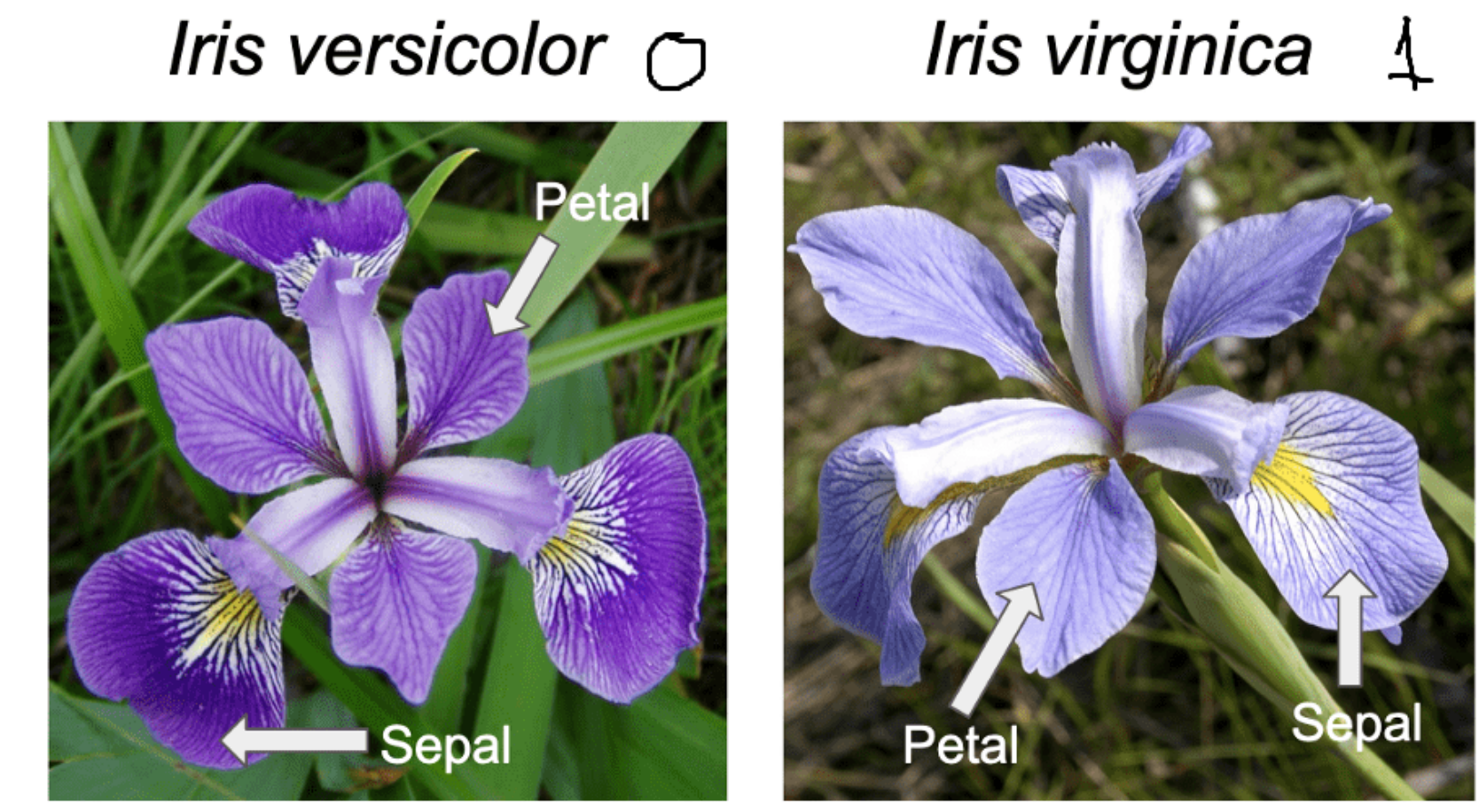
Logistic Regression

In [3]:
```
# import library
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```



In [4]:
```
# Load The Data set
url="https://raw.githubusercontent.com/svkarthik86/Meachine_Learning/main/data/modifiedIris2Classes.csv"
iris_df=pd.read_csv(url)
iris_df
```

Out[4]:

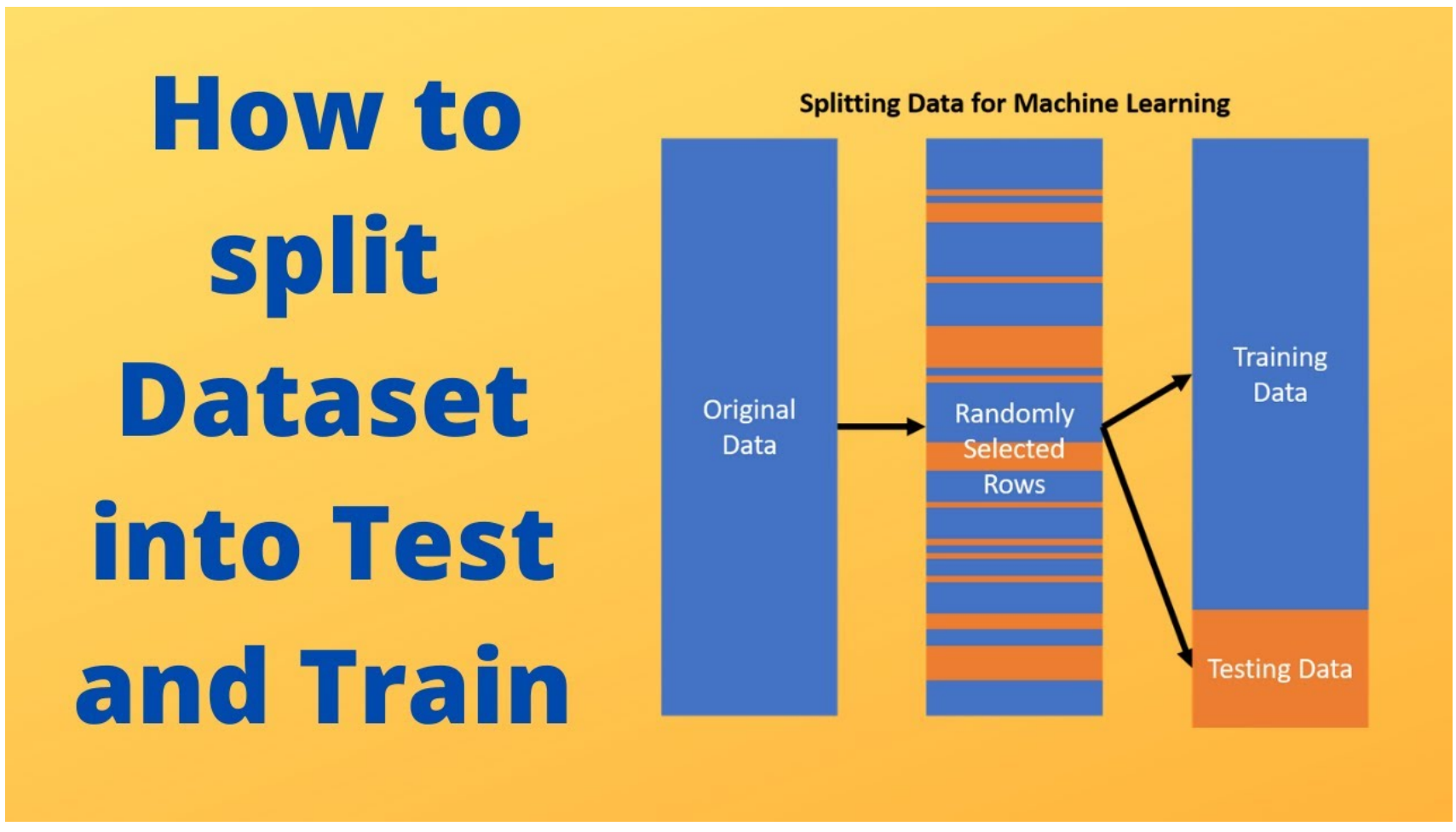
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	7.0	3.2	4.7	1.4	0
1	6.4	3.2	4.5	1.5	0
2	6.9	3.1	4.9	1.5	0
3	5.5	2.3	4.0	1.3	0
4	6.5	2.8	4.6	1.5	0
...	...	...	...	...	...
95	6.7	3.0	5.2	2.3	1
96	6.3	2.5	5.0	1.9	1
97	6.5	3.0	5.2	2.0	1
98	6.2	3.4	5.4	2.3	1
99	5.9	3.0	5.1	1.8	1

100 rows × 5 columns

In [6]:
```
#Identify X and y
X=iris_df.iloc[:,:-1]
```

In [9]:
```
y=iris_df[['target']]
```

In [ ]:
```
#Splitting Data into Training and Test Sets
```



In [13]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,random_state=123,test_size=.25,stratify=y)
```

In [14]:
```
X_train
```

Out[14]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
92	5.8	2.7	5.1	1.9
35	6.0	3.4	4.5	1.6
83	6.3	2.8	5.1	1.5
96	6.3	2.5	5.0	1.9
52	7.1	3.0	5.9	2.1
...	...	...	...	...
97	6.5	3.0	5.2	2.0
79	7.2	3.0	5.8	1.6
27	6.7	3.0	5.0	1.7
55	7.6	3.0	6.6	2.1
3	5.5	2.3	4.0	1.3

75 rows × 4 columns

In [ ]:
```
# Preprocessing
```

Standardize the Data

Logistic Regression is effected by scale so you need to scale the features in the data before using Logistic Regresison. You can transform the data onto unit scale (mean = 0 and variance = 1) for better performance. Scikit-Learn's `StandardScaler` helps standardize the dataset's features. Note you fit on the training set and transform on the training and test set.

In [22]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
# Fit on training set only.
scaler.fit(X_train)
# Apply transform to both the training set and the test set.
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

In [24]:
```
X_test
```

Out[24]:
```
array([[-1.72335854, -0.52648494, -1.24014912, -0.65013384],
       [ 0.01050828, -1.09463415, -0.01308865, -0.4145781 ],
       [-1.25048577, -1.09463415, -1.11744307, -0.88568957],
       [-0.777613  , -0.52648494, -1.24014912, -1.12124531],
       [-0.93523726,  0.32573888, -0.87203098, -1.12124531],
       [-0.777613  , -0.81055954, -1.11744307, -1.12124531],
       [-1.09286151, -1.09463415, -1.24014912, -1.35680105],
       [-1.09286151,  0.32573888, -0.99473702, -0.88568957],
       [-0.14711597, -0.24241033, -0.13579469,  0.29208912],
       [ 0.01050828,  1.46203731,  0.84585369,  1.70542354],
       [-0.777613  , -0.52648494, -0.99473702, -1.59235679],
       [ 0.16813254,  0.89388809,  0.47773554,  1.4698678 ],
       [ 2.21724788, -0.81055954,  2.4410323 ,  1.4698678 ],
       [-2.19623131, -1.09463415, -0.50391283,  0.05653338],
       [ 0.95625383,  0.60981349,  0.23232345,  1.4698678 ],
       [ 0.79862957,  0.89388809,  1.21397183,  1.4698678 ],
       [-0.46236449,  0.04166428, -0.50391283, -0.4145781 ],
       [-0.93523726, -1.09463415,  0.1096174 ,  0.76320059],
       [ 0.95625383,  0.60981349, -0.01308865, -0.4145781 ],
       [ 2.21724788, -0.24241033,  2.19562021,  0.76320059],
       [ 1.74437511, -0.24241033,  1.45938392,  0.52764485],
       [-1.40811003,  0.32573888, -0.50391283, -0.4145781 ],
       [ 0.3257568 ,  0.32573888,  0.72314764,  0.29208912],
       [-0.30474023,  0.32573888, -0.01308865,  0.29208912],
       [-0.46236449, -0.52648494,  0.23232345, -0.17902236]])
```

In [ ]:
```

```