IRIS DATASET VISUALIZATION (SEABORN + MATPLOTLIB)

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
iris = pd.read_csv(r'C:\Users\Hp\Desktop\NAYAN\DATA SCIENCE\CSV_FILES\Iris.csv
iris
```

Out[2]:

	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 [3]:
```
iris.head()
```

Out[3]:

	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 [4]:
```
iris.drop('Id', axis=1, inplace=False)
# if inplace is False return datset and remove 'Id' column
```

Out[4]:

	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
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 [5]:
```
iris.drop('Id', axis=1, inplace=True)
# if inplace is True return error because now 'Id' column has removed
```

In [6]:
```
iris.head()
```

Out[6]:

	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
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 [7]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   SepalLengthCm  150 non-null    float64
 1   SepalWidthCm   150 non-null    float64
 2   PetalLengthCm  150 non-null    float64
 3   PetalWidthCm   150 non-null    float64
 4   Species        150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

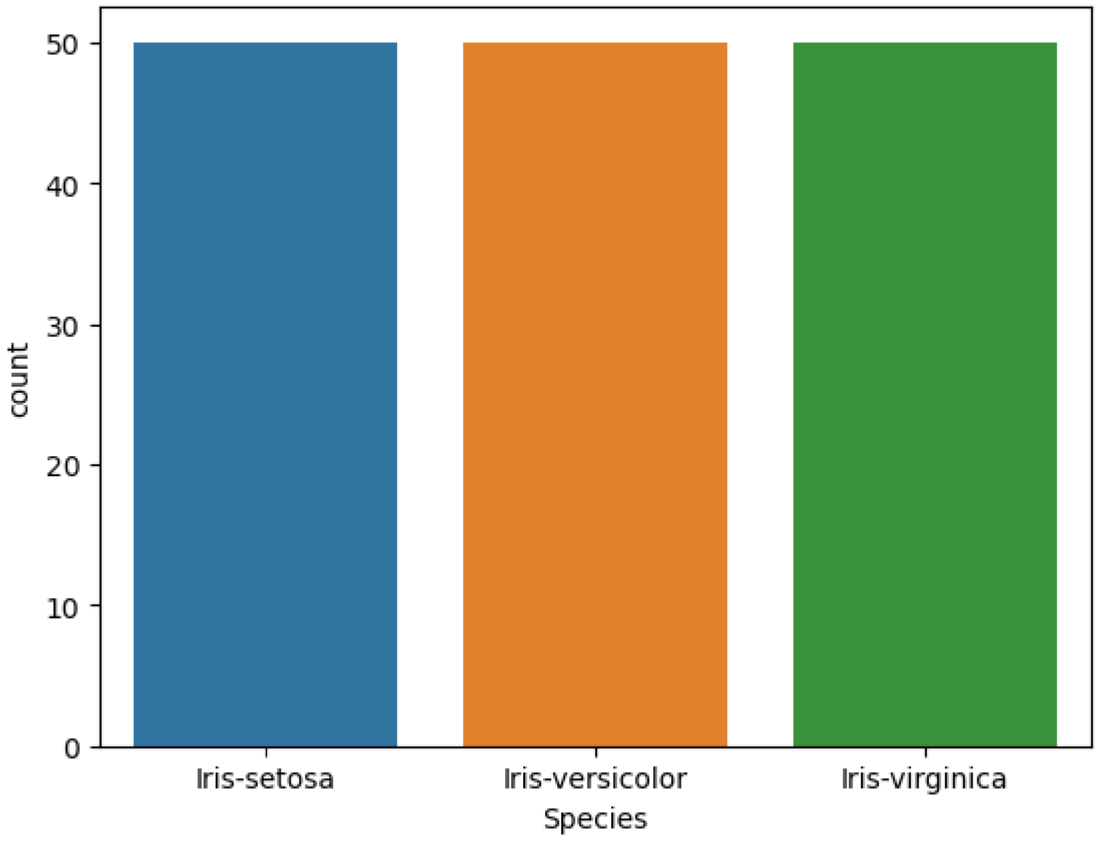
```
In [8]: iris['Species'].value_counts()
```

```
Out[8]: Iris-setosa        50
        Iris-versicolor    50
        Iris-virginica     50
        Name: Species, dtype: int64
```

Bar Plot

Here the frequency of the observation is plotted.In this case we are plotting the frequency of the three species in the Iris Dataset

```
In [9]: sns.countplot(x = 'Species', data=iris)
        plt.show()
```



Joint plot

** Jointplot is seaborn library specific and can be used to quickly visualize and analyze the relationship between two variables and describe their individual distributions on the same plot.

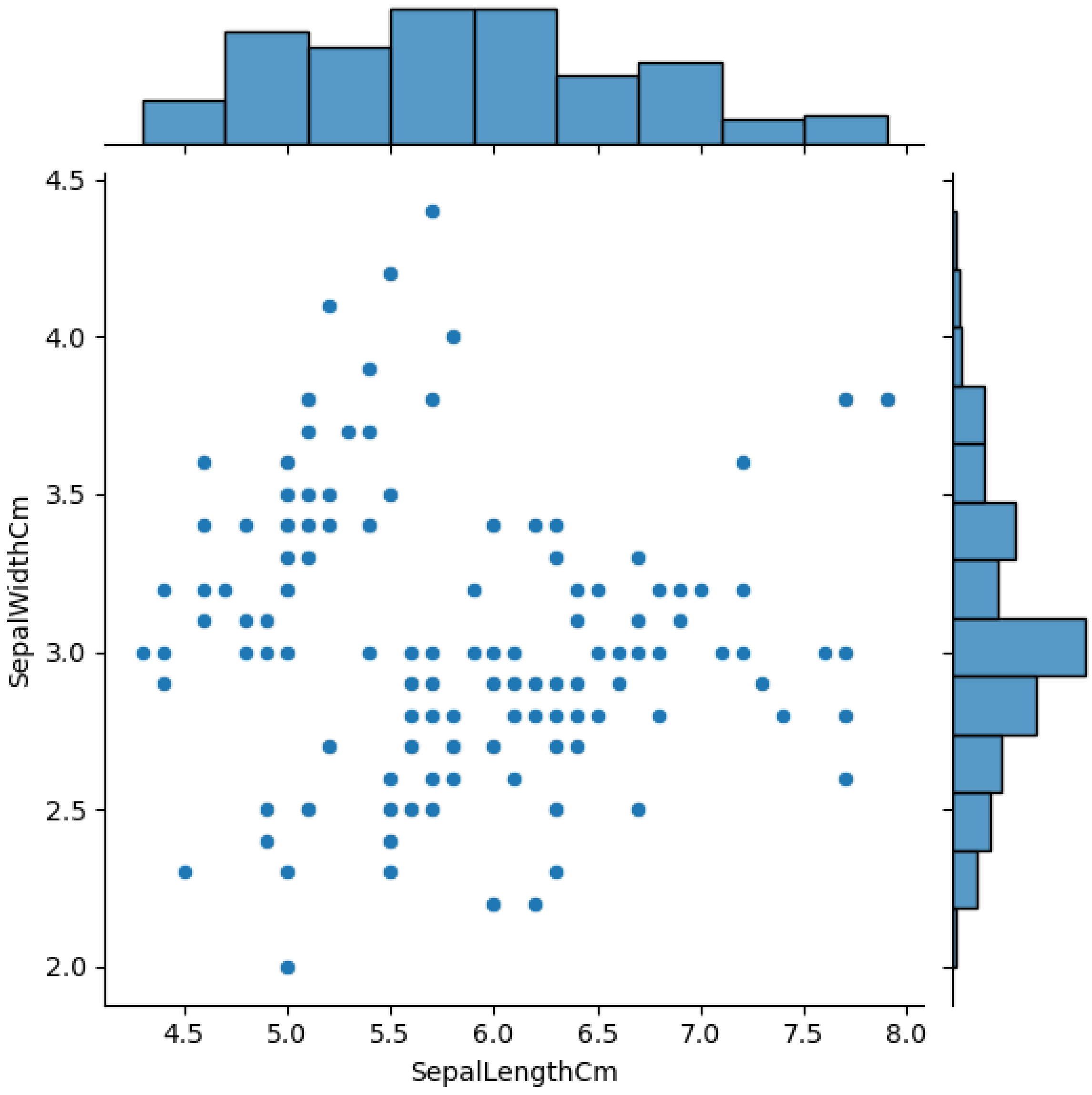
```
In [10]: iris.head()
```

Out[10]:

	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
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

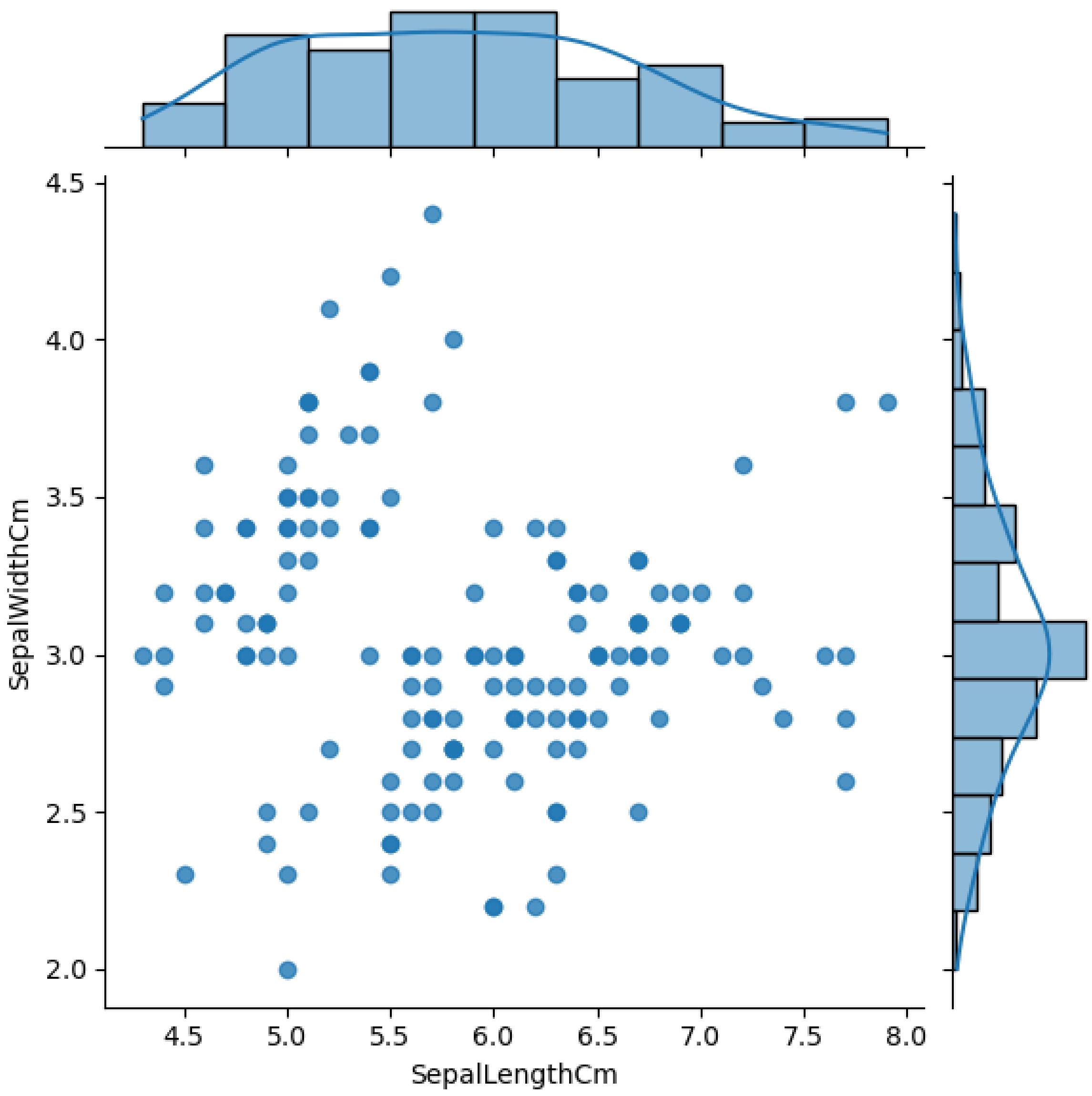
```
In [11]: sns.jointplot(x='SepalLengthCm',y='SepalWidthCm', data = iris)  # by default r
```

```
Out[11]: <seaborn.axisgrid.JointGrid at 0x2069edfed70>
```



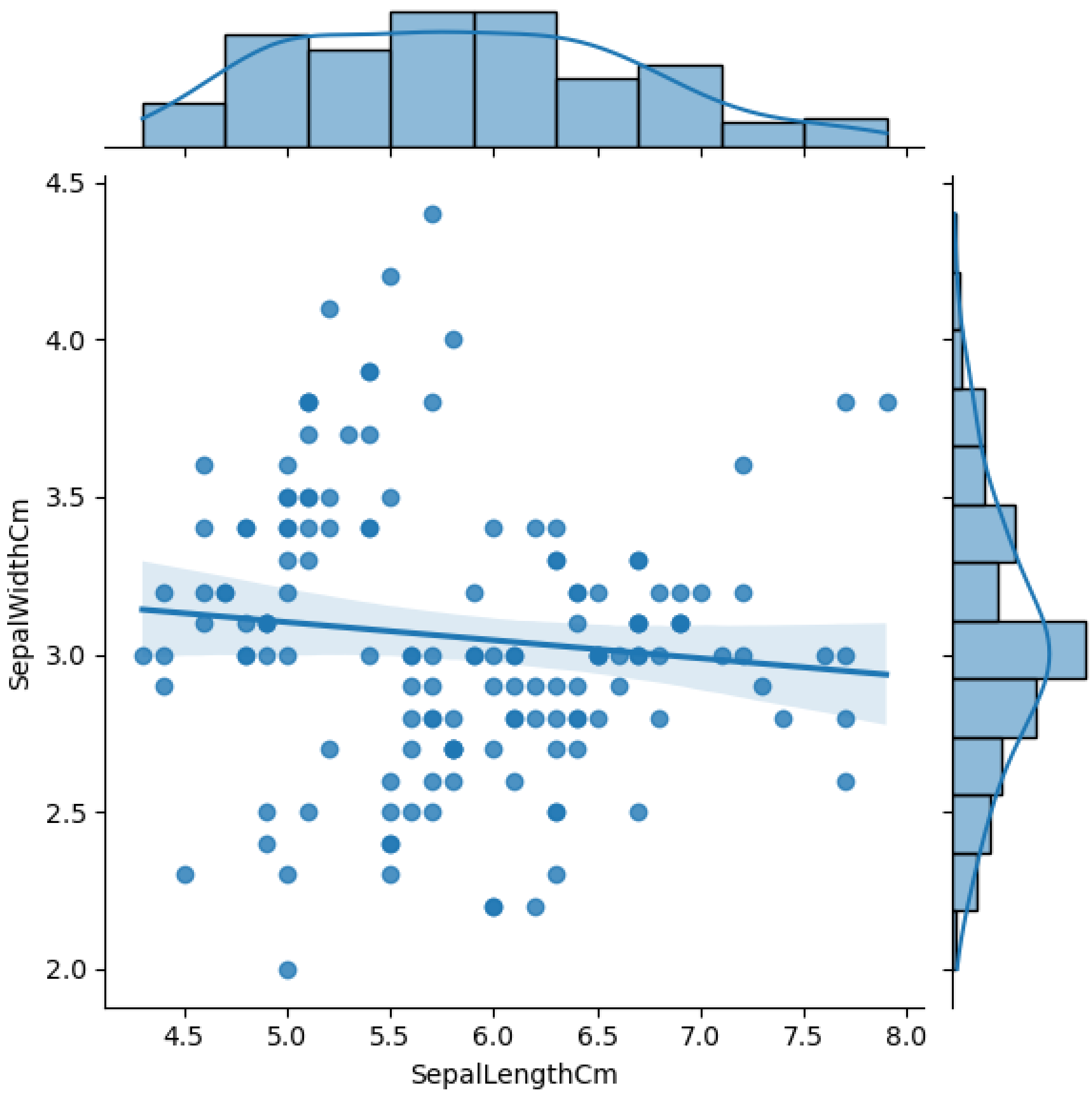
```
In [12]: # kind : { "scatter" | "kde" | "hist" | "hex" | "reg" | "resid" }
In [13]: sns.jointplot(x='SepalLengthCm',y='SepalWidthCm', data=iris, kind='reg', fit_r
```

```
Out[13]: <seaborn.axisgrid.JointGrid at 0x2069f95b370>
```

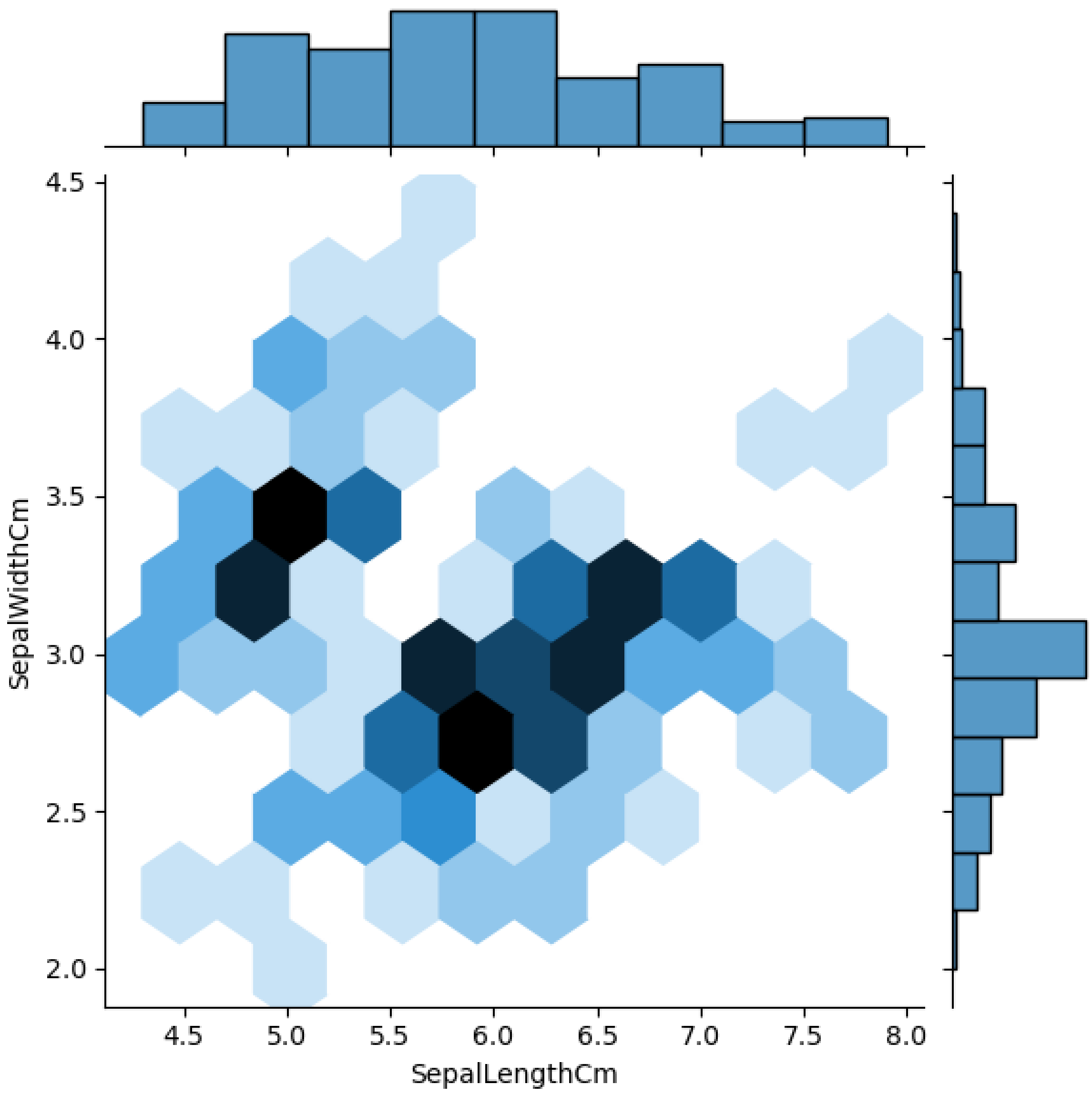


```
In [14]: sns.jointplot(x='SepalLengthCm',y='SepalWidthCm', data=iris, kind='reg')
```

```
Out[14]: <seaborn.axisgrid.JointGrid at 0x2069fc475e0>
```



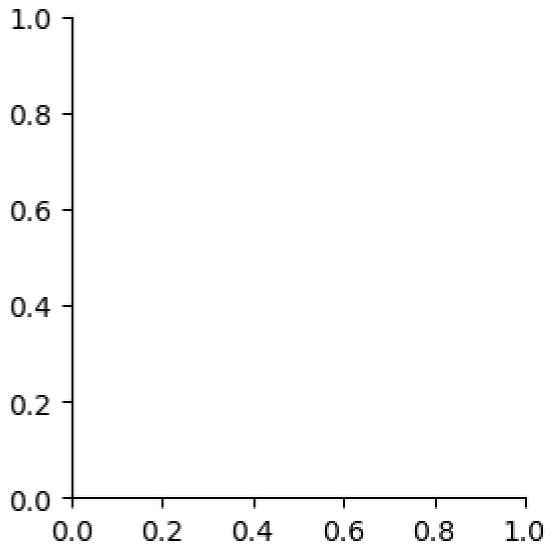
```
In [15]: fig = sns.jointplot(x='SepalLengthCm',y='SepalWidthCm', data=iris, kind='hex')
```



FacetGrid Plot

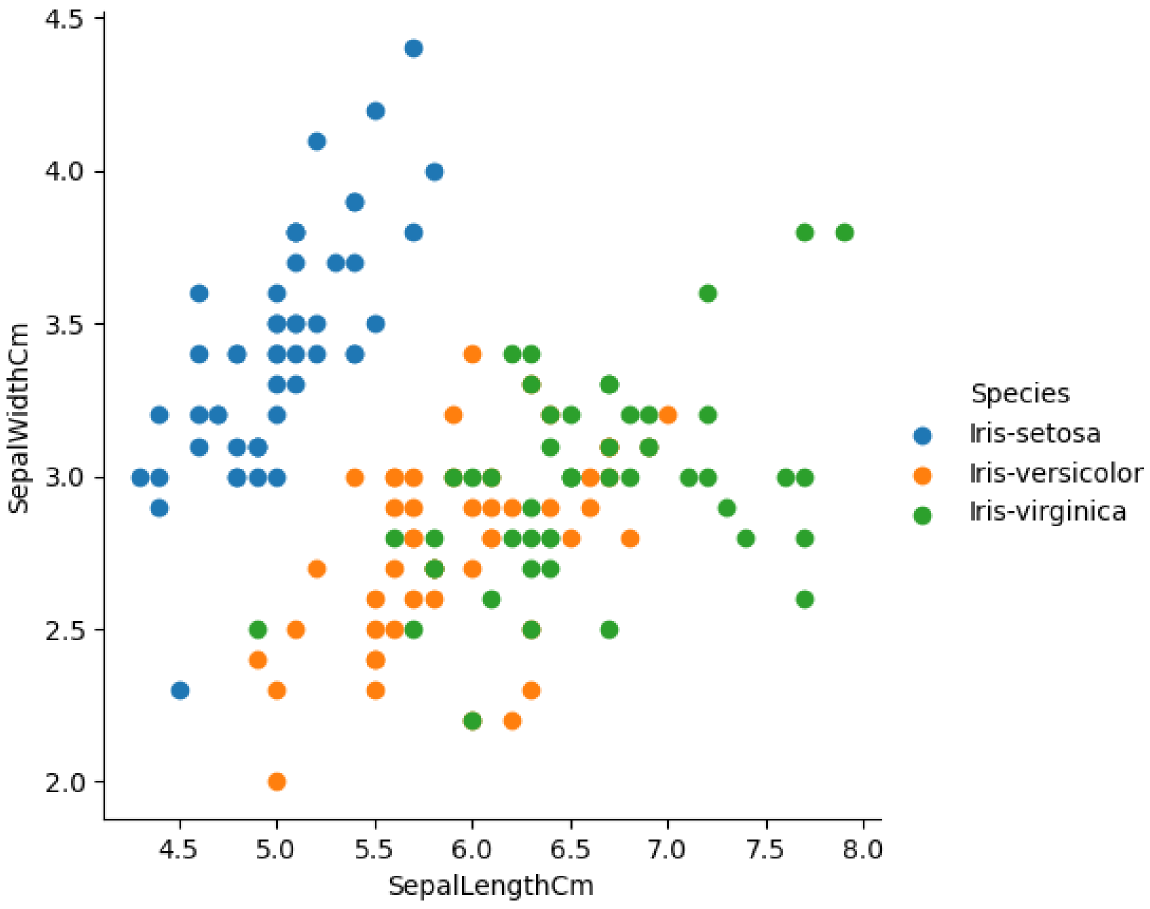
In [16]:
```
import matplotlib.pyplot as plt
%matplotlib inline
sns.FacetGrid(data = iris, hue = 'Species')
```

Out[16]:
```
<seaborn.axisgrid.FacetGrid at 0x206a5369ae0>
```



```
In [17]: sns.FacetGrid(data = iris, hue = 'Species', height= 5)\
         .map(plt.scatter, 'SepalLengthCm', 'SepalWidthCm')\
         .add_legend()
         # sns.FacetGrid(data = iris, hue = 'Species', height= 5).map(plt.scatter, 'Sep
```

```
Out[17]: <seaborn.axisgrid.FacetGrid at 0x206a5714dc0>
```



Boxplot or Whisker plot

```
In [18]: iris.head()
```

Out[18]:

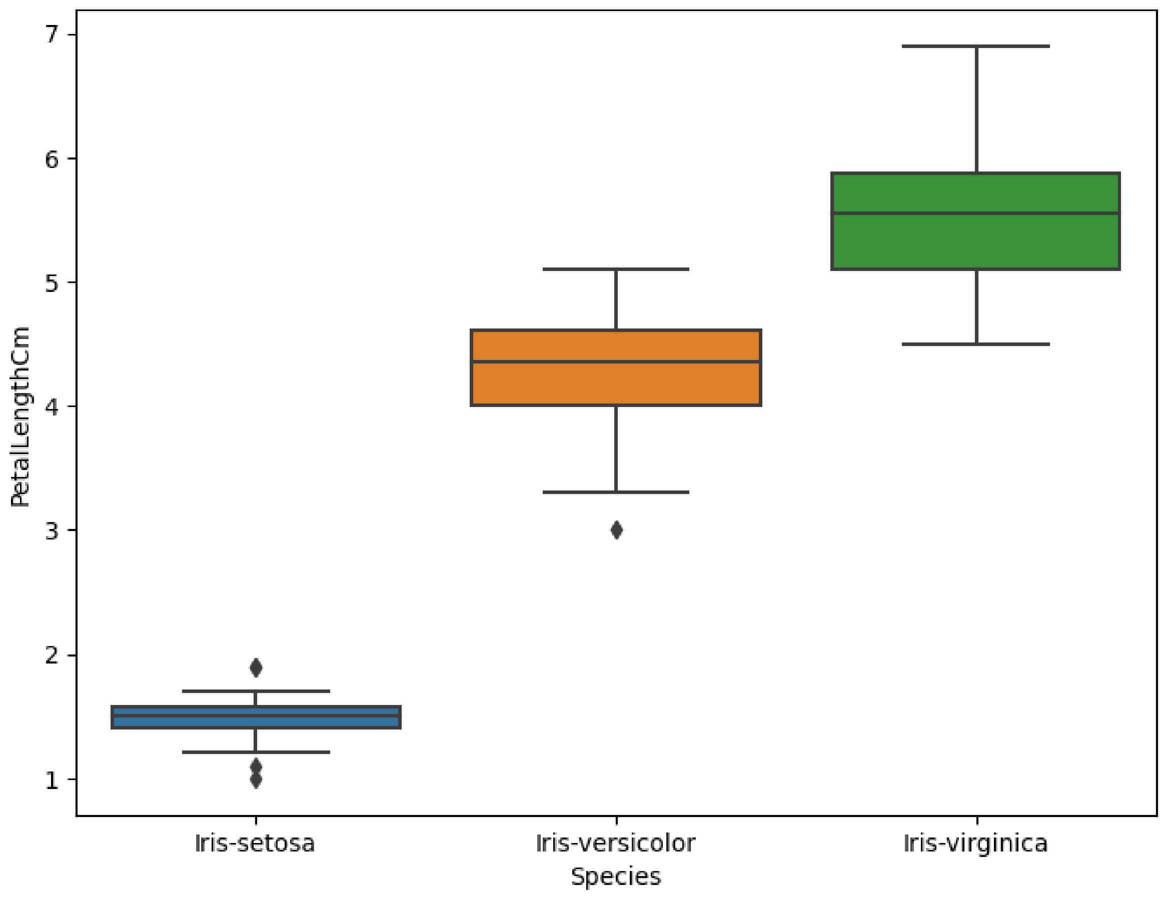
	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
In [19]: plt.gcf()  # gcf means get the current figure
```

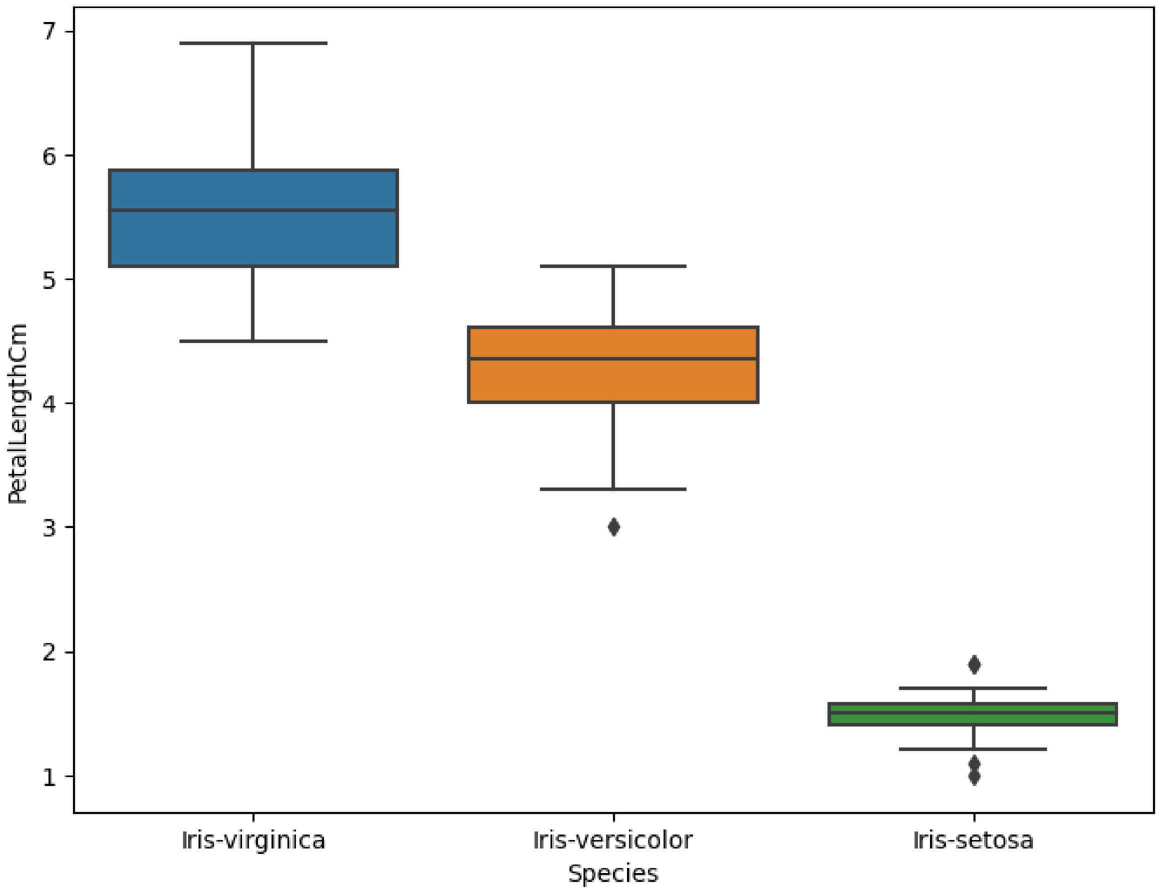
```
Out[19]: <Figure size 640x480 with 0 Axes>

         <Figure size 640x480 with 0 Axes>
```

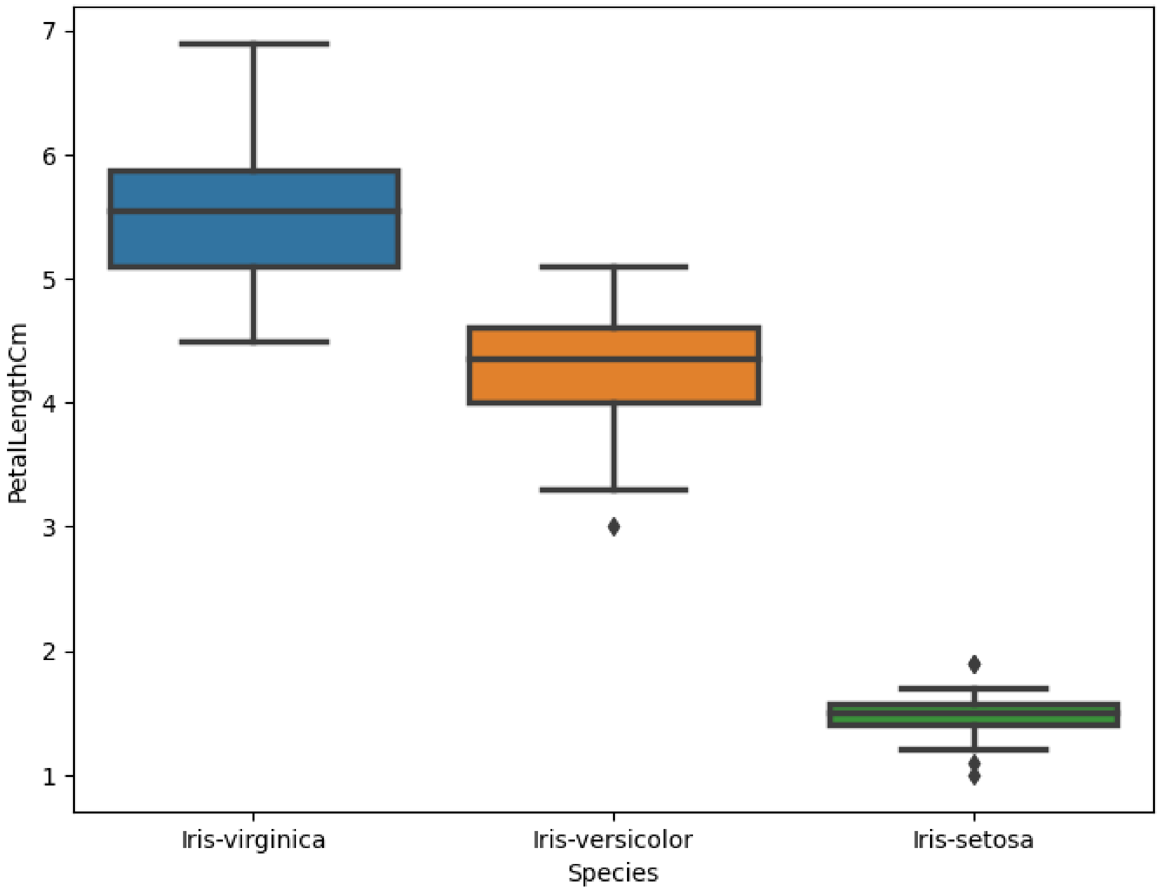
In [20]:
```
fig = plt.gcf()
fig.set_size_inches(8, 6)
fig = sns.boxplot(x='Species',y='PetalLengthCm', data = iris)
```



In [21]:
```
fig = plt.gcf()
fig.set_size_inches(8, 6)
fig = sns.boxplot(x='Species',y='PetalLengthCm', data = iris, order = ['Iris-v
```

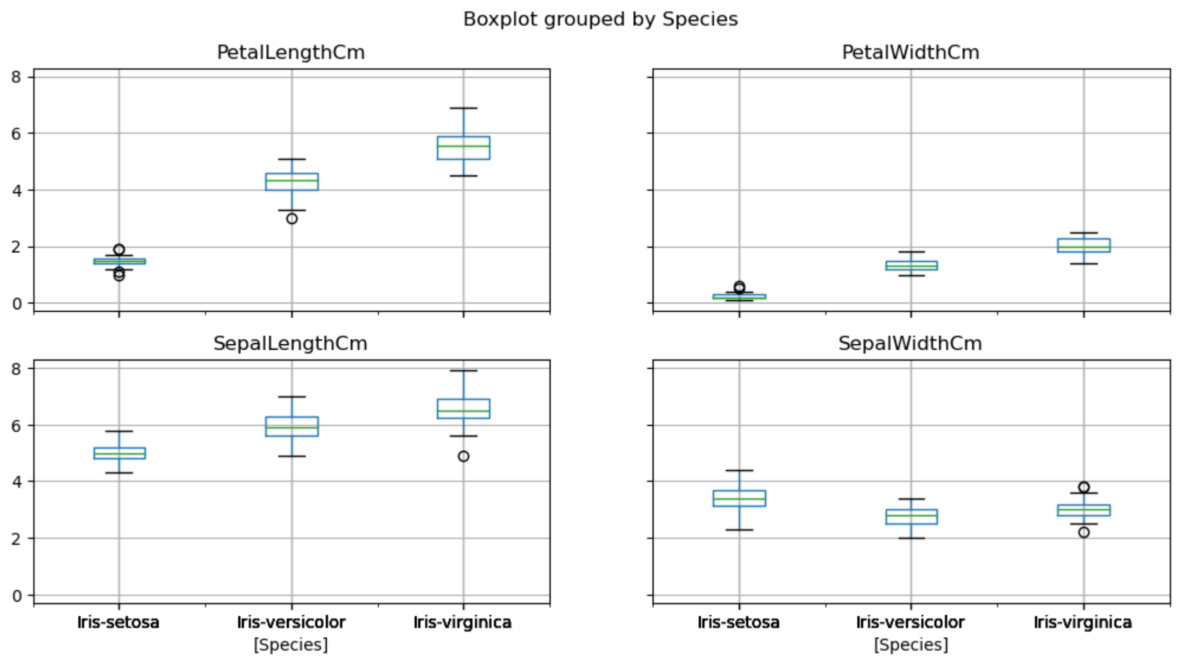


In [22]:
```
fig = plt.gcf()
fig.set_size_inches(8, 6)
fig = sns.boxplot(x='Species',y='PetalLengthCm', data = iris, order = ['Iris-v
```



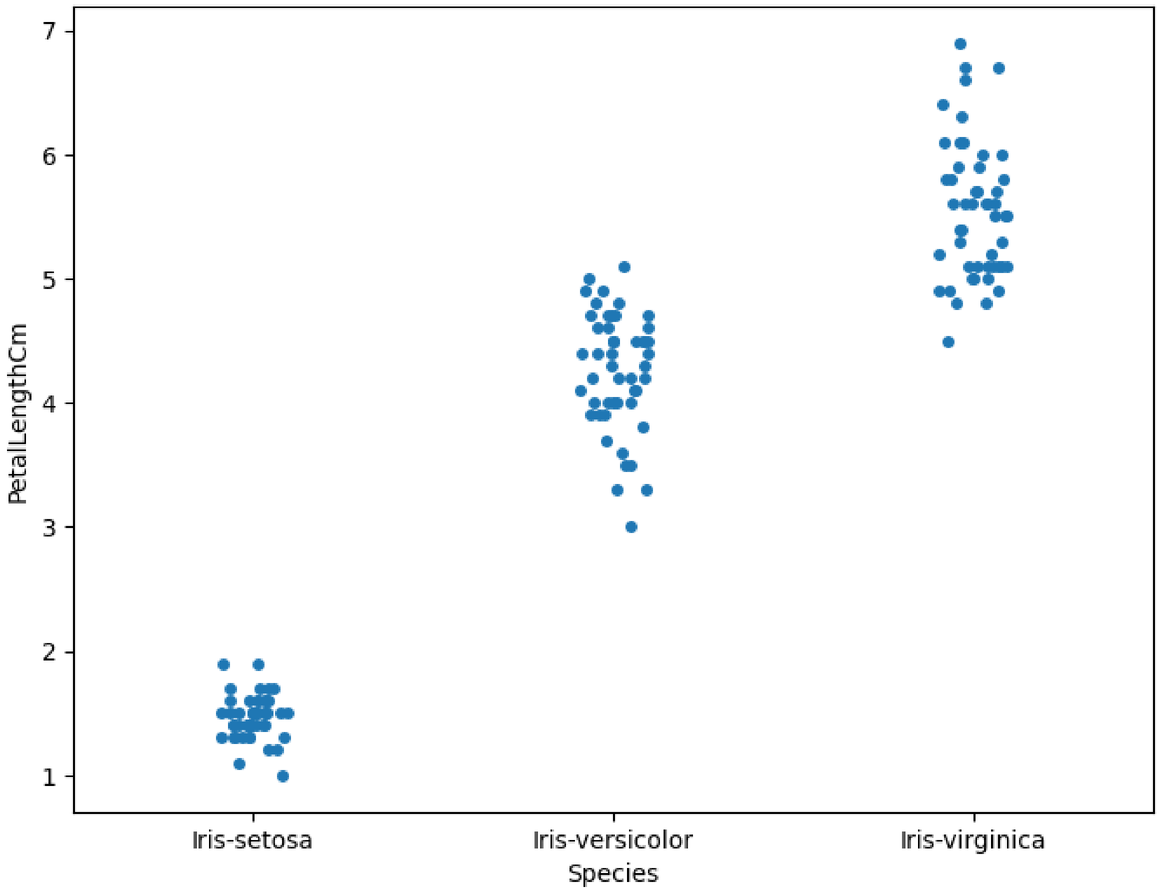
```
In [23]: iris.boxplot(by ="Species", figsize = (12, 6))
```

```
Out[23]: array([[<Axes: title={'center': 'PetalLengthCm'}, xlabel='[Species]'>,
                 <Axes: title={'center': 'PetalWidthCm'}, xlabel='[Species]'>],
                [<Axes: title={'center': 'SepalLengthCm'}, xlabel='[Species]'>,
                 <Axes: title={'center': 'SepalWidthCm'}, xlabel='[Species]'>]],
               dtype=object)
```

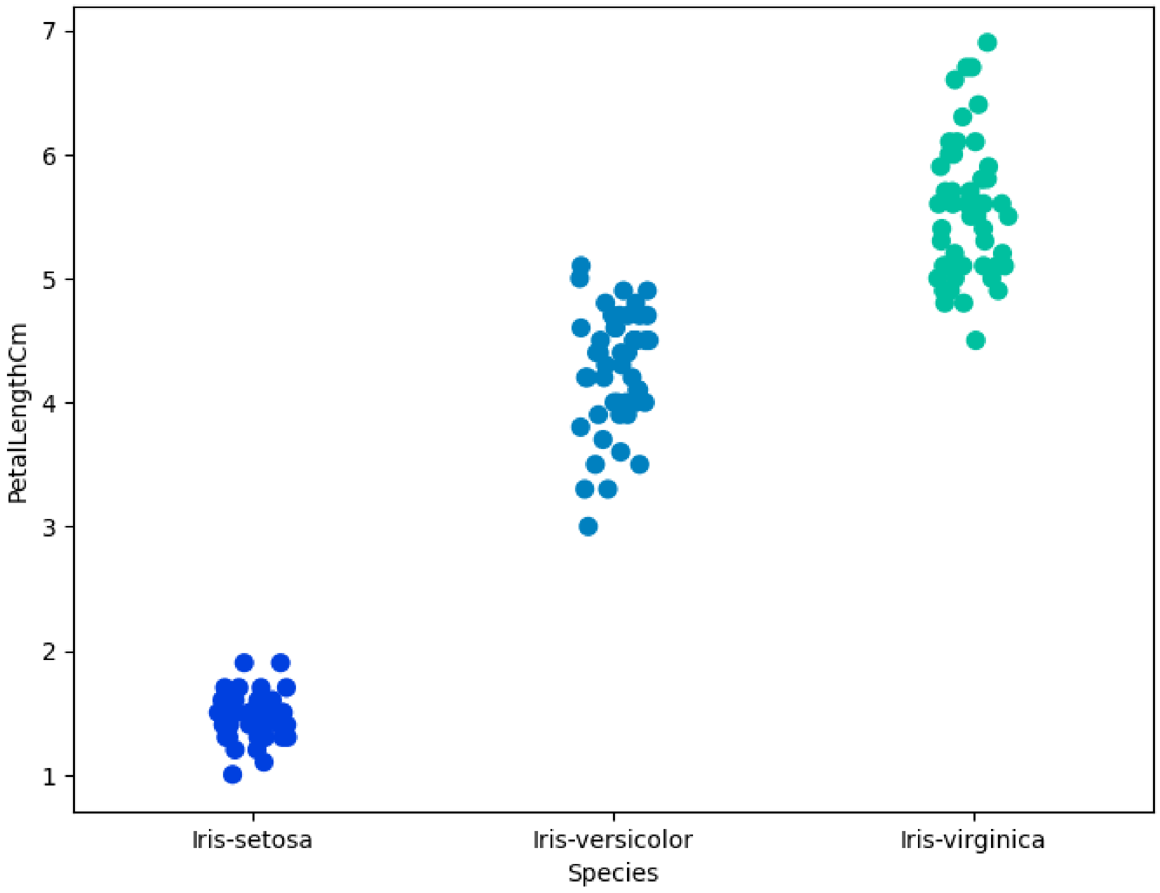


Strip Plot

```
In [24]: fig = plt.gcf()
         fig.set_size_inches(8, 6)
         fig = sns.stripplot(x='Species',y='PetalLengthCm', data = iris, jitter= True)
```

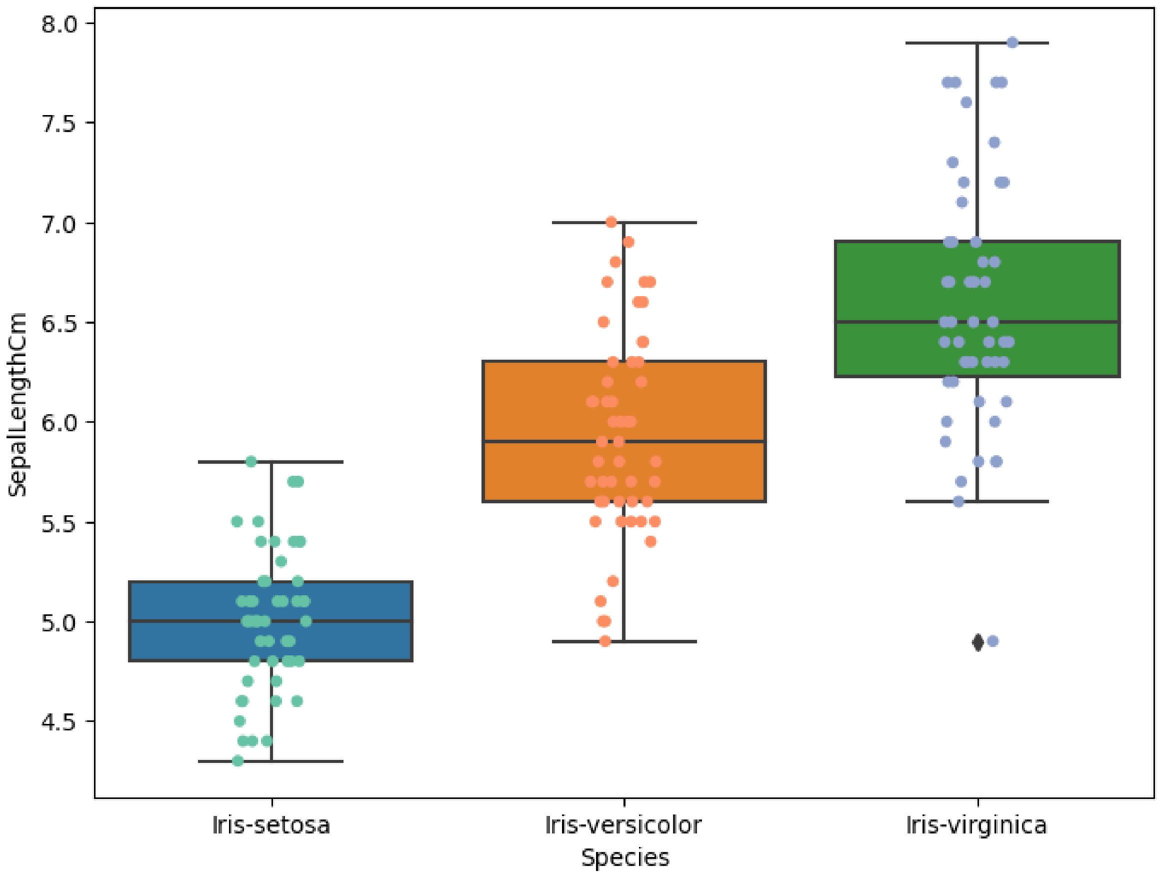


```
In [25]: fig = plt.gcf()
         fig.set_size_inches(8, 6)
         fig = sns.stripplot(x='Species',y='PetalLengthCm', data = iris, jitter= True,
```



combining box and strip plot

```
In [26]: fig = plt.gcf()
         fig.set_size_inches(8, 6)
         fig=sns.boxplot(x='Species',y='SepalLengthCm',data=iris)
         fig=sns.stripplot(x='Species',y='SepalLengthCm',data=iris, jitter = True, edge
```

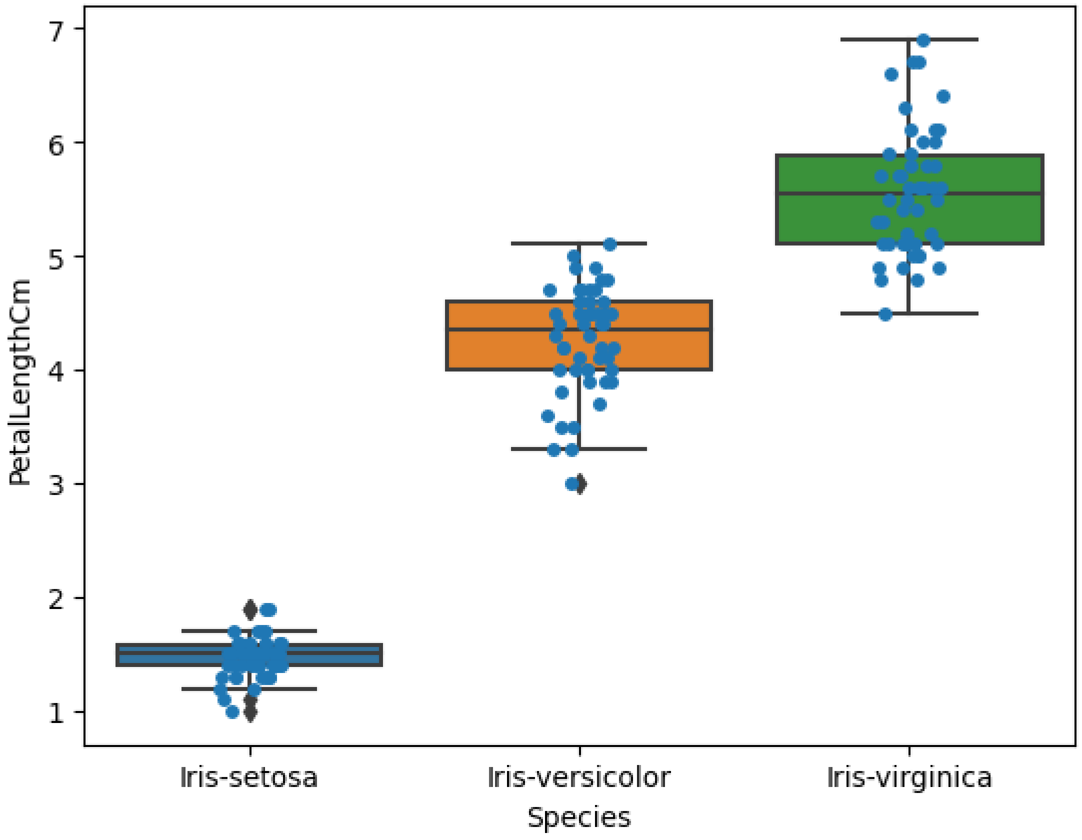


```
In [27]: ax = sns.boxplot(x = 'Species', y = 'PetalLengthCm', data = iris)
         ax = sns.stripplot(x = 'Species', y = 'PetalLengthCm', data = iris, jitter = T
         boxone = ax.artists[0]
         boxone.set_facecolor('purple')
         boxone.set_edgecolor('black')
         boxtwo = ax.artists[1]
         boxtwo.set_facecolor('red')
         boxtwo.set_edgecolor('black')
         boxthree = ax.artists[2]
         boxthree.set_facecolor('yellow')
         boxthree.set_edgecolor('black')
         plt.show()
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[27], line 3
      1 ax = sns.boxplot(x = 'Species', y = 'PetalLengthCm', data = iris)
      2 ax = sns.stripplot(x = 'Species', y = 'PetalLengthCm', data = iris,
jitter = True, edgecolor = 'gray')
----> 3 boxone = ax.artists[0]
      4 boxone.set_facecolor('purple')
      5 boxone.set_edgecolor('black')

File C:\ProgramData\anaconda3\lib\site-packages\matplotlib\axes\_base.py:145
7, in _AxesBase.ArtistList.__getitem__(self, key)
   1456 def __getitem__(self, key):
-> 1457     return [artist
   1458             for artist in self._axes._children
   1459             if self._type_check(artist)][key]

IndexError: list index out of range
```



Violin Plot

```
In [ ]: fig = plt.gcf()
        fig.set_size_inches(8, 6)
        fig = sns.violinplot(x='Species',y='SepalLengthCm', data = iris)
```

```
In [ ]: plt.figure(figsize =(8, 6))
        plt.subplot(2, 2, 1)
        sns.violinplot(x='Species',y='PetalLengthCm', data = iris)
        plt.subplot(2, 2, 2)
        sns.violinplot(x='Species',y='PetalWidthCm', data = iris)
        plt.subplot(2, 2, 3)
        sns.violinplot(x='Species',y='SepalLengthCm', data = iris)
        plt.subplot(2, 2, 4)
        sns.violinplot(x='Species',y='SepalWidthCm', data = iris)
```

Pair Plot

A “pairs plot” is also known as a scatterplot, in which one variable in the same data row is matched with another variable's value, like this: Pairs plots are just elaborations on this, showing all variables paired with all the other variables.

```
In [ ]: sns.pairplot(data=iris, kind='scatter')
```

```
In [ ]: sns.pairplot(data = iris, hue='Species')
```

Heat Map

Heat map is used to find out the correlation between different features in the dataset.High positive or negative value shows that the features have high correlation.This helps us to select the parmeters for machine learning.

```
In [ ]: fig = plt.gcf()
        fig.set_size_inches(8, 6)
        fig = sns.heatmap(data = iris.corr(), annot=True, cmap = 'cubehelix', linewidt
```

Distribution Plot

The distribution plot is suitable for comparing range and distribution for groups of numerical data. Data is plotted as value points along an axis. You can choose to display only the value points to see the distribution of values, a bounding box to see the range of values, or a combination of both as shown here.The distribution plot is not relevant for detailed analysis of the data as it deals with a summary of the data distribution.

In [ ]:
```
iris.hist(edgecolor='black', linewidth = 1.2)
fig = plt.gcf()
fig.set_size_inches(12, 6)
```

Swarm Plot

It looks a bit like a friendly swarm of bees buzzing about their hive. More importantly, each data point is clearly visible and no data are obscured by overplotting.A beeswarm plot improves upon the random jittering approach to move data points the minimum distance away from one another to avoid overlays. The result is a plot where you can see each distinct data point, like shown in below plot

In [ ]:
```
sns.set(style='darkgrid')
fig = plt.gcf()
fig.set_size_inches(10, 7)
fig = sns.swarmplot (x="Species", y="PetalLengthCm", data = iris)
```

In [ ]:
```
sns.set(style='darkgrid')
fig = plt.gcf()
fig.set_size_inches(10, 7)
ax = sns.violinplot(x="Species", y="PetalLengthCm", data = iris, inner = None)
ax = sns.swarmplot (x="Species", y="PetalLengthCm", data = iris, color='white'
```

Lm PLot

In [ ]:
```
fig = sns.lmplot(x="PetalLengthCm", y="PetalWidthCm", data = iris)
```

FacetGrid

In [ ]:
```
sns.FacetGrid(iris, hue='Species', height = 6)\
.map(sns.kdeplot, "PetalLengthCm")\
.add_legend()
plt.ioff()
plt.show()
```

catplot

In [ ]:
```
# Create a factor plot using sns.catplot
sns.catplot(x='Species', y='SepalLengthCm', data=iris,kind="point")
plt.ioff()
plt.show()
```

Boxen Plot

In [ ]:
```
fig = plt.gcf()
fig.set_size_inches(10, 7)
fig = sns.boxenplot(x='Species', y='SepalLengthCm', data=iris)
plt.show()
```

In [ ]:
```
# Create a kde plot of sepal_length versus sepal width for setosa species of f
sub = iris[iris['Species'] == 'Iris-setosa']
sns.kdeplot(data = sub, x = 'SepalLengthCm', y = 'SepalWidthCm', cmap = 'plasm
plt.title('Iris-setosa')
plt.xlabel('Sepal Length Cm')
plt.ylabel('Sepal Width Cm')
plt.show()
```

Dashboard

In [ ]:
```
sns.set_style('darkgrid')
f, axes = plt.subplots(2, 2, figsize = (15, 15))

k1 = sns.boxplot(x="Species", y="PetalLengthCm", data=iris, ax = axes[0, 0])
k2 = sns.violinplot(x="Species", y="PetalLengthCm", data = iris, ax = axes [0,
k3 = sns.stripplot(x="Species", y="PetalLengthCm", data = iris, ax = axes[1, 0

axes[1, 1].hist(iris.PetalLengthCm, bins= 50)
plt.show()
```

In the dashboard we have shown how to create multiple plots to foam a dashboard using Python.In this plot we have demonstrated how to plot Seaborn and Matplotlib plots on the same Dashboard.

Stacked Histogram

In [ ]:
```
iris['Species'] = iris['Species'].astype('category')
```

In [ ]:
```
iris.Species.cat.categories
```

In [ ]:
```
iris.Species.unique()
```

In [ ]:
```
iris[iris.Species=='Iris-setosa'].SepalLengthCm
```

In [ ]:
```
iris[iris.Species=='Iris-setosa']
```

In [ ]:
```
list1 = list()
mylabels = list()
for gen in iris.Species.cat.categories:
    list1.append(iris[iris.Species == gen].SepalLengthCm)
    mylabels.append(gen)

h = plt.hist(list1, bins=30, stacked = True, rwidth = 1, label = mylabels)
plt.legend()
plt.show()
```

With Stacked Histogram we can see the distribution of Sepal Length of Different Species together.This shows us the range of Sepan Length for the three different Species of Iris Flower.

Area Plot

Area Plot gives us a visual representation of Various dimensions of Iris flower and their range in dataset.

In [ ]:
```
t=iris.plot.area(y=['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidth
plt.show()
```

Distplot

In [ ]:
```
sns.distplot(iris['SepalLengthCm'], kde=True, bins = 20);
plt.show()
```

In [ ]:

In [ ]:

In [ ]: