Implementation of Content-Based Movie Recommendation System with Sentiment Analysis Based on Machine Learning.

Yash Kulkarni, Karan Bhat, Priti Narke, Prajwal Prince, P.M Tekade.
Information Technology department, JSPM's RSCOE Pune

ABSTRACT

In this digital era, there are endless varieties of content to be consumed like books, videos, articles, movies, etc., it has become a hectic task to find the content of one's liking. Whereas, the digital content providers want to get as many users on their service as possible for the maximum duration. Here the recommender system comes into play by which the content providers recommend users the stuff according to the users' liking and preferences.
In this paper, we have proposed a movie recommender system, "Movie Guide". The goal of Movie Guide is to provide the precise movie recommendations to users.

INTRODUCTION

In today's world, the internet has become an important part of human life. The excessive available information often creates a complication for the user. To help users cope with this information explosion Recommendation systems (RS) are deployed. RS is mostly used in e-commerce applications and knowledge management systems such as tourism, entertainment, and online shopping portals. Our sole focus in this paper is on RS for movies which are an integral source of recreation and entertainment in our lives. Web-based portals are responsible for movie suggestions to users. Genres of movies like comedy, thriller, animation, and action are the easiest category to differentiate them. Another possible way to categorize movies can be achieved on the basis of metadata such as year, language, director, or cast. Most online video-streaming services provide a number of similar movies to the user to utilize the user's previously viewed or rated history. Movie Recommendation Systems help us to search for our preferred movies and also reduce the trouble of spending a lot of time searching for favorable movies. The primary requirement of a movie recommendation system is that it should be very reliable and provide the user with the recommendation of movies that are similar to their preferences. In recent times, with an exponential increase in the amount of online data, RS is very beneficial for taking decisions in different activities of day-to-day life. RS is broadly classified into two categories: Collaborative filtering (CF) and Content-based filtering (CBF). Humans often tend to make decisions on the basis of facts, predetermined rules, and familiar information which is accessible over the internet and this predisposition of human behavior gave rise to the notion of CF. The idea of CF was introduced by Resnick et al. in net news, to help people find articles they liked in an enormous stream of available articles. CF emphasizes on helping people to make choices based on the standpoint of other people. Two users are considered like-minded when their ratings for items are similar whereas, in CBF, items are suggested through the similarity among the contextual information of the items. With the emergence of social media platforms like Quora, Facebook, Instagram, Twitter and many more similar social media services which authorize people to express their daily state of mind over the internet. Twitter is one of the most popular social network founded in 2006 where users can express their thoughts and emotions in limited characters. The Distinctive Selling Proposition of Twitter is that the users not only receive information based on their social links but also gain access to other user-specified information. "Tweets" are known as the source of information on Twitter which are of a limited character that keep users up to date on their best-loved topics, people, and movies. For the purpose of the present report, we propose a movie recommendation framework by amalgamating hybrid and sentiment scores from the database. The major offerings of the paper are as follows: 1. We are proposing a hybrid recommendation system by fusing collaborative filtering and content-based filtering. 2. Sentiment analysis is used to boost this recommendation system. 3. A detailed analysis of the proposed recommendation system is presented through extensive experiments. Eventually, a qualitative, as well as quantitative comparison with other standard models, is also shown.

RELATED STUDIES

1. CONTENT BASED FILTERING (CBF)

A Content-Based movie recommendation system utilizes the data provided by users like ratings, feedback, and reviews. To make recommendation to the user, a user profile is created using the user's previous data. As the user takes more actions or provides more inputs on the recommendation system, the engine becomes more precise and robust. To implement a content-based filtering system following steps are to be followed:

- Terms Allocation
- Terms Representation
- Learning Algorithm Selection
- Provider Recommendation

2. CALCULATION OF SIMILARITY (CS)

Calculating the similarity between movies is the goal of content-based recommender systems. The content can be anything example; text, video, and image. In our project, each movie is represented by an attribute vector. Cosine Similarity is the most used measurement for document similarity. To calculate the similarity between two attributes, we have to calculate the cosine of the angle between the attribute vector using the given Equation: $\cos(m,n) = m.n/||m||*||n||$
Where, m.n= product (dot) of the vectors 'm' & 'n'. $||m||$ & $||n||$= length of the two vectors $||m||*||n||$= cross product of the two vectors.

3.SENTIMENT ANALYSIS (SA)

Sentiment analysis is one of the Natural Language Processing fields which is committed to the judgement of subjective opinions, views, or feelings collected from a collection of sources about a specific subject. In more accurate business terms, it can be encapsulated as "Sentiment Analysis is a set of tools to pinpoint and extract opinions and utilize them for the benefit of the business operation". Such algorithms push deep into the text and find the substance those points out the feature towards the result in regular or its specific element.
The calculation is performed by: -

P (+ve | altogether liking of the movie) = P (altogether liking of the movie | +ve) * P (+ve) / P (altogether liking of the movie).

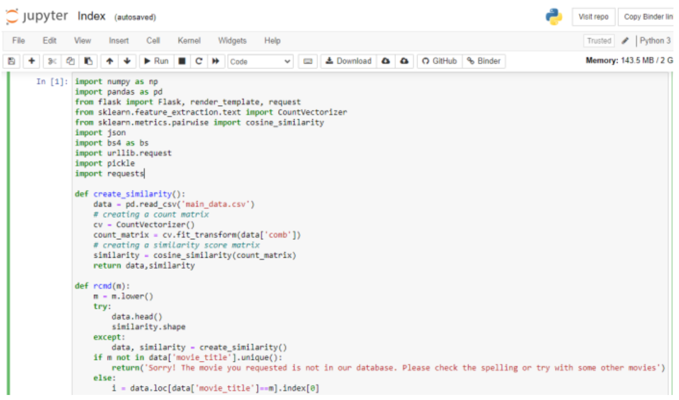
IMPLEMENTATION OF THE SYSTEM

In this paper, we put forward the Content Based Movie Recommendation using Sentiment Analysis “MOVIE GUIDE”. Movie Guide is an application which provides all the details of the requested movie such as overview, genre, release date, rating, runtime, top cast, reviews, recommended movies, etc.The details of the movies(title, genre, runtime, rating, poster, etc) are fetched using an API by TMDB named https://www.themoviedb.org/documentation/api, and using the IMDB id of the movie in the API, web scraping is done to get the reviews given by the user in the IMDB site using beautifulsoup4 beautifulsoup4 and performed sentiment analysis on those reviews. With the help of API keys all the info about the actor director and movies is fetched. Based on user reviews the sentiment is being predicted whether the movie is good or bad, top 10 reviews are displayed. Then based on those reviews top 10 similar movies will be recommended. To carry out sentiment analysis NLTK i.e. Natural Language Toolkit and TfidVectorizer is used. After that stopwords is applied to remove unnecessary English words which are not needed. Use of TfidVectorizer and fit transform is to convert the reviews and comments data into vectors to create NLP model. Algorithm used in the NLP model is MultinomialNB(). It calculates each tag's likelihood for a given sample and outputs the tag with the greatest chance.

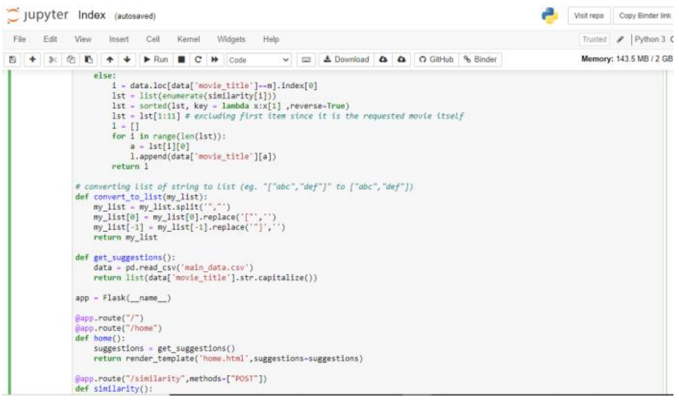
Finally similarity score is used to recommend the movies based on current genre.

How does it decide which item is most similar to the item user likes? Here is the use of similarity scores.

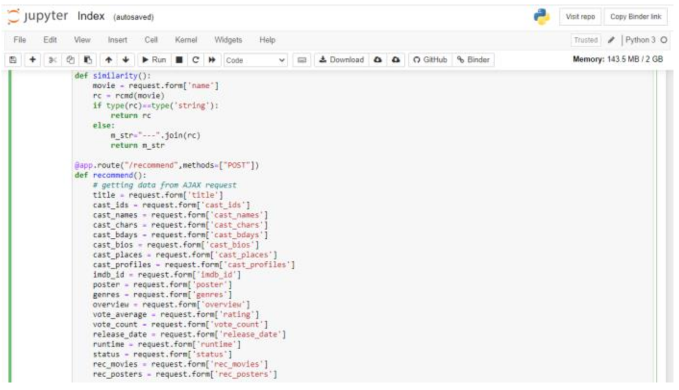
It is a numerical value ranges between zero to one which helps to determine how much two items are similar to each other on a scale of zero to one. This similarity score is obtained measuring the similarity between the text details of both of the items. So, similarity score is the measure of similarity between given text details of two items. This can be done by cosine-similarity.



```
In [1]: import numpy as np
        import pandas as pd
        from flask import Flask, render_template, request
        from sklearn.feature_extraction.text import CountVectorizer
        from sklearn.metrics.pairwise import cosine_similarity
        import json
        import bs4 as bs
        import urllib.request
        import pickle
        import requests

        def create_similarity():
            data = pd.read_csv('main_data.csv')
            # creating a count matrix
            cv = CountVectorizer()
            count_matrix = cv.fit_transform(data['comb'])
            # creating a similarity score matrix
            similarity = cosine_similarity(count_matrix)
            return data,similarity

        def rcmd(m):
            m = m.lower()
            try:
                data.head()
                similarity.shape
            except:
                data, similarity = create_similarity()
            if m not in data['movie_title'].unique():
                return('Sorry! The movie you requested is not in our database. Please check the spelling or try with some other movies')
            else:
                i = data.loc[data['movie_title']==m].index[0]
```



```
    else:
        i = data.loc[data['movie_title']==m].index[0]
        lst = list(enumerate(similarity[i]))
        lst = sorted(lst, key = lambda x:x[1] ,reverse=True)
        lst = lst[1:11] # excluding first item since it is the requested movie itself
        l = []
        for i in range(len(lst)):
            a = lst[i][0]
            l.append(data['movie_title'][a])
        return l

# converting list of string to list (eg. "["abc","def"]" to ["abc","def"])
def convert_to_list(my_list):
    my_list = my_list.split('","')
    my_list[0] = my_list[0].replace('["','')
    my_list[-1] = my_list[-1].replace('"]','')
    return my_list

def get_suggestions():
    data = pd.read_csv('main_data.csv')
    return list(data['movie_title'].str.capitalize())

app = Flask(__name__)

@app.route("/")
@app.route("/home")
def home():
    suggestions = get_suggestions()
    return render_template('home.html',suggestions=suggestions)

@app.route("/similarity",methods=["POST"])
def similarity():
```

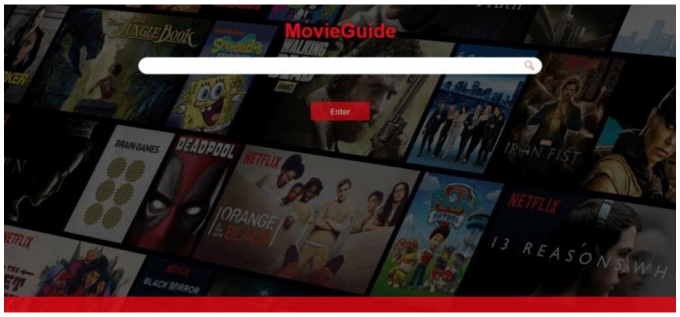


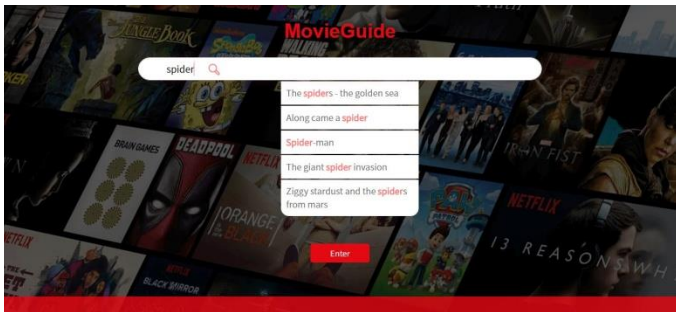
```
def similarity():
    movie = request.form['name']
    rc = rcmd(movie)
    if type(rc)==type('string'):
        return rc
    else:
        m_str="---".join(rc)
        return m_str

@app.route("/recommend",methods=["POST"])
def recommend():
    # getting data from AJAX request
    title = request.form['title']
    cast_ids = request.form['cast_ids']
    cast_names = request.form['cast_names']
    cast_chars = request.form['cast_chars']
    cast_bdays = request.form['cast_bdays']
    cast_bios = request.form['cast_bios']
    cast_places = request.form['cast_places']
    cast_profiles = request.form['cast_profiles']
    imdb_id = request.form['imdb_id']
    poster = request.form['poster']
    genres = request.form['genres']
    overview = request.form['overview']
    vote_average = request.form['rating']
    vote_count = request.form['vote_count']
    release_date = request.form['release_date']
    runtime = request.form['runtime']
    status = request.form['status']
    rec_movies = request.form['rec_movies']
    rec_posters = request.form['rec_posters']
```

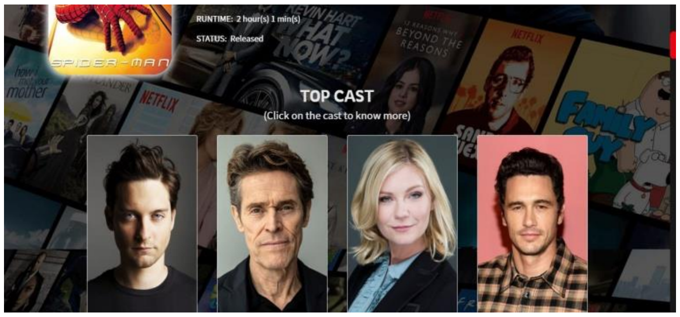
CONCLUSION

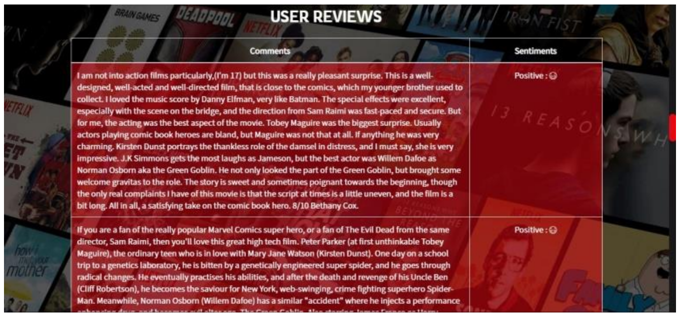
In this project, we have selected Content Based Filtering with Sentiment Analysis to develop a Movie Recommendation System. Content-based filtering methods become an advantage when dealing with a new user whereas these systems also have some drawbacks. The collaborative filtering methods are segregated into two parts in which, neighborhood methods are used to recommend plain content but they are unable to provide accuracy, and model-based methods improve the status of cold-start problems. Collaborative filtering systems are very popular because they have many booms. Hybrid systems overcome the drawbacks of both content-based and collaborative filtering systems, improve the outcome and make the system precise. Because of the information overload Recommender system has become more and more important. We attempt to find a new way to improve the accuracy of the representative of the movie, for content-based recommender system. There is no cold statement problem as we are using content-based recommender system. But it cannot recommend anything on its own, it entirely depends on users' previous interactions with the system. These shortcomings can be addressed by Collaborative Filtering technique which we will try to introduce in future. For now, we are able to work on building a content-based movie recommendation system with the help of certain technologies and some guidance.

RESULT



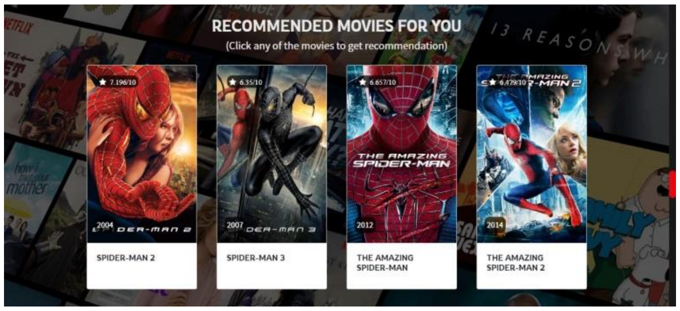

MovieGuide
spider
The spiders - the golden sea
Along came a spider
Spider-man
The giant spider invasion
Ziggy stardust and the spiders from mars
Enter





USER REVIEWS

Comments	Sentiments
I am not into action films particularly,(I'm 17) but this was a really pleasant surprise. This is a well-designed, well-acted and well-directed film, that is close to the comics, which my younger brother used to collect. I loved the music score by Danny Elfman, very like Batman. The special effects were excellent, especially with the scene on the bridge, and the direction from Sam Raimi was fast-paced and secure. But for me, the acting was the best aspect of the movie. Tobey Maguire was the biggest surprise. Usually actors playing comic book heroes are bland, but Maguire was not that at all. If anything he was very charming. Kirsten Dunst portrays the thankless role of the damsel in distress, and I must say, she is very impressive. J.K Simmons gets the most laughs as Jameson, but the best actor was Willem Dafoe as Norman Osborn aka the Green Goblin. He not only looked the part of the Green Goblin, but brought some welcome gravitas to the role. The story is sweet and sometimes poignant towards the beginning, though the only real complaints I have of this movie is that the script at times is a little uneven, and the film is a bit long. All in all, a satisfying take on the comic book hero. 8/10 Bethany Cox.	Positive : 😊
If you are a fan of the really popular Marvel Comics super hero, or a fan of The Evil Dead from the same director, Sam Raimi, then you'll love this great high tech film. Peter Parker (at first unthinkable Tobey Maguire), the ordinary teen who is in love with Mary Jane Watson (Kirsten Dunst). One day on a school trip to a genetics laboratory, he is bitten by a genetically engineered super spider, and he goes through radical changes. He eventually practises his abilities, and after the death and revenge of his Uncle Ben (Cliff Robertson), he becomes the saviour for New York, web-swinging, crime fighting superhero Spider-Man. Meanwhile, Norman Osborn (Willem Dafoe) has a similar "accident" where he injects a performance	Positive : 😊



REFERENCES

1. “Classification & regression by Random Forest,” A. Liaw et al- R News, vol. 2, no. 3, pp. 18–22, 2002..
2. https://ieeexplore.ieee.org/document/8884146?denied
3. https://scholar.google.co.in/scholar?q=survey+paper+on+machine+learning+for+parkinson+disease+prediction&hl=en&as_sdt=0&as_vis=1&oi=scholart#d=gs_qabs&u=%23p%3D5WsSs6TSE0MJ
4. ‘‘Intraday prediction of Borsa Istanbul with convolutional neural networks and feature correlations,’’ Z. Cataltepe, H. Gunduz & Y. Yaslan- Knowl.-Based Syst., vol. 137, pp. 138–148, Dec. 2017.
5. ‘‘Least squares support vector machine classifiers,’’ J. A. K. Suykens and J. Vandewalle Neural Process. Lett., vol. 9, no. 3, pp. 293–300, Jun. 1999.
6. "Improved diagnosis of Parkinson’s disease with optimized crow search algorithm", D. Gupta, S. Sundaram, A. Khanna, A. E. Hassanien and V. H. C. de Albuquerque, Comput. Electr. Eng., vol. 68, pp. 412-424, May 2018.
7. ‘‘Step length determines minimum toe clearance in older adults & people with Parkinson’s disease,’’ L. Alcock, B. Galna, R. Perkins, S. Lord, and L. Rochester- J. Biomech., vol. 7, pp. 30–36, Apr. 2018.
8. ‘‘A pervasive & sensor-free deep learning system for Parkinsonian gait analysis,’ J. Ajay, C. Song, A. Wang, J. Langan, Z. Li, and W. Xu,’ in Proc. IEEE EMBS Int. Conf. Biomed. Health Informat. (BHI), Las Vegas, NV, USA, Mar. 2018, pp. 108–111.
9. “Support vector machines for multi-instance learning,” S. Andrews, I. Tsochantaridis, and T. Hofmann, in Proc. of Neural Information Processing Systems, vol. 15, 2003, pp. 577–584.
10. ‘‘Parkinson disease gait classification with ML approach,’’ N. M. Tahir and H. H. Manap- J. Appl. Sci., vol. 12, no. 2, pp. 180–185, 2012.
11. ‘‘Using Kinect to classify Parkinson’s disease stages related to seriousness of gait impairment,’’ L. Dranca, L. de Abetxuko Ruiz de Mendarozketa, A. Goñi, A. Illarramendi, I. N. Gomez, M. D. Alvarado, and M. C. Rodríguez-Oroz- BMC Bioinf., vol. 19, no. 1, Dec. 2018, Art. no. 471.