Iris flower classification

In [1]:
```python
# importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```python
iris = pd.read_csv('IRIS.csv')
```

In [3]:
```python
iris
```

Out[3]:

	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
...	...	...	...	...	...
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

In [4]:
```python
iris.head()
```

Out[4]:

	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 [5]:
```
iris.tail()
```

Out[5]:

	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 [6]:
```
iris.info()
```

```
<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
```

In [47]:
```
iris.shape
```

Out[47]:
```
(150, 5)
```

In [48]:
```
iris.size
```

Out[48]:
```
750
```

In [49]:
```
iris.columns
```

Out[49]:
```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

In [50]:
```
iris.describe()
```

Out[50]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [51]:
```
iris.keys()
```

Out[51]:
```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

In [52]:
```
iris['species']
```

Out[52]:
```
0         Iris-setosa
1         Iris-setosa
2         Iris-setosa
3         Iris-setosa
4         Iris-setosa
            ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: species, Length: 150, dtype: object
```

In [53]:
```
iris['species'].value_counts()
```

Out[53]:
```
species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: count, dtype: int64
```

Preprocessing

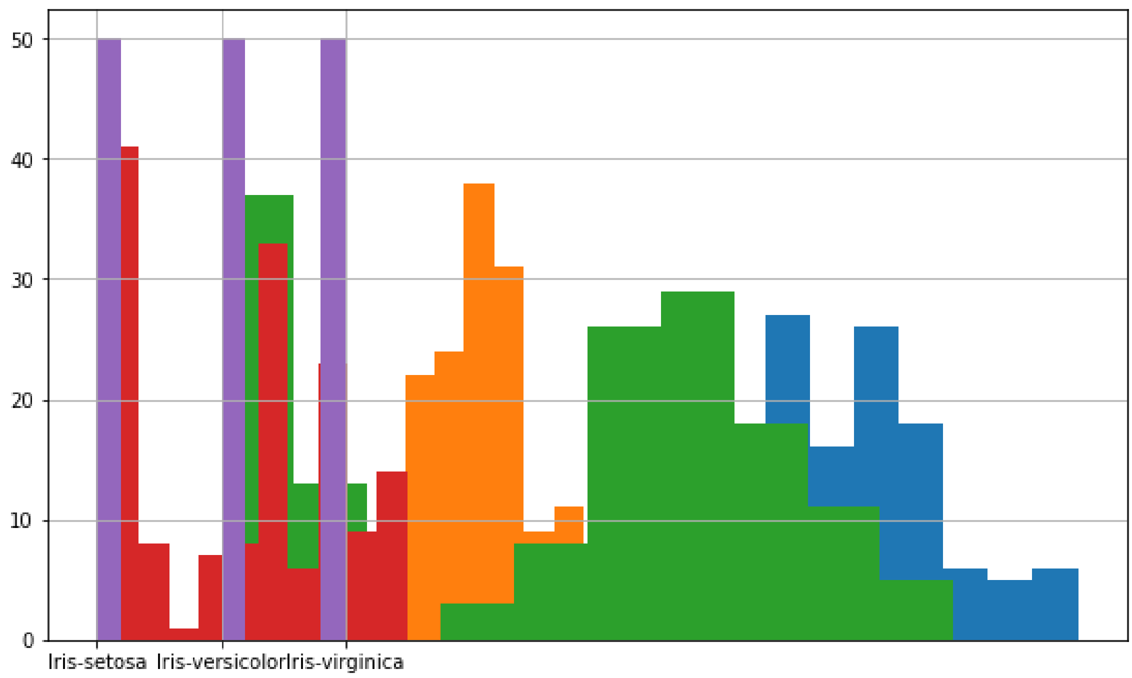
In [54]:
```
iris.isnull().sum()
```

Out[54]:
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

Data Visualization

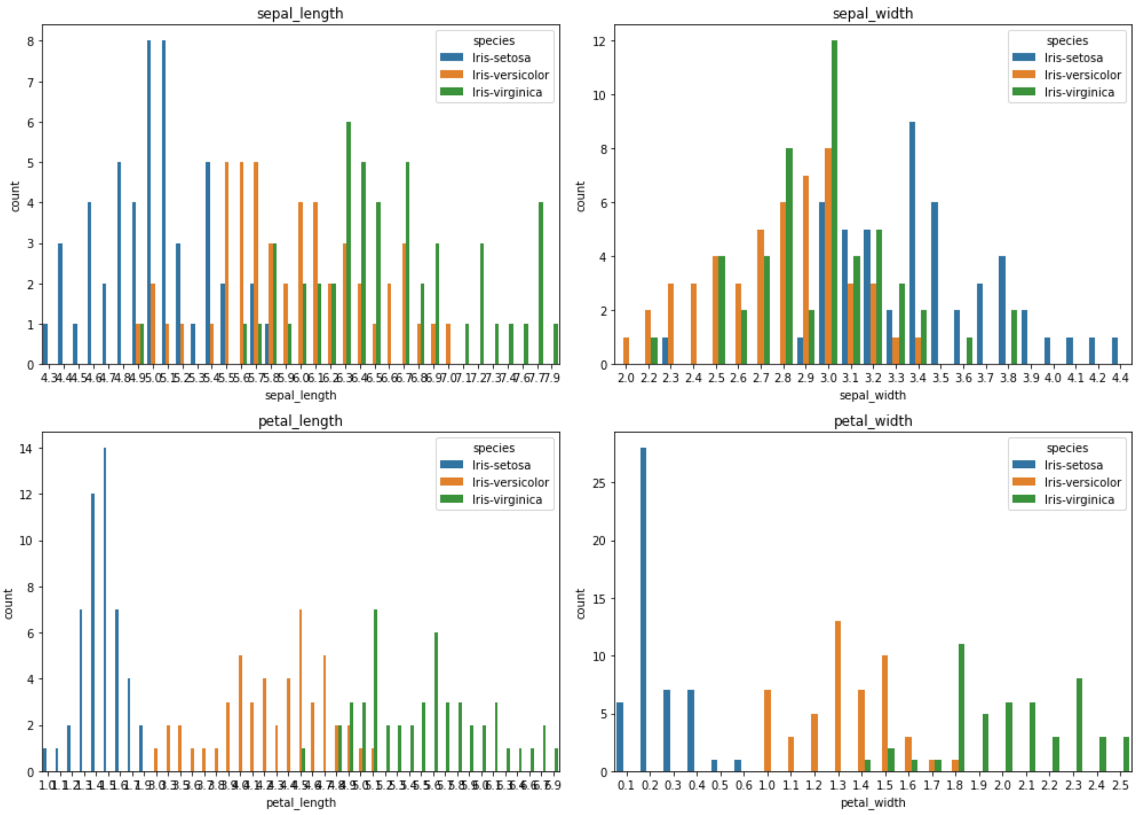
In [55]:
```
plt.figure(figsize=(10,6))
iris['sepal_length'].hist()
iris['sepal_width'].hist()
iris['petal_length'].hist()
iris['petal_width'].hist()
iris['species'].hist()
```

Out[55]: <AxesSubplot:>

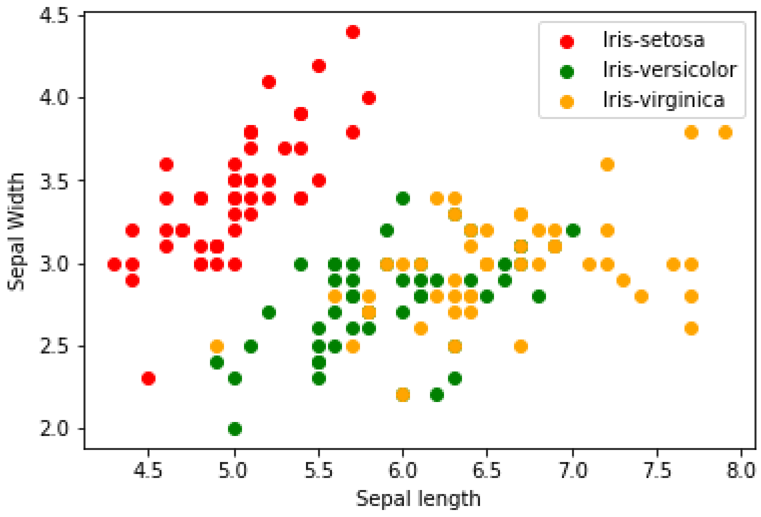


In [56]:

```
cols = [ 'sepal_length', 'sepal_width', 'petal_length', 'petal_width']
species='species'
n_rows = 2
n_cols = 2

# The subplot grid and the figure size of each graph
# This returns a Figure (fig) and an Axes Object (axs)
fig, axs = plt.subplots(n_rows, n_cols, figsize=(n_cols*7,n_rows*5))

for r in range(0,n_rows):
    for c in range(0,n_cols):

        i = r*n_cols+ c #index to go through the number of columns
        ax = axs[r][c] #Show where to position each subplot
        sns.countplot(x=cols[i], hue=species, data=iris, ax=ax)
        ax.set_title(cols[i])
        ax.legend(title=species, loc='upper right')

plt.tight_layout()   #tight_layout
```

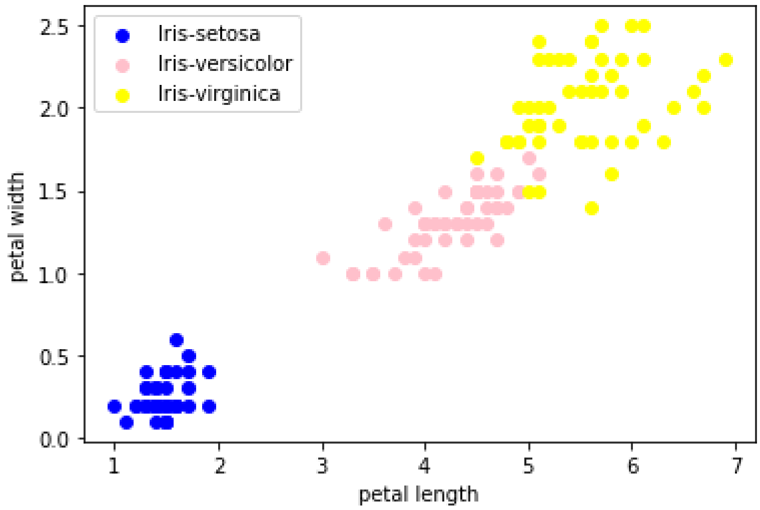


In [57]:
```
# scatter plot
# Assuming DataFrame 'iris' with columns 'species', 'sepal_height', and 'se
colors = ['red', 'green', 'orange']
species = ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']

for i in range(3):
    x = iris[iris['species'] == species[i]]
    plt.scatter(x['sepal_length'], x['sepal_width'], c=colors[i], label=spe

plt.xlabel('Sepal length')
plt.ylabel('Sepal Width')
plt.legend()
plt.show()
```



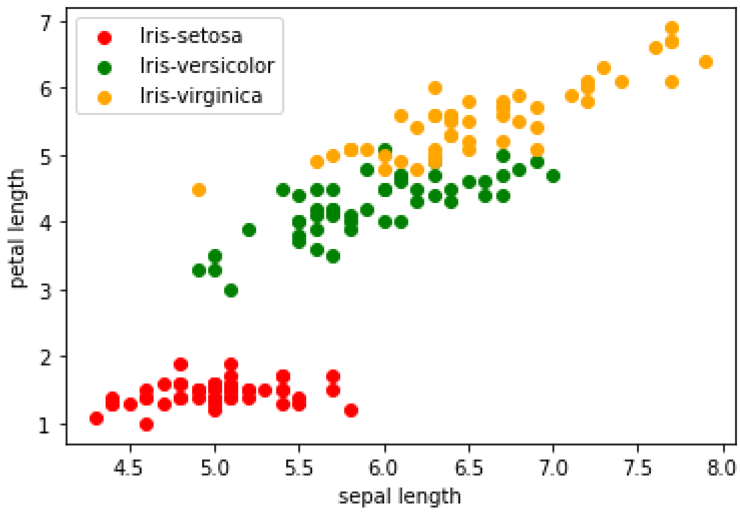
In [58]:

```python
# scatter plot basis of petal length and petal_width
import pandas as pd
import matplotlib.pyplot as plt
colors=['blue', 'pink', 'yellow']
species=['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']

for i in range(3):
        x= iris[iris['species']== species[i]]
        plt.scatter(x['petal_length'], x['petal_width'], c=colors[i], label
plt.xlabel('petal length')
plt.ylabel('petal width')
plt.legend()
plt.show()
```

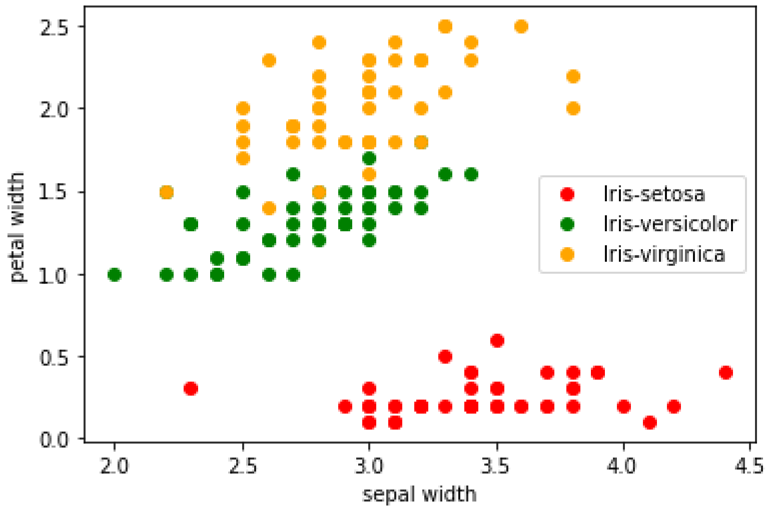


In [59]:

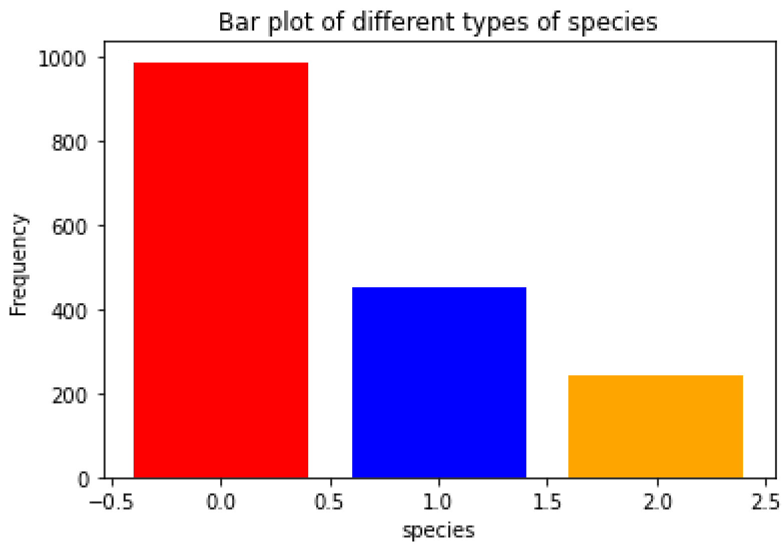
```python
colors=['red', 'green', 'orange']
for i in range(3):
        x= iris[iris['species']== species[i]]
        plt.scatter(x['sepal_length'], x['petal_length'], c=colors[i], labe
plt.xlabel('sepal length')
plt.ylabel('petal length')
plt.legend()
plt.show()
```



In [60]:
```
colors=['red', 'green', 'orange']
for i in range(3):
        x= iris[iris['species']== species[i]]
        plt.scatter(x['sepal_width'], x['petal_width'], c=colors[i], label=
plt.xlabel('sepal width')
plt.ylabel('petal width')
plt.legend()
plt.show()
```



In [61]:
```
counts = (986, 450, 240)
species = ('Iris-setosa ', 'Iris-versicolor', 'Iris-virginica' )
index = np.arange(len(species))
plt.bar(index, counts, color=['red','blue','orange'])
plt.title('Bar plot of different types of species')
plt.xlabel('species')
plt.ylabel('Frequency')
plt.show()
```



correaltion matrix

In [90]:
```python
iris.corr= pd.get_dummies(iris, columns=['species'], drop_first=True)
#corr_matrix= iris.corr
#corr_matrix()
iris.corr
```

Out[90]:

