**CREATE A CHATBOT IN PYTHON**

TEAM MEMBERS

1. **SUYAMBULAKSHMI V**
2. **SHARINITHA R**
3. **AYESHASITHIKA S**
4. **PIRITHIBA M**

**Phase 4 – Document submission**

**OBJECTIVE:**

* The Objective of this project is to create a high-quality support to users, ensuring a positive user experience and customer satisfaction chatbot in Python that provides exceptional customer service, answering user queries on a website or application.

**PREPROCESSING:**

* Here preprocessing includes,

1. Data cleaning
2. Removing stop-words
3. Stemming and lemmatization

**DATA CLEANING:**

* By using pandas library “Dataframe” ,We can organize the given dataset.

**PROGRAM:**

These are the needed libraries needed to do preprocessing process which is

1. Removing stop – words
2. Stemming
3. Lemmatizing.

import re #regular expression to extract data

import nltk

import pandas as pd

import numpy as np

import spacy #spacy have some simple algorithm to do tokenization of the given dataset

from spacy.lang.en.stop\_words import STOP\_WORDS #english stopwords are extracted

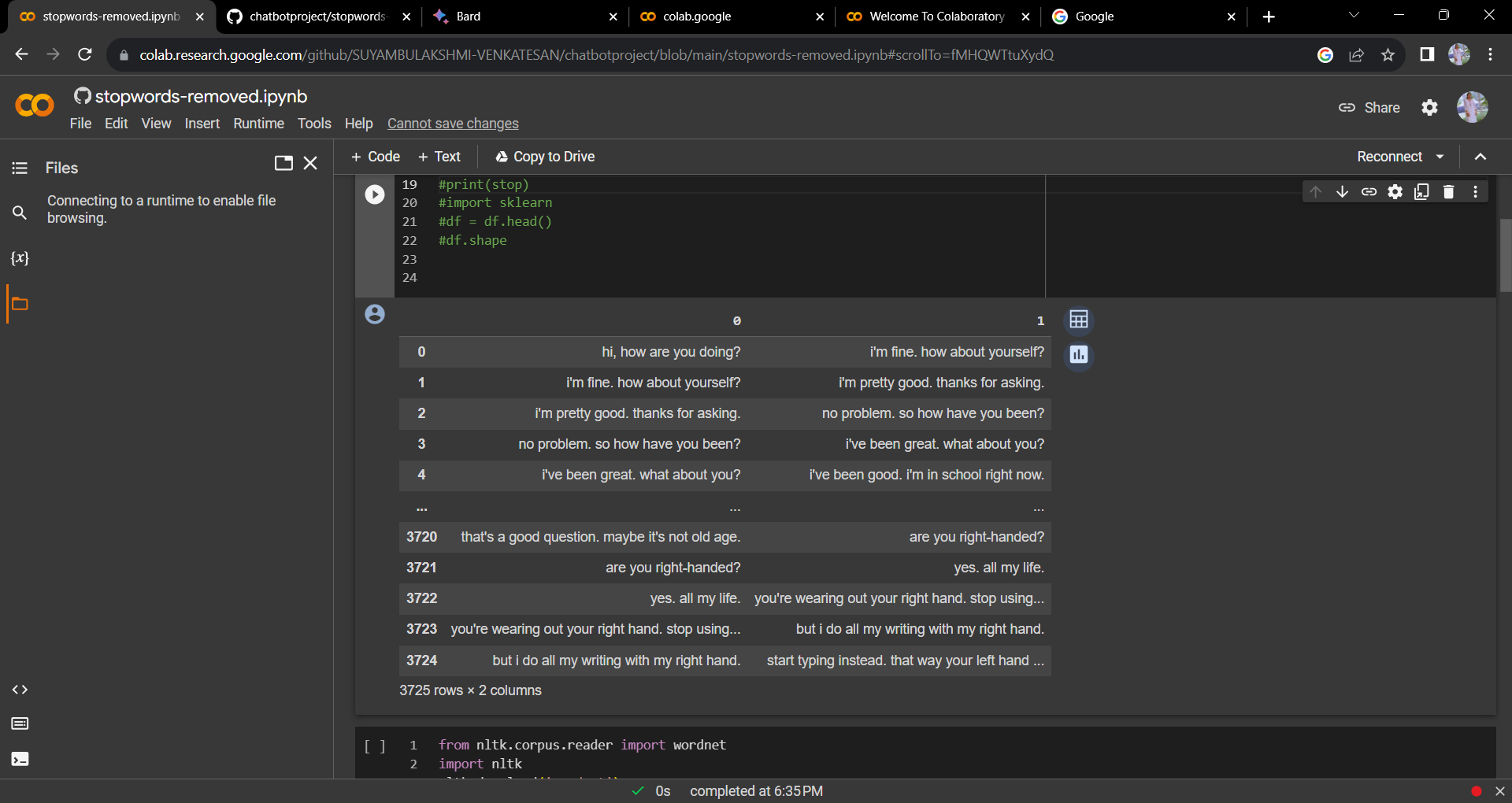
len(STOP\_WORDS) #To know about how many stopwords are available

nlp = spacy.load("en\_core\_web\_sm") #using english language as the preferred language

df = pd.read\_csv("dialogs.txt",sep='\t',header=None)

df #converted into table form

this is the output of the above program,



By using this code we can find out there is no null data.

**REMOVING STOP-WORDS:**

from nltk.corpus.reader import wordnet

import nltk

nltk.download('wordnet')

df = pd.read\_csv("dialogs.txt", sep='\t', header=None, names=["text"])

# Get the text from the DataFrame by header name

text\_by\_header\_name = df["text"].to\_string()

stopwords = set(STOP\_WORDS)

def removing\_stopwords(text):

  doc = text.split()

  no\_stop\_words =[word for word in doc if word not in stopwords]

  return " ".join(no\_stop\_words)

df["text"] = df["text"].apply(removing\_stopwords)

display(df["text"])

By using this **removing\_stopwords** function, we can able to remove stop words.

Finally we have succeccfully removed all the stop words from the dataset,

We have just executed to see the results to verify the given process is done or not.

**STEMMING AND LEMMATIZATION:**

For stemming and lemmatization , this is the code.

import nltk

nltk.download('averaged\_perceptron\_tagger')

import nltk

nltk.download('punkt')

tagger = nltk.tag.PerceptronTagger()

tagged\_text = tagger.tag(nltk.word\_tokenize(text\_by\_header\_name))

lemmatizer = nltk.WordNetLemmatizer()

#lemmatized\_text = [lemmatizer.lemmatize(token, pos) for token, pos in tagged\_text]

stemmer = nltk.PorterStemmer()

stemmed\_text = stemmer.stem(text\_by\_header\_name)

# Lemmatize the text

lemmatizer = nltk.WordNetLemmatizer()

lemmatized\_text = lemmatizer.lemmatize(text\_by\_header\_name)

print(stemmed\_text)

print(lemmatized\_text)

**TRAIN THE MODEL:**

import nltk

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.linear\_model import LogisticRegression

# Load the data

dialogs = pd.read\_csv("dialogs.txt", sep='\t', header=None, names=["text", "label"])

# Preprocess the data

# Remove stop words and punctuation

def preprocess\_text(text):

    text = nltk.word\_tokenize(text)

    text = [word for word in text if word not in STOP\_WORDS]

    text = [word for word in text if word.isalpha()]

    return " ".join(text)

dialogs["text"] = dialogs["text"].apply(preprocess\_text)

# Create TF-IDF vectors

vectorizer = TfidfVectorizer()

X = vectorizer.fit\_transform(dialogs["text"])

# Split the data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, dialogs["label"], test\_size=0.25)

# Train the model

clf = LogisticRegression()

clf.fit(X\_train, y\_train)

# Evaluate the model on the test set

y\_pred = clf.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy)

Since it is a linear model , so we use linear regression to train the model by using this code