Recommender Systems: Accurate Recommendation

Swansea University
Prifysgol Abertawe

Alistair Grom – 964398 – Swansea University | Supervisor – Dr Siyuan Liu

This project is a Content-based Recommender System that utilizes Natural Language Processing and Machine Learning to compare and find similarities in the content and metadata of movies.

Users will be able to enter their favourite movies and receive recommendations on what to watch based on the content of those films.

Aims

- To create a recommender algorithm that can accurately recommend relevant content to users.
- Use a **content-based** approach to reduce dependence on others preferences to generate **your** recommended content.
- Provide an interface for users to enter in their own choices and receive their recommendations based on the movies they like.

Motivations

Recommender Systems are a very powerful tool and are being utilised across all our content consumption platforms, Streaming services, Social Media and News sites.

When metrics outside that of just the content alone are used to recommend similar content to users recommendations become less personal and can reflect popularity and other biases not the actual similarity of the content itself.

This is the motivation behind my approach of a **Content-based Recommender System** to recommend movies to users.

My Approach to Recommendation

Data

- I am using the **IMDb dataset** which contains 46K movies containing data about:
  - director
  - overview
  - genre
  - cast

  From these metrics I have formed a 'bag of words' to get a more explanatory view of a movie based upon these above features. This being as detailed as possible is very important for a Content-based system.

Feature Extraction

- To extract meaning from these movie descriptions I have used the following **Natural Language Processing** methods.
  - **TD-IDF:** Term Frequency by Inverse Document Frequency (First Iteration)
  - **CountVectorizer** (Second Iteration)

Finding Similarities

- To compare similarities I have used a **cosine similarity function**.
- This compares the output of the Feature Extraction to find the movies that are most similar and therefore will be recommended to the user.
- We are given a matrix with the similarity of movies in the dataset to the one being compared. Higher similarity = higher decimal value.

```
[[1.         0.09534626 0.1        ... 0.12909944 0.1        0.        ]
 [0.09534626 1.         0.         ... 0.         0.09534626 0.        ]
 [0.1        0.         1.         ... 0.12909944 0.1        0.11952286]
 ...
 [0.12909944 0.         0.12909944 ... 1.         0.12909944 0.        ]
 [0.1        0.09534626 0.1        ... 0.12909944 1.         0.        ]
 [0.         0.         0.11952286 ... 0.         0.         1.        ]]
```

[A screenshot of my 2nd cosine similarity function's output]

Implementation

- The dataset to drive my algorithm is a sample of the top 20% most voted movies in the IMDb dataset.
- This project has been implemented using Python3 and libraries from sklearn and pandas.
- Jupyter notebook has been used to code the algorithm display the data to the user.
- The User is able to search for movies in the database.

```
Selection
Enter a movie (type 'done' when finished adding movies):
The Shawshank Redemption
```

- The user will then receive recommendations based on their choices, both based on all selections and individual movies.

```
-----------------------------------------------------------

Because you liked The Shawshank Redemption
                      title                     genres
                    Witness   [crime, drama, romance]
                  Dark Blue    [action, crime, drama]
              Gone Baby Gone  [crime, drama, mystery]
             Le Cercle Rouge [drama, thriller, crime]
            Dead Man Walking                  [drama]
                  Carandiru                   [drama]
         Murder in the First           [crime, drama]
        The Hudsucker Proxy           [comedy, drama]
                 GoodFellas            [drama, crime]
 Once Upon a Time in America           [drama, crime]

-----------------------------------------------------------

Because you liked Pulp Fiction
               title                      genres
               Fresh   [crime, drama, thriller]
   Reasonable Doubt           [crime, thriller]
             Cleaner [crime, thriller, mystery]
  Kill Bill: Vol. 2  [action, crime, thriller]
      A Time to Kill   [crime, drama, thriller]
        Jackie Brown   [comedy, crime, romance]
            S.W.A.T.  [action, thriller, crime]
         Freedomland   [drama, thriller, crime]
    Lakeview Terrace   [drama, crime, thriller]
   The Hateful Eight    [crime, drama, mystery]

-----------------------------------------------------------
```

```
-----------------------------------------------------------

Recommendations for You based on User Profile
                       title                      genres
                Pulp Fiction          [thriller, crime]
    The Shawshank Redemption             [drama, crime]
                       Fresh   [crime, drama, thriller]
          Out of the Furnace   [thriller, drama, crime]
              A Time to Kill   [crime, drama, thriller]
             Le Cercle Rouge   [drama, thriller, crime]
                 Freedomland   [drama, thriller, crime]
            Lakeview Terrace   [drama, crime, thriller]
           The Hateful Eight    [crime, drama, mystery]
                Meeting Evil    [crime, drama, mystery]
            Reasonable Doubt          [crime, thriller]
               Kiss of Death     [action, crime, drama]
                     Cleaner [crime, thriller, mystery]
                  The Forger   [thriller, crime, drama]
      Leon: The Professional   [thriller, crime, drama]

-----------------------------------------------------------
```

Results

- A successful Content-based recommender system was created.
- In my opinion this system provides users with an alternate style of recommendations of movies that many large scale recommenders do not offer.

Future Work

- Implement an API or web app for this system.
- Apply this algorithm to a music dataset.
- Distribute to users and get feedback on how to tweak the weights of what gives the most optimal recommendations.