Text classification I (Naïve Bayes)

CE-324: Modern Information Retrieval

Sharif University of Technology

M. Soleymani

Spring 2020

Most slides have been adapted from: Profs. Manning, Nayak & Raghavan (CS-276, Stanford)

Outline

- Text classification
  - definition
  - relevance to information retrieval
- Naïve Bayes classifier

Formal definition of text classification

- Document space X
  - Docs are represented in this (typically high-dimensional) space
- Set of classes $C = \{c_1, \dots, c_K\}$
  - Example: $C = \{\text{spam, non-spam}\}$
- Training set: a set of labeled docs. Each labeled doc $\langle d, c \rangle \in X \times C$

- Using a learning method, we find a classifier $\gamma(.)$ that maps docs to classes: $\gamma: X \to C$

Examples of using classification in IR systems

- Language identification (classes: English vs. French etc.)
- Automatic detection of spam pages (spam vs. non-spam)
- Automatic detection of secure pages for safe search
- Topic-specific or vertical search – restrict search to a "vertical" like "related to health" (relevant to vertical vs. not)
- Sentiment detection: is a movie or product review positive or negative (positive vs. negative)

- Exercise: Find examples of uses of text classification in IR

Standing queries

- The path from IR to text classification:
  - You have an information need to monitor, say:
    - Unrest in the Niger delta region
  - You want to rerun an appropriate query periodically to find new news items on this topic
  - You will be sent new documents that are found
    - I.e., it's not ranking but classification (relevant vs. not relevant)
- Such queries are called **standing queries**
  - Long used by "information professionals"
  - A modern mass instantiation is **Google Alerts**

From: Google Alerts
Subject: **Google Alert - stanford -neuro-linguistic nlp OR "Natural Language Processing" OR parser OR tagger OR ner OR "named entity" OR segmenter OR classifier OR dependencies OR "core nlp" OR corenlp OR phrasal**
Date: May 7, 2012 8:54:53 PM PDT
To: Christopher Manning

Web	3 new results for **stanford -neuro-linguistic nlp OR "Natural Language Processing" OR parser OR tagger OR ner OR "named entity" OR segmenter OR classifier OR dependencies OR "core nlp" OR corenlp OR phrasal**

Twitter / **Stanford NLP** Group: @Robertoross If you only n ...
@Robertoross If you only need tokenization, java -mx2m edu.**stanford.nlp**. process.PTBTokenizer file.txt runs in 2MB on a whole file for me.... 9:41 PM Apr 28th **...**
twitter.com/stanfordnlp/status/196459102770171905

[Java] LexicalizedParser lp = LexicalizedParser.loadModel("edu **...**
loadModel("edu/**stanford**/**nlp**/models/lexparser/englishPCFG.ser.gz");. String[] sent = { "This", "is", "an", "easy", "sentence", "." };. Tree parse = lp.apply(Arrays.
pastebin.com/az14R9nd

More Problems with Statistical **NLP** || kuro5hin.org
Tags: nlp, ai, coursera, **stanford**, **nlp**-class, cky, nltk, reinventing the wheel, **...** Programming Assignment 6 for **Stanford's nlp**-class is to implement a CKY parser .
www.kuro5hin.org/story/2012/5/5/11011/68221

Tip: Use quotes ("like this") around a set of words in your query to match them exactly. Learn more.

Delete this alert.
Create another alert.
Manage your alerts.

Spam filtering
Another text classification task

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

===

Categorization/Classification

- Given:
  - A representation of a document *d*
    - Issue: how to represent text documents.
    - Usually some type of high-dimensional space – bag of words
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$

- Determine:
  - The category of *d*: $\gamma(d) \in C$
    - $\gamma(d)$ is a classification function
  - We want to build classification functions ("classifiers").

Classification Methods (1)

- Manual classification
  - Used by the original Yahoo! Directory
  - Looksmart, about.com, ODP, **PubMed**

- Accurate when job is done by experts
- Consistent when the problem size and team is small
- Difficult and expensive to scale
  - Means we need automatic classification methods for big problems

Classification Methods (2)

- Hand-coded rule-based classifiers
  - One technique used by news agencies, intelligence agencies, etc.
  - Widely deployed in government and enterprise
  - Vendors provide “IDE” for writing such rules
- Issues:
  - Commercial systems have complex query languages
  - Accuracy can be high if a rule has been carefully refined over time by a subject expert
  - Building and maintaining these rules is expensive

Classification Methods (3): Supervised learning

- Given:
  - A document d
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set D of documents each with a label in C
- Determine:
  - A learning method or algorithm which will enable us to learn a classifier γ
  - For a test document d, we assign it the class

    $\gamma(d) \in C$

Bayes classifier

- Bayesian classifier is a probabilistic classifier:

$$c = \underset{k}{\operatorname{argmax}}\, P(C_k|d)$$
$$c = \underset{k}{\operatorname{argmax}}\, P(d|C_k)P(C_k)$$

- $d = \langle t_1, \dots, t_{L_d} \rangle$

- There are too many parameters $P(\langle t_1, \dots, t_{L_d} \rangle | C_k)$
  - 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.

Naïve bayes assumption

- Naïve bayes assumption:

$$P(d|C_k) = P\left(\langle t_1, \dots, t_{L_d} \rangle \middle| C_k\right) = \prod_{i=1}^{L_d} P(t_i|C_k)$$

  - L_d: length of doc d (number of tokens)
  - $P(t_i|C_k)$: probability of term t_i occurring in a doc of class C_k
  - $P(C_k)$: prior probability of class C_k.

Naïve bayes assumption

- Naïve bayes assumption:

$$P(d|C_k) = P\left(\langle t_1, \ldots, t_{L_d}\rangle \middle| C_k\right) = \prod_{i=1}^{L_d} P(t_i|C_k)$$

  - L_d: length of doc d (number of tokens)
  - $P(t_i|C_k)$: probability of term t_i occurring in a doc of class C_k
  - $P(C_k)$: prior probability of class C_k.

- Equivalent to (language model view):

$$P(d|C_k) = \prod_{i=1}^{|V|} P(t_i|C_k)^{tf_{t_i,d}}$$

Naive Bayes classifier

- Since log is a monotonic function, the class with the highest score does not change.

$$c = \underset{k}{\operatorname{argmax}} P(d|C_k)P(C_k) = \underset{k}{\operatorname{argmax}} P(C_k) \prod_{i=1}^{L_d} P(t_i|C_k)$$

$$c = \underset{k}{\operatorname{argmax}} \ \log P(C_k) + \sum_{i=1}^{L_d} \log P(t_i|C_k)$$

$\log P(t_i|C_k)$: a weight that indicates how good an indicator t_i is for C_k

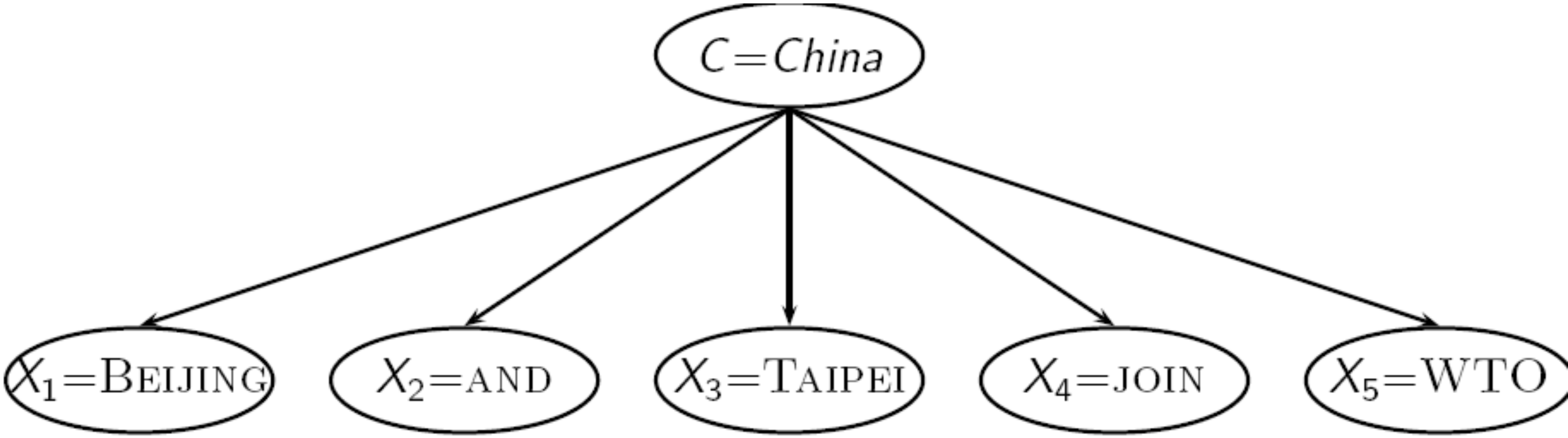
log(xy) = log(x) + log(y)

Estimating parameters

- Estimate $\hat{P}(C_k)$ and $\hat{P}(t_i|C_k)$ from training data
  - N_k: number of docs in class C_k
  - $T_{i,k}$: number of occurrence of t_i in training docs from class C_k (includes multiple occurrences)

- $\hat{P}(C_k) = \frac{N_k}{N}$
- $\hat{P}(t_i|C_k) = \frac{T_{i,k}}{\sum_{j=1}^{M} T_{j,k}}$

Problem with estimates: Zeros



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

d: $BEIGING\ AND\ TAIPEI\ JOIN\ WTO$

$$P(WTO|China) = 0$$

Problem with estimates: Zeros

- For doc d containing a term t that does not occur in any doc of a class $c \Rightarrow \hat{P}(c|d) = 0$
  - Thus d cannot be assigned to class c

- We use

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

Instead of

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

Naïve Bayes: summary

- Estimate parameters from the training corpus using add-one smoothing

- For a new doc $d = t_1, \ldots, t_{L_d}$, for each class, compute $\log P(C_k) + \sum_{i=1}^{L_d} \log P(t_i|C_k)$

- Assign doc d to the class with the largest score

Naïve Bayes: example

	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	?

- Training phase:
  - Estimate parameters of Naive Bayes classifier
- Test phase
  - Classifying the test doc

Naïve Bayes: example

$C = China$

- Estimating parameters
  - $\hat{P}(C) = \frac{3}{4}, \hat{P}(\bar{C}) = \frac{1}{4}$
  - $\hat{P}(CHINESE|C) = \frac{5+1}{8+6} = \frac{6}{14}$ $\quad \hat{P}(CHINESE|\bar{C}) = \frac{1+1}{3+6} = \frac{2}{9}$
  - $\hat{P}(TOKYO|C) = \frac{0+1}{8+6} = \frac{1}{14}$ $\quad \hat{P}(TOKYO|\bar{C}) = \frac{1+1}{3+6} = \frac{2}{9}$
  - $\hat{P}(JAPAN|C) = \frac{0+1}{8+6} = \frac{1}{14}$ $\quad \hat{P}(JAPAN|\bar{C}) = \frac{1+1}{3+6} = \frac{2}{9}$

- Classifying the test doc:
- $\hat{P}(C|d) \propto \frac{3}{4} \times \left(\frac{6}{14}\right)^3 \times \frac{1}{14} \times \frac{1}{14} \approx 0.0003$
- $\hat{P}(\bar{C}|d) \propto \frac{1}{4} \times \left(\frac{2}{9}\right)^3 \times \frac{2}{9} \times \frac{2}{9} \approx 0.0001$

$\Rightarrow \quad \hat{c} = C$

Naïve Bayes: training

TrainMultinomialNB($\mathbb{C}, \mathbb{D}$)

$V \leftarrow$ ExtractVocabulary($\mathbb{D}$)
$N \leftarrow$ CountDocs($\mathbb{D}$)
for each $c \in \mathbb{C}$
do $N_c \leftarrow$ CountDocsInClass($\mathbb{D}, c$)
    $prior[c] \leftarrow N_c / N$
    $text_c \leftarrow$ ConcatenateTextOfAllDocsInClass($\mathbb{D}, c$)
    for each $t \in V$
    do $T_{ct} \leftarrow$ CountTokensOfTerm($text_c, t$)
    for each $t \in V$
    do $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
return $V, prior, condprob$

Naïve Bayes: test

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

$W \leftarrow \textsc{ExtractTokensFromDoc}(V, d)$

for each $c \in \mathbb{C}$

do $\mathit{score}[c] \leftarrow \log \mathit{prior}[c]$

  for each $t \in W$

  do $\mathit{score}[c] += \log \mathit{condprob}[t][c]$

return $\arg\max_{c \in \mathbb{C}} \mathit{score}[c]$

Time complexity of Naive Bayes

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

Generally: $|\mathbb{C}||V| < |D|L_{ave}$

- D: training set, V: vocabulary, $\mathbb{C}$: set of classes
- L_{ave}: average length of a training doc
- L_a: length of the test doc
- M_a: number of distinct terms in the test doc

- Thus: Naive Bayes is linear in the size of the training set (training) and the test doc (testing).
  - This is optimal time.

Why does Naive Bayes work?

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

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

- Classification is about predicting the correct class and not about accurately estimating probabilities.
  - Naive Bayes is terrible for correct estimation . . .
  - but it often performs well at choosing the correct class.

Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- 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)
  - More robust to non-relevant 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
- Very fast
- Low storage requirements

Resources

- Chapter 13 of IIR