Anirudha Joshi – SEC01 (NUID 002991365)

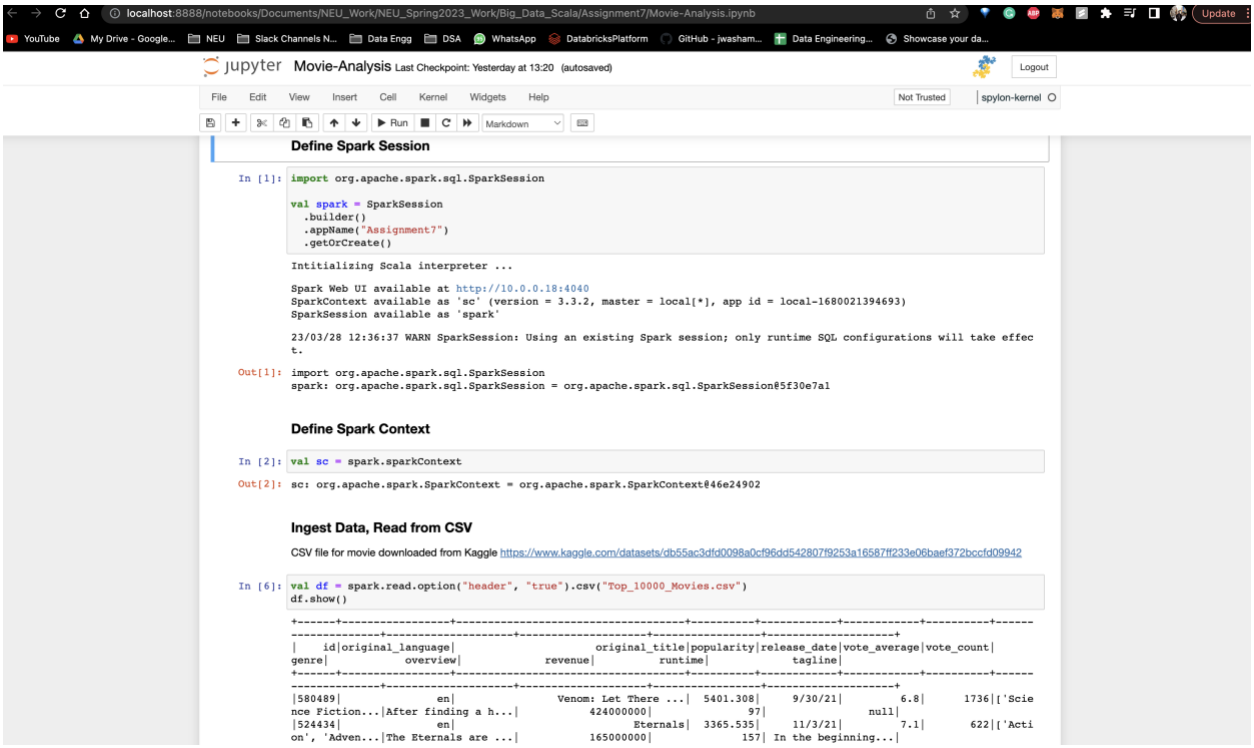
Big Data System Engineering with Scala
Spring 2023
Assignment No. 7 (Movie Analysis)

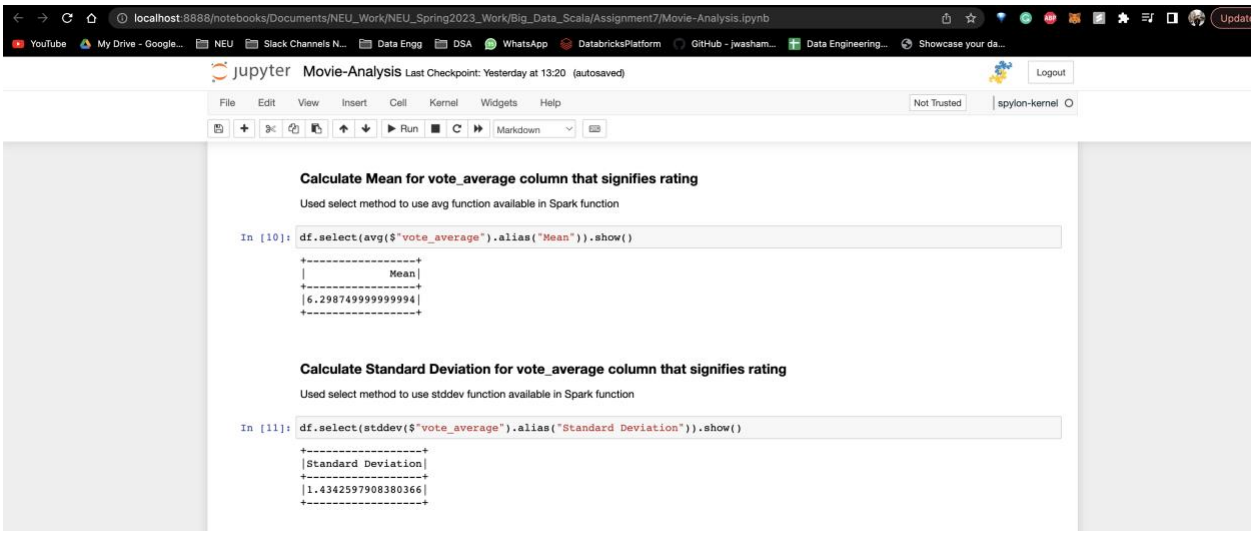
GitHub - https://github.com/anirudhajoshi2808/Anirudha_Joshi_CSYE7200

- Questions:

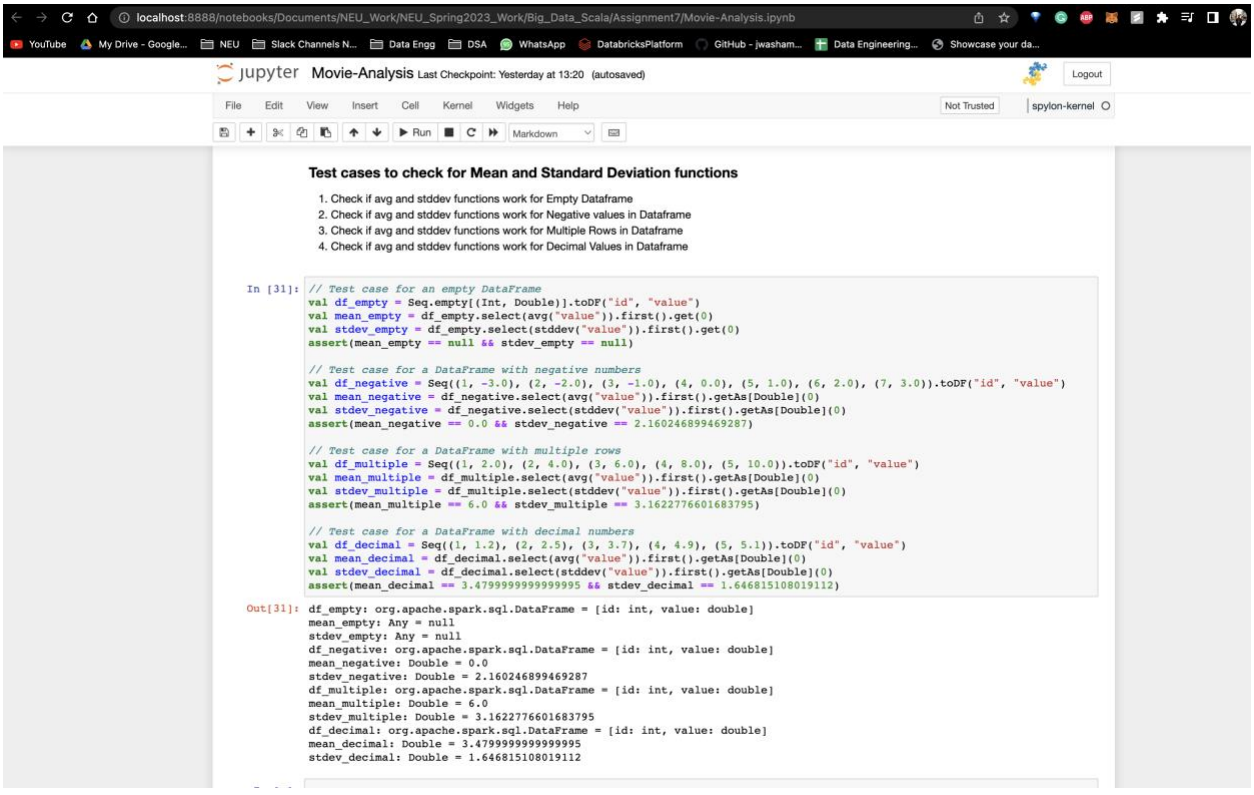
You are required to analyze a movie rating dataset. The data is stored in a CSV file (either use the one in the repository or download the latest from Kaggle). You need to read this file into spark and calculate the mean rating and standard deviation for all movies. There is no test case provided for you, so you need to write your own test cases to ensure that at least your program works well.

- Code




localhost:8888/notebooks/Documents/NEU_Work/NEU_Spring2023_Work/Big_Data_Scala/Assignment7/Movie-Analysis.ipynb
jupyter Movie-Analysis Last Checkpoint: Yesterday at 13:20 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
spylon-kernel
Run
Markdown
Calculate Mean for vote_average column that signifies rating
Used select method to use avg function available in Spark function
In [10]:
df.select(avg($"vote_average").alias("Mean")).show()
+------------------+
| Mean|
+------------------+
|6.298749999999994|
+------------------+
Calculate Standard Deviation for vote_average column that signifies rating
Used select method to use stddev function available in Spark function
In [11]:
df.select(stddev($"vote_average").alias("Standard Deviation")).show()
+------------------+
|Standard Deviation|
+------------------+
|1.4342597908380366|
+------------------+

- Unit tests


localhost:8888/notebooks/Documents/NEU_Work/NEU_Spring2023_Work/Big_Data_Scala/Assignment7/Movie-Analysis.ipynb
jupyter Movie-Analysis Last Checkpoint: Yesterday at 13:20 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
spylon-kernel
Run
Markdown
Test cases to check for Mean and Standard Deviation functions
1. Check if avg and stddev functions work for Empty Dataframe
2. Check if avg and stddev functions work for Negative values in Dataframe
3. Check if avg and stddev functions work for Multiple Rows in Dataframe
4. Check if avg and stddev functions work for Decimal Values in Dataframe
In [31]:
// Test case for an empty DataFrame
val df_empty = Seq.empty[(Int, Double)].toDF("id", "value")
val mean_empty = df_empty.select(avg("value")).first().get(0)
val stdev_empty = df_empty.select(stddev("value")).first().get(0)
assert(mean_empty == null && stdev_empty == null)
// Test case for a DataFrame with negative numbers
val df_negative = Seq((1, -3.0), (2, -2.0), (3, -1.0), (4, 0.0), (5, 1.0), (6, 2.0), (7, 3.0)).toDF("id", "value")
val mean_negative = df_negative.select(avg("value")).first().getAs[Double](0)
val stdev_negative = df_negative.select(stddev("value")).first().getAs[Double](0)
assert(mean_negative == 0.0 && stdev_negative == 2.160246899469287)
// Test case for a DataFrame with multiple rows
val df_multiple = Seq((1, 2.0), (2, 4.0), (3, 6.0), (4, 8.0), (5, 10.0)).toDF("id", "value")
val mean_multiple = df_multiple.select(avg("value")).first().getAs[Double](0)
val stdev_multiple = df_multiple.select(stddev("value")).first().getAs[Double](0)
assert(mean_multiple == 6.0 && stdev_multiple == 3.1622776601683795)
// Test case for a DataFrame with decimal numbers
val df_decimal = Seq((1, 1.2), (2, 2.5), (3, 3.7), (4, 4.9), (5, 5.1)).toDF("id", "value")
val mean_decimal = df_decimal.select(avg("value")).first().getAs[Double](0)
val stdev_decimal = df_decimal.select(stddev("value")).first().getAs[Double](0)
assert(mean_decimal == 3.4799999999999995 && stdev_decimal == 1.646815108019112)
Out[31]:
df_empty: org.apache.spark.sql.DataFrame = [id: int, value: double]
mean_empty: Any = null
stdev_empty: Any = null
df_negative: org.apache.spark.sql.DataFrame = [id: int, value: double]
mean_negative: Double = 0.0
stdev_negative: Double = 2.160246899469287
df_multiple: org.apache.spark.sql.DataFrame = [id: int, value: double]
mean_multiple: Double = 6.0
stdev_multiple: Double = 3.1622776601683795
df_decimal: org.apache.spark.sql.DataFrame = [id: int, value: double]
mean_decimal: Double = 3.4799999999999995
stdev_decimal: Double = 1.646815108019112