VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi - 590 018, Karnataka

Sentiment Analysis for Cross-Lingual Kannada English language pair

A Report submitted in partial fulfillment of the requirements for the Course

Project work on Machine Learning
(Course Code: 22AM5PWPML)

In the Department of

Machine Learning
(UG Program: B.E. in Artificial Intelligence and Machine Learning)

By

Kapilesh U (USN: **1BM20AI029)**
Manchikanti Varunsai (USN: **1BM20AI036)**
Sai Krishna Manoj Alapati (USN: **1BM20AI047)**
Varun S (USN: **1BM20AI056)**
Semester & Section: 5 A

Under the Guidance of
Dr. Sowmya Lakshmi B S
Assistant Professor
Dept. of MEL, BMSCE, Bengaluru – 19

DEPARTMENT OF MACHINE LEARNING

B.M.S COLLEGE OF ENGINEERING

(An Autonomous Institute, Affiliated to VTU)

P.O. Box No. 1908, Bull Temple Road, Bengaluru - 560 019

March - 2023

B.M.S COLLEGE OF ENGINEERING

(An Autonomous Institute, Affiliated to VTU)

P.O. Box No. 1908, Bull Temple Road, Bengaluru - 560 019

DEPARTMENT OF MACHINE LEARNING

CERTIFICATE

This is to certify that ***Mr. Kapilesh U, Mr. Manchikanti Varunsai, Mr. Sai Krishna Manoj Alapati*** and ***Mr. Varun S*** bearing USNs: ***1BM20AI029 1BM20AI036 1BM20AI047 1BM20AI056*** respectively have satisfactorily presented the Course – ***Project work on Machine Learning*** (Course code: **22AM5PWPML)** with the title ***"Sentiment Analysis for Cross-Lingual Kannada English language pair"*** in partial fulfillment of academic curriculum requirements of the 5th semester UG Program – B. E. in Artificial Intelligence and Machine Learning in the Department of Machine Learning, BMSCE, an Autonomous Institute, affiliated to Visvesvaraya Technological University, Belagavi during March 2023. It is also stated that the base work & materials considered for completion of the said course is used only for academic purpose and not used in its original form anywhere for award of any degree.

Student Signature

Signature of the Supervisor	**Signature of the Head**
Dr. Sowmya Lakshmi B S	**Dr. Gowrishankar**
Assistant Professor, Dept. of MEL, BMSCE	Prof. & Head, Dept. of MEL, BMSCE

External Examination

Examiner Name and Signature

1.

2.

ACKNOWLEDGEMENT

We express our profound gratitude to **Hon’ble Management and Dr. S Muralidhara,** Principal, BMSCE, Bengaluru for providing the necessary facilities and an ambient environment to work.

We are grateful to **Dr. Gowrishankar,** Professor & Head, Department of Machine Learning, Bengaluru, for his valuable suggestions and advice throughout our work period.

We would like to express my deepest gratitude and sincere thanks to our guide **Dr. Sowmya Lakshmi B S,** Assistant Professor, Department of Machine Learning, Bengaluru**,** for her keen interest and encouragement in guiding us and bringing the work to reality.

We would like to thank all the **Staff**, Department of Machine Learning for their support and encouragement during the course of this project.

Last but not the least, we would like to thank our parents, family and friends, without whose help and encouragement this work would have been impossible.

ABSTRACT

This report presents the findings of a project focused on developing a sentiment analysis system for the cross-lingual Kannada-English language pair. The objective of the project was to build a model that could accurately classify the sentiment of text in both languages, using a dataset of movie reviews as the primary source of data. The project involved several stages, including data collection and preprocessing, feature engineering, model selection, and evaluation. Various machine learning algorithms, including Naive Bayes, Logistic Regression, Random Forest, and Support Vector Machines, were evaluated, and the best performing model was selected for deployment. The results of the study demonstrate that the sentiment analysis system developed in this project is effective in accurately classifying the sentiment of text in both Kannada and English languages. The report concludes with a discussion of the limitations of the project, along with recommendations for future work in this area.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 ABOUT THE DOMAIN:

Natural Language Processing (NLP) is a field of computer science and artificial intelligence (AI) that deals with the interaction between computers and human languages. It involves analyzing and processing natural language data, such as text and speech, to enable machines to understand and interpret human language.

Sentiment analysis, a subfield of NLP, is the process of determining the sentiment or emotion expressed in a piece of text. It involves using various techniques, such as machine learning and natural language processing, to extract and quantify the emotional content of a piece of text, whether it's positive, negative, or neutral. Sentiment analysis is widely used in applications such as social media monitoring, customer feedback analysis, and market research, among others.

1.2 OBJECTIVE:

The objective of this project is to perform sentiment analysis for a cross-lingual Kannada-English language pair. The first step in this process is to preprocess the text by removing emojis, chorus words, and correcting spelling mistakes to ensure that the data is clean and consistent. The next step is to perform sentiment analysis on the preprocessed text using various machine learning models.

The primary focus of this project is to translate the cross-lingual text into Kannada instead of English, which has been extensively studied in the field of sentiment analysis. By utilizing Kannada translations as input to the models, we aim to improve the accuracy of the sentiment analysis for Kannada language, which has received less attention in this area.

To evaluate the performance of different machine learning models, a comparative analysis will be conducted, and the most effective model will be selected for the task at hand. The overall goal of this project is to develop an accurate and efficient method for sentiment analysis of cross-lingual Kannada-English text, which could have potential applications in fields such as social media monitoring, customer feedback analysis, and market research.

1.3 SCOPE:

Our project covers the various techniques and approaches used for cross-lingual sentiment analysis, including machine learning, natural language processing, and other related technologies. It also addresses the challenges posed by cross-lingual analysis, such as language and cultural barriers, and the strategies employed to overcome them. We hope that our project will serve as a good baseline for future research in Sentiment Analysis for Dravidian code-mixed sentences.

1.4 MOTIVATION:

Sentiment Analysis has been performed to a great extent on the monolingual datasets with satisfying accuracy but in a country like India, which has multiple spoken languages, we often encounter ourselves with code-mixed sentences. These code-mixed sentences act as a barrier to the Monolingual NLP models as they cannot perform well on these code-mixed datasets. This lack of adequate research in dealing with code-mixed sentences along with the general lack of research for the Dravidian languages serves as our motivation for this project.

1.5 ORGANIZATION OF REPORT:

This organization provides a logical flow for the report, starting with an introduction to the domain and the purpose of the report, followed by a discussion of related work and open issues, and then moving on to the specifics of the project, including data collection and detailed design. The report then discusses the results and conclusions of the project, along with any potential further enhancements, and includes any relevant publications and references. Finally, there are two appendices for additional information.

CHAPTER 2

RELATED WORK

[1]The paper provides an overview of the DravidianCodeMix 2021 shared task on sentiment detection in Tamil, Malayalam, and Kannada. It discusses the dataset used, the participating systems, the evaluation metrics, and the results obtained. The paper highlights the need for further research on sentiment analysis for Dravidian languages.

[2]The paper "Findings of the Sentiment Analysis of Dravidian Languages in Code-Mixed Text" includes previous research on sentiment analysis in code-mixed texts, studies on sentiment analysis in individual Dravidian languages, and techniques used for feature extraction and classification in sentiment analysis.

[3]The authors of the paper reviewed previous works on sentiment analysis, multilingualism, and code-mixing in natural language processing. They discussed the challenges of analyzing sentiment in code-mixed languages and proposed a novel approach to sentiment analysis on multilingual code-mixed Kannada language using a combination of deep learning and traditional machine learning techniques.

[4]The paper "NITP-AI-NLP@Dravidian-CodeMix-FIRE2020: A Hybrid CNN and Bi-LSTM Network for Sentiment Analysis of Dravidian Code-Mixed Social Media Posts" includes previous research on sentiment analysis in code-mixed texts, studies on sentiment analysis in individual Dravidian languages, and techniques used for feature extraction and classification in sentiment analysis, particularly the use of convolutional neural networks and bidirectional LSTM networks.

[5]The related work in this paper on sentiment analysis of Twitter data covers various techniques and approaches, such as lexicon-based, machine learning, and deep learning methods. The review also highlights challenges, such as handling sarcasm, noise, and subjectivity in social media text, and discusses various evaluation metrics and datasets used in previous studies.

[6]The paper "Sentiment Analysis Using Convolutional Neural Networks" includes previous research on sentiment analysis using various machine learning techniques such as Support Vector Machines and Naive Bayes classifiers, as well as studies on convolutional neural networks (CNNs) in other natural language processing tasks. The paper also discusses the use of word embeddings and data augmentation techniques in CNN-based sentiment analysis.

[7]The paper discusses the use of deep learning for sentiment analysis and identifying offensive language in multilingual code-mixed data. The authors compare their approach with existing methods and achieve better accuracy, demonstrating the effectiveness of their approach.

[8]The paper "A Deep Learning Approach Combining CNN and Bi-LSTM with SVM Classifier for Arabic Sentiment Analysis" includes previous research on sentiment analysis in Arabic language, the use of convolutional neural networks (CNNs) and bidirectional LSTM networks in sentiment analysis, and the use of support vector machines (SVMs) as classifiers. The paper also discusses the use of pre-trained word embeddings and data augmentation techniques in the proposed approach.

CHAPTER 3

OPEN ISSUES AND PROBLEM STATEMENT

Sentiment analysis in Dravidian languages is a relatively under-studied area. Despite some work done in this area, only a few researchers have focused specifically on Kannada language. One of the major challenges in conducting sentiment analysis in Kannada is the lack of readily available datasets. As a result, researchers often resort to web scraping from various sources to collect data. However, even when a large amount of data is scraped, the process of annotating the data for sentiment analysis can be time-consuming and resource-intensive. Additionally, due to inconsistencies in the data and the lack of a standardized approach to annotation, even the best sentiment analysis models may not perform well. Therefore, more research and development is needed in this area to address these challenges and to enable more accurate sentiment analysis in Dravidian languages, particularly Kannada.

The problem addressed in this project is the lack of effective sentiment analysis models for the cross-lingual Kannada-English language pair. While sentiment analysis has been extensively studied for English text, there is a need to improve its accuracy and applicability for other languages, particularly Kannada. This project aims to develop an efficient and accurate method for sentiment analysis of cross-lingual Kannada-English text, which could have significant practical applications in areas such as social media monitoring and customer feedback analysis. The project will involve preprocessing the text, performing sentiment analysis using various machine learning models, translating the text into Kannada, and conducting a comparative analysis to determine the best model for the task at hand.

CHAPTER 4

DATA COLLECTION AND VALIDATION:

This project utilizes the dataset obtained from the "**Dravidian-CodeMix-FIRE 2021**" challenge, which contains labeled sentences from YouTube comments in the Kannada-English code-mixed language. However, the raw sentences from the dataset are not directly usable for sentiment analysis models as they are unclean and contain mixed languages. Therefore, the data preprocessing step becomes crucial in this project.

In this project, we focus on converting the sentences into monolingual Kannada language as opposed to monolingual English, which has been commonly used in previous research. By doing so, we aim to improve the accuracy and applicability of sentiment analysis for the Kannada language, which is currently under-researched in this area. As shown in Table 1,the data preprocessing involves various steps such as removing special characters, stop words, emojis and punctuation marks. The primary focus is on converting the sentences into monolingual Kannada while preserving the sentiment labels of the original sentences.

Levels of Processing	**Input sentence**	**Output Sentence**
Direct sentence from the FIRE dataset (Data preprocessing)	ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ತಿಳಿದುಕೊಳ್ಳಿ .... PLEASE do not tell wrong informatoin 😑 😑 😑	ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ತಿಳಿದುಕೊಳ್ಳಿ PLEASE do not tell wrong informatoin
First level of text processing (Transliteration)	ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ತಿಳಿದುಕೊಳ್ಳಿ PLEASE do not tell wrong informatoin	ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ತಿಳಿದುಕೊಳ್ಳಿ PLEASE do not tell wrong informatoin
Second level of text processing (Spell check + Translation)	ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ತಿಳಿದುಕೊಳ್ಳಿ PLEASE do not tell wrong informatoin	ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ತಿಳಿದುಕೊಳ್ಳಿ ದಯವಿಟ್ಟು ಮಾಡು ಅಲ್ಲ ಹೇಳು ತಪ್ಪು ಮಾಹಿತಿ
Final level of preprocessing (Getting a meaningful kannada sentence)	ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ತಿಳಿದುಕೊಳ್ಳಿ ದಯವಿಟ್ಟು ಮಾಡು ಅಲ್ಲ ಹೇಳು ತಪ್ಪು ಮಾಹಿತಿ	ತಪ್ಪಾದ ಮಾಹಿತಿಯನ್ನು ಹೇಳಬೇಡಿ ದಯವಿಟ್ಟು ಕಥೆಯನ್ನು ಸರಿಯಾಗಿ ಪಡೆಯಿರಿ

Table 1: Summarization of the preprocessing before feeding the sentences into the models.

CHAPTER 5

DETAILED DESIGN

5.1 PROPOSED ARCHITECTURE:

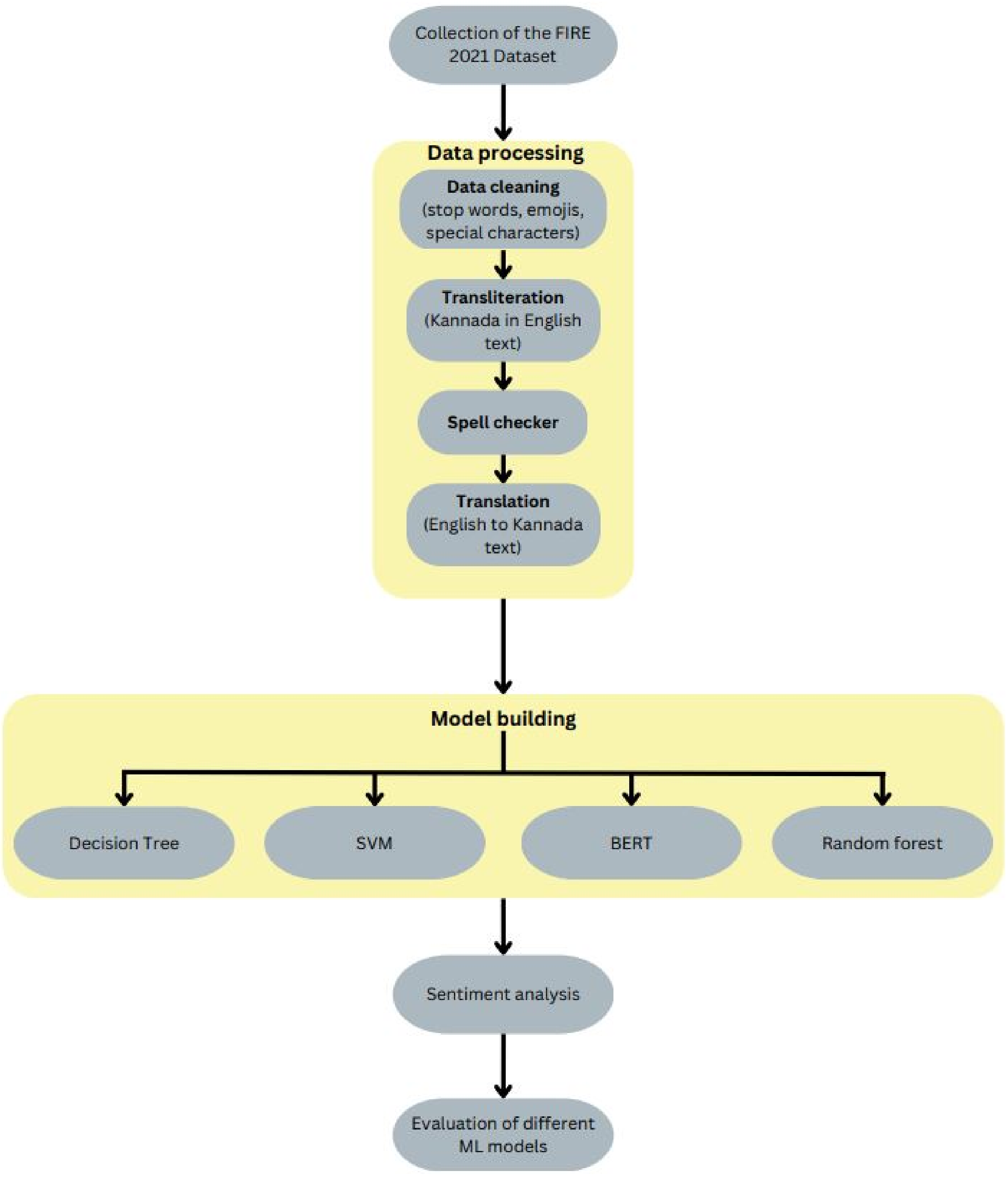

Collection of the FIRE 2021 Dataset
Data processing
Data cleaning (stop words, emojis, special characters)
Transliteration (Kannada in English text)
Spell checker
Translation (English to Kannada text)
Model building
Decision Tree
SVM
BERT
Random forest
Sentiment analysis
Evaluation of different ML models

Fig 1: Architecture Diagram

5.2 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS:

The Functional requirements are as follows:

- Type of Data: The data must be in the form of Cross Lingual Kannada-English text.
- Text input: The system should accept text input from the user.
- Text analysis: The system should be able to analyze the input text and identify which class it belongs to: “Positive, Negative, Mixed, Not-Kannada, Unknown”.
- Tools: Numpy, Pandas, Scikit-learn, iNLTK.
- Usability: The product should be easy to use, with great user experience.

The Non-Functional requirements are as follows:

- Performance: The model should process requests in real-time, with a maximum latency of X seconds.
- Scalability: The model should be able to handle an increasing number of requests as the number of users grows.
- User experience: The model should be easy to use and provide a seamless experience for the user.
- Localization: The model should support Monolingual or Cross-Lingual Kannada and English text.
- Reliability: The model should be reliable and have a high availability.

5.3 METHODOLOGY:

As proposed in our Architecture shown in Figure 1,first, we completed the collection of the cross-lingual English-Kannada FIRE dataset. We then carried out text processing at various levels to prepare the data for Sentiment Analysis. To translate the sentences to monolingual Kannada text, we used GoogleTrans.

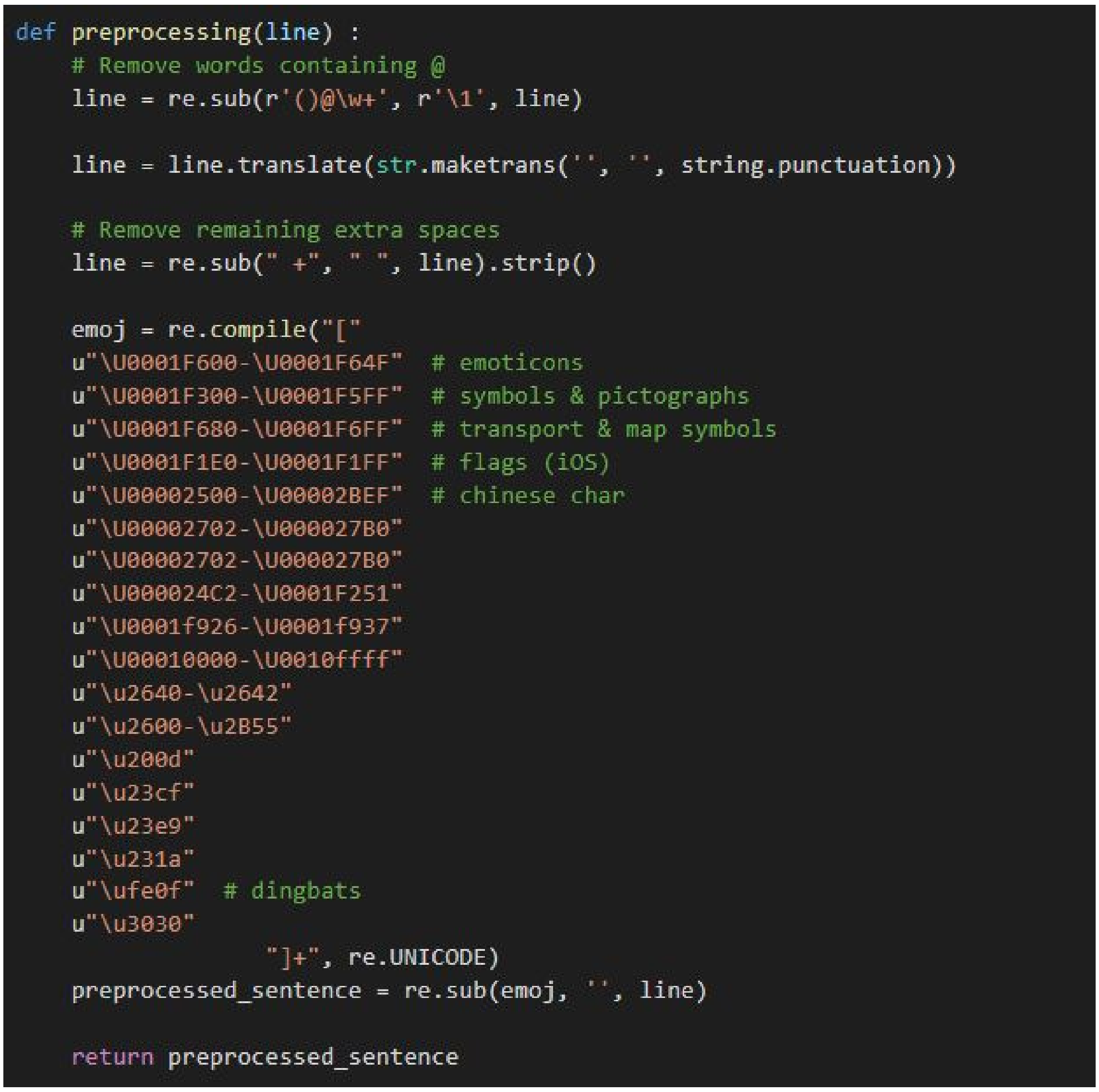
Our primary NLP model was BERT, which we fine-tuned specifically for our dataset by specifying the various parameters like max_features, learning rate, etc. Additionally, we built several machine learning models, including support vector classifiers (both linear and non-linear kernels), decision trees, random forests.

Finally, we performed Sentiment Analysis on our cross-lingual dataset, which allowed us to gain insights into the overall sentiment of the text and identify any patterns or trends.

5.4 IMPLEMENTATION:

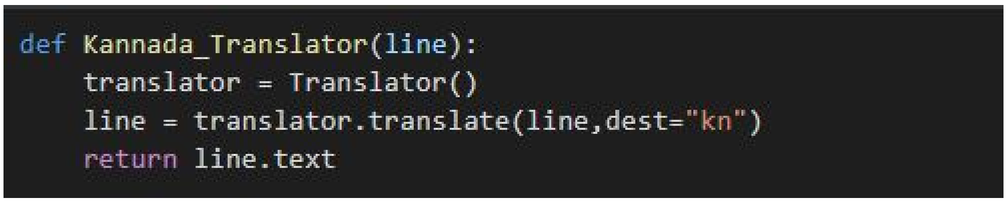
We perform different levels of text preprocessing as discussed in Chapter 4(Data collection and validation).

Fig.2, Fig.3 and Fig.4 depicts the overall text preprocessing that the data undergoes before feeding it into the model for sentiment analysis..



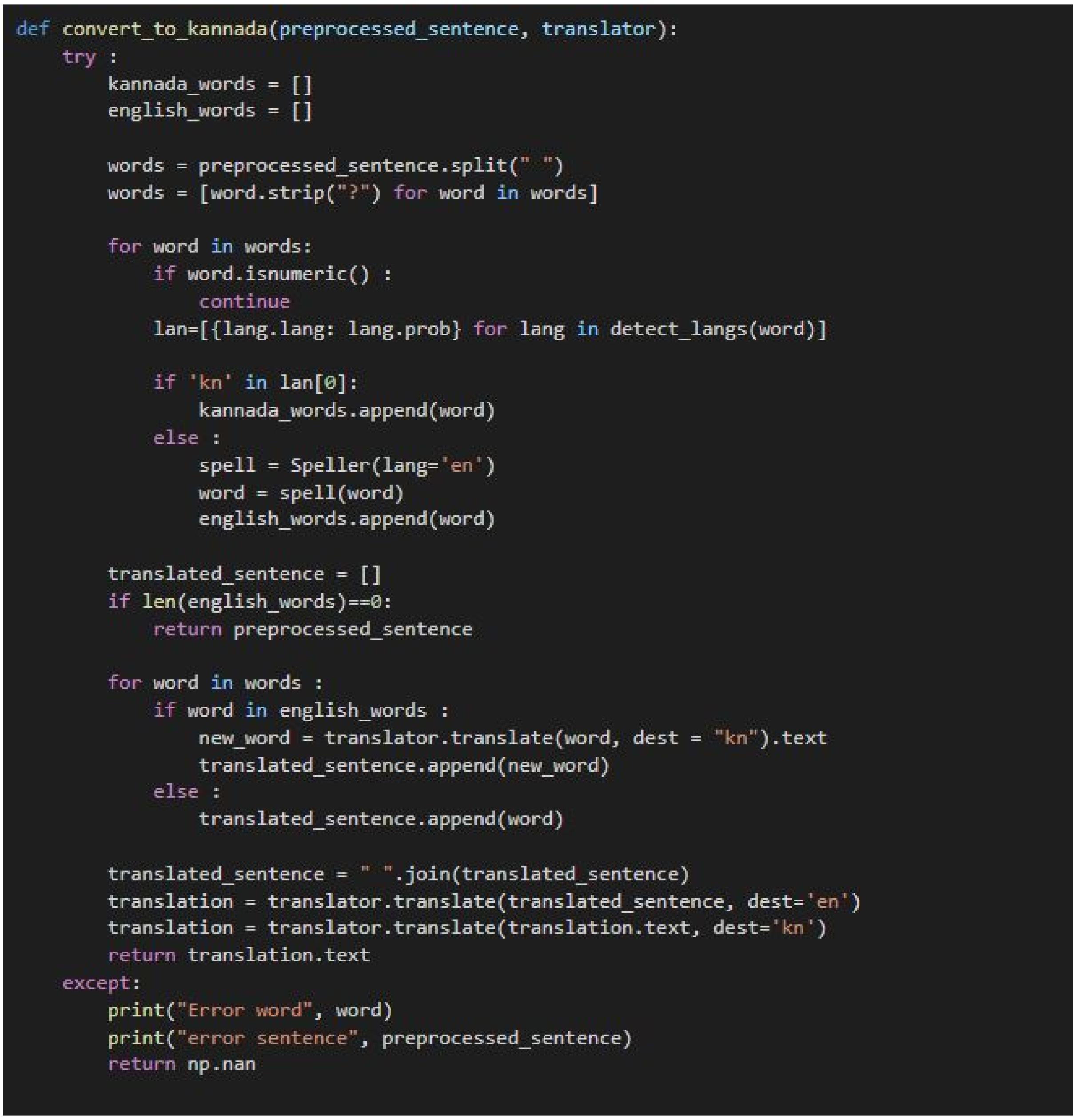
```
def preprocessing(line) :
    # Remove words containing @
    line = re.sub(r'()@\w+', r'\1', line)

    line = line.translate(str.maketrans('', '', string.punctuation))

    # Remove remaining extra spaces
    line = re.sub(" +", " ", line).strip()

    emoj = re.compile("["
    u"\U0001F600-\U0001F64F"  # emoticons
    u"\U0001F300-\U0001F5FF"  # symbols & pictographs
    u"\U0001F680-\U0001F6FF"  # transport & map symbols
    u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
    u"\U00002500-\U00002BEF"  # chinese char
    u"\U00002702-\U000027B0"
    u"\U00002702-\U000027B0"
    u"\U000024C2-\U0001F251"
    u"\U0001f926-\U0001f937"
    u"\U00010000-\U0010ffff"
    u"\u2640-\u2642"
    u"\u2600-\u2B55"
    u"\u200d"
    u"\u23cf"
    u"\u23e9"
    u"\u231a"
    u"\ufe0f"  # dingbats
    u"\u3030"
                  "]+", re.UNICODE)
    preprocessed_sentence = re.sub(emoj, '', line)

    return preprocessed_sentence
```

Fig 2. Data Cleaning



```
def Kannada_Translator(line):
    translator = Translator()
    line = translator.translate(line,dest="kn")
    return line.text
```

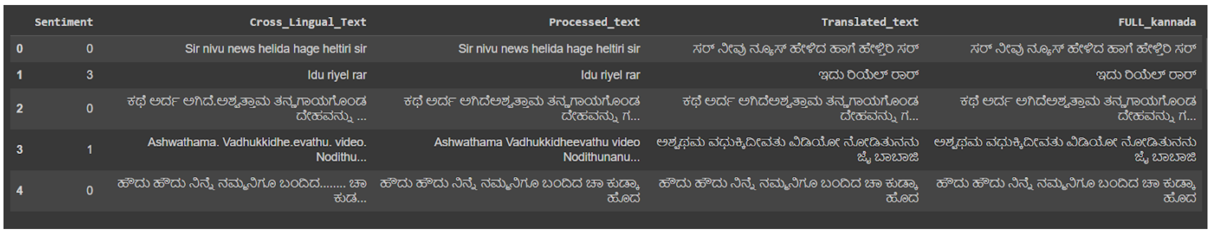
Fig 3. Transliteration of cleaned data



```python
def convert_to_kannada(preprocessed_sentence, translator):
    try :
        kannada_words = []
        english_words = []

        words = preprocessed_sentence.split(" ")
        words = [word.strip("?") for word in words]

        for word in words:
            if word.isnumeric() :
                continue
            lan=[{lang.lang: lang.prob} for lang in detect_langs(word)]

            if 'kn' in lan[0]:
                kannada_words.append(word)
            else :
                spell = Speller(lang='en')
                word = spell(word)
                english_words.append(word)

        translated_sentence = []
        if len(english_words)==0:
            return preprocessed_sentence

        for word in words :
            if word in english_words :
                new_word = translator.translate(word, dest = "kn").text
                translated_sentence.append(new_word)
            else :
                translated_sentence.append(word)

        translated_sentence = " ".join(translated_sentence)
        translation = translator.translate(translated_sentence, dest='en')
        translation = translator.translate(translation.text, dest='kn')
        return translation.text
    except:
        print("Error word", word)
        print("error sentence", preprocessed_sentence)
        return np.nan
```

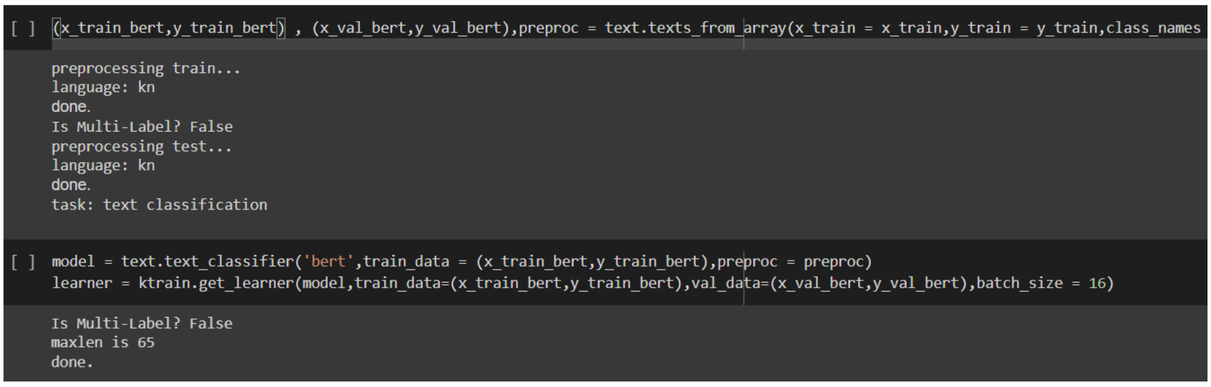
Fig 4. Spell Correction and Translation

Fig.5 shows the final output of input sentences after the preprocessing



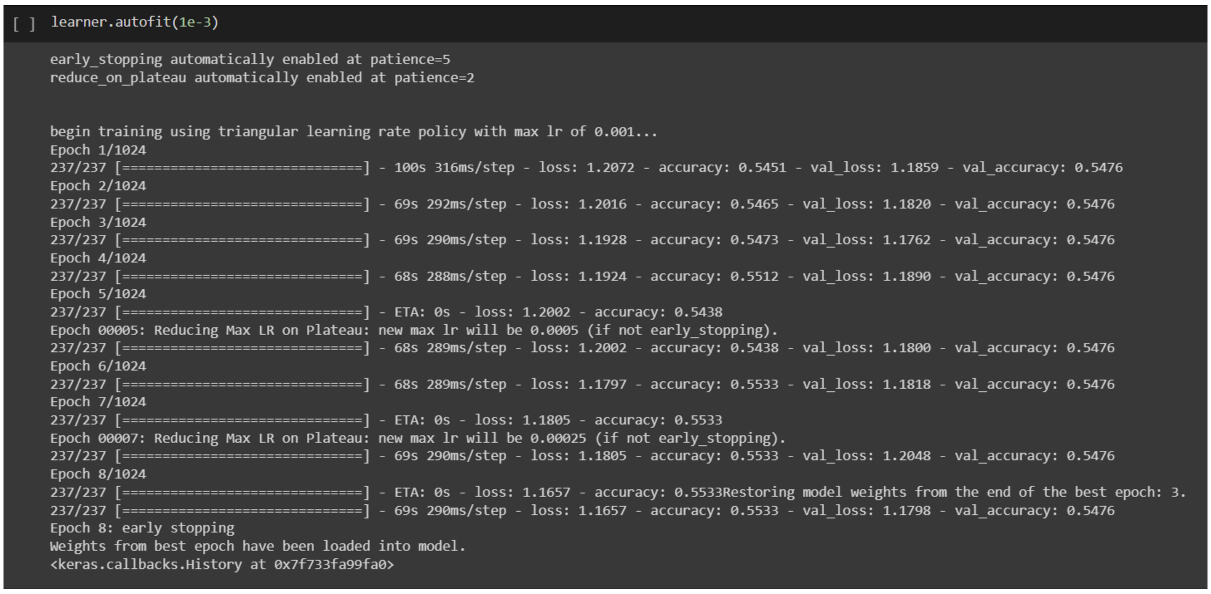
	Sentiment	Cross_Lingual_Text	Processed_text	Translated_text	FULL_kannada
0	0	Sir nivu news helida hage heltiri sir	Sir nivu news helida hage heltiri sir	ಸರ್ ನೀವು ನ್ಯೂಸ್ ಹೇಳಿದ ಹಾಗೆ ಹೇಳ್ತಿರಿ ಸರ್	ಸರ್ ನೀವು ನ್ಯೂಸ್ ಹೇಳಿದ ಹಾಗೆ ಹೇಳ್ತಿರಿ ಸರ್
1	3	Idu riyel rar	Idu riyel rar	ಇದು ರಿಯೆಲ್ ರಾರ್	ಇದು ರಿಯೆಲ್ ರಾರ್
2	0	ಕಥೆ ಅರ್ದ ಅಗಿದೆ.ಅಶ್ವತ್ತಾಮ ತನ್ನಗಾಯಗೊಂಡ ದೇಹವನ್ನು ...	ಕಥೆ ಅರ್ದ ಅಗಿದೆಅಶ್ವತ್ತಾಮ ತನ್ನಗಾಯಗೊಂಡ ದೇಹವನ್ನು ಗ...	ಕಥೆ ಅರ್ದ ಅಗಿದೆಅಶ್ವತ್ತಾಮ ತನ್ನಗಾಯಗೊಂಡ ದೇಹವನ್ನು ಗ...	ಕಥೆ ಅರ್ದ ಅಗಿದೆಅಶ್ವತ್ತಾಮ ತನ್ನಗಾಯಗೊಂಡ ದೇಹವನ್ನು ಗ...
3	1	Ashwathama. Vadhukkidhe.evathu. video. Nodithu...	Ashwathama Vadhukkidheevathu video Nodithunanu...	ಅಶ್ವಥಮ ವಧುಕ್ಕಿದೀವತು ವಿಡಿಯೋ ನೋಡಿತುನನು ಜೈ ಬಾಬಾಜಿ	ಅಶ್ವಥಮ ವಧುಕ್ಕಿದೀವತು ವಿಡಿಯೋ ನೋಡಿತುನನು ಜೈ ಬಾಬಾಜಿ
4	0	ಹೌದು ಹೌದು ನಿನ್ನೆ ನಮ್ಮನಿಗೂ ಬಂದಿದ........ ಚಾ ಕುಡ...	ಹೌದು ಹೌದು ನಿನ್ನೆ ನಮ್ಮನಿಗೂ ಬಂದಿದ ಚಾ ಕುಡ್ಕಾ ಹೊದ	ಹೌದು ಹೌದು ನಿನ್ನೆ ನಮ್ಮನಿಗೂ ಬಂದಿದ ಚಾ ಕುಡ್ಕಾ ಹೊದ	ಹೌದು ಹೌದು ನಿನ್ನೆ ನಮ್ಮನಿಗೂ ಬಂದಿದ ಚಾ ಕುಡ್ಕಾ ಹೊದ

Fig 5. Output of data processing



```
[ ] (x_train_bert,y_train_bert) , (x_val_bert,y_val_bert),preproc = text.texts_from_array(x_train = x_train,y_train = y_train,class_names

    preprocessing train...
    language: kn
    done.
    Is Multi-Label? False
    preprocessing test...
    language: kn
    done.
    task: text classification

[ ] model = text.text_classifier('bert',train_data = (x_train_bert,y_train_bert),preproc = preproc)
    learner = ktrain.get_learner(model,train_data=(x_train_bert,y_train_bert),val_data=(x_val_bert,y_val_bert),batch_size = 16)

    Is Multi-Label? False
    maxlen is 65
    done.
```

Fig 6. BERT Model



```
[ ] learner.autofit(1e-3)

    early_stopping automatically enabled at patience=5
    reduce_on_plateau automatically enabled at patience=2


    begin training using triangular learning rate policy with max lr of 0.001...
    Epoch 1/1024
    237/237 [==============================] - 100s 316ms/step - loss: 1.2072 - accuracy: 0.5451 - val_loss: 1.1859 - val_accuracy: 0.5476
    Epoch 2/1024
    237/237 [==============================] - 69s 292ms/step - loss: 1.2016 - accuracy: 0.5465 - val_loss: 1.1820 - val_accuracy: 0.5476
    Epoch 3/1024
    237/237 [==============================] - 69s 290ms/step - loss: 1.1928 - accuracy: 0.5473 - val_loss: 1.1762 - val_accuracy: 0.5476
    Epoch 4/1024
    237/237 [==============================] - 68s 288ms/step - loss: 1.1924 - accuracy: 0.5512 - val_loss: 1.1890 - val_accuracy: 0.5476
    Epoch 5/1024
    237/237 [==============================] - ETA: 0s - loss: 1.2002 - accuracy: 0.5438
    Epoch 00005: Reducing Max LR on Plateau: new max lr will be 0.0005 (if not early_stopping).
    237/237 [==============================] - 68s 289ms/step - loss: 1.2002 - accuracy: 0.5438 - val_loss: 1.1800 - val_accuracy: 0.5476
    Epoch 6/1024
    237/237 [==============================] - 68s 289ms/step - loss: 1.1797 - accuracy: 0.5533 - val_loss: 1.1818 - val_accuracy: 0.5476
    Epoch 7/1024
    237/237 [==============================] - ETA: 0s - loss: 1.1805 - accuracy: 0.5533
    Epoch 00007: Reducing Max LR on Plateau: new max lr will be 0.00025 (if not early_stopping).
    237/237 [==============================] - 69s 290ms/step - loss: 1.1805 - accuracy: 0.5533 - val_loss: 1.2048 - val_accuracy: 0.5476
    Epoch 8/1024
    237/237 [==============================] - ETA: 0s - loss: 1.1657 - accuracy: 0.5533Restoring model weights from the end of the best epoch: 3.
    237/237 [==============================] - 69s 290ms/step - loss: 1.1657 - accuracy: 0.5533 - val_loss: 1.1798 - val_accuracy: 0.5476
    Epoch 8: early stopping
    Weights from best epoch have been loaded into model.
    <keras.callbacks.History at 0x7f733fa99fa0>
```

Fig 7. Training of BERT model in the form of epochs

The above 2 images show the model building(Fig.6) of BERT and hence its training(Fig. 7) for our dataset of Dravidian codeMix 2021.

5.5 DATA FLOW AND CONTROL FLOW SEQUENCE:

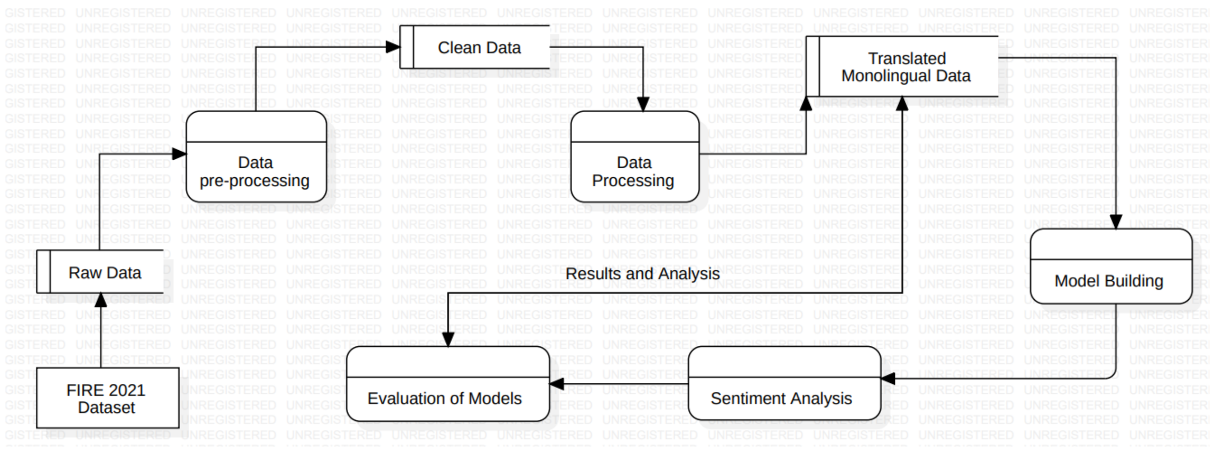


Fig 8: Data Flow Diagram

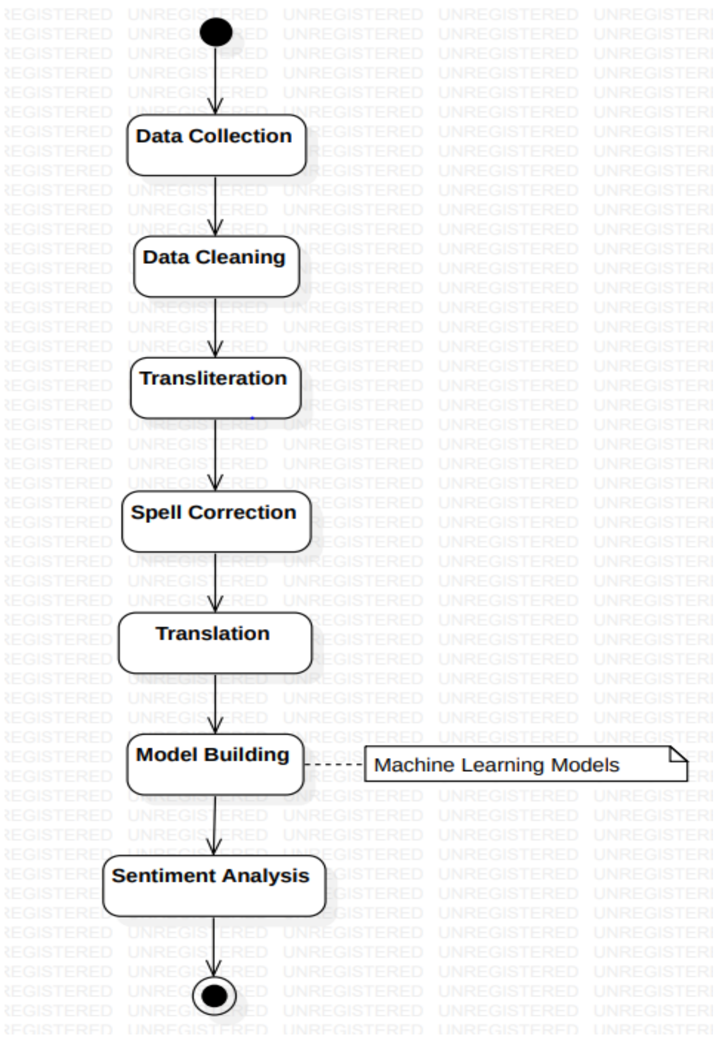


Fig 9: Control Flow Diagram

5.6 TESTING AND VALIDATION:

GIVEN INPUT	SENTENCE AFTER PROCESSING	EXPECTED OUTPUT	OBTAINED OUTPUT
U r a real indian I am full support to kannada..n.nKarnataka	ನೀವು ನಿಜವಾದ ಭಾರತೀಯ, ನಾನು ಕನ್ನಡಕ್ಕೆ ಕರ್ನಾಟಕಕ್ಕೆ ಸಂಪೂರ್ಣ ಬೆಂಬಲ ನೀಡುತ್ತೇನೆ	Positive	Positive
Ayo mostly avarige samskruti bhase barutee ansathe	ಆಯೋ ಮೋಸ್ಟ್ಲಿ ಅವರಿಗೆ ಸಂಸ್ಕೃತಿ ಭಾಸೆ ಬರುತೀ ಅನ್ಸತ್ತೆ	Positive	Positive
Guru ee desha uddhara agalla bedu bhai indian youth waste bedu	ಗುರು ಈ ದೇಶ ಉದ್ಧಾರ ಆಗಲ್ಲ ಬಿಡು ಭೈ ಇಂಡಿಯನ್ ಯೂಥ್ ವೇಸ್ಟ್ ಬೇಡು	Negative	Positive
Osam bro big fan	ಎಂಟು ಸಹೋದರ ದೊಡ್ಡ ಅಭಿಮಾನಿ	Not Kannada	Not Kannada
3ಮಿಲಿಯನ್ ಆದಮೇಲೆ ಯಾರು ನೋಡ್ತಾ ಇದ್ದೀರಾ	ಮಿಲಿಯನ್ ಆದಮೇಲೆ ಯಾರು ನೋಡ್ತಾ ಇದ್ದೀರಾ	Mixed Feeling	Positive

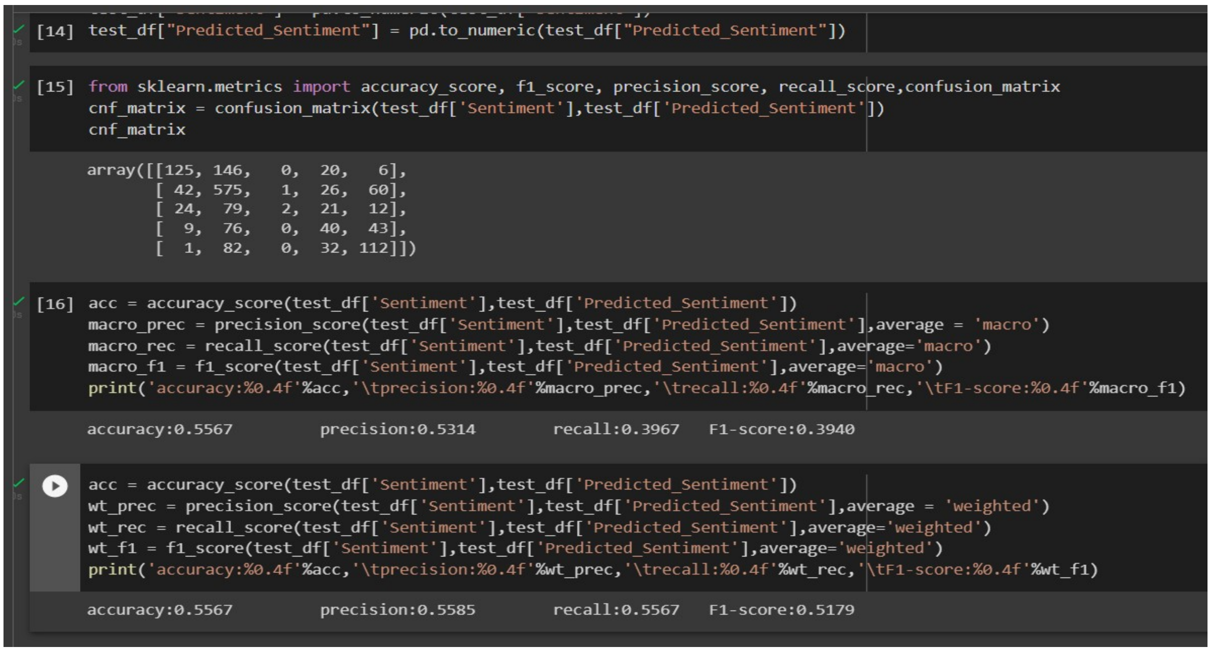
Table 2: Results of a few sentences taken from our dataset

CHAPTER 6

RESULTS AND DISCUSSION

We observed that BERT performed better than other traditional Machine Learning models. Figure 10 shows the performance metrics of the BERT model.

- **Accuracy - 55.67%**
- **Precision - 55.85%**
- **Recall - 55.67%**
- **F1-Score - 51.79%**

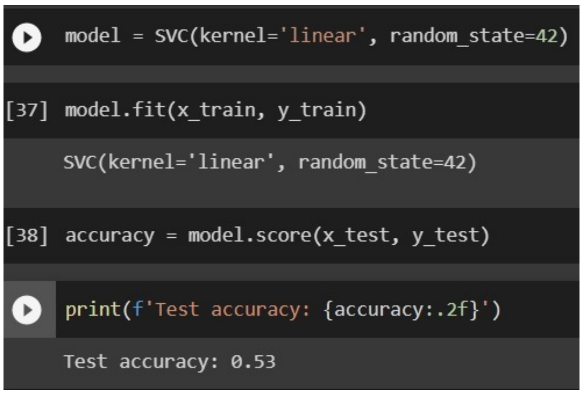


```
[14] test_df["Predicted_Sentiment"] = pd.to_numeric(test_df["Predicted_Sentiment"])

[15] from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score,confusion_matrix
     cnf_matrix = confusion_matrix(test_df['Sentiment'],test_df['Predicted_Sentiment'])
     cnf_matrix

     array([[125, 146,   0,  20,   6],
            [ 42, 575,   1,  26,  60],
            [ 24,  79,   2,  21,  12],
            [  9,  76,   0,  40,  43],
            [  1,  82,   0,  32, 112]])

[16] acc = accuracy_score(test_df['Sentiment'],test_df['Predicted_Sentiment'])
     macro_prec = precision_score(test_df['Sentiment'],test_df['Predicted_Sentiment'],average = 'macro')
     macro_rec = recall_score(test_df['Sentiment'],test_df['Predicted_Sentiment'],average='macro')
     macro_f1 = f1_score(test_df['Sentiment'],test_df['Predicted_Sentiment'],average='macro')
     print('accuracy:%0.4f'%acc,'\tprecision:%0.4f'%macro_prec,'\trecall:%0.4f'%macro_rec,'\tF1-score:%0.4f'%macro_f1)

     accuracy:0.5567         precision:0.5314        recall:0.3967   F1-score:0.3940

     acc = accuracy_score(test_df['Sentiment'],test_df['Predicted_Sentiment'])
     wt_prec = precision_score(test_df['Sentiment'],test_df['Predicted_Sentiment'],average = 'weighted')
     wt_rec = recall_score(test_df['Sentiment'],test_df['Predicted_Sentiment'],average='weighted')
     wt_f1 = f1_score(test_df['Sentiment'],test_df['Predicted_Sentiment'],average='weighted')
     print('accuracy:%0.4f'%acc,'\tprecision:%0.4f'%wt_prec,'\trecall:%0.4f'%wt_rec,'\tF1-score:%0.4f'%wt_f1)

     accuracy:0.5567         precision:0.5585        recall:0.5567   F1-score:0.5179
```

Fig 10: Metrics obtained using BERT

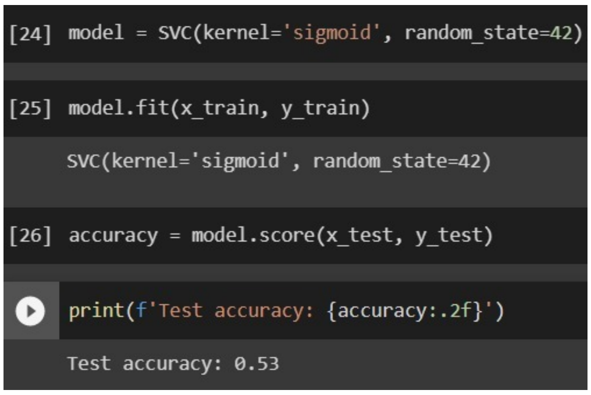
The results obtained by using Support Vector Machine model with different kernels are as follows:

- **SVC-Linear Kernel - Accuracy of 53%**
- **SVC-Sigmoid Kernel - Accuracy of 53%**
- **SVC-Polynomial Kernel - Accuracy of 49%**
- **SV-RBF Kernel - Accuracy of 52%**



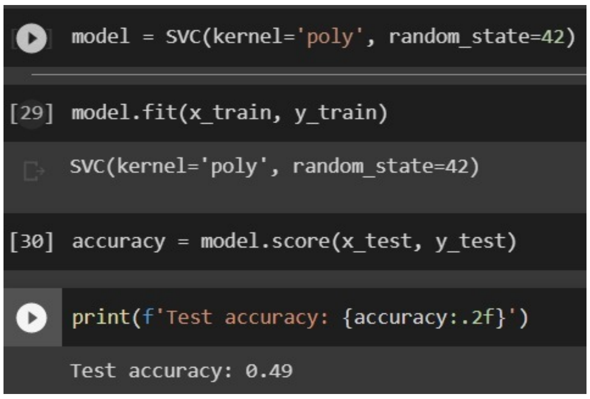
```
model = SVC(kernel='linear', random_state=42)

[37] model.fit(x_train, y_train)
     SVC(kernel='linear', random_state=42)

[38] accuracy = model.score(x_test, y_test)

print(f'Test accuracy: {accuracy:.2f}')
     Test accuracy: 0.53
```

Fig 11: SVC Model (Linear)



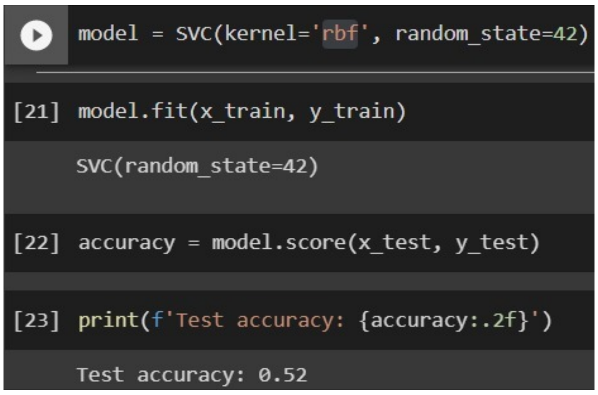
```
[24] model = SVC(kernel='sigmoid', random_state=42)

[25] model.fit(x_train, y_train)
     SVC(kernel='sigmoid', random_state=42)

[26] accuracy = model.score(x_test, y_test)

print(f'Test accuracy: {accuracy:.2f}')
     Test accuracy: 0.53
```

Fig 12: SVC Model (Sigmoid)



```
model = SVC(kernel='poly', random_state=42)

[29] model.fit(x_train, y_train)
     SVC(kernel='poly', random_state=42)

[30] accuracy = model.score(x_test, y_test)

print(f'Test accuracy: {accuracy:.2f}')
     Test accuracy: 0.49
```

Fig 13: SVC Model (Polynomial)

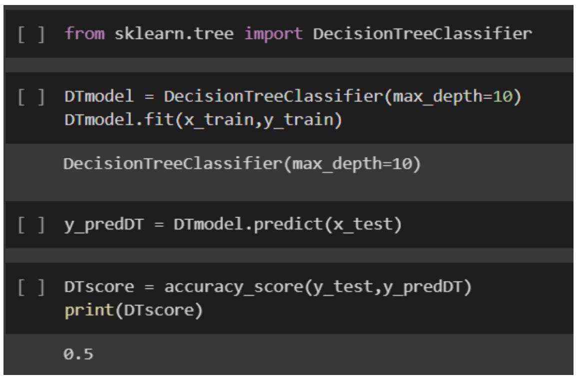


```
model = SVC(kernel='rbf', random_state=42)

[21] model.fit(x_train, y_train)
     SVC(random_state=42)

[22] accuracy = model.score(x_test, y_test)

[23] print(f'Test accuracy: {accuracy:.2f}')
     Test accuracy: 0.52
```

Fig 14: SVC Model (RBF)

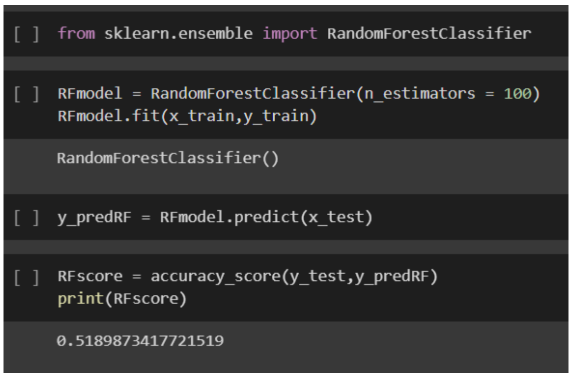
The results obtained by using Decision Tree model and Random Forest model are as follows:

- **Decision Tree - Accuracy of 50%**
- **Random Forest - Accuracy of 51.8%**



```
[ ] from sklearn.tree import DecisionTreeClassifier

[ ] DTmodel = DecisionTreeClassifier(max_depth=10)
    DTmodel.fit(x_train,y_train)
    DecisionTreeClassifier(max_depth=10)

[ ] y_predDT = DTmodel.predict(x_test)

[ ] DTscore = accuracy_score(y_test,y_predDT)
    print(DTscore)
    0.5
```

Fig 15: Decision Tree



```
[ ] from sklearn.ensemble import RandomForestClassifier

[ ] RFmodel = RandomForestClassifier(n_estimators = 100)
    RFmodel.fit(x_train,y_train)
    RandomForestClassifier()

[ ] y_predRF = RFmodel.predict(x_test)

[ ] RFscore = accuracy_score(y_test,y_predRF)
    print(RFscore)
    0.5189873417721519
```

Fig 16: Random Forest

We have inferred that certain kernels of SVC, namely Linear, Sigmoid and RBF models perform better than Decision tree and Random forest.

CHAPTER 7

CONCLUSION AND FURTHER ENHANCEMENTS

In conclusion, our project on sentiment analysis for cross-lingual Kannada-English language pairs highlights the need for more work on sentiment analysis for Dravidian languages. One major hurdle is the lack of well-annotated datasets for such languages, making it difficult to train and evaluate models for sentiment analysis. Further, research work is needed for building robust models that can handle code-mixed NLP tasks, which is a common occurrence in multilingual communities. Despite these challenges, our project aims to contribute to the ongoing efforts in this field by utilizing pre-processing techniques and ML models to analyze the sentiment of cross-lingual Kannada-English text. We hope our work will inspire more research in this area and lead to the development of more accurate and effective models for sentiment analysis in Dravidian languages.

There are several potential future enhancements that can be made to improve the cross-lingual sentiment analysis system. Firstly, incorporating more advanced pre-processing techniques such as language-specific stop-word removal, stemming and lemmatization can help to further refine the input text and improve the accuracy of the sentiment analysis. Secondly, exploring the use of deep learning models such as neural networks and transformers can potentially improve the performance of the sentiment analysis system, especially for handling code-mixed data. Additionally, leveraging domain-specific embeddings and models can help to improve the accuracy of the sentiment analysis for specific industries or domains. Lastly, integrating the sentiment analysis system with a recommendation engine can help to provide more personalized recommendations to users based on their sentiment analysis results. These enhancements can help to further refine the cross-lingual sentiment analysis system and improve its accuracy and usefulness for end-users.

REFERENCES

[1] Bharathi Raja Chakravarthi, Ruba Priyadharshini, Vigneshwaran Muralidaran, Navya Jose, Shardul Suryawanshi, Elizabeth Sherly, John P. McCra1, (2022), DravidianCodeMix: sentiment analysis and offensive language identification dataset for Dravidian languages in code-mixed text.

[2] Bharathi Raja Chakravarthi, Ruba Priyadharshini, Sajeetha Thavareesan, Dhivya Chinnappad, Durairaj Thenmozhi, Elizabeth Sherlyf, John P. McCrae, Adeep Handeh, Rahul Ponnusamy, Shubhanker Banerjee, Charangan Vasantharajan (2021) Findings of the Sentiment Analysis of Dravidian Languages in Code-Mixed Text.

[3] Satyam Dutta, Himanshi Agrawal, and Pradeep Kumar Roy (2021), Sentiment Analysis on Multilingual Code-Mixed Kannada Language.

[4] Abhinav Kumara, Sunil Saumyab and Jyoti Prakash Singh (2020), NITP-AI-NLP@Dravidian-CodeMix-FIRE2020: A Hybrid CNN and Bi-LSTM Network for Sentiment Analysis of Dravidian Code-Mixed Social Media Posts.

[5] Sahar A. El_Rahman, Feddah Alhumaidi AlOtaibi , Wejdan Abdullah AlShehri (2021), Sentiment Analysis of Twitter Data.

[6] Xi Ouyang, Pan Zhou, Cheng Hua Li, Lijun Liu(2015), Sentiment Analysis Using Convolutional Neural Networks.

[7] Kogilavani Shanmugavadivel, V. E. Sathishkumar, Sandhiya Raja, T. Bheema Lingaiah, S. Neelakandan & Malliga Subramanian(2022), Deep learning based sentiment analysis and offensive language identification on multilingual code-mixed data.

[8] Andrea Chiorrini, Claudia Diamantini, Alex Mircoli, Domenico Potena (2021), A Deep Learning Approach Combining CNN and Bi-LSTM with SVM Classifier for Arabic Sentiment Analysis.

APPENDIX - 1

RELATED MATHEMATICAL CONCEPTS

PERFORMANCE METRICS:

The four elementary matrices through which performance evaluations are predicted are : True Positive (TP), True negative (TN), False positive (FP) and false Negative (FN).

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \qquad \text{Sensitivity (Recall)} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{FP + TN} \qquad \text{Precision} = \frac{TP}{TP + FP}$$

$$\text{F-measure} = 2.\frac{Precision.Recall}{Precision + Recall} \qquad \text{AUC} = \frac{1}{2}.(Sensitivity + Specificity)$$

Fig 17: Performance Metrics

Figure 17 shows the various performance metrics used to evaluate the models and also the formula for calculating them.