Bayesian Anomaly Detection (BAD v0.1)

Tim Menzies tim@menzies.us
Lane Department of CS & EE, West Virginia University, USA

David Allen dave@antiform.com
Portland State University, Oregon, USA

Andres Orrego andres.orrego@ivv.nasa.gov
Global Science & Technology Inc, Fairmont, West Virginia

Motivation

- "I've tried A! I've tried B! Tell me what else…" (Bang)

Sukhoi Su-30 fighter jet crashed in Paris, June '99

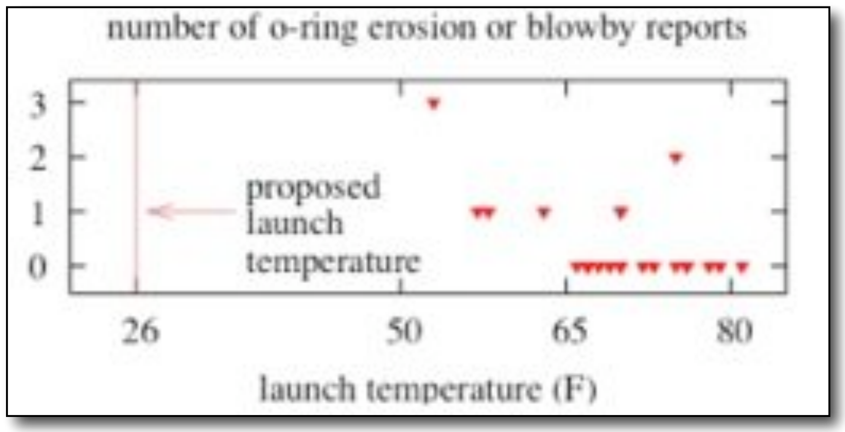
- Don't tell me what is wrong (about the software)
  - Just tell me what to do.

Context notes

- Weng-Keen: “Event detection very rare”;
  - sadly, not true in software monitoring
  - many “positive” examples
    - E.g. MAGR
    - particularly for safety-critical software
    - built using simulation-based verification:
      - Common / more common at ESA/NASA
  - some anomalies barely hide

Anomaly detection and System Safety

- Scrub launches under anomalous conditions



- Reject conclusions regarding “safe ice strikes”
  - CRATER: meteorite impact model:
    - certified for 150mph impacts of size 3 cubic inches
    - Used to argue that Columbia was not harmed on launch
  - COLUMBIA: 477mhp impact of size 1200 cubic inches

Certify software w.r.t. some "envelope of operation"

- Launch the system with an anomaly detector
- Alert if system leaves its envelope of certification
- On alert:
  - Disengage auto-pilot; wake up human pilot
  - Devote more sensor time to the anomalous event
  - If non-critical, go to safe mode
  - If critical situations, hit the eject button
  - Try and steer back to a "safe place"

- If we know a device's "envelope of certification"
  - And we know when it leaves it
  - And if a contrast set learner learns the delta between "old and safe" and "current"
  - And if that learner is constrained to only reporting the controllables
- Then that "contrast set" is a "control rule" for "get me the hell out of here"

From anomaly detection to control policies

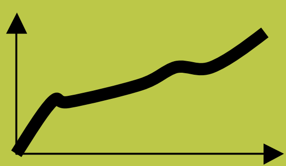
- TARx: impact rule learner
  - Consequence
    - class distribution predicted by antecedent
  - A.k.a.
    - minimal contrast set learner
    - weighted frequency association rule learning
    - impact rules
- TAR3
  - Builds conjunctions via forward select search over attributes,
    - Attributes explored in “lift order”
    - Frequency in good/frequency in bad
  - Greedy search, early stopping
- TAR4:
  - Fast heuristic Bayesian evaluation of rules

Inside a Bayesian Impact Impact Learner

O(attr*range) not O(instances)

initialized or learned incrementally

For all x= (attribute:range) **do**
LIFT1.key :=x
LIFT1.value := lift(x)
done
sort LIFT1 on value



CLIFT1= cumulative LIFT

function pick1
select lift1.value from CLIFT
(favoring high LIFT1)

Outlook			Temperature			Humidity			Windy			Play	
	Yes	No		Yes	No		Yes	No		Yes	No	Yes	No
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6			
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3			
Rainy	3/9	2/5	Cool	3/9	1/5								

Guesstimate for support

■ A new day:

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

Likelihood of the two classes

For "yes" = 2/9 × 3/9 × 3/9 × 3/9 × 9/14 = 0.0053

For "no" = 3/5 × 1/5 × 4/5 × 3/5 × 5/14 = 0.0206

Conversion into a probability by normalization:

P("yes") = 0.0053 / (0.0053 + 0.0206) = 0.205

= 0.0206 / (0.0053 + 0.0206) = 0.795

not "new example to classify" but "growing rule"

Guesstimate for yield: ∑p[H]*Uitility[H]

function learn1()
repeat Rx := Rx U pick1()
until ((Rx's lift stops growing)**OR**(Rx's support < minS))

N=20

function learnSome()
learn1() many times, **return** the N best RXs

100 times

function rx()
keep learnSome-ing till we stop seeing new treatments

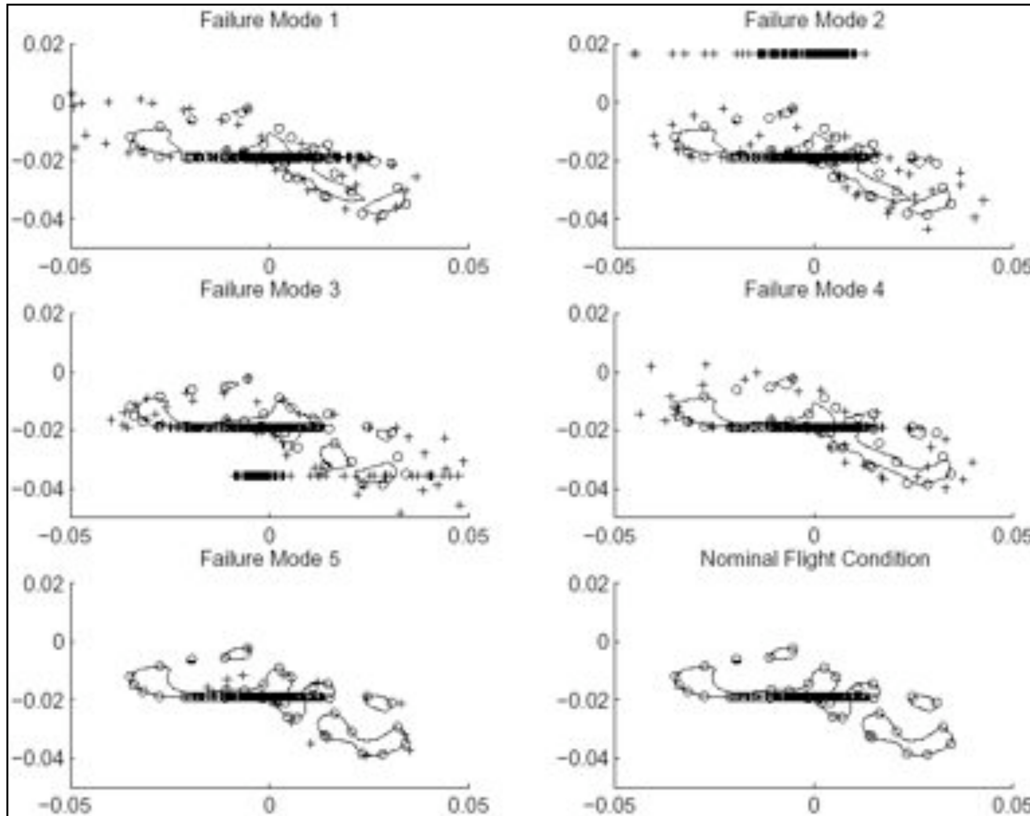
5 stale

But…

- Can we recognize the arrival of new classes?
- Assumption:
  - Devices move through modes
  - Sampling rate faster than mode changes

Constraints (a.k.a. lets make it interesting)

1. Should be able to exploit supervisor knowledge
   - Exploit known error modes
2. Should still work when unsupervised
   - Learn new modes
3. Should handle massive data sets
   - One-pass
   - Low memory footprint



Liu, Cukic, Menzies, Tools with AI, 2002

- Prior work: an SVDD solution
  - Unsatisfactory
- This work- try Bayes classifiers
  - At least: straw-man to assess other methods
  - Also, low memory/ fast runtimes

B.A.D. = bayesian anomaly detection

Bayes101

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.

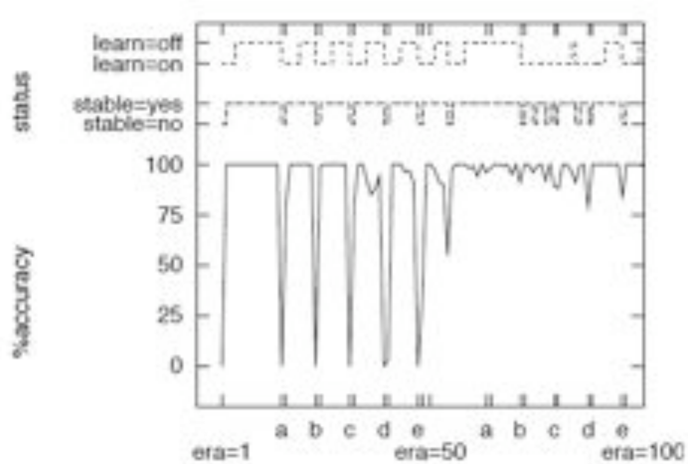
Max likelihood = 0.165

Very simple anomaly detection:
1) Process inputs in "eras" of (say) 100 instances/era
2) Track average max likelihood

http://now.unbox.org/all/trunk/doc/06/xomo2/badicml.{ppt|pdf}

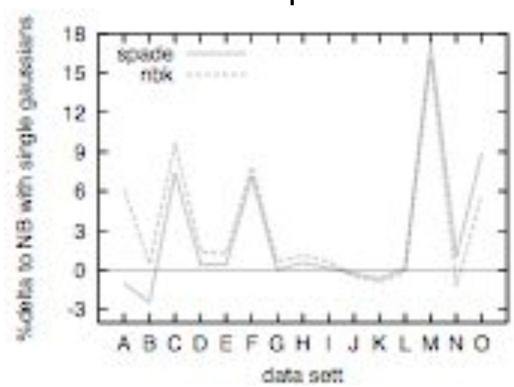
SAWTOOTH: an incremental Bayes Classifier

- SAWTOOTH:
  - Work in “windows” of 150 instances;
  - Disable learning when performance “stable”



- “Misses low-frequency events” (reviewer)
  - ?? Combine with FSS

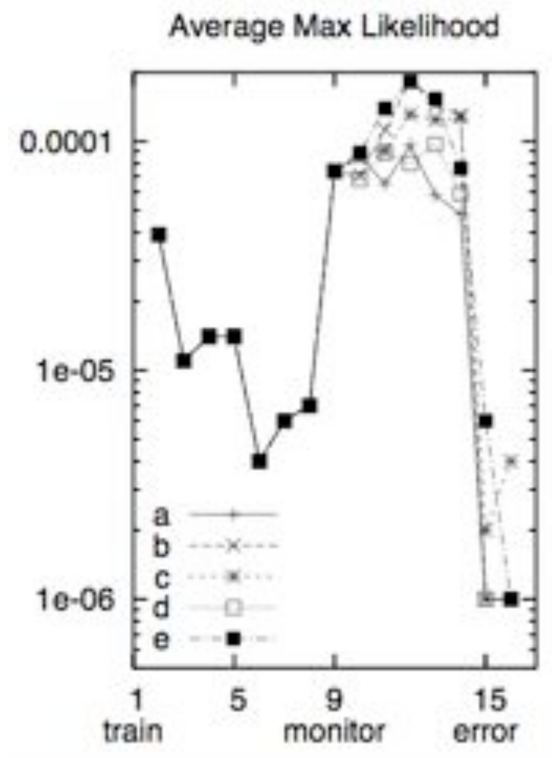
- SPADE: incremental discretizer [Orrego04]:
  - Auto-update’s SAWTOOTH’s theories
    - Shares its frequency tables
  - Like (Max-min)/N
    - but if new Max/Min older than previously seen Max/Min then…
    - …new bins are added above/below
    - If bins get too small, merge
  - Good news:
    - Runs in one pass of data
    - Very low memory overhead
    - SPADE + batch Bayes within 3% mean accuracies of N-pass discretizers



  - Bad news: “No split operator” (reviewer)

B.A.D. and a F-15 flight simulator (five different flights)

- Era size = 100 samples
  - Unsupervised learning: all classes = “class0”
- Eras:
  - 1 .. 8: Commissioning (same for each plane)
  - 9 .. 13: Fly five different missions
  - 14: Inject different errors into each plane
- Result:Massive drop in av. Max. likelihood
  - I.e. very clear indication that something novel is happening to the planes



One-sided classification: B.A.D. had no a priori knowledge of error modes

B.A.D. on 25 UCI data sets

- Emulates a device with several major modes
- Take data from UCI
  - “Blocked” data into contiguous “runs” of classes
  - Can we detect start of “novel” blocks: a class never seen before?
- Don’t expect an incremental unsupervised learner to out-perform a batch supervised learner
  - Test excludes classes that a batch classifier finds with PD < T%

Results

Data Set	Classes	PD%	FP%
segment	5	88.0	0.0
soybean	18	94.4	0.6
letter	11	99.1	0.0
audiology	4	100.0	0.0
ionosphere	2	100.0	0.0
kr-vs-kp	2	100.0	0.0
mushroom	2	100.0	0.0
primary-tumor	3	100.0	0.0
splice	3	100.0	0.0
vote	2	100.0	0.0
vowel	5	100.0	0.0
waveform-5000	2	100.0	0.0
anneal	5	100.0	2.0
autos	3	100.0	3.3
hypothyroid	2	100.0	5.0
	average:	98.8	0.7

Figure 7. Minumum PD = 0.8, z-test α = 0.00001.

Data Set	Classes	PD%	FP%
credit-g	2	30.0	35.0
breast-cancer	2	50.0	0.0
sick	2	50.0	0.0
waveform-5000	3	53.3	10.0
diabetes	2	55.0	10.0
primary-tumor	10	56.0	10.0
vehicle	3	56.7	0.0
vowel	11	74.5	4.5
colic	2	75.0	20.0
letter	26	82.7	0.8
autos	6	86.7	0.0
splice	3	93.3	6.7
soybean	19	94.7	0.5
kr-vs-kp	2	95.0	0.0
segment	6	95.0	0.0
hypothyroid	3	96.7	0.0
anneal	5	100.0	0.0
audiology	8	100.0	0.0
credit-a	2	100.0	0.0
heart-c	2	100.0	0.0
heart-h	2	100.0	0.0
ionosphere	2	100.0	0.0
mushroom	2	100.0	0.0
sonar	2	100.0	0.0
vote	2	100.0	0.0
	average:	81.8	3.9

Figure 8. Minimum PD = 0.2, z-test α = 0.00001.

Surprisingly large α value for the z-tests comparisons

Discussion

- Current experience:
  - we can build anomaly detection and controller in a single framework
  - can also generate test cases
- Success of very simple anomaly detection rig:
  - Incremental Bayes classifier
  - Very simple incremental discretion may suffice
    - **Caveat: since procedural programming monitoring has high frequency "positive" events**
- Simplicity has its virtues
  - One-pass
  - Low memory footprint
  - Can recognize new modes
  - Can be initialized with old modes
  - ?? IR for anomaly detection

- Need more case studies
  - ARES / TRICK simulation of NASA's CEV GNC system
  - **Extensions to non-relational data**
  - **Not Bayes, but Webb's AODE**
  - **Rahul's cascaded detectors & "ping"-ing on v. small training examples**
- Needs a rule generator
  - B.A.D. reports anomalies,
    - Can't describe then
  - Standard problem of explanation of mathematical systems
- Combining technologies
  - Use B.A.D. to find anomalies
  - Use (say) WSARE3to generate Bayes nets to visualize the before/after pattern
- **Is this problem best viewed NOT as "event detection" but as "active learning"?**

Questions? Comments?

Some context notes

- **domainKnowledge** -> **model**

- {**model**,**data**} -> **eventDetection**
  -> **interestingnessDectector** -> {**feedback**,**action**}

- **feedback** -> {**data**,**domainKnowledge**}

This talk:

- **Data** come from a running program
- **InterestingnessDetector** =
  - track average max. likelihood in an incremental Bayes classifier
- **Feedback**: very simple (update Bayes classifier)
- **Action**: report control rule for observables that can drive software back to "non-anomalous" zone

Tools:

- One-sided classification : seek things that aren't what we have seen before

More context notes

- Rahul: “Interactive event detection”
  - Me : runtime monitoring and control of procedural software
- James: “I’m an imposter since I’m working on the easiest image anomaly problem”
  - Me: me to!
- Weng-Keen: “New forms of interesting events appear frequently”
  - Absolutely
- Weng-Keen: “Event detection very rare”; sadly, not true in software
  - The “MAGR” example
  - So we have many “positive” examples (particularly for safety-critical software build using simulation-based verification: common/rare at ESA/NASA)
  - And some of the anomalies aren’t hiding