**Movie and TV Show Recommendation using BOW and Deep Learning Models**

**By:**

**Irfan Ahmed (iar2f@mail.umkc.edu)**

**Srikanth Yelem (sy99b@mail.umkc.edu )**

**Divyanshi Kothari (dk9x8@mail.umkc.edu)**

**Project:**

Movie and TV show recommendation by creating BOW of words from description, title, types, director, and cast. Here we are applying various models to determine which model could provide better results.

**Team Details:**

* Irfan’s contribution to the project:
  + Creating Bag of Words using RAKE
  + Linear Kernel for recommendation
  + Creating Deep learning Model using LSTM
* Srikanth’s contribution to the project:
  + Cosine Similarity model for recommendation
* Divyanshi Kothari’s contribution:
  + Using TF-IDF Technique
  + Using Sigmoid Kernel Recommendation Model

**The Story and its Details:**

**Story Summary:**

* For the story we created a student named Jaz short for Jasmine. Who is a full-time graduate student at UMKC living in North KC, MO. She has a busy schedule with her research and class work; and after a long day she likes to relax and watch her favorite movies and tv shows. She hates having to browse for long periods of time just to find out later the recommendation she received were not what she expected.

**The Data and its Details:**

* The dataset was obtained from Kaggle and the link to the dataset is in the referenced section. It has 12 columns and 7788 rows.
* The column names are the following: show\_id, type, title, director, cast, country, date\_added, release year, rating, duration, listed\_in, and description.
* There are 5377 Movies and 2410 TV shows. More data information is shown below in the images labeled figure 1, 2, and 3.

![Graphical user interface, application

Description automatically generated]()

Figure 1: Movie and TV Show count

![Table

Description automatically generated]()![Table

Description automatically generated]()

Figure 2: Rating count Figure 3: Data types

**Working Screens from the Project:**

**Irfan’s working screen from project:**

The following screen shots are for the output results for creating bag of word column.

I removed the unwanted columns from the original dataset and used RAKE to create list of keywords.

![Text

Description automatically generated]()

After the keywords were created, I mapped the data and joined the names and move types and title together.

![Graphical user interface, text

Description automatically generated]()

After properly mapping the data and joining the words with the keywords a separate column was created named BOW. Which contained title, type, director, cast and keywords.

![Graphical user interface, text, application

Description automatically generated with medium confidence]()

The BOW word dataset was then saved and used to model Linear Kernel Model and deep learning model with LSTM.

The images below are for the Linear Kernel model results.

The preprocessed dataset was used to model the recommendation. The model is decent at picking movies with similar names and keywords but is not able match greater than 50%, which is what I was aiming for. Based on the example shown below. I wanted to watch movies relating to Christmas and the input was “a Christmas prince” a movie the user had seen previously. The results below are the recommendations for movies that might be similar.

![Graphical user interface, text, application

Description automatically generated]()![Text

Description automatically generated]()

The images below are for deep learning model using LSTM.

The model summary for the sequential model is shown below along with the accuracy results.

![Table

Description automatically generated]()

Based on the loss we can see that model is heavily overfitting the dataset. Could not figure out why. It could be due to the error in how the dataset was preprocessed. Maybe if I had cleaned up the dataset before using rake the results could have been different.

Table

Description automatically generated with low confidence

Graph for history accuracy and validation loss.

Chart

Description automatically generated

Graph for history loss and validation loss.

Chart, line chart

Description automatically generated

**Srikanth’s working screens from the project:**

The following screen shots are for the output results which we used before.

The original dataset and used RAKE to create list of keywords. Where keywords created, mapped the data, and combined the names and movie types and title together. ![Text

Description automatically generated]()

![Graphical user interface, text

Description automatically generated]()

where data mapped and words joined with the keywords a separate column was created named BOW. Which contained title, type, director, cast and keywords.

![Graphical user interface, text, application

Description automatically generated with medium confidence]()

The BOW word dataset was then saved and used Cosine Similarity model for recommendation

Cosine similarity model is used for training the data and getting recommendations. Cosine Similarity calculates the similarity between two things. After computing the similarity, we further use those computed values for recommendations.

![Graphical user interface, text, application

Description automatically generated]()

We pass the title and get all the recommendations that matches the title and After taking the input from the user the input is converted to lower case to make search more efficient. Each row of BOW checks if the input string is present in that column and returns matching rows of that title.

**![Text

Description automatically generated]()**

Once the data is trained, we pass the title to the model and will get all the similar movie ids related to the title and the accuracy of movie ids with respect to the title given as an input.

![Graphical user interface, text, application

Description automatically generated]()

After getting similar movie ids for the title, we now get the movie titles for the recommended ids.

![Graphical user interface, text

Description automatically generated]()

**Divyanshi Kothari’s working screens from the project:**

Using Sigmoid Kernel Recommendation Model and TFIDF Technique:

The objective here is to clean the data, removed the unnecessary columns present in the dataset and apply the models and techniques. Below are the details:

Graphical user interface

Description automatically generated with medium confidence

Below are all the columns that are present in the Netflix dataset.

Table

Description automatically generated

The unnecessary columns present in the Netflix dataset are removed and columns like description, cast and director are kept and combined.

Graphical user interface, text, application, email

Description automatically generated

After removing the stop words, below are the words and their counts.

Text

Description automatically generated with medium confidence

Below are the resultant recommendations that are provided from this model. The accuracy of the model is given below.

Graphical user interface, text

Description automatically generated

The same has been performed on the TV shows as well. Below are the working snippets:

Graphical user interface, application

Description automatically generated

Text

Description automatically generated

**Improvements from the Previous Increment:**

* From the previous increment we have added more details to our report and redefined our story. Initially what we wanted to do did not pan out because our understanding as a group was still growing. It was not until later in the semester we decided to compare different models and a little bit of variations in the dataset like creating bag of words by combining different columns then run that dataset for different models and compare our results with each other.
* After a lot of understanding and applying various techniques, we were able to determine the performance and accuracy of each model for the recommendation. And were able to determine that which model could give a better result.

**Important Code Snippets from the Project:**

**Irfan’s code snippets.**

The code below was used to create keywords. This is where the RAKE algorithm was applied. Generally, RAKE is great for keyword extraction for large data texts.

A new column was created first and named “keywords”. Then loop was created go over each row of index and assigning score to it by creating list of the key\_words\_dict\_score.

![Graphical user interface, text, application, email

Description automatically generated]()

The second image is for the Linear Kernel model where TfidfVectorzier was applied to the BOW of words column with ngram\_range of 1-3 to make sure it would be able to obtain the best results. Then the vectorized data was passed into the linear kernel function.

![Graphical user interface, text, application, email

Description automatically generated]()

Sequential model is compared with the Linear Kernel model to see which would give the best results. This model compared poorly with he previous model and kept overfitting.

Graphical user interface, text

Description automatically generated

**Srikanth’s code snippets:**

Here in below code explains how cosine similarity recommendations has been used and implemented to get top 5 movie recommendations based on tittle, type and description.

![Graphical user interface, text, application, email

Description automatically generated]()

Cosine similarity will get movie id and shows how it check between the given movie name and other movie where data available in BOW.

![Graphical user interface, text, application, email

Description automatically generated]()

**Divyanshi’s code snippets:**

Using Sigmoid Kernel Recommendation Model and TFIDF Technique: Here the Netflix data is taken and is cleaned . Description Cast and Director are combined to get best accurate results. Here the TFIDF which is Term Frequency- Inverse Document Frequency, Vectorizer is imported. If Using TFIDF, the stop words are identified and is removed by the model.



We have used the combined data and used transformation on it. Any duplicated words are dropped and removed. All the words and their counts are then printed to check. Here the Sigmoid kernel is then applied.

Graphical user interface, text

Description automatically generated with medium confidence

Now with the words that are obtained, a recommendation model is defined. Here the list id enumerated and sorted with the scores of how accurate the recommendation would be. With the indices and scores, using a loop the kernel model recommendation movies are printed.

Graphical user interface, text, application

Description automatically generated

To find similar recommended TV Shows:

Here the TFIDF which is Term Frequency- Inverse Document Frequency, Vectorizer is imported. If Using TFIDF, the stop words are identified and is removed by the model. The Sigmoid model is then applied.

Graphical user interface, application

Description automatically generated

With the words that are obtained, a recommendation model is defined. Here the list id enumerated and sorted with the scores of how accurate the recommendation would be. With the indices and scores, using a loop the kernel model recommendation series are printed

Graphical user interface, text, application

Description automatically generated

**Working Sharing/Module Sharing Between Teammates:**

* Out meeting were conducted via zoom and everything we worked on was uploaded to Github.

**Issues and Blockages with the Project:**

* One of our teammates had to go back to India because of an emergency. So, we rushed to finalize everything and could not work out the kinks like we wanted. For future work I will be trying to improve the deep learning model with LSTM to see what changes can improve the model. First will try to improve on the preprocessing of the dataset before implementing each model.

**GitHub Link of the Project:**

* Link to the entire repository
  + <https://github.com/irfancheemaa/GroupProjectSpring2021DeepLearningPython>
* Irfan’ code links:
  + <https://github.com/irfancheemaa/GroupProjectSpring2021DeepLearningPython/blob/main/source_code_final_code/creatingBOW_by_irfan.ipynb>
  + <https://github.com/irfancheemaa/GroupProjectSpring2021DeepLearningPython/blob/main/source_code_final_code/linear_kernel_search_model_by_irfan.ipynb>
  + <https://github.com/irfancheemaa/GroupProjectSpring2021DeepLearningPython/blob/main/source_code_final_code/deep_learning_model_by_irfan.ipynb>
* Divyanshi’s code links:
  + <https://github.com/irfancheemaa/GroupProjectSpring2021DeepLearningPython/blob/main/source_code_final_code/Sigmoid_Kernal_Recommendation_Model_by_divyansi.ipynb>
* Srikanth’s code link:
  + <https://github.com/irfancheemaa/GroupProjectSpring2021DeepLearningPython/blob/main/source_code_final_code/finalproject_by_srikanth.ipynb>

**Video Link of the Project:**

* Irfan’s video link to his code:
  + <https://umkc.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=d7a600e0-4d46-4e14-a16b-ad2000037a8e>
* Srikanth’s video link to his code:
* <https://youtu.be/dZPs5m0m77I>
* Divyanshi Kothari’s link to her code:
* <https://youtu.be/r7IKFxRbxyk>

**References:**

* https://www.kaggle.com/shivamb/netflix-shows
* Class PowerPoints