



Categorization for MALACH ***AMSC 663, Semester Progress Report***

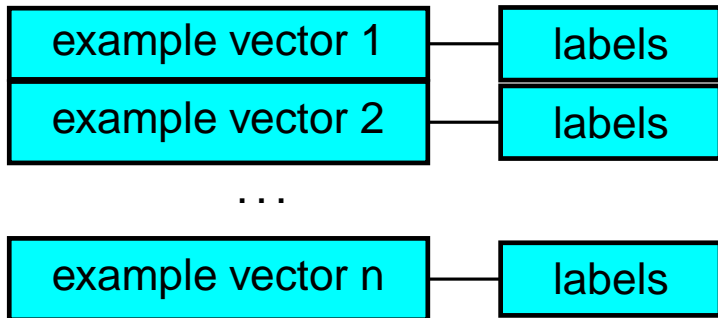
J. Scott Olsson
olsson@math.umd.edu

Supervisor: Doug Oard @ University of Maryland, College Park,
oard@umd.edu

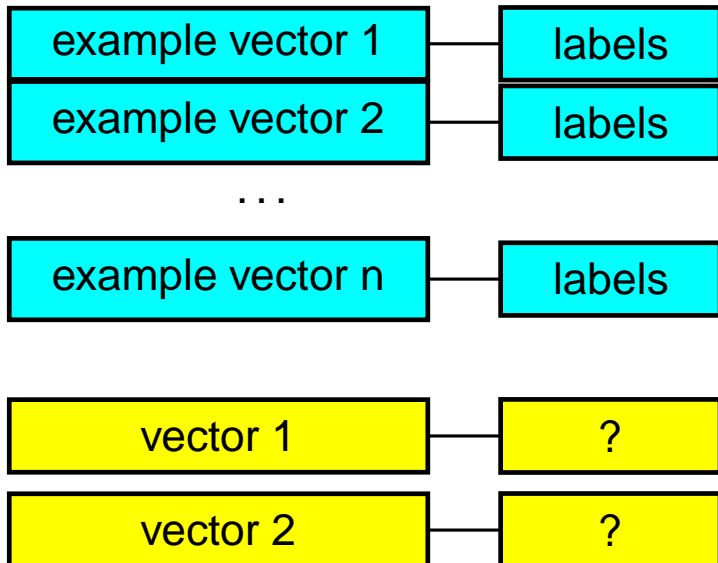
Overview

1. The problem and its challenges
2. A kNN implementation
3. Evaluation
4. What's next

The categorization problem



The categorization problem



The categorization problem

<doc>Happy the
happiest dog is
happily
barking.</doc>

example vector 1

labels

example vector 2

labels

...

example vector n

labels

vector 1

?

vector 2

?

The categorization problem

<doc>Happy the
happiest dog is
happily
barking.</doc>

?

example vector 1	labels
example vector 2	labels

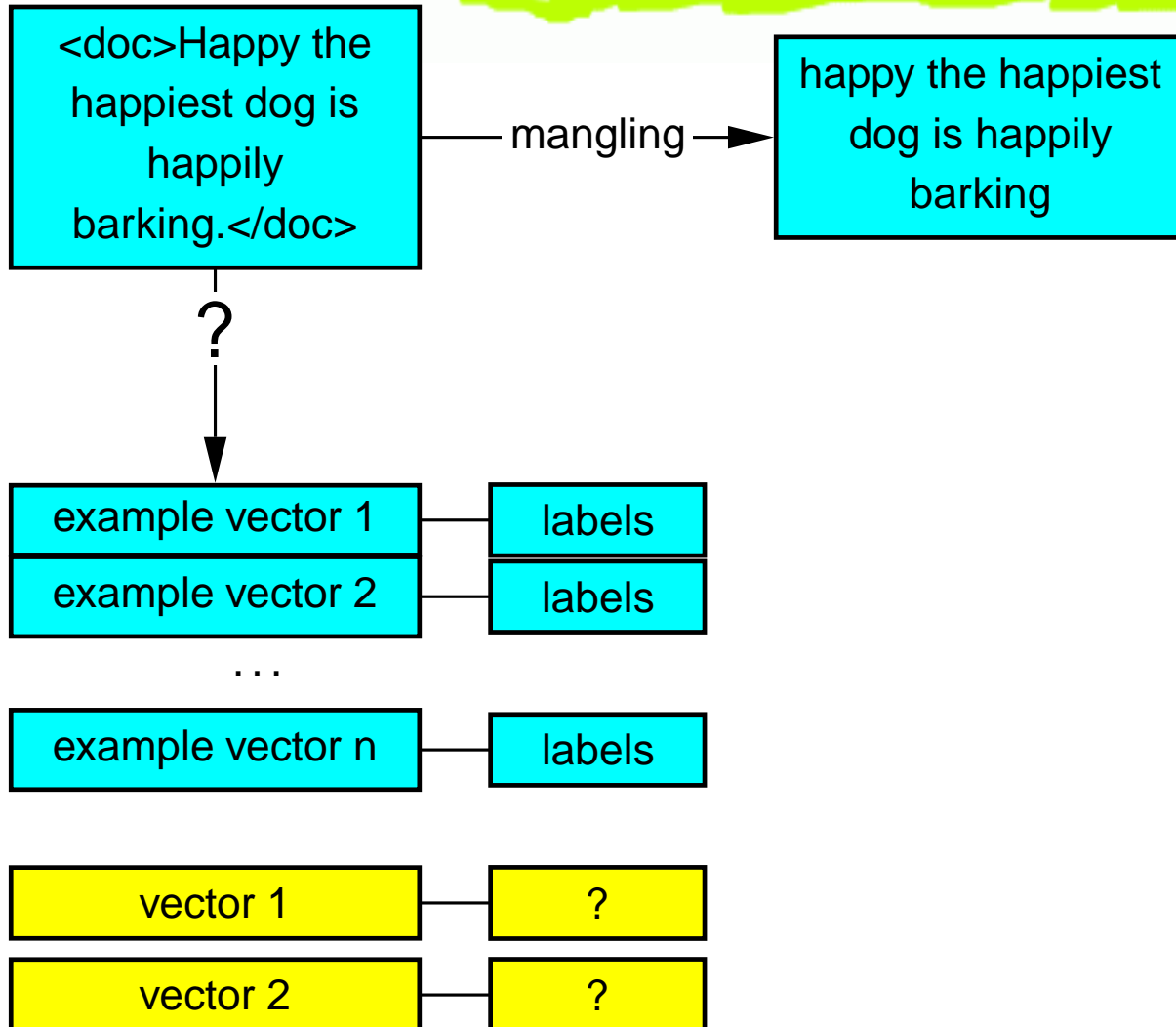
...

example vector n	labels
------------------	--------

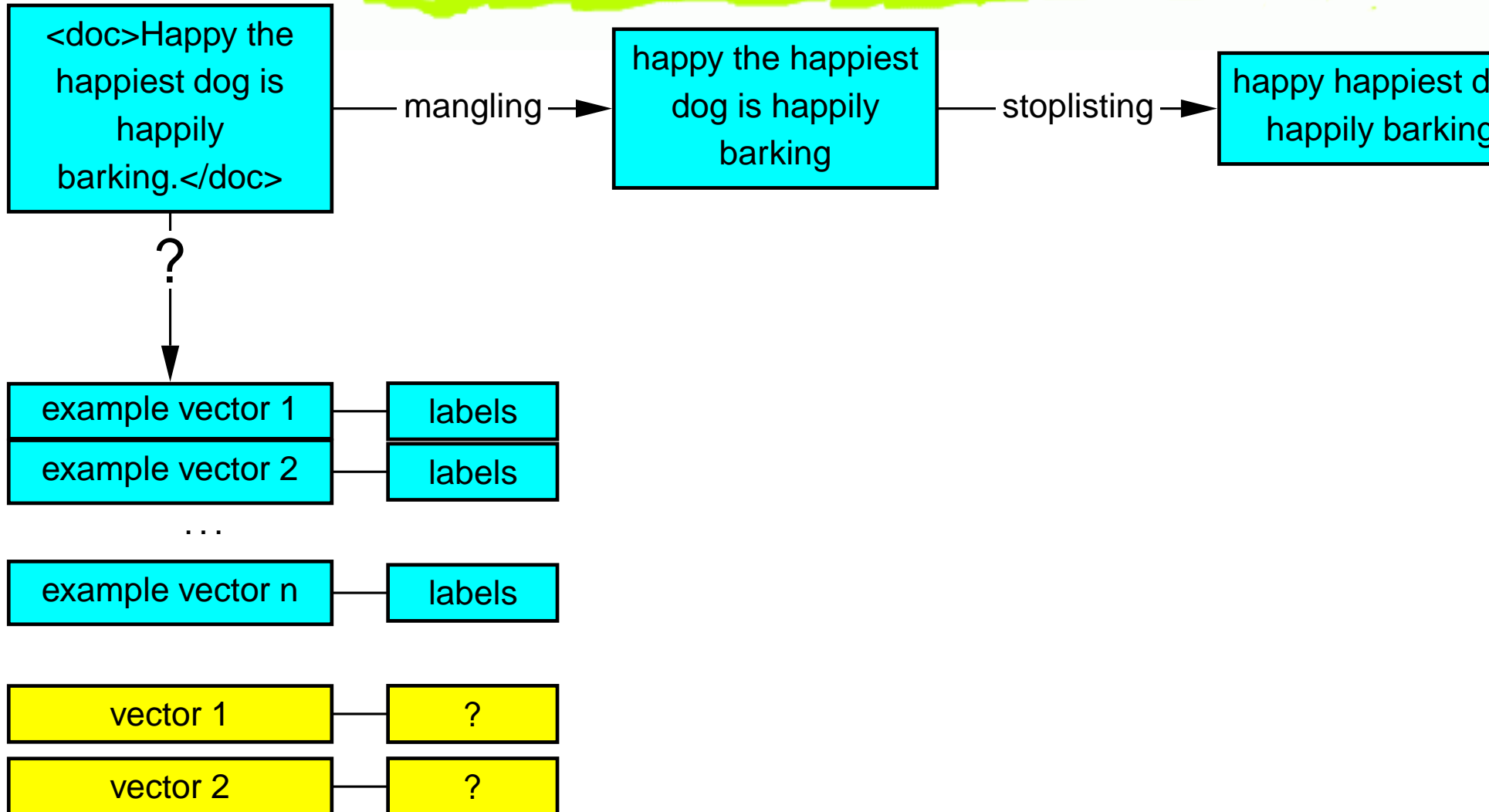
vector 1	?
----------	---

vector 2	?
----------	---

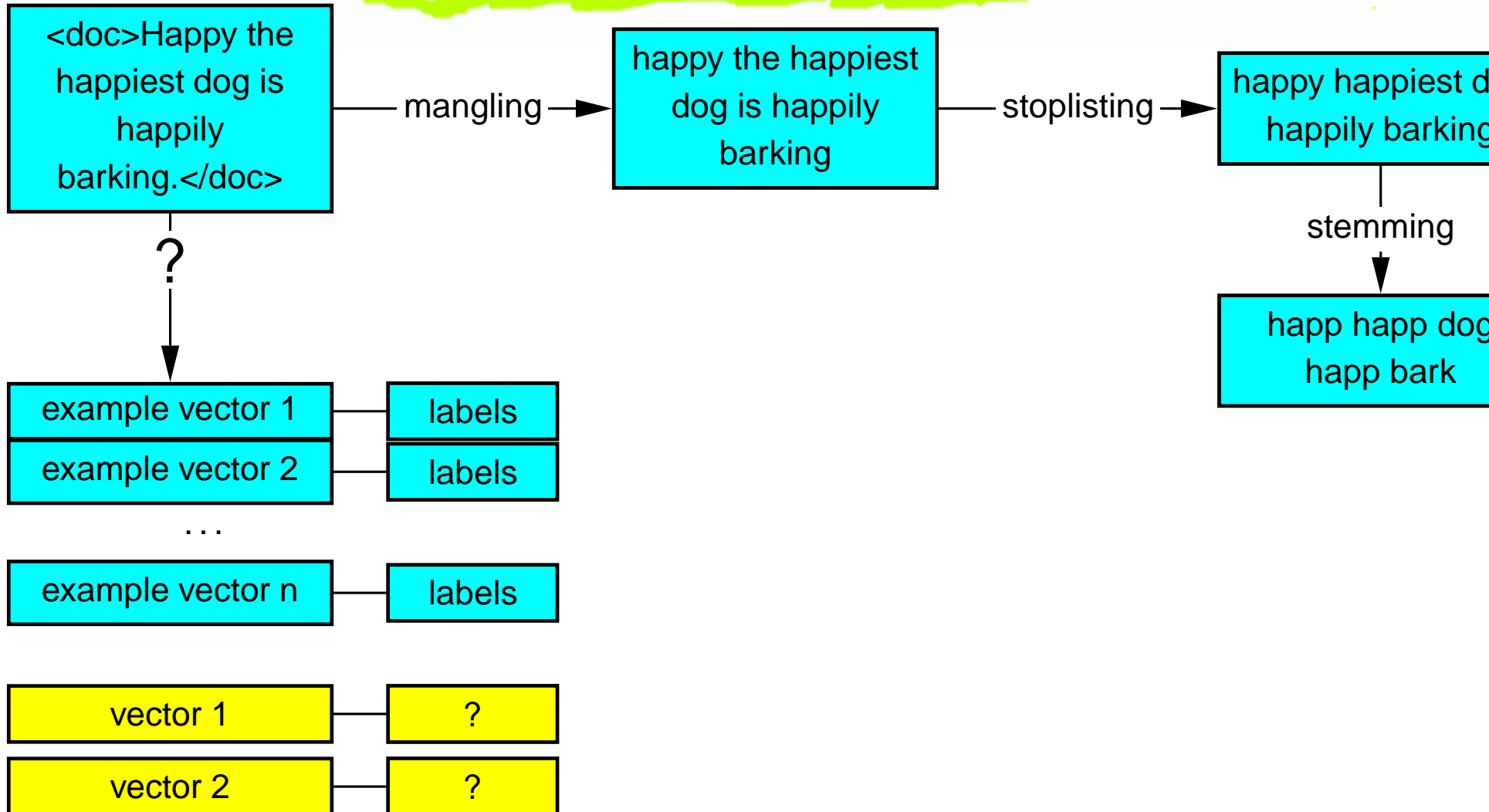
The categorization problem



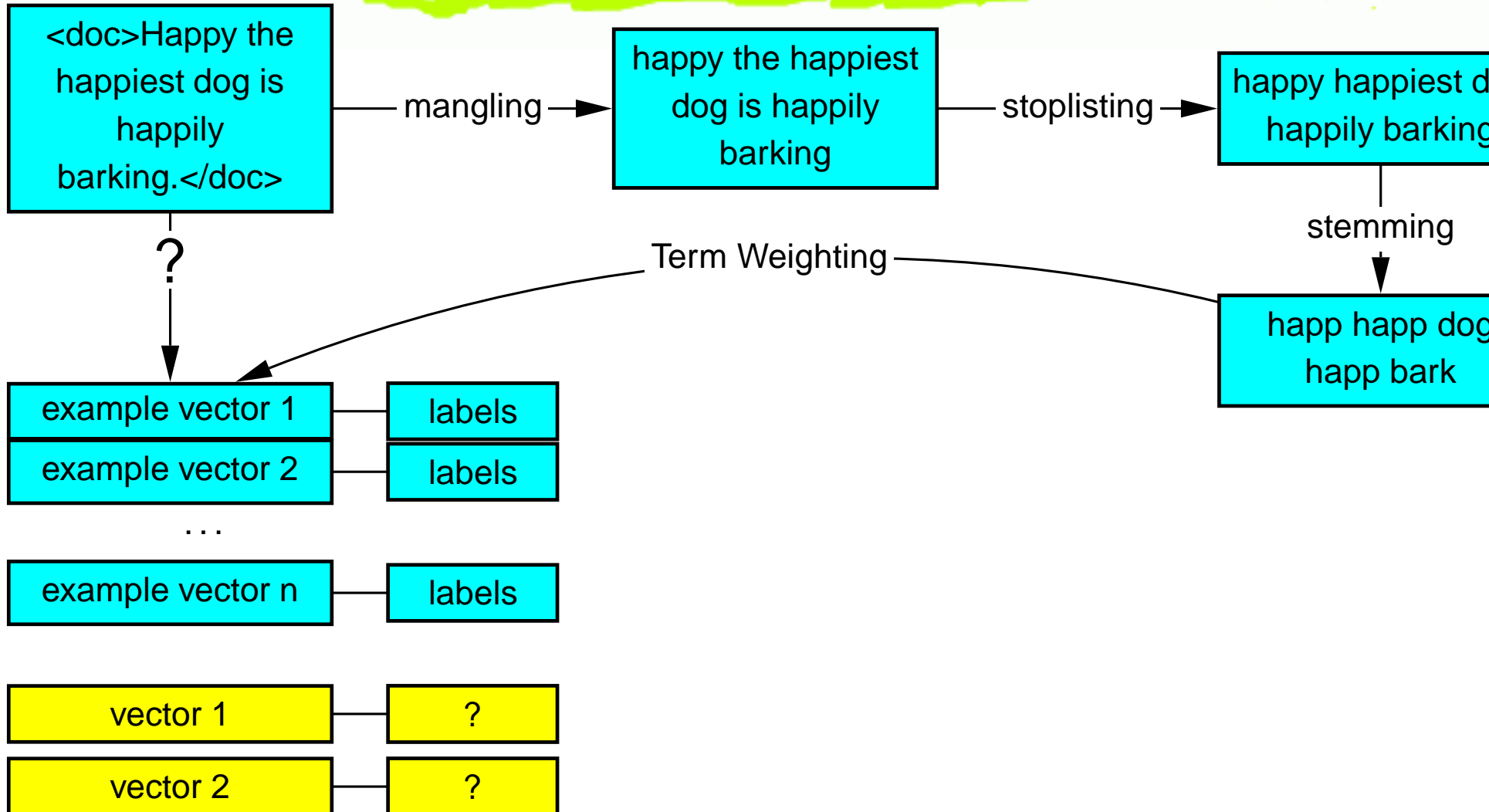
The categorization problem



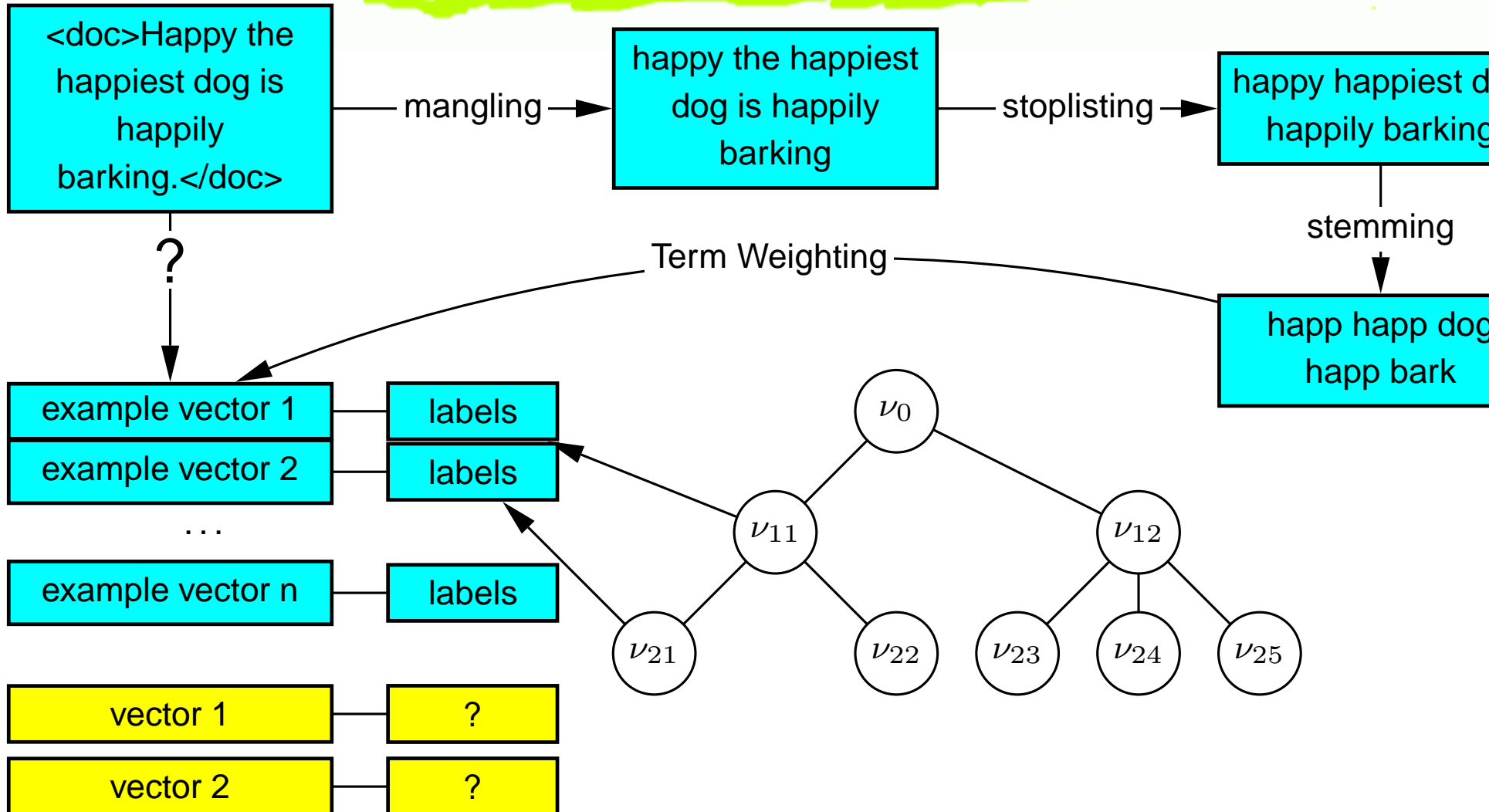
The categorization problem



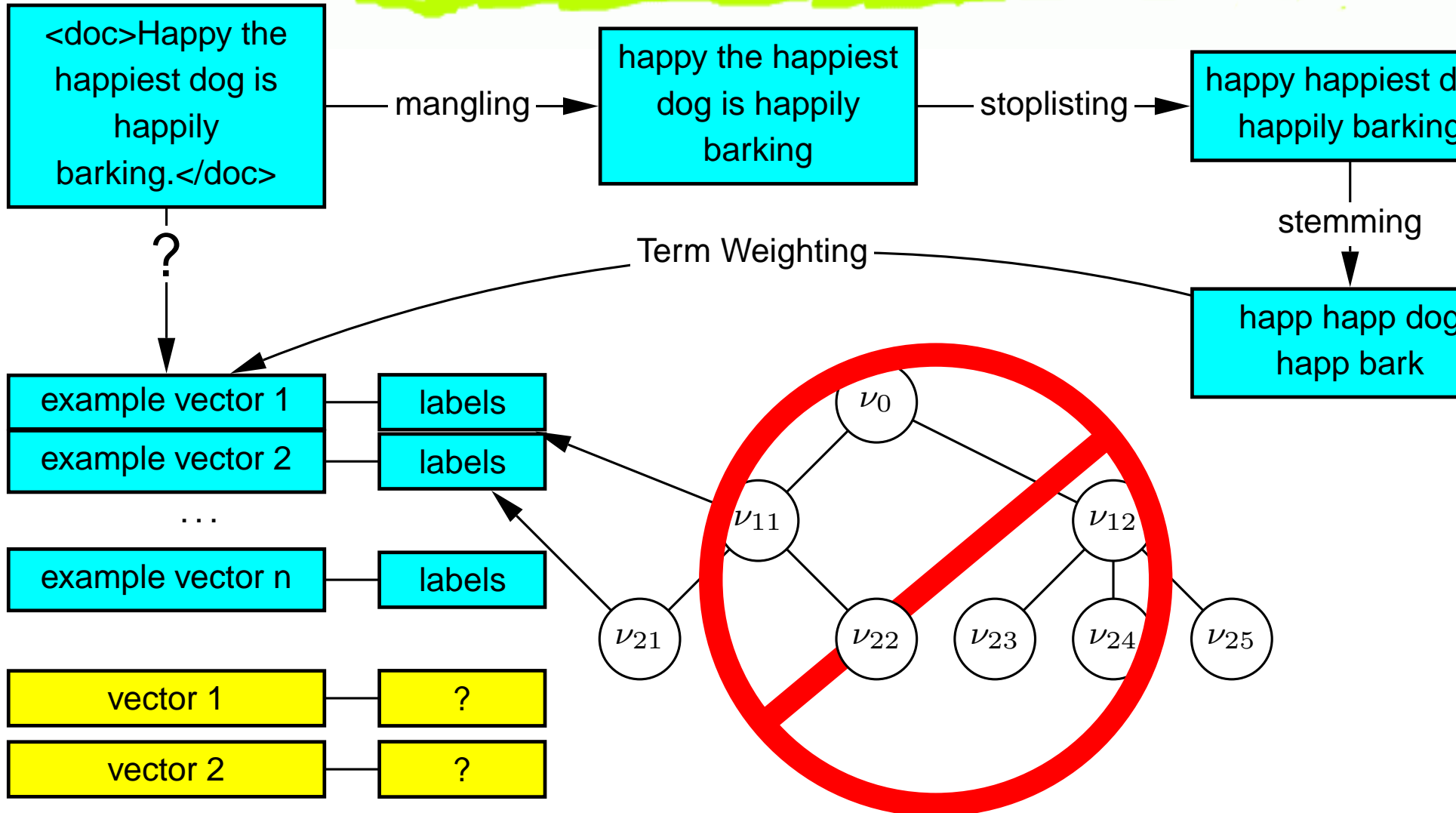
The categorization problem



The categorization problem



The categorization problem



The problem domain

The 20 Newsgroups dataset

- ⑥ well formed data
- ⑥ strict hierarchy
- ⑥ a standard testing set

The problem domain

The 20 Newsgroups dataset

- ⑥ well formed data
- ⑥ strict hierarchy
- ⑥ a standard testing set

Interview data

- ⑥ unique problems in data
- ⑥ hierarchical structure, although its usefulness remains to be determined
- ⑥ model comparisons must be in-group

Computing issues

Data can be ugly...

- ⑥ mmm... perl.

Computing issues

Data can be ugly...

- ⑥ mmm... perl.

Data structures...

- ⑥ The inverted index
- ⑥ B-Trees
- ⑥ AVL-Trees

Computing issues

Data can be ugly...

- ⑥ mmm... perl.

Data structures...

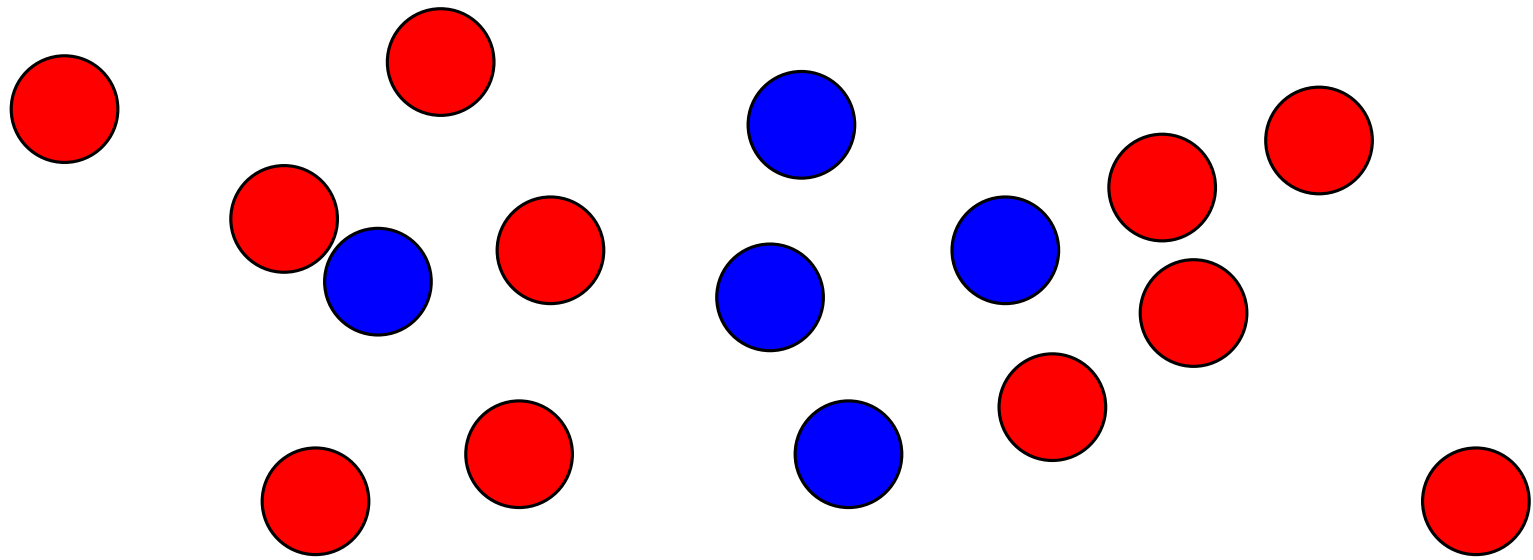
- ⑥ The inverted index
- ⑥ B-Trees
- ⑥ AVL-Trees

...and

- ⑥ Security
- ⑥ Parallelization

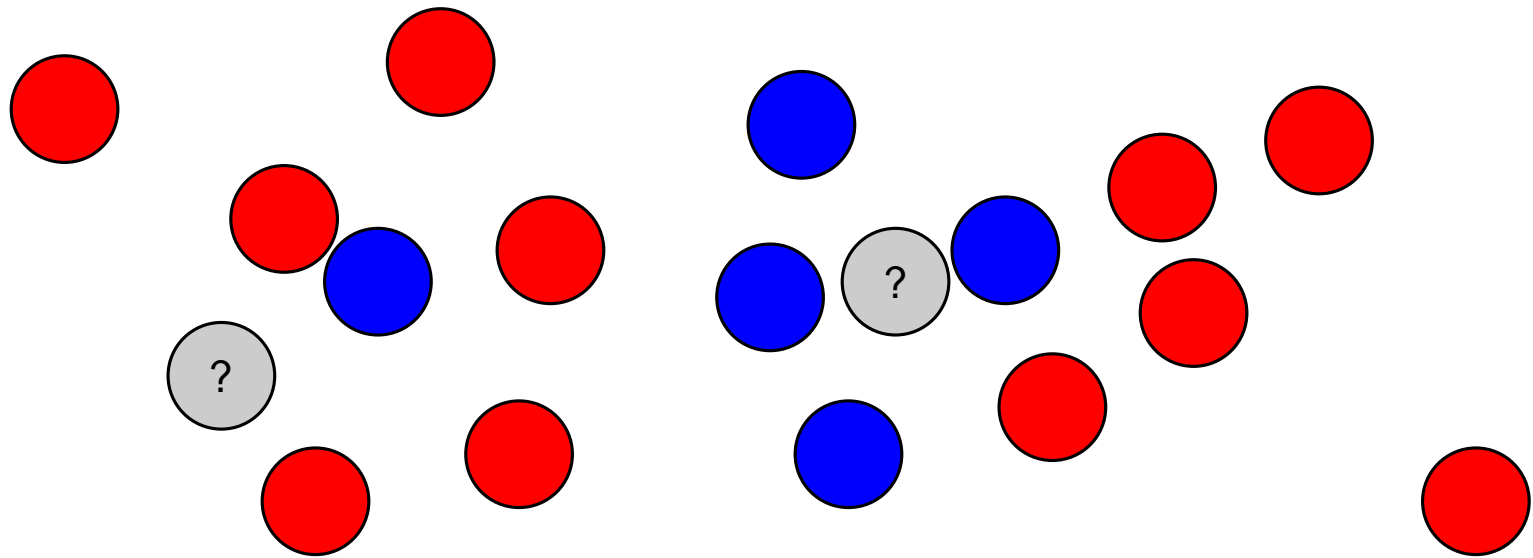
How kNN works...

Training data...



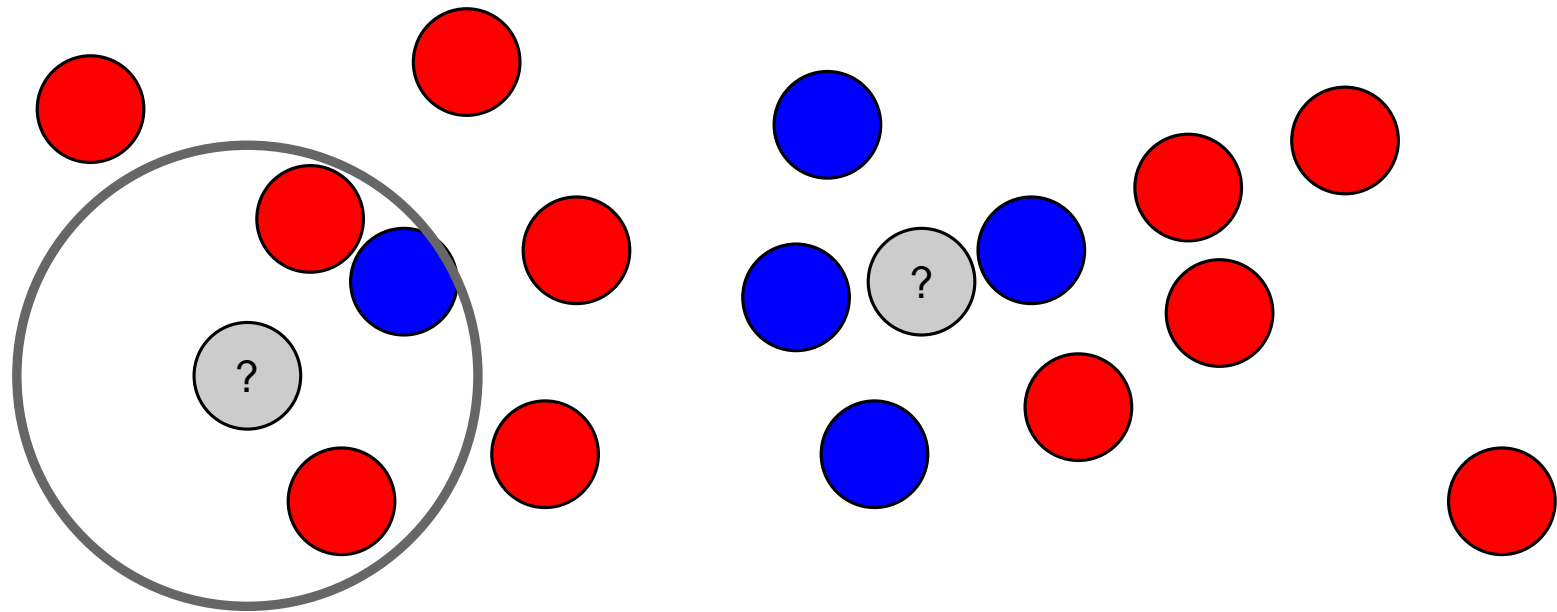
How kNN works...

Training data...



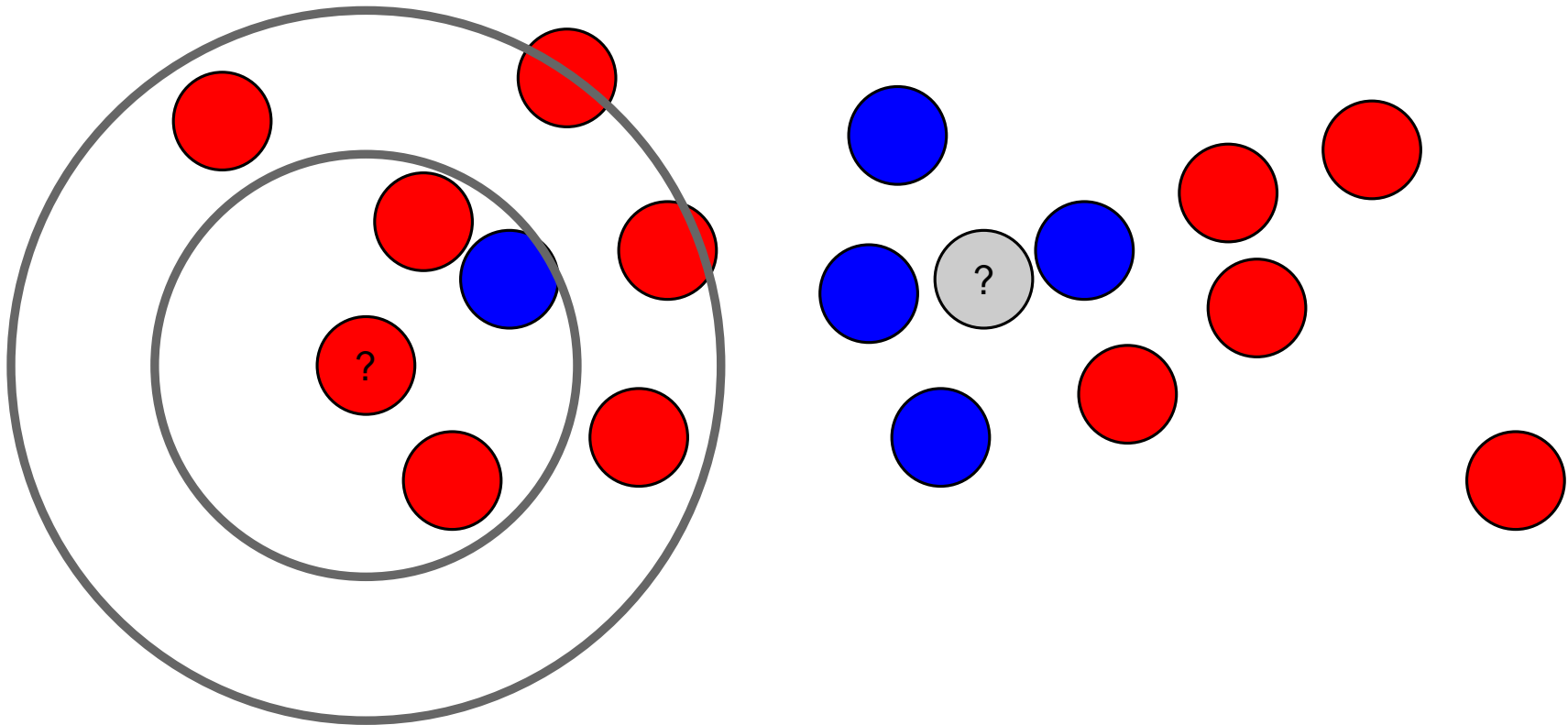
How kNN works...

Training data...



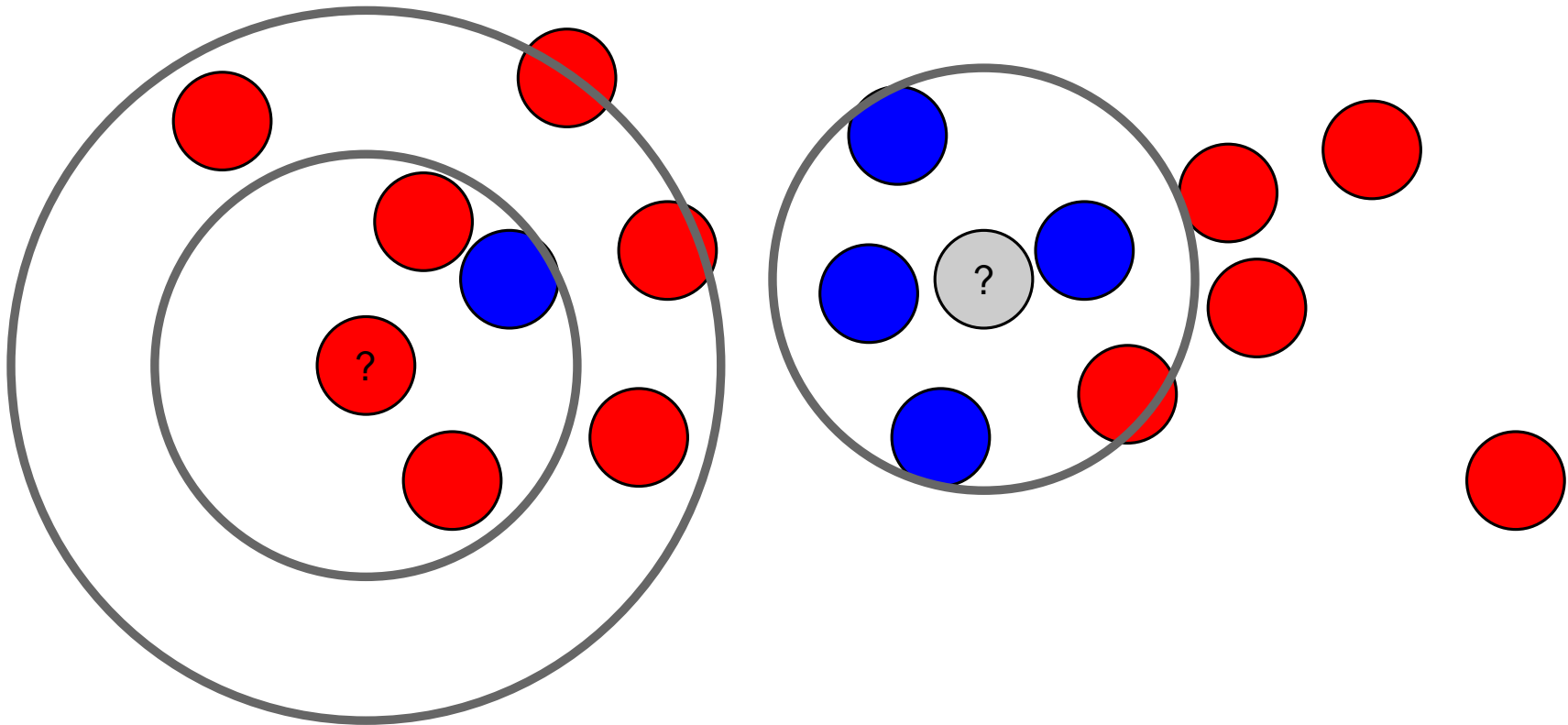
How kNN works...

Training data...



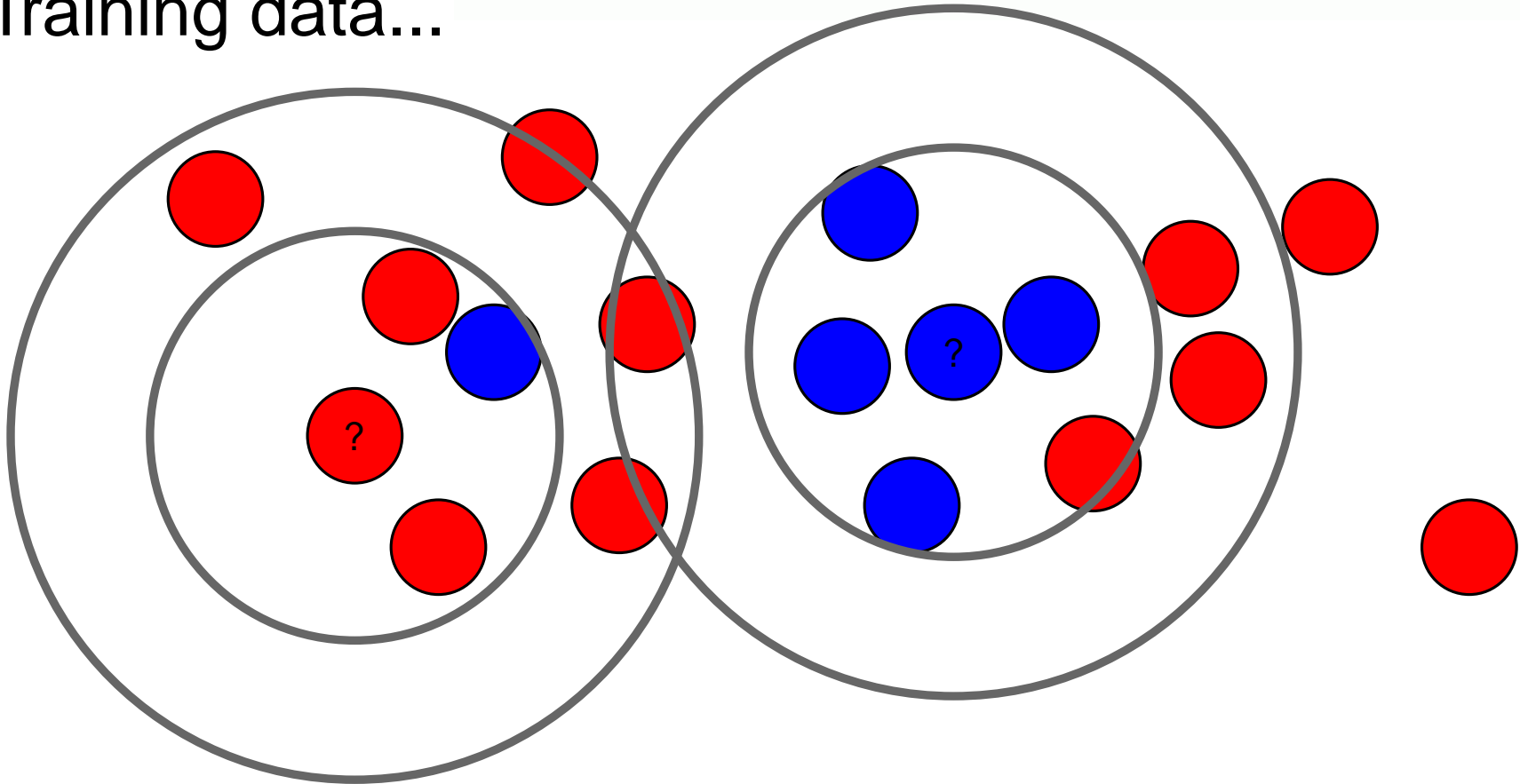
How kNN works...

Training data...



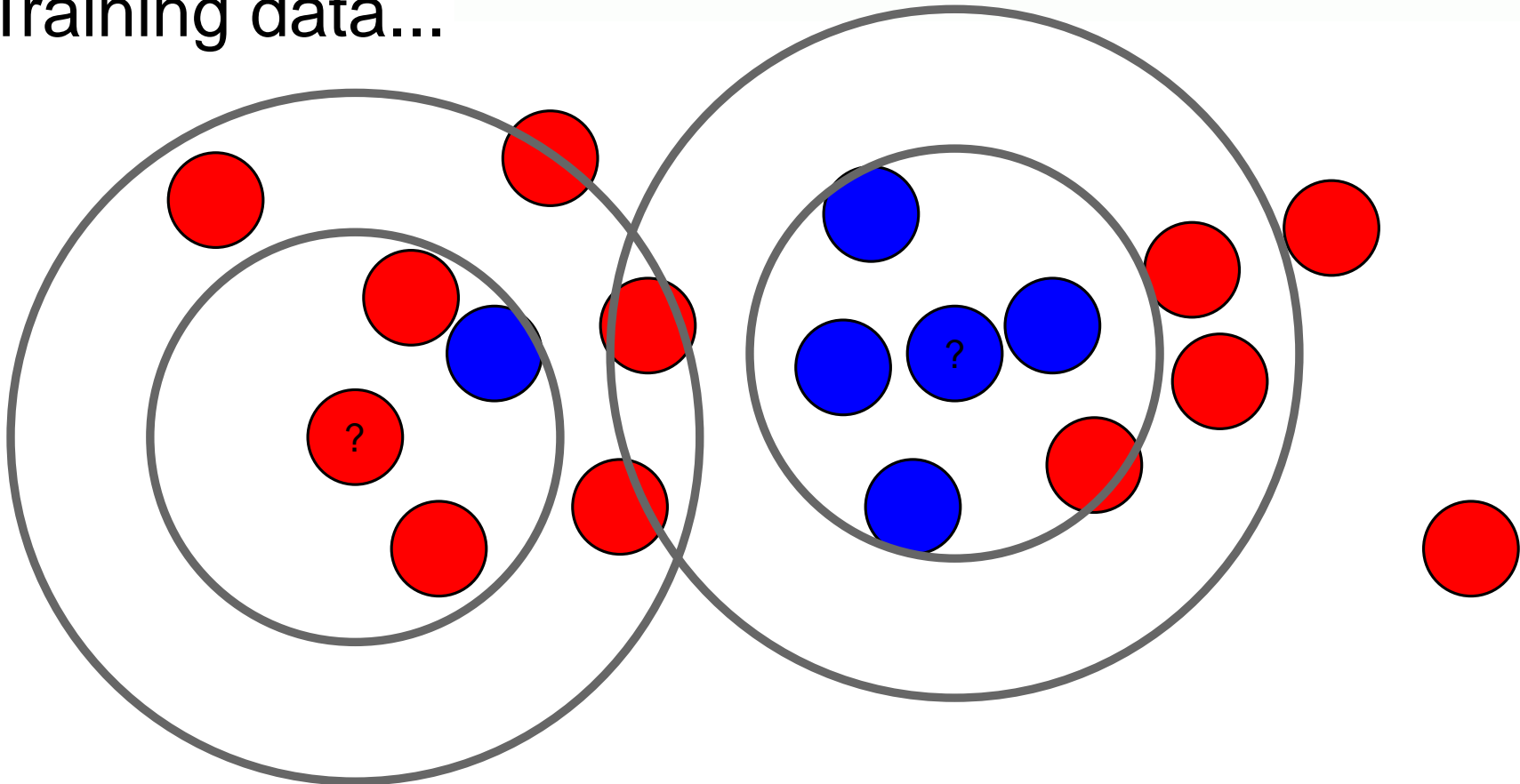
How kNN works...

Training data...



How kNN works...

Training data...



implies categories for unlabeled documents!

kNN categorization

A **simple** idea: take the category(ies) from the k "nearest" neighbors in the training set.

How to decide what documents are "nearby"?

- ⑥ Term weighting
- ⑥ Similarity measures

These are the crux of any kNN categorizer.

There are **many** ways to do each of these; the suitability of each will depend heavily on the nature of the data!

A first pass at an implemenation...

N : number of documents in the collection

TF : number of documents a term appears in

$$IDF = \log(1 + N/TF)$$

For each document term, calculate
 $DTW = (1 + \log(DTF))IDF$

DTF : number of times a term appears in a particular document

A first pass at an implementation...

N : number of documents in the collection

TF : number of documents a term appears in

$$IDF = \log(1 + N/TF)$$

For each document term, calculate
 $DTW = (1 + \log(DTF))IDF$

DTF : number of times a term appears in a particular document

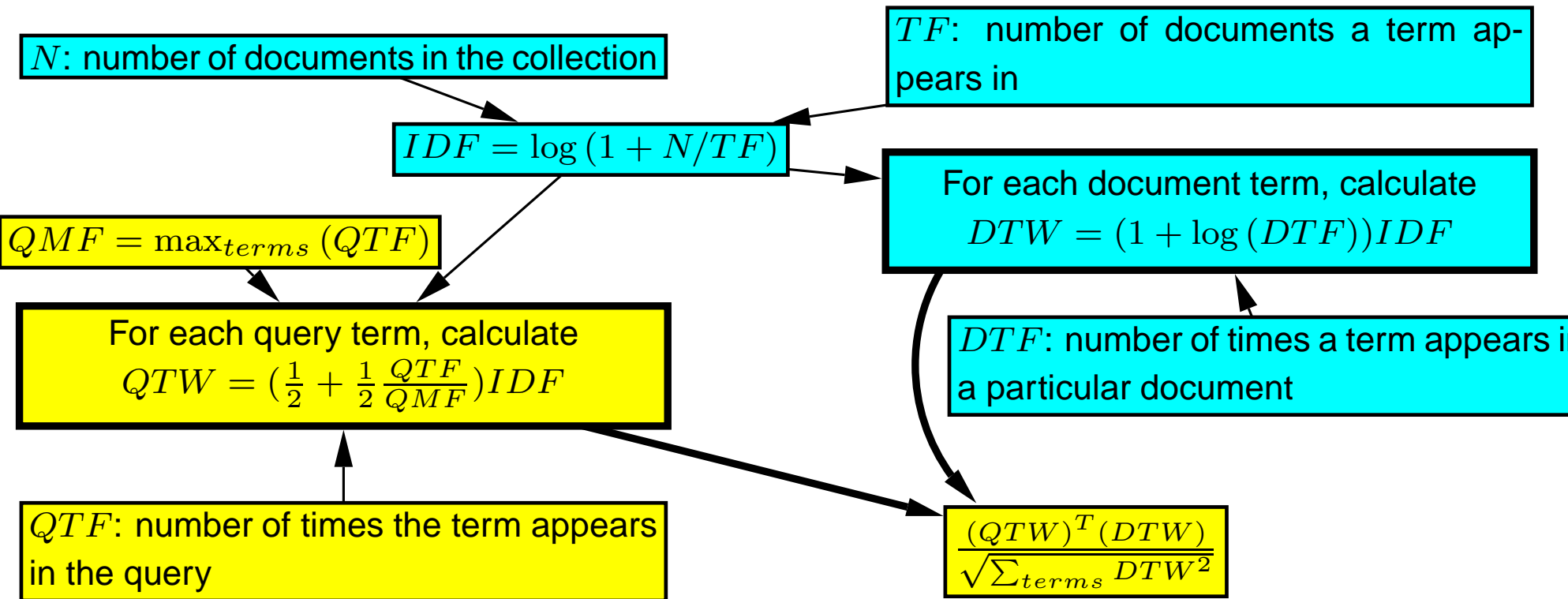
$$QMF = \max_{terms} (QTF)$$

For each query term, calculate

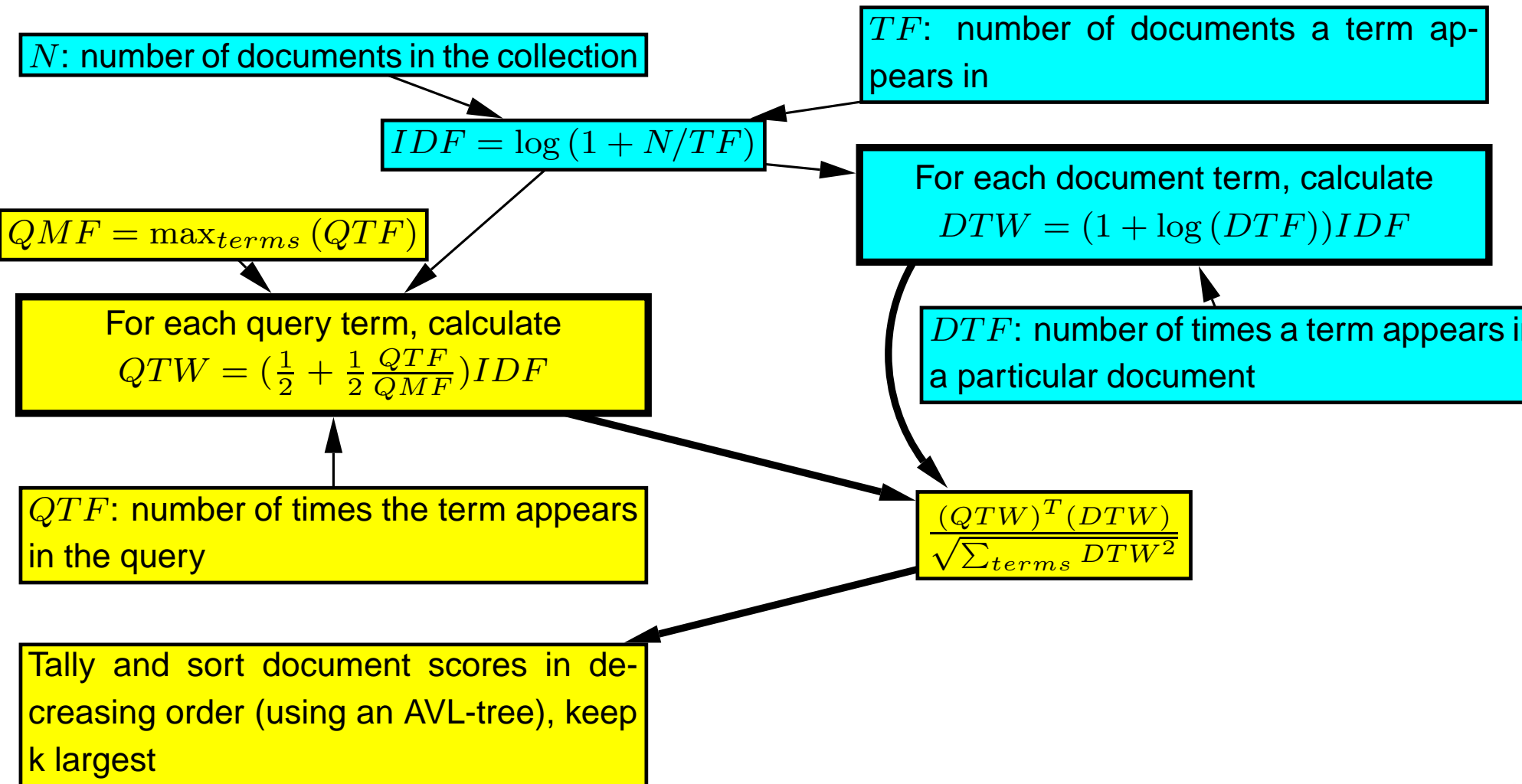
$$QTW = \left(\frac{1}{2} + \frac{1}{2} \frac{QTF}{QMF}\right) IDF$$

QTF : number of times the term appears in the query

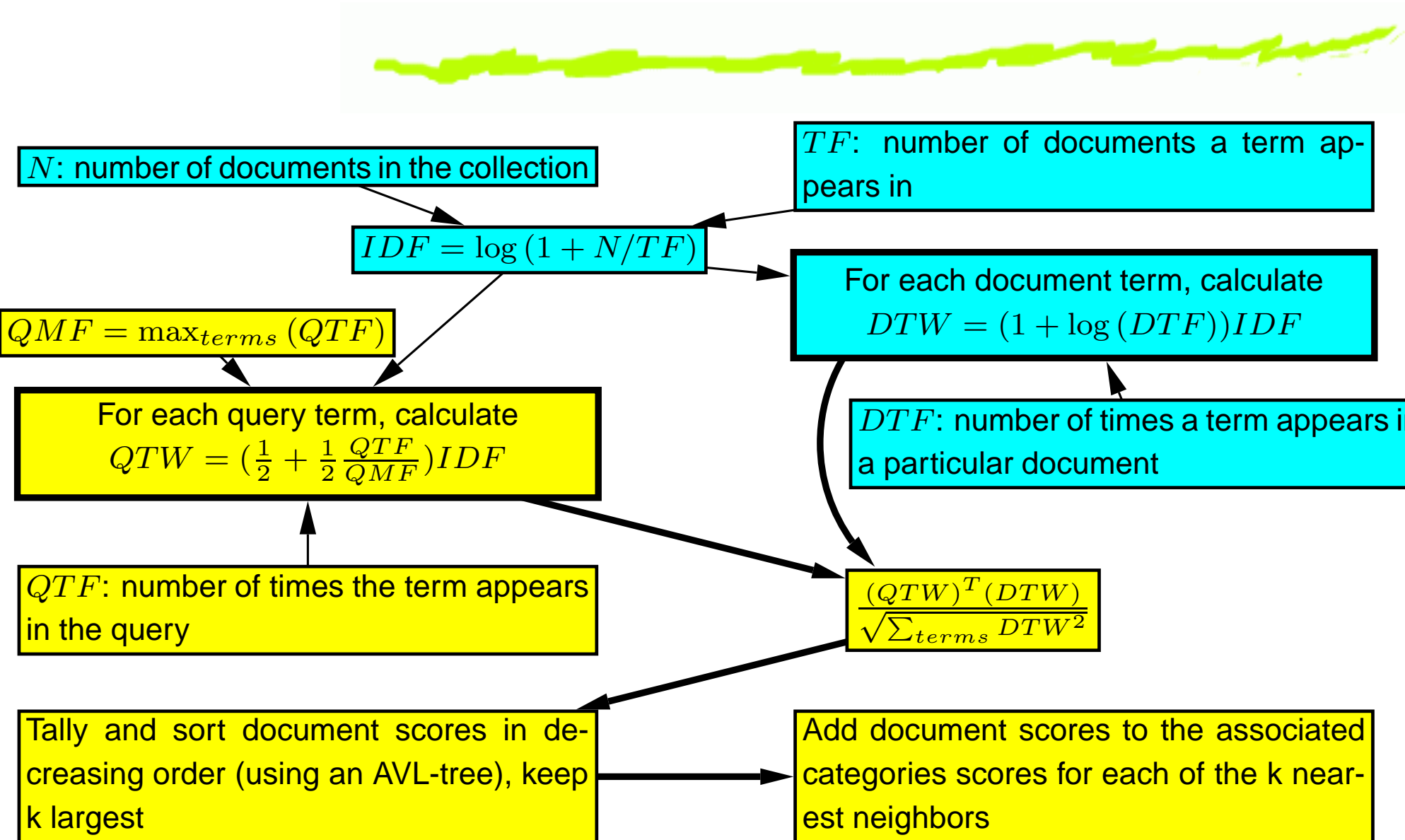
A first pass at an implementation...



A first pass at an implemenation...

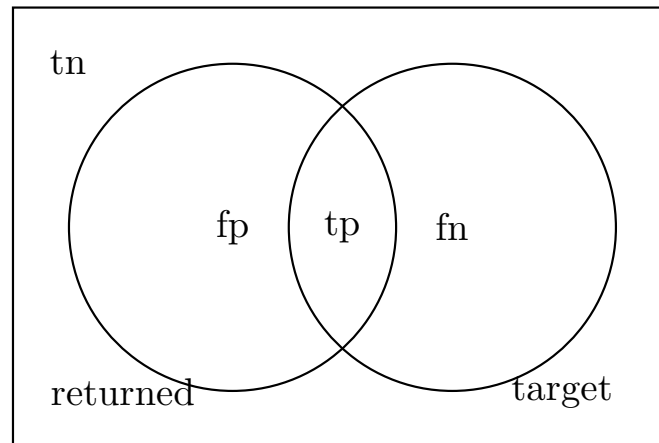


A first pass at an implemenation...



Evaluation as insight

A quick recap of definitions...



tp: true positive
fp: false positive
tn: true negative
fn: false negative

Contingency tables

	yes is correct	no is correct
yes was assigned	tp	fp
no was assigned	fn	tn

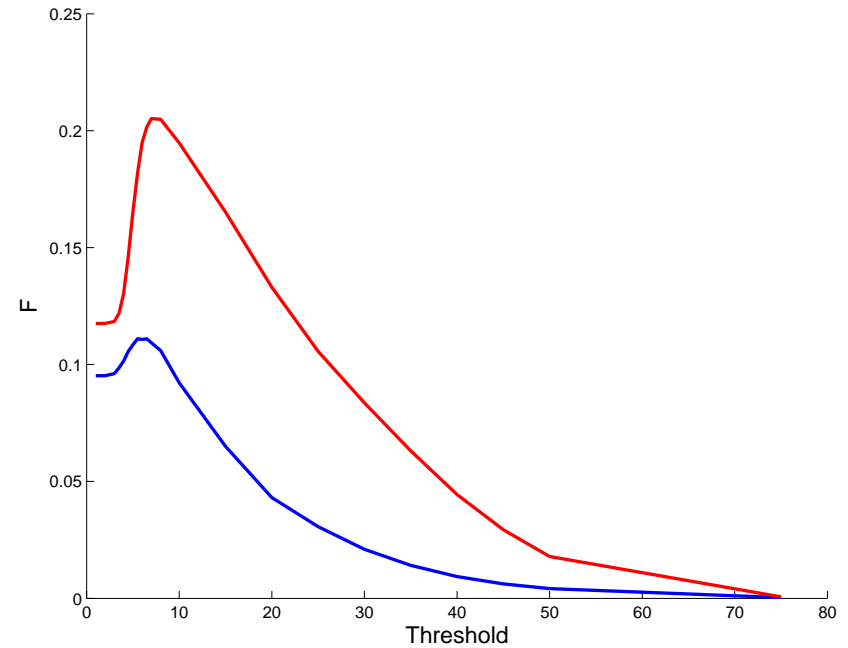
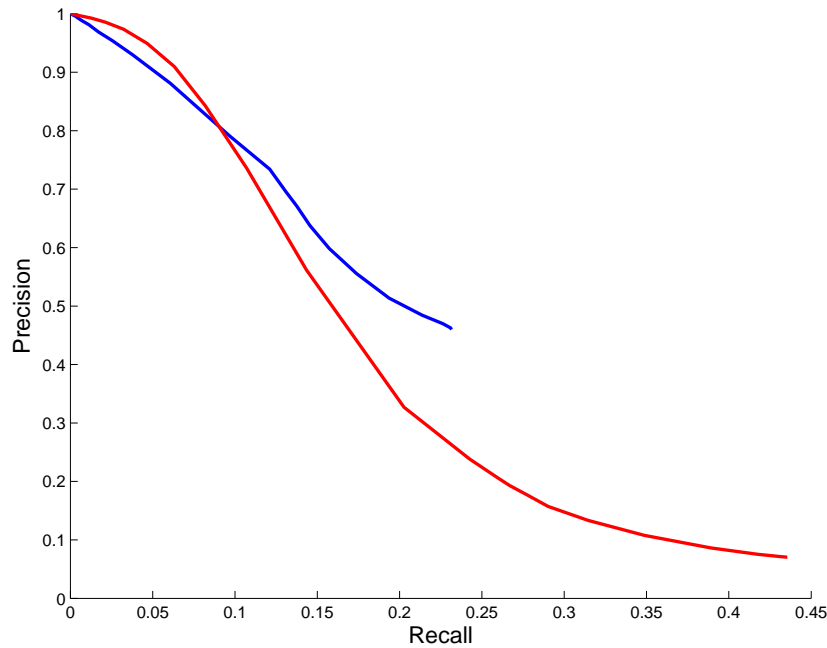
Evaluation as insight

Recall: $R = tp / (tp + fn)$, **Precision:** $P = tp / (tp + fp)$
 $F = \frac{1}{\alpha \frac{1}{P} + (1-\alpha) \frac{1}{R}}$, $\alpha = .5 \rightarrow F = \frac{2PR}{(R+P)}$ (the harmonic mean).

Macro vs. Micro-averaging

- ⑥ **Macro:** Equal weight to each category
 - △ Averaging done per **category**
- ⑥ **Micro:** Equal weight to each object
 - △ Averaging done per **decision**

Precision-Recall curves



10-NN, 100k training segments, 50k test segments
Macro-averaged, Micro-averaged

Where is this going?

Insight from kNN implementations should help us develop an understanding of the nature of the dataset:

- ⑥ why should we expect different weighting/ranking schemes to perform better or worse?
- ⑥ how can learning about the shortcoming of kNN imply that a hierarchical model might be an improvement?
- ⑥ why are certain category types easier or harder to classify and why?

A few references

- [1] *Foundations of Statistical Natural Language Processing*, Manning and Schütze. 1999. 529-544, 604-606.
- [2] *Information Retrieval, Data Structures and Algorithms*, Frakes and Baeza-Yates, 1992.
- [3] *Project homepage*, <http://www.math.umd.edu/~olsson/amsc663/>