HOGESCHOOL ROTTERDAM / CMI

Data science 1
Recommendation systems

INFDTA02-1

Number of study points:	4 ects
Course owner:	G. Costantini

Module description

Module name: Recommendation systems	Data science 1
Module code:	INFDTA02-1
Study points and hours of effort for full-time students:	This module gives 4 ects, in correspondance with 112 hours: • 8 x 3 hours frontal lecture • the rest is individual self-study activities.
Examination:	Written exam and practical assessment
Course structure:	Lectures
Prerequisite knowledge:	Object oriented programming
Learning tools:	• Book: *A Programmer's Guide to Data Mining*, author R. Zacharski (`http://guidetodatamining.com/`) • Lesson slides (pdf): found on N@tschool • Assignments, to be done at home (pdf): found on N@tschool
Connected to competences:	• Realisation • Analysis
Learning objectives:	At the end of the course, the student: • [*R*1] can implement User-Item and Item-Item recommendation systems • [*R*2] can argument choices made in the code (software engineering) • [*A*1] can explain and reflect on the results obtained from the application of the implemented recommendation systems to a specific dataset • [*A*2] can name and describe the various possible recommendation techniques (User-Item, Item-Item, Content-based, Association analysis), including advantages, disadvantages and related issues (ratings types, cold start, sparsity, ...); • [*A*3] can compare the various techniques and decide which one to use, based on the specific circumstances; • [*A*4] can manually apply the above-mentioned recommendation techniques to small sample datasets;
Course owner:	G. Costantini
Date:	5 februari 2018

1 General description

In this module you will learn how to build a recommendation system. A possible definition of recommendation systems is the following:
”A recommender system or a recommendation system (sometimes replacing ”system” with a synonym such as platform or engine) is a subclass of information filtering system that seeks to predict the ”rating” or ”preference” that a user would give to an item.

Recommender systems have become increasingly popular in recent years, and are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general. There are also recommender systems for experts, collaborators, jokes, restaurants, garments, financial services, life insurance, romantic partners (online dating), and Twitter pages.” [1]

Whenever you visit a web-shop (think about Amazon.com, for example), it’s possible that you will be recommended articles sold by such shop. The recommendation can be created through a comparison between the personal preferences and the behaviour of the visitor with those of other visitors. This comparison produces the recommendation of articles which should be suitable for the taste of the visitor. How can web-shops with millions of users and thousands of products achieve this result? In this module we are going to explore different possibilities in the creation of a recommendation system.

1.1 Relationship with other teaching units

This module builds upon the knowledge and skills coming from previous development and math courses (INFDEV03-6A Algorithms, INFDEV03-7 Statistics with R). A good knowledge of OO programming is required.

1.2 Learning materials

Mandatory learning materials are the following ones:

- *A Programmer’s Guide to Data Mining* (`http://guidetodatamining.com/`), R. Zacharski;
- Slides of the lessons (on N@tschool);
- Dataset from `http://grouplens.org/datasets/movielens/`

Recommended learning materials are the following ones:

- *Introduction to Recommender Systems* (`https://www.coursera.org/#course/recsys`), J.A. Konstan and M.D. Ekstrand;
- *Recommender Systems: An Introduction*, D. Jannach et al., 2010), ISBN: 978-0521493369;

[1] `http://en.wikipedia.org/wiki/Recommender_system`

2 Course program

In the following table you can see the program of the course, divided per lesson unit. Each lesson unit is also associated to the corresponding book chapters. Note that lesson units do not necessarily correspond to lesson weeks.

Lesson unit	**Topics**	**Book chapter**
1	• Introduction to the course; • Basic principles of User-Item; • Similarity measures (Pearson, Euclidean, Cosinus, Manhattan);	1, 2
2	• Nearest Neighbour; • Computing a prediction; • Collaborative Filtering issues; • Strategy pattern;	2
3	• Item-Item basic principles and Slope One method;	3
4	• Item-Item ACS (adjusted cosine similarity);	3
5	• Content-based filtering; • Association analysis;	4
6	• Summary and comparison between the different techniques;	
7	• Course summary and exercises in preparation for exam;	

3 Assessment

The course examination is divided in two parts: a practical assessment and a written exam. The final grade is determined for 70% by the practical assessment and for 30% by the written exam. In order to get the study points for this course, you must have a positive grade (≥ 5.5) in **both** parts (the written exam and the practical assessment).

3.1 Written exam

The **written exam** consists of open questions on the topics seen in class. The exam lasts two lesson hours (90 minutes). No help (like slides, books, etc.) is allowed during the exam.

Rules for the exam:

- It is allowed to use a non-programmable calculator.
- Complex formulas will be given.
- Important to have fully understood what the formulas mean and how to use them and how the techniques presented in the course work (with pros & cons, issues).

Formulas that you need to remember (YES) /not remember (NO):

- User-item distances/similarities: Euclidean YES, Manhattan-Pearson-Cosine NO
- User-item computing prediction: YES
- Item-item (slope one) deviations & deviation matrix: YES
- Item-item (slope one) computing prediction: NO
- Item-item ACS: NO
- Normalization: to the range [0-1] YES; to other ranges (i.e. [-1,1]) and de-normalization NO
- Sparsity coefficient: YES
- Association rule metrics (support, confidence, lift): NO

For all formulas (also the ones that you do not need to remember by heart) it holds:

- You must know what the formula means and what we use it for
- Given a dataset with numbers and the formula, you must know how to apply it

3.2 Practical assessment

The practical assessment is based on a **programming assignment**, divided in two parts. *Part 1* is the implementation of a recommendation system using the *User-Item* technique, while *part 2* is the implementation of a recommendation system using the *Item-Item* technique. You must use both systems to create recommendations in the context of two sample datasets (userItem.data and MovieLens100k). The performance of the system is also an important aspect: you will have to measure how much time it costs to produce a recommendation (starting from when the data is read) [2].
On N@tschool you can find detailed instructions for both parts.

The **practical assessment** will require you to be able to re-implement some parts of the assignments. You will not be able to use your own code, but rather you will be given a pre-made incomplete implementation which code you need to fill in.
In order to receive a grade you must successfully submit in N@tschool the assignments completed during the assessment.

Important notes:

[2] With the dataset GroupLens 100K it could take up to 3 seconds to create a recommendation.

- The *programming* for the home-assignment can be done individually or in groups of unlimited size. Note however that the practical assessment is **strictly individual**, so each student must be fully aware of how to implement the assignments completely.
- The home-assignments can be implemented in any language of your preference, but the assessment will require to know C#.
- You must have Visual Studio working on your laptop for the assessment. If you use a machine without Windows, make sure to be able to open (and run) a C# solution made with Visual Studio.
- Every student must deliver authentic materials (not coming from the internet or other sources). In case of doubts, the teacher has the possibility to do a brief oral check to verify the authorship of code: students should be able to explain the content of their code.

3.3 Retake (herkansing)

If the written exam is not sufficient, you can retry it during Week 10 of OP4. If it is sufficient it will be valid until the end of this academic year.
If the practical assessment is not sufficient, you can retry it during Week 10 of OP4. If it is sufficient it will be valid until the end of this academic year.

Attachment 1: Assessment matrix

Competence	Learning goal	Tested through
Realisation	[R1] The student can implement User-Item and Item-Item recommendation systems	Practical assessment
Realisation	[R2] The student can argument choices made in the code (software engineering)	Practical assessment
Analysis	[A1] The student can explain and reflect on the results obtained from the application of the implemented recommendation systems to a specific dataset	Written exam
Analysis	[A2] The student can name and describe the various possible recommendation techniques, including advantages, disadvantages and related issues	Written exam
Analysis	[A3] The student can compare the various techniques and decide which one to use, based on the specific circumstances	Written exam
Analysis	[A4] The student can manually apply the above-mentioned recommendation techniques to small sample datasets	Written exam