```
import pandas as pd
url = 'https://raw.githubusercontent.com/nikhilkr29/Email-Spam-Classifier-using-Naive-Bayes/refs/heads/main/emails.csv'
df = pd.read_csv(url)
df.head()
```

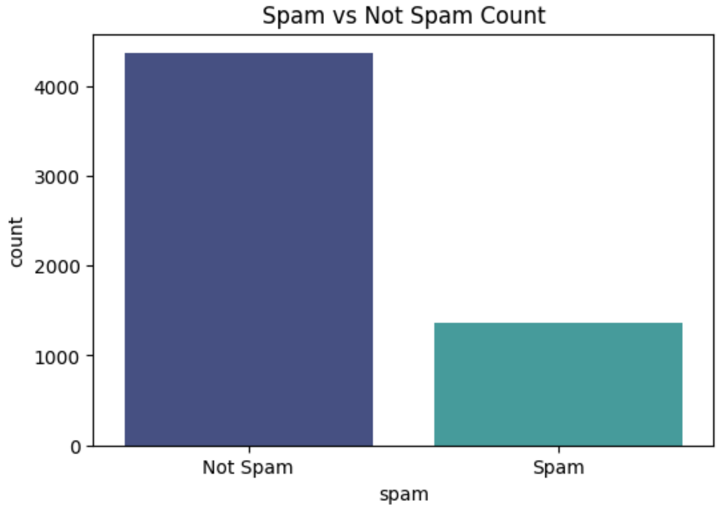
	text	spam
0	Subject: naturally irresistible your corporate...	1
1	Subject: the stock trading gunslinger fanny i...	1
2	Subject: unbelievable new homes made easy im ...	1
3	Subject: 4 color printing special request add...	1
4	Subject: do not have money , get software cds ...	1

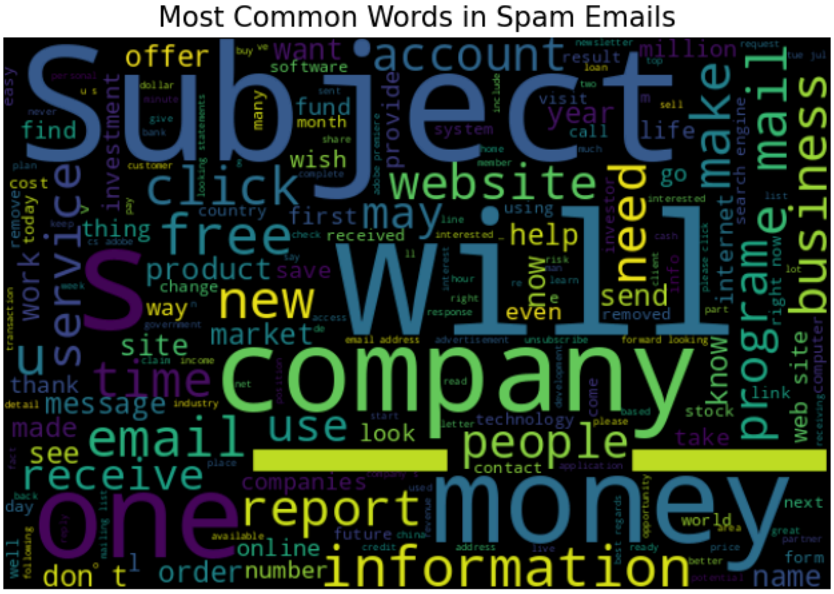
Next steps: Generate code with df View recommended plots New interactive sheet

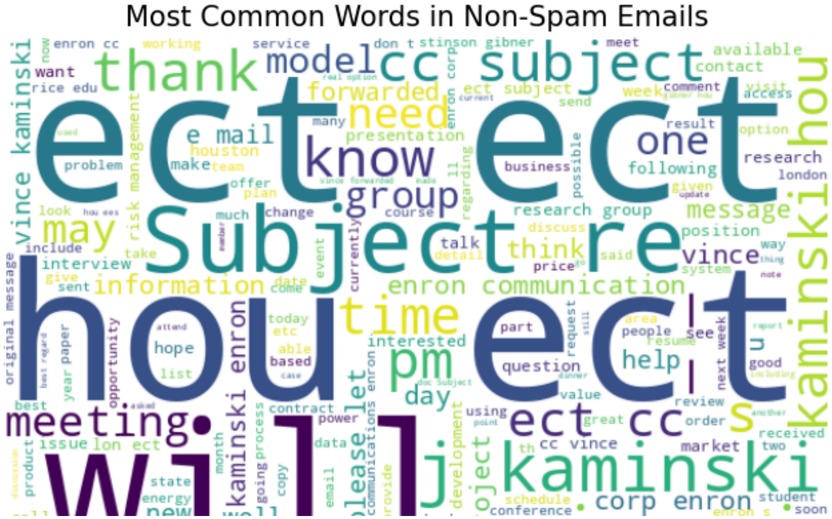
```
# Step 1: Import necessary Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

from wordcloud import WordCloud

# Step 2: Loading the  Dataset
url = 'https://raw.githubusercontent.com/nikhilkr29/Email-Spam-Classifier-using-Naive-Bayes/main/emails.csv'
df = pd.read_csv(url)
df.head()

# Step 3: Data Overview
print(df.head())
print("\nClass Distribution:")
print(df['spam'].value_counts())

# Step 4: Visualization of Spam vs Not Spam
plt.figure(figsize=(6,4))
sns.countplot(x='spam', data=df, palette='mako')
plt.title('Spam vs Not Spam Count')
plt.xticks([0, 1], ['Not Spam', 'Spam'])
plt.show()

# Step 5: WordCloud for Spam
spam_words = ' '.join(df[df['spam'] == 1]['text'])
spam_wc = WordCloud(width=600, height=400, background_color='black').generate(spam_words)

plt.figure(figsize=(8,6))
plt.imshow(spam_wc, interpolation='bilinear')
plt.axis('off')
plt.title("Most Common Words in Spam Emails", fontsize=15)
plt.show()

# Step 6: WordCloud for Non-Spam
ham_words = ' '.join(df[df['spam'] == 0]['text'])
ham_wc = WordCloud(width=600, height=400, background_color='white').generate(ham_words)

plt.figure(figsize=(8,6))
plt.imshow(ham_wc, interpolation='bilinear')
plt.axis('off')
plt.title("Most Common Words in Non-Spam Emails", fontsize=15)
plt.show()

# Step 7: Preprocessing
X = df['text']
y = df['spam']

vectorizer = CountVectorizer()
X_transformed = vectorizer.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_transformed, y, test_size=0.2, random_state=42)

# Step 8: Train Model
model = MultinomialNB()
model.fit(X_train, y_train)
```

```
# Step 9: Predictions and Evaluation
y_pred = model.predict(X_test)

print("\nAccuracy Score:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Step 10: Confusion Matrix
cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(5,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Not Spam', 'Spam'], yticklabels=['Not Spam', 'Spam'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()

# Step 11: Try Custom Input
sample = ["Congratulations! You have won a free iPhone. Click here to claim it."]
sample_transformed = vectorizer.transform(sample)
print("\nPrediction (1 = Spam, 0 = Not Spam):", model.predict(sample_transformed)[0])
```

```
                                                text  spam
0  Subject: naturally irresistible your corporate...     1
1  Subject: the stock trading gunslinger  fanny i...     1
2  Subject: unbelievable new homes made easy  im ...     1
3  Subject: 4 color printing special  request add...     1
4  Subject: do not have money , get software cds ...     1

Class Distribution:
spam
0    4360
1    1368
Name: count, dtype: int64
<ipython-input-4-2e5fc2adef00>:25: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.countplot(x='spam', data=df, palette='mako')
```







Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.