Email Spam Classification

Data Science With Python Lab Project Report

Bachelor

in

Computer Science

By

Harshitha Chandaka , Kavya Varanasi

S190987

S190406

Rajiv Gandhi University Of Knowledge And Technologies

S.M. Puram , Srikakulam -532410

Andhra Pradesh, India

Abstract

Email Spam Classification

Nowadays Email Spam has become a major problem, with rapid growth of internet users, "Email Spams" are also increasing. People are using them for illegal and unethical conducts and fraud. Sending malicious links through spam emails which can harm our system and can also seek into your system. Creating a fake profile and email account is much easy for the spammers, these spammers target those peoples who are not aware about these frauds. By sending a spam mails they will access the device or flirt the user with the messages they sent.

So, it is needed to identify those spam mails which are fraud. This project will identify those spam by using techniques of machine learning, this project will discuss the machine learning algorithms and apply all those methods on our data sets. Using the type of methods on dataset which are the collection of the mails that the user received, we can predict whether the mail is spam or not.

Email or electronic mail spam refers to the "using of email to send unsolicited emails or advertising emails to a group of recipients. Unsolicited emails mean the recipient has not granted permission for receiving those emails. This type of ideas can help the users to overcome the spam mails they receive and they can identify the spam mails.

Contents

Chapter 1

Introduction

1.1 Introduction to the Project

Here we are working on project called Classification of Email Spam. In our daily life, we are using so-many communication platforms that helps us to get an information like sharing files, media etc. So by increasing the social media platforms, simultaneously there is also increasing of unwanted messages by unknown users. This type of spams misleads the users(spammers) details. This type of frauds may interrupt in your devices through various platforms. So here we particularly focused on "Emails" which we regularly using in our daily life.

In our daily life we are using Email regularly for any official or regular work purpose. With the ever-increasing volume of email traffic, it has become essential to automatically identify and filter out unsolicited and potentially harmful emails known as spam.

We sometimes come-across with some spam mails. Spam emails not only clutter users inboxes but also pose risks such as phishing attempts, malware distribution, and fraud.If we are unknowingly accessed it, may get some fraud attentions on our device. So by identifying the spam mails in our Emails, we can easily get rid off from the spam mails.

So we are introducing a Solution for this problem called "Classification of Email Spam". So here we can identify how many Spam and Non-Spam mails in our data set. The primary objective of email spam classification is to accurately distinguish between non-spam emails and spam emails.

This information might be useful to detect the spam or ham mails, so that helps us to know how we can secure our mail from this fraud messages and also can improve email communication and security. This project may helps users to aware on spam mails.

We consider a data-set of having emails with declaration of spam or not spam. By using some of the techniques in machine learning we are classifying a data set.

1.2 Applications:

1.Email Filtering: Spam classification helps in filtering out unwanted and malicious emails from users' inboxes, ensuring that legitimate emails reach the users' attention. By accurately classifying emails as spam or non-spam, email providers can enhance the overall user experience and reduce the risk of users falling victim to phishing attacks or scams.

2.Fraud Detection: Email spam classification can be utilized as a part of fraud detection systems. It helps identify fraudulent emails, such as phishing emails that attempt to trick users into revealing sensitive information like passwords or credit card details. By accurately detecting such fraudulent emails, organizations can protect their users and prevent financial losses.

3.Malware Detection: Spam emails are often used as a delivery mechanism for malware. By classifying emails as spam, organizations can identify potentially harmful attachments or links and take appropriate actions to prevent malware infections or network breaches. This is particularly crucial for maintaining the security of corporate networks and sensitive data.

4.Content Moderation: Email spam classification can be employed for content moderation purposes in online platforms that allow users to communicate via email. By filtering out spam emails, platforms can ensure that users receive relevant and legitimate communications, thereby improving the quality of user interactions and reducing the risk of exposure to inappropriate or harmful content.

5.Productivity Enhancement: Effective spam classification helps individuals and organizations save time and effort by reducing the clutter in their email inboxes. By automatically segregating spam emails, users can focus on important and relevant emails, leading to increased productivity and improved workflow management.

6.Compliance and Legal Requirements: Certain industries, such as finance and healthcare, have strict regulatory compliance requirements regarding email communications. Spam classification helps ensure compliance with regulations by identifying and managing potentially non-compliant or malicious emails, thereby mitigating legal risks.

7.Customer Relationship Management: Spam classification can be utilized by businesses to improve their customer relationship management processes. By accurately categorizing emails, organizations can prioritize and respond promptly to customer inquiries and requests, enhancing customer satisfaction and engagement.

1.3 Motivation towards the project:

There are several motivations to make this project: **Improving user experience:** Email spam is a significant problem that affects millions of people worldwide. By developing effective spam detection algorithms, you can contribute to improving the user experience for email users. By reducing the amount of spam that reaches people's inboxes, you can help ensure that they have a more streamlined and enjoyable email experience.

Enhancing productivity: Spam emails often waste valuable time and resources as users have to sift through numerous unwanted messages. By developing an efficient spam detection system, you can help users save time and focus on more important tasks, increasing overall productivity.

Protecting against cyber threats: Spam emails are not just annoying; they can also carry various cyber threats such as malware, phishing attempts, and scams. By detecting and filtering out spam emails, you can contribute to protecting users from potential security risks and prevent them from falling victim to malicious activities.

Challenging technical problem: Developing a robust spam detection system can be a challenging technical problem that involves machine learning, natural language processing, and data analysis techniques. For individuals who enjoy tackling complex problems,

working on an email spam detection project can be intellectually stimulating and rewarding.

Contributing to research: Spam detection is an ongoing area of research, and by working on such a project, you can contribute to advancing the field. Developing innovative approaches, experimenting with new algorithms, and sharing your findings can help other researchers and practitioners in the email security domain.

Building a valuable skill set: Working on an email spam detection project provides an opportunity to gain practical experience in machine learning, data analysis, and software development. These skills are highly sought after in various industries, including cybersecurity, data science, and artificial intelligence, making this project a valuable addition to your professional portfolio.

1.4 *Problem Statememnt of the project:*

The goal of this project is to create a reliable and efficient email spam classification system that accurately identifies and filters out spam emails from users' inboxes, improving their overall email experience and reducing the risk of exposure to malicious content.

The main aims to develop an email spam classification system that efficiently distinguishes between spam and legitimate emails, providing users with a clutter-free inbox and protecting them from potential security threats. The system should be accurate, adaptable to different email environments, and optimize user experience by effectively filtering out unwanted and potentially harmful spam messages.

Chapter 2

Approach To The Project

2.1 Explain About the Project

An email spam classification project aims to develop a machine learning model that can automatically identify whether an email is spam or not based on its content. The two main components of the project are the "Category" and "Message" columns.

Category: This column represents the class or category of an email, indicating whether it is spam or not. In most cases, the category column contains binary values, where 'spam' is typically represented by 0 and 'ham' (non-spam) is represented by 1. The goal is to train a model that can accurately predict the category of unseen emails.

Message: This column contains the textual content of the email. It includes the subject line, body text, and any other relevant information. The text in this column serves as the primary input for training the classification model.

Importing Libraries: The code begins by importing the required libraries such as numpy, pandas, matplotlib, and various modules from the sklearn library.

Data Loading: The dataset, which includes the "Category" and "Message" columns, is loaded into a suitable data structure such as a pandas DataFrame. The data may be obtained from various sources, such as pre-labeled datasets or collected email samples.

Data Preprocessing: Before training the model, it's necessary to preprocess the text data in the "Message" column. This step may involve tasks such as removing special characters, converting text to lowercase, tokenizing the text into individual words, and removing

stopwords (common words that carry little meaning).

Model development: Machine learning algorithms called Logestic Regression used to trained on the labeled dataset to learn the patterns and characteristics of spam emails. The chosen model is trained using the selected features and the corresponding labels.

Feature Extraction: In order to train a machine learning model, the text data needs to be converted into numerical features. This is typically done using techniques like TF-IDF (Term Frequency-Inverse Document Frequency). These methods transform the text data into a numerical representation that captures important characteristics and patterns.

Model Evaluation: After training, the model's performance is evaluated using evaluation metrics such as accuracy score. By building an email spam classification model using the "Category" and "Message" columns, it becomes possible to automate the identification of spam emails, saving time and resources for users and organizations and improving the overall email filtering and security.

2.2 About Data Set

We took a dataset called "mail_data.csv", which contains 2 columns namely Category and Messages and 5572 rows of data. Here the Message column defines the text that present in the mails which we generally called it as subject and body of the mails. Whereas the Category column defines whether the message is spam or ham. Ham is nothing but Not spam.

Further we replace the null values in the message column will be replaced with empty string for better training. We classify the Spam and Ham as 0 and 1. Where 0 is for spam and 1 is for ham. Later we divide the dataset for training and testing using train_test_split method which is importing from sklearn library.

The dataset will slit into two parts for training and testing on our declaration of "test_size" and "random_state". Based on the test_size the remaining data will came under the train dataset. Now the dataset will train to get some accuracy score. To predict the accuracy score we have to use some classification technique. Here we used a technique called Logestic Regression on our dataset. By this process we are using our dataset which containing

messages and their categories.

2.3 Data Set

The Dataset contains two columns and 5527 rows or entries.The columns are Category and Message.Here message column contains subject and body of the mail which are in the form of string datatype.Another column is category which contains whether two categorical values they are spam and ham corresponding to the message.

Spam: Spam email refers to specially to unsolicited or unwanted email messages.These are sent for commercial purposes,intending to advertise products ,services pr promote fraudulent schemes.

Ham: It refers to non-spam emails opposite of spam it is used to represent the regular,desired emails that are not classified as spam.

2.4 Prediction technique

In our project we are using a Logestic Regression which is giving an accurate score to our project. Logestic Regression:Logistic regression is a popular statistical model used for binary classification problems. It is a supervised learning algorithm that is used to predict the probability of an event occurring based on input features. Despite its name, logistic regression is actually a classification algorithm, not a regression algorithm.

Here are some methods and techniques used to in our code:

The code imports necessary libraries such as numpy, pandas, matplotlib, sklearn, and specifically, the TfidfVectorizer from sklearn.feature_extraction.text, LogisticRegression from sklearn.linear_mo and accuracy_score from sklearn.metrics.

The code replaces any null values in the DataFrame with an empty string.

The 'Category' column in the DataFrame is modified to have binary values. The 'spam' category is replaced with 0, while all other categories (assumed to be non-spam) are replaced with 1.

The 'Message' column is assigned to the variable x, and the 'Category' column is assigned to the variable y.

Figure 2.1: Example Of Graph

The dataset is split into training and testing sets using the train_test_split function. 80% of the data is used for training, and 20% is used for testing. The random_state is set to 3 for reproducibility.

Feature extraction is performed using the TfidfVectorizer. It converts the text data in x into numerical feature vectors using the TF-IDF (Term Frequency-Inverse Document Frequency) representation. The vectorization is done separately for the training and testing sets.

The values in ytrain and ytest are converted to integers.

A logistic regression model is created using LogisticRegression().

The model is trained using the training data (x_train_f and y_train) using the fit method.

2.5 Graphs

code:

```
import matplotlib.pyplot as plt
plt.hist(mail_df['Category'], bins=10)
plt.xlabel('Category')
plt.ylabel('Frequency')
plt.title('Frequency of Spam and Ham')
plt.show()

import seaborn as sns
sns.violinplot(mail_df)
plt.xlabel('Category')
plt.ylabel('Frequency')
plt.title('Frequency of Spam and Ham')
plt.show()
```

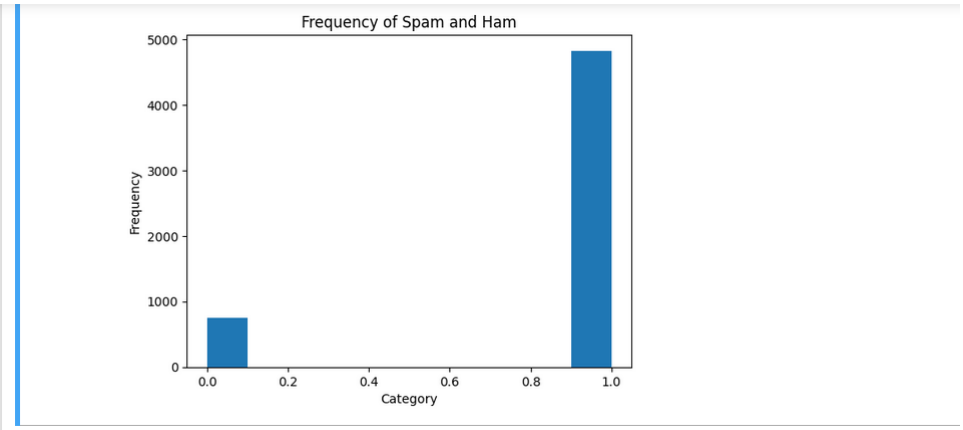

Figure 2.2: Hist plot

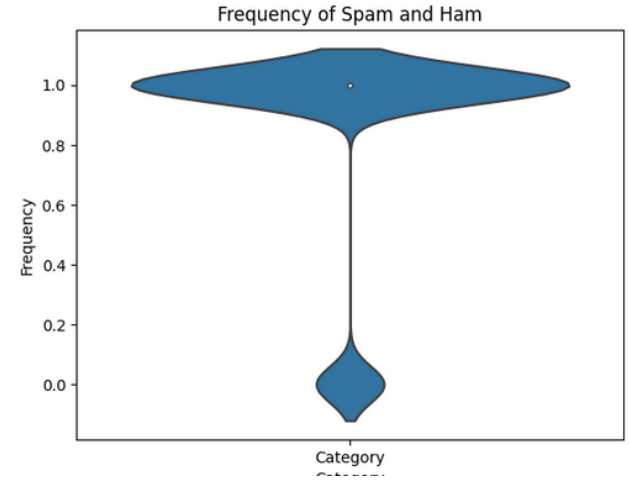

Figure 2.3: Violin Plot

2.6 Visualization

```
import seaborn as sn

sns.histplot(mail_df['Category'], kde=True)

plt.xlabel('Category')

plt.ylabel('Frquency')

plt.title('Frequency of Spam and Ham')

plt.show()
```

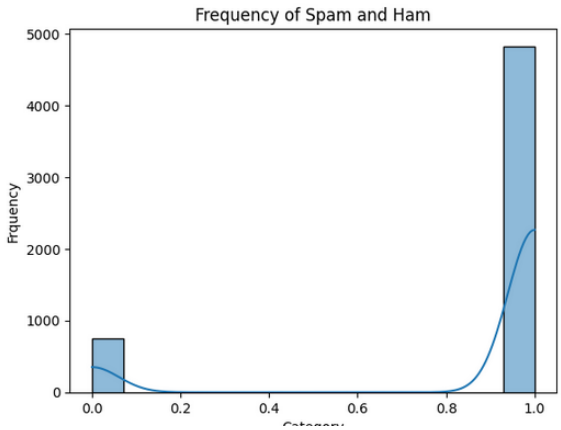

Figure 2.4: hist with KDE plot

code:

```
import seaborn as sns

sns.stripplot(mail_df['Category'])

plt.xlabel('Category')

plt.title('Frequency of Spam and Ham')

plt.show()
```

```
import numpy as np

num_ones = np.sum(df == 1)

print("Number of ones:", num_ones)
```

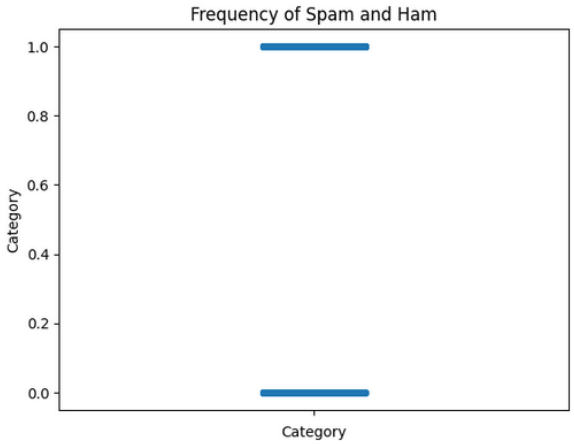

Figure 2.5: Strip Plot

```
num_zeros = np.count_nonzero(d_f == 0)

print("Number of zeros:", num_zeros)

import matplotlib.pyplot as plt

spam_count = num_zeros

non_spam_count = num_ones

labels = ['Spam', 'Non-Spam']

counts = [spam_count, non_spam_count]

colors = ['red', 'green']

plt.pie(counts, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90')

plt.title('Email Classification')

plt.axis('equal')

plt.show()
```

code:

```
data = {'Categories': ['Spam','Ham'],
        'Proportions': [num_zeros,num_ones]}


df = pd.DataFrame(data)
```

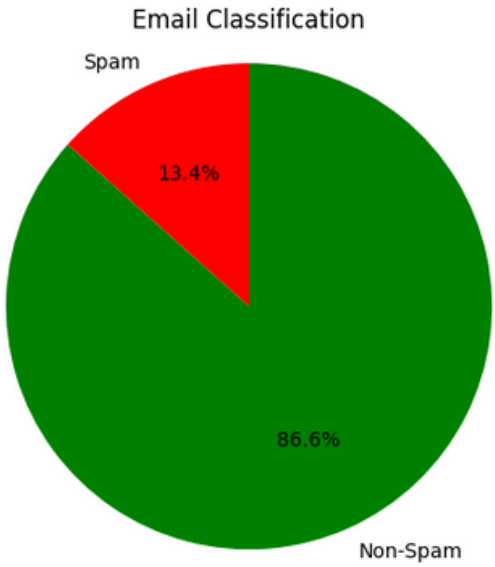

Figure 2.6: Pie Chart

```
fig = plt.figure(
    FigureClass=Waffle,
    rows=10,
    columns=10,
    values=df['Proportions'],
    labels=list(df['Categories'])
)
plt.title('Waffle Chart')
plt.show()
```

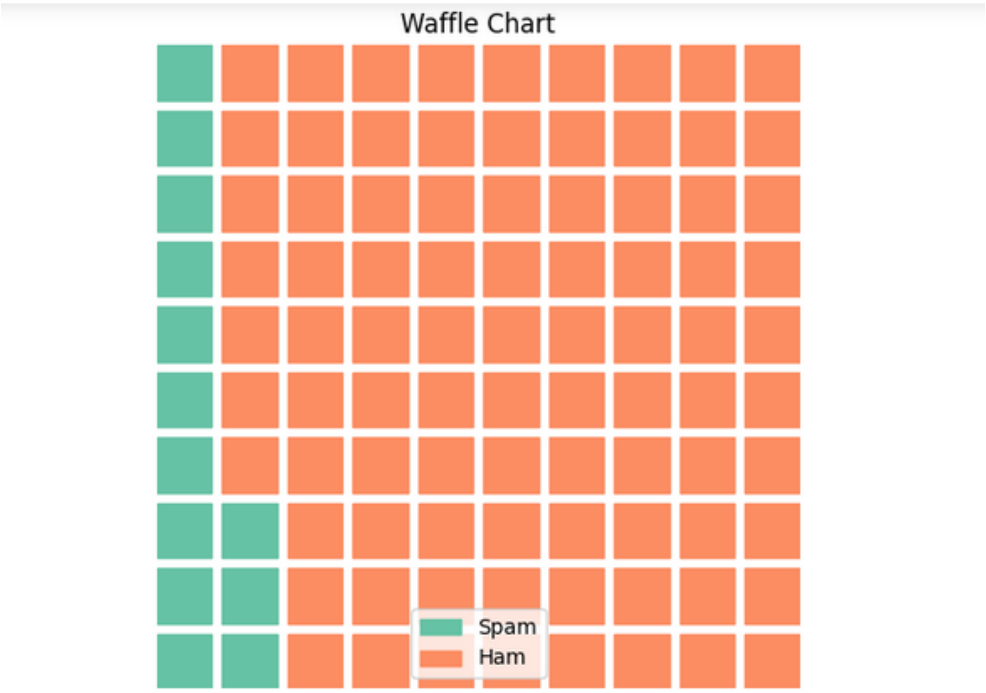

Figure 2.7: Enter Caption

Chapter 3

Code

3.1 Explaination of code with Outputs

```
import numpy as np
```

Importing Numpy: Its a powerful python library for numerical computing. it is also supports efficient array operations and mathmatical functions on linear alzebra and also random number generation,ndarrays to make mathmatical and linear operations on large datasets efficiently.

```
import pandas as pd
```

Importing Pandas: It is also powerful python library used for data manipulation and analysis.It provides easy to use data structures and data analysis tool making it is essential tool for working with structured data.

```
from sklearn.model_selection import train_test_split
```

Importing scikit-learn: It provides a wide range of tools for various machine learning tasks including classification ,regression,clustering,dimensionality reduction and model evaluation.It is also known as sklearn.

train test split: It is a utility in scikit-learn used to devide or split the dataset into training and testing dataset.It is used to assess the performance of the model on unseen data.

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

Importing TfidfVectorizer: Feature of exstracting technique commonly used in natural language processing and also in text mining tasks.It is a part of scikit learn library used to transform test document into numerical representation that machine learning algorithm can work with it.

```
from sklearn.linear_model import LogisticRegression
```

Importing LogisticRegression: It is a popular machine learning algorithm it is a simple for binary classification problems where target variable has two classes or outcomes.

```
from sklearn.metrics import accuracy_score
```

Importing Accuarcyscore: It is performance matrix provided by the scikit learn library.It commonly used to evaluate the performance of an algorithm for measuring accuracy of predictions.

```
mail_df=pd.read_csv("/home/rgukt/Downloads/mail_data.csv")
```

ReadingFile: We need to import or read the dataset on which we will do decisions and predictions and apply our machine leaning model by using pandas.Our dataset might be an xlfile or csv file.Based on the file extension we use pandas method to read the file.

```
mail_df
```

Displaying Dataset: The Dataset contains two columns and 5527 rows or entries.The colmuns are Category and Message.Here message column contains subject and body of the mail which are in the form of string datatype.Another column is category which contains whether two categorical values they are spam and ham corresponding to the message.

Spam:Spam email refers to specially to unsolicited or unwanted email messages.These are sent for commercial purposes,intending to advertise products ,services pr promote fraudulent schemes.

Ham:It refers to non-spam emails opposite of spam it is used to represent the regular,desired emails that are not classified as spam.

Out[9]:

	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 × 2 columns

Figure 3.1: Dataset

Output:

```
mail_df.fillna('',inplace=True)
mail_df
```

Replacing Null Values: We need to replace the null values with empty strings in our dataset for processing the data. Data consistency:In our dataset the spam and ham emails in string format but if we keep the null values as it is data inconsistency exist then data become non-homogenous.Hence we will get errors when processing or storing the data.To avoid null-related issues because null values can sometimes lead to errors or unexpected behaviour especially performing with strings.

To make good visualization while presenting ourdata using empty string can provide a clean and more consistent user experience.

Here we use head to display top five rows in the dataset.If we specify the number as parameter to head then that specified number of rows will be displayed.

```
for i in range(mail_df.shape[0]):
    if mail_df.loc[i, 'Category'] == 'spam':
        mail_df.loc[i, 'Category'] = 0
    else:
        mail_df.loc[i, 'Category'] = 1
```

Out[17]:

	Category	Message
0	1	Go until jurong point, crazy.. Available only ...
1	1	Ok lar... Joking wif u oni...
2	0	Free entry in 2 a wkly comp to win FA Cup fina...
3	1	U dun say so early hor... U c already then say...
4	1	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	0	This is the 2nd time we have tried 2 contact u...
5568	1	Will û b going to esplanade fr home?
5569	1	Pity, * was in mood for that. So...any other s...
5570	1	The guy did some bitching but I acted like i'd...
5571	1	Rofl. Its true to its name

5572 rows × 2 columns

Figure 3.2: Dataset after label encoding

```
mail_df
```

Output:

Label Encoding: It is method or technique used to convert labels or classes into numerical representation.In our problem this label encoding applied to assign numerical values to our spam and non-spam. spam=0 ham=1 for this we can use label encoder method from sklearn preprocessing.But we did label encoding manually by using pandas and numpy.

```
x=mail_df['Message']
y=mail_df['Category']
print(x)
print(y)
```

Output:

Assigning Independent and Dependent variables: Here we have assigned independent variable i.e Message attribute to the variable x and dependent variable i.e Category attribute to the variable y.

then print the variables x and y.

```
from sklearn.model_selection import train_test_split
```

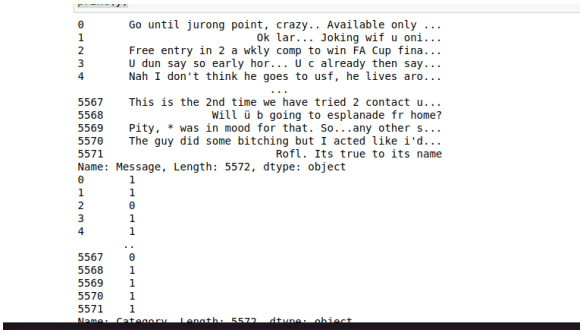

Figure 3.3: x and y columns

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=0)
```

Output:

splitting: Splitting the dataset into training and testing set by using train test split

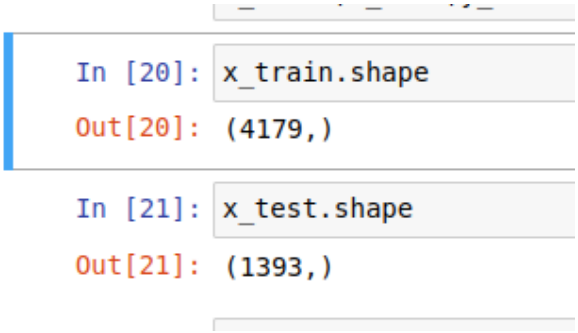

Figure 3.4: Splitting of datset

method from scikit learn model selection technique.It is used to evaluate the performance of model.It involves splitting the dataset into twos sets one for training and another one for evaluating the model performance.

It helps to assess how well the trained model generalizes on unseen data.It is a basic technique.In more advanced senarious we use techniques like cross validation,stratified sampling may be used to obtain robust performance.

Here the parameters of train test split the test size refers to the percentage of dataset is allocated for testing remaining allocated for testing this must be a float value represents percentage and if we specifies a number represents number of samples to be taken for testing from the dataset.

Random state is hyperparameter used to control any randomness invovled in machine

learning model to get consistent result.

```
from sklearn.feature_extraction.text import TfidfVectorizer
tv=TfidfVectorizer(min_df = 1,stop_words = 'english',lowercase = True)
x_train_f = tv.fit_transform(x_train)
x_test_f=tv.transform(x_test)
```

Output:

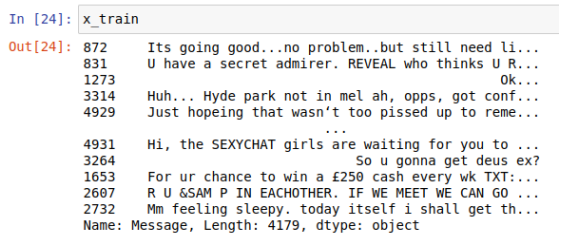

Figure 3.5: x_train before converting to numerical values

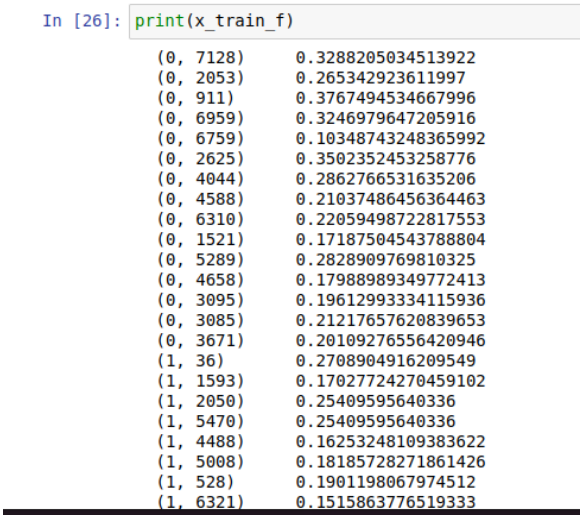

Figure 3.6: x_train after featuring

TfidfVectorizer: We need to convert the text data to feature vector that could be the input to the logistic regression for this we have used Tfidfvectorizer method.It takes three

arguments.First one is min df used to specify minimum number of words frequency to be included in vectorization process TF-IDF(term frequency inverse document frequency).

It determines the inclusion of terms based on their document frequency.It takes and integer or a float value between 0.0 and 1.0.Here we used 1 means the frequency of words should be one.

'stop words' is the next argument is used to specify a redefined set of words to be ignored during the vectorization process.When it sets to english common english words are excluded from result.

Next argument is lowercase it takes two boolean values i.e true or false .If it set to true all words will be converted to lowercase.It makes the process processing easy.you can avoid duplicate representations of the same word with different cases.Converting text to lowercase reduces the overall vocabulary size.

Lowercasing text helps to ensure consistent word frequency counts. Without converting to lowercase, the same word appearing in different cases would be treated as different terms, leading to potentially skewed word frequency counts and affecting the TF-IDF calculations.

```
y_train=y_train.astype('int')
y_test=y_test.astype('int')
```

y_train and y_test values keep it as integeres.

```
from sklearn.linear_model import LogisticRegression
logR=LogisticRegression()
logR.fit(x_train_f ,y_train)
```

Output:

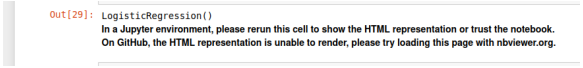

Figure 3.7: LogisticRegression

Training the model:

Now we need to train the model with training set by using Logistic Regression method.We have already imported the LogisticRegression model new assign to a variable for reusability.We have used fit it takes featurized x train and y train as parameters. The fit() method scans the input data and creates a vocabulary based on the unique words present in the Messages. It identifies all the unique words and assigns them an index in the vocabulary. The fit() method also calculates the IDF values for each term in the vocabulary. IDF represents the inverse document frequency of a term across the entire corpus and helps in assigning weights to the terms during vectorization. During the fitting process, the fit() method calculates the term frequency (TF) values for each document in the corpus. TF represents how often a term occurs in a particular document. These TF values are later used to calculate the TF-IDF weights. After fitting the vectorizer using fit(), you can use the transform() or fit transform() methods to transform new data into TF-IDF representations based on the learned vocabulary and IDF values.

the fit() method in TF-IDF vectorization is essential for creating the vocabulary, calculating IDF values, and ensuring consistency across train and test sets. It is a crucial step in preparing the vectorizer to convert raw text data into meaningful and weighted TF-IDF representations.

```
from sklearn.metrics import accuracy_score
y_pred_test=logR.predict(x_test_f)
y_pred_test
y_pred_train=logR.predict(x_train_f)
y_pred_train}
```

Output:

```
Out[39]: array([1, 0, 1, ..., 1, 1, 1])
```

Figure 3.8: y_pred_test

```
Out[40]: array([1, 0, 1, ..., 0, 1, 1])
```

Figure 3.9: y_pred_train

Prediction: After training our model, we need to predict the result for testing data for this we have used predict method from scikit-learn library.It takes one parameter that is on which we want to perform predict operation.It may be x train or x test. Here we have calculated prediction for both testing and traning.After predicting with x train f we get y_pred train and while prediction with x test f we get y_pred test.These are used to calculate the accuracy.

```
acc_on_train=accuracy_score(y_train,y_pred_train)
acc_on_train
acc_on_test=accuracy_score(y_test,y_pred_test)
acc_on_test
```

Output:

```
Out[43]: 0.9727207465900933
```

Figure 3.10: Accuracy of Training set

Accuracy score: The accuracy score is a common evaluation metric used to measure the performance of a classification model. It determines the proportion of correctly predicted labels compared to the total number of samples in the dataset. Here we have calculated the acurracy score on both training set and testing set.For testing we got 96 percent accuracy and for training we got 97 % accuracy.

Testing Model:

Figure 3.11: Accuracy of Testing set

```
input_mail = ["Hi, Summer is right around the corner in many parts of the world, and w
idf = fe.transform(input_mail)
output = model.predict(idf)
if(output==1):
    print('Not Spam/Ham Mail')
else:
    print('Spam Mail')
```

Output:

Not Spam/Ham Mail

Chapter 4

Conclusion and Future Work

Email spam classification is an important task in the field of email filtering and security. It involves the process of automatically determining whether an email is spam (unsolicited and unwanted) or ham (legitimate and desired). Classifying our mails can make user aware on fraud mails.

The project demonstrated the effectiveness of the chosen model in accurately classifying spam and non-spam emails. The results indicated a high accuracy rate, indicating the model's ability to distinguish between the two categories effectively.

By using some classification techniques on the dataset of containing mails and their categories, we got to know how to secure our mails from fraud.
Here Email spam helps individuals and organizations filter out unwanted emails, saving time and improving productivity. By accurately identifying spam emails, it reduces the risk of falling victim to scams, phishing attacks, and malware. Effective spam classification also enhances the overall email user experience by ensuring that the inbox contains relevant and important messages.
Regular updating and maintenance of the classification models are necessary due to the evolving nature of spam emails. As spammers continually adapt their techniques, the classification models should be trained on up-to-date data to ensure effectiveness.
Overall, email spam classification is an ongoing effort to combat the ever-present problem

of spam in email communication. With the use of advanced machine learning techniques and careful monitoring, it is possible to achieve high accuracy in classifying emails and providing users with a clean and secure email experience.

We can analyse, classify and detected a lot more using machine learning techniques and algorithms on different topic as we done on Email Spam Classification.