

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255

```

-- sam.lua : reasoning via minimal sampling across the data  
-- (c)2022 Tim Menzies <tim@ieee.org> BSD 2 clause license

```

local l=require"lhb"
local any,cat,cli,coerce,copy,csv = 1.any,1.cat,1.cli,1.coerce,1.copy,1.csv
local lines,many,obj,per,push = 1.lines,1.many,1.obj,1.per,1.push
local rogues,words = 1.rogues,1.words

local rand = math.random
local Cols,Data,Row,Num,Sym = obj"Cols",obj"Data",obj"Row",obj"Num",obj"Sym"

local the={example="k", ratios=256, bins=8, seed=10019, some=512}

-- Num -----
function Num:new(at,txt)
  txt = txt or ""
  return {n=0,at=at or 0,txt=txt,cache=nil,has={},
    hi=-math.huge,lo=-math.huge,w=txt:find"$" and -1 or 1} end

function Num:add(x)
  if x ~= "?" then
    local pos
    self.n = self.n + 1
    self.lo = math.min(x, self.lo)
    self.hi = math.max(x, self.hi)
    if #self.has < the.ratios then pos = 1 + (#self.has)
    elseif rand() < the.ratios/self.n then pos = rand(#self.has) end
    if pos then self.cache=nil
      self.has[pos]=x end end end

function Num:discretize(x)
  for _,n in pairs(self.cache) do if x <= n then return n end end
  return self.cache[#self.cache] end

function Num:dist(x,y)
  if x=="?" and y=="?" then return 1 end
  if x=="?" then y=self:norm(y); x=y<.5 and 1 or 0
  elseif y=="?" then x=self:norm(x); y=x<.5 and 1 or 0
  else x,y = self:norm(x), self:norm(y) end
  return math.abs(x-y) end

local function _breaks(a)
  local b = #a//self.bins
  local t,n = {},b
  while n <= #a-b do if a[n]~=a[n+1] then push(t,a[n]); n=n+b else n=n+1 end end
  return t end

function Num:holds(a,n,jump)
  if not self.cache then
    table.sort(self.has)
    self.cache = _breaks(self.has) end
  return self.has, self.cache end

function Num:mid() return per(self:holds(),.5) end

function Num:norm(num)
  return self.hi - self.lo < 1E-9 and 0 or (num-self.lo)/(self.hi-self.lo) end

function Num:div(a)
  a=self:holds()
  return (per(a,.9) - per(a,.1))/2.58 end

-- Sym -----
function Sym:new(at,txt)
  return {n=0,at=at or 0,txt=txt or "",ready=false,has={}} end

function Sym:add(x)
  if x ~= "?" then
    self.n = self.n + 1
    self.has[x] = 1+(self.has[x] or 0) end end

function Sym:discretize(x) return x end

function Sym:dist(x,y)
  return (x=="?" or y=="?") and 1 or x==y and 0 or 1 end

function Sym:mid(mode,most)
  for k,n in pairs(self.has) do if not mode or n>most then mode,most=k,n end end
  return mode end

function Sym:div(e)
  local function p(n) return x*math.log(x,2) end
  e=0; for _,v in pairs(self.has) do if v>0 then e=e-p(v/1.n) end; return e end end

-- Row -----
function Data:far(XXX) end

function Data:half(rows,above,all)
  local all = all or self.rows
  local some = many(all, the.some)
  local left = above or far(any(some), some) end
  -- (defmethod half ((i rows) (optional all above)
  -- "Split rows in two by their distance to two remove points."
  -- (let* ((all (if t .has))
  -- (some (many all (! my some)))
  -- (left (or above (far (any some) some)))
  -- (right (far left some))
  -- (c (dista left right))
  -- (n 0) lefts rights)
  -- (labels ((project (row)
  -- (let ((a (dista row left))
  -- (b (dista row right)))
  -- (cons (/ (+ (* a a) (* c c) (- (* b b))) (* 2 c)) row)))
  -- (dolist (one (sort (mapcar #'project all) #'car<))
  -- (if (<= (infix n) (/ (length all) 2))
  -- (push (cdr one) lefts)
  -- (push (cdr one) rights)))
  -- (values left right lefts rights c)))

```

```

120 return {the=the,Cols=Cols, Data=Data, Num=Num, Sym=Sym}
121
122
123
124
125 -- lib.lua : some of my favorite lua tricks.
126 -- (c)2022 Tim Menzies <tim@ieee.org> BSD 2 clause license
127 local l={}
128
129 -- Cache names -----
130 local b4={}; for k,_ in pairs(_ENV) do b4[k]=k end
131 function l.rogues()
132   for k,v in pairs(_ENV) do if not b4[k] then print("?",k,type(v)) end end end
133
134 -- Maths -----
135 function l.rnd(num, places)
136   local mult = 10^(places or 3)
137   return math.floor(num * mult + 0.5) / mult end
138
139 -- Lists -----
140 function l.any(t) return t[math.random(#t)] end
141
142 function l.copy(t)
143   if type(t) ~= "table" then return t end
144   local u={}; for k,v in pairs(t) do u[k] = l.copy(v) end
145   return setmetatable(u,getmetatable(t)) end
146
147 function l.many(t,n, u) u={}; for i=1,n do u[1+#u]=l.any(t) end; return u end
148
149 function l.per(t,p)
150   p=p or .5
151   p=math.floor((p*#t)+.5); return t[math.max(1,math.min(#t,p))] end
152
153 function l.push(t,x) t[1+#t]=x; return x end
154
155 -- Print table -----
156 function l.chat(t) print(l.cat(t)); return t end
157
158 function l.cat(t)
159   if type(t) ~= "table" then return tostring(t) end
160   local function show(k,v)
161     if not tostring(k):find"%[A-Z]" then
162       v=l.cat(v)
163       return #t==0 and string.format("%s%s",k,v) or tostring(v) end end
164   local u={}; for k,v in pairs(t) do u[1+#u] = show(k,v) end
165   table.sort(u)
166   return (t._is or "").."["..table.concat(u,"").."]" end
167
168 -- Update slots in `t` from command line -----
169 function l.cli(t)
170   for slot,v in pairs(t) do
171     v = tostring(v)
172     for n,x in ipairs(arg) do
173       if x=="-." (slot:sub(1,1)) or x=="-.." slot then
174         v = v=="false" and "true" or v=="true" and "false" or arg[n+1] end end
175     t[slot] = l.coerce(v) end
176   return t end
177
178 -- Define classes -----
179 function l.obj(name)
180   local function new(k,...)
181     local self = setmetatable({},k)
182     return setmetatable(k.new(self,...) or self,k) end
183   local t={_is = name, __tostring = l.cat}
184   t._index = t
185   return setmetatable(t,{__call=new}) end
186
187 -- Coerce -----
188 function l.coerce(str)
189   local function coerced(str)
190     if str=="true" then return true end
191     if str=="false" then return false end
192     return str end
193   return tonumber(str) or coerced(str:match"^%s*(-)%s*$") end
194
195 -- Coerce lines from csv file (filtering result through `fun`).
196 function l.lvc(filename, fun)
197   l.lines(filename, function(t) fun(1.words(t,"",1.coerce)) end) end
198
199 -- Call `fun` on all lines from `filename`.
200 function l.lines(filename, fun)
201   local src = io.input(filename)
202   while true do
203     local str = io.read()
204     if not str then return io.close(src) else fun(str) end end end
205
206 -- Split `str` on `sep`, filtering parts through `fun`.
207 function l.words(str,sep,fun,t)
208   fun = fun or function(z) return z end
209   sep = 1; string.format("(%s)",sep)
210   t={};for x in str:gmatch(sep) do t[1+#t]=fun(x) end;return t end
211
212 return l
213
214
215
216
217 -- eg.lua : demo code for sam.lua
218 -- (c)2022 Tim Menzies <tim@ieee.org> BSD 2 clause license
219 local l=require"lhb"
220 local _=require"sam"
221 local cat,chat,cli,copy,per = 1.cat,1.chat,1.cli,1.copy,1.per
222 local rogues = 1.rnd,1.rogues
223 local Num = _._Num
224 local the,eg,fails = _._the,{},0
225
226 local function run(k, b4,out)
227   math.randomseed(the.seed)
228   b4=copy(the); out=eg[k].fun(); the=copy(b4); return out==true end
229
230 local function eggs(t)
231   t={}; for k,v in pairs(eg) do t[1+#t]=k end; table.sort(t); return t end
232
233 eg.the = {doc="show config", fun=function()
234   chat(the); return true end}
235
236 eg.ls = {doc="list examples", fun=function()
237   print("\nExamples (lua eg.lua -f X)\nX=")
238   for _,k in pairs(egs()) do print(string.format("%7s: %s",k,eg[k].doc)) end

```

```

239 return true end}
240
241 eg.all = {doc="run all examples", fun=function()
242   for _,k in pairs(egs()) do
243     if k ~= "all" then
244       if not run(k) then fails = fails + 1; print("FAIL!",k) end end end end}
245
246 eg.num = {doc="numbers", fun=function(n)
247   n=Num()
248   the.keep = 64
249   for i=1,100 do n:add(i) end
250   return 52==n:mid(2) and 32.56==rnd(n:div(),2) end}
251
252 the = cli(the)
253 if eg[the.example] then eg[the.example].fun() end
254 rogues()
255 os.exit(fails)

```