Sayeed Ahmed – SEC01 (NUID 002191535)

Big Data System Engineering with Scala
Spring 2023
Assignment No. 1 - Spark

Loading the dataset into a data frame:

```
//Importing data set
val df = spark.read.option("header", "true")
                   .option("inferSchema", "true")
                   .csv("/FileStore/shared_uploads/ahmed.say@northeastern.edu/train.csv")
df.show(5)
```

▸ (3) Spark Jobs

▸ df: org.apache.spark.sql.DataFrame = [PassengerId: integer, Survived: integer ... 10 more fields]

```
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+-------+-----+--------+
|PassengerId|Survived|Pclass|                Name|   Sex| Age|SibSp|Parch|          Ticket|   Fare|Cabin|Embarked|
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+-------+-----+--------+
|          1|       0|     3|Braund, Mr. Owen ...|  male|22.0|    1|    0|       A/5 21171|   7.25| null|       S|
|          2|       1|     1|Cumings, Mrs. Joh...|female|38.0|    1|    0|        PC 17599|71.2833|  C85|       C|
|          3|       1|     3|Heikkinen, Miss. ...|female|26.0|    0|    0|STON/O2. 3101282|  7.925| null|       S|
|          4|       1|     1|Futrelle, Mrs. Ja...|female|35.0|    1|    0|          113803|   53.1| C123|       S|
|          5|       0|     3|Allen, Mr. Willia...|  male|35.0|    0|    0|          373450|   8.05| null|       S|
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+-------+-----+--------+
only showing top 5 rows

df: org.apache.spark.sql.DataFrame = [PassengerId: int, Survived: int ... 10 more fields]
```

Command took 4.50 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:04:35 PM on My Cluster

1)What is the average ticket fare for each Ticket class?

```
//1)What is the average ticket fare for each Ticket class?
df.groupBy("Pclass").agg(avg("Fare")).show()
```

▸ (2) Spark Jobs

```
+------+------------------+
|Pclass|         avg(Fare)|
+------+------------------+
|     1| 84.15468749999992|
|     3|13.675550101832997|
|     2| 20.66218315217391|
+------+------------------+
```

Command took 3.45 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:08:12 PM on My Cluster

2) What is the survival percentage for each Ticket class? Which class has the highest survival rate?

```
//What is the survival percentage for each Ticket class? Which class has the highest survival rate?
val totalDF = df.count
val survivalDF = df.filter($"survived" === 1).groupBy("pclass").count
val survivalPercentDF = survivalDF.withColumn("percentage", col("count")/totalDF *100)
survivalPercentDF.show()
val highestSurvival = survivalPercentDF.agg(max("percentage")).first().getDouble(0)
val highestSurvivalClass = survivalPercentDF.select($"pClass").filter($"percentage" === highestSurvival).first().getInt(0)
println(s"The class with the highest survival rate is: $highestSurvivalClass")
```

```
▸ (9) Spark Jobs
▸ survivalDF: org.apache.spark.sql.DataFrame = [pclass: integer, count: long]
▸ survivalPercentDF: org.apache.spark.sql.DataFrame = [pclass: integer, count: long ... 1 more field]
+------+-----+-----------------+
|pclass|count|       percentage|
+------+-----+-----------------+
|     1|  136|15.26374859708193|
|     3|  119|13.35578002244669|
|     2|   87|9.764309764309765|
+------+-----+-----------------+

The class with the highest survival rate is: 1
totalDF: Long = 891
survivalDF: org.apache.spark.sql.DataFrame = [pclass: int, count: bigint]
survivalPercentDF: org.apache.spark.sql.DataFrame = [pclass: int, count: bigint ... 1 more field]
highestSurvival: Double = 15.26374859708193
highestSurvivalClass: Int = 1

Command took 5.25 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:23:07 PM on My Cluster
```

3) Rose DeWitt Bukater was 17 years old when she boarded the titanic. She is traveling with her mother and fiancé, She is traveling first class. With the information of her age, gender, class she is traveling in, and the fact that she is traveling with one parent, find the number of passengers who could possibly be Rose.

Cmd 5

```
/*Rose DeWitt Bukater was 17 years old when she boarded the titanic. She is traveling with her
mother and fiance( they are not married yet, so they are not related). She is traveling first class.
With the information of her age, gender, class she is traveling in, and the fact that she is traveling
with one parent, find the number of passengers who could possibly be Rose.*/
val possiblyRose = df.filter(df("Age") === 17 && df("Sex") === "female").filter(col("pClass") === 1)
                     .filter(col("parch") === 1).filter(col("SibSp") === 0).count()
println(s"The number of passengers who could possibly be Rose: $possiblyRose")
```

```
▸ (2) Spark Jobs
The number of passengers who could possibly be Rose: 0
possiblyRose: Long = 0

Command took 1.73 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:30:21 PM on My Cluster
```

4) Jack Dawson born in 1892 died on April 15, 1912. He is either 20 or 19 years old. He travels 3rd class and has no relatives onboard. Find the number of passengers who could possibly be Jack? (PS: Yeah he's the guy who gets Rose)

```
/*Jack Dawson born in 1892 died on April 15, 1912. He is either 20 or 19 years old.
He travels 3rd class and has no relatives onboard. Find the number of passengers who could possibly be Jack?*/
val possiblyJack = df.filter(col("pclass") === 3).filter(col("Sex") === "male").filter((col("age") === 19)||(col("age") === 20))
                     .filter(col("parch") === 0).filter(col("SibSp") === 0).filter(col("survived") === 0).count
println(s"The number of passengers who could possibly be Jack: $possiblyJack")
```

▸ (2) Spark Jobs

```
The number of passengers who could possibly be Jack: 21
possiblyJack: Long = 21
```

Command took 0.79 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:39:21 PM on My Cluster

5)

a) Split the age for every 10 years. 1-10 as one age group, 11- 20 as another etc.

```
// Split the age into age groups of 10 years
val ageGroupDF = df.withColumn("ageGroup", when(col("age") <= 10, "0-10").when(col("age") <= 20, "11-20")
                              .when(col("age") <= 30, "21-30").when(col("age") <= 40, "31-40")
                              .when(col("age") <= 50, "41-50").when(col("age") <= 60, "51-60").when(col("age") <= 70, "61-70").otherwise("70+"))
ageGroupDF.show(10)
```

▸ (1) Spark Jobs

▸ ageGroupDF: org.apache.spark.sql.DataFrame = [PassengerId: integer, Survived: integer ... 11 more fields]

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	ageGroup
1	0	3	Braund, Mr. Owen ...	male	22.0	1	0	A/5 21171	7.25	null	S	21-30
2	1	1	Cumings, Mrs. Joh...	female	38.0	1	0	PC 17599	71.2833	C85	C	31-40
3	1	3	Heikkinen, Miss. ...	female	26.0	0	0	STON/O2. 3101282	7.925	null	S	21-30
4	1	1	Futrelle, Mrs. Ja...	female	35.0	1	0	113803	53.1	C123	S	31-40
5	0	3	Allen, Mr. Willia...	male	35.0	0	0	373450	8.05	null	S	31-40
6	0	3	Moran, Mr. James	male	null	0	0	330877	8.4583	null	Q	70+
7	0	1	McCarthy, Mr. Tim...	male	54.0	0	0	17463	51.8625	E46	S	51-60
8	0	3	Palsson, Master. ...	male	2.0	3	1	349909	21.075	null	S	0-10
9	1	3	Johnson, Mrs. Osc...	female	27.0	0	2	347742	11.1333	null	S	21-30
10	1	2	Nasser, Mrs. Nich...	female	14.0	1	0	237736	30.0708	null	C	11-20

only showing top 10 rows

```
ageGroupDF: org.apache.spark.sql.DataFrame = [PassengerId: int, Survived: int ... 11 more fields]
```

Command took 1.34 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:43:17 PM on My Cluster

b) What is the relation between the ages and the ticket fare?

Cmd 8

```
1  // Calculating the average fare for each age group
2  val avgFareByAge = ageGroupDF.groupBy("ageGroup").agg(avg("fare"))
3  avgFareByAge.show()
```

▸ (2) Spark Jobs

▸ avgFareByAge: org.apache.spark.sql.DataFrame = [ageGroup: string, avg(fare): double]

```
+--------+------------------+
|ageGroup|         avg(fare)|
+--------+------------------+
|     70+|22.262360989010993|
|   11-20|29.529531304347838|
|   21-30|28.306718695652194|
|    0-10|30.434439062500008|
|   41-50| 41.16318139534884|
|   31-40| 42.49610000000002|
|   51-60| 44.77480238095238|
|   61-70| 45.91078235294117|
+--------+------------------+

avgFareByAge: org.apache.spark.sql.DataFrame = [ageGroup: string, avg(fare): double]
```

Command took 2.72 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:45:59 PM on My Cluster

c) Which age group most likely survived?

Cmd 9

```
1  // Calculating the age group most likely survived
2  val maxSurvivedAgeGroup = ageGroupDF.groupBy("ageGroup").agg(avg("survived").as("Average"))
3                                      .sort(col("Average").desc).first().getString(0)
4  println(s"The age group with the highest survival rate is: $maxSurvivedAgeGroup")
```

▸ (2) Spark Jobs

```
The age group with the highest survival rate is: 0-10
maxSurvivedAgeGroup: String = 0-10
```

Command took 1.50 seconds -- by ahmed.say@northeastern.edu at 2/6/2023, 8:46:05 PM on My Cluster