Text classification of Dravidian languages

project report submitted by

Naganathan Meenakshi Sundareswaran

nm749@njit.edu, MS Data Science

Ying Wu College of Computing

New Jersey Institute of Technology

University Heights, Newark, NJ-07102 USA

Summary
India, being a diverse nation, has many regional languages and dialects. There are 22 official languages in India[6]. In this age of information, processing these natural language data is challenging. Further, the Indian regional languages are low resourced when compared to other languages such as English or other European languages. These regional languages have their own unique letters, language structure and signs. It is relatively less complex to deal with the data in English through standard ASCII codes than regional languages. Building the machines' abilities of understanding regional languages is strenuous and is done utilizing different methods. Tamil language is one of the morphologically rich, Dravidian languages[6]. Tamil is an official language of the state of Tamil Nadu in India and in the sovereign nations of Sri Lanka and Singapore. A humble attempt to utilize Naïve Bayes classifier, to classify Tamil text among English. A data set was created with a mix of English and Tamil text and the model was trained to classify the text as one of the two classes of the languages. Since the data collection part was tedious to make it fir for the project, text from other languages were not used for the sake of reducing unnecessary complexity. Having said that, the methodologies followed are essentially useful to be applied for other languages of the Dravidian family provided the right data resources are available.

Problem description and the methods of approach
The number digital records – not only tables of expenses or financial data but also documents in general, has only been catapulted by the ever-expanding potential of technologies which is not restricted only to software or computer related fields but also accepted and followed across domains. With the demand to be decisive and make decisions based on data of the past, new trends are practices are emerging very often. One of the key aspects in deriving decisions for businesses is to determine course of actions that are based on statistically and scientifically valid approaches. The term 'documents' being mentioned here represent the different form of text generated, stored and exchanged in the entirety of digital technology, including and not restricted to tweets or posts on social media, news published in digital format and all the contents that are available for the public access.

Companies are now competing to have an edge to introduce products that exclusively customized for the end users based on the past data they collected. To make effective decisions for businesses or for research, the text data collected should be classified and analysed in a methodical way. Classifying and analysing text data from myriad of sources will be tedious and put the very idea of effectiveness to check. The next viable options would be to automatically collect and classify the text. There comes teaching the machines to processes of collecting and classifying the text for them to be useful. Machines, in turn learn from the examples – training first and then treating the data collected on the go. This part of learning, as we have learnt, comes under the Supervised learning as the data collected need to be annotated and labelled first.

Classification is essence is the collection of approaches, which at the end results in classifying the data to a category (class) – here, classifying the text into the type to which it might belong – say, healthcare, pharmaceuticals, finance, entertainment, etc., Natural Language Processing (NLP) is the sub-category of Machine Learning or a combination of other fields like Artificial Intelligence and Deep Learning, that deals with teaching the machine to understand the natural language used by humans, be it in forms of speech or text. Understanding a language requires a complete understanding of the structure of the language (much like the syntax of a

programming language) and the semantics – the meaning conveyed by the structure. By annotating and labelling the data, and training it initially, helps the machine learning models to classify the data which is text here.

The Naive Bayes algorithm

Bayes' theorem forms the core of the whole concept of Naive Bayes classification that classifies the data based on the principle that 'each individual feature is independent of each other', which may not be real-world scenario, and thus the prefix 'Naïve'.

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as follows:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

where:

- A and B are called events.
- P(A | B) is the probability of event A, given the event B is true (has occurred). Event B is also termed as evidence.
- P(A) is the priori of A (the prior independent probability, i.e., probability of event before evidence is seen).
- P(B | A) is the probability of B given event A, i.e., probability of event B after evidence A is seen.

The Naive Bayes classifier is a simple probabilistic classifier that uses Bayes' theorem and strong (naive) independence assumptions to classify data[3]. One of the key reasons why the NB model works well in the text domain is because the evidences are "vocabularies" or "words" that exist in texts, with vocabularies ranging in size from thousands to millions. The Naive Bayes approach works effectively for text categorization problems due to the enormous size of evidence a(or vocabularies). Text classification can be done in two ways using Naive Bayes classifiers. The multi-variate Bernoulli model is one, and the multinomial model is the other. For the sake of text classification in the project the later has been used.

Dealing with text data

Since the raw text data cannot be loaded just like numerals, the text needs to be converted into a representation numerical form before being applied to the model. The sci-kit learn[1] allows us to manipulate the text data before being loaded. Some of the common procedure being followed as pre-processing are:

Tokenizing

In natural language processing applications, the raw text initially undergoes a process called tokenization. In this process, the given text is tokenized into the lexical units, which are the most basic units. After tokenization, each lexical unit is termed as token. Tokenization can be at sentence level or word level, depending on the category of the problem. Hence, there are 3 kinds of tokenization - a) sentence level tokenization b) word level tokenization and c) n-gram tokenization. Sentence level tokenization deals with the challenges like sentence ending detection and sentence boundary ambiguity. In word level tokenization, words are the lexical units, hence the whole document is tokenized to the set of words. The word level of tokenization is used in various language processing and text processing applications. The n-

gram tokenization is a token of n-words where ‘n’ indicates the number of words taken together for a lexical unit.. The regular expression operations[2] library of Python has been used to show examples of how the problem was approached.

Dropping Common Terms: Stop Words

There are some words which frequently occur in documents yet convey no additional meaning. Hence, removal of these non-informative words eases language processing tasks. To create a stop list, sort the terms by collection frequency (the total number of times each term appears in the document collection), then select the most often occurring terms (stop words). During indexing, these terms are removed.

There are other steps such as:

Lemmatization: the word is reduced to an acceptable form in the language after the removal of suffixes and prefixes. The reduced word is called “Lemma” and

Part of Speech (POS) tagging: the process of tagging words in a sentence with parts of speech like Noun, Pronoun, Verb, Adjective, etc. This process will be different for different languages owing to differences in grammatical structure

The explanation is restricted since the scope of this project is restricted to classify the text of two languages and explaining it in simple terms.

```
def punctuation_removal(tamil_text):
    punctuation_removed_words ="".join([t for t in tamil_text if t not in string.punctuation])
    return punctuation_removed_words
#sentence of Tamil text (https://ta.wikipedia.org/wiki/%E0%AE%85%E0%AE%A3%E0%AF%81,_%E0%AE%AE%E0%AF%82%E0%AE%B2%E0%AE%95%E0%AF%8
#with punctuation laoded and the punctuations are removed
words_without_punctuation = punctuation_removal("அணு, மூலக்கூறு, ஒளி இயற்பியல் என்பது பருப்பொருள்-பருப்பொருள் மற்றும
print(words_without_punctuation)
```

அணு மூலக்கூறு ஒளி இயற்பியல் என்பது பருப்பொருள்பருப்பொருள் மற்றும் ஒளிக்கற்றைபருப்பொருள் இடையேயான தொடர்புகள் மற்றும் இடைவினைகளை விவரிக்கும் இயற்பியலின் ஒரு துறை ஆகும் இந்த இடைவிளைவுகள் ஒரு சில அணுக்களுக்கு இடையே அமைவதாகும் பொதுவாக இந்த இடைவினைகளால் ஏற்படும் ஆற்றலானது இலத்திரன்வோல்ட்டில் அளக்கப்படுகிறது

```
def tokenization_of_words(tamil_text):
    t_words = re.split(" ",tamil_text)
    return t_words
tokenized_words = tokenization_of_words(words_without_punctuation)
print(tokenized_words)
```

