Text Categorization

Luboš Popelínský

Faculty of Informatics, Masaryk University Brno

popel@fi.muni.cz, www.fi.muni.cz/~popel

Based partialy on

Raymond J. Mooney’s course of Machine learning

http://www.cs.utexas.edu/~mooney/#Teaching

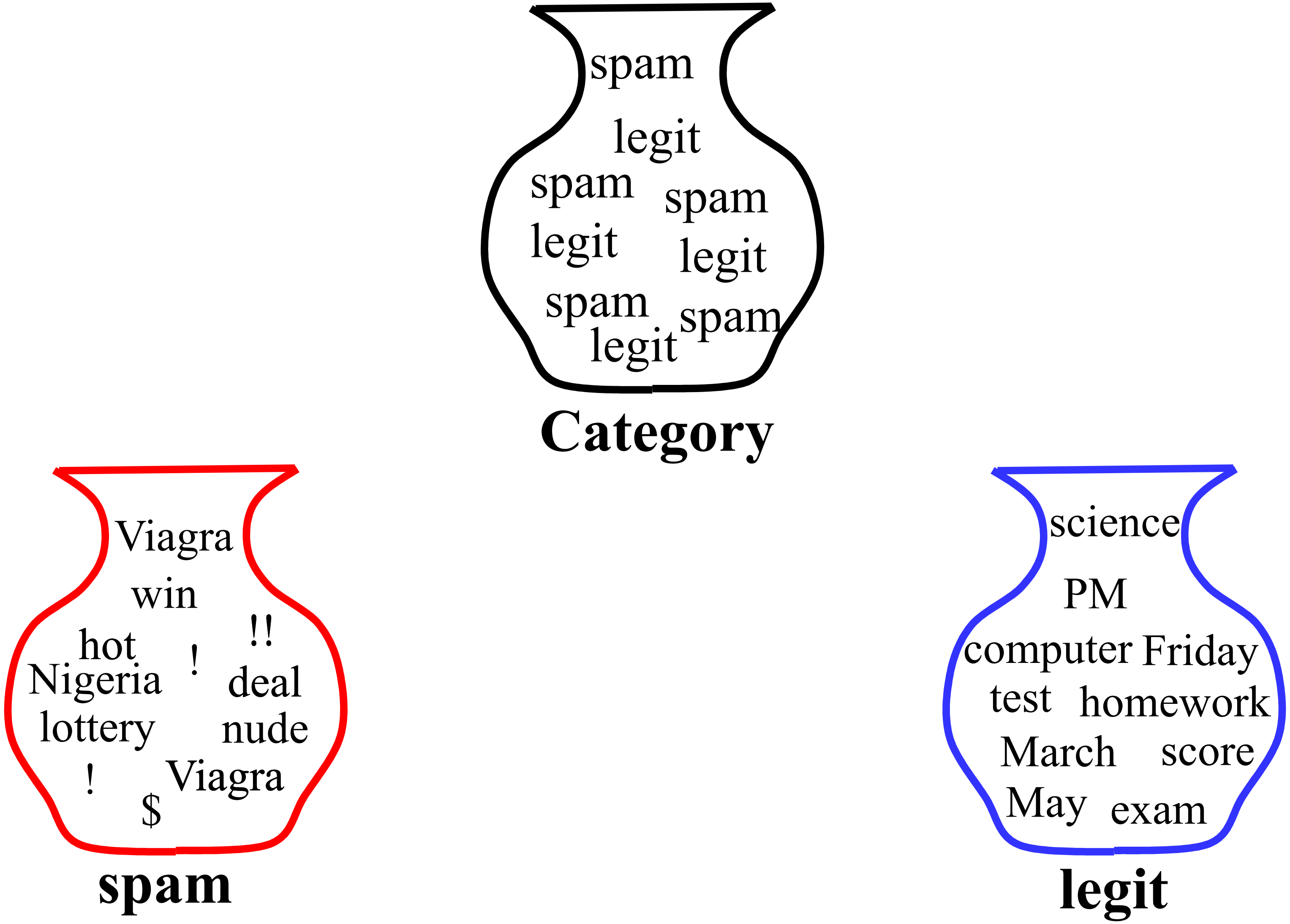
University of Texas at Austin, and

Jose Maria Gomes Hidalgo’s ECML/PKDD tutorial

