

```

1 #!/usr/bin/env lua
2 local help = [
3 binr.lua : build rules via stochastic incremental XAI
4 (c) 2023, Tim Menzies, timm@ieee.org, mit-license.org
5
6 Options:
7   -h           Show help.
8   -e era=10    Number of rows in an era
9   -b bins=7    Number of bins for discretization.
10  -B Budget=30 Max rows to eval.
11  -l lives=5   Number of lives.
12  -r repeats=20 Number of experimental repeats.
13  -s seed=42   Random number seed.
14  -f file=../data/auto93.csv []
15
16 -- coerce(s) --> v ; Return int or float or bool or string from `s'.
17 local function coerce(s)
18   if s then return tonumber(s) or smatch("^%s(-)%s$") end end
19
20 local the{}; for k,v in help:gmatch("(%)=(%)") do the[k] = coerce(v) end
21 math.randomseed(the.seed)
22
23 local DATA, NUM, SYM, COLS, clone, adds
24
25 --# Lib
26
27 local abs,exp,sqrt,log = math.abs, math.exp, math.sqrt, math.log
28 local floor,min,max,rand,cos = math.floor,math.min,math.max,math.random,math.cos
29 local say=iowrite
30 local fmt = string.format
31
32 -- sort(t,f) --> t ; Sort `t` using function `f`.
33 local sort = function(t,f) table.sort(t,f); return t end
34
35 -- lt(f) --> f ; Return a function that sorts `a` and `b` on `f`.
36 local lt = function(f) return function(a,b) return f(a) < f(b) end end
37
38 -- cat(a) --> s ; Return a string representation of array `a`.
39 local cat = function(a) return "["..table.concat(a," ").."]" end
40
41 -- o(v) --> s ; Return a string representation of `v'.
42 local function o(v, list,dict)
43   list = function(a, u)
44     for _,v in ipairs(a) do u[1+#u] = o(v) end; return cat(u) end
45   dict = function(d, u)
46     for k,v in pairs(d) do u[1+#u] = fmt("%s=%s", k, o(v)) end
47   end
48   return type(v) == "number" and fmt(v..0) or "%."..#v..", v" or
49   type(v) == "table" and tostring(v) or (#v>0 and list or dict)(v, {}) end
50
51 -- s2a(s) --> a ; Return array of words from string `s`, split on ",".
52 local function s2a(s, a)
53   a={}; for s1 in s:gmatch("([,]+)") do a[#a+1] = coerce(s1) end; return a end
54
55 -- csv(file) --> f ; Iterator that returns rows from `file`.
56 local function csv(file, src)
57   src = assert(io.open(file))
58   return function()
59     s = src:read(); if s then return s2a(s) else src:close() end end end
60
61 -- shuffle(t) --> t ; Randomly shuffle the order of elements in `t`.
62 local shuffle = function(t, n)
63   for m#=2,z,-1 do math.random(m); t[m],t[n]=t[n],t[m] end; return t end
64
65 -- cut(a0,n,data) --> t,t ; Split `a0` at `n` (if `data` exists,split that too).
66 local function cut(a0,n, data)
67   local a1,a2 = {},{}
68   for j,v in ipairs(a0) do if j <= n then a1[#a1]=v else a2[#a2]=v end end
69   return data and clone(data,a1),clone(data,a2) or a1,a2 end
70
71 -- mode(d) --> v ; Return the most frequent key in `d'.
72 local function mode(d, v_n)
73   v_n, nil, 0
74   for v1,n1 in pairs(d) do if n1>n then v_n=v1,nl end end
75   return v_end
76
77 -- box_muller(mu, sd) --> n ; Return a random number from a Gaussian `mu`,`sd'.
78 local function box_muller(mu, sd)
79   return mu + sd * sqrt(-2 * log(rand())) * cos(6.28 * rand()) end
80
81 --## Classes
82
83 -- DATA(src) --> DATA ; Create a new DATA, populated with `src'.
84 function DATA( src ) return adds(src, {n=0,rows={},cols=nil}) end
85
86 -- clone(i,src) --> DATA ; Return a new DATA with same structure as `i'.
87 function clone(i, src) return adds(src, DATA(i.cols.names)) end
88
89 -- NUM(at,s) --> NUM ; Create a NUM object to summarize numbers.
90 function NUM(at,s)
91   return {at=at, or=s, of=s, n=0, mu=0, m2=0, sd=0, bins={}, best=(tostring(s) or ""):find="#" and 1 or 0} end
92
93 -- SYM(at,s) --> SYM ; Create a SYM object to summarize symbols.
94 function SYM(at,s) return {at=at, of=s, n=0, has={}, bins={}} end
95
96 -- COLS(row) --> COLS ; Create a COLS object from a list of column names.
97 function COLS(row, t,x,y,all)
98   x,y,all = {},{},{}
99   for n,s in ipairs(row) do
100    all[n] = (smatch"A-Z" and NUM or SYM)(n,s)
101    if not s:find("%x") then
102      t[s:find("%x")] and y or x
103      t[s:find("%x")] = all[n] end end
104
105  return (all=all, x=x, y=y, names=row) end
106
107
108

```

```

109
110  ---## Methods
111
112  -- add(i,v) --> v ; Update `i` with `v`.
113  local function add(i,v)
114    if v == "?" then return v end
115    i.mu = i.mu + v
116    elseif i.mu then
117      local d = v - i.mu
118      i.mu = i.mu + d / i.n
119      i.m2 = i.m2 + d * (v - i.mu)
120      i.sd = sqrt(max(0,i.m2)/(i.n - 1))
121    elseif i.cols then
122      for _,col in pairs(i.cols.all) do add(col, v[col.at]) end
123      i.rows[1 + #i.rows] = v end end
124
125  return v end
126
127  -- add(src,it) --> it ; Update `it` with all items from `src`.
128  function adds(src, it)
129    it = it or NUM()
130    if type(src) == "string"
131      then for row in pairs(src) do add(it,row) end
132    else for _,row in pairs(src or {}) do add(it,row) end end
133    return it end
134
135  -- norm(i,v) --> n ; Normalize `v` 0..1 using `i`.
136  local function norm(i,v)
137    return (i.has or v=="?") and v or
138      1 / (1 + math.exp(-1.7 * (v - i.mu)/(i.sd + le-32))) end
139
140  -- bin(i,v) --> n ; Normalize `v` 0..bins=1 using `i`.
141  local function bin(i,v)
142    return (i.has or v=="?") and v or floor((the.bins * norm(i,v)) end
143
144  -- dist(i, row) --> n ; Return distance of `row` to best goal (using Y cols).
145  local function disty(i, row, d)
146    d=0; for _y in pairs(i.cols.y) do d = d + (norm(y, row[y.at]) - y.best)^2 end
147
148  local function disty(i,rows)
149    return sort(rows or i.rows,
150      function(a,b) return disty(i,a) < disty(i,b) end) end
151
152  ---## Think
153
154  local function scoreGet(data, row, b, n)
155    n = 0
156    for _,col in pairs(data.cols.x) do
157      b = bin(col, row[col.at])
158      if b == "?" then
159        col.bins[b] then
160          n = n + col.bins[b].mu end end end
161    return n end
162
163  local function scorePut(data, row, b, y)
164    y = disty(data, row)
165    for _,col in pairs(data.cols.x) do
166      b = bin(col, row[col.at])
167      if b == "?" then
168        col.bins[b] = col.bins[b] or NUM(col.at, b)
169        add(col.bins[b], y) end end end
170
171  local function scoreGuess(data,m,n,rows, t)
172    t = {}
173    print((m or 1)..": "..n.." ; "..#rows)
174    for n = (m or 1)..min(#rows, n or #rows) do
175      if n <= #rows then
176        t[1+n] = (scoreGet(data, rows[n]), rows[n]) end end
177    return sort(t, function(a,b) return a[1] < b[1] end) end
178
179  local function scoresSeen(data, t,m,eps)
180    t={}; for m, row in pairs(data.rows) do t[1+m] = disty(data, row) end
181    t:sort()
182    m=1; t[1+m] = t[1+m]/2.56 * 0.35
183    print(fmt("%2.2f%2.2f%2.2f%2.2f%2.2f%2.2f%2.2f%2.2f", t[1:m], t[2*m], t[3*m], t[4*m], t[5*m], t[6*m], t[7*m], t[8*m], eps))
184    return data,eps end
185
186  local function score(data,eps, labelled,rows,bestRow,besty,loves,best,y,lives,n)
187    besty = le32
188    lives = loves or the.lives
189    seen = {}
190    n=0;
191    for m,n, row in pairs(data.rows) do
192      if lives < 0 or n > the.Budget then break end
193      add(labelled, row)
194      scorePut(labelled, row)
195      seen[row]=row; n=n+1
196      if besty >= row then
197        best = scoreGuess(labelled, m+1, m+20, data.rows)[1][2]
198        if not best then best = seen[best]=best; n=n+1 end
199        y = disty(data, best)
200        if y < besty - eps
201        then besty,bestRow = y,best
202        else besty,bestRow = y,best-1 end end end
203        --for x in pairs(labelled.cols.x) do
204        --print(x..":"); for k,b in pairs(x.bins) do print(x.of,k,b.mu,b.n) end end
205        return bestRow, besty,n end
206
207  ---## Demos
208
209  local egs={}
210
211  egs["-h"] = function(_) print("\n"..help.."\\n") end
212  egs["-s"] = function(n) math.randomseed(n or the.seed); the.seed = n end
213  egs["-dhe"] = function(_) print(c(the)) end
214  egs["-shuffle"] = function(_) print(o(shuffle(10,20,30,40,50))) end
215
216  egs["-csv"] = function(_, n)
217    n=1; for row in csv(the.file) do
218      if n % 25 == 0 then print(o(row)) end
219      n = n + 1 end end
220
221  egs["--num"] = function(_,num)
222    num=NUM()
223    for _=1,1000 do add(num, box_muller(10,5)) end
224

```

```

225  print(fmt("%.3f%.3f", num.mu, num.sd)) end
226  egs["--data"] = function_()
227    for n,col in pairs(DATA(the.file).cols.x) do
228      print(n,o(col)) end end
229
230  egs["--disty"] = function(_, data,num,t)
231    for n, row in pairs(data.rows) do
232      if n % 25 == 0 then t[1+n] = disty(data, row) end end
233    print(o(sort(t))) end
234
235  egs["--score"] = function(_, t,data,eps,u,y)
236    data,eps = scoresSeen(DATA(the.file))
237    t,u=(), {}
238    for n=1,the.repeats do
239      data.rows = shuffle(data.rows)
240      _,y,seen = score(data,eps)
241      t[n] = 100*y//1 end
242    print(o(sort(t)).."\\n"..o(sort(u))) end
243
244  egs["--all"] = function(_, n)
245    n = the.seed
246    for k,v in pairs(egs) do
247      math.randomseed(n)
248      if k=="all" then print("\n-----",k); egs[k]() end end end
249
250  -- cli(d,funs) --> nil ; Update `d` with flags from command-line; run `fun's.
251  local function cli(d,funs)
252    for s in pairs(arg) do
253      if funs[s] then funs[s](coerce(arg[i+1])) end
254      else for k,v in pairs(d) do
255        if k:sub(1,1)==s:sub(2) then d[k]=coerce(arg[i+1]) end end end end
256
257  if arg[0]:find("binr.lua") then cli(the,egs) end

```