In [27]:

```
import pandas as pd
import numpy as np
```

In [28]:

```
data=pd.read_csv("Software.csv")
```

In [29]:

```
data.head()
```

Out[29]:

	id	PERCENT_PUB_DATA	ACCESS_TO_PUB_DATA	COUPLING_BETWEEN_OBJECTS	DEPTH	LACK_OF_COHESION_OF_METHODS	NUM_OF_CHILDREN
0	1	0	0	24	4	100	0
1	2	0	0	19	4	100	0
2	3	100	0	13	1	88	0
3	4	0	0	21	4	100	0
4	5	5	0	17	2	90	0

5 rows × 96 columns

In [30]:

```
data.shape
```

Out[30]:

```
(145, 96)
```

In [31]:

```
data.isnull().sum().head(96)
```

Out[31]:

```
id                          0
PERCENT_PUB_DATA            0
ACCESS_TO_PUB_DATA          0
COUPLING_BETWEEN_OBJECTS    0
DEPTH                       0
                           ..
sumNUM_OPERATORS            0
sumNUM_UNIQUE_OPERANDS      0
sumNUM_UNIQUE_OPERATORS     0
sumLOC_TOTAL                0
DL                          0
Length: 96, dtype: int64
```

In [32]:

```
print(data.columns.values)
```

```
['id' 'PERCENT_PUB_DATA' 'ACCESS_TO_PUB_DATA' 'COUPLING_BETWEEN_OBJECTS'
 'DEPTH' 'LACK_OF_COHESION_OF_METHODS' 'NUM_OF_CHILDREN' 'DEP_ON_CHILD'
 'FAN_IN' 'RESPONSE_FOR_CLASS' 'WEIGHTED_METHODS_PER_CLASS' 'minLOC_BLANK'
 'minBRANCH_COUNT' 'minLOC_CODE_AND_COMMENT' 'minLOC_COMMENTS'
 'minCYCLOMATIC_COMPLEXITY' 'minDESIGN_COMPLEXITY'
 'minESSENTIAL_COMPLEXITY' 'minLOC_EXECUTABLE' 'minHALSTEAD_CONTENT'
 'minHALSTEAD_DIFFICULTY' 'minHALSTEAD_EFFORT' 'minHALSTEAD_ERROR_EST'
 'minHALSTEAD_LENGTH' 'minHALSTEAD_LEVEL' 'minHALSTEAD_PROG_TIME'
 'minHALSTEAD_VOLUME' 'minNUM_OPERANDS' 'minNUM_OPERATORS'
 'minNUM_UNIQUE_OPERANDS' 'minNUM_UNIQUE_OPERATORS' 'minLOC_TOTAL'
 'maxLOC_BLANK' 'maxBRANCH_COUNT' 'maxLOC_CODE_AND_COMMENT'
 'maxLOC_COMMENTS' 'maxCYCLOMATIC_COMPLEXITY' 'maxDESIGN_COMPLEXITY'
 'maxESSENTIAL_COMPLEXITY' 'maxLOC_EXECUTABLE' 'maxHALSTEAD_CONTENT'
 'maxHALSTEAD_DIFFICULTY' 'maxHALSTEAD_EFFORT' 'maxHALSTEAD_ERROR_EST'
 'maxHALSTEAD_LENGTH' 'maxHALSTEAD_LEVEL' 'maxHALSTEAD_PROG_TIME'
 'maxHALSTEAD_VOLUME' 'maxNUM_OPERANDS' 'maxNUM_OPERATORS'
 'maxNUM_UNIQUE_OPERANDS' 'maxNUM_UNIQUE_OPERATORS' 'maxLOC_TOTAL'
 'avgLOC_BLANK' 'avgBRANCH_COUNT' 'avgLOC_CODE_AND_COMMENT'
 'avgLOC_COMMENTS' 'avgCYCLOMATIC_COMPLEXITY' 'avgDESIGN_COMPLEXITY'
 'avgESSENTIAL_COMPLEXITY' 'avgLOC_EXECUTABLE' 'avgHALSTEAD_CONTENT'
 'avgHALSTEAD_DIFFICULTY' 'avgHALSTEAD_EFFORT' 'avgHALSTEAD_ERROR_EST'
 'avgHALSTEAD_LENGTH' 'avgHALSTEAD_LEVEL' 'avgHALSTEAD_PROG_TIME'
 'avgHALSTEAD_VOLUME' 'avgNUM_OPERANDS' 'avgNUM_OPERATORS'
 'avgNUM_UNIQUE_OPERANDS' 'avgNUM_UNIQUE_OPERATORS' 'avgLOC_TOTAL'
 'sumLOC_BLANK' 'sumBRANCH_COUNT' 'sumLOC_CODE_AND_COMMENT'
 'sumLOC_COMMENTS' 'sumCYCLOMATIC_COMPLEXITY' 'sumDESIGN_COMPLEXITY'
 'sumESSENTIAL_COMPLEXITY' 'sumLOC_EXECUTABLE' 'sumHALSTEAD_CONTENT'
 'sumHALSTEAD_DIFFICULTY' 'sumHALSTEAD_EFFORT' 'sumHALSTEAD_ERROR_EST'
 'sumHALSTEAD_LENGTH' 'sumHALSTEAD_LEVEL' 'sumHALSTEAD_PROG_TIME'
 'sumHALSTEAD_VOLUME' 'sumNUM_OPERANDS' 'sumNUM_OPERATORS'
 'sumNUM_UNIQUE_OPERANDS' 'sumNUM_UNIQUE_OPERATORS' 'sumLOC_TOTAL' 'DL']
```

In [33]:

```
data["DL"].value_counts()
```

Out[33]:

```
FALSE    85
_TRUE    60
Name: DL, dtype: int64
```

In [34]:

```
x=data.iloc[:,1:94]
y=data.iloc[:,95]
```

In [35]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 0)
```

In [36]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

In [37]:

```
from sklearn.svm import SVC
classifier = SVC()
classifier.fit(x_train, y_train)
```

Out[37]:

```
SVC
SVC()
```

In [38]:

```
from sklearn.metrics import accuracy_score
pred_test=classifier.predict(x_test)
pred_train=classifier.predict(x_train)
print(accuracy_score(pred_test,y_test))
```

```
0.7027027027027027
```