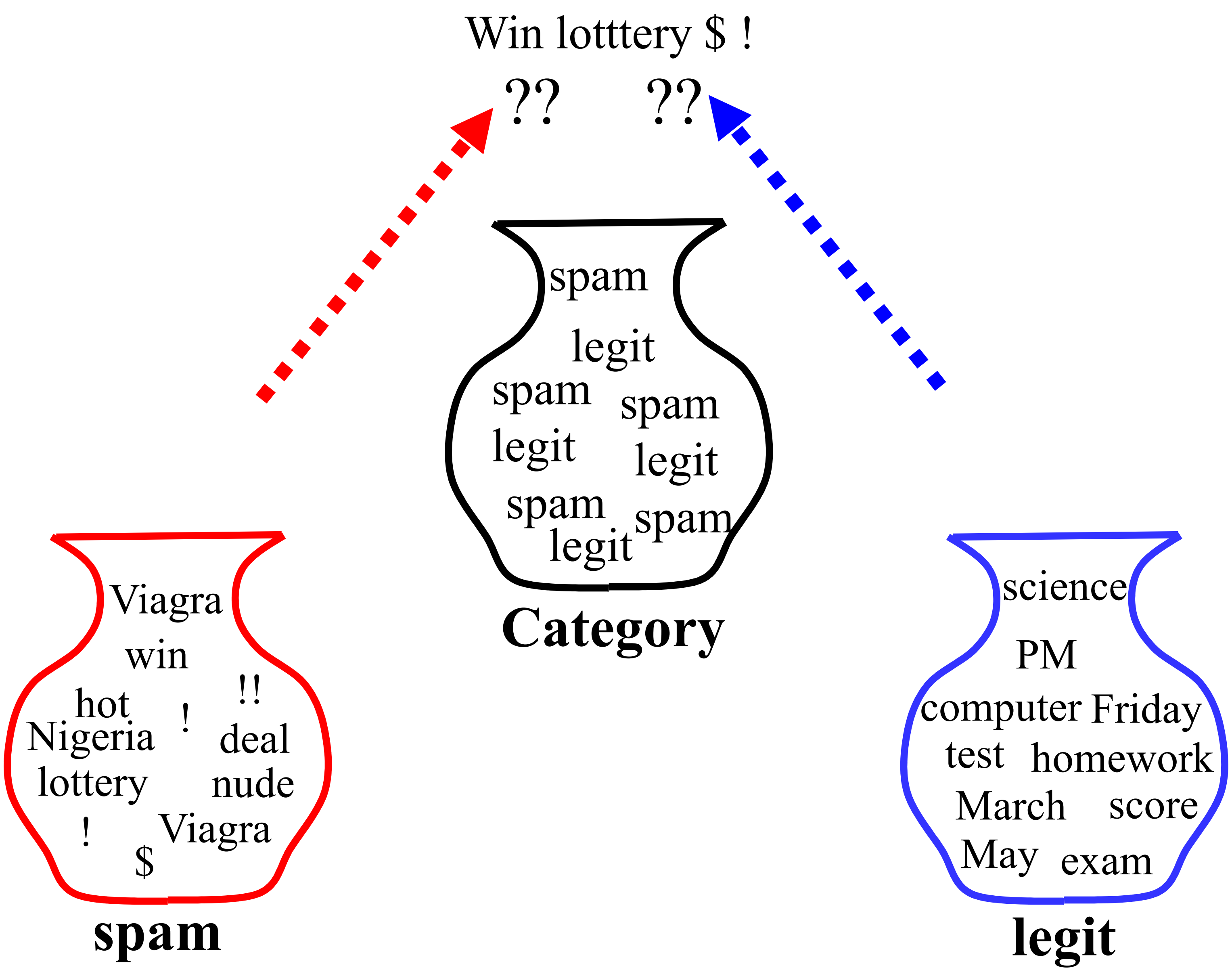
www.fi.muni.cz/~popel/nll, and maybe

Tom Mitchell’s handouts to ML book

Example


spam
legit
spam
spam
legit
legit
spam
spam
legit
Category
Viagra
win
hot
Nigeria
!
!!
deal
lottery
nude
!
Viagra
$
spam
science
PM
computer
Friday
test
homework
March
score
May
exam
legit

Example (cont.)



