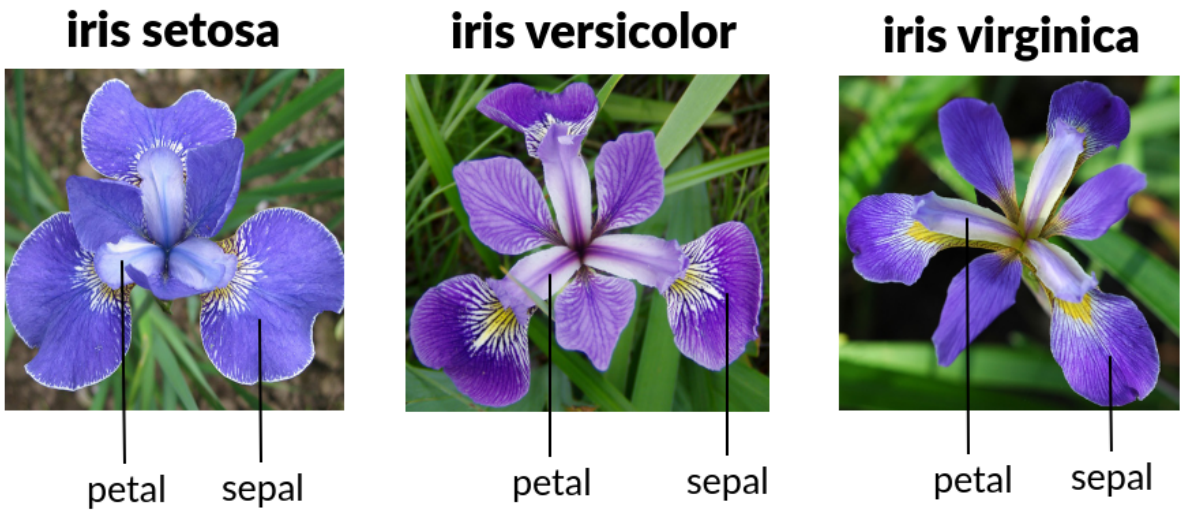
IRIS FLOWER CLASSIFICATION

The Iris dataset is a widely used benchmark dataset in the field of machine learning and statistics for classification tasks.

This dataset consists of 150 samples of iris flowers, with each sample belonging to one of three species: setosa, versicolor, or virginica. Each sample comprises four features: sepal length, sepal width, petal length, and petal width, all measured in centimeters.

The purpose of this dataset is to classify iris flowers based on these four features. It is often used for teaching and practicing various machine learning algorithms, especially for tasks related to classification, clustering, and dimensionality reduction.

The Iris dataset is relatively small, making it easily manageable for testing and implementing machine learning algorithms. It is considered a good starting point for beginners due to its simplicity and clear classification task.



Import the libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Load the dataset

```
df=pd.read_csv('/content/Iris.csv')
df
```

	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

Data Exploration

```
df.head()
```

	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

```
df.tail()
```

	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
						Iris-

```
df.shape
```

```
(150, 6)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

```
df.columns
```

```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

```
df.dtypes
```

```
Id                int64
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

```
df.describe()
```

	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

```
df.dropna()
```

	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

```
df.isnull().sum()
```

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

```
df=df.drop(['Id'],axis=1)
df
```

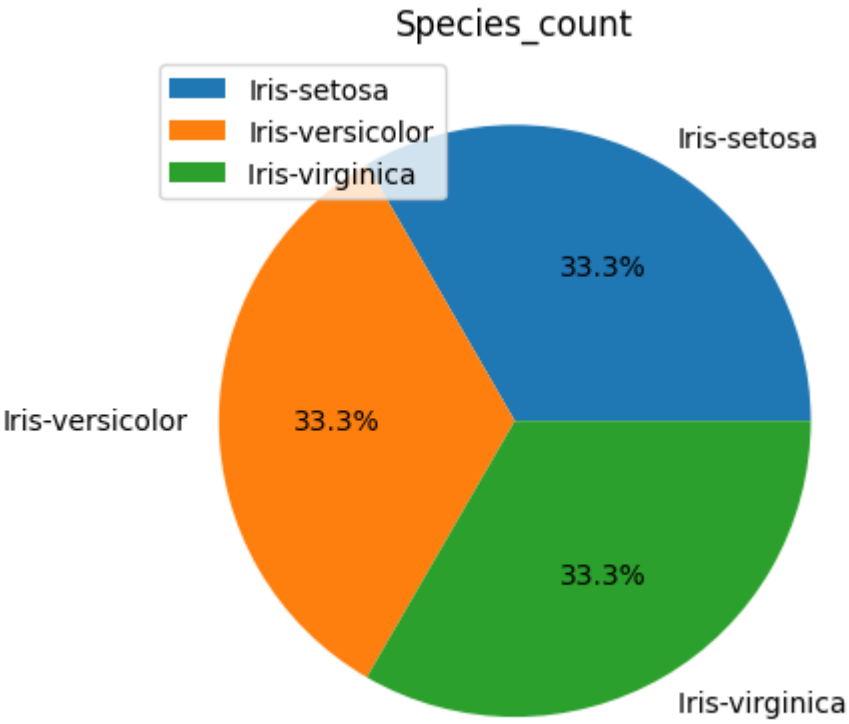
	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa

Data Visualization

```
y_counts=df['Species'].value_counts()
y_counts
```

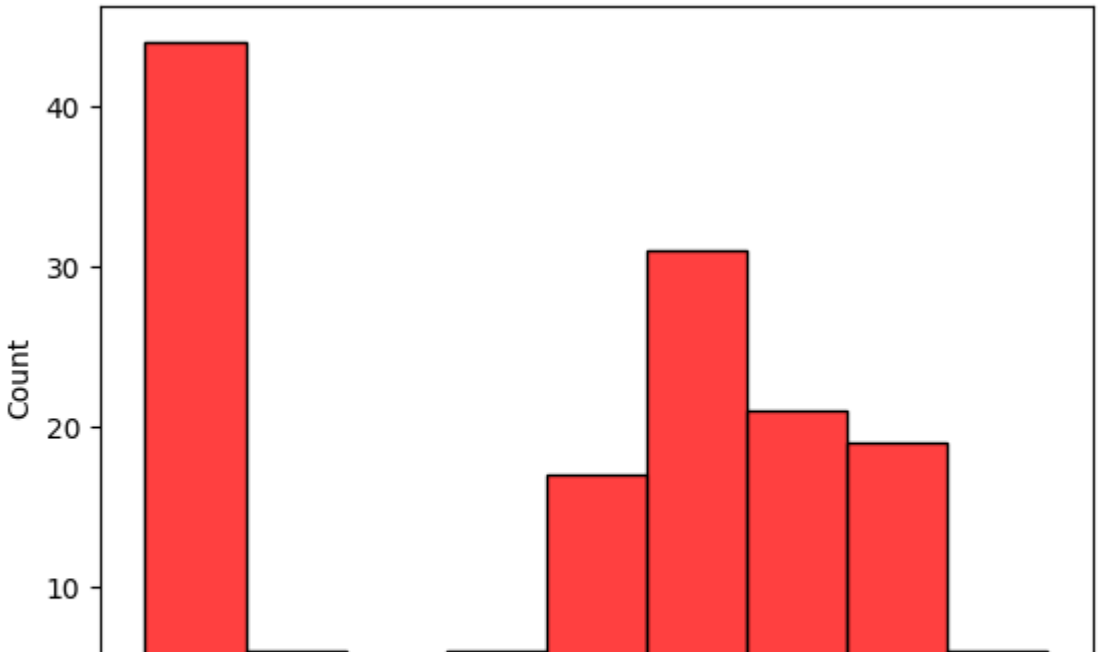
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

```
mylabels=['Iris-setosa','Iris-versicolor','Iris-virginica ']
plt.pie(y_counts,labels=mylabels,autopct="%1.1f%%")
plt.title('Species_count')
plt.legend(loc='upper left')
plt.show()
```



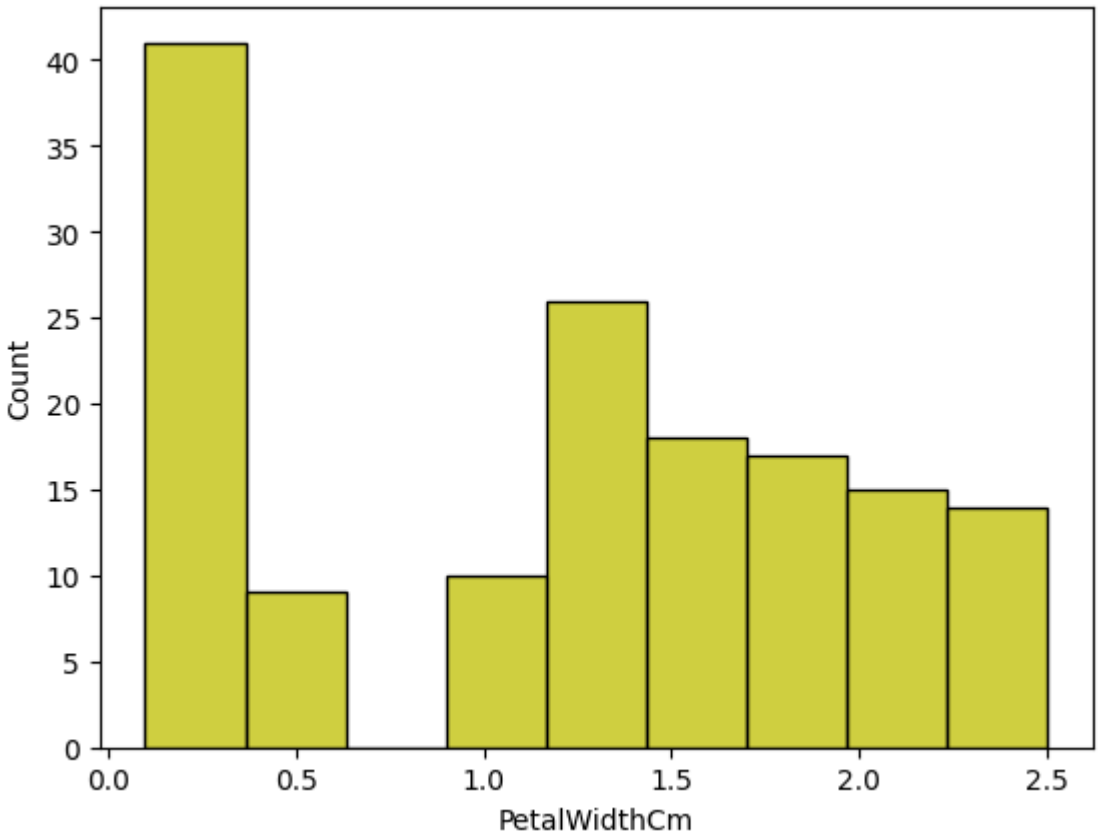

```
sns.histplot(df['PetalLengthCm'],color='r')
```

```
<Axes: xlabel='PetalLengthCm', ylabel='Count'>
```



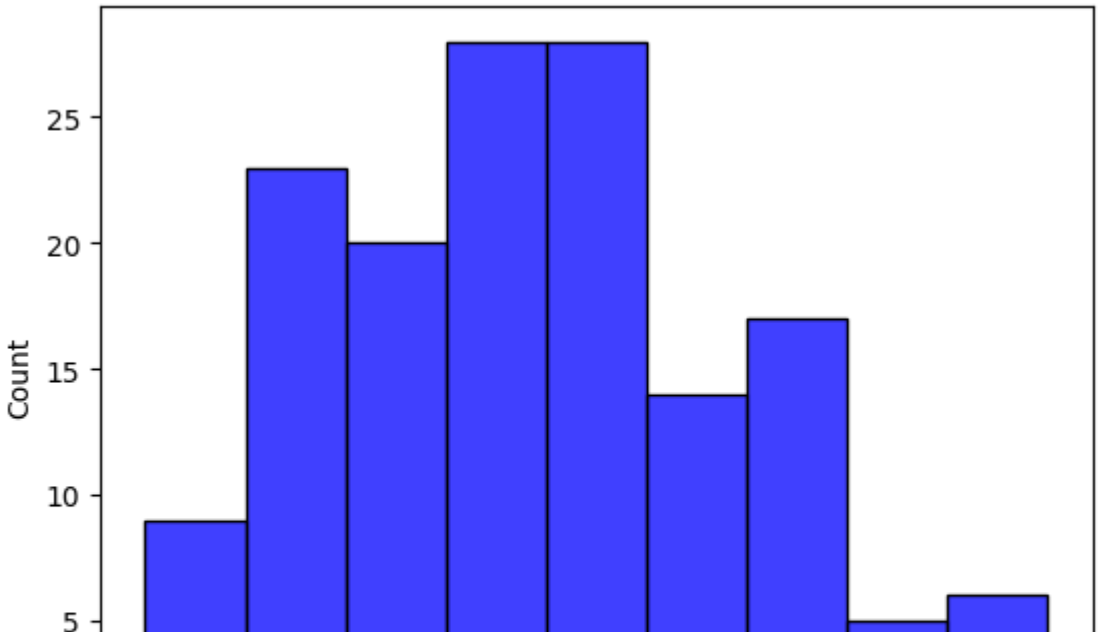

```
sns.histplot(df['PetalWidthCm'],color='y')
```

```
<Axes: xlabel='PetalWidthCm', ylabel='Count'>
```



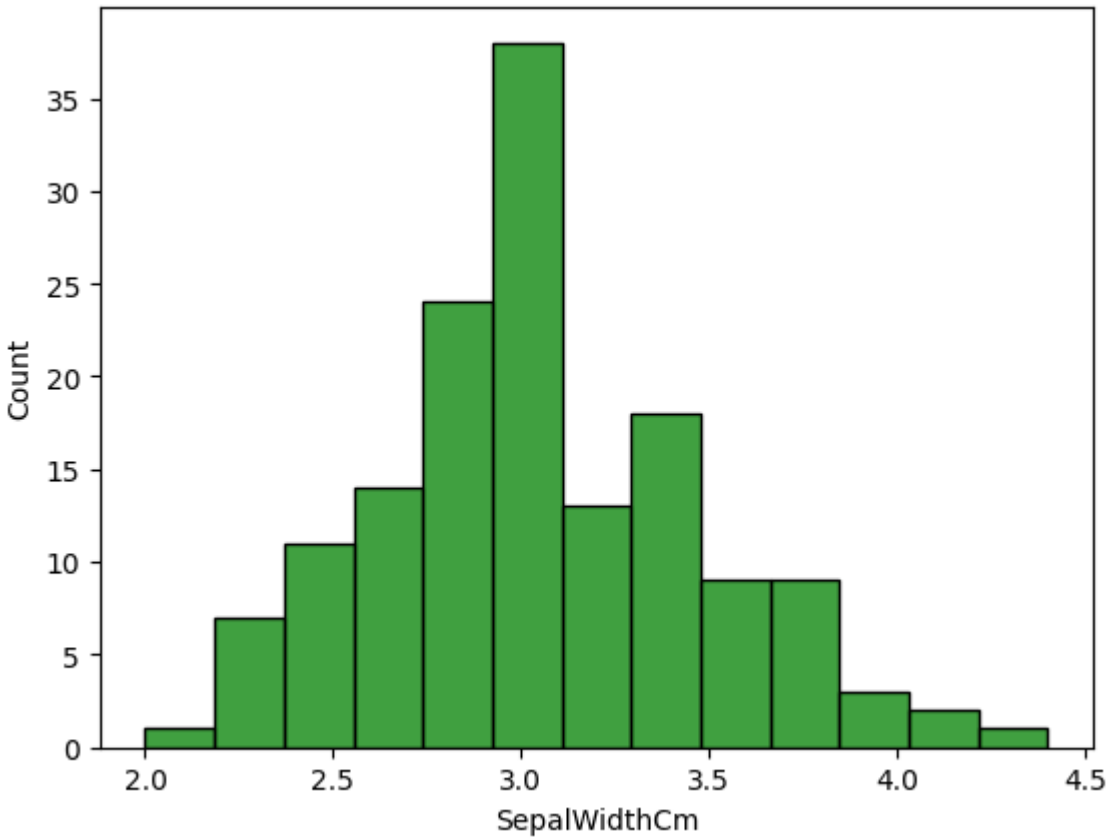

```
sns.histplot(df['SepalLengthCm'],color='b')
```

```
<Axes: xlabel='SepalLengthCm', ylabel='Count'>
```



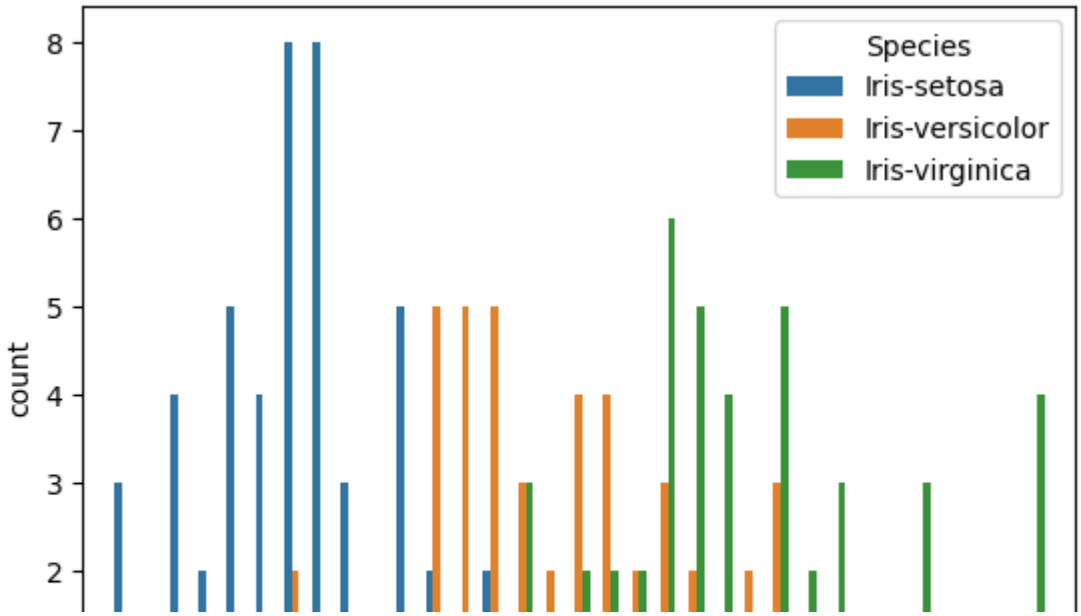

```
sns.histplot(df['SepalWidthCm'],color='g')
```

```
<Axes: xlabel='SepalWidthCm', ylabel='Count'>
```



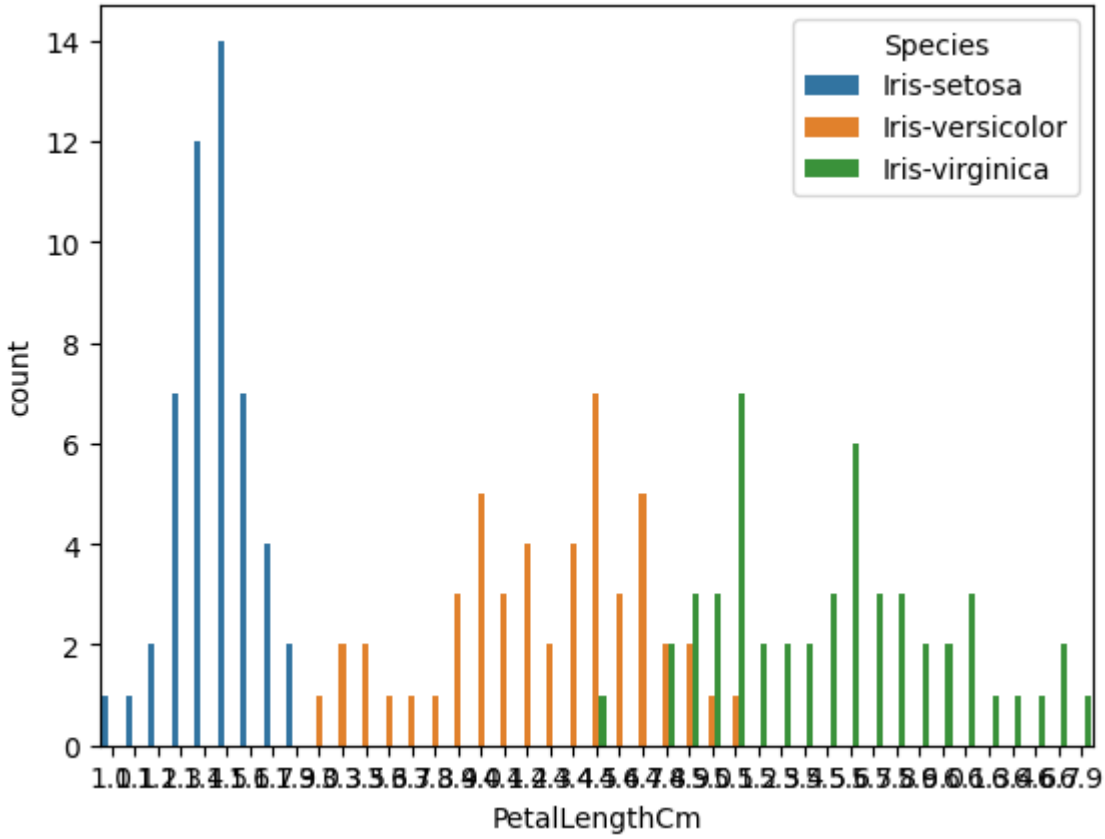

```
sns.countplot(x='SepalLengthCm',hue='Species',data=df)
```

```
<Axes: xlabel='SepalLengthCm', ylabel='count'>
```



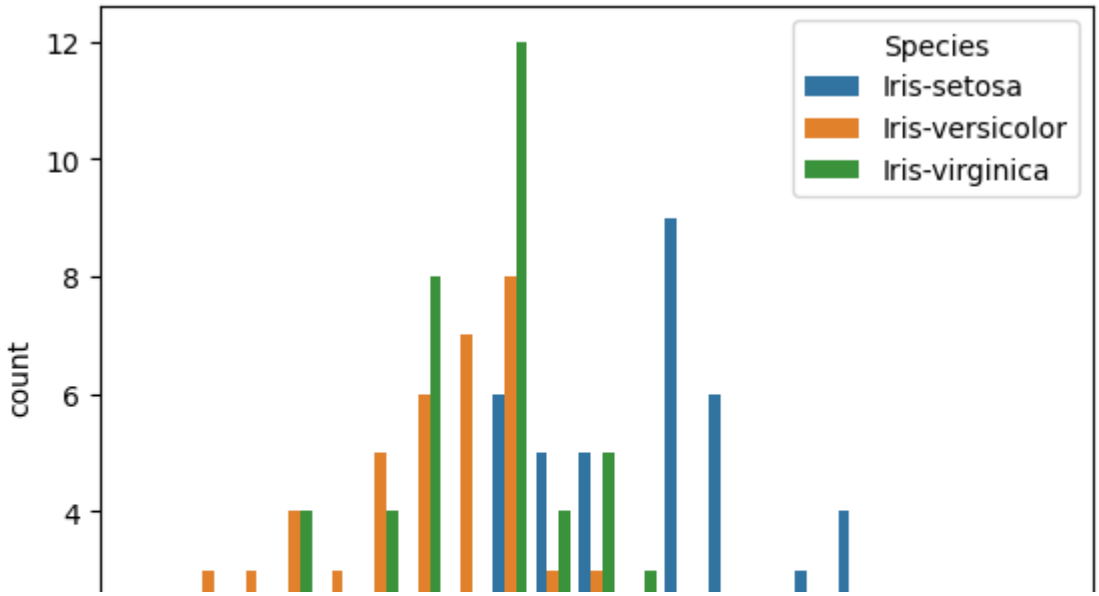

```
sns.countplot(x='PetalLengthCm',hue='Species',data=df)
```

```
<Axes: xlabel='PetalLengthCm', ylabel='count'>
```



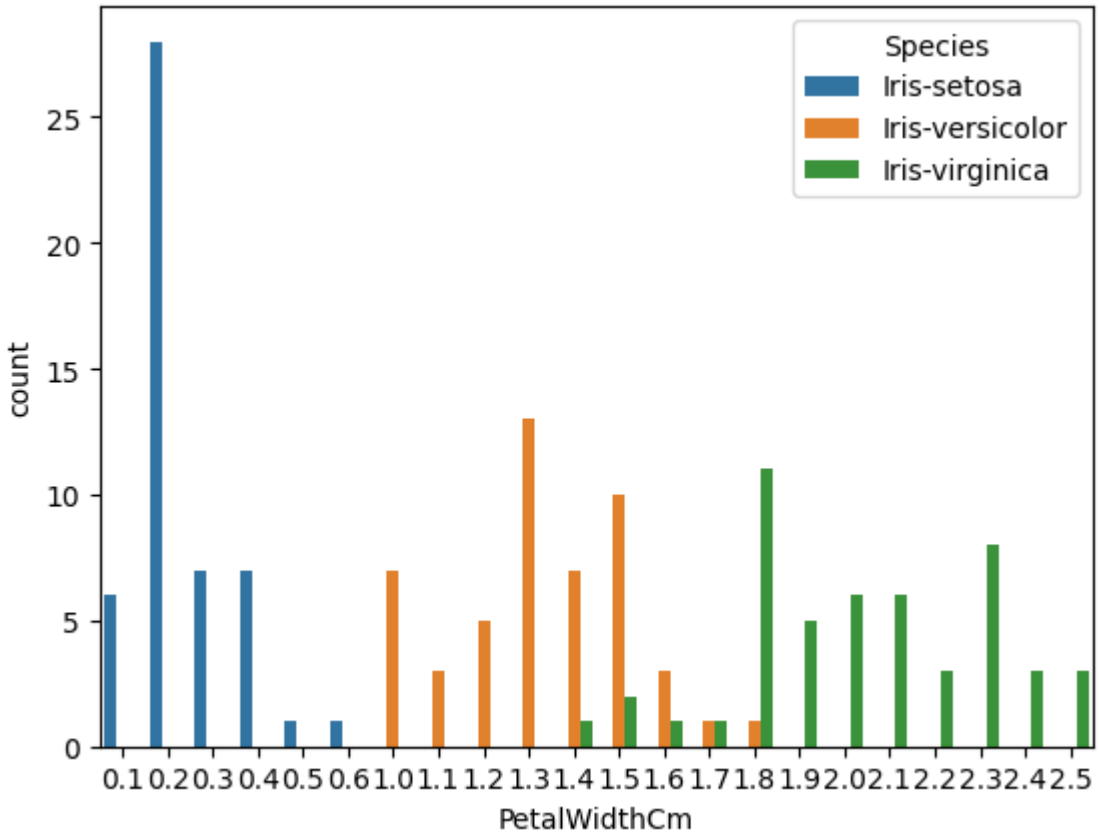

```
sns.countplot(x='SepalWidthCm',hue='Species',data=df)
```

```
<Axes: xlabel='SepalWidthCm', ylabel='count'>
```



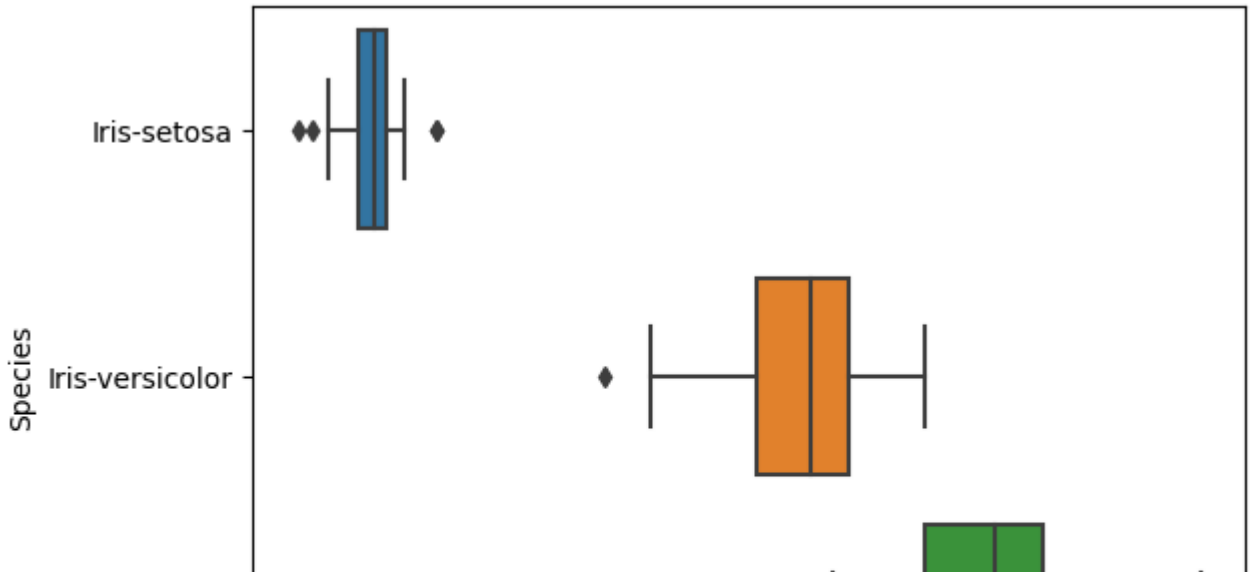

```
sns.countplot(x='PetalWidthCm',hue='Species',data=df)
```

```
<Axes: xlabel='PetalWidthCm', ylabel='count'>
```



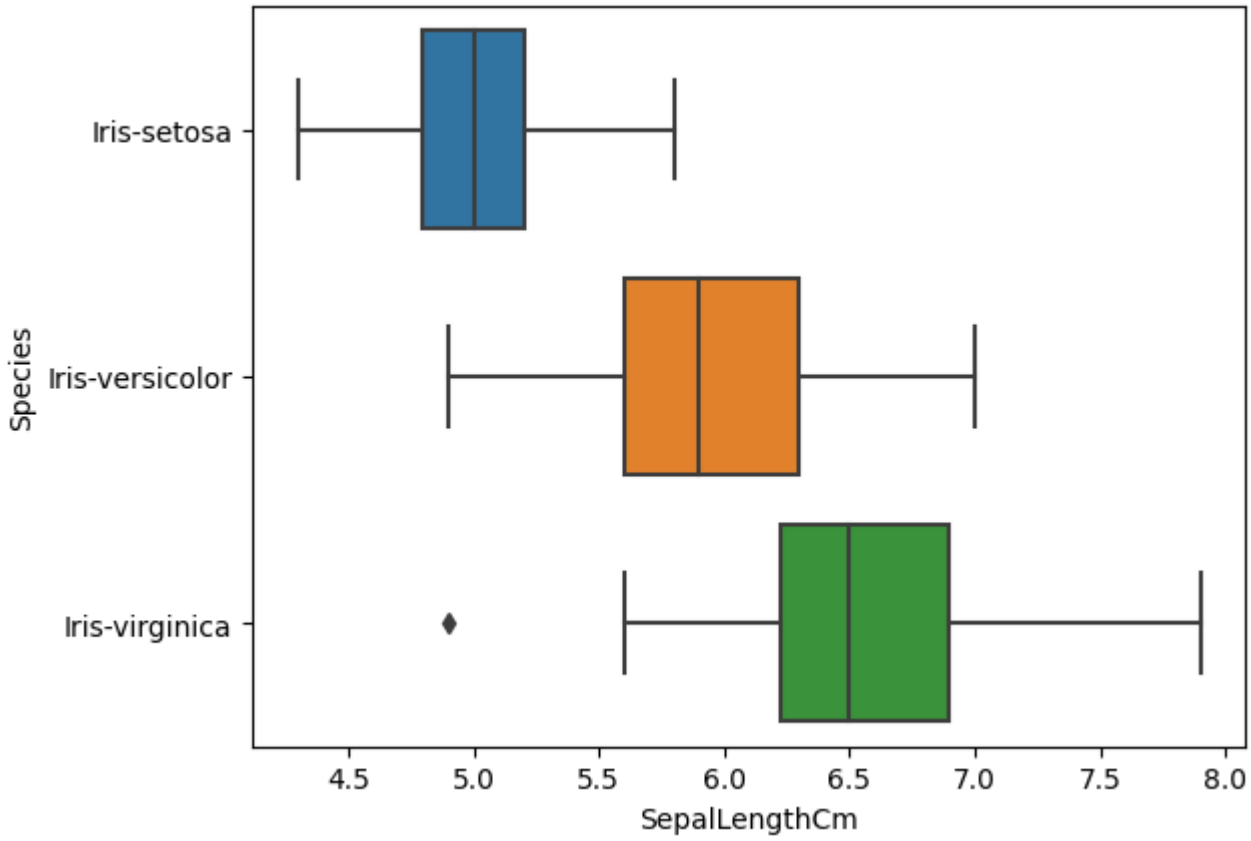

