In [1]:
```
#Experiment no.10
```

In [2]:
```
#Aim :To perform and analysis of SVM
```

In [3]:
```
#Name:Janvi R.Kale
#Roll no.:29
#sec:A
#sub:ET 1
#date:30-09-2025
```

Importing the Libraries

In [4]:
```
import pandas as pd
import numpy as np
```

Data acquisitionuing Pandas

In [5]:
```
import os
```

In [6]:
```
os.getcwd()
```

Out[6]:
```
'C:\\Users\\This PC'
```

In [7]:
```
os.chdir('C:\\Users\\This PC\\OneDrive\\Desktop\\dss practical datasets')
```

In [8]:
```
data=pd.read_csv("heart.csv")
```

In [9]:
```
data.head()
```

Out[9]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	targ
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	

In [10]:
```
data.tail()
```

Out[10]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	t
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	

In [11]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

In [12]:
```
data.describe()
```

Out[12]:

	age	sex	cp	trestbps	chol	fbs	res
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.00
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.52
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.52
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.00
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.00
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.00
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.00
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.00

In [13]:
```
data.shape
```

Out[13]:
```
(1025, 14)
```

In [14]:
```
data.size
```

Out[14]:
```
14350
```

In [15]:
```
data.ndim
```

Out[15]:
```
2
```

Data preprocessing _ data cleaning _ missing value treatment

In [16]:
```
# check Missing Value by record

data.isna()
```

Out[16]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	
0	False	False	False	False	False	False	False	False	False	False	False	Fa
1	False	False	False	False	False	False	False	False	False	False	False	Fa
2	False	False	False	False	False	False	False	False	False	False	False	Fa
3	False	False	False	False	False	False	False	False	False	False	False	Fa
4	False	False	False	False	False	False	False	False	False	False	False	Fa
...	...	...	...	...	...	...	...	...	...	...	...	
1020	False	False	False	False	False	False	False	False	False	False	False	Fa
1021	False	False	False	False	False	False	False	False	False	False	False	Fa
1022	False	False	False	False	False	False	False	False	False	False	False	Fa
1023	False	False	False	False	False	False	False	False	False	False	False	Fa
1024	False	False	False	False	False	False	False	False	False	False	False	Fa

1025 rows × 14 columns

In [17]:
```
data.isna().any()
```

Out[17]:
```
age         False
sex         False
cp          False
trestbps    False
chol        False
fbs         False
restecg     False
thalach     False
exang       False
oldpeak     False
slope       False
ca          False
thal        False
target      False
dtype: bool
```

In [18]:
```
data.isna().sum()
```

Out[18]:
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

Independent and Dependent Variables

In [19]:
```
x=data.drop("target", axis=1)
y=data["target"]
```

Splitting of DataSet into train and Test

In [20]:
```
#splitting the data into training and testing data sets
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2 ,random_st
```

Support Vector Classifier / Machine (SVC/SVM)

In [21]:
```
from sklearn import svm
svm=svm.SVC()  #svc=svm
svm.fit(x_train, y_train)
from sklearn.metrics import accuracy_score
```

In [22]:
```
y_pred3=svm.predict(x_test)
```

In [23]:
```
accuracy_score (y_test,y_pred3)
```

Out[23]:
```
0.6829268292682927
```

🔷 Conclusion: In this practical, we implemented the Support Vector Machine (SVM) algorithm for classification. We learned how SVM separates data using a hyperplane and maximizes the margin between classes for better accuracy. This practical enhanced our understanding of kernel functions and how SVM effectively handles both linear and non-linear data in machine learning.

In [ ]:
```

```