IBM Developer
SKILLS NETWORK

(https://cognitiveclass.ai)

This notebook is designed to run in a IBM Watson Studio default runtime (NOT the Watson Studio Apache Spark Runtime as the default runtime with 1 vCPU is free of charge). Therefore, we install Apache Spark in local mode for test purposes only. Please don't use it in production.

In case you are facing issues, please read the following two documents first:

https://github.com/IBM/skillsnetwork/wiki/Environment-Setup (https://github.com/IBM/skillsnetwork/wiki/Environment-Setup)

https://github.com/IBM/skillsnetwork/wiki/FAQ (https://github.com/IBM/skillsnetwork/wiki/FAQ)

Then, please feel free to ask:

https://coursera.org/learn/machine-learning-big-data-apache-spark/discussions/all (https://coursera.org/learn/machine-learning-big-data-apache-spark/discussions/all?cm_mmc=Email_Newsletter-_-Developer_Ed%2BTech-_-WW_WW-_-SkillsNetwork-Courses-IBMDeveloperSkillsNetwork-ML0201EN-SkillsNetwork-20647446&cm_mmca1=000026UJ&cm_mmca2=10006555&cm_mmca3=M12345678&cvosrc=email.Newslette

Please make sure to follow the guidelines before asking a question:

https://github.com/IBM/skillsnetwork/wiki/FAQ#im-feeling-lost-and-confused-please-help-me (https://github.com/IBM/skillsnetwork/wiki/FAQ#im-feeling-lost-and-confused-please-help-me)

If running outside Watson Studio, this should work as well. In case you are running in an Apache Spark context outside Watson Studio, please remove the Apache Spark setup in the first notebook cells.

In [1]:

```
from IPython.display import Markdown, display
def printmd(string):
    display(Markdown('# <span style="color:red">'+string+'</span>'))


if ('sc' in locals() or 'sc' in globals()):
    printmd('<<<<<!!!!! It seems that you are running in a IBM Watson Studio Apache Spa
rk Notebook. Please run it in an IBM Watson Studio Default Runtime (without Apache Spar
k) !!!!!>>>>>')
```

In [2]:

```
!pip install pyspark==2.4.5
```

```
Collecting pyspark==2.4.5
  Downloading pyspark-2.4.5.tar.gz (217.8 MB)
     |████████████████████████████████| 217.8 MB 12 kB/s s eta 0:00:01
Collecting py4j==0.10.7
  Downloading py4j-0.10.7-py2.py3-none-any.whl (197 kB)
     |████████████████████████████████| 197 kB 60.7 MB/s eta 0:00:01
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-2.4.5-py2.py3-none-any.whl s
ize=218257927 sha256=ad7ea16b94b4b427959a6d9a5aeef9d4ca07e956b19b79ba0c90f
976f776be17
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/01/c0/03/1c241c9c482b
647d4d99412a98a5c7f87472728ad41ae55e1e
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.7 pyspark-2.4.5
```

In [3]:

```
try:
    from pyspark import SparkContext, SparkConf
    from pyspark.sql import SparkSession
except ImportError as e:
    printmd('<<<<<!!!!! Please restart your kernel after installing Apache Spark !!!!!>
>>>>')
```

In [4]:

```
sc = SparkContext.getOrCreate(SparkConf().setMaster("local[*]"))

spark = SparkSession \
    .builder \
    .getOrCreate()
```

Welcome to exercise two of “Apache Spark for Scalable Machine Learning on BigData”. In this exercise you’ll apply the basics of functional and parallel programming.

Again, please use the following two links for your reference:
https://spark.apache.org/docs/latest/api/python/pyspark.html#pyspark.RDD
(https://spark.apache.org/docs/latest/api/python/pyspark.html#pyspark.RDD?cm_mmc=Email_Newsletter-_-
Developer_Ed%2BTech-_-WW_WW-_-SkillsNetwork-Courses-IBMDeveloperSkillsNetwork-ML0201EN-
SkillsNetwork-
20647446&cm_mmca1=000026UJ&cm_mmca2=10006555&cm_mmca3=M12345678&cvosrc=email.Newslette
_-Developer_Ed%2BTech-_-WW_WW-_-SkillsNetwork-Courses-IBMDeveloperSkillsNetwork-ML0201EN-
SkillsNetwork-
20647446&cm_mmca1=000026UJ&cm_mmca2=10006555&cm_mmca3=M12345678&cvosrc=email.Newslette
https://spark.apache.org/docs/latest/rdd-programming-guide.html (https://spark.apache.org/docs/latest/rdd-
programming-guide.html?cm_mmc=Email_Newsletter-_-Developer_Ed%2BTech-_-WW_WW-_-
SkillsNetwork-Courses-IBMDeveloperSkillsNetwork-ML0201EN-SkillsNetwork-
20647446&cm_mmca1=000026UJ&cm_mmca2=10006555&cm_mmca3=M12345678&cvosrc=email.Newslette
_-Developer_Ed%2BTech-_-WW_WW-_-SkillsNetwork-Courses-IBMDeveloperSkillsNetwork-ML0201EN-
SkillsNetwork-
20647446&cm_mmca1=000026UJ&cm_mmca2=10006555&cm_mmca3=M12345678&cvosrc=email.Newslette

Let’s actually create a python function which decides whether a value is greater than 50 (True) or not (False).

In [16]:

```
def gt50(i):
    if i > 50:
        return True
    else:
        return False
```

In [17]:

```
print(gt50(4))
print(gt50(51))
```

```
False
True
```

Let’s simplify this function

In [18]:

```
def gt50(i):
    return i > 50
```

In [19]:

```
print(gt50(4))
print(gt50(51))
```

```
False
True
```

Now let’s use the lambda notation to define the function.

In [20]:

```
#defined with functional programming
gt50 = lambda i: i > 50
```

In [21]:

```
print(gt50(4))
print(gt50(51))
```

```
False
True
```

In [22]:

```
#let's shuffle our list to make it a bit more interesting
#scale for so much values and parallelize it

from random import shuffle
l = list(range(100))
shuffle(l)
rdd = sc.parallelize(l)
```

Let’s filter values from our list which are equals or less than 50 by applying our “gt50” function to the list using the “filter” function. Note that by calling the “collect” function, all elements are returned to the Apache Spark Driver. This is not a good idea for BigData, please use “.sample(10,0.1).collect()” or “take(n)” instead.

In [23]:

```
rdd.filter(gt50).collect()
```

Out[23]:

```
[97,
 60,
 99,
 98,
 73,
 61,
 80,
 79,
 72,
 74,
 57,
 66,
 83,
 93,
 76,
 62,
 81,
 84,
 52,
 96,
 71,
 95,
 90,
 56,
 65,
 51,
 58,
 67,
 77,
 59,
 70,
 94,
 64,
 68,
 63,
 85,
 87,
 75,
 54,
 78,
 86,
 82,
 92,
 55,
 88,
 69,
 91,
 89,
 53]
```

We can also use the lambda function directly.

In [24]:

```
rdd.filter(lambda i: i > 50).collect()
```

Out[24]:

```
[97,
 60,
 99,
 98,
 73,
 61,
 80,
 79,
 72,
 74,
 57,
 66,
 83,
 93,
 76,
 62,
 81,
 84,
 52,
 96,
 71,
 95,
 90,
 56,
 65,
 51,
 58,
 67,
 77,
 59,
 70,
 94,
 64,
 68,
 63,
 85,
 87,
 75,
 54,
 78,
 86,
 82,
 92,
 55,
 88,
 69,
 91,
 89,
 53]
```

Let’s consider the same list of integers. Now we want to compute the sum for elements in that list which are greater than 50 but less than 75. Please implement the missing parts.

In [25]:

```
rdd.filter(lambda x: x > 50).filter(lambda x: x < 75).collect()
```

Out[25]:

```
[60,
 73,
 61,
 72,
 74,
 57,
 66,
 62,
 52,
 71,
 56,
 65,
 51,
 58,
 67,
 59,
 70,
 64,
 68,
 63,
 54,
 55,
 69,
 53]
```

In [26]:

```
rdd.filter(lambda x: x > 50).filter(lambda x: x < 75).reduce(lambda a, b: a + b)
```

Out[26]:

```
1500
```

You should see "1500" as answer. Now we want to know the sum of all elements. Please again, have a look at the API documentation and complete the code below in order to get the sum.

Thank you for completing this lab!

This notebook was created by Romeo Kienzler (https://linkedin.com/in/romeo-kienzler-089b4557) I hope you found this lab interesting and educational. Feel free to contact me if you have any questions!

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-09-29	2.0	Srishti	Migrated Lab to Markdown and added to course repo in GitLab

In [ ]: