DATE:26-10-2023

PHASE-4
DEVELOPMENT PART - II

AIM:

The aim of Phase 4 of the project is to continue the development of a fake news detection model using Natural Language Processing (NLP) techniques. Specifically, the focus is on text preprocessing, feature extraction, model training, and evaluation. The primary goal is to build and fine-tune a classification model capable of distinguishing between fake and real news articles.

PREPROCESSING:

REMOVING NULL VALUES:

(from the information above there is no null values so nothing is removed)

```
[6]: #preprocessing
     #drop null values
     data=data.dropna(axis=0)

[7]: len(data)

[7]: 44898
```

CONVERTING ALL STRINGS TO LOWERCASE:

```
[8]: #converting all strings to lowercase
     data['clean_news']=data['text'].str.lower()
     data['clean_news']

[8]: 0        bangkok (reuters) - rights groups on wednesday...
     1        on friday s broadcast of hbo s  real time,  fo...
     2        21st century wire says regardless of what one ...
     3        (reuters) - u.s. president-elect donald trump ...
     4        the  hard working  first family, in need of an...
                                    ...
     44893    21st century wire says does the american ideal...
     44894    barinas, venezuela (reuters) - tirelessly trav...
     44895    phnom penh (reuters) - cambodian prime ministe...
     44896    geneva (reuters) - the united states wants to ...
     44897    beijing (reuters) - u.s. president donald trum...
     Name: clean_news, Length: 44898, dtype: object
```

REMOVING SPECIAL CHARACTERS , EXTRA SPACES AND ESCAPE CHARACTERS

```
[9]: #removing special characters , extra spaces and escape characters
     data['clean_news']=data['clean_news'].str.replace('[^A-Za-z0-9\s]','')
     data['clean_news']=data['clean_news'].str.replace('[\n]','')
     data['clean_news']=data['clean_news'].str.replace('[\s+]',' ')
     data['clean_news']
```

```
[9]:  0        bangkok (reuters) - rights groups on wednesday...
      1        on friday s broadcast of hbo s  real time,  fo...
      2        21st century wire says regardless of what one ...
      3        (reuters) - u.s. president-elect donald trump ...
      4        the  hard working  first family, in need of an...
                                    ...
      44893    21st century wire says does the american ideal...
      44894    barinas, venezuela (reuters) - tirelessly trav...
      44895    phnom penh (reuters) - cambodian prime ministe...
      44896    geneva (reuters) - the united states wants to ...
      44897    beijing (reuters) - u.s. president donald trum...
      Name: clean_news, Length: 44898, dtype: object
```

REMOVING STOP WORDS

```
[10]: nltk.download('stopwords')
```

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

```
[10]: True
```

```
[11]: #remove stop words
      from nltk.corpus import stopwords
      stop=stopwords.words('english')
      data['clean_news']=data['clean_news'].apply(lambda x: " ".join([word for word in x.split() if word not in stop]))
      data.head()
```

[11]:

	title	text	subject	date	Target	clean_news
0	Rights groups urge EU, Japan to consider halt ...	BANGKOK (Reuters) - Rights groups on Wednesday...	worldnews	October 18, 2017	True	bangkok (reuters) - rights groups wednesday ur...
1	WATCH: IRRELEVANT DEM POLITICAL ANALYST James ...	On Friday s broadcast of HBO s Real Time, fo...	left-news	Oct 21, 2017	Fake	friday broadcast hbo real time, former clinton...
2	Trump Asks O'Reilly, 'Do you think our country...	21st Century Wire says Regardless of what one ...	US_News	February 6, 2017	Fake	21st century wire says regardless one thinks d...
3	Factbox: Trump fills top jobs for his administ...	(Reuters) - U.S. President-elect Donald Trump ...	politicsNews	December 5, 2016	True	(reuters) - u.s. president-elect donald trump ...
4	ONE LAST TIME ON OUR DIME: Mooch and Barack Ar...	The hard working First Family, in need of an...	politics	Aug 6, 2016	Fake	hard working first family, need another taxpay...

TOKENIZATION:

```
[13]: nltk.download('punkt')
```

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

```
[13]: True
```

```
[14]: #Tokenization
      from nltk.tokenize import word_tokenize
      data['tokenized_news'] = data['clean_news'].apply(lambda x: word_tokenize(x))
      data.head()
```

[14]:

	title	text	subject	date	Target	clean_news	tokenized_news
0	Rights groups urge EU, Japan to consider halt ...	BANGKOK (Reuters) - Rights groups on Wednesday...	worldnews	October 18, 2017	True	bangkok (reuters) - rights groups wednesday ur...	[bangkok, (, reuters,), -, rights, groups, we...
1	WATCH: IRRELEVANT DEM POLITICAL ANALYST James ...	On Friday s broadcast of HBO s Real Time, fo...	left-news	Oct 21, 2017	Fake	friday broadcast hbo real time, former clinton...	[friday, broadcast, hbo, real, time, ,, former...
2	Trump Asks O'Reilly, 'Do you think our country...	21st Century Wire says Regardless of what one ...	US_News	February 6, 2017	Fake	21st century wire says regardless one thinks d...	[21st, century, wire, says, regardless, one, t...
3	Factbox: Trump fills top jobs for his administ...	(Reuters) - U.S. President-elect Donald Trump ...	politicsNews	December 5, 2016	True	(reuters) - u.s. president-elect donald trump ...	[(, reuters,), -, u.s., president-elect, dona...
4	ONE LAST TIME ON OUR DIME: Mooch and Barack Ar...	The hard working First Family, in need of an...	politics	Aug 6, 2016	Fake	hard working first family, need another taxpay...	[hard, working, first, family, ,, need, anothe...

LEMMATIZATION:

[15]:
```
nltk.download('wordnet')
```

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

[15]:
```
True
```

[16]:
```
#Lemmatization
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
def lemmatize_text(tokens, lemmatizer):
    return [lemmatizer.lemmatize(token) for token in tokens]
data['lemmatized_news'] = data['tokenized_news'].apply(lambda x: lemmatize_text(x, lemmatizer))
data.head()
```

[16]:

	title	text	subject	date	Target	clean_news	tokenized_news	lemmatized_news
0	Rights groups urge EU, Japan to consider halt ...	BANGKOK (Reuters) - Rights groups on Wednesday...	worldnews	October 18, 2017	True	bangkok (reuters) - rights groups wednesday ur...	[bangkok, (, reuters,), -, rights, groups, we...	[bangkok, (, reuters,), -, right, group, wedn...
1	WATCH: IRRELEVANT DEM POLITICAL ANALYST James ...	On Friday s broadcast of HBO s Real Time, fo...	left-news	Oct 21, 2017	Fake	friday broadcast hbo real time, former clinton...	[friday, broadcast, hbo, real, time, ,, former...	[friday, broadcast, hbo, real, time, ,, former...
2	Trump Asks O'Reilly, 'Do you think our country...	21st Century Wire says Regardless of what one ...	US_News	February 6, 2017	Fake	21st century wire says regardless one thinks d...	[21st, century, wire, says, regardless, one, t...	[21st, century, wire, say, regardless, one, th...
3	Factbox: Trump fills top jobs for his administ...	(Reuters) - U.S. President-elect Donald Trump ...	politicsNews	December 5, 2016	True	(reuters) - u.s. president-elect donald trump ...	[(, reuters,), -, u.s., president-elect, dona...	[(, reuters,), -, u.s., president-elect, dona...
4	ONE LAST TIME ON OUR DIME: Mooch and Barack Ar...	The hard working First Family, in need of an...	politics	Aug 6, 2016	Fake	hard working first family, need another taxpay...	[hard, working, first, family, ,, need, anothe...	[hard, working, first, family, ,, need, anothe...

CREATE SENTENCES TO GET CLEAN TEXT AS INPUT FOR VECTORS

[17]:
```
def return_sentences(tokenized_news):
    return " ".join([word for word in tokenized_news])
```

[18]:
```
data['clean_text'] = data['lemmatized_news'].apply(lambda x : return_sentences(x))
data.head()
```

[18]:

	title	text	subject	date	Target	clean_news	tokenized_news	lemmatized_news	clean_text
0	Rights groups urge EU, Japan to consider halt ...	BANGKOK (Reuters) - Rights groups on Wednesday...	worldnews	October 18, 2017	True	bangkok (reuters) - rights groups wednesday ur...	[bangkok, (, reuters,), -, rights, groups, we...	[bangkok, (, reuters,), -, right, group, wedn...	bangkok (reuters) - right group wednesday ur...
1	WATCH: IRRELEVANT DEM POLITICAL ANALYST James ...	On Friday s broadcast of HBO s Real Time, fo...	left-news	Oct 21, 2017	Fake	friday broadcast hbo real time, former clinton...	[friday, broadcast, hbo, real, time, ,, former...	[friday, broadcast, hbo, real, time, ,, former...	friday broadcast hbo real time , former clinto...
2	Trump Asks O'Reilly, 'Do you think our country...	21st Century Wire says Regardless of what one ...	US_News	February 6, 2017	Fake	21st century wire says regardless one thinks d...	[21st, century, wire, says, regardless, one, t...	[21st, century, wire, say, regardless, one, th...	21st century wire say regardless one think don...
3	Factbox: Trump fills top jobs for his administ...	(Reuters) - U.S. President-elect Donald Trump ...	politicsNews	December 5, 2016	True	(reuters) - u.s. president-elect donald trump ...	[(, reuters,), -, u.s., president-elect, dona...	[(, reuters,), -, u.s., president-elect, dona...	(reuters) - u.s. president-elect donald trum...
4	ONE LAST TIME ON OUR DIME: Mooch and Barack Ar...	The hard working First Family, in need of an...	politics	Aug 6, 2016	Fake	hard working first family, need another taxpay...	[hard, working, first, family, ,, need, anothe...	[hard, working, first, family, ,, need, anothe...	hard working first family , need another taxpa...

PREPARE DATA FOR THE MODEL. CONVERT LABEL IN TO BINARY

[20]:
```
data['Target'] = [1 if x == 'Fake' else 0 for x in data['Target']]
data.head()
```

[20]:

	title	text	subject	date	Target	clean_news	tokenized_news	lemmatized_news	clean_text
0	Rights groups urge EU, Japan to consider halt ...	BANGKOK (Reuters) - Rights groups on Wednesday...	worldnews	October 18, 2017	0	bangkok (reuters) - rights groups wednesday ur...	[bangkok, (, reuters,), -, rights, groups, we...	[bangkok, (, reuters,), -, right, group, wedn...	bangkok (reuters) - right group wednesday ur...
1	WATCH: IRRELEVANT DEM POLITICAL ANALYST James ...	On Friday s broadcast of HBO s Real Time, fo...	left-news	Oct 21, 2017	1	friday broadcast hbo real time, former clinton...	[friday, broadcast, hbo, real, time, ,, former...	[friday, broadcast, hbo, real, time, ,, former...	friday broadcast hbo real time , former clinto...
2	Trump Asks O'Reilly, 'Do you think our country...	21st Century Wire says Regardless of what one ...	US_News	February 6, 2017	1	21st century wire says regardless one thinks d...	[21st, century, wire, says, regardless, one, t...	[21st, century, wire, say, regardless, one, th...	21st century wire say regardless one think don...
3	Factbox: Trump fills top jobs for his administ...	(Reuters) - U.S. President-elect Donald Trump ...	politicsNews	December 5, 2016	0	(reuters) - u.s. president-elect donald trump ...	[(, reuters,), -, u.s., president-elect, dona...	[(, reuters,), -, u.s., president-elect, dona...	(reuters) - u.s. president-elect donald trum...
4	ONE LAST TIME ON OUR DIME: Mooch and Barack Ar...	The hard working First Family, in need of an...	politics	Aug 6, 2016	1	hard working first family, need another taxpay...	[hard, working, first, family, ,, need, anothe...	[hard, working, first, family, ,, need, anothe...	hard working first family , need another taxpa...

SPLIT THE DATASET

```
[21]: from sklearn.model_selection import train_test_split
```

```
[22]: X_train, X_test, y_train, y_test = train_test_split(data['clean_text'], data['Target'], test_size=0.2, random_state=5)

print(X_train.shape)
print(X_test.shape)
```

```
(35918,)
(8980,)
```

FEATURE EXTRACTION
(words to vectors)

Count vectorizer which considers the frequency of occurrence of a word across the corpus.

```
[23]: from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

# Assuming 'lemmatized_news' is a column in your DataFrame 'data'
count_vectorizer = CountVectorizer()
X = count_vectorizer.fit_transform(data['clean_text'])

# Get feature names
feature_names_count = count_vectorizer.get_feature_names_out()

print("CountVectorizer feature names:", feature_names_count)
```

```
CountVectorizer feature names: ['00' '000' '0000' ... 'zzzzzzzz' 'zzzzzzzzzzzz' 'émigré']
```

TF-IDF : TERM FREQUENCY - INVERSE DOCUMENT FREQUENCY

The term frequency is the number of times a term occurs in a document. Inverse document frequency is an inverse function of the number of documents in which that a given word occurs.

The product of these two terms gives tf-idf weight for a word in the corpus.

```
[26]: tfidf_vectorizer = TfidfVectorizer()
X_tfidf = tfidf_vectorizer.fit_transform(data['clean_text'])
feature_names_tfidf = tfidf_vectorizer.get_feature_names_out()
print("TfidfVectorizer feature names:", feature_names_tfidf)
```

```
TfidfVectorizer feature names: ['00' '000' '0000' ... 'zzzzzzzz' 'zzzzzzzzzzzz' 'émigré']
```

```
[27]: tfidf = TfidfVectorizer()
tfidf_train = tfidf.fit_transform(X_train)
tfidf_test = tfidf.transform(X_test)

print(tfidf_train.shape)
print(tfidf_test.shape)
```

```
(35918, 106465)
(8980, 106465)
```

CREATE WORD EMBEDDINGS WITH GLOVE FILE:

```
[ ] from sklearn.metrics import classification_report, confusion_matrix, accuracy_score,roc_auc_score
    from keras.models import Model
    from keras.layers import Dense,Embedding,Input,LSTM, Bidirectional,GlobalMaxPool1D,Dropout
    from keras.preprocessing.text import Tokenizer
    from keras.preprocessing.sequence import pad_sequences
    from keras import Sequential
```

```
[ ]  EMBEDDING_FILE=r"/content/drive/MyDrive/AI_Phase3/glove.6B.100d.txt"
     MAX_SEQUENCE_LENGTH=100
     MAX_VOCAB_SIZE=20000
     EMBEDDING_DIM=100
     VALIDATION_SPLIT=0.2
     BATCH_SIZE=32
     EPOCHS=10
```

LOADING THE PRETRAINED WORD VECTORS

```
[ ]  print('Loading word vectors...')
     word2vec = {}
     with open(EMBEDDING_FILE) as f:
       for line in f:
         values = line.split()
         word = values[0]
         vec = np.asarray(values[1:], dtype='float32')
         word2vec[word] = vec
     print('Found %s word vectors.' % len(word2vec))
```

```
Loading word vectors...
Found 400000 word vectors.
```

CONVERT STRING INTO INTEGERS

```
[ ]  tokenizer = Tokenizer(num_words=MAX_VOCAB_SIZE)
     tokenizer.fit_on_texts(list(data['clean_text']))
     X = tokenizer.texts_to_sequences(list(data['clean_text']))

     # pad sequences so that we get a N x T matrix
     X = pad_sequences(X, maxlen=MAX_SEQUENCE_LENGTH)
     print('Shape of data tensor:', X.shape)
```

```
Shape of data tensor: (44898, 100)
```

CREATE WORD TO INTEGER MAPPING

```
[ ]  word2idx = tokenizer.word_index
     print('Found %s unique tokens.' % len(word2idx))
```

```
Found 218659 unique tokens.
```

EMBEDDING MATRIX

```
[ ]  print('Filling pre-trained embeddings...')
     num_words = min(MAX_VOCAB_SIZE, len(word2idx) + 1)
     embedding_matrix = np.zeros((num_words, EMBEDDING_DIM))
     for word, i in word2idx.items():
       if i < MAX_VOCAB_SIZE:
         embedding_vector = word2vec.get(word)
         if embedding_vector is not None:
           # words not found in embedding index will be all zeros.
           embedding_matrix[i] = embedding_vector
```

```
Filling pre-trained embeddings...
```

EMBEDDING LAYER

```
[ ]  embedding_layer = Embedding(
       num_words,
       EMBEDDING_DIM,
       weights=[embedding_matrix],
       input_length=MAX_SEQUENCE_LENGTH,
       trainable=False
     )
```

CREATE AN LSTM NETWORK WITH A SINGLE LSTM

```
[ ]  print('Building model...')

     # create an LSTM network with a single LSTM
     input_ = Input(shape=(MAX_SEQUENCE_LENGTH,))
     x = embedding_layer(input_)
     # x = LSTM(15, return_sequences=True)(x)
     x = Bidirectional(LSTM(15, return_sequences=True))(x)
     x = GlobalMaxPool1D()(x)
     output = Dense(1, activation="sigmoid")(x)

     model = Model(input_, output)
     model.compile(
       loss='binary_crossentropy',
       optimizer='adam',
       metrics=['accuracy']
     )
     model.summary()
```

```
Building model...
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 100)]             0

 embedding (Embedding)       (None, 100, 100)          2000000

 bidirectional (Bidirection  (None, 100, 30)           13920
 al)

 global_max_pooling1d (Glob  (None, 30)                0
 alMaxPooling1D)

 dense (Dense)               (None, 1)                 31

=================================================================
Total params: 2013951 (7.68 MB)
Trainable params: 13951 (54.50 KB)
Non-trainable params: 2000000 (7.63 MB)
_________________________________________________________________
```

SPLIT THE DATA SET

```
[ ] from sklearn.model_selection import train_test_split
    y=data['Target'].values
    X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.20,stratify=y, random_state=0)
```

TRAIN THE MODEL

```
[ ] print('Training model...')
    r = model.fit(
      X_train,
      y_train,
      batch_size=BATCH_SIZE,
      epochs=EPOCHS,
      validation_split=VALIDATION_SPLIT
    )
```

```
Training model...
Epoch 1/10
898/898 [==============================] - 81s 90ms/step - loss: 0.0072 - accuracy: 0.9984 - val_loss: 0.0701 - val_accuracy: 0.9802
Epoch 2/10
898/898 [==============================] - 75s 84ms/step - loss: 0.0032 - accuracy: 0.9998 - val_loss: 0.0833 - val_accuracy: 0.9800
Epoch 3/10
898/898 [==============================] - 91s 101ms/step - loss: 0.0060 - accuracy: 0.9979 - val_loss: 0.1015 - val_accuracy: 0.9705
Epoch 4/10
898/898 [==============================] - 74s 83ms/step - loss: 0.0041 - accuracy: 0.9994 - val_loss: 0.0837 - val_accuracy: 0.9801
Epoch 5/10
898/898 [==============================] - 76s 85ms/step - loss: 0.0016 - accuracy: 0.9999 - val_loss: 0.0906 - val_accuracy: 0.9781
Epoch 6/10
898/898 [==============================] - 74s 83ms/step - loss: 0.0011 - accuracy: 1.0000 - val_loss: 0.0937 - val_accuracy: 0.9793
Epoch 7/10
898/898 [==============================] - 73s 81ms/step - loss: 9.0125e-04 - accuracy: 1.0000 - val_loss: 0.0997 - val_accuracy: 0.9776
Epoch 8/10
898/898 [==============================] - 76s 85ms/step - loss: 0.0034 - accuracy: 0.9990 - val_loss: 0.0832 - val_accuracy: 0.9790
Epoch 9/10
898/898 [==============================] - 73s 81ms/step - loss: 0.0049 - accuracy: 0.9985 - val_loss: 0.0945 - val_accuracy: 0.9787
Epoch 10/10
898/898 [==============================] - 77s 86ms/step - loss: 0.0037 - accuracy: 0.9990 - val_loss: 0.0873 - val_accuracy: 0.9805
```

ACCURACY

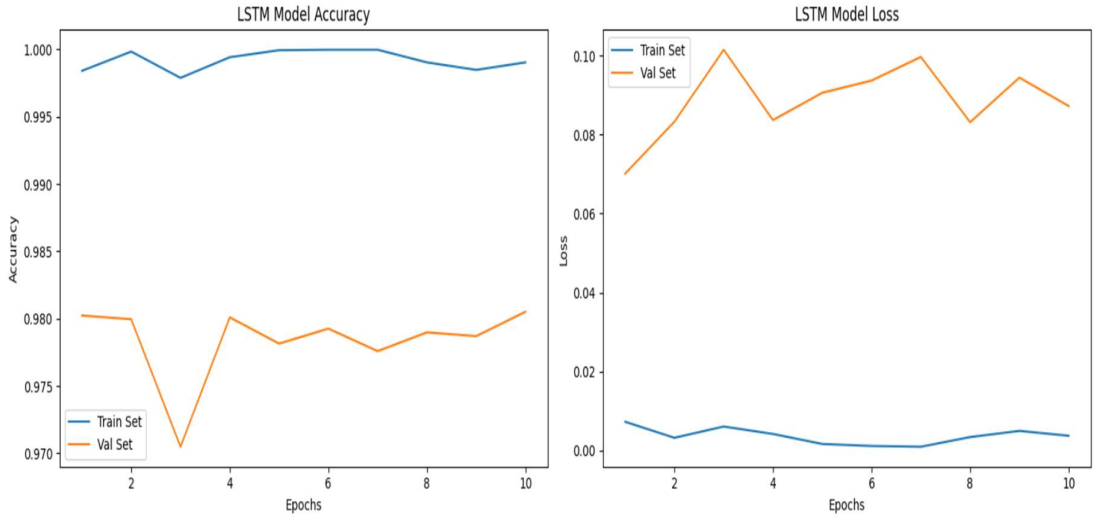
```
[ ]  acc = r.history['accuracy']
     val_acc = r.history['val_accuracy']
     loss = r.history['loss']
     val_loss = r.history['val_loss']
     epochs_range = range(1, len(r.epoch) + 1)

     plt.figure(figsize=(15,5))

     plt.subplot(1, 2, 1)
     plt.plot(epochs_range, acc, label='Train Set')
     plt.plot(epochs_range, val_acc, label='Val Set')
     plt.legend(loc="best")
     plt.xlabel('Epochs')
     plt.ylabel('Accuracy')
     plt.title('LSTM Model Accuracy')

     plt.subplot(1, 2, 2)
     plt.plot(epochs_range, loss, label='Train Set')
     plt.plot(epochs_range, val_loss, label='Val Set')
     plt.legend(loc="best")
     plt.xlabel('Epochs')
     plt.ylabel('Loss')
     plt.title('LSTM Model Loss')

     plt.tight_layout()
     plt.show()
```




```
[ ]  print("Accuracy of the model on Training Data is - " , model.evaluate(X_train,y_train)[1]*100)
     print("Accuracy of the model on Testing Data is - " , model.evaluate(X_test,y_test)[1]*100)
```

```
1123/1123 [==============================] - 24s 22ms/step - loss: 0.0184 - accuracy: 0.9961
Accuracy of the model on Training Data is -  99.6074378490448
281/281 [==============================] - 6s 22ms/step - loss: 0.0740 - accuracy: 0.9835
Accuracy of the model on Testing Data is -  98.35189580917358
```

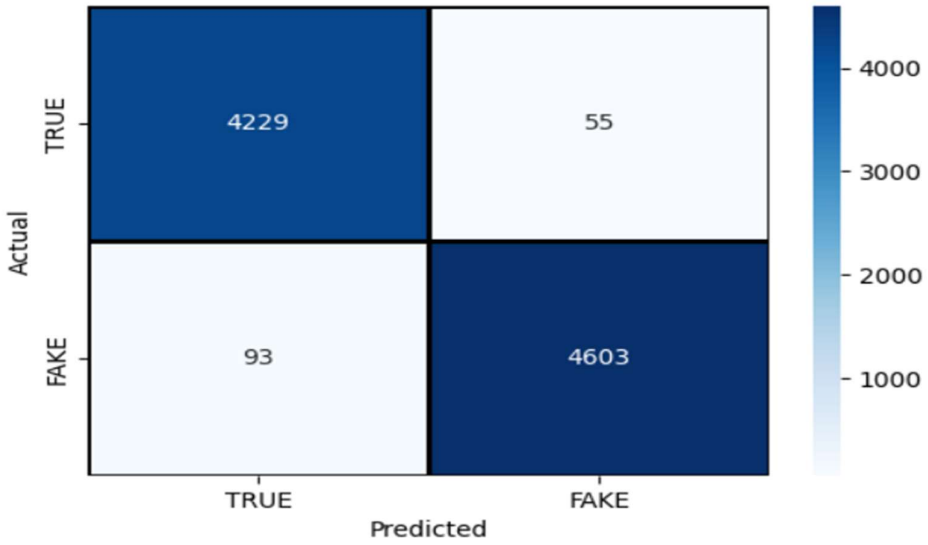
PREDICTION

```
[ ]  pred = model.predict(X_test)
     pred[:5]
```

```
281/281 [==============================] - 9s 31ms/step
array([[1.0000000e+00],
       [2.4139799e-08],
       [9.9999994e-01],
       [1.0000000e+00],
       [1.0000000e+00]], dtype=float32)
```

CONFUSION MATRIX

```
[ ]  cm = confusion_matrix(y_test,pred.round())
     cm = pd.DataFrame(cm , index = ['TRUE','FAKE'] , columns = ['TRUE','FAKE'])
     plt.figure(figsize = (6,4))
     sns.heatmap(cm,cmap= "Blues", linecolor = 'black' , linewidth = 1 , annot = True, fmt='' , xticklabels = ['TRUE','FAKE'] , yticklabels = ['TRUE','FAKE'])
     plt.ylabel('Actual')
     plt.xlabel('Predicted')
     plt.show()
```



CLASSIFICATION REPORT

```
[ ]  print(classification_report(y_test,pred.round()))
```

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.98      4284
           1       0.99      0.98      0.98      4696

    accuracy                           0.98      8980
   macro avg       0.98      0.98      0.98      8980
weighted avg       0.98      0.98      0.98      8980
```

```
[ ]  y_pred = model.predict(X_test).ravel()
```

```
281/281 [==============================] - 9s 33ms/step
```

ROC AUC PLOT

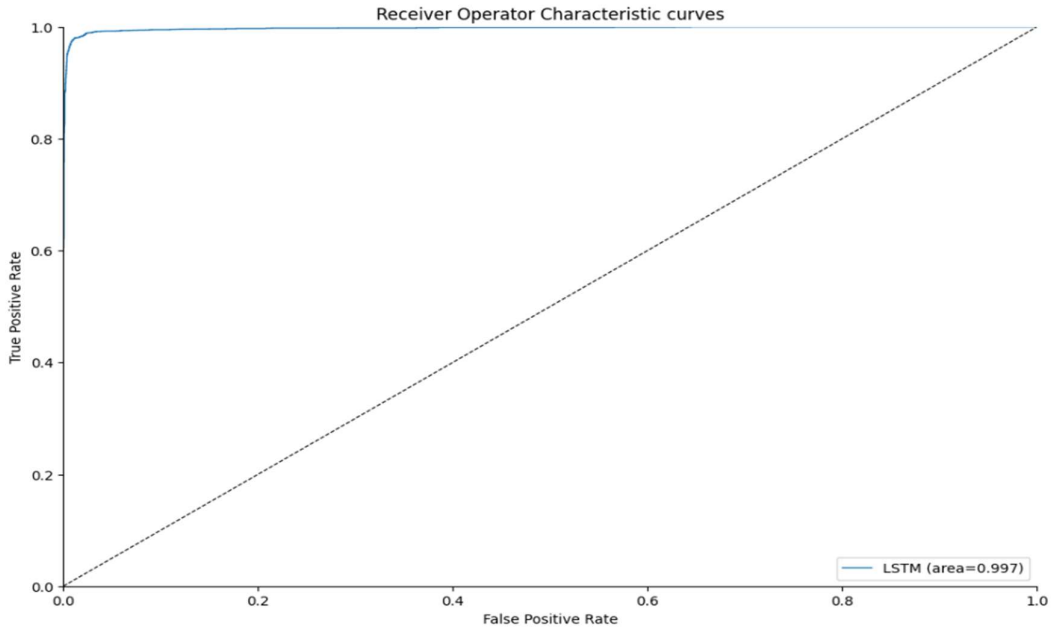
```
[ ]  def roc_auc_plot(y_true, y_proba, label=' ', l='-', lw=1.0):
         from sklearn.metrics import roc_curve, roc_auc_score
         fpr, tpr, _ = roc_curve(y_true, y_proba)
         ax.plot(fpr, tpr, linestyle=l, linewidth=lw,
                 label="%s (area=%.3f)"%(label,roc_auc_score(y_true, y_proba)))

     f, ax = plt.subplots(figsize=(12,8))

     roc_auc_plot(y_test,y_pred,label='LSTM', l='-')



     ax.plot([0,1], [0,1], color='k', linewidth=0.8, linestyle='--',
             )
     ax.legend(loc="lower right")
     ax.set_xlabel('False Positive Rate')
     ax.set_ylabel('True Positive Rate')
     ax.set_xlim([0, 1])
     ax.set_ylim([0, 1])
     ax.set_title('Receiver Operator Characteristic curves')
     sns.despine()
```



MODEL PREDICTION

```
[ ] testSent =["Trey Gowdy destroys this clueless DHS employee when asking about the due process of getting on the terror watch list. Her response is priceless:  I m sorry, um, there s nc
        "Poland s new prime minister faces a difficult balancing act trying to repair bruised relations with the European Union without alienating the eurosceptic government s core vot
    ]
```

```
[ ]  def cleanText(txt):
         txt = txt.lower()
         txt = ' '.join([word for word in txt.split() if word not in (stop)])
         txt = re.sub('[^a-z]',' ',txt)
         return txt
```

PREDICT TEXT AND TOKENIZED

```
[ ]  def predict_text(lst_text):
         test = tokenizer.texts_to_sequences(lst_text)
         # pad sequences so that we get a N x T matrix
         testX = pad_sequences(test, maxlen=MAX_SEQUENCE_LENGTH)
         df_test = pd.DataFrame(lst_text, columns = ['test_sent'])

         prediction = model.predict(testX)
         df_test['prediction']=prediction
         df_test["test_sent"] = df_test["test_sent"].apply(cleanText)
         df_test['prediction']=df_test['prediction'].apply(lambda x: "Fake" if x>=0.5 else "Real")
         return df_test
```

PREDICTION OF THE MODEL

```
[ ]  df_testsent = predict_text(testSent)
     df_testsent
```

```
1/1 [==============================] - 0s 61ms/step
```

	test_sent	prediction
0	trey gowdy destroys clueless dhs employee aski...	Fake
1	poland new prime minister faces difficult bala...	Real

CONCLUSION:

Our goal of developing an effective fake news detection model. We've made substantial progress by carefully preprocessing text data, extracting relevant features, training a classification model, and evaluating its performance. Leveraging NLP techniques, we've ensured that the model works with high-quality, cleansed text data, enabling it to distinguish between fake and real news articles. This project contributes to the fight against

misinformation, reinforcing responsible journalism, and aligning with efforts to combat fake news.