Harshil Shah – SEC01 (NUID 002780887)

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

- GitHub Repo URL - https://github.com/harshilshahneu/CSYE7200-Harshil-Shah

---

- List of Tasks Implemented

- You are to load the dataset using Spark and perform the operations to answer the following questions.

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

  (The ticket class is the pclass column. 1st = Upper; 2nd = Middle ; 3rd = Lower. Find the average cost of the ticket fare for the 1st class, the 2ns class and the 3rd class)

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

  (survival rate = number of survived per class/ total number of passengers . In the survival column, 0 = dead and 1 = survived)

  3) 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. (PS: if you watched the movie you will know if she survived or died)

  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)

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

  What is the relation between the ages and the ticket fare? Which age group most likely survived ?

- Code

1. Setup

```
harshilshah@Harshils-MacBook-Air ~ % spark-shell
23/02/06 21:50:49 WARN Utils: Your hostname, Harshils-MacBook-Air.local resolves to a loopback address: 127.0.0.1; using 10.0.0.235 instead (on interface en0)
23/02/06 21:50:49 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/02/06 21:50:52 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Spark context Web UI available at http://10.0.0.235:4040
Spark context available as 'sc' (master = local[*], app id = local-1675738252675).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.3.1
      /_/

Using Scala version 2.12.15 (OpenJDK 64-Bit Server VM, Java 19.0.1)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val dataFrame = spark.read.option("header",true).csv("/Users/harshilshah/Desktop/Scala assignments/CSYE7200-Harshil-Shah/Datasets/train.csv")
dataFrame: org.apache.spark.sql.DataFrame = [PassengerId: string, Survived: string ... 10 more fields]

scala> val castedDF = dataFrame.selectExpr("PassengerId","cast(Survived as int) Survived","Pclass","Name","Sex","cast(Age as int) Age","SibSp","Parch","Ticket","cast(Fare as double) Fare","Cabin","Embarke
d")
castedDF: org.apache.spark.sql.DataFrame = [PassengerId: string, Survived: int ... 10 more fields]
```

- **Question 1**

```
scala> castedDF.groupBy("Pclass").avg("Fare").sort("Pclass").show()
```

- **Question 2**

```
scala> val dataFrame2 =  castedDF.groupBy("Pclass").sum("Survived")
dataFrame2: org.apache.spark.sql.DataFrame = [Pclass: string, sum(Survived): bigint]

scala> val total_passengers = castedDF.count()
total_passengers: Long = 891

scala> val classes = List(1, 2, 3)
classes: List[Int] = List(1, 2, 3)

scala> val survivalRates = for (c <- classes) yield {
     |     val survived = dataFrame2.filter(s"Pclass=$c").select("sum(Survived)").head().getLong(0)
     |     (c, (survived.toFloat/total_passengers.toFloat)*100)
     | }
survivalRates: List[(Int, Float)] = List((1,15.263748), (2,9.76431), (3,13.35578))
```

- **Question 3**

```
scala> castedDF.filter(castedDF("Pclass")==="1" && castedDF("Age") === "17" && castedDF("Sex") === "female" && castedDF("Parch") === "1" && castedDF("Survived") === 1).count()
res9: Long = 0
```

- **Question 4**

```
scala> castedDF.repartition(col("PClass"), col("Sex")).filter(col("PClass") === "3" &&
     |           col("Sex") === "male" &&
     |           col("Parch") === "0" &&
     |           col("Survived") === "0" &&
     |           col("SibSp") === "0" &&
     |           col("Age").between(19, 20)).count()
res47: Long = 22
```

- **Question 5**

```
scala> val ageGroup = castedDF.withColumn("Age Group",
     |   when($"Age" <= 10 && $"Age" >= 1, "1-10")
     |   .when($"Age" <= 20, "11-20")
     |   .when($"Age" <= 30, "21-30")
     |   .when($"Age" <= 40, "31-40")
     |   .when($"Age" <= 50, "41-50")
     |   .when($"Age" <= 60, "51-60")
     |   .when($"Age" <= 70, "61-70")
     |   .when($"Age" <= 80, "71-80")
     |   .when($"Age" <= 90, "81-90")
     |   .when($"Age" > 90, "90+")
     |   .otherwise("NULL / Zero"))
```

```
scala> ageGroup.groupBy("Age Group").agg(round(avg("Fare"),2) as "Average Fare").sort("Age Group").show()
```

```
scala> ageGroup.groupBy("Age Group").agg(round(avg("Survived") * lit(100),2) as "Survival Rate").sort(desc("Survival Rate")).first()
```

- Result

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

(The ticket class is the pclass column. 1st = Upper; 2nd = Middle ; 3rd = Lower. Find the average cost of the ticket fare for the 1st class, the 2nd class and the 3rd class)

```
scala> castedDF.groupBy("Pclass").avg("Fare").sort("Pclass").show()
+------+------------------+
|Pclass|         avg(Fare)|
+------+------------------+
|     1| 84.15468749999992|
|     2| 20.66218315217391|
|     3|13.675550101832997|
+------+------------------+
```

---

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

(survival rate = number of survived per class/ total number of passengers . In the survival column, 0 = dead and 1 = survived)

```
scala> for ((c, rate) <- survivalRates) {
     |     println(s"$c\t\t%.2f".format(rate))
[    | }
1               15.26
2               9.76
3               13.36
```

- Class 1 has the highest survival rate of 15.26%

---

3) 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. (PS: if you watched the movie you will know if she survived or died)

```
scala> castedDF.filter(castedDF("Pclass")==="1" && castedDF("Age") === "17" && castedDF("Sex") === "female" && castedDF("Parch") === "1" && castedDF("Survived") === 1).count()
res9: Long = 0
```

- There are 0 passengers who could possibly be Rose

---

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)

```
scala> castedDF.repartition(col("PClass"), col("Sex")).filter(col("PClass") === "3" &&
     |           col("Sex") === "male" &&
     |           col("Parch") === "0" &&
     |           col("Survived") === "0" &&
     |           col("SibSp") === "0" &&
[    |           col("Age").between(19, 20)).count()
res47: Long = 22
```

- There are 22 passengers who could be Jack

---

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

What is the relation between the ages and the ticket fare? Which age group most likely survived ?

a.

Age Group	Average Fare
1-10	29.6
11-20	29.79
21-30	28.22
31-40	42.54
41-50	41.88
51-60	44.77
61-70	43.79
71-80	30.48
NULL / Zero	22.16

- The age group 31 to 70 spent more on ticket fares compared to other age groups

b.

```
scala> ageGroup.groupBy("Age Group").agg(round(avg("Survived") * lit(100),2) as "Survival Rate").sort(desc("Survival Rate")).first()
res31: org.apache.spark.sql.Row = [1-10,54.39]
```

- Age group 1-10 had the highest survival rate of 54.39%