MULTI-LABEL MOVIE GENRE CLASSIFICATION

Contents

1 INTRODUCTION

Classifying a text document into one or the other pre-defined classes (e.g., spam or not a spam) is an example of a standard Binary Classification problem. There are other scenarios where in one would need to classify the document into more than 2 classes. This is a multi-class classification problem where a data instance is associated with only one of the many single class labels, e.g., News Topic Classifier with the possible classes as 'Food', 'Sports', 'Politics' wherein each article can fall into only one of these classes.

In this project, we will investigate a scenario where each document can possibly be assigned to more than one class, i.e., multi-label classification. The dataset comes from IMDB contains several information about the movie. Here, we will use the plot of the movie and find out what are all the genres it falls under.

2 DATA OVERVIEW

The data set contains 30 columns – out of which 27 of them correspond to the possible 27 movie genres. These are basically our target columns. If the movie falls under a genre, then that column will indicate 1, else it will indicate 0. Each observation consists of the following data

First 3 columns are the features of the movie

- title: Title of the movie
- plot: The plot of the movie
- plot_lang: The language of the movie

The rest of the below 27 columns correspond to the target variable, i.e., the genre the movies are classified into are given in the table below

Action	Family	Reality-TV
Adult	Fantasy	Romance
Adventure	Game-Show	Sci-Fi
Animation	History	Short
Biography	Horror	Sport
Comedy	Music	Talk-Show
Crime	Musical	Thriller
Documentary	Mystery	War
Drama	News	Western

The shape of the data = (117194, 30). First couple of rows are shown below

	title	plot	Action	Adult	Adventure	Animation	...	Sport	Talk-Show	Thriller	War	Western	plot_lang
0	"#7DaysLater" (2013)	dayslater interactive comedy series feature en...	0	0	0	0	...	0	0	0	0	0	en
1	"#BlackLove" (2015) {Crash the Party (#1.9)}	week leave workshops women consider idea ladie...	0	0	0	0	...	0	0	0	0	0	en

2 rows × 30 columns

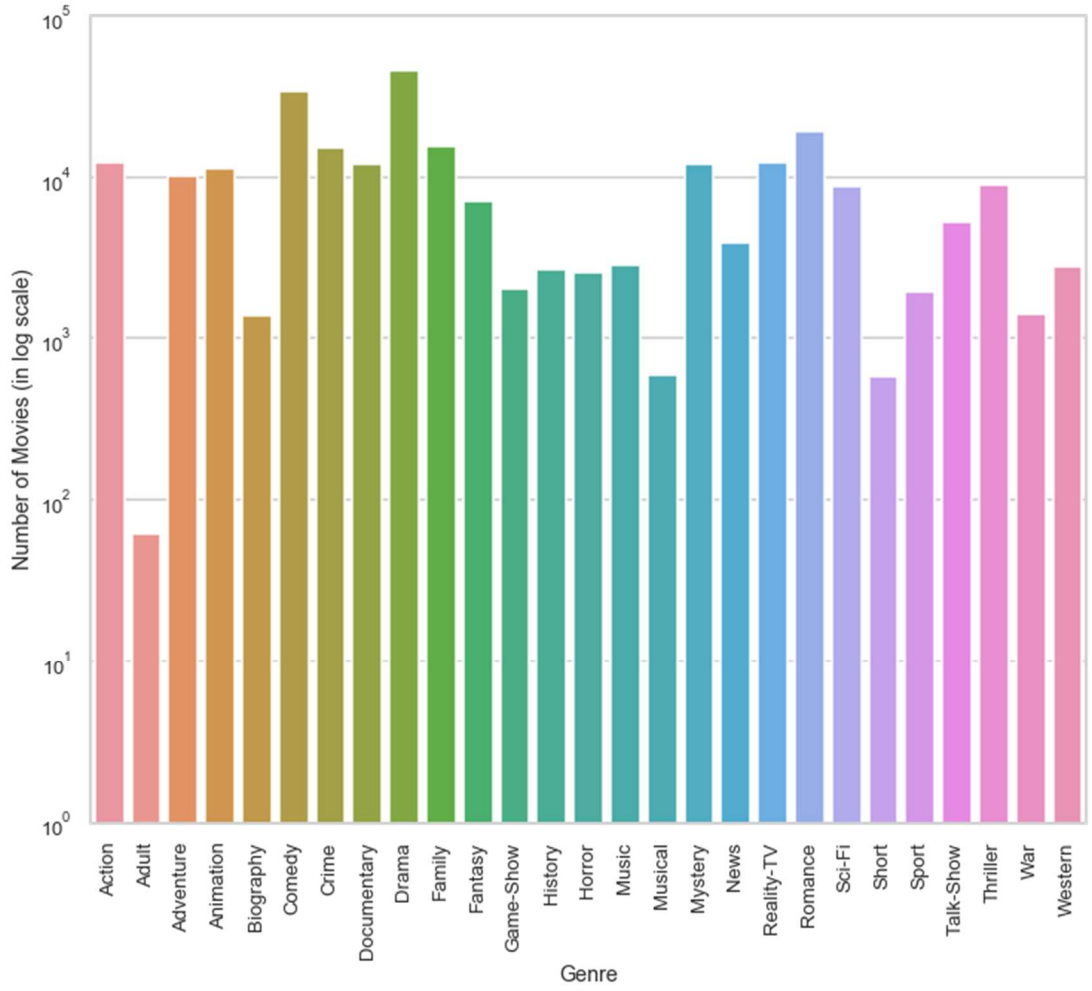
The goal of this project is to predict all the possible genres of the movies based on its plot.

The provided data consists of over 117k observations of movies along with 30 column variables. Let us investigate what each column looks like.

3 EXPLORATORY DATA ANALYSIS

3.1 Number of Movies per Genre

Below are plot that shows the distribution of the number of movies for each genre available in our data set. Note that the y-axis is in log domain.

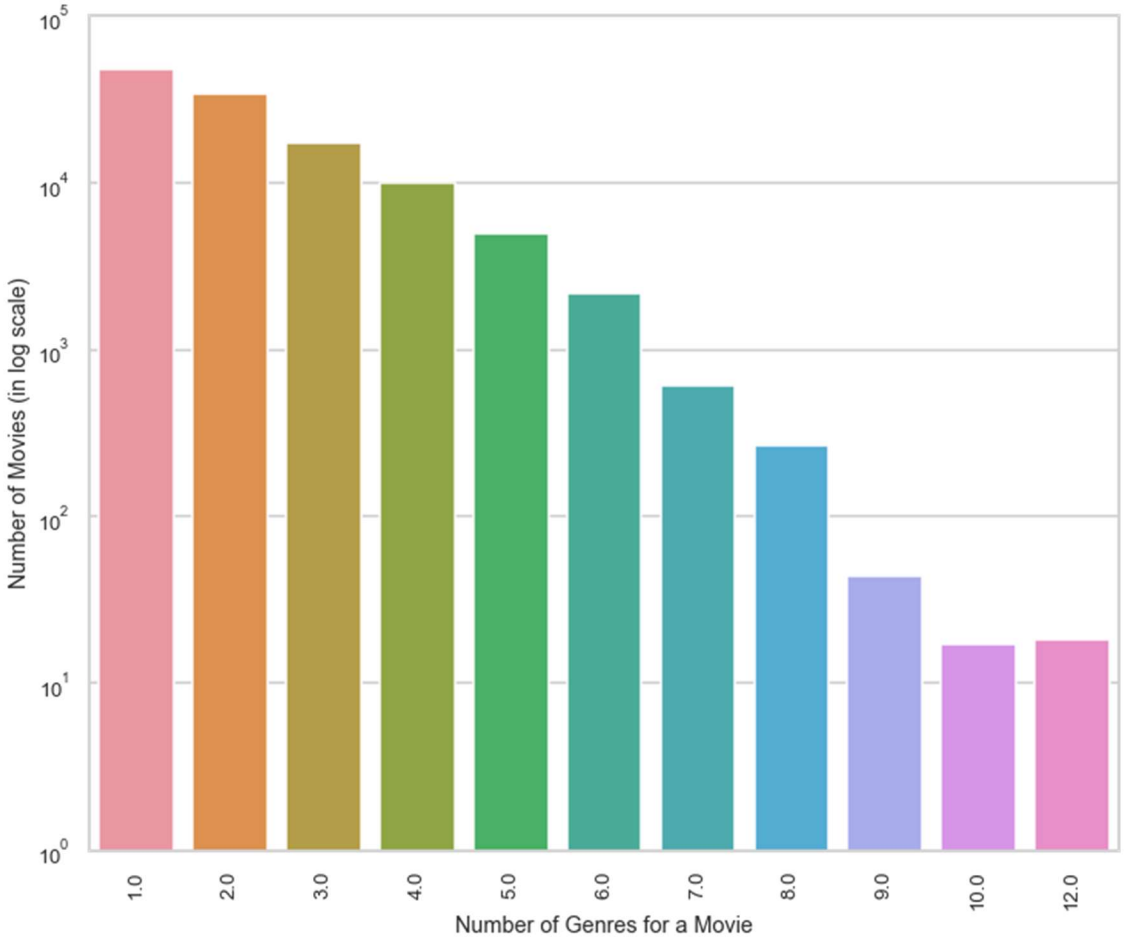


We observe the following

- Lowest Genre movies is Adult (only 61 movies)
- Highest Genre movies is Drama (45891 movies), followed by Comedy (33870)

3.2 Number of Genres per Movie

In the below plot, we show the distribution of the number of genres each movie is classified into in the data set. There are movies which fall under 12 genres!

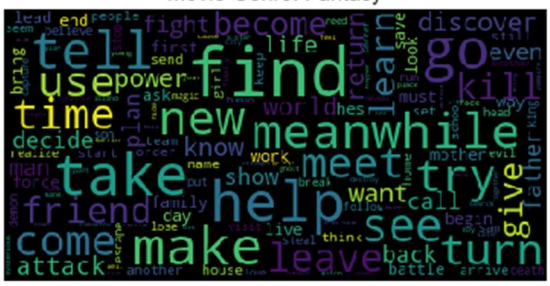


3.3 Word Cloud Plots

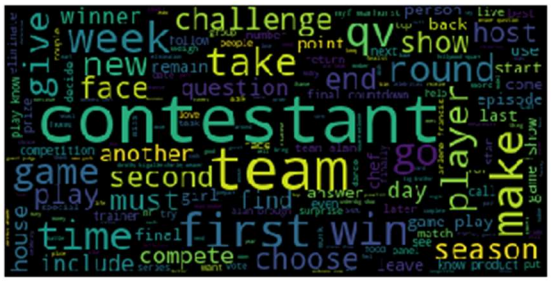
A Wordcloud (or Tag cloud) is a visual representation of text data. It displays a list of words, the importance of each being shown with font size or color. This format is useful for quickly perceiving the most prominent terms.

Below are wordcloud plots for each of the 27 genres. We can obtain very useful insights on the important features (words) for each genre.

Movie Genre: Fantasy

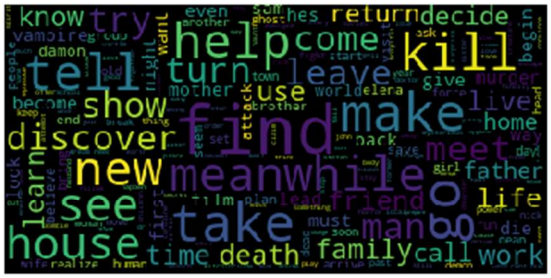


Movie Genre: Game-Show



Movie Genre: History

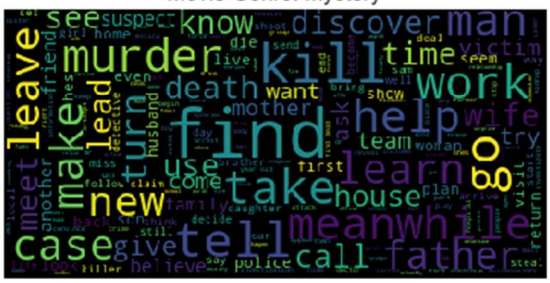
Movie Genre: Horror



Movie Genre: Music

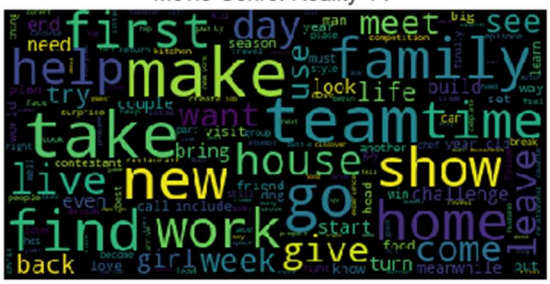
Movie Genre: Musical

Movie Genre: Mystery



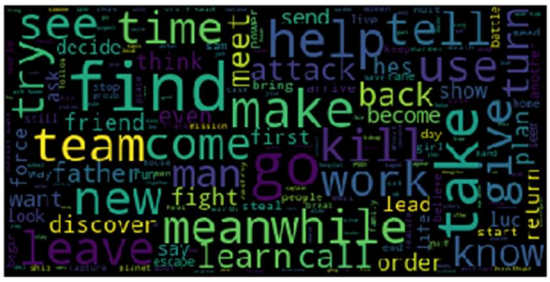
Movie Genre: News

Movie Genre: Reality-TV



Movie Genre: Romance

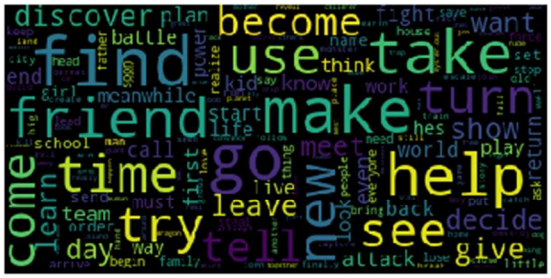
Movie Genre: Action



Movie Genre: Adult

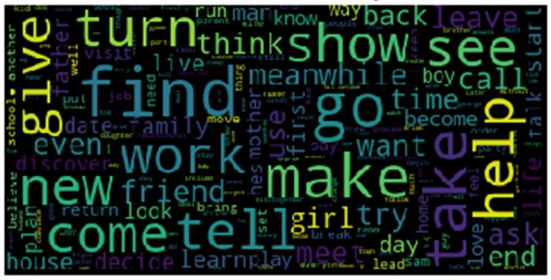
Movie Genre: Adventure

Movie Genre: Animation

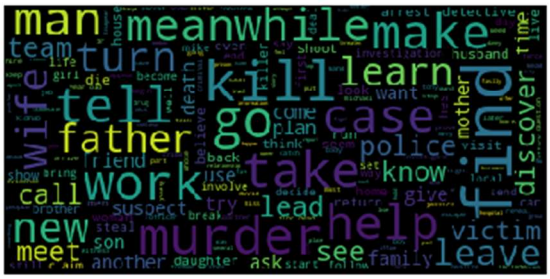


Movie Genre: Biography

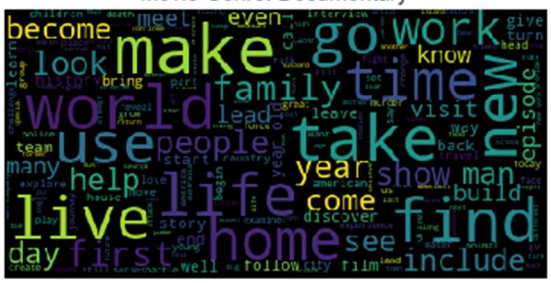
Movie Genre: Comedy



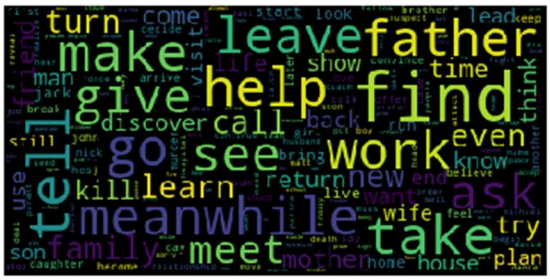
Movie Genre: Crime



Movie Genre: Documentary

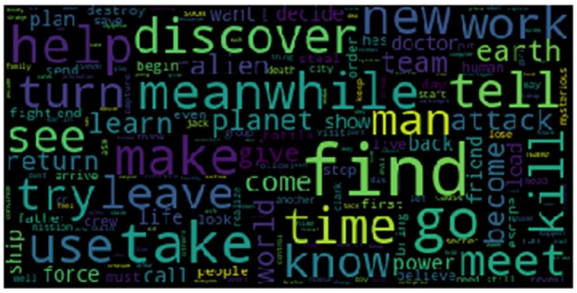


Movie Genre: Drama

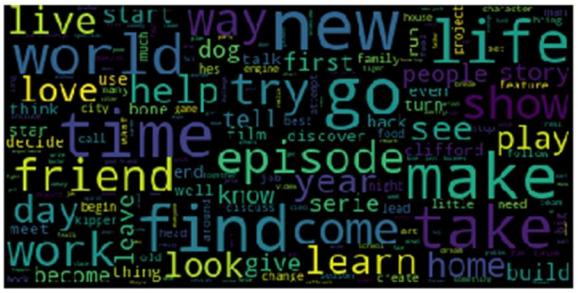


Movie Genre: Family

Movie Genre: Sci-Fi



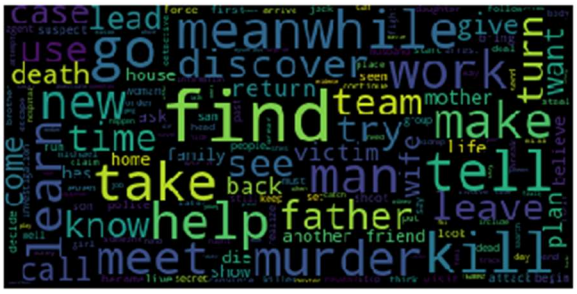
Movie Genre: Short



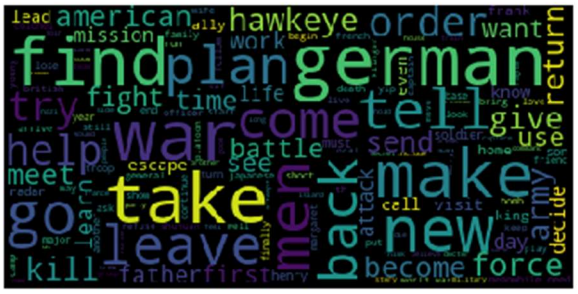
Movie Genre: Sport

Movie Genre: Talk-Show

Movie Genre: Thriller



Movie Genre: War



Movie Genre: Western

Few interesting observations from the above word cloud plots

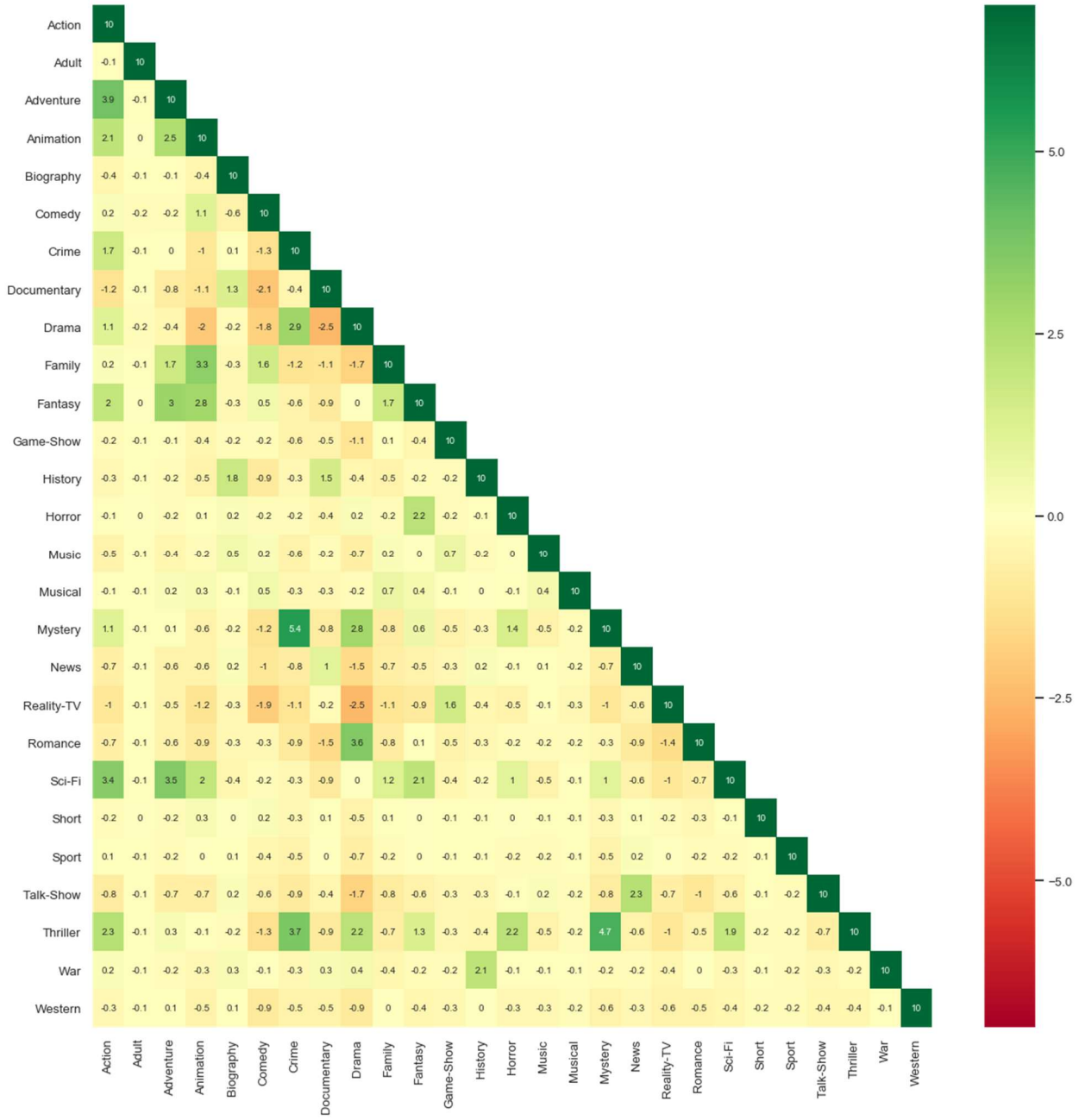
- qv is generally used as a tag to indicate a person's name when he appears in a movie as himself. Hence, we see qv show up as an important feature in News, Game-Show, Biography and Talk-Show Genres.
- german is an important feature for War based Genre

- Few obvious key words include: attack for Action, discover for Adventure, life, career for Biography, kill, murder for Crime, Mystery and Thriller, challenge, contestant, round for Game-Shows, perform, band for Music, vs, win, team for Sports, horse, sheriff for Western.

4 CORRELATION ANALYSIS

4.1 Heatmap

Below is a heatmap plot of the correlation between all the genres available



Following are few observations. The below Genre categories show strong positive correlation with each other

- Action, Adventure and Sci-Fi
- Animation, Fantasy and Family
- Crime, Thriller, Mystery and Drama
- Biography, Documentary and History
- Drama and Romance
- Game-show and Reality-TV
- Horror, Thriller and Fantasy
- Talk-show and News
- War and History

The below Genre categories show strong negative correlation with each other

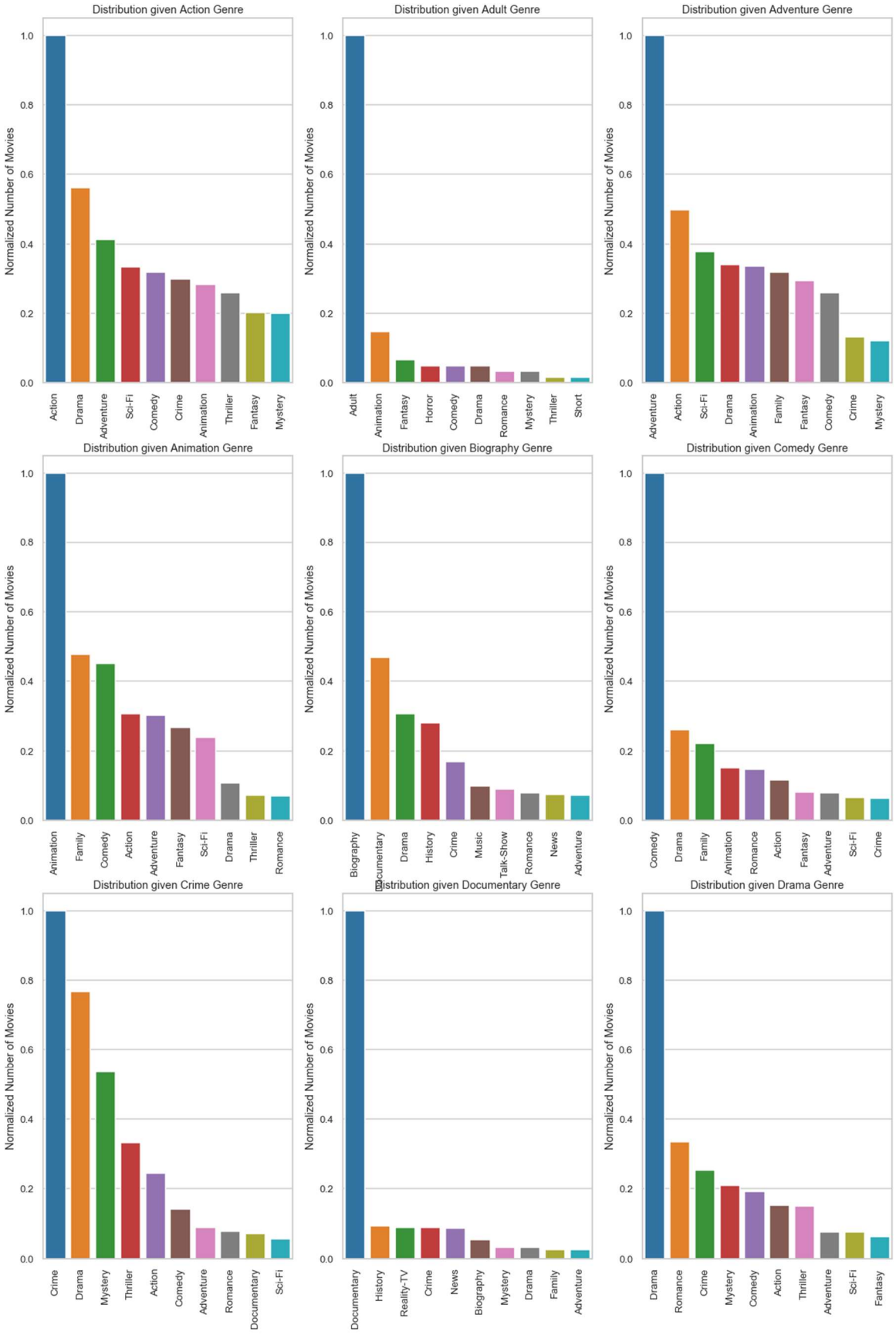
- Animation and Drama
- Comedy with Documentary and Reality-TV
- Documentary with Comedy, Drama and Romance
- Drama with Animation, Reality-TV and Comedy

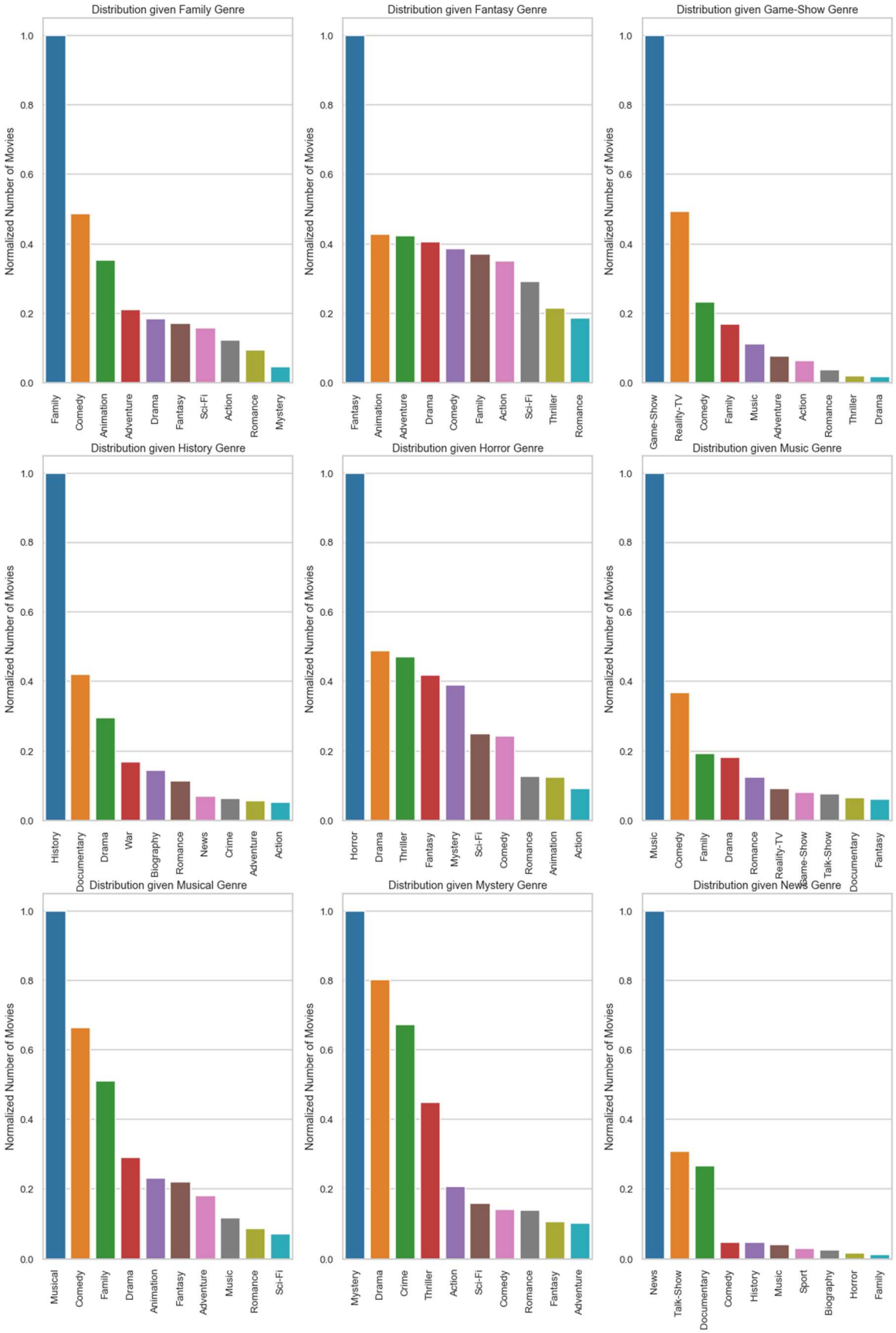
4.2 Multi-Genre Distribution Plots

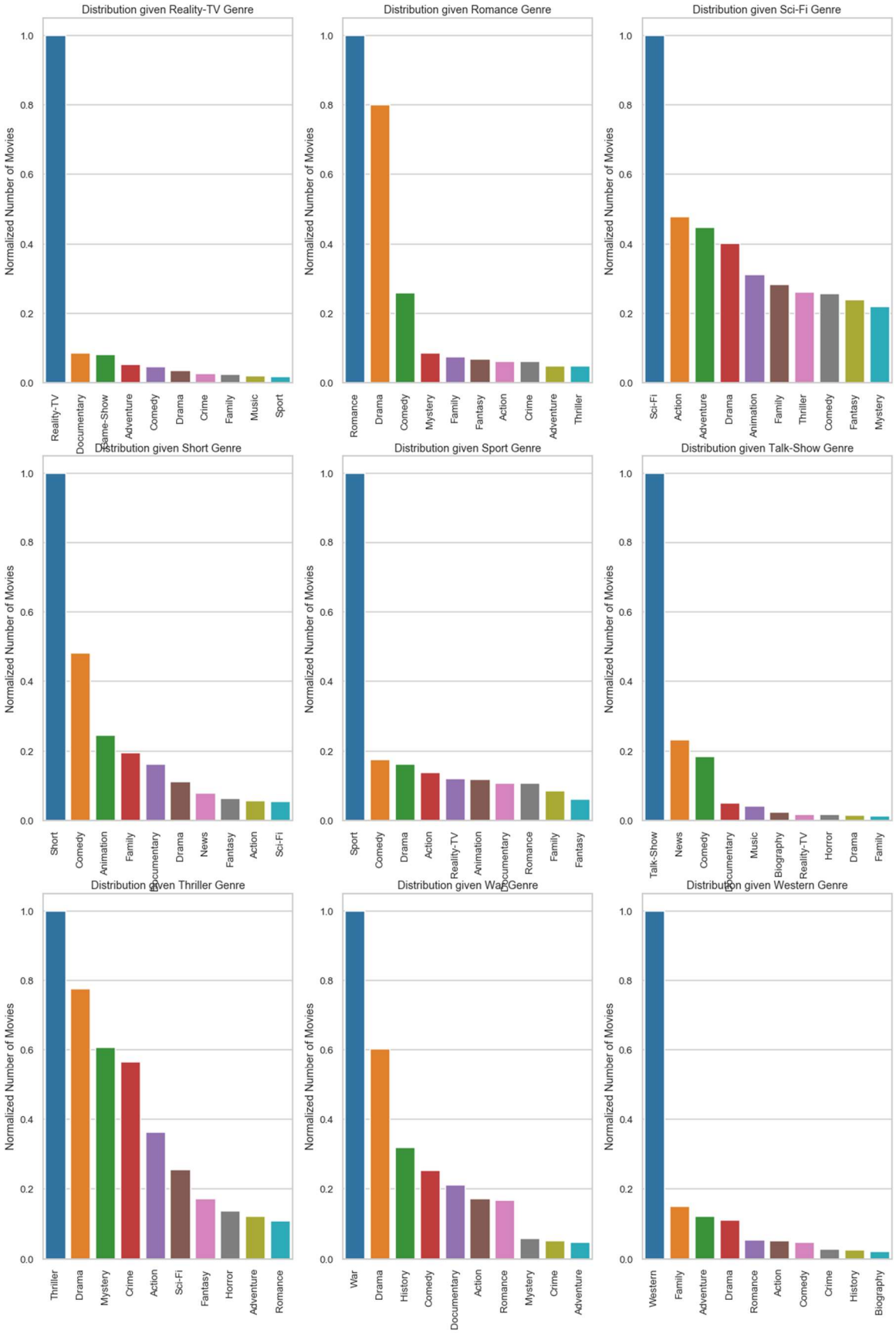
Now let us see if a movie belongs to a certain Genre, what are the other Genres it might fall under.

From the below plots, we can see the following correlations

- More than half of Action movies also fall in Drama genre
- Almost 50% of the Animation movies are also categorized as Family or Comedy
- 80% of the Crime, Mystery, Thriller and Romance movies are also categorized as Drama
- Half of the Game-Shows are also categorized as Reality TV
- 65% of Musical movies are Comedy
- 50% of Short movies are Comedy
- 60% of War movies are also Drama


Distribution given Action Genre
Normalized Number of Movies
Action Drama Adventure Sci-Fi Comedy Crime Animation Thriller Fantasy Mystery
Distribution given Adult Genre
Normalized Number of Movies
Adult Animation Fantasy Horror Comedy Drama Romance Mystery Thriller Short
Distribution given Adventure Genre
Normalized Number of Movies
Adventure Action Sci-Fi Drama Animation Family Fantasy Comedy Crime Mystery
Distribution given Animation Genre
Normalized Number of Movies
Animation Family Comedy Action Adventure Fantasy Sci-Fi Drama Thriller Romance
Distribution given Biography Genre
Normalized Number of Movies
Biography Documentary Drama History Crime Music Talk-Show Romance News Adventure
Distribution given Comedy Genre
Normalized Number of Movies
Comedy Drama Family Animation Romance Action Fantasy Adventure Sci-Fi Crime
Distribution given Crime Genre
Normalized Number of Movies
Crime Drama Mystery Thriller Action Comedy Adventure Romance Documentary Sci-Fi
Distribution given Documentary Genre
Normalized Number of Movies
Documentary History Reality-TV Crime News Biography Mystery Drama Family Adventure
Distribution given Drama Genre
Normalized Number of Movies
Drama Romance Crime Mystery Comedy Action Thriller Adventure Sci-Fi Fantasy


Distribution given Family Genre
Normalized Number of Movies
Family Comedy Animation Adventure Drama Fantasy Sci-Fi Action Romance Mystery
Distribution given Fantasy Genre
Normalized Number of Movies
Fantasy Animation Adventure Drama Comedy Family Action Sci-Fi Thriller Romance
Distribution given Game-Show Genre
Normalized Number of Movies
Game-Show Reality-TV Comedy Family Music Adventure Action Romance Thriller Drama
Distribution given History Genre
Normalized Number of Movies
History Documentary Drama War Biography Romance News Crime Adventure Action
Distribution given Horror Genre
Normalized Number of Movies
Horror Drama Thriller Fantasy Mystery Sci-Fi Comedy Romance Animation Action
Distribution given Music Genre
Normalized Number of Movies
Music Comedy Family Drama Romance Reality-TV Game-Show Talk-Show Documentary Fantasy
Distribution given Musical Genre
Normalized Number of Movies
Musical Comedy Family Drama Animation Fantasy Adventure Music Romance Sci-Fi
Distribution given Mystery Genre
Normalized Number of Movies
Mystery Drama Crime Thriller Action Sci-Fi Comedy Romance Fantasy Adventure
Distribution given News Genre
Normalized Number of Movies
News Talk-Show Documentary Comedy History Music Sport Biography Horror Family

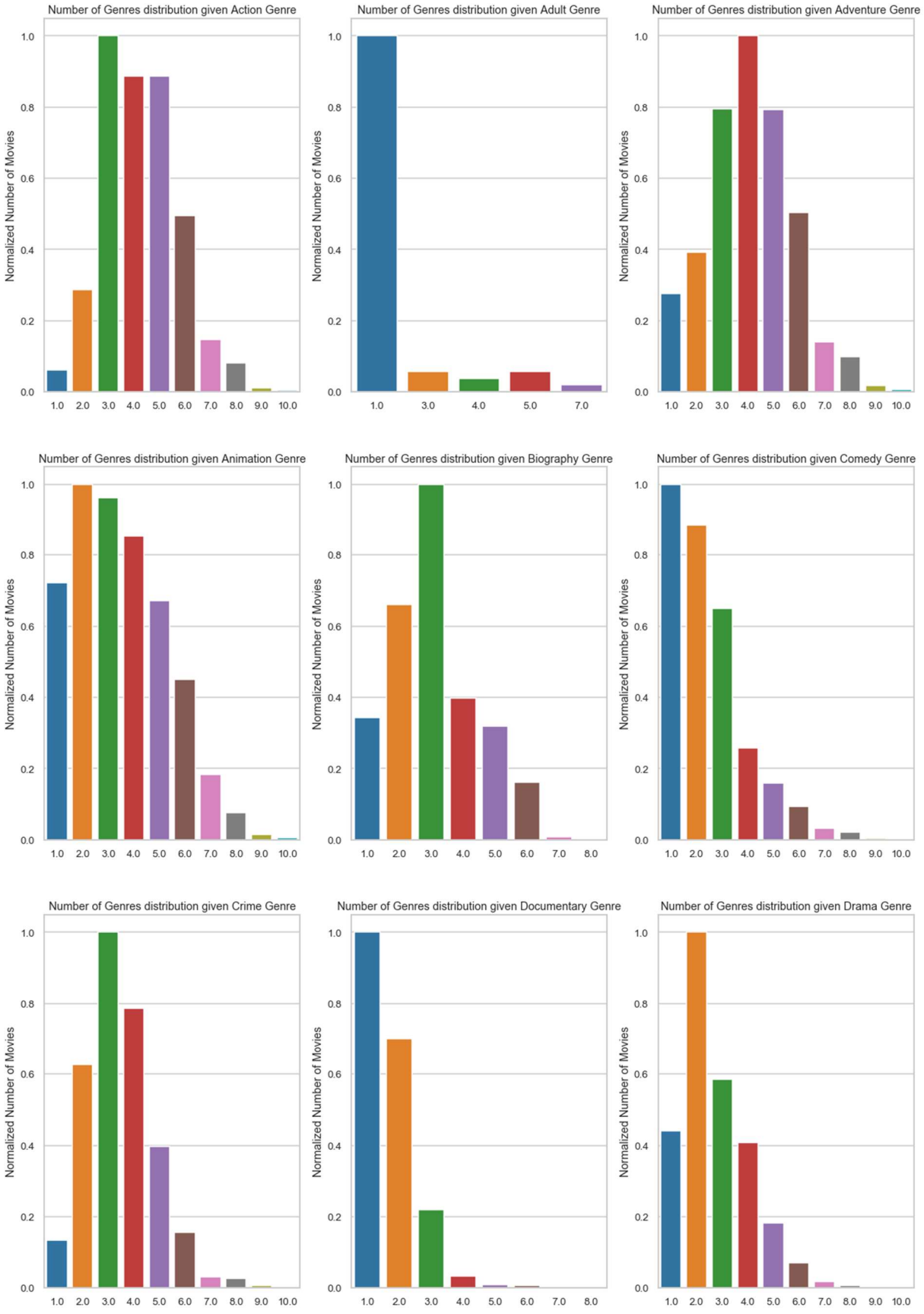

Distribution given Reality-TV Genre
Normalized Number of Movies
Reality-TV
Documentary
Game-Show
Adventure
Comedy
Drama
Crime
Family
Music
Sport
Distribution given Romance Genre
Normalized Number of Movies
Romance
Drama
Comedy
Mystery
Family
Fantasy
Action
Crime
Adventure
Thriller
Distribution given Sci-Fi Genre
Normalized Number of Movies
Sci-Fi
Action
Adventure
Drama
Animation
Family
Thriller
Comedy
Fantasy
Mystery
Distribution given Short Genre
Normalized Number of Movies
Short
Comedy
Animation
Family
Documentary
Drama
News
Fantasy
Action
Sci-Fi
Distribution given Sport Genre
Normalized Number of Movies
Sport
Comedy
Drama
Action
Reality-TV
Animation
Documentary
Romance
Family
Fantasy
Distribution given Talk-Show Genre
Normalized Number of Movies
Talk-Show
News
Comedy
Documentary
Music
Biography
Reality-TV
Horror
Drama
Family
Distribution given Thriller Genre
Normalized Number of Movies
Thriller
Drama
Mystery
Crime
Action
Sci-Fi
Fantasy
Horror
Adventure
Romance
Distribution given War Genre
Normalized Number of Movies
War
Drama
History
Comedy
Documentary
Action
Romance
Mystery
Crime
Adventure
Distribution given Western Genre
Normalized Number of Movies
Western
Family
Adventure
Drama
Romance
Action
Comedy
Crime
History
Biography

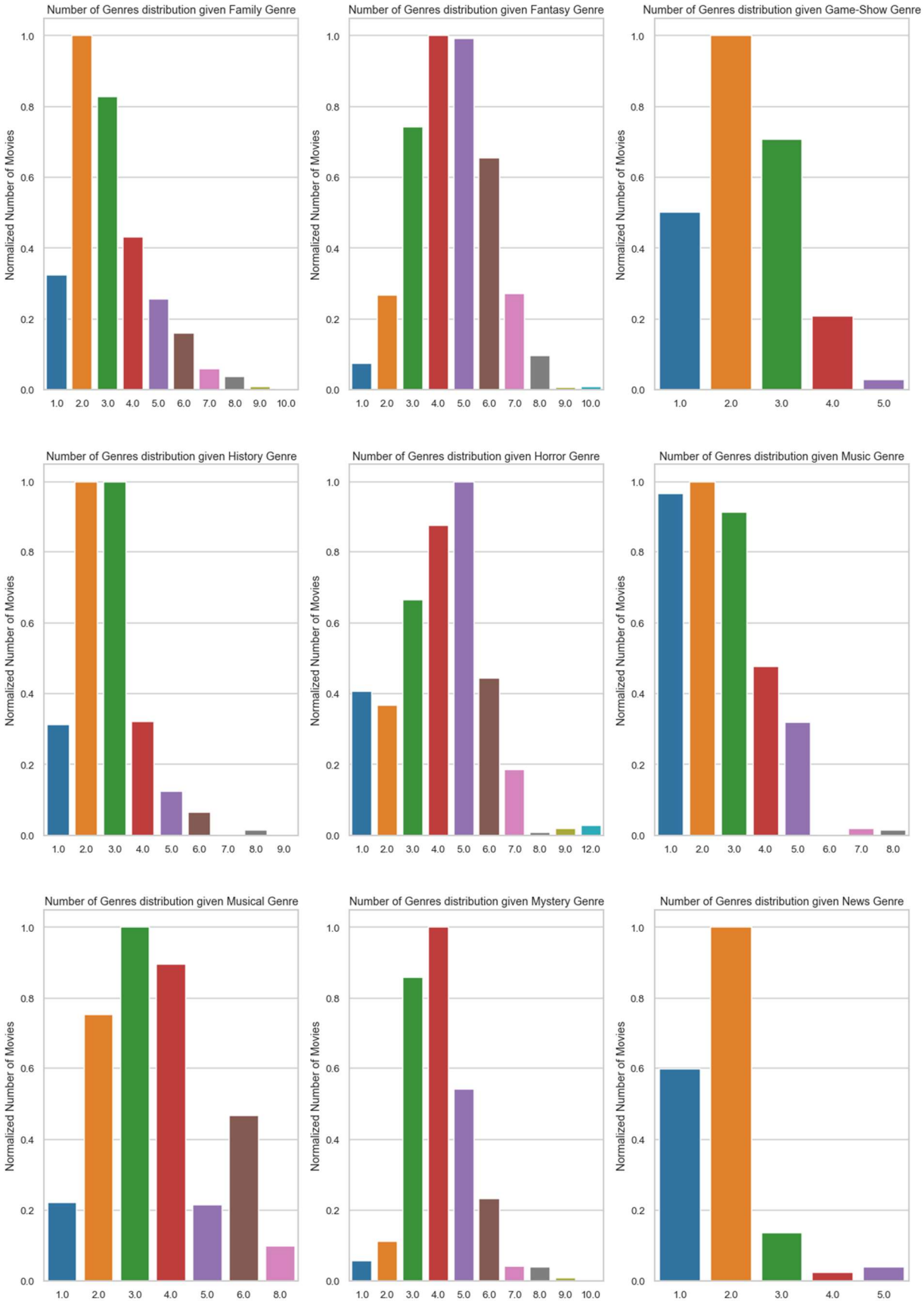
4.3 Number of Genres given a Genre

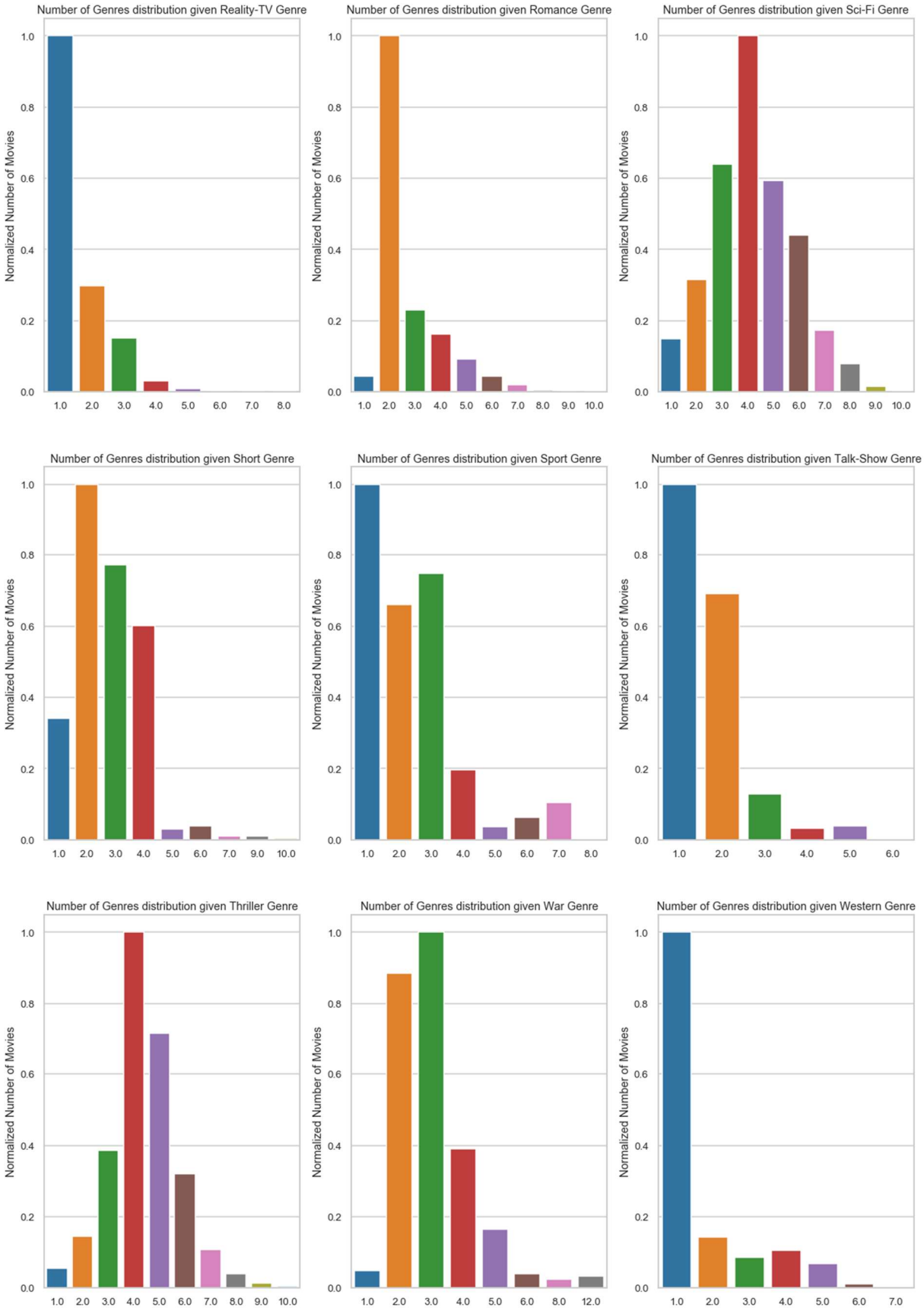
Let us see how many genres each movie is classified into for each of the 27 genres.

We can the below observations from the below plots

- Most of the Action, Adventure, Animation, Fantasy, Horror, Mystery, Sci-Fi and Thriller movies have 3 to 6 categories
- Most of the Adult, Documentary, Reality-TV, Talk-Show and Western movies have just a single label


Number of Genres distribution given Action Genre
Number of Genres distribution given Adult Genre
Number of Genres distribution given Adventure Genre
Number of Genres distribution given Animation Genre
Number of Genres distribution given Biography Genre
Number of Genres distribution given Comedy Genre
Number of Genres distribution given Crime Genre
Number of Genres distribution given Documentary Genre
Number of Genres distribution given Drama Genre
Normalized Number of Movies


Number of Genres distribution given Family Genre
Number of Genres distribution given Fantasy Genre
Number of Genres distribution given Game-Show Genre
Number of Genres distribution given History Genre
Number of Genres distribution given Horror Genre
Number of Genres distribution given Music Genre
Number of Genres distribution given Musical Genre
Number of Genres distribution given Mystery Genre
Number of Genres distribution given News Genre
Normalized Number of Movies


Number of Genres distribution given Reality-TV Genre
Number of Genres distribution given Romance Genre
Number of Genres distribution given Sci-Fi Genre
Number of Genres distribution given Short Genre
Number of Genres distribution given Sport Genre
Number of Genres distribution given Talk-Show Genre
Number of Genres distribution given Thriller Genre
Number of Genres distribution given War Genre
Number of Genres distribution given Western Genre
Normalized Number of Movies

4.4 Sentence Embedding Similarities

Universal Sentence Embeddings (USE) converts a sentence to a vectorized numeric representation while capturing the semantic meaning of the sentence and identifying order. They're considered "universal" because they promise to encode general properties of sentences due to being trained on very large datasets. In theory, they can then be applied to any downstream NLP task without requiring domain specific knowledge. For example, the vector representation of the two sentences – a) 'Will it snow tomorrow' and b) 'Global warming is real' will share a lot of similarity

Here, we plot the similarities of the vector representation of movie plots of various genres to see this behavior

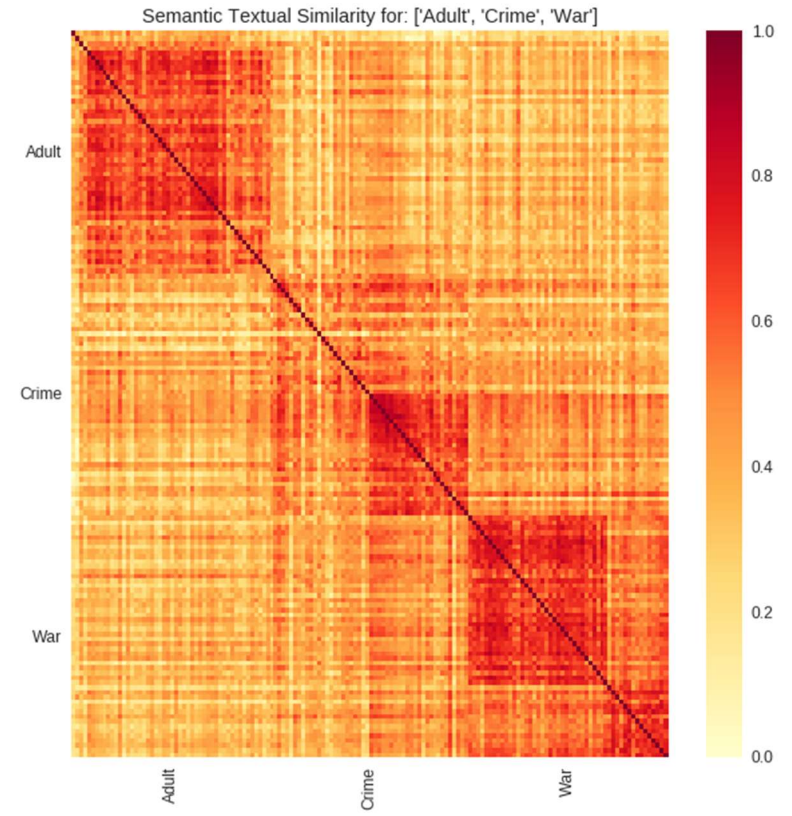
For Heatmap visualization

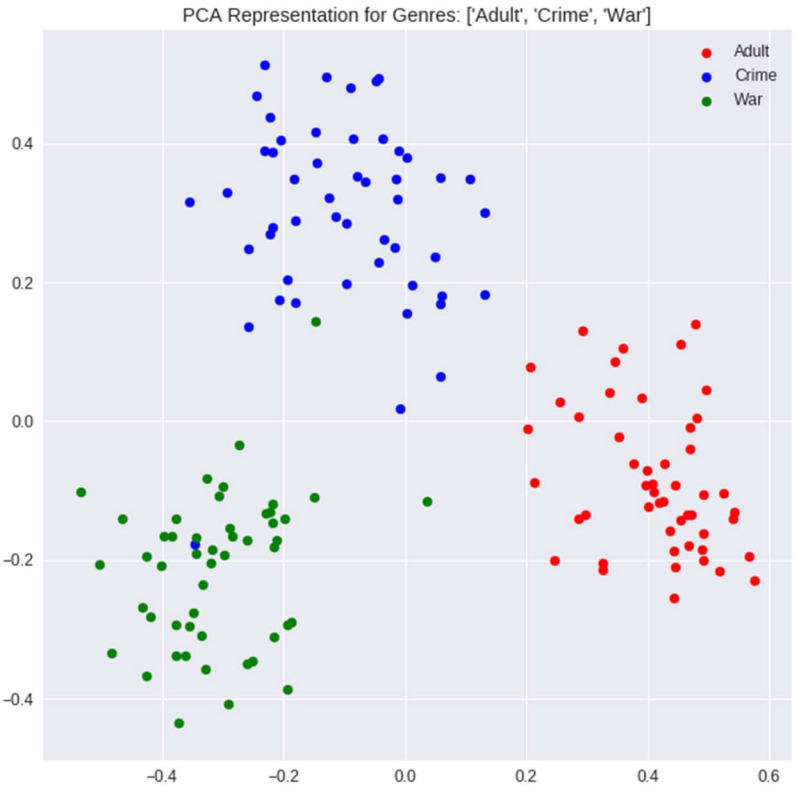
- We pick the first 50 plots from 3 genres (150 plots in all)
- Using the embedded vectors for these we plot the heatmap
- Plots from similar genre are expected to show higher correlation
    - Since we grouped the movie plots based on genres (the first 50 belonging to the first genre, the next 50 belonging to the second genre and the last 50 belonging to the third genre), we expect to see strong correlation (3 chunks of size 50x50) closer to the diagonal

For PCA 2D representation visualization

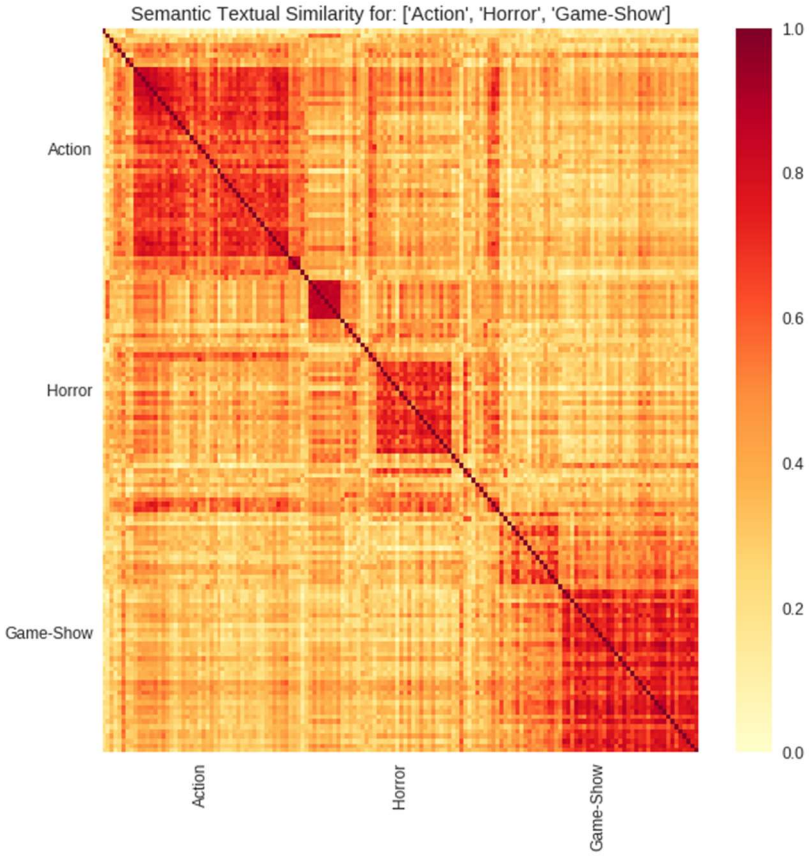
- We pick the first 50 plots from 3 genres (150 plots in all)
- We obtain the 2 largest variance components using PCA for these 150 vectors and project them along these components
- Plots from similar genre are expected to be close by and form a cluster

4.4.1 [Adult, Crime, War] Genres

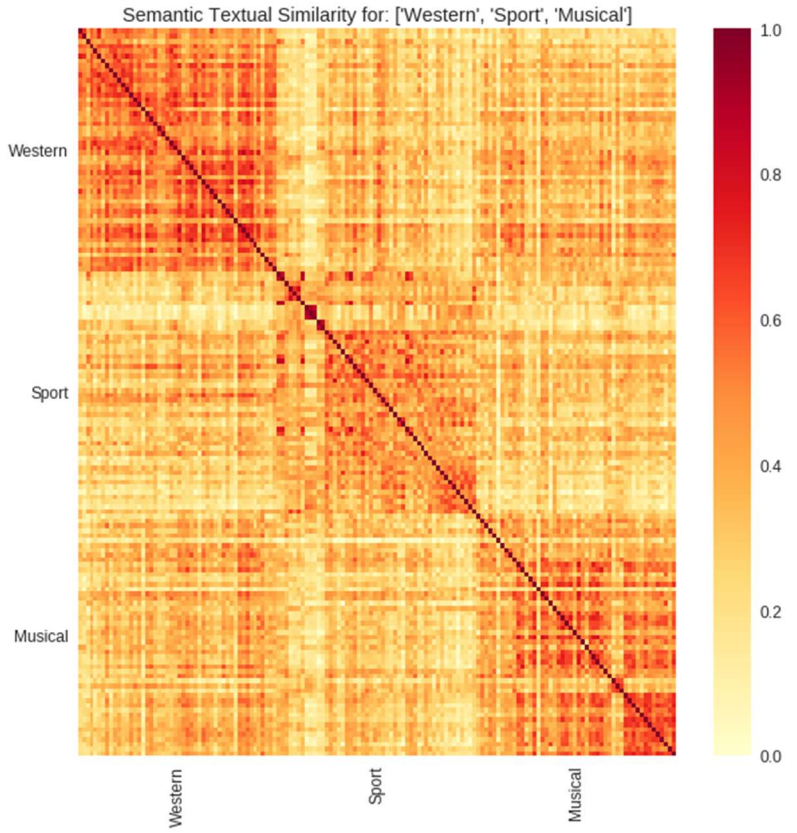


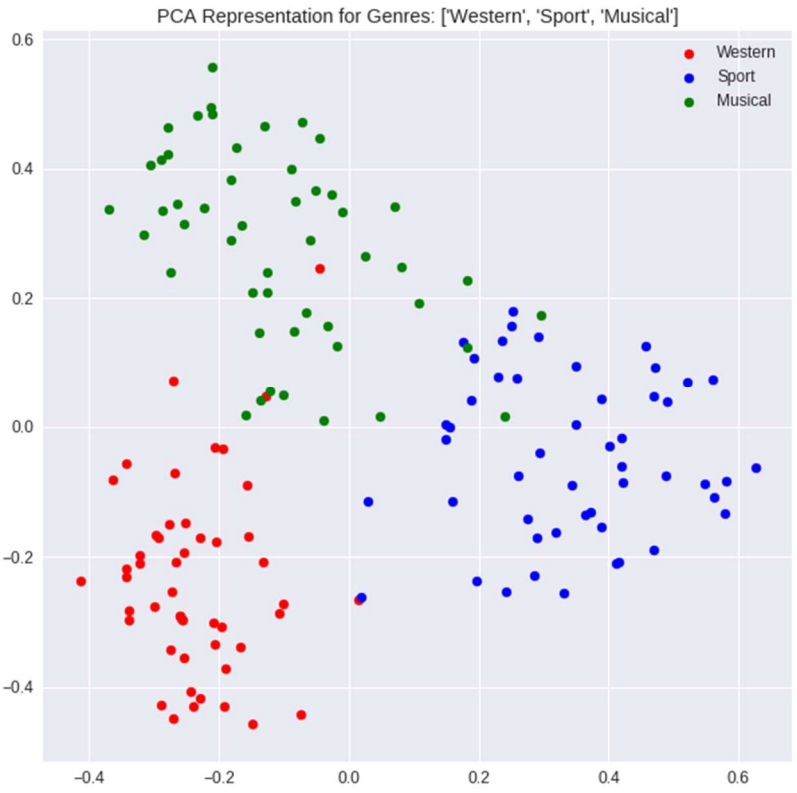


4.4.2 [Action, Horror, Game-Show] Genres



4.4.3 [Western, Sport, Musical] Genres

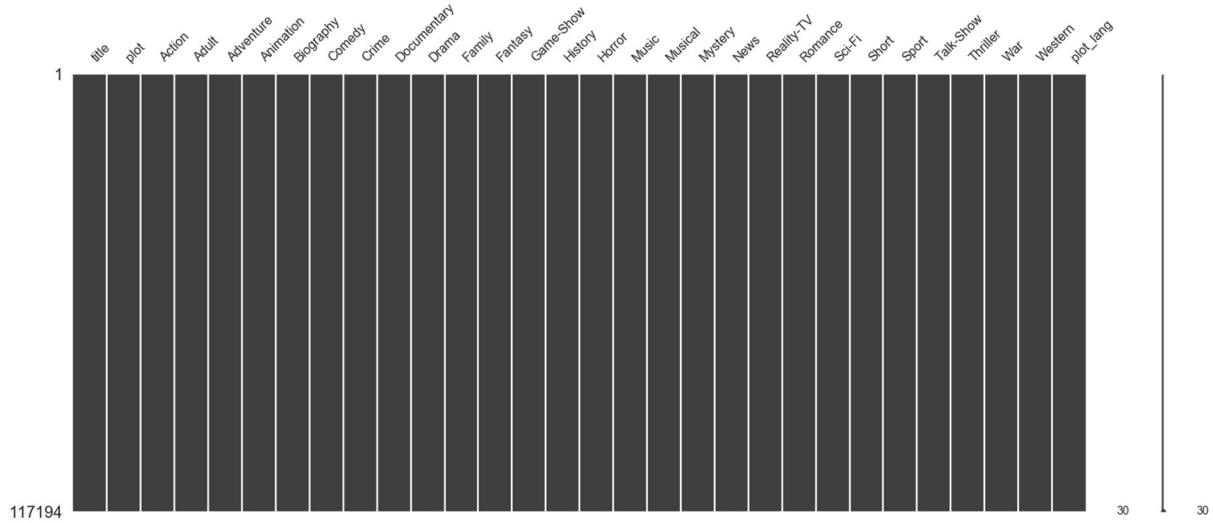




5 DATA PRE-PROCESSING

5.1 Missing Data Fields

As seen from the below figure, the provided dataset has no missing values for any features



5.2 Feature Engineering

We perform the following feature engineering before using the data

- Drop title: This project focusses on predicting the genre from the movie plot. Hence, we disregard this column. We could possibly use this column too, to enhance the prediction
- Drop plot_lang: We noticed that all the movies provided have plots in English language. Hence, drop this column

5.3 Text pre-processing

We use the following text preprocessing steps on the movie plot

- Removing any HTML tags
- Removing punctuations
- Removing any accented characters
- Keeping only alphabetic strings

- Removing 'english' stop words
- Lemmatizing
- Lower casing all the words

5.3.1 Removing HTML Tags

HTML tags are removed using the below regular expression

```
html_tag = '<.*?>'
```

This matches any expression within angular brackets.

5.3.2 Removing Punctuations

We remove the following punctuations using the below regular expression

```
re.sub(r'[?|!|\'|"|#]', '', sentence)
```

Other punctuations such as braces, comma, full stop is replaced with a space, since they separate out multiple words

```
re.sub(r'[,|.|;|:|(|)|{|}|\|/|<|>]|-', ' ', sentence)
```

5.3.3 Removing Accented Characters

All accented characters are converted and standardized into ASCII characters. E.g., converting **é** to **e**

5.3.4 Keeping Alphabetic Strings

All numerals, and non-alphabetic characters are removed using the below regular expression

```
re.sub('[^a-z A-Z]+', ' ', sentence)
```

5.3.5 Removing Stop Words

"Stop words" are the most common words in a language like "the", "a", "on", "is", "all". These words do not carry important meaning and are usually removed from texts. We use the Natural Language Toolkit (NLTK), a suite of libraries and programs for symbolic and statistical natural language processing, to remove stop words.

5.3.6 Lemmatize

Stemming and Lemmatizing are Key Normalization techniques which maps words originating from the same word to a single feature word. Stemming is a process of reducing words to their word stem, base or root form (for example, books—book, looked—look). Lemmatization is very similar to stemming, where we remove word affixes to get to the base form of a word. However, the base form in this case is known as

the root word, but not the root stem. The difference being that the root word is always a lexicographically correct word (present in the dictionary), but the root stem may not be so.

Since Stemming and Lemmatizing serve the same purpose, we use just one of them – Lemmatizer (from nltk) in this project

5.3.7 Lower Casing the words

Since features are case-sensitive, we convert every word into lower case

6 MODELLING

6.1 Modeling Overview

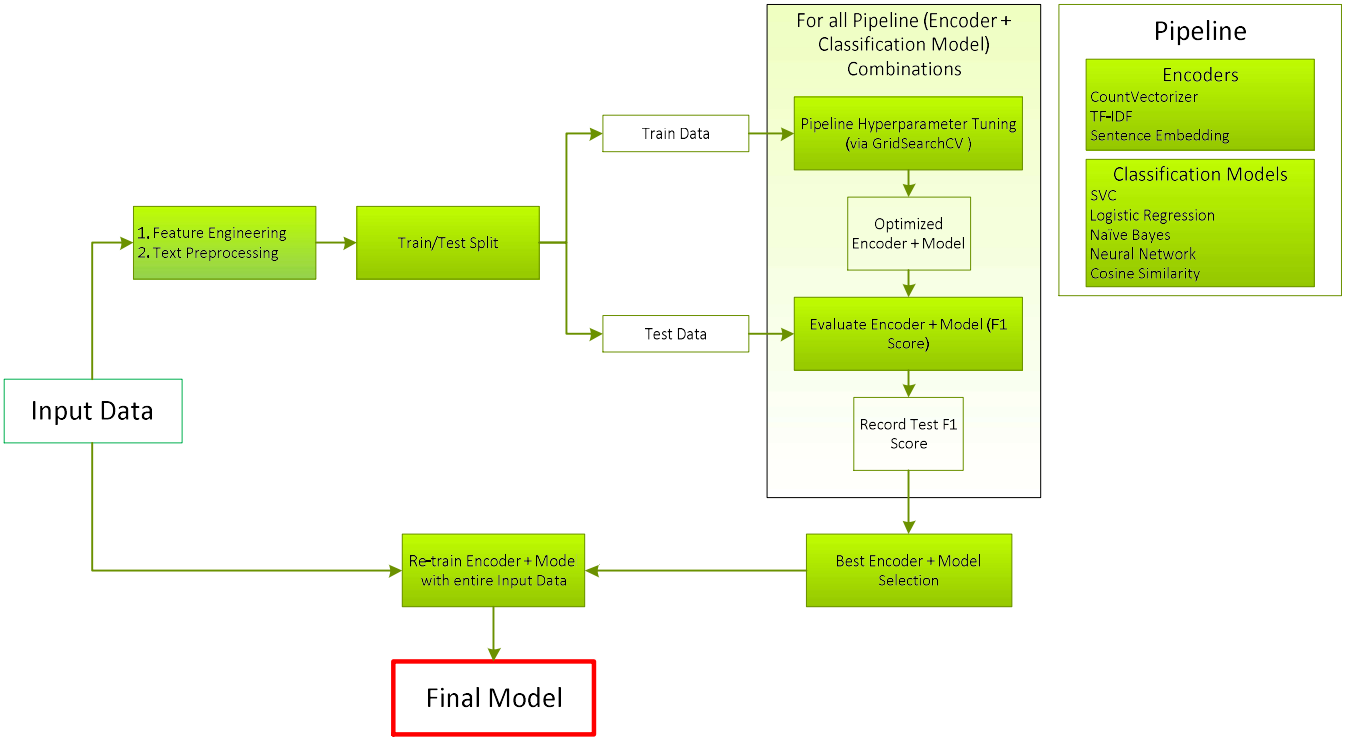


Figure 6-1: Overview of the Modelling procedure

Figure 6-1 depicts the overall procedure followed to obtain the Final Model. The provided input data is first transformed using Feature Engineering and then preprocessed. We then split the data into Train set (for Hyperparameter tuning) and Test set (for Model Evaluation). We use a Pipeline consisting of Encoder and a Classifier Model upon which the hyperparameter tuning is performed. Using F1 score as our evaluation metric, we compare various models and select the Encoder + Classification Model based on the highest F1 on the Test data. The final model (which might be used for a Web-Application) is then obtained by again training the selected Encoder+Classification Model Algorithm on the entire Input Data set.

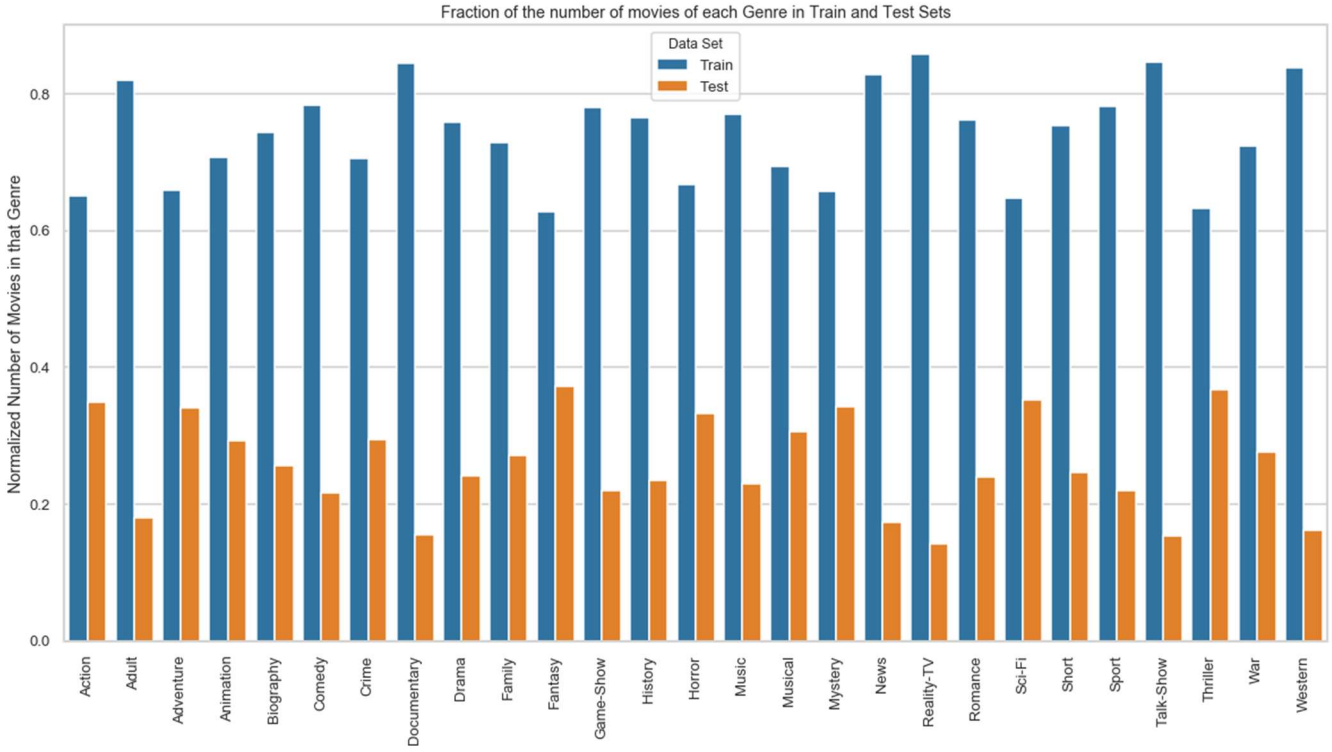
6.2 Train/Test Split

Now let us first split the provided data into a train and test sets. This is a multi-label data set with highly imbalanced labels (Adult having just 61 samples and Drama with >45000 samples). We split the provided data set ensuring that both the data sets have a minimum fraction (~0.2) of every label. The method adopted to achieve this is as follows

We loop through each category and include 0.2 fraction of that category into the test data set. Clearly at the end of the loop, the number of occurrences of each category will be greater than 0.2 fraction since most movies that are being included into the test set as a genre also are categorized with other genres.

We can further optimize the above method to ensure more even-distribution of movies into train and test data set; however, that is not really the aim of this project. Hence, we will just stick to this simplified version.

The below figure plots the distribution of each genre in our Train and Test data set. For every category, we have at least 60% of the samples in the training data set and at least 15% of the samples in the test data set which seems good for our purpose



6.3 Evaluation Metric – F1 Score

The data consists of highly imbalanced data where each movie is classified into an average of 5-6 genres out of a maximum of 27. Also, the number of movies classified as a genre range from 61 to 45000.

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observations. We can consider look at Accuracy two ways in our example

- We record a correct prediction only if all the genres match.
- We record our 27 genre predictions for each movie and calculate our accuracy as (Number of correct genre predictions for each movie)/(27*number of observations)

While the first method penalizes our model severely even if we go wrong in one out of the 27 genre predictions, the second method wouldn't be the best for this set of data given how imbalanced our data is.

Better option would be to identify how many genre labels we have correctly identified for each of the movies. Precision and Recall are metrics that measures the performance on the lowly occurring 'positive' class.

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Precision for a genre is given by

Precision(Genre=Action) = (Number of movies 'correctly' identified as Action Genre)/(Total number of movies that have been identified as Action Genre)

Recall is the ratio of correctly predicted positive observations to the all observations in actual class. Recall for a genre is given by

Recall(Genre=Action) = (Number of movies 'correctly' identified as Action Genre)/(Total number of Action Genre movies in the data set)

F1 Score is the Harmonic mean of Precision and Recall.

$$F1\ Score = 2*(Recall * Precision) / (Recall + Precision)$$

Below is the method we use to come up with a single F1 score for our model performance

- For each genre, compute Precision, Recall and F1 Score
- Compute a weighted average of the F1 score – weighted by the support (number of occurrences in the genre in our data set). This is used as our final metric.

6.4 Algorithm Details

Multi-Label Text based classification can be broadly summarized into the below 3 blocks

- Text Encoder – encodes text data into numeric vectors
- Multi-label Classification Algorithms – techniques to makes multiple label predictions
- Classification Models – ML models used to make predictions for a single class

6.4.1 Text Encoder

We cannot work with text directly when using machine learning algorithms. Instead, we need to convert the text to numbers. Text encoding is a process wherein the text data is converted into unique numeric vectors which can be consumed by ML models. Below are few text encoders that were considered for this project

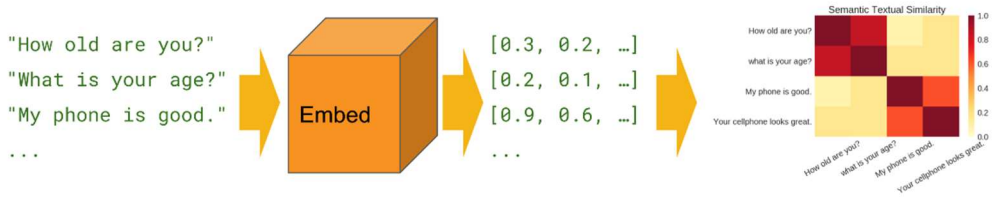
- Word Count with **Count Vectorizer** (Bag of Words Model): The CountVectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary. An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document.
- Word Frequencies with **TF-IDF Vectorizer** (Bag of Words Model): In the standard CountVectorizer model above, we used just the term frequency in a document of words in our vocabulary. In TF-IDF, we weight this term frequency by the inverse of its popularity in all documents. For example, if the word "movie" showed up in all the documents, it would not have

much predictive value. It could actually be considered a stop word. TF-IDF is obtained by down weighing its counts by 1 divided by its overall frequency.

- Sentence Embedding using **Google Universal Sentence Encoder** (USE): Universal Sentence Encoder is a tool released by Google that allows you to convert any sentence (including an entire plot) into a vector.

  There are two versions models released by Google – one of them is based on a Transformer architecture and the other one is based on Deep Averaging Network (DAN). Both models take a word, sentence or a paragraph as input and output a 512-dimensional vector. The transformer model is designed for higher accuracy, but the encoding requires more memory and computational time. The DAN model on the other hand is designed for speed and efficiency, and some accuracy is sacrificed

  Below is a figure describing sentence embedding using USE. Each of the sentences are transformed into a 512-length vector which preserves the meaning of the sentence.



6.4.2 Multi-label Classification Algorithms

Most traditional learning algorithms are developed for single-label classification problems. Therefore, a lot of approaches in the literature transform the multi-label problem into multiple single-label problems, so that the existing single-label algorithms can be used. Below are the two techniques that were considered

6.4.2.1 Binary Relevance

This is the simplest technique, which basically treats each genre as a separate single class classification problem. For each classifier, the class is fitted against all the other classes - hence `n_classes` classifiers are needed. The union of all classes that were predicted is taken as the multi-label output. We use the inbuilt sklearn OneVsRestClassifier[1] function to achieve this multi-label classification.

Probability (equal to the frequency of the genres occurrence) threshold is used for classifying each genre. In a circumstance that none of the `n_class` classifiers detect a genre, we pick one most likely genre based on the probability value.

6.4.2.2 Label Powerset

This approach does take partial correlations between genres into account. Here we treat each of the unique genre combinations found in the training data as a possible class. Hence, there can be worst case of 2^{n_genres} number of classes.

[1] OnevsRestClassifier is commonly used for multi-class classification; however, it also supports multi-label classification. To use this feature, feed the classifier an indicator matrix, in which cell [i, j] indicates the presence of label j in sample i

Below is the procedure adopted

- Transform (`n_rows x n_genres`) binary matrix from the training label set into into `n_rows x 1` label vector, where the column vector ranges from 0 to `num_genre_combinations` = number of unique values of genre combinations found in the training data set.
- Train the classifier using the training data set with labels corresponding to this `transformed n_rows x 1` column vector
- Predict the test data set using this fitted classifier. The output would be a column vector with each value ranging from 0 to `num_genre_combinations`
- Transform this column vector back to individual genres using the inverse mapping that was used in the first step
- Obtain the accuracy (precision/recall/f1 score) of the inverse transformed binary predicted genre matrix

6.4.2.3 Label Powerset with Clustering

We noticed that we had 1505 unique genre combinations in our data set. Though this is way below the maximum possible combination (which is 2^{27} = 134,217,728), this method, in general, is clearly not very robust with respect to different datasets. To achieve this, we would require a mechanism to control the maximum number of unique genre combinations. One method to go about this is to use clustering.

- Divide the (`n_rows x n_genres`) binary matrix into a k clusters using any of the well-known clustering techniques. In this section we use K-Means
- K-Means would transform the `train_y` input matrix into `n_rows` labels (ranging from 0 to k).
- The `cluster_center` (which is the mean of all the observations mapped to that cluster) would be used as a representative genre combination for that cluster (which are all provided the same label)
- `Cluster_center` are floating values (since they are averaged across several observations) from 0 to 1. Map it to either 0 or 1 using an appropriate threshold (0.85 is used this project)
- Some genres might never get included in this `cluster_center` mapping due to the above rounding operation. This happens when that genre has a very low occurrence. This would result in both precision and recall being 0 for this genre
    - In that case, look for the label which has maximum floating value for that genre and change that `cluster_center[label][genre]` to 1

6.4.3 Classification Models

These include the standard Machine Learning modeling algorithms. The algorithms that were considered include

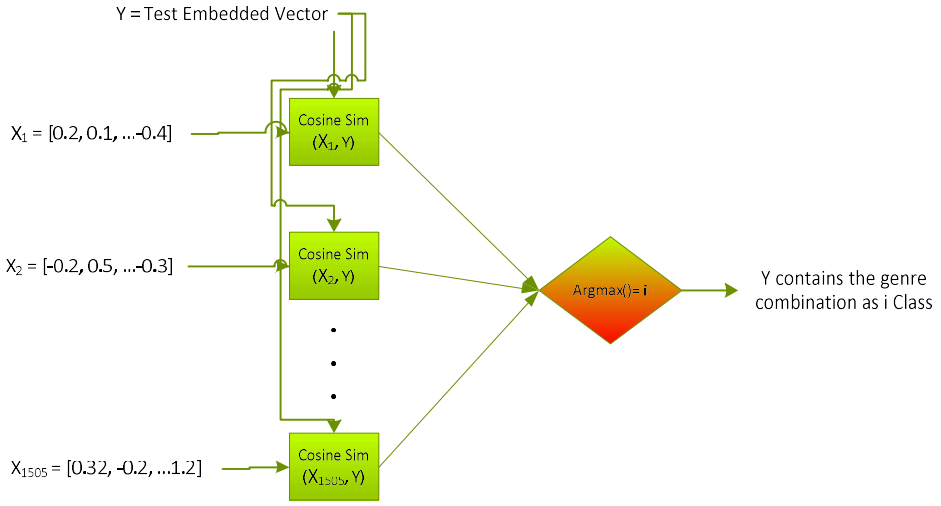
- Logistic Regression
- Naïve Bayes
- Linear Support Vector Machine Classifier
- Neural Network (Used with Sentence Embedding)

- Cosine Similarity (Used with Sentence Embedding): Cosine similarity is a metric used to determine how similar the documents. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space.

$$\text{Cosine Similarity } (x, y) = \frac{x.y}{\sqrt{(x.x)} * \sqrt{y.y}}$$

In this context, the two vectors that we will use are the embedded vector that we obtain from USE. Below is the procedure used in conjunction with Label Powerset method to make predictions using Cosine Similarity

  - Obtain the mean vector representation for each of the unique label combinations.
  - There are 1505 unique genre combinations
  - For each of these 1505 combinations, get a representative embedding vector = mean of all the vectors with similar genre combination
  - For each movie plot, get a cosine similarity with each of the 1505 genre combinations
  - Assign the genre combination which yields the maximum cosine similarity



6.4.4 Algorithm Summary

Below is a list of all the models + encoders tried out

Multi-label Classification Method	Encoder	Model
Binary Relevance	Count Vectorizer	Linear SVC
Binary Relevance	TF-IDF	Logistic Regression
Binary Relevance	TF-IDF	Naïve Bayes
Binary Relevance	TF-IDF	Linear SVC
Label Powerset	Count Vectorizer	Linear SVC
Label Powerset	TF-IDF	Naïve Bayes
Label Powerset	TF-IDF	Linear SVC

Label Powerset with Clustering (75 classes)	TF-IDF	Linear SVC
Label Powerset	Sentence Embedding	Cosine Similarity
Label Powerset	Sentence Embedding	Neural Network
Binary Relevance	Sentence Embedding	Neural Network

6.5 Binary Relevance

Binary relevance involves training classifier for each of the genres (27 in all). We then make predictions for each of them and then combine the result to declare all the genres that are predicted. All the classifiers are independently trained, and this method doesn't assume any dependency between features (genres)

6.5.1 Count Vectorizer + Linear SVC

Here, we use Count Vectorizer and 27 (=number of genres) Linear SVC classifiers. Below are the optimal hyper-parameters

Count Vectorizer	Ngram = (1, 2)	Min_df = 2	Max_df = 0.5
LinearSVC Classifier	C=1		

	Precision	Recall	F1-Score	Support
Action	0.88	0.59	0.71	4321.0
Adult	0.00	0.00	0.00	11.0
Adventure	0.87	0.53	0.66	3496.0
Animation	0.89	0.67	0.76	3333.0
Biography	0.75	0.16	0.27	354.0
Comedy	0.81	0.71	0.76	7320.0
Crime	0.88	0.68	0.76	4453.0
Documentary	0.75	0.60	0.67	1863.0
Drama	0.89	0.78	0.83	11067.0
Family	0.87	0.62	0.72	4173.0
Fantasy	0.87	0.50	0.64	2643.0
Game-Show	0.93	0.66	0.77	450.0
History	0.78	0.36	0.50	623.0
Horror	0.80	0.30	0.44	854.0
Music	0.88	0.51	0.65	654.0
Musical	0.88	0.24	0.38	182.0
Mystery	0.86	0.55	0.67	4114.0
News	0.90	0.65	0.75	681.0
Reality-TV	0.80	0.68	0.73	1748.0
Romance	0.90	0.67	0.77	4581.0
Sci-Fi	0.88	0.57	0.69	3055.0
Short	0.81	0.12	0.21	142.0
Sport	0.83	0.43	0.57	426.0
Talk-Show	0.88	0.71	0.79	809.0
Thriller	0.84	0.46	0.59	3254.0
War	0.90	0.42	0.57	388.0
Western	0.86	0.60	0.71	445.0
Avg/Total	0.86	0.63	0.72	65440.0

The F1-score result for this pipeline (encoder+model) is summarized below. An overall F1-score of 0.72 is achieved using this model

6.5.2 TF-IDF + Linear SVC

Here, we use TF-IDF vectorizer and 27 (=number of genres) Linear SVC classifiers. Below are the optimal hyper-parameters

TF-IDF Vectorizer	Ngram = (1, 2)	Min_df = 2	Max_df = 0.5
Linear SVC Classifier	C=1		

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.77 is achieved. Compared to Count Vectorizer, TF-IDF achieves a better F1-score

	Precision	Recall	F1-Score	Support
Action	0.88	0.70	0.78	4321.0
Adult	0.00	0.00	0.00	11.0
Adventure	0.85	0.64	0.73	3496.0
Animation	0.88	0.76	0.81	3333.0
Biography	0.83	0.19	0.32	354.0
Comedy	0.82	0.77	0.79	7320.0
Crime	0.85	0.78	0.82	4453.0
Documentary	0.71	0.71	0.71	1863.0
Drama	0.88	0.84	0.86	11067.0
Family	0.85	0.71	0.77	4173.0
Fantasy	0.87	0.61	0.71	2643.0
Game-Show	0.90	0.74	0.81	450.0
History	0.76	0.47	0.58	623.0
Horror	0.86	0.38	0.53	854.0
Music	0.88	0.61	0.72	654.0
Musical	0.97	0.33	0.49	182.0
Mystery	0.82	0.67	0.74	4114.0
News	0.88	0.71	0.79	681.0
Reality-TV	0.79	0.75	0.77	1748.0
Romance	0.89	0.75	0.81	4581.0
Sci-Fi	0.88	0.68	0.76	3055.0
Short	0.90	0.13	0.23	142.0
Sport	0.84	0.55	0.66	426.0
Talk-Show	0.86	0.79	0.83	809.0
Thriller	0.81	0.57	0.67	3254.0
War	0.90	0.53	0.67	388.0
Western	0.89	0.69	0.78	445.0
Avg/Total	0.85	0.72	0.77	65440.0

6.5.3 TF-IDF + Logistic Regression

Here, we use TF-IDF vectorizer and 27 (=number of genres) Logistic Regression classifiers. Below are the optimal hyper-parameters

TF-IDF Vectorizer	Ngram = (1, 1)	Min_df = 2	Max_df = 0.5

Logistic Regression Classifier	C = 100		

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.74 is achieved.

	Precision	Recall	F1-Score	Support
Action	0.74	0.79	0.76	4321.0
Adult	0.08	0.27	0.12	11.0
Adventure	0.67	0.76	0.71	3496.0
Animation	0.74	0.85	0.79	3333.0
Biography	0.20	0.56	0.30	354.0
Comedy	0.74	0.79	0.76	7320.0
Crime	0.76	0.82	0.79	4453.0
Documentary	0.48	0.76	0.59	1863.0
Drama	0.86	0.82	0.84	11067.0
Family	0.69	0.79	0.73	4173.0
Fantasy	0.65	0.75	0.70	2643.0
Game-Show	0.57	0.89	0.69	450.0
History	0.37	0.69	0.48	623.0
Horror	0.36	0.66	0.46	854.0
Music	0.53	0.79	0.64	654.0
Musical	0.25	0.66	0.36	182.0
Mystery	0.69	0.74	0.71	4114.0
News	0.52	0.81	0.64	681.0
Reality-TV	0.54	0.82	0.65	1748.0
Romance	0.78	0.81	0.79	4581.0
Sci-Fi	0.70	0.79	0.74	3055.0
Short	0.10	0.49	0.17	142.0
Sport	0.45	0.81	0.58	426.0
Talk-Show	0.56	0.87	0.68	809.0
Thriller	0.62	0.72	0.67	3254.0
War	0.47	0.75	0.58	388.0
Western	0.51	0.85	0.64	445.0
Avg/Total	0.70	0.79	0.74	65440.0

6.5.4 TF-IDF + Naïve Bayes

Here, we use TF-IDF vectorizer and 27 (=number of genres) Naïve Bayes classifiers. Below are the optimal hyper-parameters

TF-IDF Vectorizer	Ngram = (1, 2)	Min_df = 2	Max_df = 0.5
Naïve Bayes Classifier	Alpha = 0.01		

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.76 is achieved. With TF-IDF vectorizer, Linear SVC performs best with a F1 score of 0.77. Naïve Bayes classifier offers comparable performance with F1 score of 0.76. Logistic Regression is the weakest predictor among the lot offering a F1 score of 0.74

	Precision	Recall	F1-Score	Support
Action	0.84	0.70	0.76	4321.0
Adult	0.00	0.00	0.00	11.0
Adventure	0.78	0.70	0.74	3496.0
Animation	0.85	0.79	0.82	3333.0
Biography	0.45	0.32	0.38	354.0
Comedy	0.83	0.73	0.78	7320.0
Crime	0.82	0.80	0.81	4453.0
Documentary	0.60	0.74	0.66	1863.0
Drama	0.86	0.85	0.85	11067.0
Family	0.84	0.68	0.75	4173.0
Fantasy	0.81	0.65	0.72	2643.0
Game-Show	0.74	0.85	0.79	450.0
History	0.51	0.60	0.55	623.0
Horror	0.74	0.42	0.53	854.0
Music	0.67	0.61	0.64	654.0
Musical	0.94	0.37	0.54	182.0
Mystery	0.75	0.71	0.73	4114.0
News	0.64	0.81	0.71	681.0
Reality-TV	0.73	0.76	0.74	1748.0
Romance	0.84	0.72	0.78	4581.0
Sci-Fi	0.80	0.71	0.75	3055.0
Short	0.73	0.15	0.26	142.0
Sport	0.72	0.64	0.67	426.0
Talk-Show	0.62	0.86	0.72	809.0
Thriller	0.70	0.64	0.67	3254.0
War	0.72	0.63	0.68	388.0
Western	0.69	0.76	0.72	445.0
Avg/Total	0.80	0.73	0.76	65440.0

6.6 Label Powerset

In this method, we reduce a multi-label classification problem into a multi-class classification. We label each unique genre combination as a class. In our training data set, there exists 1505 different genre combinations (out of a maximum of 2^{27} combinations). Using these 1505 unique classes, we perform a multi-class classification. The classifier will pick one of these 1505 labels as the prediction

6.6.1 Count Vectorizer + Linear SVC

Here, we use Count Vectorizer and Linear SVC classifiers. Below are the optimal hyper-parameters

Count Vectorizer	Ngram = (1, 1)	Min_df = 1	Max_df = 0.5
LinearSVC Classifier	C=1		

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.79 is achieved. Higher prediction accuracy is achieved using Label Powerset compared to Binary Releavance.

	Precision	Recall	F1-Score	Support
Action	0.83	0.76	0.79	4321.0
Adult	0.75	0.27	0.40	11.0
Adventure	0.80	0.72	0.76	3496.0
Animation	0.82	0.78	0.80	3333.0
Biography	0.46	0.37	0.41	354.0
Comedy	0.85	0.79	0.82	7320.0
Crime	0.82	0.81	0.81	4453.0
Documentary	0.65	0.65	0.65	1863.0
Drama	0.89	0.84	0.86	11067.0
Family	0.79	0.78	0.79	4173.0
Fantasy	0.81	0.71	0.76	2643.0
Game-Show	0.72	0.84	0.78	450.0
History	0.60	0.56	0.58	623.0
Horror	0.63	0.53	0.57	854.0
Music	0.72	0.76	0.74	654.0
Musical	0.72	0.58	0.64	182.0
Mystery	0.82	0.74	0.78	4114.0
News	0.68	0.75	0.71	681.0
Reality-TV	0.75	0.72	0.73	1748.0
Romance	0.87	0.83	0.85	4581.0
Sci-Fi	0.82	0.73	0.77	3055.0
Short	0.44	0.39	0.41	142.0
Sport	0.67	0.66	0.66	426.0
Talk-Show	0.72	0.79	0.76	809.0
Thriller	0.78	0.68	0.72	3254.0
War	0.74	0.60	0.66	388.0
Western	0.68	0.76	0.72	445.0
Avg/Total	0.81	0.76	0.79	65440.0

6.6.2 TF-IDF + Naïve Bayes

Here, we use TF-IDF vectorizer and Naïve Bayes classifiers. Below are the optimal hyper-parameters

TF-IDF Vectorizer	Ngram = (1, 1)	Min_df = 2	Max_df = 0.5
Naïve Bayes Classifier	Alpha = 0.001		

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.64 is achieved

	Precision	Recall	F1-Score	Support
Action	0.98	0.44	0.61	4321.0
Adult	0.00	0.00	0.00	11.0
Adventure	0.98	0.37	0.54	3496.0
Animation	0.95	0.48	0.63	3333.0
Biography	0.92	0.09	0.17	354.0
Comedy	0.72	0.73	0.72	7320.0
Crime	0.91	0.53	0.67	4453.0
Documentary	0.46	0.63	0.53	1863.0
Drama	0.88	0.75	0.81	11067.0
Family	0.95	0.41	0.57	4173.0
Fantasy	0.98	0.38	0.55	2643.0
Game-Show	0.98	0.55	0.70	450.0
History	0.85	0.27	0.41	623.0
Horror	0.96	0.23	0.37	854.0
Music	0.96	0.35	0.51	654.0
Musical	0.97	0.17	0.29	182.0
Mystery	0.86	0.48	0.62	4114.0
News	0.88	0.52	0.66	681.0
Reality-TV	0.58	0.69	0.63	1748.0
Romance	0.86	0.66	0.74	4581.0
Sci-Fi	0.99	0.36	0.52	3055.0
Short	0.92	0.08	0.15	142.0
Sport	0.94	0.38	0.54	426.0
Talk-Show	0.77	0.59	0.67	809.0
Thriller	0.91	0.38	0.54	3254.0
War	0.94	0.39	0.55	388.0
Western	0.76	0.54	0.63	445.0
Avg/Total	0.87	0.54	0.64	65440.0

6.6.3 TF-IDF + Linear SVC

Here, we use TF-IDF vectorizer and Linear SVC classifiers. Below are the optimal hyper-parameters

TF-IDF Vectorizer	Ngram = (1, 2)	Min_df = 2	Max_df = 0.5
Linear SVC Classifier	C=10		

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.83 is achieved.

	Precision	Recall	F1-Score	Support
Action	0.87	0.82	0.84	4321.0
Adult	0.38	0.27	0.32	11.0
Adventure	0.85	0.80	0.82	3496.0
Animation	0.86	0.84	0.85	3333.0
Biography	0.54	0.43	0.48	354.0
Comedy	0.87	0.84	0.85	7320.0
Crime	0.86	0.86	0.86	4453.0
Documentary	0.72	0.73	0.73	1863.0
Drama	0.91	0.87	0.89	11067.0
Family	0.82	0.84	0.83	4173.0
Fantasy	0.86	0.78	0.81	2643.0
Game-Show	0.79	0.90	0.85	450.0
History	0.64	0.65	0.65	623.0
Horror	0.75	0.60	0.67	854.0
Music	0.76	0.80	0.78	654.0
Musical	0.75	0.65	0.70	182.0
Mystery	0.85	0.80	0.82	4114.0
News	0.76	0.80	0.78	681.0
Reality-TV	0.80	0.78	0.79	1748.0
Romance	0.87	0.86	0.87	4581.0
Sci-Fi	0.88	0.80	0.84	3055.0
Short	0.53	0.40	0.46	142.0
Sport	0.75	0.78	0.76	426.0
Talk-Show	0.77	0.86	0.81	809.0
Thriller	0.84	0.73	0.78	3254.0
War	0.74	0.71	0.72	388.0
Western	0.68	0.85	0.75	445.0
Avg/Total	0.85	0.82	0.83	65440.0

6.6.4 Label Powerset with Clustering + TF-IDF + Linear SVC

Here, we use 75 clusters to group the 1505 genre combinations. The cluster center is used to represent the genre combinations for these cluster. The cluster center is further quantized to binary values using a threshold of 0.85; if the genre value of the cluster center > 0.85, then the genre for that cluster center = 1, else 0.

We use TF-IDF vectorizer and Linear SVC classifiers. Below are the optimal hyper-parameters

TF-IDF Vectorizer	Ngram = (1, 2)	Min_df = 2	Max_df = 0.5
Linear SVC Classifier	C=10		

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.70 is achieved. F1-score decreases from 0.83 to 0.7 due to reduction in the label combinations from 1505 to 75.

	Precision	Recall	F1-Score	Support
Action	0.90	0.53	0.67	4321.0
Adult	0.00	0.09	0.00	11.0
Adventure	0.89	0.50	0.64	3496.0
Animation	0.90	0.65	0.75	3333.0
Biography	0.58	0.06	0.11	354.0
Comedy	0.85	0.67	0.75	7320.0
Crime	0.88	0.74	0.81	4453.0
Documentary	0.69	0.62	0.65	1863.0
Drama	0.90	0.76	0.82	11067.0
Family	0.88	0.58	0.70	4173.0
Fantasy	0.88	0.42	0.57	2643.0
Game-Show	0.88	0.52	0.66	450.0
History	0.63	0.32	0.42	623.0
Horror	0.66	0.09	0.16	854.0
Music	0.85	0.27	0.41	654.0
Musical	0.06	0.07	0.06	182.0
Mystery	0.87	0.62	0.72	4114.0
News	0.86	0.74	0.80	681.0
Reality-TV	0.72	0.77	0.75	1748.0
Romance	0.89	0.70	0.78	4581.0
Sci-Fi	0.93	0.49	0.64	3055.0
Short	0.07	0.14	0.09	142.0
Sport	0.78	0.20	0.32	426.0
Talk-Show	0.83	0.80	0.82	809.0
Thriller	0.85	0.41	0.56	3254.0
War	0.74	0.08	0.14	388.0
Western	0.70	0.57	0.62	445.0
Avg/Total	0.86	0.61	0.70	65440.0

6.7 Sentence Embedding

In this section, we obtain vectorized representation of sentence from USE. The length of the vector obtained is 512.

6.7.1 Cosine Similarity

Here we use cosine similarity to obtain the genre combination for each of the movie plot. Cosine similarity is computed between the movie plot embedding and each of the embedding vector representing the 1505 different genre combinations. The genre combination with the maximum cosine similarity is chosen as the prediction

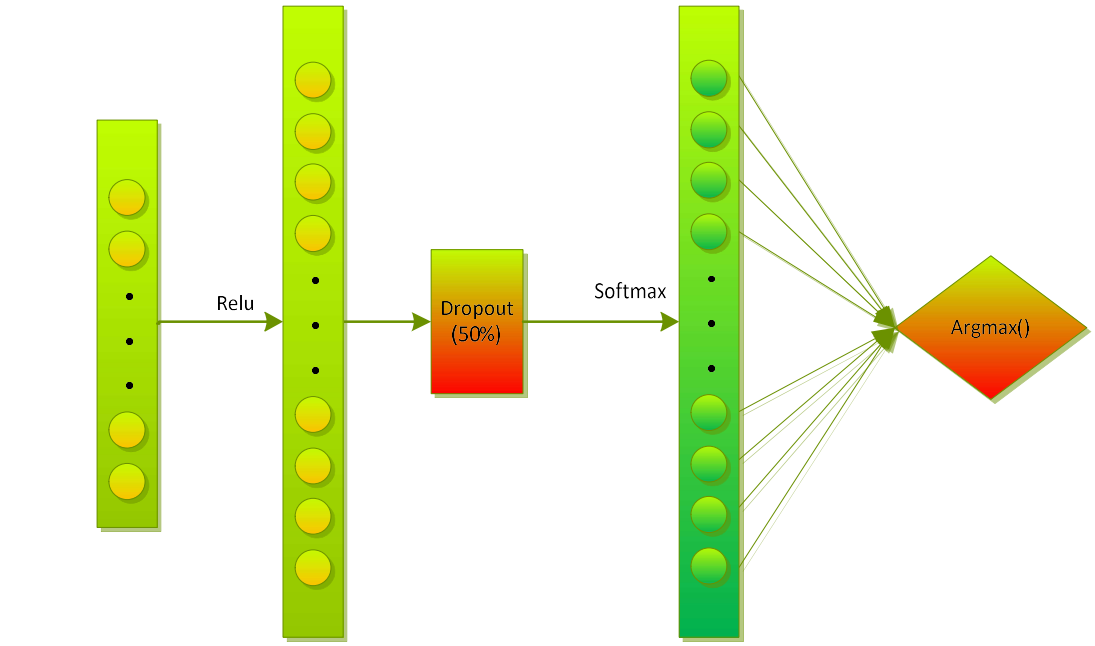
The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.60 is achieved

	Precision	Recall	F1-Score	Support
Action	0.58	0.68	0.63	4321.0
Adult	0.13	0.27	0.18	11.0
Adventure	0.50	0.66	0.57	3496.0
Animation	0.69	0.71	0.70	3333.0
Biography	0.18	0.47	0.26	354.0
Comedy	0.61	0.64	0.62	7320.0
Crime	0.65	0.67	0.66	4453.0
Documentary	0.47	0.53	0.50	1863.0
Drama	0.75	0.73	0.74	11067.0
Family	0.53	0.63	0.57	4173.0
Fantasy	0.49	0.64	0.56	2643.0
Game-Show	0.54	0.83	0.66	450.0
History	0.30	0.62	0.40	623.0
Horror	0.29	0.55	0.38	854.0
Music	0.32	0.70	0.44	654.0
Musical	0.13	0.38	0.19	182.0
Mystery	0.49	0.56	0.52	4114.0
News	0.45	0.55	0.50	681.0
Reality-TV	0.49	0.61	0.54	1748.0
Romance	0.54	0.70	0.61	4581.0
Sci-Fi	0.59	0.69	0.64	3055.0
Short	0.11	0.39	0.18	142.0
Sport	0.27	0.73	0.40	426.0
Talk-Show	0.47	0.61	0.53	809.0
Thriller	0.45	0.49	0.47	3254.0
War	0.35	0.59	0.44	388.0
Western	0.37	0.54	0.44	445.0
Avg/Total	0.57	0.65	0.60	65440.0

6.7.2 Neural Networks – Label Powerset

Here, we use a Neural Network to make prediction. The neural network used is shown in the figure below. The input layer consists of 512 nodes (equal to the number of features in the input vector). A single hidden layer comprising of a fully connected dense layer is used with 1024 nodes. The optimal size of the hidden layer is usually between the size of the input and the size of the output layers. ReLu is used as the activation function to introduce non-linearity into the prediction. 50% of the neurons are dropped out to prevent overfitting. Finally, we have a output layer with 1505 neurons. Softmax is used as the activation function at the output layer, since a single class must be picked among the 1505 classes.

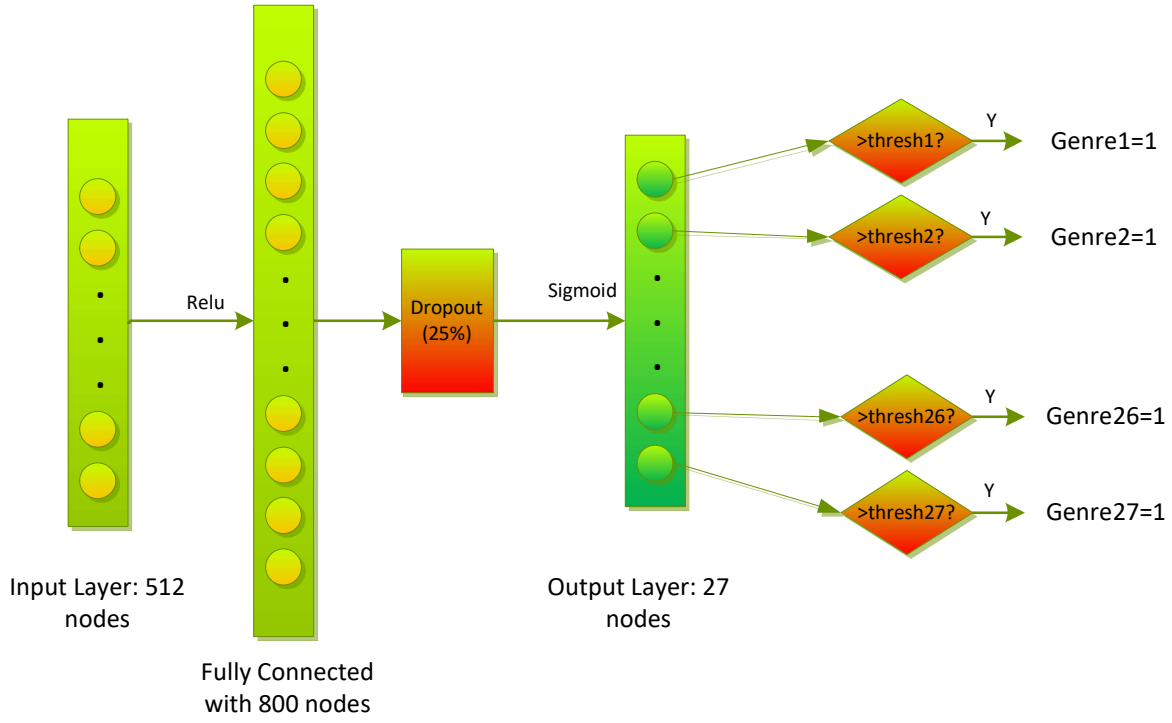
The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.62 is achieved



	Precision	Recall	F1-Score	Support
Action	0.80	0.43	0.56	4321.0
Adult	0.50	0.09	0.15	11.0
Adventure	0.80	0.41	0.54	3496.0
Animation	0.82	0.64	0.72	3333.0
Biography	0.59	0.16	0.25	354.0
Comedy	0.73	0.56	0.63	7320.0
Crime	0.82	0.60	0.69	4453.0
Documentary	0.60	0.64	0.62	1863.0
Drama	0.85	0.74	0.79	11067.0
Family	0.79	0.40	0.53	4173.0
Fantasy	0.79	0.41	0.54	2643.0
Game-Show	0.88	0.35	0.50	450.0
History	0.74	0.32	0.44	623.0
Horror	0.68	0.20	0.31	854.0
Music	0.71	0.39	0.50	654.0
Musical	0.64	0.13	0.21	182.0
Mystery	0.74	0.47	0.57	4114.0
News	0.79	0.51	0.62	681.0
Reality-TV	0.57	0.68	0.62	1748.0
Romance	0.74	0.57	0.64	4581.0
Sci-Fi	0.86	0.52	0.65	3055.0
Short	0.65	0.08	0.14	142.0
Sport	0.69	0.40	0.50	426.0
Talk-Show	0.73	0.66	0.69	809.0
Thriller	0.71	0.36	0.48	3254.0
War	0.75	0.41	0.53	388.0
Western	0.53	0.57	0.55	445.0
Avg/Total	0.77	0.54	0.62	65440.0

6.7.3 Neural Network – Binary Relevance

Here, we use Sentence Embedding along with Neural Network to make predictions. Below is the block diagram for the Neural Network used. Like the previous model, the input layer consists of 512 nodes (equal to the number of features in the input vector). We use a single hidden layer comprising of a fully connected dense layer is used with 800 nodes. ReLu is used as the activation function to introduce non-linearity into the prediction. 25% of the neurons are dropped out to prevent overfitting. Finally, we have a output layer with 27 neurons (which corresponds to each genre). We use a Sigmoid activation function here which maps the output to a number between 0 and 1 (can be interpreted as a probability). We can then use separate thresholds for each of these values to make independent predictions on 27 genres.



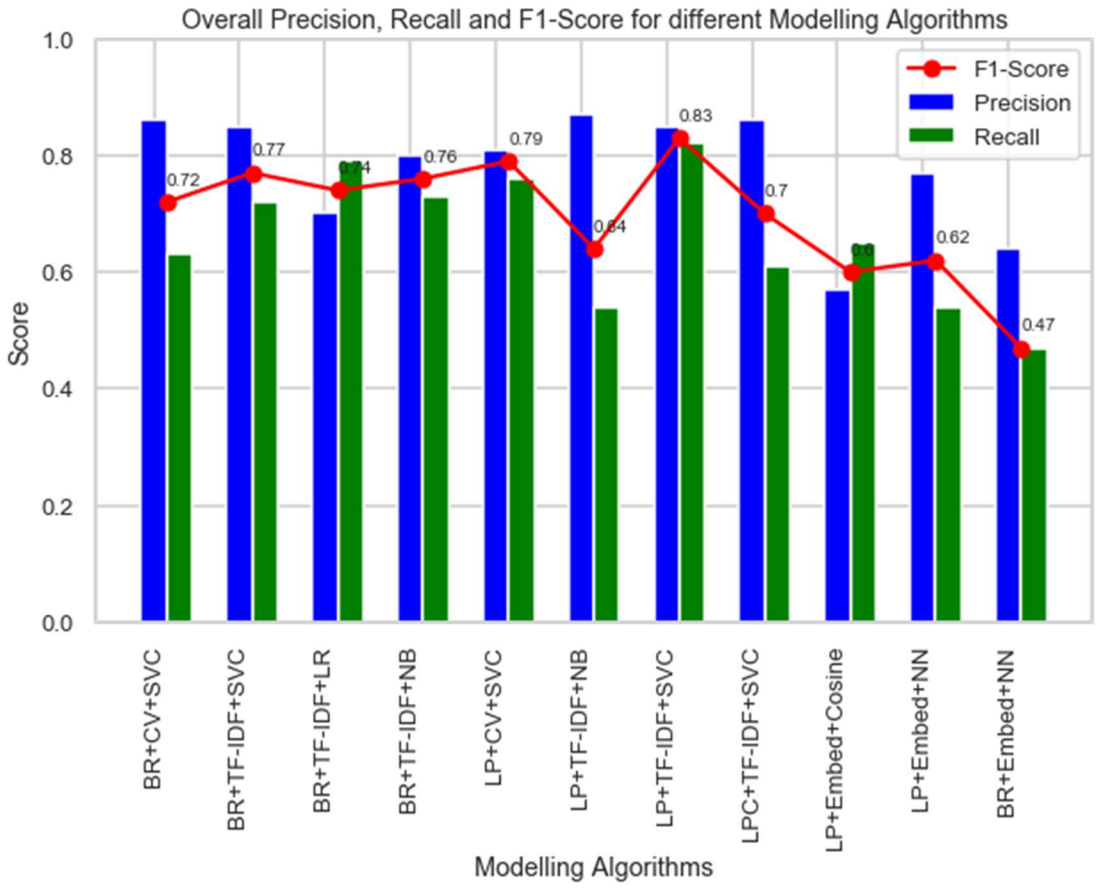
	Precision	Recall	F1-Score	Support
Action	0.59	0.58	0.58	4321.0
Adult	0.01	0.64	0.03	11.0
Adventure	0.55	0.67	0.60	3496.0
Animation	0.69	0.80	0.74	3333.0
Biography	0.12	0.61	0.20	354.0
Comedy	0.76	0.23	0.35	7320.0
Crime	0.73	0.64	0.68	4453.0
Documentary	0.57	0.64	0.61	1863.0
Drama	0.82	0.09	0.16	11067.0
Family	0.64	0.51	0.57	4173.0
Fantasy	0.46	0.68	0.55	2643.0
Game-Show	0.47	0.72	0.57	450.0
History	0.30	0.66	0.41	623.0
Horror	0.24	0.61	0.34	854.0
Music	0.36	0.63	0.46	654.0
Musical	0.05	0.57	0.10	182.0
Mystery	0.59	0.54	0.56	4114.0
News	0.44	0.75	0.55	681.0
Reality-TV	0.60	0.61	0.61	1748.0
Romance	0.74	0.28	0.40	4581.0
Sci-Fi	0.57	0.72	0.63	3055.0
Short	0.03	0.37	0.05	142.0
Sport	0.30	0.70	0.42	426.0
Talk-Show	0.55	0.73	0.63	809.0
Thriller	0.45	0.60	0.51	3254.0
War	0.25	0.71	0.37	388.0
Western	0.32	0.68	0.43	445.0
Avg/Total	0.64	0.47	0.47	65440.0

The F1-score result for this pipeline (encoder+model) is summarized in the below table. An overall F1-score of 0.64 is achieved. This is very poor when compared to the simple ML models used above. This is probably because we didn't really train the encoder with our data. The USE encoder is trained on lots of external data and might not be optimized for this purpose

7 SUMMARY & CONCLUSIONS

7.1 F1 Score

We summarize the average overall Precision, Recall and F1-score for every model we used in the below bar chart.



Acronym used for the above plot:

- Multi-label classification Technique
  - BR: Binary Relevance
  - LP: Label Powerset

    - LPC: Label Powerset with Clustering
- Text Encoding
    - CV: Count Vectorizer
    - TF-IDF: TF-IDF Vectorizer
    - Embed: Sentence Embedding via USE
- Classifier Model
    - SVC: Linear Support Vector Classifier
    - LR: Logistic Regression
    - NB: Naïve Bayes
    - Cosine: Cosine Similarity
    - NN: Neural Network

Based on the above results, the best predicting model uses TF-IDF Vectorizer, Linear Support Vector Classifier and Label Powerset approach to make multi-label classification and achieves an overall F1 score of 0.83

7.2 Summary

In this report, we trained a model to predict all the genres (up to 27 possible genres) that a movie can be classified into based on its plot via several techniques. The dataset was obtained from IMDB dataset.

We considered 3 text encoders (Count Vectorizer, TF-IDF Vectorizer, Sentence Embeddings with USE), 2 multi-label classification techniques (Binary Relevance, Label Powerset) and 5 modelling algorithms (Logistic Regression, Naïve Bayes, Linear Support Vector Classifier, Neural Networks, Cosine Similarity).

7.2.1 Data Exploration Conclusions

In this project, we looked at several aspects of movie plots and genres. Below is a short summary of the EDA conclusions

- Drama and Comedy are the most popular genre
- On an average, a movie is classified into 2 genres (and a maximum of 12 genres!)
- Wordcloud plots reveal the important feature (words) used in each genre – few of which stands out include ‘german’ for War, ‘perform, band’ for Music
- Few strongly correlated genres include – a) Crime, Mystery & Thriller, b) Drama & Romance. Animation and Drama are among genres which are negatively correlated.
- 80% of Romance, Crime, Mystery and Thriller movies are also categorized as Drama
- 50% of Animation, Musical and Short movies are also Comedies
- Most of Adult, Documentary, Reality-TV, Talk-Show and Western genres have just a single label

7.2.2 Modeling Conclusions

Below is a summary of model performances

- The best predicting model uses TF-IDF Vectorizer, Linear Support Vector Classifier and Label Powerset approach to make multi-label classification and achieves an overall F1 score of 0.83
- Models using Sentence Embedding preformed worse compared to the other Vectorizers and ML Models.

7.2.3 Limitations and Scope for Model Improvements

Below are few limitations in this analysis and ideas to further improve model prediction accuracy

- Sentence embedding doesn't require the sentence lemmatization, or stop word removal, or in fact any of the text preprocessing steps. It is supposed to work with sentences in its raw form. Use the original text before preprocessing to obtain sentence embedding
- Use Sentence Embedding with Linear SVC