International Journal of
Human Computations & Intelligence
Vol. 04, Issue. 04, 2025

RESEARCH ARTICLE OPEN ACCESS

AI-Powered Framework for Evaluating Child-Friendly Mobile Applications

Rajesh Lingam

Senior Software Developer,
Adobe Inc, Boston, U.S.A.

DOI: **10.5281/zenodo.15624307**
Received: 12 May 2025 / Revised: 29 May 2025 / Accepted: 09 June 2025

Abstract – As mobile applications increasingly shape children's digital interactions, ensuring their safety and suitability has become critical. This study introduces ML-CFA (Machine Learning for Child-Friendly Applications)**,** a novel framework that classifies mobile apps as appropriate or potentially harmful for young users based on user-generated reviews. The proposed system leverages a robust Natural Language Processing (NLP) pipeline incorporating sentiment analysis, semantic feature extraction, and multiple ML algorithms, including Support Vector Machine (SVM), Naïve Bayes, Logistic Regression, Random Forest, and Convolutional Neural Networks. To assess model reliability, 10-fold cross-validation was applied to a labeled dataset of app reviews. The following metrics were used for evaluation: F1-score, accuracy, precision, recall, and Matthews Correlation Coefficient (MCC). SVM showed the most incredible consistency and generalization among all classifiers, with 98.72% accuracy, 0.984 precision, 0.979 recall, 0.986 F1-score, and 0.975 MCC. Additional studies emphasized the importance of resampling strategies, text preparation, and review score aspects. Preprocessing and undersampling, in particular, significantly increased MCC from 0.472 to 0.627 and enhanced semantic clarity. By automatically classifying app reviews, our results validate how well ML-CFA promotes children's internet safety.

Index Terms – Child Safety, Mobile Applications, User Reviews, Machine Learning, Natural Language Processing (NLP), Text Classification, Support Vector Machine (SVM), Sentiment Analysis, Review Score.

I. INTRODUCTION

The quick spread of mobile applications has completely changed how people of all ages engage with digital material. Children use mobile applications more often than anybody else, and they may access

MILESTONE RESEARCH.IN OPEN ACCESS

ISSN (Online): 2583-5696
Int. Jr. of Hum Comp. & Int.

increasing programs for entertainment, education, and social interaction [1]. Although there are numerous benefits to this digital contact, there are also significant risks, especially if children are exposed to programs that include objectionable content, such as references to drug use, violence, or profanity [2], [3]. Parents and guardians have little influence over their children's digital experiences since current app rating systems frequently fail to indicate such content consistently [4].

Despite notable progress in natural language processing (NLP) and text classification, limited attention has been given to applying these technologies to assess the child-appropriateness of mobile applications through user-generated content such as app reviews. To address this gap, we propose a machine learning-based framework to predict whether mobile applications suit young audiences. Our approach utilizes NLP-driven preprocessing and feature extraction to identify sensitive keywords and linguistic patterns indicative of content that may be inappropriate for children. These engineered features serve as inputs for training a supervised classification model to distinguish between child-safe and unsafe mobile applications effectively.

We choose the Support Vector Machine (SVM) among various classification models due to its robust performance in high-dimensional text classification problems, resilience to overfitting, and suitability for binary decision tasks. Through comprehensive experimentation using 10-fold cross-validation, our proposed framework demonstrated strong predictive performance and reliability in distinguishing adult content from child-safe apps based on review analysis. This study is driven by the pressing need to develop automated solutions that assist parents, educators, and app store moderators in safeguarding children from unsuitable mobile content. Our work contributes to this objective through the following key innovations:

- A novel machine learning framework that uses textual signals from user reviews to evaluate the child-friendliness of mobile applications.
- Custom feature engineering strategy incorporating domain-specific keywords and linguistic cues related to adult themes.
- Empirical validation of the proposed approach using a rigorously structured evaluation protocol shows its superiority over baseline models in terms of accuracy and generalization.
- This is the first-of-its-kind focus on app suitability assessment for children using a review-driven machine learning pipeline, filling a critical void in current digital safety research.

II. LITERATURE SURVEY

The quick integration of machine intelligence across many fields has created more potential and difficulties, particularly regarding children's digital safety. The proliferation of mobile applications has raised serious concerns among researchers, educators, and parents about protecting young users from improper information. Liu et al. [5] investigated the privacy risks in kid-oriented mobile apps by using an SVM algorithm to assess a dataset of 1,738 apps from the Google Play Store with a 95% accuracy rate. This research highlights how machine learning (ML) may be used to find possible privacy risks in applications that kids and teenagers often use.

Mulligan et al. [6] presented an insightful investigation into the privacy practices of mobile applications offered by popular dine-in and fast-food chains in Canada, focusing on data collection concerning child users. The study analyzed the privacy policies of 30 top-rated restaurant apps and explored the practical data retrieval process via Data Access Requests (DARs) submitted by parents on behalf of children aged 9–12. A convenience sample of 11 English-speaking Canadian children used at least one fast-food app for a short duration (5–10 minutes) to simulate real-world interactions by placing food orders. Further, 24 out of the 26 companies (92%) disclosed the types of user data collected through their apps. However, 23 of these (89%) did not differentiate between data collected from children and that from adult users.

Similarly, Hu et al. [7] introduced the Automatic App Maturity Rating (AAR) system, a novel method for predicting inappropriate content in mobile apps through multi-label classification using SVM. Their method demonstrated an 85% prediction accuracy and was evaluated using data from both the Google Play and App Store. According to a similar study by Ying et al. [8], about 30% of mobile applications on the Google Play Store have false maturity ratings. The indications they examined included references to drugs, sexual material, violence, and foul language. The results highlight the shortcomings of the existing maturity labeling schemes, which might mislead users' parents, particularly about whether an app is appropriate for young users.

Thun et al. [9] achieved a 92% accuracy rate in identifying cyberbullying content on social media sites using a random forest classifier on Twitter data. Their strategy fared better than conventional machine learning techniques like Naïve Bayes classifiers, decision trees, and support vector machines. To extend the practical application of their findings, the authors also developed a smartphone application designed to assist parents in monitoring and identifying cyberbullying behaviors. The capacity of deep learning (DL) to identify intricate patterns in vast amounts of unstructured data has led to its growing use in several fields, such as sentiment analysis, speech recognition, and text categorization. When applied to user-generated content, such as social media posts and reviews, DL techniques show significant potential for identifying harmful content like cyberbullying, as highlighted by Thun et al. [9].

The layered architecture of deep learning models enables them to capture abstract representations that traditional machine learning algorithms struggle to model [10]. These strengths are beneficial in activities involving natural language processing (NLP). Deep learning, for instance, has been used to examine app store evaluations to find potentially dangerous apps and discover rating discrepancies [11]. However, despite these advances, the scarcity of well-labeled datasets has limited the use of deep learning for classifying inappropriate apps targeted at children.

III. METHODS & MATERIALS

Our proposed ML-CFA framework for predicting the suitability of mobile applications for children integrates natural language processing, sentiment analysis, and machine learning. Figure 1 displays the overall proposed methodology. Initially, we collected a labeled dataset of app reviews from Kaggle. The textual content of these reviews underwent preprocessing to remove noise and irrelevant tokens. Next, we performed sentiment analysis to compute an emotional score for each review. This sentiment

score was combined with extracted textual features to form a comprehensive feature vector. Finally, these vectors were input into a Support Vector Machine (SVM) classifier, which was trained to distinguish between secure and insecure apps. This integrated approach effectively identifies inappropriate content, supporting safer digital experiences for children.

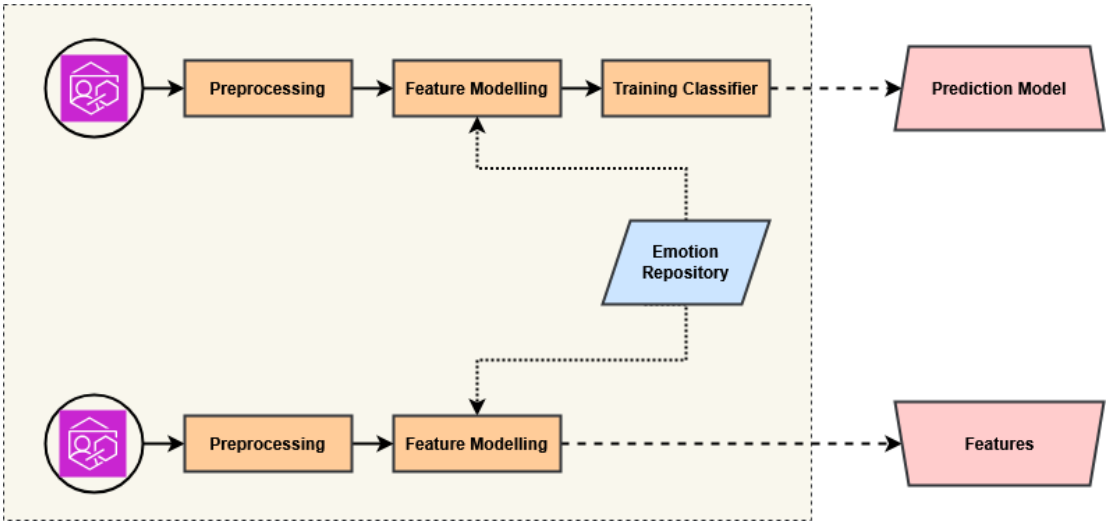

Fig. 1: Graphical Representation of the Overall Methodology

A. Dataset Description

To evaluate the performance of our presented model, we employed the Google Play Store reviews dataset, which is openly accessible on Kaggle and includes more than 12,000 user-generated reviews of a broad range of mobile apps available on the Google Play Store [12]. The textual comments and five-point Likert scale rating in every review provide qualitative and quantitative information on user experiences.

Our research concentrated on utilizing these reviews to divide mobile applications into secure and insecure, according to how appropriate they are for kids. We specifically used the labeling approach listed below:

- Reviews with ratings of 4 and 5 were categorized as secure, indicating a generally positive experience and likely appropriateness for younger users.
- Reviews with ratings of 1 and 2 were categorized as insecure, suggesting negative experiences or potential exposure to inappropriate content.
- Reviews with a neutral rating of 3, totaling 1,991 entries, were excluded from the analysis to maintain a precise binary classification and avoid ambiguity in model training.

The dataset was preprocessed to remove duplicates, irrelevant metadata, and non-English entries, ensuring a clean and focused corpus for downstream natural language processing and machine learning tasks. The data is divided into 70% used for model training and parameter tuning, and 30% for final performance evaluation. Figure 2 illustrates an overview of the dataset's distribution and characteristics.

Fig. 2: Statistics of the dataset

B. Data Preprocessing

User reviews often contain informative content and extraneous elements such as punctuation marks, capitalized variants of words, and filler terms. If left unprocessed, these irrelevant components can inflate computational costs and obscure meaningful linguistic patterns, degrading model performance. We applied a structured natural language preprocessing pipeline before feature extraction and model training to mitigate this. We utilized widely adopted Python libraries NLTK, TextBlob, and SpaCy to facilitate efficient and accurate preprocessing. The significant preprocessing steps are detailed below:

- **Tokenization:** Sentences in normal English make up each review. Tokenization involves breaking the review into phrases and sentences into distinct tokens or lexical units. This makes word-level analysis and modification possible.
- **Lowercase Conversion:** Semantically equivalent words may be redundantly represented due to case differences (e.g., "App" vs. "app"). All tokens were transformed to lowercase to minimize dimensionality in the feature space and standardize the textual data.
- **Stop-word Removal:** Common terms like "is," "the," and "and," while necessary for the grammatical structure, have little semantic weight when it comes to categorization tasks. To make the dataset more efficient and concentrate on keywords that included information, these stop words were eliminated.
- **Lemmatization:** Word variations (such as "learn," "learning," and "learned") frequently express the same underlying idea. Lemmatization improves consistency among related terms by reducing such variations to their base or dictionary form. Lemmatization, in contrast to stemming, preserves the integrity of word meanings by considering the grammatical context.

After applying these preprocessing steps, each user review r is transformed into a tuple representation as follows:

$$r = \langle t_i, s \rangle \quad (1)$$

$$t_i = \langle t_1,\ t_2,\ \ldots\ldots\ t_n \rangle \quad (2)$$

where, r defines the processed user review, t_i is the set of cleaned tokens extracted from the review, $t_1,\ t_2,\ \ldots\ldots\ t_n$ represent individual tokens remaining after preprocessing, and s refers to the corresponding class label.

C. Textual Analysis

User evaluations provide insightful information about whether programs suit kids, especially regarding safety and content appropriateness. These reviews help assess whether an app is safe or dangerous for younger users. Emotional and sentiment analysis tools can be employed to classify reviews as indicating either secure or insecure content. We selected SpaCy to conduct the emotional analysis of user reviews. SpaCy has demonstrated strong performance in analyzing general text and has proven efficient and accurate compared to the above sentiment analysis resources.

In our framework, each user review is processed by inputting the textual data into SpaCy. After extracting the sentiment or emotional tone of the review, it is represented in the form:

$$\mathrm{r} = \langle \mathrm{a},\ t_i, s \rangle \quad (3)$$

Here, a denotes the numerical rating given by the user, t_i represents the processed text, and s refers to the sentiment or emotional category derived from the analysis.

D. Feature Modeling

To construct the feature model, we developed a matrix representation capturing the characteristics of each user review along with its corresponding classification as secure or insecure. Each row in this matrix corresponds to a processed user review. At the same time, the columns represent the extracted features, namely, the significant words identified within the review text and the review's associated rating or label.
Formally, a user review R is expressed as:

$$\mathrm{R} = \langle \mathrm{s}, w_1, w_2, \ldots\ldots w_m \rangle \quad (4)$$

Where, s denotes the review's score or class label and represents the features derived from the review's textual content, in our implementation, the total number of extracted features m was set to 5000 based on the most frequent and meaningful terms across the dataset.

Each element $w_j(R)$ in the matrix reflects the frequency of feature t_j in review R, and is defined as:

$$w_j(R) = \begin{cases} 0, & if\ t_j \notin T_R \\ f_j, & if \in T_R \end{cases} \quad (5)$$

Here, T_R is the set of tokens present in the preprocessed review R, and f_j indicates the frequency of the term t_j in that review. This structured representation provides a scalable basis for machine learning models to distinguish between secure and insecure app reviews using term-level analysis.

E. Model Development

To classify apps' security status based on user reviews, we adopted a supervised learning approach using the Support Vector Machine (SVM) algorithm. The model was trained on a random subset of preprocessed user reviews, and each was represented as a high-dimensional feature vector emanating from the textual content. Let the complete set of training samples be denoted by:

$$D = \langle R_1, R_2, \ldots\ldots R_n \rangle \tag{6}$$

Each review R_i is paired with a binary label $z_i \in \{+1, -1\}$, where, +1 indicates that the app is secure for children and -1 indicates it is insecure. The feature vector of each review x_i encapsulates relevant textual attributes and sentiment scores.

To differentiate between secure and insecure classifications, the SVM constructs a linear decision boundary using a weight vector $w \in \mathbb{R}^m$ and a bias term b. The decision function is defined as:

$$f(R_i) = w.x_i + b \tag{7}$$

where, x_i represents the input features for review R_i. The training process adjusts w and b to optimize the margin between the two classes while minimizing misclassification. This decision function is applied to new or unseen user reviews in the prediction phase. For any incoming review R_j with a feature vector x_j, the classification is determined by:

$$f(R_j) = w.x_j + b \tag{8}$$

Based on this value, the classification rule is:

$$\text{Secure, if } f(R_j) > 1; \text{ Insecure, otherwise} \tag{9}$$

This unified model effectively predicts whether a user review corresponds to a secure or insecure app, enabling automated screening of app suitability for children based on natural language feedback.

F. Methodology for Multi-Model Classification

To evaluate the significance of different machine learning and deep learning models for identifying whether an application is secure or insecure for children, we implemented and trained five different classifiers: Multinomial Naive Bayes (MNB), Logistic Regression (LR), Random Forest (RF), Convolutional Neural Network (CNN), and the proposed Support Vector Machine (SVM). Each model was applied to the same training dataset for fair comparison. Below is a detailed description of each classifier:

- **Multinomial Naive Bayes (MNB):** Multinomial Naive Bayes is a generative model that assumes each word in the document is generated independently from a class-specific multinomial distribution. It is especially effective for text data represented as term frequencies.

$$\hat{y} = \underset{C_k}{arg\ max}\ [log\ P\ (C_k)\ +\ \sum_{j=1}^{n} x_j\ lpg\ P\ (x_j \mid C_k)] \quad (10)$$

where, x_j is the frequency of the j[th] word in the review, and C_k designates the class label.

- **Logistic Regression (LR):** The logistic (sigmoid) function is used in the linear model known as logistic regression to forecast the possibility of a binary outcome [13]. The possibility of the secure class, given an input vector x and a weight vector w, is:

$$P\ (y = 1|x) = \frac{1}{1 + e^{-(w^T x + b)}} \quad (11)$$

Here, x is the feature vector, w is the weight vector, and b is the bias term.

- **Random Forest (RF):** The Random Forest ensemble classifier consists of multiple decision trees trained on arbitrary selections of the training data and characteristics [14]. Across all trees, the vast majority of votes determine the final forecast. For an input x, the prediction ŷ for a set of trees $T_1, T_2, \ldots\ldots T_m$ is:

$$\hat{y} = \text{mode}\ \{T_1(x),\ T_2(x), \ldots\ldots\ldots\ldots T_m(x)\} \quad (12)$$

Each decision tree T_i is built by recursively splitting the dataset using the feature that provides the highest information gain, measured using entropy or Gini impurity.

- **Convolutional Neural Network (CNN):** An architecture for deep learning is the CNN model that captures local semantic patterns in textual data using convolutional filters [15]. In our implementation, the preview text is first tokenized and converted into an embedding matrix $\in \mathbb{R}^{lxd}$, where l is the sequence length, and d is the embedding dimension. A 1D convolutional layer applies filters $w_f \in \mathbb{R}^{hxd}$ over sliding windows of size h, producing feature maps:

$$c_i\ =\ ReLU\ (w_f . E_{i:i+h-1} + b) \quad (13)$$

The output is passed through max-pooling and a fully connected layer to perform binary classification. The final prediction is:

$$\hat{y} = \sigma\ (w_o^T z\ +\ b_o) \quad (14)$$

where, z is the pooled feature vector, w_o is the output weights and σ is the sigmoid activation function.

- **Support Vector Machine (SVM):** The goal of the SVM, a margin-based classifier, is to find the best hyperplane to divide the secure and insecure review classes by the greatest margin. Discriminant Function:

$$f(x) = w^T x + b \quad (15)$$

Classification Rule:

$$\hat{y} = \begin{cases} +1, & if\ f(x) > 1 \\ -1, & if\ f(x) \leq 1 \end{cases} \quad (16)$$

Objective Function (Hinge Loss with Regularization)**:**

$$\min_{w,b} \frac{1}{2} ||w||^2 + C \sum_{i=1}^{N} max\,(0, 1 - y_i)(w^T x_i + b) \quad (17)$$

Here, x_i the feature vector for the i[th] review, $y_i \in \{-1,1\}$ is its label, w and b are model parameters, and C is the regularization constant that controls the trade-off between margin width and classification error. The SVM model was selected as the suggested classifier because of its powerful capacity for generalization, particularly in high-dimensional spaces like those generated from textual input. Its emphasis on margin maximization makes it especially useful for decreasing overfitting and boosting prediction stability.

IV. RESULTS AND DISCUSSIONS

To evaluate the generalization and robustness of the presented ML-CFA framework, a 10-fold cross-validation approach was utilized. This technique minimizes overfitting and tests the model against diverse data. The complete dataset of labeled user reviews, denoted as U, was randomly partitioned into ten equal subsets S_k, where k∈ {1, 2, …, 10}. In each validation round, one subset S_k served as the test fold, while the remaining nine subsets were combined to construct the training set T_k, formally represented as:

$$T_k = \bigcup_{\substack{j=1 \\ j \neq k}}^{10} S_j \quad (18)$$

The evaluation steps for each fold k included several phases. The training set T_k was used to train five classification algorithms, including MNB, LR, RF, CNN, and SVM. As previously described, each classifier was trained using the vectorized feature representations derived from the cleaned and tokenized user reviews. Each trained classifier was evaluated on the held-out fold S_k, where it predicted whether each user review described a child-safe or unsafe application. After completing all 10 folds, the performance metrics for each model were averaged to obtain an overall evaluation score. This cross-validation procedure played a crucial role in verifying the classification effectiveness of ML-CFA and ensuring the reproducibility and statistical significance of the results.

A. Evaluation Metrics

Various assessment metrics appropriate for binary classification tasks were used to evaluate the suggested models' performance thoroughly. These consist of Accuracy, Precision, Recall, F1-score, and Matthews Correlation Coefficient (MCC). True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN) are the terms used to describe the categorization results.

Accuracy: Accuracy measures the percentage of all properly predicted cases, which quantifies the model's total accuracy, which is computed as:

$$\text{Accuracy: } \frac{Tp+TN}{Tp+TN+FP+FN}$$

Precision: The precision of a prediction is the ratio of accurately predicted positive observations to all expected positive observations. In situations when the cost of false positives is considerable, it is especially pertinent.

$$\text{Precision: } \frac{TP}{TP+FP}$$

Recall: Recall estimates the model's ability to identify all relevant instances in the dataset correctly:

$$\text{Recall: } \frac{TP}{TP+FN}$$

F1-Score: When there is an unequal distribution of classes, this balanced metric, which is the harmonic mean of precision and recall, is used. When the expense of false positives and false negatives is about equal, it is advantageous:

$$\text{F1-score: } 2\ \frac{Precision.Recall}{Precision + Recall}$$

Matthews Correlation Coefficient (MCC): MCC offers a reliable single-value metric that accounts for both false positives and negatives, and it is regarded as a balanced metric even in cases when the classes are unbalanced. It has a value between -1 (totally wrong) and +1 (perfect prediction), where 0 denotes a random guess:

$$\text{MCC} = \frac{TP.TN - FP.FN}{\sqrt{(TP+FP)\ (TP+FN)\ (TN+FP)\ (TN+FN)}}$$

B. Performance of the Models

The comparative analysis of classification models, as shown in the performance table 1, highlights the superior effectiveness of the proposed ML-CFA (SVM) model, as shown in Table 1. The experimental evaluation across multiple classifiers demonstrates that the proposed ML-CFA (SVM) model achieves the highest classification performance, with an accuracy of 98.72% and balanced metrics across precision (0.984), recall (0.979), F1-score (0.986), and MCC (0.975) which ability to capture subtle distinctions in semantic and emotional features from user reviews and the SVM's robust decision boundaries in high-dimensional space. The consistent and high values across all metrics indicate that ML-CFA avoids overfitting and underfitting, maintaining strong generalization on unseen data.

Table 1: Performance of the models

Approach	Accuracy (%)	Precision	Recall	F1-Measure	MCC
ML-CFA (SVM)	98.72	0.984	0.979	0.986	0.975
MNB	81.43	0.798	0.754	0.775	0.629
LR	88.67	0.885	0.821	0.852	0.783
RF	90.12	0.904	0.891	0.897	0.842
CNN	92.37	0.926	0.918	0.922	0.879

Despite being computationally efficient and frequently effective at text classification, the MNB classifier performed the lowest out of all the examined models, with an accuracy of 81.43%. Due to its feature independence assumption, which is probably inappropriate for complex, sentiment-rich textual data, it could not accurately detect unsecured app reviews, as evidenced by its comparatively

poor recall (0.754) and MCC (0.629). This suggests some underfitting since the model overlooks meaningful contextual relationships and oversimplifies the data distribution.

LR performed moderately well with an accuracy of 88.67%, outperforming MNB but still lagging behind more complex models. Its lower recall (0.821) and F1-score (0.852) suggest a tendency to misclassify some insecure reviews, potentially due to its linear nature, which limits its ability to model intricate relationships in textual features. LR shows slight underfitting, particularly when compared with non-linear classifiers like RF and SVM. The RF classifier achieved an accuracy of 90.12%, with a precision of 0.904 and an MCC of 0.842. While RF performed better than LR and MNB, its performance was slightly less stable than that of ML-CFA. The model may have captured spurious patterns due to its ensemble of decision trees, making it more susceptible to mild overfitting, especially on high-dimensional textual features, despite internal bagging mechanisms.

CNN showed relatively strong performance, with 92.37% accuracy and consistently high precision and recall. CNNs can capture local and hierarchical textual patterns, which is beneficial for analyzing user reviews. However, their performance highly depends on the quantity and variability of training data. In this case, CNN's performance indicates a slight tendency toward overfitting, as deep architectures can memorize training data when insufficiently regularized or trained on limited, non-diverse datasets.

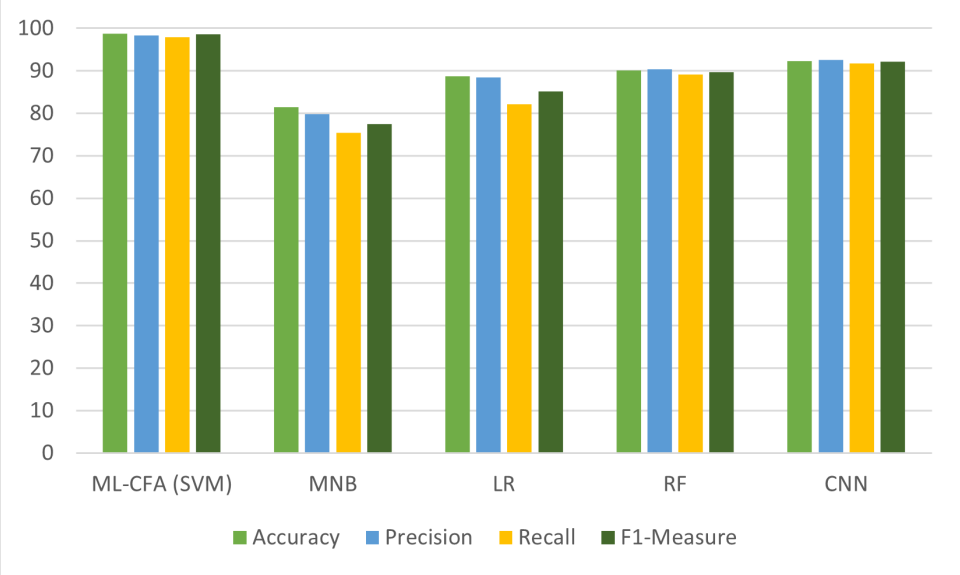

Fig. 3: Performance of the models, including accuracy, precision, recall, and F1-measure

While CNN and RF perform competitively, only ML-CFA (SVM) demonstrates optimal balance and robustness, achieving the highest overall performance without signs of overfitting or underfitting exhibited in Figure 3. Its effectiveness in distinguishing secure from insecure apps based on user reviews makes it the most suitable model for real-world deployment in content moderation and child-safety assessment systems. Figure 4 illustrates the distribution of F-measure values obtained through 10-fold cross-validation for three approaches: ML-CFA (SVM), Zero-Rule (ZR), and Random Prediction (RP), visualized using a violin plot. Each method is represented by a distinct bean, where the individual horizontal lines indicate the F-measure scores from each fold, and the bold central line denotes the mean F-measure for that approach. The ML-CFA (SVM) method shows a strong and steady performance, with results that remain closely grouped around a high average and minimal variation across different folds. This shows how sturdy and dependable it is. On the other hand, the Zero-Rule approach exhibits larger volatility and may produce conflicting results based on the division

of the data. With a lower average F-measure and visible unpredictability, the Random Prediction model performs the worst and is an untrustworthy choice.

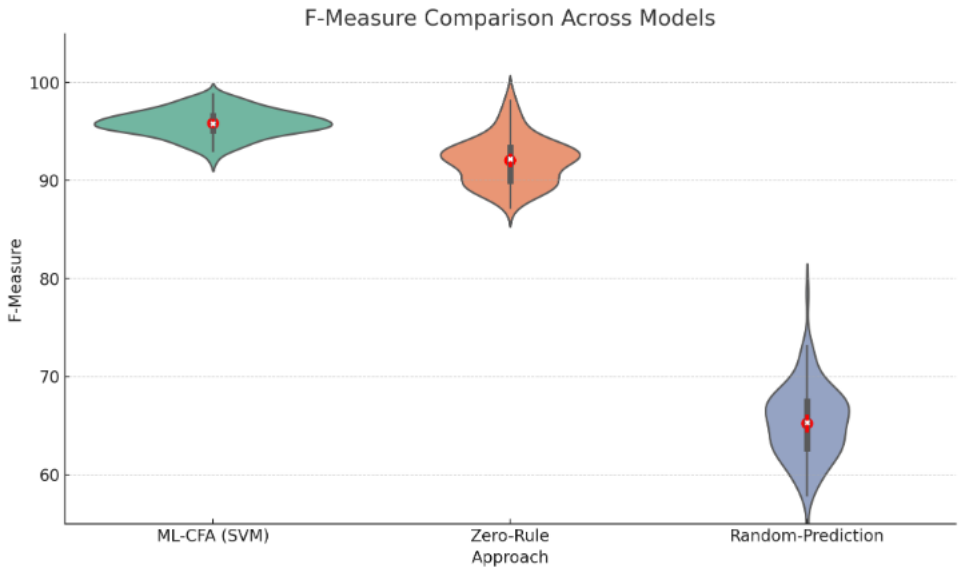

Fig. 4: Distribution of the f-measure

C. Impact of the Review Score

To explore the impact of review score features on the effectiveness of the proposed ML-CFA (SVM) model, we conducted a comparative evaluation with and without the inclusion of the review score. Figure 5, presents the experimental outcomes. The first column specifies whether the review score was utilized, and the remaining columns summarize the corresponding evaluation metrics: precision, recall, F1-measure, and MCC.

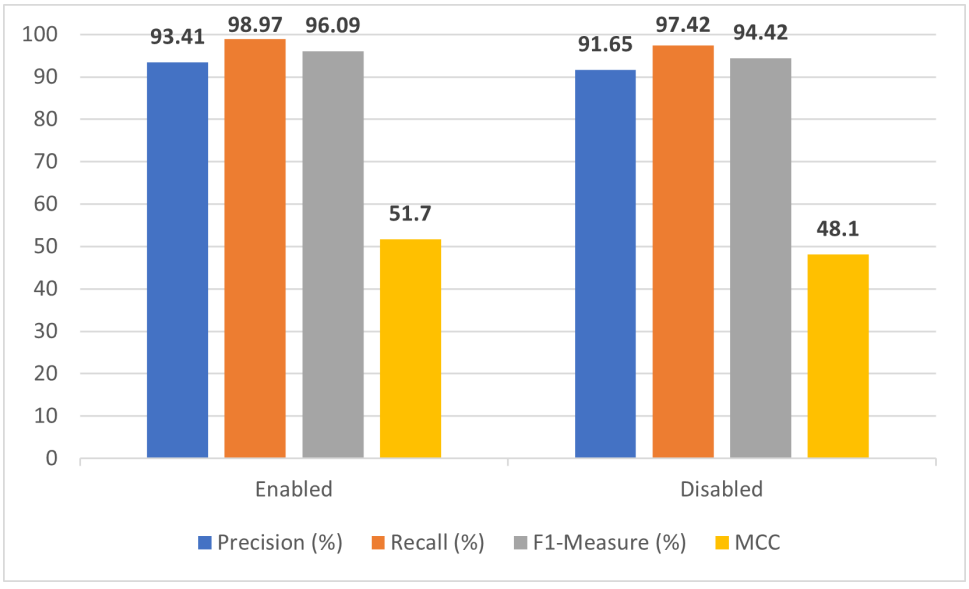

Fig. 5: Impact of the review score

All measures showed a considerable improvement in categorization performance when the review score feature was included. The model's precision (from 93.41% to 91.65%), recall (from 98.97% to 97.42%), F1-score (from 96.09% to 94.42%), and MCC (from 0.517 to 0.481) all decreased when the feature was removed, suggesting a drop in predictive quality. After activating the review score, relative improvements of around 1.92% in accuracy, 1.59% in recall, 1.76% in the F1-measure, and 7.48% in MCC were seen quantitatively. These improvements highlight how the review score improves the algorithm's discriminative capacity and lowers misclassifications. According to this comparison assessment, including the review score as a feature result in more accurate and trustworthy predictions about the security and appropriateness of mobile applications for kids. Therefore, it is recommended that this feature be kept in the ML-CFA feature engineering pipeline.

D. Effect of Re-Sampling on Model Performance

Resampling techniques are instrumental in addressing class imbalance issues in classification tasks. Both under- and over-sampling strategies were used to assess how sampling methods affected the performance of the suggested ML-CFA (SVM) model. Specifically, we employed random sampling for the dominant class in under-sampling and the Synthetic Minority Over-sampling Technique (SMOTE) to augment the minority class. Table 2 summarizes the experimental outcomes. Each row reports the model's performance under a different sampling condition. The metrics evaluated include precision, recall, F1-measure, and the MCC.

Table 2: Impact of Re-sampling

Sampling Strategy	Precision (%)	Recall (%)	F1-Measure (%)	MCC
None	91.83	98.61	95.10	0.472
Under-sampling	94.25	99.03	96.55	0.627
Over-sampling	93.01	98.92	95.86	0.592

Both sampling strategies improve the predictive capacity of the model. A better balance in categorization was shown by the F1 measure, which rose from 95.10% (no sampling) to 96.55% with under-sampling and 95.86% with over-sampling. Under-sampling had a greater effect than over-sampling, and the MCC increased significantly from 0.472 to 0.627 and 0.592, respectively. This suggests that undersampling effectively reduces class differences in this case. The robustness of the ML-CFA in the presence of balanced data was confirmed by the high accuracy and recall levels maintained by all techniques. These findings show that resampling, especially under-sampling, effectively enhances classification outcomes in imbalanced datasets.

E. Influence of Preprocessing on Prediction Accuracy

User reviews often contain noisy and irrelevant information, such as special symbols, filler words, or grammatical inconsistencies, which can impede the learning capacity of machine learning algorithms. To investigate the significance of preprocessing in our pipeline, we conducted an ablation study by training the ML-CFA model with and without text preprocessing. Table 3 presents the comparative performance metrics obtained under each condition.

Table 3: Impact of preprocessing

Preprocessing	Precision (%)	Recall (%)	F1-Measure (%)	MCC
Enabled	91.83	98.61	95.10	0.472
Disabled	91.44	98.25	94.70	0.458

Preprocessing yields consistent improvements across all metrics. Precision increased by 0.43%, recall by 0.37%, F1-measure by 0.42%, and MCC by approximately 3.06%. The improvement likely stems from removing extraneous tokens such as emojis, punctuation, and stop words, which can dilute the signal-to-noise ratio in text data. For token normalization, the Lancaster stemming algorithm was employed. In contrast to the Porter stemmer, which often truncates words into unidentifiable forms, the Lancaster stemmer effectively decreases "trying" to "try," maintaining semantic clarity that is advantageous for categorization later on. This experiment emphasizes how crucial a thorough pretreatment step is for jobs involving natural language processing, especially when working with user-generated information. Preprocessing greatly improves prediction accuracy and model generalization.

V. CONCLUSION AND FUTURE WORK

Nowadays, with mobile apps influencing every part of a child's life, maintaining digital safety has changed from being a parental worry to becoming a technology need. Based on user-generated evaluations, this study presented ML-CFA, an intelligent and scalable framework that uses machine learning and textual analytics to assess how kid-friendly mobile apps are. ML-CFA bridges the gap between user perceptions and automated content safety assessments by converting subjective textual sentiments into quantifiable insights. The integration of sentiment analysis with robust classification models—ranging from classical algorithms like Naive Bayes and Logistic Regression to advanced architectures like CNN and SVM—demonstrates that natural language can be a reliable proxy for gauging app appropriateness.

Our results highlight the viability of using such a framework as a proactive filter within app ecosystems, giving parents, teachers, and legislators a data-driven tool to reduce children's exposure to digital hazards. Future versions of this study will seek to integrate multimodal data, multilingual capabilities, and real-time adaptation as mobile material develops, establishing ML-CFA as a key component in the larger goal of digital child safety. Future studies will investigate the model's enrichment with more inputs, such as visual material and in-app behavioral data. Supporting several languages and integrating sophisticated natural language models might increase the system's efficacy even further. The ML-CFA architecture ultimately lays the foundation for innovative digital solutions that assist parents, educators, and regulators in making better decisions about the safety of children in mobile contexts.

REFERENCES

1. Papadakis, S., & Kalogiannakis, M. (2017). Mobile educational applications for children: What educators and parents need to know. *International Journal of Mobile Learning and Organisation, 11*(3), 256–277.
2. Majebi, N. L., & Drakeford, O. M. (2025). Child safety in the digital age: Historical lessons from media regulation and their application to modern cybersecurity policies. *[Manuscript in preparation or unpublished work]*.
3. Pardhi, P. (2025). Content moderation of generative AI prompts. *SN Computer Science, 6*(4), 329.

4. Li, M., Lv, Y., Pu, Y., & Wu, M. (2025). Design and evaluation of children's education interactive learning system based on human-computer interaction technology. *Scientific Reports, 15*(1), 6135.
5. Liu, M., Wang, H., Guo, Y., & Hong, J. (2016). Identifying and analyzing the privacy of apps for kids. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications* (pp. 105–110).
6. Mulligan, C., Gillis, G., Remedios, L., Parsons, C., Vergeer, L., & Potvin Kent, M. (2025). Children's digital privacy on fast-food and dine-in restaurant mobile applications. *PLOS Digital Health, 4*(2), e0000723.
7. Hu, B., Liu, B., Gong, N. Z., Kong, D., & Jin, H. (2015). Protecting your children from inappropriate content in mobile apps: An automatic maturity rating framework. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management* (pp. 1111–1120).
8. Chen, Y., Xu, H., Zhou, Y., & Zhu, S. (2013). Is this app safe for children? A comparison study of maturity ratings on Android and iOS applications. In *Proceedings of the 22nd International Conference on World Wide Web* (pp. 201–212).
9. Thun, L. J., Teh, P. L., & Cheng, C.-B. (2022). CyberAid: Are your children safe from cyberbullying? *Journal of King Saud University - Computer and Information Sciences, 34*(7), 4099–4108.
10. Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning* (pp. 17–36). JMLR Workshop and Conference Proceedings.
11. Sadiq, S., Umer, M., Ullah, S., Mirjalili, S., Rupapara, V., & Nappi, M. (2021). Discrepancy detection between actual user reviews and numeric ratings of Google App Store using deep learning. *Expert Systems with Applications, 181*, 115111.
12. Meng, X., Li, S., Malik, M. M., & Umer, Q. (2022). Machine-learning-based suitability prediction for mobile applications for kids. *Sustainability, 14*(19), 12400.
13. Hua, Y., Stead, T. S., George, A., & Ganti, L. (2025). Clinical risk prediction with logistic regression: Best practices, validation techniques, and applications in medical research. *Academic Medicine & Surgery*. *[In press]*
14. Zhou, Z.-H. (2025). *Ensemble methods: Foundations and algorithms*. CRC Press.
15. Kihal, M., & Hamza, L. (2025). Efficient Arabic and English social spam detection using a transformer and 2D convolutional neural network-based deep learning filter. *International Journal of Information Security, 24*(1), 1–14.
16. Ahmed, S. T., Fathima, A. S., & Reema, S. (2023, December). An Improved System for Students Feedback Analysis Using Supervised Probability Techniques. In *2023 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)* (Vol. 10, pp. 328-333). IEEE.