Human Language Technology: Applications to Information Access

EPFL Doctoral Course EE-724, Fall 2016

Andrei Popescu-Belis

Idiap Research Institute, Martigny

Lesson 1b: Text classification

September 22, 2016

Objectives

- Introduce the problem of text classification
- Experiment with classifiers from the WEKA environment and the Reuters data set
- Put into place the *experimental procedure* which will be used all along the HLT course
  - problem/data modeling | feature extractors | finding the best classifier | evaluating the result | analysis

Importance of analyzing results

- Running feature extractors and classifiers is not too difficult, but has little scientific value
- Where is the science?
  - understanding why the system works or not
  - what features are useful and in which conditions
  - what feature representation works best and why
  - what classifiers work best and why
- Need to report the analysis so that it is useful to other researchers, in other situations (task, data)

Text classification

- **Problem**: given a new, unseen text, assign it to one class among a list of given classes (i.e. give it a *label*)
  - two classes: binary classification
    - email message → spam | not-spam
  - multiple classes
    - news article → { politics, sports, finance, … }
  - sentiment analysis, a specific form of classification
    - movie review → { positive, negative, neutral }

- **Solution**: train a classifier using texts that have already been correctly classified by humans (supervised ML)
  - related to, but different from *clustering* (unsupervised): given a set of texts, group them into classes by similarity

Methods for text classification

- Does it require “understanding”?
  - in theory, yes, for complex cases (e.g. movie about Swissair bankruptcy → ‘entertainment’? ‘Switzerland’? ‘economy’?)
  - in practice, superficial methods work quite well

- Methods
  - “with understanding”
    - build semantic representations of texts and labels
    - match representations and choose the closest matching label
  - “without understanding”
    - derive word-based features from texts
    - learn relationship between features and labels from training data
    - apply the classifier to any new data that must be classified

Word-based text classification: how does it work?

- Principle
  - extract all words from a text (except 'stopwords' such as *and*, *the*, *of*, *it*, *is*, and many other function words)
  - learn some "rules" that relate the presence or absence of each word to a given class, by observing the training data

- Issues to solve
  - decide how to represent presence/absence of a word, possibly the number of occurrences of specific words
    ➔ feature representation
  - decide what type of "rules" to build
    ➔ classifier: decision tree, Naïve Bayes, SVM, etc.

Naïve Bayes text classification

- Naïve = assumes independence of variables
- Bayes = uses Bayes' theorem to "learn"
  - express probabilities using a "learnable" function

$$P(class|features) = P(features|class) \cdot P(class) \,/\, P(features)$$

- Given a document (*features*), choose the most probable class, i.e. the one that maximizes $P(features|class) \cdot P(class)$
  - these two terms can be computed thanks to what is learned during the training phase
  - maximum a posteriori (MAP) class

Two possible variants

- Alternative: are we interested in …
  1. whether a document contains or not a given word?
  2. how many occurrences of a given word it contains?
- Feature representations
  - if V is the vocabulary set, in a fixed order ($|V|=v$)
  1. *document* = $(e_1, e_2, \ldots, e_v)$ where $e_i \in \{0, 1\}$ indicates whether word$_i$ is present or not [“Bernoulli”]
  2. *document* = $(f_1, f_2, \ldots, f_v)$ where $f_i \in \mathbf{N}$ is the frequency (n. of occ.) of word$_i$ in the document [“multinomial”]

Derivation of Naïve Bayes models

- Assign a class to a new document using MAP

$$class_{MAP} = \arg\max P(class|features) =$$
$$= \arg\max (P(class) \cdot P(features|class))$$

- Where
  - p(class) can be estimated by the frequencies of the classes in the training data (maximum likelihood)
  - the other term depends on the variant of the model and the feature representation, Bernoulli vs. multinomial
  - both models make *strong independence assumptions!*

$P(features|class)$, with independence assumptions (wrong but efficient)

- Bernoulli model (presence vs. absence of word)

$$P((e_1, \ldots, e_v)|class) = \prod_{1 \le i \le v} P(\text{word}_i|class)$$

- Multinomial model (frequency of word)

$$P((f_1, \ldots, f_v)|class) = \prod_{1 \le i \le v} P'(\text{word}_i|\text{class})^{f_i}$$

- Estimate probabilities of the words given the class using frequency observations in the training data
  - Bernoulli: $P(\text{word}_i|class)$ = number of documents in *class* containing word_i over all documents in *class*
  - Multinomial: $P'(\text{word}_i|\text{class})$ = number of occurrences of word_i in the documents in *class* over all words in these documents
  - Smoothing: add-one ("Laplace") to numerators, and adjust denominators accordingly, to avoid zero *P* values

A public data set for text classification

- Reuters-21578 newswire articles
  - labeled with 118 possible "topic categories"
  - each document has 0, 1 or more labels (majority: 1)
  - "ModApté" split: 9603 training, 3299 testing

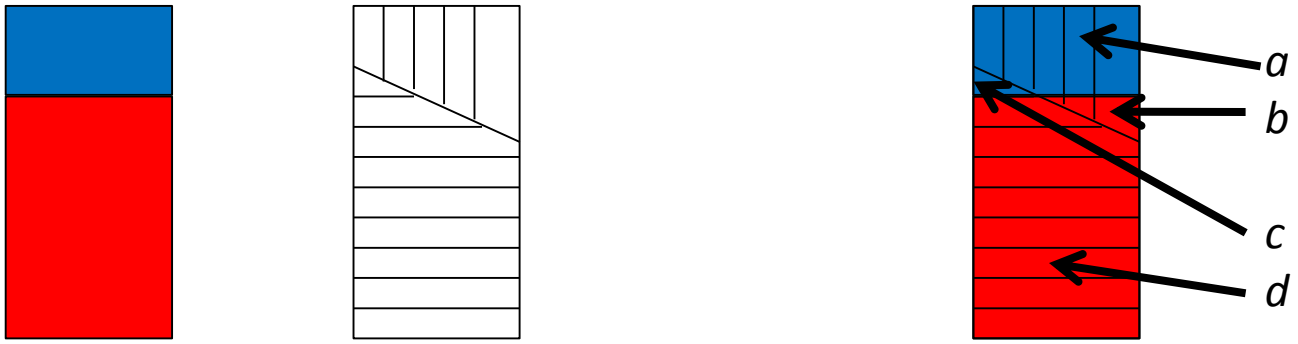
- How to model classification for this dataset?
  - either keep single-label documents and do multi-class classification
    - but then multi-label documents will never be correctly labeled
  - or define 118 binary classifiers, one for each topic
    - in practice, people often use only the 10 most frequent topics

Evaluation

- Simplest measure
  - accuracy = number of correctly labeled documents
  - not a good idea for unbalanced classes
    - because the majority answer gets high accuracy anyway
- Per class: precision and recall
  - precision = number of correct in-class documents / all in-class documents
  - recall = number of correct in-class docs / all docs assigned to class by the system
  - F1-score = harmonic mean of the two
- Micro-average (per document) vs. macro-average (per class)

Formulas for precision and recall

- Given a binary decision task
  - e.g. classify documents as positive or negative
- Goal: compare correct classification (red/blue) with the one hypothesized by a system (lines)



$P = a/(a+b)$ and $R = a/(a+c)$ and $F_1 = 2/(1/P+1/R)$

Instructions for the practical work (1/2)

- Use WEKA 3.8 as available online
  - download and install WEKA from http://www.cs.waikato.ac.nz/ml/weka/
  - open the WEKA 'Explorer' from the WEKA GUI
  - or to run it from a Command Window: `java -Xmx1000M -jar weka.jar`

- In the 'data' folder, there are four files with Reuters data
  - they include texts for two classes, 'grain' and 'corn', each with training/test data
  - examine the content of these files in a text editor
  - load one of the two training sets into Weka Explorer ('open')

- Use 'Preprocess' > 'Filter' > 'Choose' > 'Unsupervised' > 'Attribute' > 'StringToWordVector' to create word attributes from the text
  - explore the options of this filter (i.e. how to represent words as features)
  - if you need it, use the list of stop words from: http://www.idiap.ch/~apbelis/hlt-course/mallet_stoplist.txt or any other list that you find
  - pre-process the training file, save it with the new format and examine its content

Instructions for the practical work (2/2)

- Experiment with several classifiers (under ‘Classify’)
  - e.g. ZeroR (majority class), Naïve Bayes, SMO (SVM), JRip rules, J48 decision trees, nearest neighbors, etc.
  - don’t forget to indicate (menu button) that the output class to use is the first attribute, called ‘class-att’
  - train/test using 10-fold cross validation on the train file, examine the classification accuracy
  - what are the best classifiers and their scores?
- To run the trained classifier on the test file, use ‘Classifier’ > ‘Choose’ > ‘Meta’ > ‘FilteredClassifier’ indicating as options the filter & the classifier
  - then indicate the ‘supplied test set’
  - what are the best classifiers and their scores?
- If you have time: perform feature analysis (under ‘Select attributes’) and determine the most discriminative words or sets of words
  - what is the performance when using only these subsets?
- Write some notes (max 1 page), send them to apbelis@idiap.ch

Published evaluation results per class

(from Manning, Raghavan, & Schütze 2008, Table 13.9, page 261)

	NB	Rocchio	kNN	trees	SVM
earn	96	93	97	98	98
acq	88	65	92	90	94
money-fx	57	47	78	66	75
grain	79	68	82	85	95
crude	80	70	86	85	89
trade	64	65	77	73	76
interest	65	63	74	67	78
ship	85	49	79	74	86
wheat	70	69	77	93	92
corn	65	48	78	92	90
micro-avg (top 10)	82	65	82	88	92
micro-avg-D (118 classes)	75	62	n/a	n/a	87

References

- Manning, Raghavan, and Schütze, *Introduction to IR*, Chapter 13
  - available online: http://nlp.stanford.edu/IR-book/
  - see also their suggestions for further reading

- See also Manning and Schütze, *Foundations of Statistical NLP*, Chapter 16
  - for more details about text classification methods