Introduction to Machine Learning Applications

Spring 2021

Lecture-3

Lydia Manikonda

manikl@rpi.edu

Rensselaer

Today’s agenda

- Python Basics – Recap and Loops, Conditionals, Functions
- Including class exercises

Recap on Basics

- Python variables
- Data structures
  - List
  - Dictionary
  - Tuple
  - Set
- Mini class exercise

Mini Quiz

1. Suppose list1 is [3, 4, 5, 20, 5, 25, 1, 3], what is list1 after list1.pop(1)?
2. Given a string (Example: “machinElearning”) count the number of vowels present in the string.

Python fundamentals

Loops, conditionals, functions

Loops

Loops in Python

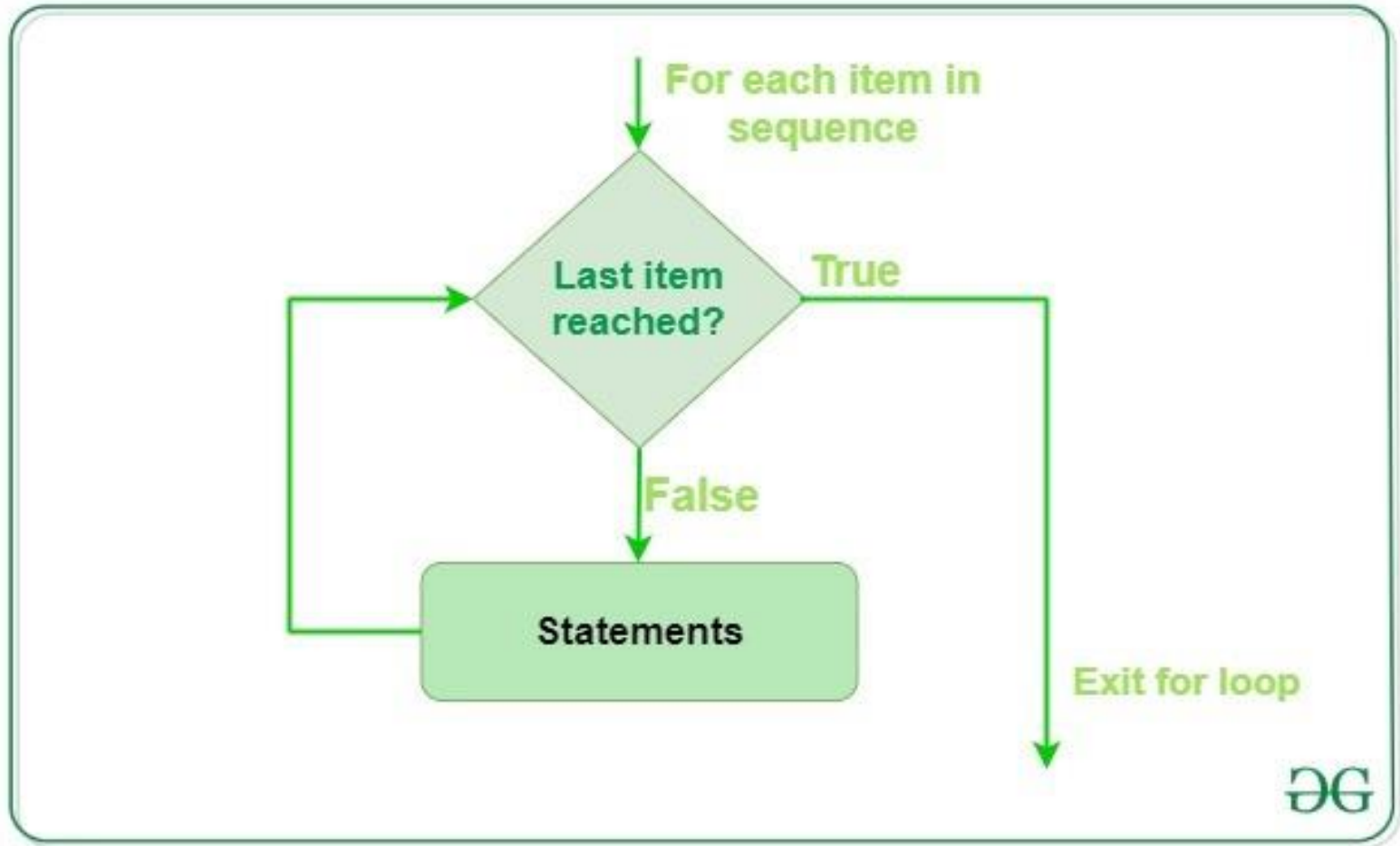
For

```
for iterator_var in sequence:
        statements(s)
```

While

```
while expression:
        statement(s)
```

for


For each item in sequence
Last item reached?
True
False
Statements
Exit for loop

for

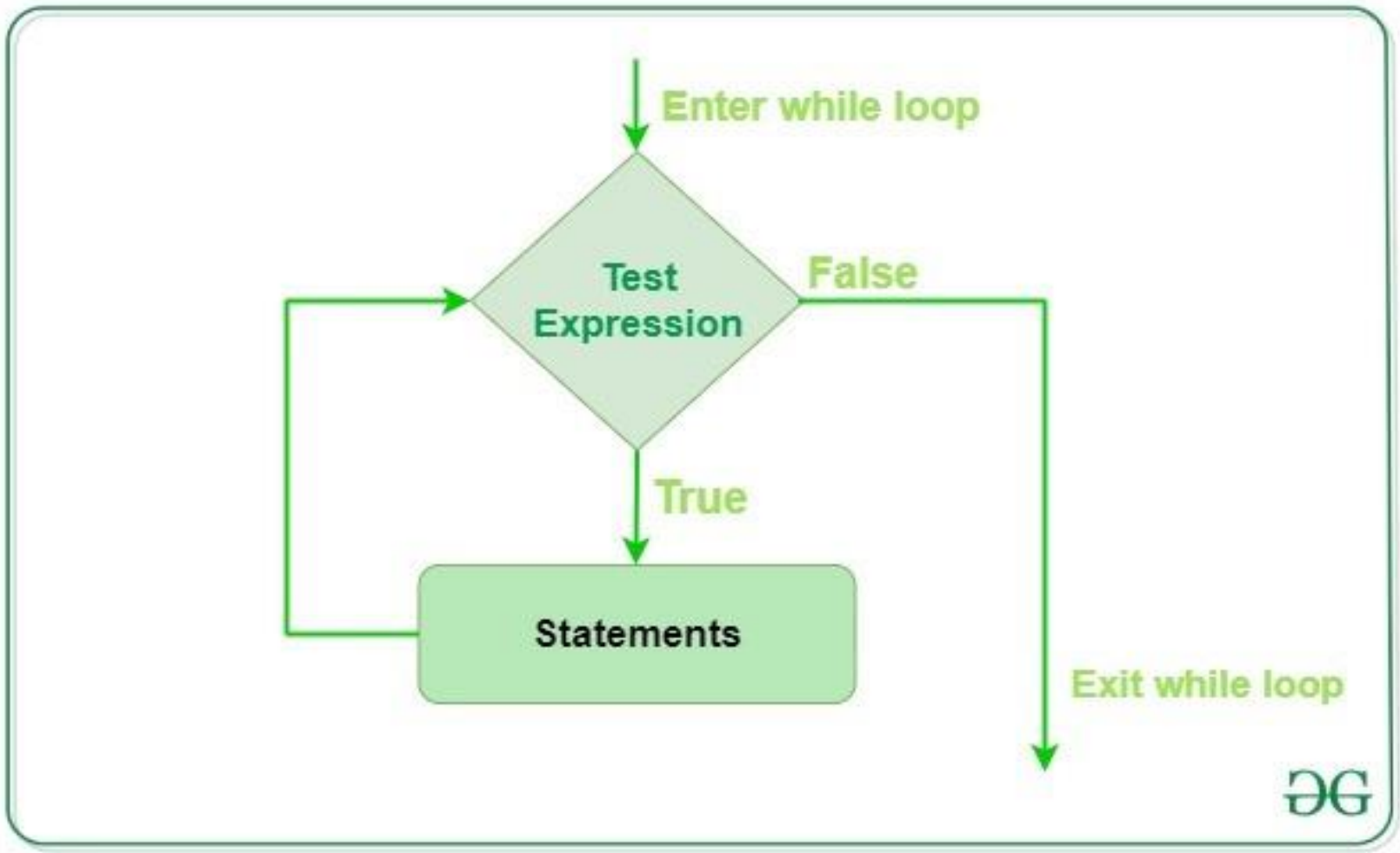
```
>> print("List Iteration")
>> list1 = [“hello", “world”]
>> for i in list1:
        print(i)

>> for i in range(0,10,1):
        print(i)

>> for letter in ‘predictiveanalytics':
        if letter == 'e' or letter == 's':
                continue
        print('Current Letter :', letter)
```

for loop -- Example

Using the *for* loop print a new list as an output with all the squares of the elements in a given list [1, 2, 3, 4, 5].

Output: [1, 4, 9, 16, 25]

while


Enter while loop
Test Expression
False
True
Statements
Exit while loop

while

```
>> count = 0
>> while (count < 3):
        count = count + 1
        print("Hello world!")
```

While

```
>> i = 0
>> a = ‘machinelearning'
>> while i < len(a):
        if a[i] == 'e' or a[i] == 's':
                i += 1
                continue
        print('Current Letter :', a[i])
        i += 1
```

while loop – Example

Write a program to print this format when given a string str1="machine" **using while loop**:

machine
machin
machi
mach
mac
ma
m

Conditionals

If condition

```
if condition:
        statement1
        statement2
# Statements to execute if condition is true
```

```
if condition:
        statement1
statement2
# Here if the condition is true, if block will consider only statement1
```

Example

```
i = 10
if (i > 15):
        print ("10 is less than 15")
print ("I am Not in if")
```

If-else

```
if (condition):
        # Executes this block if condition is true
else:
        # Executes this block if condition is false
```

Example

```
i = 20
if (i < 15):
        print ("i is smaller than 15")
        print ("i'm in if Block")
else:
        print ("i is greater than 15")
        print ("i'm in else Block")
print ("i'm not in if and not in else Block")
```

If-elif-else

```
if (condition):
        statement
elif (condition):
        statement
. .
else:
        statement
```

if-else-if

```
>> num1 = 4
>> if(num1%2 == 0):
        print("Num1 is even")
>> elif(num1%2==1):
        print("Num1 is odd")
>> else:
        print(“It never comes to this section")
```

Class Exercise

Given a list l1= [1,2,3,4,5,6,7,8,9,10], print the maximum value till a given index using a *for* loop and *if* statement.

Output: [1,2,3,4,5,6,7,8,9,10]

Functions

Functions

- Set of statements that take inputs and perform certain computations

```
>> def FindEven( x ):
        if (x % 2 == 0):
                print "even"
        else:
                print "odd"
>> FindEven (2)
>> FindEven (3)
```

Functions Example

```
def myFun(x):
        x[0] = 20

lst = [10, 11, 12, 13, 14, 15]
myFun(lst)
print(lst)
```

Pass by Reference

- When we pass a reference and change the received reference to something else, the connection between passed and received parameter is broken.

Pass by Reference – Example-1

```
def myFun(x):
        x = [20, 30, 40]

lst = [10, 11, 12, 13, 14, 15]
myFun(lst)
print(lst)
```

Pass by Reference – Example-2

```
def myFunc(x):
        x = 20
x = 10
myFunc(x)
print(x)
```

Pass by Reference

```
def swap(x, y):
        temp = x;
        x = y;
        y = temp;
# Driver code
x = 2
y = 3
swap(x, y)
print(x)
print(y)
```

Default Arguments

```
def myFun(x, y=50):
        print("x: ", x)
        print("y: ", y)

myFun(10)
```

Keyword arguments

```
def student(firstname, lastname):
        print(firstname, lastname)

student(firstname =‘John', lastname =‘Smith')
student(lastname =‘Smith', firstname =‘John')
```

Variable length Arguments

```
def myFun(*argv):
        for arg in argv:
                print (arg)

myFun('Hello', 'Welcome', 'to', 'GeeksforGeeks')
```

Lambda Functions – Anonymous functions

- ***lambda arguments: expression***

```
>> def cube(y):
        return y*y*y;
>> g = lambda x: x*x*x


>> print(g(7))
>> print(cube(5))
```

Example – Intersection of 2 lists

```
>> def ArrIntersect(a1, a2):
        result = list(filter(lambda x: x in a1, a2))
        print ("Intersection : ",result)


>> arr1 = [1, 3, 4, 5, 7]
>> arr2 = [2, 3, 5, 6]
>> ArrIntersect (arr1,arr2)
```

Class exercises

- Python notebook

Numpy

Numpy

- Fundamental package for scientific computing

- Numpy is a general-purpose array-processing package
- Used for high-performance multidimensional array computations

Numpy – Arrays

- A numpy array is a grid of values, all values are of same type
- The number of dimensions is the **rank** of an array
- A tuple of integers giving the size of an array along each dimension is called the **shape** of an array

- Initialize using nested python lists
- Access using square brackets

Numpy – Arrays

- Declaring the package

```
import numpy as np
```

- Creating an array of rank 1

```
arr = np.array([1, 2, 3])
```

- Creating an array of rank 2

```
arr = np.array([1, 2, 3], [4, 5, 6])
```

Numpy – Arrays

- Create an array with rank 1

```
>> a = np.array([1, 2, 3])
```

- Print the shape of this array

```
>> print(a.shape)
>> (3, )
```

- Print the elements at different indices

```
>> print(a[0], a[1], a[2])
>> 1 2 3
```

- Change an element of the array

```
>> a[0] = 10
```

- Print the array

```
>> print(a)
>> [10, 2, 3]
```

Numpy – Arrays

```
>> a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12], [13, 14, 15, 16]])
>> print(a)
```

Using slicing method printing a range of array

```
>> sliced_a = a[:2, ::2]
>> print(sliced_a)
```

Printing elements at specific indices

```
>> print(a[[1, 2, 1, 3],[1, 0, 2, 3]])
```

Numpy – Arrays and Functions

```
>> a = np.zeros((2, 2))
>> print(a)
>> [[0.  0.], [0.  0.]]

>> b = np.ones((1, 2))
>> print(b)
>> [[1.  1.]]

>> c = np.full((2,2), 7)
>> print(c)
>> [[7.  7.], [7.  7.]]

>> d = np.eye(2)
>> print(d)
>> [[1.  0.], [0.  1.]]

>> e = np.random.random((2, 2))
>> print(e)
>> [[], []]
```

Numpy – Arrays

- Datatypes of arrays need not be defined – numpy tries to guess the datatype

```
>> a = np.array([1.1, 2.2])
>> print(a.dtype)


>> a = np.array([1, 2], dtype=np.int64)
>> print(a.dtype)
```

Numpy – Math operations

```
>> a = np.array([[1, 2], [3, 4]], dtype=np.float64)
>> b = np.array([[4, 3], [2, 1]], dtype=np.float64)

>> sum_ab = np.add(a, b)
>> print(sum_ab)

>> sum_a = np.sum(a)
>> print(sum_a)
>> sqrt_a = np.sqrt(a)
>> print(sqrt_a)

>> trans_a = a.T
>> print(trans_a)
```

Numpy Exercises

- Python notebook