Building an AI Powered spam Classifier

Phase 3: Development part 1

Introduction:

Developing an effective Spam SMS Detection model involves training a machine learning algorithm to automatically identify and filter out unwanted, unsolicited text messages from legitimate ones.

Dataset Used:

https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset

Program:

1. Importing and loading Dataset

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re
import os
import string
import keras
import nltk
import random
import plotly.express as px
import plotly.figure_factory as ff
import spacy
```

```
from plotly import graph_objs as go
from PIL import Image
from wordcloud import WordCloud , STOPWORDS , ImageColorGenerator
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from tqdm import tqdm
from collections import Counter , defaultdict
from keras.preprocessing.text import Tokenizer
from keras.utils import pad_sequences
from keras.callbacks import ModelCheckpoint  , ReduceLROnPlateau
from keras.optimizers import Adam
from keras.models import Sequential
from keras.initializers import Constant
from keras.layers import Dense , LSTM ,Embedding , BatchNormalization ,Dropout ,Bidirectional , Flatten ,GlobalMaxPool1D
from sklearn.metrics import confusion_matrix , classification_report , accuracy_score ,f1_score
```

2. Loading the Data

```
data_path = '/kaggle/input/sms-spam-collection-dataset/spam.csv'

data =  pd.read_csv(data_path , encoding = 'latin')
data = data.drop(['Unnamed: 2' ,'Unnamed: 3' , 'Unnamed: 4'] , axis=1)
data.columns = ['Target', 'Message']
data.reset_index()
data.head()
```

[2]:

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

3. EDA [Exploratory Data Analysis]

```
# let's get length of each message

data['message_length'] = data['Message'].apply(lambda x: len(x.split(" ")))
data.head()
```

[3]:

	Target	Message	message_length
0	ham	Go until jurong point, crazy.. Available only ...	20
1	ham	Ok lar... Joking wif u oni...	6
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	28
3	ham	U dun say so early hor... U c already then say...	11
4	ham	Nah I don't think he goes to usf, he lives aro...	13

3.1 Visualizing Imbalanced Data

```
data['Target'].value_counts()
```

[4]:

```
Target
ham     4825
spam     747
Name: count, dtype: int64
```

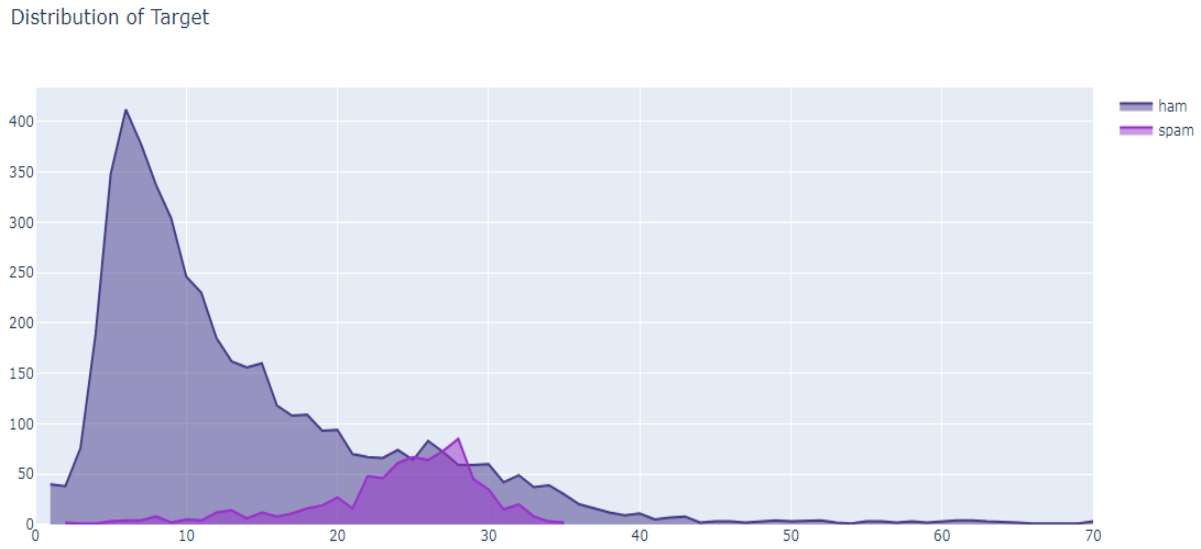
```
Ham_len = data[data['Target']=='ham']['message_length'].value_counts().sort_index()
Spam_len= data[data['Target']=='spam']['message_length'].value_counts().sort_index()


fig = go.Figure()
fig.add_trace(go.Scatter(
    x = Ham_len.index ,
    y = Ham_len.values ,
    name= 'ham' ,
    fill= 'tozeroy',
    marker_color = 'darkslateblue',

    ))
fig.add_trace(go.Scatter(
    x = Spam_len.index ,
    y = Spam_len.values ,
    name = 'spam' ,
    fill = 'tozeroy',
    marker_color = 'darkorchid' ,
    ))
```

```
fig.update_layout( title = 'Distribution of Target')
fig.update_xaxes(range =[0,70])
fig.show()
```

The below diagram shows the amount ham and spam messages in the given dataset



4. Preprocessing the Dataset

```
stop_words = stopwords.words('english') + ['u', 'im', 'c']
stemmer = nltk.SnowballStemmer('english')


def clean_text(text):
    '''Do lowercase, remove text in square brackets,links, punctuation
    and words containing numbers.'''
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    return text
```

```
def preprocessing(text):

    cleaned_text = clean_text(text)
    # remove stopwords
    cleaned_text  = ' '.join(word for word in cleaned_text.split(' ') if word not in stop_words)
    # do stem method
    cleaned_text  = ' '.join(stemmer.stem(word) for word in cleaned_text.split(' '))
    return cleaned_text
```

```
data['Cleaned_Message'] = data['Message'].apply(preprocessing)
# let's show new length after process

data['New_length'] = data['Cleaned_Message'].apply(lambda x: len(x.split(' ')))
data.head()
```

[8]:

	Target	Message	message_length	Cleaned_Message	New_length
0	ham	Go until jurong point, crazy.. Available only ...	20	go jurong point crazi avail bugi n great world...	16
1	ham	Ok lar... Joking wif u oni...	6	ok lar joke wif oni	5
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	28	free entri wkli comp win fa cup final tkts m...	23
3	ham	U dun say so early hor... U c already then say...	11	dun say earli hor alreadi say	6
4	ham	Nah I don't think he goes to usf, he lives aro...	13	nah dont think goe usf live around though	8

5. Tokens Visualisation

```
plt.figure(figsize = (16,5))
plt.title('Top Words For Ham Message')

Word_ham = WordCloud(
    background_color=None, mode="RGBA" ,
    width=800,
    height=300,
    )

Word_ham.generate(' '.join(text for text in  data.loc[ data['Target']=='ham' , 'Cleaned_Message']))
plt.imshow(Word_ham, interpolation='bilinear')
plt.axis('off')
plt.show()


plt.figure(figsize = (16,5))
plt.title('Top Words For Spam Message')
```

```
plt.figure(figsize = (16,5))
plt.title('Top Words For Spam Message')

Word_spam = WordCloud(
    background_color=None, mode="RGBA" ,
    width=800,
    height=300,
    )

Word_spam.generate(' '.join(text for text in  data.loc[ data['Target']=='spam' , 'Cleaned_Message']))
plt.imshow(Word_spam, interpolation='bilinear')
plt.axis('off')
plt.show()
```

Top Words For Ham Message

Top Words For Spam Message

Thus the project has been build by loading and preprocessing the dataset.