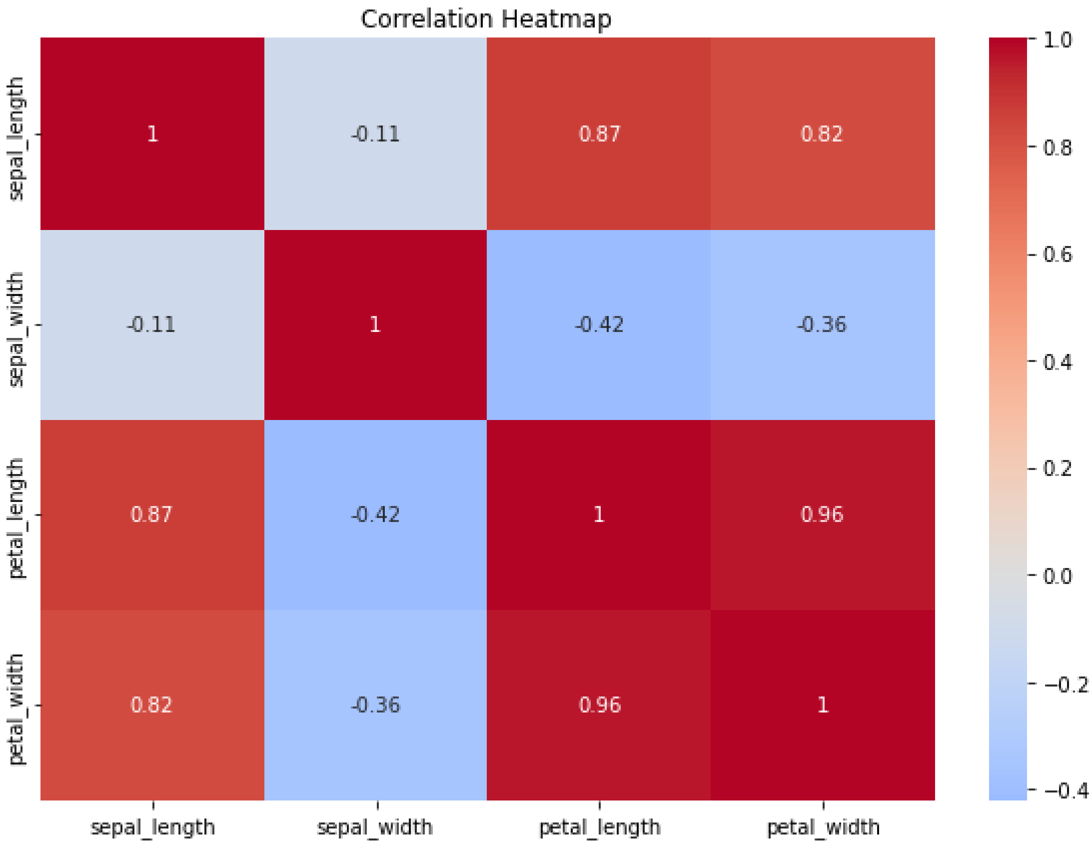
	sepal_length	sepal_width	petal_length	petal_width	species_Iris-versicolor	species_Iris-virginica
0	5.1	3.5	1.4	0.2	False	False
1	4.9	3.0	1.4	0.2	False	False
2	4.7	3.2	1.3	0.2	False	False
3	4.6	3.1	1.5	0.2	False	False
4	5.0	3.6	1.4	0.2	False	False
...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	False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

150 rows × 6 columns

In [100]:
```python

numeric_columns = iris.select_dtypes(include=['number'])  # Select only num
correlation_matrix = numeric_columns.corr()

print(correlation_matrix)

```

```
              sepal_length  sepal_width  petal_length  petal_width
sepal_length      1.000000    -0.109369      0.871754     0.817954
sepal_width      -0.109369     1.000000     -0.420516    -0.356544
petal_length      0.871754    -0.420516      1.000000     0.962757
petal_width       0.817954    -0.356544      0.962757     1.000000
```

In [105]:
```
#visualizing the correalation
numeric_columns = iris.select_dtypes(include=['number'])  # Select only num
correlation_matrix = numeric_columns.corr()

# Create a heatmap of the correlation matrix
plt.figure(figsize=(10, 7))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0)

plt.title('Correlation Heatmap')
plt.show()
```



Label encoder

In [112]:
```python
from sklearn.preprocessing import LabelEncoder

# Create an instance of LabelEncoder
la = LabelEncoder()

# Transform the 'species' column
iris['species'] = la.fit_transform(iris['species'])

iris.head()
```

Out[112]:

	sepal_length	sepal_width	petal_length	petal_width	species
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

In [115]:
```python
from sklearn.model_selection import train_test_split
x= iris.drop(columns=['species'])
y= iris['species']
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size=0.20)
```

In [117]:
```python
print(x_train)
```

```
     sepal_length  sepal_width  petal_length  petal_width
15            5.7          4.4           1.5          0.4
142           5.8          2.7           5.1          1.9
89            5.5          2.5           4.0          1.3
141           6.9          3.1           5.1          2.3
106           4.9          2.5           4.5          1.7
..            ...          ...           ...          ...
128           6.4          2.8           5.6          2.1
51            6.4          3.2           4.5          1.5
114           5.8          2.8           5.1          2.4
18            5.7          3.8           1.7          0.3
82            5.8          2.7           3.9          1.2

[120 rows x 4 columns]
```

In [119]:
```python
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
```

In [121]:
```python
print("Accuracy of a model:", model.score(x_train, y_train))
```

```
Accuracy of a model: 0.9666666666666667
```

In [122]:
```python
print("Accuracy of a testing model:", model.score(x_test, y_test))
```

```
Accuracy of a testing model: 0.9333333333333333
```

In [125]:
```
from sklearn.neighbors import KNeighborsClassifier
model= KNeighborsClassifier()
```

In [126]:
```
model.fit(x_train, y_train)
```

Out[126]:
```
KNeighborsClassifier()
```

In [127]:
```
print("Accuracy of a model:", model.score(x_train, y_train))
```

```
Accuracy of a model: 0.9666666666666667
```

In [128]:
```
print("Accuracy of a testing model:", model.score(x_test, y_test))
```

```
Accuracy of a testing model: 0.9333333333333333
```

In [ ]: