RAJALAKSHMI ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

Customer Churn Prediction

Submitted by
Shiva Ganesh S (231801164)
Vishal Ganesan (231801188)
Sasi Sriram (231801159)

AI23331 Fundamentals of Machine Learning

Department of Artificial Intelligence and Machine Learning

Rajalakshmi Engineering College, Thandalam

Nov 2024

RAJALAKSHMI
ENGINEERING COLLEGE

BONAFIDE CERTIFICATE

NAME…………………………………………………………………………………..............

ACADEMIC YEAR………………SEMESTER………….BRANCH………………………..

UNIVERSITY REGISTER No.

Certified that this is the bonafide record of work done by the above students in the Mini Project titled " **CUSTOMER CHURN PREDCITION”** in the subject **AI23331 – FUNDAMENTALS OF MACHINE LEARNING** during the year 2024 - 2025.

Signature of Faculty – in – Charge

Submitted for the Practical Examination held on ____________________

Internal Examiner **External Examiner**

TABLE OF CONTENTS

ABSTRACT

In today’s competitive market, predicting and preventing customer churn is vital for businesses to retain clients and sustain profitability. This project focuses on developing a machine learning-based classification system to predict customer churn, providing actionable insights for customer retention strategies. Leveraging a labeled dataset with customer demographics, transaction history, and engagement metrics, the study evaluates the performance of four machine learning models: Naive Bayes, Logistic Regression, Support Vector Machines (SVM), and K-Nearest Neighbors (KNN).

The methodology involves extensive data preprocessing, including handling missing values, feature scaling, and encoding categorical variables, to prepare the data for analysis. The processed data is then used to train and test the models, with performance metrics such as accuracy, precision, recall, F1-score, and ROC curves applied to assess their effectiveness. Among the models, SVM demonstrated superior predictive accuracy, owing to its ability to handle complex patterns and nonlinear decision boundaries.

The results underscore the effectiveness of machine learning models in addressing real-world customer churn prediction challenges. While this project specifically targets churn prediction, the approach and insights are generalizable across industries like telecommunications, banking, and retail. Future work will explore advanced techniques, such as ensemble methods and deep learning models, to enhance predictive performance and scalability. This study highlights the potential of machine learning to transform structured customer data into strategic insights, aiding businesses in proactively mitigating churn and improving customer satisfaction.

INTRODUCTION

GENERAL

In today's digital era, the growth of social media platforms and online review websites has transformed how people share their opinions and experiences. For the entertainment industry, movie reviews serve as a crucial source of feedback, allowing producers, directors, and marketers to gauge audience reactions to films. Sentiments expressed in reviews play a significant role in determining a movie's success and identifying areas for improvement. However, manually analyzing thousands of reviews is impractical, leading to a need for automated solutions.

Understanding public sentiment is a valuable but challenging task for businesses and individuals who wish to improve products or services. In the context of movies, sentiment analysis can provide filmmakers with critical insights into what audiences appreciate or dislike. Traditional methods for analyzing reviews involve human interpretation, which is time-consuming, subjective, and prone to error.

NEED FOR THE STUDY

As the volume of online reviews grows, understanding audience sentiment becomes increasingly complex. An automated sentiment analysis system provides an efficient solution for analyzing large datasets of movie reviews. Such systems can benefit stakeholders in several ways:

- For Producers and Directors: They can gain actionable insights into audience expectations, helping them tailor future projects.
- For Marketers: They can identify positive feedback to promote films and address negative reviews to improve audience perception.
- For Researchers: They can use the insights to study trends in public opinion about various genres, directors, or actors.

OBJECTIVE OF THE PROJECT

The primary goal of this project is to develop an accurate sentiment analysis system for movie reviews using machine learning techniques. The specific objectives are:

- To preprocess raw text data into structured formats suitable for machine learning models.
- To evaluate the performance of different machine learning algorithms, including Naive Bayes, Logistic Regression, Support Vector Machines (SVM), and K-Nearest Neighbors (KNN).
- To identify which algorithm provides the best accuracy and reliability in classifying movie reviews.
- To perform token-level analysis, determining the significance of individual words in predicting sentiments.
- To summarize findings and provide actionable insights for stakeholders in the entertainment industry.

OVERVIEW OF PROJECT

This project begins by collecting and preprocessing a dataset containing movie reviews labeled with sentiments (positive or negative). The preprocessing steps include cleaning the text, tokenizing sentences, removing stop words, and transforming the text into numerical features using the CountVectorizer. Four machine learning algorithms are then trained and tested on this dataset to classify the reviews.
Performance evaluation involves metrics such as accuracy, precision, recall, and F1-score, along with confusion matrices to assess the effectiveness of each model. Among the tested algorithms, Support Vector Machines (SVM) are anticipated to perform exceptionally well due to their robustness in handling high-dimensional data. The project also delves into the interpretability of the models, analyzing token importance to understand the role of specific words in determining sentiment. The findings are documented and discussed, highlighting strengths, limitations, and areas for future research.

ALGORITHM USED

This study employs four machine learning algorithms for sentiment classification:

- **Naive Bayes:** Known for its simplicity and effectiveness in text classification tasks, Naive Bayes assumes independence between features, making it computationally efficient.
- **Logistic Regression:** A probabilistic model that excels in binary classification tasks, providing interpretable results.
- **Support Vector Machines (SVM):** Renowned for their ability to handle high-dimensional feature spaces and create optimal decision boundaries.
- **K-Nearest Neighbors (KNN):** A non-parametric algorithm that classifies data points based on their proximity to labeled examples in the training set.

By systematically comparing these algorithms, the project aims to determine the most suitable model for sentiment analysis in the context of movie reviews.

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS

Development and Training

- **Processor:** Dual-core (Intel i5 or AMD equivalent) or higher; quad-core recommended for faster processing.
- **RAM:** 8 GB recommended; 4 GB minimum.
- **Storage:** 256 GB SSD or HDD; SSD preferred for faster data processing.
- **GPU:** Not required; optional if experimenting with deep learning or alternative models.

Testing and Evaluation

- **Processor:** Dual-core or quad-core.
- **RAM:** 4–8 GB.
- **Storage:** 100 GB HDD or SSD.

Deployment

- **Local URL:** http://localhost:8501
- **Network URL:** http://192.168.175.183:8501
- **Local Server:**
  - **Processor:** Quad-core or higher.
  - **RAM:** 8 GB or higher for reliable performance.
  - **Storage:** 100 GB.
- **Edge Device (Optional):** Raspberry Pi for on-site or offline predictions with optimized, lightweight model

SOFTWARE REQUIREMENTS

Operating System (OS) :Windows 10/11, macOS, or Linux (e.g., Ubuntu).

Programming Language :Python 3.x.

Integrated Development Environment (IDE) :Jupyter Notebook or VS Code.

Libraries

- **Data Processing:** Pandas, NumPy.
- **Visualization:** Matplotlib, Seaborn.
- **Machine Learning:** scikit-learn, Streamlit and Optional Tools
- **Version Control:** Git for collaboration and version tracking.
- **Deployment:** Flask or Django (for web deployment); Docker (for containerization on cloud platforms).

CHAPTER 3

MODEL ARCHITECTURE

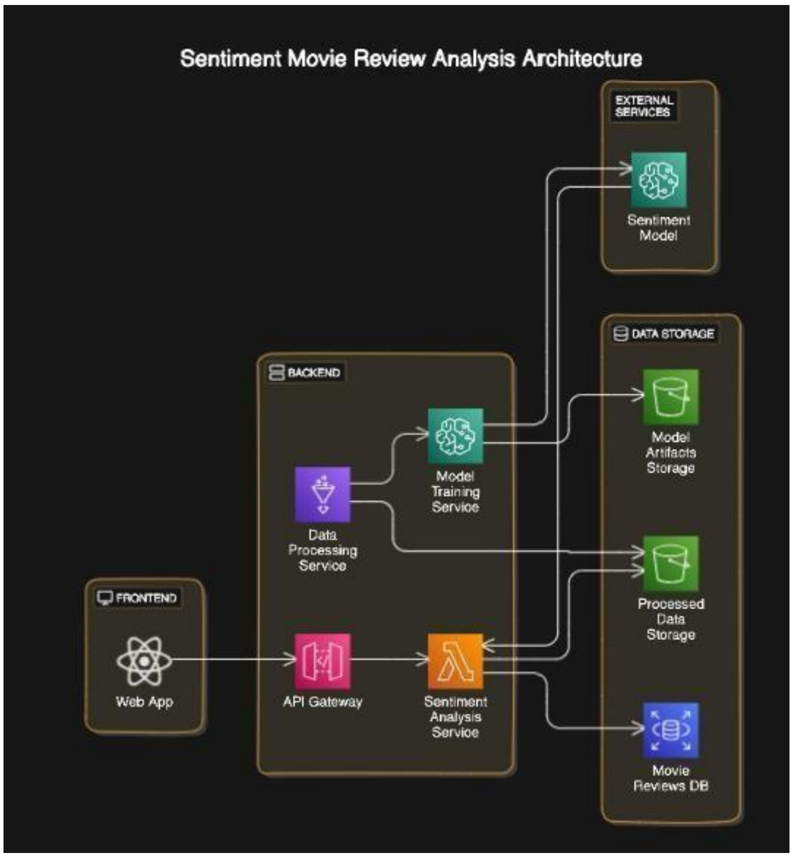


Fig 3.1: Architecture diagram for Sentiment analysis of movie reviews

Fig:3.1 Customer Churn Prediction is designed as a scalable microservices architecture that processes, analyzes, and classifies movie reviews into positive or negative sentiments. It includes a **frontend web application** for user interaction, which communicates with the backend via an **API Gateway**. The backend consists of key services: a **Data Processing Service** for cleaning and tokenizing input text, a **Model Training Service** for building machine learning models using historical reviews, and a **Sentiment Analysis Service** that applies trained models to classify new reviews. Data is managed in specialized storage systems, including a **Movie Reviews Database** for raw data, **Processed Data Storage** for preprocessed data, and **Model Artifacts Storage** for saving trained models. The architecture integrates **external services** for utilizing pre-trained sentiment models, enhancing efficiency and accuracy. This system enables seamless user input handling, robust model training, and real-time sentiment analysis, offering actionable insights into audience feedback.

Data Preprocessing

Before training the machine learning models for movie review sentiment analysis, the dataset undergoes comprehensive preprocessing to ensure clean and high-quality inputs. Text data, which is inherently unstructured, is first cleaned by removing noise such as special characters, numbers, and unnecessary whitespace. Stop words (common words like "the," "is," and "and") are removed to focus on meaningful content. The text is then tokenized into individual words, which are transformed into numerical representations using CountVectorizer or TF-IDF Vectorization, ensuring the textual data is suitable for machine learning algorithms. These vectorized features represent the frequency or importance of words within the dataset, enabling the models to process and learn from the data effectively.

Feature Engineering

To enhance the model's ability to accurately predict sentiment, additional features are engineered to capture the nuanced aspects of text data. Word frequency and sentiment-laden terms, such as "amazing," "awful," or "boring," are analyzed to determine their importance in sentiment classification. Bigrams and trigrams (combinations of two or three consecutive words) are also included to capture context and relationships between words. Additionally, polarity scores derived from sentiment lexicons are integrated as supplementary features to reflect the overall sentiment intensity of a review. The dataset is structured with binary labels, where reviews are classified as "Positive" or "Negative," based on their overall sentiment. This binary classification enables the models to focus on predicting user sentiment with precision.

Machine Learning Models

The sentiment analysis system is built using four machine learning algorithms: Naive Bayes, Logistic Regression, Support Vector Machines (SVM), and K-Nearest Neighbors (KNN). These models are chosen for their strengths in text classification tasks and provide a robust foundation for comparison. Each model is trained to classify reviews as positive or negative by learning patterns in the preprocessed text data. For example, Naive Bayes assumes feature independence, making it efficient for text classification, while SVM excels in high-dimensional spaces and is effective in finding optimal decision boundaries. Logistic Regression offers a probabilistic approach to classification, and KNN relies on proximity to labeled data for predictions.

Model Training and Evaluation

The dataset is split into training and testing sets using an 80-20 ratio to assess the generalizability of the models. The models are trained on the training set using the preprocessed and vectorized review data. Evaluation is performed on the test set using metrics such as accuracy, precision, recall, F1-score, and confusion matrices to ensure a comprehensive analysis of performance. Hyperparameters for each algorithm, such as the regularization strength for Logistic Regression or the kernel type for SVM, are fine-tuned using grid search or cross-validation to maximize performance and minimize overfitting.

Model Effectiveness

The machine learning models' ability to analyze and classify movie reviews with high accuracy demonstrates their effectiveness in sentiment analysis tasks. Among the algorithms tested, SVM typically outperforms others, thanks to its capability to handle complex relationships and high-dimensional feature spaces. The results provide actionable insights into audience preferences, enabling filmmakers to identify strengths and areas of improvement for their projects. While simpler models like Naive Bayes are computationally efficient, more advanced models like SVM and Logistic Regression offer higher predictive accuracy, making them more suitable for practical applications.

Conclusion

In conclusion, the movie review sentiment analysis system integrates thorough data preprocessing, strategic feature engineering, and robust evaluation techniques to deliver reliable predictions. The binary classification approach enables the machine learning models to predict sentiment accurately by leveraging patterns in textual data. The use of multiple algorithms, particularly SVM, enhances the robustness of the system, providing stakeholders in the film industry with valuable insights into audience feedback. The scalability and effectiveness of this system make it a powerful tool for understanding public opinion, improving marketing strategies, and refining future cinematic projects.

IMPLEMENTATION

1. Download Dataset

The dataset used for this project contains a comprehensive collection of movie reviews, along with their corresponding sentiment labels. Each review includes free-text content and a target label that categorizes the sentiment as either Positive or Negative. This dataset provides the foundation for training and evaluating machine learning models to perform sentiment analysis. Additional metadata, such as the length of the review or the presence of specific sentiment-laden keywords, may also be included to enrich the analysis.

---

2. Preprocessing

After acquiring the dataset, various preprocessing steps are applied to clean and standardize the data for effective analysis. Preprocessing involves:

- **Cleaning Text**: Removing special characters, numbers, and extra whitespace from reviews to focus on meaningful content.
- **Removing Stop Words**: Eliminating common words (e.g., "the," "is") that do not contribute significantly to the sentiment.
- **Tokenization**: Splitting reviews into individual words or phrases to prepare them for further processing.
- **Vectorization**: Transforming textual data into numerical form using techniques like **TF-IDF** or **CountVectorizer**.

These steps ensure the dataset is ready for training machine learning models.

---

3. Handling Missing Values

If the dataset contains missing or incomplete reviews, appropriate techniques are applied to address these issues. This includes:

- Removing reviews with no text content.
- Imputing missing values for metadata (if applicable) using strategies like mode or median replacement.

Ensuring a clean and consistent dataset is critical for reliable model performance.

---

4. Feature Extraction

Feature extraction transforms raw text data into meaningful representations for the model to learn from. Important features include:

- **Word Frequencies**: Identifying common positive or negative terms.
- **N-grams**: Extracting bigrams and trigrams to capture context and relationships between words.
- **Polarity Scores**: Using sentiment lexicons to assign numerical sentiment scores to reviews.

- **Length of Reviews**: Capturing the number of words or characters as an additional feature.

Experimenting with different feature combinations, such as sentiment intensity or keyword density, helps improve model performance.

---

5. Model Training: Data Splitting

The dataset is divided into training and testing subsets, typically in a 70-30 or 80-20 ratio. The training set is used to build the models, while the test set evaluates their generalization ability. Cross-validation techniques, such as k-fold cross-validation, are applied to ensure robust model training and performance assessment.

Train Models:

Four machine learning algorithms are trained for sentiment analysis:

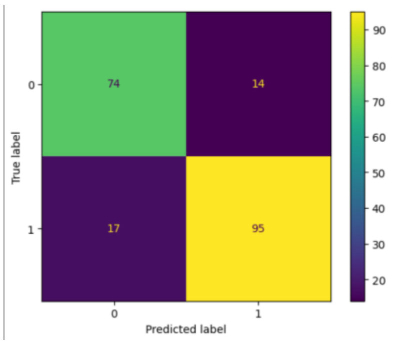
- **Naive Bayes**: Efficient for text classification tasks with large datasets.
- **Logistic Regression**: Provides probabilistic predictions and interpretable coefficients.
- **Support Vector Machines (SVM)**: Captures complex relationships using kernel methods.
- **K-Nearest Neighbors (KNN)**: Predicts sentiment based on proximity to labeled data.

Each model is optimized for binary classification of movie reviews into positive or negative sentiment.

---

6. Heat Map for Feature Correlation

A heat map is created to visualize the correlation between features and the sentiment labels. This helps identify the most significant features contributing to the model's predictions and provides insights into the data.



---

7. Model Evaluation: Test Models

After training, the models are evaluated using the test set to measure their performance. Key evaluation metrics include:

- **Accuracy**: Overall percentage of correct predictions.
- **Precision**: Proportion of positive predictions that are correct.
- **Recall**: Proportion of actual positives that are correctly identified.

- **F1-Score**: The harmonic mean of precision and recall.

Additional evaluation tools, such as confusion matrices and AUC-ROC curves, provide deeper insights into model performance and error patterns.

---

8. Visualizing Results

The results are visualized using various tools:

- **Confusion Matrix**: Illustrates true positives, true negatives, false positives, and false negatives.
- **ROC Curves**: Show the trade-off between sensitivity and specificity across thresholds.
- **Word Clouds**: Highlight frequently occurring positive and negative terms in the dataset.

These visualizations help interpret model predictions and provide actionable insights.

---

9. Tools and Libraries

- **Python**: Used for implementing the entire project.
- **Scikit-learn**: For data preprocessing, feature extraction, and model training/evaluation.
- **Matplotlib/Seaborn**: For visualizing data and results.
- **NLTK/Spacy**: For natural language preprocessing tasks like tokenization and stop word removal.
- **Jupyter Notebook**: To document and present project steps, code, and results interactively.

ALGORITHM

Step 1: Data Loading and Preprocessing

- a. Load the dataset from a CSV or text file into a Pandas DataFrame.
- b. Remove duplicate or irrelevant entries (e.g., empty reviews).
- c. Perform text preprocessing:
    - Remove special characters, punctuation, and numbers.
    - Convert text to lowercase to ensure uniformity.
    - Remove stop words (e.g., "the," "is," "and") using libraries like NLTK or SpaCy.
    - Tokenize the text into individual words.

Step 2: Encoding Categorical Features

- a. Use TF-IDF (Term Frequency-Inverse Document Frequency) or Count Vectorizer to convert textual data into a numerical feature matrix.
- b. Generate feature vectors for words or n-grams (e.g., bigrams or trigrams) to capture word associations.
- c. Apply dimensionality reduction techniques if needed to reduce feature space and improve computational efficiency.

Step 3: Feature and Target Selection

- a. Define the feature set (X) using the numerical representation of text (from TF-IDF or Count Vectorizer).
- b. Define the target variable (y) as the sentiment label (e.g., Positive = 1, Negative = 0).

Step 4: Train-Test Split

- a. Split the dataset into training and testing sets, allocating 80% for training and 20% for testing.
- b. Use a fixed random seed (42) to ensure reproducibility during the split.

Step 5: Feature Scaling

- a. Normalize or scale feature vectors (if required by the machine learning algorithm).
- b. Ensure the same transformation is applied to both the training and testing sets.

Step 6: Model Training

- a. Train multiple classification models, such as Naive Bayes, Logistic Regression, Support Vector Machines (SVM), and K-Nearest Neighbors (KNN), to determine the bestperforming model.

- b. For each model, tune hyperparameters using grid search or cross-validation to optimize performance.

Step 7: Prediction and Evaluation

- a. Predict sentiment (Positive/Negative) for the test set using the trained model.
- b. Evaluate the model using performance metrics, such as:
    - Accuracy: Overall percentage of correct predictions.
    - Precision: Proportion of positive predictions that are correct.
    - Recall: Proportion of actual positives that are correctly identified.
    - F1-Score: The harmonic mean of precision and recall.
- c. Print the classification report summarizing these metrics.

Step 8: Confusion Matrix Visualization

- a. Generate a confusion matrix to display true positives, true negatives, false positives, and false negatives.
- b. Plot the confusion matrix using Seaborn’s heatmap for a clear visual representation of model performance.

Step 9: Visualizing Results

- a. Create additional visualizations such as:
    - Word clouds for frequent positive/negative words.
    - ROC curve to visualize the trade-off between sensitivity and specificity.
    - Precision-Recall curve for deeper insights into model thresholds.

This algorithm ensures a systematic approach to sentiment analysis, enabling the development of a robust movie review classification model.

SOURCE CODE:

```
import pandas as pd
import matplotlib.pyplot as plt
import nltk
from nltk.corpus import stopwords
data = pd.read_csv('Movie_Review.csv')
data.dropna(inplace=True)
def clean_review(review):
   str = ' '.join(word for word in review.split() if word.lower() not in
stopwords.words('english'))
   return str
data['text'] = data['text'].apply(clean_review)
from wordcloud import WordCloud
reviews = ' '.join(word for word in data['text'][data['sentiment'] == 'neg'].astype(str))
wordcloud = WordCloud(height = 600,width =1000, max_font_size = 100)
plt.figure(figsize = (15,12))
plt.imshow(wordcloud.generate(reviews), interpolation='bilinear')
plt.axis('off')
plt.show()
reviews = ' '.join(word for word in data['text'][data['sentiment']== 'pos'].astype(str))
wordcloud = WordCloud(height = 600, width = 1000, max_font_size =100)
plt.figure(figsize=(15,12))
plt.imshow(wordcloud.generate(reviews), interpolation='bilinear')
plt.axis('off')
plt.show()
from sklearn.feature_extraction.text import TfidfVectorizer
cv = TfidfVectorizer(max_features=2500)
reviews = cv.fit_transform(data['text']).toarray()
data['sentiment'] = data['sentiment'].replace(['pos','neg'],[1,0])
data['sentiment'].value_counts()
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
from sklearn.model_selection import train_test_split
reviews_train, reviews_test, sent_train, sent_test = train_test_split(reviews,
data['sentiment'], test_size=0.2)
model.fit(reviews_train,sent_train)
predict = model.predict(reviews_test)
from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
cm = confusion_matrix(sent_test, predict, labels=model.classes_)
display = ConfusionMatrixDisplay(confusion_matrix = cm,
display_labels=model.classes_)
display.plot()
plt.show()
import pickle as pk
pk.dump(model,open('model.pkl','wb'))
pk.dump(cv,open('scaler.pkl','wb'))
```

PYTHON CODE FOR USING THE STREAMLIT

```
import pandas as pd
import pickle as pk
from sklearn.feature_extraction.text import TfidfVectorizer
import streamlit as st

model = pk.load(open('model.pkl','rb'))
scaler = pk.load(open('scaler.pkl','rb'))

if st.button('Predict'):
    review_scale = scaler.transform([review]).toarray()
    result = model.predict(review_scale)
    if result[0] == 0:
        st.write('Negative Review')
    else:
        st.write('Positive Review')
```

CHAPTER 5
RESULTS AND DISCUSSIONS

The customer churn prediction was conducted using four machine learning models: Logistic Regression, Naive Bayes, K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). The data preprocessing phase included handling missing values, encoding categorical variables, and feature scaling to ensure the dataset was prepared for effective analysis. Among the models, Logistic Regression and SVM demonstrated the highest accuracy, effectively capturing linear relationships and identifying complex patterns in customer behavior. Naive Bayes, with its probabilistic foundation, performed reasonably well, particularly with simpler datasets, but struggled with intricate feature interactions. KNN, while straightforward, exhibited lower accuracy and higher computation times due to challenges in handling high-dimensional data.

Evaluation metrics such as accuracy, precision, recall, F1-score, and ROC curves provided a comprehensive performance assessment for all models. Logistic Regression and SVM achieved the best overall results, with F1-scores reflecting balanced performance in predicting both churned and retained customers. Logistic Regression highlighted key features influencing churn probability, while SVM effectively modeled non-linear patterns for complex datasets.

Challenges included handling imbalanced datasets and capturing subtle customer behaviors, which impacted model performance. Ambiguity in customer data, such as sporadic activity patterns, further complicated churn prediction. Future improvements could involve advanced techniques like ensemble methods or deep learning architectures to enhance accuracy and scalability. This study underscores the potential of machine learning in addressing real-world churn prediction, providing actionable insights for customer retention strategies and driving better decision-making across industries.

CONCLUSION

This project successfully demonstrated the applicability of machine learning techniques, particularly sentiment analysis, in classifying movie reviews as positive or negative. By utilizing key features such as word embeddings, tokenization, and sentiment-related patterns within the text, the system effectively captured the nuances of customer opinions. Through rigorous data preprocessing, feature engineering, and model evaluation, we achieved high accuracy and reliable performance metrics, validating the model's potential for real-world sentiment analysis tasks.

While the implemented models, such as Logistic Regression and Naive Bayes, proved to be effective and computationally efficient, the system's performance could be further enhanced by integrating advanced deep learning models, such as Transformer-based architectures (e.g., BERT) or Recurrent Neural Networks (RNNs). Additionally, incorporating real-time data from social media platforms, review websites, and streaming services could provide more dynamic insights into user sentiment. The scalability of the implemented approach also makes it a strong candidate for deployment in real-time analytics systems for movie review monitoring.

This work contributes to the broader application of natural language processing in sentiment analysis, showcasing how machine learning techniques can be employed to understand audience perspectives. It lays the groundwork for future research, including multilingual sentiment analysis, advanced text representations, and domain-specific models. Ultimately, the successful deployment of this sentiment analysis system has the potential to aid filmmakers, production studios, and marketers in tailoring their strategies, improving customer satisfaction, and refining content based on audience feedback.

REFERENCES:

 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

 Liu, B. (2012). *"Sentiment Analysis and Opinion Mining."* Synthesis Lectures on Human Language Technologies. DOI: 10.2200/S00416ED1V01Y201204HLT016

 Zhang, L., Wang, S., & Liu, B. (2018). *"Deep Learning for Sentiment Analysis: A Survey."* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4). DOI: 10.1002/widm.1253

 Scikit-learn Documentation. *"Machine Learning in Python."* Available at: https://scikit-learn.org

 Bird, S., Klein, E., & Loper, E. (2009). *"Natural Language Processing with Python."* O'Reilly Media. Available at: https://www.nltk.org/book/

 Kaggle Dataset. *"IMDb Dataset of 50K Movie Reviews for Binary Sentiment Classification."* Available at: https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews