**1.CLEANING DATA**

import numpy as np  
import pandas as pd  
data = pd.read\_csv('Spam Email Detection - spam.csv')

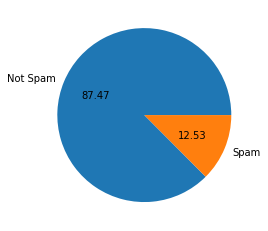
### 1. Data Cleaning  
data.info()  
data.drop(columns=['Unnamed: 2','Unnamed: 3','Unnamed: 4'],inplace = True) #droping last 3 columns  
data.shape  
data.sample(2)  
data.rename(columns={'v1':'Target','v2':'Mail-Text'},inplace=True)  
data.sample(2)

# Encoding data  
from sklearn.preprocessing import LabelEncoder  
encoder = LabelEncoder()  
data['Target'] = encoder.fit\_transform(data['Target'])  
data.tail(2)  
  
# check for missing values  
data.isna().sum()  
  
# check for duplicate values  
data.duplicated().sum()  
  
# drop duplicate values  
data = data.drop\_duplicates(keep = 'first')  
data.duplicated().sum()  
data.shape  
data.sample(4)

**2. Exploratory Data Analysis (EDA):**

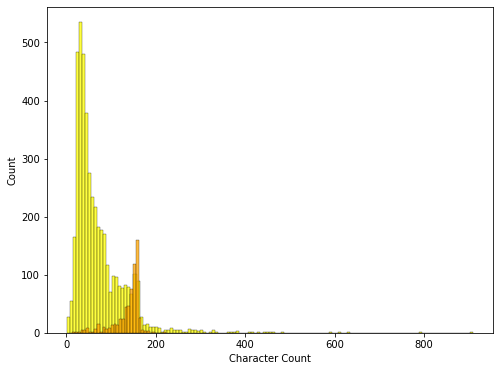
## 2. Data Visualization  
data['Target'].value\_counts()  
data.rename(columns={'Target':'Type'},inplace=True)  
data['Type'].value\_counts()

# pie chart  
import matplotlib.pyplot as plt  
plt.pie(data['Type'].value\_counts(),labels=['Not Spam','Spam'],autopct="%0.2f")  
plt.show()



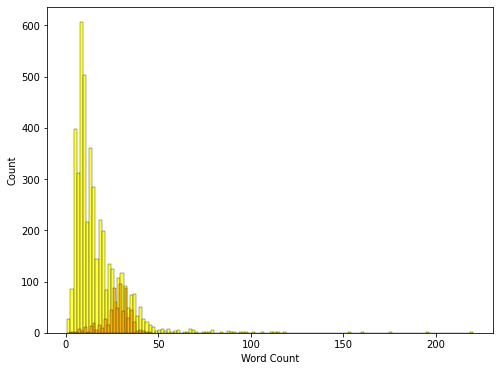
import nltk  
## if nltk is not present then run this command - pip install nltk  
nltk.download('punkt')  
data['num\_characters']=data['Mail-Text'].apply(len) # number of characters in a mail  
data.rename(columns={'num\_characters':'Character Count'},inplace=True)  
data.head(3)  
data['Mail-Text'].apply(lambda x:nltk.word\_tokenize(x)) ## Breaking sentence into number of words  
data['Mail-Text'].apply(lambda x:len(nltk.word\_tokenize(x))) ## Counting number of words used in a Mail  
data['Word Count'] = data['Mail-Text'].apply(lambda x:len(nltk.word\_tokenize(x)))  
data.head(4)  
data['Sentence Count']=data['Mail-Text'].apply(lambda x:len(nltk.sent\_tokenize(x)))  
data.tail(4)  
data.describe()  
# Describe function for Not Spam Messages  
data[data['Type'] == 0][['Character Count','Word Count','Sentence Count']].describe()  
  
# Describe function for Spam Messages  
data[data['Type'] == 1][['Character Count','Word Count','Sentence Count']].describe()

## if seaborn is not present then run this command-pip install seaborn  
import seaborn as sns  
  
## Histogram plot  
plt.figure(figsize=(8,6))  
sns.histplot(data[data['Type']==0]['Character Count'],color = 'yellow')  
sns.histplot(data[data['Type']==1]['Character Count'],color = 'orange')



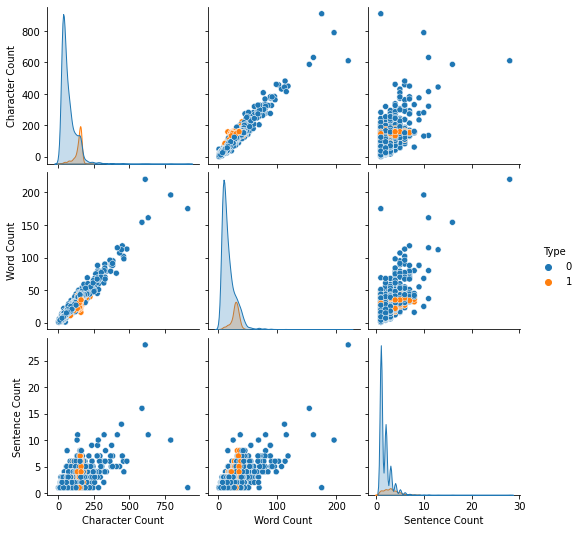
Histogram Plot 1

plt.figure(figsize=(8,6))  
sns.histplot(data[data['Type']==0]['Word Count'],color = 'yellow')  
sns.histplot(data[data['Type']==1]['Word Count'],color = 'orange')



Histogram Plot 2

sns.pairplot(data,hue='Type')



Pairplot

columns\_to\_select = ['Type', 'Character Count', 'Word Count', 'Sentence Count'] # Select only numeric columns  
numeric\_data = data[columns\_to\_select]  
  
## Heat map plot  
sns.heatmap(numeric\_data.corr(),annot=True)



Heatmap

**3. Text Preprocessing**

nltk.download('stopwords')  
from nltk.corpus import stopwords  
stopwords.words('english')  
import string  
string.punctuation  
from nltk.stem.porter import PorterStemmer  
ps = PorterStemmer()  
ps.stem('Roaming')

def transform\_text(text):  
 text = text.lower() # 1. Lower Case Convertion  
 text = nltk.word\_tokenize(text) # 2. Tokenization  
   
 y=[] # 3. Removing special Characters  
 for i in text:  
 if i.isalnum():  
 y.append(i)  
   
 text = y[:]  
 y.clear()  
   
 for i in text: # 4. Removing stop words and punctuation  
 if i not in stopwords.words('english') and i not in string.punctuation:  
 y.append(i)  
   
 text = y[:]  
 y.clear()  
   
 for i in text: # 5. Stemmimg  
 y.append(ps.stem(i))  
   
 return " ".join(y)

transform\_text('Okay name ur price as long as its legal! Wen can I pick them up? Y u ave x ams xx')  
data['Mail-Text'][100]  
data['Transformed-Text']=data['Mail-Text'].apply(transform\_text)

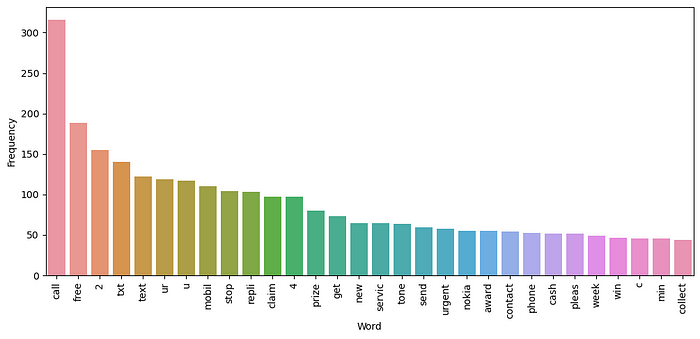
#### Word Cloud Formation:  
pip install wordcloud  
from wordcloud import WordCloud  
wc = WordCloud(width=1500,height=800,min\_font\_size=10,background\_color='white')  
spam\_wc = wc.generate(data[data['Type']==1]['Transformed-Text'].str.cat(sep=" "))  
plt.figure(figsize=(20,10))  
plt.imshow(spam\_wc)

non\_spam\_wc = wc.generate(data[data['Type']==0]['Transformed-Text'].str.cat(sep=" "))  
plt.figure(figsize=(12,5))  
plt.imshow(non\_spam\_wc)

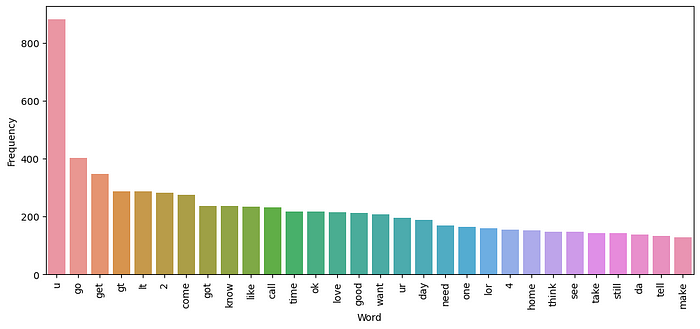
Non-spam word cloud

spam\_word = []  
for msg in data[data['Type']==1]['Transformed-Text'].tolist():  
 for word in msg.split():  
 spam\_word.append(word)  
len(spam\_word)  
from collections import Counter  
Counter(spam\_word)  
pd.DataFrame(Counter(spam\_word).most\_common(30))

data\_frame = pd.DataFrame(Counter(spam\_word).most\_common(30), columns=['Word', 'Frequency'])  
plt.figure(figsize=(12, 5))  
sns.barplot(x='Word', y='Frequency', data=data\_frame)  
plt.xticks(rotation='vertical')  
plt.show()

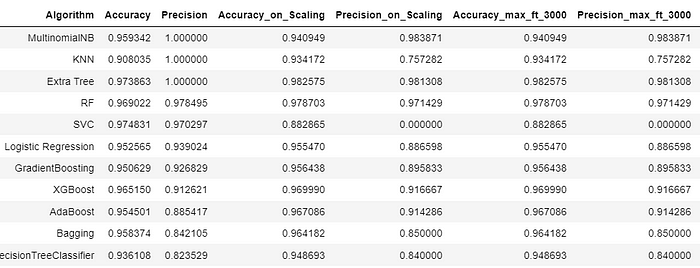


non\_spam\_word = []  
for msg in data[data['Type']==0]['Transformed-Text'].tolist():  
 for word in msg.split():  
 non\_spam\_word.append(word)  
len(non\_spam\_word)  
data\_frame = pd.DataFrame(Counter(non\_spam\_word).most\_common(30), columns=['Word', 'Frequency'])  
plt.figure(figsize=(12, 5))  
sns.barplot(x='Word', y='Frequency', data=data\_frame)  
plt.xticks(rotation='vertical')  
plt.show()

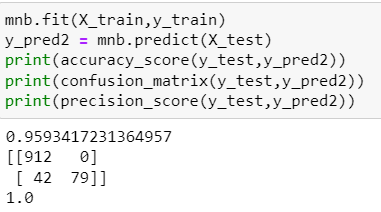


**4. Model Building:**

**5. Model Improvement:**



**6. Model Evaluation**



**Deployment:**.

