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

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 $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$
  - 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 $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$
  - 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 $\langle d, c \rangle \in \mathbb{X} \times \mathbb{C}$

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 γ that maps documents to classes:

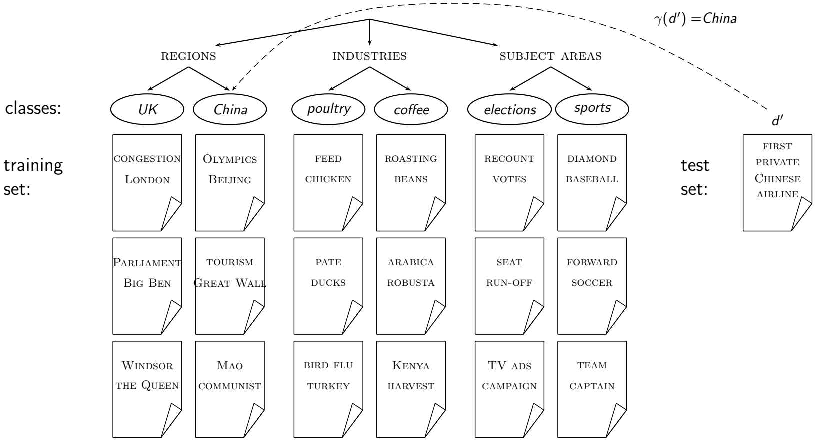
$$\gamma : \mathbb{X} \to \mathbb{C}$$

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



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
topic definition modifiers	/author = "fsmith"
	/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 γ 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 γ 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 γ 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 γ 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

文本分类/聚类（text classification/clustering）

- 朴素贝叶斯方法（Naïve Bayes）
- Question:如何计算条件概率p (c_i |d)?
- 利用概率论里的Bayes公式：

$$p(c_i \mid d) = \frac{p(c_i, d)}{p(d)} = \frac{p(d \mid c_i)\, p(c_i)}{p(d)}$$

- 如何理解Bayes公式？仅仅是数学公式的变换吗？
- 深刻理解Bayes公式是理解机器学习算法的重要基础，我认为Bayes公式是机器学习理论里最重要的基础

文本分类/聚类（text classification/clustering）

- Bayes公式
- Bayes公式实际上陈述了下列事实

$$p(c_i \mid d) = \frac{p(d \mid c_i)p(c_i)}{p(d)} \Longrightarrow posterior = \frac{likelihood \times prior}{evidence}$$

p (c_i|d): 后验概率或条件概率(posterior) p (c_i): 先验概率(prior)

p (d|c_i): 似然概率(likelihood) p (d): 证据(evidence)

- Bayes公式的意义：
  - 当观察到evidence p (d) 时，后验概率p (c_i|d)取决于似然概率p (d|c_i)和先验概率p (c_i)。因为当evidence p (d) 已知时，p (d)成为常量，Bayes公式变成

$$p(c_i \mid d) = \frac{p(d \mid c_i)p(c_i)}{p(d)} \propto p(d \mid c_i)p(c_i)$$

  - 符号$\propto$表示成正比（be proportional to）
  - 因为我们实际上关心的是p (c_i|d)的相对大小

文本分类/聚类（text classification/clustering）

- Bayes公式
- 当先验概率p (c_1)=p (c_2)=…=p (c_j) 时，公式

$$p(c_i \mid d) = \frac{p(d \mid c_i)\,p(c_i)}{p(d)} \propto p(d \mid c_i)\,p(c_i)$$

变成

$$p(c_i \mid d) \propto p(d \mid c_i)$$

- 这时给定文档d，该文档属于类别c_i的概率p (c_i |d)取决于似然概率p (d|c_i)
  - p (d|c_i)的涵义：给定文档类别c_i，由类别c_i产生文档d的可能性（likelihood）。
  - 如果类别c_i产生文档d的可能性p (d|c_i)最大，则文档d属于类别c_i的概率p (c_i |d)最大。这叫最大似然估计（Maximum Likelihood Estimation，MLE）。
  - 举例：医生根据病人的症状d要判断是否为心脏病(c_i=心脏病),。假设医生不知道其他知识，仅知道如果得了心脏病，有最大可能会出现症状d，那么医生就认为病人得心脏病的可能性最大。因此p (d| c_i=心脏病)为似然概率(likelihood)。当似然概率p (d| c_i=心脏病)最大时，医生就认为根据症状d，病人得心脏病的概率p (c_i=心脏病|d)最大。这就是Bayes公式的意义。

文本分类/聚类（text classification/clustering）

- 朴素贝叶斯方法（Naïve Bayes）
- 现在再回到Naïve Bayes分类器

$$p(c_i \mid d) = \frac{p(d \mid c_i)p(c_i)}{p(d)} \propto p(d \mid c_i)p(c_i)$$

- 对于类标签集合C 中的每个类标签c_i (i = 1, …, j), 计算条件概率p (c_i |d)，使条件概率p (c_i |d)最大的类别作为文档d最终的类别。

$$c_d = \arg\max_{c_i \in \mathrm{C}} p(c_i \mid d) = \arg\max_{c_i \in \mathrm{C}} p(d \mid c_i)p(c_i)$$

- 其中c_d为文档d被赋予的类型， c_d=使得条件概率p (c_i |d)最大的类型。根据Bayes公式， c_d=使得p (d | c_i) p (c_i)值最大的类型
- 剩下的问题是如何得到p (d | c_i) 和p (c_i)?
- 别忘了训练集和机器学习！
  - 对于Naïve Bayes，用训练集对机器进行训练就是为了算出p (d | c_i) 和p (c_i)
  - 训练的过程就是参数估计的过程。这里要估计的参数就是p (d | c_i) 和p (c_i)

文本分类/聚类（text classification/clustering）

- Naïve Bayes的参数估计
- 假设类别标签集合$C = \{c_1, c_2, \ldots, c_j\}$
- 假设训练集D包含N个文档，其中每个文档都被标上了类别标签
- 首先估计先验概率$p(c_i)$ $(i=1, \ldots, j)$

$$p(c_i) = \frac{\text{类型为 } c_i \text{的文档个数}}{\text{训练集中文档总数} \quad N}$$

文本分类/聚类（text classification/clustering）

- Naïve Bayes的参数估计
- 假设类别标签集合C = {c_1,c_2, …, c_j}
- 假设训练集D包含N个文档，其中每个文档都被标上了类别标签
- 还需要估计似然概率p (d | c_i) (i=1, …, j)
- 为了估计p (d | c_i)， 需要一个假设：Term独立性假设
  - 文档中每个term的出现是彼此独立的
- 基于这个假设，似然概率p (d | c_i) 的估计方法如下：
  - 假设文档d包含n_d个term：t_1, t_2, …, t_{n_d}
  - 根据Term的独立性假设，有

$$p(d \mid c_i) = p(t_1, t_2, \ldots, t_{n_d} \mid c_i) = p(t_1 \mid c)\, p(t_2 \mid c) \ldots p(t_{n_d} \mid c) = \prod_{1 \le k \le n_d} p(t_k \mid c_i)$$

  - 因此，估计p (d | c_i) 就需要估计p (t_k|c_i)

$$p(t_k \mid c_i) = \frac{t_k\text{在类型为 }c_i\text{的文档中出现的次数}}{\text{在类型为 }c_i\text{的文档中出现的 term 的总数}}$$

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_{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_{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:

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

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:

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

- 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.

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.
  - 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:

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

- 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.
  - 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:

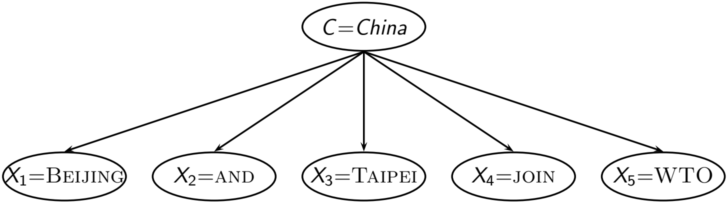
$$\hat{P}(c) = \frac{N_c}{N}$$

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

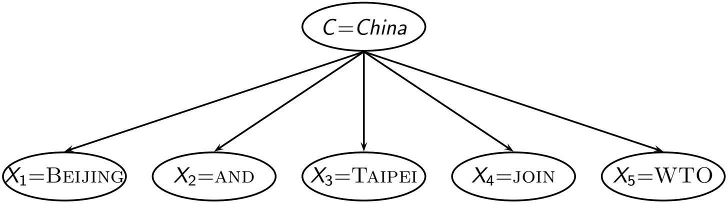
The problem with maximum likelihood estimates: Zeros



- In this example:

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

The problem with maximum likelihood estimates: Zeros



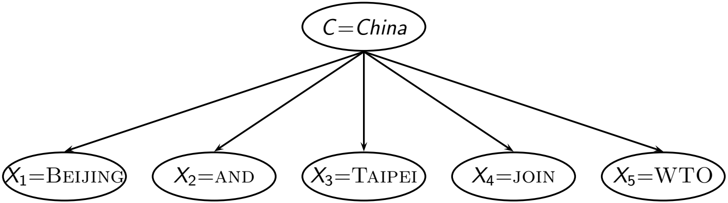
- In this example:

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

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

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

The problem with maximum likelihood estimates: Zeros



- In this example:

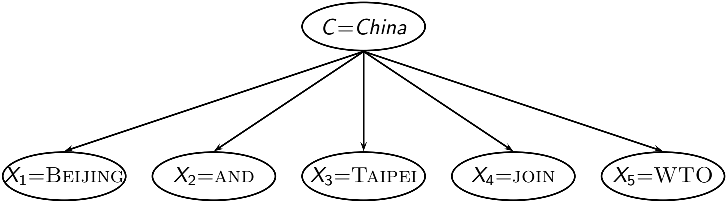
  $P(China|d) \propto P(China)P(\text{BEIJING}|China)P(\text{AND}|China)P(\text{TAIPEI}|China)P(\text{JOIN}|China)P($

- 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

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:

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

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.

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

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

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

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.

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.

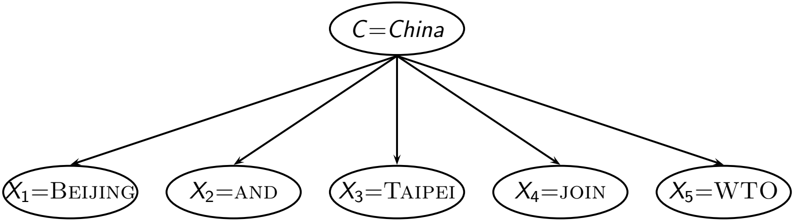
Naive Bayes conditional independence assumption

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

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

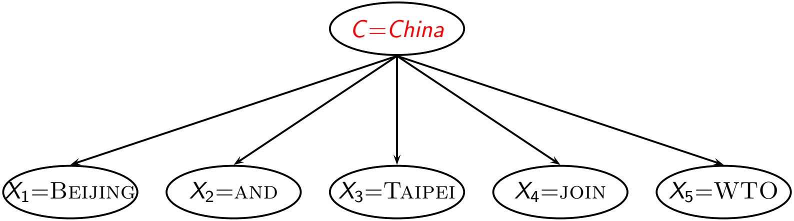
We assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(X_k = t_k|c)$.
Recall from earlier the estimates for these priors and conditional probabilities: $\hat{P}(c) = \frac{N_c}{N}$ and $\hat{P}(t|c) = \frac{T_{ct}+1}{(\sum_{t'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

Generative model



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

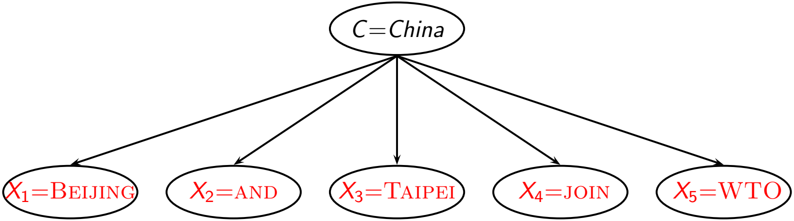
Generative model



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

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

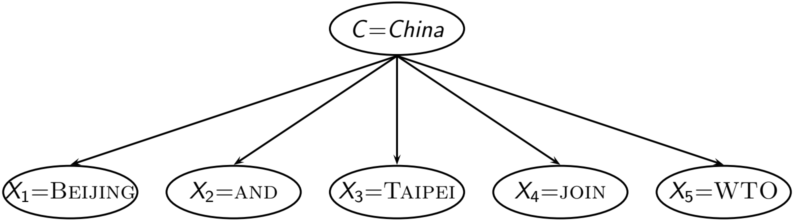
Generative model



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

- 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 $P(t_k|c)$

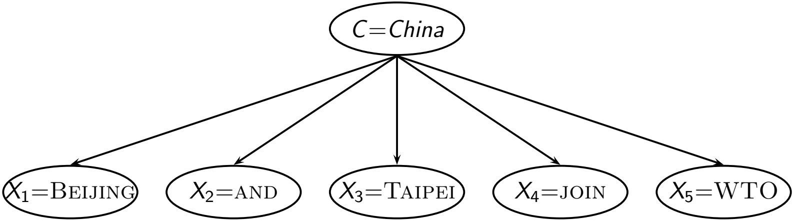
Generative model



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

- 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.

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
$\hat{P}(c)\prod_{1\leq k\leq n_d}\hat{P}(t_k\|c)$	0.00099	0.00001	
NB estimate $\hat{P}(c\|d)$	0.99	0.01	c_1

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).

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
$\hat{P}(c) \prod_{1\le k\le n_d} \hat{P}(t_k\|c)$	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).
- 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
$\hat{P}(c)\prod_{1\le k\le n_d}\hat{P}(t_k\|c)$	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).
- Classification is about predicting the correct class and not about accurately estimating probabilities.
- Correct estimation $\Rightarrow$ accurate prediction.
- But not vice versa!