Cyberbullying Detection

Mattia Segreto

1 Introduction

In last years, the phenomenon of cyberbullying increases significantly; this is likely due to the social network global spread and the growing accessibility of communication tools. To recognize and counter harmful behavior is an urgent challenge in order for promoting a safer digital environment.

The project aim is to propose a detection system for cyberbullying attacks based on the analysis of tweets. Following the detection, a second purpose has been identified: to provide an explanation of how the system reached such a conclusion.

2 Methodology

This chapter presents the methodological framework adopted for the development of the cyberbullying detection system. The approach follows a structured pipeline that includes data preprocessing, feature extraction, supervised learning techniques, and model evaluation. Particular attention is given to the classification process, which is designed in multiple stages to enhance accuracy and interpretability. Furthermore, explainability methods are integrated to improve transparency and align the system with current ethical and regulatory standards.

2.0.1 Data Visualization

Tag clouds were generated for each of the four dataset classes in order to provide a visualization that provides a qualitative overview of the most frequent words associated with each label and offering also an initial insights of the characterizing language patterns among different categories.

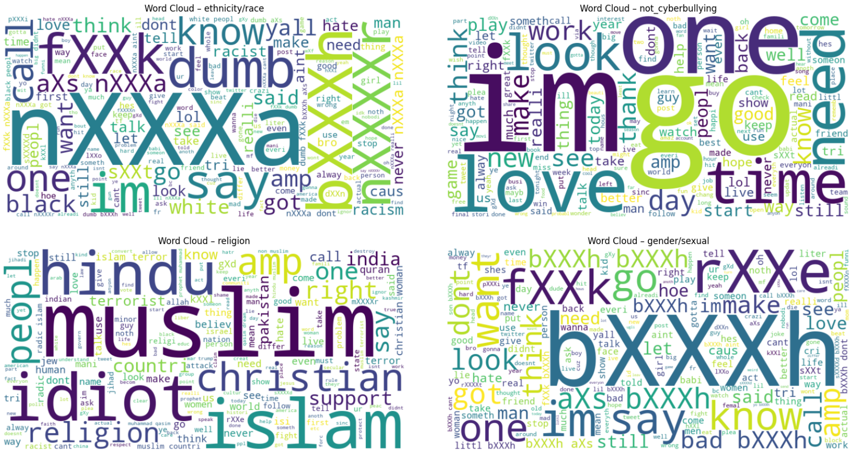


Figure 1: Tag clouds for each class.

In the first of the two following images, it's possible to asses that the classes are balanced for both the stages of the sequential classification. Indeed, the cyberbullying types occur approximately same number of times and their total is almost equivalent to the not_cyberbullying occurrences. In the second one, it's possible to analyze the tweet length distribution that, how expected, is quite short.

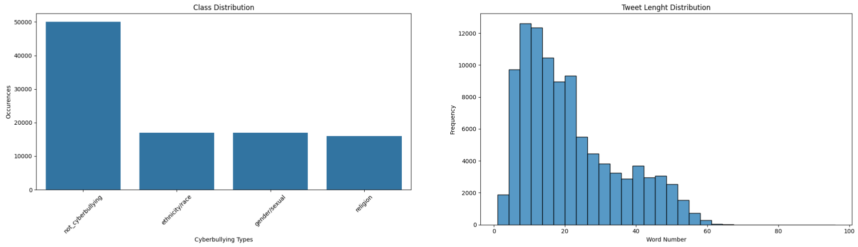


Figure 2: Tag clouds for each class.

2.0.2 Data Splitting and Stratification

The dataset was split into training and test sets using an 80/20 ratio. Stratified sampling was applied based on the multiclass label in order to preserve the original class distribution in both subsets. A fixed random seed was used to ensure reproducibility of the split.

2.1 Data Collection and Preprocessing

2.1.1 Dataset Description

The following dataset description is based on the dataset introduced in the paper *Self-Training for Cyberbully Detection: Achieving High Accuracy with a Balanced Multi-Class Dataset*[1], which was specifically created to support the classification of cyberbullying on Twitter. In their work, the authors first constructed a labeled "seed" by merging datasets already available in the literature, resulting in a corpus of approximately 34,000 tweets divided into four classes: non-cyberbullying, religion, ethnicity/race, and gender/sexuality (which are also the four label). They then used the Twitter API to collect, over a two-month period, around 4,000,000 unlabeled tweets, performing thorough preprocessing that included removing URLs, emojis, hashtags, punctuation, and stop words, normalizing the text (lowercasing, stemming/lemmatization), and converting it into numerical representations using BoW, TF-IDF, and embedding techniques (GloVe, Keras, BERT) .

To expand the labeled dataset, they adopted a self-training strategy based on an ensemble of six models (Random Forest, Decision Tree, and XGBoost with TF-IDF; BERT; LSTM; and BiLSTM with Keras embeddings). After training each model on 20% of the seed (about 7,000 tweets), the authors iteratively assigned pseudo-labels to the unlabeled tweets, including in the final dataset only those messages for which all six classifiers agreed on the same label with a confidence threshold of 0.7, yielding approximately 2.4 million pseudo-labeled tweets. Because the resulting distribution was still imbalanced, they then randomly sampled 99,991 tweets to ensure roughly 17,000 instances for each of the four classes. To validate label quality, five batches of 1,000 tweets each were manually reviewed by three social media experts, achieving an accuracy rate above 90% .

Label Distribution:

- `not_cyberbullying`: 50,000 occurrences
- `ethnicity/race`: 17,000 occurrences
- `gender/sexual`: 17,000 occurrences
- `religion`: 15,990 occurrences

Dataset Features:

- `text`: `string` (tweet text)
- `label`: `categorical` (cyberbullying class)

[1] Self-Training for Cyberbully Detection: Achieving High Accuracy with a Balanced Multi-Class Dataset, available at `https://uregina.ca/~nss373/papers/cyberbully-detection.pdf`

2.1.2 Text Cleaning

In order to reduce noise and standardize the input for the feature extraction step, a set of preprocessing techniques was applied to the raw tweets. These steps are especially important in the context of Twitter data, which is typically informal, highly variable, and often includes non-linguistic symbols. The cleaning procedure includes:

- **Lowercasing:** All text is converted to lowercase to ensure that words like `Hate` and `hate` are treated as the same token. This helps reduce the vocabulary size without losing semantic meaning.

- **Removal of URLs, mentions, and punctuation:** Tweets often contain hyperlinks, user mentions (e.g., `@username`), and various punctuation marks that do not contribute meaningfully to cyberbullying detection. Removing them simplifies the input without affecting classification performance.

- **Stopword removal:** Common words such as `the`, `is`, and `and` are removed to focus on the most informative terms in each tweet. This is particularly helpful for short texts like tweets, where maximizing the signal-to-noise ratio is essential.

- **Stemming:** Words are reduced to their root form (e.g., `harassing` → `harass`), which helps group different inflected forms of the same word under a single representation. This reduces sparsity and improves the effectiveness of models relying on token frequency or similarity. Lemmatization was not used because, given the unstructured nature of the text, it would have been less efficient: lemmatization relies on correct grammatical structure and part-of-speech tagging, which are often unreliable in noisy or informal text.

These choices make the text cleaner, more compact, and more suitable for feature extraction methods, while preserving the linguistic signals needed for cyberbullying detection.

Binary Label Creation: To support the two-stage classification pipeline, the original multiclass label has been accompanied by a binary label distinguishing between cyberbullying and non-cyberbullying content. In this transformation, all tweets originally labeled as `not_cyberbullying` were grouped under the class `0` (non-cyberbullying), while all remaining tweets were grouped under the class `1` (cyberbullying). This binary representation is particularly useful as a preliminary filter in a cascaded classification system, where the first stage detects whether a tweet is abusive at all, and only the second stage performs fine-grained categorization among the different types of cyberbullying. Such a design reduces the complexity of the fine-grained classifier and helps focus its training on content that is more likely to be harmful.

2.2 Feature Extraction

In this project, three different text vectorization methods were employed to extract meaningful features from the tweets: Bag of Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and Word2Vec(W2V-1) embeddings. These techniques allow the conversion of raw textual data into numerical representations suitable for downstream classification tasks.

BoW was used to construct a frequency matrix that reflects how often each word appears across the dataset. While effective in capturing general word usage, this method tends to overrepresent common terms such as *like*, *people*, or *know*, which may carry limited discriminative power for identifying cyberbullying content.

To mitigate this, the TF-IDF approach was also applied. It enhances the BoW representation by down-weighting ubiquitous terms and up-weighting words that are more unique to specific tweets. This helps focus the model on more informative and discriminative tokens, particularly useful in a dataset where repeated expressions of emotion or aggression are relevant.

Additionally, two Word2Vec model was trained to capture semantic and syntactic relationships between words. The model was trained on a large, unlabeled corpus composed of two datasets related to cyberbullying and offensive content, totaling approximately 70,000 preprocessed sentences. The Skip-Gram architecture (`sg=1`) was used, with a `vector_size` of 200 and a `window` of 7. Different `min_count` values were evaluated: setting it to 2 excluded 31,149 words out of 44,420 (70.1%), retaining 13,271 tokens. A lower threshold (1) excluded 58.4%, while values of 5 or 10 would have eliminated over 81% and 87% of the vocabulary, respectively.

This multi-faceted feature extraction setup provided several distinct vector representations of the text, allowing models to be trained and compared across different input spaces.

2.3 Classification Overview

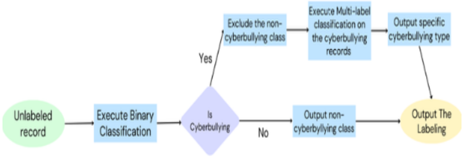

Exclude the non-cyberbullying class
Execute Multi-label classification on the cyberbullying records
Output specific cyberbullying type
Yes
Unlabeled record
Execute Binary Classification
Is Cyberbullying
No
Output non-cyberbyllying class
Output The Labeling

Figure 3: Two-stage classification pipeline.

Source: Self-Training for Cyberbully Detection: Achieving High Accuracy with a Balanced Multi-Class Dataset, available at `https://uregina.ca/~nss373/papers/cyberbully-detection.pdf`

The two-stage classification improves the system's interpretability: by splitting the process into detection and specific categorization, it becomes easier to identify where an error occurs and intervene accordingly, with greater precision in tuning and feature analysis.

Moreover, this structure mirrors human reasoning, which typically involves first assessing whether content is problematic and only then analyzing its severity or nature. As a result, errors can also be interpreted on two different levels of severity: failing to detect harmful content is clearly more critical than misclassifying the specific type of cyberbullying. This is especially true since some offensive messages may not fit neatly into a single label, and therefore an error at the multiclass stage does not necessarily represent a serious failure. This approach enables a more realistic understanding of the model's behavior and a more informed management of its performance in practical applications. For example, consider the following case: the ground truth label for the tweet "my 2282 says that you can replace the testimony of one man with that of two women" was set as religion, whereas the model predicted gender—and arguably, it was correct to do so.[2] This suggests that the reported results may underestimate the models' actual real-world performance, as some supposed "errors" stem from questionable ground truth labels rather than true misclassifications.

2.4 Binary Classification Stage

The binary classification stage aims to distinguish between tweets containing cyberbullying and those that are harmless. Each tweet is labeled as either `cyberbullying` or `not_cyberbullying`, framing the task as a supervised binary classification problem.

Nested cross-validation was performed for each of the nine model–vectorizer combinations to obtain unbiased performance estimates and tune hyperparameters.The classifiers that has been used are:Random Forest, Logistic Regression, and Linear SVM. Model evaluation employed a nested stratified k-fold framework: an outer loop estimated generalization performance, while an inner loop conducted hyperparameter tuning. From each inner fold, the optimal parameter set was recorded, and upon completion of all folds the final hyperparameters were chosen as the most frequent combination ("mode") across folds. That configuration was then used to retrain the pipeline on the entire training set. Optimization targeted the macro-F1 score to balance precision and recall, thus addressing both false positives and false negatives. Model selection involved pairwise comparisons among all pipelines, with a "win" noted when one model outperformed another on a given fold. The normality of per-fold F1 differences was assessed using the Shapiro–Wilk test; if normality held, a paired t-test was applied, otherwise the Wilcoxon signed-rank test was used. To ensure robustness and ample sample size, all comparisons were based on a repeated stratified k-fold cross-validation scheme.

[2] The tweet likely refers to verse 2:282 of the Quran, which discusses financial contracts and stipulates that, if two male witnesses are not available, one man and two women may testify. Although the rule originates from a religious text, the content of the tweet explicitly highlights a gender-related issue, leading the model to a plausible alternative classification.

2.5 Multiclass Classification Stage

The goal of the multiclass classification stage is to assign a specific type of cyberbullying to each tweet previously identified as harmful. The hyperparameter-tuning procedure was identical to that used in the binary stage: each of the three classifiers (Random Forest, Logistic Regression, and Linear SVM) was paired with each of the three vectorization methods, yielding nine model configurations. Nested stratified k-fold cross-validation was employed to optimize hyperparameters;this time targeting overall accuracy and, once tuning was complete, each pipeline was retrained on the full training set. Model selection again relied on pairwise comparisons of per-fold scores (wins counted per fold) with significance assessed via Shapiro–Wilk for normality and paired t-tests or Wilcoxon signed-rank tests as appropriate.

2.6 Explainability Module

To improve model transparency and interpretability, an explainability module was integrated into the classification pipeline. This component plays a crucial role in enhancing trust and accountability in machine learning applications, especially in sensitive tasks such as cyberbullying detection. Notably, since the best-performing model was a Random Forest trained on TF-IDF features, both global and local interpretability techniques could be effectively applied. Random Forest inherently supports feature importance analysis, while its tree-based structure allows for instance-level decomposition of predictions—making it an ideal candidate for explainability.

Global Explainability. Pattern mining techniques were applied to study the most frequent and informative word associations within the dataset. Specifically, the most frequently occurring words were analyzed, and maximal and closed itemsets meeting a predefined support threshold were extracted. Rather than focusing on association rules, which are better suited to modeling directional relationships between items, maximal and closed itemsets has been chosen. This choice was motivated by the nature of the data: tweets are extremely short and informal, making it difficult to establish strong directional associations between words. Maximal and closed itemsets allowed us to capture robust co-occurrence patterns without introducing noise or spurious dependencies, providing a clearer and more stable view of the most informative word groupings associated with different cyberbullying categories. In parallel, the `feature_importance_` attribute of the Random Forest model was used to rank features by their global relevance, offering insight into the textual elements that most influenced the model during training. This dual approach enables a comprehensive form of global explainability: on one hand, it provides an understanding of the model's behavior through feature importance; on the other, it offers an intrinsic explanation of the phenomenon itself by uncovering the underlying word patterns through pattern mining.

Local Explainability. For instance-level interpretation, the TreeInterpreter tool was employed. This method decomposes the output of the Random Forest model into individual feature contributions, making it possible to understand which specific words influenced the classification of a given tweet. Thanks to the use of TF-IDF vectorization, each explanation could be interpreted in terms of the positive or negative contribution of each word. Interestingly, the model also relies on the absence of certain words: even terms not present in the tweet can carry weight by shifting the decision boundary. This aspect highlights the nuanced reasoning adopted by the classifier and provides users with a deeper understanding of its behavior.

3 Experimental Results

This section presents the results obtained at each stage of the proposed pipeline, following the methodological steps described previously. For each stage, the evaluation metrics used, the performance of the tested models, and a short discussion of the results were reported.

3.1 Binary Classification Results

Experimental Setup and Grid Search. As described in Section 2.4, three classifiers Random Forest, Logistic Regression, and Linear SVM were evaluated in combination with three vectorization methods: Bag of Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and custom-trained Word2Vec embedding (W2V-1). This led to a total of 9 distinct model configurations. The grid of hyperparameters used during tuning was the following:

- **Logistic Regression:**
  - `C`: {0.01, 0.1, 1}
  - `class_weight`: "balanced"
- **Linear SVM:**
  - `C`: {0.01, 1, 10}
- **Random Forest:**
  - `n_estimators`: {100, 500, 1000}
  - `max_depth`: {None, 20}
  - `class_weight`: "balanced"
  - `random_state`: 42

Model evaluation Table 1 reports the optimal hyperparameters that were selected via nested cross-validation for each vectorizer–classifier pipeline. For each configuration, we show the mode of the best parameters found across the ten outer folds.

Configuration	Chosen Parameters
BoW + LogisticRegression	{’C’: 1, ’class_weight’: ’balanced’}
BoW + RandomForest	{’class_weight’: ’balanced’, ’max_depth’: None, ’n_estimators’: 1000, ’random_state’: 42}
BoW + LinearSVM	{’C’: 1}
TF-IDF + LogisticRegression	{’C’: 1, ’class_weight’: ’balanced’}
TF-IDF + RandomForest	{’class_weight’: ’balanced’, ’max_depth’: None, ’n_estimators’: 500, ’random_state’: 42}
TF-IDF + LinearSVM	{’C’: 10}
W2V-1 + LogisticRegression	{’C’: 1, ’class_weight’: ’balanced’}
W2V-1 + RandomForest	{’class_weight’: ’balanced’, ’max_depth’: None, ’n_estimators’: 1000, ’random_state’: 42}
W2V-1 + LinearSVM	{’C’: 10}

Table 1: Chosen parameters for each configuration (mode over 10 folds)

Table 2 then summarizes the key evaluation metrics—mean accuracy, precision, recall and F1-score—along with their standard deviations across the outer folds. This allows you to compare not only the average performance but also the stability of each model.

Vectorizer	Classifier	Mean Acc.	Mean Prec.	Mean Rec.	Mean F1	Std Acc.	Std Prec.	Std Rec.	Std F1
TF-IDF	RandomForest	0.995075	0.997863	0.992273	0.995060	0.000498	0.000392	0.001075	0.000501
BoW	LinearSVM	0.995037	0.998917	0.991148	0.995017	0.000562	0.000449	0.001375	0.000568
BoW	RandomForest	0.994737	0.997386	0.992073	0.994722	0.000629	0.000585	0.001090	0.000632
BoW	LogisticRegression	0.994512	0.999042	0.989973	0.994485	0.000883	0.000512	0.001954	0.000892
TF-IDF	LinearSVM	0.994249	0.998211	0.990273	0.994225	0.000829	0.000697	0.001591	0.000836
TF-IDF	LogisticRegression	0.993337	0.998989	0.987673	0.993297	0.000891	0.000422	0.001875	0.000902
W2V-1	RandomForest	0.954320	0.973461	0.934112	0.953371	0.002521	0.003389	0.004390	0.002615
W2V-1	LinearSVM	0.949120	0.955116	0.942539	0.948776	0.002436	0.003403	0.004332	0.002491
W2V-1	LogisticRegression	0.948632	0.954510	0.942164	0.948290	0.002797	0.002941	0.004430	0.002869

Table 2: Nested CV Results: mean metrics and standard deviations

Model selection In Table 3 has been presented the pairwise victory matrix across all model–vectorizer combinations. A value of 1 indicates that the model in the row significantly outperformed the model in the column in head-to-head statistical tests, whereas 0 denotes no statistically significant win.

	RF + TF-IDF	LSVM + BoW	RF + BoW	LR + BoW	LSVM + TF-IDF	LR + TF-IDF	RF + W2V-1	LSVM + W2V-1	LR + W2V-1
RF + TF-IDF	0	0	1	1	1	1	1	1	1
LSVM + BoW	0	0	1	1	1	1	1	1	1
RF + BoW	0	0	0	1	1	1	1	1	1
LR + BoW	0	0	0	0	1	1	1	1	1
LSVM + TF-IDF	0	0	0	0	0	1	1	1	1
LR + TF-IDF	0	0	0	0	0	0	1	1	1
RF + W2V-1	0	0	0	0	0	0	0	1	1
LSVM + W2V-1	0	0	0	0	0	0	0	0	1
LR + W2V-1	0	0	0	0	0	0	0	0	0

Table 3: Pairwise victory matrix (1 = win, 0 = no win)

Table 4 summarizes the total number of head-to-head victories for each pipeline, giving an at-a-glance ranking of overall comparative performance. Higher counts reflect more frequent wins against competing configurations.

Model	**Victories**
RandomForest + TF-IDF	7
LinearSVM + BoW	7
RandomForest + BoW	6
LogisticRegression + BoW	5
LinearSVM + TF-IDF	4
LogisticRegression + TF-IDF	3
RandomForest + W2V-1	2
LinearSVM + W2V-1	1
LogisticRegression + W2V-1	0

Table 4: Number of head-to-head victories per model

Focusing on the two top-ranked models, Table 5 details the results of the pairwise statistical tests used to establish significance. For each winner–opponent pair has been reported the test type, test statistic, p-value, and the final outcome (win/no win).

Winner	**Opponent**	**Test**	**Stat.**	**p-value**	**Outcome**
RandomForest + TF-IDF					
	LSVM + BoW	t-test	1.1430	0.2624	no win
	RF + BoW	t-test	5.1108	0.0000	win
	LR + BoW	t-test	5.9753	0.0000	win
	LSVM + TF-IDF	t-test	8.5484	0.0000	win
	LR + TF-IDF	t-test	18.1905	0.0000	win
	RF + W2V-1	t-test	82.9814	0.0000	win
	LSVM + W2V-1	t-test	97.2148	0.0000	win
	LR + W2V-1	t-test	100.2249	0.0000	win
LinearSVM + BoW					
	RF + TF-IDF	t-test	-1.1430	0.2624	no win
	RF + BoW	t-test	3.0751	0.0046	win
	LR + BoW	Wilcoxon	5.5000	0.0000	win
	LSVM + TF-IDF	t-test	10.2416	0.0000	win
	LR + TF-IDF	t-test	17.3463	0.0000	win
	RF + W2V-1	t-test	80.8924	0.0000	win
	LSVM + W2V-1	t-test	96.4483	0.0000	win
	LR + W2V-1	t-test	98.8127	0.0000	win

Table 5: Head-to-head statistical tests for the top two models

RandomForest + TF-IDF and LinearSVM + BoW emerged as the the pipelines with statistically superior performance, but the final choice fell on RandomForest + TF-IDF as the binary classifier to feed into the overall binary-plus-multiclass pipeline. Random Forest offers native global interpretability through its built-in feature-importance scores and seamless local explanations via TreeInterpreter, while TF-IDF delivers a normalized feature space where each term's weight can be assessed directly. In contrast, while Logistic Regression offers global interpretability through linear coefficients (coef), these are only reliable when features are standardized that is a condition not guaranteed with BoW. Another things to notice is that: the Word2Vec representations underperformed compared to BoW and TF-IDF, confirming that for very short, informal texts such as tweets, simpler frequency-based encodings often produce better results.

3.2 Multiclass Classification Results

Experimental Setup and Grid Search. As described in Section 2.4, three classifiers Random Forest, Logistic Regression, and Linear SVM were evaluated in combination with three vectorization methods: Bag of Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and custom-trained Word2Vec embedding (W2V-1). This led to a total of 9 distinct model configurations. The grid of hyperparameters used during tuning was the following:

- **Logistic Regression:**
  - `C`: {0.01, 0.1, 1}
- **Linear SVM:**
  - `C`: {0.01, 1, 10}
- **Random Forest:**
  - `n_estimators`: {100, 500, 1000}
  - `max_depth`: {None, 20}
  - `random_state`: 42

Model evaluation Table 6 reports the optimal hyperparameters that were selected via nested cross-validation for each vectorizer–classifier pipeline in the multiclass setting. For each configuration, we show the mode of the best parameters found across the ten outer folds.

Configuration	**Chosen Parameters**
BoW + LogisticRegression	`’model__C’: 1`
BoW + RandomForest	`’model__max_depth’: None, ’model__n_estimators’: 500, ’model__random_state’: 42`
BoW + LinearSVM	`’model__C’: 1`
TF-IDF + LogisticRegression	`’model__C’: 1`
TF-IDF + RandomForest	`’model__max_depth’: None, ’model__n_estimators’: 1000, ’model__random_state’: 42`
TF-IDF + LinearSVM	`’model__C’: 1`
W2V-1 + LogisticRegression	`’model__C’: 0.1`
W2V-1 + RandomForest	`’model__max_depth’: 20, ’model__n_estimators’: 1000, ’model__random_state’: 42`
W2V-1 + LinearSVM	`’model__C’: 0.01`

Table 6: Chosen parameters for each configuration (mode over 10 folds) – Multiclass

Table 7 then summarizes the key evaluation metrics—mean accuracy, precision, recall and F1-macro—along with their standard deviations across the outer folds. This allows for a comprehensive comparison of both the average performance and the consistency of each model.

Vectorizer	Classifier	Mean Acc.	Mean Prec.	Mean Rec.	Mean F1	Std Acc.	Std Prec.	Std Rec.	Std F1
TF-IDF	RandomForest	0.997650	0.997660	0.997650	0.997652	0.000726	0.000723	0.000736	0.000730
BoW	RandomForest	0.997575	0.997580	0.997578	0.997576	0.000807	0.000797	0.000812	0.000805
BoW	LogisticRegression	0.997500	0.997533	0.997469	0.997499	0.000733	0.000712	0.000756	0.000736
BoW	LinearSVM	0.997424	0.997466	0.997406	0.997434	0.000537	0.000526	0.000551	0.000539
TF-IDF	LinearSVM	0.997399	0.997442	0.997372	0.997405	0.000889	0.000869	0.000901	0.000886
TF-IDF	LogisticRegression	0.996099	0.996163	0.996053	0.996102	0.001480	0.001451	0.001499	0.001478
W2V-1	LinearSVM	0.989823	0.989897	0.989819	0.989852	0.002050	0.002053	0.002066	0.002059
W2V-1	LogisticRegression	0.989698	0.989759	0.989693	0.989721	0.002380	0.002363	0.002377	0.002371
W2V-1	RandomForest	0.967018	0.967357	0.967289	0.967150	0.003379	0.003352	0.003367	0.003378

Table 7: Nested CV Results: mean metrics and standard deviations – Multiclass

Model selection Table 8 shows the pairwise victory matrix across all model–vectorizer combinations. A value of 1 indicates that the model in the row significantly outperformed the model in the column according to a statistical test, while 0 denotes no significant win.

	RF + TF-IDF	RF + BoW	LR + BoW	LSVM + BoW	LSVM + TF-IDF	LR + TF-IDF	LSVM + W2V-1	LR + W2V-1	RF + W2V-1
RF + TF-IDF	0	1	0	1	1	1	1	1	1
RF + BoW	0	0	0	0	0	1	1	1	1
LR + BoW	0	0	0	0	0	1	1	1	1
LSVM + BoW	0	0	0	0	0	1	1	1	1
LSVM + TF-IDF	0	0	0	0	0	1	1	1	1
LR + TF-IDF	0	0	0	0	0	0	1	1	1
LSVM + W2V-1	0	0	0	0	0	0	0	0	1
LR + W2V-1	0	0	0	0	0	0	0	0	1
RF + W2V-1	0	0	0	0	0	0	0	0	0

Table 8: Pairwise victory matrix (1 = win, 0 = no win)

Table 9 reports the total number of head-to-head victories per pipeline. A higher number indicates stronger overall performance in the statistical comparison.

Model	Victories
RandomForest + TF-IDF	7
RandomForest + BoW	4
LogisticRegression + BoW	4
LinearSVM + BoW	4
LinearSVM + TF-IDF	4
LogisticRegression + TF-IDF	3
LinearSVM + W2V-1	1
LogisticRegression + W2V-1	1
RandomForest + W2V-1	0

Table 9: Number of head-to-head victories per model

Table 10 lists the statistical test results for the top-ranked pipeline. For each comparison, the test type, statistic, p-value, and outcome (win or no win) are reported.

Winner	Opponent	Test	Stat.	p-value	Outcome
RandomForest + TF-IDF					
	RF + BoW	t-test	2.2067	0.0354	win
	LR + BoW	t-test	1.8119	0.0804	no win
	LSVM + BoW	Wilcoxon	78.0000	0.0393	win
	LSVM + TF-IDF	t-test	2.6167	0.0140	win
	LR + TF-IDF	t-test	9.2252	0.0000	win
	LSVM + W2V-1	t-test	27.6067	0.0000	win
	LR + W2V-1	t-test	23.1626	0.0000	win
	RF + W2V-1	t-test	55.1649	0.0000	win

Table 10: Head-to-head statistical tests for RandomForest + TF-IDF

RandomForest + TF-IDF emerged as the pipeline with statistically superior performance so it represents the final choice as the multiclasse classifier to feed into the overall binary-plus-multiclass pipeline. Like said before, Random Forest offers native global interpretability through its built-in feature-importance scores and seamless local explanations via TreeInterpreter, while TF-IDF delivers a normalized feature space where each term's weight can be assessed directly. As reported in binary results, the Word2Vec representations underperformed compared to BoW and TF-IDF, confirming, once again, that for very short, informal texts such as tweets, simpler frequency-based encodings often produce better results.

3.3 Pipeline Evaluation

In this final evaluation, the full classification pipeline was tested in its complete form. The system first applies a binary classifier to distinguish between cyberbullying and non-cyberbullying tweets. The tweets identified as cyberbullying (true positives) are then passed to a second classifier, which assigns a specific type of cyberbullying.

This two-stage structure reflects a realistic moderation workflow, where content is first flagged as problematic and then further analyzed for nature. Confusion Matrixes were computed separately for each stage, and also globally to assess the end-to-end system.

3.3.1 Binary Stage Performance.

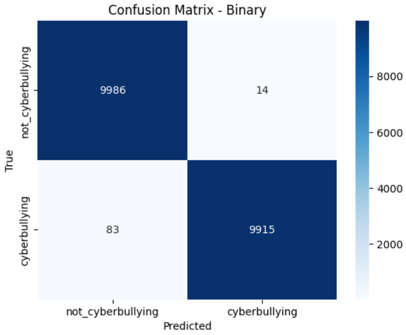


Figure 4: Confusion matrix for binary stage.

3.3.2 Multiclass Stage Performance (on true positives).

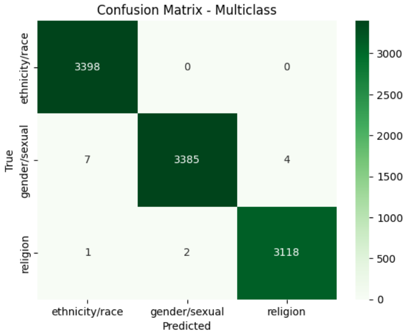


Figure 5: Confusion matrix for multiclass stage (on binary true positives).

Overall Pipeline Performance. By combining the two stages, the overall pipeline correctly classified 19887 tweets out of 19998, resulting in an **end-to-end accuracy of** 0.9944. This metric reflects the proportion of tweets that were both correctly identified as cyberbullying/non-cyberbullying, and, when applicable, correctly categorized into one of the fine-grained classes.

3.4 Explainability Insights

To support transparency and user trust, the system integrates both global and local explainability modules. These provide insight into the reasoning of the classifier, helping developers and stakeholders understand what features drive predictions and how individual decisions are made.

Global Insights. To gain a high-level understanding of the model's behavior, two complementary techniques were used.

Pattern mining was applied to identify the most frequent and informative terms associated with each cyberbullying class. These word associations provide an intuitive view of how language patterns differ across abuse categories. To better understand the structure of these patterns, the distribution of closed and maximal itemsets across the classes was analyzed.

Class	**Number of closed itemsets**	**Number of maximal itemsets**
Multiclass Target		
Ethnicity/race	27	18
Religion	58	34
Gender/sexual	14	7
Binary Target		
cyberbullying	21	19
not_cyberbullying	4	4

Table 11: Distribution of closed and maximal itemsets across multiclass and binary targets

The analysis shows that the first two categories are the ones with the largest number of closed itemsets (27 and 58 respectively), indicating a greater lexical richness and variability in the way users engage in bullying related to these aspects. In contrast, the "gender/sexual" class exhibits very few patterns (only 14 closed itemsets), suggesting a lower consistency or a higher heterogeneity in the associated language. Focusing on maximal itemsets, the most representative and non-redundant patterns, "religion" emerges with the highest number (34). In stark contrast, the gender/sexual category stands out for its sparse pattern structure, with only 14 closed and 7 maximal itemsets. This scarcity may be interpreted in two opposing ways. On one hand, it could indicate a higher linguistic variability, with users employing a wide range of terms that rarely form recurring combinations. On the other hand, it may instead reflect a highly standardized form of abuse, relying on a limited set of offensive terms that recur individually rather than in combinations, thus narrowing the semantic space and reducing the emergence of multi-term patterns. Support for the latter interpretation is provided by the frequency distribution of the top terms in this class: as shown in the figure, a small number of highly explicit words dominate the lexical landscape, with the leading term appearing over 11,000 times—far above the rest.

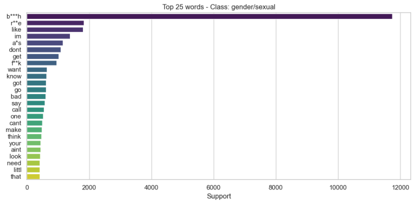
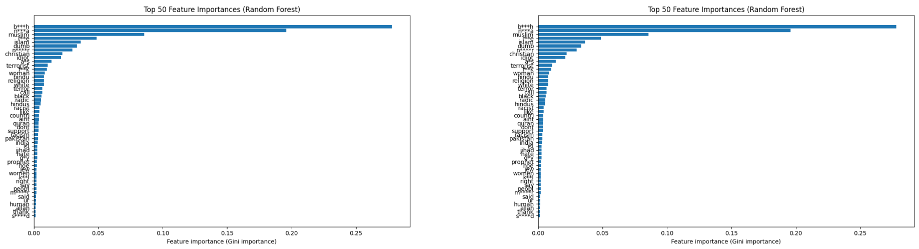


Figure 6: Top 25 words in the gender/sexual class ranked by support.

This strong lexical skew suggests that gender/sexual abuse in this dataset relies heavily on a few well-known slurs, often used in isolation, which reduces the diversity and co-occurrence necessary for richer itemset formation. In both cases, the model may face challenges: either due to lexical dispersion or due to the lack of complex, distinctive structures to anchor its predictions.
When shifting the focus to the binary setting, where the goal is to distinguish between cyberbullying and non-cyberbullying texts, a different pattern emerges. The cyberbullying class yields 21 closed itemsets and 19 maximal ones, indicating a high compression rate and limited redundancy. More strikingly, the not_cyberbullying class produces only 4 closed itemsets, all of which are also maximal. This perfect overlap highlights an extremely sparse and flat pattern structure, likely due to the lexical heterogeneity and general nature of non-abusive content. In such cases, the absence of recurring co-occurrences prevents the emergence of structured itemsets, making this class inherently less amenable to pattern-based discrimination. As a result, most meaningful structures arise only from the cyberbullying class, where offensive or harmful expressions are more likely to cluster in consistent and repeated forms. This reinforces the idea that pattern mining approaches, particularly those targeting redundancy-aware or closed/maximal itemsets, are more effective when applied to abusive content, while neutral or generic language tends to resist such compression.

In a second instance , the `feature_importances_` attribute of the Random Forest classifiers trained on TF-IDF features was analyzed. The top 50 most important terms are reported in Figure 7, showing which words had the greatest influence on the model's overall decision-making.



(a) Binary

(b) Multiclass

Figure 7: Top 50 most important features based on Random Forest `feature_importances_`.

This analysis further confirms the observations drawn from the frequent pattern mining. The concentration of highly salient terms, almost exclusively associated with the *cyberbullying* class, in the Random Forest feature importances suggests that the model does not aim to symmetrically separate the two classes. Instead, it focuses on capturing the compact and strongly lexicalized structure of class 1. This is consistent with the earlier finding that cyberbullying yielded 21 closed itemsets (19 of which were maximal), whereas not_cyberbullying produced only 4 all maximal indicating a flat and sparse pattern space. Together, these elements reinforce the idea that the model primarily learns to detect cyberbullying, relying on a small set of strongly predictive terms, while the neutral and heterogeneous nature of the other class limits its representation in both the pattern space and the learned decision boundary, an approach that closely mirrors human behavior, and which was precisely the intended goal of this work.

Local Interpretability. To explain individual predictions produced by the multiclass classifier, the TreeInterpreter tool was employed. This method breaks down the decision of a tree-based model into the contributions of each input feature, allowing for a detailed analysis of how specific terms influenced the classification outcome.

One particularly interesting insight emerging from this analysis concerns the role of feature absence. In addition to quantifying the positive or negative impact of words that appear in the input text, TreeInterpreter also highlights how the absence of certain terms can affect the prediction. In the context of TF-IDF representations, this means that words which typically characterize other classes — but are missing from the current input — can decrease the likelihood of those classes being predicted, effectively reinforcing the assigned label.

For example, consider the tweet:

```
"I don't respect your religion, it's all nonsense."
```

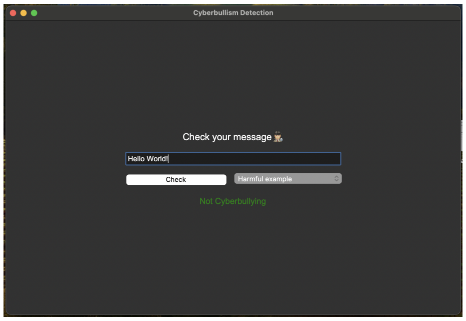
TreeInterpreter reveals that terms such as `"religion"`, `"respect"`, and `"nonsense"` made strongly positive contributions toward the prediction of the `religion` cyberbullying class. These words are often associated with dismissive or mocking discourse targeting religious identity or belief systems, which are strong indicators for this category.

At the same time, the absence of features typically linked to other classes, such as gendered terms, racial or ethnic identifiers, negatively impacted their respective class probabilities. This absence, combined with the presence of highly indicative features for the `religion` class, increased the model's overall confidence in its prediction.

This kind of interpretability not only clarifies why a particular label was assigned, but also uncovers subtle linguistic dynamics that may not be immediately apparent to a human reviewer.

4 Graphical User Interface

The graphical user interface (GUI) of the *Cyberbullying Detection* system is designed to be user-friendly while providing interpretable results. It is implemented using `Tkinter` and consists of two main views: the classification interface and the explanation interface.



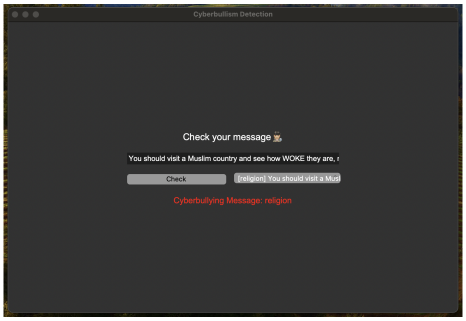
Check your message 🕵

Hello World!

Not Cyberbullying

Figure 8: Main interface: the message "Hello World!" is correctly classified as safe.

In the main window (Figure 8), the user is prompted to enter a message into a text field. Upon clicking the `Check` button, the system analyzes the content using a binary classifier trained to distinguish cyberbullying from harmless messages. If the message is safe, the interface displays a green label saying *Not Cyberbullying*. Otherwise, it shows a red warning along with the specific type of cyberbullying detected (e.g., *ethnicity*, *sexual*, etc.), as illustrated in Figure 9.



Check your message 🕵

You should visit a Muslim country and see how WOKE they are, r

[religion] You should visit a Musl

Cyberbullying Message: religion

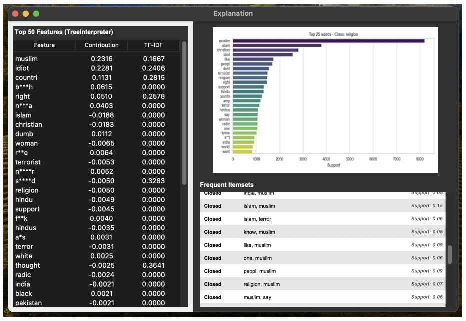
Figure 9: Example of a detected cyberbullying message classified under the "ethnicity" category.

To enhance interpretability, the application includes an additional explanation panel (Figure 10) that provides insights into the model's decision process. This window is divided into three sections:

- **Left panel – TreeInterpreter analysis:** displays the top 50 textual features (word stems) that contributed to the classification. Each row includes the word, its contribution score (positive or negative), and its TF-IDF value. This breakdown helps understand which terms were most influential in the prediction.
- **Top-right panel – Class word distribution:** a bar chart shows the 25 most frequent words associated with the predicted cyberbullying category.

This gives a visual representation of common vocabulary within the class.

- **Bottom-right panel – Closed/maximal Itemsets:** this section lists the most relevant itemsets mined from the dataset for the predicted class. These itemsets highlight frequent co-occurring patterns of words that characterize each type of cyberbullying. In addition to the itemsets is specified the type, closed or maximal, and the relative support of them.



Top 50 Features (TreeInterpreter)

Feature	Contribution	TF-IDF
muslim	0.2316	0.1667
idiot	0.2281	0.2406
countri	0.1131	0.2815
b***h	0.0615	0.0000
right	0.0510	0.2578
n***a	0.0403	0.0000
islam	-0.0188	0.0000
christian	-0.0183	0.0000
dumb	0.0112	0.0000
woman	-0.0065	0.0000
r**e	0.0064	0.0000
terrorist	-0.0053	0.0000
n****r	0.0052	0.0000
s****d	-0.0050	0.3283
religion	-0.0050	0.0000
hindu	-0.0049	0.0000
support	-0.0045	0.0000
f**k	0.0040	0.0000
hindus	-0.0035	0.0000
a*s	0.0031	0.0000
terror	-0.0031	0.0000
white	0.0025	0.0000
thought	-0.0025	0.3641
radic	-0.0024	0.0000
india	-0.0021	0.0000
black	0.0021	0.0000
pakistan	-0.0021	0.0000

Frequent Itemsets

Closed	india, muslim	Support: 0.05
Closed	islam, muslim	Support: 0.15
Closed	islam, terror	Support: 0.06
Closed	know, muslim	Support: 0.05
Closed	like, muslim	Support: 0.09
Closed	one, muslim	Support: 0.06
Closed	peopl, muslim	Support: 0.09
Closed	religion, muslim	Support: 0.07
Closed	muslim, say	Support: 0.06

Figure 10: Explanation interface: feature contributions (left), top words by support (top-right), and association rules (bottom-right).