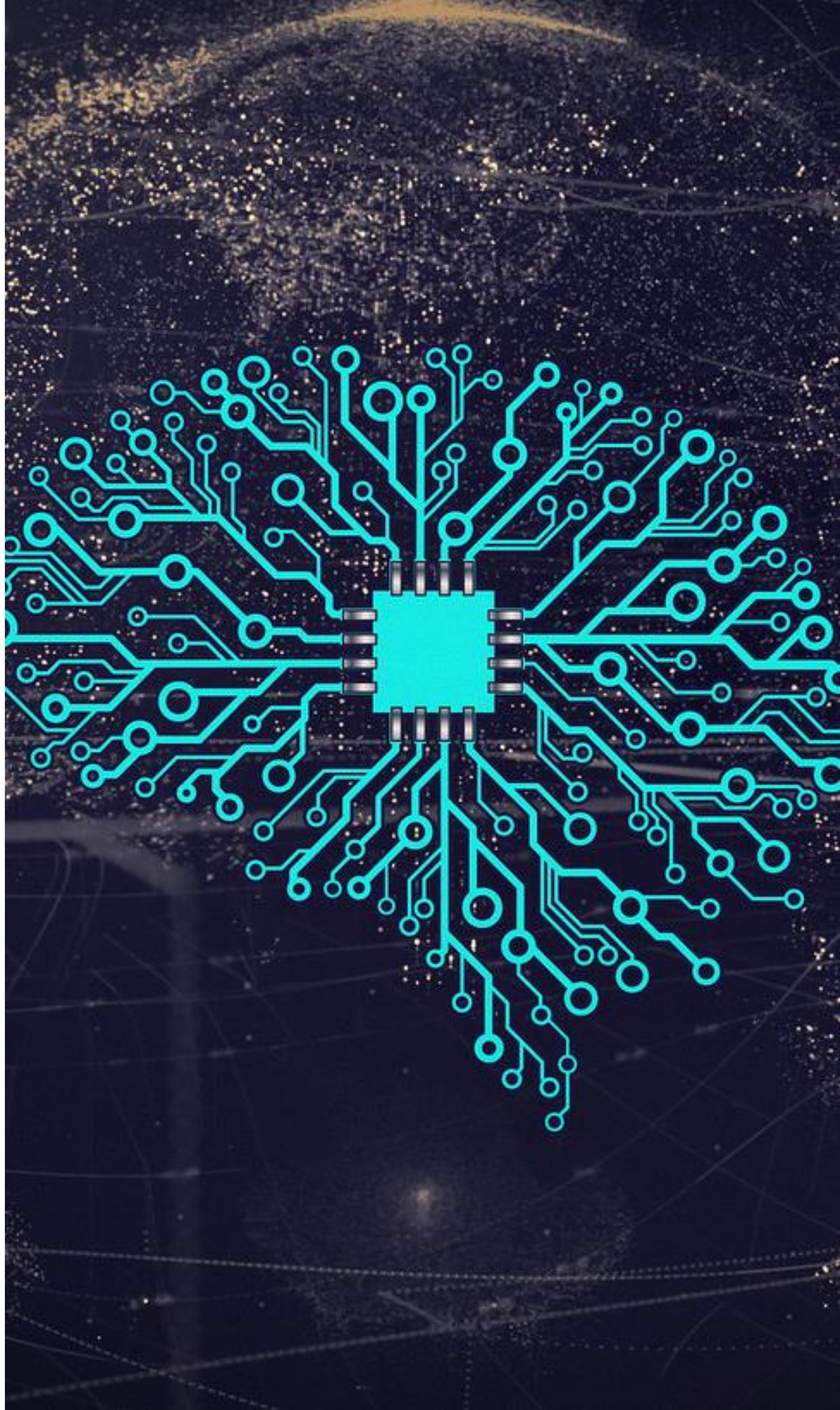
Introduction

Machine Learning with Big Data

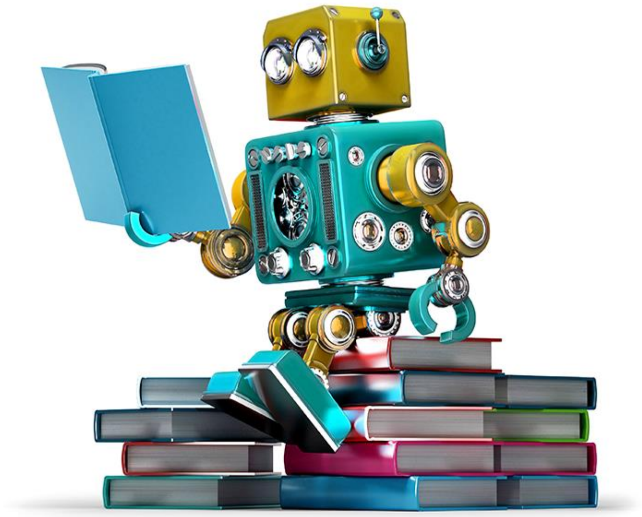
Evert Duipmans
Jeroen Linssen
Etto Salomons

SAXION UNIVERSITY OF APPLIED SCIENCES



Contents

- Evaluation of Q1 of BDT: DBDP & IML
- Organization
- Recommendation systems
  - Content-based recommendations
  - Text learning
  - Cosine similarity



SAXION UNIVERSITY OF APPLIED SCIENCES

Evaluation of Q1 of BDT

Distributed Big Data Processing
tinyurl.com/BDT22dbdp

Introduction to Machine Learning
tinyurl.com/BDT22iml

Organization

Every week one lab session

- Some theory
- Lot of time to work on your exercises
- Rooms reserved in Deventer (X2.03) and Enschede (RB4.03)

Learning materials

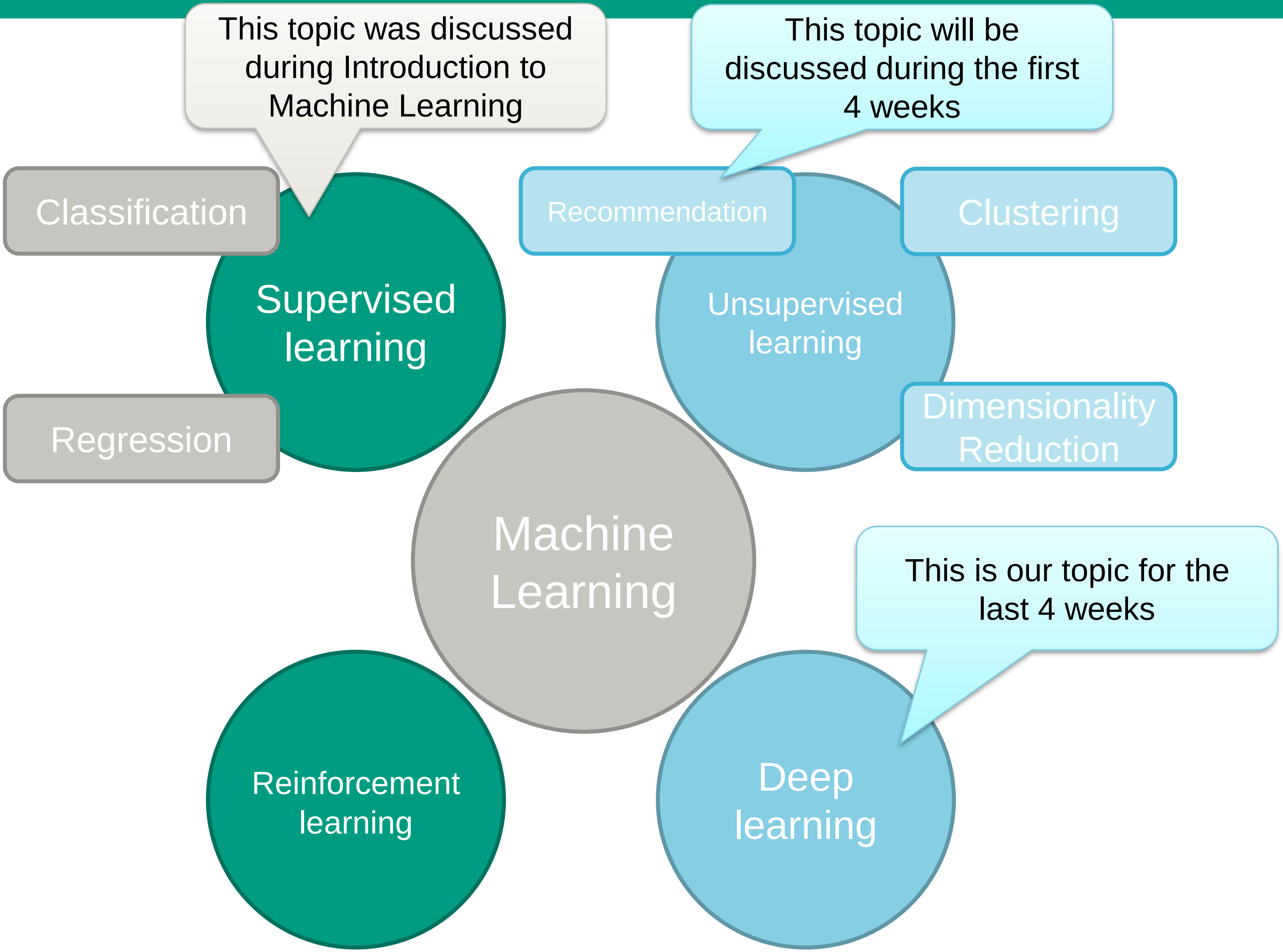
- Slides
- Videos
- Articles
- Jupyter Notebooks

Organization: assignments

1. **Recommender system (4 weeks)**
   Build your own recommendation system based on three different techniques.

2. **Deep learning (4 weeks)**
   - Training neural networks using fully connected layers and convolutional layers
   - Retraining existing neural networks

Grade = avg(assignment1, assignment2) if both >= 5

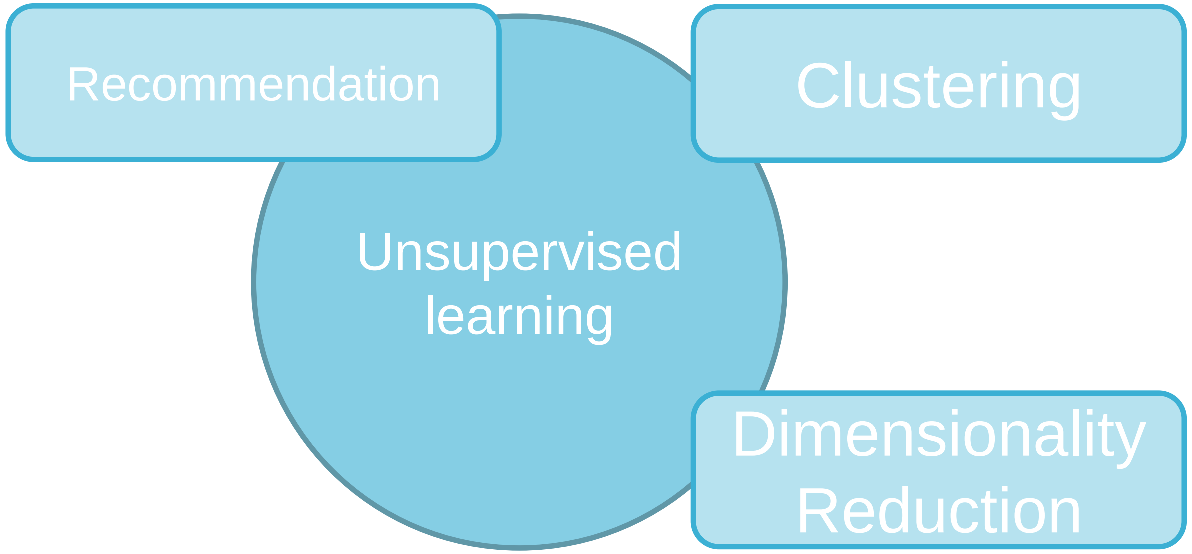

This topic was discussed during Introduction to Machine Learning
This topic will be discussed during the first 4 weeks
Classification
Regression
Supervised learning
Recommendation
Clustering
Unsupervised learning
Dimensionality Reduction
Machine Learning
This is our topic for the last 4 weeks
Reinforcement learning
Deep learning

Unsupervised learning

Trains a model with "unlabeled" data.

Used for:

- Finding patterns in data
- Finding similar users
- Detecting anomalies

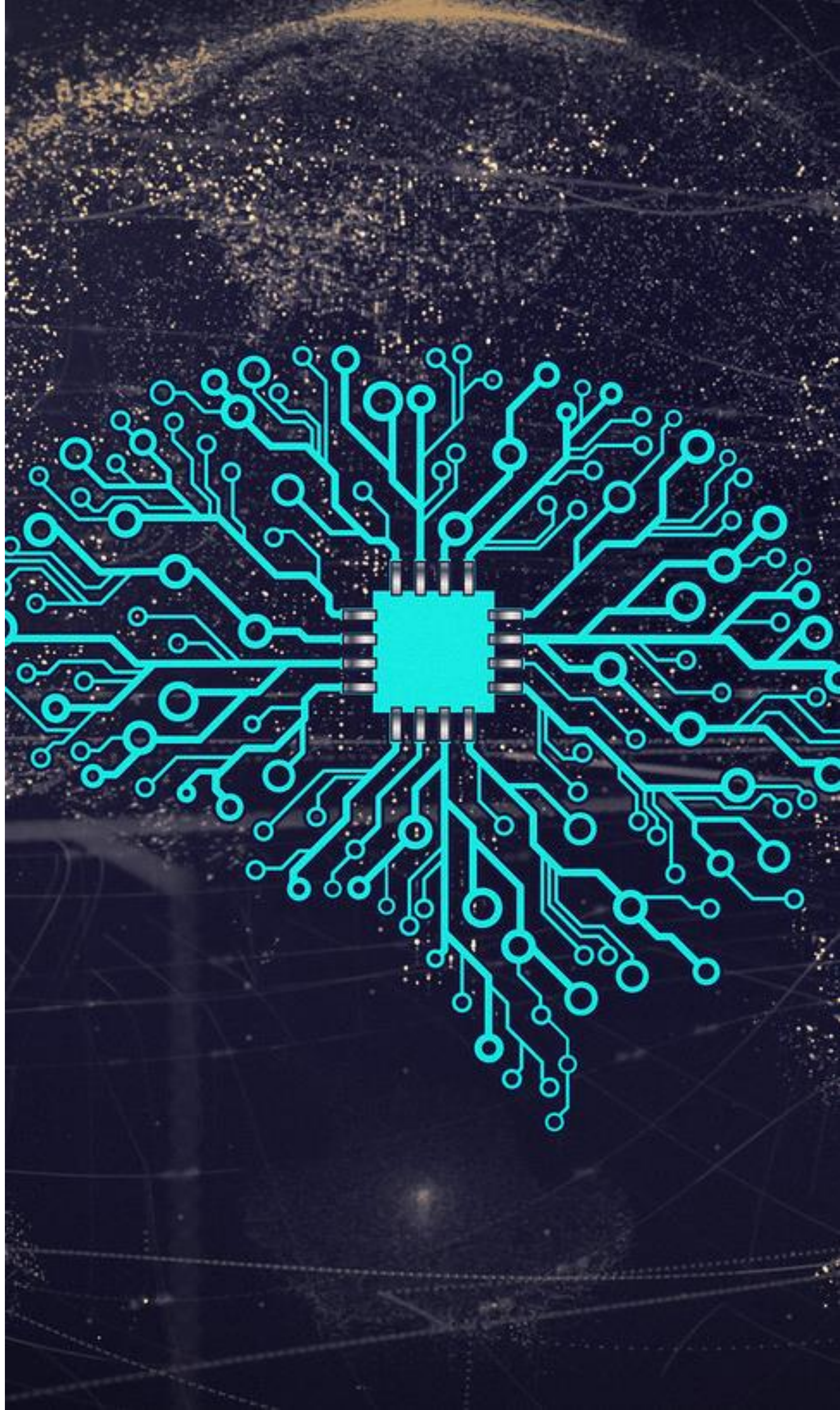


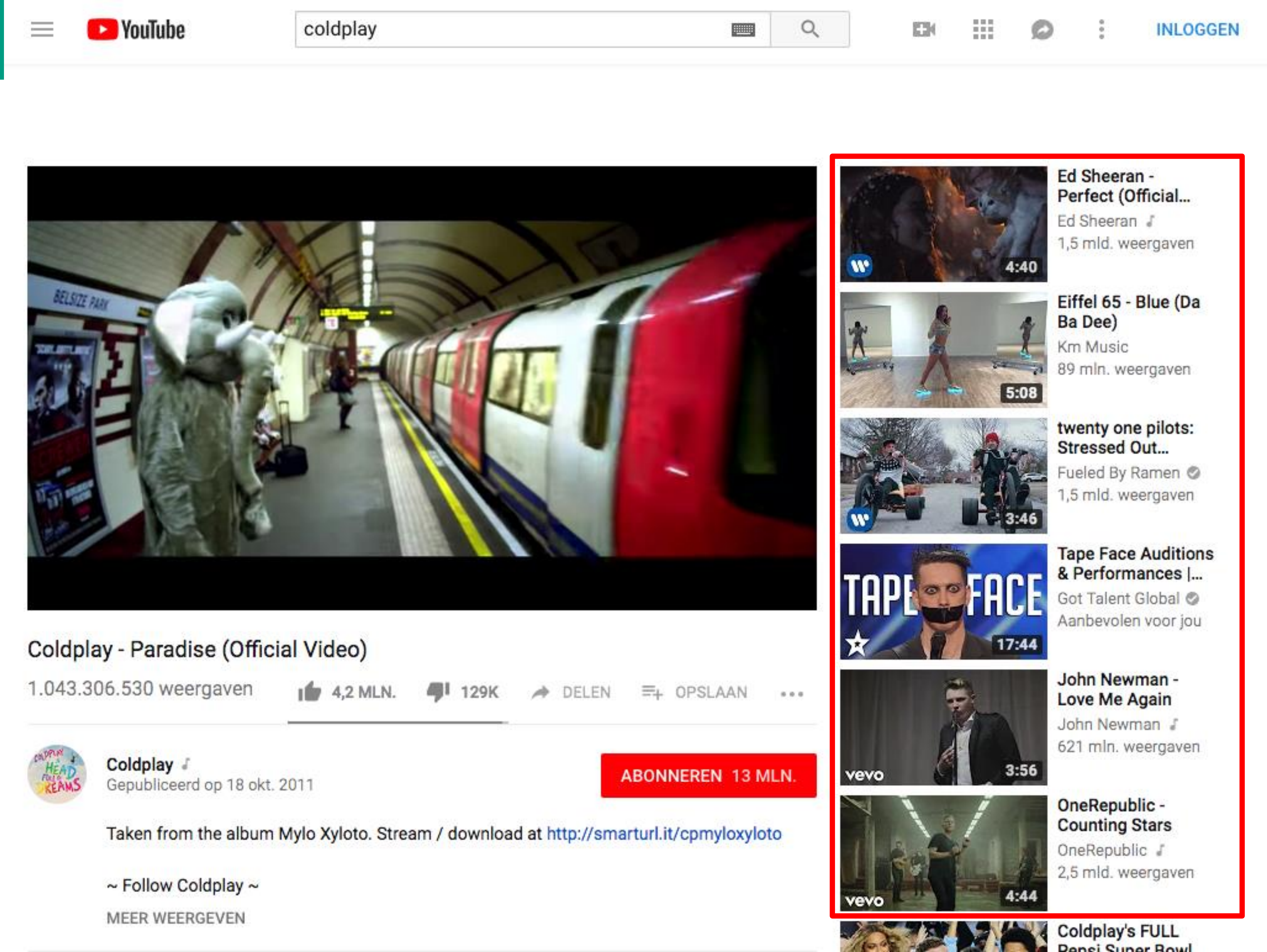
Content-based recommendations

Machine Learning with Big Data

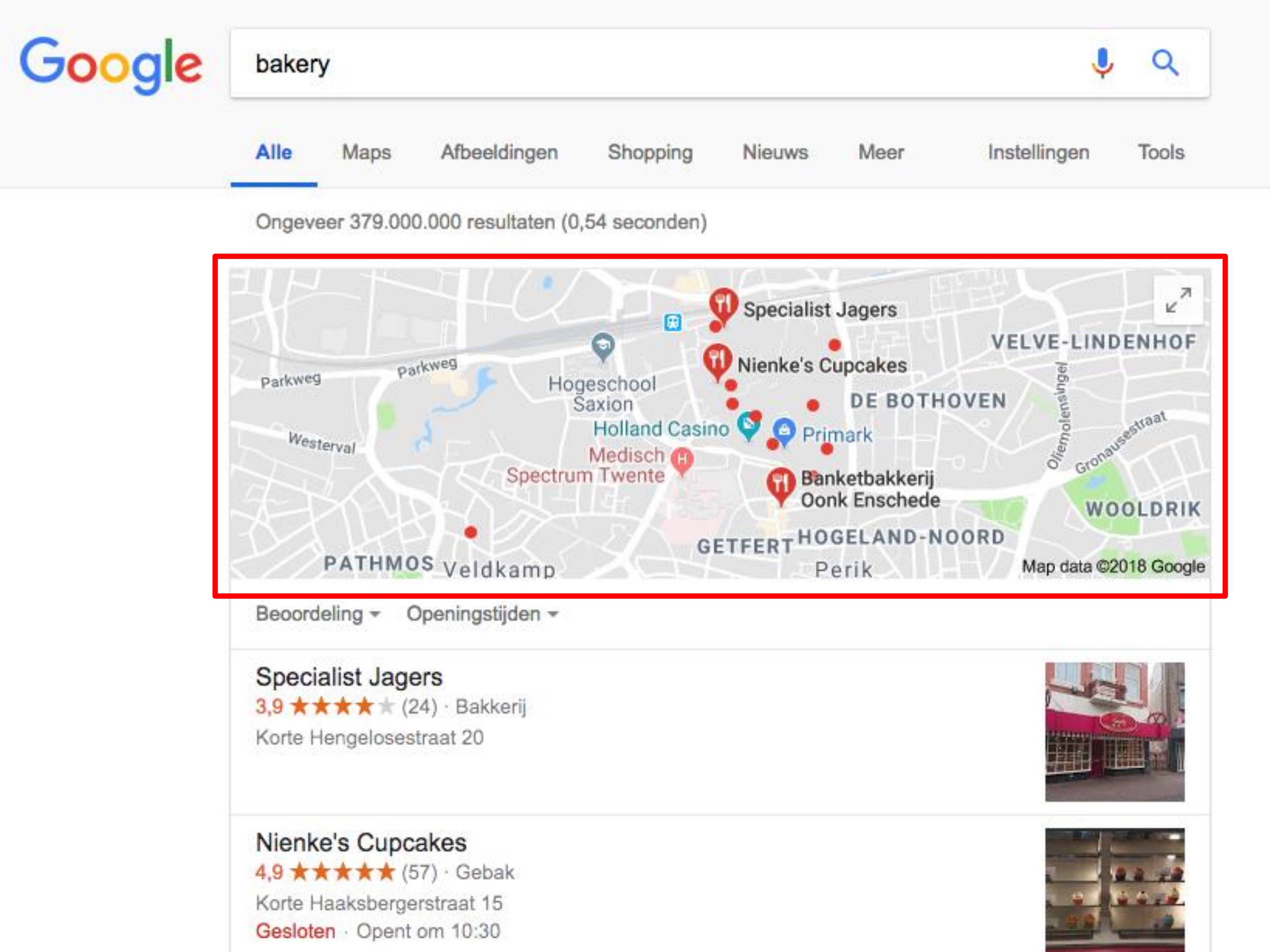
Evert Duipmans
Jeroen Linssen
Etto Salomons

SAXION UNIVERSITY OF APPLIED SCIENCES

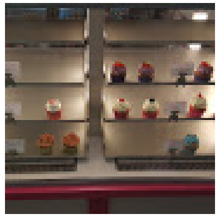




SAXION UNIVERSITY OF APPLIED SCIENCES


Google
bakery
Alle
Maps
Afbeeldingen
Shopping
Nieuws
Meer
Instellingen
Tools
Ongeveer 379.000.000 resultaten (0,54 seconden)
Specialist Jagers
Nienke's Cupcakes
VELVE-LINDENHOF
Parkweg
Parkweg
Hogeschool Saxion
DE BOTHOVEN
Holland Casino
Primark
Westerval
Medisch Spectrum Twente
Banketbakkerij Oonk Enschede
WOOLDRIK
Oliemolensingel
Gronausestraat
PATHMOS
Veldkamp
GETFERT
HOGELAND-NOORD
Perik
Map data ©2018 Google
Beoordeling
Openingstijden
Specialist Jagers
3,9 (24) · Bakkerij
Korte Hengelosestraat 20
Nienke's Cupcakes
4,9 (57) · Gebak
Korte Haaksbergerstraat 15
Gesloten · Opent om 10:30

Nienke's Cupcakes
4,9 ★★★★★ (57) · Gebak
Korte Haaksbergerstraat 15
Gesloten · Opent om 10:30



Banketbakkerij Oonk Enschede
4,3 ★★★★★ (44) · Bakkerij
Kuipersdijk 21

Meer plaatsen

Laura's Bakery - Van bakken tot borrelen
https://www.laurasbakery.nl/
Laura's **Bakery** staat vol met de lekkerste toegankelijke recepten. Van hartig tot zoet en van bakken tot borrelen. Voor ieder wat wils!

