In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
df=pd.read_excel(r"C:\Users\Biswajeet Jena\Downloads\customer_churn_large_dataset.xlsx")
df
```

Out[2]:

	CustomerID	Name	Age	Gender	Location	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn
0	1	Customer_1	63	Male	Los Angeles	17	73.36	236	0
1	2	Customer_2	62	Female	New York	1	48.76	172	0
2	3	Customer_3	24	Female	Los Angeles	5	85.47	460	0
3	4	Customer_4	36	Female	Miami	3	97.94	297	1
4	5	Customer_5	46	Female	Miami	19	58.14	266	0
...	...	...	...	...	...	...	...	...	...
99995	99996	Customer_99996	33	Male	Houston	23	55.13	226	1
99996	99997	Customer_99997	62	Female	New York	19	61.65	351	0
99997	99998	Customer_99998	64	Male	Chicago	17	96.11	251	1
99998	99999	Customer_99999	51	Female	New York	20	49.25	434	1
99999	100000	Customer_100000	27	Female	Los Angeles	19	76.57	173	1

100000 rows × 9 columns

In [3]:
```
df.shape
```

Out[3]:
```
(100000, 9)
```

In [ ]:
```

```

DATA CLEANING

In [4]:
```
df1=df.drop(['CustomerID','Name'],axis=1)
df1
```

Out[4]:

	Age	Gender	Location	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn
0	63	Male	Los Angeles	17	73.36	236	0
1	62	Female	New York	1	48.76	172	0
2	24	Female	Los Angeles	5	85.47	460	0
3	36	Female	Miami	3	97.94	297	1
4	46	Female	Miami	19	58.14	266	0
...	...	...	...	...	...	...	...
99995	33	Male	Houston	23	55.13	226	1
99996	62	Female	New York	19	61.65	351	0
99997	64	Male	Chicago	17	96.11	251	1
99998	51	Female	New York	20	49.25	434	1
99999	27	Female	Los Angeles	19	76.57	173	1

100000 rows × 7 columns

In [5]:
```
df1.shape
```

Out[5]:
```
(100000, 7)
```

In [6]:
```
df1.columns
```

Out[6]:
```
Index(['Age', 'Gender', 'Location', 'Subscription_Length_Months',
       'Monthly_Bill', 'Total_Usage_GB', 'Churn'],
      dtype='object')
```

In [7]:
```
df1.columns.isna()
```

Out[7]:
```
array([False, False, False, False, False, False, False])
```

In [8]:
```
df1.dtypes
```

Out[8]:
```
Age                             int64
Gender                         object
Location                       object
Subscription_Length_Months      int64
Monthly_Bill                  float64
Total_Usage_GB                  int64
Churn                           int64
dtype: object
```

In [9]:
```
df1.replace({'Female':1,'Male':0},inplace=True)
```

In [10]:
```
df1
```

Out[10]:

	Age	Gender	Location	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn
0	63	0	Los Angeles	17	73.36	236	0
1	62	1	New York	1	48.76	172	0
2	24	1	Los Angeles	5	85.47	460	0
3	36	1	Miami	3	97.94	297	1
4	46	1	Miami	19	58.14	266	0
...	...	...	...	...	...	...	...
99995	33	0	Houston	23	55.13	226	1
99996	62	1	New York	19	61.65	351	0
99997	64	0	Chicago	17	96.11	251	1
99998	51	1	New York	20	49.25	434	1
99999	27	1	Los Angeles	19	76.57	173	1

100000 rows × 7 columns

In [11]:
```
dummy=pd.get_dummies(df1.Location)
dummy
```

Out[11]:

	Chicago	Houston	Los Angeles	Miami	New York
0	0	0	1	0	0
1	0	0	0	0	1
2	0	0	1	0	0
3	0	0	0	1	0
4	0	0	0	1	0
...	...	...	...	...	...
99995	0	1	0	0	0
99996	0	0	0	0	1
99997	1	0	0	0	0
99998	0	0	0	0	1
99999	0	0	1	0	0

100000 rows × 5 columns

In [12]:
```
df2=pd.concat([df1,dummy],axis=1)
df2
```

Out[12]:

	Age	Gender	Location	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn	Chicago	Houston	Los Angeles	Miami	New York
0	63	0	Los Angeles	17	73.36	236	0	0	0	1	0	0
1	62	1	New York	1	48.76	172	0	0	0	0	0	1
2	24	1	Los Angeles	5	85.47	460	0	0	0	1	0	0
3	36	1	Miami	3	97.94	297	1	0	0	0	1	0
4	46	1	Miami	19	58.14	266	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...
99995	33	0	Houston	23	55.13	226	1	0	1	0	0	0
99996	62	1	New York	19	61.65	351	0	0	0	0	0	1
99997	64	0	Chicago	17	96.11	251	1	1	0	0	0	0
99998	51	1	New York	20	49.25	434	1	0	0	0	0	1
99999	27	1	Los Angeles	19	76.57	173	1	0	0	1	0	0

100000 rows × 12 columns

In [13]:
```
df3=df2.drop("Location",axis=1)
df3
```

Out[13]:

	Age	Gender	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Churn	Chicago	Houston	Los Angeles	Miami	New York
0	63	0	17	73.36	236	0	0	0	1	0	0
1	62	1	1	48.76	172	0	0	0	0	0	1
2	24	1	5	85.47	460	0	0	0	1	0	0
3	36	1	3	97.94	297	1	0	0	0	1	0
4	46	1	19	58.14	266	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...
99995	33	0	23	55.13	226	1	0	1	0	0	0
99996	62	1	19	61.65	351	0	0	0	0	0	1
99997	64	0	17	96.11	251	1	1	0	0	0	0
99998	51	1	20	49.25	434	1	0	0	0	0	1
99999	27	1	19	76.57	173	1	0	0	1	0	0

100000 rows × 11 columns

In [14]:
```
df3.columns.isnull()
```

Out[14]:
```
array([False, False, False, False, False, False, False, False, False,
       False, False])
```

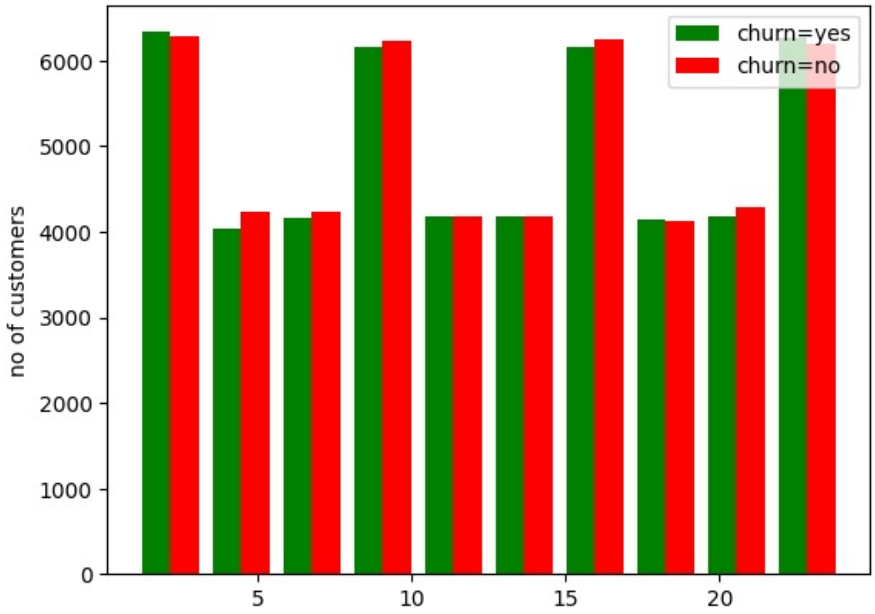
In [15]:
```
df3.dtypes
```

Out[15]:
```
Age                           int64
Gender                        int64
Subscription_Length_Months    int64
Monthly_Bill                float64
Total_Usage_GB                int64
Churn                         int64
Chicago                       uint8
Houston                       uint8
Los Angeles                   uint8
Miami                         uint8
New York                      uint8
dtype: object
```

In [16]:
```
Subscription_Length_Months_churn_yes=df3[df3.Churn==1].Subscription_Length_Months
Subscription_Length_Months_churn_no=df3[df3.Churn==0].Subscription_Length_Months
```

In [17]:
```
plt.hist([Subscription_Length_Months_churn_yes,Subscription_Length_Months_churn_no],label=['churn=yes','churn=n
plt.legend()
plt.xlabel('')
plt.ylabel('no of customers')
```

Out[17]: Text(0, 0.5, 'no of customers')



In [ ]:

FEATURE SCALING

In [18]:
```
x=df3.drop('Churn',axis=1)
x
```

Out[18]:

	Age	Gender	Subscription_Length_Months	Monthly_Bill	Total_Usage_GB	Chicago	Houston	Los Angeles	Miami	New York
0	63	0	17	73.36	236	0	0	1	0	0
1	62	1	1	48.76	172	0	0	0	0	1
2	24	1	5	85.47	460	0	0	1	0	0
3	36	1	3	97.94	297	0	0	0	1	0
4	46	1	19	58.14	266	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...
99995	33	0	23	55.13	226	0	1	0	0	0
99996	62	1	19	61.65	351	0	0	0	0	1
99997	64	0	17	96.11	251	1	0	0	0	0
99998	51	1	20	49.25	434	0	0	0	0	1
99999	27	1	19	76.57	173	0	0	1	0	0

100000 rows × 10 columns

In [19]:
```
y=df3.Churn
y
```

Out[19]:
```
0        0
1        0
2        0
3        1
4        0
        ..
99995    1
99996    0
99997    1
99998    1
99999    1
Name: Churn, Length: 100000, dtype: int64
```

In [20]:
```
from sklearn.model_selection import train_test_split
```

In [21]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [22]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.fit_transform(x_test)
```

In [23]:
```
x_train_scaled
```

In [23]:
```
x_train_scaled
```

Out[23]:
```
array([[0.28846154, 1.        , 0.39130435, ..., 0.        , 0.        ,
        0.        ],
       [0.25      , 0.        , 0.13043478, ..., 0.        , 0.        ,
        0.        ],
       [0.42307692, 0.        , 0.56521739, ..., 0.        , 1.        ,
        0.        ],
       ...,
       [0.65384615, 1.        , 0.08695652, ..., 1.        , 0.        ,
        0.        ],
       [0.28846154, 0.        , 0.56521739, ..., 0.        , 1.        ,
        0.        ],
       [0.76923077, 0.        , 0.43478261, ..., 0.        , 0.        ,
        0.        ]])
```

In [24]:
```
len(x_train_scaled)
```

Out[24]:
```
80000
```

In [25]:
```
x_test_scaled
```

Out[25]:
```
array([[0.82692308, 1.        , 1.        , ..., 0.        , 0.        ,
        1.        ],
       [0.57692308, 1.        , 0.13043478, ..., 0.        , 1.        ,
        0.        ],
       [0.07692308, 1.        , 0.30434783, ..., 0.        , 0.        ,
        0.        ],
       ...,
       [0.34615385, 1.        , 0.17391304, ..., 0.        , 0.        ,
        1.        ],
       [0.48076923, 1.        , 0.82608696, ..., 1.        , 0.        ,
        0.        ],
       [0.98076923, 1.        , 0.7826087 , ..., 1.        , 0.        ,
        0.        ]])
```

In [26]:
```
len(x_test_scaled)
```

Out[26]:
```
20000
```

In [ ]:
```

```

In [27]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report, con
```

In [28]:
```
model=LogisticRegression()
model.fit(x_train_scaled,y_train)
```

Out[28]:
▾ LogisticRegression

LogisticRegression()

In [29]:
```
yp=model.predict(x_test_scaled)
```

In [30]:
```
yp[:10]
```

Out[30]:
```
array([1, 0, 0, 1, 0, 1, 0, 0, 1, 0], dtype=int64)
```

In [31]:
```
y_test[:10]
```

Out[31]:
```
35215    1
53853    1
94918    1
46550    1
88915    1
10911    0
24210    1
66559    0
35328    1
79308    1
Name: Churn, dtype: int64
```

In [32]:
```
model.score(x_test_scaled,y_test)
```

Out[32]:
```
0.4974
```

In [37]:
```
precision,recall,f1
```

Out[37]:
```
(0.49542058327307786, 0.41208901363271855, 0.4499288606763708)
```

In [38]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [39]:
```
rf_model = RandomForestClassifier()
rf_model.fit(x_train_scaled,y_train)
```

Out[39]:
```
RandomForestClassifier
RandomForestClassifier()
```

In [40]:
```
yp1=rf_model.predict(x_test_scaled)
yp1
```

Out[40]:
```
array([0, 1, 0, ..., 1, 1, 1], dtype=int64)
```

In [41]:
```
yp1[:10]
```

Out[41]:
```
array([0, 1, 0, 1, 0, 0, 0, 0, 0, 1], dtype=int64)
```

In [42]:
```
y_test[:10]
```

Out[42]:
```
35215    1
53853    1
94918    1
46550    1
88915    1
10911    0
24210    1
66559    0
35328    1
79308    1
Name: Churn, dtype: int64
```

In [43]:
```
rf_model.score(x_test_scaled,y_test)
```

Out[43]:
```
0.49895
```

In [44]:
```
precision1 = precision_score(y_test, yp1)
recall1 = recall_score(y_test, yp1)
f11 = f1_score(y_test, yp1)
```

In [45]:
```
precision1,recall1,f11
```

Out[45]:
```
(0.49765600583394104, 0.47884923817161185, 0.48807151979565777)
```

In [46]:
```
from sklearn.model_selection import GridSearchCV
```

In [47]:
```
param_grid = {
    'C': [0.1, 1, 10],
    'penalty': ['l1', 'l2'],
    'solver': ['liblinear', 'saga']
}
```

In [50]:
```
grid_search = GridSearchCV(estimator=model, param_grid=param_grid, scoring='accuracy', cv=5)
```

In [52]:
```
grid_search.fit(x_train_scaled, y_train)
```

```
C:\Users\Biswajeet Jena\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\linear_model\_sag.py:
350: ConvergenceWarning: The max_iter was reached which means the coef_ did not converge
  warnings.warn(
C:\Users\Biswajeet Jena\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\linear_model\_sag.py:
350: ConvergenceWarning: The max_iter was reached which means the coef_ did not converge
  warnings.warn(
```

Out[52]:
```
GridSearchCV
estimator: LogisticRegression
LogisticRegression
```

In [58]:
```
best_params = grid_search.best_params_
best_params
```

Out[58]:
```
{'C': 1, 'penalty': 'l1', 'solver': 'liblinear'}
```

In [57]:
```
accuracy = accuracy_score(y_test, yp)
print(f"Accuracy: {accuracy:.2f}")
```

```
Accuracy: 0.50
```

In [ ]: