TOPIC NAME: - AUTOMATICALLY COLLECT AND SUMMARIZE USER COMMENTS FROM GITHUB AND THE GOOGLE PLAY STORE. DEVELOP A CLASSIFIER TO IDENTIFY NEGATIVE COMMENTS

**Table of Contents**

[Introduction 3](#_Toc184244812)

[Motivation 3](#_Toc184244813)

[Problem Statement 3](#_Toc184244814)

[Proposed Work 3](#_Toc184244815)

[Techniques 3](#_Toc184244816)

[Implementation Details 4](#_Toc184244817)

[Data Collection 4](#_Toc184244818)

[Data Preprocessing 4](#_Toc184244819)

[Splitting Data 6](#_Toc184244820)

[Model Development 7](#_Toc184244821)

[Evaluation Method 8](#_Toc184244822)

[Implementation details of GUI 8](#_Toc184244823)

[Results 9](#_Toc184244824)

[1. Random Forest Classifier 10](#_Toc184244825)

[2. Logistic Regression Classifier 11](#_Toc184244826)

[3. Support Vector Classifier (SVC) 12](#_Toc184244827)

[Summary of Model Performance 13](#_Toc184244828)

[User Feedback Integration 14](#_Toc184244829)

[Conclusion 14](#_Toc184244830)

[References 15](#_Toc184244831)

# Introduction

## Motivation

User feedback is an essential part of software application development and improvement, but the sheer volume of comments across platforms like GitHub and the Google Play Store can overwhelm developers and prove hard to sift through to identify meaningful insights. Manually reviewing all these comments would be incredibly time-consuming and incidentally susceptible to human error. The collection and analysis of user comments can be automated to allow developers to quickly identify critical issues and prioritize their responses to enhance user satisfaction and software quality.

## Problem Statement

This task would require sorting through a plethora of comments made by users to discern actionable feedback, mainly negative sentiment. The purpose of this project would be to automate this process to collect, summarize, and classify comments from GitHub and the Google Play Store. Negative feedback should be identified in the shortest possible time for the developer to focus on crucial points without sifting through loads of reviews (Mata, Mata, and Martins, 2020).

# Proposed Work

## Techniques

The proposed system will include the following key components:

1. ***Comment Summary:*** Applying NLP techniques to comment summaries collected, emphasizing key insights and trends.
2. ***Classification of Negative Comments:*** Developing a machine learning classifier that can detect negative comments using sentiment analysis. This will help in focusing on issues that need to be attended to the most.
3. ***GUI integration:*** The GUI was also created for the enhancements of the application by applying the CustomTkinter library. Through this GUI, users could put in their reviews and see the feedback immediately about the predicted sentiment (Wankhade, Rao, and Kulkarni, 2022).

# Implementation Details

## Data Collection

A screenshot of a computer

Description automatically generated

**Figure 1: Data Collection**

(Source: Obtained from Jupyter Notebook)

This would begin by gathering user reviews from GitHub and Google Play Store. To do this for. Collected data would consist of comments with metadata including timestamp and the user rating.

## Data Preprocessing

After data collection, preprocessing is needed to get the data ready for analysis. It involves the following steps:

**A screenshot of a computer code

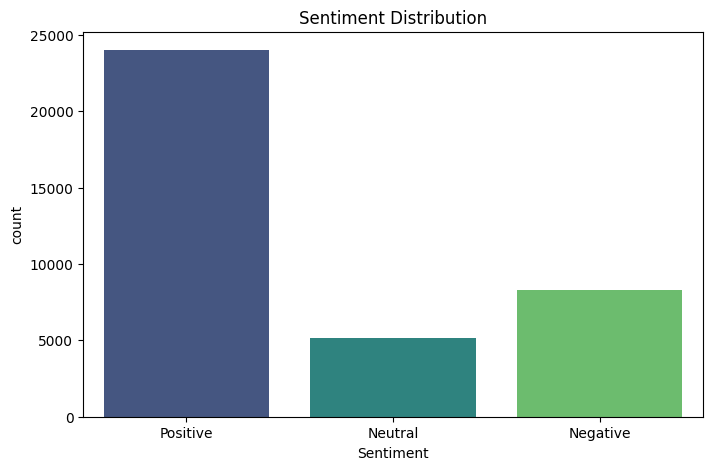
Description automatically generated**

**Figure 2: Data Processing**

(Source: Obtained from Jupyter Notebook)

* Cleaning: The removal of all irrelevant information or noise in the text.
* Tokenization: Comments are broken down into individual words or tokens.
* Stopword Removal: Common words that do not contribute to sentiment are eliminated. For example, "and," "the."
* Lowercasing: All text is changed to lowercase for uniformity (Cui et al., 2023).

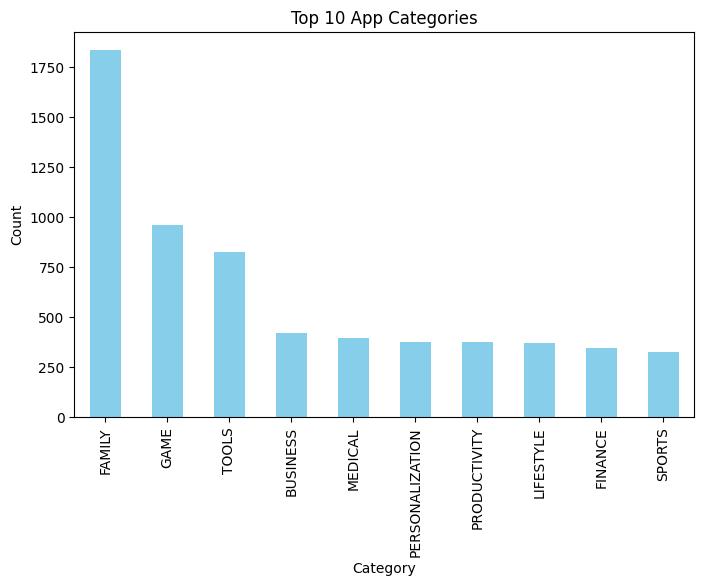
Carrying out EDA on the data gathered helps analyze sentiment distribution and identify which features are being discussed the most through user reviews. For example,



**Figure 3: Sentiment Distribution**

(Source: Obtained from Jupyter Notebook)

Showing what percentage of reviews are positive, negative, or neutral



**Figure 4: Top Categories**

(Source: Obtained from Jupyter Notebook)

Identifying categories of apps that generate maximum feedback. For doing all this, one could make use of visualization libraries such as Matplotlib or Seaborn.

## Splitting Data

A screen shot of a computer code

Description automatically generated

**Figure 5: Train test split**

(Source: Obtained from Jupyter Notebook)

We will divide our dataset into training and testing sets to assess model performance.

## Model Development

We will use several machine learning algorithms such as Random Forests, Logistic Regression, and Support Vector Classifiers (SVC) for classifying negative comments. Each model will be trained using TF-IDF vectorized features from the processed reviews. The model development process included three different classifiers: Random Forest, Logistic Regression, and Support Vector Classifier (SVC) (Singh et al., 2022).

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

**Figure 6: Model implementation**

(Source: Obtained from Jupyter Notebook)

1. ***Random Forest Classifier:*** This ensemble method was implemented with 100 decision trees to improve the prediction accuracy. The model was trained on 80% of the dataset after preprocessing the text data and converting it into TF-IDF features. The classifier had an accuracy of around 92.36% on the test set.
2. ***Logistic Regression:*** The model was trained similarly with the TF-IDF features. With a maximum of 200 iterations for convergence, it proved robust, giving an accuracy of around 91.90%. Logistic regression is well suited for binary classification problems such as sentiment analysis.
3. ***Support Vector Classifier (SVC):*** With a linear kernel, the model was trying to find the best hyperplane to classify the data into positive and negative sentiments. It had the maximum accuracy of about 93.06%, which shows its power in handling complex data distributions.

## Evaluation Method

Train the Classifiers We will train every model in the following way We will measure the performance of every trained model using an accuracy score and a confusion matrix (Sharma, Ali, and Kabir, 2024). We will make use of a summarization technique to group comments according to their sentiment, enabling developers to easily derive conclusions from user feedback. For measuring the performance of our system, we will compare predictions from the classifier with manually labelled dataset of user comments. Key metrics for evaluation will include precision, recall, F1-score, and overall accuracy. In addition, we will collect developers' feedback on this system in practice so that this system can be used pragmatically and practically (Kaur and Sharma, 2023).

## Implementation details of GUI

**A screenshot of a computer

Description automatically generated**

**Figure 7: GUI implementation**

(Source: Obtained from Jupyter Notebook)

* User Input Collection: The app provides a GUI built using CustomTkinter to input the reviews given by the users. Using a text field, comments can be entered by users, which are then further processed to predict the sentiment of such comments using the trained models: Random Forest, Logistic Regression, and Support Vector Classifier (SVC).
* Prediction Display: On submitting a review, the application applies the predict\_sentiment function on the input text. The input text is converted to TF-IDF features, and each of the three classifiers is applied for the prediction of sentiment. This real-time GUI will reflect the output, giving instant feedback to the users of how their comments are interpreted by the models (Chen et al., 2024).
* Feedback Loop: For the model to be more accurate, a feedback mechanism where the users have the choice of saying if the prediction given aligns with how they feel about the review could be developed. This is achieved through an application where, after displaying the predictions, users are required to confirm if the prediction was correct or not. These extra data will be collected and stored for further analysis.
* User Engagement: Allowing users by providing such insight generated from aggregated feedback enhances their experience of the application. For example, indicating statistics about "85% of users thought our prediction was accurate" makes them believe in the system even more and encourages more people to provide feedback (Li et al., 2024).

# Results

All of these models were successful in classifying the negative comments, but the best-performing model was the Support Vector Classifier. The confusion matrices of the three models: Random Forest, Logistic Regression, and Support Vector Classifier (SVC), give an insight into which of these models performs better concerning the classification of the comments as positive or negative. Below are the explanations of the confusion matrices for each model and analysis of strengths and weaknesses for each (Virvou, 2023).

## 1. Random Forest Classifier

A screenshot of a computer

Description automatically generated

**Figure 8: Random Forest Classifier accuracy matrices**

(Source: Obtained from Jupyter Notebook)

Confusion Matrix:

Interpretation:

* True Positives (TP): 4640 comments were rightly classified to have a positive nature.
* True Negatives (TN): 1321 comments were rightly classified to have a negative nature.
* False Positives (FP): 370 comments wrongly labeled as positive were actually negative in nature.
* False Negatives (FN): 123 comments were classified as negative when they were positive.

Performance Metrics:

* Accuracy: Around 92.36%

The model has a high true positive rate, meaning it is very good at classifying positive comments. However, there are many false positives that indicate some negative comments are incorrectly classified.

## 2. Logistic Regression Classifier

**A screenshot of a computer

Description automatically generated**

**Figure 9: Logistic Regression accuracy matrices**

(Source: Obtained from Jupyter Notebook)

Confusion Matrix:

Explanation:

* True Positives (TP): 4654 comments are correctly classified as positive.
* True Negatives (TN): 1277 comments are correctly classified as negative.
* False Positives (FP): 414 comments were classified as positive that were actually negative.
* False Negatives (FN): 109 comments were classified as negative that were actually positive.

Performance Metrics:

* Accuracy: Around 91.90%

This model too does well in picking positive comments but has a little higher false positive than the Random Forest model which indicates that it might not be so sensitive to negative sentiment identification.

## 3. Support Vector Classifier (SVC)

A screenshot of a computer

Description automatically generated

**Figure 10: Support Vector accuracy matrices**

(Source: Obtained from Jupyter Notebook)

Confusion Matrix:

Interpretation:

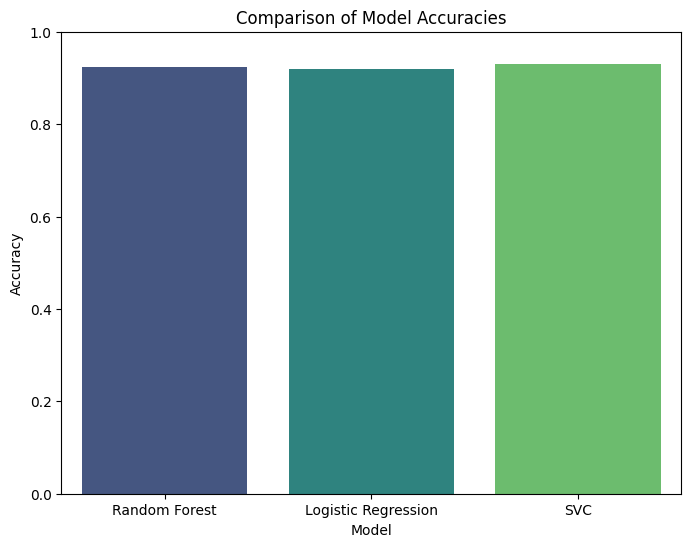
* True Positives (TP): 4608 comments were actually positive
* True Negatives (TN): 1398 comments that were really negative
* False Positives (FP): 293 comments that were mistakenly labeled as positive, but negative.
* False Negatives (FN): 155 comments mistakenly labeled as negative, while they were actually positive.

Performance Metrics:

* Accuracy: around 93.06%

The SVC model has the highest accuracy of all, which displays robust performance on both positive and negative sentiments. However, the model also has a quite significant number of false negatives as compared to other models.

## Summary of Model Performance



**Figure 11: Comparison of all models**

(Source: Obtained from Jupyter Notebook)

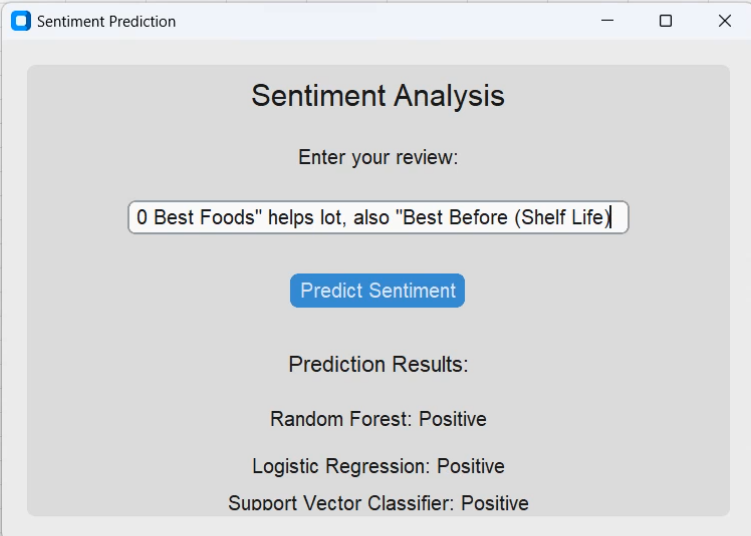
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **True Positives** | **True Negatives** | **False Positives** | **False Negatives** | **Accuracy** |
| Random Forest | 4640 | 1321 | 370 | 123 | 92.36% |
| Logistic Regression | 4654 | 1277 | 414 | 109 | 91.90% |
| Support Vector Classifier | 4608 | 1398 | 293 | 155 | 93.06% |

The confusion matrices reveal that the three models are all pretty good at classifying user comments, but in different ways.

* The Random Forest Classifier has an impressive performance in true positive identification but only a moderate rate in false positive.
* The Logistic Regression Classifier performs comparably but with more false positives than the model in Random Forest.
* The Support Vector Classifier also obtained the highest accuracy but had a significant number of false negatives compared to the other two models.

This understanding may guide further refinement for each model including tuning parameters or the application of ensemble methods for increased classification accuracy over all the sentiment categories.

# User Feedback Integration



**Figure 12: GUI Sentiment Analysis**

(Source: Obtained from Jupyter Notebook)

Integrating feedback from the user into the sentiment analysis system is integral for continuous development and ensuring the model remains timely and productive. It follows the general procedure of user input collection, predicting model performance on the user input, and using such information to maximize model productivity and user performance.

# Conclusion

The main objective of this project is to automate the process of collecting and analyzing user comments from GitHub and Google Play Store efficiently. With a robust classification system that identifies negative feedback and summation techniques, developers would be able to focus on critical areas that need improvement rather than being overwhelmed by volume. The successful completion of this project will not only enhance the quality of software but also improve user satisfaction by addressing concerns in a timely manner. Future work may include expanding this tool's capabilities to include sentiment analysis across other platforms or integrating more advanced NLP techniques for deeper insights into user feedback.

# References

Chen, W., Shen, Z., Pan, Y., Tan, K. and Wang, C., 2024. Applying Machine Learning Algorithm to Optimize Personalized Education Recommendation System. Journal of Theory and Practice of Engineering Science, 4(01), pp.101-108.

Cui, J., Wang, Z., Ho, S.B. and Cambria, E., 2023. Survey on sentiment analysis: evolution of research methods and topics. Artificial Intelligence Review, 56(8), pp.8469-8510.

Kaur, G. and Sharma, A., 2023. A deep learning-based model using hybrid feature extraction approach for consumer sentiment analysis. Journal of big data, 10(1), p.5.

Li, X., Zheng, H., Chen, J., Zong, Y. and Yu, L., 2024. User Interaction Interface Design and Innovation Based on Artificial Intelligence Technology. Journal of Theory and Practice of Engineering Science, 4(03), pp.1-8.

Mata, P., Mata, M.N. and Martins, J.N., 2020. Sentiment analysis: A literature review. In 1st International Conference on Management, Technology and Tourism: Social Value Creation–ICOMTT2020. Instituto Politécnico de Santarém.

Sharma, N.A., Ali, A.S. and Kabir, M.A., 2024. A review of sentiment analysis: tasks, applications, and deep learning techniques. International Journal of Data Science and Analytics, pp.1-38.

Singh, C., Imam, T., Wibowo, S. and Grandhi, S., 2022. A deep learning approach for sentiment analysis of COVID-19 reviews. Applied Sciences, 12(8), p.3709.

Singh, S.N. and Sarraf, T., 2020, January. Sentiment analysis of a product based on user reviews using random forests algorithm. In 2020 10th International conference on cloud computing, data science & engineering (Confluence) (pp. 112-116). IEEE.

Virvou, M., 2023. Artificial Intelligence and User Experience in reciprocity: Contributions and state of the art. Intelligent Decision Technologies, 17(1), pp.73-125.

Wankhade, M., Rao, A.C.S. and Kulkarni, C., 2022. A survey on sentiment analysis methods, applications, and challenges. Artificial Intelligence Review, 55(7), pp.5731-5780.