# Introduction to Information Retrieval
http://informationretrieval.org

# IIR 13: Text Classification & Naive Bayes

Hinrich Schütze

Institute for Natural Language Processing, Universität Stuttgart

2008.06.10

# Overview

# Outline

# Relevance feedback

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant.

# Relevance feedback

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant.
- We then use this information to return better search results.

# Relevance feedback

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant.
- We then use this information to return better search results.
- This is a form of text classification.

# Relevance feedback

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant.
- We then use this information to return better search results.
- This is a form of text classification.
- Two "classes": relevant, nonrelevant

# Relevance feedback

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant.
- We then use this information to return better search results.
- This is a form of text classification.
- Two "classes": relevant, nonrelevant
- For each document, decide whether it is relevant or nonrelevant

# Relevance feedback

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant.
- We then use this information to return better search results.
- This is a form of text classification.
- Two "classes": relevant, nonrelevant
- For each document, decide whether it is relevant or nonrelevant
- The problem space relevance feedback belongs to is called classification.

# Relevance feedback

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant.
- We then use this information to return better search results.
- This is a form of text classification.
- Two "classes": relevant, nonrelevant
- For each document, decide whether it is relevant or nonrelevant
- The problem space relevance feedback belongs to is called classification.
- The notion of classification is very general and has many applications within and beyond information retrieval.

From information retrieval to text classification:

standing queries – Google Alerts

# Another TC task: spam filtering

```
From: ‘‘’’ <takworlld@hotmail.com>
Subject: real estate is the only way... gem  oalvgkay

Anyone can buy real estate with no money down

Stop paying rent TODAY !

There is no need to spend hundreds or even thousands for
similar courses

I am 22 years old and I have already purchased 6 properties
using the
methods outlined in this truly INCREDIBLE ebook.

Change your life NOW !

=================================================
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm
=================================================
```

# Formal definition of TC: Training

Given:

- A document space $\mathbb{X}$

# Formal definition of TC: Training

Given:

- A document space $\mathbb{X}$
  - Documents are represented in this space, typically some type of high-dimensional space.

# Formal definition of TC: Training

Given:

- A document space $\mathbb{X}$
  - Documents are represented in this space, typically some type of high-dimensional space.
- A fixed set of classes $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$

# Formal definition of TC: Training

Given:

- A document space $\mathbb{X}$
  - Documents are represented in this space, typically some type of high-dimensional space.
- A fixed set of classes $\mathbb{C} = \{c_1, c_2, \dots, c_J\}$
  - The classes are human-defined for the needs of an application (e.g., spam vs. non-spam).

# Formal definition of TC: Training

Given:

- A document space $\mathbb{X}$
  - Documents are represented in this space, typically some type of high-dimensional space.
- A fixed set of classes $\mathbb{C} = \{c_1, c_2, \dots, c_J\}$
  - The classes are human-defined for the needs of an application (e.g., spam vs. non-spam).
- A training set $\mathbb{D}$ of labeled documents with each labeled document $\langle d, c \rangle \in \mathbb{X} \times \mathbb{C}$

# Formal definition of TC: Training

Given:

- A document space $\mathbb{X}$
  - Documents are represented in this space, typically some type of high-dimensional space.
- A fixed set of classes $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$
  - The classes are human-defined for the needs of an application (e.g., spam vs. non-spam).
- A training set $\mathbb{D}$ of labeled documents with each labeled document $\langle d, c \rangle \in \mathbb{X} \times \mathbb{C}$

Using a learning method or learning algorithm, we then wish to learn a classifier $\gamma$ that maps documents to classes:

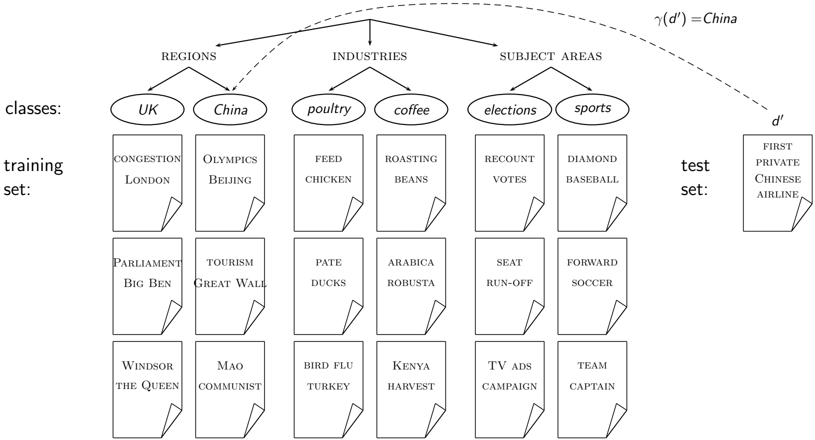
$$\gamma : \mathbb{X} \rightarrow \mathbb{C}$$

# Formal definition of TC: Application/Testing

Given: a description $d \in \mathbb{X}$ of a document

Determine: $\gamma(d) \in \mathbb{C}$, that is, the class that is most appropriate for $d$

# Topic classification



Many search engine functionalities are based on classification.

Examples?

# Applications of text classification in IR

- Language identification (classes: English vs. French etc.)

# Applications of text classification in IR

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam, example: googel.org)

# Applications of text classification in IR

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam, example: googel.org)
- The automatic detection of sexually explicit content (sexually explicit vs. not)

# Applications of text classification in IR

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam, example: googel.org)
- The automatic detection of sexually explicit content (sexually explicit vs. not)
- Sentiment detection: is a movie or product review positive or negative (positive vs. negative)

# Applications of text classification in IR

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam, example: googel.org)
- The automatic detection of sexually explicit content (sexually explicit vs. not)
- Sentiment detection: is a movie or product review positive or negative (positive vs. negative)
- Topic-specific or *vertical* search – restrict search to a "vertical" like "related to health" (relevant to vertical vs. not)

# Applications of text classification in IR

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam, example: googel.org)
- The automatic detection of sexually explicit content (sexually explicit vs. not)
- Sentiment detection: is a movie or product review positive or negative (positive vs. negative)
- Topic-specific or *vertical* search – restrict search to a "vertical" like "related to health" (relevant to vertical vs. not)
- Machine-learned ranking function in ad hoc retrieval (relevant vs. nonrelevant)

# Applications of text classification in IR

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam, example: googel.org)
- The automatic detection of sexually explicit content (sexually explicit vs. not)
- Sentiment detection: is a movie or product review positive or negative (positive vs. negative)
- Topic-specific or *vertical* search – restrict search to a "vertical" like "related to health" (relevant to vertical vs. not)
- Machine-learned ranking function in ad hoc retrieval (relevant vs. nonrelevant)
- Semantic Web: Automatically add semantic tags for non-tagged text (e.g., for each paragraph: relevant to a vertical like health or not)

# Classification methods: 1. Manual

- Manual classification was used by Yahoo in the beginning of the web. Also: ODP, PubMed

# Classification methods: 1. Manual

- Manual classification was used by Yahoo in the beginning of the web. Also: ODP, PubMed
- Very accurate if job is done by experts

# Classification methods: 1. Manual

- Manual classification was used by Yahoo in the beginning of the web. Also: ODP, PubMed
- Very accurate if job is done by experts
- Consistent when the problem size and team is small

# Classification methods: 1. Manual

- Manual classification was used by Yahoo in the beginning of the web. Also: ODP, PubMed
- Very accurate if job is done by experts
- Consistent when the problem size and team is small
- Manual classification is difficult and expensive to scale.

# Classification methods: 1. Manual

- Manual classification was used by Yahoo in the beginning of the web. Also: ODP, PubMed
- Very accurate if job is done by experts
- Consistent when the problem size and team is small
- Manual classification is difficult and expensive to scale.
- $\rightarrow$ We need automatic methods for classification.

# Classification methods: 2. Rule-based

- Our Google Alerts example was rule-based classification.

# Classification methods: 2. Rule-based

- Our Google Alerts example was rule-based classification.
- There are "IDE" type development enviroments for writing very complex rules efficiently. (e.g., Verity)

# Classification methods: 2. Rule-based

- Our Google Alerts example was rule-based classification.
- There are "IDE" type development enviroments for writing very complex rules efficiently. (e.g., Verity)
- Often: Boolean combinations (as in Google Alerts)

# Classification methods: 2. Rule-based

- Our Google Alerts example was rule-based classification.
- There are "IDE" type development enviroments for writing very complex rules efficiently. (e.g., Verity)
- Often: Boolean combinations (as in Google Alerts)
- Accuracy is very high if a rule has been carefully refined over time by a subject expert.

# Classification methods: 2. Rule-based

- Our Google Alerts example was rule-based classification.
- There are "IDE" type development enviroments for writing very complex rules efficiently. (e.g., Verity)
- Often: Boolean combinations (as in Google Alerts)
- Accuracy is very high if a rule has been carefully refined over time by a subject expert.
- Building and maintaining rule-based classification systems is expensive.

# A Verity topic (a complex classification rule)

```
comment line                # Beginning of art topic definition
top-level topic             art ACCRUE
                                /author = "fsmith"
topic definition modifiers      /date   = "30-Dec-01"
                                /annotation = "Topic created
                                               by fsmith"
subtopictopic               * 0.70 performing-arts ACCRUE
  evidencetopic             ** 0.50 WORD
  topicdefinition modifier      /wordtext = ballet
  evidencetopic             ** 0.50 STEM
  topicdefinition modifier      /wordtext = dance
  evidencetopic             ** 0.50 WORD
  topicdefinition modifier      /wordtext = opera
  evidencetopic             ** 0.30 WORD
  topicdefinition modifier      /wordtext = symphony
subtopic                    * 0.70 visual-arts ACCRUE
                            ** 0.50 WORD
                                /wordtext = painting
                            ** 0.50 WORD
                                /wordtext = sculpture
subtopic                    * 0.70 film ACCRUE
                            ** 0.50 STEM
                                /wordtext = film
subtopic                    ** 0.50 motion-picture PHRASE
                            *** 1.00 WORD
                                /wordtext = motion
                            *** 1.00 WORD
                                /wordtext = picture
                            ** 0.50 STEM
                                /wordtext = movie
subtopic                    * 0.50 video ACCRUE
                            ** 0.50 STEM
                                /wordtext = video
                            ** 0.50 STEM
                                /wordtext = vcr
                            # End of art topic
```

# Classification methods: 3. Statistical/Probabilistic

- As per our definition of the classification problem – text classification as a learning problem

# Classification methods: 3. Statistical/Probabilistic

- As per our definition of the classification problem – text classification as a learning problem
- Supervised learning of a the classification function $\gamma$ and its application to classifying new documents

# Classification methods: 3. Statistical/Probabilistic

- As per our definition of the classification problem – text classification as a learning problem
- Supervised learning of a the classification function $\gamma$ and its application to classifying new documents
- We will look at a couple of methods for doing this: Naive Bayes, Rocchio, kNN

# Classification methods: 3. Statistical/Probabilistic

- As per our definition of the classification problem – text classification as a learning problem
- Supervised learning of a the classification function $\gamma$ and its application to classifying new documents
- We will look at a couple of methods for doing this: Naive Bayes, Rocchio, kNN
- No free lunch: requires hand-classified training data

# Classification methods: 3. Statistical/Probabilistic

- As per our definition of the classification problem – text classification as a learning problem
- Supervised learning of a the classification function $\gamma$ and its application to classifying new documents
- We will look at a couple of methods for doing this: Naive Bayes, Rocchio, kNN
- No free lunch: requires hand-classified training data
- But this manual classification can be done by non-experts.

# Outline

1. Text classification
2. Naive Bayes
3. Evaluation of TC
4. NB independence assumptions

# The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.

# The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We compute the probability of a document $d$ being in a class $c$ as follows:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

# The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We compute the probability of a document $d$ being in a class $c$ as follows:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

- $P(t_k|c)$ is the conditional probability of term $t_k$ occurring in a document of class $c$

# The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We compute the probability of a document $d$ being in a class $c$ as follows:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

- $P(t_k|c)$ is the conditional probability of term $t_k$ occurring in a document of class $c$
- $P(t_k|c)$ as a measure of how much evidence $t_k$ contributes that $c$ is the correct class.

# The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We compute the probability of a document $d$ being in a class $c$ as follows:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

- $P(t_k|c)$ is the conditional probability of term $t_k$ occurring in a document of class $c$
- $P(t_k|c)$ as a measure of how much evidence $t_k$ contributes that $c$ is the correct class.
- $P(c)$ is the prior probability of $c$.

# The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We compute the probability of a document $d$ being in a class $c$ as follows:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

- $P(t_k|c)$ is the conditional probability of term $t_k$ occurring in a document of class $c$
- $P(t_k|c)$ as a measure of how much evidence $t_k$ contributes that $c$ is the correct class.
- $P(c)$ is the prior probability of $c$.
- If a document's terms do not provide clear evidence for one class vs. another, we choose the one that has a higher prior probability.

# Maximum a posteriori class

- Our goal is to find the “best” class.

# Maximum a posteriori class

- Our goal is to find the "best" class.
- The best class in Naive Bayes classification is the most likely or maximum a posteriori (MAP) class $c_{\text{map}}$:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \hat{P}(c|d) = \arg\max_{c \in \mathbb{C}} \ \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)$$

# Maximum a posteriori class

- Our goal is to find the "best" class.
- The best class in Naive Bayes classification is the most likely or maximum a posteriori (MAP) class $c_{\text{map}}$:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \hat{P}(c|d) = \arg\max_{c \in \mathbb{C}} \ \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)$$

- We write $\hat{P}$ for $P$ since these values are estimates from the training set.

# Taking the log

- Multiplying lots of small probabilities can result in floating point underflow.

# Taking the log

- Multiplying lots of small probabilities can result in floating point underflow.
- Since $\log(xy) = \log(x) + \log(y)$, we can sum log probabilities instead of multiplying probabilities.

# Taking the log

- Multiplying lots of small probabilities can result in floating point underflow.
- Since $\log(xy) = \log(x) + \log(y)$, we can sum log probabilities instead of multiplying probabilities.
- Since log is a monotonic function, the class with the highest score does not change.

# Taking the log

- Multiplying lots of small probabilities can result in floating point underflow.
- Since $\log(xy) = \log(x) + \log(y)$, we can sum log probabilities instead of multiplying probabilities.
- Since log is a monotonic function, the class with the highest score does not change.
- So what we usually compute in practice is:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

# Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \operatorname*{arg\,max}_{c \in \mathbb{C}} \, [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

# Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \, [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

- Simple interpretation:

# Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \, [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator $t_k$ is for $c$.

# Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \, [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator $t_k$ is for $c$.
  - The prior $\log \hat{P}(c)$ is a weight that indicates the relative frequency of $c$.

# Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \left[ \log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c) \right]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator $t_k$ is for $c$.
  - The prior $\log \hat{P}(c)$ is a weight that indicates the relative frequency of $c$.
  - The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class.

# Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} [\log \hat{P}(c) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|c)]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator $t_k$ is for $c$.
  - The prior $\log \hat{P}(c)$ is a weight that indicates the relative frequency of $c$.
  - The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class.
  - We select the class with the most evidence.

# Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \underset{c \in \mathbb{C}}{\arg\max}\, [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

- Simple interpretation:
  - Each conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator $t_k$ is for $c$.
  - The prior $\log \hat{P}(c)$ is a weight that indicates the relative frequency of $c$.
  - The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class.
  - We select the class with the most evidence.
- Questions?

# Parameter estimation

- How to estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from training data?

# Parameter estimation

- How to estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from training data?
- Prior:

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

# Parameter estimation

- How to estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from training data?
- Prior:

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

- $N_c$: number of docs in class $c$; $N$: total number of docs

# Parameter estimation

- How to estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from training data?
- Prior:

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

- $N_c$: number of docs in class $c$; $N$: total number of docs
- Conditional probabilities:

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

# Parameter estimation

- How to estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from training data?
- Prior:

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

- $N_c$: number of docs in class $c$; $N$: total number of docs
- Conditional probabilities:

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

- $T_{ct}$ is the number of tokens of $t$ in training documents from class $c$ (includes multiple occurrences)

# Parameter estimation

- How to estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from training data?
- Prior:

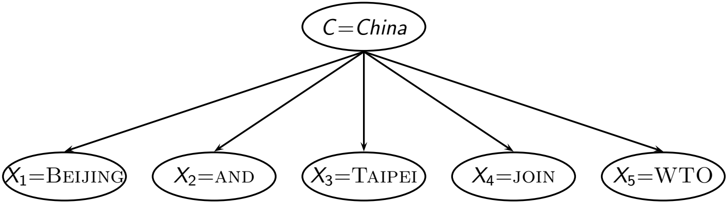
$$\hat{P}(c) = \frac{N_c}{N}$$

- $N_c$: number of docs in class $c$; $N$: total number of docs
- Conditional probabilities:

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

- $T_{ct}$ is the number of tokens of $t$ in training documents from class $c$ (includes multiple occurrences)
- We've made a Naive Bayes independence assumption here: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$

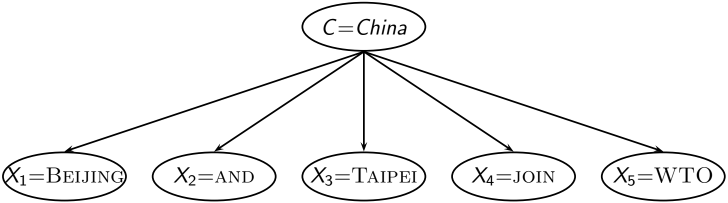
# The problem with maximum likelihood estimates: Zeros




- In this example:

$P(China|d) \propto P(China)P(\text{BEIJING}|China)P(\text{AND}|China)P(\text{TAIPEI}|China)P(\text{JOIN}|China)P($

# The problem with maximum likelihood estimates: Zeros



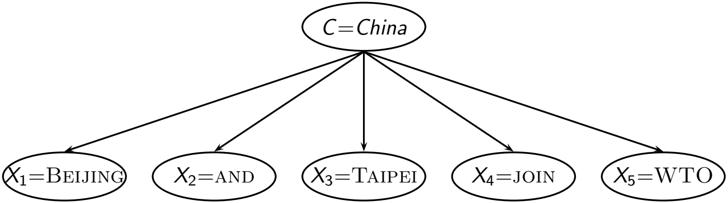

- In this example:

  $P(China|d) \propto P(China)P(\text{BEIJING}|China)P(\text{AND}|China)P(\text{TAIPEI}|China)P(\text{JOIN}|China)P($

- If there were no occurrences of WTO in documents in class China, we get a zero estimate for the corresponding parameter:

$$\hat{P}(\text{WTO}|China) = \frac{T_{China,\text{WTO}}}{\sum_{t' \in V} T_{China,t'}} = 0$$

# The problem with maximum likelihood estimates: Zeros




- In this example:

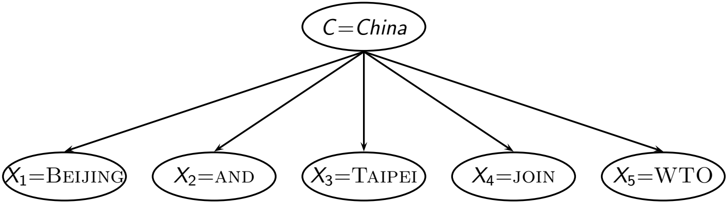
  $P(China|d) \propto P(China)P(\text{BEIJING}|China)P(\text{AND}|China)P(\text{TAIPEI}|China)P(\text{JOIN}|China)P($

- If there were no occurrences of WTO in documents in class China, we get a zero estimate for the corresponding parameter:

$$\hat{P}(\text{WTO}|China) = \frac{T_{China,\text{WTO}}}{\sum_{t' \in V} T_{China,t'}} = 0$$

- We will get $P(China|d) = 0$ for any document with WTO!

# The problem with maximum likelihood estimates: Zeros




- In this example:

  $P(China|d) \propto P(China)P(\text{Beijing}|China)P(\text{and}|China)P(\text{Taipei}|China)P(\text{join}|China)P($

- If there were no occurrences of WTO in documents in class China, we get a zero estimate for the corresponding parameter:

$$\hat{P}(\text{WTO}|China) = \frac{T_{China,\text{WTO}}}{\sum_{t' \in V} T_{China,t'}} = 0$$

- We will get $P(China|d) = 0$ for any document with WTO!
- Zero probabilities cannot be conditioned away.

# To avoid zeros: Add-one smoothing

- Add one to each count to avoid zeros:

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V}(T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B}$$

# To avoid zeros: Add-one smoothing

- Add one to each count to avoid zeros:

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V}(T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B}$$

- $B$ is the number of different words (in this case the size of the vocabulary: $|V| = M$)

# Naive Bayes: Summary

- Estimate parameters from training corpus using add-one smoothing

# Naive Bayes: Summary

- Estimate parameters from training corpus using add-one smoothing
- For a new document, for each class, compute sum of (i) log of prior and (ii) logs of conditional probabilities of the terms

# Naive Bayes: Summary

- Estimate parameters from training corpus using add-one smoothing
- For a new document, for each class, compute sum of (i) log of prior and (ii) logs of conditional probabilities of the terms
- Assign document to the class with the largest score

# Naive Bayes: Training

```
TrainMultinomialNB(C, D)
V ← ExtractVocabulary(D)
N ← CountDocs(D)
for each c ∈ C
do N_c ← CountDocsInClass(D, c)
    prior[c] ← N_c / N
    text_c ← ConcatenateTextOfAllDocsInClass(D, c)
    for each t ∈ V
    do T_ct ← CountTokensOfTerm(text_c, t)
    for each t ∈ V
    do condprob[t][c] ← (T_ct + 1) / Σ_t' (T_ct' + 1)
return V, prior, condprob
```

# Naive Bayes: Testing

ApplyMultinomialNB($\mathbb{C}$, $V$, *prior*, *condprob*, $d$)

$W \leftarrow$ ExtractTokensFromDoc($V$, $d$)
**for each** $c \in \mathbb{C}$
**do** $score[c] \leftarrow \log prior[c]$
  **for each** $t \in W$
  **do** $score[c] += \log condprob[t][c]$
**return** $\arg\max_{c \in \mathbb{C}} score[c]$

# Example: Data

| | docID | words in document | in $c = China$? |
|---|---|---|---|
| training set | 1 | Chinese Beijing Chinese | yes |
| | 2 | Chinese Chinese Shanghai | yes |
| | 3 | Chinese Macao | yes |
| | 4 | Tokyo Japan Chinese | no |
| test set | 5 | Chinese Chinese Chinese Tokyo Japan | ? |

# Example: Parameter estimates

Priors: $\hat{P}(c) = 3/4$ and $\hat{P}(\overline{c}) = 1/4$
Conditional probabilities:

$$\begin{aligned}
\hat{P}(\text{Chinese}|c) &= (5+1)/(8+6) = 6/14 = 3/7 \\
\hat{P}(\text{Tokyo}|c) = \hat{P}(\text{Japan}|c) &= (0+1)/(8+6) = 1/14 \\
\hat{P}(\text{Chinese}|\overline{c}) &= (1+1)/(3+6) = 2/9 \\
\hat{P}(\text{Tokyo}|\overline{c}) = \hat{P}(\text{Japan}|\overline{c}) &= (1+1)/(3+6) = 2/9
\end{aligned}$$

The denominators are $(8+6)$ and $(3+6)$ because the lengths of $text_c$ and $text_{\overline{c}}$ are 8 and 3, respectively, and because the constant $B$ is 6 as the vocabulary consists of six terms.

# Example: Classification

$$\hat{P}(c|d_5) \propto 3/4 \cdot (3/7)^3 \cdot 1/14 \cdot 1/14 \approx 0.0003$$
$$\hat{P}(\overline{c}|d_5) \propto 1/4 \cdot (2/9)^3 \cdot 2/9 \cdot 2/9 \approx 0.0001$$

Thus, the classifier assigns the test document to $c = China$. The reason for this classification decision is that the three occurrences of the positive indicator CHINESE in $d_5$ outweigh the occurrences of the two negative indicators JAPAN and TOKYO.

# Time complexity of Naive Bayes

| mode | time complexity |
|---|---|
| training | $\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}} + \lvert\mathbb{C}\rvert\lvert V\rvert)$ |
| testing | $\Theta(L_{\text{a}} + \lvert\mathbb{C}\rvert M_{\text{a}}) = \Theta(\lvert\mathbb{C}\rvert M_{\text{a}})$ |

- $L_{\text{ave}}$: the average length of a doc, $L_{\text{a}}$: length of the test doc, $M_{\text{a}}$: number of distinct terms in the test doc

# Time complexity of Naive Bayes

| mode | time complexity |
|---|---|
| training | $\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}} + \lvert\mathbb{C}\rvert\lvert V\rvert)$ |
| testing | $\Theta(L_{\text{a}} + \lvert\mathbb{C}\rvert M_{\text{a}}) = \Theta(\lvert\mathbb{C}\rvert M_{\text{a}})$ |

- $L_{\text{ave}}$: the average length of a doc, $L_{\text{a}}$: length of the test doc, $M_{\text{a}}$: number of distinct terms in the test doc
- $\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}})$ is the time it takes to compute all counts.

# Time complexity of Naive Bayes

| mode | time complexity |
|---|---|
| training | $\Theta(\vert\mathbb{D}\vert L_{\text{ave}} + \vert\mathbb{C}\vert\vert V\vert)$ |
| testing | $\Theta(L_{\text{a}} + \vert\mathbb{C}\vert M_{\text{a}}) = \Theta(\vert\mathbb{C}\vert M_{\text{a}})$ |

- $L_{\text{ave}}$: the average length of a doc, $L_{\text{a}}$: length of the test doc, $M_{\text{a}}$: number of distinct terms in the test doc
- $\Theta(|\mathbb{D}|L_{\text{ave}})$ is the time it takes to compute all counts.
- $\Theta(|\mathbb{C}||V|)$ is the time it takes to compute the parameters from the counts.

# Time complexity of Naive Bayes

| mode | time complexity |
|---|---|
| training | $\Theta(\vert\mathbb{D}\vert L_{\text{ave}} + \vert\mathbb{C}\vert\vert V\vert)$ |
| testing | $\Theta(L_{\text{a}} + \vert\mathbb{C}\vert M_{\text{a}}) = \Theta(\vert\mathbb{C}\vert M_{\text{a}})$ |

- $L_{\text{ave}}$: the average length of a doc, $L_{\text{a}}$: length of the test doc, $M_{\text{a}}$: number of distinct terms in the test doc
- $\Theta(|\mathbb{D}|L_{\text{ave}})$ is the time it takes to compute all counts.
- $\Theta(|\mathbb{C}||V|)$ is the time it takes to compute the parameters from the counts.
- Generally: $|\mathbb{C}||V| < |\mathbb{D}|L_{\text{ave}}$

# Time complexity of Naive Bayes

| mode | time complexity |
|---|---|
| training | $\Theta(\|\mathbb{D}\|L_{\text{ave}} + \|\mathbb{C}\|\|V\|)$ |
| testing | $\Theta(L_{\text{a}} + \|\mathbb{C}\|M_{\text{a}}) = \Theta(\|\mathbb{C}\|M_{\text{a}})$ |

- $L_{\text{ave}}$: the average length of a doc, $L_{\text{a}}$: length of the test doc, $M_{\text{a}}$: number of distinct terms in the test doc
- $\Theta(|\mathbb{D}|L_{\text{ave}})$ is the time it takes to compute all counts.
- $\Theta(|\mathbb{C}||V|)$ is the time it takes to compute the parameters from the counts.
- Generally: $|\mathbb{C}||V| < |\mathbb{D}|L_{\text{ave}}$
- Why?

# Time complexity of Naive Bayes

| mode | time complexity |
|---|---|
| training | $\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}} + \lvert\mathbb{C}\rvert\lvert V\rvert)$ |
| testing | $\Theta(L_{\text{a}} + \lvert\mathbb{C}\rvert M_{\text{a}}) = \Theta(\lvert\mathbb{C}\rvert M_{\text{a}})$ |

- $L_{\text{ave}}$: the average length of a doc, $L_{\text{a}}$: length of the test doc, $M_{\text{a}}$: number of distinct terms in the test doc
- $\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}})$ is the time it takes to compute all counts.
- $\Theta(\lvert\mathbb{C}\rvert\lvert V\rvert)$ is the time it takes to compute the parameters from the counts.
- Generally: $\lvert\mathbb{C}\rvert\lvert V\rvert < \lvert\mathbb{D}\rvert L_{\text{ave}}$
- Why?
- Test time is also linear (in the length of the test document).

# Time complexity of Naive Bayes

| mode | time complexity |
|---|---|
| training | $\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}} + \lvert\mathbb{C}\rvert\lvert V\rvert)$ |
| testing | $\Theta(L_{\text{a}} + \lvert\mathbb{C}\rvert M_{\text{a}}) = \Theta(\lvert\mathbb{C}\rvert M_{\text{a}})$ |

- $L_{\text{ave}}$: the average length of a doc, $L_{\text{a}}$: length of the test doc, $M_{\text{a}}$: number of distinct terms in the test doc
- $\Theta(\lvert\mathbb{D}\rvert L_{\text{ave}})$ is the time it takes to compute all counts.
- $\Theta(\lvert\mathbb{C}\rvert\lvert V\rvert)$ is the time it takes to compute the parameters from the counts.
- Generally: $\lvert\mathbb{C}\rvert\lvert V\rvert < \lvert\mathbb{D}\rvert L_{\text{ave}}$
- Why?
- Test time is also linear (in the length of the test document).
- Thus: Naive Bayes is linear in the size of the training set (training) and the test document (testing). This is optimal.

# Naive Bayes: Analysis

- Now we want to gain a better understanding of the properties of Naive Bayes.

# Naive Bayes: Analysis

- Now we want to gain a better understanding of the properties of Naive Bayes.
- We will formally derive the classification rule . . .

# Naive Bayes: Analysis

- Now we want to gain a better understanding of the properties of Naive Bayes.
- We will formally derive the classification rule . . .
- . . . and state the assumptions we make in that derivation explicitly.

# Derivation of Naive Bayes rule

We want to find the class that is most likely given the document:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} P(c|d)$$

Apply Bayes rule $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \frac{P(d|c)P(c)}{P(d)}$$

Drop denominator since $P(d)$ is the same for all classes:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} P(d|c)P(c)$$

# Too many parameters / sparseness

$$
\begin{aligned}
c_{\text{map}} &= \underset{c \in \mathbb{C}}{\arg\max}\ P(d|c)P(c) \\
&= \underset{c \in \mathbb{C}}{\arg\max}\, P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)P(c)
\end{aligned}
$$

Why can't we use this to make an actual classification decision?

# Too many parameters / sparseness

$$
\begin{aligned}
c_{\text{map}} &= \arg\max_{c \in \mathbb{C}} \; P(d|c)P(c) \\
&= \arg\max_{c \in \mathbb{C}} P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)P(c)
\end{aligned}
$$

Why can't we use this to make an actual classification decision?

- There are two many parameters $P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)$, one for each unique combination of a class and a sequence of words.

# Too many parameters / sparseness

$$
\begin{aligned}
c_{\text{map}} &= \arg\max_{c \in \mathbb{C}} \; P(d|c)P(c) \\
&= \arg\max_{c \in \mathbb{C}} P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)P(c)
\end{aligned}
$$

Why can't we use this to make an actual classification decision?

- There are two many parameters $P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)$, one for each unique combination of a class and a sequence of words.
- We would need a very, very large number of training examples to estimate that many parameters.

# Too many parameters / sparseness

$$\begin{aligned} c_{\text{map}} &= \underset{c \in \mathbb{C}}{\arg\max}\ P(d|c)P(c) \\ &= \underset{c \in \mathbb{C}}{\arg\max}\, P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)P(c) \end{aligned}$$

Why can't we use this to make an actual classification decision?

- There are two many parameters $P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)$, one for each unique combination of a class and a sequence of words.
- We would need a very, very large number of training examples to estimate that many parameters.
- This the problem of data sparseness.

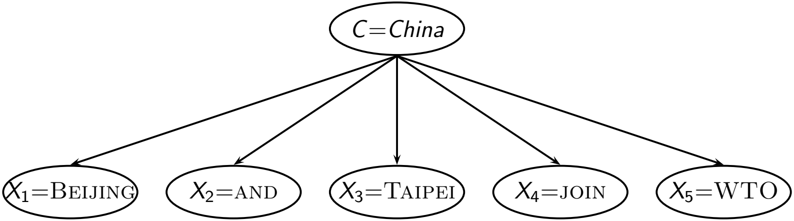
# Naive Bayes conditional independence assumption

To reduce the number of parameters to a manageable size, we make the Naive Bayes conditional independence assumption:

$$P(d|c) = P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

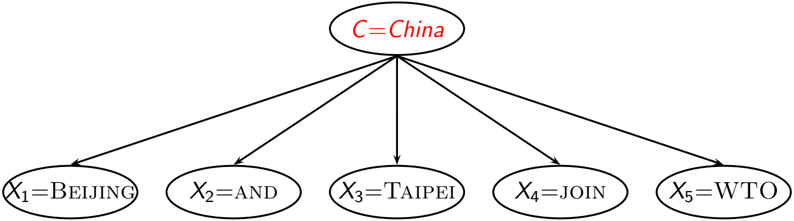
We assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(X_k = t_k|c)$.
Recall from earlier the estimates for these priors and conditional probabilities: $\hat{P}(c) = \frac{N_c}{N}$ and $\hat{P}(t|c) = \frac{T_{ct}+1}{(\sum_{t' \in V} T_{ct'})+B}$

# Generative model




$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$

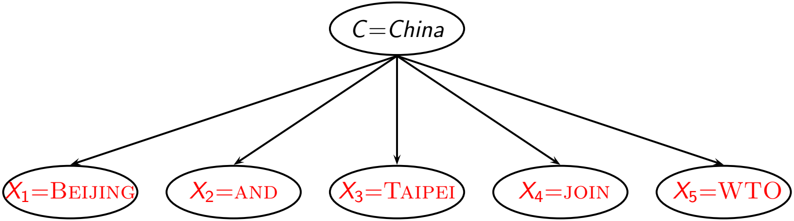
# Generative model




$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$

- Generate a class with probability $P(c)$

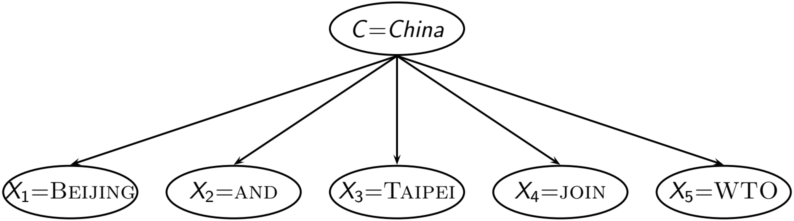
# Generative model




$P(c|d) \propto P(c) \prod_{1 \le k \le n_d} P(t_k|c)$

- Generate a class with probability $P(c)$
- Generate each of the words (in their respective positions), conditional on the class, but independent of each other, with probability $P(t_k|c)$

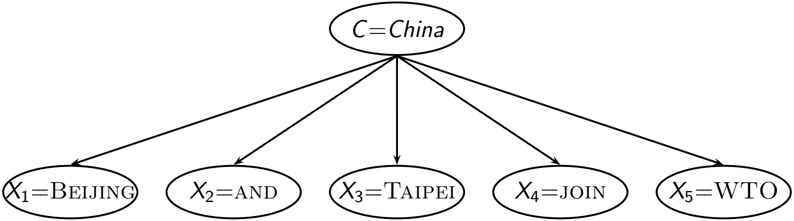
# Generative model




$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$

- Generate a class with probability $P(c)$
- Generate each of the words (in their respective positions), conditional on the class, but independent of each other, with probability $P(t_k|c)$
- To classify docs, we "reengineer" this process and find the class that is most likely to have generated the doc.

# Generative model




$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$

- Generate a class with probability $P(c)$
- Generate each of the words (in their respective positions), conditional on the class, but independent of each other, with probability $P(t_k|c)$
- To classify docs, we "reengineer" this process and find the class that is most likely to have generated the doc.
- Questions?

# Second independence assumption

- $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$

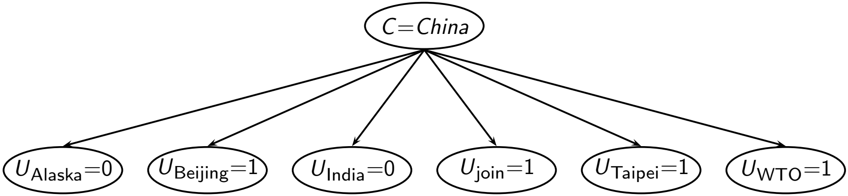
# Second independence assumption

- $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- For example, for a document in the class *UK*, the probability of generating QUEEN in the first position of the document is the same as generating it in the last position.

# Second independence assumption

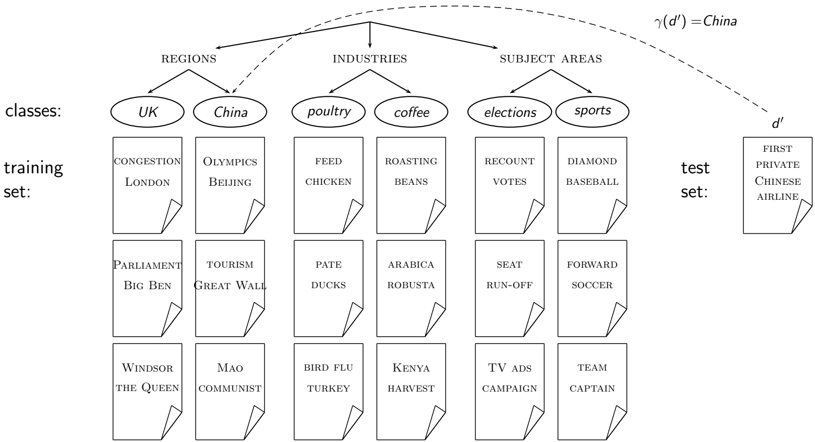
- $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- For example, for a document in the class *UK*, the probability of generating QUEEN in the first position of the document is the same as generating it in the last position.
- The two independence assumptions amount to the bag of words model.

# A different Naive Bayes model: Bernoulli model



# Outline

# Evaluation on Reuters



# Example: The Reuters collection

| symbol | statistic | value |
|---|---|---|
| $N$ | documents | 800,000 |
| $L$ | avg. # word tokens per document | 200 |
| $M$ | word types | 400,000 |
| | avg. # bytes per word token (incl. spaces/punct.) | 6 |
| | avg. # bytes per word token (without spaces/punct.) | 4.5 |
| | avg. # bytes per word type | 7.5 |
| | non-positional postings | 100,000,000 |

# Example: The Reuters collection

| symbol | statistic | value |
|---|---|---|
| $N$ | documents | 800,000 |
| $L$ | avg. # word tokens per document | 200 |
| $M$ | word types | 400,000 |
| | avg. # bytes per word token (incl. spaces/punct.) | 6 |
| | avg. # bytes per word token (without spaces/punct.) | 4.5 |
| | avg. # bytes per word type | 7.5 |
| | non-positional postings | 100,000,000 |

| type of class | number | examples |
|---|---|---|
| region | 366 | UK, China |
| industry | 870 | poultry, coffee |
| subject area | 126 | elections, sports |

# A Reuters document

REUTERS

**You are here:** Home > News > Science > **Article**

Go to a Section: U.S. International Business Markets Politics Entertainment Technology Sports Oddly Enoug

## Extreme conditions create rare Antarctic clouds

Tue Aug 1, 2006 3:20am ET

Email This Article | Print This Article | Reprints

[-] Text [+]

SYDNEY (Reuters) - Rare, mother-of-pearl colored clouds caused by extreme weather conditions above Antarctica are a possible indication of global warming, Australian scientists said on Tuesday.

Known as nacreous clouds, the spectacular formations showing delicate wisps of colors were photographed in the sky over an Australian meteorological base at Mawson Station on July 25.

# Evaluating classification

- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).

# Evaluating classification

- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
- It's easy to get good performance on a test set that was available to the learner during training (e.g., just memorize the test set).

# Evaluating classification

- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
- It's easy to get good performance on a test set that was available to the learner during training (e.g., just memorize the test set).
- Measures: Precision, recall, $F_1$, classification accuracy

# Naive Bayes vs. other methods

| (a) | NB | Rocchio | kNN | SVM |
|---|---|---|---|---|
| micro-avg-L (90 classes) | 80 | 85 | 86 | 89 |
| macro-avg (90 classes) | 47 | 59 | 60 | 60 |

| (b) | NB | Rocchio | kNN | trees | SVM |
|---|---|---|---|---|---|
| earn | 96 | 93 | 97 | 98 | 98 |
| acq | 88 | 65 | 92 | 90 | 94 |
| money-fx | 57 | 47 | 78 | 66 | 75 |
| grain | 79 | 68 | 82 | 85 | 95 |
| crude | 80 | 70 | 86 | 85 | 89 |
| trade | 64 | 65 | 77 | 73 | 76 |
| interest | 65 | 63 | 74 | 67 | 78 |
| ship | 85 | 49 | 79 | 74 | 86 |
| wheat | 70 | 69 | 77 | 93 | 92 |
| corn | 65 | 48 | 78 | 92 | 90 |
| micro-avg (top 10) | 82 | 65 | 82 | 88 | 92 |
| micro-avg-D (118 classes) | 75 | 62 | n/a | n/a | 87 |

Evaluation measure: $F_1$

# Naive Bayes vs. other methods

(a)

| | NB | Rocchio | kNN | | SVM |
|---|---|---|---|---|---|
| micro-avg-L (90 classes) | 80 | 85 | 86 | | 89 |
| macro-avg (90 classes) | 47 | 59 | 60 | | 60 |

(b)

| | NB | Rocchio | kNN | trees | SVM |
|---|---|---|---|---|---|
| earn | 96 | 93 | 97 | 98 | 98 |
| acq | 88 | 65 | 92 | 90 | 94 |
| money-fx | 57 | 47 | 78 | 66 | 75 |
| grain | 79 | 68 | 82 | 85 | 95 |
| crude | 80 | 70 | 86 | 85 | 89 |
| trade | 64 | 65 | 77 | 73 | 76 |
| interest | 65 | 63 | 74 | 67 | 78 |
| ship | 85 | 49 | 79 | 74 | 86 |
| wheat | 70 | 69 | 77 | 93 | 92 |
| corn | 65 | 48 | 78 | 92 | 90 |
| micro-avg (top 10) | 82 | 65 | 82 | 88 | 92 |
| micro-avg-D (118 classes) | 75 | 62 | n/a | n/a | 87 |

Evaluation measure: $F_1$

Naive Bayes does pretty well, but some methods beat it consistently (e.g., SVM).

# Outline

# Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.

# Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

# Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

- Examples for why this assumption is not really true?

# Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

- Examples for why this assumption is not really true?
- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$

# Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

- Examples for why this assumption is not really true?
- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- Examples for why this assumption is not really true?

# Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

- Examples for why this assumption is not really true?
- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- Examples for why this assumption is not really true?
- How can Naive Bayes work if it makes such inappropriate assumptions?

# Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.

# Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.
- Example:

| | $c_1$ | $c_2$ | class selected |
|---|---|---|---|
| true probability $P(c\|d)$ | 0.6 | 0.4 | $c_1$ |
| $\hat{P}(c)\prod_{1\le k\le n_d}\hat{P}(t_k\|c)$ | 0.00099 | 0.00001 | |
| NB estimate $\hat{P}(c\|d)$ | 0.99 | 0.01 | $c_1$ |

# Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.
- Example:

| | $c_1$ | $c_2$ | class selected |
|---|---|---|---|
| true probability $P(c\|d)$ | 0.6 | 0.4 | $c_1$ |
| $\hat{P}(c)\prod_{1\le k\le n_d}\hat{P}(t_k\|c)$ | 0.00099 | 0.00001 | |
| NB estimate $\hat{P}(c\|d)$ | 0.99 | 0.01 | $c_1$ |

- Double counting of evidence causes underestimation (0.01) and overestimation (0.99).

# Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.
- Example:

| | $c_1$ | $c_2$ | class selected |
|---|---|---|---|
| true probability $P(c\|d)$ | 0.6 | 0.4 | $c_1$ |
| $\hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k\|c)$ | 0.00099 | 0.00001 | |
| NB estimate $\hat{P}(c\|d)$ | 0.99 | 0.01 | $c_1$ |

- Double counting of evidence causes underestimation (0.01) and overestimation (0.99).
- Classification is about predicting the correct class and not about accurately estimating probabilities.

# Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.
- Example:

| | $c_1$ | $c_2$ | class selected |
|---|---|---|---|
| true probability $P(c\|d)$ | 0.6 | 0.4 | $c_1$ |
| $\hat{P}(c)\prod_{1\leq k\leq n_d}\hat{P}(t_k\|c)$ | 0.00099 | 0.00001 | |
| NB estimate $\hat{P}(c\|d)$ | 0.99 | 0.01 | $c_1$ |

- Double counting of evidence causes underestimation (0.01) and overestimation (0.99).
- Classification is about predicting the correct class and not about accurately estimating probabilities.
- Correct estimation $\Rightarrow$ accurate prediction.

# Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.
- Example:

| | $c_1$ | $c_2$ | class selected |
|---|---|---|---|
| true probability $P(c\|d)$ | 0.6 | 0.4 | $c_1$ |
| $\hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k\|c)$ | 0.00099 | 0.00001 | |
| NB estimate $\hat{P}(c\|d)$ | 0.99 | 0.01 | $c_1$ |

- Double counting of evidence causes underestimation (0.01) and overestimation (0.99).
- Classification is about predicting the correct class and not about accurately estimating probabilities.
- Correct estimation $\Rightarrow$ accurate prediction.
- But not vice versa!

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features
- A good dependable baseline for text classification (but not the best)

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features
- A good dependable baseline for text classification (but not the best)
- Optimal if independence assumptions hold (never true for text, but true for some domains)

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features
- A good dependable baseline for text classification (but not the best)
- Optimal if independence assumptions hold (never true for text, but true for some domains)
- Very fast

# Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features
- A good dependable baseline for text classification (but not the best)
- Optimal if independence assumptions hold (never true for text, but true for some domains)
- Very fast
- Low storage requirements

# Resources

- Chapter 13 of IIR

# Resources

- Chapter 13 of IIR
- Resources at `http://ifnlp.org/ir`

# Resources

- Chapter 13 of IIR
- Resources at `http://ifnlp.org/ir`
- Calais: Automatic Semantic Tagging

# Resources

- Chapter 13 of IIR
- Resources at `http://ifnlp.org/ir`
- Calais: Automatic Semantic Tagging
- Weka: A data mining software package that includes an implementation of Naive Bayes

# Resources

- Chapter 13 of IIR
- Resources at `http://ifnlp.org/ir`
- Calais: Automatic Semantic Tagging
- Weka: A data mining software package that includes an implementation of Naive Bayes
- Reuters-21578 – the most famous text classification evaluation set (but now it's too small for realistic experiments)