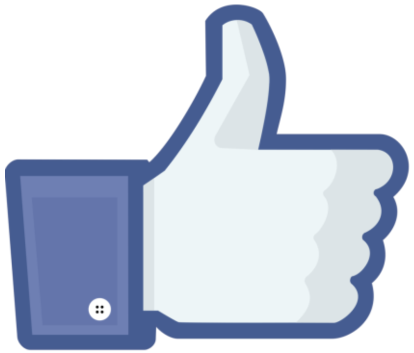
Bakery - Wikipedia
https://en.wikipedia.org/wiki/Bakery Vertaal deze pagina
A **bakery** is an establishment that produces and sells flour-based food baked in an oven such as bread, cookies, cakes, pastries, and pies. Some retail **bakeries** ...
History · Specialities · Commercialization

Appèl: "Met The Bakery ontstijgen we de standaard catering"
www.schoolfacilities.nl/.../3736-appel-met-the-bakery-ontstijgen-we-de-standaard-cat...
19 jan. 2016 - Zeer opvallend aanwezig is bijvoorbeeld The **Bakery**, één van de nieuwe concepten die Appèl bij Saxion heeft geïntroduceerd. De bakkerij ...

Recommender systems
What data is used?

Explicit ratings

- Rate content (stars, like/dislike, …)
- Requires extra work for the user
- Cultural differences
- People rate different
- Data is often sparse



Implicit ratings

- Things you do: click on links, read article, add to cart, buy things, how long did you watch a video?
- Lots of companies use sales data
- Things you consume

Recommender systems
Top-N list

The overall goal of a recommendation systems is to:

Recommend *n* relevant items to the user

Recommender systems
Problems

1. Cold-start problem
    - First-time user
    - Not enough user interactions for a particular item

2. Sparsity problem
    - Users only rated a very small number of items

3. First-rater problem
    - Items that have not been rated will never be recommended

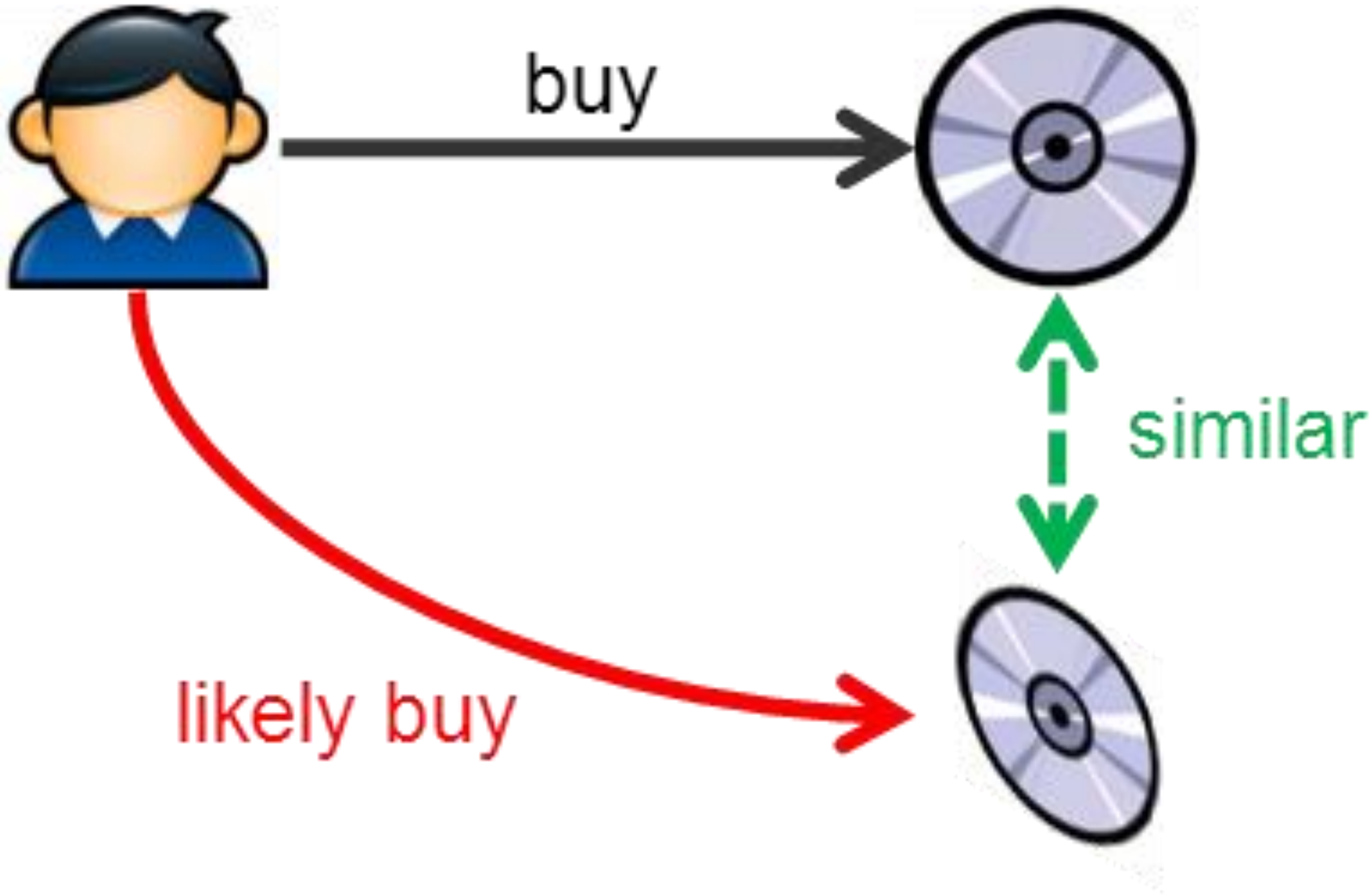
Recommender systems
Approaches

Two possible approaches to building recommender systems:

1. **Content-based (week 1)**
   Recommend items with the same properties

2. **Collaborative filtering (week 2 and 3)**
   Recommend based on ratings of similar users

Content-based recommendations



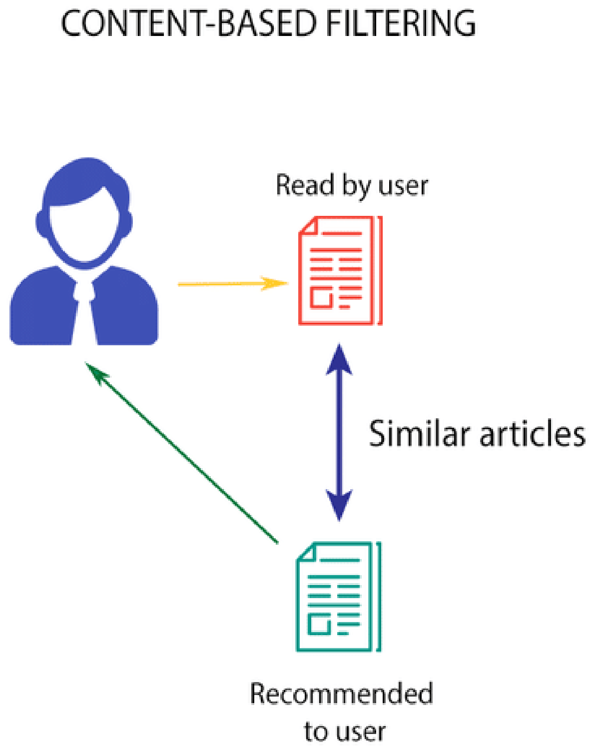
SAXION UNIVERSITY OF APPLIED SCIENCES

Content-based recommendations

General idea:
Recommend items to the user that are similar to items that the user already likes.

Examples

- Recommend books from the same authors or the same genre
- Recommend music from the same artist, collaborations or music with the same BPM



Content-based recommendations
Music example

1. Create an **item profile** for each album in the catalog.
   Each profile consists of features, such as: artists, titles, genre, band members, …

2. Create a **user profile** consisting of the item profiles of the items that have been purchased by the user.

3. Find items that are similar to the items in the user profile.

4. Filter the list and recommend the top-N items.

Content-based recommendations

1. Create item profiles

Extract features

Title	Artist	Members	Genre
Nevermind	Nirvana	Kurt Cobain	Rock, Grunge
Night visions	Imagine Dragons	Dan Reynolds	Rock
Hot fuss	Killers	Brandon Flowers	Rock, Pop
Greatest hits	Nickelback	Chad Kroeger	Awful
Vultures	Kensington	Eloi Youssef	Rock
Overexposed	Maroon 5	Adam Levine	Pop

Content-based recommendations

2. Create user profile

Extract features

Title	Artist	Members	Genre
Flamingo	Brandon Flowers	Brandon Flowers	Rock, Pop

Content-based recommendations

3. Find similar items

Title	Artist	Members	Genre
Flamingo	Brandon Flowers	Brandon Flowers	Rock, Pop

Find similarities

Title	Artist	Members	Genre
Nevermind	Nirvana	Kurt Cobain	Rock, Grunge
Night visions	Imagine dragons	Dan Reynolds	Rock
Hot fuss	Killers	Brandon Flowers	Rock, Pop
Greatest hits	Nickelback	Chad Kroeger	Awful
Vultures	Kensington	Eloi Youssef	Rock
Overexposed	Maroon 5	Adam Levine	Pop

Recommend

Title	Artist	Members	Genre
Hot fuss	Killers	Brandon Flowers	Rock, Pop
Night visions	Imagine dragons	Dan Reynolds	Rock
Vultures	Kensington	Eloi Youssef	Rock

Content-based recommendations
Pros

1. No need for data on other users
   No cold-start or sparsity problems

2. Possible to recommend new items
   No first-rater problem

3. Insight in the recommendation
   You can understand why the item has been recommended by listing its features

Content-based recommendations
Cons

1. Finding the best features to represent items is hard

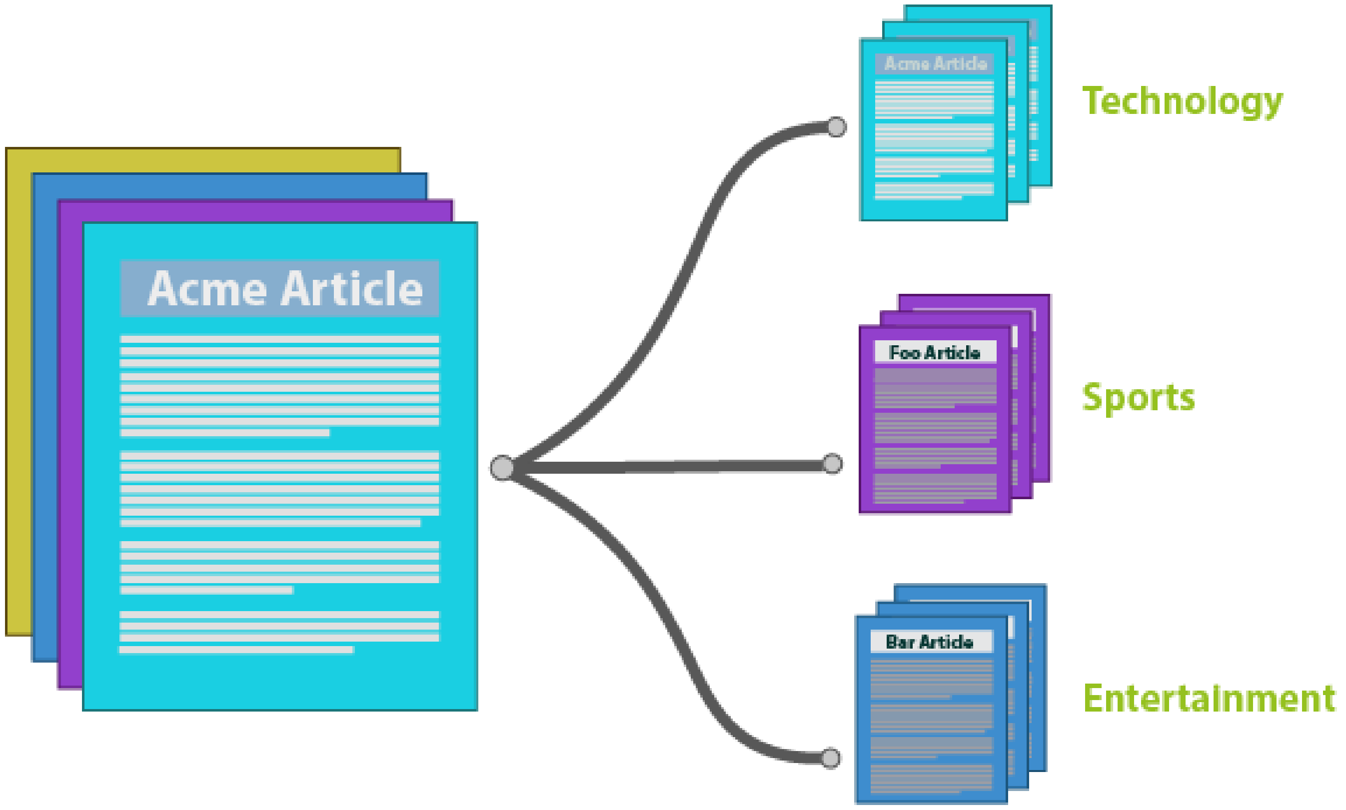
2. How to recommend to new users?

3. Overspecialization
   You will never get a recommendation of items outside of your user profile

Content-based recommendations

1. How to extract features from CDs, books, films, …?
2. How to calculate similarity between items?
3. How to validate your recommendations? (part of lecture 3)

Text learning



	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	1	1	1							
Doc 2	1		1	1	1	1				
Doc 3					1	1	1	2	1	1

Bag-of-words model

1. Get all words from a text
2. Insert words in dictionary (hash map)
3. Create a sample for each sentence
4. Each element in the vector represents a word (index from the dictionary)
5. For each word, the number of occurences is stored in the vector

	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	1	1	1							
Doc 2	1		1	1	1	1				
Doc 3					1	1	1	2	1	1

Bag-of-words model
Training

John likes to watch movies. Mary likes movies too.

John also likes to watch football games.

Create Feature vectors

[1, 2, 1, 1, 2, 0, 0, 0, 1, 1]

[1, 1, 1, 1, 0, 1, 1, 1, 0, 0]

The word “likes” occurs once in the 2nd sentence

Feature vectors

Create Dictionary

Word	Position
john	0
likes	1
to	2
watch	3
movies	4
also	5
football	6
games	7
mary	8
too	9

Dictionary

Bag-of-words model
Questions

- Does the word order matter?
- Do long phrases give different input vectors?
- Can we handle complex phrases?

Bag-of-words model
Scikit learn

Bag-of-words model a.k.a. CountVectorizer
(from `sklearn.feature_extraction.text`)

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
string1 = "hi Jan can you send me the presentation of week six sincerely Evert"
string2 = "hi Evert please find attached the presentation file best Jan"

# Add to list
email_list = [string1, string2]

# Create the dictionary
vectorizer.fit(email_list)
# Create feature vectors
bag_of_words = vectorizer.transform(email_list)

# Tuple: (document number, word number) => number of occurences
print(bag_of_words)

# Print the feature number
print(vectorizer.vocabulary_.get("please"))
```

NumPy
Sparse vs dense

A sparse matrix is a matrix that contains mainly the value 0

For text learning (and working with ratings), it is often useful to choose a different matrix representation

NumPy has support for sparse matrices (csr_matrix)

Most operations can be performed on normal matrices and sparse matrices

Read the following article for more information:
https://machinelearningmastery.com/sparse-matrices-for-machine-learning/

Bag-of-words model
How to deal with...

Not all words are equal
What to do with “low-information” words?

Stop words = low information word that occurs frequently
Examples: and, the, in, for you, will, have, be

Stop words

Use NLTK (Natural Language ToolKit) package

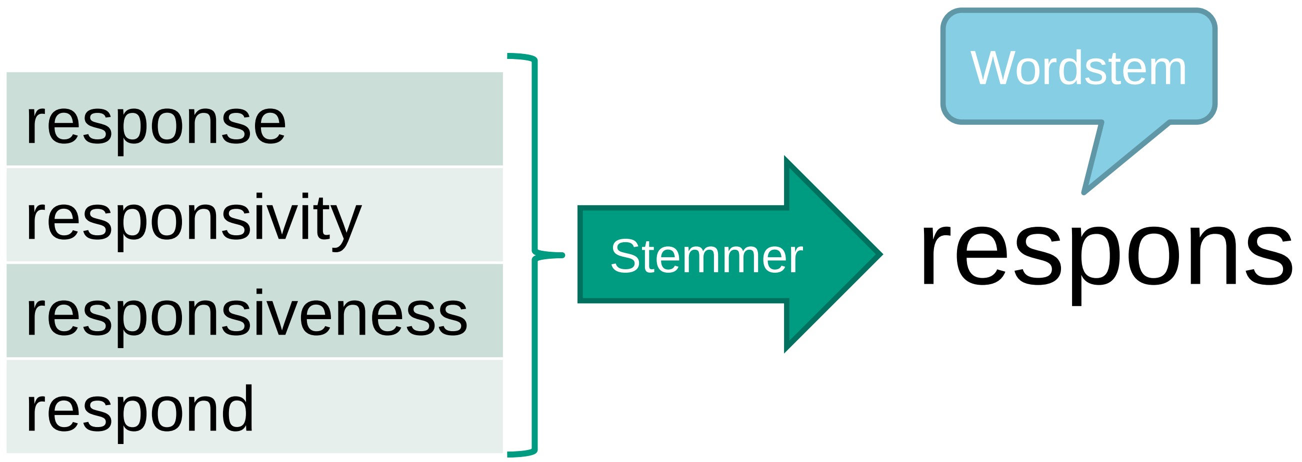
Can be used to remove stopwords from sentences

```
from nltk.corpus import stopwords
sw = stopwords.words("english")
print(sw)
```

i
me
my
myself
we
our
ours
ourselves
you
you're
you've
you'll
you'd
your
yours
yourself
yourselves
he
him
his
himself
she
she's
her
hers
herself
it
it's
its
itself
they
them
their
theirs
themselves
what
which
who
whom
this
that
that'll
these
those
am
is
are
was
were
be
been
being
have
has
had
having
do
does
did
doing
a
an
the
and
but
if
or
because
as
until
while
of
at
by
for
with
about
against
between
into
through
during
before
after
above
below
to
from
up
down
in
out
on
off
over
under
again
further
then
once
here
there
when
where
why
how
all
any
both
each
few
more
most
other
some
such
no
nor
not
only
own
same
so
than
too
very
s
t
can
will
just
don
don't
should
should've
now
d
ll
m
o
re
ve
y
ain
aren
aren't
couldn
couldn't
didn
didn't
doesn
doesn't
hadn
hadn't
hasn
hasn't
haven
haven't
isn
isn't
ma
mightn
mightn't
mustn
mustn't
needn
needn't
shan
shan't
shouldn
shouldn't
wasn
wasn't
weren
weren't
won
won't
wouldn
wouldn't

Stemming

Not all unique words are different... Maybe we can reduce the number of possibilities?



Words with (almost) the same meaning

Stemming
Python example

Use NLTK (Natural Language ToolKit) package

