PRML Project

Movie Recommendation System

Vageesh (B20AI049)

Rahul Barodia (B20CS047)

Priyank Mandal (B20AI055)

Abstract:

In this project we were asked to experiment with a real world dataset and use machine learning algorithms to make a movie recommendation system.

We were expected to use three to four concepts from the course for doing. We were also expected to submit a report with a brief description of our ideas , experiments and results along with proper documented codes.We also deployed our project in a web application form into our local machine. After performing all the required tasks, herein lies our final report.

Methodology and Preprocessing:

Movie Recommendation system is a system that predicts or filters movie preferences according to the user's choices. The movies can also be predicted based on the genre of movie. Various Machine learning models can be applied for making such a system.

Throughout the course we studied various ML models like linear regression model, logistics regression, KNN , SVM, Xgboost , lightGBM , etc. Now we have to figure out what models out of these we can use in our project. We identified that for predicting or recommending good movies we need to find similarities between the movies.

This led us to use models like **KNN and Neural Networks**. We also made two models from scratch named **Content based and Genre based**.

Coming to the preprocessing of the dataset, we could see that the movies dataset has genre column as a string separated by " | ", this needed to be processed in such a way that it could be used to group the similar genre movies. For this we used an NLP component where we use a tokenizer which converts the strings to vectors with certain words and the word counts in them. This converts the string data to a vector for further analysis. The **Sklearn's Count Vectorizer** was used for this task.

The other data in movies dataset and the ratings needn't be preprocessed and can be used in the way they are.

Working and Experimentation:

Content Based Recommendation:

This is the first model that was implemented from scratch for the recommendation system. The concept behind the working of this model is basically based on the **similarity between two vectors**.

The angle between the two vectors gives an intuitive approach to how similar the two vectors are to each other in terms of the direction. Hence this would mean if the angle is extremely small then the two vectors are almost identical and if the angle is large then the vectors would vary greatly. As a result the **Cosine Similarity** function of sklearn was used for this as the **output of the cosine function varies inversely with the angle** and hence is more intuitive for this problem.

$$\cos\theta = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}$$

So, for all the samples of the dataset the cosine similarity is calculated and then based on this the top similar movies are used for recommendation of a given movie. Certain functions were made to diversify the number of recommendations of this model.

The Neural Network Model:

The Neural network embeddings is a method to represent discrete categorical variables as continuous vectors. As a learned **low-dimensional representation**, they are useful for finding similar categories as input into machine learning models.

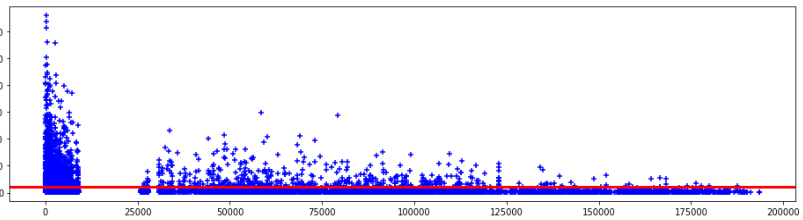
Firstly we **encoded** movie id and user id using LabelEncoder imported from sklearn. Then we normalized the ratings and also splitted rating dataset into test and train. Then we imported all the necessary libraries and defined a function called **RecommendorV1**. Then fitted the train data into it and validated it into the train dataset. Thus model was trained. Then we took random user id . Further we projected the movies user has watched as well as the movies the user has not watched and predicted the labeled movie ids. And finally unlabelled them into original movie ids.

The KNN Model:

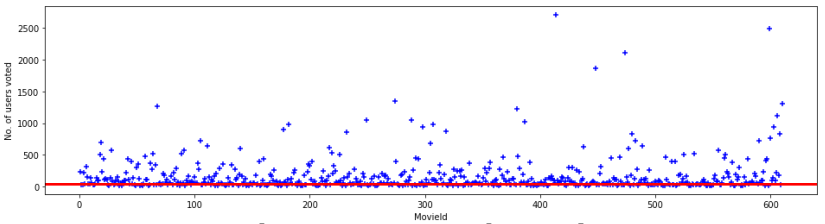
The third model implemented for the movie recommendation is the KNN model, where the concept of KNN is used on the dataset. The dataset used here wasn't directly taken; it had to be modified in order to use it. Here we had decided on using the **ratings as a criteria of grouping**, as in the movies with **similar ratings can be clustered together** to get recommendations of a certain movie in a certain cluster.

For the modified dataset we took the ratings dataset and created a **pivot matrix** out of it with userID on the columns and the movieID on the rows, and the values filled in a particular cell were the ratings given by a certain user to a certain movie. After making the matrix there were cells which had NaN values (All users didn't rate all the movies, hence NaN), These were first filled with values of 0 in them.

Now the matrix required further processing, as all the movies and all the user data cannot be used and we needed to **set a certain threshold** to filter out the data which had **significance** out of the whole data. The arrays with information on the votes casted by every user and the votes received by each movie were calculated and plotted.



The votes by each user



The votes to each movie

As we can see from the plotted graph the threshold of **10** in the **votes by each user** and **50** on **votes to each movie**, does separate out the majority of the significant data for further analysis.

As there is a huge number of 0 entries in the matrix it needed to be converted into a **CSR Matrix.** This matrix was then fitted in the KNN model and then the main function for recommendation was made and the recommendations were reported.

Genre Based Recommendation:

This is the last model for the recommendation system and this was made from scratch. The main idea behind the working of this model comes from **how the K-Means algorithm works**. The K-Means algorithm uses **minimum l2-Norm** as a metric to decide upon which point belongs to which cluster.

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

This idea was used in this genre based recommendation to find the nearest neighbors. The **norm** was calculated on the **genre array for each movie** and the **cluster center** was the **movie whose recommendations are to be predicted**. Then the algorithm iterated over the desired number of movies (**specified by the user**) to be recommended.

Hence, the recommended movies were then reported.

Results and Performance Analysis:

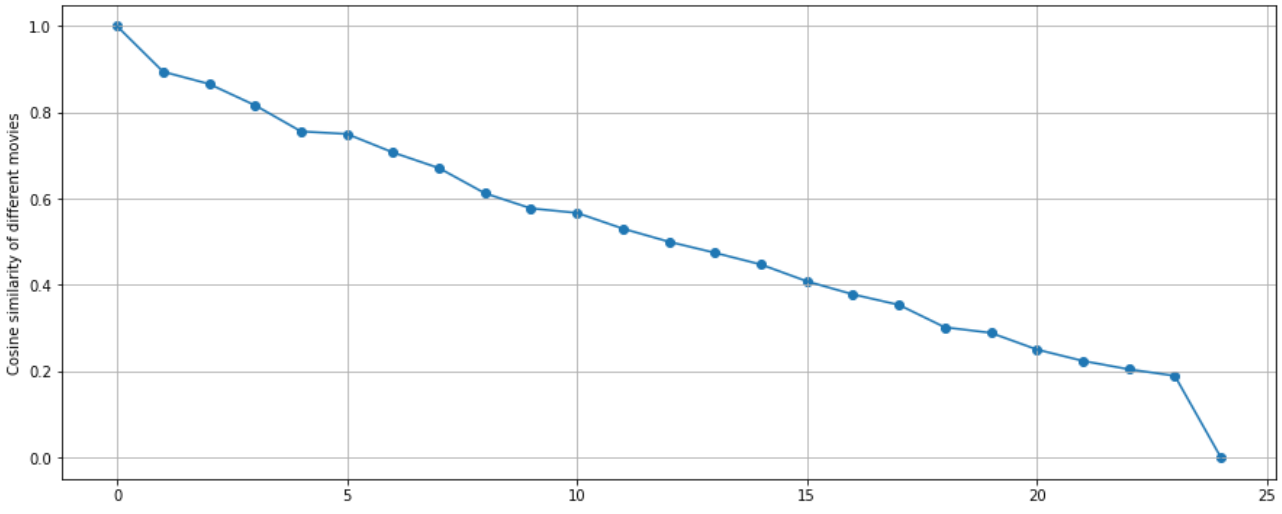
For the uniformity between the results, the recommendation for all the models was done on the movie "Iron Man" and for each model 20 recommendations were made and reported, Except the neural network as it recommends for a particular user.

1) Content based model:

The recommendations:

```
Top 20 Recommendation of Iron Man (Content Based) :
Six-String Samurai
Spacehunter: Adventures in the Forbidden Zone
Fantastic Four
Rampage
Power Rangers
Power/Rangers
Star Wars: Episode I - The Phantom Menace
Bulletproof Monk
Superman
Lost in Space
Superman III
Superman IV: The Quest for Peace
Fantastic Four: Rise of the Silver Surfer
Assassin's Creed
Hulk
Terminator 3: Rise of the Machines
20,000 Leagues Under the Sea
Tron
Independence Day: Resurgence
Demolition Man
```

The following was the trend for the cosine similarity of all the movies in the dataset with respect to the movie "Iron Man":



The movies with the **highest values of the cosine similarity** are reported for the recommendations.

2) Neural network:

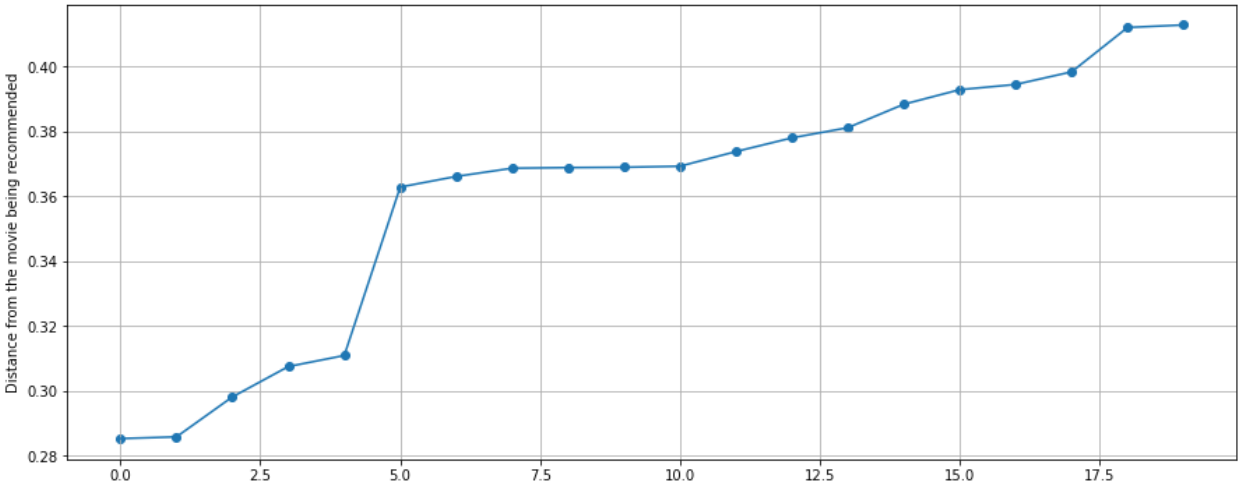
```
The movie recommendations for the user 10 (Neural Network)
Music From Another Room
A Midsummer Night's Dream
Weekend (a.k.a. Le Week-end) (Week End)
Adventures of Mary-Kate and Ashley, The: The Case of the Christmas Caper
Inhale
Perfect Murder, A
Blood Beach
Red Cliff (Chi bi)
Manufacturing Consent: Noam Chomsky and the Media
Starter for 10
Unknown Known, The
Orca: The Killer Whale
Love and a .45
In the Company of Men
Batman & Robin
Sweet Liberty
Amityville Horror, The
She's Out of Control
Head Above Water
10 Cent Pistol
```

3) K-NN Model:

The recommendations are:

```
The reccomendation of Iron Man (KNN Model) :
Avengers, The
Dark Knight, The
WALL·E
Iron Man 2
Avatar
Batman Begins
Star Trek
Watchmen
Guardians of the Galaxy
Up
Inception
Kung Fu Panda
District 9
Sherlock Holmes
X-Men: First Class
Pirates of the Caribbean: At World's End
Thor
Pirates of the Caribbean: Dead Man's Chest
Pirates of the Caribbean: The Curse of the Black Pearl
Star Wars: Episode III - Revenge of the Sith
```

The plot of the distance of each movie in the dataset from the movie that is being recommended:



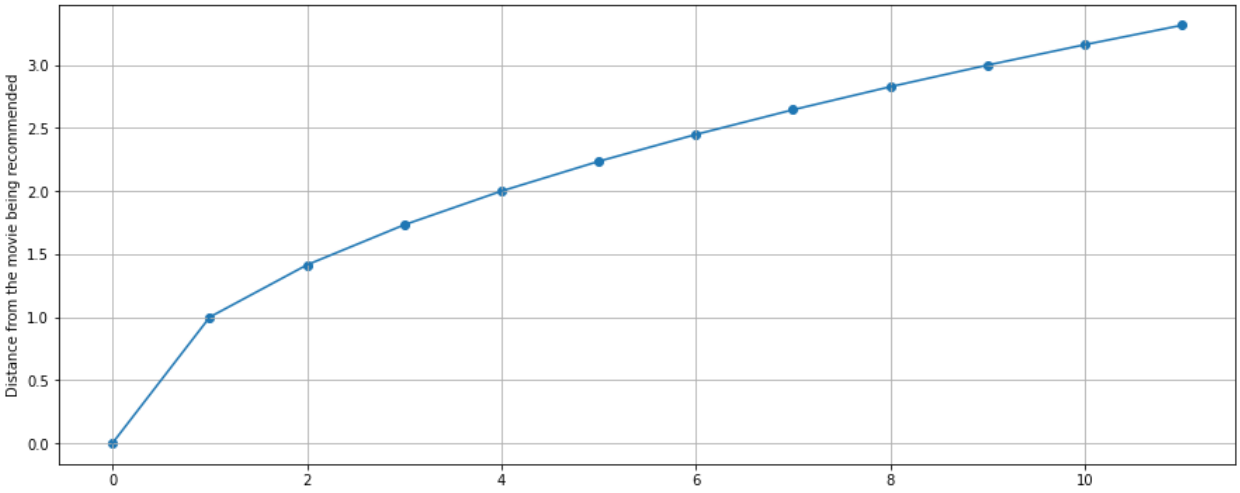
Here the **movies with the least distance** from the **movie that is being recommended** are taken in ascending order and then the top movies are reported as the recommended movies.

4) Genre based Model:

The recommended movies:

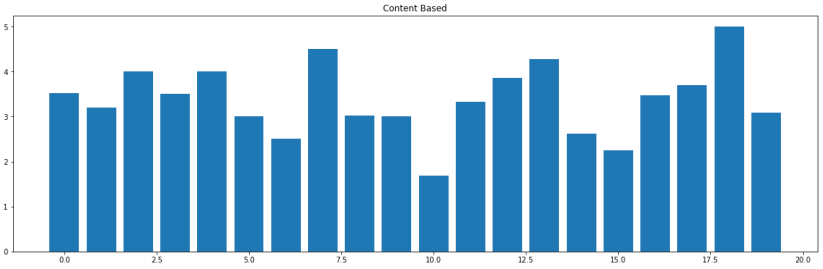
```
The Recommendations for Iron Man are (Genre Based) :
Waterworld
Star Wars: Episode IV - A New Hope
Stargate
Demolition Man
Star Wars: Episode V - The Empire Strikes Back
Star Wars: Episode VI - Return of the Jedi
Star Trek III: The Search for Spock
Lost in Space
Rocketeer, The
Tron
Six-String Samurai
Logan's Run
Star Wars: Episode I - The Phantom Menace
Superman
Superman III
Superman IV: The Quest for Peace
Mad Max
Mad Max Beyond Thunderdome
X-Men
Godzilla 2000 (Gojira ni-sen mireniamu)
```

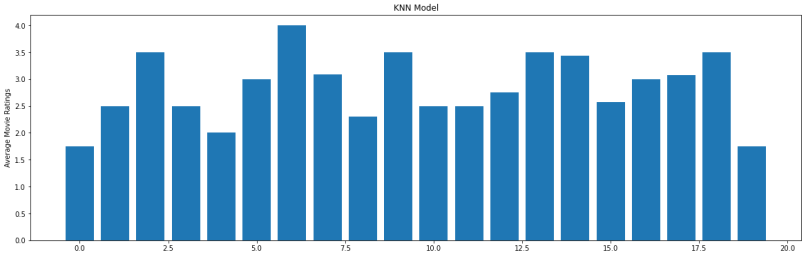
The plot of the distance of the each movie from the movie being recommended:

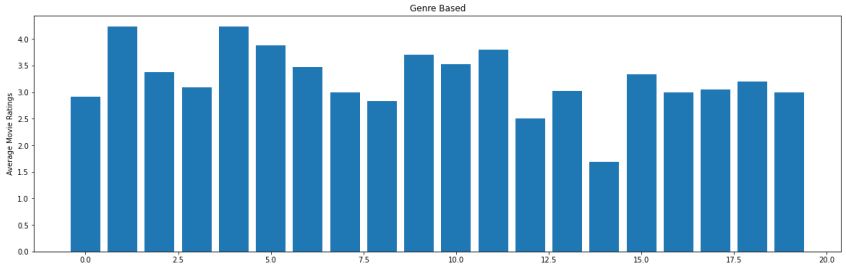


Here the **movies with the least distance** from the **movie that is being recommended** are taken in ascending order and then the top movies are reported as the recommended movies.

Coming to the performance of each model, we can't directly compare them as there **isn't any logical metric for comparing them on the same grounds**. Hence, we made a function that takes in the **recommendations of each model** and then **calculates the average ratings** of that movie from the **ratings dataset** and then stores this data for all the movies. This data is then plotted for visualization and **based on the best average ratings of each model the best model is decided**.





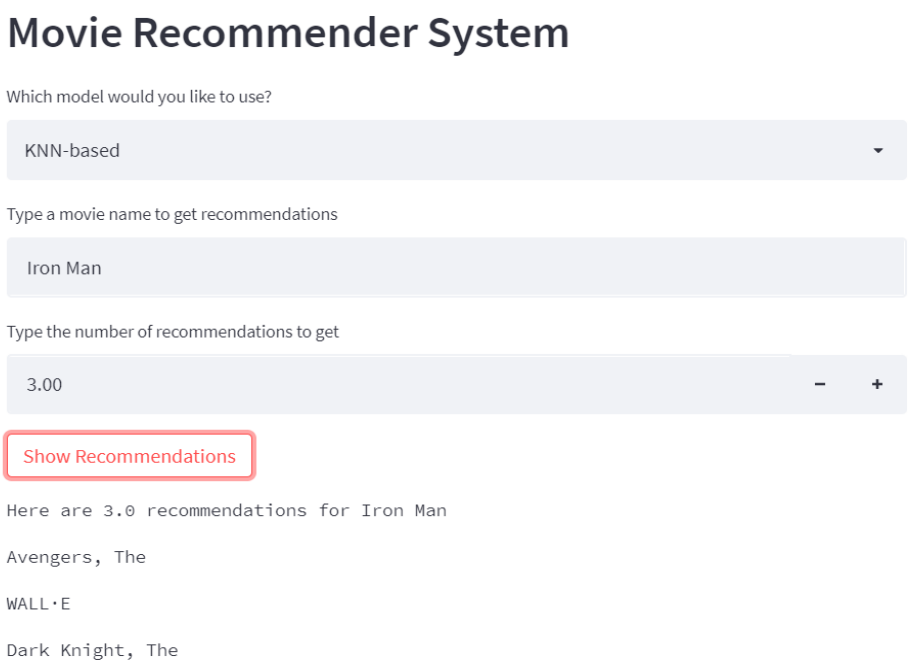


And from the average ratings the best model is:

```
The best performing model for the recommendation of Iron Man is Content Based
The average rating of the recommendation is 3.3775314728448627
```

Deployment on local machine-

For deployment on a local machine I imported two of the original datasets and did the same preprocessing we did in the main file. We made two models, one for KNN and one for Genre based. Now we made a select box in Streamlit for two different models KNN and Genre based on the option selected; the respective output is computed and printed.



Contribution:

Vageesh - ANN model

Content Based model

Priyank - KNN model

Genre based model

Performance Analysis

Rahul - Preprocessing

Deployment on local machine