Distributed and Parallel Databases (2023) 41:37–52
https://doi.org/10.1007/s10619-021-07331-4

Sentimental analysis from imbalanced code-mixed data using machine learning approaches

R. Srinivasan[1] · C. N. Subalalitha[1]

Accepted: 3 March 2021 / Published online: 20 March 2021

Abstract
Knowledge discovery from various perspectives has become a crucial asset in almost all fields. Sentimental analysis is a classification task used to classify the sentence based on the meaning of their context. This paper addresses class imbalance problem which is one of the important issues in sentimental analysis. Not much works focused on sentimental analysis with imbalanced class label distribution. The paper also focusses on another aspect of the problem which involves a concept called "Code Mixing". Code mixed data consists of text alternating between two or more languages. Class imbalance distribution is a commonly noted phenomenon in a code-mixed data. The existing works have focused more on analyzing the sentiments in a monolingual data but not in a code-mixed data. This paper addresses all these issues and comes up with a solution to analyze sentiments for a class imbalanced code-mixed data using sampling technique combined with levenshtein distance metrics. Furthermore, this paper compares the performances of various machine learning approaches namely, Random Forest Classifier, Logistic Regression, XGBoost classifier, Support Vector Machine and Naïve Bayes Classifier using F1- Score.

Keywords Code-mixed data · Sentimental analysis · Imbalanced data · Machine learning · Sampling

✉ R. Srinivasan
srinirvs89@gmail.com

C. N. Subalalitha
subalalitha@gmail.com

[1] Department of Computer Science and Engineering, SRM Institute of Science and Technology, Kattankulathur 603 203, India

1 Introduction

Due to the rapid development of social media, users are allowed to share, discuss or communicate their information very easily [1]. The information spans many interests right from politics, reviews on different products, academics, and much more. Due to the lockdown of COVID 19, the number of online users on social media have increased extensively. Social media sites also provide free open source and user-friendly approaches so that users can write their information in own native language or code-mixed data [2]. Code-mixed is a way of writing scripts using different languages or at least using more than one language or expressing the views in the social media.

Example 1

Code-mixed data: Trailer late ah parthavanga like podunga.

Translated data: Those who watched the trailer late, please like it.

In Example 1, “Trailer”, “like” are English words are mixed with tamil words to represent the sentence. It was observed that mainly in India, people prefer code mixed language while communicating in the social media [3]. This due to the fact that India is a multilingual country where people speak in different Indian language while their educational medium is through English. This is the main reason behind why Indians widely mix English with native language to express their views on social media. Also, the same scenario is observed in many other multilingual countries.

The application areas of the code-mixed data are Machine Translation (MT), Mixed Script Information Retrieval (MSIR), Language Identification, Sentimental analysis etc. Sentimental analysis (often known as Opinion Mining) is used to identify the emotions present in the given text. Sentimental analysis of a code-mixed social data is one of the challenging tasks of Machine Learning (ML) application. Preprocessing techniques that are normally used in sentimental analysis of a monolingual text includes, stemming, Parts of Speech Tagging (POS), Morphological analysis are insufficient to analyze sentiments in a code-mixed data. The reason behind this is that code-mixed data does not have a well-defined grammatical sentence and also consists of more unseen lexicons. To sum up, the major challenges of extracting sentiments in a code-mixed social text are:

(1) No word order: In code-mixed data, word order is completely lost. User can construct their own structure to form a sentence. For example, “Thalaivaru vera vera vera level pannitaru” which literally means, “The leader has performed extremely well”. In above code-mixed example, 1 English word “level” is present, whereas, the rest of the 5 words are Tamil words transliterated in English. It can be seen that the word “vera” is repeated twice to express the superlative performance of the leader. However, the meaning remains the same even when the word, “vera” is used once. These induces varied sentence structures that increases the complexity of analyzing sentiments in a code-mixed sentence. Tamil language sentences are already partially free word ordered, which at times

become intractable to capture the context. These code-mixed sentences worsen the scenario even more.

(2) Spelling variations: As the words used in code mixed sentences are user specific, there is no proper spelling rules. It creates a major problem to normalize those words while analyzing the sentiments. For example, the word "super" can be written as "superu", "sooper" and "suuuuupar" as the word "super" is used commonly by the Tamilians. While analyzing such words which are essential to capture the sentiments, it becomes difficult to normalize.
(3) Creative Spellings: Apart from the native language, users also create their own spelling for English as well while writing their opinions on social media. For example, "Youtube" is written as "Utube". "Great" as "gr8". This also induces complexity while performing a sentiment analysis.
(4) Abbreviations: In social media, people use abbreviations to represent a phrase. For example, FDFS represents "first day first show". This should be tackled while performing sentiment analysis.
(5) No Capitalization: Capitalization is also not followed by the user on social media which makes the process of identifying the sentence beginnings.

The above said challenges pertains to sentimental analysis of a code-mixed data. This paper also addresses yet another problem faced called, "imbalanced class label distribution" that exist in sentimental analysis even while doing for monolingual texts using Machine Learning approaches [4]. Imbalancing refers to the availability of more data tagged for one class compared to the other classes. Popular imbalanced datasets are credit card fraudulent, software defect prediction [5], and airline data [6]. These imbalances in class labels affects the accuracy of the classification task. The removal of the minority class labels also would affect the overall accuracy of the classification task. The proposed work uses a sampling technique to solve the class imbalance problem in a code-mixed data.

The main contributions of our work are as follows:

(1) To classify the code-mixed data and non-code-mixed data from the given corpus using an enhanced spell-checking algorithm.
(2) To create a lexicon dictionary for this code-mixed corpus. With the help of dictionary, the words with spelling variations in the corpus are normalized with the help of Levenshtein distance metric.
(3) To extract sentiments using various machine learning techniques and apply the sampling methods to solve the class imbalance problem.

The rest of the paper is organized as follows: Sect. 2, highlights the related work and Sect. 3, describe about the preprocessing and feature extraction techniques of the proposed methodology, Sect. 4 describes the result analysis of the proposed work, Sect. 5 describes the conclusion and future enhancement.

2 Literature survey

The literature survey has been done in three dimensions namely on the works done on sentimental analysis for monolingual data, sentimental analysis done on code mixed data and sentimental analysis done on data that has imbalanced class label distribution.

2.1 Sentimental analysis

Nasukawa et al. was first to coin the term sentiment analysis in the year 2003 [7] but linguistics people had already done a little research on the sentiment before the year 2000 [8]. Sentimental analysis was once a highly challenging task in the field of NLP in the beginning of twenty-first century. Later, Sentimental analysis started reaching many other fields other than computer science. For instance, in 2007, sentimental analysis was used in management studies by Archak et al., to derive the price of the product [9]. Later sentimental analysis was applied for many tasks in management studies to automatically obtain the opinion of products, price prediction and feedback of the products [10–13].

Liu et al. has identified three levels namely, document level, sentence level and entity level at which the sentimental analysis could be done. Later, different techniques have been investigated and applied to all these levels. The techniques can be broadly classified into three categories, namely, dictionary-based approaches, machine learning techniques and hybrid approaches. Dictionary based approaches mainly focusses on predetermined lexicons. Dictionary based approaches can work well on the monolingual data [14] as the standard lexicons are built and stored in dictionary. Dictionary based approaches does not need a training data and it is extremely hard for cross-domain or multilingual data [15].

Dictionary based approaches is entity specific, it is not suitable for all the domains. Machine learning methods, supervised approach, unsupervised and the semi-supervised approach are well distributed to the sentimental analysis for the monolingual data. The main disadvantage of the machine learning approach needs a large training data related to specific domain. Pollyanna Goncalves et al. compared an eight different types of hybrid approach for the sentimental analysis [16]. The above-mentioned techniques are well suited to analyses the sentiment for monolingual data. But the problem addressed in the paper is to analyzes the sentiments on the code-mixed data.

2.2 Sentimental analysis for code-mixed data

Code-mixing is a current trend in the field of sentimental analysis and transliteration. The techniques for extracting the proper sentiment in the code-mixed data is difficult [17]. Vijay et al. had done a research on the Hindi-English code-mixed social text data [18]. Code-mixed data needs a lot of preprocessing step compare to the monolingual data. The author proposed a lot of preprocessing techniques related to the mixed script and achieves the 58.2% accuracy using Support Vector Machine (SVM) classifier.

Shalini et al. proposed a distributed representation for extracting sentiments on different code-mixed text such as Kannada-English, Hindi-English, and Bengali-English [19]. The author has applied a variety of techniques to Sentimental Analysis on Indian Languages (SAIL) such as SVM, FastText, Bi-directional Long Short-Term Memory (Bi-LSTM) and Convolutional Neural Network (CNN). CNN are applied with different filter size and achieves an accuracy of 71.5% Kannada-English dataset, whereas Bi-LSTM produces a good accuracy of 60.2% for Hindi-English dataset and 72.2% for Bengali-English dataset.

Choudhary et al. proposed a novel approach and outperforms a best result compared to state of the approaches in Sentimental Analysis of Code-Mixed Text (SACMT) by 7.6% accuracy and 10.1% in F-Score [20]. Mishra et al. proposed a different Machine Learning and Deep Learning approaches to process the sentimental analysis on the Indian Languages [21]. The author produced an output of 69% F1-Score in Bengali-English dataset and 58% of F1-Score in Hindi-English dataset. Adhering to the aforementioned methods, we introduced a technique called Levenshtein distance to preprocess the code-mixed social text data. Another problem addressed in this paper is class imbalanced distribution. To overcome the class imbalance problem in code-mixed social text, we review the class imbalance distribution approaches in the following section.

2.3 Sentimental analysis for imbalanced class label distribution

Class imbalance is a major problem in the classification task which leads to minority samples being wrongly classified [22]. Many machine learning approaches have been used to solve the class imbalance problem in the past-decade. Haixiang et al. addressed one of the state-of-the-art techniques to solve the imbalance problem using two predominant methods [23]. The first method made changes in the preprocessing technique and applied cost sensitive learning methods to solve class imbalance problem. The second method made a minor change in the existing machine learning algorithms or generating hybrid model to solve the class imbalance issue. Li et al. proposed a minor change in the oversampling technique to solve the imbalance problem in the sentiment analysis [24]. The novel oversampling techniques produced improved F1-score of 81.5% compared to state-of-the-art approaches. Liu et al. introduced a new smoothing technique named as Random Over Sampling Expected Smoothing (ROSE) to handle imbalanced prior probability value [25]. The author proved that ROSE technique is more powerful compared to the other smoothing approaches. Lu et al. discovered a complexity measure called Individual Bayes Imbalance Impact Index (IBI3) to check whether the dataset is worth to apply sampling methods or not [26]. After completing the preprocessing, dataset is applied with vector space resampling methods to find out the best F1-score value. Especially, oversampling method is considered to solve the class imbalance problem for the code-mixed social text data. A variety of hybrid machine learning techniques have been proposed to solve the class imbalanced issue such as ROSE, oversampling methods, vector space resampling techniques etc. To the best of our knowledge, levenshtein distance has not been used as preprocessing technique so far to solve the spelling variations in the Tamil-English code-mixed data. In addition, resampling techniques have never been attempted for code-mixed data to solve the class imbalance problem. The proposed work attempts to use Levenshtein distance and resampling techniques to explore the Tamil-English code-mixed data and further analyze sentiments present in it using various machine-learning approaches.

3 Proposed work

3.1 Dataset description

Chakravarthi et al. have created a dataset (Tamil-English) to extract sentiments from code-mixed social text data. The authors have created a bilingual dataset for Indian languages namely, Tamil-English and Malayalam-English [27, 28]. The dataset is scrapped from the Youtube comments by using tool called YouTube Comment Scraper tool. The proposed work mainly focusses on Tamil-English dataset and extracts sentiment from it. Table 1 describes the dataset used in the proposed approach. The dataset consists of 15,744 sentences and it is divided into three categories. 11,335 sentences are considered as training data, 1260 sentences are considered as validation set and 3149 sentences are used as test data. The sentiments

Table 1 Dataset description

Categories	Training data	Validation data	Test data
Positive	7627	856	2076
Negative	1448	165	424
Mixed Feelings	1283	141	377
Not_Tamil	368	29	100
Unknown State	609	68	173

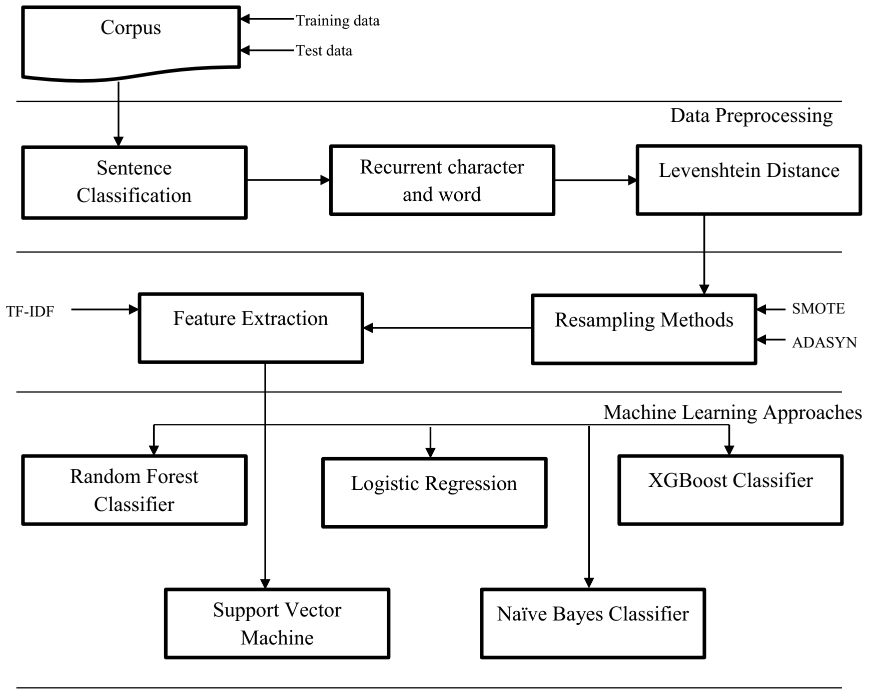

Fig. 1 Architecture of proposed methodology

present in the dataset are divided into five categories namely, Positive, Negative, Mixed Feelings, Not_Tamil and Unknown State. The proposed architecture is shown in Fig. 1.

3.2 Data preprocessing

Data preprocessing is an important step that helps to enhance and extract the meaningful insights from the data. Data preprocessing (also called as Data Cleaning) technique helps to remove the errors and inconsistencies present in the data [29]. Sometimes inconsistencies of the data that creates the illogical or missing important information that affects the accuracy of the data. The following actions are to be taken in the preprocessing stage:

(1) Sentence are divided into tokens and special characters are removed
(2) The characters are converted either into uppercase or lower case
(3) Usually stop word removal forms one of the most important steps in preprocessing. Since the stop word removal was found to be a difficult procedure, we have replaced that by removing the words that has less than 2-character length.

3.2.1 Sentence classification

The dataset is first divided into Tamil-English script and English (monolingual) script. The sentences which are not in Tamil-English script will be eliminated in the initial step.

```
Algorithm 1: Sentence Classification
        Input: Sentence (S)
        Notations: $\alpha$ refers to English script (monolingual), $\beta$ refers to Tamil-English script (code-
        mixed data)
        Output: Sentence Classification ($\alpha$ or $\beta$)
Step 1: Read S
Step 2: Calculate the length of S
Step 3: Split S into $w_1$, $w_2$,,……,$w_n$
Step 4: for each word $w_i$ =1 to n
                if langdetect($w_i$) is equal to english
                   Increase the count value
                else
                   break
        end for
Step 5: if count value is equal to number of words extract from S
                Label $\alpha$
        else
                Label $\beta$
Step 6: end
```

The sentence classification algorithm is used to classify the code-mixed (β) and non-code-mixed (α) sentences. First, Sentences (S) are given as input to the algorithm and is divided into number of words w_1, w_2,……, wn. For each word in the sentence, it is checked if it is English or not using language detector. The sentence containing non-English words are counted as code mixed sentences. After applying the sentence classification algorithm on the dataset, 541 sentences are identified as English script (α) and are removed. This is one of the reasons for the increased F1-Score.

3.2.2 Recurrent character

In social media, people tend to use many variants for a single word to express their varied emotion. For example, people type "wowww" for the word, "Wow" to express their increased degree of emotion. These word variations are to be normalized prior to feature extraction which is essential in capturing the sentiments more precisely. Recurrent character has to be by comparing it with the base word form.

3.2.3 Recurrent word

Similar to removing the recurrent characters, recurrent words are also removed. For example, "Thala vera vera vera level panitaru" which means the person has performed extremely well. The word, "vera" is repeated to express more degree of applause. Since this paper is targeting to detect positive emotion and not aiming at classifying the varying degrees in the positive emotion, these repeated words are removed as part of normalization procedure.

3.2.4 Levenshtein distance

The term Levenshtein distance may also be referred to edit distance was coined in the year 1965 and is used to find the distance between the words [30]. Levenshtein distance finds out the minimum number of single characters edits that is required to normalize it with the base word [30]. Levenshtein distance has four operations namely, identity, insertion, substitution and deletion. The distance can be calculated by using Eq. 1.

$$d(0,0) = 0, d(i,j) = \min \begin{cases} d(i-1,j)+1 & \text{deletion} \\ d(i,j-1)+1 & \text{insertion} \\ d(i-1,j-i)+1 & \text{substitution} \end{cases} \tag{1}$$

In Eq. 1, d (i, j) is the distance between the i characters of string S_1 and j characters of string S_2. For example: Let S_1 = Talaivar and S_2 = Thalaivar be two strings. Here S_1 is the source string and S_2 is target string (base word). S_1[1…. m] and S_2[1…. n] where m and n are length of the strings, S_1 and S_2. When each character in S_1 is compared with the targeted string S_2, the character 'h' is inserted in S_1 to match S_2. The minimum edit distance required to normalize a from source string to the target string is 1. In this paper, we have set the maximum edit distance as two because it was observed, when the edit distance is greater than 2, the word transforms into a different word not matching with the meaning of the base word. These normalization procedures have aided in reducing the computational complexity and also resulted in increased F-score.

3.3 Resampling techniques

Resampling techniques are very important to solve the class imbalance problem. Resampling can be done in different ways namely, Randomized Exact Test (also called as Permutation test), Cross-validation, Jackknife and Bootstrapping techniques [32]. Furthermore, hybrid techniques can be generated with the help of bootstrapping techniques. Oversampling and under sampling techniques are very helpful to solve the imbalanced class label distribution for multi-class classification [33].

This paper also addresses the problem of imbalanced class label distribution in multi-class classification task for code-mixed data. The problem of using under sampling techniques, number of sentences in the majority class can be equivalent to the number of sentences in the minority class. This training data consists of 368 sentences named as "Unknown state" considered to be minority class. After applying the under-sampling technique, 1840 sentences are considered as training data from the 11,335 sentences. The ratio of selected sentences for training from overall sentences is reduced to 1:6 approximately. After applying the under-sampling techniques, the size of the dataset gets reduced. Oversampling technique is just opposite to the under-sampling technique. Oversampling technique helps to increase the size of the training dataset by duplicating the minority class data. Oversampling could be best option to balance the data that suffers from imbalanced class label distribution [33]. This paper also addresses two different techniques in oversampling techniques namely, Synthetic Minority Over-Sampling (SMOTE) and Adaptive Synthetic (ADASYN).

3.4 Feature extraction

Term Frequency and Inverse Document Frequency (Tf-Idf) is a method to convert the text into a vector form. The Tf-Idf is proposed by jones in 1972 and it is useful for information retrieval and text classification process [31]. Tf-Idf is a statistical measure that helps to assign a weight for each word in the corpus. The term frequency (TF) is defined as number of times a word occur in a document whereas, Inverse Document Frequency (IDF) increases the weight for rarely occurring words but decreases the weight for frequently occurring unimportant words. Feature Extraction for sentimental analysis can be done using Bag of Words (BoW), Word2vec, Global Vector Representation (GloVe). BOW gives higher weightage to the frequently occurring words in the document. Word2Vec, GloVe are pretrained model to convert the text into vector. Since it is a code-mixed data, no pre-trained word embeddings are available. Also, code mixed data usually contains words that has less frequency but needs to be given weightage and hence TF-IDF has been chosen for extracting the features.

3.5 Machine learning approaches

After completing the preprocessing techniques, the next task is to apply the ML algorithms for the classifying the sentiments of the code-mixed data. There are atleast two problems that can occur while applying ML algorithms for any data. The first problem is to find relevant feature extraction technique and the second is to find a suitable classification algorithm. It was observed that when most suitable feature extraction techniques are chosen, any classification algorithm was able to classify the sentiments from the code-mixed data. The proposed work has analyzed by using

Table 2 F1-score value using SMOTE technique

Model	Positive	Negative	Mixed Feelings	Unknown state	Not_Tamil
Random Forest Classifier	0.81	0.19	0.03	0	0.04
Logistic regression	0.81	0.26	0.03	0	0.08
XGBoost classifier	0.8	0.19	0.02	0	0.08
SVM	0.8	0.12	0.01	0	0.02
Naïve Bayes	0.8	0	0	0	0.06

Table 3 F1-score value using ADASYN technique

Model	Positive	Negative	Mixed Feelings	Unknown state	Not_Tamil
Random Forest Classifier	0.81	0.21	0.03	0.02	0.05
Logistic regression	0.81	0.26	0.03	0	0.08
XGBoost classifier	0.8	0.19	0.02	0.02	0.09
SVM	0.8	0.11	0.01	0	0.02
Naïve Bayes	0.79	0	0	0	0.06

various classification algorithms namely, Probabilistic models (Naïve Bayes classifier (NB)), linear classifier (Support Vector Machine classifier (SVM)), Decision Based (Random Forest Classifier and XGBoost classifier), and Statistical Model (Logistic Regression).

4 Evaluation and result analysis

This section describes about the evaluation metrics and results obtained for the dataset. Precision, Recall, F1-Score and Accuracy are commonly used evaluation measures for any classification problem but while using machine learning algorithms, for an imbalanced dataset appropriate evaluate metric has to be chosen. Various evaluation metrics were analyzed such as, Accuracy, Macro average F1 score and Micro average F1 score. Since accuracy and micro average F1-score are not preferred metrics for evaluating the class imbalance data, the macro average F1-score is chosen along with recall and precision as evaluation metrics. The formula for evaluation metrics is shown in equation:

$$\text{Recall} = \text{ ratio of correctly predicted sentence to the overall observation in the actual class.}$$

$$\text{precision } = \text{ratio of correctly predicted sentences to the total predicted sentences.}$$

F1 Score is the weighted mean of precision and recall. In order to emphasize the importance of identifying the code-mixed data while analyzing the sentiments, we identified sentiments with and without code mixed data. Section 4.1 discusses about

Table 4 F1-score value using SMOTE technique

Model	Positive	Negative	Mixed Feelings	Unknown state	Not_Tamil
Random Forest Classifier	0.8	0.24	0.04	0.18	0.6
Logistic regression	0.68	0.34	0.19	0.26	0.56
XGBoost classifier	0.79	0.27	0.1	0.26	0.51
SVM	0.8	0.17	0.02	0.19	0.5
Naïve Bayes	0.8	0	0	0	0.05

Table 5 F1-score value using ADASYN technique

Model	Positive	Negative	Mixed Feelings	Unknown state	Not_Tamil
Random Forest Classifier	0.81	0.18	0.04	0.08	0.59
Logistic regression	0.81	0.28	0.03	0.12	0.53
XGBoost classifier	0.8	0.19	0.03	0.12	0.51
SVM	0.8	0.12	0.01	0.1	0.53
Naïve Bayes	0.78	0	0	0	0.05

Table 6 F1-score value using SMOTE technique

Model	Positive	Negative	Mixed Feelings	Unknown state	Not_Tamil
Random Forest Classifier	0.8	0.24	0.04	0.18	0.6
Logistic regression	0.68	0.37	0.19	0.36	0.56
XGBoost classifier	0.8	0.24	0.1	0.22	0.51
SVM	0.8	0.23	0.04	0.18	0.5
Naïve Bayes	0.8	0	0	0	0.05

the results that are obtained without classifying the code mixed and non-code mixed data. It was observed that when the sentiment analysis was done after removing the non-code mixed data followed by classifying codemixed and non-code mixed data, it resulted in a better F1-score. The next section describes about the results obtained without classifying code mixed and non-code mixed data.

Table 7 F1-Score value using ADASYN technique

Model	Positive	Negative	Mixed Feelings	Unknown state	Not_Tamil
Random Forest Classifier	0.81	0.16	0.02	0.06	0.59
Logistic regression	0.81	0.29	0.02	0.12	0.53
XGBoost classifier	0.8	0.19	0.03	0.21	0.51
SVM	0.8	0.14	0.01	0.2	0.53
Naïve Bayes	0.78	0	0	0	0.05

4.1 Sentiment analysis without separating code mixed and non-code mixed data

This dataset consists of 11,335 training sentences and 3149 test sentences. Two resampling techniques namely SMOTE and ADASYN techniques were used and their performance were compared. Table 2 shows the result obtained using SMOTE sampling technique and Table 2 shows the results obtained by using ADSYN technique. It was observed that these resampling techniques improved the F1-score by 50%.

It can be observed from Tables 2 and 3 that, Logistic regression performs compared to other techniques, whereas naïve bayes fails to predict other classes except positive. Also, it can be observed that even after applying sampling techniques, the classification algorithms are more biased towards the positive class. This class imbalance is due to the non-separation of code mixed and non-code mixed data. The next section describes the influence of identifying the sentiments after classifying the code mixed and non-code-mixed data.

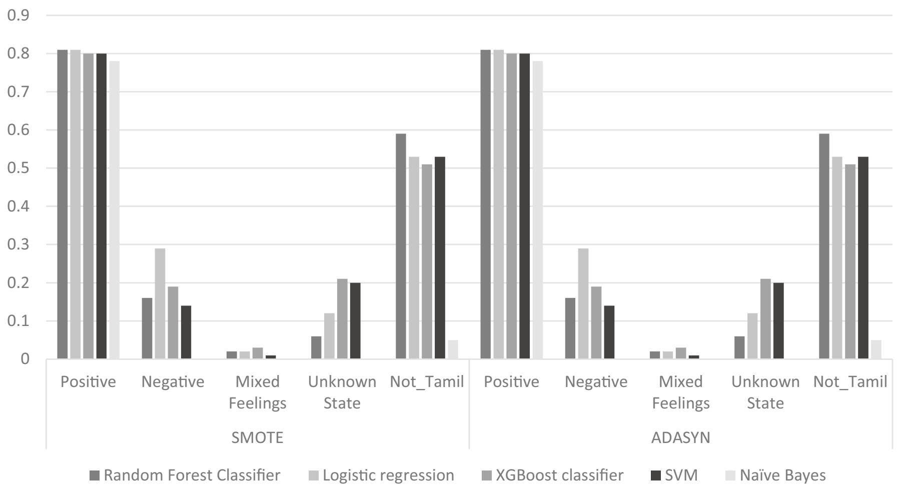

Fig. 2 Results of all relations using ML approaches

4.2 Results obtained after separating the classifying code mixed and non-code mixed data

Out of 15,744 sentences, 541 sentences are non-code-mixed sentences which are identified and removed before preprocessing the data. This resulted in a better distribution of sentiments across all categories overcoming the hurdles observed in Sect. 4.1. This can also be very well observed from Table 4 and 5 where an increased F1-score for each class was obtained and also resulted in an improved averaged F1-score computed for all classes. Averaged F1-score value increased by 2% after removing the non-code-mixed data.

The code-mixed sentences also contained other Indian languages such as, Hindi, Telegu in this dataset. After removing the non-code-mixed data, sampling techniques was observed to perform better for even "Not_Tamil" category compared to the previous Tables 1 and 2 due to the usage o SMOTE and ADASYN resampling techniques. But it was observed that the class imbalance still existed despite all these efforts. So far, all the experiments were done excluding the Levenshtein distance preprocessing step. It was observed that Levenshtein distance had a major role in identifying the code-mixed data. In order to highlight the usage of Levenshtein distance when compared to the other preprocessing techniques, a separate analysis was done using Levenshtein distance which are shown in Tables 6 and 7.

4.3 Result after using Levenshtein distance

As discussed in Sect. 3.2.4 the Levenshtein distance helps to minimize the spelling error using minimum edit distance. In code-mixed data, spelling variations is a major problem but can be alleviated using Levenshtein distance. Tables 6 and 7 discusses the result after applying the Levenshtein distance.

The overall performance is almost similar to the previous results shown in Table 6. Since code-mixed data, also contained Hindi, Telugu and Malayalam language words many code-mixed words could not be identified. For Example, "PINK FILM COPY HAI", Here "Hai" is confused with the English "Hai". Since the corpus was built by extracting youtube comments, they are many inconsistencies such as spelling mistakes, many language code-mixing words etc.

Figure 2 depicts the performance of each relation using various ML approaches. The Levenshtein distance has overcome the spelling mistakes present in the data and many strong features for each class has alleviated the class imbalance problem to an extent. Still the class imbalance problem was not completely tackled as features contributing to one class sometimes pull down the other class. For instance, "semaya irukumnu ethirpaarthen (I thought it would be so good)". This either pulls the sentence to positive or negative due to the ambiguous multi word context. Furthermore, it was observed that, a single ML algorithm performed better for a particular class and not performing well while looking at all the classes. This was the main reason behind attempting many ML classifiers to check their multi class handling capability. Out of all ML techniques, Logistic regression has shown better performance compared to that of rest of the classifiers.

5 Conclusion and future enhancement

The proposed work has attempted to classify sentiments from a code-mixed data that contains, majorly Tamil and also, Hindi, Telugu, Malayalam words. The proposed work has brought a lot of observations while processing code mixed data at various levels namely, preprocessing, feature extraction and classification. This paper has proposed levenshtein distance as the preprocessing technique for Tamil-English code-mixed data. This has improved the results as it worked well with identifying spelling variations that persisted in the code-mixed data written on social media. Then the proposed approach experiments revealed that the class imbalance problem can be alleviated by removing the non-code mixed data. Also, the influence of using resampling techniques such as SMOTE and ADASYN were also discussed. The combination of levenshtein distance with sampling techniques helped to increase the F1-Score but there is still a gap observed in the class imbalance problem. The future work can target at improving the F1-score by finding a strong feature extraction techniques or hybrid approaches that can helps to solve the class imbalanced problem existing in the code-mixed social text data.

References

1. Wang, Y., Rao, Y., Zhan, X., Chen, H., Luo, M., Yin, J.: Sentiment and emotion classification over noisy labels. Knowl. Based Syst. (2016). https://doi.org/10.1016/j.knosys.2016.08.012
2. Sreelakshmi, K., Premjith, B., Soman, K.P.: Detection of hate speech text in Hindi-English code-mixed data. ProcediaComput. Sci. **171**, 737–744 (2020). https://doi.org/10.1016/j.procs.2020.04.080
3. Yadav, S., Chakraborty, T.: Unsupervised sentiment analysis for code-mixed data. Comput. Language. arXiv:2001.11384 (2020).
4. Yang, Q., Wu, X.: 10 Challenging problems in data mining research. Int. J. Inf. Technol. Decis. Mak. (IJITDM) **5**, 597–604 (2006). https://doi.org/10.1142/S0219622006002258
5. Jiang, H.: Sentiment analysis on imbalanced airline data. In: Proceedings of Jiang2016AnalysisOI, (2003).
6. Wang, S., Yao, X.: Using class imbalance learning for software defect prediction. IEEE Trans. Reliab. **62**, 434–443 (2013). https://doi.org/10.1109/TR.2013.2259203
7. Nasukawa, T., Yi, J.: Sentiment analysis: capturing favorability using natural language processing. In: Proceedings of the 2nd International Conference on Knowledge Capture, pp. 70–77 (2003). https://doi.org/10.1145/945645.945658
8. Liu, B.: Sentiment analysis and opinion mining. Synth. Lect. Hum. Lang. Technol. **5**(1), 1–167 (2012). https://doi.org/10.2200/S00416ED1V01Y201204HLT016
9. Archak, N., Ghose, A., Ipeirotis, P.: Deriving the pricing power of product features by mining consumer reviews. Working Papers, vol 57. NET Institute (2007). https://doi.org/10.1287/mnsc.1110.1370
10. Chen, J.I.Z., Lai, K.-L.: Machine learning based energy management at Internet of Things network nodes. J. Trends Comput. Sci. Smart Technol. **3**(2020), 127–133 (2020)
11. Chen, Y., Xie, J.: Online Consumer Review: Word-of-mouth as a new element of marketing communication mix. Manage. Sci. **54**, 477–491 (2008). https://doi.org/10.1287/mnsc.1070.0810
12. Ghose, A., Ipeirotis, P., Sundararajan, A.: Opinion mining using econometrics: a case study on reputation systems. In: Proceedings of the Association for Computational Linguistics (ACL) (2007)
13. Park, D.-H., Lee, J., Han, I.: The effect of on-line consumer reviews on consumer purchasing intention: the moderating role of involvement. Int. J. Electron. Commer. **11**, 125–148 (2007). https://doi.org/10.2753/JEC1086-4415110405

14. Sasidhar, T., Premjith, B., Soman, K.P.: Emotion detection in Hinglish (Hindi+English) code-mixed social media text. ProcediaComput. Sci. **171**, 1346–1352 (2020). https://doi.org/10.1016/j.procs.2020.04.144
15. Nguyen, T., Nguyen, L., Cao, T. (2017). Sentiment analysis on medical text using combination of machine learning and SO-CAL scoring. In: Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES), pp. 49–54. https://doi.org/10.1109/IESYS.2017.8233560.
16. Gonçalves, P., Araújo, M., Benevenuto, F., Cha, M.: Comparing and combining sentiment analysis methods. In: COSN 2013—Proceedings of the 2013 Conference on Online Social Networks, pp. 27–38 (2013). https://doi.org/10.1145/2512938.2512951
17. Shekhar, S., Sharma, D., Agarwal, D., Pathak, Y.: Artificial immune systems-based classification model for code-mixed social media data. IRBM (2020). https://doi.org/10.1016/j.irbm.2020.07.004
18. Raj, J.S.: Machine learning based resourceful clustering with load optimization for wireless sensor networks. J. Ubiquit. Comput. Commun. Technol. (UCCT) **2**(1), 29–38 (2020)
19. Shalini, K., Hb, B.G., Kumar, M., Soman, K.P.: Sentiment analysis for code-mixed Indian social media text with distributed representation. In: 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1126–1131 (2018). https://doi.org/10.1109/ICACCI.2018.8554835.
20. Choudhary, N., Singh, R., Bindlish, I., Shrivastava, M.: Sentiment analysis of code-mixed languages leveraging resource rich languages. In: 19th International Conference on Computational Linguistics and Intelligent Text Processing, 2018, Hanoi, Vietnam (2018)
21. Mishra, P., Danda, P., Dhakras, P.: Code-mixed sentiment analysis using machine learning and neural network approaches (2018)
22. López, V., Fernández, A., García, S., Palade, V., Herrera, F.: An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics. Inf. Sci. **250**, 113–141 (2013). https://doi.org/10.1016/j.ins.2013.07.007
23. Haixiang, G., Li, Y., Shang, J., Mingyun, G., Yuanyue, H., Gong, B.: Learning from class-imbalanced data: review of methods and applications. Expert Syst. Appl. (2016). https://doi.org/10.1016/j.eswa.2016.12.035
24. Li, Y., Guo, H., Zhang, Q., Gu, M., Yang, J.: Imbalanced text sentiment classification using universal and domain-specific knowledge. Knowl. Based Syst. (2018). https://doi.org/10.1016/j.knosys.2018.06.019
25. Duraipandian, M.: Performance evaluation of routing algorithm for Manet based on the machine learning techniques. J. Trends Comput. Sci. Smart Technol. (TCSST) **1**(01), 25–38 (2019)
26. Lu, Y., Cheung, Y., Tang, Y.: Bayes imbalance impact index: a measure of class imbalanced dataset for classification problem. IEEE Trans. Neural Netw. Learn. Syst. **31**(9), 3525–3539 (2019)
27. Chakravarthi, B., Muralidaran, V., Priyadharshini, R., McCrae, J.: Corpus creation for sentiment analysis in code-mixed Tamil-English text. Comput. Language. arXiv:2006.00206 (2020).
28. Chakravarthi, B.R., Jose, N., Suryawanshi, S., Sherly, E., McCrae, J.P.: A sentiment analysis dataset for code-mixed Malayalam-English. In: Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020), pp. 177–184 (2020)
29. Asif, M., Ishtiaq, A., Ahmad, H., Aljuaid, H., Shah, J.: Sentiment analysis of extremism in social media from textual information. Telematics Inform. **48**, 101345 (2020). https://doi.org/10.1016/j.tele.2020.101345
30. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals. Sov. Phys. Doklady **10**, 707 (1966)
31. Jones, K.: A statistical interpretation of term specificity in retrieval. J. Document. **60**, 493–502 (2004). https://doi.org/10.1108/00220410410560573
32. Chong, H., Yu, C.H.: Resampling methods: concepts, applications, and justification. Pract. Assess. Res. Eval. (2003). https://doi.org/10.7275/9cms-my97
33. Oliveira, D., Porpino, T., Cavalcanti, G., Ing Ren, T.: A bootstrap-based iterative selection for ensemble generation. In: 2015 International Joint Conference on Neural Networks (IJCNN) (2015). https://doi.org/10.1109/IJCNN.2015.7280695.