# Predict mood of the song from lyrics

Manojit Chakraborty Sayan Ghosh Shubham Das Siddesh Sawant

**International Institute of Information Technology, Hyderbad**

Course : Statistical Mathods in AI

**Group ID 38**

April 29, 2019

## Project Overview

### Main Goal of the Project

Develop a Mood Classification System from song lyrics as well by combining a wide range of semantic and stylistic features extracted from textual lyrics.

### Problem definition

The primary focus of the project is to develop a model that analyses a song by its lyrics and to detect the mood in the song. A few of such work have been done on English songs, but we are to explore it by also putting Hindi songs to it. The methodology used by far is the translation of songs in other languages to English and then analyse, because a lot of annotated datasets are available in English.

## Project Overview

### Results of the project

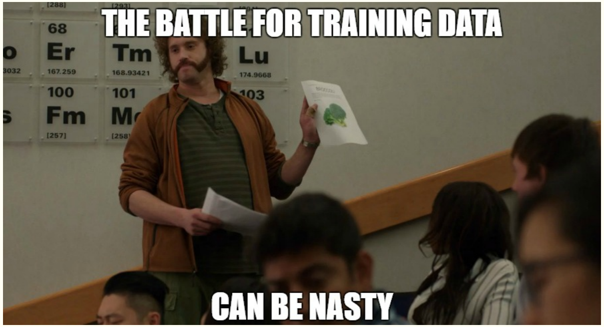
The project aims at developing a system to predict songs' mood and clustering them which can be used in a Recommendation System based on users' listening habit and their choice of genres. The project is also based on predicting mood of Bollywood songs by translating their lyrics to Hindi and using approaches of NLP, Polarity based Classification by combining a wide range of semantic and stylistic features extracted from the lyrics.

## Project Milestones and Timelines

- Studying project related research papers and literature.
- Understanding and interpreting the implementations already out there. [ *27th March, 2019* ]
- Exploring places from where dataset to be created.
- Data preprocessing and feature selection. [ *10th April, 2019* ]
- Model exploration and selection.
- Model Implementation
- Measuring performance of the model. [ *17th April, 2019* ]
- Improving the model by changing certain parameters.
- Reporting the results and inferences. [ *29th April, 2019* ]
- Project Report, Presentation Preparation. [ *29th April, 2019* ]

## Main Challenges

- Training Dataset Preparation (Using English Song Lyrics)
- Test Dataset Preparation (Using Hindi Song Lyrics translated to english)
- Data Preprocessing
- Feature Engineering
- Machine Learning Model Preparation
- Model Selection and Prediction



## Training Dataset Preparation

The dataset that we start with is a 10000 Song subset of the **Million Song Dataset**. Now we do the following :

- Store the dataset into a Pandas Dataframe with features **File Name, Artist Name, Song Name**
- Using the Artist Name and Song Title, We are fetching lyrics for all the songs using **PyLyrics** package, which uses `LyricWikia.com` API to get lyrics for songs
- We are creating our model using english lyrics. So all the song lyrics containing any other language are removed from the Dataset.
- Now we will use `Last.FM` API to extract Tags for the remaining 3000 songs in our dataset. Tags can be based on Genre, Mood, Artist Type etc.

## Training Dataset Preparation

In the paper **Lyric Text Mining in Music Mood Classification**, `Hu et.al`, Last.FM tags are being grouped into 18 categories according to different human moods. We have taken 10 groups from it and distributed them into our Mood Categories - **Happy, Sad, Angry, Relax.**

- Happy Tags : cheerful, cheer up, festive, jolly, gleeful etc.
- Sad Tags : sadness, unhappy, melancholic, depress etc.
- Angry Tags : anger, angry, choleric, fury, outraged, rage etc.
- Relaxed Tags : calm, comfort, quiet, serene, mellow etc.

By correlating the Tags that we found from `Last.FM` and the tag groups generated by us, we are creating the Class Labels for Moods in our Dataset.
From the paper **Multimodal Music Mood Classification by Fusion of Audio and Lyrics**, `Hao et.al` , we get to know about 777 other songs, already categorized into Happy, Sad, Angry, Relaxed. We append this dataset with our previous dataset for our final training dataset.

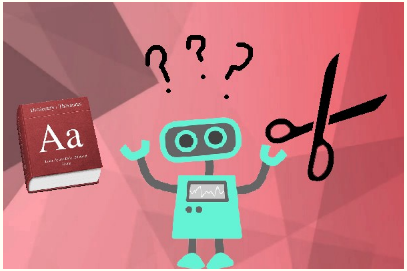
## Test Dataset Preparation

- We manually collected over 250 Hindi song lyrics and stored into dataframe
- Using **Google Translate API**, we auto-translated them into English lyrics and manually labelled them for performance checking.
- As our test dataset is labelled manually and training dataset is auto-labelled from the tags of Last.FM and Hao's paper, we add a fraction of translated-to-english lyrics to our training dataset to reduce the bias of the testing dataset.

## Data Preprocessing

We have lyrics in our data. To Preprocess the lyrics column we do following preprocessing steps

- **Tokenization** : Taking a text or set of text and breaking it up into its individual words
- **Stop-word Removal** : Stop words such as *the, a, an, in* are removed from lyrics
- **Punctuation Removal** : Removes punctuation from lyrics
- **Stemming** : Reducing inflected (or sometimes derived) words to their word stem, base or root form
- **Lemmatization** : Process of grouping together the inflected forms of a word so they can be analyzed as a single item, identified by the word's lemma, or dictionary form



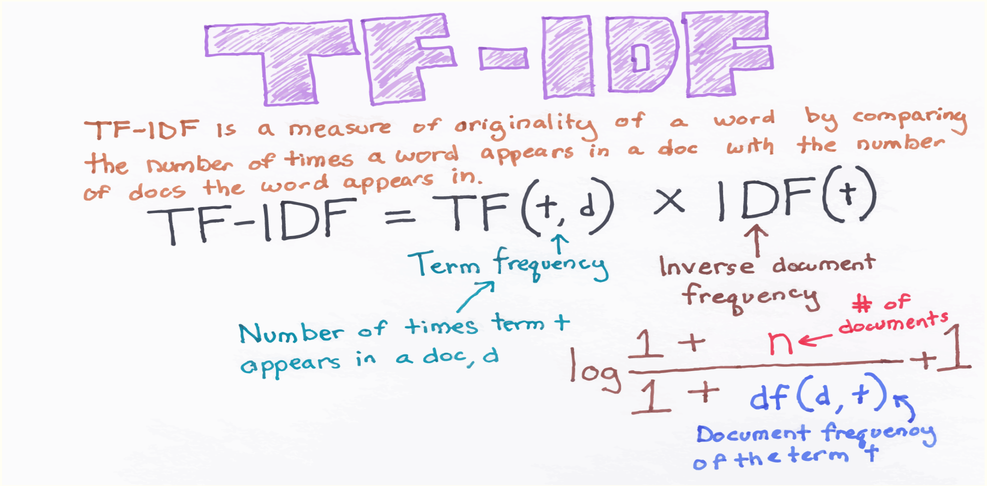
## Feature Engineering

### Wordlevel TfIdf

tfidf, short for **term frequency inverse document frequency**, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The tfidf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word

### N GRAM MODEL

An **n-gram** is a contiguous sequence of n items from a given sample of text or speech. An n-gram model is a type of probabilistic language model for predicting the next item in such a sequence in the form of a (n - 1) order Markov model. Two benefits of n-gram models are simplicity and scalability : with larger n, a model can store more context with a well-understood space-time tradeoff, enabling small experiments to scale up efficiently.



TF-IDF

TF-IDF is a measure of originality of a word by comparing the number of times a word appears in a doc with the number of docs the word appears in.

$$\text{TF-IDF} = \text{TF}(t, d) \times \text{IDF}(t)$$

Term frequency

Number of times term t appears in a doc, d

Inverse document frequency

$$\log \frac{1 + n}{1 + df(d, t)} + 1$$

# of documents

Document frequency of the term t

## Learning Models

We have created Training, Validation datasets for each of these 3 NLP Models

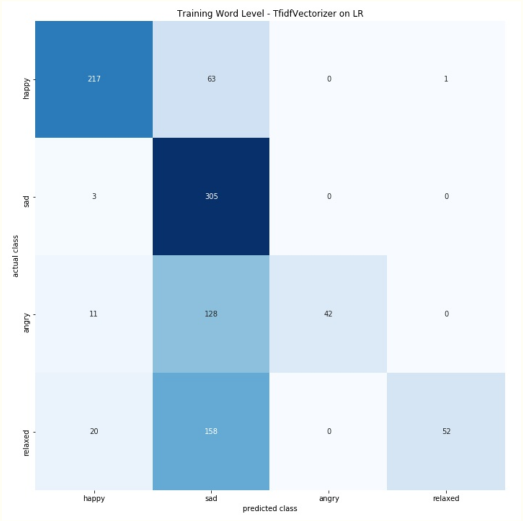
- **CountVectorizer**
- **WordLevel tfidf**
- **NGram Vectors**
- **Word2vec Embedding vectors**

The following classifiers are used for model preparation and testing :

- Random Forest
- Multinomial Naive Bayes
- Logistic Regression
- Ensemble - Bagging
- Ensemble - Boosting
- Support Vector Machines
- Convolutional Neural Net

# Validation Dataset Results

The validation dataset, which was created from a fraction of the training dataset of english song lyrics, saw an accuracy of atmost 53% for the TFIDF model using Logistic Regression Model as the machine learning model. Here is the confusion matrix -



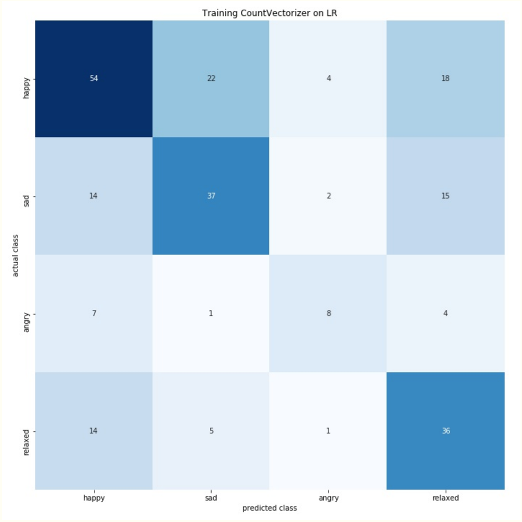
## Classifier Results

Here are the accuracy scores on our test datasets for each 3 NLP models

| Classifier | CountVect | TfidfVect | NGramVect |
|---|---|---|---|
| Naive Bayes | 47.2 | 40.5 | **54.1** |
| Random Forest | 57.2 | **58.7** | 52.5 |
| Logistic Regression | 55.8 | 56.2 | 46.3 |
| Boosting | 46 | 42.5 | 33.1 |
| Bagging | 55.3 | 53.5 | 51.7 |
| SVM | 55.7 | **60** | **55** |

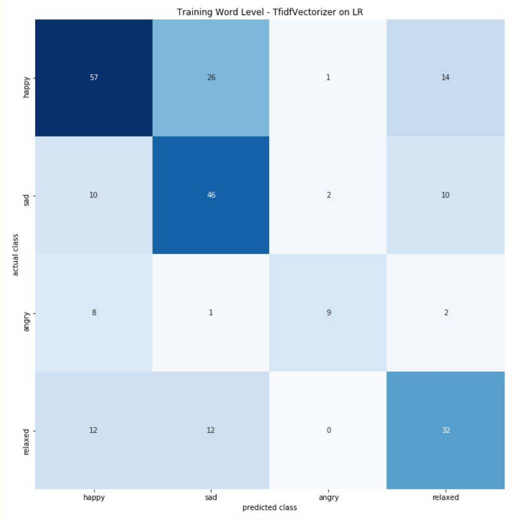
Table: Different agents and corresponding networks

# Logistic Reg on CountVectorizer



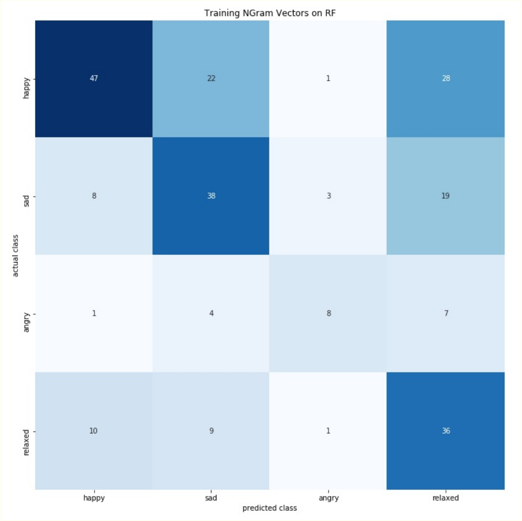
|  | happy | sad | angry | relaxed |
|---|---|---|---|---|
| happy | 54 | 22 | 4 | 18 |
| sad | 14 | 37 | 2 | 15 |
| angry | 7 | 1 | 8 | 4 |
| relaxed | 14 | 5 | 1 | 36 |

# Random Forest on TfidfVectorizer



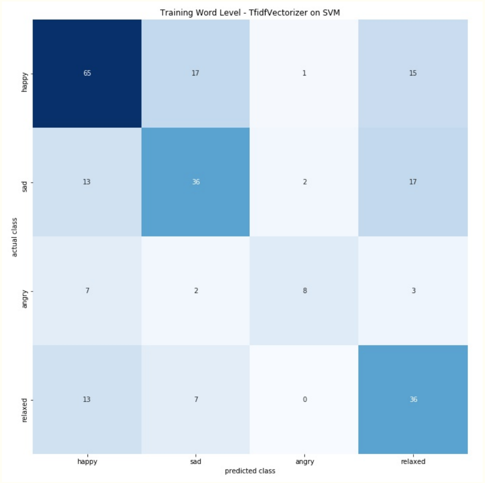
| | happy | sad | angry | relaxed |
|---|---|---|---|---|
| happy | 57 | 26 | 1 | 14 |
| sad | 10 | 46 | 2 | 10 |
| angry | 8 | 1 | 9 | 2 |
| relaxed | 12 | 12 | 0 | 32 |

## Random Forest-Bagging on NGram Vectorizer



| actual \ predicted | happy | sad | angry | relaxed |
|---|---|---|---|---|
| happy | 47 | 22 | 1 | 28 |
| sad | 8 | 38 | 3 | 19 |
| angry | 1 | 4 | 8 | 7 |
| relaxed | 10 | 9 | 1 | 36 |

## Best Model for us



|  | happy | sad | angry | relaxed |
|---|---|---|---|---|
| happy | 65 | 17 | 1 | 15 |
| sad | 13 | 36 | 2 | 17 |
| angry | 7 | 2 | 8 | 3 |
| relaxed | 13 | 7 | 0 | 36 |

Figure: SVM with TFIDF Vectorizer

## Classification Report

During Validation with English lyrics validation dataset,along with a highest accuracy of 56%, we got the following -

- **Precision** : 0.63
- **Recall** : 0.56
- **F1-Score** : 0.58

During testing with Bollywood Hindi Songs translated-to-English lyrics dataset, using SVM, we got an accuracy of 60% and the following -

- **Precision** : 0.63
- **Recall** : 0.62
- **F1-Score** : 0.63

## Future works

- We trained our model using Word2Vec along with CNN with pre-trained genism library.However, we obtained approx. 40% accuracy due to which we decided not to go along with it due to small size of training data and might need to change the parameters to obtain better results.
- Also, we can further add a Web based Search and Recommender Engine where a user can obtain his songs according to his choice of mood using our model.
- Using Selenium and BeautifulSoup4 library we can automate the action of retrieving and parsing the data and then passing it through Google translator API to the get the English lyrics for generating our test dataset.

## Bibliography

[1] Xue, H., Xue, L., Su, F. Multimodal Music Mood Classification by Fusion of Audio and Lyrics. MMM 2015, Part II, LNCS 8936, pp. 26â37, 2015.

[2] X. Hu, J. S. Downie, A.Ehmann. Lyric Text Mining in Music Mood Classification. ISMIR 2009, Oct. 2009.

[3] Sebastian Raschka. MusicMood: Predicting the mood of music from song lyrics using machine learning. arXiv:1611.00138v1 [cs.LG] 1 Nov 2016

[4] Cecilia Ovesdotter Alm. Emotions from text: machine learning for text-based emotion prediction. (HLT/EMNLP), pages 579â586, Vancouver, October 2005.

[5] Patra, Das, BandyoPadhyay. Automatic Music Mood Classification of Hindi Songs. IJCNLP 2013, pages 24â28, Nagoya, Japan, October 14, 2013.