Course 8

Vector models
Similarity-based methods
The Local Rank Distance

In the P4 & P5 problems …

- Our questions are something like this:
  - In P4: is this text segment a title, an author or a body of text?
  - In P5: to which letter does this shape resemble?
- Both are expressed as **classification** problems
- Let’s see in this course if we can come closer to solve these problems...

This is what we want to do…

- **Retrieve** objects (from a collection)
  - based on a query
- **Classify** objects (of a collection)
  - on a number of known categories
- **Cluster** objects (of a collection)
  - organize objects in groups, which are similar

Please answer the questions…

- How can Boolean retrieval be applied to our problems?
- How does tf-idf work here?

Boolean retrieval applied to P5

- "Let's consider that an object (document) is characterized by a set of *l* Boolean parameters (terms), and s_i, the parameter i ($1 <= i <= l$), has the value 1 if the object has that property and 0 otherwise."
- ⇔ A shape is characterized by a set of *l* Boolean parameters (features), and s_i, the parameter i ($1 <= i <= l$), has the value 1 if the object has that property and 0 otherwise.

Examples of Boolean features

- rectangular area of the shape (after normalization against the greatest shape area)
  - discretize in 4 values: [0, 0.25), [0.25, 0.5), [0.5, 0.75),[0.75, 1]
  - if the area belongs to the interval [0, 0.25) then A0.25=1, else A0.25=0
  - constraints: exactly one feature among A0.25, A0.5, A0.75, A1 equals 1, the rest being 0

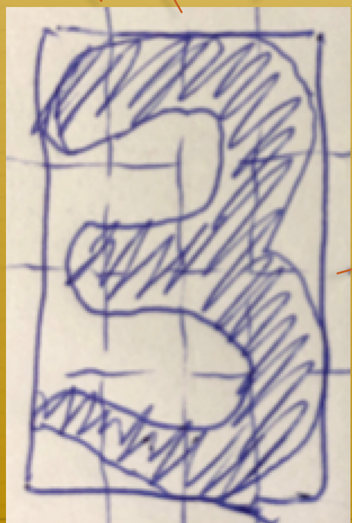
Examples of Boolean features

- # intersections of 3-lines equidistant horizontal&vertical grid with the shape:
  - constraint: exactly one feature of the three equals 1, the rest being 0

3V0.25=1; 2V0.25=0; 1V0.25=0

3V0.5=1; 2V0.5=0; 1V0.5=0

3V0.75=0; 2V0.75=0; 1V0.75=1



3H0.25=0; 2H0.25=0; 1H0.25=1

3H0.5=0; 2H0.5=0; 1H0.5=1

3H0.75=0; 2H0.75=0; 1H0.75=1

TF-IDF applied to P5

- Modify somehow the problem: instead of retrieving documents that fit the query, retrieve shapes that resemble one given letter
- For each feature *f* and shape *s*, compute: $\text{tf-idf}_{f,s} = \text{tf}_{f,s} * \text{idf}_f$
- But what are in our case $\text{tf}_{f,s}$ and idf_f?

Similarity-based Learning

- Learning based on pairwise similarities between the training samples
- SbL processes can be:
  - **supervised**: estimate the class label of a test sample using both the pairwise similarities between the labeled training samples, and the similarities between the test sample and the set of training samples
  - **unsupervised**: find some hidden structure in the unlabeled training samples, using pairwise similarities between samples
- The pairwise relationship can be: a similarity, a dissimilarity, or a distance function
  - advantage of SbL: does not require direct access to the features, as long as the similarity function is well defined and can be computed for any pair of samples
  - feature space is not required to be an Euclidean space

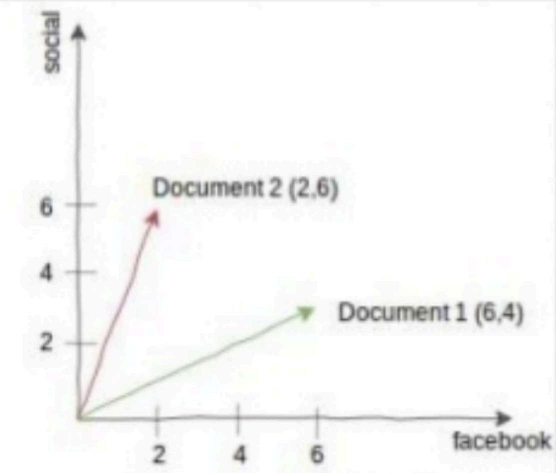
SbL methods

- Used in:
  - **computer vision**: computing similarity between images for object recognition and image retrieval (measuring distances between shapes)
  - **computational biology**: obtain phylogenetic trees, compare DNA sequences (distance measures for strings: Hamming distance, edit distance, rank distance, etc.)
  - **natural language processing**: information retrieval, text mining for document classification, authorship and native language identication or Arabic dialect identification
    - cosine similarity between TF-IDF vectors, string kernels (similarity between strings by counting common character n-grams)

The Vector Space Model

- The representation of a set of objects as vectors in a common vector space is known as the **vector space model**.
  - dimensions are features (words, similarities between a sample and training samples, TF-IDF, etc.)
  - queries are also vectors

	doc1	doc2
facebook	6	2
social	3	6



From Suzan Verberne: Word2Vec Tutorial, Leiden Univ., March 2018

Dot product based similarity

- A vector $\vec{V}(d)$ – derived from the object *d*, with one component for each feature
  - the value of a component being a number, or the *tf-idf* weighting score, or anything else
- How do we quantify the similarity between two objects in this vector space?
  - 1st attempt: compute the magnitude of the vector difference between the vectors of the two objects… critics!
  - A better solution: compute the *cosine similarity*:

$$\mathrm{sim}(d_1, d_2) = \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{|\vec{V}(d_1)||\vec{V}(d_2)|}$$

Properties of the inner (dot) product

- $\langle \cdot, \cdot \rangle$: $V \times V \rightarrow$ **R** with the properties:
  - symmetry: $\langle x, y \rangle = \langle y, x \rangle$
  - linearity: $\langle ax, y \rangle = a\langle y, x \rangle$; $\langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$
  - positive definiteness: $\langle x, x \rangle \geq 0$; $\langle x, x \rangle = 0 \Leftrightarrow x = 0$

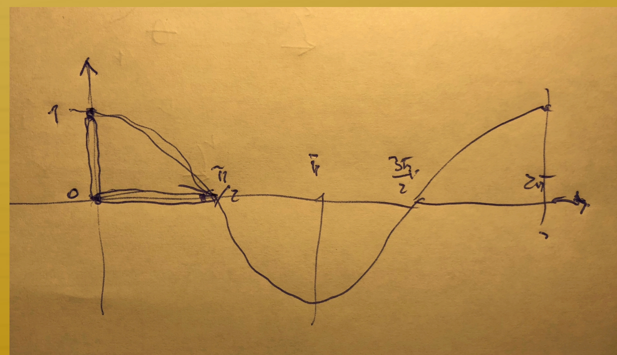
$$\left\langle \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\rangle := x^{\mathsf{T}} y = \sum_{i=1}^{n} x_i y_i = x_1 y_1 + \cdots + x_n y_n,$$

where x^T is the transpose of x.

Computing similarity

- Dot product: $\sum_{i=1}^{M} x_i y_i$
- Euclidean length: $\sqrt{\sum_{i=1}^{M} \vec{V}_i^2(d)}$
- Unit vectors: $\vec{v}(d_1) = \vec{V}(d_1)/|\vec{V}(d_1)|$ and $\vec{v}(d_2) = \vec{V}(d_2)/|\vec{V}(d_2)|$
- Similarity: $\text{sim}(d_1, d_2) = \vec{v}(d_1) \cdot \vec{v}(d_2)$
- Similarity = cosine of the angle between the two objects

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



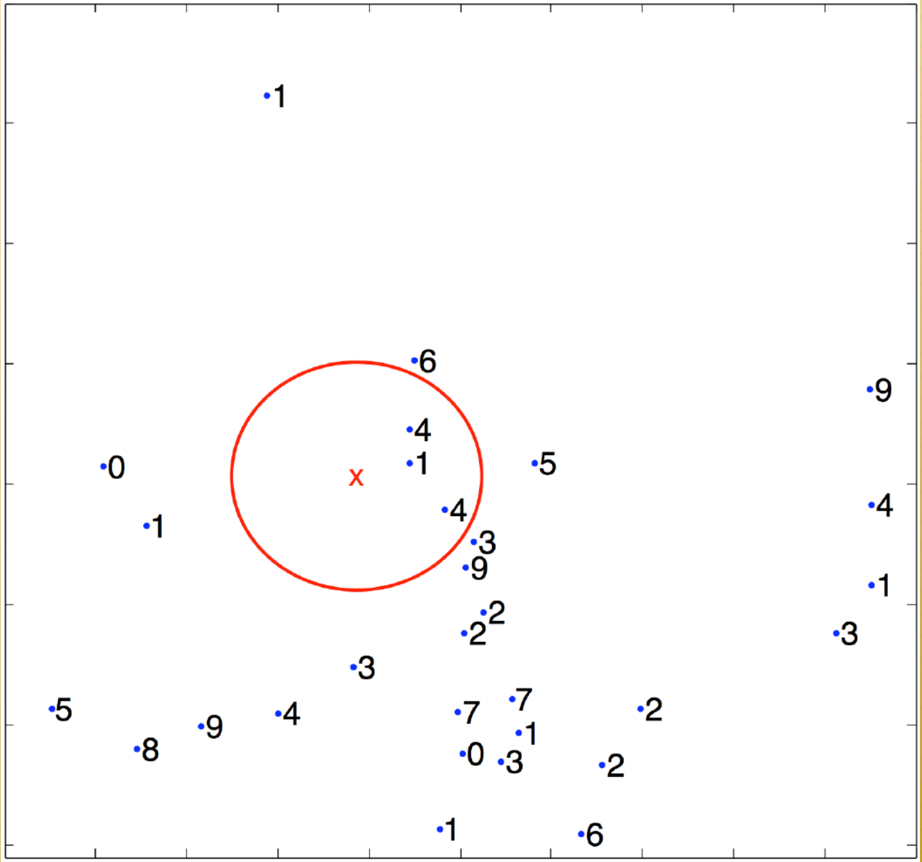
How to use the similarity measure?

- Given an object *d*, find those in the collection *C* most similar with it.
  - similarity, being a cosine, can be made to be within [0, 1]
  - define a threshold *t* in the upper part of [0, 1], or decide *N* = number of retained most similar objects with *d*
    - retain all $d_x \in C$ such as sim(d_x, *d*) ≥ *t*, or:
    - rank all objects d_x in C in the descending order of sim(d_x, *d*) and retain the first *N* objects
  - => this is a retrieval problem
  - => exercise: how would you use the similarity measure to solve a classification and a clustering problem

Nearest Neighbor (*k*-NN) algorithm

Input:
$S = \{(x_i, t_i) \mid x_i \in \mathbf{R}^m, t_i \in \mathbf{N}, i \in \{1, 2, \ldots n\}$ - the set of n training samples and labels;
$Z = \{z_i \mid z_i \in \mathbf{R}^m, i \in \{1, 2, \ldots l\}$ - the set of l test samples;
k - the number of neighbors;
Δ - a distance measure.
Initialization:
$Y \leftarrow \varnothing$;
Computation:
for $z_i \in Z$ do
    $\mathcal{N} \leftarrow$ the nearest k neighbors to z_i from S according to Δ;
    $y \leftarrow$ the majority label obtained through a voting scheme on $\mathcal{N}$;
    $Y \leftarrow Y \cup \{z_i, y\}$;
Output:
$Y = \{(z_i, y_i) \mid z_i \in Z, y_i \in \mathbf{N}, i \in \{1, 2, \ldots l\}\}$ - the set of predicted labels for the test samples in Z.

Example: 3-NN in a 2-dimensional space for handwritten digit recognition

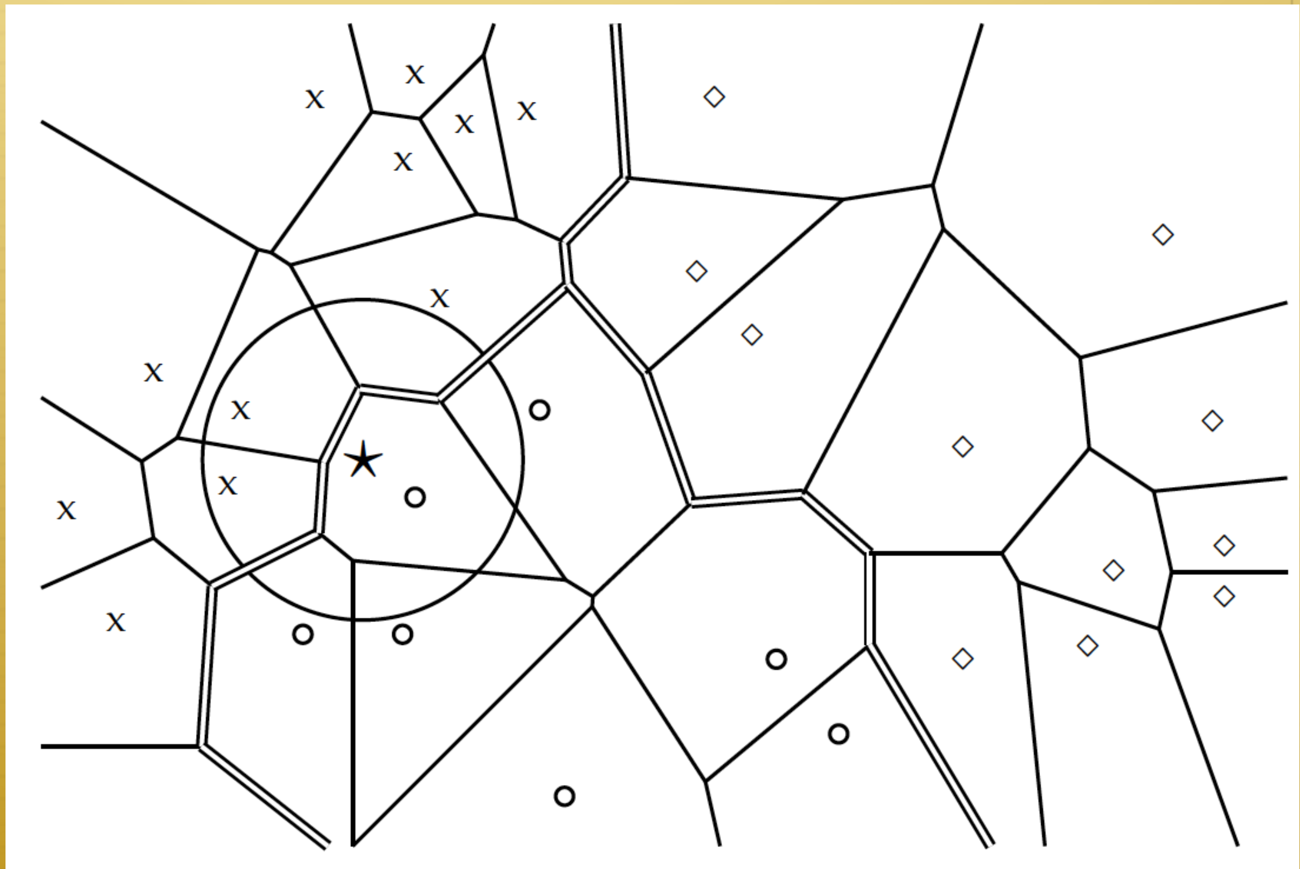


The new sample (*x*) is assigned a label given by the majority of labels in the set of 3 closest neighbors => 4.

As seen, the *k*-NN algorithm does not involve training at all; the decision is solely based on the nearest *k* neighbors of an object with respect to a similarity or distance function.

1-NN: Voronoi tessellation (*parchetare*)

The plane is segmented into polygons, such that each polygon contain all points around a certain object that are closer to that object than to other objects.



From (Manning *et al.*, 2009)

k-NN classifiers

- Performance of a *k*-NN classifier depends on the strength and the discriminatory power of the distance measure used
  - in computer vision: a good choice of the distance metric can help to achieve invariance with respect to transformations: scale, rotation, luminosity and contrast
- Similarity measures are best testing on *k*-NN
  - handwritten digit recognition: **tangent distance** [Simard *et al*., 1996] and the **shape matching distance** [Belongie *et al*., 2002]

On the other hand…

Deep learning

We will come back to this topic in course 8…

- DL provides a way to transform one feature representation into another, by better disentangling the factors of variation that explain the observed data.
- DL algorithms are aimed at discovering multiple levels of representation, or a hierarchy of features.
- The goal of DL is to replace features handcrafted by engineers with features that are learned from data into an end-to-end fashion.
- The success of DL comes from: end-to-end learning process provides a better feature representation when there is enough training data.

P2 – Plant breeding

The Local Rank Distance

Computational biology – the task

- Align reads sampled from several mammals ⇔ human mitochondrial DNA sequence genome.
  - One possible goal: maximize the number of aligned reads sampled from the human genome (true positives), and minimize the number of aligned reads sampled from other mammals (false positives)

Local Rank Distance

- The problem: given a collection R of short DNA reads, and a collection $\mathcal{G}$ of genomes, finds the genome $G \in \mathcal{G}$ that gives a minimum score with respect to R (Ionescu, 2018)
  - used to determine the place of an individual in a phylogenetic tree, by finding the most similar organism in the phylogenetic tree
  - evaluate the performance level of the rank-based aligners and compare them with other alignment tools

LRD - notations

- x - a string over an alphabet Σ
- $|x|$ - the length of x
- strings are indexed starting from position 1, i.e. $x = x[1]x[2]\ldots\, x[|x|]$.
- $x[i : j]$ - the substring $x[i]\; x[i + 1]\ldots\, x[j - 1]$ of x.
- Linear distances between p-gram matches are call p-mers.

