

▼ Import required library

```
import pandas as pd
import numpy as np
import tensorflow
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout, Embedding
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing import sequence
```

▼ Read dataset

```
df = pd.read_csv('/content/spam.csv', delimiter=',', encoding='latin-1')
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

▼ Pre-Processing

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

```
X = df.v2
```

```
Y = df.v1
encoder = LabelEncoder()
Y = encoder.fit_transform(Y)
Y = Y.reshape(-1,1)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
```

```
tokenizer = Tokenizer(num_words=2000, lower=True)
tokenizer.fit_on_texts(X_train)
sequences = tokenizer.texts_to_sequences(X_train)
X_train = sequence.pad_sequences(sequences, maxlen=200)
```

▼ Create Model

```
model = Sequential()
```

▼ Add layers

```
model.add(Embedding(2000, 50, input_length=200))
model.add(LSTM(64))
model.add(Dense(256, activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(1, activation="sigmoid"))
```

```
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
embedding (Embedding)	(None, 200, 50)	100000
lstm (LSTM)	(None, 64)	29440
dense (Dense)	(None, 256)	16640
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 1)	257
=====		
Total params: 146,337		
Trainable params: 146,337		
Non-trainable params: 0		

▼ Compile the Model

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

▼ Fit the Model

```
model.fit(X_train, y_train, batch_size=128, epochs=10, validation_split=0.2)
```

```
Epoch 1/10
28/28 [=====] - 12s 342ms/step - loss: 0.3261 - accuracy: 0.50
Epoch 2/10
28/28 [=====] - 9s 310ms/step - loss: 0.0879 - accuracy: 0.90
Epoch 3/10
28/28 [=====] - 9s 306ms/step - loss: 0.0399 - accuracy: 0.95
Epoch 4/10
28/28 [=====] - 9s 307ms/step - loss: 0.0298 - accuracy: 0.95
Epoch 5/10
28/28 [=====] - 9s 330ms/step - loss: 0.0216 - accuracy: 0.95
Epoch 6/10
28/28 [=====] - 9s 306ms/step - loss: 0.0163 - accuracy: 0.95
Epoch 7/10
28/28 [=====] - 9s 306ms/step - loss: 0.0096 - accuracy: 0.95
Epoch 8/10
28/28 [=====] - 9s 306ms/step - loss: 0.0073 - accuracy: 0.95
Epoch 9/10
28/28 [=====] - 9s 304ms/step - loss: 0.0061 - accuracy: 0.95
Epoch 10/10
28/28 [=====] - 9s 305ms/step - loss: 0.0067 - accuracy: 0.95
<keras.callbacks.History at 0x7fafc6ac1550>
```

▼ Save the Model

```
model.save("model.h5")
```

▼ Test the Model

```
test_sequences = tokenizer.texts_to_sequences(X_test)
X_test = sequence.pad_sequences(test_sequences, maxlen=200)
acc = model.evaluate(X_test, y_test)
```

```
35/35 [=====] - 1s 26ms/step - loss: 0.1049 - accuracy: 0.97
```

```
def predict(message):  
    txt = tokenizer.texts_to_sequences(message)  
    txt = sequence.pad_sequences(txt, maxlen=200)  
    preds = model.predict(txt)  
    if preds > 0.5:  
        print("Spam")  
    else:  
        print("Not Spam")
```

```
predict(["I HAVE A DATE ON SUNDAY WITH WILL!!"])
```

```
1/1 [=====] - 0s 432ms/step  
Not Spam
```

```
predict(["WINNER!! As a valued network customer you have been selected to receivea 🎁900 pi
```

```
1/1 [=====] - 0s 24ms/step  
Spam
```

