MOVIE RECOMMENDATION SYSTEM

By

DANIEL ADETUNJI

dgadetunji@gmail.com

Recommendation systems predict user preference and ratings of an item. They are one of the most popular applications used in data science. Various companies make use of them to make suggestion based on their clients' preferences, to help them meet their needs. Media companies like Netflix, make use of recommendation systems to help users decide which movie or documentary they would enjoy watching. Other product based companies like Amazon, utilize this system in order to suggest products for customers.

It turns out that there are (mostly) three ways to build a recommendation engine:

1. **Popularity based recommendation engine**

This is popularly referred to as the simplest kind of recommendation engine that you will come across. Companies like Netflix and YouTube make use of this algorithm to to make trending lists. For each movie or video, it records the number of views and subsequently lists them in descending order (from views with the most frequency to lowest number of views).

2. **Content based recommendation engine**

Netflix makes use of this type of recommendation when you initially set up your account (movies in 'Your List'). This type of recommendation systems, takes in a movie that a user currently likes as input; it analyzes the contents (storyline, genre, cast, director etc.) of the movie to find out other movies which have similar content. Then it ranks similar movies according to their similarity scores and finally recommends the most relevant movies to the user.

3. Collaborative filtering based recommendation engine

This algorithm at first tries to find similar users based on their activities and preferences (for example, both the users watch same type of movies or movies directed by the same director). Now, between these users (say, A and B) if user A has seen a movie that user B has not seen yet, then that movie gets recommended to user B and vice-versa. In other words, the recommendations get filtered based on the collaboration between similar user's preferences (thus, the name "Collaborative Filtering"). One typical application of this algorithm can be seen in the Amazon e-commerce platform, where you get to see the "Customers who viewed this item also viewed" and "Customers who bought this item also bought" list.

The algorithm used in here will be solely based on the content-based recommendation system.

- Importing the libraries
- Reading the data
- Renaming and cleaning the dataset
- Importing Scikit-learn's Tfidfvectorizer. Tfidfvectorizer aims to convert a collection of raw documents to a matrix of TF-IDF features. The TF stands for "Term frequency", while the IDF stands for "Inverse Data Frequency".
- Fitting the TF-IDF on the 'overview' text.

  It should be noted that, this is dtype object; hence, conversion to Unicode string is necessary. The "astype('U').values" function makes it possible to transform text data into Unicode.

- Importing scikit-Learn's sigmoid kernel
- Computing the sigmoid kernel
- Reverse mapping of indices and movie titles
- Defining a function "give_recommendation" that returns 10 movies that are similar to the one requested. It involves the following;
  - Getting the index corresponding to original title
  - Getting the pairwise similarity scores
  - Sorting the movies
  - Obtaining the scores of the 10 most similar movies
  - Indexing the movies
  - Finally, returning the Top 10 most similar movies