CSE4214: Assignment

Monday, February 18, 2019

Zabir Haque

MD Abdullah Al Nasim

LAB 01

Question 1

append() and extend() function in python

(a) Append Function

append(): Appends object at the end.
```
x = [1, 2, 3]
x.append([4, 5])
print (x)
```

(b) What is the output?

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

(c) Extend Function

extend(): Extends list by appending elements from the iterable.

(d) Write the extend Code

```
x = [1, 2, 3]
x.extend([4, 5])
print (x)
```

(e) What is the output?

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

Question 2

Zip and Enumerate in python

(a) What does zip() do in python?

> **Python zip():** The zip() function take iterables (can be zero or more), makes iterator that aggregates elements based on the iterables passed, and returns an iterator of tuples.

(b) Why do we use zip in Python?

> **Python zip():** zip() in Python. The purpose of zip() is to map the similar index of multiple containers so that they can be used just using as single entity. Unzipping means converting the zipped values back to the individual self as they were.

(c) What is the use of zip in Python?

> Basically, .zip is a container itself. It holds the real file inside. Similarly, Python zip is a container that holds real data inside. Python zip function takes iterable elements as input, and returns iterator.

Question 3

How do you create a dictionary in Python?

> **i** Create a new dictionary-In order to construct a dictionary you can start with an empty one.
>
> **ii** Add a value to the dictionary.
>
> **iii** Remove a key and it's value.
>
> **iv** Check the length.
>
> **v** Test the dictionary.
>
> **vi** Get a value of a specified key.
>
> **vii** Print all keys with a for loop.
>
> **viii** Print all key and values.

Question 4

How to index in python?

> There are 2types of indexing.
> **1.Pure integer indexing**
> **2.Boolean indexing**

(a) Pure integer indexing:

Integer array indexing allows selection of arbitrary items in the array based on their N-dimensional index. Each integer array represents a number of indexes into that dimension.When the index consists of as many integer arrays as the array being indexed has dimensions, the indexing is straight forward, but different from slicing.
Example:
»> x = np.array([[1, 2], [3, 4], [5, 6]])
»> x[[0, 1, 2], [0, 1, 0]]
array([1, 4, 5])

(b) Boolean indexing:

This advanced indexing occurs when obj is an array object of Boolean type, such as may be returned from comparison operators. A single boolean index array is practically identical to x[obj.nonzero()] where, as described above, obj.nonzero() returns a tuple (of length obj.ndim) of integer index arrays showing the True elements of obj. However, it is faster when obj.shape == x.shape.
If obj.ndim == x.ndim, x[obj] returns a 1-dimensional array filled with the elements of x corresponding to the True values of obj. The search order will be row-major, C-style. If obj has True values at entries that are outside of the bounds of x, then an index error will be raised. If obj is smaller than x it is identical to filling it with False

LAB 02

Question 5

What is the Lambda expression?

A lambda function is a small anonymous function.A lambda function can take any number of arguments, but can only have one expression.**The power of lambda is better shown when you use them as an anonymous function inside another function.Syntax:-lambda arguments: expression**The expression is evaluated and returned.

Question 6

What is Slicing in python?

We can also call out a range of characters from the string. Say we would like to just print the word Shark. We can do so by creating a slice, which is a sequence of characters within an original string. With slices, we can call multiple character values by creating a range of index numbers separated by a colon string[x:y].Slicing can not only be used for lists, tuples or arrays, but custom data structures as well, with the slice object.**Slicing arguments: [start:end:step], [row:column], [start:end,start:end]**

Question 7

Listing 1: Lab 1 and 2 sample Code

```python
# coding: utf-8

# # Lab 01

# ## List

# In[5]:


for i in range (1, 10, 1):
    print(i);

list1 = [0,0,10]

print(min(list1))

L = ['A', 'B', 'C']

for x in L:
    print(x)

for i in range (len(L)):
    L[i] = L[i] + ' is zombie'

for x in L:
    print(x)


# ### append

# In[6]:


x = [1,2,3]
x.append([4,5])
print(x)
```

```
38
39
40  # ### extend
41
42  # In [7]:
43
44
45  x.extend([8,9])
46  print(x)
47
48
49  # ### enumerate, zip
50
51  # In [8]:
52
53
54  days = ["Sun","Mon","Tue"]
55  daysFr = ["Dim","Lun","Mar"]
56
57  for m in range (len (days)):
58      print(m+1,days[m])
59
60  for i,m in enumerate (days,start=1):
61      print(i,m)
62
63  for m in zip (days,daysFr):
64      print(m)
65
66  for i,m in enumerate (zip(days,daysFr),start=1):
67      print(i,m[0],"=",m[1],"in French")
68
69
70  # # Lab 02
71
72  # ## Lambda arguments: expression
73
74  # In [9]:
75
76
77  lambda_test = lambda a:a+10
78  print(lambda_test(5))
79
80  new_lambda_test = lambda q,w,y,z : q+w+y+z+10
81  print(new_lambda_test(20,20,20,20))
82  #exit()
83
84  pairs = [(1, 'one'),(2, 'two'), (3, 'three'), (4, 'four')]
85  pairs.sort( key =lambda pair: pair[1])
86  print(pairs)
```

```
87
88
89   # ## Anonymous functions
90
91   # In [13]:
92
93
94   #anonymous functions; accept multiple arguments but expression
95   def make_incrementor(n):
96       return lambda x : x+n
97   f = make_incrementor(42)
98   print(f(0))
99
100  evens = [2,4,6,8,10,12,14,16,18,20]
101
102  odds = [1, 3, 5,7,9]
103  #evenSquared_new = [e ** for e in evens if e>4 and e<16]
104
105  list1 = [2,3,4]
106  list2 = [3,1,7]
107
108  for i in range (len(list1)):
109      print(max(list1[i],list2[i]))
110
111  task1= [['a', 1], ['b', 2],['c', 3],['d', 4],['e', 5],
112  ['f', 6],['g', 7],['h', 8],['i', 9],['j', 10]]
113  print(len(task1))
114
115
116  # ### Import
117
118  # In [14]:
119
120
121  import numpy as np
122  from matplotlib import pyplot as plt
123  from sklearn import svm
124  from matplotlib import style
125  #from matplotlib.pyplot as plt
126
127
128  # In [18]:
129
130
131  list1 = [1,2,3,4,5,6]
132  list2 = [10,9,8,7,6,5]
133  #print(list1 * list2)
134  #convert to numpy
135  a1 = np.array(list1)
```

```
136  a2 = np.array(list2)
137  print(a1*a2)
138
139  print(np.linspace(0,10,5))
140  print(np.arange(0,10,2))
141
142  #a = np.array([1,2,3,4,5], dtype = np.float64)
143
144  a = np.arange(6)
145  print(a)
146
147  print('b')
148  b = np.arange(12).reshape(4,3)
149  print(b)
150  print(b.sum(axis=0)) #row wise for 0, 1 for column
151  print(b.min(axis=1))
152
153  print('c')
154
155
156  # ### Matrix
157
158  # In[19]:
159
160
161  #2matrix ,3 row,4 column
162  c = np.arange(24).reshape(2,3,4)
163  print(c)
164
165  a = np.array([[1,2,3],[4,5,6],[7,8,9]])
166  print(a[:, 0])
167  print(a[:, 1])
168  print(a[:, 2])
169
170  print(a[0:2, 1:3])
171  print(a[0:2, 0:2])
172
173
174  print(a[-8:17:1])
175  # last theke 8ta badd diye oita theke 17
176  #er ag prjnt 1 kore barbe
177
178
179  # ## SVM
180
181  # In[20]:
182
183
184  x = np.linspace(0,20,200)
```

```
185  y1 = np.exp(−0.1*x)*np.sin(x)
186  y2 = np.exp(−0.3*x)*np.sin(x)
187
188  plt.plot(x,y1)
189  plt.plot(x,y2)
190  plt.title('just enough!')
191  plt.show()
192
193
194
195  style.use("ggplot")
196
197  X = np.array([[1,2],
198               [5,8],
199               [1.5, 1.8],
200               [8,8],
201               [1,0.6],
202               [9,11]])
203
204  Y= [0,1,0,1,0,1]
205  clf = svm.SVC(kernel='linear', C = 1.0)
206  clf.fit(X,Y)
207  #print(clf.predict(np.array([10,10]).reshape(1, −1)))
208
209  w = clf.coef_[0]
210  a = −w[0]/w[1]
211  xx = np.linspace(0,12).reshape(−1,1)
212  yy = a*xx − (clf.intercept_[0] / w[1])
213
214  plt.plot(xx,yy, 'k—', label = 'Decision Boundary')
215
216
217  plt.scatter(X[:, 0], X[:, 1])
218  plt.legend()
219  plt.show()
```

Question 8

What is SVM?

> SVM is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs. Simply put, it does some extremely complex data transformations, then figures out how to seperate your data based on the labels or outputs you've defined.

LAB 03

Question 9

Listing 2: Lab 3 sample code

```
# coding: utf-8

# In[1]:


import pandas as pd
import numpy as np


# In[5]:


s = pd.Series([0, 1, 4, 9, 16, 25], name='squares')
print(s.index)
print(s.values, s.index)
print(s[2:4])


# In[6]:


pop2014 = pd.Series([100, 99.3, 95.5], index=['java','c','c++'])
print(pop2014)


# In[11]:


pop2015 = pd.Series({'java':100, 'c':99.3, 'c++':95.5})
print(pop2015)


# In[12]:


print(pop2014.index)


# In[14]:

```

```
42
43  print(pop2014.iloc[0:2])
44  print(pop2014.loc['c'])
45
46
47  # In[15]:
48
49
50  twoYears = pd.DataFrame({'2014': pop2014, '2015': pop2015 })
51  print(twoYears)
52
53
54  # In[16]:
55
56
57  twoYears['Average'] = 0.5* (twoYears['2014']+twoYears['2015'])
58  print(twoYears)
59
60
61  # In[18]:
62
63
64  test_data = pd.DataFrame
65  (np.random.choice(['a','b','c','d'], (3,3)),
66  index=[1,2,3], columns=['AA','BB','CC'])
67  print(test_data)
68
69
70  # In[19]:
71
72
73  open('tips.csv','r').readlines()[:10]
74
75
76  # In[20]:
77
78
79  tips=pd.read_csv('tips.csv')
80  tips.head()
81
82
83  # In[21]:
84
85
86  print(tips[:10])
87
88
89  # In[22]:
90
```

```
91
92    tips.mean()
93
94
95    # In[23]:
96
97
98    tips['tip'].mean()
99
100
101   # In[24]:
102
103
104   tips.dtypes
105
106
107   # In[25]:
108
109
110   tips.describe()
111
112
113   # In[26]:
114
115
116   tips.groupby('gender').mean()
117
118
119   # In[29]:
120
121
122   t=tips.groupby(['gender','smoker']).mean()
123   print(t['tip'])
124
125
126   # In[30]:
127
128
129   pd.pivot_table(tips, 'total_bill', 'gender', 'smoker')
130
131
132   # In[31]:
133
134
135   pd.pivot_table(tips, 'total_bill',
136   ['gender', 'smoker'], ['day','time'])
137
138
139   # In[34]:
```

```
140
141
142  from matplotlib import pyplot as plt
143  url='http://archive.ics.uci.edu/ml/
144  machine-learning-databases/iris/iris.data'
145  df = pd.read_csv(url)
146  df.head()
147
148
149  # In[35]:
150
151
152  df.columns = ['sepal_length', 'sepal_width',
153  'petal_length', 'petal_width', 'flower_type']
154  df.head()
155
156
157  # In[36]:
158
159
160  df['flower_type'] = df['flower_type'].astype('category')
161
162
163  # In[39]:
164
165
166  df.flower_type = df.flower_type.cat.rename_categories([0,1,2])
167  df.head()
168  print(df[40:50])
169
170
171  # In[40]:
172
173
174  print(len(df))
175
176
177  # In[41]:
178
179
180  df['flower_type'].describe()
181
182
183  # In[42]:
184
185
186  df.hist()
187  plt.show()
188
```

Question 9 continued on next page...   Page 13 of 14

```
189
190   # In [43]:
191
192
193   pd.scatter_matrix(df, diagonal='kde')
194   plt.show()
195
196
197   # In [44]:
198
199
200   df.to_csv('iris_normalized.csv')
```

Question 10

Is most late riser likes movies most or sports?

I need to find the accuracy of the movies or sports.By using the accuracy I can find out what are their activity.Then I can find it.

Question 11

Find the number of people how many wants to become software engineer?

I need to use classifier algorithm to categorized future plan of every people.