**BUILDING A SMARTER AI-POWERED SPAM IDENTIFIER**

**PHASE 4 SUBMISSION**

**FEATURE EXTRACTION AND MODEL PREDICTION:**

Data source:

Data link:( https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset)

**SUBMITTED BY**

NAME:S.PRIYADHARSHINI

REG NO:912421104030

**FEATURE EXTRACTION**

Feature extraction is a critical step in building a smarter AI-powered spam identifier. The goal is to convert raw text data into a format that machine learning models can understand and use for classification. Here are some common feature extraction techniques you can consider for my project:

1. **Bag of Words (BoW)**:
   * Create a vocabulary of unique words from your dataset.
   * Represent each document as a vector, where each element corresponds to the frequency of a word in the document.
   * Use techniques like TF-IDF (Term Frequency-Inverse Document Frequency) to give more weight to important words.
2. **N-grams**:
   * Instead of individual words, consider combinations of words (n-grams) as features.
   * For example, you might use bigrams (pairs of adjacent words) or trigrams (triplets of adjacent words) as features.
3. **Word Embeddings**:
   * Utilize pre-trained word embeddings such as Word2Vec, GloVe, or FastText.
   * These embeddings can capture semantic relationships between words and can be used to represent documents as dense vectors.
4. **Character-Level Features**:
   * Extract character-level n-grams or sequences to capture patterns at a finer level.
   * This can be useful for detecting spammy text that may contain irregular character patterns.
5. **Sender Information**:
   * Include features related to the sender's email address or domain.
   * Features might include the sender's reputation, known spammy domains, or IP address information.
6. **Message Metadata**:
   * Extract features from message metadata, such as the timestamp, subject, and message length.
   * Spam messages often exhibit certain patterns in these metadata.
7. **Text Statistics**:
   * Calculate statistics about the text, such as the number of uppercase letters, special characters, or links in the message.
   * Spam messages often use excessive capitalization and special characters.
8. **Language Features**:
   * Consider linguistic features like the language of the text or the readability score.
   * Some spam messages are written in non-standard or poorly constructed language.
9. **Contextual Features**:
   * Capture the context of the message, including previous interactions with the sender and the user's profile.
   * This can help identify spam messages that may appear legitimate based on context.
10. **Topic Modeling**:
    * Apply topic modeling techniques such as Latent Dirichlet Allocation (LDA) to identify the main topics in a message.
    * Spam messages may contain distinct topics related to various scams or promotions.
11. **Network Analysis**:
    * Analyze the network relationships between senders and recipients to identify unusual patterns.
    * For example, if a sender is sending messages to a large number of recipients, it may be indicative of spam.
12. **Semantic Analysis**:
    * Use semantic analysis techniques to understand the meaning of the text.
    * This can help in identifying spam messages that use obfuscation techniques to avoid simple keyword-based detection.
13. **User Behavior Features**:
    * Consider the behavior of the user, such as their history of marking messages as spam or the frequency of opening messages from specific senders.
14. **External Data**:
    * Incorporate external data sources, such as blacklists of known spammers or known malicious domains.
15. **Text Vectorization Techniques**:
    * Explore advanced text vectorization techniques, such as Doc2Vec or Word Movers' Distance, which capture semantic information in documents.

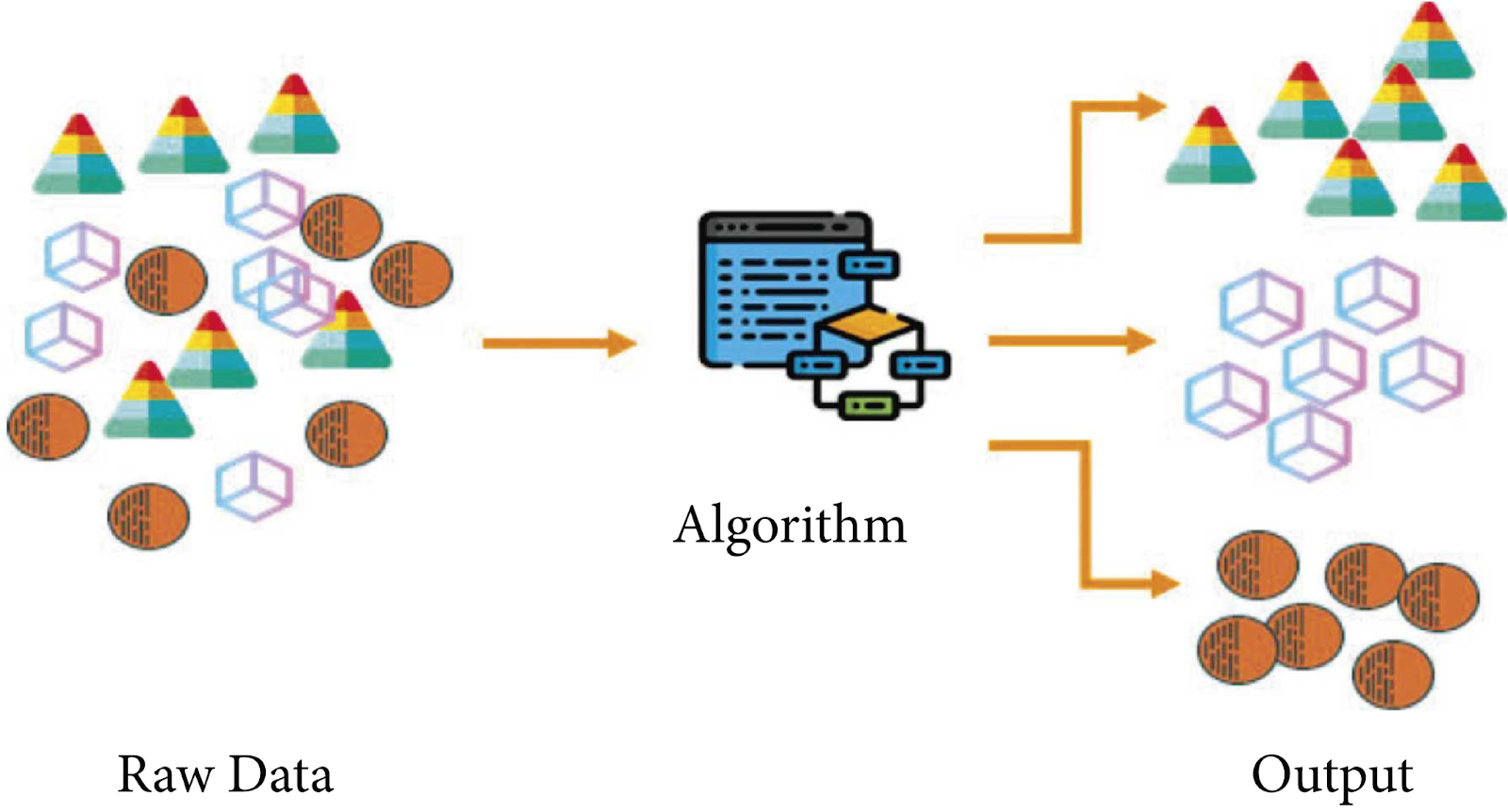
Model prediction

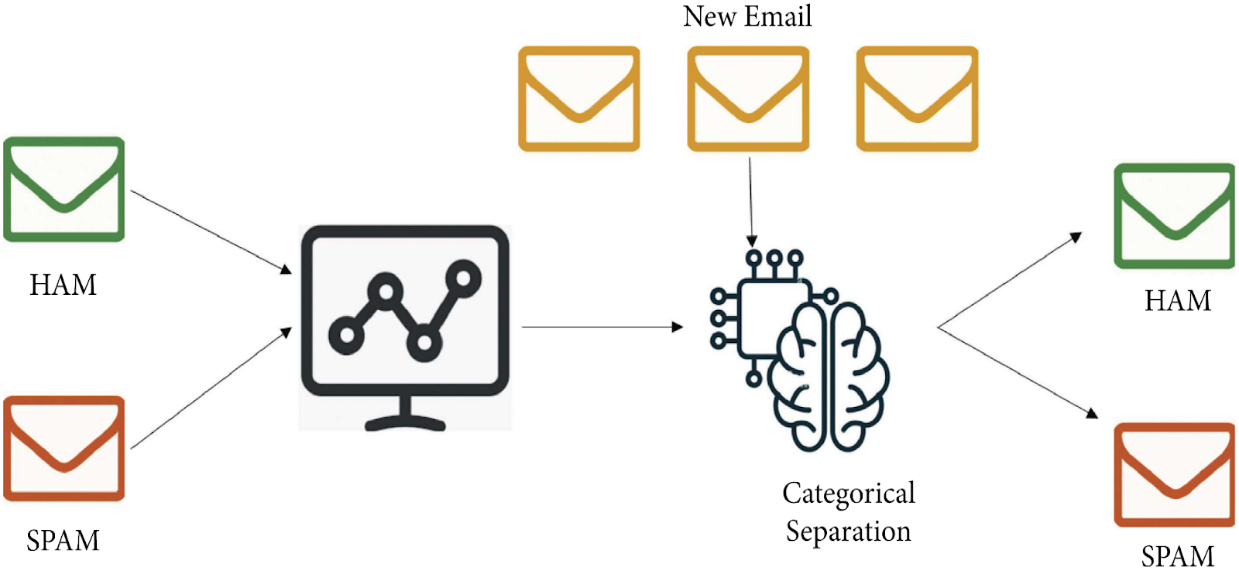
Building a smarter AI-powered spam identifier is an ambitious project that can help improve email and message filtering systems. To create an effective AI model for spam identification, you'll need to follow a systematic approach. Here's a prediction of the key components and steps you might consider:

