Building a Recommendation system for e-commerce

AI Ukraine 2017

About me

Alex Konduforov

Data Science Group Leader @

Co-organizer @ Kharkiv AI club

Business

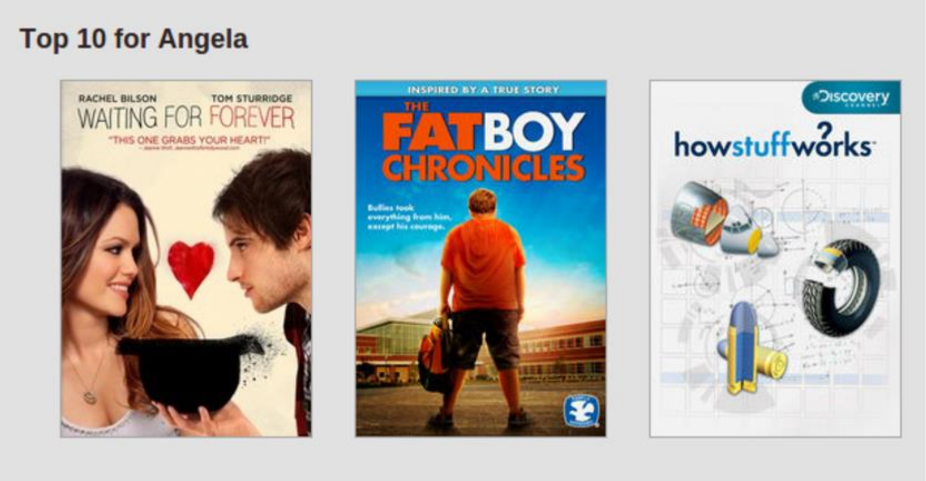
“31% of ecommerce revenues were generated from personalized product recommendations” - *Barilliance.com, 2014*

“Already, 35% of what consumers purchase on Amazon and 75% of what they watch on Netflix come from product recommendations based on such algorithms” - *McKinsey*

Evolution

Amazon

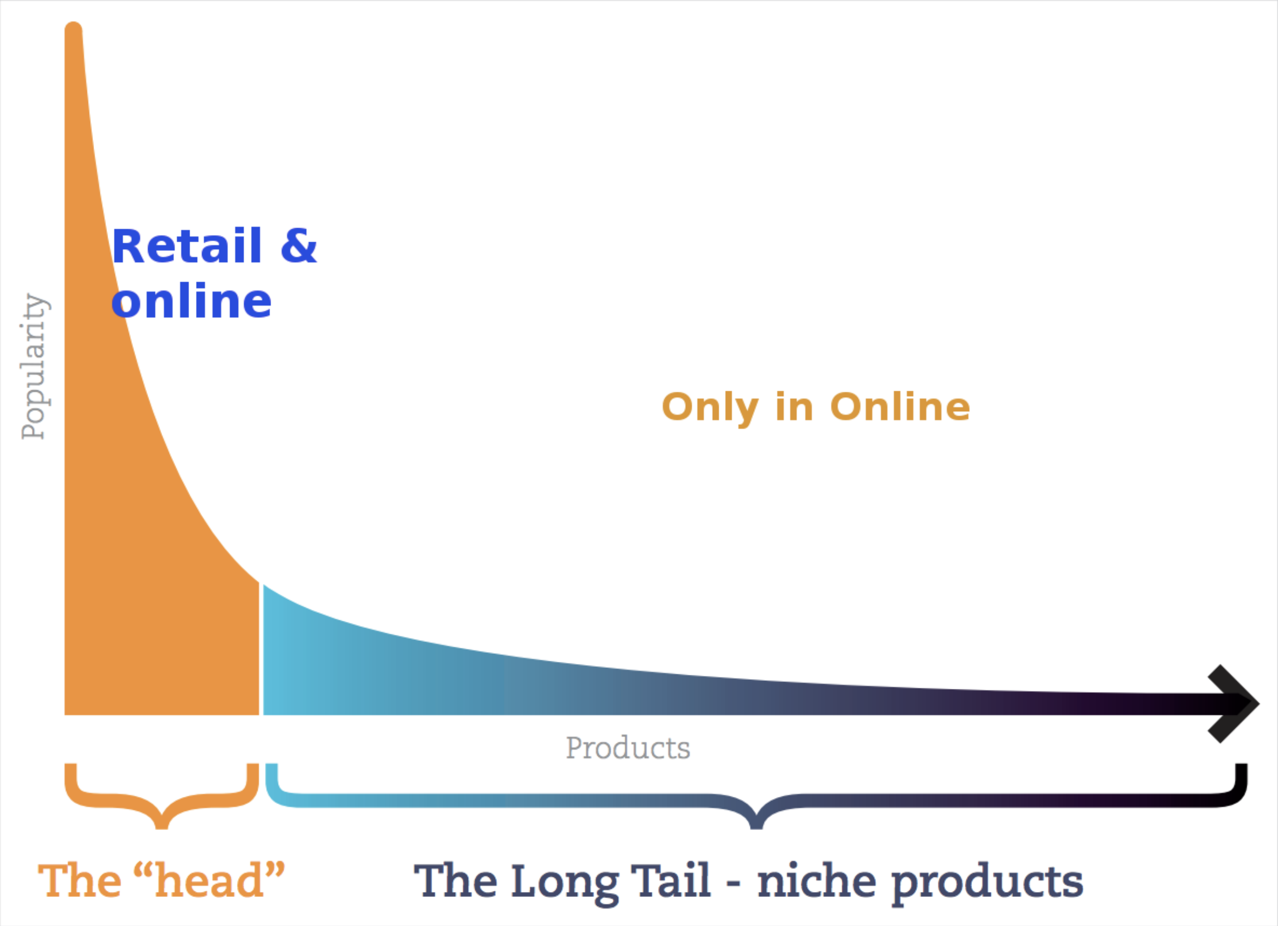
Netflix



Why recommendations so important

Traditional Retail can serve only most popular products.

Online can serve much more products, but it’s overwhelming for customers.



AI UKRAINE

How to apply

Website recommendations

- Main goals: cross-sale, save customer time

Personalized marketing emails

- Main goals: return customer on the website, upsale

AI UKRAINE

Recommendation systems

Formulation of the problem

Goal of recommendation system is to predict blanks in the utility matrix

	LOTR	Star Wars	GoT	Matrix
Alice	5			2
Bob	4	5	3	
Carol		3	4	
David			5	5

Gathering Data

Explicit

- Ask people to rate items
- Cons: doesn’t scale, only a small fractions of users leave ratings and reviews

Implicit

- Inferences from user actions
- Cons: only one value, no difference between dislike and unknown

