AI-spam\_classifier

Preprocessing data

import numpy as np

import pandas as pd

df = pd.read\_csv('C:\Users\dhanasree\AI-spam\_classifier\AI-spam\_classifier\spam\_ham\_dataset.csv')

df.sample(5)

df.shape

**Data cleaning**

df.info()

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

df.sample(5)

df.rename(columns={'v1':'target','v2':'text'},inplace=True)

df.sample(5)

from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()

df['target'] = encoder.fit\_transform(df['target'])

df.head()

df.isnull().sum()

df.duplicated().sum()

df = df.drop\_duplicates(keep='first')

df.duplicated().sum()

df.shape

**EDA**

df.head()

df['target'].value\_counts()

import matplotlib.pyplot as plt

plt.pie(df['target'].value\_counts(), labels=['ham','spam'],autopct="%0.2f")

plt.show()

import nltk

nltk.download('punkt')

df['num\_characters'] = df['text'].apply(len)

df.head()

df['num\_words'] = df['text'].apply(lambda x:len(nltk.word\_tokenize(x)))

df.head()

df['num\_sentences'] = df['text'].apply(lambda x:len(nltk.sent\_tokenize(x)))

df.head()

df[['num\_characters','num\_words','num\_sentences']].describe()

df[df['target'] == 0][['num\_characters','num\_words','num\_sentences']].describe()

df[df['target'] == 1][['num\_characters','num\_words','num\_sentences']].describe()

import seaborn as sns

plt.figure(figsize=(12,6))

sns.histplot(df[df['target'] == 0]['num\_characters'])

sns.histplot(df[df['target'] == 1]['num\_characters'],color='red')

plt.figure(figsize=(12,6))

sns.histplot(df[df['target'] == 0]['num\_words'])

sns.histplot(df[df['target'] == 1]['num\_words'],color='red')

sns.pairplot(df,hue='target')

sns.heatmap(df.corr(),annot=True)

**Data Preprocessing**

def transform\_text(text):

text = text.lower()

text = nltk.word\_tokenize(text)

y = []

for i in text:

if i.isalnum():

y.append(i)

text = y[:]

y.clear()

for i in text:

if i not in stopwords.words('english') and i not in string.punctuation:

y.append(i)

text = y[:]

y.clear()

for i in text:

y.append(ps.stem(i))

return " ".join(y)

transform\_text("I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.")

df['text'][10]

from nltk.stem.porter import PorterStemmer

ps = PorterStemmer()

ps.stem('loving')

df['transformed\_text'] = df['text'].apply(transform\_text)

df.head()

from wordcloud import WordCloud

wc = WordCloud(width=500,height=500,min\_font\_size=10,background\_color='white')

spam\_wc = wc.generate(df[df['target'] == 1]['transformed\_text'].str.cat(sep=" "))

plt.figure(figsize=(15,6))

plt.imshow(spam\_wc)

ham\_wc = wc.generate(df[df['target'] == 0]['transformed\_text'].str.cat(sep=" "))

plt.figure(figsize=(15,6))

plt.imshow(ham\_wc)

df.head()

spam\_corpus = []

for msg in df[df['target'] == 1]['transformed\_text'].tolist():

for word in msg.split():

spam\_corpus.append(word)

len(spam\_corpus)

from collections import Counter

sns.barplot(pd.DataFrame(Counter(spam\_corpus).most\_common(30))[0],pd.DataFrame(Counter(spam\_corpus).most\_common(30))[1])

plt.xticks(rotation='vertical')

plt.show()

ham\_corpus = []

for msg in df[df['target'] == 0]['transformed\_text'].tolist():

for word in msg.split():

ham\_corpus.append(word)

len(ham\_corpus)

from collections import Counter

sns.barplot(pd.DataFrame(Counter(ham\_corpus).most\_common(30))[0],pd.DataFrame(Counter(ham\_corpus).most\_common(30))[1])

plt.xticks(rotation='vertical')

plt.show()

df.head()