# Final Project Report

Annie Udhani

5/9/2021

## Introduction

In this final project, we will be exploring and applying various NLP and neural net technique to the different textual and image datasets. The objective of this exercise is to demonstrate the practical knowledge of various ML/NLP techniques learned throughout the course by applying it to the real world datasets

We will namely be performing the below analysis:

1. Word Cloud Formation
2. Sentiment Analysis
3. Topic Modeling
4. Image classification using Convolutional neural net

## Data definition

For the purpose of this project, we would be using the below two datasets: 1. **Vaccination Tweets, India**: This dataset is sourced from Kaggle. The dataset contains various tweets from Indian twitter user which contains the word vaccination. The tweets are for the four months of year from January 2021 to April 2021 separated in different files where each file contains tweets for a month. The various column attributes and data definition are summarized in below table:

*Vaccination Tweets - Data definition*

| Field Name | Field Description | Data type |
|---|---|---|
| date | Date of tweet | date |
| tweet | Text describing the tweet | string |
| replies_count | Count of Replies recieved on tweet | integer |
| retweets_count | Count of Retweets recieved on tweet | integer |
| likes_count | Count of Likes recieved on tweet | integer |
| hashtags | Hashtags if present in tweet | string |

2. **Spam Text Message Classification Dataset**: This dataset is also sourced from Kaggle and contains 5157 examples of text messages and corresponding label classified as spam or ham(not spam). We would be utilizing this dataset in the later sections of this project to run our text classification algorithm using artificial neural networks. The

various column attributes and data definition for this dataset are summarized in below table:

*Spam Text Classification Dataset - Data definition*

| Field Name | Field Description | Data type |
|---|---|---|
| Category | Category describing whether a message is spam or not | string |
| Message | Text content of actual message | string |

3. **Imagenette dataset**: Imagenette is a dataset which is extracted from the ImageNet dataset(14M labelled images across 20k categories). Imagenette was made from Imagenet containing around 13k images across 10 categories. This was created for researcher and neural network practitioner to easily run their neural nets on. I would be using this dataset to run a convolutional neural network in the later section of this project. 10 Categories that are used in this dataset are:

*Labels used in Imagenette dataset*

| # | Categories |
|---|---|
| 1 | tench |
| 2 | English springer |
| 3 | cassette player |
| 4 | chain saw |
| 5 | church |
| 6 | French horn |
| 7 | garbage truck |
| 8 | gas pump |
| 9 | golf ball |
| 10 | parachute |

## NLP Analysis And Artifical Neural Network

In this section we will be performing Natural language processing on tweets datasets using various analysis techniques. But before performing any analysis, let's import our tweets dataset into our workspace.

## Importing tweet dataset and loading required libraries

```
library(tm)

## Loading required package: NLP

library(readr)
library(tidytext)
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:NLP':
##
##     annotate

library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(topicmodels)
library(SnowballC)
# reading tweet files
library(readr)
jan_tweets <- read_csv("C:\\Users\\Annie\\Documents\\ML\\End term\\Sentiment
Analysis\\January_Vaccination_Tweets.csv")

##
## -- Column specification --------------------------------------------------
------
## cols(
##   date = col_date(format = ""),
##   tweet = col_character(),
##   replies_count = col_character(),
##   retweets_count = col_character(),
##   likes_count = col_character(),
##   hashtags = col_character()
## )

feb_tweets <- read_csv("C:\\Users\\Annie\\Documents\\ML\\End term\\Sentiment
Analysis\\February_Vaccination_Tweets.csv")

##
## -- Column specification --------------------------------------------------
------
## cols(
##   date = col_date(format = ""),
##   tweet = col_character(),
##   replies_count = col_character(),
##   retweets_count = col_character(),
##   likes_count = col_character(),
```

```
##   hashtags = col_character()
## )

mar_tweets <- read_csv("C:\\Users\\Annie\\Documents\\ML\\End term\\Sentiment
Analysis\\March_Vaccination_Tweets.csv")

##
## -- Column specification --------------------------------------------------
------
## cols(
##   date = col_date(format = ""),
##   tweet = col_character(),
##   replies_count = col_character(),
##   retweets_count = col_character(),
##   likes_count = col_double(),
##   hashtags = col_character()
## )

apr_tweets <- read_csv("C:\\Users\\Annie\\Documents\\ML\\End term\\Sentiment
Analysis\\April_Vaccination_Tweets.csv")

##
## -- Column specification --------------------------------------------------
------
## cols(
##   date = col_date(format = ""),
##   tweet = col_character(),
##   replies_count = col_character(),
##   retweets_count = col_character(),
##   likes_count = col_character(),
##   hashtags = col_character()
## )

jan_tweets <- jan_tweets["tweet"]
feb_tweets <- feb_tweets["tweet"]
mar_tweets <- mar_tweets["tweet"]
apr_tweets <- apr_tweets["tweet"]
all_tweets <- rbind(jan_tweets, feb_tweets, mar_tweets, apr_tweets)
```

Once the dataset is imported, let's foucs on various NLP analysis on the imported datasets to answer various questions.

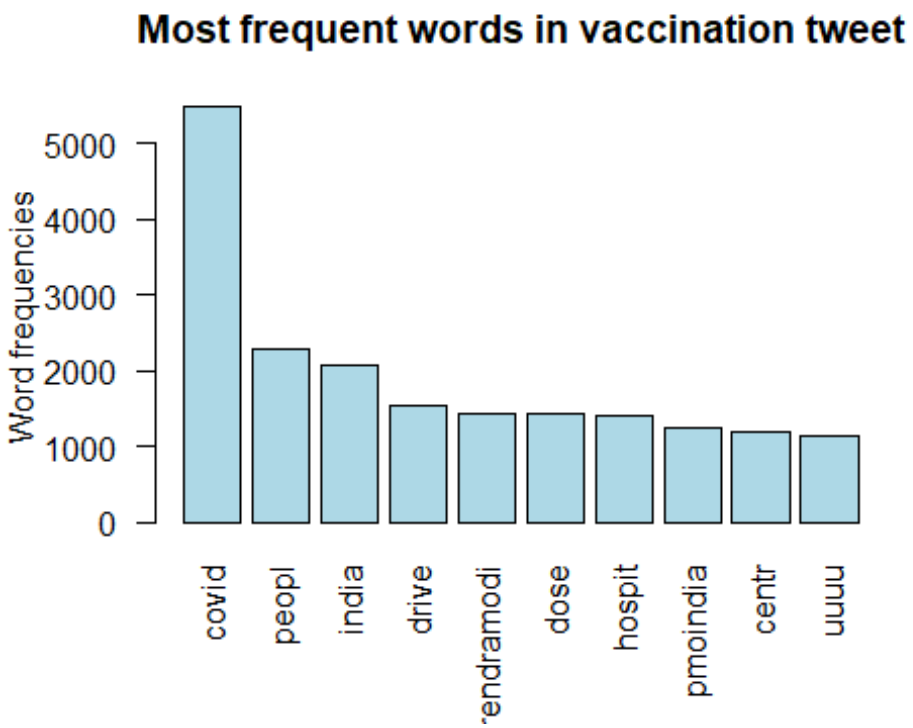
1. **Word Embedding and Word Cloud Creation** : We will be applying NLP technique using R and will be forming a word cloud on the vaccination tweets data that we sourced from Kaggle. For creating such a cloud, we will be embedding sentences in the form of a Document Term Matrix. And the question that we will be trying to answer with this analysis is - **What are the top trending words that general population is using in tweet containing vaccincation keyword?** This can help us to understand the general mood and trending behavior of population in general, especially at a crucial time when vaccination drive is being carried on, in the whole country of India.

## Cleaning text data in tweets

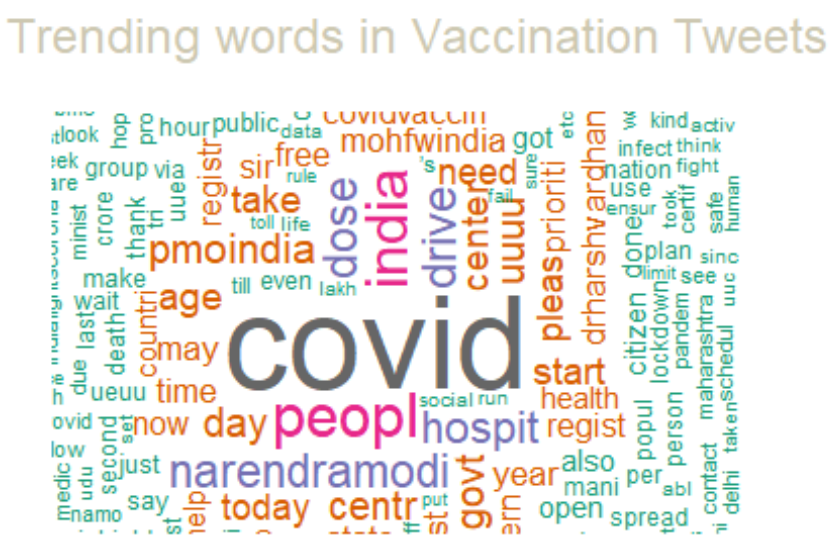
```
all_tweets <- na.omit(all_tweets)
# Transform and clean the text
library("tm")
docs <- Corpus(VectorSource(all_tweets))
# Convert the text to lower case
docs <- tm_map(docs, content_transformer(tolower))
# Remove numbers
docs <- tm_map(docs, removeNumbers)
# Remove english common stopwords
docs <- tm_map(docs, removeWords, stopwords("english"))
# Remove punctuations
docs <- tm_map(docs, removePunctuation)
# Eliminate extra white spaces
docs <- tm_map(docs, stripWhitespace)
# Text stemming (reduces words to their root form)
docs <- tm_map(docs, stemDocument)
# Remove additional stopwords
docs <- tm_map(docs, removeWords, c("film", "movi", "one", "two","can", "will
", "vaccin", "amp", "get", "uuu"))
```

## Creating word cloud

```
#Create term document matrix
dtm <- TermDocumentMatrix(docs)
#Remove less frequent terms to reduce the dimension of the DTM
dtm <- removeSparseTerms(dtm, 0.95)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
#findFreqTerms(dtm, lowfreq = 100, highfreq = Inf)
head(d, 10)

##                       word freq
## covid                covid 5486
## peopl                peopl 2295
## india                india 2083
## drive                drive 1536
## narendramodi narendramodi 1433
## dose                  dose 1421
## hospit              hospit 1392
## pmoindia          pmoindia 1237
## centr                centr 1185
## uuuu                  uuuu 1148

barplot(d[1:10,]$freq, las = 2, names.arg = d[1:10,]$word,
        col ="lightblue", main ="Most frequent words in vaccination tweet",
        ylab = "Word frequencies")
```




```
# Generate the WordCloud
library("wordcloud")

## Loading required package: RColorBrewer

library("RColorBrewer")
wordcloud(words = d$word, freq = d$freq, min.freq = 10,
          max.words=500, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
title(main = "Trending words in Vaccination Tweets",
      font.main = 1, col.main = "cornsilk3", cex.main = 1.5)
```



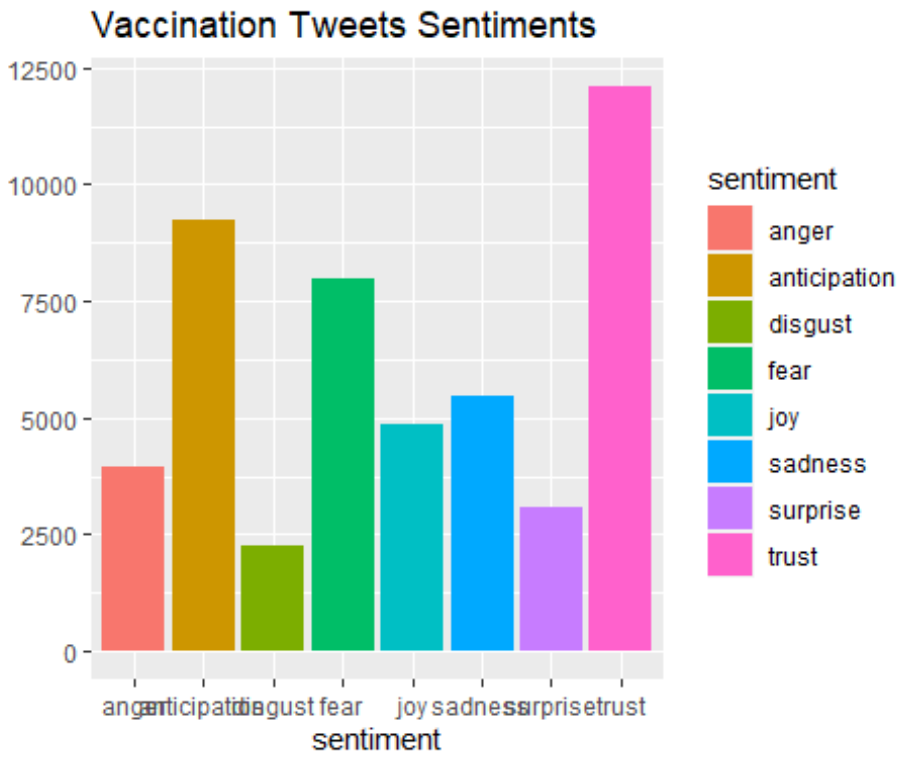

We can clearly see that covid, corona, covishield(a vaccine against coronavirus), hospitals are the word that are being used a lot in these tweets.

2. **Sentiment Analysis** : With this analysis, we would like to figure out the general sentiment of overall population. The question that we can answer using sentiment analysis here is that **How are people feeling in the midst of pandemic? Are they anxious and fearful? Do they have a sense of overall trust?** Let's perform sentiment analysis on the data to see the different sentiments of audience tweeting about it.

## Sentiment Analysis in R:

```
body <- data.frame(na.omit(all_tweets))
library('syuzhet')
sentences <- data.frame(get_sentences(body$tweet))
names(sentences)[1] <- "Text"
sentences$Text <- as.character(sentences$Text)

sentences$syuzhet = get_sentiment(sentences$Text, method="syuzhet")
sentences$bing = get_sentiment(sentences$Text, method="bing")
sentences$afinn = get_sentiment(sentences$Text, method="afinn")
sentences$nrc = get_sentiment(sentences$Text, method="nrc")
emotions<-get_nrc_sentiment(sentences$Text)
n = names(emotions)
for (nn in n) sentences[, nn] = emotions[nn]
```

```
td<-data.frame(t(emotions))
#The function rowSums computes column sums across rows for each level of a gr
ouping variable.
td_new <- data.frame(rowSums(td[2:dim(sentences)[1]]))
#Transformation and  cleaning
names(td_new)[1] <- "count"
td_new <- cbind("sentiment" = rownames(td_new), td_new)
rownames(td_new) <- NULL
td_new2<-td_new[1:8,]
library("ggplot2")
qplot(sentiment, data=td_new2, weight=count, geom="bar",fill=sentiment)+ggtit
le("Vaccination Tweets Sentiments")
```




As evident from above sentiment bar chart, currently trust, anticipation and fear is the most dominant sentiment in the Indian population for vaccination.

3. **Topic Modeling** : We will perform one more analysis here to answer **what are the abstract topics these tweets can be categorized into.** We will be using LDA analysis to formulate multiple abstract topics and understand which topic these documents fall into.

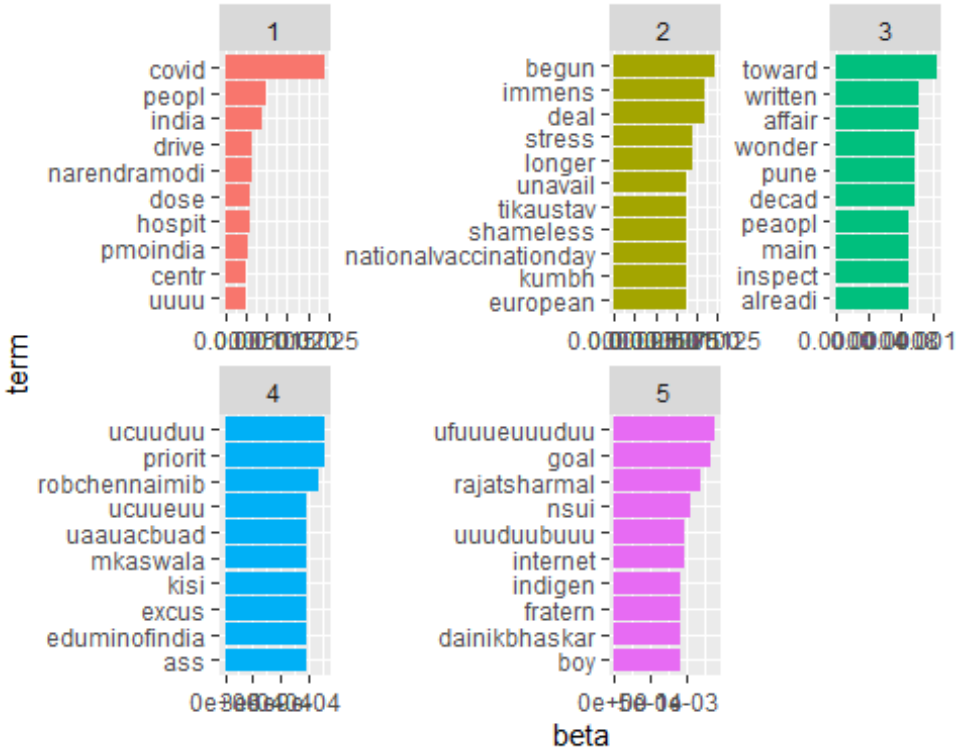
## Topic Modeling in R:

```
dtm <- DocumentTermMatrix(docs)

#Run Latent Dirichlet Allocation (LDA) using Gibbs Sampling
```

```r
ldaOut <- LDA(dtm, method="gibbs", k = 5, control = list(seed = 1234))
terms(ldaOut,6)
```

```
##      Topic 1        Topic 2    Topic 3   Topic 4         Topic 5
## [1,] "covid"        "begun"    "toward"  "priorit"       "ufuuueuuuduu"
## [2,] "peopl"        "deal"     "affair"  "ucuuduu"       "goal"
## [3,] "india"        "immens"   "written" "robchennaimib" "rajatsharmal"
## [4,] "drive"        "longer"   "decad"   "ass"           "nsui"
## [5,] "narendramodi" "stress"   "pune"    "eduminofindia" "internet"
## [6,] "dose"         "european" "wonder"  "excus"         "uuuduubuuu"
```

```r
#Extract the per-topic-per-word probabilities, called "beta",
#from the model using tidytext package
tweet_topics <- tidy(ldaOut, matrix = "beta")

#Manipulate data to plot per topic per word probabilities
tweet_top_terms <- tweet_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

tweet_top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic,scales = "free") +
  coord_flip()
```




4. **Text Classification**: We will be using Spam Text Message classification dataset sourced from Kaggle for this purpose. We will be using 80% of sourced data to train an **Artificial Neural networks**. To train the neural net, we will be converting the text data in every message into a sequence of words using Keras Tokenizer. Every index in the sequence will correspond to a word in Tokenizer's dict and we will limit words to a max of 20,000 unique words in the Tokenizer's dict for the purpose of this project. Once we have created this quantitative feature set from text, we will feed this features along with labels into a very artificial neural net with no hidden layer using sigmod activation function.

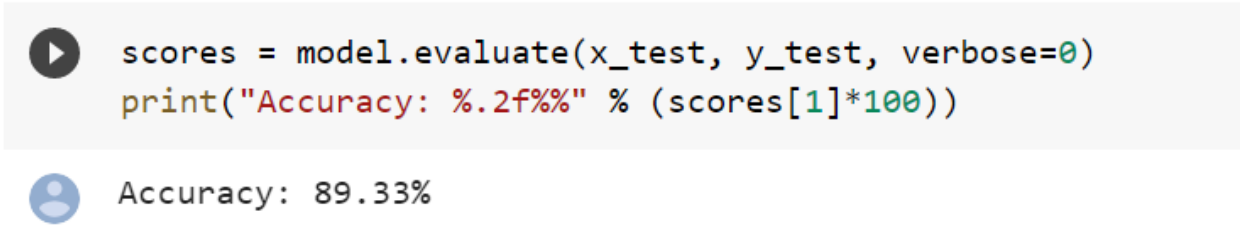
The question that we are trying to solve here is given a dump of text message – **How accurately our ML model can help us to focus on important information by filtering out spam messages from non-spam text messages.**

We will be performing all these process using python. I have written the python code on python notebook and have run it using Google Collaboratory. Please refer to python notebook named **Spam_Text_Classification.ipynb** for more details on code.

I am putting down the training set accuracy and test set accuracy obtained after running the code in Google Collab.

```
[ ] 63/63 [==============================] - 0s 4ms/step - loss: 0.2340 - accuracy: 0.9058 - val_loss: 0.3447 - val_accuracy: 0.8677
    Epoch 8/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.2087 - accuracy: 0.9175 - val_loss: 0.3270 - val_accuracy: 0.8700
    Epoch 9/20
    63/63 [==============================] - 0s 3ms/step - loss: 0.2120 - accuracy: 0.9190 - val_loss: 0.3823 - val_accuracy: 0.8700
    Epoch 10/20
    63/63 [==============================] - 0s 3ms/step - loss: 0.2131 - accuracy: 0.9134 - val_loss: 0.3187 - val_accuracy: 0.8722
    Epoch 11/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1969 - accuracy: 0.9265 - val_loss: 0.3227 - val_accuracy: 0.8857
    Epoch 12/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1888 - accuracy: 0.9301 - val_loss: 0.3206 - val_accuracy: 0.8879
    Epoch 13/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1908 - accuracy: 0.9306 - val_loss: 0.3134 - val_accuracy: 0.8924
    Epoch 14/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1827 - accuracy: 0.9266 - val_loss: 0.3081 - val_accuracy: 0.8924
    Epoch 15/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1790 - accuracy: 0.9365 - val_loss: 0.3269 - val_accuracy: 0.8946
    Epoch 16/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1831 - accuracy: 0.9309 - val_loss: 0.3290 - val_accuracy: 0.8812
    Epoch 17/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1772 - accuracy: 0.9347 - val_loss: 0.3294 - val_accuracy: 0.8812
    Epoch 18/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1671 - accuracy: 0.9385 - val_loss: 0.3197 - val_accuracy: 0.8946
    Epoch 19/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1840 - accuracy: 0.9301 - val_loss: 0.3588 - val_accuracy: 0.8744
    Epoch 20/20
    63/63 [==============================] - 0s 4ms/step - loss: 0.1649 - accuracy: 0.9428 - val_loss: 0.3531 - val_accuracy: 0.8812
```

Test Set accuracy on Spam Text Classification data set is found to be 89.33%. For more details on code, please refer to **Spam_Text_Classification.ipynb** file attached with this report.




5. **Convolutional Neural Network (CNN)**: In today's world, convolutional neural network has find a lot of applications in solving modern problems mainly related to image or speech analysis due to the fact that it does take into account the spatial property contained in the data which a regular Artificial neural network does not support out of the box. We will be using similar application of CNN here to classify a set of images into 10 different categories. Once we perform the classification using CNN, we will measure the performance of classification on test set.

The question that we are trying to solve here is given a dump of text message – **Can we rely on Machine to act as an eye for us for the problem of identifying the objects and classifying them into the right category?**

We will be using the imagenette dataset described in data definiton section above for this purpose. Dataset contains around 13k images across 10 categories. We will use a CNN with convolutional hidden layer.

Below are the training accuracy results after fitting the CNN model on Imagenette dataset. For more details, please refer to **Imagenette_CNN.ipynb** file attached with this report.

```
[ ]  Epoch 8/20
     74/74 [==============================] - 1033s 14s/step - loss: 2.0124 - accuracy: 0.3032
     Epoch 9/20
     74/74 [==============================] - 1026s 14s/step - loss: 1.9823 - accuracy: 0.2949
     Epoch 10/20
     74/74 [==============================] - 1023s 14s/step - loss: 1.8915 - accuracy: 0.3399
     Epoch 11/20
     74/74 [==============================] - 1038s 14s/step - loss: 1.8923 - accuracy: 0.3487
     Epoch 12/20
     74/74 [==============================] - 1034s 14s/step - loss: 1.7628 - accuracy: 0.3968
     Epoch 13/20
     74/74 [==============================] - 1039s 14s/step - loss: 1.7314 - accuracy: 0.4070
     Epoch 14/20
     74/74 [==============================] - 1032s 14s/step - loss: 1.7042 - accuracy: 0.4149
     Epoch 15/20
     74/74 [==============================] - 1038s 14s/step - loss: 1.5804 - accuracy: 0.4663
     Epoch 16/20
     74/74 [==============================] - 1022s 14s/step - loss: 1.5178 - accuracy: 0.4951
     Epoch 17/20
     74/74 [==============================] - 1024s 14s/step - loss: 1.4962 - accuracy: 0.5040
     Epoch 18/20
     74/74 [==============================] - 1027s 14s/step - loss: 1.4005 - accuracy: 0.5299
     Epoch 19/20
     74/74 [==============================] - 1014s 14s/step - loss: 1.3738 - accuracy: 0.5576
     Epoch 20/20
     74/74 [==============================] - 1026s 14s/step - loss: 1.2789 - accuracy: 0.5636
```

Test Set accuracy that we found for this data set is around 54.38%.

```
[ ]  scores = model.evaluate(X_test, Y_test, verbose=0)
     print("Accuracy: %.2f%%" % (scores[1]*100))

     Accuracy: 54.38%
```

## Conclusion

Machine Learning with its modern state of art NLP techniques and Neural network plays a very pivotal role in our lives. We are using a lot of tools, utilities, websites, etc. without even realizing that we are being served by these algorithms automatically.

In this report, we used a few state of art ML and NLP technique and algorithms to answer a few questions around the dataset efficiently. We answered how predominant fear or trust is among Indian population around Covid 19 and its vaccination. We tried to answer how can we model these tweets by users into different topics. We tried to solve a problem faced by the modern world where everyone is being bombarded with spam text messages. We

were able to efficiently differentiate between spam and non-spam text by efficiently classifying them into the right bucket with 89.33% accuracy, leaving it for user to action on them. We also tried to solve the problem of manual human intervention to identify images and classify them into the right category with an accuracy of 54.38%.

## References:


1. India Vaccination Tweets dataset: https://www.kaggle.com/vineethakkinapalli/vaccination-tweets-india-january-to-april-2021/activity
2. Spam Text Classification Data: https://www.kaggle.com/team-ai/spam-text-message-classification
3. Imagenett DataSet: https://github.com/fastai/imagenette