```
sns.boxplot(x=df['PetalLengthCm'],y=df['Species'])
```

```
<Axes: xlabel='PetalLengthCm', ylabel='Species'>
```



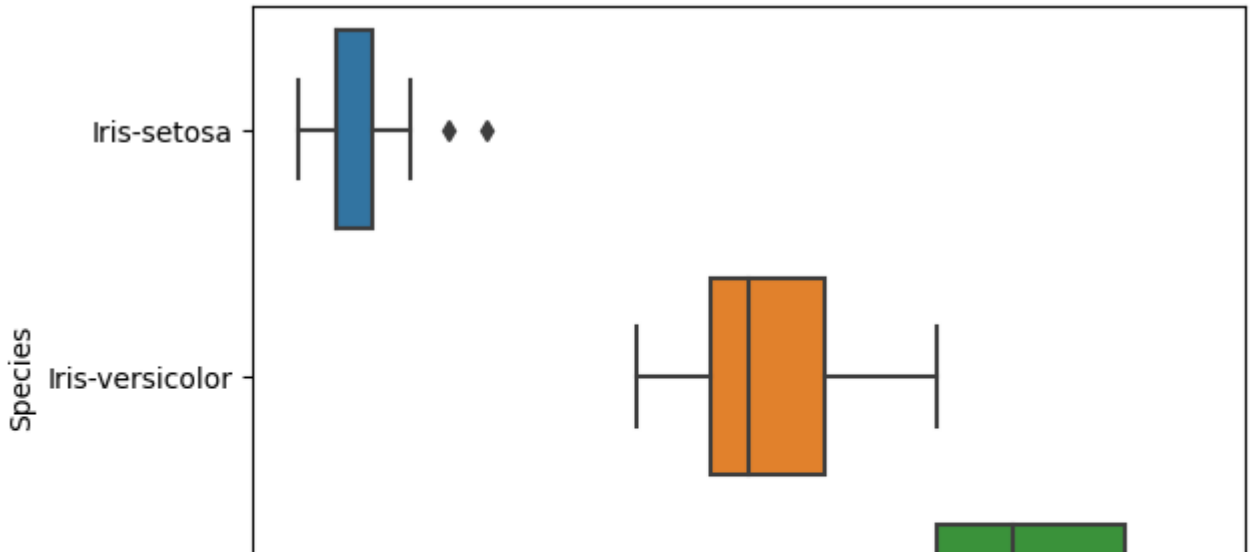

```
sns.boxplot(x=df['SepalLengthCm'],y=df['Species'])
```

```
<Axes: xlabel='SepalLengthCm', ylabel='Species'>
```



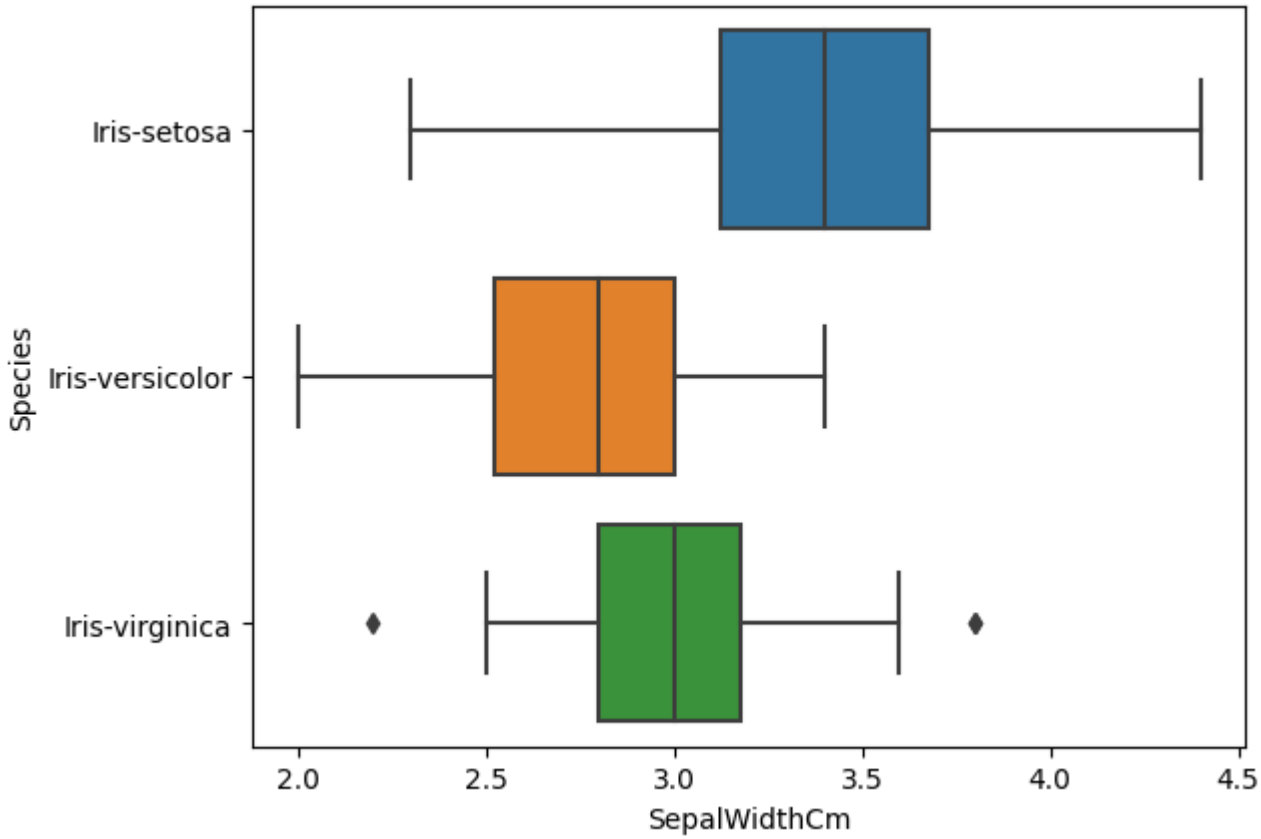

```
sns.boxplot(x=df['PetalWidthCm'],y=df['Species'])
```

```
<Axes: xlabel='PetalWidthCm', ylabel='Species'>
```



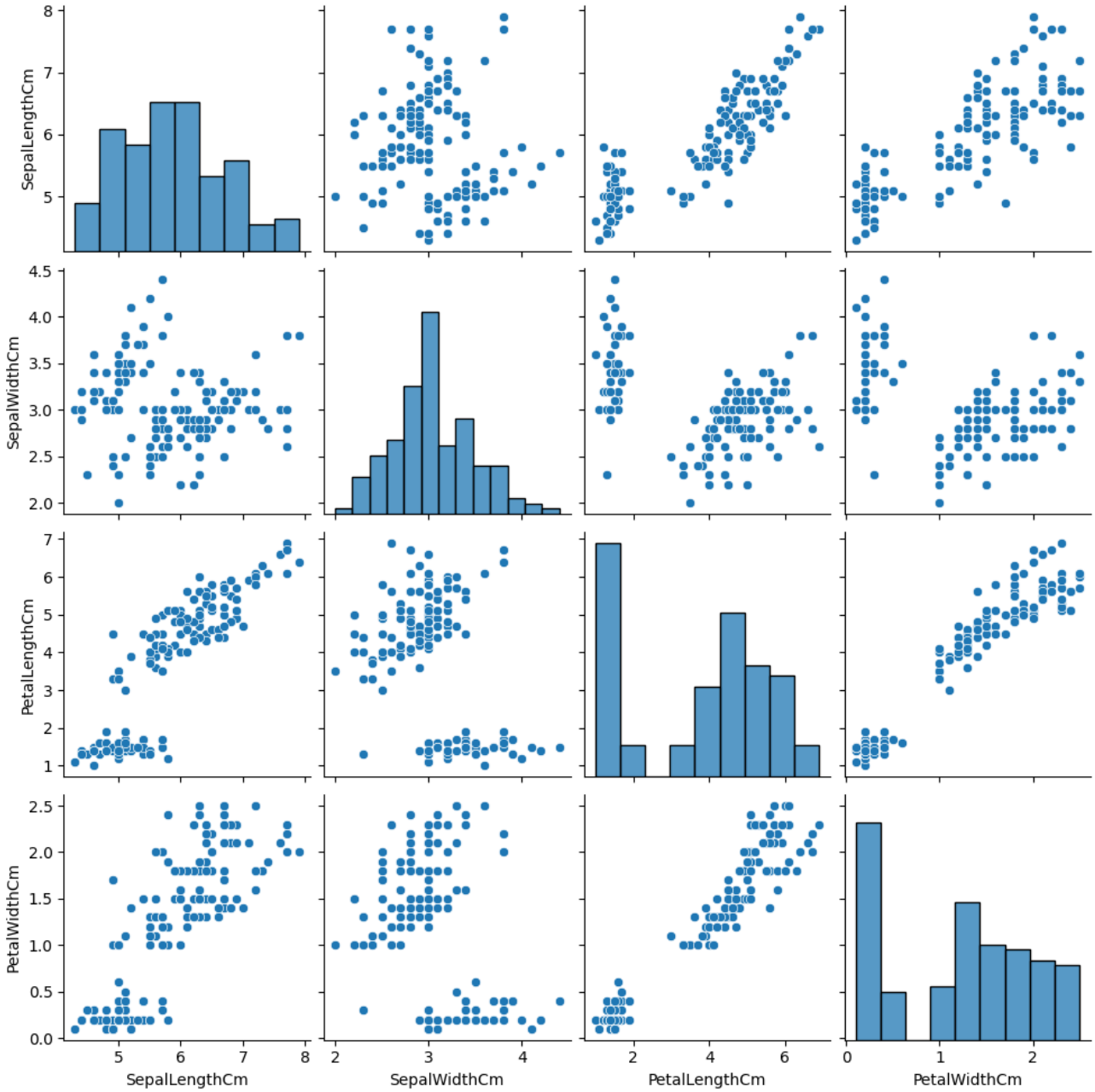

```
sns.boxplot(x=df['SepalWidthCm'],y=df['Species'])
```

```
<Axes: xlabel='SepalWidthCm', ylabel='Species'>
```




```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x78a7a40fa950>
```




```
sns.heatmap(df.corr(),annot=True)
```

```
<ipython-input-28-8df7bcac526d>:1: FutureWarning: The default value of numeric_only :
  sns.heatmap(df.corr(),annot=True)
<Axes: >
```

Separating x and y

```
x=df.drop(['Species'],axis=1)
y=df['Species']
```

Splitting training and testing data

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=42)
```

```
x_train
```

```
       [-1.37805466,  0.47275953, -1.50218532, -1.40173942],
       [ 0.3100623 , -0.25367584,  0.48403749,  0.21861991],
       [ 0.79238143, -0.49582097,  0.42561917,  0.35364985],
       [ 0.43064208, -0.49582097,  0.54245581,  0.75873969],
       [ 1.39528035,  0.47275953,  0.48403749,  0.21861991],
       [ 0.67180165,  0.47275953,  0.8345474 ,  1.43388941],
       [-0.89573553,  1.92563026, -1.32693037, -1.40173942],
       [ 1.27470056,  0.2306144 ,  0.89296572,  1.16382952],
       [ 0.06890273, -0.01153072,  0.1919459 ,  0.35364985],
       [ 0.79238143, -0.01153072,  0.77612908,  1.02879957],
       [-0.17225683, -0.98011121, -0.21698232, -0.32149987],
       [-0.77515575, -0.73796609,  0.01669095,  0.21861991],
       [ 0.3100623 , -0.01153072,  0.42561917,  0.21861991],
       [-1.61921423, -1.70654658, -1.50218532, -1.26670948],
       [ 0.91296121, -0.25367584,  0.42561917,  0.08358997],
       [-0.4134164 , -0.98011121,  0.30878254, -0.05143998],
       [-0.65457597,  1.68348514, -1.38534869, -1.40173942],
       [-0.29283662, -0.01153072,  0.13352758,  0.08358997],
       [ 1.7570197 , -0.25367584,  1.41873058,  0.75873969],
       [ 1.033541  ,  0.71490465,  1.06822067,  1.16382952],
       [-0.89573553,  1.68348514, -1.38534869, -1.13167953],
       [-1.1368951 , -1.46440146, -0.33381896, -0.32149987],
       [ 1.033541  ,  0.71490465,  1.06822067,  1.7039493 ],
       [ 1.63643991, -0.01153072,  1.12663899,  0.4886798 ],
       [-1.1368951 ,  0.2306144 , -1.38534869, -1.53676936],
       [ 1.033541  ,  0.2306144 ,  1.00980236,  1.56891935],
       [-1.1368951 , -0.01153072, -1.443767  , -1.40173942],
       [ 1.27470056,  0.2306144 ,  0.60087413,  0.35364985],
       [ 1.87759948, -0.49582097,  1.30189395,  0.89376963],
       [ 0.55122187, -0.25367584,  1.00980236,  0.75873969],
       [-0.17225683, -0.49582097,  0.13352758,  0.08358997],
       [ 0.79238143, -0.01153072,  0.95138404,  0.75873969],
       [ 0.55122187, -1.70654658,  0.30878254,  0.08358997],
       [ 0.67180165, -0.25367584,  0.25036422,  0.08358997],
       [-0.29283662, -0.49582097,  0.60087413,  1.02879957],
       [ 0.06890273, -0.01153072,  0.71771076,  0.75873969],
       [-0.53399618,  0.95704977, -1.26851205, -1.40173942],
       [ 0.3100623 , -0.49582097,  0.07510927,  0.08358997],
       [-1.1368951 , -1.22225633,  0.36720086,  0.62370974],
       [-0.05167705,  2.40992051, -1.56060364, -1.40173942],
       [-0.05167705, -0.98011121,  0.07510927, -0.05143998],
       [ 1.51586013, -0.01153072,  1.18505731,  1.16382952]])
```

```
x_test
```

```
array([[ 0.3100623 , -0.49582097,  0.48403749, -0.05143998],
       [-0.17225683,  1.92563026, -1.26851205, -1.26670948],
       [ 2.23933883, -0.98011121,  1.76924049,  1.43388941],
       [ 0.18948252, -0.25367584,  0.36720086,  0.35364985],
       [ 1.15412078, -0.49582097,  0.54245581,  0.21861991],
       [-0.53399618,  0.95704977, -1.38534869, -1.13167953],
       [-0.29283662, -0.25367584, -0.15856401,  0.08358997],
       [ 1.27470056,  0.2306144 ,  0.71771076,  1.43388941],
       [ 0.43064208, -1.9486917 ,  0.36720086,  0.35364985],
       [-0.05167705, -0.73796609,  0.01669095, -0.05143998],
       [ 0.79238143,  0.47275953,  0.71771076,  1.02879957],
       [-1.25747488, -0.01153072, -1.443767  , -1.53676936],
       [-0.4134164 ,  1.19919489, -1.50218532, -1.40173942],
       [-1.1368951 ,  0.2306144 , -1.38534869, -1.53676936],
       [-0.89573553,  1.92563026, -1.38534869, -1.26670948],
```

```
       [ 0.55122187,  0.71490465,  0.48403749,  0.4886798 ],
       [ 0.79238143, -0.01153072,  1.12663899,  1.29885946],
       [-0.29283662, -1.22225633,  0.01669095, -0.18646992],
       [-0.17225683, -0.49582097,  0.36720086,  0.08358997],
       [ 0.67180165, -0.49582097,  1.00980236,  1.29885946],
       [-1.37805466,  0.47275953, -1.32693037, -1.40173942],
       [ 0.3100623 , -0.01153072,  0.60087413,  0.75873969],
       [-1.01631531,  0.95704977, -1.32693037, -1.13167953],
       [ 0.67180165, -0.49582097,  1.00980236,  1.16382952],
       [ 2.4804984 ,  1.92563026,  1.4771489 ,  1.02879957],
       [ 1.033541  , -0.01153072,  0.77612908,  1.43388941],
       [ 1.033541  , -1.22225633,  1.12663899,  0.75873969],
       [ 1.15412078,  0.47275953,  1.18505731,  1.43388941],
       [-1.25747488, -0.01153072, -1.443767  , -1.26670948],
       [-1.25747488,  0.2306144 , -1.32693037, -1.40173942],
       [-1.49863445,  1.44134002, -1.67744028, -1.40173942],
       [-0.17225683,  3.378501  , -1.38534869, -1.13167953],
       [ 1.033541  ,  0.2306144 ,  0.30878254,  0.21861991],
       [-1.25747488,  0.95704977, -1.32693037, -1.40173942],
       [-1.73979401,  0.47275953, -1.50218532, -1.40173942],
       [ 0.55122187, -1.22225633,  0.65929245,  0.89376963],
       [ 0.67180165,  0.47275953,  0.36720086,  0.35364985],
       [-0.77515575,  1.19919489, -1.38534869, -1.40173942],
       [-1.01631531,  1.44134002, -1.443767  , -1.40173942],
       [-0.77515575,  2.65206563, -1.38534869, -1.53676936],
       [-0.05167705, -0.73796609,  0.71771076,  0.89376963],
       [ 0.18948252,  0.95704977,  0.36720086,  0.4886798 ],
       [ 1.033541  ,  0.2306144 ,  0.48403749,  0.35364985],
       [-0.53399618,  2.16777538, -1.50218532, -1.13167953],
       [-0.53399618,  1.68348514, -1.38534869, -1.40173942]])
```

```
y_train
```

```
81     Iris-versicolor
133     Iris-virginica
137     Iris-virginica
75     Iris-versicolor
109     Iris-virginica
           ...
71     Iris-versicolor
106     Iris-virginica
14         Iris-setosa
92     Iris-versicolor
102     Iris-virginica
Name: Species, Length: 105, dtype: object
```

```
y_test
```

```
73     Iris-versicolor
18         Iris-setosa
118     Iris-virginica
78     Iris-versicolor
76     Iris-versicolor
31         Iris-setosa
64     Iris-versicolor
141     Iris-virginica
68     Iris-versicolor
82     Iris-versicolor
```

```
110     Iris-virginica
12         Iris-setosa
36         Iris-setosa
9          Iris-setosa
19         Iris-setosa
56     Iris-versicolor
104     Iris-virginica
69     Iris-versicolor
55     Iris-versicolor
132     Iris-virginica
29         Iris-setosa
127     Iris-virginica
26         Iris-setosa
128     Iris-virginica
131     Iris-virginica
145     Iris-virginica
108     Iris-virginica
143     Iris-virginica
45         Iris-setosa
30         Iris-setosa
22         Iris-setosa
15         Iris-setosa
65     Iris-versicolor
11         Iris-setosa
42         Iris-setosa
146     Iris-virginica
51     Iris-versicolor
27         Iris-setosa
4          Iris-setosa
32         Iris-setosa
142     Iris-virginica
85     Iris-versicolor
86     Iris-versicolor
16         Iris-setosa
10         Iris-setosa
Name: Species, dtype: object
```

Performing Normalization

```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(x_train)
x_train=scaler.transform(x_train)
x_test=scaler.transform(x_test)
```

Model Creation

```
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=5)
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
y_pred
```

```
array(['Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',
```

```
              'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',
              'Iris-versicolor', 'Iris-virginica', 'Iris-setosa', 'Iris-setosa',
              'Iris-setosa', 'Iris-setosa', 'Iris-versicolor', 'Iris-virginica',
              'Iris-versicolor', 'Iris-versicolor', 'Iris-virginica',
              'Iris-setosa', 'Iris-virginica', 'Iris-setosa', 'Iris-virginica',
              'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
              'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
              'Iris-setosa', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',
              'Iris-virginica', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',
              'Iris-setosa', 'Iris-virginica', 'Iris-versicolor',
              'Iris-versicolor', 'Iris-setosa', 'Iris-setosa'], dtype=object)
```

```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,Confus
print("accuracy score is",accuracy_score(y_test,y_pred))
```

```
accuracy score is 1.0
```

```
print("classification report is",classification_report(y_test,y_pred))
```

```
classification report is                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        19
Iris-versicolor       1.00      1.00      1.00        13
 Iris-virginica       1.00      1.00      1.00        13

       accuracy                           1.00        45
      macro avg       1.00      1.00      1.00        45
   weighted avg       1.00      1.00      1.00        45
```

```
labels=['Iris-setosa','Iris-versicolor','Iris-virginica']
result=confusion_matrix(y_test,y_pred)
cmd=ConfusionMatrixDisplay(result,display_labels=labels)
cmd.plot()
print(result)
```

```
[[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
```