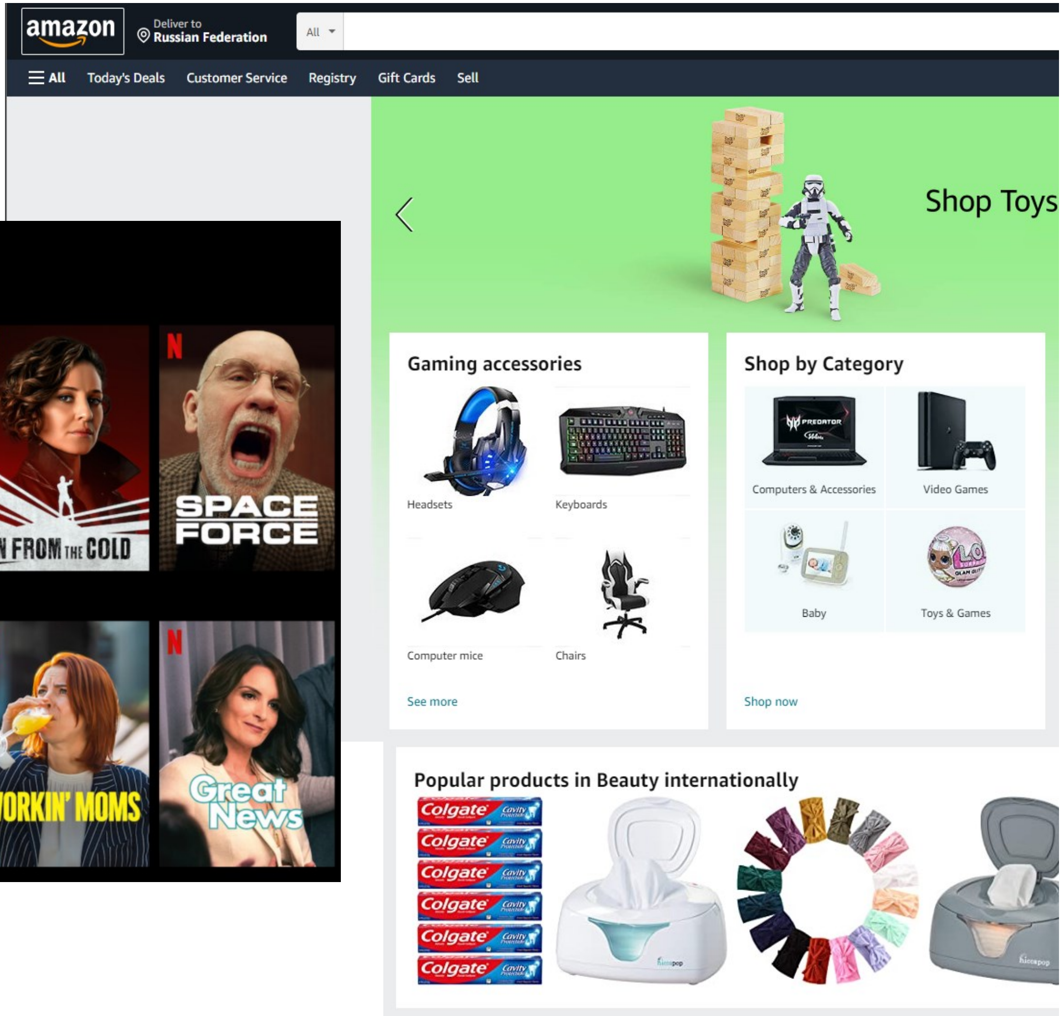
Recommender Systems

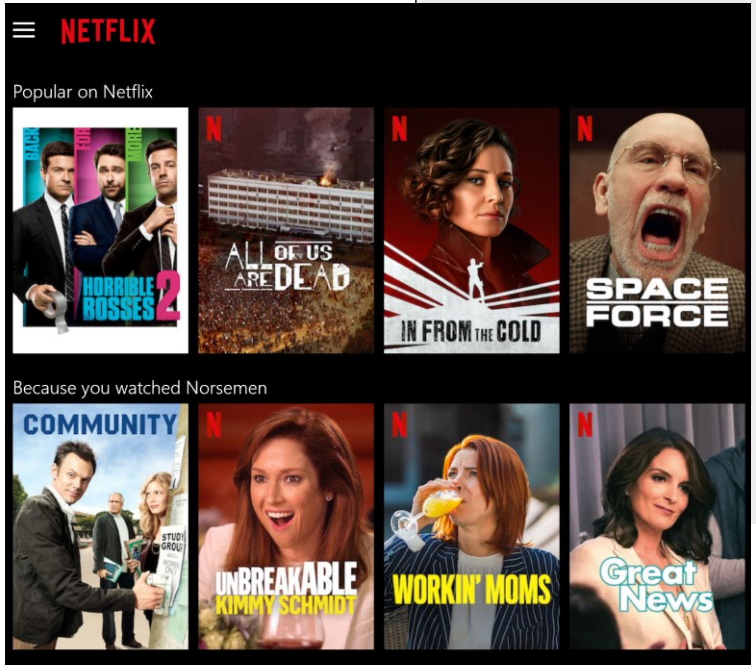
Lecture 2

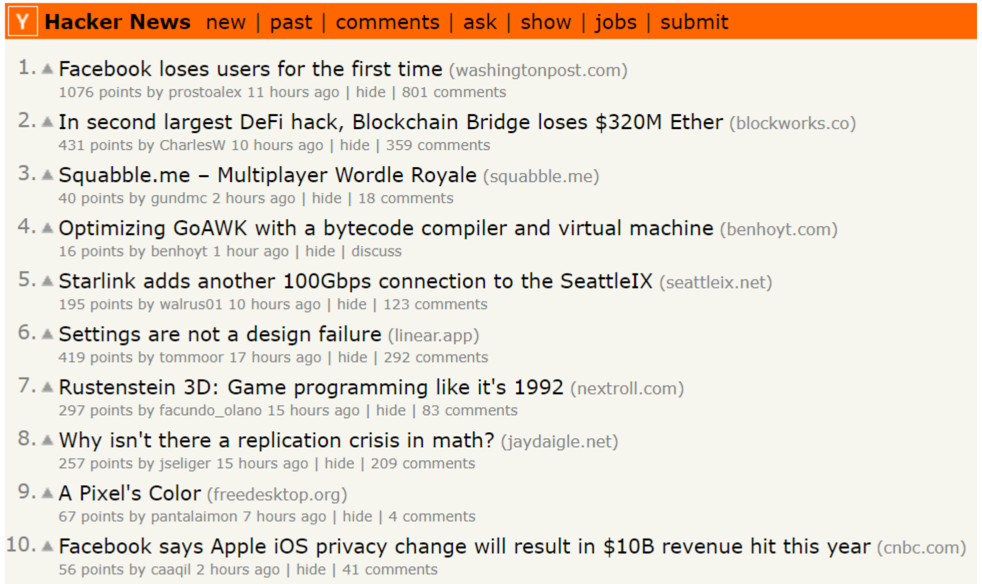
Today's Lecture

- Popularity-based models
- Content-based recommendations
- Baseline Predictors

Usecases for popularity-based predictions







Popularity-based scoring

Task:

- assign a popularity score based on accumulated feedback from users

- mostly based on heuristics
  - e.g., content “interestingness” ↔ consumption frequency

- typical challenges / issues
  - dynamics and trends
    - e.g., seasonal purchases
  - non-homogeneous distribution
    - e.g., experts vs non-experts
  - attacks and fraud

Examples of user feedback

1. Ratings
2. Upvotes/Downvotes
3. Likes (w/o dislikes)

Таблица сравнения

Grow Food	Performance Food
Официальный сайт:growfood.pro	Официальный сайт:p-food.ru
Рейтинг: 8.77	Рейтинг: 9.59
Просмотров: 127 518	Просмотров: 36 058
Отзывов: 14	Отзывов: 8

Hacker News new | past | comments | ask | show | jobs | submit

1. ▲ Seawater electrolysis by adjusting the local reaction environment of a catalyst (nature.com)
   99 points by CharlesW 2 hours ago | hide | 46 comments
2. ▲ Show HN: I trained an AI model on 120M+ songs from iTunes (maroofy.com)
   91 points by subtech 2 hours ago | hide | 60 comments
3. ▲ Play Counter Strike 1.6, with full multiplayer, in the browser (play-cs.com)
   951 points by philosopher1234 10 hours ago | hide | 372 comments
4. ▲ I'm Now a Full-Time Professional Open Source Maintainer (filippo.io)
   191 points by chmaynard 4 hours ago | hide | 29 comments
5. ▲ An obituary for the man who saved North Carolina from Nuclear Disaster (ncrabbithole.com)
   75 points by joeatwork 2 hours ago | hide | 3 comments

Rating-based scoring

Popularity-based recommendations

naïve estimation of item j popularity:

$$\mathrm{score}_{\mathrm{POP}}(j) =$$

- U_j - set of users who rated item j
- r_{ij} - rating of item j provided by user i

Popularity-based recommendations

$$\text{score}_{\text{POP}}(j) = \frac{1}{|U_j|} \sum_{i \in U_j} r_{ij}$$

What are potential flaws?

- insufficient amount of data
  - more data means higher reliability /certainty
  - an item with 100 ratings and average score 4.95 vs item with two 5-star ratings
- different dynamics
  - trends
  - seasonality

In the early days

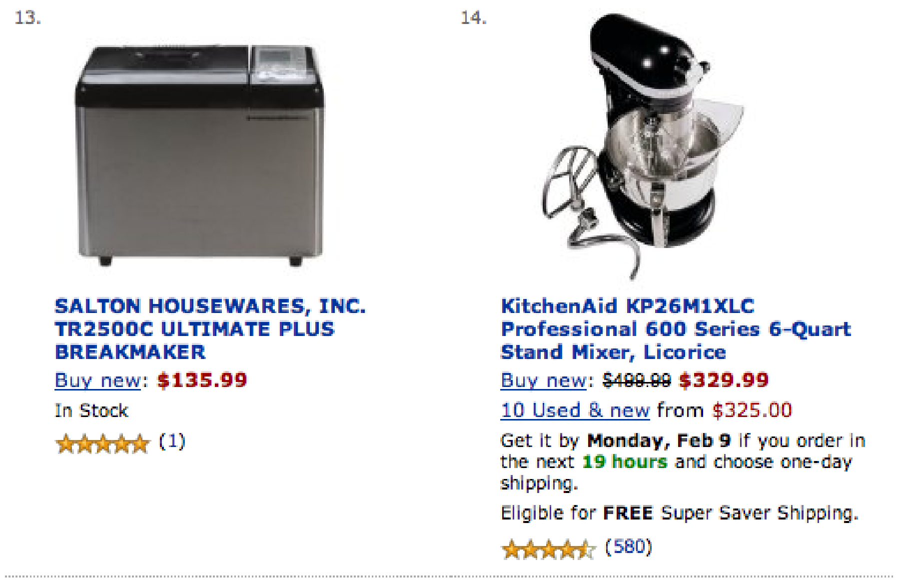
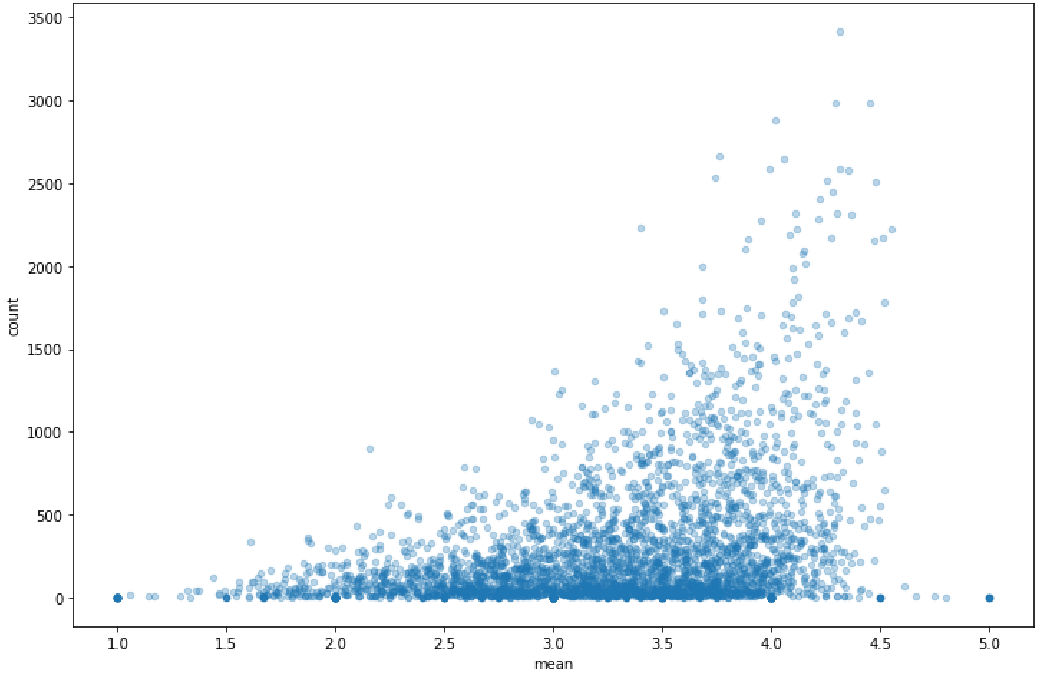


Image from https://www.evanmiller.org/how-not-to-sort-by-average-rating.html

Popular vs niche products

Simple adjustment of popularity

Average rating distribution on Movielens-1M



minimization problem:

Popularity vs trends in data

Bayesian averaging

combination of prior belief and observed sample average:

$$\mu_{BA} = \alpha \cdot \mu_{PB} + (1 - \alpha) \cdot \mu_{SA}, \qquad 0 \leq \alpha \leq 1$$

assume we're in the future and already have enough data:

x_k – sample

C – amount of prior samples

n – amount of newly added samples

Bayesian averaging

$$\mu_{BA} = \frac{C\bar{\mu} + \sum_{k=1}^{n} x_k}{C + n}, \qquad \alpha = \frac{C}{C + n}$$

- For item rating data:

$$\mu_j = \frac{C\overline{r_j} + \sum_{i \in U_j} r_{ij}}{C + |U_j|}$$

U_j - set of users, who rated items j

- too high C → need more observations
- too low C → unreliable estimate

Other possible adjustments

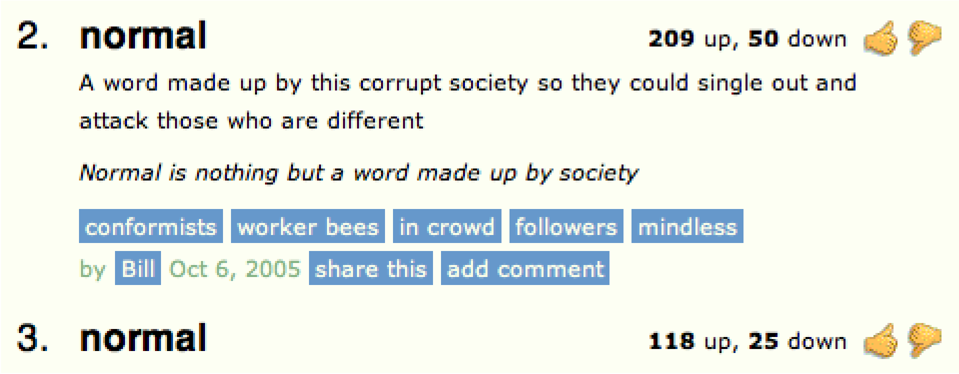
- dampening weights based on recency
- activity of users (bots/trolls or not invested users)
- side information (features, attributes)
  - typically rules-based

Scoring for binary feedback

Possible scoring strategies

- Amount of positive feedback
- \# positive feedback – # negative feedback
- proportion of positive feedback

In the early days



Urban Dictionary, image credit: https://www.evanmiller.org/how-not-to-sort-by-average-rating.html

Estimating popularity on binary feedback

- lower bound of Wilson score confidence interval*
  - estimate for a Bernoulli parameter
  - estimates the bound with 95% probability given current observations

$$\left(\hat{p} + \frac{z_{\alpha/2}^2}{2n} - z_{\alpha/2}\sqrt{\left[\hat{p}(1-\hat{p}) + z_{\alpha/2}^2/4n\right]/n}\right) \Big/ \left(1 + z_{\alpha/2}^2/n\right)$$

  - $\hat{p}$ is the observed fraction of positive ratings
  - $z_{\alpha/2}$ is the $1 - \alpha/2$ quantile of the standard normal distribution
  - n is the total number of ratings

Assumptions: data follows a binomial distribution with

- fixed probability of a success,
- statistically independent trials.

*Example of calculation in python
https://gist.github.com/aditya00kumar/ab81a77258c70f9c3a811f3763a6fb62

Laplace (add-one) smoothing

- $r_{ij} \in \{0, 1\}$ – binary feedback
  $\text{upvotes}_j = \{r_{ij} > 0, i \in U_j\}$

- $r_{ij} \in \{-1, 0, 1\}$

- $r_{ij} \in \{1, 2, 3, 4, 5\}$

Some other schemes: Valcarce, D., Parapar, J. and Barreiro, Á., 2016, June. Additive smoothing for relevance-based language modelling of recommender systems. In *Proceedings of the 4th Spanish Conference on Information Retrieval* (pp. 1-8).

Note on popularity / average ratings

- Horror movies ratings are typically lower, even if a user actually likes it.

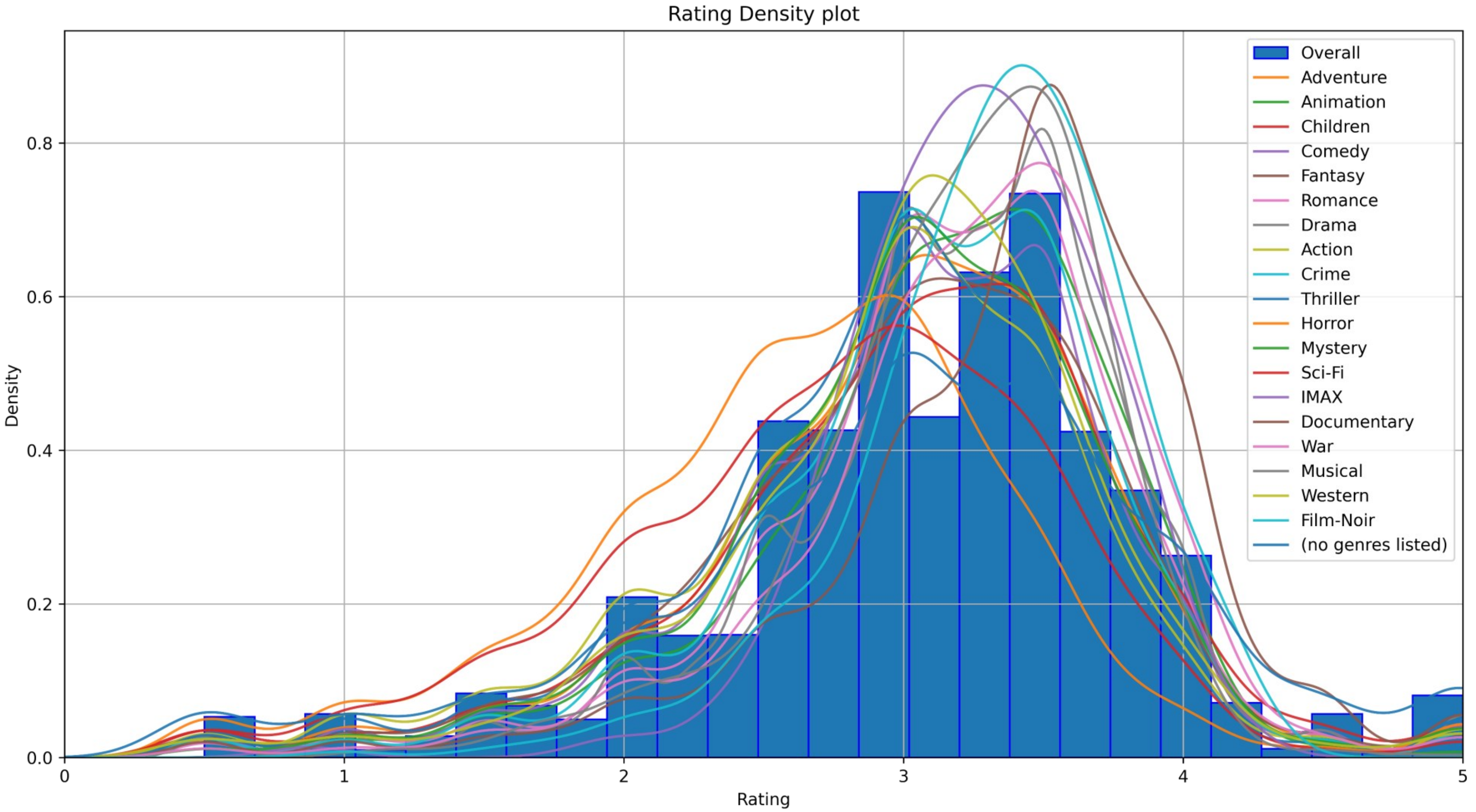
- “Ghostbusters” Is A Perfect Example Of How Internet Movie Ratings Are Broken.

