Name-Istiaque Mannafee Shaikat
ID-303527

spark_1a

June 30, 2019

```
In [3]: from pyspark.sql import SparkSession
        from pyspark import SparkContext, SparkConf
        conf = SparkConf().setAppName("Spark Count")
        sc = SparkContext.getOrCreate(conf=conf)

        spark = SparkSession \
            .builder \
            .appName("Python Spark create RDD example") \
            .config("spark.some.config.option", "some-value") \
            .getOrCreate()
```

1 Creating two list a and b as tuple

```
In [5]: a = [("spark",), ("rdd",), ("python",), ("context",), ("create",), ("class",)]
        b = [("operation",), ("apache",), ("scala",), ("lambda",),("parallel",),("partition",)]
```

2 Creating spark dataframe and giving alias name as ta and tb

```
In [10]: tba = spark.createDataFrame(a,['col_ta',])
         tbb=spark.createDataFrame((b),['col_tb',])
         ta = tba.alias('ta')
         tb = tbb.alias('tb')
         ta.show()
         tb.show()
```

```
+-------+
| col_ta|
+-------+
|  spark|
|    rdd|
| python|
|context|
| create|
|  class|
+-------+
```

```
+---------+
|   col_tb|
+---------+
|operation|
|   apache|
|    scala|
|   lambda|
| parallel|
|partition|
+---------+
```

3 Joining the table using right outer join

- comparing the data of ta and tb to create right outer join

```
In [11]: right_outer_join = ta.join(tb, ta.col_ta == tb.col_tb,how='right_outer').show()
```

```
+------+---------+
|col_ta|   col_tb|
+------+---------+
|  null|operation|
|  null|   lambda|
|  null|partition|
|  null| parallel|
|  null|    scala|
|  null|   apache|
+------+---------+
```

4 Joining the table using Full outer join

comparing the data of ta and tb to create full outer join

```
In [12]: full_outer_join = ta.join(tb, ta.col_ta == tb.col_tb,how='full_outer').show()
```

```
+-------+---------+
| col_ta|   col_tb|
+-------+---------+
|   null|operation|
|   null|   lambda|
|context|     null|
|   null|partition|
| create|     null|
|    rdd|     null|
```

```
|   null| parallel|
|   null|    scala|
|   null|   apache|
|  spark|     null|
|  class|     null|
| python|     null|
+-------+---------+
```

5 Counting 's' using map and reduce

The data is parallelized then flatmap is used to map the data and reducebykey function is used

```
In [15]: #count "s" using map reduce
         char_counts_a = sc.parallelize(a).flatMap(lambda each: each[0].count('s')).map(lambda
         lis_a = char_counts_a.show()
         char_counts_b = sc.parallelize(b).flatMap(lambda each: each[0].count('s')).map(lambda
         lis_b = char_counts_b.show()

         print("For a the occurance of s is :",char_counts_a)
         print("For a the occurance of s is :",char_counts_b)
         print("For a the occurance of s is :",lis_a,'****',"For b the occurance of s is :",lis
```

```
For a the occurance of s is : PythonRDD[99] at RDD at PythonRDD.scala:53
For a the occurance of s is : PythonRDD[105] at RDD at PythonRDD.scala:53
```

For a the occurance of s is : 3**** For b the occurance of s is : 1

6 Counting 'S' using using aggregate function

The data is parallelized then agregated using lambda function to count 's' for both a and b

```
In [16]: #count using aggregate fuction
         count_a_aggregate=sc.parallelize(a).aggregate(0, lambda i, x: i + x[0].count('s'), lam
         print("For a the occurance of s is :",count_a_aggregate)
         count_b_aggregate=sc.parallelize(b).aggregate(0, lambda i, x: i + x[0].count('s'), lam
         print("For b the occurance of s is :",count_b_aggregate)
```

```
For a the occurance of s is : 3
For b the occurance of s is : 1
```

```
In [ ]:
```

spark_1b

June 30, 2019

```
In [23]: from pyspark.sql import SparkSession
         from pyspark import SparkContext, SparkConf
         import pyspark.sql.functions as fn
         import matplotlib.pyplot as plt
         from pyspark_dist_explore import hist

         conf = SparkConf().setAppName("Spark Count")
         sc = SparkContext.getOrCreate(conf=conf)

         spark = SparkSession \
             .builder \
             .appName("Python Spark create RDD example") \
             .config("spark.some.config.option", "some-value") \
             .getOrCreate()
```

