**DISCUSSION ON JOB TRAINING – 1**

**SPAM MAIL CLASSIFICATION**

**Abstract:**

This project focuses on developing a machine learning model to classify emails as spam or not-spam. The dataset used is from 'spam.csv', containing email messages labeled as spam or non-spam. Initial data preprocessing involved removing unnecessary columns, duplicates, and NaN values. Text cleaning techniques, including punctuation and stop words removal, were applied using NLTK.

Exploratory Data Analysis (EDA) visualized the distribution of spam and not-spam emails using bar and pie charts. Word clouds were generated to visualize the most frequent words in both spam and not-spam emails. Feature engineering employed TF-IDF vectorization to convert text data into numerical features, limiting the number of features to 2500.

Addressing class imbalance, Synthetic Minority Over-sampling Technique (SMOTE) was applied. The model was trained using a Decision Tree Classifier and evaluated for accuracy.

Serialization of the TF-IDF vectorizer and trained model was performed for future use. A prediction function was developed to classify user-input emails as spam or not-spam.

This project demonstrates a comprehensive approach to spam classification, encompassing data preprocessing, feature engineering, model training, evaluation, and prediction. The abstract summary provides an overview of the project's objectives, methodologies, and outcomes, highlighting its potential applications in email filtering and spam detection systems.

**Introduction:**

In recent years, internet has become an integral part of life. With increased use of internet, numbers of email users are increasing day by day. This increasing use of email has created problems caused by unsolicited bulk email messages commonly referred to as Spam. Email has now become one of the best ways for advertisements due to which spam emails are generated. Spam emails are the emails that the receiver does not wish to receive. a large number of identical messages are sent to several recipients of email. Spam usually arises as a result of giving out our email address on an unauthorized or unscrupulous website. There are many of the effects of Spam.

Fills our Inbox with number of ridiculous emails. Degrades our Internet speed to a great extent. Steals useful information like our details on you Contact list. Alters your search results on any computer program. Spam is a huge waste of everybody’s time and can quickly become very frustrating if you receive large amounts of it. Identifying these spammers and the spam content is a laborious task. Even though extensive number of studies have been done, yet so far the methods set forth still scarcely distinguish spam surveys, and none of them demonstrate the benefits of each removed element compose. In spite of increasing network communication and wasting lot of memory space, spam messages are also used for some attack. Spam emails, also known as non-self, are unsolicited commercial or malicious emails, sent to affect either a single individual or a corporation or a bunch of people. Besides advertising, these may contain links to phishing or malware hosting websites found out to steal confidential information. to solve this problem the different spam filtering techniques are used. The spam filtering techniques are accustomed protect our mailbox for spam mails.

**Aim:**

Nowadays, a big part of people rely on available email or messages sent by the stranger. The possibility that anybody can leave an email or a message provides a golden opportunity for spammers to write spam message about our different interests. Spam fills inbox with number of ridiculous emails. Degrades our internet speed to a great extent. Steal’s useful information like our details on our contact list. Identifying these spammers and also the spam content can be a hot topic of research and laborious tasks. Email spam is an operation to send messages in bulk by mail. Since the expense of the spam is borne mostly by the recipient, it is effectively postage due advertising. Spam email is a kind of commercial advertising which is economically viable because email could be a very cost-effective medium for sender. With this proposed model the specified message can be stated as spam or not using Bayes theorem and Naive Bayes Classifier and Also IP addresses of the sender are often detected.

**Objective:**

* Develop strategies or tools to effectively manage and filter spam emails to reduce clutter and ensure important messages are easily accessible.
* Investigate methods to mitigate the impact of spam on internet speed, whether through network optimization or more efficient email protocols.
* Implement measures to prevent the theft of sensitive information through spam emails, such as educating users about phishing techniques and enhancing email encryption
* Research and deploy solutions to prevent spam emails from influencing search engine results and potentially leading users to malicious websites.
* Develop efficient spam identification and filtering techniques to reduce the time users spend dealing with spam emails and improve productivity.
* Explore new approaches or technologies to more accurately identify spammers and their tactics, making it easier to combat spam at its source.
* Continuously refine and innovate spam filtering methods, incorporating machine learning, artificial intelligence, and other advanced technologies to stay ahead of evolving spamming techniques.

**Purpose:**

The purpose of this passage is to highlight the growing problem of spam emails and its detrimental effects on internet users. It discusses how spam inundates inboxes, slows down internet speeds, compromises privacy by stealing information, and alters search results. Additionally, it emphasizes the frustration and time wastage caused by dealing with spam. The passage also mentions the challenges in identifying and filtering spam effectively despite various studies and techniques. Overall, it aims to raise awareness about the seriousness of the spam issue and the need for effective spam filtering techniques to protect users' mailboxes.

**Scope:**

The passage primarily focuses on the issue of spam emails and its consequences on internet users. It covers various aspects of spam, including its definition, how it is generated, its effects such as filling inboxes, degrading internet speed, stealing information, and altering search results. The passage also briefly mentions the use of spam for malicious purposes such as phishing and distributing malware. Additionally, it touches upon the challenges associated with identifying and filtering spam effectively despite the existence of numerous studies and techniques. However, the passage does not delve deeply into specific technical details of spam filtering methods or the intricacies of spam-related cyber-attacks. It primarily

aims to inform readers about the problems caused by spam and the importance of implementing effective spam filtering techniques.

**Methodology:**

1. **Data Preprocessing:**
   * Import necessary libraries.
   * Read data from 'spam.csv' file into a DataFrame.
   * Drop unnecessary columns and duplicates, handle missing values.
   * Encode the target variable 'v1' using LabelEncoder.
   * Remove punctuations and stopwords from the text data.
2. **Exploratory Data Analysis (EDA):**
   * Visualize the distribution of spam and non-spam messages using bar and pie charts.
   * Generate word clouds for both spam and non-spam messages.
3. **Feature Engineering:** 
   * Convert text data into numerical format using TF-IDF vectorization.
4. **Handling Class Imbalance:**
   * Utilize SMOTE (Synthetic Minority Over-sampling Technique) to address class imbalance by oversampling the minority class.
5. **Model Building:**

* Split the data into training and testing sets.
* Initialize a Decision Tree classifier and fit it to the training data.
* Evaluate the model's accuracy on the test set.

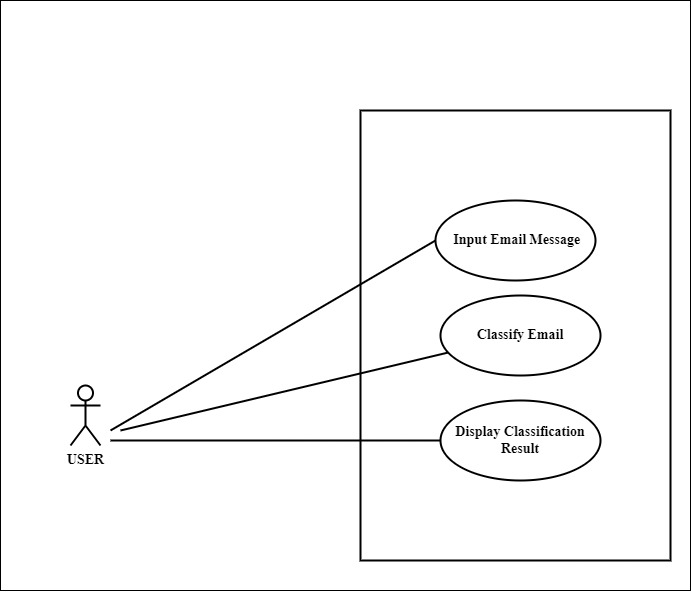
1. **Model Saving:**
   * Save the trained model and TF-IDF vectorizer using pickle for future use.
2. **Prediction Function:**
   * Define a function predict\_sentiment() to classify input messages as spam or not-spam.
   * Preprocess the input message, transform it using the TF-IDF vectorizer, and predict its class using the trained model.
3. **User Interaction:**
   * Prompt the user to input a mail message.
   * Utilize the ‘predict\_sentiment()’ function to classify the message as spam or not-spam and display the result.

**Advantages:**

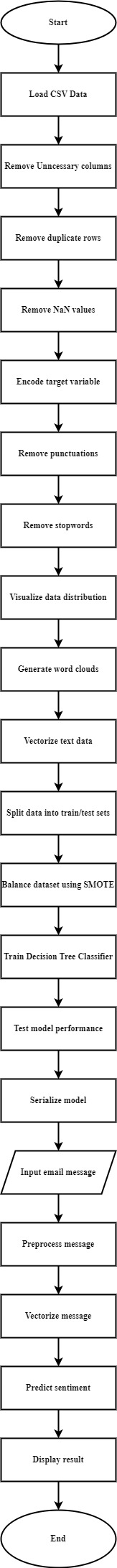
**ADVANTAGES:**

1. Customized the preprocessing steps, such as removing punctuations and stop-words, which can improve the quality of features extracted from the text data.
2. By visualizing the data distribution using bar charts, pie charts, and word clouds, you gain insights into the characteristics of spam and non-spam emails, which can inform feature engineering and model selection.
3. Using TF-IDF vectorization, you transform the textual data into numerical features, capturing the importance of words in each document relative to the entire corpus. This can improve the model's ability to distinguish between spam and non-spam emails.
4. Addressing class imbalance using SMOTE helps to alleviate the problem of having significantly more examples of one class than the other, which can lead to biased models.
5. Choosing Decision Tree Classifier for its simplicity and interpretability. Decision trees are easy to understand and can handle both numerical and categorical data.
6. By serializing the trained model using pickle, you can save it to disk and reload it later for making predictions without the need to retrain.
7. Providing a user-friendly interface for users to input email messages and receive predictions on whether they are spam or not-spam.

**UseCase Diagram:**

****

**Activity** **Diagram:**

****

**Conclusion:**

In conclusion, the proliferation of spam emails has become a significant issue in today's digital landscape. From inundating our inboxes with ridiculous messages to potentially compromising our personal information, the effects of spam are wide-ranging and detrimental. Not only does it degrade internet speed and waste valuable memory space, but it also poses serious security risks, as spam messages can serve as vectors for phishing attempts and malware distribution.

Despite efforts to combat spam through various filtering techniques, including content-based analysis and sender reputation systems, the problem persists. Identifying and mitigating spam remains a challenging task, requiring ongoing research and innovation in cybersecurity.

In the face of this persistent threat, it is crucial for individuals and organizations to remain vigilant and employ robust spam filtering measures to protect themselves from unwanted solicitations and potential security breaches. By staying informed about the latest developments in spam detection and prevention, we can work towards creating a safer and more secure online environment for all users.

**IMPLEMENTATION**

**IMPORT LIBRARY**

import string

import warnings

warnings.filterwarnings('ignore')

import pandas as pd

from sklearn.feature\_extraction.text import TfidfVectorizer

import matplotlib.pyplot as plt

from wordcloud import WordCloud

import pickle

from sklearn.preprocessing import LabelEncoder

import nltk

nltk.download('punkt')

nltk.download('stopwords'

from nltk.corpus import stopwords

from imblearn.over\_sampling import SMOTE

from sklearn.model\_selection import train\_test\_split

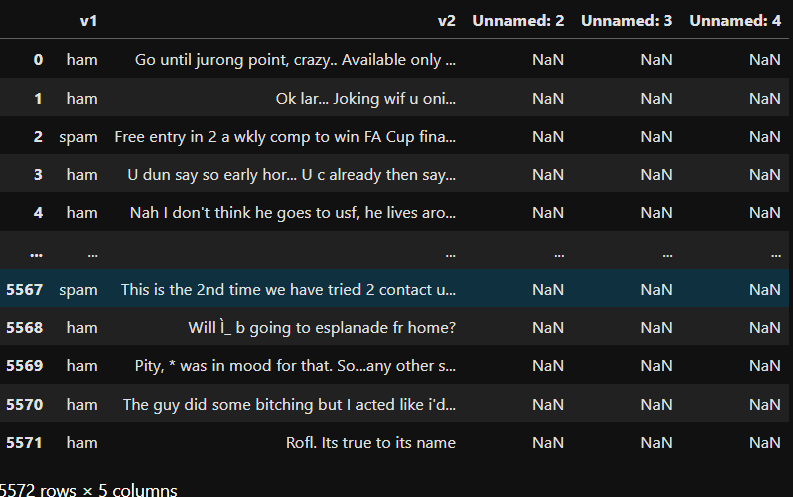
from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

**READ DATASET**

data = pd.read\_csv('spam.csv')

data

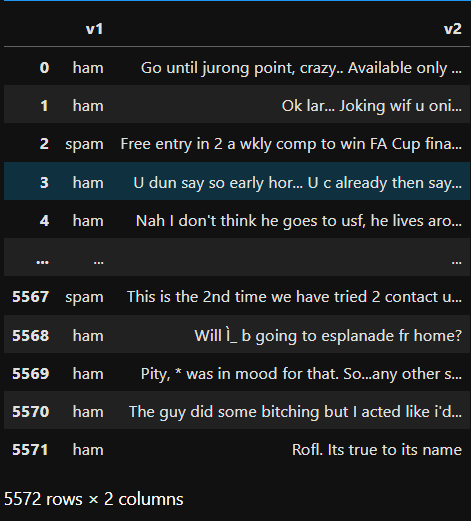
**OUTPUT**:

**DROPPING UNNECESSARY COLUMNS**

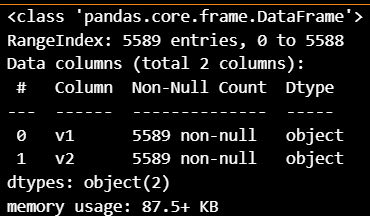
data.drop(columns=['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], inplace=True)

data

**OUTPUT:**

****

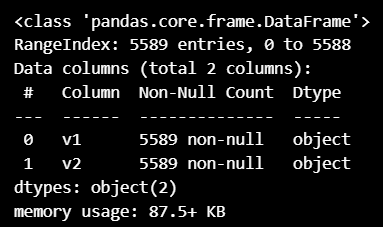
data.info()

**OUTPUT:**

data.dropna(inplace=True)

data.info()

**OUTPUT:**

****

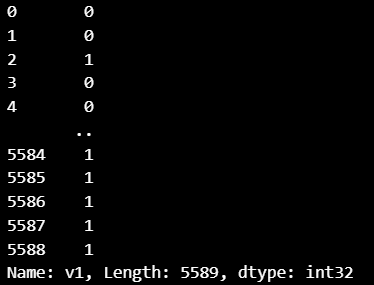
**LABEL ENCODING A COLUMN**

le = LabelEncoder()

data['v1'] = le.fit\_transform(data['v1'])

print(data['v1'])

**OUTPUT:**

****

**REMOVING STOPWORDS**

stp\_words = stopwords.words('english')

def clean\_message(message):

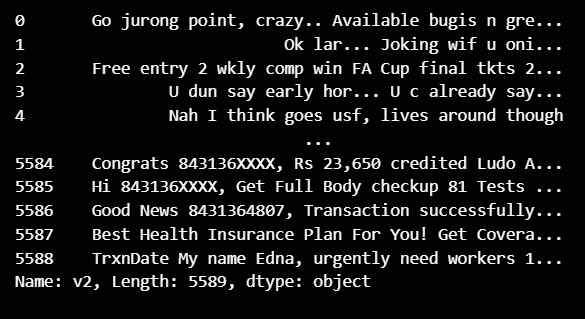
clean\_message = " ".join(word for word in message.split() if word not in stp\_words)

return clean\_message

data['v2'] = data['v2'].apply(clean\_message)

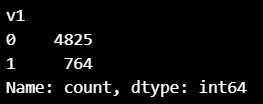
print(data.v2)

**OUTPUT:**

****

**VALUE COUNTS TYPICALLY REFERS TO A METHOD USED TO COUNT THE OCCURRENCES OF UNIQUE VALUES WITHIN A DATASET OR A SPECIFIC COLUMN**

data['v1'].value\_counts()

**OUTPUT**

**BAR GRAPH**

trans = data['v1'].value\_counts()

print(trans)

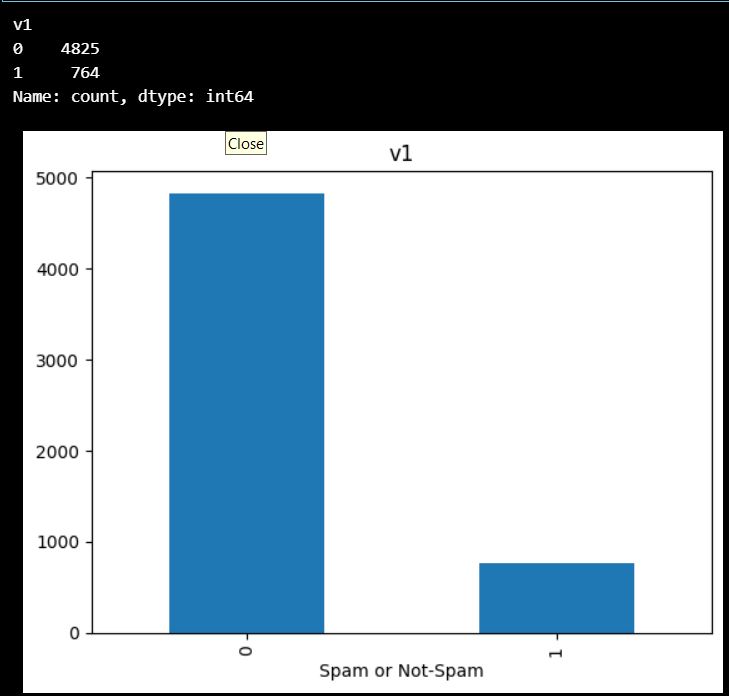
trans.plot.bar()

plt.title('v1')

plt.xlabel("Spam or Not-Spam")

plt.show()

**OUTPUT:**

****

**PIE CHART**

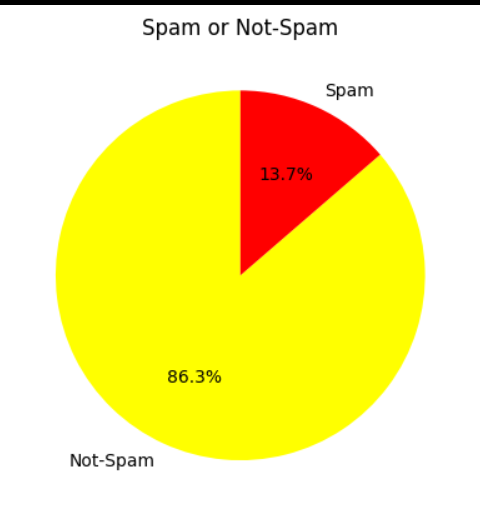
x = data['v1'].value\_counts()

y = 'Not-Spam', 'Spam'

plt.pie(x,labels = y,autopct = '%1.1f%%',startangle = 90,colors = ['yellow','red'])

plt.title('Spam or Not-Spam')

plt.show()

**OUTPUT:**

**WORD CLOUD**

consolidated = ' '.join(word for word in data['v2'][data['v1'] == 0].astype(str))

wordCloud = WordCloud(width = 1600,height = 800,max\_font\_size = 110)

plt.figure(figsize = (15,10))

plt.imshow(wordCloud.generate(consolidated),interpolation = 'bilinear')

plt.axis('off')

plt.show()

**OUTPUT:**

consolidated = ' '.join(word for word in data['v2'][data['v1'] == 1].astype(str))

wordCloud = WordCloud(width = 1600,height = 800,max\_font\_size = 110)

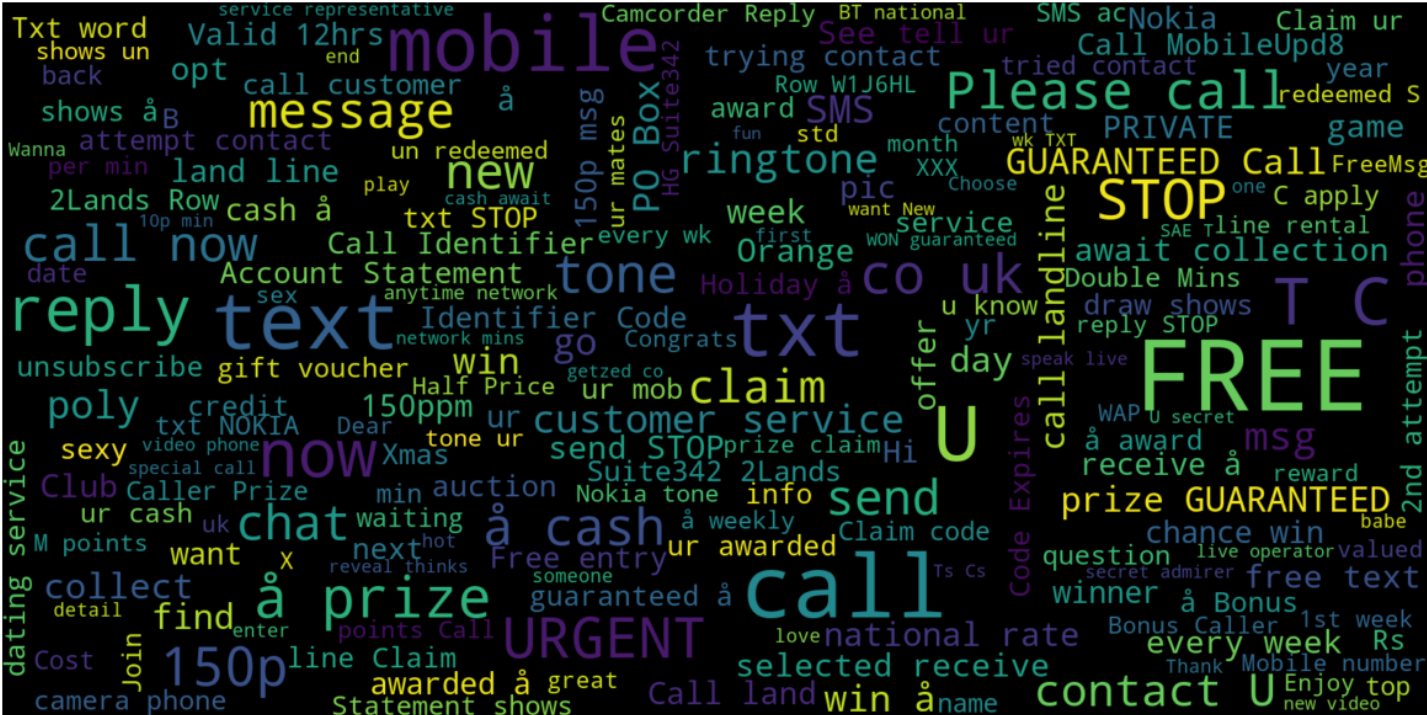
plt.figure(figsize = (15,10))

plt.imshow(wordCloud.generate(consolidated),interpolation = 'bilinear')

plt.axis('off')

plt.show()

**OUTPUT:**

****

cv = TfidfVectorizer(max\_features=2500)

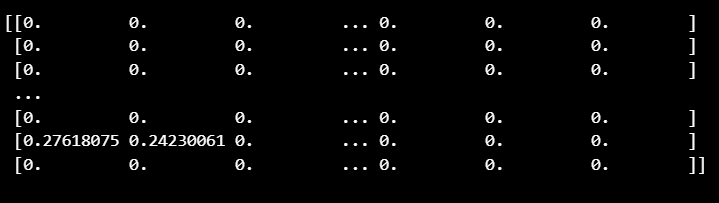
X = cv.fit\_transform(data['v2']).toarray()

print(X)

Y = data['v1']

with open('cv.pkl','wb') as file:

pickle.dump(cv, file)

**OUTPUT:**

#Initialize SMOTE with a sampling strategy (you can adjust it as needed)

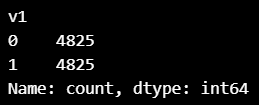
smote = SMOTE(sampling\_strategy = 'auto',random\_state = 42)

#Apply SMOTE to resample the dataset

X\_resampled,y\_resampled = smote.fit\_resample(X,Y)

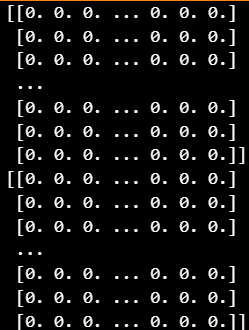
y\_resampled.value\_counts()

**OUTPUT**

****

print(x\_train)

print(x\_test)

**OUTPUT**:

print(x\_train.shape)

print(x\_test.shape)

**OUTPUT:**

****

**TRAINING AND PREDICITNG**

model = DecisionTreeClassifier()

#Model fitting

model.fit(x\_train,y\_train)

#testing the model

pred = model.predict(x\_test)

#model accuracy

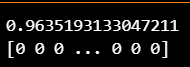
print(accuracy\_score(y\_test,pred))

print(pred)

import pickle

pickle.dump(model,open('model\_save.pkl','wb'))

model = pickle.load(open('model\_save.pkl','rb'))

**OUTPUT:**

def predict\_sentiment(message\_text):

#Preprocess the input review

cleaned\_message = clean\_message(message\_text)

#Transfrom the review using the TF-IDF vectorizer

transformed\_message = cv.transform([cleaned\_message]).toarray()

#Predict sentiment using the trained model

prediction = model.predict(transformed\_message)

if prediction[0] == 1:

return 'Spam'

else:

return 'Not-Spam'

#Now you can use the predict\_sentiment function to classify reviews

input\_message = input('Enter the Mail::')

result = predict\_sentiment(input\_message)

print(f'This Mail is {result}')

**OUTPUT:**