

```

1 local b4={}; for k, _ in pairs(_ENV) do b4[k]=k end
2 local add,big,col,cs,v,fyl,id,is,kl,lt,map,oo
3 local per,push, rand, ranges,read, result, seed, splice, str
4 local help={
5   SAMPLE: while not end of time, look around, see what's what
6   (c) 2022 Tim Menzies, timm@ieee.org, BSD2 license
7
8   INSTALL: requires: lua 5.4+
9             download: sample.lua
10            test      : lua sample.lua -h
11
12   USAGE: lua sample.lua [OPTIONS]
13
14             defaults
15   ~~~~~~
16   -S --Seed      random number seed      = 10019
17   -h --How       optimize for (helps,hurts,tabu) = helps
18   -b --bins      number of bins           = 16
19   -m --min       min1 size (for pass1)     = .5
20   -M --Min       min2 size (for pass2)     = 10
21   -p --p         distance coefficient      = 2
22   -s --some      sample size             = 512
23
24   OPTIONS (other):
25   -f --file      csv file with data = ../etc/data/auto93.csv
26   -g --go        start up action       = nothing
27   -v --verbose   show details          = false
28   -h --help      show help             = false]]
29
30 -- ## Convert help text to settings
31 -- String to thing.
32 function read(str)
33   str = str:match("%s*(-)%s*$")
34   return math.tointeger(str) or tonumber(str) or str end
35
36 -- (1) parse 'help'<br>(2) make 'THE' settings;<br>(3) also make a 'backup'.
37 local THE, backup = {}, {}
38 help:gsub("[^-][^%s%+]"^"%n"%"%s%+)",function(key,x)
39   for n,flag in ipairs(arg) do
40     if flag=="(-"..key:sub(1,1)) or flag=="(-"..key) then
41       x = x=="false" and"true" or x=="true" and"false" or arg[n+1] end end
42   x = read(x)
43   backup[key] = x
44   THE[key] = x end)
45
46 if THE.help then os.exit(print(help:gsub("%u[%u%d]+","%27[1:31m%1270m]"))) end
47
48 -----
49 function str(i, j)
50   if type(i)=="table" then return tostring(i) end
51   if #i>0 then return table.concat(map(i,tostring),",") end
52   j={}; for k,v in pairs(i) do j[1+#j] = string.format("%s %s",k,v) end
53   table.sort(j)
54   return (i.is or "").."["..table.concat(j,",").."]" end
55
56 local _id=0
57 function is(name, t)
58   local function new(kl,...)
59     _id = _id+1
60     local x=setmetatable({id=_id,kl}; kl.new(x,...); return x end
61   t = {__tostring=str, is=name}; t.__index=t
62   return setmetatable(t, {__call=new}) end
63
64 local ROW,ROWS,SYM,NUM,SOME = is"ROW",is"ROWS",is"SYM",is"NUM",is"SOME"

```

```

65 -----
66 function col(i,has,at,txt)
67   l,n,i,at,i.txt = 0, at or 0, txt or ""
68   i,x=i.txt:find"%s" and -1 or 1
69   i.has = has end
70
71 function add(i,x,inc,fun)
72   if x == "?" then
73     inc = inc or 1
74     i.n = i.n + inc
75     fun() end
76   return end
77
78 function SOME.new(i, ...) col(i,{}); i.ok=false; end
79 function SOME.sorted(i, a)
80   if not i.ok then table.sort(i.has) end; i.ok=true; return i.has end
81 function SOME.add(i,x)
82   return add(i,x,1,function() a)
83     a = i.has
84     if #a < THE.some then i.ok=false; push(a,x)
85     elseif rand() < THE.some/i.n then i.ok=false; a[rand(#a)]=x end end) end
86
87 -----
88 function NUM.new(i, ...) col(i,SOME(),...); i.mu,i.lo,i.hi=0,big,-big end
89 function NUM.clone(i) return NUM(i.at, i.txt) end
90 function NUM.add(i,x)
91   return add(i,x,1,function() d)
92     i.has=add(x)
93     d = x - i.mu
94     i.mu = i.mu + d/i.n
95     i.hi = math.max(x, i.hi); i.lo=math.min(x, i.lo) end ) end
96
97 function NUM.merge(i,j, k)
98   local k = NUM(i.at, i.txt)
99   for _,x in pairs(i.has.has) do k:add(x) end
100  for _,x in pairs(j.has.has) do k:add(x) end
101  return k end
102
103 function NUM.mid(i) return i.mu end
104 function NUM.div(i, a)
105   a=i.has:sorted(); return (per(a, .9) - per(a, .1))/2.56 end
106
107 function NUM.bin(i,x, b)
108   b = (col.hi - col.lo)/THE.bins; return math.floor(v/b+.5)*b end
109
110 -----
111 function SYM.new(i, ...) col(i,{}); i.most, i.mode=0,nil end
112 function SYM.clone(i) return SYM(i.at, i.txt) end
113 function SYM.add(i,x,inc)
114   return add(i,x,inc,function()
115     i.has[x] = (inc or 1) + (i.has[x] or 0)
116     if i.has[x] > i.most then i.most, i.mode = i.has[x],x end end) end
117
118 function SYM.merged(i,j, k)
119   local k = SYM(i.at, i.txt)
120   for x,n in pairs(i) do k:add(x,n) end
121   for x,n in pairs(j) do k:add(x,n) end
122   return k end
123
124 function SYM.mid(i) return i.mode end
125 function SYM.div(i)
126   e=0;for k,n in pairs(i.has) do if n>0 then e=e-n/i.n*math.log(n/i.n,2)end end
127   return e end
128
129 function SYM.bin(i,x) return x end
130
131 -----
132 function SYM.score(i,want, wants,donts)
133   local b, r, z, how = 0, 0, 1/big, {}
134   how.helps= function(b,r) return (b<r or b+r < .05) and 0 or b^2/(b+r) end
135   how.hurts= function(b,r) return (r<b or b+r < .05) and 0 or r^2/(b+r) end
136   how.tabu = function(b,r) return 1/(b+r*z) end
137   for v,n in pairs(i.ys.has) do if v==want then b = b+n else r=r+n end end
138   return how[the.How](b(wants+z), r/(donts+z)) end
139
140 -----
141 function ROW.new(i,of,cells) i.of,i.cells,i.evaluated = of,cells,false end
142 function ROW.__lt(i,j, n,s1,s2,v1,v2)
143   i.evaluated = true
144   j.evaluated = true
145   s1, s2, n = 0, 0, #i.of.ys
146   for _,col in pairs(i.of.ys) do
147     v1,v2 = col:norm(i.cells[col.at]), col:norm(j.cells[col.at])
148     s1 = s1 - 2.7183^(col.w * (v1 - v2) / n)
149     s2 = s2 - 2.7183^(col.w * (v2 - v1) / n) end
150   return s1/n < s2/n end
151
152 function ROW.within(i,range, lo,hi,at,v)
153   lo, hi, at = range.xlo, range.xhi, range.ys.at
154   v = i.cells[at]
155   return v=="?" or lo==hi and v==lo or lo<v and v<hi end

```

```

156 -----
157 function ROWS.new(i,src)
158   i.has={}; i.col={}; i.ys={}; i.names={}
159   if type(src)=="string" then for row in csv( src) do i:add(row) end
160   else for _,row in pairs(src) do i:add(row) end end end
161
162 function ROWS.clone(i,with, j)
163   j=ROWS({i.names}); for _,r in pairs(with or {}) do j:add(r) end; return j end
164
165 function ROWS.add(i,row)
166   local function header( col)
167     i.names = row
168     for at,s in pairs(row) do
169       col = push(i.cols, (s:find"^[A-Z]" and NUM or SYM)(at,s))
170       if not s:find"$" then
171         if s:find"$" then i.klass = col end
172         push(s:find"^[a-z]" and i.ys or i.xs, col) end end
173   end
174   if #i.cols==0 then header(row) else
175     row = push(i.has, row.cells and row or ROW(i,row))
176     for _,col in pairs(i.cols) do col:add(row.cells[col.at]) end end end
177
178 function ROWS.bestRest(i, n,m)
179   table.sort(i.has)
180   n = #i.has
181   m = n^the.min
182   return splice(i.has, 1, m), splice(i.has, n - m) end
183
184 function ROWS.mid(i, p,t)
185   t={}; for _,col in pairs(i.ys) do t[col.txt]=col:mid(p) end; return t end
186
187 function ROWS.splits(i,bests0,rests0)
188   most,range,range1,score = 1
189   for _,col in pairs(i.xs) do
190     for _,range0 in ranges(col,bests0,rests0) do
191       score = range0:score(1,#bests0,#rests0)
192       if score==most then most,range1 = score,range0 end end end
193   local bests1, rests1 = {},{}
194   for _,rows in pairs(bests0,rests0) do
195     for _,row in pairs(rows) do
196       push(row,within(range1) and bests1 or rests1, row) end end
197   return bests1, rests1, range1 end
198
199 function ROWS.contrast(i,bests0,rests0, hows,stop)
200   stop = stop or #bests0/4
201   hows = hows or {}
202   bests1, rests1,range = i:splits(bests0,rests0)
203   if (#bests0 + #rests0) > stop and (#bests1 < #bests0 or #rests1 < #rests0) then
204     push(hows,range)
205     return i:contrast(bests1, rests1, hows, stop) end
206   return hows0,bests0 end
207
208 -----
209 function ranges(col, ...)
210   local function xpand(t)
211     for j=2,#t do t[j].xlo = t[j-1].xhi end
212     t[1].xlo, t[#t].xhi = -big, big
213     return t end
214   local function merged(i,j,min, k)
215     k = i:merge(j)
216     if i.n < min or j.n < min or k:div() <=(i.n*i:div() + j.n*j:div())/k.n then
217       local function merge(b4,min, t,j,a,b,c)
218         t,j = {},1
219         while j <= #b4 do
220           a, b = b4[j], b4[j+1]
221           if b then
222             c = merged(a.ys, b.ys, min)
223             if c then
224               j = j + 1
225               a = (xlo=a.xlo, xhi=b.xhi, ys=c) end end
226           t[#t+1] = a
227           j = j + 1 end
228         return #b4 == #t and t or merge(t,min)
229       end
230       local known,out,n,v,x = {},{},{}, 0
231       for klass,rows in pairs(...) do
232         n = n + #rows
233         for _,row in pairs(rows) do
234           v = row.cells[col.at]
235           if v ~= "?" then
236             x = col:bin(v)
237             known[x] = known[x] or push(out,{xlo=v, xhi=v, ys=col:clone()})
238             if v < known[x].xlo then known[x].xlo = v end -- works for string or num
239             if v > known[x].xhi then known[x].xhi = v end -- works for string or num
240             known[x].ys:add(klass) end end end
241       table.sort(out,lt("xlo"))
242       out = col.is=="NUM" and xpand(merge(out, n^THE.bins)) or out
243       return #out < 2 and {} or out end

```

```

244 -----
245 function ROWS.new(i,src)
246   i.has={}; i.col={}; i.ys={}; i.names={}
247   if type(src)=="string" then for row in csv( src) do i:add(row) end
248   else for _,row in pairs(src) do i:add(row) end end end
249
250 function ROWS.clone(i,with, j)
251   j=ROWS({i.names}); for _,r in pairs(with or {}) do j:add(r) end; return j end
252
253 function ROWS.add(i,row)
254   local function header( col)
255     i.names = row
256     for at,s in pairs(row) do
257       col = push(i.cols, (s:find"^[A-Z]" and NUM or SYM)(at,s))
258       if not s:find"$" then
259         if s:find"$" then i.klass = col end
260         push(s:find"^[a-z]" and i.ys or i.xs, col) end end
261   end
262   if #i.cols==0 then header(row) else
263     row = push(i.has, row.cells and row or ROW(i,row))
264     for _,col in pairs(i.cols) do col:add(row.cells[col.at]) end end end
265
266 function ROWS.bestRest(i, n,m)
267   table.sort(i.has)
268   n = #i.has
269   m = n^the.min
270   return splice(i.has, 1, m), splice(i.has, n - m) end
271
272 function ROWS.mid(i, p,t)
273   t={}; for _,col in pairs(i.ys) do t[col.txt]=col:mid(p) end; return t end
274
275 function ROWS.splits(i,bests0,rests0)
276   most,range,range1,score = 1
277   for _,col in pairs(i.xs) do
278     for _,range0 in ranges(col,bests0,rests0) do
279       score = range0:score(1,#bests0,#rests0)
280       if score==most then most,range1 = score,range0 end end end
281   local bests1, rests1 = {},{}
282   for _,rows in pairs(bests0,rests0) do
283     for _,row in pairs(rows) do
284       push(row,within(range1) and bests1 or rests1, row) end end
285   return bests1, rests1, range1 end
286
287 function ROWS.contrast(i,bests0,rests0, hows,stop)
288   stop = stop or #bests0/4
289   hows = hows or {}
290   bests1, rests1,range = i:splits(bests0,rests0)
291   if (#bests0 + #rests0) > stop and (#bests1 < #bests0 or #rests1 < #rests0) then
292     push(hows,range)
293     return i:contrast(bests1, rests1, hows, stop) end
294   return hows0,bests0 end
295
296 -----
297 function ranges(col, ...)
298   local function xpand(t)
299     for j=2,#t do t[j].xlo = t[j-1].xhi end
300     t[1].xlo, t[#t].xhi = -big, big
301     return t end
302   local function merged(i,j,min, k)
303     k = i:merge(j)
304     if i.n < min or j.n < min or k:div() <=(i.n*i:div() + j.n*j:div())/k.n then
305       local function merge(b4,min, t,j,a,b,c)
306         t,j = {},1
307         while j <= #b4 do
308           a, b = b4[j], b4[j+1]
309           if b then
310             c = merged(a.ys, b.ys, min)
311             if c then
312               j = j + 1
313               a = (xlo=a.xlo, xhi=b.xhi, ys=c) end end
314           t[#t+1] = a
315           j = j + 1 end
316         return #b4 == #t and t or merge(t,min)
317       end
318       local known,out,n,v,x = {},{},{}, 0
319       for klass,rows in pairs(...) do
320         n = n + #rows
321         for _,row in pairs(rows) do
322           v = row.cells[col.at]
323           if v ~= "?" then
324             x = col:bin(v)
325             known[x] = known[x] or push(out,{xlo=v, xhi=v, ys=col:clone()})
326             if v < known[x].xlo then known[x].xlo = v end -- works for string or num
327             if v > known[x].xhi then known[x].xhi = v end -- works for string or num
328             known[x].ys:add(klass) end end end
329       table.sort(out,lt("xlo"))
330       out = col.is=="NUM" and xpand(merge(out, n^THE.bins)) or out
331       return #out < 2 and {} or out end

```

```

244 -----
245 oo = function(i) print(str(i)) end
246 big = math.huge
247 fyi = function(...) if THE.verbose then print(...) end end
248 fmt = table.format
249 rand= math.random
250
251 function push(t,x)      t[1+#t]=x; return x end
252 function map(t,f,  u)  u={}; for k,v in pairs(t) do u[1+#u]=f(v) end return u end
253 function per(t,p)      p=p*#t//1; return t[math.max(1,math.min(#t,p))] end
254 function lt()          return function(a,b) return a[x] < b[x] end end
255
256 function splice( t, i, j, k,      u)
257 u={}; for n=(i or 1)//1, (j or #t)//1, (k or 1)//1 do u[1+#u]=t[n] end return u
258 end
259
260 function csv(csvfile)
261 csvfile = io.input(csvfile)
262 return function(s, t)
263   s=io.read()
264   if not s then io.close(csvfile) else
265     t={}; for x in s:gmatch("[^\r]+") do t[1+#t] = read(x) end
266     return t end end end

```

```

266 -----
267 local fails,go,no=0,{},{}
268
269 function go.the() fyi(str(THE)); str(THE) return true end
270
271 function go.some( s)
272   THE.some = 16
273   s=SOME(); for i=1,10000 do s:add(i) end; oo(s:sorted())
274   oo(s:sorted())
275   return true end
276
277 function go.num( n)
278   n=NUM(); for i=1,10000 do n:add(i) end; oo(n)
279   return true end
280
281 function go.sym( s)
282   s=SYM(); for i=1,10000 do s:add(math.random(10)) end;
283   return s.has[9]==1045 end
284
285 function go.csv()
286   for row in csv(THE.file) do oo(row) end; return true; end
287
288 function go.rows( rows)
289   Rows = ROWS(THE.file);
290   map(rows.ys,print); return true; end
291
292 function go.mid( r)
293   r= ROWS(THE.file)
294
295 end
296 -----
297 local going={}
298 for s,_ in pairs(go) do going[1+#going]=s end
299 table.sort(going)
300
301 for _,s in pairs(go[THE.go] and {THE.go} or going) do
302   for k,v in pairs(backup) do THE[k]=v end
303   math.randomseed(THE.Seed)
304   io.write(".")
305   result = go[s]()
306   if result ~= true then
307     fails = fails + 1
308     print("--Error",s,status) end end
309
310 for k,v in pairs(_ENV) do if not b4[k] then print("?",k,type(v)) end end
311 os.exit(fails)

```