LRD – informal definition

- Given a fixed integer $p \geq 1$ (substring lengths), a threshold $m \geq 1$ (maximum distance the two substrings could be found), and two strings x and y over an alphabet Σ, the Local Rank Distance between x and y, denoted by $\Delta_{LRD}(x, y)$, is as follows: for each position i in x ($1 \leq i \leq |x| - p + 1$), the algorithm searches for a certain position j in y ($1 \leq j \leq |y| - p + 1$) such that $x[i : i + p] = y[j : j + p]$ and $|i - j|$ is minimized. If j exists and $|i - j| < m$, then the offset $|i - j|$ is added to the Local Rank Distance. Otherwise, the maximal offset m is added to the Local Rank Distance.

LRD – formal definition

- Let $x, y \in \Sigma^*$ be two strings, and let $p \geq 1$ and $m \geq 1$ be two fixed integer values. The Local Rank Distance between x and y is defined as: $\Delta_{LRD}(x,y) = \Delta_{left}(x,y) + \Delta_{right}(x,y)$

where:

$$\Delta_{left}(x,y) = \sum_{i=1}^{|x|-p+1} \min\{|i-j| \; such \; that \\ 1 \leq j \leq |y|-p+1 \text{ and } x[i:i+p] = y[j:j+p]\} \cup \{m\}$$

$$\Delta_{right}(x,y) = \sum_{j=1}^{|y|-p+1} \min\{|j-i| \; such \; that \\ 1 \leq i \leq |x|-p+1 \text{ and } y[j:j+p] = x[i:i+p]\} \cup \{m\}$$

LRD – example

- Given two strings s_1 = CCGAATACG and s_2 = TGACTCA, and the maximum offset m = 10, the LRD of 1-mers (single characters) between s_1 and s_2 can be computed as follows:

$$\Delta_{LRD}(s_1, s_2) = \Delta_{left} + \Delta_{right}$$

every 1-mers from s_1

$$\Delta_{left} = |1-4| + |2-4| + |3-2| + |4-3| + |5-3| + |6-5| + |7-7| + |8-6| + |9-2| = 19,$$

every 1-mers from s_2

$$\Delta_{right} = |1-6| + |2-3| + |3-4| + |4-2| + |5-6| + |6-8| + |7-7| = 12.$$

Easy to see: $\Delta_{LRD}(s_1, s_2) = \Delta_{LRD}(s_2, s_1)$

LRD exercise – do it yourself!

- For the two strings in the previous example compute LRD for 2-mers.

Design for *P2 – Plant breeding*

- Step 1. Segmentation
  - apply regular expressions for segmenting chromosomal nucleotides strings (ADN) at the following levels:
    - codons (3-grams of nucleotides), genes (by recognizing START-STOP pairs, exones (effects) and introns (no effects), as well as for tagging the proteines (after identification of classes);
- Step 2: Ontological organization of effects
  - identify unique labels in effects (by sorting effects descriptors and eliminating duplicates)
  - use features (strings of protein classes) to recognize <u>hierarchies of effect labels</u> (for instance, a class B is a descendent of a class A if the features of B includes (all) features of A, eventually more

given!

Design for *P2 – Plant breeding*

- Step 3. recognition of hidden actuators (influence of external factors)
  - for instance, by identifying (almost) identical inputs to which correspond different effects
  - try to cluster the differences and give names to these clusters
  - confront with known actuators in the learning sets
  - how to separate multiple influences?
- Step 4: Learning to identify effects associated to inputs
  - apply word2vec? etc. to learn to associate inputs (gene sequences) to effects

References

- S., Belongie, J. Malik and J. Puzicha (2002). Shape matching and object recognition using shape contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(4):509-522, April.
- Radu Tudor Ionescu (2018). Habilitation thesis. Knowledge Transfer between Computer Vision, Text Mining and Computational Biology: New Chapters, University of Bucharest.
- Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze (2009). An Introduction to Information Retrieval, Cambridge UP.
- Patrice Simard, Yann LeCun, John S. Denker and Bernard Victorri (1996). Transformation Invariance in Pattern Recognition, Tangent Distance and Tangent Propagation. Neural Networks: Tricks of the Trade.