AI UKRAINE

Main approaches

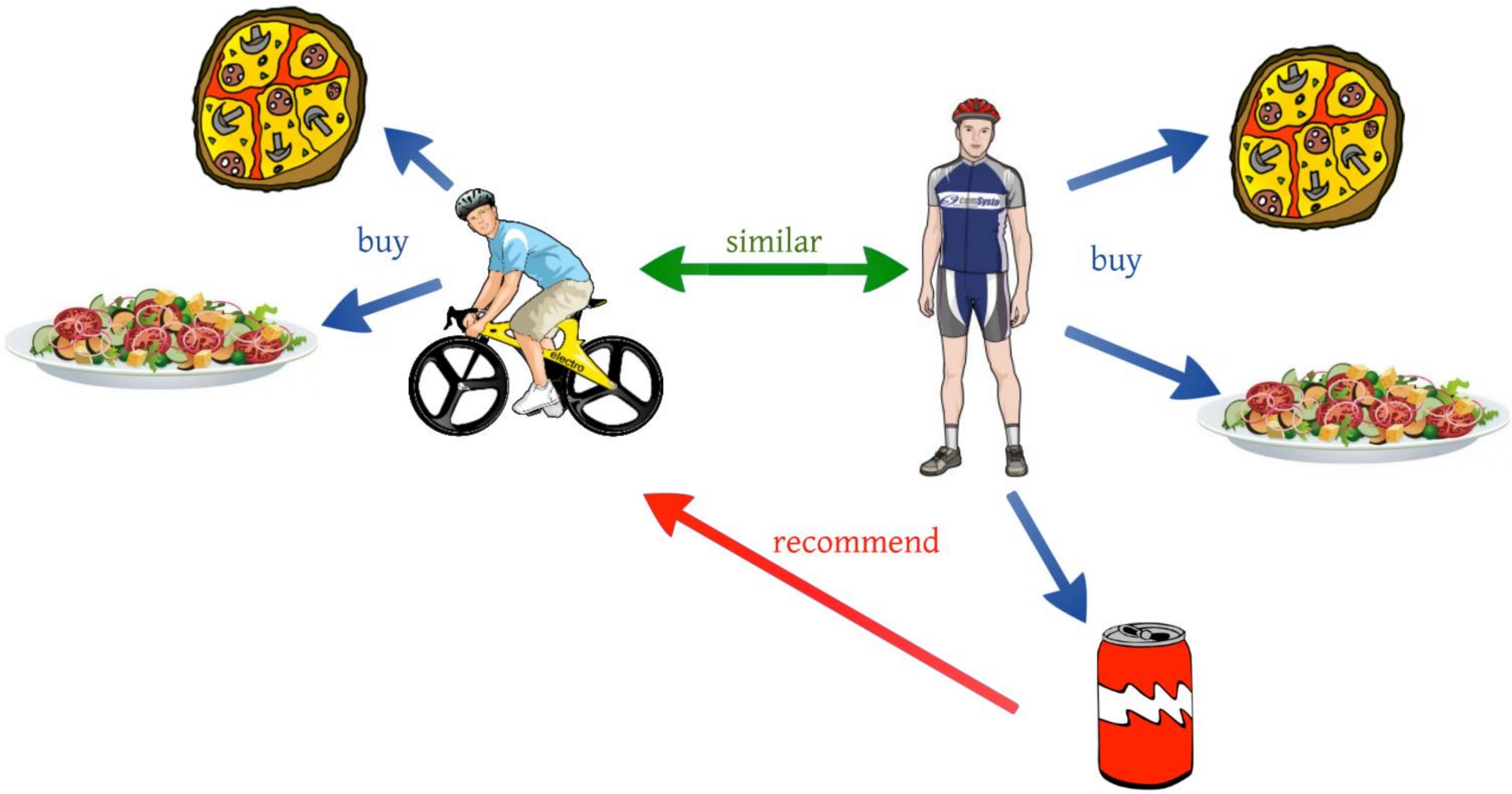
- Non-personalized Summary Statistics
- Content-based Filtering
- Collaborative Filtering (nearest neighbors)
  - User-User
  - Item-Item
  - Matrix Factorization
- Hybrid
- Probability models
- etc.

Ecommerce specifics

- Implicit customer feedback (views, purchases, other actions)
- Utility matrix with only 1’s
  - Possible to calculate some score but more complicated
- Collaborative Filtering + Matrix Factorization
- Not every similarity/distance works

Collaborative Filtering

Method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating)



Types of Collaborative Filtering

User-to-user

1. Look for users who share the same rating patterns with the active user (the user whom the prediction is for)
2. Use the ratings from those like-minded users to calculate a prediction for the active user

Item-to-item

1. Build an item-item matrix determining relationships between pairs of items
2. Infer the tastes of the current user by examining the matrix and matching that user's data

User-to-user

- Let r_x be the vector of user x's ratings
- Let N be the set of k users most similar to x who have also rated item i
- Prediction for user x and item i

- Option 1: $r_{xi} = 1/k \sum_{y \in N} r_{yi}$
- Option 2: $r_{xi} = \sum_{y \in N} s_{xy} r_{yi} / \sum_{y \in N} s_{xy}$

where $s_{xy} = \text{sim}(x,y)$

Similarity

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Jaccard similarity

- sim(A,B) = | $r_A \cap r_B$ | / | $r_A \cup r_B$ |
- sim(A,B) = 1/5; sim(A,C) = 2/4
  - sim(A,B) < sim(A,C)

Ignores rating values

Pearson similarity (~cosine)

$$\mathrm{simil}(x,y) = \frac{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r_x})(r_{y,i} - \bar{r_y})}{\sqrt{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r_x})^2 \sum_{i \in I_{xy}} (r_{y,i} - \bar{r_y})^2}}$$

Contrary to cosine treats missing values not as negatives, but as zeros

Item-to-item

- For item ***i***, find other similar items
- Estimate rating for item ***i*** based on ratings for similar items
- Can use same similarity metrics and prediction functions as in user-user model

$$\boxed{r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}}$$

s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***
N(i;x)… set items rated by ***x*** similar to ***i***

AI UKRAINE

User-based vs. item-based

In practice, item-based CF outperforms user-based CF in many cases

Item-based CF pros:

- better when user size is large
- better for new users
- no need to recalculate so often as user-based (caching)
- more likely to converge => better accuracy

AI UKRAINE

Matrix Factorization

Approximates the utility matrix as product of low-rank matrices

Identifies latent features

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	X		X		X	
User 2		X	X			
User 3				X		X
User 4					X	
User 5	X	X		X		X
User 6			X	X		
User 7	X	X	X		X	X
User 8		X		X		
User 9			X			

R

$\cong$

	UF1	UF2
User 1		
User 2		
User 3		
User 4		
User 5		
User 6		
User 7		
User 8		
User 9		

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
IF1						
IF2						

$P \times Q$

Matrix Factorization algorithm

- Initialize P and Q with small random numbers
- Teach P and Q
    - Alternating Least Squares
    - Stochastic Gradient Descent

$$r'_{ui} = p_u^T q_i$$

$$L = \sum_{u,i \in S} (r_{ui} - \mathbf{x}_u^{\mathsf{T}} \cdot \mathbf{y}_i)^2 + \lambda_x \sum_u \|\mathbf{x}_u\|^2 + \lambda_y \sum_u \|\mathbf{y}_i\|^2$$

AI UKRAINE

MF example

Latent features are calculated via MF:

Rating Matrix (User × Item):

User \ Item	W	X	Y	Z
A		4.5	2.0	
B	4.0		3.5	
C		5.0		2.0
D		3.5	4.0	1.0

=

User Matrix:

A	*1.2*	*0.8*
B	*1.4*	*0.9*
C	*1.5*	*1.0*
D	*1.2*	*0.8*

X

Item Matrix:

W	X	Y	Z
1.5	*1.2*	*1.0*	*0.8*
1.7	*0.6*	*1.1*	*0.4*

Evaluation

Academic metrics:

- RMSE
- MAE
- Precision/Recall

(all may have low correlation with actual user satisfaction)

Business metrics:

- CTR/CVR
- ROI
- CLV (Customer Lifetime Value)

Customer metrics:

- Coverage – covering more items for recommendations
- Diversity – higher variety of items (rich-get-richer effect)
- Novelty – recommending new items

AI UKRAINE

Sparsity problem

There is an approximate threshold of 99.5% sparsity for CF to work

- Add product views, shopping cart and other activities
  - Decreases sparsity
- Matrix Factorization, SVD
  - No zeros
- Content description
  - Hybrid content-based + collaborative filtering

Cold start problem

User cold start: new users

- Non-personalized recommendations: most popular, highly rated
- Use user profile (age, gender, etc.) and segment

Item cold start: new items

- Don’t recommend (what about news?)
- Use item content if available

Scalability problem

Amazon had 30+ mln of customers and several million catalog items.

Solution:

- Reduce number of customers by randomly sampling them or discarding customers with few purchases
- Reduce number of items by discarding very popular or unpopular items
- Dimensionality reduction techniques such as clustering

Other challenges

- Gray sheep
- Diversity and the long tail (rich-get-richer effect)
- Shilling attacks
- Privacy
  - EU has quite strict rules and culture of data privacy
  - Netflix was sued for dataset publication => cancellation of a second Netflix Prize competition in 2010

Implementation questions

- For CF+MF - automatic model updates? how frequently?
- How and where to store MF model?
- Emails - track recommended items and don’t duplicate

Tools

Language / Stack	Tools / Libraries
R	recommenderlab, recosystem
Python	Scikit-learn crab, implicit, python-recsys, Surprise GraphLab Create ($$$)
Java	LensKit, Cofi Apache Mahout
C++	SVDFeature, Waffles, Graphchi, LIBMF GraphLab Create ($$$)
C#	Nreco
Node.JS	raccoon
SaaS	Google Cloud Prediction API Amazon Machine Learning PredictionIO SuggestGrid

https://github.com/grahamjenson/list_of_recommender_systems

Materials

- A Gentle Introduction to Recommender Systems with Implicit Feedback
- Matrix Factorization: A Simple Tutorial and Implementation on Python
- Matrix Factorization Model in Collaborating Filtering
- Finding similar music using Matrix Factorization
- Mining of Massive Databases (Stanford), Chapter 9
- AI Ukraine 2014 - Сергей Николенко - Рекомендательные системы
- Recommender Systems specialization (Coursera)

AI UKRAINE

Thank you!

Skype: alex_konduforov

Email: alex.konduforov@altexsoft.com

AI
UKRAINE