Text Classification
kNN and Linear Classifier

Reference: Introduction to Information Retrieval
by C. Manning, P. Raghavan, H. Schutze

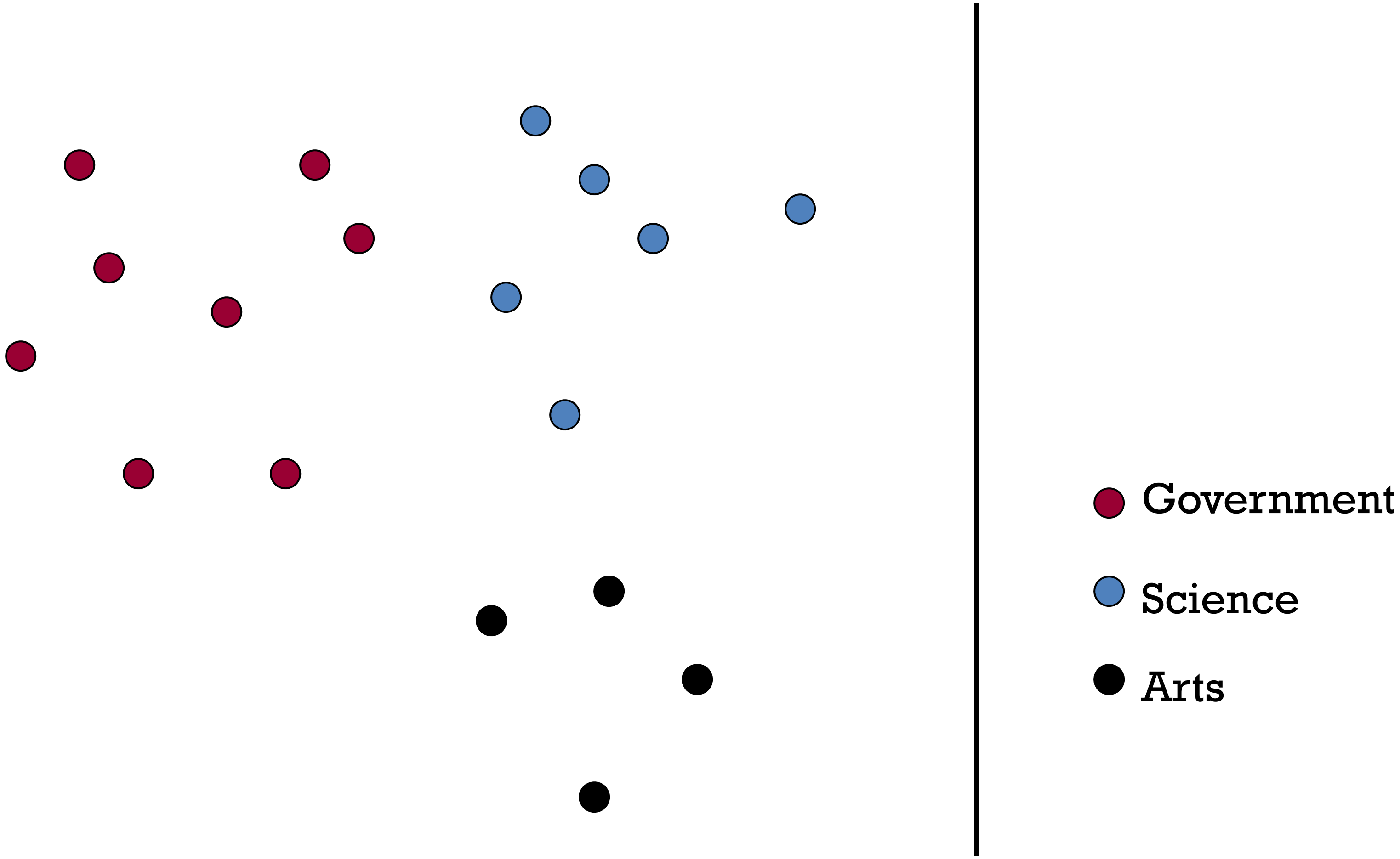
Recall: Vector Space Representation

- Each document is a vector, one component (term weight) 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?
  - Recall that Naïve Bayes classification does not make use of the term weight.

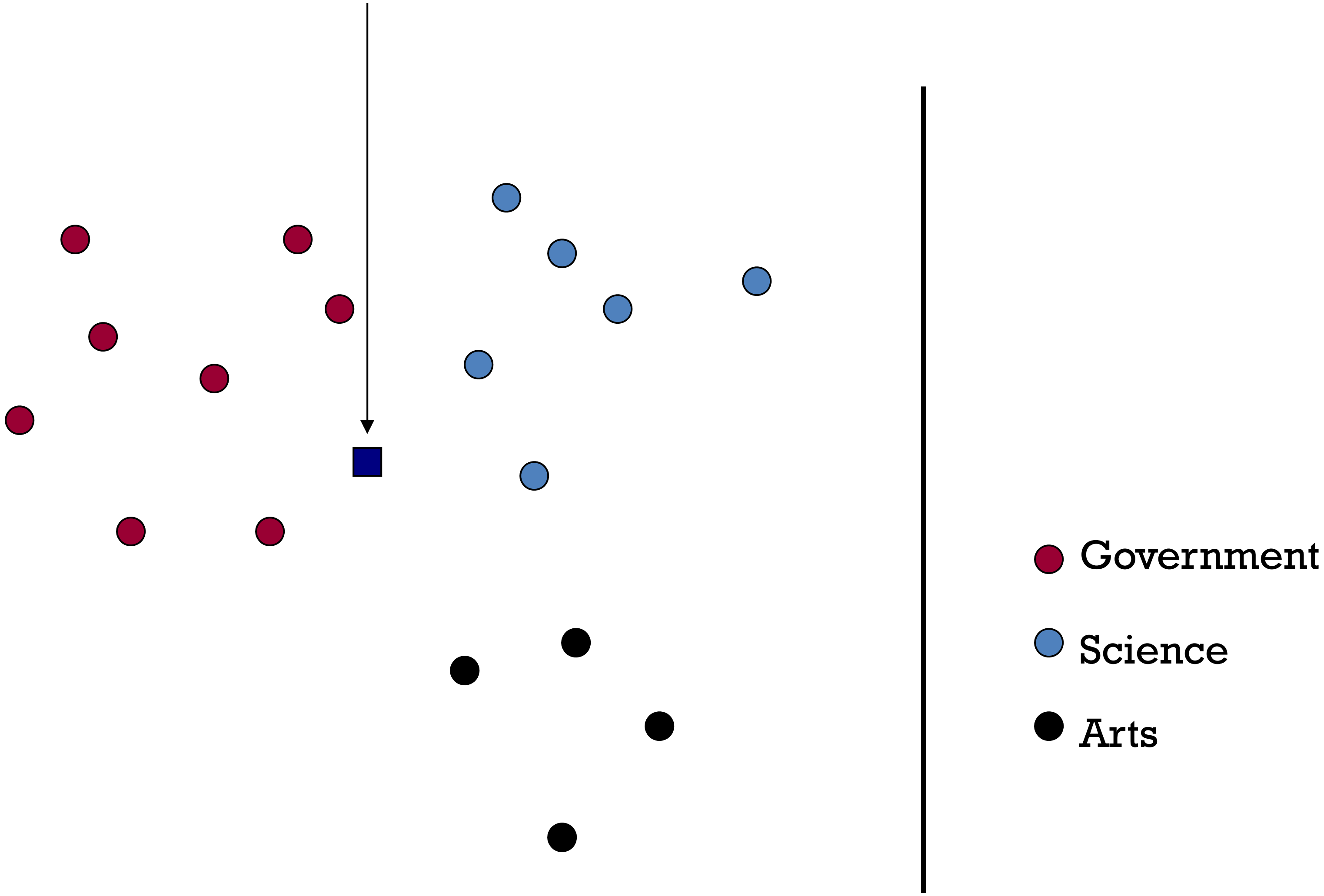
Classification Using Vector Spaces

- As before, the training set is a set of documents, each labeled with its class (e.g., topic)
- In vector space based representation, this set corresponds to a labeled set of points (or, equivalently, vectors) in the vector space
- We define surfaces to delineate classes in the space

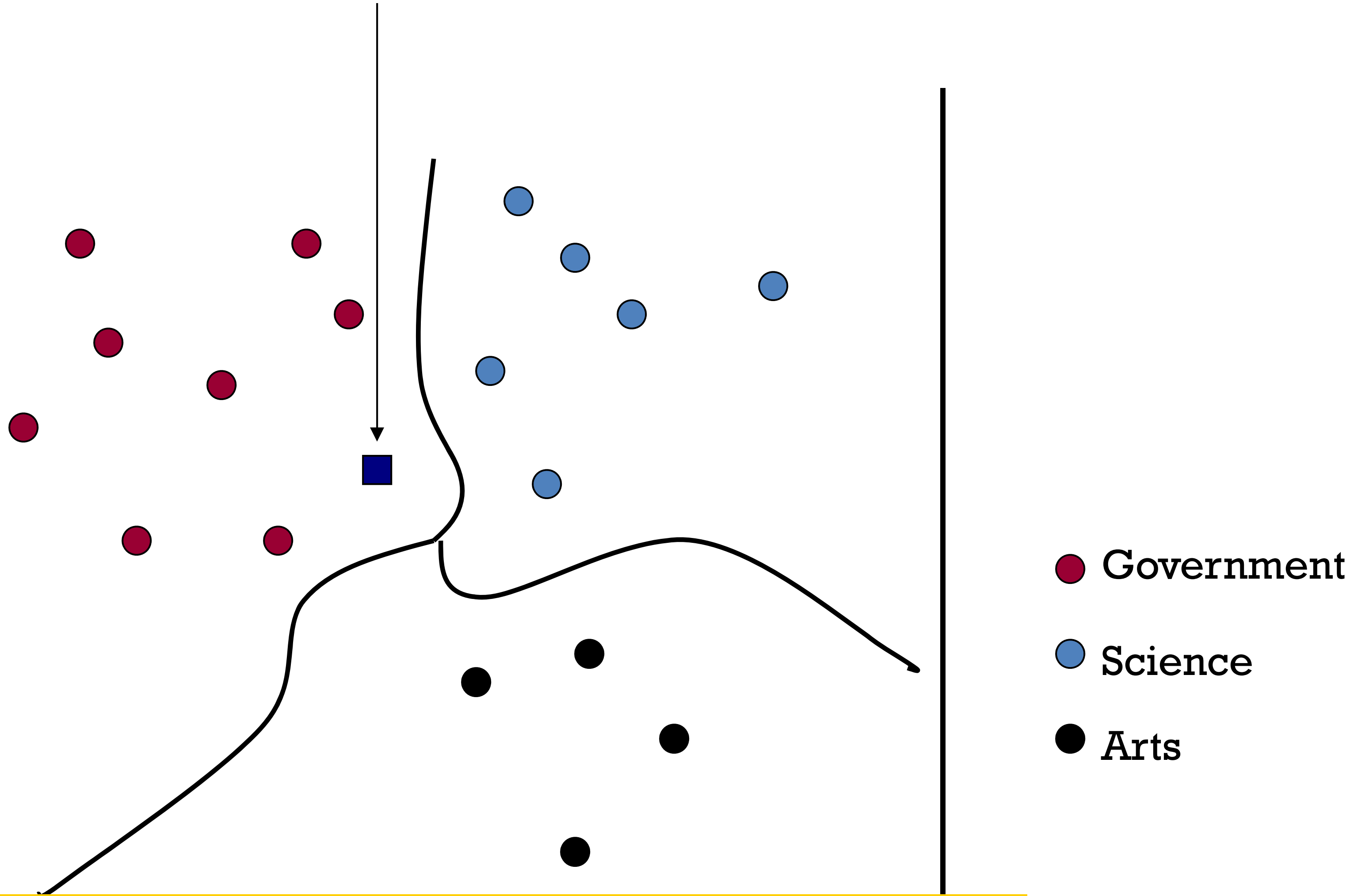
Documents in a Vector Space



Test Document of what class?



Test Document = Government



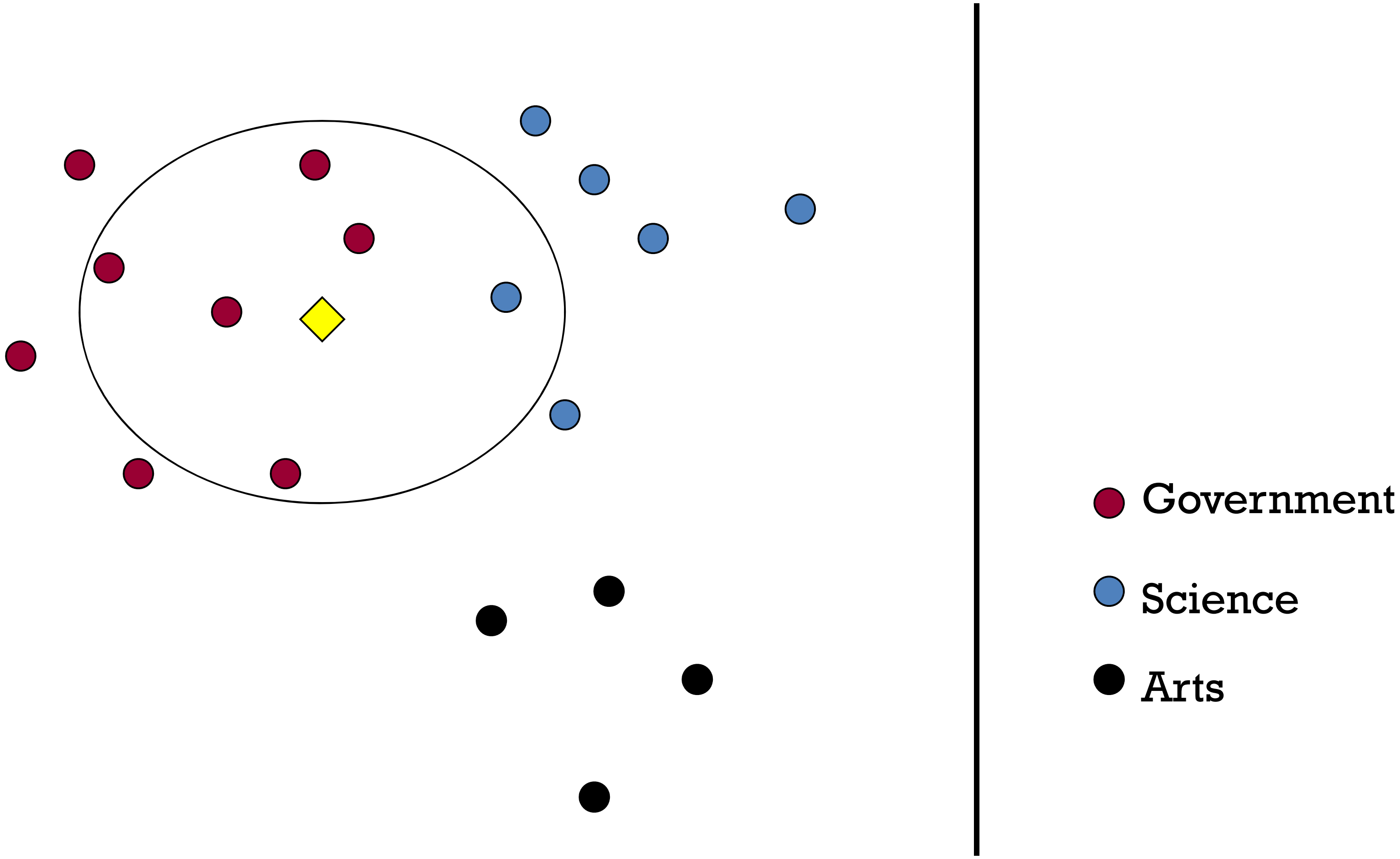
The main strategy is how to find good separators

k Nearest Neighbor Classification

kNN = k Nearest Neighbor

- Use standard TF-IDF weighted vectors to represent text documents
- To classify 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]

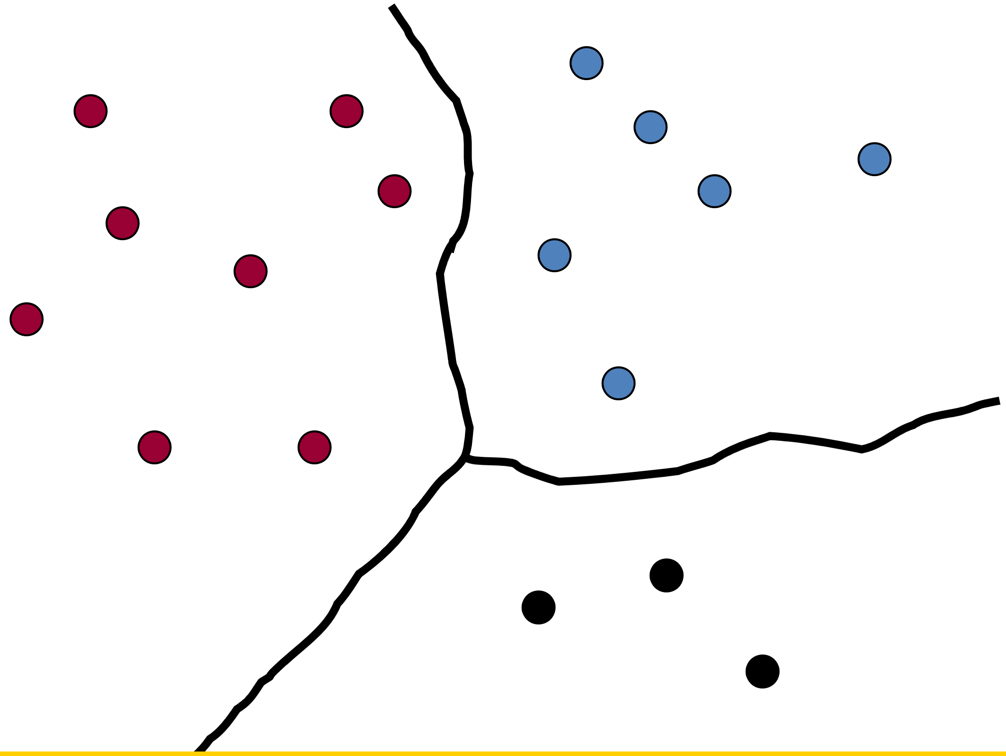
Example: k=6 (6NN)



Nearest-Neighbor Learning Algorithm

- Learning is just storing the representations of the training examples in *D*.
- Testing instance *x :*
  - Compute similarity between *x* and all examples in *D*.
  - Assign *x* the category of the most similar example in *D* (under 1NN)
  - More robust alternative is to find the k most-similar examples and return the majority category of these k examples.
- Also called:
  - Case-based learning, Memory-based learning ,Lazy learning

kNN decision boundaries



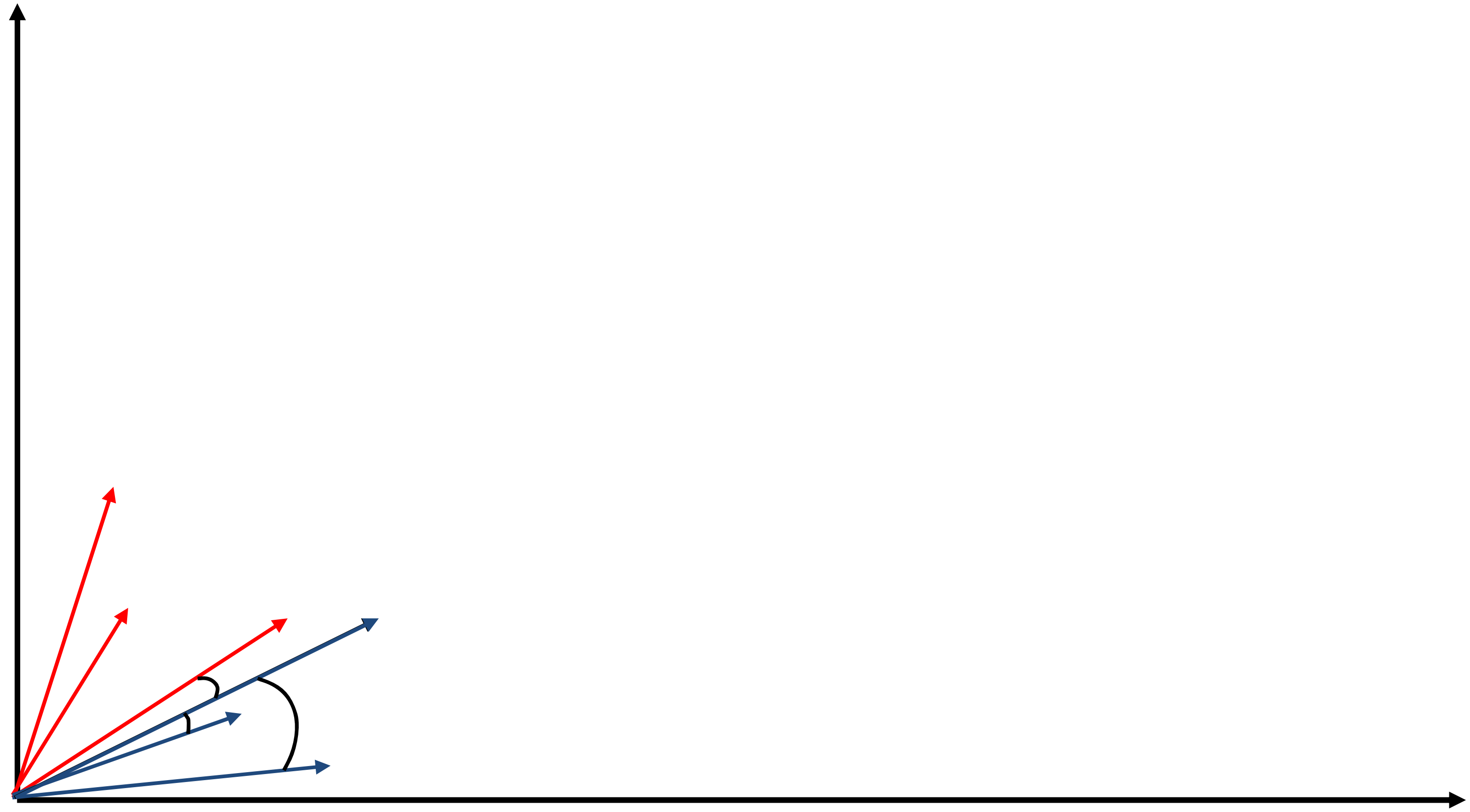
Boundaries are in principle arbitrary surfaces – but usually polyhedra

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

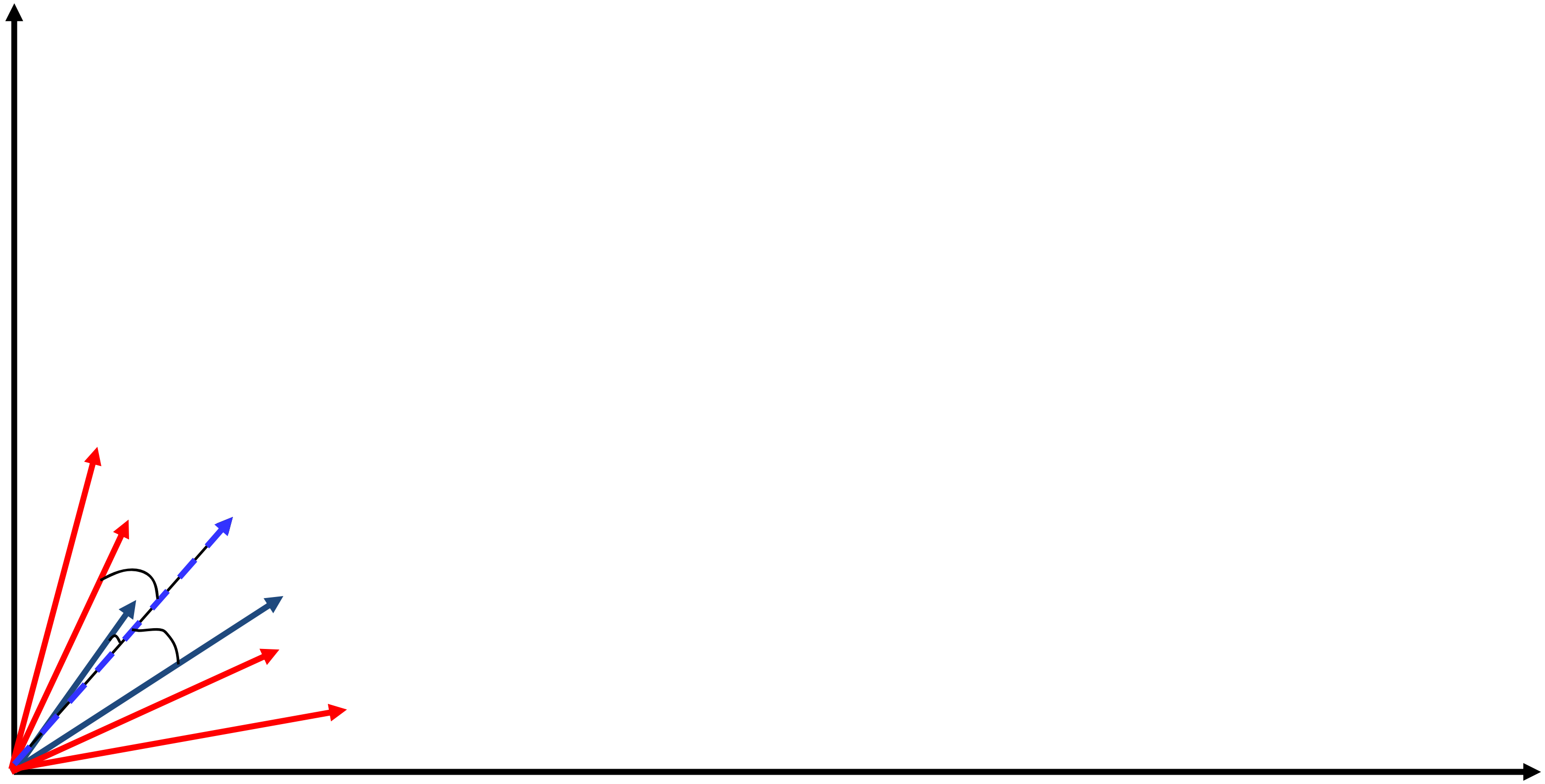
- Nearest neighbor method depends on a similarity (or distance) metric.
- For text, cosine similarity of tf.idf weighted vectors is typically most effective.

Illustration of 3 Nearest Neighbor for Text Vector Space



3 Nearest Neighbor Comparison

- Nearest Neighbor tends to handle polymorphic categories better.



k Nearest Neighbor

- Value of *k* is typically odd to avoid ties; 3 and 5 are most common, but larger values between 50 to 100 are also used.
- Alternatively, we can select *k* that gives best results on a held-out portion of the training set.

kNN Algorithm
Training (Preprocessing) and Testing

```
TRAIN-KNN($\mathbf{C}$,$\mathbf{D}$)
  $\mathbf{D}' \leftarrow$ PREPROCESS($\mathbf{D}$)
  $k \leftarrow$ SELECT-K($\mathbf{C}$,$\mathbf{D}'$)
  return $\mathbf{D}'$, $k$

APPLY-KNN($\mathbf{C}$,$\mathbf{D}'$, $k$, $d$)
  $S_k \leftarrow$ COMPUTENEARESTNEIGHBORS($\mathbf{D}'$, $k$, $d$)
  for each $c_j \in \mathbf{C}$
    do $p_{\boldsymbol{j}} \leftarrow \left|S_k \cap c_j\right|/k$
  return $\mathrm{argmax}_j p_j$
```

p_j is an estimate for $P\left(c_j \middle| S_k\right) = P(c_j|d)$

c_j denotes the set of all documents in the class c_j

kNN: Discussion

- No feature selection necessary
- Scales well with large number of classes
  - Don’t need to train *n* classifiers for *n* classes
- Classes can influence each other
  - Small changes to one class can have ripple effect
- Can avoid training if preferred
- May be more expensive at test time

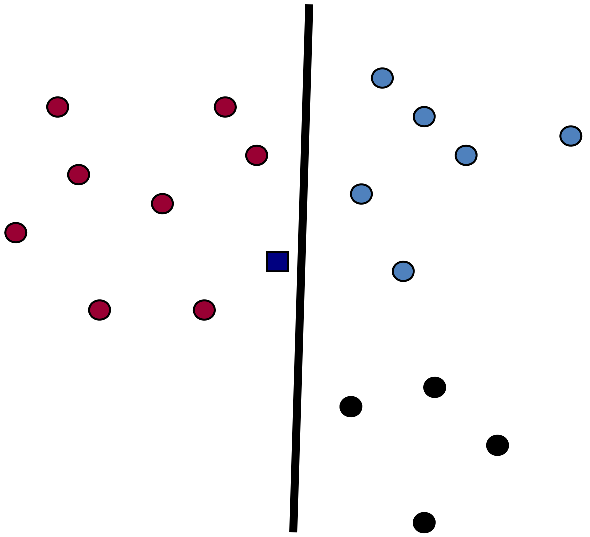
Linear Classifiers

Linear Classifiers

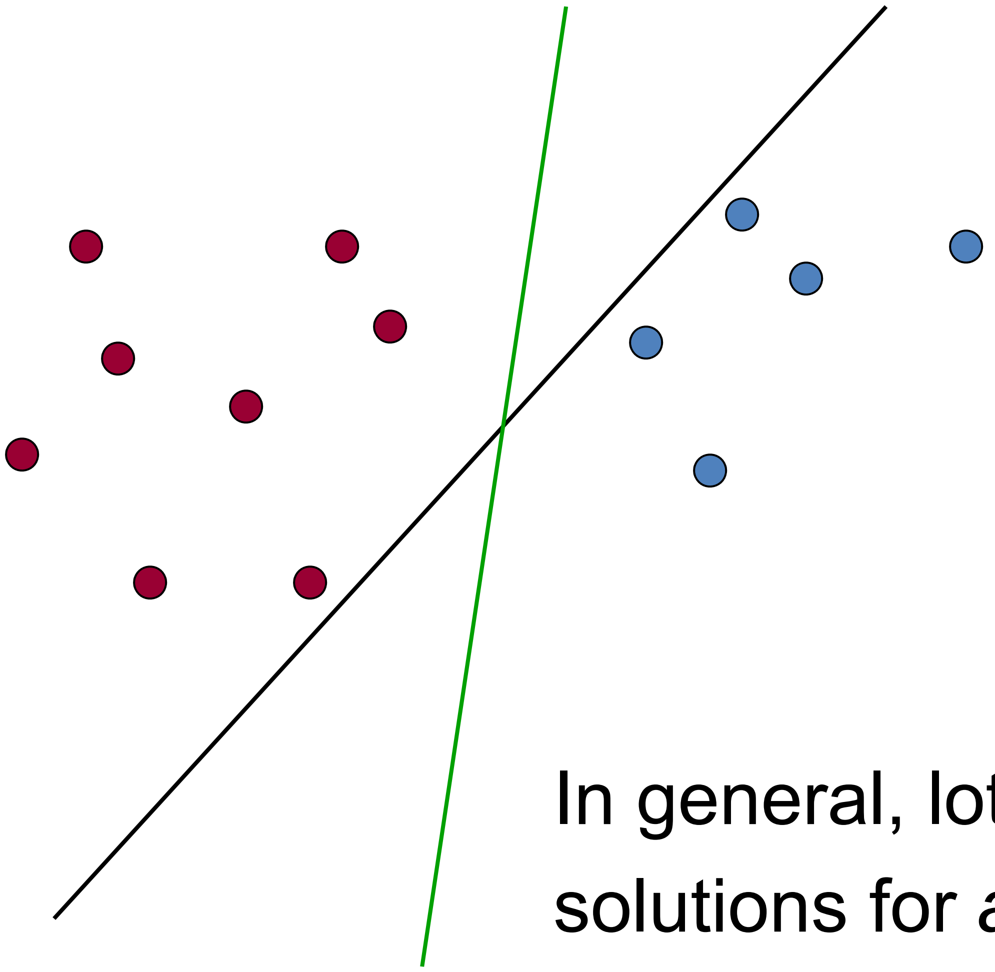
- Consider 2 class problems
  - Deciding between two classes, perhaps, government and non-government
    - One-versus-rest classification
- How do we define (and find) the separating surface?
- How do we decide which region a test doc is in?

Separation by Hyperplanes

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



Which Hyperplane?

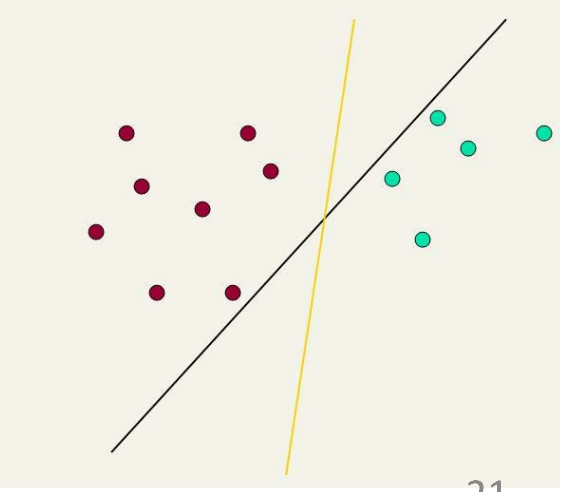


In general, lots of possible solutions for *a,b,c.*

Which Hyperplane?

- Lots of possible solutions for *a,b,c.*
- Some methods find an optimal separating hyperplane

  [according to some criterion of expected goodness]
- Which points should influence optimality?
  - All points
    - Linear regression
    - Naïve Bayes
  - Only "difficult points" close to decision boundary
    - Support vector machines



High-Dimensional Linear Classifier

- For general linear classifiers, assign the document *d* with *m* features $d=(d_1,\ldots d_M)$ to one class if:

$$\left(\sum_{i=1}^{M} w_i d_i\right) - \theta > 0$$

Otherwise, assign to the other class.

Applying Linear Classifier

```
APPLYLINEARCLASSIFIER($\vec{w}$, θ, $\vec{d}$)
score ← $\sum_{i=1}^{M} w_i d_i$
if score > θ
  then return 1
  else return 0
```

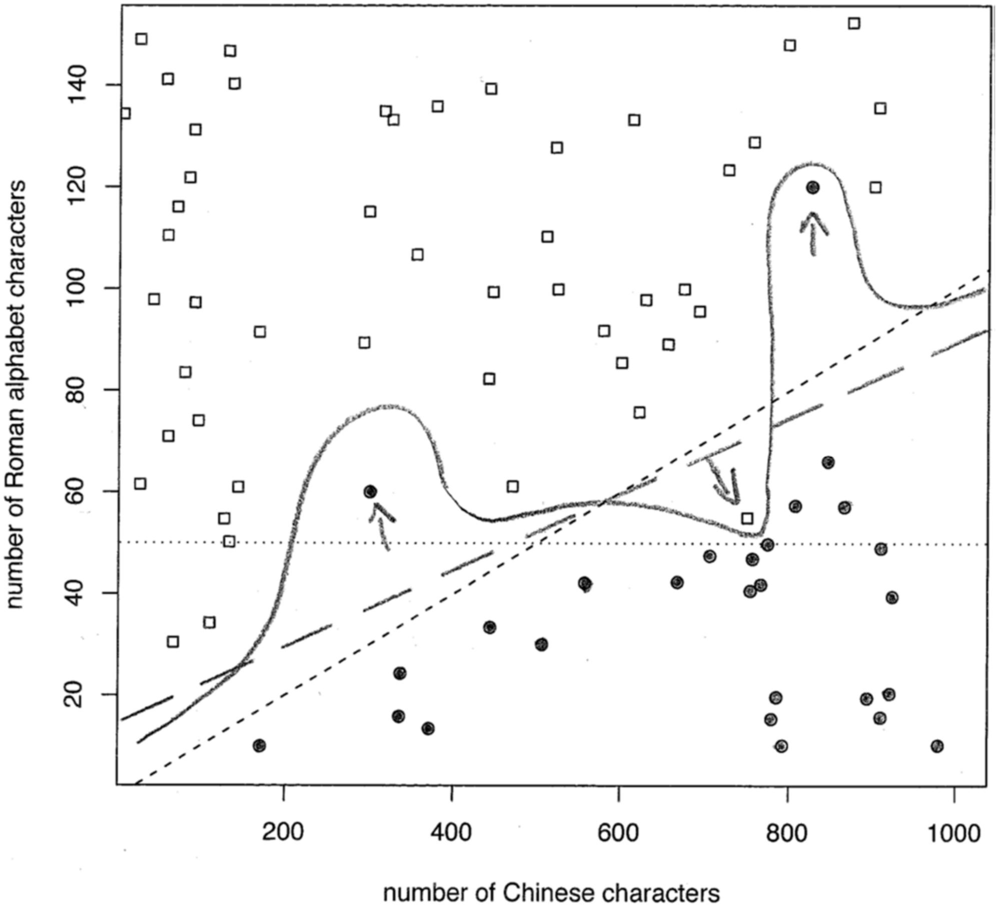
Linear Classifiers

- Many common text classifiers are linear classifiers
  - Naïve Bayes
  - Logistic regression
  - Support vector machines (with linear kernel)
  - Linear regression
- Despite this similarity, noticeable performance differences

Linear Problem

- A linear problem - The underlying distributions of the two classes are separated by a line.
- This separating line is called class boundary.
  - It is the “true” boundary and we distinguish it from the decision boundary that the learning method computes

Linear Problem

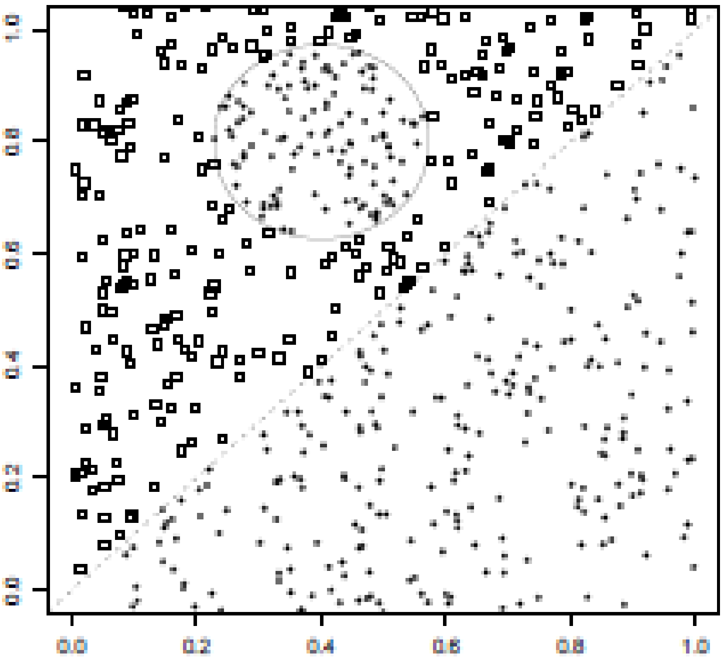


- A linear problem that classifies whether a Web page is a Chinese Web page (solid circles)
- The class boundary is represented by short dashed line
- There are three noise documents

Linear Problem

- If there exists a hyperplane that perfectly separates the two classes, then we call the two classes linearly separable.
- If linear separability holds, there is an infinite number of linear separators.
  - We need a criterion for selecting among all decision hyperplanes.

Nonlinear Problem



- If a problem is nonlinear problem and its class boundaries cannot be approximated well with linear hyperplanes, then nonlinear classifiers are better
- An example of a nonlinear classifier is kNN

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.
  - Quite common for documents

document	**class**
1	c_1, c_3, c_4
2	c_3, c_5
:	:

Set of Binary Classifiers: Any-of

- Build a separator between each class and its complementary set (docs from all other classes).
- Decompose into $|c|$ binary classification problems.
- Given test doc, evaluate it for membership in each class.
- Apply decision criterion of classifiers independently
- Though maybe you could do better by considering dependencies between categories

More Than Two Classes

- 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

Set of Binary Classifiers: One-of

- Build a separator between each class and its complementary set (docs from all other classes).
- Given test doc, evaluate it for membership in each class.
- Assign document to class with:
  - maximum score
  - maximum confidence
  - maximum probability