```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
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,
roc_curve, roc_auc_score, classification_report

df = pd.read_csv(r"D:\Bharat Intern\spam.csv",encoding='latin1')

df
```

```
        v1                                                  v2 Unnamed:
2  \
0      ham  Go until jurong point, crazy.. Available only ...
NaN
1      ham                      Ok lar... Joking wif u oni...
NaN
2     spam  Free entry in 2 a wkly comp to win FA Cup fina...
NaN
3      ham  U dun say so early hor... U c already then say...
NaN
4      ham  Nah I don't think he goes to usf, he lives aro...
NaN
...    ...                                                ...         .
..
5567  spam  This is the 2nd time we have tried 2 contact u...
NaN
5568   ham              Will Ì_ b going to esplanade fr home?
NaN
5569   ham  Pity, * was in mood for that. So...any other s...
NaN
5570   ham  The guy did some bitching but I acted like i'd...
NaN
5571   ham                         Rofl. Its true to its name
NaN

     Unnamed: 3 Unnamed: 4
0           NaN        NaN
1           NaN        NaN
2           NaN        NaN
3           NaN        NaN
4           NaN        NaN
...         ...        ...
5567        NaN        NaN
5568        NaN        NaN
5569        NaN        NaN
5570        NaN        NaN
```

```
5571        NaN        NaN

[5572 rows x 5 columns]
```

```
df.isnull().sum()
```

```
v1               0
v2               0
Unnamed: 2    5522
Unnamed: 3    5560
Unnamed: 4    5566
dtype: int64
```

```
df = df.drop(["Unnamed: 2","Unnamed: 3","Unnamed: 4"],axis = 1)
```

```
df.rename(columns ={"v1":"Category","v2":"Message"},inplace = True)
```

```
df
```

```
     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 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 x 2 columns]
```

```
df.info()
```

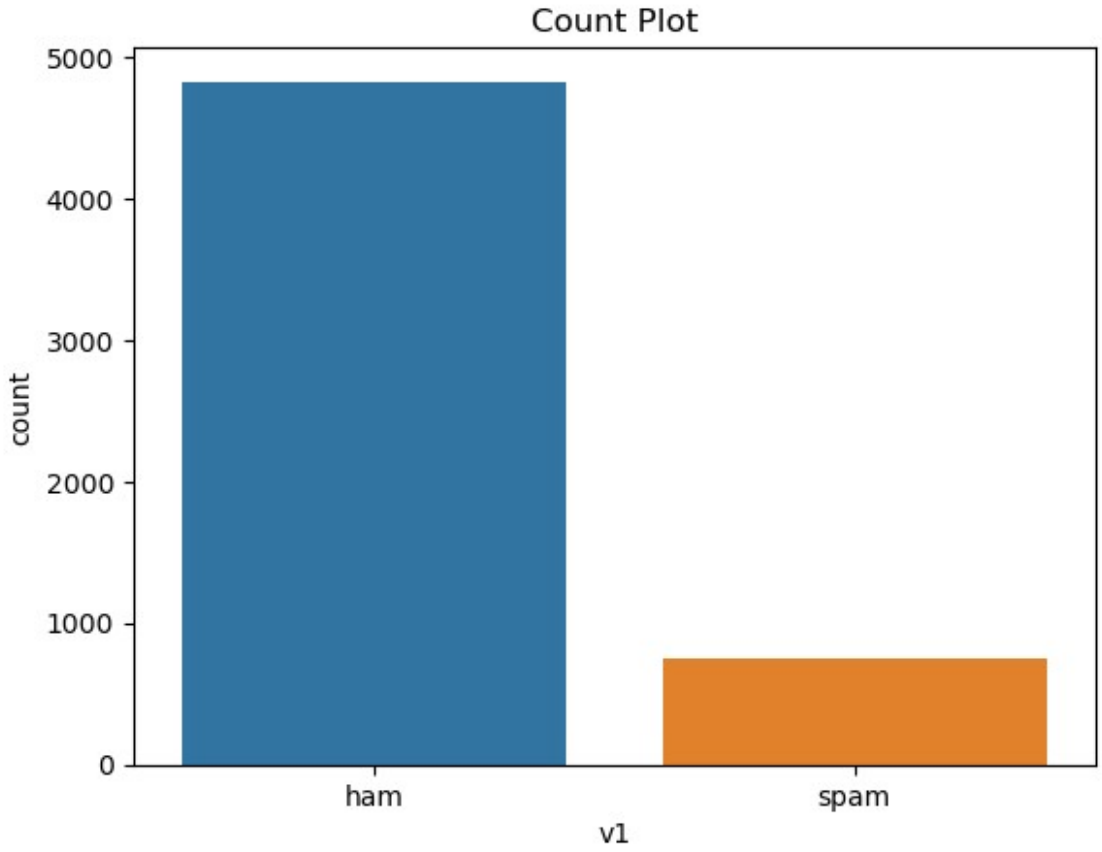
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Category  5572 non-null   object
 1   Message   5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB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

```
df.describe()
```

```
       Category                 Message
count      5572                    5572
unique        2                    5169
```

```
top         ham  Sorry, I'll call later
freq       4825                      30
```

```
sns.countplot(data=df,x='Category')
plt.xlabel('v1')
plt.ylabel('count')
plt.title('Count Plot')
plt.show()
```




```
df.loc[df["Category"] == "spam", "Category"] = 0
df.loc[df["Category"] == "ham", "Category"] = 1
```

```
df
```

```
     Category                                            Message
0           1  Go until jurong point, crazy.. Available only ...
1           1                      Ok lar... Joking wif u oni...
2           0  Free entry in 2 a wkly comp to win FA Cup fina...
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 aro...
...       ...                                                ...
5567        0  This is the 2nd time we have tried 2 contact u...
5568        1              Will Ì_ b going to esplanade fr home?
```

```
5569        1  Pity, * was in mood for that. So...any other s...
5570        1  The guy did some bitching but I acted like i'd...
5571        1                        Rofl. Its true to its name

[5572 rows x 2 columns]
```

```
X = df["Message"]
Y = df["Category"]
```

```
X
```

```
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: Message, Length: 5572, dtype: object
```

```
Y
```

```
0       1
1       1
2       0
3       1
4       1
       ..
5567    0
5568    1
5569    1
5570    1
5571    1
Name: Category, Length: 5572, dtype: object
```

```
X_train,X_test,Y_train,Y_test =
train_test_split(X,Y,test_size=0.2,random_state=2)
```

```
print(X.shape)
print(X_train.shape)
print(X_test.shape)
```

```
(5572,)
(4457,)
(1115,)
```

```
print(Y.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(5572,)
(4457,)
(1115,)
```

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)

Y_train = Y_train.astype("int")
Y_test = Y_test.astype("int")
```

Model training

```
model = LogisticRegression()
model.fit(X_train_features,Y_train)
```

```
LogisticRegression()
```

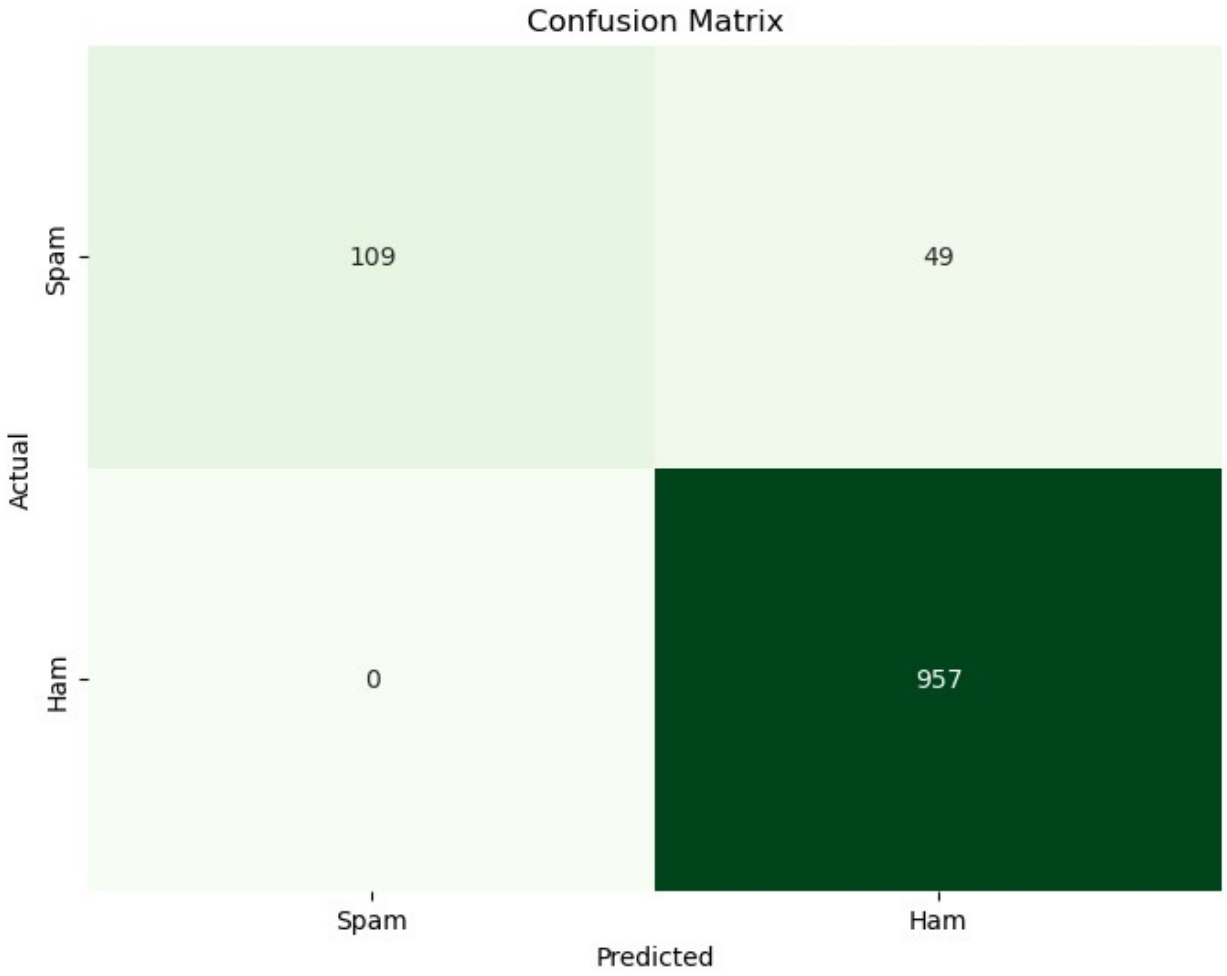
Model Evalution

```
prediction_on_training_data = model.predict(X_train_features)
accuracy_on_training_data = accuracy_score(Y_train,
prediction_on_training_data)

prediction_on_test_data = model.predict(X_test_features)
accuracy_on_test_data = accuracy_score(Y_test,
prediction_on_test_data)

# Print accuracy
print('Accuracy on training data: {}
%'.format(accuracy_on_training_data * 100))
print('Accuracy on test data: {} %'.format(accuracy_on_test_data *
100))
```

```
Accuracy on training data: 97.1729863136639 %
Accuracy on test data: 95.60538116591928 %
```

Confusion matrix

```
conf_matrix = confusion_matrix(Y_test, prediction_on_test_data)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Greens",
cbar=False,
            xticklabels=['Spam', 'Ham'], yticklabels=['Spam', 'Ham'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



Classification report

```
classification_rep = classification_report(Y_test,
prediction_on_test_data,)
print("Classification Report:")
print(classification_rep)
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.69      0.82       158
           1       0.95      1.00      0.98       957

    accuracy                           0.96      1115
   macro avg       0.98      0.84      0.90      1115
weighted avg       0.96      0.96      0.95      1115
```

```
TP = conf_matrix[1, 1]
TN = conf_matrix[0, 0]
FP = conf_matrix[0, 1]
FN = conf_matrix[1, 0]

accuracy = (TP + TN) / (TP + TN + FP + FN)
precision = TP / (TP + FP)
recall = TP / (TP + FN)
specificity = TN / (TN + FP)

print("Accuracy : ",accuracy)
print("Precision : ",precision)
print("Recall : ",recall)
print("Specificity : ",specificity)
```

```
Accuracy :  0.9560538116591928
Precision :  0.9512922465208747
Recall :  1.0
Specificity :  0.689873417721519
```

```
from sklearn.naive_bayes import MultinomialNB

# NAVE Byse

NVB_MDL = MultinomialNB()

NVB_MDL.fit(X_train_features,Y_train)
```

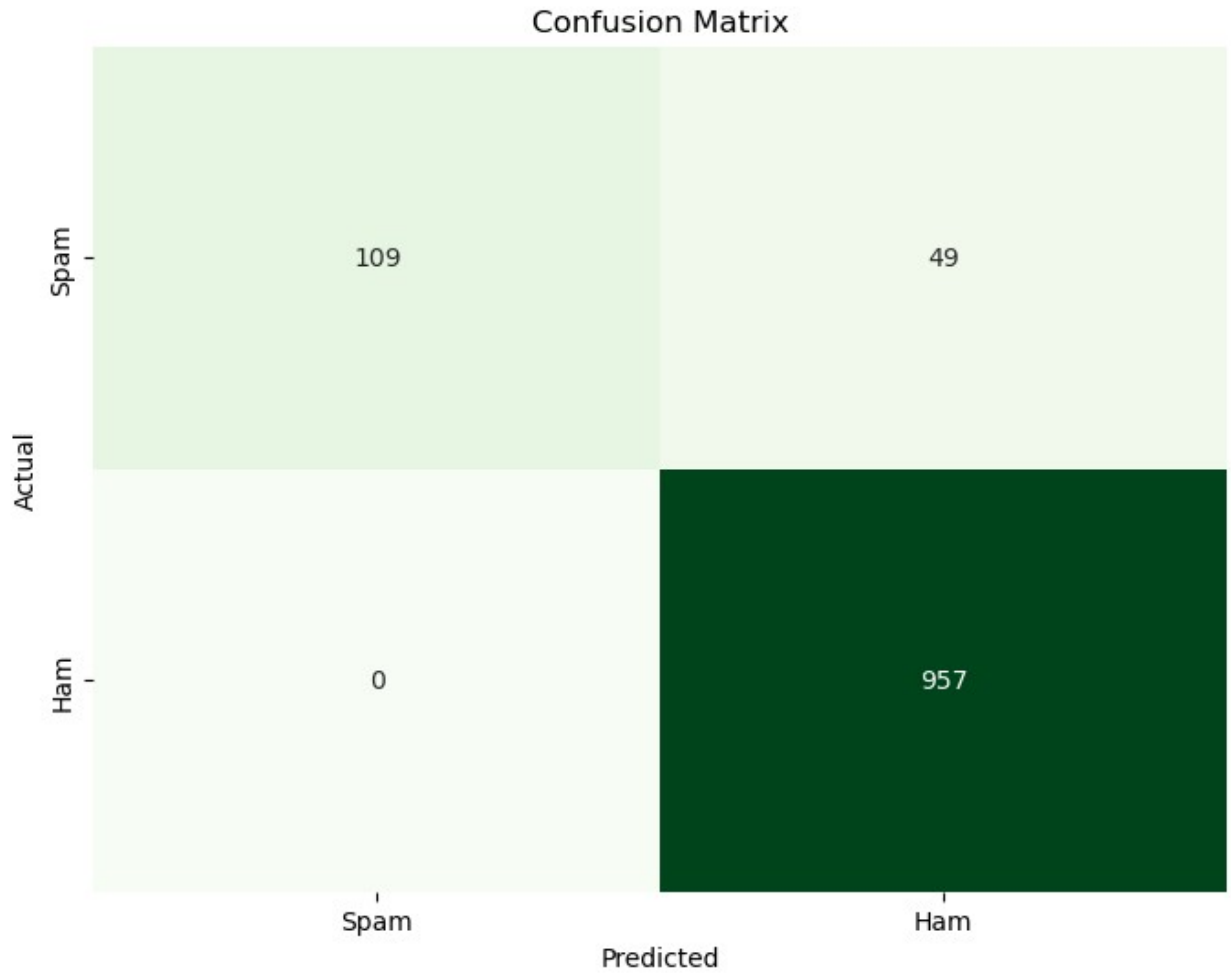
```
MultinomialNB()
```

```
prediction_on_NBtraining_data = model.predict(X_train_features)
accuracy_on_NBtraining_data = accuracy_score(Y_train,
prediction_on_NBtraining_data)

prediction_on_NBtest_data = model.predict(X_test_features)
accuracy_on_test_data = accuracy_score(Y_test,
prediction_on_NBtest_data)

# Print accuracy
print('Accuracy on training data: {}
%'.format(accuracy_on_NBtraining_data * 100))
```

```
print('Accuracy on test data: {} %'.format(accuracy_on_test_data *
100))
```

```
Accuracy on training data: 97.1729863136639 %
Accuracy on test data: 95.60538116591928 %
```

```
conf_matrix = confusion_matrix(Y_test, prediction_on_NBtest_data)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Greens",
cbar=False,
            xticklabels=['Spam', 'Ham'], yticklabels=['Spam', 'Ham'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



Test your model

```
input_your_mail = ["Congratulations! You have won a free vacation to an exotic destination. Click the link to claim your prize now!"]
input_data_features = feature_extraction.transform(input_your_mail)
prediction = model.predict(input_data_features)
print(prediction)


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

```
[0]
Spam Mail
```

```
input_your_mail = ["Meeting reminder: Tomorrow, 10 AM, conference room. See you there!"]
input_data_features = feature_extraction.transform(input_your_mail)
prediction = model.predict(input_data_features)
print(prediction)


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

```
[1]
Ham Mail
```