Import all require modules

In [1]:

```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
import nltk
nltk.download('punkt') #for word tokenization
nltk.download('stopwords') #for removing or getting list of stopwords
nltk.download('wordnet') #for lemmatization
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\swapn\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\swapn\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\swapn\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[2]:

```
True
```

In [3]:

```
from nltk.tokenize import word_tokenize
from nltk.corpus import  stopwords
from nltk.stem import WordNetLemmatizer
import matplotlib.pyplot as plt
from wordcloud import WordCloud
```

In [4]:

```
df = pd.read_csv("google.csv")
```

In [5]:

```
df
```

Out[5]:

	Names	feedback	review-snippet	review-full-text
0	Komit Bagate	Positive:	NaN	NaN
1	omkar thorat	Positive:	I enrolled for data science course. I got to l...	I enrolled for data science course. I got to l...
2	Apoorva Shelar	Positive:	All the teaching staff and non-teaching staff ...	All the teaching staff and non-teaching staff ...
3	Shailesh Jadhav	NaN	IT vedant is very excellent institution, Becau...	IT vedant is very excellent institution, Becau...
4	Siddhi Lale	Positive:	NaN	NaN
...	...	...	...	...
281	HersheyOP	NaN	NaN	NaN
282	Yogesh Bhurawane	Critical:	NaN	NaN
283	Sunil Bhave	NaN	NaN	NaN
284	swapnil gondkar	NaN	NaN	NaN
285	shelar vaibhav	NaN	NaN	NaN

286 rows × 4 columns

In [6]:

```
df.drop(["review-snippet"],axis=1,inplace=True)
df.drop(["Names"],axis=1,inplace=True)
```

In [7]:

```
df
```

Out[7]:

	feedback	review-full-text
0	Positive:	NaN
1	Positive:	I enrolled for data science course. I got to l...
2	Positive:	All the teaching staff and non-teaching staff ...
3	NaN	IT vedant is very excellent institution, Becau...
4	Positive:	NaN
...	...	...
281	NaN	NaN
282	Critical:	NaN
283	NaN	NaN
284	NaN	NaN
285	NaN	NaN

286 rows × 2 columns

missing value treatment

In [8]:

```
#measure the nan(null) values in df
df.isnull().sum()
```

Out[8]:

```
feedback            186
review-full-text    213
dtype: int64
```

In [9]:

```
#percentage of nan(null) values in df
nullper=(df.isnull().sum()/len(df))*100
print(nullper)
```

```
feedback            65.034965
review-full-text    74.475524
dtype: float64
```

In [10]:

```
df.dropna(axis=0 ,inplace=True) #delete those rows who has NAN values
```

In [11]:

```
df                    #35 rows are non null it is not good for data
```

Out[11]:

	feedback	review-full-text
1	Positive:	I enrolled for data science course. I got to l...
2	Positive:	All the teaching staff and non-teaching staff ...
6	Positive:	I was looking forward to enhancing my technica...
7	Positive:	I joined I.T. Vedant in march 2020 to learn Py...
11	Positive:	I enrolled for data science course. I got to l...
12	Positive:	All the teaching staff and non-teaching staff ...
16	Positive:	I was looking forward to enhancing my technica...
17	Positive:	I joined I.T. Vedant in march 2020 to learn Py...
20	Positive:	Itvedant is a great institute to look for, if ...
21	Positive:	I have joined this institute on 3rd Sept. 2019...
22	Positive:	I just attended the counselling session and it...
23	Positive:	I am currently enrolled in Data Science course...
26	Positive:	I have enrolled for the Data science class in ...
28	Positive:	I joined ITVedant in December 2020. I enrolled...
31	Positive:	I had completed Data science course from Itved...
32	Positive:	Joined Itvedant on August 2020 , So here's my ...
33	Positive:	Itvedant have not only a student friendly envi...
34	Positive:	ItVedant is a amazing institute for different ...
37	Positive:	I joined IT Vedant for data science course.Tea...
38	Positive:	IT-Vedant is a very good Institute where you c...
40	Positive:	Hi, during lockdown I had so much of spare tim...
43	Positive:	I joined Itvedant few months back to enroll my...
45	Positive:	Firstly I really want to appreciate and thank ...
52	Positive:	IT Vedant is a good institute they actually he...
53	Positive:	Itvedant is the best place for learning variou...
54	Positive:	Itvedant is best training institute for IT cou...
55	Positive:	I would highly recommend ITVedant. The institu...
56	Positive:	Personal attention given by the faculty is why...
64	Positive:	Good institude .specially namrata mam provide...
67	Positive:	One of the best institute for python , dbms , ...
73	Positive:	I had enrolled for Data science basic course a...
78	Positive:	I joined ITVedant in December 2019. I enrolled...
85	Positive:	"Perfection & excellence way of teaching"".SHI...
88	Positive:	ITVedant gives me best in class career guidanc...

	feedback	review-full-text
207	Positive:	(Translated by Google) Meena mam, chetna mam, ...

In [12]:

```
df.info() #35 rows are non null
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 35 entries, 1 to 207
Data columns (total 2 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   feedback          35 non-null     object
 1   review-full-text  35 non-null     object
dtypes: object(2)
memory usage: 840.0+ bytes
```

In [13]:

```
df.isnull().sum()
```

Out[13]:

```
feedback            0
review-full-text    0
dtype: int64
```

In [14]:

```
# for each column, get value counts in decreasing order and take the index (value) of most
#df_most_common_imputed = df.apply(lambda x: x.fillna(x.value_counts().index[0]))
#df_most_common_imputed
```

In [15]:

```
#missing value treatment in categorical values using mode
df['feedback'] = df['feedback'].fillna(df['feedback'].mode()[0])
df['review-full-text'] = df['review-full-text'].fillna(df['review-full-text'].mode()[0])
```

In [16]:

```
df
```

Out[16]:

	feedback	review-full-text
1	Positive:	I enrolled for data science course. I got to l...
2	Positive:	All the teaching staff and non-teaching staff ...
6	Positive:	I was looking forward to enhancing my technica...
7	Positive:	I joined I.T. Vedant in march 2020 to learn Py...
11	Positive:	I enrolled for data science course. I got to l...
12	Positive:	All the teaching staff and non-teaching staff ...
16	Positive:	I was looking forward to enhancing my technica...
17	Positive:	I joined I.T. Vedant in march 2020 to learn Py...
20	Positive:	Itvedant is a great institute to look for, if ...
21	Positive:	I have joined this institute on 3rd Sept. 2019...
22	Positive:	I just attended the counselling session and it...
23	Positive:	I am currently enrolled in Data Science course...
26	Positive:	I have enrolled for the Data science class in ...
28	Positive:	I joined ITVedant in December 2020. I enrolled...
31	Positive:	I had completed Data science course from Itved...
32	Positive:	Joined Itvedant on August 2020 , So here's my ...
33	Positive:	Itvedant have not only a student friendly envi...
34	Positive:	ItVedant is a amazing institute for different ...
37	Positive:	I joined IT Vedant for data science course.Tea...
38	Positive:	IT-Vedant is a very good Institute where you c...
40	Positive:	Hi, during lockdown I had so much of spare tim...
43	Positive:	I joined Itvedant few months back to enroll my...
45	Positive:	Firstly I really want to appreciate and thank ...
52	Positive:	IT Vedant is a good institute they actually he...
53	Positive:	Itvedant is the best place for learning variou...
54	Positive:	Itvedant is best training institute for IT cou...
55	Positive:	I would highly recommend ITVedant. The institu...
56	Positive:	Personal attention given by the faculty is why...
64	Positive:	Good institude .specially namrata mam provide...
67	Positive:	One of the best institute for python , dbms , ...
73	Positive:	I had enrolled for Data science basic course a...
78	Positive:	I joined ITVedant in December 2019. I enrolled...
85	Positive:	"Perfection & excellence way of teaching"".SHI...
88	Positive:	ITVedant gives me best in class career guidanc...

	feedback	review-full-text
207	Positive:	(Translated by Google) Meena mam, chetna mam, ...

In [17]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
for col in df:
    df['feedback']=le.fit_transform(df['feedback'])
```

In [18]:

```
df
```

Out[18]:

	feedback	review-full-text
1	0	I enrolled for data science course. I got to l...
2	0	All the teaching staff and non-teaching staff ...
6	0	I was looking forward to enhancing my technica...
7	0	I joined I.T. Vedant in march 2020 to learn Py...
11	0	I enrolled for data science course. I got to l...
12	0	All the teaching staff and non-teaching staff ...
16	0	I was looking forward to enhancing my technica...
17	0	I joined I.T. Vedant in march 2020 to learn Py...
20	0	Itvedant is a great institute to look for, if ...
21	0	I have joined this institute on 3rd Sept. 2019...
22	0	I just attended the counselling session and it...
23	0	I am currently enrolled in Data Science course...
26	0	I have enrolled for the Data science class in ...
28	0	I joined ITVedant in December 2020. I enrolled...
31	0	I had completed Data science course from Itved...
32	0	Joined Itvedant on August 2020 , So here's my ...
33	0	Itvedant have not only a student friendly envi...
34	0	ItVedant is a amazing institute for different ...
37	0	I joined IT Vedant for data science course.Tea...
38	0	IT-Vedant is a very good Institute where you c...
40	0	Hi, during lockdown I had so much of spare tim...
43	0	I joined Itvedant few months back to enroll my...
45	0	Firstly I really want to appreciate and thank ...
52	0	IT Vedant is a good institute they actually he...
53	0	Itvedant is the best place for learning variou...
54	0	Itvedant is best training institute for IT cou...
55	0	I would highly recommend ITVedant. The institu...
56	0	Personal attention given by the faculty is why...
64	0	Good institude .specially namrata mam provide...
67	0	One of the best institute for python , dbms , ...
73	0	I had enrolled for Data science basic course a...
78	0	I joined ITVedant in December 2019. I enrolled...
85	0	"Perfection & excellence way of teaching"".SHI...
88	0	ITVedant gives me best in class career guidanc...

	feedback	review-full-text
207	0	(Translated by Google) Meena mam, chetna mam, ...

In [19]:

```
stop = stopwords.words("english")
def clean_text(text):
  tokens = word_tokenize(text.lower())
  # Filter only alphabets
  word_tokens = [t for t in tokens if t.isalpha()]
  clean_tokens = [t for t in word_tokens if t not in stop]
  lemma = WordNetLemmatizer()
  lemma_tokens = [lemma.lemmatize(t) for t in clean_tokens]
  return " ".join(lemma_tokens)
```

In [20]:

```
df['review-full-text'] = df['review-full-text'].apply(clean_text)
```

In [21]:

```
df['review-full-text'].head()
```

Out[21]:

```
1     enrolled data science course got learn many th...
2     teaching staff staff professional available as...
6     looking forward enhancing technical knowledge ...
7     joined vedant march learn python language hone...
11    enrolled data science course got learn many th...
Name: review-full-text, dtype: object
```

In [22]:

```
x= df['review-full-text']
y= df['feedback']
```

In [23]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)
```

In [24]:

```
sent_len = []
for t in df['review-full-text']:
  sent_len.append(len(word_tokenize(t)))
df['sent_len'] = sent_len
df.head()
```

Out[24]:

	feedback	review-full-text	sent_len
1	0	enrolled data science course got learn many th...	39
2	0	teaching staff staff professional available as...	32
6	0	looking forward enhancing technical knowledge ...	31
7	0	joined vedant march learn python language hone...	80
11	0	enrolled data science course got learn many th...	39

In [25]:

```
max(sent_len)
```

Out[25]:

```
96
```

In [26]:

```
np.quantile(sent_len, 0.95)
```

Out[26]:

```
80.0
```

In [27]:

```
max_len = 80
```

In [28]:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.preprocessing.text import Tokenizer
# Creates dictionary and every unique word is given number key
from tensorflow.keras.preprocessing import sequence
# To perform the padding of the documents with zero's to make the length of the
# document common
from tensorflow.keras.layers import (LSTM, Dropout, Embedding, SimpleRNN, GRU)
# All the index numbers are converted to vectors using Embedding
# SimpleRNN allows to implement the RNN architecture - activation function -tanh
# Dropout - manage overfitting of model
```

In [29]:

```
# Tokenization
tok = Tokenizer(char_level=False, split=" ")
tok.fit_on_texts(x_train)
```

In [30]:

```
tok.index_word
```

Out[30]:

```
{1: 'itvedant',
 2: 'mam',
 3: 'also',
 4: 'course',
 5: 'staff',
 6: 'knowledge',
 7: 'teaching',
 8: 'good',
 9: 'institute',
 10: 'would',
 11: 'experience',
 12: 'sir',
 13: 'help',
 14: 'data',
 15: 'science',
 16: 'best',
 17: 'faculty',
 18: 'concept',
```

In [31]:

```
vocab_len = len(tok.index_word)
vocab_len
```

Out[31]:

```
452
```

In [32]:

```
seq_train = tok.texts_to_sequences(x_train)
seq_train
```

Out[32]:

```
[[24,
  8,
  9,
  107,
  13,
  185,
  186,
  17,
  76,
  77,
  32,
  18,
  187,
  188,
  189,
  108,
  108,
  190.
```

In [33]:

```
seq_padded_train = sequence.pad_sequences(seq_train, maxlen=max_len)
seq_padded_train
```

Out[33]:

```
array([[  0,   0,   0, ...,  10, 109,  25],
       [  0,   0,   0, ...,  27, 203, 114],
       [  0,   0,   0, ..., 112, 120,  65],
       ...,
       [  0,   0,   0, ..., 178,  87,  20],
       [  0,   0,   0, ...,  27,  18,  68],
       [  0,   0,   0, ...,  35,  66, 136]])
```

In [34]:

```
model = Sequential()
# vectorization
model.add(Embedding(vocab_len+1,80, input_length=max_len, mask_zero=True))
# RNN layer
model.add(SimpleRNN(32, activation="tanh"))
# ANN's hidden layer
model.add(Dense(32, activation="relu"))
# To check on overfitting
model.add(Dropout(0.2))
# output layer
model.add(Dense(1, activation="sigmoid"))
```

In [35]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 80, 80)            36240
_________________________________________________________________
simple_rnn (SimpleRNN)       (None, 32)                3616
_________________________________________________________________
dense (Dense)                (None, 32)                1056
_________________________________________________________________
dropout (Dropout)            (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 33
=================================================================
Total params: 40,945
Trainable params: 40,945
Non-trainable params: 0
_________________________________________________________________
```

In [36]:

```
model.compile(loss="binary_crossentropy", optimizer="adam")
```

In [37]:

```
model.fit(seq_padded_train, y_train, batch_size=50, epochs=50)
```

```
Epoch 1/50
1/1 [==============================] - 2s 2s/step - loss: 0.7059
Epoch 2/50
1/1 [==============================] - 0s 24ms/step - loss: 0.6281
Epoch 3/50
1/1 [==============================] - 0s 16ms/step - loss: 0.5646
Epoch 4/50
1/1 [==============================] - 0s 16ms/step - loss: 0.5087
Epoch 5/50
1/1 [==============================] - 0s 16ms/step - loss: 0.4624
Epoch 6/50
1/1 [==============================] - 0s 16ms/step - loss: 0.4177
Epoch 7/50
1/1 [==============================] - 0s 24ms/step - loss: 0.3525
Epoch 8/50
1/1 [==============================] - 0s 16ms/step - loss: 0.3031
Epoch 9/50
1/1 [==============================] - 0s 16ms/step - loss: 0.3182
Epoch 10/50
1/1 [==============================] - 0s 24ms/step - loss: 0.2506
Epoch 11/50
1/1 [==============================] - 0s 24ms/step - loss: 0.2430
Epoch 12/50
1/1 [==============================] - 0s 16ms/step - loss: 0.2372
Epoch 13/50
1/1 [==============================] - 0s 16ms/step - loss: 0.1934
Epoch 14/50
1/1 [==============================] - 0s 24ms/step - loss: 0.2071
Epoch 15/50
1/1 [==============================] - 0s 24ms/step - loss: 0.1508
Epoch 16/50
1/1 [==============================] - 0s 16ms/step - loss: 0.1661
Epoch 17/50
1/1 [==============================] - 0s 16ms/step - loss: 0.1233
Epoch 18/50
1/1 [==============================] - 0s 24ms/step - loss: 0.1135
Epoch 19/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0986
Epoch 20/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0884
Epoch 21/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0786
Epoch 22/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0788
Epoch 23/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0613
Epoch 24/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0849
Epoch 25/50
1/1 [==============================] - ETA: 0s - loss: 0.059 - 0s 16ms/step
- loss: 0.0597
Epoch 26/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0685
Epoch 27/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0516
Epoch 28/50
```

```
1/1 [==============================] - 0s 24ms/step - loss: 0.0424
Epoch 29/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0426
Epoch 30/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0340
Epoch 31/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0349
Epoch 32/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0374
Epoch 33/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0215
Epoch 34/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0314
Epoch 35/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0178
Epoch 36/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0236
Epoch 37/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0211
Epoch 38/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0288
Epoch 39/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0198
Epoch 40/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0175
Epoch 41/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0129
Epoch 42/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0155
Epoch 43/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0156
Epoch 44/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0120
Epoch 45/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0103
Epoch 46/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0117
Epoch 47/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0105
Epoch 48/50
1/1 [==============================] - 0s 24ms/step - loss: 0.0120
Epoch 49/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0128
Epoch 50/50
1/1 [==============================] - 0s 16ms/step - loss: 0.0081
```

Out[37]:

```
<tensorflow.python.keras.callbacks.History at 0x1b6ed27ed00>
```

In [38]:

```
seq_test = tok.texts_to_sequences(x_test)
seq_test
```

```
  23,
  5,
  31,
  32,
  45,
  31,
  21,
  176,
  54,
  59,
  87,
  54,
  214,
  61,
  104,
  11,
  31,
  52,
  100,
  323
```

In [39]:

```
seq_padded_test = sequence.pad_sequences(seq_test, maxlen=max_len)
seq_padded_test
```

Out[39]:

```
array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,  46,   1, 278, 279,  19, 304, 305,  47, 280,
         54, 281, 282, 283,  54,  71,  81,  23,   5,  31,  32,  45,  31,
         21, 176,  54,  59,  87,  54, 214,  61, 104,  11,  31,  52, 100,
        323, 158],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 172, 402, 174,
        115,   6, 181,  46,   1,   9, 403, 404,  19,  33,  99,   7,   5,
         55,   3,  17,  56,  26,  60,  68,  76,  22,  69,  59,  87, 405,
         30,  11],
       [ 46,  24, 373,  39,  33, 103, 175, 374,  48, 375,   8, 376, 161,
         17,  74, 377, 106,  31,   8,  16, 164,  21, 378, 176,  54, 379,
         17, 177,  42,  71, 178,  33, 380, 116,  21, 175, 381, 171, 382,
         74, 383,  75, 384,  23, 156, 385, 386,  10,  97,  75, 387, 388,
        389, 390, 391, 392,  29, 393, 394, 179,  24, 395, 146, 147,  95,
        176,  24,  35,  66, 396, 397, 398, 399,  58, 180, 143,  24,  54,
        400, 401],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,  19,  14,  15,   4,  35,  39,  79,  36,   4,   4, 444,
        445, 183, 446, 447,  23, 173,  69, 183,   7,  17,  56, 117,   3,
         51,   7,   5,   3,  64,   6,   3, 448,  22,   6,  60,  59, 178,
         87,  20],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,  16,   9,  33, 382,  12,  16,   5,   5,
         56,  32],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,  46,   1, 403,  33,  99,   4,  95, 181,  16,   7,   5,
         10,  29, 154,  40,  95, 132,  31, 422, 201,  49,  73,  25,   1,
         52,  92],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
        314,  61,  17,  10,  25,  24, 418, 424, 100, 422,  18,  84,  87,
        405,   8,   4,  39, 190,   5,  30,  80, 101,  12, 150,  12,  50,
         12, 165],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   1, 343,  16,  42,  92, 367,   3,  66,
```

```
 280,  72, 411,  12, 418,   3,  72,  12, 287,  12, 101,  30,  21,
  11,   1],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,  46,   9,  30,  11,   9,  21,  35,  22,  18, 316,
   5,   6,   4, 261,  21,  37,  62,  26, 169,   3, 261,  81, 405,
 316,   9],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   1,  30,   9, 366,  52, 141,  18,
  71,  17,  55,   7,  30,  57,  16, 164,  22,   6,  13,  18, 343,
  87, 345],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0, 367,  95, 104,  51, 108, 422,  21,  48, 353,  24,  21,  47,
 323,   7,  51,   7,   5,  28, 154, 148, 101,  12,   7,  16,  57,
 422,  39]])
```

In [40]:

```
y_hat = model.predict(seq_padded_test)
```

In [41]:

```
# y_hat contains probability
y_hat = np.where(y_hat>=0.5, 1, 0)
```

In [42]:

```
from sklearn.metrics import classification_report
print(classification_report( y_test, y_hat))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11

    accuracy                           1.00        11
   macro avg       1.00      1.00      1.00        11
weighted avg       1.00      1.00      1.00        11
```

In [43]:

```
model = Sequential()
# vectorization
model.add(Embedding(vocab_len+1,80, input_length=max_len, mask_zero=True))
# RNN layer
# model.add(SimpleRNN(32, activation="tanh"))
model.add(LSTM(32, activation="tanh"))
# ANN's hidden layer
model.add(Dense(32, activation="relu"))
# To check on overfitting
model.add(Dropout(0.2))
# output layer
model.add(Dense(1, activation="sigmoid"))
model.compile(loss="binary_crossentropy", optimizer="adam")
model.fit(seq_padded_train, y_train, batch_size=50, epochs=50)
```

```
Epoch 1/50
1/1 [==============================] - 5s 5s/step - loss: 0.6984
Epoch 2/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6914
Epoch 3/50
1/1 [==============================] - 0s 40ms/step - loss: 0.6872
Epoch 4/50
1/1 [==============================] - 0s 40ms/step - loss: 0.6833
Epoch 5/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6794
Epoch 6/50
1/1 [==============================] - 0s 40ms/step - loss: 0.6744
Epoch 7/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6682
Epoch 8/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6623
Epoch 9/50
1/1 [==============================] - 0s 24ms/step - loss: 0.6592
Epoch 10/50
1/1 [==============================] - 0s 24ms/step - loss: 0.6519
Epoch 11/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6443
Epoch 12/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6371
Epoch 13/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6258
Epoch 14/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6184
Epoch 15/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6093
Epoch 16/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5936
Epoch 17/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5808
Epoch 18/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5654
Epoch 19/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5542
Epoch 20/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5381
Epoch 21/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5213
Epoch 22/50
```

```
1/1 [==============================] - 0s 32ms/step - loss: 0.4892
Epoch 23/50
1/1 [==============================] - 0s 40ms/step - loss: 0.4581
Epoch 24/50
1/1 [==============================] - 0s 32ms/step - loss: 0.4081
Epoch 25/50
1/1 [==============================] - 0s 32ms/step - loss: 0.3893
Epoch 26/50
1/1 [==============================] - 0s 32ms/step - loss: 0.3336
Epoch 27/50
1/1 [==============================] - 0s 32ms/step - loss: 0.2916
Epoch 28/50
1/1 [==============================] - 0s 40ms/step - loss: 0.2449
Epoch 29/50
1/1 [==============================] - 0s 32ms/step - loss: 0.2177
Epoch 30/50
1/1 [==============================] - 0s 32ms/step - loss: 0.1736
Epoch 31/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0958
Epoch 32/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0836
Epoch 33/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0872
Epoch 34/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0512
Epoch 35/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0602
Epoch 36/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0458
Epoch 37/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0290
Epoch 38/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0248
Epoch 39/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0150
Epoch 40/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0182
Epoch 41/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0259
Epoch 42/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0235
Epoch 43/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0161
Epoch 44/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0100
Epoch 45/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0114
Epoch 46/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0102
Epoch 47/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0165
Epoch 48/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0078
Epoch 49/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0075
Epoch 50/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0078
```

Out[43]:

```
<tensorflow.python.keras.callbacks.History at 0x1b6f7a7bc70>
```

In [44]:

```
y_hat = model.predict(seq_padded_test)
y_hat = np.where(y_hat>=0.5, 1, 0)
print(classification_report(y_test, y_hat))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11

    accuracy                           1.00        11
   macro avg       1.00      1.00      1.00        11
weighted avg       1.00      1.00      1.00        11
```

In [45]:

```
model = Sequential()
# vectorization
model.add(Embedding(vocab_len+1,80, input_length=max_len, mask_zero=True))
# RNN layer
model.add(GRU(32, activation="tanh"))
# ANN's hidden layer
model.add(Dense(32, activation="relu"))
# To check on overfitting|
model.add(Dropout(0.2))
# output layer
model.add(Dense(1, activation="sigmoid"))
model.compile(loss="binary_crossentropy", optimizer="adam")
model.fit(seq_padded_train, y_train, batch_size=50, epochs=50)
```

```
Epoch 1/50
1/1 [==============================] - 5s 5s/step - loss: 0.6966
Epoch 2/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6919
Epoch 3/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6844
Epoch 4/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6797
Epoch 5/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6710
Epoch 6/50
1/1 [==============================] - 0s 40ms/step - loss: 0.6617
Epoch 7/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6542
Epoch 8/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6519
Epoch 9/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6446
Epoch 10/50
1/1 [==============================] - 0s 24ms/step - loss: 0.6311
Epoch 11/50
1/1 [==============================] - 0s 24ms/step - loss: 0.6197
Epoch 12/50
1/1 [==============================] - 0s 32ms/step - loss: 0.6198
Epoch 13/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5993
Epoch 14/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5958
Epoch 15/50
1/1 [==============================] - 0s 24ms/step - loss: 0.5812
Epoch 16/50
1/1 [==============================] - 0s 24ms/step - loss: 0.5577
Epoch 17/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5530
Epoch 18/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5333
Epoch 19/50
1/1 [==============================] - 0s 32ms/step - loss: 0.5127
Epoch 20/50
1/1 [==============================] - 0s 32ms/step - loss: 0.4968
Epoch 21/50
1/1 [==============================] - 0s 32ms/step - loss: 0.4662
Epoch 22/50
1/1 [==============================] - 0s 32ms/step - loss: 0.4587
```

```
Epoch 23/50
1/1 [==============================] - 0s 32ms/step - loss: 0.4306
Epoch 24/50
1/1 [==============================] - 0s 32ms/step - loss: 0.4222
Epoch 25/50
1/1 [==============================] - 0s 32ms/step - loss: 0.3975
Epoch 26/50
1/1 [==============================] - 0s 32ms/step - loss: 0.3636
Epoch 27/50
1/1 [==============================] - 0s 32ms/step - loss: 0.3346
Epoch 28/50
1/1 [==============================] - 0s 40ms/step - loss: 0.3125
Epoch 29/50
1/1 [==============================] - 0s 32ms/step - loss: 0.2654
Epoch 30/50
1/1 [==============================] - 0s 32ms/step - loss: 0.2420
Epoch 31/50
1/1 [==============================] - 0s 32ms/step - loss: 0.2191
Epoch 32/50
1/1 [==============================] - 0s 32ms/step - loss: 0.1869
Epoch 33/50
1/1 [==============================] - 0s 24ms/step - loss: 0.1301
Epoch 34/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0990
Epoch 35/50
1/1 [==============================] - 0s 32ms/step - loss: 0.1037
Epoch 36/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0769
Epoch 37/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0564
Epoch 38/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0381
Epoch 39/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0383
Epoch 40/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0302
Epoch 41/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0227
Epoch 42/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0094
Epoch 43/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0174
Epoch 44/50
1/1 [==============================] - 0s 40ms/step - loss: 0.0065
Epoch 45/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0138
Epoch 46/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0068
Epoch 47/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0041
Epoch 48/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0091
Epoch 49/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0056
Epoch 50/50
1/1 [==============================] - 0s 32ms/step - loss: 0.0033
```

Out[45]:

```
<tensorflow.python.keras.callbacks.History at 0x1b6813d18e0>
```

In [46]:

```
y_hat = model.predict(seq_padded_test)
y_hat = np.where(y_hat>=0.5, 1, 0)
print(classification_report(y_test, y_hat))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11

    accuracy                           1.00        11
   macro avg       1.00      1.00      1.00        11
weighted avg       1.00      1.00      1.00        11
```

In [ ]:

```

```