In [1]:
```
# importing all libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
import sklearn
%matplotlib inline
```

In [2]:
```
#importing  data into jupyter environment
data=pd.read_excel('D:\\MBA RAJESH(D)\\MBA SEM 3\\GE 372 PREDICTIVE ANALYTICS\\python\\end sem exam\\practice\\indian t
```

In [3]:
```
#checking imported data
data.head()
```

Out[3]:

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	Total intl minutes
0	KS	128	415	No	Yes	25	265.1	110	45.07	197.4	99	16.78	244.7	91	11.01	10.0
1	OH	107	415	No	Yes	26	161.6	123	27.47	195.5	103	16.62	254.4	103	11.45	13.7
2	NJ	137	415	No	No	0	243.4	114	41.38	121.2	110	10.30	162.6	104	7.32	12.2
3	OH	84	408	Yes	No	0	299.4	71	50.90	61.9	88	5.26	196.9	89	8.86	6.6
4	OK	75	415	Yes	No	0	166.7	113	28.34	148.3	122	12.61	186.9	121	8.41	10.1

In [4]:
```
data.tail()
```

Out[4]:

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	To minu
3328	WI	114	415	No	Yes	26	137.1	88	23.31	155.7	125	13.23	247.6	94	11.14	1
3329	AL	106	408	No	Yes	29	83.6	131	14.21	203.9	131	17.33	229.5	73	10.33	
3330	VT	60	415	No	No	0	193.9	118	32.96	85.0	110	7.23	210.1	134	9.45	1
3331	WV	159	415	No	No	0	169.8	114	28.87	197.7	105	16.80	193.7	82	8.72	1
3332	CT	184	510	Yes	No	0	213.8	105	36.35	159.6	84	13.57	139.2	137	6.26	

In [5]:
```
data.sample(10)
```

Out[5]:

	State	Account length	Area code	International plan	Voice mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	To minu
2011	MN	79	408	No	Yes	32	50.6	62	8.60	201.4	87	17.12	146.8	121	6.61	
2602	OH	88	408	No	No	0	274.6	105	46.68	161.1	121	13.69	194.4	123	8.75	
3236	UT	44	415	No	No	0	202.6	89	34.44	163.0	96	13.86	268.1	151	12.06	
2568	WA	143	510	No	No	0	160.4	120	27.27	285.9	104	24.30	182.5	85	8.21	
31	MA	78	415	No	No	0	130.8	64	22.24	223.7	116	19.01	227.8	108	10.25	1
2685	MI	36	510	No	Yes	29	281.4	102	47.84	202.2	76	17.19	187.2	113	8.42	
1922	WV	64	415	No	Yes	22	200.4	80	34.07	131.1	84	11.14	230.7	67	10.38	
1372	WA	130	510	No	No	0	124.3	70	21.13	270.7	99	23.01	239.5	83	10.78	
2809	ND	123	408	No	No	0	159.1	94	27.05	241.6	119	20.54	202.4	120	9.11	
1274	VT	57	415	No	No	0	169.6	96	28.83	234.7	112	19.95	285.4	83	12.84	1

In [6]:
```
data.dtypes
```

Out[6]:
```
State                     object
Account length             int64
Area code                  int64
International plan         object
Voice mail plan            object
Number vmail messages      int64
Total day minutes         float64
Total day calls            int64
Total day charge          float64
Total eve minutes         float64
Total eve calls            int64
Total eve charge          float64
Total night minutes       float64
Total night calls          int64
Total night charge        float64
Total intl minutes        float64
Total intl calls           int64
Total intl charge         float64
Customer service calls     int64
Churn                       bool
dtype: object
```

In [7]:
```
data.shape
```

Out[7]:
```
(3333, 20)
```

In [8]:
```
#getting data types and missing values
data.info()
## seems no null values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3333 entries, 0 to 3332
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   State                   3333 non-null   object
 1   Account length          3333 non-null   int64
 2   Area code               3333 non-null   int64
 3   International plan      3333 non-null   object
 4   Voice mail plan         3333 non-null   object
 5   Number vmail messages   3333 non-null   int64
 6   Total day minutes       3333 non-null   float64
 7   Total day calls         3333 non-null   int64
 8   Total day charge        3333 non-null   float64
 9   Total eve minutes       3333 non-null   float64
 10  Total eve calls         3333 non-null   int64
 11  Total eve charge        3333 non-null   float64
 12  Total night minutes     3333 non-null   float64
 13  Total night calls       3333 non-null   int64
 14  Total night charge      3333 non-null   float64
 15  Total intl minutes      3333 non-null   float64
 16  Total intl calls        3333 non-null   int64
 17  Total intl charge       3333 non-null   float64
 18  Customer service calls  3333 non-null   int64
 19  Churn                   3333 non-null   bool
dtypes: bool(1), float64(8), int64(8), object(3)
memory usage: 498.1+ KB
```

In [9]:

```python
# conforming for null values
data.isnull().sum()
## no null values confirmed
```

Out[9]:

```
State                     0
Account length            0
Area code                 0
International plan         0
Voice mail plan            0
Number vmail messages      0
Total day minutes          0
Total day calls            0
Total day charge           0
Total eve minutes          0
Total eve calls            0
Total eve charge           0
Total night minutes        0
Total night calls          0
Total night charge         0
Total intl minutes         0
Total intl calls           0
Total intl charge          0
Customer service calls     0
Churn                      0
dtype: int64
```

In [10]:

```
# checking for duplicates
data.duplicated().sum()
## no duplicate values in the data
```

Out[10]:

```
0
```

In [11]:

```
# examining object columns
data.select_dtypes(include=['object','bool'])
## these columns require transformations
```

Out[11]:

	State	International plan	Voice mail plan	Churn
0	KS	No	Yes	False
1	OH	No	Yes	False
2	NJ	No	No	False
3	OH	Yes	No	False
4	OK	Yes	No	False
...	...	...	...	...
3328	WI	No	Yes	False
3329	AL	No	Yes	False
3330	VT	No	No	False
3331	WV	No	No	False
3332	CT	Yes	No	False

3333 rows × 4 columns

In [12]:

```
# examiniing numerric columns
data.select_dtypes(include=['int64','float64'])
```

Out[12]:

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	Total intl minutes	Total intl calls	Total intl charge	Custome servic call
0	128	415	25	265.1	110	45.07	197.4	99	16.78	244.7	91	11.01	10.0	3	2.70	
1	107	415	26	161.6	123	27.47	195.5	103	16.62	254.4	103	11.45	13.7	3	3.70	
2	137	415	0	243.4	114	41.38	121.2	110	10.30	162.6	104	7.32	12.2	5	3.29	
3	84	408	0	299.4	71	50.90	61.9	88	5.26	196.9	89	8.86	6.6	7	1.78	
4	75	415	0	166.7	113	28.34	148.3	122	12.61	186.9	121	8.41	10.1	3	2.73	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
3328	114	415	26	137.1	88	23.31	155.7	125	13.23	247.6	94	11.14	11.5	7	3.11	
3329	106	408	29	83.6	131	14.21	203.9	131	17.33	229.5	73	10.33	8.1	3	2.19	
3330	60	415	0	193.9	118	32.96	85.0	110	7.23	210.1	134	9.45	13.2	8	3.56	
3331	159	415	0	169.8	114	28.87	197.7	105	16.80	193.7	82	8.72	11.6	4	3.13	
3332	184	510	0	213.8	105	36.35	159.6	84	13.57	139.2	137	6.26	5.0	10	1.35	

3333 rows × 16 columns

In [13]:

```
#getting descriptive statistics
data.describe()
## seems all columns to be following normal distribution
```

Out[13]:

	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	
count	3333.000000	3333.000000	3333.000000	3333.000000	3333.000000	3333.000000	3333.000000	3333.000000	3333.000000	3333.000000	3
mean	101.064806	437.182418	8.099010	179.775098	100.435644	30.562307	200.980348	100.114311	17.083540	200.872037	
std	39.822106	42.371290	13.688365	54.467389	20.069084	9.259435	50.713844	19.922625	4.310668	50.573847	
min	1.000000	408.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	23.200000	
25%	74.000000	408.000000	0.000000	143.700000	87.000000	24.430000	166.600000	87.000000	14.160000	167.000000	
50%	101.000000	415.000000	0.000000	179.400000	101.000000	30.500000	201.400000	100.000000	17.120000	201.200000	
75%	127.000000	510.000000	20.000000	216.400000	114.000000	36.790000	235.300000	114.000000	20.000000	235.300000	
max	243.000000	510.000000	51.000000	350.800000	165.000000	59.640000	363.700000	170.000000	30.910000	395.000000	

In [14]:

```
# getting skewness to conform normal distribution
data.skew(numeric_only=True)
## seems number vmail messages, customer service calls not following normal disribution
```

Out[14]:

```
Account length            0.096606
Area code                 1.126823
Number vmail messages     1.264824
Total day minutes        -0.029077
Total day calls          -0.111787
Total day charge         -0.029083
Total eve minutes        -0.023877
Total eve calls          -0.055563
Total eve charge         -0.023858
Total night minutes       0.008921
Total night calls         0.032500
Total night charge        0.008886
Total intl minutes       -0.245136
Total intl calls          1.321478
Total intl charge        -0.245287
Customer service calls    1.091359
Churn                     2.018356
dtype: float64
```

In [15]:

```
## getting kurtosis
```

```
data.kurtosis()
## seems customer service calls haing some outliers
```

```
C:\Users\G RAJESH\AppData\Local\Temp\ipykernel_17744\1271178775.py:2: FutureWarning: Dropping of nuisance columns in Da
taFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select onl
y valid columns before calling the reduction.
  data.kurtosis()
```

Out[15]:

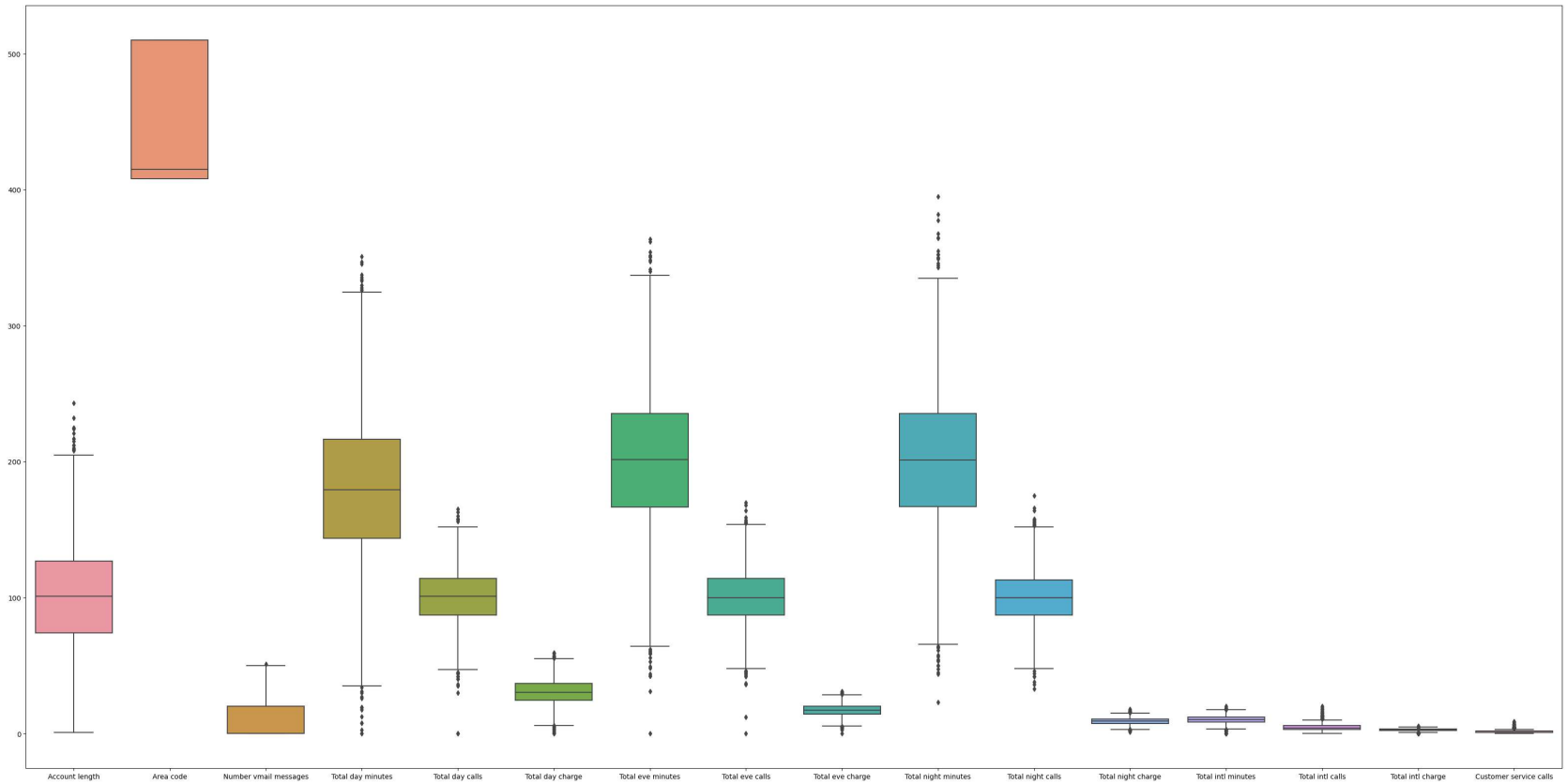
```
Account length           -0.107836
Area code                -0.705632
Number vmail messages    -0.051129
Total day minutes        -0.019940
Total day calls           0.243182
Total day charge         -0.019812
Total eve minutes         0.025630
Total eve calls           0.206156
Total eve charge          0.025487
Total night minutes       0.085816
Total night calls        -0.072020
Total night charge        0.085663
Total intl minutes        0.609185
Total intl calls          3.083589
Total intl charge         0.609610
Customer service calls    1.730914
Churn                     2.075006
dtype: float64
```

In [16]:

```
# getting boxplot to identify outliers
plt.figure(num=1,
          figsize=(40,20))
sb.boxplot(data=data.select_dtypes(['int64','float64']))
## outliers in various columns detected,
## observed that all these columns are in same scale
## So, normalization is requilred before outlier removal
```

Out[16]:

```
<AxesSubplot:>
```



normalizaing only numeric data

In [17]:
```
# importing normalizer
from sklearn.preprocessing import StandardScaler
```

In [18]:
```
#creating object for Standard Scaler
scaler=StandardScaler()
```

In [19]:
```
#fitting datta to scaler object
scaler.fit(data.select_dtypes(include=['int64','float64']))
```

Out[19]:
```
StandardScaler()
```

In [20]:
```
# normalizing data
norm_data_numeric=pd.DataFrame(scaler.transform(data.select_dtypes(include=['int64','float64'])))
```

```
norm_data_numeric
```

Out[20]:

	0	1	2	3	4	5	6	7	8	9	10	11	
0	0.676489	-0.523603	1.234883	1.566767	0.476643	1.567036	-0.070610	-0.055940	-0.070427	0.866743	-0.465494	0.866029	-0.085
1	0.149065	-0.523603	1.307948	-0.333738	1.124503	-0.334013	-0.108080	0.144867	-0.107549	1.058571	0.147825	1.059390	1.240
2	0.902529	-0.523603	-0.591760	1.168304	0.675985	1.168464	-1.573383	0.496279	-1.573900	-0.756869	0.198935	-0.755571	0.703
3	-0.428590	-0.688834	-0.591760	2.196596	-1.466936	2.196759	-2.742865	-0.608159	-2.743268	-0.078551	-0.567714	-0.078806	-1.303
4	-0.654629	-0.523603	-0.591760	-0.240090	0.626149	-0.240041	-1.038932	1.098699	-1.037939	-0.276311	1.067803	-0.276562	-0.049
...	...	...	...	...	...	...	...	...	...	...	...	...	
3328	0.324873	-0.523603	1.307948	-0.783616	-0.619735	-0.783352	-0.892994	1.249304	-0.894089	0.924094	-0.312165	0.923158	0.452
3329	0.123950	-0.688834	1.527146	-1.766002	1.523186	-1.766281	0.057580	1.550515	0.057183	0.566148	-1.385473	0.567197	-0.765
3330	-1.031361	-0.523603	-0.591760	0.259367	0.875326	0.258985	-2.287299	0.496279	-2.286193	0.182492	1.732232	0.180474	1.061
3331	1.455068	-0.523603	-0.591760	-0.183166	0.675985	-0.182793	-0.064693	0.245270	-0.065786	-0.141834	-0.925484	-0.140330	0.488
3332	2.082955	1.718817	-0.591760	0.624778	0.227466	0.625153	-0.816080	-0.808966	-0.815203	-1.219628	1.885562	-1.221396	-1.876

3333 rows × 16 columns

In [21]:

```
data.select_dtypes(include=['int64','float64']).columns
```

Out[21]:

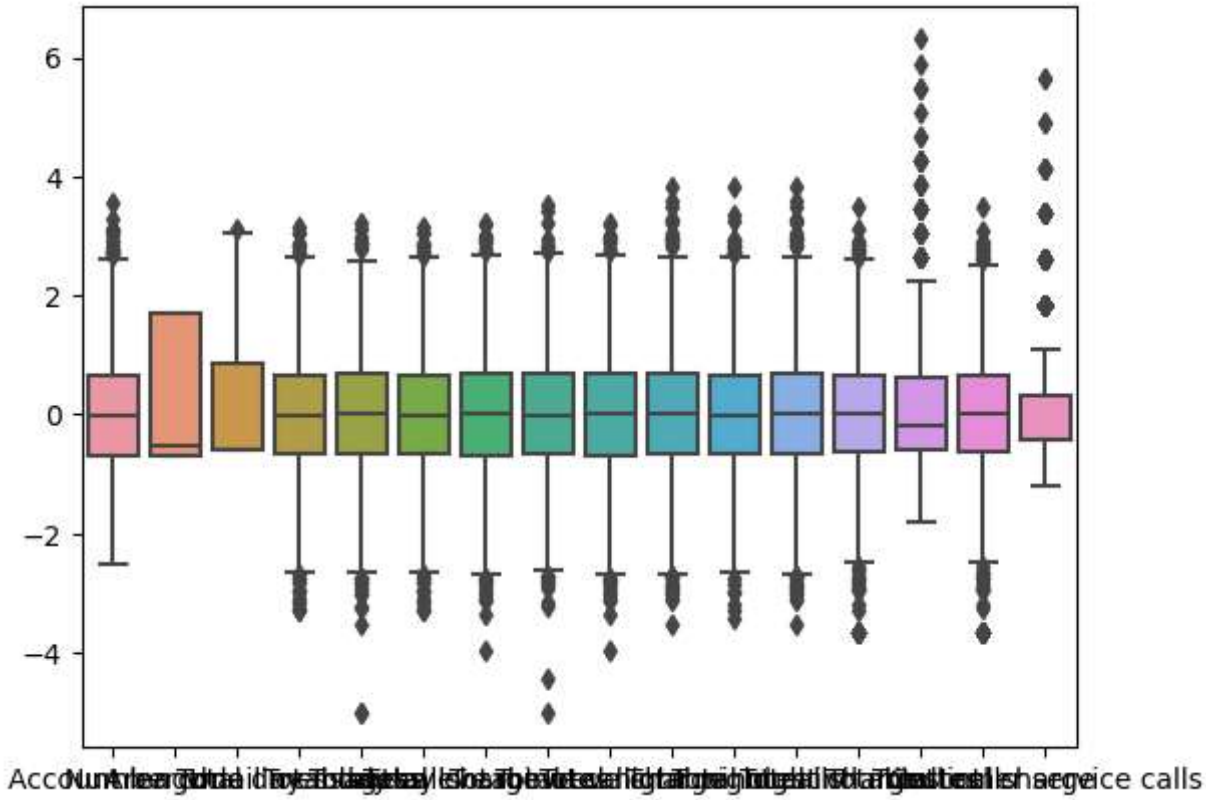
```
Index(['Account length', 'Area code', 'Number vmail messages',
       'Total day minutes', 'Total day calls', 'Total day charge',
       'Total eve minutes', 'Total eve calls', 'Total eve charge',
       'Total night minutes', 'Total night calls', 'Total night charge',
       'Total intl minutes', 'Total intl calls', 'Total intl charge',
       'Customer service calls'],
      dtype='object')
```

In [22]:

```
norm_data_numeric.columns=['Account length', 'Area code', 'Number vmail messages',
       'Total day minutes', 'Total day calls', 'Total day charge',
       'Total eve minutes', 'Total eve calls', 'Total eve charge',
       'Total night minutes', 'Total night calls', 'Total night charge',
       'Total intl minutes', 'Total intl calls', 'Total intl charge',
       'Customer service calls']
```

In [23]:
```
sb.boxplot(data=norm_data_numeric)
```

Out[23]:
```
<AxesSubplot:>
```



In [24]:
```
data_cat=data.select_dtypes(include=['object','bool'])
data_cat
```

Out[24]:

	State	International plan	Voice mail plan	Churn
0	KS	No	Yes	False
1	OH	No	Yes	False
2	NJ	No	No	False
3	OH	Yes	No	False
4	OK	Yes	No	False
...	...	...	...	...
3328	WI	No	Yes	False
3329	AL	No	Yes	False
3330	VT	No	No	False
3331	WV	No	No	False
3332	CT	Yes	No	False

3333 rows × 4 columns

In [25]:

```
data_to_be_preprocessed=pd.concat([data_cat,norm_data_numeric],
                                  axis=1)
data_to_be_preprocessed
```

Out[25]:

	State	International plan	Voice mail plan	Churn	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	
0	KS	No	Yes	False	0.676489	-0.523603	1.234883	1.566767	0.476643	1.567036	-0.070610	-0.055940	-0.070427	0
1	OH	No	Yes	False	0.149065	-0.523603	1.307948	-0.333738	1.124503	-0.334013	-0.108080	0.144867	-0.107549	1
2	NJ	No	No	False	0.902529	-0.523603	-0.591760	1.168304	0.675985	1.168464	-1.573383	0.496279	-1.573900	-0
3	OH	Yes	No	False	-0.428590	-0.688834	-0.591760	2.196596	-1.466936	2.196759	-2.742865	-0.608159	-2.743268	-0
4	OK	Yes	No	False	-0.654629	-0.523603	-0.591760	-0.240090	0.626149	-0.240041	-1.038932	1.098699	-1.037939	-0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
3328	WI	No	Yes	False	0.324873	-0.523603	1.307948	-0.783616	-0.619735	-0.783352	-0.892994	1.249304	-0.894089	0
3329	AL	No	Yes	False	0.123950	-0.688834	1.527146	-1.766002	1.523186	-1.766281	0.057580	1.550515	0.057183	0
3330	VT	No	No	False	-1.031361	-0.523603	-0.591760	0.259367	0.875326	0.258985	-2.287299	0.496279	-2.286193	0
3331	WV	No	No	False	1.455068	-0.523603	-0.591760	-0.183166	0.675985	-0.182793	-0.064693	0.245270	-0.065786	-0
3332	CT	Yes	No	False	2.082955	1.718817	-0.591760	0.624778	0.227466	0.625153	-0.816080	-0.808966	-0.815203	-1

3333 rows × 20 columns

In [26]:

```
data_to_be_preprocessed['cust leaves or not ']=data_to_be_preprocessed['Churn']
data_to_be_preprocessed
```

Out[26]:

	State	International plan	Voice mail plan	Churn	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	...	Total eve calls	Total eve charge	Total night minutes
0	KS	No	Yes	False	0.676489	-0.523603	1.234883	1.566767	0.476643	1.567036	...	-0.055940	-0.070427	0.866743
1	OH	No	Yes	False	0.149065	-0.523603	1.307948	-0.333738	1.124503	-0.334013	...	0.144867	-0.107549	1.058571
2	NJ	No	No	False	0.902529	-0.523603	-0.591760	1.168304	0.675985	1.168464	...	0.496279	-1.573900	-0.756869
3	OH	Yes	No	False	-0.428590	-0.688834	-0.591760	2.196596	-1.466936	2.196759	...	-0.608159	-2.743268	-0.078551
4	OK	Yes	No	False	-0.654629	-0.523603	-0.591760	-0.240090	0.626149	-0.240041	...	1.098699	-1.037939	-0.276311
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3328	WI	No	Yes	False	0.324873	-0.523603	1.307948	-0.783616	-0.619735	-0.783352	...	1.249304	-0.894089	0.924094
3329	AL	No	Yes	False	0.123950	-0.688834	1.527146	-1.766002	1.523186	-1.766281	...	1.550515	0.057183	0.566148
3330	VT	No	No	False	-1.031361	-0.523603	-0.591760	0.259367	0.875326	0.258985	...	0.496279	-2.286193	0.182492
3331	WV	No	No	False	1.455068	-0.523603	-0.591760	-0.183166	0.675985	-0.182793	...	0.245270	-0.065786	-0.141834
3332	CT	Yes	No	False	2.082955	1.718817	-0.591760	0.624778	0.227466	0.625153	...	-0.808966	-0.815203	-1.219628

3333 rows × 21 columns

In [27]:

```
#deleting Churn column
data_to_be_preprocessed.drop(columns='Churn',
                             inplace=True)
data_to_be_preprocessed
```

Out[27]:

	State	International plan	Voice mail plan	Account length	Area code	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes
0	KS	No	Yes	0.676489	-0.523603	1.234883	1.566767	0.476643	1.567036	-0.070610	-0.055940	-0.070427	0.866743
1	OH	No	Yes	0.149065	-0.523603	1.307948	-0.333738	1.124503	-0.334013	-0.108080	0.144867	-0.107549	1.058571
2	NJ	No	No	0.902529	-0.523603	-0.591760	1.168304	0.675985	1.168464	-1.573383	0.496279	-1.573900	-0.756869
3	OH	Yes	No	-0.428590	-0.688834	-0.591760	2.196596	-1.466936	2.196759	-2.742865	-0.608159	-2.743268	-0.078551
4	OK	Yes	No	-0.654629	-0.523603	-0.591760	-0.240090	0.626149	-0.240041	-1.038932	1.098699	-1.037939	-0.276311
...	...	...	...	...	...	...	...	...	...	...	...	...	...
3328	WI	No	Yes	0.324873	-0.523603	1.307948	-0.783616	-0.619735	-0.783352	-0.892994	1.249304	-0.894089	0.924094
3329	AL	No	Yes	0.123950	-0.688834	1.527146	-1.766002	1.523186	-1.766281	0.057580	1.550515	0.057183	0.566148
3330	VT	No	No	-1.031361	-0.523603	-0.591760	0.259367	0.875326	0.258985	-2.287299	0.496279	-2.286193	0.182492
3331	WV	No	No	1.455068	-0.523603	-0.591760	-0.183166	0.675985	-0.182793	-0.064693	0.245270	-0.065786	-0.141834
3332	CT	Yes	No	2.082955	1.718817	-0.591760	0.624778	0.227466	0.625153	-0.816080	-0.808966	-0.815203	-1.219628

3333 rows × 20 columns

In [28]:
```
# exporting above data into excel for preprocessing and outlier removal
data_to_be_preprocessed.to_excel('D:\\MBA RAJESH(D)\\MBA SEM 3\\GE 372 PREDICTIVE ANALYTICS\\python\\end sem exam\\indi
```

importing preprocessed data

In [29]:
```
data_final=pd.read_excel('D:\\MBA RAJESH(D)\\MBA SEM 3\\GE 372 PREDICTIVE ANALYTICS\\python\\end sem exam\\practice\\in
```

In [30]:
```
data_final
```

Out[30]:

	Int. plan	vocie. mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	Total intl minutes	Tc
0	0	1	1.234883	1.566767	0.476643	1.567036	-0.070610	-0.055940	-0.070427	0.866743	-0.465494	0.866029	-0.085008	-0.
1	0	1	1.307948	-0.333738	1.124503	-0.334013	-0.108080	0.144867	-0.107549	1.058571	0.147825	1.059390	1.240482	-0.
2	0	0	-0.591760	1.168304	0.675985	1.168464	-1.573383	0.496279	-1.573900	-0.756869	0.198935	-0.755571	0.703121	0.
3	1	0	-0.591760	0.801056	-0.121381	0.801216	0.386928	0.044463	0.386648	0.059881	0.914473	0.061821	-1.410498	0.
4	1	0	-0.591760	-0.418205	-1.068253	-0.418264	-1.930341	-0.306949	-1.931206	0.216112	-0.209945	0.215631	-1.123906	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2337	0	0	-0.591760	0.441154	-0.769241	0.441529	-2.232079	0.496279	-2.232829	-0.685675	1.118913	-0.685257	0.524001	-0.
2338	0	0	-0.591760	1.840366	0.626149	1.840312	0.024053	-0.507756	0.024700	-0.871570	0.147825	-0.869830	0.058288	-0.
2339	0	1	1.307948	-0.783616	-0.619735	-0.783352	-0.892994	1.249304	-0.894089	0.924094	-0.312165	0.923158	0.452353	1.
2340	0	1	1.527146	-1.766002	1.523186	-1.766281	0.057580	1.550515	0.057183	0.566148	-1.385473	0.567197	-0.765665	-0.
2341	0	0	-0.591760	-0.183166	0.675985	-0.182793	-0.064693	0.245270	-0.065786	-0.141834	-0.925484	-0.140330	0.488177	-0.

2342 rows × 17 columns

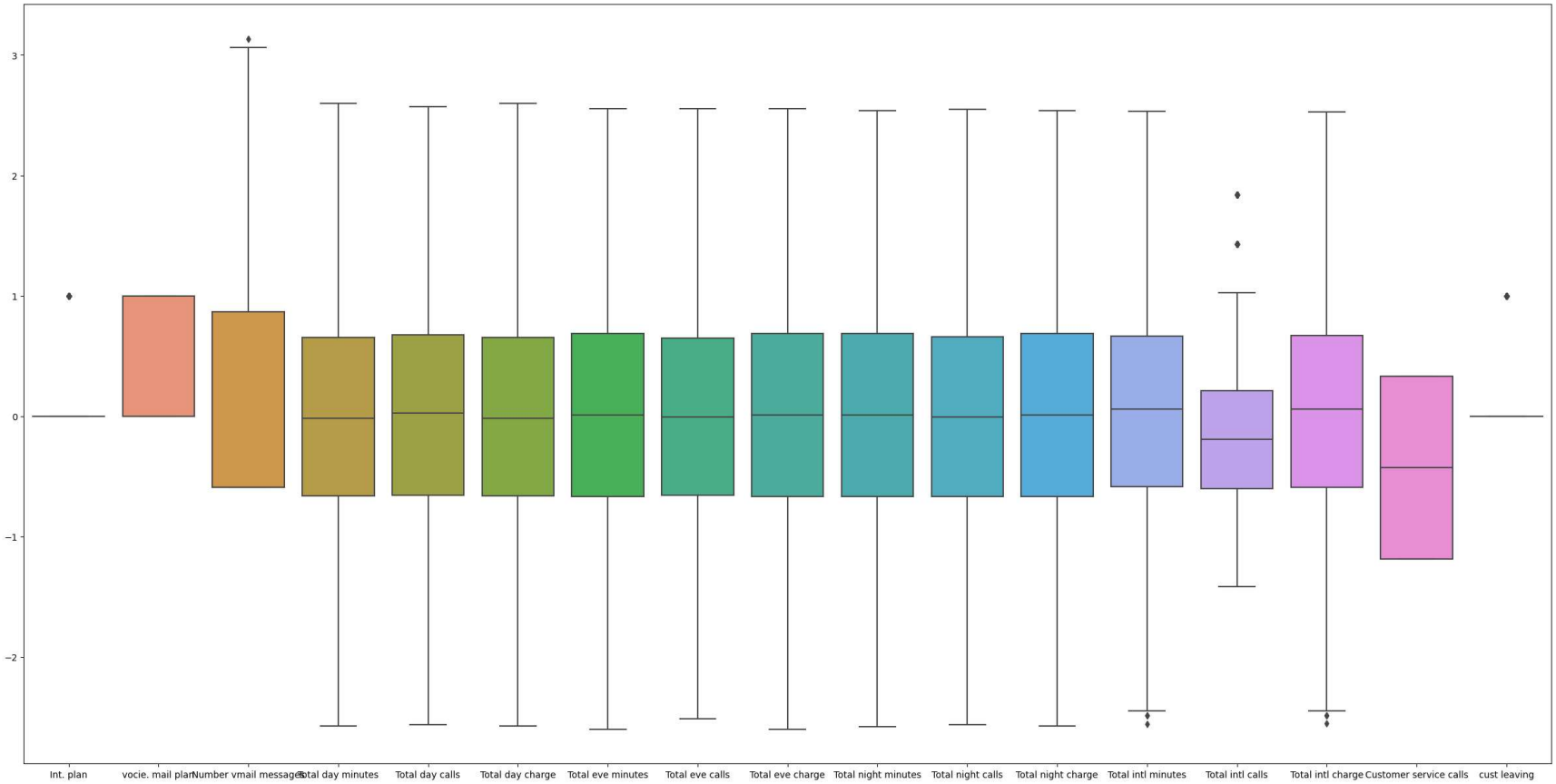
after preprocessing data is reduced to 2342 rows

In [31]:

```
# confirming outlier removal
plt.figure(figsize=(30,15))
sb.boxplot(data=data_final)
```

Out[31]:

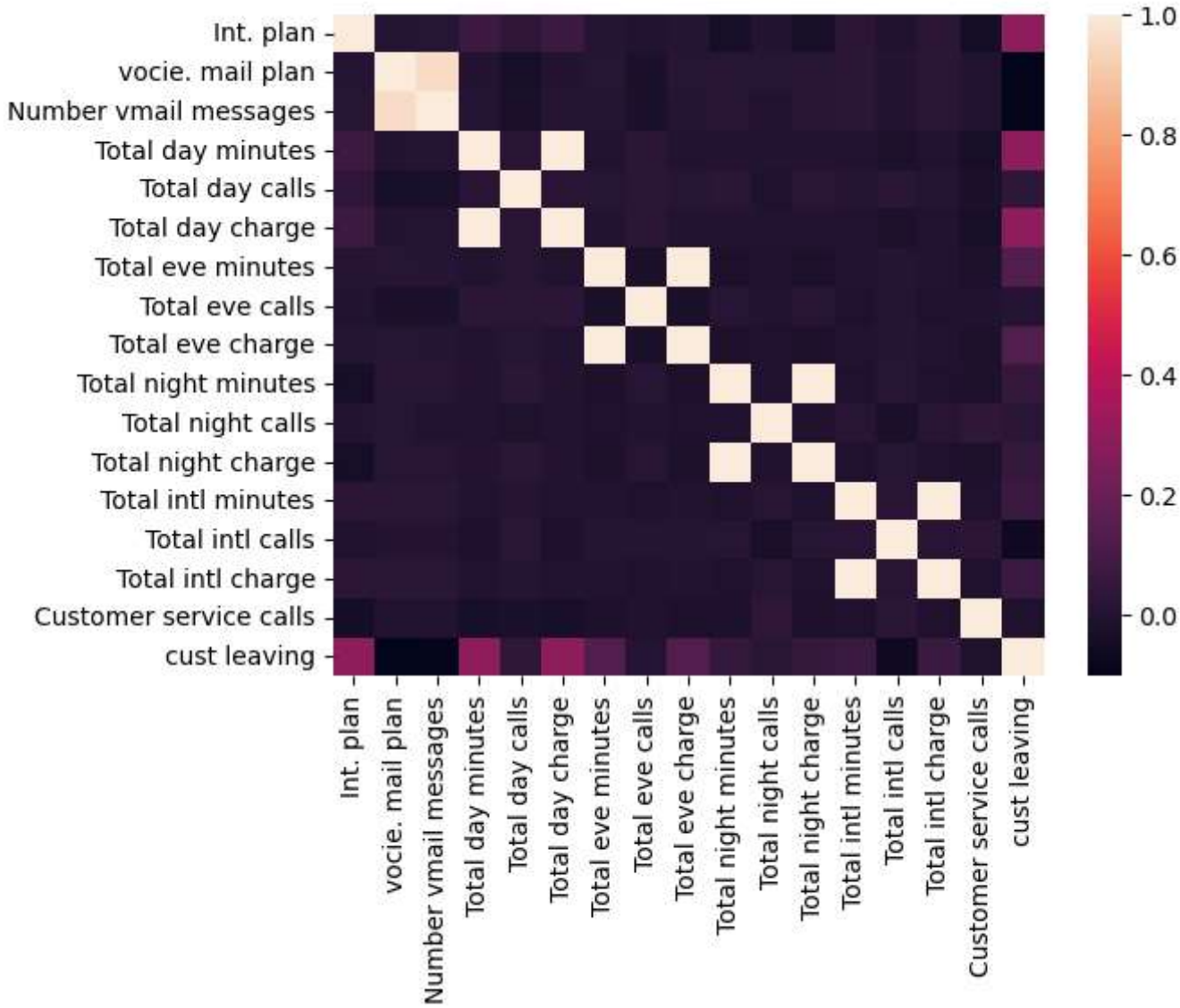
```
<AxesSubplot:>
```



from above box plot, we can observe that there are no outlier and data is normalized

In [32]:
```
# obtaining correlations and getting heat map
sb.heatmap(data_final.corr())
```

Out[32]:
```
<AxesSubplot:>
```



we can deploy classification alogorithms on this data

In [33]:
```
iv=data_final.iloc[:,:-1]
iv
```

Out[33]:

	Int. plan	vocie. mail plan	Number vmail messages	Total day minutes	Total day calls	Total day charge	Total eve minutes	Total eve calls	Total eve charge	Total night minutes	Total night calls	Total night charge	Total intl minutes	Tc
0	0	1	1.234883	1.566767	0.476643	1.567036	-0.070610	-0.055940	-0.070427	0.866743	-0.465494	0.866029	-0.085008	-0.
1	0	1	1.307948	-0.333738	1.124503	-0.334013	-0.108080	0.144867	-0.107549	1.058571	0.147825	1.059390	1.240482	-0.
2	0	0	-0.591760	1.168304	0.675985	1.168464	-1.573383	0.496279	-1.573900	-0.756869	0.198935	-0.755571	0.703121	0.
3	1	0	-0.591760	0.801056	-0.121381	0.801216	0.386928	0.044463	0.386648	0.059881	0.914473	0.061821	-1.410498	0.
4	1	0	-0.591760	-0.418205	-1.068253	-0.418264	-1.930341	-0.306949	-1.931206	0.216112	-0.209945	0.215631	-1.123906	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2337	0	0	-0.591760	0.441154	-0.769241	0.441529	-2.232079	0.496279	-2.232829	-0.685675	1.118913	-0.685257	0.524001	-0.
2338	0	0	-0.591760	1.840366	0.626149	1.840312	0.024053	-0.507756	0.024700	-0.871570	0.147825	-0.869830	0.058288	-0.
2339	0	1	1.307948	-0.783616	-0.619735	-0.783352	-0.892994	1.249304	-0.894089	0.924094	-0.312165	0.923158	0.452353	1.
2340	0	1	1.527146	-1.766002	1.523186	-1.766281	0.057580	1.550515	0.057183	0.566148	-1.385473	0.567197	-0.765665	-0.
2341	0	0	-0.591760	-0.183166	0.675985	-0.182793	-0.064693	0.245270	-0.065786	-0.141834	-0.925484	-0.140330	0.488177	-0.

2342 rows × 16 columns

In [34]:

```
dv=data_final.iloc[:,-1]
dv
```

Out[34]:

```
0       0
1       0
2       0
3       0
4       0
       ..
2337    0
2338    1
2339    0
2340    0
2341    0
Name: cust leaving, Length: 2342, dtype: int64
```

splitting data into train and test data

In [35]:
```
# importing train test split
from sklearn.model_selection import train_test_split
```

In [36]:
```
# splitting
iv_train,iv_test,dv_train,dv_test=train_test_split(iv,dv,
                                                   test_size=0.2,
                                                   random_state=1)
```

In [37]:
```
# checking splitted data
dv_test
```

Out[37]:
```
560     1
685     0
65      0
1294    0
1359    1
       ..
851     0
1231    0
447     0
1860    0
1422    0
Name: cust leaving, Length: 469, dtype: int64
```

In [ ]:

building algorithms

kNN

In [38]:
```
# importing knn algorithm
from sklearn.neighbors import KNeighborsClassifier
```

In [39]:
```
#creating object for KNN
knn_obj=KNeighborsClassifier()
```

In [40]:
```
# fitting data to Knn algorithm
knn_obj.fit(iv_train,dv_train)
```

Out[40]:
```
KNeighborsClassifier()
```

In [41]:
```
# getting predictions
knn_pred=knn_obj.predict(iv_test)
```

```
C:\Users\G RAJESH\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other red
uction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along.
In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which th
e statistic is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False
to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [42]:
```
# importing confusion matrix, confusion matrix display and classification report
from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay,classification_report
```

In [43]:
```
# getting confusion matrix
confusion_matrix(dv_test,knn_pred)
```

Out[43]:
```
array([[418,   0],
       [ 36,  15]], dtype=int64)
```

In [44]:
```
# getting classification report
print(classification_report(dv_test,knn_pred))
```

```
              precision    recall  f1-score   support

           0       0.92      1.00      0.96       418
           1       1.00      0.29      0.45        51

    accuracy                           0.92       469
   macro avg       0.96      0.65      0.71       469
weighted avg       0.93      0.92      0.90       469
```

In [45]:
```
from sklearn.metrics import roc_curve,roc_auc_score,RocCurveDisplay
```

In [46]:
```
print(roc_curve(dv_test,knn_pred))
```

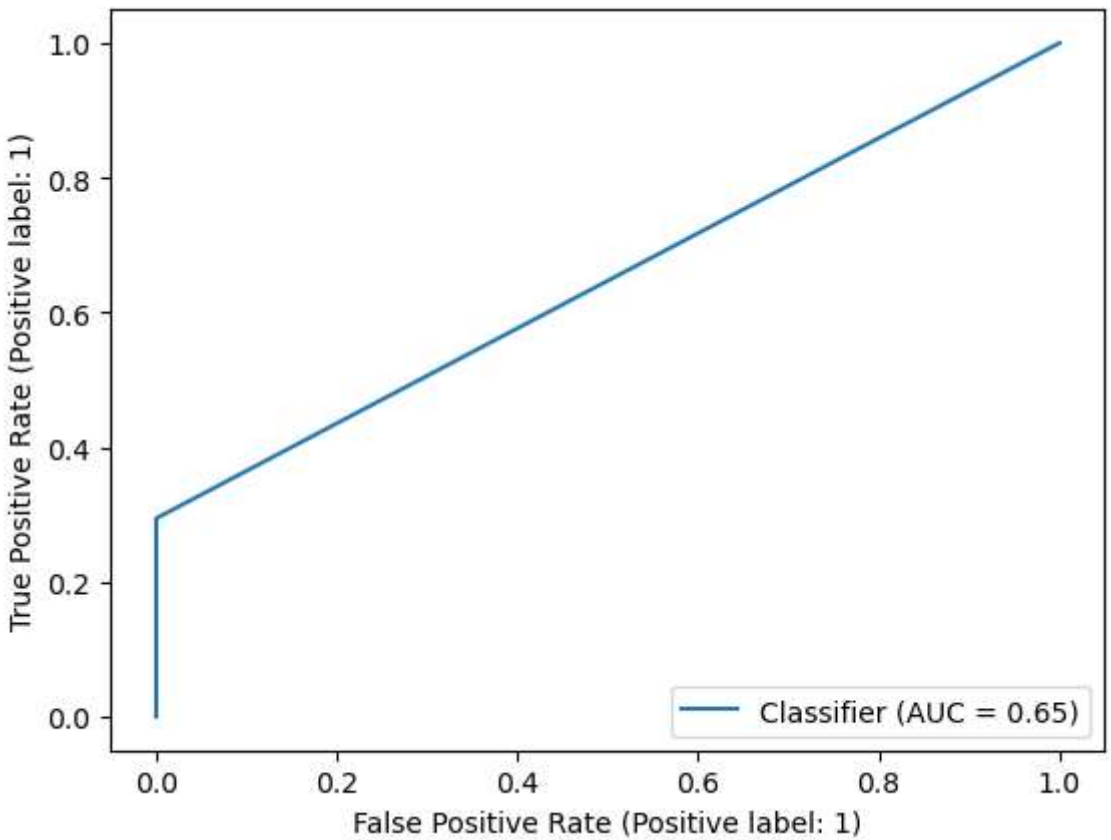
```
(array([0., 0., 1.]), array([0.        , 0.29411765, 1.        ]), array([2, 1, 0], dtype=int64))
```

In [47]:
```
roc_auc_score(dv_test,knn_pred)
```

Out[47]:
```
0.6470588235294118
```

In [48]:
```
RocCurveDisplay.from_predictions(dv_test,knn_pred)
```

Out[48]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x1d6689e4fa0>
```



In [ ]:

In [ ]:

Logistic regression

In [49]:
```
# importing algorithm
from sklearn.linear_model import LogisticRegression
```

In [50]:
```
# creating object
logi_obj=LogisticRegression()
```

In [51]:
```
#fitting data
logi_obj.fit(iv_train,dv_train)
```

Out[51]:
```
LogisticRegression()
```

In [52]:
```
#getting predictions
logi_pred=logi_obj.predict(iv_test)
```

logistic regression evaluaton

In [53]:
```
# getting confusion matrix
confusion_matrix(dv_test,logi_pred)
```

Out[53]:
```
array([[413,   5],
       [ 34,  17]], dtype=int64)
```

In [54]:
```
# getting classification report
print(classification_report(dv_test,logi_pred))
```

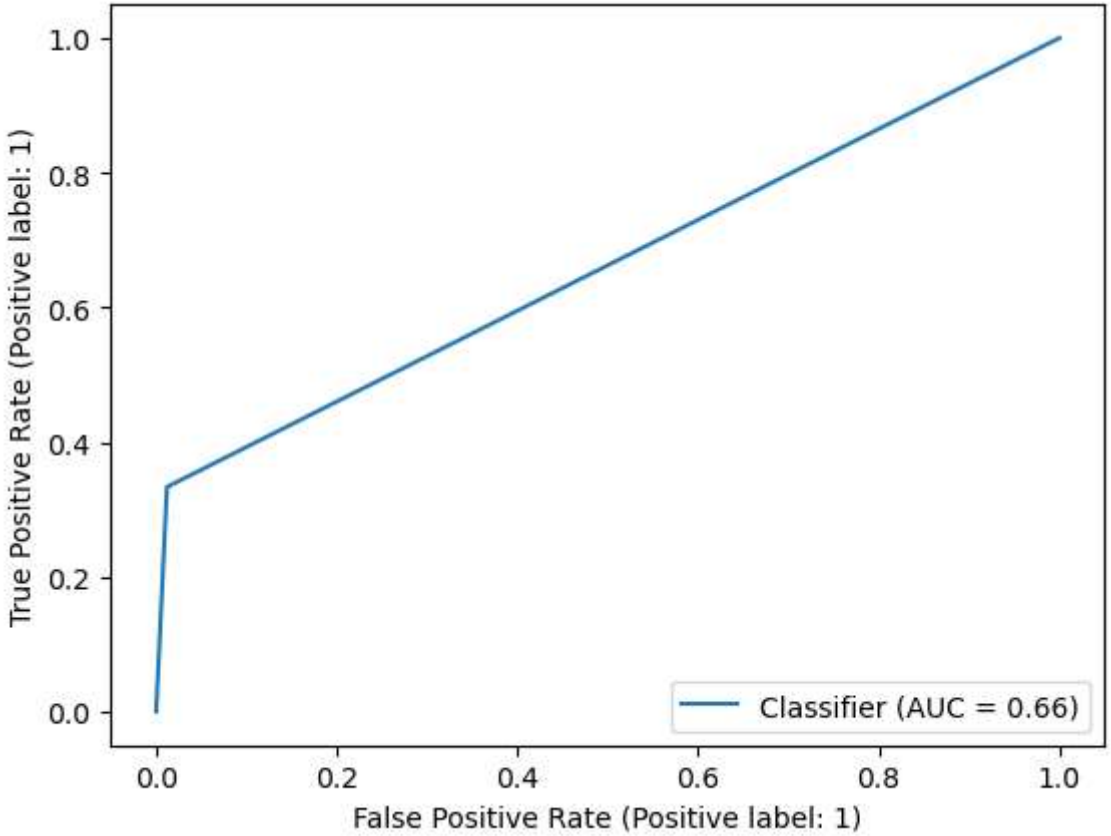
```
              precision    recall  f1-score   support

           0       0.92      0.99      0.95       418
           1       0.77      0.33      0.47        51

    accuracy                           0.92       469
   macro avg       0.85      0.66      0.71       469
weighted avg       0.91      0.92      0.90       469
```

In [55]:
```
RocCurveDisplay.from_predictions(dv_test,logi_pred)
```

Out[55]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x1d668db27f0>
```



In [ ]:

In [ ]:

decision tree

In [56]:
```
# importing decision tree
from sklearn.tree import DecisionTreeClassifier
```

In [57]:
```
# creating object
dt_obj=DecisionTreeClassifier()
```

In [58]:
```
# fitting data
dt_obj.fit(iv_train,dv_train)
```

Out[58]:
```
DecisionTreeClassifier()
```

In [59]:
```
# getting predictions
dt_pred=dt_obj.predict(iv_test)
```

Decision tree evaluation

In [60]:
```
# getting confusion matrix
confusion_matrix(dv_test,dt_pred)
```

Out[60]:
```
array([[391,  27],
       [ 21,  30]], dtype=int64)
```

In [61]:
```
#getting classification report
print(classification_report(dv_test,dt_pred))
```

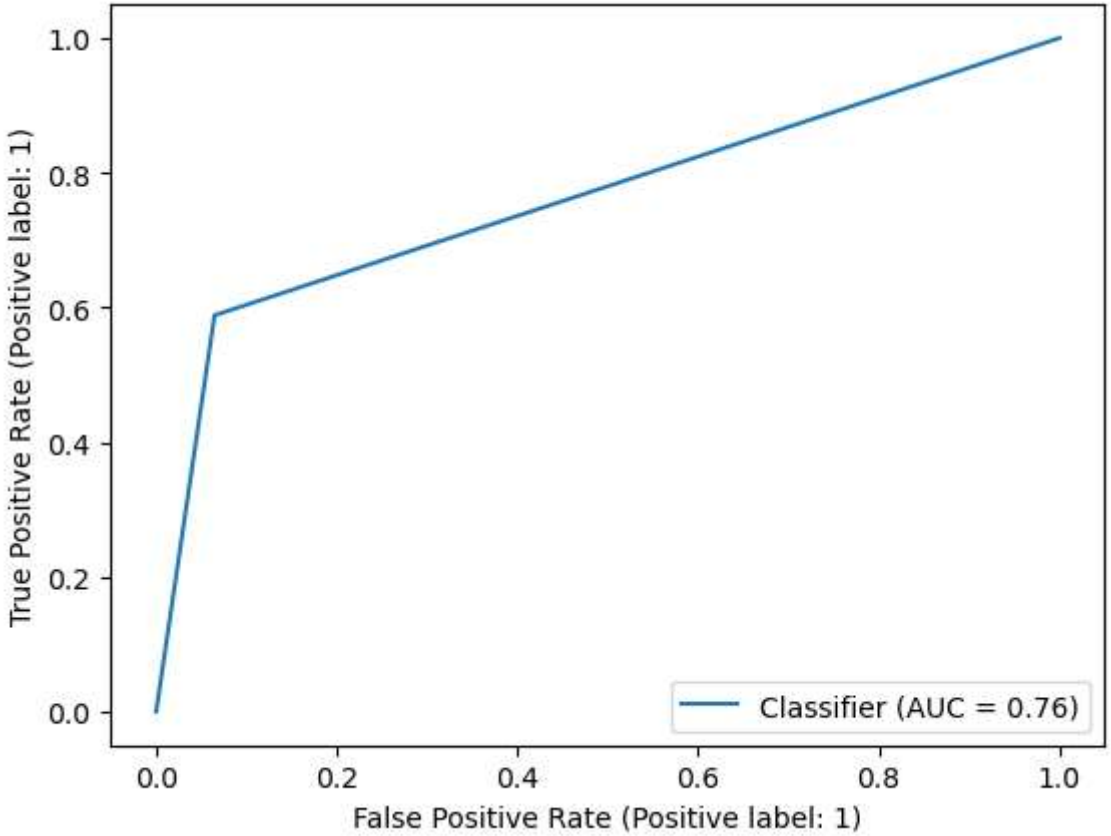
```
              precision    recall  f1-score   support

           0       0.95      0.94      0.94       418
           1       0.53      0.59      0.56        51

    accuracy                           0.90       469
   macro avg       0.74      0.76      0.75       469
weighted avg       0.90      0.90      0.90       469
```

In [62]:
```
RocCurveDisplay.from_predictions(dv_test,dt_pred)
```

Out[62]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x1d668b4f040>
```



In [ ]:

In [ ]:

Random Forest

In [63]:
```
#

from sklearn.ensemble import RandomForestClassifier
```

In [64]:
```
random_obj=RandomForestClassifier()
```

In [65]:
```
random_obj.fit(iv_train,dv_train)
```

Out[65]:
```
RandomForestClassifier()
```

In [66]:
```
random_pred=random_obj.predict(iv_test)
```

In [67]:
```
confusion_matrix(dv_test,random_pred)
```

Out[67]:
```
array([[414,   4],
       [ 18,  33]], dtype=int64)
```

In [68]:
```
print(classification_report(dv_test,random_pred))
```

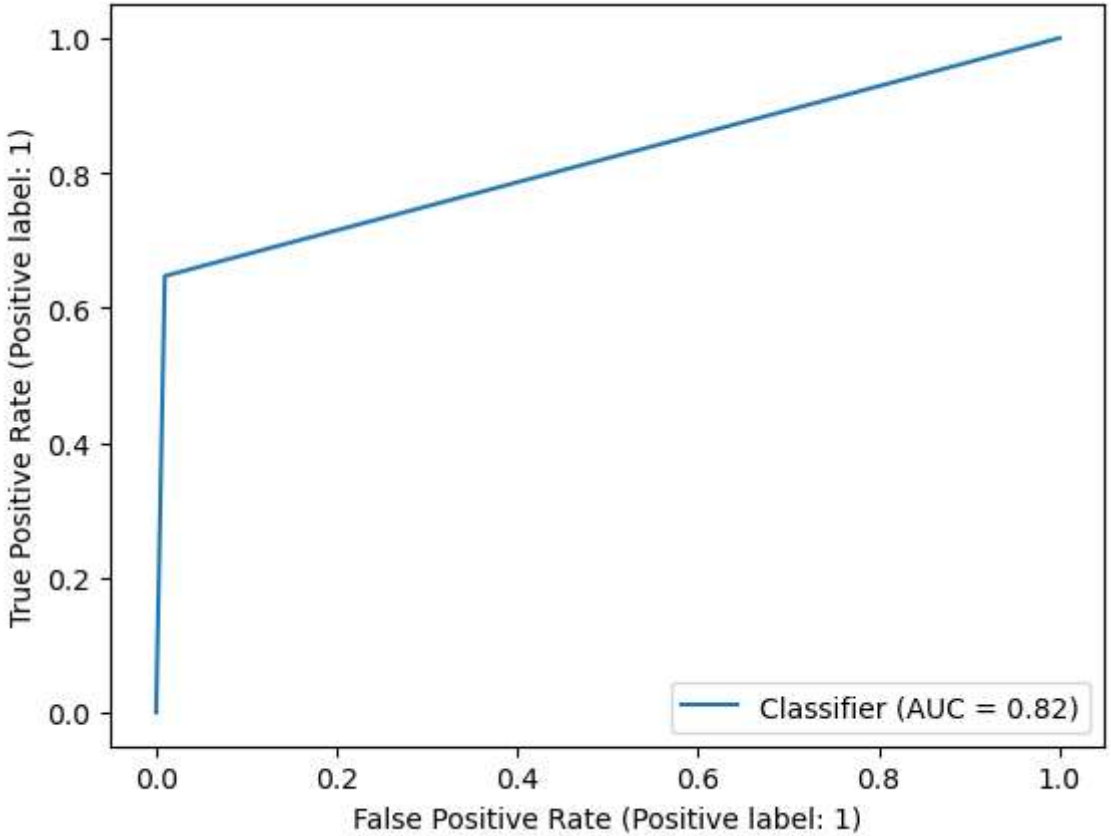
```
              precision    recall  f1-score   support

           0       0.96      0.99      0.97       418
           1       0.89      0.65      0.75        51

    accuracy                           0.95       469
   macro avg       0.93      0.82      0.86       469
weighted avg       0.95      0.95      0.95       469
```

In [69]:
```
RocCurveDisplay.from_predictions(dv_test,random_pred)
```

Out[69]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x1d668ec8f70>
```



sampling to avoid overfitting

In [71]:
```
#installing imblearn
pip install imblearn
```

```
Collecting imblearn
  Downloading imblearn-0.0-py2.py3-none-any.whl (1.9 kB)
Collecting imbalanced-learn
  Downloading imbalanced_learn-0.10.1-py3-none-any.whl (226 kB)
     ------------------------------------ 226.0/226.0 kB 275.9 kB/s eta 0:00:00
Collecting joblib>=1.1.1
  Downloading joblib-1.2.0-py3-none-any.whl (297 kB)
     ------------------------------------ 298.0/298.0 kB 1.4 MB/s eta 0:00:00
Requirement already satisfied: scipy>=1.3.2 in c:\users\g rajesh\anaconda3\lib\site-packages (from imbalanced-learn->im
blearn) (1.9.1)
Requirement already satisfied: scikit-learn>=1.0.2 in c:\users\g rajesh\anaconda3\lib\site-packages (from imbalanced-le
arn->imblearn) (1.0.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\g rajesh\anaconda3\lib\site-packages (from imbalanced-l
earn->imblearn) (2.2.0)
Requirement already satisfied: numpy>=1.17.3 in c:\users\g rajesh\anaconda3\lib\site-packages (from imbalanced-learn->i
mblearn) (1.21.5)
Installing collected packages: joblib, imbalanced-learn, imblearn
  Attempting uninstall: joblib
    Found existing installation: joblib 1.1.0
    Uninstalling joblib-1.1.0:
      Successfully uninstalled joblib-1.1.0
Successfully installed imbalanced-learn-0.10.1 imblearn-0.0 joblib-1.2.0
Note: you may need to restart the kernel to use updated packages.
```

In [72]:
```
from imblearn.over_sampling import SMOTE
```

In [73]:
```
# creating object and fitting data to it
sm_obj=SMOTE(random_state=1)
iv_re,dv_re=sm_obj.fit_resample(iv_train,dv_train)
```

In [74]:
```
#checking
iv_re.shape,dv_re.shape
```

Out[74]:
```
((3346, 16), (3346,))
```

In [75]:
```
iv_train.shape,dv_train.shape
```

Out[75]:
```
((1873, 16), (1873,))
```

In [76]:
```
dv_train.value_counts()
```

Out[76]:
```
0    1673
1     200
Name: cust leaving, dtype: int64
```

In [77]:
```
dv_re.value_counts()
```

Out[77]:
```
1    1673
0    1673
Name: cust leaving, dtype: int64
```

In [78]:
```
# creating object for random forest
re_rand_obj=RandomForestClassifier(random_state=1)
```

In [79]:
```
re_rand_obj.fit(iv_re,dv_re)
```

Out[79]:
```
RandomForestClassifier(random_state=1)
```

In [80]:
```
re_rand_pred=re_rand_obj.predict(iv_test)
```

In [81]:
```
print(classification_report(dv_test,re_rand_pred))
```

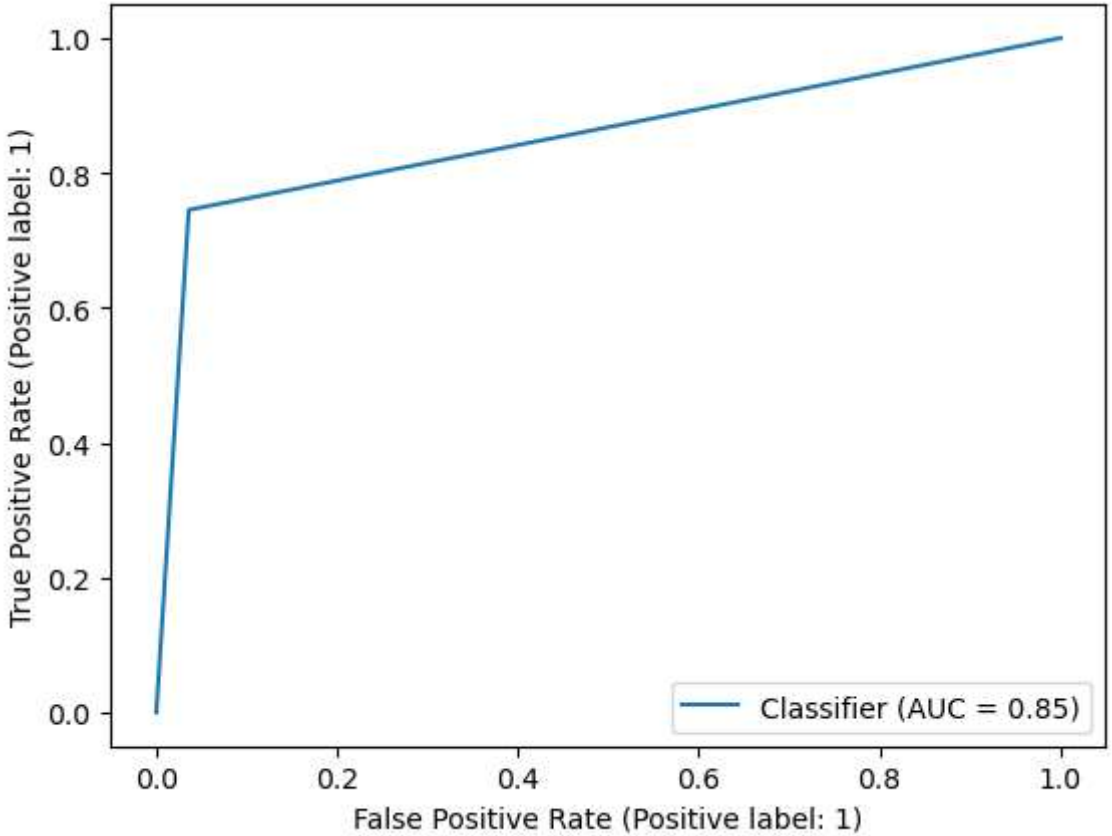
```
              precision    recall  f1-score   support

           0       0.97      0.96      0.97       418
           1       0.72      0.75      0.73        51

    accuracy                           0.94       469
   macro avg       0.84      0.85      0.85       469
weighted avg       0.94      0.94      0.94       469
```

In [82]:
```
RocCurveDisplay.from_predictions(dv_test,re_rand_pred)
```

Out[82]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x1d66a3c2af0>
```



area under curve has increased from 82% to 85% after balancing...