

Sep 14, 13 17:21

573:proj1d:Naveen Kumar Lekkalapudi

Page 1/1

```

#Info for table
#
csvindex = -1 #initialized to -1 as lists start at zero
colname = {k: [] for k in range(1)} #stores dict of names of columns
5 data = {k: [] for k in range(1)} #stores dict of list of lists containing each row
test = [] #stores test data
#
#metadata
#
10 order = {k:dict.fromkeys(colname) for k in range(1)} #stores colnames and index of column in csv
klass = {k: [] for k in range(1)} #dict of list of klass columns
more = {k: [] for k in range(1)} #dict of list of more columns
less = {k: [] for k in range(1)} #dict of list of less columns
num = {k: [] for k in range(1)} #dict of list of num columns
15 term = {k: [] for k in range(1)} #dict of list of term columns
dep = {k: [] for k in range(1)} #dict of list of dependent columns
indep = {k: [] for k in range(1)} #dict of list of independent columns
nump = {k: [] for k in range(1)} #dict of list containing nump column names
wordp = {k: [] for k in range(1)} #dict of list containing wordp column names
20 #
#for nump values
#
hi = {k:dict.fromkeys(nump) for k in range(1)} #dictionary containing highest values of nump columns
lo = {k:dict.fromkeys(nump) for k in range(1)} #dictionary containing lowest values of nump columns
25 mu = {k:dict.fromkeys(nump) for k in range(1)} #dictionary containing mean values of nump columns
m2 = {k:dict.fromkeys(nump) for k in range(1)} #dictionary containing m2 values of nump columns
sd = {k:dict.fromkeys(nump) for k in range(1)} #dictionary containing std dev of nump columns
#
#for wordp values
30 #
mode = {k:dict.fromkeys(wordp) for k in range(1)} #dictionary containing mode of wordp columns
most = {k:dict.fromkeys(wordp) for k in range(1)} #dictionary containing most occurred item of wordp columns
count = {k:dict(dict.fromkeys(wordp)) for k in range(1)} #dictionary of dictionaries of count of each item in each wordp column
#
35 #for all
#
n = {k:dict.fromkeys(colname) for k in range(1)} #stores number of elements in each column
isnum = True
#
40 #for the zeror
#
hypotheses = {}
#
#for naive bayes
45 l = {} #dictionary of likelihood
#

```

Sep 14, 13 18:33

573:proj1d:Naveen Kumar Lekkalapudi

Page 1/1

```

import re
from globfile import *
from random import *
from math import *
5  PI = 3.1415926535
   EE = 2.7182818284

def line(csvfile): #returns formatted line from the csvfile
    l = csvfile.readline()
10    endcommare = re.compile('.*,')
    if l != '':
        l = l.split('#')[0]
        l = l.replace('\t','')
        l = l.replace('\n','')
15        l = l.replace(' ','')
        endcomma = endcommare.match(l)
        if endcomma:
            return l+line(csvfile)
        else:
            return l
20    else:
        return -1

def rowprint(row): #returns neat rows
25    columns = [ "%15s" % cell for cell in row]
    columns.append("%4s" % '#')
    return ' '.join(columns)

def expected(row,z): #returns expected outcome
30    out = [c for c in colname[z]]
    for c in row:
        if c in wordp[z]:
            out[colname[z].index(c)] = str(mode[z][c])
        else:
35            out[colname[z].index(c)] = str('%0.2f' % round(mu[z][c],2))
    return out

def indexes(lst):
    out = []*len(lst)
40    for i in lst:
        out[i] = i
    return out

def newdlist(name, key):
45    name[key] = []

def newddict(name,key):
    name[key] = {}

50 def newddictdict(name,key,c):
    name[key][c] = {}

def indexes(data,z):
    return data[z]
55

def shuffled(rows):
    shuffle(rows)

def norm(x,m,s):
60    s += 1/10**23
    return 1/sqrt(2*PI*s**2)*EE**(-1*(x-m)**2/(2*s**2))

```

Sep 14, 13 18:34

573:proj1d:Naveen Kumar Lekkalapudi

Page 1/1

```

#!/usr/bin/env python
from lib import *
from reader import *
from xval import *
5 from math import *

def nb(test,data,hypotheses,z,k,m):
    total = 0.0
    acc = 0.0
10    for h in hypotheses:
        total += len(data[h])
        where = klassAt(z)
        for t in test:
            want = t[where]
            got = likelihood(t,data,total,hypotheses,l,z,k,m)
15            if want == got:
                acc+=1.0
            print '%0.2f' % round(100*acc/len(test),2),

20 def likelihood(t,data,total,hypotheses,l,z,k,m):
    like = -0.1*10**23
    best = ''
    total += k*len(hypotheses)
    for h in hypotheses:
25        nh = len(data[h])*0.1
        prior = (nh+k) / total
        tmp = log(prior)
        for c in term[h]:
            try:
30                ind = colname[h].index(c)
                x = t[ind]
                if x == '?':
                    continue
                y = count[h][c][x]
35                tmp += log((y + m*prior)/(nh+m))
            except KeyError:
                continue
        for c in num[h]:
            ind = colname[h].index(c)
            x = t[ind]
            if x == '?':
40                continue
            y = norm(x, mu[h][c], sd[h][c])
            tmp += log(y)
45        l[h] = tmp
        if tmp >= like:
            like = tmp
            best = h
    return best

```

Sep 15, 13 23:44      **573:proj1d:Naveen Kumar Lekkalapudi**      Page 1/1

```

diabetes
zeror
63.40 62.09 64.71 61.44 73.20 54.25 66.67 71.90 62.09 70.59 64.05 69.28 64.71 64
.71 62.75 65.36 67.32 64.05 60.13 68.63 60.78 66.67 61.44 65.36 71.90
diabetes
5 nb
75.16 75.82 77.12 75.82 76.47 75.82 74.51 76.47 77.12 77.12 76.47 69.93 81.05 73
.20 81.05 80.39 71.90 75.16 78.43 75.16 78.43 69.93 76.47 81.70 73.86
soybean
zeror
11.76 12.50 11.03 14.71 12.50 14.71 11.76 14.71 11.03 15.44 16.18 11.76 7.35 17.
65 13.97 17.65 9.56 10.29 11.76 13.24 11.03 11.03 9.56 16.18 19.12
10 soybean
nb
61.03 69.85 74.26 80.88 82.35 79.41 88.24 86.76 85.29 86.03 83.82 86.03 91.18 84
.56 91.91 86.03 91.18 88.97 83.82 90.44 86.03 87.50 88.97 88.24 88.97
iris
zeror
15 26.67 33.33 20.00 46.67 26.67 30.00 36.67 26.67 40.00 33.33 26.67 33.33 33.33 30
.00 30.00 23.33 33.33 13.33 36.67 33.33 20.00 36.67 26.67 43.33 20.00
iris
nb
96.67 90.00 100.00 96.67 93.33 93.33 96.67 86.67 100.00 100.00 93.33 93.33 90.00
100.00 100.00 96.67 96.67 93.33 100.00 93.33 96.67 96.67 96.67 96.67 93.33

```

Sep 15, 13 23:42

**573:proj1d:Naveen Kumar Lekkalapudi**

Page 1/1

```
from reader import *
from table import *
from sys import argv
from xval import *

5 csvfile = open('../data/iris.csv','r')
  readCsv(csvfile,argv[1]) #takes predicted value as arguement
  xvals(data,5,5,'zeror',argv[1],1,2)

10 #tableprint(argv[1])
```

Sep 13, 13 19:26

573:proj1d:Naveen Kumar Lekkalapudi

Page 1/2

```

import re
from lib import *

def makeTable(lst,z):
    newdlist(klass,z)
    newddict(order,z)
    newdlist(less,z)
    newdlist(num,z)
    newdlist(term,z)
    newdlist(dep,z)
    newdlist(indep,z)
    newdlist(nump,z)
    newdlist(wordp,z)
    newdlist(colname,z)
    newdlist(data,z)
    newddict(count,z)
    newddict(n,z)
    newddict(mode,z)
    newddict(most,z)
    newddict(hi,z)
    newddict(lo,z)
    newddict(mu,z)
    newddict(m2,z)
    newddict(sd,z)
    newdlist(data,z)

    csvindex = -1
    for csvcol in lst:
        isnum=True
        csvindex+=1
        ignore = re.match('\?.*$',csvcol)
        if ignore:
            continue
        else:
            colname[z].append(csvcol)
            order[z][csvcol] = csvindex
            klasschk = re.match('=..*$',csvcol)
            morechk = re.match('+..*$',csvcol)
            lesschk = re.match('-.*$',csvcol)
            numchk = re.match('\$..*$',csvcol)
            if klasschk:
                dep[z].append(csvcol)
                klass[z].append(csvcol)
                isnum = False
            elif morechk:
                dep[z].append(csvcol)
                more[z].append(csvcol)
            elif lesschk:
                dep[z].append(csvcol)
                less[z].append(csvcol)
            elif numchk:
                indep[z].append(csvcol)
                num[z].append(csvcol)
            else:
                indep[z].append(csvcol)
                term[z].append(csvcol)
                isnum = False
            n[z][csvcol] = 0
            if isnum:
                nump[z].append(csvcol)
                hi[z][csvcol] = 0.1 * (-10**13)
                lo[z][csvcol] = 0.1 * (10**13)
                mu[z][csvcol] = 0.0
                m2[z][csvcol] = 0.0
                sd[z][csvcol] = 0.0
            else:
                wordp[z].append(csvcol)
                count[z][csvcol] = {}
                mode[z][csvcol] = 0
                most[z][csvcol] = 0

def addRow(lst,z):
    temp = []

```

Sep 13, 13 19:26

573:proj1d:Naveen Kumar Lekkalapudi

Page 2/2

```

skip = False
for c in klass[z]:
    csvindex = order[z][c]
    item = lst[csvindex]
    if item != z:
        skip = True
    if z == "both":
        skip = False
for c in colname[z]:
    csvindex = order[z][c]
    item = lst[csvindex]
    uncertain = re.match('\?$',str(item))
    if skip:
        return
    if uncertain:
        temp.append(item)
    else:
        n[z][c] += 1
        if c in wordp[z]:
            temp.append(item)
        try:
            new = count[z][c][item] = count[z][c][item] + 1
            if new > most[z][c]:
                most[z][c] = new
                mode[z][c] = item
            except KeyError:
                count[z][c][item] = 1
                if count[z][c][item] > most[z][c]:
                    most[z][c] = 1
                    mode[z][c] = item
        else:
            item = float(item)
            temp.append(item)
            if item > hi[z][c]:
                hi[z][c] = item
            if item < lo[z][c]:
                lo[z][c] = item
            delta = item - mu[z][c]
            mu[z][c] += delta / n[z][c]
            m2[z][c] += delta * (item - mu[z][c])
            if n[z][c] > 1:
                sd[z][c] = (m2[z][c] / (n[z][c] - 1))**0.5
data[z].append(temp)

def readCsv(csvfile,z):
    seen = False
    FS = ','
    while True:
        lst = line(csvfile)
        if lst == -1:
            print 'WARNING: empty or missing file'
            return -1
        lst = lst.split(FS)
        if lst != ['']:
            if seen:
                addRow(lst,z)
            else:
                seen = True
                makeTable(lst,z)

```

Sep 13, 13 19:26

573:proj1d:Naveen Kumar Lekkalapudi

Page 1/1

```

from globfile import *
from lib import *
def tableprint(z): #prints table with the summary
    print rowprint(colname[z]), '%10s' % 'notes'
    print rowprint(expected(colname[z],z)), '%10s' % 'expected'
5     temp = [ c for c in range(len(colname[z]))]
        for c in colname[z]:
            if c in nump[z]:
                temp[colname[z].index(c)] = str('%0.2f' % round(sd[z][c],2))
10            else:
                temp[colname[z].index(c)] = str('%0.2f' % round(float(most[z][c])/fl
oat(n[z][c]),2))
        print rowprint(temp), '%10s' % 'certainty'
        for row in data[z]:
            print rowprint(row)
15
def klassl(data, z):
    for k in klass[z]:
        return data[colname[z].index(k)]

20 def klassAt(z):
    for k in klass[z]:
        return colname[z].index(k)

```

Sep 15, 13 23:43

573:proj1d:Naveen Kumar Lekkalapudi

Page 1/1

```

#!/bin/python
from lib import *
from reader import *
from table import *
5 from zeror import *
from nb import *

def xvals(data,x,b,f,z,k,m):
    rows = indexes(data,z)
10    s = int(len(rows)/b)
    while x>0:
        shuffled(rows)
        for bl in range(0,b):
            xval(bl*s, (bl+1)*s, data, rows, f, z, k, m)
15    x=x-1

def xval(start, stop, data, rows, f, z, k, m):
    rmax = len(rows)
    test = []
20    temp = ""
    newddict(data,z)
    for r in range(0, rmax):
        d = rows[r]
        if r ≥ start ^ r < stop:
            test.append(d)
25        else:
            temp = klass1(d, z)
            try:
                hypotheses[temp] += 1
30                if hypotheses[temp] == 1:
                    makeTable(colname[z],temp)
                    addRow(d,temp)
            except KeyError:
                hypotheses[temp] = 1
35                if hypotheses[temp] == 1:
                    makeTable(colname[z],temp)
                    addRow(d,temp)
    #zeror(test, data, hypotheses, z)
    #xvalTest1(test,data,hypotheses)
40    nb(test,data,hypotheses,z,k,m)

def xvalTest1(test,data,hypotheses):
    print "\n=====
45    for h in hypotheses:
        tableprint(h)

```



Sep 14, 13 18:12    **573:proj1d:Naveen Kumar Lekkalapudi**    Page 1/1

```
from reader import *
from xval import *
from lib import *

5 def zeror(test,data,hypotheses,z):
    hmost = -10**23
    acc = 0
    got = ""
    for h in hypotheses:
10         these = len(data[h])
            if these > hmost:
                hmost = these
                got = h
    #print "#got: ",got
    where = klassAt(z)
15     for t in test:
        want = t[where]
        if want == got:
            acc+=1.0
20     print '%0.2f' % round(100*acc/len(test),2),
```