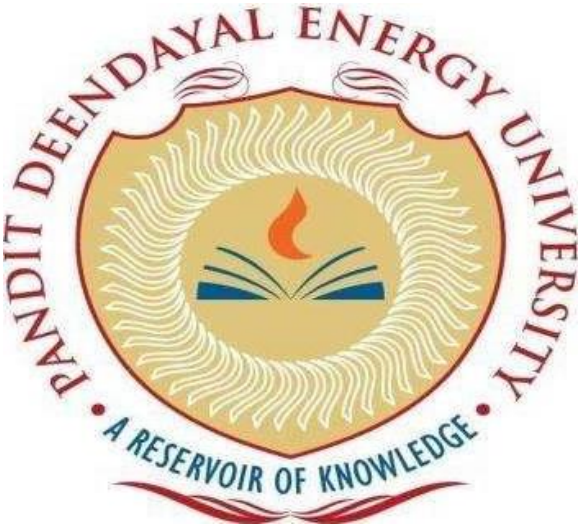
Pandit Deendayal Energy University, Gandhinagar

School of Technology

Department of Computer Science & Engineering

Big Data Analytics LAB

(23CP309P)



Name: **OM M PATEL**

Enrolment No: **21BCP094**

Semester: **VI**

Division: **II (G3)**

Branch: **Computer Science and Engineering**

Lab1 - OM M PATEL - 21BCP094

BIG DATA ANALYTICS LAB

```
1  print("Hello World")
```

```
Hello World
```

Command took 2.15 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 09:40:04 on Big_Data_Lab

Cmd 2

```
1  #Factorial
2  def factorial(n):
3      if(n==0 or n==1):
4          return 1
5      else:
6          return n * factorial(n-1)
7
8  n = int(input("Enter your number"))
9  print(factorial(n))
10
```

```
Enter your number 7
5040
```

Command took 4.95 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 09:50:50 on Big_Data_Lab

Cmd 3

```
1  def prime(n):
2      if n <= 1:
3          print(f"{n} is not a prime number")
4      else:
5          for i in range(2, int(n**0.5) + 1):
6              if n % i == 0:
7                  print(f"{n} is not a prime number")
8                  break
9          else:
10             print(f"{n} is a prime number")
11
12 n = int(input("Enter your number"))
13 prime(n)
14
```

```
Enter your number 4
4 is not a prime number
```

Command took 1.82 minutes -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 09:53:14 on Big_Data_Lab

Cmd 4

```
1  ## reverse a string
2
3  n = "Hello"
4  p=''.join(reversed(n))
5  print(p)
6
```

```
olleH
```

Command took 0.08 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 09:56:30 on Big_Data_Lab

Cmd 5

```
1  ## Reverse String Function
2
3  def Rever_str(n):
4      str = ''
5      for char in n:
6          str = char + str
7      return str
8
9  n = 'Hello'
10 print(Rever_str(n))
```

```
H
e
l
l
o
olleH
```

Command took 0.09 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 09:59:49 on Big_Data_Lab

Cmd 6

```
1  ##fibonacci Sequence
2  ## 0 + 1 + 1 + 2 + 3 + 5 + 8 + 13....
3  def fib(n, a=0, b=1):
4      if n == 0:
5          return []
6      elif n == 1:
7          return [a]
8      else:
9          return [a] + fib(n - 1, b, a + b)
10
11 n = 10
12 print(fib(n))
13
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

Command took 0.11 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 10:24:58 on Big_Data_Lab

Cmd 7

```
## frequency of a character
def freq(n):
    frequency = {}
    for char in n:
        if char in frequency:
            frequency[char] += 1
        else:
            frequency[char] = 1
    return frequency

n = "pneumonoultramicroscopicsilicovolcanoconiosis"
print(freq(n))

```

```
{'p': 2, 'n': 4, 'e': 1, 'u': 2, 'm': 2, 'o': 9, 'l': 3, 't': 1, 'r': 2, 'a': 2, 'i': 6, 'c': 6, 's': 4, 'v': 1}
```

Command took 0.13 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 10:26:06 on Big_Data_Lab

Cmd 8

%fs is used to access file storage system , also we cant use normal os functions , we need spark for that,

```
%fs
ls /FileStore/tables/Iris.csv

```

Table +

	path	name	size	modificationTime
1	dbfs:/FileStore/tables/Iris.csv	Iris.csv	5107	1704862664000

1 row | 1.80 seconds runtime

Command took 1.80 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 10:32:46 on Big_Data_Lab

Cmd 9

```
# spark is imp to work with lots of CSV files
# in csv our 1st line is called header, which is the columns
df = spark.read.format('csv').option('inferSchema','true').option('header','true').load('/FileStore/tables/Iris.csv')
```

(2) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [Id: integer, SepalLengthCm: double ... 4 more fields]

Command took 1.51 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 10:43:18 on Big_Data_Lab

Cmd 10

```
## the above 2 jobs helps in saving computational power, it will take the same energy. helps in saving cloud costs.
## Creating/uploading files sends them to the Hadoop file system and converts it acc to its methodologies
## jobs are written in java
## data bricks can communication with different cloud systems, can do encryption and decryption and etc.
## Data bricks is efficient way to connect with big data without managing stages, java jobs, parallel computing, optimizing,creating structure
display(df)
```

(1) Spark Jobs

Table +

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
1	1	5.1	3.5	1.4	0.2	Iris-setosa
2	2	4.9	3	1.4	0.2	Iris-setosa
3	3	4.7	3.2	1.3	0.2	Iris-setosa
4	4	4.6	3.1	1.5	0.2	Iris-setosa
5	5	5	3.6	1.4	0.2	Iris-setosa
6	6	5.4	3.9	1.7	0.4	Iris-setosa

150 rows | 0.56 seconds runtime

Command took 0.56 seconds -- by om.pce21@sot.pdpu.ac.in at 10/01/2024, 10:48:18 on Big_Data_Lab

Lab2 - OM M PATEL - 21BCP094

```
%python
print('Hello World')
```

```
Hello World
```

Command took 1.69 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:19:01 on My Cluster_Lab_2

Cmd 2

SCALA BASICS

Cmd 3

typecasting isn't needed in scala , + in scala refers to concatenation

Cmd 4

```
var value  =  5
```

```
value: Int = 5
```

Command took 0.70 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:20:53 on My Cluster_Lab_2

Cmd 5

```
println("Hello World")
println("hey"+value)

```

```
Hello World
hey5
```

Command took 0.84 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:20:56 on My Cluster_Lab_2

Cmd 6

```
var num = List(1,2,3,45,6)
```

```
num: List[Int] = List(1, 2, 3, 45, 6)
```

Command took 0.98 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:21:37 on My Cluster_Lab_2

Cmd 7

```
var str = List("om","krishna","a","b")
```

```
str: List[String] = List(om, krishna, a, b)
```

Command took 0.80 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:22:29 on My Cluster_Lab_2

Cmd 8

```
var k = List(1,2,"om","Krishna")
```

```
k: List[Any] = List(1, 2, om, Krishna)
```

Command took 0.41 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:22:45 on My Cluster_Lab_2

Cmd 9

```
num.head
```

```
res2: Int = 1
```

Command took 0.87 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:23:07 on My Cluster_Lab_2

Cmd 10

```
// where we dont give any parameter , we shouldn't use paranthesis , as it works like property
num.tail
```

```
res3: List[Int] = List(2, 3, 45, 6)
```

Command took 0.94 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:23:51 on My Cluster_Lab_2

Cmd 11

```
num.sum
```

```
res4: Int = 57
```

Command took 0.46 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:24:08 on My Cluster_Lab_2

Cmd 12

```
// num.take throwns first n values
num.take(2)
```

```
res5: List[Int] = List(1, 2)
```

Command took 0.33 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:24:36 on My Cluster_Lab_2

Cmd 13

```
var temp = List(1,1,1,2,3,3,2,4,5,6,7,8,7,8,5)
//variable_name.distinct returns the distinct values from the list
temp.distinct
```

```
temp: List[Int] = List(1, 1, 1, 2, 3, 3, 2, 4, 5, 6, 7, 8, 7, 8, 5)
res6: List[Int] = List(1, 2, 3, 4, 5, 6, 7, 8)
```

Command took 0.49 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:27:02 on My Cluster_Lab_2

Cmd 14

```
temp(0)
```

```
res7: Int = 1
```

Command took 0.30 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:27:14 on My Cluster_Lab_2

Cmd 15

```
// variable_name(index) used for getting the element at that index
```

```
1   // variable_name(index) used for getting the element at that index
2   temp(4)
```

```
res9: Int = 3
```

Command took 0.42 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:27:46 on My Cluster_Lab_2

Cmd 16

```
1   temp(12)
```

```
res10: Int = 7
```

Command took 0.34 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:27:53 on My Cluster_Lab_2

Cmd 17

```
1   temp(15)
2   // index outofboundexception will be thrown , which shows that java runs in background of scala.
```

```
  IndexOutOfBoundsException: 15
```

Command took 0.49 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:28:04 on My Cluster_Lab_2

Cmd 18

```
1   num(1) = 10
2   //as lists are immutable (cant update) in java , so here too we get the error
3   // lists can be concanted
4
```

```
command-554296061079856:1: error: value update is not a member of List[Int]
num(1) = 10
^
```

Command took 0.14 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:30:09 on My Cluster_Lab_2

Cmd 19

```
1   temp.size
2   // variable_name.size/length both gives length of our list
```

```
res15: Int = 15
```

Command took 0.31 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:30:47 on My Cluster_Lab_2

Cmd 20

```
1   temp.length
```

```
res16: Int = 15
```

Command took 0.33 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:30:54 on My Cluster_Lab_2

Cmd 21

```
1   println("Min Value: ",temp.min) //returns the min value from the list
2   println("Max Value: ",temp.max) //returns max value from list
```

```
(Min Value: ,1)
(Max Value: ,8)
```

Command took 0.37 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:32:57 on My Cluster_Lab_2

Cmd 22

```
1   temp.isEmpty
```

```
res24: Boolean = false
```

Command took 0.77 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:34:16 on My Cluster_Lab_2

Cmd 23

```
1   // As java is faster than python , scala's statistical analysis is faster than python.
2
3   var empty = List()
4   empty.isEmpty
```

```
empty: List[Nothing] = List()
res25: Boolean = true
```

Command took 0.32 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:36:35 on My Cluster_Lab_2

Cmd 24

```
1   var num = Array(1,2,3,4,5,6)
2   //we can create two tyoes of array num , string/characters
```

```
num: Array[Int] = Array(1, 2, 3, 4, 5, 6)
```

Command took 0.34 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:37:40 on My Cluster_Lab_2

Cmd 25

```
1   var str = Array("om","Krishna","dev")
```

```
str: Array[String] = Array(om, Krishna, dev)
```

Command took 0.47 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:38:01 on My Cluster_Lab_2

Cmd 26

```
1   //Array are mutable
2   num(1)
```

```
res27: Int = 2
```

Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:38:20 on My Cluster_Lab_2

Cmd 27

```
1   num(1) = 55
```

```
num(1) = 99
```

Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:38:37 on My Cluster_Lab_2

Cmd 28

```
num
```

```
res32: Array[Int] = Array(1, 55, 3, 4, 5, 6)
```

Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:38:40 on My Cluster_Lab_2

Cmd 29

```
str(2)
```

```
res34: String = dev
```

Command took 0.25 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:38:50 on My Cluster_Lab_2

Cmd 30

```
str(2) = "Dev"
```

Command took 0.26 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:39:01 on My Cluster_Lab_2

Cmd 31

```
str
```

```
res36: Array[String] = Array(om, Krishna, Dev)
```

Command took 0.28 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:39:05 on My Cluster_Lab_2

Cmd 32

```
str.tail
```

```
res37: Array[String] = Array(Krishna, Dev)
```

Command took 0.32 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:39:27 on My Cluster_Lab_2

Cmd 33

```
num.tail
// List functions are applicable on array to
```

```
res38: Array[Int] = Array(55, 3, 4, 5, 6)
```

Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:39:38 on My Cluster_Lab_2

Cmd 34

```
// Array Buffer  :

// Mutable Collection: ArrayBuffer is a mutable collection in Scala, meaning you can modify its elements after creation.
// Resizable: It allows dynamic resizing, making it convenient for scenarios where the size of the collection needs to change.
// Efficient Operations: ArrayBuffer provides efficient append and remove operations compared to regular arrays, which require resizing or creati
// Indexed Access: Elements in an ArrayBuffer are accessed by index, similar to arrays.
```

Command took 0.30 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:41:48 on My Cluster_Lab_2

Cmd 35

```
import scala.collection.mutable.ArrayBuffer
```

```
import scala.collection.mutable.ArrayBuffer
```

Command took 0.22 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:40:46 on My Cluster_Lab_2

Cmd 36

```
var cars = new ArrayBuffer[String]
```

```
cars: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer()
```

Command took 0.82 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:43:14 on My Cluster_Lab_2

Cmd 37

```
cars += "Jaguar"
```

```
res49: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer(Jaguar)
```

Command took 0.31 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:45:05 on My Cluster_Lab_2

Cmd 38

```
cars += "BMW"
```

```
res50: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer(Jaguar, BMW)
```

Command took 0.33 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:45:06 on My Cluster_Lab_2

Cmd 39

```
cars
```

```
res51: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer(Jaguar, BMW)
```

Command took 0.25 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:45:09 on My Cluster_Lab_2

Cmd 40

```
cars += "AUDI"
```

```
res52: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer(Jaguar, BMW, AUDI)
```

Command took 0.35 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:45:12 on My Cluster_Lab_2

Cmd 41

```
// .trimend(n) lets u trim n values from end
cars.trimEnd(1)
```

```
2   cars.trimEnd(1)
```

Command took 0.28 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:45:14 on My Cluster_Lab_2

Cmd 42

```
1   cars
```

```
res54: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer(Jaguar, BMW)
```

Command took 0.33 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:45:17 on My Cluster_Lab_2

Cmd 43

```
1   cars.insert(2,"range rover")
```

Command took 0.77 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:46:12 on My Cluster_Lab_2

Cmd 44

```
1   cars
```

```
res57: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer(Jaguar, BMW, range rover, Ford)
```

Command took 0.26 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:46:21 on My Cluster_Lab_2

Cmd 45

```
1   cars.insert(0,"TATA")
```

Command took 0.29 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:46:40 on My Cluster_Lab_2

Cmd 46

```
1   cars
```

```
res60: scala.collection.mutable.ArrayBuffer[String] = ArrayBuffer(TATA, Jaguar, BMW, range rover, Ford)
```

Command took 0.29 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:46:43 on My Cluster_Lab_2

Cmd 47

```
1   cars.isEmpty
```

```
res61: Boolean = false
```

Command took 1.19 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:49:23 on My Cluster_Lab_2

Cmd 48

```
1   cars.size
```

```
res62: Int = 5
```

Command took 0.21 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:49:58 on My Cluster_Lab_2

Cmd 49

```
1   // Unique Scala functionalities
2
3   //Transform & Map
4   var num = List(1,2,3,4)
```

```
num: List[Int] = List(1, 2, 3, 4)
```

Command took 0.37 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:53:16 on My Cluster_Lab_2

Cmd 50

```
1   val a = num.map( x => x+3 )
```

```
a: List[Int] = List(4, 5, 6, 7)
```

Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:53:51 on My Cluster_Lab_2

Cmd 51

```
1   val b = a.map( x => -x)
```

```
b: List[Int] = List(-4, -5, -6, -7)
```

Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:54:27 on My Cluster_Lab_2

Cmd 52

```
1   val c = b.map( x => x*x )
```

```
c: List[Int] = List(16, 25, 36, 49)
```

Command took 0.64 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:54:50 on My Cluster_Lab_2

Cmd 53

```
1   val fruits = List("Orange","banana","Apple","kiwi","grapes")
```

```
fruits: List[String] = List(Orange, banana, Apple, kiwi, grapes)
```

Command took 0.26 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:55:17 on My Cluster_Lab_2

Cmd 54

```
1   var fruits_size = fruits.map(x => (x,x.size))
```

```
fruits_size: List[(String, Int)] = List((Orange,6), (banana,6), (Apple,5), (kiwi,4), (grapes,6))
```

Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:56:55 on My Cluster_Lab_2

Cmd 55

```
1   val fruit_char = fruits.map( x => x.toCharArray )
```

```
fruit_char: List[Array[Char]] = List(Array(O, r, a, n, g, e), Array(b, a, n, a, n, a), Array(A, p, p, l, e), Array(k, i, w, i), Array(g, r, a, p, e, s))
```

Command took 0.47 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 09:58:54 on My Cluster_Lab_2

Cmd 56

```
1  val fruit_larger = fruits.filter( x => ( x.length>5 ))
2
```

```
fruit_larger: List[String] = List(Orange, banana, grapes)
```

Command took 0.32 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:00:50 on My Cluster_Lab_2

Cmd 57

```
1  var fruit_smallest = fruits.filter( x => (x.size <= 4))
```

```
fruit_smallest: List[String] = List(kiwi)
```

Command took 0.20 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:01:34 on My Cluster_Lab_2

Cmd 58

```
1  //practice
2  // 1) multiply by 10
3  // 2) filter more than 75
4  // 3) filter and save between 60 & 75 ,convert back to 1 to 10
5  var ratings  = List(2.4,5.6,8.9,7.3)
```

```
ratings: List[Double] = List(2.4, 5.6, 8.9, 7.3)
```

Command took 0.23 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:07:30 on My Cluster_Lab_2

Cmd 59

```
1  var rat_10 = ratings.map( x=> x*10)
2
```

```
rat_10: List[Double] = List(24.0, 56.0, 89.0, 73.0)
```

Command took 0.74 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:07:32 on My Cluster_Lab_2

Cmd 60

```
1  var rat_high = rat_10.filter( x => x>75)
```

```
rat_high: List[Double] = List(89.0)
```

Command took 0.16 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:09:52 on My Cluster_Lab_2

Cmd 61

```
1  var between_rat = rat_10.filter( x => (x>60 && x<75))
```

```
between_rat: List[Double] = List(73.0)
```

Command took 0.66 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:11:29 on My Cluster_Lab_2

Cmd 62

```
1  var normal_rating = between_rat.map( x => x/10)
```

```
normal_rating: List[Double] = List(7.3)
```

Command took 0.21 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:11:36 on My Cluster_Lab_2

Cmd 63

```
1  // functions
```

Cmd 64

```
1  //here 100,200 are default value
2  // Double = is must before {} , as it specifies what we will getin return
3  // not writing = / double will throw error
4
5  def add(a:Double=100,b:Double=200):Double =
6  {
7    var sum: Double = 0
8    sum = a+b
9    return sum
10 }
```

```
add: (a: Double, b: Double)Double
```

Command took 0.18 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:29:00 on My Cluster_Lab_2

Cmd 65

```
1  add(11,12)
```

```
res70: Double = 23.0
```

Command took 0.23 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:29:04 on My Cluster_Lab_2

Cmd 66

```
1  //conditional
2
3  def cond(a:Double=2,b:Double=3):Boolean = {
4    a > b
5  }
```

```
cond: (a: Double, b: Double)Boolean
```

Command took 0.20 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:30:46 on My Cluster_Lab_2

Cmd 67

```
1  cond(2,3)
```

```
res71: Boolean = false
```

Command took 0.17 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:30:51 on My Cluster_Lab_2

Cmd 68

```
1  //if else
2  def CheckNumber(num: Int): String = {
3    if(num<0) { "Negative Number "}
4    else if (num == 0) {" Zero "}
5    else { " Positive Number "}
6  }
7
8
```

```
CheckNumber: (num: Int)String
```

Command took 0.71 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:47:48 on My Cluster_Lab_2

Cmd 69

```
1  var answer = CheckNumber(-5)
2
```

```
answer: String = "Negative Number "
```

Command took 0.22 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:48:27 on My Cluster_Lab_2

Cmd 70

```
1  var answer = CheckNumber(7)
2
```

```
answer: String = " Positive Number "
```

Command took 0.16 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:48:35 on My Cluster_Lab_2

Cmd 71

```
1  var answer = CheckNumber(0)
2
```

```
answer: String = " Zero "
```

Command took 0.21 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:48:44 on My Cluster_Lab_2

Cmd 72

```
1  def multiplyMatrices(matrixA: Array[Array[Int]], matrixB: Array[Array[Int]]): Array[Array[Int]] = {
2
3    require(matrixA.nonEmpty && matrixA(0).nonEmpty && matrixB.nonEmpty && matrixB(0).nonEmpty,
4      "Input matrices must not be empty")
5
6    val rowsA = matrixA.length
7    val colsA = matrixA(0).length
8    val colsB = matrixB(0).length
9
10   require(matrixB.length == colsA, "Number of columns in matrixA must be equal to the number of rows in matrixB")
11
12   val result = Array.ofDim[Int](rowsA, colsB)
13
14   for (i <- 0 until rowsA) {
15     for (j <- 0 until colsB) {
16       var sum = 0
17       for (k <- 0 until colsA) {
18         sum += matrixA(i)(k) * matrixB(k)(j)
19       }
20       result(i)(j) = sum
21     }
22   }
23
24   result
25 }
26
27 // Example usage:
28 val matrixA = Array(Array(1, 2, 3), Array(4, 5, 6))
29 val matrixB = Array(Array(7, 8, 0), Array(11, 12, 9),Array(1,2,3))
30
31 val resultMatrix = multiplyMatrices(matrixA, matrixB)
32
33 // Display the result matrix
34 println("RESULT")
35 resultMatrix.foreach(row => println(row.mkString("\t")))
36 println("\n\n")
37
```

```
RESULT
32      38      27
89      104     63
```

```
multiplyMatrices: (matrixA: Array[Array[Int]], matrixB: Array[Array[Int]])Array[Array[Int]]
matrixA: Array[Array[Int]] = Array(Array(1, 2, 3), Array(4, 5, 6))
matrixB: Array[Array[Int]] = Array(Array(7, 8, 0), Array(11, 12, 9), Array(1, 2, 3))
resultMatrix: Array[Array[Int]] = Array(Array(32, 38, 27), Array(89, 104, 63))
```

Command took 0.36 seconds -- by om.pce21@sot.pdpu.ac.in at 17/01/2024, 10:44:00 on My Cluster_Lab_2

Lab3 - OM M PATEL - 21BCP094

```
1  //converting List to RDD
2  var a = sc.parallelize(List("A","B","C","D"))
```

```
a: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[0] at parallelize at command-1164433993013367:1
```

Command took 17.99 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:29:34 on My Cluster lab 3

Cmd 2

```
1  // RDD Transformations are spark operations when executed on RDD, it result in a single or multiple new without updating an ex
2  // Resilient Distributed Dataset- kind of data structure - it is scattered data struture in a cluster
3  // it helps in memory efficiency as it is scattered
4  // easy to process parallel computing as it is very lazy
5  // Driver program can be accessed. by parallezing existing connection
```

Cmd 3

```
1  val b = a.map(x => (x,1))
2  b.collect
```

(1) Spark Jobs

```
b: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[1] at map at command-1164433993013369:1
res0: Array[(String, Int)] = Array((A,1), (B,1), (C,1), (D,1))
```

Command took 3.67 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:43:04 on My Cluster lab 3

Cmd 4

```
1  val c = a.map(x => (x.length))
2  c.collect
```

(1) Spark Jobs

```
c: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[2] at map at command-1164433993013370:1
res1: Array[Int] = Array(1, 1, 1, 1)
```

Command took 0.78 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:45:44 on My Cluster lab 3

Cmd 5

```
1  val a = sc.parallelize(List(1,2,3,4,5)).map(x => (x,x+1,x+2))
2  a.collect
```

(1) Spark Jobs

```
a: org.apache.spark.rdd.RDD[(Int, Int, Int)] = MapPartitionsRDD[4] at map at command-1164433993013371:1
res2: Array[(Int, Int, Int)] = Array((1,2,3), (2,3,4), (3,4,5), (4,5,6), (5,6,7))
```

Command took 1.07 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:48:18 on My Cluster lab 3

Cmd 6

```
1  val a = sc.parallelize(List(1,2,3,4,5)).flatMap(x => List(x,x+1,x+2))
2  a.collect
```

(1) Spark Jobs

```
a: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[6] at flatMap at command-1164433993013372:1
res4: Array[Int] = Array(1, 2, 3, 2, 3, 4, 3, 4, 5, 4, 5, 6, 5, 6, 7)
```

Command took 0.49 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:49:41 on My Cluster lab 3

Cmd 7

```
1  val rrda = sc.parallelize(List("aaaa","bbbb","ccc"))
2  rrda.filter(_.equals("aaaa")).collect
```

(1) Spark Jobs

```
rrda: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[7] at parallelize at command-1164433993013373:1
res6: Array[String] = Array(aaaa)
```

Command took 0.58 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:50:43 on My Cluster lab 3

Cmd 8

```
1  var city = sc.parallelize(List(("Mumbai",4000),("Delhi",2000),("Chennai",10000),("Kolkatta",7000),("Ahmedabad",3000)))
```

```
city: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[10] at parallelize at command-1164433993013374:1
```

Command took 0.38 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:57:25 on My Cluster lab 3

Cmd 9

```
1  city.filter(_._1.contains("ai")).collect
```

(1) Spark Jobs

```
res8: Array[(String, Int)] = Array((Mumbai,4000), (Chennai,10000))
```

Command took 0.49 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 09:57:29 on My Cluster lab 3

Cmd 10

```
1  val fc = city.filter(city => city._2 > 1000 && city._2 < 7000)
2  fc.collect
```

(1) Spark Jobs

```
fc: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[15] at filter at command-1164433993013376:1
res15: Array[(String, Int)] = Array((Mumbai,4000), (Delhi,2000), (Ahmedabad,3000))
```

Command took 0.61 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:03:22 on My Cluster lab 3

Cmd 11

```
1  val ab = city.filter(city => city._1.contains("ta") || city._1.contains("hi"))
2  ab.collect
3
```

(1) Spark Jobs

```
ab: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[18] at filter at command-1164433993013377:1
res18: Array[(String, Int)] = Array((Delhi,2000), (Kolkatta,7000))
```

Command took 0.65 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:05:47 on My Cluster lab 3

Cmd 12

```
1  val bc = city.filter(city => city._1.contains("i") && city._2<4000)
2  bc.collect
```

(1) Spark Jobs

```
bc: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[19] at filter at command-1164433993013378:1
res19: Array[(String, Int)] = Array((Delhi,2000))
```

Command took 0.60 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:09:41 on My Cluster lab 3

Cmd 13

```
1  city.filter(_._2>3000).filter(_._2<6000).collect
```

(1) Spark Jobs

```
res20: Array[(String, Int)] = Array((Mumbai,4000))
```

Command took 1.90 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:15:00 on My Cluster lab 3

Cmd 14

```
1  city.filter(x => x._2>3000 && x._2<6000).collect
```

(1) Spark Jobs

```
res21: Array[(String, Int)] = Array((Mumbai,4000))
```

Command took 0.38 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:15:35 on My Cluster lab 3

Cmd 15

```
1  // Generating Arraty of many values in a single call
2  val x = sc.parallelize( 1 to 1000)
3  x.collect
4
```

(1) Spark Jobs

```
x: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[23] at parallelize at command-1164433993013381:2
res22: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 3
52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85
5, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 1
9, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 1
3, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 2
7, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 2
1, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 3
5, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 3
9, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 3
3, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 4
7, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 4
1, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 5
5, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 5
9, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 6
3, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 6
7, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 7
1, 722, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 7
5, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 7
9, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 8
3, 854, 855, 856, 857, 858, 859, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876, 877, 878, 879, 8
```

Command took 0.43 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:16:15 on My Cluster lab 3

Cmd 16

```
1  //This line is sampling around 20% of the elements from the RDD x all of them will be unique(without replacement (false)).
2  var x20 = x.sample(false,0.2)
3  x20.collect
```

(1) Spark Jobs

```
x20: org.apache.spark.rdd.RDD[Int] = PartitionwiseSampledRDD[25] at sample at command-1164433993013382:2
res25: Array[Int] = Array(16, 19, 26, 38, 47, 48, 52, 55, 58, 60, 62, 74, 87, 97, 108, 109, 110, 111, 112, 115, 127, 129, 133, 137, 14
0, 239, 243, 249, 251, 254, 255, 267, 269, 273, 278, 289, 292, 294, 305, 308, 311, 321, 324, 326, 336, 337, 342, 348, 355, 364, 368, 3
```

```
0, 483, 487, 494, 513, 528, 530, 549, 557, 562, 569, 579, 595, 604, 605, 611, 615, 618, 622, 626, 628, 629, 638, 640, 651, 652, 653, 6
1, 752, 753, 755, 761, 764, 767, 769, 777, 780, 784, 797, 800, 810, 817, 819, 829, 831, 832, 833, 846, 850, 855, 860, 861, 864, 868, 8
6, 948, 951, 955, 971, 974, 978, 983, 987, 990, 993, 999)
```

Command took 0.78 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:18:15 on My Cluster lab 3

Cmd 17

```
1  x20.count()
```

(1) Spark Jobs

```
res27: Long = 185
```

Command took 1.18 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:19:00 on My Cluster lab 3

Cmd 18

```
1
2  //This line is sampling around 20% of the elements from the RDD x where some of them may be repeated(with replacement (true)).
3  var xtrue = x.sample(true,0.2)
4  xtrue.collect
```

(1) Spark Jobs

```
xtrue: org.apache.spark.rdd.RDD[Int] = PartitionwiseSampledRDD[26] at sample at command-1164433993013384:1
res28: Array[Int] = Array(1, 3, 16, 21, 24, 46, 66, 68, 72, 97, 97, 103, 106, 108, 117, 120, 120, 122, 144, 145, 153, 154, 156, 159, 1
2, 262, 268, 269, 271, 277, 281, 282, 284, 294, 295, 298, 301, 303, 303, 304, 311, 326, 333, 337, 337, 355, 360, 363, 377, 385, 394, 3
4, 476, 477, 479, 482, 485, 494, 496, 501, 506, 520, 524, 555, 561, 566, 566, 567, 572, 580, 584, 586, 588, 589, 589, 596, 598, 602, 6
2, 654, 658, 670, 674, 675, 676, 690, 694, 694, 698, 701, 710, 724, 724, 726, 735, 746, 747, 750, 750, 762, 770, 772, 778, 782, 784, 7
8, 884, 886, 891, 894, 895, 899, 901, 909, 912, 913, 917, 919, 924, 927, 933, 934, 949, 951, 952, 952, 960, 962, 965, 965, 976, 979, 9
```

Command took 0.47 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:19:46 on My Cluster lab 3

Cmd 19

```
1  xtrue.count()
```

(1) Spark Jobs

```
res29: Long = 207
```

Command took 0.45 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:22:50 on My Cluster lab 3

Cmd 20

```
1  val z = sc.parallelize( 8 to 15)
2  z.collect
```

(1) Spark Jobs

```
z: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[30] at parallelize at command-2423797189157357:1
res32: Array[Int] = Array(8, 9, 10, 11, 12, 13, 14, 15)
```

Command took 0.41 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:32:10 on My Cluster lab 3

Cmd 21

```
1  val y = sc.parallelize(1 to 5)
2  y.collect
```

(1) Spark Jobs

```
y: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[31] at parallelize at command-1164433993013386:1
res33: Array[Int] = Array(1, 2, 3, 4, 5)
```

Command took 0.39 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:32:13 on My Cluster lab 3

Cmd 22

```
1  var zy_union = z.union(y).collect
```

(1) Spark Jobs

```
xy_union: Array[Int] = Array(8, 9, 10, 11, 12, 13, 14, 15, 1, 2, 3, 4, 5)
```

Command took 1.18 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:32:32 on My Cluster lab 3

Cmd 23

```
1  var yz_union = y.union(z).collect
```

(1) Spark Jobs

```
yz_union: Array[Int] = Array(1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15)
```

Command took 0.36 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:33:06 on My Cluster lab 3

Cmd 24

```
1  var zy_intersection = z.intersection(y).collect
```

(1) Spark Jobs

```
zy_intersection: Array[Int] = Array()
```

Command took 1.13 seconds -- by om.pce21@sot.pdpu.ac.in at 31/01/2024, 10:33:47 on My Cluster lab 3

Lab4 - OM M PATEL - 21BCP094

```
1
```

```
state: scala.collection.immutable.Map[String,String] = Map(NY -> New York, CA -> California, FL -> Florida)
```

Command took 0.38 seconds -- by a user at 14/02/2024, 09:32:26 on unknown compute

Cmd 2

```
1
```

```
countries: scala.collection.immutable.Map[String,String] = Map(USA -> America, IN -> India)
```

Command took 0.48 seconds -- by a user at 14/02/2024, 09:32:27 on unknown compute

Cmd 3

```
1
```

```
brodState: org.apache.spark.broadcast.Broadcast[scala.collection.immutable.Map[String,String]] = Broadcast(3)
```

Command took 0.45 seconds -- by a user at 14/02/2024, 09:32:30 on unknown compute

Cmd 4

```
1
```

```
brodContries: org.apache.spark.broadcast.Broadcast[scala.collection.immutable.Map[String,String]] = Broadcast(4)
```

Command took 0.32 seconds -- by a user at 14/02/2024, 09:32:32 on unknown compute

Cmd 5

```
1  val state = Map(("NY", "New York"), ("CA", "California"), ("FL", "Florida"))
2  val countries = Map(("USA", "America"), ("IN", "India"))
3  val brodState = spark.sparkContext.broadcast(state)
4  val brodContries = spark.sparkContext.broadcast(countries)
5  val data = Seq(("A", "B", "IN", "CA"),
6                 ("D", "E", "USA", "CA"),
7                 ("G", "H", "IN", "NY"),
8                 ("J", "K", "USA", "FL"))
9
10 val columns = Seq("firstname", "lastname", "Country", "State")
11
12 import spark.sqlContext.implicits._
13
14 val df = data.toDF(columns:_*)
15 val df2 = df.map(row => {
16   val country = row.getString(2)
17   val state = row.getString(3)
18   val fullState = brodState.value.get(state).get
19   val fullCountry = brodContries.value.get(country).get
20   (row.getString(0), row.getString(1), fullCountry, fullState)
21 }).toDF(columns:_*)
```

```
state: scala.collection.immutable.Map[String,String] = Map(NY -> New York, CA -> California, FL -> Florida)
countries: scala.collection.immutable.Map[String,String] = Map(USA -> America, IN -> India)
brodState: org.apache.spark.broadcast.Broadcast[scala.collection.immutable.Map[String,String]] = Broadcast(5)
brodContries: org.apache.spark.broadcast.Broadcast[scala.collection.immutable.Map[String,String]] = Broadcast(6)
data: Seq[(String, String, String, String)] = List((A,B,IN,CA), (D,E,USA,CA), (G,H,IN,NY), (J,K,USA,FL))
columns: Seq[String] = List(firstname, lastname, Country, State)
import spark.sqlContext.implicits._
df: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string ... 2 more fields]
df2: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string ... 2 more fields]
```

Command took 4.38 seconds -- by a user at 14/02/2024, 09:35:08 on unknown compute

Cmd 6

```
1  df2.show(4)
```

```
+---------+--------+-------+----------+
|firstname|lastname|Country|     State|
+---------+--------+-------+----------+
|        A|       B|  India|California|
|        D|       E|America|California|
|        G|       H|  India|  New York|
|        J|       K|America|   Florida|
+---------+--------+-------+----------+
```

Command took 7.38 seconds -- by a user at 14/02/2024, 09:37:15 on unknown compute

Cmd 7

```
1  val longAcc = spark.sparkContext.longAccumulator("SUM")
```

```
longAcc: org.apache.spark.util.LongAccumulator = LongAccumulator(id: 111, name: Some(SUM), value: 0)
```

Command took 0.45 seconds -- by a user at 14/02/2024, 09:39:26 on unknown compute

Cmd 8

```
1  val rdd = spark.sparkContext.parallelize(Array(1, 2, 3, 4, 5))
2  rdd.foreach(x => longAcc.add(x))
3  rdd.collect
```

```
rdd: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[4] at parallelize at command-414821095629915:1
res3: Array[Int] = Array(1, 2, 3, 4, 5)
```

Command took 1.35 seconds -- by a user at 14/02/2024, 09:39:28 on unknown compute

Cmd 9

```
1  longAcc.value
```

```
res4: Long = 15
```

Command took 0.36 seconds -- by a user at 14/02/2024, 09:41:34 on unknown compute

Cmd 10

```
1  spark.sparkContext.setLogLevel("Error")
```

Command took 0.32 seconds -- by a user at 14/02/2024, 09:41:51 on unknown compute

Cmd 11

```
1  val inputRDD = spark.sparkContext.parallelize(List(("2", 1), ("B", 30), ("A", 20), ("B", 30), ("C", 40), ("B", 60)))
```

```
inputRDD: org.apache.spark.rdd.RDD[(String, Int)] = ParallelCollectionRDD[5] at parallelize at command-414821095629929:1
```

Command took 0.44 seconds -- by a user at 14/02/2024, 09:42:51 on unknown compute

Cmd 12

```
1  val listRDD = spark.sparkContext.parallelize(List(1, 2, 3, 4, 5, 2, 3))
```

```
listRDD: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[6] at parallelize at command-414821095629930:1
```

Command took 0.31 seconds -- by a user at 14/02/2024, 09:43:40 on unknown compute

Cmd 13

```
1  def param0 = (acc:Int, v:Int) => acc + v
2  def param1 = (acc1:Int, acc2:Int) => acc1  + acc2
3  println("Aggregate: " + listRDD.aggregate(0) (param0, param1))
```

```
Aggregate: 20
param0: (Int, Int) => Int
param1: (Int, Int) => Int
```

Command took 0.59 seconds -- by a user at 14/02/2024, 09:47:23 on unknown compute

Cmd 14

```
1  def param3 = (acc:Int, v:(String, Int)) => acc + v._2
2  def param2 = (acc1: Int, v2: Int) => acc1 + v2
3  println("Aggregate: " + inputRDD.aggregate(0) (param3, param2))
```

```
Aggregate: 181
param3: (Int, (String, Int)) => Int
param2: (Int, Int) => Int
```

Command took 0.66 seconds -- by a user at 14/02/2024, 09:48:55 on unknown compute

Cmd 15

```
1  val rdd2 = rdd.map(f => {
2    val country = f._3
3    val state = f._4
4    val fullCountry = brodContries.value.get(country).get
5    val fullState = brodState.value.get(state).get
6    (f._1, f._2, fullCountry, fullState)
7  })
```

```
command-414821095629916:4: error: not found: value brodContries
  val fullCountry = brodContries.value.get(country).get
                    ^
command-414821095629916:5: error: not found: value brodState
  val fullState = brodState.value.get(state).get
                  ^
```

Command took 0.14 seconds -- by a user at 14/02/2024, 09:23:58 on unknown compute

Cmd 16

```
1  println(rdd2.collect().mkString("\n"))
```

```
command-414821095629918:1: error: not found: value rdd2
println(rdd2.collect().mkString("\n"))
        ^
```

Command took 29.21 seconds -- by a user at 14/02/2024, 09:20:59 on unknown compute

```
1  val sqlContext = new org.apache.spark.sql.SQLContext(sc)
```

```
command-1530478450986451:1: warning: constructor SQLContext in class SQLContext is deprecated (since 2.0.0): Use SparkSession.builder instead
val sqlContext = new org.apache.spark.sql.SQLContext(sc)
                 ^
sqlContext: org.apache.spark.sql.SQLContext = org.apache.spark.sql.SQLContext@2c91ea81
```

Command took 7.66 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:27:49 on My Cluster Lab5

Cmd 2

```
1  val a = sc.parallelize(1 to 10)
```

```
a: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at command-1530478450986452:1
```

Command took 1.47 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:28:05 on My Cluster Lab5

Cmd 3

```
1  a.collect
```

(1) Spark Jobs

```
res0: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

Command took 2.66 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:28:15 on My Cluster Lab5

Cmd 4

```
1  val b = a.map(x=>(x, x + 1))
```

```
b: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[1] at map at command-1530478450986454:1
```

Command took 0.84 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:28:23 on My Cluster Lab5

Cmd 5

```
1  b.collect
```

(1) Spark Jobs

```
res1: Array[(Int, Int)] = Array((1,2), (2,3), (3,4), (4,5), (5,6), (6,7), (7,8), (8,9), (9,10), (10,11))
```

Command took 0.61 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:28:30 on My Cluster Lab5

Cmd 6

```
1  val df = b.toDF("First", "Second")
2  df.show
```

(3) Spark Jobs

df: org.apache.spark.sql.DataFrame = [First: integer, Second: integer]

```
+-----+------+
|First|Second|
+-----+------+
|    1|     2|
|    2|     3|
|    3|     4|
|    4|     5|
|    5|     6|
|    6|     7|
|    7|     8|
|    8|     9|
|    9|    10|
|   10|    11|
+-----+------+
```

```
df: org.apache.spark.sql.DataFrame = [First: int, Second: int]
```

Command took 15.17 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:28:39 on My Cluster Lab5

Cmd 7

```
1  val a = List(("Tom", 5), ("Jerry", 2), ("Donald", 7))
```

```
a: List[(String, Int)] = List((Tom,5), (Jerry,2), (Donald,7))
```

Command took 0.56 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:28:58 on My Cluster Lab5

Cmd 8

```
1  val df = a.toDF("Name", "Age")
```

df: org.apache.spark.sql.DataFrame = [Name: string, Age: integer]

```
df: org.apache.spark.sql.DataFrame = [Name: string, Age: int]
```

Command took 1.93 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:29:09 on My Cluster Lab5

Cmd 9

```
1  df.show
2
```

```
+------+---+
|  Name|Age|
+------+---+
|   Tom|  5|
| Jerry|  2|
|Donald|  7|
+------+---+
```

Command took 0.65 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:29:26 on My Cluster Lab5

Cmd 10

Cmd 10

```
1  val a = Seq(("Tom", 5), ("Jerry", 2), ("Donald", 7))
```

```
a: Seq[(String, Int)] = List((Tom,5), (Jerry,2), (Donald,7))
```

Command took 0.38 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:29:37 on My Cluster Lab5

Cmd 11

```
1  val df = a.toDF("Name", "Age")
```

df: org.apache.spark.sql.DataFrame = [Name: string, Age: integer]

```
df: org.apache.spark.sql.DataFrame = [Name: string, Age: int]
```

Command took 1.22 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:29:51 on My Cluster Lab5

Cmd 12

```
1  df.show
```

```
+------+---+
|  Name|Age|
+------+---+
|   Tom|  5|
| Jerry|  2|
|Donald|  7|
+------+---+
```

Command took 1.79 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:30:11 on My Cluster Lab5

Cmd 13

```
1  df.registerTempTable("Cartoon")
```

```
command-1530478450986463:1: warning: method registerTempTable in class Dataset is deprecated (since 2.0.0): Use createOrReplaceTempView(viewNa
df.registerTempTable("Cartoon")
   ^
```

Command took 0.57 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:30:23 on My Cluster Lab5

Cmd 14

```
1  df.createOrReplaceTempView("Cartoon")
```

Command took 0.72 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:30:29 on My Cluster Lab5

Cmd 15

```
1  sqlContext.sql("select * from Cartoon where Name='Tom'").show
```

```
+----+---+
|Name|Age|
+----+---+
| Tom|  5|
+----+---+
```

Command took 0.72 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:30:37 on My Cluster Lab5

Cmd 16

```
1  sqlContext.sql("select count(*) from Cartoon").show
```

(2) Spark Jobs

```
+--------+
|count(1)|
+--------+
|       3|
+--------+
```

Command took 4.54 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:30:44 on My Cluster Lab5

Cmd 17

Question: To create a JSON File, upload to DBFS,

printSchema() select query with all names filter and identify age > 23 groupBy Age count it and show it

Cmd 18

```
1  val df1 = spark.read.format("json").load("dbfs:/FileStore/shared_uploads/abhinav.sce21@sot.pdpu.ac.in/data-1.json")
```

```
AnalysisException: [PATH_NOT_FOUND] Path does not exist: dbfs:/FileStore/shared_uploads/abhinav.sce21@sot.pdpu.ac.in/data-1.json.
```

Command took 3.68 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:31:22 on My Cluster Lab5

Cmd 19

```
1  df1.show
```

```
command-1530478450986469:1: error: not found: value df1
df1.show
^
```

Command took 0.17 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:31:31 on My Cluster Lab5

Cmd 20

```
1  df1.printSchema()
```

Cmd 21

```
1  df1.select("Name", "Age").show()
```

```
command-1530478450986480:1: error: not found: value df1
df1.select("Name", "Age").show()
^
```

Command took 0.05 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:48:24 on My Cluster Lab5

Cmd 22

```
1  df1.createOrReplaceTempView("Employee")
```

Cmd 23

```
1  df1.filter(df1("age") > 23).show()
```

Cmd 24

```
1  df1.groupBy("age").count().show
```

Cmd 25

```
1  val rdda = sc.parallelize(1 to 1000)
2  rdda.collect()
3  val rddb = sc.parallelize(List("BMW", "Mercedes", "Toyota", "Audi"))
4  rddb.collect()
```

Cmd 26

```
1  rdda.partitions.length
```

Cmd 27

```
1  rddb.partitions.length
```

Cmd 28

```
1  val rdda = sc.parallelize(1 to 1000, 10)
2  rdda.collect()
3  rdda.partitions.length
```

Cmd 29

```
1  rdda.take(10)
```

Cmd 30

```
1  rdda.count()
```

Cmd 31

```
1  rdda.saveAsTextFile("dbfs:/FileStore/shared_uploads/abhinav.sce21@sot.pdpu.ac.in/random.txt")
```

Cmd 32

```
1  val rddRead = sc.textFile("dbfs:/FileStore/shared_uploads/abhinav.sce21@sot.pdpu.ac.in/random.txt")
```

Cmd 33

```
1  rddRead.count()
```

Cmd 34

```
1  rddRead.take(10)
```

```
command-1530478450986491:1: error: not found: value rddRead
rddRead.take(10)
^
```

Command took 0.06 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 09:50:55 on My Cluster Lab5

Lab5 - OM M PATEL - 21BCP094

Continuation of Lab 4_2

Cmd 2

```
1  val book = sc.textFile("dbfs:/FileStore/shared_uploads/om.pce21@sot.pdpu.ac.in/book.txt")
```

```
book: org.apache.spark.rdd.RDD[String] = dbfs:/FileStore/shared_uploads/om.pce21@sot.pdpu.ac.in/book.txt MapPartitionsRDD[76] at textFile at c
```

Command took 0.96 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:02:11 on My Cluster Lab5

Cmd 3

```
1  val a = book.collect()
```

(1) Spark Jobs

```
a: Array[String] = Array(1. Paradigms of Aritificial Intelligence Programming: Case Studies in Common Lisp, 2. Code: The Hidden Language of Co
Intelligence: A modern approach, 5. ON LISP, 6. ANSI Common LISP, 7. LISP in small pieces, 8. The little lisper, 9. The seasoned schemer)
```

Command took 0.83 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:02:13 on My Cluster Lab5

Cmd 4

```
1  val b = a.map(x=>(x,1))
```

```
b: Array[(String, Int)] = Array((1. Paradigms of Aritificial Intelligence Programming: Case Studies in Common Lisp,1), (2. Code: The Hidden La
s,1), (4. Aritificial Intelligence: A modern approach,1), (5. ON LISP,1), (6. ANSI Common LISP,1), (7. LISP in small pieces,1), (8. The little
```

Command took 0.59 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:02:15 on My Cluster Lab5

Cmd 5

```
1  val b = a.map(_=>(_,1))
```

```
command-1530478450986476:1: error: missing parameter type for expanded function ((x$2: <error>) => scala.Tuple2(x$2, 1))
val b = a.map(_=>(_,1))
                  ^
```

Command took 0.15 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:02:17 on My Cluster Lab5

Cmd 6

```
1  val result = book.flatMap(_.split(" ")).map((_, 1)).reduceByKey(_ + _).filter(_._2 > 1).sortBy(_._2, false).collect()
2
```

(2) Spark Jobs

```
result: Array[(String, Int)] = Array((LISP,3), (The,3), (in,2), (Common,2), (of,2), (Aritificial,2))
```

Command took 1.10 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:02:19 on My Cluster Lab5

Lab6 - OM M PATEL - 21BCP094

Data Preprocessing

Cmd 2

```
1
2  from pyspark.sql import SparkSession as ss
```

Command took 0.06 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:41 on My Cluster Lab5

Cmd 3

```
1  spark = ss.builder.appName('data_processing').getOrCreate()
```

Command took 0.12 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:44 on My Cluster Lab5

Cmd 4

```
1  df = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/shared_uploads/om.pce21@sot.pdpu.ac.in/sample_data.csv")
2
```

(1) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [ratings: string, age: string ... 3 more fields]

Command took 0.97 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:46 on My Cluster Lab5

Cmd 5

```
1  df.count()
2  df.columns
```

(2) Spark Jobs

```
Out[63]: ['ratings', 'age', 'experience', 'family', 'mobile']
```

Command took 0.33 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:48 on My Cluster Lab5

Cmd 6

```
1  print(df.count(), len(df.columns))
```

(2) Spark Jobs

```
47 5
```

Command took 0.41 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:49 on My Cluster Lab5

Cmd 7

```
1  df.printSchema()
```

```
root
 |-- ratings: string (nullable = true)
 |-- age: string (nullable = true)
 |-- experience: string (nullable = true)
 |-- family: string (nullable = true)
 |-- mobile: string (nullable = true)
```

Command took 0.05 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:52 on My Cluster Lab5

Cmd 8

```
1  df.show(5)
```

(1) Spark Jobs

```
+-------+---+----------+------+------------+
|ratings|age|experience|family|      mobile|
+-------+---+----------+------+------------+
|      3| 32|       9.0|     3|        Vivo|
|      4| 28|       8.5|     2|     Samsung|
|      5| 35|      10.2|     4|      iPhone|
|      2| 40|      12.0|     1|     OnePlus|
|      4| 27|       7.8|     3|Google Pixel|
+-------+---+----------+------+------------+
only showing top 5 rows
```

Command took 0.39 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:54 on My Cluster Lab5

Cmd 9

```
1  from pyspark.sql.types import StringType,DoubleType,IntegerType
```

Command took 0.07 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:58 on My Cluster Lab5

Cmd 10

```
1  df.withColumn('age_after_10_years',(df['age']+10)).show(10,False)
```

(1) Spark Jobs

```
+-------+---+----------+------+------------+------------------+
|ratings|age|experience|family|mobile      |age_after_10_years|
+-------+---+----------+------+------------+------------------+
|3      |32 |9.0       |3     |Vivo        |42.0              |
|4      |28 |8.5       |2     |Samsung     |38.0              |
|5      |35 |10.2      |4     |iPhone      |45.0              |
|2      |40 |12.0      |1     |OnePlus     |50.0              |
|4      |27 |7.8       |3     |Google Pixel|37.0              |
|3      |33 |9.5       |2     |OPPO        |43.0              |
|5      |29 |8.0       |4     |Huawei      |39.0              |
|4      |36 |10.8      |3     |Xiaomi      |46.0              |
|3      |31 |9.2       |2     |LG          |41.0              |
|4      |34 |9.7       |3     |Sony        |44.0              |
+-------+---+----------+------+------------+------------------+
only showing top 10 rows
```

Command took 0.47 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:20:59 on My Cluster Lab5

Cmd 11

```
#Filter the records
df.filter(df['mobile']=='Sony').show()
```

(1) Spark Jobs

```
+-------+---+----------+------+------+
|ratings|age|experience|family|mobile|
+-------+---+----------+------+------+
|      4| 34|       9.7|     3|  Sony|
|      2| 39|      11.5|     1|  Sony|
+-------+---+----------+------+------+
```

Command took 0.42 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:02 on My Cluster Lab5

Cmd 12

```
df.filter((df['mobile']=='Vivo') & (df['experience']>5)).show()
```

(1) Spark Jobs

```
+-------+---+----------+------+------+
|ratings|age|experience|family|mobile|
+-------+---+----------+------+------+
|      3| 32|       9.0|     3|  Vivo|
|      3| 31|       9.0|     2|  Vivo|
+-------+---+----------+------+------+
```

Command took 0.39 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:04 on My Cluster Lab5

Cmd 13

```
#filter multple conditions
df.filter(df['mobile']=='OnePlus').filter(df['experience']>10).show()
```

(1) Spark Jobs

```
+-------+---+----------+------+-------+
|ratings|age|experience|family| mobile|
+-------+---+----------+------+-------+
|      2| 40|      12.0|     1|OnePlus|
+-------+---+----------+------+-------+
```

Command took 0.48 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:06 on My Cluster Lab5

Cmd 14

```
#Distinct Values in a column
df.select('mobile').distinct().show()
```

(2) Spark Jobs

```
+----------+
|    mobile|
+----------+
|   Infinix|
|     Nokia|
|      Sony|
|   Alcatel|
|  Motorola|
|      OPPO|
|    Realme|
|    iPhone|
|    Huawei|
|    Xiaomi|
|      Asus|
|    Lenovo|
|   Samsung|
|       HTC|
|Blackberry|
|        LG|
|     Pixel|
|   OnePlus|
+----------+
```

Command took 0.80 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:08 on My Cluster Lab5

Cmd 15

```
df.select('age').distinct().show()
```

(2) Spark Jobs

```
+---+
|age|
+---+
| 29|
| 30|
| 34|
| 28|
| 35|
| 31|
| 27|
| 26|
| 40|
| 38|
| 33|
| 32|
| 36|
```

```
| 37|
| 39|
+---+
```

Command took 0.47 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:10 on My Cluster Lab5

Cmd 16

```
1  df.select('experience').distinct().show()
```

(2) Spark Jobs

```
+----------+
|experience|
+----------+
|       8.5|
|       8.2|
|       8.3|
|       9.2|
|      10.8|
|       7.5|
|       9.0|
|      12.0|
|      10.2|
|      10.0|
|       9.5|
|      11.5|
|       7.8|
|       9.8|
|       7.0|
|       8.7|
|       8.8|
|      11.0|
```

Command took 0.48 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:12 on My Cluster Lab5

Cmd 17

```
1  df.select('ratings').distinct().show()
```

(2) Spark Jobs

```
+-------+
|ratings|
+-------+
|      3|
|      5|
|      4|
|      2|
+-------+
```

Command took 0.50 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:14 on My Cluster Lab5

Cmd 18

```
1  df.select('family').distinct().show()
```

(2) Spark Jobs

```
+------+
|family|
+------+
|     3|
|     1|
|     4|
|     2|
+------+
```

Command took 0.46 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:16 on My Cluster Lab5

Cmd 19

```
1  #counting distinct items in a column
2  df.select('mobile').distinct().count()
```

(3) Spark Jobs

```
Out[77]: 25
```

Command took 1.80 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:18 on My Cluster Lab5

Cmd 20

```
1  df.select('age').distinct().count()
```

(3) Spark Jobs

```
Out[78]: 15
```

Command took 0.77 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:20 on My Cluster Lab5

Cmd 21

```
1  df.select('experience').distinct().count()
```

(3) Spark Jobs

```
Out[79]: 23
```

Command took 0.52 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:23 on My Cluster Lab5

Cmd 22

```
1  df.select('ratings').distinct().count()
```

(3) Spark Jobs

Out[80]: 4

Command took 0.56 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:24 on My Cluster Lab5

Cmd 23

```
df.select('family').distinct().count()
```

(3) Spark Jobs

Out[81]: 4

Command took 0.48 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:26 on My Cluster Lab5

Cmd 24

```
#GroupBy

df.groupBy('mobile').count().show(5,False)
```

(2) Spark Jobs

```
+--------+-----+
|mobile  |count|
+--------+-----+
|Infinix |2    |
|Nokia   |2    |
|Sony    |2    |
|Alcatel |2    |
|Motorola|2    |
+--------+-----+
only showing top 5 rows
```

Command took 1.11 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:28 on My Cluster Lab5

Cmd 25

```
df.groupBy('mobile').count().orderBy('count',ascending=False).show()
```

(2) Spark Jobs

```
+----------+-----+
|    mobile|count|
+----------+-----+
|   OnePlus|    3|
|   Infinix|    2|
|     Nokia|    2|
|      Sony|    2|
|   Alcatel|    2|
|  Motorola|    2|
|      OPPO|    2|
|    Realme|    2|
|    iPhone|    2|
|    Huawei|    2|
|    Xiaomi|    2|
|      Asus|    2|
|    Lenovo|    2|
|   Samsung|    2|
|       HTC|    2|
|Blackberry|    2|
|        LG|    2|
|     Pixel|    2|
+----------+-----+
```

Command took 0.67 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:31 on My Cluster Lab5

Cmd 26

```
df.groupBy('age').count().orderBy('count',ascending=False).show()
```

(2) Spark Jobs

```
+---+-----+
|age|count|
+---+-----+
| 29|    5|
| 31|    5|
| 30|    4|
| 34|    4|
| 37|    4|
| 28|    3|
| 35|    3|
| 27|    3|
| 33|    3|
| 32|    3|
| 36|    3|
| 26|    2|
| 38|    2|
| 39|    2|
| 40|    1|
+---+-----+
```

Command took 0.51 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:33 on My Cluster Lab5

Cmd 27

```
df.groupBy('experience').count().orderBy('count',ascending=False).show()
```

(2) Spark Jobs

```
+----------+-----+
|experience|count|
+----------+-----+
|       8.5|    5|
|       9.0|    5|
|      10.2|    3|
|       8.2|    2|
|       8.3|    2|
|       9.2|    2|
|       7.5|    2|
|      10.0|    2|
|      10.3|    2|
|       9.5|    2|
|      11.5|    2|
|       7.8|    2|
|       9.8|    2|
|       8.7|    2|
|       9.7|    2|
|       8.0|    2|
|      10.5|    2|
|      10.8|    1|
Command took 0.52 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:35 on My Cluster Lab5
```

Cmd 28

```
1  df.groupBy('ratings').count().orderBy('count',ascending=False).show()
```

(2) Spark Jobs

```
+-------+-----+
|ratings|count|
+-------+-----+
|      4|   15|
|      3|   13|
|      5|   12|
|      2|    7|
+-------+-----+

Command took 0.54 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:38 on My Cluster Lab5
```

Cmd 29

```
1  df.groupBy('family').count().orderBy('count',ascending=False).show()
```

(2) Spark Jobs

```
+------+-----+
|family|count|
+------+-----+
|     3|   15|
|     2|   13|
|     4|   12|
|     1|    7|
+------+-----+

Command took 0.65 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:21:39 on My Cluster Lab5
```

Cmd 30

```
1  #Value Counts
2
3  a = df.groupBy('mobile').mean()
4  a.show()
```

(2) Spark Jobs

a: pyspark.sql.dataframe.DataFrame = [mobile: string]

```
+----------+
|    mobile|
+----------+
|   Infinix|
|     Nokia|
|      Sony|
|   Alcatel|
|  Motorola|
|      OPPO|
|    Realme|
|    iPhone|
|    Huawei|
|    Xiaomi|
|      Asus|
|    Lenovo|
|   Samsung|
|       HTC|
|Blackberry|
|        LG|
|     Pixel|
|   OnePlus|
Command took 0.63 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:22:54 on My Cluster Lab5
```

Cmd 31

```
1  df.groupBy('mobile').sum().show()
```

(2) Spark Jobs

```
+----------+
|    mobile|
```

```
+----------+
|   Infinix|
|     Nokia|
|      Sony|
|   Alcatel|
|  Motorola|
|      OPPO|
|    Realme|
|    iPhone|
|    Huawei|
|    Xiaomi|
|      Asus|
|    Lenovo|
|   Samsung|
|       HTC|
|Blackberry|
|        LG|
|     Pixel|
|   OnePlus|
```

Command took 1.27 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:23:10 on My Cluster Lab5

Cmd 32

```
1  df.groupBy('mobile').min().show(5,False)
```

(2) Spark Jobs

```
+-------+
|mobile |
+-------+
|OPPO   |
|iPhone |
|Samsung|
|OnePlus|
|Vivo   |
+-------+
only showing top 5 rows
```

Command took 0.47 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:23:00 on My Cluster Lab5

Cmd 33

```
1
```

Cmd 34

```
1  #UDF (User Defined Functions)
2  from pyspark.sql.functions import udf
```

Command took 0.13 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:23:46 on My Cluster Lab5

Cmd 35

```
1  #Normal Function
2  def price_range(brand):
3      if brand in ['Samsung','iPhone']:
4          return 'High Price'
5      elif brand == 'Xiaomi':
6          return 'Mid Price'
7      else:
8          return 'Low Price'
```

Command took 0.13 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:26:06 on My Cluster Lab5

Cmd 36

```
1  #create udf using python function
2  brand_udf = udf(price_range,StringType())
3
4  #Applying UDF on data Frame
5  df.withColumn('price_range',brand_udf(df['mobile'])).show()
```

(1) Spark Jobs

```
+-------+---+----------+------+------------+-----------+
|ratings|age|experience|family|      mobile|price_range|
+-------+---+----------+------+------------+-----------+
|      3| 32|       9.0|     3|        Vivo|  Low Price|
|      4| 28|       8.5|     2|     Samsung| High Price|
|      5| 35|      10.2|     4|      iPhone| High Price|
|      2| 40|      12.0|     1|     OnePlus|  Low Price|
|      4| 27|       7.8|     3|Google Pixel|  Low Price|
|      3| 33|       9.5|     2|        OPPO|  Low Price|
|      5| 29|       8.0|     4|      Huawei|  Low Price|
|      4| 36|      10.8|     3|      Xiaomi|  Mid Price|
|      3| 31|       9.2|     2|          LG|  Low Price|
|      4| 34|       9.7|     3|        Sony|  Low Price|
|      5| 30|       8.3|     4|      Realme|  Low Price|
|      2| 38|      11.5|     2|        Asus|  Low Price|
|      3| 26|       7.0|     1|       Nokia|  Low Price|
|      4| 37|      10.0|     3|    Motorola|  Low Price|
|      5| 29|       8.5|     4|      Lenovo|  Low Price|
|      3| 32|       9.2|     2|         HTC|  Low Price|
|      4| 35|      10.5|     3|  Blackberry|  Low Price|
|      5| 31|       8.8|     4|         ZTE|  Low Price|
```

Command took 2.90 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:27:03 on My Cluster Lab5

Cmd 37

```
#Using Lambda Functions
class_age = lambda age: "experienced" if int(age) > 30 else "young"
age_udf = udf(class_age, StringType())

# Apply the UDF to create a new column 'class_age'
df = df.withColumn('age', df['age'].cast(IntegerType()))
df = df.withColumn('class_age', age_udf(df['age']))
df.show()

```

(1) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [ratings: string, age: integer ... 4 more fields]

```
+-------+---+----------+------+------------+-----------+
|ratings|age|experience|family|      mobile|  class_age|
+-------+---+----------+------+------------+-----------+
|      3| 32|       9.0|     3|        Vivo|experienced|
|      4| 28|       8.5|     2|     Samsung|      young|
|      5| 35|      10.2|     4|      iPhone|experienced|
|      2| 40|      12.0|     1|     OnePlus|experienced|
|      4| 27|       7.8|     3|Google Pixel|      young|
|      3| 33|       9.5|     2|        OPPO|experienced|
|      5| 29|       8.0|     4|      Huawei|      young|
|      4| 36|      10.8|     3|      Xiaomi|experienced|
|      3| 31|       9.2|     2|          LG|experienced|
|      4| 34|       9.7|     3|        Sony|experienced|
|      5| 30|       8.3|     4|      Realme|      young|
|      2| 38|      11.5|     2|        Asus|experienced|
|      3| 26|       7.0|     1|       Nokia|      young|
|      4| 37|      10.0|     3|    Motorola|experienced|
|      5| 29|       8.5|     4|      Lenovo|      young|
|      3| 32|       9.2|     2|         HTC|experienced|
|      4| 35|      10.5|     3|  Blackberry|experienced|
|      5| 31|       8.8|     4|         ZTE|experienced|
Command took 0.86 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:39:03 on My Cluster Lab5
```

Cmd 38

```
from pyspark.sql.functions import pandas_udf, PandasUDFType

```

Command took 0.03 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:40:51 on My Cluster Lab5

Cmd 39

```
#udf with two columns
def prod(rating,exp):
    try:
        rating = float(rating)
        exp = float(exp)
        return rating * exp
    except ValueError:
        return None
```

Command took 0.08 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:44:31 on My Cluster Lab5

Cmd 40

```
#create udf using python function
prod_udf = udf(prod, DoubleType())

```

Command took 0.05 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:44:34 on My Cluster Lab5

Cmd 41

```
df = df.withColumn('product', prod_udf(df['ratings'], df['experience']))
df.show()
```

(1) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [ratings: string, age: integer ... 5 more fields]

```
+-------+---+----------+------+------------+-----------+------------------+
|ratings|age|experience|family|      mobile|  class_age|           product|
+-------+---+----------+------+------------+-----------+------------------+
|      3| 32|       9.0|     3|        Vivo|experienced|              27.0|
|      4| 28|       8.5|     2|     Samsung|      young|              34.0|
|      5| 35|      10.2|     4|      iPhone|experienced|              51.0|
|      2| 40|      12.0|     1|     OnePlus|experienced|              24.0|
|      4| 27|       7.8|     3|Google Pixel|      young|              31.2|
|      3| 33|       9.5|     2|        OPPO|experienced|              28.5|
|      5| 29|       8.0|     4|      Huawei|      young|              40.0|
|      4| 36|      10.8|     3|      Xiaomi|experienced|              43.2|
|      3| 31|       9.2|     2|          LG|experienced|27.599999999999998|
|      4| 34|       9.7|     3|        Sony|experienced|              38.8|
|      5| 30|       8.3|     4|      Realme|      young|              41.5|
|      2| 38|      11.5|     2|        Asus|experienced|              23.0|
|      3| 26|       7.0|     1|       Nokia|      young|              21.0|
|      4| 37|      10.0|     3|    Motorola|experienced|              40.0|
|      5| 29|       8.5|     4|      Lenovo|      young|              42.5|
|      3| 32|       9.2|     2|         HTC|experienced|27.599999999999998|
|      4| 35|      10.5|     3|  Blackberry|experienced|              42.0|
|      5| 31|       8.8|     4|         ZTE|experienced|              44.0|
Command took 0.65 seconds -- by om.pce21@sot.pdpu.ac.in at 21/02/2024, 10:44:36 on My Cluster Lab5
```

```
1  # 21BCP094
2  import pandas as pd
3  import numpy as np
4  import matplotlib.pyplot as plt
```

Command took 3.45 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:27:09 on My Cluster

Cmd 2

```
1  # 21BCP094
2  from sklearn.datasets import fetch_california_housing
3  from sklearn.model_selection import train_test_split
4
5  # Fetch the data
6  data = fetch_california_housing(as_frame=True)
7  data
```

```
Out[3]: {'data':        MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
 0      8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
 1      8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
 2      7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
 3      5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
 4      3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85
 ...       ...       ...       ...        ...         ...       ...       ...
 20635  1.5603      25.0  5.045455   1.133333       845.0  2.560606     39.48
 20636  2.5568      18.0  6.114035   1.315789       356.0  3.122807     39.49
 20637  1.7000      17.0  5.205543   1.120092      1007.0  2.325635     39.43
 20638  1.8672      18.0  5.329513   1.171920       741.0  2.123209     39.43
 20639  2.3886      16.0  5.254717   1.162264      1387.0  2.616981     39.37

        Longitude
 0        -122.23
 1        -122.22
 2        -122.24
 3        -122.25
 4        -122.25
 ...          ...
 20635    -121.09
```

Command took 0.35 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:27:22 on My Cluster

Cmd 3

```
1  X = data.data
2  y = data.target
3
4  # Split the data into training and testing sets
5  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=19)
6
```

Command took 0.11 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:29:30 on My Cluster

Cmd 4

```
1
2  # Print the shapes of the train and test sets
3  print("Train set shapes:")
4  print("X_train :", X_train.shape)
5  print("y_train:", y_train.shape)
6  print("\n\n Test set shapes:")
7  print("X_test:", X_test.shape)
8  print("y_test:", y_test.shape)
```

```
Train set shapes:
X_train : (16512, 8)
y_train: (16512,)


 Test set shapes:
X_test: (4128, 8)
y_test: (4128,)
```

Command took 0.05 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:29:31 on My Cluster

Cmd 5

```
1  # 21BCP094
2  y.isna().sum()
```

```
Out[15]: 0
```

Command took 0.09 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:29:32 on My Cluster

Cmd 6

```
1  # 21BCP094
2  X.isna().sum()
```

```
Out[16]: MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
dtype: int64
```

Command took 0.17 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:29:33 on My Cluster

Cmd 7

Lab7 - OM M PATEL - 21BCP094

```
# 21BCP094
from sklearn.linear_model import LinearRegression
m1 = LinearRegression()
m1.fit(X_train, y_train)
```

```
Out[19]: LinearRegression()
```

Command took 0.11 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:29:58 on My Cluster

Cmd 8

```
# 21BCP094
#b
print(m1.intercept_)
```

```
-37.28532899875165
```

Command took 0.07 seconds -- by a user at 05/04/2024, 23:21:08 on unknown compute

Cmd 9

```
# 21BCP094
#x
print(m1.coef_)
```

```
[ 4.40834835e-01  9.58577759e-03 -1.16109449e-01  7.45916246e-01
 -2.76667785e-06 -4.55818430e-03 -4.22322783e-01 -4.36138307e-01]
```

Command took 0.12 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:30:14 on My Cluster

Cmd 10

```
# 21BCP094
y_pred = m1.predict(X_test)
print(y_pred[:5])
```

```
[2.68402677 0.69960477 2.11086189 2.22278729 4.17861913]
```

Command took 0.09 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:30:27 on My Cluster

Cmd 11

```
# 21BCP094
from sklearn import metrics
import numpy as np

r2_score = metrics.r2_score(y_test, y_pred)
print("MAE: ", metrics.mean_absolute_error(y_test, y_pred))
print("MSE: ", metrics.mean_squared_error(y_test, y_pred))
print("RMSE: ", np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
print("R2 Score:", r2_score)
```

```
MAE:  0.532539783180095
MSE:  0.530318019930561
RMSE:  0.7282293731583209
R2 Score: 0.6001714745777522
```

Command took 0.11 seconds -- by om.pce21@sot.pdpu.ac.in at 09/04/2024, 10:31:35 on My Cluster

Lab8 - OM M PATEL - 21BCP094

```
import pandas as pd
import io

column = ['CRIM', 'ZN', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX', 'PTRATIL','B', 'LSTAT', 'MEDV']
csv_string = dbutils.fs.head("dbfs:/FileStore/shared_uploads/om.pce21@sot.pdpu.ac.in/housing-2.csv")
csv_string
```

```
Out[55]: ' 0.00632  18.00   2.310  0  0.5380  6.5750  65.20  4.0900   1  296.0  15.30 396.90   4.98  24.00\n 0.02731   0.00   7.070  0  0.4690  6.4216
0   7.070  0  0.4690  7.1850  61.10  4.9671   2  242.0  17.80 392.83   4.03  34.70\n 0.03237   0.00   2.180  0  0.4580  6.9980  45.80  6.0622   3  222
1470  54.20  6.0622   3  222.0  18.70 396.90   5.33  36.20\n 0.02985   0.00   2.180  0  0.4580  6.4300  58.70  6.0622   3  222.0  18.70 394.12   5.21
311.0  15.20 395.60  12.43  22.90\n 0.14455  12.50   7.870  0  0.5240  6.1720  96.10  5.9505   5  311.0  15.20 396.90  19.15  27.10\n 0.21124  12.50
93  16.50\n 0.17004  12.50   7.870  0  0.5240  6.0040  85.90  6.5921   5  311.0  15.20 386.71  17.10  18.90\n 0.22489  12.50   7.870  0  0.5240  6.377
50   7.870  0  0.5240  6.0090  82.90  6.2267   5  311.0  15.20 396.90  13.27  18.90\n 0.09378  12.50   7.870  0  0.5240  5.8890  39.00  5.4509   5  31
5.9490  61.80  4.7075   4  307.0  21.00 396.90   8.26  20.40\n 0.63796   0.00   8.140  0  0.5380  6.0960  84.50  4.4619   4  307.0  21.00 380.02  10.2
4  307.0  21.00 395.62   8.47  19.90\n 1.05393   0.00   8.140  0  0.5380  5.9350  29.30  4.4986   4  307.0  21.00 386.85   6.58  23.10\n 0.78420   0.0
14.67  17.50\n 0.80271   0.00   8.140  0  0.5380  5.4560  36.60  3.7965   4  307.0  21.00 288.99  11.69  20.20\n 0.72580   0.00   8.140  0  0.5380  5.
0.00   8.140  0  0.5380  5.5700  98.10  3.7979   4  307.0  21.00 376.57  21.02  13.60\n 0.85204   0.00   8.140  0  0.5380  5.9650  89.20  4.0123   4
6.1420  91.70  3.9769   4  307.0  21.00 396.90  18.72  15.20\n 0.98843   0.00   8.140  0  0.5380  5.8130 100.00  4.0952   4  307.0  21.00 394.54  19.8
4  307.0  21.00 394.33  16.30  15.60\n 0.84054   0.00   8.140  0  0.5380  5.5990  85.70  4.4546   4  307.0  21.00 303.42  16.51  13.90\n 0.67191   0.0
14.81  16.60\n 0.95577   0.00   8.140  0  0.5380  6.0470  88.80  4.4534   4  307.0  21.00 306.38  17.28  14.80\n 0.77299   0.00   8.140  0  0.5380  6.
0.00   8.140  0  0.5380  6.6740  87.30  4.2390   4  307.0  21.00 380.23  11.98  21.00\n 1.13081   0.00   8.140  0  0.5380  5.7130  94.10  4.2330   4
6.0720 100.00  4.1750   4  307.0  21.00 376.73  13.04  14.50\n 1.38799   0.00   8.140  0  0.5380  5.9500  82.00  3.9900   4  307.0  21.00 232.60  27.7
4  307.0  21.00 358.77  18.35  13.10\n 1.61282   0.00   8.140  0  0.5380  6.0960  96.90  3.7598   4  307.0  21.00 248.31  20.34  13.50\n 0.06417   0.0
9.68  18.90\n 0.09744   0.00   5.960  0  0.4990  5.8410  61.40  3.3779   5  279.0  19.20 377.56  11.41  20.00\n 0.08014   0.00   5.960  0  0.4990  5.8
0.00   5.960  0  0.4990  5.9660  30.20  3.8473   5  279.0  19.20 393.43  10.13  24.70\n 0.02763  75.00   2.950  0  0.4280  6.5950  21.80  5.4011   3
7.0240  15.80  5.4011   3  252.0  18.30 395.62   1.98  34.90\n 0.12744   0.00   6.910  0  0.4480  6.7700   2.90  5.7209   3  233.0  17.90 385.41   4.8
3  233.0  17.90 383.37   5.81  25.30\n 0.15936   0.00   6.910  0  0.4480  6.2110   6.50  5.7209   3  233.0  17.90 394.46   7.44  24.70\n 0.12269   0.0
9.55  21.20\n 0.17142   0.00   6.910  0  0.4480  5.6820  33.80  5.1004   3  233.0  17.90 396.90  10.21  19.30\n 0.18836   0.00   6.910  0  0.4480  5.7
Command took 0.42 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:32 on Lab8
```

Cmd 2

```
data = pd.read_csv(io.StringIO(csv_string), header=None, delimiter="\s+", names=column)
data
```

Command took 0.09 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:32 on Lab8

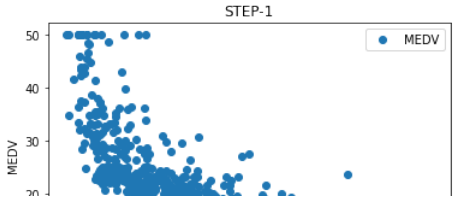
Cmd 3

```
df_1 = data.loc[:, ['LSTAT', 'MEDV']]
df_1.head()
```

Command took 0.13 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:33 on Lab8

Cmd 4

```
import matplotlib.pyplot as plt
df_1.plot(x='LSTAT', y='MEDV', style='o')
plt.xlabel('LSTAT')
plt.ylabel('MEDV')
plt.title('STEP-1')
plt.show()
```



: 03/04/2024, 10:08:34 on Lab8

Cmd 5

```
X = pd.DataFrame(df_1['LSTAT'])
y = pd.DataFrame(df_1['MEDV'])

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state = 1)

print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(354, 1)
(152, 1)
(354, 1)
(152, 1)
```

Command took 0.10 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:35 on Lab8

Cmd 6

```
from sklearn.linear_model import LinearRegression
m1 = LinearRegression()
m1.fit(X_train,y_train)
```

Out[60]: LinearRegression()

Command took 0.12 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:37 on Lab8

Cmd 7

```
print(m1.intercept_)
```

```
[34.22183685]
```

Command took 0.06 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:40 on Lab8

Cmd 8

```
print(m1.coef_)
```

```
[[-0.9166916]]
```

Command took 0.12 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:40 on Lab8

Cmd 9

```
m1_y_pred = m1.predict(X_test)
print(m1_y_pred)
```

```
[[27.31914909]
 [27.63999115]
 [16.98803475]
 [26.79663488]
 [24.88074943]
 [24.02822625]
 [29.91338632]
 [22.26817837]
 [17.79472336]
 [26.14578384]
 [27.12664386]
 [29.99588857]
 [21.74566416]
 [24.83491485]
 [23.47821128]
 [23.10236773]
 [12.91792404]
 [29.97755474]
 [27.41081825]
 [ 7.15193387]
 [23.67988344]]
```

Command took 0.06 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:43 on Lab8

Cmd 10

```
from sklearn import metrics
import numpy as np
print("MAE: ", metrics.mean_absolute_error(y_test,m1_y_pred))
print("MSE: ", metrics.mean_squared_error(y_test,m1_y_pred))
print("RMSE: ", metrics.mean_squared_error(y_test, m1_y_pred, squared=False))
```

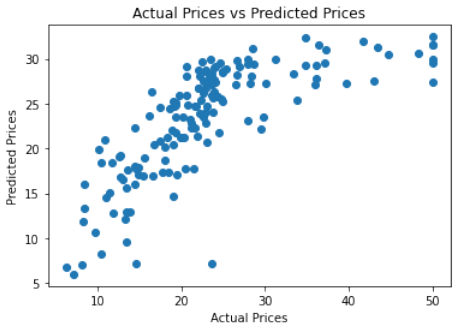
```
MAE:  4.815209094507989
MSE:  42.62024347153971
RMSE:  6.528418144661057
```

Command took 0.15 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:45 on Lab8

Cmd 11

```
plt.scatter(y_test, m1_y_pred)
```

```
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual Prices vs Predicted Prices")
plt.show()
```



Command took 0.27 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:47 on Lab8

Cmd 12

```
def plot_regression_line(X,y,b):
    plt.scatter(X,y, color = 'm', marker = 'o' , s =30)
    plt.plot(X,m1_y_pred,color = 'g')
    plt.xlabel('X')
    plt.ylabel('y')
    plt.show()
```

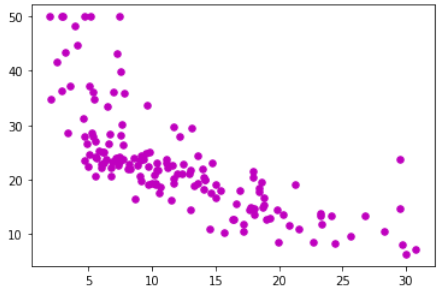
Command took 0.15 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:53 on Lab8

Cmd 13

```
import numpy as np
import matplotlib.pyplot as plt

def estimate_coef(X,y):
    return(m1.intercept_,m1.coef_)


b = estimate_coef(X_test,y_test)
print("Estimated_coefficient:\nb_0 = {} \
    \nb_1 = {}".format(m1.intercept_,m1.coef_))
plot_regression_line(X_test,y_test,b)
```

```
Estimated_coefficient:
b_0 = [34.22183685]
b_1 = [[-0.9166916]]
Unexpected exception formatting exception. Falling back to standard exception
```

```
Traceback (most recent call last):
  File "/databricks/python/lib/python3.9/site-packages/pandas/core/indexes/base.py", line 3621, in get_loc
    return self._engine.get_loc(casted_key)
  File "pandas/_libs/index.pyx", line 136, in pandas._libs.index.IndexEngine.get_loc
  File "pandas/_libs/index.pyx", line 142, in pandas._libs.index.IndexEngine.get_loc
TypeError: '(slice(None, None, None), None)' is an invalid key

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/databricks/python/lib/python3.9/site-packages/IPython/core/interactiveshell.py", line 3378, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "<command-1832767187777456>", line 11, in <module>
    plot_regression_line(X_test,y_test,b)
  File "<command-1832767187777453>", line 3, in plot_regression_line
    plt.plot(X,m1_y_pred,color = 'g')
  File "/databricks/python/lib/python3.9/site-packages/matplotlib/pyplot.py", line 2757, in plot
    return gca().plot(
  File "/databricks/python/lib/python3.9/site-packages/matplotlib/axes/_axes.py", line 1632, in plot
    lines = [*self._get_lines(*args, data=data, **kwargs)]
  File "/databricks/python/lib/python3.9/site-packages/matplotlib/axes/_base.py", line 312, in __call__
```




```
  InvalidIndexError: (slice(None, None, None), None)
```

Command took 0.40 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:08:56 on Lab8

Cmd 14

```
## Multivariate Linear Regression
X_2 = data.drop('MEDV', axis=1)
```

```
3   y_2 = data['MEDV']
```

Command took 0.04 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:09:14 on Lab8

Cmd 15

```
1   X_2train, X_2test, y_2train, y_2test = train_test_split(X_2, y_2, test_size=0.3, random_state=19)
2
```

Command took 0.15 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:09:29 on Lab8

Cmd 16

```
1   m2 = LinearRegression()
2   m2.fit(X_2train, y_2train)
3   m2_y_pred = m2.predict(X_2test)
4   m2_y_pred
5
```

```
Out[70]: array([23.21372914, 17.87577243, 15.77371112, 24.53621366, 20.751189  ,
       18.03026337, 20.91121742, 31.12725576, 20.71649629, 13.93406891,
       20.8325338 , 28.17944833, 17.36718485, 14.18145744, 27.50211423,
       19.77448389, 21.98284261, 22.32156203, 18.88414078, 32.86762607,
       13.34739954, 20.88782359, 32.031498  , 31.23883329, 20.26450697,
        9.06334217, 17.33228818, 17.66576791, 16.42829643, 27.45157611,
       38.64372366, 23.34106991, 17.26738241, 24.36216727, 24.97473948,
       33.03031697, 23.02580961, 23.24783313, 16.57178058, 17.44731533,
       20.5426661 , 23.67849039, 13.0806751 , 30.87394414, 21.8442593 ,
       21.98712355, 26.62222064, 22.20067076, 20.3535696 , 14.72855052,
       22.58841471, 44.3890252 , 14.74314929, 36.94830952, 34.4891329 ,
       15.66264845, 21.88805057, 17.5580889 , 20.67505316, 10.15590883,
       15.50784366, 21.28606992, 14.04872144, 14.32821946, 17.77130569,
       15.86821935,  9.83070988, 29.28447385, 25.9498077 , 20.07011094,
       23.23412684, 23.585168  , 21.64385664, 17.12893386, 16.58371562,
       26.20837525, 22.00752462, 18.75839078, 21.92504804, 24.47903772,
       28.05260834, 28.5644098 , 25.71345199, 30.74658407, 18.10185434,
       23.6120034 , 22.23431279, 24.40276682, 34.4387266 , 16.28560945,
       30.22365756, 31.83055935, 12.3179448 , 27.75606606, 33.9901935 ,
       35.79724853, 17.12239344, 22.30879027, 18.48378808, 34.98846395,
       17.63811843, 17.55239905, 24.43333221, 40.84117991,  8.78333085,
```

Command took 0.08 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:09:59 on Lab8

Cmd 17

```
1   print("Intercept:", m2.intercept_)
2   print("Coefficients:", m2.coef_)
```

```
Intercept: 37.079701809560405
Coefficients: [ 5.81283179e-02 -1.44178703e-02  3.14267031e+00 -1.46241729e+01
  3.17185366e+00 -1.48911967e-02 -1.52304609e+00  2.05864635e-01
 -1.15454709e-02 -8.12378178e-01  9.36564796e-03 -4.92767238e-01]
```

Command took 0.13 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:10:03 on Lab8

Cmd 18

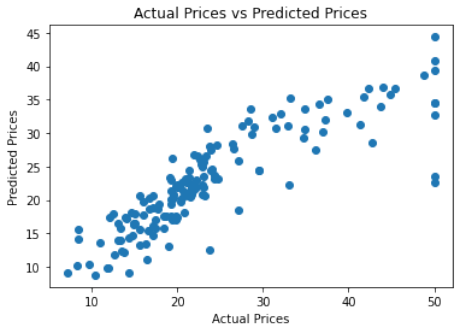
```
1   print("MAE:", metrics.mean_absolute_error(y_2test, m2_y_pred))
2   print("MSE:", metrics.mean_squared_error(y_2test, m2_y_pred))
3   print("RMSE:", np.sqrt(metrics.mean_squared_error(y_2test, m2_y_pred)))
4
```

```
MAE: 3.658285405825878
MSE: 31.23457178671033
RMSE: 5.5887898320397
```

Command took 0.06 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:10:21 on Lab8

Cmd 19

```
1   plt.scatter(y_2test, m2_y_pred)
2   plt.xlabel("Actual Prices")
3   plt.ylabel("Predicted Prices")
4   plt.title("Actual Prices vs Predicted Prices")
5   plt.show()
6
```



Command took 0.29 seconds -- by om.pce21@sot.pdpu.ac.in at 03/04/2024, 10:10:35 on Lab8