

Oct 31, 14 11:53 genic Page 1/3

```

from __future__ import division, print_function
import sys, random, re
sys.dont_write_bytecode = True

5 #####
def genic0(**d): return o(
    k=16,
    era=5000,
    num='$',
10    klass='=',
    seed=1).update(**d)

def rows0(**d): return o(
    skip="?",
    sep = ',',
15    bad = r'(["\\t\\r\\n]|#.*)',
    ).update(**d)

#####
20 rand= random.random
seed= random.seed

def say(c):
    sys.stdout.write(str(c))

25 def g(lst,n=3):
    for col,val in enumerate(lst):
        if isinstance(val,float):
            val = round(val,n)
30    lst[col] = val
    return lst

class o:
    "Define a bag of names slots with no methods."
35 def __init__(i,**d): i.update(**d)
    def update(i,**d):
        i.__dict__.update(**d); return i
    def __repr__(i) :
        d = i.__dict__
40    show = ['%s%s' % (k,d[k])
            for k in sorted(d.keys())
            if k[0] is not '_']
    return '{'+'.join(show)+'}'

45 #####
def rows(file,w=None):
    " "Leaps over any columns marked 'skip'.
    Turn strings to numbers or strings.
    Kill comments. Join lines that end in 'sep'." ""
50 w = w v rows0()
    def atom(x):
        try : return int(x)
        except ValueError:
            try : return float(x)
55    except ValueError : return x
    def lines():
        n,kept = 0,""
        for line in open(file):
            now = re.sub(w.bad,"",line)
60            kept += now
            if kept:
                if not now[-1] == w.sep:
                    yield n, map(atom, kept.split(w.sep))
                    n += 1
65            kept = ""
    todo = None
    for n,line in lines():
        todo = todo v [col for col,name
                        in enumerate(line)
70                        if not w.skip in name]
    yield n, [ line[col] for col in todo ]

```

Oct 31, 14 11:53 genic Page 2/3

```

75 def header(w,row):
    def numOrSym(val):
        return w.num if w.opt.num in val else w.sym
    def indepOrDep(val):
        return w.dep if w.opt.klass in val else w.indep
80    for col,val in enumerate(row):
        numOrSym(val).append(col)
        indepOrDep(val).append(col)
        w.name[col] = val
        w.index[val] = col

85 def data(w,row):
    for col in w.num:
        val = row[col]
        w.min[col] = min(val, w.min.get(col,val))
90    w.max[col] = max(val, w.max.get(col,val))

def indep(w,cols):
    for col in cols:
        if col in w.indep: yield col

95 #####
def nearest(w,row):
    def norm(val,col):
        lo, hi = w.min[col], w.max[col]
100    return (val - lo) / (hi - lo + 0.00001)
    def dist(centroid):
        n,d = 0,0
        for col in indep(w, w.num):
            x1,x2 = row[col], centroid[col]
105            n1,n2 = norm(x1,col), norm(x2,col)
            d += (n1 - n2)**2
            n += 1
        for col in indep(w, w.sym):
            x1,x2 = row[col], centroid[col]
110            d += (0 if x1 == x2 else 1)
            n += 1
        return d**0.5 / n**0.5
        lo, out = 10**32, None
        for n,(_,centroid) in enumerate(w.centroids):
115            d = dist(centroid)
            if d < lo:
                lo,out = d,n
        return out

120 def move(w,new,n):
    u0,old = w.centroids[n]
    u1 = 1
    out = [None]*len(old)
    for col in w.sym:
125        x0,x1 = old[col], new[col]
        out[col] = x1 if rand() < 1/(u0+u1) else x0
    for col in w.num:
        x0,x1 = old[col], new[col]
        out[col] = (u0*x0 + u1*x1)/(u0+u1)
130    w.centroids[n] = (u0 + u1, out)

def less(w) :
    b4 = len(w.centroids)
    rare = w.opt.era/w.opt.k
135    w.centroids = [(1,row) for u,row in
                    w.centroids if u < rare]
    now=len(w.centroids)
    print("-",b4 - now)

140
145

```

Oct 31, 14 11:53 genic Page 3/3

```

def genic(src='data/diabetes.csv',opt=None):
150    w = o(num=[], sym=[], dep=[], indep=[],
        centroids=[],
        min={}, max={}, name={}, index={},
        opt=None v genic0())
    for n,row in rows(src):
155        if n == 0:
            header(w,row)
        else:
            data(w,row)
            if len(w.centroids) < w.opt.k:
160                say("+")
                w.centroids += [(1,row)]
                continue
            move(w,row,nearest(w,row))
            if 0 == (n % w.opt.era):
165                say(n)
                less(w)
            return sorted(w.centroids,reverse=True)

if __name__ == '__main__':
170    src='data/diabetes2.csv'
    if len(sys.argv) == 2:
        src= sys.argv[1]
    opt=genic0()
    clusters = genic(src)
175    seed(opt.seed)
    print("")
    for m,(n,centroid) in enumerate(clusters):
        rare = opt.era/opt.k
        if n > rare:
180            print(m+1,n,"-",g(centroid,2))

```