```lua
local the,_ = require"the", require"lib"
local has2,has3,inc,inc2,inc3   = _.has2,_.has3,_.inc,_.inc2,_.inc3
local push,sort,collect,items   = _.push,_.sort,_.collect,_.items
local map,down1,rnds,oo,class,OBJ = _.map,_.down1,_.rnds,_.oo,_.class,_.OBJ

local NB=class("NB",OBJ)
function NB:new(data, this)
  self.n, self.nh, self.wait         = 0,0, the.wait
  self.e, self.h, self.log,self.cols = {},{},{},nil
  for row in items(data) do
    if   not self.cols
    then self.cols= collect(row,function(j,s) return {name=s,indep=j~=#row} end)
    else self:test(row); self:train(row) end end  end

function NB:test(row)
  if self.n > the.wait then
    push(self.log,{want=row[#row], got=self:classify(row)}) end end

function NB:train(row)
  local more, kl = false, row[#row]
  for col,x in pairs(row) do
    if x ~="?" then
      more = true
      inc3(self.e, col, x, kl)  end end
  if more then
    self.n = self.n + 1
    if not self.h[kl] then self.nh = self.nh + 1 end
    inc(self.h, kl) end end

function NB:classify(t,use)
  local hi,out = -math.huge
  for h,val in pairs(self.h) do
    local prior = ((self.h[h] or 0) + the.K)/(self.n + the.K*self.nh)
    local l = math.log(prior)
    for col,x in pairs(t) do
      if x ~= "?" and self.cols[col].indep then
        l = l + math.log((has3(self.e,col,x,h) + the.M*prior) /
                         ((self.h[h] or 0) + the.M)) end end
    if l>hi then hi,out=l,h end end
  return out end

function NB:score()
  local a,n = 0,#self.log
  for key,x in pairs(self.log) do if x.want==x.got then a=a+1/n end end
  return acc,self.log end

return NB
```