import pandas as pd

import re

import nltk

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer

from sklearn.feature\_extraction.text import CountVectorizer

import seaborn as sns

import matplotlib.pyplot as plt

from tqdm.auto import tqdm

import time

data=pd.read\_csv("spam.csv",encoding = "latin")

data.head()

data.info()

data.tail()

data.isna().any()

data.isna().sum()

data['v2'].nunique()

data.shape

data['v2'].drop\_duplicates(inplace = True)

data

# Create a bar plot of the class distribution

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

class\_counts.plot(kind='bar')

plt.title('Class Distribution of Spam/Ham')

plt.xlabel('Spam/Ham')

plt.ylabel('Number of Mails')

plt.show()

from collections import Counter

import re

import nltk

from nltk.corpus import stopwords

# Concatenate all tweet texts into a single string

all\_text = ' '.join(data['v2'].values)

# Remove URLs, mentions, and hashtags from the text

all\_text = re.sub(r'http\S+', '', all\_text)

all\_text = re.sub(r'@\S+', '', all\_text)

all\_text = re.sub(r'#\S+', '', all\_text)

# Split the text into individual words

words = all\_text.split()

# Count the frequency of each word

word\_counts = Counter(words)

top\_words = word\_counts.most\_common(100)

top\_words

# Create a bar chart of the most common words

top\_words = word\_counts.most\_common(10) # Change the number to show more/less words

x\_values = [word[0] for word in top\_words]

y\_values = [word[1] for word in top\_words]

plt.bar(x\_values, y\_values)

plt.xlabel('Word')

plt.ylabel('Frequency')

plt.title('Most Commonly Used Words')

plt.show()

# Clean the data

def clean\_text(text):

# Remove HTML tags

text = re.sub('<.\*?>', '', text)

# Remove non-alphabetic characters and convert to lowercase

text = re.sub('[^a-zA-Z]', ' ', text).lower()

# Tokenize the text

words = nltk.word\_tokenize(text)

# Remove stopwords

words = [w for w in words if w not in stopwords.words('english')]

# Stem the words

stemmer = PorterStemmer()

words = [stemmer.stem(w) for w in words]

# Join the words back into a string

text = ' '.join(words)

return text

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

# train a Logistic Regression Model

clf = LogisticRegression()