

Assignment -4

PROJECT NAME	Digital Naturalist – AI Enabled Tool for Biodiversity Researchers
NAME	Amethyst Sardonyx P
ROLL NO	923319104005
TEAM ID	PNT2022TMID49396

1. Import the necessary libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Model
from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding
from keras.optimizers import RMSprop
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.utils import pad_sequences
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
```

2. Read dataset and do pre-processing

(i) Read dataset

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

Unnamed: 0 Unnamed: 1 Unnamed: 2

v1 v2 3 4

2

3

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 spamFree entry in 2 a wkly comp to win FA Cup

fin... 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



(ii) Preprocessing the dataset

```
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.v1le = LabelEncoder() Y = le.fit_transform(Y)
```

```
Y = Y.reshape(-1,1)
```

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)
```

```
max_words=1000 max_len
```

```
= 150
```

```
tok = Tokenizer(num_words=max_words) tok.fit_on_texts(X_train)
```

```
sequences = tok.texts_to_sequences(X_train)
```

```
sequences_matrix = pad_sequences(sequences,maxlen=max_len)
```

3.4. Create model and Add Layers(LSTM ,Dense-(Hidden Layers), Output)

```
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, 150)] 0

embedding (Embedding) (None, 150, 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 5. Compile

the model

model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy']) 7. Train

and Fit the model

model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
validation_split=0.2)

Epoch 1/10

30/30 [=====] - 8s 263ms/step - loss: 0.0060 - accurac

Epoch 30/30 Epoch 30/30 [=====] - 8s

2/10 263ms/step - loss: 0.0572 - accurac

[=====] - 8s

263ms/step - loss: 0.0036 - accurac 3/10 Epoch

4/10

30/30 Epoch [=====] 5/10 accurac

[=====] - 8s 262ms/step - loss: 0.0038 -

30/30 Epoch [=====] [=====] - 8s - 8s 263ms/step 0.0018 0.0022 accurac

30/30 Epoch 6/10 7/10 261ms/step - loss: - loss: - accurac -

[=====]

30/30 310ms/step - loss: 0.0020 - accurac [=====] - 9s

Epoch 8/10

30/30 Epoch 30/30 Epoch [=====] 9/10 [=====]

===== - 8s - 8s 261ms/step 264ms/step 0.0015 0.0015 - accurac -
=====]

10/10 - loss: - loss: accurac

30/30 263ms/step - loss: 0.0021 - accurac

[=====] - 8s

<keras.callbacks.History at 0x7f2b60b5f110>

6. Save the model

```
model.save('sms_classifier.h5')
```

Preprocessing the Test Dataset

```
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = pad_sequences(test_sequences, maxlen=max_len)
```

7. Testing the model

```
accr = model.evaluate(test_sequences_matrix,Y_test)
```

27/27 [=====] - 1s 21ms/step - loss: 0.2618 - accuracy

```
print('Test set\n Loss: {:.3f}\n Accuracy: {:.3f}'.format(accr[0],accr[1]))
```

Test set Loss: 0.262
Accuracy: 0.977