Movie exercise with MapReduce

Michael Munatsi 628114

<https://github.com/munatsimike/movie-exercise-with-mapreduce.git>

**Introduction**

The document contains all the steps taken to sort movies and genres based on rating using Hadoop MapReduce. The script has two classes: SortMovie and SortGenre for sorting movies and genres. Movies are from the ml-100k dataset.

**Joining tables**

The first step was to join the rating and item files using the Jupyter notebook. The result was one table with the following columns;

1. User id,
2. Movie id,
3. Movie title
4. Rating
5. A column for each genre

**SortMovie class**

The second step was to create a class (SortMovie) that counts and sorts movies using the merged table. The class has one mapper and two reducers.

**First mapper**

The mapper splits each row into an array and creates key-value pairs. Movie id and name is the key, and 1 is the value for each rating.

**First reducer**

The first reducer receives key-value pairs from the first mapper and counts ratings for each unique movie

**Second reducer**

The second reducer sorts the movies by rating and yields the results.

**Testing the script**

First, the script was tested using the command line locally and displayed movies sorted in alphabetical order. Next was to use Hadoop, and below are the steps.

1. Upload the python script and movie.csv file to HDFS using Ambari.
2. Connect to HDFS using putty.
3. Copy script and movile.csv from hdfs to the local file system.
4. Run the script.

The results are similar to those obtained when the script was tested locally. Refer to fig1 and fig2 below.

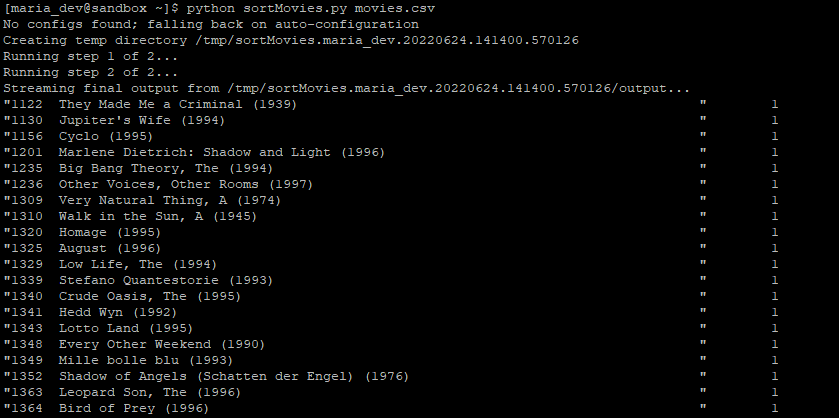


Fig 1 top movies

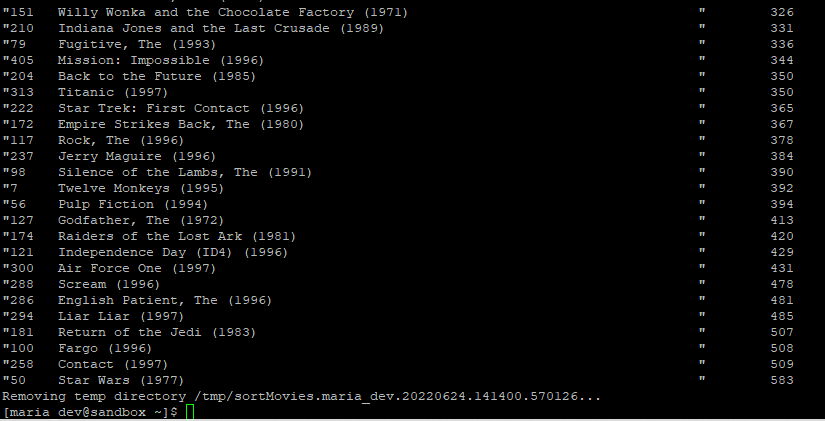


Fig 2 bottom movies

**SortGenre class**

This chapter contains steps for sorting genres by total rating.

The class has two mappers and two reducers, and the course uses the same movie.csv file.

**First Mapper**

The first mappers split the row, check the movie genre and create a key-value pair. Genre is the key and 1 for the rating.

The class uses a dictionary that contains all genres. The genre keys in the dictionary correspond to the column indexes in the split row. The column index is used to get the genre in the dictionary if a genre is matched.

**First Reducer**

The first reducer receives the key-value pairs from the first mapper and counts ratings for each unique genre.

**Second Mapper.**

The second mapper converts the ratings to integers. Ratings came out as strings from the first reducer, causing an error when trying to sort them.

**Second reducer**

The second reducer sorts the genres by their total ratings and yields the results.

**Testing the script**

Testing the script is similar to the one discussed in the sorting movie section. Fig 3 shows the results.

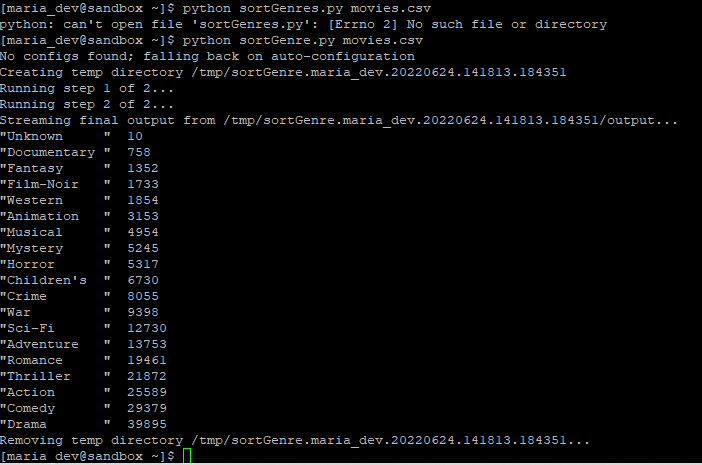


Fig 3