Information Retrieval

WS 2017 / **2018**

Lecture 11, Tuesday January 16th, 2018
(Classification, Naive Bayes)

Prof. Dr. Hannah Bast
Chair of Algorithms and Data Structures
Department of Computer Science
University of Freiburg

UNI FREIBURG

Overview of this lecture

- Organizational
  - Your experiences with ES10 — LSI, Eigenvectors, etc.
  - Register for the exam — Deadline is **31.01.2018**
  - Official evaluation of the course — Details on Slide 6 and ES11
- Contents
  - Classification — introduction and examples
  - Probability recap — two one-slide crash courses
  - Naïve Bayes — algorithm, example, implementation
  - ES11: learn to predict the **genre** and **rating** from a given movie description using Naïve Bayes

Your experiences with ES10   1/3

- Summary / excerpts
  - Many of you liked this exercise and found it useful
  - Many of you also found it less effort then previous exercises
  - Typesetting in LaTeX naturally took some time
  - Some confusion about reordering columns/rows of U, S, V
  - "My 'life-function' is highly non-linear and not differentiable"

    Yes, but the basic buildings blocks are linear operators!
  - "My first 'own' SVD after studying math for 7 semesters"
  - "Without eigenvectors we would be … directionless"
  - "My life w/o eigenvectors would involve a lot more alcohol"

Your experiences with ES10 2/3

Then the direction of the other EV must be $\begin{pmatrix} 3 \\ 4 \\ 0 \end{pmatrix}$

$x_2 = -\frac{3}{4}x_1$ $\quad x_1 = -\frac{4}{3}x_2$ $\quad 384x_1 + 512x_2 = 0$

$128 \cdot 3 \quad 128 \cdot 4$

$3x_1 + 4x_2 = 0$

$\Rightarrow$ Eigenvector $\begin{pmatrix} 4 \\ -3 \\ 0 \end{pmatrix}$

- Bits from master solution for ES10
  - Computing eigenvectors and eigenvalues of $A \cdot A^T$

$$A = \begin{pmatrix} 13 & 5 & 5 & 0 & 13 \\ 9 & 15 & 15 & 0 & 9 \\ 0 & 0 & 0 & 20 & 0 \end{pmatrix} \; ; \quad A \cdot A^T = \begin{pmatrix} 388 & 384 & 0 \\ 384 & 612 & 0 \\ 0 & 0 & 400 \end{pmatrix}$$

Eigenvalue $\lambda_1 = 400$

One eigenvector is obviously $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} =: x_1 \quad A \cdot A^T \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 400 \end{pmatrix}$

$$\det(A \cdot A^T - \lambda \cdot I) = \begin{vmatrix} 388-\lambda & 384 & 0 \\ 384 & 612-\lambda & 0 \\ 0 & 0 & 400-\lambda \end{vmatrix} = (400-\lambda) \cdot \underbrace{\begin{vmatrix} 388-\lambda & 384 \\ 384 & 612-\lambda \end{vmatrix}}_{(*)}$$

$$(*) = (388-\lambda)(612-\lambda) - 384^2 = \lambda^2 - \underbrace{1000}_{p+q}\lambda + \underbrace{388 \cdot 612 - 384^2}_{90.000 \;=\; p \cdot q}$$

$(\lambda - p) \cdot (\lambda - q)$

$p = 900$, $q = 100$ our other two eigenvalues

Eigenvector for $\lambda = 100$

$$\begin{pmatrix} 388 & 384 & 0 \\ 384 & 612 & 0 \\ 0 & 0 & 400 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 100 \cdot x_1 \\ 100 \cdot x_2 \\ 100 \cdot x_3 \end{pmatrix}$$

$\Rightarrow x_3 = 0$

$\Rightarrow \begin{array}{l} 388x_1 + 384x_2 = 100x_1 \\ 384x_1 + 612x_2 = 100 \cdot x_2 \end{array}$

Your experiences with ES10 3/3

- A practical demonstration of PLSI
  - PLSI is a probabilistic variant of LSI
  - PLSI computes an approximate matrix decomposition, just like LSI, but all entries in the two factors are non-negative
  - The process to obtain this decomposition is also different from LSI (no eigenvectors + a different optimization method)
  - Let's look at an example decomposition for a dataset of movie descriptions (IMDB)

Official course evaluation

- Instructions
  - You should have received an email from EvaSys Admin on Saturday, **January 13** with a link to an evaluation form
  - We are **very** interested in your feedback
  - Please take your time for this
  - Please be honest and concrete
  - The **free text comments** are most interesting for us

  Please complete until the deadline for ES11

  The evaluation is centralized, and once it is closed, there is nothing we can do to about that anymore

Classification 1/5

- Problem
  - Given **objects** and **classes**
  - Goal: given an object, predict to which class it belongs
  - To achieve that, we are given a **training set** of objects, each labeled with the class to which it belongs
  - From that we can (try to) learn which kind of objects belong to which class
  - Two examples on the next two slides

Classification 2/5

- Example 1 (natural language text)
  - **Training** set of documents, each labeled with its class

    In a small island off the American coast, the Whateleys live in an old mill where a mysterious bloody being … Horror

    A train robbery complexed with many different individuals from various levels of society who work together … Western

    Two hearing protection product sales reps have mixed fortunes in the exercise of their trades. They first have to … Comedy

  - **Prediction**

    Professor Ayres discovers a secret in an ancient stone and when he opens a crypt, he revives … **which class?**

Classification 3/5

- Example 2 (artificial documents)
  - Training set of documents, each labeled with its class

aba	A
baabaaa	A
bbaabbab	B
abbaa	A
abbb	B
bbbaab	B

Note: just two words
(a and b, spaces omitted)
and two classes

  - Prediction

abababa	which class?
baaaaaa	which class?

Classification 4/5

- Difference to K-means
  - K-means can also be seen as assigning (predicting) a class label for each object … each cluster = one class
  - **Difference 1:** the clusters have no "names"
  - **Difference 2:** k-means has no learning phase (where it could learn how objects and classes relate)

    This is called **unsupervised** learning … in contrast, a method like Naive Bayes does **supervised** learning
  - **Difference 3:** classification methods do soft clustering = for each object, output a probability for each class

    But one often wants only the most probable class

Classification 5/5

- Quality evaluation
  - Given a **test set** of labeled documents, and the predictions from a classification algorithm
  - For each class c let:

    D_c = documents labeled c (in the test set)

    D'_c = documents classified as c (by the algorithm)
  - Then (note that these are per class)

    Precision $P := |D'_c \cap D_c| / |D'_c|$
    Recall $R := |D'_c \cap D_c| / |D_c|$
    F-measure $F := 2 \cdot P \cdot R / (P + R)$

    Note that $P = R = F = 100\%$ if and only if $D_c = D'_c$

Probability recap 1/3

- Motivation
  - In this lecture, we will look at Naive Bayes, one of the simplest (and most widely used) classification algorithms
  - Naive Bayes makes **probabilistic assumptions**
  - For that, two very basic concepts from probability theory need to be understood:

    Maximum Likelihood Estimation (MLE)

    Conditional probabilities and Bayes Theorem
  - The following two slides are to refresh your memory concerning both of these

Probability recap 2/3

because ln is a monotone function

- Maximum Likelihood Estimation (MLE)
  - Consider a sequence of coin flips, for example

    $x \; x \; 1-x \; 1-x \; \dots$

    HHTTTTTTHTTTTTHTTHTT (5 times H, 15 times T)
  - Which Pr(H) and Pr(T) are the most likely?
  - Looks like Pr(H) = ¼ and Pr(T) = ¾ ... let's prove this

$x := Pr(H) \Rightarrow Pr(T) = 1 - x$

$Pr(HHTT \dots HTT) = x^5 \cdot (1-x)^{15}$

Find x which maximizes this

That's the MLE principle

Equivalently, maximize $g(x) = \ln\left[x^5 \cdot (1-x)^{15}\right]$

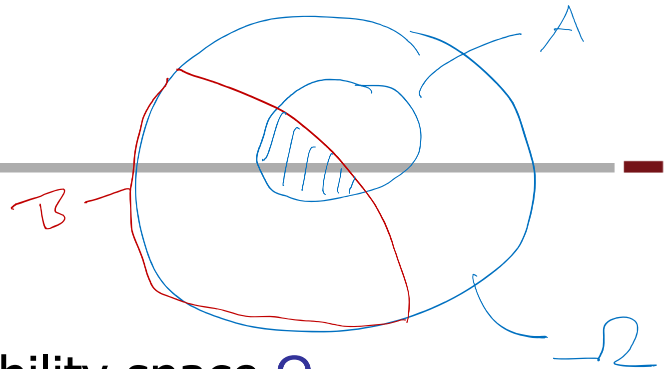
$= 5 \cdot \ln x + 15 \cdot \ln(1-x)$

$g'(x) = \frac{5}{x} - \frac{15}{1-x} = 0 \iff 5(1-x) = 15x \iff 5 = 20x \iff x = \frac{1}{4}$

$g''(x) = -\frac{5}{x^2} + \frac{15}{(1-x)^2}$ ... check that $g''(\frac{1}{4}) < 0$

$\Rightarrow$ MAX

Probability recap 3/3



- Conditional probabilities
  - Let A and B be events in a probability space Ω
  - Denote by Pr(A | B) the probability of A ∩ B in the space B

(1) $\Pr(A \mid B) := \Pr(A \cap B) \,/\, \Pr(B)$

(2) $\Pr(A \mid B) \cdot \Pr(B) = \Pr(B \mid A) \cdot \Pr(A)$

  - The latter is called **Bayes Theorem**, after Thomas Bayes, 1701 – 1760

  - For an intuitive understanding, assume that Ω is finite, and all x in Ω equiprobable:

Rolling a die $\Rightarrow \Omega = \{1, 2, 3, 4, 5, 6\}$

Event $A = \{2, 4, 6\}$ „rolling an even number"

Event $B = \{1, 2, 3\}$ „rolling a number ≤ 3"

$$\Pr(A \mid B) = \frac{|A \cap B|}{|B|} = \frac{|A \cap B| \,/\, |\Omega|}{|B| \,/\, |\Omega|} = \frac{\Pr(A \cap B)}{\Pr(B)}$$

$$\Pr(A) = \frac{|A|}{|\Omega|} = \frac{3}{6} = \frac{1}{2}$$

$$\Pr(B) = \frac{|B|}{|\Omega|} = \frac{3}{6} = \frac{1}{2}$$

Naive Bayes 1/11

- Probabilistic assumptions
  - Underlying probability distributions:

    A distribution p_c over the classes … where $\Sigma_c\ p_c = 1$

    For each c, a distr. p_{wc} over the words … where $\Sigma_w\ p_{wc} = 1$
  - Naïve Bayes assumes the following process for generating a document D with m words $W_1 \ldots W_m$ and class label C:

    Pick $C=c$ with prob. p_c , then pick each word $W_i=w$ with probability p_{wc} , independent of the other words
  - This is clearly unrealistic (hence the name **Naive** Bayes): e.g. when "relativity" is present, "theory" is more likely, and word order carries important information, too
  - Anyway, this gives us something well-defined

Naive Bayes 2/11

- Learning phase
  - For a **training set** T of objects, let:

    T_c = the set of documents from class c

    n_{wc} = #occurrences of word w in documents from T_c

    n_c = #occurrences of all words in documents from T_c
  - We compute the following probabilities or likelihoods:

    $\mathbf{p_c \coloneqq |T_c| \ / \ |T|}$ global likelihood of a class

    $\mathbf{p_{wc} \coloneqq n_{wc} \ / \ n_c}$ likelihood of a word for a class
  - The rationale behind these formulas is MLE (see slide 13)
  - Beware: n_{wc} and hence p_{wc} are often **zero** … see slide 21

Naive Bayes 3/11

- Learning phase, example
  - Consider Example 2 (artificial documents)

aba	A
baabaaa	A
bbaabbab	B
abbaa	A
abbb	B
bbbaab	B

$|T_A| = 3$, $|T_B| = 3$, $|T| = 6$

$n_{aA} = 10$, $n_{bA} = 5$, $\Rightarrow n_A = 15$

$n_{aB} = 6$, $n_{bB} = 12$ $\Rightarrow n_B = 18$

The probabilities we learn:

$P_A = \frac{|T_A|}{|T|} = \frac{1}{2}$, $P_B = \frac{|T_B|}{|T|} = \frac{1}{2}$

$P_{aA} = \frac{10}{15} = \frac{2}{3}$, $P_{bA} = \frac{1}{3}$

$P_{aB} = \frac{6}{18} = \frac{1}{3}$, $P_{bB} = \frac{2}{3}$

Naive Bayes 4/11

$Pr(A | B) = Pr(B | A) \cdot Pr(A) / Pr(B)$

- Prediction
  - For a given document **d** we want to compute the probability of each class **c**, given document d:

    $\mathbf{Pr(C=c \mid D=d)}$

  - Using Bayes Theorem, we have:

    $Pr(C=c \mid D=d) = Pr(D=d \mid C=c) \cdot Pr(C=c) / Pr(D=d)$

  - Using our (naive) probabilistic assumptions, we have:

$$Pr(D=d \mid C=c) = Pr(W_1=w_1 \cap \ldots \cap W_m=w_m \mid C=c)$$
$$= \Pi_{i=1,\ldots,m} Pr(W_i=w_i \mid C=c)$$

Naive Bayes 5/11

- Prediction ... continued
  - We thus obtain that

    $\Pr(C=c \mid D=d)$

    $= \Pi_{i=1,\dots,m} \Pr(W_i=w_i \mid C=c) \cdot \Pr(C=c) / \Pr(D=d)$

    $= \boldsymbol{\Pi_{i=1,\dots,m}\, p_{w_i c} \cdot p_c} / \Pr(D=d)$
  - Note 1: for $\Pi_{i=1,\dots,m}\, p_{w_i c}$ just take the p_{wc} for all words w in the document and multiply them (if a word w occurs multiple times, also take the factor p_{wc} multiple times)
  - Note 2: the $\Pr(D=d)$ is the same for all classes c

    We can hence compute the class c with the largest $\Pr(C=c \mid D=d)$ entirely from the learned p_{wc} and p_c

Naive Bayes 6/11

- Prediction, example
  - Consider Example 2 (artificial documents)

aba	A
baabaaa	A
bbaabbab	B
abbaa	A
abbb	B
bbbaab	B

$p_A = p_B = \frac{1}{2}$

$p_{aA} = \frac{2}{3}, \; p_{bA} = \frac{1}{3}$

$p_{aB} = \frac{1}{3}, \; p_{bB} = \frac{2}{3}$

  - Let us predict the class for aab … A or B ?

$\Pr(A \mid aab) = p_{aA} \cdot p_{aA} \cdot p_{bA} \cdot p_A / \Pr(aab) = \frac{2}{3} \cdot \frac{2}{3} \cdot \frac{1}{3} \cdot \frac{1}{2} / \Pr(aab)$

$\Pr(B \mid aab) = p_{aB} \cdot p_{aB} \cdot p_{bB} \cdot p_B / \Pr(aab) = \frac{1}{3} \cdot \frac{1}{3} \cdot \frac{2}{3} \cdot \frac{1}{2} / \Pr(aab)$

$\Rightarrow \Pr(A \mid aab) > \Pr(B \mid aab)$ $\Rightarrow$ predict A

Naive Bayes 7/11

- Smoothing
  - Problem: $\Pr(C=c \mid D=d) = 0$ if only one single $p_{wc} = 0$

    This happens rather easily, namely when d contains a word that did not occur in the training set for class c
  - Therefore, during training we actually compute

    $p_{wc} := (n_{wc} + \boldsymbol{\varepsilon}) / (n_c + \boldsymbol{\varepsilon} \cdot \textbf{\#vocabulary})$

    This is like adding every word ε times for every class

    Note that the change in the denominator is necessary, so that the p_{wc} still sum to 1 for all w and a fixed class c

    For ES11, take ε = 1/10 … a larger ε would add too much noise for the short documents from our datasets

Naive Bayes 8/11

- Smoothing ... continued
  - So $p_{wc} = 0$ is a problem, what about $\mathbf{p_c = 0}$ for a class c?
  - When $p_c = 0$, then Pr(C=c | D=d) = 0 for **any** document d
  - When $p_c = 0$, therefore class c would never be predicted
  - When does $p_c = 0$ happen?

    It happens if and only if $|T_c| = 0$, that is, when there is no document from class c in the training set
  - But if we did not see any document from a particular class c during training, we can learn nothing for that class, and we cannot meaningfully predict it
  - So Pr(C=c | D=d) = 0 is actually meaningful in that case and we do not need any additional smoothing to deal with that

Naive Bayes 9/11

$\ln \prod p_i = \sum_i \ln p_i$

- Numerical stability
  - Problem: a product of many small probabilities quickly becomes zero due to limited precision on the computer

    For example, the smallest positive number that can be represented with an 8-byte double is ≈ 10^{-308} ≈ 2^{-1024}

    Thus multiplying 52 probabilities < 10^{-6} is already zero
  - Therefore, compute the **log**-probabilities … then products of probabilities translate into sums of log-probabilities

    Log-probabilities also give you the most likely class, because log is a monotone function

    Don't take exp in the end! With a double, exp(-1000) = 0

Naive Bayes 10/11

- Some possible refinements
  - Instead of words, we could take any other quantifiable aspect of a document as so-called "feature"

    For example, also consider all (two-word) phrases
  - Don't use non-predictive words like "and" as features

    For example, omit the most frequent words or omit all words from a predefined list of so-called "stopwords"
  - In training, replace the word frequencies n_{wc} by $tf.idf_{wc}$

    And correspondingly, replace n_c by Σ_w $tf.idf_{wc}$
  - For ES11, these are all optional

    However, you must use stopwords.txt to filter which words to show in the end for each class

Naive Bayes 11/11

- Linear algebra (LA)
  - Assume the documents are given as a term-document matrix, like we have seen it many times now

    For ES11, we provide you with the code to construct the document-term matrix with simple tf entries

  - Then all the necessary computations can again be done very elegantly and efficiently using matrix operations

    Whenever you have to compute a large number of (weighted) sums in a uniform manner, this calls for LA

    However, if you feel more comfortable with (boring and inefficient) for-loops, you can use those for ES11 too

References

- Further reading
  - Textbook Chapter 13: Text classification & Naive Bayes

    http://nlp.stanford.edu/IR-book/pdf/13bayes.pdf
  - Advanced material on the whole subject of learning

    Elements of Statistical Learning, Springer 2009
- Wikipedia
  - http://en.wikipedia.org/wiki/Naive_Bayes_classifier
  - http://en.wikipedia.org/wiki/Bayes'_theorem
  - http://en.wikipedia.org/wiki/Maximum_likelihood