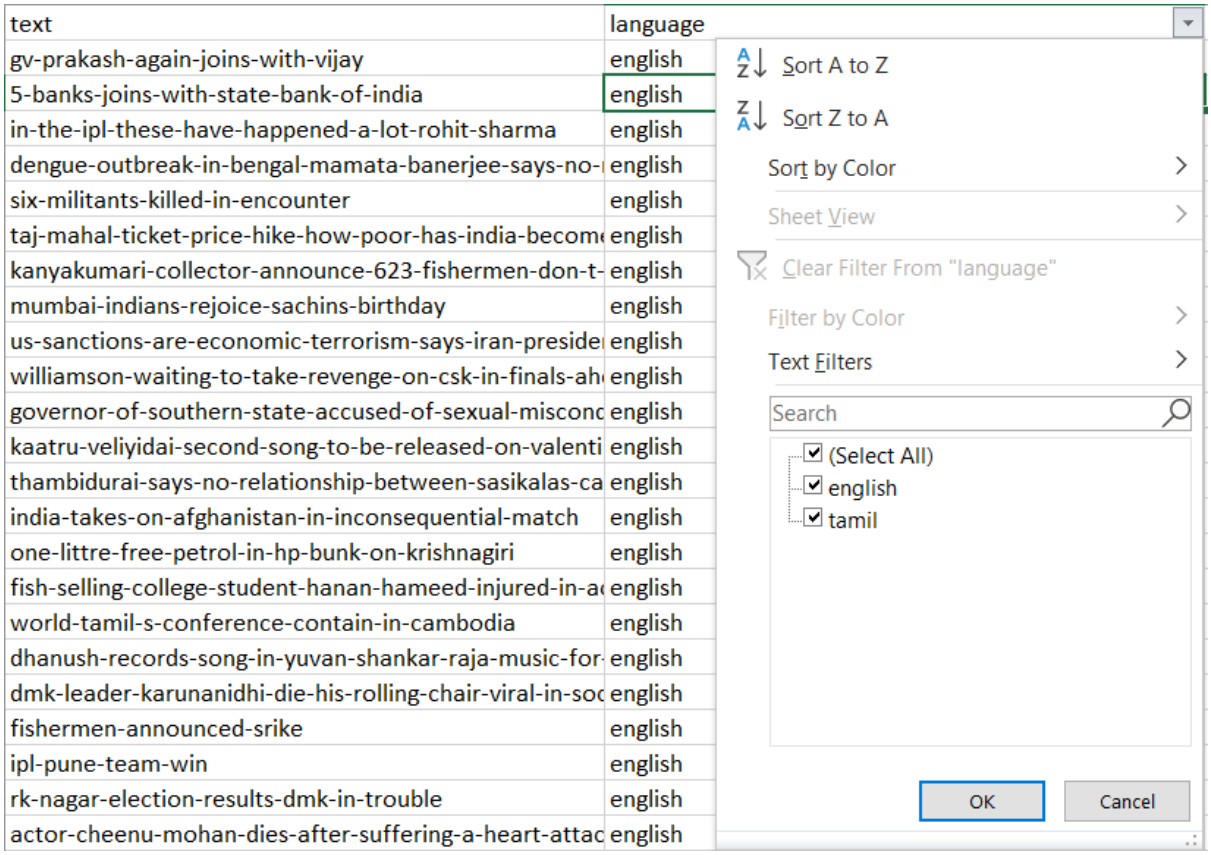
['அணு', 'மூலக்கூறு', 'ஒளி', 'இயற்பியல்', 'என்பது', 'பருப்பொருள்பருப்பொருள்', 'மற்றும்', 'ஒளிக்கற்றைபருப்பொருள்', 'இடையேயான', 'தொடர்புகள்', 'மற்றும்', 'இடைவினைகளை', 'விவரிக்கும்', 'இயற்பியலின்', 'ஒரு', 'துறை', 'ஆகும்', 'இந்த', 'இடைவிளைவுகள்', 'ஒரு', 'சில', 'அணுக்களுக்கு', 'இடையே', 'அமைவதாகும்', 'பொதுவாக', 'இந்த', 'இடைவினைகளால்', 'ஏற்படும்', 'ஆற்றலானது', 'இலத்திரன்வோல்ட்டில்', 'அளக்கப்படுகிறது']

```
stop_words = nltk.corpus.stopwords.words("tamil")
def stop_words_removal(tamil_text):
    words_removed = " ".join([t for t in tamil_text if t not in stop_words])
    return words_removed
stop_words_removed = stop_words_removal(tokenized_words)
print(stop_words_removed, end=' ')
```

அணு மூலக்கூறு ஒளி இயற்பியல் பருப்பொருள்பருப்பொருள் ஒளிக்கற்றைபருப்பொருள் இடையேயான தொடர்புகள் இடைவினைகளை விவரிக்கும் இயற்பியலின் துறை இடைவிளைவுகள் அணுக்களுக்கு இடையே அமைவதாகும் பொதுவாக இடைவினைகளால் ஏற்படும் ஆற்றலானது இலத்திரன்வோல்ட்டில் அளக்கப்படுகிறது

Data pre-processing

The data collected from the resource[8] was initially made to fit for the experiment as the prime is to recognize Tamil among English. The data was then annotated or labelled to train the model



text	language
gv-prakash-again-joins-with-vijay	english
5-banks-joins-with-state-bank-of-india	english
in-the-ipl-these-have-happened-a-lot-rohit-sharma	english
dengue-outbreak-in-bengal-mamata-banerjee-says-no-	english
six-militants-killed-in-encounter	english
taj-mahal-ticket-price-hike-how-poor-has-india-becom	english
kanyakumari-collector-announce-623-fishermen-don-t-	english
mumbai-indians-rejoice-sachins-birthday	english
us-sanctions-are-economic-terrorism-says-iran-preside	english
williamson-waiting-to-take-revenge-on-csk-in-finals-ah	english
governor-of-southern-state-accused-of-sexual-misconc	english
kaatru-veliyidai-second-song-to-be-released-on-valenti	english
thambidurai-says-no-relationship-between-sasikalas-ca	english
india-takes-on-afghanistan-in-inconsequential-match	english
one-littre-free-petrol-in-hp-bunk-on-krishnagiri	english
fish-selling-college-student-hanan-hameed-injured-in-a	english
world-tamil-s-conference-contain-in-cambodia	english
dhanush-records-song-in-yuvan-shankar-raja-music-for	english
dmk-leader-karunanidhi-die-his-rolling-chair-viral-in-soc	english
fishermen-announced-srike	english
ipl-pune-team-win	english
rk-nagar-election-results-dmk-in-trouble	english
actor-cheenu-mohan-dies-after-suffering-a-heart-attac	english

The languages are assigned to classes such that English: Class 0 and Tamil Class 1

```
df_dataset = pd.read_csv(r'C:\Users\Naganathan Vijay\Tamil_NLP_project\CS675_project_Data\train_data\train_dataset_5.csv')
print(df_dataset.head())
```

```
                                                text language
0  பேருதான் விஷால், உடம்பெல்லாம் விஷம்: தாணு காட்டம்    tamil
1                  பிக் பாஸ் வீட்டிற்குப் புது வரவு    tamil
2  பண மோசடி: 'பேட் மேன்' பட தயாரிப்பாளர் பிரேர்னா...    tamil
3        சென்சார் போர்ட் மீது வித்யா பாலன் விமர்சனம்    tamil
4  நாளை 2-வது ஒரு நாள் போட்டி: மீண்டும் அசத்துமா ...    tamil
```

```
print(df_dataset.language.value_counts())
```

```
tamil      3000
english    1800
Name: language, dtype: int64
```

```
df_dataset['classify'] = df_dataset.language.map({'english':0, 'tamil': 1})
df_dataset.head()
```

	text	language	classify
0	பேருதான் விஷால், உடம்பெல்லாம் விஷம்: தாணு காட்டம்	tamil	1
1	பிக் பாஸ் வீட்டிற்குப் புது வரவு	tamil	1
2	பண மோசடி: 'பேட் மேன்' பட தயாரிப்பாளர் பிரேர்னா...	tamil	1
3	சென்சார் போர்ட் மீது வித்யா பாலன் விமர்சனம்	tamil	1
4	நாளை 2-வது ஒரு நாள் போட்டி: மீண்டும் அசத்துமா ...	tamil	1

The data was then trained using modules from Scikit-Learn[1] library. Earlier the same library was utilized to create the vector representation of the texts of the two languages.

```
X = df_dataset.language
y = df_dataset.classify
print(X.shape)
print(y.shape)
```

```
(4800,)
(4800,)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
print("X_train: ", X_train.shape)
print("X_test: ", X_test.shape)
print("y_train: ", y_train.shape)
print("y_test: ", y_test.shape)
```

```
X_train:  (3600,)
X_test:  (1200,)
y_train:  (3600,)
y_test:  (1200,)
```

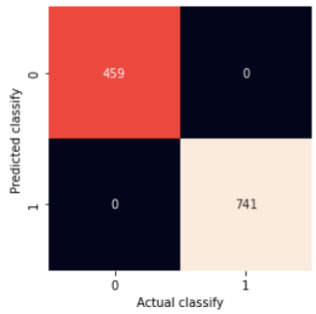
```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
X_train_array = vectorizer.fit_transform(X_train)
X_test_array = vectorizer.transform(X_test)
print(X_train_array)
print(X_test_array)
```

```
  (0, 1)        1
  (1, 0)        1
  (2, 0)        1
  (3, 0)        1
  (4, 0)        1
  (5, 1)        1
  (6, 1)        1
```

```
from sklearn.naive_bayes import MultinomialNB
mnb = MultinomialNB()
NB_model = mnb.fit(X_train_array,y_train)
y_predicted = NB_model.predict(X_test_array)
y_predicted
```

```
array([1, 1, 0, ..., 0, 1, 1], dtype=int64)
```

```
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
matrix = confusion_matrix(y_test,y_pred)
sns.heatmap(matrix.T, square = True, annot=True, fmt = "d", cbar = False)
plt.xlabel("Actual classify")
plt.ylabel("Predicted classify")
plt.show()
print("Accuracy Score is: ",round(accuracy_score(y_test, y_predicted),4))
```




```
Accuracy Score is:  1.0
```

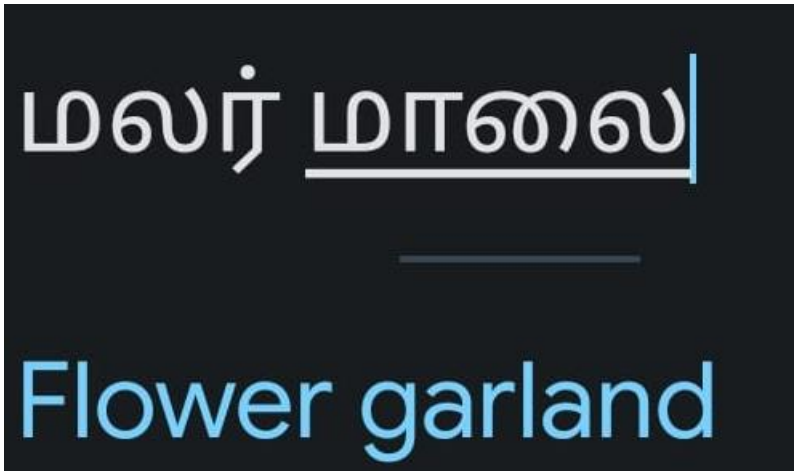
Challenges faced

Since Tamil is still a low resources language, finding a right data set is still a challenging task and the found data should meet the aim of the task taken. Here, the dataset source was created for a different intention but was then manipulated to fir the scope of the project and its results. The real-world scenarios will be different.

Conclusion and room for future works

The experiment resulted in a clean classification with a Accuracy Score of 1.0 as it's only two languages used. More challenging and efficient tasks can be carried out with more regional languages and with right dataset. As stated earlier, Tamil is a low resourced, morphologically rich language. Unlike English, the language has more inflections. One of the key hurdles is identifying the text and classifying the text and the meaning based on contest. The example provided from the source[9] explains the core of the problem. அணி (Team - ani) is a stop word in the sports domain. It has the following inflected term: அணிகளுக்கு (for teams – Anikalukku), அணிக்கு (To the team - Aṇikku), அணியின் (Of the team- Aniyin), அணிக்கும் (for the Team- Anikkum).

One more example is to check with the Google translate[11] where a meaning of the word changes contextually, which might be common in English too. However, when it comes to regional languages such as Tamil, teaching a machine might be challenging.



மலர் மாலை – Flower garland; here மாலை – noun is the word for Garland

Another meaning for மாலை is Evening

But if intended to type the plural of Garlands, what we get is plural of evening

There are various attempts to improve the process of classifying and processing Tamil language including sentimental analysis and machine translation. Open-source contributions such as Indic NLP are contributing to the resources and the ease of computing of regional languages. With practices of TF (Term Frequency) – number of times a term appearing in a document, TF-IDF (term-frequency-inverse-document frequency), complex models can be created and trained. With the advent of deep learning and reinforcement learning, the process can be even more fine tuned and more accurate translation and classification can be implemented.

References

1. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html
2. https://docs.python.org/3/library/re.html
3. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.678.537&rep=rep1&type=pdf
4. https://link.springer.com/content/pdf/10.1007/s42452-020-2983-x.pdf
5. https://en.wikipedia.org/wiki/Languages_with_official_status_in_India
6. https://en.wikipedia.org/wiki/Dravidian_languages
7. https://en.wikipedia.org/wiki/Tamil_language
8. https://www.kaggle.com/code/nainakader/tamil-news-classification/data?select=checkpoint.pth
9. https://thesai.org/Downloads/Volume12No12/Paper_67-Towards_Stopwords_Identification.pdf
10. https://github.com/anoopkunchukuttan/indic_nlp_library
11. https://translate.google.com/?sl=ta&tl=en&op=translate