- IMDb average user rating: 4.1 out of 10, of 12,921 reviewers
- IMDb average user rating among *men*: 3.6 out of 10, of 7,547 reviewers
- IMDb average user rating among *women*: 7.7 out of 10, of 1,564 reviewers

Source: http://fivethirtyeight.com/features/ghostbusters-is-a-perfect-example-of-how-internet-ratings-are-broken/

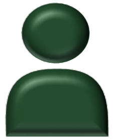
Content-based Recommendations

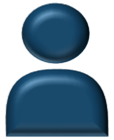
Ratings distribution over movie genres (ML-25M)

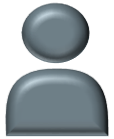


Using content for matching

			
Good for gaming	+		+
Good for movies	+	+	+
Good for TV shows	+		
Blue-ray support		+	+


Gamer


Cinema fan


“Family guy”

generating recommendations require:

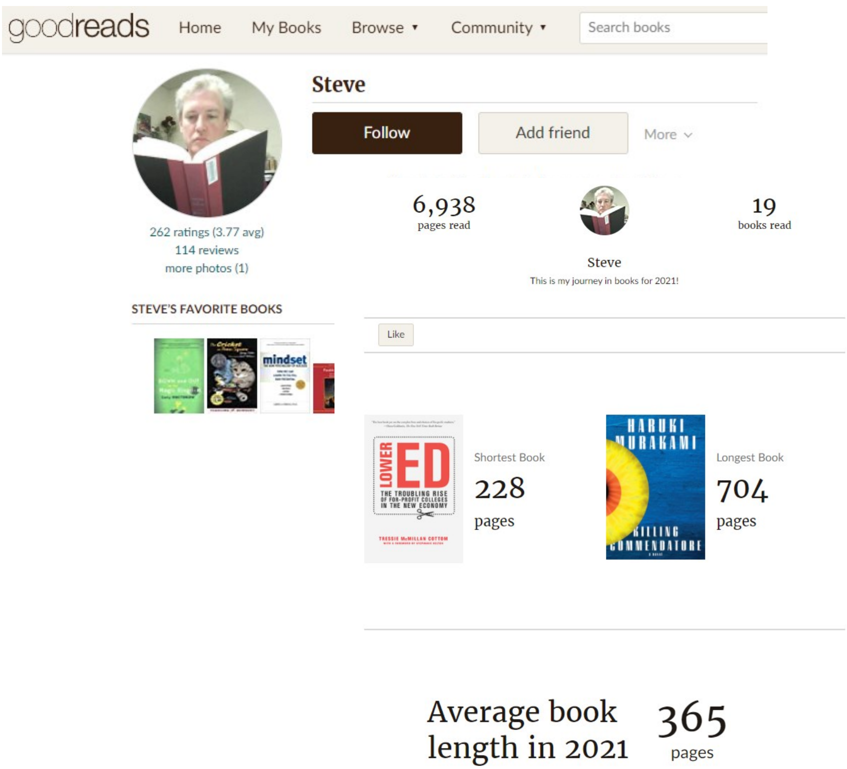
- item features to compute items relevance
- user profile information to match items
- previous user actions to match items
- user attributes for tailoring predictions
- criteria specified by user / expert

- knowledge-based
- case-based
- constraint-based

Typical use case: “cold” items

Example of content features

Users



More attributes:
- demographics
- location
- occupation
- ...

Items

Other features:
- price
- format/style
- language
- ...

