Northeastern

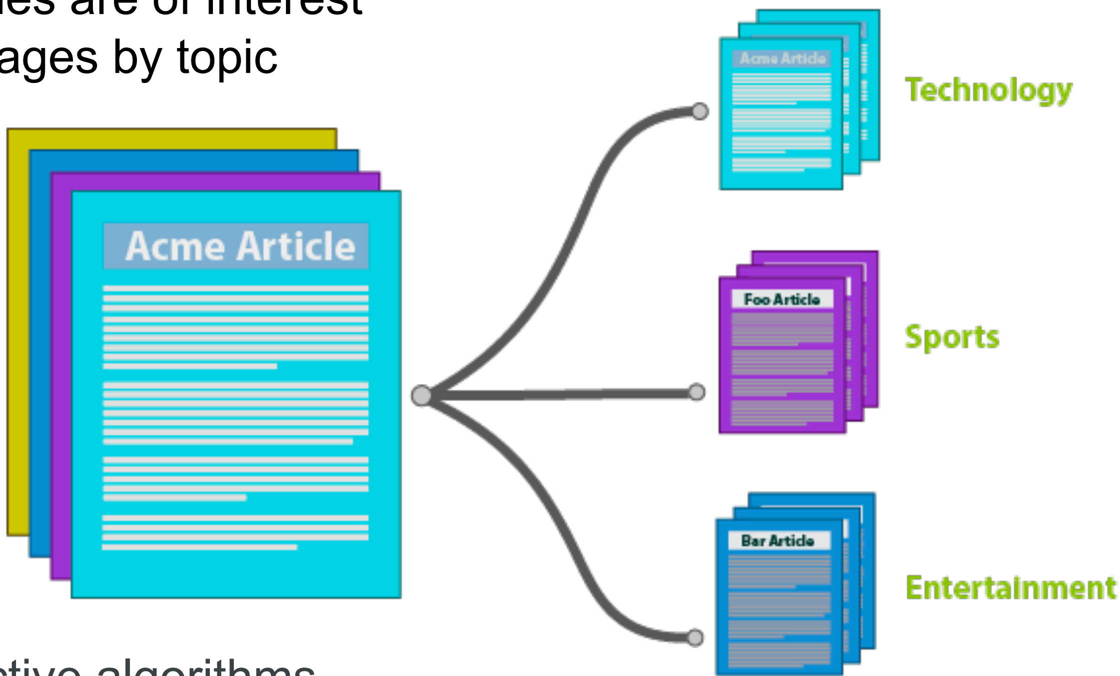
EECE2300
Computational Methods for Data Analytics

Naïve Bayes for Text Classification

Learning to Classify Text

Why?

- ➤ Learn which news articles are of interest
- ➤ Learn to classify web pages by topic



Naïve Bayes is among most effective algorithms

What attributes should we use to represent text documents??

Learning to Classify Text: Bayes-Approach

Target concept *Interesting?* : *Document → {+, -}*

1. Represent each document by vector of words
   - One attribute per word position in document
2. Learning: Use training examples to estimate
   - P(+)
   - P(-)
   - P(doc | +)
   - P(doc | -)

Naïve Bayes conditional independence assumption

$$P(doc|v_j) = \prod_{i=1}^{length(doc)} P(a_i = w_k|v_j)$$

Where $P(a_i = w_k|v_j)$ is the probability that word in position *i* is w_k, given v_j

One more assumption: $P(a_i = w_k|v_j) = P(a_m = w_k|v_j), \angle i, m$

Naïve Bayes for Text Classification
An Example: Movie Reviews

A set of reviews (documents) and a classification

Doc	Text	Class
1	I loved the movie	+
2	I hated the movie	-
3	A great movie. Good movie.	+
4	Poor acting	-
5	Great acting. A good movie	+

10 unique words
<I, loved, the, movie, a, hated, great, good, poor, acting>

Naïve Bayes for Text Classification
An Example: Movie Reviews

Convert the documents into feature sets, where the attributes are possible words, and the values are the number of times a word occurs in the given document.

Doc	I	loved	the	acting	movie	a	hated	great	good	poor	Class
1	1	1	1		1						+
2	1		1		1		1				-
3					2	1		1	1		+
4				1						1	-
5				1	1	1		1	1		+

Let's look at the probabilities per outcome (+ or -)

Naïve Bayes for Text Classification
An Example: Movie Reviews

Doc	I	loved	the	acting	movie	a	hated	great	good	poor	Class
1	1	1	1		1						+
3					2	1		1	1		+
5				1	1	1		1	1		+

$$p(+) = \frac{3}{5} = 0.6$$

Compute: $p(I\,|+)$ $p(love|+)$ $p(\,acting|+)$ $p(great\,|+)$

$p(the\,|+)$ $p(good\,|+)$ $p(\,movie|+)$ $p(a\,|+)$

Let *n* be the number of words in the (+) case: 14. n_k the number of times word *k* occurs in these cases (+)

Let $P(w_k|+) = \frac{n_k+1}{n+|Vocabulary|}$

Naïve Bayes for Text Classification
An Example: Movie Reviews

Doc	I	loved	the	acting	movie	a	hated	great	good	poor	Class
1	1	1	1		1						+
3					2	1		1	1		+
5				1	1	1		1	1		+

$$p(+) = \frac{3}{5} = 0.6 \quad P(w_k|+) = \frac{n_k + 1}{n + |Vocabulary|}$$

$$p(I\ |+) = \frac{1+1}{14+10} = 0.0833;$$

$$p(loved|+) = 0.0833$$

$$p(the\ |+) = 0.0833$$

$$p(\ movie|+) = 0.2083$$

$$p(a\ |+) = 0.125$$

$$p(great\ |+) = 0.125$$

$$p(\ acting|+) = 0.0833$$

$$p(good\ |+) = 0.125$$

$$p(hated|+) = 0.417$$

$$p(poor|+) = 0.417$$

Naïve Bayes for Text Classification
An Example: Movie Reviews

Doc	I	loved	the	acting	movie	a	hated	great	good	poor	Class
2	1		1		1		1				-
4				1						1	-

$$p(-) = \frac{2}{5} = 0.4 \quad P(w_k|-) = \frac{n_k + 1}{n + |Vocabulary|}$$

$$p(I\,|-) = \frac{1+1}{6+10} = 0.125$$

$$p(the\,|-) = 0.125$$

$$p(a\,|-) = 0.125$$

$$p(\,acting|-) = 0.125$$

$$p(hated|-) = 0.125$$

$$p(loved|-) = 0.0625$$

$$p(\,movie|-) = 0.125$$

$$p(great\,|-) = 0.0625$$

$$p(good\,|-) = 0.125$$

$$p(poor|-) = 0.0625$$

Naïve Bayes for Text Classification
An Example: Movie Reviews

Now that we've trained our classifier

Let's classify a new sentence according to:

$$V_{\mathrm{NB}} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_{w \in words} P(w|v_j)$$

where V stands for "value" or "class"

"I hated the poor acting"

$$if\ v_j = +; \quad p(+)p(I|+)p(hated|+)p(the|+)\ p(poor|+)\ p(acting|+) = 6.03x10^{-7}$$
$$if\ v_j = -; \quad p(-)p(I|-)p(hated|-)\ p(the|-)\ p(poor|-)\ p(acting|-) = 1.22x10^{-5}$$

Learning to Classify Text

Learn_naive_Bayes_text(Examples, V)

1. Collect all words and other tokens that occur in *Examples*

- Vocabulary (V) ← All distinct words and tokens in *Examples*

2. Calculate the required $P(v_j)$ and $P(w_k|\, v_j)$ probability terms

- For each target value v_j in V do
  - $docs_j$ ← subset of *Examples* for which the target value is v_j
  - $P(v_j) \leftarrow \frac{|docs|}{|Examples|}$
  - $Text_j$ ← a single document created by concatenating all members of $docs_j$
  - n ← total number of words in $Text_j$ (counting duplicates each time)
  - For each word w_k in V
    - n_k ← number of times word w_k occurs in $Text_j$
    - $P(w_k|\, v_j) \leftarrow \frac{n_k+1}{n+|V|}$

Learning to Classify Text

Classify_naive_Bayes_text(Doc)

- Positions ← all word positions that contain tokens found in Vocabulary
- Return V_{NB} where

$$V_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_{w \in words} P(w|v_j)$$