Language

- lecture today: English
- course materials: English
- rest of lectures, your presentations: probably English
- personal consultations, project interface: English, Czech, Slovak
Very Brief Overview

- project-based course
- projects typically in teams (2-4 students)
- 6 lectures, consultations, presentations
- attendance registered (although not strictly compulsory)
Your Experience?

- machine learning, data mining
- information retrieval
- web implementation (PHP/Python, databases, JavaScript, ...)

A: good, B: reasonable, C: basic or none
Today

- motivation
- main notions
- course organization
- project discussion – mapping of preferences, brainstorming
Motivation

- information overload
  - many choices available
  - “the paradox of choice” (jam experiment, choice overload)
- recommender system
  - provide aid
  - set of items + user “context” $\Rightarrow$ selection of items (predicted to be “good” for the user)
Motivation

1. What recommender systems do you know?
2. What recommender systems would you like to have?
Examples of Applications

- movies, online videos
- music
- books
- software (apps)
- products in general
- people (dating, friends)
- services (restaurants, accommodation, ...)
- research articles
- jokes
personalization, adaptation

recommender systems
focus of the course on recommendations

sometimes excursion into related techniques (personalization, adaptation)
  - educational applications: mastery learning
Value of Recommendations

- Netflix: 2/3 of the movies watched
- Amazon: 35% sales
- Google news: recommendations ⇒ 38% more clickthrough
Recommender systems reduce information overload by estimating relevance

Recommender Systems: An Introduction (slides)
Types of Recommender Systems

Personalized recommendations

User profile & contextual parameters

Recommendation component

Recommendation list

Recommender Systems: An Introduction (slides)
Types of Recommender Systems

Collaborative: "Tell me what's popular among my peers"

- User profile & contextual parameters
- Community data
- Recommendation component
- Recommendation list

Recommender Systems: An Introduction (slides)
Types of Recommender Systems

Content-based: "Show me more of the same what I've liked"

User profile & contextual parameters

Recommendation component

Recommendation list

Product features

<table>
<thead>
<tr>
<th>Title</th>
<th>Genre</th>
<th>Actors</th>
<th>...</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>item</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>i1</td>
<td>0.9</td>
</tr>
<tr>
<td>i2</td>
<td>1</td>
</tr>
<tr>
<td>i3</td>
<td>0.3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Types of Recommender Systems

Knowledge-based: "Tell me what fits based on my needs"

Product features

Knowledge models

Title, Genre, Actors, ...

item | score
--- | ---
i1 | 0.9
i2 | 1
i3 | 0.3
... | ...
Types of Recommender Systems

Hybrid: combinations of various inputs and/or composition of different mechanism

- User profile & contextual parameters
- Community data
- Product features
- Knowledge models

Recommendation component

Recommendation list

<table>
<thead>
<tr>
<th>item</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>i1</td>
<td>0.9</td>
</tr>
<tr>
<td>i2</td>
<td>1</td>
</tr>
<tr>
<td>i3</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Recommender Systems: An Introduction (slides)
Recommender System Functions

- provider’s point of view
- user’s point of view
Recommender System Functions

Provider’s point of view:

- sell more items
- sell more diverse items (long tail)
- increase user satisfaction, fidelity
- better understand what users want

Long tail:
Recommender System Functions

User’s point of view:
- looking for something:
  - find some good items
  - find all good items (closer to information retrieval)
  - recommend a sequence, a bundle
- just browsing
- side-effects (collaborative filtering systems):
  - express self
  - help others
  - influence others
Warning: Implementing Personalized Systems is Difficult

- (sometimes) complex algorithms
- (always) difficult debugging, testing, evaluation
  - personalization $\Rightarrow$ different behaviour for each user
  - hard to distinguish bugs and surprising results
Usefulness of Recommendations

Implementing recommendations is non-trivial.

Is it worthwhile? It depends...

