In [ ]:
```
TASK - 3 IRIS FLOWER CLASSIFICATION
```

In [ ]:
```
CODSOFT
```

Importing Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score
```

Importing Dataset

In [62]:
```
df=pd.read_csv('IRIS.csv')
```

In [64]:
```
df.head()
```

Out[64]:

	sepal_length	sepal_width	petal_length	petal_width	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

In [65]:
```
df.head(10)
```

Out[65]:

	sepal_length	sepal_width	petal_length	petal_width	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
5	5.4	3.9	1.7	0.4	Iris-setosa
6	4.6	3.4	1.4	0.3	Iris-setosa
7	5.0	3.4	1.5	0.2	Iris-setosa
8	4.4	2.9	1.4	0.2	Iris-setosa
9	4.9	3.1	1.5	0.1	Iris-setosa

In [66]:
```
df.tail()
```

Out[66]:

	sepal_length	sepal_width	petal_length	petal_width	species
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

In [67]:
```
df.shape
```

Out[67]:
```
(150, 5)
```

In [68]:
```
df.isnull().sum()
```

Out[68]:
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

In [69]:
```
df.dtypes
```

Out[69]:
```
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
species          object
dtype: object
```

In [71]:
```
data=df.groupby('species')
```

In [72]:
```
data.head()
```

Out[72]:

	sepal_length	sepal_width	petal_length	petal_width	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
50	7.0	3.2	4.7	1.4	Iris-versicolor
51	6.4	3.2	4.5	1.5	Iris-versicolor
52	6.9	3.1	4.9	1.5	Iris-versicolor
53	5.5	2.3	4.0	1.3	Iris-versicolor
54	6.5	2.8	4.6	1.5	Iris-versicolor
100	6.3	3.3	6.0	2.5	Iris-virginica
101	5.8	2.7	5.1	1.9	Iris-virginica
102	7.1	3.0	5.9	2.1	Iris-virginica
103	6.3	2.9	5.6	1.8	Iris-virginica
104	6.5	3.0	5.8	2.2	Iris-virginica

In [73]:
```
df['species'].unique()
```

Out[73]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

In [74]:
```
df.info()
```

Out[74]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

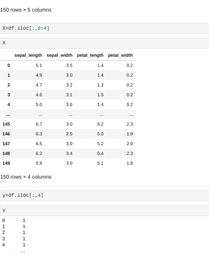
Visualizing The Dataset

In [79]:
```
plt.boxplot(df['sepal_length'])
```

Out[79]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x2bf88ffac20>,
  <matplotlib.lines.Line2D at 0x2bf88f72aa0>],
 'caps': [<matplotlib.lines.Line2D at 0x2bf899e036b0>,
  <matplotlib.lines.Line2D at 0x2bf899e03950>],
 'boxes': [<matplotlib.lines.Line2D at 0x2bf88ffa778>],
 'medians': [<matplotlib.lines.Line2D at 0x2bf899e72b0>],
 'fliers': [<matplotlib.lines.Line2D at 0x2bf899ef550>],
 'means': []}
```

In [82]:
```
plt.boxplot(df['sepal_width'])
```

Out[82]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x2bf8929a3d0>,
  <matplotlib.lines.Line2D at 0x2bf8929d3b0>],
 'caps': [<matplotlib.lines.Line2D at 0x2bf8929d9b0>,
  <matplotlib.lines.Line2D at 0x2bf8929dc50>],
 'boxes': [<matplotlib.lines.Line2D at 0x2bf8929237f0>],
 'medians': [<matplotlib.lines.Line2D at 0x2bf8892a8550>],
 'fliers': [<matplotlib.lines.Line2D at 0x2bf892ab7f0>],
 'means': []}
```



In [83]:
```
plt.boxplot(df['petal_length'])
```

Out[83]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x2bf892ca860>,
  <matplotlib.lines.Line2D at 0x2bf892cab00>],
 'caps': [<matplotlib.lines.Line2D at 0x2bf892cad80>,
  <matplotlib.lines.Line2D at 0x2bf892ae048>],
 'boxes': [<matplotlib.lines.Line2D at 0x2bf892c2cb0>],
 'medians': [<matplotlib.lines.Line2D at 0x2bf892ae2e0>],
 'fliers': [<matplotlib.lines.Line2D at 0x2bf892ae578>],
 'means': []}
```


In [84]:
```
sns.heatmap(df.corr())
```

Out[84]:
```
C:\Users\INDIA\AppData\Local\Temp\ipykernel_10180\583507773.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr())
<Axes: >
```


Data Preparation

In [86]:
```
sp={'Iris-setosa':1,'Iris-versicolor':2,'Iris-virginica':3}
```

In [88]:
```
df.species=[sp[i] for i in df.species]
```

In [89]:
```
df
```

Out[89]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	1
1	4.9	3.0	1.4	0.2	1
2	4.7	3.2	1.3	0.2	1
3	4.6	3.1	1.5	0.2	1
4	5.0	3.6	1.4	0.2	1
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	3
146	6.3	2.5	5.0	1.9	3
147	6.5	3.0	5.2	2.0	3
148	6.2	3.4	5.4	2.3	3
149	5.9	3.0	5.1	1.8	3

150 rows × 5 columns

In [90]:
```
X=df.iloc[:,0:4]
```

In [91]:
```
X
```

Out[91]:

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

In [92]:
```
y=df.iloc[:,4]
```

In [93]:
```
y
```

Out[93]:
```
0      1
1      1
2      1
3      1
4      1
      ..
145    3
146    3
147    3
148    3
149    3
Name: species, Length: 150, dtype: int64
```

In [94]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.33,random_state=42)
```

Traning Model

In [95]:
```
model=LinearRegression()
```

In [96]:
```
model.fit(X,y)
```

Out[96]:
```
LinearRegression()
```

In [97]:
```
model.score(X,y) #coef of prediction
```

Out[97]:
```
0.9384223675331595
```

In [98]:
```
model.coef_
```

Out[98]:
```
array([-0.10074146, -0.04424045,  0.22708138,  0.60989412])
```

In [99]:
```
model.intercept_
```

Out[99]:
```
1.1920839948281841
```

Making Predictions

In [100]:
```
y_pred=model.predict(X_test)
```

Model Evolution

In [101]:
```
print("Mean squared error: %.2f" % np.mean((y_pred - y_test) ** 2))
```

```
Mean squared error: 0.04
```

In [ ]:
```
**************END OF CODE**************
```

In [ ]:
```
*****************THANK YOU*****************
```