

```
In [2]: import pandas as pd
import numpy as np
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
```

```
In [3]: # Loading the data from csv file to a pandas Dataframe
raw_mail_data = pd.read_csv(r"C:\Users\annu\Downloads\mail_data.csv")
```

```
In [5]: raw_mail_data.sample(10)
```

	Category	Message
4675	ham	I'm leaving my house now.
929	ham	Oh all have to come ah?
2607	ham	R U &SAM P IN EACHOTHER. IF WE MEET WE CAN GO ...
614	ham	THANX4 TODAY CER IT WAS NICE 2 CATCH UP BUT WE...
1194	spam	Sex up ur mobile with a FREE sexy pic of Jorda...
650	spam	You have won ?1,000 cash or a ?2,000 prize! To...
152	ham	Yup i thk cine is better cos no need 2 go down...
3492	ham	Ok.
4871	ham	Dip's cell dead. So i m coming with him. U bet...
3253	ham	Come to medical college at 7pmforward i...

```
In [9]: raw_mail_data.isnull().sum()
```

```
Out[9]: Category    0
Message    0
dtype: int64
```

```
In [12]: # replace the null values with a null string
raw_mail=raw_mail_data.where((pd.notnull(raw_mail_data)), '')
```

```
In [13]: raw_mail.shape
```

```
Out[13]: (5572, 2)
```

```
In [14]: raw_mail.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...

```
In [18]: #Label Encoding for target column
# Label spam mail as 0; ham mail as 1;
raw_mail.loc[raw_mail['Category']=='spam','Category']=0
raw_mail.loc[raw_mail['Category'] == 'ham', 'Category'],] = 1
```

```
In [19]: # separating the data as texts and Label
```

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

```
In [20]: print(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 ü 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
```

```
In [21]: print(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
```

Splitting the data into training data & test data

```
In [22]: X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2, random_state=3)
```

```
In [23]: print(X.shape)
print(X_train.shape)
print(X_test.shape)
```

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

Feature Extraction

```
In [25]: # 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')
```

```
In [26]: print(X_train)
```

```
3075      Don know. I did't msg him recently.
1787    Do you know why god created gap between your f...
1614              Thnx dude. u guys out 2nite?
4304              Yup i'm free...
3266    44 7732584351, Do you want a New Nokia 3510i c...
...
789      5 Free Top Polyphonic Tones call 087018728737,...
968    What do u want when i come back?.a beautiful n...
1667    Guess who spent all last night phasing in and ...
3321    Eh sorry leh... I din c ur msg. Not sad ahead...
1688    Free Top ringtone -sub to weekly ringtone-get ...
Name: Message, Length: 4457, dtype: object
```

```
In [27]: print(X_train_features)
```

```
(0, 5413)    0.6198254967574347
(0, 4456)    0.4168658090846482
(0, 2224)    0.413103377943378
(0, 3811)    0.34780165336891333
(0, 2329)    0.38783870336935383
(1, 4080)    0.18880584110891163
(1, 3185)    0.29694482957694585
(1, 3325)    0.31610586766078863
(1, 2957)    0.3398297002864083
(1, 2746)    0.3398297002864083
(1, 918)     0.22871581159877646
(1, 1839)    0.2784903590561455
(1, 2758)    0.3226407885943799
(1, 2956)    0.33036995955537024
(1, 1991)    0.33036995955537024
(1, 3046)    0.2503712792613518
(1, 3811)    0.17419952275504033
(2, 407)     0.509272536051008
(2, 3156)    0.4107239318312698
(2, 2404)    0.45287711070606745
(2, 6601)    0.6056811524587518
(3, 2870)    0.5864269879324768
(3, 7414)    0.8100020912469564
(4, 50)      0.23633754072626942
(4, 5497)    0.15743785051118356
:           :
(4454, 4602) 0.2669765732445391
(4454, 3142) 0.32014451677763156
(4455, 2247) 0.37052851863170466
(4455, 2469) 0.35441545511837946
(4455, 5646) 0.33545678464631296
(4455, 6810) 0.29731757715898277
(4455, 6091) 0.23103841516927642
(4455, 7113) 0.30536590342067704
(4455, 3872) 0.3108911491788658
(4455, 4715) 0.30714144758811196
(4455, 6916) 0.19636985317119715
(4455, 3922) 0.31287563163368587
(4455, 4456) 0.24920025316220423
(4456, 141)  0.292943737785358
(4456, 647)  0.30133182431707617
(4456, 6311) 0.30133182431707617
(4456, 5569) 0.4619395404299172
(4456, 6028) 0.2103488800987115
(4456, 7154) 0.24083218452280053
(4456, 7150) 0.3677554681447669
(4456, 6249) 0.17573831794959716
(4456, 6307) 0.2752760476857975
(4456, 334)  0.2220077711654938
(4456, 5778) 0.16243064490100795
(4456, 2870) 0.31523196273113385
```

Training the Model with Logistic Regression

```
In [28]: model = LogisticRegression()
```

```
In [29]: # training the Logistic Regression model with the training data
model.fit(X_train_features, Y_train)
```

```
Out[29]: LogisticRegression
LogisticRegression()
```

```
In [30]: # prediction on training data
```

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

```
In [31]: print('Accuracy on training data : ', accuracy_on_training_data)
```

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

```
In [32]: # prediction on test data
```

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

```
In [33]: print('Accuracy on test data : ', accuracy_on_test_data)
```

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

Building a Predictive System

```
In [38]: input_mail = ["You have won ?1,000 cash or a ?2,000 prize! To..."]
# convert text to feature vectors
input_data_features = feature_extraction.transform(input_mail)
```

```
# making prediction
```

```
prediction = model.predict(input_data_features)
print(prediction)
```

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

else:
    print('Spam mail')
```

```
[0]
Spam mail
```

```
In [ ]:
```