Next Generation “Treatment Learning”

(finding the diamonds in the dust)

Tim Menzies

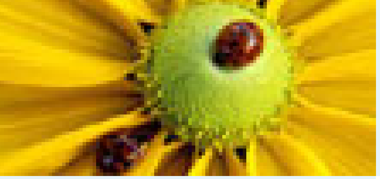
UpLift Systems

timm@bart-massey.com

The strangest thing...

“In any field, find the strangest thing, and explore it” – John Wheeler

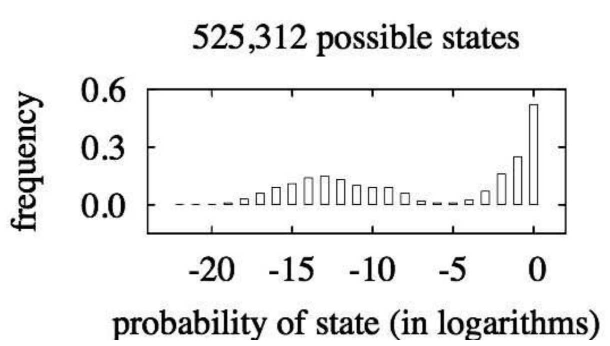
- Q: How have dummies (like me) managed to gain (some) control over a (seemingly) complex world?
- A: The world is simpler than we think.
  - Models contain clumps
  - A few collar variables decide which clumps to use.
- TAR2,TAR3,TAR4:
  - Data miners that assume clumps/collars
  - Reports effects never seen before
  - Finds solutions faster than other methods
  - Returns tiniest theories
  - Scales to infinite data streams ($\Longleftarrow$ new result)



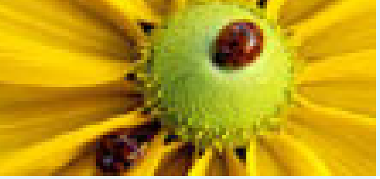
How Complex are our Models?

- COLLARS-
  A small number few variables controls the rest:
  - DeKleer [1986]: "Minimal environments" in the ATMS;
  - Menzies and Singh [2003]: "Tiny minimal environments";
  - Crawford and Baker [1994]: "Master variables" in scheduling;
  - Williams et al. [2003]: 'Backdoors" in satisfiability.

- CLUMPS-
  - Druzdzel [1994]. Commonly, a few states; very rarely, most states;
  - Pelanek [2004]. "Straight jackets" in formal models: state spaces usually sparse, small diameter, many diamonds.



25,000 states in IEEE1394

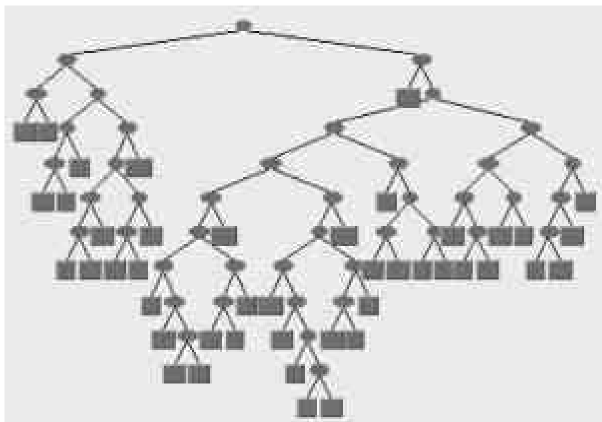


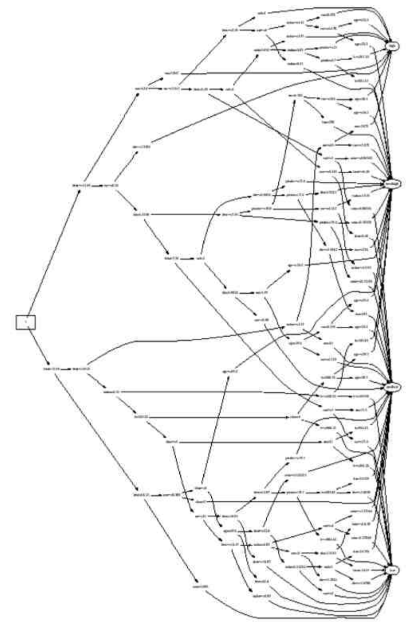
Exploiting Simplicity

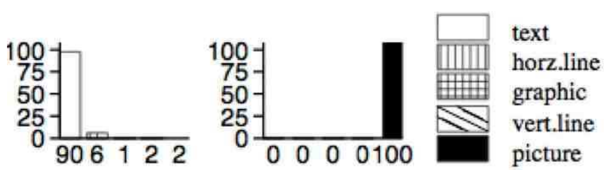
- If clumps
  - most of the action in a small number of states
  - effective search space = small
- If collars:
  - A few variables that switch you between states
- Treatment learning
  - If a few variables control the rest, then..
    - All paths $inputs \rightarrow outputs$ use the collars (by definition).
  - So don't search for the collars:
    - They'll find you.
    - Just sample, and count frequencies F.
  - Divide output $good$ and bad
    - Focus on ranges R_i with large $\frac{F(R_i|good)}{F(R_i|bad)}$
- Great way to learn tiny theories.

Learns Smaller Theories

find graphics on a page from 11 features	find good housing in Boston

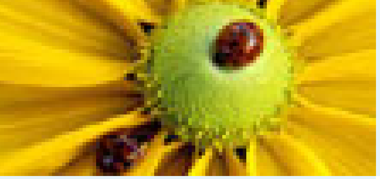






$34 \leq height < 86 \wedge$
$3.9 \leq mean\_tr < 9.5$

$6.7 \leq RM < 9.8 \wedge$
$12.6 \leq PTRATION < 15.9$



Why Learn Small Theories?

Reduce Uncertainty:

Linear regression: $\sigma^2 \propto |variables|$ (Miller [2002]);

“Pluralitas non est ponenda sine neccesitate”:

MDL (Wallace and Boulton [1968]); FSS (Hall and Holmes [2003])

Explanation:

Smaller theories are easier to explain (or audit).

Performance:

The simpler the target concept, the faster the learning.

Construction cost:

Need fewer sensors and actuators.

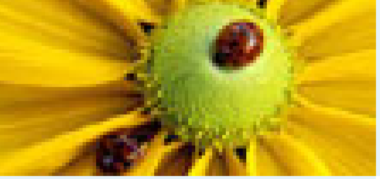
Operations cost:

Less to do: important for manual procedures;

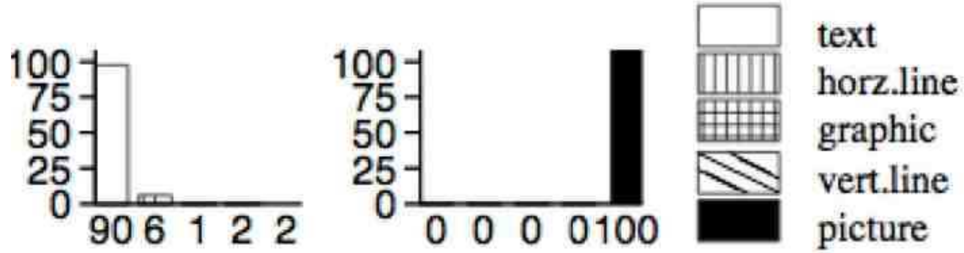
Less to watch: important for data-intensive tasks like security monitoring.

Pruning is good modeling:

Real world data often has noisy, irrelevant, redundant variables.



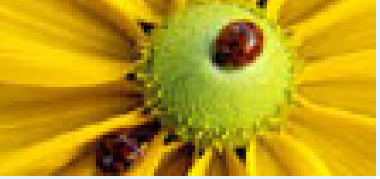
So What is Treatment Learning?



$$34 \leq height < 86 \ \wedge 3.9 \leq mean\_tr < 9.5$$

- E: training data with examples of $R_i \rightarrow C$
  - R_i: attribute ranges
  - C: classes with utilities $\{U_1 < U_2 < .. < U_C\}$
  - $F_1\%, F_2\%, ..., F_C\%$: frequencies of C in E
- T treatment of size X: $\{R_1 \wedge R_2 ... \wedge R_X\}$;
  - $T \cap E \rightarrow e \subseteq E$ with frequencies $f_1\%, f_2\%, ... f_C\%$
  - seek smallest T with largest $lift = \left(\sum_C U_C f_C\right) / \left(\sum_C U_C F_C\right)$
- This talk:
  - Implementation, examples, a new scale-up method

In practice...



The TAR3 Treatment Learner

- Assume clumps and collars
  - Just thrash around some.
- Build treatments $\{R_1 \wedge R_2 \ldots \wedge R_X\}$ of size X
  - FIRST try $X = 1$
  - THEN use the $X = 1$ results to guide the $X > 1$ search.
- Hu [2002] :: grow *treatments* via a stochastic search.
  - Discretization: equal frequency binning
- Empirically:
  - Run times linear on treatment SIZE, number of examples
  - Works as well as TAR2's complete search

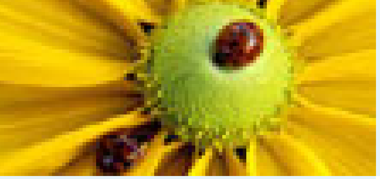
```
function ONE(x = random(SIZE) )
   x timesDo
      treatment = treatment + ANYTHING()
   return treatment

function ANYTHING()
   return a random range from CDF(lift1)

function SOME()
   REPEATS timesDo
      treatments = treatments + ONE()
   sort treatments on lift
   return ENOUGH top items

function TAR3(lives = LIVES )
   for every range r do lift1[r]= lift(r)
   repeat
     before = size(temp)
     temp  = union(temp, SOME())
     if (before==size(temp))
     then lives--
     else lives = LIVES
   until lives == 0
   sort temp on lift;
   return ENOUGH top items
```

Useful defaults: $<$SIZE=10, REPEATS=100, ENOUGH=20, LIVES=5$>$

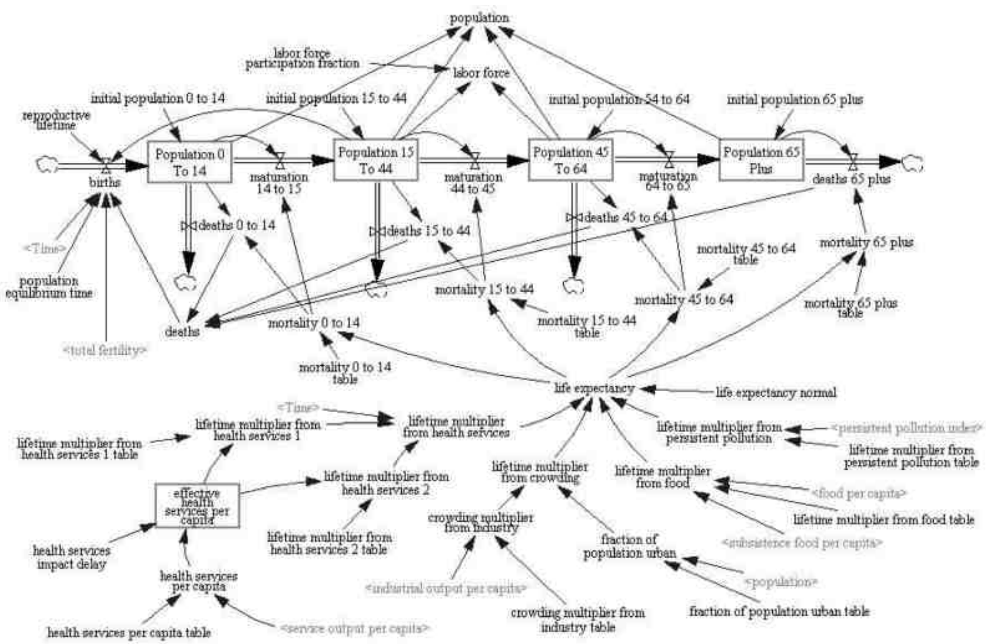


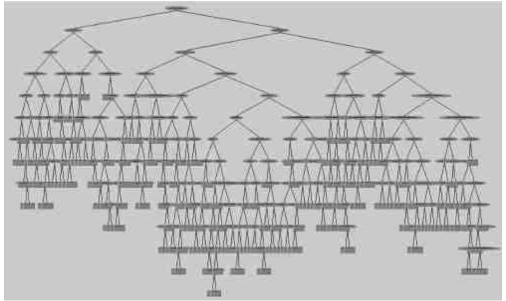
Saving the World

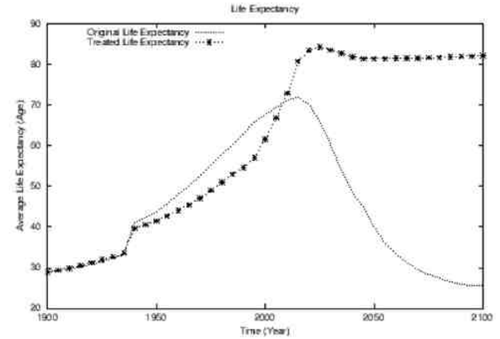
"Limits to Growth" :: Meadows et al. [1972]

A second look at "Limits to Growth": Geletko and Menzies [2003]

Vensim's World-3 (1991): 295 variables

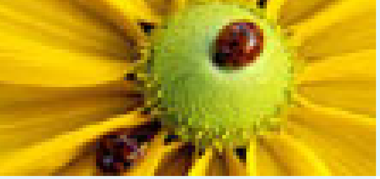






Happily ever after if

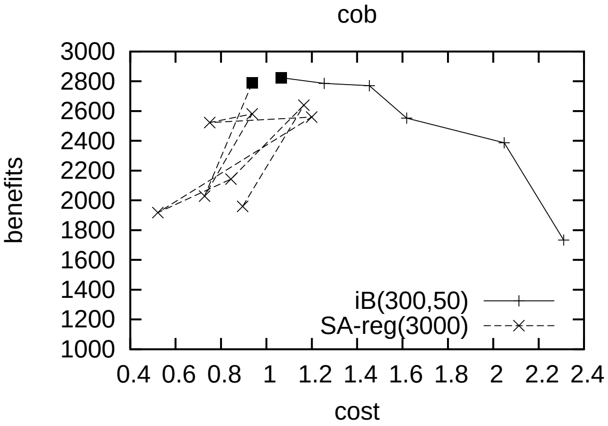
- family size ≤ 2, menstruation onset > 18, industrial capital output = [3..5).
- This happy ending is *not* mentioned in Meadows et al. [1972].



Compared with More Complete Search

- DDP requirements models from deep-space missions (from JPL).
- Iterative learning: $simulation_i \rightarrow learn \rightarrow constrain \rightarrow simulation_{i+1}$

$$SA = \frac{\frac{benefit}{maxBenefit} + (1 - \frac{cost}{maxCost})}{\left(2 * \text{number of selected mitigations}\right) + 1}$$



TAR3: 7*300 samples

SA: 9*3000 samples

Learns Very Tiny Theories

Introduction

In practice...
- Algorithm
- Saving the World
- Compare
- Learns Very Tiny Theories

Scaling Up

Related Work

And so...

Questions? Comments?

- Compare with feature subset selection: Hall and Holmes [2003]
- For each class $c \in C$
  - Give c the largest utility U_c.
  - Find treatments for c
- $Selected$ = all attributes in treatments for all $c \in C$.
- $Accuracy$= $selected$'s performance in some *target learner*.
- Menzies et al. [2005]

domain	% of attributes ignored	accuracy improvement
Anneal	81.6%	2.66%
credit-g	75.0%	2.17%
Soybean	54.3%	0.65%
vote	62.5%	0.21%
breast-c	77.8%	0.00%
Segment	78.9%	-0.51%
Ionosphere	94.1%	-0.90%
Diabetes	87.5%	-2.28%
lymph	83.3%	-2.75%
HorseColic	90.9%	-4.45%
average	78.6%	-1.13%

#attributes selected (target learner = C4.5)								
	original	ig	cfs	cbs	rlf	wrp	pc	select
Soybean	35	19	24	35	32	19	30	▶ 16
Anneal	38	17	21	15	20	18	36	▶ 7
vote	16	12	10	▶ 6	11	9	11	▶ 6
credit-g	20	8	7	8	9	8	▶ 4	5
Segment	19	16	12	9	13	9	16	▶ 4
lymph	18	6.8	5.3	4	4	6	9	▶ 3
breast-c	9	4	4	7	7	4	4	▶ 2
Horse colic	22	4	4	▶2	3	5	3	▶ 2
Ionosphere	34	12	7	9	9	7	10	▶ 2
Diabetes	8	33	3	4	4	4	6	▶1

Scaling Up

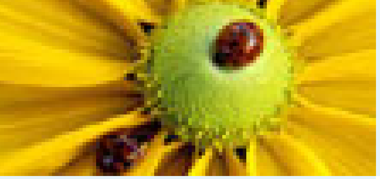
TAR3 is not a Data Miner

The data mining desiderata :: Bradley et al. [1998]:

- Requires one scan, or less of the data
- On-line, anytime algorithm
- Suspend-able, stoppable, resumable
- Efficiently and incrementally add new data to existing models
- Works within the available RAM

TAR3 is *not* a data miner

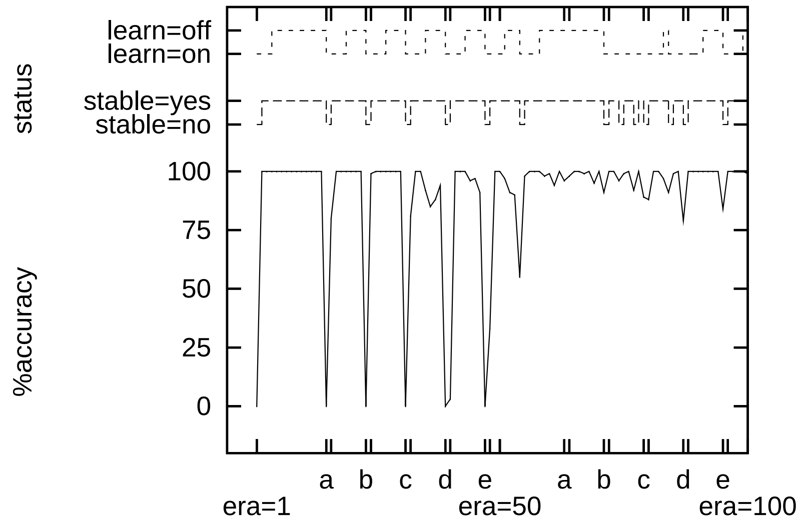
- Stores all examples in RAM
- Requires at three scans
  1. discretization
  2. collect statistics, build treatments
  3. rank generated theories

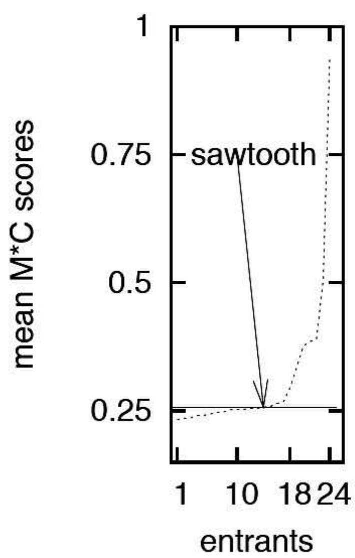


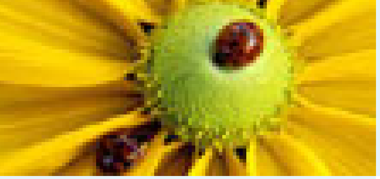
SAWTOOTH is a data miner

SAWTOOTH= incremental NaïveBayes classifier Menzies and Orrego [2005]

- Exploits the "saturation effect":
  - Learners performance improves and plateaus, after 100s of examples
  - Processes data in chunks (window = 250)
  - Disables learning while performance stable
- One-pass through the data
  - Incremental discretization of numeric data (SPADE)
  - Input each example, converted to frequency counts, then deletes
- Results
  - Small memory; scales.
  - Recognizes and reacts to concept drift
- Can we model treatment learning as a NaïveBayes classifier?







NaïveBayes classifiers

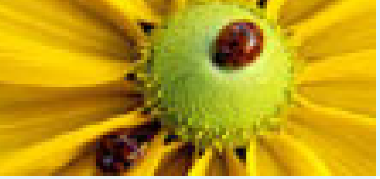
evidence E, hypothesis H

$$\overbrace{P(H|E)}^{future=} = \overbrace{\left(\prod_i P(E_i|H)\right)}^{now*} * \overbrace{\frac{P(H)}{P(E)}}^{past}$$

	E_1	E_2	E_3
$H = car$	job	suburb	wealthy?
ford	tailor	NW	y
ford	tailor	SE	n
ford	tinker	SE	n
bmw	tinker	NW	y
bmw	tinker	NW	y
bmw	tailor	NW	y

	$P(E_i\|H)$		
P(H)	job	suburb	wealthy?
ford:3=0.5	tinker:1=0.33	NW:1=0.33	y:1=0.33
	tailor:2=0.67	SE:2=0.67	n:2=0.67
bmw:3=0.5	tinker:2=0.67	NW:3=1.00	y:3=1.00
	tailor:1=0.33	SE:0=0.00	n:0=0.00

- E = job=tailor & suburb=NW
- likelihood = $L(bmw|E)$ = $\prod_i P(E|bmw) * P(bmw)$ =0.33*1.00*0.5 =0.16500
- $L(ford|E)$ = $\prod_i P(E|ford) * P(ford)$ =0.67*0.33*0.5 =0.11055
- $Prob(bmw|E) = \frac{L(bmw|E)}{L(bmw|E)+L(ford|E)}$ = 59.9%
- $Prob(ford|E) = \frac{L(ford|E)}{L(bmw|E)+L(ford|E)}$ = 40.1%
- So our tailor drives a bmw
- Naïve: assumes independence; counts single attribute ranges (not combinations)
  - But optimal under the one-zero assumption Domingos and Pazzani [1997].
  - Incremental simple, fast learning/classification speed, low storage space.



CUBE & TAR4

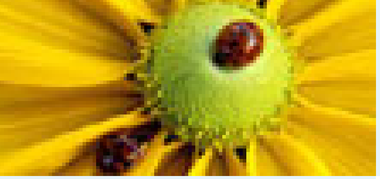
outlook	U_1: minimize temperature	humidity	windy	U_2: maximize play	up_i	$down_i$
overcast	64	65	TRUE	yes=1	1.00	0
rainy	68	80	FALSE	yes=1	0.87	0.13
...	...	...	...	...	...	...
sunny	80	90	TRUE	no=0	0.11	0.89
sunny	85	85	FALSE	no=0	0.00	1

- Examples are placed in a U-dimensional hypercube (one dimension for each utility):
  - apex = best = {1,1,1,1...};
  - base = worst = {0,0,0,0,...}
- $example_i$ has distance $0 \leq D_i \leq 1$ from apex (normalized by $U^{0.5}$)
- Each range $R_j \in example_i$ adds $down_i = D_i$ and $up_i = 1 - D_i$ to $F(R_j|base)$ and $F(R_j|apex)$.

$$
\begin{aligned}
P(apex) &= \textstyle\sum_i up_i / \left(\sum_i up_i + \sum_i down_i\right) \\
P(base) &= \textstyle\sum_i down_i / \left(\sum_i up_i + \sum_i down_i\right) \\
P(R_j|apex) &= F(R_j|apex) / \textstyle\sum_i up_i \\
P(R_j|base) &= F(R_j|base) / \textstyle\sum_i down_i \\
L(apex|R_k \wedge R_l \wedge \ldots) &= \textstyle\prod_x P(R_x|apex) * P(apex) \\
L(base|R_k \wedge R_l \wedge \ldots) &= \textstyle\prod_x P(R_x|base) * P(base)
\end{aligned}
$$

TAR4.0: Bayesian treatment learner = find the *smallest* treatment T that *maximizes*:

$$P(apex|T) = \frac{L(apex|T)}{L(apex|T) + L(base|T)}$$; didn't work: out-performed by TAR3



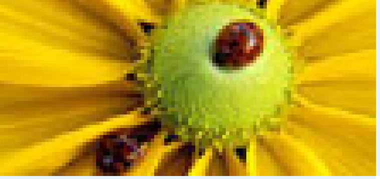
Why did TAR4.0 fail?

- Hypothesis: muddled-up by dependent attributes;
- "Naïve" Bayes: assume independence, keeps *singleton* counts.

	E_1	E_2	E_3
$H = car$	job	suburb	wealthy?
ford	tailor	NW	y
ford	tailor	SE	n
ford	tinker	SE	n
bmw	tinker	NW	y
bmw	tinker	NW	y
bmw	tailor	NW	y

E	P(bmw|E)	P(ford|E)
$job = tailor$ & $suburb = NW$	59.9%	40.1%
$job = tailor$ & $suburb = NW$ & $wealthy = y$	81%	19.0%

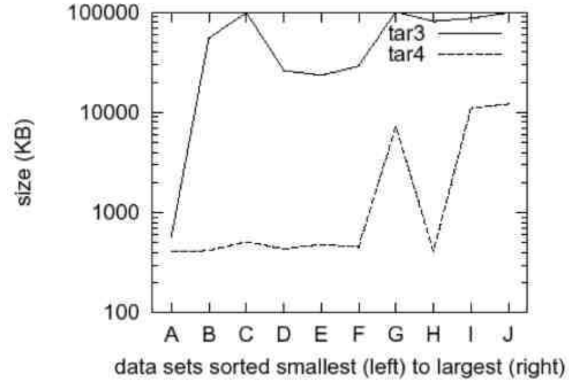
- Adding redundant information radically changes probabilities? Bad!
- Note: gets class probabilities WRONG, but RANKS classes correctly Domingos and Pazzani [1997]
- We asked TAR4.0 to do what you must never do:
  - compare numeric of probabilities of the same class in NaïveBayes.



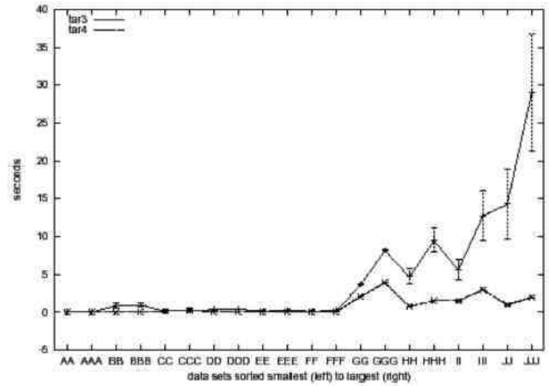
TAR4.1

- Prune treatments with low support in the data.
- What does "support" mean?
  - Maximal when includes all examples from a class
  - $0 \leq support \leq 1$
  - $support = likelihood = \prod_x P(R_x|H) * P(H)$
- $probability * support = \frac{L(apex|E)^2}{L(apex|E)+L(base|E)}$
- Worked!
  - Much faster, less memory than TAR3:
    - No need for a second scan
    - No need to hold examples in RAM
  - Bayesian guess-timate for $support$ of best class (almost) the same as TAR3
  - No connection treatment size to guess-timate error.
- But why did it work so well?

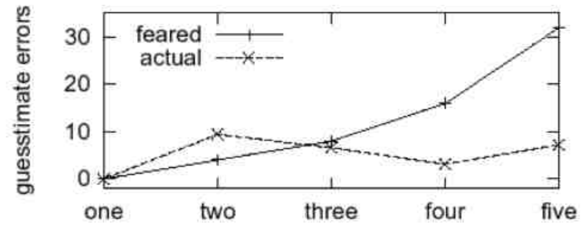
less memory

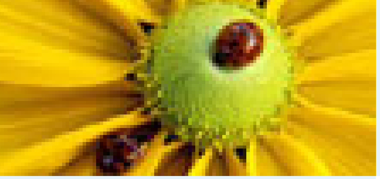


faster, less variance



lift errors small





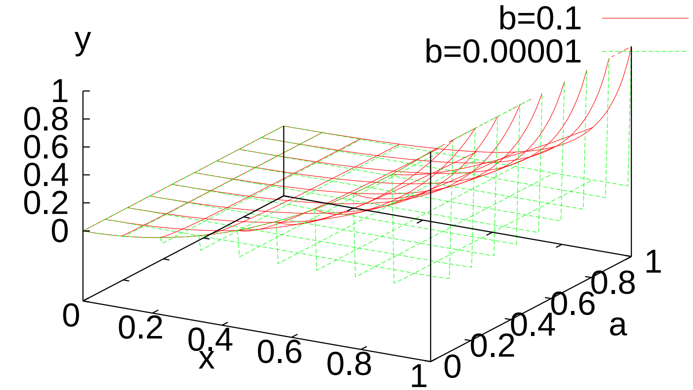
When Won't Dependencies Confuse TAR4?

- $T' = T + t$ where t is an attribute dependent on members of T;
- TAR4.1 *not* confused by t when it ignores treatments that use it.

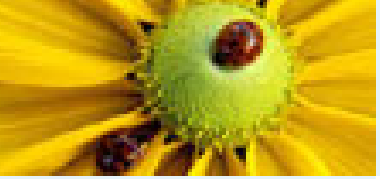
$$
\begin{aligned}
a = \quad & L(apex|T') = \quad \overbrace{P(t|apex)}^{x} * \textstyle\prod_i P(T_i|apex) * P(apex) \\
b = \quad & L(base|T') = \quad \underbrace{P(t|base)}_{y} * \textstyle\prod_i P(T_i|base) * P(base)
\end{aligned}
$$

- Then when is $support * probability$ increased by ignoring x and y?

$$
\left(\overbrace{\frac{(a/x)^2}{a/x + b/y}}^{\text{ignoring } x \text{ and } y} > \overbrace{\frac{a^2}{a+b}}^{\text{using } x \text{ and } y} \right) \Longrightarrow y > \frac{bx^2}{b + a - xa}
$$

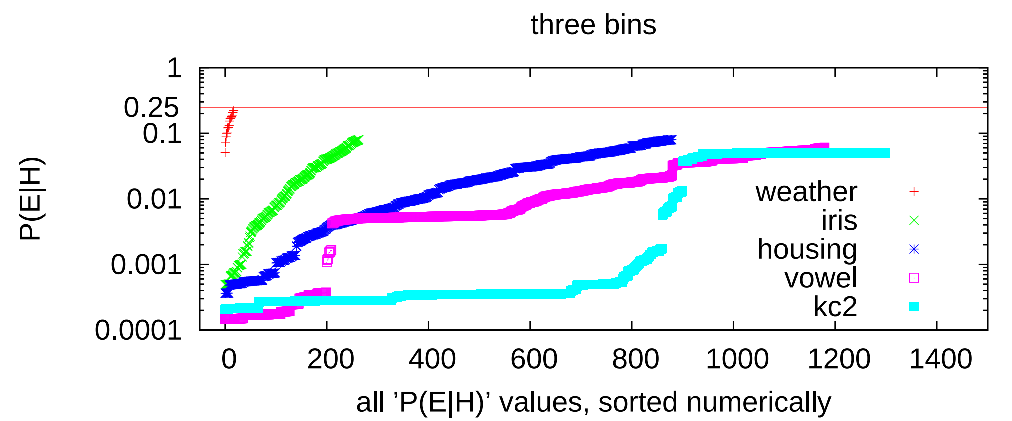


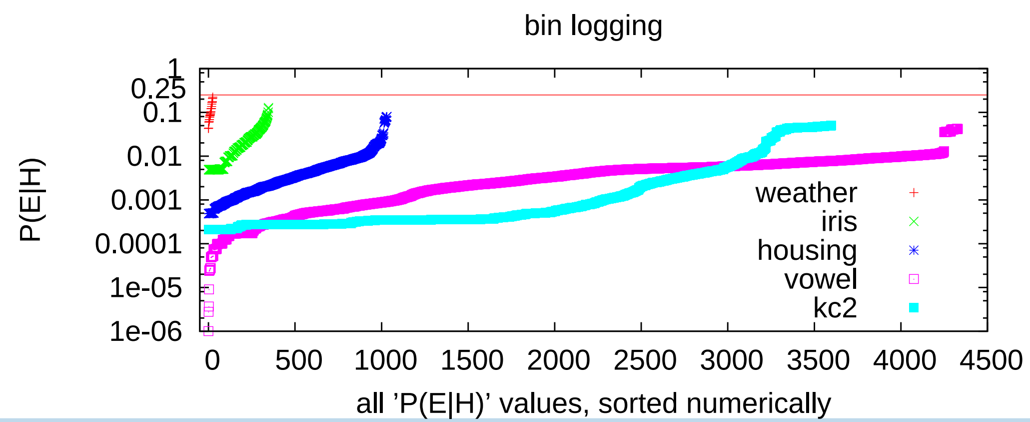
- And for TAR4.0:s pre-condition for no confusion: $\frac{(a/x)}{a/x+b/y} > \frac{a}{a+b}$

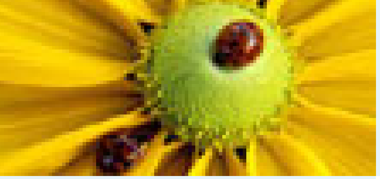


Typical Values and Constraints:: $\frac{(a/x)^2}{a/x+b/y}$

$0 < i \leq 20$	; treatment size
$b < a$	; *apex* is better than *base*
$10^{-10} < x \leq y \leq 0.25$	; see graphs
$0 < a \leq x^i \leq x \leq 0.25$	; *a* combines many *x*-like numbers
$0 < b \leq y^i \leq y \leq 0.25$	; *b* combines many *y*-like numbers

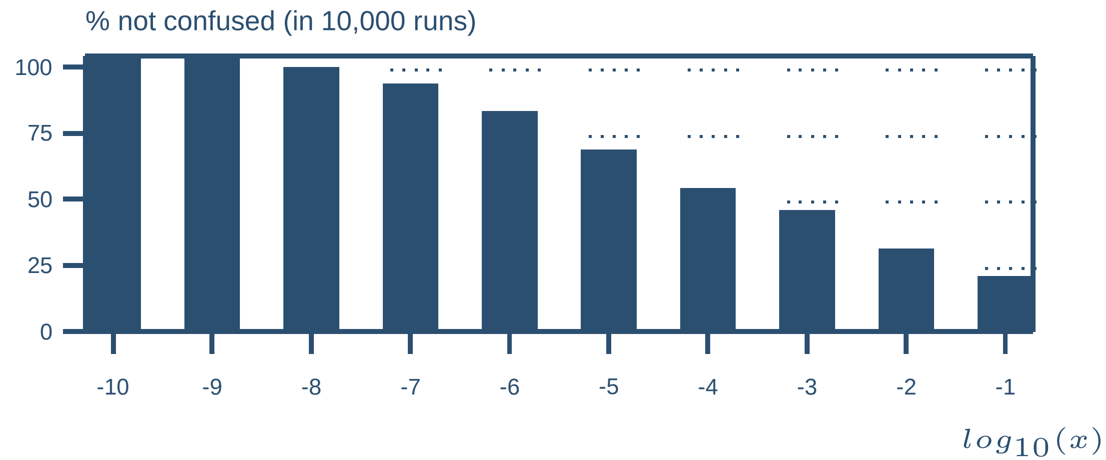






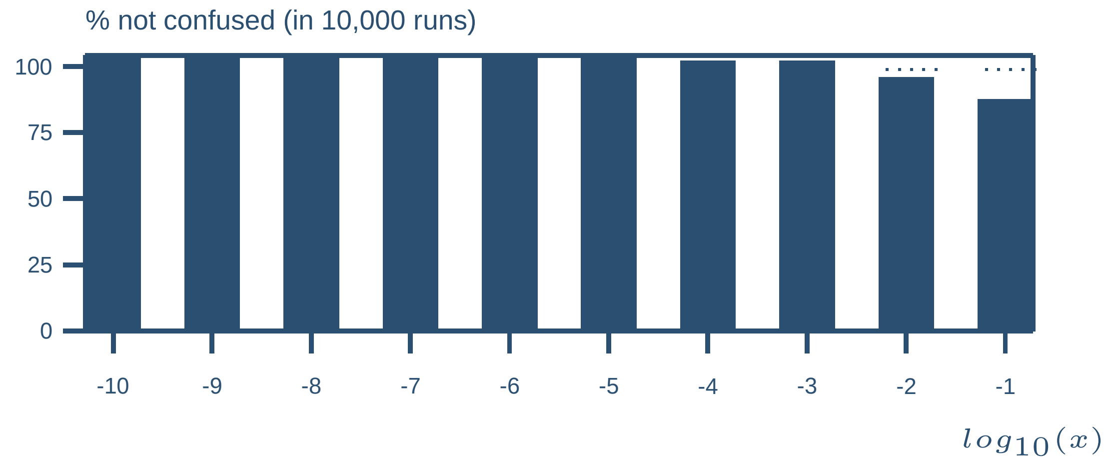
TAR4.1 Works

- Pick {a,b,x,y,i} at random within typical values; reject those violate our constraints;
- Check pre-conditions; report rounded log_{10} values;
- TAR4.0: not confused when $\left(\frac{(a/x)}{a/x+b/y} > \frac{a}{a+b}\right)$

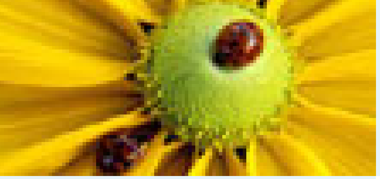


Often confused.

- TAR4.1: not confused when $\left(\frac{(a/x)^2}{a/x+b/y} > \frac{a^2}{a+b}\right)$

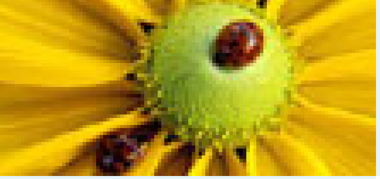


Rarely confused.



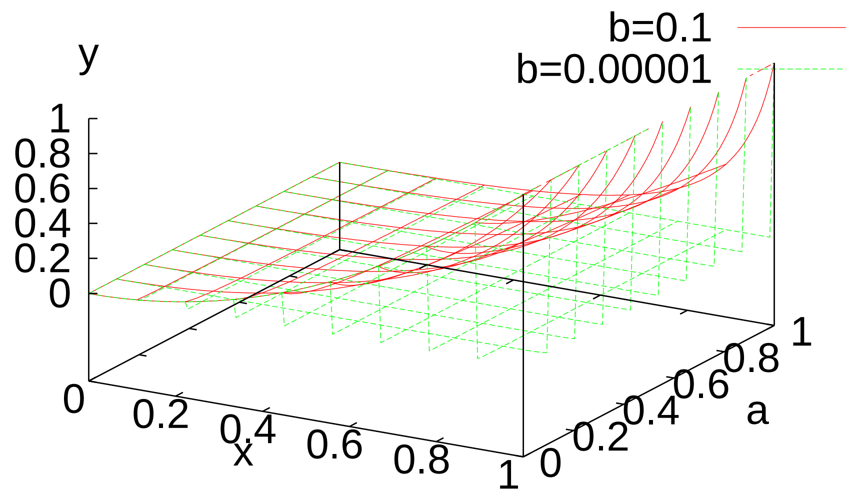
So What?

- Mathematically, TAR4.0 will always fails (except for $x \ll 1$);
- TAR4.1 succeeds since pre-condition is usually satisfied
  - In 96.52% of our simulations
- So, theoretically and empirically:
  - Bayesian treatment learning with CUBE can guess effect of treatments using frequency counts,
  - Does not need a second scan of the data (providing you use $support * probability$)
  - Now we have a data miner TAR4.1.
- By the way,
  - No need for Bayes nets in this domain
  - Why doesn't this mean that treatments will *never* grow beyond size=1?



But Why Big Treatments?

- When are larger treatments acceptable; i.e. $\left(\frac{(a/x)^2}{a/x+b/y} < \frac{a^2}{a+b}\right)$?
- When is $y < \frac{bx^2}{b+a-xa}$.

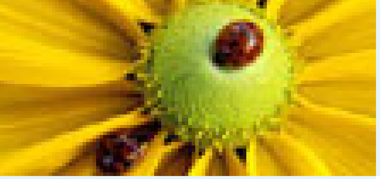


- When x is large and y is much smaller than x
- i.e. when some attribute ranges has a high frequency in the apex *and* a much lower frequency in the base.
- If collars then such ranges are not common; i.e. dependencies unlikely.

Related Work

References

- SAWTOOTH
  - Tim Menzies and Andres Orrego. Incremental discreatization and bayes classifiers handles concept drift and scaled very well. 2005. Submitted, IEEE TKDE, Available from `http://menzies.us/pdf/05sawtooth.pdf`
- Treatment learning
  - R. Clark. Faster treatment learning, 2005
  - D. Geletko and T. Menzies. Model-based software testing via treatment learning. In *IEEE NASE SEW 2003*, 2003. Available from `http://menzies.us/pdf/03radar.pdf`
  - Y. Hu. Treatment learning, 2002. Masters thesis, Unviersity of British Columbia, Department of Electrical and Computer Engineering. In preperation
  - T. Menzies, R. Gunnalan, K. Appukutty, Srinivasan A, and Y. Hu. Learning tiny theories. In *International Journal on Artificial Intelligence Tools (IJAIT), to appear*, 2005. Available from `http://menzies.us/pdf/03select.pdf`
  - T. Menzies and Y. Hu. Just enough learning (of association rules): The TAR2 treatment learner. In *Artificial Intelligence Review (to appear)*, 2006. Available from `http://menzies.us/pdf/02tar2.pdf`
  - T. Menzies and Y. Hu. Data mining for very busy people. In *IEEE Computer*, November 2003. Available from `http://menzies.us/pdf/03tar2.pdf`



References (2)

- Phase transition
  - H.H. Hoos and T. Stutzle. Evaluating las vegas algorithms - pitfalls and remedies. In *Proc. of UAI-98*, 1998. Available from `http://www.cs.ubc.ca/~hoos/Publ/uai98.ps`
  - David G. Mitchell, Bart Selman, and Hector J. Levesque. Hard and easy distributions for SAT problems. In Paul Rosenbloom and Peter Szolovits, editors, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 459–465, Menlo Park, California, 1992. AAAI Press. Available from http://www.citeseer.ist.psu.edu/mitchell92hard.html

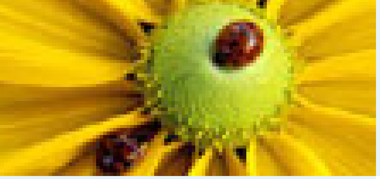
- Contrast set learners
  - S.B. Bay and M.J. Pazzani. Detecting change in categorical data: Mining contrast sets. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, 1999. Available from `http://www.ics.uci.edu/~pazzani/Publications/stucco.pdf`
  - C.H. Cai, A.W.C. Fu, C.H. Cheng, and W.W. Kwong. Mining association rules with weighted items. In *Proceedings of International Database Engineering and Applications Symposium (IDEAS 98)*, August 1998. Available from `http://www.cse.cuhk.edu.hk/~kdd/assoc_rule/paper.pdf`

References (3)

Introduction
In practice...
Scaling Up
Related Work
- References
- References (2)
- References (3)
- References (4)
- References (5)
And so...
Questions? Comments?

- Collars and clumps
  - J. Crawford and A. Baker. Experimental results on the application of satisfiability algorithms to scheduling problems. In *AAAI '94*, 1994
  - J. DeKleer. An Assumption-Based TMS. *Artificial Intelligence*, 28:163–196, 1986
  - M.J. Druzdzel. Some properties of joint probability distributions. In *Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 187–194, 1994
  - R. Pelanek. Typical structural properties of state spaces. In *Proceedings SPIN'04 Workshop*, 2004. Available from `http://www.fi.muni.cz/~xpelanek/publications/state_spaces.ps`
  - R. Williams, C.P. Gomes, and B. Selman. Backdoors to typical case complexity. In *Proceedings of IJCAI 2003*, 2003. `http://www.cs.cornell.edu/gomes/FILES/backdoors.pdf`

- Data mining
  - Paul S. Bradley, Usama M. Fayyad, and Cory Reina. Scaling clustering algorithms to large databases. In *Knowledge Discovery and Data Mining*, pages 9–15, 1998. Available from `http://citeseer.ist.psu.edu/bradley98scaling.html`
  - P. Domingos and G. Hulten. Mining high-speed data streams. In *Knowledge Discovery and Data Mining*, pages 71–80, 2000. URL `citeseer.ist.psu.edu/domingos00mining.html`



References (4)

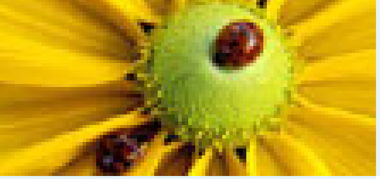
- Feature subset selection
  - M.A. Hall and G. Holmes. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions On Knowledge And Data Engineering*, 15(6): 1437– 1447, 2003
  - Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997. URL `citeseer.nj.nec.com/kohavi96wrappers.html`
  - A. Miller. *Subset Selection in Regression (second edition)*. Chapman & Hall, 2002. ISBN 1-58488-171-2

- Why Does NaïveBayes Work?
  - P. Langley and S. Sage. Induction of selective bayesian classifiers. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 399–406, 1994. Available from `http://lyonesse.stanford.edu/~langley/papers/select.uai94.ps`
  - Pedro Domingos and Michael J. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997. URL `citeseer.ist.psu.edu/domingos97optimality.html`

References (5)

- Machine learning
  - R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, 1992. ISBN: 1558602380
- Misc
  - D.H. Meadows, D.L. Meadows, J. Randers, and W.W. Behrens. *The Limits to Growth*. Potomac Associates, 1972
- MDL & MML
  - C.S. Wallace and D.M. Boulton. An information measure for classification. *Computer Journal*, vol 11.2:185–194, 1968
  - R.A. Baxter and J.J. Oliver. MDL and MML: Similarities and differences. Technical report, Computer Science, Monash University, Melbourne, Australia, March 1995. Available from `http://citeseer.ist.psu.edu/baxter95mdl.html`

And so...



Success Despite Complexity

- Maybe....
  - The world is not as complex as we thing
  - Real world models clump, have collars.
  - Possible to quickly search, find ways to select for preferred states.
- Ultimately, this is an empirical study.
  - Q: When does a clumping/collaring-inspired search engine succeed?
  - A: Often
    - Reports effects never seen before (limits to growth)
    - Finds solutions faster than other methods (JPL).
    - Returns tiniest theories (fss)
    - Scales to infinite data streams (TAR4.1)
- Many applications. May I try this on your problems?

A Final Word

- Sometimes the world is complex:
  - 2% optimizing air-flow over leading wing in trans-sonic range
  - synthesis of optimized code for complex engineering problems
- And sometimes it ain't.
  - Try the simple solution before the more complex.
  - Benchmark the complex against the seemingly less sophisticated.
  - Warning: your straw man may not burn

Questions? Comments?