```
from nltk.stem.snowball import SnowballStemmer
stemmer = SnowballStemmer("english")
print(stemmer.stem("responsiveness"))  # respons
print(stemmer.stem("unresponsive"))    # unrespons
```

tf-idf representation

Some words in a document are more important than others

tf-idf exposes this information

Bag of words

	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	1	1	1							
Doc 2	1		1	1	1	1				
Doc 3					1	1	1	2	1	1

tf-idf

	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	0.18	**0.48**	0.18							
Doc 2	0.18		0.18	**0.48**	0.18	0.18				
Doc 3					0.18	0.18	**0.48**	**0.95**	**0.48**	**0.48**

tf-idf representation

Reflects how important a word is to a document in a collection or corpus

tf = Term frequency
Like bag-of-words, number of occurences of a word

idf = inverse document frequency
Weighting by how often a word occurs in the corpus

$tf{-}idf(t) = tf(t) \times idf(t)$
t is a given word

Idea: rare words score higher

tf-idf representation

tf = term frequency
Like bag-of-words, number of occurences of a word

$$tf(t) = \frac{number\ of\ times\ term\ t\ appears\ in\ a\ document}{total\ number\ of\ terms\ in\ the\ document}$$

John also likes to watch football games.

$$tf('also') = \frac{1}{7}$$

tf-idf representation

idf = inverse document frequency
Weighting by how often a word occurs in the corpus

$$idf(t) = \log\left(\frac{total\ number\ of\ documents}{number\ of\ documents\ containing\ t}\right)$$

John likes to watch movies. Mary likes movies too.

John also likes to watch football games.

$$idf('also') = \log\left(\frac{2}{1}\right) = 0.69$$

tf-idf representation

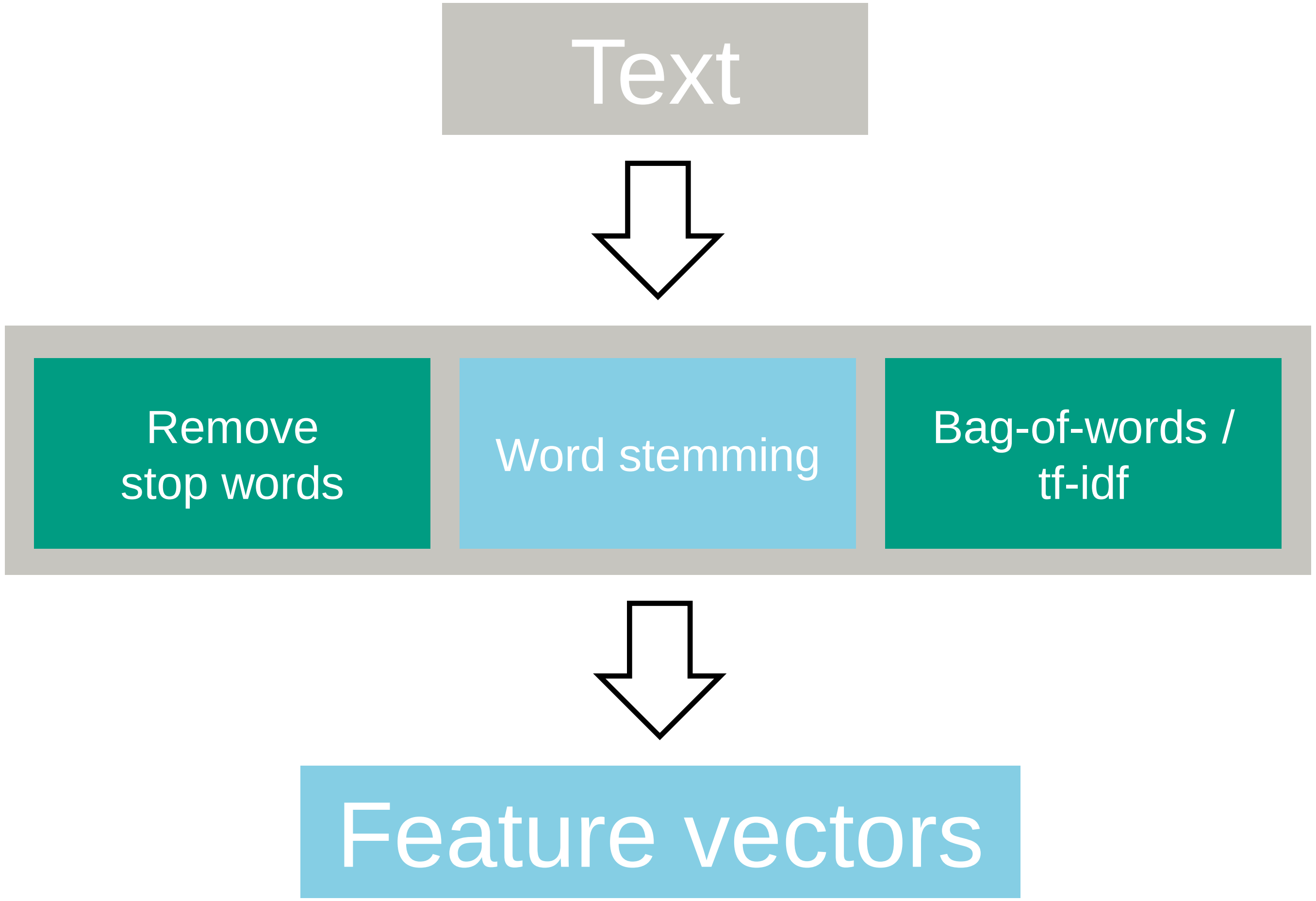
$tf\text{-}idf(t) = tf(t) \times idf(t)$
t is a given word

$$tf('also') = \frac{1}{7}$$

$$idf('also') = \log(\frac{2}{1}) = 0.69$$

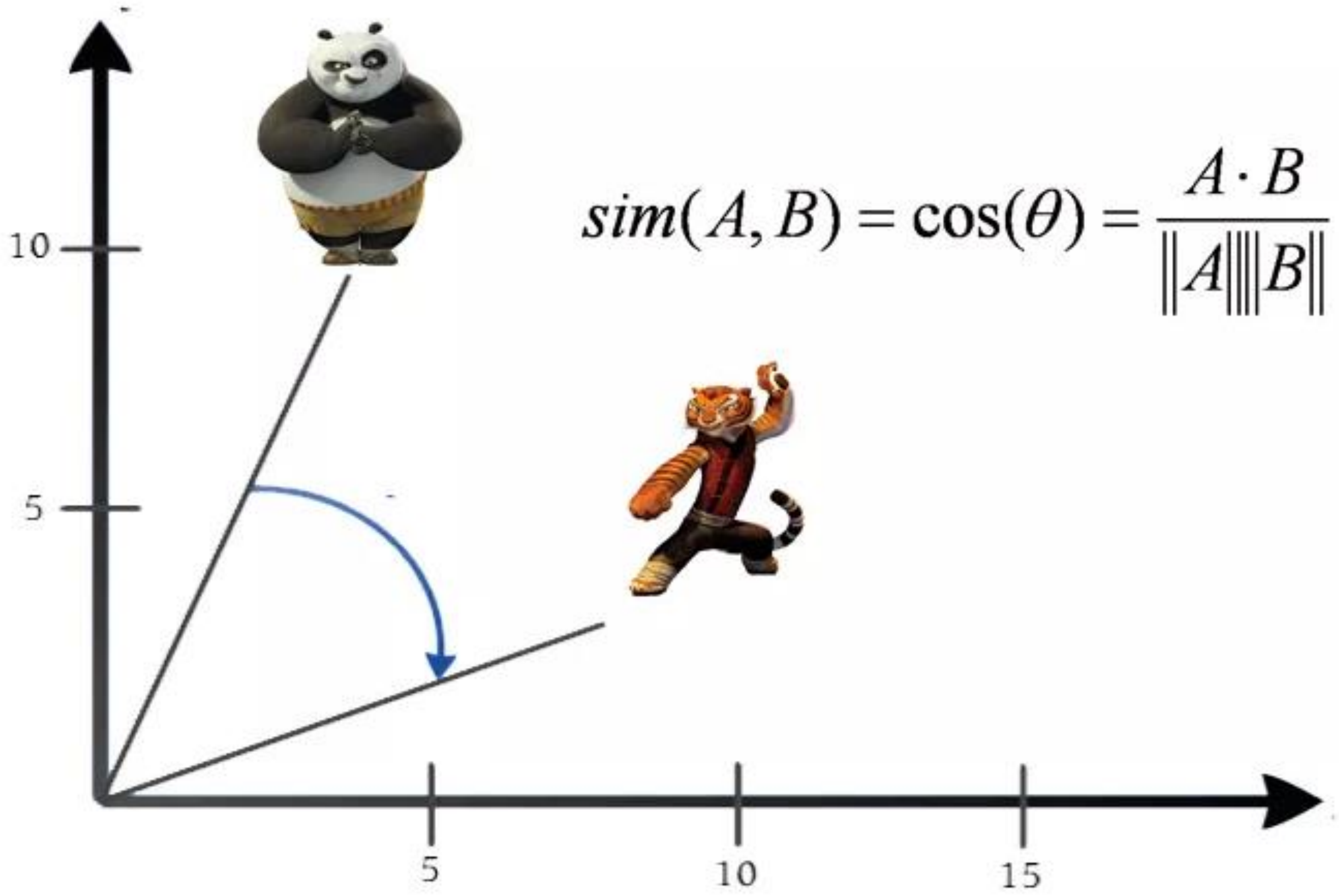
$$tf\text{-}idf('also') = \frac{1}{7} \times 0.69 = 0.0986$$

Text learning summarized



SAXION UNIVERSITY OF APPLIED SCIENCES

Calculating similarities



$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

Cosine similarity

The metric that is used most often in content-based recommendation is cosine similarity.

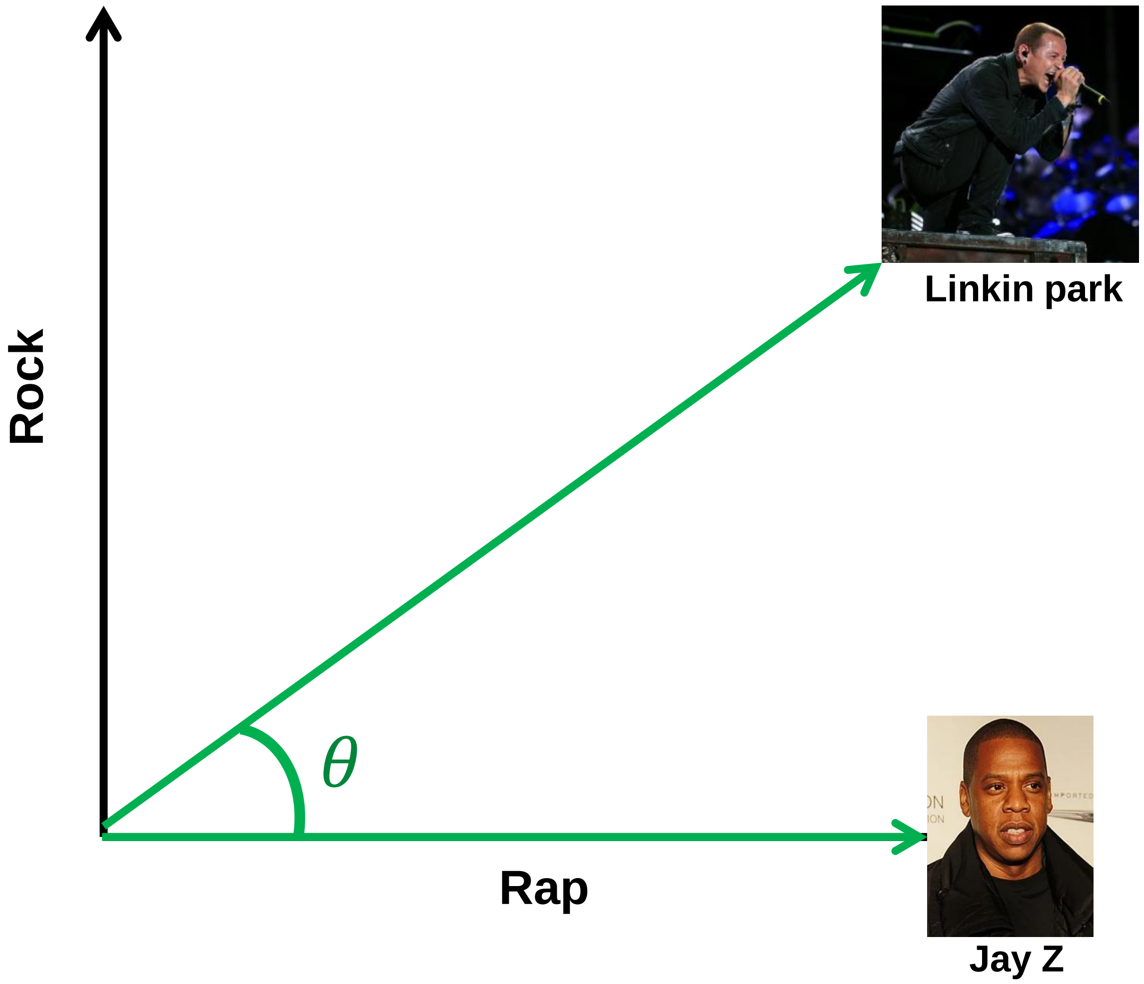
In short: calculate the angle between vectors. Small angles are similar items.

Artist	Genre
AC/DC	Rock
Jay Z	Rap
Linkin Park	Rock, Rap
50 Cent	Rap

Extract features

Artist	Rock	Rap
AC/DC	1	0
Jay Z	0	1
Linkin Park	1	1
50 Cent	0	1

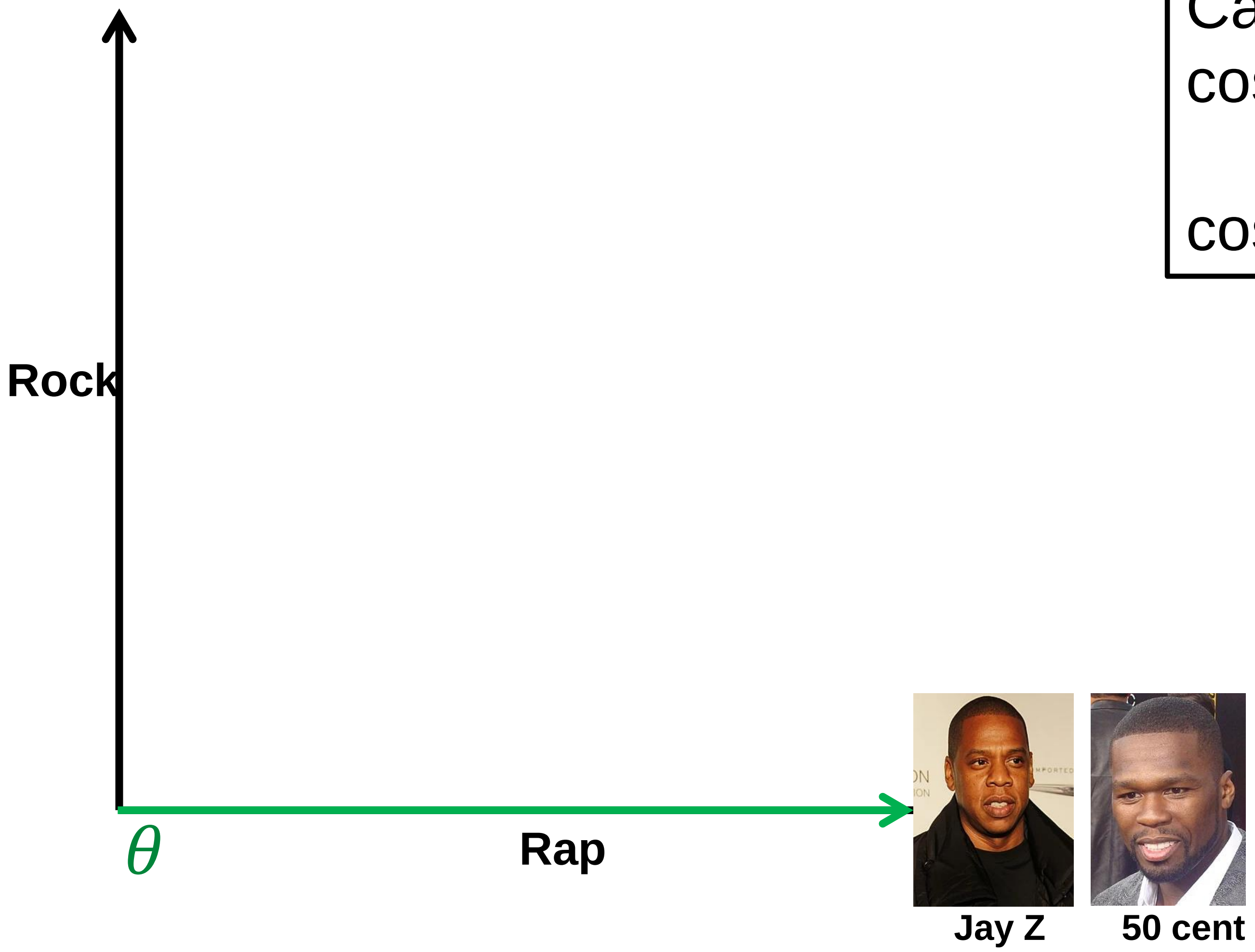
Cosine similarity



Calculating the cosine of angle θ=45°

cos(45) = 0.7071…

Cosine similarity



Calculating the cosine of angle θ=0°

cos(0) = 1

Cosine similarity
Formula

Cosine similarity = 1: items are similar

Cosine similarity = 0: items are not similar at all

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

Example similarity(Linkin Park, Jay Z)
Let vector A be an item with values [1,1]
Let vector B be an item with values [1,0]

$$similarity(A,B) = \frac{(1*1)+(1*0)}{\sqrt{1^2+1^2}*\sqrt{1^2+0^2}} = \frac{1}{\sqrt{2}*\sqrt{1}} = 0.7071$$

Cosine similarity
Formula

Example similarity(Jay Z, 50 cent)
Let vector A be an item with values [1,0]
Let vector B be an item with values [1,0]

$$similarity(A,B) = \frac{(1*1)+(0*0)}{\sqrt{1^2+0^2}*\sqrt{1^2+0^2}} = \frac{1}{1*1} = 1$$

Example similarity(AC-DC, Jay Z)
Let vector A be an item with values [0,1]
Let vector B be an item with values [1,0]

$$similarity(A,B) = \frac{(0*1)+(1*0)}{\sqrt{0^2+1^2}*\sqrt{1^2+0^2}} = \frac{0}{1*1} = 0$$

Calculating similarities

SAXION UNIVERSITY OF APPLIED SCIENCES

Finding the k-best matches?

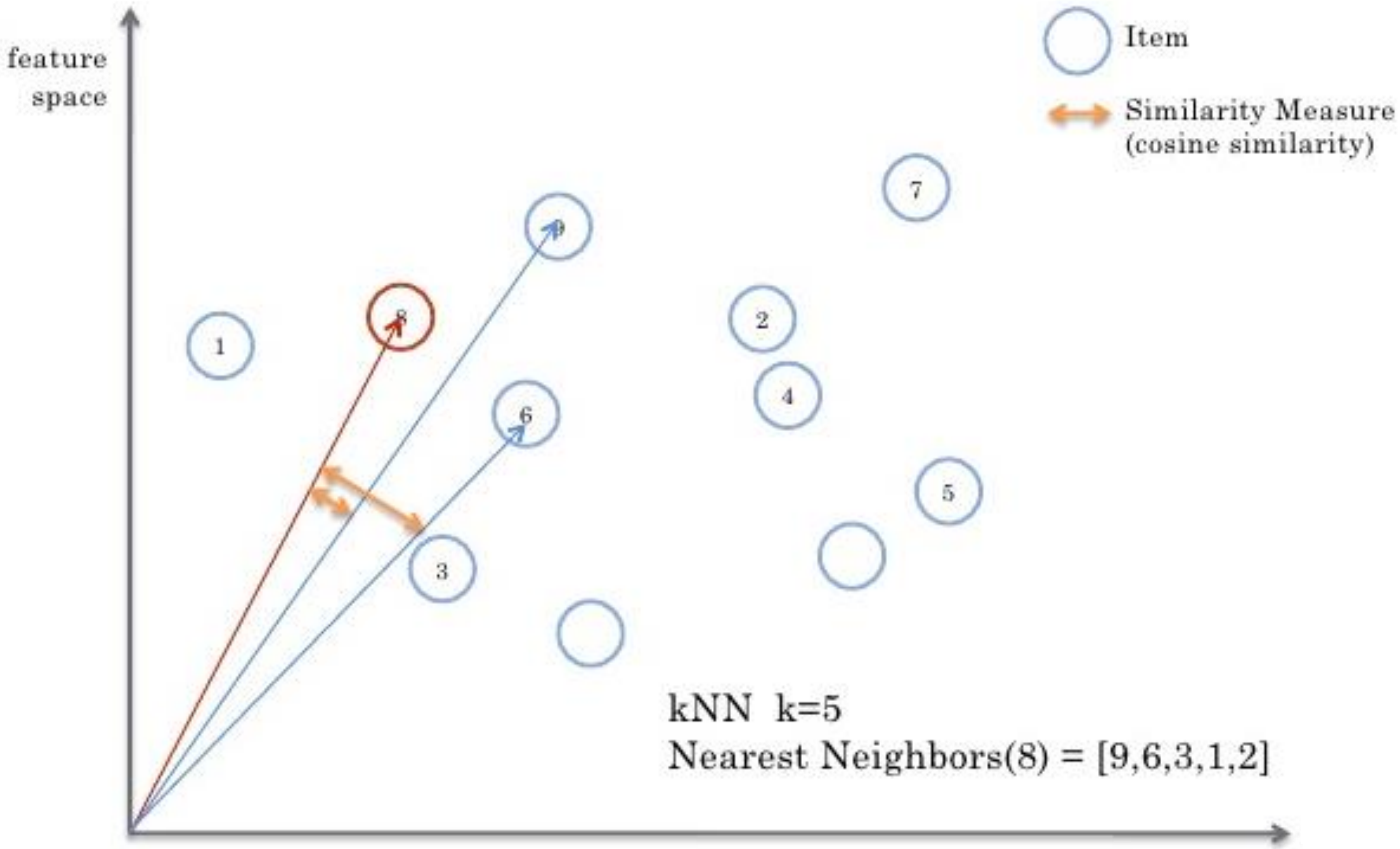
Sounds like a machine learning algorithm that we have already seen…

K-nearest neighbours

We **DON'T** want the classifier variant!

Use cosine similarity as metric (instead of eucledian distance)

Finding the k-best matches?



Questions

Assignments

Enroll in a group!

Assignment 1: recommender systems
Many websites give users the possibility to rate items nowadays. Companies such as Amazon, Netflix, YouTube, IMDB and Bol.com use this information to recommend similar items to their users. The MovieLens dataset is a free dataset with a collection of movie ratings.

In this assignment you will build two recommendation systems, using the following techniques: content-based and collaborative filtering.

Pro-tip: finish this assignment in week 4

References

- http://datameetsmedia.com/bag-of-words-tf-idf-explained/
- http://dataaspirant.com/2015/04/11/five-most-popular-similarity-measures-implementation-in-python
- ACM Building Recommender Systems with Machine Learning and AI
- Mining Massive Datasets (mmds.org)
- Udacity Intro to Machine Learning