MovieAnalysis

November 8, 2023

[5]:
```
from pyspark.sql.functions import *
```

[8]:
```
## file path for datasets
filepath='/tmp/input_data/'
```

[9]:
```
## Loading movies dataset to movies datafframe
movies_df=spark.read\
            .format('csv')\
            .option('inferSchema', 'true')\
            .option('header', 'true')\
            .load(filepath+'movies.csv')
movies_df.show(10)
```

```
+-------+--------------------+--------------------+
|movieId|               title|              genres|
+-------+--------------------+--------------------+
|      1|    Toy Story (1995)|Adventure|Animati…|
|      2|      Jumanji (1995)|Adventure|Childre…|
|      3|Grumpier Old Men …|      Comedy|Romance|
|      4|Waiting to Exhale…|Comedy|Drama|Romance|
|      5|Father of the Bri…|              Comedy|
|      6|         Heat (1995)|Action|Crime|Thri…|
|      7|      Sabrina (1995)|      Comedy|Romance|
|      8| Tom and Huck (1995)|  Adventure|Children|
|      9| Sudden Death (1995)|              Action|
|     10|    GoldenEye (1995)|Action|Adventure|…|
+-------+--------------------+--------------------+
only showing top 10 rows
```

[10]:
```
## Loading ratings csc
ratings_df=spark.read\
            .format('csv')\
            .option('inferSchema', 'true')\
            .option('header', 'true')\
            .load(filepath+'ratings.csv')
ratings_df.show(10)
```

```
+------+-------+------+---------+
|userId|movieId|rating|timestamp|
+------+-------+------+---------+
|     1|      1|   4.0|964982703|
|     1|      3|   4.0|964981247|
|     1|      6|   4.0|964982224|
|     1|     47|   5.0|964983815|
|     1|     50|   5.0|964982931|
|     1|     70|   3.0|964982400|
|     1|    101|   5.0|964980868|
|     1|    110|   4.0|964982176|
|     1|    151|   5.0|964984041|
|     1|    157|   5.0|964984100|
+------+-------+------+---------+
only showing top 10 rows
```

[11]:
```
## loading tags dataset
tags_df=spark.read\
            .format('csv')\
            .option('inferSchema', 'true')\
            .option('header', 'true')\
            .load(filepath+'tags.csv')
tags_df.show(10)
```

```
+------+-------+----------------+----------+
|userId|movieId|             tag| timestamp|
+------+-------+----------------+----------+
|     2|  60756|           funny|1445714994|
|     2|  60756| Highly quotable|1445714996|
|     2|  60756|    will ferrell|1445714992|
|     2|  89774|    Boxing story|1445715207|
|     2|  89774|             MMA|1445715200|
|     2|  89774|       Tom Hardy|1445715205|
|     2| 106782|           drugs|1445715054|
|     2| 106782|Leonardo DiCaprio|1445715051|
|     2| 106782| Martin Scorsese|1445715056|
|     7|  48516|    way too long|1169687325|
+------+-------+----------------+----------+
only showing top 10 rows
```

[12]:
```
## fixing timestamp column with proper format

ratings_df1=ratings_df.withColumn("timestamp", from_unixtime(col("timestamp")))

ratings_df1.show(10)
```

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

```
|userId|movieId|rating|          timestamp|
+------+-------+------+-------------------+
|     1|      1|   4.0|2000-07-30 18:45:03|
|     1|      3|   4.0|2000-07-30 18:20:47|
|     1|      6|   4.0|2000-07-30 18:37:04|
|     1|     47|   5.0|2000-07-30 19:03:35|
|     1|     50|   5.0|2000-07-30 18:48:51|
|     1|     70|   3.0|2000-07-30 18:40:00|
|     1|    101|   5.0|2000-07-30 18:14:28|
|     1|    110|   4.0|2000-07-30 18:36:16|
|     1|    151|   5.0|2000-07-30 19:07:21|
|     1|    157|   5.0|2000-07-30 19:08:20|
+------+-------+------+-------------------+
only showing top 10 rows
```

