```
TOP      = 10
MaxLIVES = 5
LIVES    = MaxLIVES
SAMPLES  = 20

which2()
        counts()
        for class1 in classes do
                which = round0(class1)
                which = rounds(class1, which)
                print "rules for class1 are " which

counts()
        for row in rows do
                NUMBER[class of row]++
                NUMBER["all"]++
                for col in row
                        FREQUENCY[class of row, col, value of col]++
                        FREQUENCY["all",        col, value of col]++

round0(class1) {
        for (class2,col,value) in FREQUENCY do
                if (class1 == class2)
                        best0 = FREQUENCY[class2,col,value]
                        bests = NUMBER[class2]
                        rest0 = FREQUENCY["all", col,value] - best0
                        rests = NUMBER["all"] - bests
                        best  = best0/bests
                        rest  = rest0/rests
                        if (best > rest)
                                push (best^2/(best + rest), col,value) onto SCORE
        return TOP items of SCORE

rounds(class1,which0)
        if   no improvement from last round
                then LIVES--
                else LIVES=MaxLIVES
        if LIVES < 1
                then return which
        sample = explode(normalize(which0))
        which1 = copy(which0)
        SAMPLES times repeat             ; e.g 20 times repeat
                one = any item from sample
                two = any item from sample
                new = combine one and two
                push (abcd(new,class1), new) onto which1
        which2 = TOP items of which1
        return rounds(class1,which2)

normalize(list)
        for key in list do total =+ array[key]
        for key in list do list[key] = round (100*list[key]/total)
        return list

explode(list) { ; e.g. a[apples]=90, a[bananas]=10
        for key in list
                for i = 1 to list[key]
                        push key onto out
        return out   ; e.g. contains 90 apples and 10 bananas

abcd(class1, rule)
        for row in   rows
                triggered = rule matches row
                if (class of row == class1) {
                        {if triggered then d++ else b++ }
                } else {
                        {if triggered then c++ else a++ }
                }
        probability of detection = pd = d/(d+b)
        probability of false alarm = pf =c/(c+a)
        precision = prec =d/(d+c)
        accuracy = acc = (a+d)/(a+b+c+d)
        support = s = (c+d)/(a+b+c+d)
        fmeasure = 2*pd*prec/(pd+prec)
        return support * fmeasure
```