Introduccion a PySpark

December 17, 2024

Diego Godinez Bravo

Centro de Investigación en Matemáticas

Maestría en Cómputo Estadístico

1 PySpark

1.1 PySpark Dataframe

[141]:
```python
import pyspark
import pandas as pd
from pyspark.sql import SparkSession
from pyspark.sql import functions as F
from pyspark.sql.functions import col, when, count # load libraries
```

[5]:
```python
spark = SparkSession.builder\
        .appName("Intro to Spark Dataframes")\
        .getOrCreate() # create a Spark session
```

[10]:
```python
spark # spark session I've created
```

[10]:
```
<pyspark.sql.session.SparkSession at 0x75e1f83aa000>
```

[19]:
```python
path = '/home/aspphem/Desktop/Statistics-with-Python/Cartwheeldata.csv' # file␣
 ↪path
df = spark.read.csv(path, header = True) # read a csv file
```

[23]:
```python
type(df) # pyspark dataframe object
```

[23]:
```
pyspark.sql.dataframe.DataFrame
```

[20]:
```python
df.printSchema() # print out the schema in tree format
```

```
root
 |-- ID: string (nullable = true)
 |-- Age: string (nullable = true)
 |-- Gender: string (nullable = true)
 |-- GenderGroup: string (nullable = true)
 |-- Glasses: string (nullable = true)
 |-- GlassesGroup: string (nullable = true)
 |-- Height: string (nullable = true)
 |-- Wingspan: string (nullable = true)
 |-- CWDistance: string (nullable = true)
 |-- Complete: string (nullable = true)
 |-- CompleteGroup: string (nullable = true)
 |-- Score: string (nullable = true)
```

[21]:
```
df = spark.read.option('header', 'true').csv(path, inferSchema = True) #␣
 ↪overwrite existing data frame and add inferSchema attribute
```

[22]:
```
df.printSchema() # print out the schema in tree format
```

```
root
 |-- ID: integer (nullable = true)
 |-- Age: integer (nullable = true)
 |-- Gender: string (nullable = true)
 |-- GenderGroup: integer (nullable = true)
 |-- Glasses: string (nullable = true)
 |-- GlassesGroup: integer (nullable = true)
 |-- Height: double (nullable = true)
 |-- Wingspan: double (nullable = true)
 |-- CWDistance: integer (nullable = true)
 |-- Complete: string (nullable = true)
 |-- CompleteGroup: integer (nullable = true)
 |-- Score: integer (nullable = true)
```

[64]:
```
print("Dataframe shape: {} rows, {} columns".format(df.count(), len(df.
 ↪columns))) # dataframe dimension
```

```
Dataframe shape: 25 rows, 12 columns
```

[13]:
```
df.columns # columns names
```

[13]:
```
['ID',
 'Age',
 'Gender',
 'GenderGroup',
 'Glasses',
 'GlassesGroup',
 'Height',
 'Wingspan',
 'CWDistance',
 'Complete',
 'CompleteGroup',
 'Score']
```

[118]:

```
df.show(5) # preview data
```

```
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
| ID|Age|Gender|GenderGroup|Glasses|GlassesGroup|Height|Wingspan|CWDistance|Comp
lete|CompleteGroup|Score|
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
|  1| 56|     F|          1|      Y|           1|  62.0|    61.0|        79|
Y|            1|    7|
|  2| 26|     F|          1|      Y|           1|  62.0|    60.0|        70|
Y|            1|    8|
|  3| 33|     F|          1|      Y|           1|  66.0|    64.0|        85|
Y|            1|    7|
|  4| 39|     F|          1|      N|           0|  64.0|    63.0|        87|
Y|            1|   10|
|  5| 27|     M|          2|      N|           0|  73.0|    75.0|        72|
N|            0|    4|
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
only showing top 5 rows
```

[172]:

```
updated_df = df.select('Gender', 'Height', 'CWDistance', 'Score')
updated_df.show(5) # selecting columns
```

```
+------+------+----------+-----+
|Gender|Height|CWDistance|Score|
+------+------+----------+-----+
|     F|  62.0|        79|    7|
|     F|  62.0|        70|    8|
|     F|  66.0|        85|    7|
|     F|  64.0|        87|   10|
|     M|  73.0|        72|    4|
+------+------+----------+-----+
only showing top 5 rows
```

[173]:

```
updated_df.describe().show() # dataframe summary
```

```
+-------+------+-----------------+------------------+------------------+
|summary|Gender|           Height|        CWDistance|             Score|
+-------+------+-----------------+------------------+------------------+
|  count|    25|               25|                25|                25|
|   mean|  NULL|            67.65|             82.48|               6.4|
| stddev|  NULL|4.431186823715139|15.058552387264852|2.5331140255951103|
|    min|     F|             61.5|                63|                 2|
|    max|     M|             75.0|               115|                10|
+-------+------+-----------------+------------------+------------------+
```

[174]:

```
updated_df = updated_df.withColumn('Height > 67', updated_df['Height'] >= 67)
updated_df.show(5) # adding columns in data frame
```

```
+------+------+----------+-----+-----------+
|Gender|Height|CWDistance|Score|Height > 67|
+------+------+----------+-----+-----------+
|     F|  62.0|        79|    7|      false|
|     F|  62.0|        70|    8|      false|
|     F|  66.0|        85|    7|      false|
|     F|  64.0|        87|   10|      false|
|     M|  73.0|        72|    4|       true|
+------+------+----------+-----+-----------+
only showing top 5 rows
```

[175]:

```
updated_df = updated_df.drop('Score')
updated_df.show(5) # drop the columns
```

```
+------+------+----------+-----------+
|Gender|Height|CWDistance|Height > 67|
+------+------+----------+-----------+
|     F|  62.0|        79|      false|
|     F|  62.0|        70|      false|
|     F|  66.0|        85|      false|
|     F|  64.0|        87|      false|
|     M|  73.0|        72|       true|
+------+------+----------+-----------+
only showing top 5 rows
```

[176]:

```
updated_df = updated_df.withColumnRenamed("CWDistance", "CartwheelDistance")
updated_df.show(5) # rename columns
```

```
+------+------+-----------------+-----------+
|Gender|Height|CartwheelDistance|Height > 67|
+------+------+-----------------+-----------+
|     F|  62.0|               79|      false|
|     F|  62.0|               70|      false|
|     F|  66.0|               85|      false|
|     F|  64.0|               87|      false|
|     M|  73.0|               72|       true|
+------+------+-----------------+-----------+
only showing top 5 rows
```

1.1.1 Filter Operations

```
[177]: df.show(5) # preview data
```

```
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
| ID|Age|Gender|GenderGroup|Glasses|GlassesGroup|Height|Wingspan|CWDistance|Comp
lete|CompleteGroup|Score|
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
|  1| 56|     F|          1|      Y|           1|  62.0|    61.0|        79|
Y|            1|    7|
|  2| 26|     F|          1|      Y|           1|  62.0|    60.0|        70|
Y|            1|    8|
|  3| 33|     F|          1|      Y|           1|  66.0|    64.0|        85|
Y|            1|    7|
|  4| 39|     F|          1|      N|           0|  64.0|    63.0|        87|
Y|            1|   10|
|  5| 27|     M|          2|      N|           0|  73.0|    75.0|        72|
N|            0|    4|
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
only showing top 5 rows
```

```
[178]: df.agg(F.mean('CWDistance')).collect()[0][0] # mean value of cartwheel distance
```

```
[178]: 82.48
```

```
[182]: df.filter('CWDistance<82').show(5) # cartwheel distance less than the mean␣
         ↪value
```

```
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
| ID|Age|Gender|GenderGroup|Glasses|GlassesGroup|Height|Wingspan|CWDistance|Comp
lete|CompleteGroup|Score|
+---+---+------+-----------+-------+------------+------+--------+----------+----
----+-------------+-----+
|  1| 56|     F|          1|      Y|           1|  62.0|    61.0|        79|
Y|            1|    7|
|  2| 26|     F|          1|      Y|           1|  62.0|    60.0|        70|
Y|            1|    8|
|  5| 27|     M|          2|      N|           0|  73.0|    75.0|        72|
N|            0|    4|
|  6| 24|     M|          2|      N|           0|  75.0|    71.0|        81|
N|            0|    3|
| 10| 33|     F|          1|      Y|           1|  63.0|    60.0|        65|
Y|            1|    8|
+---+---+------+-----------+-------+------------+------+--------+----------+----
```

```
----+-------------+------+
only showing top 5 rows
```

[188]:

```
df.filter(df['CWDistance']<82).select(['Age', 'Gender', 'CWDistance', 'Score']).
 ↪show(5) # select specific columns
```

```
+---+------+----------+-----+
|Age|Gender|CWDistance|Score|
+---+------+----------+-----+
| 56|     F|        79|    7|
| 26|     F|        70|    8|
| 27|     M|        72|    4|
| 24|     M|        81|    3|
| 33|     F|        65|    8|
+---+------+----------+-----+
only showing top 5 rows
```

[187]:

```
df.filter((df['CWDistance']<82) &
          (df['Age']<=30)).select(['Age', 'Gender', 'CWDistance', 'Score']).
 ↪show(5) # combine two specific conditions
```

```
+---+------+----------+-----+
|Age|Gender|CWDistance|Score|
+---+------+----------+-----+
| 26|     F|        70|    8|
| 27|     M|        72|    4|
| 24|     M|        81|    3|
| 28|     F|        79|   10|
| 23|     F|        66|    4|
+---+------+----------+-----+
only showing top 5 rows
```

[190]:

```
df.filter(~(df['CWDistance']<82)).select(['Age', 'Gender', 'CWDistance',␣
 ↪'Score']).show(5) # ~ not operator; anything that is greater than the mean␣
 ↪value will be given
```

```
+---+------+----------+-----+
|Age|Gender|CWDistance|Score|
+---+------+----------+-----+
| 33|     F|        85|    7|
| 39|     F|        87|   10|
| 28|     M|       107|   10|
| 22|     F|        98|    9|
| 29|     M|       106|    5|
+---+------+----------+-----+
only showing top 5 rows
```

1.2 PySpark Handling Missing Values

```
[156]: updated_df = df.select('Gender', 'Height', 'CWDistance', 'Score')
```

```
[157]: updated_df.select([count(when(col(c).isNull(), c)).alias(c) for c in updated_df.
         ↪columns]).show() # check for NULL values
```

```
+------+------+----------+-----+
|Gender|Height|CWDistance|Score|
+------+------+----------+-----+
|     0|     0|         0|    0|
+------+------+----------+-----+
```

```
[158]: updated_df = updated_df.replace({'F': None}, subset = ['Gender'])
       updated_df.show(5) # adding NULL values
```

```
+------+------+----------+-----+
|Gender|Height|CWDistance|Score|
+------+------+----------+-----+
|  NULL|  62.0|        79|    7|
|  NULL|  62.0|        70|    8|
|  NULL|  66.0|        85|    7|
|  NULL|  64.0|        87|   10|
|     M|  73.0|        72|    4|
+------+------+----------+-----+
only showing top 5 rows
```

```
[159]: print("Updated dataframe shape: {} rows, {} columns".format(updated_df.count(),␣
         ↪len(updated_df.columns))) # dataframe dimension
```

```
Updated dataframe shape: 25 rows, 4 columns
```

```
[160]: updated_df = updated_df.na.drop() # by default 'how = any' so it will drop a␣
         ↪row if it contains any nulls ('how = all' will drop a row only if all its␣
         ↪values are NULL)
       updated_df.show(5) # drop rows with NULL values
```

```
+------+------+----------+-----+
|Gender|Height|CWDistance|Score|
+------+------+----------+-----+
|     M|  73.0|        72|    4|
|     M|  75.0|        81|    3|
|     M|  75.0|       107|   10|
|     M|  74.0|       106|    5|
|     M|  69.5|        96|    6|
+------+------+----------+-----+
only showing top 5 rows
```

[161]:

```python
print("Updated dataframe shape: {} rows, {} columns".format(updated_df.count(), len(updated_df.columns))) # dataframe dimension
```

```
Updated dataframe shape: 13 rows, 4 columns
```

[163]:

```python
restored_df = df.select('Gender', 'Height', 'CWDistance', 'Score').replace({85: None}, subset = ['CWDistance'])
restored_df.show(5) # define a new dataframe with NULL values on it
```

```
+------+------+----------+-----+
|Gender|Height|CWDistance|Score|
+------+------+----------+-----+
|     F|  62.0|        79|    7|
|     F|  62.0|        70|    8|
|     F|  66.0|      NULL|    7|
|     F|  64.0|        87|   10|
|     M|  73.0|        72|    4|
+------+------+----------+-----+
only showing top 5 rows
```

[164]:

```python
restored_df.groupBy('CWDistance').count().orderBy(F.col("count").desc()).show() # get unique values and the no. of times each value appears
```

```
+----------+-----+
|CWDistance|count|
+----------+-----+
|      NULL|    2|
|        72|    2|
|        79|    2|
|        66|    2|
|        65|    1|
|       115|    1|
|       101|    1|
|        81|    1|
|        96|    1|
|        92|    1|
|        64|    1|
|       107|    1|
|        87|    1|
|        63|    1|
|        82|    1|
|        70|    1|
|        98|    1|
|        90|    1|
|       106|    1|
|        67|    1|
+----------+-----+
only showing top 20 rows
```

[152]:
```
restored_df.agg(F.mean('CWDistance')).collect()[0][0] # mean value of cartwheel distance
```

[152]:
```
82.26086956521739
```

[165]:
```
from pyspark.ml.feature import Imputer

imputer = Imputer(
    inputCols = ['CWDistance'],
    outputCols = ['{}_imputed'.format(c) for c in ['CWDistance']]
).setStrategy('mean') # create a new column with the NULL values of the specified column replaced by the mean value
```

[166]:
```
restored_df = imputer.fit(restored_df).transform(restored_df)
restored_df.show(5) # add imputation cols to df
```

```
+------+------+----------+-----+------------------+
|Gender|Height|CWDistance|Score|CWDistance_imputed|
+------+------+----------+-----+------------------+
|     F|  62.0|        79|    7|                79|
|     F|  62.0|        70|    8|                70|
|     F|  66.0|      NULL|    7|                82|
|     F|  64.0|        87|   10|                87|
|     M|  73.0|        72|    4|                72|
+------+------+----------+-----+------------------+
only showing top 5 rows
```

[191]:
```
spark.stop() # stop spark session
```