Spam Mail

PREDICTION

-By Avinandan Kumar Sharma

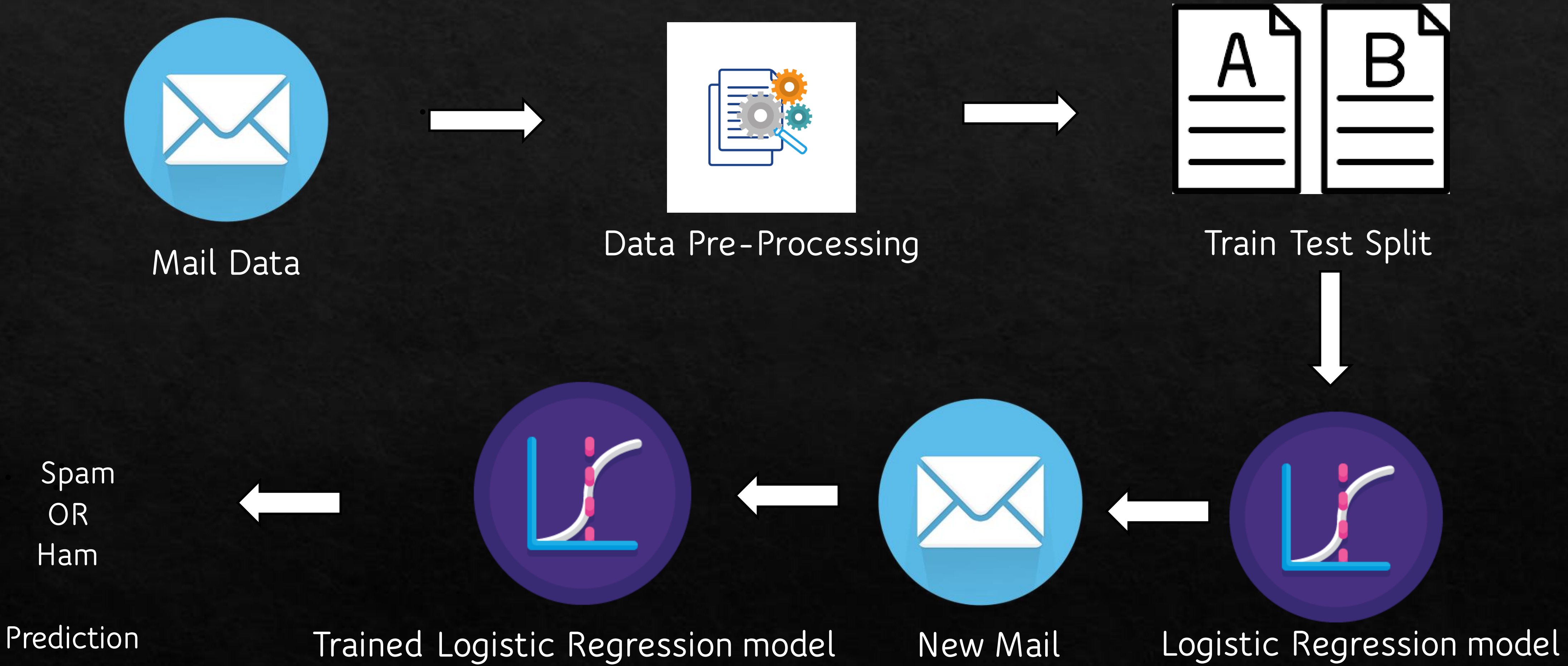
About the Project

- This is a project I am working on while learning concepts of data science and machine learning. The goal here is to identify whether an email is spam or ham. We will take a dataset of labeled email messages and apply classification techniques. We can later test the model for accuracy and performance on unclassified email messages. Similar techniques can be applied to other NLP applications like sentiment analysis etc.

Ham or Spam

- Spam e-mail has become a very serious problem. especially for teaching purposes, is to build models to predict if a message is spam or not. Our dataset called Spam contains the subject lines and the target which takes values `0` and `1` for ham and spam respectively.

Work Flow



Mail Data

Data Pre-Processing

Train Test Split

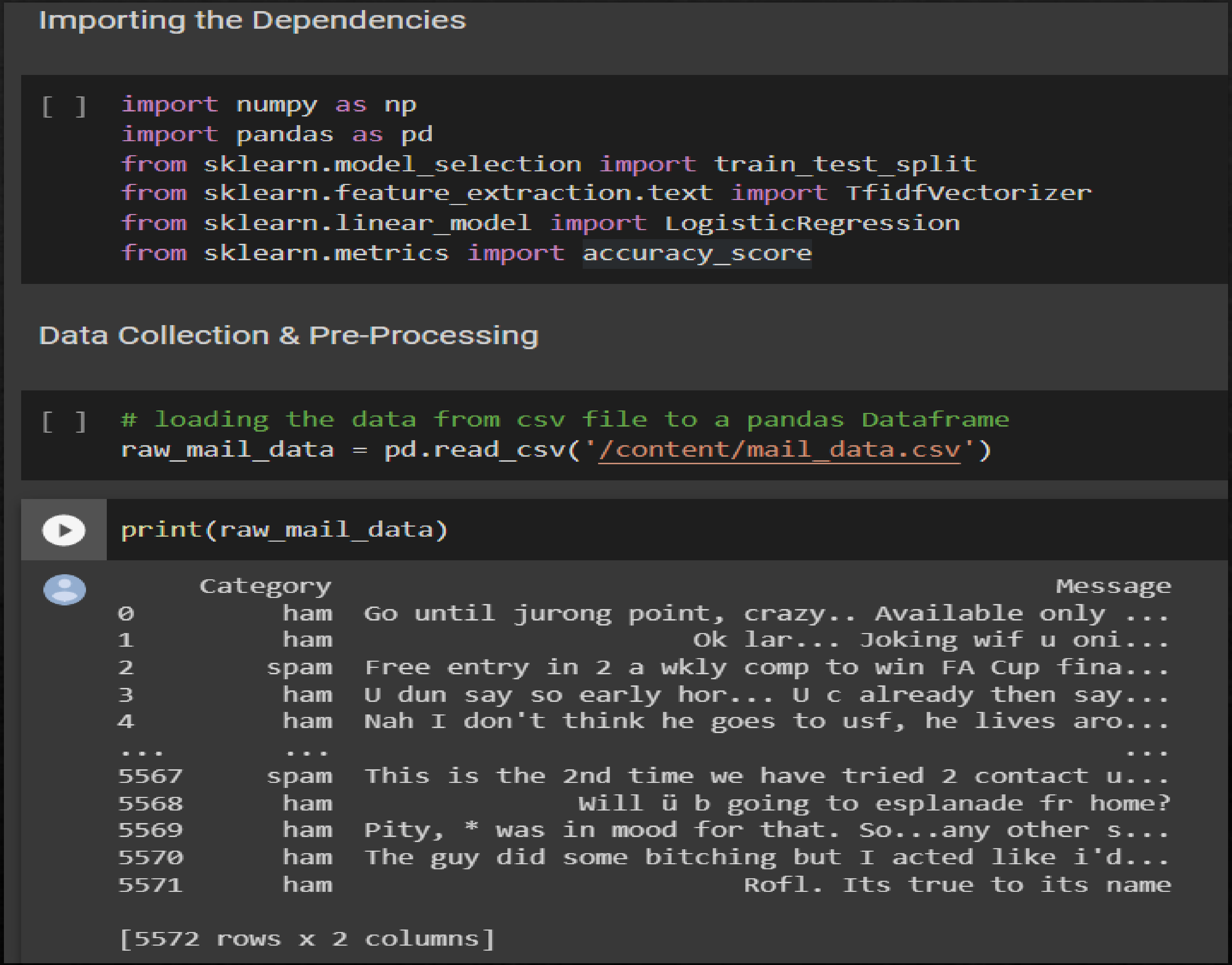
Logistic Regression model

New Mail

Trained Logistic Regression model

Spam
OR
Ham

Prediction



Importing the Dependencies

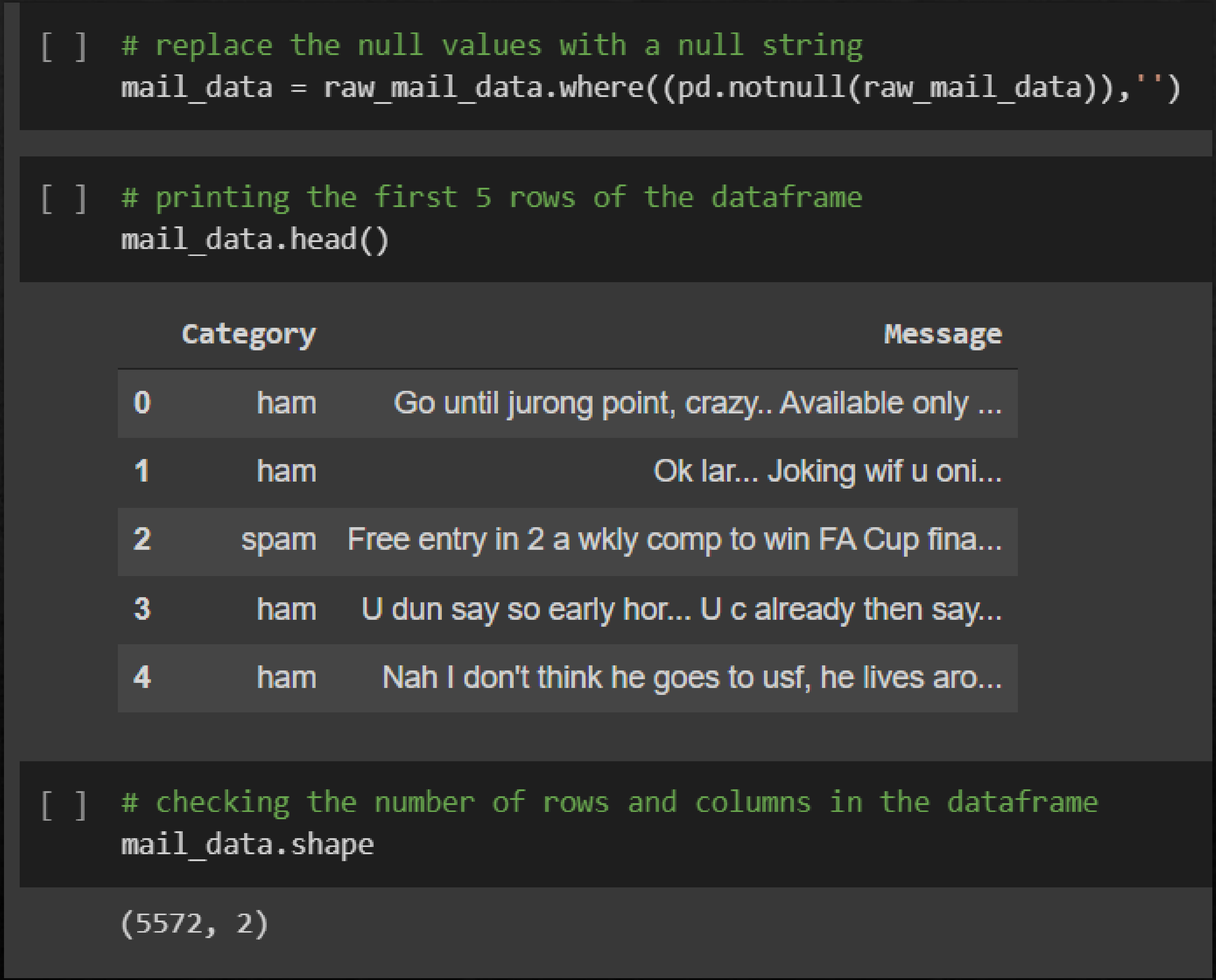
```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

Data Collection & Pre-Processing

```python
# loading the data from csv file to a pandas Dataframe
raw_mail_data = pd.read_csv('/content/mail_data.csv')
```

```python
print(raw_mail_data)
```

```
     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 ü 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]
```



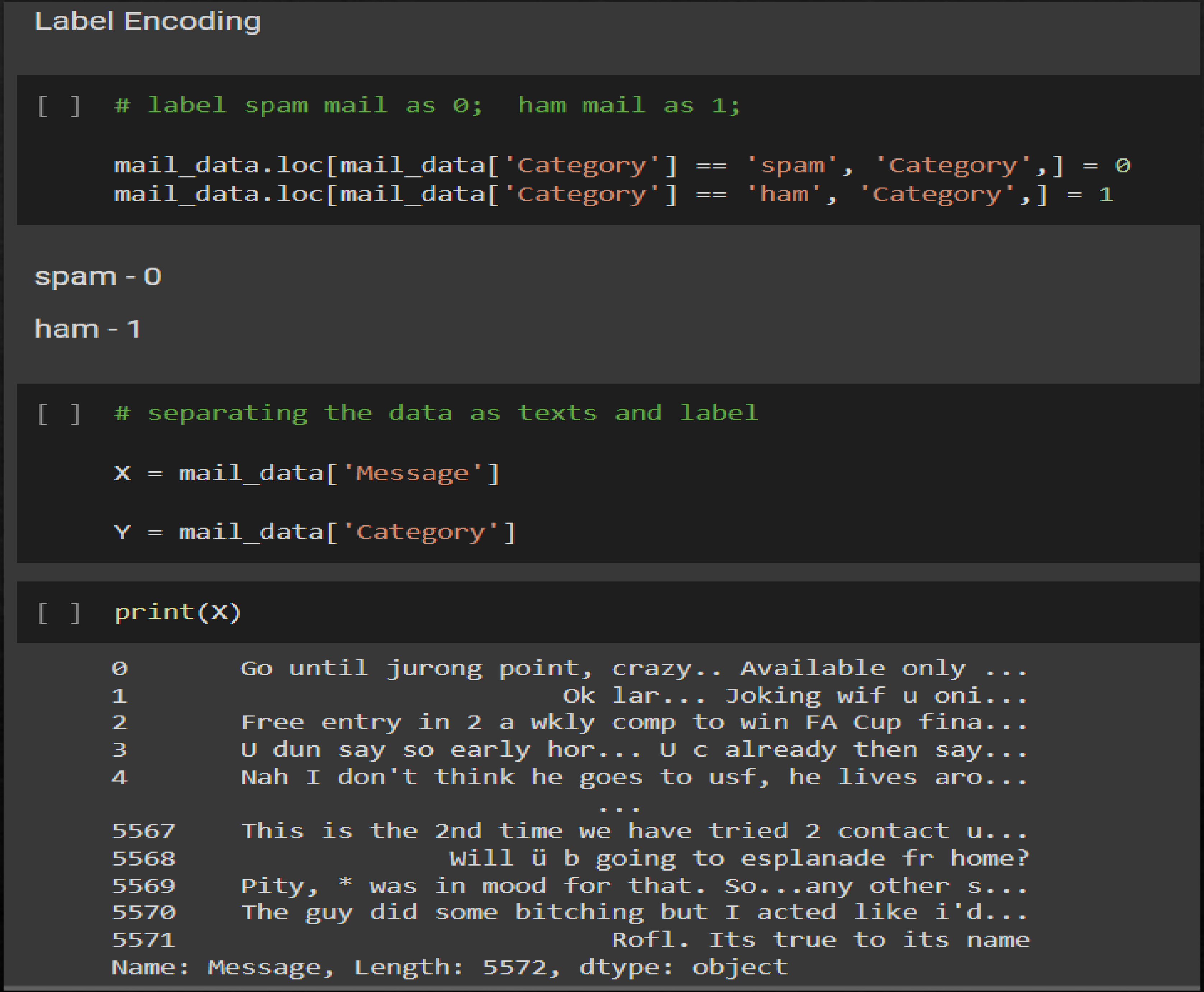
```
# replace the null values with a null string
mail_data = raw_mail_data.where((pd.notnull(raw_mail_data)),'')
```

```
# printing the first 5 rows of the dataframe
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...

```
# checking the number of rows and columns in the dataframe
mail_data.shape
```

```
(5572, 2)
```



Label Encoding

```
[ ]  # label spam mail as 0;  ham mail as 1;

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

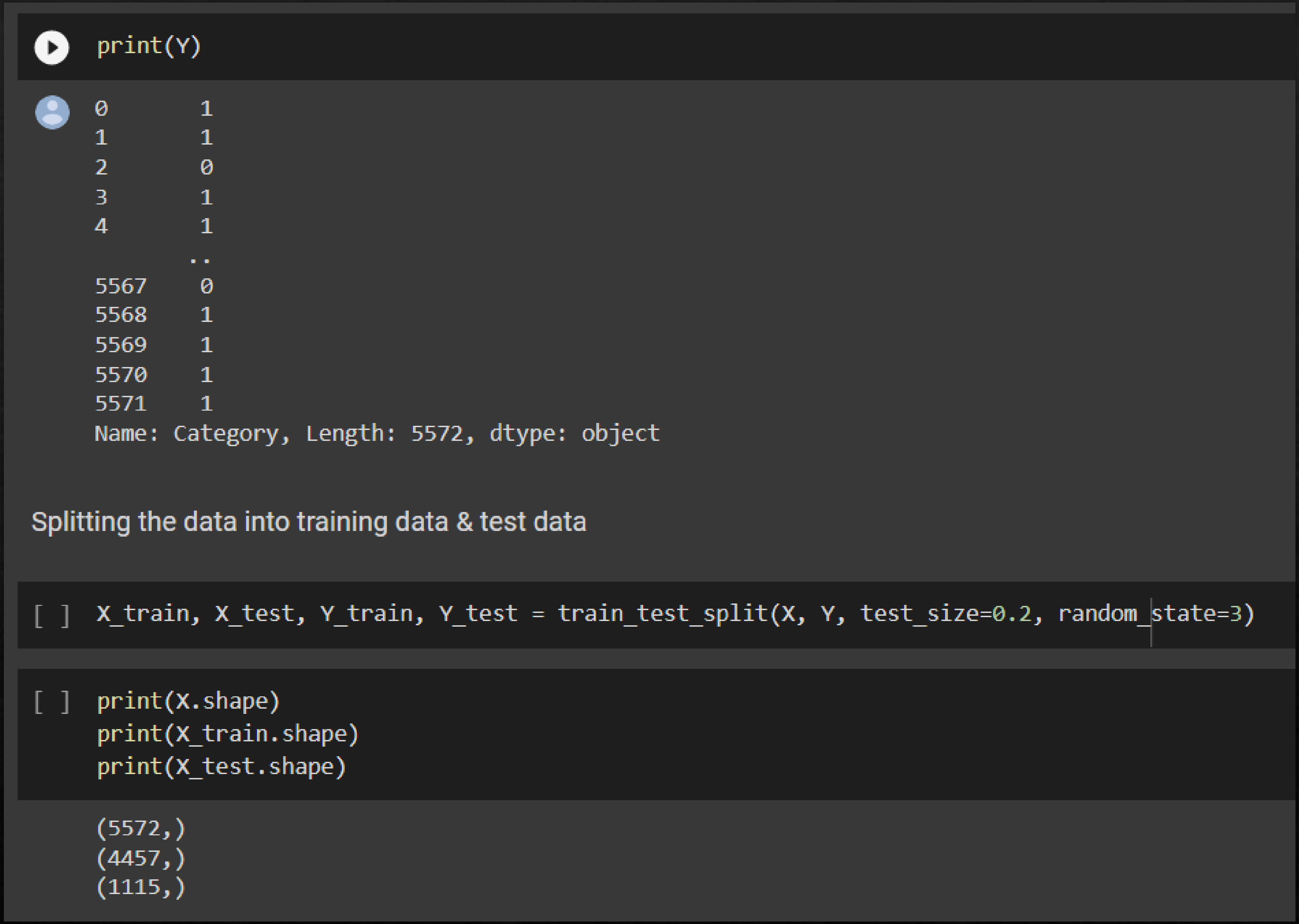
spam - 0

ham - 1

```
[ ]  # separating the data as texts and label

     X = mail_data['Message']

     Y = mail_data['Category']
```

```
[ ]  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
```



```
print(Y)
```

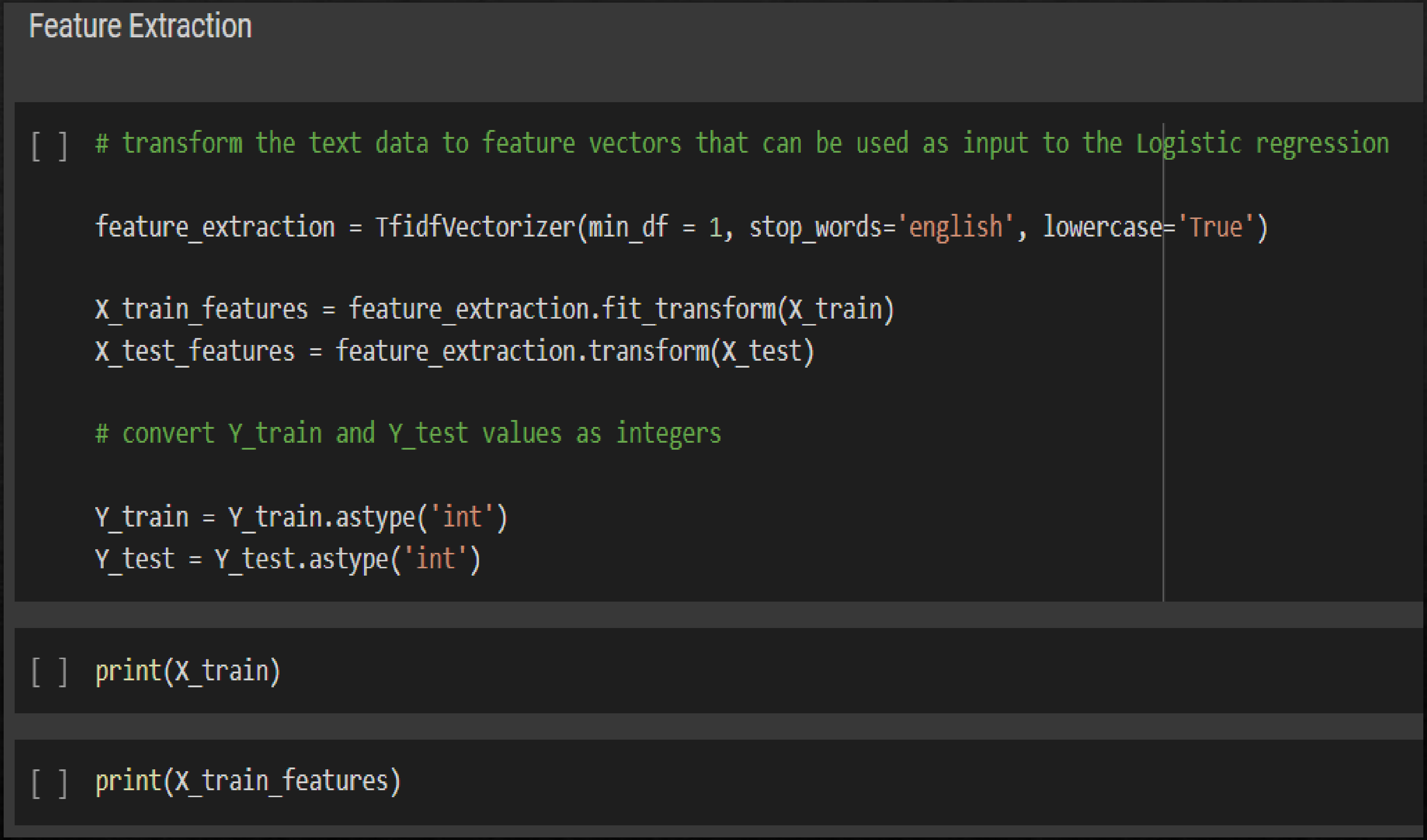
```
0       1
1       1
2       0
3       1
4       1
       ..
5567    0
5568    1
5569    1
5570    1
5571    1
Name: Category, Length: 5572, dtype: object
```

Splitting the data into training data & test data

```
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,)
```

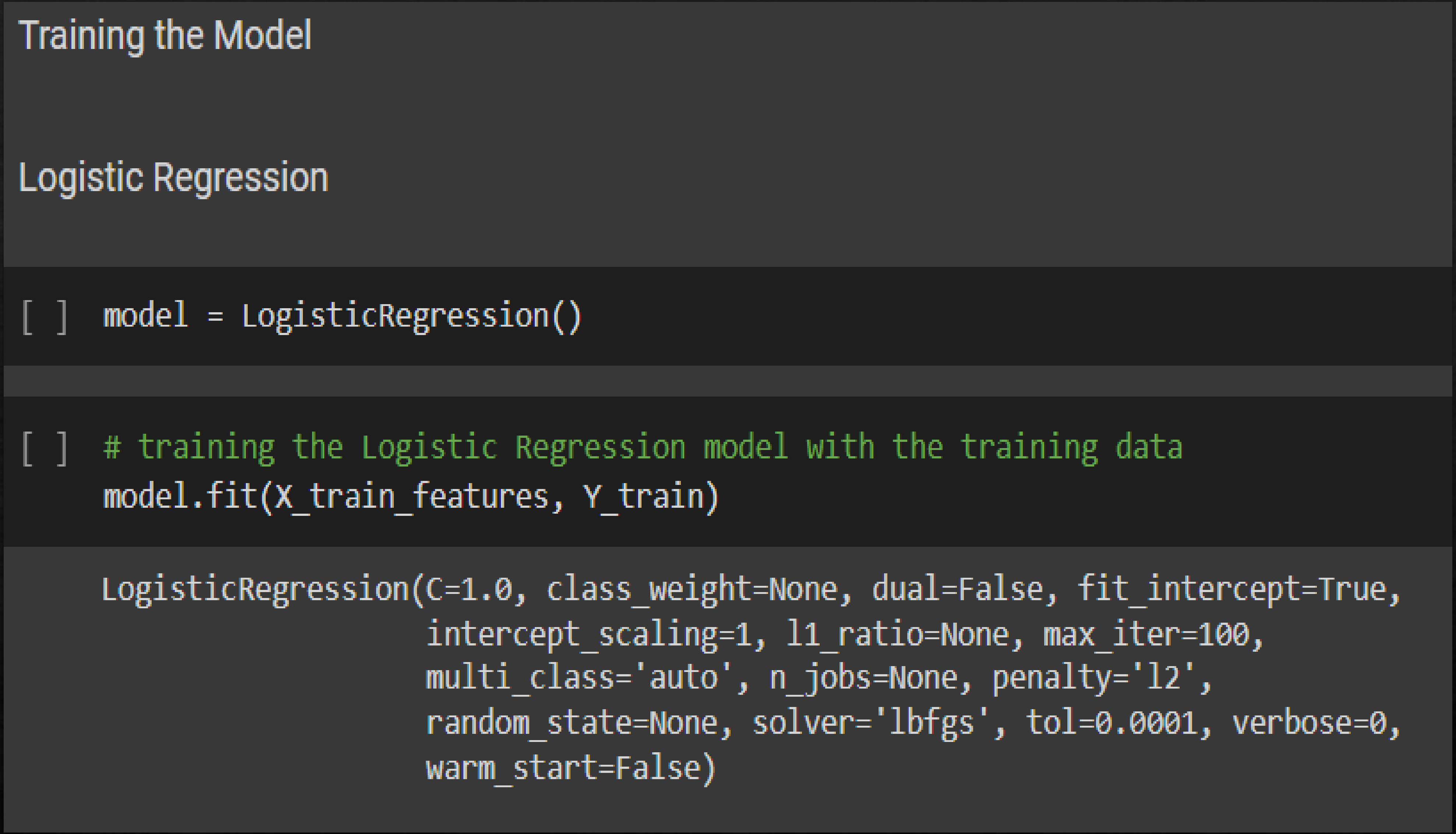


Feature Extraction

```
# 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')
```

```
print(X_train)
```

```
print(X_train_features)
```



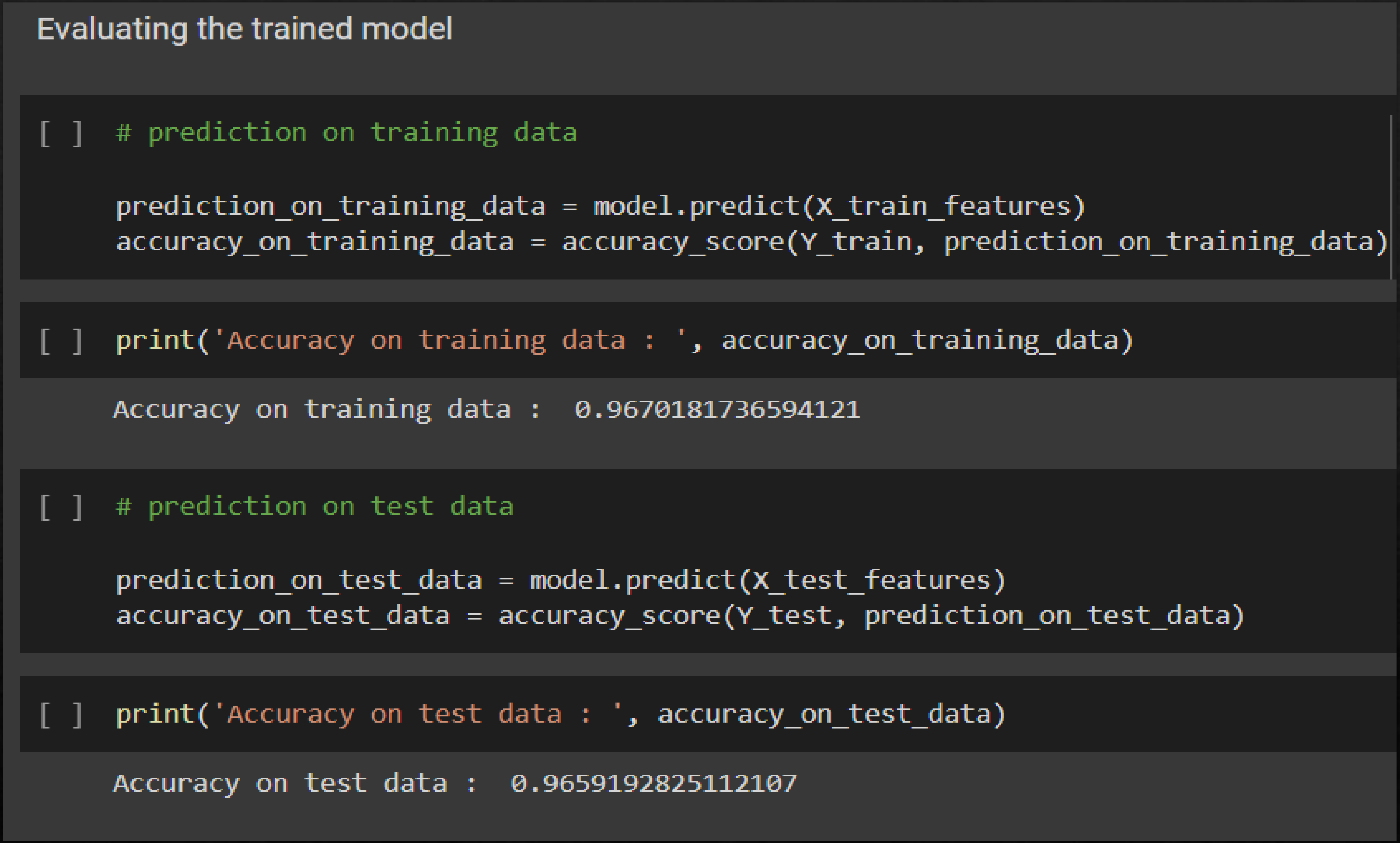
Training the Model

Logistic Regression

```
model = LogisticRegression()
```

```
# training the Logistic Regression model with the training data
model.fit(X_train_features, Y_train)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```



Evaluating the trained model

```
[ ] # 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)
```

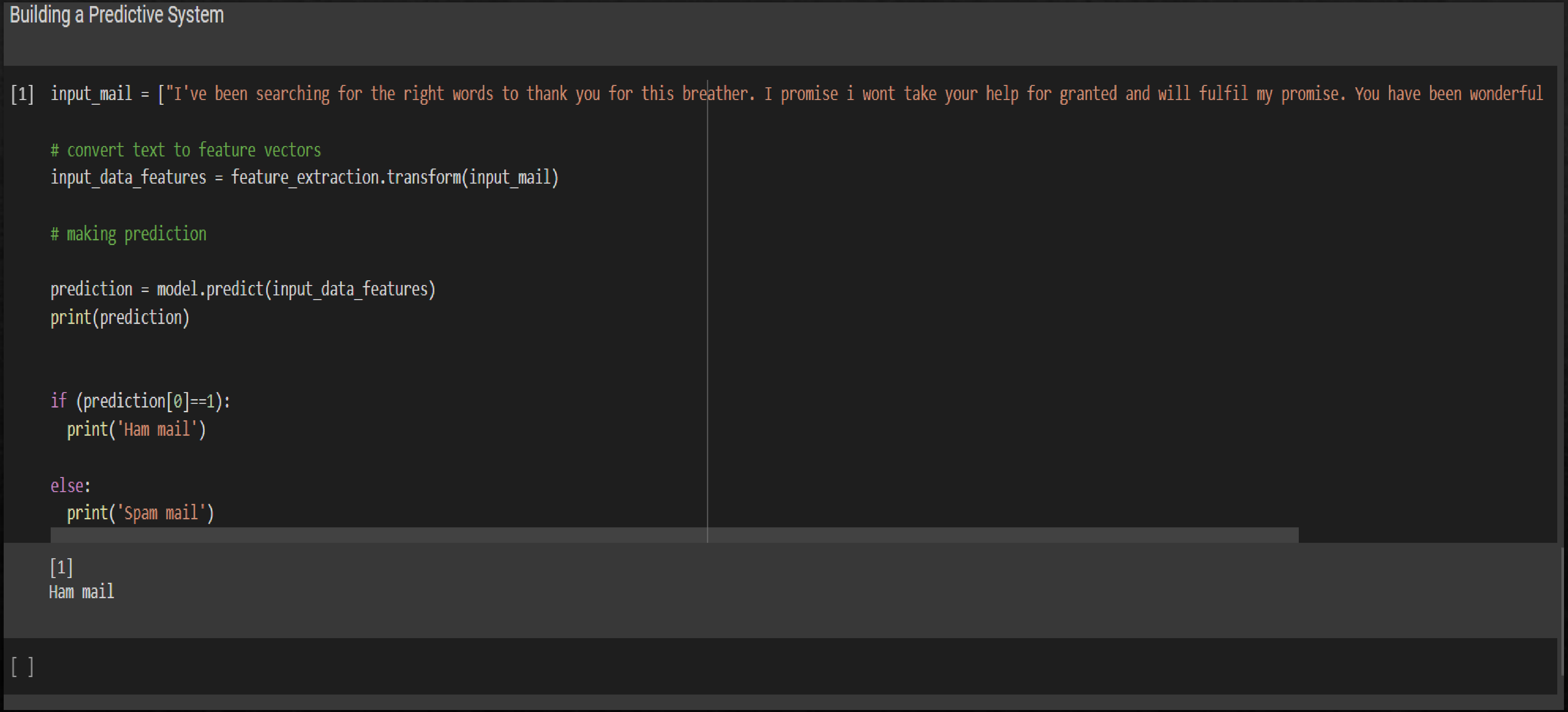
```
[ ] print('Accuracy on training data : ', accuracy_on_training_data)
```

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

```
[ ] # 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)
```

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

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



Building a Predictive System

```
[1] 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 fulfil my promise. You have been wonderful

    # 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')
```

```
[1]
Ham mail
```

```
[ ]
```

Discussion

- We provided a practical and reproducible example of how you can build a decent Ham or Spam algorithm. This is one of the main tasks in the field of NLP. Our model achieved an **AUC score of 95%** on the test dataset which is really good. We were also able to add features and also to identify the features which are more likely to appear in a Spam email and vice versa.