

UNIVERSITÀ DEGLI STUDI DI BARI  
Facoltà di Scienze Matematiche, Fisiche e Naturali  
Dipartimento di Informatica

# Text Categorization

---

Prof. Giovanni Semeraro  
Dott. Pasquale Lops  
Dott. Marco Degemmis

Corso di Gestione della Conoscenza d'Impresa  
Anno Accademico 2005-2006



## Outline

- Problem Definition: Categorization
- Machine Learning (ML) e Text Categorization (TC)
- Indexing
- Building a Classifier
- Evaluation
- Applications of ML & TC (& IR)
- References

## Il task della classificazione

- Dati:
  - ✓ La descrizione di una istanza,  $x \in X$ , dove  $X$  è lo spazio delle istanze
  - ✓ Un insieme finito di categorie:  $C = \{c_1, c_2, \dots, c_n\}$
- Determina:
  - ✓ La categoria di  $x$ :  $c(x) \in C$ , dove  $c(x)$  è una funzione di classificazione il cui dominio è  $X$  e il cui range è  $C$ .

## Machine Learning for Categorization

- Un esempio di training è un'istanza  $x \in X$ , associata alla categoria di appartenenza  $c(x)$ :  $\langle x, c(x) \rangle$  dove  $c$  è una funzione di classificazione *non nota*
- Dato un insieme di esempi di training,  $D$
- Trova una ipotesi di funzione di classificazione,  $h(x)$ , tale che:

$$\forall \langle x, c(x) \rangle \in D : h(x) = c(x)$$

*Consistenza*

## Sample Category Learning Problem

- Spazio delle istanze:  $\langle \text{size, color, shape} \rangle$ 
  - ✓  $\text{size} \in \{\text{small, medium, large}\}$
  - ✓  $\text{color} \in \{\text{red, blue, green}\}$
  - ✓  $\text{shape} \in \{\text{square, circle, triangle}\}$
- $C = \{\text{positive, negative}\}$
- $D$ :

Example	Size	Color	Shape	Category
1	small	red	circle	positive
2	large	red	circle	positive
3	small	red	triangle	negative
4	large	blue	circle	negative

Text Categorization

5

## Generalizzazione

- L'obiettivo è quello di generalizzare le osservazioni (esempi) al fine di classificare correttamente istanze non appartenenti al training set.
- Memorizzare semplicemente gli esempi garantisce la **consistenza** ma non generalizza!
- Il processo di apprendimento mira a definire un **modello generale** di classificazione

Text Categorization

6

## Text Categorization

- ➔ Il termine **Text Categorization** designa l'assegnazione automatica di una o più categorie a testi scritti in linguaggio naturale
- ➔ Applicazioni:
  - ✓ Web pages
    - Recommending
    - Yahoo-like classification
  - ✓ Newsgroup Messages
    - Recommending
    - spam filtering
  - ✓ News articles
    - Personalized newspaper
  - ✓ Email messages
    - Folderizing
    - spam filtering

Text Categorization

7

## Definizione del problema (1)

- ➔ Formalmente il problema di TC consiste nell'approssimare la funzione target:
  - ✓  $\phi': D \times C \rightarrow \{T, F\}$  attraverso la funzione
  - ✓  $\phi: D \times C \rightarrow \{T, F\}$
- ➔  $\phi$  si dice classificatore e deve risultare il più fedele possibile a  $\phi'$
- ➔  $D$  è l'insieme dei documenti
- ➔  $C$  è l'insieme delle categorie

Text Categorization

8

## Definizione del problema (2)

- Single-label
  - ✓ Binario
- Multi-label
- Il single label è possibile solo nel caso vi sia indipendenza tra le categorie (solitamente vero)
- Utilizzando il single label binario, classificare sotto C vuol dire risolvere esattamente  $|C|$  volte il problema di categorizzazione single-label binario
- Il caso SL binario è
  - ✓ Più generale
  - ✓ Usato molto frequentemente
  - ✓ Molto dettagliato in letteratura

## Machine Learning e TC

- I primi sistemi di TC utilizzavano *knowledge engineering*
- Learning Algorithms:
  - ✓ **Bayesian (naïve)**
  - ✓ Neural network
  - ✓ **Relevance Feedback (Rocchio)**
  - ✓ Rule based (Ripper)
  - ✓ **Nearest Neighbor (case based)**
  - ✓ Support Vector Machines (SVM)
- Costruzione induttiva di un classificatore che svolge le funzioni di un sistema esperto
- Vantaggi: buona manutenibilità (non è necessario ingegnerizzare di continuo le regole di classificazione)
- Svantaggi: necessita base di documenti pre-classificati che permettano la costruzione del classificatore

## Train and Test

- Sia dato  $D = \{d_1, \dots, d_{|D|}\}$  e  $C = \{c_1, \dots, c_{|C|}\}$
- La funzione  $\phi'$  è nota per ogni valore di  $\langle d_j, c_i \rangle \in D \times C$ 
  - ✓ Se il suo valore è True allora si dice che  $\langle d_j, c_i \rangle$  è un esempio positivo per  $\phi'$ , in caso contrario si dice negativo
- Definiamo due sotto-insiemi di D:
  - ✓ Tr (Training Set)
  - ✓ Te (Test Set)
  - ✓  $D = \text{Tr} \cup \text{Te}$
- Il classificatore è costruito a partire dagli esempi di Tr
- Il testing (validazione) viene effettuato usando gli esempi di Te

Text Categorization

11

## Indicizzazione

- Il classificatore deve ricevere una codifica compatta del testo da classificare, perciò usiamo tecniche di indicizzazione
- Solitamente un documento  $d_j$  viene rappresentato come un insieme di pesi, che indicano la presenza (o la frequenza) di termini tratti da un insieme di parole (o frasi)
- La scelta tipica per tale insieme è il *bag of words*
- In alternativa si possono usare frasi denotate
  - ✓ Sintatticamente (frasi del linguaggio)
  - ✓ Statisticamente (insiemi di parole)
- Solitamente i pesi hanno valori in  $[0, 1]$  oppure assumono i soli valori 0 e 1 (caso binario)

Text Categorization

12

## Indicizzazione (2)

- La formula standard per il calcolo dei pesi è:

$$tfidf(t_k, d_j) = \#(t_k, d_j) \cdot \log \frac{|Tr|}{\#Tr(t_k)}$$

- Per far ricadere i pesi in  $[0,1]$  bisogna usare la *normalizzazione del coseno*:

$$w_{kj} = \frac{tfidf(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|} (tfidf(t_s, d_j))^2}}$$

dove  $T$  è l'insieme dei termini (*features*) che compaiono almeno una volta in nei documenti di  $Tr$

- Prima dell'indicizzazione vengono rimosse le parole come articoli e proposizioni non relazionate coi topic in esame
- L'uso dello *stemming* è controverso (raggruppamento di parole con la stessa radice morfologica)

## Dimensionality reduction

- La dimensione dello spazio dei termini può costituire un problema perché:
  - ✓ Gli algoritmi di learning non "scalano" facilmente su grandi valori della dimensione
  - ✓ Se la dimensione è alta spesso si verificano fenomeni di *overfitting*
- Abbiamo due scelte:
  - ✓ Riduzione locale (un insieme di termini diverso per ciascuna categoria)
  - ✓ Riduzione globale (il set di termini è valido per qualunque categoria)
- La riduzione può essere:
  - ✓ Per selezione
  - ✓ Per estrazione

## Dimensionality reduction (selection)

- ➔ **TSR** (Term Space Reduction) seleziona un sotto-insieme dell'insieme dei termini che massimizzi la effectiveness al momento della indicizzazione
- ➔ **Wrapper** partendo da un insieme di termini applica l'algoritmo di learning a sotto-insieme di termini ottenuti eliminando un termine per volta, fino ad ottenere la effectiveness massima
- ➔ **Filtering** invece seleziona un sotto-insieme dei termini in accordo ai valori forniti da una funzione che misura "l'importanza" dei termini nel processo di categorizzazione

## Funzioni per la selection

Function	Denoted by	Mathematical form
Document frequency	$\#(t_k, c_i)$	$P(t_k c_i)$
DIA association factor	$z(t_k, c_i)$	$P(c_i t_k)$
Information gain	$IG(t_k, c_i)$	$\sum_{c \in \{c_i, \bar{c}_i\}} \sum_{t \in \{t_k, \bar{t}_k\}} P(t, c) \cdot \log \frac{P(t, c)}{P(t) \cdot P(c)}$
Mutual information	$MI(t_k, c_i)$	$\log \frac{P(t_k, c_i)}{P(t_k) \cdot P(c_i)}$
Chi-square	$\chi^2(t_k, c_i)$	$\frac{ Tr  \cdot [P(t_k, c_i) \cdot P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i) \cdot P(\bar{t}_k, c_i)]^2}{P(t_k) \cdot P(\bar{t}_k) \cdot P(c_i) \cdot P(\bar{c}_i)}$
NGL coefficient	$NGL(t_k, c_i)$	$\frac{\sqrt{ Tr  \cdot [P(t_k, c_i) \cdot P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i) \cdot P(\bar{t}_k, c_i)]}}{\sqrt{P(t_k) \cdot P(\bar{t}_k) \cdot P(c_i) \cdot P(\bar{c}_i)}}$
Relevancy score	$RS(t_k, c_i)$	$\log \frac{P(t_k c_i) + d}{P(t_k \bar{c}_i) + d}$
Odds Ratio	$OR(t_k, c_i)$	$\frac{P(t_k c_i) \cdot (1 - P(t_k \bar{c}_i))}{(1 - P(t_k c_i)) \cdot P(t_k \bar{c}_i)}$
GSS coefficient	$GSS(t_k, c_i)$	$P(t_k, c_i) \cdot P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i) \cdot P(\bar{t}_k, c_i)$

## Metodi di costruzione

- Classificatori probabilistici
- Classificatori basati su alberi di decisione
- Classificatori basati su regole di decisione
- Metodi di regressione
- Metodi on-line
- Metodo di Rocchio
- Reti neurali
- Classificatori example-based
- SVM

## Using Relevance Feedback (Rocchio)

- Metodi di Relevance feedback possono essere utilizzati in TC.
- I documenti possono essere rappresentati usando vettori di pesi TF/IDF (normalizzati mediante la frequenza massima dei termini)
- Per ogni categoria, si computa un vettore *prototype* sommando i vettori dei documenti di training appartenenti alla categoria.
- Si assegnano documenti di testing alla categoria con il vettore prototipo più vicino secondo la similarità del coseno

## Rocchio Algorithm (Training)

Assume the set of categories is  $\{c_1, c_2, \dots, c_n\}$   
For  $i$  from 1 to  $n$  let  $\mathbf{p}_i = \langle 0, 0, \dots, 0 \rangle$  (*init. prototype vectors*)  
For each training example  $\langle x, c(x) \rangle \in Tr$   
    Let  $\mathbf{d}$  be the frequency normalized TF/IDF term vector  
        for doc  $x$   
    Let  $i = j: (c_j = c(x))$   
        (*sum all the document vectors in  $c_i$  to get  $\mathbf{p}_i$* )  
    Let  $\mathbf{p}_i = \mathbf{p}_i + \mathbf{d}$

Text Categorization

19

## Rocchio Algorithm (Test)

Given test document  $x$   
Let  $\mathbf{d}$  be the TF/IDF weighted term vector for  $x$   
Let  $m = -2$  (*init. maximum cosSim*)  
For  $i$  from 1 to  $n$ :  
    (*compute similarity to prototype vector*)  
    Let  $s = \text{cosSim}(\mathbf{d}, \mathbf{p}_i)$   
    if  $s > m$   
        let  $m = s$   
        let  $r = c_i$  (*update most similar class prototype*)  
Return class  $r$

Text Categorization

20

## Proprietà dell'alg. di Rocchio

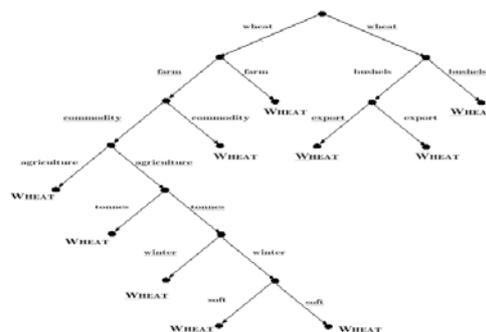
- Non garantisce un'ipotesi consistente.
- Forma una semplice generalizzazione degli esempi in ogni classe (*prototipo*)
- La classificazione è basata sulla similarità dei documenti ai vettori prototipo
- I vettori prototipo non necessitano di medie/normalizzazioni poichè la similarità del coseno non è influenzata dalla lunghezza

Text Categorization

21

## Alberi di decisione (1)

- Un classificatore basato su alberi di decisione ha:
  - ✓ per nodi interni i *termini* usati per l'indicizzazione
  - ✓ le diramazioni dei nodi sono test sui termini
  - ✓ per foglie le *categorie* previste



Text Categorization

22

## Alberi di decisione (2)

- La classificazione avviene testando ricorsivamente i valori dei pesi relativi ai termini associati ai nodi interni.
- Un metodo per il learning è invece il seguente:
  - ✓ Verificare se tutti gli esempi di training appartengono alla stessa categoria ( $c_i$  o la sua negazione).
  - ✓ In caso negativo, selezionare un termine  $t_k$  che partizioni il training-set in classi di documenti in cui al termine  $t_k$  sia assegnato lo stesso valore, ponendo ciascuna classe in un sotto-albero diverso.
  - ✓ Ripetere ricorsivamente sui sotto-alberi finché tutte le foglie non contengono esempi appartenenti alla stessa categoria (che diviene l'etichetta della foglia)
- La scelta di  $t_k$  per partizionare l'albero viene effettuata massimizzando l'*information gain*

Text Categorization

23

## Regole di decisione

- Una regola di decisione per categorizzare un testo è espressa in forma normale disgiuntiva (disgiunzione di clausole congiuntive)

<b>if</b>	$((wheat \ \& \ \textit{farm})$	<b>or</b>
	$(wheat \ \& \ \textit{commodity})$	<b>or</b>
	$(bushels \ \& \ \textit{export})$	<b>or</b>
	$(wheat \ \& \ \textit{tonnes})$	<b>or</b>
	$(wheat \ \& \ \textit{winter} \ \& \ \neg \ \textit{soft}))$	<b>then</b> WHEAT <b>else</b> $\neg$ WHEAT

- I classificatori basati su regole di decisione sono simili a quelli basati su alberi di decisione ma sono in genere più compatti.
- I letterali denotano la presenza (o l'assenza) di termini nel documento sotto esame, mentre la testa della clausola stabilisce di classificare sotto una data categoria.

Text Categorization

24

## Regole di decisione (2)

- ➔ Inizialmente ogni esempio di training viene trasformato in una clausola:
  - ✓  $\eta_1, \eta_2, \eta_3 \dots \eta_N \rightarrow \gamma_i$
- ➔ I termini  $\eta_i$  rappresentano la presenza di termini nel documento in esame mentre  $\gamma_i$  vale  $c_i$  oppure la sua negazione.
- ➔ Un set di clausole è già un classificatore che però causa *overfitting*.
- ➔ Attraverso una procedura di generalizzazione il set viene semplificato il più possibile giungendo ad una rappresentazione compatta che non sacrifichi però la "copertura"

## Nearest-Neighbor Learning Algorithm

- ➔ L'apprendimento consiste solo nel memorizzare gli esempi di training presenti in  $Tr$
- ➔ Data un'istanza di testing  $x$ :
  - ✓ Calcola la similarità tra  $x$  e tutti gli esempi in  $Tr$ .
  - ✓ Assegna a  $x$  la categoria dell'esempio più simile in  $Tr$ .
- ➔ Non effettua esplicitamente una generalizzazione o determina prototipi di categoria
- ➔ AKA:
  - ✓ Case-based
  - ✓ Memory-based
  - ✓ Lazy learning

## K Nearest-Neighbor

- Effettuare la classificazione basandosi esclusivamente sull'esempio più vicino (simile) può generare errori:
  - ✓ Esempio atipico
  - ✓ Rumore (errori) nell'etichettare gli esempi di training
- Alternativa: trovare i  $k$  esempi più simili e restituire la classe della maggioranza di questi
- Tipicamente  $k$  è dispari per evitare parità (3 e 5 sono i valori più comuni)

Text Categorization

27

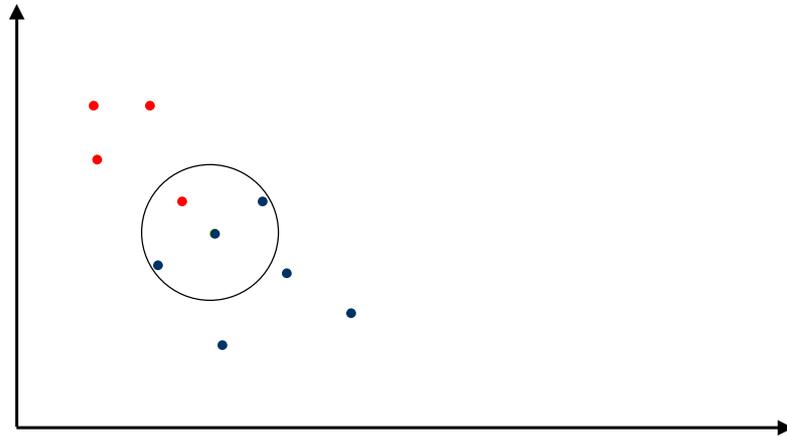
## Similarity Metrics

- Il metodo Nearest neighbor dipende da una misura di similarità (distanza)
- Per spazi  $m$ -dimensionali continui la distanza più semplice è quella *euclidea*
- Per spazi  $m$ -dimensionali binari la distanza più semplice è quella di *Hamming* (numero di valori diversi)
- Testi: la similarità del coseno tra vettori TF-IDF è quella più efficace.

Text Categorization

28

### 3 Nearest Neighbor (Euclidian Distance)



Text Categorization

29

### K Nearest Neighbor for Text

**Training:**

For each training example  $\langle x, c(x) \rangle \in Tr$

    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  $\langle x, c(x) \rangle \in Tr$

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

Sort examples,  $x$ , in  $Tr$  by decreasing value of  $s_x$

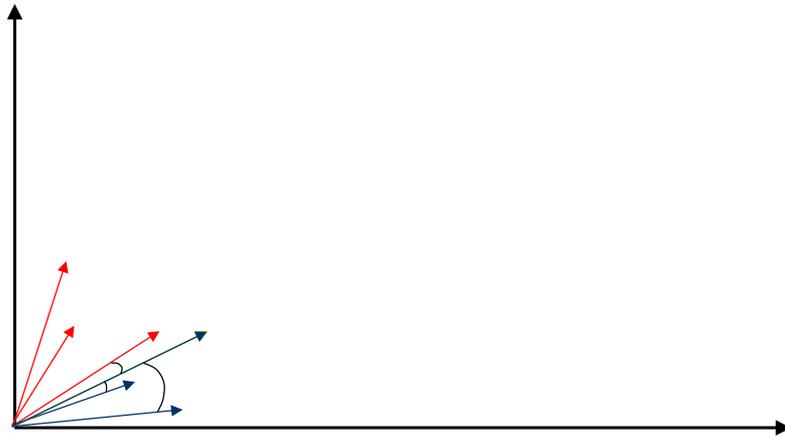
Let  $N$  be the first  $k$  examples in  $Tr$  (get most similar neighbors)

Return the majority class of examples in  $N$

Text Categorization

30

### 3 Nearest Neighbor for Text



Text Categorization

31

### Bayesian Methods

- Learning and classification methods based on probability theory.
- Bayes theorem plays a critical role in probabilistic learning and classification.
- Uses *prior* probability of each category given no information about an item.
- Categorization produces a *posterior* probability distribution over the possible categories given a description of an item.

Text Categorization

32

## Axioms of Probability Theory

- All probabilities between 0 and 1

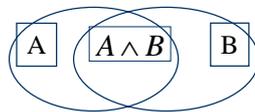
$$0 \leq P(A) \leq 1$$

- True proposition has probability 1, false has probability 0.

$$P(\text{true}) = 1 \quad P(\text{false}) = 0.$$

- The probability of disjunction is:

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$



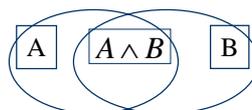
Text Categorization

33

## Conditional Probability

- $P(A | B)$  is the probability of  $A$  given  $B$
- Assumes that  $B$  is all and only information known.
- Defined by:

$$P(A | B) = \frac{P(A \wedge B)}{P(B)}$$



Text Categorization

34

## Independence

- ↪  $A$  and  $B$  are *independent* iff:

$$P(A | B) = P(A)$$

$$P(B | A) = P(B)$$

- ↪ Therefore, if  $A$  and  $B$  are independent:

$$P(A | B) = \frac{P(A \wedge B)}{P(B)} = P(A)$$

$$P(A \wedge B) = P(A)P(B)$$

## Bayes Theorem

$$P(H | E) = \frac{P(E | H)P(H)}{P(E)}$$

Simple proof from definition of conditional probability:

$$P(H | E) = \frac{P(H \wedge E)}{P(E)}$$

$$P(E | H) = \frac{P(H \wedge E)}{P(H)}$$

$$P(H \wedge E) = P(E | H)P(H)$$

$$P(H | E) = \frac{P(E | H)P(H)}{P(E)}$$

## Bayesian Categorization

- Let set of categories be  $\{c_1, c_2, \dots, c_n\}$
- Let  $E$  be description of an instance.
- Determine category of  $E$  by determining for each  $c_i$

$$P(c_i | E) = \frac{P(c_i)P(E | c_i)}{P(E)}$$

- $P(E)$  can be determined since categories are complete and disjoint.

$$\sum_{i=1}^n P(c_i | E) = \sum_{i=1}^n \frac{P(c_i)P(E | c_i)}{P(E)} = 1$$

$$P(E) = \sum_{i=1}^n P(c_i)P(E | c_i)$$

Text Categorization

37

## Bayesian Categorization (cont.)

- Need to know:
  - ✓ Priors:  $P(c_i)$
  - ✓ Conditionals:  $P(E | c_i)$
- $P(c_i)$  are easily estimated from data.
  - ✓ If  $n_i$  of the examples in  $D$  are in  $c_i$ , then  $P(c_i) = n_i / |D|$
- Assume instance is a conjunction of binary features:

$$E = e_1 \wedge e_2 \wedge \dots \wedge e_m$$

- Too many possible instances (exponential in  $m$ ) to estimate all  $P(E | c_i)$

Text Categorization

38

## Naïve Bayesian Categorization

- If we assume features of an instance are independent given the category ( $c_i$ ) (*conditionally independent*):

$$P(E | c_i) = P(e_1 \wedge e_2 \wedge \dots \wedge e_m | c_i) = \prod_{j=1}^m P(e_j | c_i)$$

- Therefore, we then only need to know  $P(e_j | c_i)$  for each feature and category.

## Naïve Bayes Example

- $C = \{\text{allergy, cold, well}\}$
- $e_1 = \text{sneeze}; e_2 = \text{cough}; e_3 = \text{fever}$
- $E = \{\text{sneeze, cough, } \neg\text{fever}\}$

Prob	Well	Cold	Allergy
$P(c_i)$	0.9	0.05	0.05
$P(\text{sneeze}   c_i)$	0.1	0.9	0.9
$P(\text{cough}   c_i)$	0.1	0.8	0.7
$P(\text{fever}   c_i)$	0.01	0.7	0.4

## Naïve Bayes Example (cont.)

Probability	Well	Cold	Allergy
$P(c_i)$	0.9	0.05	0.05
$P(\text{sneeze}   c_i)$	0.1	0.9	0.9
$P(\text{cough}   c_i)$	0.1	0.8	0.7
$P(\text{fever}   c_i)$	0.01	0.7	0.4

$E = \{\text{sneeze, cough, -fever}\}$

$$P(\text{well} | E) = (0.9)(0.1)(0.1)(0.99)/P(E) = 0.0089/P(E)$$

$$P(\text{cold} | E) = (0.05)(0.9)(0.8)(0.3)/P(E) = 0.01/P(E)$$

$$P(\text{allergy} | E) = (0.05)(0.9)(0.7)(0.6)/P(E) = 0.019/P(E)$$

Most probable category: allergy

## Estimating Probabilities

- Normally, probabilities are estimated based on observed frequencies in the training data.
- If  $D$  contains  $n_i$  examples in category  $c_i$ , and  $n_{ij}$  of these  $n_i$  examples contains feature  $e_j$ , then:

$$P(e_j | c_i) = \frac{n_{ij}}{n_i}$$

- However, estimating such probabilities from small training sets is error-prone.
- If due only to chance, a rare feature,  $e_k$ , is always false in the training data,  $\forall c_i : P(e_k | c_i) = 0$ .
- If  $e_k$  then occurs in a test example,  $E$ , the result is that  $\forall c_i : P(E | c_i) = 0$  and  $\forall c_i : P(c_i | E) = 0$

## Naïve Bayes for Text

- Modeled as generating a bag of words for a document in a given category by repeatedly sampling with replacement from a vocabulary  $V = \{w_1, w_2, \dots, w_m\}$  based on the probabilities  $P(w_j | c_i)$ .
- Smooth probability estimates with Laplace  $m$ -estimates assuming a uniform distribution over all words ( $p = 1/|V|$ ) and  $m = |V|$   
*Laplace correction* to deal with missing words  
→ 0 probabilities
  - ✓ Equivalent to a virtual sample of seeing each word in each category exactly once.

## Text Naïve Bayes Algorithm (Train)

Let  $V$  be the vocabulary of all words in the documents in  $D$   
For each category  $c_i \in C$   
Let  $D_i$  be the subset of documents in  $D$  in category  $c_i$   
 $P(c_i) = |D_i| / |D|$   
Let  $T_i$  be the concatenation of all the documents in  $D_i$   
Let  $n_i$  be the total number of word occurrences in  $T_i$   
For each word  $w_j \in V$   
Let  $n_{ij}$  be the number of occurrences of  $w_j$  in  $T_i$   
Let  $P(w_j | c_i) = (n_{ij} + 1) / (n_i + |V|)$

## Text Naïve Bayes Algorithm (Test)

Given a test document  $x$   
Let  $n$  be the number of word occurrences in  $x$   
Return the category:

$$\operatorname{argmax}_{c_i \in \mathcal{C}} P(c_i) \prod_{i=1}^n P(a_i | c_i)$$

where  $a_i$  is the word occurring the  $i$ -th position in  $x$

## Underflow Prevention

- 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.

## Naïve Bayes Posterior Probabilities

- Classification results of naïve Bayes (the class with maximum posterior probability) are usually fairly accurate.
- However, due to the inadequacy of the conditional independence assumption, the actual posterior-probability numerical estimates are not.
  - ✓ Output probabilities are generally very close to 0 or 1.

## Valutazione di un classificatore (1)

- La valutazione è sperimentale perché il problema non ha una specifica formale che consenta un altro tipo di valutazione
- La valutazione deve essere effettuata su dati di test indipendenti dai dati di training (solitamente insiemi disgiunti di istanze)
- *Classification accuracy*:  $c/n$ , dove  $n$  è il numero totale di istanze di testing e  $c$  è il numero di istanze di testing classificate correttamente dal sistema
- I risultati possono variare in base all'uso di diversi training e testing sets
- Solitamente si mediano i risultati ottenuti su diversi training and test sets ("splittando" l'insieme iniziale dei dati) per ottenere risultati più significativi

## Valutazione di un classificatore (2)

- Le misure standard di efficacia per i classificatori sono tratte dall' I.R.: *precision* ( $\pi$ ) e *recall* ( $\rho$ )
- $\pi$  e  $\rho$  sono ottenuti mediando su tutte le categorie
- $\pi$  può essere pensata come la "correttezza" del classificatore,  $\rho$  invece come la sua "completezza"

Category $c_i$		expert judgments	
		YES	NO
classifier	YES	$TP_i$	$FP_i$
judgments	NO	$FN_i$	$TN_i$

Category set $\mathcal{C} = \{c_1, \dots, c_{ \mathcal{C} }\}$		expert judgments	
		YES	NO
classifier	YES	$TP = \sum_{i=1}^{ \mathcal{C} } TP_i$	$FP = \sum_{i=1}^{ \mathcal{C} } FP_i$
judgments	NO	$FN = \sum_{i=1}^{ \mathcal{C} } FN_i$	$TN = \sum_{i=1}^{ \mathcal{C} } TN_i$

$$\hat{\pi}_i = \frac{TP_i}{TP_i + FP_i}$$

$$\hat{\rho}_i = \frac{TP_i}{TP_i + FN_i}$$

Text Categorization

49

## Valutazione di un classificatore (3)

- **Microaveraging**

$$\hat{\pi}^\mu = \frac{TP}{TP + FP} = \frac{\sum_{i=1}^{|\mathcal{C}|} TP_i}{\sum_{i=1}^{|\mathcal{C}|} (TP_i + FP_i)}$$

$$\hat{\rho}^\mu = \frac{TP}{TP + FN} = \frac{\sum_{i=1}^{|\mathcal{C}|} TP_i}{\sum_{i=1}^{|\mathcal{C}|} (TP_i + FN_i)}$$

- **Macroaveraging**

$$\hat{\pi}^M = \frac{\sum_{i=1}^{|\mathcal{C}|} \hat{\pi}_i}{|\mathcal{C}|} \quad \hat{\rho}^M = \frac{\sum_{i=1}^{|\mathcal{C}|} \hat{\rho}_i}{|\mathcal{C}|}$$

Text Categorization

50

## Valutazione di un classificatore (1)

- La valutazione è sperimentale perché il problema non ha una specifica formale che consenta un altro tipo di valutazione
- La valutazione deve essere effettuata su dati di test indipendenti dai dati di training (solitamente insiemi disgiunti di istanze)
- *Classification accuracy*:  $c/n$ , dove  $n$  è il numero totale di istanze di testing e  $c$  è il numero di istanze di testing classificate correttamente dal sistema
- I risultati possono variare in base all'uso di diversi training e testing sets
- Solitamente si mediano i risultati ottenuti su diversi training and test sets ("splittando" l'insieme iniziale dei dati) per ottenere risultati più significativi

Text Categorization

51

## K-Fold Cross-Validation (1)

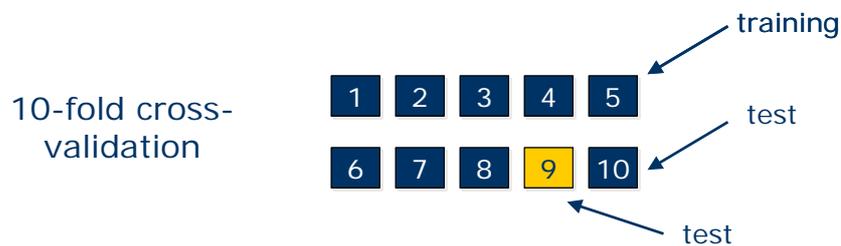
- Teoricamente, test and training sets devono essere indipendenti ad ogni prova (trial).
  - ✓ Richiede una notevole quantità di dati etichettati
- Partiziona i dati in  $K$  insiemi disgiunti.
- Esegue  $K$  prove: ogni prova usa un diverso fold (insieme) per il testing, addestrando il sistema sugli altri  $K-1$  insiemi.
- Le metriche di valutazione sono mediate sulle  $K$  prove
- La procedura garantisce l'indipendenza
- Tipicamente,  $K = 10$

Text Categorization

52

## K-fold cross validation (2)

- Partiziono  $D$  ottenendo  $K$  insiemi  $Te_i$
- Applico iterativamente il train-and-test a  $\langle Tr_i = D - Te_i, Te_i \rangle$
- La effectiveness del sistema è ottenuta mediando quella dei  $K$  classificatori



Text Categorization

53

## Curve di Learning

- Nella pratica, ottenere dataset etichettati non è facile
- È interessante osservare come varia l'accuratezza predittiva di un classificatore in base al numero di istanze usate nella fase di learning
- *Le curve di apprendimento (learning curves)* rappresentano la classification accuracy su dati di test indipendenti (asse  $Y$ ) rispetto al numero di esempi di training (asse  $X$ )

Text Categorization

54

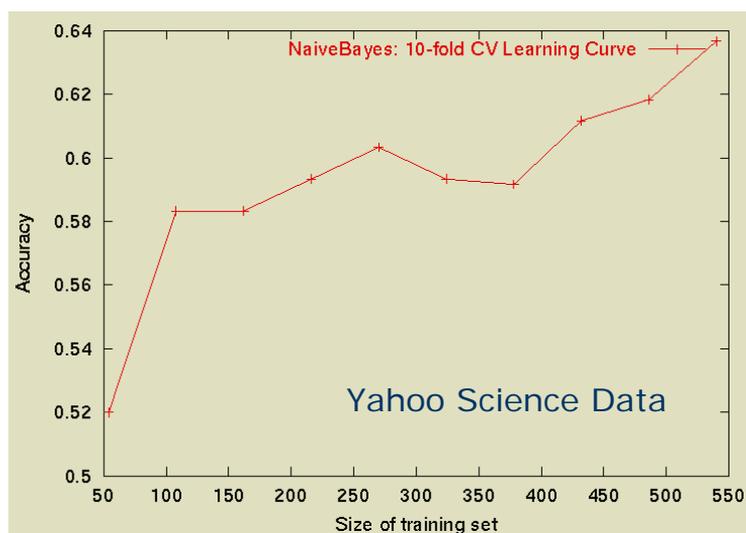
## K-Fold Learning Curves

- Curve di learning mediate sulle varie prove indipendenti
- *K-fold cross validation: K prove*
- Per ogni prova, si addestra il classificatore su frazioni del training set con un numero crescente di istanze, misurando l'accuratezza sui dati di test per ogni punto della curva di learning desiderata

Text Categorization

55

## Sample Learning Curve



Text Categorization

56

## Applications of ML & TC (& IR) (1)

### **SMART** by Salton (Cornell Univ.)

**Goal:** Information Retrieval

**Features:**

- Vector Space model
- Stopword removal
- Stemming
- TF-IDF weighting
- CosSim metric
- Relevance feedback

Salton & McGill, "**Introduction to Modern Information Retrieval**", McGraw-Hill, 1983.

<http://www.cs.cornell.edu/Info/Department/Annual95/Faculty/Salton.html>

Text Categorization

57

## Applications of ML & TC (& IR) (2)

### **Remembrance Agent** by Rhodes & Starner (MIT)

**Goal:**

Remind user of files, old emails and references deemed relevant to what the user is currently writing or reading (Emacs plug-in).

B.J. Rhodes & T. Starner, Remembrance Agent: A continuously running automated information retrieval system, *Proc. 1st Int. Conf. on The Practical Application Of Intelligent Agents and Multi Agent Technology (PAAM '96)*, 487-495, 1996.  
<http://www.bradleyrhodes.com/Papers/remembrance.html>

**Download:** <http://www.remem.org/>

### **LYCOS**

**Goal:** Search engine (find relevant documents on WWW)

<http://www.lycos.com>

Text Categorization

58

## Applications of ML & TC (& IR) (3)

**LIRA** by Balabanovic & Shoham (Stanford Univ.)

**Goal:** Search the web

**Features:**

- Document ranking
- Selection of the *best p* pages
- Receive an updated evaluation for each page presented
- Update the search and selection heuristics
- Relevance feedback (Rocchio)

M. Balabanovic & Y. Shoham, Learning information retrieval agents: Experiments with automated Web browsing, Proc. 1995 AAAI Spring Symp. on Information Gathering from Heterogeneous, Distributed Environments, Stanford, 1995.

<http://robotics.stanford.edu/users/marko/lira/demo1.html>

Text Categorization

59

## Applications of ML & TC (& IR) (4)

**LETIZIA** by Henry Lieberman (MIT)

**Goal:** (page) Recommender

**Features:**

- Assist WWW browsing
- Recommends pages to visit that are accessible through links in the current web page
- Learns user likes (and dislikes) from his/her behavior
- (Resource limited) Breadth-first search

<http://web.media.mit.edu/~lieber/Lieberary/Letizia/Letizia-Intro.html>

Demo: <http://web.media.mit.edu/~lieber/Lieberary/Letizia/Letizia.mov>

Text Categorization

60

## Applications of ML & TC (& IR) (5)

### WEBDOGGIE (WEBHOUND)

by P. Maes and Y. Lashkari (MIT)

**Goal:** Collaborative Filtering of Web pages

**Features:**

- Users (a lot of) rate Web pages
- Pages recommended to you if they are liked by people with similar interests (CF or Social Filtering)

<http://webhound.www.media.mit.edu/projects/webhound/doc/webhound.html>

Text Categorization

61

## Applications of ML & TC (& IR) (6)

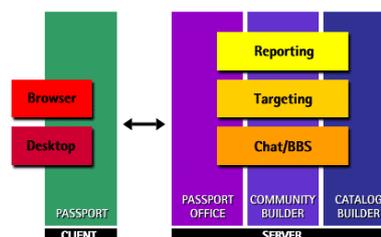
### FIREFLY by P. Maes (MIT)

**Goal:** Recommender

**Features:**

- Recommends albums, movies, artists,...
- Implements Collaborative Filtering
- Free service to users
- Supported by entertainment advertising and sales
- Internet startup FireFly Network Inc. purchased by Microsoft (~\$40 million)

3 main components:  
[Firefly Passport Office™](#),  
[Firefly Community Navigator™](#), and  
[Firefly Catalog Navigator™](#)



Text Categorization

62

## Applications of ML & TC (& IR) (7)

### WebWatcher

by D. Freitag, T. Joachims & T. Mitchell (CMU)

**Goal:** A "tour guide" agent for the Web

**Features:**

- Learns to become a specialist at a particular web site
- **Personal WebWatcher** stays with a single user, becoming a specialist in that user's interests
- User tells what kind of info. is seeking
- Accompanies the user as he/she browses the web
- Highlights links that it believes relevant for the user
- Learns from feedback
- >8500 users

<http://www-2.cs.cmu.edu/afs/cs/project/theo-6/web-agent/www/project-home.html>

Demo: <http://www-2.cs.cmu.edu/afs/cs/project/theo-6/web-agent/www/wwdemo.html>

Text Categorization

63

## Applications of ML & TC (& IR) (8)

### NewsWeeder by Lang & Mitchell (CMU)

**Goal:** Netnews filtering

**Features:**

- User rates each article read from 1 to 5
- Mosaic-based interface
- Learns a user profile based on the ratings
- Exploits user profile to find unread news interesting for the user

With WebWatcher, part of the **Pleiades** project

<http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-5/www/pleiades.html>

Text Categorization

64

## Applications of ML & TC (& IR) (9)

### Syskill & Webert

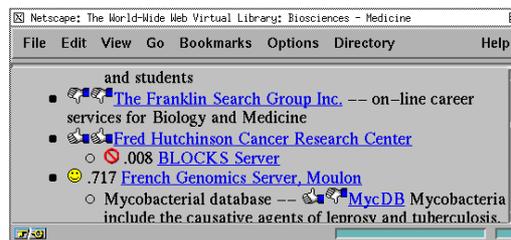
by M. Pazzani, J. Muramatsu & D. Billsus (UCI)

**Goal:** Ranking Web pages

**Features:**

- User rates pages on a 3-point scale
- Learns a user profile by analyzing page content
- Exploits user profile 1) to suggest which links a user would be interested in exploring, 2) to construct a proper LYCOS query to find pages that would interest a user
- Rank pages accessible from the current page (search engine result page, too)

<http://www.ics.uci.edu/~pazzani/Syskill.html>



Text Categorization

65

## Applications of ML & TC (& IR) (10)

### DICA by

Starr, Ackerman & Pazzani (UCI)

**Goal:** Web site monitoring

**Features:**

- Track relevant changes to your favourite web sites
- Topic driven
- Learns a profile for each topic

<http://www.uci.edu/~pazzani>

Text Categorization

66

## Applications of ML & TC (& IR) <sup>(11)</sup>

### InfoFinder by B. Krulwich (Andersen Consulting)

**Goal:** Find interesting web pages

**Features:**

- Take in input your interest
- Learns a decision tree
- Boolean retrieval model (Boolean query)
- Direct + and – feedback from the user

<http://www.ac.com/cstar/hsil/agents/>

## References

- M. Ackerman, D. Billsus, S. Gaffney, S. Hettich, G. Khoo, D. J. Kim, R. Klefstad, C. Lowe, A. Ludeman, J. Muramatsu, K. Omori, M. Pazzani, D. Semler, B. Starr & P. Yap, Learning Probabilistic User Profiles: Applications for Finding Interesting Web Sites, Notifying Users of Relevant Changes to Web Pages, and Locating Grant Opportunities, *AI Magazine* 18(2):47-56, 1997.
- M. Balabanovic & Y. Shoham, Learning information retrieval agents: Experiments with automated Web browsing, *Proc. of the 1995 AAAI Spring Symp. on Information Gathering from Heterogeneous, Distributed Environments*, Stanford, 1995.
- Firefly Network, Inc. 1997. *Collaborative Filtering Technology: An Overview*. <http://www.firefly.net/company/CollaborativeFiltering.fly>
- T. Joachims, D. Freitag & T. M. Mitchell, Web Watcher: A Tour Guide for the World Wide Web, *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI 97)*, 770-777, Nagoya, Japan, August 23-29, Morgan Kaufmann, 1997.
- B. Krulwich & C. Burkey, The InfoFinder Agent: Learning User Interests through Heuristic Phrase Extraction, *IEEE Expert* 12(5):22-27, 1997.
- Y. Lashkari, M. Metral & P. Maes: Collaborative Interface Agents. *Proc. of the 12th National Conf. on Artificial Intelligence (AAAI 1994)*, 444-449, Seattle, WA, USA, July 31 - August 4, AAAI Press.
- H. Lieberman, Letizia: An Agent That Assists Web Browsing, *Proc. of the 14th Int. Joint Conference on Artificial Intelligence (IJCAI 95)*, 924-929, Montréal, Québec, Canada, August 20-25, Morgan Kaufmann, 1995.
- T.M. Mitchell, "**Machine Learning**", McGraw-Hill, 1997.
- P. Maes, Intelligent Software: Easing the Burdens that Computers Put on People. *IEEE Expert*, 11(6) 62-63, 1996.
- P. Maes, *Information filtering system. Overview, slides*. Available at: [http://infoweb.vub.ac.be/~lasse.software\\_agents/infofiltering/sld002.htm](http://infoweb.vub.ac.be/~lasse.software_agents/infofiltering/sld002.htm).
- M. Pazzani, "**Machine Learning and Information Filtering on the Internet**", IJCAI-97 Tutorial, Nagoya, Japan, Aug 1997.
- M. Pazzani, J. Muramatsu & D. Billsus, Syskill & Webert: Identifying Interesting Web Sites, *Proc. of the 13th National Conf. on Artificial Intelligence and 8th Innovative Applications of Artificial Intelligence Conf. (AAAI 96/IAAI 96)*, 54-61, August 4-8, Portland, Oregon, AAAI Press/The MIT Press, 1996.
- B.J. Rhodes & T. Starner, Remembrance Agent: A continuously running automated information retrieval system, *Proc. of the 1st Int. Conf. on The Practical Application Of Intelligent Agents and Multi Agent Technology (PAAM '96)*, 487-495, 1996.
- Salton, G., & McGill, M. J. "**Introduction to Modern Information Retrieval**", McGraw-Hill, 1983.
- F. Sebastiani, "**Machine Learning in Automated Text Categorization**", *ACM Computing Surveys*, 34(1):1-47, 2002.

## Credits

- Prof. Mooney, Professor of Computer Science, University of Texas, Austin