Text Categorization

(M&S Ch 16)

NLP problems as classification

- Language identification
- Word sense disambiguation
- PP attachment
- POS tagging
- NP chunking
- Text categorization
  - by topic
  - by genre
  - by author

Machine learning algorithms (classification)

- Decision trees
- Naïve Bayes
- K nearest neighbour
- Perceptron
- Support Vector Machines
- Many others

Genre identification

- Genre: Type or style of a text.
  - Newspaper, novel, magazine, inter-office memo, love letter, legal document, scientific paper, …
  - Personal home page, FAQ, list of links, ...
- Depends on what the text is like, not where it comes from.
- Need to choose a set of features

Some different genres of text

MONTEVIDEO, Uruguay—A bus filled with people coming back from a political rally collided with another vehicle and plunged off a 50-foot cliff, killing 19 people and injuring 20, police said Sunday. The bus was taking supporters of the ruling Colorado Party home to Montevideo, the capital, after a meeting in the province of Florida with presidential candidate Jorge Batlle.

For a partnership, the employer is considered to be a corporation where all the shares with full voting rights are owned by the members in the same proportion as the member's share of the income or loss of the partnership. For an individual, the employer is considered to be a corporation where the individual owns all the shares with full voting rights of the corporation's capital stock.

Stylistic dimensions and their cues I

- Genre as cluster or combination of positions on stylistic dimensions.
  - Degree of formality, of impersonality, of involvement, of context-dependence, ...
- Position on dimension indicated by combination of cues in the text.
  - Frequent use of “you”, passive sentences, …

    Measure as frequency per thousand words of text (sometimes, per million)

Stylistic dimensions and their cues II

  - Frequent use of long words, short sentences, uncommon words, ratio of adjectives to nouns, ...

    Measured as average length over text, average frequency rank, ratio, ...

- Need comparison point: average value of cue across all kinds of texts.
- A two-stage process
  - Cues in text
  - Stylistic dimension
  - Genre

Genre - Position on stylistic dimensions

Official documents - Low involvement, low narrative, very high context-independence, high abstractness

General fiction - Moderate involvement, high narrative, moderate context- independence, fairly low abstractness

Personal letters - High involvement, moderate narrative, moderate context-independence, fairly low abstractness

Stylistic dimensions

- Involvement: Text directly addresses the reader.
- Narrative: Text describes a sequence of events that happened.
- Context-independence: Text doesn’t draw on external circumstances for meaning.
- Abstractness: Text concerns abstract ideas rather than specific instances.

Dimension - Cues in text

- Involvement - “you”, “that”-deletion, private verbs, low type–token ratio ...
- Narrative - past tense, third-person pronouns, public verbs ...
- Context-independence relative clauses, absence of time and place, adverbs, ...
- Abstractness - passives, conjunct words, past participial wh-is deletion …

Example of training data

File Name, First pers. pronoun, Second pers., Third person, Conjunctions, VBPresent, VBPast, VBPrivate, VBPublic, VBSaying, Nouns, Q_N, Adverbs, That, There, Split infinitiv, Hifalutin, Sentence length, Token lentgh, Type/Token Ratio, Class

A01, 3.05, 0.00, 21.34, 1.74, 33.97, 77.09, 5.23, 12.63, 12.63, 313.15, 0.44, 20.47, 12.20, 0.00, 0.00, 50.96, 21.46, 4.94, 0.38, I

A02, 1.71, 1.28, 18.80, 0.43, 34.62, 54.27, 4.70, 9.40, 7.26, 341.88, 0.85, 19.66, 4.70, 0.00, 0.00, 35.47, 20.35, 4.90, 0.40, II

A03, 0.85, 0.85, 20.90, 1.28, 27.72, 67.38, 3.84, 11.94, 8.96, 311.30, 0.43, 20.47, 5.12, 0.00, 0.00, 34.97, 19.87, 4.83, 0.36, III

A04, 0.00, 0.00, 22.68, 2.22, 41.80, 62.69, 8.89, 4.45, 3.11, 293.02, 0.89, 41.35, 14.67, 0.00, 0.00, 66.25, 25.85, 5.16, 0.40, I

……..

Other kinds of text classification

- By topic.
  - For news articles, business documents, incoming e-mail.
- For filtering of spam, threats, ...

Features for topic classification

- Bag of words.
- Topic classification is based on a count of word-types associated with each topic—typically several thousand.
- For each word-type i,

$$\text{scaledCount}(i) = \text{round}(10\ (1+\log tf_i)/(1+\log len))$$

where tf_i is the frequency of word-type i in the document and len is the length in tokens of the document.

Authorship attribution

- Authorship attribution or “fingerprinting”: Matching a sample of writing to an attested corpus of an author’s work.
- Try to find author’s unconscious signature.
- Can’t use stylistic dimensions.
- Need statistical tests for significance.

Some history

- First computer-based study: Mosteller and Wallace on The Federalist Papers (1964)
- Much controversy on statistical methods, choice of features, meaning of results, and everything else.
- Forensic applications.
- High-profile results by Donald Foster (Vassar College):
  - Attribution of A Funerall Elegye to Shakespeare (1989).
  - Matched Primary Colors to Joe Klein (1996).
  - Matching the Unabomber's manifesto to other writing by Theodore Kaczynski (1996).

Authors' signatures

- Average sentence length; type–token ratio.
- Proportional pairs.
- Word-length frequency distribution.
- Distinctiveness ratios, for words such as:

  according, *also*, *although*, *always*, *an*, *apt*, *both*, *by*, *commonly*, *consequently*, *considerable(ly)*, *direction*, *enough*, *innovation(s)*, *kind*, *language*, *matter(s)*, *of*, *on*, *particularly*, *probability*, *there*, *this*, *though*, *to*, *upon*, *vigor(ous)*, *while*, *whilst*, *work(s)*

Vector-space text classifiers (I)

- Text classifier or document classifier for genres, topics, languages, etc.
- Represent document as vector

$$\vec{x} = (x_1, x_2, ..., x_n)$$

of feature values or cues.

- A classifier maps a vector x to a class $c \in C$

where C is the set of possible document classes.

Vector-space text classifiers (II)

- Many methods for classifying or clustering objects in vector space.
- Parametric approaches; two main kinds:
  - For each class c, vector of weights w and threshold t such that:

$$class(\vec{x}) = \begin{cases} c, \text{if } \vec{x} \cdot \vec{w} \geq t \\ \neg c, \text{otherwise} \end{cases}$$

  - Decision trees - look at each component of the vector

Vector-space text classifiers (III)

- This is binary classification; need a classifier for each class.
- Sometimes want m-ary classification; sometimes want ranked list of possible class memberships.
  - Document could be members of several classes, or could have degree of membership.
- Some classification methods can be m-ary.
- Not always clear how to combine binary classifiers for m-ary classification.

Vector-space text classifiers (IV)

- To find weights or a decision tree, train a parametric classifier on a set of pre-classified documents.
- Need to decide what features will be used; refine as necessary.
- Can't expect 100% accuracy.
  - Limited number of features, imperfectly chosen.
  - Many classes are fuzzy; even humans disagree.
  - Document could be in several classes.

Decision trees

- Information-gain algorithms for building decision tree from training data (vectors with known classification).
  - Greedy algorithm builds tree top down.
  - At each node, determine the test that "best" splits the remaining data.
  - "Best" split is the one that adds the most information.
- Avoid overfitting by pruning the tree after it's built.
- ML tools: C4.5, C5.0, Weka.

Example of decision tree for genre classification

```
third pers <= 47.97 :
|   type/tok ratio <= 0.36 :
|   |   second pers <= 0 :
|   |   |   third pers <= 21.19 :
|   |   |   |   verbs saying <= 0 : II (3.0/1.0)
|   |   |   |   verbs saying > 0 : III (46.0/1.0)
|   |   |   third pers > 21.19 :
|   |   |   |   that <= 8.96 : II (11.0/1.0)
|   |   |   |   that > 8.96 :
|   |   |   |   |   split infinitives > 0.48 : I (2.0/1.0)
|   |   |   |   |   split infinitives <= 0.48 :
|   |   |   |   |   |   third pers > 32.18 : III (8.0)
|   |   |   |   |   |   third pers <= 32.18 :
|   |   |   |   |   |   |   that <= 13.32 : III (5.0/1.0)
|   |   |   |   |   |   |   that > 13.32 :
|   |   |   |   |   |   |   |   first pers <= 4 : I (2.0)
|   |   |   |   |   |   |   |   first pers > 4 : II (4.0)
```

The perceptron

- Learn weights w and threshold t for binary classification, if the training data is linearly separable by a hyperplane in the n-dimesional space

Perceptron learning algorithm

Intialize $\vec{w} = 0, t = 0$

Repeatedly go through the training set and classify each $\vec{x}_j$ with the current $\vec{w}$ and t.

If correct, do nothing.

If false negative, $t = t - 1$, $\vec{w} = \vec{w} + \vec{x}_j$

If false positive, $t = t + 1$, $\vec{w} = \vec{w} - \vec{x}_j$

Stop when all $\vec{x}_j$ are classified correctly.

k nearest neighbours (kNN)

- A lazy learning method for m-ary classification of documents that produces ranked lists.
  - Non-parametric.
  - No training session per se, but continues to learn while in use.
- Basic idea: A document x to be classified is probably in the same classes as the ones in the training set that are closest or most similar to it. Likelihood increases with closeness or similarity.
  - Each time a new document is classified, can add it to the training set.

kNN (memory-based learning)

- More generally: Let D be the k closest documents to x. For each document d in D and each class c of which d is a member, add sim(x,d) to the likelihood score that d is in c.
- Sort the likelihood scores to produce a ranked list.
- Or make a binary decision for each class c depending on whether or not its score exceeds a threshold t.
  - Find (learn) t that optimizes performance.

Setting some values

- We need a value for k and a function sim.
- sim could be cosine measure: the cosine of the angle between the vectors d and x (n-dimensional space).

Efficiency of kNN

- Simple implementation: Storage and search are both O(n p), where n is the dimensionality of the vector space and p is the size of the training set.
- Much research on more-efficient approximate methods.

Evaluating the results

- Results on training corpus might not be mirrored in the real world.
- Want to avoid overfitting.
- Need separate test data. Hold out 10–20% of the corpus.
  - N-fold cross-validation
  - Leave-one-out cross-validation
- Separate development and validation test sets.
- Need measure of performance and comparison to baseline.

Measures of performance

- If binary classification of M texts as members or not members of class c

Predicted Actual	c	not c
c	True Positive TP	False Negative FN
not c	False Positive FP	True Negative TN

Measures of performance

- Accuracy = TP + TN / (TP+FP+TN+FN)
- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)
- F-measure: trade-off between recall and precision:

$$F = \frac{2PR}{P+R} = \frac{2}{\frac{1}{R}+\frac{1}{P}}$$

- What about more than 2 classes?

Baseline performance

- Baseline: The minimum performance level that you're trying to improve on.
- Could be performance of competing system.
- Could be performance of dumb but easy method:
  - Random choice, most-frequent answer, very simple heuristic, ...
- Comparison should be made on the same test data for results to be fully meaningful.

More on Text Categorization

Naïve Bayes Classifiers

Text Categorization: attributes

- Representations of text are very high dimensional (one feature for each word).
- High-bias algorithms that prevent overfitting in high-dimensional space are best.
- For most text categorization tasks, there are many irrelevant and many relevant features.
- Methods that combine evidence from many or all features (e.g. Naïve Bayes, kNN, neural-nets) tend to work better than ones that try to isolate just a few relevant features (standard decision-tree or rule induction)

Bayesian Methods

- Learning and classification methods based on probability theory.
- Bayes theorem plays a critical role in probabilistic learning and classification.
- Build a *generative model* that approximates how data is produced
- Uses *prior* probability of each category given no information about an item.
- Categorization produces a *posterior* probability distribution over the possible categories given a description of an item.

Bayes' Rule

$$P(C, X) = P(C \mid X)P(X) = P(X \mid C)P(C)$$

$$\boxed{P(C \mid X) = \frac{P(X \mid C)P(C)}{P(X)}}$$

Naive Bayes Classifiers

Task: Classify a new instance based on a tuple of attribute values

$$\langle x_1, x_2, \ldots, x_n \rangle$$

$$c_{MAP} = \underset{c_j \in C}{\operatorname{argmax}} \; P(c_j \mid x_1, x_2, \ldots, x_n)$$

$$c_{MAP} = \underset{c_j \in C}{\operatorname{argmax}} \; \frac{P(x_1, x_2, \ldots, x_n \mid c_j) P(c_j)}{P(x_1, x_2, \ldots, x_n)}$$

$$c_{MAP} = \underset{c_j \in C}{\operatorname{argmax}} \; P(x_1, x_2, \ldots, x_n \mid c_j) P(c_j)$$

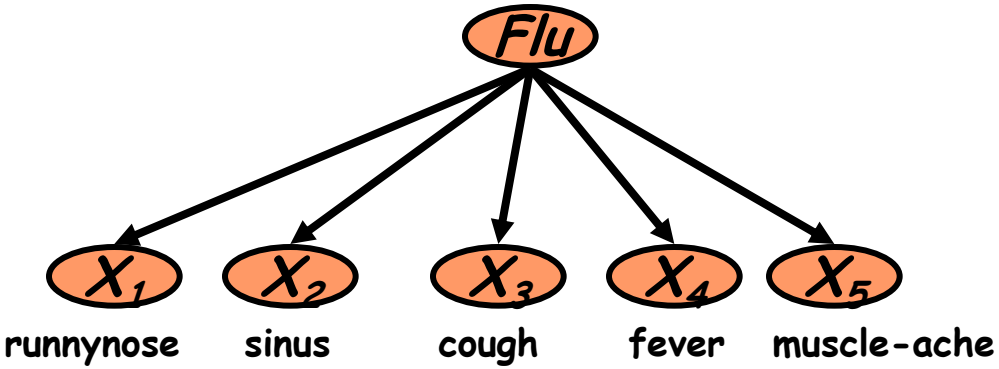
Naïve Bayes Classifier: Assumptions

- $P(c_j)$
  - Can be estimated from the frequency of classes in the training examples.
- $P(x_1,x_2,\ldots,x_n|c_j)$
  - $O(|X|^n \bullet |C|)$
  - Could only be estimated if a very, very large number of training examples was available.

Conditional Independence Assumption:

$\Rightarrow$ Assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities.

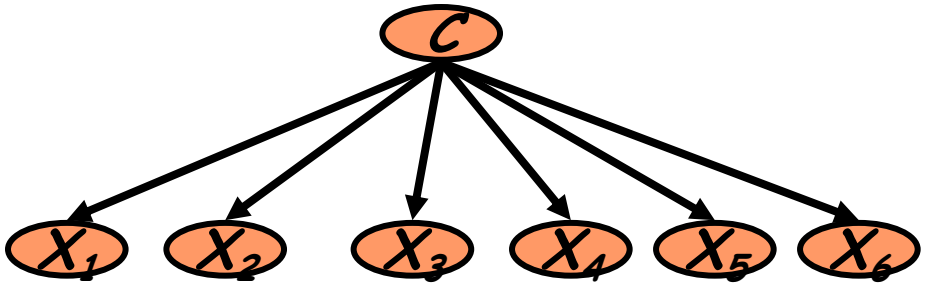
The Naïve Bayes Classifier



- **Conditional Independence Assumption:** features are independent of each other given the class:

$$P(X_1,\ldots,X_5 \mid C) = P(X_1 \mid C) \bullet P(X_2 \mid C) \bullet \cdots \bullet P(X_5 \mid C)$$

Learning the Model

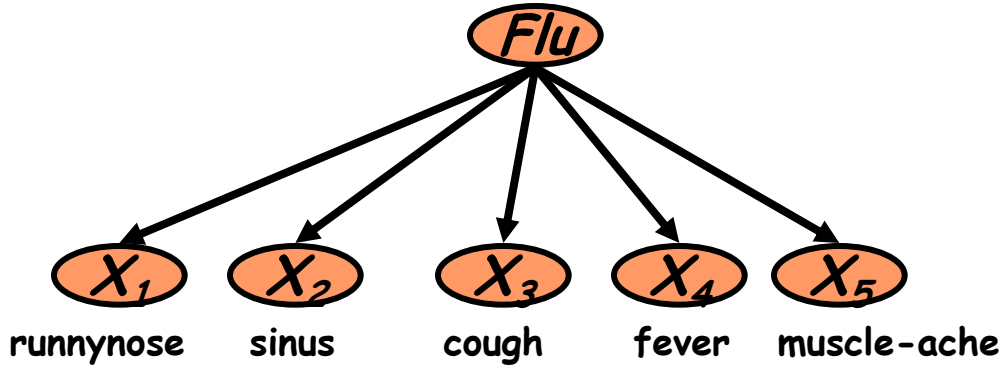


- Common practice: maximum likelihood
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{N(C = c_j)}{N}$$

$$\hat{P}(x_i \mid c_j) = \frac{N(X_i = x_i, C = c_j)}{N(C = c_j)}$$

Problem with Max Likelihood



$$P(X_1,\ldots,X_5 \mid C) = P(X_1 \mid C) \bullet P(X_2 \mid C) \bullet \cdots \bullet P(X_5 \mid C)$$

- What if we have seen no training cases where patient had no flu and muscle aches?

$$\hat{P}(X_5 = t \mid C = nf) = \frac{N(X_5 = t, C = nf)}{N(C = nf)} = 0$$

- Zero probabilities cannot be conditioned away, no matter the other evidence!

$$\ell = \arg\max_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

Smoothing to Avoid Overfitting

$$\hat{P}(x_i \mid c_j) = \frac{N(X_i = x_i, C = c_j) + 1}{N(C = c_j) + k}$$

# of values of X_i

- Somewhat more subtle version

overall fraction in data where $X_i = x_{i,k}$

$$\hat{P}(x_{i,k} \mid c_j) = \frac{N(X_i = x_{i,k}, C = c_j) + mp_{i,k}}{N(C = c_j) + m}$$

extent of "smoothing"

Using Naive Bayes Classifiers to Classify Text: Basic method

- Attributes are text positions, values are words.

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}}\ P(c_j)\prod_i P(x_i \mid c_j)$$

$$= \underset{c_j \in C}{\operatorname{argmax}}\ P(c_j)P(x_1 = \text{"our"} \mid c_j)\cdots P(x_n = \text{"text"} \mid c_j)$$

- Naive Bayes assumption is clearly violated.
- Still too many possibilities
- Assume that classification is *independent* of the positions of the words (Use same parameters for each position)

Training (learning)

- From training corpus, extract *Vocabulary*
- Calculate required $P(c_j)$ and $P(x_k \mid c_j)$ terms
  - For each c_j in C do
    - $docs_j \leftarrow$ subset of documents for which the target class is c_j
    - $$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$
    - $Text_j \leftarrow$ single document containing all $docs_j$
    - for each word x_k in *Vocabulary*
      - $n_k \leftarrow$ number of occurrences of x_k in $Text_j$

$$P(x_k \mid c_j) \leftarrow \frac{n_k + 1}{n + |Vocabulary|}$$

Testing (Classifying)

- positions ← all word positions in current document which contain tokens found in *Vocabulary*
- Return c_{NB}, where

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} \; P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$