```
import numpy as np
import pandas as pd
import re

import tensorflow as tf
from tensorflow.keras.layers import Embedding, Dense, GlobalAveragePooling1D
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

```
test= pd.read_csv('/content/SMS_test (2).csv',encoding='latin1')
train= pd.read_csv('/content/SMS_train (1).csv',encoding='latin1')
```

```
test
```

	S. No.	Message_body	Label
0	1	UpgrdCentre Orange customer, you may now claim...	Spam
1	2	Loan for any purpose £500 - £75,000. Homeowner...	Spam
2	3	Congrats! Nokia 3650 video camera phone is you...	Spam
3	4	URGENT! Your Mobile number has been awarded wi...	Spam
4	5	Someone has contacted our dating service and e...	Spam
...	...	...	...
120	121	7 wonders in My WORLD 7th You 6th Ur style 5th...	Non-Spam
121	122	Try to do something dear. You read something f...	Non-Spam
122	123	Sun ah... Thk mayb can if dun have anythin on....	Non-Spam
123	124	SYMPTOMS when U are in love: "1.U like listeni...	Non-Spam
124	125	Great. Have a safe trip. Dont panic surrender ...	Non-Spam

125 rows × 3 columns

```
train
```

	S. No.	Message_body	Label
0	1	Rofl. Its true to its name	Non-Spam
1	2	The guy did some bitching but I acted like i'd...	Non-Spam
2	3	Pity, * was in mood for that. So...any other s...	Non-Spam
3	4	Will ü b going to esplanade fr home?	Non-Spam
4	5	This is the 2nd time we have tried 2 contact u...	Spam
...	...	...	...
952	953	hows my favourite person today? r u workin har...	Non-Spam
953	954	How much you got for cleaning	Non-Spam
954	955	Sorry da. I gone mad so many pending works wha...	Non-Spam
955	956	Wat time ü finish?	Non-Spam
956	957	Just glad to be talking to you.	Non-Spam

957 rows × 3 columns

```
train.drop('S. No.',axis=1,inplace=True)
test.drop('S. No.',axis=1,inplace=True)
```

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 957 entries, 0 to 956
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Message_body  957 non-null    object
 1   Label         957 non-null    object
dtypes: object(2)
memory usage: 15.1+ KB
```

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125 entries, 0 to 124
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Message_body  125 non-null    object
 1   Label         125 non-null    object
dtypes: object(2)
memory usage: 2.1+ KB
```

```
train.columns=['text','label']
test.columns=['text','label']
```

```
train['label']=train['label'].replace(['Spam','Non-Spam'],[1,0])
test['label']=test['label'].replace(['Spam','Non-Spam'],[1,0])
```

```
train
```

	text	**label**
0	Rofl. Its true to its name	0
1	The guy did some bitching but I acted like i'd...	0
2	Pity, * was in mood for that. So...any other s...	0
3	Will ü b going to esplanade fr home?	0
4	This is the 2nd time we have tried 2 contact u...	1
...	...	...
952	hows my favourite person today? r u workin har...	0
953	How much you got for cleaning	0
954	Sorry da. I gone mad so many pending works wha...	0
955	Wat time ü finish?	0
956	Just glad to be talking to you.	0

957 rows × 2 columns

```
import string
def clean_text(text):
     # Remove special characters and numbers
    text = re.sub(r'[^A-Za-zÀ-ú ]+', '', text)

    # Convert to lower case
    text = text.lower()

    # Remove extra whitespace
    text = re.sub(r'\s+', ' ', text).strip()
    return text

train['text'] = train['text'].apply(clean_text)
test['text'] = test['text'].apply(clean_text)
```

```
train
```

	text	label
0	rofl its true to its name	0
1	the guy did some bitching but i acted like id ...	0
2	pity was in mood for that soany other suggestions	0
3	will b going to esplanade fr home	0
4	this is the nd time we have tried contact u u ...	1
...	...	...
952	hows my favourite person today r u workin hard...	0
953	how much you got for cleaning	0
954	sorry da i gone mad so many pending works what...	0
955	wat time finish	0
956	just glad to be talking to you	0

957 rows × 2 columns

```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer, SnowballStemmer
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

```
def remove_stopwords(texto):
    stop_words = set(stopwords.words('english'))
    tokens = nltk.word_tokenize(texto.lower())
    return " ".join([token for token in tokens if token not in stop_words])



train['text'] = train['text'].apply(remove_stopwords)
```

```
train['text'] = train['text'].apply(remove_stopwords)
test['text'] = test['text'].apply(remove_stopwords)
```

```
def normalize_text(text):
    stemmer = SnowballStemmer("english")
    normalized_text = []
    for word in text.split():
        stemmed_word = stemmer.stem(word)
        normalized_text.append(stemmed_word)

    return ' '.join(normalized_text)


train['text'] = train['text'].apply(normalize_text)
test['text'] = test['text'].apply(normalize_text)
```

```
train
```

	text	label
0	rofl true name	0
1	guy bitch act like id interest buy someth els ...	0
2	piti mood soani suggest	0
3	b go esplanad fr home	0
4	nd time tri contact u u pound prize claim easi...	1
...	...	...
952	how favourit person today r u workin hard coul...	0
953	much got clean	0
954	sorri da gone mad mani pend work	0
955	wat time finish	0
956	glad talk	0

957 rows × 2 columns

```
# Maximum number of words to be considered in the vocabulary
max_words = 10000
# Maximum number of tokens in a sequence
max_len = 200

tokenizer = Tokenizer(num_words = max_words)
tokenizer.fit_on_texts(train['text'])
# Converts texts into strings of numbers
sequences_train = tokenizer.texts_to_sequences(train['text'])
sequences_val = tokenizer.texts_to_sequences(test['text'])
# Mapping words to indexes
word_index = tokenizer.word_index
```

```
data_train = pad_sequences(sequences_train, maxlen = max_len)
data_val = pad_sequences(sequences_val, maxlen = max_len)
```

```
model = tf.keras.Sequential()
model.add(Embedding(max_words, 35, input_length = max_len))
model.add(GlobalAveragePooling1D())
model.add(Dense(64, activation = 'relu'))
# model.add(Dense(128, activation = 'relu'))
model.add(Dense(1, activation = 'sigmoid'))

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

```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 200, 16)           160000

 global_average_pooling1d_1  (None, 16)                0
  (GlobalAveragePooling1D)

 dense_1 (Dense)             (None, 32)                544

 dense_2 (Dense)             (None, 1)                 33

=================================================================
Total params: 160577 (627.25 KB)
Trainable params: 160577 (627.25 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
history = model.fit(data_train, train['label'], epochs = 15, batch_size = 64, validation_c
```

```
Epoch 1/15
15/15 [==============================] - 4s 184ms/step - loss: 0.6715 - accuracy: 0.78
Epoch 2/15
15/15 [==============================] - 3s 191ms/step - loss: 0.6172 - accuracy: 0.87
Epoch 3/15
15/15 [==============================] - 3s 183ms/step - loss: 0.5514 - accuracy: 0.87
Epoch 4/15
15/15 [==============================] - 1s 93ms/step - loss: 0.4774 - accuracy: 0.872
Epoch 5/15
15/15 [==============================] - 2s 134ms/step - loss: 0.4115 - accuracy: 0.87
Epoch 6/15
15/15 [==============================] - 1s 105ms/step - loss: 0.3783 - accuracy: 0.87
Epoch 7/15
15/15 [==============================] - 1s 45ms/step - loss: 0.3697 - accuracy: 0.872
Epoch 8/15
15/15 [==============================] - 2s 95ms/step - loss: 0.3685 - accuracy: 0.872
Epoch 9/15
```

```
15/15 [==============================] - 1s 94ms/step - loss: 0.3670 - accuracy: 0.872
Epoch 10/15
15/15 [==============================] - 1s 45ms/step - loss: 0.3655 - accuracy: 0.872
Epoch 11/15
15/15 [==============================] - 0s 33ms/step - loss: 0.3642 - accuracy: 0.872
Epoch 12/15
15/15 [==============================] - 1s 82ms/step - loss: 0.3626 - accuracy: 0.872
Epoch 13/15
15/15 [==============================] - 1s 51ms/step - loss: 0.3608 - accuracy: 0.872
Epoch 14/15
15/15 [==============================] - 2s 110ms/step - loss: 0.3593 - accuracy: 0.87
Epoch 15/15
15/15 [==============================] - 1s 43ms/step - loss: 0.3575 - accuracy: 0.872
```

```
model.save('/content/nlp.h5')
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079: UserWarning
  saving_api.save_model(
```

```
from tensorflow.keras.models import load_model
model = load_model('/content/nlp.h5')
```

```
def predict_text(text):
    # Apply the same preprocessing steps as during training
    text = clean_text(text)
    text = remove_stopwords(text)
    text = normalize_text(text)

    sequence = tokenizer.texts_to_sequences([text])
    padded_sequence = pad_sequences(sequence, maxlen=max_len)

    prediction = model.predict(padded_sequence)[0][0]

    return prediction


text_to_predict = "Congratulations! Upon reviewing your application, we would like to invi
prediction = predict_text(text_to_predict)
print(f"Predicted Probability: {prediction}")

classification = 'spam' if prediction >= 0.5 else 'non spam'
print(f"Class: {classification}")
```

```
1/1 [==============================] - 0s 28ms/step
Predicted Probability: 0.11614471673965454
Class: non spam
```