Simple rating prediction

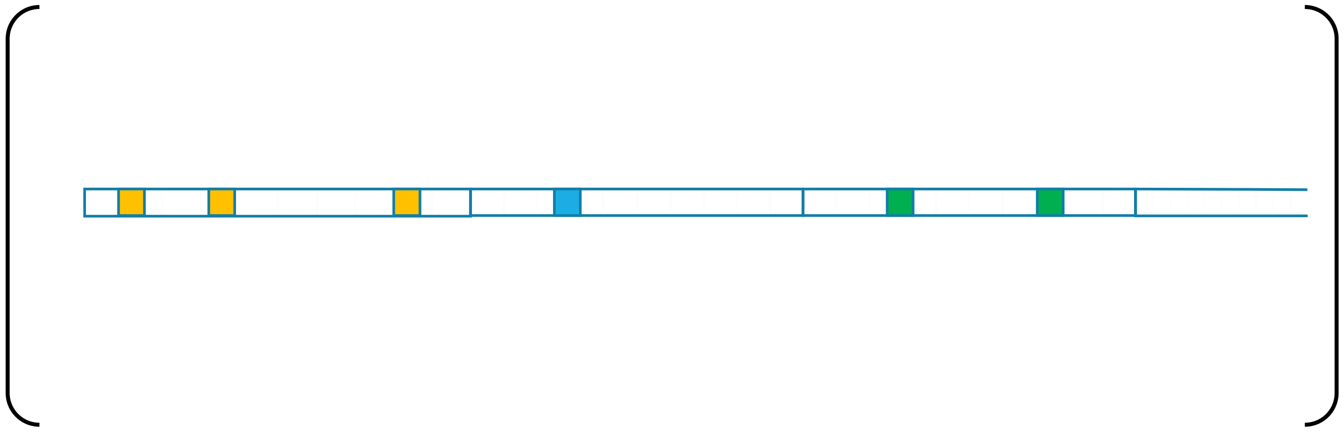
Task: find a utility function f:

$$f(\text{item features}) \rightarrow \text{feedback}$$

Linear regression model: $\quad X\boldsymbol{w} = \boldsymbol{y}$

Linear w.r.t. what?

$x_k^\top$



Linear Regression

$$\boldsymbol{y} = X\boldsymbol{w} + \boldsymbol{\epsilon}$$

$$y_k = \boldsymbol{x}_k^\top \boldsymbol{w} + \epsilon_k$$

Under assumption of normally distributed noise, standard linear least squares problem:

$$||\boldsymbol{y} - X\boldsymbol{w}||_2^2 \rightarrow \min$$

Multicollinearity problem

Multicollinearity problem

Typical ways to combat the ill-posedness:

- add regularization term
  - e.g., l2-penalty $\lambda||\boldsymbol{w}||^2$
- feature selection
  - hard problem
- feature transformation
  - e.g., dimensionality reduction

Ridge regression

How to select features?

Types of features:

- Known knowns
- Known unknowns
- Unknown unknowns

Adding/omitting regressors:

- Missing important variables → bias
- Adding meaningless variables → overfitting

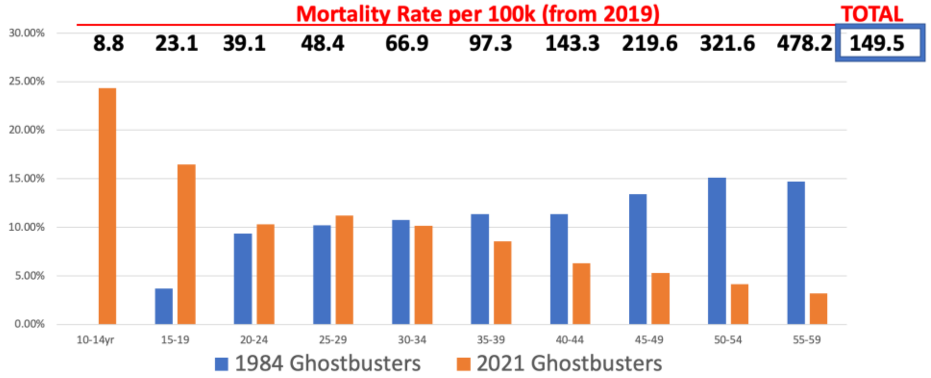
- No standard "silver bullet" recipe exists.
- Choose the simplest possible model, but not simpler.

Simpson Paradox

Correlation vs causation

Is watching the 1984 Ghostbusters movie killing people?

Age Distribution of UK residents Age 10-59yr
watching 1984 Ghostbusters and 2021 Ghostbusters



Images from https://www.covid-datascience.com/post/is-watching-the-1984-ghostbusters-movie-killing-people-a-statistician-s-perspective

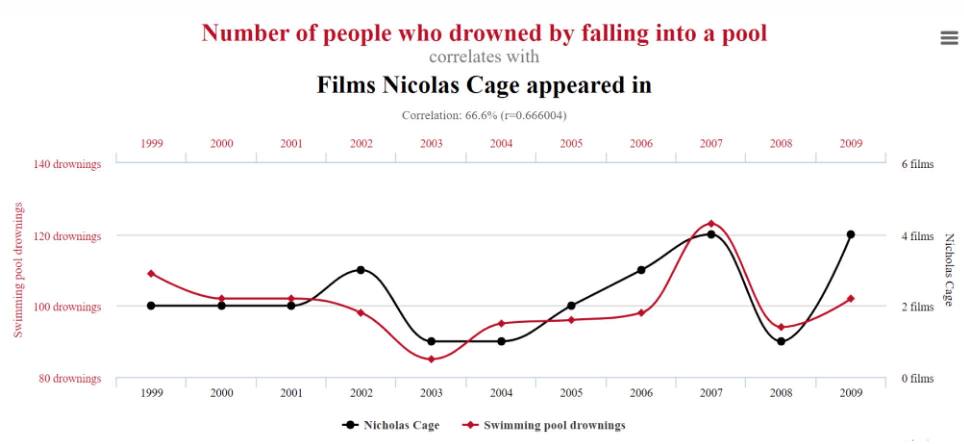


Image from: https://tylervigen.com/spurious-correlations

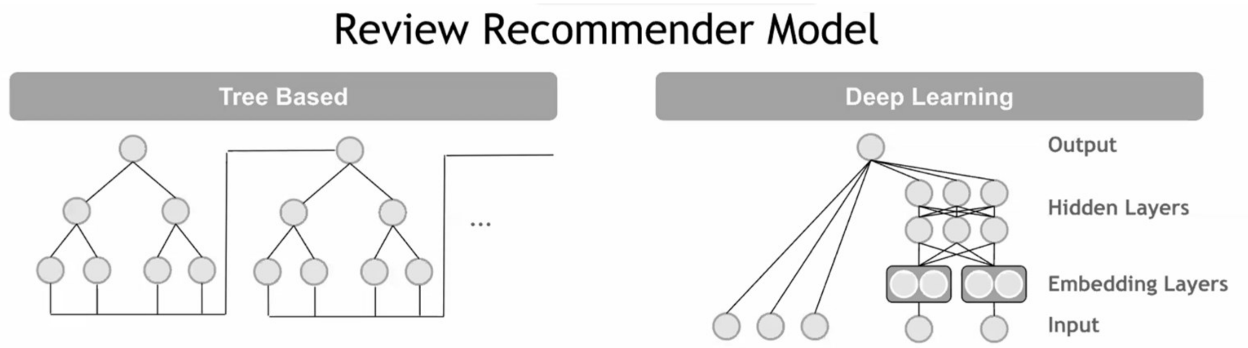
Other methods used for content-based filtering

- SVM
- Decision Tree-based models
  - CatBoost
  - LightGBM
  - XGBoost
  - Automated feature selection, e.g. as in LightAutoML
- Neural Networks

Feature engineering usecase

Feature Type	Example	Feature Engineering
Categorical	• User ID / Item ID • Brand • Main Category	• Target Encoding • Count Encoding • Categorify + Combining Categories
Unstructured list	• Keywords • Subcategories • Colors	• Target Encoding • Count Encoding • Categorify
Numeric	• Price • Deliver time • Avg. reviews	• Binning • Normalization • Gauss Rank
Timestamp	• Timestamp	• Extract month, weekday, weekend, hour
Timeseries	• Events in order • Time since last event	• # of events in past X • Difference in time (lag)
Image	• Product image	• Extract latent representation with deep learning
Text	• Description	• Extract latent representation with deep learning
Social graph	• Follower/Following graph	• Link analysis
Geo location	• Addresses	• Distances to point of interest

- Massive experiments in parallel
- Trying, not guessing



Types:
- CatBoost
- XGBoost
- LightGBM

Components:
- Defines split by information gain per feature
- Does not require normalization of input features

XGBoost cannot handle raw categorical features

Types:
- Wide And Deep
- DeepFM
- DLRM

Components:
- Embedding Layers
- Feed-Forward Layers
- Requires normalization of input features

How to Build a Winning Deep Learning Recommender System
https://medium.com/rapids-ai/winning-solution-of-recsys2020-challenge-gpu-accelerated-feature-engineering-and-training-for-cd67c5a87b1f
https://www.youtube.com/watch?v=bHuww-l_Sq0

Classification

Task: predict whether user will like an item or not.

Example:

$$y_k = \begin{cases} 1, & \text{rating above threshold}, \\ 0, & \text{otherwise}. \end{cases}$$

In linear regression: $y_k \approx \boldsymbol{x}_k^\top \boldsymbol{w}$.

How to translate $\boldsymbol{x}_k^\top \mathbf{w} \in \mathbb{R}$ into binary domain?

Classification – probabilistic formulation

Logistic Regression for classification task

The model:

$$p_{\mathbf{w}}(y_k = 1 \mid \boldsymbol{x}_k) \sim \frac{1}{1 + e^{-\boldsymbol{x}_k^\top \boldsymbol{w}}}$$

Target:

$$y_k = \begin{cases} 1, & \text{if } \boldsymbol{x}_k^\top \mathbf{w} > 0, \\ 0, & \text{otherwise.} \end{cases}$$

- Our observations data can be viewed as a set of random trials.
- Outcome of each trial is binary (coin flip).

$$y \sim \text{Bernulli}(\mu)$$

$$E[y] = \mu, \qquad 0 \leq \mu \leq 1$$

A bit of probability theory

- We aim to maximize the joint probability distribution (likelihood)

$$p_w(y, X) = \prod_{k=1}^{n} p_{\mathbf{w}}(y_k, \boldsymbol{x}_k), \qquad p_{\mathbf{w}}(y_k, \boldsymbol{x}_k) = p_{\mathbf{w}}(\, y_k \mid \boldsymbol{x}_k\,) \cdot p_{\mathbf{w}}(\boldsymbol{x}_k)$$

$$\operatorname{argmax}_{\mathbf{w}} \prod_{k=1}^{n} p_{\mathbf{w}}(y_k, \boldsymbol{x}_k) = \operatorname{argmax}_{\mathbf{w}} \log \prod_{k=1}^{n} p_{\mathbf{w}}(y_k, \boldsymbol{x}_k)$$

$$= \operatorname{argmax}_{\mathbf{w}} \left(\log \prod_{k=1}^{n} p_{\mathbf{w}}(\, y_k \mid \boldsymbol{x}_k\,) + \log \prod_{k=1}^{n} p_{\mathbf{w}}(\boldsymbol{x}_k) \right)$$

$$= \operatorname{argmax}_{\mathbf{w}} \sum_{k=1}^{n} \log p_{\mathbf{w}}(\, y_k \mid \boldsymbol{x}_k\,)$$

Maximum likelihood derivation

$$\mathcal{L}(w) = \sum_{k=1}^{n} \log p_{\boldsymbol{w}}(\, y_k \mid \boldsymbol{x_k}\,)$$

- The probability of each outcome is parametrized via:

$$p_w(\, y_k \mid \boldsymbol{x}_k\,) = p_{\boldsymbol{w}}(\, y = 1 \mid \boldsymbol{x_k}\,)^{y_k} \cdot \left(1 - p_{\boldsymbol{w}}(\, y = 1 \mid \boldsymbol{x}_k\,)\right)^{(1-y_k)}$$

Logistic Regression Objective

More concise/practical form with $y_k \in \{-1, +1\}$ instead of $\{0, 1\}$:

$$\mathcal{L}(\boldsymbol{w}) = -\sum_{k=1}^{n} \log\left(1 + e^{-y_k \cdot \boldsymbol{x}_k^\top \boldsymbol{w}}\right)$$

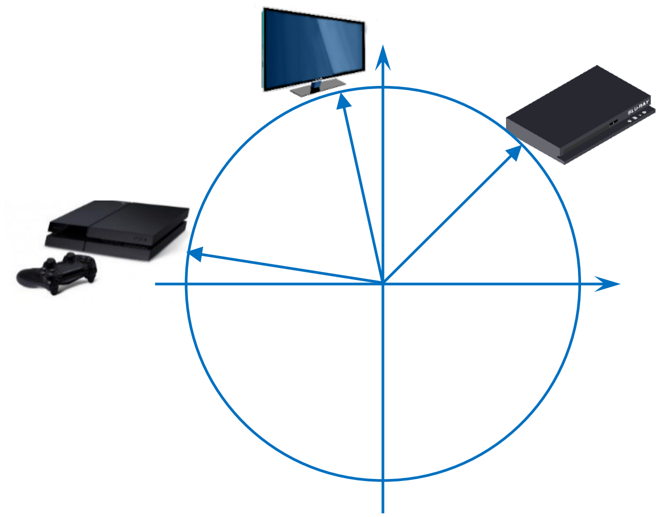
- Turns into standard loss minimization:

$$\boldsymbol{w}^* = \mathrm{argmin}_{\mathbf{w}}\, \mathcal{Q}(\boldsymbol{w}), \qquad \mathcal{Q}(\boldsymbol{w}) = -\mathcal{L}(\boldsymbol{w}) + \lambda \|\boldsymbol{w}\|^2$$

- Can be solved with e.g.:
  - SGD
  - Newton-Raphson (Iteratively Reweighted Least Squares)

Similarity-based models

Each item as a vector in d-dimensional feature space.



Given two item feature vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, we can estimate their proximity in on a unit sphere:

$$\cos(\alpha) = \frac{\boldsymbol{x}_i^\top \boldsymbol{x}_j}{\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{x}_j\|}$$

What happens if some features are missing?

How to match with a user?

$\boldsymbol{x}_k^\top$

User profile:

$$\overline{\boldsymbol{x}}_u = \sum_{j \in I_u} w_{uj} \boldsymbol{x}_j \,, \qquad I_u - \text{set of items of user } u$$

The simplest case:

$$w_{uj} = \frac{1}{|I_u|} \quad \text{or} \quad w_{uj} = \frac{r_{uj}}{\sum_k r_{uk}} \,,$$

Relevance estimation:

$$r_{ui} = \frac{\overline{\boldsymbol{x}}_u^\top \boldsymbol{x}_i}{\|\overline{\boldsymbol{x}}_u^\top\| \cdot \|\boldsymbol{x}_i\|}$$

How to evaluate

Intuitive heuristic:

- Based on proximity to user preferences:
  - must be close to what user likes
  - must be distant from what user dislikes

- Construction of evaluation dataset:
  - select *k* pairs of likes/dislikes for each user
    - must be excluded from training data
  - rank other items (e.g., user favorites) based on their proximity to test pairs