iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** IX **Month of publication:** September 2021

DOI: https://doi.org/10.22214/ijraset.2021.37817

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue IX Sep 2021- Available at www.ijraset.com

Building A Movie Recommendation System Using Collaborative Filtering With TF-IDF

S. A. Azeem Farhan[1], Dr. K. Sagar[2], Smt. T. Suvarna Kumari[3]
[1, 2, 3]*Computer Science and Engineering, Chaitanya Bharathi Institute of Technology, Osmania University*

Abstract: ***The recommendation problem involves the prediction of a set of items that maximize the utility for users. As a solution to this problem, a recommender system is an information filtering system that seeks to predict the rating given by a user to an item. There are theree types of recommendation systesms namely Content based, Collaborative based and the Hybrid based Recommendation systems. The collaborative filtering is further classified into the user based collaborative filtering and item based collaborative filtering. The collaborative filtering (CF) based recommendation systems are capable of grasping the interaction or correlation of users and items under consideration. We have explored most of the existing collaborative filtering-based research on a popular TMDB movie dataset. We found out that some key features were being ignored by most of the previous researches. Our work has given significant importance to 'movie overviews' available in the dataset. We experimented with typical statistical methods like TF-IDF , By using tf-idf the dimensions of our courps(overview and other text features) explodes, which creates problems ,we have tackled those problems using a dimensionality reduction technique named Singular Value Decomposition(SVD). After this preprocessing the Preprocessed data is being used in building the models. We have evaluated the performance of different machine learning algorithms like Random Forest and deep neural networks based Bi-LSTM. The experiment results provide a reliable model in terms of MAE(mean absolute error) ,RMSE(Root mean squared error) and the Bi-LSTM turns out to be a better model with an MAE of 0.65 and RMSE of 1.04 ,it generates more personalized movie recommendations compared to other models.***

Keywords: Recommender system, item-based collaborative filtering, Natural Language Processing, Deep learning.

I. INTRODUCTION

Recommender systems is an information filtering technique. It helps in providing the information which a particular person might be intrestedin. There are a wide variety of applications for recommendation systems. These have become increasingly popular over the last few years and are now utilized in most online platforms that we use. Such platforms' content ranges from movies, music, books, and videos to friends and stories on social media platforms, products on e-commerce websites, individuals on professional and dating websites, and Google search results. They must venture into new domains in order to learn more about the user while also making the most of what they already know about the user. Three main approaches are used for our recommender systems. One is Demographic Filtering i.e They offer generalized recommendations to every user, based on movie popularity and/or genre. Users with comparable demographic characteristics receive the same movie recommendations from the System. Because each user is unique, this technique is thought to be overly simplistic. The primary premise of this method is that films that are more popular and highly acclaimed are more likely to be loved by the general public. The second type of filtering is content-based filtering, in which we aim to profile users' interests based on the data we collect and then recommend goods based on that profile. The other is collaborative filtering, where we try to group similar users together and use information about the group to make recommendations to the user.

II. LITERATURE SURVEY

A recommender system or a recommendation system (sometimes replacing "system with a synonym such as platform or engine) is a type of information filtering system that attempts to forecast a user's "rating" or "preference" for a given item. Recommender systems are used in a wide range of applications, including movies, music, news, books, research articles, search queries, social tagging, and general items. There are also recommender systems for experts, collaborators, jokes, restaurants, clothing, financial services, life insurance, romantic partners (online dating), and Twitter sites are all examples.

GaojunLiu and xingyuWu[1] in “Using collaborative filtering Algorithms combined with Doc2vec for Movie recommendation” have suggested that we could use collaborative filtering where movies were recommended based on their features.

They've used the Doc2Vec models like "PV-DM (Paragraph Vector-Distributed Memory Model) and PV-DBOW (Paragraph Vector-Distributed Bag Of Words)" and handled the Cold start and sparcity problem. the proposed algorithm solves the deficiencies of the traditional recommendation algorithm model to a certain extent and improves the recommendation effect.

Rishabh Ahuja, Arun Solanki, Anand Nayyar[2] in "Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor" ,various tools and techniques have been used to build recommender systems. Various algorithms such as K-Means Clustering, KNN, Collaborative Filtering, Content-Based Filtering have been described in detail.The results given by the proposed system are better than the existing technique on the basis of RMSE value but with less number of clusters.

Meenu Gupta et al[3],their approach is based on cosine similarity using k-nearest neighbor with the help of a collaborative in filtering technique. To avoid the use of content-based filtering, the Item-based CF filtering approach is used for obtaining better results.

Barman, Surajit et al[6] in "Movie Recommendation based on User Similarity of Consumption Pattern Change" have presented an approach to calculate the similarity among the items based on the genre of items. Any item may belong to more than one genre or category. Based on items propensityt o a specific genre or category they have proposed a new item-item-based similarity metric.

Das D., Chidananda H.T., Sahoo L et al[7] in "Personalized Movie Recommendation System Using Twitter Data"tried to recommend a list of movies to specific Twitter users using content-based collaborative filtering approach by analyzing rating datasets collected from Twitter.they have got a pretty good results but they further recommended to use the various other textual data to develop a more powerful and efficient recommended.

Cheng, J., & Zhang, L et al[8] in "Jaccard Coefficient Based Bi-clustering and Fusion Recommender System for Solving Data Sparsity" have proposed an efficient rating-based recommender algorithm, JCBiFu which uses the density peak clustering method to cluster the user-item rating matrix, and estimates the missing values for sparsity data to cope with the sparsity problem in cold-start settings. the recommendation quality in terms of RMSE and MAE shows that JC-BiFu generates more accurate recommendations with less prediction error compare with other methods. However, If there exists a brand new user that have not rate any items, or a brand new item that has not been rated by any users, JC-BiFu cannot make good recommendations.

Hao wang et al[9] said that ,the ratings are often very sparse in many applications, causing CF-based methods to degrade significantly in their recommendation performance.

To address this sparsity problem, auxiliary information such as item content information may be utilized. Collaborative topic regression (CTR) is an appealing recent method taking this approach which tightly couples the two components that learn from two different sources of information

Ivica obadic et al [10] ,proposed Using a model-based approach and current breakthroughs in deep learning, a method for successfully tackling the item-cold start problem has been developed. They employed a latent factor model for recommendation and a convolutional neural network to estimate the latent variables from item descriptions when usage data was unavailable. To train the convolutional neural network, latent components created by applying matrix factorization to the existing consumption data are used as ground truth. The convolutional neural network uses the textual description of the new objects to construct latent factor representations for them. The results from the experiments reveal that the proposed approach significantly outperforms several baseline estimators.

III. PROPOSED SYSTEM

The collaborative filtering (CF) based recommendation systems are capable of grasping the interaction or correlation of users and items under consideration. We have explored most of the existing collaborative filtering-based research on a popular TMDB movie dataset. We found out that some key features were being ignored by most of the previous researches. Our work has given significant importance to 'movie overviews' available in the dataset.

We experimented with typical statistical methods like TF-IDF , By using tf-idf the dimensions of our courps(overview and other text features) explodes, which creates problems ,we have tackled those problems using a dimensionality reduction technique named Singular Value Decomposition(SVD). After this preprocessing the Preprocessed data is being used in building the models. We have evaluated the performance of different machine learning algorithms like Random Forest and deep neural networks based Bi-LSTM. The experiment results provide a reliable model in terms of MAE(mean absolute error) ,RMSE(Root mean squared error) and the Bi-LSTM turns out to be a better model with an MAE of 0.65 and RMSE of 1.04 ,it generates more personalized movie recommendations compared to other models.

.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue IX Sep 2021- Available at www.ijraset.com

IV. WORKFLOW OF A PROPOSED SYSTEM

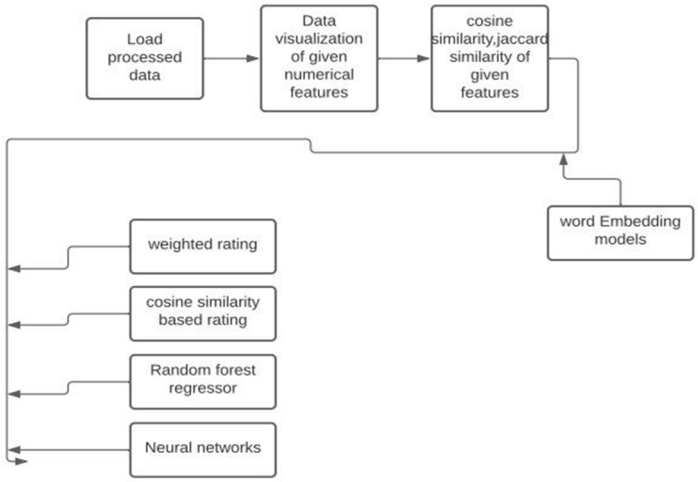


Fig 4.1 Workflow of A Proposed System

V. METHODOLOGY

1) *Preprocessing:* Intitially the dataset is incomplete, inconsistent, inaccurate and often lacks specific attribute values/trends. Preprocessing helps enhancing the quality of data so that meaningful insights from the data can be acquired. We start of the preprocessing by checking the summary of the dataset and then performing the data analysis on the features provided in the dataset. Then extracting the features like Actors, directors, genres, production companies, budget, overviews etc .The columns like genres, actors, production companies, directors consists too more than 3,4 values, but not every value is important for predicting the target, we are just going to consider the top 2-3 values from them and making the dataset ready for the next step feature selection. Removing all the unneccesary columns as we have already extracted the information from the dataset.
2) *Feature Selection:* The data available in context of movies ,mostly those relating to the names of entities is nonnumerical and hence the basic requirement is to convert it into processable format. The very first thing to do is to make tokens of a sentence and then using the NLP technique Term Frequency-Inverse Document Frequency method.TF-IDF, or (Term Frequency(TF) — Inverse Document Frequency(IDF)), is a strategy for determining the meaning of sentences made up of words that cancels out the shortcomings of the Bag of Words technique, which is useful for text classification or assisting a machine in reading words in numbers. At first we are going to break the sentences of textual feature the overview and then tokenization is being and also removing the symbols and the seperators. Tokenization breaks the raw text into words, sentences called tokens. After the tokenization and stemming have been completed, the process of reducing a word to its word stem, which affixes to suffixes and prefixes or to the roots of words known as a lemma, is known as stemming. Then, with the help of ngrams, we'll turn this corpus into a document feature matrix. DFM or document feature matrix refers to documents in rows and features as columns. Then comes the important step that is ,we are going to apply the TF-IDF formula on this document feature matrix. method. TF-IDF is a combination of two different words i.e. Term Frequency and Inverse Document Frequency. A Term Frequency is the number of times a term appears in a document (synonymous with bag of words). The number of times a term appears in a corpus of documents is known as the Inverse Document Frequency. After obtaining both these values we are going to multiply them and the natural log is being used with the idf and the formula for the TF-IDF goes by

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue IX Sep 2021- Available at www.ijraset.com

Again, With tf-idf a problem referred to as curse of dimensionality comes into picture, as the dimensions of our courps(overview and other text features) explodes, which creates a series of problems, However these problems can be tackled by using dimensionality reduction techniques.

As a dimensionality reduction technique, we'll employ SVD (Singular value decomposition). Singular Value Decomposition, or SVD, is one of numerous strategies for reducing the dimensionality of a data set, or the number of columns. Why would we want to lower the number of dimensions in the first place? More columns in predictive analytics usually equals more time spent building models and score data. If some columns have no predictive value, that means you're wasting time, or even worse, such columns add noise to the model, lowering model quality and forecast accuracy. Dimensionality reduction can be accomplished by simply deleting columns, such as those that may appear to be collinear with others or those that have been found as not being very predictive of the goal as determined by an attribute importance ranking technique. However, it is also possible to achieve this by generating new columns from linear combinations of the original columns. The resulting converted data set can be fed into machine learning algorithms in both scenarios, resulting in faster model build times, faster scoring times, and more accurate models. While SVD can be used to reduce dimensionality, it is most commonly utilised in digital signal processing for noise reduction, image compression, and other applications. The SVD algorithm divides a m x n matrix, M, of real or complex values into three component matrices using a factorization of the type USV*. A m x p matrix is U. S is a diagonal p x p matrix. V* is the transpose of V, a p x n matrix, or the conjugate transpose if M contains complex values, and V is a n x p matrix. The rank is the value of p. The singular values of M pertain to the diagonal entries of S. We can produce at least 10 new features by using SVD on our corpus**.**

3) *Cosine Similarity:* The metric of cosine similarity is used to determine how similar two items are, regardless of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. we are going to perform the cosine similarity between the new extracted features we got and store it in a separate column The smaller the angle between the two vectors, the more similar they are to each other. If the angle between the two vectors is 90 degrees, the cosine similarity will be 0. This indicates that the two vectors are perpendicular to one another. which means they have no correlation between them. The formula goes by

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\|\vec{b}\|}$$
$$\|\vec{a}\| = \sqrt{a_1^2 + a_2^2 + a_3^2 + \cdots + a_n^2}$$
$$\|\vec{b}\| = \sqrt{b_1^2 + b_2^2 + b_3^2 + \cdots + b_n^2}$$

Here “a” and “b” are the vectors in multidimensional space.
-1 value indicates that two vectors are strongly opposite to each other,0 indicates the independent vectors and 1 indicates the two vectors are strongly related with each other.

4) *Weighted Rating:* The first model that we are going to build for just testing we are going to perform the weighted rating which actually takes the inputs ratings, vote count, vote average. The formula goes by

$$W = \frac{Rv + Cm}{v + m}$$

Here W=weighted rating
R=average for the movie as a number
V=number of votes for the movie
M=minimum votes required to be listed
C=the mean vote across the whole report

After performing the weighted rating we are going to perform the tf-idf vectorization on the movie overviews and the cosine similarity metric is applied between them and then we are going to define the method called get_recommendation( ),which actually takes the particular movie as input and outputs the movies which are similar based on the overviews**.**

5) *Recurrent Neural Networks:* A Recurrent Neural Network (RNN) is a type of Artificial neural network which uses sequential data or Time series data. RNN are the state of the art algorithm for sequential data. Recurrent neural networks address this issue. They are networks with loops in them, allowing information to persist. It has Vanishing, Exploding Gradient-Problem. To overcome this problem we are using here Long and Short term Memory.

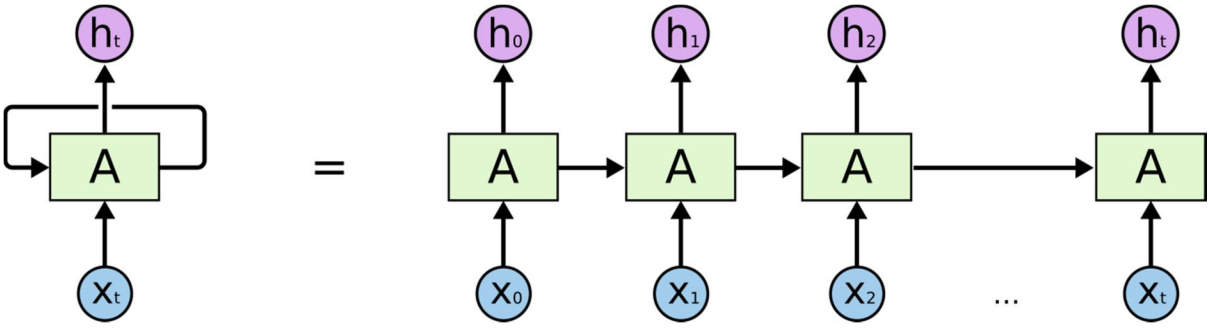


Fig 5.1 An unrolled Recurrent Neural Network

6) *LSTM Networks:* RNN consumes sentences word by word, updating the hidden state as each word is processed. Simple RNNs employ the tanh or sigmoid activation functions, which fail to grasp numerous long-term dependencies in sentences because they place greater emphasis on the most recent words encountered. During training, this also has the issue of vanishing and exploding gradients. LSTM learns to selectively forget and recall rather than storing every detail of a sentence in state. The gating mechanism is used to accomplish this.

It consists of three gates and a candidate memory cell. Gates aids in the selective forgetting and remembering of information in a sentence, with the contents being stored in a memory cell. The figure depicts the structure of an LSTM network. The layers of the LSTM are made up of the following components.

a) Forget Gate (ft): It chooses which information from the previous memory cell to delete.
b) Input Gate (it): It chooses which data to send to each cell to be remembered.
c) Candidate Memory (C´ t): It keeps track of the data from the current input.
d) Output Gate (ot): The current hidden state is computed using the tanh function, which squashes cell memory to lie between -1 and 1. At this moment, the output gate determines what should be output in the hidden state.
e) Cell Memory (Ct): It is the real cell memory that has been updated after relevant information has been added and unneeded data has been removed.

LSTMs are specifically developed to prevent the problem of long-term dependency.

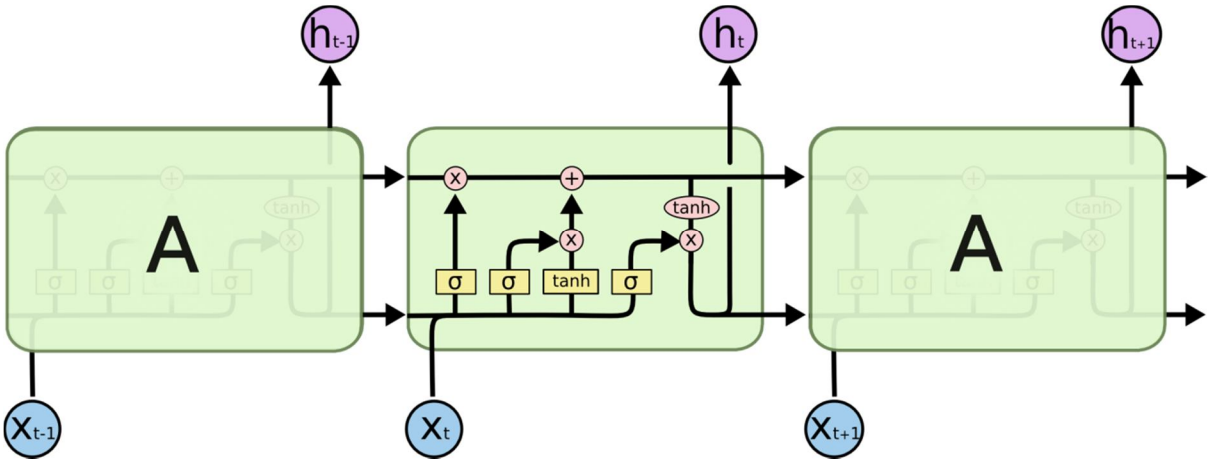


Fig 5.2 The LSTM contains four interacting layers

LSTMs have a chain-like structure as well, but the repeating module is different. Instead of one neural network layer, there are four, each interacting in a unique way.

7) *Bi-directional Long and Short Term Memory:* Bidirectional LSTMs are a type of LSTM that can be used to increase model performance in sequence classification issues. These bidirectional LSTM will manage your inputs in two directions: from the past to the future and from the future to the past.

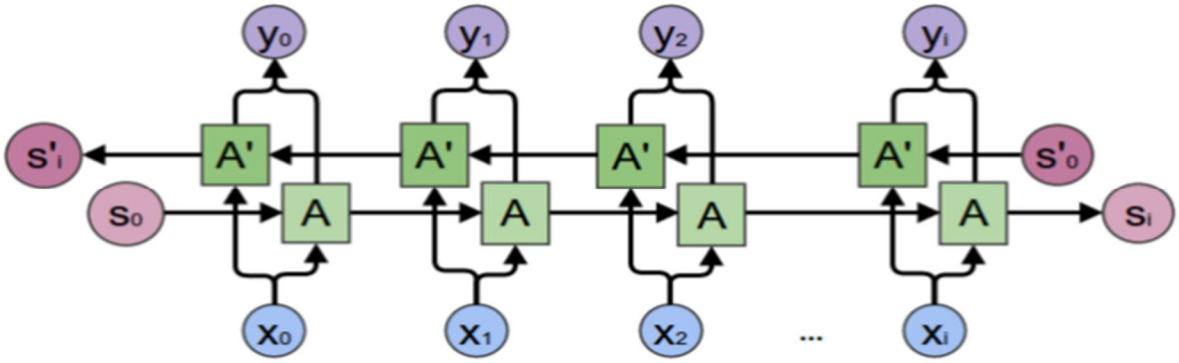


Fig 5.3 Bidirectional LSTM

VI. IMPLEMENTATION

1) *Dataset:* The dataset used is the TMDB movies dataset which is available on the Kaggle.com.

It contains more than 5000 movies and their rating and basic information, including user ratings and revenue data. A successful movie is evaluated by its popularity, vote average score(Ratings) and revenue.

It contains 2 CSV files

- tmdb_5000_credits.csv
- tmdb_5000_movies.csv

The tmdb_5000_credits consists of 4 colums which holds the information about the the cast,crew,title,movieid.

The tmdb_5000_movies consists of 20 columns.It tholds the information about the movie genre,overviews,vote_average,budget,keywords,original language,original language,tagline,popularity etc.

Programming Language and IDE Used for Data Analysis, Feature Extraction, Feature Engineering,Data Preprocessing, Data Visualization: R Programming, Rstudio

Programming Language and IDE used for model building:Python,Jupyter Notebook

a) We started of by loading the dataset into the workspace.
b) By checking the summary of the dataset which outputs mean median,min,max values of the numerical features present in the datastet and also the length,class,mode of the categorical fetaures.
c) Data visualization of the features is performes.
d) The NLP steps like tokanization,stemming,ngrams is being done on Features Overview,Genre,Actors
e) TF-IDF is being applied after the above step.
f) With TF-IDF curse of dimensionality comes into picture ,applying SVD and getting the top 10 Features out of the features
g) The first model which we build is based on weighted rating and cosine similarity.
h) Using the clean and the processed data the second model which is based on Random Forest regressor is being built,where we are predicting the ratings and calculating the
i) Using the metrics MAE,RMSE ,since it is a regression mode
j) Using the clean and processed data the third model which is based on Bi-Directional LSTM is built,similar to the above model this model also predicts the ratings.
k) The metrics employed are MAE,RMSE
l) Comparing both the models i.e Random forest regressor and the Bi-Directional LSTM in terms of MAE,RMSE.

VII. RESULTS AND DISCUSSIONS

In the very first model we are going to calculate the weighted rating and also we are going to be testing whether we can make use of overview feature or not by applying the cosine similarity between the overviews.

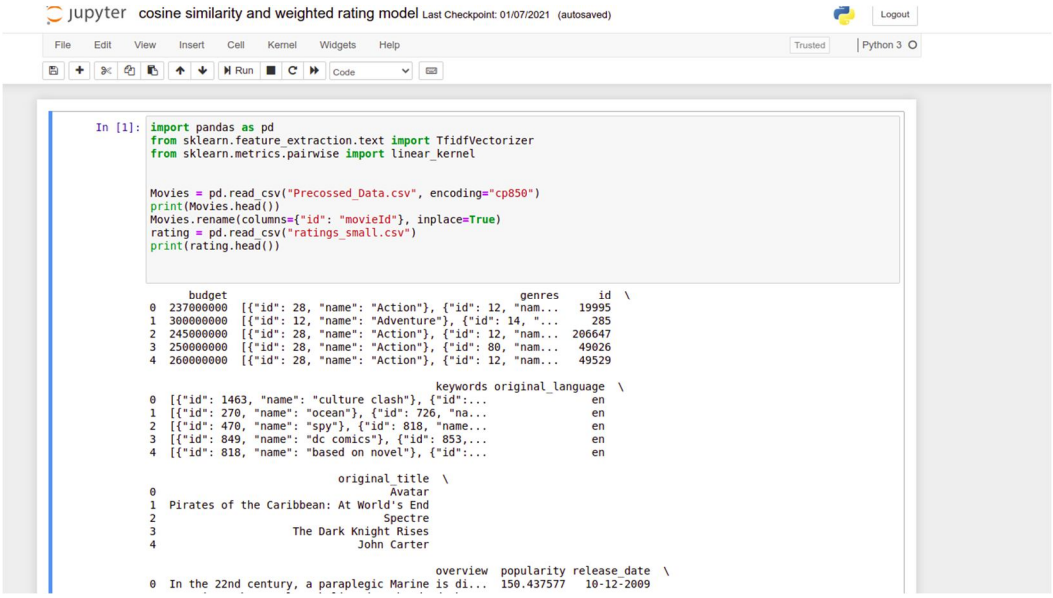


Figure 7.1 Weighted rating model

	score
0	(6.857592342055094, 129.4363873922748)
1	(6.343194977806219, 109.06992528839811)
2	(7.164067959293552, 95.04219789184255)
3	(6.92154222856083, 136.11367322379755)
4	(6.92154222856083, 136.11367322379755)
5	(6.92154222856083, 136.11367322379755)
6	(6.92154222856083, 136.11367322379755)
7	(6.92154222856083, 136.11367322379755)
8	(6.92154222856083, 136.11367322379755)
9	(6.92154222856083, 136.11367322379755)
10	(6.92154222856083, 136.11367322379755)
11	(6.92154222856083, 136.11367322379755)
12	(6.92154222856083, 136.11367322379755)
13	(6.92154222856083, 136.11367322379755)
14	(6.92154222856083, 136.11367322379755)
15	(6.92154222856083, 136.11367322379755)
16	(6.92154222856083, 136.11367322379755)
17	(6.92154222856083, 136.11367322379755)
18	(6.92154222856083, 136.11367322379755)
19	(6.92154222856083, 136.11367322379755)
20	(6.92154222856083, 136.11367322379755)
21	(6.92154222856083, 136.11367322379755)
22	(6.92154222856083, 136.11367322379755)
23	(6.92154222856083, 136.11367322379755)
24	(6.92154222856083, 136.11367322379755)
25	(6.92154222856083, 136.11367322379755)
26	(6.92154222856083, 136.11367322379755)
27	(6.92154222856083, 136.11367322379755)
28	(6.92154222856083, 136.11367322379755)
29	(6.92154222856083, 136.11367322379755)
...	...
18010	(7.4216777596897625, 72.78719886199126)
18011	(7.4216777596897625, 72.78719886199126)
18012	(7.4216777596897625, 72.78719886199126)
18013	(7.4216777596897625, 72.78719886199126)

Fig 7.2 weighted rating score

The weighted rating score can be seen in the above figure which is being calculated by using the imdb formula and it takes the features like vote count,vote average,revenue and the popularity into the consideration and then the formula is being applied on those features to obtain the weighted rating score.

```
In [4]: print(get_movies("Avengers: Age of Ultron"))

7                       Avengers: Age of Ultron
26                   Captain America: Civil War
16                                 The Avengers
227                              Knight and Day
233        Star Wars: Episode I - The Phantom Menace
3743                          Leaving Las Vegas
2430                                  Evil Dead
79                                   Iron Man 2
85           Captain America: The Winter Soldier
1981                                 Piranha 3D
240                                Dante's Peak
Name: title, dtype: object

In [5]: print(get_movies("Pirates of the Caribbean: Dead Man's Chest"))

12    Pirates of the Caribbean: Dead Man's Chest
0                                         Avatar
1       Pirates of the Caribbean: At World's End
2                                        Spectre
3                          The Dark Knight Rises
4                                    John Carter
5                                   Spider-Man 3
6                                        Tangled
7                        Avengers: Age of Ultron
8             Harry Potter and the Half-Blood Prince
9               Batman v Superman: Dawn of Justice
Name: title, dtype: object

In [6]: print(get_movies("Harry Potter and the Half-Blood Prince"))

8         Harry Potter and the Half-Blood Prince
114          Harry Potter and the Goblet of Fire
113    Harry Potter and the Order of the Phoenix
```

Fig 7.3 Results for cosine similairty between the overviews of items

The above figure shows the output of the function get_movies function where related movies are being displayed when the input is being passed.Here the input is being taken as a static type input.And recommendations of Three different inputs is being displayed.

```
In [3]: clf = RandomForestRegressor(n_estimators=10, max_features='auto', oob_score=True)
        cv = RepeatedKFold(n_splits=10, n_repeats=1, random_state=1)
        n_scores = cross_val_score(clf, X_train, y_train, scoring=make_scorer(metrics.mean_squared_error), cv=cv, n_
        print("SCOORE:::: %3f (%.3f)" % (np.mean(n_scores), np.std(n_scores)))

/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_validation.py:514: DataConv
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sampl
for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/forest.py:737: UserWarning: Some in
o not have OOB scores. This probably means too few trees were used to compute any reliable oob estimates.
  warn("Some inputs do not have OOB scores. "
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_validation.py:514: DataConv
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sampl
for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/forest.py:737: UserWarning: Some in
o not have OOB scores. This probably means too few trees were used to compute any reliable oob estimates.
  warn("Some inputs do not have OOB scores. "
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_validation.py:514: DataConv
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sampl
for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/forest.py:737: UserWarning: Some in
o not have OOB scores. This probably means too few trees were used to compute any reliable oob estimates.
  warn("Some inputs do not have OOB scores. "
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_validation.py:514: DataConv
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sampl
for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/forest.py:737: UserWarning: Some in
o not have OOB scores. This probably means too few trees were used to compute any reliable oob estimates.
  warn("Some inputs do not have OOB scores. "
/home/azeemfarhan/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_validation.py:514: DataConv
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sampl
for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
```

Fig 7.4 Random Forest Regressor model

In the above screen we are defining the random forest regressor model where we performed the 10 fold cross validation on the training set and the metric employed is mean squared error

```
Y6          0.000224
Y9          0.000200
Y1          0.000189
Y4          0.000133
Y2          0.000083
Y5          0.000003
Y8          0.000002
dtype: float64
```

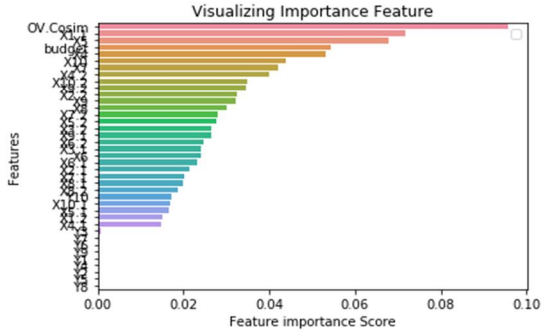


Fig 7.5 Feature importance score

In the above output screen the fitting of the data into the model is being done and also the important function which is feature importance score is being used. which outputs the importance of the features present in the dataset which also include the features which we have created like x1,x2,x3 etc and the importance is being displayed in terms of the percentage.

```
In [8]: from math import sqrt
        y_pred = clf.predict(X_test)
        MAE=mean_absolute_error(y_test, y_pred)
        MSE=mean_squared_error(y_test,y_pred)
        RMSE=sqrt(MSE)

In [9]: print("MAE:",MAE)
        print("MSE:",MSE)
        print("RMSE:",RMSE)

        MAE: 0.7924635669673838
        MSE: 1.352976960444136
        RMSE: 1.1631753781971728

In [ ]:
```

Fig 7.6 MAE, MSE, RMSE of Random Forest Regressor

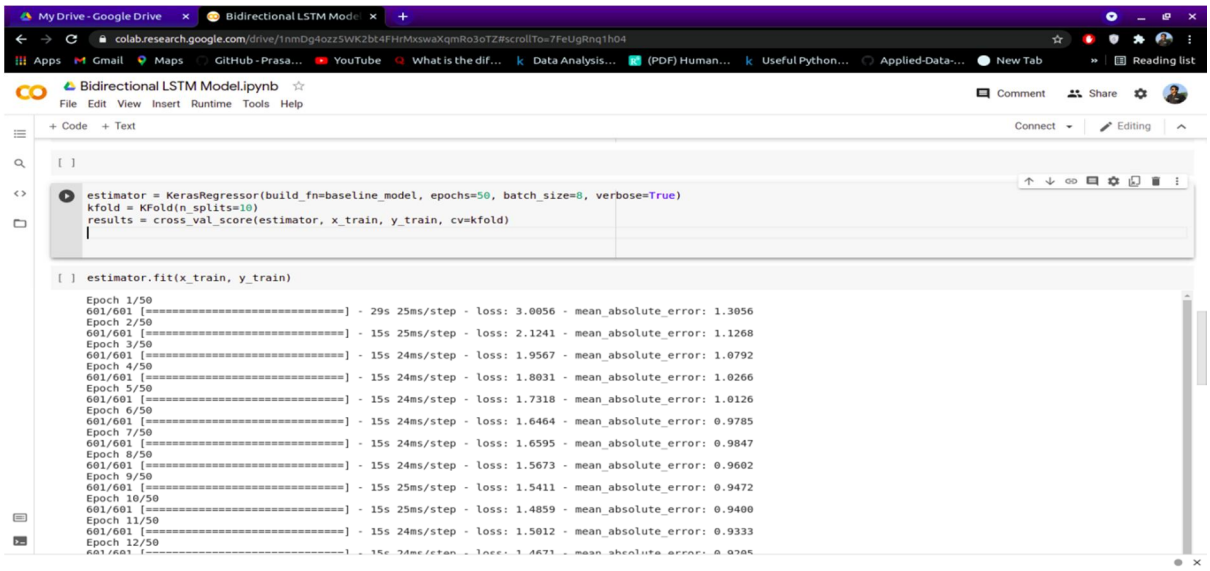


Fig 7.7 Training the Neural Network model with 50 epochs

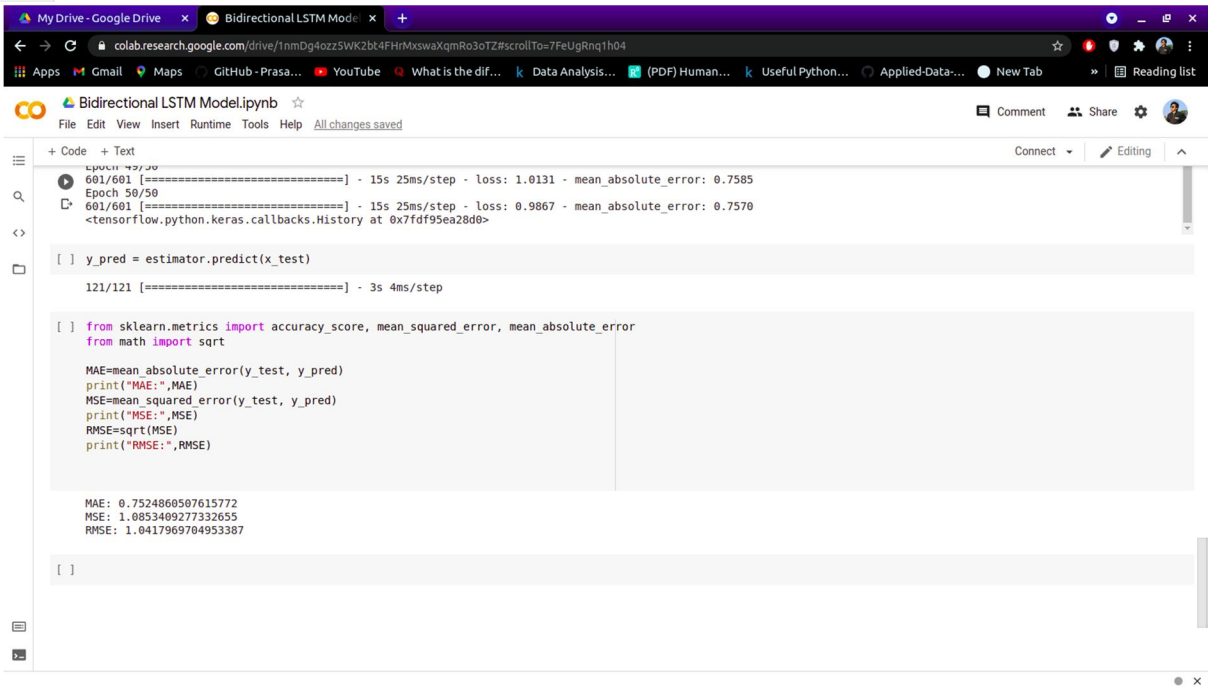


Fig 7.8 MAE, MSE, RMSE Of BiDirectional LSTM

Fig 7.9 comparison plot of Random Forest Regressor and BI_Directrional LSTM

The above plot compares the MAE, MSE, RMSE values of the Random Forest Regressor and the Bi-Directional LSTM. And from the above Plot we can say that the Bi-directional LSTM performed better in terms of the above mentioned metrics when compared with the Random Forest Regressor.

VIII. CONCLUSION

A. As the traditional collaborative filtering suffers from the problems like Data sparcity and cold start, we have tried to use the features of the items to a good extent, the features such as overview, production companies, actors, directors, genre, etc will have a huge impact in developing a recommendation system.

B. In this paper, we proposed to use the NLP technique TF-IDF on various features of the items to handle the problems of traditional collaborative filtering. Further, using the bi-directional LSTM's for building model and we got the MAE of 0.75 and RMSE of 1.04.

C. Also, we have used random forest regress or algorithm which will output the best features which can be used in building the models and boosting the accuracy of models. The experiment on the TMDB dataset verify that the proposed method handles the limitations of the traditional collaborative filtering to some extent by improvising the recommendations.

IX. ACKNOWLEDGEMENT

While bringing out this thesis to its final form, I came across a number of people whose contributions in various ways helped my field of research and they deserve special thanks. It is a pleasure to convey my gratitude to all of them.First and foremost, I would like to express my deep sense of gratitude and indebtedness to my supervisors Dr.K.Sagar & Smt.T.Suvarna Kumari for their invaluable encouragement, suggestions, and support from an early stage of this research and providing me extraordinary experiences throughout the work.I am also thankful to the Head of the Department **Dr.Y.Rama Devi** for providing excellent infrastructure and such a nice atmosphere for completing this project successfully.Finally, I would like to take this opportunity to thank my family and friends for their support throughout this work. I also sincerely acknowledge and thank all those who have directly or indirectly their supported in the completion of this work.

REFERENCES

[1] Gaojun Liu,xingyu Wu: "Using collaborative filtering Algorithms combined with Doc2vec for Movie recommendation" 2019 IEEE 3rd information Technology, Networking ,Electronic and Automation Control Conference.

[2] Rishabh Ahuja,Arun Solanki,Anand Nayyar : "Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor" 2019 9th international conference on Cloud computing, Data science & Engineering.

[3] Meenu Gupta, Aditya Thakkar,Aashish,Vishal Gupta,Dhruv Pratap Singh Rathore: "Movie Recommender System Using Collaborative Filtering" 2020 International Conference on Electronics and Sustainable Communication Systems.

[4] Ching-Seh (Mike) Wu,Deepti Garg,Unnathi Bhandary: "Movie Recommendation System Using Collaborative Filtering" 2018 IEEE 9th International Conference on Software Engineering and Service Science

[5] Kim, M., Jeon, S., Shin, H., Choi, W., Chung, H., &Nah,Y:"Movie Re commendation based on User Similarity of Consumption Pattern Change".2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering

[6] Barman, Surajit Das, Mahamudul Hasan, and Falguni Roy:"A Genre-Based Item- Item CollaborativeFiltering: Facing the Cold-Start Problem".2019 ACM International Conference Proceeding Series .

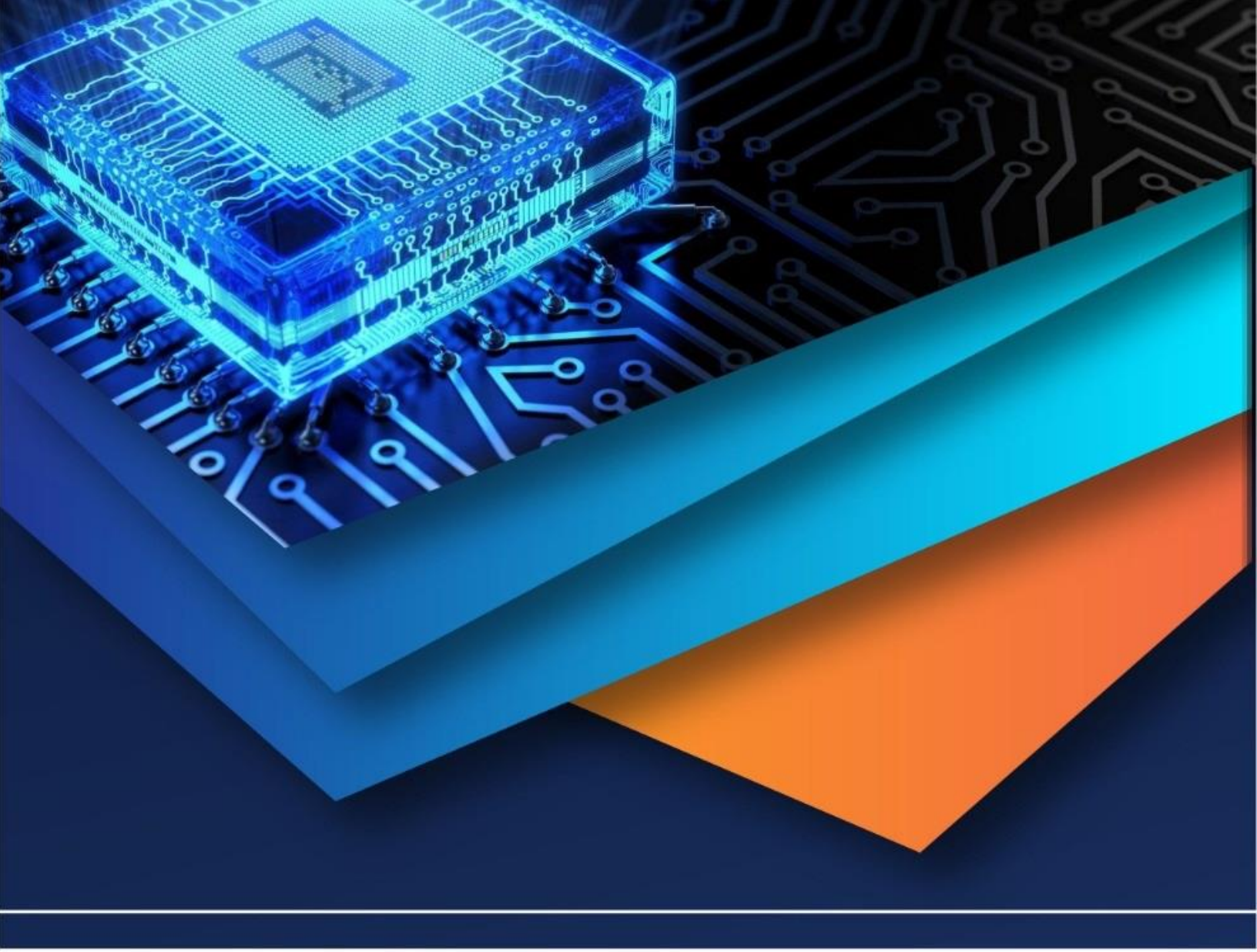
[7] Das D., Chidananda H.T., Sahoo L, Pattnaik P., Rautaray S., Das H., Nayak J:" Personalized Movie Recommendation System Using Twitter Data" 2018 Progress in Computing, Analytics and Networking. Advances in Intelligent Systems and Computing, vol 710. Springer

[8] Cheng, J., & Zhang, L: "Jaccard CoefficientBased Bi-clustering and Fusion Recommender System for Solving Data Sparsity".2019 Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 369-380). Springer

[9] Wang H, Wang N, Yeung D Y.: "Collaborative deep learning for recommender systems"2015 SIGKDD International Conference on Knowledge Discovery and Data Mining.Sydney, Australia, 2015: 1235-1244

[10] Obadić I, Madjarov G, Dimitrovski : "Addressing Item-Cold Start Problem in Recommendation Systems Using Model Based Approach and Deep Learning"2017 International Conference on Ict Innovations.

[11] Qaiser, Shahzad & Ali, Ramsha. Text Mining: "Use of TF-IDF to Examine the Relevance of Words to Documents" 2018 International Journal of Computer Applications.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY