**Problem Definition and Innovation**

**Problem Definition:**

The problem is to develop a fake news detection model using a Kaggle dataset. The goal is to distinguish between genuine and fake news articles based on their titles and text. This project involves using natural language processing (NLP) techniques to preprocess the text data, building a machine learning model for classification, and evaluating the model's performance.

**Steps to implement the Design Thinking:**

**Step 1:**

* Each article is tagged as Fake or True. we take it from such datasets on platforms like Kaggle. Here we have attached the dataset that given on the problem statement of fake news and detection using NLP.

**Step 2:**

* Preprocess the text data:
* Clean the text by removing special characters, and punctuation.Tokenize the text into words or sub word tokens. Remove stop words.We can add the label in the dataset like fake as 0 and true as 1.

**Step 3:**

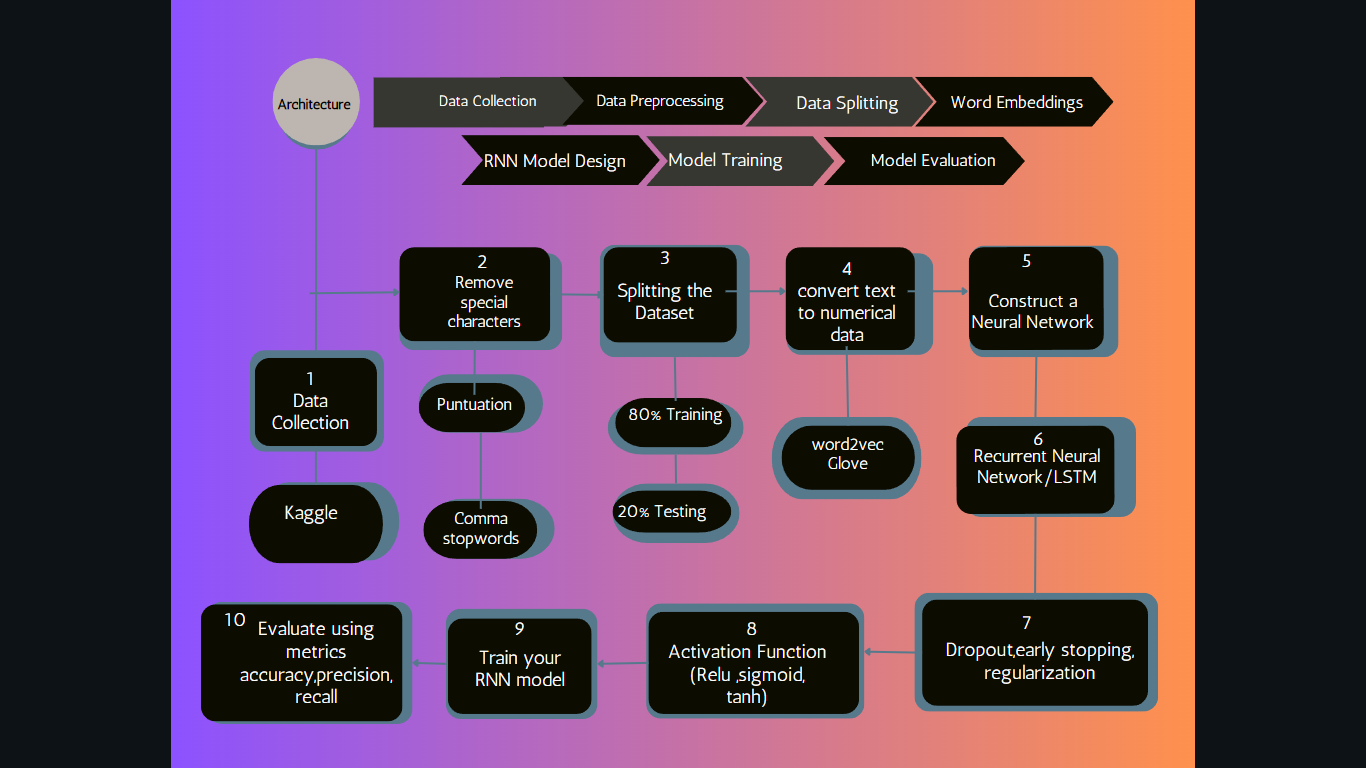
* Split your dataset into training and test sets. A common split might be 80% for training, 20% for testing.

**Step 4:**

* Convert text data into numerical vectors using word embeddings like Word2Vec and Glove. These embeddings capture semantic meaning.

**Step 5:**

* Choose an RNN architecture such as simple RNN or LSTM. These RNN variants are good choices for sequential data like text. Define the input layer to accept the word embeddings.



Include techniques like dropout, early stopping, regularization to prevent overfitting. Add a fully connected layer(s) with appropriate activation functions like sigmoid, Tanh, Relu etc.

**Step 6:**

* Train your RNN model on the training data. Optimize your model using gradient descent-based optimizers like Adam or RMSprop.

**Step 7:**

* Evaluate your RNN model's performance on the test set using metrics like accuracy, precision, recall, F1-score, and confusion matrix.

**ARCHITECTURE:**

**DATA COLLECTION**

**DATA SPLITTING**

**DATA PREPROCESSING**

**WORD EMBEDDINGS**

**RNN MODEL DESIGN**

**MODEL TRAINING**

**MODEL**

**EVALUATION**

Let see the data analysis and data cleaning operation done in the dataset,

**#import the csv file**

import pandas as pd

fake=pd.read\_csv("/content/Fake.csv")

true=pd.read\_csv("/content/True.csv")

**#size of the dataset**

fake.shape

(23481, 4)

true.shape

(21417, 4)

**#No of null values in the dataset**

fake.isnull().sum()

title 0 text 0 subject 0 date 0 dtype: int64

true.isnull().sum()

title 0 text 0 subject 0 date 0 dtype: int64

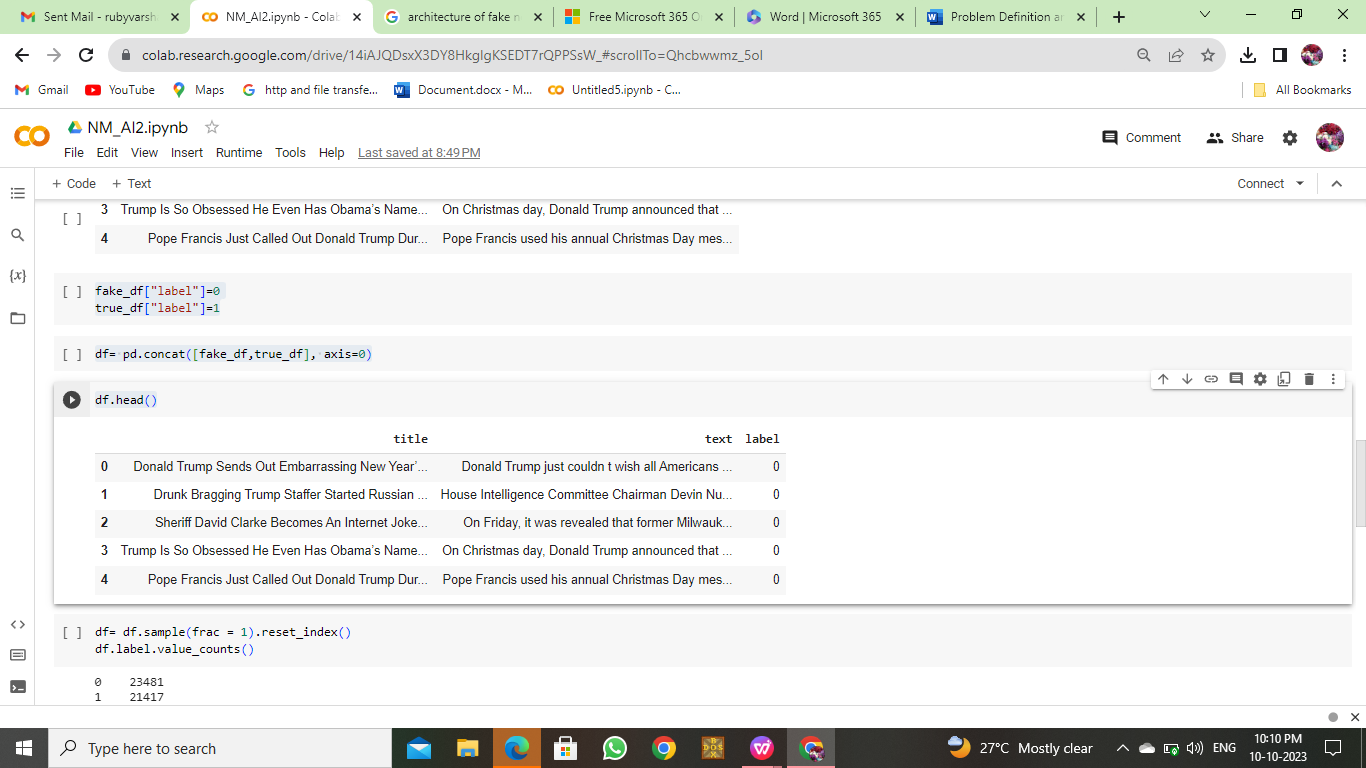
fake\_df=fake.drop(['date', 'subject'], axis=1)

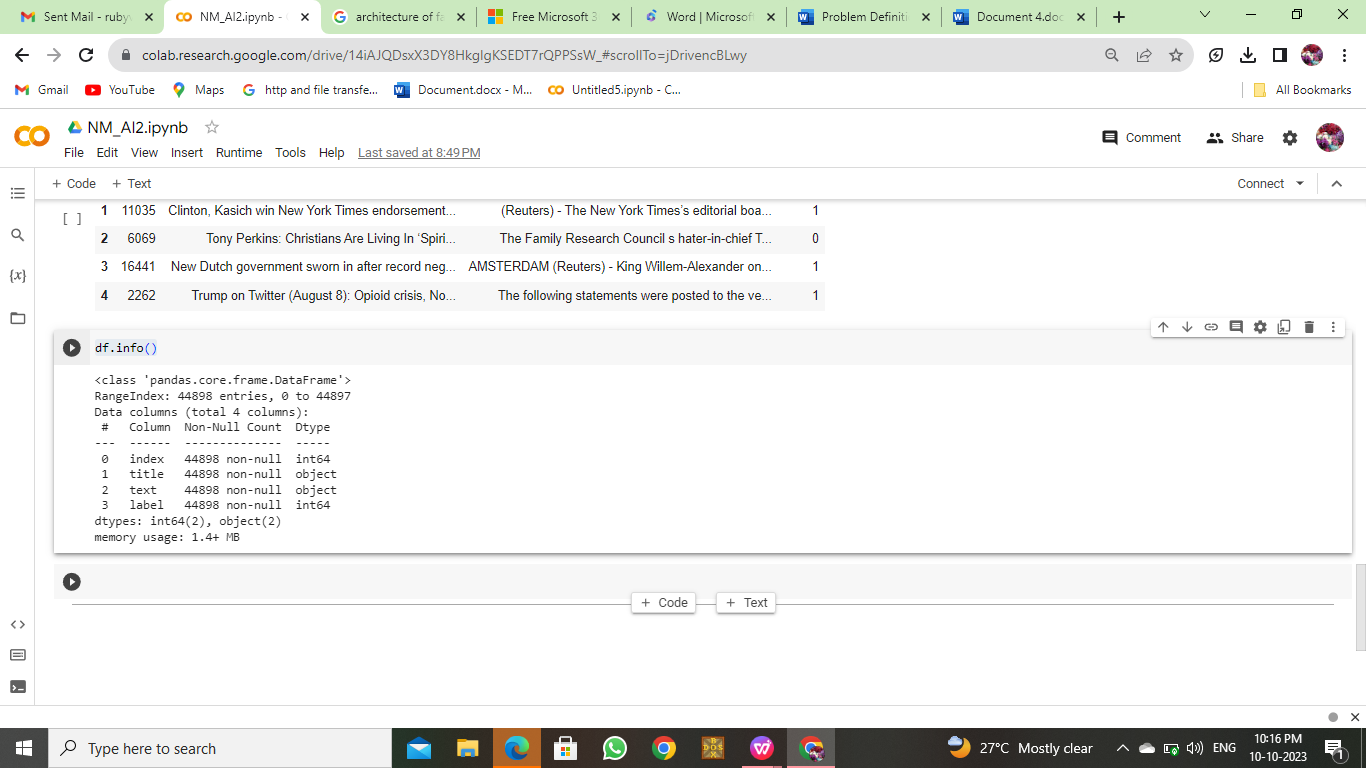
true\_df=true.drop(['date', 'subject'], axis=1)

