Kristianstad University
SE-291 88 Kristianstad
+46 44-250 30 00
www.hkr.se

Big Data Project:

Data Analysis of IMDb Data

Ahmed Radwan & Jwan Mardini

Content

Introduction

In this project, we analyzed a large IMDb movie dataset using PySpark, a powerful framework for distributed data processing [1]. The objective was to uncover trends in the movie industry, focusing on key aspects such as production patterns over the years and the release trends of different genres. By utilizing PySpark's scalability and efficiency, we processed the dataset to extract meaningful insights from a vast amount of data.

Our analysis began with a deep dive into the annual production trends, where we examined the number of films released each year, isolating the timeframe between 1990 and 2024. This involved data cleaning, renaming columns for clarity, type conversions, and filtering the dataset to focus on relevant time periods. We extended the analysis to study the release trends of the top 10 movie genres over a narrower timeframe, from 2000 to 2024. This analysis provided insights into the volume of films released in each genre over time, revealing how the industry has shifted its focus across different genres.

This report highlights the methodologies employed, the challenges encountered, and the findings of our analysis. By leveraging PySpark, we demonstrate how modern big data tools can be applied to efficiently process and analyze large datasets, providing valuable insights into the evolution of the film industry.

Method

Dataset Considered

The project utilizes IMDB datasets in TSV (Tab-Separated Values) format, which contain detailed metadata about movies, including their titles, genres, release years, and ratings. The datasets collectively amount to approximately **9 GB**, making them suitable for distributed processing. These datasets were ingested into PySpark DataFrames to leverage the distributed computing power of Apache Spark for processing large-scale data efficiently [2].

Data Preprocessing

1. **For Yearly Movie Distribution Analysis**:

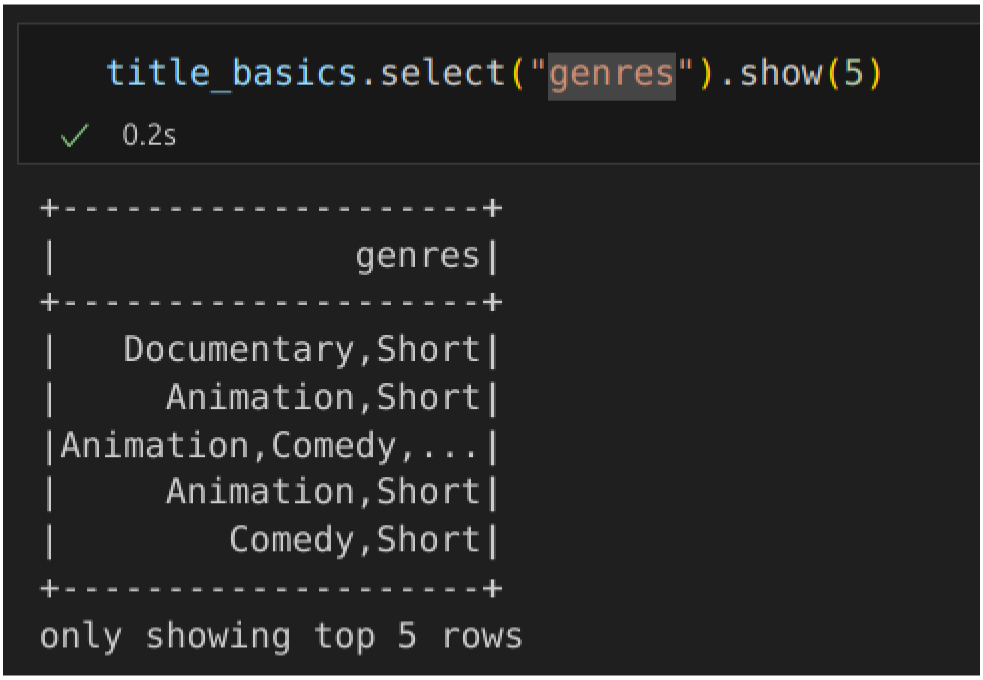
   Data Cleaning:

   a. Null or missing values in the startYear column were removed. Only movies with valid release years were retained.
   b. **Filtering Relevant Data**: Movies released between 1990 and 2024 were selected by filtering the startYear column.

c. **Aggregation**: The dataset was grouped by the startYear column, and the count of movies for each year was calculated using PySpark's groupBy and count operations.
d. **Transformation**: The resulting DataFrame was sorted in descending order to identify trends and peaks in movie production.

2. **For Genre Trends Analysis (2000–2024)**:

a. **Exploding Genre Data**: Since many movies had multiple genres listed in a single cell (e.g., "Action, Comedy"), the genres column was exploded into multiple rows, with each row representing one movie-genre pair. This allowed for accurate analysis of genre trends.



```
title_basics.select("genres").show(5)
✓ 0.2s

+--------------------+
|              genres|
+--------------------+
|   Documentary,Short|
|     Animation,Short|
|Animation,Comedy,...|
|     Animation,Short|
|        Comedy,Short|
+--------------------+
only showing top 5 rows
```

Figure 1: A Screenshot showing multiple genres for a movie

b. **Filtering Data**: The analysis focused on movies released between 2000 and 2024. This range was chosen to capture modern trends in the film industry.
c. **Aggregation**: The data was grouped by startYear and genres, and the count of movies in each genre per year was calculated.
d. **Selection of Top Genres**: The top 10 genres with the highest total movie counts were identified, and only these genres were retained for trend visualization.

Data Analysis

- For **Yearly Movie Distribution**, the results highlight the number of movies released per year between 1990 and 2024, providing insights into the growth of the film industry.

- For **Genre Trends Analysis**, the yearly distribution of the top 10 genres between 2000 and 2024 was visualized to uncover trends in movie production across different genres.

Working Mechanism and Flowchart

Mechanism and Workflow:

1. **Data Loading**: The datasets were loaded into PySpark DataFrames for parallel processing.
2. **Data Cleaning**: Null values and irrelevant rows were filtered out.
3. **Data Aggregation**: Movies were grouped by year and genre, and their counts were computed.
4. **Visualization**: Trends in yearly movie releases and genres were visualized using Matplotlib and Plotly.

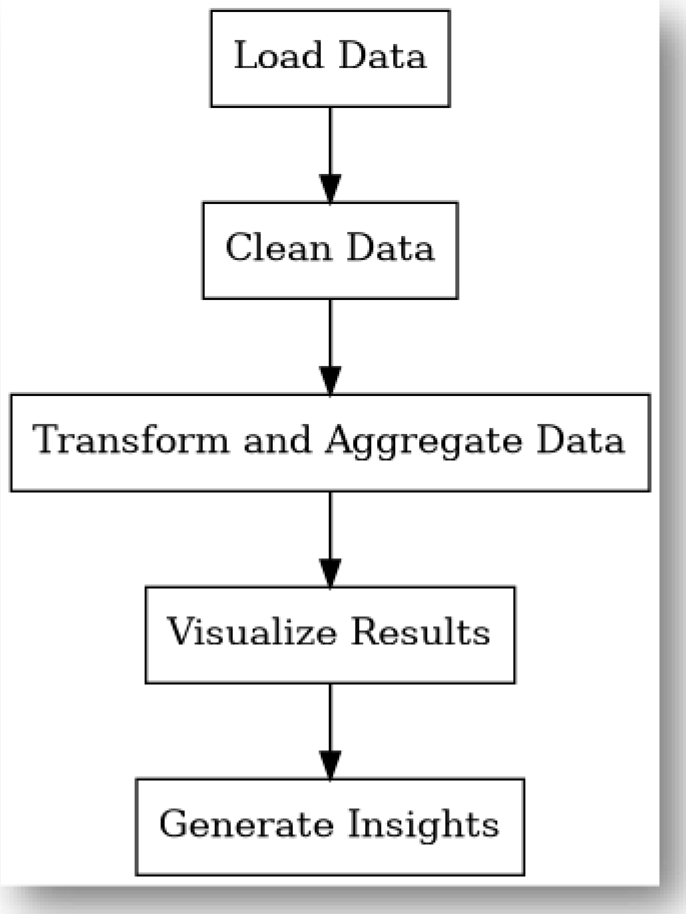


Figure 2: The flowchart of our workflow

Results

1. Yearly Distribution of Movies

- From 1990 to 2024, there has been a consistent increase in the number of movies released annually. Peaks in production were observed in recent years, reflecting the growth of the global film industry.

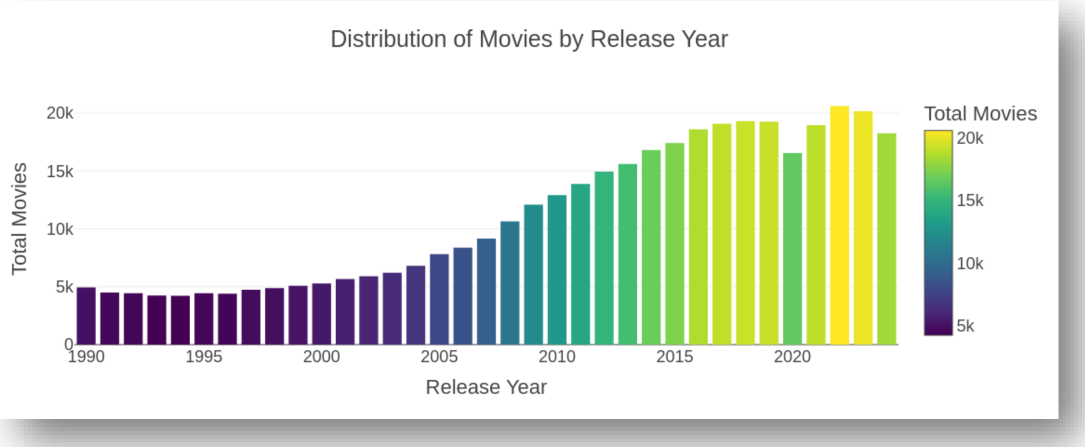


Figure 3: The Yearly Distribution of Movies

2. Genre Trends (2000–2024)

- Action, Drama, and Comedy emerged as the top genres. Drama experienced the highest consistency in releases, while genres like Horror and Thriller showed periodic surges.

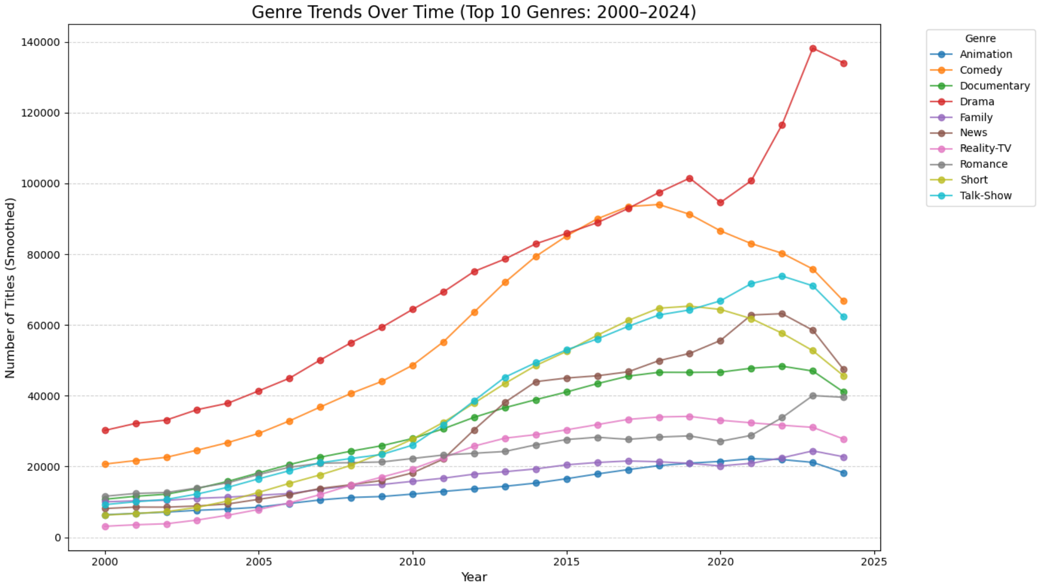


Figure 4: Genre Trends Over Time

Evaluation of Performance and Learning

- This project provided valuable experience in handling and analyzing large datasets using distributed computing tools. Key contributions include efficient preprocessing and meaningful visualization of trends. The use of PySpark allowed for seamless processing of large data volumes, which would have been computationally expensive on traditional systems.

- From this project, I gained proficiency in PySpark, including operations like data cleaning, aggregation, and visualization. Additionally, this project emphasized the importance of storytelling through data visualization, a crucial skill in data science.

Visualization

The following visualizations were created:

1. **Top 10 Most Prolific Directors**: A horizontal bar plot was created to show the number of movies each of the top 10 directors has worked on.
2. **Top 10 Most Prolific Actors**: Another bar plot was used to visualize the most prolific actors, displaying the number of movies associated with each actor.
3. **Genre Distribution**: A pie chart was used to show the distribution of genres in the dataset.

Security

To enhance visibility into file operations, we enabled DEBUG-level logs for HDFS. This allows detailed tracking of read and write actions, which is critical for monitoring, auditing, and identifying potential security issues.

We modified the `log4j.properties` file with the following entries:

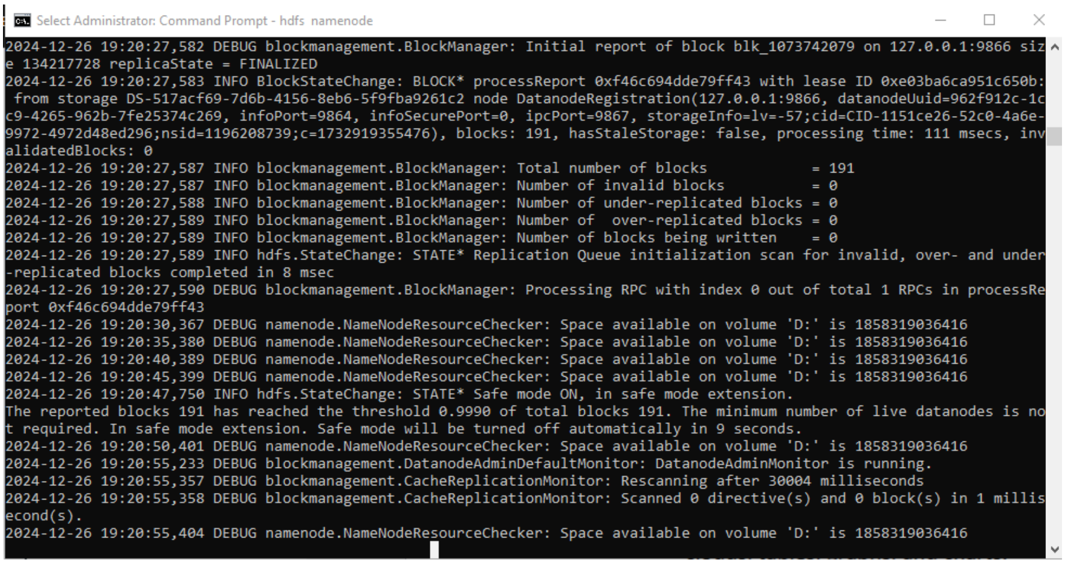
“

```
log4j.logger.org.apache.hadoop.hdfs=DEBUG, console

log4j.additivity.org.apache.hadoop.hdfs=false
```

“

This configuration ensures that HDFS operations are logged in the console with detailed information. After applying the changes and restarting Hadoop services, the logs provided comprehensive insights into HDFS activity [3].



```
2024-12-26 19:20:27,582 DEBUG blockmanagement.BlockManager: Initial report of block blk_1073742079 on 127.0.0.1:9866 siz
e 134217728 replicaState = FINALIZED
2024-12-26 19:20:27,583 INFO BlockStateChange: BLOCK* processReport 0xf46c694dde79ff43 with lease ID 0xe03ba6ca951c650b:
 from storage DS-517acf69-7d6b-4156-8eb6-5f9fba9261c2 node DatanodeRegistration(127.0.0.1:9866, datanodeUuid=962f912c-1c
c9-4265-962b-7fe25374c269, infoPort=9864, infoSecurePort=0, ipcPort=9867, storageInfo=lv=-57;cid=CID-1151ce26-52c0-4a6e-
9972-4972d48ed296;nsid=1196208739;c=1732919355476), blocks: 191, hasStaleStorage: false, processing time: 111 msecs, inv
alidatedBlocks: 0
2024-12-26 19:20:27,587 INFO blockmanagement.BlockManager: Total number of blocks            = 191
2024-12-26 19:20:27,587 INFO blockmanagement.BlockManager: Number of invalid blocks          = 0
2024-12-26 19:20:27,588 INFO blockmanagement.BlockManager: Number of under-replicated blocks = 0
2024-12-26 19:20:27,589 INFO blockmanagement.BlockManager: Number of  over-replicated blocks = 0
2024-12-26 19:20:27,589 INFO blockmanagement.BlockManager: Number of blocks being written    = 0
2024-12-26 19:20:27,589 INFO hdfs.StateChange: STATE* Replication Queue initialization scan for invalid, over- and under
-replicated blocks completed in 8 msec
2024-12-26 19:20:27,590 DEBUG blockmanagement.BlockManager: Processing RPC with index 0 out of total 1 RPCs in processRe
port 0xf46c694dde79ff43
2024-12-26 19:20:30,367 DEBUG namenode.NameNodeResourceChecker: Space available on volume 'D:' is 1858319036416
2024-12-26 19:20:35,380 DEBUG namenode.NameNodeResourceChecker: Space available on volume 'D:' is 1858319036416
2024-12-26 19:20:40,389 DEBUG namenode.NameNodeResourceChecker: Space available on volume 'D:' is 1858319036416
2024-12-26 19:20:45,399 DEBUG namenode.NameNodeResourceChecker: Space available on volume 'D:' is 1858319036416
2024-12-26 19:20:47,750 INFO hdfs.StateChange: STATE* Safe mode ON, in safe mode extension.
The reported blocks 191 has reached the threshold 0.9990 of total blocks 191. The minimum number of live datanodes is no
t required. In safe mode extension. Safe mode will be turned off automatically in 9 seconds.
2024-12-26 19:20:50,401 DEBUG namenode.NameNodeResourceChecker: Space available on volume 'D:' is 1858319036416
2024-12-26 19:20:55,233 DEBUG blockmanagement.DatanodeAdminDefaultMonitor: DatanodeAdminMonitor is running.
2024-12-26 19:20:55,357 DEBUG blockmanagement.CacheReplicationMonitor: Rescanning after 30004 milliseconds
2024-12-26 19:20:55,358 DEBUG blockmanagement.CacheReplicationMonitor: Scanned 0 directive(s) and 0 block(s) in 1 millis
econd(s).
2024-12-26 19:20:55,404 DEBUG namenode.NameNodeResourceChecker: Space available on volume 'D:' is 1858319036416
```

Figure 5: A screenshot showing the logging process

Security Implications

While DEBUG logs improve monitoring and troubleshooting, they can expose sensitive information. To mitigate risks in production, logs should be properly secured, rotated, and restricted to authorized personnel.

Conclusion

From the analysis, we concluded that the IMDb dataset offers valuable insights into the film industry, particularly in terms of prolific directors and actors. The results were not only interesting but also highlighted some interesting relationships, such as the prominence of certain genres like "Drama" and "Comedy". Despite challenges with data size and processing times, the use of Apache Spark allowed us to handle the dataset efficiently.

For future improvements, data processing time can be reduced by optimizing the Spark jobs and reducing unnecessary data storage during intermediate steps. Moreover, using more advanced filtering techniques could improve the relevance of the words in the word clouds and visualizations.

References

1. Quick Start - Spark 3.5.4 Documentation [Internet]. Apache.org. 2024 [cited 2024 Dec 28]. Available from: https://spark.apache.org/docs/latest/quick-start.html

2. Imdb.com. 2024 [cited 2024 Dec 28]. Available from: https://developer.imdb.com/non-commercial-datasets/

3. Apache Hadoop 3.4.1 – Hadoop: Setting up a Single Node Cluster. [Internet]. Apache.org. 2024 [cited 2024 Dec 28]. Available from: https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html