In [1]:
```
import pandas as pd
D=pd.read_csv(r"C:\Users\Admin\Downloads\smart_home_device_usage_data.c
D
```

Out[1]:

	UserID	DeviceType	UsageHoursPerDay	EnergyConsumption	UserPreferences	Malfunc
0	1	Smart Speaker	15.307188	1.961607	1	
1	2	Camera	19.973343	8.610689	1	
2	3	Security System	18.911535	2.651777	1	
3	4	Camera	7.011127	2.341653	0	
4	5	Camera	22.610684	4.859069	1	
...	...	...	...	...	...	
5398	5399	Thermostat	4.556314	5.871764	1	
5399	5400	Lights	0.561856	1.555992	1	
5400	5401	Smart Speaker	11.096236	7.677779	0	
5401	5402	Security System	8.782169	7.467929	0	
5402	5403	Thermostat	13.540381	9.043076	0	

5403 rows × 8 columns

In [106]:
```
D.columns
```

Out[106]:
```
Index(['UserID', 'DeviceType', 'UsageHoursPerDay', 'EnergyConsumption',
       'UserPreferences', 'MalfunctionIncidents', 'DeviceAgeMonths',
       'SmartHomeEfficiency'],
      dtype='object')
```

In [2]:
```
D.isnull().sum()
```

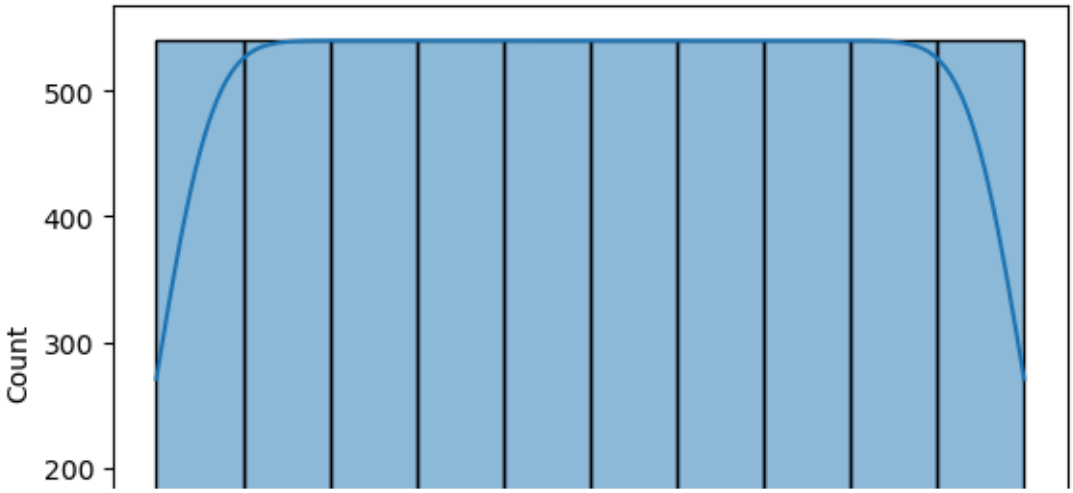
Out[2]:
```
UserID                0
DeviceType            0
UsageHoursPerDay      0
EnergyConsumption     0
UserPreferences       0
MalfunctionIncidents  0
DeviceAgeMonths       0
SmartHomeEfficiency   0
dtype: int64
```

In [3]:
```
1 D.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5403 entries, 0 to 5402
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   UserID                5403 non-null   int64  
 1   DeviceType            5403 non-null   object 
 2   UsageHoursPerDay      5403 non-null   float64
 3   EnergyConsumption     5403 non-null   float64
 4   UserPreferences       5403 non-null   int64  
 5   MalfunctionIncidents  5403 non-null   int64  
 6   DeviceAgeMonths       5403 non-null   int64  
 7   SmartHomeEfficiency   5403 non-null   int64  
dtypes: float64(2), int64(5), object(1)
memory usage: 337.8+ KB
```

In [4]:
```
1 import matplotlib.pyplot as plt
2 import seaborn as sns
3 for i in D.columns:
4     sns.histplot(D[i],bins=10,kde=True)
5     plt.show()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [5]:
```
1 D['UsageHoursPerDay'].mean()
```

Out[5]: 12.052992010466317

In [6]:
```
1 D['UsageHoursPerDay'].median()
```

Out[6]: 11.903768445051607

In [7]:
```
1 D['UsageHoursPerDay'].mode()[0]
```

Out[7]: 0.5012414329089748

In [8]:
```
1 D['EnergyConsumption'].mean()
```

Out[8]:
```
5.054301881355049
```

In [9]:
```
1 D['EnergyConsumption'].median()
```

Out[9]:
```
5.007047305947374
```

In [10]:
```
1 D['EnergyConsumption'].mode()[0]
```

Out[10]:
```
0.1015616713227616
```

In [11]:
```
1 D['DeviceAgeMonths'].mean()
```

Out[11]:
```
30.312233944105127
```

In [12]:
```
1 D['DeviceAgeMonths'].median()
```

Out[12]:
```
30.0
```

In [13]:
```
1 D['DeviceAgeMonths'].mode()[0]
```

Out[13]:
```
13
```

In [14]:
```
1 D['UsageHoursPerDay'].unique()
```

Out[14]:
```
array([15.30718848, 19.97334329, 18.91153466, ..., 11.09623585,
       8.78216919, 13.54038109])
```

In [15]:
```
A=D.loc[(D['UsageHoursPerDay']>20)&(D['EnergyConsumption']>9)]
A
```

Out[15]:

	UserID	DeviceType	UsageHoursPerDay	EnergyConsumption	UserPreferences	Malfunc
131	132	Camera	23.824094	9.916783	0	
262	263	Camera	20.689126	9.574219	1	
273	274	Smart Speaker	22.461781	9.107254	1	
312	313	Thermostat	20.679406	9.102111	0	
360	361	Smart Speaker	20.213066	9.369793	0	
...	...	...	...	...	...	
5054	5055	Camera	20.010880	9.269652	1	
5071	5072	Thermostat	21.093378	9.566639	1	
5079	5080	Camera	21.943399	9.268205	1	
5167	5168	Smart Speaker	22.077539	9.317121	1	
5260	5261	Camera	21.438391	9.998071	1	

103 rows × 8 columns

In [16]:
```
A['SmartHomeEfficiency'].value_counts() #at high value of UsageHoursPer
```

Out[16]:
```
SmartHomeEfficiency
0    72
1    31
Name: count, dtype: int64
```

In [17]:
```
D['DeviceType'].value_counts()
```

Out[17]:
```
DeviceType
Smart Speaker      1108
Camera             1101
Lights             1087
Security System    1068
Thermostat         1039
Name: count, dtype: int64
```

In [18]:
```
S=D.loc[(D['DeviceType']=='Smart Speaker')&(D['EnergyConsumption']>=8)&
S
```

Out[18]:

	UserID	DeviceType	UsageHoursPerDay	EnergyConsumption	UserPreferences	Malfunc
50	51	Smart Speaker	15.202794	9.260120	1	
58	59	Smart Speaker	16.163266	9.861676	0	
136	137	Smart Speaker	9.884316	8.494920	0	
194	195	Smart Speaker	12.572135	9.181754	0	
286	287	Smart Speaker	7.836459	8.985337	0	
...	...	...	...	...	...	
5018	5019	Smart Speaker	22.644072	9.314185	1	
5020	5021	Smart Speaker	5.388577	9.696368	0	
5032	5033	Smart Speaker	11.332785	8.078466	0	
5069	5070	Smart Speaker	16.839995	8.603232	0	
5170	5171	Smart Speaker	21.185917	8.206446	0	

100 rows × 8 columns

In [19]:
```
S['SmartHomeEfficiency'].value_counts()
```

Out[19]:
```
SmartHomeEfficiency
0    85
1    15
Name: count, dtype: int64
```

In [20]:
```
E=D.loc[(D['DeviceType']=='Thermostat')&(D['UsageHoursPerDay']>=22)&(D[
E
```

Out[20]:

	UserID	DeviceType	UsageHoursPerDay	EnergyConsumption	UserPreferences	Malfunc
138	139	Thermostat	23.488186	5.951844	0	
2211	2212	Thermostat	23.547888	2.386310	0	
3075	3076	Thermostat	23.696852	8.010203	0	
3243	3244	Thermostat	22.723328	2.994062	0	
3779	3780	Thermostat	22.839250	4.432218	0	
4104	4105	Thermostat	23.545233	5.931486	1	
4885	4886	Thermostat	22.445682	8.366797	0	
4925	4926	Thermostat	22.306454	1.926897	1	
4974	4975	Thermostat	22.658449	7.914858	0	
5117	5118	Thermostat	22.803371	6.955159	0	

In [21]:
```
D.groupby('DeviceType')[['UsageHoursPerDay']].mean().reset_index().sort
```

Out[21]:

	DeviceType	UsageHoursPerDay
0	Camera	12.113435
4	Thermostat	12.105753
1	Lights	12.052646
2	Security System	12.016149
3	Smart Speaker	11.979308

In [22]:
```
#UsageHoursPerDay of camera on average is max as compared to others
```

In [23]:
```
R=D.loc[(D['SmartHomeEfficiency']==1)&(D['UserPreferences']==1)]
R
```

Out[23]:

	UserID	DeviceType	UsageHoursPerDay	EnergyConsumption	UserPreferences	Malfunc
0	1	Smart Speaker	15.307188	1.961607	1	
1	2	Camera	19.973343	8.610689	1	
2	3	Security System	18.911535	2.651777	1	
4	5	Camera	22.610684	4.859069	1	
5	6	Thermostat	3.422127	5.038625	1	
...	...	...	...	...	...	
5386	5387	Security System	20.393943	3.104494	1	
5387	5388	Lights	10.532275	5.634707	1	
5388	5389	Thermostat	13.472427	6.728036	1	
5393	5394	Security System	18.847219	5.649036	1	
5396	5397	Camera	19.301279	0.792446	1	

1838 rows × 8 columns

In [24]:
```
R['DeviceType'].value_counts()# from SmartHomeEfficiency & UserPreferer
```

Out[24]:
```
DeviceType
Camera             390
Smart Speaker      385
Lights             372
Security System    352
Thermostat         339
Name: count, dtype: int64
```

In [25]:
```
DS=pd.get_dummies(D['DeviceType'],drop_first=True).replace({True:1,Fals
DS
```

Out[25]:

	Lights	Security System	Smart Speaker	Thermostat
0	0	0	1	0
1	0	0	0	0
2	0	1	0	0
3	0	0	0	0
4	0	0	0	0
...	...	...	...	...
5398	0	0	0	1
5399	1	0	0	0
5400	0	0	1	0
5401	0	1	0	0
5402	0	0	0	1

5403 rows × 4 columns

In [26]:
```
N=pd.concat([D,DS],axis=1)
N
```

Out[26]:

	UserID	DeviceType	UsageHoursPerDay	EnergyConsumption	UserPreferences	Malfunc
0	1	Smart Speaker	15.307188	1.961607	1	
1	2	Camera	19.973343	8.610689	1	
2	3	Security System	18.911535	2.651777	1	
3	4	Camera	7.011127	2.341653	0	
4	5	Camera	22.610684	4.859069	1	
...	...	...	...	...	...	
5398	5399	Thermostat	4.556314	5.871764	1	
5399	5400	Lights	0.561856	1.555992	1	
5400	5401	Smart Speaker	11.096236	7.677779	0	
5401	5402	Security System	8.782169	7.467929	0	
5402	5403	Thermostat	13.540381	9.043076	0	

5403 rows × 12 columns

In [27]:
```
N.drop(columns="DeviceType",inplace=True)
```

In [28]:
```
N.drop(columns="UserID",inplace=True)
```

In [29]:
```
N
```

Out[29]:

	UsageHoursPerDay	EnergyConsumption	UserPreferences	MalfunctionIncidents	Device
0	15.307188	1.961607	1	4	
1	19.973343	8.610689	1	0	
2	18.911535	2.651777	1	0	
3	7.011127	2.341653	0	3	
4	22.610684	4.859069	1	3	
...	...	...	...	...	
5398	4.556314	5.871764	1	0	
5399	0.561856	1.555992	1	4	
5400	11.096236	7.677779	0	0	
5401	8.782169	7.467929	0	2	
5402	13.540381	9.043076	0	0	

5403 rows × 10 columns

In [30]:
```
FS=N.drop(columns='UserPreferences',axis=1)
T=N['UserPreferences']
```

In [31]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(FS,T,train_size=0.65,ran
```

In [47]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
Log=LogisticRegression()
params={"C":[0.2,0.4,0.006,0.8],"penalty":["l1","l2"]}
G=GridSearchCV(Log,param_grid=params,scoring="accuracy",cv=6)
```

In [48]:
```
1 G.fit(X_train,y_train)
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.
py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown i
n:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sc
ikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-reg
ression (https://scikit-learn.org/stable/modules/linear_model.html#logisti
c-regression)
  n_iter_i = _check_optimize_result(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.
py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown i
n:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sc
ikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-reg
ression (https://scikit-learn.org/stable/modules/linear_model.html#logisti
c-regression)
  n_iter_i = _check_optimize_result(
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\model_selection\_valida
tion.py:378: FitFailedWarning:
24 fits failed out of a total of 48.
The score on these train-test partitions for these parameters will be set
to nan.
If these failures are not expected, you can try to debug them by setting e
rror_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------
------
24 fits failed with the following error:
Traceback (most recent call last):
  File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\model_selection
\_validation.py", line 686, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\linear_model\_l
ogistic.py", line 1162, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
             ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\linear_model\_l
ogistic.py", line 54, in _check_solver
    raise ValueError(
ValueError: Solver lbfgs supports only 'l2' or 'none' penalties, got l1 pe
nalty.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\model_selection\_searc
h.py:952: UserWarning: One or more of the test scores are non-finite: [
nan 0.79236992        nan 0.79237041        nan 0.79180012
        nan 0.79293972]
  warnings.warn(
```

Out[48]:
```
GridSearchCV(cv=6, estimator=LogisticRegression(),
             param_grid={'C': [0.2, 0.4, 0.006, 0.8], 'penalty': ['l1', 'l
2']},
             scoring='accuracy')
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [49]:
```
1 G.best_params_
```

Out[49]:
```
{'C': 0.8, 'penalty': 'l2'}
```

In [50]:
```
1 model=G.best_estimator_
2 model
```

Out[50]:
```
LogisticRegression(C=0.8)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [51]:
```
1 pred=model.predict(X_test)
2 pred
```

Out[51]:
```
array([0, 0, 1, ..., 0, 1, 0], dtype=int64)
```

In [52]:
```
1 model.score(X_train,y_train)
```

Out[52]:
```
0.7943605810310452
```

In [53]:
```
1 model.score(X_test,y_test)
```

Out[53]:
```
0.7928118393234672
```

In [54]:
```
1 from sklearn.metrics import classification_report,accuracy_score,confus
```

In [55]:
```
1 accuracy_score(y_test,pred)
```

Out[55]:
```
0.7928118393234672
```

In [56]:
```
1 print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

           0       0.74      0.88      0.81       927
           1       0.86      0.71      0.78       965

    accuracy                           0.79      1892
   macro avg       0.80      0.79      0.79      1892
weighted avg       0.80      0.79      0.79      1892
```

In [57]:
```
confusion_matrix(y_test,pred)
```

Out[57]:
```
array([[815, 112],
       [280, 685]], dtype=int64)
```

In [58]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
SVC=SVC()
prm={"gamma":[0.4,0.6,0.8],"kernel":['rbf']}
g=GridSearchCV(SVC,param_grid=prm,scoring='accuracy',cv=6)
```

In [59]:
```
g.fit(X_train,y_train)
```

Out[59]:
```
GridSearchCV(cv=6, estimator=SVC(),
             param_grid={'gamma': [0.4, 0.6, 0.8], 'kernel': ['rbf']},
             scoring='accuracy')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [60]:
```
g.best_params_
```

Out[60]:
```
{'gamma': 0.4, 'kernel': 'rbf'}
```

In [61]:
```
L=g.best_estimator_
L
```

Out[61]:
```
SVC(gamma=0.4)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [62]:
```
predict=L.predict(X_test)
predict
```

Out[62]:
```
array([0, 1, 1, ..., 0, 1, 0], dtype=int64)
```

In [63]:
```
L.score(X_train,y_train)
```

Out[63]:
```
0.9928795215038451
```

In [64]:
```
L.score(X_test,y_test)
```

Out[64]:
```
0.6094080338266384
```

In [65]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier
KNC=KNeighborsClassifier()
params={'n_neighbors':[4,6,5,10]}
I=GridSearchCV(KNC,param_grid=params,scoring="accuracy",cv=7)
```

In [66]:
```
I.fit(X_train,y_train)
```

Out[66]:
```
GridSearchCV(cv=7, estimator=KNeighborsClassifier(),
             param_grid={'n_neighbors': [4, 6, 5, 10]}, scoring='accurac
y')
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [67]:
```
I.best_params_
```

Out[67]:
```
{'n_neighbors': 5}
```

In [68]:
```
J=I.best_estimator_
J
```

Out[68]:
```
KNeighborsClassifier()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [69]:
```
P=J.predict(X_test)
P
```

Out[69]:
```
array([0, 1, 1, ..., 0, 1, 0], dtype=int64)
```

In [70]:
```
J.score(X_train,y_train)
```

Out[70]:
```
0.743377954998576
```

In [71]:
```
J.score(X_test,y_test)
```

Out[71]:
```
0.5724101479915433
```

In [107]:
```
from sklearn.ensemble import RandomForestClassifier
R1= RandomForestClassifier(n_estimators=25)
```

In [108]:
```
1 R1.fit(X_train,y_train)
```

Out[108]:
```
RandomForestClassifier(n_estimators=25)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [109]:
```
1 R1.score(X_train,y_train)
```

Out[109]:
```
0.9982910851609228
```

In [110]:
```
1 R1.score(X_test,y_test)
```

Out[110]:
```
0.8002114164904862
```

In [111]:
```
1 from sklearn.ensemble import AdaBoostClassifier
2 AD=AdaBoostClassifier(n_estimators=50)
3 AD.fit(X_train,y_train)
```

Out[111]:
```
AdaBoostClassifier()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [112]:
```
1 AD.score(X_train,y_train)
```

Out[112]:
```
0.801196240387354
```

In [113]:
```
1 AD.score(X_test,y_test)
```

Out[113]:
```
0.7875264270613108
```

In [115]:
```
1 H={'models':["Log","KNN","SVC","R1","AD"],"Train":[79.43,99.28,74.33,99
2 H
```

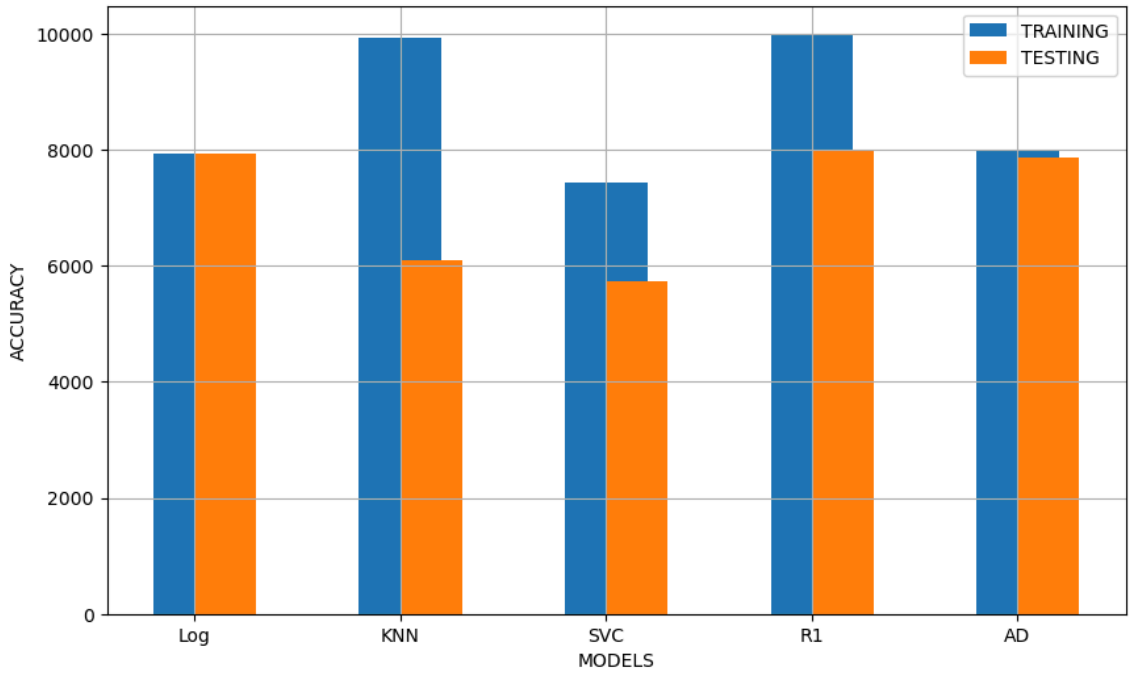
Out[115]:
```
{'models': ['Log', 'KNN', 'SVC', 'R1', 'AD'],
 'Train': [79.43, 99.28, 74.33, 99.82, 80.11],
 'Test': [79.28, 60.94, 57.24, 80.02, 78.75]}
```

In [116]:
```
1 H=pd.DataFrame(H)
2 H
```

Out[116]:

	models	Train	Test
0	Log	79.43	79.28
1	KNN	99.28	60.94
2	SVC	74.33	57.24
3	R1	99.82	80.02
4	AD	80.11	78.75

In [117]:
```
plt.figure(figsize=(10,6))
plt.bar(H['models'],H['Train']*100,align='center',width=0.4,label='TRAI
plt.bar(H['models'],H['Test']*100,align='edge',width=0.3,label='TESTIN
plt.grid()
plt.legend()
plt.xlabel('MODELS')
plt.ylabel('ACCURACY')
plt.show()
```



In [75]:
```
#from above plot,Logistic Regression is working good
```

In [76]:
```
from sklearn.naive_bayes import GaussianNB  , BernoulliNB  ,ComplementN

Gn=GaussianNB()
Gn.fit(X_train,y_train)
```

Out[76]:
```
GaussianNB()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [77]:
```
models={'Gaussian':GaussianNB(),
        'Bernoulian':BernoulliNB(),
        'Complement':ComplementNB(),
        'Categorical':CategoricalNB()}
```

In [78]:
```
models.keys()
```

Out[78]:
```
dict_keys(['Gaussian', 'Bernoulian', 'Complement', 'Categorical'])
```

In [79]:
```
1 models.values()
```

Out[79]:
```
dict_values([GaussianNB(), BernoulliNB(), ComplementNB(), CategoricalNB
()])
```

In [80]:
```
1 from sklearn.metrics import classification_report,accuracy_score,confus
```

In [81]:

```
Data1=[]

for k,l in models.items():
    l.fit(X_train,y_train)
    TR=l.score(X_train,y_train)
    TE=l.score(X_test,y_test)
    pred=l.predict(X_test)


    Data1.append([k,TR,TE])
    print(k.upper())
    print(classification_report(y_test,pred))
    print(confusion_matrix(y_test,pred))
    print('__'*40)
```

```
GAUSSIAN
              precision    recall  f1-score   support

           0       0.72      0.93      0.81       927
           1       0.90      0.65      0.76       965

    accuracy                           0.79      1892
   macro avg       0.81      0.79      0.78      1892
weighted avg       0.81      0.79      0.78      1892

[[859  68]
 [336 629]]
```

```
BERNOULIAN
              precision    recall  f1-score   support

           0       0.72      0.93      0.81       927
           1       0.90      0.65      0.76       965

    accuracy                           0.79      1892
   macro avg       0.81      0.79      0.78      1892
weighted avg       0.81      0.79      0.78      1892

[[859  68]
 [336 629]]
```

```
COMPLEMENT
              precision    recall  f1-score   support

           0       0.72      0.92      0.81       927
           1       0.89      0.65      0.76       965

    accuracy                           0.78      1892
   macro avg       0.81      0.79      0.78      1892
weighted avg       0.81      0.78      0.78      1892

[[852  75]
 [334 631]]
```

```
CATEGORICAL
              precision    recall  f1-score   support

           0       0.72      0.93      0.81       927
           1       0.90      0.65      0.76       965

    accuracy                           0.79      1892
   macro avg       0.81      0.79      0.78      1892
weighted avg       0.81      0.79      0.78      1892

[[859  68]
 [336 629]]
```

In [82]:
```
Data1
```

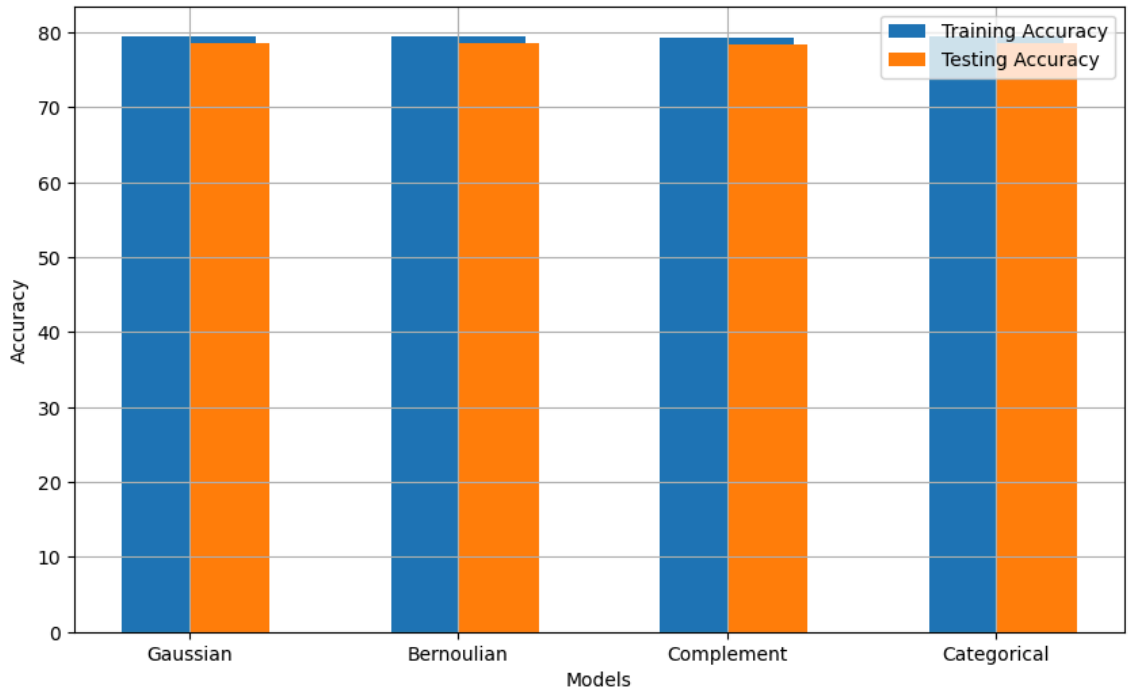
Out[82]:
```
[['Gaussian', 0.7949302193107377, 0.7864693446088795],
 ['Bernoulian', 0.7949302193107377, 0.7864693446088795],
 ['Complement', 0.7932213044716605, 0.7838266384778013],
 ['Categorical', 0.7952150384505838, 0.7864693446088795]]
```

In [83]:
```
R=pd.DataFrame(Data1,columns=('model name','Train','Test'))
R
```

Out[83]:

	model name	Train	Test
0	Gaussian	0.794930	0.786469
1	Bernoulian	0.794930	0.786469
2	Complement	0.793221	0.783827
3	Categorical	0.795215	0.786469

In [84]:
```
plt.figure(figsize=(10,6))
plt.bar(R['model name'],R['Train']*100,align='center',width=0.5,label='
plt.bar(R['model name'],R['Test']*100,align='edge',width=0.3,label='Tes
plt.grid()
plt.legend()
plt.xlabel('Models')
plt.ylabel('Accuracy')
plt.show()
```



#ANN

In [85]:
```
1 !pip install tensorflow
```

```
Defaulting to user installation because normal site-packages is not writea
ble
Requirement already satisfied: tensorflow in c:\users\admin\appdata\roamin
g\python\python311\site-packages (2.17.0)
Requirement already satisfied: tensorflow-intel==2.17.0 in c:\users\admin
\appdata\roaming\python\python311\site-packages (from tensorflow) (2.17.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\admin\appdata\ro
aming\python\python311\site-packages (from tensorflow-intel==2.17.0->tenso
rflow) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from tensorflow-intel==2.17.0->te
nsorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in c:\users\admin\appd
ata\roaming\python\python311\site-packages (from tensorflow-intel==2.17.0-
>tensorflow) (24.3.25)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in c:\u
sers\admin\appdata\roaming\python\python311\site-packages (from tensorflow
-intel==2.17.0->tensorflow) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\admin\appda
ta\roaming\python\python311\site-packages (from tensorflow-intel==2.17.0->
tensorflow) (0.2.0)
Requirement already satisfied: h5py>=3.10.0 in c:\users\admin\appdata\roam
ing\python\python311\site-packages (from tensorflow-intel==2.17.0->tensorf
low) (3.11.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from tensorflow-intel==2.17.0->te
nsorflow) (18.1.1)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.3.1 in c:\users\admin\a
ppdata\roaming\python\python311\site-packages (from tensorflow-intel==2.1
7.0->tensorflow) (0.4.1)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from tensorflow-intel==2.17.0->te
nsorflow) (3.4.0)
Requirement already satisfied: packaging in c:\programdata\anaconda3\lib\s
ite-packages (from tensorflow-intel==2.17.0->tensorflow) (23.1)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.
3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\programdata\anaconda3\lib\sit
e-packages (from tensorflow-intel==2.17.0->tensorflow) (3.20.3)
Requirement already satisfied: requests<3,>=2.21.0 in c:\programdata\anaco
nda3\lib\site-packages (from tensorflow-intel==2.17.0->tensorflow) (2.31.
0)
Requirement already satisfied: setuptools in c:\programdata\anaconda3\lib
\site-packages (from tensorflow-intel==2.17.0->tensorflow) (68.2.2)
Requirement already satisfied: six>=1.12.0 in c:\programdata\anaconda3\lib
\site-packages (from tensorflow-intel==2.17.0->tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\admin\appdata
\roaming\python\python311\site-packages (from tensorflow-intel==2.17.0->te
nsorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\programdata
\anaconda3\lib\site-packages (from tensorflow-intel==2.17.0->tensorflow)
(4.9.0)
Requirement already satisfied: wrapt>=1.11.0 in c:\programdata\anaconda3\l
ib\site-packages (from tensorflow-intel==2.17.0->tensorflow) (1.14.1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\admin\appda
ta\roaming\python\python311\site-packages (from tensorflow-intel==2.17.0->
tensorflow) (1.66.1)
Requirement already satisfied: tensorboard<2.18,>=2.17 in c:\users\admin\a
ppdata\roaming\python\python311\site-packages (from tensorflow-intel==2.1
7.0->tensorflow) (2.17.1)
Requirement already satisfied: keras>=3.2.0 in c:\users\admin\appdata\roam
ing\python\python311\site-packages (from tensorflow-intel==2.17.0->tensorf
```

```
low) (3.5.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
c:\users\admin\appdata\roaming\python\python311\site-packages (from tensor
flow-intel==2.17.0->tensorflow) (0.31.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in c:\programdata\anac
onda3\lib\site-packages (from tensorflow-intel==2.17.0->tensorflow) (1.26.
4)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\programdata\anacon
da3\lib\site-packages (from astunparse>=1.6.0->tensorflow-intel==2.17.0->t
ensorflow) (0.41.2)
Requirement already satisfied: rich in c:\programdata\anaconda3\lib\site-p
ackages (from keras>=3.2.0->tensorflow-intel==2.17.0->tensorflow) (13.3.5)
Requirement already satisfied: namex in c:\users\admin\appdata\roaming\pyt
hon\python311\site-packages (from keras>=3.2.0->tensorflow-intel==2.17.0->
tensorflow) (0.0.8)
Requirement already satisfied: optree in c:\users\admin\appdata\roaming\py
thon\python311\site-packages (from keras>=3.2.0->tensorflow-intel==2.17.0-
>tensorflow) (0.12.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\programdata
\anaconda3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==
2.17.0->tensorflow) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\li
b\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.17.0->tenso
rflow) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anacon
da3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.17.0-
>tensorflow) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anacon
da3\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.17.0-
>tensorflow) (2024.2.2)
Requirement already satisfied: markdown>=2.6.8 in c:\programdata\anaconda3
\lib\site-packages (from tensorboard<2.18,>=2.17->tensorflow-intel==2.17.0
->tensorflow) (3.4.1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in
c:\users\admin\appdata\roaming\python\python311\site-packages (from tensor
board<2.18,>=2.17->tensorflow-intel==2.17.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\programdata\anaconda3
\lib\site-packages (from tensorboard<2.18,>=2.17->tensorflow-intel==2.17.0
->tensorflow) (2.2.3)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\programdata\anacond
a3\lib\site-packages (from werkzeug>=1.0.1->tensorboard<2.18,>=2.17->tenso
rflow-intel==2.17.0->tensorflow) (2.1.3)
Requirement already satisfied: markdown-it-py<3.0.0,>=2.2.0 in c:\programd
ata\anaconda3\lib\site-packages (from rich->keras>=3.2.0->tensorflow-intel
==2.17.0->tensorflow) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in c:\programdata\a
naconda3\lib\site-packages (from rich->keras>=3.2.0->tensorflow-intel==2.1
7.0->tensorflow) (2.15.1)
Requirement already satisfied: mdurl~=0.1 in c:\programdata\anaconda3\lib
\site-packages (from markdown-it-py<3.0.0,>=2.2.0->rich->keras>=3.2.0->ten
sorflow-intel==2.17.0->tensorflow) (0.1.0)
```

In [86]:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Dropout
```

In [87]:
```
model= Sequential([

    Dense(60,input_shape=(X_train.shape[1],),activation='relu'),#input
    Dense(30,activation='relu'),#hidden layer
    Dropout(0.3),
    Dense(30,activation='relu'),
    Dense(30,activation='relu'),#hidden layer
    Dense(1,activation='sigmoid'),#output layer
])
```

```
C:\Users\Admin\AppData\Roaming\Python\Python311\site-packages\keras\src\layers\core\dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

In [88]:
```
model.compile(
    loss='binary_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```

In [89]:
```
1 model.fit(X_train,y_train,epochs=25)
```

```
Epoch 1/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.5281 - loss: 0.7537
Epoch 2/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.5600 - loss: 0.6923
Epoch 3/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.6273 - loss: 0.6602
Epoch 4/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step - accuracy: 0.7022 - loss: 0.5975
Epoch 5/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7449 - loss: 0.5085
Epoch 6/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7866 - loss: 0.4870
Epoch 7/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7767 - loss: 0.4833
Epoch 8/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7793 - loss: 0.4606
Epoch 9/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7789 - loss: 0.4660
Epoch 10/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7957 - loss: 0.4342
Epoch 11/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7865 - loss: 0.4372
Epoch 12/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7899 - loss: 0.4582
Epoch 13/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7914 - loss: 0.4451
Epoch 14/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7882 - loss: 0.4495
Epoch 15/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7939 - loss: 0.4364
Epoch 16/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7891 - loss: 0.4378
Epoch 17/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7994 - loss: 0.4172
Epoch 18/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.8012 - loss: 0.4237
Epoch 19/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.7964 - loss: 0.4227
Epoch 20/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7925 - loss: 0.4370
Epoch 21/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7914 - loss: 0.4388
Epoch 22/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7950 - loss: 0.4263
Epoch 23/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.8016 - loss: 0.4328
Epoch 24/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.7975 - loss: 0.4266
Epoch 25/25
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.7869 - loss: 0.4444
```

Out[89]:
```
<keras.src.callbacks.history.History at 0x211d416f990>
```

In [90]:
```
pred=model.predict(X_test)
pred
```

```
60/60 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
```

Out[90]:
```
array([[0.22438397],
       [0.43660703],
       [0.92852753],
       ...,
       [0.01154943],
       [0.40716162],
       [0.03224894]], dtype=float32)
```

In [91]:
```
pred1=[]
for i in pred:
    if i<=0.5:
        pred1.append(0)
    else:
        pred1.append(1)
```

In [92]:
```
pred1
```

Out[92]:
```
[0,
 0,
 1,
 0,
 0,
 1,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
```

In [93]:
```
from sklearn.metrics import confusion_matrix,accuracy_score,classificat
confusion_matrix(y_test,pred1)
```

Out[93]:
```
array([[849,  78],
       [332, 633]], dtype=int64)
```

In [94]:
```
accuracy_score(y_test,pred1)
```

Out[94]:
```
0.7832980972515856
```

In [95]:
```
print(classification_report(y_test,pred1))
```

```
              precision    recall  f1-score   support

           0       0.72      0.92      0.81       927
           1       0.89      0.66      0.76       965

    accuracy                           0.78      1892
   macro avg       0.80      0.79      0.78      1892
weighted avg       0.81      0.78      0.78      1892
```

In [96]:
```
model1=Sequential([
    Dense(50,input_shape=(X_train.shape[1],),activation="elu"),
    Dense(25,activation="elu"),
    Dense(20,activation="elu"),
    Dropout(0.1),
    Dense(30,activation="elu"),
    Dense(1,activation="sigmoid"),
])
```

```
C:\Users\Admin\AppData\Roaming\Python\Python311\site-packages\keras\src\layers\core\dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

In [97]:
```
model.compile(
    loss='binary_crossentropy',
    optimizer='adagrad',
    metrics=['accuracy']
)
```

In [98]:
```
pred2=model1.predict(X_test)
pred2
```

```
60/60 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
```

Out[98]:
```
array([[0.864488  ],
       [0.87920016],
       [0.7097148 ],
       ...,
       [0.7411331 ],
       [0.8166284 ],
       [0.748945  ]], dtype=float32)
```

In [99]:
```
pred3=[]
for i in pred:
    if i<=0.5:
        pred3.append(0)
    else:
        pred3.append(1)
```

In [100]:
```
1 pred3
```

Out[100]:
```
[0,
 0,
 1,
 0,
 0,
 1,
 0,
 0,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
```

In [101]:
```
1 from sklearn.metrics import confusion_matrix,accuracy_score,classificat
```

In [102]:
```
1 confusion_matrix(y_test,pred3)
```

Out[102]:
```
array([[849,  78],
       [332, 633]], dtype=int64)
```

In [103]:
```
1 accuracy_score(y_test,pred3)
```

Out[103]:
```
0.7832980972515856
```

In [104]:
```
1 print(classification_report(y_test,pred3))
```

```
              precision    recall  f1-score   support

           0       0.72      0.92      0.81       927
           1       0.89      0.66      0.76       965

    accuracy                           0.78      1892
   macro avg       0.80      0.79      0.78      1892
weighted avg       0.81      0.78      0.78      1892
```

#Project Report

#The goal is to predict the efficiency of smart home devices based on the available features like EnergyConsumption,UsageHoursPerDay, and DeviceAgeMonths.
#This helps identify underperforming devices and optimize their use,leading to better energy consumption and user satisfaction.

#Discussion: #•Insights: The analysis shows that older devices with higher energy consumption and more malfunction incidents tend to be less efficient. #•Model Comparison: Both ML models and ANN performed well, but the ANN showed better generalization on unseen data. #The dataset could benefit from additional features such as device maintenance history,Brands or more granular usage data.

#Conclusion : #This project successfully developed predictive models to estimate smart home device efficiency. #Using machine learning and ANN, we demonstrated how energy

In [ ]:
```
1
```