In [ ]:
```
# Name :- Sarthak Pagar
# Roll No. :- 40
# Class :- TE(IT)
# Practical 5 :- Perform the following operations using Python on the Air qu
                 # b. Data integration
                 # c. Data transformation
                 # d. Error correcting
                 # e. Data model building
```

In [30]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix, classificatior
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
```

In [32]:
```
df = pd.read_csv("Heart.csv")
```

In [33]:
```
df.head()
```

Out[33]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa
0	63	1	3	145	233	1	0	150	0	2.3	0	0
1	37	1	2	130	250	0	1	187	0	3.5	0	0
2	41	0	1	130	204	0	0	172	0	1.4	2	0
3	56	1	1	120	236	0	1	178	0	0.8	2	0
4	57	0	0	120	354	0	1	163	1	0.6	2	0

In [35]:
```
# a) Data Cleaning
df = df.drop_duplicates()
```

In [36]:
```
# Count ,min,max ,etc of each column
df.describe()
```

Out[36]:

	age	sex	cp	trtbps	chol	fbs
count	302.00000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	54.42053	0.682119	0.963576	131.602649	246.500000	0.149007
std	9.04797	0.466426	1.032044	17.563394	51.753489	0.356686
min	29.00000	0.000000	0.000000	94.000000	126.000000	0.000000
25%	48.00000	0.000000	0.000000	120.000000	211.000000	0.000000
50%	55.50000	1.000000	1.000000	130.000000	240.500000	0.000000
75%	61.00000	1.000000	2.000000	140.000000	274.750000	0.000000
max	77.00000	1.000000	3.000000	200.000000	564.000000	1.000000

In [37]:

```
# Information about each column data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 302 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       302 non-null    int64
 1   sex       302 non-null    int64
 2   cp        302 non-null    int64
 3   trtbps    302 non-null    int64
 4   chol      302 non-null    int64
 5   fbs       302 non-null    int64
 6   restecg   302 non-null    int64
 7   thalachh  302 non-null    int64
 8   exng      302 non-null    int64
 9   oldpeak   302 non-null    float64
 10  slp       302 non-null    int64
 11  caa       302 non-null    int64
 12  thall     302 non-null    int64
 13  output    302 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 35.4 KB
```

In [38]:

```
#Finding null values in each column
df.isna().sum()
```

Out[38]:
```
age         0
sex         0
cp          0
trtbps      0
chol        0
fbs         0
restecg     0
thalachh    0
exng        0
oldpeak     0
slp         0
caa         0
thall       0
output      0
dtype: int64
```

In [39]:
```
# b) Data Integration
df.fbs.unique()
```

Out[39]:
```
array([1, 0], dtype=int64)
```

In [40]:
```
df1 = df[['age','cp','chol','thalachh']]
```

In [41]:
```
df2 = df[['exng','slp','output']]
```

In [42]:
```
merging=pd.concat([df1,df2],axis=1)
merging
```

Out[42]:

	age	cp	chol	thalachh	exng	slp	output
0	63	3	233	150	0	0	1
1	37	2	250	187	0	0	1
2	41	1	204	172	0	2	1
3	56	1	236	178	0	2	1
4	57	0	354	163	1	2	1
...	...	...	...	...	...	...	...
298	57	0	241	123	1	1	0
299	45	3	264	132	0	1	0
300	68	0	193	141	0	1	0
301	57	0	131	115	1	1	0
302	57	1	236	174	0	1	0

302 rows × 7 columns

In [43]:
```
# d) Error Correcting
df.columns
```

Out[43]:
```
Index(['age', 'sex', 'cp', 'trtbps', 'chol', 'fbs', 'restecg', 'thalachh',
       'exng', 'oldpeak', 'slp', 'caa', 'thall', 'output'],
      dtype='object')
```

In [45]:
```
# Function to Remove Outliers
def remove_outliers(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    threshold = 1.5 * IQR
    outlier_mask = (column < Q1 - threshold) | (column > Q3 + threshold)
    return column[~outlier_mask]
```

In [47]:
```
# Remove outliers for each column using a loop
col_name = ['cp','thalachh','exng','oldpeak','slp','caa']
for col in col_name:
    df[col] = remove_outliers(df[col])
```

In [48]:
```
# Dropping Null Values after Outlier Removal
df = df.dropna()
```

In [49]:
```
# Dropping Unnecessary Column
df = df.drop('fbs',axis=1)
```

In [50]:
```
# splitting data using train test split
x = df[['cp','thalachh','exng','oldpeak','slp','caa']]
y = df.output
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_stat
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[50]:
```
((220, 6), (55, 6), (220,), (55,))
```

In [51]:
```
# c) Data transformation
from sklearn.preprocessing import StandardScaler
```

In [52]:
```
scaler = StandardScaler()
```

In [53]:
```
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

In [54]:
```
# e) Data model building
model = LogisticRegression()
model.fit(x_train_scaled, y_train)
```

Out[54]:
```
▾ LogisticRegression
LogisticRegression()
```

In [55]:
```
# Make predictions on the test set
y_pred = model.predict(x_test_scaled)
```

In [57]:
```
# Evaluate the model's accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Logistic Regression Accuracy:", accuracy)
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Logistic Regression Accuracy: 0.8363636363636363
Classification Report:
               precision    recall  f1-score   support

           0       0.91      0.74      0.82        27
           1       0.79      0.93      0.85        28

    accuracy                           0.84        55
   macro avg       0.85      0.83      0.83        55
weighted avg       0.85      0.84      0.83        55
```

In [60]:
```
# Classification model using Decision Tree
from sklearn.tree import DecisionTreeClassifier
dtc=DecisionTreeClassifier(criterion='entropy')
dtc.fit(x_train_scaled,y_train)
y_pred_dtc=dtc.predict(x_test_scaled)
```

In [61]:
```
print("Decision Tree Accuracy:", accuracy_score(y_test, y_pred_dtc))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_dtc))
print("Classification Report:\n", classification_report(y_test, y_pred_dtc))
```

```
Decision Tree Accuracy: 0.7818181818181819
Confusion Matrix:
 [[20  7]
 [ 5 23]]
Classification Report:
               precision    recall  f1-score   support

           0       0.80      0.74      0.77        27
           1       0.77      0.82      0.79        28

    accuracy                           0.78        55
   macro avg       0.78      0.78      0.78        55
weighted avg       0.78      0.78      0.78        55
```

In [ ]: