Report assignment

This report describes the recommendation system assignment which is part of the assessment procedure in the Database Management and Digital Tools module. The purpose of the assignment is to make recomendations of movies based on the starring, title and the summary of the movie.

This report is an interpretation of the output that is generated from the SQL commands that was runned on a Raspberry Pi Model B. The SQL commands converted the results into a CSV file, all the files can be accessed via the following GitHub Repository:
https://github.com/wtr08/recommendation_system.git

Favorite movies

As instructed in the booklet, these are my favorite movies that were used in this assignment.

- Interstellar
- The Wolf of wallstreet
- I Am Legend (Not really but some of my fav movies were not in the data set)

Results

Beneath, the results are described by, the aforementioned, three types of search methods: Title, starring and summary.

Summary Results

The summary words that are converted into a list of lexemes seems to have a big impact in the quantity of the results. However, the quality of the results are dubious. The quality of the recommendations differ per movie. Take for example the movie Interstellar, the recommendations set from interstellar have a lot of movies that had high ranking. However, the movies that are on this dataset is not close to space or interstellar movies. I expected the Martian movie, that is included in the Metacritic data is, but unfortunately is nowhere to be found in the recommendations.

Title Results

The quantity of results is a lot less than summary. And the ranking is also a lot lower, because the threshold was decreased to get at least a decent amount of results. The natural language processing in just title only makes sense if there are a lot of words in the title. For example interstellar, I am Legend, and especially the wolf of wall street, the words are not the well used across that contains it in other movie titles.

Starring results

The quantity in results is really low and the threshold is also decreased in a lot of places. However, the movie The wolf Of Wall street has a lot of movies in it that is starred by Leonardo Dicaprio. Unlike the other movies, the wolf of wall street shows a lot of quality results.

Overall

In conclusion, the number of words that in that specific column is really important of the quantity of results. The less words you can use, the less results you will get. This is because there is less frequency of words in for example Starring and title, so it was expected to have a low results. Personally, I think this is a good starting point for creating a recommendation system, but there needs to be a lot of adjustments to make it a good recommendation system.

Python - Results

The exercise for the Python assignment was to loop through the **userReviews.csv** that has 203.515 records. In the favorite movie **I Am Legend** (Specified as: Specific movie) we need to scan for individuals reviews (specified as: Authors). We check on these reviews if the Authors have reviewed other movies **higher** than the *specific movie*. All these movies will be added into a new dataset called *recommendations*. Eventually, the recommendations dataset will be exported to a csv-file.

At the end, my result with the *specific movie* is that we still have a large dataset with 4.984 records. This is for the end-user an overload on movies and therefore not desirable. So it would be great to have a sort of ranking system, like the SQL assignment.

Furthermore, the cause of the big dataset can have a big load time to finish the process and gather all the recommendations. Therefore better iteration techniques need to be specified.

Normal loop - Slow

```
data = # dataset import
for row in range(0, len(data)):
  # code
```

Iterrows() — Better, but still slow

The iterrows funtion (Stands for itterate for each row or the well know FOR EACH loop) is a function that Pandas provide to loop through a dataframe. It is much faster than a for loop. But this is still not fast enough for me

```
data = # dataset import
for i, r in data.iterrows():
  # code
```

Itertuples() — Best

Because itterrows() stores everything in a Series list. Ittertuples() makes everything an object. So it is much faster. This is used in filter2.py

```
data = # dataset import
for r in data.itertuples(): # code
```

Recommendations

The idea of "I like this movie, but this one was even better" is a pretty interesting idea. However, the following problems occur when reading the results:

- Movies that are not related to the current movie. For example comparing a horror movie with a comedy movie is comparing apples with oranges.
- Author can review the specific movie really low, so every other movie (even when the other movie is rated with a score of 2) will be in the

dataset.

- Every Author that leaves a review is implemented in the dataset. Authors with a summary attached and have better count score as better credibility.

In conclusion, the above problems can be converted into new criteria's to reduce the dataset and have a better result in recommendations.