Machine Learning Homework 1_Report

2017112167 이수진

1. Write a Python program to count the number of strings where the string length is 3 or more and the first and last character are same from a given list of strings.

Input List : ['cabc', 'xyza', 'abbc', '13221']
Output : 2

<PROGRAM CODE>

```
#1

input_list =['cabc', 'xyza','abbc','13221']

count=0

for x in range(len(input_list)):
    if len(input_list)>=3:
        if input_list[x][0]==input_list[x][-1]:
            count=count+1

print(count)
```

<RESULT>

```
In [279]: #1

input_list =['cabc', 'xyza','abbc','13221']

count=0

for x in range(len(input_list)):
    if len(input_list)>=3:
        if input_list[x][0]==input_list[x][-1]:
            count=count+1

print(count)
```

```
2
```

2. Write a Python program to get a list, sorted in increasing order by the first element in each tuple (inner list) from a list.

Input List : [[2, 6], [1, 2], [3, 4], [5, 3], [4, 1]]
Output : [[1, 2], [2, 6], [3, 4], [4, 1], [5, 3]]

<PROGRAM CODE>

#2

```
input_list = [[2, 6], [1, 2], [3, 4], [5, 3], [4, 1]]
input_list.sort()
print(input_list)
```

<RESULT>

```
In [280]: #2

          input_list = [[2, 6], [1, 2], [3, 4], [5, 3], [4, 1]]
          input_list.sort()
          print(input_list)

          [[1, 2], [2, 6], [3, 4], [4, 1], [5, 3]]
```

*** Select ONE arff file from e-class. Change it to csv file. The csv file must contain numbers and/or strings only, each of which is separated by commas. In dong so, you have to modify arff file by removing header part (% and @ part) of the data.**

3. Write Python code for the following tasks

1) read csvfile into a two dimension list (called "a_list")

e.g.: csvfile=

1 0 2 3 1

0 1 1 2 0

0 1 0 1 1

0 0 2 3 1

a_list=[[1,0,2,3,1], [0,1,1,2,0], [0,1,0,1,1], [0,0,2,3,1]]

<PROGRAM CODE>

#3-1

```
import pandas as pd
import numpy as np

flags_data = pd.read_csv('/Users/soojinlee/python/flags2.csv',header=0)
a_list = np.array(flags_data)

a_list
```

<RESULT>

In [281]:
```
#3-1

import pandas as pd
import numpy as np

flags_data = pd.read_csv('/Users/soojinlee/python/flags2.csv',header=0)
a_list = np.array(flags_data)

a_list
```

Out[281]:
```
array([[1, 'Afghanistan', 5, ..., 0, 'black', 'green'],
       [2, 'Albania', 3, ..., 0, 'red', 'red'],
       [3, 'Algeria', 4, ..., 0, 'green', 'white'],
       ...,
       [192, 'Zaire', 4, ..., 0, 'green', 'green'],
       [193, 'Zambia', 4, ..., 0, 'green', 'brown'],
       [194, 'Zimbabwe', 4, ..., 0, 'green', 'green']], dtype=object)
```

2) show the number of columns(attributes) and number of rows(records), respectively.

<PROGRAM CODE>

#3-2

column_num = len(a_list[0])
row_num = len(a_list)
print("the number of columns :",column_num ,"₩nthe number of rows :",row_num)

<RESULT>

In [282]:
```
#3-2

column_num = len(a_list[0])
row_num = len(a_list)
print("the number of columns :",column_num ,"\nthe number of rows :",row_num)
```

```
the number of columns : 31
the number of rows : 194
```

3) write a Python program that shows the first 5 rows from the "a_list".