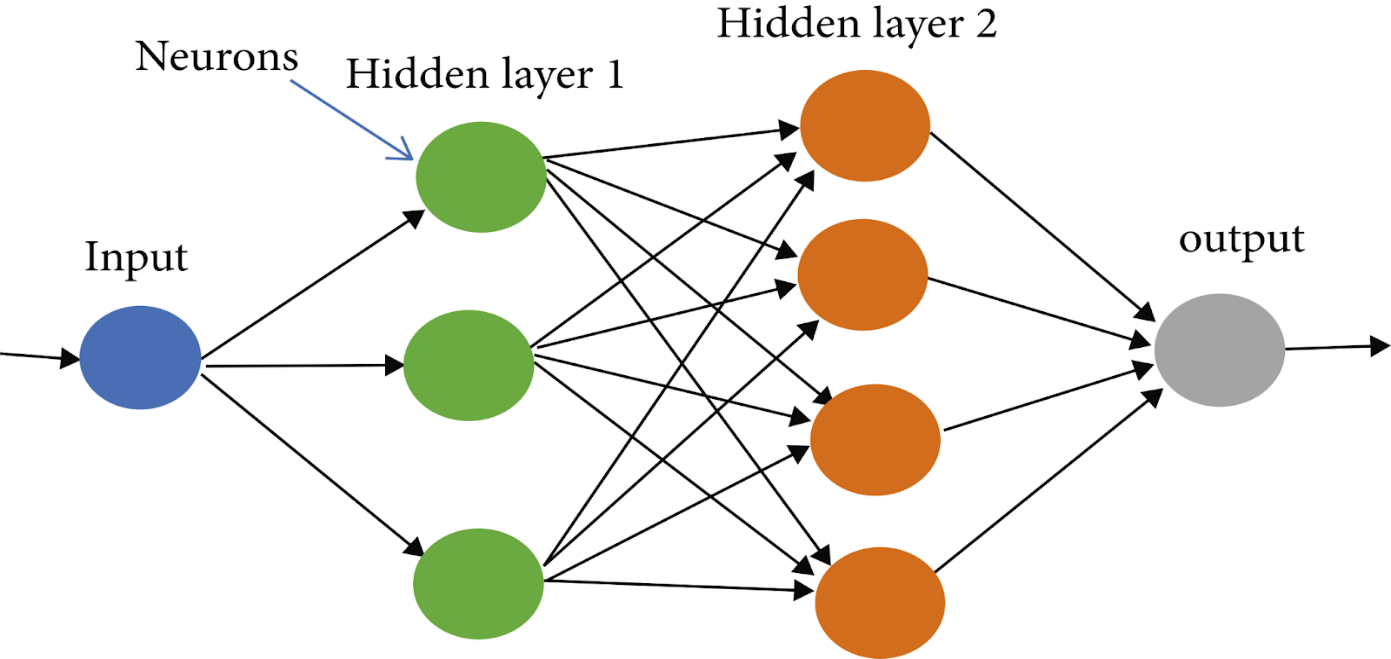
1. **Data Collection**:
   * Gather a large and diverse dataset of emails, messages, or communication data that includes both spam and non-spam examples. The dataset should cover various types of spam, such as phishing, promotional, and fraudulent messages.
2. **Data Preprocessing**:
   * Clean and preprocess the data, which may involve tasks like text normalization, removing special characters, and tokenization.
3. **Feature Engineering**:
   * Extract relevant features from the text data. Common features might include word frequency, sender information, IP addresses, and more.
4. **Model Selection**:
   * Choose an appropriate machine learning or deep learning model for spam detection. Popular options include:
     + Naive Bayes
     + Random Forest
     + Support Vector Machines
     + Recurrent Neural Networks (RNN)
     + Convolutional Neural Networks (CNN)
     + Transformers (e.g., BERT, GPT-3, or successors)
5. **Model Training**:
   * Split your dataset into training, validation, and testing sets. Train your selected model on the training data and fine-tune hyperparameters to optimize performance. You might also consider data augmentation techniques.
6. **Evaluation Metrics**:
   * Measure the performance of your model using standard evaluation metrics like accuracy, precision, recall, F1-score, and ROC AUC.
7. **Model Optimization**:
   * Implement techniques like hyperparameter tuning, model ensemble methods, or transfer learning to further improve the model's performance.
8. **Scalability**:
   * Ensure your model can handle a large volume of data in real-time. You may need to optimize its architecture for scalability and efficiency.
9. **Real-time Inference**:
   * Develop an API or service for real-time spam identification. The system should accept incoming messages, process them, and return spam predictions.
10. **User Interface**:
    * Create a user-friendly interface where users can manage and review their spam messages, marking false positives and false negatives to improve the system's accuracy.
11. **Feedback Loop**:
    * Implement a feedback mechanism to continuously improve the model by learning from user feedback and adapting to evolving spam tactics.
12. **Security and Privacy**:
    * Ensure the system complies with data privacy regulations and consider security measures to protect against abuse or hacking attempts.
13. **Deployment and Monitoring**:
    * Deploy the spam identifier to your desired platform and monitor its performance in production. Regularly update the model to stay ahead of new spam trends.
14. **Documentation and Reporting**:
    * Document your project thoroughly, including data sources, model architecture, training processes, and results. Create clear reports for stakeholders.
15. **Maintenance**:
    * Maintain the spam identifier over time, making necessary updates to keep it effective as spamming techniques evolve.

Building a smarter AI-powered spam identifier is an ongoing process that requires continuous improvement and adaptation. By following these steps, you can create a robust system to identify and filter out spam effectively.

PICTURE DEFINITION FOR MY PROJECT







PROGRAM :

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

raw\_spam=pd.read\_csv('/spam.csv',encoding='latin-1')

print(raw\_spam)

mail\_data=raw\_spam.where((pd.notnull(raw\_spam)),'')

mail\_data.head()

mail\_data.shape

mail\_data.loc[mail\_data['v1'] == 'spam','v1',] = 0

mail\_data.loc[mail\_data['v2']=='ham','v2',] = 1

x=mail\_data['v2']

y=mail\_data['v1']

print(x)

print(y)

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)

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)

y\_train=y\_train.astype('int')

y\_test=y\_test.astype('int')

print(x\_train)

print(x\_train\_features)

model = LogisticRegression()

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

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)

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)

input\_mail=["I HAVE A DATE ON SUNDAY WITH WILL!!,,,"]

input\_data\_features=feature\_extraction.transform(input\_mail)

prediction=model.predict(input\_data\_features)

print(prediction)

if prediction[0] == 1:

print('Ham mail')

else:

print('Spam mail')

OUTPUT:

import numpy as np

import pandas as pd

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

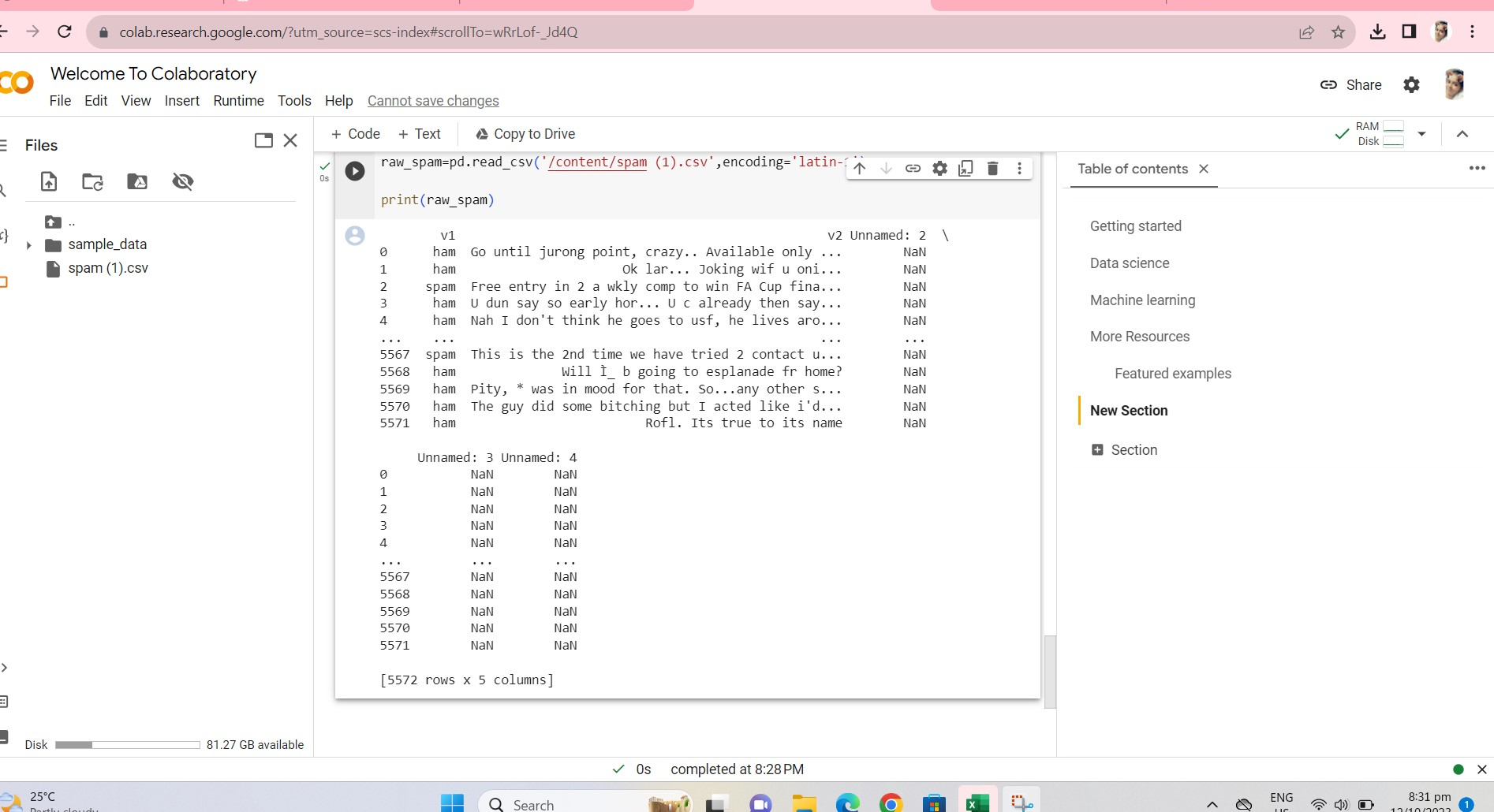
from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

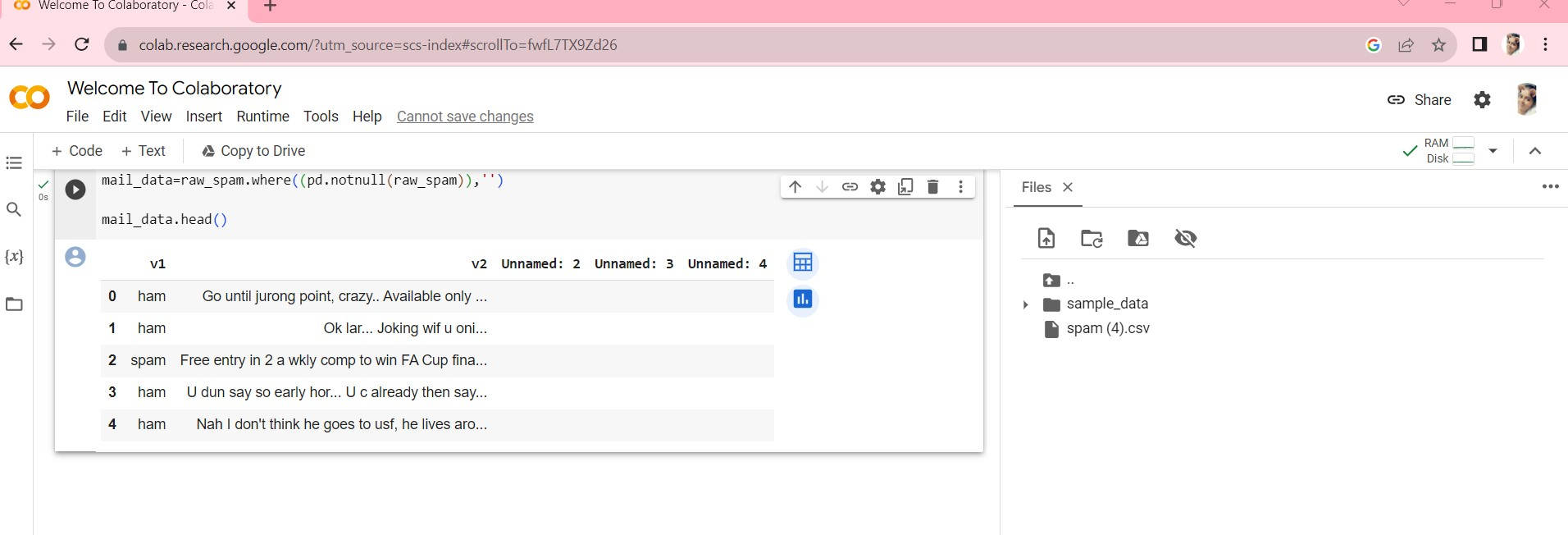
raw\_spam=pd.read\_csv(&#39;/spam.csv&#39;,encoding=&#39;latin-1&#39;)

print(raw\_spam)

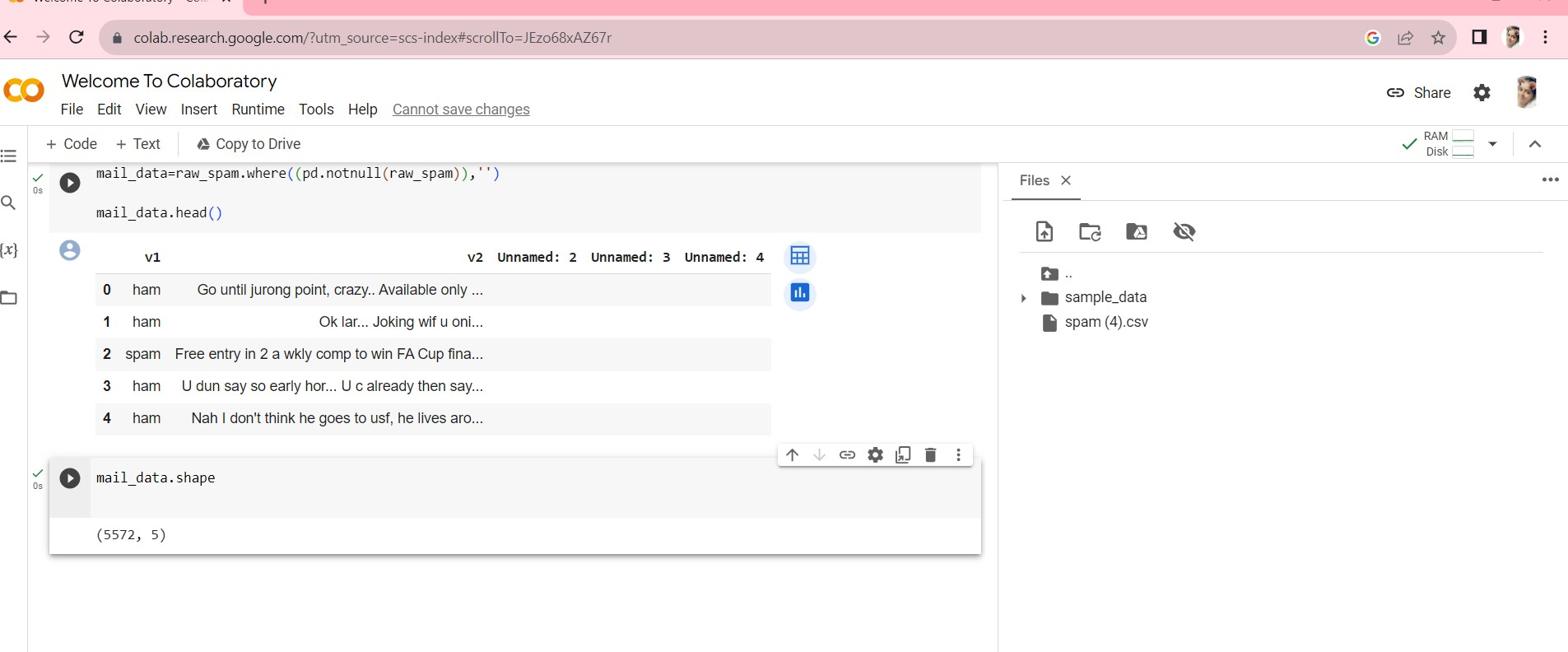


mail\_data=raw\_spam.where((pd.notnull(raw\_spam)),&#39;&#39;)

mail\_data.head()



mail\_data.shape



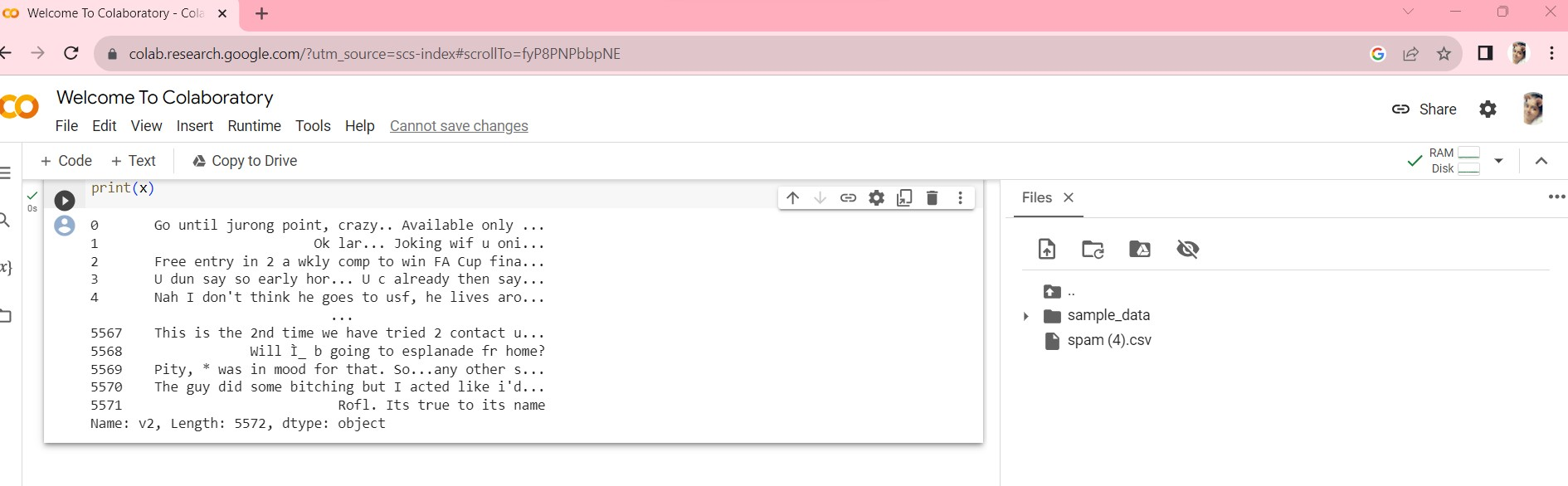
mail\_data.loc[mail\_data[&#39;v1&#39;] == &#39;spam&#39;,&#39;v1&#39;,] = 0

mail\_data.loc[mail\_data[&#39;v2&#39;]==&#39;ham&#39;,&#39;v2&#39;,] = 1

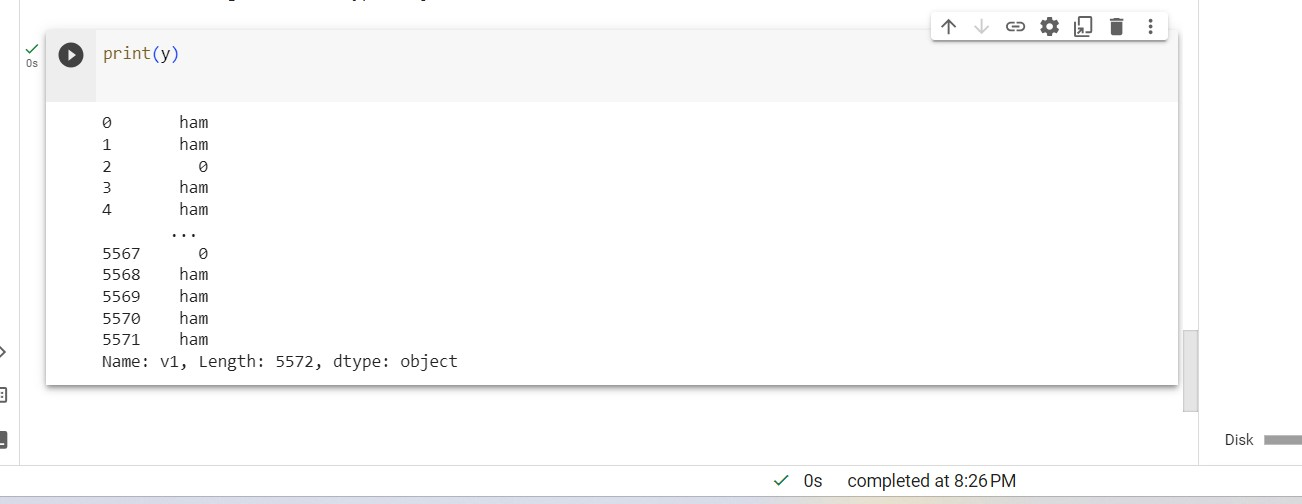
x=mail\_data[&#39;v2&#39;]

y=mail\_data[&#39;v1&#39;]

print(x)



print(y)



x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state

=3)

print(x.shape)



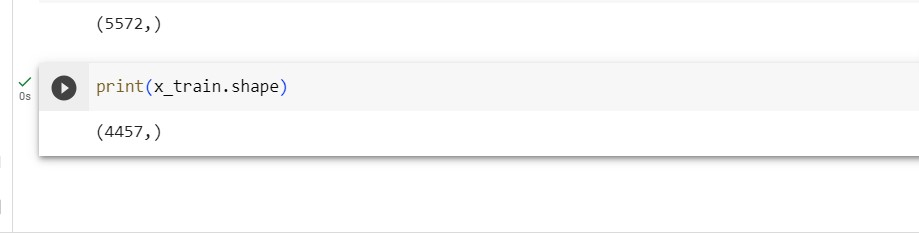
feature\_extraction=TfidfVectorizer(min\_df=

1,stop\_words=&#39;english&#39;,lowercase=True)

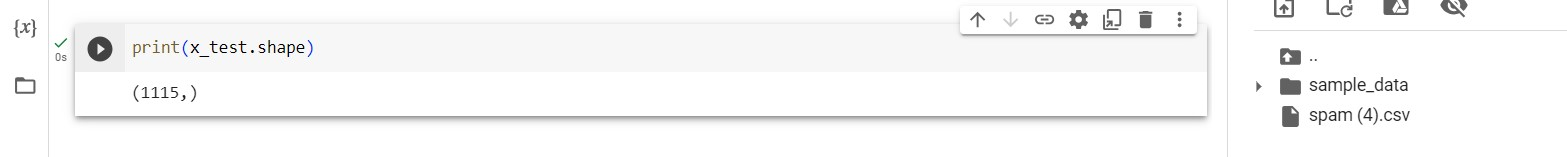
x\_train\_features = feature\_extraction.fit\_transform(x\_train)

x\_test\_features =feature\_extraction.transform(x\_test)

print(x\_train.shape)



print(x\_test.shape)



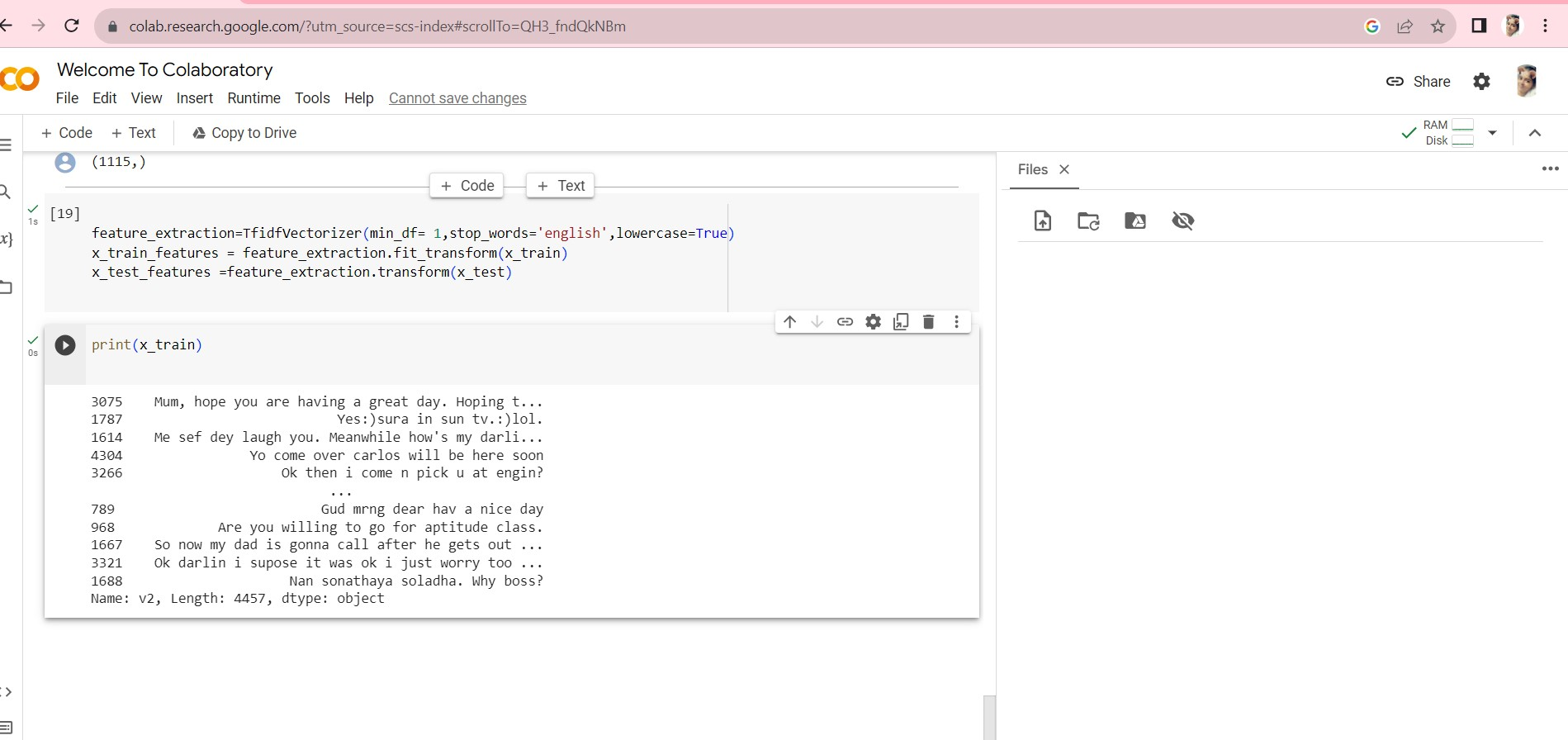
feature\_extraction=TfidfVectorizer(min\_df=

1,stop\_words=&#39;english&#39;,lowercase=True)

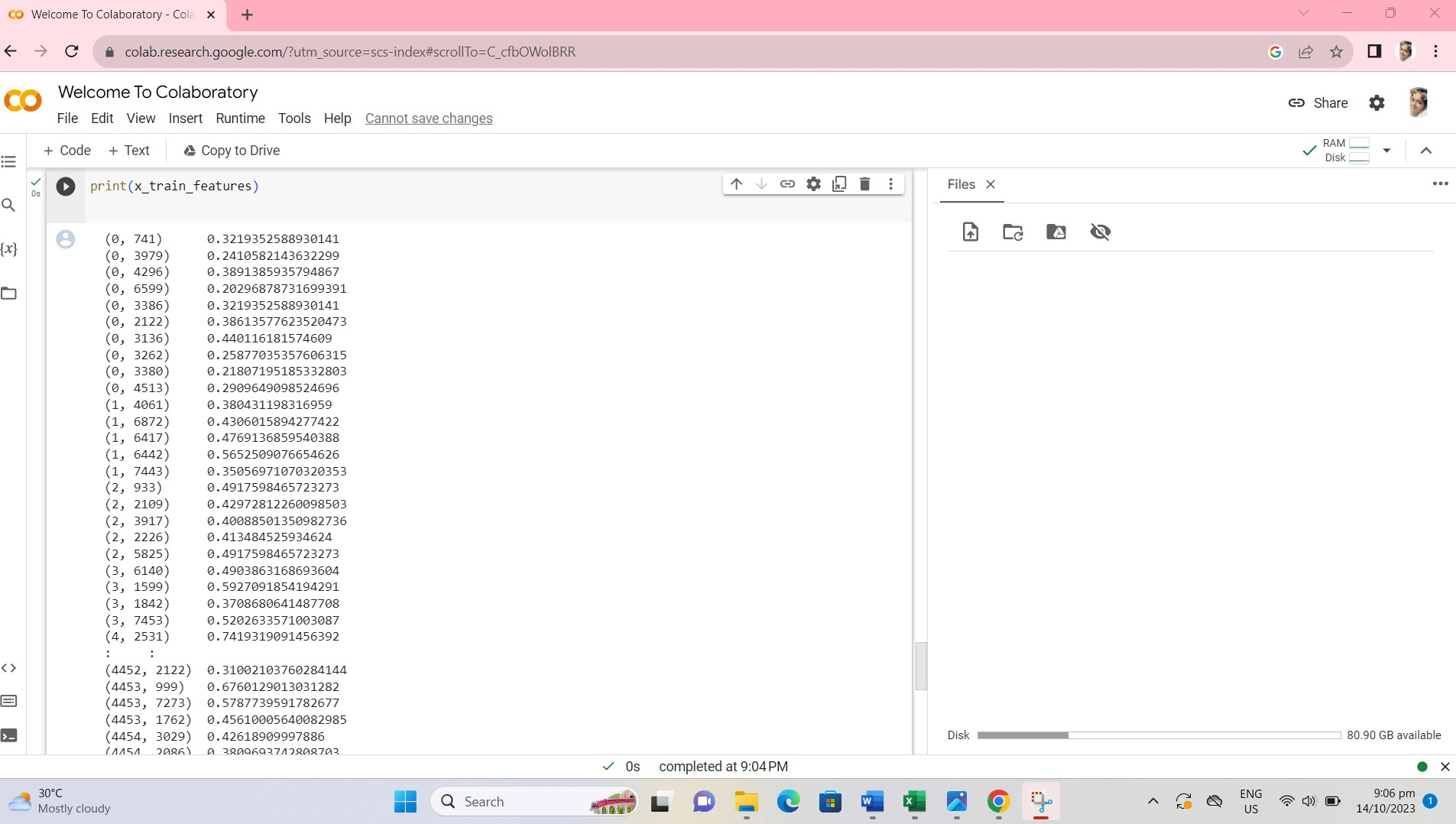
x\_train\_features = feature\_extraction.fit\_transform(x\_train)

x\_test\_features =feature\_extraction.transform(x\_test)

print(x\_train)



print(x\_train\_features)



Statistical diagram :

Program:

import matplotlib.pyplot as plt

# Example performance metrics

metrics = [&#39;Accuracy&#39;, &#39;Precision&#39;, &#39;Recall&#39;, &#39;F1-Score&#39;]

values = [0.95, 0.92, 0.89, 0.91]

plt.figure(figsize=(8, 6))

plt.bar(metrics, values, color=[&#39;blue&#39;, &#39;green&#39;, &#39;red&#39;, &#39;purple&#39;])

plt.ylim(0, 1) # Set the y-axis limits

# Add labels to the bars

for i, v in enumerate(values):

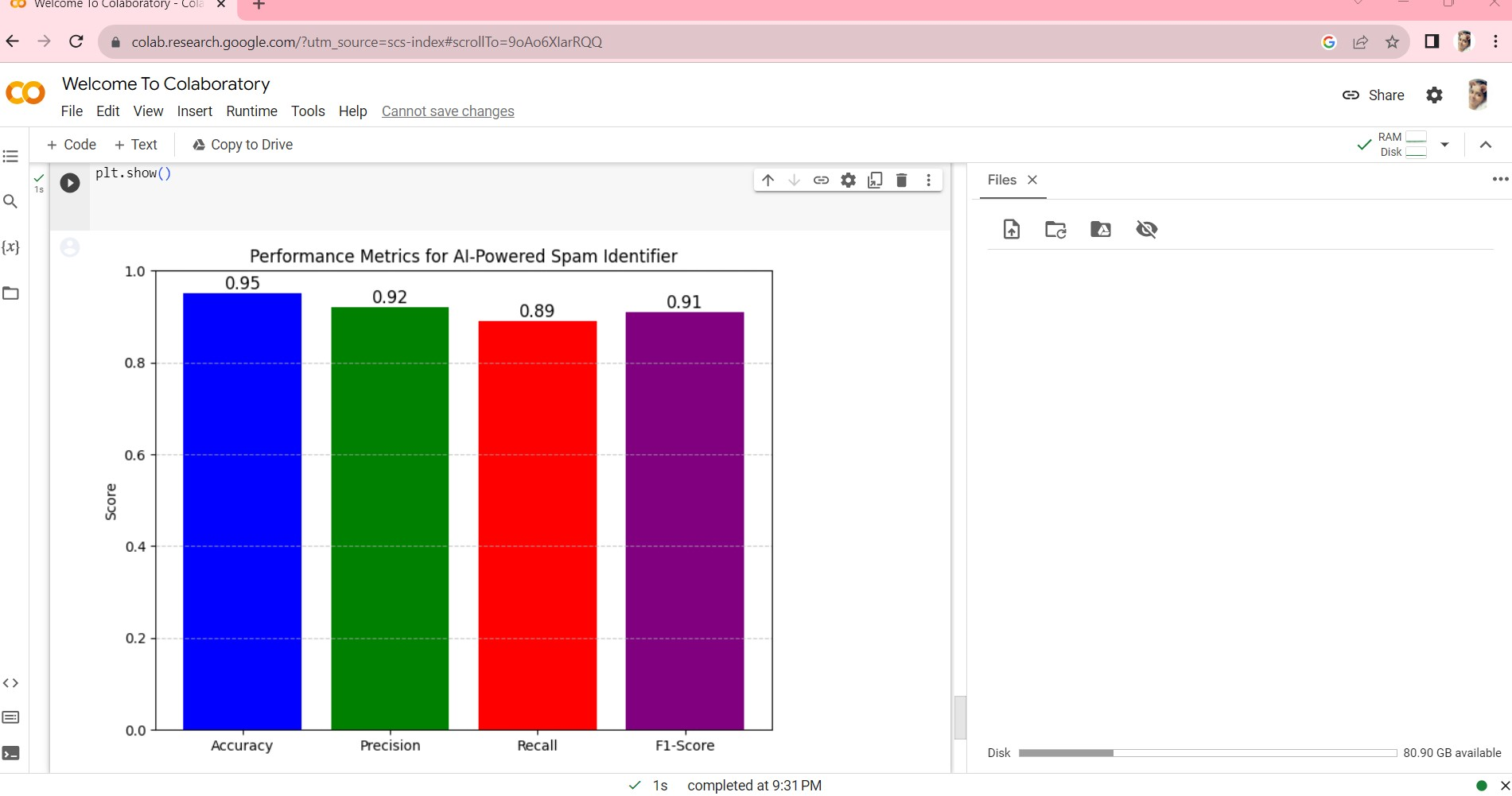
plt.text(i, v, f&#39;{v:.2f}&#39;, ha=&#39;center&#39;, va=&#39;bottom&#39;, fontsize=12)

plt.title(&#39;Performance Metrics for AI-Powered Spam Identifier&#39;)

plt.ylabel(&#39;Score&#39;)

plt.grid(axis=&#39;y&#39;, linestyle=&#39;--&#39;, alpha=0.7)

plt.show()



ADVANTAGES:

Building a smarter AI-powered spam identifier offers several significant advantages:

1. **Efficient Spam Filtering**: A smarter AI-powered system can filter out spam emails, messages, and content more accurately and efficiently, reducing the burden on users and organizations to manually deal with spam.
2. **Improved User Experience**:
   * Reduced spam means users are less likely to encounter annoying or harmful content.
   * A cleaner inbox or communication platform leads to a better user experience and increased user satisfaction.
3. **Enhanced Security**:
   * By accurately identifying and filtering spam, the system can protect users from phishing attacks and malicious links that are often distributed through spam messages.
   * This enhances the overall security of users and organizations.
4. **Time and Resource Savings**:
   * Users save time that would otherwise be spent sorting through spam.
   * Organizations save resources by reducing the time and effort needed to manage spam-related issues.
5. **Customization**:
   * AI-powered spam filters can be customized to match the specific needs and preferences of individual users or organizations.
   * They can adapt to evolving spam tactics and user behaviors.
6. **Adaptability**:
   * AI models can continuously learn from new data, making them adaptable to changing spam patterns and emerging threats.
7. **Scalability**:
   * AI-based systems can scale to handle large volumes of data and users, making them suitable for both individual users and large organizations.
8. **Reduced False Positives**:
   * Smarter AI models are less likely to mark legitimate messages as spam, reducing false positives and the risk of missing important communications.
9. **Multi-Modal Capabilities**:
   * AI models can analyze various forms of content, including text, images, and URLs, making them effective in identifying spam across different mediums.
10. **Real-Time Detection**:
    * AI models can operate in real-time, swiftly identifying and blocking spam as it arrives, thereby minimizing its impact.
11. **Compliance and Legal Benefits**:
    * Organizations can use AI-powered spam filters to ensure compliance with email and communication regulations.
    * They can also reduce the legal risks associated with unwanted or illegal content.
12. **Data Insights**:
    * By analyzing the characteristics of spam content, AI systems can provide insights into emerging spam tactics, which can be useful for cybersecurity and threat intelligence efforts.
13. **Reduced Phishing and Malware Risks**:
    * AI-powered spam identifiers can identify and block phishing attempts and malware-laden spam, reducing the risk of security breaches.
14. **Reduced Overhead**:
    * With automated spam filtering, organizations can reduce the need for manual moderation and user reports, lowering operational overhead.
15. **Adaptive Countermeasures**:
    * AI-powered systems can adapt to new evasion techniques employed by spammers, making it harder for spammers to bypass filters.
16. **Protecting Reputation**:
    * For businesses, using a smarter AI-powered spam identifier can help maintain a positive brand reputation by ensuring that customers receive legitimate and safe communications.

In summary, building a smarter AI-powered spam identifier offers numerous advantages, including improved efficiency, enhanced security, a better user experience, and the ability to adapt to evolving threats. These benefits make such projects highly valuable for both individual users and organizations.