Vector Space Classification in IR

Reading:
Textbook, Chapter 14

April 22, 2010

Some slides adapted from
http://www-nlp.stanford.edu/IR-book/newslides.html

1

The rest of text classification

– Vector space methods for text classification

– Support Vector Machines

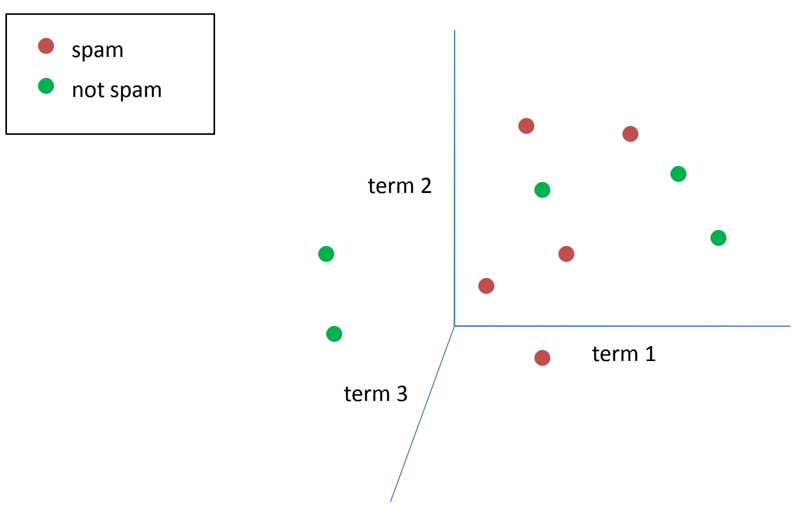
– Text-specific issues in classification

2

Recall: Vector Space Representation

- Each document is a vector, one component for each term (= word).
- 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?

3

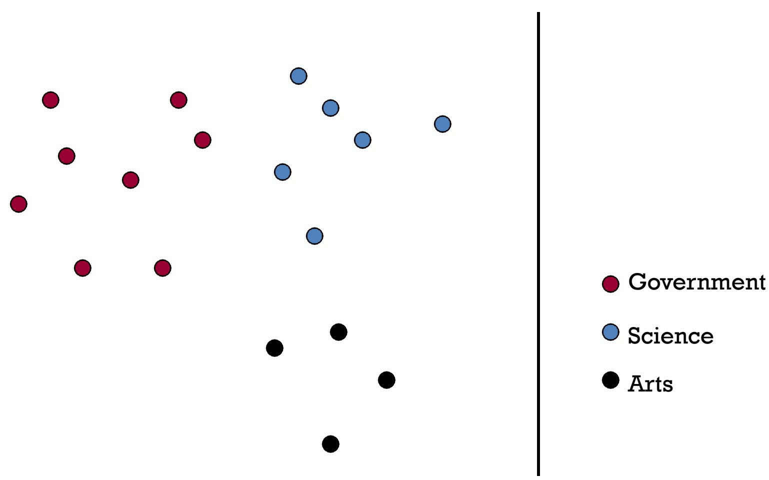


4

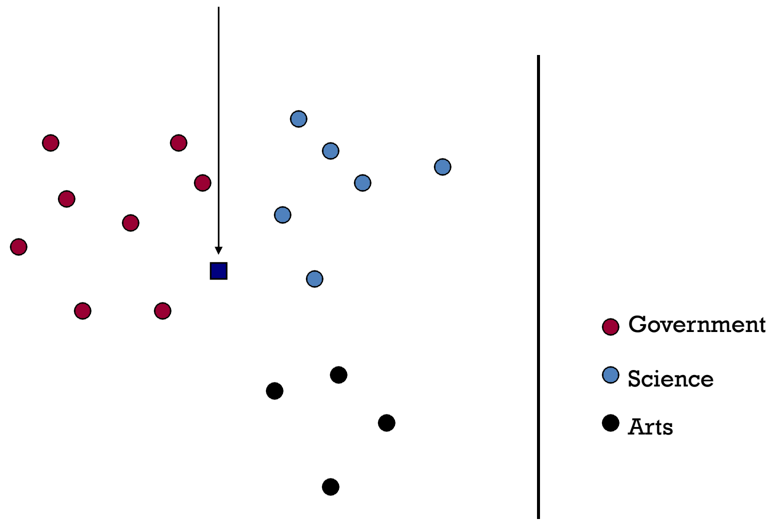
Classification Using Vector Spaces

- 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

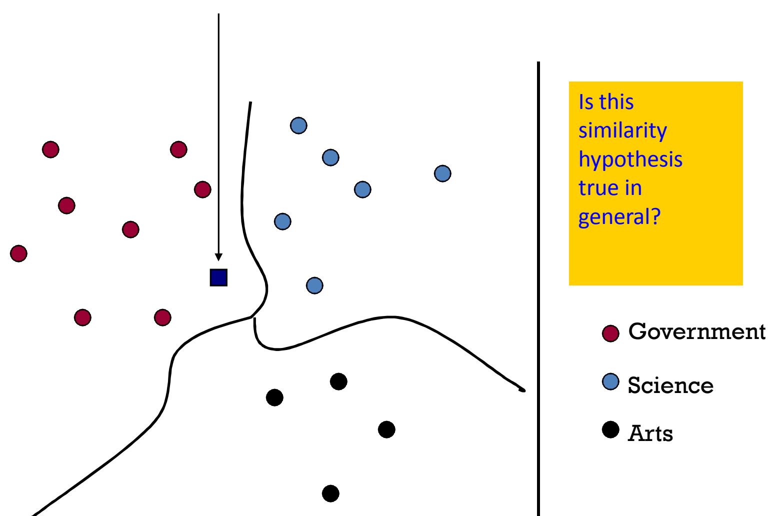


Test Document of what class?



7

Test Document = Government



Our main topic today is how to find good separators

8

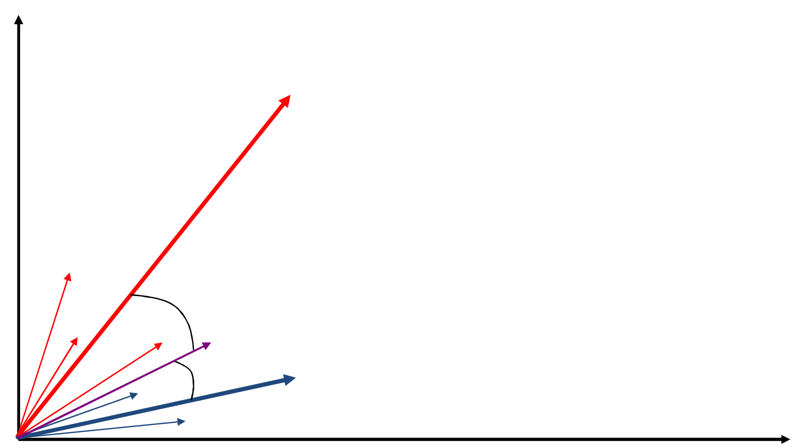
Using Rocchio for text classification

- Use standard tf-idf weighted vectors to represent text documents
- For training documents in each category, compute a prototype vector by averaging the vectors of the training documents in the category.
  - Prototype = centroid of members of class
- Assign test documents to the category with the closest prototype vector based on cosine similarity.

$$\text{Assign } d \text{ to class } c = \arg\max_{c'} \cos(\vec{\mu}(c'), \vec{v}(d))$$

9

Illustration of Rocchio Text Categorization



10

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.
- *Note that centroid will in general* not *be a unit vector even when the inputs are unit vectors.*

11

Rocchio Properties

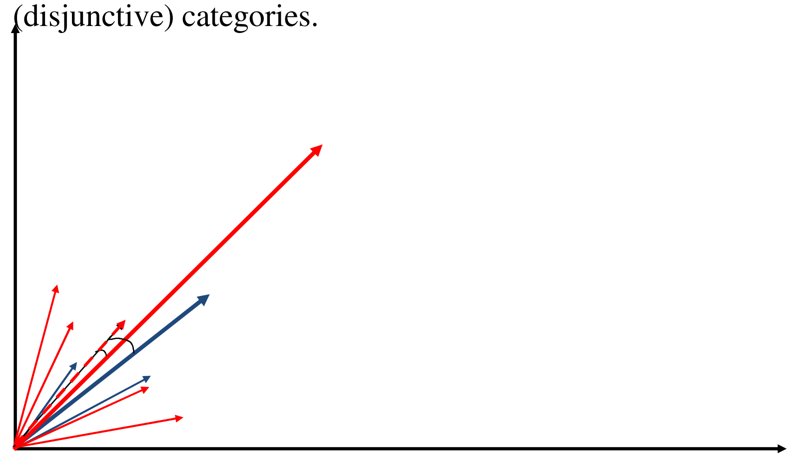
- Forms a simple generalization of the examples in each class (a *prototype*).
- Prototype vector does not need to be averaged or otherwise normalized for length since cosine similarity is insensitive to vector length.
- Classification is based on similarity to class prototypes.
- Does not guarantee classifications are consistent with the given training data.

