In [1]:
```
#Harini Rajarathinam
#Spam Email Classifier
```

In [15]:
```
import numpy as np
import pandas as pd
import string
import nltk
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import StratifiedKFold, GridSearchCV
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_
```

In [16]:
```
# Load data
df = pd.read_csv(r"C:\Users\rcher\Documents\Important docs\My Data Analytics pr

# Rename columns
df = df[['v1', 'v2']]
df = df.rename(columns={'v1': 'label', 'v2': 'message'})

# Preprocess text data
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    # Remove punctuation
    text = text.translate(str.maketrans('', '', string.punctuation))
    # Convert to lowercase
    text = text.lower()
    # Remove stopwords
    text = ' '.join([word for word in text.split() if word not in stop_words])
    return text

df['message'] = df['message'].apply(preprocess_text)
```

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

In [17]:
```
# Vectorize text data
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(df['message'])

# Convert labels to binary values
y = np.where(df['label'] == 'spam', 1, 0)
```

In [18]:
```
# Split data into training and testing sets using stratified k-fold cross-vali
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
for train_index, test_index in skf.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

# Tune hyperparameters
param_grid = {'alpha': [0.1, 0.5, 1.0, 2.0]}
clf = GridSearchCV(MultinomialNB(), param_grid, cv=skf, scoring='f1_macro')
clf.fit(X_train, y_train)

# Test the model
y_pred = clf.predict(X_test)
```

In [21]:
```
for value in y_pred:
    print(value)
```

```
0
1
0
1
1
0
0
0
0
1
0
0
0
0
0
0
1
0
0
0
```

In [20]:
```
# Evaluate the model using multiple metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
auc_roc = roc_auc_score(y_test, y_pred)

print(f"Accuracy: {accuracy:.2%}")
print(f"Precision: {precision:.2%}")
print(f"Recall: {recall:.2%}")
print
```

```
Accuracy: 97.40%
Precision: 90.00%
Recall: 90.60%
```

Out[20]:
```
<function print>
```