# DATA MINING PROJECT REPORT
# CSCI-B565

# AMAZON-PRIME VIDEO RECOMMENDATION SYSTEM

**TEAM**

**SAI SRIKAR GANDHE**
**UID:2001097989**
**EMAIL: sgandhe@iu.edu**

**SONAL**
**UID:2001093898**
**EMAIL: sona@iu.edu**

# ABSTRACT

In recent years, recommendation systems have evolved as a solution to the problem of information overload by presenting users with the most appropriate products from a vast amount of data. The goal of online collaborative movie ideas for media products is to assist customers in obtaining their favorite films by pinpointing precisely identical people or films from their prior shared searches. The lack of data makes neighbor choosing more difficult with the rapid development of movies and consumers. This study demonstrates the capability of ML models to recommend the user with the top 5 recommendations when the user starts searching for a particular movie or TV-show with respective to its genre, description, cast, director, etc. In particular, four standards models, such as KNN, Gaussian Naive Bayes, Complement Naive Bayes, and Bernoulli Naive Bayes have been used in this study.

# KEYWORDS

- **Datamining:** Data extraction has become incredibly difficult. Because of the large size of the data set, traditional data analysis tools and methodologies are frequently ineffective. As a result, we employ data mining technology. Data mining is the process of identifying usable information in massive data sources automatically. To improve information retrieval systems, data mining methods have been used.

- **Dataset:** It is a collection of data.

- **Data Preprocessing**: The goal of preprocessing is to convert raw input data into a format suited for further analysis. It entails combining data from several sources, cleaning the data to reduce noise and duplicate observations, and picking records and features.

- **Machine learning**: Machine learning is a branch of artificial intelligence that is widely described as a machine's ability to mimic intelligent human behavior.

- **Recommendation System:** A recommender system, often known as a recommendation system (occasionally replaced by a synonym such as platform or engine), is a type of information filtering system that provides suggestions for items that are most relevant to a certain user.

# INTRODUCTION

Due to the rise of websites like YouTube, Amazon, Netflix, and many others over the past couple decades, recommender systems have become more and more common in our lives. Whether it is in e-commerce (which suggests to customers articles that may interest them) or online advertising, recommender systems are becoming an essential component of our daily online activity (suggest to users the proper contents, matching their tastes). Recommender systems are really critical in some industries as they can generate a huge amount of income when they are efficient or also be a way to stand out significantly from competitors. In this study, we will use various machine learning models to provide the top 5 recommendations to the user when they begin looking for a specific movie or TV show based on the genre, summary, cast, director, etc. This study has specifically used four standard models KNN, Gaussian Naive Bayes, Complement Naive Bayes, and Bernoulli Naive Bayes.

**About the dataset:** This data set contains a list of all shows available for streaming on Amazon Prime. This data was obtained in May 2022 and contains data that is available in the United States.

**Attributes**:

- **Show_id:** It provides the title ID on JustWatch.
- **Type:** Whether it is a TV Show or a Movie.
- **Title**: The name of the Movie or a TV Show.
- **Director:** The name of the director for movie or a TV Show.
- **Cast**: The cast for the Movie or a TV Show.
- **Country**: It provides name of the country.
- **date_added**: It provides the date added for the movie or a tv show.

- **Rating**: It contains the rating of the movie.
- **Duration**: It contains the duration of the movie or a TV Show.
- **listed _in**: It contains the genres of the movie or a TV Show.
- **Description**: It contains the description of the movie or a TV show.

# Methods

We propose building the system on Collaborative and Content-based filtering technique for recommendation system. In our proposed methodology, there are four main components that are
- Data Preprocessing
- Data Visualization
- Feature extraction
- Model implementation
- Accuracy metrics.

## Data Preprocessing:

### Step1:
Data after reading the CSV file.

```
data=pd.read_csv('amazon_prime_titles.csv')
```

```
# It displayes first five rows of the data.
data.head()
```

| | show_id | type | title | director | cast | country | date_added | release_year | rating | duration | listed_in | description |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | s1 | Movie | The Grand Seduction | Don McKellar | Brendan Gleeson, Taylor Kitsch, Gordon Pinsent | Canada | March 30, 2021 | 2014 | NaN | 113 min | Comedy, Drama | A small fishing village must procure a local d... |
| 1 | s2 | Movie | Take Care Good Night | Girish Joshi | Mahesh Manjrekar, Abhay Mahajan, Sachin Khedekar | India | March 30, 2021 | 2018 | 13+ | 110 min | Drama, International | A Metro Family decides to fight a Cyber Crimin... |
| 2 | s3 | Movie | Secrets of Deception | Josh Webber | Tom Sizemore, Lorenzo Lamas, Robert LaSardo, R... | United States | March 30, 2021 | 2017 | NaN | 74 min | Action, Drama, Suspense | After a man discovers his wife is cheating on ... |
| 3 | s4 | Movie | Pink: Staying True | Sonia Anderson | Interviews with: Pink, Adele, Beyoncé, Britney... | United States | March 30, 2021 | 2014 | NaN | 69 min | Documentary | Pink breaks the mold once again, bringing her ... |
| 4 | s5 | Movie | Monster Maker | Giles Foster | Harry Dean Stanton, Kieran O'Brien, George Cos... | United Kingdom | March 30, 2021 | 1989 | NaN | 45 min | Drama, Fantasy | Teenage Matt Banting wants to work with a famo... |

Fig: First Five rows of the data set.

## Step2:

Removed null values present in the dataset for attributes country, date added, rating by using mode values of that specific column. Removed rows of cast and directors having null values

```
# Here we replcaed with mode values.
data['country'] = data['country'].fillna(data['country'].mode()[0])
data['date_added'] = data['date_added'].fillna(data['date_added'].mode()[0])
data['rating'] = data['rating'].fillna(data['rating'].mode()[0])
```

```
data = data.dropna( how='any',subset=['cast', 'director'])
```

Fig: Removing Null values

```
missingno.bar(data,fontsize=10,figsize=(10,5))
plt.title('Checking whether there are still Missing values in each column',fontsize=20)
```

```
Text(0.5, 1.0, 'Checking whether there are still Missing values in each column')
```

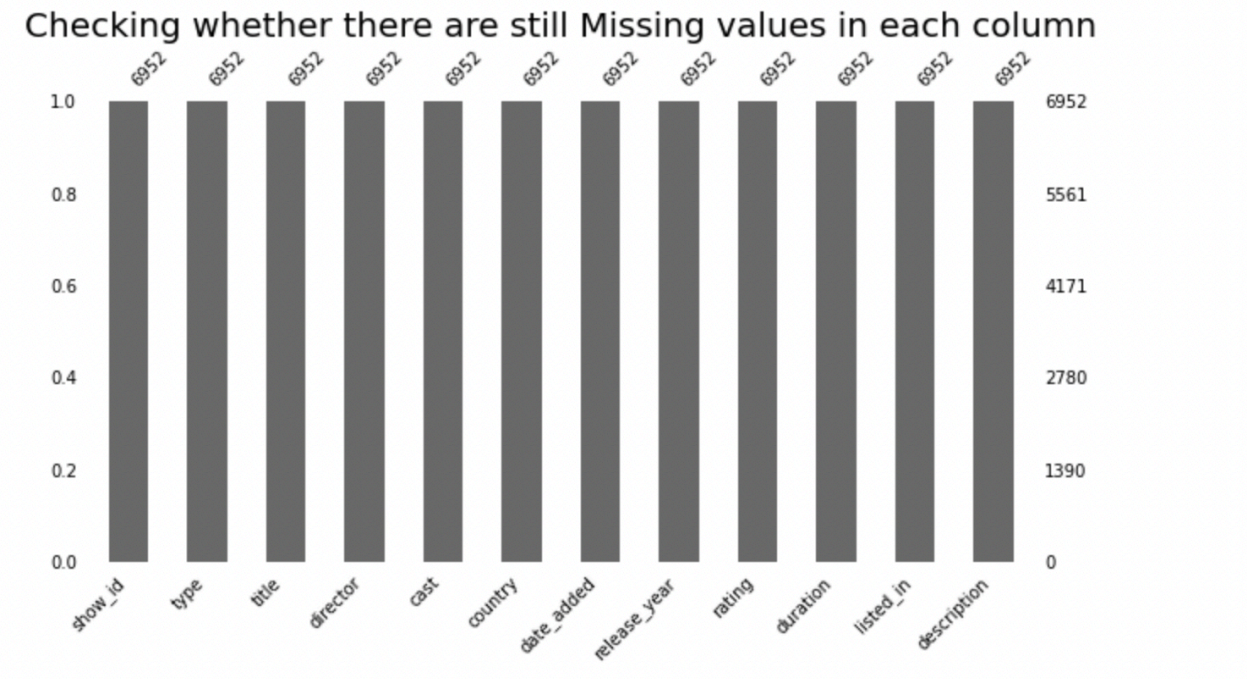



Fig: Checking if there are any null values left

Removed stop words.


Fig: WordCloud for Keywords in data

## Data Visualizations:

We have visualized genres based upon on count (i.e, Movies/Tv Shows)

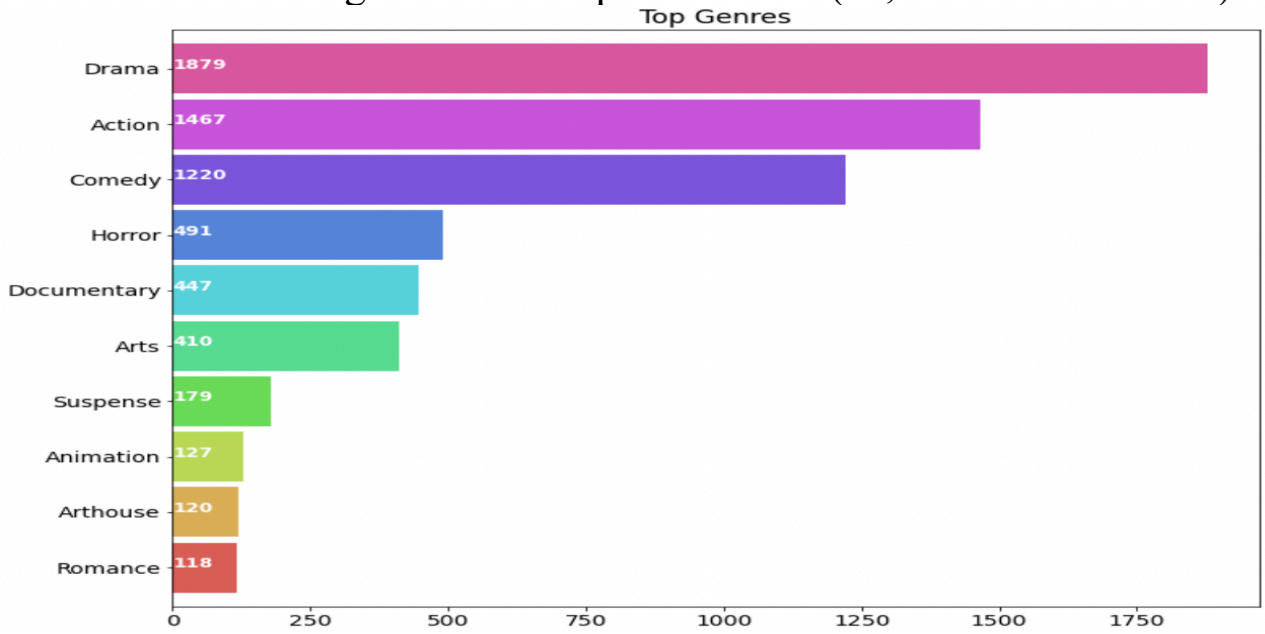



Fig: No of Movies or a tv shows in data set based on genre.

We have generated Top 5 Actors in the data set.

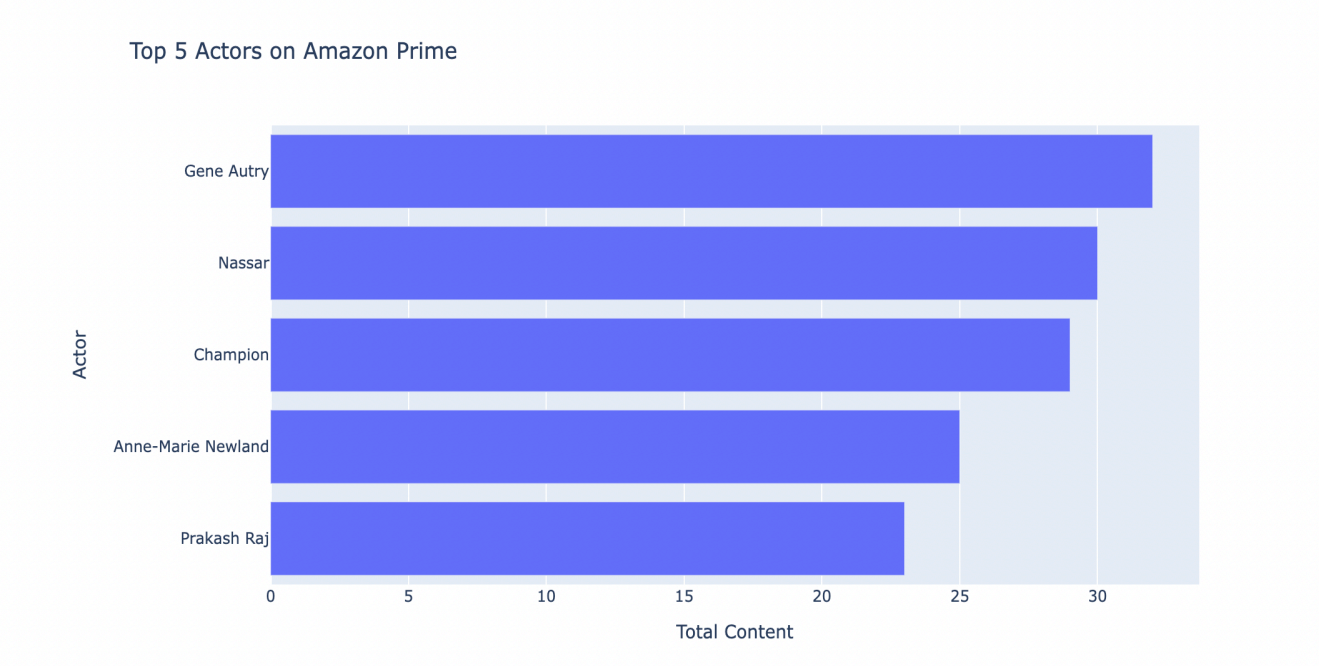



**Fig:** Top 5 Actors in the data set

We have generated rating of different contents on the data set.

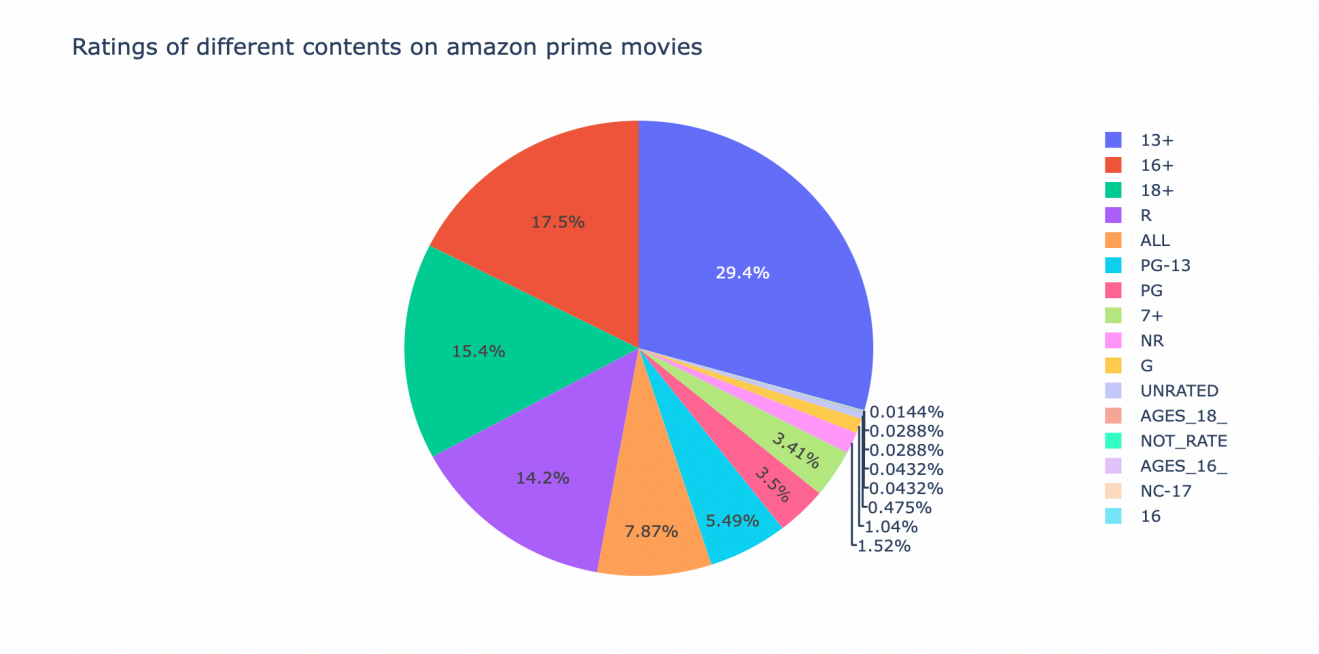



Fig: Pie-graph for movies based upon their ratings

We have generated top 5 directors in the data set.

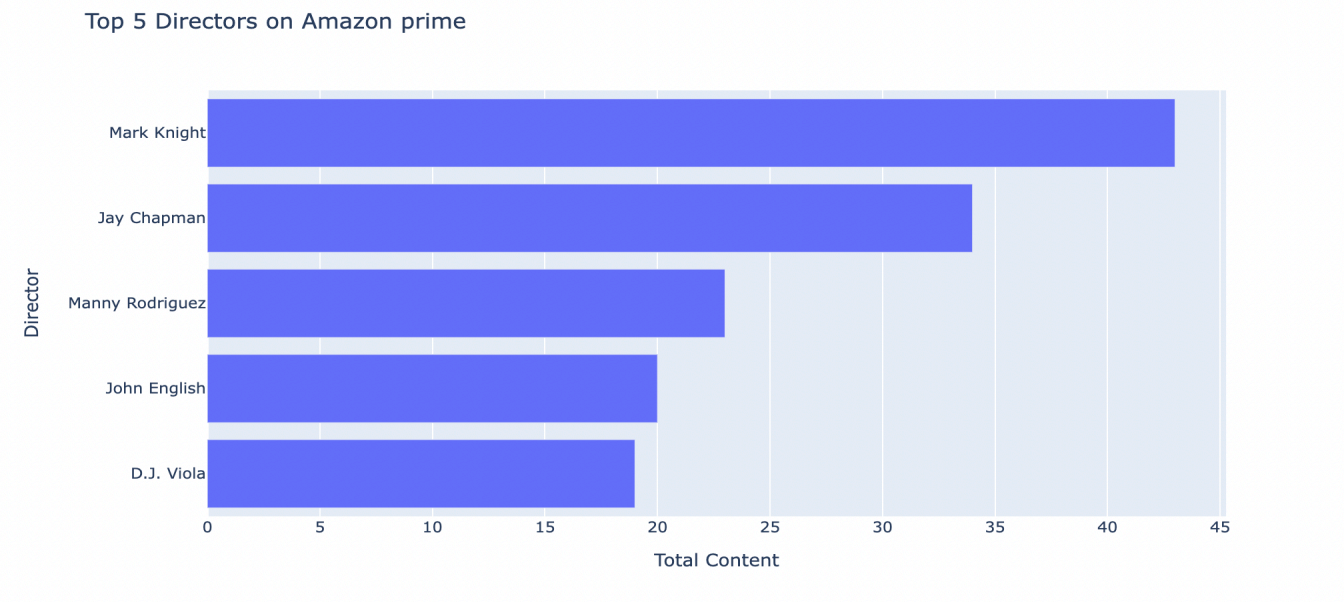



Fig: Top 5 directors in the data set

We have also generated Genres by Countries.

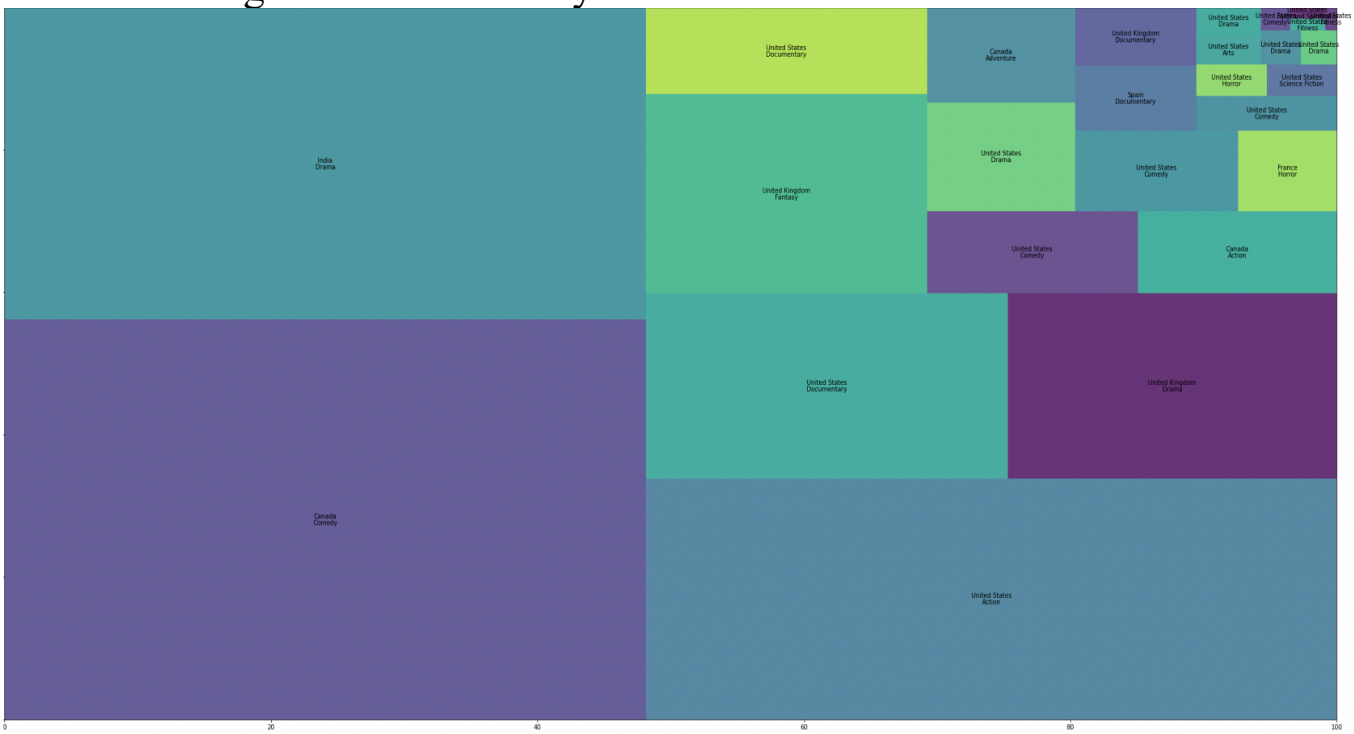



Fig: Genres based on countries

## Feature Extraction:

**Tokenization:** It is the process of segmenting a stream of text into “tokens,” which are words, symbols, and other meaningful components. This is done so that we can consider the tokens as separate parts that combine to create a title, description, cast name, or director name.

**Stop Word Removal:** Stop words are a category of extremely common expressions that, when used in a text, are considered useless because they don’t add any new information. Examples include, among others, “a", "an," "the," "he," "she," "by," and “on."

**TF-IDF:** The TF-IDF statistic gauges a word's applicability to a particular document or set of documents (term frequency-inverse document frequency). It was initially made for searching and recovering documents. But over time, it has also been utilized in recommendation engines and machine learning models.

**Count Vectorizer:** A given text is converted into a vector using the frequency (count) of each word in the full text. This is useful when we have several such texts and want to make each word into a vector (for using in further text analysis).

**Model Implementation:**

**Content-Based Filtering:** Content-based filtering is a form of recommender system that attempt to estimate what a user may like based on previous behavior. By matching keywords and attributes assigned to objects in a database (for example, items in an online marketplace) to a user profile, content-based filtering generates recommendations. Data acquired from a user's actions, such as purchases, ratings (likes and dislikes), downloads, products searched for on a website and/or placed to a basket, and product link clicks, are used to build the user profile.

**Collaborative-Based Filtering:** Collaborative filtering is a technique that employs the reactions of other users to filter out items that a user may be interested in. It functions by searching a large group of people for users who have similar interests to a given user. It considers the products they like and combines them to get a ranked list of recommendations.

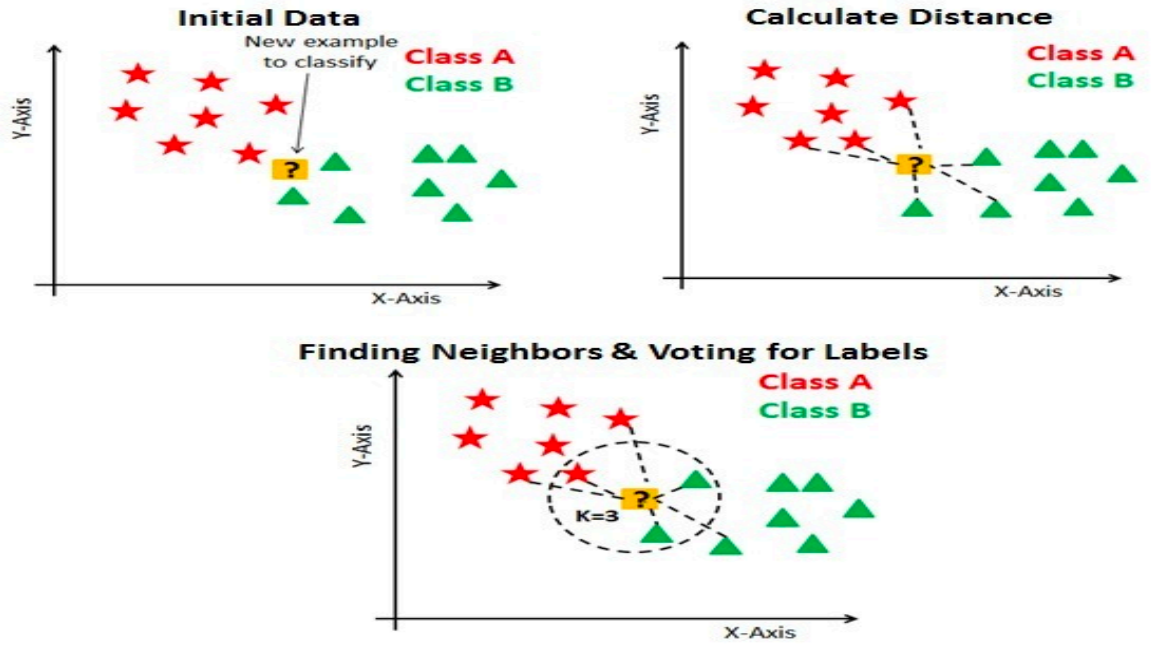



Fig: KNN Classifier

**KNN:** The k-nearest neighbors' algorithm (k-NN) was developed by Evelyn Fix and Joseph Hodges in 1951 and was later enhanced by Thomas

Cover. It is a non-parametric supervised learning method. Regression and classification are two of its uses. A data collection's k closest training examples serve as the input in both scenarios. KNN is an excellent go-to model for implementing item-based collaborative filtering, as well as a solid foundation for developing recommender systems. KNN makes no assumptions about the underlying data distribution, instead relying on item feature similarity.

**Naive Bayes:** Simple "probabilistic classifiers" known as "Naive Bayes classifiers" makes use of the Bayes theorem and build stronger (naive) assumptions about the independence of the characteristics (see Bayes classifier).
Since a Naive Bayes text classifier is based on the Bayes' Theorem, which lets us to compute the conditional probability of occurrence of two events based on the probabilities of occurrence of each individual event, encoding those values is incredibly useful.

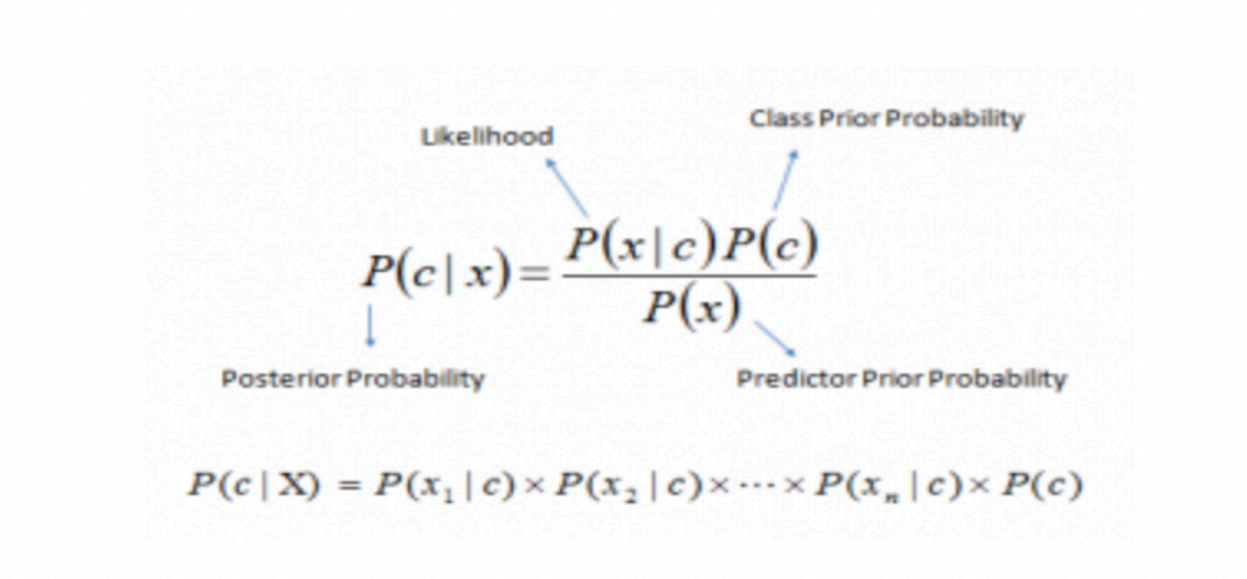



Fig: Naïve Bayes

## Types of Naïve bayes:

**Gaussian Naive Bayes:** In Gaussian Naive Bayes, a Gaussian distribution for the continuous values of each feature is taken for granted. The Gaussian distribution is sometimes known as the Normal distribution. When the feature values are plotted, it produces a bell-shaped curve that is symmetric around the mean of the feature values.

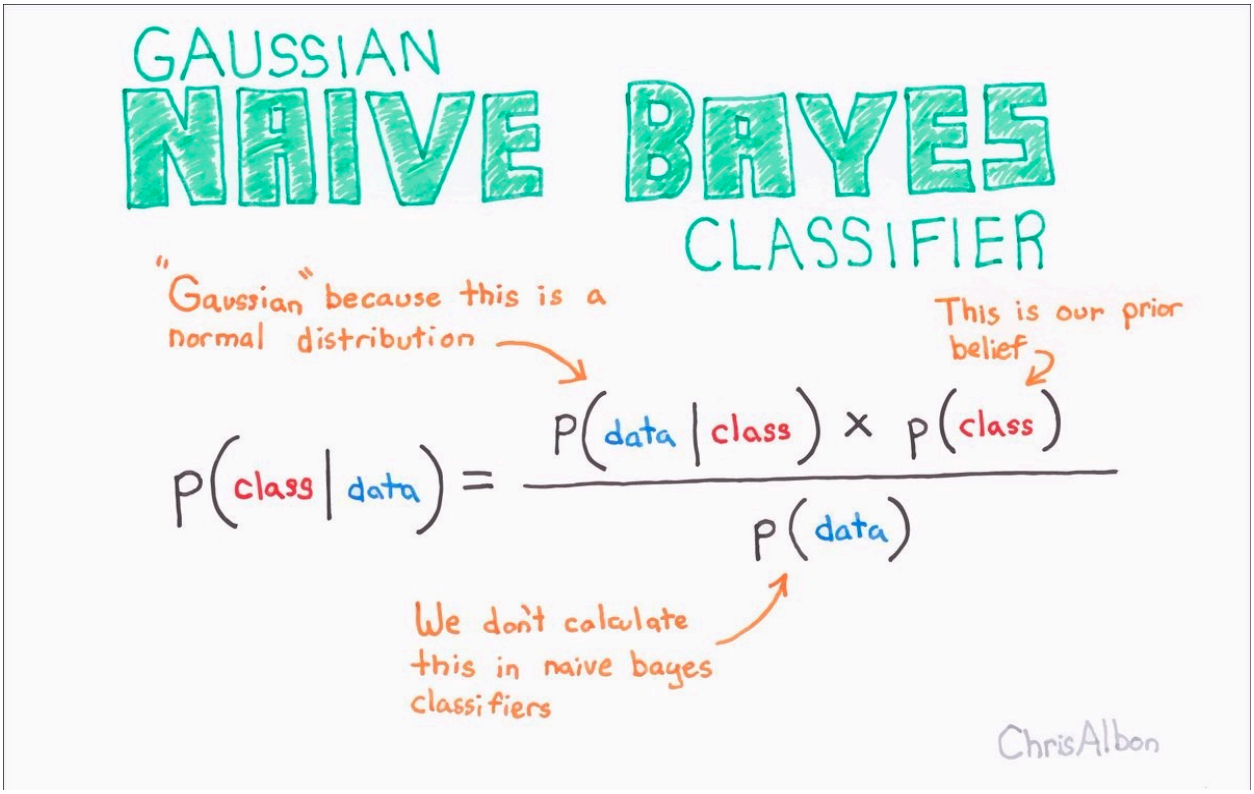
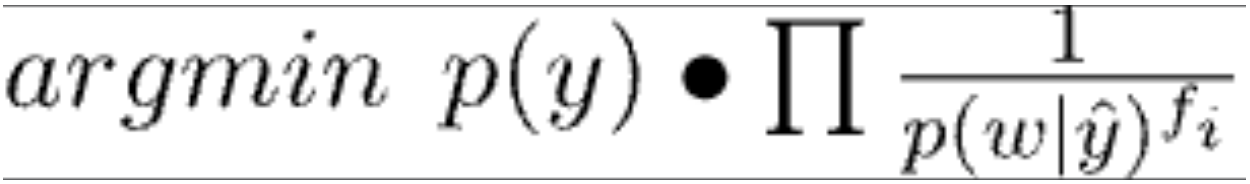



Fig: Gaussian Naïve Bayes

**Complement Naive Bayes:** Complement Naive Bayes is a modification of the Multinomial Naive Bayes method. Unbalanced datasets behave badly with Multinomial Naive Bayes. A dataset that is imbalanced has more samples of one class than any other class. This demonstrates that the sample distribution is inconsistent. Instead of calculating the likelihood of an item belonging to a certain class, we calculate the chance of the item belonging to all classes in complement Naive Bayes. Complement Naive Bayes is named after the literal meaning of the word, complement.



$$argmin\ p(y) \bullet \prod \frac{1}{p(w|\hat{y})^{f_i}}$$

Fig: Complement Naïve Bayes

**Bernoulli Naive Bayes:** The features of the multivariate Bernoulli event model are independent Booleans (binary variables) that describe inputs. This model is widely employed for document classification problems in place of term frequencies, where binary term occurrence characteristics (whether a word appears in a document or not) are used (i.e., frequency of a word in the document).

$$p(\mathbf{x} \mid C_k) = \prod_{i=1}^{n} p_{ki}^{x_i} (1 - p_{ki})^{(1-x_i)}$$

Fig: Bernoulli Naïve Bayes

# RESULTS

Recommendations using Cosine Similarity with Count vectorizer:

```
g_recommendations_new('Global Meltdown', cosine_sim2)
```

```
(7127    Act 4 - Title before 1C onboarding 2
 7128    Act 4 - Title before 1C onboarding 1
 8755                         ACT 2 - TITLE 9
 8756                         ACT 2 - TITLE 8
 8757                         ACT 2 - TITLE 7
 Name: title, dtype: object,
 [(4837, 0.7071067811865475),
  (4838, 0.7071067811865475),
  (6215, 0.7071067811865475),
  (6216, 0.7071067811865475),
  (6217, 0.7071067811865475)])
```

```
g_recommendations_new('ACT 2 - TITLE 9', cosine_sim2)
```

```
(7128    Act 4 - Title before 1C onboarding 1
 8755                         ACT 2 - TITLE 9
 8756                         ACT 2 - TITLE 8
 8757                         ACT 2 - TITLE 7
 9463                   Clip: Act 5 - Title 3
 Name: title, dtype: object,
 [(4838, 1.0), (6215, 1.0), (6216, 1.0), (6217, 1.0), (6791, 1.0)])
```

Fig: Cosine Similarity with Count Vectorizer

```
g_recommendations_new('Act 4 - Title before 1C onboarding 2', cosine_sim2)
```

```
(7128    Act 4 - Title before 1C onboarding 1
 8755                         ACT 2 - TITLE 9
 8756                         ACT 2 - TITLE 8
 8757                         ACT 2 - TITLE 7
 9463                   Clip: Act 5 - Title 3
 Name: title, dtype: object,
 [(4838, 1.0), (6215, 1.0), (6216, 1.0), (6217, 1.0), (6791, 1.0)])
```

Fig: Cosine Similarity with Count Vectorizer

Recommendations using Cosine Similarity with TF-IDF vectorizer:

```
g_recommendations_new('The Grand Seduction',tfdf_similarity)
```

```
(151       War of Likes
 156             Walter
 164    Waiting on Mary
 201     Valentine DayZ
 275     Traci Townsend
 Name: title, dtype: object,
 [(84, 1.0), (89, 1.0), (93, 1.0), (114, 1.0), (158, 1.0)])
```

```
g_recommendations_new('Take Care Good Night',tfdf_similarity)
```

```
(36             You're Not You
 73     Words On Bathroom Walls
 83            Within Our Gates
 122          Where Hands Touch
 124    When in your reflection
 Name: title, dtype: object,
 [(21, 1.0), (43, 1.0), (48, 1.0), (68, 1.0), (70, 1.0)])
```

Fig: Cosine Similarity with TF-IDF Vectorizer

Naïve Bayes:

Gaussian Naïve Bayes:

| | precision | recall | f1-score |
|---|---|---|---|
| 0 | 0.64 | 0.90 | 0.75 |
| 1 | 0.80 | 0.44 | 0.57 |
| accuracy | | | 0.68 |
| macro avg | 0.72 | 0.67 | 0.66 |
| weighted avg | 0.72 | 0.68 | 0.67 |

Fig: Classification Report for Gaussian Naïve Bayes

Complement Naïve Bayes:

| | precision | recall | f1-score |
|---|---|---|---|
| 0 | 0.75 | 0.90 | 0.82 |
| 1 | 0.86 | 0.67 | 0.75 |
| accuracy | | | 0.79 |
| macro avg | 0.80 | 0.78 | 0.78 |
| weighted avg | 0.80 | 0.79 | 0.79 |

Fig: Classification Report for Complement Naïve Bayes

Bernoulli Naïve Bayes:

| | precision | recall | f1-score |
|---|---|---|---|
| 0 | 0.80 | 0.80 | 0.80 |
| 1 | 0.78 | 0.78 | 0.78 |
| accuracy | | | 0.79 |
| macro avg | 0.79 | 0.79 | 0.79 |
| weighted avg | 0.79 | 0.79 | 0.79 |

Fig: Classification Report for Bernoulli Naïve Bayes

KNN Classifiers:

Recommendation using KNN:

```
recommend('Monster Maker')
```

```
Selected Movie:  Monster Maker

Recommended Movies:

Revenge in Kind | Genres: 'Drama'
David's Mother | Genres: 'Drama'
Entanglement | Genres: 'Drama'
Where Hands Touch | Genres: 'Drama'
Sundown | Genres: 'Drama'
```

Fig: Recommendation using KNN

| | precision | recall | f1-score |
|---|---|---|---|
| 0 | 0.67 | 0.89 | 0.76 |
| 1 | 0.86 | 0.60 | 0.71 |
| accuracy | | | 0.74 |
| macro avg | 0.76 | 0.74 | 0.73 |
| weighted avg | 0.77 | 0.74 | 0.73 |

Fig: Classification Report for KNN

# CONCLUSION

For recommending movies by using cosine similarity with tf-idf vectorizer gave us 1 similarity and with count vectorizer we got good similarity which is greater than 0.7.

For KNN model we have obtained 74% accuracy with top 5 recommendations. For Naïve bayes models we have obtained 79% accuracy for both Complement Naïve Bayes, Bernoulli Naïve Bayes and for Gaussian Naïve Bayes we have obtained 68% accuracy.

Therefore, we conclude that Complement Naïve Bayes and Bernoulli Naïve Bayes performs better with an accuracy of 79% when compared to KNN Classifier and Gaussian Naïve Bayes.

# **REFERENCES**:


- https://www.kaggle.com/datasets/victorsoeiro/amazon-prime-tv-shows-and-movies?select=titles.csv
- https://scikit-learn.org/stable/modules/naive_bayes.html#gaussian-naive-bayes
- https://scikit-learn.org/stable/modules/naive_bayes.html#complement-naive-bayes
- https://scikit-learn.org/stable/modules/naive_bayes.html#bernoulli-naive-bayes
- https://www.kaggle.com/code/niharika41298/recommendation-systems-in-a-nutshell
- Textbook: Introduction to Data Mining
- https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.datacamp.com%2Ftutorial%2Fk-nearest-neighbor-classification-scikit-learn&psig=AOvVaw3t6hW9B0sJk9GaW_3B00qG&ust=1670982649120000&source=images&cd=vfe&ved=0CA4QjRxqFwoTCNjilMu99fsCFQAAAAAdAAAAABAE
- https://www.google.com/url?sa=i&url=https%3A%2F%2Fuc-r.github.io%2Fnaive_bayes&psig=AOvVaw1gM9mHeChCD79mMyoabc5n&ust=1670982752439000&source=images&cd=vfe&ved=0CA8QjRxqFwoTCJC26Py99fsCFQAAAAAdAAAAABAE
- https://www.google.com/url?sa=i&url=https%3A%2F%2Fbecominghuman.ai%2Fnaive-bayes-theorem-d8854a41ea08&psig=AOvVaw1gM9mHeChCD79mMyoabc5n&ust=1670982752439000&source=images&cd=vfe&ved=0CA8QjRxqFwoTCJC26Py99fsCFQAAAAAdAAAAABAJ
- https://www.geeksforgeeks.org/complement-naive-bayes-cnb-algorithm/