

Sep 13, 13 18:11

573:proj1c: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 = {}

```

Sep 13, 13 18:11

573:proj1c:Naveen Kumar Lekkalapudi

Page 1/1

```

import re
from globfile import *
from random import *

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

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

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

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

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

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

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

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

    def shuffled(rows):
        shuffle(rows)

```

Sep 13, 13 18:51

**573:proj1c:Naveen Kumar Lekkalapudi**

Page 1/1

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

5 csvfile = open('../data/weather1.csv','r')
  readCsv(csvfile,argv[1]) #takes predicted value as argument
  xvals(data,2,2,'zeror',argv[1])

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

Sep 13, 13 18:29

573:proj1c: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 18:29

573:proj1c: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 18:40

573:proj1c: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'
5     print rowprint(expected(colname[z],z)), '%10s' % 'expected'
    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 13, 13 19:12

573:proj1c:Naveen Kumar Lekkalapudi

Page 1/1

```

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

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

15 def xval(start, stop, data, rows, f, z):
    rmax = len(rows)
    test = []
    hypotheses = {}
    temp = ""
20 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
                if hypotheses[temp] == 1:
                    makeTable(colname[z],temp)
                    addRow(d,temp)
            except KeyError:
                hypotheses[temp] = 1
                if hypotheses[temp] == 1:
                    makeTable(colname[z],temp)
                    addRow(d,temp)
30 zeror(test, data, hypotheses, z)
    xvalTest1(test,data,hypotheses)

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

```

Sep 13, 13 18:40 **573:proj1c: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 100*acc/len(test),"\t"
```