rules.lua

Contents

```
  -file   ../../data/auto93.csv
  -tiny  .5
  -dull  .35
  -rest  3
  -seed  10019
  -p     2]]
```

```
local function rand(lo,hi)
  the.seed = (16807 * the.seed) % 2147483647
  return (lo or 0) + ((hi or 1) - (lo or 0)) * the.seed / 2147483647 end
local function randi(lo,hi) return math.floor(0.5 + rand(lo,hi)) end
```

```
local function push(t,x)    table.insert(t,x); return x end
local function sort(t,f)    table.sort(t,f); return t end
local function any(t)       return t[randi(1,#t)] end
local function many(t,n, u) u={};for j=1,n do push(u,any(t)) end; return u end
local function map(t,f,  u)
  u={}; for _,v in pairs(t) do push(u, f(v)) end; return u end
```

```
local function sorter(n,     ta, tb)
  return function(a,b,     ta,tb)
    a,b = a[n],b[n]
    ta,tb = type(a),type(b)
    if ta=="number" and tb=="number" then return a < b end
    return ta..tostring(a) < tb..tostring(b) end end
```

```
local fmt=string.format
local function slots(t,  u)
  u={};for k,_ in pairs(t) do push(u,k)end; return u end
local function o(t,  u)
  u={}; for _,k in pairs(sort(slots(t))) do push(u,fmt(":%s %s",k,t[k])) end
  return (t._is or "").."{"..table.concat(u," ").."}" end
```

```
local function atom(x)
  if x=="true" then return true elseif x=="false" then return false end
  return tonumber(x) or x end
```

```
local function csv(file)
  file = io.input(file)
  return function(    t)
    x=io.read();
    if x then
      t={}; for y in x:gsub("%s+",""):gmatch"([^,]+)" do push(t,atom(y)) end
      return #t>0 and t
    else io.close(file) end end end
```

```
local function settings(help,       t)
  t = {}
  help:gsub("\n  [-]([^%s]+)[^\n]*%s([^%s]+)", function(flag, x)
    for n,txt in ipairs(arg) do
      if   txt:sub(1,1)=="-" and flag:match("^"..txt:sub(2)..".*")
      then x = x=="false" and"true" or x=="true" and"false" or arg[n+1] end end
    t[flag] = atom(x) end)
  return t end
```

```
local function new(mt,t)  return setmetatable(t,mt) end
local function klass(s, t)
  t = {_is=s, __tostring=o}
  t.__index = t
  return setmetatable(t,{__call=function(_,...) return t.new(...) end}) end
```

```lua
function per(t,p,m,n,f)
  f = function(x) return x end
  m = m or 1
  n = n or #t
  return f(t[(m+(n - m)*p + .5)//1]) end
```

```lua
local BAG=klass""
function BAG.new(t) return new(BAG,t or {}) end
```

```lua
local SYM=klass"SYM"
function SYM.new(at,s) return new(SYM,{at=at,s=s}) end
```

```lua
function SYM.add(i,x)  return x end
```

```lua
local NUM=klass"NUM"
function NUM.new(at,s, big)
  big = math.huge
  return new(NUM,{lo=big,hi=-big,at=at,s=s,w=(s or ""):find"-" and -1 or 1}) end
```

```lua
function NUM.add(i,x)  i.lo = math.min(x,i.lo); i.hi = math.max(x,i.hi) end
```

```lua
function NUM.norm(i,x) return i.hi-i.lo < 1E-9 and 0 or (x-i.lo)/(i.hi-i.lo) end
```

```lua
local COLS=klass"COL"
function COLS.new(t,     i,where,now)
  i = new(COLS,{all=BAG(), x=BAG(), y=BAG()})
  for at,s in pairs(t) do
    now = push(i.all, (s:find"^[A-Z]" and NUM or SYM)(at,s))
    if not s:find":" then
      push((s:find"-" or s:find"+") and i.y or i.x, now) end end
  return i end
```

```lua
function COLS.add(i,t,      add)
  t = t.has and t.has or t
  function add(col) x=t[col.at]; if x~="?" then col:add(x) end; return x end
  return map(i.all, add) end
```

```lua
function COLS.better(i,row1,row2)
  local s1,s2,e,n,a,b = 0,0,10,#i.y
  for _,col in pairs(i.y) do
    a  = col:norm(row1.has[col.at])
    b  = col:norm(row2.has[col.at])
    s1 = s1 - e^(col.w * (a-b)/n)
    s2 = s2 - e^(col.w * (b-a)/n) end
  return s1/n < s2/n end
```

```lua
local EG=klass"EG"
local id=0
function EG.new(t) id=id+1; return new(EG, {has=t, id=id}) end
```

```lua
local EGS=klass"EGS"
function EGS.new() return new(EGS, {rows={}, cols=nil}) end
```

```lua
function EGS.file(i,file)
  for row in csv(file) do
    if i.cols then push(i.rows, EG(i.cols:add(row))) else i.cols=COLS(row) end end
  return i end
```

```
function EGS.bestRest(i)
  local t0,t,tmp = {},{},{}
  i.rows = sort(i.rows, function(a,b) return i.cols:better(a,b) end)
  for j,x in pairs(i.rows) do push(j <= (#i.rows)^the.best and t0 or tmp, x) end
  t = many(tmp, the.rest*#t0)
  return t0,t end
```

```
local RANGES=klass"RANGES"
function RANGES.new()
  return new(RANGES,{{col=col, lo=lo, hi=hi or lo}, ys=SYM(),all={}}) end
```

```
function div(t,at,lo,hi,tiny,dull,out)
  function x(i)       return t[i].has[at] end
  function per(i,j,p) return x(i+(j-i)*p//1) end
  function sd(i,j)    return (per(i,j,.9) - per(i,j,.1))/2.56 end
  tiny = tiny or (#t)^the.tiny
  dull = dull or sd(1,#t)*the.dull
  out  = out or {}
  best = sd(lo,hi)
  for j=lo,hi do
    k=j+1
    if j-lo >= min and
       hi-j >= min and
       x(j) ~= x(k) and
       x(j)  - x(lo) >= dull and
       x(hi) - x(k)  >= dull
    then
      xpect = ((j-lo)*sd(lo,j) + (hi-k)*sd(k,hi))/(hi-lo+1)
      if xpect*1.01 < best then
        best,cut = xpect,k end end end end
```

```
the = settings(help)
local i = EGS.new()
local t0,t = i:file(the.file):bestRest()
print(#t0, #t)
```

–for *,col in pairs(i.cols.x) do for* ,x in ipairs(sort(t0, sorter(col.at))) do print(x.has[col.at]) end end

```
local t={{1},{2},{3},{"?"},{10},{80},{8},{"?"},{"?"}, {2},{2},{11},{12}}
local t=sort(t, sorter(1))
```

“‘lua for k,v in ipairs(t) do print(k,v[1]) end for k,v in pairs(_ENV) do if not b4[k] then print(“?”,k,type(v)) end end