Recommender Systems 2

Internet Analytics (COM-308)

Prof. Matthias Grossglauser
School of Computer and Communication Sciences

EPFL

Overview

- Content-based recommenders:
  - Here, content=text (prose in a news article, user-provided tags for music, reviews of a product…)
- Vector space model
  - Each dimension ~ one term (word)
- TF-IDF metric:
  - Frequency in doc makes that word important in that doc
  - Frequency in many docs makes a word less important
- Probabilistic model for text classification
  - Naïve Bayes: every word is i.i.d. given class
- Smoothing:
  - Dealing with rare words not seen in training
  - Regularizer

Overview: recommender systems

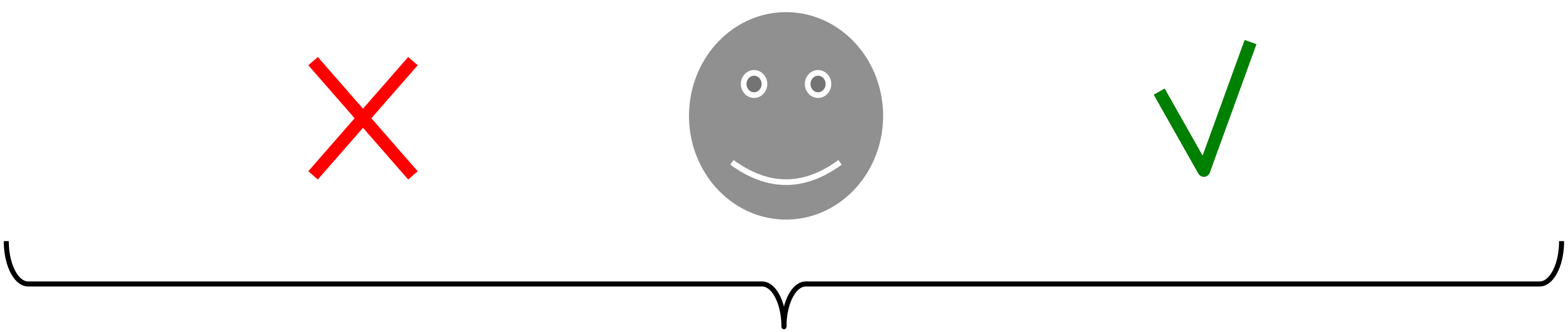
- Content-based recommenders

item 1:
"Plane hijacked…"

item 4:
"50.3% vote yes…"

item 2:
"soccer game…"

item 3:
"swiss skiers win…"



Model / user profile
new content → predicted rating

Basic idea

- Recommend to user u items similar to the ones he/she liked before
  - Collaborative filtering: similar item = liked by people who share u's tastes
  - Content-based: similar = with similar content features as previously liked items
- What features:
  - Context-dependent
  - Images&music: signal properties (rhythm, instruments,...); meta-information; tags;...
    - Pandora: music genome project, ~ 400 features
  - Text: easiest & most widespread
    - Prose, tags,...

Vector space model

- Compact description of a document
  - Ignores order – “bag of words”
- One dimension per term/word
  - Typically very sparse
- Count vector:
  - f_i = # of occurrences of word i in document
- Note: not reversible, ignores order of words
  - The meaning of a sentence would be lost on a human reader!
  - (a a be human lost meaning of on reader sentence the would!)

Profile from words

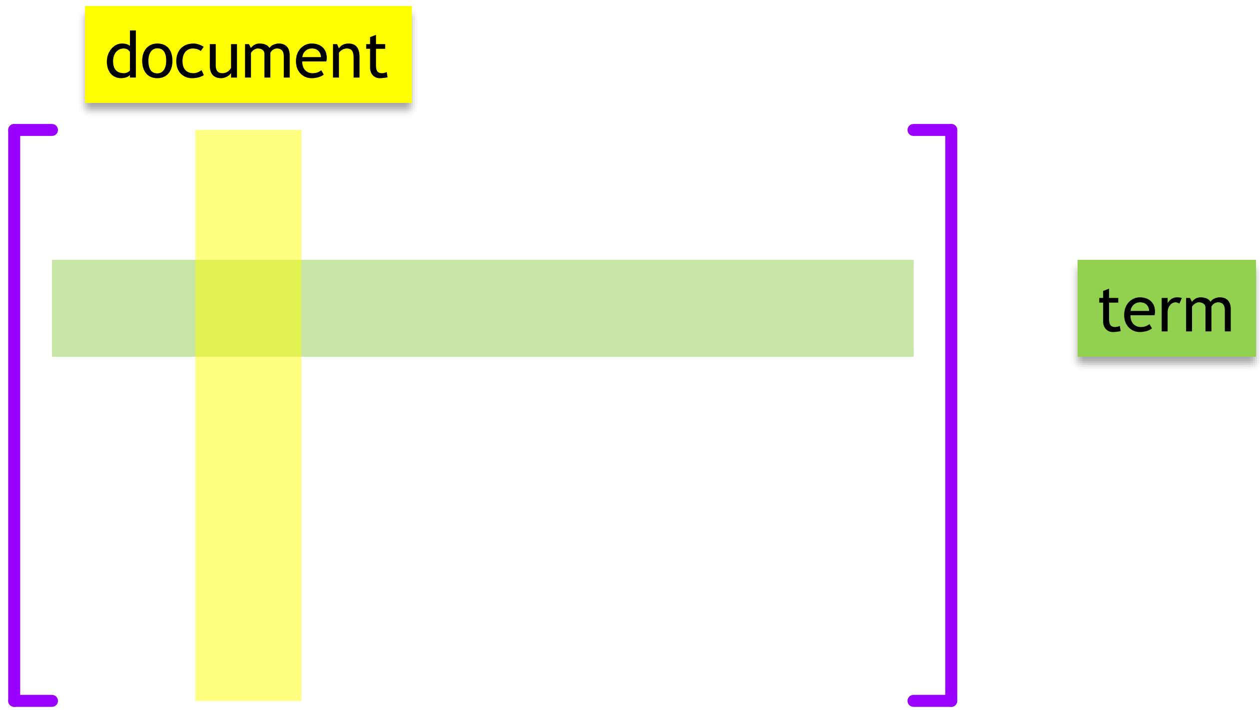
- How to create a useful profile of a document?
  - Frequent words are characteristic of “topic”
  - Document A: (“Probability”:50, “Markov”:20, “Poisson”:15,…)
  - Document B: (“Wimbledon”:30, “Federer”:8, “Nadal”:5,…)
- TF: Term Frequency
  - Function of one document j (not the whole corpus)
  - Def: f_{ij} = # of occurrences (frequency) of word i in doc j
  - Def: $TF_{ij} = \dfrac{f_{ij}}{\max_k f_{kj}}$
  - Importance of word i in document j