**#Labelling the dataset**

fake\_df["label"]=0

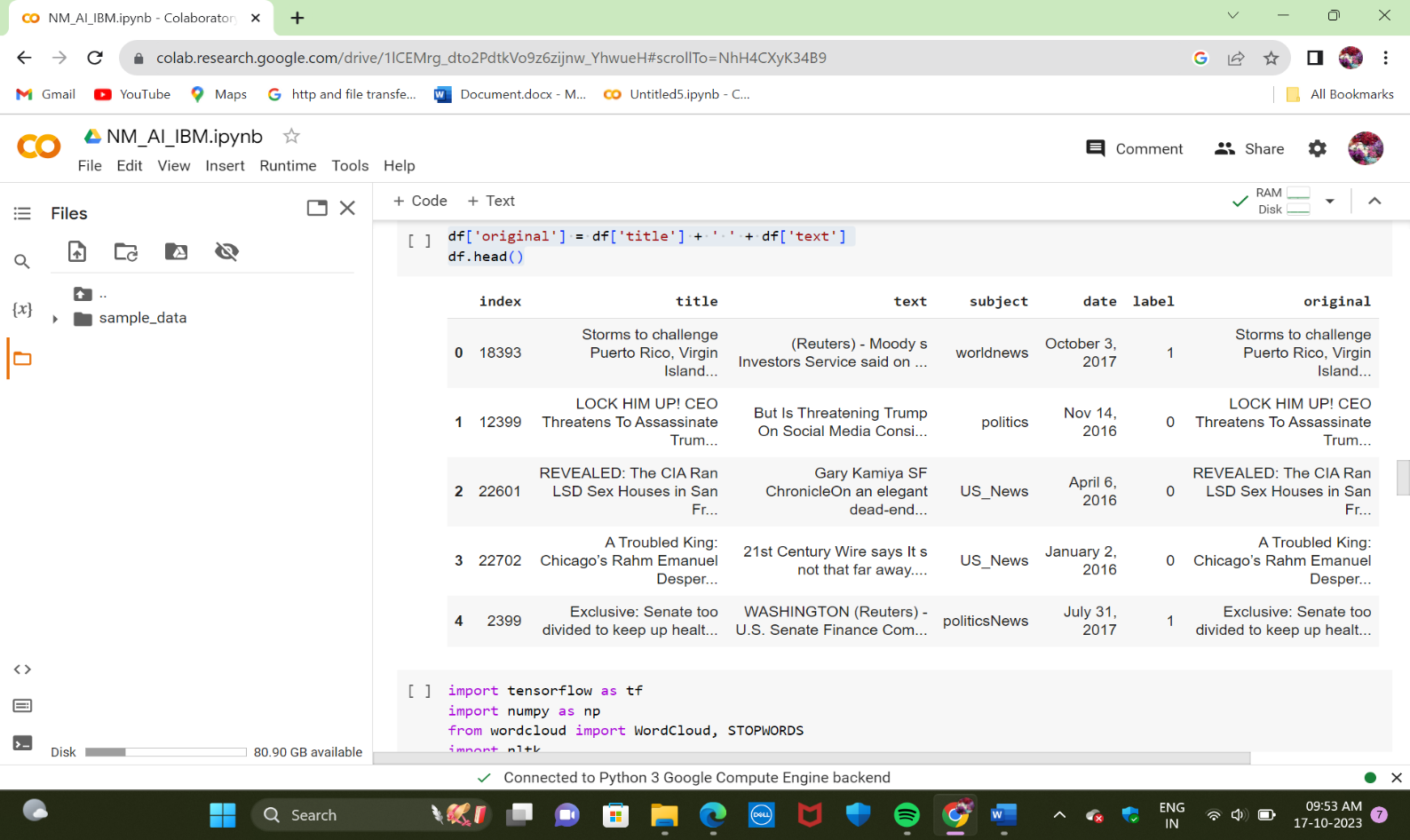
true\_df["label"]=1

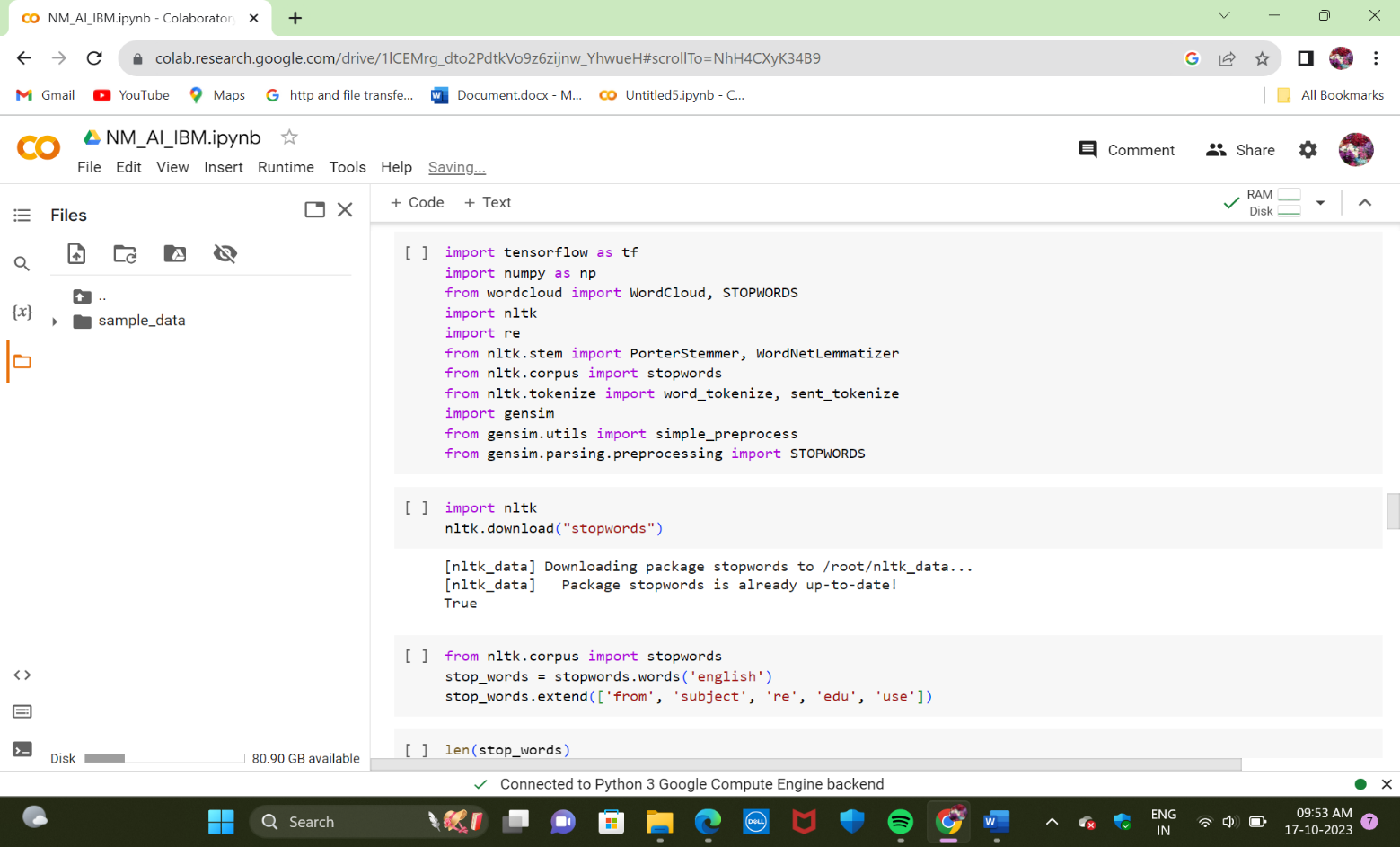




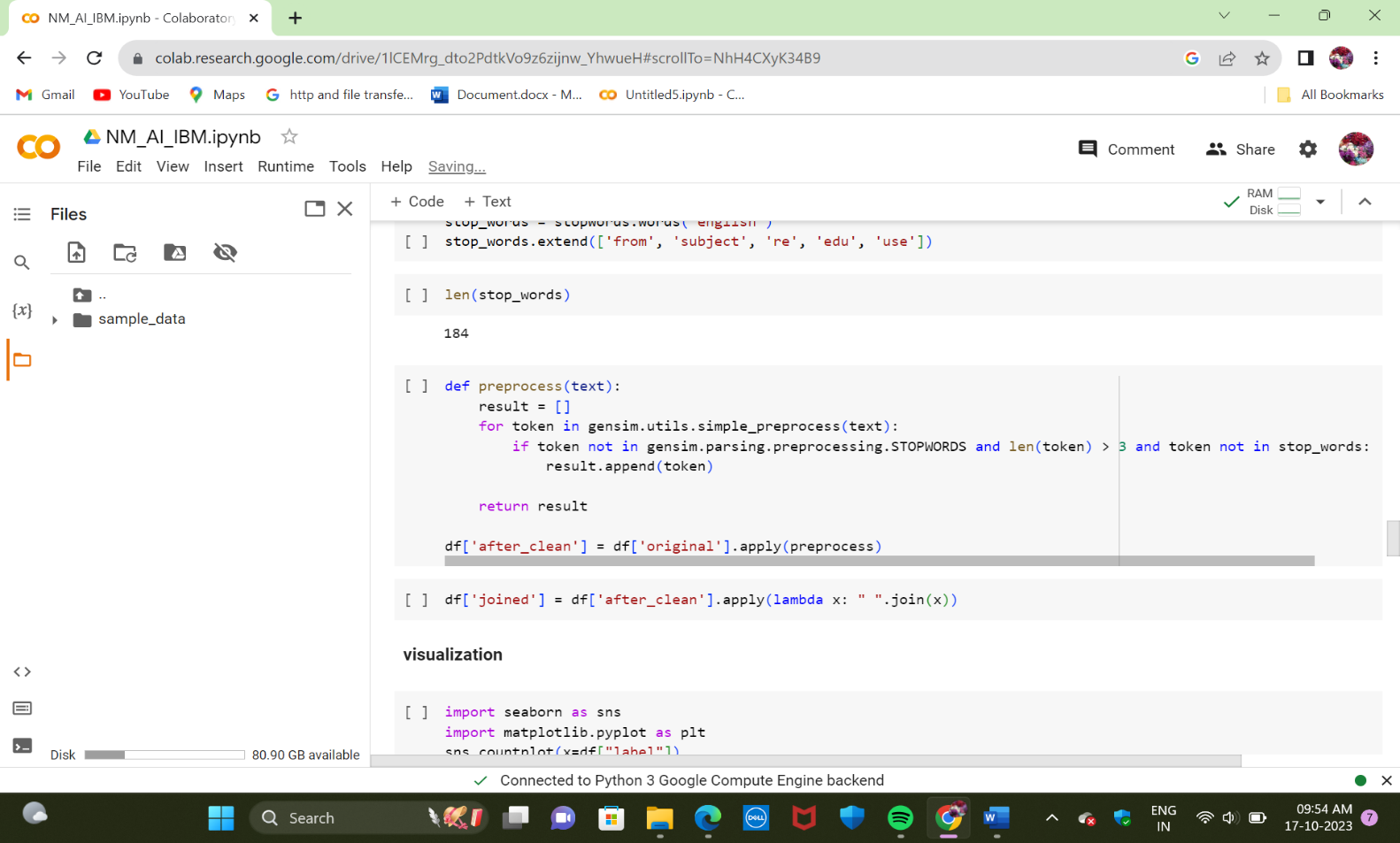
Let see the data preprocessing techniques in the dataset,

**#create a new column in a DataFrame 'df' named 'original' by concatenating the 'title' and 'text' columns with a space in between, and then display the first few rows of the DataFrame using 'df.head()'.**

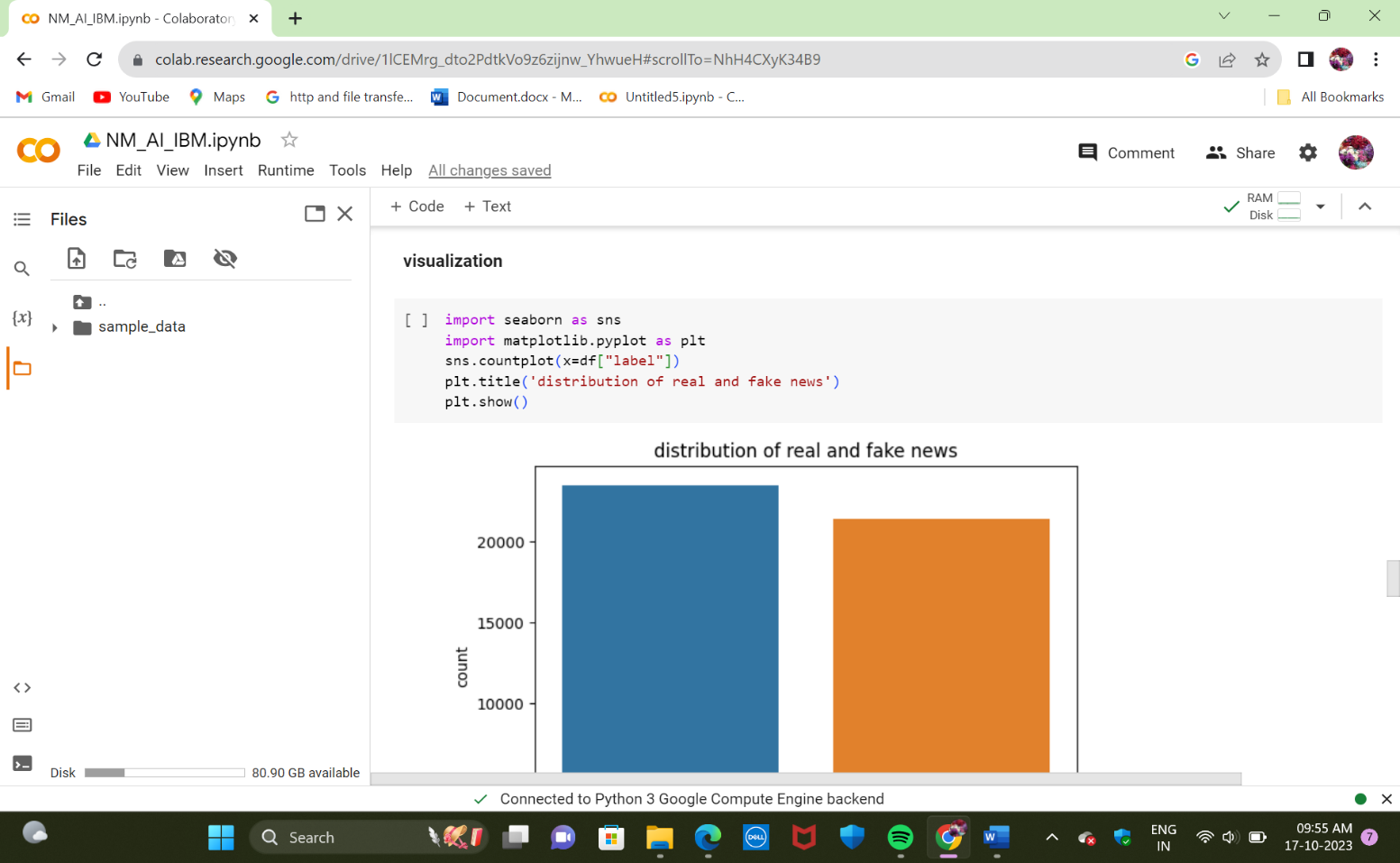
****

**#This imports various natural language processing and machine learning libraries, such as TensorFlow, NLTK, Gensim, and WordCloud, to perform text processing and analysis, including tasks like text tokenization, word cloud generation, and potentially natural language understanding. **

**#The code likely involves text preprocessing, such as stemming, lemmatization, and stop word removal, and may be used for text data analysis or machine learning tasks involving text data.**

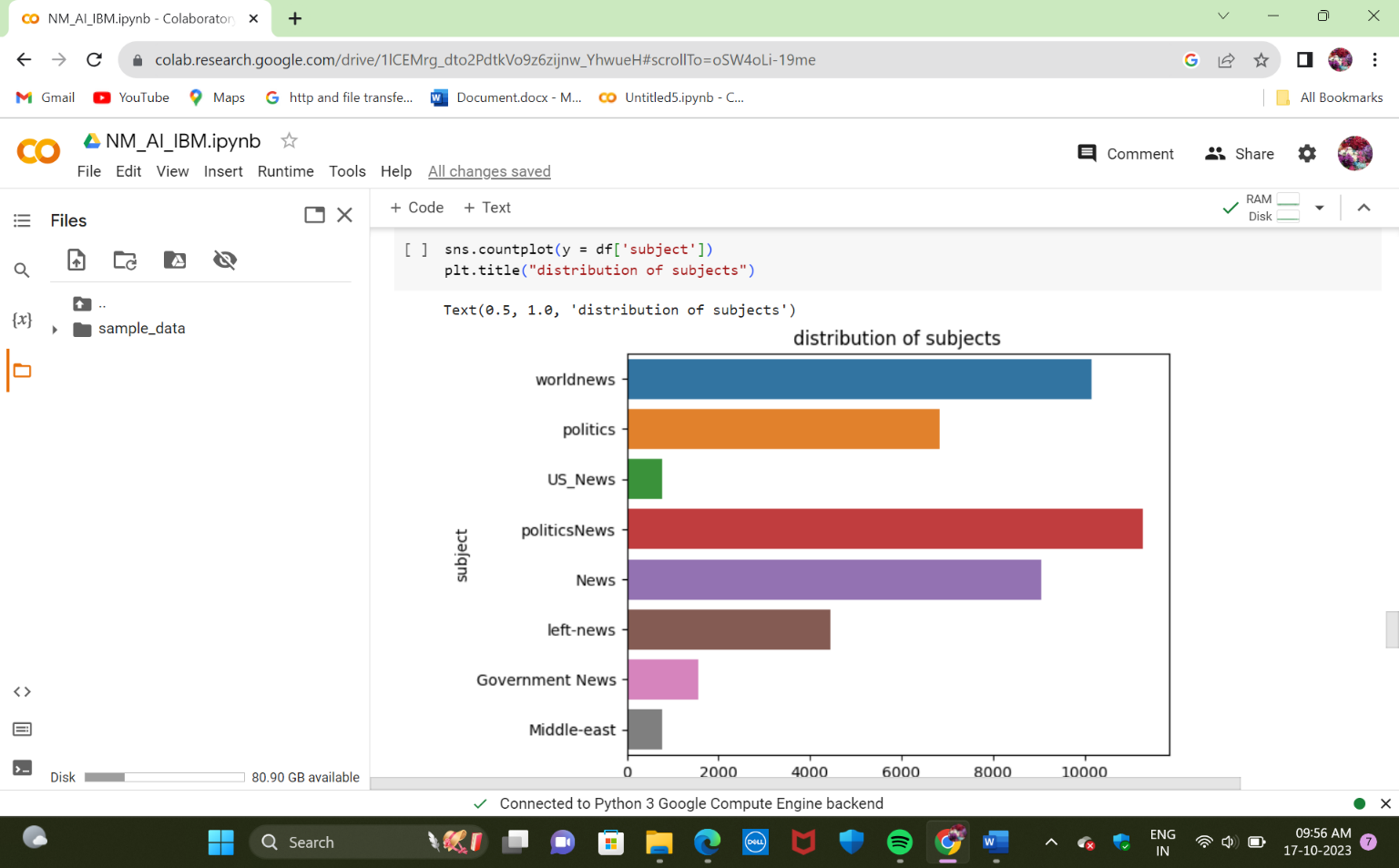
****

**#This code uses Seaborn and Matplotlib to create a countplot showing the distribution of "real" and "fake" news labels in a DataFrame and displays it with a title.**

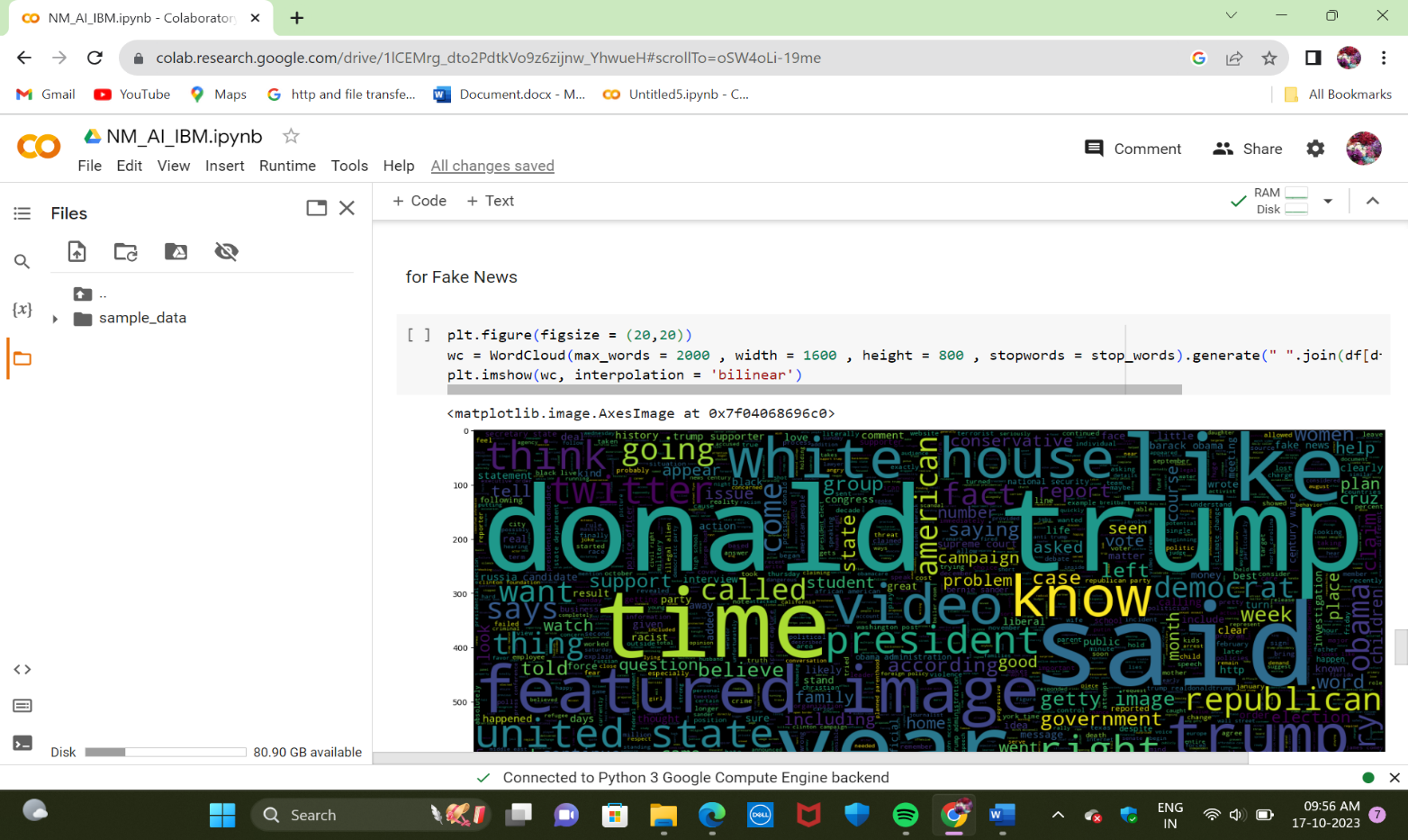
****

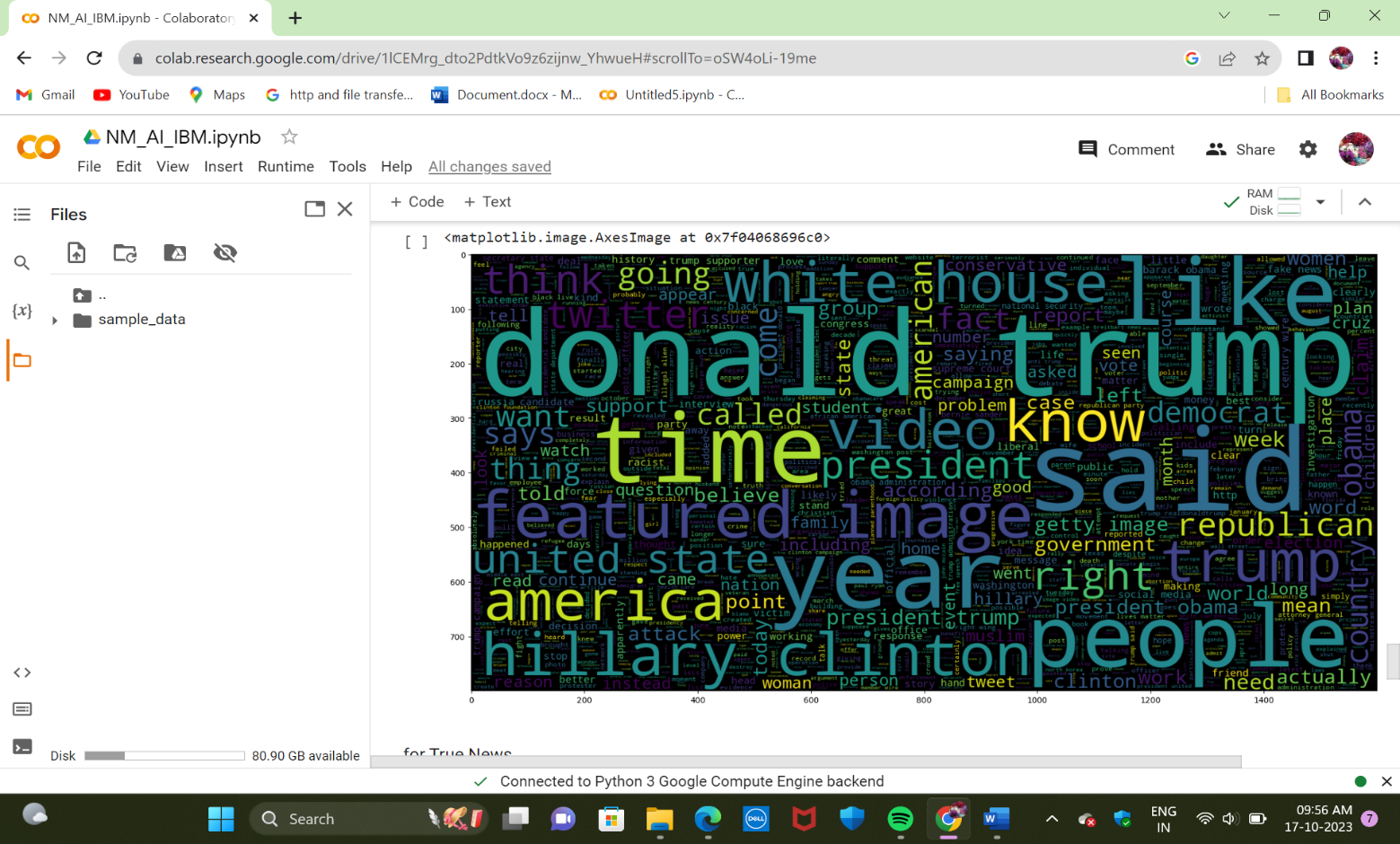
****

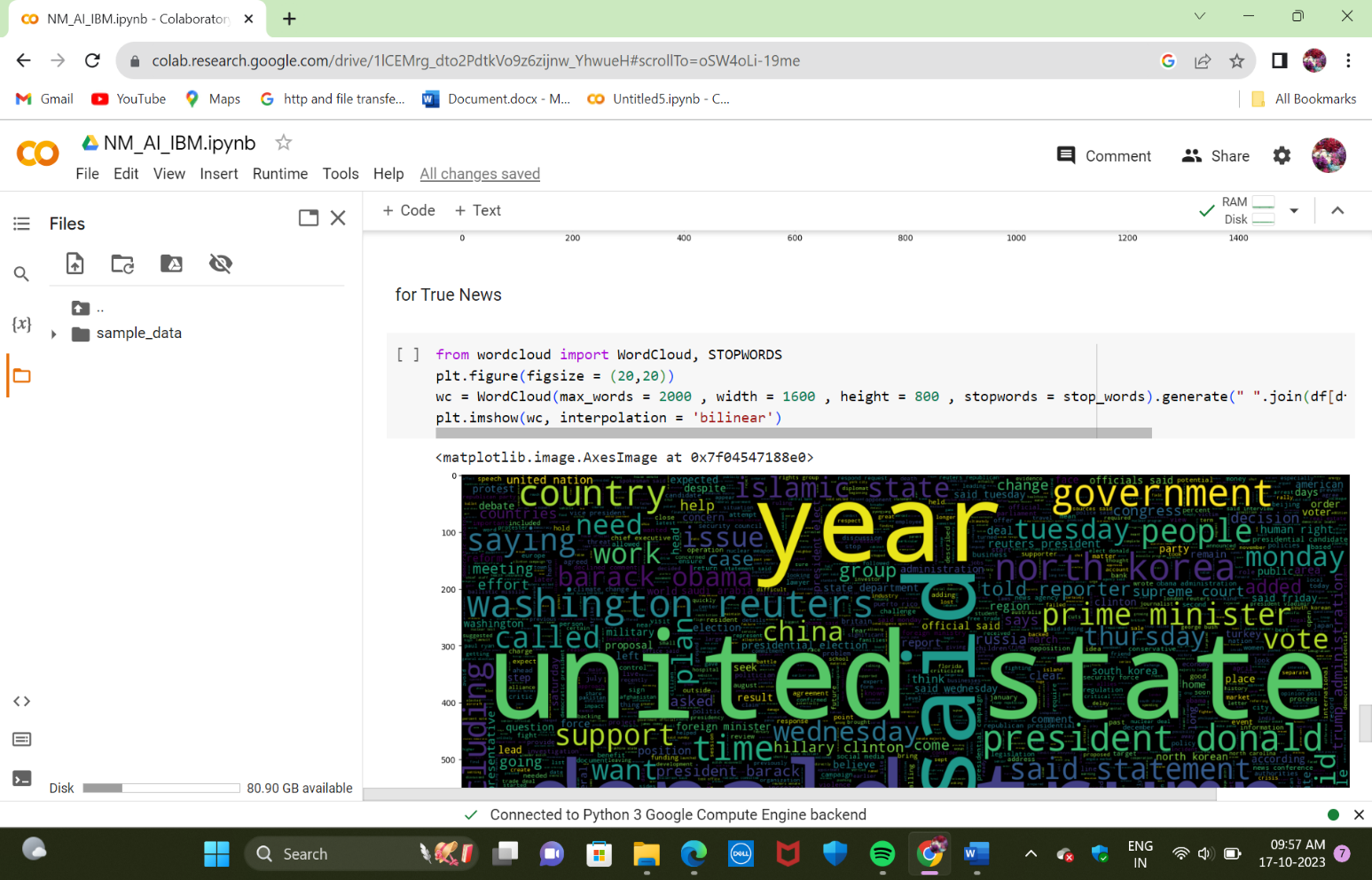
**#This code creates a countplot of the distribution of subjects in a DataFrame and sets the plot title accordingly.**

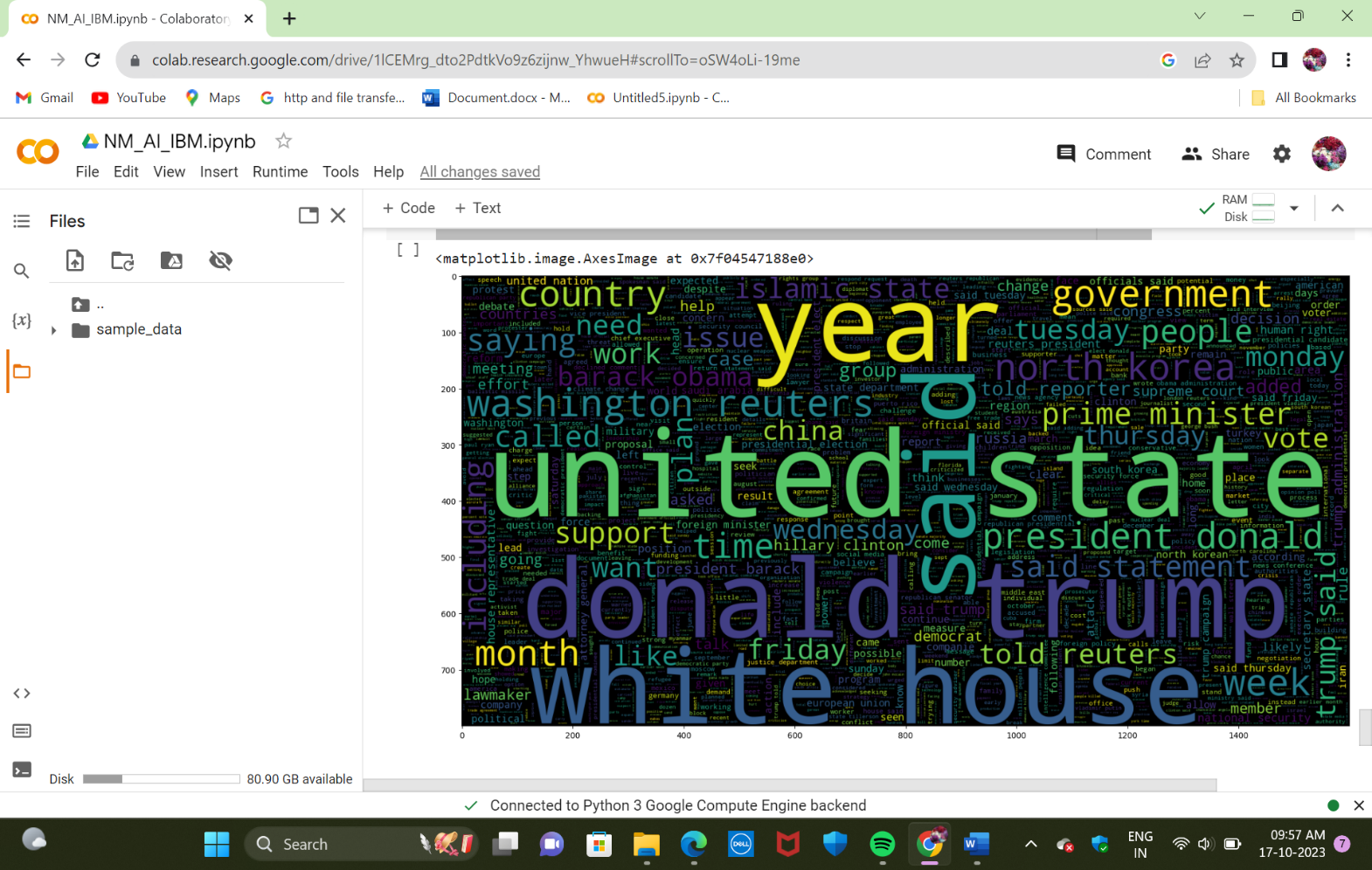
****

**#This code generates a Word Cloud visualization for the text data in a DataFrame where the "label" column has the value 0, with specific formatting options for word cloud size and stopwords, and displays the result.**

****

****

****

****