In [ ]:

```
problem statement : Iris species prediction
```

In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [4]:

```
df=pd.read_csv(r"C:\Users\Irfan\Downloads\Iris.csv")
df
```

Out[4]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

target is available : supervised descrete:classificaiton

In [6]:

```
df.describe().T
```

Out[6]:

	count	mean	std	min	25%	50%	75%	max
SepalLengthCm	150.0	5.843333	0.828066	4.3	5.1	5.80	6.4	7.9
SepalWidthCm	150.0	3.054000	0.433594	2.0	2.8	3.00	3.3	4.4
PetalLengthCm	150.0	3.758667	1.764420	1.0	1.6	4.35	5.1	6.9
PetalWidthCm	150.0	1.198667	0.763161	0.1	0.3	1.30	1.8	2.5

In [8]:

```
from sympy.combinatorics import Permutation, Cycle
from sympy.combinatorics.subsets import Subset
```

In [15]:

```
X=df[['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm']]
y=df['Species']
```

In [27]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,train_size=0.70,random_state=0)
print(X_train.shape,y_train.shape)
print(X_test.shape,y_test.shape)
```

```
(105, 4) (105,)
(45, 4) (45,)
```

In [28]:

```
from sklearn.preprocessing import StandardScaler
scalar=StandardScaler()
X_train_transformed=scalar.fit_transform(X_train)
print(X_train_transformed.shape)
X_test_transformed=scalar.transform(X_test)
print(X_test_transformed.shape)
```

```
(105, 4)
(45, 4)
```

In [29]:

```
from sklearn.linear_model import LogisticRegression
L_regression=LogisticRegression()
L_regression.fit(X_train_transformed,y_train)
y_test_pred=L_regression.predict(X_test_transformed)
from sklearn import metrics
metrics.accuracy_score(y_test,y_test_pred)
```

Out[29]:

```
0.9777777777777777
```

In [ ]:

```

```