Iris Dataset

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
import seaborn as sns
plt.style.use('ggplot')
```

Putting Data into a Data Frame

To view the data more easily we can put this information into a data frame by using the Pandas library. Let's create a data frame to store the data information about the flowers' features first.

In [5]:
```
df = pd.read_csv(r"C:\Users\logap\Downloads\archive(4)\Iris.csv")
```

In [6]:
```
df
```

Out[6]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

In [7]:
```
df.shape
```

Out[7]:
```
(150, 6)
```

Using data.head() will automatically output the first 5 rows of data, but if we can also specify how many rows we want in the brackets, data.head(5).

In [8]:
```
df.head(5)
```

Out[8]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [22]:
```
df.tail()
```

Out[22]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

Exploratory Data Analysis (EDA)

Data Cleaning

It's super important to look through your data, make sure it is clean, and begin to explore relationships between features and target variables. Since this is a relatively simple data set there is not much cleaning that needs to be done, but let's walk through the steps.

In [22]:
```
df.dtypes
```

Out[22]:
```
Id                 int64
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

Statistical Overview

In [23]:
```
df.describe()
```

Out[23]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

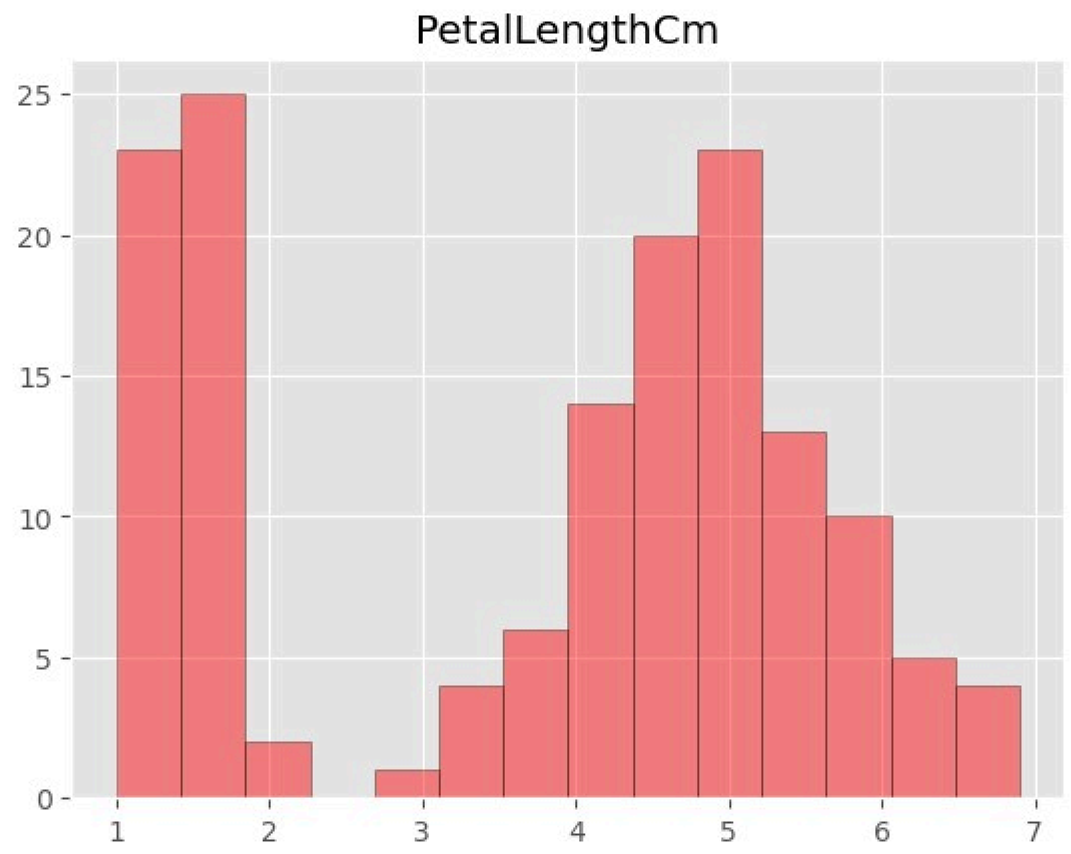
Check for Missing Values

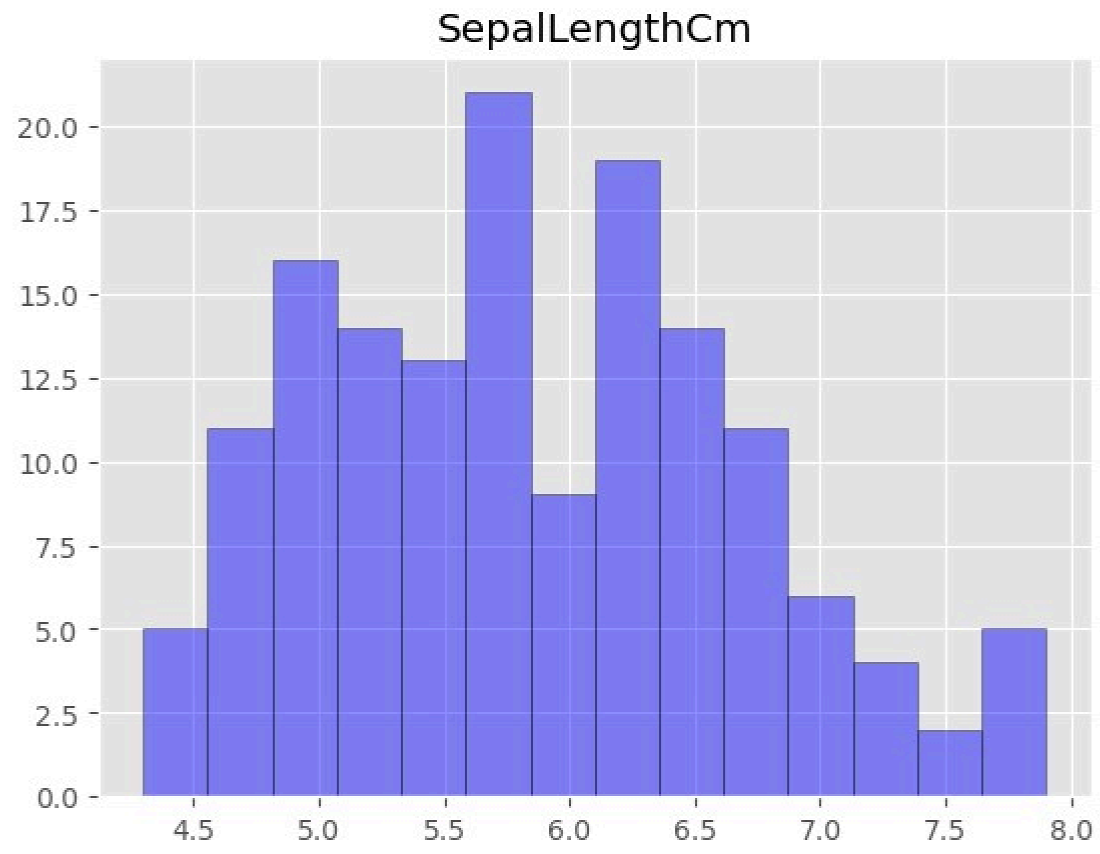
In [24]:
```
df.isnull().sum()
```

Out[24]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

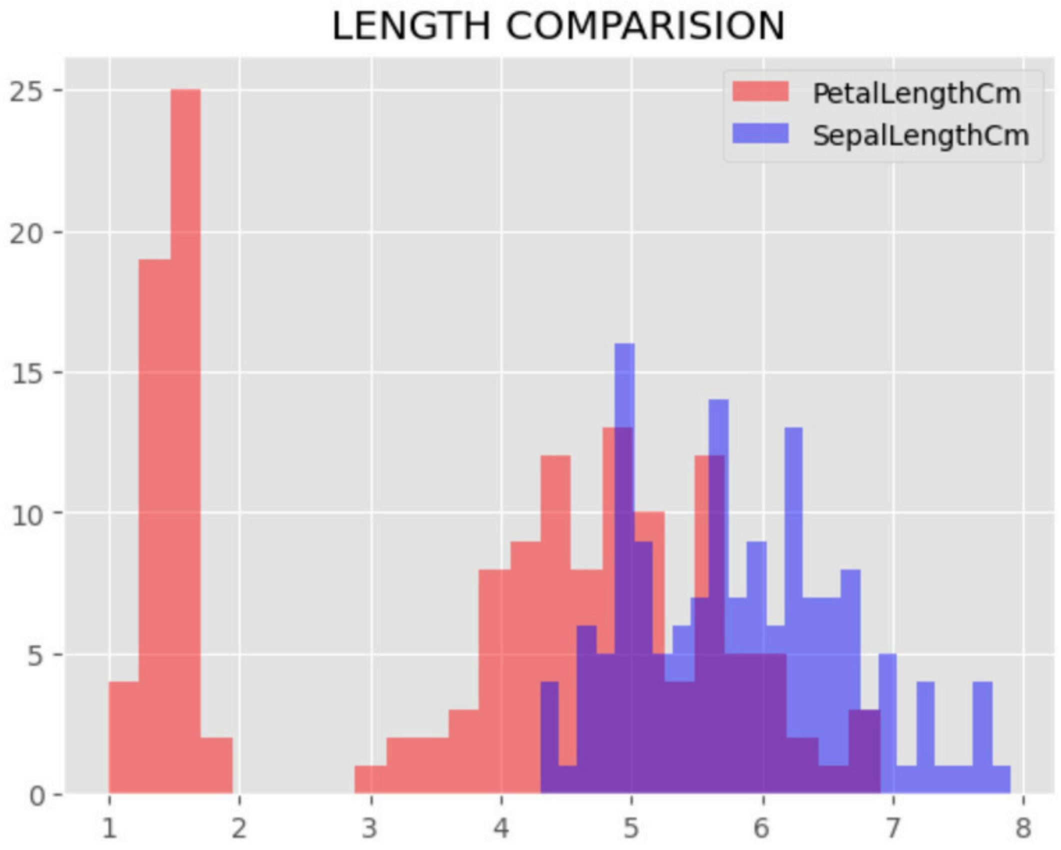
Data Visualization

In [15]:
```
series1 = ['PetalLengthCm']
series2 = ['SepalLengthCm']

df.hist(series1,bins=14, alpha=0.45,color='red',edgecolor='black')

df.hist(series2,bins=14, alpha=0.45,color='blue',edgecolor='black')

plt.show()
```

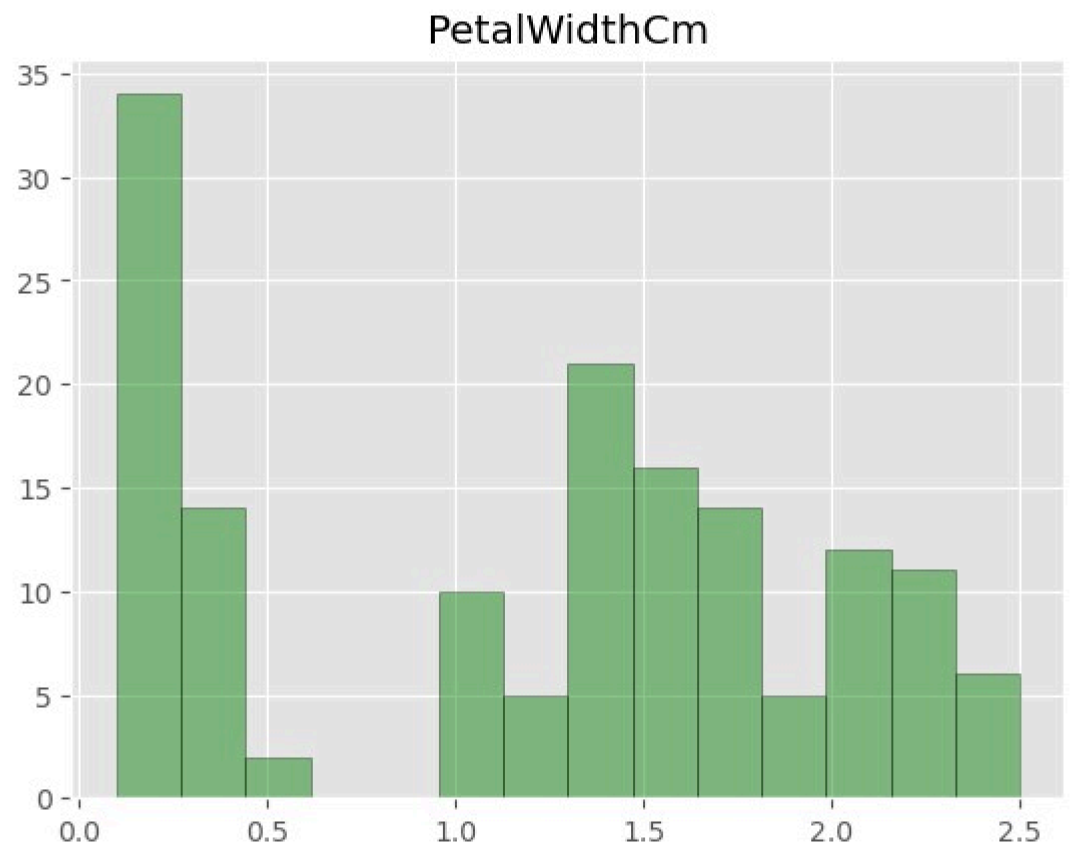

PetalLengthCm
25
20
15
10
5
0
1
2
3
4
5
6
7

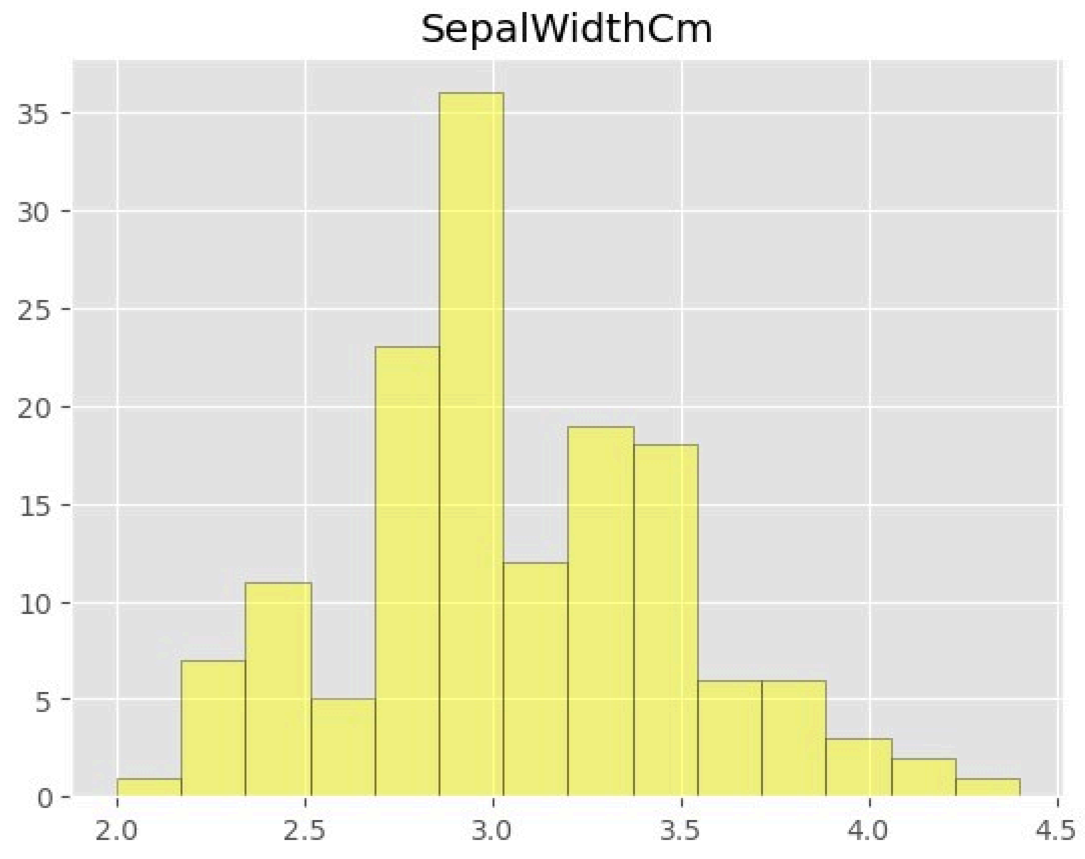

SepalLengthCm
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

In [16]:
```
plt.hist(df['PetalLengthCm'], bins=25, alpha=0.45, color='red')
plt.hist(df['SepalLengthCm'], bins=25, alpha=0.45, color='blue')

plt.title("LENGTH COMPARISION")
plt.legend(["PetalLengthCm","SepalLengthCm"])

plt.show()
```

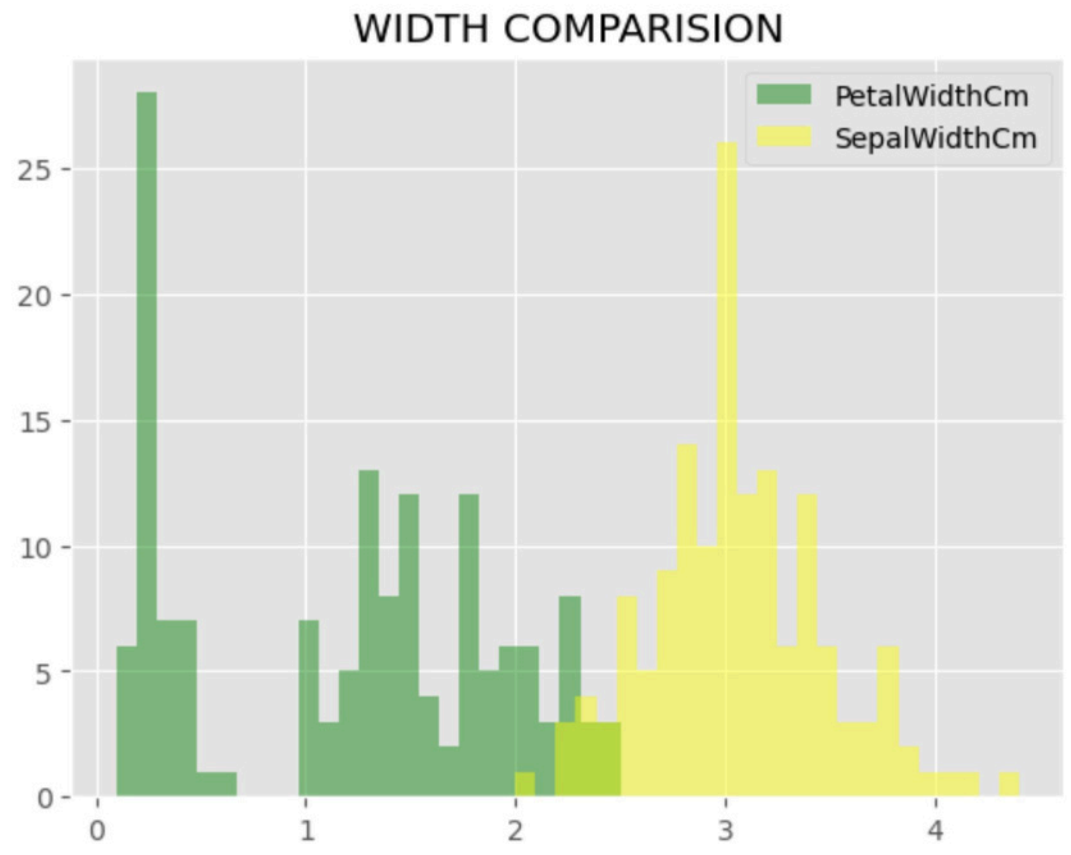


In [17]:
```
series1 = ['PetalWidthCm']
series2 = ['SepalWidthCm']

df.hist(series1,bins=14, alpha=0.45,color='green',edgecolor='black')

df.hist(series2,bins=14, alpha=0.45,color='yellow',edgecolor='black')

plt.show()
```


PetalWidthCm
35
30
25
20
15
10
5
0
0.0
0.5
1.0
1.5
2.0
2.5


SepalWidthCm
35
30
25
20
15
10
5
0
2.0
2.5
3.0
3.5
4.0
4.5

In [6]:
```
plt.hist(df['PetalWidthCm'], bins=25, alpha=0.45, color='green')
plt.hist(df['SepalWidthCm'], bins=25, alpha=0.45, color='yellow')

plt.title("WIDTH COMPARISION")
plt.legend(["PetalWidthCm","SepalWidthCm"])

plt.show()
```



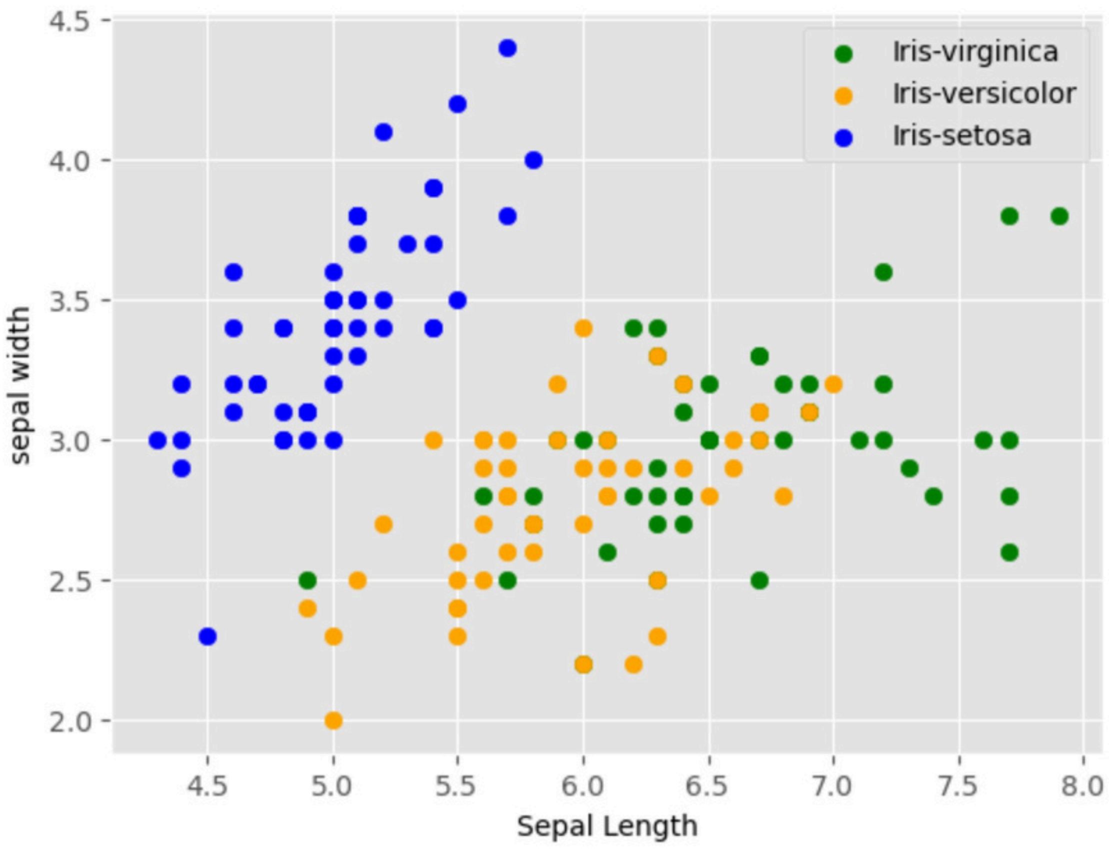
In [13]:
```
colors=['green','orange','blue']
species=['Iris-virginica','Iris-versicolor','Iris-setosa']
```

Scatter Plot

To start looking at the relationships between features, we can create scatter plots to further visualize the way the different classes of flowers relate to sepal and petal data.

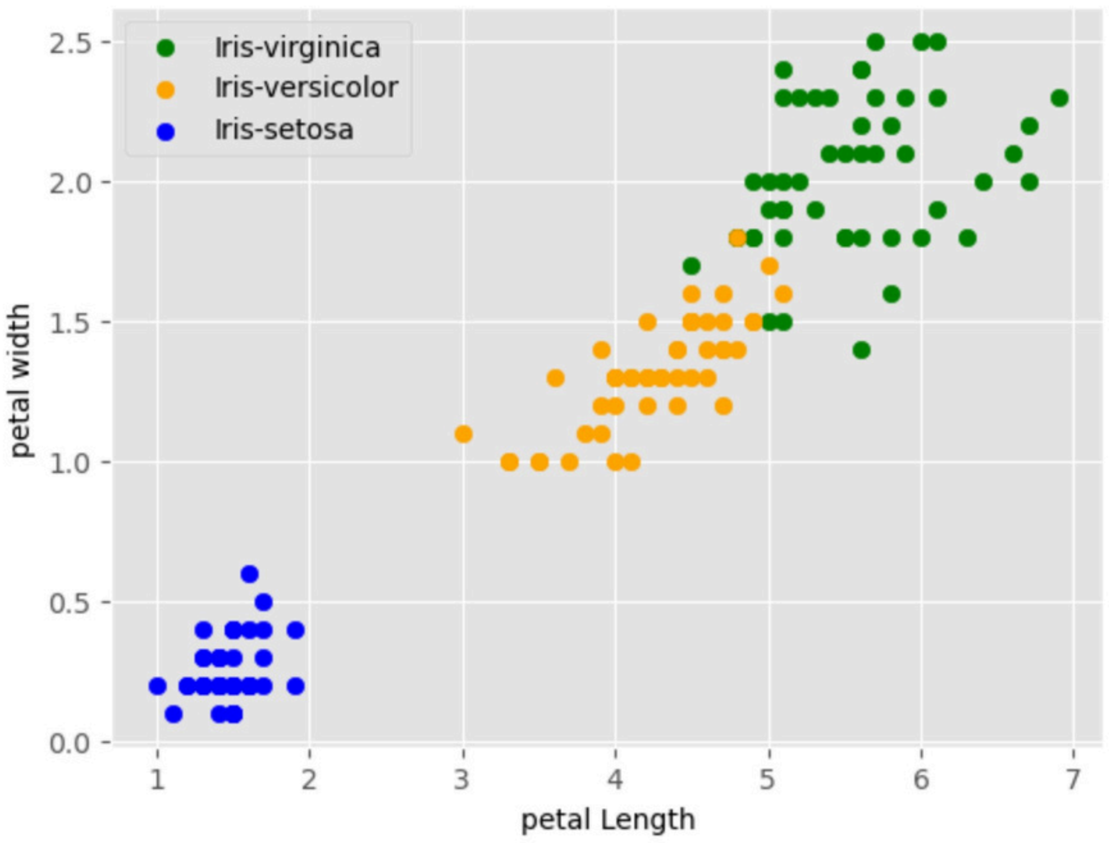
In [14]:
```
for i in range(3):
    x=df[df['Species']==species[i]]
    plt.scatter(x['SepalLengthCm'],x['SepalWidthCm'],c= colors[i],label=species[i])

plt.xlabel("Sepal Length")
plt.ylabel("sepal width")
plt.legend()
```

Out[14]:
```
<matplotlib.legend.Legend at 0x26cb3604040>
```

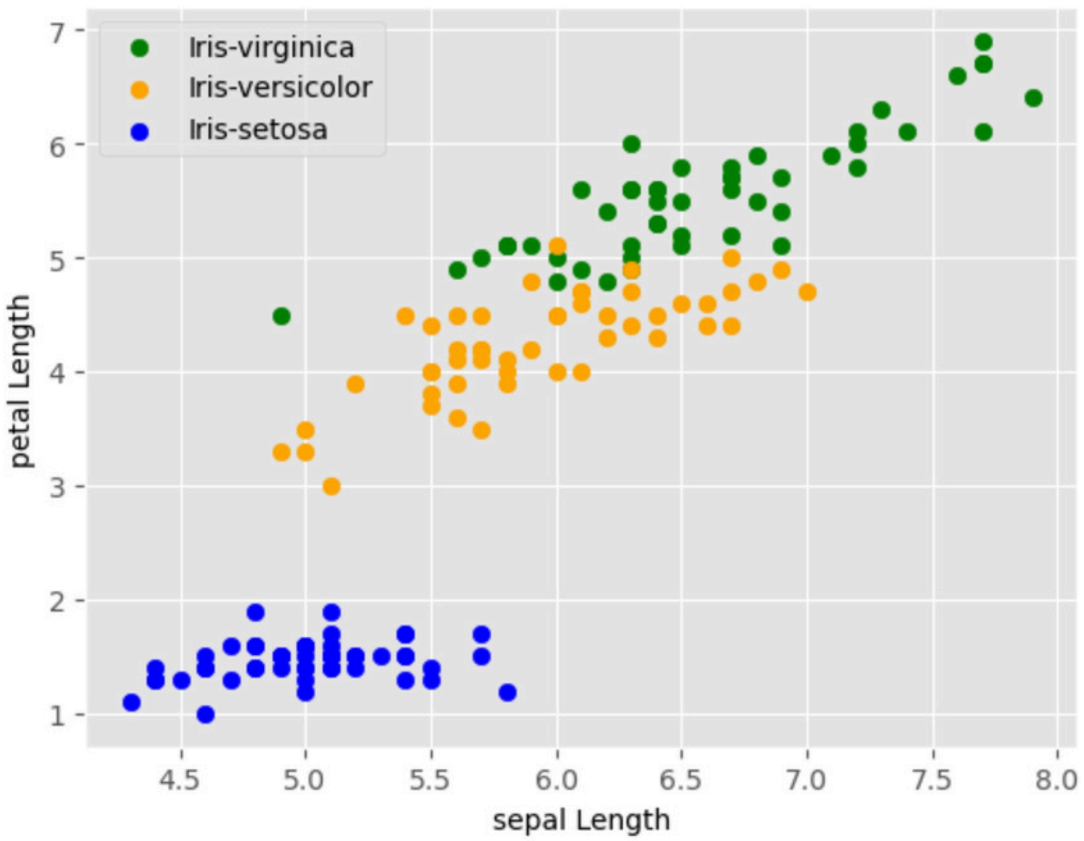


In [15]:
```
for i in range(3):
    x=df[df['Species']==species[i]]
    plt.scatter(x['PetalLengthCm'],x['PetalWidthCm'],c= colors[i],label=species[i])

plt.xlabel("petal Length")
plt.ylabel("petal width")
plt.legend()
```

Out[15]:
```
<matplotlib.legend.Legend at 0x26cb32293c0>
```

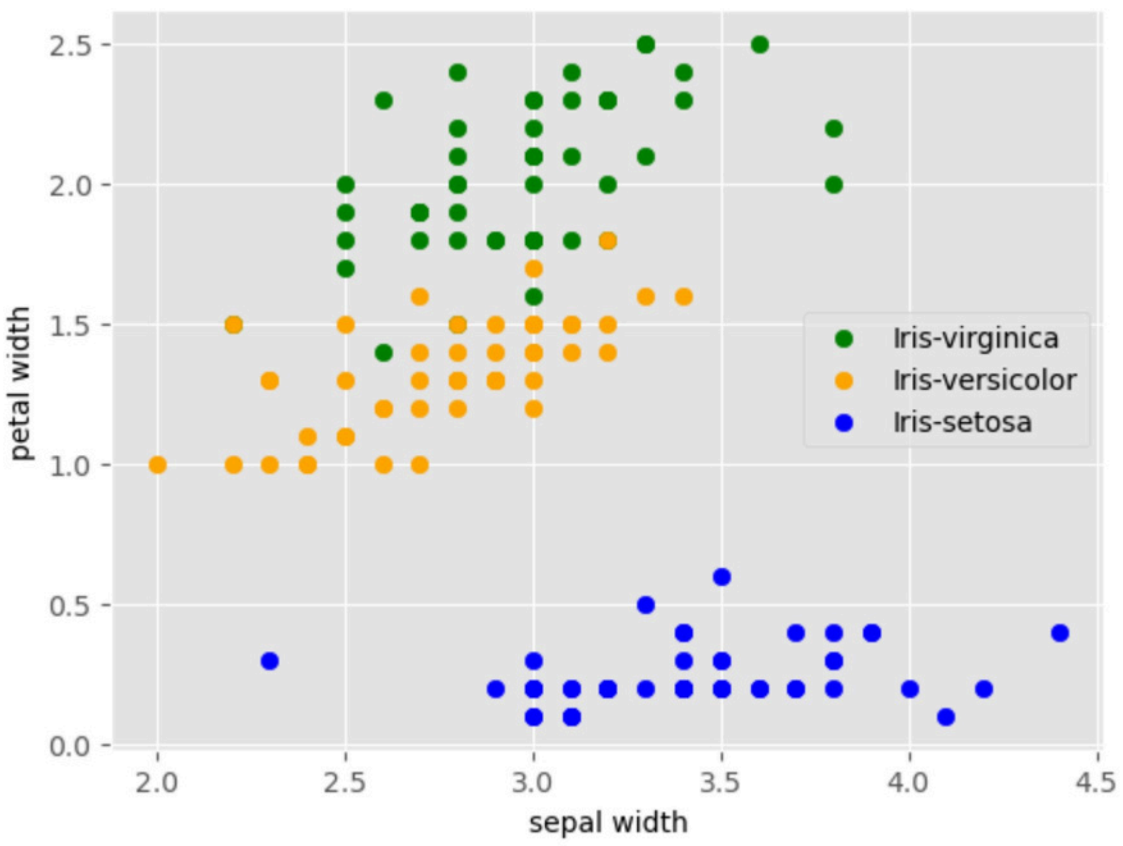


```
In [16]: for i in range(3):
             x=df[df['Species']==species[i]]
             plt.scatter(x['SepalLengthCm'],x['PetalLengthCm'],c= colors[i],label=species[i]

         plt.xlabel("sepal Length")
         plt.ylabel("petal Length")
         plt.legend()

Out[16]: <matplotlib.legend.Legend at 0x26cb47ba500>
```



In [17]:
```
for i in range(3):
    x=df[df['Species']==species[i]]
    plt.scatter(x['SepalWidthCm'],x['PetalWidthCm'],c= colors[i],label=species[i])

plt.xlabel("sepal width")
plt.ylabel("petal width")
plt.legend()
```

Out[17]:
```
<matplotlib.legend.Legend at 0x26cb48a3310>
```



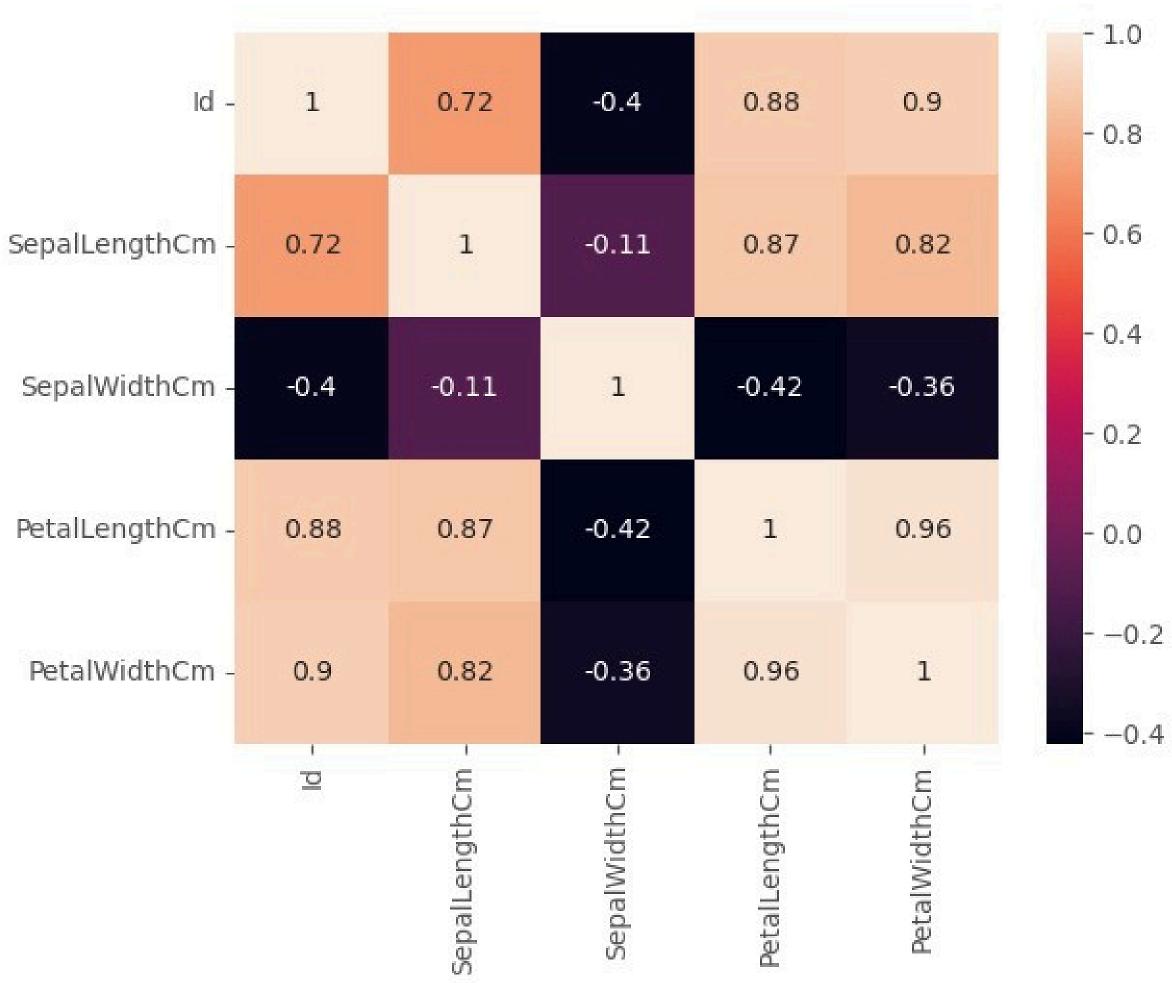
```
In [20]: corr=df.corr(numeric_only=True)
```

Correlation

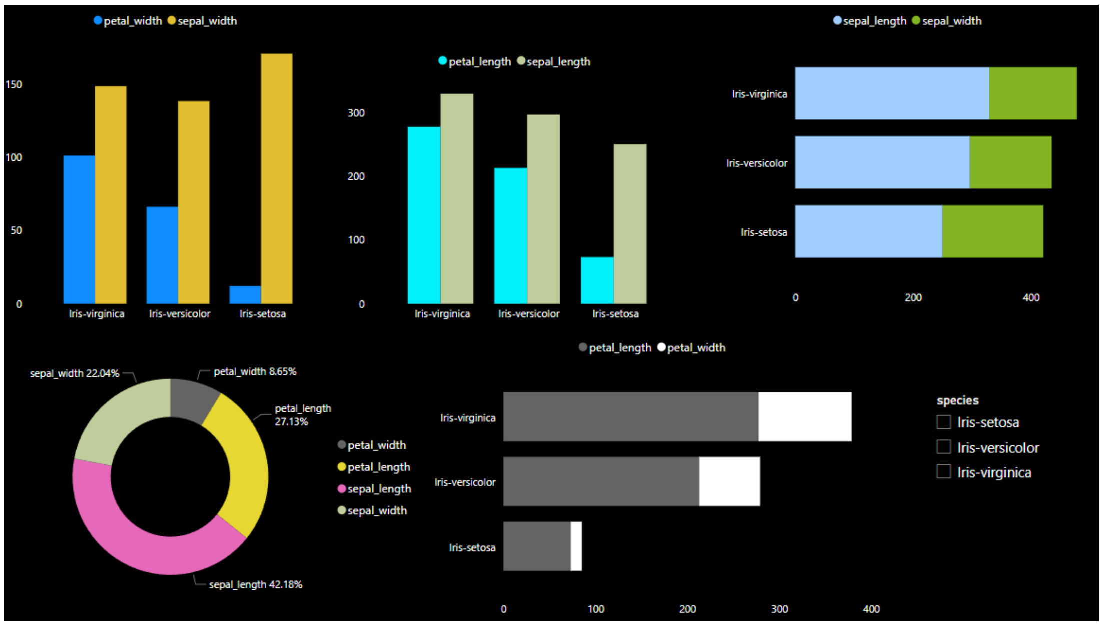
The Seaborn library has a great heat map visual for mapping the correlations between features. The higher the number is, the greater the correlation between the two elements. A high positive correlation indicates that the two elements have a positive linear relationship (as one increases the other also increases), and a low negative correlation indicates a negative linear relationship (as one increases the other decreases).

```
In [21]: sns.heatmap(corr,annot=True)
```

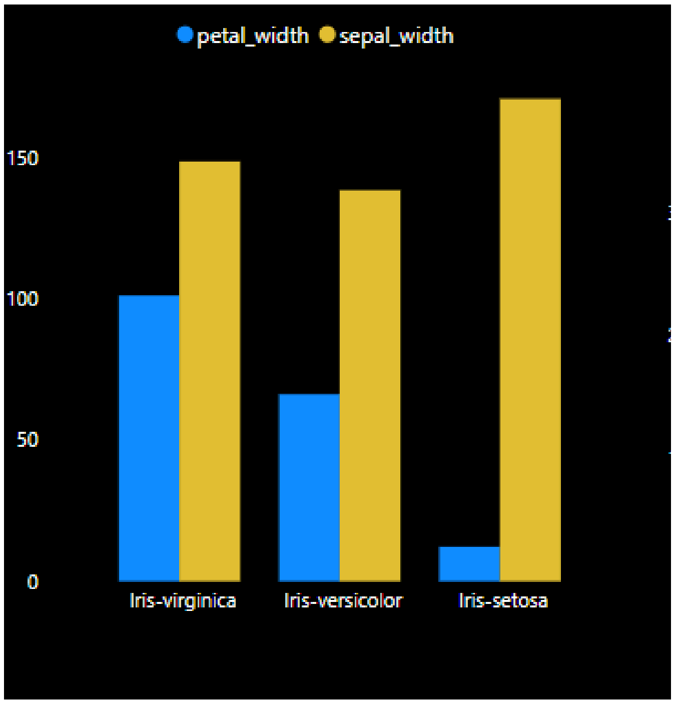
```
Out[21]: <Axes: >
```


Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
1 0.72 -0.4 0.88 0.9
0.72 1 -0.11 0.87 0.82
-0.4 -0.11 1 -0.42 -0.36
0.88 0.87 -0.42 1 0.96
0.9 0.82 -0.36 0.96 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

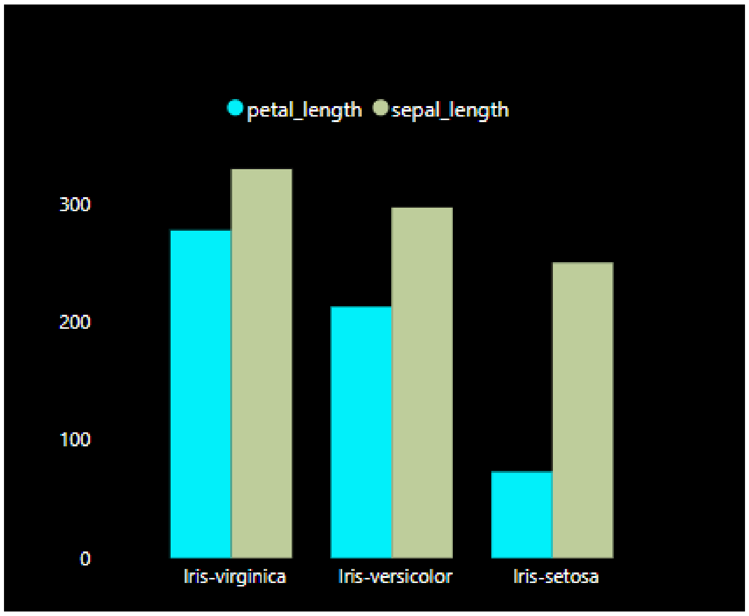
Iris Dashboard



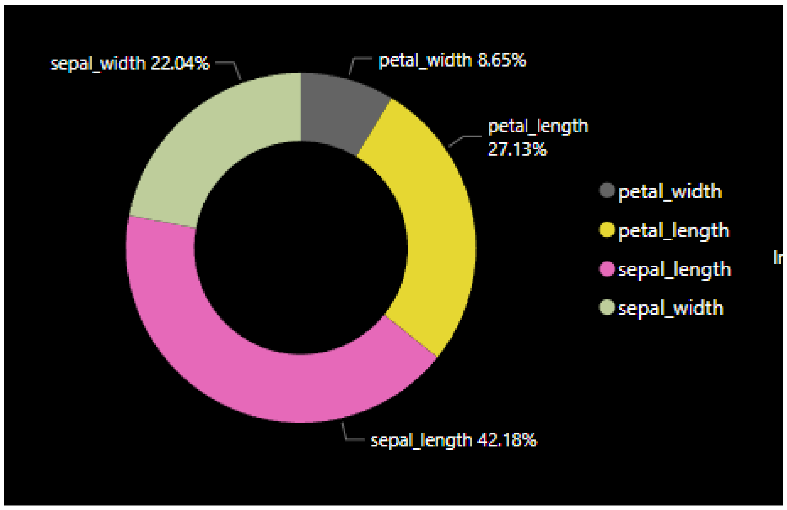
The charts appear to depict data about iris flowers, including petal and sepal length and width of three iris species: Iris setosa, Iris versicolor, and Iris virginica. The data is also presented as percentages. Overall, the computer screen shows a data visualization of iris flower measurements.



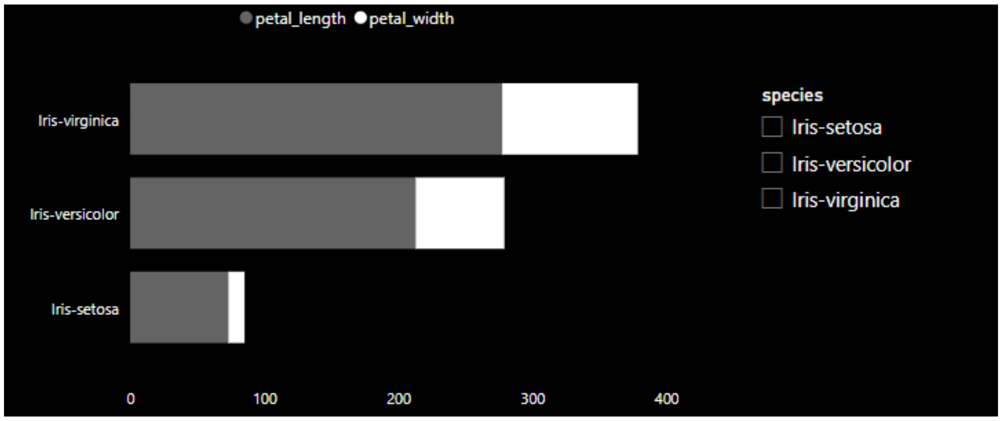
Bar graph that appears to show the distribution of sepal width and petal width of three iris flower species: Iris setosa, Iris versicolor, and Iris virginica. The x-axis of the graph lists the three species, and the y-axis lists the counts. For each species, the graph has two bars, one for sepal width and one for petal width. The iris setosa flowers appear to have the smallest petal width and sepal width, while the iris virginica flowers appear to have the largest petal width and sepal width.



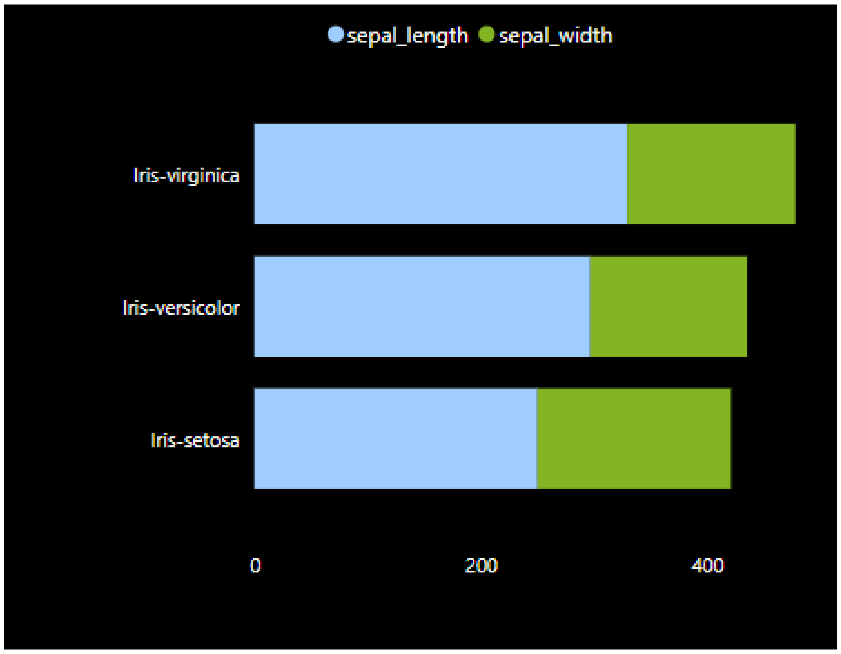
The graph shows the petal and sepal width of three iris flower species. Iris setosa has the smallest petal and sepal width, while Iris virginica has the largest. Overall, the iris species seem to have distinct ranges for petal and sepal width.



This chart divides iris flower features into petal and sepal dimensions, further dividing them by width and length. Petal length makes up the largest portion (27.13%) followed by sepal length (42.18%). Petal width (8.65%) and sepal width (22.04%) comprise the remaining portions.



The graph shows the distribution of petal width and sepal width of three iris flower species: Iris setosa, Iris versicolor, and Iris virginica. Iris setosa has the smallest petal and sepal width, while Iris virginica has the largest. Overall, the iris species seem to have distinct ranges for petal and sepal width.



This chart shows the petal and sepal width of three iris species: Iris setosa, Iris versicolor, and Iris virginica. Iris setosa has the smallest measurements, while Iris virginica has the largest. The iris species appear to have distinct ranges in petal and sepal width.

Conclusion

Hopefully this walk-through helped to show some major steps in the process of a data science project. Of course this is not an exhaustive list of steps that could be taken with this data set, but it aims to carefully show some of the important steps of classification.

This is a classic data set because it is relatively straightforward, but the steps highlighted here can be applied to a classification project of any kind. Follow for more simple (and advanced) data set walk-throughs in the future!