Foundations of Machine Learning
Ranking

Mehryar Mohri
Courant Institute and Google Research
mohri@cims.nyu.edu

Motivation

- Very large data sets:
  - too large to display or process.
  - limited resources, need priorities.
  - $\longrightarrow$ ranking more desirable than classification.
- Applications:
  - search engines, information extraction.
  - decision making, auctions, fraud detection.
- Can we learn to predict ranking accurately?

Related Problem

- Rank aggregation: given n candidates and k voters each giving a ranking of the candidates, find ordering as close as possible to these.
  - closeness measured in number of pairwise misrankings.
  - problem NP-hard even for $k = 4$ (Dwork et al., 2001).

This Talk

- Score-based ranking
- Preference-based ranking

Score-Based Setting

- Single stage: learning algorithm
  - receives labeled sample of pairwise preferences;
  - returns scoring function $h\colon U \to \mathbb{R}$.
- Drawbacks:
  - h induces a linear ordering for full set U.
  - does not match a query-based scenario.
- Advantages:
  - efficient algorithms.
  - good theory: VC bounds, margin bounds, stability bounds (FISS 03, RCMS 05, AN 05, AGHHR 05, CMR 07).

Score-Based Ranking

- **Training data**: sample of i.i.d. labeled pairs drawn from $U \times U$ according to some distribution D,

$$S = \Big((x_1, x'_1, y_1), \ldots, (x_m, x'_m, y_m)\Big) \in U \times U \times \{-1, 0, +1\},$$

with $y_i = \begin{cases} +1 & \text{if } x'_i >_{\text{pref}} x_i \\ 0 & \text{if } x_i =_{\text{pref}} x'_i \text{ or no information} \\ -1 & \text{if } x'_i <_{\text{pref}} x_i. \end{cases}$

- **Problem**: find hypothesis $h: U \to \mathbb{R}$ in H with small generalization error

$$R(h) = \Pr_{(x,x') \sim D} \Big[(f(x, x') \neq 0) \wedge (f(x, x')\big(h(x') - h(x)\big) \leq 0) \Big].$$

Notes

- Empirical error:

$$\widehat{R}(h) = \frac{1}{m} \sum_{i=1}^{m} 1_{(y_i \neq 0) \wedge (y_i(h(x'_i) - h(x_i)) \leq 0)} \,.$$

- The relation $x \,\mathcal{R}\, x' \Leftrightarrow f(x, x') = 1$ may be non-transitive (needs not even be anti-symmetric).
- Problem different from classification.

Distributional Assumptions

- Distribution over points: m points (literature).
  - labels for pairs.
  - $\longrightarrow$ squared number of examples $O(m^2)$.
  - dependency issue.
- Distribution over pairs: m pairs.
  - label for each pair received.
  - independence assumption.
  - same (linear) number of examples.

Confidence Margin in Ranking

- Labels assumed to be in $\{+1, -1\}$.
- Empirical margin loss for ranking: for $\rho > 0$,

$$\widehat{R}_\rho(h) = \frac{1}{m} \sum_{i=1}^{m} \Phi_\rho\Big(y_i\big(h(x'_i) - h(x_i)\big)\Big).$$

$$\widehat{R}_\rho(h) \leq \frac{1}{m} \sum_{i=1}^{m} 1_{y_i[h(x'_i) - h(x_i)] \leq \rho}.$$

Marginal Rademacher Complexities

- Distributions:
  - D_1 marginal distribution with respect to the first element of the pairs;
  - D_2 marginal distribution with respect to second element of the pairs.
- Samples: $S_1 = \big((x_1, y_1), \ldots, (x_m, y_m)\big)$

$$S_2 = \big((x'_1, y_1), \ldots, (x'_m, y_m)\big).$$

- Marginal Rademacher complexities:

$$\mathfrak{R}_m^{D_1}(H) = \mathrm{E}[\widehat{\mathfrak{R}}_{S_1}(H)] \quad \mathfrak{R}_m^{D_2}(H) = \mathrm{E}[\widehat{\mathfrak{R}}_{S_2}(H)].$$

Ranking Margin Bound

(Boyd, Cortes, MM, and Radovanovich 2012; MM, Rostamizadeh, and Talwalkar, 2012)

- Theorem: let H be a family of real-valued functions. Fix $\rho > 0$, then, for any $\delta > 0$, with probability at least $1 - \delta$ over the choice of a sample of size m, the following holds for all $h \in H$:

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho}\left(\mathfrak{R}_m^{D_1}(H) + \mathfrak{R}_m^{D_2}(H)\right) + \sqrt{\frac{\log \frac{1}{\delta}}{2m}}.$$

Ranking with SVMs

see for example (Joachims, 2002)

- Optimization problem: application of SVMs.

$$\min_{\mathbf{w},\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i$$

$$\text{subject to: } y_i\Big[\mathbf{w}\cdot\big(\mathbf{\Phi}(x'_i)-\mathbf{\Phi}(x_i)\big)\Big] \geq 1-\xi_i$$

$$\xi_i \geq 0, \quad \forall i \in [1,m]\,.$$

- Decision function:

$$h\colon x \mapsto \mathbf{w}\cdot\mathbf{\Phi}(x)+b.$$

Notes

- The algorithm coincides with SVMs using feature mapping
$$(x, x') \mapsto \mathbf{\Psi}(x, x') = \mathbf{\Phi}(x') - \mathbf{\Phi}(x).$$
- Can be used with kernels:
$$\begin{aligned} K'((x_i, x'_i), (x_j, x'_j)) &= \mathbf{\Psi}(x_i, x'_i) \cdot \mathbf{\Psi}(x_j, x'_j) \\ &= K(x_i, x_j) + K(x'_i, x'_j) - K(x'_i, x_j) - K(x_i, x'_j). \end{aligned}$$
- Algorithm directly based on margin bound.

Boosting for Ranking

- Use weak ranking algorithm and create stronger ranking algorithm.
- Ensemble method: combine base rankers returned by weak ranking algorithm.
- Finding simple relatively accurate base rankers often not hard.
- How should base rankers be combined?

CD RankBoost

(Freund et al., 2003; Rudin et al., 2005)

$H \subseteq \{0,1\}^X$. $\epsilon_t^0 + \epsilon_t^+ + \epsilon_t^- = 1, \epsilon_t^s(h) = \Pr_{(x,x') \sim D_t}\Big[\text{sgn}\big(f(x,x')(h(x') - h(x))\big) = s\Big]$.

$\text{RankBoost}(S = ((x_1, x'_1, y_1) \ldots, (x_m, x'_m, y_m)))$

$\quad$ **for** $i \leftarrow 1$ **to** m **do**

$\qquad D_1(x_i, x'_i) \leftarrow \frac{1}{m}$

$\quad$ **for** $t \leftarrow 1$ **to** T **do**

$\qquad h_t \leftarrow$ base ranker in H with smallest $\epsilon_t^- - \epsilon_t^+ = -\mathrm{E}_{i \sim D_t}\Big[y_i\big(h_t(x'_i) - h_t(x_i)\big)\Big]$

$\qquad \alpha_t \leftarrow \frac{1}{2} \log \frac{\epsilon_t^+}{\epsilon_t^-}$

$\qquad Z_t \leftarrow \epsilon_t^0 + 2[\epsilon_t^+ \epsilon_t^-]^{\frac{1}{2}}$ $\quad \triangleright$ normalization factor

$\qquad$ **for** $i \leftarrow 1$ **to** m **do**

$\qquad\qquad D_{t+1}(x_i, x'_i) \leftarrow \frac{D_t(x_i, x'_i) \exp\Big[-\alpha_t y_i\big(h_t(x'_i) - h_t(x_i)\big)\Big]}{Z_t}$

$\quad \varphi_T \leftarrow \sum_{t=1}^T \alpha_t h_t$

$\quad$ **return** φ_T

Notes

- Distributions D_t over pairs of sample points:
  - originally uniform.
  - at each round, the weight of a misclassified example is increased.
  - observation: $D_{t+1}(x, x') = \frac{e^{-y[\varphi_t(x') - \varphi_t(x)]}}{|S| \prod_{s=1}^t Z_s}$, since

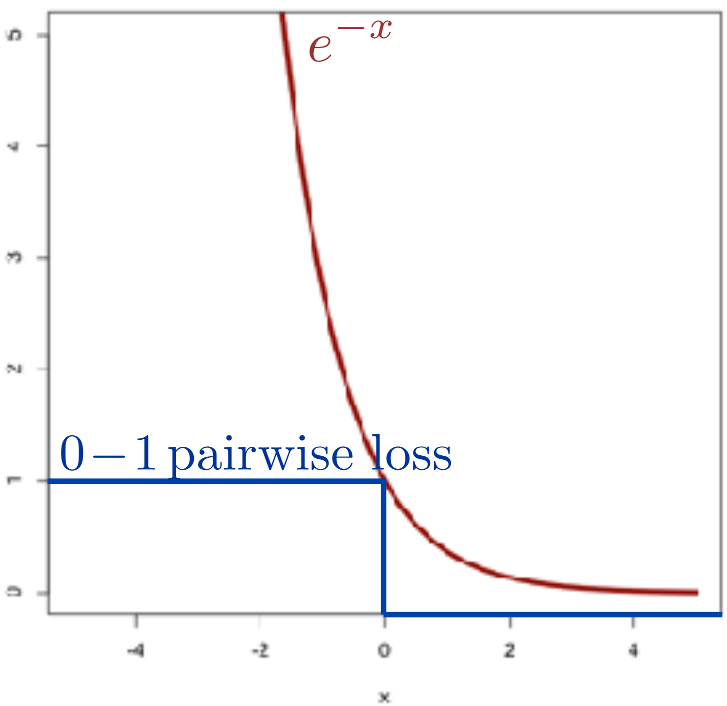
$$D_{t+1}(x, x') = \frac{D_t(x, x') e^{-y\alpha_t[h_t(x') - h_t(x)]}}{Z_t} = \frac{1}{|S|} \frac{e^{-y \sum_{s=1}^t \alpha_s [h_s(x') - h_s(x)]}}{\prod_{s=1}^t Z_s}.$$

- weight assigned to base classifier h_t: α_t directly depends on the accuracy of h_t at round t .

Coordinate Descent RankBoost

- **Objective Function**: convex and differentiable.

$$F(\boldsymbol{\alpha}) = \sum_{(x,x',y)\in S} e^{-y[\varphi_T(x') - \varphi_T(x)]} = \sum_{(x,x',y)\in S} \exp\Big(-y \sum_{t=1}^{T} \alpha_t [h_t(x') - h_t(x)]\Big).$$



- Direction: unit vector $\mathbf{e}_t$ with

$$\mathbf{e}_t = \operatorname*{argmin}_t \frac{dF(\boldsymbol{\alpha} + \eta \mathbf{e}_t)}{d\eta}\bigg|_{\eta=0}.$$

- Since $F(\boldsymbol{\alpha} + \eta \mathbf{e}_t) = \sum_{(x,x',y)\in S} e^{-y\sum_{s=1}^T \alpha_s[h_s(x')-h_s(x)]} e^{-y\eta[h_t(x')-h_t(x)]}$,

$$\begin{aligned}
\frac{dF(\alpha + \eta e_t)}{d\eta}\bigg|_{\eta=0} &= -\sum_{(x,x',y)\in S} y[h_t(x') - h_t(x)] \exp\Big[-y\sum_{s=1}^T \alpha_s[h_s(x') - h_s(x)]\Big] \\
&= -\sum_{(x,x',y)\in S} y[h_t(x') - h_t(x)] D_{T+1}(x, x')\Big[m\prod_{s=1}^T Z_s\Big] \\
&= \boxed{-[\epsilon_t^+ - \epsilon_t^-]}\Big[m\prod_{s=1}^T Z_s\Big].
\end{aligned}$$

Thus, direction corresponding to base classifier selected by the algorithm.

- **Step size**: obtained via

$$
\begin{aligned}
&\frac{dF(\boldsymbol{\alpha} + \eta \mathbf{e}_t)}{d\eta} = 0 \\
&\Leftrightarrow -\sum_{(x,x',y)\in S} y[h_t(x') - h_t(x)] \exp\Big[-y\sum_{s=1}^{T} \alpha_s [h_s(x') - h_s(x)]\Big] e^{-y[h_t(x') - h_t(x)]\eta} = 0 \\
&\Leftrightarrow -\sum_{(x,x',y)\in S} y[h_t(x') - h_t(x)] D_{T+1}(x,x') \Big[m \prod_{s=1}^{T} Z_s\Big] e^{-y[h_t(x') - h_t(x)]\eta} = 0 \\
&\Leftrightarrow -\sum_{(x,x',y)\in S} y[h_t(x') - h_t(x)] D_{T+1}(x,x') e^{-y[h_t(x') - h_t(x)]\eta} = 0 \\
&\Leftrightarrow -[\epsilon_t^+ e^{-\eta} - \epsilon_t^- e^{\eta}] = 0 \\
&\Leftrightarrow \boxed{\eta = \frac{1}{2}\log\frac{\epsilon_t^+}{\epsilon_t^-}}.
\end{aligned}
$$

Thus, step size matches base classifier weight used in algorithm.

Bipartite Ranking

- Training data:
  - sample of negative points drawn according to D_-
  $$S_- = (x_1, \ldots, x_m) \in U.$$
  - sample of positive points drawn according to D_+
  $$S_+ = (x'_1, \ldots, x'_{m'}) \in U.$$

- Problem: find hypothesis $h \colon U \to \mathbb{R}$ in H with small generalization error

$$R_D(h) = \Pr_{x \sim D_-, x' \sim D_+} \big[h(x') < h(x) \big].$$

Notes

- Connection between AdaBoost and RankBoost (Cortes & MM, 04; Rudin et al., 05).
  - if constant base ranker used.
  - relationship between objective functions.
- More efficient algorithm in this special case (Freund et al., 2003).
- Bipartite ranking results typically reported in terms of AUC.

AdaBoost and CD RankBoost

- Objective functions: comparison.

$$
\begin{aligned}
F_{\text{Ada}}(\boldsymbol{\alpha}) &= \sum_{x_i \in S_- \cup S_+} \exp\left(-y_i f(x_i)\right) \\
&= \sum_{x_i \in S_-} \exp\left(+f(x_i)\right) + \sum_{x_i \in S_+} \exp\left(-f(x_i)\right) \\
&= F_-(\alpha) + F_+(\alpha).
\end{aligned}
$$

$$
\begin{aligned}
F_{\text{Rank}}(\boldsymbol{\alpha}) &= \sum_{(i,j) \in S_- \times S_+} \exp\left(-\left[f(x_j) - f(x_i)\right]\right) \\
&= \sum_{(i,j) \in S_- \times S_+} \exp\left(+f(x_i)\right) \exp\left(-f(x_i)\right) \\
&= F_-(\alpha) F_+(\alpha).
\end{aligned}
$$