[13]:
```
##Showing the aggregated number of ratings per year
ratings_per_year=ratings_df1.groupBy(substring("timestamp", 0, 4).
 ↪alias("year"))\
                    .agg(count("*").alias("count"))
ratings_per_year.show()
```

```
[Stage 13:>                                                         (0 + 1) / 1]
+----+-----+
|year|count|
+----+-----+
|2016| 6703|
|2012| 4656|
|2017| 8198|
|2014| 1439|
|2013| 1664|
|2005| 5813|
|2000|10061|
|2002| 3478|
|2009| 4158|
|2018| 6418|
|2006| 4059|
|2004| 3279|
|2011| 1690|
|2008| 4351|
|1999| 2439|
|1997| 1916|
|2007| 7114|
|1996| 6040|
|2015| 6616|
|1998|  507|
+----+-----+
```

```
only showing top 20 rows
```

[14]:
```
##Show the average monthly number of ratings

avg_monthly_ratings=ratings_df1.groupBy(substring("timestamp", 6, 2).
 ↪alias("month"))\
                                .agg(count("rating").alias("count_rating"))\
                                .orderBy("month")

avg_monthly_ratings.show()
```

```
[Stage 16:>                                                         (0 + 1) / 1]

+-----+------------+
|month|count_rating|
+-----+------------+
|   01|        8684|
|   02|        7635|
|   03|        8880|
|   04|        7727|
|   05|       10883|
|   06|        8825|
|   07|        6950|
|   08|        9074|
|   09|        8510|
|   10|        7148|
|   11|        9676|
|   12|        6844|
+-----+------------+
```

[16]:
```
##Show the rating levels distribution

rating_levels=ratings_df1.groupBy(col("rating").alias("rating_dist"))\
                         .agg(count("rating").alias("count"))\
                         .orderBy("rating_dist")

rating_levels.show(10)
```

```
[Stage 19:>                                                         (0 + 1) / 1]

+-----------+-----+
|rating_dist|count|
+-----------+-----+
|        0.5| 1370|
|        1.0| 2811|
```

```
|        1.5| 1791|
|        2.0| 7551|
|        2.5| 5550|
|        3.0|20047|
|        3.5|13136|
|        4.0|26818|
|        4.5| 8551|
|        5.0|13211|
+-----------+-----+
```

[18]:
```
tags_df1=tags_df.withColumn("timestamp", from_unixtime(col("timestamp")))

movies_df.show(5)
tags_df1.show(5)
ratings_df1.show(5)
```

```
+-------+--------------------+--------------------+
|movieId|               title|              genres|
+-------+--------------------+--------------------+
|      1|    Toy Story (1995)|Adventure|Animati…|
|      2|      Jumanji (1995)|Adventure|Childre…|
|      3|Grumpier Old Men …|      Comedy|Romance|
|      4|Waiting to Exhale…|Comedy|Drama|Romance|
|      5|Father of the Bri…|              Comedy|
+-------+--------------------+--------------------+
only showing top 5 rows

+------+-------+---------------+-------------------+
|userId|movieId|            tag|          timestamp|
+------+-------+---------------+-------------------+
|     2|  60756|          funny|2015-10-24 19:29:54|
|     2|  60756|Highly quotable|2015-10-24 19:29:56|
|     2|  60756|   will ferrell|2015-10-24 19:29:52|
|     2|  89774|   Boxing story|2015-10-24 19:33:27|
|     2|  89774|            MMA|2015-10-24 19:33:20|
+------+-------+---------------+-------------------+
only showing top 5 rows

+------+-------+------+-------------------+
|userId|movieId|rating|          timestamp|
+------+-------+------+-------------------+
|     1|      1|   4.0|2000-07-30 18:45:03|
|     1|      3|   4.0|2000-07-30 18:20:47|
|     1|      6|   4.0|2000-07-30 18:37:04|
|     1|     47|   5.0|2000-07-30 19:03:35|
|     1|     50|   5.0|2000-07-30 18:48:51|
```

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

[19]:
```
## Showing 18 movies that are tagged  butnot rated
tagged_not_rated=movies_df.join(tags_df, tags_df.movieId == movies_df.movieId,␣
 ↪'inner')\
                          .join(ratings_df, ratings_df.movieId == movies_df.
 ↪movieId, 'left')\
                          .filter(col("rating").isNull())\
                          .select("title").distinct()

print(tagged_not_rated.count())
tagged_not_rated.show()
```

```
18
+--------------------+
|               title|
+--------------------+
|Mutiny on the Bou…|
|Call Northside 77…|
|Color of Paradise…|
|For All Mankind (…|
|Browning Version,…|
|I Know Where I'm …|
|        Proof (1991)|
|Twentieth Century…|
|Innocents, The (1…|
|In the Realms of …|
|Parallax View, Th…|
|Road Home, The (W…|
|Roaring Twenties,…|
|  Chalet Girl (2011)|
|      Scrooge (1970)|
|      Niagara (1953)|
|  Chosen, The (1981)|
|This Gun for Hire…|
+--------------------+
```

[20]:
```
##Display movies that have rating but no tag

rated_not_tagged=movies_df.join(ratings_df, ratings_df.movieId == movies_df.
 ↪movieId, 'inner')\
                           .join(tags_df, tags_df.movieId == movies_df.
 ↪movieId, 'left')\
                           .filter(col("tag").isNull())
```

```
print(rated_not_tagged.count())
rated_not_tagged.select("title").distinct().show()
```

```
52549
+--------------------+
|               title|
+--------------------+
|Gulliver's Travel…|
|Before Night Fall…|
| Three Wishes (1995)|
| If Lucy Fell (1996)|
|First Blood (Ramb…|
|Don't Tell Mom th…|
| Nut Job, The (2014)|
|22 Jump Street (2…|
|Starship Troopers…|
|Voices from the L…|
|My Father the Her…|
|    Dead Meat (2004)|
|National Lampoon'…|
|7th Voyage of Sin…|
|     Ip Man 3 (2015)|
| Just Friends (2005)|
|I Love You Philli…|
|Tom Segura: Disgr…|
|    Fair Game (1995)|
|Problem Child (1990)|
+--------------------+
only showing top 20 rows
```

[21]:
```
#Focusing on the rated untagged movies with more than 30 user ratingshow the␣
 ↪top 10 movies in terms of average rating and number of ratings

top_10_rated_untagged=rated_not_tagged.groupBy(movies_df.movieId, "title")\
                                      .agg(avg("rating").alias("avg_rating"),
                                           count("rating").
 ↪alias("num_ratings"))\
                                      .orderBy(col("avg_rating").desc(),␣
 ↪col("num_ratings").desc())

top_10_rated_untagged.show(10)
```

```
[Stage 53:>                                                         (0 + 1) / 1]
+-------+--------------------+----------+-----------+
|movieId|               title|avg_rating|num_ratings|
+-------+--------------------+----------+-----------+
```

```
|  78836|Enter the Void (2…|       5.0|          2|
|     53|     Lamerica (1994)|       5.0|          2|
|   6442| Belle époque (1992)|       5.0|          2|
|   3473|Jonah Who Will Be…|       5.0|          2|
|     99|Heidi Fleiss: Hol…|       5.0|          2|
|   1151| Lesson Faust (1994)|       5.0|          2|
|   2512|Ballad of Narayam…|       5.0|          1|
| 136353|Scooby-Doo! and t…|       5.0|          1|
|   1631|Assignment, The (…|       5.0|          1|
| 130978|Love and Pigeons …|       5.0|          1|
+-------+--------------------+----------+-----------+
only showing top 10 rows
```

[22]:
```
#What is the average number of tags per movie in tagsDF? And the
#average number of tags per user?
#How does it compare with the average number of tags a user assigns to a movie?
total_tags=tags_df1.agg(count("tag").alias("count_tag")).
 ↪collect()[0]['count_tag']
print(total_tags)

no_of_movies=tags_df1.select("movieId").distinct().count()
print(no_of_movies)

avg_tags_per_movie= total_tags/no_of_movies
print(int(avg_tags_per_movie))
```

```
3683
1572
2
```

[23]:
```
#Identify the users that tagged movies without rating them
users_tagged_not_rated=movies_df.join(tags_df, tags_df.movieId == movies_df.
 ↪movieId, 'inner')\
                          .join(ratings_df, ratings_df.movieId == movies_df.
 ↪movieId, 'left')\
                          .filter(col("rating").isNull())\
                          .select(tags_df1.userId).distinct()

users_tagged_not_rated.show()
```

```
+------+
|userId|
+------+
|   474|
|   318|
|   543|
```

```
|   288|
+------+
```

[24]:
```
#What is the average number of ratings per user in ratings DF? And the average number of ratings per movie?
count_ratings=ratings_df1.agg(count("rating").alias("count_rating")).collect()[0]['count_rating']
print(count_ratings)
total_users=ratings_df1.select("userId").distinct().count()
print(total_users)

avg_ratings_per_user=count_ratings/total_users
print(avg_ratings_per_user)
```

```
100836
610
165.30491803278687
```

