New Section

Mayuri Mukunda Jamdar

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import pandas as pd
import numpy as np
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
```

```
# Required Libraries
```

```
# Data Gathering
```

```
df = pd.read_csv("/content/SMSSpamCollection.txt", sep = '\t', names = ['Label','Msg'])
df.head()
```

	Label	Msg
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...

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Label   5572 non-null   object
 1   Msg     5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

```
df.isna().sum()
```

	0
Label	0
Msg	0

dtype: int64

```
df['Label'].value_counts()
```

Label	count
ham	4825
spam	747

dtype: int64

```
df['Label'].value_counts()
```

Label	count
ham	4825
spam	747

```
corpus = []
lm = WordNetLemmatizer()
for i in range (len(df)):
    review = re.sub('^a-zA-Z0-9',' ',df['Msg'][i])
    review = review.lower()
    review = review.split()
    review = [data for data in review if data not in stopwords.words('english')]
    review = [lm.lemmatize(data) for data in review]
    review = " ".join(review)
    corpus.append(review)
```

```
df['Msg'][0]
```

```
len(df['Msg'])
```

```
5572
```

```
len(corpus)
```

```
5572
```

```
df['Msg']=corpus
df.head()
```

	Label	Msg
0	ham	go jurong point, crazy.. available bugis n gre...
1	ham	ok lar... joking wif u oni...
2	spam	free entry 2 wkly comp win fa cup final tkts 2...
3	ham	u dun say early hor... u c already say...
4	ham	nah think go usf, life around though

```
#4.Model Building
#4.1 Data Splitting
```

```
x = df['Msg']
y = df['Label']
```

```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.3, random_state = 10)
```

```
len(x_train), len(y_train)
```

```
(3900, 3900)
```

```
len(x_test),len(y_test)
```

```
(1672, 1672)
```

```
#4.2 Vectorization (Convert Text Data Into The Vectors)
```

```
tf_obj = TfidfVectorizer()
x_train_tfidf = tf_obj.fit_transform(x_train).toarray()
x_train_tfidf
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
```

```
           ...,
           [0., 0., 0., ..., 0., 0., 0.],
           [0., 0., 0., ..., 0., 0., 0.],
           [0., 0., 0., ..., 0., 0., 0.]])
```

```
x_train_tfidf.shape
```

```
(3900, 6931)
```

```
#4.3 Pipeline
```

```
text_mnb = Pipeline([('tfidf',TfidfVectorizer()),('mnb',MultinomialNB())])
```

```
text_mnb.fit(x_train,y_train)
```


```
#Accuracy Score on Testing Data
y_pred_test = text_mnb.predict(x_test)
print("Accuracy Score:", accuracy_score(y_test,y_pred_test)*100)
```

```
Accuracy Score: 95.8732057416268
```

```
#Accuracy Score on Training Data
y_pred_train = text_mnb.predict(x_train)
print("Accuracy Score:",accuracy_score(y_train,y_pred_train)*100)
```

```
Accuracy Score: 98.3076923076923
```

```
#Confusion Matrix on Testing Data
y_pred_test = text_mnb.predict(x_test)
print("Confusion Matrix on Test Data:\n", confusion_matrix(y_test,y_pred_test))
```

```
Confusion Matrix on Test Data:
 [[1457    0]
 [  69  146]]
```

```
#Classification Report on Testing Data
y_pred_test = text_mnb.predict(x_test)
print("Classification Reportx on Test Data:\n", classification_report(y_test,y_pred_test))
```

```
Classification Reportx on Test Data:
                  precision    recall  f1-score   support

              ham       0.95      1.00      0.98      1457
             spam       1.00      0.68      0.81       215

         accuracy                           0.96      1672
        macro avg       0.98      0.84      0.89      1672
     weighted avg       0.96      0.96      0.96      1672
```

```
#Prediction on User_data
```

```
def preprocess_data(text):
    review = re.sub('^a-zA-Z0-9',' ',text)
    review = review.lower()
    review = review.split()
    review = [data for data in review if data not in stopwords.words('english')]
    review = [lm.lemmatize(data) for data in review]
    review = " ".join(review)
    return [review]
```

```
user_data = df['Msg'][0]
print(user_data)
user_data = preprocess_data(user_data)
user_data
```

```
go jurong point, crazy.. available bugis n great world la e buffet... cine got amore wat...
['go jurong point, crazy.. available bugis n great world la e buffet... cine got amore wat...']
```

```
text_mnb.predict(user_data)[0]


class prediction:

    def __init__(self,data):
        self.data = data

    def user_data_preprocessing(self):
        lm = WordNetLemmatizer()
        review = re.sub('^a-zA-Z0-9',' ',self.data)
        review = review.lower()
        review = review.split()
        review = [data for data in review if data not in stopwords.words('english')]
        review = [lm.lemmatize(data) for data in review]
        review = " ".join(review)
        return [review]

    def user_data_prediction(self):
        preprocess_data = self.user_data_preprocessing()

        if text_mnb.predict(preprocess_data)[0] == 'spam':
            return 'This Message is Spam'

        else:
            return 'This Message is Ham'


df.head()
```

	Label	Msg
0	ham	go jurong point, crazy.. available bugis n gre...
1	ham	ok lar... joking wif u oni...
2	spam	free entry 2 wkly comp win fa cup final tkts 2...
3	ham	u dun say early hor... u c already say...
4	ham	nah think go usf, life around though

```
user_data = df['Msg'][2]
print(user_data)
prediction(user_data).user_data_prediction()
```

```
free entry 2 wkly comp win fa cup final tkts 21st may 2005. text fa 87121 receive entry question(std txt rate)t&c's apply 0845281007
'This Message is Spam'
```

```
user_data = df['Msg'][3]
print(user_data)
prediction(user_data).user_data_prediction()
```

```
u dun say early hor... u c already say...
'This Message is Ham'
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.