BLG 540E
TEXT RETRIEVAL SYSTEMS

Text Classification

Arzucan Özgür

Faculty of Computer and Informatics, İstanbul Techical University

April 1, 2011

Standing queries

- The path from IR to text classification:
  - You have an information need to monitor, say:
    - Multicore computer chips
  - You want to rerun an appropriate query periodically to find new news items on this topic
  - You will be sent new documents that are found
    - I.e., it's text classification not ranking
- Such queries are called ***standing queries***
  - Long used by "information professionals"
  - A modern mass instantiation is **Google Alerts**
- Standing queries are (hand-written) text classifiers

Spam recognition: Another text classification task

Return-Path: <ig_esq@rediffmail.com>
X-Sieve: CMU Sieve 2.2
From: "Ibrahim Galadima" <ig_esq@rediffmail.com>
Reply-To: galadima_esq@netpiper.com
To: webmaster@aclweb.org
Date: Tue, 14 Jan 2003 21:06:26 -0800
Subject: Gooday

DEAR SIR

FUNDS FOR INVESTMENTS

THIS LETTER MAY COME TO YOU AS A SURPRISE SINCE I HAD NO PREVIOUS CORRESPONDENCE WITH YOU

I AM THE CHAIRMAN TENDER BOARD OF INDEPENDENT NATIONAL ELECTORAL COMMISSION INEC I GOT YOUR CONTACT IN THE COURSE OF MY SEARCH FOR A RELIABLE PERSON WITH WHOM TO HANDLE A VERY CONFIDENTIAL TRANSACTION INVOLVING THE ! TRANSFER OF FUND VALUED AT TWENTY ONE MILLION SIX HUNDRED THOUSAND UNITED STATES DOLLARS US$20M TO A SAFE FOREIGN ACCOUNT

THE ABOVE FUND IN QUESTION IS NOT CONNECTED WITH ARMS, DRUGS OR MONEY LAUNDERING IT IS A PRODUCT OF OVER INVOICED CONTRACT AWARDED IN 1999 BY INEC TO A

More Text Classification Examples
Many search engine functionalities use classification

- Assigning labels to documents or web-pages:
- Labels are most often topics such as Yahoo-categories
  - *"finance," "sports," "news>world>asia>business"*
- Labels may be genres
  - *"editorials" "movie-reviews" "news"*
- Labels may be opinion on a person/product
  - *"like", "hate", "neutral"*
- Labels may be domain-specific
  - *"interesting-to-me" : "not-interesting-to-me"*
  - *"contains adult language" : "doesn't"*
  - *language identification: English, French, Chinese, …*
  - *search vertical: about Linux versus not*
  - *"spam" : "not spam"*

Classification Methods (1)

- Manual classification
  - Used by the original Yahoo! Directory
  - about.com, ODP, PubMed
  - Very accurate when job is done by experts
  - Consistent when the problem size and team is small
  - Difficult and expensive to scale
    - Means we need automatic classification methods for big problems

Classification Methods (2)

- Automatic document classification
  - Hand-coded rule-based systems
    - One technique used by Reuters, CIA, etc.
    - It's what Google Alerts is doing (Standing queries)
    - Companies provide "IDE" for writing such rules
    - E.g., assign category if document contains a given boolean combination of words

      (multicore OR multi-core) AND (chip OR processor OR microprocessor)
    - Accuracy is often very high if a rule has been carefully refined over time by a subject expert
    - Building and maintaining these rules is expensive

A Verity topic
A complex classification rule

```
comment line          # Beginning of art topic definition
top-level topic       art ACCRUE
                          /author = "fsmith"
topic definition modifiers
                          /date   = "30-Dec-01"
                          /annotation = "Topic created
                                         by fsmith"
subtopic topic        * 0.70 performing-arts ACCRUE
  evidence topic      ** 0.50 WORD
  topic definition modifier
                          /wordtext = ballet
  evidence topic      ** 0.50 STEM
  topic definition modifier
                          /wordtext = dance
  evidence topic      ** 0.50 WORD
  topic definition modifier
                          /wordtext = opera
  evidence topic      ** 0.30 WORD
  topic definition modifier
                          /wordtext = symphony
subtopic              * 0.70 visual-arts ACCRUE
                      ** 0.50 WORD
                          /wordtext = painting
                      ** 0.50 WORD
                          /wordtext = sculpture
subtopic              * 0.70 film ACCRUE
                      ** 0.50 STEM
                          /wordtext = film
subtopic              ** 0.50 motion-picture PHRASE
                      *** 1.00 WORD
                          /wordtext = motion
                      *** 1.00 WORD
                          /wordtext = picture
                      ** 0.50 STEM
                          /wordtext = movie
subtopic              * 0.50 video ACCRUE
                      ** 0.50 STEM
                          /wordtext = video
                      ** 0.50 STEM
                          /wordtext = vcr
                      # End of art topic
```

- Note:
  - maintenance issues
  - Hand-weighting of terms

[Verity was bought by Autonomy.]

Classification Methods (3)

- Supervised learning of a document-label assignment function
  - Many systems partly rely on machine learning (Autonomy, Microsoft, Yahoo!, …)
    - k-Nearest Neighbors (simple, powerful)
    - Naive Bayes (simple, common method)
    - Support-vector machines (new, more powerful)
    - … plus many other methods
    - No free lunch: requires hand-classified training data
    - But data can be built up (and refined) by amateurs
- Many commercial systems use a mixture of methods

Categorization/Classification

- Given:
  - A description of an instance, $d \in X$
    - X is the *instance space*.
      - Issue: how to represent text documents.
      - Usually some type of high-dimensional space
  - A fixed set of classes:

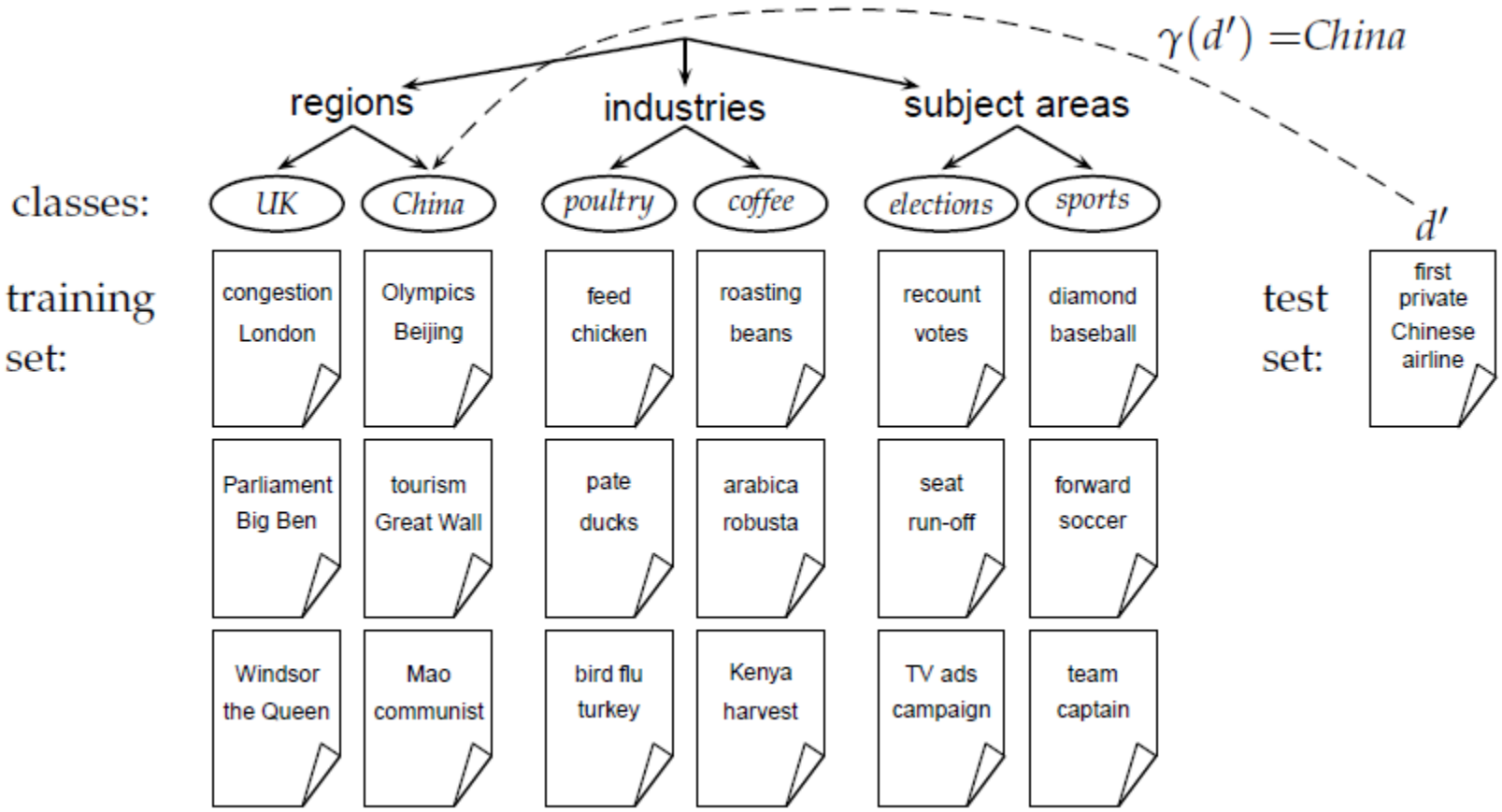
    $C = \{c_1, c_2, \ldots, c_J\}$
- Determine:
  - The category of d: $\gamma(d) \in C$, where $\gamma(d)$ is a *classification function* whose domain is X and whose range is C.
    - We want to know how to build classification functions ("classifiers").

Supervised Classification

- Given:
  - A description of an instance, $d \in X$
    - X is the *instance space*.
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set D of labeled documents with each labeled document $\langle d,c \rangle \in X \times C$
- Determine:
  - A learning method or algorithm which will enable us to learn a classifier $\gamma: X \rightarrow C$
  - For a test document d, we assign it the class $\gamma(d) \in C$

Document Classification



Naive Bayes

Bayes' Rule for text classification

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

$$P(c,d) = P(c \mid d)P(d) = P(d \mid c)P(c)$$

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

Naive Bayes Classifiers

Task: Classify a new instance *d* based on a tuple of attribute values into one of the classes $c_j \in C$

$$d = < x_1, x_2, \ldots, x_n >$$

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

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

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

MAP is "maximum a posteriori" = most likely class

Naïve Bayes Classifier: Naïve Bayes Assumption

- $P(c_j)$
  - Can be estimated from the frequency of classes in the training examples.
- $P(x_1, x_2, \ldots, x_n | c_j)$

Naïve Bayes Conditional Independence Assumption:

- Assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(x_i|c_j)$.

Example

- p(well)=0.9, p(cold)=0.05, p(allergy)=0.05
  - p(sneeze|well)=0.1
  - p(sneeze|cold)=0.9
  - p(sneeze|allergy)=0.9
  - p(cough|well)=0.1
  - p(cough|cold)=0.8
  - p(cough|allergy)=0.7
  - p(fever|well)=0.01
  - p(fever|cold)=0.7
  - p(fever|allergy)=0.4

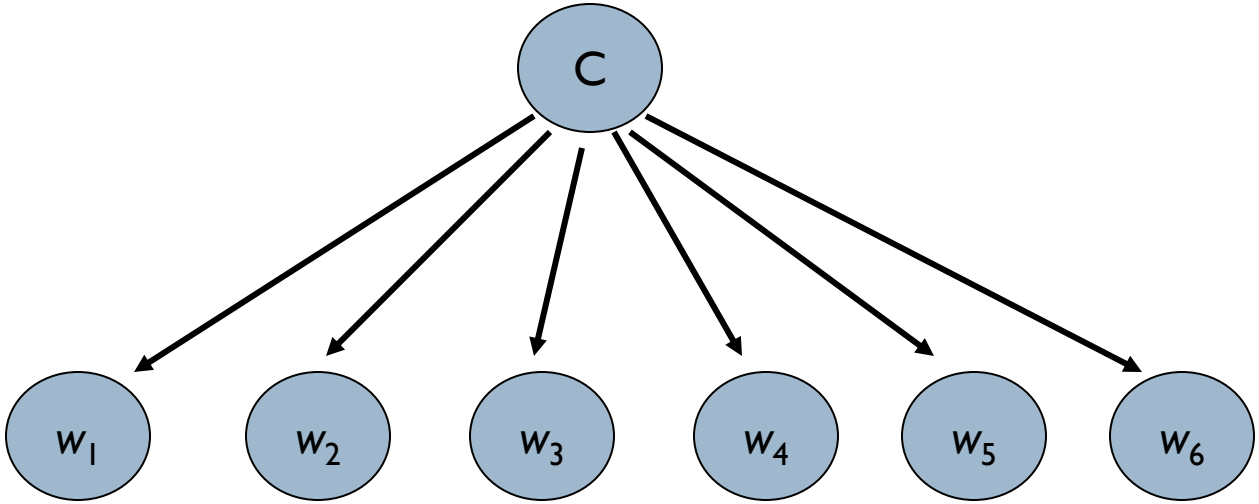
Example from Ray Mooney

Example (cont'd)

- Features: sneeze, cough, no fever
- P(well|e)=(.9) * (.1)(.1)(.99) / p(e)=0.0089/p(e)
- P(cold|e)=(.05) * (.9)(.8)(.3) / p(e)=0.01/p(e)
- P(allergy|e)=(.05) * (.9)(.7)(.6) / p(e)=0.019/p(e)
- P(e) = 0.0089+0.01+0.019=0.379
- P(well|e)=.23
- P(cold|e)=.26
- P(allergy|e)=.50

Example from Ray Mooney

Naïve Bayes via a class conditional language model = multinomial NB



- Effectively, the probability of each class is done as a class-specific unigram language model

Using Multinomial Naive Bayes Classifiers to Classify Text: Basic method

- Attributes are text positions, values are words.

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

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

- Still too many possibilities
- Assume that classification is *independent* of the positions of the words
  - Use same parameters for each position
  - Result is bag of words model (over *tokens* not *types*)

Naive Bayes: 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 $Text_j$
      - $$P(x_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Naive Bayes: Classifying

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

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

Underflow Prevention: using logs

- Multiplying lots of probabilities, which are between 0 and 1 by definition, can result in floating-point underflow.
- Since log(*xy*) = log(*x*) + log(*y*), it is better to perform all computations by summing logs of probabilities rather than multiplying probabilities.
- Class with highest final un-normalized log probability score is still the most probable.

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

Naive Bayes Classifier

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

- Simple interpretation: Each conditional parameter log $P(x_i|c_j)$ is a weight that indicates how good an indicator x_i is for c_j.
- The prior log $P(c_j)$ is a weight that indicates the relative frequency of c_j.
- The sum is then a measure of how much evidence there is for the document being in the class.
- We select the class with the most evidence for it

Two Models

- Model 1: Multinomial = Class conditional unigram
  - One feature X_i for each word pos in document
    - feature's values are all words in dictionary
  - Value of X_i is the word in position i
  - Naïve Bayes assumption:
    - Given the document's topic, word in one position in the document tells us nothing about words in other positions
  - Second assumption:
    - Word appearance does not depend on position

$$P(X_i = w \mid c) = P(X_j = w \mid c)$$

Two Naive Bayes Models

- Model 2: Multivariate Bernoulli
  - One feature X_w for each word in dictionary
  - X_w = true in document d if w appears in d
  - Naive Bayes assumption:
    - Given the document's topic, appearance of one word in the document tells us nothing about chances that another word appears

Parameter estimation

- Multivariate Bernoulli model:
- $\hat{P}(X_w = t \mid c_j) =$ fraction of documents of topic c_j in which word w appears

- Multinomial model:

$$\hat{P}(X_i = w \mid c_j) = \frac{\text{fraction of times in which word } w \text{ appears among all words in documents of topic } c_j}{}$$

  - Can create a mega-document for topic j by concatenating all documents in this topic
  - Use frequency of w in mega-document

Example: Multinomial Naive Bayes

	docID	words in document	in $c = China$?
training set	1	Chinese Beijing Chinese	yes
	2	Chinese Chinese Shanghai	yes
	3	Chinese Macao	yes
	4	Tokyo Japan Chinese	no
test set	5	Chinese Chinese Chinese Tokyo Japan	?

$\hat{P}(c) = 3/4$ and $\hat{P}(\overline{c}) = 1/4$

$$\begin{aligned}
\hat{P}(\text{Chinese}|c) &= (5+1)/(8+6) = 6/14 = 3/7 \\
\hat{P}(\text{Tokyo}|c) = \hat{P}(\text{Japan}|c) &= (0+1)/(8+6) = 1/14 \\
\hat{P}(\text{Chinese}|\overline{c}) &= (1+1)/(3+6) = 2/9 \\
\hat{P}(\text{Tokyo}|\overline{c}) = \hat{P}(\text{Japan}|\overline{c}) &= (1+1)/(3+6) = 2/9
\end{aligned}$$

$$\begin{aligned}
\hat{P}(c|d_5) &\propto 3/4 \cdot (3/7)^3 \cdot 1/14 \cdot 1/14 \approx 0.0003. \\
\hat{P}(\overline{c}|d_5) &\propto 1/4 \cdot (2/9)^3 \cdot 2/9 \cdot 2/9 \approx 0.0001.
\end{aligned}$$

Example: Multivariate Bernoulli

	docID	words in document	in $c = China$?
training set	1	Chinese Beijing Chinese	yes
	2	Chinese Chinese Shanghai	yes
	3	Chinese Macao	yes
	4	Tokyo Japan Chinese	no
test set	5	Chinese Chinese Chinese Tokyo Japan	?

$\hat{P}(c) = 3/4$ and $\hat{P}(\overline{c}) = 1/4$

$$
\begin{aligned}
\hat{P}(\text{Chinese}|c) &= (3+1)/(3+2) = 4/5 \\
\hat{P}(\text{Japan}|c) = \hat{P}(\text{Tokyo}|c) &= (0+1)/(3+2) = 1/5 \\
\hat{P}(\text{Beijing}|c) = \hat{P}(\text{Macao}|c) = \hat{P}(\text{Shanghai}|c) &= (1+1)/(3+2) = 2/5 \\
\hat{P}(\text{Chinese}|\overline{c}) &= (1+1)/(1+2) = 2/3 \\
\hat{P}(\text{Japan}|\overline{c}) = \hat{P}(\text{Tokyo}|\overline{c}) &= (1+1)/(1+2) = 2/3 \\
\hat{P}(\text{Beijing}|\overline{c}) = \hat{P}(\text{Macao}|\overline{c}) = \hat{P}(\text{Shanghai}|\overline{c}) &= (0+1)/(1+2) = 1/3
\end{aligned}
$$

$$
\begin{aligned}
\hat{P}(c|d_5) &\propto \hat{P}(c) \cdot \hat{P}(\text{Chinese}|c) \cdot \hat{P}(\text{Japan}|c) \cdot \hat{P}(\text{Tokyo}|c) \\
&\quad \cdot (1 - \hat{P}(\text{Beijing}|c)) \cdot (1 - \hat{P}(\text{Shanghai}|c)) \cdot (1 - \hat{P}(\text{Macao}|c)) \\
&= 3/4 \cdot 4/5 \cdot 1/5 \cdot 1/5 \cdot (1-2/5) \cdot (1-2/5) \cdot (1-2/5) \\
&\approx 0.005
\end{aligned}
$$

$$
\begin{aligned}
\hat{P}(\overline{c}|d_5) &\propto 1/4 \cdot 2/3 \cdot 2/3 \cdot 2/3 \cdot (1-1/3) \cdot (1-1/3) \cdot (1-1/3) \\
&\approx 0.022
\end{aligned}
$$

Feature Selection: Why?

- Text collections have a large number of features
  - 10,000 – 1,000,000 unique words … and more
- May make using a particular classifier feasible
  - Some classifiers can't deal with 100,000 of features
- Reduces training time
  - Training time for some methods is quadratic or worse in the number of features
- Can improve generalization (performance)
  - Eliminates noise features
  - Avoids overfitting

Feature selection: The X^2 test

- For a term *t*:

		I_t	
		0	1
C	0	k_{00}	k_{01}
	1	k_{10}	k_{11}

- C=class, i_t = feature
- Testing for independence:
  $P(C=0,I_t=0)$ should be equal to $P(C=0)\ P(I_t=0)$
  - $P(C=0) = (k_{00}+k_{01})/n$
  - $P(C=1) = 1-P(C=0) = (k_{10}+k_{11})/n$
  - $P(I_t=0) = (k_{00}+K_{10})/n$
  - $P(I_t=1) = 1-P(I_t=0) = (k_{01}+k_{11})/n$

Feature selection: The X^2 test

$$X^2 = \frac{n(k_{11}k_{00} - k_{10}k_{01})^2}{(k_{11}+k_{10})(k_{01}+k_{00})(k_{11}+k_{01})(k_{10}+k_{00})}$$

- High values of X^2 indicate lower belief in independence.
- In practice, compute X^2 for all words and pick the top k among them.

Feature selection via Mutual Information

- In training set, choose *k* words which best discriminate (give most info on) the categories.
- The Mutual Information between a word, class is:

$$I(w,c) = \sum_{e_w \in \{0,1\}} \sum_{e_c \in \{0,1\}} p(e_w, e_c) \log \frac{p(e_w, e_c)}{p(e_w)p(e_c)}$$

e_w = 1 if term occurs in document, 0 otherwise

e_c = 1 if document in class c, 0 otherwise

Measures how much information presence/absence of a term contributes to the correct classification decision on the class.

I(w,c) is 0 if term's distribution is the same in the class and in the collection as a whole.

I(w,c) is maximum when the term only occurs in the documents belonging to class c.

Feature selection via MI (contd.)

- For each category we build a list of *k* most discriminating terms.
- For example (on 20 Newsgroups):
  - ***sci.electronics:*** circuit, voltage, amp, ground, copy, battery, electronics, cooling, …
  - ***rec.autos:*** car, cars, engine, ford, dealer, mustang, oil, collision, autos, tires, toyota, …

Vector Space Classification

Recall: Vector Space Representation

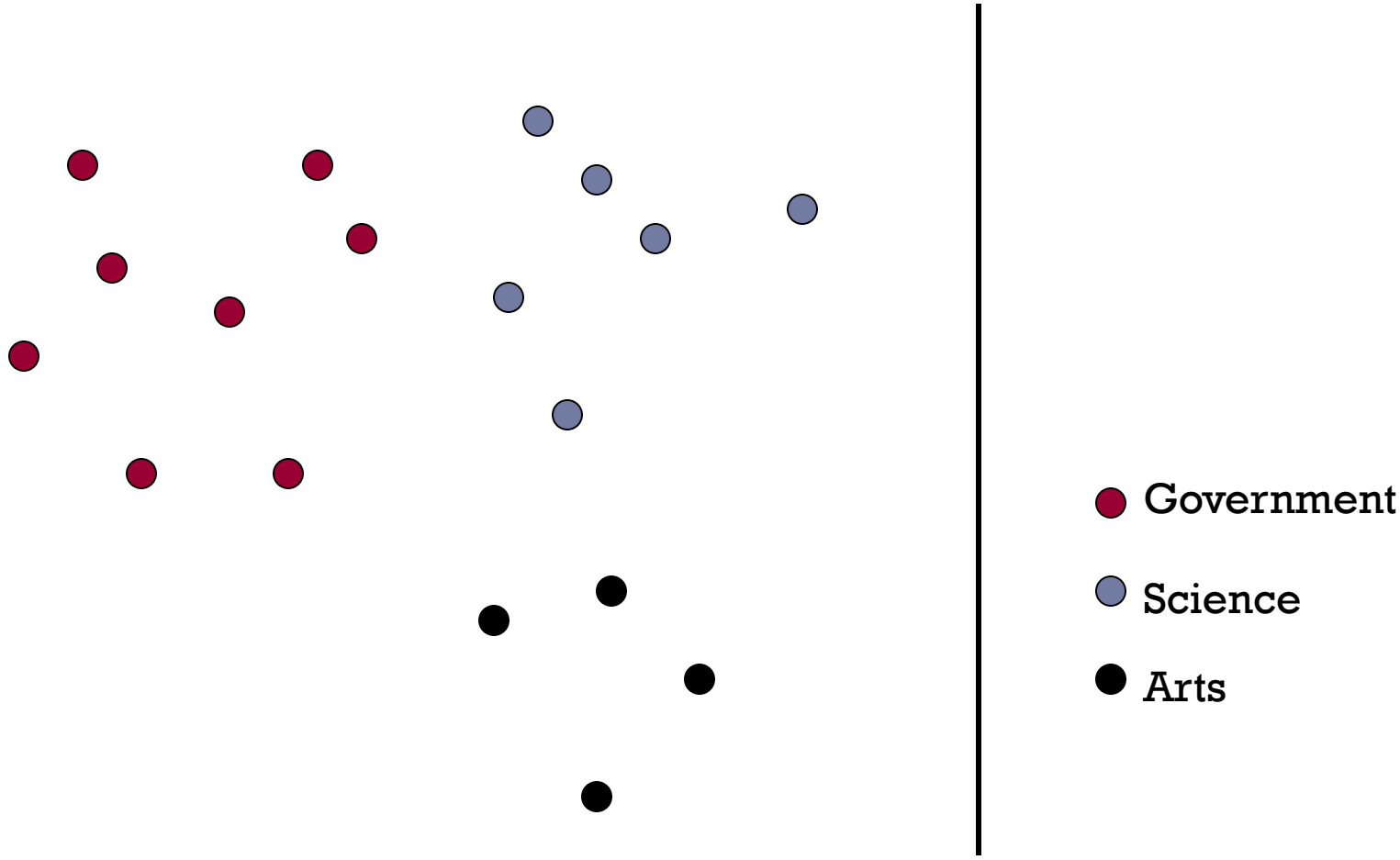
- Each document is a vector, one component for each term (= word).
- Normally normalize vectors to unit length.
- High-dimensional vector space:
  - Terms are axes
  - 10,000+ dimensions, or even 100,000+
  - Docs are vectors in this space

- How can we do classification in this space?

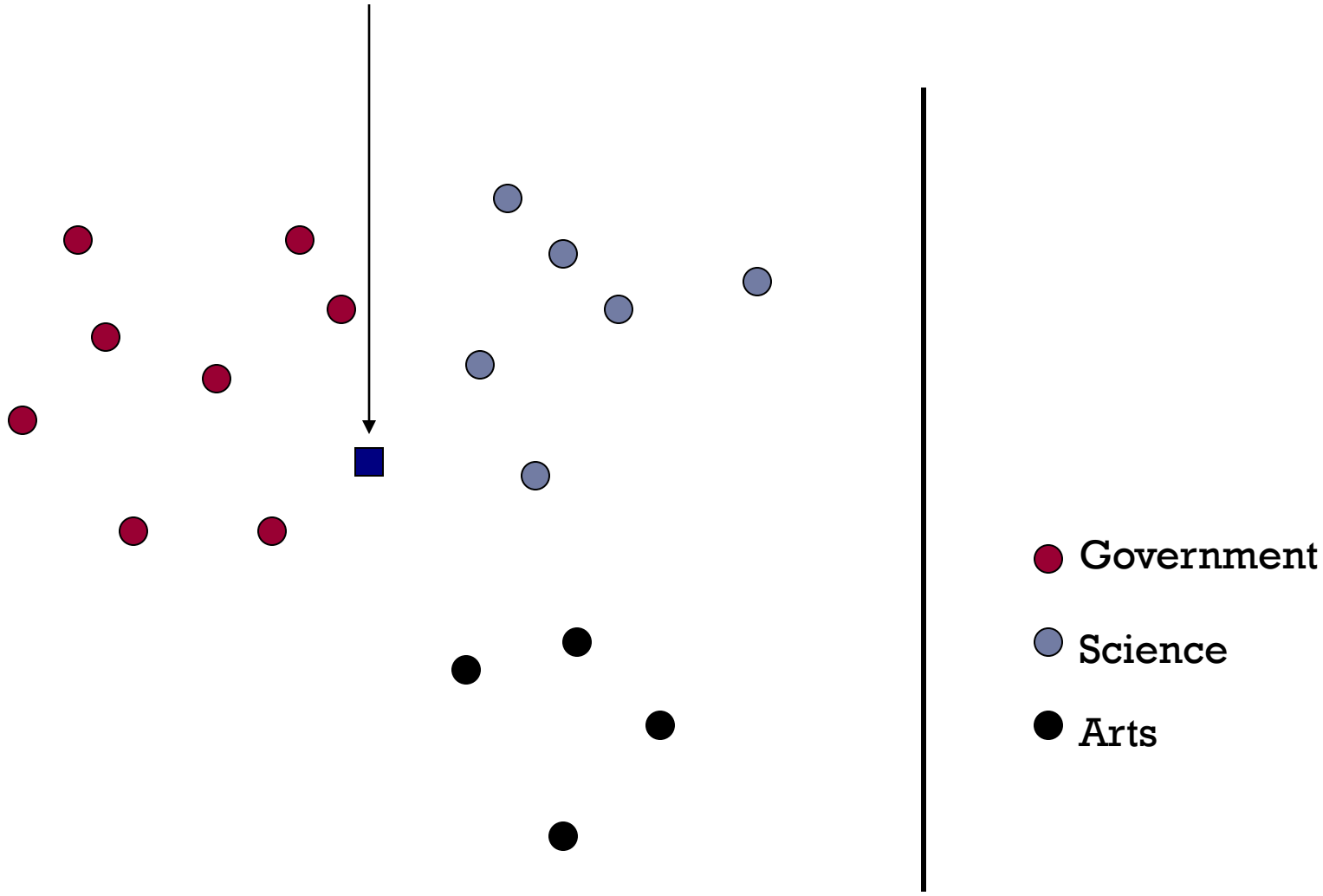
Classification Using Vector Spaces

- As before, the training set is a set of documents, each labeled with its class (e.g., topic)
- In vector space classification, this set corresponds to a labeled set of points (or, equivalently, vectors) in the vector space
- Premise 1: Documents in the same class form a contiguous region of space
- Premise 2: Documents from different classes don't overlap (much)
- We define surfaces to delineate classes in the space

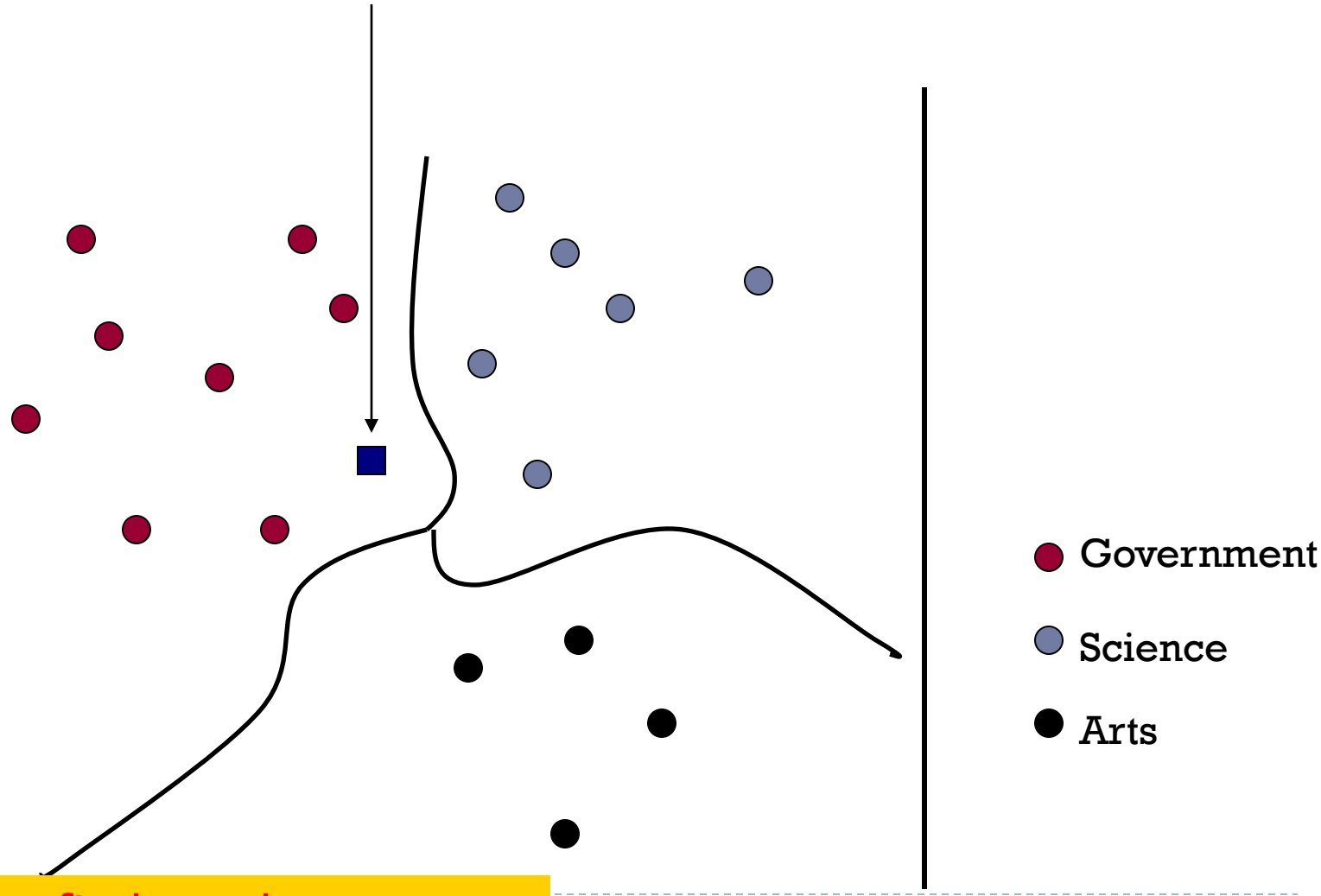
Documents in a Vector Space



Test Document of what class?



Test Document = Government



We want to find good separators

Rocchio

Definition of centroid

$$\vec{\mu}(c) = \frac{1}{|D_c|} \sum_{d \in D_c} \vec{v}(d)$$

- Where D_c is the set of all documents that belong to class c and $v(d)$ is the vector space representation of d.

Rocchio classification

- Rocchio forms a simple representation for each class: the centroid/prototype
- Classification is based on similarity to / distance from the prototype/centroid
- It is little used outside text classification
  - It has been used quite effectively for text classification
  - But in general worse than Naïve Bayes
- Again, cheap to train and test documents

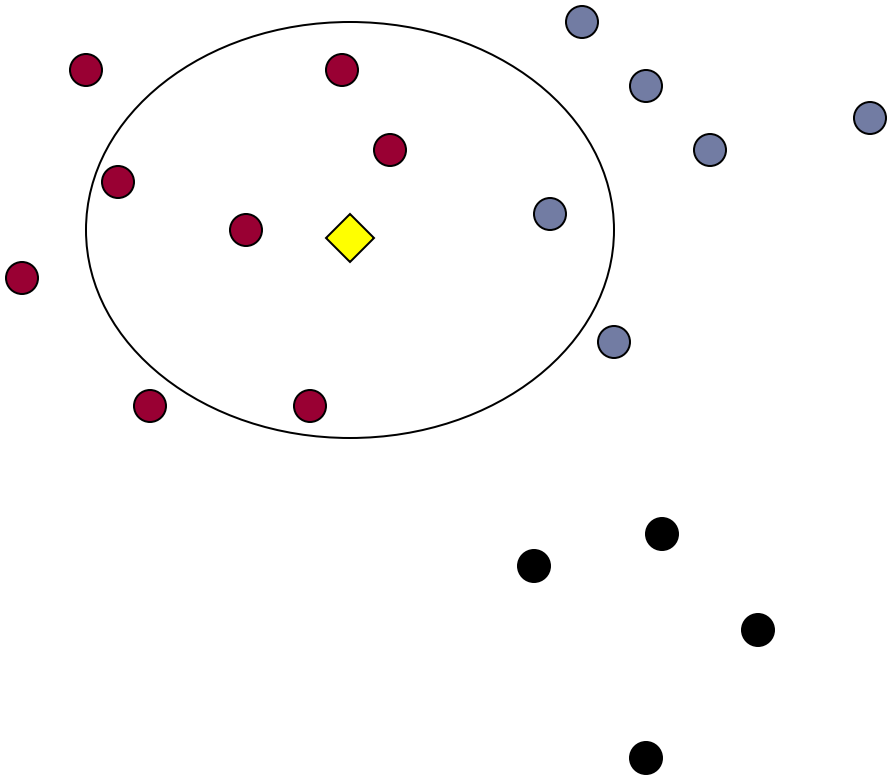
k Nearest Neighbor

k Nearest Neighbor Classification

- kNN = k Nearest Neighbor

- To classify a document *d* into class c:
- Define *k*-neighborhood N as *k* nearest neighbors of *d*
- Count number of documents i in N that belong to c
- Estimate P(c|*d*) as i/k
- Choose as class argmax_c P(c|*d*) [= majority class]

Example: k=6 (6NN)



P(science|◆)?

- Government
- Science
- Arts

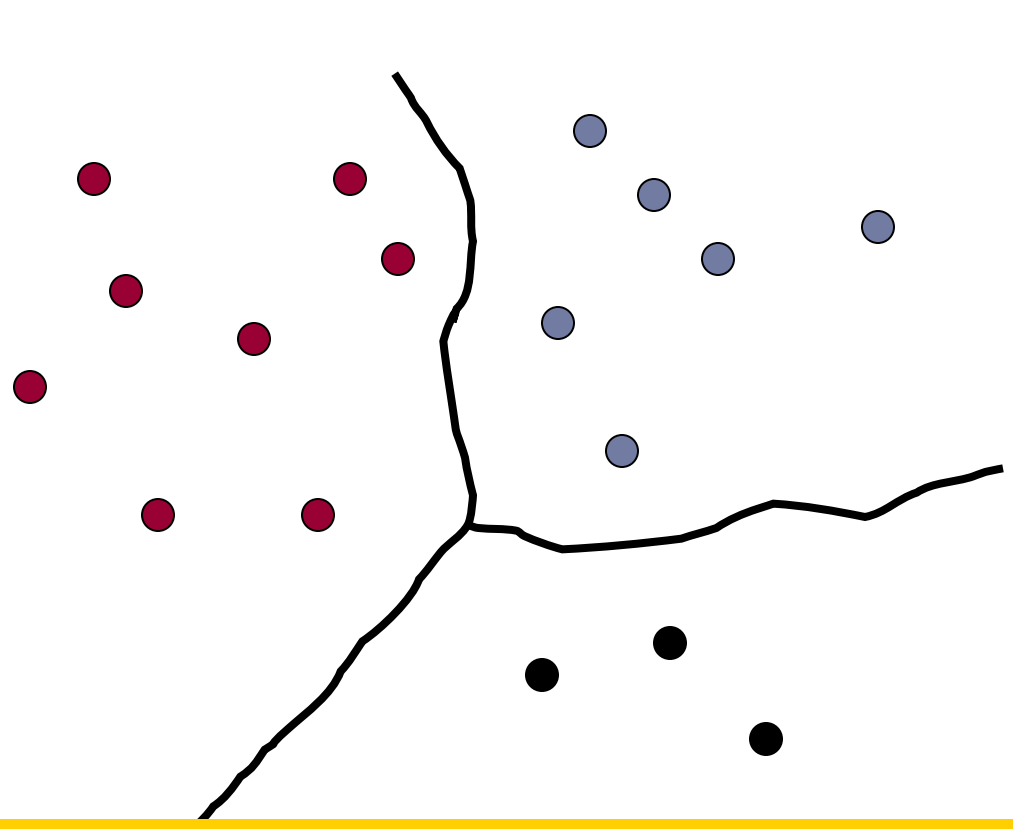
Nearest-Neighbor Learning Algorithm

- Learning is just storing the representations of the training examples in *D*.
- Testing instance *x (under 1NN)*:
  - Compute similarity between *x* and all examples in *D*.
  - Assign *x* the category of the most similar example in *D*.
- Does not explicitly compute a generalization or category prototypes.
- Also called:
  - Case-based learning
  - Memory-based learning
  - Lazy learning

k Nearest Neighbor

- Using only the closest example (1NN) to determine the class is subject to errors due to:
  - A single atypical example.
  - Noise (i.e., an error) in the category label of a single training example.
- More robust alternative is to find the *k* most-similar examples and return the majority category of these *k* examples.
- Value of *k* is typically odd to avoid ties; 3 and 5 are most common.

kNN decision boundaries



Boundaries are in principle arbitrary

Government

Science

Arts

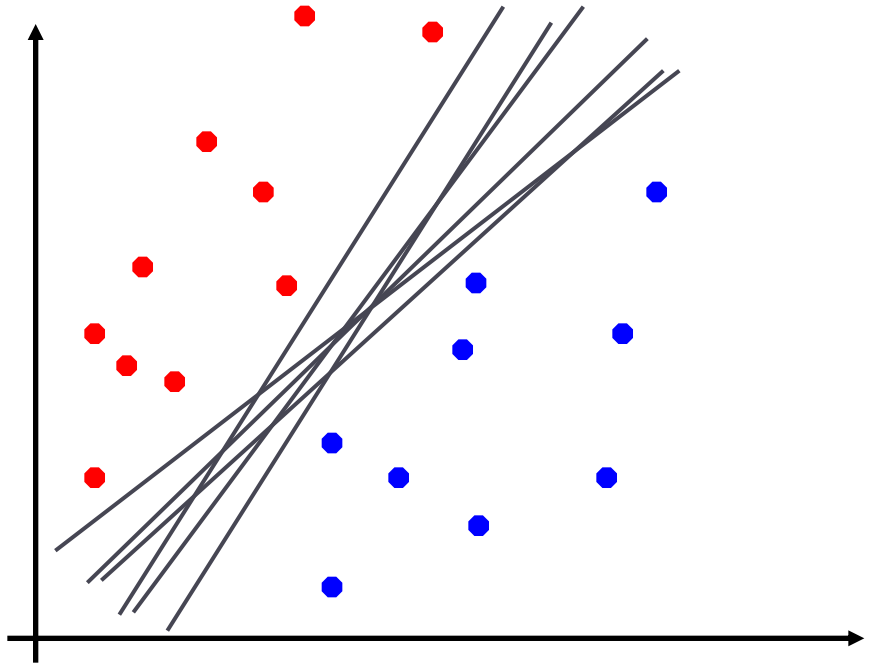
kNN gives locally defined decision boundaries between classes – far away points do not influence each classification decision (unlike in Naïve Bayes, Rocchio, etc.)

Similarity Metrics

- Nearest neighbor method depends on a similarity (or distance) metric.
- For text, cosine similarity of tf.idf weighted vectors is typically most effective.

Support Vector Machines

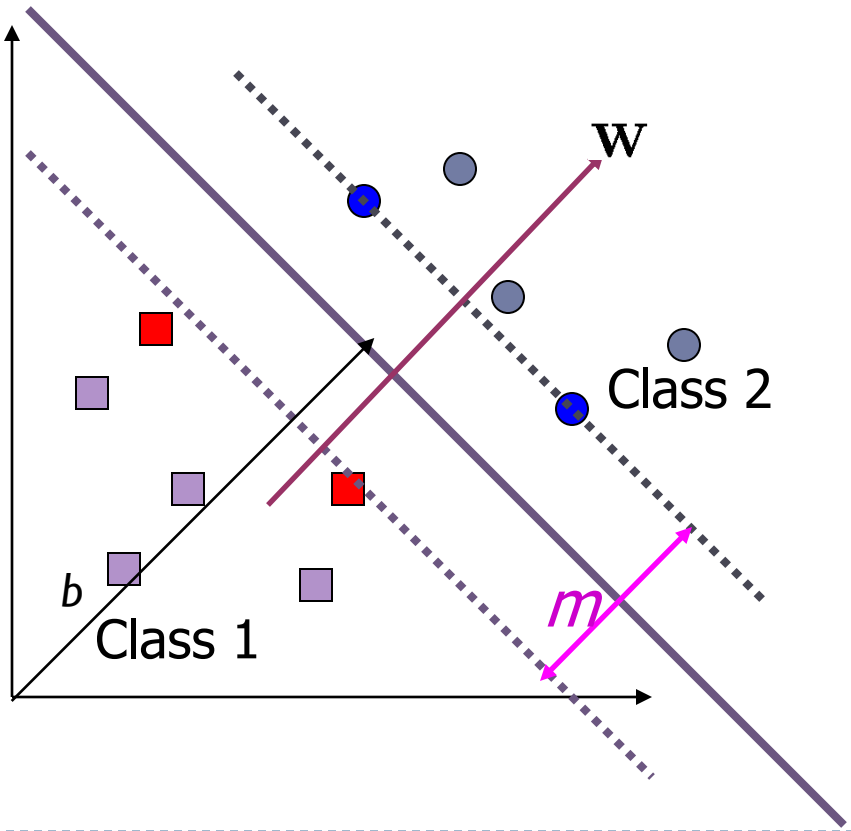
SVM - Decision Boundary



- Linearly separable case
- Infinitely many boundaries
- Which one is the best?

SVM - Best Boundary

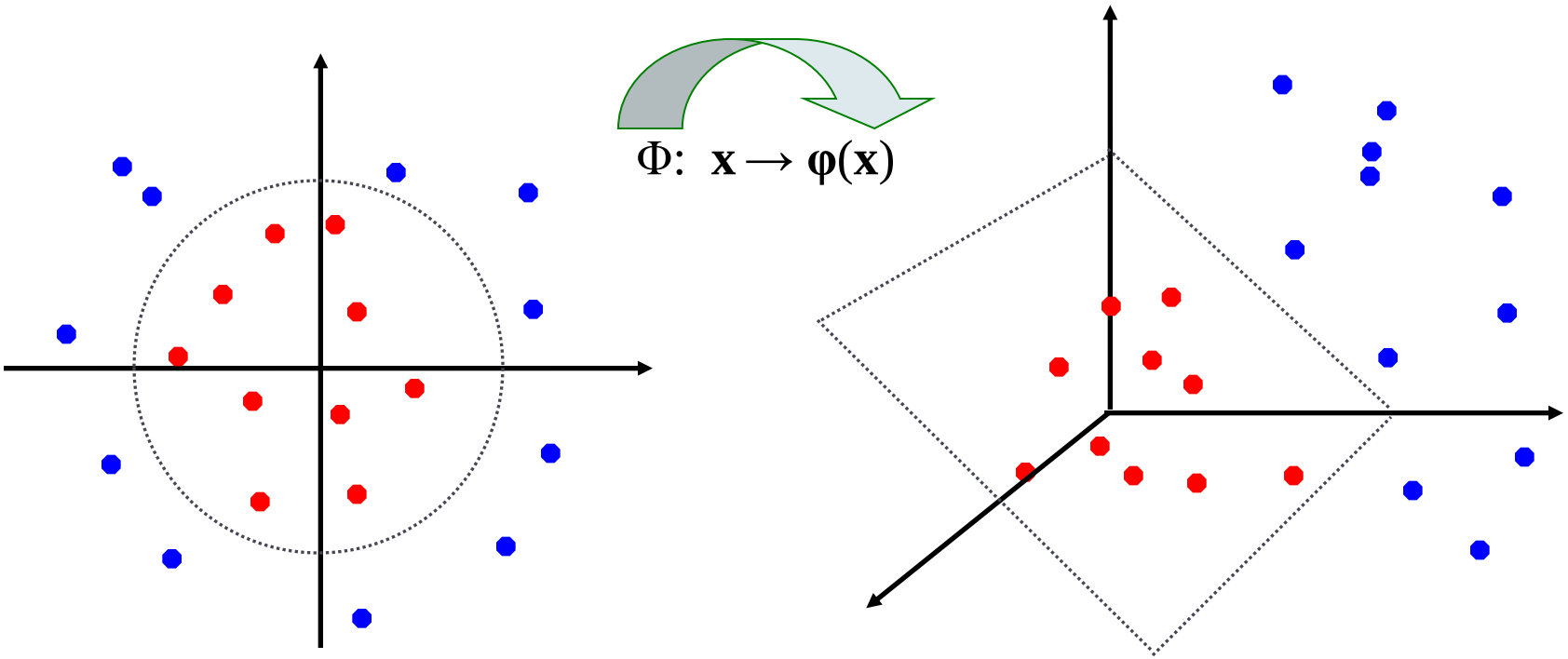
- The decision boundary should be as far away from the data of both classes as possible
  - We should maximize the margin, *m*



Implementations:
- Quadratic optimization
- Use toolkit (e.g., libSVM, Thorsten Joachims's svmlight)

Non-Linear SVM

- Idea: original feature space is mapped to higher-dimensional feature space where the training set is separable:



Example

$\Phi : \Re^2 \rightarrow \Re^3$ (mapping to a higher-dimensional space)

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) \equiv (x_1^2, \sqrt{2} x_1 x_2, x_2^2)$$

The Kernel Trick

- The linear classifier relies on inner product between vectors $K(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{x}_i^T\mathbf{x}_j$
- After transformation Φ: $\mathbf{x} \rightarrow \phi(\mathbf{x})$, the inner product becomes:

  $K(\mathbf{x}_i,\mathbf{x}_j)= \boldsymbol{\phi}(\mathbf{x}_i)^T\boldsymbol{\phi}(\mathbf{x}_j)$

- A *kernel function* is a function that is equivalent to an inner product in some feature space.
- A kernel function *implicitly* maps data to a high-dimensional space without the need to compute each $\boldsymbol{\phi}(\mathbf{x})$ explicitly.

The kernel trick

$$< \Phi(x), \Phi(x') >= (x_1^2, \sqrt{2}x_1x_2, x_2^2)(x'_1{}^2, \sqrt{2}x'_1x'_2, x'_2{}^2)^T = < x, x' >^2 = k(x, x')$$

Polynomial kernel: $k(x, x') = (< x, x' > + c)^d$

Sigmoid kernel: $k(x, x') = \tanh(k < x, x' > + \theta)$

RBF kernel: $k(x, x') = \exp(-\|x - x'\|^2 / (2\sigma^2))$

Many other kernels are useful for IR:
e.g., string kernels, tree kernels, etc.

Evaluation: Classic Reuters-21578 Data Set

- Most used data set
- 21578 documents
- 9603 training, 3299 test articles (ModApte/Lewis split)
- 118 categories
  - An article can be in more than one category
  - Learn 118 binary category distinctions
- Average document: about 90 types, 200 tokens
- Average number of classes assigned
  - 1.24 for docs with at least one category
- Only about 10 out of 118 categories are large

Common categories (#train, #test)

- Earn (2877, 1087)
- Acquisitions (1650, 179)
- Money-fx (538, 179)
- Grain (433, 149)
- Crude (389, 189)
- Trade (369,119)
- Interest (347, 131)
- Ship (197, 89)
- Wheat (212, 71)
- Corn (182, 56)

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

&#3;</BODY></TEXT></REUTERS>

Per class evaluation measures

- Recall: Fraction of docs in class *i* classified correctly:
- Precision: Fraction of docs assigned class *i* that are actually about class *i*:
- Accuracy: (1 - error rate) Fraction of docs classified correctly:

Micro- vs. Macro-Averaging

- If we have more than one class, how do we combine multiple performance measures into one quantity?
- Macroaveraging: Compute performance for each class, then average.
- Microaveraging: Collect decisions for all classes, compute contingency table, evaluate.

Evaluating Categorization

- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
  - Sometimes use cross-validation (averaging results over multiple training and test splits of the overall data)
- It's easy to get good performance on a test set that was available to the learner during training (e.g., just memorize the test set).
- Measures: precision, recall, F1, classification accuracy
- *Classification accuracy*: *c*/*n* where *n* is the total number of test instances and *c* is the number of test instances correctly classified by the system.
  - Adequate if one class per document
  - Otherwise F measure for each class

Micro- vs. Macro-Averaging: Example

Class 1

	Truth: yes	Truth: no
Classifier: yes	10	10
Classifier: no	10	970

Class 2

	Truth: yes	Truth: no
Classifier: yes	90	10
Classifier: no	10	890

Micro Ave. Table

	Truth: yes	Truth: no
Classifier: yes	100	20
Classifier: no	20	1860

- Macroaveraged precision: (0.5 + 0.9)/2 = 0.7
- Microaveraged precision: 100/120 = .83

- Microaveraged score is dominated by score on common classes

(a)

	NB	Rocchio	kNN	SVM
micro-avg-L (90 classes)	80	85	86	89
macro-avg (90 classes)	47	59	60	60

(b)

	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

Evaluation measure: F_1

Conclusion

- SVMs are widely considered to be the best method for text classification (look at papers by Sebastiani, Christianini, Joachims), e.g. 86% accuracy on Reuters.
- NB also good in many circumstances

Resources

- *Introduction to Information Retrieval*, chapters 13, 14, 15.
- Some slides were adapted from
  - Prof. Dragomir Radev's lectures at the University of Michigan:
    - http://clair.si.umich.edu/~radev/teaching.html
  - the book's companion website:
    - http://nlp.stanford.edu/IR-book/information-retrieval-book.html

- Weka: A data mining software package that includes an implementation of many Machine Learning algorithms
- Reuters-21578 – the most famous text classification evaluation set
  - Still widely used (but now it's too small for realistic experiments – you should use Reuters RCV1)
- SVM Implementations: SVMlight, LibSVM, etc.