Quantitative and qualitative computational analysis of language and text similarities, clustering and classification

Damir Ćavar

CLS 2010, August 2010

University of Zadar

Agenda

- Dimensionality Reduction
- Naive Bayesian classifier

Dimensionality Reduction

- List of tokens
  - *Peter reads a book . John and Mary read some newspaper .*
  - Two tokens with the same meaning, type etc.: *reads*, *read*
  - Normalization via lemmatization
    - *reads* → *read*

Dimensionality Reduction

- Removing stop-words
- Stemming
- Lemmatization
- Thesaurus-based mapping to hypernyms (e.g. WordNet)
- ...

Dimensionality Reduction

- Term-based models become large:
  - N-gram models
  - Vector matrix
- Goal:
  - Reduce model size with maximized performance.

Dimensionality Reduction

- Document frequency thresholding
- X2
- Mutual Information
- Information Gain

Document Frequency

- DF: Number of documents in which a term *x* occurs
  - Threshold
  - Calculate DF for each term in DC_{tr}
  - Assumption: rare terms not informative for category prediction
    - contrary to Zipf

Document Frequency

- DF: Number of documents in which a term x occurs
  - low-DF terms are rather informative

Dimension Reduction

- Identify terms with no contribution to a class or all classes.
  - Selection of significant terms
  - Elimination of noise

chi^2 (χ2) Test

- Literature:
  - Manning, Raghavan & Schütze (2009)

chi^2 ($\chi 2$) Test

- Observation:
  - Number of documents: $|DC| = 801948$
  - Documents labeled as c_i or not, containing t_j or not:

	c_i	$\neg c_i$	**total**
t_j	49	27652	27701
$\neg t_j$	141	774106	774247
total	190	801758	**801948**

chi² (χ2) Test

- **Research hypothesis:** The term and the class label are dependent variables
  - $P(t_j c_i) \neq P(t_j)P(c_i)$
- **Null hypothesis:** The term and the class label are independent variables
  - $P(t_j c_i) = P(t_j)P(c_i)$

chi^2 ($\chi 2$) Test

- Observation, Expectation (Null Hypothesis)
  - Expectation: P(c) * P(t) = Row-total * Column-total / Total

	c_i	$\neg c_i$	**total**
t_j	49 6.56	27652 27694.44	27701
$\neg t_j$	141 183.44	774106 774063.56	774247
total	190	801758	**801948**

chi² (χ2) Test

$$\chi^2 = \sum \frac{(observation - expectation)^2}{expectation}$$

$$\chi^2 = \frac{(49 - 6.56)^2}{6.56} + \frac{(27652 - 27694.44)^2}{27694.44} + \frac{(141 - 183.44)^2}{183.44} + \frac{(774106 - 774063.56)^2}{774063.56} = 284.45$$

chi^2 (χ2) Test

- Degree of freedom: (rows - 1) * (columns - 1) = 1

p	χ2 critical value
0.1	2.71
0.05	3.84
0.01	6.63
0.005	7.88
0.001	10.83

- See online table

chi^2 (χ2) Test

- For $P(\chi^2 > 6.63) < 0.01$, i.e. the Null Hypothesis (independence assumption) can be rejected with 99% confidence.
  - The class label and token seem to be dependent.

chi^2 (χ2) Test

- Dimension reduction:
  - Apply the χ^2 test to all tokens for all classes and eliminate tokens that appear to be independent of the class label.

chi^2 (χ2) Test

- Problems
  - Iterative use of the χ^2 test increases the error.
    - 1000 rejections with 0.05 error probability lead to an average of 50 wrong decisions.
  - Here: The test is meant to be for "relative" importance of features.

Mutual Information

- For a term *t* and category *c*:

$$I(t,c) = log \frac{P(tc)}{P(t)P(c)}$$

- How much information does *t* provide about *c*?
  - Compare to χ^2: log ratio of Research and Null hypothesis, or observation and expectation.

Mutual Information

- For a term *t* and category *c* (Yang & Pederson 1997):

	c_i	$\neg c_i$	total
t_j	**A** 49	**B** 27652	27701
$\neg t_j$	**C** 141	**D** 774106	774247
total	190	801758	**N** **801948**

$$I(t,c) \approx \frac{A \times N}{(A+C) \times (A+B)}$$

Mutual Information

- For a term *t* and category *c* (Manning ea. 2009):

	c_i	$\neg c_i$	**total**
t_j	**A** 49	**B** 27652	**N_1** 27701
$\neg t_j$	**C** 141	**D** 774106	774247
total	**N_2** 190	801758	**N** **801948**

$$I(U;C) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} P(U=e_t, C=e_c) log_2 \frac{P(U=e_t, C=e_c)}{P(U=e_t)P(C=e_c)}$$

Mutual Information

$$I(U;C) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} P(U = e_t, C = e_c) log_2 \frac{P(U = e_t, C = e_c)}{P(U = e_t)P(C = e_c)}$$

- Example:
  - P(1,1) = 49 / 801948
  - P(U=1)=27701 / 801948
  - P(C=1)=190 / 801948

Mutual Information

- Bias for terms with low frequencies
  - Score not comparable between terms with varying frequency
- Equivalent to *Information Gain* (Manning et al. 2009)

Algorithms 1

Naive Bayes TC

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

- document *d*
- class *c*
- conditional probability of term t_k occurring in a document class *c*: $P(t_k|c)$

Naive Bayes TC

- After tokenization and stop-word removal:
- "Germany won the world championship."
  - {Germany, won, world, championship}
  - $n_d = 4$

Naive Bayes TC

- Find best class: *maximum a posteriori* (MAP) class c_{map}:

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

- P-cap is the estimated probability using a training corpus.

Naive Bayes TC

- High number of float multiplications can result in a floating point underflow.
  - We add the logs of the probabilities, maintaining the relative order:
    - highest is most probable (log is monotonic)

Naive Bayes TC

- Sums of logs:

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

Naive Bayes TC

- Estimation:
  - N_c is the number of documents in class C in the training corpus

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

  - T_{ct} is the frequency of token t in the documents in c, ($T_{ct'}$ all t)

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

Naive Bayes TC

- To avoid 0 probabilities:
  - Smoothing: e.g. *add-one*

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

Naive Bayes TC

```
TrainNB(C, D)
   V <- ExtractVocabulary(D)
   N <- CountDocs(D)
   for each c in C
   do N_c <- CountDocsInClass(D, c)
      prior[c] <- Nc/N
      text_c <- ConcatenateTextOfAllDocsInClass(D, c)
      for each t in V
      do Tct <- CountTokensOfTerm(text_c, t)
      for each t in V
      do condprob[t][c] → (T_ct + 1)/(sum (T_ct' + 1))
return V, prior, condprob

ApplyNB(C, V, prior, condprob, d)
   W → ExtractTokensFromDoc(V, d)
   for each c in C
   do score[c] → log prior[c]
      for each t in W
      do score[c] += log condprob[t][c]
   return arg max_c_in_C score[c]
```

Naive Bayes TC

- Example:
  - model generator: make-docmodel.py
  - classifier: BM1.py
    - command line: python BM1.py my.txt

Manipulations

- Weighting of terms
- Dimension reduction
  - Elimination of stop-words
  - MI, Chi^2, frequency-based, etc.