Assignment-4

Fertilizer recommendation system for Disease prediction

Date	27 October 2022
Name	Kalaiselvi S
Team_ID	PNT2022TMID33521
Maximum marks	2 marks

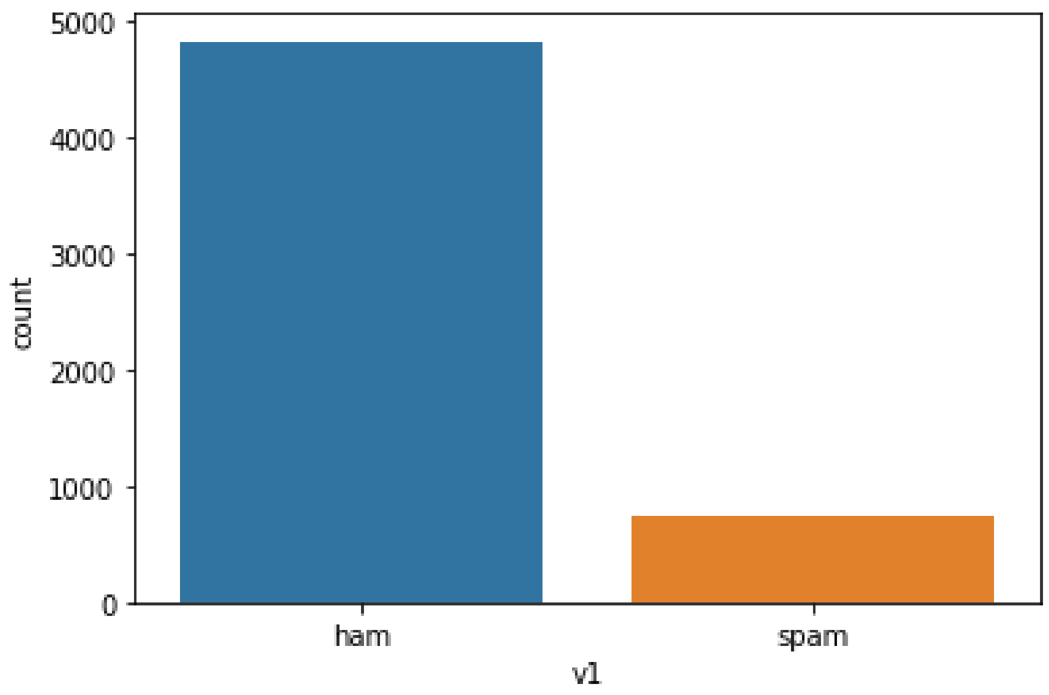
```
import pandas as pd  import numpy as np  import
matplotlib.pyplot as plt  import seaborn as sns  from
sklearn.model_selection   import train_test_split  from
sklearn.preprocessing  import LabelEncoder from
tensorflow.keras.models import Model
from tensorflow.keras.layers    import LSTM,Activation, Dense,
Dropout, Input, Embedding   from tensorflow.keras.optimizers
import RMSprop from tensorflow.keras.preprocessing.text  import
Tokenizer  from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.utils      import to_categorical    from
tensorflow.keras.callbacks   import EarlyStopping  %matplotlib
inline import  csv   with  open('/spam.csv',  'r')  as  csvfile:
reader
= csv.reader(csvfile) df =  pd.read_csv(r'/spam.csv',encoding='latin-1')
df.head()
```

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

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5197dac250>
```



```
X = df.v2 Y = df.v1  le = 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=0.20)
```

```
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 =
sequence.pad_sequences(sequences,maxlen=max_len)
```

```
def RNN():
    inputs = Input(name='inputs',shape=[max_len])     layer =
Embedding(max_words,50,input_length=max_len)(inputs)     layer =
LSTM(128)(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('tanh')(layer)     model =
Model(inputs=inputs,outputs=layer)     return model
```

```
model                 =                 RNN()                 model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accura
cy','mse','mae'])  Model:
```

```
"model"
________________________

________________________


_________________

 Layer (type)                Output Shape              Param #
=================================================================  inputs
(InputLayer)         [(None, 150)]             0
embedding (Embedding)       (None, 150, 50)          50000
lstm (LSTM)                 (None, 128)              91648   FC1 (Dense)
(None, 256)               33024
activation (Activation)     (None, 256)              0
dropout (Dropout)           (None, 256)              0

         out_layer (Dense)           (None, 1)                257
         activation_1 (Activation)   (None, 1)                0

=================================================================
Total params: 174,929
Trainable params: 174,929
Non-trainable params: 0
_________________________________________________________________
```

```
model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,

validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.
0001)])
```

```
Epoch 1/10
```

```
28/28 [==============================] - 17s 486ms/step - loss: 0.2960 -
accuracy: 0.8819 - mse: 0.0821 - mae: 0.1563 - val_loss: 0.1341 -
val_accuracy: 0.9675 - val_mse: 0.0344 - val_mae: 0.1237 Epoch 2/10  28/28
[==============================] - 13s 462ms/step - loss: 0.1149 - accuracy:
0.9764 - mse: 0.0381 - mae: 0.1538 - val_loss: 0.1321 - val_accuracy: 0.9798
- val_mse: 0.0437 - val_mae: 0.1695   <keras.callbacks.History at
0x7f5193192590>  test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix =
sequence.pad_sequences(test_sequences,maxlen=max_len) accr =
model.evaluate(test_sequences_matrix,Y_test)
35/35 [==============================] - 3s 78ms/step - loss: 0.1590 - accuracy:
0.9812 - mse: 0.0451 - mae: 0.1733


print('Test set\n  Loss: {:0.3f}\n  Accuracy:  {:0.3f}'.format(accr[0],accr[1]))

Test set
  Loss:    0.159          Accuracy:    0.981
model.save("./assign4model.h5")          from
tensorflow.keras.models import load_model m2
= load_model("./assign4model.h5") m2.evaluate(test_sequences_matrix,Y_test)
35/35 [==============================] - 3s 68ms/step - loss: 0.1590 - accuracy:
0.9812 - mse: 0.0451 - mae: 0.1733

[0.1589982509613037,
 0.9811659455299377,
 0.04506031796336174,
 0.17333826422691345]
```