Preference Learning

Modelli e Metodi

Linee Guida dell'Esposizione

Contenuti

1. **Introduzione**
   - *Need for Preferences*
   - *Label Preferences*
   - *Object Preferences*

2. **Learning Label Preferences**
   - *Constraint Classification*
   - *Pairwise Preference Ranking*

3. **Learning Object Preferences**
   - *Comparison Training*
   - *Order Things*
   - *PRank*

4. **Approcci Alternativi**
   - *Sets of Objects*
   - *GPLM*

5. **Evaluating Rankings**
   - *Kendall's Tau e Spearman Rank Correlation*
   - *Valutazione Empirica*

Introduzione

Need for Preferences

- **Preferenza vs Equivalenza**
  - a $\succ$ b → "l'alternativa a è preferibile all'alternativa b"
- **Preferenze singole e di gruppo**
- **Classi di problemi**
  - Label e Object
- **Approcci**
  - Relazionale → *pairwise*
  - Numerico → *utility function* ($a \succ b \Leftrightarrow f(a) \geq f(b)$)

Introduzione

Need for Preferences

Considerazioni

– Utility function → assume *modello numerico implicito*

– Una funzione di utilità induce una relazione di preferenza che impone un *ordine totale*, ma una relazione di preferenza non necessariamente induce un ordine totale (transitività)

– Predizione assume un *ordinamento totale* ma informazioni spesso incomplete e/o inconsistenti

		Approcci	
		Utility Function	*Pairwise Preference*
Modelli	*Learning Label Preference*	**Constraint classification**	**Pairwise** preference learning
	Learning Object Preference	**Comparison training**	**Learning to order things**

Introduzione

Problema

Label Preferences

Per ogni istanza x da uno *spazio delle istanze* X, apprendere una *relazione di preferenza* $\wp_x \subseteq \mathcal{L}x\mathcal{L}$ su un insieme finito di *label* (o alternative) λ.

$(\lambda_1,\lambda_2) \in \wp_x$ → l'istanza x preferisce λ_1 a λ_2

Training Set = insieme di istanze con conoscenza parziale della relazione di preferenza

Dati

- Un insieme di *training instances* $\{ x_k \mid k=1,\ldots,n \} \subseteq X$
- Un insieme finito di *label* $\mathcal{L}=\{\lambda_i \mid i=1,\ldots,c\}$
- $\forall x_k \in Tr$, un insieme di *pairwise pref.* $(\lambda_i \succ_{X_k} \lambda_j)$

Trovare

- Una funzione per predire $\wp_x \subseteq \mathcal{L}x\mathcal{L}$

Generalizzazione di: *Classification, Multiclass-Classific.* e *Ranking*

Introduzione

Problema

Object Preferences

Per ogni sottoinsieme *O* di una classe *C* di oggetti (es. pagine ritornate da una *search engine*), apprendere una funzione di ordinamento attraverso la quale indurre un *ranking*

Training Set = insieme di *pairwise pref.* tra istanze

Dati

– Un insieme di *training instances* $D=\{ o_k \mid k=1,\ldots,n \} \subseteq C$

– Un insieme di *pairwise pref.* $\wp \subseteq D \times D$, dove $(o_i,o_j) \in \wp \rightarrow o_i \succ o_j$

Trovare

– Una funzione di ordinamento per ogni sottoinsieme di oggetti $O \subseteq C$

Learning Label Preferences

Constraint Classification

Framework proposto da S.Har-Peld et al. (2002)

Relazioni di preferenza → **vincolo qualitativo** sulla funzione di predizione ($\lambda_i \succ \lambda_j$)

Ad ogni esempio è associato un insieme di vincoli su più classi: ogni vincolo specifica l'ordine relativo tra coppie di classi, per l'esempio.

Esempio (x,y) dove y è un'ordine parziale sulle possibili *label* (|y| ≤ c) con c=4
(x,{(2,3),(2,4)})

Obiettivo → apprendere un classificatore maggiormente consistente con tali vincoli

Sia $W = \{w_1,...,w_c \mid w_i \in \Re^d, i = 1,...,c\}$. Data un istanza $x \in \Re^d$, un *classificatore lineare* $h : \Re^d \to S^k$ (permutazione delle k *label*) può essere calcolato come:

$$h(x) = \underset{i=1,...,c}{\arg sort}\, w_i \cdot x$$

Dove *argsort* ritorna una permutazione di {1,..,c} dove *i* precede *j* se $w_i \cdot x > w_j \cdot x$

Learning Label Preferences

Constraint Classification

L'apprendimento è possibile trasformando il problema in un problema di classificazione binaria in uno spazio di dimensione maggiore.
Ogni esempio $(x, y) \in \Re^d \times \overline{S}^c$ diventa un insieme di esempi in $\Re^{cd} \times \{-1,1\}$ in cui ogni vincolo (i,j) contribuisce con un esempio positivo ed uno negativo.

Esempio: $\mathcal{L}=\{1,2,3\}, x \in \Re^n$

(x,y)=(x,{(2,4),(2,3)}) ➔ $x_{POS}^{(2,4)} = [0...0, x_1...x_n, 0...0, -x_{1...} - x_n, 0...0]^{\mathrm{T}} \in \Re^{4n}$

$x_{NEG}^{(2,4)} = [0...0, -x_1... - x_n, 0...0, x_{1...} x_n, 0...0]^{\mathrm{T}} \in \Re^{4n}$

Il framework generalizza i problemi di multilabel-classification, label ranking e binary classification

Problem	Internal Representation	Output Space ($\mathcal{Y}$)	Hypothesis	Size of Mapping
binary	$w \in \mathbb{R}^d$	$\{-1, 1\}$	$\operatorname{sign} w \cdot x$	1
multiclass	$(w_1, \ldots, w_k) \in \mathbb{R}^{kd}$	$\{1, \ldots, k\}$	$\operatorname{argmax}_{\{1,\ldots,k\}} w_i \cdot x$	$k-1$
l-multilabel	$(w_1, \ldots, w_k) \in \mathbb{R}^{kd}$	$\{1, \ldots, k\}^l$	$\operatorname{argmax}^l_{\{1,\ldots,k\}} w_i \cdot x$	$l(k-l)$
ranking	$(w_1, \ldots, w_k) \in \mathbb{R}^{kd}$	S^k	$\operatorname{argsort}_{\{1,\ldots,k\}} w_i \cdot x$	$k-1$
constraint*	$(w_1, \ldots, w_k) \in \mathbb{R}^{kd}$	$\bar{S}^k$	$\operatorname{argsort}_{\{1,\ldots,k\}} w_i \cdot x$	–
c-constraint*	$(w_1, \ldots, w_k) \in \mathbb{R}^{kd}$	$\bar{S}^k_c$	$\operatorname{argsort}_{\{1,\ldots,k\}} w_i \cdot x$	c

Table 1: Definitions for various learning problems (notice that the hypothesis for constraint classification is always a full order) and the size of the resultant mapping to c-constraint classification. argmax^l is a variant of argmax that returns the l maximal indices with respect to $w_i \cdot x$. argsort is a linear sorting function (see Definition 2.6).

Learning Label Preferences

Pairwise Preference Learning

Framework proposto da J.Furnkranz, E.Hullermeier (2003)
In realtà è un'estensione di una soluzione al multilabel-classification

Per ogni possibile coppia di *label* (λ_i, λ_j) con $1 \leq i < j \leq c$ → **M_{ij}** che decide per ogni esempio se $\lambda_i \succ \lambda_j$ o $\lambda_j \succ \lambda_i$
Il modello viene quindi allenato con tutti gli esempi x_k in Tr per cui $\lambda_i \succ_{X_k} \lambda_j$ o $\lambda_j \succ_{X_k} \lambda_i$

Classificazione: un esempio viene sottoposto a tutti i modelli: se il modello M_{ij} predice che $\lambda_i \succ \lambda_j$ allora viene assegnato un voto per λ_i.
→ $M_{ij}(x)=1$ se $\lambda_i \succ \lambda_j$,0 altrimenti. (è possibile anche *soft* in [0,1])

Viene indotta una *relazione di preferenza* $R_x(\lambda_i, \lambda_j) = \begin{cases} M_{ij}(x) & se \quad i < j \\ 1 - M_{ij} & se \quad i > j \end{cases}$

In un problema di multilabel-classifcation viene predetta la *label* con il maggior numero di voti.

Per indurre un ranking sulle label → ordinate in base al numero di voti (*voting*)

$$S(\lambda_i) = \sum_{\lambda_i \neq \lambda_j} R_x(\lambda_i, \lambda_j) \quad \text{e} \quad (\lambda_i \succ \lambda_j) \Leftrightarrow (S(\lambda_i) > S(\lambda_j))$$

Sono necessari c(c-1)/2 modelli

Learning Object Preferences

Comparison Training

Approccio proposto da G.Tesauro (1989)

Si basa su un'architettura *simmetrica* di **reti neurali** che possono essere allenate attraverso la rappresentazione di due stati e un segnale che esprime un giudizio di preferenza tra i due.

Simmetria permette di sostituire le due componenti con un unica rete che restituisce in output valori in $\mathbb{R}$ per i singoli stati.

Anche *Haddavvy* (2003) utilizza dei dati in forma di raffronti tra coppie per allenare una rete neurale che restituisce in output un valore in {0,1} a seconda che il primo elemento della coppia sia preferito al secondo o meno.

Learning Object Preferences

Order Things

Modello definito da W.Cohen et al. (1999)

Spazio delle ipotesi: funzioni di preferenza PREF:$X \times X \rightarrow [0,1]$

PREF(u,v)=r se $r \sim 1$ u preferibile a v

se $r \sim 0$ v preferibile a u

se $r \sim ½$ astensione

Si assume PREF combinazione lineare di N funzioni di preferenza primitive $\mathbf{R_1},\ldots,\mathbf{R_N}$

Sia f:X→S (insieme totalmente ordinato con >) *ordering function into S* che induce la relazione

$$R_{f_i}(u,v) = \begin{cases} 1 & se \quad f_i(u) > f_i(v) \\ 0 & se \quad f_i(u) < f_i(v) \\ ½ & se \quad f_i(u) = \phi \ o \ f_i(v) = \phi \end{cases}$$

ϕ =*unranked*

è un *rank ordering* per *X* su *S*

$$PREF(u,v) = \sum_{i=1}^{N} w_i R_{f_i}(u,v)$$

dove *w* è un vettore dei pesi $(w_1,\ldots,w_N)$ in $[0,1]^N$ con $\Sigma\, w_i = 1$

L'ordinamento ricercato è quello che massimizza AGREE(ρ,PREF) definita come

$$AGREE(\rho, PREF) = \sum_{u,v:\rho(u)>\rho(v)} PREF(u,v)$$

Learning Object Preferences

Order Things

Il problema è NP-Complete se $|S| \geq 3$

Se $|S|=2$ (i.e. $S=\{0,1\}$) allora l'ordinamento ρ tale che $\rho(u)= \Sigma_i w_i f_i(u)$ massimizza AGREE(ρ,PREF)

Calcolo del Vettore dei Pesi

Apprendimento iterativo: ad ogni iterazione si presume di ricevere da un esperto un insieme di istanze X e un insieme di N funzioni di preferenza R_i.

L'algoritmo di apprendimento calcola $R_i(u,v)$ per ogni funzione di preferenza e ogni coppia di istanze.

A questo punto si presume di ricevere un feedback dall'utente; sulla base del quale viene aggiornato il vettore dei pesi

$$w_i^{t+1} = \frac{w_i^t \cdot \beta^{Loss(R_i^t F^t)}}{Z_t} \qquad Loss(R,F) \overset{def}{=} \frac{\sum_{u,v \in F} (1 - R(u,v))}{|F|}$$

(con $\beta \in [0,1]$ e Z_t normalizzatore)

Ordinamento

Algoritmo greedy che utilizza un grafo in cui i nodi sono le istanze; ad ogni nodo v è associato un peso determinato da PREF(v,u)-PREF(u,v) per ogni u.

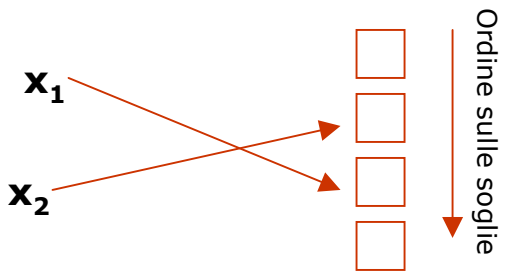
Learning Object Preferences

PRank

Modello definito da K.Crammer e Y.Singer (1998)

Dato un insieme di coopie istanza-ranking $(\vec{x}^1, y^1),\ldots,(\vec{x}^t, y^t)$
e y^t appartiene ad un insieme finito totalmente ordinato $Y_>=\{1,2,\ldots,k\}$

L'ordine totale su *Y* induce un ordine parziale sulle istanze:



Definiamo *H* come *ranking rule* se è una funzione che mappa le istanze sul rank: $H : \Re^n \rightarrow Y$

H utilizza un vettore di pesi $w \in \Re^n$ e un insieme di soglie $b = \{b_1 \leq b_2 \leq \ldots \leq b_{k-1}\}$

$$H(\vec{x}) = \min_{r\in\{1,\ldots,k\}} \{\vec{w}\cdot\vec{x} - b_r < 0\}$$

Learning Object Preferences

Algoritmo PRank

Initialize: Set $\mathbf{w}^1 = 0$, $b_1^1, \ldots, b_{k-1}^1 = 0, b_k^1 = \infty$.
Loop: For $t = 1, 2, \ldots, T$

- Get a new rank-value $\mathbf{x}^t \in \mathbb{R}^n$.
- Predict $\hat{y}^t = \min_{r \in \{1,\ldots,k\}} \{r : \mathbf{w}^t \cdot \mathbf{x}^t - b_r^t < 0\}$.
- Get a new label y^t.
- If $\hat{y}^t \neq y^t$ update $\mathbf{w}^t$ (otherwise set $\mathbf{w}^{t+1} = \mathbf{w}^t$, $\forall r : b_r^{t+1} = b_r^t$) :
  1. For $r = 1, \ldots, k-1$: If $y^t \leq r$ Then $y_r^t = -1$ Else $y_r^t = 1$.
  2. For $r = 1, \ldots, k-1$: If $(\mathbf{w}^t \cdot \mathbf{x}^t - b_r^t) y_r^t \leq 0$ Then $\tau_r^t = y_r^t$ Else $\tau_r^t = 0$.
  3. Update $\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t + (\sum_r \tau_r^t)\mathbf{x}^t$. For $r = 1, \ldots, k-1$ update: $b_r^{t+1} \leftarrow b_r^t - \tau_r^t$

Output : $H(\mathbf{x}) = \min_{r \in \{1,\ldots,k\}} \{r : \mathbf{w}^{T+1} \cdot \mathbf{x} - b_r^{T+1} < 0\}$.

Figure 2: The PRank algorithm.

Il rank è suddiviso in k intervalli virtuali:
→Una predizione è corretta se $y_r(\boldsymbol{w}\cdot\boldsymbol{x}-b_r)>0$ altrimenti aggiorna il vettore dei pesi e il valore delle soglie in modo da "avvicinare" i valori di $\boldsymbol{w}\cdot\boldsymbol{x}$ e b_r

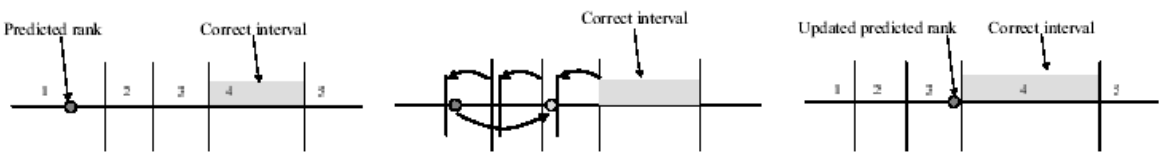


Figure 1: An Illustration of the update rule.

Approcci Alternativi

Sets of Objects

Modello definito da M.desJardins, E.Eaton (2006)

Approccio particolare basato sulla intuizione che a volte un utente può essere interessato non a un insieme rilevante di risultati ma ad un **insieme ottimo** di risultati.

Es: playlist musicale o dratf NBA/NFL

Selezionare i primi k elementi di un insieme rilevante non tiene conto che gli oggetti di un insieme possono **interagire** tra loro (effetto portfolio) in modo da:
– *accrescere* (per complementarietà) la rilevanza globale
– *decrescere* (per incompatibilità o ridondanza) la rilevanza globale

L'approccio delineato propone un modello di apprendimento basandosi su un Tr che esprime implicitamente le preferenze sugli insiemi.
Sono introdotte delle misure di:
- *qualità* (per ogni singola feature)
- *diversità* (misura della diversità desiderata tra gli elementi dell'insieme)
- *peso* associato alle diverse feature
- *importanza relativa* della diversità rispetto alla qualità

Approcci Alternativi

GPLM

Framework definito da F.Aiolli (2005)

Modello generalizzato già accennato a lezione, che estende l'approccio Constraint Classifcation.

Come l'approccio CC:

- rappresenta le relazioni di preferenza come vincoli sulle possibili predizioni; la violazione di un vincolo implica un costo associato alla predizione

- permette di risolvere problemi di apprendimento delle preferenze riducendoli a problemi di classifcazione binari in uno spazio delle istanze aumentato.

Estende l'approccio CC poiché permette di esprimere congiunzioni di vincoli

- qualitativi → $u(\vec{x}_i, y_r), u(\vec{x}_j, y_s)$
- quantitativi → $(u(\vec{x}, y), \tau)$ *o* $(\tau, u(\vec{x}, y))$ $\tau \in \Re$ (simili soglie in PRank)

Per ogni esempio → un esempio nello spazio aumentato per ogni *congiunto* dei vincoli

$u(d, y) = \vec{w} \cdot \phi(d, y)$

$\phi(d, y)$ = rappresentazione congiunta dell'esempio delle coppie istanza-label

Approcci Alternativi

GPLM

Il metodo permette di modellare anche i problemi di pairwise learning

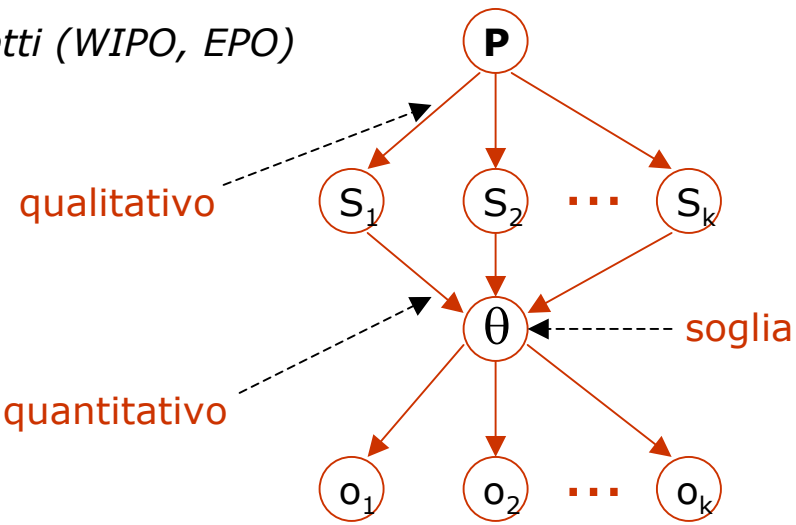
GPLM ***Pairwise Pref. Leraning***

R_1=(2,3,1,4)

R_2=(2,1,3,4)

Applicazione pratica: *Classificazione Brevetti (WIPO, EPO)*

(brev,(P,S_1,S_2,…,S_k))



Evaluating Rankings

Alcune precisazioni su come valutare la consistenza tra *ranking*

→In tutti i casi in cui si utilizza predice un ordine totale o si produce comunque un ordinamento è necessario valutarne la misura di consistenza con l'ordinamento indotto dalle preferenze sugli esempi (vedi ad es. AGREE in Order Things)

Kendall's Tau

Il *coefficiente di correlazione di Kendall* (1938) è un coefficiente utilizzato appunto per valutare il grado di *corrispondenza* tra due *ranking* R_a *e* R_b sullo stesso dominio *D*

Proprietà:

- se la corrispondenza tra R_a e R_b è perfetta (i.e. $R_a = R_b$) Kendall's tau=1

- se la incorrispondenza tra R_a e R_b è perfetta (i.e. $R_a = R_b^{-1}$) Kendall's tau=-1

- per tutti gli altri casi il coefficiente varia in [-1,1]; se R_a e R_b sono indipendenti (i.e. non hanno elementi in comune) Kendall's tau=0

$$\tau = \frac{2P}{\frac{1}{2}n(n-1)} - 1 = \frac{4P}{n(n-1)} - 1$$

dove n=numero istanze, P=somma su tutte le istanze, delle istanze classificate dopo una data istanza in entrambi gli ordinamenti

Evaluating Rankings

Kendall's Tau

Il *Kendall Tau Distance* tra R_a e R_b,**ordini totali** sullo stesso dominio *D*

$$K(R_a, R_b) = \sum_{(i,j)\in P} \overline{K}_{ij}(R_a, R_b)$$

Dove: P={(i,j)|i ≠ j e i,j∈D} e $\overline{K}_{ij}(R_a, R_b) = \begin{cases} 0 & se\ i\ e\ j\ sono\ nello\ stesso\ ordine\ in\ R_a\ e\ in\ R_b \\ 1 & altrimenti \end{cases}$

Variante **P-Kendall's Tau** (Partial Order) → tra R_a e R_b,**ordini parziali** sul dominio *D*

Valuta i casi di *incomparabilità* tra istanze secondo un <u>parametro di penalizzazione *p*</u>

$$K^p(R_a, R_b) = \sum_{(i,j)\in P} \overline{K}^p_{ij}(R_a, R_b)$$

Dove: P={(i,j)|i ≠ j e i,j∈D}, 0<p≤1

$$\overline{K}^p_{ij}(R_a, R_b) = \begin{cases} i\ e\ j\ sono\ comparabili\ sia\ in\ R_a\ sia\ in\ R_b & \begin{matrix} 0 & stesso\ ordine \\ 1 & altrimenti \end{matrix} \\ i\ e\ j\ sono\ incomparabili\ sia\ in\ R_a\ sia\ in\ R_b & 0 \\ i\ e\ j\ sono\ incomparabili\ in\ uno\ dei\ due\ ordinamenti & p \end{cases}$$

Evaluating Rankings

Spearman's Rho

Il *coefficiente di correlazione di Spearman* è una misura di correlazione che verifica quanto sia possibile esprimere la relazione tra due *ranking* R_a *e* R_b attraverso una funzione monotona.

ρ varia in [-1,1] come il coefficiente di correlazione Kendall's Tau

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

(non considera i casi di incomparabilità)

dove n=numero di coppie di istanze, d_i=differenza nei due rank dei valori assegnati alle istanze che compongono una coppia

$$\rho = \frac{\sum (rank_a - (n+1)/2)(rank_b - (n+1)/2)}{n(n-1)(n+1)/12}$$

Evaluating Rankings

Valutazione Empirica

Joachims propone un modello empirico di valutazione dei ranking ottenuti basato sulla considerazione che l'informazione è data implicitamente dall'utente che seleziona alcuni tra i link visualizzati come risultato della query mentre ne trascura altri.

Informazione implicita → preferenze binaria con le seguenti assunzioni:
- le pagine selezionate siano preferite alle pagine non selezionate
- l'utente selezioni le pagine più significative

I *click* di un utente siano un informazione facilmente rilevabile che di fatto non impone alcun overhead all'utente, propone un metodo di valutazione dei ranking restituiti da diverse *search engine* sulla base della stessa query

Scenario sperimentale:
- un utente definisce una query, tale query viene sottoposta a due *search engine* A e B;
- l'ordinamento restituito è una *combinazione* dei risultati così che i primi *k* link sono composti dai primi k_a link di A e i primi k_b link di B
- i click dell'utente sono memorizzati.
- se l'utente seleziona più spesso i link restituiti da A è ragionevole dedurre che i primi k link di A siano preferibili ai primi k link di B e il rank di A sia quindi **preferibile**

Riferimenti

Riferimenti Bibliografici

Constraint Classification
S.Har-Peld, D.Roth, D.Zimak – "Constraint Classification for Multiclass Classification and Ranking"

Pairwise Preference Learning
J.Furnkranz, E.Hullermeier – "Pairwise Preference Learning and Ranking"

Order Things
W.Cohen, R.Shapire, Y.Singer – "Learning to Order Things"

Comparison Training
G.Tesauro – "Connectionist Learning of Expert Preferences by Comparison Learning"

PRank
K.Crammer, Y.Singer – "Pranking with Ranking"

Sets of Objects
M.desJardins, E.Eaton – "Learning User Preferences for Sets of Objects"

GPLM
F.Aiolli – "A Preference Model for Structured Supervised Learning Task"

Valutazione Empirica
T.Joachims – "Evaluating Retrieval Performances using ClickThrough Data"