



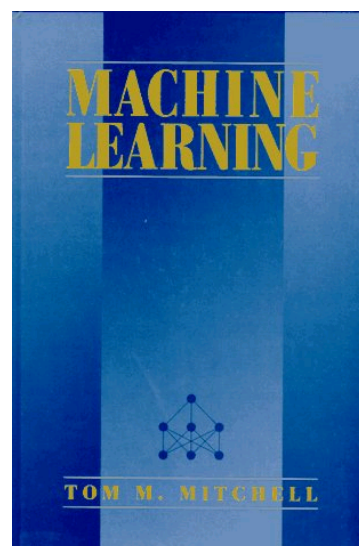
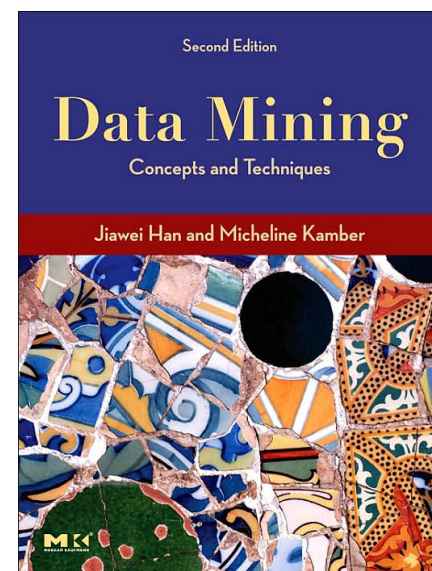
Information Retrieval & Text Mining

Data Mining and Text Mining (UIC 583 @ Politecnico di Milano)

References

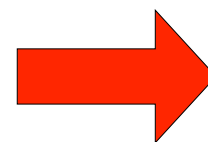
- Jiawei Han and Micheline Kamber, "Data Mining: Concepts and Techniques", The Morgan Kaufmann Series in Data Management Systems (Second Edition)
 - ▶ Chapter 10

- Tom M. Mitchell. "Machine Learning" McGraw Hill 1997
 - ▶ Chapter 6



Introduction

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	FALSE	No
Sunny	Hot	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Rainy	Mild	High	FALSE	Yes
Rainy	Cool	Normal	FALSE	Yes
Rainy	Cool	Normal	TRUE	No
Overcast	Cool	Normal	TRUE	Yes
Sunny	Mild	High	FALSE	No
Sunny	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	FALSE	Yes
Sunny	Mild	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Rainy	Mild	High	TRUE	No



Play?

Introduction



Sorry mom, no flowers this year. Flowers might be the Mother's Day gift of choice as recession fears and soaring oil and gas prices are forcing Americans to curb their spending.



Russia investors are finding it easy to turn a blind eye to the expulsion of U.S. diplomats as the country's increasingly bellicose rhetoric over Georgia continues.



Video games don't create killers, new book says. SAN FRANCISCO (Reuters) - Playing video games does not turn children into violent, blood-thirsty super-killers, according to a new book by a pair of Harvard researchers.

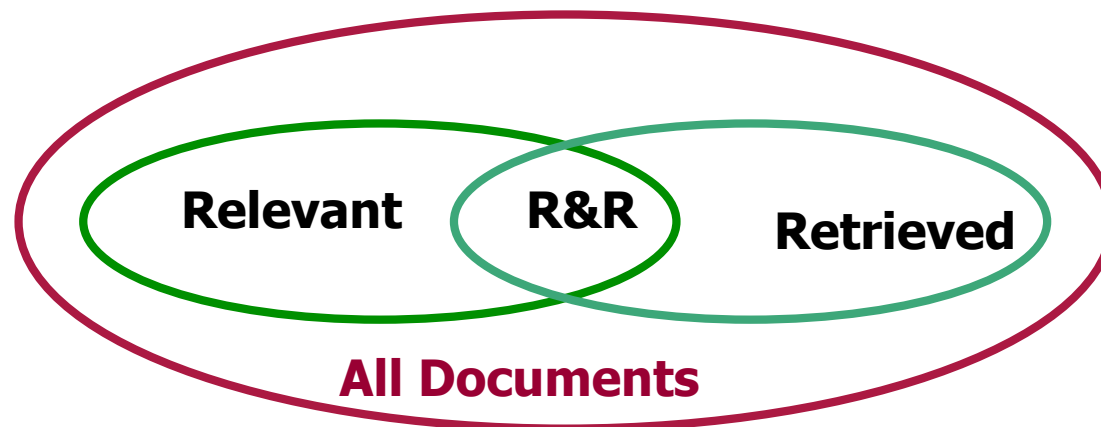


Interesting?

Information Retrieval

- ❑ IR deals with the problem of locating **relevant** documents with respect to the user **input** or **preference**
- ❑ Typical IR systems
 - ▶ Online library catalogs
 - ▶ Online document management systems
- ❑ Typical IR issues
 - ▶ management of unstructured documents
 - ▶ approximate search
 - ▶ relevance
- ❑ Main IR approaches
 - ▶ “**pull**” for short-term information need
 - ▶ “**push**” for long-term information need (e.g., recommender systems)

Basic Measures for Text Retrieval



$$precision = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Retrieved\}|}$$

$$recall = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Relevant\}|}$$

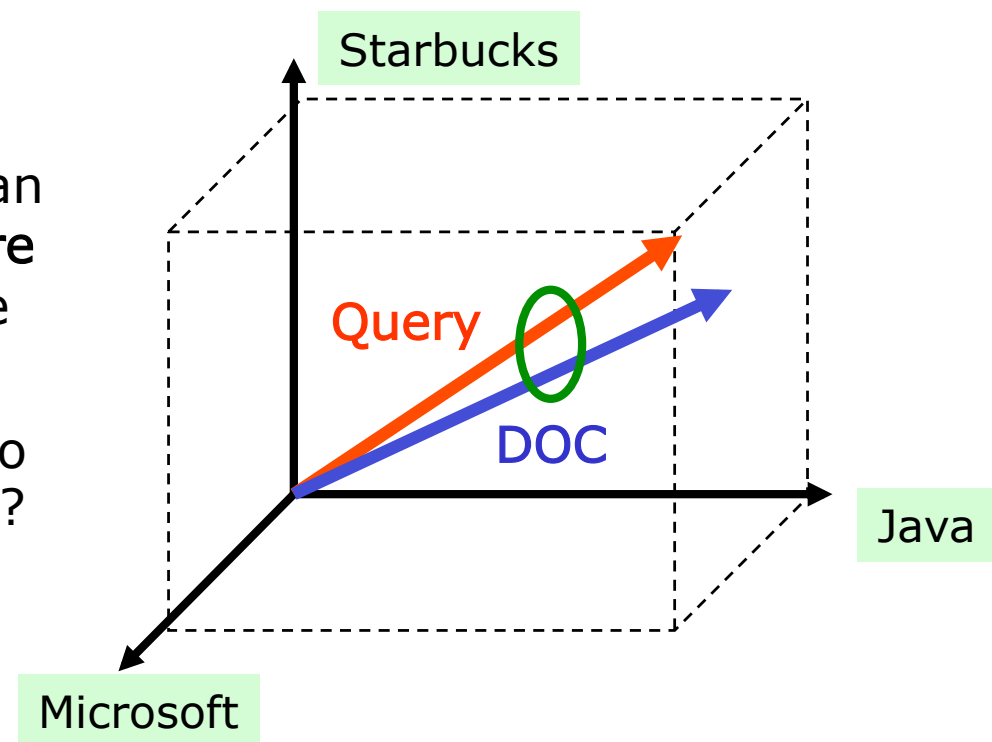
$$F_{score} = \frac{recall \times precision}{(recall + precision) / 2}$$

- ❑ **Document Selection (keyword-based retrieval)**
 - ▶ Query defines a set of requisites
 - ▶ Only the documents that satisfy the query are returned
 - ▶ A typical approach is the **Boolean Retrieval Model**
- ❑ **Document Ranking (similarity-based retrieval)**
 - ▶ Documents are ranked on the basis of their relevance with respect to the user query
 - ▶ For each document a “degree of relevance” to the query is measured
 - ▶ A typical approach is the **Vector Space Model**

- ❑ A query is composed of keywords linked by the three logical connectives: **not**, **and**, **or**
 - ▶ E.g.: “car and repair”, “plane or airplane”
- ❑ In the Boolean model each document is either relevant or non-relevant, depending it matches or not the query
- ❑ Limitations
 - ▶ Generally not suitable to satisfy information need
 - ▶ Useful only in very specific domain where users have a big expertise

Vector Space Model

- ❑ A document and a query are represented as **vectors** in high-dimensional space corresponding to all the **keywords**
- ❑ Relevance is measured with an appropriate **similarity measure** defined over the vector space
- ❑ Issues:
 - ▶ How to select keywords to capture “basic concepts” ?
 - ▶ How to assign weights to each term?
 - ▶ How to measure the similarity?



- ❑ Text is preprocessed through **tokenization**
- ❑ **Stop list** and **word stemming** are used to identify significant keywords
 - ▶ Stop List
 - e.g. "a", "the", "always", "along"
 - ▶ Word stemming
 - e.g. "computer", "computing", "computerize" => "compute"

❑ Term Frequency (TF)

- ▶ Computed from frequency of a term t in a document d
- ▶ More frequent a term is \rightarrow more relevant it is

❑ Inverse Document Frequency (IDF)

$$\text{IDF}(t) = \log \frac{|D|}{|D_t|}$$

- ▶ D is the documents collection, D_t is the subset of D that contains t
- ▶ Less frequent among documents \rightarrow more discriminant
 - e.g., “database” in a collection of papers on DBMS

❑ Mixing TF and IDF

$$\text{TF-IDF}(d, t) = \text{TF}(d, t) \times \text{IDF}(t)$$

How to Measure Similarity?

- Given two documents (or a document and a query)

$$D_i = (w_{i1}, w_{i2}, \dots, w_{iN}) \quad D_j = (w_{j1}, w_{j2}, \dots, w_{jN})$$

- Similarity definition

- ▶ dot product

$$Sim(D_i, D_j) = \sum_{t=1}^N w_{it} * w_{jt}$$

- ▶ normalized dot product (or cosine)

$$Sim(D_i, D_j) = \frac{\sum_{t=1}^N w_{it} * w_{jt}}{\sqrt{\sum_{t=1}^N (w_{it})^2 * \sum_{t=1}^N (w_{jt})^2}}$$

- ❑ Approaches presented so far involves high dimensional space (huge number of keywords)
 - ▶ Computationally expensive
 - ▶ Difficult to deal with **synonymy** and **polysemy** problems
 - “vehicle” is similar to “car”
 - “mining” has different meanings in different contexts
- ❑ Dimensionality reduction techniques
 - ▶ Latent Semantic Indexing (LSI)
 - ▶ Locality Preserving Indexing (LPI)
 - ▶ Probabilistic Semantic Indexing (PLSI)

Latent Semantic Indexing (LSI)

- Let x_i be vectors representing documents and X (term frequency matrix) the all set of documents:

$$\vec{x}_1, \dots, \vec{x}_n \in R^m \quad X = [\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n]$$

- Let use the **singular value decomposition (SVD)** to reduce the size of frequency table:

$$X = U\Sigma V^T$$

- Approximate X with X_k that is obtained from the first K vectors of U
- It can be shown that such transformation minimizes the error for the reconstruction of X

Locality preserving Indexing (LPI)

- Goal is preserving the **locality** information
 - ▶ Two documents close in the original space should be close also in the transformed space
- More formally

$$\vec{x}_1, \dots, \vec{x}_n \in R^m$$

Set of Documents

$$S \in R^{n \times m}$$

Similarity Matrix

$$\vec{a}^* = \operatorname{argmin}_{\vec{a}} \sum_{i,j} (\vec{a}^T \vec{x}_i - \vec{a}^T \vec{x}_j)^2 S_{ij} \quad \longrightarrow \quad X' = \vec{a}^* \vec{a}^{*T} X$$

Optimal transformation

- Similarity matrix:

$$S_{ij} = \begin{cases} \frac{\vec{x}_i^T \vec{x}_j}{\|\vec{x}_i^T \vec{x}_j\|}, & \text{if } \vec{x}_i \text{ is in the } p \text{ nearest neighbors of } \vec{x}_j \text{ or viceversa} \\ 0, & \text{otherwise} \end{cases}$$

Probabilistic Latent Semantic Indexing (PLSI)

- ❑ Similar to LSI but does not apply SVD to identify the k most relevant features
- ❑ Assumption: all the documents have k common “themes”
- ❑ Word distribution in documents can be modeled as

$$p_{d_i}(w) = \sum_{j=1}^k [\pi_{d_i,j}] p(w|\theta_j)$$

The equation $p_{d_i}(w) = \sum_{j=1}^k [\pi_{d_i,j}] p(w|\theta_j)$ is shown. A blue circle highlights the term $\pi_{d_i,j}$, and a red circle highlights the term $p(w|\theta_j)$. A yellow callout box labeled "Mixing weights" points to the blue circle, and another yellow callout box labeled "Theme distributions" points to the red circle.

- ❑ Mixing weights are identified with Expectation-Maximization (EM) algorithms and define new representation of the documents

Text Mining

- ❑ Text Mining aims to **extract useful knowledge from text documents**
- ❑ Approaches
 - ▶ **Keyword-based**
 - Relies on IR techniques
 - ▶ **Tagging**
 - Manual tagging
 - Automatic categorization
 - ▶ **Information-extraction**
 - Natural Language Processing (NLP)
- ❑ Tasks
 - ▶ **Keyword-Based Association Analysis**
 - ▶ **Document Classification**

Gov. Schwarzenegger helps inaugurate pricey new bridge approach

The Associated Press

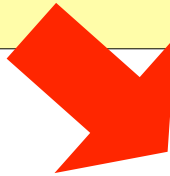
Article Launched: 04/11/2008 01:40:31 PM PDT

SAN FRANCISCO—It briefly looked like a **scene** out of a "**Terminator**" **movie**, with **Governor Arnold Schwarzenegger** standing in the middle of San Francisco wielding a blow-torch in his hands. Actually, the **governor** was just helping to inaugurate a new approach to the San Francisco-Oakland Bay **Bridge**.

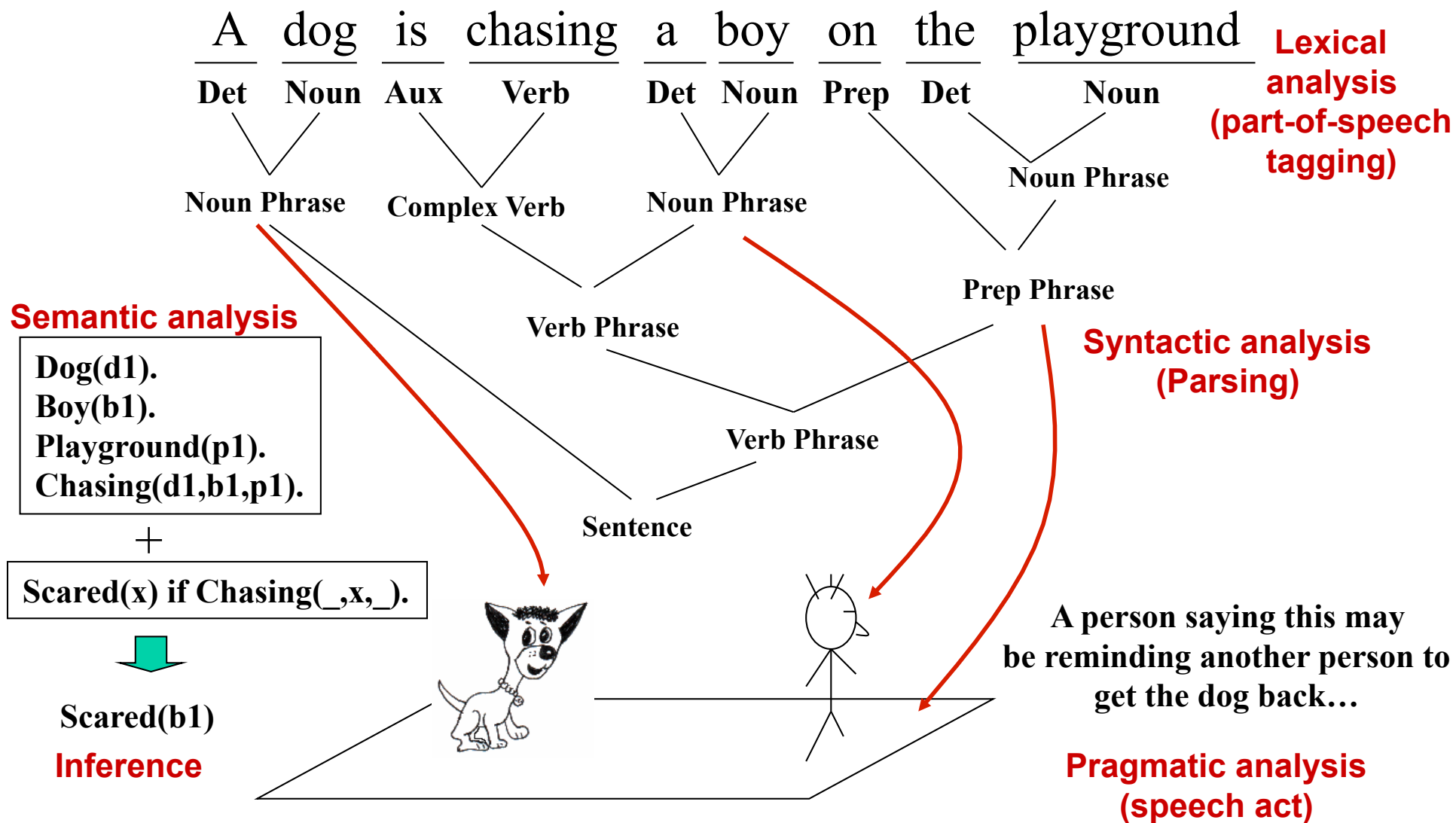
Caltrans thinks the new approach will make it faster for commuters to get on the **bridge** from the San Francisco side.

The new section of the highway is scheduled to open tomorrow morning and cost 429 million dollars to construct.

- ❑ Entertainment or Politics?
- ❑ Bag-of-tokens approaches have severe limitations

- 
- Schwarzenegger
 - Bridge
 - Caltrans
 - Governor
 - Scene
 - Terminator

Natural Language Processing



(Taken from ChengXiang Zhai, CS 397cxz – Fall 2003)

❑ Ambiguity

A man saw a boy with a telescope.

❑ Computational Intensity imposes a context horizon.

Text Mining NLP Approach

- ▶ Locate promising fragments using **fast IR methods** (bag-of-tokens)
- ▶ Only apply **slow NLP techniques** to promising fragments

- ❑ Aims to discover **sets of keywords** that occur frequently together in the documents
- ❑ Relies on the usual techniques for mining **associative and correlation rules**
- ❑ Each document is considered as a transaction of type
 $\{\text{document id, \{set of keywords\}}\}$
- ❑ Association mining may discover **set of consecutive or closely-located keywords**, called **terms or phrase**
 - ▶ **Compound** (e.g., $\{\text{Stanford,University}\}$)
 - ▶ **Noncompound** (e.g., $\{\text{dollars,shares,exchange}\}$)
- ❑ Once discovered the most frequent terms, **term-level mining** can be applied most effectively (w.r.t. single word level)

- ❑ Solve the problem of **labeling automatically text documents** on the basis of
 - ▶ Topic
 - ▶ Style
 - ▶ Purpose
- ❑ Usual classification techniques can be used to learn from a training set of **manually labeled documents**
- ❑ Which features? **Keywords** can be thousands...
- ❑ Major approaches
 - ▶ Similarity-based
 - ▶ Dimensionality reduction
 - ▶ Naïve Bayes text classifiers

- ❑ Exploits IR and k-nearest-neighbor classifier
 - ▶ For a new document to classify, the k most similar documents in the training set are retrieved
 - ▶ Document is classified on the basis of the class distribution among the k documents retrieved
 - Majority vote
 - Weighted vote
 - ▶ Tuning k is very important to achieve a good performance
- ❑ Limitations
 - ▶ **Space overhead** to store all the documents in training set
 - ▶ **Time overhead** to retrieve the similar documents

Dimensionality Reduction for Text Classification

- ❑ As in the **Vector Space Model** used for IR, the goal is to reduce the number of features to represents text
- ❑ Usual dimensionality reduction approaches in IR are based on the **distribution of keywords** among the **whole documents** database
- ❑ In text classification it is important to consider also the **correlation between keywords and classes**
 - ▶ Rare keywords have an high TF-IDF but might be uniformly distributed among classes
 - ▶ LSI and LPI do not take into account classes distributions
- ❑ Usual classification techniques can be then applied on reduced features space:
 - ▶ SVM
 - ▶ Bayesian classifiers

□ Definitions

- ▶ Category Hypothesis Space: $H = \{C_1, \dots, C_n\}$
- ▶ Document to Classify: D
- ▶ Probabilistic model:

$$P(C_i | D) = \frac{P(D | C_i)P(C_i)}{P(D)}$$

- We chose the class C^* such that

$$C^* = \arg \max_C P(C|D) = \arg \max_C P(D|C)P(C)$$

□ Issues

- ▶ Which features?
- ▶ How to compute the probabilities?

Naïve Bayes for Text (2)

- Features can be simply defined as the words in the document
- Let a_i be a keyword in the doc, and w_j a word in the vocabulary, we get:

$$P(D|C) = P(a_1 = w_{j_1}, a_2 = w_{j_2}, \dots, a_n = w_{j_n} | C)$$

Example

$H = \{\text{like}, \text{dislike}\}$

$D = \text{"Our approach to representing arbitrary text documents is disturbingly simple"}$

$$P(D|Like) = P(a_1 = \text{our}, a_2 = \text{approach}, \dots, a_n = \text{simple} | Like)$$

Naïve Bayes for Text (2)

- Features can be simply defined as the words in the document
- Let a_i be a keyword in the doc, and w_j a word in the vocabulary, we get:

$$P(D|C) = P(a_1 = w_{j_1}, a_2 = w_{j_2}, \dots, a_n = w_{j_n} | C)$$

- **Assumptions**

- ▶ Keywords distributions are inter-independent
- ▶ Keywords distributions are order-independent

$$P(D|C) = \prod_{i=1}^n P(w_{j_i} | C)$$

$$P(D|Like) = P(\text{our}|Like)P(\text{approach}|Like) \dots P(\text{simple}|Like)$$

Naïve Bayes for Text (3)

- ❑ How to compute probabilities?
 - ▶ Simply counting the occurrences may lead to wrong results when probabilities are small
- ❑ M-estimate approach adapted for text:

$$P(w_k|C) = \frac{N_{c,k} + 1}{N_c + |\text{Vocabulary}|}$$

- ▶ N_c is the whole number of word positions in documents of class C
- ▶ $N_{c,k}$ is the number of occurrences of w_k in documents of class C
- ▶ $|\text{Vocabulary}|$ is the number of distinct words in training set
- ▶ Uniform priors are assumed

Naïve Bayes for Text (4)

- Final classification is performed as

$$C^* = \arg \max_C P(C) \prod_{i=1}^n P(w_{j_i} | C)$$

- Despite its simplicity Naïve Bayes classifiers works very well in practice
- Applications
 - ▶ Newsgroup post classification
 - ▶ NewsWeeder (news recommender)

❑ Movie Review Data

- ▶ <http://www.cs.cornell.edu/people/pabo/movie-review-data/>

❑ Load Text Data in Weka (CLI)

Example directory layout for `TextDirectoryLoader`:

```
...
|
+- text_example
   |
   +- class1
      |
      + file1.txt
      |
      + file2.txt
      |
      ...
   +- class2
      |
      + another_file1.txt
      |
      + another_file2.txt
      |
      ...
```

The above directory structure can be turned into an ARFF file like this:

```
java weka.core.converters.TextDirectoryLoader -dir text_example > text_example.arff
```


- ❑ From text to word vectors: StringToWordVector
- ❑ Naïve Bayes Classifier
- ❑ Stoplist
 - ▶ <https://code.google.com/p/stop-words/>
- ❑ Attribute Selection