Applied Analytics & Predictive Modeling

Spring 2021

Lecture-2

Lydia Manikonda

manikl@rpi.edu

Rensselaer

Some of the slides adapted from Intro to Data Mining Tan et al. 2nd edition

Agenda

- Revision – Intro to Data Mining
- Revision – Python basics – variables, data structures

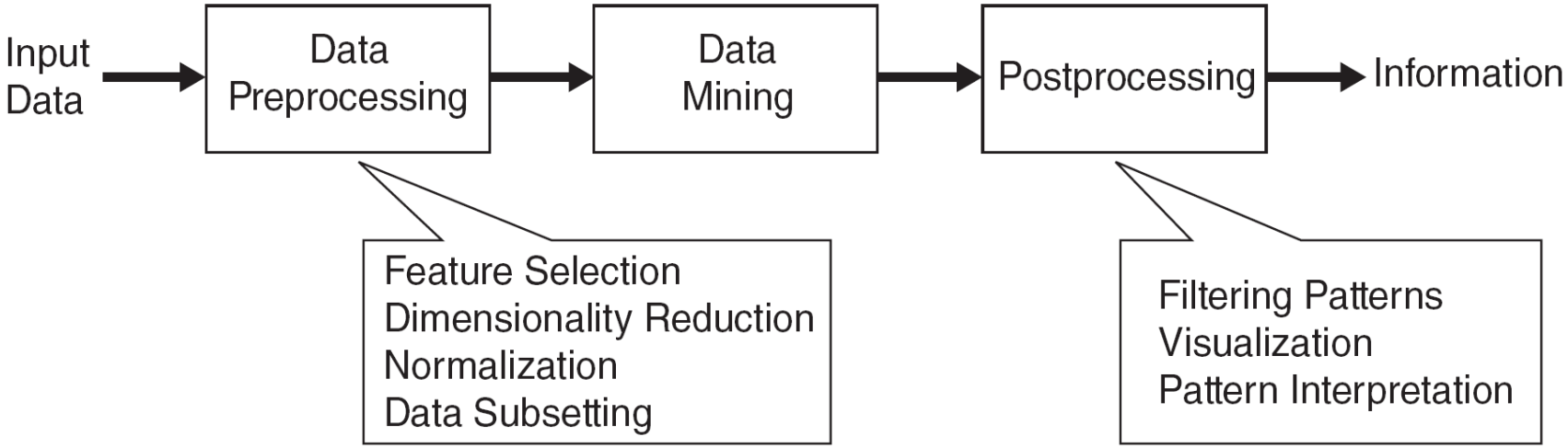
- Python basics – loops, conditionals, functions, packages
- Colab – Jupyter notebook environment
- Research Translation Exercise – for 6000 level only

Why Data Mining? Commercial Viewpoint

- Lots of data is being collected and warehoused
  - Web data
    - Yahoo has Peta Bytes of web data
    - Facebook has billions of active users
  - purchases at department/ grocery stores, e-commerce
    - Amazon handles millions of visits/day
  - Bank/Credit Card transactions
- Computers have become cheaper and more powerful
- Competitive Pressure is Strong
  - Provide better, customized services for an edge (e.g. in Customer Relationship Management)

What is Data Mining?

- Many Definitions
  - Non-trivial extraction of implicit, previously unknown and potentially useful information from data
  - Exploration & analysis, by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns



What is NOT Data Mining?

- What is not Data Mining?
  - Look up phone number in phone directory
  - Query a Web search engine for information about “Amazon”

- What is Data Mining?
  - Certain names are more prevalent in certain US locations (O’Brien, O’Rourke, O’Reilly… in Boston area)
  - Group together similar documents returned by search engine according to their context (e.g., Amazon rainforest, Amazon.com)

Python fundamentals

Basics, loops, conditionals, functions, packages

Basics

Language introduction, setup, variables, data structures

First program in Python

```
>> #Begins -- Comments
>> print(“Hello World”)
>> #Ends – Comments
```

is used for single line comment in Python

""" this is a comment """ is used for multi line comments

Variables and Data Structures

- In programming languages such as C, C++ or C#, you need to declare the **type of variables** exclusively.
  - Data types can be int, float, char, String, etc.
- Python – take a variable and the value assigned to it automatically tells the data type.

```
>> myVar = 2 #int
>> print(myVar)

>> myVar2 = 2.5 #float
>> print(myVar2)

>> myVar3 = “Hello World!” #string
>> print(myVar3)
```

Data Structures

- Create a variable and assign any value you want!

- Python has 4 types of inbuilt data structures
- **List**
- **Dictionary**
- **Tuple**
- **Set**

List

- Most basic data structure in Python programming language.
- Mutable data structure
  - Elements of this list can be altered after creating the data structure

1. append() – used to add elements in the list
2. insert() – used to add elements in the list at a certain index till the last element

List

append()

```
>> #Create an empty list
>> list1=[]

>> #Append elements to the list
>> list1.append(2)
>> list1.append(4.5)
>> list1.append("four")

>> print(list1)
```

insert()

```
>> list1 = [1, 2, 3, 4, 5]
>> list1.insert(5, 10)
>> print(list1)

>> list1.insert(1,10)
>> list1.insert(8,20)
>> print(list1)
```

Dictionary

- An unordered collection of data values in Python.
- It is used to store data values like a map.
- Unlike other Data Types that hold only single value as an element, Dictionary holds <key:value> pair.
- Dictionary values can be of any datatype – can be duplicated no repeated keys.

Dictionary

```
>> diction1={}
>> print(diction1)

>> diction1 = {1: ‘First’, 2: ‘Python’, 3: ‘Dictionary’}
>> print(diction1)

>> diction1 = {1: ‘First’, 2: [1,2,3,4]}
>> print(diction1)
```

Dictionary

```
>> diction1={}
>> diction1[0]=2
>> diction1[1]=4
>> diction1[2]=“Hello”
>> diction1[“3”]=“It is possible”
```

Tuple

- Tuple is a collection of Python objects much like a list.
- The sequence of values stored in a tuple can be of any type, and they are indexed by integers.
- The important difference between a list and a tuple is that **tuples are immutable**.

Tuple

```
>> tuple1=()
>> print(tuple1)

>> tuple1=(1,2,3,4,5)
>> print(tuple1)

>> tuple1=('hello', 'world')
>> print(tuple1)
```

Tuple

```
>> list1=[1,2,3,4,5]
>> list1[1]=3
>> print(list1)

>> list1=[7,6,5,4,3,2,1,0]
>> print(list1)

>> mytuple=(0,1,2,3,4,5,6,7)
>> print(mytuple)
>> mytuple[1]=3
```

Concatenate tuples

```
>> Tuple1 = (0, 1, 2, 3)
>> Tuple2 = (‘hello’, ‘world’)
>> Tuple3 = Tuple1 + Tuple2
>> print(Tuple3)
```

Set

- Set is an unordered collection of data type that is iterable, mutable and has no duplicate elements.
- Highly optimized method compared to list because it is very easy to check whether an element is present or not.

Set

```
>> set1 = set()
>> print(set1)

>> set1 = set("Predictive")
>> print(set1)

>> s1="Predictive"
>> set1 = set(s1)
>> print(set1)

>> set1=set(["I", "love", "analytics"])
>> print(set1)
```

Take input from the user

- input() function is used to take input from the user

```
>> # Python program to get input from user
>> name = input("Enter the course name: ")
>> # user entered the name ‘PredictiveModel'
>> print("I registered for ", name)
```

Loops

Loops in Python

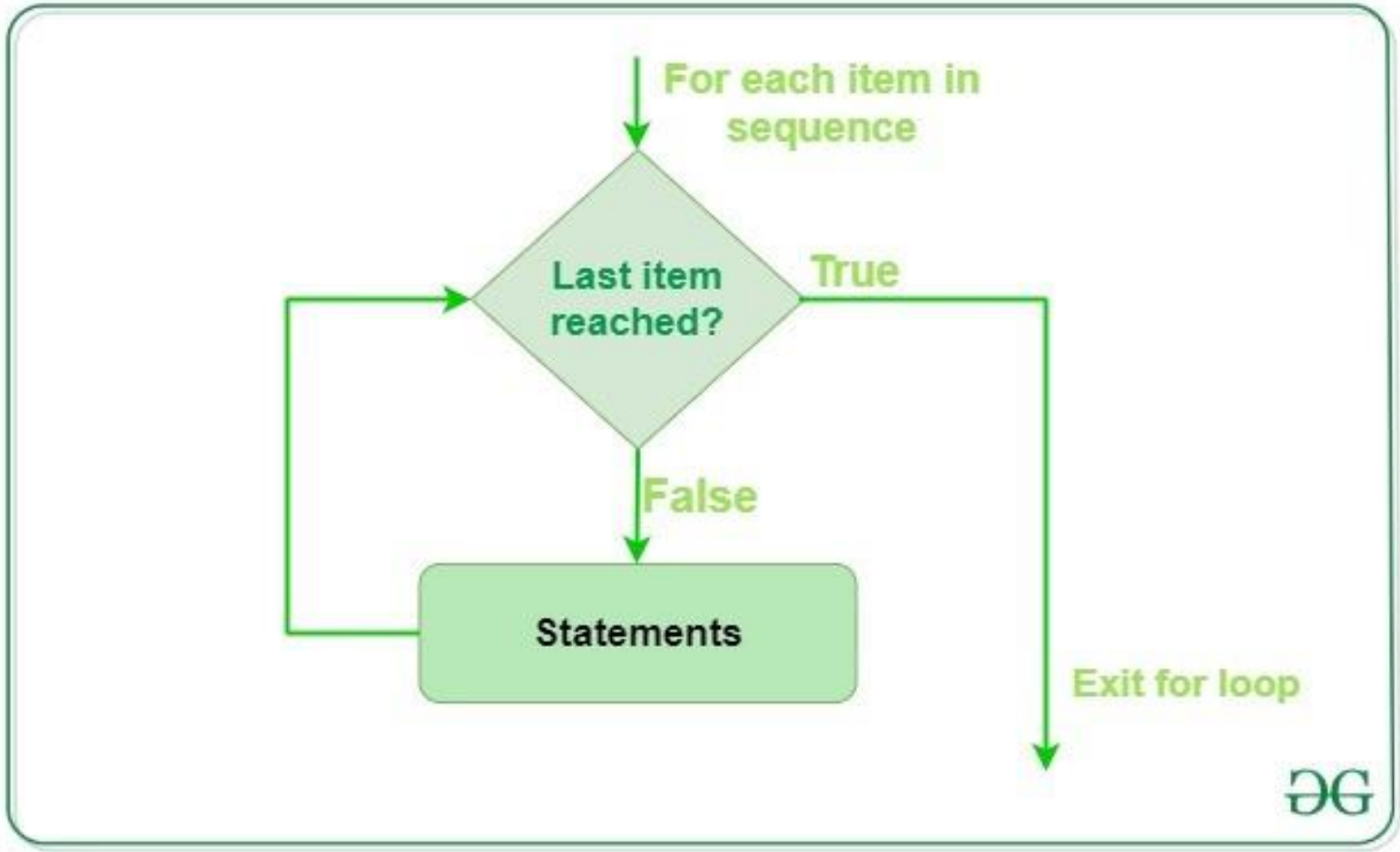
For

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

While

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

for



Geeksforgeeks.org

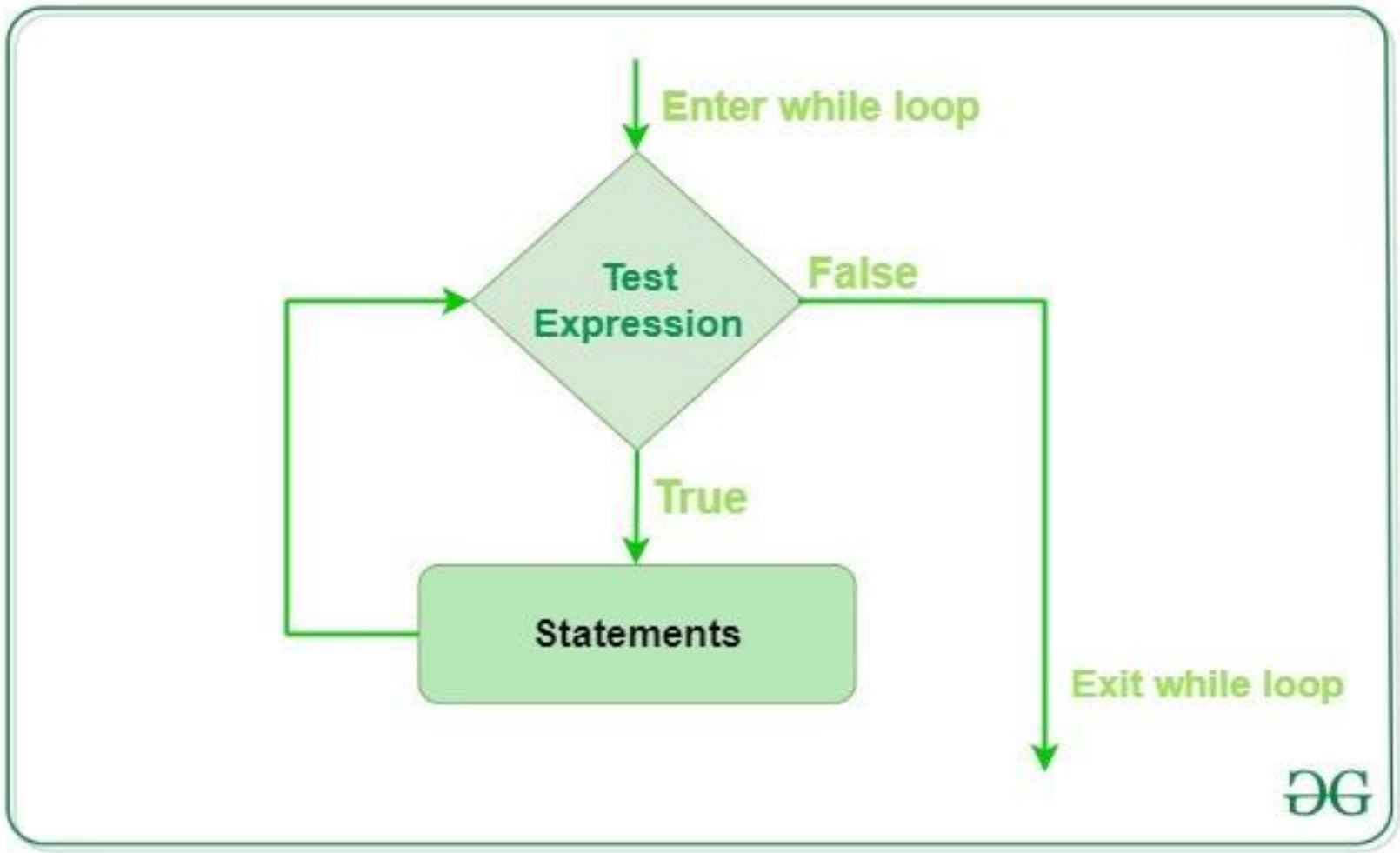
for

```
>> 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

Given a list l1= [1,2,3,4,5,6,7,8,9,10], print only the even number indices using a *for* loop.

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 = ‘predictiveanalytics'
>> 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

Given a list l1= [1,2,3,4,5,6,7,8,9,10], print only the numbers at the odd indices using a *while* loop.

Conditionals

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 prints these statements")
```

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)
```

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))
```

Functions examples

1. Write a function Square that takes an integer argument and outputs the square value of this argument. For example, if the input is 3, output should be 9.
2. y = 8
   z = lambda x : x * y
   print z(6)

Revising all the concepts – Exercises

1. Given a list of keywords, create a dictionary of the keywords and their frequencies as the values.

   Input: Keywords = [‘hello’, ‘I’, ‘am’, ‘fine’, ‘but’, ‘fine’, ‘is’, ‘fine’, ’hello’, ‘to’, ‘you’, ‘fine’]

   Dictionary: {‘hello’: 2, ‘I’:1, ‘am’:1, ‘fine’:4, ‘but’:1, ‘is’:1, ‘to’:1, ‘you’:1 }

Packages

3 different packages that we will use in this class

Packages – Numpy

Numerical computations

Packages – Pandas

Data handling

Research Translation Exercise

- **For 6000 level ONLY**
- Due: 02/04/2021 11:59 pm ET via Blackboard
- No late submissions are allowed

- Choose one visualization of your choice: https://github.com/d3/d3/wiki/Gallery (code is available on this site to modify/use with your own dataset)
- Write a 1-page summary on this visualization. This should include:
  - Assess the visualization based on the data set and the motivation for that visual representation.
  - What are the technical aspects that you appreciate?
  - What would you like to change or add?
  - Any other significant technical aspects that you can think of?