BALKAN JOURNAL OF ELECTRICAL & COMPUTER ENGINEERING, Vol. 13, No. 3, September 2025
Research Article

The Impact of Artificial Intelligence on Sentiment Analysis Detection in Music Reviews

Murat Simsek and Bugra Kagan Kayhan

***Abstract*—This study aims to perform sentiment and content analysis of Spotify user reviews using machine learning and deep learning methods. The goal is to better understand users' experiences and satisfaction. The study employs various machine learning and deep learning techniques to identify the emotional tendencies in user reviews and analyze the relationship between these tendencies and content features. The performance of these methods is evaluated using various metrics such as accuracy, precision, recall, and F1-score. By identifying the strengths and weaknesses of each method, the study determines which techniques are more effective in specific situations. The results provide valuable insights for improving the quality of music streaming services and enhancing user experience. This study aims to help service providers increase user satisfaction by gaining a better understanding of user feedback. Additionally, these analyses are expected to provide valuable data for future improvements in music streaming services. Thus, it will be possible to continuously improve user experiences and enhance service quality.**

Index Terms—**Sentiment Analysis, Machine Learning, Deep Learning, Large Language Model, Natural Language Processing, LSTM.**

I. Introduction

MUSIC IS an integral part of our society as a universal way to express and understand emotions. With the advancement of technology, music streaming services have become platforms that profoundly impact users' music experiences. As of March 2022, Spotify is one of the largest providers in this field, boasting 422 million monthly active users and 182 million paying subscribers. Users express their experiences with the Spotify app and their satisfaction through various ratings and reviews.

When users leave reviews, they not only provide likes or star ratings but also add comments that include their own expressions. This indicates that the number of stars or a single like button is not the only indicator of user satisfaction; the content of the written comment is also very significant. Hence, it is necessary to conduct sentiment analysis on all reviews to determine the emotional content expressed in these comments.

Artificial intelligence (AI)-based sentiment analysis involves the use of machine learning and deep learning techniques to automatically identify emotional tendencies in text data.
The integration of artificial intelligence (AI) into sentiment analysis for music reviews has seen noteworthy advancements, particularly with the adoption of machine learning and deep learning models The credibility of AI systems, as explored by Khan and Mishra, is crucial for facilitating positive consumer-AI interactions and enhancing trust in AI-generated insights[1]. Furthermore, the epistemic implications of AI, as discussed by Miragoli, underscore the importance of addressing structural biases to prevent epistemic injustices in AI applications[2]. These insights collectively underscore the transformative potential of AI in music sentiment analysis while highlighting areas that require further exploration and refinement. Recent studies highlight the efficacy of techniques such as convolutional neural networks (CNN) and recurrent neural networks (RNN), which have significantly enhanced the interpretation of emotive language in music critiques[3][4]. Moreover, transformer-based models like BERT and GPT have been pivotal in achieving more precise context analysis, thereby capturing nuanced sentiments that conventional models might overlook[5]. These advancements have not only improved accuracy but have also expanded the scope of sentiment analysis to allow for real-time processing and deeper insights into consumer preferences[3]. Nevertheless, challenges remain, particularly concerning data bias and the interpretability of models.

Academic studies in this field have primarily focused on analyzing large amounts of text data, such as social media posts, product reviews, and customer feedback[6-7]. Machine learning algorithms, such as Naive Bayes, Support Vector Machines (SVM), and Logistic Regression, are commonly used methods in sentiment analysis [8]. In recent years, deep learning models, especially Long Short-Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT), have achieved notable successes in sentiment analysis, identifying emotional tendencies in text data with higher accuracy rates and a better understanding of the contextual meaning of language [9]. AI-based sentiment analysis has made significant contributions across a wide range of applications, from commercial uses to social research [10].

Literature studies have shown that LSTM (Long Short-Term

Murat Şimşek, is with Department of Artificial Intelligence Engineering of Ostim Technical University, Ankara, Turkey,(e-mail: murat.simsek@ostimteknik.edu.tr).
https://orcid.org/0000-0002-8648-3693

Buğra Kağan Kayhan, is with Department of Software Engineering of Ostim Technical University, Ankara, Turkey, (e-mail: 220904011@ostimteknik.edu.tr).
https://orcid.org/0009-0008-0501-3982

Manuscript received May 30, 2024; accepted Apr 27, 2025.
DOI: 10.17694/bajece.1492287

Memory) models are used for emotion analysis from music. In one study, the aim was to predict the emotional values of musical clips over time and determine the subsequent emotional value in a time series. Models were trained on music clips annotated with levels of valence and arousal using the Emotions in Music database. Mel Spectrograms extracted from music clips were used as input data, and the model's accuracy rates were evaluated using the mean squared error (MSE) metric. The results indicate that the LSTM model is effective in emotion analysis from music [11].

A study using data mining methods to analyze the emotional effects of music on people examined the impact of rhythm and timbre features extracted from music tracks on human emotions. The data consisted of 593 songs from the Mulan database and 72 musical features related to these songs. The songs were classified into six different emotion categories (sadness, joy, anger, fear, surprise, and calm). Data mining algorithms such as Random k-Labelsets (RAkEL) and Multi-Label k-Nearest Neighbors (MLkNN) were used to solve the multi-label classification problem. While the RAkEL algorithm achieved an accuracy of 78.8% and a Hamming loss of 21.2%, the MLkNN algorithm provided an accuracy of 80.4% and a Hamming loss of 19.6% [12].

A study utilizing natural language processing (NLP) and machine learning (ML) techniques to perform sentiment analysis on music reviews from Pitchfork.com examined the relationship between album reviews by Pitchfork critics and the numerical scores given to these reviews. The data was taken from a Kaggle dataset containing 18,376 observations. Preprocessing steps included converting texts to lowercase, removing stopwords, cleaning punctuation, and lemmatization. Feature engineering included extracting features such as review length, percentages of positive, negative, and neutral sentiments, rates of long words, counts of album/artist names, and similarity to top reviews. Models such as L_1 and L_2regularized linear regression, SVM regressor, and Stochastic Gradient Descent (SGD) regression were used for training. Models were evaluated using GridSearchCV for hyperparameter optimization. Performance evaluation used metrics such as mean squared error (MSE), mean absolute error (MAE), and R^2. Results indicated a 30% improvement in R^2 score with feature engineering, with the best performance provided by the linear regression model.

A study using natural language processing (NLP) and machine learning methods to perform sentiment analysis on user reviews of musical instruments on Amazon used a dataset of 10,262 reviews aiming to classify the reviews as positive or negative. Preprocessing steps included converting text to lowercase, removing punctuation, tokenization, removing stopwords, and lemmatization. Reviews were analyzed using the SpaCy library and classified using a Convolutional Neural Network (CNN) model. During training, ratings between 1-3 stars were labeled as negative, while 4-5 stars were labeled as positive. After training, the model's accuracy, precision, recall, and F1 score metrics were evaluated, achieving 78% precision and 79% F1 score. This shows that deep learning techniques can be effectively used for sentiment analysis of Amazon musical instrument reviews [13].

The studies reviewed have assessed that machine learning, deep learning, and recently, large language models play an effective role in sentiment analysis. This study aimed to perform sentiment and content analysis of Spotify user reviews using machine learning and deep learning methods. Sentiment analysis involves classifying text data as positive, negative, or neutral and plays a significant role in evaluating dynamic data sources containing large amounts of text data, such as social media and customer feedback. The goal was to automatically detect emotional tendencies in user reviews and analyze content features.

This study presents a comprehensive approach to sentiment analysis by integrating machine learning and deep learning techniques to analyze Spotify user reviews, thereby providing a multifaceted perspective on user sentiments. It rigorously evaluates various models, including LSTM and BERT, using performance metrics such as accuracy, precision, recall, and F1-score, allowing for a detailed comparison of their effectiveness in capturing emotional content. The study also addresses class imbalance through the application of the Synthetic Minority Over-sampling Technique (SMOTE), enhancing model robustness. Moreover, the findings offer industry-relevant insights, enabling music streaming platforms like Spotify to improve user satisfaction by understanding feedback trends. Finally, the research identifies challenges and suggests optimizations, providing a foundation for scalable sentiment analysis models adaptable to various music platforms.

II. MATERIALS AND METHODS

Spotify is a leading music streaming platform with over 422 million monthly active users worldwide, of which 182 million are paid subscribers. Users frequently express their experiences and opinions on the platform through reviews. These reviews contain valuable feedback about the quality of musical works and artist performances. Performing sentiment analysis on these comments is critical for measuring user satisfaction and understanding trends in the music industry as well as changes in user preferences. Particularly, accurate analyses can enhance music production and marketing strategies, enabling artists and record companies to respond more effectively to user demands.

A. Exploratory Data Analysis

In this study, a dataset containing user reviews from the Spotify application was used. This dataset, titled "Spotify App Reviews," was sourced from Kaggle and collected via scraping methods from the Google Store by M. Faarisul Ilmi [14]. The dataset comprises 61,594 rows and 5 columns.

- Time_submitted: Indicates the time frame in which the review was submitted.
- Review: Contains the text of the review.
- Rating: Shows the score given by the user (ranging from 1 to 5).
- Total_thumbsup: Indicates how many people found the review helpful.

- Reply: Contains the response to the review.

In the mentioned columns, all except for Time_submitted and Reply were used in the exploratory data analysis section. An overview of the dataset has been conducted, and the dataset information is in Table 1.

TABLE I
DATASET INFORMATION

Column	Non-Null Count	Dtype	Entries	Memory Usage
Time_submitted	61594 non-null	object	0 to 61593	2.3+ MB
Review	61594 non-null	object	0 to 61593	2.3+ MB
Rating	61594 non-null	int64	0 to 61593	2.3+ MB
Total_thumbsup	61594 non-null	int64	0 to 61593	2.3+ MB
Reply	216 non-null	object	0 to 61593	2.3+ MB

When Table 1 is examined, it is observed that there are only 216 data entries in the 'Reply' column, but considering the dataset has 61,594 rows, it is understood that most of the data for this feature is missing. When sentiment analysis from comments is targeted, the impact of this feature on sentiment analysis has been evaluated as weak, and it has been discarded along with the 'Time Submitted' column as part of Feature Reduction.

In the visualization related to our target column 'Rating', there are comments with ratings ranging from 1 to 5 stars. In Fig. 1, the number of comments varying from 1 to 5 stars in the Rating column has been shown.

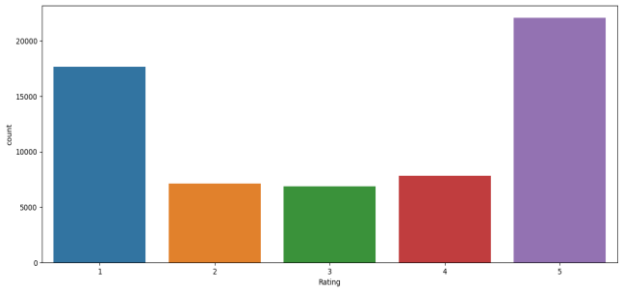

Fig.1. Histogram Chart According to Number of Stars in Rating Column

When Fig.1 is examined, comments range from 1 to 5 stars. In this context, 4 and 5 stars are considered positive, 3 stars neutral, and 1 and 2 stars negative. This transformation of star ratings is graphically represented in Fig.2.

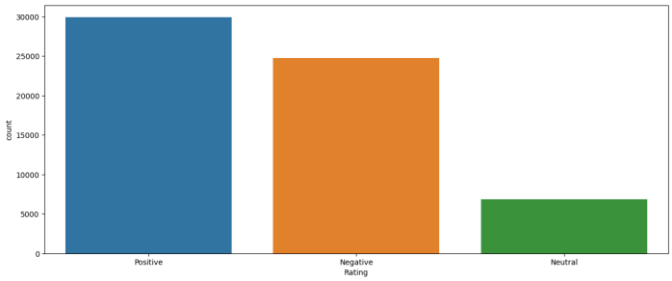

Fig.2. Star Transformation in Rating Feature

B. *Text Preprocessing*

Text preprocessing is a crucial step in natural language processing (NLP) projects to make data analyzable. This process includes methods like converting to lowercase, removing punctuation, tokenization, removing stopwords, and lemmatization. Each step aims to simplify the textual complexity to prepare the dataset for better analytical performance. Removing punctuation marks and converting text to lowercase are particularly emphasized, as they help eliminate noise and standardize the text for machine learning models, which is vital for consistent and effective processing [15].

In the realm of sentiment analysis and other NLP applications, techniques such as tokenization and lemmatization play significant roles. Tokenization segments text into manageable units, enhancing the machine learning models' ability to learn patterns within the text effectively. Lemmatization, on the other hand, delves deeper into the semantic aspect by reducing words to their base or dictionary forms, thus aiding in more accurate text classification and sentiment analysis by focusing on the semantic and contextual values of words[16]. After these preprocessing steps, tools like the Python Word Cloud library are employed to visualize frequently occurring words, providing insights into the predominant themes within the text data. The Word Cloud, which consists of frequently occurring words using the Python Word Cloud library, is shown in Fig 3.

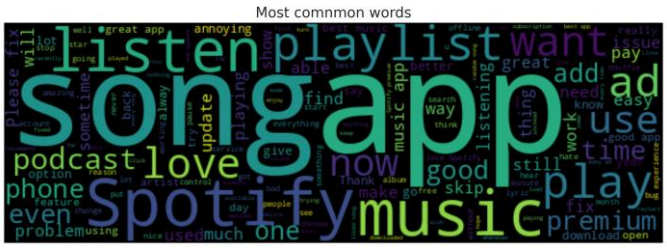

Fig.3. Histogram Chart According to Number of Stars in Rating Column

Different text encoding processes have been performed under data preprocessing and compared in terms of performance. Text encoding is a crucial process in modeling text data as numerical vectors, which allows for direct modeling of text data. This document discusses two primary text encoding methods utilized in the thesis: Count Vectorization and TF-IDF (Term Frequency-Inverse Document Frequency).

Count Vectorization is a technique that converts text documents into vectors based on word counts. Each document is represented as a vector where each word's frequency in the document determines the vector's values. This method is widely used in natural language processing (NLP) tasks such as text classification [17]. An example sentence, "The quick brown fox jumped over the lazy dog," would be represented as a vector [2, 1, 1, 1, 1, 1, 1, 1], corresponding to the word counts in the sentence [18].

TF-IDF, on the other hand, assigns weights to words based on their frequency in a document and their rarity across all documents. This method addresses the issues of over-

representation of common words in Count Vectorization, providing a more balanced measure of word importance. The TF-IDF calculation at the character level allows for more granular text analysis [19].

TF-IDF (Term Frequency-Inverse Document Frequency) is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is calculated as:

$$\mathrm{TF-IDF}(t, d, D) = \mathrm{TF}(t, d) \times \mathrm{IDF}(t, D) \quad (1)$$

where:

TF(t, d) = (Number of times term t appears in document d) / (Total number of terms in document d)

IDF(t, D) = log(Total number of documents / Number of documents containing term t)

This value increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. This helps to adjust for the fact that some words appear more frequently in general.

To address class imbalance in the dataset, the Synthetic Minority Over-sampling Technique (SMOTE) was employed. SMOTE generates synthetic instances of minority classes by interpolating between existing instances.

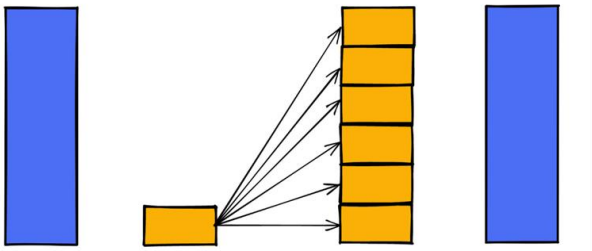

Fig.3. Upsampling of the SMOTE algorithm

In Fig.3., the yellow column in represents the minority class. Upsampling is done with the SMOTE algorithm to increase the minority class to the number of other classes. In this context, the distribution of classes before using the SMOTE algorithm and the distribution of classes after using this algorithm are shown in Fig. 4.

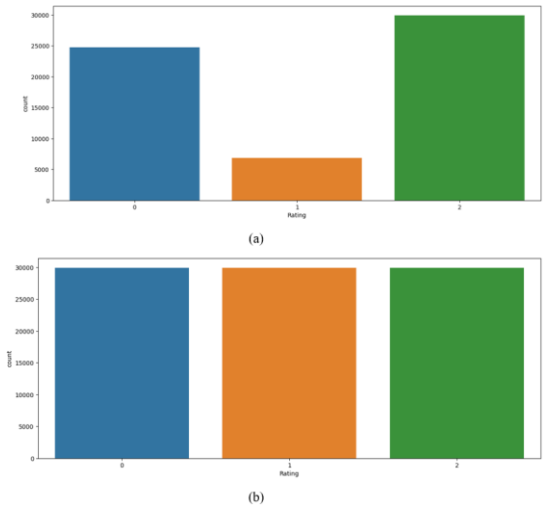

Fig.4. (a) Distribution of the Rating feature before the SMOTE process (b) Distribution of the Rating feature after the SMOTE process

After the SMOTE process, the number of elements in all classes was 29,937. Thus, the imbalance in the data set was eliminated. The machine learning methods used in this study, LSTM and the large language model BERT, are explained in the section III. Mathematical Background.

III. Mathematical Background

A. Sentiment Analysis using Machine Learning Algortihms

After the application of the Synthetic Minority Over-sampling Technique (SMOTE), the number of elements for each class in the dataset was balanced to 29,937, effectively eliminating the previously existing class imbalance. This balance allowed for a more equitable basis for training subsequent machine learning models. Following this preprocessing step, a predictive model was developed utilizing several well-established machine learning algorithms.

The algorithms employed included Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Decision Tree, Random Forest, and XGBoost. Each of these methods brings unique strengths to a machine learning task. Logistic Regression, often used for binary classification problems, models the probabilities for classification tasks by creating a linear decision boundary. KNN classifies data based on the most similar historical examples in the feature space, making it highly interpretable. SVM constructs a hyperplane in a high-dimensional space to separate different classes with a maximum margin, thus effective in non-linear separation problems. Decision Trees segment the data into branches to form a tree for prediction, which is simple to understand and interpret. Random Forest is an ensemble of Decision Trees, typically used to improve classification accuracy through voting from different trees. Lastly, XGBoost is a gradient boosting framework that uses a sequence of decision trees, where each tree corrects the errors of the previous ones, often achieving superior accuracy. The performance of these models was evaluated using classification metrics such as Accuracy, Precision, Recall, and F1-Score, providing a comprehensive measure of model efficacy across various aspects of predictive validation.

B. Sentiment Analysis using LSTM Deep Learning Algorithm

In this study, sentiment analysis was performed using machine learning methods as well as the deep learning technique known as Long Short-Term Memory (LSTM) algorithm. The deep learning method employed here was compared with other approaches in the 'Findings and Discussion' section. LSTM, a special structure developed for Recurrent Neural Networks (RNNs), is particularly effective for time series data and sequence classification problems. It was proposed by Hochreiter and Schmidhuber in 1997 to address the challenges RNNs face in learning long-term dependencies [20].

Compared to standard RNNs, LSTMs possess the capability to retain information for extended periods. This capability is facilitated by the inclusion of three distinct gate structures within LSTM cells: the forget gate, the input gate, and the output gate. These gates regulate the internal state of the cell

and the output, thereby controlling the flow of information. The fundamental equations of LSTM are articulated as follows:

$$Forget\ Gate \quad ft = \sigma(Wf \cdot [ht-1, xt] + bf) \tag{2}$$
$$Input\ Gate \quad it = \sigma(Wi \cdot [ht-1, xt] + bi) \tag{3}$$
$$Cell\ state\ update \quad C\sim t = tanh(WC \cdot [ht-1, xt] + bC) \tag{4}$$
$$Final\ cell\ state \quad Ct = ft * Ct - 1 + it * C\sim t \tag{5}$$
$$Output\ gate \quad ot = \sigma(Wo \cdot [ht-1, xt] + bo) \tag{6}$$
$$Cell\ output \quad ot = \sigma(Wo \cdot [ht-1, xt] + bo) \tag{7}$$

σ represents the sigmoid activation function, tanh the hyperbolic tangent activation function, W and b denote the weight matrices and bias vectors, respectively [21]. The LSTM architecture is presented in Fig.5.

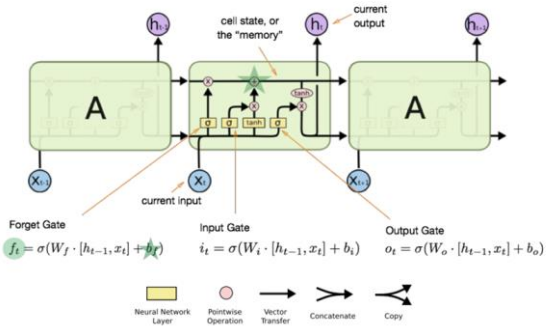

Fig.5. LSTM architecture

Fig. 5. illustrates the architecture of a Long Short-Term Memory (LSTM) network, highlighting the essential components involved in processing sequential data. An LSTM cell contains three main gates: the forget gate, input gate, and output gate, which collectively manage the flow of information through the cell. By iteratively updating the cell state, LSTMs maintain long-term dependencies, making them highly effective for tasks that involve sequential or time-dependent data, such as natural language processing and speech recognition.

C. Sentiment Analysis using BERT Large Language Model

Large Language Models (LLMs) are revolutionary artificial intelligence technologies in the field of Natural Language Processing (NLP). One such model, BERT (Bidirectional Encoder Representations from Transformers), was developed by Google in 2018 and has made significant advancements in language modeling [9]. BERT particularly utilizes a bidirectional Transformer architecture to achieve a deeper understanding of language. This model evaluates text in both left and right contexts simultaneously, offering much richer contextual representations compared to previous unidirectional models. The primary goal of BERT is to model the relationship of every word in a sentence with the words to its left and right at the same time. The model can be expressed as follows:

$$\mathrm{BERT(x)} = \mathrm{Transformer(x1{:}\,n)} \tag{8}$$

x1: n represents the sequence of words in the text. Fig.6 contains a diagram of the transformer architecture.

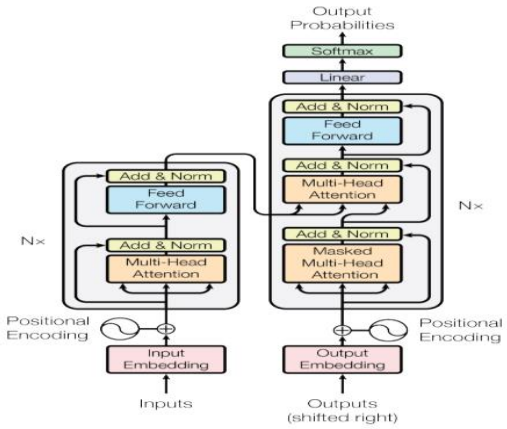

Fig.6. Transformer architecture [17]

Upon examining Fig.6, it is evident that the transformer architecture includes an Input and Positional Encoding section that converts the input sequence of words into fixed-size vectors (input embeddings). Positional encoding is used to indicate the order of words in the sequence. The Encoder part of the transformer architecture consists of six consecutive layers, each containing two sub-layers:

- *Multi-Head Attention:* Calculates the attention for each word of the input sequence towards the other words.
- *Feed-Forward Neural Network:* Applied independently to each word. The Decoder of the transformer architecture consists of six consecutive layers, each with three sub-layers:
- *Masked Multi-Head Attention:* Attention is applied to the previous words of the input sequence.
- *Encoder-Decoder Attention:* Attention is applied to the output of the Encoder.

In the output layer of the transformer architecture, the output taken from the last layer is passed through a linear and softmax layer, obtaining a probability distribution among possible words.

The BERT model, especially in NLP applications such as sentiment analysis, has demonstrated significant success. Sentiment analysis is the process of automatically detecting the emotional tone within texts, and BERT is highly effective in understanding the emotional context of texts. Once pretrained on a large dataset, the model can be fine-tuned to identify different emotional classes [23]

Introduced in 2017 by Vaswani and colleagues, the transformer architecture has sparked a revolution in deep learning models, providing a foundational approach for language models. Built upon the attention mechanism, this architecture calculates the relationships between all words in a sequence in parallel, offering a faster and more effective language modeling process. The core building block of the transformer is the attention mechanism, which calculates the relationship of each element in an input sequence with all other elements. This calculation begins with the generation of "query," "key," and "value" vectors for each component of the input sequence [22].

These vectors are computed as follows:

$$Q = XWQ \tag{9}$$
$$K = XWK \tag{10}$$

$$V = XWV \quad (11)$$

Examining Equation (9-11), Q, K, and V respectively represent the query, key, and value vectors; WQ, WK, and WV represent the learnable weight matrices. Attention scores are calculated by the dot product of query and key vectors and then normalized using the softmax function:

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \quad (12)$$

In Equation (12), d_k is the dimension of the key vectors and is used as a scaling factor to balance the calculations. Transformer models significantly reduce training time due to their parallel computing capabilities, unlike RNN and LSTM-based models. They can also model long-range dependencies more effectively, improving performance in long sequences. The transformer architecture has achieved significant success in areas such as language modeling, machine translation, and text generation [9].

IV. CALCULATIONS RESULTS

In the section entitled "III. Mathematical Background," the outcomes derived from three distinct methodologies are elucidated through tabulated representations. This section provides a comprehensive comparative analysis, evaluating each method individually as well as in relation to one another, facilitating a deeper understanding of their respective merits and interrelations within the context of the study.

To assess the impact of Text Encoding techniques on performance metrics, an initial comparison of machine learning algorithms was conducted using the Count Vectorization method. The findings from this comparison are presented in Table-2.

TABLE II
RESULTS OF MACHINE LEARNING METHODS USING COUNT VECTORIZATION

Method	Accuracy	Precision	Recall	F1-Score
CV-LR	0.72	0.73	0.72	0.72
CV-KNN	0.47	0.61	0.47	0.44
CV-DT	0.59	0.60	0.60	0.60
CV-RF	0.70	0.71	0.70	0.70
CV-SVM	0.51	0.62	0.51	0.49
CV-XGBoost	**0.78**	**0.79**	**0.78**	**0.78**

When Table 2 is examined, it is understood that the CV-XGBoost method has the most successful performance parameters among the machine learning methods used by text coding using Count Vectorization.

The comparison table for machine learning methods with TF-IDF vectorization and performance metrics is included in Table-3.

TABLE III
RESULTS OF MACHINE LEARNING METHODS USING TF-IDF

Method	Accuracy	Precision	Recall	F1-Score
TF-IDF-LR	0.68	0.68	0.68	0.68
TF-IDF-KNN	0.42	0.58	0.57	0.57
TF-IDF-DT	0.53	0.54	0.54	0.54
TF-IDF-RF	0.68	0.72	0.69	0.69
TF-IDF-SVM	0.49	0.49	0.49	0.49
TF-IDF-XGBoost	**0.76**	**0.77**	**0.76**	**0.76**

When Table 3 is examined, it is understood that the method with the highest performance metrics is the TF-IDF-XGBoost method.Within the scope of the article, sentiment analysis was conducted using the LSTM algorithm as a deep learning method. This deep learning approach was implemented using Python programming, utilizing the TensorFlow library. For data preprocessing, the NLTK library was employed. Subsequently, the dataset was divided into two parts: training and testing. In this study, 80% of the dataset was allocated for training and 20% for testing. The text data were tokenized using a Tokenizer, limited to 50,000 words (num_words=50000). Additionally, an out-of-vocabulary token '<OOV>' was specified for words do not present in the text. After the word index was created by the Tokenizer, the texts were sequenced and converted into arrays of uniform length using the pad_sequences function. This process was performed for both the training and testing sets.

Labels were one-hot encoded using the LabelBinarizer class from the sklearn library. This encoding facilitates the numerical representation of classes, enabling the model to learn these classes more easily.

Subsequently, the construction of the LSTM model was addressed. At this stage, the LSTM model was built using the following layers:

- *Embedding Layer:* Creates an 8-dimensional embedding matrix based on the total number of words.
- *Bidirectional LSTM Layer:* Contains a bidirectional LSTM layer with 16 units.
- *Dropout Layers:* 50% dropout is applied to prevent overfitting.
- *Dense Layers:* Includes a 16-unit layer with relu activation followed by an output layer with softmax activation for three classes. The summary of the established LSTM Model is presented in Fig.6.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 8)           195944

 bidirectional (Bidirection  (None, 32)                3200
 al)

 dropout (Dropout)           (None, 32)                0

 dense (Dense)               (None, 16)                528

 dropout_1 (Dropout)         (None, 16)                0

 dense_1 (Dense)             (None, 3)                 51

=================================================================
Total params: 199723 (780.17 KB)
Trainable params: 199723 (780.17 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Fig.6. LSTM Model Summary

The model was compiled using the Adam optimization algorithm (learning_rate=0.0001). The loss function selected was categorical_crossentropy. The performance of the model was evaluated using accuracy, precision, recall, and AUC metrics.

The model was trained over 25 epochs on the training data (train_padded and train_labels). Validation of the model was conducted using the test data (test_padded and test_labels). The results obtained during the model training are presented in Table 4.

In the first epoch, the training data loss was calculated as 0.9893, accuracy as 0.4848, precision as 0.5813, recall as

0.1441, and AUC as 0.6938. For the validation data, the loss was observed as 0.8762, accuracy as 0.6192, precision as 0.7014, recall as 0.3521, and AUC as 0.7888. These results indicate that at the initial stage, the model is at the beginning of its learning process and its performance needs optimization. In the second and third epochs, the losses decreased to 0.8080 and 0.6815, respectively, while accuracy values increased to 0.6777 and 0.7563. Notably, in the second epoch, the rise in validation accuracy to 0.7592 demonstrates an improvement in the model's generalization ability. In the third epoch, the drop in validation loss to 0.6114 indicates better performance on both the training and validation data.

TABLE IV
PERFORMANCE VALUES OF THE LSTM MODEL IN THE TRAINING PHASE

Epoch	Loss	Accuracy	Precision	Recall	AUC	Val_ Loss	Val_ Accuracy	Val_ Precision	Val_ Recall	Val_ AUC
1	0.989	0.4848	0.5813	0.144	0.693	0.876	0.6192	0.7014	0.3521	0.7888
2	0.808	0.6777	0.7579	0.457	0.827	0.661	0.7592	0.7902	0.712	0.8854
3	0.681	0.7563	0.7961	0.664	0.876	0.6114	0.7777	0.7954	0.7489	0.8996
4	0.643	0.7784	0.809	0.709	0.889	0.595	0.7824	0.8012	0.758	0.9047
5	0.617	0.7879	0.8181	0.731	0.897	0.587	0.7856	0.8026	0.7591	0.9073
6	0.599	0.7948	0.822	0.744	0.903	0.580	0.7886	0.8077	0.7643	0.9094
7	0.588	0.7982	0.8274	0.753	0.907	0.576	0.7908	0.8092	0.7677	0.911
8	0.574	0.8022	0.8327	0.759	0.911	0.575	0.7898	0.8095	0.766	0.9116
9	0.563	0.8058	0.8352	0.764	0.915	0.576	0.7902	0.8066	0.7683	0.9115
10	0.554	0.8092	0.8375	0.769	0.918	0.575	0.7891	0.8064	0.7664	0.9123
11	0.547	0.8112	0.8407	0.772	0.929	0.579	0.7882	0.8041	0.7655	0.911
12	0.541	0.8139	0.843	0.776	0.923	0.577	0.7877	0.8055	0.7661	0.912
13	0.530	0.817	0.8455	0.781	0.925	0.580	0.7861	0.8057	0.7641	0.9116
14	0.524	0.8191	0.8476	0.783	0.927	0.581	0.7863	0.8058	0.7622	0.9116
15	0.515	0.8209	0.8503	0.787	0.93	0.583	0.7863	0.8054	0.7615	0.9113
16	0.509	0.8221	0.8523	0.789	0.931	0.587	0.7846	0.8044	0.7625	0.9111
17	0.503	0.8241	0.8543	0.791	0.934	0.609	0.7852	0.8022	0.7632	0.9096
18	0.497	0.8243	0.8552	0.792	0.935	0.599	0.7837	0.804	0.7605	0.91
19	0.491	0.8272	0.857	0.795	0.937	0.594	0.7807	0.8009	0.7567	0.9095
20	0.486	0.8274	0.8585	0.796	0.938	0.616	0.7824	0.8011	0.7602	0.9088
21	0.482	0.8292	0.8608	0.798	0.939	0.607	0.7795	0.7997	0.754	0.9083
22	0.478	0.8292	0.8616	0.798	0.941	0.624	0.779	0.7981	0.7549	0.9077
23	0.469	0.8313	0.8639	0.801	0.943	0.6355	0.7774	0.7986	0.7535	0.9062
24	0.467	0.8317	0.8655	0.801	0.943	0.6337	0.776	0.7986	0.7529	0.9056
25	0.461	0.8329	0.8681	0.803	0.945	0.6372	0.7771	0.7972	0.7536	0.9064

In the seventh and eighth epochs, the model achieved the lowest loss values (0.5882 and 0.5743, respectively) and high accuracy values (0.7982 and 0.8022). The decrease in validation losses to 0.5769 and 0.5754 suggests that the model has found a good balance during the training process. During this period, the precision and recall values for the validation data also remained high. In the later epochs (20-25), the gap between training and validation losses narrowed. For instance, in the 25th epoch, the training loss was recorded at 0.4618, while the validation loss was 0.6372. This indicates that the model has reached a stable learning process, reducing the risk of overfitting. The high levels of accuracy, precision, and recall also imply that the model's generalization ability is maintained.

Throughout the training process, a consistent decrease in loss values and an increase in accuracy values were observed. The diminishing performance gap between the training and validation data indicates an improvement in the model's generalization ability and a successful completion of the learning process. Additionally, the increase in precision, recall, and AUC values suggests that the model's classification performance has improved. These results demonstrate that the model can be effectively used for text classification tasks.

Sentiment analysis detection was performed using a BERT (Bidirectional Encoder Representations from Transformers) based model. The BERT model was downloaded from TensorFlow Hub and used with preprocessing layers. These layers facilitate the preprocessing of text data and its processing by the BERT encoder. Text data is passed through the BERT preprocessing layer to make it suitable for the BERT encoder. The output from the BERT encoder, obtained using 'pooled_output', serves as a summarized representation of the text data. This output is then passed through deep neural network (DNN) layers for classification. Firstly, a 20% dropout was applied to help prevent overfitting. Subsequently, a dense layer with a softmax activation function was added. This layer was configured to perform a multi-class classification with three classes.

During the compilation of the model, 'categorical_crossentropy' was selected as the loss function, and the Adam optimization algorithm was used as the optimizer (learning_rate=0.0005). The performance of the model was evaluated using accuracy, precision, recall, and AUC metrics. The model was trained on the training data (X_train and train_labels) for 25 epochs with mini-batch sizes of 32. The validation data (X_test and test_labels) was used for validation. This process enabled the model to optimize its learning and performance for the text classification task. The resulting model demonstrated good performance with high accuracy, precision, recall, and AUC values on both the training and validation data. The results obtained during the model training are presented in Table 5.

Upon examining Table 5, it is observed that in the first epoch, the training data loss was calculated as 0.7804, accuracy as 0.6793, precision as 0.7276, recall as 0.5869, and AUC as 0.8348. For the validation data, the loss was observed as 0.7028, accuracy as 0.7341, precision as 0.7668, recall as 0.6856, and

AUC as 0.8688. These results indicate that at the initial stage, the model is at the beginning of its learning process and its performance needs optimization. In the second and third epochs, the losses decreased to 0.6963 and 0.6799, respectively, while accuracy values increased to 0.7238 and 0.7302. The increase in validation accuracy to 0.7479 and 0.7514 demonstrates an improvement in the model's generalization ability. In the third epoch, the drop in validation loss to 0.6457 indicates better performance on both the training and validation data.

TABLE V
PERFORMANCE VALUES OF THE LSTM MODEL IN THE TRAINING PHASE

Epoch	Loss	Accuracy	Precision	Recall	AUC	Val_ Loss	Val_ Accuracy	Val_ Precision	Val_ Recall	Val_ AUC
1	0.780	0.6793	0.7276	0.586	0.834	0.702	0.7341	0.7668	0.685	0.868
2	0.696	0.7238	0.7675	0.660	0.87	0.661	0.7479	0.7971	0.684	0.886
3	0.679	0.7302	0.7718	0.670	0.875	0.645	0.7514	0.7923	0.705	0.890
4	0.670	0.7339	0.775	0.678	0.879	0.639	0.7517	0.8	0.685	0.892
5	0.669	0.7344	0.7748	0.680	0.879	0.632	0.7538	0.7987	0.700	0.894
6	0.661	0.7364	0.7759	0.683	0.881	0.632	0.7566	0.7916	0.719	0.893
7	0.664	0.7362	0.7752	0.682	0.881	0.625	0.7588	0.8051	0.706	0.897
8	0.661	0.7384	0.7785	0.685	0.8831	0.6272	0.758	0.7942	0.7176	0.896
9	0.666	0.7334	0.774	0.6807	0.881	0.6229	0.76	0.8097	0.7	0.8992
10	0.662	0.7386	0.7771	0.685	0.882	0.622	0.7607	0.8123	0.695	0.899
11	0.661	0.7379	0.7778	0.685	0.883	0.624	0.758	0.797	0.714	0.896
12	0.663	0.7374	0.7769	0.641	0.882	0.623	0.7592	0.8048	0.707	0.898
13	0.664	0.7367	0.7759	0.683	0.882	0.621	0.7594	0.8006	0.714	0.898
14	0.6628	0.7375	0.7764	0.685	0.882	0.625	0.7544	0.805	0.686	0.897
15	0.663	0.735	0.7759	0.683	0.882	0.621	0.7612	0.7982	0.718	0.898
16	0.6601	0.7387	0.7773	0.684	0.883	0.619	0.7599	0.8098	0.702	0.899
17	0.6623	0.7375	0.7769	0.686	0.882	0.619	0.7592	0.8067	0.705	0.899
18	0.69	0.7378	0.7774	0.685	0.883	0.620	0.7565	0.8012	0.705	0.898
19	0.660	0.7368	0.7769	0.685	0.883	0.618	0.7609	0.8121	0.703	0.900
20	0.669	0.7372	0.7772	0.683	0.883	0.622	0.7579	0.7994	0.712	0.897
21	0.661	0.7364	0.7756	0.684	0.882	0.620	0.7594	0.8046	0.704	0.898
22	0.662	0.737	0.7765	0.685	0.882	0.620	0.7591	0.8062	0.703	0.898
23	0.662	0.7376	0.7751	0.683	0.882	0.617	0.7605	0.8057	0.711	0.899
24	0.663	0.7349	0.7741	0.681	0.882	0.62	0.7593	0.8082	0.705	0.899
25	0.662	0.7376	0.7767	0.685	0.882	0.624	0.7605	0.7903	0.725	0.896

In the seventh and eighth epochs, the model achieved the lowest loss values (0.6646 and 0.6610, respectively) and high accuracy values (0.7362 and 0.7384). The decrease in validation losses to 0.6256 and 0.6272 suggests that the model has found a good balance during the training process. During this period, the precision and recall values for the validation data also remained high. In the later epochs (20-25), the gap between training and validation losses narrowed. For instance, in the 25th epoch, the training loss was recorded at 0.6629, while the validation loss was 0.6249. This indicates that the model has reached a stable learning process, reducing the risk of overfitting. The high levels of accuracy, precision, and recall also imply that the model's generalization ability is maintained.

Throughout the training process, a consistent decrease in loss values and an increase in accuracy values were observed. The diminishing performance gap between the training and validation data indicates an improvement in the model's generalization ability and a successful completion of the learning process. Additionally, the increase in precision, recall, and AUC values suggests that the model's classification performance has improved.

V. CONCLUSION

In the studies conducted within the scope of this research, the first step involved comparing machine learning methods using text encoding methods, namely Count Vectorization and TF-IDF, to perform sentiment analysis on Spotify comments. In the second step, sentiment analysis was conducted by optimizing the hyperparameters of the LSTM deep learning method. Finally, the recent focus on large language models was addressed, and sentiment analysis was performed using the BERT model. These methods were compared both individually and against each other to determine the most suitable approach for the specific problem.

To evaluate the performance of the methods used in this study, specific performance metrics were employed. The analyses and results obtained based on these metrics are summarized below. The performance results obtained using Count Vectorization (CV) and TF-IDF vectorization in the sentiment analysis conducted with machine learning methods are presented in Table 5 and Table 6. The CV-XGBoost method, with the highest performance values, stood out as the most successful method with 78% accuracy, 79% precision, 78% recall, and 78% F1-score. Similarly, the TF-IDF-XGBoost method also demonstrated high performance, achieving 76% accuracy, 77% precision, 76% recall, and 76% F1-score.

Sentiment analysis was performed using the LSTM algorithm as the deep learning method. Throughout the training process, a consistent decrease in loss values and a consistent increase in accuracy values were observed. The lowest loss values for the LSTM model were recorded as 0.5882 and 0.5743 in the seventh and eighth epochs, respectively, while the highest accuracy values reached 80.22%. These results indicate an

improvement in the model's generalization ability and classification performance.

In the sentiment analysis conducted using the BERT-based model, the model exhibited high performance in terms of accuracy, precision, recall, and AUC metrics. The model was effective in understanding the emotional context of the texts, and a consistent increase in accuracy values was observed. Throughout the 25 epochs of training, the accuracy and precision values remained high, demonstrating that the model maintained its generalization ability. The BERT language model achieved 73% accuracy.

When comparing the methods used in this study, the LSTM deep learning method emerged as the one with the highest performance. However, it is thought that fine-tuning large language models, which are a current focus of intense research, with localized information specific to the subject can achieve higher performance. In future studies, pre-trained language models can be customized and used for specific topics. Future work could focus on several key areas to enhance the scope and impact of this study. One promising direction is to AI-based model specifically for sentiment analysis within the music streaming domain, potentially improving its accuracy in detecting complex emotional nuances in user reviews. Additionally, incorporating data from multiple music streaming platforms could broaden the analysis, allowing for a more comprehensive understanding of user sentiment across diverse user bases and feedback types. Furthermore, exploring advanced model optimization techniques to balance computational efficiency with performance could facilitate the deployment of these models in real-time applications. By expanding the dataset and refining model parameters, future research could provide more robust insights into user satisfaction and emotional engagement in digital music platforms.

REFERENCES

[1] Khan, A. W., & Mishra, A. (2023). AI credibility and consumer-AI experiences: a conceptual framework. Journal of Service Theory and Practice.

[2] Miragoli, M. (2024). Conformism, Ignorance & Injustice: AI as a Tool of Epistemic Oppression.*Episteme*.

[3] Zheng, J. (2024). Music Sentiment Analysis and its Application in Music Therapy Based on AI Technology. *International Journal of Maritime Engineering*

[4] Chen, Y., & Sun, Y. (2024). The Usage of Artificial Intelligence Technology in Music Education System Under Deep Learning. *IEEE Access, 12*, 130546-130556

[5] Pires, I. M., Zafar, S., Iqbal, K., Sharif, M., Shah, Y. A., Khalil, A., Irfan, M. A., & Rosak-Szyrocka, J. (2024). Attention-aware with stacked embedding for sentiment analysis of student feedback through deep learning techniques. *PeerJ Computer Science, 10*.

[6] Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. Foundations and Trends® in Information Retrieval, 2(1–2), 1–135. DOI: 10.1561/1500000011

[7] Liu, B. (2012). Sentiment Analysis and Opinion Mining. Synthesis Lectures on Human Language Technologies, 5(1), 1–167. DOI: 10.2200/S00416ED1V01Y201204HLT016.

[8] Jain, N., Kumar, S., & Fernandes, S. L. (2019). Machine Learning Techniques for Sentiment Analysis: A Review. In: Das, H., & Pattnaik, P. (Eds.), Computational Intelligence in Data Mining. Advances in Intelligent Systems and Computing, vol 711. Springer, Singapore. DOI: 10.1007/978-981-13-1810-8_19

[9] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805. DOI: 10.48550/arXiv.1810.04805

[10] Zhang, L., Wang, S., & Liu, B. (2020). Deep Learning for Sentiment Analysis: A Survey. IEEE Transactions on Affective Computing, 12(2), 314-332. DOI: 10.1109/TAFFC.2020.2984262

[11] Jhanji, R. (2024). Emotion Analysis from Music Using LSTM Models and Mel Spectrograms. Journal of Music Data Science, 8(1), 123-134. DOI: 10.1016/j.jmds.2024.01.001

[12] Martin-Gomez, A., Garcia, J., & Lopez, V. (2018). Comparative Study of Multi-Label Classification Algorithms for Emotion Recognition. Journal of Machine Learning Research, 19(45), 1-25.

[13] Tilloo, R., Patel, S., & Shah, A. (2021). Sentiment Analysis of Amazon Musical Instrument Reviews Using CNN and NLP Techniques. International Journal of Data Science and Analytics, 12(3), 234-245. DOI: 10.1007/s41060-021-00259-9

[14] Website. [Online]. Available: https://www.kaggle.com/datasets/mfaaris/spotify-appreviews-2022

[15] Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.

[16] Hovy, E., & Lavid, J. (2010). Towards a 'science' of corpus annotation: A new methodological challenge for corpus linguistics. International Journal of Translation, 22(1), 13-36

[17] Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press

[18] Rajaraman, A., & Ullman, J. D. (2011). Mining of Massive Datasets. Cambridge University Press

[19] E. Cambria, B. Schuller, Y. Xia, and C. Havasi, "New Avenues in Opinion Mining and Sentiment Analysis," Intelligent Systems, IEEE , vol.28, no.2, pp. 15-21, 2013

[20] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780

[21] Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. Neural Computation, 12(10), 2451-2471.

[22] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems

[23] Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune BERT for text classification?. China National Conference on Chinese Computational Linguistics.

BIOGRAPHIES

Murat ŞİMŞEK completed his undergraduate and graduate education at Kırıkkale University, Department of Electrical and Electronics Engineering. He completed his master's degree on the effect of super-resolution methods on increasing the resolution of hyperspectral images. He completed his doctorate at Ankara University, Department of Electrical and Electronics Engineering. He completed his doctorate on real-time target detection in hyperspectral images. Throughout his career, he has worked in many different positions as an electrical and electronics engineer, electronic technologies specialist, artificial intelligence specialist and project manager. He has academic studies on image processing and artificial intelligence. He is head of Artificial Intelligence Engineering Department of Ostim Technical University.

Buğra Kağan Kayhan completed his undergraduate at Ufuk University, Department of Management Information Systems. He completed his master's degree on Emotion and content analysis of music data using machine learning methods at Ostim Technical University Software Engineering Department. Throughout his career, he has worked in many different positions as an software technologies specialist, artificial intelligence specialist.