Exploring Class Imbalance in Hate Speech and Offensive Language Detection: An Examination of Its Effect on Model Accuracy

Daphne Babirye (2023)

Table of Contents

Table of Tables

Abstract

This research addresses the growing issue of hate speech and offensive language on social media platforms, particularly Twitter, which has significant implications for individual well-being and societal cohesion. Given the sheer volume of content generated on this platform, manual moderation and community reporting are insufficient. Consequently, we turn to automated methods for identifying and moderating such language. Specifically, this study focuses on the development of a multi-classification machine learning model for detecting hate speech and offensive language, and assessing the impact of imbalanced data on model performance, as well as exploring effective strategies for dealing with such imbalances.

The study utilized three machine learning models - Naive Bayes, Support Vector Machines (SVM), and Random Forest - applied to both balanced and imbalanced datasets. Model performances were assessed using precision, recall, F1-score, and accuracy. Our findings indicated that all models struggled with correctly classifying the minority class in the imbalanced dataset, but showed improved precision, recall, and F1-scores when the data was balanced. The Random Forest model emerged as the most effective, yielding an overall accuracy of 0.89 and 0.90 for the imbalanced and balanced test data respectively. However, the study further highlighted the importance of considering multiple evaluation metrics beyond accuracy, especially in the context of imbalanced data, providing a more comprehensive assessment of a model's performance. For future work, exploration of other data balancing techniques, feature extraction techniques, and deep learning models is suggested.

1. Introduction

The emergence of social media networks, such as Twitter, has revolutionized communication by providing a seamless and real-time way for individuals to express their thoughts, opinions, and sentiments (Kaplan & Haenlein, 2010). Twitter, with its 280-character limit, has become a microblogging platform where people across the world can engage in open discussions on various topics. However, this openness has also paved the way for the spread of negative discussions, such as hate speech and offensive language, which has become an issue of significant concern (Davidson et al., 2017).

Hate speech is commonly defined as any form of expression that belittles or insults individuals or groups based on specific attributes such as race, complexion, ethnic background, gender, sexual orientation, nationality, religion, or other distinguishing factors (wikipedia, 2023). Offensive language, on the other hand, may not target specific groups or individuals but still contains elements that are generally deemed socially or morally unacceptable (Waseem & Hovy, 2016). Both hate speech and offensive language can lead to harmful consequences, including psychological harm to targeted individuals, incitement of violence, and a deterioration of social cohesion (Barbara Perry & Patrik Olsson, 2009).

While multiple strategies have been put into action to curb the spread of such content on social media, including manual moderation and community reporting, these measures are not entirely effective given the sheer volume of content generated daily on these platforms (Schmidt & Wiegand, 2017). As a result, there is a growing need for automated methods of identifying and moderating hate speech and offensive language. Machine learning, and in particular multi-classification models, present promising avenues for this task (Fortuna & Nunes, 2018).

This research aims to develop a multi-classification model that can efficiently identify hate speech and offensive language in Tweets. We believe that the development and deployment of such models can significantly improve the moderation and management of content on Twitter, thus making it a safer and more inclusive platform.

1.1.Related work

Advancements in machine learning technologies such as Natural Language Processing (NLP) has stimulated a surge in research concerning hate speech and offensive language detection, especially post-2020.

Khouloud Mnassri, Praboda Rajapaksha, Reza Farahbakhsh, Noel Crespi, (2023) addressed the challenge of detecting hate speech and offensive language by proposing a multi-task joint learning model integrating emotional features alongside traditionally annotated dataset features. Utilizing BERT and mBERT transformer models, their shared encoder was designed to jointly learn abusive content detection and emotional features, increasing data efficiency and reducing overfitting. Despite modest performance improvement in hate speech detection (3% over baseline models) and no significant performance enhancement in offensive language detection, their approach showed potential by reducing false positives in both tasks, underscoring the potential of leveraging emotional knowledge in this domain.

The application of deep learning methods in this area has further improved the performance of these models. Badjatiya et al., (2017) demonstrated the effectiveness of using Long Short-Term Memory (LSTM) networks for detection of hate speech, while Gambäck & Sikdar, (2017) applied Convolutional Neural Networks (CNN) to the same task, achieving promising results.

However, as pointed out by Fortuna & Nunes (2018), the issue of imbalanced data in hate speech detection is still a significant challenge that affects the performance of these models. This reflects the reality that hate speech instances often represent a small fraction of the data in real-world scenarios, leading to models that are biased towards non-hateful classes.

1.2.Data Description

The data that will be used in this study on hate speech and offensive language detection is text data in form of 24,783 tweets from the twitter website. The is publicly available on the Kaggle website(https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset?resource=download). The dataset consists of 7 columns, with the first columns indicating the count of CrowdFlower (CF) participants who categorized every tweet while the following 3 columns indicate the number of CF users who coded a particular tweet into a particular category. Column 5 and 6 are the columns that make up the dataset for the study. The class column indicates the target and corresponding classes. The was categorized into the following classes: hate speech: "0", offensive language: "1", or neither: "2" (Andrii Samoshyn, 2020). The data set contains 7 columns . The tweet column provided the text data for the analysis.

Table 1: Extract of the First 5 Columns of the dataset

	count	Hate speech	Offensive language	neither	class	tweet
0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	6	0	6	0	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...

The dataset is imbalanced, with offensive language category having the highest percentage of tweets (77%), followed by the neither category (16.8%), while the hate speech category had the least number of tweets with only 5.8 percent of the total number of tweets (***Table 2***).

Table 2: Count of Tweets By Category

Category	Tweet	% Tweets
hate_speech	1430	5.770084
neither	4163	16.7978
offensive_language	19190	77.43211

Given the subject matter of the investigation, it's crucial to acknowledge that the dataset used in the study includes text which may be perceived as discriminatory, misogynistic, homophobic, or broadly offensive.

1.3.Objectives of the study

The objectives of the study are to;

- Develop a multi-classification model that can efficiently identify hate speech and offensive language from social media posts and explore ways of dealing with imbalanced data.
- Investigate the impact of imbalanced data on the performance of our multi-classification model and to explore effective strategies for handling such imbalances

2. Research

An additional challenge in developing a machine learning model for identifying hate speech and offensive language in tweets is dealing with class imbalance, a common issue in many real-world classification problems (Japkowicz & Stephen, 2002). In our context, class imbalance refers to a disproportion between the number of tweets that contain hate speech or offensive language and those that do not. Given the pervasiveness of neutral or non-offensive content compared to hate speech or offensive content, models trained on such imbalanced datasets may be biased towards the majority class and perform poorly on the minority class (Chawla et al., 2002).

Overcoming the issue of imbalanced data in machine learning can significantly enhance model performance. Several strategies such as synthetic minority over-sampling techniques (SMOTE), under-sampling the majority class and over-sampling the minority class have been proposed to deal with imbalanced data (Chawla et al., 2002); (Han et al., 2005). Furthermore, cost-sensitive learning and the application of appropriate evaluation metrics, such as F1-score, Area Under the Receiver Operating Characteristic Curve (AUC-ROC), and the Matthews correlation coefficient, which account for class imbalances, can aid in model evaluation (Charles Elkan, 2001; Powers, 2020).

In light of this, this study also aims to investigate the impact of imbalanced data on the performance of our multi-classification model and to explore effective strategies for handling such imbalances. This dual focus not only addresses the challenge of identifying hate speech and offensive language but also contributes to the broader discourse on handling imbalanced data in machine learning applications.

SMOTE is a method that is widely used for addressing class imbalance in datasets, as it enhances representation of minority class instances by generating synthetic samples. SMOTE was introduced by Chawla et al., (2002), and operates by selecting similar instances in the feature space, and then creating new samples along the lines between these instances. This process broadens the decision surface, resulting in improved model performance on minority class instances.

SMOTE has been chosen for this study due to its several advantages. It not only increases minority class instances, but also enhances decision boundaries' diversity by creating synthetic instances. This leads to a more robust classifier that performs well on both majority and minority class instances. SMOTE's effectiveness in improving classifier performance in the face of class imbalance has been widely recognized in various studies, such as the one by Joloudari et al.(2023) and Batista et al. (2004).

3. Methodology

3.1.Data Pre-processing

3.1.1. Data Cleaning

In order to prepare the data for modelling, a number of pre-processing steps were implemented as provided below;

The data was first checked for missing values and it was found that the data had no missing values. Checking for missing values is crucial as they could cause issues during model training or lead to biased results due to incomplete information. If found, suitable strategies like deletion or imputation should be adopted. The data was also checked for duplicates and it was found that there were no duplicates. Checking for duplicates is equally important to avoid model overfitting to repeated instances, thereby ensuring better generalization to unseen data. In essence, these steps ensure that the model has high-quality, diverse, and complete data for learning.

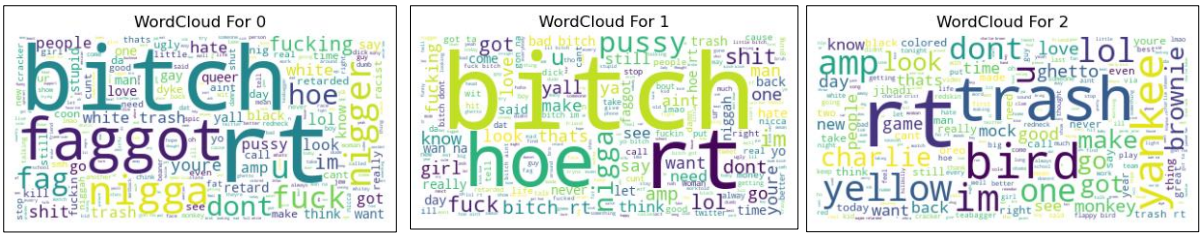
The text was first converted to lowercase to ensure uniformity in the dataset. It's essential because in text analytics, the algorithm treats 'Word' and 'word' as two separate entities due to case difference, which could lead to inaccurate analysis. The next step involved removing numbers from the text. The rationale is that numbers often do not carry significant meaning in text classification tasks, especially if they are used in different contexts. Punctuation was removed next, as punctuation marks generally do not carry meaningful information for text classification tasks. Also, removing them can reduce the complexity of the dataset. All extra spaces and newline characters were then removed to reduce noise in the dataset and ensure each document is a single string entity.

The text was then tokenized into individual words. Tokenization is the process of breaking down text into words, phrases, symbols, or other meaningful elements (tokens), which gives structure to the previously unstructured text. The 'stop words', often the most common words in a language (e.g., 'is', 'the', 'and'), were removed. These words usually do not carry important meaning and are removed to reduce the dataset's dimensionality. Lemmatization was performed to reduce words to their base or root form (e.g., 'running' to 'run'). It helps in grouping together different modified forms of a word so they can be analyzed as a single item, reducing the complexity of the data. Finally, the cleaned, lemmatized tokens were joined back together to form a single string of text. This is generally done to ease the use of cleaned data in machine learning models, which often expect input in the form of a single string per document.

These pre-processing steps help to clean and standardize the data, making it easier for machine learning algorithms to extract meaningful patterns.

Figure 1provides a visualisation of the key words in each of the classes of the tweets, hate speech: “0”, offensive language: “1”, or neither: “2”.

Figure 1: Data Visualisation of key words for each of the classes (word Clouds)



3.1.2. Split the data into train and test

In this stage of data pre-processing , the main step was the splitting of the dataset into training and testing subsets. The objective of this process is to train your model on one portion of your data (the training set), and then evaluate its performance on unseen data (the test set), which helps avoid overfitting.

The train_test_split function from the sklearn.model_selection module was used. This function shuffles the dataset and then splits it into two parts according to a given ratio. In this study, 80 percent of the data was split to training data and 20 percent to the test set. Stratified sampling was used to preserve the distribution of target values in both the training and test sets. This is important for maintaining a representative sample, particularly when dealing with imbalanced classes, as it ensures that each class is adequately represented in both sets. The random state was set to 283 for reproducibility. This means that despite the random process of splitting the data, the same train-test split will be obtained each time the code is run. This is essential for evaluating how various models perform on the same dataset **Table *3*** indicates the distribution of the imbalanced data across the train and test data sets.

Table 3: Distribution of Data Across Train and Test Sets

	Train Set	Test Set
1	15352	3838
2	3330	833
0	1144	286

3.1.3. Word encoding

The Term Frequency-Inverse Document Frequency (TF-IDF) encoding was used to convert the text data into numerical form that can be understood and processed by machine learning algorithms.

The this particular encoder was chosen because it captures both the frequency of each word in each document (Term Frequency) and the inverse of the frequency of the word across all documents (Inverse Document Frequency). This allows the model to not only understand the context and semantics of the text data, but also to assign more weight to words that are important and unique to each document. This is particularly useful in text classification tasks, where the presence or absence of specific words can often be a strong indicator of the class of the document.

To undertake the encoding task, the TF-IDF vectorizer was fit on the training data to learn the vocabulary. Both the training and test data were then transformed into TF-IDF encoded vectors using the transform method. This method converts each tweet into a vector where each

dimension corresponds to a unique word from the training vocabulary, and each value represents the importance of that word in the given tweet(scikit-learn, 2023).

3.1.4. Smote for balancing the dataset

In order to explore the impact of imbalanced and balanced data on model performance and explore, the study also implements machine learning models on a balanced dataset. The SMOTE technique which is explained in section 2 (Research)is used to balance the data.

The initial step was to create a case of a SMOTE class, with the random state set to 283 to ensure reproducibility. The training data then underwent fitting with the SMOTE object, resulting in the generation of synthetic samples from the underrepresented class. This process continued until the number of synthetic samples equalled that of the overrepresented class.. The class distribution was checked both before and after the application of SMOTE to verify the successful execution of the over-sampling operation (Table 4). The imbalanced learn library was used to perform this operation.

Table 4: Samples Per Class Before and After Application of SMOTE

Class	Samples Before SMOTE	Samples After SMOTE
0	1144	15352
1	15352	15352
2	3330	15352

3.2.Methods

In the initial phase of the multi-class classification task, three models were implemented: Support Vector Machines (SVM), Random Forest, and Naive Bayes. Naive Bayes is a simple but effective probabilistic classification model, that is ideal for large dimensional datasets. SVMs, being powerful and flexible, excel at handling high-dimensional data and find the optimal hyperplane for class separation. Random Forest, an ensemble method, is robust and proficient in handling numerous features, offering reliable feature importance estimates. These models were employed to assess their individual performance on the classification task.

The models were applied to both the balanced and imbalanced datasets to assess the impact of imbalanced classes on model performance.

Grid Search was used for hyperparameter optimization on the best model for both the balanced and imbalanced datasets. The Random Forest model was the best model for both the imbalaced and balanced datasets and combinations of a number of parameters were explored to obtain the best model. For the imbalanced data, the following combinations were explored; 'n_estimators' (100, 200), 'max_depth' (None, 5), 'min_samples_split' (2, 5), 'min_samples_leaf' (1, 2), and a fixed 'random_state' (283). The best parameters found were 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100, 'random_state': 283, yielding a top cross-validation score of about 0.886. For the balanced data, combinations of 'n_estimators' (100, 200), 'max_depth' (None, 5), 'min_samples_split' (2, 5), 'min_samples_leaf' (1, 2), and a set 'random_state' (283) were explored. The best parameters were 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200, 'random_state': 283, giving a top cross-validation score of approximately 0.965.

4. Evaluation

4.1.Model Exploration

As indicate in the previous section, the study explored three models; Vector Machines (SVM), Naive Bayes, Support, and Random Forest. The results provided indicate the performance of the three models, on both imbalanced and balanced datasets. The evaluation metrics used are Precision, Recall, F1-Score, and Accuracy.

Precision refers to the ratio of correctly identified positive instances within the predicted class (0, 1 or 2) of the model, while recall measures the ratio of accurately identified positive instances out of the total actual positives. On the other hand, The F1-Score, a metric that seeks to strike a balance between Precision and Recall, is calculated as the harmonized mean of the two measures.. It's particularly useful in the case of class imbalance. Lastly, accuracy is the proportion of total predictions that were correct. It is useful when the target class is well balanced but can be misleading when the class distribution is imbalanced.

In the imbalanced Data the Naive Bayes model achieved an accuracy of 79%. However, its performance was very poor on class 0, with no correct predictions (precision and recall scores are both 0). Its F1-score for class 2 was also relatively low (0.20), suggesting that the model had difficulties distinguishing these classes in the imbalanced dataset. The Support Vector Machines (SVM) model outperformed Naive Bayes on the imbalanced data, achieving an overall accuracy of 80%. It performed reasonably well for class 1 and 2, but had a low F1-score for class 0 (0.11), indicating the model was also not able to handle the imbalance well. Lastly, the random forest model had better performance than Naive Bayes and comparable performance with SVM. However, its performance for class 0 was still poor (F1-score 0.17), suggesting difficulties with the imbalanced data (**Table *5***).

Table 5: Imbalanced Data Results During Model Exploration

Model	Class	precision	recall	f1-score	Accuracy
Naïve Bayes	**0**	0.00	0.00	0.00	0.79
	1	0.79	1.00	0.88	
	2	0.97	0.11	0.20	
Support Vector Machines	**0**	0.73	0.06	0.11	0.80
	1	0.91	0.97	0.94	
	2	0.84	0.84	0.84	
Random Forest	**0**	0.58	0.10	0.17	0.89
	1	0.90	0.97	0.94	
	2	0.85	0.79	0.82	

In the balanced data the naive bayes model achieved an accuracy of 92%, significantly better than in the imbalanced data. It performed well across all classes, indicating the balancing helped improve its performance. The SVM model saw substantial improvement on the balanced data, achieving an accuracy of 97%. It performed excellently across all classes, suggesting it could handle the balanced data well. Like the SVM, the Random Forest model also performed excellently on the balanced data (accuracy of 97%). The performance across all classes was also substantially improved compared to the imbalanced data (***Table 6***).

Table 6: Balanced Data Results During Model Exploration

Model	Class	precision	recall	f1-score	Accuracy
Naïve Bayes	**0**	0.87	0.99	0.92	0.92
	1	0.96	0.81	0.88	
	2	0.95	0.97	0.96	
Support Vector Machines	**0**	0.99	0.98	0.98	0.97
	1	0.95	0.97	0.96	
	2	0.97	0.97	097	
Random Forest	**0**	0.98	0.99	0.98	0.97
	1	0.98	0.94	0.96	
	2	0.96	0.99	0.98	

Overall, the improvement in performance from the imbalanced to the balanced datasets is clear. In the imbalanced dataset, all three models struggled to correctly classify the minority class (class 0), resulting in poor precision, recall, and F1-scores for this class. However, these scores improved dramatically in the balanced dataset, indicating that the models were better able to handle the class distribution. This shows the importance of addressing class imbalance in the data preparation stage of a machine learning workflow. The results also highlight the importance of evaluating model performance using a range of metrics. Accuracy alone may not provide a complete picture, particularly in the context of imbalanced datasets. By considering precision, recall, and the F1-score, we can gain a better understanding of a model's performance across different classes.

4.2.Final Model Evaluation on Test Data

4.2.1. Best Model Selection

The Random Forest model is selected as the best model due to its high precision, recall, and F1-score values across all classes for both balanced and imbalanced datasets, indicating its ability to accurately predict and cover a higher number of actual positive samples for each class as seen in section 4.1. Its superior performance, coupled with Random Forest's robustness to overfitting and effectiveness in handling high dimensional data, makes it a preferred choice for this text classification task.

4.2.2. Evaluation on Test data

In the Imbalanced dataset; Class 0 ‘s Precision was 0.58, which means when the model predicted a sample to be class 0, it was correct 58% of the time. However, recall was only 0.14, meaning it could only identify 14% of all true class 0 samples. For Class 1**,** the model performed well, with a precision of 0.90 and recall of 0.98, indicating it correctly identified 98% of all true class 1 samples and was correct 90% of the time when it predicted class 1. The model also performed reasonably for classification of class 2, with a precision of 0.88 and recall of 0.76, suggesting it correctly identified 76% of all true class 2 samples. The model's overall accuracy on the imbalanced test data was 0.89. However, the accuracy can be misleading in an imbalanced context as the model may simply be predicting the majority class well, which was the case here.

In the Balanced dataset, The model's performance improved for Class 0 with a recall of 0.34, up from 0.14 on the imbalanced dataset, showing the model could now identify 34% of all true class 0 samples. Class 1’s Precision was 0.92, and recall was 0.96, slightly lower than on the

imbalanced data. However, the balanced model was still robust in predicting class 1. For Class 2, the model maintained a good performance, with a precision of 0.86 and recall of 0.82, indicating that it correctly identified 82% of all true class 2 samples. The model's overall accuracy on the balanced test data was 0.90, a slight improvement from the imbalanced data.

From these results, we can conclude that balancing the classes improved the model's ability to predict the minority class (class 0) without significantly affecting its ability to predict the other classes. This demonstrates the importance of treating class imbalance in datasets, as it enables the model to predict all classes more effectively.

Additionally, the findings emphasize the importance of incorporating other performance metrics, namely; recall, precision, and F1-score, alongside accuracy, especially with imbalanced data, as they provide a more holistic view of the model's performance. In this case, they highlighted the model's improved performance in predicting the minority class after balancing the classes, which the accuracy metric alone could not reveal

Table 7: Classification Report of Random Forest Model on Test Data

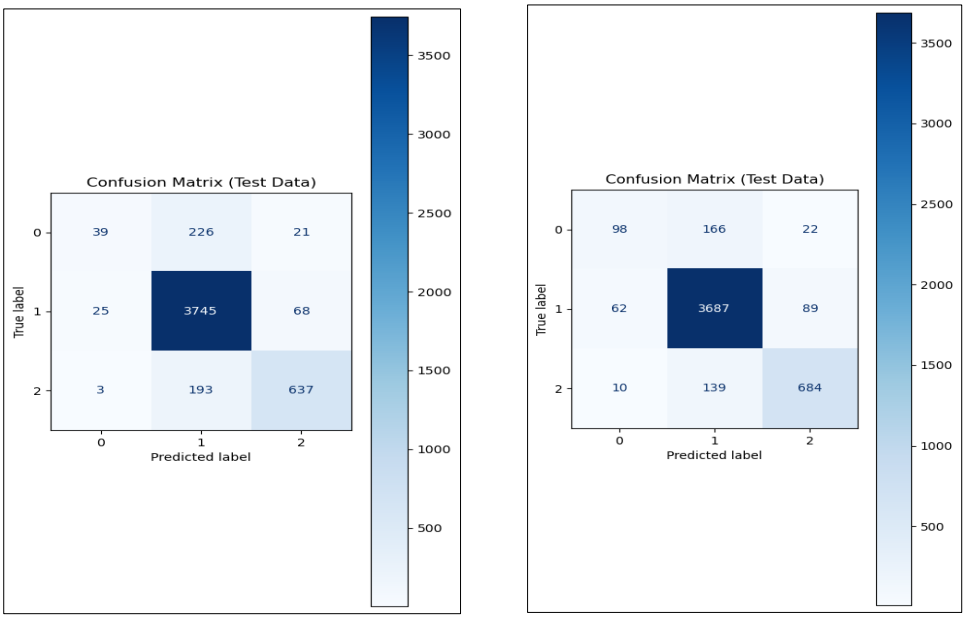
Dataset	Class	precision	recall	f1-score	Accuracy
Imbalanced Data	**0**	0.58	0.14	0.22	0.89
	1	0.90	0.98	0.94	
	2	0.88	0.76	0.82	
Balanced Data	**0**	0.58	0.34	0.43	0.90
	1	0.92	0.96	0.94	
	2	0.86	0.82	0.84	

The confusions matrices for the imbalanced and balanced datasets in

Figure 2 show details of the number of samples that were correctly predicted or incorrectly predicted when the model was tested on new data. As indicated in the previous section, there was great improvement in the prediction of class 0 and class 1 which is hate speech and neither respectively on the balanced dataset as compared to the imbalanced dataset. This indicates the improved performance of predicting the minority class when data is balanced across classes.

Figure 2: Confusion Matrices for Balanced and Imbalanced Datasets

Imbalanced Data **Balanced Data**



4.3.Conclusion and Future Work

4.3.1. Conclusion

This study explored the application of three machine learning models, Naive Bayes, Support Vector Machines (SVM), and Random Forest, in a multi-class text classification task on balanced and imbalanced datasets. The Random Forest model consistently outperformed the others in terms of precision, recall, and F1-score across all classes. The study highlighted the impact of class imbalance on model performance, with models performing markedly better on balanced data. Furthermore, balancing the classes improved the models' ability to predict minority classes, demonstrating the significance of treating class imbalance in datasets. The results also stressed the importance of using multiple evaluation metrics, including precision, recall, and F1-score, particularly with imbalanced data, as they provide a more comprehensive view of a model's performance.

4.3.2. Possible Areas of Future Work

While the study provided good results, several areas of potential exploration for future work are identified for improved performance;

i. Experimentation with other data balancing techniques: oversampling, under-sampling, or combined methods could be explored to handle the imbalanced data challenge.
ii. Other Feature Extraction Techniques: while TF-IDF was effective, alternative techniques such as Word2Vec, GloVe, or BERT could be utilized to capture semantic meanings and contextual information in the text data.

iii. Deep Learning Models: Neural networks and other deep learning models such as LSTM (Long Short Term Memory) or transformers could be employed, which might offer improved performance on such tasks.

iv. Hyperparameter Tuning: A more extensive grid search or other hyperparameter optimization methods could be performed to further enhance the model performance.

v. Multilingual and Multidomain Text: Future research could include dealing with text data from different domains or languages, which could help the model to be more robust and generalized.

5. References

Andrii Samoshyn. (2020). *Hate Speech and Offensive Language Dataset* [Corporate]. Kaggle. https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset

Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). Deep Learning for Hate Speech Detection in Tweets. *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*, 759–760. https://doi.org/10.1145/3041021.3054223

Barbara Perry & Patrik Olsson. (2009). *Cyberhate: The globalization of hate: Information & Communications Technology Law: Vol 18, No 2*. https://www.tandfonline.com/doi/abs/10.1080/13600830902814984

Charles Elkan. (2001). *(PDF) The Foundations of Cost-Sensitive Learning*. Department of Computer Science and Engineering 0114 University of California, San Diego.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. *Proceedings of the International AAAI Conference on Web and Social Media*, *11*(1), Article 1. https://doi.org/10.1609/icwsm.v11i1.14955

Fortuna, P., & Nunes, S. (2018). A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys*, *51*(4), 85:1-85:30. https://doi.org/10.1145/3232676

Gambäck, B., & Sikdar, U. K. (2017). Using Convolutional Neural Networks to Classify Hate-Speech. *Proceedings of the First Workshop on Abusive Language Online*, 85–90. https://doi.org/10.18653/v1/W17-3013

Han, H., Wang, W.-Y., & Mao, B.-H. (2005). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In D.-S. Huang, X.-P. Zhang, & G.-B. Huang (Eds.), *Advances in Intelligent Computing* (Vol. 3644, pp. 878–887). Springer Berlin Heidelberg. https://doi.org/10.1007/11538059_91

Joloudari, J. H., Marefat, A., Nematollahi, M. A., Oyelere, S. S., & Hussain, S. (2023). Effective Class-Imbalance Learning Based on SMOTE and Convolutional Neural Networks. *Applied Sciences*, *13*(6), Article 6. https://doi.org/10.3390/app13064006

Kaplan, A. M., & Haenlein, M. (2010). Users of the world, unite! The challenges and opportunities of Social Media. *Business Horizons*, *53*(1), 59–68. https://doi.org/10.1016/j.bushor.2009.09.003

Khouloud Mnassri, Praboda Rajapaksha, Reza Farahbakhsh, Noel Crespi. (2023). *Khouloud Mnassri, Praboda Rajapaksha, Reza Farahbakhsh, Noel Crespi Samovar, Telecom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France—Google Search*. Samovar, Telecom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France. https://www.google.com/search?client=firefox-b-d&q=Khouloud+Mnassri%2C+Praboda+Rajapaksha%2C+Reza+Farahbakhsh%2C+Noel+Crespi+Samovar%2C+Telecom+SudParis%2C+Institut+Polytechnique+de+Paris%2C+91120+Palaiseau%2C+France

Powers, D. M. W. (2020). *Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation* (arXiv:2010.16061). arXiv. https://doi.org/10.48550/arXiv.2010.16061

Schmidt, A., & Wiegand, M. (2017). A Survey on Hate Speech Detection using Natural Language Processing. *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, 1–10. https://doi.org/10.18653/v1/W17-1101

scikit-learn. (2023). *6.2. Feature extraction*. Scikit-Learn. https://scikit-learn/stable/modules/feature_extraction.html

Waseem, Z., & Hovy, D. (2016). Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. *Proceedings of the NAACL Student Research Workshop*, 88–93. https://doi.org/10.18653/v1/N16-2013

wikipedia. (2023). Hate speech laws by country. In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Hate_speech_laws_by_country&oldid=1146830275