1 Reading the json the data using pyspark dataframe

```
In [24]: df = spark.read.json(r"E:\Documents\University of Hildesheim\Distributed data analyti
         df.show()
```

```
+------------------+-----------------+----------+---------+------+----+
|            course|              dob|first_name|last_name|points|s_id|
+------------------+-----------------+----------+---------+------+----+
|Humanities and Art| October 14, 1983|      Alan|      Joe|    10|   1|
|  Computer Science|September 26, 1980|    Martin|  Genberg|    17|   2|
|    Graphic Design|     June 12, 1982|     Athur|   Watson|    16|   3|
|    Graphic Design|     April 5, 1987|  Anabelle|  Sanberg|    12|   4|
|        Psychology| November 1, 1978|      Kira| Schommer|    11|   5|
|          Business| 17 February 1981| Christian|   Kiriam|    10|   6|
|  Machine Learning|   1 January 1984|   Barbara|  Ballard|    14|   7|
|     Deep Learning| January 13, 1978|      John|     null|    10|   8|
|  Machine Learning| 26 December 1989|    Marcus|   Carson|    15|   9|
|           Physics| 30 December 1987|     Marta|   Brooks|    11|  10|
|    Data Analytics|     June 12, 1975|     Holly| Schwartz|    12|  11|
|  Computer Science|      July 2, 1985|     April|    Black|  null|  12|
|  Computer Science|     July 22, 1980|     Irene|  Bradley|    13|  13|
|        Psychology|  7 February 1986|      Mark|    Weber|    12|  14|
```

```
|       Informatics|      May 18, 1987|     Rosie|   Norman|     9|  15|
|          Business|   August 10, 1984|    Martin|   Steele|     7|  16|
|  Machine Learning|  16 December 1990|     Colin| Martinez|     9|  17|
|    Data Analytics|              null|   Bridget|    Twain|     6|  18|
|          Business|      7 March 1980|   Darlene|    Mills|    19|  19|
|    Data Analytics|      June 2, 1985|   Zachary|     null|    10|  20|
+------------------+------------------+----------+---------+------+----+
```

2 Replace the null values in column['points'] by the mean of all points

- the function fill_with_mean uses aggregate function to find the avg of all the values in points column.
- then that value is the replaced with the null values by selecting only points column excluding other columns from the process.

```
In [25]: def fill_with_mean(df, exclude=set()):
             stats = df.agg(*(
                 fn.avg(c).alias(c) for c in df.columns if c not in exclude
             ))
             return df.na.fill(stats.first().asDict())
         df=fill_with_mean(df, ['course', 'dob', 'first_name', 'last_name','s_id'])
         df.show()
```

```
+------------------+------------------+----------+---------+------+----+
|            course|               dob|first_name|last_name|points|s_id|
+------------------+------------------+----------+---------+------+----+
|Humanities and Art|  October 14, 1983|      Alan|      Joe|    10|   1|
|  Computer Science|September 26, 1980|    Martin|  Genberg|    17|   2|
|    Graphic Design|     June 12, 1982|     Athur|   Watson|    16|   3|
|    Graphic Design|     April 5, 1987|  Anabelle|  Sanberg|    12|   4|
|        Psychology|  November 1, 1978|      Kira| Schommer|    11|   5|
|          Business|  17 February 1981| Christian|   Kiriam|    10|   6|
|  Machine Learning|    1 January 1984|   Barbara|  Ballard|    14|   7|
|     Deep Learning|  January 13, 1978|      John|     null|    10|   8|
|  Machine Learning|  26 December 1989|    Marcus|   Carson|    15|   9|
|           Physics|  30 December 1987|     Marta|   Brooks|    11|  10|
|    Data Analytics|     June 12, 1975|     Holly| Schwartz|    12|  11|
|  Computer Science|      July 2, 1985|     April|    Black|    11|  12|
|  Computer Science|     July 22, 1980|     Irene|  Bradley|    13|  13|
|        Psychology|   7 February 1986|      Mark|    Weber|    12|  14|
|       Informatics|      May 18, 1987|     Rosie|   Norman|     9|  15|
|          Business|   August 10, 1984|    Martin|   Steele|     7|  16|
|  Machine Learning|  16 December 1990|     Colin| Martinez|     9|  17|
|    Data Analytics|              null|   Bridget|    Twain|     6|  18|
|          Business|      7 March 1980|   Darlene|    Mills|    19|  19|
|    Data Analytics|      June 2, 1985|   Zachary|     null|    10|  20|
```

```
+-----------------+-----------------+---------+---------+------+----+
```

3 Replace the null value(s) in column dob by “unknown”

```
In [26]: df=df.na.fill('unknown',['dob'])
         df.show()
```

```
+-----------------+-----------------+---------+---------+------+----+
|           course|              dob|first_name|last_name|points|s_id|
+-----------------+-----------------+---------+---------+------+----+
|Humanities and Art|  October 14, 1983|      Alan|      Joe|    10|   1|
|  Computer Science|September 26, 1980|    Martin|  Genberg|    17|   2|
|    Graphic Design|     June 12, 1982|     Athur|   Watson|    16|   3|
|    Graphic Design|     April 5, 1987|  Anabelle|  Sanberg|    12|   4|
|        Psychology|  November 1, 1978|      Kira| Schommer|    11|   5|
|          Business| 17 February 1981| Christian|   Kiriam|    10|   6|
|  Machine Learning|    1 January 1984|   Barbara|  Ballard|    14|   7|
|     Deep Learning|  January 13, 1978|      John|     null|    10|   8|
|  Machine Learning| 26 December 1989|    Marcus|   Carson|    15|   9|
|           Physics| 30 December 1987|     Marta|   Brooks|    11|  10|
|    Data Analytics|     June 12, 1975|     Holly| Schwartz|    12|  11|
|  Computer Science|      July 2, 1985|     April|    Black|    11|  12|
|  Computer Science|     July 22, 1980|     Irene|  Bradley|    13|  13|
|        Psychology|   7 February 1986|      Mark|    Weber|    12|  14|
|       Informatics|      May 18, 1987|     Rosie|   Norman|     9|  15|
|          Business|   August 10, 1984|    Martin|   Steele|     7|  16|
|  Machine Learning| 16 December 1990|     Colin| Martinez|     9|  17|
|    Data Analytics|          unknown|   Bridget|    Twain|     6|  18|
|          Business|      7 March 1980|   Darlene|    Mills|    19|  19|
|    Data Analytics|      June 2, 1985|   Zachary|     null|    10|  20|
+-----------------+-----------------+---------+---------+------+----+
```

4 Replace the null value(s) in column last name by “–”

```
In [27]: df=df.na.fill('--',['last_name'])
         df.show()
```

```
+-----------------+-----------------+---------+---------+------+----+
|           course|              dob|first_name|last_name|points|s_id|
+-----------------+-----------------+---------+---------+------+----+
|Humanities and Art|  October 14, 1983|      Alan|      Joe|    10|   1|
|  Computer Science|September 26, 1980|    Martin|  Genberg|    17|   2|
|    Graphic Design|     June 12, 1982|     Athur|   Watson|    16|   3|
```

```
|    Graphic Design|     April 5, 1987|  Anabelle|  Sanberg|    12|   4|
|        Psychology|  November 1, 1978|      Kira| Schommer|    11|   5|
|          Business|  17 February 1981| Christian|   Kiriam|    10|   6|
|  Machine Learning|    1 January 1984|   Barbara|  Ballard|    14|   7|
|     Deep Learning|  January 13, 1978|      John|       --|    10|   8|
|  Machine Learning|  26 December 1989|    Marcus|   Carson|    15|   9|
|           Physics|  30 December 1987|     Marta|   Brooks|    11|  10|
|    Data Analytics|     June 12, 1975|     Holly| Schwartz|    12|  11|
|  Computer Science|      July 2, 1985|     April|    Black|    11|  12|
|  Computer Science|     July 22, 1980|     Irene|  Bradley|    13|  13|
|        Psychology|   7 February 1986|      Mark|    Weber|    12|  14|
|       Informatics|      May 18, 1987|     Rosie|   Norman|     9|  15|
|          Business|   August 10, 1984|    Martin|   Steele|     7|  16|
|  Machine Learning|  16 December 1990|     Colin| Martinez|     9|  17|
|    Data Analytics|           unknown|   Bridget|    Twain|     6|  18|
|          Business|      7 March 1980|   Darlene|    Mills|    19|  19|
|    Data Analytics|      June 2, 1985|   Zachary|       --|    10|  20|
+------------------+------------------+----------+---------+------+----+
```

5 Converting dob column to DD-MM-YYYY

- first the date the converted by using to_data function which converts the date to YYYY-MM-DD

```
In [28]: def to_date_(col, formats=("MMM dd, yyyy", "dd MMM yyyy")):
             return fn.coalesce(*[fn.to_date(col, f) for f in formats])
         df=df.withColumn("dob",to_date_("dob"))
         df.show()
```

```
+------------------+----------+----------+---------+------+----+
|            course|       dob|first_name|last_name|points|s_id|
+------------------+----------+----------+---------+------+----+
|Humanities and Art|1983-10-14|      Alan|      Joe|    10|   1|
|  Computer Science|1980-09-26|    Martin|  Genberg|    17|   2|
|    Graphic Design|1982-06-12|     Athur|   Watson|    16|   3|
|    Graphic Design|1987-04-05|  Anabelle|  Sanberg|    12|   4|
|        Psychology|1978-11-01|      Kira| Schommer|    11|   5|
|          Business|1981-02-17| Christian|   Kiriam|    10|   6|
|  Machine Learning|1984-01-01|   Barbara|  Ballard|    14|   7|
|     Deep Learning|1978-01-13|      John|       --|    10|   8|
|  Machine Learning|1989-12-26|    Marcus|   Carson|    15|   9|
|           Physics|1987-12-30|     Marta|   Brooks|    11|  10|
|    Data Analytics|1975-06-12|     Holly| Schwartz|    12|  11|
|  Computer Science|1985-07-02|     April|    Black|    11|  12|
|  Computer Science|1980-07-22|     Irene|  Bradley|    13|  13|
|        Psychology|1986-02-07|      Mark|    Weber|    12|  14|
```

```
|       Informatics|1987-05-18|     Rosie|   Norman|     9|  15|
|          Business|1984-08-10|    Martin|   Steele|     7|  16|
|  Machine Learning|1990-12-16|     Colin| Martinez|     9|  17|
|    Data Analytics|      null|   Bridget|    Twain|     6|  18|
|          Business|1980-03-07|   Darlene|    Mills|    19|  19|
|    Data Analytics|1985-06-02|   Zachary|       --|    10|  20|
+------------------+----------+----------+---------+------+----+
```

