EMAIL SPAM DETECTION USING MACHINE LEARNING

In [1]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix
import nltk
from nltk.corpus import stopwords
from collections import Counter
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
#Load Dataset
df=pd.read_csv("C:/Users/hp/Downloads/archive (5)/spam.csv",encoding='latin-1')
```

In [3]:
```
#To get information about the dataset
df.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 [4]:
```
#Drop unwanted columns
columns_drop=["Unnamed: 2","Unnamed: 3","Unnamed: 4"]
df.drop(columns=columns_drop,inplace=True)
```

In [5]:
```
#Rename columns for better understanding
new_columns={"v1":"Categories","v2":"Messages"}
df.rename(columns=new_columns,inplace=True)
```

In [6]:
```
#For handling missing values
dataset=df.where((pd.notnull(df)),  );dataset
```

Out[6]:

	Categories	Messages
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...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will Ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

In [7]:
```
dataset.head(10)
```
Out[7]:

	Categories	Messages
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...
5	spam	FreeMsg Hey there darling it's been 3 week's n...
6	ham	Even my brother is not like to speak with me. ...
7	ham	As per your request 'Melle Melle (Oru Minnamin...
8	spam	WINNER!! As a valued network customer you have...
9	spam	Had your mobile 11 months or more? U R entitle...

In [8]:
```
dataset.describe()
```
Out[8]:

	Categories	Messages
count	5572	5572
unique	2	5169
top	ham	Sorry, I'll call later
freq	4825	30

In [9]:
```
dataset.info()
```

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

In [10]:
```
dataset.shape
```

Out[10]:
```
(5572, 2)
```

In [11]:
```
#Encode spam as 0 and ham as 1 in the 'Categories' column

dataset.loc[dataset["Categories"]=="spam","Categories"]=0
dataset.loc[dataset["Categories"]=="ham","Categories"]=1
```

In [12]:
```python
#Split the dataset into feature(x) and target variable(y)

x=dataset["Messages"]
y=dataset["Categories"]
print(x)
print(y)
```

```
0       Go until jurong point, crazy.. Available only ...
1                           Ok lar... Joking wif u oni...
2       Free entry in 2 a wkly comp to win FA Cup fina...
3       U dun say so early hor... U c already then say...
4       Nah I don't think he goes to usf, he lives aro...
                              ...                        
5567    This is the 2nd time we have tried 2 contact u...
5568                Will Ì_ b going to esplanade fr home?
5569    Pity, * was in mood for that. So...any other s...
5570    The guy did some bitching but I acted like i'd...
5571                           Rofl. Its true to its name
Name: Messages, Length: 5572, dtype: object
0       1
1       1
2       0
3       1
4       1
       ..
5567    0
5568    1
5569    1
5570    1
5571    1
Name: Categories, Length: 5572, dtype: object
```

In [13]:
```python
#Split the dataset into training and testing dataset

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

In [14]:
```
#TF-IDF for feature extraction
feature_extraction=TfidfVectorizer(min_df=1,stop_words="english",lowercase=True)
x_train_features=feature_extraction.fit_transform(x_train)
x_test_features=feature_extraction.transform(x_test)
```

In [15]:
```
#Convert target variables to integers

y_train=y_train.astype("int")
y_test=y_test.astype("int")
```

In [16]:
```
#Choose the model

model=LogisticRegression()

# Fit it to the training data

model.fit(x_train_features,y_train)
```

Out[16]:
```
LogisticRegression()
```

In [17]:
```
#Predict on the training dataset and calculate accuracy

predict=model.predict(x_train_features)
accuracy=accuracy_score(y_train,predict)
print("Accuracy on training dataset is" ,accuracy)
```

```
Accuracy on training dataset is 0.9694862014808167
```

In [18]:
```
#Define a sample email and predict its category(spam/ham)

email=["Sorry, I'll call later in meeting"]
input_data=feature_extraction.transform(email)
prediction=model.predict(input_data)
print(prediction)
```

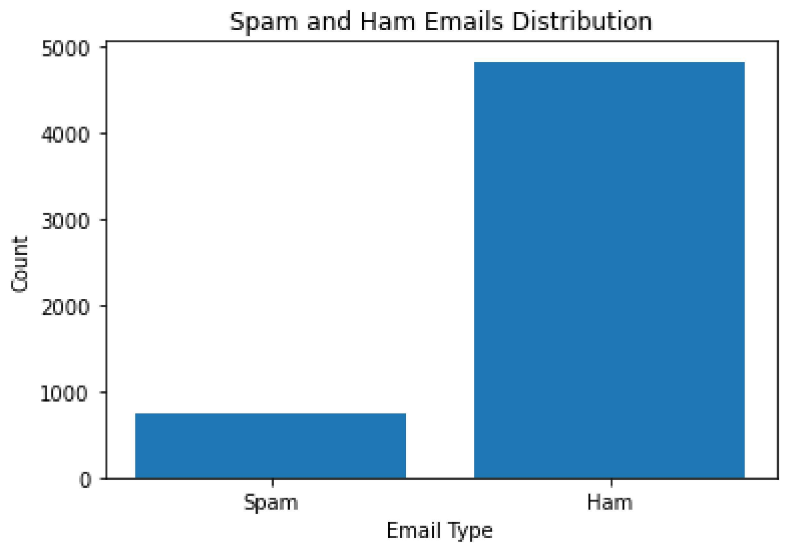
```
[1]
```

```
In [19]: #Interpret prediction result

         if prediction[0]==1:
             print("Ham mail")
         else:
             print("Spam mail")

         Ham mail
```

```
In [20]: #Define another sample email and predict its category(spam/ham)

         email=["Customer service annoncement. You have a New Years delivery waiting for you. Please call 07046744435 r
         input_data=feature_extraction.transform(email)
         prediction=model.predict(input_data)
         print(prediction)

         [0]
```

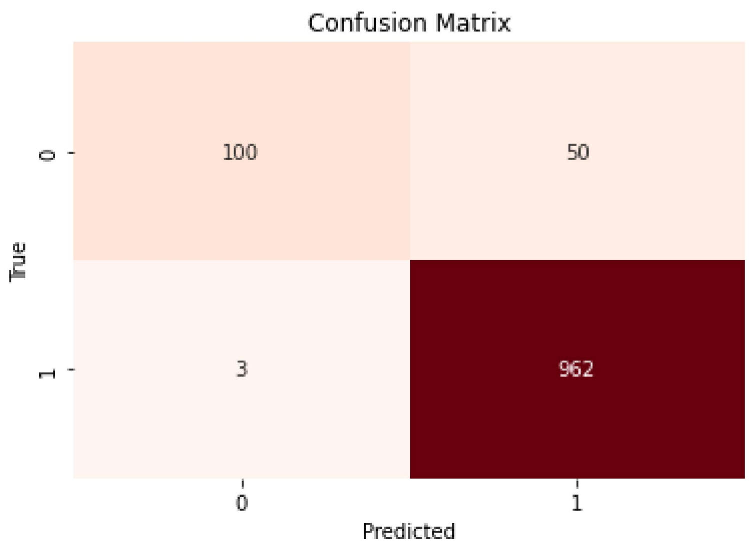
```
In [21]: #Interpret prediction result

         if prediction[0]==1:
             print("Ham mail")
         else:
             print("Spam mail")

         Spam mail
```

In [22]:
```
#Visualize and count the distribution of spam and ham mails

spam_count=dataset[dataset['Categories']==0].shape[0]
ham_count=dataset[dataset['Categories']==1].shape[0]
plt.bar(['Spam','Ham'],[spam_count,ham_count])
plt.xlabel('Email Type')
plt.ylabel('Count')
plt.title('Spam and Ham Emails Distribution')
plt.show()
```



In [23]:
```
prediction_test=model.predict(x_test_features)
#Create a confusion matrix

cm=confusion_matrix(y_test,prediction_test)
```

In [24]:
```python
# Visualize the confusion matrx
plt.figure(figsize=(6,4))
sns.heatmap(cm,annot=True,fmt="d",cmap="Reds",cbar=False)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```

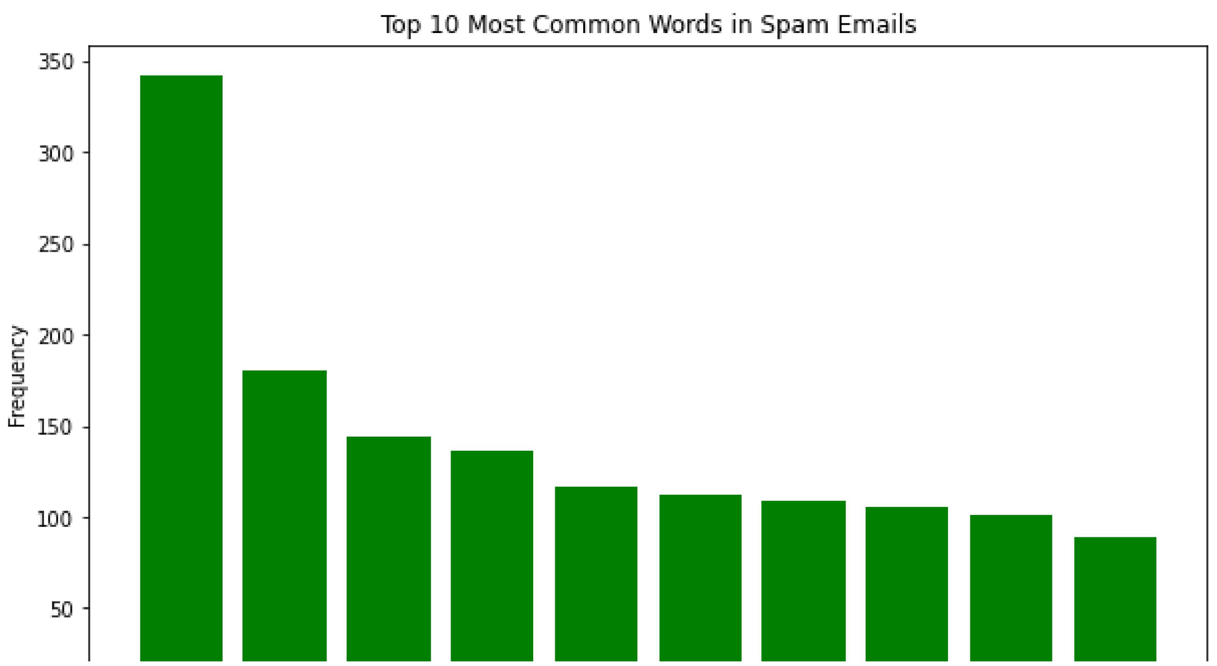


In [25]:
```python
nltk.download('stopwords')
```

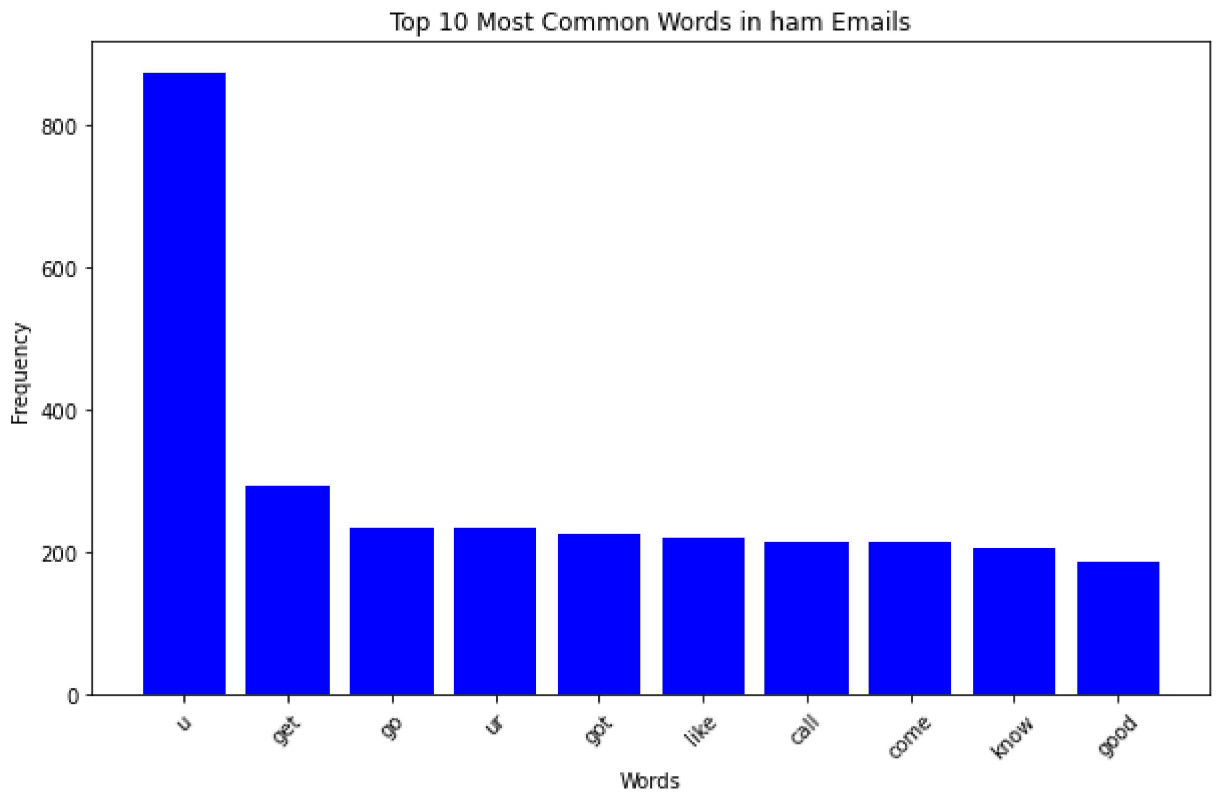
```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[25]:
```
True
```

In [26]:
```
#Count and visualize the most common words in spam emails

stop_words=set(stopwords.words('english'))
spam_words=" ".join(dataset[dataset['Categories']==0]['Messages']).split()
ham_words=" ".join(dataset[dataset['Categories']==1]['Messages']).split()
spam_word_freq=Counter([word.lower() for word in spam_words if word.lower() not in stop_words and word.isalph


plt.figure(figsize=(10,6))
plt.bar(*zip(*spam_word_freq.most_common(10)), color='g')
plt.xlabel('Words')
plt.ylabel('Frequency')
plt.title('Top 10 Most Common Words in Spam Emails')
plt.xticks(rotation=45)
plt.show()
```



```
In [27]: #Count and visualize the most common words in ham emails

         ham_word_freq=Counter([word.lower() for word in ham_words if word.lower() not in stop_words and word.isalpha(


         plt.figure(figsize=(10,6))
         plt.bar(*zip(*ham_word_freq.most_common(10)), color='b')
         plt.xlabel('Words')
         plt.ylabel('Frequency')
         plt.title('Top 10 Most Common Words in ham Emails')
         plt.xticks(rotation=45)
         plt.show()
```


Top 10 Most Common Words in ham Emails
Frequency
u
get
go
ur
got
like
call
come
know
good
Words