Information Retrieval

WS 2015 / 2016

Lecture 10, Tuesday January 12th, 2016
(Classification, Naive Bayes)

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

■ Organizational
  – Your results + experiences with ES9 k-means
  – Date and place for the exam

■ Contents
  – Classification introduction and examples
  – Probability recap two one-slide crash courses
  – Naïve Bayes algorithm, example, implementation
  – Exercise Sheet 10: learn to predict the genre and rating from a given movie description using Naïve Bayes
Your experiences with ES9

- Summary / excerpts
  - Quite hard and time-consuming for many of you
  - Bug in sparse normalization we provided
    - Was pointed out on the forum and then fixed soon
  - Using numpy / scipy in the right way was non-trivial
  - Easy to make small mistakes which are hard to debug
  - Still lots of problems getting the linear algebra right
  - For the geometric toy example from the lecture, normalizing the points gives different clusters ... was explained in forum
  - Great support on the forum, at all times
Your results for ES9

- For our dataset (≈ 200,000 docs, 50 clusters)
  - Relatively few iterations (20 - 30) are enough
  - Pretty fast, even with Python (≈ 1 second / iteration)

  That is the power of linear algebra: two years ago, even the C++ implementations were 10 x slower on a smaller dataset

  - Some centroids are meaningful, others not so much:
    written silent british comedy American starring frank
    it was at won nominated awards academy award best
    s an the her in to of story about who
    jung jin park korea soo ki lee kim south korean
Exam

- **Written exam**
  - For all **except** the B.Sc. Computer Science students
  - Date: **Tuesday, February 23, 2016**  14 – 16 h
  - Place: HS026 ... and maybe also HS036
    - Depends on the number of participants

- **Oral exam**
  - **Only** for the B.Sc. Computer Science students
  - Date: **Wednesday, February 24, 2016, afternoon**
    - A time slot will be allocated to you by the Prüfungsamt
  - Place: **my office** (building 51, second floor, room 28)
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

Flying Saucer Rock n Roll from 1998 is a 12-minute spoof of a 1950s black and white science fiction B-movie ... Comedy

Tainted is a 1988 low-budget suspense drama about a school teacher married to the owner of a crematorium ... Thriller

Toby the pup in the museum is the first cartoon in a series of twelve. Toby works as a janitor in a museum ... Animation

- Prediction

Heavy Times one summer afternoon, out of boredom and peer pressure, three best friends go to visit ... which class?
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
  
  Just two words (a and b, spaces omitted), and two classes

- Prediction

  abababa         which class?
  baaaaaa         which class?
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 Naïve 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
Quality evaluation

- Given a test set of labeled documents, and the predictions from a classification algorithm

- For each class $c$ let:
  
  $D_c = \text{documents labeled } c \quad (\text{in the test set})$
  
  $D'_c = \text{documents classified as } c \quad (\text{by the algorithm})$

- Then (note that these are per class)

  **Precision** $P := \frac{|D'_c \cap D_c|}{|D'_c|}$

  **Recall** $R := \frac{|D'_c \cap D_c|}{|D_c|}$

  **F-measure** $F := \frac{2 \cdot P \cdot R}{P + R}$

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 Naïve Bayes, one of the simplest (and most widely used) classification algorithms
- Naïve 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
Maximum Likelihood Estimation (MLE)

- Consider a sequence of coin flips, for example
  \[\text{HHTTTTTTTTTTHTTTHTTT}\] (5 times H, 15 times T)
- Which \(\Pr(H)\) and \(\Pr(T)\) are the most likely?
- Looks like \(\Pr(H) = \frac{1}{4}\) and \(\Pr(T) = \frac{3}{4}\) ... let's prove this

\[
x := \Pr(H), \quad \text{then} \quad \Pr(T) = 1 - x
\]

\[
\Pr(\text{HHTTTTTTTTTTHTTTHTTT}) = x^5 \cdot (1-x)^{15}
\]

Find \(x\) such that \(x^5 \cdot (1-x)^{15}\) is maximized

Equivalently, find \(x\) such that \(g(x) := 5 \ln(x) + 15 \ln(1-x)\) is maximized

\[
g'(x) = \frac{5}{x} - \frac{15}{1-x} = 0 \quad \Rightarrow \quad 5(1-x) = 15 \cdot x
\]

\[
5 = (15+15) \cdot x \quad \Rightarrow \quad \Pr(H) = x = \frac{5}{5+15} = \frac{1}{4}
\]

\[
\Rightarrow \Pr(T) = \frac{15}{5+15} = \frac{3}{4}
\]
Conditional probabilities

- Let $A$ and $B$ be events in a probability space $\Omega$
- Denote by $\Pr(A \mid B)$ the probability of $A \cap B$ 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 $\Omega$ is finite, and all $x$ in $\Omega$ equiprobable:

  \[
  \Pr(A \mid B) = \frac{|A \cap B|}{|B|} = \frac{|A \cap B \setminus B|}{|B \setminus B|} = \frac{\Pr(A \cap B)}{\Pr(B)}
  \]

  \[
  \Pr(B \mid A) = \frac{|A \cap B|}{|A|} = \frac{|A \cap B \setminus A|}{|A \setminus A|} = \frac{\Pr(A \cap B)}{\Pr(A)}
  \]
Probabilistic assumption

- Underlying probability distributions:
  
  A distribution \( p_c \) over the classes ... where \( \sum_c p_c = 1 \)
  
  For each \( c \), a distr. \( p_{wc} \) over the words ... where \( \sum_w p_{wc} = 1 \)
  
- Naïve Bayes assumes the following process for generating a document \( D \) with \( m \) words \( W_1...W_m \) 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

- Anyway, this gives us something well-defined
Learning phase

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

  $T_c = \text{the set of documents from class } c$

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

  $n_c = \#\text{occurrences of all words in documents from } T_c$

- We compute the $p_c$ and $p_{wc}$ using simple maximum likelihood estimation (MLE), as explained on Slide 10

  $p_c := |T_c| / |T| \quad \text{global likeliness of a class}$

  $p_{wc} := n_{wc} / n_c \quad \text{likeliness of a word for a class}$

  Beware: $n_{wc}$ and hence $p_{wc}$ are often zero ... see slide 20
Learning phase, example

Consider Example 2 (artificial documents)

aba \ A
baabaaa \ A
bbaabbab \ B
abbaa \ A
abbb \ B
bbbaaab \ B

| \( |T_A| = 3\) | \( |T_B| = 3\) | \( |T| = 6\) | \( P_A = P_B = \frac{3}{6} = \frac{1}{2} \)
| \( M_{aA} = 10\) | \( M_{bA} = 5\) | \( M_A = 15\) | \( P_{aA} = \frac{2}{3} \) | \( P_{bA} = \frac{1}{3} \)
| \( M_{aB} = 6\) | \( M_{bB} = 12\) | \( M_B = 18\) | \( P_{aB} = \frac{1}{3} \) | \( P_{bB} = \frac{2}{3} \)
Prediction

- For a given document $d$ we want to compute
  \[
  \Pr(C=c \mid D=d) \quad \text{... for each class } c
  \]
  The probability of class $c$, given document $d$

- Using Bayes Theorem, we have:
  \[
  \Pr(C=c \mid D=d) = \frac{\Pr(D=d \mid C=c) \cdot \Pr(C=c)}{\Pr(D=d)}
  \]

- Using our (naïve) probabilistic assumptions, we have:
  \[
  \Pr(D=d \mid C=c) = \Pr(W_1=w_1 \cap \ldots \cap W_m=w_m \mid C=c)
  \]
  \[\overset{\text{"naïve assumption"}}{=} \prod_{i=1,\ldots,m} \Pr(W_i=w_i \mid C=c)\]
Prediction ... continued

- We thus obtain that $\Pr(C=c \mid D=d)$

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

  $= \prod_{i=1,...,m} p_{w_i|c} \cdot p_c / \Pr(D=d)$

  For the product in the front 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 that the $\Pr(D=d)$ is the same for all $c$

  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)
    
    \[
    \begin{array}{l|c}
    \text{Document} & \text{Class} \\
    \hline
    \text{aba} & A \\
    \text{baabaaa} & A \\
    \text{bbaabbab} & B \\
    \text{abbaa} & A \\
    \text{abbb} & B \\
    \text{bbbaab} & B \\
    \end{array}
    \]

  - Let us predict the class for aab ... A or B?
    
    \[
    \begin{align*}
    q_A & = \Pr(C = A | D = \text{aab}) = \frac{P_a A \cdot P_a A \cdot P_b A \cdot P_A}{Pr(D = \text{aab})} = \frac{2/3 \cdot 2/3 \cdot 1/2 \cdot 1/2}{Pr(D = \text{aab})} \\
    q_B & = \Pr(C = B | D = \text{aab}) = \frac{P_B B \cdot P_B B \cdot P_B B \cdot P_B}{Pr(D = \text{aab})} = \frac{4/3 \cdot 4/3 \cdot 2/3 \cdot 1/2}{Pr(D = \text{aab})}
    \end{align*}
    \]

  - Recall from the training:
    \[
    P_A = P_B = \frac{1}{2}, \quad P_a A = \frac{2}{3}, \quad P_B B = \frac{2}{3}
    \]

  - Another example:
    \[
    \text{abababa} \rightarrow ?
    \]
Smoothing

- Problem: when only one $p_{wc} = 0$, then $Pr(C=c \mid D=d) = 0$

  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

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

  This is like adding every word $\varepsilon$ times for every class

  For ES10, take $\varepsilon = 1/10$ ... for short docs, a larger $\varepsilon$ would add too much noise
Smoothing ... continued

- What about $p_c = 0$ for a class $c$?
  
  This means, that $|T_c| = 0$, that is, there was no document from class $c$ in the training set.

- When $p_c = 0$, then $\Pr(C=c \mid D=d) = 0$ for any document $d$.
  
  But that is reasonable: 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 no smoothing needed for that case.
Naive Bayes 9/11

- 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 \( \approx 10^{-308} \).

Then multiplying 52 probabilities \(< 10^{-6}\) is already zero.

- Therefore, compute the \textbf{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.

Beware: don't take exp in the end! \( \exp(-1000) = 0 \) (with a double).
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

- Omit non-predictive words like "and"

  For example, omit the most frequent words

- In training, replace the word frequencies $n_{wc}$ by $tf.idf_{wc}$

  And correspondingly, replace $n_c$ by $\sum_w tf.idf_{wc}$

- For ES10, none of these are required ... but feel free to play around with them
Linear algebra (LA)

- Assume the documents are given as a term-document matrix, like we have seen it many times now.

  For ES10, 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 ES10 too.
References

- Further reading
  - Textbook Chapter 13: Text classification & Naive Bayes
  - Advanced material on the whole subject of learning
    Elements of Statistical Learning, Springer 2009

- Wikipedia