Importing Libraries

```
import numpy as np
import pandas as pd
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
```

Data Collection and Pre_Processing

```
raw_mail_data = pd.read_csv("/content/mail_data.csv")
raw_mail_data.head()
```

	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...

Next steps: Generate code with `raw_mail_data` View recommended plots

Replace the null values with a null string

```
mail_data = raw_mail_data.where((pd.notnull(raw_mail_data)),'')
```

```
mail_data.head()
```

	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...

Next steps: Generate code with `mail_data` View recommended plots

```
mail_data.shape
```

```
(5572, 2)
```

Label Encoding

```
# Spam as 0 and ham as 1

mail_data.loc[mail_data['Category'] == 'spam', 'Category'] = 0
mail_data.loc[mail_data['Category'] == 'ham', 'Category'] = 1
```

```
# seprating the data as texts and label

X = mail_data['Message']
Y = mail_data['Category']
```

Splitting the data into training data and test data

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=3)
```

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

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

Feature Extraction

```
# transform the text data to feature vectors that can be used as input to the logistic regression

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)

# convert Y_train and Y_test values as integers

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

Train the Model

Logistic Regression

```
model = LogisticRegression()
```

```
# training the logistic regression with the training data
model.fit(X_train_features, Y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

Evaluating the trained model

```
# prediction on training data

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

```
print('Accuracy on training data : ', accuracy_on_training_data)
```

```
Accuracy on training data :  0.9670181736594121
```

```
# prediction on test data

prediciton_on_test_data = model.predict(X_test_features)
accuracy_on_test_data = accuracy_score(Y_test, prediciton_on_test_data)
```

```
print('Accuracy on test data : ', accuracy_on_test_data)
```

```
Accuracy on test data :  0.9659192825112107
```

Buliding a Predictive System

```
input_mail = ["I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fu
```

```
# convert text to feature vectors
input_data_features = feature_extraction.transform(input_mail)

# making prediciton
prediction = model.predict(input_data_features)
print(prediction)

if prediction[0] == 1:
  print('Ham mail')
else:
  print('Spam mail')
```

```
[1]
Ham mail
```