

ASSIGNMENT 4

UMAPRIYA SELVAM

Import required library

```
✓ [32] import pandas as pd
0s      import numpy as np
      from keras.layers import *
      from keras.models import Model
      from keras.optimizers import RMSprop
      from keras.preprocessing import sequence
      from keras.utils import pad_sequences
      from keras.preprocessing.text import Tokenizer
      import warnings
      warnings.filterwarnings("ignore")
      from sklearn.model_selection import train_test_split
      from sklearn.feature_extraction.text import TfidfVectorizer
      from sklearn.linear_model import LogisticRegression
      from sklearn.metrics import accuracy_score\
```

```
✓ [2] from google.colab import drive
17s     drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
✓ [3] df=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/spam.csv", encoding="ISO-8859-1")
1s
```

```
✓ [8] max_len=200
0s     max_words = 1000
```

```
[ ] df.head()
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

Data preprocessing

```
[ ] #Replacing null values
md=df.where((pd.notnull(df)), '')
```

```
[ ] md.head()
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...			
1	ham	Ok lar... Joking wif u oni...			
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...			
3	ham	U dun say so early hor... U c already then say...			
4	ham	Nah I don't think he goes to usf, he lives aro...			

```
[ ] md.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)
md.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
#   Column  Non-Null Count  Dtype
---  -
0   v1       5572 non-null    object
1   v2       5572 non-null    object
dtypes: object(2)
memory usage: 87.2+ KB
```

```
[ ] md.head()
```

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

```
[ ] #checking number of rows and columns
md.shape
```

```
(5572, 2)
```

Label encoding

```
[ ] #Label spam=0,ham=1
md.loc[md['v1']=='spam','v1',]=0
md.loc[md['v1']=='ham','v1',]=1
```

Seperating text and label

```
[ ] md.head
```

```
<bound method NDFrame.head of      v1
0      1  Go until jurong point, crazy.. Available only ...
1      1                               Ok lar... Joking wif u oni...
2      0  Free entry in 2 a wkly comp to win FA Cup fina...
3      1  U dun say so early hor... U c already then say...
4      1  Nah I don't think he goes to usf, he lives aro...
... ..
5567  0  This is the 2nd time we have tried 2 contact u...
5568  1                               Will i_ b going to esplanade fr home?
5569  1  Pity, * was in mood for that. So...any other s...
5570  1  The guy did some bitching but I acted like i'd...
5571  1                               Rofl. Its true to its name

[5572 rows x 2 columns]>
```

```
[ ] x=md['v2']
    y=md['v1']
```



0s

[18] `print(x)`

```
0      Go until jurong point, crazy.. Available only ...
1                Ok lar... Joking wif u oni...
2      Free entry in 2 a wkly comp to win FA Cup fina...
3      U dun say so early hor... U c already then say...
4      Nah I don't think he goes to usf, he lives aro...
      ...
5567    This is the 2nd time we have tried 2 contact u...
5568                Will i_ b going to esplanade fr home?
5569    Pity, * was in mood for that. So...any other s...
5570    The guy did some bitching but I acted like i'd...
5571                Rofl. Its true to its name
Name: v2, Length: 5572, dtype: object
```



0s

[19] `print(y)`

```
0      1
1      1
2      0
3      1
4      1
      ..
5567    0
5568    1
5569    1
5570    1
5571    1
Name: v1, Length: 5572, dtype: object
```

✓ 1s

```
inputs = Input(name='inputs', shape=[max_len])
layer = Embedding(max_words,50,input_length=max_len)(inputs)
layer = LSTM(64)(layer)
layer = Dense(256,name='FC1')(layer)
layer = Activation('relu')(layer)
layer = Dropout(0.5)(layer)
layer = Dense(1,name='out_layer')(layer)
layer = Activation('sigmoid')(layer)
model = Model(inputs=inputs,outputs=layer)
model.summary()
```

Model: "model"

Layer (type)	Output Shape	Param #
inputs (InputLayer)	[(None, 200)]	0
embedding (Embedding)	(None, 200, 50)	50000
lstm (LSTM)	(None, 64)	29440
FC1 (Dense)	(None, 256)	16640
activation (Activation)	(None, 256)	0
dropout (Dropout)	(None, 256)	0
out_layer (Dense)	(None, 1)	257
activation_1 (Activation)	(None, 1)	0
Total params: 96,337		
Trainable params: 96,337		
Non-trainable params: 0		

Splitting data into training and testing set

```
✓ [21] x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=3)
```

0s

```
[ ] print(x.shape)
    print(x_test.shape)
    print(x_train.shape)
```

(5572,)

(1115,)

(4457,)

Feature extraction

```
✓ [22] #Transform text data into feature vectors
1s fe=TfidfVectorizer(min_df=1,stop_words='english',lowercase='true')
```

```
x_train_features=fe.fit_transform(x_train)
```

```
x_test_features=fe.transform(x_test)
```

```
y_train=y_train.astype('int')
```

```
y_test=y_test.astype('int')
```

```
✓ [23] print(x_train_features)
```

0s

(0, 741)	0.3219352588930141
(0, 3979)	0.2410582143632299
(0, 4296)	0.3891385935794867
(0, 6599)	0.20296878731699391
(0, 3386)	0.3219352588930141

Training the model

```
✓ [34] tok = Tokenizer(num_words=max_words)
0s tok.fit_on_texts(x_train)
sequences = tok.texts_to_sequences(x_train)
sequences_matrix = pad_sequences(sequences,maxlen=max_len)

✓ [27] model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
0s

✓ [36] model.fit(sequences_matrix,y_train,batch_size=128,epochs=10,validation_split=0.2)
2m

Epoch 1/10
28/28 [=====] - 14s 382ms/step - loss: 0.3457 - accuracy: 0.8690 - val_loss: 0.1574 - val_accuracy: 0.9585
Epoch 2/10
28/28 [=====] - 10s 363ms/step - loss: 0.1060 - accuracy: 0.9734 - val_loss: 0.0979 - val_accuracy: 0.9731
Epoch 3/10
28/28 [=====] - 10s 359ms/step - loss: 0.0539 - accuracy: 0.9849 - val_loss: 0.0432 - val_accuracy: 0.9888
Epoch 4/10
28/28 [=====] - 10s 354ms/step - loss: 0.0435 - accuracy: 0.9871 - val_loss: 0.0404 - val_accuracy: 0.9922
Epoch 5/10
28/28 [=====] - 10s 360ms/step - loss: 0.0300 - accuracy: 0.9905 - val_loss: 0.0382 - val_accuracy: 0.9910
Epoch 6/10
28/28 [=====] - 10s 357ms/step - loss: 0.0235 - accuracy: 0.9933 - val_loss: 0.0501 - val_accuracy: 0.9865
Epoch 7/10
28/28 [=====] - 10s 377ms/step - loss: 0.0210 - accuracy: 0.9941 - val_loss: 0.0401 - val_accuracy: 0.9922
Epoch 8/10
28/28 [=====] - 10s 355ms/step - loss: 0.0151 - accuracy: 0.9947 - val_loss: 0.0497 - val_accuracy: 0.9888
Epoch 9/10
28/28 [=====] - 12s 431ms/step - loss: 0.0132 - accuracy: 0.9958 - val_loss: 0.0807 - val_accuracy: 0.9854
Epoch 10/10
28/28 [=====] - 13s 476ms/step - loss: 0.0103 - accuracy: 0.9975 - val_loss: 0.0521 - val_accuracy: 0.9888
<keras.callbacks.History at 0x7ff7df574a50>
```

Preprocessing the Test Dataset

```
✓ [41]
0s test_sequences = tok.texts_to_sequences(x_test)
test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)
```

Testing model

```
✓ [42] accr = model.evaluate(test_sequences_matrix,y_test)
1s

35/35 [=====] - 1s 39ms/step - loss: 0.0569 - accuracy: 0.9865
```