Why not?

12

Rocchio Anomaly

- Prototype models have problems with polymorphic (disjunctive) categories.



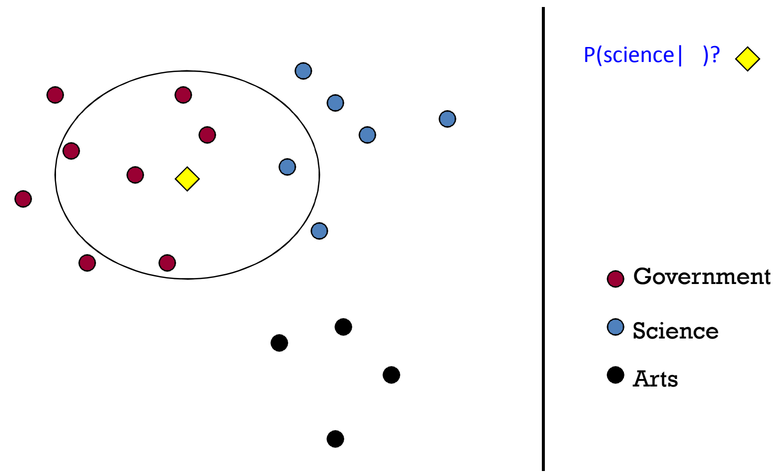
13

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]

14

Example: k=6 (6NN)



Nearest-Neighbor Learning Algorithm

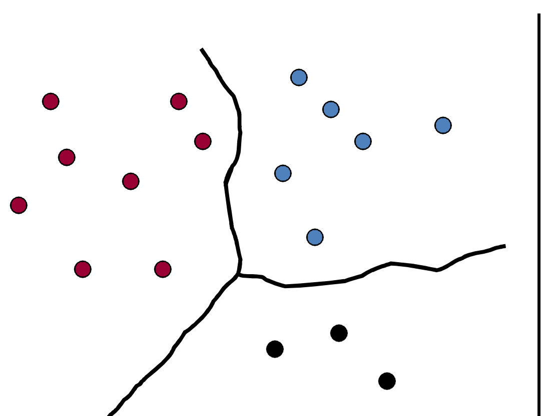
- Learning is just storing the representations of the training examples in *D*.
- Testing instance *x (under 1NN)*:
  - Compute similarity between *x* and all examples in *D*.
  - Assign *x* the category of the most similar example in *D*.
- Does not explicitly compute a generalization or category prototypes.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

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.

17

kNN decision boundaries



Boundaries are in principle arbitrary surfaces – but usually polyhedra

- 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.)

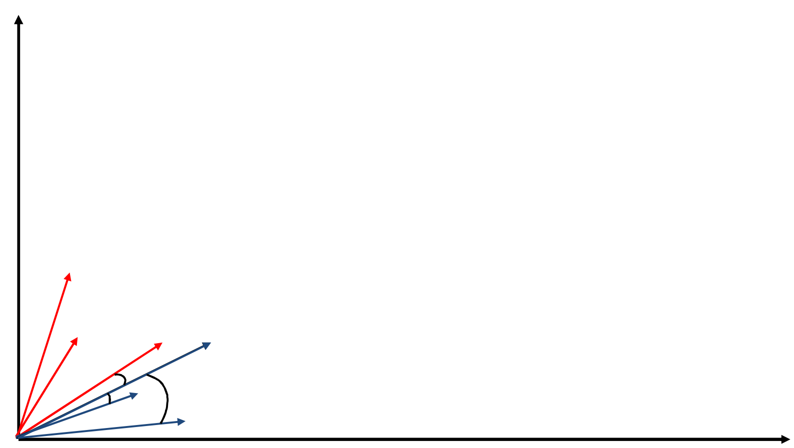
18

Similarity Metrics

- Nearest neighbor method depends on a similarity (or distance) metric.
- Simplest for continuous *m*-dimensional instance space is *Euclidean distance*.
- Simplest for *m*-dimensional binary instance space is *Hamming distance* (number of feature values that differ).
- For text, cosine similarity of tf.idf weighted vectors is typically most effective.

19

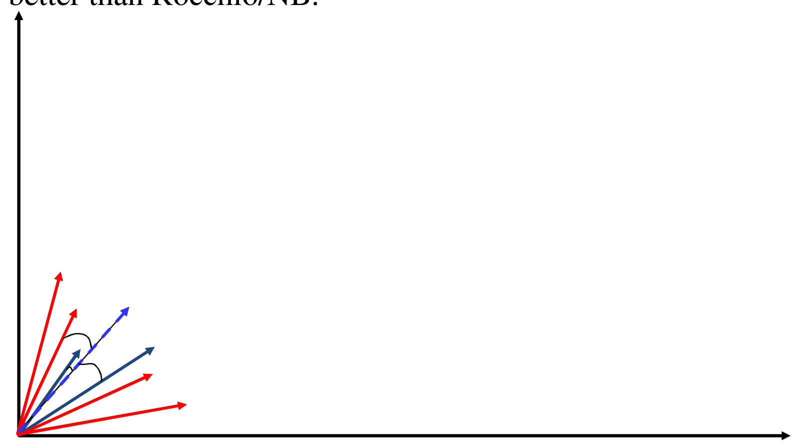
Illustration of 3 Nearest Neighbor for Text Vector Space



20

3 Nearest Neighbor vs. Rocchio

- Nearest Neighbor tends to handle polymorphic categories better than Rocchio/NB.



21

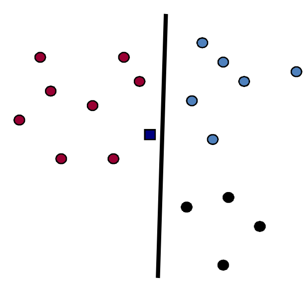
Linear classifiers and binary and multiclass classification

- Consider 2 class problems
  - E.g., spam vs. not spam
- How do we define (and find) the separating surface?
- How do we decide which region a test doc is in?

22

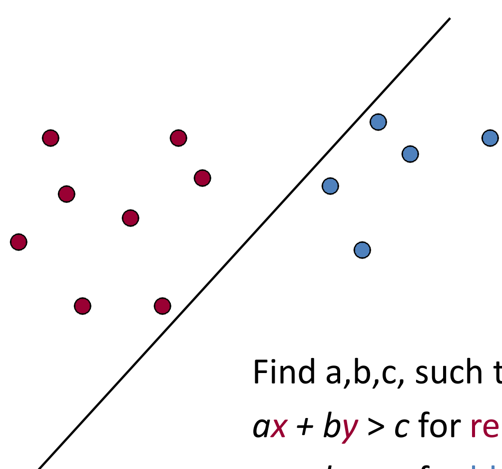
Separation by Hyperplanes

- A strong high-bias assumption is *linear separability*:
  - in 2 dimensions, can separate classes by a line
    - separator can be expressed as $ax + by = c$
  - in higher dimensions, need hyperplanes



23

Linear programming / Perceptron



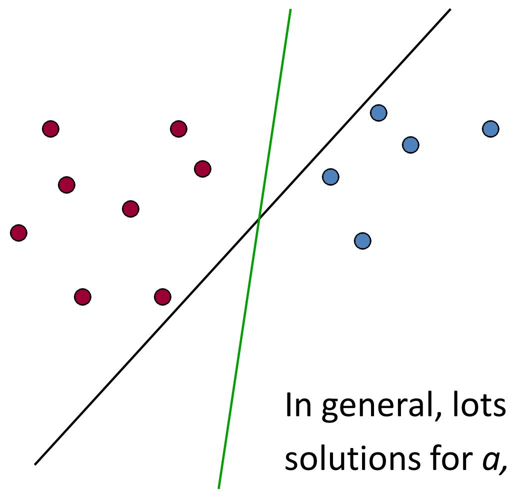
Find a,b,c, such that

$ax + by > c$ for red points

$ax + by < c$ for blue points.

24

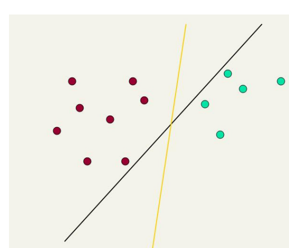
Which Hyperplane?



In general, lots of possible solutions for *a,b,c.*

Which Hyperplane?

- Lots of possible solutions for *a,b,c.*
- Most text-classification methods find an optimal separating hyperplane
- Which points should influence optimality?
  - All points
    - Linear/logistic regression
    - Naïve Bayes
  - Only “difficult points” close to decision boundary
    - Support vector machines



Linear Classifiers

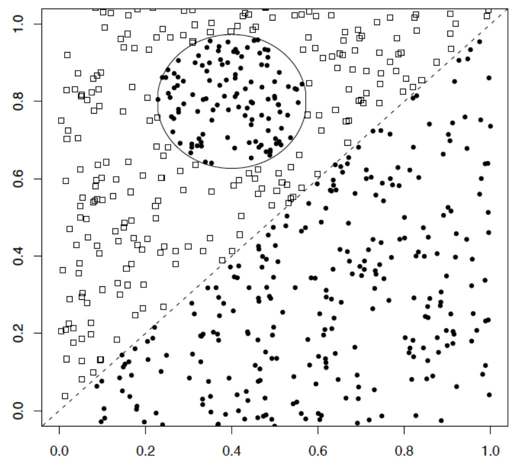
- Many common text classifiers are linear classifiers
  - Naïve Bayes
  - Perceptron
  - Rocchio
  - Logistic regression
  - Support vector machines (with linear kernel)
  - Linear regression with threshold

27

- Despite this similarity, noticeable performance differences
  - For separable problems, there is an infinite number of separating hyperplanes. Which one do you choose?
  - What to do for non-separable problems?
  - Different training methods pick different hyperplanes
- Classifiers more powerful than linear often don't perform better on text problems. Why?

28

A nonlinear problem



- A linear classifier like Naïve Bayes does badly on this task
- kNN will do very well (assuming enough training data)

29

More Than Two Classes

- Any-of or multivalue classification
  - Classes are independent of each other.
  - A document can belong to 0, 1, or >1 classes.
  - Decompose into *n* binary problems
  - Quite common for documents
- One-of or multinomial or polytomous classification
  - Classes are mutually exclusive.
  - Each document belongs to exactly one class
  - E.g., digit recognition is polytomous classification
    - Digits are mutually exclusive
  - "One against all" classification

30

Confusion Matrix

assigned class / true class	*money-fx*	*trade*	*interest*	*wheat*	*corn*	*grain*
money-fx	95	0	10	0	0	0
trade	1	1	90	0	1	0
interest	13	0	0	0	0	0
wheat	0	0	1	34	3	7
corn	1	0	2	13	26	5
grain	0	0	2	14	5	10

► **Table 14.5** A confusion matrix for Reuters-21578. For example, 14 documents from *grain* were incorrectly assigned to *wheat*. Adapted from Picca et al. (2006).

31

Bias, Variance, and Noise

- **Bias:**
  - Classifier cannot learn the correct hypothesis (no matter what training data is given), and so incorrect hypothesis *h* is learned. The **bias is the average error** of h over all possible training sets.
- **Variance:**
  - Training data is not representative enough of all data, so the learned classifier *h* varies from one training set to another.
- **Noise:**
  - Training data contains errors, so incorrect hypothesis *h* is learned.

32

Which classifier do I use for a given text classification problem?

- Is there a learning method that is optimal for all text classification problems?
- No, because there is a tradeoff between bias and variance.
- Factors to take into account:
  - How much training data is available?
  - How simple/complex is the problem? (linear vs. nonlinear decision boundary)
  - How noisy is the data?
  - How stable is the problem over time?
    - For an unstable problem, it's better to use a simple and robust classifier.

"A machine with too much capacity is like a botanist with a photographic memory who, when presented with a new tree, concludes that it is not a tree because it has a different number of leaves from anything she has seen before; a machine with too little capacity is like the botanist's lazy brother, who declares that if it's green, it's a tree. Neither can generalize well." (C. Burges, *A tutorial on support vector machines for pattern recognition).*