Text Categorization Applications

- Web pages
  - Recommending
  - Yahoo-like classification
- Newsgroups/Emails
  - Recommending
  - spam filtering
  - Sentiment analysis for marketing
- News articles
  - Personalized newspaper
- Scientific papers
  - Filtering

Task

Based on the text of a document in a collection, assign to each document one or more labels

Usable e.g. for

- topic recognition
- genre detection
- authorship recognition
- author’s meaning (agree/disagree, against/for)
- language recognition

Benchmarks: 20 Newsgroup, Reuters data

Text Categorization and Text Mining

- Text mining

- Unsupervised learning

  words:

  e.g. Latent Semantic Analysis, Key Phrase Extraction

  documents:

  e.g. Document clustering, Topic Detection
- Supervised learning

  words:

  e.g. Part-of-Speech Tagging, Word Sense Disambiguation

  documents:

  e.g. Text categorization/classification/filtering

  Information Extraction

Word Sense Disambiguation

Fui a um ***casamento***.
Casamento é uma experiência interessante.

Ela me ***contou*** tudo.
Eu ***contei*** pelo menos vinte pessoas.
Você pode ***contar*** comigo.

Isto não está ***direito***.
Ela está na faculdade de ***direito***.

… fazer …

Other Applications

Context-sensitive spelling checking – peace of chocolate

Named entity recognition – name, date, location, $, @, URL

Coreference resolution – Carlos não fica na casa. El …

Sentiment classification – positive or negative

Text summarisation

Text Categorization

- The dominant approach is

    Given a set of manually classified (labeled) documents

    Use ML techniques to induce an automatic classifier of new documents

- See [Sebastiani 02] for an in-depth survey

Bag of words

Documents represented as bag of words

a structure of documents, including sentences, is ignored

Decision of a classifier =
based on word frequency in positive and negative examples

What words to choose to reach the best accuracy?

Maybe all …

Text Categorization Methods

- Representations of text are very high dimensional (one feature for each word).
- Vectors are sparse since most words are rare.
- For most text categorization tasks, there are many irrelevant and many relevant features.
- Methods that sum evidence from many or all features (e.g. naïve Bayes, neural-net, SVM) tend to work better than ones that try to isolate just a few relevant features (decision-tree or rule induction).

Text Categorization Algorithm

- Select terms (words, bi-, trigrams, named entities)

- Perform morphological analysis (lemmatization, tagging)
  lemmatization = sou, es → ser

- Term clustering = ser, estar, ficar → artificial term SEF

- Remove nonsignificant terms (stoplist)

- Select terms with highest discriminative power

- Learn classifier

The Vector-Space Model

- Assume *t* distinct terms remain after preprocessing; call them index terms or the vocabulary.
- These "orthogonal" terms form a vector space.

  Dimension = t = |vocabulary|
- Each term, i, in a document or query, j, is given a real-valued weight, w_{ij}.
- Both documents and queries are expressed as t-dimensional vectors:

  $d_j = (w_{1j}, w_{2j}, \ldots, w_{tj})$

Document Collection

- A collection of *n* documents can be represented in the vector space model by a term-document matrix.
- An entry in the matrix corresponds to the "weight" of a term in the document; zero means the term has no significance in the document or it simply doesn't exist in the document.

$$
\begin{pmatrix}
 & T_1 & T_2 & \ldots & T_t \\
D_1 & w_{11} & w_{21} & \ldots & w_{t1} \\
D_2 & w_{12} & w_{22} & \ldots & w_{t2} \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & \vdots & & \vdots \\
D_n & w_{1n} & w_{2n} & \ldots & w_{tn}
\end{pmatrix}
$$

Naïve Bayes for Text

- Modeled as generating a bag of words for a document in a given category by repeatedly sampling with replacement from a vocabulary $V = \{w_1, w_2, \ldots w_m\}$ based on the probabilities $P(w_j \mid c_i)$.
- Smooth probability estimates with Laplace *m*-estimates assuming a uniform distribution over all words ($p = 1/|V|$) and $m = |V|$
  - Equivalent to a virtual sample of seeing each word in each category exactly once.

Text Naïve Bayes Algorithm (Train)

Let V be the vocabulary of all words in the documents in D

For each category $c_i \in C$

Let D_i be the subset of documents in D in category c_i

$P(c_i) = |D_i| / |D|$

Let T_i be the concatenation of all the documents in D_i

Let n_i be the total number of word occurrences in T_i

For each word $w_j \in V$

Let n_{ij} be the number of occurrences of w_j in T_i

Let $P(w_j \mid c_i) = (n_{ij} + 1) / (n_i + |V|)$

Text Naïve Bayes Algorithm (Test)

Given a test document X

Let n be the number of word occurrences in X

Return the category:

$$\underset{c_i \in C}{\operatorname{argmax}} P(c_i) \prod_{i=1}^{n} P(a_i \mid c_i)$$

where a_i is the word occurring the ith position in X

Text Categorization: Summary

- Bag of words
- Vector model
- Feature/Term selection and Term extraction
- Naïve Bayes Classifier, SVM, neural nets
- Applications

Textual Similarity Metrics

- Measuring similarity of two texts is a well-studied problem.
- Standard metrics are based on a “bag of words” model of a document that ignores word order and syntactic structure.
- May involve removing common “stop words” and stemming to reduce words to their root form.
- Vector-space model from Information Retrieval (IR) is the standard approach.
- Other metrics (e.g. edit-distance) are also used.

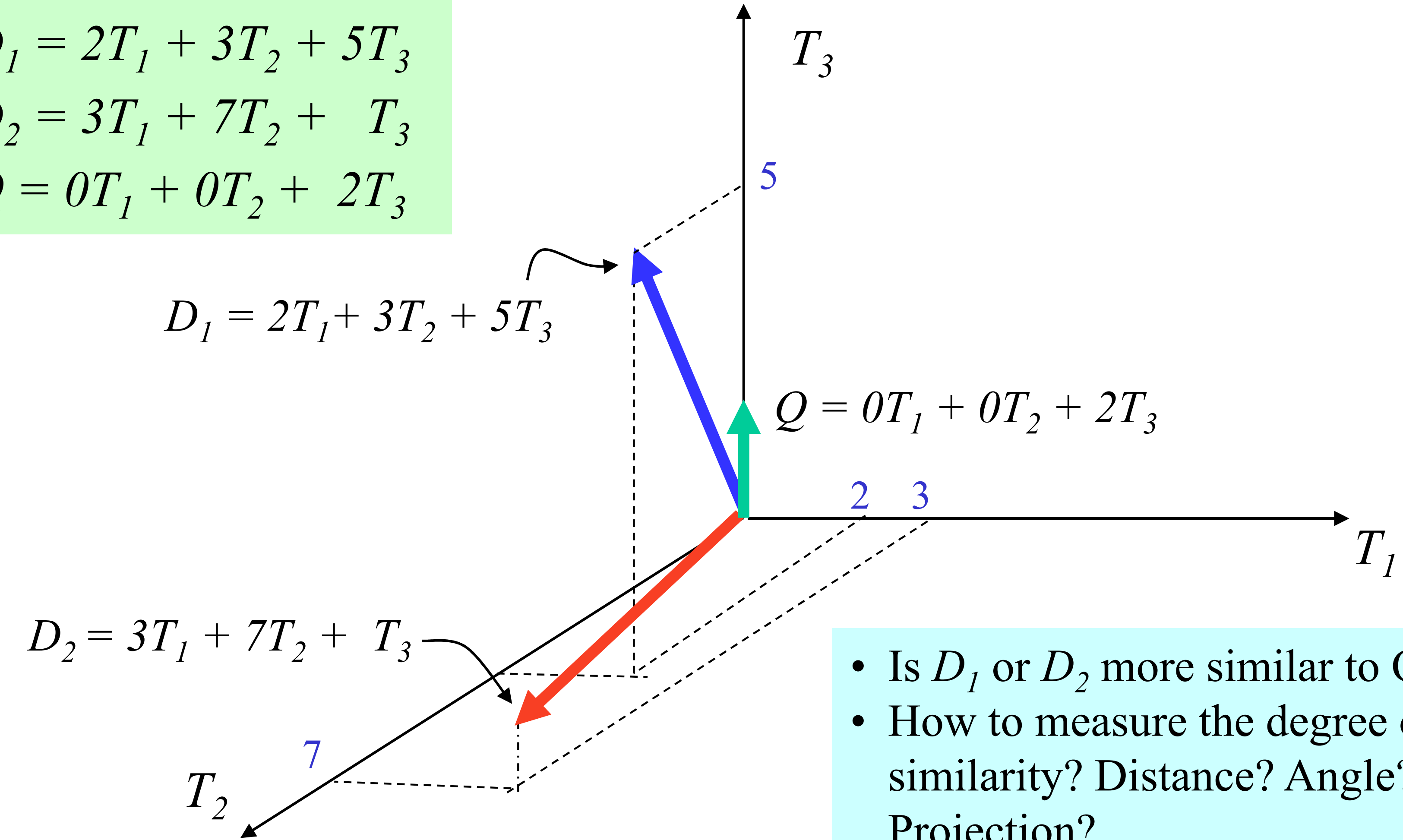
Graphic Representation

Example:

$D_1 = 2T_1 + 3T_2 + 5T_3$

$D_2 = 3T_1 + 7T_2 + T_3$

$Q = 0T_1 + 0T_2 + 2T_3$



- Is D_1 or D_2 more similar to Q?
- How to measure the degree of similarity? Distance? Angle? Projection?

Term Weights: Term Frequency

- More frequent terms in a document are more important, i.e. more indicative of the topic.

  f_{ij} = frequency of term i in document j

- May want to normalize *term frequency* (*tf*) by dividing by the frequency of the most common term in the document:

  $$tf_{ij} = f_{ij} / max_i\{f_{ij}\}$$

Term Weights: Inverse Document Frequency

- Terms that appear in many *different* documents are *less* indicative of overall topic.

  df_i = document frequency of term i

  = number of documents containing term i

  idf_i = inverse document frequency of term i,

  $= \log_2 (N/ df_i)$

  (N: total number of documents)
- An indication of a term's *discrimination* power.
- Log used to dampen the effect relative to *tf*.

TF-IDF Weighting

- A typical combined term importance indicator is *tf-idf weighting*:

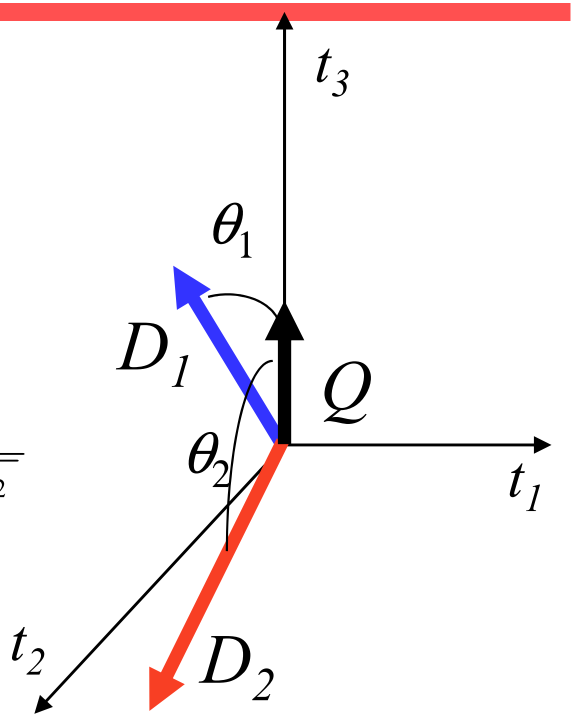
$$w_{ij} = tf_{ij}\, idf_i = tf_{ij} \log_2 (N/\, df_i)$$

- A term occurring frequently in the document but rarely in the rest of the collection is given high weight.
- Many other ways of determining term weights have been proposed.
- Experimentally, *tf-idf* has been found to work well.

Cosine Similarity Measure

- Cosine similarity measures the cosine of the angle between two vectors.
- Inner product normalized by the vector lengths.

$$\text{CosSim}(\boldsymbol{d}_j, \boldsymbol{q}) = \frac{\vec{d}_j \cdot \vec{q}}{\left|\vec{d}_j\right| \cdot \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \cdot \sum_{i=1}^{t} w_{iq}^2}}$$



$D_1 = 2T_1 + 3T_2 + 5T_3$ CosSim(D_1 , Q) = $10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$
$D_2 = 3T_1 + 7T_2 + 1T_3$ CosSim(D_2 , Q) = $2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$
$Q = 0T_1 + 0T_2 + 2T_3$

D_1 is 6 times better than D_2 using cosine similarity but only 5 times better using inner product.

Text Document Filtering

Example: filtering interesting news from a news group

Idea:

1. label interesting documents (positive examples)
2. and noninteresting ones (negative examples)
3. run a learning algorithm

also useful for spam detection and filtering web pages