

Natural Language Processing

Lecture 3

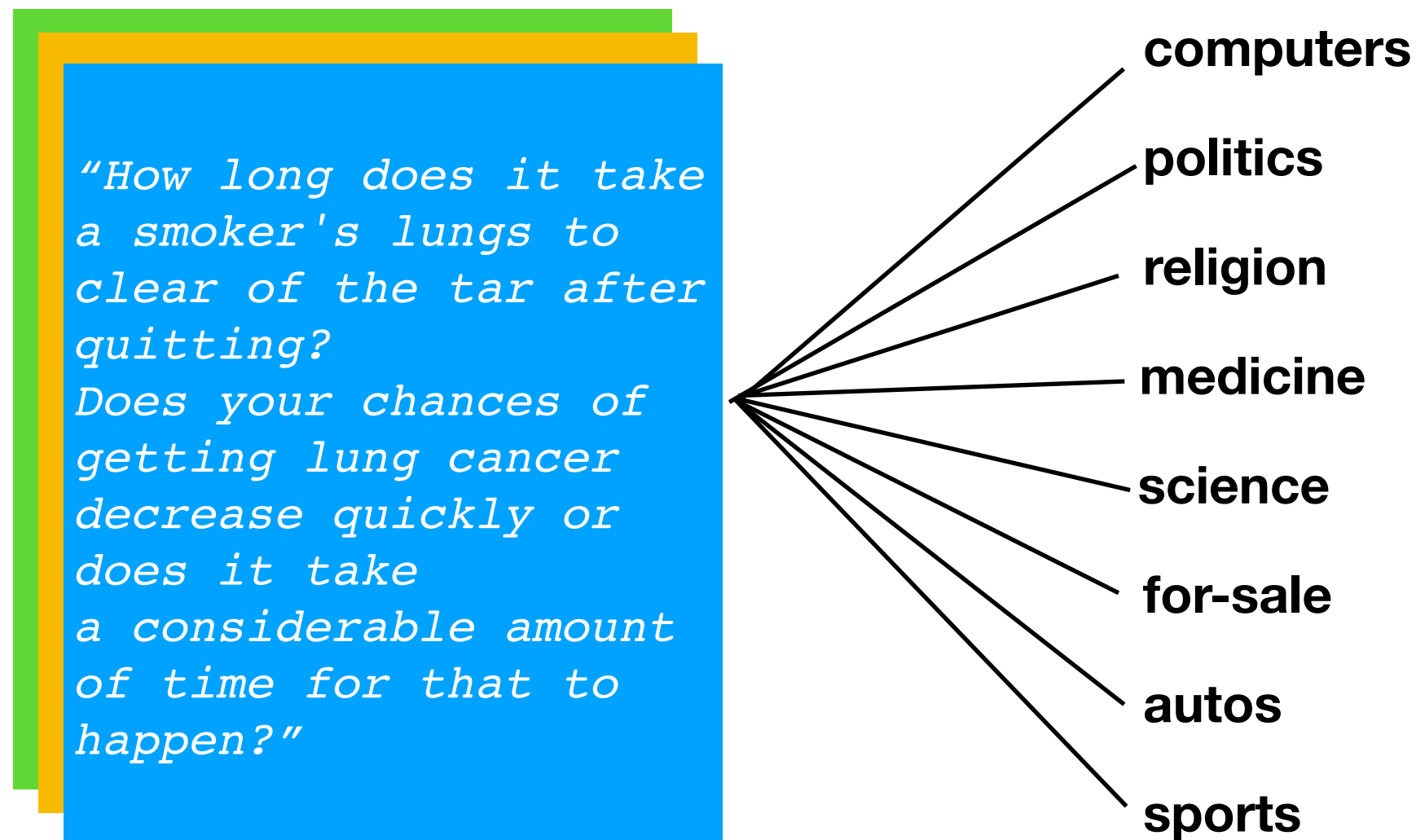
Naive Bayes' Classifier and Text Classification

9/13/24

COMS W4705
Daniel Bauer

Text Classification

- Given a representation of some document d , identify which class $c \in C$ the document belongs to.



From the 20-Newsgroups data set:

<http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html>

Text Classification

- Applications:
 - Spam detection.
 - Mood / Sentiment detection.
 - Author identification.
 - Identifying political affiliation.
 - Word Sense Disambiguation.
 - ...

Text Classification

- This is a machine learning problem.
- Can use different ML techniques.
 - **Supervised ML:** Fixed set of classes C .
Train a classifier from a set of labeled <document,class> pairs.
 - Discriminative vs. Generative models.
 - Unsupervised ML: Unknown set of classes C .
Topic modeling.
- How do we represent each document?
(feature representation).

Supervised Learning

- Given: Training data consisting of training examples $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$, where \mathbf{x}_i is an input example (a d -dimensional vector of attribute values) and y_i is the label.

example					label
1	x_{11}	x_{12}	...	x_{1d}	y_1
...
i	x_{i1}	x_{i2}	...	x_{id}	y_i
...
n	x_{n1}	x_{n2}	...	x_{nd}	y_n

- Goal: learn a hypothesis function $h(x)$ that approximates the true relationship between x and y . This functions should
 - 1) ideally be consistent with the training data.
 - 2) generalize to unseen examples.
- In NLP y_i typically form a finite, discrete set.

Representing Documents

to be, or not to be

- Set-of-words representation.
- Bag-of-words representation (Multi-set).
- Vector-space model: Each word corresponds to one dimension in vector space. Entries are either:
 - Binary (Word appears / does not appear)
 - (Weighted) frequency counts
 - Probabilities.

*to or
be not*

*be to or not
be to*

be	2
⋮	⋮
not	1
⋮	⋮
or	1
⋮	⋮
to	2

Probabilities in NLP

- Ambiguity is everywhere in NLP. There is often *uncertainty* about the “correct” interpretation. Which is more likely:
 - Speech recognition: “*recognize speech*” vs. “*wreck a nice beach*”
 - Machine translation: “*l’avocat general*”: “*the attorney general*” vs. “*the general avocado*”
 - Text classification: is a document that contains the word “*rice*” more likely to be about politics or about agriculture?
What if it also includes several occurrences of the word “*stir*”?
- Probabilities make it possible to combine evidence from multiple sources systematically to (using Bayesian statistics)

Bayesian Statistics

- Typically, we observe some evidence (for example, words in a document) and the goal is to infer the “correct” interpretation (for example, the topic of a text).
- Probabilities express the degree of belief we have in the possible interpretations.
- **Prior probabilities:** Probability of an interpretation prior to seeing any evidence.
- **Conditional (Posterior) probability:** Probability of an interpretation after taking evidence into account.

Probability Basics

- Begin with a **sample space** Ω
 - Each $\omega \in \Omega$ is a possible basic outcome / “possible world” (e.g. the 6 possible rolls of a die).
- A **probability distribution** assigns a probability to each basic outcome.

$$P(\omega) \leq 1.0 \text{ for every } \omega \in \Omega$$

$$\sum_{\omega \in \Omega} P(\omega) = 1.0$$

- E.g: six-sided die

$$P(1) + P(2) + P(3) + P(4) + P(5) + P(6) = 1.0$$

Events

- An *event* A is any subset of Ω .

$$P(A) = \sum_{\omega \in A} P(\omega)$$

- Example:

$$P(\text{die roll} < 4) = P(1) + P(2) + P(3) = 1/6 + 1/6 + 1/6 = 1/2$$

Random Variables

- A random variable is a function from basic outcomes to some range, e.g. real numbers or booleans.

$$\textit{Odd}(1) = \textit{true}$$

- A distribution P induces a probability distribution for any random variable.

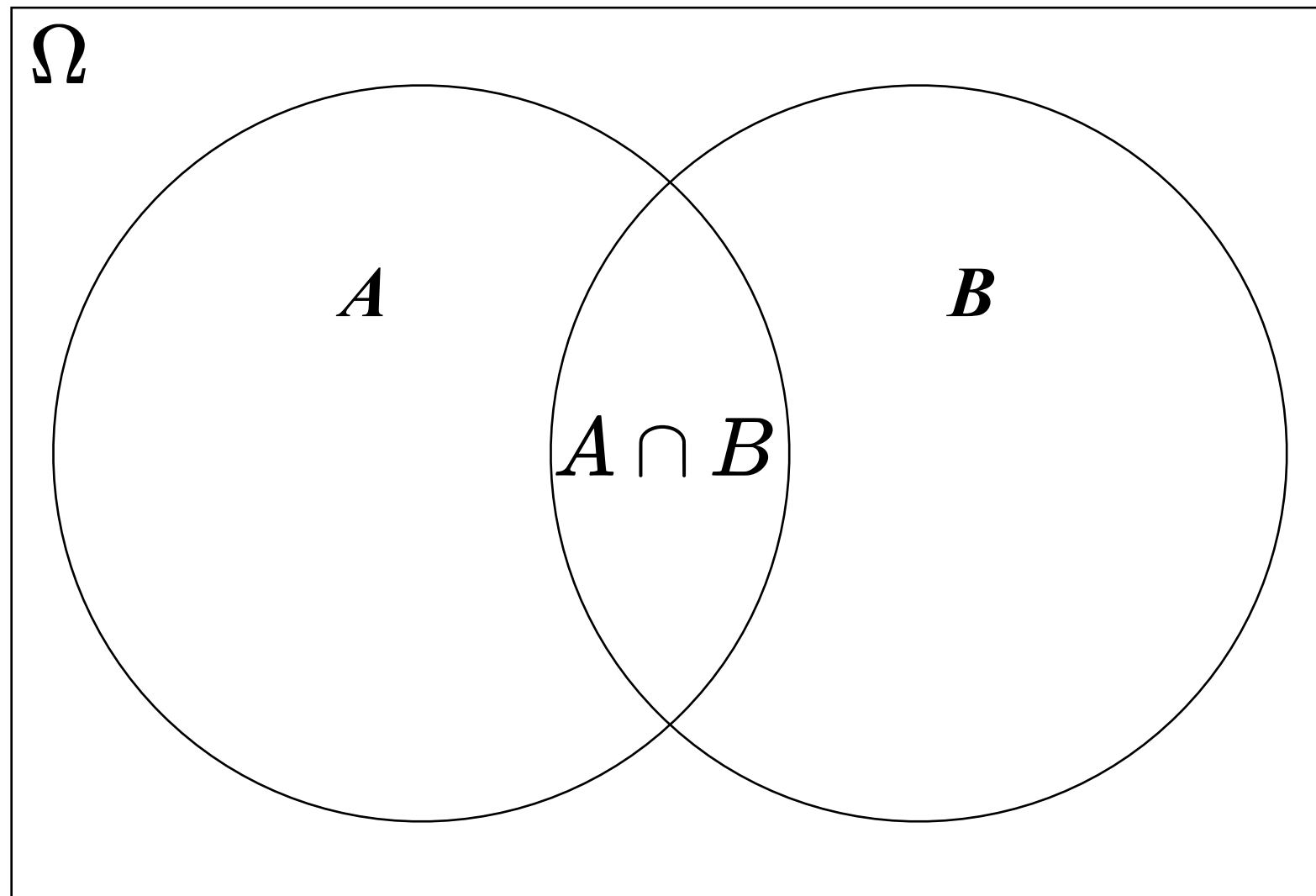
$$P(X = x_i) = \sum_{\{\omega: X(\omega)=x_i\}} P(\omega)$$

- E.g $P(\textit{Odd} = \textit{true}) = P(1) + P(3) + P(5) = 1/2$

Two six-sided dice example

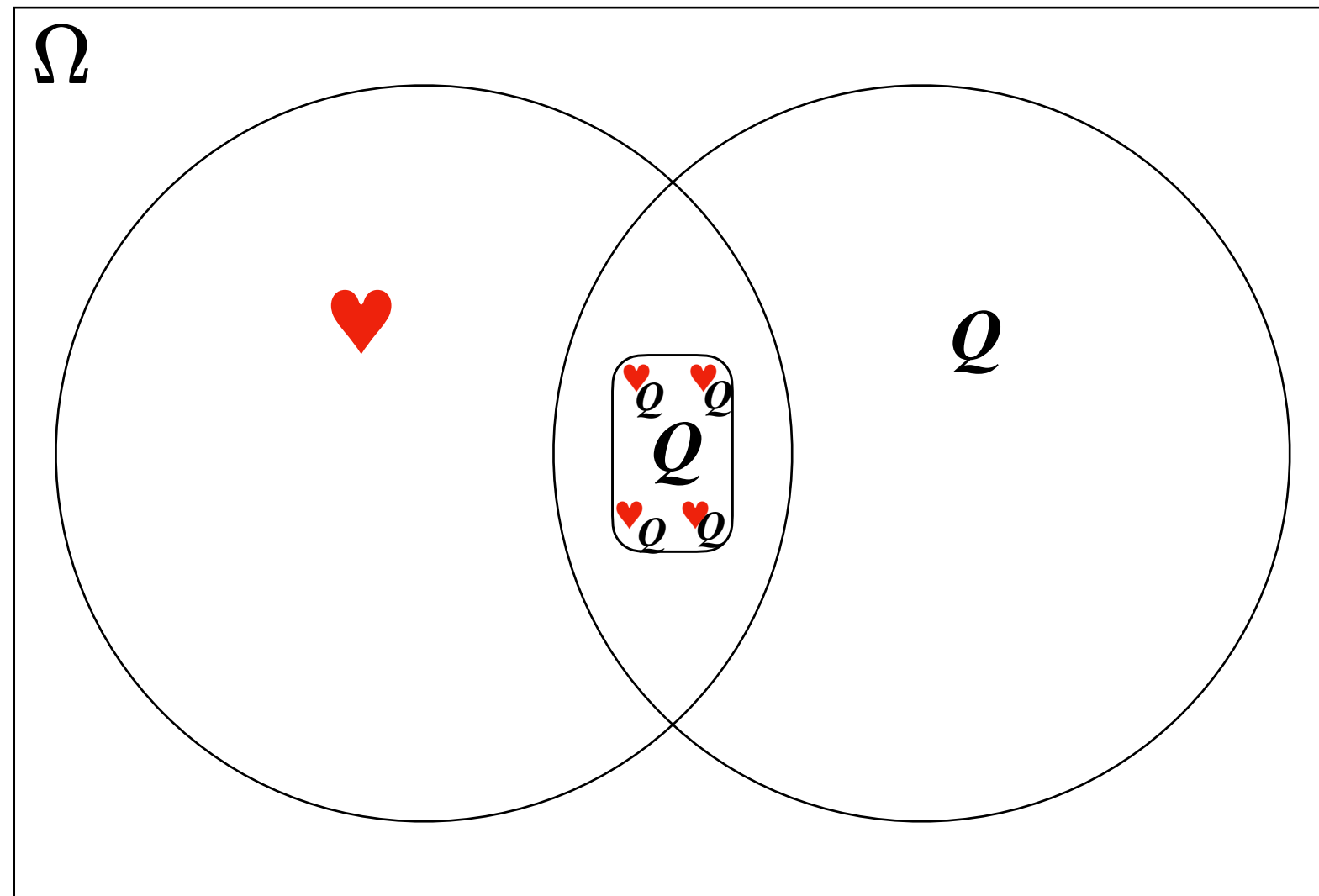
Joint and Conditional Probability

Joint probability: $P(A \cap B)$ also written as $P(A, B)$



Joint and Conditional Probability

Joint probability: $P(A \cap B)$ also written as $P(A, B)$

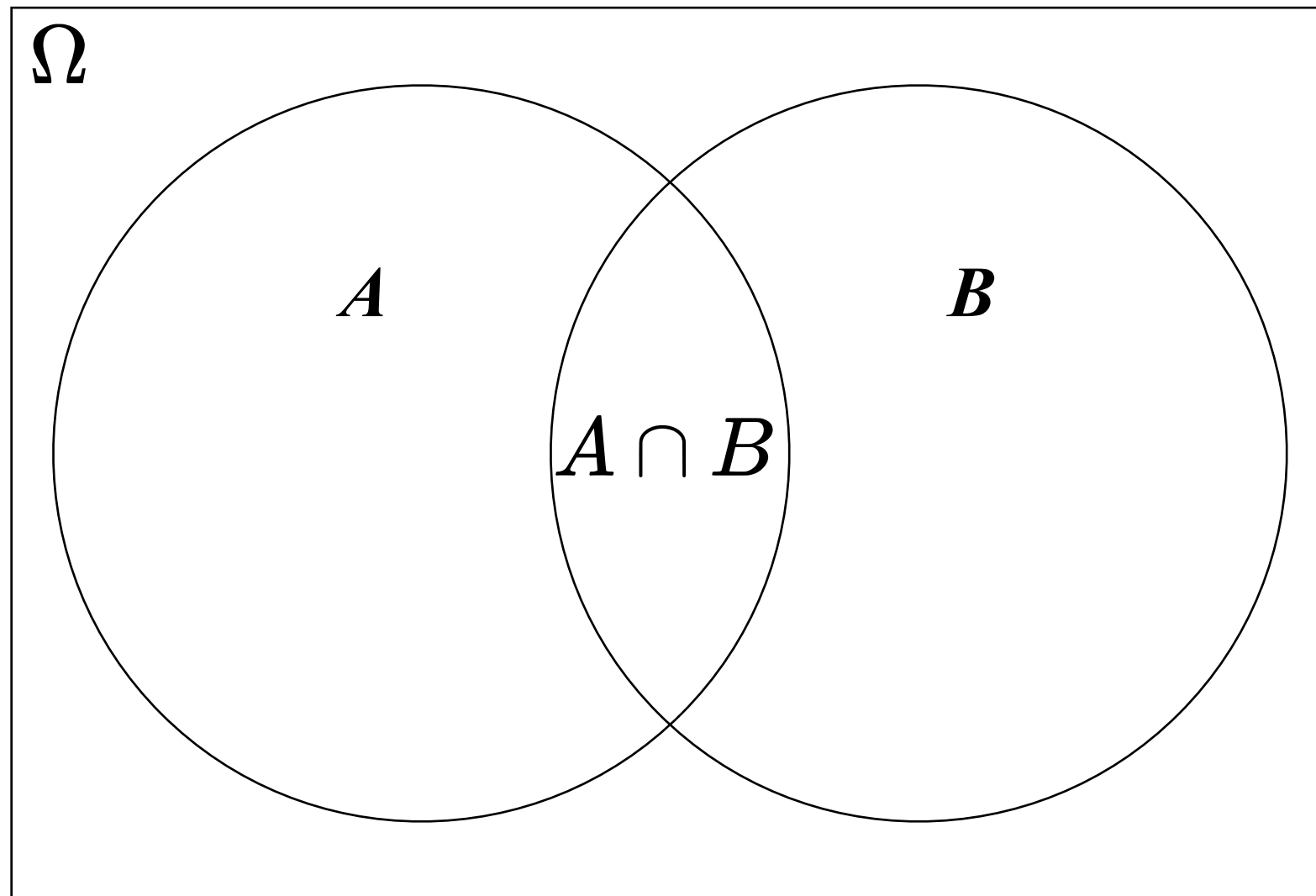


Assume 52 deck of cards. Draw a random card.

$$P(\heartsuit) = 1/4 \quad P(Q) = 1/13 \quad P(\heartsuit, Q) = 1/4 \times 1/13 = 1/52$$

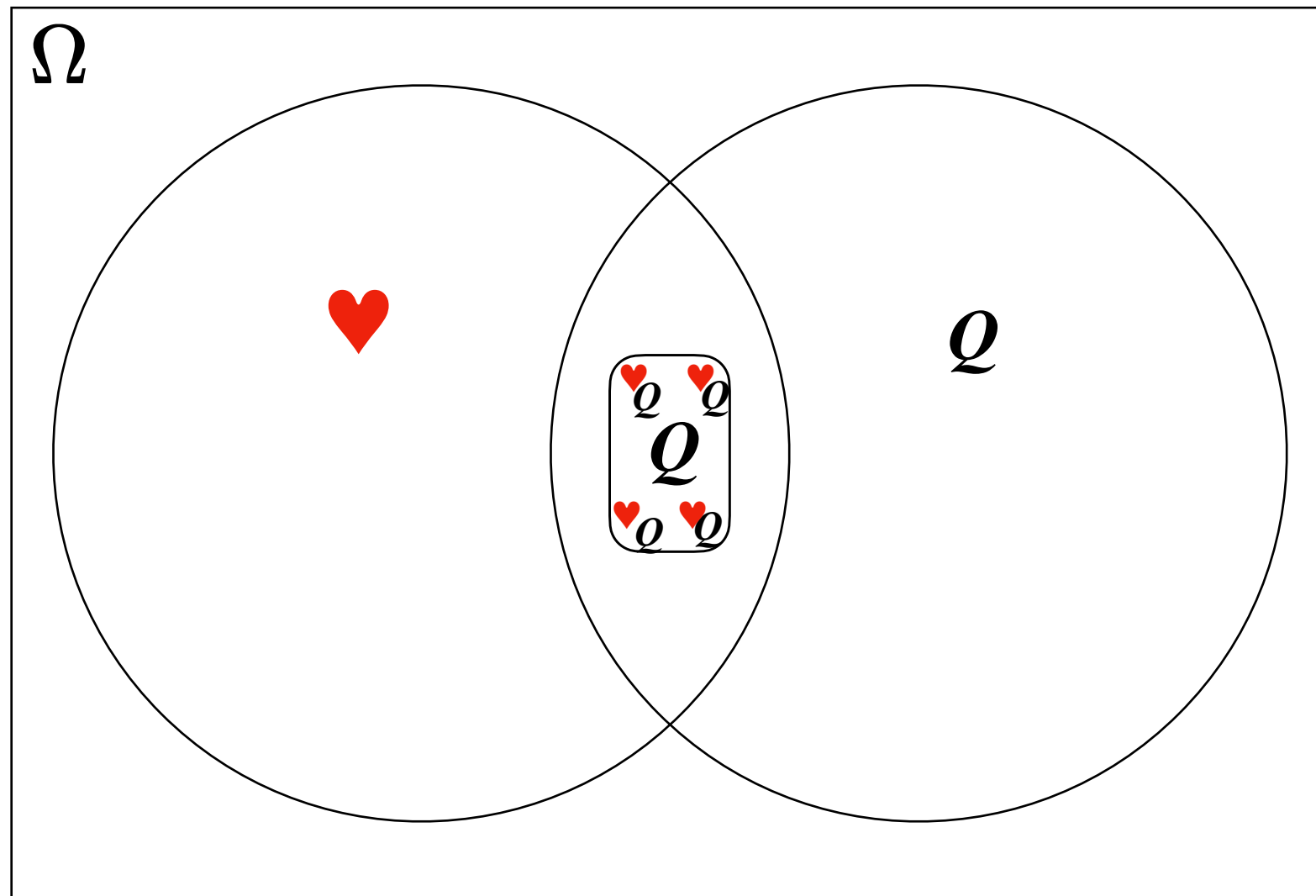
Joint and Conditional Probability

Conditional probability: $P(A|B) = \frac{P(A, B)}{P(B)}$



Joint and Conditional Probability

Conditional probability: $P(A|B) = \frac{P(A, B)}{P(B)}$



$$P(\heartsuit, Q) = 1/52$$

$$P(Q) = 1/13$$

$$P(\heartsuit|Q) = \frac{1/52}{1/13} = \frac{13}{52} = \frac{1}{4}$$

Rules for Conditional Probability

- Product rule: $P(A, B) = P(B) \cdot P(A|B) = P(A) \cdot P(B|A)$
- Chain rule (generalization of product rule):

$$P(A_n, \dots, A_1) = P(A_n | A_{n-1}, \dots, A_1) \cdot P(A_{n-1}, \dots, A_1)$$

- **Bayes' Rule:**

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

Independence

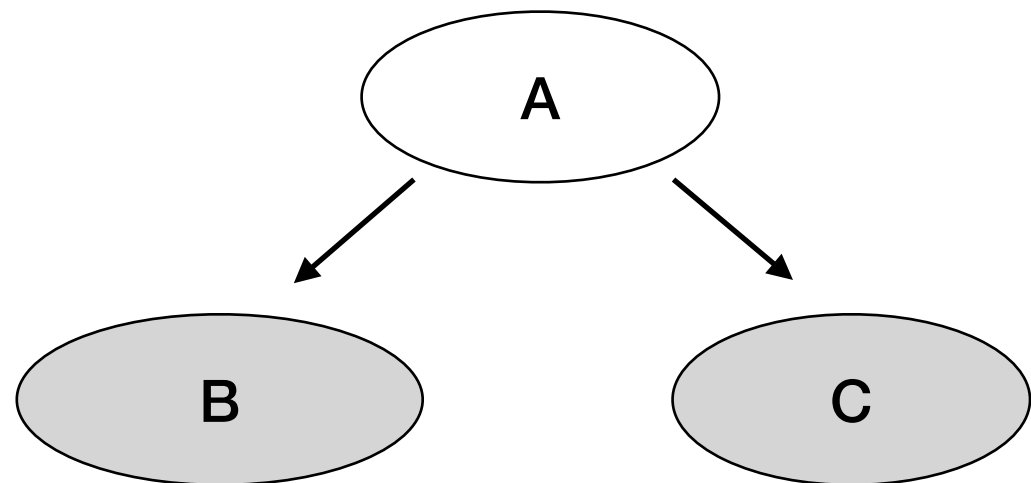
- Two events are independent if $P(A) = P(A|B)$
or equivalently $P(A, B) = P(A) \cdot P(B)$ (if $P(B) > 0$)

- Two events are **conditionally independent** if:

$$P(B, C|A) = P(B|A)P(C|A)$$

or equivalently

$$P(B|A, C) = P(B|A) \text{ and } P(C|A, B) = P(C|A)$$



Probabilities and Supervised Learning

- Given: Training data consisting of training examples
data = $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$,

Goal: Learn a mapping h from x to y .

- We would like to learn this mapping using $P(y|x)$.
- Two approaches:
 - Discriminative algorithms learn $P(y|x)$ directly.
 - Generative algorithms use Bayes rule

$$P(y|x) = \frac{P(x|y) \cdot P(y)}{P(x)}$$

Discriminative Algorithms

- Model conditional distribution of the label given the data $P(y|x)$
- Learns decision boundaries that separate instances of the different classes.
- To predict a new example, check on which side of the decision boundary it falls.
- Examples:
linear and log-linear models, support vector machine (SVM), decision trees, random forests, ...

Generative Algorithms

- Assume the observed data is being “generated” by a “hidden” class label.
- Build a **different conditional distribution** for each class.
- To predict a new example, check it under each of the models and see which one matches best.

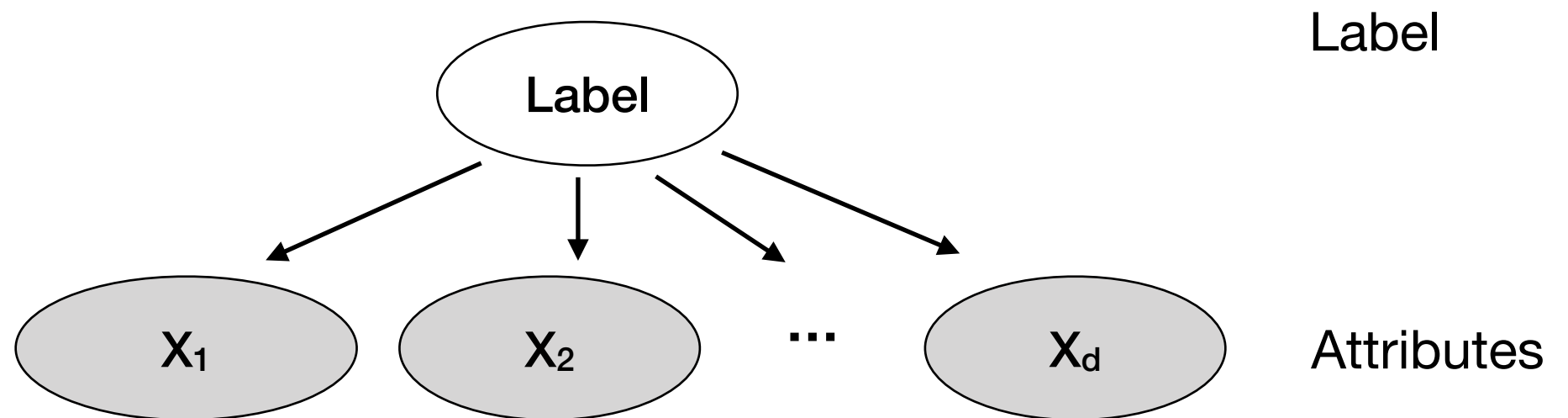
- Estimate $P(x|y)$ and $P(y)$. Then use Bayes rule

$$P(y|x) = \frac{P(x|y) \cdot P(y)}{P(x)}$$

- Examples:

Naive Bayes, Hidden Markov Models, Gaussian Mixture Models, PCFGs, ...

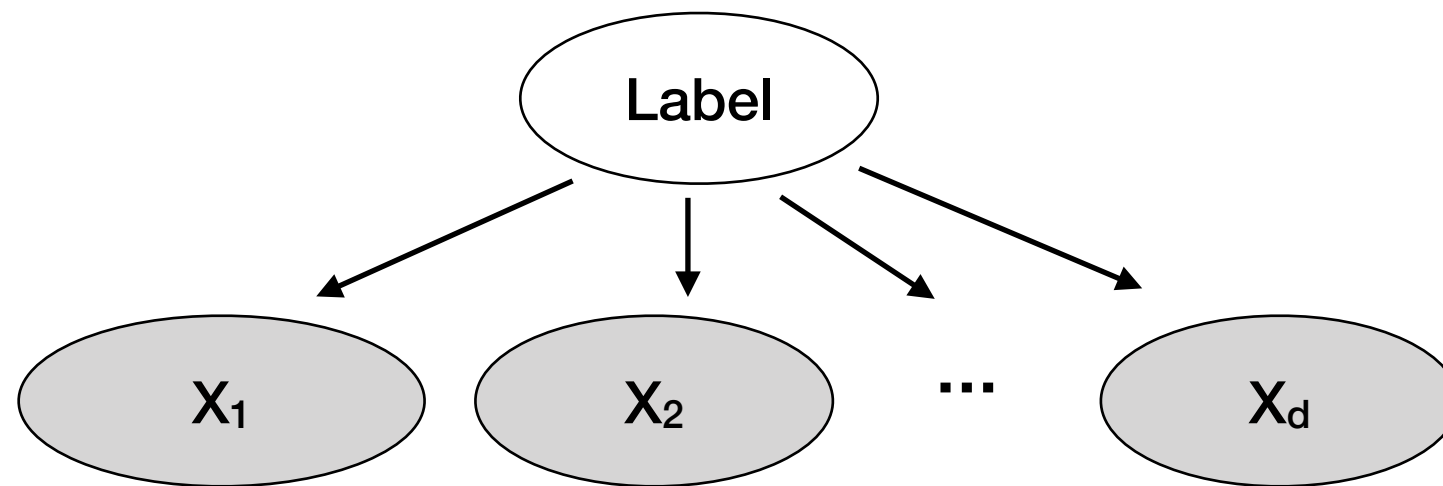
Naive Bayes



$$\mathbf{P}(Label, X_1, \dots, X_d) = \mathbf{P}(Label) \prod_i P(X_i | Label)$$

$$\begin{aligned} \mathbf{P}(Label | X_1, \dots, X_d) &= \frac{\mathbf{P}(Label) \prod_i P(X_i | Label)}{\prod_i P(X_i)} \\ &= \alpha [\mathbf{P}(Label) \prod_i P(X_i | Label)] \end{aligned}$$

Naive Bayes Classifier



$$\mathbf{P}(Label|X_1, \dots, X_d) = \alpha[\mathbf{P}(Label) \prod_i P(X_i|Label)]$$

$$y^* = \arg \max_y P(y) \prod_i P(x_i|y)$$

Note that the normalizer α does no longer matter for the argmax because α is independent of the class label.

Training the Naive Bayes' Classifier

- Goal: Use the training data to estimate $P(\text{Label})$ and $P(X_i|\text{Label})$ from training data.
- Estimate the prior and posterior probabilities using **Maximum Likelihood Estimates (MLE)**:

$$P(y) = \frac{\text{Count}(y)}{\sum_{y' \in Y} \text{Count}(y')}$$

$$P(x_i|y) = \frac{\text{Count}(x_i, y)}{\sum_{x'} \text{Count}(x', y)}$$

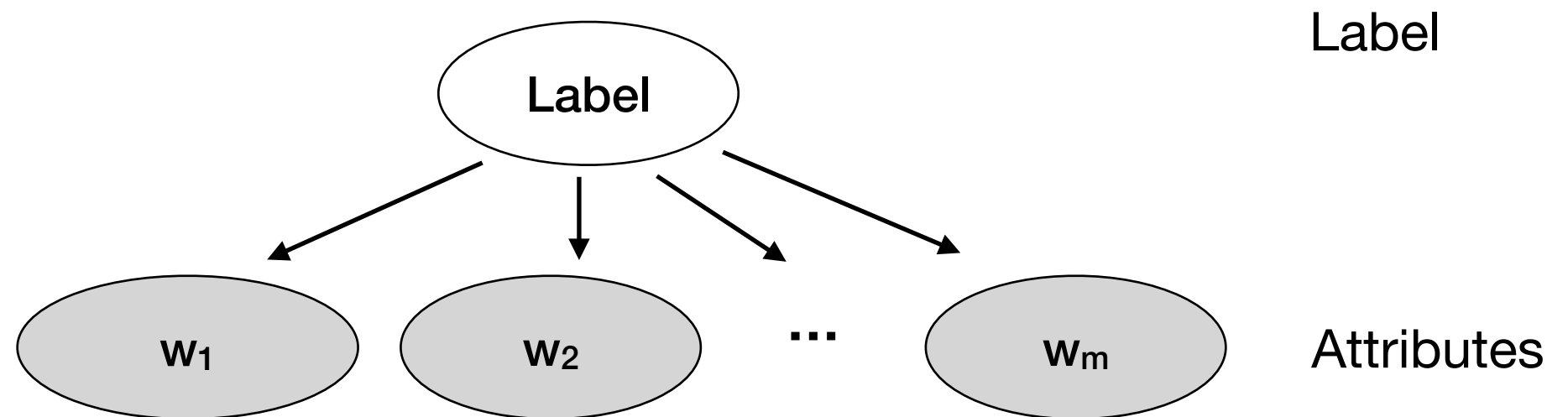
- I.e. we just count how often each token in the document appears together with each class label.

Why the Independence Assumption Matters

- Without the independence assumption we would have to estimate $\mathbf{P}(X_1, \dots, X_d | Label)$
- There would be many combinations of x_1, \dots, x_d that are never seen (sparse data).
- The independence assumption allows us to estimate each $\mathbf{P}(X_1 | label)$ independently.

Is this a safe assumption for documents?
Are the words really independent of each other?

Naive Bayes for Text Classification



- For text classification, the input is not normally a document vector with d dimensions (where d is vocabulary size)
- Instead of using a bag-of-words input vector, $w_1 \dots w_m$ are the input words in the test document (i.e. w_1 is the first word in the doc, w_m is the last word in the doc).

Training the Naive Bayes' Classifier

- Ways to improve this model?
- Some issues to consider...
 - What if there are words that do not appear in the training set? What if it appears only once?
 - What if the plural of a word never appears in the training set?
 - How are extremely common words (e.g., “the”, “a”) handled?

What is a Word?

- e.g., are “*Cat*”, “*cat*” and “*cats*” the same word?
- “*September*” and “*Sept*”?
- “*zero*” and “*oh*”?
- Is “*_*” a word? “*.*”? “***”? “*(*”?
- How many words are there in “*don’t*” ? “*Gonna*” ? “*I.B.M.*”?
- In Japanese and Chinese text -- how do we identify a word?
- ...

Text Normalization

- Every NLP task requires some text normalization.
- Segmenting / tokenizing words in running text.
- Normalizing word forms (lemmatization or stemming, possibly replacing named-entities).
- Sentence splitting.

Linguistic Terminology

- **Sentence:** Unit of written language.
- **Utterance:** Unit of spoken language.
- Word **Form:** the inflected form as it actually appears in the corpus. *“produced”*
- Word **Stem:** The part of the word that never changes between morphological variations. *“produc”*
- **Lemma:** an abstract base form, shared by word forms, having the same **stem**, part of speech, and word sense – stands for the **class** of words with **stem**.
“produce”
- **Type:** number of distinct words in a corpus (vocabulary size).
- **Token:** Total number of word occurrences.

Tokenization

- Tokenization: The process of segmenting text (a sequence of characters) into a sequence of tokens (words).

"Mr. O'Neill thinks that the boys' stories about Chile's capital aren't amusing."

mr.	o'neill	thinks	that	the	boys'	stories	about	Chile's	capital	are	n't
amusing	.										

- Simple (but weak) approach: Separate off punctuation. Then split on whitespaces.
- Typical implementations use regular expressions (finite state automata).

Tokenization Issues

- Dealing with punctuation (some may be part of a word)
“Ph.D.”, “O’Reilly”, “pick-me-up”
- Which tokens to include (punctuation might be useful for parsing, but not for text classification)?
- Language dependent: Some languages don’t separate words with whitespaces.
de: “*Lebensversicherungsgesellschaftsangestellter*”

zh: 日文章鱼怎么说? - *Japanese Octopus* how say?

日文章鱼怎么说? - *Day article fish* how say?

Chinese example from Sproat (1996)

Lemmatization

- Converting Lemmas into their base form.

“Mr. O'Neill thinks that the boys' stories about Chile's capital aren't amusing.”

mr.	o'neill	think	that	the	boy	story	about	chile's	capital	are	n't
amusing		.									

PER	PER	think	that	the	boy	story	about	LOC	's	capital	are	n't
amusing		.										