In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Load Iris Dataset

In [2]:
```
from sklearn.datasets import load_iris
```

In [3]:
```
iris = load_iris()
data = pd.DataFrame(iris.data, columns = iris.feature_names)
target = pd.DataFrame(iris.target, columns=['Target'])
df = pd.concat([data, target], axis= 1)
```

In [5]:
```
df.head(10)
```

Out[5]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
5	5.4	3.9	1.7	0.4	0
6	4.6	3.4	1.4	0.3	0
7	5.0	3.4	1.5	0.2	0
8	4.4	2.9	1.4	0.2	0
9	4.9	3.1	1.5	0.1	0

In [6]:
```
from sklearn.model_selection import train_test_split
```

In [12]:
```
X_train, X_test, y_train, y_test = train_test_split(df, target, test_size=
```

Initialize Logistic Regression

In [13]:
```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter= 1000)
```

In [14]:
```
# Fit the model on training dataset
```

In [15]:
```
model.fit(X_train,y_train)
```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\utils\validation.py:114
3: DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

Out[15]:
```
LogisticRegression(max_iter=1000)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [16]:
```
# Predict the Model on test data
```

In [18]:
```
predictions = model.predict(X_test)
```

In [19]:
```
# Calculate Accuracy
```

In [20]:
```
from sklearn.metrics import accuracy_score, classification_report, confusio
```

In [21]:
```
accuracy = accuracy_score(y_test, predictions)
print(f"Accuracy : {accuracy}")

print("Classification Report :")
print(classification_report(y_test, predictions))


print("Confusion Matrix :")
print(confusion_matrix(y_test, predictions))
```

```
Accuracy : 1.0
Classification Report :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00         8
           2       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30

Confusion Matrix :
[[13  0  0]
 [ 0  8  0]
 [ 0  0  9]]
```

In [ ]:

In [22]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
normalized_df = scaler.fit_transform(df)
```

In [23]:
```
from sklearn.linear_model import LogisticRegression
```

In [24]:
```
Normalize_model = LogisticRegression(max_iter=1000)
Normalize_model.fit(X_train, y_train)
```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\utils\validation.py:114
3: DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

Out[24]:
```
LogisticRegression(max_iter=1000)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [26]:
```
prediction = Normalize_model.predict(X_test)
```

In [27]:
```
# To Check Accuracy
```

In [32]:
```
from sklearn.metrics import accuracy_score, confusion_matrix, classificatio
```

In [33]:
```
Accuracy = accuracy_score(y_test, prediction)
print(f"Accuracy : {Accuracy}")

print("Confusion Matrix : ")
print(confusion_matrix(y_test, prediction))

print("Classification Report")
print(classification_report(y_test, prediction))
```

```
Accuracy : 1.0
Confusion Matrix : 
[[13  0  0]
 [ 0  8  0]
 [ 0  0  9]]
Classification Report
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00         8
           2       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

In [ ]: