

```

1 ---
2 ---
3 ---
4 ---
5 ---
6 ---
7 ---
8
9 local the,help = {},[[
10 brknbad: explore the world better, explore the world for good.
11 (c) 2022, Tim Menzies
12
13
14      Ba 56      Bad <----- planning= (better - bad)
15      |         |         monitor = (bad - better)
16      |         |         v
17      |         |         Better
18      |         |         4
19      |         |         .-----
20
21 USAGE:
22   ./bnb [OPTIONS]
23
24 OPTIONS:
25   -bins -b max. number of bins          = 16
26   -best -B best set                      = .5
27   -cohen -c cohen                        = .35
28   -far -F how far to go for far          = .9
29   -goal -g goal                          = recurrence-events
30   -K -K manage low class counts          = 1
31   -leaves -l number of items in leaves   = .5
32   -M -M manage low evidence counts       = 2
33   -p -p coefficient on distance          = 2
34   -rest -R rest is -R*best               = 4
35   -some -s sample size for distances     = 512
36   -seed -S seed                         = 10019
37   -wait -w wait                         = 10
38
39 OPTIONS (other):
40   -dump -d on error, dump stack+exi      = false
41   -file -f file name                     = ../etc/data/breastcancer.csv
42   -help -h show help                     = false
43   -todo -t start up action                = nothing
44 ]]
45
46 local used={}
47 local function cli(long,key,short,x)
48   assert(not used[short], "repeated short flag["..short.."]")
49   used[short]=short
50   for n,flag in ipairs(arg) do
51     if flag==short or flag==long then
52       x = x=="false" and true or x=="true" and "false" or arg[n+1] end end
53   if type(x)=="string" then
54     x = x:match("^%s*(-)%s*$")
55     if x=="true" then x=true
56     elseif x=="false" then x=false
57     else x=tonumber(x) or x end end
58   the[key]=x end
59
60 help:gsub("\n ([-]|([%s+])|[%s+]+(-[%s+])|[%s+]*s([%s+])",cli)
61 if the.help then os.exit(print(help)) end
62 return the
63
64 local b4={} for k,_ in pairs(_ENV) do b4[k]=k end
65 local R = require
66 local the = R"the"
67 local lib = R"lib"
68 local abcd = R"abcd"
69 local bin, rule = R"bin", R"rule"
70 local num, sym = R"num", R"sym"
71 local ako, egs, seen, cluster = R"ako", R"egs", R"seen", R"cluster"
72 local learn101, learn201, learn301 = R"learn101", R"learn201", R"learn301"
73
74 local ish,items,o,oo,powerset = lib.ish,lib.items,lib.o,lib.oo,lib.powerset
75 local rnds, rnd = lib.rnds, lib.rnd
76
77 -- ## Conventions:
78 -- lower case for instance methods, leading upper case for class methods (e.g.
79 -- start ach file witha sime new method that lists the attributes
80 -- creation, management of sets of instances)
81
82 ---
83 ---
84 ---
85 ---
86 ---
87 ---
88 ---
89 ---
90 ---

```

```

91 ---
92 ---
93
94 local fails=0
95 local function ok(test,msg)
96   print("", test and "PASS" or "FAIL",msg or "")
97   if not test then
98     fails = fails+1 ; if the and the.dump then assert(test,msg) end end end
99
100 local go={}
101 function go.copy( t,u)
102   t={a={b={c=10},d={e=200}}, f=300}
103   u= lib.copy(t)
104   t.a.b.c= 20
105   print(u.a.b.c)
106   oo(t)
107   oo(u)
108   lib.dent(u) end
109
110 function go.rnd()
111   oo(rnds(23.111111)) end
112
113 function go.collect()
114   local function aux(x,y) return x*y end
115   oo(lib.collect({10,20,30},aux)) end
116
117 function go.ent()
118   local a,b = lib.ent{a=9,b=7}
119   ok(ish(lib.ent{a=9,b=7}, .98886), "entropy") end
120
121 function go.items()
122   for x in items{10,20,30} do oo(x) end
123   local n=0
124   for x in items(the.file) do n=n+1; if n<=5 then oo(x) end end end
125
126 function go.powerset()
127   for _,x in pairs(powerset{10,20,30,40,50}) do oo(x) end end
128
129 function go.many( t)
130   local o,many=lib.o,lib.many
131   t={};for j = 1,1000 do t[#t+1] = j end
132   print(900,"+", o(many(t, 10, 900)))
133   print(1,100, o(many(t, 10, 1, 100)))
134   print(300,700, o(many(t, 10, 300, 700))) end
135
136 function go.new()
137   lib.dent(seen.new{"Name","Age","gender","Weight-"}) end
138
139 -- function go.clone( i,t,best,rest, x)
140 -- i={rows={},cols=nil}
141 -- the.file = "../etc/data/auto93.csv"
142 -- bins=explain(the.file)
143 -- for _,row in pairs(i.rows) do
144 --   x=row[col].at end end
145
146 function go.egs( i)
147   i=egs.Init(the.file)
148   ok(7==i.cols.x[2].has["lt40"], "counts")
149   ok(286 == #i.rows,"egs") end
150
151 function go.dist( i)
152   local any= lib.any
153   i=egs.Init(the.file)
154   local yes=true
155   for j=1,1000 do
156     if (j % 50)==0 then io.write(".") end
157     local d = egs.dist(i, any(i.rows), any(i.rows))
158     yes = yes and d>=0 and d<=1 end
159   ok(yes, "dist") end
160
161 function go.half( i)
162   the.file = "../etc/data/diabetes.csv"
163   i = egs.Init(the.file)
164   local lefts,rights,left,right,border,c= egs.half(i)
165   print("rows",#i.rows)
166   ok(384 == #lefts.rows, "left")
167   ok(384 == #rights.rows, "rights") end
168
169 function go.cluster( i)
170   the.file = "../etc/data/diabetes.csv"
171   i = egs.Init(the.file)
172   cluster.show(cluster.new(i))
173 end
174
175 local function qq(i,q)
176   print(q[1], fmt ("%15s =%-8s best= %s/%s rest= %s/%s",
177     i.cols[q[2]].name, q[3],q[4],q[5],q[6],q[7])) end
178
179 function go.nbl()
180   local i = nbl(the.file);
181   local acc, out = score(i); print(acc); map(out,function(q) qq(i,q) end) end
182
183 function go.nb2()
184   the.file = "../etc/data/diabetes.csv"
185   the.goal = "positive"
186   local i = nb2(the.file);
187   abcd(i.log,true) end
188
189 function go.nb2a()
190   the.file = "../etc/data/diabetes.csv"
191   the.goal = "positive"
192   for _,bins in pairs{2,5,9} do
193     print(bins)
194     the.bins = bins
195     local i = nb2(the.file);
196     abcd(i.log,true) end end
197
198 function go.bins( t)
199   local t,n = {},30
200   for j=1,n do push(t, {x=j,y=j<.6*n and 1 or j<.8*n and 2 or 3}) end
201   map(bins(t,20),oo) end
202
203 function go.nb3()
204   the.file = "../etc/data/diabetes.csv"
205   the.goal = "positive"
206   the.bins = 16
207   local i = nb3(the.file);
208   abcd(i.log,true)
209   local acc, out = score(i); map(out,function(q) qq(i,q) end) end
210
211 fails = 0
212 local defaults=lib.copy(the)
213 local todos = defaults.todo == "all" and slots(go) or {defaults.todo}
214 for _,todo in pairs(todos) do
215   the = lib.copy(defaults)
216   math.randomseed(the.seed or 10019)
217   if go[todo] then go[todo]() end end
218
219 for k,v in pairs(_ENV) do if not b4[k] then print("???",k,type(v)) end end
220 os.exit(fails)
221

```

```

222 ---
223 ---
224 ---
225 ---
226 ---
227 local lib = require"lib"
228 local has2,has3,inc,inc2,sort = lib.has2,lib.has3,lib.inc,lib.inc2,lib.sort
229
230 local nb={}
231 function nb.new() return {
232   h={}, nh=0,e={}, n=0, wait=the.wait,
233   bests=0,rests=0,best={}, rest={},log=log or {}, cols={} end
234
235 function nb.classify(i,t,use)
236   local hi,out = -1
237   for h,val in pairs(i.h) do
238     local prior = ((i.h[h] or 0) + the.K)/(i.n + the.K*i.nh)
239     local l = prior
240     for col,x in pairs(t) do
241       if x ~= "?" and i.cols[col].indep then
242         l=l*(has3(i.e,col,x,h) + the.M*prior)/((i.h[h] or 0) + the.M) end end
243       if l>hi then hi,out=l,h end end
244     return out end
245
246 function nb.test(i,t)
247   if i.n > the.wait then push(i.log,{want=t[#t], got=nb.classify(i,t)}) end end
248
249 function nb.train(i,t)
250   local more,kl = false, t[#t]
251   for col,x in pairs(t) do
252     if x ~= "?" then
253       more = true
254       inc3(i.e, col, x, kl)
255       if col ~= #t then
256         inc2(kl==the.goal and i.best or i.rest, col,x) end end end
257   if more then
258     i.n = i.n + 1
259     if not i.h[kl] then i.nh = i.nh + 1 end
260     inc(i.h, kl)
261     if kl==the.goal then i.bests=i.bests+1 else i.rests=i.rests+1 end end end
262
263 function nb.score(i)
264   local acc,out=0,{}
265   for key,x in pairs(i.log) do if x.want==x.got then acc=acc+1/#i.log end end
266   for col,xns in pairs(i.best) do
267     for x,b in pairs(xns) do
268       local r = has2(i.rest,col,x)
269       local rl = r/i.rests
270       local bl = b/i.bests
271       push(out, {100*(bl^2*(bl+rl))/1, col,x,b,i.bests,r,i.rests}) end end
272   return acc, sort(out,down1) end
273
274 return function(data, log)
275   local i = nb.new()
276   for row in items(data) do
277     if #i.cols == 0
278       then i.cols=collect(row,function(j,s) return {name=s,indep=truej==#row} end)
279       else test(i,row); train(i,row) end end
280   return i end
281
282 ---
283 ---
284 ---
285 ---
286 local R=require
287 local the,lib,ako, nbl = R"the",R"lib",R"ako", R"learn101"
288 local collect = lib.collect
289
290 return function(data, log)
291   local tmp,xnums = {}
292   local function discretize(c,x, col)
293     if x ~= "?" then
294       col = xnums[c]
295       if col then x=(x - col.lo) // ((col.hi - col.lo+1E-32) / the.bins) end end
296     return x end
297
298   local function xnum(c,name)
299     if ako.xnum(name) then return {lo=1E32, hi=-1E32} end end
300
301   local function train(c,x, col)
302     col = xnums[c]
303     if col and x ~= "?" then
304       col.hi = math.max(x, col.hi)
305       col.lo = math.min(x, col.lo) end
306     return x end
307
308   for row in items(data) do
309     push(tmp, row)
310     if xnums then collect(row, train)
311     else xnums = collect(row,xnum) end end
312   for j=2,#tmp do tmp[j] = collect(tmp[j], discretize) end
313   return nbl(tmp) end
314
315 ---
316 ---
317 ---
318 ---
319 local R=require
320 local nbl,bin,lib = R"learn101", R"bin", R"lib"
321 local collect,push = lib.collect,lib.push
322
323 return function(data, log)
324   local tmp, xnums = {}
325   local function discretize(c,x, col)
326     if x ~= "?" then
327       col = xnums[c]
328       if col then
329         for _,one in pairs(col.bins) do
330           if one.lo <= x and x < one.hi then return one.id end end end end
331       return x end
332
333   local function xnum(c,name)
334     if ako.xnum(name) then return {name=name, xys={},bins={}} end end
335
336   local function train(c,x,row)
337     if xnums[c] and x ~= "?" then push(xnums[c].xys, {x=x,y= row[#row]}) end end
338
339   for row in items(data) do
340     push(tmp,row)
341     if xnums then collect(row, function(c,x) return train(c,x,row) end)
342     else xnums = collect(row,xnum) end end
343   for where,col in pairs(xnums) do
344     col.bins = bin.Xys(col.xys,where); print(col.name,#col.bins) end
345   for j=2,#tmp do tmp[j] = collect(tmp[j], discretize) end
346   return nbl(tmp) end
347

```

```

348 ---
349 ---
350 ---
351 ---
352 ---
353 local the=require"the"
354 local lib=require"lib"
355 local fmt,per,push,sort = lib.fmt, lib.per, lib.push, lib.sort
356
357 local bin={}
358 function bin.new(id,at,name,lo,hi,n,div)
359   {id=id,at=at,name=name,lo=lo,hi=hi,n=n,div=div} end
360
361 function bin.show(i,negative)
362   local x,lo,hi,big, s = i.name, i.lo, i.hi, math.huge
363   if negative then
364     if lo==hi then s=fmt("%s!=%s",x,lo)
365     elseif hi==big then s=fmt("%s< %s",x,lo)
366     elseif lo==big then s=fmt("%s>=%s",x,hi)
367     else s=fmt("%s< %s and %s>=%s",x,lo,x,hi) end
368   else
369     if lo==hi then s=fmt("%s==%s",x,lo)
370     elseif hi==big then s=fmt("%s>=%s",x,lo)
371     elseif lo==big then s=fmt("%s< %s",x,hi)
372     else s=fmt("%s<=%s< %s",lo,x,hi) end end
373   return s end
374
375 function bin.select(i,row)
376   local x, lo, hi = row[i.at], i.lo, i.hi
377   return x=="?" or lo == hi and lo == x or lo <= x and x < hi end
378
379 --- class methods
380
381 function bin.Merges(bins)
382   local j,n,new = 0,length(bins),{}
383   while j <= n do
384     j=j+1
385     a=bins[j]
386     if j < n then
387       b = bins[j+1]
388       if a.hi == b.lo then
389         a.hi = b.hi
390         a.div = (a.div*a.n + b.div*b.n)/(a.n+b.n)
391         a.n = a.n + b.n
392         j = j + 1 end end
393     push(new,a) end
394     return #new < #bins and bin.Merges(new) or bins end
395
396 local argmin
397 function bin.Xys(xys,at,name)
398   xys = sort(xys, upx)
399   local triviallySmall = the.cohen*(per(xys,.9).x - per(xys, .1).x)/2.56
400   local enoughItems = #xys / the.bins
401   local out = {}
402   argmin(1,xys, xys, triviallySmall, enoughItems, -math.huge, at.name, out)
403   out[#out].hi = math.huge
404   return out end
405
406 function argmin(lo, hi, xys, triviallySmall, enoughItems, b4, at, name,out)
407   local function add(f,z) f[z] = (f[z] or 0) + 1 end
408   local function sub(f,z) f[z] = f[z] - 1 end
409   local lhs, rhs, cut, div, xpect, xy = {},{}
410   for j=lo,hi do add(rhs, xys[j].y) end
411   div = ent(rhs)
412   if hi-lo+1 > 2*enoughItems then
413     for j=lo,hi - enoughItems do
414       add(lhs, xys[j].y)
415       sub(rhs, xys[j].y)
416       local n1,n2 = j - lo +1, hi-j
417       if n1 > enoughItems and n2 > enoughItems and
418         xys[j].x ~ xys[j+1].x and -- there is a break here
419         xys[j].x - xys[lo].x > triviallySmall and
420         xys[hi].x - xys[j].x > triviallySmall
421       then xpect = (n1*ent(lhs) + n2*ent(rhs)) / (n1+n2)
422         if xpect < div then -- cutting here simplifies things
423           cut, div = j, xpect end end end
424     end -- end if
425     if cut
426     then b4 = argmin(lo, cut, xys,triviallySmall,enoughItems,b4,at,name,out)
427     b4 = argmin(cut+1,hi , xys,triviallySmall,enoughItems,b4,at,name,out)
428     else -- if no cut then the original div was never updates and is still correct
429       b4 = push(out, bin.new(#out+1,at,name,b4,xys[hi].x, hi-lo+1,div)).hi end
430     return b4 end
431
432 return bin
433
434 ---
435 ---
436 ---
437 ---
438 ---
439 local lib=require"lib"
440 local bin=require"bin"
441 local map,push,sort = lib.map, lib.push, lib.sort
442
443 local rule={}
444 function rule.new(bins, t)
445   t = {}
446   for key,one in pairs(bins) do
447     t[one.at]=t[one.at] or {}; push(t[one.at],one) end
448   return {bins=t} end
449
450 function rule.selects(i,row)
451   local function ors(bins)
452     for key,x in pairs(bins) do if bin.select(x,row) then return true end end
453     return false end
454   for at,bins in pairs(i.bins) do if not ors(bins) then return false end end
455   return true end
456
457 function rule.show(i,bins)
458   local cat, order, ors
459   cat = function(t,sep) return table.concat(t,sep) end
460   order= function(a,b) return a.lo < b.lo end
461   ors= function(bins)
462     return cat(map(bin.Merges(sort(bins,order)),bin.show)," or ") end
463   return cat(map(i.bins, ors)," and ") end
464
465 return rule
466

```

```

468 ---
469 ---
470 ---
471 ---
472 ---
473 local ako={}
474
475 ako.num = function(x) return x:find("[A-Z]" end
476 ako.goal = function(x) return x:find("[+]" end
477 ako.klass = function(x) return x:find["$" end
478 ako.ignore = function(x) return x:find["$" end
479 ako.weight = function(x) return x:find["$" and -1 or 1 end
480 ako.xnum = function(x) return ako.num(x) and not ako.goal(x) end
481
482 return ako
483 ---
484 ---
485 ---
486 ---
487 local ako = require"ako"
488
489 local num = {}
490 function num.new(at,name)
491   return (at==at or 0, name=name or "",
492     num=true, indep=false, n=0, w = ako.weight(name or ""),
493     lo=math.huge, hi=-math.huge, mu=0, m2=0, sd=0, bins={}) end
494
495 function num.add(i,x, d)
496   if x ~= "?" then
497     i.n = i.n+1
498     i.lo = math.min(x, i.lo)
499     i.hi = math.max(x, i.hi)
500     d = x - i.mu
501     i.mu = i.mu + d/i.n
502     i.m2 = i.m2 + d*(x - i.mu)
503     i.sd = ((i.m2<0 or i.n<2) and 0) or ((i.m2/(i.n - 1))^0.5) end
504   return x end
505
506 return num
507 ---
508 ---
509 ---
510 ---
511 ---
512 local sym = {}
513
514 function sym.new(at,name)
515   return (at==at or 0, name=name or "",
516     num=false, indep=false, n=0,
517     has={}, most=0, mode=nil) end
518
519 function sym.add(i,x)
520   if x ~= "?" then
521     i.n = i.n + 1
522     i.has[x] = 1 + (i.has[x] or 0)
523     if i.has[x] > i.most then
524       i.mode,i.most = x,i.has[x] end end
525   return x end
526
527 return sym
528 ---
529 ---
530 ---
531 ---
532 local R=require
533 local ako,lib,sym,num = R"ako",R"lib",R"sym",R"num"
534 local norm,o,oo,push = lib.norm, lib.o, lib.oo, lib.push
535
536 local seen = {}
537 function seen.new(names)
538   return seen.init({names=names, klass=nil,xy={}, x={}, y={}},names) end
539
540 function seen.init(i, names)
541   for at,name in pairs(names) do
542     local now = (ako.num(name) and num.new or sym.new)(at,name)
543     push(i.xy, now)
544     if not ako.ignore(name) then
545       if not ako.goal(name) then now.indep = true end
546       if ako.klass(name) then i.klass=now end
547       push(now.indep and i.x or i.y, now) end end
548   return i end
549
550 function seen.add(i,row)
551   for _,col in pairs(i.xy) do
552     (col.nump and num or sym).add(col, row[col.at]) end
553   return row end
554
555 function seen.better(i,row1,row2)
556   local s1, s2, n, e = 0, 0, #i.y, math.exp(1)
557   for _,col in pairs(i.y) do
558     local a = norm(col.lo, col.hi, row1[col.at])
559     local b = norm(col.lo, col.hi, row2[col.at])
560     s1 = s1 - e^(col.w * (a - b) / n)
561     s2 = s2 - e^(col.w * (b - a) / n) end
562   return s1 / n < s2 / n end
563
564 return seen
565

```

```

566 ---
567 ---
568 ---
569 ---
570 ---
571 local R = require
572 local the,seen,lib = R"the", R"seen", R"lib"
573 local map,sort,upl = lib.map,lib.sort,lib.upl
574 local items,push,slice = lib.items,lib.push,lib.slice
575 local any,many,cos = lib.any, lib.many, lib.cosine
576 local o,oo,per,norm = lib.o, lib.oo, lib.per, lib.norm
577
578 ---
579 ---
580 local eggs={}
581 function eggs.new() return {rows={}, cols=nil} end
582
583 function eggs.Init(data, i)
584   i = eggs.new()
585   for row in items(data) do
586     if not i.cols then i.cols=seen.new(row) else eggs.add(i,row) end end
587   return i end
588
589 function eggs.add(i,row)
590   push(i.rows, seen.add(i.cols, row)) end
591
592 ---
593 ---
594 function eggs.mid(i,cols)
595   local function mid(col) return col.nump and col.mu or col.mode end
596   return map(cols or i.cols.y, mid) end
597
598 function eggs.div(i,cols)
599   local function div(col) return col.nump and col.sd or ent(col.has) end
600   return map(cols or i.cols.y, div) end
601
602 function eggs.clone(old,rows)
603   local i={rows={}, cols=seen.new(old.cols.names)}
604   for key,row in pairs(rows or {}) do seen.add(i.cols,row) end
605   return i end
606
607 ---
608 ---
609 function eggs.dist(i,row1,row2)
610   local function sym(c,x,y) return x==y and 0 or 1 end
611   local function num(c,x,y)
612     if x=="?" then y = norm(c.lo, c.hi, y); x==y<.5 and 1 or 0
613     elseif y=="?" then x = norm(c.lo, c.hi, x); y==x<.5 and 1 or 0
614     else
615       x,y = norm(c.lo, c.hi, x), norm(c.lo, c.hi, y) end
616     return math.abs(x-y) end
617   local function dist(c,x,y)
618     return x=="?" and y=="?" and 1 or (c.nump and num or sym)(c,x,y) end
619   local d, n = 0, #i.cols.x
620   for key,c in pairs(i.cols.x) do d=d+dist(c, row1[c.at], row2[c.at])^the.p end
621   return (d/n)^(1/the.p) end
622
623 function eggs.neighbors(i, r1, rows)
624   return sort(map(rows or i.rows,
625     function(r2) return {egs.dist(i,r1,r2),r2} end), upl) end
626
627 function eggs.half(i, rows)
628   local project,far,some,left,right,c,lefts,rights,border
629   rows = rows or i.rows
630   far = function(r,t) return per(egs.neighbors(i,r,t), the.far)[2] end
631   project = function(r)
632     return {cos(egs.dist(i,left,r), egs.dist(i,right,r),c),r} end
633   some = many(rows, the.some)
634   left = far(any(some), some)
635   right = far(left, some)
636   c = egs.dist(i,left,right)
637   lefts,rights = egs.clone(i), egs.clone(i)
638   for n, projection in pairs(sort(map(rows,project), upl)) do
639     if n==#rows//2 then border = projection[1] end
640     egs.add(n <= #rows//2 and lefts or rights, projection[2]) end
641   return lefts, rights, left, right, border, c end
642
643 ---
644 ---
645 function eggs.bestRest(i)
646   i.rows = sort(i.rows, function(a,b) return seen.better(i.cols,a,b) end)
647   local n = (#i.rows)^the.best
648   return slice(i.rows, 1, n) -- top n things
649   many(i.rows, n-the.rest, n+1) end -- some sample of the rest
650
651 function eggs.Contrasts(i, rows1, rows2)
652   local function contrast(col)
653     local function asBin(x,ys, n,div)
654       n,div = ent(ys)
655       return bin.new(id, col.at, col.name, x, x, n, div) end
656     local symbols, xys, x = {},{}
657     for klass,rows in pairs{rows1,rows2} do
658       for key,row in pairs(rows) do
659         x = row[col.at]
660         if x ~= "?" then
661           if not col.nump then inc2(symbols,x,klass) end
662           push(xys, {x=x, y=klass}) end end end
663     return col.nump and bins(xys, col.at) or collect(symbols, asBin) end
664   local out, tmp = {}
665   for key,col in pairs(i.cols.x) do
666     tmp = contrast(col)
667     if #tmp > 1 then
668       for key,one in pairs(tmp) do push(out, one) end end end
669   return out end
670
671 function eggs.xplain(i)
672   best, rest = egs.bestRest(i)
673   return egs.contrasts(i, best,rest) end
674
675 return eggs
676
677 ---
678 ---
679 ---
680 ---
681 local R = require
682 local the,egs,lib = R"the", R"egs", R"lib"
683 local o,fmt,rnds = lib.o, lib.fmt, lib.rnds
684
685 local cluster={}
686 function cluster.new(top,egs1, i,lefts,rights)
687   egs1 = egs1 or top
688   i = {egs=egs1, top=top, rank=0}
689   lefts, rights, i.left, i.right, i.border, i.c = egs.half(top, egs1.rows)
690   if #egs1.rows >= 2*(#top.rows)^the.leaves then
691     if #lefts.rows < #egs1.rows then
692       i.lefts = cluster.new(top, lefts)
693       i.rights = cluster.new(top, rights) end end
694   return i end
695
696 function cluster.leaf(i) return not (i.lefts or i.rights) end
697
698 function cluster.show(i, pre, front)
699   pre = pre or ""
700   local front = fmt("%s%s", pre, #i.egs.rows)
701   if cluster.leaf(i)

```

```

702 then print (fmt("%-20s", front, o(rnds(egs.mid(i.egs,i.egs.cols.y))))))
703 else print (front)
704 if i.lefts then cluster.show(i.lefts, " |"..pre)
705 if i.rights then cluster.show(i.rights, " |"..pre) end end end end
706 return cluster
707
708

```

```

709 ---
710 ---
711 ---
712 ---
713
714 local lib=require"lib"
715
716 local abcd={}
717
718 function abcd.new(data,rx)
719   {data= data or "data",rx= rx or "rx",
720     known={},a={},b={},c={},d={},yes=0,no=0} end
721
722 function abcd.exists(i,x, new)
723   new = not i.known[x]
724   lib.inc(i.known,x)
725   if new then
726     i.a[x]=i.yes + i.no; i.b[x]=0; i.c[x]=0; i.d[x]=0 end end
727
728 function abcd.report(i, p,out,a,b,c,d,pd,pf,pn,f,acc,g,prec)
729   p = function (z) return math.floor(100*z + 0.5) end
730   out= {}
731   for x,xx in pairs( i.known ) do
732     pd,pf,pn,prec,g,f,acc = 0,0,0,0,0,0,0
733     a= (i.a[x] or 0); b= (i.b[x] or 0); c= (i.c[x] or 0); d= (i.d[x] or 0);
734     if b+d > 0 then pd = d / (b+d) end
735     if a+c > 0 then pf = c / (a+c) end
736     if a+c > 0 then pn = (b+d) / (a+c) end
737     if c+d > 0 then prec = d / (c+d) end
738     if 1-pf+pd > 0 then g=2*(1-pf) * pd / (1-pf+pd) end
739     if prec*pd > 0 then f=2*prec*pd / (prec + pd) end
740     if i.yes + i.no > 0 then
741       acc= i.yes / (i.yes + i.no) end
742     out[x] = {data=i.data,rx=i.rx,num=i.yes+i.no,a=a,b=b,c=c,d=d,acc=p(acc),
743               prec=p(prec), pd=p(pd), pf=p(pf),f=p(f), gp=p(g), class=x} end
744   return out end
745
746 function abcd.pretty(t)
747   print" "
748   local s1 = "%10s| %10s| %4s| %4s| %4s| %4s "
749   local s2 = "| %3s| %3s| %4s| %4s| %3s| %3s| "
750   local d,s = "----", (s1 .. s2)
751   print (fmt(s,"db","rx","a","b","c","d","acc","pd","pf","prec","f","g"))
752   print (fmt(s,d,d,d,d,d,d,d,d,d,d,d,d))
753   for key,x in pairs(lib.slots(t)) do
754     local u = t[x]
755     print (lib.fmt(s.." %s", u.data,u.rx,u.a, u.b, u.c, u.d,
756                   u.acc, u.pd, u.pf, u.prec, u.f, u.g, x)) end end
757
758 function abcd.adds(gotwants, show,data, rx)
759   local i = abcd.is(data,rx)
760   for key,one in pairs(gotwants) do
761     abcd.exists(i,one.want)
762     abcd.exists(i,one.got)
763     if one.want == one.got then i.yes=i.yes+1 else i.no=i.no+1 end
764     for x,xx in pairs(i.known) do
765       if one.want == x
766       then lib.inc(one.want == one.got and i.d or i.b, x)
767       else lib.inc(one.got == x and i.c or i.a, x) end end end
768   return show and abcd.pretty(abcd.report(i)) or abcd.report(i) end
769
770 return abcd.adds
771

```

```

772 ---
773 ---
774 ---
775 ---
776 ---
777 local lib={}
778
779 ---
780 ---
781 ---
782 function lib.per(t,p) return t[ (p or .5)*#t//1 ] end
783
784 function lib.ent(t)
785 local n=0; for _,m in pairs(t) do n = n+m end
786 local e=0; for _,m in pairs(t) do if m>0 then e = e+m/n*math.log(m/n,2) end end
787 return -e,n end
788
789 function lib.norm(lo,hi,x) return math.abs(hi-lo)<1E-9 and 0 or (x-lo)/(hi-lo) end
790
791 function lib.cosine(a,b,c)
792 return math.max(0,math.min(1, (a^2+c^2-b^2)/(2*c+1E-32))) end
793
794 ---
795 ---
796 ---
797 function lib.ish(x,y,z) return math.abs(x-y) <= (z or 0.001) end
798
799 ---
800 ---
801 ---
802 ---
803 function lib.inc(f,a,n) f=f or {};f[a]=(f[a] or 0) + (n or 1) return f end
804 function lib.inc2(f,a,b,n) f=f or {};f[a]=lib.inc(f[a] or {},b,n); return f end
805 function lib.inc3(f,a,b,c,n) f=f or {};f[a]=lib.inc2(f[a] or {},b,c,n);return f end
806
807 function lib.has(f,a) return f[a] or 0 end
808 function lib.has2(f,a,b) return f[a] and lib.has(f[a],b) or 0 end
809 function lib.has3(f,a,b,c) return f[a] and lib.has2(f[a],b,c) or 0 end
810
811 ---
812 ---
813
814 lib.unpack = table.unpack
815
816 function lib.push(t,x) t[1 + #t] = x; return x end
817
818 function lib.powerset(s)
819 local function aux(s)
820 local t = {}
821 for i = 1, #s do
822 for j = 1, #t do
823 t[#t+1] = {s[i], lib.unpack(t[j])} end end
824 return t end
825 return lib.sort(aux(s), function(a,b) return #a < #b end) end
826
827 ---
828 ---
829 ---
830
831 function lib.map(t, f, u)
832 u={}; for k,v in pairs(t) do u[1+#u]=f(v) end; return u end
833 function lib.collect(t,f,u)
834 u={}; for k,v in pairs(t) do u[k]=f(k,v) end; return u end
835 function lib.copy(t, u)
836 if type(t) ~= "table" then return t end
837 u={}; for k,v in pairs(t) do u[lib.copy(k)] = lib.copy(v) end; return u end
838
839 ---
840 ---
841 ---
842
843 function lib.sort(t,f) table.sort(t,f); return t end
844
845 function lib.upx(a,b) return a.x < b.x end
846 function lib.upl(a,b) return a[1] < b[1] end
847 function lib.downl(a,b) return a[1] > b[1] end
848
849 function lib.slots(t, u)
850 local function public(k) return tostring(k):sub(1,1) ~= "_" end
851 u={}; for k,v in pairs(t) do if public(k) then u[1+#u]=k end end
852 return lib.sort(u) end
853
854 ---
855 ---
856
857 function lib.any(a,lo,hi)
858 lo,hi = lo or 1, hi or #a; return a[ (lo+(hi-lo)*math.random())//1 ] end
859
860 function lib.many(a,n,lo,hi, u)
861 u={}; for j=1,n do lib.push(u, lib.any(a,lo,hi)) end; return u end
862
863 function lib.slice(a,lo,hi, u)
864 u,lo,hi = {},lo or 1,hi or #a; for j=lo,hi do u[1+#u]=a[j] end; return u end

```

```

865 ---
866 ---
867 ---
868 ---
869 ---
870 function lib.words(s,sep, t)
871 sep="([^\s\.\-])"
872 t={}; for y in s:gmatch(sep) do t[1+#t] = y end; return t end
873
874 function lib.things(s) return lib.map(lib.words(s), lib.thing) end
875
876 function lib.thing(x)
877 x = x:gmatch("%s*(-)%s*$")
878 if x=="true" then return true elseif x=="false" then return false end
879 return tonumber(x) or x end
880
881 function lib.items(src,f)
882 local function file(f)
883 src,f = io.input(src),(f or lib.things)
884 return function(x) x=io.read()
885 if x then return f(x) else io.close(src) end end end
886 local function tbl(t, x)
887 x,f = 0, f or function(z) return z end
888 return function() if x< #src then x=x+1; return f(src[x]) end end end
889 if src then
890 return type(src) == "string" and file(f) or tbl() end end
891
892 ---
893 ---
894 ---
895
896 lib.fmt = string.format
897
898 function lib.oo(t) print(lib.o(t)) end
899
900 function lib.o(t, seen, u)
901 if type(t)~="table" then return tostring(t) end
902 seen = seen or {}
903 if seen[t] then return "..." end
904 seen[t] = t
905 local function show1(x) return lib.o(x, seen) end
906 local function show2(k) return lib.fmt("%s %s",k, lib.o(t[k],seen)) end
907 u = #t>0 and lib.map(t,show1) or lib.map(lib.slots(t),show2)
908 return (t._is or "")..{"..table.concat(u, " ").."}" end
909
910 function lib.dent(t, _seen,pre)
911 pre,seen = pre or "", seen or {}
912 if seen[t] then t= "..." end
913 if type(t)~="table" then return print(pre .. tostring(t)) end
914 seen[t]=t
915 for key,k in pairs(lib.slots(t)) do
916 local v = t[k]
917 io.write(lib.fmt("%s:%s %s",pre,k, type(v)=="table" and "\n" or " "))
918 if type(v)=="table"
919 then lib.dent(v,seen,"|"..pre)
920 else print(v) end end end
921
922 function lib.rnds(t,f)
923 return lib.map(t, function(x) return lib.rnd(x,f) end) end
924
925 function lib.rnd(x,f)
926 return lib.fmt(type(x)=="number" and (x~x//1 and f or "%5.2f") or "%s",x) end
927
928 return lib

```