Importing Modules

In [13]:
```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from sklearn.metrics import accuracy_score, classification_report
```

In [2]:
```
import os
dataset_location = r'C:\Users\hardi\OneDrive\Documents\spam_mail'
os.chdir(dataset_location)
```

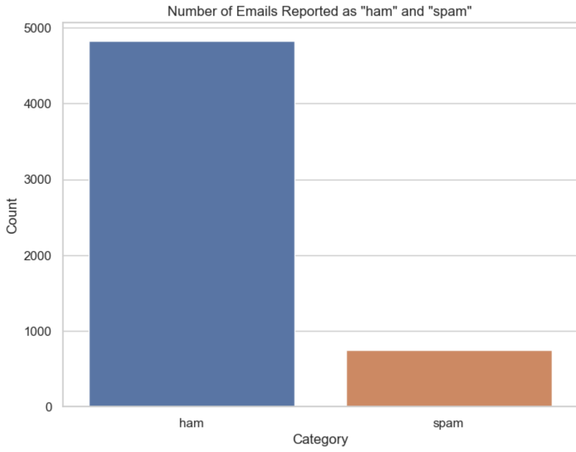
Analysing the Dataset

In [3]:
```
file_name = 'mail_data.csv'
data = pd.read_csv(file_name)
print(data)
```

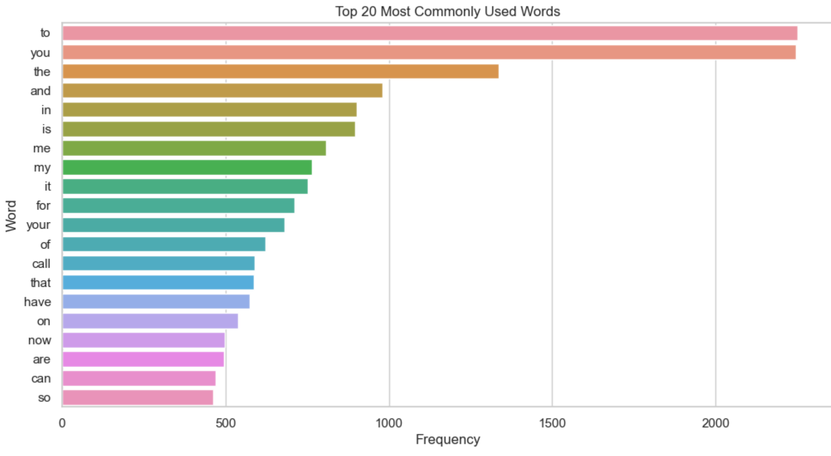
```
     Category                                            Message
0         ham  Go until jurong point, crazy.. Available only ...
1         ham                      Ok lar... Joking wif u oni...
2        spam  Free entry in 2 a wkly comp to win FA Cup fina...
3         ham  U dun say so early hor... U c already then say...
4         ham  Nah I don't think he goes to usf, he lives aro...
...       ...                                                ...
5567     spam  This is the 2nd time we have tried 2 contact u...
5568      ham              Will ü 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 x 2 columns]
```

In [4]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")
plt.figure(figsize=(8, 6))
sns.countplot(x='Category', data=data)
plt.title('Number of Emails Reported as "ham" and "spam"')
plt.xlabel('Category')
plt.ylabel('Count')
plt.show()
```



In [5]:
```
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(data['Message'])
word_frequencies = X.sum(axis=0)
word_frequency_df = pd.DataFrame({'Word': vectorizer.get_feature_names_out(), 'Frequency': word_frequencies.A.ravel()})
word_frequency_df = word_frequency_df.sort_values(by='Frequency', ascending=False)
N = 20
plt.figure(figsize=(12, 6))
sns.barplot(x='Frequency', y='Word', data=word_frequency_df.head(N))
plt.title(f'Top {N} Most Commonly Used Words')
plt.xlabel('Frequency')
plt.ylabel('Word')
plt.show()
```



In [6]:
```
stop_words = set(stopwords.words('english'))
data['Processed_Message'] = data['Message'].apply(lambda x: ' '.join([word for word in word_tokenize(x) if word.lower() not in stop_words]))
```

In [7]:
```
X = data['Processed_Message']
y = data['Category']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [8]:
```
vectorizer = CountVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)
```

In [9]:
```
clf = MultinomialNB()
clf.fit(X_train_vec, y_train)
```

Out[9]:
```
MultinomialNB()
```

Making predictions on the Dataset

In [10]:
```
y_pred = clf.predict(X_test_vec)
```

In [11]:
```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.9829596412556054
```

In [12]:
```
new_message = ["Congratulations! You've won a free vacation!"]
new_message_vec = vectorizer.transform(new_message)
prediction = clf.predict(new_message_vec)
print("Predicted Category for New Message:", prediction[0])
```

```
Predicted Category for New Message: spam
```

Classification Report

In [14]:
```
class_report = classification_report(y_test, y_pred, target_names=['ham', 'spam'])
print(class_report)
```

```
              precision    recall  f1-score   support

         ham       0.99      0.99      0.99       966
        spam       0.95      0.93      0.94       149

    accuracy                           0.98      1115
   macro avg       0.97      0.96      0.96      1115
weighted avg       0.98      0.98      0.98      1115
```