- Is there “large” number of items?
- Do users know exactly what are they looking for?
RecSys and Information Retrieval

**Information retrieval** is the activity of obtaining information resources relevant to an information need from a collection of information resources. (Wikipedia)

The goal of a **Recommender System** is to generate meaningful recommendations to a collection of users for items or products that might interest them. (Melville, Sindhwni)

- RecSys and IR closely connected (many similar or analogical techniques)
- different goals:
  - IR – “I know what I’m looking for”
  - RecSys – “I’m not sure what I’m looking for”
Serendipity

- unsought finding
- unexpected, but useful result
- do not recommend items the user already knows or would find anyway, try something more interesting
- example – books:
  - I like books by Remarque, Potok, Skácel
  - recommending another book by Remarque not very useful
  - recommending Munro = serendipity
A Brief History

- 1990s – first systems (e.g., GroupLens), basic algorithms
- 1995-2000 – rapid commercialization, challenges of scale
- 2000-2005 – research explosion, mainstream applications
- 2006 – Netflix prize
- 2007 – the first Recommender Systems conference
- 2010s – applications common
- now – very active research, many applications
Netflix Prize

- Netflix – video rental company
- contest: 10% improvement of the quality of recommendations
- collaborative filtering
- prize: 1 million dollars
- data: user ID, movie ID, time, rating
Recommender Systems Conference Today

- very large conference
- insight into both current research and applications

commercial sponsors RecSys conference:

**DIAMOND SUPPORTER**
- Spotify

**PLATINUM SUPPORTERS**
- criteo labs
- HUAWEI
- hulu
- Netflix
- pandora
- zalando

**GOLD SUPPORTERS**
- amazon
- contentwise
- Medium
Collaborative Filtering

“tell me what’s popular among my peers (≈similar user)”

- one of the most often and successfully used techniques
- widely applicable, does not need any domain knowledge
- interesting analogies, metaphors, questions
  - ants, social insect: communication via pheromone
  - recommender systems: people ∼ ants, ratings (clicks) ∼ pheromone
  - between human intelligence and (good old-fashioned) artificial intelligence
Ratings

- recommender systems (particularly collaborative filtering) rely on user “ratings”
- rating of item $\sim$ how much the user likes the item
- many different forms of ratings
- what kinds of ratings do you know (can you imagine)?
- what are their advantages and disadvantages?
Ratings

- explicit
  - Likert scale (5 stars), like/dislike
  - require additional effort from users

- implicit
  - click through rate, buying an item, visiting a page, viewing a video, dwell time
  - easier to collect, less precise
  - more “honest” (Netflix example: highly rated vs watched)

Recommended reading: https://www.wired.com/2013/08/qq_netflix-algorithm

“We know that many of the ratings are aspirational rather than reflecting your daily activity.”
Potential Downside

- serving “low instincts” instead of “high aspirations”?
- news, optimizing clicks:
  - sex, tragedy, fear, celebrity
  - thorough analysis, complex problems
Personalization in general, collaborative filtering specifically

- “filter bubbles”
- news, social media
- users only see what they are expected to like
  - good for business (in the short term)
  - potentially bad (in the long term) for users and society
Downsides: What does it mean for us?

- do not “throw away” collaborative filtering techniques
- be aware of the limitations
- try to address limitations in suitable way (depending on the application)
Goals, Evaluation

- What is the goal of the system?
- How do we evaluate a recommender system?
- What is a “good” recommender system?
- How do we quantify the performance?

*important topics of the course*
RecSys and Educational Domain

- learning materials – direct application
- problems, exercises:
  - users ~ students
  - items ~ problems
  - ratings ~ performance (correctness of answers, problem solving times)
Personalization in Education

- adaptive learning, personalized learning, ...
- well-known:
  - open systems: Khan Academy
  - commercial companies: Pearson, Knewton
- Adaptive Learning group:
  www.fi.muni.cz/adaptivelearning/
Course Organization

- 6 weeks
  - lectures: main notions of the field
  - discussions: relations of notions to your projects

- November
  - work on projects
  - individual consultations

- December
  - presentation of projects
Focus of This Course

- practical experience
- collaborative filtering
- educational applications
- evaluation (illustration of methodological issues relevant not just for RecSys)

focus on consultations / discussions (good lectures available online)
Preliminary Schedule – Lectures

- Sep 26: Collaborative filtering
- Oct 3: Other recommendation techniques
- Oct 10: Evaluation
- Oct 17: Educational recommender systems, practical experiences
- Oct 24: Practical aspects; Case studies
Prerequisites

- programming
- math (basic linear algebra, statistics)
- (basics of machine learning – not strictly necessary)

(depends also on the choice of project)
Materials, Sources

- Introduction to Recommender Systems book
  - http://www.recommenderbook.net/
  - slides freely available – more details than in course slides
- Recommender Systems Handbook
  - electronic version available from MU
- Video lectures: Coursera, Machine learning summer school

(links at the course web page)
Projects

2 main options:

- “application”: development of a simple recommender system recommended for AP, INS students
- “research”: implementation and experimental evaluation of algorithms used by recommender systems recommended for UMI students

“hybrids” possible (e.g., extension / analysis of data from your own system)
team project (1-4 students)
goal: build a simple recommender system
requirements: simple web portal implementation (e.g., Python / MySQL / JavaScript)
  note: consultations will be about “recommendation topics”, not about implementation details
Ideas for Simple Recommender System

- “short text” recommendations: jokes, quotes, poetry, baby names, recipes
- travel, “local” recommendations (Brno): restaurants, cultural events, places, holiday locations, tourist attractions, geocaching
- educational recommendations: courses (MU, MOOC), foreign language vocabulary, learning materials
- product recommendation (specialized for a particular domain): board games, beers, specific movie genre
- personalized guides: TV program, museum guide
Typical Steps

- clarification of the purpose (for whom? why?), specific aspects of the domain, hypothetical business model
- getting/generating data
- basic analysis of data
- implementation of a simple web system
- design and implementation of several recommendation techniques
- evaluation
- presentation
Focus of Project

- “simple domains” (e.g., jokes, English vocabulary)
  - several recommendation algorithms (different types)
  - collection of your own data (ratings, feedback), analysis, evaluation

- “complex domains” (e.g., extension of an existing system)
  - analysis of existing data (what can we use for recommendations)
  - “design” of recommendations, formulation of aims, ...
  - evaluation: proposal, first steps
Advice 1

- prefer larger team (3 or 4 students)
- clear division of tasks, responsibilities
- use version control system (GitHub, gitlab.fi.muni.cz, ...)

experience from previous years:

- prefer something rather simple, but done well, focus on recommendation aspects
- ambitious projects often:
  - too much time on technical aspects (getting and cleaning data, implementation infrastructure)
  - little time left for recommendations
“Research”: Models, Evaluation

individual project or group in (mainly) “competitive mode”

- develop a model for predicting user ratings / student performance
- evaluate the model, visualize results
- provided: specific datasets (movies, slepemapy.cz data), guidelines, baseline model implementations (in Python)

requirements: data analysis (Python recommended), implementation of machine learning techniques
Projects from Previous Years

- products: board games, video games, wine, beer, PC parts
- funny quotes, jokes, recipes, blog posts, jobs, anime/manga, geocaching, linux applications
- educational resources, English vocabulary
- analysis of data from existing systems: movies, music, slepemapy, board games, blog system
- implementation of techniques into a real e-shop
Course Deliverables

- source code with basic documentation
- presentation
- individual report (2-3 pages)
  - description of individual contribution to the project
  - connection with course topics
  - discussion of related research papers
standard way:

- interesting project, presentation, report
- active participation during semester (attendance registered)

special cases (poor attendance, weak project, unclear contribution to the project, etc):

- revision of the project
- individual “examination” (discussion) at the end of semester
Discussion

- questions
- your project ideas
- potential groups