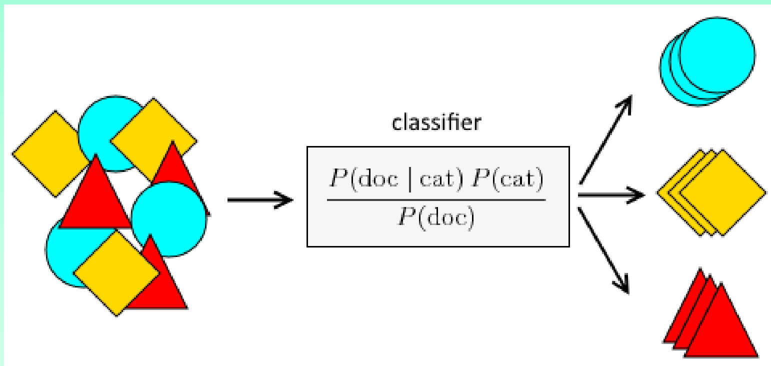
Text Classification



Prof. Rinaldo Lima – rjlima01@gmail.com
DEINFO – MESTRADO - PPGIA

Tópicos da Aula

- The Task of Text Classification (TC)
- Document Representation
  - Vector Model (Geometric Model)
  - Bag of Words
- Text Classification Algorithms
- Implemention Aspects
- TC Applications

3.2

The Task of Text Classification

Text Classification: definition

- *Input*:
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
- *Output*: a predicted class $c \in C$

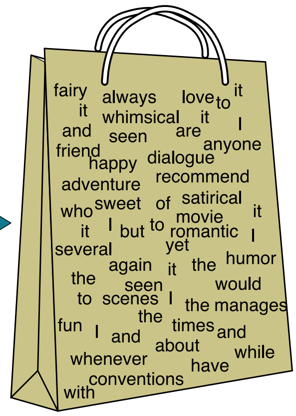
But how to represent the input (D, C)?

Document Representation

Bag of Words

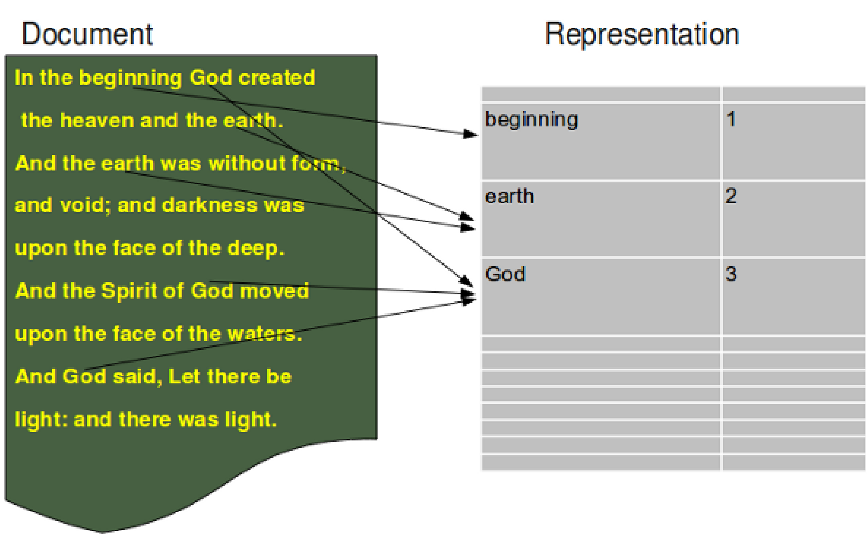
Document Representation: Bag of Words (BOW)

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

Document Representation: BOW



Classification Hypothesis

γ(I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun… It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet.)=c

Classification Hypothesis

γ(I **love** this movie! It's **sweet**, but with **satirical** humor. The dialogue is **great** and the adventure scenes are **fun…** It manages to be **whimsical** and **romantic** while **laughing** at the conventions of the fairy tale genre. I would **recommend** it to just about anyone. I've seen it **several** times, and I'm always **happy** to see it **again** whenever I have a friend who hasn't seen it yet.)=c

Classification Hypothesis

$\gamma($

x **love** xxxxxxxxxxxxxxxx **sweet**
xxxxxxx **satirical** xxxxxxxxxx
xxxxxxxxxxx **great** xxxxxxx
xxxxxxxxxxxxxxxxxxx **fun** xxxx
xxxxxxxxxxxxx **whimsical** xxxx
romantic xxxx **laughing**
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxx **recommend** xxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx **several** xxxxxxxxxxxxxxxxx
xxxxx **happy** xxxxxxxxx **again**
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxx

$)=c$

Using a subset of the words

Classification Hypothesis

$\gamma($

great	2
love	2
recommend	1
laugh	1
happy	1
...	...

$)=c$

Predicting the Future...

Is there a parttern here?

word1	word2	word3	word4	word5	...	wordN	label
0	1	0	1	0	...	1	1
1	0	1	0	0	...	0	0
1	0	0	1	1	...	0	1
0	1	0	0	1	...	1	0
1	0	0	1	0	...	1	1
0	1	0	0	1	...	0	1
0	0	1	1	0	...	0	0
0	0	1	0	1	...	0	1

Figure 3.5. Spreadsheet with no Obvious Patterns

Predicting the Future...

word1	word2	word3	word4	word5	...	wordN	label
0	1	1	1	0	...	1	1
1	0	1	0	0	...	1	0
1	1	0	1	1	...	0	1
0	1	0	0	1	...	1	0
1	1	1	1	0	...	1	1
0	1	0	1	1	...	0	1
1	0	1	1	0	...	1	0
0	1	1	1	1	...	0	1

Figure 3.4. Predictive Patterns in a Spreadsheet

Is there a parttern now?

Predicting the Future

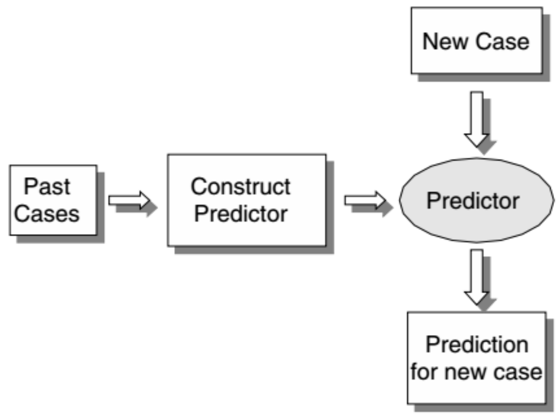


Figure 3.1. Predicting the Future Based on the Past

The classical prediction problem for text is text classification
The goal is to assign a category or topic to a new document

Training a Classifier

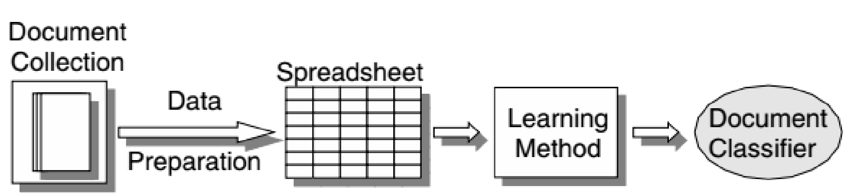


Figure 3.7. From Text to Classifiers

The classical prediction problem for text is text classification
The goal is to assign a category or topic to a new document

Document Representation

Vector Model

Vector Space Model

- The Vector Model is also known as **Document-term Matrix** representation
- Each row of this matrix constitutes a **binary vector** representing a document ***D*** in the collection
- Two documents have exactly the same vector representation in the vector space if they just contain the same words, even in diferent word

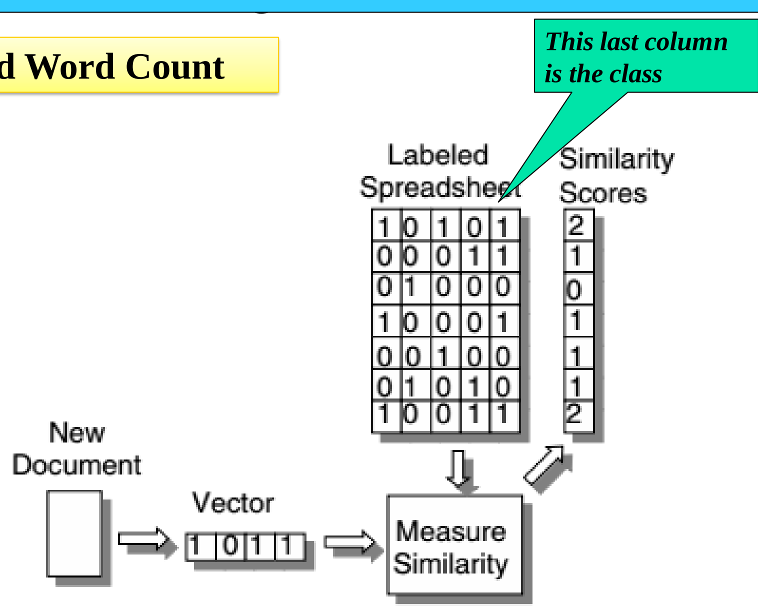
Vector Space Model

Vector Similarity

- The most obvious measure of similarity between documents is a **count of their shared words**
- We look at all the words in the new document and for each document in the collection, we count how many of these words appear together
- The quality can be good for a relatively small vocabulary, but performance can degrade with larger dictionaries.

Computing Similarity Scores for Documents

Shared Word Count



Vector Space Model: Similarity Scores

Word Count and Bonus

- In high dimensions, it is difficult to readily discriminate between predictive and weakly predictive words
- Intead, we can compute the similarity between a new document that contains K words and document D(i) as:

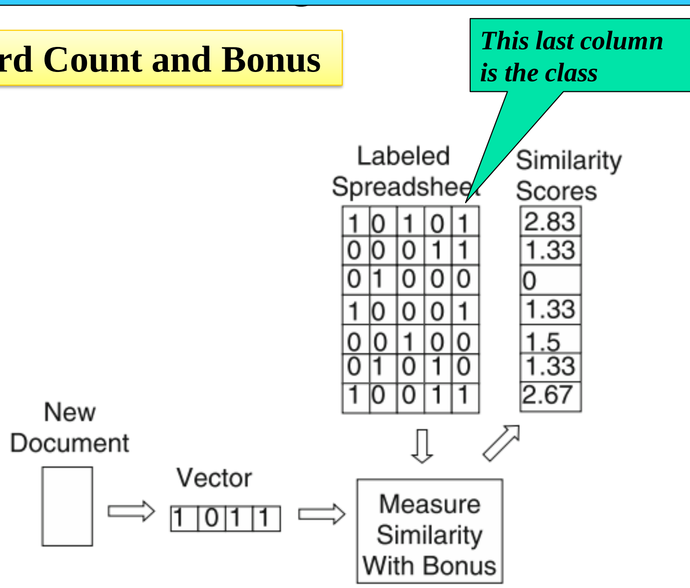
$$\text{Similarity}(D(i)) = \sum_{j=1}^{K} w(j),$$

$$w(j) = \begin{cases} 1 + 1/\text{df}(j), & \text{if word } (j) \text{ occurs in both documents,} \\ 0, & \text{otherwise.} \end{cases}$$

- The bonus is 1/df(j) where df(j) is the number of documents in which the word j occurs in the collection, a variant of idf (inverse document frequence)

Computing Similarity Scores for Documents

Word Count and Bonus



Vector Generation

For Prediction

Vector Generation for Prediction (Feature Generation)

- The collective set of features is called Dictionary
- The dictionary of words covers all the possibilities and correspond to the number of columns in the dataset

Hash Trick

Corpus-based Frequencies

Algorithm for FG

```
Input:
  ts, all the tokens in the document collection
  k, the number of features desired
Output:
  fs, a set of k features
Initialize:
  hs := empty hashtable

for each tok in ts do
  If hs contains tok then
    i := value of tok in hs
    increment i by 1
  else
    i := 1
  endif
  store i as value of tok in hs
endfor
sk := keys in hs sorted by decreasing value
fs := top k keys in sk
output fs
```

Figure 2.4. Generating Features from Tokens

Vector Generation for Prediction (Feature Generation)

- Each word type correspond to a feature (column)
- Example of a document vector using binary scoring method

Table 2.2. Dictionary Feature Transformations

Word Pairs, Collocations
Frequencies
tf-idf

Table 2.3. Thresholding Frequencies to Three Values

0 - word did not occur
1 - word occurred once
2 - word occurred 2 or more times

Converting docs to vectors

```
Input:
  fs, a set of k features
  dc, a collection of n documents
Output: ss, a spreadsheet with n rows and k columns
Initialize: i := 1

for each document d in dc, do
  j := 1
  for each feature f in fs, do
    m := number of occurrences of f in d
    if (m > 0) then ss(row=i, col=j) := 1;
    else ss(row=i, col=j) := 0 ;
    endif
    increment j by 1
  endfor
  increment i by 1
endfor
output ss
```

TEXT CLASSIFICATION

via

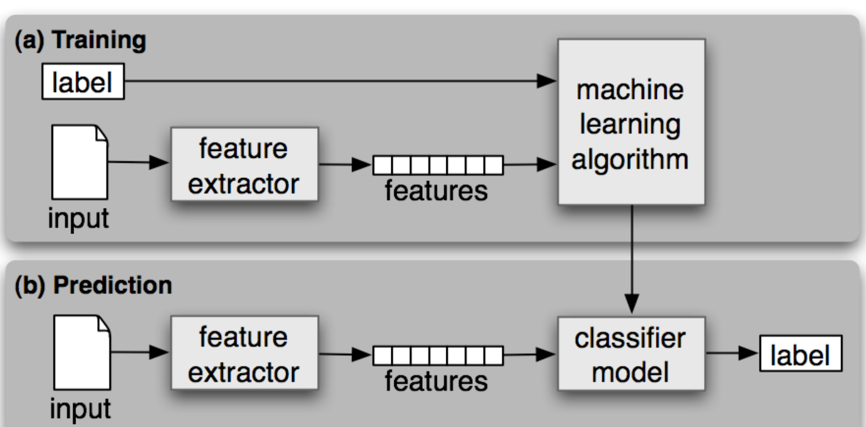
Supervised Machine Learning

Text Classification as a Supervised Learning Task

Classification Methods: Supervised Machine Learning

- *Input:*
  - a document d
  - a fixed set of classes $C = \{c_1, c_2,\ldots, c_J\}$
  - A training set of m hand-labeled documents $(d_1,c_1),\ldots,(d_m,c_m)$
- *Output:*
  - a learned classifier $\gamma: d \rightarrow c$

Text Classification as a Supervised Learning Task



Text Classification as a Supervised Learning Task

- Any kind of classifier
  - Naïve Bayes
  - Logistic regression
  - Support-vector machines
  - k-Nearest Neighbors

TEXT CLASSIFICATION:

Naive Bayes Algorithm

Naive Bayes

Bayes' Rule Applied to Documents and Classes

- For a document *d* and a class *c*

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$

Naive Bayes

Naïve Bayes Classifier (I)

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Dropping the denominator

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c)P(c)$$

Document d represented as features x1..xn

Naive Bayes

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

O($|X|^n \bullet |C|$) parameters

Could only be estimated if a very, very large number of training examples was available.

How often does this class occur?

We can just count the relative frequencies in a corpus

Naive Bayes

Multinomial Naïve Bayes Independence Assumptions

$$P(x_1, x_2, \ldots, x_n \mid c)$$

- **Bag of Words assumption**: Assume position doesn't matter
- **Conditional Independence**: Assume the feature probabilities $P(x_i|c_j)$ are independent given the class c.

$$P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

Naive Bayes

Multinomial Naïve Bayes Classifier

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

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \prod_{x \in X} P(x \mid c)$$

Naive Bayes

Learning the Multinomial Naïve Bayes Model

- First attempt: maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

Naive Bayes

Multinomial Naïve Bayes: Learning

- From training corpus, extract *Vocabulary*

- Calculate $P(c_j)$ terms
  - For each c_j in C do
    $docs_j \leftarrow$ all docs with class $=c_j$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

- Calculate $P(w_k \mid c_j)$ terms
  - $Text_j \leftarrow$ single doc containing all $docs_j$
  - For each word w_k in *Vocabulary*
    $n_k \leftarrow$ # of occurrences of w_k in $Text_j$

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

Naive Bayes Algorithm: Sentiment Analysis

$|V| = 20$
$Nw+ = 9$
$Nw- = 14$

	Cat	Documents
Training	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Test	?	predictable with no fun

The prior $P(c)$ for the two classes is computed via Eq. 6.12 as $\frac{N_c}{N_{doc}}$: $P(-) = \frac{3}{5}$ $P(+) = \frac{2}{5}$

$$P(\text{“predictable”}|-) = \frac{1+1}{14+20} \qquad P(\text{“predictable”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“no”}|-) = \frac{1+1}{14+20} \qquad P(\text{“no”}|+) = \frac{0+1}{9+20}$$

$$P(\text{“fun”}|-) = \frac{0+1}{14+20} \qquad P(\text{“fun”}|+) = \frac{1+1}{9+20}$$

$$P(-)P(S|-) = \frac{3}{5} \times \frac{2 \times 2 \times 1}{34^3} = 6.1 \times 10^{-5} \qquad P(+)P(S|+) = \frac{2}{5} \times \frac{1 \times 1 \times 2}{29^3} = 3.2 \times 10^{-5}$$

Classe Negativa **Classe Positiva**

The model thus predicts the class *negative* for the test sentence.

Naive Bayes Algorithm: Example

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

$$\hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

$|V| = 6$

Conditional Probabilities:

P(Chinese|*c*) = (5+1) / (8+6) = 6/14 = 3/7

P(Tokyo|*c*) = (0+1) / (8+6) = 1/14

P(Japan|*c*) = (0+1) / (8+6) = 1/14

P(Chinese|*j*) = (1+1) / (3+6) = 2/9

P(Tokyo|*j*) = (1+1) / (3+6) = 2/9

P(Japan|*j*) = (1+1) / (3+6) = 2/9

Choosing a class:

P(c|d5) $\propto$ 3/4 * $(3/7)^3$ * 1/14 * 1/14 $\approx$ 0.0003

P(j|d5) $\propto$ 1/4 * $(2/9)^3$ * 2/9 * 2/9 $\approx$ 0.0001

41

Naive Bayes Algorithm

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms
  N_{doc} = number of documents in D
  N_c = number of documents from D in class c
  $logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$
  $V \leftarrow$ vocabulary of D
  $bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c
  for each word w in V   # Calculate $P(w|c)$ terms
    $count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
    $loglikelihood[w,c] \leftarrow \log \frac{count(w,c)\ +\ 1}{\sum_{w' \text{ in } V} (count\ (w',c)\ +\ 1)}$
return *logprior*, *loglikelihood*, *V*

function TEST NAIVE BAYES(*testdoc*, *logprior*, *loglikelihood*, C, V) **returns** best *c*

for each class $c \in C$
  $sum[c] \leftarrow logprior[c]$
  for each position *i* in *testdoc*
    $word \leftarrow testdoc[i]$
    if $word \in V$
      $sum[c] \leftarrow sum[c] + loglikelihood[word,c]$
return $\text{argmax}_c\ sum[c]$

Figure 6.2 The naive Bayes algorithm, using add-1 smoothing. To use add-α smoothing instead, change the $+1$ to $+\alpha$ for loglikelihood counts in training.

TEXT CLASSIFICATION:

K Nearest Neighbors Algorithm

K-Nearest Neighbors: Finding Similar Docs

- K-NN is one of the most prominent approaches to TC
- K is the parameter model
- K also indicates de number of neighbors to be considered

1. Compute the similarity of newDoc to all documents in collection {D(l)}.
2. Select the k documents that are most similar to newDoc.
3. The answer is the label that occurs most frequently in the k selected documents.

Figure 3.8. Basic Nearest-Neighbor Algorithm for Documents

K-Nearest Neighbors: Finding Similar Docs

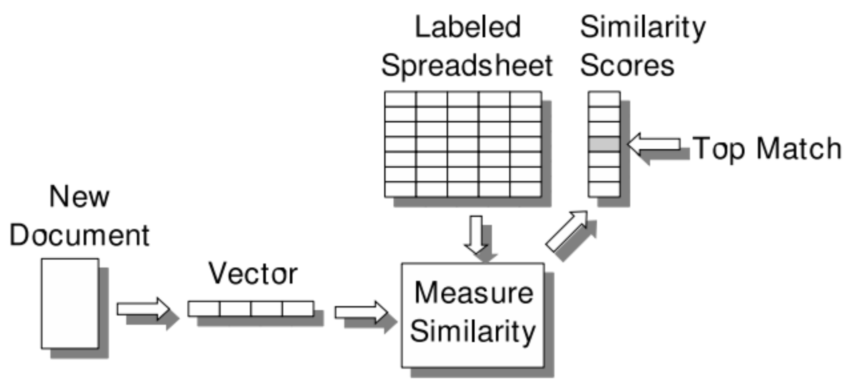
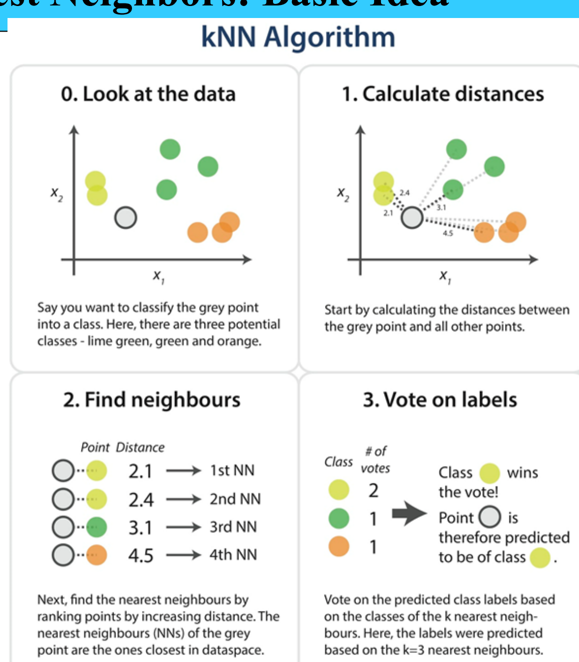


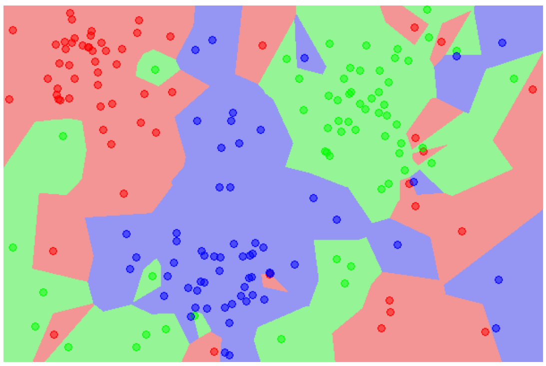
Figure 3.9. Finding Similar Documents

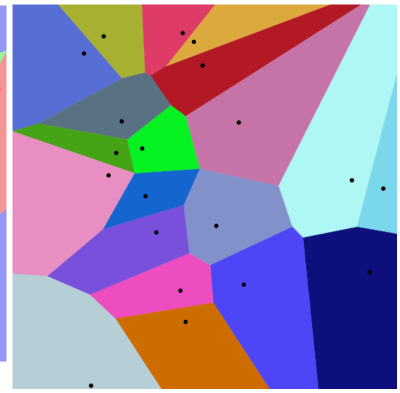
K-Nearest Neighbors: Basic Idea



K-Nearest Neighbors: Geometric Interpretation

Voronoi Diagram with Decision Boundaries





K = 1

K = 3

Text Classification:

Evaluation Methodology

Text Classification: Evaluation Methodology

Estimating Current and Future Prediction

Training and Test datasets

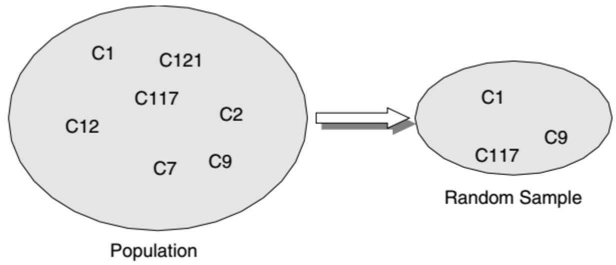


Figure 3.20. Drawing a Random Sample from a Population

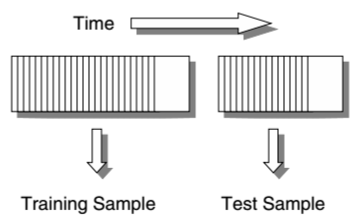
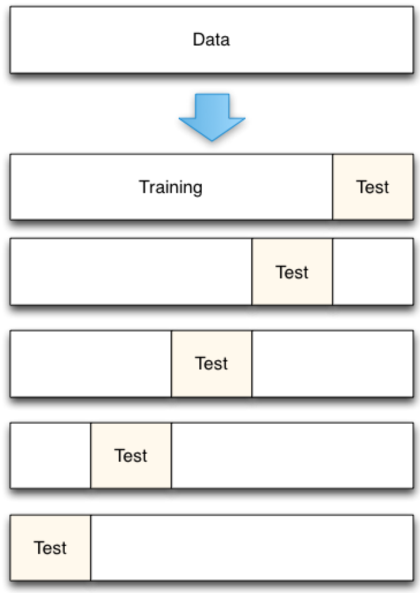


Figure 3.21. Partitioning Documents into Training and Test Sets

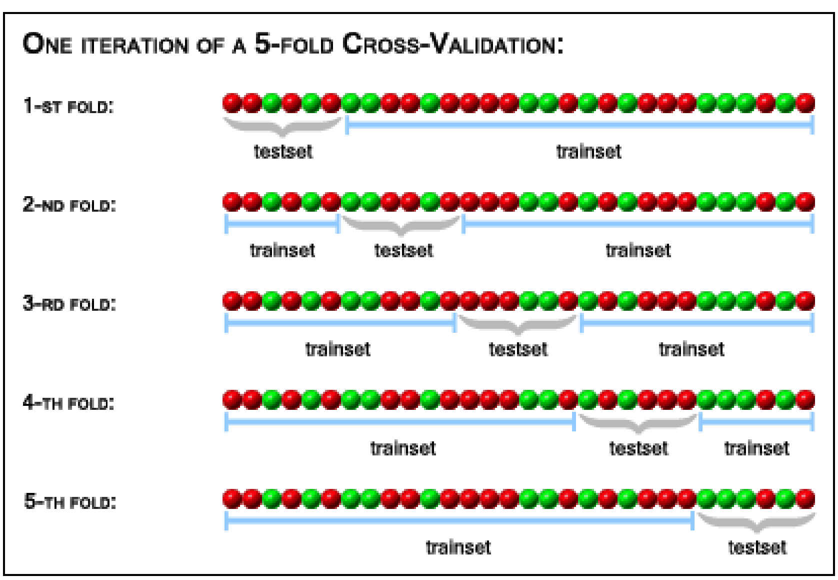
Evaluation Methodology: n-Fold CrossValidation

- N is the number of folds.
  - A typical value is N= 10
- We randonly choose a **training** and **test** datasets
  1. Train the classifier using the training dataset
  2. Compute the error rate or other measurement on the test dataset
- This process is repeated with a different randomly selected training and test datasets
- We do this sampling N times and average these N runs to get an average error rate
- It is mainly used for small datasets

Evaluation Methodology: n-Fold CrossValidation



Evaluation Methodology: n-Fold CrossValidation



Text Classification: Evaluation Methodology

Measure for Classification

The standard measure for classification is the error rate

$$\text{Error rate}(erate) = \frac{\text{number of errors}}{\text{number of documents}},$$

$$\text{Standard Error}(SE) = \sqrt{\frac{erate * (1 - erate)}{\text{number of documents}}}.$$

Text Classification: Evaluation Methodology

Measures for Classification

Precision is to measure the quality of our predictions only based on what our predictor **claims to be positive** (regardless of all it might miss)

$$\text{Precision} = \frac{\text{All we predicted correctly}}{\text{All we predicted, correctly or wrongly}}$$

Text Classification: Evaluation Methodology

Measures for Classification

Recall is to measure such quality with respect to the mistakes we did (what should have been predicted as positive but we flagged as negative)

$$\text{Recall} = \frac{\textbf{All we predicted correctly}}{\textbf{All we should have predicted}}$$

Text Classification: Evaluation Methodology

Measures for Classification

- F-mesure is the harmonic average between Precision and Recall

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Confusion Matrix (Contigence Table)

		Actual Value (as confirmed by experiment)	
		positives	negatives
Predicted Value (predicted by the test)	positives	**TP** True Positive	**FP** False Positive
	negatives	**FN** False Negative	**TN** True Negative

Confusion Matrix (Contigence Table)

Predictive Model: Evaluation

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

		actual result / classification	
		yes	**no**
predictive result / classification	**yes**	**tp** (true positive)	**fp** (false positive) ← Type 1 error
	no	**fn** (false negative)	**tn** (true negative)

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Confusion Matrix (Contigence Table) : Example

numerical form

predicted → real ↓	Class_pos	Class_neg
Class_pos	114	86
Class_neg	7	93

→

percentage form

predicted → real ↓	Class_pos	Class_neg
Class_pos	38%	29%
Class_neg	2%	31%

numerical form

predicted → real ↓	Class_1	Class_2	Class_3
Class_1	94	16	10
Class_2	21	113	16
Class_3	4	4	92

→

percentage form

predicted → real ↓	Class_1	Class_2	Class_3
Class_1	25%	4%	3%
Class_2	6%	31%	4%
Class_3	1%	1%	25%

Text Classification: Evaluation Methodology

Trade-off between P and R

Increasing the precision lowers the recall and vice-versa.

But, is it possible to adjust the precision and recall of a classifier?

- For k-NN methods a value of K < 3 boost **R**, while values greater than 3 would boost **P**
- For the NB classifier, the threshold for a class can be altered from 0.5 to some other value.
  - Lower values boost **R**, while higher values boost **P**

Text Classification:

Implementation Aspects

Implemention Aspects concerning Performance

- Stopwords
- Lemmatization
- Frequent Words - using just N top frequent words
- Removing Rare Words (usually typos)
- Synomyms - using just one word sense
- Local Dictionary – restricting features generation by class
- **Feature Selection by Attribute Ranking**
  - The goal is to select a set of features for each class by ranking features attributes according to their predictive abilities for the category under consideration.

    Ex. **Information Gain**

Implemention Aspects concerning Performance

Underflow Prevention: log space

- Multiplying lots of probabilities can result in floating-point underflow.
- Since log(*xy*) = log(*x*) + log(*y*)
  - Better to sum logs of probabilities instead of multiplying probabilities.
- Class with highest un-normalized log probability score is still most probable.

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

- Model is now just max of sum of weights

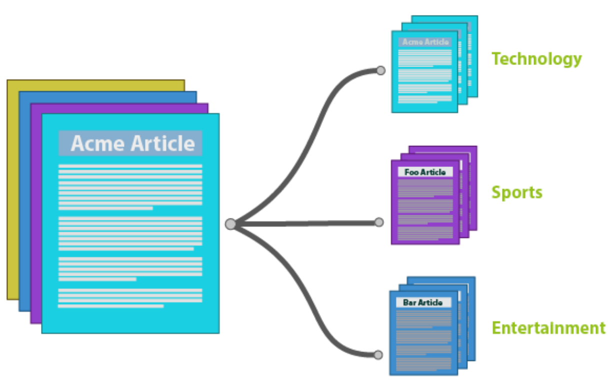
Token Weighting Scoring Methods

- **Binary** - assigning 0 or 1 if a given word w appears in a document
- **Term Frequency** (TF) – the number of times a word appears in a document
- **N-Grams** - Unigram, bigram, trigram as features (or columns) in a document
- **Word Count with Bonus**
- **TF/IDF** **-** Term Frequency/Inverse Document Frequency
  → (Next classes)

TEXT CLASSIFICATION:

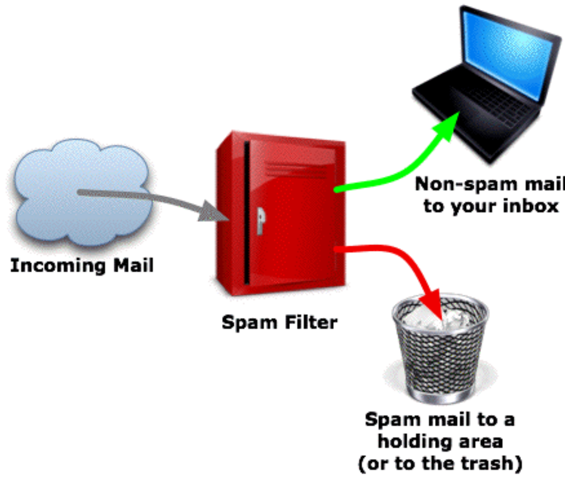
APPLICATIONS

Text Categorization



- Automatic categorization of News Articles: Sports, Science, Health, etc
- Articles categorization into subject areas, etc

Spam Filtering



- The most robust solutions to spam filtering are based on Naive Bayes

A tipical Text Classification Task: Spam filtering

...	unsubscribe	...	enlargement	...	ink	...	spam
...	yes	...	yes	...	yes	...	true
...	no	...	no	...	no	...	false
...	...	...	...	...	...	...	...

Abstract spreadsheet for spam prediction

Classes or Categories

Sentiment Analysis

- Subjectivity Determination
- Polarity Classification

TC Dataset

Reuters Text Categorization data set (Reuters-21578) document

<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="12981" NEWID="798">

<DATE> 2-MAR-1987 16:51:43.42</DATE>

<TOPICS><D>livestock</D><D>hog</D></TOPICS>

<TITLE>AMERICAN PORK CONGRESS KICKS OFF TOMORROW</TITLE>

<DATELINE> CHICAGO, March 2 - </DATELINE><BODY>The American Pork Congress kicks off tomorrow, March 3, in Indianapolis with 160 of the nations pork producers from 44 member states determining industry positions on a number of issues, according to the National Pork Producers Council, NPPC.

Delegates to the three day Congress will be considering 26 resolutions concerning various issues, including the future direction of farm policy and the tax law as it applies to the agriculture sector. The delegates will also debate whether to endorse concepts of a national PRV (pseudorabies virus) control and eradication program, the NPPC said.

A large trade show, in conjunction with the congress, will feature the latest in technology in all areas of the industry, the NPPC added. Reuter

Text Classification – Comparing k-NN and NB

Equipe 1: NB and variant
Equipe 2: k-NN and variant

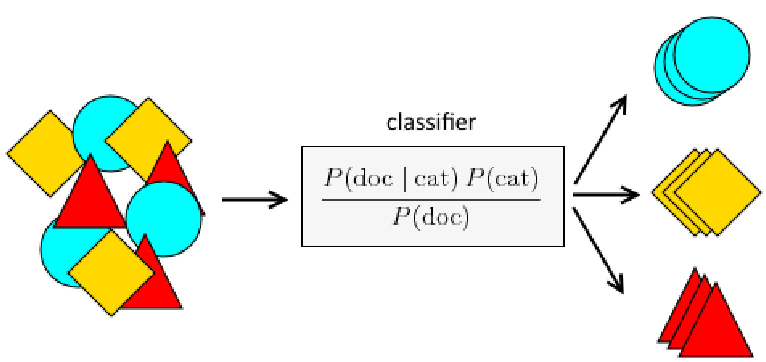
Usar o Corpus Reuters de Classificação de Texto

Cada equipe deve implementar a versão clássica do kNN e do NB e pelo menos uma de suas variações que serão fornecidas depois

Os algoritmos devem ser implementados.

Proibido usar os algoritmos de bibliotecas já prontas

Text Classification



Prof. Rinaldo Lima – rjlima01@gmail.com
DEINFO – MESTRADO - PPGIA