Analysis of IMDb Movie Data using an AWS EMR Cluster

By: Jonathan Glasgow

Introduction:

IMDb is the world's most popular source of information when it comes to movies, TV, streaming, and any other sort of visual entertainment of similar mediums. The company has information on nearly every movie or show ever made. Such information includes: the title, region, language, type of medium, if the movie is an adult movie, what year it begins and ends, the length of each episode or movie, the genre, rating information, and much more. If there's anything you need to know about a movie or show, IMDb has it. The purpose of this project is to take a look at that massive pile of information, which contains thousands upon thousands of rows of data, and narrow it down to a more manageable and current scope. Specifically, to analyze the data from all movies released during 2021 in the US, that also had at least 1,000 ratings.

Implementation of Cluster and Data:

Using Amazon Web Services' EMR and S3 environments, I set up a cluster to store the data in, and then used spark and python code to sort and perform work on the data itself. The steps used to set up the cluster are listed in the file: Hadoop Multi-Node Cluster Setup with AWS.

To get the data, I downloaded three files from: https://datasets.imdbws.com/.
The data downloaded is included in the zip file, they include:

- title.ratings.tsv
- title.basics.tsv
- title.akas.tsv

All of IMDb's datasets and information on each can be found here:
https://www.imdb.com/interfaces/

Dataset Descriptions:
In the following will be all the information I used from the datasets downloaded. Some information the datasets included were not used and were filtered out. Such information will not be listed, but can be found at the link above pertaining to IMDb's interfaces.

title.akas.tsv:

- titleId (string) - alphanumeric unique identifier of the title
- region (string) - the region for this version of the title

title.basics.tsv:

- tconst (string) - alphanumeric unique identifier of the title
- titleType (string) – the type/format of the title (e.g. movie, short, tvseries, tvepisode, video, etc)
- primaryTitle (string) – the more popular title / the title used by the filmmakers on promotional materials at the point of release
- startYear (YYYY) – represents the release year of a title. In the case of TV Series, it is the series start year
- genres (string array) – includes up to three genres associated with the title

title.ratings.tsv:

- tconst (string) - alphanumeric unique identifier of the title
- averageRating – weighted average of all the individual user ratings
- numVotes - number of votes the title has received

Implementation of Python Code:

To filter and sort the data, the following code was implemented in a file named **final-project.py**:

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
from pyspark.sql.functions import desc
import pyspark.sql.functions as F

S3_DATA_SOURCE_PATH_1 = 's3://movie-data-bucket-1234/data-source/basics-data.tsv'
S3_DATA_SOURCE_PATH_2 = 's3://movie-data-bucket-1234/data-source/rating-data.tsv'
S3_DATA_SOURCE_PATH_3 = 's3://movie-data-bucket-1234/data-source/akas-data.tsv'
S3_DATA_OUTPUT_PATH = 's3://movie-data-bucket-1234/data-source/movie-data-output-final'


def main():
    spark = SparkSession.builder.appName('SparkApp').getOrCreate()
    basicsData = spark.read.csv(S3_DATA_SOURCE_PATH_1, sep=r'\t', header=True)
    ratingData = spark.read.csv(S3_DATA_SOURCE_PATH_2, sep=r'\t', header=True)
    akasData = spark.read.csv(S3_DATA_SOURCE_PATH_3, sep=r'\t', header=True)
    sB = basicsData.where((col('startYear') == 2021) & (col('titleType') == 'movie'))
    sR = ratingData.where((col('numVotes') > 1000))
    sA = akasData.where((col('region')) == "US")


    joinedS1 = sB.join(sR, ["tconst"], 'inner')
    joinedS2 = joinedS1.withColumn("tconst", col("tconst")).join(sA.withColumn("tconst", col("titleId")), on="tconst")


    selectedS = joinedS2.select("primaryTitle", "runtimeMinutes", "genres", "averageRating", "numVotes")
    selectedS.write.mode('overwrite').csv(S3_DATA_OUTPUT_PATH, sep=r'\t')
    print('Selected data was successfully saved to s3: %s' % S3_DATA_OUTPUT_PATH)

if __name__ == '__main__':
    main()
```

The path to the data in S3 is listed as the variables **S3_DATA_SOURCE_PATH_1** through 3, with each representing their respective file as shown. From basics-data.tsv, I took all the information from the titles with the type of “movie” that were released in 2021. I then filtered through **rating-data.tsv** for votes **exceeding 1000**, so that any review average would hold some actual merit. Finally, I filtered through **akas-data.tsv** to get all the rows containing the country “US”, to get all of the movies released in the United States.

The three filtered data sets, **sB**, **sR**, and **sA**, were then joined together by their **titleID: “tconst”.** The **akas data set** had a different name for its title ID than the other two sets, which were named “tconst”, so the join statement was different for that one to account for the difference in names.

After the filtration and joining of data sets was complete, I then made sure to further simplify the joined set by selecting only the information absolutely needed and creating another set. This process helps make it easier to analyze the important data. The columns: “primaryTitle”, “runtimeMinutes”, “genres”, “averageRating”, and “numVotes”, were assigned to the new set: “selectedS”. The new dataset was then uploaded to S3 as an output of multiple csv files. I merged these files together in excel, deleted any

duplicate data due to the nature of the storage, and then analyzed the remaining data. This data is named **Movie Data Final Rated** and is included in the zip.

Project Testing, Compilation, and Execution:

The entire project was tested by using VSCode to create the python program, and PuTTY to ssh into the AWS EMR cluster to compile and run the python code on the data stored in S3. Some interesting hurdles of the project included:

- Figuring out how to access and work with the data stored in S3
- Joining the tables without losing important information
- Selecting the relevant information, and properly outputting it all to a useful format

Analysis of Data and Results:

From all the movies released in the US, with at least 1000 ratings:

Top 10 Worst Rated Movies in 2021:

Title	Length	Genre	Average Rating	Number of Votes
The Cost of Deception	125	Crime,Drama,History	1.5	36642
Radhe	135	Action,Crime,Thriller	1.8	173920
Cosmic Sin	88	Action,Adventure,Sci-Fi	2.5	11173
Vanquish	94	Thriller	2.7	3546
Karen	89	Crime,Drama,Thriller	2.8	1761
15/07 Safak Vakti	95	Action,Drama,War	2.8	20362
The Resort	75	Horror	3	2074
Apex	94	Action,Thriller	3.1	2599
Level	62	Comedy,Fantasy	3.1	1268
Hungama 2	156	Comedy	3.1	5939

Only 3/10 movies exceeded 100 minutes in length. With a few outliers, shorter movies may be received in poorer light than longer ones.

Top 10 Best Rated Movies in 2021:

Title	Length	Genre	Average Rating	Number of Votes
Jayanti	144	Drama	9.9	1275
Ooriki Uttharana	122	Drama	9.8	1045
Kapata Nataka Sutradhari	116	Thriller	9.7	1922
Jai Bhim	164	Crime,Drama	9.5	150391
Methagu	100	Biography,History	9.5	8594
Maanaadu	147	Action,Adventure,Sci-Fi	9.4	12637
Ravana Lanka	\N	Action,Crime,Thriller	9.3	1140
Secrets of Sinauli	56	Documentary,History	9.3	1665
Ram Asur	132	Action	9.3	1101
Natyam	136	Drama	9.1	1058

Some of the titles are not created in the US, but because they're still released in it and to streaming services, they are still added to the database with the tag, and thus, this dataset. It also may be important to note that 9/10 of the movies shown are 100 or more minutes in length.

Top 10 Most Popular Movies in 2021:

Title	Length	Genre	Average Rating	Number of Votes
Dune	155	Action,Adventure,Drama	8.2	365155
Zack Snyder's Justice League	242	Action,Adventure,Fantasy	8.1	347946
Black Widow	134	Action,Adventure,Sci-Fi	6.7	290429
The Suicide Squad	132	Action,Adventure,Comedy	7.3	256918
Shang-Chi and the Legend of the Ten Rings	132	Action,Adventure,Fantasy	7.6	227641
No Time to Die	163	Action,Adventure,Thriller	7.4	222527
Free Guy	115	Action,Adventure,Comedy	7.2	221826
Cruella	134	Adventure,Comedy,Crime	7.4	193318
Godzilla vs. Kong	113	Action,Sci-Fi,Thriller	6.4	182108
The Tomorrow War	138	Action,Adventure,Drama	6.6	178875

Dune is not only the most popular movie of the year so far, but also the best rated out of the Top 10. This is a clear indication to anyone viewing the data that the movie was extremely successful. To point on the previous table's point as well, 10/10 movies in this table exceed 100 minutes in length, and all movies received an average rating above 6.6. The most popular movies are generally longer, and rated higher than others.

Summary:

By utilizing IMDb's data and AWS' EMR cluster, I was able to successfully store large amounts of data, filter important information, and create new datasets to use for analysis. This data and system can be used in many different ways, but for this example it proved useful in showcasing the popularity, and lack thereof, of movies released in the US in 2021.