TF-IDF: A measure of word importance

- Problem:
  - Most frequent terms would be (in English): the, be, to, of, and, a, in, that, have, I, it, for, not, on, with, he, as, you, do, at,…
  - No information, because common to all docs
  - We want words that are frequent **only** in target docs
- IDF: Inverse Document Frequency
  - Function of whole corpus
  - Def: $n_i = \#$ documents j where word i occurs (at least once)
  - Def: $IDF_i = -\log_2 \frac{n_i}{N}$
  - If I know word i, number of bits of information I learn about which document is the target within corpus

TF-IDF vector space model

- Document profile D within a corpus:
  - $TFIDF_{ij} = TF_{ij} \times IDF_i$
  - $=$



  - High score: word frequent in this document, but not in most other documents

TF-IDF vector space model

- Vectors are high-dimensional but sparse
- Refinements: text preprocessing
  - Remove stop words: the, be, to, of, and, a,…
  - Stemming & lemming: transforming
    - “the boy's cars are different colors” ->
      “the boy car be differ color” [Manning et al.]
  - Vector cutoff to most important terms
  - Allow multi-word (“multi-gram”) terms (“United States”)

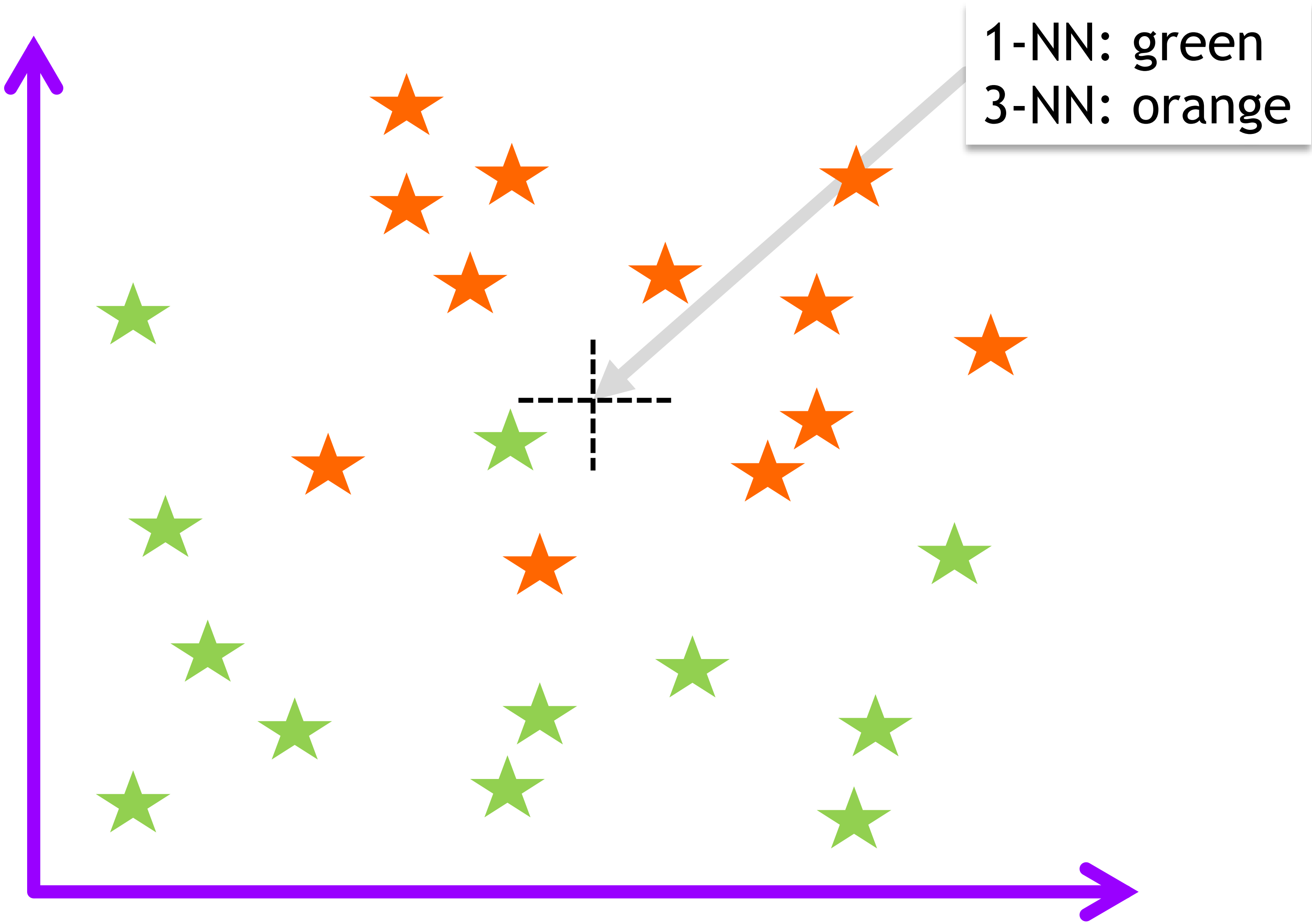
Queries and recommendations

- User profile (query) Q:
  - Explicit: e.g., declaring an interest ("north korea")
  - Implicit: ratings (e.g., thumbs up/down)
- Explicit:
  - These models are from information retrieval:
    - Searching by query: return most similar docs to query
    - Query terms → TF-IDF vector Q
- Assumption:
  - Likelihood that user profile Q likes document D: $sim(Q, TFIDF_{*,D})$, where
  - Usually:
    - $sim(x, y) = \cos(\sphericalangle x, y)$

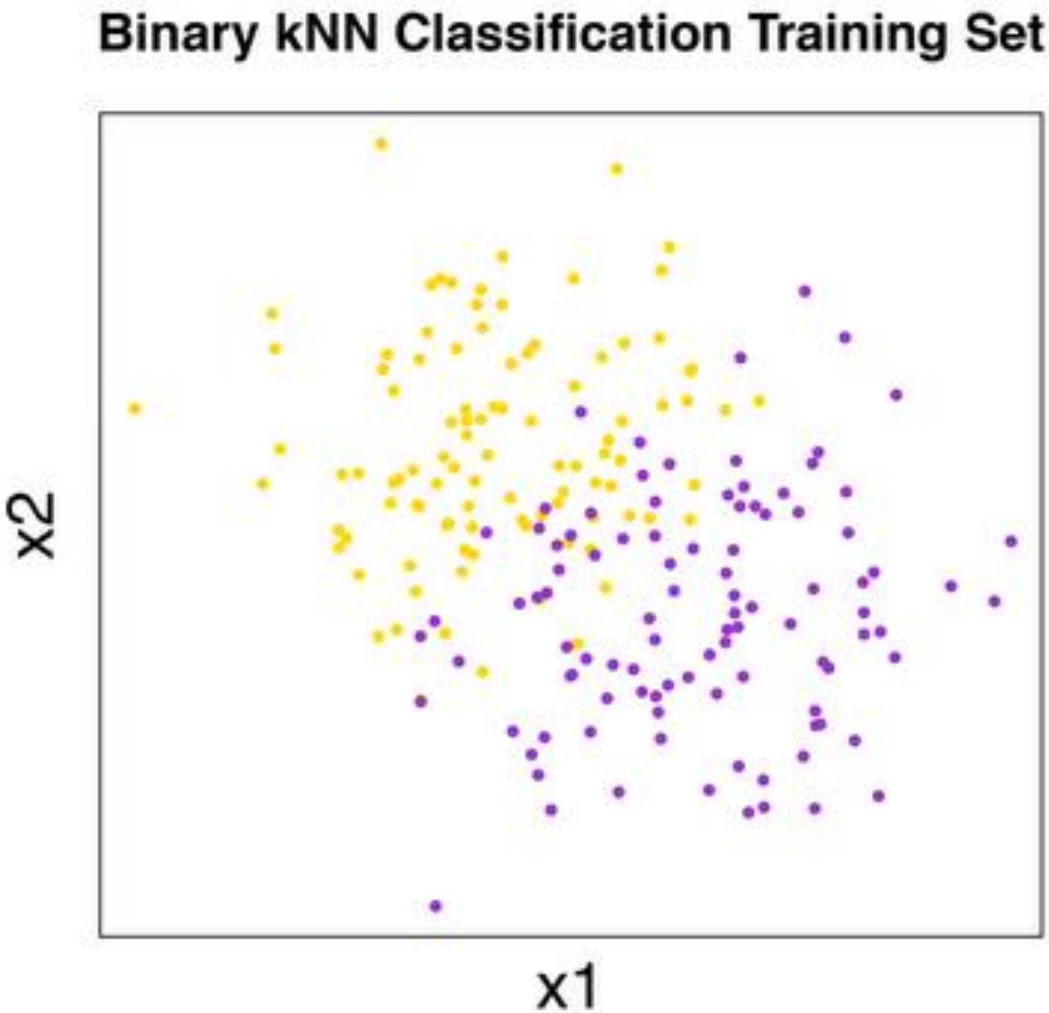
From queries to ratings

- Implicit: user rates documents rather than queries:
  - Treat highly rated/liked docs as “positive queries”, low rated/not liked as “negative queries”
  - Past ratings are “green/red” points in a high-dimensional vector space
- How to rate a new document D?
  - Classification problem: many methods
  - Generic non-parametric method: kNN (k nearest neighbors)
  - Select k rated docs in Q closest to D according to $sim(Q, D)$; majority in this set is predictor

kNN classifier

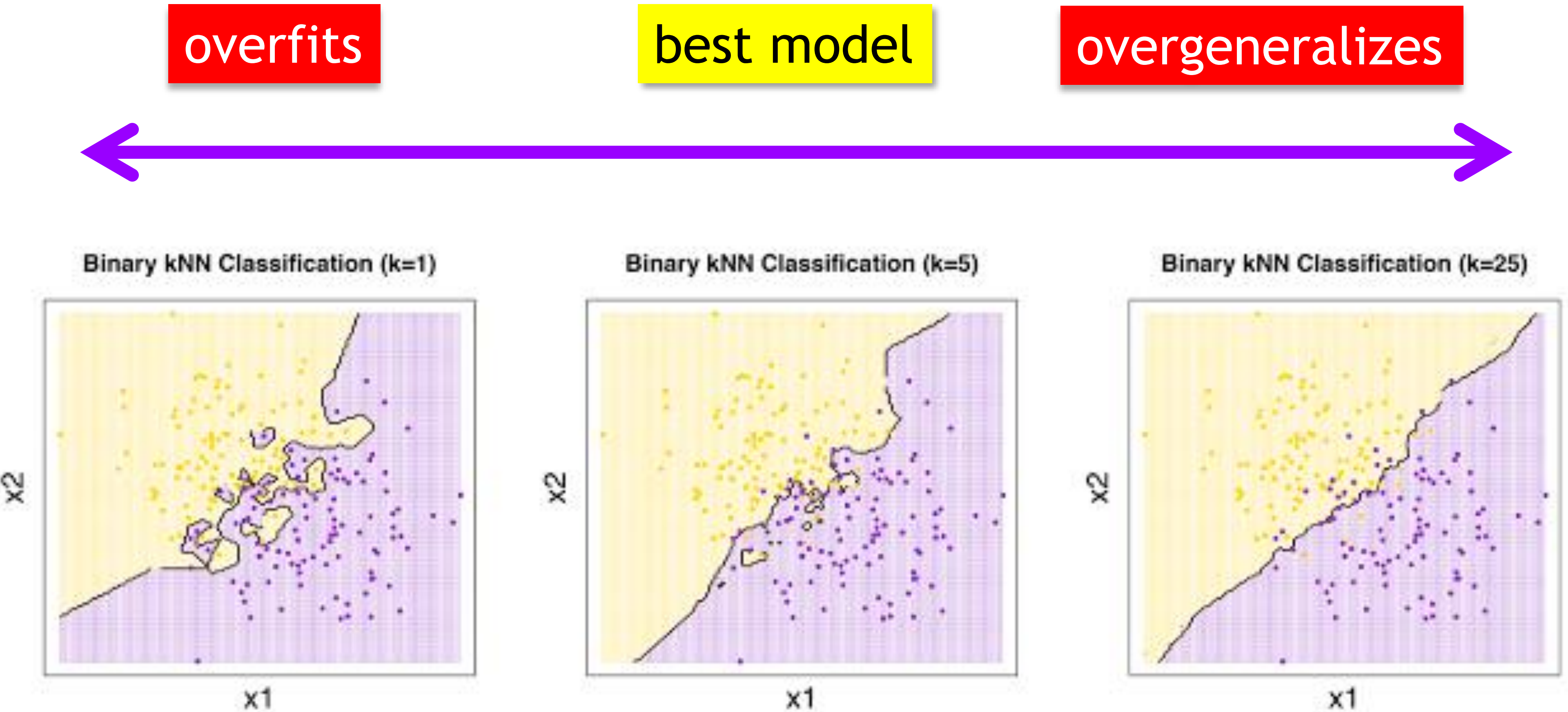


kNN classifier: selecting *k*



[Burton DeWilde: Data Science Rules (datasciencerules.blogspot.com), Oct 2012]

kNN: impact of k



[Burton DeWilde: Data Science Rules (datasciencerules.blogspot.com), Oct 2012]

Critique of vector-space approach

- Assumptions implicit in approach
  - “small angle between TF-IDF vectors means document close to query”: intuitively ok
  - Quantities do not have “physical meaning”, purely heuristic
- We would like a clean model: assumptions, performance measure we can optimize & compare
  - Probabilistic model: rigorous treatment of uncertainty

Probabilistic models

- Significant uncertainty in predictions
  - Quantization effects: like/dislike -> how much?
  - Context: e.g.: dislike right now (mood), or dislike categorically?
  - Errors, confusions, etc.
- Uncertainty → model explicitly as probability
  - Make assumptions explicit
  - Easier to interpret significance
  - Result comes with measure of uncertainty (confidence interval, etc.)

Bayesian inference

- Statistical inference: frequentist (non-Bayesian)
  - Observation X
  - Model: $p_\theta(x)$: distribution of X, depending on hidden parameter θ
  - Goal: infer θ from observation(s) of X
  - Maximum Likelihood estimator: $\hat{\theta} = \max_\theta p_\theta(X)$
    - Estimated parameter best explains observed data

Bayesian inference

- Statistical inference: Bayesian
  - We know something about θ: prior knowledge about the problem
  - θ is a random variable with a known distribution: prior
  - Model: $p(X|\theta)$: distribution of X, conditional on hidden random variable θ
  - Bayes' rule:

$$P(\theta|X) = \frac{P(\theta, X)}{P(X)} = \frac{P(X|\theta)P(\theta)}{\sum_{\theta'} P(X|\theta')P(\theta')}$$

  - Maximum A Posteriori (MAP) estimator:

$$\hat{\theta} = \max_{\theta} P(\theta|X)$$

  - But the full posterior distribution $P(\theta|X)$ carries additional information!
    - How certain/uncertain are we about θ given data X

Example: Max-Likelihood vs Bayesian

- Medical test
  - You take a medical test whose accuracy is 90% - that is, prob. test gives right result = 0.9
  - Frequentist:
    - $P(pos|sick) = 0.9$; $P(pos|healthy) = 0.1$
    - ML: $X = \text{pos} \rightarrow \hat{\theta} = \text{sick}$
    - Test comes back positive → you conclude you are sick

Example: ML vs Bayesian

- Medical test:
  - Bayesian:
    - Medical test; prior = one in a million: $P(sick) = 10^{-6}$
    - If test comes back positive:
    - $P(sick|pos) = \frac{P(pos|sick)P(sick)}{P(pos|sick)P(sick)+P(pos|healthy)P(healthy)}$
    - $P(sick|pos) \cong 0.9 \times 10^{-5}$
    - You conclude you are very likely healthy!
  - Watch out: doctors apparently do not always get this intuitively right

Naïve Bayes classifier

- Need a probabilistic model for a document
- Simplest model:
  - Naïve = independent terms (features)
  - Each word is generated according to i.i.d. distribution

$$P(X_1, \dots, X_n | \theta) = \prod_i P(X_i | \theta)$$

- Hidden variable θ:
  - Relevant (good, G) or not relevant (bad, B)
- Observable variable:
  - Message = set of words $(x_1, x_2, \dots, x_n)$
- Classify message into (G, B)
- Model $p(X|\{G, B\}), p(\{G, B\})$:
  - Learn from data

Example: naïve Bayes classifier learning

- Training set:

Get nice watch	Cheap replica watch
New York rocks!	New cheap loan
Watch for rocks	Get lottery million
	Million dollar watch

- Prior: $P(\theta = G) = \frac{3}{7}; P(\theta = B) = \frac{4}{7}$
- Conditional word distributions $P(X|\theta)$:

X	Get	nice	watch	new	york	rocks	for	cheap	replica	loan	lottery	million	dollar	perfect
$9 \times P(X\|G)$	1	1	2	1	1	2	1	0	0	0	0	0	0	0
$12 \times P(X\|B)$	1	0	2	1	0	0	0	2	1	1	1	2	1	0

Example: naïve Bayes classifier

- Classifying sentences $M = (X_1, X_2, X_3, \dots)$:
  - «get new watch»:

$$P(G|M) =$$

$$= \frac{P(X_1|G)P(X_2|G)P(X_3|G)P(G)}{P(X_1|G)P(X_2|G)P(X_3|G)P(G) + P(X_1|B)P(X_2|B)P(X_3|B)P(B)} =$$

$$= \frac{9^{-3} \cdot 1 \cdot 1 \cdot 2 \cdot 3/7}{9^{-3} \cdot 1 \cdot 1 \cdot 2 \cdot \frac{3}{7} + 12^{-3} \cdot 1 \cdot 1 \cdot 2 \cdot 4/7} = 0.64$$

X	Get	nice	watch	new	york	rocks	for	cheap	replica	loan	lottery	million	dollar	perfect
$9 \times P(X\|G)$	1	1	2	1	1	2	1	0	0	0	0	0	0	0
$12 \times P(X\|B)$	1	0	2	1	0	0	0	2	1	1	1	2	1	0

Example: naïve Bayes classifier

- Classifying sentences $M = (X_1, X_2, X_3, \dots)$:
  - «cheap replica rocks»:

$$P(G|M) =$$

$$= \frac{\boxed{=0}\; P(X_3|G)P(G)}{\boxed{=0}\; P(X_3|G)P(G) + P(X_1|B)P(X_2|B)\boxed{=0}\, P(B)}$$

- Undefined!

X	get	nice	watch	new	york	rocks	for	cheap	replica	loan	lottery	million	dollar	perfect
$9 \times P(X\|G)$	1	1	2	1	1	2	1	0	0	0	0	0	0	0
$12 \times P(X\|B)$	1	0	2	1	0	0	0	2	1	1	1	2	1	0

Problem with unseen training terms

- Sparsity problem:
  - If alphabet of words is large w.r.t. training set, there are some words x we never see (e.g., $x =$"mesonoxian")
    - Estimate: $P(\text{mesonoxian}|\{G, B\}) = 0$
  - If target message contains "mesonoxian":

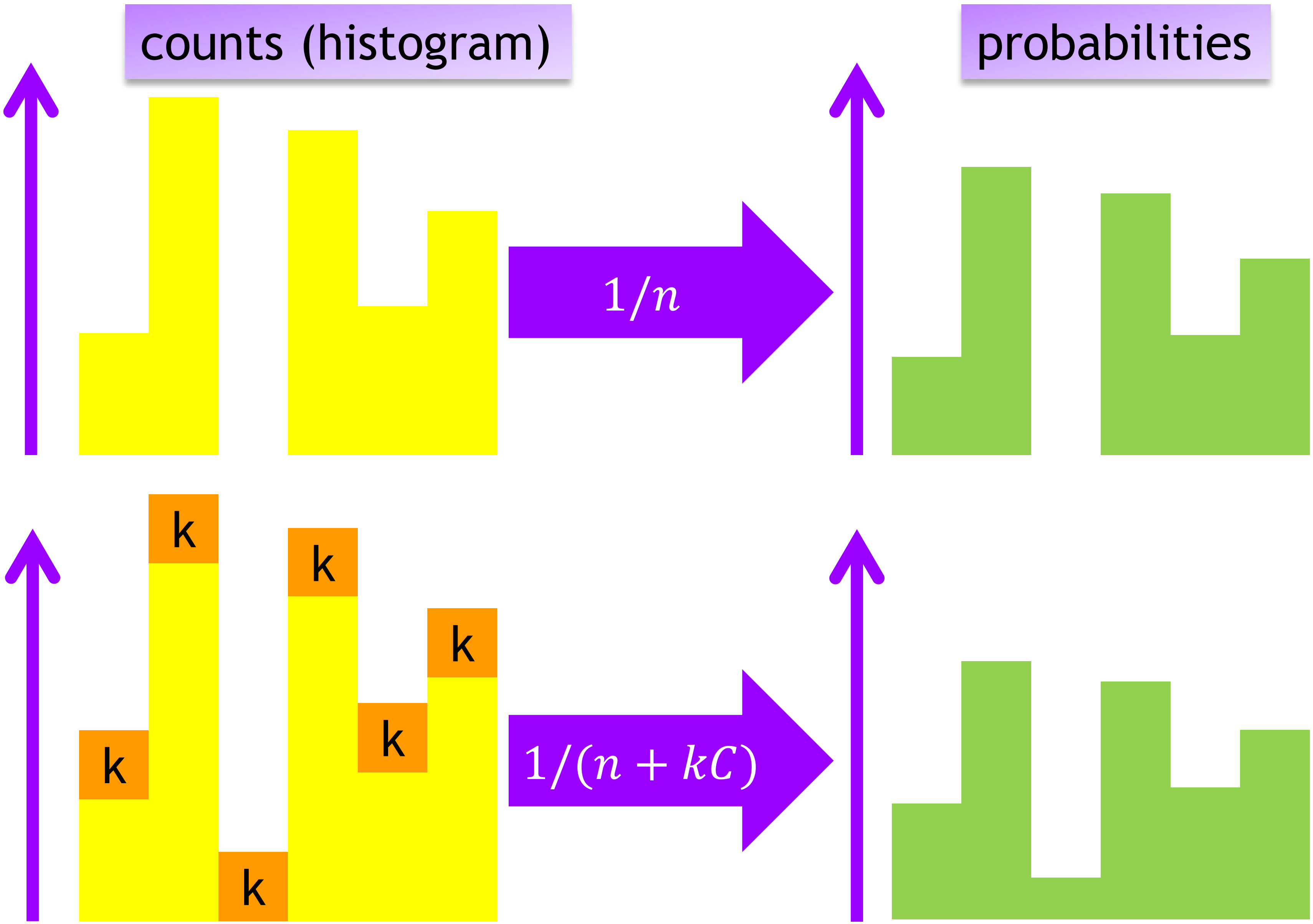
$$P(\{G, B\}) = \frac{P(x|\theta)P(\theta)}{\sum_{\theta'} P(x|\theta')P(\theta')} = \frac{0}{0}$$

- Problem:
  - We estimate a distribution from a very small set of samples – a form of overfitting
  - How to correctly estimate very rare words?
- Approach 1:
  - Ignore unseen words → simple, but crude; throws away information

Laplace smoothing

- Idea: assume every word occurs at least once
  - Aka “additive smoothing”, “add-one smoothing”
- Bias towards uniform distribution
  - A form of regularization
- Estimate of a distribution over domain $D = \{1, \ldots, C\}$ from data set $\{x_1, x_2, \ldots, x_n\}$
  - Unsmoothed: $p(X = x) = \frac{|\{x_i : x_i = x\}|}{n}$ (n=# samples)
  - Smoothed: assume k “fake” observations for each class
$$p(X = x) = \frac{|\{x_i : x_i = x\}| + k}{n + kC}$$
  - Empty dataset ($n = 0$) → $P(X|\theta)$ uniform
  - Large dataset ($n \gg 1$) → smoothed $P(X|\theta) \cong$ unsmoothed $P(X|\theta)$

Laplace smoothing



Example: Laplace-smoothed classifier

- Sentence M =«cheap replica rocks»:

$$P(G|M) =$$

$$= \frac{P(X_1|G)P(X_2|G)P(X_3|G)P(G)}{P(X_1|G)P(X_2|G)P(X_3|G)P(G) + P(X_1|B)P(X_2|B)P(X_3|B)P(B)} =$$

$$= \frac{23^{-3} \cdot 1 \cdot 1 \cdot 3 \cdot 4/9}{23^{-3} \cdot 1 \cdot 1 \cdot 3 \cdot 4/9 + 26^{-3} \cdot 3 \cdot 2 \cdot 1 \cdot 5/9} = 0.37$$

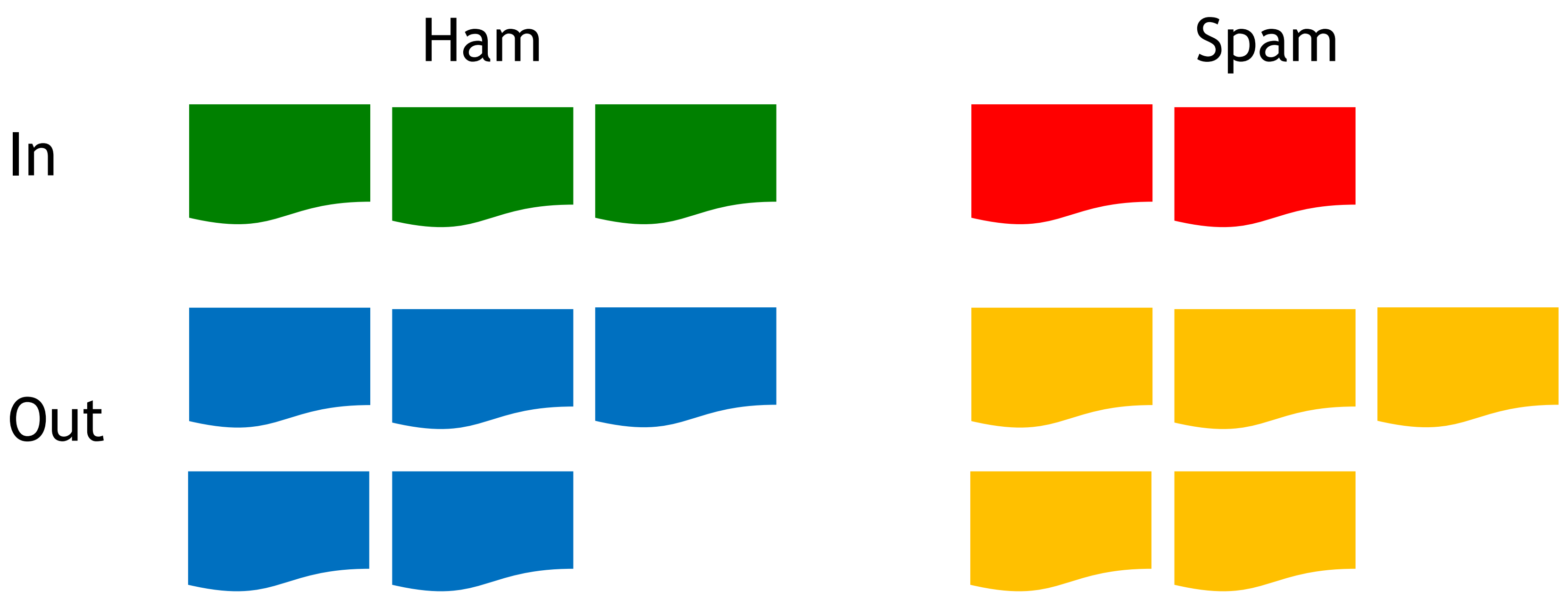
- Advantages:
  - We can compute an estimate for any message
  - For small training sets → avoids overfitting

Note: we smoothed the prior as well: (3/7,4/7)→(4/9,5/9)

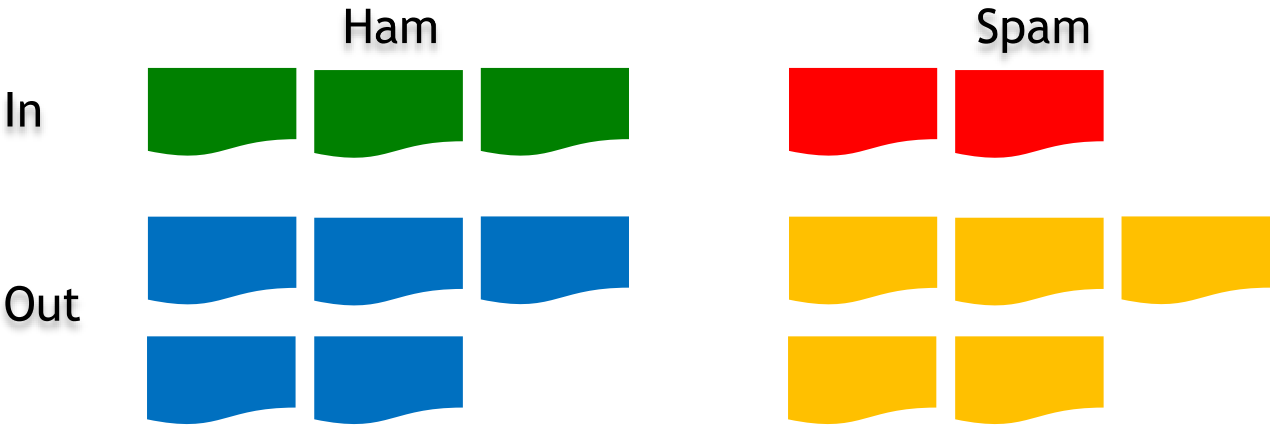
X	Get	nice	watch	new	york	rocks	for	cheap	replica	loan	lottery	million	dollar	perfect
$23 \times P(X\|G)$	2	2	3	2	2	3	2	1	1	1	1	1	1	1
$26 \times P(X\|B)$	2	1	3	2	1	1	1	3	2	2	2	3	2	1

Precision and recall

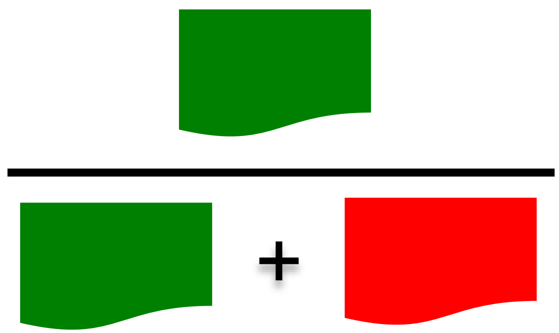
- Problem: how to set the threshold for ham/spam?
  - Too restrictive: ham gets deleted
  - Too permissive: spam gets through
- Information-retrieval performance metrics:
  - Precision: % of search results that are ham vs spam
  - Recall: % of all ham that are in search results



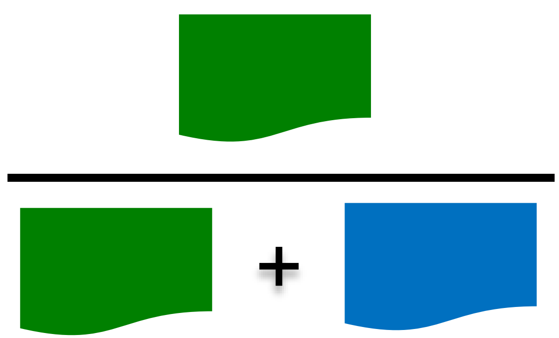
Precision and recall



- Precision =



- Recall =



Critique of methods so far

- Both models treat any two words as completely independent signals
- But language has a lot of ambiguity and overlap:
  - Two words can mean something very similar:
    - "happy" vs "joyful", "rich"vs "wealthy"
  - One word can mean different things:
    - "match": soccer game or a device to light a fire
    - "right": opposite of left or correct
- Approaches we saw today do not learn and exploit these relationships
- Next week: word embeddings → map words in low-dimensional feature space

RecSys: content vs collaborative

Pros of content-based
Independent of other users → no cold start problem for new items (item comes with features)
Independent of other users → can recommend for unique tastes, no "trend to average"
Can provide explanation for recommendation (e.g., matching keywords)

Cons of content-based
Multimedia etc.: hard to identify features
Independent of other users → no discovery or "surprises"
Cold start problem for new user

- In practice: combination
  - Lack of ratings, few users → rely more on content
  - Lots of users, few tags → collaborative

Summary

- Content: text, tags, user comments, subtitles,…
- Collaborative filtering vs content-based:
  - Blind to content vs blind to other users
- Classical approaches from information retrieval:
  - Vector space models, similarity metrics
- More modern probabilistic approaches from ML:
  - Naïve Bayes, language models (n-grams), word embeddings
- Other application for naïve Bayes: spam filtering
  - $P(B) \cong 0.8 \ldots 0.9$

References

- [C Aggarwal: Content-Based Recommender Systems, 2016]
- [P Lops, M de Gemmis, G Semeraro: Content-based Recommender Systems: State of the Art and Trends, 2011]
- [A. Rajaraman, J. D. Ullman: Mining of Massive Datasets, Cambridge, 2012 (chapter 9)]
- [S. Russell, P. Norvig: Artificial Intelligence – A Modern Approach (3rd ed), Pearson, 2010 (chapter22)]
- [W. B. Croft, D. Metzler, T. Strohman: Search Engines – Information Retrieval in Practice, Addison Wesley, 2010 (chapters 7&10)]