<PROGRAM CODE>

#3-3

a_list[:5]

<RESULT>

```
In [283]: #3-3

          a_list[:5]
```

```
Out[283]: array([[1, 'Afghanistan', 5, 1, 648, 16, 10, 2, 0, 3, 5, 1, 1, 0, 1, 1,
                  1, 0, 'green', 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 'black', 'green'],
                 [2, 'Albania', 3, 1, 29, 3, 6, 6, 0, 0, 3, 1, 0, 0, 1, 0, 1, 0,
                  'red', 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 'red', 'red'],
                 [3, 'Algeria', 4, 1, 2388, 20, 8, 2, 2, 0, 3, 1, 1, 0, 0, 1, 0, 0,
                  'green', 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 'green', 'white'],
                 [4, 'American-Samoa', 6, 3, 0, 0, 1, 1, 0, 0, 5, 1, 0, 1, 1, 1, 0,
                  1, 'blue', 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 'blue', 'red'],
                 [5, 'Andorra', 3, 1, 0, 0, 6, 0, 3, 0, 3, 1, 0, 1, 1, 0, 0, 0,
                  'gold', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 'blue', 'red']],
                dtype=object)
```

4) write a Python program which randomly shuffles 'a_list' data

<PROGRAM CODE>

```
#3-4
import numpy as np
import copy

shuffled_list = copy.copy(a_list)
np.random.shuffle(shuffled_list)
shuffled_list
```

<RESULT>

```
In [284]: #3-4
          import numpy as np
          import copy

          shuffled_list = copy.copy(a_list)
          np.random.shuffle(shuffled_list)
          shuffled_list
```

```
Out[284]: array([[82, 'India', 5, ..., 0, 'orange', 'green'],
                 [134, 'Panama', 2, ..., 0, 'white', 'white'],
                 [138, 'Philippines', 6, ..., 0, 'blue', 'red'],
                 ...,
                 [90, 'Jamaica', 1, ..., 0, 'gold', 'gold'],
                 [192, 'Zaire', 4, ..., 0, 'green', 'green'],
                 [153, 'Somalia', 4, ..., 0, 'blue', 'blue']], dtype=object)
```

4. Using the "a_list" in question 3. write Python code for the following tasks

1) given a column(attribute) number, write a program that shows the values of the column.

<PROGRAM CODE>

```
#4-1
import numpy as np

column_x=input("put a column number : ")
column_values=a_list[:,int(column_x)-1:int(column_x)]
column_values_1d = column_values.ravel()
column_values_1d
```

<RESULT>

```
In [487]: #4-1
          import numpy as np

          column_x=input("put a column number : ")
          column_values=a_list[:,int(column_x)-1:int(column_x)]
          column_values_1d = column_values.ravel()
          column_values_1d

          put a column number : 3

Out[487]: array([5, 3, 4, 6, 3, 4, 1, 1, 2, 2, 6, 3, 1, 5, 5, 1, 3, 1, 4, 1, 5, 2,
                 4, 2, 1, 5, 3, 4, 5, 4, 4, 1, 4, 1, 4, 4, 2, 5, 2, 4, 4, 6, 1, 1,
                 3, 3, 3, 4, 1, 1, 2, 4, 1, 4, 4, 3, 2, 6, 3, 3, 2, 6, 4, 4, 3, 3,
                 4, 3, 3, 1, 1, 6, 1, 4, 4, 2, 1, 1, 5, 3, 3, 5, 6, 5, 5, 3, 5, 3,
                 4, 1, 5, 5, 5, 4, 6, 5, 5, 5, 4, 4, 4, 3, 3, 4, 4, 5, 5, 4, 3, 6,
                 4, 4, 1, 6, 3, 5, 1, 4, 4, 6, 5, 3, 1, 6, 1, 4, 4, 6, 5, 5, 3, 5,
                 5, 2, 6, 2, 2, 6, 3, 3, 1, 5, 3, 4, 3, 4, 5, 4, 4, 4, 5, 6, 4, 4,
                 5, 5, 3, 5, 4, 1, 1, 1, 4, 2, 4, 3, 3, 5, 5, 4, 5, 4, 6, 2, 4, 5,
                 1, 6, 5, 4, 3, 2, 1, 1, 5, 6, 3, 2, 5, 6, 3, 4, 4, 4], dtype=object)
```

2) show the reversed elements of q. 1) (We don't actually change the values a_list)

<PROGRAM CODE>

#4-2

column_values_1d[::-1]

<RESULT>

```
In [488]: #4-2
          column_values_1d[::-1]

Out[488]: array([4, 4, 4, 3, 6, 5, 2, 3, 6, 5, 1, 1, 2, 3, 4, 5, 6, 1, 5, 4, 2, 6,
                 4, 5, 4, 5, 5, 3, 3, 4, 2, 4, 1, 1, 1, 4, 5, 3, 5, 5, 4, 4, 6, 5,
                 4, 4, 4, 5, 4, 3, 4, 3, 5, 1, 3, 3, 6, 2, 2, 6, 2, 5, 5, 3, 5, 5,
                 6, 4, 4, 1, 6, 1, 3, 5, 6, 4, 4, 1, 5, 3, 6, 1, 4, 4, 6, 3, 4, 5,
                 5, 4, 4, 3, 3, 4, 4, 4, 5, 5, 5, 6, 4, 5, 5, 5, 1, 4, 3, 5, 3, 5,
                 5, 6, 5, 3, 3, 5, 1, 1, 2, 4, 4, 1, 6, 1, 1, 3, 3, 4, 3, 3, 4, 4,
                 6, 2, 3, 3, 6, 2, 3, 4, 4, 1, 4, 2, 1, 1, 4, 3, 3, 3, 1, 1, 6, 4,
                 4, 2, 5, 2, 4, 4, 1, 4, 1, 4, 4, 5, 4, 3, 5, 1, 2, 4, 2, 5, 1, 4,
                 1, 3, 1, 5, 5, 1, 3, 6, 2, 2, 1, 1, 4, 3, 6, 4, 3, 5], dtype=object)
```

5. Using the "a_list", write Python code for the following tasks

1) define a function "divide_train_test(in_list, prop)" function where

input: 1) in_list: a 2D list, 2) prop: proportion of training data

output: train_data (first "prop" percent of in_list), test_data (the rest of in_list)

<PROGRAM CODE>

#5-1

```
def divide_train_test(in_list,prop):
    train_ind = int(len(in_list)*prop)
    train_data = in_list[:train_ind]
    test_data = in_list[train_ind:]

    return [train_data, test_data]
```

<RESULT>

```
In [489]: #5-1

def divide_train_test(in_list,prop):
    train_ind = int(len(in_list)*prop)
    train_data = in_list[:train_ind]
    test_data = in_list[train_ind:]

    return [train_data, test_data]
```

2) run divide_train_test(a_list, prop) TWO times using prop=0.7, 0.9, respectively, and show the result.
e.g.: divide_train_test([[1,2,3], [5,1,8], [8,5,2], [0,3,6], [1,7,3]], 0.8)
returns [[[1,2,3], [5,1,8], [8,5,2]], [[0,3,6], [1,7,3]]]
 # train_data test_data

<PROGRAM CODE>

```
#5-2
print(divide_train_test(a_list,0.7))
print(divide_train_test(a_list,0.9))
```

<RESULT>

```
In [490]: #5-2
print(divide_train_test(a_list,0.7))
print(divide_train_test(a_list,0.9))

[array([[1, 'Afghanistan', 5, ..., 0, 'black', 'green'],
       [2, 'Albania', 3, ..., 0, 'red', 'red'],
       [3, 'Algeria', 4, ..., 0, 'green', 'white'],
       ...,
       [133, 'Pakistan', 5, ..., 0, 'white', 'green'],
       [134, 'Panama', 2, ..., 0, 'white', 'white'],
       [135, 'Papua-New-Guinea', 6, ..., 0, 'red', 'black']], dtype=object), array([[136, 'Parguay', 2, ..., 1, 're
d', 'blue'],
       [137, 'Peru', 2, ..., 0, 'red', 'red'],
       [138, 'Philippines', 6, ..., 0, 'blue', 'red'],
       ...,
       [192, 'Zaire', 4, ..., 0, 'green', 'green'],
       [193, 'Zambia', 4, ..., 0, 'green', 'brown'],
       [194, 'Zimbabwe', 4, ..., 0, 'green', 'green']], dtype=object)]
[array([[1, 'Afghanistan', 5, ..., 0, 'black', 'green'],
       [2, 'Albania', 3, ..., 0, 'red', 'red'],
       [3, 'Algeria', 4, ..., 0, 'green', 'white'],
       ...,
       [172, 'Togo', 4, ..., 0, 'red', 'green'],
       [173, 'Tonga', 6, ..., 0, 'white', 'red'],
       [174, 'Trinidad-Tobago', 2, ..., 0, 'white', 'white']],
      dtype=object), array([[175, 'Tunisia', 4, 1, 164, 7, 8, 2, 0, 0, 2, 1, 0, 0, 0, 1, 0, 0,
        'red', 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 'red', 'red'],
       [176, 'Turkey', 5, 1, 781, 45, 9, 2, 0, 0, 2, 1, 0, 0, 0, 1, 0, 0,
        'red', 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 'red', 'red'],
       [177, 'Turks-Cocos-Islands', 1, 4, 0, 0, 1, 1, 0, 0, 6, 1, 1, 1,
        1, 1, 0, 1, 'blue', 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 'white',
        'blue'],
       [178, 'Tuvalu', 6, 2, 0, 0, 1, 1, 0, 0, 5, 1, 0, 1, 1, 1, 0, 0,
        'blue', 0, 1, 1, 1, 9, 0, 0, 0, 0, 0, 'white', 'blue'],
```

```
 [179, 'UAE', 5, 1, 84, 1, 8, 2, 1, 3, 4, 1, 1, 0, 0, 1, 1, 0,
  'green', 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 'red', 'black'],
 [180, 'Uganda', 4, 1, 236, 13, 10, 5, 0, 6, 5, 1, 0, 0, 1, 1, 1,
  0, 'gold', 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 'black', 'red'],
 [181, 'UK', 3, 4, 245, 56, 1, 1, 0, 0, 3, 1, 0, 1, 0, 1, 0, 0,
  'red', 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 'white', 'red'],
 [182, 'Uruguay', 2, 3, 178, 3, 2, 0, 0, 9, 3, 0, 0, 1, 1, 1, 0, 0,
  'white', 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 'white', 'white'],
 [183, 'US-Virgin-Isles', 1, 4, 0, 0, 1, 1, 0, 0, 6, 1, 1, 1, 1, 1,
  0, 0, 'white', 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 'white', 'white'],
 [184, 'USA', 1, 4, 9363, 231, 1, 1, 0, 13, 3, 1, 0, 1, 0, 1, 0, 0,
  'white', 0, 0, 0, 1, 50, 0, 0, 0, 0, 0, 'blue', 'red'],
 [185, 'USSR', 5, 1, 22402, 274, 5, 6, 0, 0, 2, 1, 0, 0, 1, 0, 0,
  0, 'red', 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 'red', 'red'],
 [186, 'Vanuatu', 6, 2, 15, 0, 6, 1, 0, 0, 4, 1, 1, 0, 1, 0, 1, 0,
  'red', 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 'black', 'green'],
 [187, 'Vatican-City', 3, 1, 0, 0, 6, 0, 2, 0, 4, 1, 0, 0, 1, 1, 1,
  0, 'gold', 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 'gold', 'white'],
 [188, 'Venezuela', 2, 4, 912, 15, 2, 0, 0, 3, 7, 1, 1, 1, 1, 1, 1,
  1, 'red', 0, 0, 0, 0, 7, 0, 0, 1, 1, 0, 'gold', 'red'],
 [189, 'Vietnam', 5, 1, 333, 60, 10, 6, 0, 0, 2, 1, 0, 0, 1, 0, 0,
  0, 'red', 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 'red', 'red'],
 [190, 'Western-Samoa', 6, 3, 3, 0, 1, 1, 0, 0, 3, 1, 0, 1, 0, 1,
  0, 0, 'red', 0, 0, 0, 1, 5, 0, 0, 0, 0, 0, 'blue', 'red'],
 [191, 'Yugoslavia', 3, 1, 256, 22, 6, 6, 0, 3, 4, 1, 0, 1, 1, 1,
  0, 0, 'red', 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 'blue', 'red'],
 [192, 'Zaire', 4, 2, 905, 28, 10, 5, 0, 0, 4, 1, 1, 0, 1, 0, 0, 1,
  'green', 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 'green', 'green'],
 [193, 'Zambia', 4, 2, 753, 6, 10, 5, 3, 0, 4, 1, 1, 0, 0, 0, 1, 1,
  'green', 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 'green', 'brown'],
 [194, 'Zimbabwe', 4, 2, 391, 8, 10, 5, 0, 7, 5, 1, 1, 0, 1, 1, 1,
  0, 'green', 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 'green', 'green']],
dtype=object)]
```

6. Write Python code for the tasks.

1) define a function "min_max_avg" which takes a list of numbers and returns [minimum, maximum, average] of the list

e.g.: def min_max_avg(in_list):

<PROGRAM CODE>

#6-1

```
def min_max_avg(in_list):
    minmaxavg=[min(in_list),max(in_list),sum(in_list)/len(in_list)]
    return minmaxavg
```

<RESULT>

```
In [491]: #6-1

          def min_max_avg(in_list):
              minmaxavg=[min(in_list),max(in_list),sum(in_list)/len(in_list)]
              return minmaxavg
```

2) randomly generate 10 numbers and, calculate the average, minimum, and maximum values using above "min_max_avg" function

e.g.: mean_min_max([1,6,2,8,3,5,-4,2]) returns [-4, 8, 2.875]

<PROGRAM CODE>

#6-2

```
random_list=np.random.randint(-100,100,10)
print(random_list)
```

print(min_max_avg(random_list))

<RESULT>

```
In [492]: #6-2
random_list=np.random.randint(-100,100,10)
print(random_list)
print(min_max_avg(random_list))
```

```
[-72 -23  76 -42   4 -27  61 -65 -73 -94]
[-94, 76, -25.5]
```

3) define a function "equ_interval" which divides a value range into n equal intervals.
input: 1) list [min, max] of range, 2) number of intervals
output: list of (equal distance) intervals
e.g.: equ_interval([-4, 8], 3) returns [[-4,0], [0,4], [4,8]]

<PROGRAM CODE>

```
#6-3
def equ_interval(minmax,interval_num):
    minr=minmax[0]
    maxr=minmax[1]
    interval_size=int((maxr-minr)/interval_num)
    out_equ=[[0]*2 for i in range(interval_num)]
    out_equ[0][0]=0
    for x in range(int(interval_num)):
        out_equ[x][0]=minr+interval_size*x
        out_equ[x][1]=minr+interval_size*(x+1)

    return out_equ
```

<RESULT>

```
In [493]: #6-3
def equ_interval(minmax,interval_num):
    minr=minmax[0]
    maxr=minmax[1]
    interval_size=int((maxr-minr)/interval_num)
    out_equ=[[0]*2 for i in range(interval_num)]
    out_equ[0][0]=0
    for x in range(int(interval_num)):
        out_equ[x][0]=minr+interval_size*x
        out_equ[x][1]=minr+interval_size*(x+1)

    return out_equ
```

4) run equ_interval 2 times by using different values of list and number of intervals.

<PROGRAM CODE>

#6-4

```
print(equ_interval([-4,8],3))
print(equ_interval([0,25],5))
```

<RESULT>

```
In [494]: #6-4
          print(equ_interval([-4,8],3))
          print(equ_interval([0,25],5))

          [[-4, 0], [0, 4], [4, 8]]
          [[0, 5], [5, 10], [10, 15], [15, 20], [20, 25]]
```

7. Write Python code for the following tasks.

1) define a function "no_of_values" which takes a list and returns the number of values in the list.

<PROGRAM CODE>

#7-1

```
def no_of_values(in_list):
    return len(in_list) #1d
```

<RESULT>

```
In [495]: #7-1

          def no_of_values(in_list):
              return len(in_list) #1d
```

2) define a function "no_of_dis_val" which takes a list and returns the number of "distinct" values in the list.

e.g.: a_list=[0,1,1,2,0]

no_of_dis_val(a_list) returns 3 ==> 3 unique values

This means a_list contains 3 distinct values

<PROGRAM CODE>

#7-2

```
from collections import Counter

def no_of_dis_val(in_list):
    distinct_list= (Counter(in_list).keys())
```

return len(distinct_list) #1d

<RESULT>

```
In [425]: #7-2

          from collections import Counter

          def no_of_dis_val(in_list):
              distinct_list= (Counter(in_list).keys())

              return  len(distinct_list) #1d
```

3) for every attribute in "a_list", calculate the number of values and distinct values, respectively, using q 1) and q 2).

<PROGRAM CODE>

```
#7-3
for x in range(1,len(a_list[0])+1):
    attributes_val = a_list[:,x-1:x]
    into_1d = attributes_val.ravel()
    # print(into_1d)
    print("attribute : ",x, ", val :", no_of_values(into_1d),", dis :",no_of_dis_val(into_1d))
```

<RESULT>

```
In [501]: #7-3

          for x in range(1,len(a_list[0])+1):
              attributes_val = a_list[:,x-1:x]
              into_1d = attributes_val.ravel()
              # print(into_1d)
              print("attribute : ",x, ", val :", no_of_values(into_1d),", dis :",no_of_dis_val(into_1d))

attribute :  1 , val : 194 , dis : 194
attribute :  2 , val : 194 , dis : 194
attribute :  3 , val : 194 , dis : 6
attribute :  4 , val : 194 , dis : 4
attribute :  5 , val : 194 , dis : 136
attribute :  6 , val : 194 , dis : 48
attribute :  7 , val : 194 , dis : 10
attribute :  8 , val : 194 , dis : 8
attribute :  9 , val : 194 , dis : 5
attribute :  10 , val : 194 , dis : 12
attribute :  11 , val : 194 , dis : 8
attribute :  12 , val : 194 , dis : 2
attribute :  13 , val : 194 , dis : 2
attribute :  14 , val : 194 , dis : 2
attribute :  15 , val : 194 , dis : 2
attribute :  16 , val : 194 , dis : 2
attribute :  17 , val : 194 , dis : 2
attribute :  18 , val : 194 , dis : 2
attribute :  19 , val : 194 , dis : 8
attribute :  20 , val : 194 , dis : 4
attribute :  21 , val : 194 , dis : 3
attribute :  22 , val : 194 , dis : 2
attribute :  23 , val : 194 , dis : 3
attribute :  24 , val : 194 , dis : 14
attribute :  25 , val : 194 , dis : 2
attribute :  26 , val : 194 , dis : 2
attribute :  27 , val : 194 , dis : 2
attribute :  28 , val : 194 , dis : 2
attribute :  29 , val : 194 , dis : 2
attribute :  30 , val : 194 , dis : 7
attribute :  31 , val : 194 , dis : 8
```

4) plot a graphic table(e.g.: bar graph) by your favorite color using matplotlib as follows: X axis: index of attribute, Y axix: number of distinct values.

<PROGRAM CODE>

```
#7-4
import matplotlib.pyplot as plt
plot_x=[]
plot_y=[]
for x in range(1,len(a_list[0])+1):
    attributes_val = a_list[:,x-1:x]
    into_1d = attributes_val.ravel()
    # print(into_1d)
    plot_x.append(x)
    plot_y.append(no_of_dis_val(into_1d))
plt.plot(plot_x, plot_y,'red')
plt.xlabel('index of attribute')
plt.ylabel('number of distinct values')
plt.show()
```

<RESULT>

In [497]:
```
#7-4
import matplotlib.pyplot as plt
plot_x=[]
plot_y=[]
for x in range(1,len(a_list[0])+1):
    attributes_val = a_list[:,x-1:x]
    into_1d = attributes_val.ravel()
    # print(into_1d)
    plot_x.append(x)
    plot_y.append(no_of_dis_val(into_1d))
plt.plot(plot_x, plot_y,'red')
plt.xlabel('index of attribute')
plt.ylabel('number of distinct values')
plt.show()
```