AdaBoost and CD RankBoost

(Rudin et al., 2005)

- Property: AdaBoost (non-separable case).
  - constant base learner $h = 1 \longrightarrow$ equal contribution of positive and negative points (in the limit).
  - consequence: AdaBoost asymptotically achieves optimum of CD RankBoost objective.
- Observations: if $F_+(\boldsymbol{\alpha}) = F_-(\boldsymbol{\alpha})$,

$$\begin{aligned} d(F_{\text{Rank}}) &= F_+ d(F_-) + F_- d(F_+) \\ &= F_+ \big(d(F_-) + d(F_+)\big) \\ &= F_+ d(F_{\text{Ada}}). \end{aligned}$$

Bipartite RankBoost - Efficiency

- Decomposition of distribution: for $(x, x') \in (S_-, S_+)$,

$$D(x, x') = D_-(x) D_+(x').$$

- Thus,

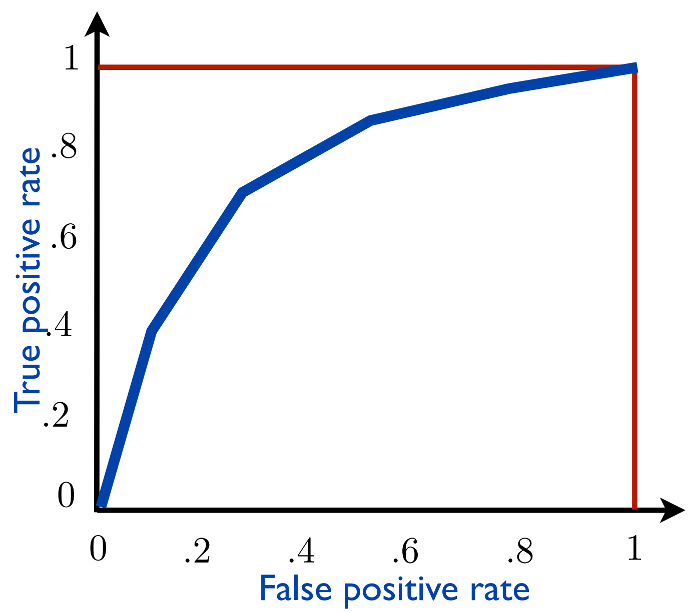
$$D_{t+1}(x, x') = \frac{D_t(x, x') e^{-\alpha_t [h_t(x') - h_t(x)]}}{Z_t}$$
$$= \frac{D_{t,-}(x) e^{\alpha_t h_t(x)}}{Z_{t,-}} \frac{D_{t,+}(x') e^{-\alpha_t h_t(x')}}{Z_{t,+}},$$

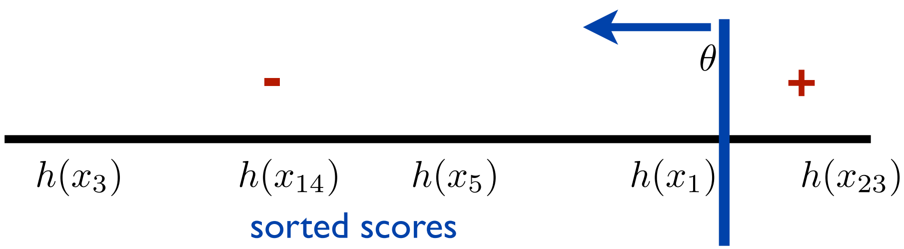
with $Z_{t,-} = \sum_{x \in S_-} D_{t,-}(x) e^{\alpha_t h_t(x)} \quad Z_{t,+} = \sum_{x' \in S_+} D_{t,+}(x') e^{-\alpha_t h_t(x')}.$

ROC Curve

(Egan, 1975)

- Definition: the receiver operating characteristic (ROC) curve is a plot of the true positive rate (TP) vs. false positive rate (FP).
  - TP: % positive points correctly labeled positive.
  - FP: % negative points incorrectly labeled positive.

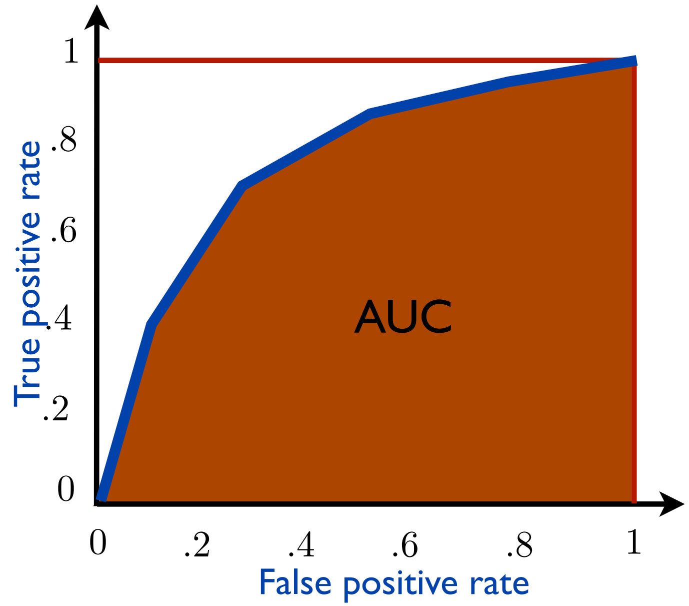




Area under the ROC Curve (AUC)

(Hanley and McNeil, 1982)

- Definition: the AUC is the area under the ROC curve. Measure of ranking quality.



- Equivalently,

$$AUC(h) = \frac{1}{mm'} \sum_{i=1}^{m} \sum_{j=1}^{m'} 1_{h(x'_j) > h(x_i)} = \Pr_{\substack{x \sim \widehat{D}_- \\ x' \sim \widehat{D}_+}} [h(x') > h(x)]$$

$$= 1 - \widehat{R}(h).$$

This Talk

- Score-based ranking
- Preference-based ranking

Preference-Based Setting

- Definitions:
  - U: universe, full set of objects.
  - V: finite query subset to rank, $V \subseteq U$.
  - τ^*: target ranking for V (random variable).
- Two stages: can be viewed as a reduction.
  - learn preference function $h\colon U \times U \to [0, 1]$.
  - given V, use h to determine ranking σ of V.
- Running-time: measured in terms of |calls to h|.

Preference-Based Ranking Problem

- Training data: pairs (V, τ^*) sampled i.i.d. according to D:

$$(V_1, \tau_1^*), (V_2, \tau_2^*), \ldots, (V_m, \tau_m^*) \qquad V_i \subseteq U.$$

subsets ranked by different labelers.

learn classifier

preference function $h \colon U \times U \to [0, 1]$.

- Problem: for any query set $V \subseteq U$, use h to return ranking $\sigma_{h,V}$ close to target τ^* with small average error

$$R(h, \sigma) = \mathop{\mathrm{E}}_{(V,\tau^*) \sim D} [L(\sigma_{h,V}, \tau^*)].$$

Preference Function

- $h(u, v)$ close to 1 when u preferred to v, close to 0 otherwise. For the analysis, $h(u, v) \in \{0, 1\}$.
- Assumed pairwise consistent:
$$h(u, v) + h(v, u) = 1.$$
- May be non-transitive, e.g., we may have
$$h(u, v) = h(v, w) = h(w, u) = 1.$$
- Output of classifier or 'black-box'.

Loss Functions

(for fixed (V, τ^*))

- Preference loss:

$$L(h, \tau^*) = \frac{2}{n(n-1)} \sum_{u \neq v} h(u, v) \tau^*(v, u).$$

- Ranking loss:

$$L(\sigma, \tau^*) = \frac{2}{n(n-1)} \sum_{u \neq v} \sigma(u, v) \tau^*(v, u).$$

(Weak) Regret

- Preference regret:

$$\mathcal{R}'_{class}(h) = \operatorname*{E}_{V,\tau^*}[L(h_{|V}, \tau^*)] - \operatorname*{E}_{V} \min_{\tilde{h}} \operatorname*{E}_{\tau^*|V}[L(\tilde{h}, \tau^*)].$$

- Ranking regret:

$$\mathcal{R}'_{rank}(A) = \operatorname*{E}_{V,\tau^*,s}[L(A_s(V), \tau^*)] - \operatorname*{E}_{V} \min_{\tilde{\sigma} \in S(V)} \operatorname*{E}_{\tau^*|V}[L(\tilde{\sigma}, \tau^*)].$$

Deterministic Algorithm

(Balcan et al., 07)

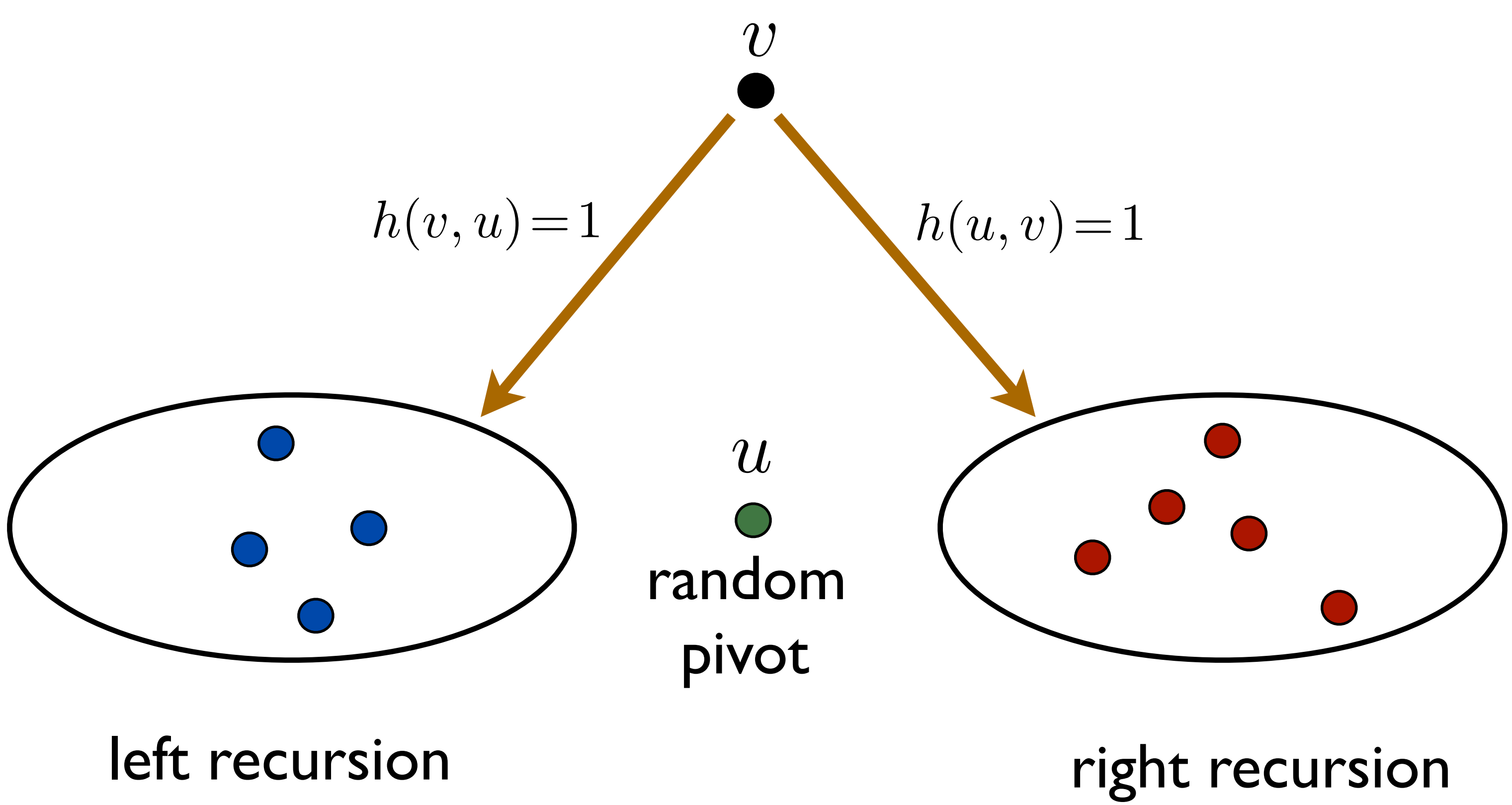
- Stage one: standard classification. Learn preference function $h\colon U \times U \to [0, 1]$.
- Stage two: sort-by-degree using comparison function h.
  - sort by number of points ranked below.
  - quadratic time complexity $O(n^2)$.

Randomized Algorithm

(Ailon & MM, 08)

- Stage one: standard classification. Learn preference function $h\colon U \times U \to [0,1]$.
- Stage two: randomized QuickSort (Hoare, 61) using h as comparison function.
    - comparison function non-transitive unlike textbook description.
    - but, time complexity shown to be $O(n \log n)$ in general.

Randomized QS



Deterministic Algo. - Bipartite Case

$(V = V_+ \cup V_-)$ (Balcan et al., 07)

- Bounds: for deterministic sort-by-degree algorithm
  - expected loss:

$$\mathop{\mathrm{E}}_{V,\tau^*}[L(A(V),\tau^*)] \leq 2 \mathop{\mathrm{E}}_{V,\tau^*}[L(h,\tau^*)].$$

  - regret:

$$\mathcal{R}'_{rank}(A(V)) \leq 2\,\mathcal{R}'_{class}(h).$$

- Time complexity: $\Omega(|V|^2)$.

Randomized Algo. - Bipartite Case

$(V = V_+ \cup V_-)$ (Ailon & MM, 08)

- Bounds: for randomized QuickSort (Hoare, 61).
  - expected loss (equality):

$$\underset{V,\tau^*,s}{\mathrm{E}}[L(Q_s^h(V),\tau^*)] = \underset{V,\tau^*}{\mathrm{E}}[L(h,\tau^*)].$$

  - regret:

$$\mathcal{R}'_{rank}(Q_s^h(\cdot)) \leq \mathcal{R}'_{class}(h) \quad .$$

- Time complexity:
  - full set: $O(n \log n)$.
  - top *k*: $O(n + k \log k)$.

Proof Ideas

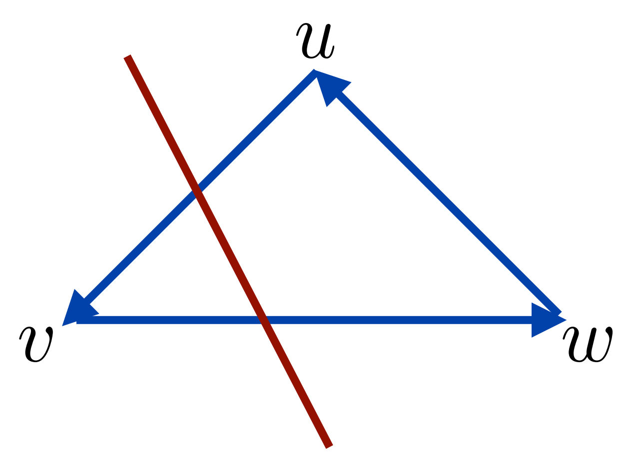
- QuickSort decomposition:

$$p_{uv} + \frac{1}{3} \sum_{w \notin \{u,v\}} p_{uvw} \Big(h(u,w)h(w,v) + h(v,w)h(w,u) \Big) = 1.$$

- Bipartite property:

$$\tau^*(u,v) + \tau^*(v,w) + \tau^*(w,u) =$$

$$\tau^*(v,u) + \tau^*(w,v) + \tau^*(u,w).$$



Lower Bound

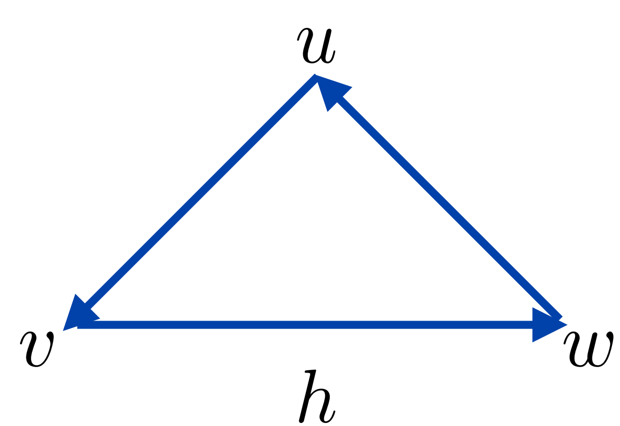
- **Theorem**: for any deterministic algorithm A, there is a bipartite distribution for which

$$\mathcal{R}_{rank}(A) \geq 2\,\mathcal{R}_{class}(h).$$

  - thus, factor of 2 $=$ best in deterministic case.
  - randomization necessary for better bound.

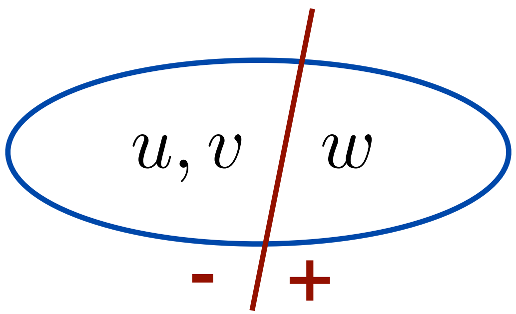
- **Proof**: take simple case $U = V = \{u, v, w\}$ and assume that h induces a cycle.
  - up to symmetry, A returns

    u, v, w or w, v, u.

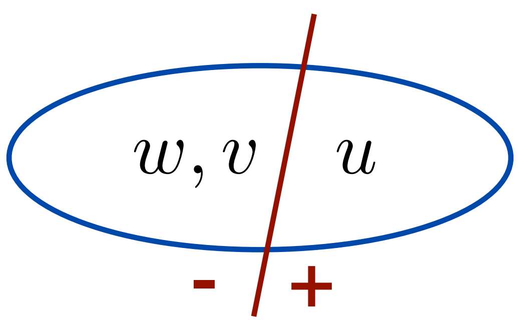


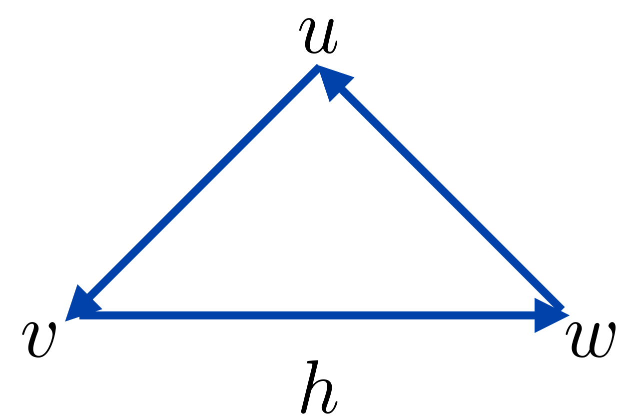
Lower Bound

- If A returns u, v, w, then choose τ^* as:



- If A returns w, v, u, then choose τ^* as:





$$L[h, \tau^*] = \frac{1}{3};$$

$$L[A, \tau^*] = \frac{2}{3}.$$

Guarantees - General Case

- Loss bound for QuickSort:

$$\mathop{\mathrm{E}}_{V,\tau^*,s}[L(Q_s^h(V),\tau^*)] \leq 2 \mathop{\mathrm{E}}_{V,\tau^*}[L(h,\tau^*)].$$

- Comparison with optimal ranking (see (CSS 99)):

$$\mathrm{E}_s[L(Q_s^h(V),\sigma_{optimal})] \leq 2\, L(h,\sigma_{optimal})$$

$$\mathrm{E}_s[L(h,Q_s^h(V))] \leq 3\, L(h,\sigma_{optimal}),$$

where $\sigma_{optimal} = \operatorname*{argmin}_{\sigma} L(h,\sigma).$

Weight Function

- Generalization:

$$\tau^*(u, v) = \sigma^*(u, v)\, \omega(\sigma^*(u), \sigma^*(v)).$$

- Properties: needed for all previous results to hold,
  - symmetry: $\omega(i, j) = \omega(j, i)$ for all i, j.
  - monotonicity: $\omega(i, j), \omega(j, k) \leq \omega(i, k)$ for $i < j < k$.
  - triangle inequality: $\omega(i, j) \leq \omega(i, k) + \omega(k, j)$ for all triplets i, j, k.

Weight Function - Examples

- Kemeny: $w(i,j) = 1, \ \forall i, j.$

- Top-k: $$w(i,j) = \begin{cases} 1 & \text{if } i \leq k \text{ or } j \leq k; \\ 0 & \text{otherwise.} \end{cases}$$

- Bipartite: $$w(i,j) = \begin{cases} 1 & \text{if } i \leq k \text{ and } j > k; \\ 0 & \text{otherwise.} \end{cases}$$

- *k*-partite: can be defined similarly.

(Strong) Regret Definitions

- Ranking regret:

$$\mathcal{R}_{rank}(A) = \operatorname*{E}_{V,\tau^*,s}[L(A_s(V), \tau^*)] - \min_{\tilde{\sigma}} \operatorname*{E}_{V,\tau^*}[L(\tilde{\sigma}_{|V}, \tau^*)].$$

- Preference regret:

$$\mathcal{R}_{class}(h) = \operatorname*{E}_{V,\tau^*}[L(h_{|V}, \tau^*)] - \min_{\tilde{h}} \operatorname*{E}_{V,\tau^*}[L(\tilde{h}_{|V}, \tau^*)].$$

- All previous regret results hold if for $V_1, V_2 \supseteq \{u, v\}$,

$$\operatorname*{E}_{\tau^*|V_1}[\tau^*(u, v)] = \operatorname*{E}_{\tau^*|V_2}[\tau^*(u, v)]$$

for all u, v (pairwise independence on irrelevant alternatives).

References

- Agarwal, S., Graepel, T., Herbrich, R., Har-Peled, S., & Roth, D. (2005). Generalization bounds for the area under the roc curve. *JMLR* 6, 393–425.

- Agarwal, S., and Niyogi, P. (2005). Stability and generalization of bipartite ranking algorithms. COLT (pp. 32–47).

- Nir Ailon and Mehryar Mohri. An efficient reduction of ranking to classification. In Proceedings of *COLT* 2008. Helsinki, Finland, July 2008. Omnipress.

- Balcan, M.-F., Bansal, N., Beygelzimer, A., Coppersmith, D., Langford, J., and Sorkin, G. B. (2007). Robust reductions from ranking to classification. In Proceedings of *COLT* (pp. 604–619). Springer.

- Stephen Boyd, Corinna Cortes, Mehryar Mohri, and Ana Radovanovic (2012). Accuracy at the top. In *NIPS 2012*.

- Cohen, W. W., Schapire, R. E., and Singer, Y. (1999). Learning to order things. J. Artif. Intell. Res. (JAIR), 10, 243–270.

- Cossock, D., and Zhang, T. (2006). Subset ranking using regression. COLT (pp. 605–619).

References

- Corinna Cortes and Mehryar Mohri. AUC Optimization vs. Error Rate Minimization. In *Advances in Neural Information Processing Systems (NIPS 2003)*, 2004. MIT Press.
- Cortes, C., Mohri, M., and Rastogi, A. (2007a). An Alternative Ranking Problem for Search Engines. Proceedings of *WEA* 2007 (pp. 1–21). Rome, Italy: Springer.
- Corinna Cortes and Mehryar Mohri. Confidence Intervals for the Area under the ROC Curve. In *Advances in Neural Information Processing Systems (NIPS 2004)*, 2005. MIT Press.
- Crammer, K., and Singer, Y. (2001). Pranking with ranking. Proceedings of NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada] (pp. 641–647). MIT Press.
- Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank Aggregation Methods for the Web. *WWW 10*, 2001. ACM Press.
- J. P. Egan. *Signal Detection Theory and ROC Analysis*. Academic Press, 1975.
- Yoav Freund, Raj Iyer, Robert E. Schapire and Yoram Singer. An efficient boosting algorithm for combining preferences. *JMLR* 4:933-969, 2003.

References

- J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 1982.
- Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Large margin rank boundaries for ordinal regression. Advances in Large Margin Classifiers, pages 115–132, 2000.
- Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press. 2012.
- Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the Web, *Stanford Digital Library Technologies Project*, 1998.
- Lehmann, E. L. (1975). Nonparametrics: Statistical methods based on ranks. San Francisco, California: Holden-Day.
- Cynthia Rudin, Corinna Cortes, Mehryar Mohri, and Robert E. Schapire. Margin-Based Ranking Meets Boosting in the Middle. In *Proceedings of The 18th Annual Conference on Computational Learning Theory (COLT 2005)*, pages 63-78, 2005.
- Thorsten Joachims. Optimizing search engines using clickthrough data. *Proceedings of the 8th ACM SIGKDD*, pages 133-142, 2002.