MOVIE RECOMMENDATION SYSTEM USING COSINE SIMILARITY

Authors:

NEELAY JAGANI- 1814024
JAI MEHTA- 1814036
PANKTI NANAVATI- 1814045
JASH MEHTA- 1814054

I. INTRODUCTION

In today’s world, where the people are constantly involved in some or the other activities, Recommendation systems have gained immense popularity. Since the people are always busy they tend to fall short of time for the infinite number of tasks they have at their hands. This is where the recommendation systems come to the help, allowing people to make right choices without having in-depth knowledge about the things that people are dealing with.

The recommendation systems help in searching for the content that would be likely to interest an individual, based on a number of factors that allow creation of a personalised list for the individual. Recommendation Systems are built using algorithms that go through various preferences of an individual and aid in creating the personalised list. These preferences involve attributes like profile, browsing history, what other people with similar traits/demographics are watching, and how likely you are to watch those movies. On performing predictive modelling and heuristics on the available data the results are achieved.

We have been using applications like Amazon, Netflix, Hotstar etc… which give us recommendations. We might often end up wondering how they give us such accurate recommendations. The answer to this quest is that we often rate products and media on the internet, such data are gathered by these applications and on the basis of our ratings and our preferences they create a list of recommendations for us, using the recommender systems. The two main types of recommender systems are either collaborative or content-based filters.

Often used in recommender systems, Content-based Filtering is a Machine Learning technique that uses similarities in features to make decisions, based on the knowledge gathered about the user. Collaborative Filtering, needs users’ historical preference on a set of items. Because it’s based on historical data, the core assumption here is that the users who have agreed in the past tend to also agree in the future.

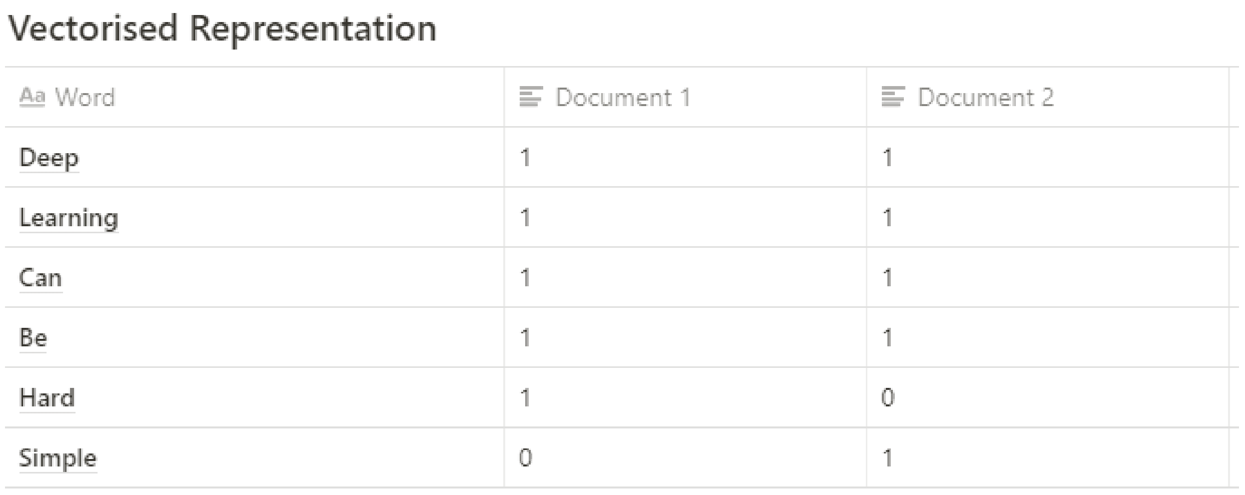
Using the content based filtering recommender system we have built a movie recommendation system that uses cosine similarity as a metric to find the similarity between the movies a user searches for and recommends movies to the user based on the keywords, cast, genre and directors of the entered movie.

II. APPLICATIONS

1. *Document Similarity*

Identification of the similarity between documents is an application of cosine similarity.
Documents can be checked for similarity by converting the words/sentences/phrases in each document into vectors. Then, these vectors are substituted in the cosine similarity formula to get the cosine similarity value.
The cosine similarity ranges from 0 to 1. When the cosine similarity comes out to be 1, it means that the two documents are similar while when the cosine similarity comes out to be 0, it means that the two documents are not similar.

Let's consider simple example of two documents each containing one sentence:
Deep Learning can be hard - Document 1
Deep Learning can be simple - Document 2
Step 1: Initially, we get the vector representation of all the texts/words in the two documents, as shown below



Vectorised Representation

Word	Document 1	Document 2
Deep	1	1
Learning	1	1
Can	1	1
Be	1	1
Hard	1	0
Simple	0	1

Document 1: - [1, 1, 1, 1, 1, 0] Therefore, let's consider A = [1, 1, 1, 1, 1, 0]
Document 2: - [1, 1, 1, 1, 0, 1] Therefore, let's consider B = [1, 1, 1, 1, 0, 1]
A and B assigned above are the two vectors having 6-dimension vector space.
Step 2: Finding the cosine similarity
cosine similarity (CS) = (A . B) / (||A|| ||B||)

- The dot product between A and B: 1.1 + 1.1 + 1.1 + 1.1 + 1.0 + 0.1 = 4
- The magnitude of the vector A: $\sqrt{1.0^2 + 1.0^2 + 1.0^2 + 1.0^2 + 1.0^2 + 0.0^2} = 2.2360$
- The magnitude of the vector B: $\sqrt{1.0^2 + 1.0^2 + 1.0^2 + 1.0^2 + 0.0^2 + 1.0^2} = 2.2360$
- The cosine similarity: (4) / (2.2360679775*2.2360679775) = 0.80 (80% similarity between the sentences in both document)

2. Pose Matching

In Pose Matching key points of joint locations in one image is compared to key points from another image. Thus, pose matching is done by comparison of key points.
Deriving the position or orientation of the body parts and joints from an image or series of images is known as pose estimation. It is usually a computer vision task which is solved using various Deep Learning approaches like PoseNet, Convolutional Pose Machine, Stacked hourglass etc.

Let's consider an example where there is a requirement to measure the similarity between two poses in Image A and Image B. Steps for the process are mentioned below:

A. Initially, identify the pose information and derive the location key points(joints) in Image A

B. Identify the x,y location of all the respective joints required for comparison. Deep Learning solution to pose estimation usually provides information on the location of the joints within a particular pose, along with an estimation confidence score.

C. Repeat above two steps (Step A and Step B) for Image B

D. Form a vector containing all x, y positions of Image A

E. Form a vector containing all x, y positions of Image B

F. Both vectors should have the same order of x, y positions of each joint

G. Perform cosine similarity using both vectors to obtain a number between 0 and 1.

3. Movie Recommendation System

Let's consider two persons, Person A(Bernard) and Person B(Clarissa) have similar movie preferences for only two movie reviews. The range of reviews is 0 to 5. 0 means that the person has not watched the movie, 1 stands for bad movie review while 5 stands for best movie review.

Name	Iron Man (2008)	Pride & Prejudice (2005)
Bernard	4	3
Clarissa	5	5

From the table given above, we formulate the two separate vectors for each person's review. Vector b stands for Person A and vector c stands for Person B.

$$\vec{b} = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \qquad \vec{c} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

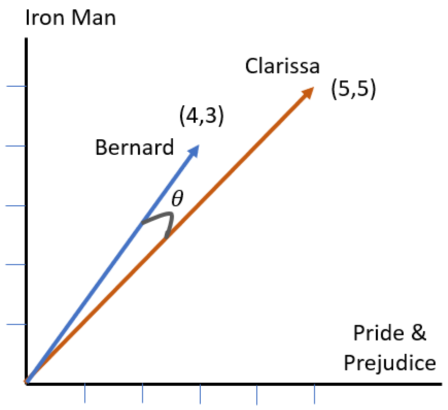
Then, we substitute the values of both vectors in the formula mentioned below to get the similarity measure.

$$similarity = \cos\theta = \frac{b.c}{\|b\|\|c\|}$$

$$b.c \Rightarrow Is\ the\ Dot\ product\ of\ the\ two\ vectors$$

$$\|b\|\|c\| \Rightarrow Is\ the\ product\ of\ each\ vector's\ magnitude$$

The graph shown below represents each vector by a line. The angle between two vectors is shown as θ. The lesser the angle θ, more is the similarity and vice versa.



$Calculating$:

$b.c = \sum_{i=1}^{n} b_i c_i = (4 \times 5) + (3 \times 5) = 35$

$\|b\| = \sqrt{4^2 + 3^2} = 5$

$\|c\| = \sqrt{5^2 + 5^2} = 5\sqrt{2}$

$similarity = \frac{35}{5 \times 5\sqrt{2}} \sim 0.989$

We get the similarity measure as 0.989 which is very close to 1. This implies that the two movie reviewers i.e. Bernard and Clarissa have almost similar movie preferences.

III. METHODOLOGY AND IMPLEMENTATION

Recommender systems are a form of machine learning algorithm that provides users with "related" recommendations. Content-based Filtering is used in recommendation systems that recommend the next product based on our previous behaviour, whereas Collaborative Filtering is used in recommendation systems that recommend the next product based on the habits and interests of other users that are close to us. A similarity matrix is used by recommendation systems to suggest the next most comparable product to the customer. The Cosine Similarity measure is one of the textual similarity measures used to construct this similarity matrix. The metric of cosine similarity is used to determine how identical two elements are. The value of cosine similarity ranges from 0 to 1.

The first step is to choose features, and in our context, we've chosen four key features based on which we'll make movie recommendations to the consumer. Keywords, cast, genres, and director are the four features chosen so a person who enjoys one horror film will most likely enjoy another horror film. Any users might enjoy seeing their favourite actors in the film's cast. Any users might enjoy seeing their favourite actors in the film's cast. Others may enjoy films directed by a certain individual. Our selected four features, when combined, are sufficient to train our recommendation algorithm. And every row's features will be consolidated into a single column.

Now, the next step is extraction of features. We will use **Countvectorizer** to count the number of texts, and to make the converted matrix easier to comprehend, we'll convert it to an array.

Now it's time to measure the cosine similarity. Cosine Similarity calculates similarity as the normalised dot product of input variables X and Y. To determine the cosine angle between the two vectors in the count matrix actually built by Countvectorizer, we then use sklearn cosine similarity function. For any of the films, the result is a matrix that is a numpy sequence of cosine similarity numbers.
Now, we will take the name of the movie from the user based on which we have to give our recommendation to the user.
We get the index of the film based on the user's input and use the enumerate approach to apply a counter to the cosine similarity matrix and show it as a list of related movies with the similarity value for each index.

Now, after getting the similar movies list we then sort the list in descending order as we want the most similar movies to be displayed first.
Thus, on the basis of this the most similar movies based on cosine similarity in their text will be then displayed to the user.

IV. RESULTS AND DISCUSSION

The implementation of Movie Recommendation System is able to generate five similar movies, these similarities we are checking with respect to genre, cast, keyword and director. Mathematically, if we are checking for similarities, two things are said to be similar if the angle between them is zilch, and are said completely independent with no similarity when the angle is approaching 90 degrees. Traversing through the entire dataset and finding similarity, we are generating all movies with highest similarity or the lowest angle between entities. The top five movies with highest similarity are displayed to the user as a recommendation.

V. REFERENCES

[1] Singh, Ramni & Maurya, Sargam & Tripathi, Tanisha & Narula, Tushar & Srivastav, Gaurav. (2020). Movie Recommendation System using Cosine Similarity and KNN. 2249-8958. 10.35940/ijeat.E9666.069520.

[2] I. Survyana Wahyudi, A. Affandi and M. Hariadi, "Recommender engine using cosine similarity based on alternating least square-weight regularization," 2017 15th International Conference on Quality in Research (QiR) : International Symposium on Electrical and Computer Engineering, 2017, pp. 256-261, doi: 10.1109/QIR.2017.8168492.

[3] Vijay Kotu, Bala Deshpande,Chapter 4 - Classification,Editor(s): Vijay Kotu, Bala Deshpande,Data Science (Second Edition),Morgan Kaufmann,2019,Pages 65-163,ISBN 9780128147610, https://doi.org/10.1016/B978-0-12-814761-0.00004-6.

[4] Rahutomo, Faisal & Kitasuka, Teruaki & Aritsugi, Masayoshi. (2012). Semantic Cosine Similarity.

[5] Gunawan, Dani & Sembiring, C & Budiman, Mohammad. (2018). The Implementation of Cosine Similarity to Calculate Text Relevance between Two Documents. Journal of Physics: Conference Series. 978. 012120. 10.1088/1742-6596/978/1/012120.

[6] A. R. Lahitani, A. E. Permanasari and N. A. Setiawan, "Cosine similarity to determine similarity measure: Study case in online essay assessment," 2016 4th International Conference on Cyber and IT Service Management, 2016, pp. 1-6, doi: 10.1109/CITSM.2016.7577578.

[7]https://towardsdatascience.com/understanding-cosine-similarity-and-its-application-fd42f585296a

[8]https://medium.com/@bkexcel2014/building-movie-recommender-systems-using-cosine-similarity-in-python-eff2d4e60d24

[9]https://www.slideshare.net/AbhishekJaisingh/movie-recommendation-project?qid=6980cae4-c750-455d-bd02-7cc72a46139e&v=&b=&from_search=5

[10]https://towardsdatascience.com/how-to-build-from-scratch-a-content-based-movie-recommender-with-natural-language-processing-25ad400eb243

[11]https://www.geeksforgeeks.org/cosine-similarity/

Originality report

COURSE NAME

Plagerism

STUDENT NAME

PANKTI NANAVATI

FILE NAME

1814024, 1814036, 1814045, 1814054

REPORT CREATED

Apr 30, 2021

Summary

Flagged passages	0	0%
Cited/quoted passages	0	0%