Computational Methods for Data Analysis

Vector Space Categorization; (On Text)

Alessandra Giordani
Department of Computer Science and Information Engineering
University of Trento
Email: agiordani@disi.unitn.it

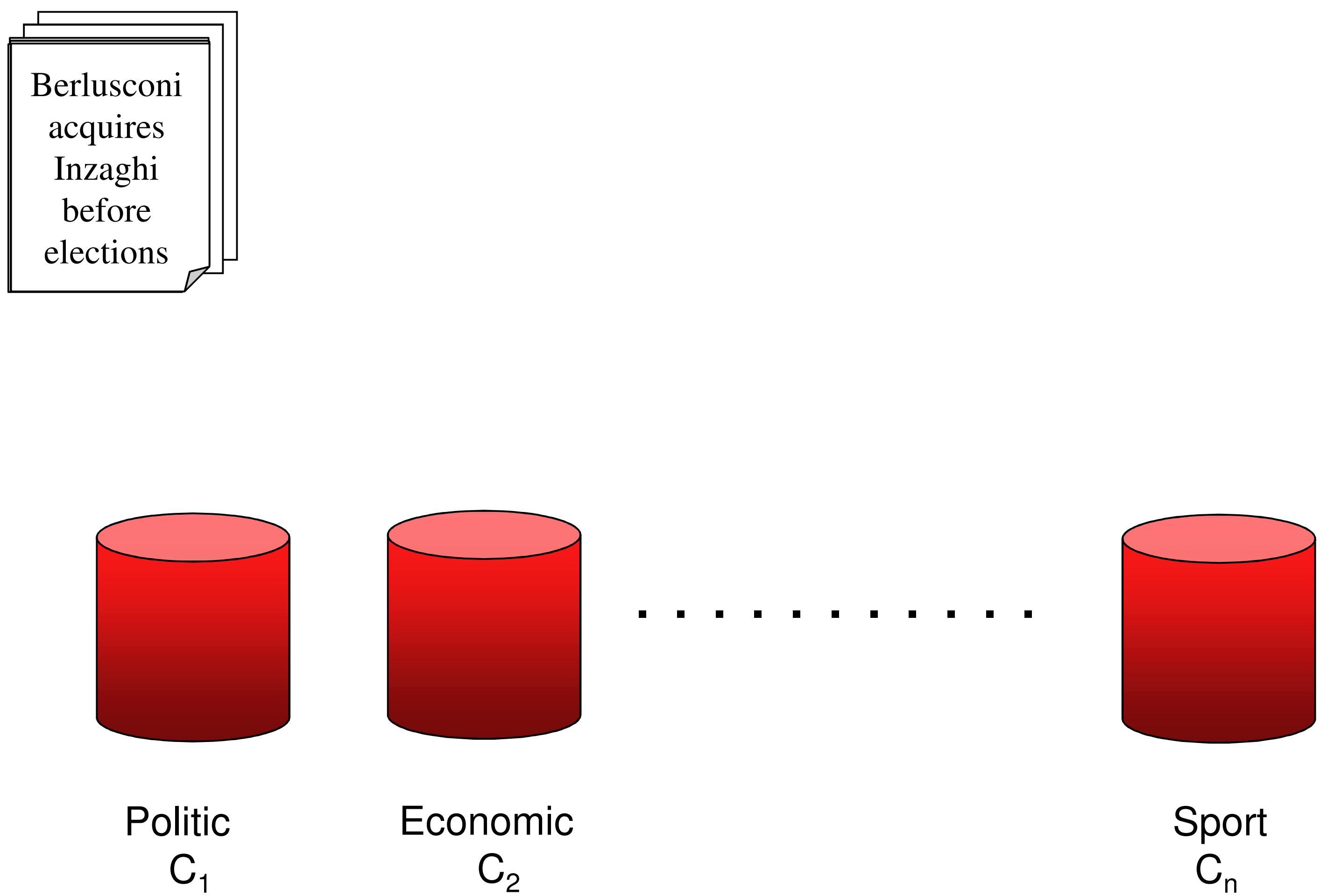
Course Schedule

- April 16: 15:45 - 18:15
- May 7: 15:45 - 18:15
- May 14: 15:45 - 18:15
- May 16: 14:30 - 17:00
- May 23: 14:30 - 17:00
- May 28: 14:30 - 17:00
- May 30: 14:30 - 17:00

Outline

- Text Categorization and Optimization
  - TC Introduction
  - TC designing steps
  - Rocchio text classifier
  - Support Vector Machines
  - Performance evaluation
  - The Parameterized Rocchio Classifier (PRC)
  - Evaluation of PRC against Rocchio and SVM

Introduction to Text Categorization



Text Classification Problem

- Given:
  - a set of target categories: $C = \{ C^1, .., C^n \}$
  - the set T of documents,

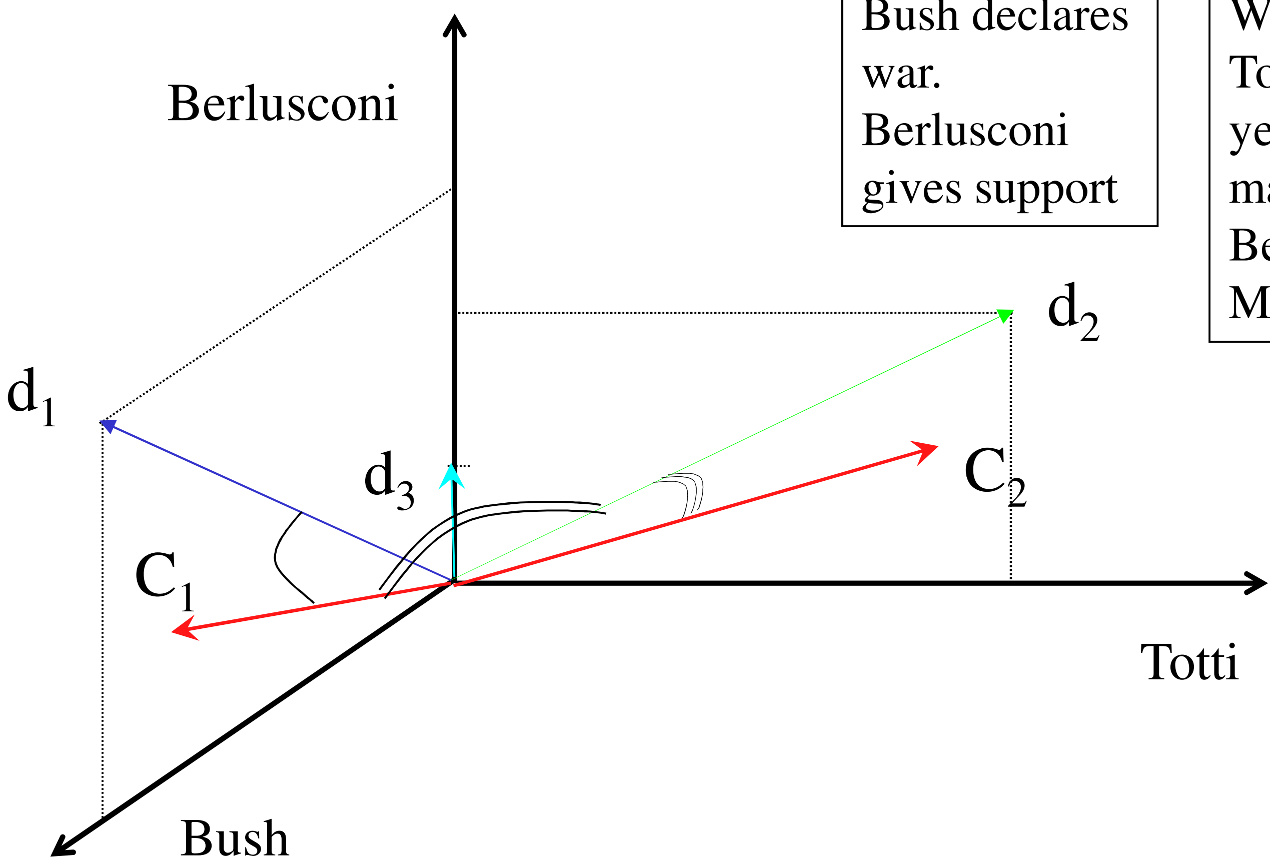
  define

  $$f : T \rightarrow 2^C$$

- VSM (Salton89')
  - Features are dimensions of a Vector Space.
  - Documents and Categories are vectors of feature weights.
  - d is assigned to C^i if

$$\vec{d} \cdot \vec{C^i} > th$$

The Vector Space Model



d_1: Politic

Bush declares war. Berlusconi gives support

d_2: Sport

Wonderful Totti in the yesterday match against Berlusconi's Milan

d_3:Economic

Berlusconi acquires Inzaghi before elections

C_1 : Politics Category

C_2 : Sport Category

Automated Text Categorization

- A corpus of pre-categorized documents
- Split document in two parts:
  - Training-set
  - Test-set
- Apply a supervised machine learning model to the training-set
  - Positive examples
  - Negative examples
- Measure the performances on the test-set
  - e.g., Precision and Recall

Feature Vectors

- Each example is associated with a vector of *n* feature types (e.g. unique words in TC)

$$\vec{x} = (0, \; ..,1,..,0,..,0, \; ..,1,..,0,..,0, \; ..,1,..,0,..,0, \; ..,1,..,0,.., \; 1)$$

acquisition buy market sell stocks

- The dot product $\vec{x} \cdot \vec{z}$ counts the number of features in common
- This provides a sort of *similarity*

Text Categorization phases

- Corpus pre-processing (e.g. tokenization, stemming)
- Feature Selection (optionally)
  - Document Frequency, Information Gain, χ_2 , mutual information,...
- Feature weighting
  - for documents and profiles
- Similarity measure
  - between document and profile (e.g. scalar product)
- Statistical Inference
  - threshold application
- Performance Evaluation
  - Accuracy, Precision/Recall, BEP, f-measure,..

Feature Selection

- Some words, i.e. features, may be irrelevant
- For example, “function words” as: “the”, “on”,”those”…
- Two benefits:
  - efficiency
  - Sometime the accuracy
- Sort features by relevance and select the *m*-best

Statistical Quantity to sort feature

- Based on corpus counts of the pair <feature,category>

- A is the number of documents in which both f and c occur, i.e. (f, c);
- B is the number of documents in which only f occurs, i.e. $(f, \bar{c})$;
- C is the number of documents in which only c occurs, i.e. $(\bar{f}, c)$;
- D is the number of documents in which neither f nor c occur, i.e. $(\bar{f}, \bar{c})$;
- N is the total number of documents, i.e. $A + B + C + D$.

Statistical Selectors

- Chi-square, Pointwise MI and MI

$$\chi^2(f,c) = \frac{N \times (AD - CB)^2}{(A+C)(B+D)(A+B)(C+D)}$$

$$PMI(f,c) = log\frac{P(f,c)}{P(f) \times P(c)}$$

$$MI(f) = -\sum_{c \in \mathcal{C}} P(c)log(P(c)) + P(f)\sum_{c \in \mathcal{C}} P(c|f)log(P(c|f)) + P(\bar{f})\sum_{c \in \mathcal{C}} P(c|\bar{f})log(P(c|\bar{f}))$$

(f, C)

Chi-Square Test

$$X^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i},$$

- O_i = an observed frequency;
- E_i = an expected (theoretical) frequency, asserted by the null hypothesis;
- n = the number of cells in the table.

Just an intuitions from Information Theory of MI

- $MI(X,Y) = H(X)-H(X|Y) = H(Y)-H(Y|X)$
- If X very similar to Y, $H(Y|X) = H(X|Y) = 0$

$\Rightarrow$ $MI(X,Y)$ is maximal

Probability Estimation

- $P(f, c)$ is the probability that f and c co-occurs and can be estimated by A/N;
- $P(f)$ is the probability of f, estimated by $(A + B)/N$;
- $P(c)$ is the probability of c, estimated by $(A + C)/N$;
- $P(c|f)$ is the probability of c by considering only the documents that contain f. It can be estimated by $\frac{P(f,c)}{P(f)}$.
- $P(\bar{f})$ is the probability that f does not occur, estimated by $(C + D)/N$;

Probability Estimation (con't)

- $P(c|\bar{f})$ is the probability of c by considering only the documents that do not contain f. It can be estimated by $\frac{P(\bar{f},c)}{P(\bar{f})}$. In turn, $P(\bar{f},c)$ is estimated by C/N.

- C is the collection of categories, i.e. $\{c_1, c_2, ..., c_n\}$. Note that PMI and χ^2 are defined on only two categories, i.e. c and *not* c whereas MI can be evaluated on $n > 2$ categories[7].

For example, we can apply the above formulas to evaluate the PMI as follows:

$$\mathrm{PMI} = \log \frac{\mathrm{N}}{\mathrm{A}+\mathrm{B}} \times \frac{\mathrm{N}}{\mathrm{A}+\mathrm{C}} \times \frac{\mathrm{A}}{\mathrm{N}} = \log \frac{\mathrm{A} \times \mathrm{N}}{(\mathrm{A}+\mathrm{C})(\mathrm{A}+\mathrm{B})}$$

Global Selectors

$$PMI_{max}(f) = \max_{c\in\mathcal{C}} \ \mathrm{PMI}(f,c)$$

$$PMI_{avg}(f) = \sum_{c\in\mathcal{C}} P(c) \times \mathrm{PMI}(f,c)$$

$$\chi^2_{max}(f) = \max_{c\in\mathcal{C}} \quad \chi^2(f,c)$$

$$\chi^2_{avg}(f) = \sum_{c\in\mathcal{C}} P(c) \times \chi^2(f,c)$$

Document weighting: an example

- N, the overall number of documents,
- N_f, the number of documents that contain the feature f
- o_f^d the occurrences of the features f in the document d
- The weight f in a document is:

$$\omega_f^d = \left(\log \frac{N}{N_f} \right) \times o_f^d = IDF(f) \times o_f^d$$

Inverse Document Frequency

- The weight can be normalized:

$$\omega'^d_f = \frac{\omega_f^d}{\sqrt{\sum_{t \in d} (\omega_t^d)^2}}$$

Similarity estimation

- Given the document and the category representation

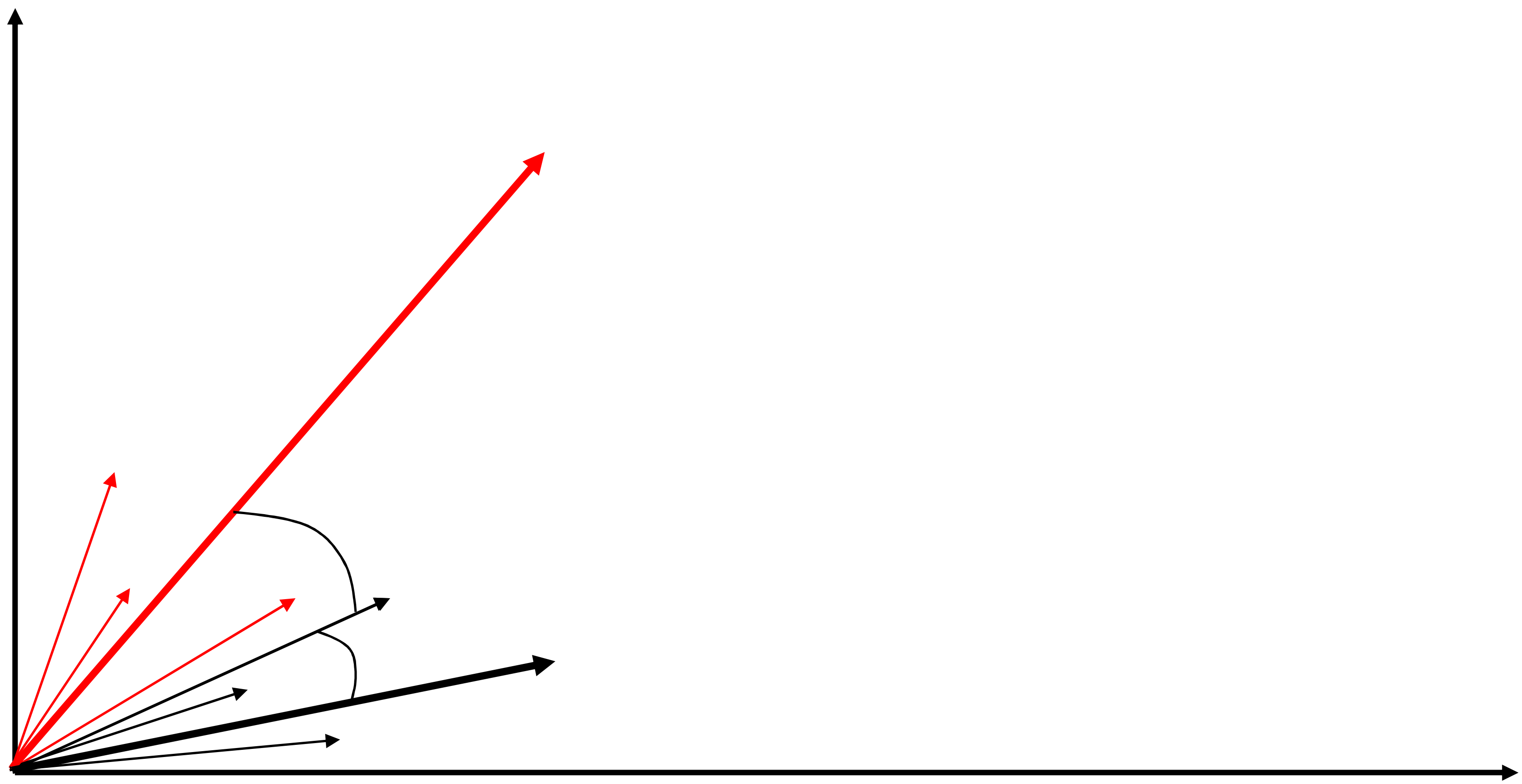
$$\vec{d} = \left\langle \omega_{f_1}^{d}, \ldots, \omega_{f_n}^{d} \right\rangle, \quad \vec{C}_i = \left\langle \Omega_{f_1}^{i}, \ldots, \Omega_{f_n}^{i} \right\rangle$$

- It can be defined the following similarity function (cosine measure

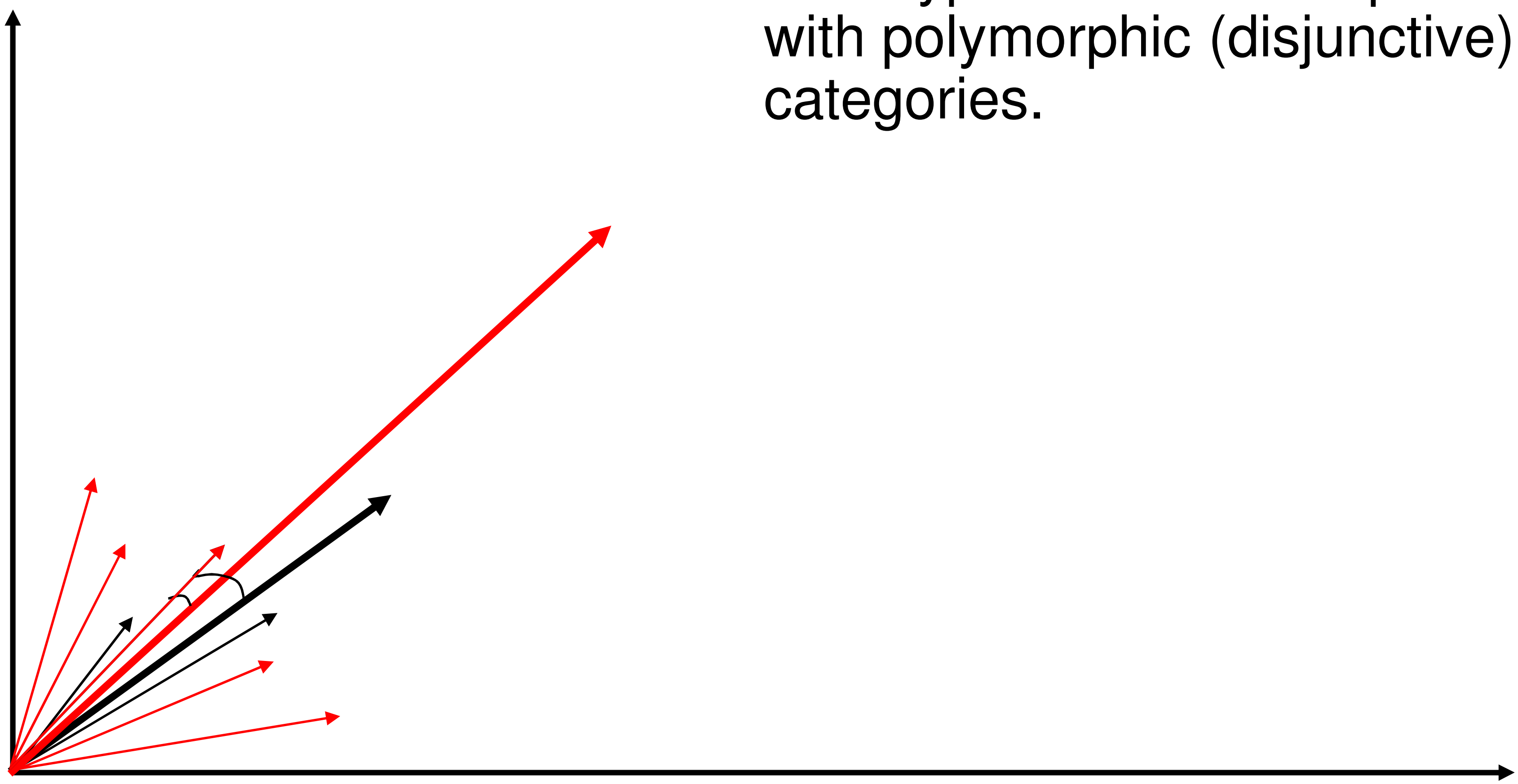
$$s_{d,i} = \cos(\vec{d}, \vec{C}_i) = \frac{\vec{d} \cdot \vec{C}^i}{\|\vec{d}\| \times \|\vec{C}_i\|} = \frac{\sum_f \omega_f^d \times \Omega_f^i}{\|\vec{d}\| \times \|\vec{C}_i\|}$$

- *d* is assigned to C^i if $\vec{d} \cdot \vec{C}^i > \sigma$

Bidimensional view of Rocchio categorization



Rocchio problems



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

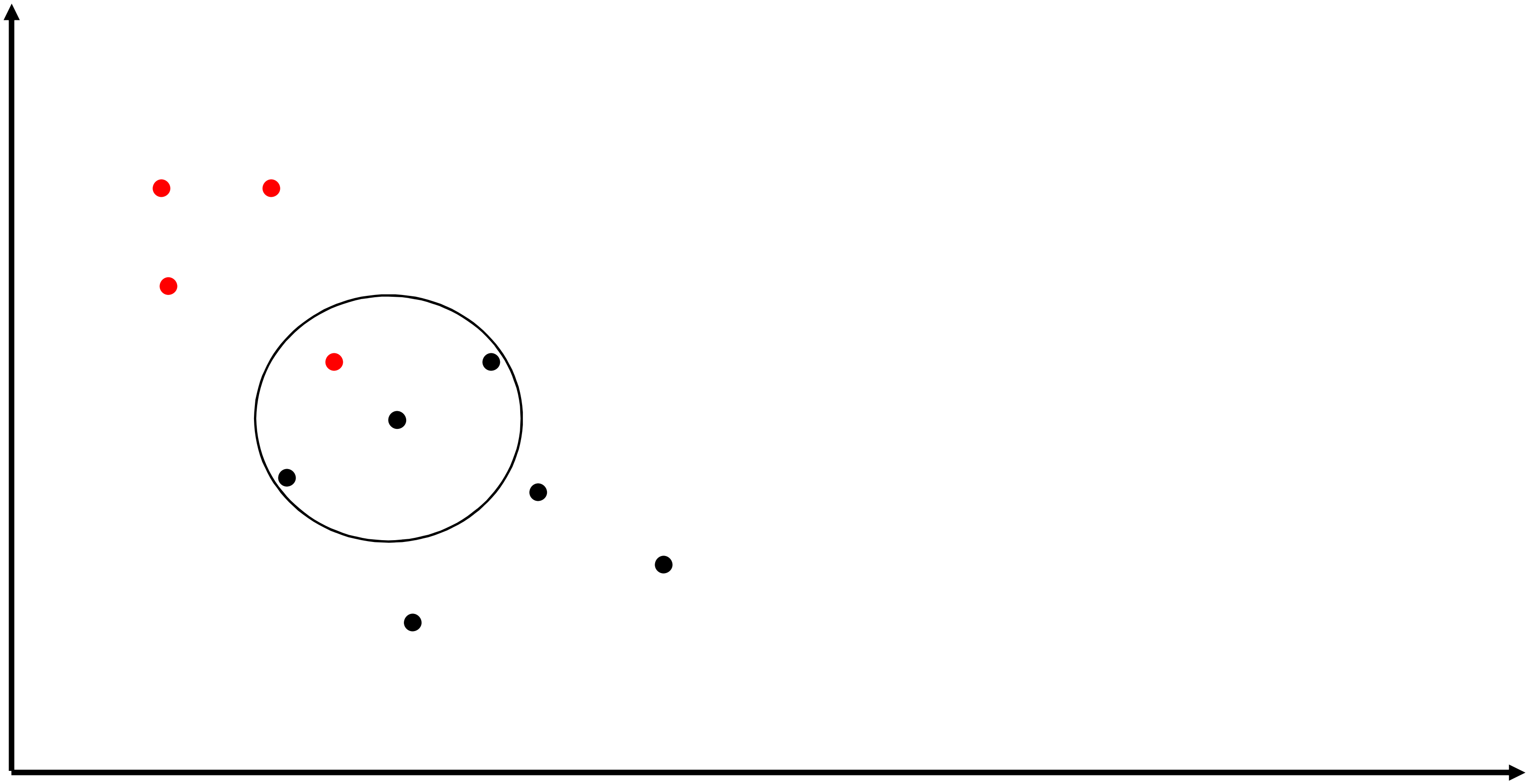
Nearest-Neighbor Learning Algorithm

- 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*.
- Does not explicitly compute a generalization or category prototypes.
- Also called:
  - Case-based
  - Memory-based
  - Lazy learning

K Nearest-Neighbor

- Using only the closest example to determine categorization is subject to errors due to:
  - A single atypical example.
  - Noise (i.e. 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, 3 and 5 are most common.

3 Nearest Neighbor Illustration
(Euclidian Distance)



K Nearest Neighbor for Text

Training:

For each each training example $<x, c(x)> \in D$

Compute the corresponding TF-IDF vector, $\mathbf{d}_x$, for document x

Test instance *y*:

Compute TF-IDF vector $\mathbf{d}$ for document y

For each $<x, c(x)> \in D$

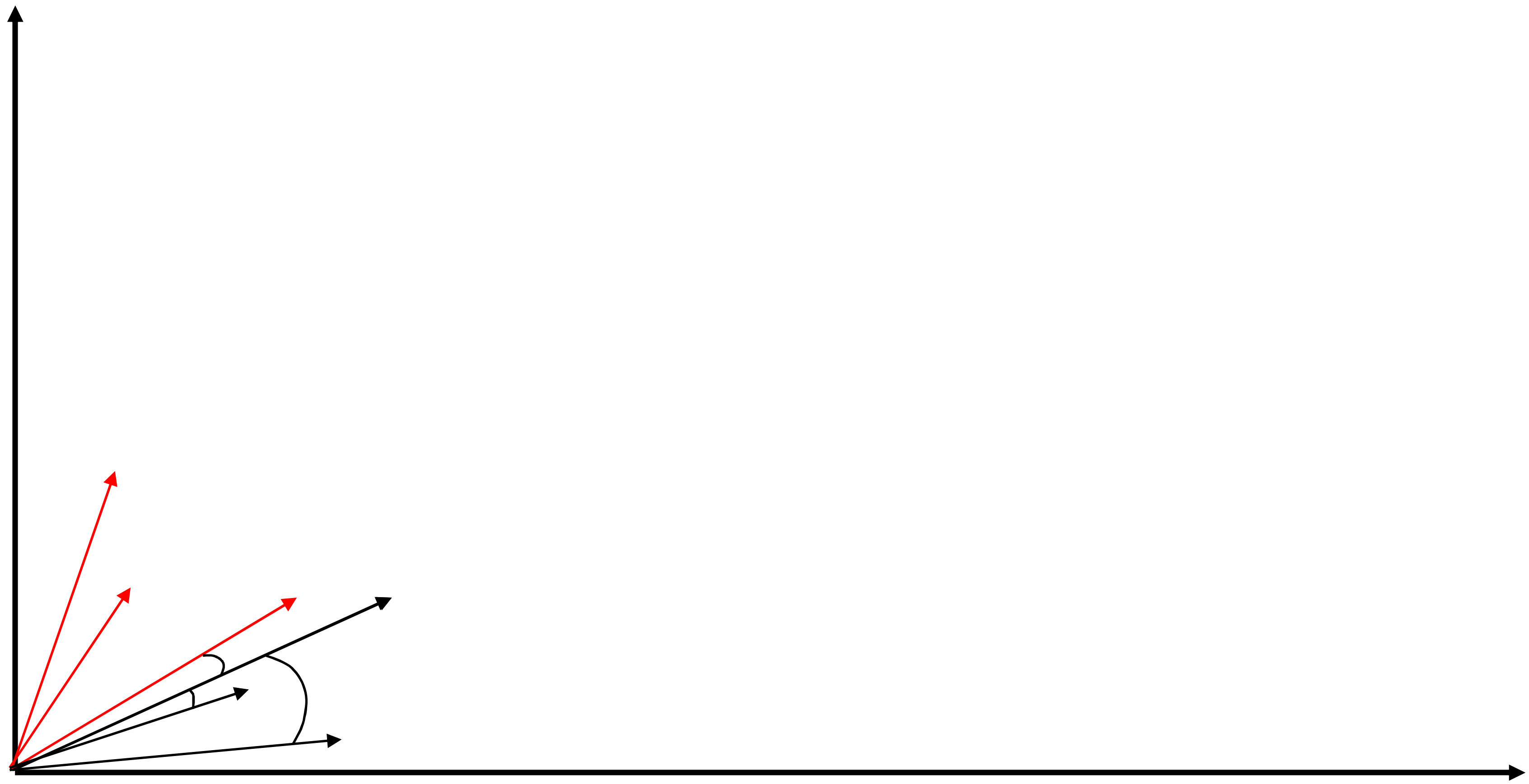
Let $s_x = \text{cosSim}(\mathbf{d}, \mathbf{d}_x)$

Sort examples, x, in D by decreasing value of s_x

Let N be the first k examples in D. (*get most similar neighbors*)

Return the majority class of examples in N

Illustration of 3 Nearest Neighbor for Text



A state-of-the-art classifier: Support Vector Machines

- The Vector $\vec{C}^i$ satisfies:

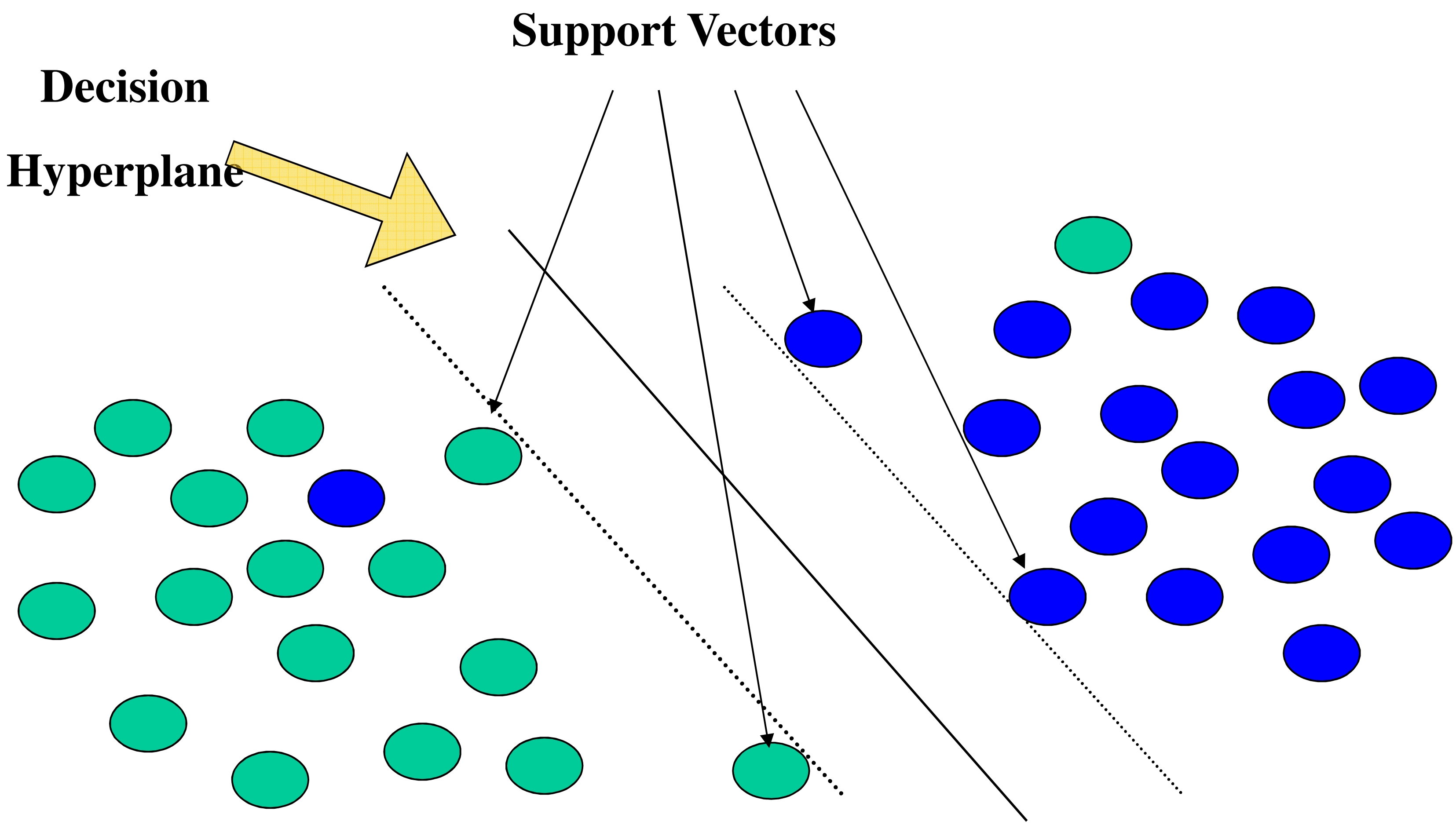
$$\min \left| \vec{C}^i \right|$$

$$\vec{C}^i \times \vec{d} - th \geq +1, \ \text{ if } d \in T_i$$

$$\vec{C}^i \times \vec{d} - th \leq -1, \text{ if } d \notin T_i$$

- *d* is assigned to C^i if $\vec{d} \times \vec{C}^i > th$

SVM

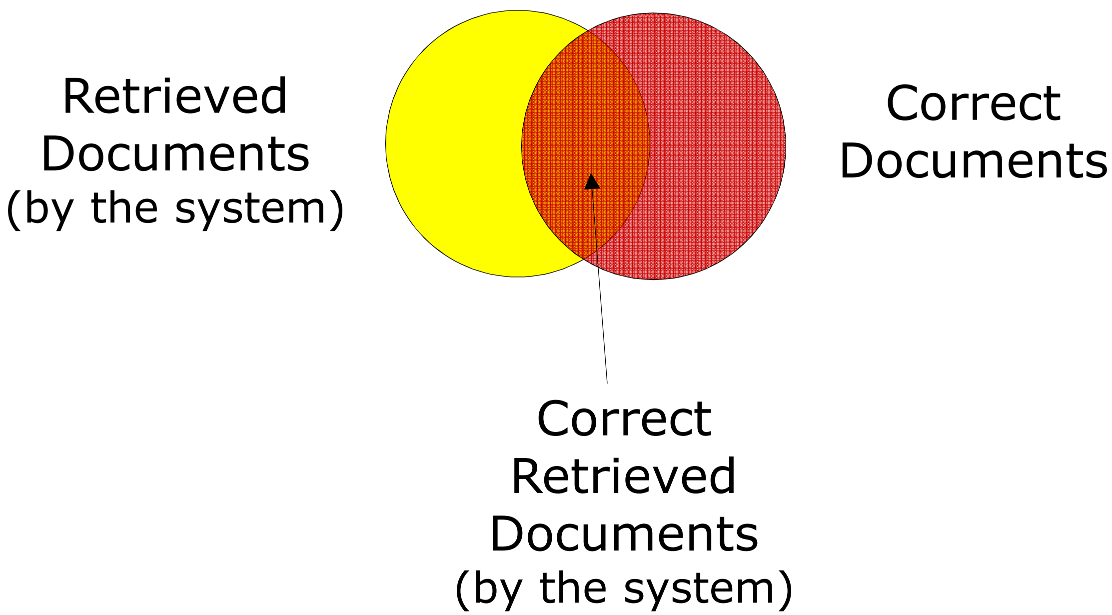


Other Text Classifiers

- *RIPPER* [Cohen and Singer, 1999] uses an extended notion of a profile. It learns the contexts that are positively correlated with the target classes, i.e. words co-occurrence.
- EXPERT uses as context nearby words (sequence of words).
- *CLASSI* is a system that uses a neural network-based approach to text categorization [Ng *et al.*, 1997]. The basic units of the network are only perceptrons.
- *Dtree* [Quinlan, 1986] is a system based on a well-known machine learning model.
- *CHARADE* [I. Moulinier and Ganascia, 1996] and *SWAP*1 [Apt´e *et al.*, 1994] use machine learning algorithms to inductively extract Disjunctive Normal Form rules from training documents.

Performance Measurements

- Given a set of document *T*
- Precision = # Correct Retrieved Document / # Retrieved Documents
- Recall = # Correct Retrieved Document/ # Correct Documents



Precision and Recall of C_i

- a, corrects
- b, mistakes
- c, not retrieved

The *Precision* and *Recall* are defined by the above counts:

$$Precision_i = \frac{a_i}{a_i + b_i}$$

$$Recall_i = \frac{a_i}{a_i + c_i}$$

Performance Measurements (cont'd)

- Breakeven Point
  - Find thresholds for which

    Recall = Precision
  - Interpolation
- f-measure
  - Harmonic mean between precision and recall
- Global performance on more than two categories
  - Micro-average
    - The counts refer to classifiers
  - Macro-average (average measures over all categories)

F-measure e MicroAverages

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

$$\mu Precision = \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} a_i + b_i}$$

$$\mu Recall = \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} a_i + c_i}$$

$$\mu BEP = \frac{\mu Precision + \mu Recall}{2}$$

$$\mu f_1 = \frac{2 \times \mu Precision \times \mu Recall}{\mu Precision + \mu Recall}$$

Parameter Estimation Procedure

- Validation-set of about 30% of the training corpus
- for all $\rho \in [0,30]$
  - TRAIN the system on the remaining material
  - Measure the BEP on the validation-set
- Select the ρ associated with the highest *BEP*
- re-TRAIN the system on the entire training-set
- TEST the system based on the obtained parameterized model
- For more reliable results:
  - 20 validation-sets and made the ρ average
- The Parameterized Rocchio Classifier will refer to as PRC

Comparative Analysis

- Rocchio literature parameterization
  - $\rho = 1$ ($\gamma = \beta = 1$) *and* $\rho = 1/4$ ($\gamma = 4$, $\beta = 16$)
- Reuters fixed test-set
  - Other literature results
- SVM
  - To better collocate our results
- Cross Validation (20 samples)
  - More reliable results
- Cross corpora/language validation
  - Reuters, Ohsumed (English) and ANSA (Italian)

Cross-Validation

1. Generate n random splits of the corpus. For each split j, 70% of data can be used for training (LS^j) and 30% for testing (TS^j).

2. For each split j
   (a) Generate m validation sets, ES^j_k of about 10/30% of LS^j.
   (b) Learn the classifiers on $LS^j - ES^j{}_k$ and for each $ES^j{}_k$ evaluate: (i) the threshold associated to the BEP and (ii) the optimal parameter ρ.
   (c) Learn the classifiers Rocchio, SVMs and PRC on LS^j: in case of PRC use the estimated $\bar{\rho}$.
   (d) Evaluate f_1 on TS_j (use the estimated thresholds for Rocchio and PRC) for each category and account data for the final processing of the global μf_1.

3. For each classifier evaluate the mean and the Standard Deviation for f_1 and μf_1 over the TS_j sets.

N-fold cross validation

- Divide training set in *n* parts
  - One is used for testing
  - *n-1* for training
- This can be repeated *n* times for *n* distinct test sets
- Average and Std. Dev. are the final performance index

Ohsumed and ANSA corpora

- Ohsumed:
  - Including 50,216 medical abstracts.
  - The first 20,000 documents year 91,
  - 23 *MeSH diseases* categories [Joachims, 1998]
- ANSA:
  - 16,000 news items in Italian from the ANSA news agency.
  - 8 target categories,
  - 2,000 documents each,
  - e.g. Politics, Sport or Economics.
- Testing 30 %

An Ohsumed document:
Bacterial Infections and Mycoses

Replacement of an aortic valve cusp after neonatal endocarditis. Septic arthritis developed in a neonate after an infection of her hand. Despite medical and surgical treatment endocarditis of her aortic valve developed and the resultant regurgitation required emergency surgery. At operation a new valve cusp was fashioned from preserved calf pericardium. Nine years later she was well and had full exercise tolerance with minimal aortic regurgitation.

Cross validation on Ohsumed/ANSA (20 samples)

	Rocchio		PRC	SVM
Ohsumed	BEP		f1	f1
MicroAvg.	ρ=.25	ρ=1		
(23 cat.)	54.4 ± .5	61.8 ±.5	65.8±.4	68.37±.5

	Rocchio		PRC
ANSA	BEP		f1
MicroAvg.	ρ=.25	ρ=1	
(8 cat.)	61.76 ±.5	67.23 ±.5	71.00 ±.4

Computational Complexity

- PRC
  - Easy to implement
  - Low training complexity: O(*n*m log n*m*)
    - (*n* = number of doc and *m* = max num of features in a document)
  - Low classification complexity:

    *min{O(M), O(m*log(M))}* (*M* is the max num of features in a profile)
  - *Good accuracy: the second top accurate classifier on Reuters*
- SVM
  - More complex implementation
  - Higher Learning time > $O(n^2)$ (to solve the quadratic optimization problem)
  - Actually is linear for linear SVMs
  - Low complexity of classification phase (for linear SVM) =

    *min{O(M), O(m*log(M))}*

From Binary to Multiclass classifiers

- Three different approaches:
- **ONE-vs-ALL (OVA)**
  - Given the example sets, {E1, E2, E3, …} for the categories: {C1, C2, C3,…} the binary classifiers: {b1, b2, b3,…} are built.
  - For b1, E1 is the set of positives and E2∪E3 ∪… is the set of negatives, and so on
  - <u>For testing</u>: given a classification instance x, the category is the one associated with the maximum margin among all binary classifiers

From Binary to Multiclass classifiers

- **ALL-vs-ALL (AVA)**
  - Given the examples: {E1, E2, E3, …} for the categories {C1, C2, C3,…}
    - build the binary classifiers:

      {b1_2, b1_3,…, b1_n, b2_3, b2_4,…, b2_n,…,bn-1_n}
    - by learning on E1 (positives) and E2 (negatives), on E1 (positives) and E3 (negatives) and so on…
  - <u>For testing</u>: given an example x,
    - all the votes of all classifiers are collected
    - where $b_{E1E2} = 1$ means a vote for C1 and $b_{E1E2} = -1$ is a vote for C2
  - Select the category that gets more votes