Movie Genre Classification

Dommeti Surya Vamsi
Computer Science Engineering
Amrita Vishwa Vidyapeetham
cb.en.u4cse20019

Kaki Sri Satvika
Computer Science Engineering
Amrita Vishwa Vidyapeetham
cb.en.u4cse20029

Meenakshi P
Computer Science Engineering
Amrita Vishwa Vidyapeetham
cb.en.u4cse20039

Pranav M
Computer Science Engineering
Amrita Vishwa Vidyapeetham
cb.en.u4cse20050

Yadava Krishnaa P
Computer Science Engineering
Amrita Vishwa Vidyapeetham
cb.en.u4cse20072

***Abstract*— Movie genre classification is a task in which a machine learning model is trained to predict the genre of a movie based on its features. These features can include information about the plot, characters, dialog, and other elements of the movie. The model is trained using a dataset of movies, where the genre of each movie is known. The goal of the project is to build a model that is able to accurately predict the genre of a movie based on its features. This can be useful for various applications, such as recommending movies to users based on their preferred genres or improving the organization and search-ability of movies within a media library. The project may involve using techniques such as feature engineering, model selection in order to optimize the performance of the model.**

I. Introduction

Movie genre classification is the task of classifying a movie into one or more predefined categories, such as action, comedy, drama, horror, and so on. This task is important for a number of applications, including recommendation systems, content-based filtering, and data mining. There are a number of approaches that can be used to classify movies into genres, including the use of machine learning algorithms, natural language processing techniques, and manual annotation. Some common features that are used to classify movies into genres include the movie's plot, the actors and actresses, the settings, and the dialog. In this way, movie genre classification can help users discover new movies that they might enjoy and can also help movie studios and distributors better understand the preferences of their audience.

II. Dataset Description

This dataset is acquired from Kaggle is all about Movies/ TV Shows that are available on Amazon Prime, Hotstar, Netflix, Bflix. It consists of 9 columns: Movie Title, Cast, Brief Description Of The Plot, Duration, Rating On IMDB, Voted By People, Year, Genre, Certificate.There is a total of 7912 unique movie titlesAll data is taken from IMDB website by web scraping and it contains total of 53 genre

III. Data Preprocessing

Data preprocessing is the process of preparing a dataset for analysis. In the context of movie genre classification using natural language processing (NLP), data preprocessing might involve the following steps:

1) <u>Tokenization</u>: This involves dividing the movie descriptions into individual words or smaller units called tokens. This is typically done using techniques such as splitting the text on spaces or using regular expressions to identify word boundaries.

2) <u>Removing stop words</u>: This involves removing common words such as "the" and "and" that do not convey much meaning. These words are often removed because they occur frequently in the text but do not provide much information for the purposes of classification.

3) <u>Stemming or lemmatization</u>: This involves reducing words to their base form, so that words with the same meaning but different inflections are treated the same (e.g., "jumping" and "jumps" are both reduced to "jump"). This can be useful for reducing the dimensionality of the dataset and improving the performance of the classification model.

4) <u>Encoding</u>: This involves converting the words or tokens into numerical representations that can be used by machine learning algorithms. There are a number of ways to do this, such as using one-hot encoding or term frequency-inverse document frequency (TF-IDF).

5) <u>Normalization</u>: This involves scaling the numerical representations of the words to a common range, such as between 0 and 1. This can be useful for ensuring that the features are on a similar scale and do not dominate the model due to their large values.

By preprocessing the movie descriptions, it is possible to extract relevant features that can be used to classify the movies into different genres. This can be an important step in building an effective movie genre classification model using NLP techniques.

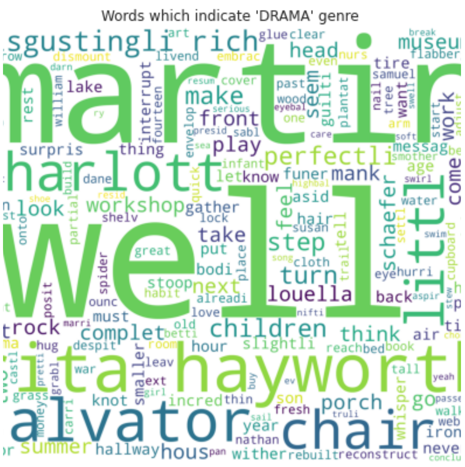
Words which indicate 'DRAMA' genre

Figure 1- Word Cloud for Drama

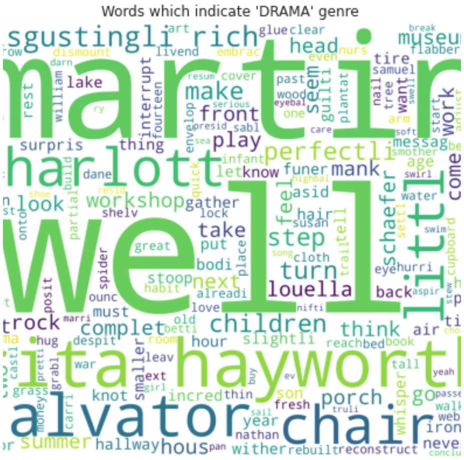

Figure 2- Word Cloud for Drama

Words which indicate 'ACTION' genre
neo
readi
morpheu
ben
yoda
winston
tantz
know
jedi
brownston

Figure 3- Word Cloud for Action

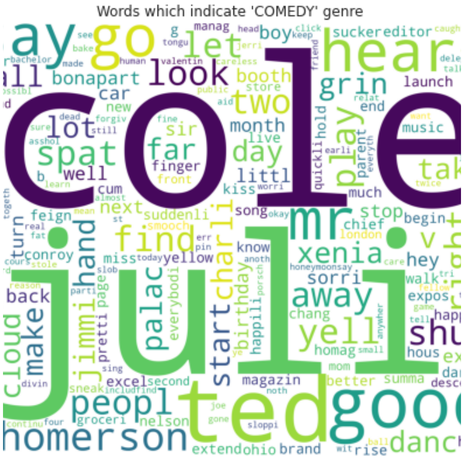

Figure 4- Word Cloud for Comedy

IV. Expolaratory Data Analysis (EDA)

Exploratory data analysis (EDA) is a method used to analyze and synthesize a dataset to gain a better understanding of its characteristics and relationships. It is commonly used as the first step in the data science process before building more formal models or conducting statistical tests. EDA can be useful for movie genre classification as it helps identify patterns and trends in the data that can inform the development of machine learning models. Techniques used in EDA for movie genre classification include data visualization, summarization, and identifying outliers. By performing EDA on a movie genre classification dataset, it is possible to get a better understanding of the data and inform the development of more sophisticated classification models.

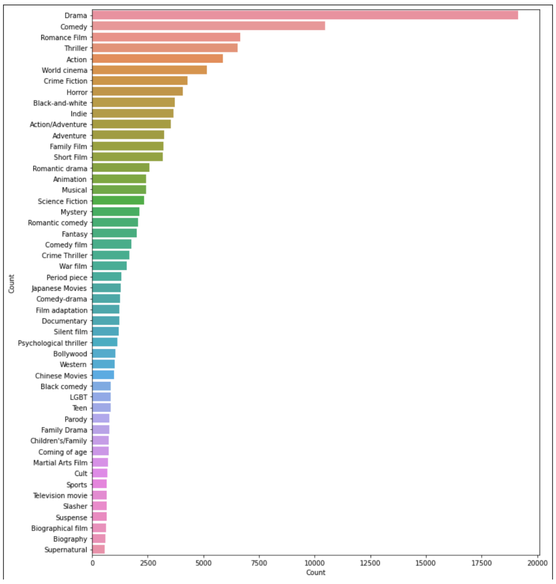

Figure 5- EDA

V. Feature reduction

Principal Component Analysis (PCA): By using PCA, data with more number of features are projected to a new space with less dimensions. PCA finds the covariance among the extracted features to find the pattern in the data and it can be calculated using the Eq.

$$C(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

Then it calculates the eigenvectors and eigenvalues from the covariance matrix. k largest eigen vectors are chosen by arranging the eigenvalues corresponding to them in decreasing order where k is the new dimension. This algorithm creates the projection matrix using chosen eigenvectors. Finally PCA converts the d-dimension features to k-dimension features with the help of projection matrix.

In the context of movie genre classification, PCA might be applied to a dataset of movies to reduce the number of features while preserving as much of the relevant information as possible. For example, if the movie dataset has a large number of features, such as the plot, the actors and actresses, the settings, and the dialog, PCA could be used to identify a smaller number of principal components that capture most of the variance in the data.

To classify a new movie, the PCA-transformed features of the movie could be used as input to a classifier, such as a support vector machine or a decision tree. The classifier would then make a prediction based on the lower-dimensional representation of the movie's features.

PCA can be useful for movie genre classification by reducing the dimensionality of the dataset, which can improve the performance and interpretability of the *classification model. However, it can also lose some of the original information in the data, so it is important to carefully evaluate the trade-off between dimensionality reduction and information loss.*

VI. DATA VISUALISATION

Data visualization is the process of creating graphical representations of data in order to better understand and communicate its characteristics and trends. It can be a powerful tool for exploring and understanding data, as it allows you to see patterns and relationships that may not be immediately apparent from looking at raw data. There are many different types of data visualizations, including line graphs, bar charts, scatter plots, and heat maps, among others

A. *Data Visualisation used in this project:*

1) Box Plot: A box plot, also known as a box-and-whisker plot, is a graphical representation of a dataset that provides a summary of the distribution, including the median, interquartile range, and upper and lower quartiles.

2) Regression Plot: A regression plot is a graphical representation of the relationship between two variables, showing the trend of one variable as a function of the other. It is commonly used to understand the strength and direction of the relationship between the variables, and to identify any potential outliers in the data.

3) Violin Plot: A violin plot is a graphical representation of a dataset that shows the distribution of the data across different levels of a categorical variable, with the thickness of the plot indicating the relative frequency of the data points. It is a combination of a box plot and a kernel density plot, and can be useful for visualizing the distribution of the data and comparing it across different groups.

4) Heat Map: A heat map is a graphical representation of data where the values are represented as colours, with higher values being indicated by warmer colours and lower values being indicated by cooler colours. It is commonly used to visualize the distribution and intensity of data across a two-dimensional space.

5) Histogram: A histogram is a type of bar graph that displays the frequency of data within certain intervals. It is used to show the distribution of numerical data. It is a graphical representation of the data distribution, showing how often each value or range of values occurs.

6) Count Plot: A count plot is a graphical representation of a dataset that shows the frequency or count of the data within different categories or bins. It is similar to a histogram, but rather than showing the distribution of the data, it shows the count of the data within each category. Count plots can be useful for understanding the underlying distribution of categorical data and for comparing the counts of different categories.

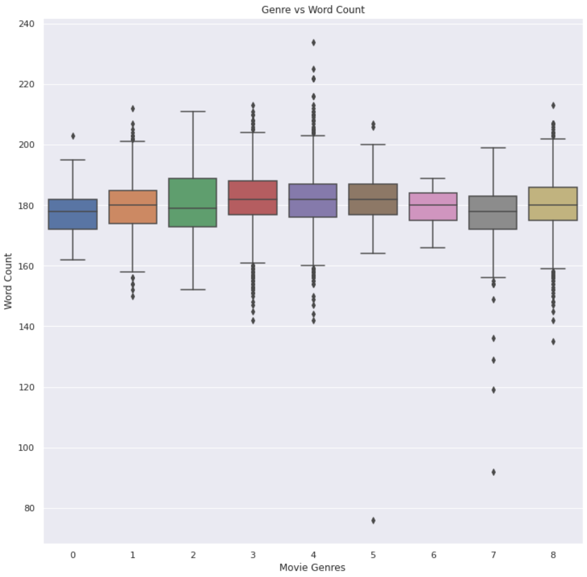

Figure 6 - Box Plot

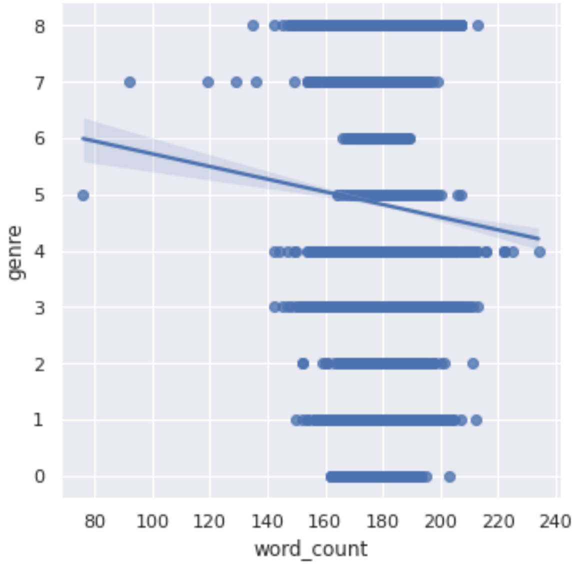

Figure 7 - Regression Plot

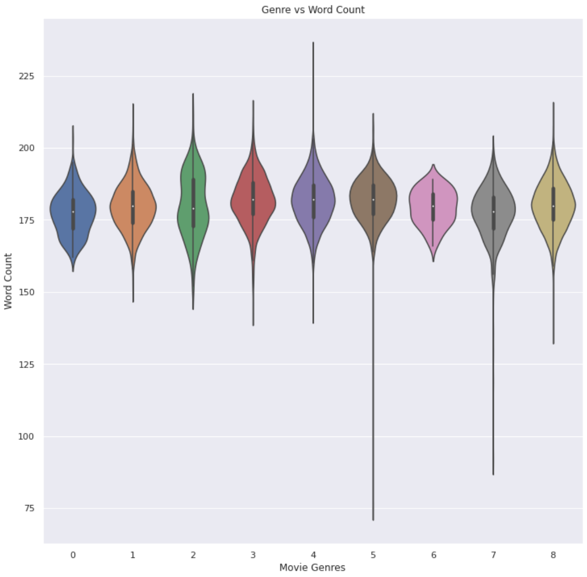

Figure 8 - Violin Plot

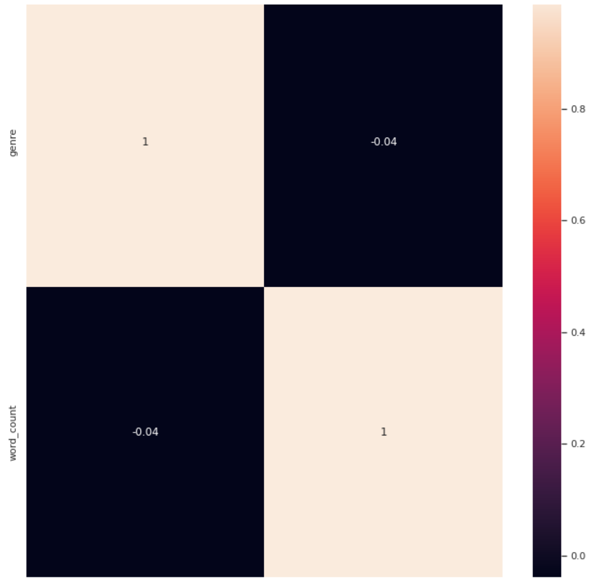

Figure 9 - Heat Map

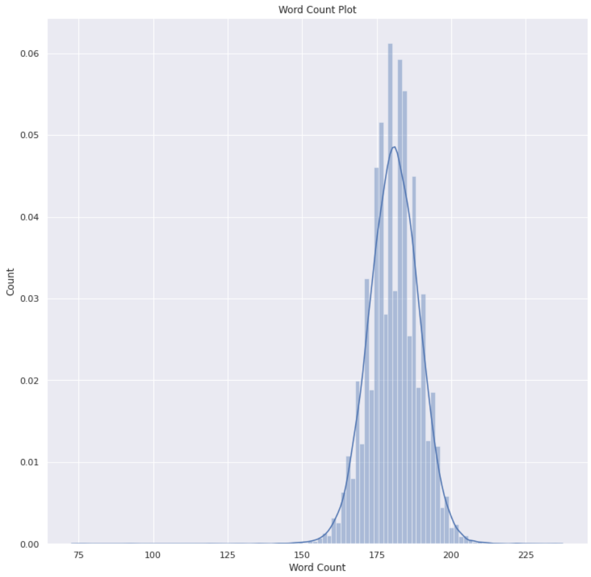

Figure 10 - Histogram

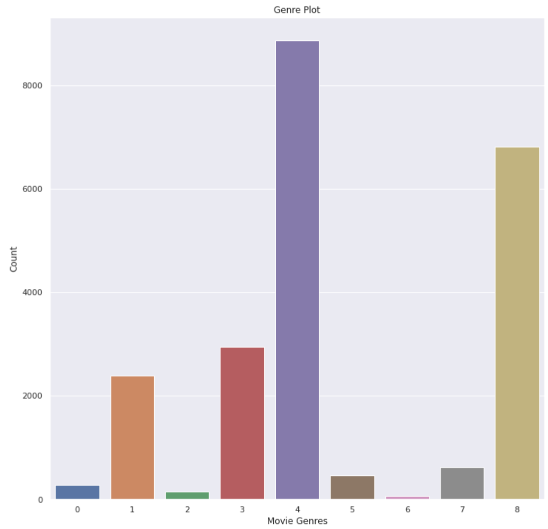

Figure 11 - Count Plot

VII. Feature Engineering

Feature engineering is the process of extracting and creating features (i.e., input variables) from raw data that can be used in machine learning models. It is a crucial step in the machine learning process, as the quality and relevance of the features can significantly affect the performance of the model.

Some common techniques for feature engineering include:

- Data cleaning: This involves removing or correcting invalid or missing data.
- Feature selection: This involves selecting a subset of the most relevant features from the dataset.
- Feature extraction: This involves creating new features from the raw data, such as by combining or transforming existing features.
- Feature scaling: This involves transforming the features to a common scale, such as by normalizing or standardizing them.

Feature engineering can be a time-consuming and iterative process, but it is essential for building high-performing machine learning models.

A. *Feature Engineering used in this project:*

a) ***Naïve Bayes****: Naive Bayes is a machine learning algorithm that can be used for classification tasks. It is based on the idea of using Bayes' Theorem to make predictions based on the probability of certain events occurring.*

In the context of movie genre classification, a Naive Bayes classifier might be trained on a dataset of movies, with each movie being labeled with one or more genres. The classifier would learn the probability of certain words or phrases occurring in movies belonging to each genre. For example, the classifier might learn that the word "gun" is more likely to occur in action movies, while the word "romance" is more likely to occur in romantic comedies.

To classify a new movie, the classifier would use the probabilities learned during training to compute the probability that the movie belongs to each genre. The genre with the highest probability would be chosen as the predicted genre for the movie.

Naive Bayes is a simple and fast algorithm that can be effective for movie genre classification, especially when working with large datasets. However, it makes the assumption that the features (i.e., the words or phrases) are independent of one another, which may not always be the case in practice.

The probability equation is given in

$$P(X \mid Y) = P(Y \mid X) * P(X) / P(Y)$$

Where P(X | Y) indicates the posterior probability

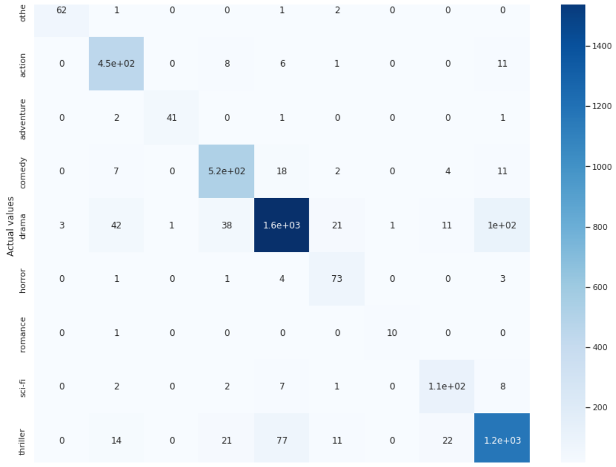

Figure 12- Naive Bayes Confusion Matrix

b) **Decision Tree**: This algorithm divides the features in the dataset to form tree. A decision tree includes the root node, decision node and leaf node. Entropy and Information gain for all the features are calculated. The formula to calculate Entropy and information gain are given in

$$E = \sum_{i=1}^{n} -f_i * \log_2 f_i$$

$$gain(d, x) = E(d) - E(d, x)$$

where fi is the number of samples of label i in a node, n denotes the number of distinct labels, d represents target label, x represents the attribute to be divided on. The attribute which has high Information gain or low Entropy is selected as the root node. This calculation is made recursively for subdata to construct decision tree. The leaf node indicates the output of the new data.

In the context of movie genre classification, a decision tree classifier might be trained on a dataset of movies, with each movie being labeled with one or more genres. The classifier would learn to make predictions based on the features of the movies, such as the plot, the actors and actresses, the settings, and the dialog.

To classify a new movie, the classifier would follow the decisions in the tree based on the features of the movie, until it reaches a leaf node, which would represent the predicted genre for the movie.

Decision trees are simple and easy to interpret, and they can be effective for movie genre classification. However, they can be prone to overfitting, especially if the tree becomes too deep, and they may not be as accurate as some other machine learning algorithms.

c) **SVM**: SVM classifier divides n-dimensional space to different labels by using boundary. The extreme points, otherwise known as support vectors are used to generate the decision boundaries. Position of new input point in the n-dimensional space determines output class of the new point. For both regression and classification, the SVM classifier can be used. It is difficult to construct the hyperplane or the decision boundary for non-linear data or high dimensional data using SVM. Kernel functions are used in SVM to solve this issue by linear classifier. The different SVM kernel used are Sigmoid kernel, Polynomial kernel and Gaussian radial basis function kernel.

$$K(X_i, X_j) = (X_i \,.\, X_j + 1)^d$$
$$K(X_i, X_j) = exp(-r\,||X_i - X_j||^2)$$
$$K(x, y) = \tanh(\gamma x^T y \,+\, r)$$

In the context of movie genre classification, an SVM classifier might be trained on a dataset of movies, with each movie being labeled with one or more genres. The classifier would learn to classify the movies based on their features, such as the plot, the actors and actresses, the settings, and the dialog.

To classify a new movie, the classifier would use the hyperplane learned during training to predict the genre of the movie. If the movie is on one side of the hyperplane, it will be classified as one genre; if it is on the other side, it will be classified as a different genre.

SVMs are powerful and accurate algorithms that can be effective for movie genre classification, especially when working with high-dimensional data. However, they can be sensitive to the choice of kernel function and other hyperparameters, and they may not perform as well on datasets with large amounts of noise or overlap between classes.

d) **K-Nearest Neighbors**: The k closest neighbors are found using this approach nevertheless of target class. The number of nearest neighbors to be considered is passed as the parameter and represented as k. The distance between the new input and the trained points are calculated. The minimum distanced k trained points are considered and the class of that points are counted. The class with maximum number of points close to the new point is given as the output. Euclidean distance is used by KNN to find the distance between points. The formula to find distance is given in Eq

$$d_{st} = \sqrt{\sum_{j=1}^{n} (x_{sj} \,-\, y_{tj})^2}$$

In the above equation x indicates the new input point, y indicates the trained point and the number of features is given as n.

In the context of movie genre classification, a KNN classifier might be trained on a dataset of movies, with each movie being labeled with one or more genres. The classifier would learn to classify the movies based on their features, such as the plot, the actors and actresses, the settings, and the dialog.

To classify a new movie, the classifier would find the K nearest neighbors to the movie in the training set, based on the similarity of their features. The predicted genre for the movie would be the most common genre among the K nearest neighbors.

KNN is a simple and intuitive algorithm that can be effective for movie genre classification, especially when working with high-dimensional data. However, it can be computationally expensive, especially for large datasets, and it may not perform as well as some other algorithms on datasets with a large number of features or classes.

e) **Random Forest**: A random forest is an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. The random forest algorithm can be summarized as follows:

1. Select N random samples from the training set.
2. Build a decision tree for each sample.
3. Choose the number M of trees you want in your forest.
4. For each new sample, predict the class by the majority vote of its M trees.

Mathematically, the prediction made by a single decision tree can be expressed as follows:

$$h(x) = \sum (w_j * x_j) + b$$

where x is the input data, wj is the weight assigned to feature j, and b is the bias term.

In a random forest, the prediction is made by aggregating the predictions of each individual tree (h(x)), such that the final prediction is given by:

$$h(x) = \frac{1}{M} * \sum \left(h(x)^1 + h(x)^2 + \ldots + h(x)^M\right)$$

where M is the number of trees in the forest.

In the context of movie genre classification, a Random Forest classifier might be trained on a dataset of movies, with each movie being labeled with one or more genres. The classifier would learn to classify the movies based on their features, such as the plot, the actors and actresses, the settings, and the dialog.

To classify a new movie, the classifier would pass the movie through each of the decision trees in the forest, and the predicted genre for the movie would be the one that received the most votes from the individual trees.

Random Forest is a powerful and accurate algorithm that can be effective for movie genre classification, especially when working with high-dimensional data. It is resistant to overfitting and can handle a large number of features, but it can be computationally expensive and may not perform as well on datasets with a large number of classes.

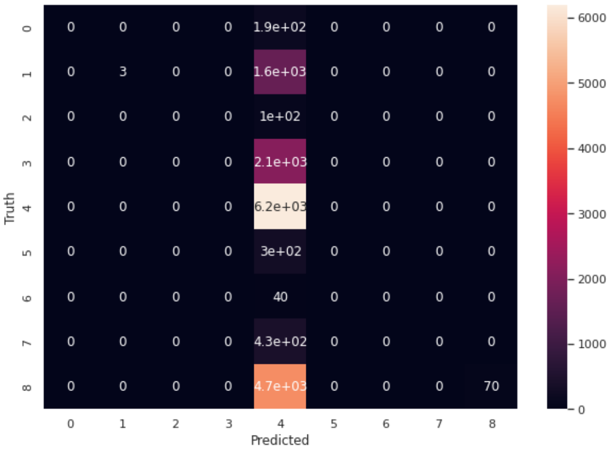

Figure 10- Random Forest Confusion Matrix

f) **Perceptron**: Perceptron is a type of machine learning algorithm that can be used for classification tasks. It works by making predictions based on a linear combination of the input features, with the coefficients of the combination being learned from the training data.

In the context of movie genre classification, a Perceptron classifier might be trained on a dataset of movies, with each movie being labeled with one or more genres. The classifier would learn to classify the movies based on their features, such as the plot, the actors and actresses, the settings, and the dialog.

To classify a new movie, the classifier would compute the linear combination of the movie's features and apply a threshold function to the result to determine the predicted genre for the movie. If the result is above the threshold, the movie will be classified as one genre; if it is below the threshold, it will be classified as another genre..

Perceptron is a simple and fast algorithm that can be effective for movie genre classification, especially when working with linearly separable data. However, it is limited to linear decision boundaries and may not perform as well as some other algorithms on more complex datasets.

g) **K means clustering**: K-means clustering is a machine learning algorithm that can be used for unsupervised classification tasks, where the data is not labeled with class labels. It works by dividing the data into K clusters, where each cluster is represented by the mean (i.e., the centroid) of the data points in the cluster.

In the context of movie genre classification, K-means clustering might be applied to a dataset of movies, with the goal of discovering natural groupings or clusters of movies based on their features. For example, the algorithm might discover that there is a cluster of movies with similar plots, settings, and dialog, and that this cluster corresponds to the action genre.

To classify a new movie, the algorithm would assign the movie to the cluster with the closest centroid, based on the similarity of the movie's features to the centroid. The predicted genre for the movie would then be the genre associated with the cluster to which the movie was assigned.

K-means clustering can be effective for movie genre classification, especially when the data is well-separated into distinct clusters. However, it is sensitive to the initial placement of the centroids and may not perform well if the clusters are overlapping or have complex shapes.

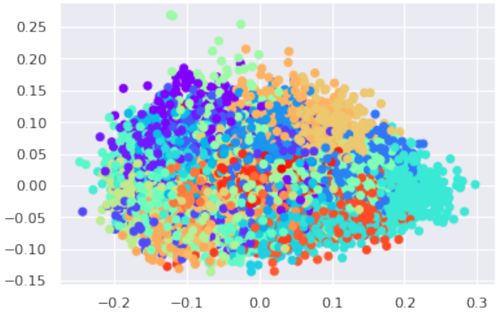

Figure 10- Before Clustering

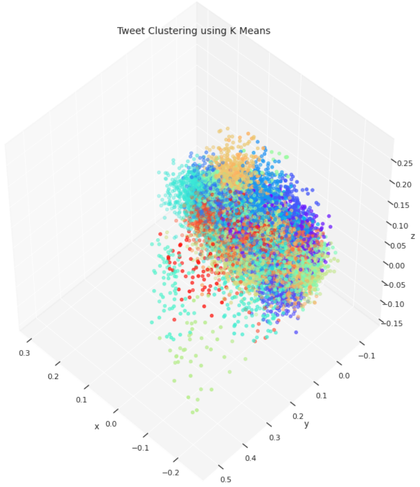

Figure 10- After Clustering

VIII. Result and Discussion

The dataset used for this paper is Movie-genre dataset. There are totally 22,579 samples distributed unevenly across Nine classes. There are 2100 samples in each of the Nine classes. We have implemented the feature extraction, feature reduction and classification modules with denoising using different algorithms to compare the performance. The Multinomial Naive Bayes model gives the accuracy of 89.57%, when it is used with Hyper-parameter then the model gives the accuracy of 91.34% when the alpha value is 0.1, K-Nearest Neighbors model gives the accuracy of 88.45%, Decision Tree model gives the accuracy of 69.23%, SVM model gives the accuracy of 85.00%, Random Forest model gives the accuracy of 39.67%, Perceptron model gives the accuracy of 66.18% .

Table 1 shows the result of machine learning algorithms along with the testing time. Among all the algorithms Naive Bayes with Hyperparameter shows better accuracy of 91.34%

IX. Conclusion

This report presents a multi-level genre-based textual classification of automated sentiment analysis using machine learning algorithms. In our project, we examined accuracy based on a data set of 22,580 scripts. We have successfully tried to analyze the sentiment of the script along with its respective genre which will help users to gain a better understanding and easy interpretation of the script along with the general feedback of other users on the script. The outcomes of different algorithms for the mentioned genres are analyzed and compared with the algorithm to its best output. To compare the efficiency of the algorithm, the classification report is also used with the percentage of accuracy of the testing and training data. Thus, Multinomial Naive Bayes Algorithm ended up being the best model amongst all for textual genre classification model for all these nine above-mentioned genres.

X. References

*A. **Data availability :***

The data used in this study is available as open-source at https://www.kaggle.com/datasets/whenamancodes/popular-movies-datasets-58000-movies?select=movies.csv

*B. **References :***

- Natural Language Toolkit: https://www.nltk.org/
- SVM: https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm
- Perceptron: https://www.javatpoint.com/perceptron-in-machine-learning
- https://github.com/Wonuabimbola/movie-genre-prediction/tree/main/data
- https://www.researchgate.net/publication/338468885_Movie_Genre_Classification_using_Machine_Learning_and_Natural_Language_Processing
- https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114

Table 1
Results of machine learning algorithms along with Accuracy.

Methods	Accuracy
Multinomial Naive Bayes	89.57%
Naive Bayes with Hyperparameter (alpha = 0.1)	91.34%
KNN	88.45%
Decision Tree	69.23%
SVM	85.00%
Random Forest	39.67%
Perceptron	66.18%