In [1]:
```
import pandas as pd
import re #regular expressions
import nltk #natural language tool kit
```

In [2]:
```
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer #used to reduce words to the root forms
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
```

In [3]:
```
# Download Stopwords
nltk.download("stopwords")
stemmer = PorterStemmer()
stop_words = set(stopwords.words("english"))
```

```
[nltk_data] Downloading package stopwords to C:\Users\Melvin
[nltk_data]     Chalwa\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

Preprocessing data

In [4]:
```
# Load dataset
df = pd.read_csv("Downloads/spam.csv", encoding="latin-1")[["v1", "v2"]]
df.columns = ["label", "message"]
df["label"] = df["label"].map({"ham": 0, "spam": 1})
df.head()
```

Out[4]:

	label	message
0	0	Go until jurong point, crazy.. Available only ...
1	0	Ok lar... Joking wif u oni...
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
3	0	U dun say so early hor... U c already then say...
4	0	Nah I don't think he goes to usf, he lives aro...

In [5]:
```
def preprocess_text(text):
    text = re.sub(r"\W", " ", text)  # Remove special characters
    text = text.lower()  # Convert to lowercase
    words = text.split()
    words = [stemmer.stem(word) for word in words if word not in stop_words]  # Remove stop
    return " ".join(words)
```

In [6]:
```
df["cleaned_message"] = df["message"].apply(preprocess_text)
print(df.head())
```

```
   label                                            message  \
0      0  Go until jurong point, crazy.. Available only ...
1      0                      Ok lar... Joking wif u oni...
2      1  Free entry in 2 a wkly comp to win FA Cup fina...
3      0  U dun say so early hor... U c already then say...
4      0  Nah I don't think he goes to usf, he lives aro...

                                  cleaned_message
0  go jurong point crazi avail bugi n great world...
1                             ok lar joke wif u oni
2  free entri 2 wkli comp win fa cup final tkt 21...
3               u dun say earli hor u c alreadi say
4              nah think goe usf live around though
```

Training an ML Model

In [7]:
```
from sklearn.feature_extraction.text import TfidfVectorizer #converts text to numerical for
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
```

In [8]:
```
vectorizer = TfidfVectorizer(max_features=3000)
X = vectorizer.fit_transform(df["cleaned_message"])
y = df["label"]
```

In [9]:
```
# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [10]:
```
# Fit logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)
```

Out[10]:
```
LogisticRegression
(https://scikit-learn.org/1.6/modules/generated/sklearn.linear_model.LogisticRegression.html)
LogisticRegression()
```

Evaluating Model Perfomance

In [11]:
```
y_pred = model.predict(X_test)
print(f"Accuracy: {accuracy_score(y_test, y_pred) * 100:.2f}%")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 95.43%
              precision    recall  f1-score   support

           0       0.95      1.00      0.97       965
           1       0.96      0.69      0.80       150

    accuracy                           0.95      1115
   macro avg       0.96      0.84      0.89      1115
weighted avg       0.95      0.95      0.95      1115
```

In [12]:
```python
def predict_email(email_text):
    processed_text = preprocess_text(email_text)
    vectorized_text = vectorizer.transform([processed_text])
    prediction = model.predict(vectorized_text)
    return "Spam" if prediction[0] == 1 else "Not Spam"
```

In [13]:
```python
email = "Congratulations! You've won a free iPhone. Click here to claim now."
print(f"Email: {email}\nPrediction: {predict_email(email)}")
```

```
Email: Congratulations! You've won a free iPhone. Click here to claim now.
Prediction: Spam
```