An Improved Branch-and-Bound Method for Maximum Monomial Agreement

Noam Goldberg
RUTCOR
Rutgers University

Joint work with:
Jonathan Eckstein, MSIS and RUTCOR
Rutgers University

January 20, 2009

Outline

Problem setting

We are given:

- a set of binary data $A \in \{0,1\}^{m \times n}$
- binary classes: $\Omega^+ \subset \{1,\ldots,m\}$ and $\Omega^- = \{1,\ldots,m\} \setminus \Omega^+$
- each observation i has weight $w(i)$.

$w(i)$	a_{i1}	a_{i2}	a_{i3}	a_{i4}	$+/-$
0.2374	1	0	0	1	$+$
1.7456	1	1	1	0	$+$
0.4357	0	0	0	1	$+$
1.4357	1	1	0	0	$-$
0.5127	1	1	0	1	$-$

Definitions

A monomial (a.k.a. a term)

$$p_{J,C}(x) = \prod_{j \in J} x_j \prod_{c \in C} (1 - x_c)$$

where J and C are disjoint subsets of $\{1, \ldots, n\}$

A monomial's *coverage*

Given $A \in \{0,1\}^{m \times n}$ where A_i is the i^{th} row of A

$$\text{Cover}_A(J, C) = \{i \in \{1, ..., m\} \mid p_{J,C}(A_i) = 1\}$$

Statement of the problem

Maximum Monomial Agreement (MMA)

Given $A \in \{0,1\}^{m \times n}$, $w : \{1,\ldots,m\} \to \mathbf{R}$, $\Omega^+ \subseteq \{1,\ldots,m\}$, $\Omega^- = \{1,\ldots,m\} \setminus \Omega^+$, find a monomial corresponding to $J, C \subseteq \{1,\ldots,n\}$, $J \cap C = \emptyset$ such that:

$$f(J,C) = \left| w(\mathrm{Cover}(J,C) \cap \Omega^+) - w(\mathrm{Cover}(J,C) \cap \Omega^-) \right|$$

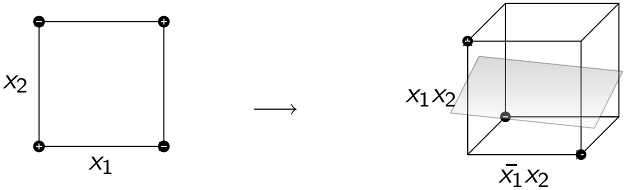
is maximized.

- A monomial classifier's accuracy is maximized by the above objective.
- It is precisely the objective of the "weak learner" subproblem within boosting (e.g., LP-Boost) when combining monomials.

Motivation

- Weighted voting classification methods:
  1. combine simple classifiers $h_p : \{0,1\}^n \to \{-1,0,1\}$ for $p \in \mathcal{P}$
  2. find a separating hyperplane $g(x) = \alpha_0 + \sum_p \alpha_p h_p(x)$ in $[0,1]^{|\mathcal{P}|}$, e.g., maximizing the L_1 margin
  3. classify any $x \in \{0,1\}^n$ based on $sgn(g(x))$
- A boosting algorithm iteratively finds g by linearly combining the classifiers h_p computed by a "base learner" / subroutine.

  Note: the weight of observation i, $w(i)$, is a dual variable value of an LP.



Our solution approach

- The MMA problem is $\mathcal{NP}$-hard (Kearns, Schapire & Sellie 1994)
- Solving the problem exactly using a B&B algorithm can improve on classification performance when used with robust boosting algorithm (Goldberg & Shan 2007)
- We also solve the problem using a branch-and-bound algorithm.
  - We improve the simple upper bound used in the previous branch-and-bound algorithm (GS 2007).
  - We consider more sophisticated dynamic branching schemes.

Branch-and-bound overview

A branch-and-bound algorithm is an enumeration (search tree) scheme whose practical efficiency depends on the ability to prune the search space as quickly as possible. More concretely the ability to prune depends on:

- effective heuristics for branching – recursively partitioning the search space into subproblems
- the quality of bound on the objective function of descendent subproblems
- the queueing discipline of “open subproblems”

Our solution technique

- We define a subproblem as a partition of the variable indices $\{1, \ldots, n\}$ into (J, C, E, F), where:
  - J - the set of variables fixed to be **in** the monomial
  - C - the set of variables fixed to be **complemented in** the monomial
  - E - the set of variables fixed to be **excluded** from the monomial
  - F - the set of **free** variables
- At the root we start with all variables free - $(\emptyset, \emptyset, \emptyset, \{1, \ldots, n\})$
- We branch by removing one or more variables from F, adding each to one of J, C or E.

$2 \times |F|$ branching (based on G & S)

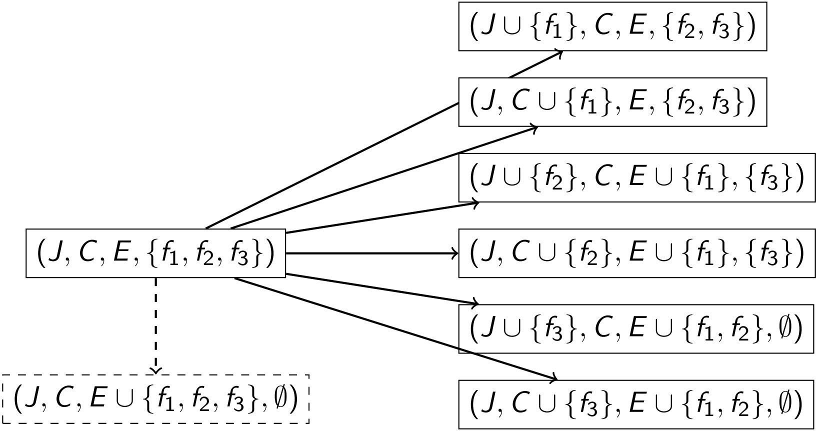


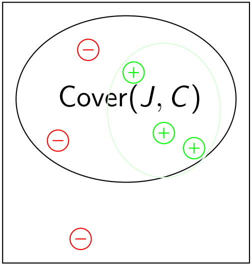
Figure: Branching of subproblem (J, C, E, F) into $2 \times |F|$ children. In the previous algorithm the ordering f_1, f_2, f_3 is lexical (i.e., it is static).

Simple upper bound (GS 2007)

$$u_{\text{gs}}(J, C) = \max\{w(\text{Cover}(J, C) \cap \Omega^+), w(\text{Cover}(J, C) \cap \Omega^-)\}$$

For example:

- $f(J, C) = 1$
- but $u_{gs}(J, C) = 3$.
- may be optimistic...

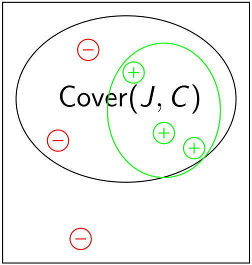


Simple upper bound (GS 2007)

$$u_{\mathrm{gs}}(J,C) = \max\{w(\mathrm{Cover}(J,C) \cap \Omega^{+}), w(\mathrm{Cover}(J,C) \cap \Omega^{-})\}$$

For example:

- $f(J,C) = 1$
- but $u_{gs}(J,C) = 3$.
- may be optimistic...



Inseparability

Definition

Two binary vectors $x, y \in \{0,1\}^n$ are *inseparable* with respect to a set $E \subseteq \{1, ..., n\}$, if, for all $j \in \{1, ..., n\} \setminus E$, one has $x_j = y_j$.

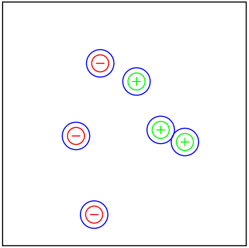
- Example: If $\{2, 4\} \subseteq E$, then A_1 and A_2 are inseparable

A_1	1	0	0	1
A_2	1	1	0	0

Inseparability equivalence classes

- E induces a partition of the observations into equivalence classes V_ℓ^E, for $\ell = 1, \ldots, q(E)$
- For given E the objective of any monomial restricted to V_ℓ^E is (tightly) bounded by $\left| w_\ell^+(J, C, E) - w_\ell^-(J, C, E) \right|$ where $w_\ell^+(J, C, E)$ is the sum of weights of positive observations in $\text{Cover}(J, C) \cap V_\ell^E$

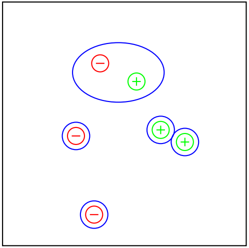
$$E = \emptyset, \ F = \{f_1, f_2, f_3, f_4\}$$



Inseparability equivalence classes

- E induces a partition of the observations into equivalence classes V_ℓ^E, for $\ell = 1, \ldots, q(E)$
- For given E the objective of any monomial restricted to V_ℓ^E is (tightly) bounded by $\left|w_\ell^+(J,C,E) - w_\ell^-(J,C,E)\right|$
  where $w_\ell^+(J,C,E)$ is the sum of weights of positive observations in $\text{Cover}(J,C) \cap V_\ell^E$

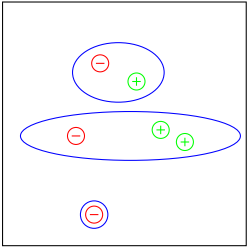
$E = \{f_1\}$, $F = \{f_2, f_3, f_4\}$



Inseparability equivalence classes

- E induces a partition of the observations into equivalence classes V_ℓ^E, for $\ell = 1, \ldots, q(E)$
- For given E the objective of any monomial restricted to V_ℓ^E is (tightly) bounded by $\left|w_\ell^+(J, C, E) - w_\ell^-(J, C, E)\right|$
  where $w_\ell^+(J, C, E)$ is the sum of weights of positive observations in $\mathrm{Cover}(J, C) \cap V_\ell^E$

$E = \{f_1, f_2\}$, $F = \{f_3, f_4\}$



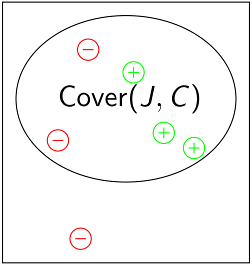
An improved upper bound

$$u(J,C,E) = \max \left\{ \begin{array}{l} \sum_{\ell=1}^{q(E)} \left(w_\ell^+(J,C,E) - w_\ell^-(J,C,E) \right)_+, \\ \sum_{\ell=1}^{q(E)} \left(w_\ell^-(J,C,E) - w_\ell^+(J,C,E) \right)_+ \end{array} \right\}$$

$$= u_{\text{gs}}(J,C) - \sum_{\ell=1}^{q(E)} \min\{w_\ell^+(J,C,E), w_\ell^-(J,C,E)\}$$

Back to our example:

- $f(J,C) = 1$
- $u_{gs}(J,C) = 3$
- $u(J,C,E) = (1-1) + (2-1) = 1$ which is tight



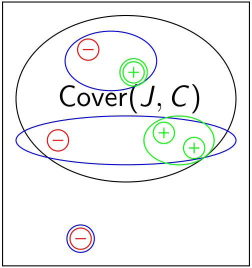
An improved upper bound

$$u(J,C,E) = \max \left\{ \begin{array}{l} \sum_{\ell=1}^{q(E)} \left(w_\ell^+(J,C,E) - w_\ell^-(J,C,E) \right)_+, \\ \sum_{\ell=1}^{q(E)} \left(w_\ell^-(J,C,E) - w_\ell^+(J,C,E) \right)_+ \end{array} \right\}$$

$$= u_{\mathrm{gs}}(J,C) - \sum_{\ell=1}^{q(E)} \min\{w_\ell^+(J,C,E), w_\ell^-(J,C,E)\}$$

Back to our example:

- $f(J,C) = 1$
- $u_{gs}(J,C) = 3$
- $u(J,C,E) = (1-1) + (2-1) = 1$ which is tight



Branching on a k-set of variables

- Branching on all of F entails a large branching factor
- Given k, we find $\{f_1, \ldots, f_k\} \subseteq F$ that maximizes the *inseparability* $\phi(J, C, \cdot)$. Recall:

$$u(J, C, E) = u_{\mathrm{gs}}(J, C) - \phi(J, C, E)$$

where

$$\phi(J, C, E) = \sum_{\ell=1}^{q(E)} \min\{w_\ell^+(J, C, E), w_\ell^-(J, C, E)\}$$

- This problem itself is *NP*-hard
- $\phi(J, C, \cdot)$ is supermodular
- Use a reverse greedy heuristic to find $\{f_1, \ldots, f_k\}$ and then partition into subproblems corresponding to excluding $\emptyset, \{f_1\}, \{f_1, f_2\}, \ldots, \{f_1, \ldots, f_k\}$.

Three way branching ($k = 1$)

$$\begin{tikzcd} & \boxed{(J \cup \{f_1\}, C, E, \{f_2, f_3\})} \\ \boxed{(J, C, E, \{f_1, f_2, f_3\})} \arrow[ur] \arrow[r] \arrow[dr] & \boxed{(J, C \cup \{f_1\}, E, \{f_2, f_3\})} \\ & \boxed{(J, C, E \cup \{f_1\}, \{f_2, f_3\})} \end{tikzcd}$$

Figure: Branching of subproblem (J, C, E, F) into three children.

When branching on a single variable:

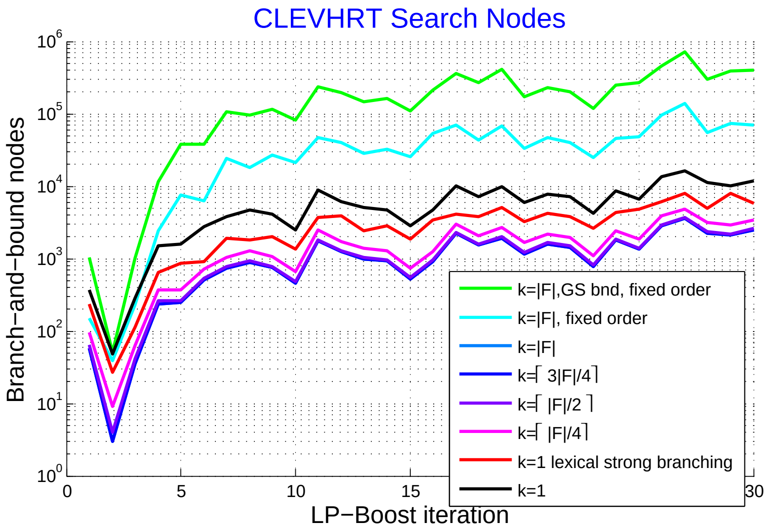
- we select the variable j that minimizes the maximum bound of the associated children (breaking ties lexicographically).
- it is less computationally intensive per search node than branching on k-sets

Computational results - cont'd

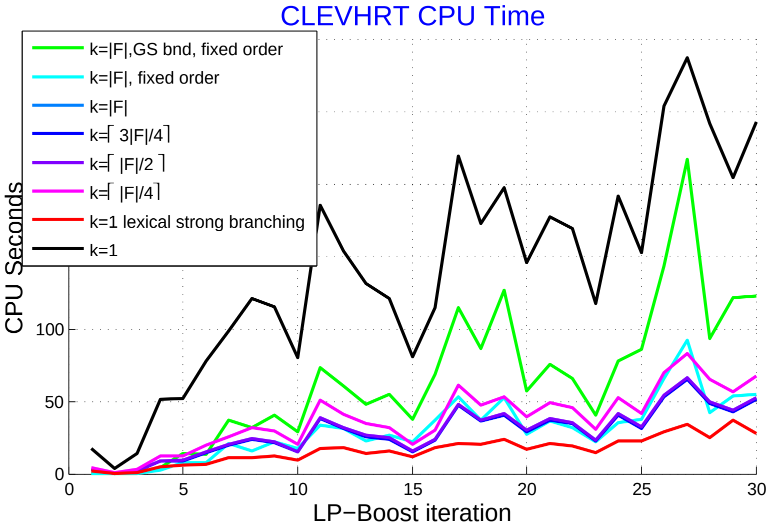
Dataset , # Features	LP-Boost Iters	u_{gs} **bound** $k = \|F\|$ CPU Sec	BB Nodes	$k = \|F\|$ CPU Sec	BB Nodes	$k = \lceil \|F\| /2 \rceil$ CPU Sec	BB Nodes	lex strong branching $k = 1$ CPU Sec	BB Nodes
SPECTHRT	1-15	0.6	7791.5	0.2	21.5	0.2	22.4	0.1	50.5
$n = 22$	16-30	1.8	22751.1	0.4	74.6	0.4	78.5	0.3	162.9
CLEVHRT	1-15	29.7	89551.2	16.9	626.2	17.3	654.7	9.5	1638.3
$n = 35$	16-30	99.8	317537.9	40.7	1872.3	41.2	1941.6	23.6	4831.6
HEPATITIS	1-15	17.3	83365.6	7.2	442.5	7.3	464.7	3.2	917.1
$n = 37$	16-30	Q LIMIT		13.5	970.9	13.8	1021.5	7.3	2650.3
PIMA	1-15	Q LIMIT		89.2	1290.5	90.0	1323.2	66.0	4606.3
$n = 33$	16-30	Q LIMIT		269.9	5499.3	269.6	5582.7	224.7	20907.1
CMC	1-15	Q LIMIT		1161.0	969.3	1158.6	972.7	496.9	3647.3
$n = 58$	16-30	Q LIMIT		LIMIT		LIMIT		1738.6	19437.3
HUNGHRT	1-15	Q LIMIT		LIMIT		LIMIT		264.0	14169.3
$n = 72$	16-30	Q LIMIT		LIMIT		LIMIT		763.6	47657.2

Table: Runtime and node averages over the specified LP-Boost (Demiriz, Bennett and Shawe-Taylor 2002) iterations, applying our algorithm to (binarized) UCI datasets. "Q LIMIT" indicates an iteration encountered the 500,000-node queue limit, and "LIMIT" indicates an iteration encountered the 1-hr time limit.

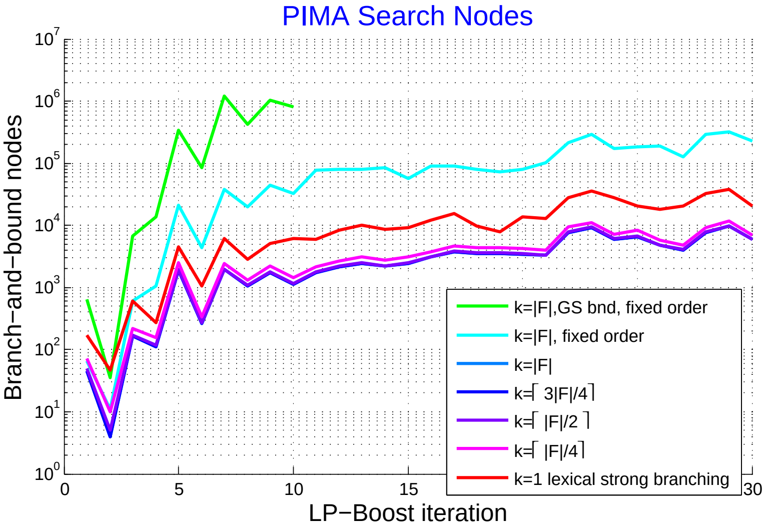
Computational results



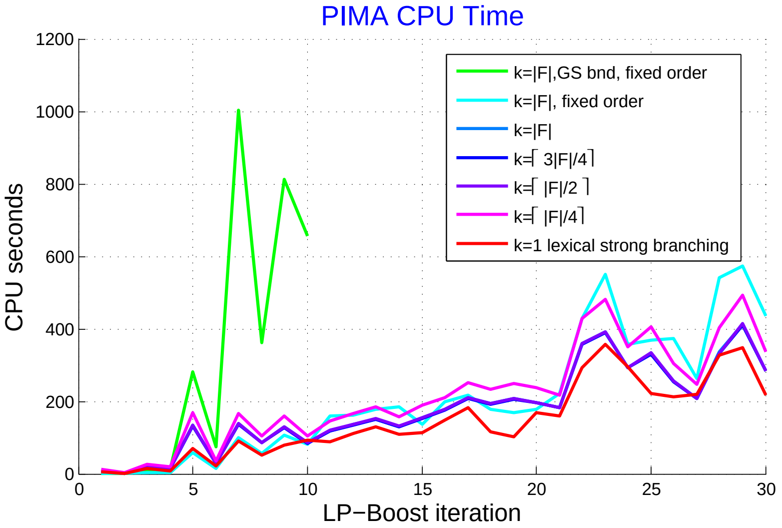
Computational results - cont'd



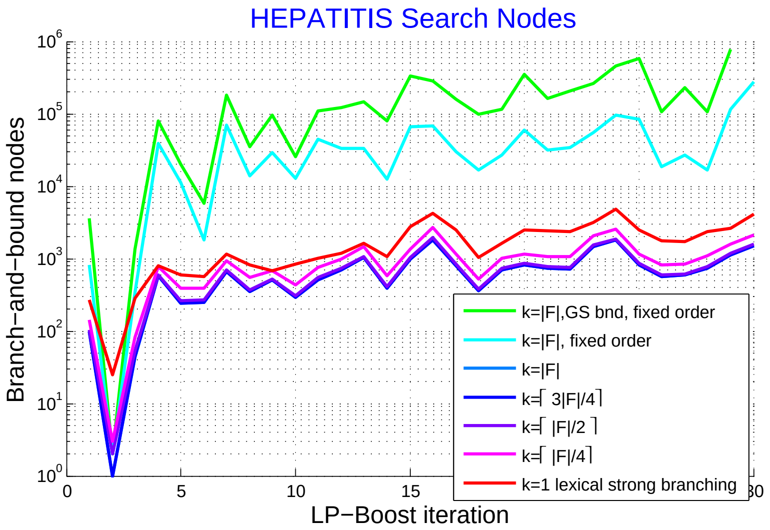
Computational results - cont'd



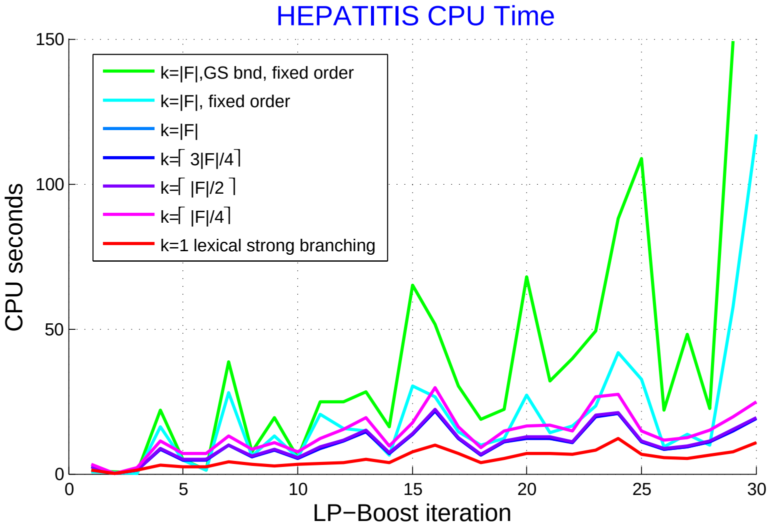
Computational results - cont'd



Computational results - cont'd



Computational results - cont'd



Conclusions and future work

- Three way branching is faster than both the previous (G&S) algorithm and k-set branching
- k-set branching seems to have the fewest search nodes, particularly for $k \geq \lceil |F| / 2 \rceil$

- Extend the problem and algorithm in order to incorporate within a new boosting formulation with a discrete information complexity penalty ("L_0 regularization").

Thank you!

To contact me:

Noam Goldberg
Email: ngoldberg@rutcor.rutgers.edu