```lua
-- ## class NUM: summarize numbers
local all = require"all"
local obj,push,the = all.obj, all.push, all.the
local SOME = require"some"

--> NUM(at:?int, txt:?str) :NUM -> Summarize a stream of numbers.
local NUM = obj("NUM", function(i,at,txt)
  i.at, i.txt, i.n, i.kept =  at or 0, txt or "", 0, SOME(the.Some)
  i.w =  i.txt:find"-$"  end)

--> add(i:NUM: x:num, n:?int=1) -> `n` times,update `i`'s SOME object.
function NUM.add(i,x,n)
  if x ~="?" then
    for _ = 1,(n or 1) do i.n=i.n+1; i.kept:add(x) end end end

--> clone(i:(SYM|NUM)) :(SYM|NUM) -> Return a class of the same structure.
function NUM.clone(i) return NUM(i.at, i.txt) end

--> div(i:NUM):tab -> Return `div`ersity of a column
-- (its tendency _not_ to be a its central tendency). To understand this code
-- recall &pm;1 to &pm;2 sds covers 66 to 95% of the Gaussian prob. In between,
-- at &pm;1.28, we cover 90%. So (p90-p10)/(2*1.28) returns one sd.
function NUM.div(i)
  local a=i.kept:has(); return (per(a,.9) - per(a,.1))/2.56 end

--> like(i:NUM, x:any) -> Return the likelihood that `x` belongs to `i`.
function NUM.like(i,x,...)
  local sd,mu=i:div(), i:mid()
  if sd==0 then return x==mu and 1 or 1/big end
  return math.exp(-.5*((x - mu)/sd)^2) / (sd*((2*math.pi)^0.5)) end

--> mid(i:NUM)):tab -> Return a columns' `mid`ddle
function NUM.mid(i)
  local a=i.kept:has(); return per(a,.5) end

--> norm(i:NUM, x:num):num -> Normalize `x` 0..1 for lo..hi,
function NUM.norm(i,x)
  local a=i.kept:has(); return (a[#a]-a[1])<1E-9 or (x-a[1])/(a[#a]-a[1]) end

return NUM
```