A Review and Experiment on Feature Extraction Methods for Lyrics Based Music Genre Classification

Yun-Ting Lai
113423040
eleanor128@gmail.com

Wei-Siang Chen
113423052
nicolews.chen@gmail.com

ABSTRACT
This study explores the feasibility of using song lyrics alone for music genre classification. We applied four different feature extraction methods: Bag-of-Words, TF-IDF, Word2Vec, and Part-of-Speech (POS) with Named Entity Recognition (NER). To evaluate their effectiveness, we utilized two classification models: Random Forest and XGBoost. Additionally, we examined the impact of stop-word removal by conducting two experiments, one retaining stop words and another removing them. Exploratory Data Analysis (EDA) was performed as part of the data preprocessing process, including label encoding and data visualization, to gain deeper insights into the dataset. The results indicate that retaining stop words improves classification accuracy across different feature extraction methods and classification models.

Keywords
Music Genre Classification; Lyrics Feature Extraction; Classification; Lyrics Based Classification; Bag-of-words; TF-IDF; Word2Vec; Random Forest; XGBoost.

1. INTRODUCTION
1.1 The Meaning of Feature Extraction in Classification
Feature extraction is a vital step in the machine learning pipeline that helps improve the performance of classification models. It is a process used to extract the most distinct features present in a dataset, such as image, text or audio signals[1]. The extracted features serve as a compact and meaningful representation of the data, enabling more efficient and accurate classification models.

One of the primary benefits of feature extraction is to reduce the dimension. High-dimensional datasets often contain redundant or irrelevant information that can lead to increased computational costs and model overfitting. By doing feature extraction, we can reduce the dimension of the feature space, leads to a faster processing time[2], Moreover, keeping the most relevant features also helps improve the accuracy of the algorithm[3].

1.2 Why is It a Problem Worth Investigating?
Music recommendation systems are already a common feature in lots of areas, letting users find new music more easily. Most of these systems require automatic music analysis, such as classification according to genre, content, or artist[4]. By deepening our understanding of music classification, we can potentially enhance the accuracy and personalization of these recommendation systems, offering users more tailored music suggestions.

Unlike traditional text retrieval tasks, such as searching the web or office documents, analyzing song lyrics presents unique challenges due to the distinctive nature of lyrics themselves. Lyrics are not simply textual content; they are often imbued with abstract stylistic and linguistic dimensions. In addition to following certain structural patterns, they frequently include slang, figurative language, and rhythmic elements, which significantly complicate the analysis and processing of the text[5].

Moreover, song lyrics tend to use fewer words to convey ideas, similar to poetry. Lyrics also possess a more rhyming structure, often designed to fit the melody, making them even more distinct from traditional forms of text[5][6], in order to fit the melody. Given these considerable differences between traditional text and song lyrics, we aim to explore whether the process of feature extraction can remain the same as in traditional text classification, or whether it needs to be adapted to account for these specific characteristics.

1.3 Define the Goals of this Survey
Given the unique characteristics of song lyrics, the primary objective of this study is to explore whether lyrics alone can be used to classify music genres. The second objective is to evaluate the performance of different feature extraction methods. Lastly, this study aims to explore whether traditional text processing techniques, such as stop-word removal, can be effectively applied to lyric-based classification tasks.

To ensure a smooth execution of our experiments, we have followed these key steps:

a) Dataset Selection – Identifying a suitable dataset that contains song lyrics labeled with their respective genres.

b) Literature Review – Examining previous research to understand existing approaches and potential directions for lyric-based genre classification.

c) Feature Extraction – Implementing fundamental feature extraction techniques to represent lyrical content numerically.

d) Performance Evaluation – Comparing different feature extraction methods and classification models to assess their effectiveness in genre prediction.

This study aims to contribute to the field by systematically analyzing the impact of various text-based feature extraction methods on music genre classification. The insights gained from this research could provide a foundation for future studies that seek to enhance text-based music classification approaches.

2. RELATED WORK
2.1 Research Criteria
For this research survey, we primarily used Google Scholar and Web of Science to find relevant academic literature. These platforms provide access to high-quality research papers, ensuring the credibility of our sources.

To achieve our research objectives, we used key search terms such as "music classification based on lyrics", "lyrics feature extraction", "feature extraction classification", "music genre classification", "Bag-of-Words", "Word2Vec", and "TF-IDF". We also combined these keywords in various ways to expand our search results and

capture different perspectives on lyrics-based music classification and feature extraction.

After identifying relevant papers, we reviewed their references to discover additional studies which are closely related to our topic. To ensure research quality, we evaluated papers based on citation count and the reputation of the publishing journal, prioritizing widely cited works from well-established, peer-reviewed sources.

2.2 Why is Music Classification Based on Lyrics Important?

Music classification problems nowadays can be broadly categorized into two main areas: music emotion recognition (MER)[7], which focuses on associating music with specific emotional states, or music classification based on audio features[8], where aspects such as rhythm, tempo. While these approaches have been extensively studied, comparatively less research has focused on classifying music solely based on lyrics [9], [10].

Lyrics provide a unique perspective on music, as they often convey themes, emotions, and cultural elements that may not be fully captured through audio analysis alone. Furthermore, different genres exhibit distinct lyrical characteristics; for instance, rap lyrics tend to contain more slang and complex rhyme schemes, while country lyrics often emphasize storytelling and personal narratives. This raises an important question: can lyrics alone provide sufficient information for music genre classification?

Another challenge in genre classification is that the boundary of different genre is blurry[11]. While lyrics can effectively identify thematic categories such as “love songs” or “Christmas carols”[5], their capability to distinguish between broader genres—such as pop versus rock—remains uncertain.

Additionally, different music genres—such as rock, rap, and country—have their own distinctive styles and language features. These factors make song lyrics different from traditional text retrieval tasks and add complexity to their analysis. This opens up an intriguing area of research, as it invites further exploration into the unique characteristics of lyrics across different genres.

2.3 What are the key features of song lyrics?

Lyrics are not just text itself. Given its special characteristic, we can roughly divide lyrics into five dimensions [4]: vocabulary, style, semantics, orientation and song structure.

Vocabulary: This includes the words the lyrics used, the frequency of some words, usage of slang and even the vocabulary’s richness. In a study which aims to classify songs popular songs by ages, it found that the usage of swear words have been used ten times more frequently [12].

Style: This includes the arrangement of words, sentence complexity, or even rhythms. Different lyrics composer may have different styles. For analyzing, Part-of-Speech (POS) can be a useful tool to analyze the sentence structure and rhythmic pattern [4].

Semantics: This refers to how lyrics vary based on the topic or theme of a song. For example, love songs often contain more emotionally expressive words, while Christmas songs frequently include words related to holiday celebrations.

Orientation: M. Fell et al. [4] defines this aspect for how the song narrates. They analyze the ratio of past-tense verbs in the lyrics to outline the timeline of the narrative. A higher proportion of past-tense verbs may indicate that the song describes past events or memories, whereas a lower ratio suggests a focus on the present or future.

Song structure: Song structure is also a key characteristic that defines how lyrics are organized within a song. Some songs repeatedly emphasize a catchy section, where the lyrics also appear in recurring multi-line blocks.

2.4 What methods are useful in this topic?

2.4.1 Bag-of-words

Bag-of-Words is a widely used method in natural language processing. Instead of focusing on the meaning of the whole paragraph, it only considers the frequency of each word, ignoring word order and grammar. This method first constructs a dictionary of unique words from the given text and then represents each sentence as a vector based on word frequency. For example: given two sentences “I like dog” and “Dog don’t like cat”, the resulting dictionary will be ["I","like","dog","don’t","cat"]. Therefore, the two given sentences can be represented as a vector:

“I like dog” => [1,1,1,0,0]

“Dog don’t like cat” => [0,1,1,1,1]

This method is also used by many machine learning algorithms, such as Maximum Entropy, support vector machines, and Naive Bayes[13]. However, it does not capture semantic relationships between words; therefore, TF-IDF and word embeddings (e.g., Word2Vec, GloVe) are often used as improvements.

2.4.2 TF-IDF

TF-IDF is a more sophisticated method to weigh the importance of a word or term in the given text document. It is composed of TF (term frequency) and IDF (inverse document frequency).

TF (term frequency) means the frequency which a term appears in the given document, reflecting the importance of that term. The formula is given as follows:

$$TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ a\ document}$$

IDF (inverse document frequency) measures how important a word is in the entire corpus, or how many different documents the term appears in. The formula is given as follows:

$$IDF(t) = log(\frac{N}{df(t)})$$

Where N is the total number of documents in the corpus, and df(t) is the number of documents containing the term t.

Lastly, the entire formula is given as follows:

$$TF - IDF(t) = TF(t) \times IDF(t)$$

This weighting scheme is based on the assumption that terms are of importance when they occur more frequently in one document, and at the same time less frequently in the rest of the document collection[5].

2.4.3 Word embeddings

Though TF-IDF improves Bag-of-Words by considering the frequency of words in a document and their importance across a corpus, it still has some limitations. For example, it doesn’t capture the semantic relationships between words, meaning that it treats similar words as completely separate entities. For example, “dog” and “puppy” have similar meanings, but this method treats them as different words with no relation. That’s why we need word embeddings. Word embedding includes several methods, using dense vectors in a high-dimensional space to show the relationships between similar words.

Here, we are going to go through one of the words embedding methods:

2.4.4 Word2Vec

Word2Vec is a double-layer neural network which captures information about the meaning of the word based on the surrounding words. The major concept is that a word's meaning is defined by its neighbors, with the more similar meaning or usage, the value of the vector will be closer. Therefore, it took a text corpus as a input, and generates a set of feature vectors for words in that corpus as the output[14]. It first turns each word from the input into a vector, then uses its neighbor to learn word representations. The main training model of Word2Vec includes two options:

a) Continuous bag of words

b) Skip-gram

For example, given a sentence: "Cat is a cute animal" and selecting "cute" as our target word. Continuous bag of words is to use the context to predict a target word, in this case, ['cat', 'is', 'a', 'animal'] are all the neighboring words of the target word, so we can use these words to predict the word "cute".

On the other hand, skip-gram does the opposite job. It is to use words to predict a target context. That is, using "cute" to predict ['cat', 'is', 'a', 'animal'].

2.4.5 Part-of-speech (POS)

Part-of-speech (POS) tagging is an important task in Natural Language Processing (NLP). It gives words in a sentence a specific grammatical tag like nouns, pronouns, and relative pronouns, helping with tasks like sentence structure analysis and sentiment detection. Figure 1 illustrates the list of tagger, which is from the Penn Treebank tag set [15]. Although there are several methods to predict part-of-speech (POS) tags, in this paper, we are going to use the CoreNLP package, which was developed by Stanford, to predict the POS tags in lyrics. The POS tagging method used in the package is the maximum entropy.

The maximum entropy model is a probabilistic model that assigns probabilities to sequences based on the principle of maximizing entropy subject to certain constraints represented by data features. The maximum entropy model operates under the assumption that, among all possible distributions that are consistent with the observed data, the one that maximizes entropy should be chosen to avoid making unwarranted assumptions about the unobserved data. In the maximum entropy model published in 2003, the authors found that the surrounding words and their tags have direct help to give a tag for the current word [16], [17].

CC	Coordinating conjunction	TO	to
CD	Cardinal number	UH	Interjection
DT	Determiner	VB	Verb, base form
EX	Existential there	VBD	Verb, past tense
FW	Foreign word	VBG	Verb,gerund/present participle
IN	Preposition/subord.conjunjunction	VBN	Verb, past participle
JJ	Adjective	VBP	Verb, non-3rd ps. sing. present
JJR	Adjective, comparative	VBZ	Verb,3rd ps.sing. present
JJS	Adjective, superlative	WDT	wh-determiner
LS	List item marker	WP	wh-pronoun
MD	Modal	WP$	Possessive wh-pronoun
NN	Noun, singular or or mass	WRB	wh-adverb
NNS	Noun, plural	#	Pound sign
NNP	Proper noun, singular	$	Dollar sign
NNPS	Proper noun, plural	.	Sentence-final punctuation
PDT	Predeterminer	,	Comma
POS	POS Possessive ending	:	Colon, semi-colon
PRP	Personal pronoun	(	Left bracket character
PP$	Posssessive pronoun	)	Right bracket character
RB	Adverb	"	Straight double quote
RBR	Adverb, comparative	‘	Left open single quote
RBS	Adverb, superlative	“	Left open double quote
BP	Particle	’	Right open single quote
SYM	Symbol (mathematical or scientific)	”	Right open double quote

Figure 1. The Penn Treebank POS Tagset

2.4.6 Named Entity Recognition (NER)

In Natural Language Processing (NLP), Named-entity-recognition (NER) involves identifying and classifying named entities within text into predefined categories such as persons, organizations, locations, dates, etc. Named Entity Recognition (NER) most commonly uses the BIO tagging scheme, where B indicates the beginning of an entity, I represents the inside of an entity, and O denotes words that do not belong to any entity [18]. For example, the sentence: *"Apple was founded by Steve Jobs."* After NER tagging, it becomes: *"Apple/B-ORG was/O founded/O by/O Steve/B-PER Jobs/I-PER."* Figures 2 illustrates the default NER tagset from CoreNLP. Although there are various techniques to implement NER, in this study, we are going to use the NER tagging method from Stanford CoreNLP, which is based on Conditional Random Fields (CRF) [17], to predict NER in music lyrics.

In Conditional Random Fields (CRF), authors from the Stanford NLP team proposed a method to enhance the NER system by incorporating long-distance dependency models. The method, Gibbs sampling, enforces the labelling consistency and consistency of extraction templates, which means that during the model training process, the system will enforce consistency in the labelling of the same entity across different occurrences. Based on the experiments, the CRF model with Gibbs sampling reduced the errors significantly and upgraded the F-1 score [19].

`PERSON` , `LOCATION` , `ORGANIZATION` , `MISC` , `MONEY` , `NUMBER` , `ORDINAL` , `PERCENT` , `DATE` , `TIME` , `DURATION` , `SET` , `EMAIL` , `URL` , `CITY` , `STATE_OR_PROVINCE` , `COUNTRY` , `NATIONALITY` , `RELIGION` , `TITLE` , `IDEOLOGY` , `CRIMINAL_CHARGE` , `CAUSE_OF_DEATH` , `HANDLE`

Figure 2. The default NER tag set from CoreNLP [20]**.**

2.5 Model Algorithms

2.5.1 Random Forest

Random forest is one of the ensemble learning algorithms, which can be applied to both regression and classification problems. In this study, we utilize random forest algorithm to classify music genres based on lyrics.

This algorithm integrates with a large number of decision trees, and it uses random operation to ensure that each decision tree is trained on a unique subset of the data. Additionally, another random operation is used for selecting a subset of features on node split in every tree. The randomness that gives random forest can avoid

over-fitting problems if we use single decision tree algorithm [14], [21].

In the final phase of the random forest algorithm, based on the prediction results from single decision tree, it uses majority voting mechanism to determine the model's final decision [21]. Figure 3 illustrates the process of random forest algorithm.

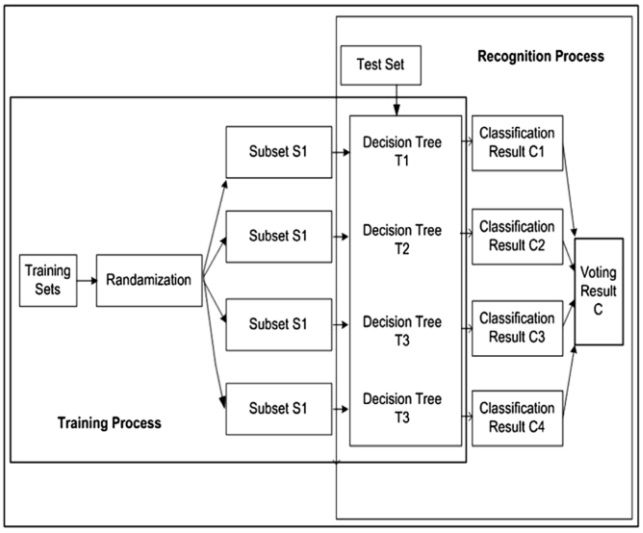


Figure 3. Process of Random Forest algorithm [21]**.**

2.5.2 XGBoost (eXtreme Gradient Boost)

XGBoost is an abbreviation of eXtreme Gradient Boost. It is an ensemble learning algorithms and an improved version of the Gradient Boosted Decision Tree (GBDT) algorithm, which means that XGBoost is a decision tree-based algorithm [22].

The core concept of XGBoost is to utilize gradient boosting to learn the residuals from the previous tree continuously and apply gradient descent to minimize the loss function. This enables the new tree to focus on reducing the residuals during training, finally improving accuracy. Moreover, XGBoost adds regularization in order to mitigate the risk of over-fitting [23], [24]. Figure 4 illustrates the training process of XGBoost.

The ensemble mechanism of XGBoost differs from that of Random Forest. Random Forest trains a number of decision trees simultaneously; however, the XGBoost trains decision trees sequentially to correct the residuals of the previous tree.

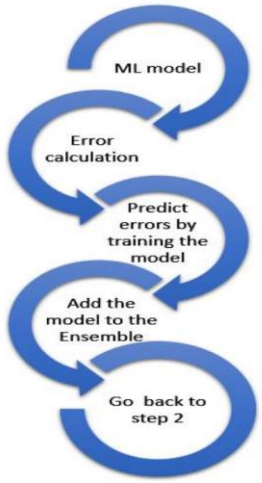


Figure 4. The training process of XGBoost [22]**.**

2.6 Evaluation Metrics

Evaluation metrics are always used to evaluate the model performance when a machine learning or deep learning model has been trained. In this study, we use confusion matrix to assess the model performance.

2.6.1 Confusion Matrix

Confusion matrix is one of the evaluation tools to evaluate the model performance, especially suitable for evaluating classification models. It is an $n \times n$ (n stands for the number of categories) array which represents the relationship between prediction results and ground truth. There are four measurements derived from the confusion matrix, as shown in Table 1.

Table 1. Confusion matrix

		Ground Truth	
		Positive	Negative
Predicted	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

1. True Positive (TP): The number of positive samples that the model predicts correctly.

2. False Positive (FP): The number of negative samples predicted by the model as positive.

3. True Negative (TN): The number of negative samples that the model predicts correctly.

4. False Negative (FN): The number of positive samples predicted by the model as negatives.

2.6.2 Accuracy

Accuracy is going to calculate the proportion of correct predictions from this model.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

2.6.3 Precision

Precision represents the proportion of samples predicted to be positive that are actually positive.

$$Precision = \frac{TP}{TP + FP}$$

2.6.4 Recall

Recall represents the proportion of all positive classes that are correctly predicted.

$$Recall = \frac{TP}{TP + FN}$$

2.6.5 F1 Score

The F1-score represents the harmonic mean of Precision and Recall. Additionally, in this study, because each class is equally important, we are going to look at the F1 score more crucially.

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

2.6.6 Maro/Micro/Weighted average

In multi-label classification, due to imbalanced data, different evaluation metrics have been discussed in [25]. The macro-average computes the arithmetic mean of Precision, Recall, and F1-score across all categories. The weighted-average calculates Precision, Recall, and F1-score weighted by the number of samples in each category. The micro-average aggregates the rue positives (TP),

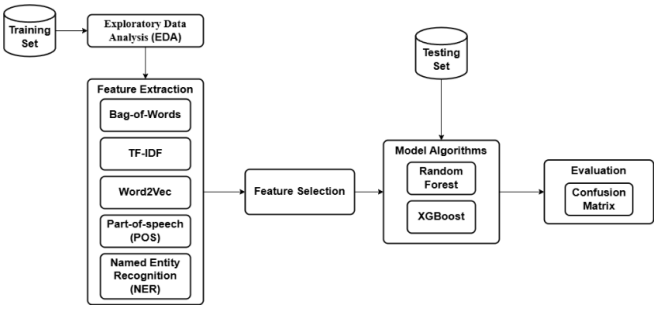
false positives (FP), and false negatives (FN) across all categories before computing Precision, Recall, and F1-score [25] [26]. In this study, although each class is considered equally important, the dataset is slightly imbalanced. Therefore, we apply the weighted-average F1 score to evaluate our multi-label classification performance. The equation of weighted-average F1 score is as follows, where N_i represents the number of samples in class i [26], [27].

$$Weighted\ F1\ score = \frac{N_1 \times F1_1 + N_2 \times F1_2 + \cdots + N_n \times F1_n}{N_1 + N_2 + \cdots + N_n}$$

3. METHODOLOGY

3.1 Methodology Flow

Figure 5 illustrates the methodology flowchart which we will elaborate on each step in detail.



Figures 5. The flowchart of research methodology.

3.2 Dataset and Exploratory Data Analysis (EDA)

3.2.1 Dataset

The dataset utilized in this study is sourced from Kaggle and is titled "English Music Lyrics 5 Genres (500k)" by Anurag Ghosh, last updated one year ago. This dataset consists of two CSV files: one designated for training, which contains 500,000 unique entries, and another for testing, which includes 50,000 unique entries.

Each of these files comprises two columns. The first column contains English music lyrics, spanning across five distinct genres: country, rap, metal, rock, and pop. The second column represents the corresponding genre of each song.

The primary reason for choosing this dataset is its large volume, containing 500,000 unique music lyrics, which provides a sufficient amount of data to effectively train our model. Additionally, the dataset is well-structured and straightforward, consisting of only two essential columns: one for the lyrics and another for the corresponding genre. This simplicity ensures that the dataset is both easy to process and perfectly aligned with our research objectives.

3.2.2 Exploratory Data Analysis

Because we wanted to reduce training time and maintain consistency, we set a fixed random seed, ensuring that our training and testing data remain the same across different runs. To make computation more efficient, we sampled 5,000 entries from the training dataset and 500 entries from the testing dataset while preserving the original 9:1 ratio. Since the original dataset consists of 500,000 training samples and 50,000 testing samples, this down sampling maintains the dataset's structure while significantly reducing processing time. We also use data visualization to ensure that our training dataset is balanced across different genres.

Moreover, after extracting POS and NER tag, we use heat maps and correlation coefficient analysis to observe the relationship between our various features and music genres before moving to the training phase.

3.3 Preprocessing

3.3.1 Re-encoding Labels

In order to facilitate model training, we re-encoded the music genres into numerical values. The mapping is as follows: country → 0, rap → 1, metal → 2, rock → 3, and pop → 4. This transformation allows our models to process categorical data more efficiently.

3.3.2 Stop Words

Stop words are common words such as 'the,' 'is,' and 'and' that do not carry significant meaning in a paragraph or text. Initially, we believed that stop words were not significant features for song classification. Therefore, to enhance the quality of our text features, we removed stop words from the lyrics. We utilized the NLTK stop words list and applied it to our preprocessed lyrics data. By filtering out these words, we aimed to ensure that our model would focus on more meaningful terms that contribute to classification.

However, we later discovered that keeping the stop words actually improved our accuracy[28]. As a result, we examined and compared the differences between keeping and removing the stop words in our model, which led to better insights into their impact on classification performance.

Furthermore, we found that many song lyrics contained descriptive words, such as "[Verse]" and "[Chorus]," which often appear at the beginning of the lyrics. These terms, while common in the context of the lyrics, do not contribute meaningful information for song classification and could potentially introduce noise into our model. To address this, we customized our stop words list to include these specific terms.

Figure 6 presents the word cloud for each genre after removing stop words. In these word clouds, the size of each word reflects its frequency within the corresponding genre, with larger words indicating higher occurrences.

3.4 Feature Extraction

As one of our goals is to keep the approach as simple as possible, we have decided to select some of the most used methods for

Figures 6. Word cloud of each genre after removing the stop words

similar tasks based on the papers we reviewed. Starting from the most simple one to the more complex one:

3.4.1 Bag-of-words

As one of the most commonly used feature extraction techniques, Bag-of-words doesn't consider the sequence between different

words. Despite its simplicity, Bag-of-words effectively captures word frequency, making it a solid baseline for text classification tasks.

After data preprocessing, we selected the top 5,000 most frequently occurring words in our lyrics dataset. This helps reduce computational complexity while retaining the most informative words for classification.

3.4.2 TF-IDF

As an enhancement of the Bag-of-Words approach, TF-IDF assigns weights to words based not only on their frequency within a given document but also on their importance across the entire corpus. This helps us downweight the words which commonly appear across the entire corpus while highlighting some more valuable words.

Same as the way we apply Bag-of-words, we also selected the top 5,000 most frequently occurring words in our lyrics dataset to improve computational efficiency.

3.4.3 Word2Vec

Inspired by a study that successfully applied Word2Vec to song genre classification[14] , we decided to apply this technique into our experiment. Word2Vec represents words as dense vector embeddings, capturing semantic relationships between words. This allows the model to leverage contextual meaning, making it more sophisticated than Bag-of-words and TF-IDF.

Because the basic training unit is a word, we first need to tokenize the lyrics to split sentences into words. After that we then applied for the Word2Vec model.

3.4.4 Part-of-speech (POS) and Named Entity Recognition (NER)

In this study, we aim to investigate whether the classification of a song can be predicted solely based on the proportion of different POS and named entities (NE) within its lyrics. To achieve this, we utilize the Stanford NLP toolkit, as introduced in Sections 2.4.5 and 2.4.6, to extract POS and NE from the lyrics. We then compute the proportion of each POS and NE within the song.

3.5 Feature Selection

After extracting POS and NER features, we identified a total of 63 features. Additionally, we observed that the POS tagging system distinguishes parts of speech in great detail. For example, "JJ" represents adjectives, "JJR" denotes comparative adjectives, and "JJS" indicates superlative adjectives. However, since all three belong to the adjective category, we merged them into a single feature representing adjectives. The same approach was applied to nouns, verbs, and adverbs.

Furthermore, we removed certain POS categories that were deemed less relevant, such as symbols, “to”, auxiliary verbs, and possessive-related tags. After merging and removing specific features, we retained 42 features without stop words removal and 34 features with stop words removal.

3.6 Machine Learning Classifier and Model Prediction Results

Lyrics, text data, is transformed into numerical representations through feature extraction techniques, including Bag-of-Words (BoW), TF-IDF, Word2Vec, Part-of-Speech (POS) tagging, and Named Entity Recognition (NER). These techniques help the model capture both semantic and statistical characteristics of the text. Subsequently, feature selection is performed to identify the most representative features, improving the model’s efficiency and accuracy.

After feature selection, the data is fed into machine learning models, primarily Random Forest and XGBoost, for classification training. These algorithms are well-suited for handling structured data and can effectively learn patterns within the text. Once training is complete, the model is evaluated using a test dataset. A Confusion Matrix is employed to analyze performance metrics, including accuracy, precision, and recall, providing insights into the model’s effectiveness.

Once the model has been trained and its predictions have been released. We applied different feature extraction methods and ran them through both models, obtaining various results. However, most combinations indicated that XGBoost performed better than Random Forest.

4. EXPERIMENT AND EVALUATION

4.1 Dataset Description

This dataset came from Kaggle and is about the English lyrics from five music genres: country, pop, metal, rap, and rock. It contains two CSV files which are the training set and the testing set. There are 500,000 samples of lyrics in the training set and 50,000 samples in the testing set. However, due to time constraints, we randomize the sampling of 5,000 and 500 instances from each dataset, which keeps the ratio between training and testing as the original proportion. Figure 9 provides a visual representation of the distribution, ensuring that no single genre dominates the dataset, with only a small 1.7% difference between the largest and smallest values.

Additionally, Figures 7 and 8 provide heatmaps: one with stop word removal and the other without. These heatmaps are based on correlation coefficients and show the correlation between each feature and the correlation of the extracted POS and NER features for the music genres. As a result, the features of the data without stop word removal have a higher correlation than others.

4.2 Experiment Method

We divided the lyrics into those with and without stop words removed and used all the feature extraction methods mentioned in 3.4 and combined POS+NER and TF-IDF methods to run the Random Forest and XGBoost models separately. Therefore, there are a total of 20 experimental combinations. Finally, we use the confusion matrix to evaluate the performance of the model. We also focus on the weight F1 score indicator as the basis for our assessment.

4.3 Results Evaluation

The results of all experimental combinations are shown in Table 2 and 3. Each combination of confusion matrix is shown in the Appendix.

4.3.1 Random Forest

Table 2. Results using Random Forest

Method	Stop Words Removed	Accuracy	Weighted F1-score
Bag-of-word	Y	0.54	0.53
	N	0.61	0.61
TF-IDF	Y	0.56	0.56
	N	0.63	0.63

Word2Vec	Y	0.51	0.51
	N	0.55	0.55
POS + NER	Y	0.35	0.34
	N	0.49	0.49
POS + NER + TF-IDF	Y	0.572	0.57
	N	0.608	0.61

4.3.2 XGBoost

Table 3. Results using XGBoost

Method	Stop Words Removed	Accuracy	Weighted F1-score
Bag-of-word	Y	0.61	0.59
	N	**0.65**	**0.63**
TF-IDF	Y	0.60	0.59
	N	**0.65**	**0.63**
Word2Vec	Y	0.51	0.51
	N	0.56	0.56
POS + NER	Y	0.362	0.36
	N	0.474	0.47
POS + NER + TF-IDF	Y	0.554	0.56
	N	0.616	0.62

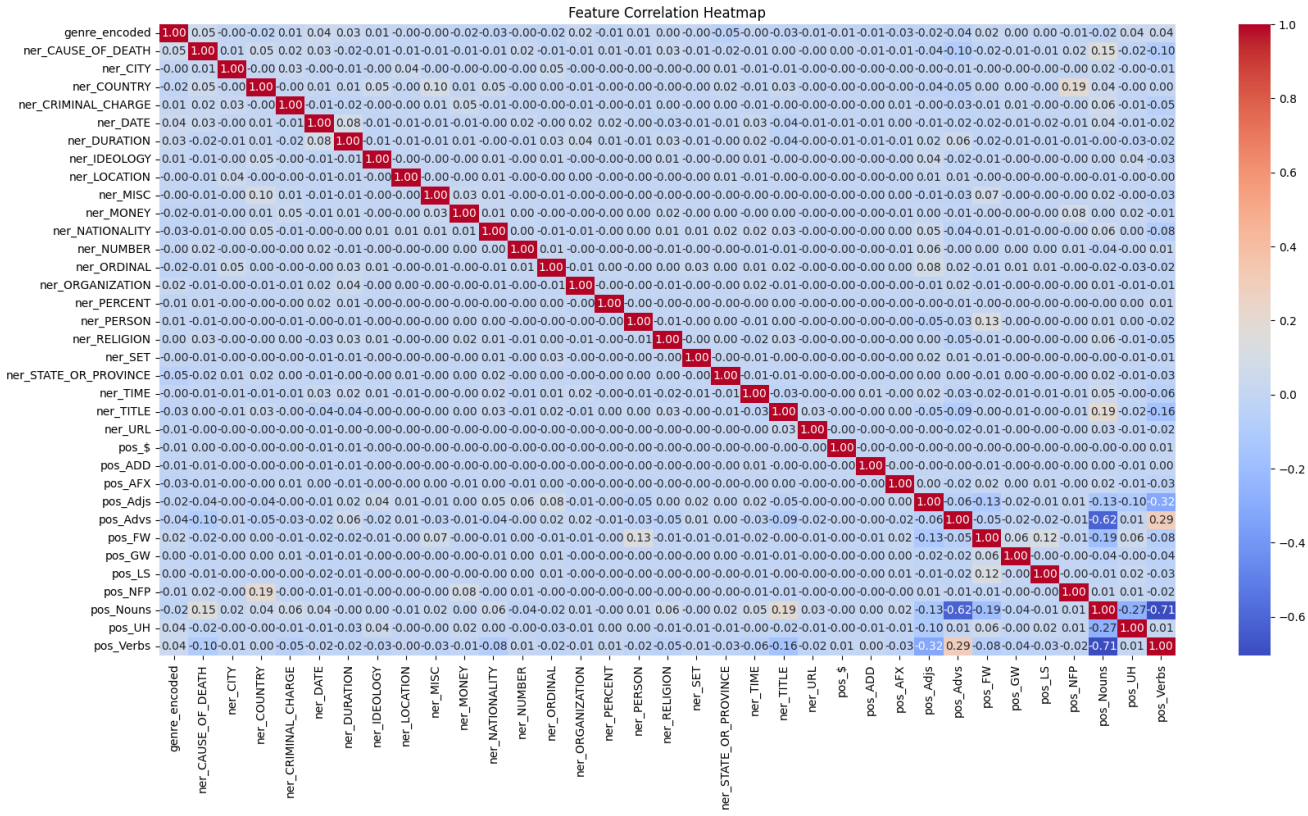


Figure 7. The heatmaps with stop words removed.

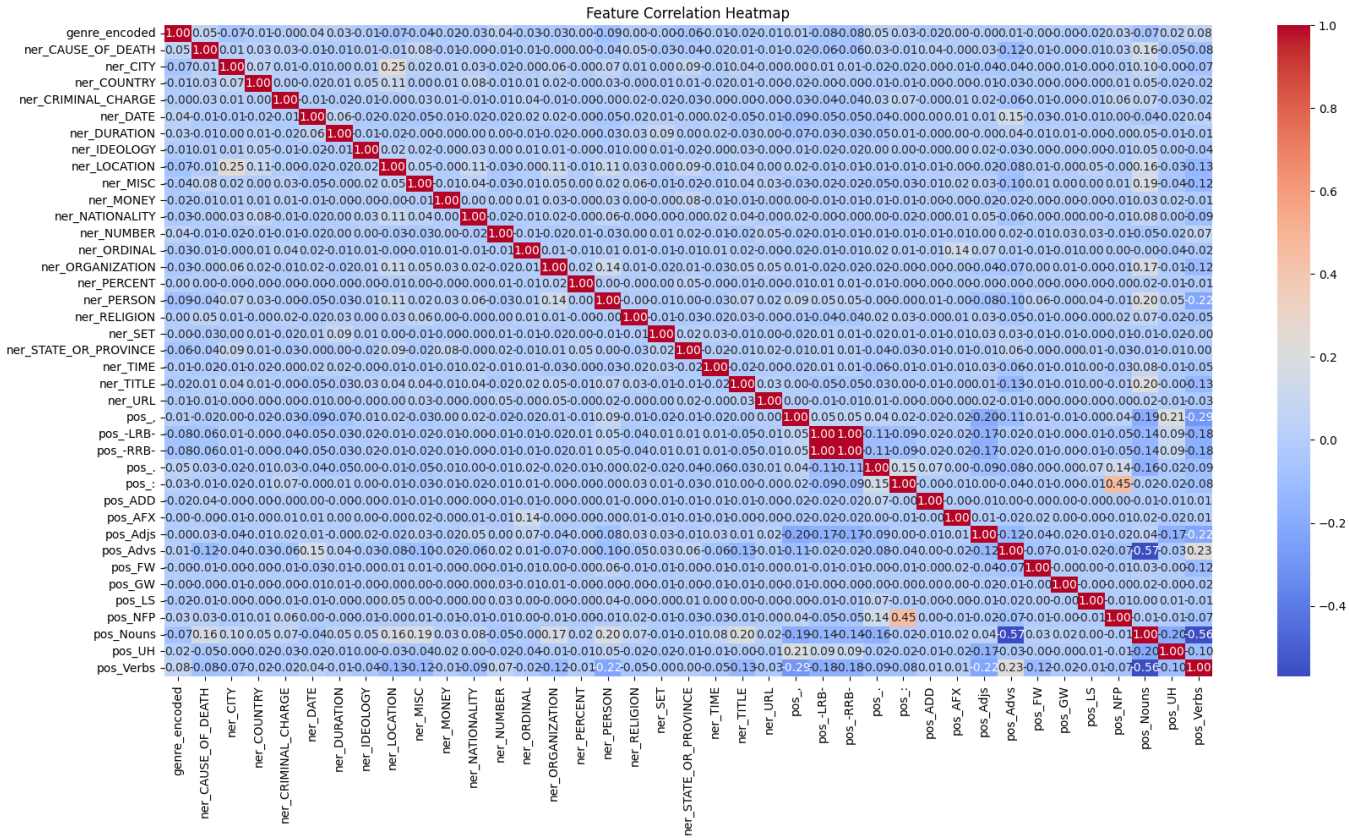


Figure 8. The heatmaps with stop words kept.

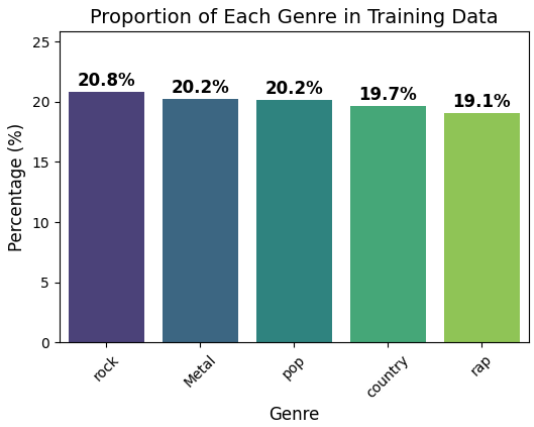


Figure 9. Proportion of each genre in training data.

5. CONCLUSION

This study explored the feasibility of using lyrics-based feature extraction methods for music genre classification. We experimented with Bag-of-Words, TF-IDF, Word2Vec, and Part-of-Speech (POS) with Named Entity Recognition (NER) and evaluated their effectiveness using two machine learning classifiers: Random Forest and XGBoost. Additionally, we examined the impact of stop-word removal on classification performance.

Our results indicate that TF-IDF and Bag-of-Words yielded the highest classification accuracy, particularly when stop words were retained. In contrast, Word2Vec and POS+NER features exhibited lower classification performance, likely due to the inherent challenges of capturing semantic and structural nuances in song lyrics. XGBoost consistently outperformed Random Forest across all feature extraction methods, suggesting that gradient-boosting techniques are better suited for handling textual features in this classification task.

The findings also highlight that lyrics-based classification faces significant limitations, primarily due to the subtle differences between genres and the overlap in lyrical content. Unlike traditional text classification tasks, song lyrics contain poetic and rhythmic elements that make distinguishing between genres more complex. Our results align with existing research suggesting that lyrics alone may not be sufficient for highly accurate genre classification, and incorporating audio-based features could enhance performance [29].

Overall, this study contributes to the growing body of research on text-based music classification by systematically evaluating different feature extraction techniques. Future work could explore deep learning approaches, such as transformer-based language models, or multimodal classification methods that combine lyrical and audio features to improve genre classification accuracy.

6. REFERENCES

[1] A. O. Salau and S. Jain, "Feature Extraction: A Survey of the Types, Techniques, Applications," in *2019 International Conference on Signal Processing and Communication (ICSC)*, NOIDA, India: IEEE, Mar. 2019, pp. 158–164. doi: 10.1109/ICSC45622.2019.8938371.

[2] Ø. Due Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition-A survey," *Pattern Recognition*, vol. 29, no. 4, pp. 641–662, Apr. 1996, doi: 10.1016/0031-3203(95)00118-2.

[3] H. Liang, X. Sun, Y. Sun, and Y. Gao, "Text feature extraction based on deep learning: a review," *J Wireless Com Network*, vol. 2017, no. 1, p. 211, Dec. 2017, doi: 10.1186/s13638-017-0993-1.

[4] M. Fell and C. Sporleder, "Lyrics-based Analysis and Classification of Music".

[5] R. Mayer, R. Neumayer, and A. Rauber, "RHYME AND STYLE FEATURES FOR MUSICAL GENRE CLASSIFICATION BY SONG LYRICS".

[6] A. Singhi and D. G. Brown, "ARE POETRY AND LYRICS ALL THAT DIFFERENT?," 2014.

[7] S.-H. Chen, Y.-S. Lee, W.-C. Hsieh, and J.-C. Wang, "Music emotion recognition using deep Gaussian process," in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, Hong Kong: IEEE, Dec. 2015, pp. 495–498. doi: 10.1109/APSIPA.2015.7415321.

[8] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "Music classification via the bag-of-features approach," *Pattern Recognition Letters*, vol. 32, no. 14, pp. 1768–1777, Oct. 2011, doi: 10.1016/j.patrec.2011.06.026.

[9] A. Tsaptsinos, "Lyrics-Based Music Genre Classification Using a Hierarchical Attention Network," Jul. 15, 2017, *arXiv*: arXiv:1707.04678. doi: 10.48550/arXiv.1707.04678.

[10] S. Cs and M. Leszczynski, "Music Genre Classification using Song Lyrics".

[11] N. Scaringella, G. Zoia, and D. Mlynek, "Automatic genre classification of music content: a survey," *IEEE Signal Process. Mag.*, vol. 23, no. 2, pp. 133–141, Mar. 2006, doi: 10.1109/MSP.2006.1598089.

[12] M. P. Barman, A. Awekar, and S. Kothari, "Decoding The Style And Bias of Song Lyrics," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Paris France: ACM, Jul. 2019, pp. 1165–1168. doi: 10.1145/3331184.3331363.

[13] Y. HaCohen-Kerner, D. Miller, and Y. Yigal, "The influence of preprocessing on text classification using a bag-of-words representation," *PLoS ONE*, vol. 15, no. 5, p. e0232525, May 2020, doi: 10.1371/journal.pone.0232525.

[14] A. Kumar, A. Rajpal, and D. Rathore, "Genre Classification using Word Embeddings and Deep Learning," in *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Bangalore: IEEE, Sep. 2018, pp. 2142–2146. doi: 10.1109/ICACCI.2018.8554816.

[15] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a Large Annotated Corpus of English: The Penn Treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[16] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network," in *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2003, pp. 252–259. Accessed: Mar. 09, 2025. [Online]. Available: https://aclanthology.org/N03-1033/

[17] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*,

K. Bontcheva and J. Zhu, Eds., Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 55–60. doi: 10.3115/v1/P14-5010.

[18] Z. Nasar, S. W. Jaffry, and M. K. Malik, "Named Entity Recognition and Relation Extraction: State-of-the-Art," *ACM Comput. Surv.*, vol. 54, no. 1, p. 20:1-20:39, 11 2021, doi: 10.1145/3445965.

[19] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling," in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL`05)*, K. Knight, H. T. Ng, and K. Oflazer, Eds., Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 363–370. doi: 10.3115/1219840.1219885.

[20] "CoreNLP NER | Datasaur." Accessed: Mar. 16, 2025. [Online]. Available: https://docs.datasaur.ai/assisted-labeling/ml-assisted-labeling/corenlp-ner

[21] A. Parmar, R. Katariya, and V. Patel, "A Review on Random Forest: An Ensemble Classifier," in *International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018*, J. Hemanth, X. Fernando, P. Lafata, and Z. Baig, Eds., Cham: Springer International Publishing, 2019, pp. 758–763. doi: 10.1007/978-3-030-03146-6_86.

[22] S. S. Azmi and S. Baliga, "An Overview of Boosting Decision Tree Algorithms utilizing AdaBoost and XGBoost Boosting strategies," vol. 07, no. 05, 2020.

[23] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA: ACM, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[24] T. R. Mahesh, V. Vinoth Kumar, V. Muthukumaran, H. K. Shashikala, B. Swapna, and S. Guluwadi, "Performance Analysis of XGBoost Ensemble Methods for Survivability with the Classification of Breast Cancer," *Journal of Sensors*, vol. 2022, pp. 1–8, Sep. 2022, doi: 10.1155/2022/4649510.

[25] D. Krstinić, A. K. Skelin, I. Slapničar, and M. Braović, "Multi-Label Confusion Tensor," *IEEE Access*, vol. 12, pp. 9860–9870, 2024, doi: 10.1109/ACCESS.2024.3353050.

[26] H. F. SOON, "Imbalanced Network Attack Class Classification Using an Ensemble Machine Learning Approach.," 2024. [Online]. Available: https://yamanashi.repo.nii.ac.jp/record/2000283/files/Text_2024KD_156.pdf

[27] A. A. Abbasi, A. Zameer, E. Mushtaq, and M. A. Z. Raja, "Cost-sensitive stacked long short-term memory with an evolutionary framework for minority class detection," *Applied Soft Computing*, vol. 165, p. 112098, Nov. 2024, doi: 10.1016/j.asoc.2024.112098.

[28] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and Word2vec for text classification with semantic features," in *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, Beijing, China: IEEE, Jul. 2015, pp. 136–140. doi: 10.1109/ICCI-CC.2015.7259377.

[29] S. Ho, "Using sentence embeddings to explore the similarities and differences in song lyrics," TDS Archive. Accessed: Mar. 16, 2025. [Online]. Available: https://medium.com/data-science/using-sentence-embeddings-to-explore-the-similarities-and-differences-in-song-lyrics-1820ac713f00

7. WORK DIVISION

Name	Work description	Contribution
Yun-Ting Lai 113423040	1. Survey feature extraction method (Bag-of-words, TF-IDF, Word2Vec) 2. Data Preprocessing 3. Implementation (Bag-of-words, TF-IDF, Word2Vec)	50%
Wei-Siang Chen 113423052	1. Survey feature extraction method (POS, NER) 2. Survey model algorithm (Random Forest, XGBoost) 3. Implementation: POS+NER, TF-TDF+POS+NER 4. Feature Selection	50%

8. Appendix

8.1 Confusion Matrix-Random Forest

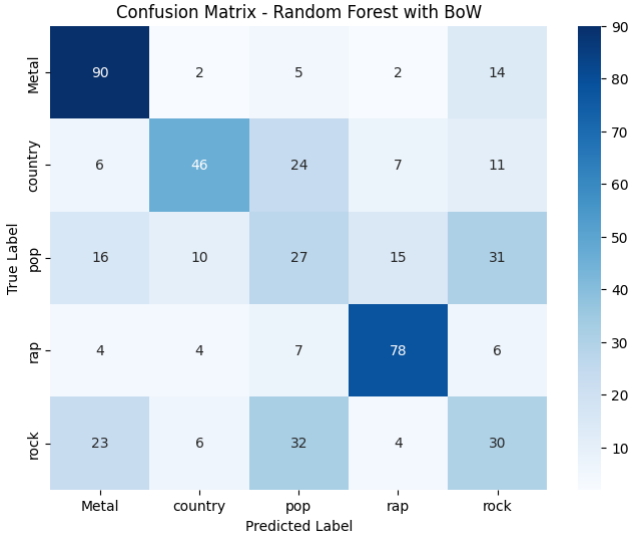


Figure 10. BoW Stop Words Removed

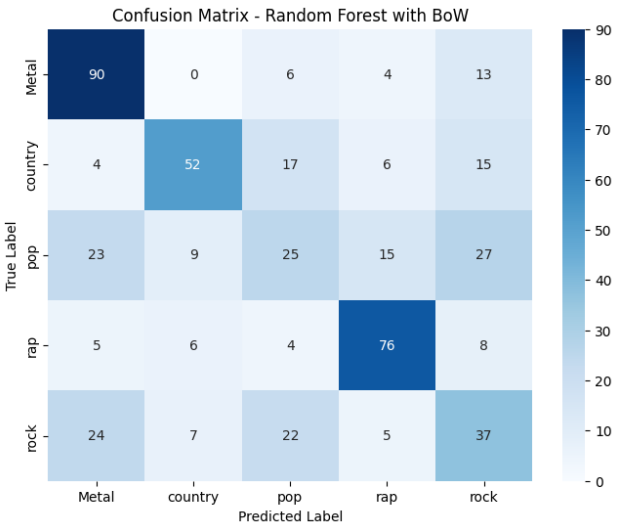


Figure 11. BoW Stop Words Kept

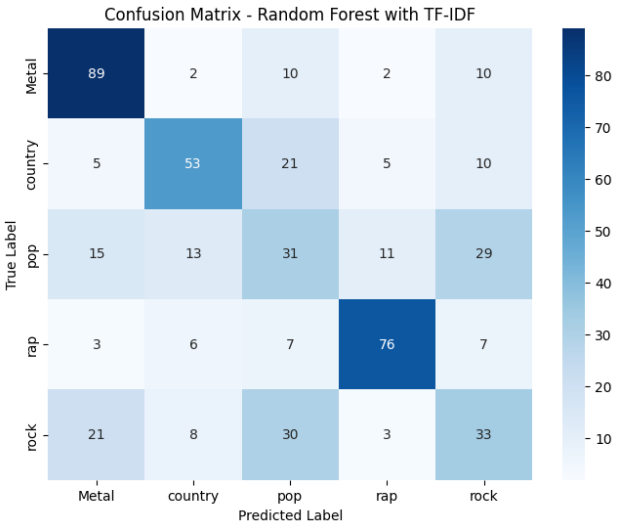


Figure 12. TF-IDF Stop Words Removed

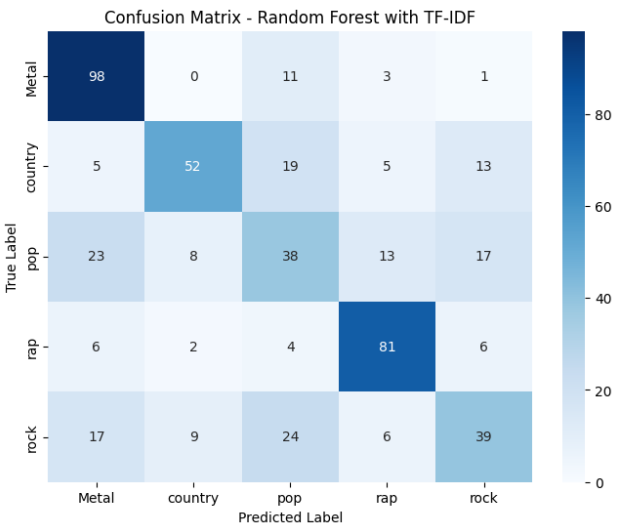


Figure 13. TF-IDF Stop Words Kept

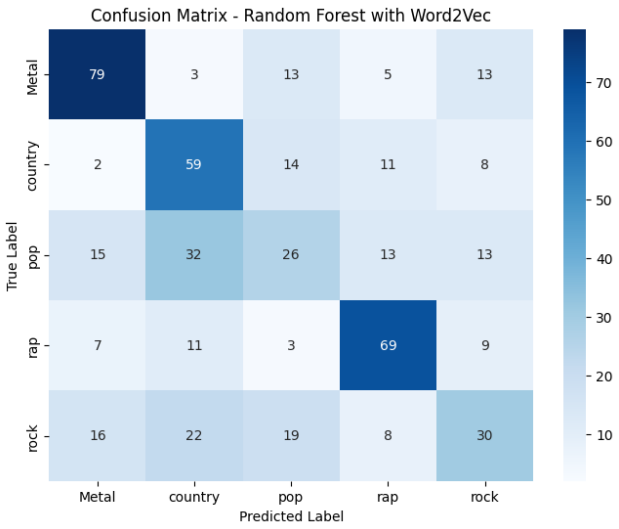


Figure 14. Word2Vec Stop Words Removed

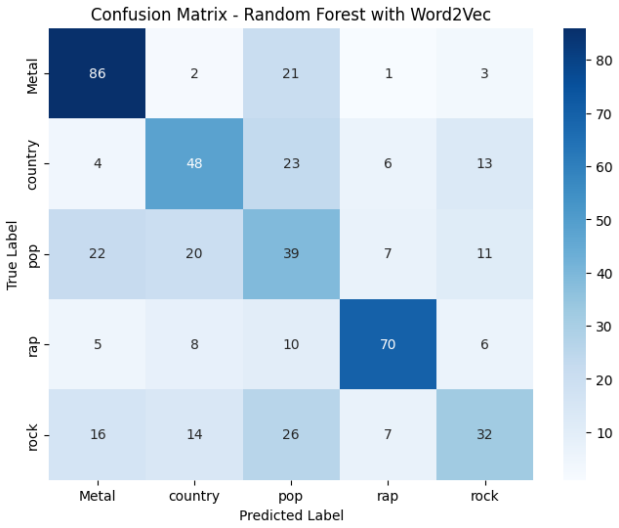


Figure 15. Word2Vec Stop Words Kept

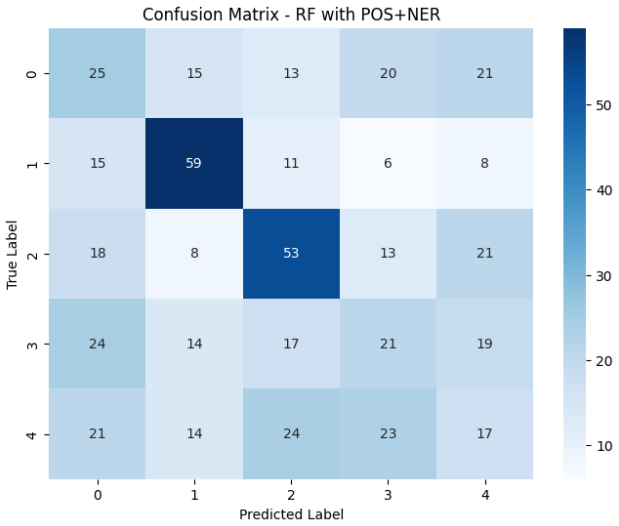


Figure 16. POS+NER with Stop Words Removal

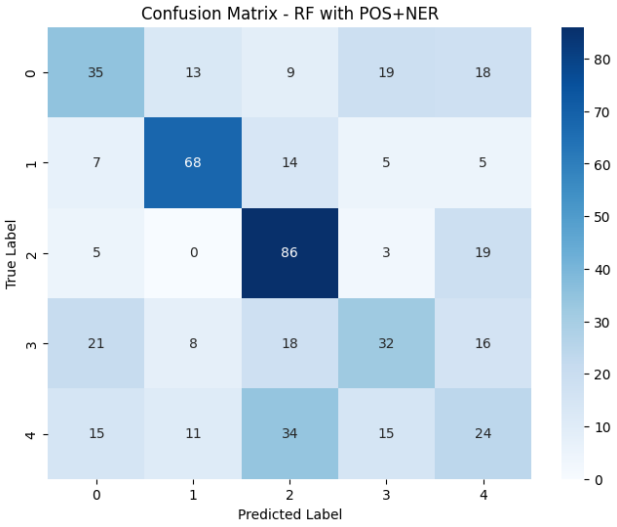


Figure 17. POS+NER without Stop Words Removal

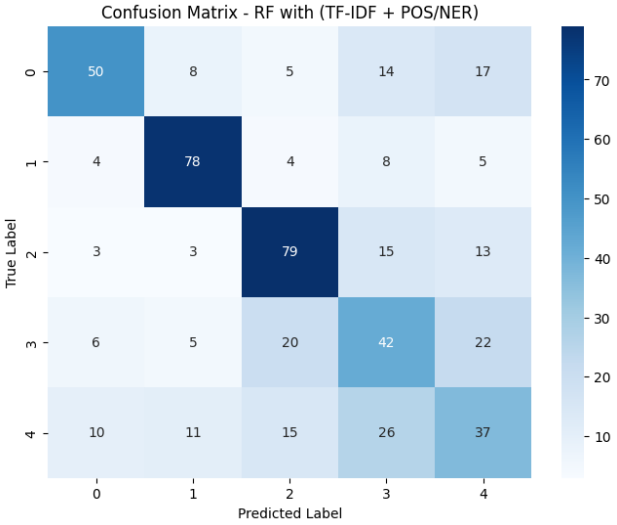


Figure 18. TF-IDF+POS+NER Stop Words Removed

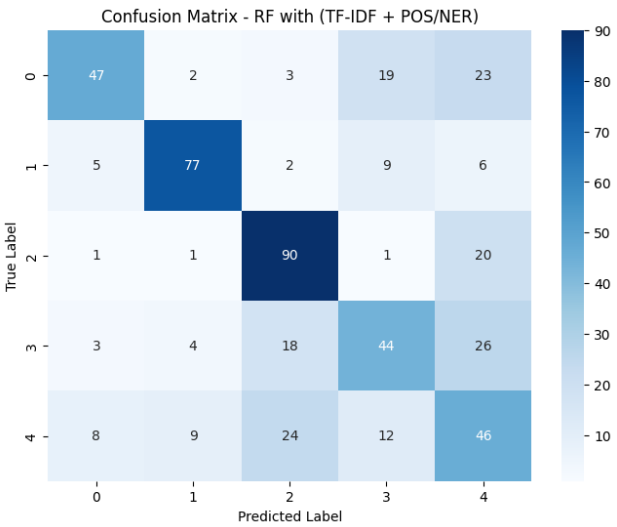


Figure 19. TF+IDF+POS+NER Stop Words Kept

8.2 Confusion Matrix-XGBoost

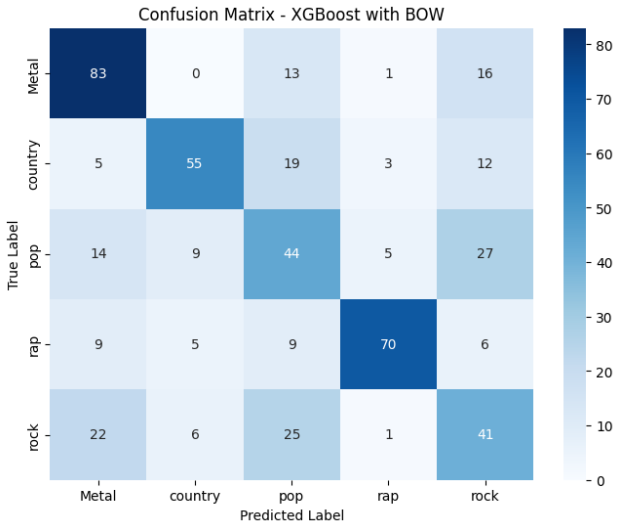


Figure 20. BoW Stop Words Removed

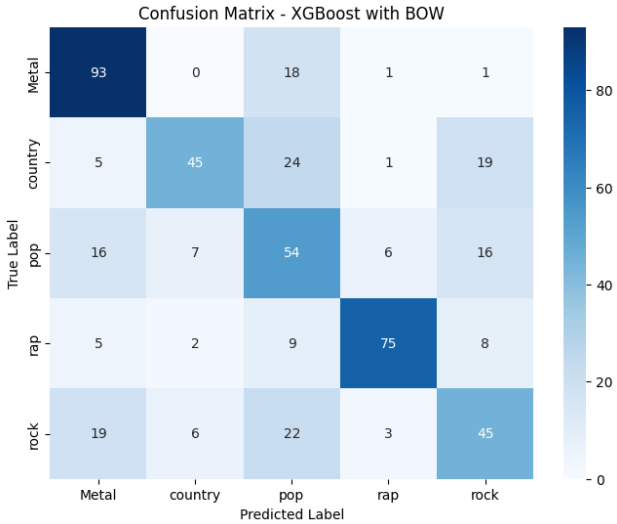


Figure 21. BoW Stop Words Kept

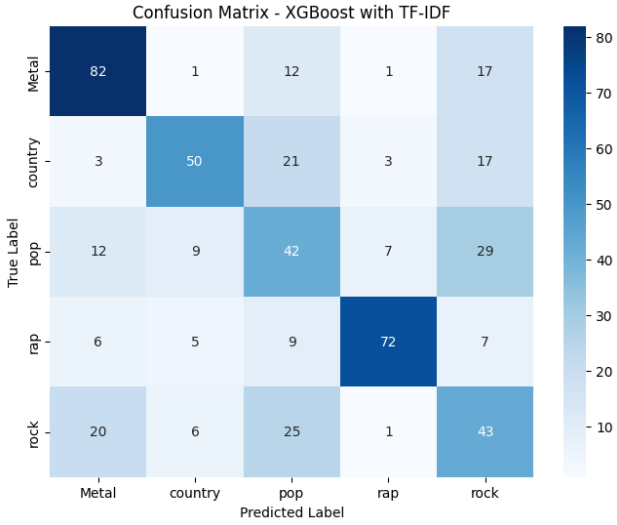


Figure 22. TF-IDF Stop Words Removed

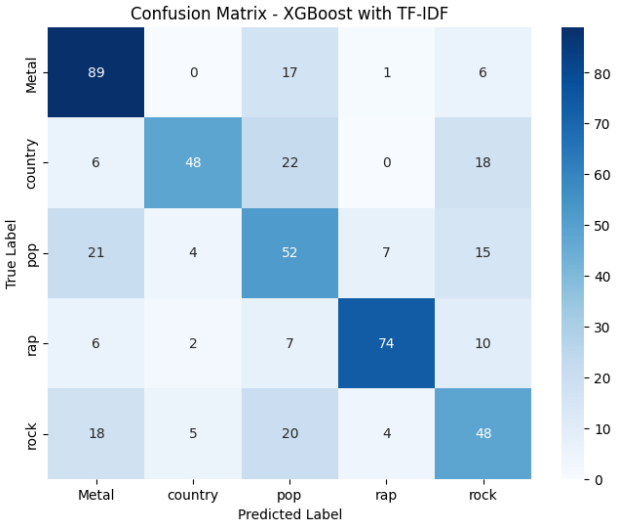


Figure 23. TF-IDF Stop Words Kept

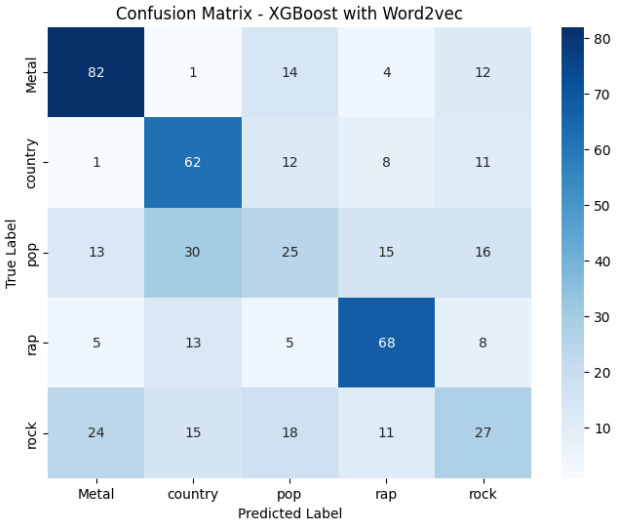


Figure 24. Word2Vec Stop Words Removed

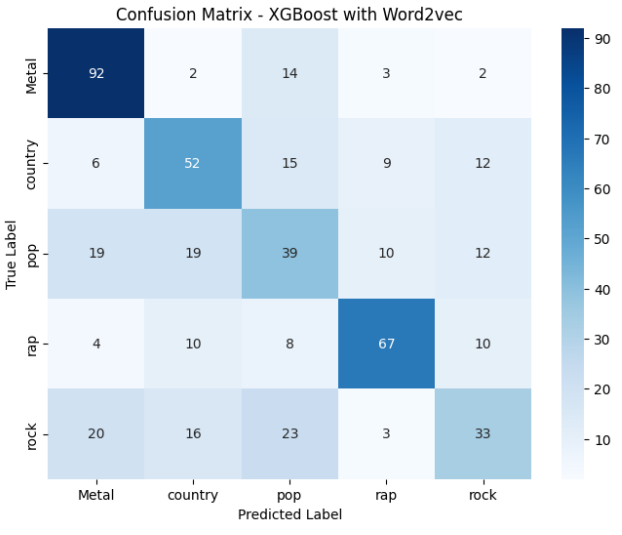


Figure 25. Word2Vec Stop Words Kept

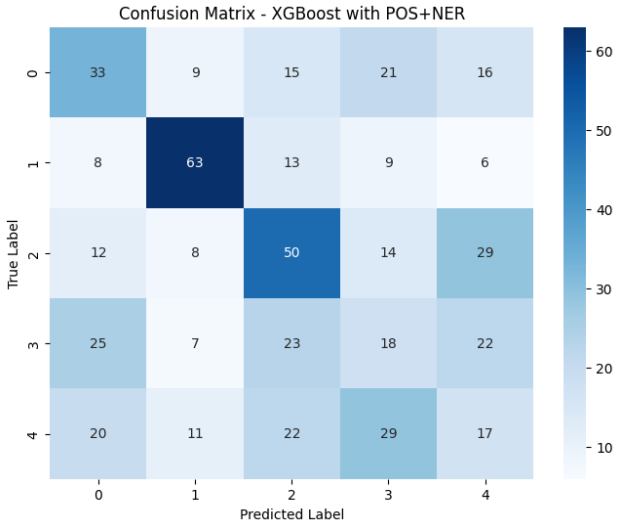


Figure 26. POS+NER Stop Words Removed

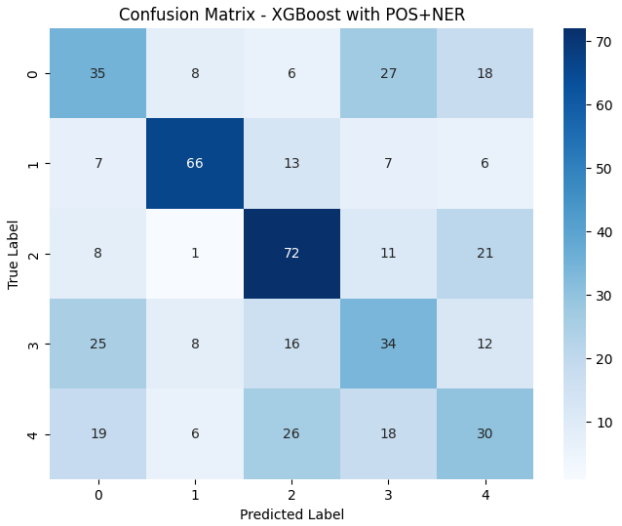


Figure 27. POS+NER Stop Words Kept

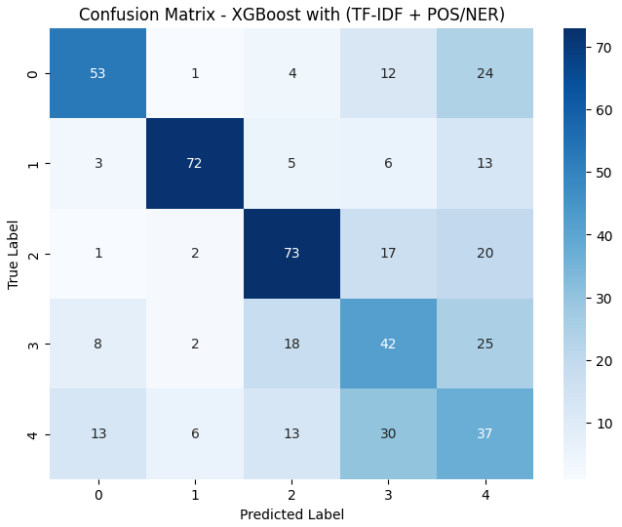


Figure 28. TF-IDF+POS+NER Stop Words Removed

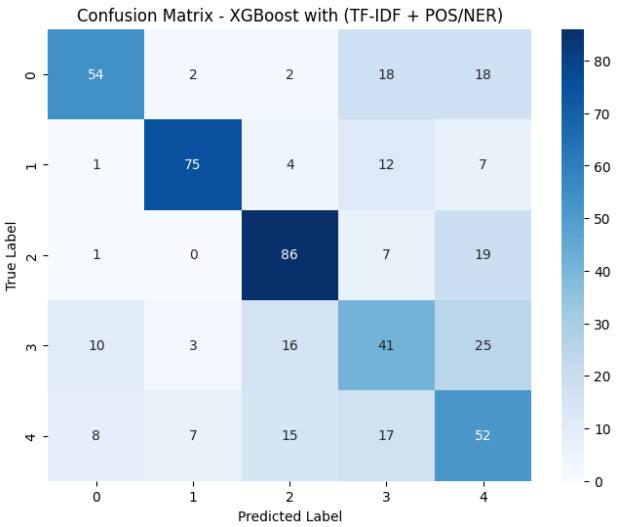


Figure 29. TF+IDF+POS+NER Stop Words Kept