

# Machine Learning to aid mental health among youth during COVID-19

Dissertation submitted in part fulfilment of the requirements

for the degree of

MSc Data Analytics

At Dublin Business School

Yash Vijay Modha

## Declaration

I declare that this applied project that I have submitted to Dublin Business School for the award of MSc Data Analytics is the result of my own investigations, except where otherwise stated, where it is clearly acknowledged by references. Furthermore, this work has not been submitted for any other degree.

Signature: Yash Vijay Modha

Student Number: 10548773

Date: 24<sup>th</sup> May 2021

## Acknowledgements

I would like to offer my sincere gratitude to the Dublin Business School for allowing me with this opportunity to conduct the research.

I would like to thank my supervisor Ms. Terri Hoare for all her guidance from the beginning for all the extensive suggestions towards the constructive planning and development of my research. I really appreciate her enthusiasm and thank her for investing her precious time throughout the dissertation building process. I would also like to thank her for suggesting me to use H2O.ai due to which I was able to pre-process the data quickly and efficiently. I would also like to thank H2O.ai for providing me a better way to contribute to my research.

I would also like to thank all the professors at the University who guided me throughout the academic year.

Lastly I would like to grant my sincere gratitude towards my family and friends for all their support throughout the process.

## Abstract

This research presents a comparative study of specialized deep learning and state of the art machine learning approaches for multiclass classification of six emotions: joy, sadness, surprise, fear, love, anger. Specialized deep learning algorithm Bi-LSTM; state of the art H2O ANN; traditional SVM and state of the art H2O Gradient Boosting Machines are applied on vectorised text. Cross validated performance using different vectorization techniques: text to sequences, word to vectors and TF-IDF are presented. The specialized Bi-LSTM on text to sequence vectorised data outperforms SVM and both outperform H2O ANN. A Gradient Boosted Machine age classifier is used to stratify test data. The traditional TF-IDF applied SVM outperforms the Bi-LSTM model on both Youth and Adult test data. The research is further extended to present a chatbot deployment of the emotion classifier.

## Table of contents

Chapter No.	Title	Page No.
	Declaration	2
	Acknowledgements	3
	Abstract	4
	List of Figures	6
	List of Tables	7
	List of Abbreviations	8
	List of Algorithms	9
1	Introduction	10
	1.1 Corona virus and mental health	10
	1.2 Research Problem	10
	1.3 Scope	11
	1.4 Limitations	11
2	Literature Review	12
3	Research Methodology and Methods	15
	3.1 Introduction	15
	3.2 CRISP-DM	15
4	Discussion	31
	5.1 Data Overview	31
	5.2 Model Results	31
	5.3 Results	41
	5.4 Limitations	42
	5.5 Future research	42
5	Conclusions	43
6	References	44
	Appendices	47

## List of Figures

Figure No.	Title	Page No.
1.1.1	Emotions trends during the early stages of the COVID-19 pandemic.	10
3.1	CRISP-DM Methodology	15
3.3.1	Histogram for emotions	17
3.3.2	TF-IDF pre-processing	18
3.3.3	Word2vec synonyms	19
3.3.4	Word2vec data sample	19
3.3.5	Word2vec pre-processing	19
3.3.6	Histogram for text	20
3.3.7	Text to sequence	20
3.3.8	Label to index	20
3.3.9	m3inference example	21
3.4.1	Basic LSTM cell	23
3.4.2	Hidden State	23
3.4.3	Sequential Model Summary	24
3.4.4	SVM possible hyperplanes	25
3.4.5	Hinge Loss Function	25
3.4.6	Regularization parameter Loss Function	25
3.4.7	Gradients	25
3.4.8	Gradient – No misclassification	26
3.4.9	Gradient – Misclassification	26
3.4.10	Feedforward ANN	27
3.6.1	Flask API	30
3.6.2	‘Shelbot’ Emotion Chatbot	30
4.5.1	LSTM model summary	31
4.5.4	Word2vec H2O workflow	34

## List of Tables

Table No.	Title	Page No.
3.3.1	Emotion Counts	18
4.5.1	LSTM Confusion Matrix	31
4.5.2	LSTM Emotion Age	32
4.5.3	TF-IDF LinearSVC Confusion matrix	32
4.5.4	Word2vec SVM Rapid Miner performance	32
4.5.5	Word2vec SVM Rapid Miner Confusion matrix	33
4.5.6	TF-IDF LinearSVC Emotion Age	33
4.5.7	Word2vec 100 H2O Deep Learning performance	34
4.5.8	Word2vec 100 H2O Deep Learning Confusion matrix	34
4.5.9	Word2vec 500 H2O Deep Learning performance	35
4.5.10	Word2vec 500 H2O Deep Learning Confusion matrix	35
4.5.11	TF-IDF H2O Deep Learning performance	35
4.5.12	TF-IDF H2O Deep Learning Confusion matrix	36
4.5.13	Word2vec Rapid Miner Deep Learning Confusion matrix	36
4.5.14	Word2vec Rapid Miner Deep Learning performance	36
4.5.15	TF-IDF Deep Learning H2O Age Confusion matrix	37
4.5.16	TF-IDF Deep Learning H2O Age performance	37
4.5.17	Word2vec Deep Learning H2O Confusion matrix	37
4.5.18	Word2vec Deep Learning H2O performance	37
4.5.19	Word2vec Deep Learning H2O Emotion Age performance	38
4.5.20	TF-IDF GBM H2O Age Confusion matrix	38
4.5.21	TF-IDF GBM H2O Age performance	38
4.5.22	Word2vec GBM H2O Age Confusion matrix	39
4.5.23	Word2vec GBM H2O Age performance	39
4.5.24	Word2vec GBM H2O Age counts	39
4.5.25	Comparison of accuracy scores for emotion classification	40
4.5.26	Comparison of recall scores for emotion classification	40
4.5.27	Comparison of approaches for age classification	41
4.5.28	Comparison of age classifier model recall scores	41
4.5.29	Comparison of approaches applied on Youth and Adult classified emotion data	41

## List of Abbreviations

- AI – Artificial Intelligence
- ML – Machine Learning
- Word2vec – Word to vector
- CRISP-DM - Cross-Industry Standard Process for Data Mining
- LSTM – Long Short Term Memory
- SVM – Support Vector Machine
- GBM – Gradient Boosting Machines
- GPU – Graphical Processing Unit
- GUI – Graphical User Interface
- ANN – Artificial Neural Network
- TFIDF – Term Frequency Inverse Document Frequency
- GloVe – Global Vectors for Word Representation
- RM - RapidMiner
- DL – Deep Learning
- CNN – Convolutional Neural Network
- LinearSVC – Linear Support Vector Classifier
- RNN – Recurrent Neural Network
- GBDT – Gradient Boosted Decision Trees
- NLTK – Natural Language Toolkit
- Bi-LSTM – Bi-directional Long Short Term Memory
- 2D – Two Dimensional
- XGB – Xtreme Gradient Boosting
- API – Application Programming Interface

- BR – Binary Relevance
- MLKNN – Multilabel K Nearest Neighbor
- NB – Naïve Bayes
- SSWE – Sentiment-specific Word Embeddings
- KNN – K Nearest Neighbor
- ETC – Extra Trees Classifier
- XGBoost – Xtreme Gradient Boost
- YALE – Yet Another Learning Environment

### List of Algorithms

- Bi-directional Long Short Term Memory
- Linear Support Vector Classifier
- Gradient Boosting Machine
- Deep Learning
- Word to vectors
- Term Frequency Inverse Document Frequency

## Chapter 1 - Introduction

### 1.1 Corona virus and Mental Health

Since the beginning of first reported cases of covid when it was disclosed by the Chinese authorities in January 2020 (*About the virus*, no date), a lot of people began expressing their thoughts and emotions on numerous social media websites like Twitter, Instagram, Facebook etc. There was a shift of emotions from anger to fear due to the shortages of COVID-19 tests as well as limited medical supplies. Other emotions like sadness and joy also flared up. Accelerated progression within a span of few weeks were observed pointing to global COVID-19 sentiments. (Lwin *et al.*, 2020) Since the beginning of this pandemic people are freely expressing their emotions through social media sites, these are not only emotions but also their mental state that needs to be addressed for the betterment of their future.

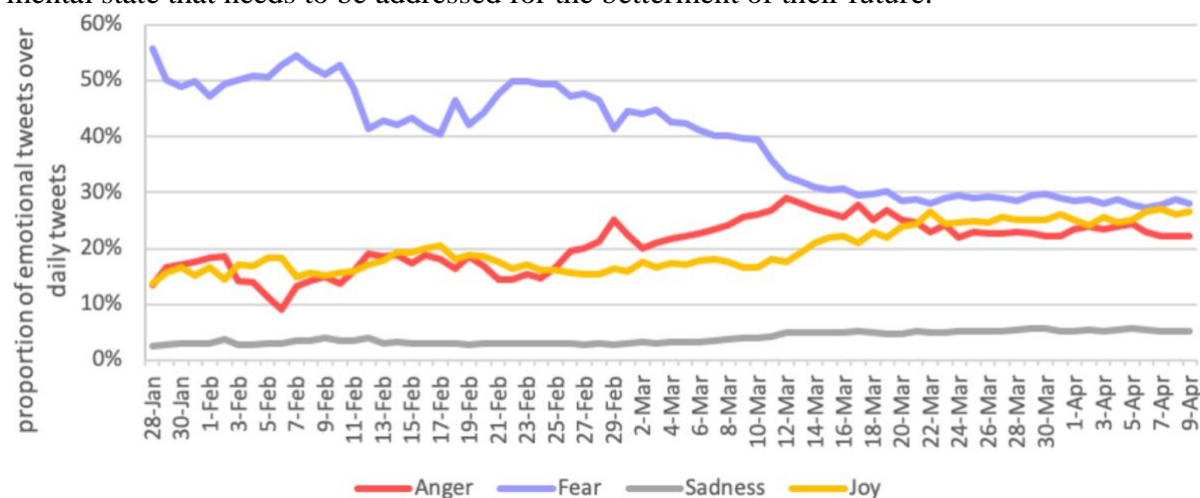


Figure 1.1.1 (Lwin *et al.*, 2020)

### 1.2 Research Problem

Emotion classification is an important multiclass classification problem and needs to be addressed especially during these crucial times. The mental state of health of people all around the globe is vulnerable and the best solution is to keep yourself occupied in indoor activities. This research aims at creating an accurate model that can classify emotions in text and implement the same in a chatbot which can help the chatbot suggest relevant activities based on the emotion that the people are expressing. This research is conducted using specialised deep learning approaches and state of the art deep learning approaches to classify emotions based on text and comparison of all of these approaches to better identify the best suited approach.

**Research Question:** Comparative study of traditional machine learning and state of the art deep learning approaches for multiclass classifications of emotions in raw text.

**Aim:** Accurately identify a specific emotion associated with raw text to provide assistance to individuals experiencing distress while living their everyday lives.

**Objective:** Comparison of classifier models built using a traditional SVM algorithm, a state of the art general purpose ANN algorithm and a specialised deep learning Bi-LSTM

algorithm. In addition, a comparison will be made using both a TF-IDF and a Word2vec vectorization.

**Hypothesis:** The selection of vectorization techniques will be important. Specialised deep learning approaches will outperform state of the art traditional approaches and if it will be capable on age

### 1.3 Scope

The best three models with different vectorizations were selected for the comparative study for ease of computation as the time frame of this research is limited. The scope of this research is build a model that accurately identifies the emotion associated with text, various models are used for classification and one of them is used in a chatbot through which the chatbot can identify emotion in the conversation and suggest activities.

### 1.4 Limitations

- The amount of data available for emotion classification is limited while deep learning requires more data to train.
- The dataset used for creating an age classifier was created from scratch.
- Due to local infrastructure of the machine, the model performance was limited.

## Chapter 2 – Literature Review

The COVID-19 pandemic is seen to be adversely affecting people all around the world and the noticeable effects have been both physical and mental. People have been seen struggling to cope with the ongoing conditions to keep their mental health sane. A lot of organisations and social groups have come up with various activities to help people with their emotional stress amidst these difficult times. But despite the availability of these platforms there are a bunch of people who refrain from seeking help publicly and may find having a private mode of conversation to be of great help.

This research study has come up with a platform that will aid people to express their feelings privately and get suitable suggestions to cope with their emotions.

The research for multiclass emotion classification is conducted to determine the best suitable approach between the traditional and deep learning approaches that provides satisfactory accuracy and performance metrics. Some of the concepts, models that are vital to this research therefore the following journals and books were referred :

In Ireland, the COVID-19 pandemic has taken a significant toll on the emotional health of the youth rather than the older generation. The 18-34 age group most likely felt anxious (51.2%) than those aged 70+ (13%); discouraged (45.2%) than those aged 70+ (14.5%), lonely (41.5%) than those aged 70+ (17.2%). The home quarantine during the pandemic has disrupted the active lifestyle of youth with respect to exercise, socializing, and academic learning. These studies also show that younger adults feel more mental stress than adults and elderly people. (Social Impact of COVID-19 by Age Group April 2020 - CSO - Central Statistics Office, 2020) Anxiety and depression are generally seen in college students, with upto 74% reporting their first appearance before age 24. (Mody and Mody, 2019). Novel coronavirus has many consequences like economic catastrophe, stress, anxiety and future burden. This has resulted into people sharing their thoughts on social media websites and can indicate and identify their state of mental health in communities. (Guntuku *et al.*, 2020)

(Rajabi, Shehu and Uzuner, 2020) This research makes use of CNN-BiLSTM for emotion classification in short text and gives an accuracy of about 85.1% which in their case outperforms any other deep learning approaches. The researchers compared various deep learning CNN approaches like basic, multifilter, static multifilter with GloVe embeddings, non-static multifilter with GloVe embeddings, multichannel multifilter, multichannel multifilter CNN-BiLSTM approaches.

(Gohil and Patel, 2019) This research utilizes use of supervised learning (Logistic Regression, Multinomial Naïve bayes and LinearSVC) and hybrid learning approaches (using both TF-IDF and SenticNet) along with Tf-Idf and SenticNet is used for primary and secondary feature generation for classification of eight basic emotions namely joy, trust, fear, surprise, sadness, anticipation, anger, disgust. Hybrid approach improves the LinearSVC classifier performance.

(Chawla and Mehrotra, 2018) This research makes use of ensemble methods for multi label classification of text. Ensemble methods include Multinomial naïve bayes, Multiclass SVM with linear kernel, Logistic regression and stochastic gradient descent were used along with

TF-IDF for feature selection, uni-grams and bigrams and 5-fold cross validation for splitting the dataset into test and train set. Logistic regression performed the best among all the algorithms with an F1 score of 0.89. The goal of this study was to analyze various emotions associated and more than 100 emotions were identified.

(Lazemi and Ebrahimpour-Komleh, 2018) This research realizes the use of hybrid deep learning model a combination of CNN and RNN for multi-label classification of eight emotion categories. This hybrid model is compared with Binary Relevance (BR), RANdom k-label (RAKEL) and Multi-label k Nearest Neighbor (MLKNN) algorithms and the results of this study demonstrates that hybrid deep-learning model outperforms all of the above algorithms.

(Karna, Juliet and Joy, 2020) This research does a comparative study of deep learning Long Short Term Memory (LSTM) versus existing machine learning method - Support vector machine for emotion classification of six emotions. LSTM proves to outperform all other learning methods with an accuracy of 94.7%.

(Zhang *et al.*, 2018) This research makes use of Multi-task Convolutional neural network on five public datasets which are SemEval, Fairy Tales, ISEAR, TEC, CBET to classify the problem of multi emotion classification in a single line of text. A multi-task CNN is used for multi emotion prediction with different degrees in a single sentence.

(Gupta *et al.*, 2017) This research makes use of semantic and sentiment-based embeddings with LSTM approach which outperforms traditional and state of the art deep learning approaches. The research has a comparative study of different embeddings like word2vec, Fasttext, GloVe and SSWE using LSTM network. They have also compared various other approaches like NB, SVM, GBDT, CNN SSWE, CNN GLoVe, CNN-NAVA and SS-LSTM. This approach is context aware.

(Namrutha Sridhar, Mrinalini and Vijayalakshmi, 2020) This research makes use of word2vec and word mover's distance embedding with traditional machine learning classifiers – K-nearest neighbor (KNN), extra trees classifier (ETC) and Multilayer perceptron (MLP). ETC outperforms all other classifiers and delivers an accuracy of 95%.

(*Practical machine learning with H2O: powerful, scalable techniques for deep learning and AI | Cook, Darren | download*, no date) This book gives in detail knowledge of how to make the best use of the H2O.ai platform and relieves from hassle of searching through websites like stackoverflow for coding solutions.

(Fei *et al.*, 2020) This research has a comparative study of different deep learning algorithms such as Bi-LSTM and CNN for emotion classification. The researchers used a combination of Bi-LSTM model with self-attention for better emotion classification. This model is called SA-BiLSTM, despite creating this model, it only achieves an accuracy of average about 88%.

(AlBalooshi, Rahmanian and Venkatesh Kumar, 2018) This research utilizes TFIDF with logistic regression, multinomial naïve bayes, xgboost, CNN+BiLSTM, CNN+LSTM (fast text word vector ) for classification of four emotions neutral, joy, sadness and anger and gets an accuracy of 91%, 85%, 85%, 84.9% and 85.7% respectively.

From the above mentioned list of research papers it is evident that most of research studies carried out in the past have used deep learning CNN-LSTM algorithm while other researchers have used traditional machine learning approaches like SVM, KNN, GBDT, NB using TF-IDF and deep learning embeddings. Most of the research conducted was observed to lean more towards experimentation using LSTM approaches. No comparative study between traditional machine learning and specialised deep learning approaches was completed. Further, none of the research presented the H2O.ai state of the art general purpose ANN deep learning algorithm. This research aims to do a comparative study of the state of the art deep learning and traditional machine learning approaches. TF-IDF and Word2vec will be used for vectorization. Previous research also did not look at age, this research will evaluate classifier performance based on age stratification Youth/Adult based on intended use in a conversational agent.

## Chapter 3 - Research Methodology

### 3.1 Introduction

The purpose of this chapter is to examine the research methodology and methods adopted by the researcher to carry out the research. This chapter mainly focuses on analysing the research problem, objectives, hypotheses, the strategy and design, ethical considerations to carry out the research and lastly the research authenticity and research constraints. This research solely matches the work flow of CRISP-DM and hence forms to be the solid base for the research methodology.

### 3.2 CRISP-DM

Cross Industry Standard Process for Data Mining is used in this research. CRISP-DM is an approach which is responsive to the business, tooling, and application. The six phases of CRISP-DM methodology provide the best framework for achieving better and faster results. CRISP-DM is a detailed methodology and process model for data mining that offers a full blueprint for everyone. The figure below demonstrates the phases of the CRISP-DM model.

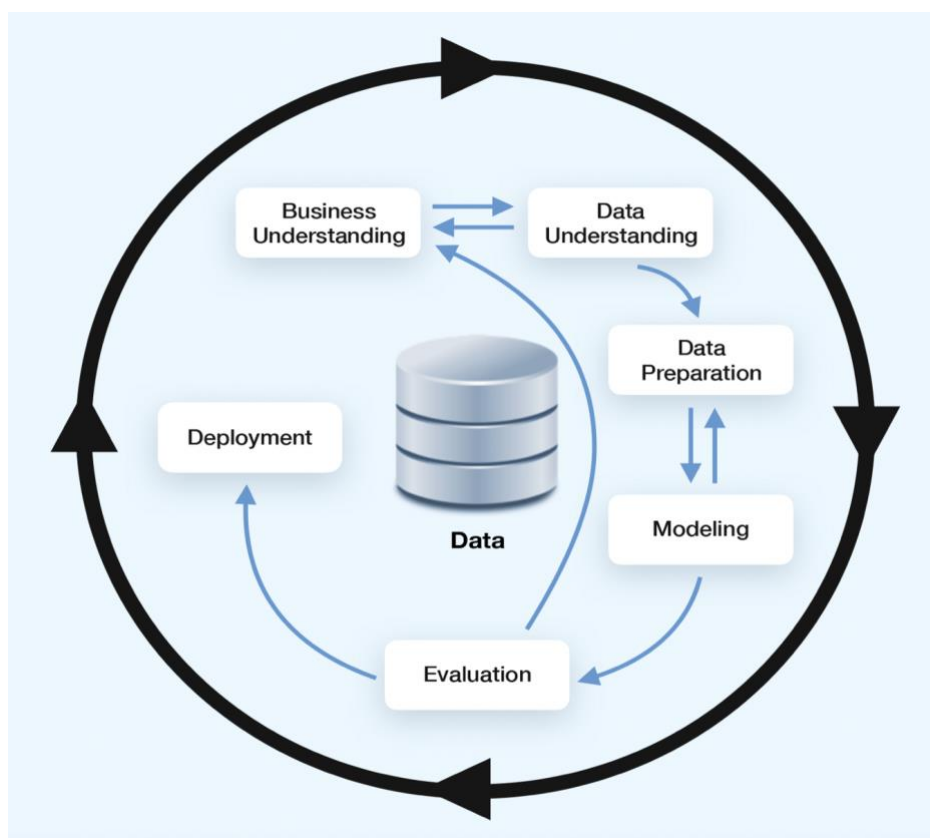


Fig 3.1 CRISP-DM Methodology (Quantum, 2019)

CRISP-DM methodology has six steps:

- 1) Business Understanding
- 2) Data Understanding
- 3) Data Preparation
- 4) Modelling
- 5) Evaluation
- 6) Deployment

The research will be explained in detail based on six phases of the CRISP-DM methodology

## 1. Business Understanding

The first phase of CRISP-DM methodology constitutes of Business Understanding. This step involves understanding of the business objectives, determining business goals, deciding the objectives and producing a project plan. The purpose of this research is to create a model which accurately identifies each human emotion of the six emotions - 'joy', 'sadness', 'surprise', 'anger', 'love' and 'fear' from raw textual data. With the ongoing home quarantine conditions, people are often seen complaining about their mental health and how it is affected to varying levels and it is therefore essential to address their issues by identifying the various emotions that they express on social media. The objective of this research is to perform a comparison of different text pre-processing methods along with Machine learning based and deep learning models to identify the appropriate model that accurately identifies emotion in text. This research also makes a comparative study of using standard ways of implementing models versus using the driverless H2O.ai.

## 2. Data Understanding

The data understanding phase involves data collection, once the appropriate data requirement is fulfilled, the collected data is analysed to find fruitful insights as well as any problems associated with the data are identified. The process of data understanding covers four steps including initial data collection, data description, data exploration and data quality checks.

Emotion classification, a dataset available on Github was used for this research (Kandic, 2020). The dataset illustrates the text and related emotions containing 416810 records with six different emotions such as 'joy', 'sadness', 'anger', 'love', 'surprise', 'fear'. This dataset contains the following columns:

- **Text/tweet:** This column contains the text from social media posts and comments.
- **emotions:** Will accurately reflect one emotion from the above mentioned set of emotions relating to the content in the text column.

There is no data readily available for age classification. In order to perform classification of age based on text, a new dataset is created using tweets from twitter. The efficient way of collecting twitter data is to use twitter scrapper. The twitter scrapper used here is called 'Twint'.

Twint lets anyone scrape tweets without the need of twitter API and uses the twitter search operators to fetch tweets. The biggest advantage of using twint over other scrappers is that it overcomes the limitation of twitter API to scrape more than 3200 tweets a day. This makes data collection faster and easier with just a few lines of code. A combination of multiple tags were used for example 'quarantine' and 'anger', 'quarantine' and 'joy' etc. for all emotions. Multiple csv files were created each having 6500 rows approximately and combined together using cat command in terminal to produce a single file called "data\_emo.csv" and this dataset comprises of 36927 rows and 36 columns.

### 3. Data Preparation

The process of data preparation includes all activities to create the final data to be fed into the model. The five steps in data preparation are data selection, data cleaning and data construction, data integration and data formatting. Various pre-processing steps are required to prepare this dataset suitable for modelling. As this research is a comparative study the pre-processing steps will vary according to each model that is utilized. The initial dataset with 416810 rows and 3 columns are imbalanced with most records for emotion 'joy'. The below figure presents the histogram of the initial dataset. The initial data is highly imbalanced with values in the below table.

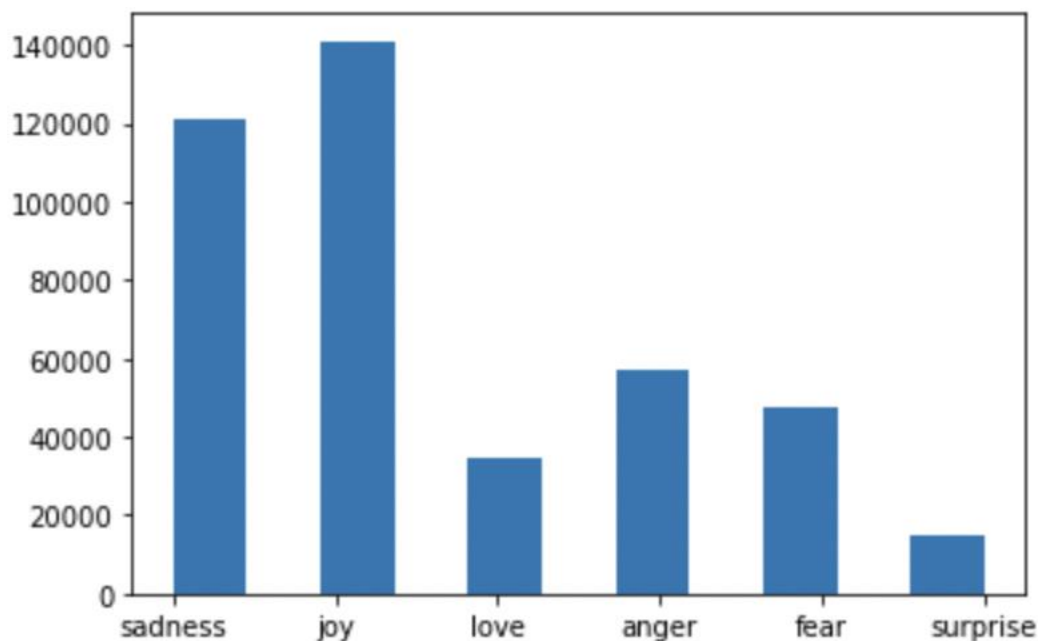


Fig 3.3.1 Histogram for emotions

Emotions	Counts
Sadness	121187
Joy	141067
Love	34554
Anger	57317
Fear	47712
Surprise	14972

Table 3.3.1 Emotion counts

The following basic steps are required in order to prepare data for all models:

To balance the dataset, under-sampling from ‘imblearn’ library is used which reduces the data count according to emotion which has the least number of records, in this case the emotion with least records is ‘surprise’ with 14972 records. (Lemaître, Nogueira and Aridas, 2017) The two columns are split into two arrays and passed onto the un-sampler which results into two arrays with balanced emotions which are then concatenated to a pandas data frame. Hence each emotion in the dataset is dropped to 14972 records and the final dataset comprises of 14972 X 6 emotions which equals to 89832 records. The data can also be over sampled with emotion having the highest records by duplicating the emotions with least records, this will increase the overall size of entire dataset which in turn will affect the training time of the model as well as time for each pre-processing step. The dataset was checked for any NA values but none were found.

The next step for data preparation requires vectorization of the text column so that it can be fed into the model. Before vectorizing, the data is tokenized into words and stop words are removed using the NLTK ‘stopwords’ package. Different vectorization techniques are used and the vectorization techniques are :

- **Term Frequency Inverse Document Frequency (TF-IDF)** – It is used to convert raw document text to TF-IDF features. TF-IDF is a statistical measure used for the evaluation of word relevance to a document. The TF-IDF features are in the form of a sparse matrix of rows x columns. TF-IDF of text data is performed using TfidfVectorizer from scikit-learn. TF-IDF is used with ‘max\_features’ set to 1000 to avoid kernel crash and faster vectorization.

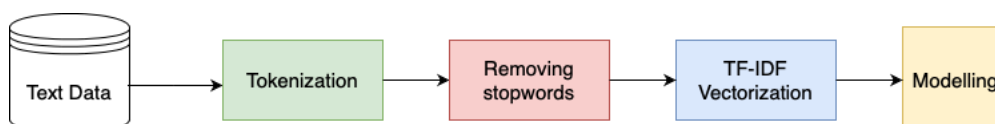


Fig 3.3.2 TF-IDF pre-processing

- H2O Word to vectors (Word2vec)** – The Word2vec algorithm takes text corpus as an input and outputs word vectors. The algorithm learns vector representations of the words after creating a vocabulary from the training text results. Each unique term in the sample corpus is assigned to a corresponding vector in the vector space, which may have hundreds of dimensions. Furthermore, words in the corpus with related contexts are clustered together in the space. Word2vec model is trained on the tokenized text data with two different vector sizes 100 and 500 respectively, the same text data is transformed into vectors using the trained model. The figure 3.3.2 shows an example of emotion “joy” and all the related synonyms are assigned a vector which closely resembles to the emotion ‘joy’ The figure 3.3.3 shows sample of word2vec vectorized words.

```
w2v_model.find_synonyms("joy", count = 10)
OrderedDict([('appreciation', 0.6557760834693909),
             ('surge', 0.6304664015769958),
             ('perseverance', 0.6291489005088806),
             ('peace', 0.6247363090515137),
             ('lightness', 0.6240936517715454),
             ('pang', 0.6164207458496094),
             ('sorrow', 0.6112083196640015),
             ('happiness', 0.6023762822151184),
             ('excitement', 0.5972407460212708),
             ('fulfillment', 0.5965934991836548)])
```

Fig 3.3.3 Word2vec synonyms

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
	0.366039	-0.0324376	-0.00187924	-0.180652	-0.166589	-0.0971331	-0.0350903	-0.0511834	0.0156354	0.0924405
	0.198182	-0.000403768	-0.051575	0.0647123	-0.125064	0.0678918	0.0343621	-0.141687	-0.0109452	0.0786987
	0.215091	0.0555911	-0.00340998	-0.108022	0.0493674	0.0151618	-0.0578741	-0.075961	0.0350991	0.100946
	0.252205	0.0845787	0.16633	-0.0532249	-0.0703063	0.0485581	0.0633434	0.0307089	0.066765	0.0306866
	0.287921	0.0786954	0.00248199	-0.0760914	0.0618797	0.0943083	-0.100776	-0.079598	-0.0875732	0.176648
	0.191358	-0.031261	0.00343239	-0.046088	-0.0350843	0.0701278	-0.0990105	-0.050441	-0.0973092	0.118371
	0.28619	0.110982	0.0194198	-0.014751	-0.0310557	0.0720263	-0.125884	-0.142248	-0.0372277	0.201246
	0.252945	0.0336698	-0.0613053	-0.160871	-0.0323433	0.0151399	-0.135185	-0.0445481	-0.116269	0.164998
	0.243443	0.0999315	-0.042677	-0.184684	0.132208	-0.0750835	-0.0644668	0.0234464	0.0137511	0.206867
	0.10834	0.0695935	0.0835814	-0.162348	-0.0206454	0.0549046	-0.113198	-0.0511006	0.042577	0.0545051

Fig 3.3.4 Word2vec data sample

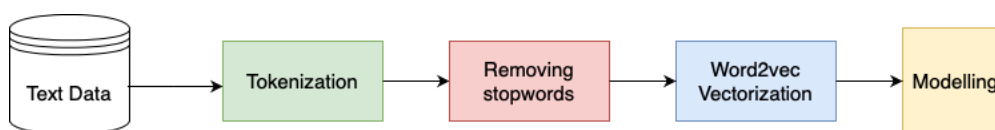


Fig 3.3.5 Word2vec pre-processing

- Scikit-learn text-to-sequences** – It converts text in the text column of dataset to sequence of integers and only the words known by the tokenizer will be taken into account. Using padding and truncating, the texts in the text column are converted to sequences of integers of fixed length and if the record exceeds that fixed length then a few integers are chopped off vice versa if the record is under the fixed length then the

remaining spaces are filled with zeros to maintain the dimension. The data is pre-processed with tokenizer num\_words set to 10000 and oov\_token set to '<UNK>', therefore the best 10000 words will be selected and the remaining words will be replaced by oov\_token. Using the text to sequences from the tokenizer, each text in the dataframe is converted to sequence of integers along with post padding and truncating of sequences with max length as 55. Therefore each line/row in the text column will be replaced by sequences of integers of length 55. Depending on most number of texts with length 55, the max length is set to 55.

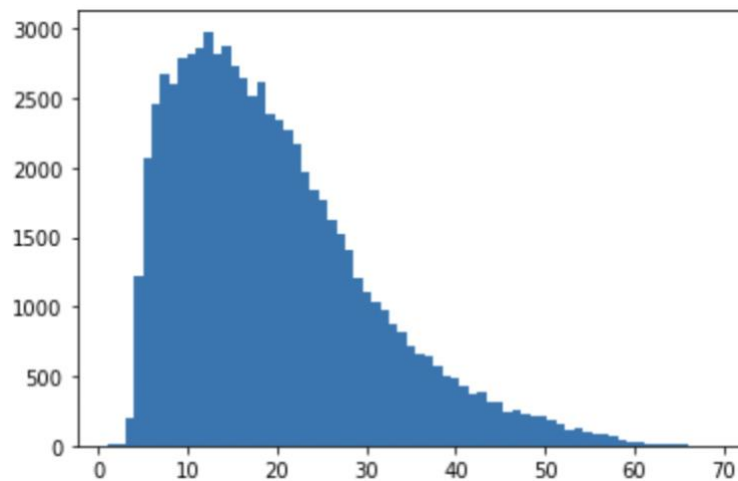


Fig 3.3.6 Histogram for text

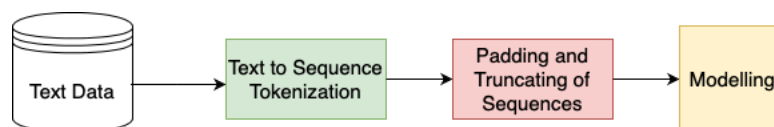


Fig 3.3.7 Text to sequence

- After the data pre – preparation, the prepared data is split into train and test frames using scikit-learn for implementation, whereas for H2O the data is split by using the 'split\_frame' function provided by H2O.

Each emotion in the emotion column is assigned an integer in the target column. Therefore, each emotion is replaced with its corresponding numeric value and ranges from 0 to 5. The target column is required by all the models for training and testing.

	C1	C2	C3	C4
0	i feel sometimes caring sometimes distressed sometimes inadequate sometimes afraid sometimes hurt sometimes ashamed or lonely or left out or sometimes tender exhilarated euphoric delighted	jovial serene perplexed or just downright rotten	love	1
1	i'm supposed to start my new job today and i can't help but feel nervous		fear	5
2	i feel like giving up other times i am so excited that i can bear to be a little more patient		joy	4
3	i suddenly feel shy		fear	5
4	i haven't sleep properly in two days but i'm feeling romantically frustrated tonight		anger	2
5	i've been feeling it lately and since today isn't quite as hot as it has been over the past week i turned on my oven		love	1
6	i should feel pretty good		joy	4
7	i have a feeling my favorites least hated are still my favorites least hated unless they got sent home		anger	2
8	i feel dazed and washed out but the stronger beer is now helping to fill in the void		surprise	0
9	i feel really frustrated when people ask me you're thirty and not married		anger	2

Fig 3.3.8 Label to index

For **age classification**, the data\_emo.csv initially comprised of 36927 rows and 36 columns and out of these 36 columns, the rows containing tweets in languages other than English were dropped using column 'language', only the relevant four columns were selected which added value to this classifier. The four columns are 'id', 'username', 'name', 'tweet'. Further the data was cleaned with the following steps:

- Converting text from uppercase to lowercase
- Removing numbers
- Removing hyperlinks
- Removing hashtags and mentions
- Removing emojis and Unicode
- Replacing special word like '&' with 'and'
- Auto correcting incorrect words that were used

After performing the above steps on the data, the data was saved as 'covi.csv' with columns – id, username, name, tweet. The username column of this dataset was used for age classification using the m3inference. M3 in the m3inference stands for 'multimodal', 'multiattribute' and 'multilingual'. Social media profiles are utilized to infer demographic attributes. M3 is a deep learning based system that works on twitter data by utilizing the username, description and profile picture of an individual to classify the gender as well as the age group that the person belongs to. The below figure demonstrates the working of m3inference:

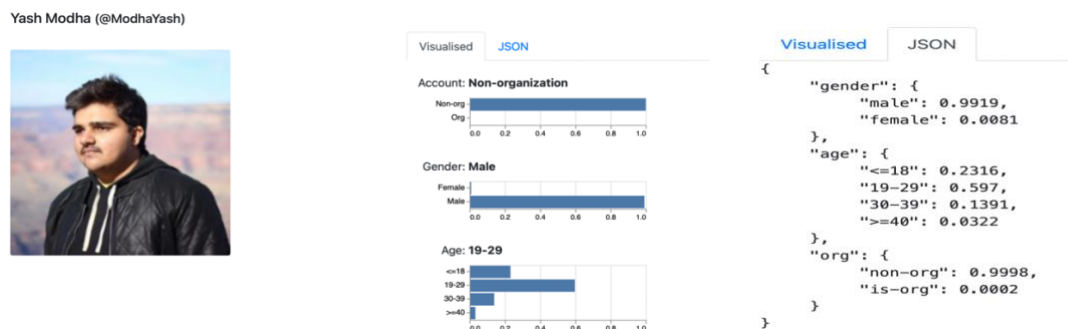


Fig 3.3.9 m3inference example

The United Nations defines age group 15 – 24 as 'Youth' (Nations, no date). M3inference provides an output of age groups as <=18, 19-29, 30-39 and >=40 in a json list file. In order to fulfil the purpose of this research, Youth is categorized as any value which is less than or equal to age 29, while Adult is categorized as any value which is greater than or equal to 30. The value of age groups are compared, if the value for a particular age group is greater in

comparison to the other values mentioned in the group then the person falls under that category. This categorization is performed by calling m3inference API and twitter API is required to use m3inference. The python code for m3inference iterates over usernames from the covi.csv file and creates a separate text file with all the data related to each username. This text file is passed to a python code made with conditional statements that classifies each username with age groups either Youth or Adult depending upon the above condition. This dataset containing the age group is combined with the dataset containing tweets. (Wang *et al.*, 2019)

An age classifier built using the age dataset is deployed on the emotion dataset. Two new datasets from emotion dataset are created one with Age group as Youth and other with Age group as Adult. All of the above pre-processing steps are performed on both of these datasets and the same models are applied on both datasets for emotion classification.

#### 4. Modelling

Distinct models are implemented in the modelling phase depending on comprehensive research. Individual models have individual pre-processing steps required to fit the data. The modelling phase includes pre-processing steps along with models. This stage comprises of model selection, model creation and model assessment.

The best models implemented for this emotion classification problem are:

##### I. Bidirectional Long Short Term Memory (Bi-LSTM)

LSTM also called as Long Short Term Memory's history dates back to 1995-1997. Sepp Hochreiter and Jürgen Schmidhuber proposed LSTM. LSTM was developed to solve the vanishing gradient descent problem. (Hochreiter and Schmidhuber, 1997)

In the field of deep learning, LSTM is based on artificial recurrent neural network architecture. LSTM is a supervised learning algorithm where training data is required for a model to learn. LSTM is able to process whole sequences of data and has feedback connections meaning it works on present input by taking into consideration the previous output also called as feedback and stores into the memory for small duration thus justifying why it is called Long short term memory. The basic unit of LSTM comprises of a cell, 'input gate', 'output gate' and 'forget gate'. Over a random time period the cell remembers the value while the three gates control the circulation of data in and out of the cell. The figure below shows the basic LSTM cell with cell and three gates along with input modulation gate. The arrows depict the recursive nature of the cell, and this allows information from previous intervals to be stored within LSTM cell. The input gate is called the save vector, the input gate decides if the information is allowed to enter the cell state. The forget gate is called the remember vector, it decides which information to forget by the cell state and also controls the amount of information from previous state. It multiplies the information to forget by multiplying it by 0. The output gate is also called as focus vector, it decides which values are eligible to be moved to the next hidden state. (Kang, 2017)

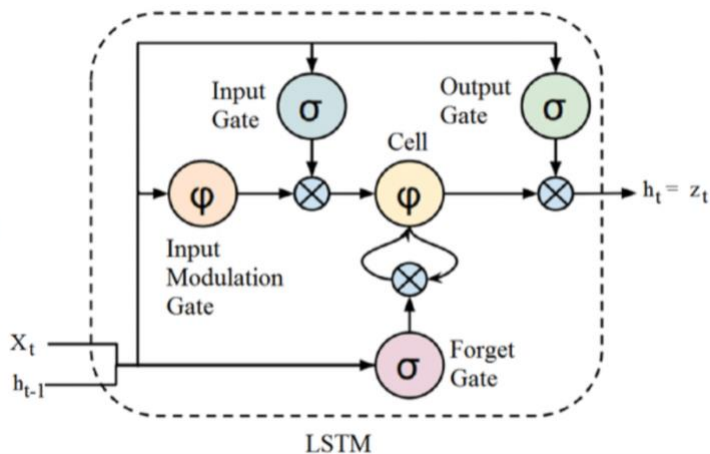


Fig 3.4.1 Basic LSTM cell(Kang, 2017)

The hidden state is also called as working memory, it decides information that should be taken to the next sequence. Below figure displays the hidden state.

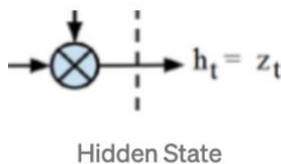
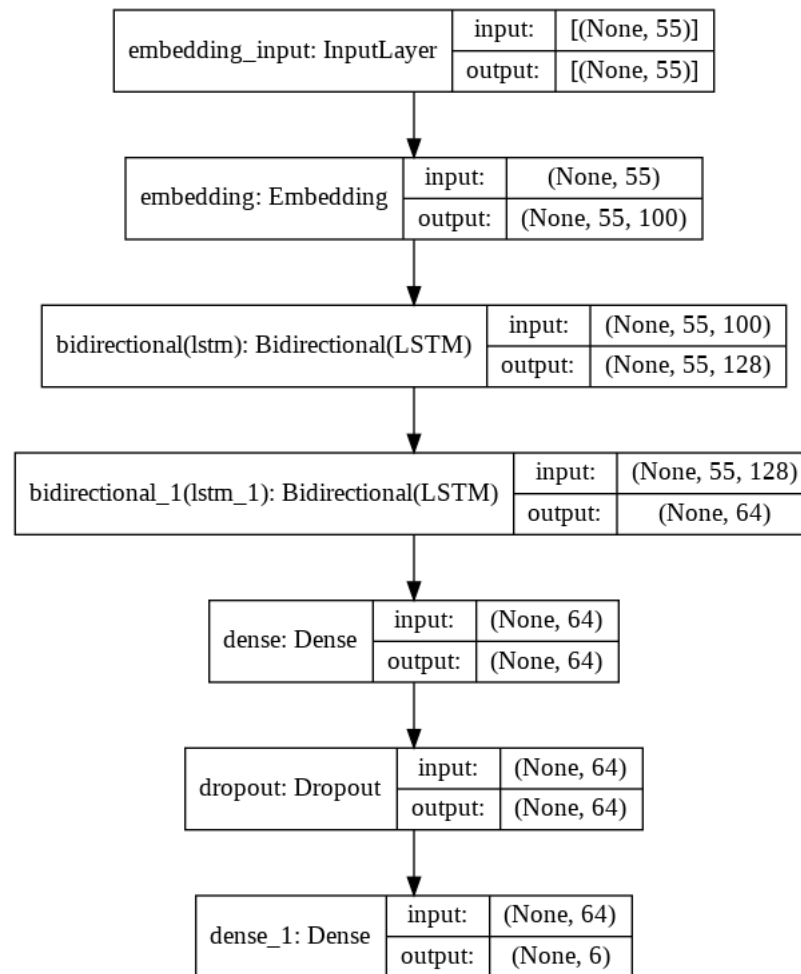


Fig 3.4.2 Hidden state(Kang, 2017)

The advantage of using LSTM over other model is that LSTM remembers the previous state and was brought into the picture to solve the vanishing gradient descent problem. LSTM provides large range of parameters and are able to model long term sequence dependencies.

Tensorflow keras Sequential Bi-directional LSTM model is implemented using python on the dataset. The dataset is pre-processed using the tokenizer, Scikit-learn text-to-sequences along with padding and truncating of sequences to a fixed length of sequences. The figure below shows the summary of the model used:



*Fig 3.4.3 Sequential Model Summary*

The saved Bi-LSTM model is used for making emotion predictions on age group (Youth/Adult) stratified emotion dataset.

## II. Support Vector Machine (SVM)

In 1963, Vladimir N. Vapnik and Alexey Ya. Chervonenkis invented the Support Vector Machine algorithm. (Cortes and Vapnik, 1995) In 1992 SVM classifier was derived using Support Vector Machine Algorithm. SVM are supervised machine learning algorithms that are used for classification and regression analysis problems.

A SVM constructs a hyperplane in an infinite multidimensional space with datapoints and the goal of the algorithm is to fit the hyperplane in such a way that it can distinctly classify the data points. Data points located on the respective sides of the hyperplane can be associated with distinct classes. If the number of features is less than 3 then the hyperplane is a single line, as opposite to this if the features is greater than or equal to 3 then hyperplane is a 2D plane. (Gandhi, 2018)

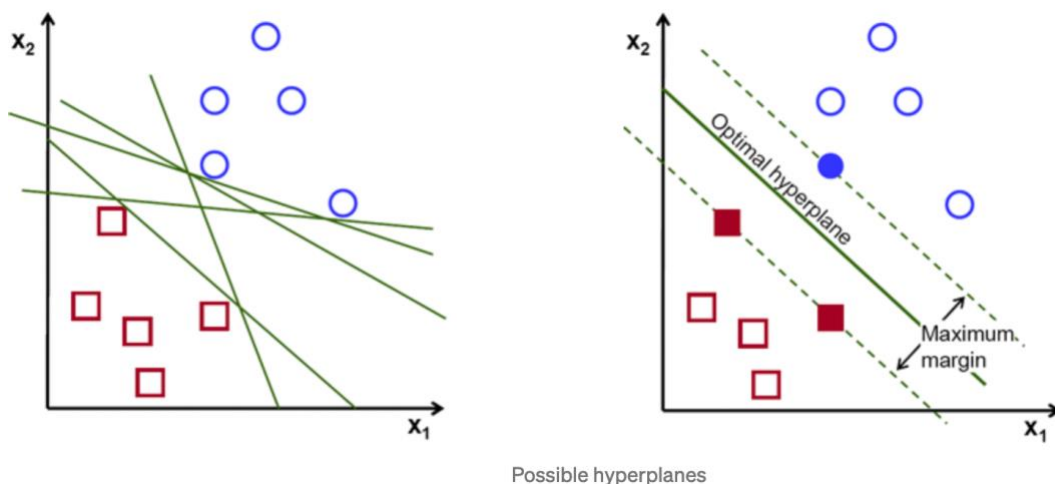


Fig 3.4.4 SVM possible hyperplanes (Gandhi, 2018)

Above diagram shows comparison of different hyperplanes, the optimal hyperplane is the one which has maximum margin that is maximum distance between datapoints of classes. To achieve maximum margin, the hinge loss function is utilized. The hinge loss function is –

$$C(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases} \quad c(x, y, f(x)) = (1 - y * f(x)) +$$

Fig 3.4.5 Hinge Loss Function (Gandhi, 2018)

The estimated value and the existing value are associated with identical sign if the amount is 0, if that is not the case then the loss value is measured. In addition to this a regularization parameter is accreted to the cost parameter. The regularization parameter guides the SVM optimization about the amount of misclassification that can be avoided with respect to each training example. The primary motive of the regularization parameter is to stabilise the margin maximization and loss. After adding the regularization parameter to the hinge loss function, the function looks like-

$$\min_w \lambda ||w||^2 + \sum_{i=1}^n (1 - y_i \langle x_i, w \rangle) +$$

Fig 3.4.6 Regularization parameter Loss Function (Gandhi, 2018)

To find the gradients in relation to the weights, partial derivatives are chosen. Therefore, the gradients are utilized to modify weights. The gradients are highlighted in the below figure-

$$\frac{\delta}{\delta w_k} \lambda ||w||^2 = 2\lambda w_k$$

$$\frac{\delta}{\delta w_k} (1 - y_i \langle x_i, w \rangle) = \begin{cases} 0, & \text{if } y_i \langle x_i, w \rangle \geq 1 \\ -y_i x_{ik}, & \text{else} \end{cases}$$

Fig 3.4.7 Gradients (Gandhi, 2018)

In the absence of misclassification, the gradient needs to be revamped using the regularization parameter. The gradient revamp when there is no misclassification is depicted in the below figure-

$$\omega = \omega - \alpha \cdot (2\lambda\omega)$$

Fig 3.4.8 Gradient – No misclassification (Gandhi, 2018)

As opposite to the above mentioned scenario – in presence of misclassification, the gradient needs to be revamped using the loss as well as regularization parameter. The gradient revamp when there is misclassification is listed in the below figure – (Gandhi, 2018)

$$\omega = \omega + \alpha \cdot (Y_i \cdot X_i - 2\lambda\omega)$$

Fig 3.4.9 Gradient – Misclassification (Gandhi, 2018)

The advantages of using SVM are:

- it does not overfit.
- SVM works well on semi structured and unstructured data like text.
- SVM scales well to high dimensional data.
- SVM models practise generalization therefore risk of overfitting is less.

Bigram TF-IDF vectorization with max\_features set to 1000 and tokenizer is defined to remove stopwords from the text is used with LinearSVC from scikit-learn that is Linear Support Vector Classifier. The LinearSVC uses the linear kernel that provides low bias and one which better suits for larger dataset. Python is used for implementation.

Word2vec with vector size of 100 with labels as text is used with SVM in rapid miner.

The saved LinearSVC model is used for making emotion predictions on age group (Youth/Adult) stratified emotion dataset.

### III. Deep Learning Artificial Neural Network using H2O

H2O.ai was found in 2012 by Cliff Click and Sri Satish Ambati. (*H2O.ai - Crunchbase Company Profile & Funding*, no date) H2O.ai is fully open source and distributed in-memory machine learning and AI platform. It is based on java programming language. The most extensively used statistical and machine learning algorithms such as GBM, XGB, DL, Word2vec etc are supported by H2O.

Deep learning in H2O.ai is based on feedforward artificial neural network and is trained with stochastic gradient descent using back propagation. The baseline deep learning model contains various hidden layers comprising of neurons with different activation functions.

In a feedforward artificial neural network, the information flows only in one direction that is forward from input layer to hidden layer to output layer. Each layer in artificial neural network comprises of collection of neurons. An ANN resembles the working of a human brain. In the diagram below, each arrow represents the flow of information from one neuron to another. The

signal between each neuron is controlled by an integer value called as weight. The ANN learns from itself by adjusting the weights according to the quality of output it predicts.

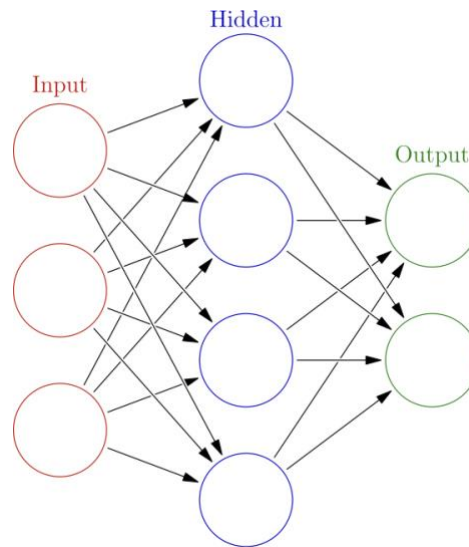


Fig 3.4.10 Feedforward ANN

The back propagation is a training algorithm in which, when a output is passed it adjusts its weight accordingly to produce the preferred output.

The predominant advantage of using ANN is that ANN has the ability to learn and model complex relationships between input and output data. ANN is able to generalize and can derive relationships on unseen data.(Mahanta, 2017)

Two different word2vec vectorizations 100 vector size and 500 vector size are used for comparison. H2ODeeplearningestimator from H2O.estimators is implemented using python. The target variable is set to emotions. For the multi emotion classification, H2ODeeplearningestimator requires a few parameters like distribution which specifies the type of problem (Bernoulli for binary class or multinomial for multiclass), auc\_type which takes macro, weighted and micro as input.

The base H2ODeepLearningestimator is implemented with parameters distribution set to “multinomial” for multiclass classification and auc\_type as “MACRO\_OVR”. The above model is implemented with scikit-learn’s TF-IDF vectorizer with vector size of 1000.

The saved H2ODeepLearningestimator model along with the Word2vec 100 vec size and 500 vector size pre-processing steps is used for making emotion predictions on age group (Youth/Adult)stratified emotion dataset respectively.

#### IV. Rapid Miner

Rapidminer is a tool that provides an environment for data preparation, machine learning, deep learning, text mining, and predictive analytics using a GUI. The Rapid miner, earlier known as

YALE (Yet Another Learning environment) was developed in the year 2001 by Ralf Klinkenberg, Ingo Mierswa, and Simon Fischer at the Artificial Intelligence Unit of the Technical University of Dortmund. In 2006, a company named Rapid-I established by Ingo Mierswa and Ralf Klinkenberg backed up the YALE. Therefore, in the year 2013 Rapid-I was rebranded to RapidMiner ('German Predictive Analytics Startup Rapid-I Rebrands As RapidMiner, Takes \$5M From Open Ocean, Earlybird To Tackle The U.S. Market', no date). Rapidminer is built using java programming language. (*Interview with RapidMiner's Ingo Mierswa, Ralf Klinkenberg, part 1*, no date)

The advantages of using RapidMiner is that it provides a graphical user interface for its process and avoids the need to code which prevents errors. Each step is a process and each process is carried out with the use of different operators. The elimination to write code makes it suitable for all kinds of learners and educators.

Similar to the python implementation of H2O, the Deep Learning model used in RapidMiner uses the H2O, the dataset implemented here includes the Word2vec having vector size of 100 and the target column with all six emotions as text.

## V. Gradient Boosting Machine (GBM)

GBM - Gradient Boosting Machine is a technique of altering the weak learners into strong learners. The introduction of the AdaBoost algorithm makes it simpler to describe the gradient boosting algorithm. The function of the AdaBoost algorithm commences by guiding a decision tree wherein an identical weight is allotted to every single observation. The estimation of the first tree is then followed by the process of rising the weights of those observations found to be challenging to classify and instead drop the weights of those that are simpler to classify. The second tree is

developed thereafter on the weighted data. Gradient Boosting instructs various models in a steady and chronological manner. The method by which the two algorithms recognize the limitations of the weak learners indicates the distinction between AdaBoost and Gradient Boosting Algorithm. Gradient boosting algorithm recognizes shortcomings by using gradients in loss function. A loss function is a metric that illustrates a model's quality of fitting on the underlying data. The major advantage of using gradient boosting is that it permits the utilization of user specified cost function. (Singh, 2018)

Gradient Boosted Machine is implemented in H2O using the H2OGradientBoostingEstimator with Word2vec and TF-IDF vectorizations for age classification.

Gradient Boosting Machine with Word2vec 100 vector size is used for classifying age groups (Youth/Age) to create Youth emotion data and Adult emotion data.

## 5. Evaluation

Evaluation phase is carried out to assess the design of the model as well as if it fits the business goals. The models are deployed once their models results are evaluated.

Cross validation approach is used here to evaluate the performance of the machine learning model. The vectorized dataset is split using `train_test_split` from the scikit learn using the 80:20 ratio. The train size is set to 80 and test size to 20. Similarly for H2O, the dataset is split using `split_frame` from h2o. The training frame is used to train the model while the testing frame is used for predictions and performance evaluation. The accuracy of the model is calculated by mean of predicted and true scores. While the recall scores for each emotion is calculated using the formula:  $\frac{\text{true positive}}{(\text{true positive} + \text{false negative})}$ .

The above cross validation techniques are used for age dataset and the age classifier applied on emotion dataset is split into Youth and Adult dataset respectively.

The confusion matrix is plotted for each model to check the performance of prediction.

## 6. Deployment

The deployment phase includes deployment of a suitable model which fulfils the business goals efficiently. The gained knowledge is organized in a manner that a business is able to use it. Depending on the business demand the deployment can be a report or an application of the model in a process.

For this research, the deployment phase is application of emotion classification model for a chatbot so that it can identify emotions based on raw text and give replies. The H2O deep learning model is implemented in this chatbot which takes string as input, splits it into words, applies the word2vec model on the text to convert it into vectors and passes it to the deep learning model for prediction of emotion. The model displays a number which is the integer for emotion. The integer is converted to emotion using conditional statements. Each emotion is assigned to a list of replies and when a emotion is identified by the model, the chatbot gives random reply from the list.

### Chatbot

A Flask POST API is created using H2O deep learning artificial neural network model and a word to vector model with vector size 500. A chatbot app created in Swift programming language is linked to the Flask API using localhost. Swift is a programming language created by Apple for its macOS, iOS, watchOS etc. devices. The app sends a message by requesting the Flask POST API and after the model is ready with the prediction and the reply, it gives response back to the app. The below figure illustrates the working of chatbot app.

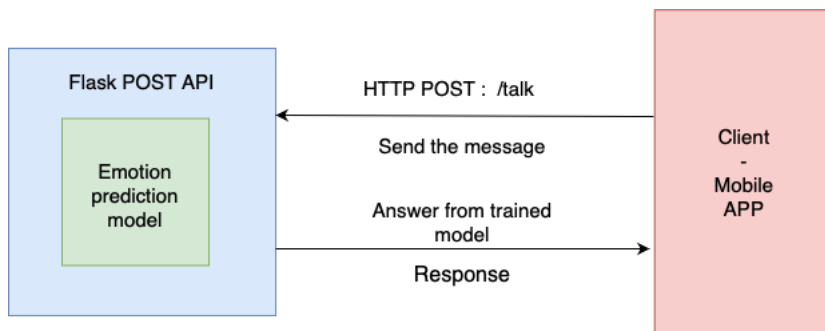


Fig 3.6.1 Flask API

The below figures demonstrate the chatbot application :

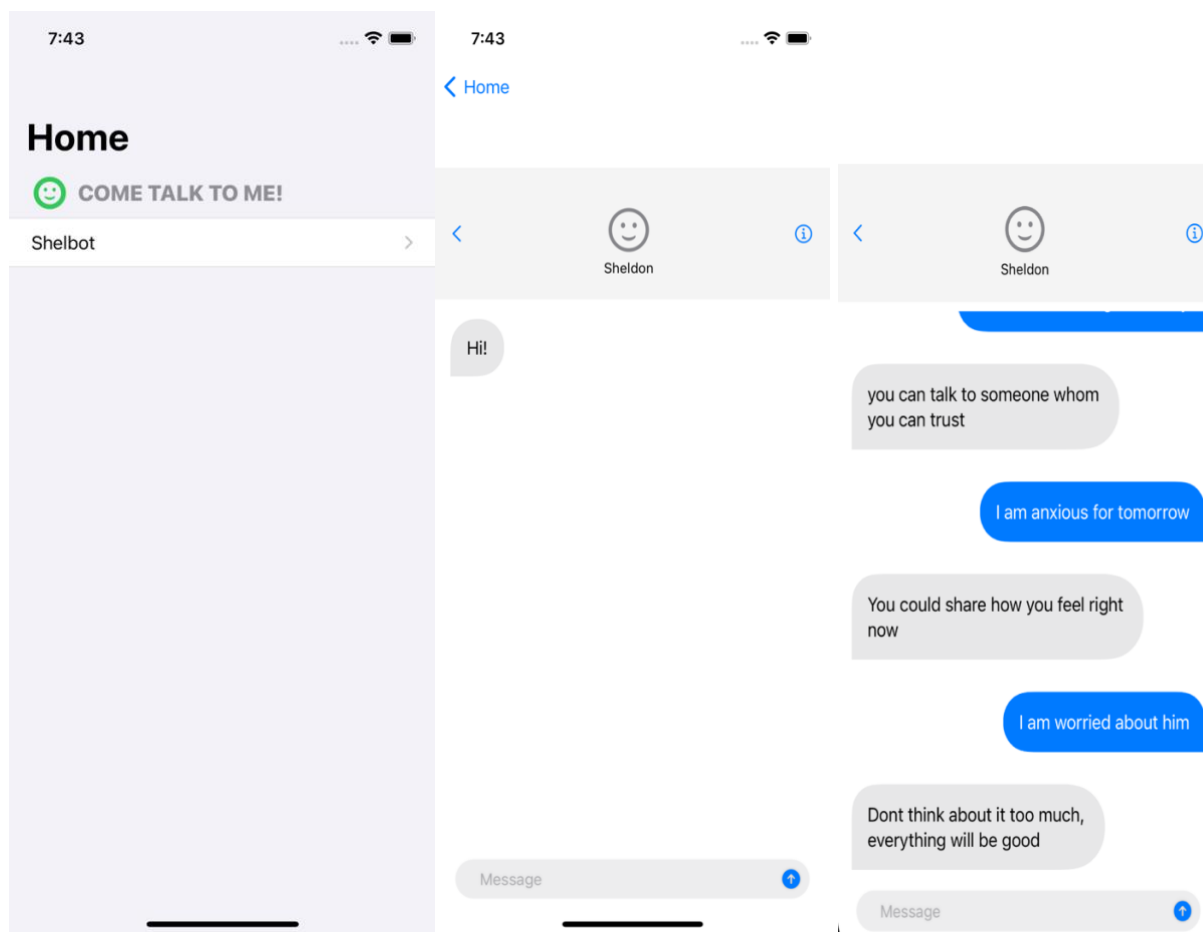


Fig 3.6.2 'Shelbot' Emotion Chatbot

When a person sends a message, the bot gives replies based on predefined messages from a list. Each emotion is assigned to a list of messages which when an emotion is identified by the model the reply is taken randomly from that particular emotion list.

## Chapter 4 – Discussion

### 5.1 Data Overview

Data Overview comprises of summary of data. Each model requires dataset that is tailored specifically to needs of the model as well as the tool behind it, Therefore, different datasets are derived from the balanced dataset. The differences are in terms of column names, the type of target column and the each emotion with its particular integer. Thus, for each distinct model different data pre-processing steps are required to be followed and there is no generalization of data.

### 5.2 Model Results and Discussion

The following are the model results:

#### I. Bidirectional – Long Short Term Memory

The figure below represents the layers of Bi-LSTM model:

```

Model: "sequential"
Layer (type)                Output Shape                Param #
=====
embedding (Embedding)       (None, 55, 100)            1000000
bidirectional (Bidirectional (None, 55, 128)            84480
bidirectional_1 (Bidirection (None, 64)                41216
dense (Dense)                (None, 64)                 4160
dropout (Dropout)           (None, 64)                 0
dense_1 (Dense)             (None, 6)                  390
=====
Total params: 1,130,246
Trainable params: 1,130,246
Non-trainable params: 0
  
```

*Fig 4.5.1 LSTM model summary*

The table below represents the confusion matrix:

anger	2918	8	1	10	3	124
sadness	73	2815	3	2	1	66
surprise	0	1	2972	0	1	1
joy	15	1	26	2786	231	5
love	2	0	0	3	2956	2
fear	20	4	239	5	1	2672
	anger	sadness	surprise	joy	love	fear

*Table 4.5.1 LSTM Confusion matrix*

The Training Accuracy of the Bi-LSTM model is 95.09% and testing accuracy is 95.28%.

- Bi-LSTM Emotion Age (Youth/Adult)

The saved Bi-LSTM model with the same pre-processing steps and saved tokenizer is deployed on the age group stratified youth emotion and adult emotion dataset. The below given table represents the accuracy scores for each dataset:

No.	Vectorization	Models	Youth emotion data accuracy	Adult emotion data accuracy
1	Text-to-Sequences	Bi-LSTM	95.698%	94.907%

*Table 4.5.2 Bi-LSTM Emotion Age*

## II. Support Vector Machine

- TF-IDF LinearSVC

Below table represents the confusion matrix for LinearSVC with TF-IDF vectorization which achieved training accuracy of 97.87% and testing accuracy of 91.5%

surprise	2748	14	67	29	76	26
sadness	8	2750	189	8	2	18
joy	30	245	2546	14	93	13
anger	6	7	3	2856	4	87
fear	52	13	107	22	2848	22
love	36	45	12	258	19	2694
	surprise	sadness	joy	anger	fear	love

*Table 4.5.3 TF-IDF LinearSVC Confusion matrix*

- Word2vec – Rapid Miner – SVM

The dataset used here is Word2vec with 100 vector size dataset. This dataset is made using the H2O word2vec model trained and implemented on the text part of the dataset with a concatenation of emotions as text. This is the same dataset used for the Deep Learning model. The following are the results achieved from the RapidMiner SVM model:

The table below represents the performance metrics of the RapidMiner SVM model

No.	Metrics	Value
1	Accuracy	66.13%

*Table 4.5.4 Word2vec Rapid Miner SVM performance*

Confusion Matrix:

	true love	true fear	true joy	true anger	true surprise	true sadness
pred. love	2052	70	336	121	112	88
pred. fear	104	1980	99	299	323	274
pred. joy	529	216	2212	231	360	293
pred. anger	77	119	43	1512	26	185
pred. surprise	46	207	72	58	2012	43
pred. sadness	186	402	232	773	161	2111
class recall	68.54%	66.13%	73.88%	50.50%	67.20%	70.51%

*Table 4.5.5 Word2vec Rapid Miner SVM Confusion matrix*

- TF-IDF – LinearSVC – Emotion Age (Youth/Adult)

The saved LinearSVC model is deployed on the Youth and Adult stratified emotion data, the below given table represents the accuracy scores for each age groups:

No.	Vectorization	Model	Youth emotion data accuracy	Adult emotion data accuracy
1	TF-IDF	LinearSVC	96.593%	96.964%

*Table 4.5.6 TF-IDF LinearSVC Emotion Age*

### III. H2O Deep Learning

- Word2vec – H2O Deep Learning

The Word2vec pre-processing is sub-divided into two datasets of different vector sizes 100 and 500, the H2O Deep learning model is implemented for different vector size datasets. Below figure demonstrates the workflow:

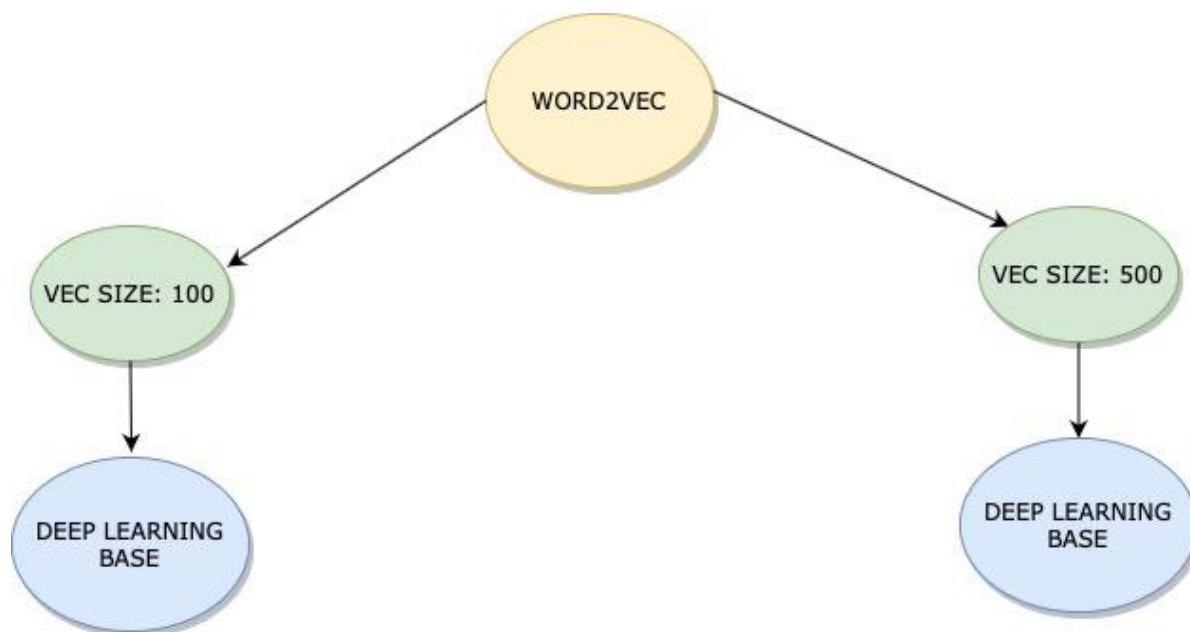


Fig 4.5.4 Word2vec H2O workflow

Performance metrics for 100 vector size word2vec:

Deep learning base model with parameter distribution set to 'multinomial' and auc\_type to 'WEIGHTED\_OVR'

No.	Metrics	Value
1	Accuracy	75.548%
2	MSE	0.203654
3	RMSE	0.451281
4	Mean Per-Class Error	0.244806
5	logloss	0.668183
6	AUC	0.956324
7	AUCPR	0.852823

Table 4.5.7 Word2vec 100 H2O Deep Learning performance

Confusion Matrix:

	surprise	love	anger	sadness	joy	fear
surprise	2542	44	61	101	114	143
love	71	2389	82	83	344	62
anger	45	56	2319	310	134	144
sadness	53	66	246	2281	206	188
joy	126	307	124	225	2094	92
fear	266	34	223	352	106	1994

Table 4.5.8 Word2vec 100 H2O Deep Learning Confusion matrix

Performance metrics for 500 vector size word2vec:

Deep learning base model with parameter distribution set to ‘multinomial’ and auc\_type to ‘WEIGHTED\_OVR

No.	Metrics	Value
1	Accuracy	80.50%
2	MSE	0.158106
3	RMSE	0.397626
4	Mean Per-Class Error	0.195763
5	logloss	0.63769124
6	AUC	0.971825
7	AUCPR	0.899324

Table 4.5.9 Word2vec 500 H2O Deep Learning performance

Confusion matrix:

	surprise	love	anger	sadness	joy	fear
surprise	2655	22	18	56	69	207
love	18	2787	18	70	167	40
anger	28	91	2243	329	121	208
sadness	36	68	170	2357	140	230
joy	67	343	47	229	2159	118
fear	237	26	105	195	46	2326

Table 4.5.10 Word2vec 500 H2O Deep Learning Confusion matrix

The rise in the vector size from 100 to 500 improves the accuracy score by 4.952%

- TF-IDF – H2O Deep Learning

The H2O deep learning model uses the TF-IDF as pre-processing step with max features set to 1000 and a tokenizer with removal of stopwords using nltk stopwords. The H2O deep learning model has hyperparameters as distribution set to ‘multinomial’ and auc\_type set to ‘MACRO\_OVR’. The table below shows the model performance metrics:

No.	Metrics	Value
1	Accuracy	86.41%
2	MSE	0.1149624
3	RMSE	0.3390610
4	Mean Per-Class Error	0.1356313
5	AUC	0.9809000
6	AUCPR	0.9331939

Table 4.5.11 TF-IDF H2O Deep Learning performance

The below table represents the confusion matrix

fear	2446	62	83	239	99	12
sadness	109	2494	221	14	100	22
joy	65	143	2486	55	63	252
surprise	70	33	57	2792	13	10
anger	126	134	161	18	2605	20
love	17	26	201	7	10	2702
	fear	sadness	joy	surprise	anger	love

Table 4.5.12 TF-IDF H2O Deep Learning Confusion matrix

- Word2vec – Rapid Miner – Deep Learning

The dataset used here is Word2vec with 100 vector size dataset. This dataset is made using the H2O word2vec model trained and implemented on the text part of the dataset with a concatenation of emotions as text.

Confusion Matrix:

	true love	true fear	true joy	true anger	true surprise	true sadness
pred. love	1946	28	274	78	64	64
pred. fear	150	2282	204	398	596	484
pred. joy	662	210	2277	227	523	398
pred. anger	147	164	87	1986	55	353
pred. surprise	14	126	33	15	1689	17
pred. sadness	75	184	119	290	67	1678
class recall	65.00%	76.22%	76.05%	66.33%	56.41%	56.05%

Table 4.5.13 Word2vec Rapid Miner Deep Learning Confusion matrix

The below table shows the RapidMiner Deep Learning model results in tabular form:

No.	Metrics	Value
1	Accuracy	66.01%

Table 4.5.14 Word2vec Rapid Miner Deep Learning performance

- TF-IDF – Deep learning H2O – Age

The dataset used here is balance\_01.csv, after vectorization the dataset is passed through H2O Deep Learning estimator with distribution set to “Bernoulli” for binary class classification

and auc\_type to “MACRO\_OVR”. The class 0 is Adult and class 1 is Youth. The following results were achieved:

	true.Adult	true.Youth
pred.Adult	816	3103
pred.Youth	287	3636
Total	1103	6739

Table 4.5.15 TF-IDF Deep Learning H2O Age Confusion matrix

The below table represents the performance metrics:

No.	Metrics	Value
1	Accuracy	56.835%
2	AUC	0.6734521
3	MSE	0.2411777
4	RMSE	0.4910985
5	AUCPR	0.6552955
6	logloss	0.7037739
7	mean_per_class_error	0.3728848
8	Gini	0.34690430

Table 4.5.16 TF-IDF Deep Learning H2O Age performance

- Word2vec – Deep H2O – Age

The dataset used here is balance\_01.csv, after Word2vec vectorization the dataset is passed through H2ODeeplearningestimator with distribution set to “Bernoulli” for binary class classification and auc\_type to “MACRO\_OVR”. The class 0 is Adult and class 1 is Youth. The following results were achieved:

	true.Adult	true.Youth
pred.Adult	1702	2272
pred.Youth	658	3179
Total	2360	5451

Table 4.5.17 Word2vec Deep Learning H2O Age Confusion matrix

The below table presents the performance metrics:

No.	Metrics	Value
1	Accuracy	62.553%
2	AUC	0.710186
3	MSE	0.219597
4	RMSE	0.468612
5	AUCPR	0.700345
6	logloss	0.635406
7	mean_per_class_error	0.338897
8	Gini	0.420373

Table 4.5.18 Word2vec Deep Learning H2O Age performance

- Word2vec – Deep learning H2O – Emotion Age (Youth/Adult)

The dataset used here is youth.csv and adult.csv along with saved word2vec 100 vector model as well as word2vec 500 vector model and deep learning model trained on 100 vector as well as deep learning model trained on 500 vectors. The below table represents the accuracy score of H2O deep learning emotion classifier models on youth and adult data:

No.	Vectorization	Model	Youth emotion data accuracy	Adult emotion data accuracy
1	Word2vec – 100 vector	H2O Deep Learning	81.974%	74.437%
2	Word2vec – 500 vector	H2O Deep Learning	86.465%	78.893%

Table 4.5.19 Word2vec Deep Learning H2O Emotion Age performance

#### IV. Gradient Boosting Machine H2O

- TF-IDF – GBM H2O – Age

The dataset used here is balance\_01.csv, after TF-IDF vectorization the dataset is passed through H2O Gradient Boosting Estimator with distribution set to “Bernoulli” for binary class classification and auc\_type to “MACRO\_OVR”. The class 0 is Adult and class 1 is Youth. The following results were achieved:

	true.Adult	true.Youth
pred.Adult	1308	2611
pred.Youth	452	3471
Total	1760	6082

Table 4.5.20 TF-IDF GBM H2O Age Confusion matrix

The below table represents the performance metrics:

No.	Metrics	Value
1	Accuracy	60.941%
2	AUC	0.7052805
3	MSE	0.2225089
4	RMSE	0.4717085
5	AUCPR	0.7052805
6	logloss	0.6364197
7	mean_per_class_error	0.3384232
8	Gini	0.4105611

Table 4.5.21 TF-IDF GBM H2O Age performance

- Word2vec – GBM H2O – Age

The dataset used here is balance\_01.csv, after Word2vec vectorization the dataset is passed through H2OGradientboostingestimator with distribution set to “Bernoulli” for binary class classification and auc\_type to “MACRO\_OVR”. The class 0 is Adult and class 1 is Youth. The following results were achieved:

	true.Adult	true.Youth
pred.Adult	1908	2066
pred.Youth	672	3165
Total	2580	5231

Table 4.5.22 Word2vec GBM H2O Age Confusion matrix

No.	Metrics	Value
1	Accuracy	64.960%
2	AUC	0.7303463
3	MSE	0.2101872
4	RMSE	0.4584618
5	AUCPR	0.7116886
6	logloss	0.6091689
7	mean_per_class_error	0.3206857
8	Gini	0.4606926

Table 4.5.23 Word2vec GBM H2O Age performance

This model is deployed on the emotion dataset with 89832 records, the below table represents the results:

Age Group	Count
Adult	2402
Youth	87790

Table 4.5.24 Word2vec GBM H2O Age counts

From 89832, 2402 records were identified as Adults and 87790 records were identified as Youth.

### 5.3 Results

The below given tables represent the comparison of different metrics and different models for different datasets respectively.

#### Comparison of approaches

- Comparison of accuracy scores for emotion classification

No.	Vectorization	Models	Train - Accuracy	Test- Accuracy
1	Text-to-Sequences	Bi-LSTM	95.09%	95.28%
2	TF-IDF	LinearSVC	97.87%	91.5%
3	TF-IDF	H2O Deep Learning	89.81%	86.41%
4	Word2vec – 100 vector	H2O Deep Learning	83.373%	75.548%
5	Word2vec – 500 vector	H2O Deep Learning	87.750%	80.50%
6	Word2vec – 100 vector	Rapid Miner Deep Learning	67.58%	66.01%
7	Word2vec – 100 vector	Rapid Miner SVM	66.9%	66.13%

Table 4.5.25 Comparison of accuracy scores for emotion classification

The above table represents comparison of accuracy scores of each model with different vectorizations. The Bi-LSTM model outperforms all other models and provides the best accuracy score. The TF-IDF vectorized data used with the model outperforms the Word2vec vectorized data with model.

- Comparison of recall scores for emotion classification

No.	Vectorization with models	Surprise	Love	Anger	Sadness	Joy	Fear
1	T2S – Bi-LSTM	91.70%	92.57%	93.36%	99.50%	99.28%	93.10%
2	TF-IDF – LinearSVC	95.35%	94.19%	89.61%	89.46%	87.07%	93.62%
3	TF-IDF – H2O – DL	77.47%	89.53%	89.34%	86.34%	86.24%	90.14%
4	Word2vec – 100 vec – H2O – DL	81.92%	82.49%	75.90%	68.05%	69.85%	76.02%
5	Word2vec – 500 vec – H2O – DL	87.31%	83.52%	86.24%	72.84%	79.90%	74.34%
6	Word2vec – 100 vec – RM – DL	56.41%	65.00%	66.33%	56.05%	76.05%	76.22%
7	Word2vec – 100 vec – RM – SVM	67.20%	68.54%	50.50%	70.51%	73.88%	66.13%

Table 4.5.26 Comparison of recall scores for emotion classification

The above table depicts the comparison of recall scores for each emotion. The first three models in the tables are observed to have better and consistent recall scores for each emotion than the rest with the Bi-LSTM outperforms all other models with recall scores of about 99% for some of the emotions.

- Comparison of approaches for Age classification

No.	Vectorization	Model	Train Accuracy	Test Accuracy
1	TF-IDF	Deep Learning	62.361%	56.835%
2	TF-IDF	GBM	62.118%	60.941%
3	Word2vec	Deep Learning	67.710%	62.553%
4	Word2vec	GBM	68.701%	64.960%

*Table 4.5.27 Comparison of approaches for age classification*

The above table depicts the comparison of approaches for age classification that vectorization is important as the Word2vec GBM outperforms TF-IDF GBM and the Word2vec Deep learning outperforms TF-IDF Deep Learning. The best model from the four model is Word2vec GBM with better train and test accuracy.

- Comparison of age classifier model recall scores

No.	Vectorization	Model	Youth	Adult
1	TF-IDF	H2O Deep Learning	53.95%	73.98%
2	TF-IDF	H2O GBM	57.07%	74.32%
3	Word2vec	H2O Deep Learning	58.32%	72.12%
4	Word2vec	H2O GBM	60.50%	73.95%

*Table 4.5.28 Comparison of age classifier model recall scores*

The above table represents comparison of age classifier model accuracies. All of the above models achieved better recall scores for Adult data than Youth data. The Word2vec H2O GBM model outperformed all other models with better recall scores for Youth as well as Adult class.

- Comparison of approaches applied on Youth and Adult classified emotion data

No.	Vectorization	Models	Youth data emotion accuracy	Adult data emotion accuracy
1	Text-to-Sequences	Bi-LSTM	95.698%	94.907%
2	TF-IDF	LinearSVC	96.593%	96.964%
3	Word2vec – 100 vector	H2O Deep Learning	81.974%	74.437%
4	Word2vec – 500 vector	H2O Deep Learning	86.465%	78.893%

*Table 4.5.29 Comparison of approaches applied on Youth and Adult classified emotion data*

The above table represents the accuracy scores emotion classifier models applied on Youth and Adult stratified emotion data. The Linear SVC outperforms the Bi-LSTM with a fractional margin while the word2vec 500 vector size H2O DL model outperforms 100 vector size H2O DL model.

## 5.4 Limitations

There are various limitations to this research, the below points elaborate each limitation with respect to the tools, environment and the technological restrains:

1. The dataset used here comprises of 89832 records but with only six emotions. It is difficult to get dataset that suits to all the emotions. The dataset related to in-detailed emotions like neutral, stressed, tired etc are difficult to derive.
2. This research is a multiclass classification problem and the suitable models that can perform this classification are limited.
3. Time constrains - Each model takes a lot of time to train, therefore after every parameter tuning step the model has to rerun again to check the performance metrics.
4. H2O.ai is a fresh tool and hence the features provided by sklearn, tensorflow and pandas are not available in h2o at this given point of time while writing this research. For example, sklearn provides a solution to get the recall scores while there is no such support provided by h2o for the same.
5. When importing the dataset into H2O python as h2o frame, the H2O considers first row which is the column heading as a record instead of heading and assigns another heading name. Therefore, the dataset's heading need to be removed before passing the dataset to H2O.
6. Pre-processing steps in sklearn are time consuming, it takes a lot of time to pre-process text to vectors.
7. Pre-processing steps in Rapid Miner studio takes a lot of time (days) therefore it is not feasible to make use of studio for data pre-processing.
8. It is not feasible to use rapid miner on the cloud as "rapid miner go" only supports data file upto 50mb. Therefore, rapid miner implementation through rapid miner studio takes a lot of time and the performance is also low as it utilizes the available laptop infrastructure (RAM, CPU, GPU etc). There is no version control between the H2O in python and H2O in rapid miner hence there are differences between performance metrics of H2O python and H2O rapid miner. The H2O version in Rapid miner was found out to be lower than the H2O python version.
9. For age classification through m3inference, the dataset with username had to be divided into multiple files as the twitter API only allows a certain number of requests at a time. If the entire file is passed through then a few parts of the data will have blank spaces which will return errors while passing it through the age categorizer python code.

## 5.5 Future work

The future works includes :

- Creating a model which can classify all the emotions in text, even multiple emotions in a single line of text.
- The performance of the age classifier model can be improved and then implemented for various applications.
- The age classifier can be further implemented in a chatbot which can identify the age group of a person and can provide replies based on specific age groups.
- The age classifier can be set to three categories Teens, Youth and Adult as M3inference returns multiple classes '<=18', '19-29', '30-39' and '>=40'.

## Chapter 5 – Conclusion

Comparing the performance metrics of state of the art approaches and specialized deep learning approaches, specialized deep learning Bi-LSTM outperforms state of the art LinearSVC with an accuracy score of 95.28% and recall of each class to be above 90% while LinearSVC with TF-IDF has an accuracy score of 91.51% and recall of each class near 90%. The vectorization proves to have an important role here, it is observed that TF-IDF outperforms Word2vec model in terms of accuracy and recall. The TF-IDF with H2O Deep Learning ANN having accuracy of 86.41% outperforms the Word2vec 100 vector size with H2O Deep Learning having accuracy of 75.548%. It is observed that the Word2vec 500 vector size with H2O Deep Learning ANN having accuracy of 80.50% outperformed Word2vec 100 vector size with H2O Deep Learning ANN with an increased accuracy of about 4.95%. The algorithm used for age classification are GBM and Deep Learning ANN. GBM proved to have a better accuracy score in comparison with Deep Learning ANN when used with Word2vec vectorization. GBM with Word2vec vectorization has an accuracy score of 64.96% while Deep Learning with Word2vec has an accuracy score of 62.55%. The TF-IDF GBM with accuracy 60.94% outperforms the TF-IDF Deep Learning with accuracy 56.84%. The TF-IDF LinearSVC with an accuracy of 96.593% on Youth emotion data and 96.964% on Adult emotion data outperforms the specialized deep learning Bi-LSTM with an accuracy of 95.698% on Youth emotion data and accuracy of 94.907% on Adult emotion data. The difference between both the models is fractional. Both Bi-LSTM and LinearSVC outperform H2O Word2vec Deep learning ANN with vector sizes 100 and 500. The pre-processing vectorization steps prove to be important and affect the performance of each model.

## Chapter 6 – Referencing

About the virus (no date). Available at: <https://www.euro.who.int/en/health-topics/health-emergencies/coronavirus-covid-19/novel-coronavirus-2019-ncov> (Accessed: 26 April 2021).

AlBalooshi, H., Rahmanian, S. and Venkatesh Kumar, R. (2018) ‘EmotionX-SmartDubai\_NLP: Detecting User Emotions In Social Media Text’, in Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media. Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media, Melbourne, Australia: Association for Computational Linguistics, pp. 45–49. doi: 10.18653/v1/W18-3508.

Chawla, S. and Mehrotra, M. (2018) ‘An Ensemble-Classifier Based Approach for Multiclass Emotion Classification of Short Text’, in 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO). 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), pp. 768–774. doi: 10.1109/ICRITO.2018.8748757.

Cortes, C. and Vapnik, V. (1995) ‘Support-vector networks’, *Machine Learning*, 20(3), pp. 273–297. doi: 10.1007/BF00994018.

Fei, R. et al. (2020) ‘A Deep Learning Method Based Self-Attention and Bi-directional LSTM in Emotion Classification’, undefined. Available at: </paper/A-Deep-Learning-Method-Based-Self-Attention-and-in-Fei-Zhu/26fc6f4a8d374b632db3c9518269fb5cfe0db667> (Accessed: 6 May 2021).

Gandhi, R. (2018) Support Vector Machine — Introduction to Machine Learning Algorithms, Medium. Available at: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47> (Accessed: 11 May 2021).

‘German Predictive Analytics Startup Rapid-I Rebrands As RapidMiner, Takes \$5M From Open Ocean, Earlybird To Tackle The U.S. Market’ (no date) TechCrunch. Available at: <https://social.techcrunch.com/2013/11/04/german-predictive-analytics-startup-rapid-i-rebrands-as-rapidminer-takes-5m-from-open-ocean-earlybird-to-tackle-the-u-s-market/> (Accessed: 23 May 2021).

Gohil, L. and Patel, D. (2019) ‘Multilabel Classification for Emotion Analysis of Multilingual Tweets’, in. doi: 10.35940/ijitee.a5320.119119.

Guntuku, S. C. et al. (2020) ‘Tracking Mental Health and Symptom Mentions on Twitter During COVID-19’, *Journal of General Internal Medicine*. doi: 10.1007/s11606-020-05988-8.

Gupta, U. et al. (2017) ‘A Sentiment-and-Semantics-Based Approach for Emotion Detection in Textual Conversations’, ArXiv.

H2O.ai - Crunchbase Company Profile & Funding (no date) Crunchbase. Available at: <https://www.crunchbase.com/organization/h2o-2> (Accessed: 12 May 2021).

Hochreiter, S. and Schmidhuber, J. (1997) 'Long Short-term Memory', *Neural computation*, 9, pp. 1735–80. doi: 10.1162/neco.1997.9.8.1735.

Interview with RapidMiner's Ingo Mierswa, Ralf Klinkenberg, part 1 (no date). Available at: <https://www.kdnuggets.com/2010/02/f-interview-rapid-i-founders.html> (Accessed: 23 May 2021).

kandicst/Emotion-Classification (no date) GitHub. Available at: <https://github.com/kandicst/Emotion-Classification> (Accessed: 6 May 2021).

Kang, E. (2017) Long Short-Term Memory (LSTM): Concept, Medium. Available at: <https://medium.com/@kangeugine/long-short-term-memory-lstm-concept-cb3283934359> (Accessed: 11 May 2021).

Karna, M., Juliet, D. S. and Joy, R. C. (2020) 'Deep learning based Text Emotion Recognition for Chatbot applications', in 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184). 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), pp. 988–993. doi: 10.1109/ICOEI48184.2020.9142879.

Lazemi, S. and Ebrahimpour-Komleh, H. (2018) 'Multi-Emotion Extraction from Text Using Deep Learning', *International Journal of Web Research*, 1(1), pp. 62–67. doi: 10.22133/ijwr.2018.70577.

Lemaître, G., Nogueira, F. and Aridas, C. K. (2017) 'Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning', *The Journal of Machine Learning Research*, 18(1), pp. 559–563.

Lwin, M. et al. (2020) 'Global Sentiments Surrounding the COVID-19 Pandemic on Twitter: Analysis of Twitter Trends', *JMIR public health and surveillance*. doi: 10.2196/19447.

Mahanta, J. (2017) Introduction to Neural Networks, Advantages and Applications, Medium. Available at: <https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207> (Accessed: 11 May 2021).

Namrutha Sridhar, B. V., Mrinalini, K. and Vijayalakshmi, P. (2020) 'Data Annotation and Multi-Emotion Classification for Social Media Text', in 2020 International Conference on Communication and Signal Processing (ICCSP). 2020 International Conference on Communication and Signal Processing (ICCSP), pp. 1011–1015. doi: 10.1109/ICCSP48568.2020.9182362.

Nations, U. (no date) Youth, United Nations. United Nations. Available at: <https://www.un.org/en/global-issues/youth> (Accessed: 20 May 2021).

Practical machine learning with H2O: powerful, scalable techniques for deep learning and AI | Cook, Darren | download (no date). Available at: <https://2lib.org/book/5686437/c92816> (Accessed: 4 May 2021).

Quantum (2019) Data Science project management methodologies, Medium. Available at: <https://medium.datadriveninvestor.com/data-science-project-management-methodologies-f6913c6b29eb> (Accessed: 7 May 2021).

Rajabi, Z., Shehu, A. and Uzuner, O. (2020) ‘A Multi-channel BiLSTM-CNN Model for Multilabel Emotion Classification of Informal Text’, in 2020 IEEE 14th International Conference on Semantic Computing (ICSC). 2020 IEEE 14th International Conference on Semantic Computing (ICSC), pp. 303–306. doi: 10.1109/ICSC.2020.00060.

Singh, H. (2018) Understanding Gradient Boosting Machines, Medium. Available at: <https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab> (Accessed: 18 May 2021).

statinfer (2017) ‘204.6.8 SVM: Advantages Disadvantages and Applications | Statinfer’, 14 March. Available at: <https://statinfer.com/204-6-8-svm-advantages-disadvantages-applications/> (Accessed: 11 May 2021).

‘Understanding of LSTM Networks’ (2020) GeeksforGeeks, 10 May. Available at: <https://www.geeksforgeeks.org/understanding-of-lstm-networks/> (Accessed: 11 May 2021).

Wang, Z. et al. (2019a) ‘Demographic Inference and Representative Population Estimates from Multilingual Social Media Data’, The World Wide Web Conference, pp. 2056–2067. doi: 10.1145/3308558.3313684.

‘youth-definition.pdf’ (no date). Available at: <https://www.un.org/esa/socdev/documents/youth/fact-sheets/youth-definition.pdf> (Accessed: 16 May 2021).

Zhang, Y. et al. (2018) ‘Text Emotion Distribution Learning via Multi-Task Convolutional Neural Network’, in IJCAI. doi: 10.24963/ijcai.2018/639.

## Appendices

### I. Contents of the Artifacts

#### A. Emotion Classification

##### 1. Datasets

- **emotion\_data.csv** – The initial dataset downloaded from github
- **shuffled.csv** – The dataset created from emotion\_data.csv after using undersample.ipynb to balance the dataset
- **data.csv** – The dataset with target column derived from emotion column that is label to ids - surprise : 0, love : 1, Anger : 2, Sadness : 3, Joy : 4, fear : 5.
- **word2vec-label.csv** – The dataset passed through rapid miner auto model.

##### 2. Model Results

- **RM-TrainTest.docx** – This file contains RapidMiner results for train and test split word2vec 100 vector data.
- **ModelResults.docx** - This file contains all the model results with screenshots and tables.

##### 3. Python code

- **SVM-TFIDF.ipynb** – Code for implementing LinearSVC by TF-IDF vectorization using shuffled.csv dataset.
- **TFIDF-DEEP-H2O.ipynb** – Code for implementing H2ODeepLearningEstimator with TF-IDF vectorization using shuffled.csv dataset.
- **100-WORD2VEC-DEEP-H2O.ipynb** – Code for implementing H2ODeepLearningEstimator with Word2vec 100 vector size vectorization using data.csv dataset.
- **500-WORD2VEC-DEEP-H2O.ipynb** – Code for implementing H2ODeepLearningEstimator with Word2vec 500 vector size vectorization using data.csv dataset.
- **Bi\_LSTM.ipynb (google colab)** – Code for implementing Bi-LSTM with text to sequences and padding truncating of sequences for pre-processing using data.csv dataset.
- **Undersample.ipynb** – Code for balancing the emotion\_data.csv and saved as shuffled.csv

##### 4. Readme:

Refer the attached readme file for installation of H2O and other packages.(All the .ipynb files are run on google colab/jupyter notebook)

## B. Age Classification

### 1. Datasets

- **data-emo.csv** – initial dataset scrapped from twitter using twint\_twitter.py
- **covi.csv** – dataset created from data-emo.csv using data\_cleaning.py
- **users folder** – 64 ‘.csv’ files made by splitting covi.csv
- **raw-m3 folder** – 64 ‘.txt’ files made using m3inference.py on csv files from users folder
- **gender-class folder** – files containing ‘.csv’ file and ‘.txt’ file for gender classification using m3inference.py
- **cleaaaaa.csv** – using age-categorizer.py on 64 ‘.txt’ files and storing it in cleaaaaa.csv file.
- **clean1.csv** – using cleaaaaa-to-clean1.ipynb on cleaaaaa.csv to get clean1.csv
- **clean3.csv** – using clean1-to-clean3.ipynb to get clean3.csv
- **balance-01.csv** – using undersampler-age.ipynb to get balanced dataset balance-01.csv

### 2. Model Results

- **ModelResults.docx** – This file contains all the results of the models.

### 3. Python code

- **AGE-WORD2VEC.ipynb** – Code for implementing H2OGradientBoostingEstimator and H2ODeepLearningEstimator by word2vec 100 vector size vectorization using balance-01.csv dataset.
- **AGE-TFIDF.ipynb** – Code for implementing H2OGradientBoostingEstimator and H2ODeepLearningEstimator by TF-IDF (max features set to 1000) vectorization using balance-01.csv dataset.
- **twint-twitter.py** – Code for getting tweets from twitter
- **data-cleaning.py** – Code for cleaning dataset
- **m3-inference.py** – Code for getting age groups using m3inference
- **age-categorizer.ipynb** – Code for classifying age groups based on conditional statements
- **cleaaaaa-to-clean1.ipynb** – Code for cleaning dataset
- **clean1-to-clean3.ipynb** – Code for cleaning dataset
- **Undersampler-age.ipynb** – Code for balancing imbalanced dataset

### 4. Readme:

Refer the attached readme file for installing and running python code, code run on google colab is specified near to filename in brackets.

## C. Age-Emotion Classification

### 1. Dataset

- **data.csv** – dataset from emotion classification
- **youth.csv** – data created by deploying GBM age classifier model on data.csv having only youth records
- **adult.csv** - data created by deploying GBM age classifier model on data.csv having only adult records
- **age\_emot.csv** – similar dataset to data.csv but with a column having youth as 1 and adult as 0 values

### 2. Model Results

- **ModelResults.docx** – This file contains all the results of the models.

### 3. Python code

- **AGE-TO-EMOT.ipynb** – Word2vec GBM age classifier from AGE-WORD2VEC.ipynb is implemented on the data.csv dataset and three new datasets are created – youth.csv, adult.csv and age\_emot.csv. All the above emotion classification models are used for prediction on youth.csv as well as adult.csv data.

## D. Chatbot

### 1. Models

- **w2v\_emot500.hex** – The Word2vec 500 vector size vectorization model.
- **500-WORD2VEC-DEEP-H2O** - The H2O Deep Learning model trained using the 500 vector size word to vectorized text data.

### 2. Python code

- **app.py** – This app is run using pycharm and comprises of the code for loading models, code for predictions and the flask api. This ‘app.py’ serves as an API for the chatbot app

### 3. Swift code

- **HealthHack** – This folder contains the chatbot app named as ‘HealthHack.xcodeproj’, this file only works with the xcode in macos.

### 4. Readme

Refer the readme file for the working of this app.