Text Classification

Using web-chatter data to classify Patients.

PROBLEM STATEMENT

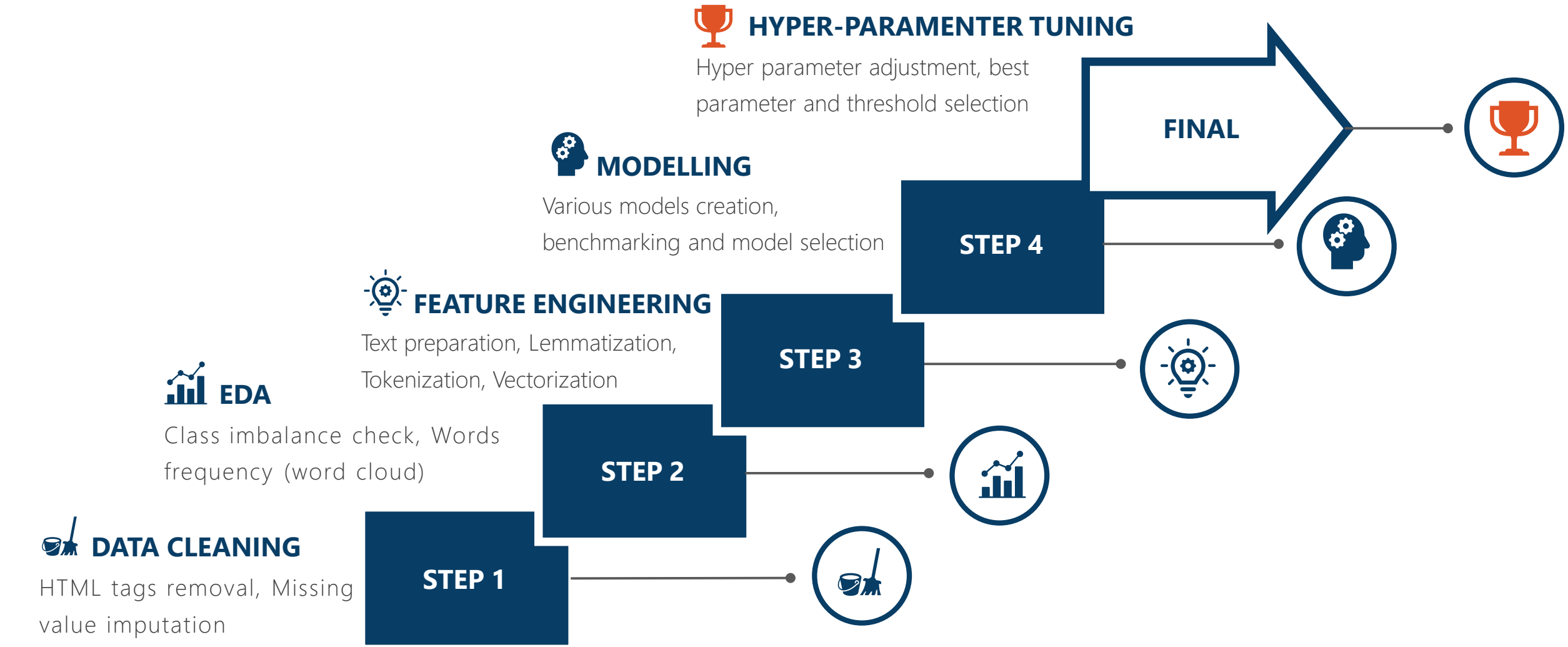
Objective

- Automate the process of gaining insights from social media conversations.
- Build an intelligent pipeline that can segregate patient conversations from the rest of the group.

Features

Header	Description
Source	Type of Social Media Post
Host	Domain of Social Media Post
Link	Complete URL of Post
Date	Date of Post
Time	Time stamp of post in Eastern Time
Time(GMT)	Time stamp of Post in GMT
Title	Title of the Post
Trans_Conv_Text	Actual Text Conversation of the Post
Patient_Tag	Patient Flag (1=Patient, 0=Non-Patient)

WORKFLOW OVERVIEW



DATA CLEANING

1. Irrelevant Features Removal:-

- Date and Timestamp have no relevance if they are not taken in local time as data is scraped from different social media platforms having different local times.
- Link and Host also do not provide any relevant information.
- Rows with missing labels are also of no real use to us.

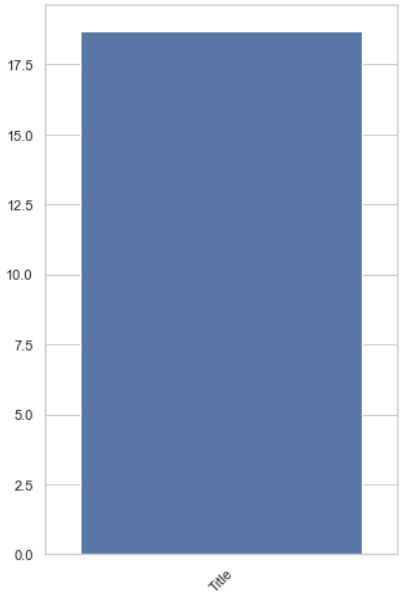
```
In [4]: training_data.drop(['Date(ET)', 'Time(ET)', 'time(GMT)', 'Link', 'Host'], axis=1, inplace=True)
        test_data.drop(['Date(ET)', 'Time(ET)', 'time(GMT)', 'Link', 'Host'], axis=1, inplace=True)

In [5]: training_data.dropna(axis=0, subset=['Patient_Tag'], inplace=True)
        all_columns = training_data.columns

In [6]: training_data.dropna(axis=0, subset=['TRANS_CONV_TEXT'], inplace=True)
```

DATA CLEANING

2. Missing Values Handling:-

```
training_data[training_data['Title'].isnull()].head(10)
```

	Source	Title	TRANS_CONV_TEXT	Patient_Tag
10	FORUMS	NaN	Looknsee Photography wrote: When I was living ...	1
18	FORUMS	NaN	Originally Posted by Birddog Cause of death wa...	0
25	BLOG	NaN	STILLWATER, Okla. (AP) ? Medical examiner spok...	0
34	BLOG	NaN	I am so, so, so congested. I am not feeling we...	1
35	Facebook	NaN	" Had lost a lot of weight then in no time st...	1
38	FACEBOOK	NaN	Matthew 25 31 When the Son of man shall come i...	1
40	Facebook	NaN	" I WUD LIKE TO THANK MY EMPLOYER FOR CANCELL...	1
49	FORUMS	NaN	Quote: Originally Posted by winston53660 Actua...	1
51	Facebook	NaN	"expi.co Nancy Davis Reagan, the widow of Pres...	0
53	FORUMS	NaN	Hi Pat, it's many, many years since I had my E...	1



There is no logical way of imputing titles into a social media post. So, those are replaced with blank space characters so that they can be filtered out later.

```
In [9]: training_data['Title'].fillna(' ', inplace=True);
        test_data['Title'].fillna(' ', inplace=True);
```

DATA CLEANING

3. HTML Tags and Punctuations removal :-

Since the data is scraped from web, there is a fair chance it has some of its web html embeddings transferred along with useful data. To get rid of those a simple function was implemented; which when applied to the dataframe, gets rid of those tags.

In [13]:
```
from bs4 import BeautifulSoup

def remove_html_tags(text):
    soup = BeautifulSoup(text, 'lxml')
    only_text = soup.get_text()
    return only_text
```

In [14]:
```
import string

def remove_punctuation(text):
    text_without_punctuation = ''.join([k for k in text if k not in string.punctuation])
    text_without_punctuation = text_without_punctuation
    return text_without_punctuation
```

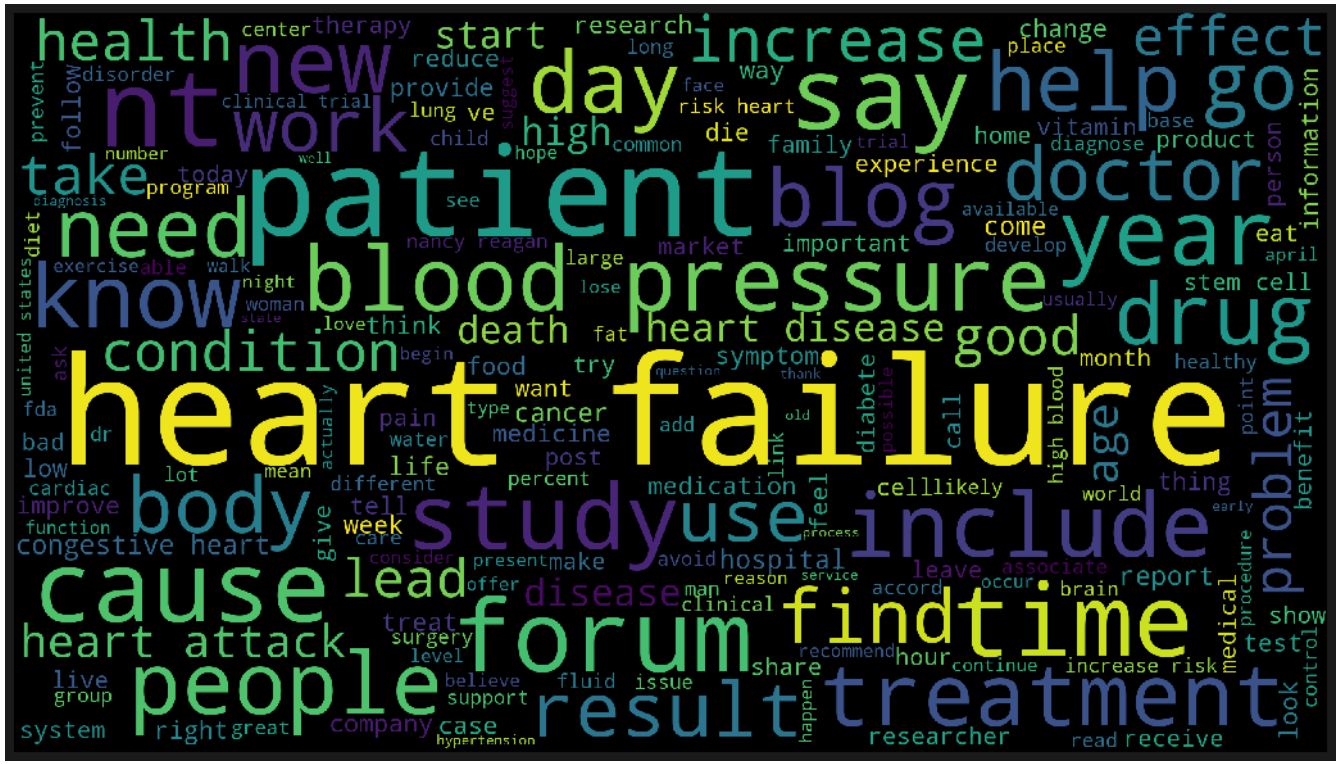
In [15]:
```
training_data['Combined_CONV_text'] = training_data['Combined_CONV_text'].apply(lambda x: remove_html_tags(x))
test_data['Combined_CONV_text'] = test_data['Combined_CONV_text'].apply(lambda x: remove_html_tags(x))

training_data['Combined_CONV_text'] = training_data['Combined_CONV_text'].apply(lambda x: remove_punctuation(x))
test_data['Combined_CONV_text'] = test_data['Combined_CONV_text'].apply(lambda x: remove_punctuation(x))
```

Inbuilt function of BeautifulSoup was used due to ease of use, robustness and speed compared to custom regular-expression filter.

EDA

1. Word Cloud:-
   - Word cloud helps us visualize the most frequent words in our corpus. It is a good indicator of how good the data set is and what targets can we set for our models.

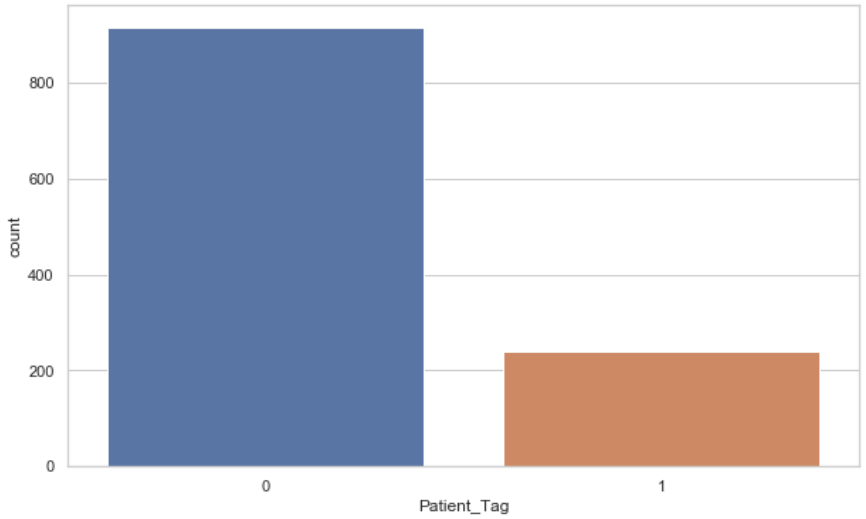
```
In [32]: from wordcloud import WordCloud

In [33]: text = ' '.join(X_train['Combined_CONV_text'])
         # text
         wordcloud = WordCloud(width=2560, height=1440).generate(text)

         barplot_dim = (20, 8)
         ax = plt.subplots(figsize=barplot_dim, facecolor='k')
         plt.imshow(wordcloud)
         plt.axis("off")
         plt.tight_layout(pad=0)
         plt.show()
```



Word cloud shows very relevant features to patients. This is a good indicator that the dataset is polished and relevant inferences can be easily targeted from the models.

EDA

2. Class Imbalance:-
   - This is a classification problem and a highly imbalanced class may result in improper training of the model.



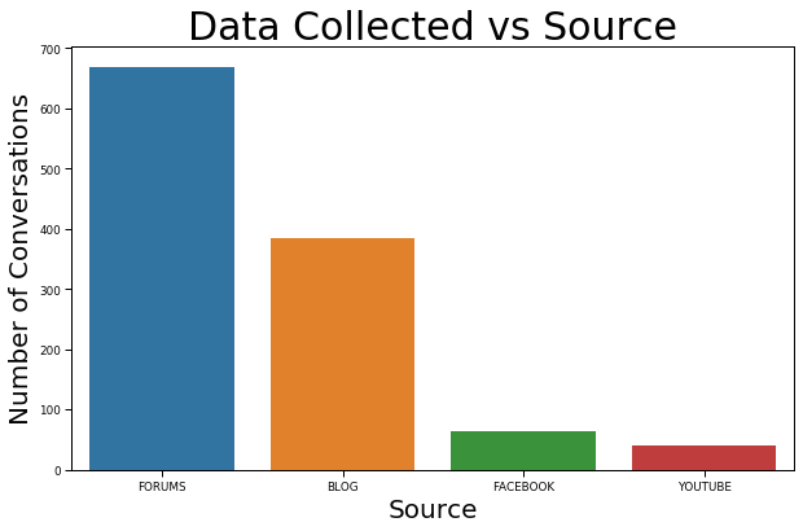
The class imbalance is not too bad.
If we classify all points as Not-Patients our Metrics would be:-
ROC = 0.5
Accuracy = 0.79
Precision = 0.63
Recall = 0.79
F1-Score = 0.70

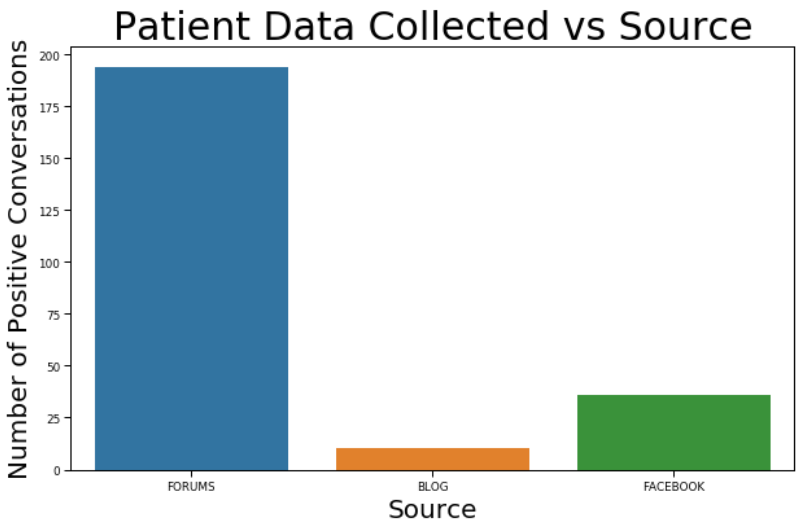
Decided performance metric and it's target based on data availability, relevance and class-imbalance.

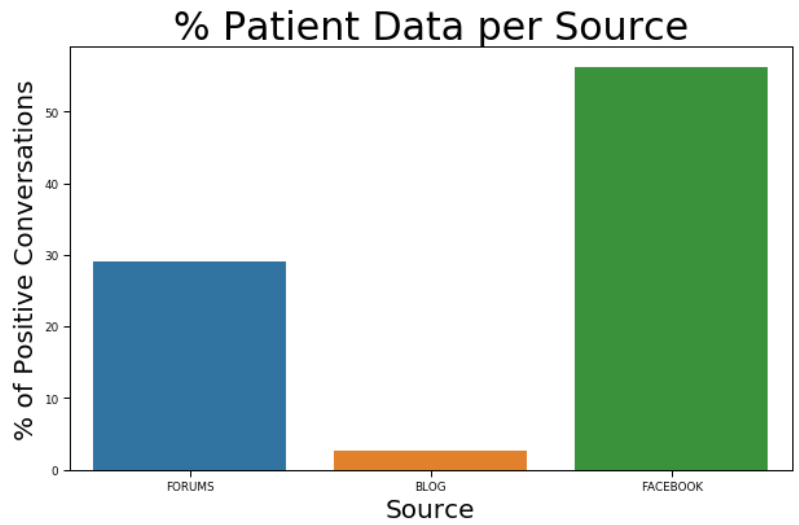
RECALL > 0.9-0.95

EDA

3. Social Media Strength:-
   - We can determine which type of social media has highest number and highest percentage of patient communication.



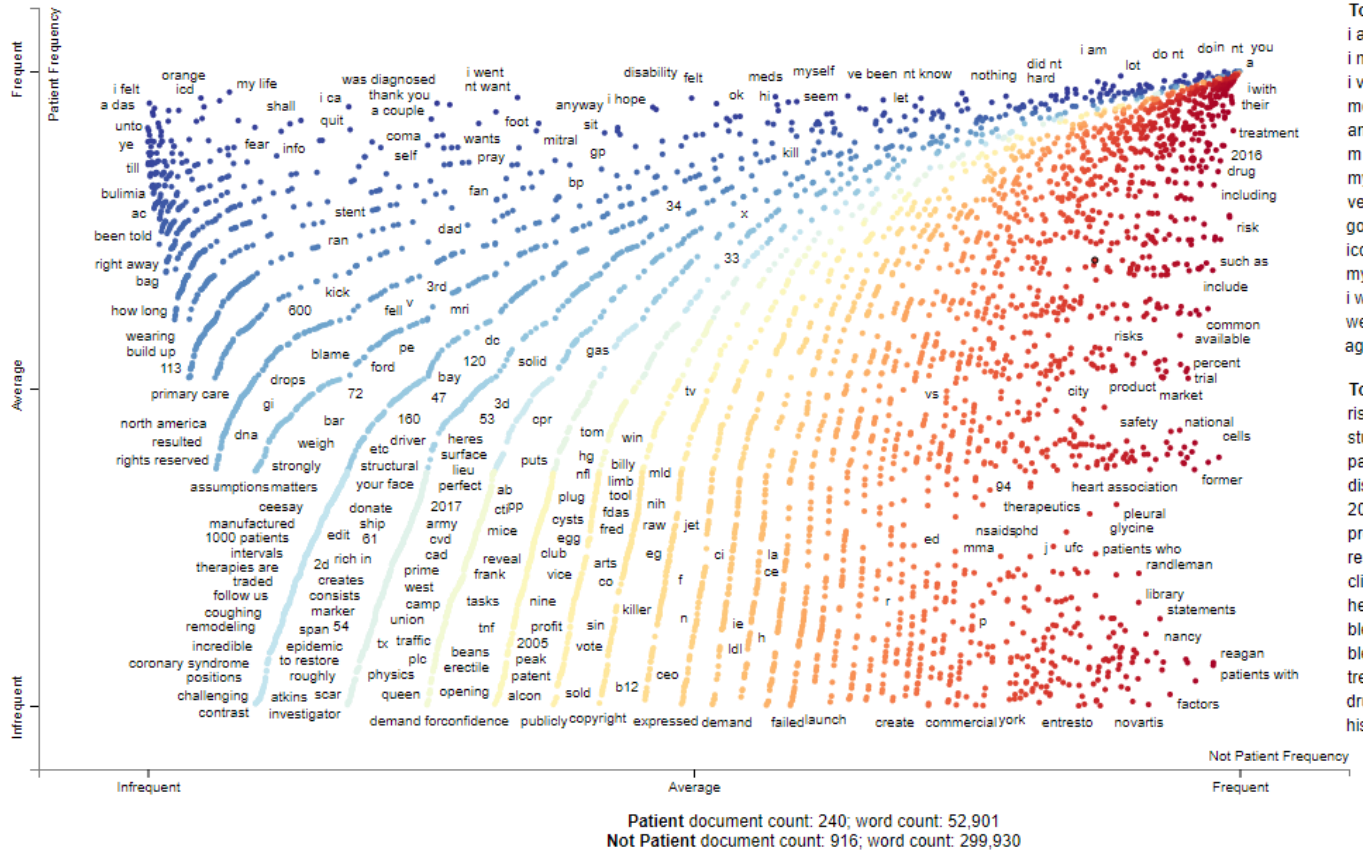




We can tell from this plot that, conversations from ‘YouTube’ are very unlikely to be from a patient.

EDA

4. Scattertext:-
   - We can determine which words affect the identification of patient the most.



Top Patient	Characteristic
i am	congestive
i m	lasix
i ve	randleman
me	forwardlooking
am	entresto
m	facebook
my life	ixmyelocelt
ve	reagans
got	saxagliptin
icd	alogliptin
my	youtube
i went	cardiologist
went	cardiomyopathy
agent orange	syncardia
	humira
Top Not Patient	
risk	apnea
study	lifethreatening
patients	twitter
disease	splitdose
2016	ejection
pressure	pleural
research	vericel
clinical	ixcelldcm
health	diastolic
blog	companys
blood pressure	eteplirsen
treatment	andor
drug	fibrillation
his	effusion
	yelchin

We can clearly identify terms like ‘I am’, ‘me’ etc are a good identifier of a person speaking about themselves. Doing ‘Stop-Word removal’, which is a very essential step of NLP might make the model worse due to loss of key information.

FEATURE ENGINEERING

1. Joining Title and Text:-
   - Title also consists of relevant description of text. So that can also be considered as part of conversation.
   - Source can be added into the same. Train-Validation split with stratified labels to avoid any leakage in further steps.

In [10]:
```
training_data['Combined_CONV_text'] = training_data['Title'] + '. ' + training_data['TRANS_CONV_TEXT'] + '. ' + training_data['Sc
test_data['Combined_CONV_text'] = test_data['Title'] + '. ' + test_data['TRANS_CONV_TEXT'] + '. ' + test_data['Source']

training_data.drop(['Title', 'TRANS_CONV_TEXT'], axis=1, inplace=True)
test_data.drop(['Title', 'TRANS_CONV_TEXT'], axis=1, inplace=True)
```

2. Text preparation:-
   - Converted text to lower case, numbers and extra white spaces are removed.

In [21]:
```
import re

def prepare_text(text):
    #convert all to lowercase
    text = str(text).lower()

    #remove numbers
    text = re.sub(r'\d+', '', text)

    #remove extra white spaces
    text = text.strip()

    return text
```

In [22]:
```
X_train['Combined_CONV_text'] = X_train['Combined_CONV_text'].apply(lambda x: prepare_text(x))
X_valid['Combined_CONV_text'] = X_valid['Combined_CONV_text'].apply(lambda x: prepare_text(x))
X_test['Combined_CONV_text'] = X_test['Combined_CONV_text'].apply(lambda x: prepare_text(x))
```

	Combined_CONV_text
848	versailles health care center celebrates natio...
489	exercise pathophysiology of heart failure pa...
262	right side heart valves slower thats what ive ...
550	heart failure survivor here diagnosed at when...
1101	irregular heart beats after a gram of coke las...
569	umusai on bridgehead customers tried to restra...
269	matthew then shall they also answer him sayi...
217	monitise share chat chat about moni shares s...
628	cardiologue hf avignon cardiologist h f ph...
826	anyone take blood pressure medication re anyon...

FEATURE ENGINEERING

3. Lemmatization:-
   - Converting text to their base Lemmas to reduce the dimensionality of word vectors.

In [26]:
```
def lemmatizer(text, nlp=nlp):
    doc = nlp(text)
    lemma_list = []
    for token in doc:
        lemma_list.append(token.lemma_)

    lemmatized_sentence = " ".join(lemma_list)

    return lemmatized_sentence
```

In [27]:
```
X_train['Combined_CONV_text'] = X_train['Combined_CONV_text'].apply(lambda x: lemmatizer(x))
X_valid['Combined_CONV_text'] = X_valid['Combined_CONV_text'].apply(lambda x: lemmatizer(x))
X_test['Combined_CONV_text'] = X_test['Combined_CONV_text'].apply(lambda x: lemmatizer(x))
```

	Combined_CONV_text
848	versailles health care center celebrate nation...
489	exercise pathophysiology of heart failure p...
262	right side heart valve slow that s what -PRON-...
550	heart failure survivor here diagnose at when...
1101	irregular heart beat after a gram of coke last...
569	umusai on bridgehead customer try to restrain ...
269	matthew then shall -PRON- also answer -PRON...
217	monitise share chat chat about moni share ...
628	cardiologue hf avignon cardiologist h f ...
826	anyone take blood pressure medication re anyon...

3. Vectorization:-
   - Various word vectorization methods were used; like TF-IDF, Count Vectorizer and a pre-trained vector model from Spacy. The one with best performance was chosen for the respective models.

MODELLING

Initially classical ML algorithms were employed with increasing order of complexity to benchmark their performance and gaze if Deep Learning is actually required for this scenario.

1. Logistic Regression
2. Multinomial Naive Bayes
3. K-Nearest Neighbours (KNN)
4. Support Vector Machine (SVM)
5. Random Forest
6. Extreme Gradient Boosting (XGBoost)

Algorithm	ROC-AUC	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.75	0.87	0.86	0.86	0.86
Multinomial NB	0.82	0.75	0.87	0.75	0.78
KNN	0.86	0.88	0.89	0.88	0.88
SVM	0.82	0.90	0.90	0.90	0.90
Random Forest	0.81	0.89	0.89	0.89	0.89
XGBoost	0.85	0.91	0.91	0.91	0.91

Hyper-Parameter tuning using Hyperopt was done on the Top-3 algorithms based on our performance metrics.

Hyper-Parameter tuning for all the selected algorithms was done with 3-fold stratified cross-validation on the training dataset for better generalisation and later check on validation dataset.

HYPER PARAMETER TUNING

After Hyper-Parameter tuning, the following performance were observed.

Algorithm	ROC-AUC	Accuracy	Precision	Recall	F1-Score
KNN	0.89	0.89	0.91	0.89	0.90
SVM	0.84	0.89	0.90	0.90	0.90
XGBoost	0.87	0.91	0.91	0.91	0.91

The target was achieved, but further exploration was necessary to see if it can better.
So, two more models were created using Deep Learning:-

1. Simple Dense Neural Network
2. Bi-LSTM based RNN model

Their various Hyper-Parameters like number of layers, number of nodes, batch size and learning rates were tuned using Tensorboard. Subsequently the threshold tuning was done manually to select the best threshold.

Algorithm	ROC-AUC	Accuracy	Precision	Recall	F1-Score
DNN	0.82	0.84	0.86	0.87	0.86
Transfer Learning	0.87	0.91	0.91	0.91	0.91

HYPER PARAMETER TUNING

Threshold tuning summary:-

Iterations	Batch Size	Best Threshold	Validation				
			Best ROC-AUC	Best Acc	Best Precision	Best Recall	Best F-Score
2	10	0.2	0.84	0.88	0.89	0.88	0.88
4	10	0.1	0.89	0.89	0.91	0.89	0.89
5	10	0.3	0.81	0.89	0.89	0.89	0.89
6	10	0.5	0.81	0.86	0.87	0.86	0.87
7	10	0.6	0.86	0.90	0.90	0.90	0.90
8	10	0.2	0.83	0.87	0.88	0.87	0.87
10	10	0.1	0.80	0.89	0.89	0.89	0.89
2	15	0.1	0.87	0.88	0.90	0.88	0.89
4	15	0.4	0.87	0.89	0.90	0.89	0.90
5	15	0.6	0.87	0.87	0.90	0.87	0.88
6	15	0.5	0.80	0.88	0.88	0.88	0.88
7	15	0.5	0.80	0.89	0.88	0.89	0.88
8	15	0.4	0.78	0.87	0.86	0.87	0.87
10	15	0.1	0.79	0.88	0.87	0.88	0.87
2	20	0.3	0.83	0.88	0.88	0.88	0.88
4	20	0.2	0.84	0.89	0.89	0.89	0.89
5	20	0.1	0.75	0.87	0.87	0.87	0.86
6	20	0.2	0.80	0.88	0.88	0.88	0.88
7	20	0.1	0.87	0.88	0.90	0.88	0.89
8	20	0.4	0.80	0.90	0.89	0.90	0.89
10	20	0.2	0.87	0.91	0.91	0.91	0.91

RECALL > 0.9-0.95

Based on our decided metric the last iteration was selected for the final model and used for submission.

Thank You!