```
    #Info for table
    #
    csvindex = -1 #initialized to -1 as lists start at zero
    colname = [] #stores names of columns
5   data = [] #stores list of lists containing each row
    #
    #metadata
    #
    order = dict.fromkeys(colname) #stores colnames and index of column in csv
10  klass = [] #list of klass columns
    more = [] #list of more columns
    less = [] #list of less columns
    num = [] #list of num columns
    term = [] #list of term columns
15  dep = [] #list of dependent columns
    indep = [] #list of independent columns
    nump = [] #list containing nump column names
    wordp = [] #list containing wordp column names
    #
20  #for nump values
    #
    hi = dict.fromkeys(nump) #dictionary containing highest values of nump columns
    lo = dict.fromkeys(nump) #dictionary containing lowest values of nump columns
    mu = dict.fromkeys(nump) #dictionary containing mean values of nump columns
25  m2 = dict.fromkeys(nump) #dictionary containing m2 values of nump columns
    sd = dict.fromkeys(nump) #dictionary containing std dev of nump columns
    #
    #for wordp values
    #
30  mode = dict.fromkeys(wordp) #dictionary containing mode of wordp columns
    most = dict.fromkeys(wordp) #dictionary containing most occured item of wordp co
    lumns
    count = dict((dict.fromkeys(wordp))) #dictionary of dictionaries of count of eac
    h item in each wordp column
    #
    #for all
35  #
    n = dict.fromkeys(colname) #stores number of elements in each column
    isnum = True
```

```
     import re
     from globfile import *

     def line(csvfile): #returns formatted line from the csvfile
5        l = csvfile.readline()
         endcommare = re.compile('.*,$')
         if l ≠ '':
             l = l.split('#')[0]
             l = l.replace('\t','')
10           l = l.replace('\n','')
             l = l.replace(' ','')
             endcomma = endcommare.match(l)
             if endcomma:
                 return l+line(csvfile)
15           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): #returns expected outcome
         out = [c for c in colname]
         for c in row:
             if c in wordp:
                 out[colname.index(c)] = str(mode[c])
30           else:
                 out[colname.index(c)] = str('%0.2f' % round(mu[c],2))
         return out
```

```
     from reader import *
     from table import *
     from sys import argv
     csvfile = open('../data/weather1.csv','r')
5    readCsv(csvfile,argv[1],argv[2]) """takes predicted column as
                     first arguement and predicted value as second arguement"""
     tableprint(csvfile)
```

```
     import re
     from lib import *

     def makeTable(str,csvindex):
5        for csvcol in str:
             isnum=True
             csvindex+=1
             ignore = re.match('\?.*$',csvcol)
             if ignore:
10               continue
             else:
                 colname.append(csvcol)
                 order[csvcol] = csvindex
                 klasschk = re.match('=.*$',csvcol)
15               morechk = re.match('\+.*$',csvcol)
                 lesschk = re.match('-.*$',csvcol)
                 numchk = re.match('\$.*$',csvcol)
                 if klasschk:
                     dep.append(csvcol)
20                   klass.append(csvcol)
                     isnum = False
                 elif morechk:
                     dep.append(csvcol)
                     more.append(csvcol)
25               elif lesschk:
                     dep.append(csvcol)
                     less.append(csvcol)
                 elif numchk:
                     indep.append(csvcol)
30                   num.append(csvcol)
                 else:
                     indep.append(csvcol)
                     term.append(csvcol)
                     isnum = False
35               n[csvcol] = 0
                 if isnum:
                     nump.append(csvcol)
                     hi[csvcol] = 0.1 * (-10**13)
                     lo[csvcol] = 0.1 * (10**13)
40                   mu[csvcol] = 0.0
                     m2[csvcol] = 0.0
                     sd[csvcol] = 0.0
                 else:
                     wordp.append(csvcol)
45                   count[csvcol] = dict()
                     mode[csvcol] = 0
                     most[csvcol] = 0

     def addRow(str,predclass,pred,colname,data,csvindex):
50       temp = []
         skip = False
         if predclass in klass:
             csvindex = order[predclass]
             item = str[csvindex]
55           if item ≡ pred:
                 skip = False
             elif pred ≡ 'both':
                 skip = False
             else:
60               skip = True
         else:
             print 'WARNING: Class to be predicted is not in klass'
         if skip:
             return
65       for c in colname:
             csvindex = order[c]
             item = str[csvindex]
             uncertain = re.match('\?',item)
             if uncertain:
70               temp.append(item)
             else:
                 n[c] += 1
                 if c in wordp:
```

```
                     temp.append(item)
75                   try:
                         new = count[c][item] = count[c][item] + 1
                         if new > most[c]:
                             most[c] = new
                             mode[c] = item
80                   except KeyError:
                         count[c][item] = 1
                         if count[c][item] > most[c]:
                             most[c] = 1
                             mode[c] = item
85                   else:
                         item = float(item)
                         temp.append(item)
                         if item > hi[c]:
                             hi[c] = item
90                       if item < lo[c]:
                             lo[c] = item
                         delta = item - mu[c]
                         mu[c] += delta / n[c]
                         m2[c] = delta * (item - mu[c])
95                       if n[c] > 1:
                             sd[c] = (m2[c] / (n[c] - 1)**0.5)
             data.append(temp)

     def readCsv(csvfile,predclass,pred):
100      seen = False
         FS = ','
         while True:
             str = line(csvfile)
             if str ≡ -1:
105              print 'WARNING: empty or missing file'
                 return -1
             str = str.split(FS)
             if str ≠ ['']:
                 if seen:
110                  addRow(str,predclass,pred,colname,data,csvindex)
                 else:
                     seen = True
                     makeTable(str,csvindex)
```

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

```
      WARNING: empty or missing file
         outlook      -$humidity        windy         =play     #      notes
          sunny         81.83           FALSE          yes      #   expected
           0.36         21.94            0.57          0.64      # certainity
5         sunny         90.0            FALSE           no       #
          sunny         90.0            TRUE            no       #
        overcast        86.0            FALSE          yes       #
          rainy         96.0            FALSE          yes       #
          rainy         80.0            FALSE          yes       #
10        rainy          ?              TRUE            no       #
        overcast        65.0            TRUE           yes       #
          sunny          ?              FALSE           no       #
          sunny         70.0            FALSE          yes       #
          rainy         80.0            FALSE          yes       #
15        sunny         70.0            TRUE           yes       #
        overcast        90.0            TRUE           yes       #
        overcast        75.0            FALSE          yes       #
          rainy         90.0            TRUE            no       #
      WARNING: empty or missing file
20       outlook      -$humidity        windy         =play     #      notes
        overcast        79.11           FALSE          yes      #   expected
           0.44          6.72            0.67          1.00      # certainity
        overcast        86.0            FALSE          yes       #
          rainy         96.0            FALSE          yes       #
25        rainy         80.0            FALSE          yes       #
        overcast        65.0            TRUE           yes       #
          sunny         70.0            FALSE          yes       #
          rainy         80.0            FALSE          yes       #
          sunny         70.0            TRUE           yes       #
30      overcast        90.0            TRUE           yes       #
        overcast        75.0            FALSE          yes       #
      WARNING: empty or missing file
         outlook      -$humidity        windy         =play     #      notes
          sunny         90.00           TRUE            no      #   expected
35         0.60          0.00            0.60          1.00      # certainity
          sunny         90.0            FALSE           no       #
          sunny         90.0            TRUE            no       #
          rainy          ?              TRUE            no       #
          sunny          ?              FALSE           no       #
40        rainy         90.0            TRUE            no       #
```