



# Movie genre classification using binary relevance, label powerset, and machine learning classifiers

Sanjay Kumar<sup>1</sup> · Nikhil Kumar<sup>1</sup> · Aditya Dev<sup>1</sup> · Siraz Naorem<sup>1</sup>

Received: 18 January 2021 / Revised: 22 February 2022 / Accepted: 11 May 2022 /

Published online: 11 June 2022

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

## Abstract

Multi-label text classification (MLTC) is a technique to categorize texts into more than a single category and used extensively in various real-life problems. Such classifications problems are challenging and dependent on many factors and changes according to the problem. Movie genre classification is a popular multi-label text classification problem as movies may belong to multiple genres at the same time. The major factors used for movie genre classification are based on parameters like movie plot, title, summary, and subtitles. In recent years, some neural networks based approaches are proposed for solving such problems, which turns the solution into resource intensive and time consuming activities. In this paper, we propose a novel method of movie genre classification using a combination of problem transformation techniques, namely binary relevance (BR) and label powerset (LP), text vectorizers and machine learning classifier models. We perform binary relevance task (BR) that converts multi-label classification tasks into independent binary classification tasks whereas label powerset transforms a multi-label problem into a multiclass problem with one multiclass classifier trained on all unique label combinations found in the training data. Further, we apply text vectorizers namely, CV (Count Vectorizer) and TF-IDF (Term Frequency - Inverse Document Frequency) to tokenize the textual data to build a word vocabulary followed by employing various classifiers i.e., Logistic Regression (LR), Multinomial Naive Bayes (MNB), K-Nearest Neighbor (KNN), Support Vector Classifier (SVC) with the combination of different vectorizers and problem transformation methods. To test the effectiveness of these combinations, we use the k-fold cross-validation technique. We construct different combination using problem transformation approaches, text vectorizers and classifier models leading to overall 16 different combinations for classifying movies into appropriate genres. Finally, we evaluate the performance of each combination on publicly available IMDb datasets with target on 27 major parent genres using different performance measures and reveal that the best result is obtained using the combination comprising of label powerset (LP) as Problem transformation approach, TF-IDF as the text vectorizer and support vector classifier (SVC) as the machine learning classifier model with a commendable accuracy of 0.95 and F1-score of 0.86.

---

✉ Sanjay Kumar  
sanjay.kumar@dtu.ac.in

**Keywords** Binary relevance · Label powerset · Machine learning classifiers · Movie genre classification · Multi-label text classification · Support vector classifier

## 1 Introduction

Nowadays, in numerous real-life applications like applications like recommender system, music categorization and semantic scene classification, Multi-label text classification (MLTC) methods are extensively used [1]. For movie based recommendation systems, movie genre has a crucial role. Multi-label text classification (MLTC) is a most popular task in natural language processing (NLP) that focuses to allocate multiple labels for each given text [12]. It is progressively required in various latest applications and technologies, such as document classification [16], tag recommendation [31], and context suggestion [37]. Movie genre classification is a popular Multi-label text classification (MLTC) problem as a movie can belong to multiple genres and identifying correct genre is a challenging problem because movie is associated with one or more and it is difficult to predict the possible genre corresponding to that movie as there can be many combination of genre is possible. Automation in movie genre classification can be useful in applications like checking for a specific type of film in multimedia databases based on indexing, filtering the movies based on user preference modeling, and provides content filtering and summarizing in movies [38]. Therefore, many research work has been proposed for automatic movie genre classification [5, 26, 35]. To cater to such a large audience of people, movies aren't static to their plots and their story revolves around a diverse range of categories such as romance, crime, thriller, action, mystery or even horror just to name a few. In order to intrigue a large audience and introduce them to possible more categories of movie, it is common for many movies to have a mix of many genres together. Ghostbusters is a sci-fi as well as horror movie, Jumanji is a mystery as well as action movie, Avengers is a sci-fi movie with aspects of a thriller, action, adventure and fantasy genre, the list goes on and on. That simple observation makes the movie genre task very suitable and a reliable way to test out multi-labelling classifications. Multi-label classifications has much more complications, challenges and factors that might affect its accuracy with respect to that of single category classification. Another such example is that of photos and posts on social media sites which can relate to multiple hashtags – wildlife, photography, nature etc. A great example- a college document can be categorized according to the department such as Physics and Application, Computer Science, another categorization in the area of Theory and Biology and a third categorization as Mathematics and Application in Physics. This issue would then have at least six labels or classes (Physics, Theory, Application, Biology, Mathematics and Computer Science) [6]. Today also, solving text classification is the most major concern for multi-label classification techniques. Since the possibilities of a single input being assigned many labels exists, there exists a multiple classification problem, a problem which needs to be addressed and viewed differently from its singular classification counterpart. In recent years, numerous deep learning and artificial neural network based techniques have been introduced to solve various real-life problems like computer vision, image and video captioning, multi-label text classification (MLTC) with an improved performance [3, 9, 21]. However, such methods usually demand intensive resources and require more time to complete the executions.

In this paper, we propose a novel method of movie genre classification based on textual information and extracting features from synopses and plots. Movies can have a wide range of associated parameters such as trailer, plot summary or synopsis, overview, poster, and subtitles which can facilitate the classification of movies into its suitable multiple genres.

The proposed approach utilizes problem transformation approach namely binary relevance, and label powerset, where binary relevance breaks down multi-label classification into distinct binary classification problems, and label powerset converts the multi-label classification problem into multiclass classification problem by generating the set of all possible combinations of the labels present in the dataset. It is then followed by the two approaches of text vectorizers namely, count vectorizer, and TF-IDF leading to encoding of the textual data into a numerical format, which is fit to be fed into machine learning classifiers for classification task. We use classical machine learning classifiers namely, multinomial naive Bayes (MNB), logistic regression (LR), K-nearest neighbor (KNN), support vector classifier (SVC) in the paper. By combining problem transformation approach, text vectorizers and machine learning classifiers, we obtain 16 different combination for classifying movies into their appropriate genres. We test the different combination of models on datasets fetched from websites IMDb by focussing on 27 major unique genres. We evaluate the them on the basis of performance measures like precision, recall, F1-score and accuracy. The result obtained by the proposed model outperforms several existing methods of movie genre classification. The main contributions of the proposed work can be summarized as follows:

- We devise a generic framework to apply popular problem transformation techniques on different movie datasets to transform the problem.
- We exploit plot summary, overview and subtitles of the movies for performing classification task by utilising text vectorizers and machine learning classifiers.
- We apply  $k$ -fold cross validation to validate the model combinations. It helps to estimate the suitability of the movie genre classifiers for each dataset.
- Experimental results obtained on different dataset using the multiple evaluation criteria disclose the benefits of the proposed model.

The rest of the paper is organized as follows. Section 2 discusses some related work in the field of multi-label text classification in the context of movie genre classification. Section 3 presents some background topics like the definition of multi-label text classification problems, basic details about various problem transformation techniques, count vectorizer, and machine learning classifiers related to our research work. Section 4 describes the proposed methodology. Section 5 discusses various datasets used in this study and different performance criteria. Section 6 presents intensive experimental results performed. Finally, Section 7 concludes our work.

## 2 Related work

Multi-label text classification problems come under the category of supervised learning, where data can be categorized into multiple tags or labels. This is contrary to the conventional task of single-label classification (i.e., multiclass or binary), where each data is only concerned with only one class label. A progressive consideration is given to multi-label context as it is applicable to a wide range of domains, including video, scene categorization, and text classification. Due to such vast potential requirements, many research works have been conducted to find different algorithms and approaches to solve multi-label problems. There are numerous methods to solve the multi-label classification problems. One of them is the problem transformation approach (PTA), which is further divided into three categories a) Transformation into binary classification problem called binary relevance (BR) method, a baseline approach to tackling the multi-label problems. We present below some notable recent works done in multi-label text classification problems with a focus on

movie genre classification. Doshi and Zadrozny [10] proposed a movie genre prediction by applying discourse features derived through topological data analysis (TDA), namely homological persistence. They analyzed movie data in a manner insensitive to the particular metric chosen and improved classification results of movie genre detection on the IMDb dataset. Hoang [14] explored several Gated Recurrent Units (GRU) based neural networks to predict movie genres based on plot summaries. He also utilized Naive Bayes, Word2Vec with XGBoost, and Recurrent Neural Networks classifiers, K-binary transformation, and rank method for movie genre prediction. Ertugrul and Karagoz [11] performed movie genre classification from plot summaries of movies using bidirectional LSTM (Bi-LSTM). The summary of the movie plots is divided into sentences, wherein each sentence is then assigned a genre. They experimented with both RNN and BiLSTM and other algorithms such as logistic regression to conclude that BiLSTM achieved the best with an F1-Score of 67.61. Portolese et al. [27] proposed a method for extracting features from the P-TMDB dataset for Portuguese movie synopses. Their approach utilized TF-IDF classifiers and a combination of other feature groups scoring an f1-score of 0.611. Christiyanto Saputra et al. [28] aimed to classify documents by their content. The movie synopses used in their dataset were of Polish origin. Their study experimented with a lot of traditional machine learning algorithms, with their best model being a combination of TF-IDF, Unigrams, and SVM, achieving an F1-Score of 0.45. Mangolin et al. [24] proposed a multimodal approach for multi-label classification problems. The dataset was treated and curated from TMDb, consisting of 152,622 movie entries with their features ranging from subtitles, synopses, and movie titles. The genres were to be chosen from 18 different possible labels. They computed different kinds of descriptors, such as Mel Frequency Cepstral Coefficients (MFCCs), Statistical Spectrum Descriptor (SSD), Local Binary Pattern (LBP) from spectrograms, Long short-term memory(LSTM), and convolutional neural networks (CNN). The final model used classifiers such as Binary Relevance and ML-kNN techniques. Their final model of fusion of an LSTM trained on synopses and another LSTM model trained on subtitles scored the best result of F-Score (0.674) and AUC-PR (0.725) metrics.

Tianshi Wang et al. [34] proposed a solution to multi-label text classification problem where new words and text categories are put under inspection for more accurate and robust classifications. Word embeddings and clustering is used to select semantic words suited for a semantic representation model and deep neural network (DSRM-DNN). Their final model classifies text elements by a weighted combination of word attributes which work as the elements of DSRM-DNN. During the classification process, words that appear in low frequency are subject to semantic words under sparse constraint and re-expressed. Linkun Cai et al. [4] introduced a method of tagging documents with a series of labels to preserve semantic relations between words, which are otherwise ignored and leads to information loss for multi-label text classification. Their idea revolves around explicit labels, which preserve semantics and hence, improve multi-label text classification. Their first model comprises a hybrid neural network that utilizes a pre-trained BERT model. Their second model proposes a neoteric attention mechanism to establish semantic connections between labels and words. Yu et al. [39] proposed a framework for movie trailer genre classification consisting of two modules. The first module, named spatio-temporal adopts advanced convolution neural networks to extract spatio-temporal features from trailer frames that capture the said features. The second module, named the attention-based sequential module, is designed to process the extracted spatio-temporal features for capturing global high-level sequential semantic concepts. Xia et al. [36] attempted to solve several existing issues, such as exploring weights of classifiers for better classification selection and researching the

relationship between pairwise label correlations, and investigating multi-classification performance. Their novel solution consisted of a stacked ensemble approach to exploit label correlations and the process of learning weights for classifiers acting as ensemble members. Further, an optimization algorithm was developed based on accelerated proximal gradient and block coordinate descent techniques to efficiently achieve the optimal ensemble solution.

After a thorough review of past attempts and advancements, in this paper, we intend to focus on textual information and extracting features from synopses and plots instead of unstructured data like images and posters. Since neural networks based models are usually complicated and demand intensive resources, we attempt to find a better combination of methods and techniques such that it performs better than the previous similar works and requires ordinarily available resources. Our model aims to develop a much better execution by combining traditional machine learning models along with problem transformation approaches and text vectorizers. We carefully select a set of unique genres suitable for proper analysis as many different genres somehow share the same base genre. For example, Ghost, Monster, Vampire, Zombie, Slasher genres share the same parent genre, i.e., Horror. These selected genres include the parent genres, which reduce the training efforts and, as the classification is now limited to the most important base genres only, helps in improving the model's performance.

### 3 Background

#### 3.1 Multi-label text classification

In most cases, the majority of the classification algorithms work with only single labels, i.e., each and every instance of the dataset is related to only one single label. Such examples assume that only a single label is enough for describing all the characteristics and properties of the instance. However, an instance in real life can have a set of many labels associated with itself. In such cases, the algorithms need to be able to associate multiple labels with the instance simultaneously [12, 19, 32].

In more formal terms, assume a training set  $D$ , with  $N$  instances. Each instance is represented by the notation  $E_i$  from a set of instances  $E_i = (x_i, Y_i)$ , where  $0 < i < N$  and  $i$  is a positive integer. Let each and every instance of this dataset be associated with multiple labels, represented by  $x_i = (x_{i1}, x_{i2}, \dots, x_{iM})$  and a subset of labels  $Y_i$ . Let  $L$  be a superset of all the possible  $q$  labels such that  $L = \{y_j : j = 1..q\}$  and  $Y_i \subseteq L$ .

Table 1 shows the respective representation. Hence, for any unknown set of labels  $U_j$  for an  $j$ th instance  $E = (x_j, U_j)$ ,

we can have our algorithm form an accurate classifier  $H$ . The result classifier  $H$  would be capable of predicting a correct set of labels denoted by  $Y$ , such that  $H(E) \rightarrow Y$ , where the label  $Y$  is related to the instance  $E$ .

#### 3.2 Problem transformation approach

The main notion behind PTA is for transforming the multi-labeling problem into a single-label classification task. PTA is independent of any algorithm method [12], since PTA functions without directly affecting the classification. From a wide variety of possible PTAs, some are more widely used in literature [12, 30], which are:

**Table 1** Representation of Multi-label Classification with  $x_{.j}$  labels and  $Y_{.i}$  as the correct set of labels of the instance  $E$

	x				Y
$E_1$	$x_{11}$	$x_{12}$	...	$x_{1M}$	$Y_1$
$E_2$	$x_{21}$	$x_{22}$	...	$x_{2M}$	$Y_2$
⋮	⋮	⋮	⋮	⋮	⋮
$E_N$	$x_{N1}$	$x_{N2}$	...	$x_{NM}$	$Y_N$

- (i) Binary Relevance :- PTA breaks down the multi-label classification task into distinct single-label binary classification tasks. Each and every of the single labels are distinct for every q labels in the training set  $L = \{y_1, y_2, \dots, y_q\}$ . The q binary datasets, denoted by  $D_{y_j}$ , where each  $D_{y_j}$  contains all the examples of the original training set, for  $j = 1..q$ , are associated with a binary label  $y_j$  signifying the presence or absence of the label in that example [40]. After the transformation, a new set of q binary classifiers is constructed, denoted by  $H_j(E), j = 1..q$  for the training dataset. The classifier can be summarized as in (3.2)

$$H_{BR} = \{C_{y_i}((x, y_j)) \rightarrow y_j' \in \{0, 1\} | y_j \in L : j = 1..q\}$$

Table 2 depicts all the possible four binary tables of the resultant transformation of the labels present in Table 1. For any  $i$ th label,  $y_i$  signifies that the label is present. Else if not present, it is denoted by  $\neg y_i$ .

- (ii) Label powerset :- Label powerset (LP) is a kind of problem transformation approach which considers each and every unique set of the labels existing in the training set as a single class. Label Powerset is a problem transformation approach to multi-label classification that transforms a multi-label problem to a multi-class problem with 1 multi-class classifier trained on all unique label combinations found in the training data. The method maps each combination to a unique combination id number, and performs multi-class classification using the classifier as multi-class classifier and combination ids as classes.

This single class is one of the many classes that are possible from the training set itself.

For an unknown example, LP would result in outputting the most probable class as a result, which is a set of the classes from the original dataset. The benefit of LP is that LP takes into consideration the correlation between different labels, which is otherwise ignored in BR. However, as the dataset increases, so do the complexity of the classes that are associated for even a few examples [12, 30].

**Table 2** An example of binary relevance

Y	$\neg y_1$	Y	$\neg y_2$	Y	$y_3$	Y	$\neg y_4$
$E_1$	$\neg y_1$	$E_1$	$\neg y_2$	$E_1$	$y_3$	$E_1$	$\neg y_4$
$E_2$	$\neg y_1$	$E_2$	$\neg y_2$	$E_2$	$y_3$	$E_2$	$y_4$
$E_3$	$\neg y_1$	$E_3$	$\neg y_2$	$E_3$	$\neg y_3$	$E_3$	$y_4$
$E_4$	$\neg y_1$	$E_4$	$\neg y_2$	$E_4$	$y_3$	$E_4$	$\neg y_4$

Table 3 depicts the notion behind LP. First sub-table shows the set of instances in the original dataset, while the second table shows the same set of instances after applying LP, where notation  $y_{i,j}, \dots, k$  shows the conjunction between the different labels  $y_i \cap y_j \cap y_k$ . While the first sub table has a set of  $\{y_1, y_2, y_3, y_4\}$  after LP, the set of multiclass labels are from a set of  $\{y_{2,3}, y_{1,3,4}, y_4, y_{2,3}\}$ .

### 3.3 Text vectorizers

Machine learning algorithms function on numeric data only. Hence, we need to convert our documents into vector representations so that we can apply numeric machine learning and perform significant analytic; for that, we use Text encoding, which is a process wherein the text data is converted into unique numeric vectors that can be utilized by ML models. From a wide variety of text vectorizer models, we have used a rather very popular representation method from all the options available, Bag-of-word model [18] that has been extensively used in many kinds of literature for the purpose of object categorization. The main idea is to quantify each key point extracted into one of the visual words and then represent each image with a visual word histogram.

- (i) Count Vectorizer:- 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. Count vectorizer is almost similar to One Hot encoding. The only difference is instead of checking whether the particular word is present or not, it will count the words that are present in the document [15]
- (ii) TF-IDF :- Term frequency-inverse document frequency is numerical statistics that demonstrate how important a word is to a document in a collection or corpus [33]. It is generally utilized as a weighted factor in the search for information retrieval.

TF: It count the occurrences of each term in each document.  $tf(t,d)$ , the number of times that term  $t$  occurs in document  $d$ .

$$tf(t, d) = \frac{f_d(t)}{\max_{f_d(w) : w \in d}$$

IDF: Quantifies how much information is being imparted by word processed, depending on its occurrence signifying how rare or common the word is. Total number of documents in the corpus  $N = |D|$

$$idf(t, D) = \ln\left(\frac{|D|}{|d \in D : t \in d|}\right)$$

**Table 3** An example of label powerset

Y		Y	
E <sub>1</sub>	Y <sub>1</sub> = {y <sub>2</sub> ,y <sub>3</sub> }	E <sub>1</sub>	y <sub>2,3</sub>
E <sub>2</sub>	Y <sub>2</sub> = {y <sub>1</sub> ,y <sub>3</sub> ,y <sub>4</sub> }	E <sub>2</sub>	y <sub>1,3,4</sub>
E <sub>3</sub>	Y <sub>3</sub> = {y <sub>4</sub> }	E <sub>3</sub>	y <sub>4</sub>
E <sub>4</sub>	Y <sub>4</sub> = {y <sub>2</sub> ,y <sub>3</sub> }	E <sub>4</sub>	y <sub>2,3</sub>

then TF-IDF is calculated as

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D)$$

### 3.4 Machine learning classifiers

The process used for predicting the class belonging to the data points being considered is known as classification. Classes are generally termed as targets or labels or categories. Classification predictive modeling approximates the task of assigning the result of a mapping function ( $f$ ) from a range of input variables ( $x$ ) to discrete output variables ( $y$ ). From a wide variety of classifiers available in a text classification problem, some of them used are:

- (i) **Logistics Regression(LR)** : It is often used when the output of the data is binary, which is referred to as one class or another, or either a 0 or 1 [7]. It is a classification algorithm used when the value of the labeled variable is categorical in nature. It utilizes the sigmoid function for classification. Logistics regression does not require input features to be scaled, and it does not require any tuning. It generates well-calibrated predicted probabilities and acts as a baseline algorithm to compare the performance of other algorithms. However, it requires a large sample size for stable results; otherwise leads to overfitting.
- (ii) **Multinomial Naive Bayes (MNB)** : It is a learning algorithm that is frequently employed to tackle text classification problems. Normally, multinomial naive Bayes is computationally very efficient and easy to implement. Also, it has better speed and accuracy for the large dataset. However, it does always provide accurate results in some cases where there is inter-dependency among variables. The multinomial event model frequently referred to as multinomial naive Bayes or MNB for short—generally outperforms the multivariate one. Let the set of classes be denoted by  $C$ . Let the vocabulary size be  $N$  [2, 29]. Then the testing document is assigned by MNB to the class with the highest probability outcome  $Pr(c/t_i)$ , which utilizes Bayesian rule, is given by:

$$Pr(c|t_i) = \frac{Pr(c)Pr(t_i|c)}{Pr(t_i)}, c \in C$$

- (iii) **K-Nearest Neighbor (KNN)**: To classify a new document, KNN finds the k-nearest neighbors among the training documents and uses the categories of the k-nearest neighbors to label the new document. It is a case-based and non-parametric learning classification method. In general, KNN requires no training before making predictions. Also, new data can be added continuously without impacting the overall accuracy. However, it requires huge computations in the large dataset that leads to degradation in performance. This method keeps all the training instances for classification. To a large extent, its performance heavily depends upon two factors, a suitable similarity function and a proper value for parameter k [20]. The distance functions used in KNN are:

$$Euclidean = \sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

$$Manhattan = \sum_{i=1}^k |x_i - y_i|$$

$$Manhattan = \left( \sum_{i=1}^k (|x_i - y_i|^q) \right)^{\frac{1}{q}}$$



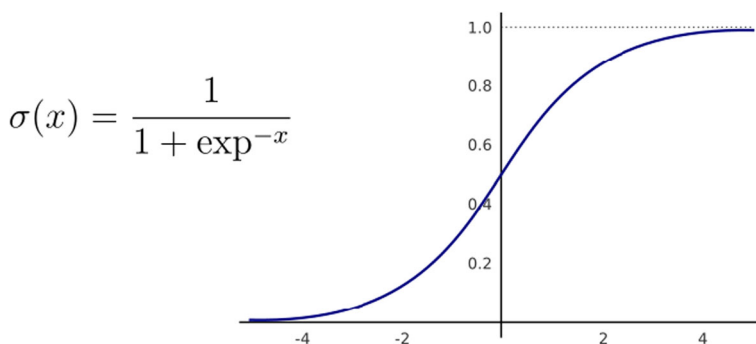
- (iv) **Support Vector Classifier (SVC):** It is used for the classification of input data that are supplied and computed in batch. Usually, SVC provides better accuracy than the other classifiers with less chances of overfitting. However, it is computationally very expensive that leads to an increase in training time. Also, It performs well for binary classification problems, and in the case of multi-class problems, it breaks down the problem into pairs of classes (Fig. 1). The main aim of a Linear SVC (Support Vector Classifier) is to fit the provided data and return a “best fit” hyperplane that divides or categorizes the data. From there, after getting the hyperplane, then some features are fed into the classifier to see what the “predicted” class is [22]. Figure 2 depicts SVC classifier with hyperplanes and support vectors. In addition to a single class SVC, there is also a multi-class SVM algorithm in the literature that facilitates active learning [8].

## 4 Proposed methodology

This section describes the working of the proposed movie genre classification model which utilizes the problem transformation approaches, count vectorizers and various machine learning classifiers. The model is supplemented by appropriate components such as text pre-processing, custom prediction functions, evaluation, and so on. The prediction method used varies depending on which transformation method we are using. Binary Relevance utilizes a Series of probability threshold values relating to each genre, depending on its occurrence in the original dataset. On the other hand, Label Powerset undergoes a dimensional reduction process through K-Means and Principal Component Analysis (PCA) to reduce the complexity of the number of classes being formed. In both cases, the dataset is split for validation, and final testing is done on a pre-processed test dataset. The best combination of encoder and classifier evaluates the final test document on the basis of different performance metrics, helping us choose the best one. We present below the different parts of our proposed model.

### 4.1 Multi-label classification using binary relevance

This approach tends to serve as the simplest method of the bunch by treating each genre as its own separate classes, which can then be predicted using single class classification



**Fig. 1** Sigmoid function curve for different values of  $x$

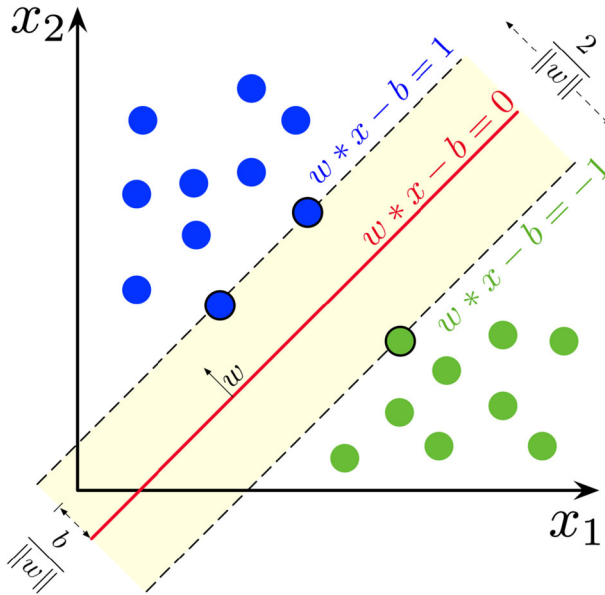


Fig. 2 Representation of SVC classifier with hyperplanes and support vectors

techniques. The outline of the proposed algorithm is presented in Algorithm 1. For each classifier, the class is fitted against all the other classes. Hence,  $n\_classes$  classifiers are needed. Finally, the union of all predicted classes serves as the output of the multi-labeling process. We use the inbuilt learn One Vs. rest classifier function to achieve multi-label classification [25]. The probability (equal to the frequency of the genre’s occurrence) threshold is used for classifying each genre. In the circumstance that none of the  $n\_class$  classifiers detect a genre, we pick one most likely genre based on the probability value.

---

**Algorithm 1** Proposed binary relevance algorithm.

---

**Input:**  $H$ : Best Classifier,  $P$ : Probability Threshold,  $testing\_labels$

**Output:**  $PredictedLabels$

- 1:  $categoryColumns \leftarrow$  All unique categories of testing labels
  - 2:  $PredictedLabels \leftarrow$  Dataframe with columns equal to category Columns
  - 3:  $prob \leftarrow H.predictProba(testing\_labels)$
  - 4: **for**  $index, col \leftarrow enumerate$  to  $categoryColumns$  **do**
  - 5:      $PredictedLabels[col] \leftarrow prob[:, index] > P[index]$
  - 6: **end for**
  - 7: **return**  $PredictedLabels$
- 

#### 4.2 Multi-label classification using label powerset

The multi-label classification using label powerset does not take into account the partial correlations between genres. Instead, we treat every possible genre combination that can be found in the training data as a new class. Thus, in the worst case, there can be a  $2^{n\_genres}$  number of classes. 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 the worst case of  $2n_{\text{genres}}$  number of classes. The overall structure of the proposed algorithm is summarized in Algorithm 2.

---

**Algorithm 2** Proposed label powerset algorithm.

---

**Input:**  $H$ : Best Classifier,  $\text{Class2GenreMap}$ ,  $\text{testing\_labels}$

**Output:**  $\text{PredictedLabels}$

```

1: if condition1 then
2:    $\text{PredictedLabels} \leftarrow \text{DataFrame}(\text{index} \leftarrow \text{XTest.index}, \text{columns} \leftarrow \text{Class2GenreMap})$ 
3: else
4:    $\text{PredictedLabels} \leftarrow \text{DataFrame}(\text{index} \leftarrow \text{range}(\text{Xtest.Shape}[0], \text{columns} \leftarrow \text{Class2Genre}))$ 
5: end if
6:  $\text{numClasses} \leftarrow \text{Class2GenreMap.shape}[0]$ 
7:  $\text{yClass} \leftarrow \text{Classifier.predictProba}(\text{XTest})$ 
8: for  $\text{index} \leftarrow 0$  to  $\text{numClasses}$  do
9:   if condition1 then
10:     $\text{yPred.loc}[\text{yClass} == \text{idx}, :] \leftarrow \text{Class2GenreMap.loc}[\text{idx}, :].\text{values}$ 
11:   end if
12: end for
13: return  $\text{PredictedLabels}$ 

```

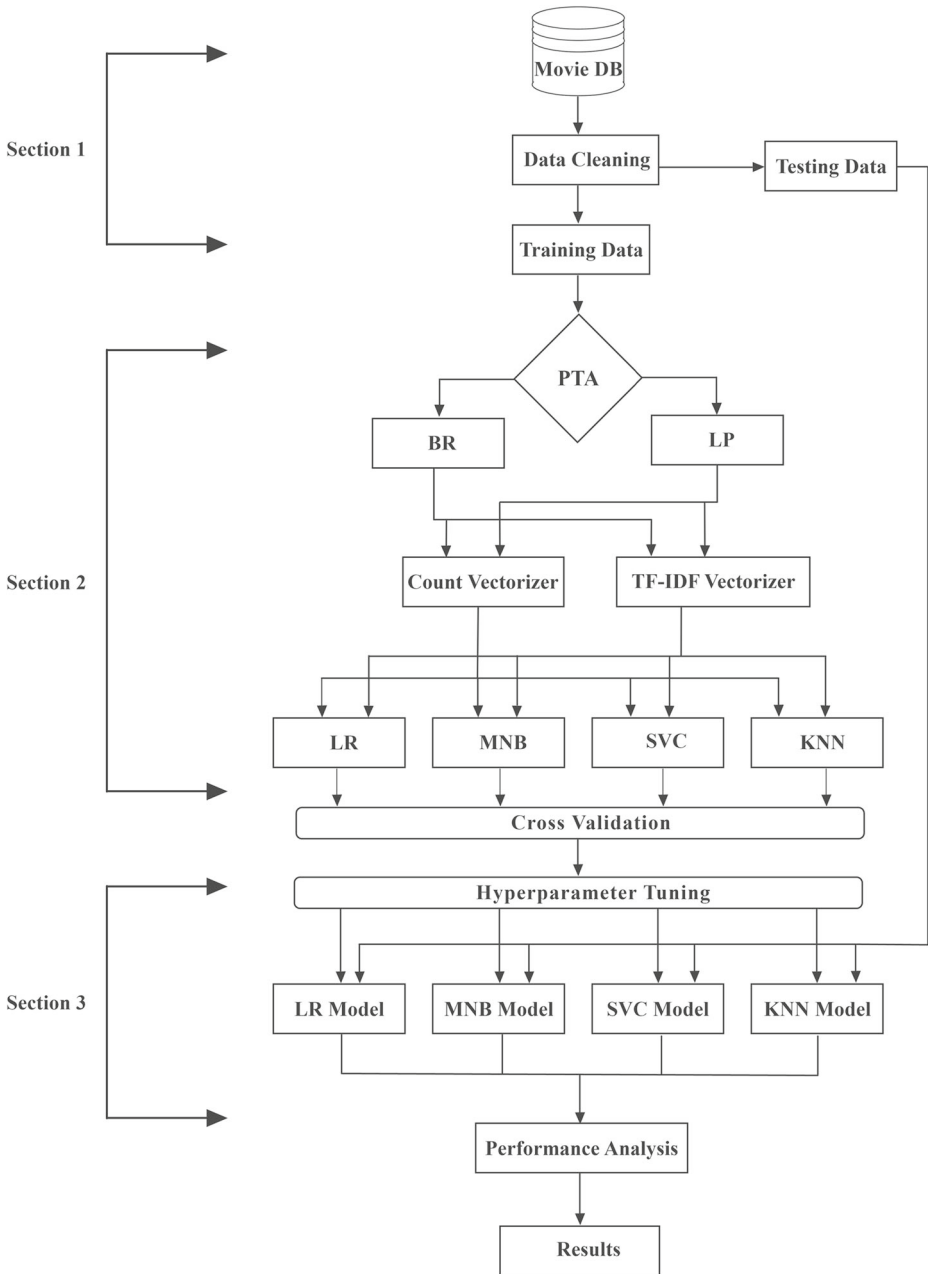
---

Transform ( $n_{\text{rows}} \times n_{\text{genres}}$ ) binary matrix from the training label set into  $n_{\text{rows}} \times 1$  label vector, where the column vector ranges from 0 to  $\text{num\_genre\_combinations} =$  a 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 that transformed  $n_{\text{rows}} \times 1$  column vector. Predict the test data set using the aforementioned fitted classifier. The output would be a column vector with each value ranging from 0 to  $\text{num\_genre\_combinations}$ . Transform this column vector back to individual genres using the inverse mapping that was used in the first step. Then we obtain the accuracy of the inverse transformed binary predicted genre matrix.

Figure 3 depicts the complete model overview of the proposed work, which can be broadly divided into three major sections. A detailed description of each section is presented below.

**Section 1:** First, we are dealing with the raw data collected from the Internet Movie Database (IMDB). The database consists of numerous categories, such as movie titles, directors, plots, casts, etc. From all these, the categories we have selected for the model are movie-plot, genres, and languages. The movie plot is selected as the main feature or independent variable for the model; movie genres will act as the labels or the dependent variables, whereas language is used in the pre-processing section. Next pre-processing has been applied to the raw dataset obtained. Steps is further divided as:-

1. Check for missing data fields: The used dataset has no missing values for any features.
2. Removal of non-English data: using the plot-language column, we removed all the data points based on the different languages.



**Fig. 3** Proposed model overview

3. **Feature Engineering:** We've dropped title from the dataset because our research work is focused on classifying genres using plot or synopsis, but we can possibly be used to enhance the classification. Also, removed the plot-language column as it has no role in genre prediction.

#### 4. Data Cleaning:

- i Removing HTML tags:  $htmlTag = '< .*? >'$
- ii Removing English stop words using nltk package [23]
- iii Lemmatizing words: We remove word affixes to get to the base form of a word
- iv Lower casing all the words.

#### 5. Finally, binarizing the dataset on the basis of all the genres found.

With the above section, the data pre-processing part has been completed.

**Section 2:** This section is solely based on the problem transformation approach (PTA) [Section 3.2]. We use two approaches of PTA, namely, BR [Section 3.2] and LP [Section 3.2]. For the implementation of these two, various combinations of text-vectorizers and ML classifiers have been formed. As there are 2 different vectorizers used in the model, i.e. CV [Section 3.3] and TF-IDF [Section 3.3], and 4 different classifiers, i.e. LR [Section 3.4], MNB [Section 3.4], KNN [Section 3.4], Linear SVC [Section 2]; in total 16 combinations are formed to be validated, tested and evaluated. Below is a brief description of two approaches of PTA, namely, binary relevance and label powerset utilized by us.

1. Binary Relevance - Under binary relevance, first, a custom function is applied to find the probability threshold for the occurrences of each of the genres as follows:

$$Prob\_thresh = \frac{\text{Total positive occurrences of genre}}{\text{Total instances}}$$

If the value of *Prob\_thresh* comes to be greater than 0.5, then we take *Prob\_thresh* as 0.5; otherwise, its value is taken as it is.

Now, the main algorithm for BR is implemented as described in Section 4.1. Finally, the output is predicted using the help of a custom prediction function that compares the outcome of the *Prob\_thresh* with an inbuilt model function *predict\_proba*, which gives the probability of the occurrence of 1 as the output. The pseudo algorithm is as follows:-

- i `prob = classifier.predict_proba(test plot)`
- ii `predictions = (prob > prob_thresh)`

If `prob > threshold` of that genre, then the value of the prediction will be true otherwise false. Predictions represent the final output respective to each of the genre.

2. Label Powerset – The remaining combinations are applied under LP. In this problem transformation approach, as mentioned above in Section 4.2, the combinations of all the genres are taken as separate classes, but large set genres lead to an unfeasibly large number of combination possibilities. For example, if there are  $n$  genres, then the total possible combinations will be  $2^n$ . Hence, to reduce the complexity, we have used clustering, and for that, a custom function is being used:-

- i Converting genre matrix(*train.y*) into label vectors (*y\_cluster\_labels*). For the mentioned conversion, we have used sklearn's pandas library to vectorize the group of genres:-  
`y_cluster_labels = pandas.groupby(list of all genres).ngroup()`
- ii Making *class\_to\_genre\_map* to retrace the original genres from the label vectors.

**Table 4** Different tested values of max\_df, min\_df and n\_gram for CV and TF-IDF vectorizers

S.No	vectorizer	min_df	max_df	n_grams
1	cv	1,2	0.25,0.5	(1,1),(1,2)
2	tf-idf	1,2,3,5	0.25,0.5,0.75	(1,1),(1,2)

- iii Training of the model is done using plot-summary as the independent variable and y\_cluster\_labels as the dependent variable
- iv After the predictions, the outcome comes out to be in the form of vectors, which is further converted into their corresponding genre groups using class.to\_genre\_map.

3. To get the best outcomes from the above PTAs, hyper-parameter tuning of text-vectorizers and classifiers plays an extremely important role. The parameter values completely depend upon the dataset used, also the type of text vectorizers and classifiers used. For our dataset, the parameters used are min\_df, max\_df, n\_gram [33].

**Section 3:** Now, after the validation, hyper-parameter tuning is performed on each of the combinations we made in Section 1. This step involves the tuning of the internal parameters of text-vectorizers as well as classifiers. Table 4 shows the values for min\_df, max\_df and n\_grams for the two different vectorizers CV and TF-IDF. Table 5 shows the parameters of class\_weight and inverse of regularisation strength denoted by C. Lastly, Table 6 shows the different parameter values for alpha, fit\_prior, class\_prior, n\_neighbours and weights for different classifiers.

## 5 Datasets and evaluation criteria

For the purpose of evaluating our proposed method, we choose the IMDb dataset [17], which is publicly available. The dataset consists of almost every major movie released between 2000 and 2012. Since not all major movies are in English, we apply appropriate pre-processing to get a better selective dataset for better predictions. The IMDb dataset consists of various features like ID, title, directors, language, plot description, genres, revenue, etc. Among all these variety of features, the required features are plot description, genres and language. Textual plot or overview is the independent variable whereas genres are the dependent variables. Language feature is used to remove all the non English instances to make it easier for the model to be trained. Preprocessing further helps us extract only the useful features from the dataset itself by removing punctuation, tags, numbers, and stopwords. We carefully select a set of unique genres suitable for proper analysis as many different genres somehow share the same base genre. For example, Ghost, Monster, Vampire, Zombie,

**Table 5** Various tested values of C and class\_weights for LR and SVC classifiers

S.No	Classifier	C	class_weight
1	LR	0.1,1,10,100	—
2	SVC	5,10,20,50,100	balanced

**Table 6** Various tested values of alpha, n\_neighbor, fit\_prior, weight for MNB and KNN classifiers

S.No	Classifier	Alpha	fit_prior	class_prior	n_neighbor	Weight
3	MNB	0.001,0.01,0.1,1	true	none	–	–
4	KNN	–	–	–	5,6,10,20,30	balanced,distance

Slasher genres share the same parent genre, i.e., Horror. These selected genres include the parent genres, which reduce the training efforts and, as the classification is now limited to the most important base genres, helps in improving the model's performance. The resultant dataset contains total 117194 instances and 27 unique genres for our study. Table 7 lists the name of 27 unique genres available and Fig. 4 depicts the number of movies for each genre in the extracted datasets.

### 5.1 Evaluation criteria

Several methods for evaluation for multi-label text classifications technique have been put forward. Using only a single evaluation metric may give false insights into the result, thereby giving incorrect interpretations of the metrics. We have also averaged the traditionally calculated metrics such as Accuracy, Precision, Recall, and F1-score by averaging the metrics across all genres. For this purpose, we assume  $K_i$  to be a support value that denotes the number of movies containing the  $i^{th}$  unique genre present in the dataset. Since our treated IMDb dataset contains 27 unique genres, those 27 genres are distributed for values of  $i$  starting from 1 and ending till 27, covering all the unique genres present. To compare actual labels with the predicted labels for every test example, the result can then be averaged across all examples on the test set. The traditional metrics for Accuracy, Precision, Recall and F1 Score for each  $i^{th}$  genre in the dataset are denoted by the  $A_i$ ,  $P_i$ ,  $R_i$  and  $F1_i$  respectively. For the multi-label text classification, the adapted metrics are denoted by  $A_{avg}$ ,  $P_{avg}$ ,  $R_{avg}$ ,  $F1_{avg}$  calculated by taking the sum of product of the traditional metric with its support value  $K_i$  for every  $i^{th}$  genre. This summation is then divided by the sum of all support values  $K_i$  taken together. Upon following this procedure for every traditional metric, we obtain the following adapted four evaluation metrics as follows:

**Table 7** List of 27 unique major genres used for classification

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

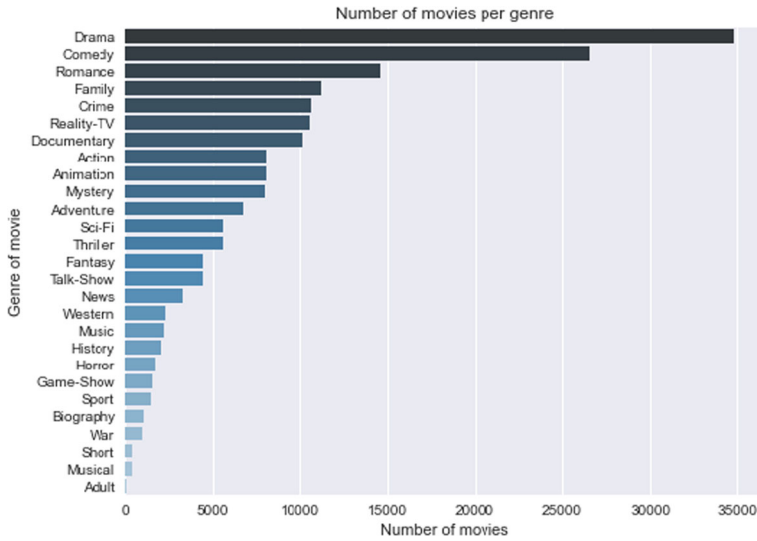


Fig. 4 Number of movies per genres in IMDb dataset

*Accuracy<sub>avg</sub>* : Accuracy for every instance is the ratio of predicted output, which is correct, to the total number of all possible outputs for that particular instance.

$$Accuracy_{avg} = \frac{\sum_{i=1}^n (A_i * K_i)}{\sum_{i=1}^n K_i}$$

*Precision<sub>avg</sub>* : It is the ratio of predicted correct labels to the total number of actual labels, averaged across all instances

$$Precision_{avg} = \frac{\sum_{i=1}^n (P_i * K_i)}{\sum_{i=1}^n K_i}$$

*Recall<sub>avg</sub>* : A recall is the ratio of correct labels to the total number of predicted labels, averaged across all instances.

$$Recall_{avg} = \frac{\sum_{i=1}^n (R_i * K_i)}{\sum_{i=1}^n K_i}$$

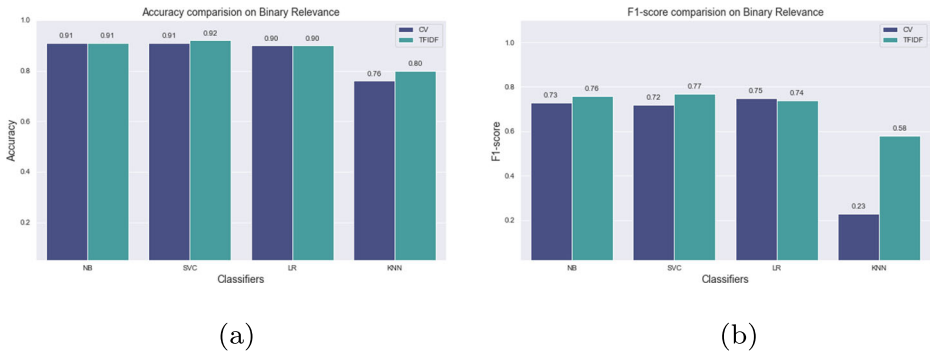
*F1 – Score<sub>avg</sub>* : It is the harmonic mean of *Precision<sub>avg</sub>* and *Recall<sub>avg</sub>*.

$$F1score_{avg} = \frac{\sum_{i=1}^n (F1_i * K_i)}{\sum_{i=1}^n K_i}$$

## 6 Experimental results and analysis

In this section, we present the detailed experimental results and analysis of the proposed model and some recently introduced techniques of movie genre classification. We performed the experiment using the dataset and obtained the value of different evaluation metrics as discussed in Section 5. We utilized the k-fold cross-validation technique to evaluate proposed models and other movie genre classification models on the various considered





**Fig. 5** Accuracy and F1-Score result on the IMDb dataset obtained by binary relevance based method in combination with text vectorizer, namely, count vectorizer (CV) and TF-IDF, and machine learning classifier technique, namely, LR, NB, SVM, and KNN (a) Accuracy (b) F1-score

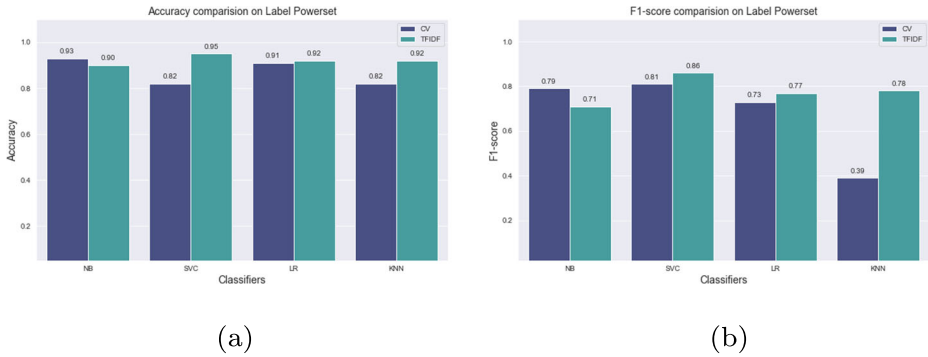
datasets. Here, the value of  $k$  was considered as 5 for performing the evaluation. The experiment is performed on a personal computer with a Windows 10 operating system, an Intel i7-9750H processor with 12 CPUs running at 2.6GHz, 8GB RAM, 1 TB hard disk space. We conducted the simulations in python programming language by utilizing libraries like Scikit-learn, Numpy, Pandas, and NLTK.

### 6.1 F1-score and accuracy results for binary relevance based method

Figure 5 displays the F1-score and accuracy results on the IMDb dataset obtained by the binary relevance based method in combination with text vectorizer, namely, count vectorizer (CV) and TF-IDF, and machine learning classifiers, namely, LR, NB, SVM, and KNN. The light blue bar represents CV, and dark blue represents TF-IDF. In terms of text vectorizers, we notice that TF-IDF performed slightly better than count vectorizer (CV) for the binary relevance method with all the machine learning classifiers in the case of accuracy. Another prominent observation is that KNN performed at the least caliber among all the classifiers. It is observed that the support vector classifier (SVC) performed best among all classifiers based on accuracy. However, logistic regression and Naive Bayes classifiers also performed well. Now, in the case of the F1-score, the trend is similar to that of accuracy, where SVC as the machine learning classifier and TF-IDF as text vectorizer performed best. Finally, we can conclude that binary relevance with TF-IDF and SVC model produced the best accuracy as 0.92 and F1-score as 0.77.

### 6.2 F1-score and accuracy results for label powerset based method

Figure 6 displays the F1-score and accuracy results on the IMDb dataset obtained by label powerset based method in combination with text vectorizers, namely, count vectorizer (CV) and TF-IDF, and machine learning classifier model, namely, LR, NB, SVM, and KNN. The obtained results reveal a similar trend of accuracy as it is when we used Binary Relevance. Label Powerset also sees KNN struggling to perform well. However, unlike Binary Relevance, KNN performs close to the better models, often matching their accuracy. From the result, we can see that due to underperforming in other more overlapping genres, we can rule out KNN from being the best of the bunch. We again see TF-IDF performing better than CV when averaged across all the genres. The average accuracy calculated reveals that

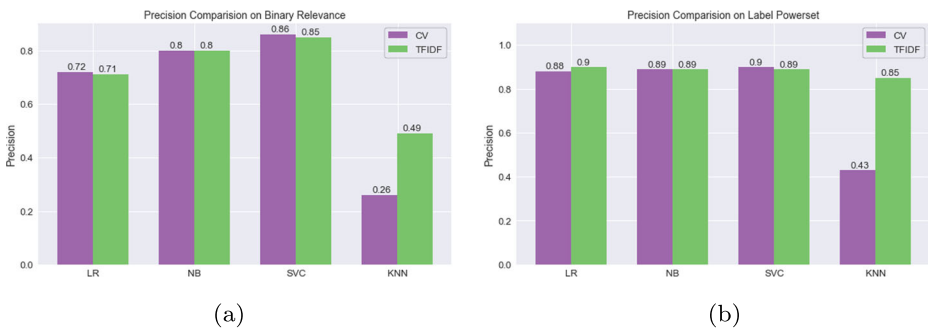


**Fig. 6** Accuracy and F1-score result on the IMDb dataset obtained by label powerset based method in combination with text vectorizer, namely, count vectorizer (CV) and TF-IDF, and machine learning classifier, namely, LR, NB, SVM, and KNN (a) Accuracy result, (b) F1-score result

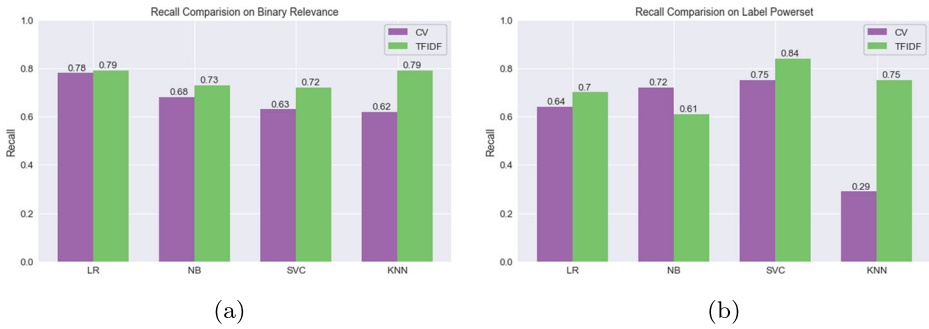
SVC is the best classifier model here. The performance of SVC is best and most stable among both problem transformation approaches, text vectorizers, and classification models used. Finally, we can conclude that label powerset with TF-IDF and SVC model produced the best accuracy as 0.95 and F1-score as 0.86.

### 6.3 Precision

Precision is the fraction of relevant instances among the retrieved instances. Figure 7a depicts the average precision obtained across all the genres on the dataset using binary relevance as the problem transformation approach (PTA). Similarly, Fig. 7b shows the average precision obtained using label powerset (LP). The performance of KNN is most affected by the change of PTA and the Vectorizer being used, where it can be seen performing as well as the other combinations when using LP and TF-IDF. However, in terms of the best from the bunch, SVC still shows a better score than the rest.



**Fig. 7** Precision result based on different classifier-vectorizer and PTA combinations (a) Precision obtained using Binary Relevance based method, (b) Precision obtained using Label Powersetbased method



**Fig. 8** Recall result based on different classifier-vectorizer and PTA combinations (a) Recall obtained using Binary Relevance based method, (b) Recall obtained using Label Powerset based method

## 6.4 Recall

Figure 8 presents the average recall value obtained across all the genres of the IMDb dataset. Figure 8a depicts the average recall obtained across all the genres on the dataset using binary relevance as the PTA. Similarly, Fig. 8b shows the average recall obtained using label powerset (LP). In the case of binary relevance (BR), all the classifiers show better recall values when using TF-IDF instead of count vectorizer (CV). In case of label powerset, there is the same trend, except for Naive Bayes (NB), where it is shown to favor count vectorizer (CV) more. In terms of the best results using the binary relevance based method, logistics regression (LR) is easily seen to perform the best, where even using CV as the vectorizer has a better recall than recall of runner-up combination using TF-IDF. Similar to the previous patterns, the best result of recall is achieved by combining label powerset (LP) as problem transformation approach, TF-IDF as the text vectorizer, and SVC as the machine learning classifier.

## 6.5 Accuracy, precision, recall and F1-score results of all combinations of problem transformation approaches, text vectorizers, and machine learning classifiers

We present the combined results of our adapted combinations using problem transformation approaches, text vectorizers, and machine learning classifiers based on all four performance criteria, namely, accuracy, precision, recall, and F1-score. By combining problem transformation approaches, text vectorizers, and machine learning classifiers, we obtained 16 different combinations of models for the movie genre classification. Table 8 lists the results obtained by 16 different combinations of models for all the considered evaluation criteria. From the result, it is clear that the best result is obtained using the model consist of label powerset (LP) as problem transformation technique, TF-IDF as text vectorizer technique, and support vector classifier (SVC) as machine learning classifier producing an overall accuracy of 0.95 and F1- score of 0.86. The second best result achieves an accuracy of 0.93 and and F1- score of 0.79 by using the combination of label powerset (LP), count vectorizer (CV), and multinomial naive Bayes (MNB). The third best result is obtained by the combination of binary relevance, TF-IDF, and SVC with an accuracy of 0.92 and an F1-score of 0.77.

**Table 8** Summary of performances based on accuracy, precision, recall, and F1 score obtained by 16 different combinations of models by combining problem transformation approaches, text vectorizers, and machine learning classifier

PTA	Vectorizers	ML Classifier	Accuracy	Precision	Recall	F-1 Score
BR	CV	NB	0.91	0.80	0.68	0.73
BR	CV	SVC	0.91	0.86	0.63	0.72
BR	CV	LR	0.90	0.72	0.78	0.75
BR	CV	KNN	0.76	0.26	0.62	0.23
BR	TF-IDF	NB	0.91	0.80	0.73	0.76
<b>BR</b>	<b>TF-IDF</b>	<b>SVC</b>	<b>0.92</b>	0.85	0.72	<b>0.77</b>
BR	TF-IDF	LR	0.90	0.71	0.79	0.74
BR	TF-IDF	KNN	0.80	0.49	0.79	0.58
<b>LP</b>	<b>CV</b>	<b>NB</b>	<b>0.93</b>	0.89	0.72	<b>0.79</b>
LP	CV	SVC	0.82	0.90	0.75	0.39
LP	CV	LR	0.91	0.88	0.64	0.73
LP	CV	KNN	0.82	0.43	0.29	0.39
LP	TF-IDF	NB	0.90	0.89	0.61	0.71
<b>LP</b>	<b>TF-IDF</b>	<b>SVC</b>	<b>0.95</b>	0.89	0.84	<b>0.86</b>
LP	TF-IDF	LR	0.92	0.90	0.70	0.77
LP	TF-IDF	KNN	0.92	0.85	0.73	0.78

## 6.6 Result comparison with other popular methods

In Table 8, we have summarized the results obtained by 16 different combinations for classifying movies into appropriate genres by combining problem transformation approaches, text vectorizers, and machine learning classifier models. Among all the 16 combinations adopted by us, the best result is produced by the model consist of label powerset (LP) as problem transformation technique, TF-IDF as text vectorizer technique, and support vector classifier (SVC) as a machine learning model. We also performed the accuracy and F1-score results comparison of our best model, which is comprising of label powerset (LP), TF-IDF, and support vector classifier (SVC) with some recently proposed popular models of movie genre classification. Table 9 depicts the result obtained by our best model and other three popular models along with the dataset used. From the results, it is clear that our proposed model outperforms existing models, namely, multimodal approach [24], topological data analysis [10] and gated recurrent units (GRU) based method [14] where multimodal approach and gated recurrent units (GRU) method are based on neural networks, and topological data analysis is based on an approach to the study of movie genre datasets using techniques from topology. We also computed the total time required by the proposed, GRU-based, and Multimodal models on the IMDb and TMDb datasets for movie genre classification. The total time for the IMDb dataset required nearly 3 hours 14 minutes for our best performing model composed of Label Powerset, SVC, and TF-IDF. On the other hand, the GRU-based model required around 4 hours 48 minutes for the same model on our pre-configured systems. Hence, our approach improves the time odds by nearly 32.63%. In the case of the TMDb dataset, the Multimodal model required nearly 2 hours 38 minutes. On the other hand, our best model required around 1 hour 45 minutes, which improves the time

**Table 9** F1-score and accuracy result comparison of our best model comprising of LP, TF-IDF, and SVC with other three existing methods

Paper Title	Dataset source	Dataset type	Best metrics	Our Best metrics
Multimodal Approach [24]	TMDB	Synopsis	F1-Score 0.488 Accuracy 0.74	F1-Score 0.498 Accuracy 0.84
Multimodal Approach [24]	Subtitles.org	Subtitles	F1-Score 0.436 Accuracy 0.68	F1-Score 0.76 Accuracy 0.85
Topological data analysis (TDA) [10]	IMDB	Synopsis	F1-Score 0.71 Accuracy 0.83	F1-Score 0.86 Accuracy 0.95
Gated Recurrent Units (GRU) [14]	IMDB	Synopsis	F1-Score 0.56 Accuracy 0.82	F1-Score 0.86 Accuracy 0.95

by 33.54%. Hence, we can conclude that the proposed method performs better than neural network-based models and is also less computationally intensive.

## 7 Conclusion

This paper devised a generic framework to apply popular problem transformation techniques, text vectorizers, and machine learning classifier models on different movie datasets to perform movie genre classification based on textual information and extracting features from synopses and plots. In this study, we utilized two problem transformation techniques, namely binary relevance (BR) and label powerset (LP), two text vectorizers, namely, count vectorizer (CV) and TF-IDF, and four machine learning classifier models, namely, logistic regression (LR), multinomial naive Bayes (MNB), K-nearest neighbor (KNN), support vector classifier (SVC). By combining different problem transformation approaches, text vectorizers, and machine learning classifiers, we have obtained 16 different combinations for classifying movies into their appropriate genres. Finally, we conclude that the best result is obtained using the model consist of label powerset (LP) as problem transformation technique, TF-IDF as text vectorizer technique, and support vector classifier (SVC) as a machine learning model producing an overall accuracy of 0.95 and F1-score of 0.86 on IMDb dataset. The second best result achieves an accuracy of 0.93 and F1-score of 0.79 by using a combination of label powerset (LP), count vectorizer (CV), and multinomial naive Bayes (MNB). Our proposed method produces exemplary results as compared to some recent, including neural network based methods. The future work of the research can include exploring more combinations with universal sentence embedder as it takes raw input and will help in reducing a huge amount of preprocessing work which is usually required in text classification problems.

## Declarations

**Conflict of Interests** The authors declare that they have no conflict of interest. The authors did not receive support from any organization for the submitted work

## References

1. Berger MJ (2015) Large scale multi-label text classification with semantic word vectors. Technical report, Stanford University
2. Bhowmik A, Kumar S, Bhat N (2019) Eye disease prediction from optical coherence tomography images with transfer. Learning engineering applications of neural networks. EANN, Communications in Computer and Information
3. Bhowmik A, Kumar S, Bhat N (2021) Evolution of automatic visual description techniques-a methodological survey. *Multimedia Tools and Applications*, pp 1–45
4. Cai L, Song Y, Liu T, Zhang K (2020) A hybrid BERT model that incorporates label semantics via adjustable attention for Multi-Label text classification. *IEEE Access* 8:152183–92
5. Chu WT, Guo HJ (2017) Movie genre classification based on poster images with deep neural networks. In: proceedings of the workshop on multimodal understanding of social, affective and subjective attributes, pp 39–45
6. de Carvalho AC, Freitas AA (2009) A tutorial on multi-label classification techniques. *Found Comput Intell* 5:177–95
7. Divya R, Kumari RS (2021) Genetic algorithm with logistic regression feature selection for Alzheimer's disease classification. *Neural Computing and Applications*

8. Dong S (2021) Multi class SVM algorithm with active learning for network traffic classification. *Expert Syst Appl* 176:114885
9. Dong S, Wang P, Abbas K (2021) A survey on deep learning and its applications. *Comput Sci Rev* 40:100379
10. Doshi P, Zadrozny W (2018) Movie genre detection using topological data analysis. In: *International conference on statistical language and speech processing*. Springer, Cham, pp 117–128
11. Ertugrul AM, Karagoz P (2018) Movie genre classification from plot summaries using bidirectional lstm. In: *2018 IEEE 12th International conference on semantic computing (ICSC)*. IEEE, pp 248–251
12. Ganda D, Buch R (2018) A survey on multi label classification. *Recent Trends Program Lang* 5(1):19–23
13. Godbole S, Sarawagi S (2004) Discriminative methods for multi-labeled classification. In: *Pacific-Asia conference on knowledge discovery and data mining*. Springer, Berlin, pp 22–30
14. Hoang Q (2018) Predicting movie genres based on plot summaries. arXiv:1801.04813
15. [https://scikit-learn.org/stable/modules/generated/sklearn.feature\\_extraction.text.CountVectorizer.html](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html) [Accessed: 20-Aug-2020] (2020)
16. Huang Y, Chen J, Zheng S, Xue Y, Hu X (2021) Hierarchical multi-attention networks for document classification. *Int J Mach Learn Cybern* 12(6):1639–47
17. Imdb data <ftp://ftp.fu-berlin.de/pub/misc/movies/database/>
18. Jiang H, Xiao Y, Wang W (2020) Explaining a bag of words with hierarchical conceptual labels. *World Wide Web*, 1–21
19. Katyal S, Kumar S, Sakhuja R, Gupta S (2018) Object detection in foggy conditions by fusion of saliency map and YOLO. In: *2018 12th International conference on sensing technology (ICST)*, pp 154–159
20. Khurana G, Bawa NK (2021) Weed detection approach using feature extraction and KNN classification. In: *Advances in electromechanical technologies*. Springer, Singapore, pp 671–679
21. Kumar S, Kumar M (2019) Predicting customer churn using artificial neural network. In: *International conference on engineering applications of neural networks*, pp 299–306
22. Longato E, Acciaroli G, Facchinetti A, Maran A, Sparacino G (2020) Simple linear support vector machine classifier can distinguish impaired glucose tolerance versus type 2 diabetes using a reduced set of CGM-based glycemic variability indices. *J Diabetes Sci Technol* 14(2):297–302
23. Loper E, Bird S (2002) Nltk: the natural language toolkit. arXiv:0205028
24. Mangolin RB, Pereira RM, Britto AS, Silla CN, Feltrim VD, Bertolini D, Costa YM (2020) A multimodal approach for multi-label movie genre classification. *Multimed Tools Appl* 7:1–26
25. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J (2011) Scikit-learn: machine learning in python. *J Mach Learn Res* 12:2825–30
26. Pobar M, Ivasic-Kos M (2017) Multi-label poster classification into genres using different problem transformation methods. In: *International conference on computer analysis of images and patterns*, pp 367–378
27. Portolese G, Domingues MA, Feltrim VD (2019) Exploring textual features for multi-label classification of portuguese film synopses. In: *EPIA Conference on artificial intelligence*. Springer, Cham, pp 669–681
28. Saputra AC, Sitepu AB, Sigit PW, Tetuko PG, Nugroho GC (2019) The classification of the movie genre based on synopsis of the Indonesian film. In: *International conference of artificial intelligence and information technology (ICAIIIT)*, pp 201–204
29. Sinha A, Ganguly J (2021) Categorization of videos based on text using multinomial naïve bayes classifier. In: *Proceedings of international conference on frontiers in computing and systems*, pp 299–308
30. Spolaôr N, Cherman EA, Monard MC, Lee HD (2013) A comparison of multi-label feature selection methods using the problem transformation approach. *Electron Notes Theor Comput Sci* 292:135–51
31. Sun J, Zhu M, Jiang Y, Liu Y, Wu L (2021) Hierarchical attention model for personalized tag recommendation. *J Assoc Inf Sci Technol* 72:173–189
32. Taiwiah CA, Sheng V (2013) A study on multi-label classification. *Advances in data mining. Applications and theoretical aspects. Lect Notes Comput Sci* 7987:137–50
33. Ullman J (2011) *Mining of massive datasets*. Cambridge University Press
34. Wang T, Liu L, Liu N, Zhang H, Zhang L, Feng S (2020) A multi-label text classification method via dynamic semantic representation model and deep neural network. *Appl Intell* 50(8):2339–51
35. Wehrmann J, Lopes MA, Barros RC (2018) Self-attention for synopsis-based multi-label movie genre classification. In: *The 31th International FLAIRS conference*
36. Xia Y, Chen K, Yang Y (2021) Multi-label classification with weighted classifier selection and stacked ensemble. *Inf Sci* 557:421–42
37. Yang P, Sun X, Li W, Ma S, Wu W, Wang H (2018) SGM: sequence generation model for multi-label classification. arXiv:1806.04822

38. Yong ZJ, Hoo WL (2020) Movie genre filtering for automated parental control. In: International conference on intelligent robotics and applications. Springer, Cham, pp 244–253
39. Yu Y, Lu Z, Li Y, Liu D (2021) ASTS: attention based spatio-temporal sequential framework for movie trailer genre classification. *Multimed Tools Appl* 80(7):9749–64
40. Zhang ML, Li YK, Liu XY, Geng X (2018) Binary relevance for multi-label learning: an overview. *Front Comput Sci* 12(2):191–202

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

## Affiliations

Sanjay Kumar<sup>1</sup>  · Nikhil Kumar<sup>1</sup> · Aditya Dev<sup>1</sup> · Siraz Naorem<sup>1</sup>

Nikhil Kumar  
nikhilkumar\_2k17se70@dtu.ac.in

Aditya Dev  
adityadev\_2k17se10@dtu.ac.in

Siraz Naorem  
siraznaorem\_2k17se113@dtu.ac.in

<sup>1</sup> Department of Computer Science and Engineering, Delhi Technological University, New Delhi, 110042, India