YYYY-MM-DD is the converted to DD-MM-YYYY using the function date_format

```
In [29]: df=df.withColumn('dob', fn.date_format('dob', 'dd-MM-yyyy'))
         df.show()
```

```
+------------------+----------+----------+---------+------+----+
|            course|       dob|first_name|last_name|points|s_id|
+------------------+----------+----------+---------+------+----+
|Humanities and Art|14-10-1983|      Alan|      Joe|    10|   1|
|  Computer Science|26-09-1980|    Martin|  Genberg|    17|   2|
|    Graphic Design|12-06-1982|     Athur|   Watson|    16|   3|
|    Graphic Design|05-04-1987|  Anabelle|  Sanberg|    12|   4|
|        Psychology|01-11-1978|      Kira| Schommer|    11|   5|
|          Business|17-02-1981| Christian|   Kiriam|    10|   6|
|  Machine Learning|01-01-1984|   Barbara|  Ballard|    14|   7|
|     Deep Learning|13-01-1978|      John|       --|    10|   8|
|  Machine Learning|26-12-1989|    Marcus|   Carson|    15|   9|
|           Physics|30-12-1987|     Marta|   Brooks|    11|  10|
|    Data Analytics|12-06-1975|     Holly| Schwartz|    12|  11|
|  Computer Science|02-07-1985|     April|    Black|    11|  12|
|  Computer Science|22-07-1980|     Irene|  Bradley|    13|  13|
|        Psychology|07-02-1986|      Mark|    Weber|    12|  14|
|       Informatics|18-05-1987|     Rosie|   Norman|     9|  15|
|          Business|10-08-1984|    Martin|   Steele|     7|  16|
|  Machine Learning|16-12-1990|     Colin| Martinez|     9|  17|
|    Data Analytics|      null|   Bridget|    Twain|     6|  18|
|          Business|07-03-1980|   Darlene|    Mills|    19|  19|
|    Data Analytics|02-06-1985|   Zachary|       --|    10|  20|
+------------------+----------+----------+---------+------+----+
```

6 The age is callculated by subtracting the current time and the dob column and casting to interger

```
In [30]: df=df.withColumn("age_in_years",(fn.months_between(fn.to_date(fn.current_date(),'dd-MM
         df.show()
```

```
+------------------+----------+----------+---------+------+----+------------+
|            course|       dob|first_name|last_name|points|s_id|age_in_years|
+------------------+----------+----------+---------+------+----+------------+
|Humanities and Art|14-10-1983|      Alan|      Joe|    10|   1|          35|
|  Computer Science|26-09-1980|    Martin|  Genberg|    17|   2|          38|
|    Graphic Design|12-06-1982|     Athur|   Watson|    16|   3|          37|
|    Graphic Design|05-04-1987|  Anabelle|  Sanberg|    12|   4|          32|
|        Psychology|01-11-1978|      Kira| Schommer|    11|   5|          40|
|          Business|17-02-1981| Christian|   Kiriam|    10|   6|          38|
|  Machine Learning|01-01-1984|   Barbara|  Ballard|    14|   7|          35|
|     Deep Learning|13-01-1978|      John|       --|    10|   8|          41|
|  Machine Learning|26-12-1989|    Marcus|   Carson|    15|   9|          29|
|           Physics|30-12-1987|     Marta|   Brooks|    11|  10|          31|
|    Data Analytics|12-06-1975|     Holly| Schwartz|    12|  11|          44|
|  Computer Science|02-07-1985|     April|    Black|    11|  12|          33|
|  Computer Science|22-07-1980|     Irene|  Bradley|    13|  13|          38|
|        Psychology|07-02-1986|      Mark|    Weber|    12|  14|          33|
|       Informatics|18-05-1987|     Rosie|   Norman|     9|  15|          32|
|          Business|10-08-1984|    Martin|   Steele|     7|  16|          34|
|  Machine Learning|16-12-1990|     Colin| Martinez|     9|  17|          28|
|    Data Analytics|      null|   Bridget|    Twain|     6|  18|        null|
|          Business|07-03-1980|   Darlene|    Mills|    19|  19|          39|
|    Data Analytics|02-06-1985|   Zachary|       --|    10|  20|          34|
+------------------+----------+----------+---------+------+----+------------+
```

7 Granting some points for good performed students in the class

- At first the standard deviation is calculated for all the students
- then the mean is calculated to add it with standard deviation and find the perform variable
- which is used to give full marks to students who has current marks greater than the perform variable

```
In [32]: def fill_with_std(df, exclude=set()):
             stats = df.agg(*(
                 fn.stddev(c).alias(c) for c in df.columns if c not in exclude
             ))
             return df.fill(stats.first().asDict())
         df_stats= df.select(fn.stddev(df.points).alias('std')).collect()
         std = df_stats[0]['std']
         df_stats_mean= df.select(fn.mean(df.points).alias('std')).collect()
         mean = df_stats_mean[0]['std']
         perform=std+mean
         df = df.withColumn("points", fn.when(df.points > perform, 20).otherwise(fn.col('points
         df.show(truncate=False)
```

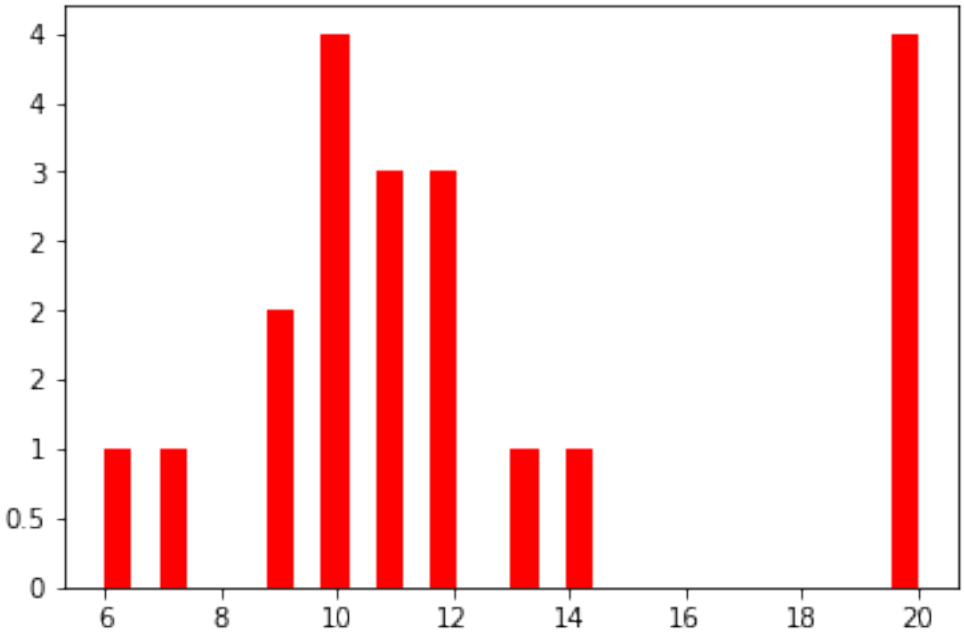
```
+------------------+----------+----------+---------+------+----+------------+
```

```
|course            |dob       |first_name|last_name|points|s_id|age_in_years|
+------------------+----------+----------+---------+------+----+------------+
|Humanities and Art|14-10-1983|Alan      |Joe      |10    |1   |35          |
|Computer Science  |26-09-1980|Martin    |Genberg  |20    |2   |38          |
|Graphic Design    |12-06-1982|Athur     |Watson   |20    |3   |37          |
|Graphic Design    |05-04-1987|Anabelle  |Sanberg  |12    |4   |32          |
|Psychology        |01-11-1978|Kira      |Schommer |11    |5   |40          |
|Business          |17-02-1981|Christian |Kiriam   |10    |6   |38          |
|Machine Learning  |01-01-1984|Barbara   |Ballard  |14    |7   |35          |
|Deep Learning     |13-01-1978|John      |--       |10    |8   |41          |
|Machine Learning  |26-12-1989|Marcus    |Carson   |20    |9   |29          |
|Physics           |30-12-1987|Marta     |Brooks   |11    |10  |31          |
|Data Analytics    |12-06-1975|Holly     |Schwartz |12    |11  |44          |
|Computer Science  |02-07-1985|April     |Black    |11    |12  |33          |
|Computer Science  |22-07-1980|Irene     |Bradley  |13    |13  |38          |
|Psychology        |07-02-1986|Mark      |Weber    |12    |14  |33          |
|Informatics       |18-05-1987|Rosie     |Norman   |9     |15  |32          |
|Business          |10-08-1984|Martin    |Steele   |7     |16  |34          |
|Machine Learning  |16-12-1990|Colin     |Martinez |9     |17  |28          |
|Data Analytics    |null      |Bridget   |Twain    |6     |18  |null        |
|Business          |07-03-1980|Darlene   |Mills    |20    |19  |39          |
|Data Analytics    |02-06-1985|Zachary   |--       |10    |20  |34          |
+------------------+----------+----------+---------+------+----+------------+
```

8 Histogram on the new points

```
In [33]: a=df.select('points')
         fig, ax = plt.subplots()
         hist(ax, a, bins = 30, color=['red'])
```

```
Out[33]: (array([1., 0., 1., 0., 0., 0., 2., 0., 4., 0., 3., 0., 3., 0., 0., 1., 0.,
                 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 4.]),
          array([ 6.        ,  6.46666667,  6.93333333,  7.4       ,  7.86666667,
                  8.33333333,  8.8       ,  9.26666667,  9.73333333, 10.2       ,
                 10.66666667, 11.13333333, 11.6       , 12.06666667, 12.53333333,
                 13.        , 13.46666667, 13.93333333, 14.4       , 14.86666667,
                 15.33333333, 15.8       , 16.26666667, 16.73333333, 17.2       ,
                 17.66666667, 18.13333333, 18.6       , 19.06666667, 19.53333333,
                 20.        ]),
          <a list of 30 Patch objects>)
```



In [ ]:

spark_2

June 30, 2019

In [2]:
```
from pyspark.sql import SparkSession
from pyspark import SparkContext, SparkConf
import pyspark.sql.functions as F
from pyspark.sql.types import Row
from pyspark.sql import SQLContext
import matplotlib.pyplot as plt
import numpy as np
from pyspark_dist_explore import hist
from pyspark.sql.types import IntegerType
import datetime
import time
```

1 Importing the data using spark dataframe from tags.dat

In [3]:
```
conf = SparkConf().setAppName("Spark Count")
sc = SparkContext.getOrCreate(conf=conf)

spark = SparkSession \
    .builder \
    .appName("Python Spark create RDD example") \
    .config("spark.some.config.option", "some-value") \
    .getOrCreate()


df = spark.read.option("header", "false") \
    .option("delimiter", ":") \
    .option("inferSchema", "true") \
    .csv(r"E:\Documents\University of Hildesheim\Distributed data analytics\lab8\ml-10
df.show()
```

```
+---+----+-----+----+------------------+----+----------+
|_c0| _c1|  _c2| _c3|               _c4| _c5|       _c6|
+---+----+-----+----+------------------+----+----------+
| 15|null| 4973|null|       excellent!|null|1215184630|
| 20|null| 1747|null|         politics|null|1188263867|
| 20|null| 1747|null|           satire|null|1188263867|
| 20|null| 2424|null|  chick flick 212|null|1188263835|
```

```
| 20|null| 2424|null|               hanks|null|1188263835|
| 20|null| 2424|null|                ryan|null|1188263835|
| 20|null| 2947|null|              action|null|1188263755|
| 20|null| 2947|null|                bond|null|1188263756|
| 20|null| 3033|null|               spoof|null|1188263880|
| 20|null| 3033|null|           star wars|null|1188263880|
| 20|null| 7438|null|              bloody|null|1188263801|
| 20|null| 7438|null|             kung fu|null|1188263801|
| 20|null| 7438|null|           Tarantino|null|1188263801|
| 21|null|55247|null|                   R|null|1205081506|
| 21|null|55253|null|               NC-17|null|1205081488|
| 25|null|   50|null|        Kevin Spacey|null|1166101426|
| 25|null| 6709|null|         Johnny Depp|null|1162147221|
| 31|null|   65|null|        buddy comedy|null|1188263759|
| 31|null|  546|null|strangely compelling|null|1188263674|
| 31|null| 1091|null|         catastrophe|null|1188263741|
+---+----+-----+----+--------------------+----+----------+
only showing top 20 rows
```

2 Final data with column headings

```
In [4]: columns_to_drop = ['_c1', '_c3','_c5']
        df = df.drop(*columns_to_drop)
        df = df.select(F.col("_c0").alias("UserID"), F.col("_c2").alias("MovieID"),F.col("_c4")
        df=df.orderBy('UserID', ascending=True)
        df = df.withColumn("Timestamp", df["Timestamp"].cast(IntegerType())) #casting the colu
        df.show()
```

```
+------+-------+--------------------+----------+
|UserID|MovieID|                 Tag| Timestamp|
+------+-------+--------------------+----------+
|    15|   4973|          excellent!|1215184630|
|    20|   2424|               ryan|1188263835|
|    20|   1747|            politics|1188263867|
|    20|   2424|     chick flick 212|1188263835|
|    20|   7438|              bloody|1188263801|
|    20|   7438|             kung fu|1188263801|
|    20|   7438|           Tarantino|1188263801|
|    20|   2947|              action|1188263755|
|    20|   1747|              satire|1188263867|
|    20|   2947|                bond|1188263756|
|    20|   3033|               spoof|1188263880|
|    20|   3033|           star wars|1188263880|
|    20|   2424|               hanks|1188263835|
|    21|  55253|               NC-17|1205081488|
```

```
|    21|  55247|                   R|1205081506|
|    25|     50|        Kevin Spacey|1166101426|
|    25|   6709|         Johnny Depp|1162147221|
|    31|     65|        buddy comedy|1188263759|
|    31|    546|strangely compelling|1188263674|
|    31|   1091|         catastrophe|1188263741|
+------+-------+--------------------+----------+
only showing top 20 rows
```

2.0.1 The dataframe is grouped by UserID and an aggregation using the function collect_list to get all the timestamp for each userID

```
In [5]: df_grp=df.withColumn('UserID',df.UserID.cast("int")).groupBy(['UserID']).agg(F.collect_
        df_grp=df_grp.orderBy('UserID', ascending=True)
        df_grp.show()
```

```
+------+--------------------+
|UserID|                time|
+------+--------------------+
|    15|        [1215184630]|
|    20|[1188263867, 1188...|
|    21|[1205081506, 1205...|
|    25|[1166101426, 1162...|
|    31|[1188263759, 1188...|
|    32|        [1164735331]|
|    39|[1188263791, 1188...|
|    48|[1215135611, 1215...|
|    49|[1188264255, 1188...|
|    75|        [1162160415]|
|    78|        [1176691425]|
|   109|[1211433235, 1165...|
|   127|[1188265347, 1188...|
|   133|[1188265396, 1188...|
|   146|[1226742764, 1196...|
|   147|[1162188712, 1162...|
|   170|        [1162209176]|
|   175|[1188441420, 1192...|
|   181|[1188266123, 1188...|
|   190|[1151700107, 1151...|
+------+--------------------+
only showing top 20 rows
```

3 Converting pyspark dataframe to pandas dataframe for data calculations

```
In [6]: df_grp=df_grp.toPandas()
        print(df_grp)
```

```
      UserID                                                  time
0         15                                         [1215184630]
1         20  [1188263867, 1188263867, 1188263835, 118826383...
2         21                           [1205081506, 1205081488]
3         25                           [1166101426, 1162147221]
4         31  [1188263759, 1188263674, 1188263741, 118826370...
5         32                                         [1164735331]
6         39  [1188263791, 1188263843, 1188263764, 118826378...
7         48                           [1215135611, 1215135517]
8         49  [1188264255, 1188264178, 1188264095, 118826409...
9         75                                         [1162160415]
10        78                                         [1176691425]
11       109  [1211433235, 1165555281, 1165555288, 123112228...
12       127  [1188265347, 1188265347, 1188265347, 118826536...
13       133  [1188265396, 1188265375, 1188265375, 118826537...
14       146  [1226742764, 1196517851, 1213424486, 121342443...
15       147                           [1162188712, 1162188631]
16       170                                         [1162209176]
17       175                           [1188441420, 1192990133]
18       181   [1188266123, 1188266123, 1188266123, 1188266123]
19       190  [1151700107, 1151700139, 1140037169, 115170012...
20       222                                         [1188267261]
21       233  [1156523175, 1168109250, 1173525910, 117352591...
22       240                                         [1139349414]
23       249                                         [1162271746]
24       267                                         [1192049634]
25       283              [1137972371, 1137273990, 1137274016]
26       284  [1188274135, 1188274458, 1188274459, 118827445...
27       299  [1188275113, 1188275133, 1188275084, 118827508...
28       325  [1185486223, 1185663289, 1172743571, 117274367...
29       374  [1188280075, 1188280075, 1188280075, 118828007...
...      ...                                                 ...
3979   71226                           [1215852911, 1215792475]
3980   71241  [1188168485, 1188168459, 1188168459, 118816843...
3981   71254  [1188170431, 1188170431, 1188172249, 118817224...
3982   71262  [1188171125, 1188170879, 1188170905, 118817106...
3983   71278                           [1215027062, 1219168674]
3984   71287  [1188177633, 1188177725, 1188177750, 118817776...
3985   71288                                       [1140897965]
3986   71291  [1188178536, 1188178608, 1188178673, 118817832...
3987   71301  [1188180451, 1188180451, 1188180451, 118818045...
3988   71307                                       [1188278930]
```

```
3989   71315  [1188185498, 1188185498, 1188185651, 118818565...
3990   71323  [1188187590, 1188187633, 1188187609, 118818761...
3991   71331  [1137174983, 1137174978, 1137174978, 113717499...
3992   71394  [1188832352, 1188832312, 1188832326, 118883227...
3993   71414  [1188233873, 1188233873, 1188233671, 118823367...
3994   71420  [1206662843, 1206662843, 1206662717, 120666271...
3995   71424  [1188236322, 1188236322, 1188236863, 118823686...
3996   71448  [1188248846, 1188248808, 1188248785, 118824873...
3997   71455  [1206704501, 1163327521, 1206706041, 116211036...
3998   71478  [1139430266, 1139430266, 1139430266, 113976980...
3999   71483                                      [1139090411]
4000   71487              [1215191810, 1215193060, 1215191063]
4001   71497  [1188331015, 1188330757, 1188331010, 118825637...
4002   71509  [1214944808, 1186508886, 1214943600, 121494350...
4003   71512                                      [1189368506]
4004   71525                                      [1150030459]
4005   71529              [1162098757, 1162098746, 1162098746]
4006   71534  [1196645880, 1196647394, 1196646489, 119664714...
4007   71536              [1162110221, 1162126283, 1162110266]
4008   71556  [1188263570, 1188263571, 1188263606, 118826362...

[4009 rows x 2 columns]
```

4 Finding out the tagging frequency of each user after separating the Timestamps for each user

- Calulated the mean and standard deviation for each user

```
In [7]: sess=[]
        tags=[]
        for userid in range (0,len(df_grp)):
            session=1
            frequen=1
            freq=[]
            for i in range (1,len(df_grp['time'][userid])):
                    if(df_grp['time'][userid][i]-df_grp['time'][userid][i-1])>1800:
                        session+=1
                        freq.append(frequen)
                        frequen=0
                    frequen+=1
            freq.append(frequen)
            tags.append(freq)
            sess.append(session)
        df_grp["Session"]=sess
        df_grp["frequency"]=tags
        mean_of_frq=[]
```

```
        sum_of_frq=[]
        stddev=[]
        for i in tags:
            mean=np.mean(i)
            total=np.sum(i)
            std=np.std(i)
            mean_of_frq.append(mean)
            sum_of_frq.append(total)
            stddev.append(std)
        df_grp["mean_of_frequency"]= mean_of_frq
        df_grp["sum_of_frequency"]= sum_of_frq
        df_grp["stddev"]=stddev
        df=spark.createDataFrame(df_grp)
        # df = df.drop(*columns_to_drop)
        df.show()
```

```
+------+--------------------+-------+--------------------+------------------+----------------+
|UserID|                time|Session|           frequency| mean_of_frequency|sum_of_frequency|
+------+--------------------+-------+--------------------+------------------+----------------+
|    15|        [1215184630]|      1|                 [1]|               1.0|               1|
|    20|[1188263867, 1188...|      1|                [12]|              12.0|              12|
|    21|[1205081506, 1205...|      1|                 [2]|               2.0|               2|
|    25|[1166101426, 1162...|      1|                 [2]|               2.0|               2|
|    31|[1188263759, 1188...|      1|                 [5]|               5.0|               5|
|    32|        [1164735331]|      1|                 [1]|               1.0|               1|
|    39|[1188263791, 1188...|      1|                 [5]|               5.0|               5|
|    48|[1215135611, 1215...|      1|                 [2]|               2.0|               2|
|    49|[1188264255, 1188...|      1|                [15]|              15.0|              15|
|    75|        [1162160415]|      1|                 [1]|               1.0|               1|
|    78|        [1176691425]|      1|                 [1]|               1.0|               1|
|   109|[1211433235, 1165...|      9|[3, 2, 6, 2, 5, 2...|2.7777777777777777|              25|
|   127|[1188265347, 1188...|      1|                [26]|              26.0|              26|
|   133|[1188265396, 1188...|      1|                 [5]|               5.0|               5|
|   146|[1226742764, 1196...|    705|[2, 3, 3, 1, 3, 1...|2.3375886524822693|            1648|
|   147|[1162188712, 1162...|      1|                 [2]|               2.0|               2|
|   170|        [1162209176]|      1|                 [1]|               1.0|               1|
|   175|[1188441420, 1192...|      2|              [1, 1]|               1.0|               2|
|   181|[1188266123, 1188...|      1|                 [4]|               4.0|               4|
|   190|[1151700107, 1151...|      8|[3, 3, 2, 1, 5, 6...|              3.25|              26|
+------+--------------------+-------+--------------------+------------------+----------------+
only showing top 20 rows
```

5 Mean and standard deviation of the tagging frequency of All user

```
In [8]: mean_std=df.select(F.mean(df['sum_of_frequency']).alias('average'),F.stddev(df['sum_of_
        mean_std.show()
```

```
+------------------+------------------+
|           average|standard_deviation|
+------------------+------------------+
|23.287353454726865| 171.3979804629899|
+------------------+------------------+
```

6 List of users with a mean tagging frequency within the two standard deviation from

In [9]:
```
l_bound=23.287353454726865-2*171.3979804629899
u_bound=23.287353454726865+2*171.3979804629899
df.filter(df['mean_of_frequency'].between(l_bound,u_bound)).show()
```

```
+------+--------------------+-------+--------------------+------------------+----------------+
|UserID|                time|Session|           frequency| mean_of_frequency|sum_of_frequency|
+------+--------------------+-------+--------------------+------------------+----------------+
|    15|        [1215184630]|      1|                 [1]|               1.0|               1|
|    20|[1188263867, 1188...|      1|                [12]|              12.0|              12|
|    21|[1205081506, 1205...|      1|                 [2]|               2.0|               2|
|    25|[1166101426, 1162...|      1|                 [2]|               2.0|               2|
|    31|[1188263759, 1188...|      1|                 [5]|               5.0|               5|
|    32|        [1164735331]|      1|                 [1]|               1.0|               1|
|    39|[1188263791, 1188...|      1|                 [5]|               5.0|               5|
|    48|[1215135611, 1215...|      1|                 [2]|               2.0|               2|
|    49|[1188264255, 1188...|      1|                [15]|              15.0|              15|
|    75|        [1162160415]|      1|                 [1]|               1.0|               1|
|    78|        [1176691425]|      1|                 [1]|               1.0|               1|
|   109|[1211433235, 1165...|      9|[3, 2, 6, 2, 5, 2...|2.7777777777777777|              25|
|   127|[1188265347, 1188...|      1|                [26]|              26.0|              26|
|   133|[1188265396, 1188...|      1|                 [5]|               5.0|               5|
|   146|[1226742764, 1196...|    705|[2, 3, 3, 1, 3, 1...|2.3375886524822693|            1648|
|   147|[1162188712, 1162...|      1|                 [2]|               2.0|               2|
|   170|        [1162209176]|      1|                 [1]|               1.0|               1|
|   175|[1188441420, 1192...|      2|              [1, 1]|               1.0|               2|
|   181|[1188266123, 1188...|      1|                 [4]|               4.0|               4|
|   190|[1151700107, 1151...|      8|[3, 3, 2, 1, 5, 6...|              3.25|              26|
+------+--------------------+-------+--------------------+------------------+----------------+
only showing top 20 rows
```

In [ ]: