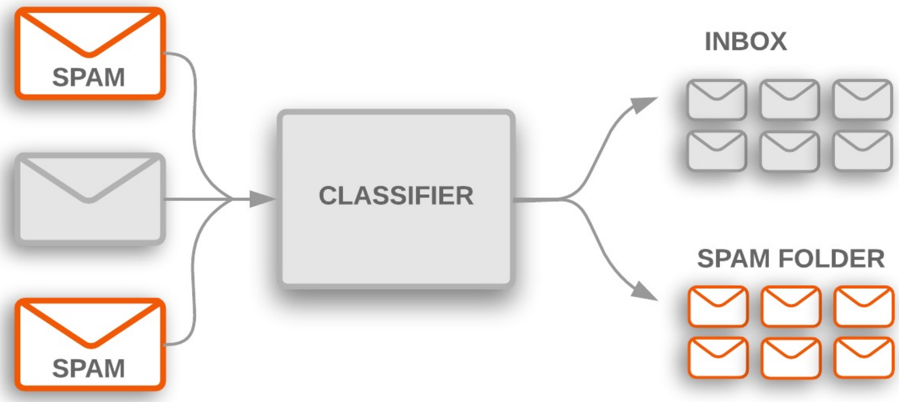
CSE 4392 Special Topics
Natural Language Processing

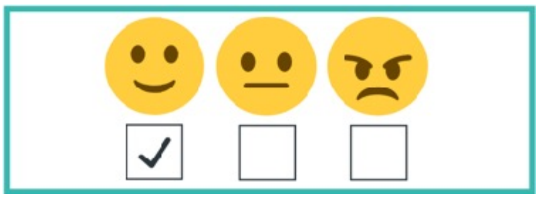
Text Classification

2024 Spring

Why Classify?



Spam detection

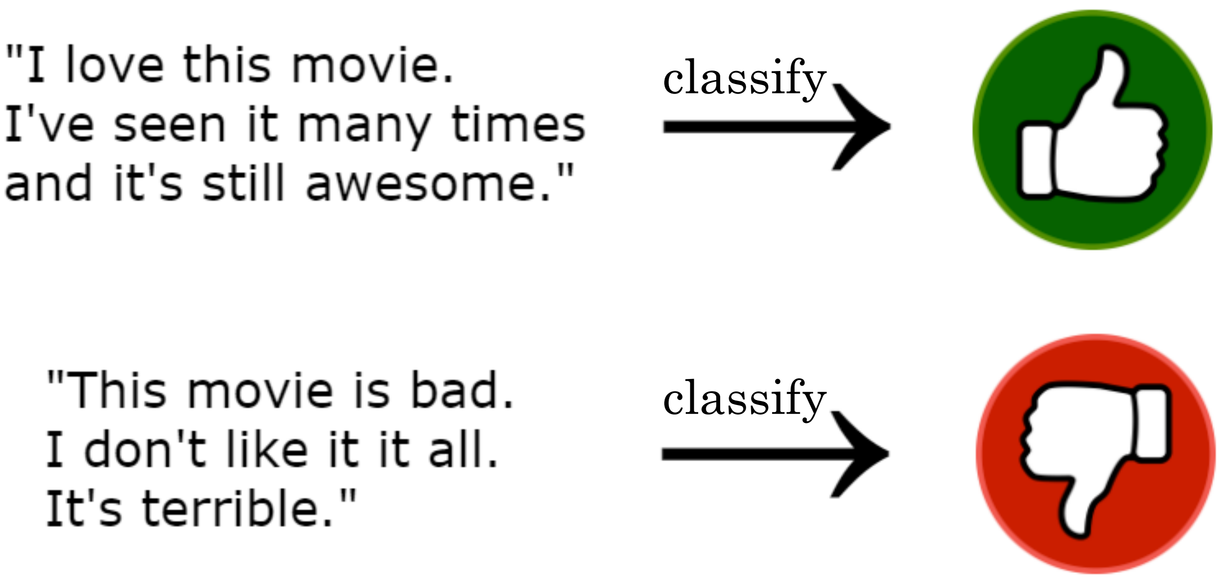


Sentiment analysis

- Authorship attribution
- Language detection
- News categorization

The Classification: The Task

- Inputs:
  - A document d
  - A set of classes $C = \{c_1, c_2, c_3, \ldots, c_m\}$
- Output:
  - Predicted class c for document d



Rule-based Classification

- Combination of features on words in the document, and meta-data:
  - IF there exists word w in document d, and w is in {good, great, awesome, extraordinary, …}

    THEN output POSITIVE as class label
  - IF the email subject contains any words in {“casino”, “weeds”, “viagra”, …}

    THEN output SPAM as class label
- Can be very accurate
- But hard and tedius to define (there can be many of them, some even unknown to us!)
- Not easily generalizable (may not apply in other domains or scenarios)

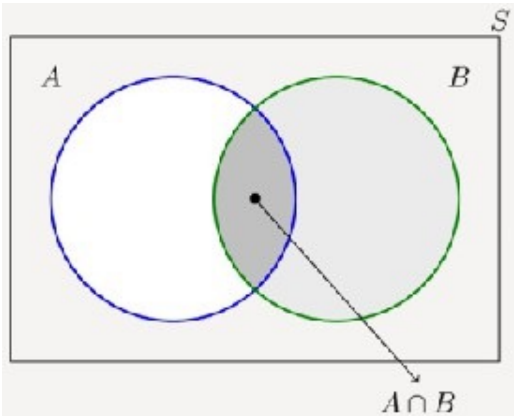
Supervised Learning: Use Statistics

- Data-driven approach
- Let the machine figure out the best patterns to use
- Inputs:
  - Set of *m* classes $C = \{c_1, c_2, \ldots, c_m\}$
  - Set of *n* 'labeled' documents: $\{(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)\}$

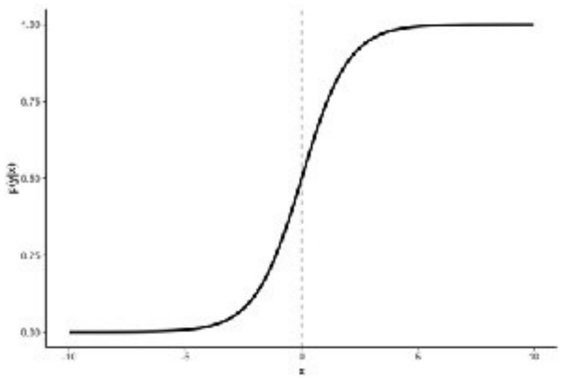
Key questions:
1) The form of F?
2) How to learn F?

- Output:
  - Trained classifier, $F : d \rightarrow c$

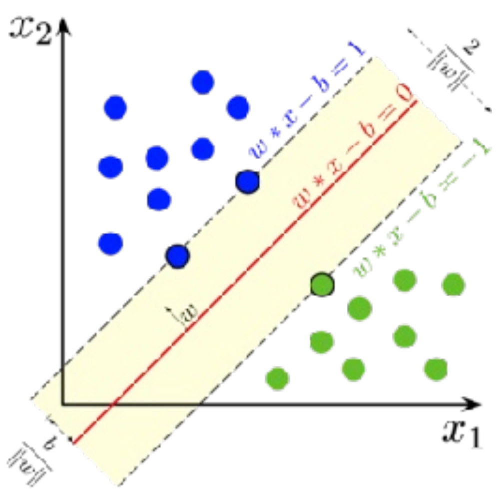
Types of Supervised Classifiers



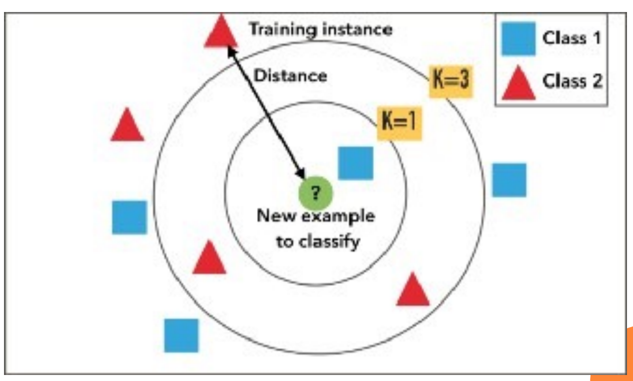
Naive Bayes



Logistic regression



Support vector machines



k-nearest neighbors

Quiz

- Which of the four types of the classifiers has an inference cost porportional to the size of the training data?

a) Naïve Bayes

b) Logistic Regression

c) Support Vector Machine

d) K-Nearest Neighbors

Multinomial Naive Bayes

- Simple classification model making use of Bayes rule

- Bayes Rule:

class

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)}$$

docment

- Makes strong (naive) independence assumptions

Predicting a Class

- Best class:

$$C_{MAP} = \underset{c \in C}{\arg\max}\, P(c|d)$$
$$= \underset{c}{\arg\max}\, \frac{P(c)P(d|c)}{P(d)}$$
$$= \underset{c}{\arg\max}\, P(c)P(d|c)$$

d is a document
c is a class

- MAP = Maximum *a Posteriori*
- P(c) → Prior probability of class *c*
- P(d) → constant for *d*, so omitted

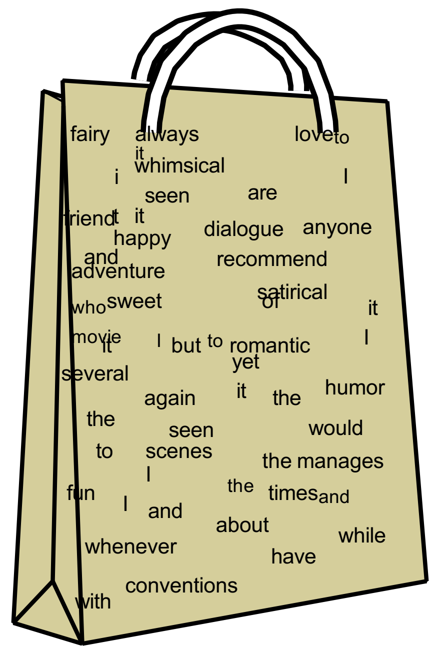
How to compute P(d | c)?

- Option 1: represent the entire sequence of words
  - $P(w_1, w_2, w_3, \ldots, w_k \mid c)$ *(too many sequences!)*

- Option 2: Bag of words
  - Assume position of each word is irrelevant (both absolute and relative)
  - $P(w_1, w_2, w_3, \ldots, w_k \mid c) = P(w_1|c)\, P(w_2|c) \ldots P(w_k|c)$
  - Probability of each word is conditionally independent of each other given class c

Bag of words

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

PREDICTING WITH NAIVE BAYES

- Mathematically, we now have:

$$
\begin{aligned}
C_{MAP} &= \arg\max_{c} P(d|c)P(c) \\
&= \arg\max_{c} P(w_1, w_2, \dots, w_k|c)P(c) \\
&= \arg\max_{c} P(c)\prod_{i=1}^{k} P(w_i|c)
\end{aligned}
$$

(Using the BOW assumption!)

Naive Bayes as a Generative Model

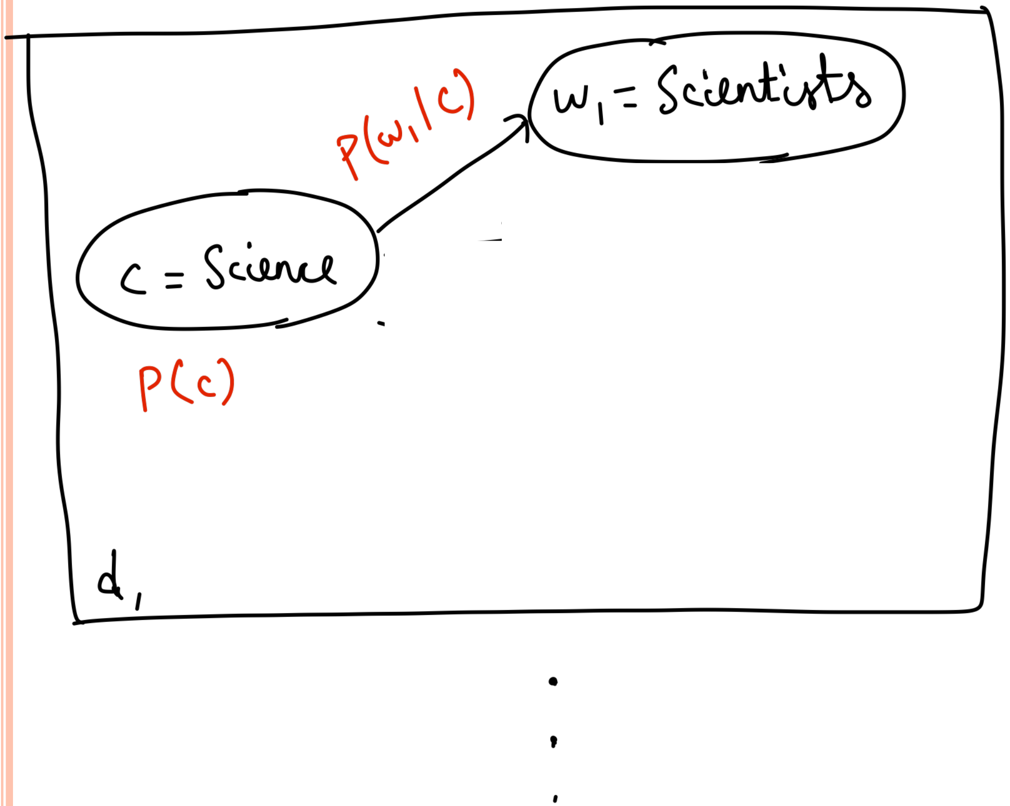
c = Science

$P(c)$

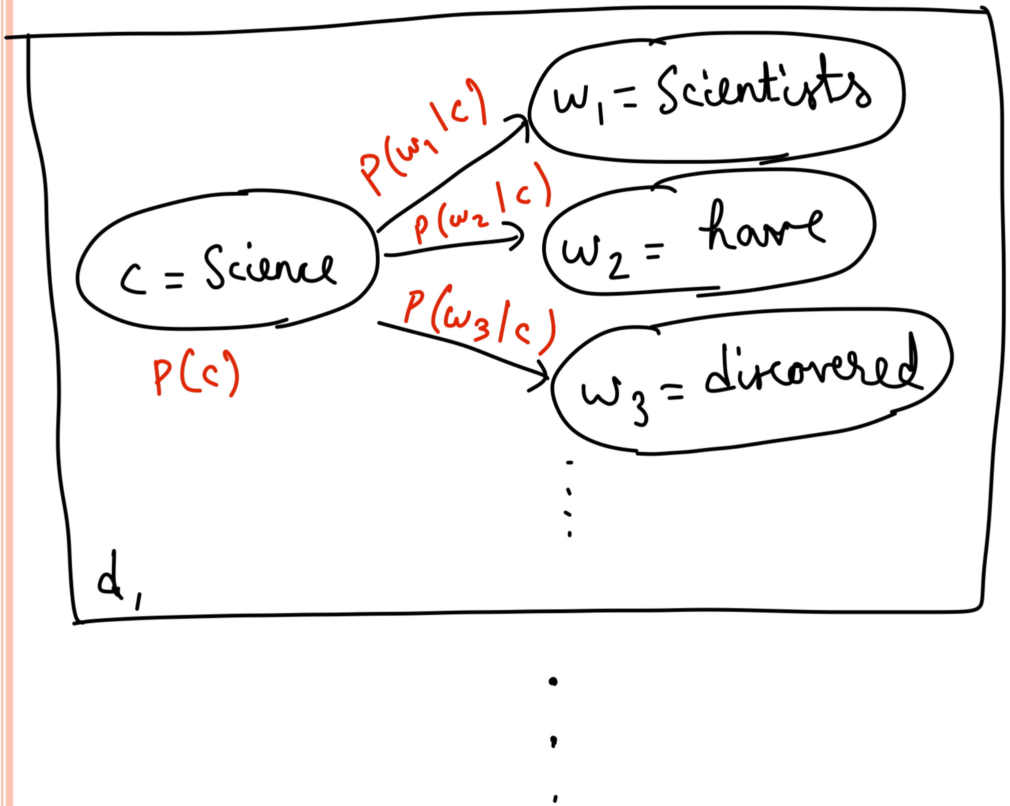
d_1

$\vdots$

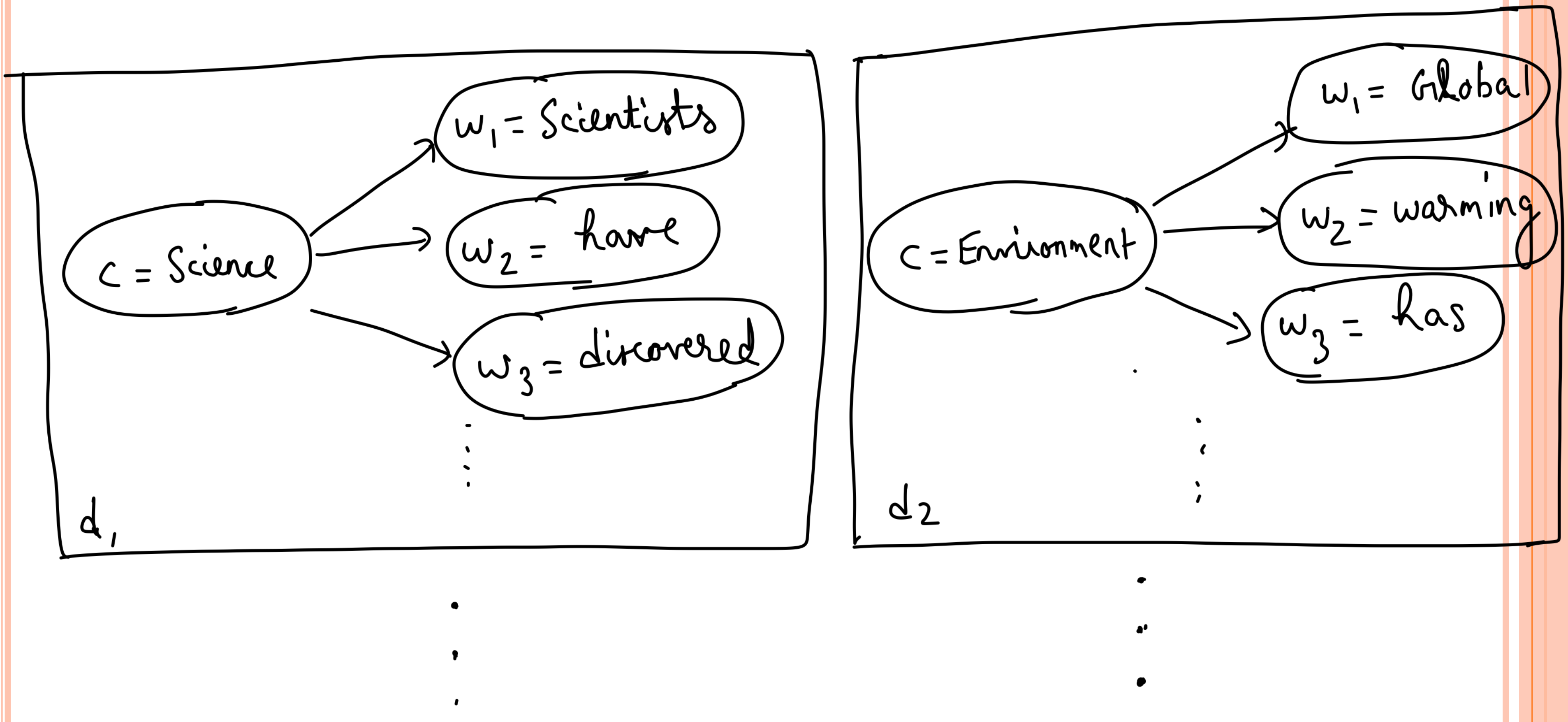
NAIVE BAYES AS A GENERATIVE MODEL



NAIVE BAYES AS A GENERATIVE MODEL



Naive Bayes as a generative model



Generate the entire data set one document at a time

ESTIMATING PROBABILITIES

- Maximum likelihood estimates:

\# of documents in class c_j

$$\hat{P}(c_j) = \frac{Count(class = c_j)}{\sum_c Count(class = c)}$$

Total # of documents

$$\hat{P}(w_i | c_j) = \frac{Count(w_i, c_j)}{\sum_w Count(w, c_j)}$$

DATA SPARSITY

- What if count('amazing', *positive*) = *0*?
  - Implies P('amazing' | *positive*) = 0
- Given a review document, d = "…. most amazing movie ever …"

$$C_{MAP} = \arg\max_{c} \hat{P}(c) \prod_{i=1}^{k} P(w_i|c)$$

$$= \arg\max_{c} \hat{P}(c) * 0$$

SOLUTION: SMOOTHING!

- Laplace smoothing:

$$\hat{P}(w_i|c) = \frac{Count(w_i, c) + \alpha}{\sum_w Count(w, c) + \alpha\, |V|}$$

Vocabulary Size

- Simple, easy to use
- Effective in practive

Overall Process

Input: Set of annotated documents $\{(d_i, c_i)\}_{i=1}^{n}$

1. Compute vocabulary set **V** of all words

2. Calculate

$$\hat{P}(c_j) = \frac{Count(\#docs\ in\ c_j)}{Total\ \#\ docs}$$

3. Calculate

$$\hat{P}(w_i|c) = \frac{Count(w_i, c) + \alpha}{\sum_{w \in V}[Count(w, c) + \alpha]}$$

4. (Prediction) Given document $d = (w_1, w_2, \ldots, w_k)$

$$C_{MAP} = \arg\max_{c} \hat{P}(c) \prod_{i=1}^{k} \hat{P}(w_i|c)$$

Naïve Bayes Classification Example

$$\hat{P}(c) = \frac{N_c}{N}$$

$$\hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

Choosing a class:

P(c|d5) $\propto$ 3/4 * $(3/7)^3$ * 1/14 * 1/14
≈ 0.0003

Conditional Probabilities:

P(Chinese|*c*) = (5+1) / (8+6) = 6/14 = 3/7

P(Tokyo|*c*) = (0+1) / (8+6) = 1/14

P(Japan|*c*) = (0+1) / (8+6) = 1/14

P(Chinese|*j*) = (1+1) / (3+6) = 2/9

P(Tokyo|*j*) = (1+1) / (3+6) = 2/9

P(Japan|*j*) = (1+1) / (3+6) = 2/9

P(j|d5) $\propto$ 1/4 * $(2/9)^3$ * 2/9 * 2/9
≈ 0.0001

Quiz

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	6	Macao Chinese Visit Tokyo Chinese	?

- Given the above training documents d1-d4, and their class labels, compute $P(c \mid d_6)$, after applying add-1 smoothing.

FEATURES

- In general, Naive Bayes can use any set of features, not just words
  - URLs, email addresses, Capitalization, …
  - Domain knowledge crucial to performance

Rank	Category	Feature	Rank	Category	Feature
1	Subject	Number of capitalized words	1	Subject	Min of the compression ratio for the bz2 compressor
2	Subject	Sum of all the character lengths of words	2	Subject	Min of the compression ratio for the zlib compressor
3	Subject	Number of words containing letters and numbers	3	Subject	Min of character diversity of each word
4	Subject	Max of ratio of digit characters to all characters of each word	4	Subject	Min of the compression ratio for the lzw compressor
5	Header	Hour of day when email was sent	5	Subject	Max of the character lengths of words
		(a)			(b)
		Spam URLs Features			
1	URL	The number of all URLs in an email	1	Header	Day of week when email was sent
2	URL	The number of unique URLs in an email	2	Payload	Number of characters
3	Payload	Number of words containing letters and numbers	3	Payload	Sum of all the character lengths of words
4	Payload	Min of the compression ratio for the bz2 compressor	4	Header	Minute of hour when email was sent
5	Payload	Number of words containing only letters	5	Header	Hour of day when email was sent

Top features for Spam detection

Naive Bayes and Language Models

- If features = bag of words, each class is a unigram language model!

- For class c, assigning each word: $P(w|c)$
  assigning sentence: $P(S|c) = \prod_{w \in S} P(w|c)$

Class *positive*

0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film
…	

I	love	this	fun	film
0.1	0.1	.05	0.01	0.1

P(s | pos) = 0.0000005

Naïve Bayes as a Language Model

- Which class assigns the higher probability to s?

Model pos	
0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film

Model neg	
0.2	I
0.001	love
0.01	this
0.005	fun
0.1	film

I	love	the	fun	film
0.1	0.1	0.01	0.05	0.1
0.2	0.001	0.01	0.005	0.1

$$P(s|pos) > P(s|neg)$$

Evaluation

- Consider binary classification

- Table of predictions

Confusion Matrix

Truth

Predicted

	Positive	Negative
Positive	100	5
Negative	45	100

false positives

false negatives

- Ideally, we want:

	Positive	**Negative**
Positive	145	0
Negative	0	105

Evaluation Metrics

		Truth	
		Positive	Negative
Predicted	Positive	100	5
	Negative	45	100

- True positive: Predicted + and actual +
- True negative: Predicted - and actual -
- False positive: Predicted + and actual -
- False negative: Predicted - and actual +

$$Accuracy = \frac{TP + TN}{Total} = \frac{200}{250} = 80\%$$

Evaluation Metrics

Truth

Predicted

	Positive	Negative
Positive	100	5
Negative	45	100

	Positive	Negative
Positive	10	45
Negative	5	190

- True positive: Predicted + and actual +
- True negative: Predicted - and actual -
- False positive: Predicted + and actual -
- False negative: Predicted - and actual +

$$Accuracy = \frac{TP + TN}{Total} = \frac{200}{250} = 80\%$$

Still the same result!
Accuracy is a coarse-grain measure.
Also not suitable for retrieval (finding true positives).