SPAM MAIL DETECTION USING LOGISTIC REG.

This Spam Mail Prediction project focuses on developing a machine learning model to classify emails as either "spam" or "ham" (not spam) using logistic regression.

Importing the Dependencies

In [ ]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

pd.set_option('display.max_colwidth', None)
```

A. Data Collection and Pre-processing

In [4]:
```
mail_data=pd.read_csv('mail_data.csv')
```

In [6]:
```
print(mail_data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Category  5572 non-null   object
 1   Message   5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
None
```

In [8]:
```
mail_data.head()
```

Out[8]:

	Category	Message
0	ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's
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 around here though

In [10]:
```
#checking no,of rows and columns
```

```
mail_data.shape
```

Out[10]: (5572, 2)

B. LABEL ENCODING

In [13]:
```
#label data  spam as 0, non-spam as 1

mail_data['Category'] = mail_data['Category'].map({"spam": 0, "ham": 1})
```

In [23]:
```
mail_data['Category'].value_counts()
```

Out[23]:
```
Category
1    4825
0     747
Name: count, dtype: int64
```

In [25]:
```
mail_data.head()
```

Out[25]:

	Category	Message
0	1	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
1	1	Ok lar... Joking wif u oni...
2	0	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's
3	1	U dun say so early hor... U c already then say...
4	1	Nah I don't think he goes to usf, he lives around here though

In [29]:
```
#separating data as texts and label

x= mail_data['Message']
y=mail_data['Category']
```

In [ ]:
```
# print(x)
```

In [ ]:
```
# print(y)
```

Spliting data into training and test dataset

In [36]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2,random_st
```

In [ ]:
```
print(x.shape)
print(x_train.shape)
print(x_test.shape)
```

Feature Selection

In [159…
```
# transform text data to feature  vectors so that it can be used as input to log

feature_ext=TfidfVectorizer(min_df= 1,stop_words='english',lowercase=True)
```

In [56]:
```
X_train_features=feature_ext.fit_transform(x_train)
X_test_features= feature_ext.transform(x_test)

Y_train = y_train.astype('int')
Y_test = y_test.astype('int')
```

In [ ]:
```
print(x_train)
print(X_train_features)
```

C. Training the Model

Logistic Regression

In [67]:
```
model= LogisticRegression()
```

In [69]:
```
#training model
model.fit(X_train_features,Y_train)
```

Out[69]:
```
LogisticRegression()
```

D. Evaluating the trained Model

In [72]:
```
#prediction on training data
pred_on_train_data = model.predict(X_train_features)

accuracy_on_training_data= accuracy_score(y_train,pred_on_train_data)
```

In [74]:
```
print('Accuracy on training  data: ',accuracy_on_training_data)
```

```
Accuracy on training  data:  0.96881310298407
```

In [76]:
```
#prediction on test data
pred_on_test_data = model.predict(X_test_features)

accuracy_on_test_data= accuracy_score(y_test,pred_on_test_data)
```

In [78]:
```
print('Accuracy on test data: ',accuracy_on_test_data)
```

```
Accuracy on test data:  0.9560538116591928
```

In [139…
```
# Evaluating the model

confusion = confusion_matrix(y_test, pred_on_test_data)

precision = precision_score(y_test, pred_on_test_data)
recall = recall_score(y_test,pred_on_test_data)
f1 = f1_score(y_test, pred_on_test_data)

print(f'Confusion Matrix: {confusion} \n')
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
```

```
Confusion Matrix: [[112  46]
 [  3 954]]

Precision: 0.954
Recall: 0.9968652037617555
F1 Score: 0.974961676034747
```

Building a predictive system

In [117…

```python
def predict_spam_or_ham(email_text):
    # Ensure input is not empty
    if not email_text.strip():
        print("Error: Input email is empty.")
        return None

    # Transform input text using the trained feature extractor
    try:
        input_data_features = feature_ext.transform([email_text])

        # Make a prediction
        prediction = model.predict(input_data_features)

        # Return 1 for non-spam and 0 for spam
        return int(prediction[0])

    except ValueError as e:
        print("Error during prediction:", e)
        return None


email_text = input("Enter a mail: ")
result = predict_spam_or_ham(email_text)

if result is not None:
    if result == 1:
        print("This is a non-spam (ham) mail.")
    else:
        print("This is a spam mail.")
```

```
This is a non-spam (ham) mail.
```

In [ ]:

In [ ]: