In [1]:
```
import numpy as np
import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.model_selection import train_test_split
```

In [3]:
```
df=pd.read_csv(r"spam.csv",encoding='Windows-1252')
df.head()
```

Out[3]:

	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 [4]:
```
df.describe()
```

Out[4]:

	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

In [6]:
```
ps=PorterStemmer()
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Out[6]:
```
True
```

In [7]:
```
data=[]
for i in range(0,5572):
    message=df["v2"][i]
    message=message.lower()
    message=re.sub('[^a-z]',' ',message)
    message=message.split()
    message=[ps.stem(word) for word in message if not word in set(stopwords.words
    message=' '.join(message)
    data.append(message)
```

In [8]:
```
data
```

Out[8]:
```
['go jurong point crazi avail bugi n great world la e buffet cine got amor wa
t',
 'ok lar joke wif u oni',
 'free entri wkli comp win fa cup final tkt st may text fa receiv entri quest
ion std txt rate c appli',
 'u dun say earli hor u c alreadi say',
 'nah think goe usf live around though',
 'freemsg hey darl week word back like fun still tb ok xxx std chg send rcv',
 'even brother like speak treat like aid patent',
 'per request mell mell oru minnaminungint nurungu vettam set callertun calle
r press copi friend callertun',
 'winner valu network custom select receivea prize reward claim call claim co
de kl valid hour',
 'mobil month u r entitl updat latest colour mobil camera free call mobil upd
at co free',
 'gonna home soon want talk stuff anymor tonight k cri enough today',
 'six chanc win cash pound txt csh send cost p day day tsandc appli repli hl
info',
 'urgent week free membership prize jackpot txt word claim c www dbuk net lcc
```

In [9]:
```
cv=CountVectorizer(max_features=7000)
x=cv.fit_transform(data).toarray()
x.shape
```

Out[9]:
```
(5572, 6221)
```

In [10]:
```
df["v1"].loc[df["v1"]=="spam"]=0.0
df["v1"].loc[df["v1"]=="ham"]=1.0
df["v1"]
```

Out[10]:
```
0       1.0
1       1.0
2       0.0
3       1.0
4       1.0
       ...
5567    0.0
5568    1.0
5569    1.0
5570    1.0
5571    1.0
Name: v1, Length: 5572, dtype: object
```

In [12]:
```
y=df.iloc[:,0:1].values
y=np.asarray(y).astype("float64")
y
```

Out[12]:
```
array([[1.],
       [1.],
       [0.],
       ...,
       [1.],
       [1.],
       [1.]])
```

In [13]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [14]:
```
model=Sequential()
```

In [15]:
```
model.add(Dense(units=5572,activation='relu',kernel_initializer='random_uniform'))
#hidden layer
model.add(Dense(units=6000,activation='relu',kernel_initializer='random_uniform'))
model.add(Dense(units=6000,activation='relu',kernel_initializer='random_uniform'))
model.add(Dense(units=6000,activation='relu',kernel_initializer='random_uniform'))
model.add(Dense(units=6000,activation='relu',kernel_initializer='random_uniform'))
model.add(Dense(units=6000,activation='relu',kernel_initializer='random_uniform'))
#output layer
model.add(Dense(units=1,activation='sigmoid',kernel_initializer='random_uniform'))
```

In [16]:
```
model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])
```

In [17]:
```
tr=model.fit(x_train,y_train,epochs=10,batch_size=32)
```

```
Epoch 1/10
140/140 [==============================] - 252s 2s/step - loss: 1.7631 - accura
cy: 0.9524
Epoch 2/10
140/140 [==============================] - 241s 2s/step - loss: 0.0611 - accura
cy: 0.9939
Epoch 3/10
140/140 [==============================] - 247s 2s/step - loss: 0.6032 - accura
cy: 0.9794
Epoch 4/10
140/140 [==============================] - 247s 2s/step - loss: 0.4485 - accura
cy: 0.9910
Epoch 5/10
140/140 [==============================] - 247s 2s/step - loss: 0.1708 - accura
cy: 0.9897
Epoch 6/10
140/140 [==============================] - 246s 2s/step - loss: 2.3482 - accura
cy: 0.9865
Epoch 7/10
140/140 [==============================] - 246s 2s/step - loss: 0.1099 - accura
cy: 0.9971
Epoch 8/10
140/140 [==============================] - 246s 2s/step - loss: 0.0037 - accura
cy: 0.9996
Epoch 9/10
140/140 [==============================] - 246s 2s/step - loss: 0.0033 - accura
cy: 0.9987
Epoch 10/10
140/140 [==============================] - 247s 2s/step - loss: 0.1228 - accura
cy: 0.9939
```

In [18]:
```
model.save("sms.h5")
```

In [19]:
```
ypred=model.predict(x_test)
ypred
```

```
35/35 [==============================] - 15s 399ms/step
```

Out[19]:
```
array([[1.],
       [1.],
       [1.],
       ...,
       [1.],
       [1.],
       [1.]], dtype=float32)
```

In [20]:
```
y_test
```

Out[20]:
```
array([[1.],
       [1.],
       [1.],
       ...,
       [1.],
       [1.],
       [1.]])
```

In [21]:
```
text=model.predict(cv.transform(["Wishing you a very happy Birthday to you ! "]))
text>0.5
```

```
1/1 [==============================] - 0s 305ms/step
```

Out[21]:
```
array([[ True]])
```

In [22]:
```
class_name=["ham","spam"]
pred_id=text.argmax(axis=1)[0]
pred_id
print(str(class_name[pred_id]))
```

```
ham
```

In [ ]:
```

```