CSE4214: Assignment

Monday, March 11, 2019

Zabir Haque

MD Abdullah Al Nasim
150104085

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

Question 2 continued on next page. . .

(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

Question 4 continued on next page. . .

(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?

Question 6 continued on next page...

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

```

# 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)
```

Question 7 continued on next page...

```python


# ### extend

# In[7]:


x.extend([8,9])
print(x)


# ### enumerate, zip

# In[8]:


days = ["Sun","Mon","Tue"]
daysFr = ["Dim","Lun","Mar"]

for m in range (len (days)):
    print(m+1,days[m])

for i,m in enumerate (days,start=1):
    print(i,m)

for m in zip (days,daysFr):
    print(m)

for i,m in enumerate (zip(days,daysFr),start=1):
    print(i,m[0],"=",m[1],"in French")


# # Lab 02

# ## Lambda arguments: expression

# In[9]:


lambda_test = lambda a:a+10
print(lambda_test(5))

new_lambda_test = lambda q,w,y,z : q+w+y+z+10
print(new_lambda_test(20,20,20,20))
#exit()

pairs = [(1, ’one’),(2, ’two’), (3, ’three’), (4, ’four’)]
pairs.sort( key=lambda pair: pair[1])
print(pairs)
```

Question 7 continued on next page...

```


# ## Anonymous functions

# In[13]:


#anonymous functions; accept multiple arguments but expression
def make_incrementor(n):
    return lambda x : x+n
f = make_incrementor(42)
print(f(0))

evens = [2,4,6,8,10,12,14,16,18,20]

odds = [1, 3, 5,7,9]
#evenSquared_new = [e ** for e in evens if e>4 and e<16]

list1 = [2,3,4]
list2 = [3,1,7]

for i in range (len(list1)):
    print(max(list1[i],list2[i]))

task1= [['a', 1], ['b', 2],['c', 3],['d', 4],['e', 5],
['f', 6],['g', 7],['h', 8],['i', 9],['j', 10]]
print(len(task1))


# ### Import

# In[14]:


import numpy as np
from matplotlib import pyplot as plt
from sklearn import svm
from matplotlib import style
#from matplotlib.pyplot as plt


# In[18]:


list1 = [1,2,3,4,5,6]
list2 = [10,9,8,7,6,5]
#print(list1 * list2)
#convert to numpy
a1 = np.array(list1)
```

```
a2 = np.array(list2)
print(a1*a2)

print(np.linspace(0,10,5))
print(np.arange(0,10,2))

#a = np.array([1,2,3,4,5], dtype = np.float64)

a = np.arange(6)
print(a)

print('b')
b = np.arange(12).reshape(4,3)
print(b)
print(b.sum(axis=0)) #row wise for 0, 1 for column
print(b.min(axis=1))

print('c')


# ### Matrix

# In[19]:


#2matrix ,3 row,4 column
c = np.arange(24).reshape(2,3,4)
print(c)

a = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(a[:, 0])
print(a[:, 1])
print(a[:, 2])

print(a[0:2, 1:3])
print(a[0:2, 0:2])


print(a[-8:17:1])
# last theke 8ta badd diye oita theke 17
#er ag prjnt 1 kore barbe


# ## SVM

# In[20]:


x = np.linspace(0,20,200)
```

Question 7 continued on next page. . .

```
y1 = np.exp(-0.1*x)*np.sin(x)
y2 = np.exp(-0.3*x)*np.sin(x)

plt.plot(x,y1)
plt.plot(x,y2)
plt.title('just enough!')
plt.show()



style.use("ggplot")

X = np.array([[1,2],
             [5,8],
             [1.5, 1.8],
             [8,8],
             [1,0.6],
             [9,11]])

Y= [0,1,0,1,0,1]
clf = svm.SVC(kernel='linear', C = 1.0)
clf.fit(X,Y)
#print(clf.predict(np.array([10,10]).reshape(1, -1)))

w = clf.coef_[0]
a = -w[0]/w[1]
xx = np.linspace(0,12).reshape(-1,1)
yy = a*xx - (clf.intercept_[0] / w[1])

plt.plot(xx,yy, 'k--', label = 'Decision Boundary')


plt.scatter(X[:, 0], X[:, 1])
plt.legend()
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]:

```

Question 9 continued on next page...

```

print(pop2014.iloc[0:2])
print(pop2014.loc['c'])


# In[15]:


twoYears = pd.DataFrame({'2014': pop2014, '2015': pop2015 })
print(twoYears)


# In[16]:


twoYears['Average'] = 0.5* (twoYears['2014']+twoYears['2015'])
print(twoYears)


# In[18]:


test_data = pd.DataFrame
(np.random.choice(['a','b','c','d'], (3,3)),
index=[1,2,3], columns=['AA','BB','CC'])
print(test_data)


# In[19]:


open('tips.csv','r').readlines()[:10]


# In[20]:


tips=pd.read_csv('tips.csv')
tips.head()


# In[21]:


print(tips[:10])


# In[22]:

```

```

tips.mean()


# In[23]:


tips[’tip’].mean()


# In[24]:


tips.dtypes


# In[25]:


tips.describe()


# In[26]:


tips.groupby(’gender’).mean()


# In[29]:


t=tips.groupby([’gender’,’smoker’]).mean()
print(t[’tip’])


# In[30]:


pd.pivot_table(tips, ’total_bill’, ’gender’, ’smoker’)


# In[31]:


pd.pivot_table(tips, ’total_bill’,
[’gender’, ’smoker’], [’day’,’time’])


# In[34]:
```

```


from matplotlib import pyplot as plt
url='http://archive.ics.uci.edu/ml/
machine-learning-databases/iris/iris.data'
df = pd.read_csv(url)
df.head()


# In[35]:


df.columns = ['sepal_length', 'sepal_width',
'petal_length', 'petal_width', 'flower_type']
df.head()


# In[36]:


df['flower_type'] = df['flower_type'].astype('category')


# In[39]:


df.flower_type = df.flower_type.cat.rename_categories([0,1,2])
df.head()
print(df[40:50])


# In[40]:


print(len(df))


# In[41]:


df['flower_type'].describe()


# In[42]:


df.hist()
plt.show()

```

Question 9 continued on next page. . .

```

# In[43]:


pd.scatter_matrix(df, diagonal='kde')
plt.show()


# In[44]:


df.to_csv('iris_normalized.csv')
```

Question 10

What is KNN?

A powerful classification algorithm used in pattern recognition.K nearest neighbors stores all available cases and classifies new cases based on a similarity measure(e.g distance function). A non-parametric lazy learning algorithm (An Instancebased Learning method

Question 11

Write algorithm of KNN

i All the instances correspond to points in an n-dimensional feature space.

ii Each instance is represented with a set of numerical attributes.

iii Each of the training data consists of a set of vectors and a class label associated with each vector.

iv Classification is done by comparing feature vectors of different K nearest points.

v Select the K-nearest examples to E in the training set.

vi Assign E to the most common class among its K-nearest neighbors.

Question 12

How to choose K?

i If K is too small it is sensitive to noise points.

ii Larger K works well. But too large K may include majority points from other classes.

iii Rule of thumb is K < sqrt(n), n is number of examples.

Question 13 continued on next page...

Question 13

Strength of KNN

i If Very simple and intuitive.

ii Can be applied to the data from any distribution.

iii Good classification if the number of samples is large enough.

Question 14

Weakness of KNN

i Takes more time to classify a new example.

ii need to calculate and compare distance from new example to all other examples.

iii Choosing k may be tricky.

iv Need large number of samples for accuracy.