EX.No:3

Reg.No:953622243037

Date:04.10.2023

Objective:

**Perform exploratory data analysis (EDA) on with datasets like email data set. Export all your emails as a dataset, import them inside a panda’s data frame, visualize them  
and get different insights from the data**.

**Dataset:**

**The dataset is obtained by daily analysis of the email from your mail to know about the mail that you have been received and sent as the dataset.**

Program:

import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt

from google.colab import drive  
drive.mount('/content/drive/')

Drive already mounted at /content/drive/; to attempt to forcibly remount, call drive.mount("/content/drive/", force\_remount=True).

from pydrive.auth import GoogleAuth  
from pydrive.drive import GoogleDrive

import mailbox  
mboxfile = "/content/All mail Including Spam and Trash.mbox"  
mbox = mailbox.mbox(mboxfile)  
mbox

<mailbox.mbox at 0x7bf9f0c5fd30>

for key in mbox[0].keys():  
 print(key)

X-GM-THRID  
X-Gmail-Labels  
Delivered-To  
Received  
X-Google-Smtp-Source  
X-Received  
ARC-Seal  
ARC-Message-Signature  
ARC-Authentication-Results  
Return-Path  
Received  
Received-SPF  
Authentication-Results  
DKIM-Signature  
DKIM-Signature  
X-On-bounce-route-to  
Date  
From  
Reply-To  
To  
Message-ID  
Subject  
Mime-Version  
Content-Type  
Content-Transfer-Encoding  
Auto-Submitted  
Feedback-ID  
X-SES-Outgoing

import csv

with open('mailbox.csv', 'w') as outputfile:  
 writer = csv.writer(outputfile)  
 writer.writerow(['subject','from','date','to','label','thread','Cc'])  
 for message in mbox:  
 writer.writerow([  
 message['subject'],  
 message['from'],  
 message['date'],  
 message['to'],  
 message['X-Gmail-Labels'],  
 message['X-GM-THRID'],  
 message['Cc']  
 ]  
)

dfs = pd.read\_csv('mailbox.csv', names=['subject', 'from', 'date', 'to','label', 'thread','Cc'])

dfs.dtypes

subject object  
from object  
date object  
to object  
label object  
thread object  
Cc object  
dtype: object

#converting the date  
dfs['date'] = dfs['date'].apply(lambda x: pd.to\_datetime(x,errors='coerce', utc=True))

# Removing NaN values  
  
dfs = dfs[dfs['date'].notna()]  
dfs.to\_csv('gmail.csv')

#Applying descriptive statistics  
dfs.info()

<class 'pandas.core.frame.DataFrame'>  
Int64Index: 190 entries, 1 to 190  
Data columns (total 7 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 subject 173 non-null object   
 1 from 190 non-null object   
 2 date 190 non-null datetime64[ns, UTC]  
 3 to 189 non-null object   
 4 label 190 non-null object   
 5 thread 190 non-null object   
 6 Cc 39 non-null object   
dtypes: datetime64[ns, UTC](1), object(6)  
memory usage: 11.9+ KB

dfs.head(10)

subject \  
1 Recent Canvas Notifications   
2 Welcome to a wonderful world of learning   
3 Recent Canvas Notifications   
4 Folder shared with you: "Cod Soft Intern"   
5 Item shared with you: "Copy of priyar.ipynb"   
6 Participate in =?utf-8?Q?World=E2=80=99s?= Big...   
7 Recent Canvas Notifications   
8 Delivery Status Notification (Failure)   
9 Fwd: Perform exploratory data analysis (EDA) o...   
10 Perform exploratory data analysis (EDA) on dat...   
  
 from \  
1 Canvas Free for Teachers <notifications@instru...   
2 "Coursera" <Coursera@email.coursera.org>   
3 Canvas Free for Teachers <notifications@instru...   
4 "SelvaBirunda RIT (via Google Drive)" <selvabi...   
5 "LAKSHMI PRIYA K (via Google Drive)" <drive-sh...   
6 Unstop <noreply@dare2compete.news>   
7 Canvas Free for Teachers <notifications@instru...   
8 Mail Delivery Subsystem <mailer-daemon@googlem...   
9 ILAKKIYAN G <953622243037@ritrjpm.ac.in>   
10 DHANESH KUMAR P <953622243021@ritrjpm.ac.in>   
  
 date to \  
1 2023-10-03 00:20:26+00:00 22AD019@ritrjpm.ac.in   
2 2023-09-28 15:36:16+00:00 <953622243037@ritrjpm.ac.in>   
3 2023-09-30 15:31:54+00:00 22AD019@ritrjpm.ac.in   
4 2023-09-28 10:17:22+00:00 ad2022a@ritrjpm.ac.in   
5 2023-10-02 15:01:48+00:00 953622243037@ritrjpm.ac.in   
6 2023-10-03 09:20:24+00:00 953622243037@ritrjpm.ac.in   
7 2023-10-02 00:21:37+00:00 22AD019@ritrjpm.ac.in   
8 2023-10-03 10:00:58+00:00 953622243037@ritrjpm.ac.in   
9 2023-10-03 09:58:04+00:00 kumaearavinth960@gmail.com   
10 2023-10-03 09:30:37+00:00 ad2022a@ritrjpm.ac.in   
  
 label thread \  
1 Inbox,Opened,Category Updates 1778691529068117300   
2 Inbox,Opened,Category Updates 1778296162936398061   
3 Inbox,Opened,Category Updates 1778477081829506592   
4 Inbox,Category Forums,Unread 1778276099729942871   
5 Inbox,Opened,Category Personal 1778656382872939118   
6 Inbox,Category Updates,Unread 1778725502352379084   
7 Inbox,Important,Category Updates,Unread 1778601005464320334   
8 Inbox,Important,Opened,Category Updates 1778726346006386433   
9 Sent,Opened 1778726346006386433   
10 Inbox,Opened,Category Forums 1778726346006386433   
  
 Cc   
1 NaN   
2 NaN   
3 NaN   
4 ad2022b@ritrjpm.ac.in   
5 NaN   
6 NaN   
7 NaN   
8 NaN   
9 NaN   
10 NaN

#Data refactoring  
import re  
def extract\_email\_ID(string):  
 email = re.findall(r'<(.+?)>', string)  
 if not email:  
 email = list(filter(lambda y: '@' in y, string.split()))  
 return email[0] if email else np.nan

dfs['from'] = dfs['from'].apply(lambda x: extract\_email\_ID(x))

myemail = '953622243037@ritrjpm.ac.in'  
dfs['label'] = dfs['from'].apply(lambda x: 'sent' if x==myemail else 'inbox')

#Dropping columns  
dfs.drop(columns='to', inplace=True)

dfs

subject \  
1 Recent Canvas Notifications   
2 Welcome to a wonderful world of learning   
3 Recent Canvas Notifications   
4 Folder shared with you: "Cod Soft Intern"   
5 Item shared with you: "Copy of priyar.ipynb"   
.. ...   
186 RIT Photography Club - Office Bearers - Willin...   
187 Fwd: dev ass.docx   
188 Re: dev ass.docx   
189 Fwd: dev ass.docx   
190 dev ass.docx   
  
 from date label \  
1 notifications@instructure.com 2023-10-03 00:20:26+00:00 inbox   
2 Coursera@email.coursera.org 2023-09-28 15:36:16+00:00 inbox   
3 notifications@instructure.com 2023-09-30 15:31:54+00:00 inbox   
4 selvabirunda@ritrjpm.ac.in 2023-09-28 10:17:22+00:00 inbox   
5 drive-shares-dm-noreply@google.com 2023-10-02 15:01:48+00:00 inbox   
.. ... ... ...   
186 venkateshm@ritrjpm.ac.in 2023-09-22 04:00:28+00:00 inbox   
187 953622243037@ritrjpm.ac.in 2023-09-26 07:06:16+00:00 inbox   
188 953622243037@ritrjpm.ac.in 2023-09-26 07:05:21+00:00 inbox   
189 953622243037@ritrjpm.ac.in 2023-09-26 06:54:45+00:00 inbox   
190 953622243037@ritrjpm.ac.in 2023-09-23 11:34:46+00:00 inbox   
  
 thread Cc   
1 1778691529068117300 NaN   
2 1778296162936398061 NaN   
3 1778477081829506592 NaN   
4 1778276099729942871 ad2022b@ritrjpm.ac.in   
5 1778656382872939118 NaN   
.. ... ...   
186 1777708830374716171 NaN   
187 1777828025496575628 NaN   
188 1777828025496575628 NaN   
189 1777828025496575628 NaN   
190 1777828025496575628 NaN   
  
[190 rows x 6 columns]

import pytz  
def refactor\_timezone(x):  
 est = pytz.timezone('US/Eastern')  
 return x.astimezone(est)

dfs['date'] = dfs['date'].apply(lambda x: refactor\_timezone(x))

**Observation:**

**The above help us to convert the day of the week variable into the name of the day, as in, Saturday, Sunday, and so on.**  
dfs['dayofweek'] = dfs['date'].apply(lambda x: x.day\_name())  
dfs['dayofweek'] = pd.Categorical(dfs['dayofweek'], categories=['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday','Saturday', 'Sunday'], ordered=True)

**Observation:**

**The above program help us to do same process for the time of the**

**Day.**dfs['timeofday'] = dfs['date'].apply(lambda x: x.hour + x.minute/60+ x.second/3600)

dfs['hour'] = dfs['date'].apply(lambda x: x.hour)

dfs['year\_int'] = dfs['date'].apply(lambda x: x.year)

dfs['year'] = dfs['date'].apply(lambda x: x.year +x.dayofyear/365.25)

dfs.index = dfs['date']  
del dfs['date']

#EDA  
#1. How many emails did I send during a given timeframe?  
#2. At what times of the day do I send and receive emails with Gmail?  
#3. What is the average number of emails per day?  
#4. What is the average number of emails per hour?  
#5. Whom do I communicate with most frequently?  
#6. What are the most active emailing days?  
#7. What am I mostly emailing about?

#Number of emails  
print(dfs.index.min().strftime('%a, %d %b %Y %I:%M %p'))  
print(dfs.index.max().strftime('%a, %d %b %Y %I:%M %p'))  
print(dfs['label'].value\_counts())

Sun, 11 Dec 2022 02:33 AM  
Tue, 03 Oct 2023 06:00 AM  
inbox 171  
sent 19  
Name: label, dtype: int64

#two sub-dataframe—one for sent emails and another for received emails:  
sent = dfs[dfs['label']=='sent']  
received = dfs[dfs['label']=='inbox']

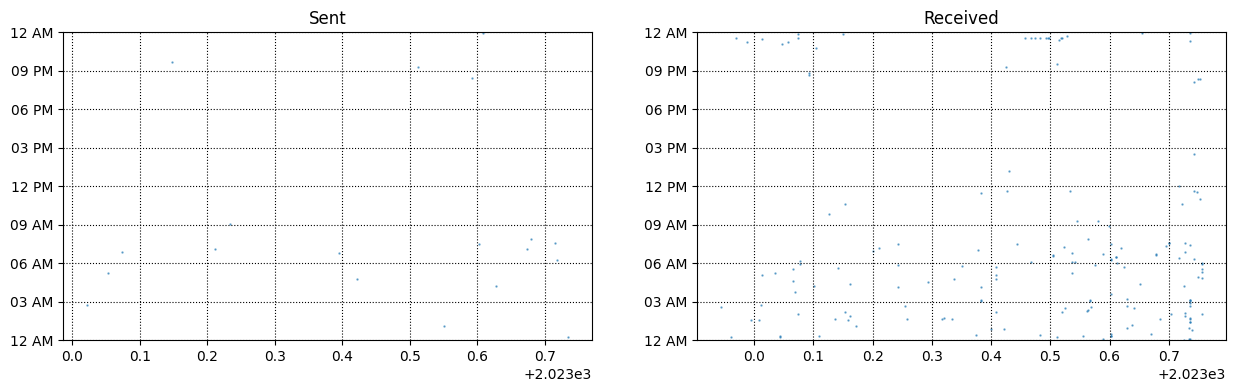
import matplotlib.pyplot as plt  
from matplotlib.ticker import MaxNLocator  
from scipy import ndimage  
import matplotlib.gridspec as gridspec  
import matplotlib.patches as mpatches