[25]:
```
# What is the predominant (frequency based) genre per rating level?

from pyspark.sql.window import Window

joined_df=ratings_df.join(movies_df, ratings_df.movieId == movies_df.movieId,'inner')
exploded_df = joined_df.withColumn("genre", explode(split("genres", "\|")))
grouped_df = exploded_df.groupBy("rating", "genre").count()

window=Window.partitionBy("rating").orderBy(col("count").desc())
ranked_df=grouped_df.withColumn("rank", rank().over(window)).filter(col("rank")== 1).orderBy(col("rating").desc())

ranked_df.select("rating", "genre").show()
```

```
[Stage 80:>                                                 (0 + 1) / 1]

+------+------+
|rating| genre|
+------+------+
|   5.0| Drama|
|   4.5| Drama|
|   4.0| Drama|
|   3.5| Drama|
|   3.0|Comedy|
|   2.5|Comedy|
|   2.0|Comedy|
|   1.5|Comedy|
|   1.0|Comedy|
|   0.5|Comedy|
```

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

[26]:
```
#What is the predominant tag per genre and the most tagged genres?
joined_df=tags_df.join(movies_df, tags_df.movieId == movies_df.movieId, 'inner')
exploded_df=joined_df.withColumn("genre", explode(split("genres", "\|")))
grouped_df=exploded_df.groupBy("genre", "tag").count()

window=Window.partitionBy("genre").orderBy(desc("count"))
ranked_df=grouped_df.withColumn("rank", rank().over(window)).filter(col("rank")␣
 ↪== 1)

ranked_df.select("genre", "tag").groupBy("genre").agg(collect_list("tag")).
 ↪show(10)
```

```
+------------------+--------------------+
|             genre|   collect_list(tag)|
+------------------+--------------------+
|(no genres listed)|[quirky, understa…|
|            Action|         [superhero]|
|         Adventure|         [superhero]|
|         Animation|            [Disney]|
|          Children|            [Disney]|
|            Comedy|  [In Netflix queue]|
|             Crime|  [In Netflix queue]|
|       Documentary|  [In Netflix queue]|
|             Drama|  [In Netflix queue]|
|           Fantasy|            [Disney]|
+------------------+--------------------+
only showing top 10 rows
```

[27]:
```
#What are the most predominant (popularity based) movies?
predominant_df=movies_df.join(ratings_df, ratings_df.movieId == movies_df.
 ↪movieId, 'inner')\
                        .groupBy("title").count()\
                        .orderBy(desc("count"))

predominant_df.show(10)
```

```
+--------------------+-----+
|               title|count|
+--------------------+-----+
| Forrest Gump (1994)|  329|
|Shawshank Redempt…|  317|
| Pulp Fiction (1994)|  307|
|Silence of the La…|  279|
```

```
|  Matrix, The (1999)|  278|
|Star Wars: Episod…|  251|
|Jurassic Park (1993)|  238|
|   Braveheart (1995)|  237|
|Terminator 2: Jud…|  224|
|Schindler's List …|  220|
+--------------------+-----+
only showing top 10 rows
```

[27]:
```
#Top 10 movies in terms of average rating (provided more than 30 users reviewed␣
 ↪them)
joined_df==movies_df.join(ratings_df, ratings_df.movieId == movies_df.movieId,␣
 ↪'inner')
grouped_df=joined_df.groupBy("title").agg(avg("rating").alias("avg_rating"),␣
 ↪count("rating").alias("count_ratings"))
filtered_df=grouped_df.filter(col("count_ratings") > 30).
 ↪orderBy(desc("avg_rating"))

filtered_df.show(5)
```

```
+--------------------+-----+
|               title|count|
+--------------------+-----+
| Forrest Gump (1994)|  329|
|Shawshank Redempt…|  317|
| Pulp Fiction (1994)|  307|
|Silence of the La…|  279|
|  Matrix, The (1999)|  278|
|Star Wars: Episod…|  251|
|Jurassic Park (1993)|  238|
|   Braveheart (1995)|  237|
|Terminator 2: Jud…|  224|
|Schindler's List …|  220|
+--------------------+-----+
only showing top 10 rows
```

[ ]:
```

```