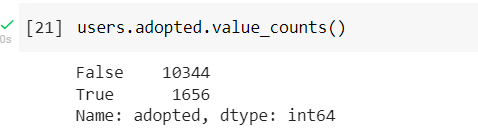
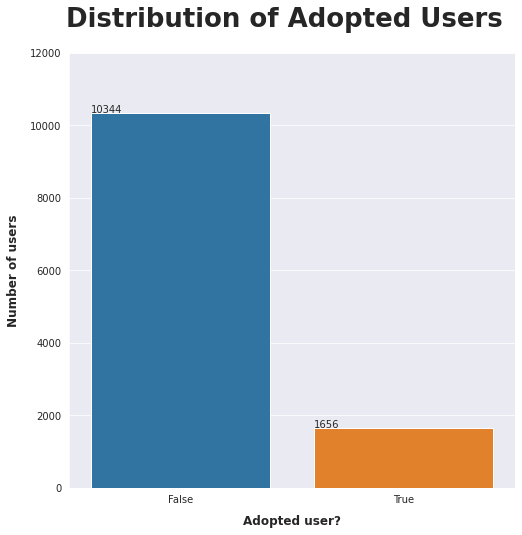
* Given are 12000 users data

# Column Non-Null

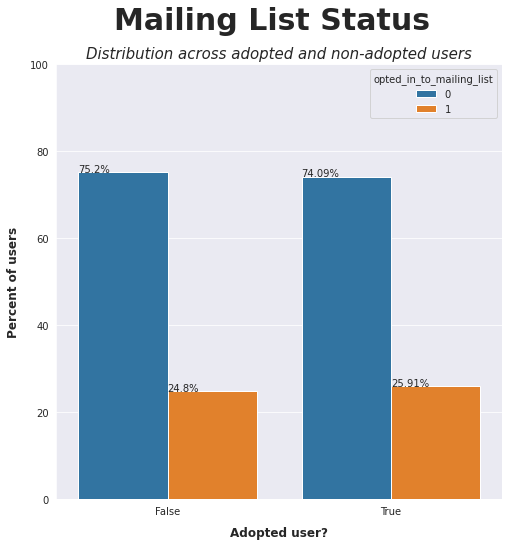
--- ------ --------

0 object id 12000

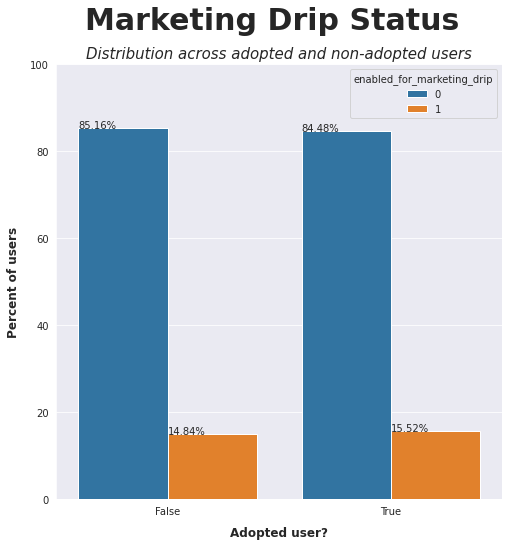
* After working with the provided file of user data I could derive that about 1656 users can been put into the category of adopted users which is users who have visited atleat 3 different days in a 7 days span of time.



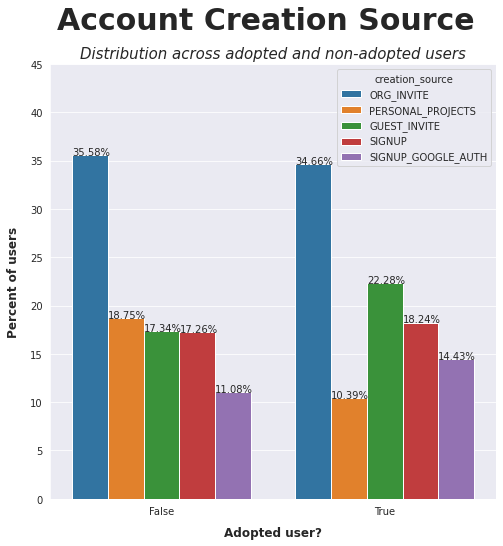
* Next ,I worked with finding the number of adopted and non-adopted users who opted in for mail list.
* I derived that about:
  + 24.8% of non-adopted users have opted for mail list
  + 75.2% of non-adopted users have not opted for mail list
  + 25.91of non-adopted users have opted for mail list
  + 74.09% of non-adopted users have not opted for mail list
* Next, I compared the marketing drip status of users to find out which set of them belonged to adopted users.



* I derived that about:
  + 14.84% of non-adopted users have opted for marketing drip status
  + 15.52% of adopted users have opted for marketing drip status
  + 85.16% of non-adopted users have not opted for marketing drip status
  + 84.48% of adopted users have not opted for marketing drip status



* Next, I have tried to find the account creation source adopted and non-adopted users, and by the following graph I could say that people from organization invites as permanent members and people with guest invites have more chance to be an adopted user and people invited to view personal projects are very likely to be adopted users.



* From the minimal research I have done with the data I could conclude that people who have opted in for mail list and marketing drip and people sourced from organizational invites have higher chances to get converted to adopted users. And further research can be done to find the relation between the above mentioned categories to find more accurate conclusion and also we could need an additional information regarding logout time to find the total login hours to predict more accurate results.
* Code used:

import pandas as pd

import numpy as np

import seaborn as sns

from datetime import datetime

from dateutil import parser

from matplotlib import pyplot as plt

users = pd.read\_csv('takehome\_users.csv',encoding='ISO-8859-1')

user\_engagement = pd.read\_csv('takehome\_user\_engagement.csv')

users.info()

# creating a list of users with more than 3 visits

user\_id\_counts = user\_engagement.user\_id.value\_counts() >= 3

potential\_adopted\_users = [user\_id for user\_id in user\_id\_counts.index if user\_id\_counts[user\_id] == True]

# creating a list of users with more than 3 visits in last 7 days

adopted\_users = []

for user in potential\_adopted\_users:

user\_dates = user\_engagement[user\_engagement.user\_id == user].time\_stamp.tolist()

user\_dates = [parser.parse(date) for date in user\_dates]

queue = user\_dates[0:3]

adopted = False

cur\_index = 2

while adopted == False:

if ((queue[-1] - queue[0]).days <= 7):

adopted = True

else:

try:

cur\_index += 1

queue.pop(0)

queue.append(user\_dates[cur\_index])

except:

break

if adopted == True:

adopted\_users.append(user)

len(adopted\_users)

#creating a column to indicate if user is adopted

def adopted\_function(x, adopted\_users\_list):

if x in adopted\_users\_list:

return True

else:

return False

users['adopted'] = users['object\_id'].apply(lambda x: adopted\_function(x, adopted\_users))

users.adopted.value\_counts()

sns.set\_style('darkgrid')

fig,ax1 = plt.subplots(figsize=(8, 8))

sns.countplot(x=users.adopted, ax=ax1)

# set titles

plt.suptitle('Distribution of Adopted Users', fontsize=26, y=0.96, fontweight='bold')

ax1.set\_ylabel('Number of users', labelpad=10, fontsize=12, fontweight='bold')

ax1.set\_xlabel('Adopted user?', labelpad=10,fontsize=12, fontweight='bold')

ax1.set\_ylim(0,12000) # set ylim

for p in ax1.patches:

txt = str(p.get\_height().round(2))

txt\_x = p.get\_x()

txt\_y = p.get\_height()

ax1.text(txt\_x,txt\_y,txt)

fig,ax1 = plt.subplots(figsize=(8, 8))

x, y = 'adopted', 'opted\_in\_to\_mailing\_list'

# group df to properly show percentage

viz\_df = (users.groupby(x)[y].value\_counts(normalize=True).mul(100).rename('percent').reset\_index())

sns.barplot(x=x, y='percent', hue=y, data=viz\_df, ax=ax1)

# set titles

ax1.set\_ylim(0,100)

plt.suptitle('Mailing List Status', fontsize=30, y=0.98, fontweight='bold')

plt.title('Distribution across adopted and non-adopted users', fontsize=15, style='italic')

ax1.set\_ylabel('Percent of users', labelpad=10, fontsize=12, fontweight='bold')

ax1.set\_xlabel('Adopted user?', labelpad=10,fontsize=12, fontweight='bold')

for p in ax1.patches:

txt = str(p.get\_height().round(2)) + '%'

txt\_x = p.get\_x()

txt\_y = p.get\_height()

ax1.text(txt\_x,txt\_y,txt)

fig,ax1 = plt.subplots(figsize=(8, 8))

x, y = 'adopted', 'enabled\_for\_marketing\_drip'

# group df to properly show percentage

viz\_df = (users.groupby(x)[y].value\_counts(normalize=True).mul(100).rename('percent').reset\_index())

sns.barplot(x=x, y='percent', hue=y, data=viz\_df, ax=ax1)

# set titles

ax1.set\_ylim(0,100)

plt.suptitle('Marketing Drip Status', fontsize=30, y=0.98, fontweight='bold')

plt.title('Distribution across adopted and non-adopted users', fontsize=15, style='italic')

ax1.set\_ylabel('Percent of users', labelpad=10, fontsize=12, fontweight='bold')

ax1.set\_xlabel('Adopted user?', labelpad=10,fontsize=12, fontweight='bold')

for p in ax1.patches:

txt = str(p.get\_height().round(2)) + '%'

txt\_x = p.get\_x()

txt\_y = p.get\_height()

ax1.text(txt\_x,txt\_y,txt)

fig,ax1 = plt.subplots(figsize=(8, 8))

x, y = 'adopted', 'creation\_source'

# group df to properly show percentage

viz\_df = (users.groupby(x)[y].value\_counts(normalize=True).mul(100).rename('percent').reset\_index())

sns.barplot(x=x, y='percent', hue=y, data=viz\_df, ax=ax1)

# set titles

ax1.set\_ylim(0,45)

plt.suptitle('Account Creation Source', fontsize=30, y=0.98, fontweight='bold')

plt.title('Distribution across adopted and non-adopted users',fontsize=15, style='italic')

ax1.set\_ylabel('Percent of users', labelpad=10, fontsize=12, fontweight='bold')

ax1.set\_xlabel('Adopted user?', labelpad=10,fontsize=12, fontweight='bold')

for p in ax1.patches:

txt = str(p.get\_height().round(2)) + '%'

txt\_x = p.get\_x()

txt\_y = p.get\_height()

ax1.text(txt\_x,txt\_y,txt)