Problem Statement :- SMS SPAM Classification

Problem Statement:
Over recent years, as the popularity of mobile phone devices has increased, Short
Message Service (SMS) has grown into a multi-billion dollar industry. At the same time,
reduction in the cost of messaging services has resulted in growth in unsolicited
commercial advertisements (spams) being sent to mobile phones. Due to Spam SMS,
Mobile service providers suffer from some sort of financial problems as well as it reduces
calling time for users. Unfortunately, if the user accesses such Spam SMS they may face
the problem of virus or malware. When SMS arrives at mobile it will disturb mobile user
privacy and concentration. It may lead to frustration for the user. So Spam SMS is one of
the major issues in the wireless communication world and it grows day by day.

Perform the Below Tasks to complete the assignment:-
- Download the Dataset:- https://www.kaggle.com/code/kredy10/simple-lstm-for-text-classification/data
- Import required library
- Read dataset and do pre-processing
- Create Model
- Add Layers (LSTM, Dense-(Hidden Layers), Output)
- Compile the Model
- Fit the Model
- Save The Model
- Test The Model

In [1]:
```
path=("C:/Users/santh/IBM/spam.csv")
```

Import required library

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import keras
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 to_categorical, pad_sequences
from keras.callbacks import EarlyStopping
from sklearn import feature_extraction, model_selection, naive_bayes, metrics, s
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import precision_recall_fscore_support as score
%matplotlib inline
```

Read dataset

In [3]:
```
data = pd.read_csv(path,encoding = "ISO-8859-1")
```

In [4]:
```
data.shape
```

Out[4]:
```
(5572, 5)
```

In [5]:
```
data.head()
```

Out[5]:

	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

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

In [7]:
```
data.describe()
```

Out[7]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
count	5572	5572	50	12	6
unique	2	5169	43	10	5
top	ham	Sorry, I'll call later	bt not his girlfrnd... G o o d n i g h t . . .@"	MK17 92H. 450Ppw 16"	GNT:-)"
freq	4825	30	3	2	2

Data Preprocessing

In [8]:
```
data = data.drop(['Unnamed: 2','Unnamed: 3','Unnamed: 4'],axis=1)
data = data.rename(columns={'v1':'label','v2':'Text'})
data['label_enc'] = data['label'].map({'ham':0,'spam':1})
data.head()
```

Out[8]:

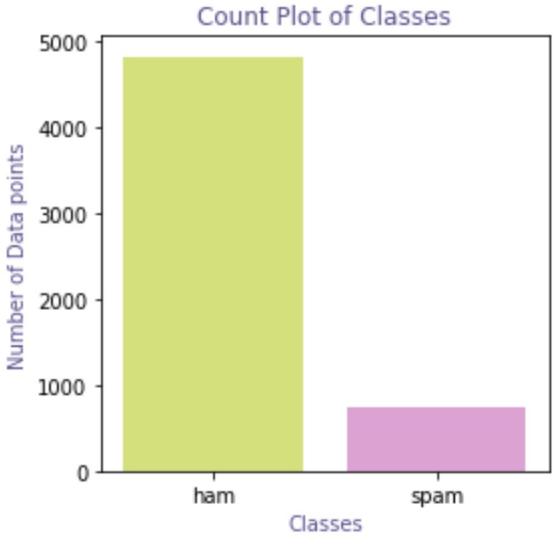
	label	Text	label_enc
0	ham	Go until jurong point, crazy.. Available only ...	0
1	ham	Ok lar... Joking wif u oni...	0
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	ham	U dun say so early hor... U c already then say...	0
4	ham	Nah I don't think he goes to usf, he lives aro...	0

In [9]:
```
data['label'].value_counts()
```

Out[9]:
```
ham     4825
spam     747
Name: label, dtype: int64
```

In [10]:
```
#Palette
cols= ["#E1F16B", "#E598D8"]
#first of all let us evaluate the target and find out if our data is imbalanced
plt.figure(figsize=(4,4))
sns.countplot(x= data['label'], palette= cols)
plt.title("Count Plot of Classes", color="#58508d")
plt.xlabel("Classes", color="#58508d")
plt.ylabel("Number of Data points", color="#58508d")
```

Out[10]:
```
Text(0, 0.5, 'Number of Data points')
```



Data is imbalanced.

86.6 % are "ham" messages and remaining 13.4 % are "spam" messages

Train-test split

In [11]:
```
# Splitting data for Training and testing
from sklearn.model_selection import train_test_split

X, y = np.asanyarray(data['Text']), np.asanyarray(data['label_enc'])
new_data = pd.DataFrame({'Text': X, 'label': y})
X_train, X_test, y_train, y_test = train_test_split(
    new_data['Text'], new_data['label'], test_size=0.2, random_state=42)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

Out[11]:
```
((4457,), (4457,), (1115,), (1115,))
```

In [12]:
```
# Tokenizer
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 = keras.utils.pad_sequences(sequences,maxlen=max_len)
```

Build The Model-LSTM

In [13]:
```
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)
```

In [14]:
```
model = Model(inputs=inputs,outputs=layer)
```

In [15]:
```
model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'
```

```
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
_________________________________________________________________
```

In [16]:
```
model.fit(sequences_matrix,y_train,batch_size=128,epochs=10,validation_split=0.2
```

```
Epoch 1/10
28/28 [==============================] - 9s 239ms/step - loss: 0.3434 - accurac
y: 0.8558 - val_loss: 0.1784 - val_accuracy: 0.9271
Epoch 2/10
28/28 [==============================] - 6s 228ms/step - loss: 0.1018 - accurac
y: 0.9739 - val_loss: 0.0611 - val_accuracy: 0.9821
```

Out[16]:
```
<keras.callbacks.History at 0x1e5a44be8f0>
```

Test the Model

In [17]:
```
#processing test data
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = keras.utils.pad_sequences(test_sequences,maxlen=max_len)
#evaluation of our model
accr = model.evaluate(test_sequences_matrix,y_test)
print('Test set\nLoss: {:0.3f}\nAccuracy: {:0.3f}'.format(accr[0],accr[1]))
```

```
35/35 [==============================] - 1s 38ms/step - loss: 0.0718 - accuracy:
0.9803
Test set
Loss: 0.072
Accuracy: 0.980
```

Save The Model

In [18]:
```
model.save('spam_lstm_model_1.h5')
```