In [1]:
```
import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer
import seaborn as sns
import matplotlib.pyplot as plt
from tqdm.auto import tqdm
import time
```

In [2]:
```
df =pd.read_csv('C:\\Users\\saswa\\OneDrive\\Desktop\\Pinaki_Spam_Email_Detection\\Spam Email Detection-spam.csv')
df
```

Out[2]:

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

5572 rows × 5 columns

In [3]:
```
df.drop(['Unnamed: 2','Unnamed: 3', 'Unnamed: 4'], axis = 1, inplace = True)
```

In [4]:
```
df.head()
```

Out[4]:

	v1	v2
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...

In [5]:
```
df.tail()
```

Out[5]:

	v1	v2
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will 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

In [6]:
```
df.isna().any()
```

Out[6]:
```
v1    False
v2    False
dtype: bool
```

In [7]:
```
df.isna().sum()
```

Out[7]:
```
v1    0
v2    0
dtype: int64
```

In [8]:
```
df['v2'].nunique()
```

Out[8]:
```
5163
```

In [9]:
```
df.shape
```

Out[9]:
```
(5572, 2)
```

In [10]:
```
df['v2'].drop_duplicates(inplace = True)
```

In [11]:
```
df.shape
```

Out[11]:
```
(5572, 2)
```

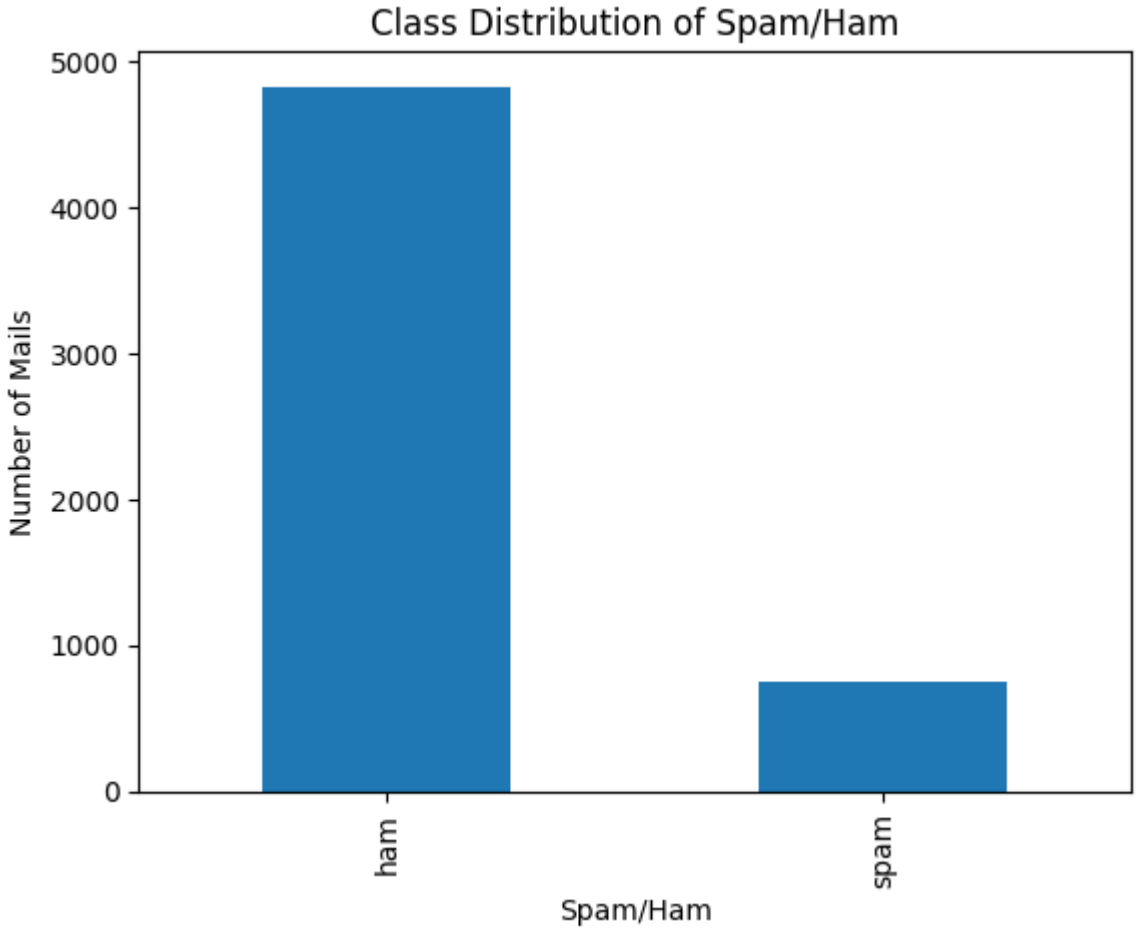
In [12]:
```
df
```

Out[12]:

	v1	v2
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 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 [13]:
```
class_counts = df['v1'].value_counts()
class_counts.plot(kind='bar')
plt.title('Class Distribution of Spam/Ham')
plt.xlabel('Spam/Ham')
plt.ylabel('Number of Mails')
plt.show()
```



In [14]:
```
from collections import Counter
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
```

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

In [15]:
```
all_text = ' '.join(df['v2'].values)
```

```
all_text = re.sub(r'http\S+', '', all_text)
all_text = re.sub(r'@\S+', '', all_text)
all_text = re.sub(r'#\S+', '', all_text)
```

In [16]:
```
words = all_text.split()
```

In [17]:
```
stop_words = set(stopwords.words('english'))
words = [word for word in words if not word in stop_words]
```

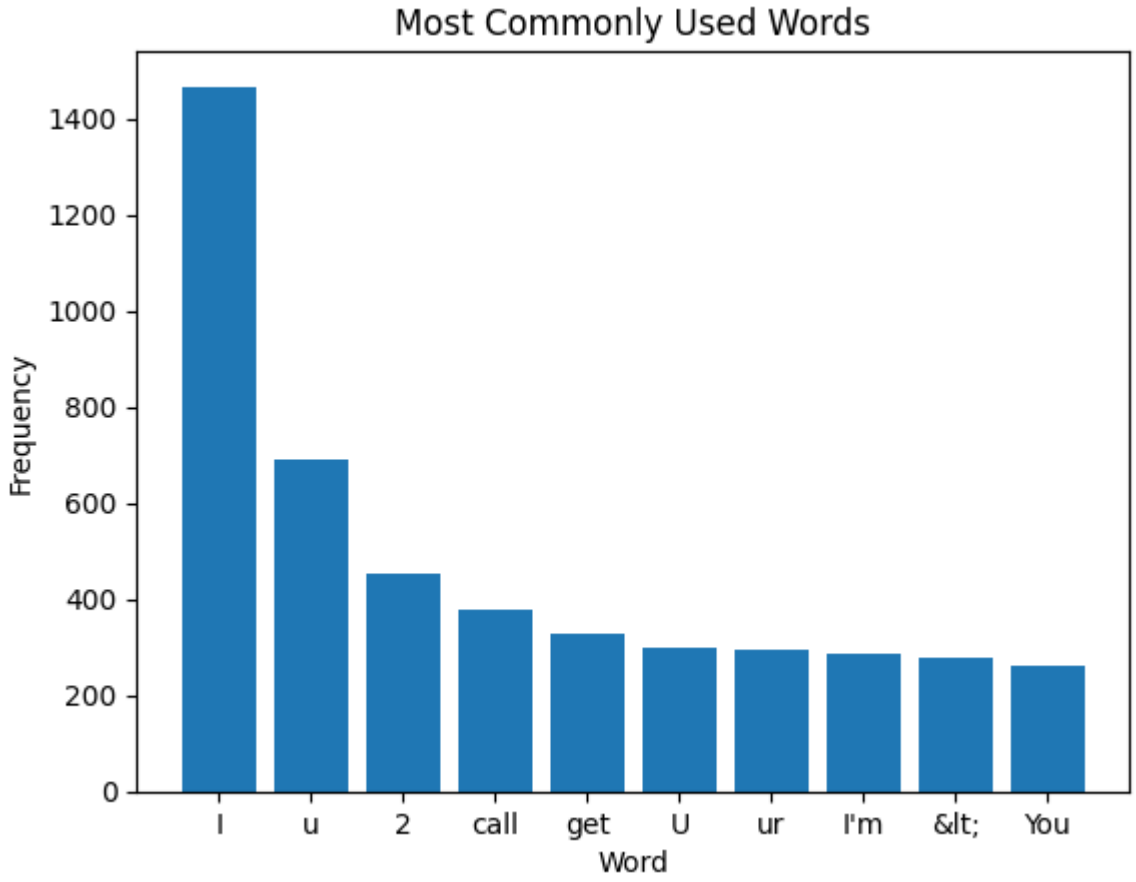
In [18]:
```
word_counts = Counter(words)
top_words = word_counts.most_common(100)
top_words
```

Out[18]:
```
[('I', 1466),
 ('u', 692),
 ('2', 453),
 ('call', 376),
 ('get', 326),
 ('U', 299),
 ('ur', 293),
 ("I'm", 286),
 ('&lt;', 276),
 ('You', 263),
 ('4', 249),
 ('.', 235),
 ('go', 234),
 ('know', 224),
 ('like', 222),
 ('got', 204),
 ('come', 198),
 ('?', 187),
 ('...', 163),
 ('want', 157),
 ('Call', 155),
 ('time', 154),
 ('send', 150),
 ('going', 142),
 ('need', 141),
 ('n', 137),
 ("I'll", 137),
 ('How', 137),
 ('still', 134),
 ('If', 133),
 ('one', 132),
 ('But', 131),
 ('No', 126),
 ('text', 126),
 ('Just', 119),
 ('We', 119),
 ('So', 118),
 ('love', 114),
 ('good', 114),
 ('think', 113),
```

```
('Do', 113),
('see', 113),
('r', 113),
('back', 111),
('home', 107),
('&', 107),
('_', 105),
('tell', 104),
('Your', 104),
('take', 101),
('What', 101),
('day', 101),
('free', 99),
('My', 99),
('And', 98),
('Ok', 97),
('me.', 97),
('dont', 97),
('The', 95),
('mobile', 94),
('A', 92),
("i'm", 91),
('FREE', 90),
('make', 90),
('new', 89),
('-', 88),
('phone', 88),
('later', 87),
('give', 87),
('now.', 86),
('much', 83),
('Have', 83),
('&amp;', 82),
('you.', 82),
('ask', 82),
('To', 79),
('Are', 78),
('This', 77),
('Hey', 76),
('great', 75),
('txt', 75),
```

```
('way', 75),
('reply', 75),
('Can', 74),
('claim', 73),
('say', 72),
('da', 72),
('Good', 72),
('e', 71),
('meet', 71),
('Its', 70),
('really', 69),
('number', 69),
('week', 68),
('Txt', 67),
('lor.', 67),
('contact', 67),
('would', 66),
('said', 65),
('1', 64)]
```

In [19]:
```
top_words = word_counts.most_common(10)
x_values = [word[0] for word in top_words]
y_values = [word[1] for word in top_words]
plt.bar(x_values, y_values)
plt.xlabel('Word')
plt.ylabel('Frequency')
plt.title('Most Commonly Used Words')
plt.show()
```



In [20]:
```
def clean_text(text):
    text = re.sub('<.*?>', '', text)
    text = re.sub('[^a-zA-Z]', ' ', text).lower()
    words = nltk.word_tokenize(text)
    words = [w for w in words if w not in stopwords.words('english')]
    stemmer = PorterStemmer()
    words = [stemmer.stem(w) for w in words]
    text = ' '.join(words)
    return text
```

In [21]:
```
import nltk
nltk.download('punkt')
```

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

Out[21]: True

In [23]:
```
%%time
tqdm.pandas()
df['cleaned_text'] = df['v2'].progress_apply(clean_text)
```

```
  0%|          | 0/5572 [00:00<?, ?it/s]
CPU times: total: 1min 11s
Wall time: 1min 21s
```

In [24]:
```
cv = CountVectorizer(max_features=5000)
X = cv.fit_transform(df['cleaned_text']).toarray()
y = df['v1']
```

In [25]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [26]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
clf = LogisticRegression()
```

In [27]:
```
clf.fit(X_train, y_train)
```

Out[27]:
```
▾ LogisticRegression

LogisticRegression()
```

In [28]:
```
y_pred = clf.predict(X_test)
```

In [29]:
```
y_pred
```

Out[29]:
```
array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'ham'], dtype=object)
```

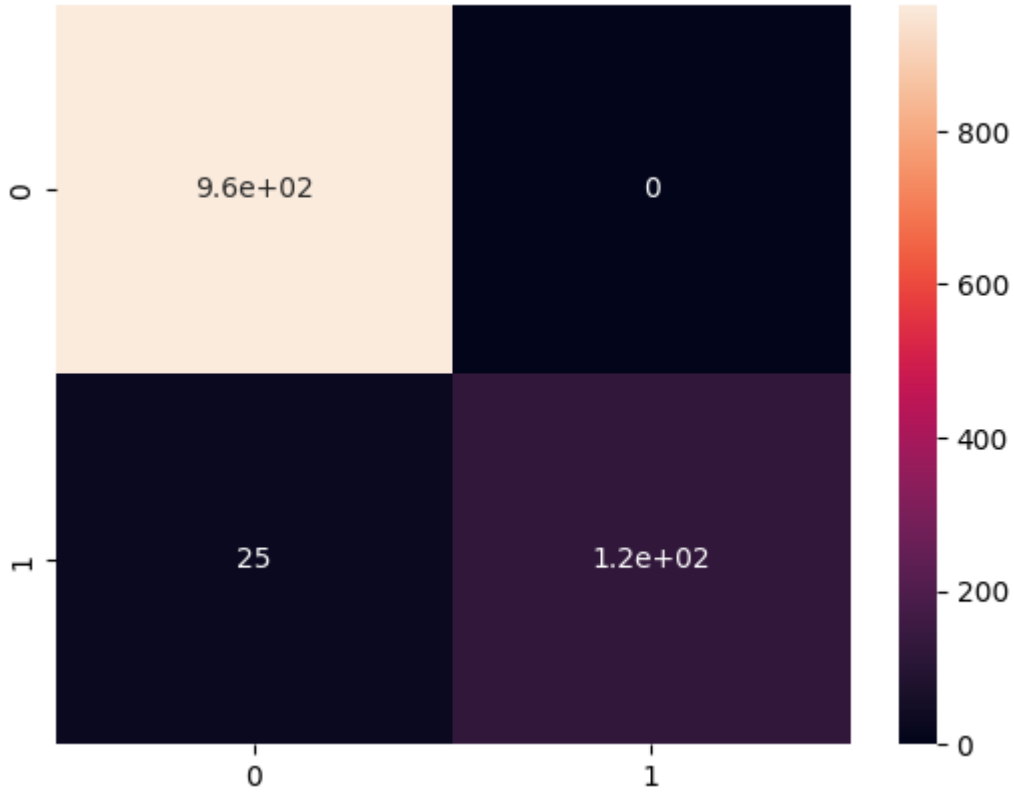
In [30]:
```
acc = accuracy_score(y_test, y_pred)
print("Accuracy:", acc)
```

```
Accuracy: 0.9775784753363229
```

In [31]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

import seaborn as sns
sns.heatmap(cm, annot=True)
```

Out[31]: <Axes: >



In [32]:
```
cm
```

Out[32]:
```
array([[965,   0],
       [ 25, 125]], dtype=int64)
```

In [33]:
```
from sklearn.metrics import classification_report
```

```
report = classification_report(y_test, y_pred)
print(report)
```

```
              precision    recall  f1-score   support

         ham       0.97      1.00      0.99       965
        spam       1.00      0.83      0.91       150

    accuracy                           0.98      1115
   macro avg       0.99      0.92      0.95      1115
weighted avg       0.98      0.98      0.98      1115
```

In [ ]: