**Machine Learning Algorithm**

The preprocessing and feature extraction process can be seen in the **Machine\_Learning\_Preprocessing\_Data.ipynb file.**

The SVM model can be seen in the **TestSVM.ipynb file**

We programmed a SVM algorithm with the purpose of automatically performing morpheme extraction on Tamil. Morpheme extraction is an essential part of morphological analysis. We performed a supervised classification algorithm to perform morpheme identification of words in a text as well as morpheme tagging.

For preprocessing, the words in our dataset were put into a dataframe stored in our PostgresSQL server and called. We retrieved the form, lemma (root), part of speech identifier, and had identified the morphemes within each word. We also created a separate dataframe containing all Tamil morphemes. After consulting the paper 'Morpheme Extraction and Lemmatization for Tamil using Machine Learning’, we identified the current letter of the word, prefix, vowel/consonant, noun, and verb as vital features for the algorithm [1]. We programmed to manually fill the features\_df with the appropriate features. We identified if the current letter of the word was a vowel (1) or consonant (0) and if the word was a noun indicated by a 1 in the respective column and a verb represented by a 1 in the respective column. We extracted the letters before the current letter to fill the prefix column. Finally, we manually coded to identify the morpheme boundaries in each word as our output label for the model.

Before implementing the model, Label Encoding was used on the letters column and prefix columns to allow for numerical values only. Features selected were the index, vowel/consonant, noun, verb, letter\_label\_encoded, and prefix\_label\_encoded. A Standard Scaler was applied to avoid numerical overflow while running the model. Furthermore, the output label for our model was defined as morpheme boundary.

After experimenting, we opted to utilize the default 75% training and 25% testing data split. We ran the model using a linear kernel and received an accuracy of 50%. We then tried a polynomial kernel which resulted in an increased accuracy of 70%. Ultimately, we selected a rbf kernel with a gamma of 0.8 to achieve an accuracy of 76%.

We opted to use a supervised machine learning approach after referencing the paper 'Morpheme Extraction and Lemmatization for Tamil using Machine Learning' which used SVM to segment Tamil morphemes with a promising F score of 90%. [1]

A Support Vector machine is a good choice since this is a binary classification problem and it can provide a high accuracy with less computational power. It calculates the ideal hyperplane to separate the data by maximizing the margin distance at feature boundaries. We plan to utilize this model on a bigger dataset. In the future, we also plan to use our tagged dataset to train and run an SVM on Old Tamil to segment morphemes. There will be a loss in accuracy due to changes in grammar rules but the hope is to either use FSA or an unsupervised approach to accommodate these changes. Our second Tamil database will be used as training data to refine the SVM before running it on an old Tamil dataset.

Pros of SVM:

* Well developed statistical learning theory
* Can identify the non-linear relationships of morphological features from training datasets
* Risk of overfitting is low
* Can solve complex problems with appropriate kernel function

Limitations of SVM:

* Can have long training time for large datasets
* Limited to parameters C and gamma

References:

[1] Kumar, M. A., & Kp, S. (2015). AMRITA\_CEN@FIRE-2014: Morpheme Extraction and Lemmatization for Tamil using Machine Learning. <https://doi.org/10.1145/2824864.2824883>