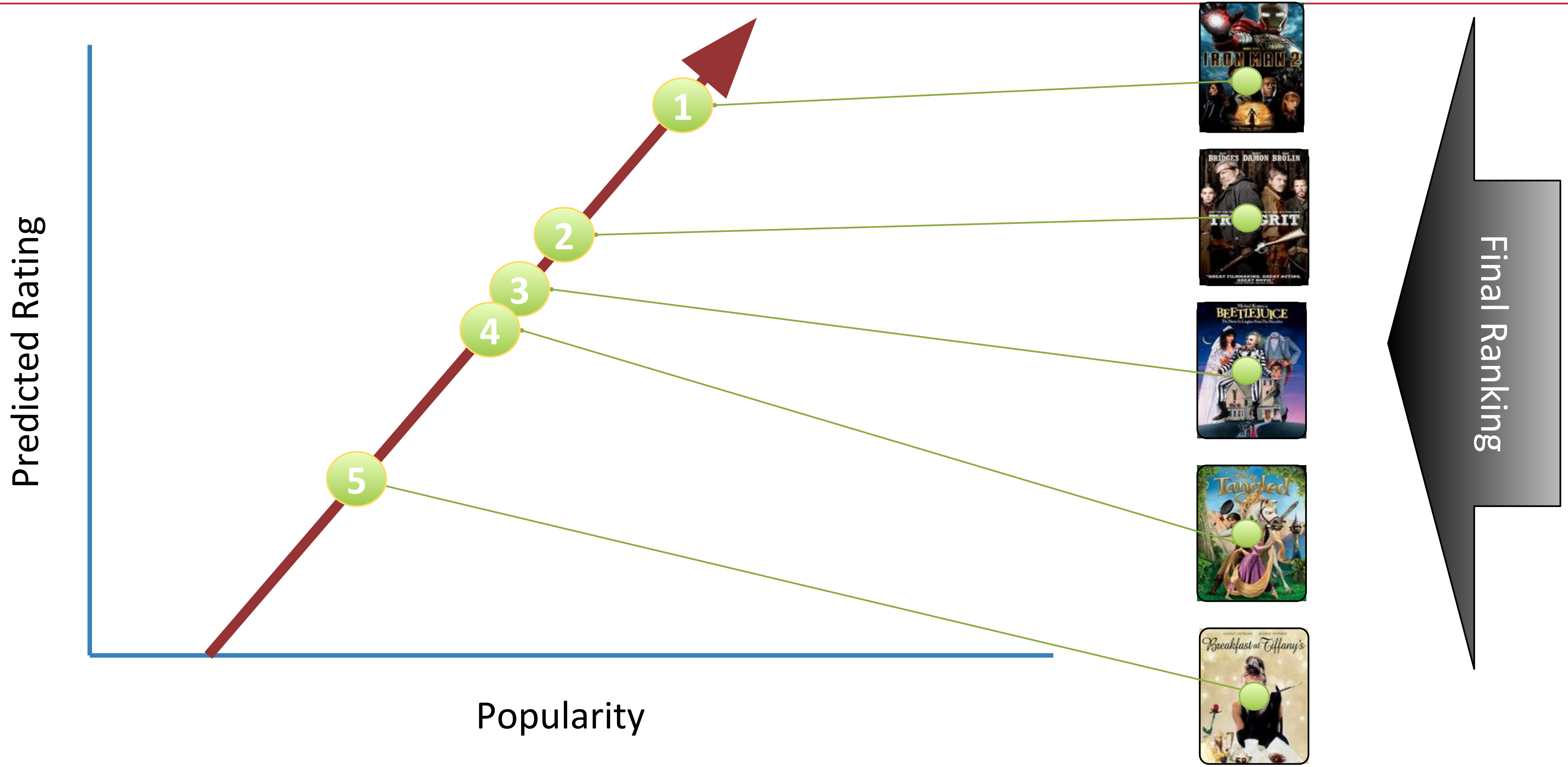
Example: Two features, linear model



Linear Model:

$$f_{rank}(u,v) = w_1\, p(v) + w_2\, r(u,v) + b$$

Example: Two features, linear model



Learning to rank

- Machine learning problem: goal is to construct ranking model from training data
- Training data can be a partial order or binary judgments (relevant/not relevant).
- Resulting order of the items typically induced from a numerical score
- Learning to rank is a key element for personalization
- You can treat the problem as a standard supervised classification problem

Learning to rank - Metrics

- Quality of ranking measured using metrics as
  - Normalized Discounted Cumulative Gain
  - Mean Reciprocal Rank (MRR)
  - Fraction of Concordant Pairs (FCP)
  - Others...
- But, it is hard to optimize machine-learned models directly on these measures (e.g. non-differentiable)
- Recent research on models that directly optimize ranking measures

Ranking - Quora Feed

Goal: Present most *interesting stories* for a

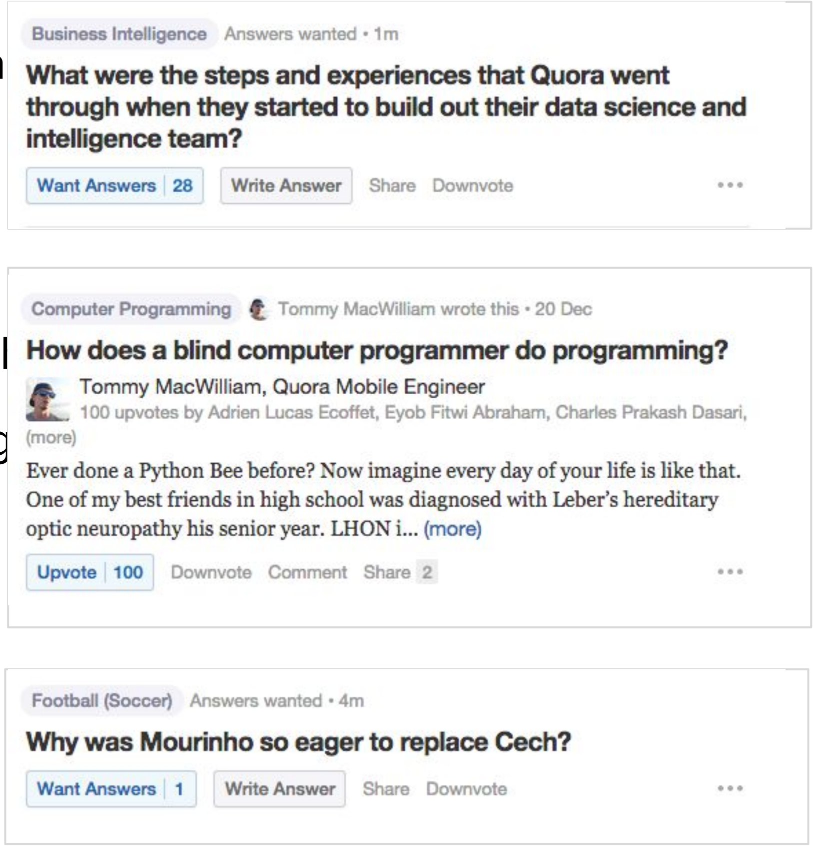
Interesting = topical relevance +

social relevance + timeliness

Stories = questions + answers

ML: Personalized learning-to-rank approacl

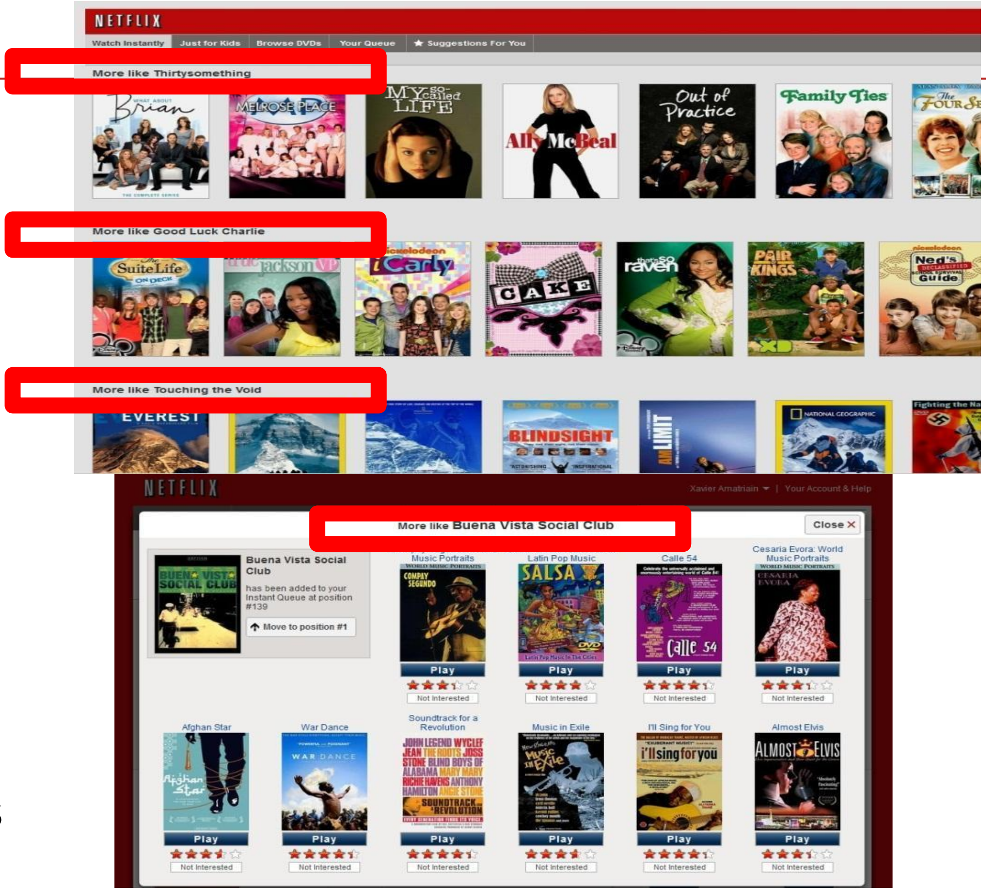
Relevance-ordered vs time-ordered = big g



3.2 Similarity

Similars

- Displayed in many different contexts
  - In response to user actions/context (search, queue add…)
  - More like… rows



Similars: Related Questions

- Given interest in question A (source) what other questions will be interesting?
- Not only about similarity, but also “interestingness”
- Features such as:
    - Textual
    - Co-visit
    - Topics
    - …
- Important for logged-out use case

RELATED QUESTIONS

How do you decide to regularize between L1/L2 or best/greedy subset selection?

What's a good way to provide intuition as to why the lasso (L1 regularization) results in sparse weight vectors?

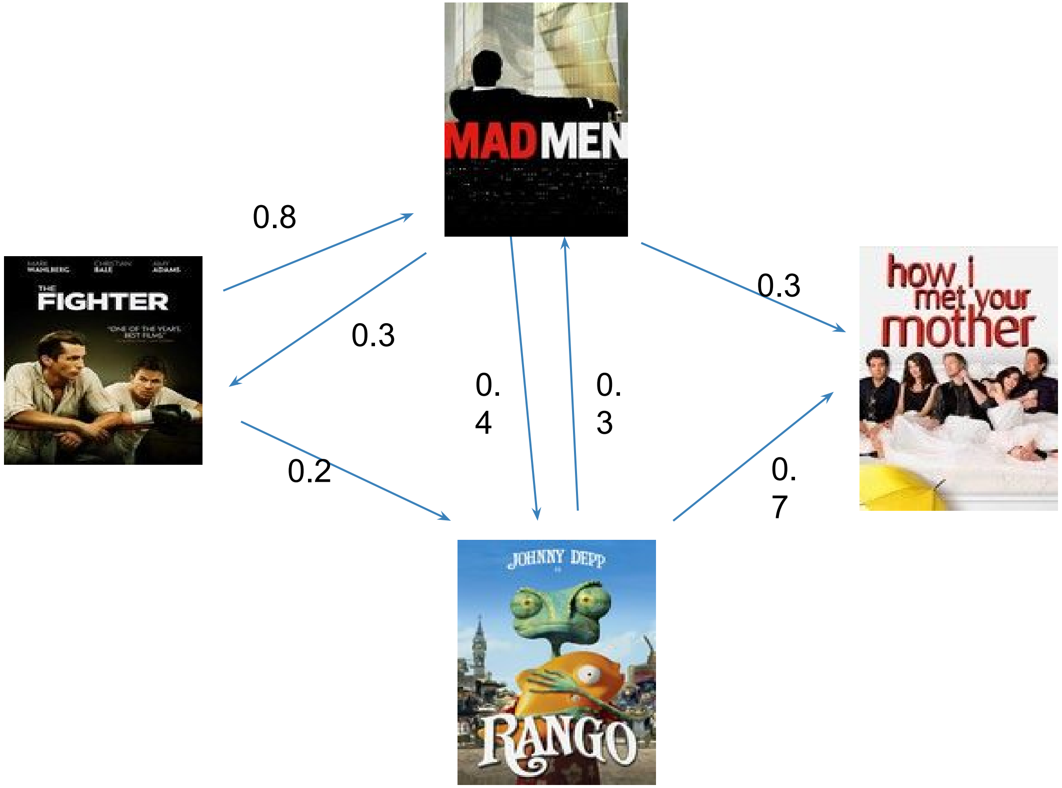
What is the difference between normalization, standardization, and regularization for data?

Why is L1 regularization supposed to lead to sparsity than L2?

What are the conditions of using L1 and L2 regularization respectively?

What are some papers/talks/lectures/notes that give high-level overviews of regularization, especially L1 and L2 regulariz... (continue)

Graph-based similarities



Example of graph-based similarity: SimRank

- SimRank (Jeh & Widom, 02): “two objects are similar if they are referenced by similar objects.”

$$s(a,b) = \frac{C}{|I(a)||I(b)|} \sum_{i=1}^{|I(a)|} \sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b))$$

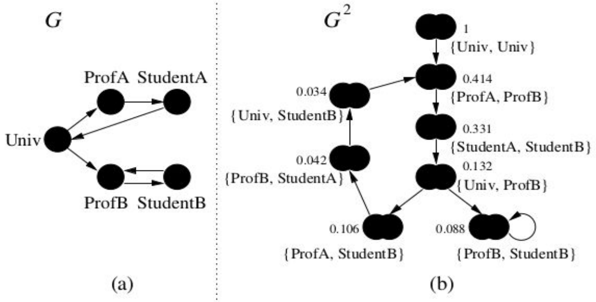


Figure 1: A small Web graph G and simplified node-pairs graph G^2. SimRank scores using parameter $C = 0.8$ are shown for nodes in G^2.

Similarity ensembles

- Similarity can refer to different dimensions
  - Similar in metadata/tags
  - Similar in user play behavior
  - Similar in user rating behavior
  - …
- Combine them using an ensemble
  - Weights are learned using regression over existing response
  - Or… some MAB explore/exploit approach
- The final concept of “similarity” responds to what users vote as similar

3.3 Social Recommendations

Recommendations - Users

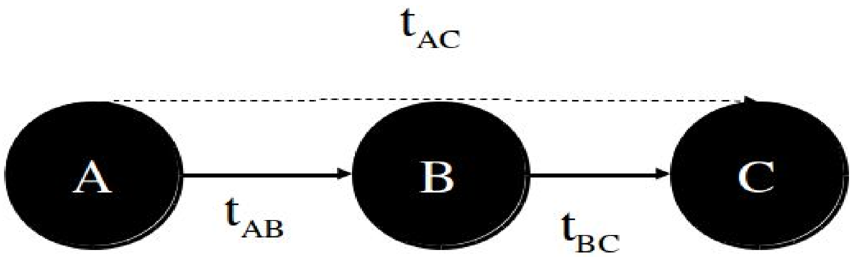
- **Goal: Recommend new users to follow**
- Based on:
  - Other users followed
  - Topics followed
  - User interactions
  - User-related features
  - ...

User Trust/Expertise Inference

- **Goal: Infer user's trustworthiness in relation to a given topic**
- We take into account:
  - Answers written on topic
  - Upvotes/downvotes received
  - Endorsements
  - ...
- Trust/expertise propagates through the network
- Must be taken into account by other algorithms

Social and Trust-based recommenders

- A social recommender system recommends items that are "popular" in the social proximity of the user.
- Social proximity = trust (can also be topic-specific)
- Given two individuals - the *source* (node A) and *sink* (node C) - derive how much the source should trust the sink.
- Algorithms
  - Advogato (Levien)
  - Appleseed (Ziegler and Lausen)
  - MoleTrust (Massa and Avesani)
  - TidalTrust (Golbeck)



Other ways to use Social

- Social connections can be used in combination with other approaches
- In particular, “friendships” can be fed into collaborative filtering methods in different ways
    - replace or modify user-user “similarity” by using social network information
    - use social connection as a part of the ML objective function as regularizer
    - ...

3.4 Explore/Exploit

Explore/Exploit

- One of the key issues when building any kind of personalization algorithm is how to trade off:
  - **Exploitation**: Cashing in on what we know about the user right now
  - **Exploration**: Using the interaction as an opportunity to learn more about the user
- We need to have informed and optimal strategies to drive that tradeoff
  - **Solution**: pick a reasonable set of candidates and show users only “enough” to gather information on them

Multi-armed Bandits

- Given possible strategies/candidates (slot machines) pick the arm that has the maximum potential of *being good* (minimize **regret**)

- Naive strategy: ε-greedy
  - Explore with a small probability ε (e.g. 5%) -> choose an arm at random
  - Exploit with a high probability (1 - ε) (e.g. 95%) -> choose the best-known arm so far
- Translation to recommender systems
  - Choose an arm = choose an item/choose an algorithm (MAB testing)
- Thompson Sampling

  Given a posterior distribution, sample on each iteration and choose the action that maximizes the expected reward

Multi-armed Bandits

Explore-Exploit in Top-N Recommender Systems via Gaussian Processes

Hastagiri P Vanchinathan
ETH Zürich
hastagiri@inf.ethz.ch

Isidor Nikolic *
Microsoft, Zürich
inikolic@microsoft.com

Fabio De Bona
Google, Zürich
fdb@google.com

Andreas Krause
ETH Zürich
krausea@ethz.ch

A Contextual-Bandit Approach to Personalized News Article Recommendation

Lihong Li†, Wei Chu†,
†Yahoo! Labs
lihong,chuwei@yahoo-inc.com

John Langford‡
‡Yahoo! Labs
jl@yahoo-inc.com

Robert E. Schapire+*
+Dept of Computer Science
Princeton University
schapire@cs.princeton.edu

Context Adaptation in Interactive Recommender Systems

Negar Hariri
DePaul University
Chicago, IL 60604, USA
nhariri@cs.depaul.edu

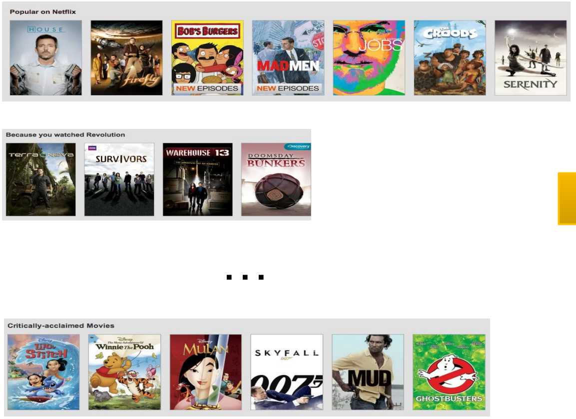
Bamshad Mobasher
DePaul University
Chicago, IL 60604, USA
mobasher@cs.depaul.edu

Robin Burke
DePaul University
Chicago, IL 60604, USA
rburke@cs.depaul.edu

3.5 Page Optimization

Page Composition

10,000s of possible rows



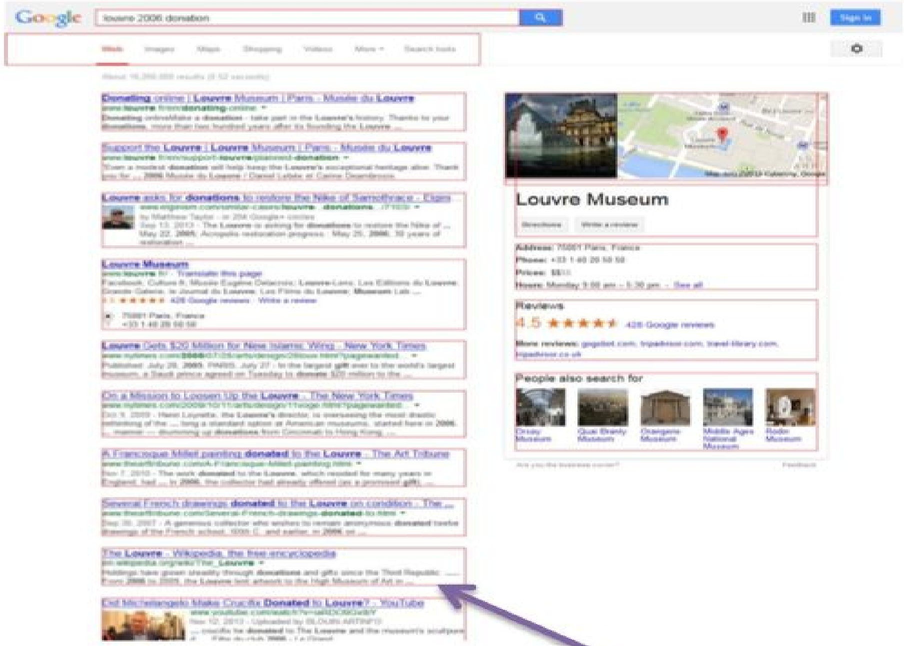
Variable number of possible videos per row (up to thousands)

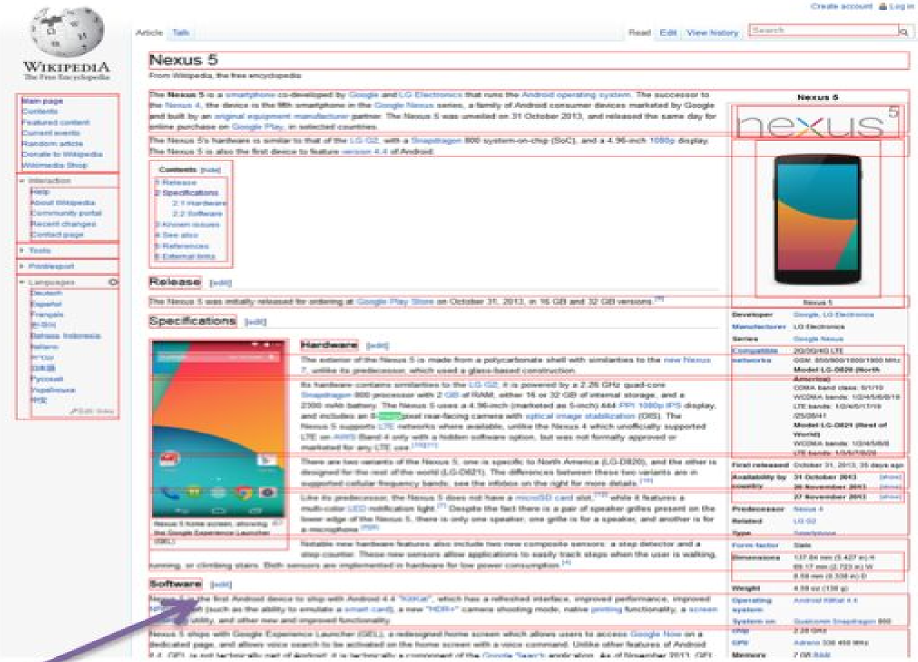
1 personalized page

10-40 rows

per device

Page Composition





From “Modeling User Attention and
Interaction on the Web” 2014 - PhD Thesis by Dmitry Lagun (Emory U.)

User Attention Modeling

Web Search (Google)

Social Network (Twitter)

News (CNN)

Shopping (Amazon)

From “Modeling User Attention and
Interaction on the Web” 2014 - PhD Thesis by Dmitry Lagun (Emory U.)

Page Composition

Accurate	vs.	Diverse
Discovery	vs.	Continuation
Depth	vs.	Coverage
Freshness	vs.	Stability
Recommendations	vs.	Tasks

- To put things together we need to combine different elements
  - Navigational/Attention Model
  - Personalized Relevance Model
  - Diversity Model

Beyond Ranking: Optimizing Whole-Page Presentation

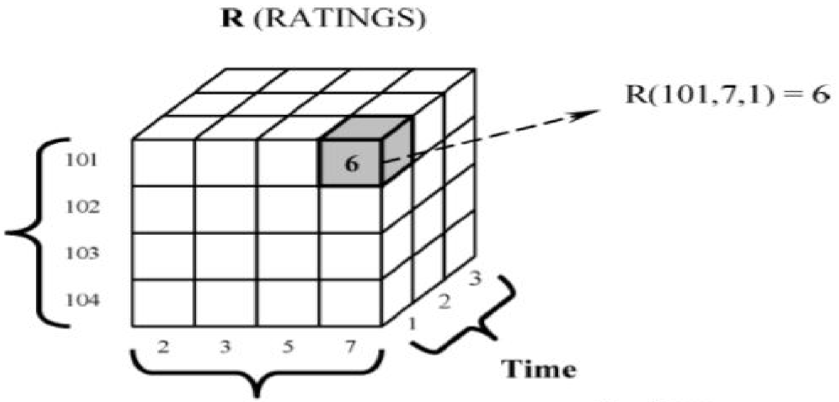
Yue Wang[1*], Dawei Yin[2], Luo Jie[3†], Pengyuan Wang[2], Makoto Yamada[2,4],
Yi Chang[2], Qiaozhu Mei[1,5]
[1]Department of EECS, University of Michigan, Ann Arbor, MI, USA
[2]Yahoo Labs, 701 First Avenue, Sunnyvale, CA, USA
[3]Snapchat, Inc., 64 Market St, Venice, CA, USA
[4]Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji, Kyoto, Japan

Fair and Balanced: Learning to Present News Stories

Amr Ahmed[*1], Choon Hui Teo[*1], S.V. N. Vishwanathan[2], Alex Smola[1]
[*] Co-first authors.
[1]Yahoo! Research, Santa Clara, CA 95053, USA
[2]Purdue University, West Lafayette, IN 47907, USA
{amahmed,choonhui,smola}@yahoo-inc.com, vishy@stat.purdue.edu

3.6 Beyond user/rating

N-dimensional model



User

Id	Name	Age
101	John	25
102	Bob	18
103	Alice	27
104	Mary	21

Item

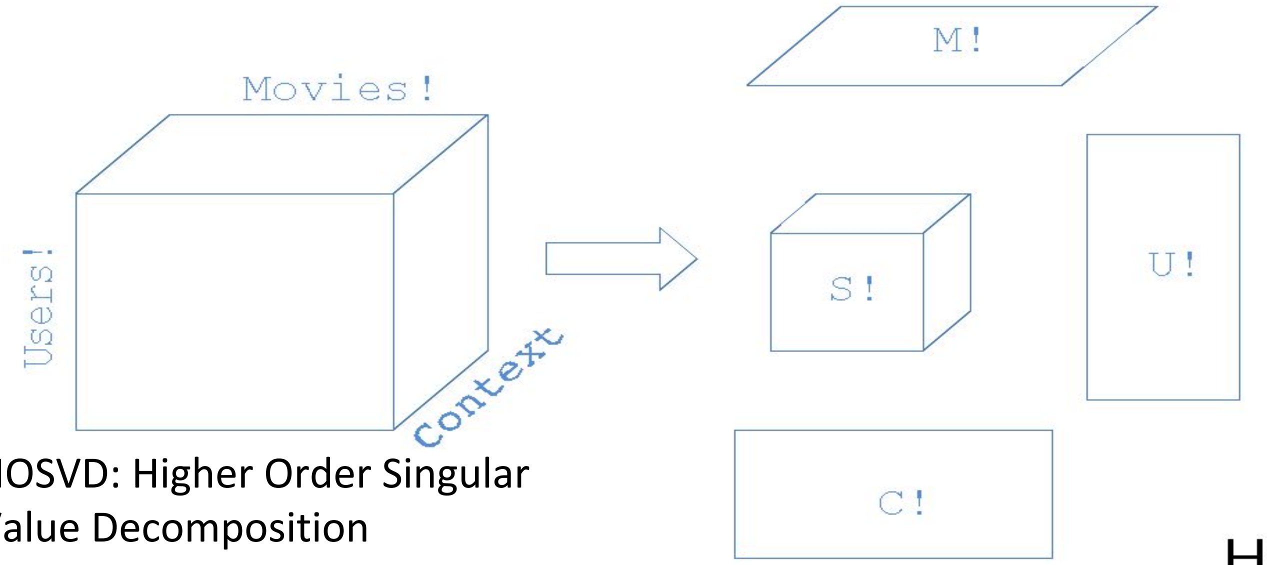
Id	Name	Cost
2	AB17	250.00
3	AB23	299.95
5	XY70	150.00
7	ZZ55	115.50

Time

Id	Name
1	Weekday
2	Weekend
3	Holiday

[Adomavicius et al., 2005]

Tensor Factorization



HOSVD: Higher Order Singular Value Decomposition

$U \in \mathbb{R}^{n\times d_U}$, $M \in \mathbb{R}^{m\times d_M}$ and $C \in \mathbb{R}^{c\times d_C}$

$S \in \mathbb{R}^{d_U\times d_M\times d_C}$

$$R[U,M,C,S] := L(F,Y) + \Omega[U,M,C] + \Omega[S]$$

HOSVD Model

Tensor Factorization

$$R[U, M, C, S] := L(F, Y) + \Omega[U, M, C] + \Omega[S]$$

Where:

$$\Omega[F] = \lambda_M \|M\|_F^2 + \lambda_U \|U\|_F^2 + \lambda_C \|C\|_F^2 \qquad \Omega[S] := \lambda_S \|S\|_F^2$$

- We can use a simple squared error loss function:

$$l(f, y) = \frac{1}{2}(f - y)^2$$

- Or the absolute error loss

$$l(f, y) = |f - y|$$

- The loss function over all users becomes

$$L(F, Y) = \sum_i^n \sum_j^m l(f_{ij}, y_{ij})$$

Factorization Machines

- Generalization of regularized matrix (and tensor) factorization approaches combined with linear (or logistic) regression
- Problem: Each new adaptation of matrix or tensor factorization requires deriving new learning algorithms
  - Hard to adapt to new domains and add data sources
  - Hard to advance the learning algorithms across approaches
  - Hard to incorporate non-categorical variables

Factorization Machines

- Approach: Treat input as a real-valued feature vector
  - Model both linear and pair-wise interaction of *k* features (i.e. polynomial regression)
  - Traditional machine learning will overfit
  - Factor pairwise interactions between features
  - Reduced dimensionality of interactions promote generalization
  - Different matrix factorizations become different feature representations
  - Tensors: Additional higher-order interactions
- Combines "generality of machine learning/regression with quality of factorization models"

Factorization Machines

- Each feature gets a weight value and a factor vector
  - O(*dk*) parameters

$$b \in \mathbb{R}, \mathbf{w} \in \mathbb{R}^d, \mathbf{V} \in \mathbb{R}^{d \times k}$$

- Model equation:

$$f(\mathbf{x}) = b + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} x_i x_j \mathbf{v}_i^T \mathbf{v}_j$$

O(d^2)

$$= b + \sum_{i=1}^{d} w_i x_i + \frac{1}{2} \sum_{f=1}^{k} \left(\left(\sum_{i=1}^{d} x_i v_{i,f} \right)^2 - \sum_{i=1}^{d} x_i^2 v_{i,f}^2 \right)$$

O(*kd*)

Factorization Machines

- Two categorical variables (*u*, *i*) encoded as real values:

Feature vector $\mathbf{x}$

	User				Movie				
	A	B	C	...	TI	NH	SW	ST	...
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...

- FM becomes identical to MF with biases:

$$f(\mathbf{x}) = b + w_u + w_i + \mathbf{v}_u^T \mathbf{v}_i$$

From Rendle (2012) KDD Tutorial

Factorization Machines

- Makes it easy to add a time signal

Feature vector $\mathbf{x}$

	User				Movie					Time
	A	B	C	...	TI	NH	SW	ST	...	
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...	0.2
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...	0.6
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...	0.61
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...	0.3
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...	0.5
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...	0.1
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...	0.8

- Equivalent equation:

$$f(\mathbf{x}) = b + w_u + w_i + x_t w_t + \mathbf{v}_u^T \mathbf{v}_i + x_t \mathbf{v}_u^T \mathbf{v}_t + x_t \mathbf{v}_i^T \mathbf{v}_t$$

From Rendle (2012) KDD Tutorial

Factorization Machines (Rendle, 2010)

- L2 regularized
  - Regression: Optimize RMSE
  - Classification: Optimize logistic log-likelihood
  - Ranking: Optimize scores
- Can be trained using:
  - SGD
  - Adaptive SGD
  - ALS
  - MCMC

Gradient:

$$\frac{\partial}{\partial\theta} f(\mathbf{x}) = \begin{cases} 1 & \text{if } \theta \text{ is } b \\ x_i & \text{if } \theta \text{ is } w_i \\ x_i \sum_{j=1}^{d} v_{j,f} x_j - v_{i,f} x_i^2 & \text{if } \theta \text{ is } v_{i,f} \end{cases}$$

Least squares SGD:

$$\theta' = \theta - \eta \left((f(\mathbf{x}) - y) \frac{\partial}{\partial\theta} f(\mathbf{x}) + \lambda_\theta \theta \right)$$

Factorization Machines (Rendle, 2010)

- Learning parameters:
  - Number of factors
  - Iterations
  - Initialization scale
  - Regularization (SGD, ALS) – Multiple
  - Step size (SGD, A-SGD)
  - MCMC removes the need to set those hyperparameters

3.7 Deep Learning

(See Balázs Hidasi's slides)

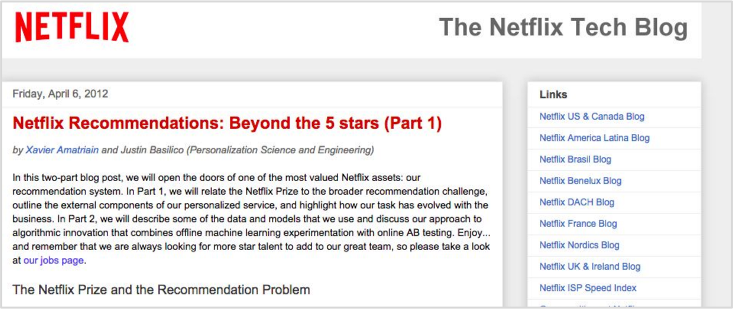
4. Lessons Learned

1. Implicit signals beat explicit ones (almost always)

Quora

Implicit vs. Explicit

- Many have acknowledged that implicit feedback is more useful
- Is implicit feedback really always more useful?
- If so, why?


NETFLIX The Netflix Tech Blog

Friday, April 6, 2012

Netflix Recommendations: Beyond the 5 stars (Part 1)

by Xavier Amatriain and Justin Basilico (Personalization Science and Engineering)

In this two-part blog post, we will open the doors of one of the most valued Netflix assets: our recommendation system. In Part 1, we will relate the Netflix Prize to the broader recommendation challenge, outline the external components of our personalized service, and highlight how our task has evolved with the business. In Part 2, we will describe some of the data and models that we use and discuss our approach to algorithmic innovation that combines offline machine learning experimentation with online AB testing. Enjoy... and remember that we are always looking for more star talent to add to our great team, so please take a look at our jobs page.

The Netflix Prize and the Recommendation Problem

Links

- Netflix US & Canada Blog
- Netflix America Latina Blog
- Netflix Brasil Blog
- Netflix Benelux Blog
- Netflix DACH Blog
- Netflix France Blog
- Netflix Nordics Blog
- Netflix UK & Ireland Blog
- Netflix ISP Speed Index


YouTube Official Blog

What's happening on YouTube

Five Stars Dominate Ratings

Posted: Tuesday, September 22, 2009

This graph fascinated us and so we wanted to share. It shows the number of videos that receive specific star

YouTube Comes To A 5-Star Realization: Its Ratings Are Useless

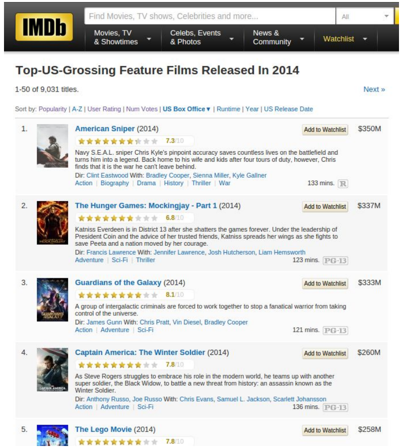
Posted Sep 22, 2009 by MG Siegler (@parislemon)

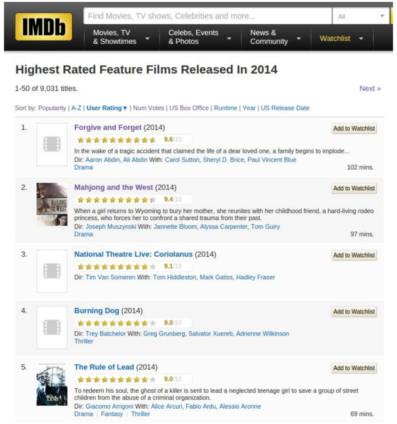
56 SHARES

Seems like when it comes to ratings it's pretty much all or nothing. Great videos prompt action; anything less prompts indifference. Thus, the ratings system is primarily being used as a seal of approval, not as an editorial indicator of what the community thinks about a video. Rating a video joins favoriting and sharing as a way to tell the world that *this* is something you love.

Implicit vs. Explicit

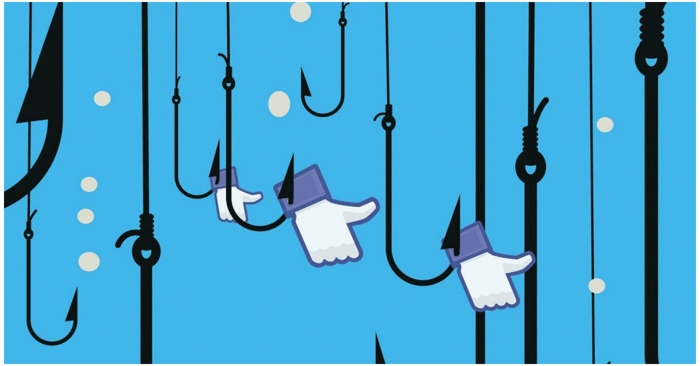
- Implicit data is (usually):
  - More dense, and available for all users
  - Better representative of user behavior vs. user reflection
  - More related to final objective function
  - Better correlated with AB test results
- E.g. Rating vs watching

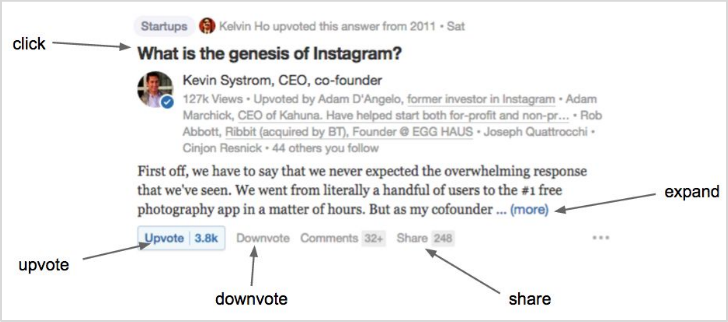




Implicit vs. Explicit

- However
  - It is not always the case that direct implicit feedback correlates well with long-term retention
  - E.g. clickbait
- Solution:
  - Combine different forms of implicit + explicit to better represent long-term goal



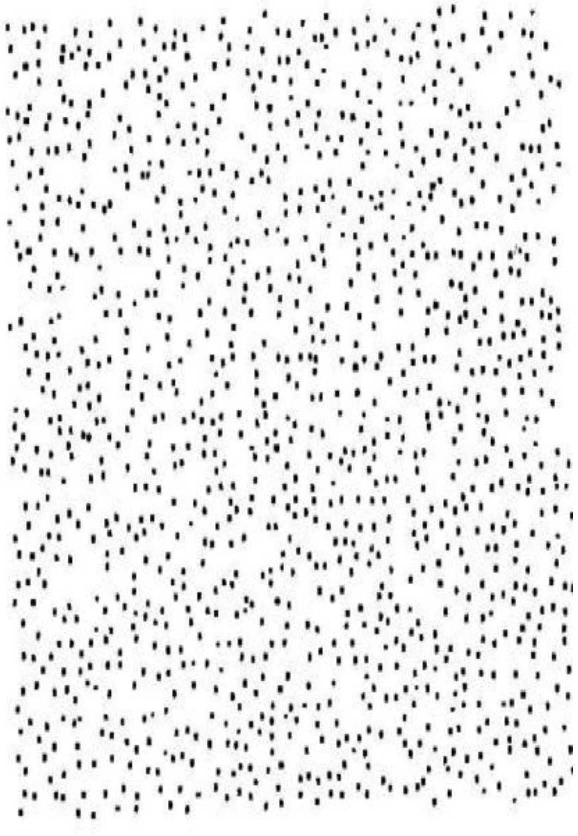


2. be thoughtful about your Training Data

Quora

Defining training/testing data

- Training a simple binary classifier for good/bad answer
  - Defining positive and negative labels -> Non-trivial task
  - *Is this a positive or a negative?*
    - funny uninformative answer with many upvotes
    - short uninformative answer by a well-known expert in the field
    - very long informative answer that nobody reads/upvotes
    - informative answer with grammar/spelling mistakes
    - ...



3. Your Model will learn what you teach it to learn

Quora

Training a model

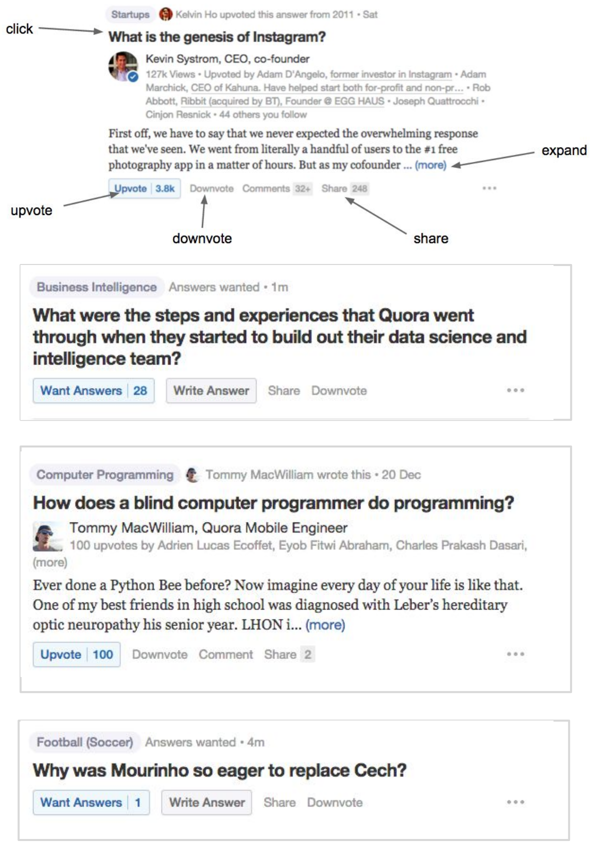
- Model will learn according to:
    - Training data (e.g. implicit and explicit)
    - Target function (e.g. probability of user reading an answer)
    - Metric (e.g. precision vs. recall)
- Example 1 (made up):
    - *Optimize probability of a user going to the cinema to watch a movie and rate it "highly" by using purchase history and previous ratings. Use NDCG of the ranking as final metric using only movies rated 4 or higher as positives.*

Example 2 - Quora's feed

- Training data = implicit + explicit
- Target function: Value of showing a story to a user ~ weighted sum of actions:

  $$v = \Sigma_a v_a 1\{y_a = 1\}$$

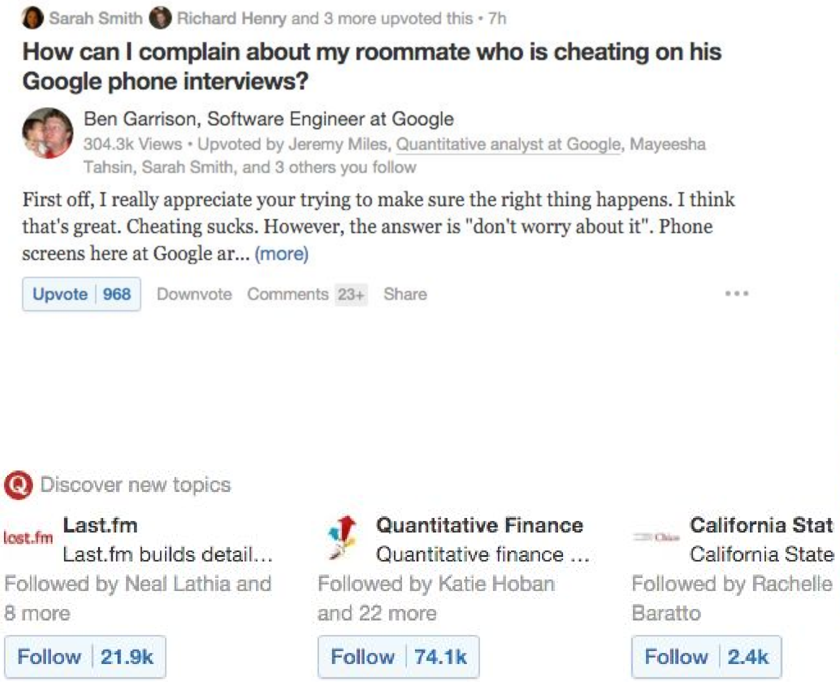
  - predict probabilities for each action, then compute expected value: $v\_pred = E[V | x] = \Sigma_a v_a p(a | x)$
- Metric: any ranking metric

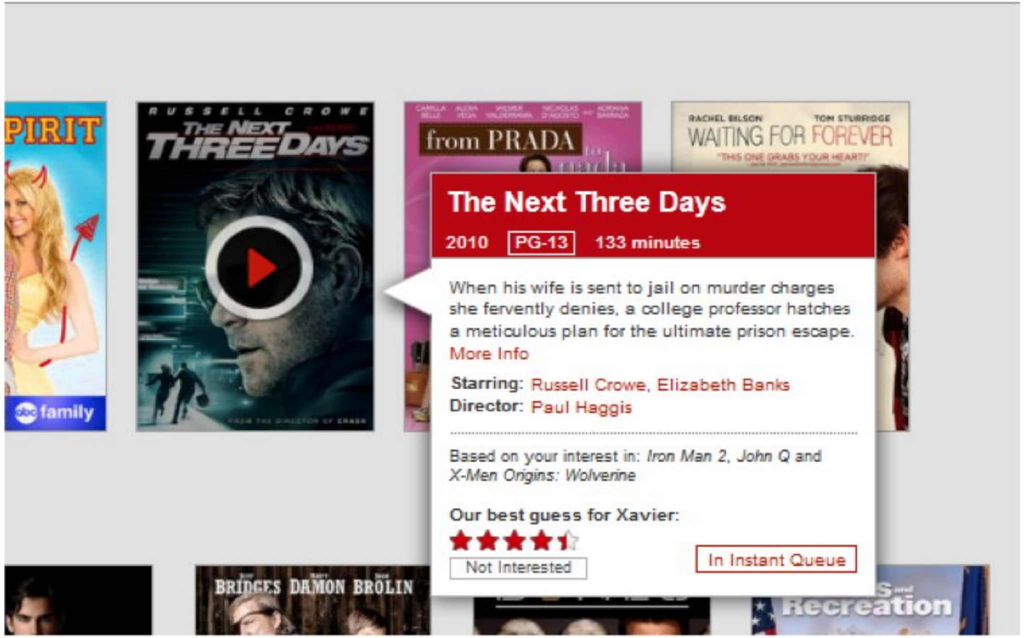


4. Explanations might matter more than the prediction

Quora

Explanation/Support for Recommendations



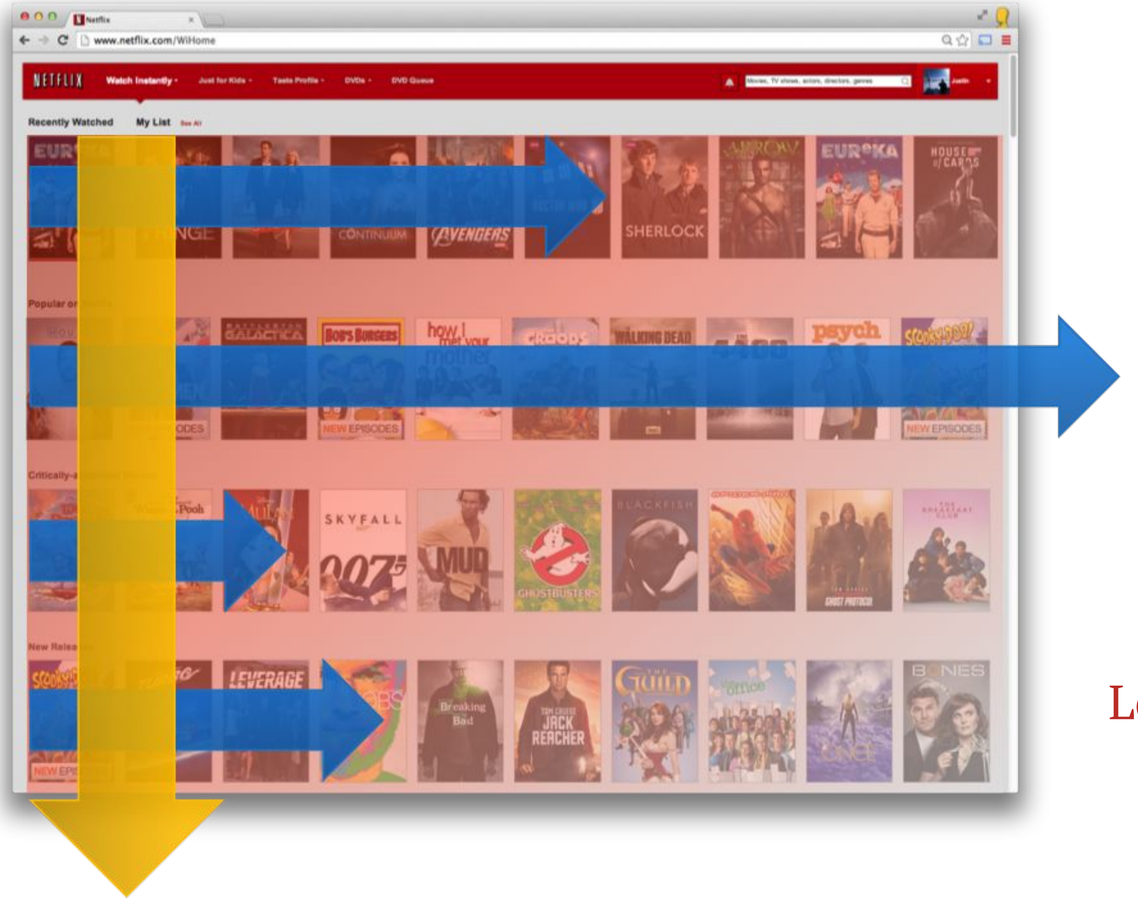


5. Learn to deal with Presentation Bias

Quora

2D Navigational modeling

More likely
to see



Less likely

The curse of presentation bias

- User can only click on what you decide to show
  - But, what you decide to show is the result of what your model predicted is good
- Simply treating things you show as negatives is not likely to work
- Better options
  - Correcting for the probability a user will click on a position -> Attention models
  - Explore/exploit approaches such as MAB

Collaborative Competitive Filtering: Learning Recommender Using Context of User Choice

Shuang Hong Yang
Georgia Tech
shy@gatech.edu

Bo Long
Yahoo! Labs
bolong@yahoo-inc.com

Alex Smola
Yahoo! Research
smola@yahoo-inc.com

Hongyuan Zha
Georgia Tech
zha@cc.gatech.edu

Zhaohui Zheng
Yahoo! Labs Beijing
zhaohui@yahoo-inc.com

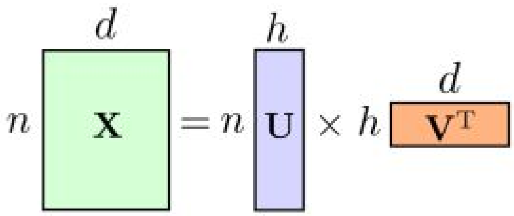
6. If You Have to Pick one single approach, Matrix factorization is your best bet

Quora

Matrix Factorization

Quora

- MF can be interpreted as
  - Unsupervised:
    - Dimensionality Reduction a la PCA
    - Clustering (e.g. NMF)
  - Supervised:
    - Labeled targets ~ regression
- Very useful variations of MF
  - BPR, ALS, SVD++
  - Tensor Factorization, Factorization Machines
- However...



7. Everything is an ensemble

Quora

Ensembles

- Netflix Prize was won by an ensemble
  - Initially Bellkor was using GDBTs
  - BigChaos introduced ANN-based ensemble
- Most practical applications of ML run an ensemble
  - Why wouldn't you?
  - At least as good as the best of your methods
  - Can combine different approaches (e.g. CF and content-based)
  - Can use different models at the ensemble layer: LR, GDBTs, RFs, ANNs...

The BellKor Solution to the Netflix Grand Prize

Yehuda Koren

August 2009

The BigChaos Solution to the Netflix Grand Prize

Andreas Töscher and Michael Jahrer

commendo research & consulting
Neuer Weg 23, A-8580 Köflach, Austria
{andreas.toescher,michael.jahrer}@commendo.at

Robert M. Bell*

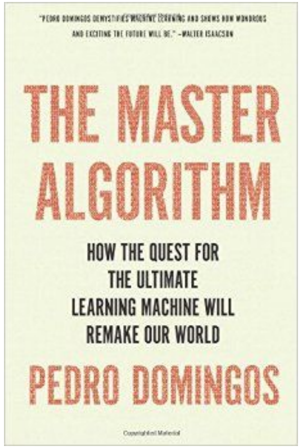
AT&T Labs - Research
Florham Park, NJ

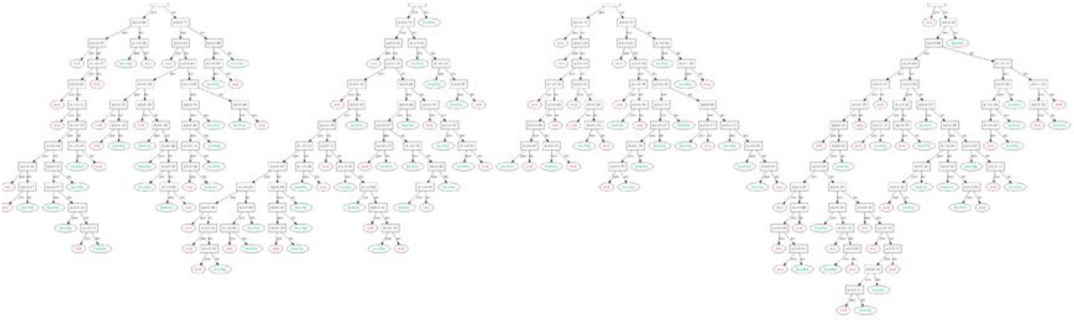
September 5, 2009

Ensembles & Feature Engineering

Quora

- Ensembles are the way to turn any model into a feature!
- E.g. Don’t know if the way to go is to use Factorization Machines, Tensor Factorization, or RNNs?
    - Treat each model as a “feature”
    - Feed them into an ensemble





8. Building Recommender Systems is also about Feature Engineering

Quora

Feature Engineering

Need for feature engineering

In many cases an understanding of the domain will lead to optimal results.

Feature Engineering Example - Quora Answer Ranking

Quora

What is a good Quora answer?

- truthful
- reusable
- provides explanation
- well formatted
- ...

What music do data scientists usually listen to while working?

Paula Griffin, data scientist and biostatistics PhD ... (more)
13 upvotes by William Chen, Alexandr Wang (王誉舜), Sheila Christine Lee, (more)

I was figuring that this question was just fishing for someone to answer that Big Data is their favorite band. Unfortunately, the question log indicates this was asked about 6 months before their EP came out, so there goes that theory.

This is going to be a pretty odd list, but here's the list, in order of decreasing social acceptability:

- Electropop -- Banks and CHVRCHES are my favorites at the moment.
- Miscellaneous alt-rock -- this category basically includes anything I found out about from listening to Sirius XM in the car.
- Nerd rock -- What kind of geek would I be if Jonathan Coulton wasn't on this list?

Shankar Iyer, data scientist at Quora
10 upvotes by William Chen, Sheila Christine Lee, Don van der Drift, (more)

Based on the Pandora stations that I've been listening to, my recent work-time listening consists of:

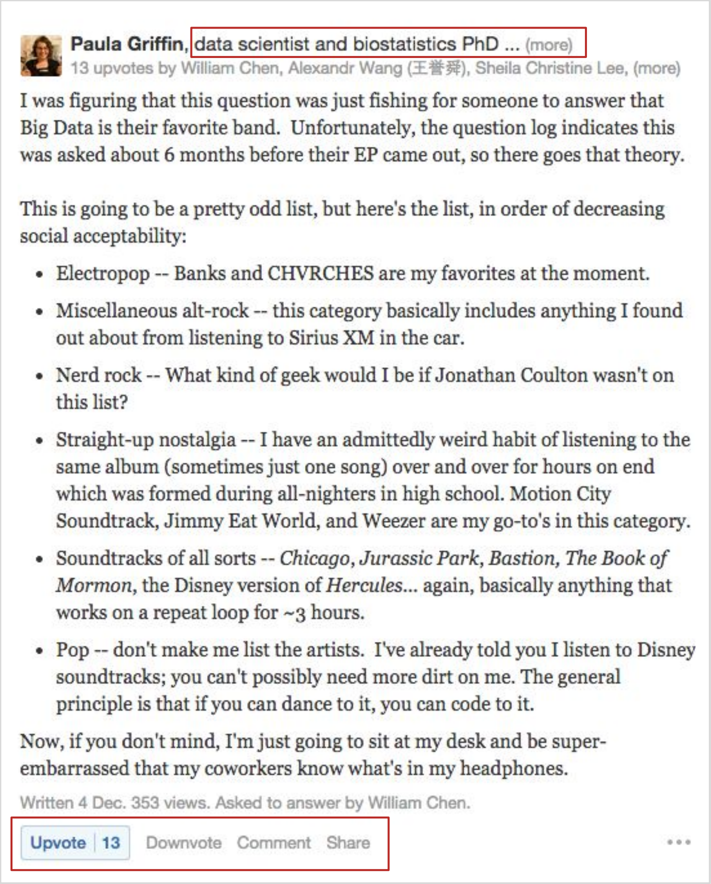
1. **Acoustic folk music:** John Fahey, Leo Kottke, Six Organs of Admittance, etc.
2. **Post-Rock / Ambient Music:** Sigur Rós, Gregor Samsa, the Japanese Mono, Eluvium, El Ten Eleven, etc.
3. **Hindustani:** mostly Vishwa Mohan Bhatt
4. **Carnatic:** recently Rajeswari Pariti
5. **Classical Guitar:** recently Paul Galbraith, Konrad Ragossnig, etc.

Feature Engineering Example - Quora Answer Ranking

Quora

How are those dimensions translated into features?

- Features that relate to the answer quality itself
- Interaction features (upvotes/downvotes, clicks, comments…)
- User features (e.g. expertise in topic)



Paula Griffin, data scientist and biostatistics PhD ... (more)
13 upvotes by William Chen, Alexandr Wang (王誉舜), Sheila Christine Lee, (more)

I was figuring that this question was just fishing for someone to answer that Big Data is their favorite band. Unfortunately, the question log indicates this was asked about 6 months before their EP came out, so there goes that theory.

This is going to be a pretty odd list, but here's the list, in order of decreasing social acceptability:

- Electropop -- Banks and CHVRCHES are my favorites at the moment.
- Miscellaneous alt-rock -- this category basically includes anything I found out about from listening to Sirius XM in the car.
- Nerd rock -- What kind of geek would I be if Jonathan Coulton wasn't on this list?
- Straight-up nostalgia -- I have an admittedly weird habit of listening to the same album (sometimes just one song) over and over for hours on end which was formed during all-nighters in high school. Motion City Soundtrack, Jimmy Eat World, and Weezer are my go-to's in this category.
- Soundtracks of all sorts -- *Chicago*, *Jurassic Park*, *Bastion*, *The Book of Mormon*, the Disney version of *Hercules*... again, basically anything that works on a repeat loop for ~3 hours.
- Pop -- don't make me list the artists. I've already told you I listen to Disney soundtracks; you can't possibly need more dirt on me. The general principle is that if you can dance to it, you can code to it.

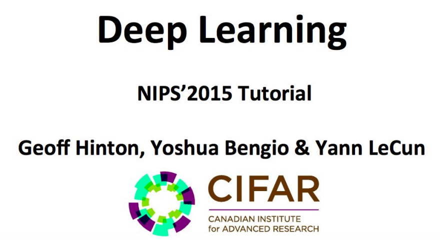
Now, if you don't mind, I'm just going to sit at my desk and be super-embarrassed that my coworkers know what's in my headphones.

Written 4 Dec. 353 views. Asked to answer by William Chen.

Upvote | 13 Downvote Comment Share

Feature Engineering

- Properties of a well-behaved ML feature:
  - Reusable
  - Transformable
  - Interpretable
  - Reliable



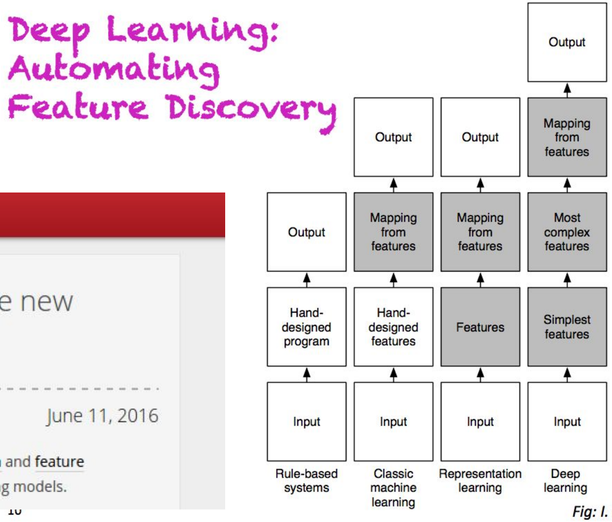
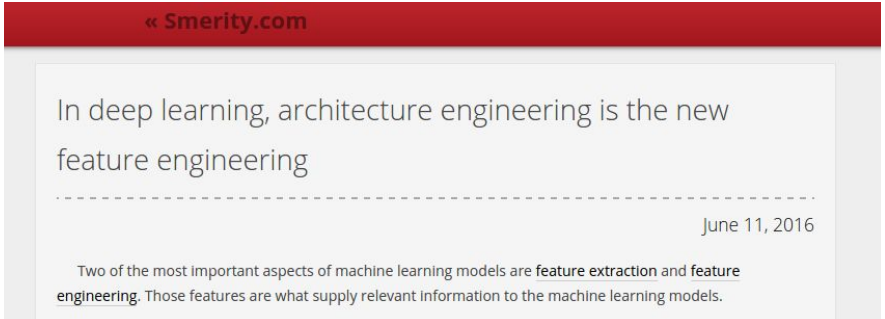


Fig: I. Goodfellow



« Smerity.com

In deep learning, architecture engineering is the new feature engineering

June 11, 2016

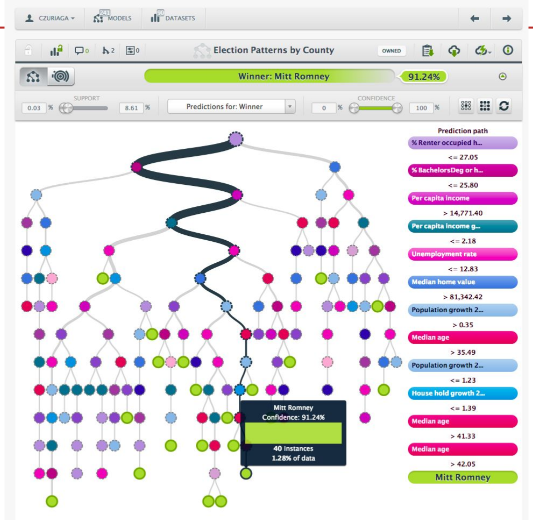
Two of the most important aspects of machine learning models are feature extraction and feature engineering. Those features are what supply relevant information to the machine learning models.

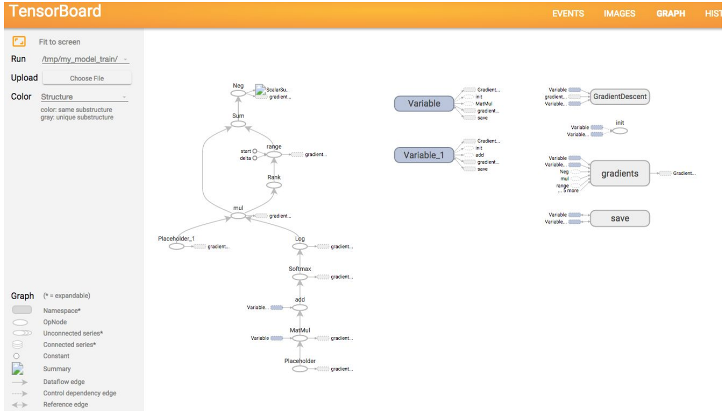
9. Why you should care about answering questions (about your recsys)

Quora

Model debuggability

- Value of a model = value it brings to the product
- Product owners/stakeholders have expectations on the product
- It is important to answer questions to why did something fail
- Model debuggability is so important it can determine:
  - Particular model to use
  - Features to rely on
  - Implementation of tools

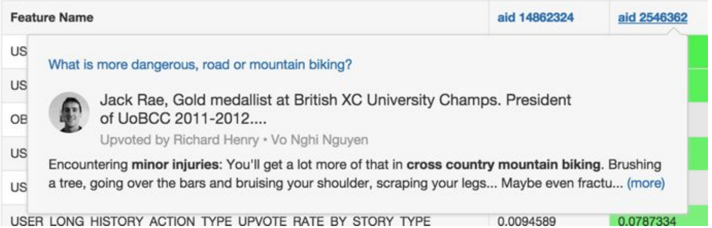


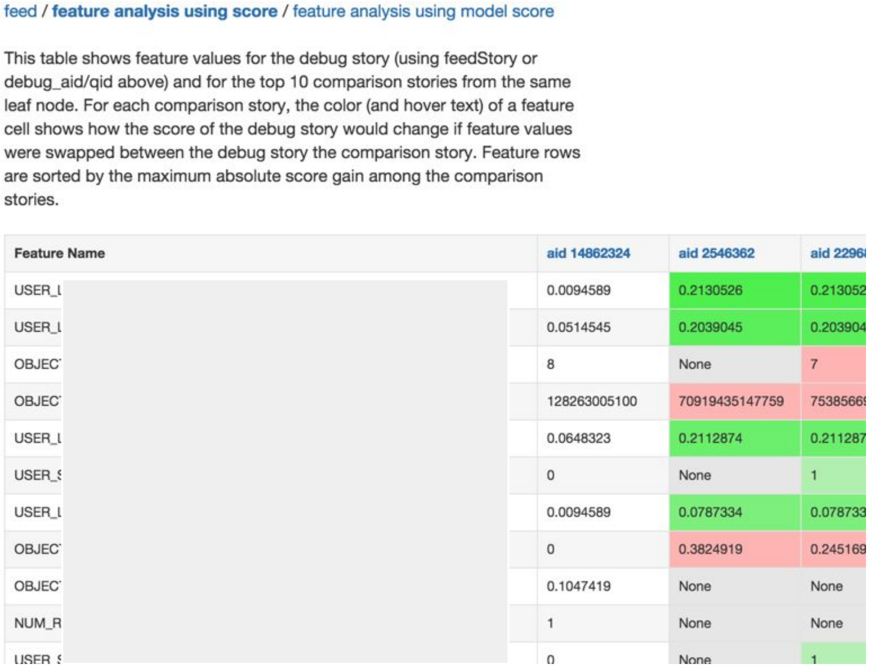


Model debuggability

Quora

- E.g. Why am I seeing or not seeing this on my homepage feed?

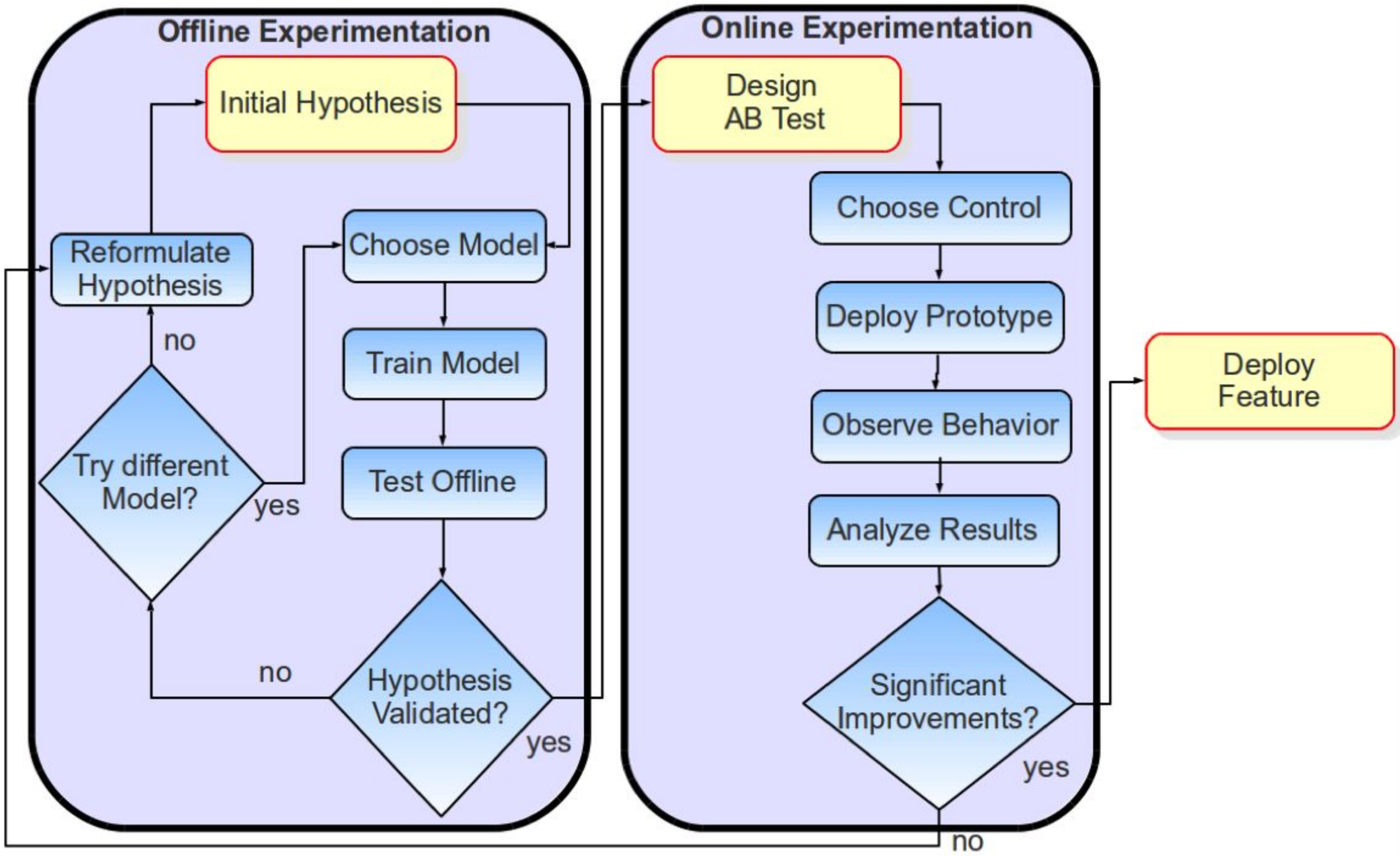




10. Data and Models are great. You know what's even better?

The right evaluation approach!

Offline/Online testing process



11. You don't need to distribute your Recsys

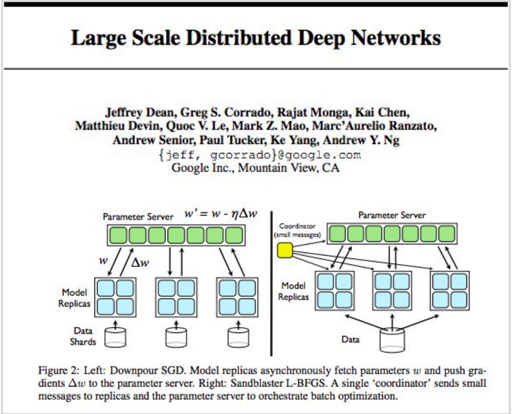
Quora

Distributing Recommender Systems

Quora

- Most of what people do in practice can fit into a multi-core machine
    - As long as you use:
        - Smart data sampling
        - Offline schemes
        - Efficient parallel code
- (... but not Deep ANNs)

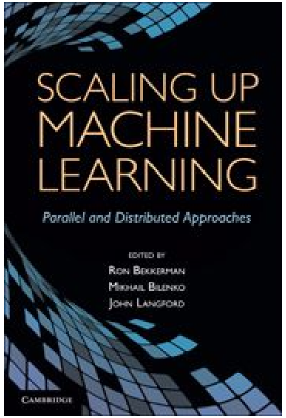
- Do you care about costs? How about latencies or system complexity/debuggability?



Large Scale Distributed Deep Networks

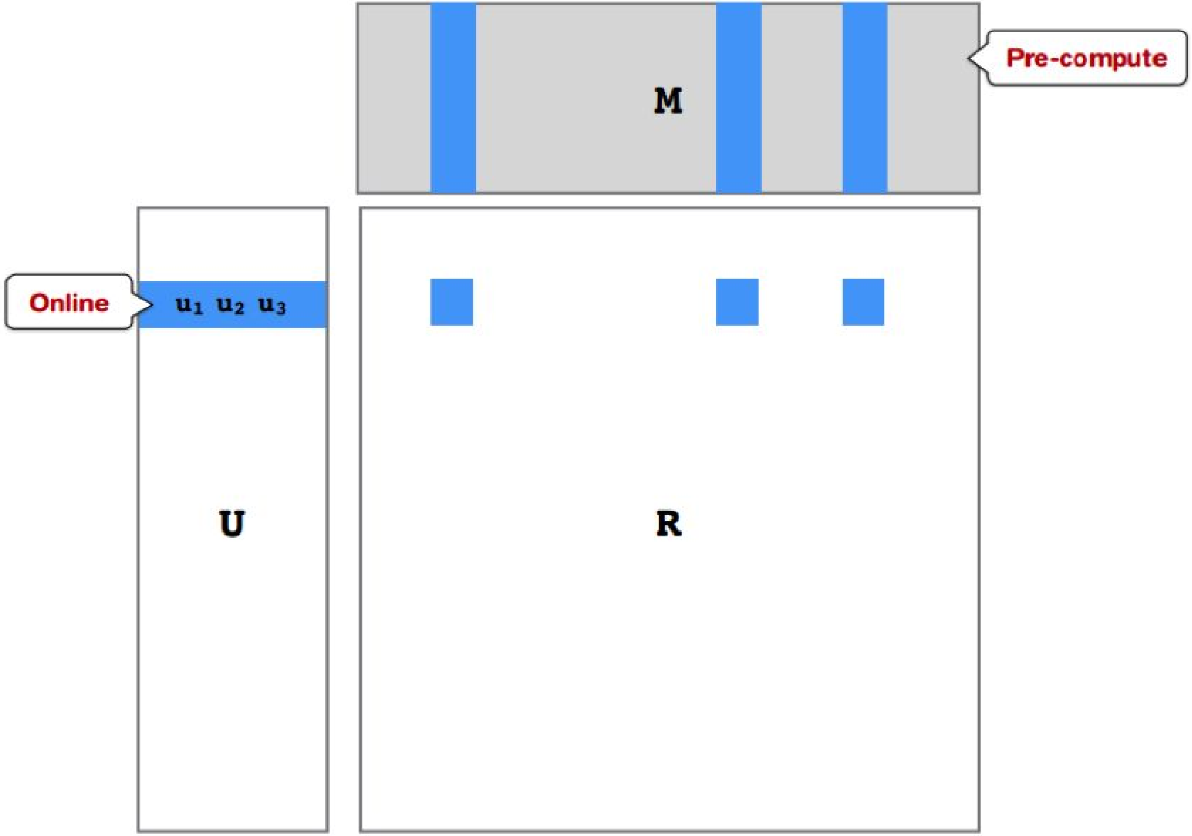
Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen,
Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato,
Andrew Senior, Paul Tucker, Ke Yang, Andrew Y. Ng
{jeff, gcorrado}@google.com
Google Inc., Mountain View, CA

Figure 2: Left: Downpour SGD. Model replicas asynchronously fetch parameters w and push gradients Δw to the parameter server. Right: Sandblaster L-BFGS. A single 'coordinator' sends small messages to replicas and the parameter server to orchestrate batch optimization.



Matrix Factorization Example

Quora

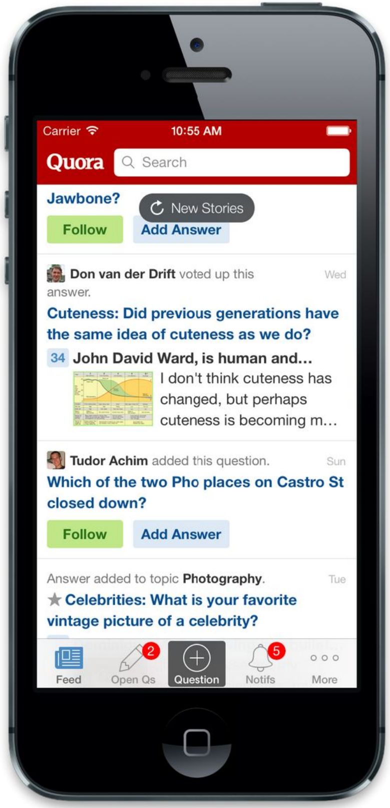


12. The UI is the only communication channel with what matters the most:

<u>Users</u>

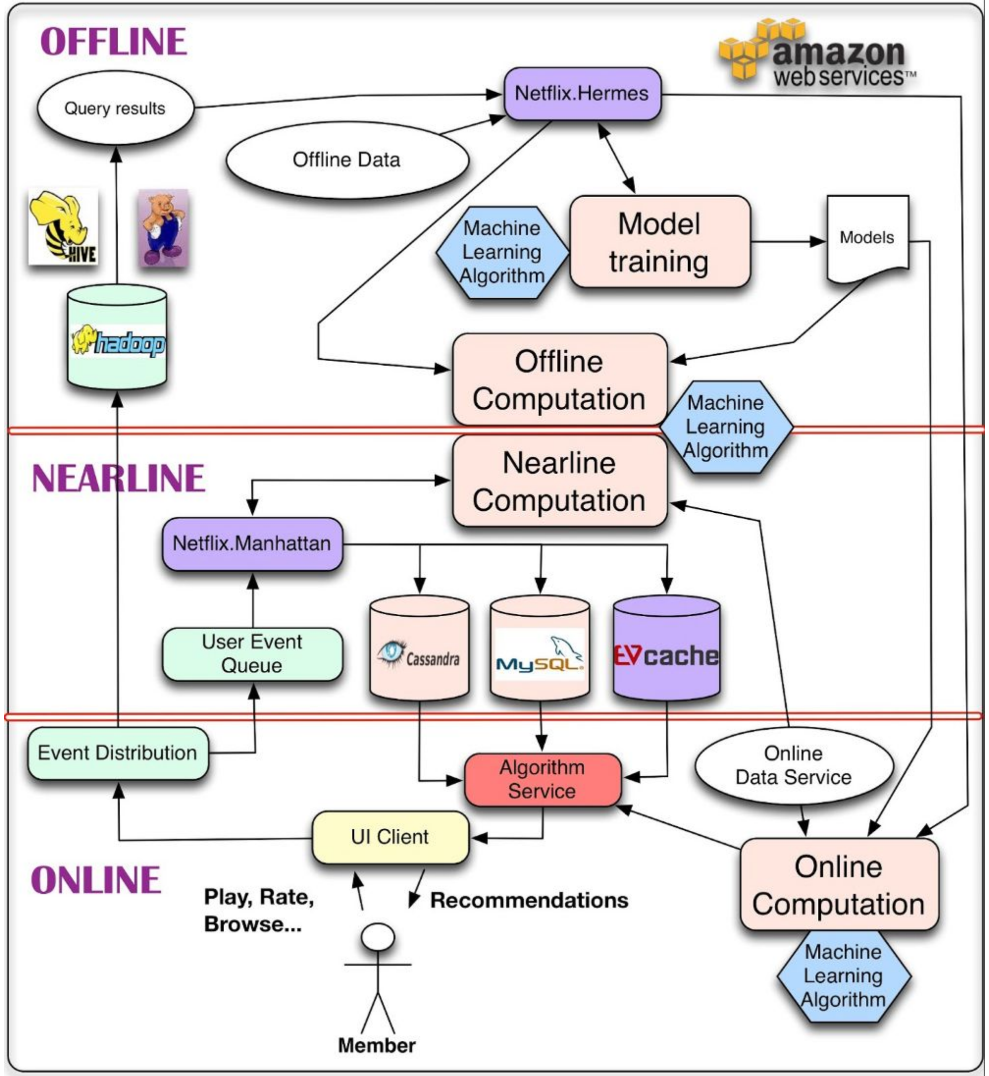
Quora

UI->Algorithm->UI

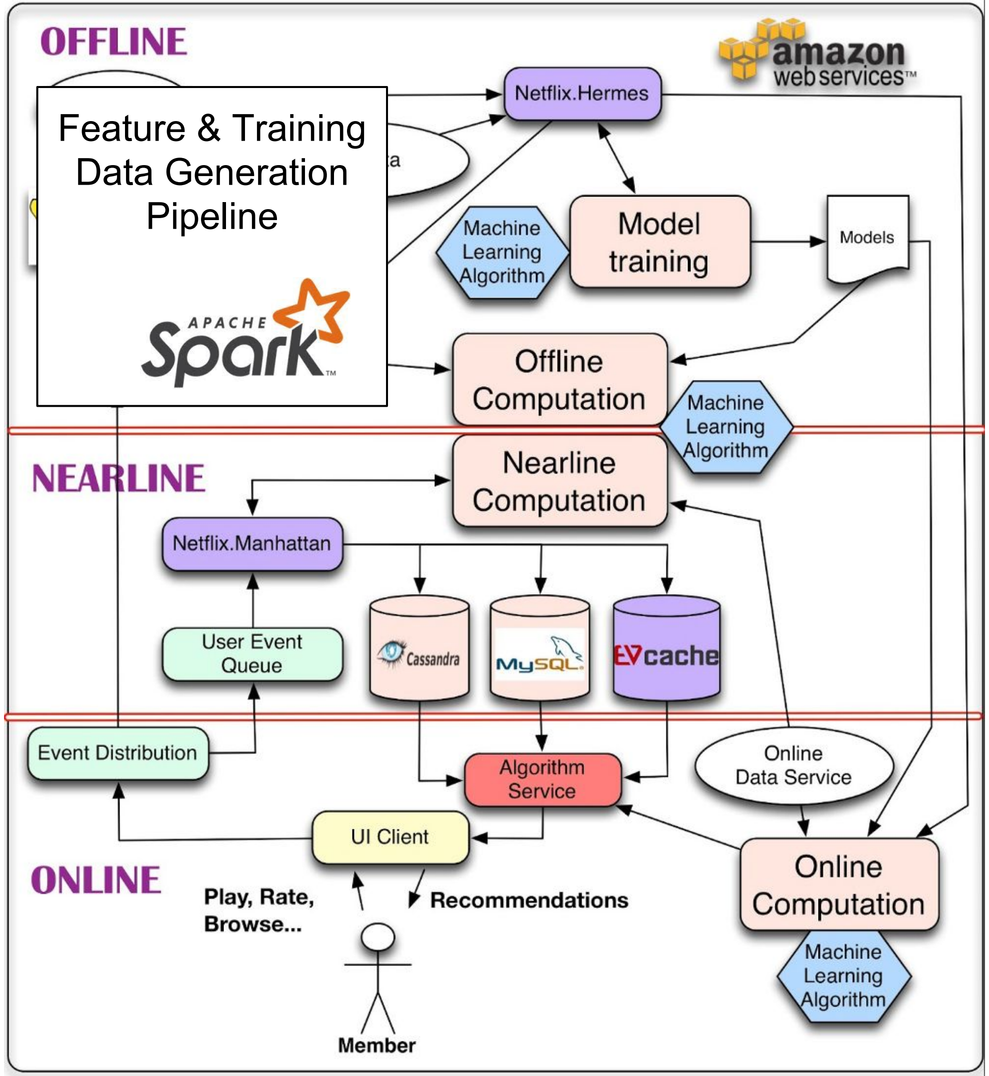


- The UI generates the user feedback that we will input into the algorithms
- The UI is also where the results of our algorithms will be shown
- A change in the UI might require a change in algorithms and viceversa

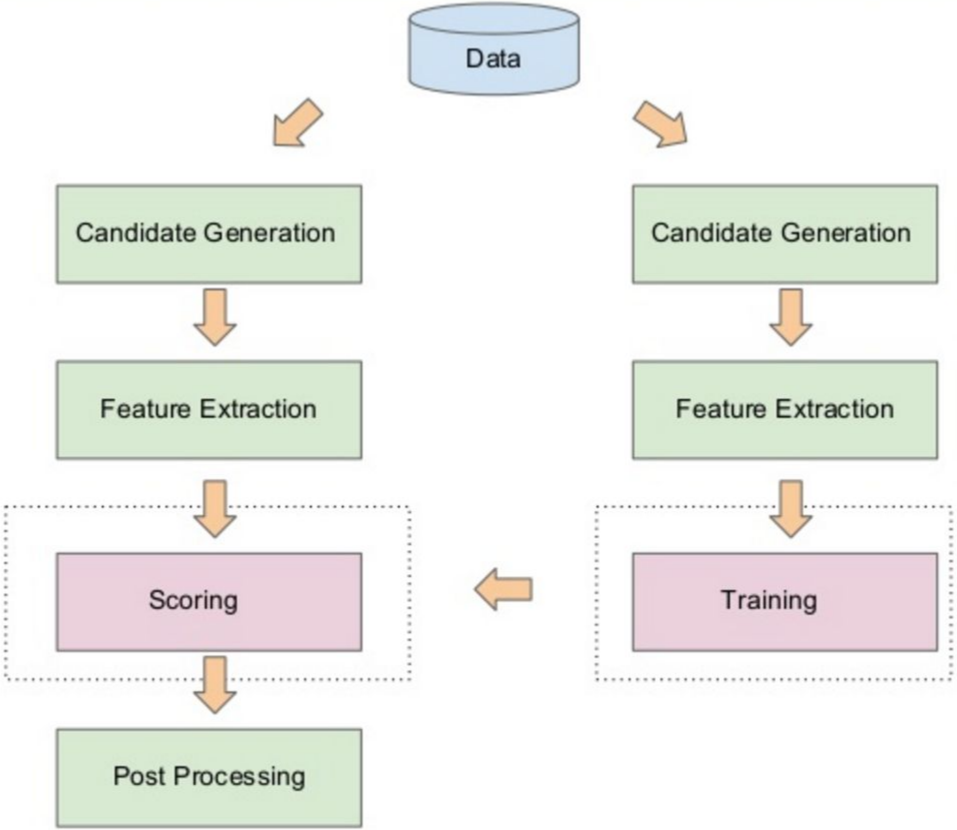
5. A Recsys Architectural Blueprint



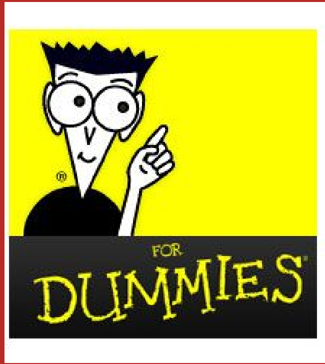
OFFLINE
Query results
Netflix.Hermes
Offline Data
Machine Learning Algorithm
Model training
Models
Offline Computation
Machine Learning Algorithm
NEARLINE
Nearline Computation
Netflix.Manhattan
User Event Queue
Cassandra
MySQL
EVcache
Event Distribution
Algorithm Service
Online Data Service
UI Client
ONLINE
Play, Rate, Browse...
Recommendations
Online Computation
Machine Learning Algorithm
Member


OFFLINE
amazon web services
Feature & Training Data Generation Pipeline
APACHE Spark
Netflix.Hermes
Machine Learning Algorithm
Model training
Models
Offline Computation
Machine Learning Algorithm
NEARLINE
Nearline Computation
Netflix.Manhattan
User Event Queue
Cassandra
MySQL
EVcache
Event Distribution
Algorithm Service
Online Data Service
UI Client
ONLINE
Play, Rate, Browse...
Recommendations
Online Computation
Machine Learning Algorithm
Member

- We want the same code/systems/tools to work for both experimentation & production.
- But we need to carefully “control” the production code to keep it be fast.
- So need to “control” offline experimentation systems too.



6. Building a state-of-the-art Recsys



6.1 Training, testing, and metrics

Training, testing, metrics

- As mentioned in the lessons, this is essential
- Choose implicit data and metrics that connect to your business goal
- Sample negatives smartly
- Select validation and test set carefully (e.g. avoid time traveling)

Training, testing, metrics

- For metrics, prefer ranking or ranking-related metrics

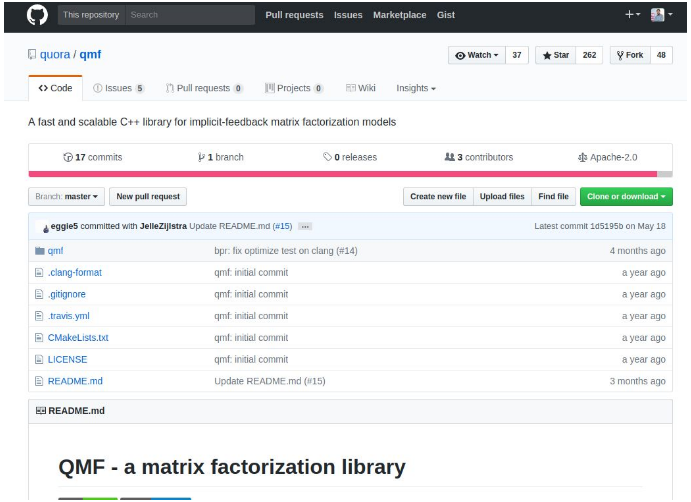
6.2 Implicit Matrix Factorization

Implicit Matrix Factorization

- Experience says, best single (simple) approach: implicit matrix factorization:
  - ALS. Alternating Least Squares (Hu et al. 2008)
  - BPR. Bayesian Personalized Ranking (Rendle et al. 2009)

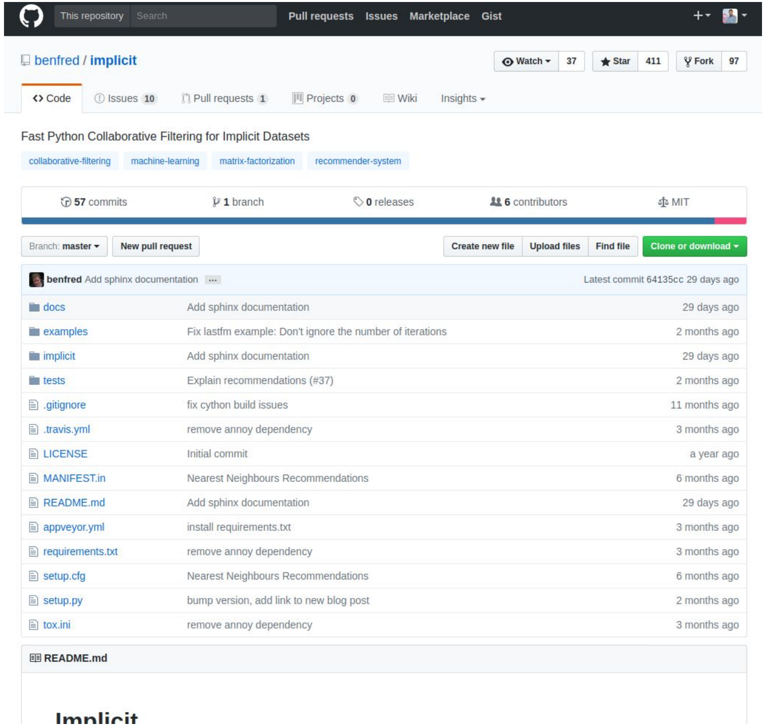
Recommended Implementations

- Quora's QMF
  - Efficient compiled C++ code
  - Supports many evaluation metrics



Recommended Implementations

- Implicit
  - Efficient
  - Python
  - Well-maintained



Others?

- Sorry to say, but I cannot recommend any others (no, not Mahout)

6.3 A/B Test

AB Test

- So, you have your first implementation
  - Have tuned hyperparameters to optimize offline metric
  - How do you know this is working?
- Run AB Test!
  - Make sure offline metric (somewhat) correlates to online effect

AB Test

- Ideally, you would run several AB tests with different offline metrics and data sampling strategies

6.4 Ensemble

Ensemble

- Now, it’s time to turn the model into a signal
- Brainstorm about some simple potential features that you could combine with implicit MF
  - E.g. user tenure, average rating for the item, price of the item…
- Add to MF through an ensemble

Ensemble

- What model to use at the ensemble layer?
  - Always favor most simple -> L2-regularized Logistic Regression
  - Eventually introduce models that can benefit from non-linear effects and many features -> Gradient Boosted Decision Trees
  - Explore Learning-to-rank models -> LambdaRank

6.5 Iterate, Feature Engineering

Iterate

- Experiment/add more features
- Experiment with more complex models
- Do both things in parallel
- Continue AB testing

7. Practical exercise

Exercise

- Train an ALS implicit matrix factorization recommender system
- Do basic feature engineering to add other features
- Add the mix to an XGBoost-based ensemble
- This is very close to what you could be using in real-life (minus scalability/performance issues)

8. Future Research Directions

Many interesting future directions

1. Indirect feedback
2. Value-awareness
3. Full-page optimization
4. Personalizing the how

Others

- Intent/session awareness
- Interactive recommendations
- Context awareness
- Deep learning for recommendations
- Conversational interfaces/bots for recommendations
- …

Indirect Feedback

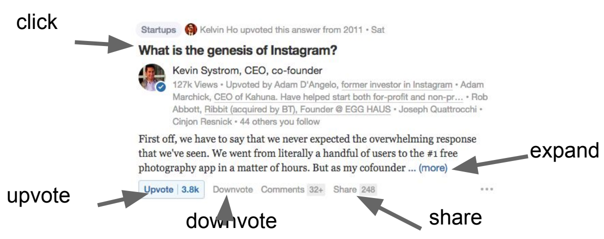
Challenges

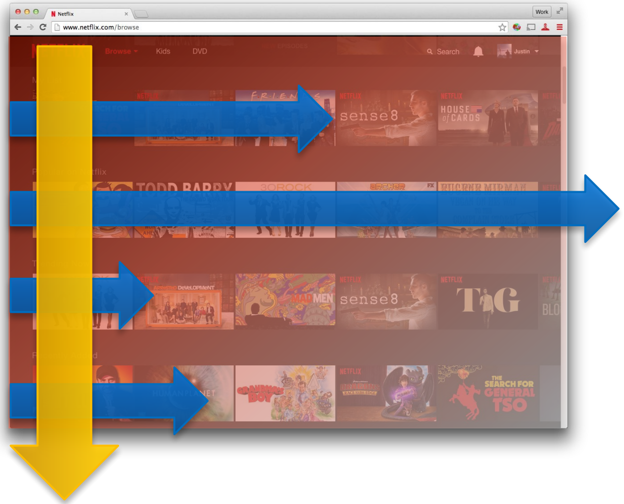
- User can only click on what you show
  - But, what you show is the result of what your model predicted is good
- No counterfactuals
- Implicit data has no real “negatives”

Potential solutions

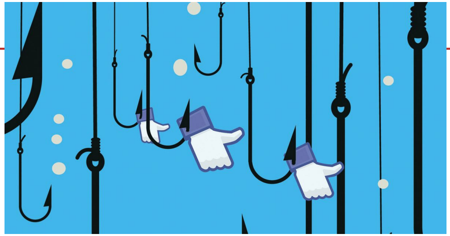
- Attention models
- Context is also indirect/implicit feedback
- Explore/exploit approaches and learning across time
- ...

Quora





Value-aware recommendations



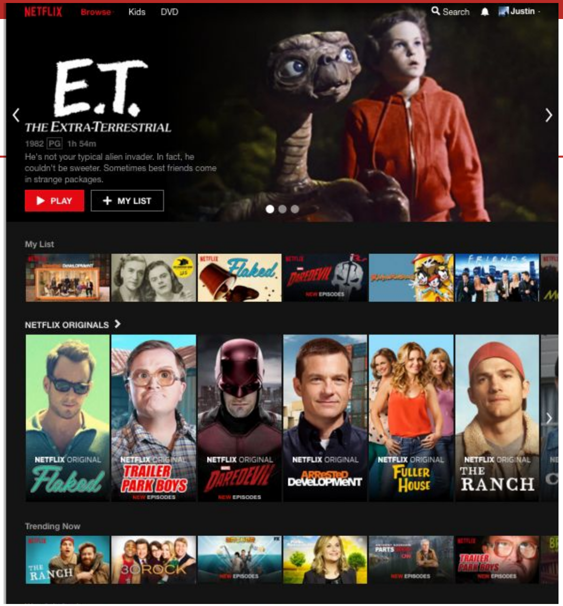
- Recsys optimize for probability of action
- Not all clicks/actions have the same "reward"
    - Different margin in ecommerce
    - Different "quality" of content
    - Long-term retention vs. short-term clicks (clickbait)
    - …
- In Quora, the value of showing a story to a user is approximated by weighted sum of actions:
    - $v = \sum_a v_a 1\{y_a = 1\}$
- Extreme application of value-aware recommendations: suggest items to **create** that have the highest value

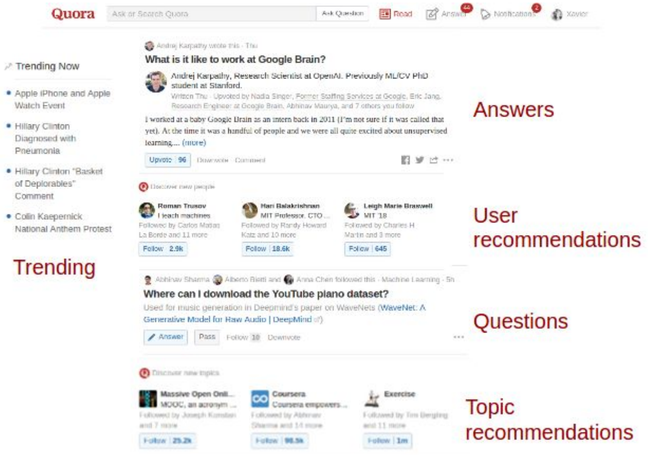
    Netflix: Which shows to produce or license

    Quora: Answers and questions that are not in the service

Full-page optimization

- Recommendations are rarely displayed in isolation
  - Rankings are combined with many other elements to make a page
- Want to optimize the **whole page**
- **Jointly** solving for set of items and their placement
- While incorporating
  - Diversity, freshness, exploration
  - Depth and coverage of the item set
  - Non-recommendation elements (navigation, editorial, etc.)
- Needs work hand-in-hand with the UX





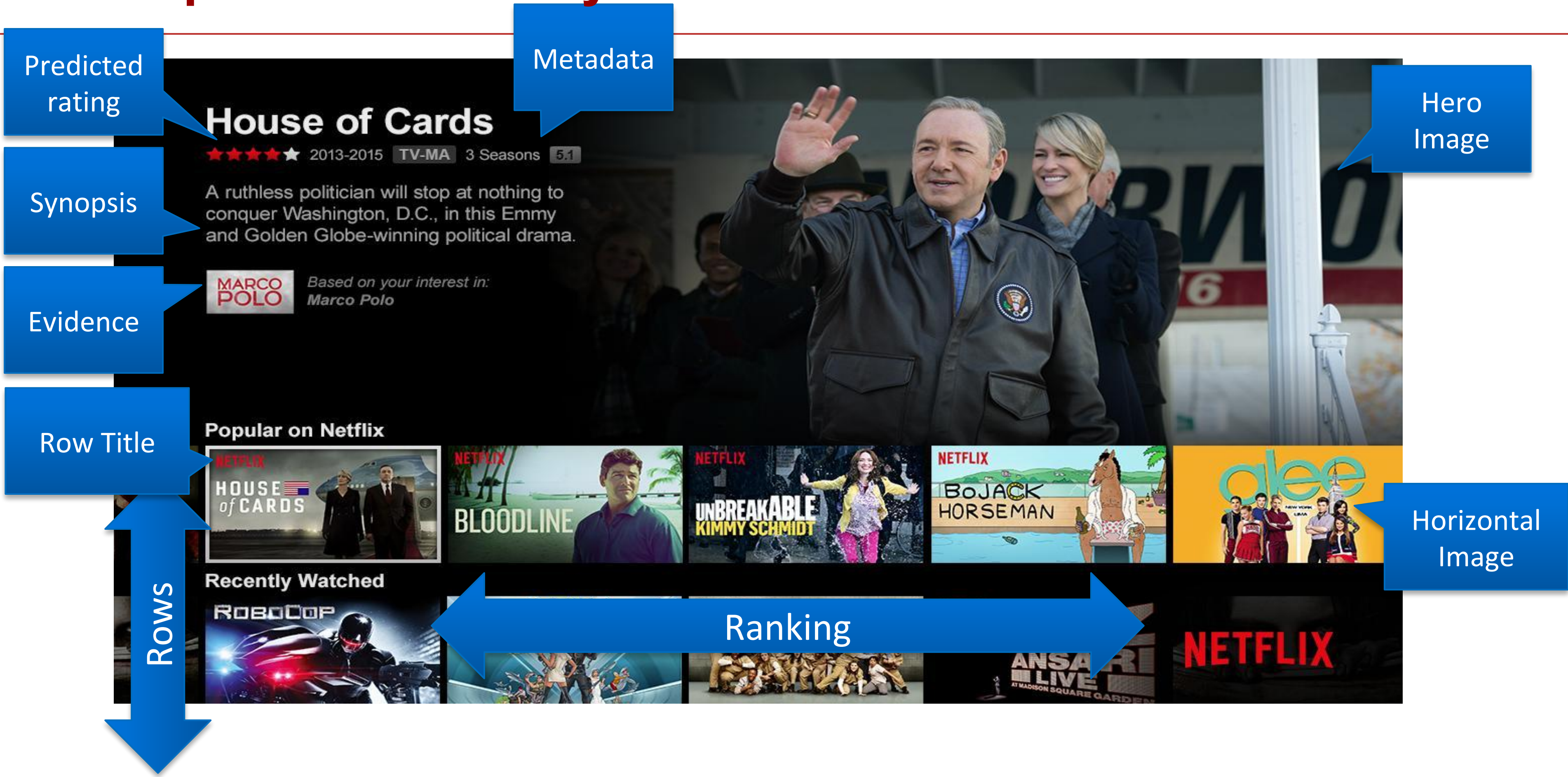
Personalizing how we recommend (not just what)

- **Algorithm level**: Ideal balance of diversity, novelty, popularity, freshness, etc. may depend on the person

- **Display level**: How you present items or explain recommendations can also be personalized
    - Select the **best information** and presentation for a user **to quickly decide** whether or not they want an item

- **Interaction level**: Balancing the needs of lean-back users and power users

Example rows and beyond



9. Conclusions

Conclusions

- Recommendation is about much more than just predicting a rating
- All forms of recommendation require of a tight connection with the UI
  - Capture the right kind of feedback
    - Explicit/implicit feedback
    - Correct for presentation bias
    - ...
  - Present the recommendations correctly
    - Explanations
    - Diversity
    - Exploration/Exploitation
    - ....

Conclusions

- For the algorithm:
    - Use implicit feedback if possible
    - Build a Matrix Factorization recommender system
    - Think of using ensembles and turning your problem into a feature engineering problem
    - Always think of the metric you are optimizing to and the data you are using
- Whatever you do in the lab, you should trust your AB tests

10. References

Other resources

- 4 hour video of my lecture at MLSS at CMU (Youtube)
- *“Recommender systems in industry: A netflix case study”* (X. Amatriain, J. Basilico) in Recommender System Handbook
- *“Past, Present, and Future of Recommender Systems: An Industry Perspective”* (X. Amatriain, J. Basilico. Recsys 2016)
- *“Mining large streams of user data for personalized recommendations”* (X. Amatriain) - ACM SigKDD Explorations Newsletter
- *“Big & personal: data and models behind netflix recommendations”* (X. Amatriain) - ACM Workshop on Big Data
- Visit my slideshare page: https://www.slideshare.net/xamat