#create a function that takes a dataframe as an input and creates a plot  
def plot\_todo\_vs\_year(df, ax, color='C0', s=0.5, title=''):  
 ind = np.zeros(len(df), dtype='bool')  
 est = pytz.timezone('US/Eastern')  
 df[~ind].plot.scatter('year', 'timeofday', s=s, alpha=0.6, ax=ax,color=color)  
 ax.set\_ylim(0, 24)  
 ax.yaxis.set\_major\_locator(MaxNLocator(8))  
 ax.set\_yticklabels([datetime.datetime.strptime(str(int(np.mod(ts,24))), "%H").strftime("%I %p") for ts in ax.get\_yticks()]);  
 ax.set\_xlabel('')  
 ax.set\_ylabel('')  
 ax.set\_title(title)  
 ax.grid(ls=':', color='k')  
 return ax

import datetime  
fig, ax = plt.subplots(nrows=1, ncols=2, figsize=(15, 4))  
plot\_todo\_vs\_year(sent, ax[0], title='Sent')  
plot\_todo\_vs\_year(received, ax[1], title='Received')

<ipython-input-92-375026574b8d>:8: UserWarning: FixedFormatter should only be used together with FixedLocator  
 ax.set\_yticklabels([datetime.datetime.strptime(str(int(np.mod(ts,24))), "%H").strftime("%I %p") for ts in ax.get\_yticks()]);

<Axes: title={'center': 'Received'}>



**Observation:**

**The above grap shows the average emails per day and hour that has received from your mail id**. **The above graph represent the email that had been received is more than the the email that had been sent. The email that had received more at time between the 12 am to 12 pm in the seventh day of the week.**

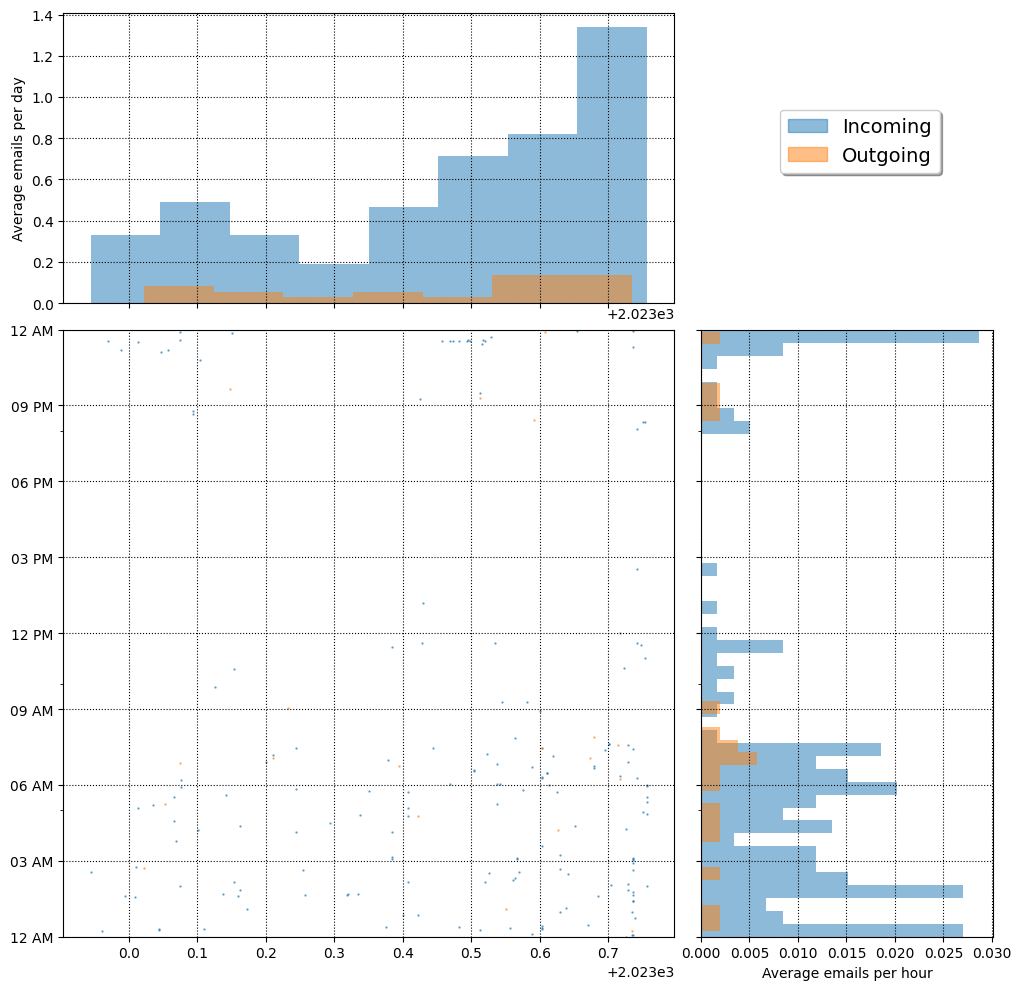
def plot\_number\_perday\_per\_year(df, ax, label=None, dt=0.3,\*\*plot\_kwargs):  
 year = df[df['year'].notna()]['year'].values  
 T = year.max() - year.min()  
 bins = int(T / dt)  
 weights = 1 / (np.ones\_like(year) \* dt \* 365.25)  
 ax.hist(year, bins=bins, weights=weights, label=label,\*\*plot\_kwargs);  
 ax.grid(ls=':', color='k')

#creates a function that plots the average number of emails per day  
def plot\_number\_perdhour\_per\_year(df, ax, label=None, dt=1,smooth=False,weight\_fun=None, \*\*plot\_kwargs):  
 tod = df[df['timeofday'].notna()]['timeofday'].values  
 year = df[df['year'].notna()]['year'].values  
 Ty = year.max() - year.min()  
 T = tod.max() - tod.min()  
 bins = int(T / dt)  
 if weight\_fun is None:  
 weights = 1 / (np.ones\_like(tod) \* Ty \* 365.25 / dt)  
 else:  
 weights = weight\_fun(df)  
 if smooth:  
 hst, xedges = np.histogram(tod, bins=bins,weights=weights);  
 x = np.delete(xedges, -1) + 0.5\*(xedges[1] - xedges[0])  
 hst = ndimage.gaussian\_filter(hst, sigma=0.75)  
 f = interp1d(x, hst, kind='cubic')  
 x = np.linspace(x.min(), x.max(), 10000)  
 hst = f(x)  
 ax.plot(x, hst, label=label, \*\*plot\_kwargs)  
 else:  
 ax.hist(tod, bins=bins, weights=weights, label=label,\*\*plot\_kwargs);  
 ax.grid(ls=':', color='k')  
 orientation = plot\_kwargs.get('orientation')  
 if orientation is None or orientation == 'vertical':  
 ax.set\_xlim(0, 24)  
 ax.xaxis.set\_major\_locator(MaxNLocator(8))  
 ax.set\_xticklabels([datetime.datetime.strptime(str(int(np.mod(ts,24))), "%H").strftime("%I %p") for ts in ax.get\_xticks()]);  
 elif orientation == 'horizontal':  
 ax.set\_ylim(0, 24)  
 ax.yaxis.set\_major\_locator(MaxNLocator(8))  
 ax.set\_yticklabels([datetime.datetime.strptime(str(int(np.mod(ts, 24))), "%H").strftime("%I %p") for ts in ax.get\_yticks()]);

# create a class that plots the time of the day versus year for all the emails within the given timeframe  
class TriplePlot:  
 def \_\_init\_\_(self):  
 gs = gridspec.GridSpec(6, 6)  
 self.ax1 = plt.subplot(gs[2:6, :4])  
 self.ax2 = plt.subplot(gs[2:6, 4:6], sharey=self.ax1)  
 plt.setp(self.ax2.get\_yticklabels(), visible=False);  
 self.ax3 = plt.subplot(gs[:2, :4])  
 plt.setp(self.ax3.get\_xticklabels(), visible=False);  
 def plot(self, df, color='darkblue', alpha=0.8, markersize=0.5,yr\_bin=0.1, hr\_bin=0.5):  
 plot\_todo\_vs\_year(df, self.ax1, color=color, s=markersize)  
 plot\_number\_perdhour\_per\_year(df, self.ax2, dt=hr\_bin,color=color, alpha=alpha, orientation='horizontal')  
 self.ax2.set\_xlabel('Average emails per hour')  
 plot\_number\_perday\_per\_year(df, self.ax3, dt=yr\_bin,color=color, alpha=alpha)  
 self.ax3.set\_ylabel('Average emails per day')

#instantiate the class to plot the graph:  
import matplotlib.gridspec as gridspec  
import matplotlib.patches as mpatches  
import datetime  
plt.figure(figsize=(12,12));  
tpl = TriplePlot()  
tpl.plot(received, color='C0', alpha=0.5)  
tpl.plot(sent, color='C1', alpha=0.5)  
p1 = mpatches.Patch(color='C0', label='Incoming', alpha=0.5)  
p2 = mpatches.Patch(color='C1', label='Outgoing', alpha=0.5)  
plt.legend(handles=[p1, p2], bbox\_to\_anchor=[1.45, 0.7],fontsize=14, shadow=True);

<ipython-input-92-375026574b8d>:8: UserWarning: FixedFormatter should only be used together with FixedLocator  
 ax.set\_yticklabels([datetime.datetime.strptime(str(int(np.mod(ts,24))), "%H").strftime("%I %p") for ts in ax.get\_yticks()]);  
<ipython-input-96-70eade5c248e>:31: UserWarning: FixedFormatter should only be used together with FixedLocator  
 ax.set\_yticklabels([datetime.datetime.strptime(str(int(np.mod(ts, 24))), "%H").strftime("%I %p") for ts in ax.get\_yticks()]);

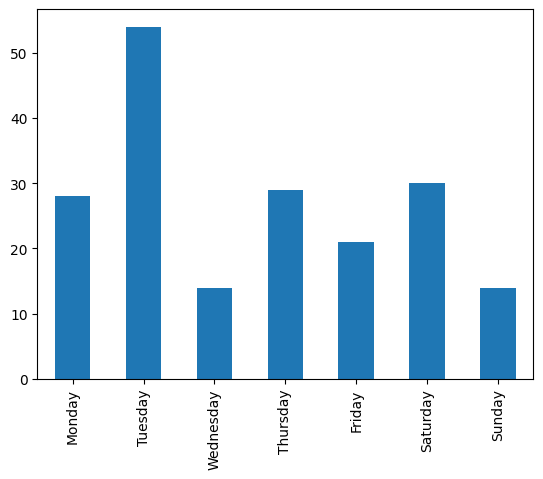


Observation:

The above graph show the number of emails received per day from your mail. The incoming email is more than the number email that has been incoming. The average mail per day which had been received more in Sunday and least in the Wednesday. The average mail per hour the email had been received more at 12 am and least between the 9AM to 3PM.

#find the busiest day of the week in terms of emails:  
counts = dfs.dayofweek.value\_counts(sort=False)  
counts.plot(kind='bar')

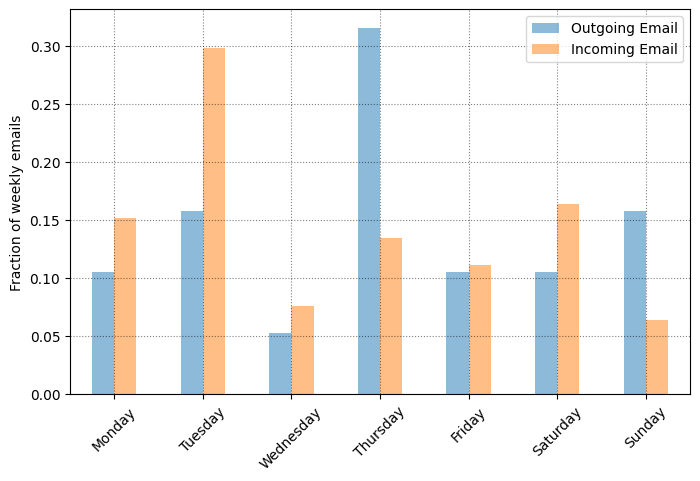
<Axes: >



**Observation:**

**The above graph shows the most active days for receiving and sending emails separately from your dataset. The above shows the most busiet day is Tuesday and most laziest day is Wednesday and there is slight varaiance between the Monday ,Thursday,Saturday.**

sdw = sent.groupby('dayofweek').size() / len(sent)  
rdw = received.groupby('dayofweek').size() / len(received)  
df\_tmp = pd.DataFrame(data={'Outgoing Email': sdw, 'Incoming Email':rdw})  
df\_tmp.plot(kind='bar', rot=45, figsize=(8,5), alpha=0.5)  
plt.xlabel('');  
plt.ylabel('Fraction of weekly emails');  
plt.grid(ls=':', color='k', alpha=0.5)



**Observation:**

**The above graph shows that the weekly emails that has been incoming email**

**and the outgoing email tracked in the above graph. The maximum number of email that had been as outgoing email in thursday and the minimum outgoing email in Wednesday . The maximum number incoming email in Tuesday and the minimum number of email that has been Sunday.**

import matplotlib.pyplot as plt  
import seaborn as sns  
plt.figure(figsize=(10,10))  
dfs

subject \  
date   
2023-10-02 20:20:26-04:00 Recent Canvas Notifications   
2023-09-28 11:36:16-04:00 Welcome to a wonderful world of learning   
2023-09-30 11:31:54-04:00 Recent Canvas Notifications   
2023-09-28 06:17:22-04:00 Folder shared with you: "Cod Soft Intern"   
2023-10-02 11:01:48-04:00 Item shared with you: "Copy of priyar.ipynb"   
... ...   
2023-09-22 00:00:28-04:00 RIT Photography Club - Office Bearers - Willin...   
2023-09-26 03:06:16-04:00 Fwd: dev ass.docx   
2023-09-26 03:05:21-04:00 Re: dev ass.docx   
2023-09-26 02:54:45-04:00 Fwd: dev ass.docx   
2023-09-23 07:34:46-04:00 dev ass.docx   
  
 from label \  
date   
2023-10-02 20:20:26-04:00 notifications@instructure.com inbox   
2023-09-28 11:36:16-04:00 Coursera@email.coursera.org inbox   
2023-09-30 11:31:54-04:00 notifications@instructure.com inbox   
2023-09-28 06:17:22-04:00 selvabirunda@ritrjpm.ac.in inbox   
2023-10-02 11:01:48-04:00 drive-shares-dm-noreply@google.com inbox   
... ... ...   
2023-09-22 00:00:28-04:00 venkateshm@ritrjpm.ac.in inbox   
2023-09-26 03:06:16-04:00 953622243037@ritrjpm.ac.in inbox   
2023-09-26 03:05:21-04:00 953622243037@ritrjpm.ac.in inbox   
2023-09-26 02:54:45-04:00 953622243037@ritrjpm.ac.in inbox   
2023-09-23 07:34:46-04:00 953622243037@ritrjpm.ac.in inbox   
  
 thread Cc \  
date   
2023-10-02 20:20:26-04:00 1778691529068117300 NaN   
2023-09-28 11:36:16-04:00 1778296162936398061 NaN   
2023-09-30 11:31:54-04:00 1778477081829506592 NaN   
2023-09-28 06:17:22-04:00 1778276099729942871 ad2022b@ritrjpm.ac.in   
2023-10-02 11:01:48-04:00 1778656382872939118 NaN   
... ... ...   
2023-09-22 00:00:28-04:00 1777708830374716171 NaN   
2023-09-26 03:06:16-04:00 1777828025496575628 NaN   
2023-09-26 03:05:21-04:00 1777828025496575628 NaN   
2023-09-26 02:54:45-04:00 1777828025496575628 NaN   
2023-09-23 07:34:46-04:00 1777828025496575628 NaN   
  
 dayofweek timeofday hour year\_int year   
date   
2023-10-02 20:20:26-04:00 Monday 20.340556 20 2023 2023.752909   
2023-09-28 11:36:16-04:00 Thursday 11.604444 11 2023 2023.741958   
2023-09-30 11:31:54-04:00 Saturday 11.531667 11 2023 2023.747433   
2023-09-28 06:17:22-04:00 Thursday 6.289444 6 2023 2023.741958   
2023-10-02 11:01:48-04:00 Monday 11.030000 11 2023 2023.752909   
... ... ... ... ... ...   
2023-09-22 00:00:28-04:00 Friday 0.007778 0 2023 2023.725530   
2023-09-26 03:06:16-04:00 Tuesday 3.104444 3 2023 2023.736482   
2023-09-26 03:05:21-04:00 Tuesday 3.089167 3 2023 2023.736482   
2023-09-26 02:54:45-04:00 Tuesday 2.912500 2 2023 2023.736482   
2023-09-23 07:34:46-04:00 Saturday 7.579444 7 2023 2023.728268   
  
[190 rows x 10 columns]

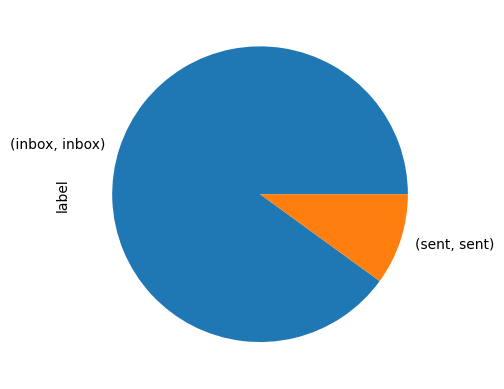
<Figure size 1000x1000 with 0 Axes>

dfs.groupby('label').groups.keys()

dict\_keys(['inbox', 'sent'])

lg = dfs.groupby('label')  
counts = lg.label.value\_counts(sort=False)  
counts.plot(kind='pie')

<Axes: ylabel='label'>



**Observation:**

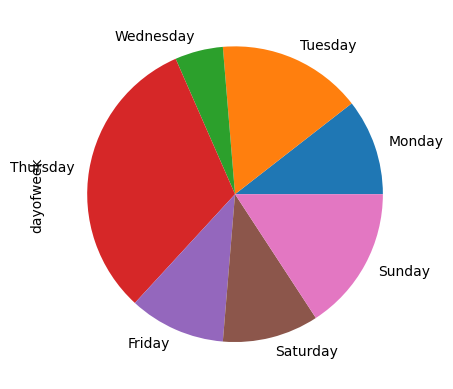
**The above graphs shows that the number of emails that has been sent and reveceived as email in the form of pie chart. The pie chart shoes that the maximum number of email has been received from the mail id and minmum number of email that had been sent from the inbox.**

# Get values items from group with value sent  
se=lg.get\_group("sent")  
se

subject \  
date   
2023-09-05 07:53:28-04:00 Re: Accenture Innovation Challenge 2023   
2023-08-04 20:26:01-04:00 Re: Accenture Innovation Challenge 2023   
2023-09-03 07:04:59-04:00 Fwd: Fast-Track Interview Opportunity with Acc...   
2023-05-24 06:45:11-04:00 Fwd: Join the Altair Global Student Contest 2023!   
2023-01-08 02:42:32-05:00 Fwd: Unlock 2023 with Data Science | Resolutio...   
2023-08-17 04:13:10-04:00 Fwd: Internships in association with AICTE   
2023-07-06 21:17:30-04:00 Fwd: Ramco Institute of Technology: Invitation...   
2023-06-03 04:45:20-04:00 Fwd: Internship & Career Opt. in Germany: Appl...   
2023-09-18 07:35:25-04:00 Fwd: URAM Scholarship Program 2023-24: Call Fo...   
2023-03-18 07:04:08-04:00 Fwd: I Semester Exam Fee Payment Status as on ...   
2023-07-20 01:05:39-04:00 Fwd: ICT Academy - HONEYWELL INDIA YOUTH TECKA...   
2023-09-19 06:13:29-04:00 Intern registration   
2023-01-27 06:52:08-05:00 Fwd: AIDS DUE ON 27.1.23   
2023-01-19 05:14:54-05:00 Fwd: Get Placed with Job-a-Thon!   
2023-03-26 09:02:07-04:00 Fwd: EXAM FEE RECEIVED DETAILS 24.03.2023   
2023-02-23 21:39:17-05:00 Fwd: G20 presidency of India - Institute level...   
2023-08-10 23:55:13-04:00 II semester Training with Internship Willingne...   
2023-09-25 00:14:03-04:00 Re: EY - Contest - Hiring - 2027,2026,2025 & 2...   
2023-08-08 07:27:32-04:00 Fwd: ICT Academy - launches the 6th season of ...   
  
 from label thread \  
date   
2023-09-05 07:53:28-04:00 kaliappan@ritrjpm.ac.in sent 1773346693505257505   
2023-08-04 20:26:01-04:00 kaliappan@ritrjpm.ac.in sent 1773346693505257505   
2023-09-03 07:04:59-04:00 kaliappan@ritrjpm.ac.in sent 1776014185824504634   
2023-05-24 06:45:11-04:00 kaliappan@ritrjpm.ac.in sent 1766772073169530495   
2023-01-08 02:42:32-05:00 kaliappan@ritrjpm.ac.in sent 1754439383498542821   
2023-08-17 04:13:10-04:00 kaliappan@ritrjpm.ac.in sent 1774463225953813662   
2023-07-06 21:17:30-04:00 kaliappan@ritrjpm.ac.in sent 1770722623774375044   
2023-06-03 04:45:20-04:00 kaliappan@ritrjpm.ac.in sent 1767670498074944914   
2023-09-18 07:35:25-04:00 kaliappan@ritrjpm.ac.in sent 1777375075198988525   
2023-03-18 07:04:08-04:00 kaliappan@ritrjpm.ac.in sent 1760703264081860447   
2023-07-20 01:05:39-04:00 kaliappan@ritrjpm.ac.in sent 1771914735275306869   
2023-09-19 06:13:29-04:00 kaliappan@ritrjpm.ac.in sent 1777460495498808036   
2023-01-27 06:52:08-05:00 kaliappan@ritrjpm.ac.in sent 1756176430463975696   
2023-01-19 05:14:54-05:00 kaliappan@ritrjpm.ac.in sent 1755445521195468040   
2023-03-26 09:02:07-04:00 kaliappan@ritrjpm.ac.in sent 1761435465303619964   
2023-02-23 21:39:17-05:00 kaliappan@ritrjpm.ac.in sent 1758678373212873483   
2023-08-10 23:55:13-04:00 kaliappan@ritrjpm.ac.in sent 1773903402288280671   
2023-09-25 00:14:03-04:00 kaliappan@ritrjpm.ac.in sent 1777981034069119798   
2023-08-08 07:27:32-04:00 kaliappan@ritrjpm.ac.in sent 1773660081591867249   
  
 Cc \  
date   
2023-09-05 07:53:28-04:00 Jothilakshmi S <jothilakshmi@ritrjpm.ac.in>, \...   
2023-08-04 20:26:01-04:00 SelvaBirunda RIT <selvabirunda@ritrjpm.ac.in>,...   
2023-09-03 07:04:59-04:00 NaN   
2023-05-24 06:45:11-04:00 NaN   
2023-01-08 02:42:32-05:00 NaN   
2023-08-17 04:13:10-04:00 NaN   
2023-07-06 21:17:30-04:00 NaN   
2023-06-03 04:45:20-04:00 NaN   
2023-09-18 07:35:25-04:00 NaN   
2023-03-18 07:04:08-04:00 vimal s <vimal@ritrjpm.ac.in>   
2023-07-20 01:05:39-04:00 NaN   
2023-09-19 06:13:29-04:00 NaN   
2023-01-27 06:52:08-05:00 NaN   
2023-01-19 05:14:54-05:00 NaN   
2023-03-26 09:02:07-04:00 NaN   
2023-02-23 21:39:17-05:00 NaN   
2023-08-10 23:55:13-04:00 NaN   
2023-09-25 00:14:03-04:00 ad2021@ritrjpm.ac.in, ad2022a@ritrjpm.ac.in, a...   
2023-08-08 07:27:32-04:00 NaN   
  
 dayofweek timeofday hour year\_int year   
date   
2023-09-05 07:53:28-04:00 Tuesday 7.891111 7 2023 2023.678987   
2023-08-04 20:26:01-04:00 Friday 20.433611 20 2023 2023.591376   
2023-09-03 07:04:59-04:00 Sunday 7.083056 7 2023 2023.673511   
2023-05-24 06:45:11-04:00 Wednesday 6.753056 6 2023 2023.394251   
2023-01-08 02:42:32-05:00 Sunday 2.708889 2 2023 2023.021903   
2023-08-17 04:13:10-04:00 Thursday 4.219444 4 2023 2023.626968   
2023-07-06 21:17:30-04:00 Thursday 21.291667 21 2023 2023.511978   
2023-06-03 04:45:20-04:00 Saturday 4.755556 4 2023 2023.421629   
2023-09-18 07:35:25-04:00 Monday 7.590278 7 2023 2023.714579   
2023-03-18 07:04:08-04:00 Saturday 7.068889 7 2023 2023.210815   
2023-07-20 01:05:39-04:00 Thursday 1.094167 1 2023 2023.550308   
2023-09-19 06:13:29-04:00 Tuesday 6.224722 6 2023 2023.717317   
2023-01-27 06:52:08-05:00 Friday 6.868889 6 2023 2023.073922   
2023-01-19 05:14:54-05:00 Thursday 5.248333 5 2023 2023.052019   
2023-03-26 09:02:07-04:00 Sunday 9.035278 9 2023 2023.232717   
2023-02-23 21:39:17-05:00 Thursday 21.654722 21 2023 2023.147844   
2023-08-10 23:55:13-04:00 Thursday 23.920278 23 2023 2023.607803   
2023-09-25 00:14:03-04:00 Monday 0.234167 0 2023 2023.733744   
2023-08-08 07:27:32-04:00 Tuesday 7.458889 7 2023 2023.602327

# How many mail sent in week day  
counts = se.dayofweek.value\_counts(sort=False)  
counts.plot(kind='pie')

<Axes: ylabel='dayofweek'>

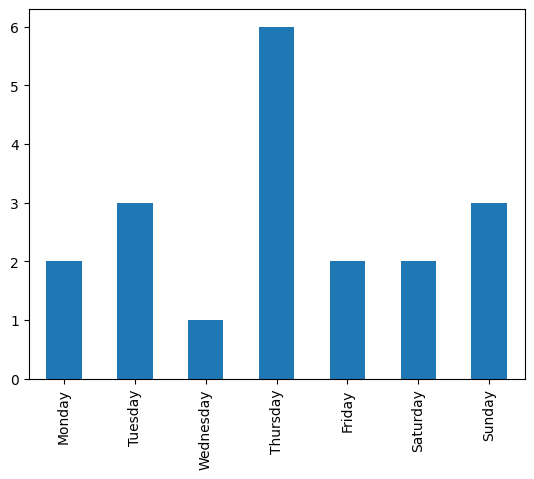


**Observation:**

**The above graph show that how many mail sent in week day from your mail id as in form of pie chart**. **The pie chart shoes that the maximum number of email received and sent from the mail id on Thursday and minmum number of email received and sent from the mail id on wednesday. There has been equal number of email received and sent from the mail id on Friday and Saturday.**

counts = se.dayofweek.value\_counts(sort=False)  
counts.plot(kind='bar')

<Axes: >

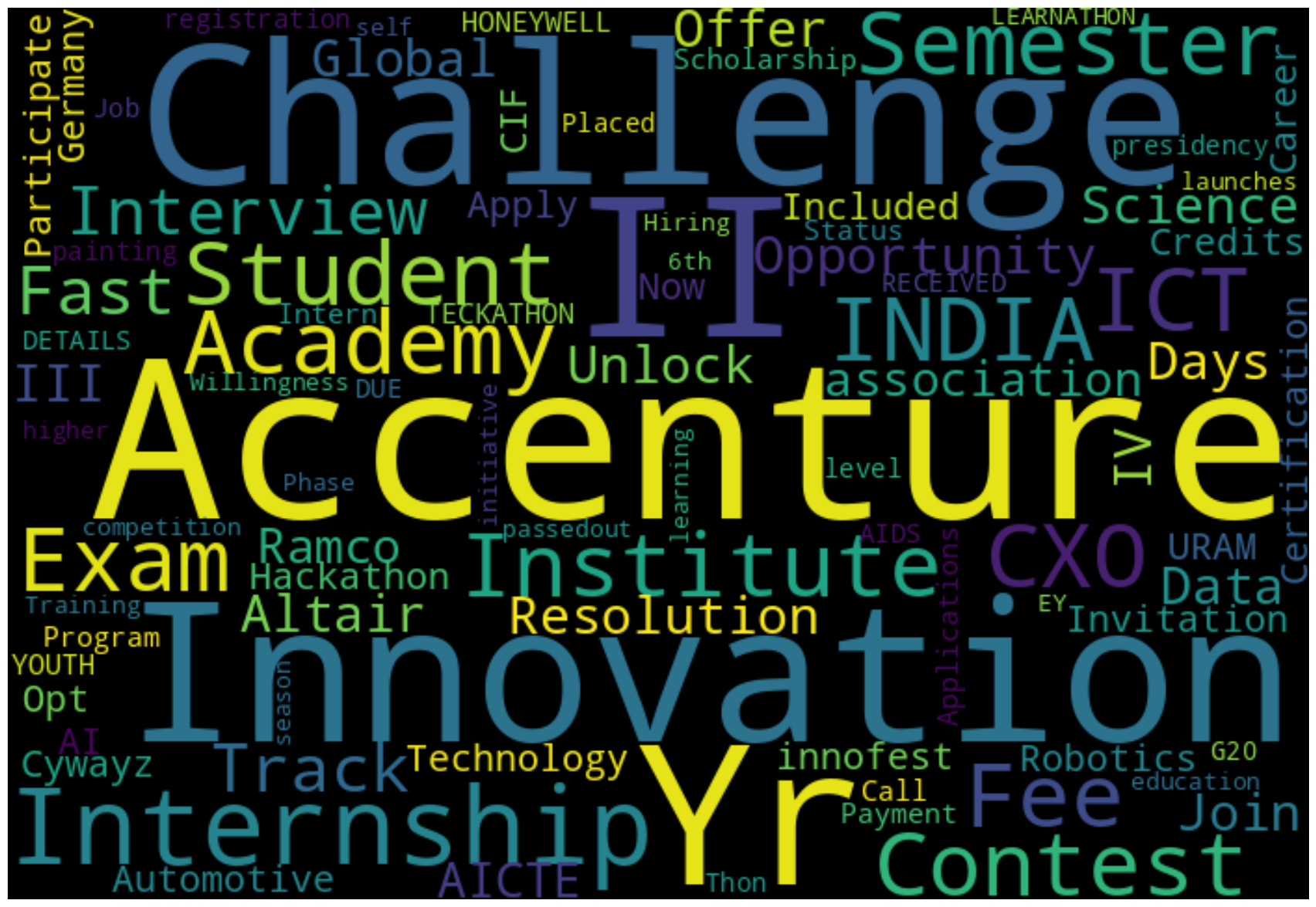


**Observation:**

**From the above graph we can the data that has visulised that has been taken from the dataset. the maximum number of email received and sent from the mail id on Thursday and minmum number of email received and sent from the mail id on wednesday. There has been equal number of email received and sent from the mail id on Friday and Saturday.**

from wordcloud import WordCloud  
df\_no\_arxiv = dfs[dfs['from'] != 'no-reply@arXiv.org']  
text = ' '.join(map(str, sent['subject'].values))  
from wordcloud import WordCloud  
df\_no\_arxiv = dfs[dfs['from'] != 'no-reply@arXiv.org']  
text = ' '.join(map(str, sent['subject'].values))

#plot the word cloud:  
stopwords = ['Re', 'Fwd', '3A\_']  
wrd = WordCloud(width=700, height=480, margin=0, collocations=False)  
for sw in stopwords:  
 wrd.stopwords.add(sw)  
wordcloud = wrd.generate(text)  
plt.figure(figsize=(25,15))  
plt.imshow(wordcloud, interpolation='bilinear')  
plt.axis("off")  
plt.margins(x=0, y=0)



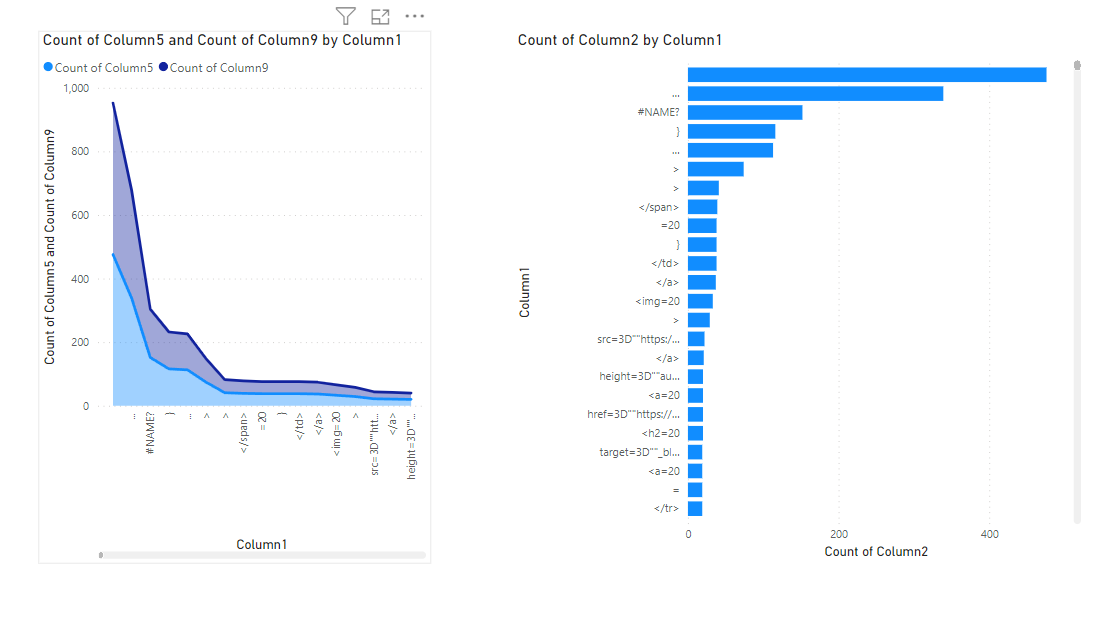
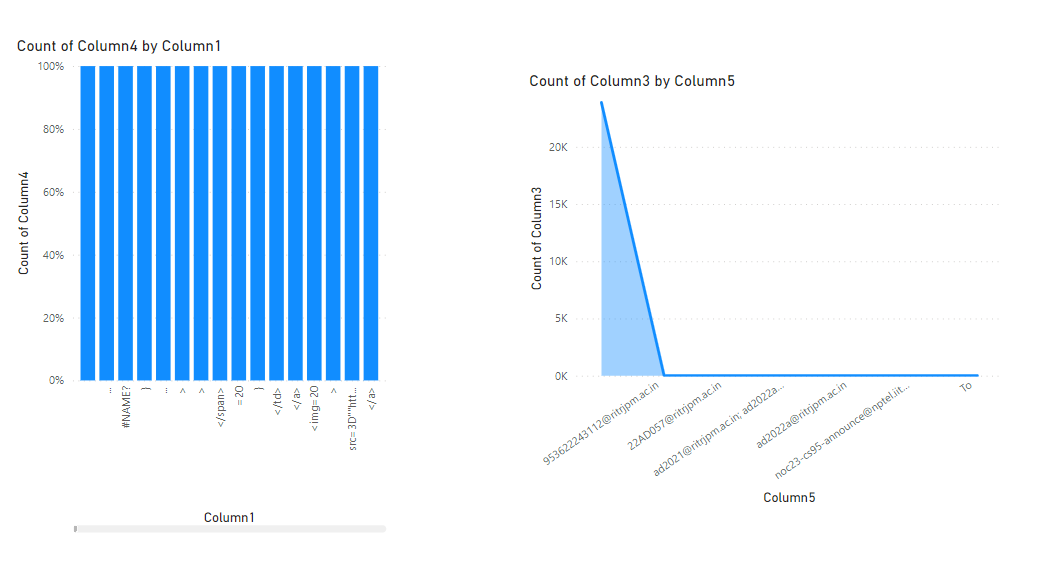
**Observation:**

**The above represent the plotting of word from your received and sent mail from your mail id. The random words that has been showed in the above figure. The words that has been displayed in the figure has different font size and different colour in the above figure.**

**Result:**

**Thus the performing of exploratory data analysis (EDA) on with datasets like email data set is completed successfully.**

**Power BI:**

****