In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

In [2]:
```
iris=pd.read_csv(r"D:\Full Stack Data Science\17 Aug\17th\IRIS DATASET _ A
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.shape
```

Out[3]: (150, 6)

In [4]:
```
1 iris.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
```

In [5]:
```
1 iris.columns
```

Out[5]:
```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthC
m',
       'Species'],
      dtype='object')
```

In [6]:
```
1 iris.isnull().sum()
```

Out[6]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [7]:
```
1 iris.drop(columns='Id',axis=1,inplace=True)
```

In [8]:
```
1 iris.head()
```

Out[8]:

	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 [9]:
```
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
```

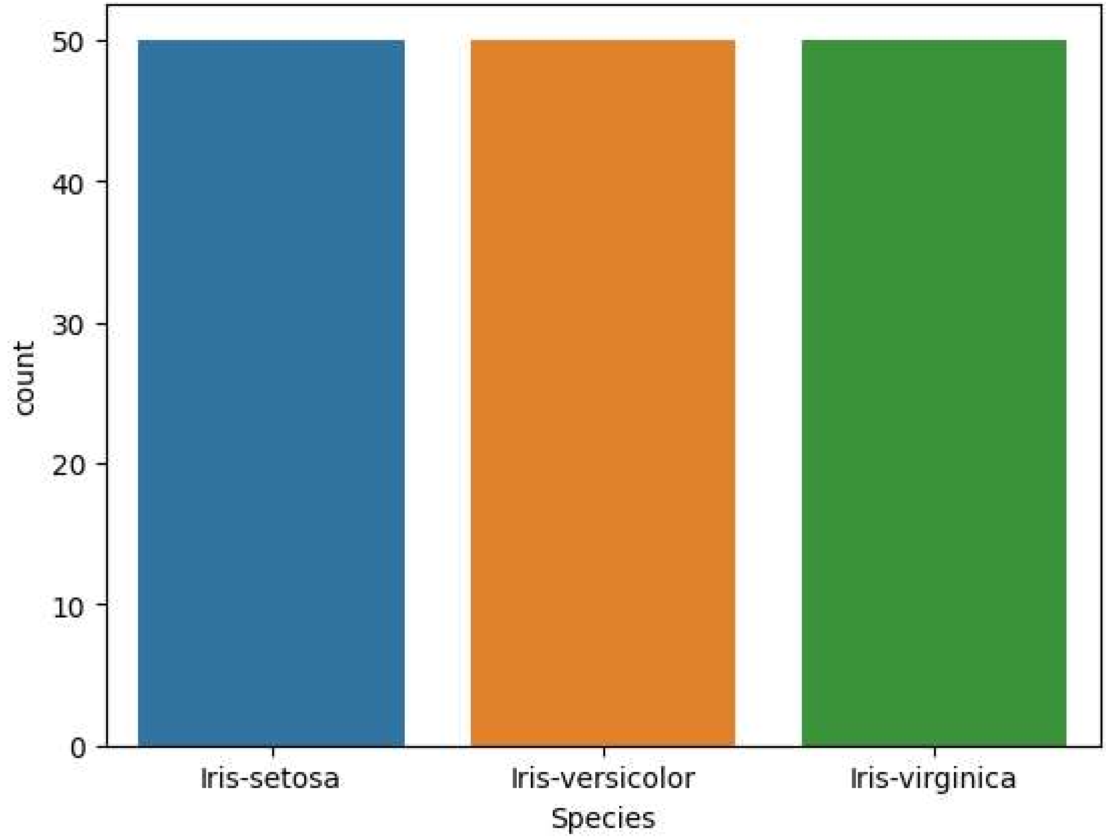
In [10]:
```
iris.Species.value_counts()
```

Out[10]:
```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: count, dtype: int64
```

1) Bar Plot

- Bar plots are use in comparing different categories or groups in a dataset.

In [11]:
```
sns.countplot(data=iris,x='Species')
plt.show()
```



We have 3 Species and the count of the Species are equal

In [12]:
```
iris.head(2)
```

Out[12]:

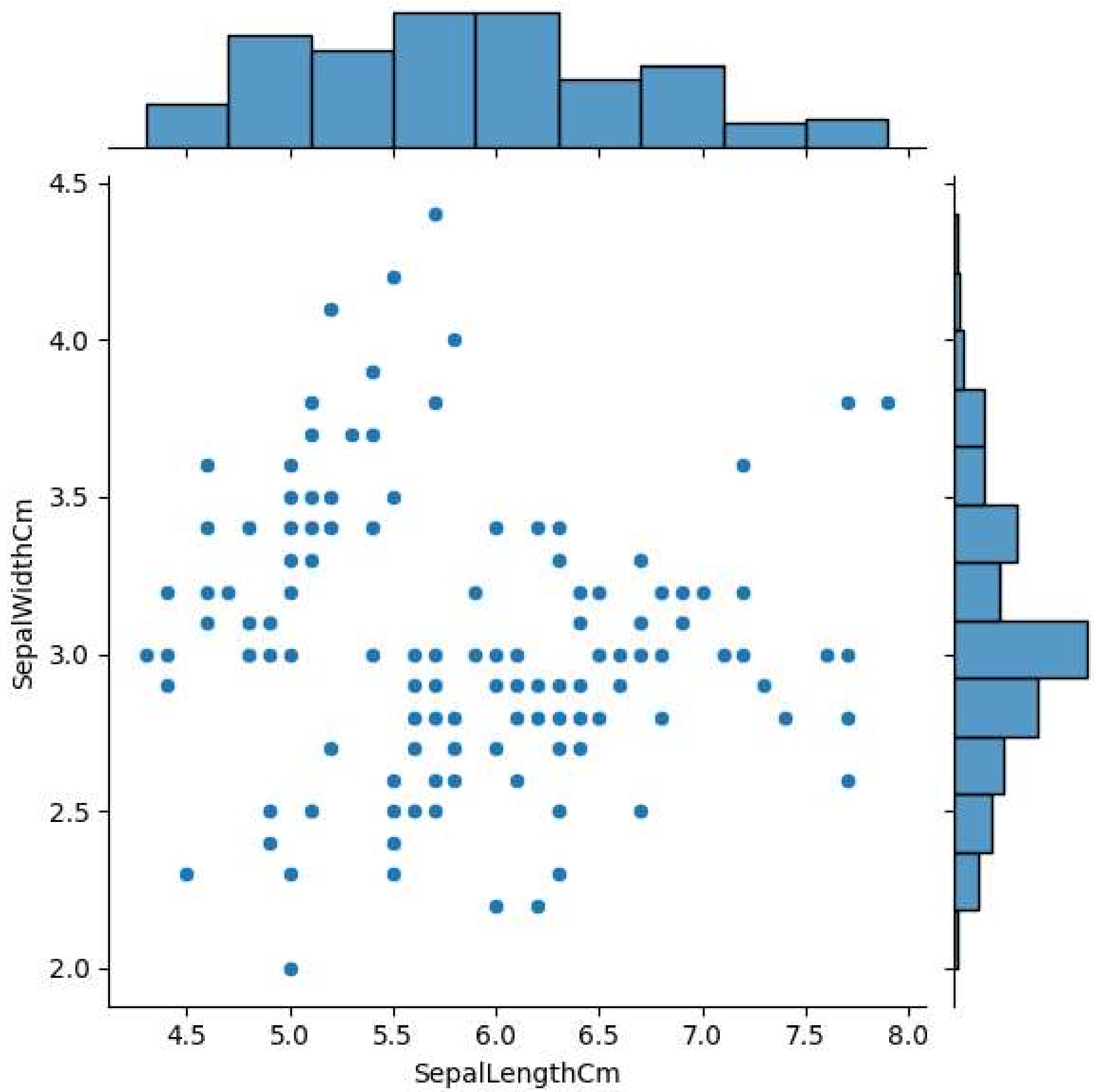
	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa

2) Joint Plot

- It is useful to understand the relationship between two variables,including their correlation and distribution patterns.

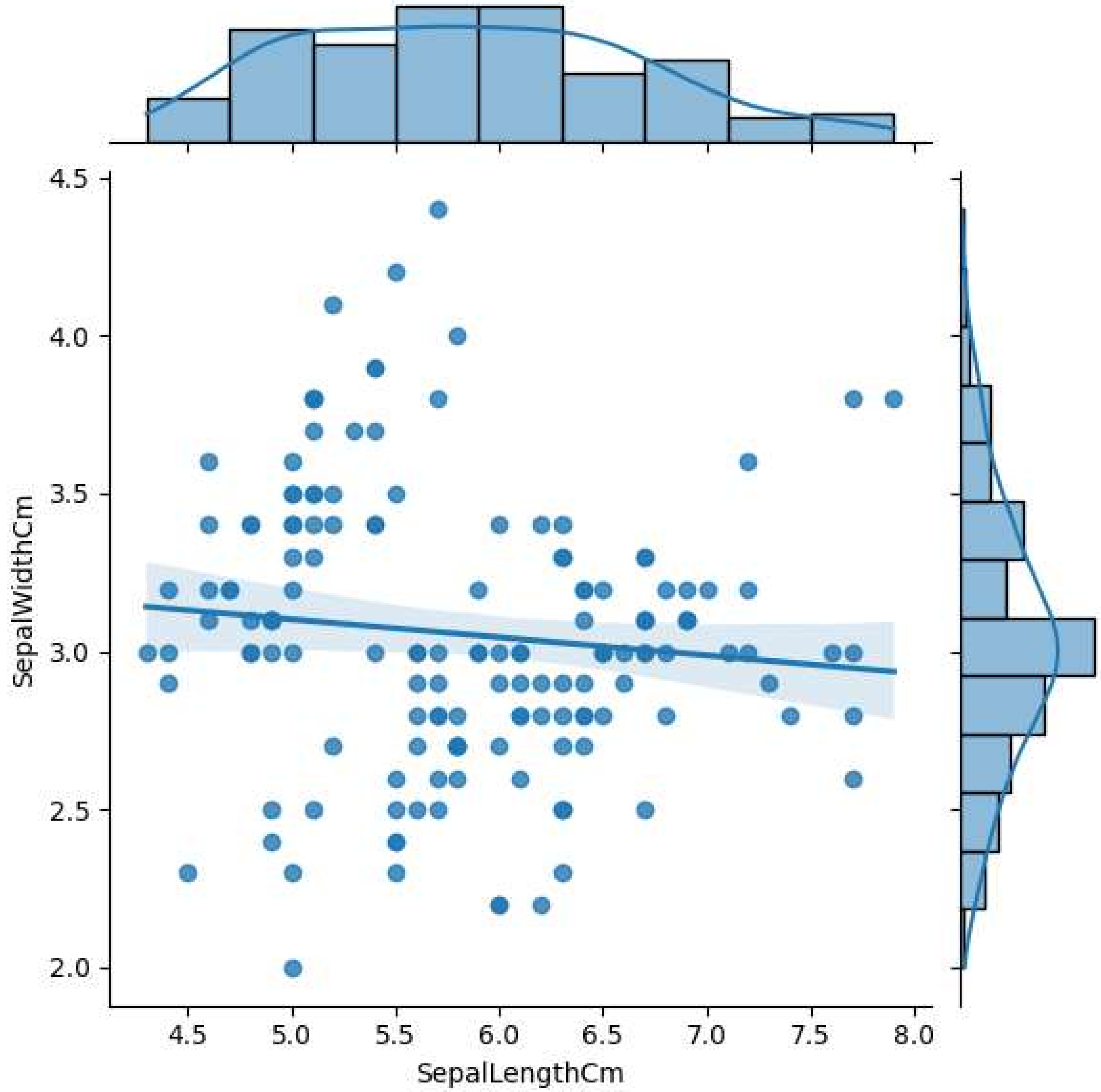
In [13]:
```
sns.jointplot(data=iris,x='SepalLengthCm',y='SepalWidthCm')
```

Out[13]: <seaborn.axisgrid.JointGrid at 0x1ecea473b10>



In [14]:

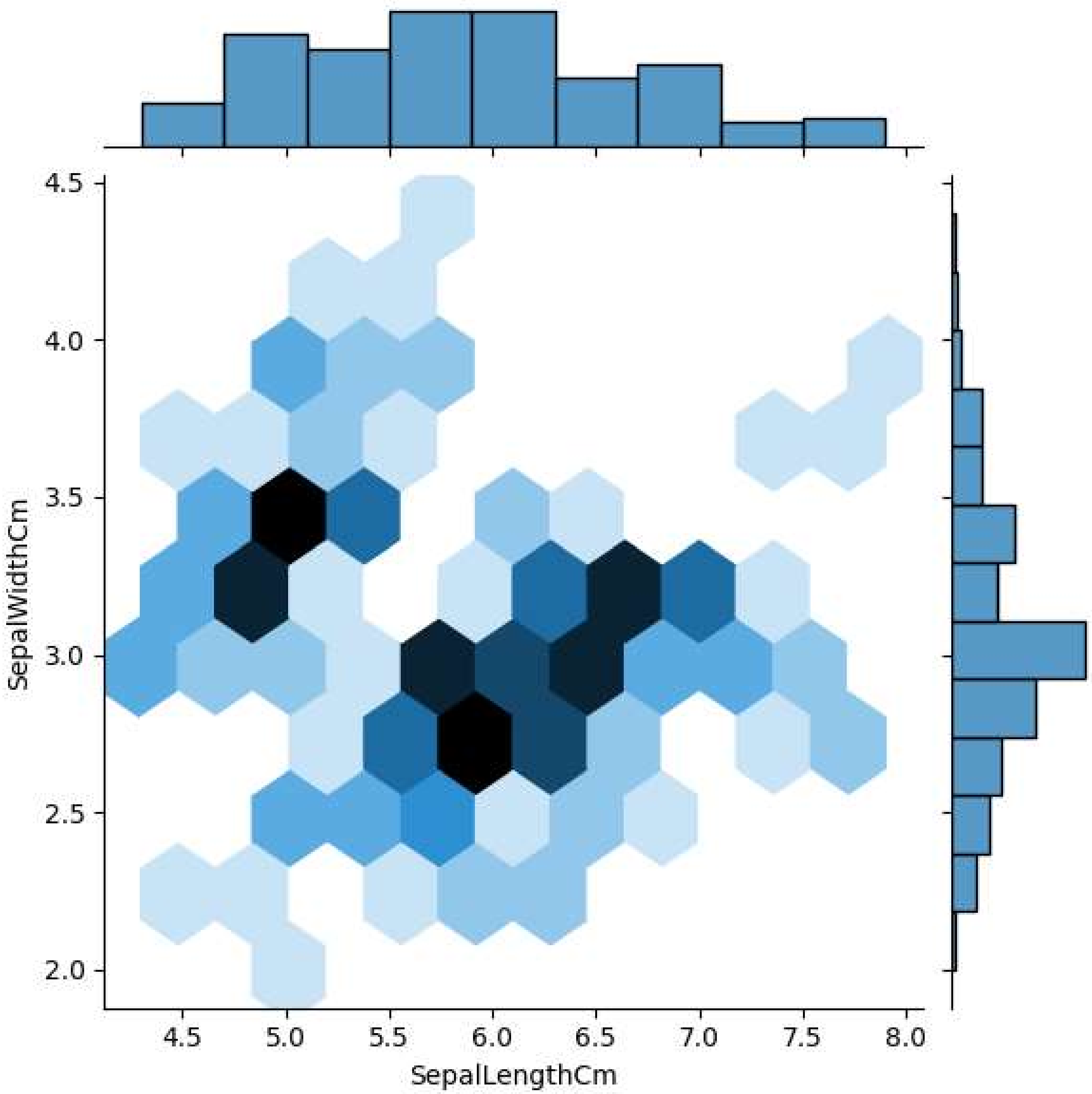
```
sns.jointplot(data=iris,x='SepalLengthCm',y='SepalWidthCm',kind='reg');
```



- Sepal length points are Uniformly distributed.
- Sepal Width points are Normally distributed.
- Correlation between Sepal length & Sepal width is negative correlation (-1 to 0).

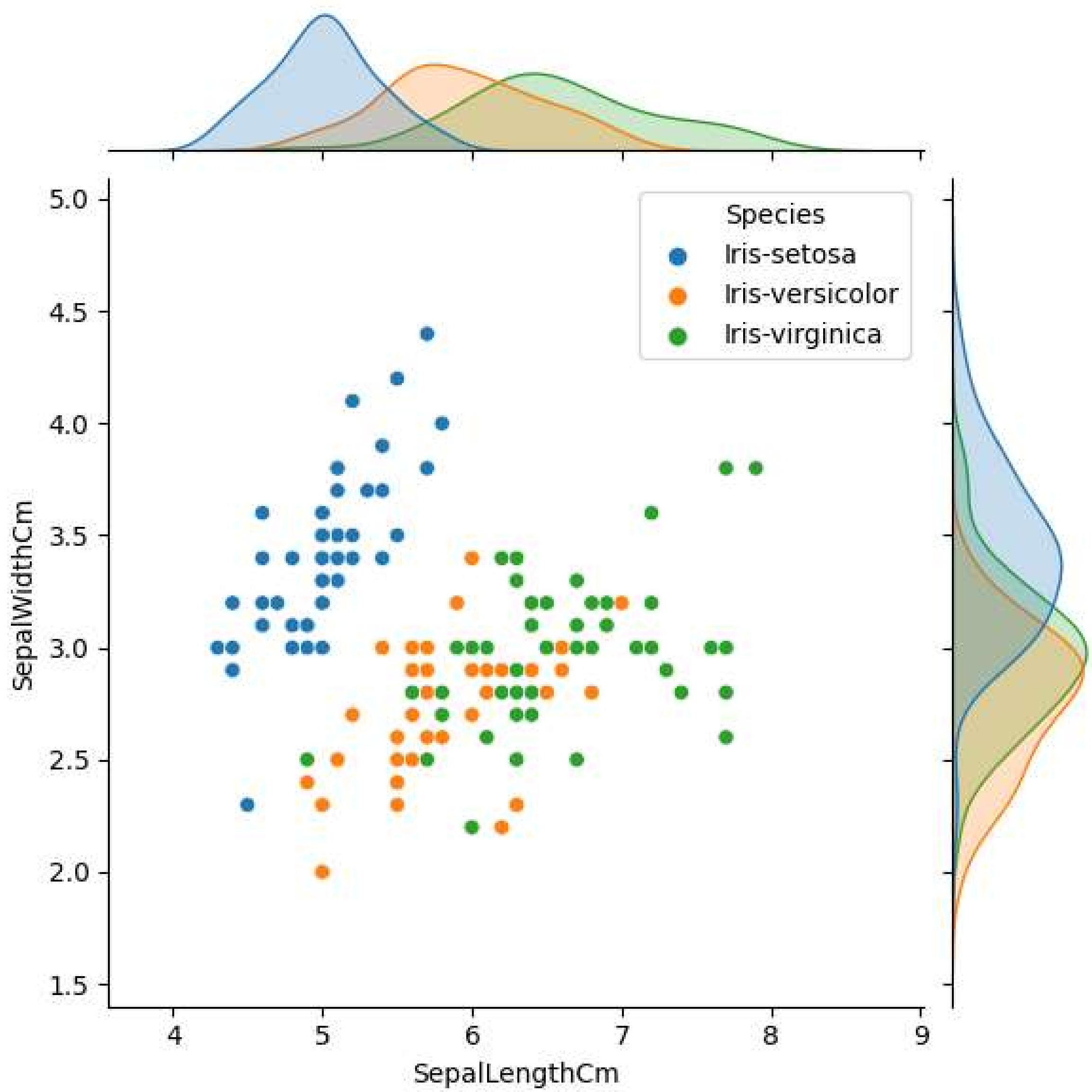
In [15]:

```
1 sns.jointplot(data=iris,x='SepalLengthCm',y='SepalWidthCm',kind='hex');
```



In [16]:

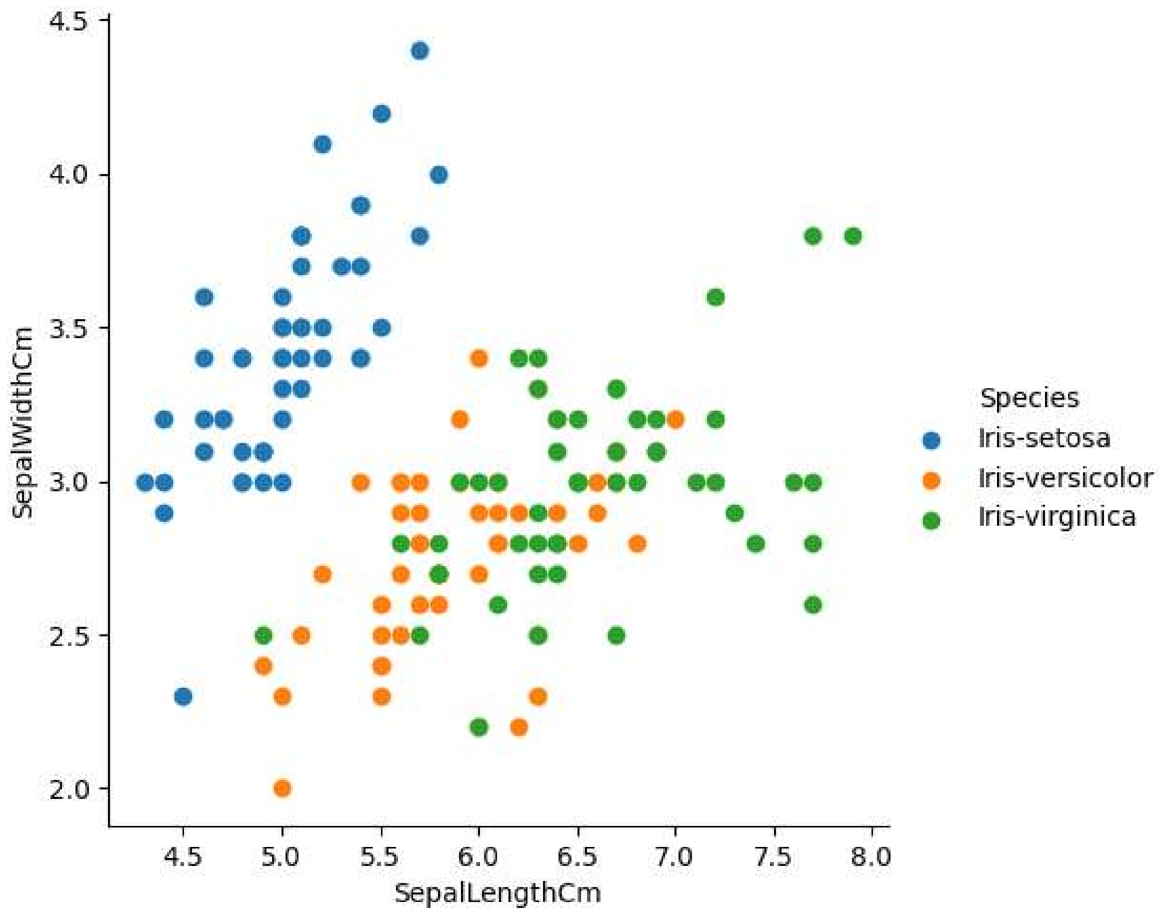
```
sns.jointplot(data=iris,x='SepalLengthCm',y='SepalWidthCm',hue='Species')
```



3) FacetGrid Plot

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

Out[17]: <seaborn.axisgrid.FacetGrid at 0x1eced344690>

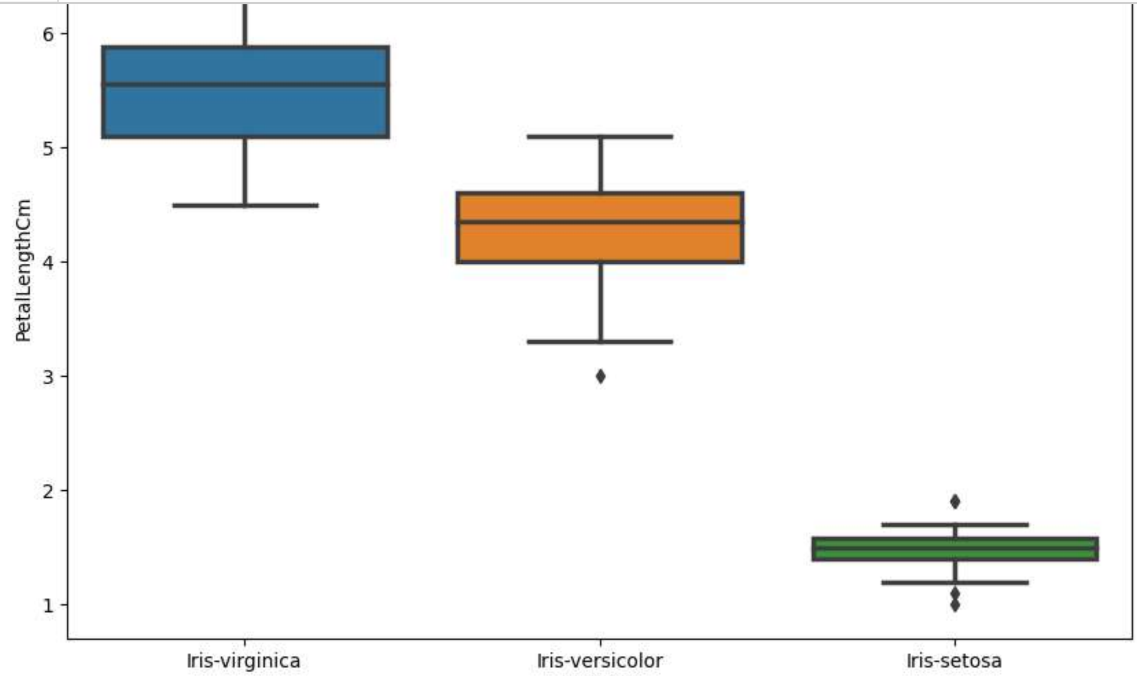


4) Box Plot and Whisker Plot

- It is a graphical representation of the distribution of a dataset.
- It gives summary statistics such as median,quartiles and outliers.

In [18]:

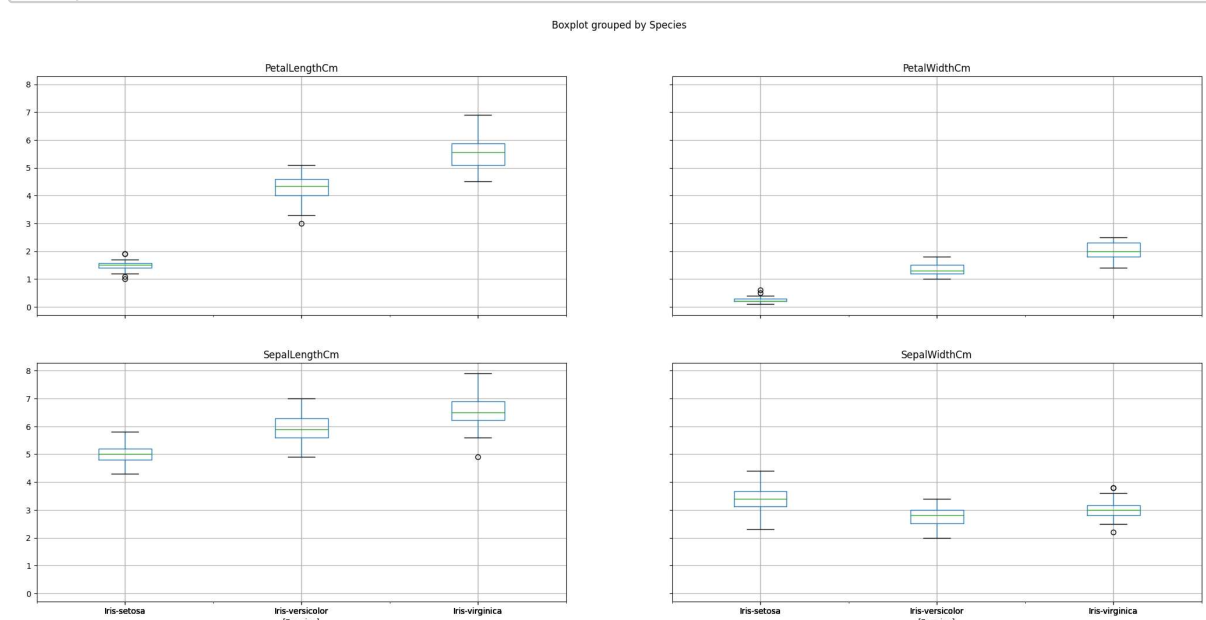
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(data=iris,x='Species',y='PetalLengthCm',order=['Iris-virg
# data-Dataset
# order-we change the order of species
# linewidth- border width of boxplot
# orient- vertical orient
# Dodge-
```



- Petal Length of Species 'Iris-virginica' is higher than other species.
- Iris-setosa has the shortest petal length.

In [19]:

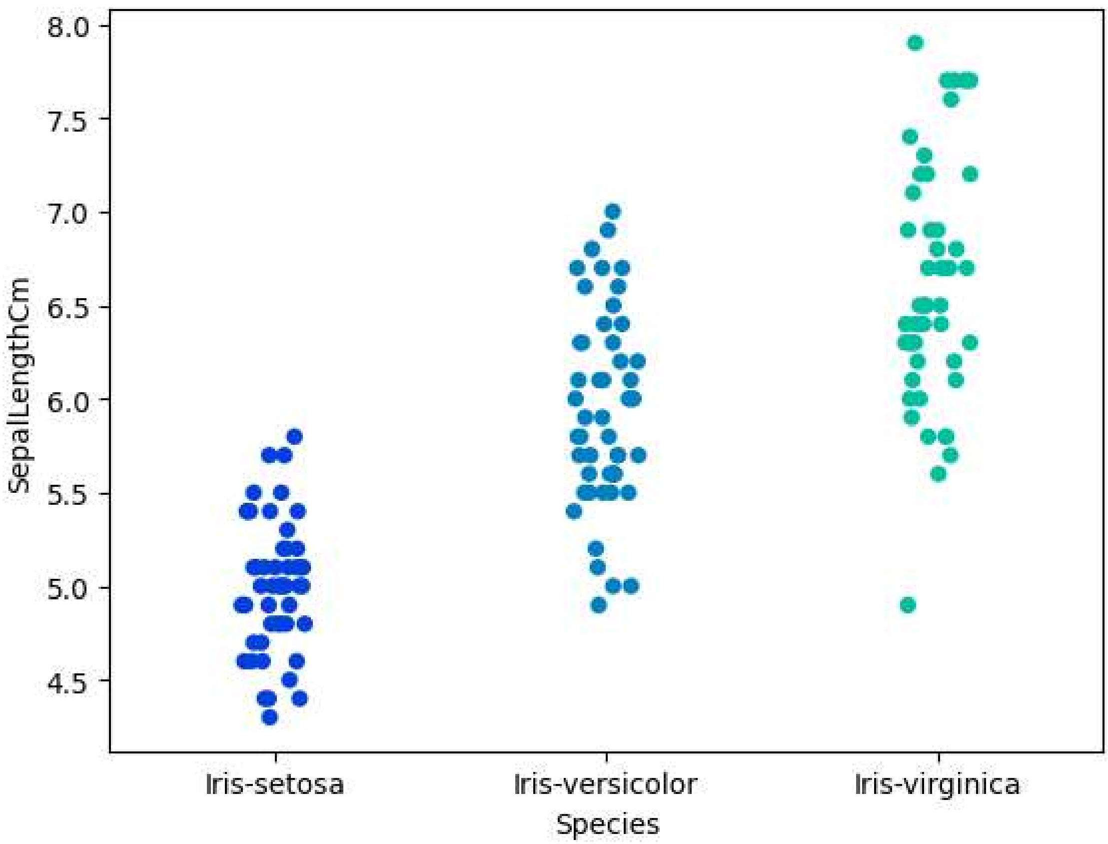
```
iris.boxplot(by='Species',figsize=(25,12));
```



5) Strip Plot

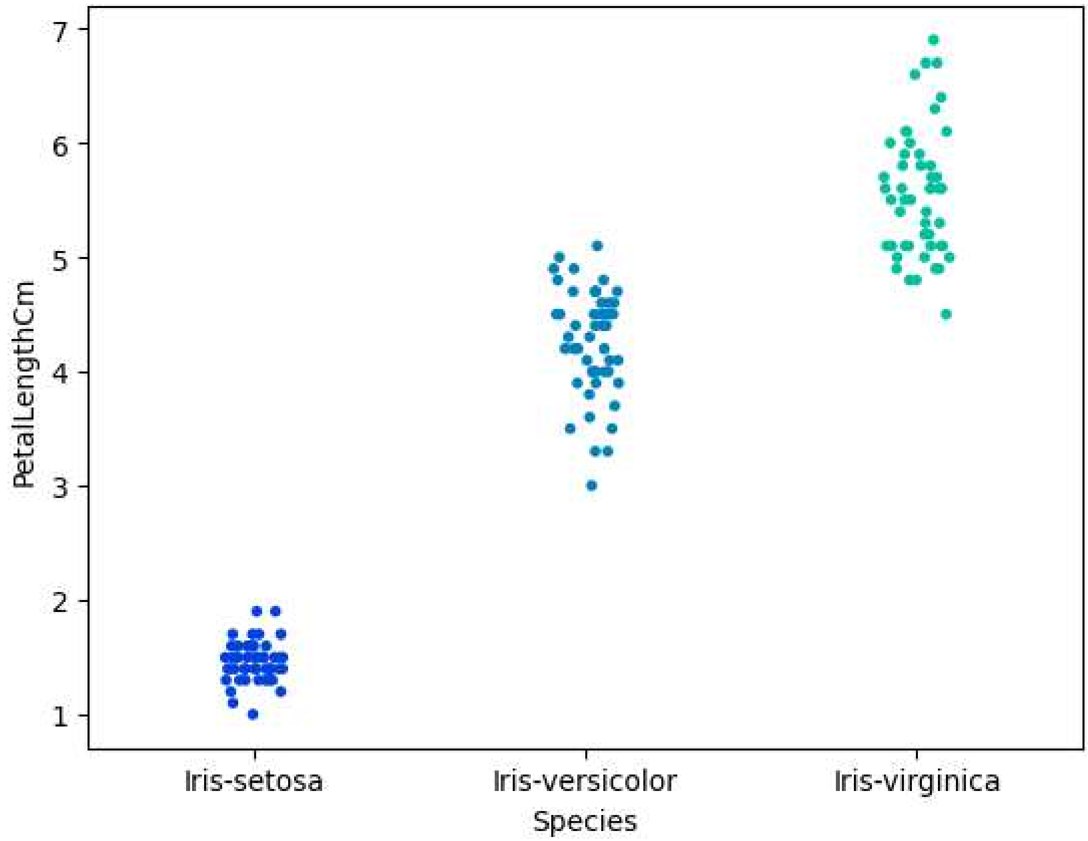
- Strip plot displays individual data points along an axis.
- Its useful for visualizing the distribution of data points within a single categorical variable.

In [20]:
```
fig=plt.gcf()
fig=sns.stripplot(data=iris,x='Species',y='SepalLengthCm',jitter=True,size
```



In [21]:

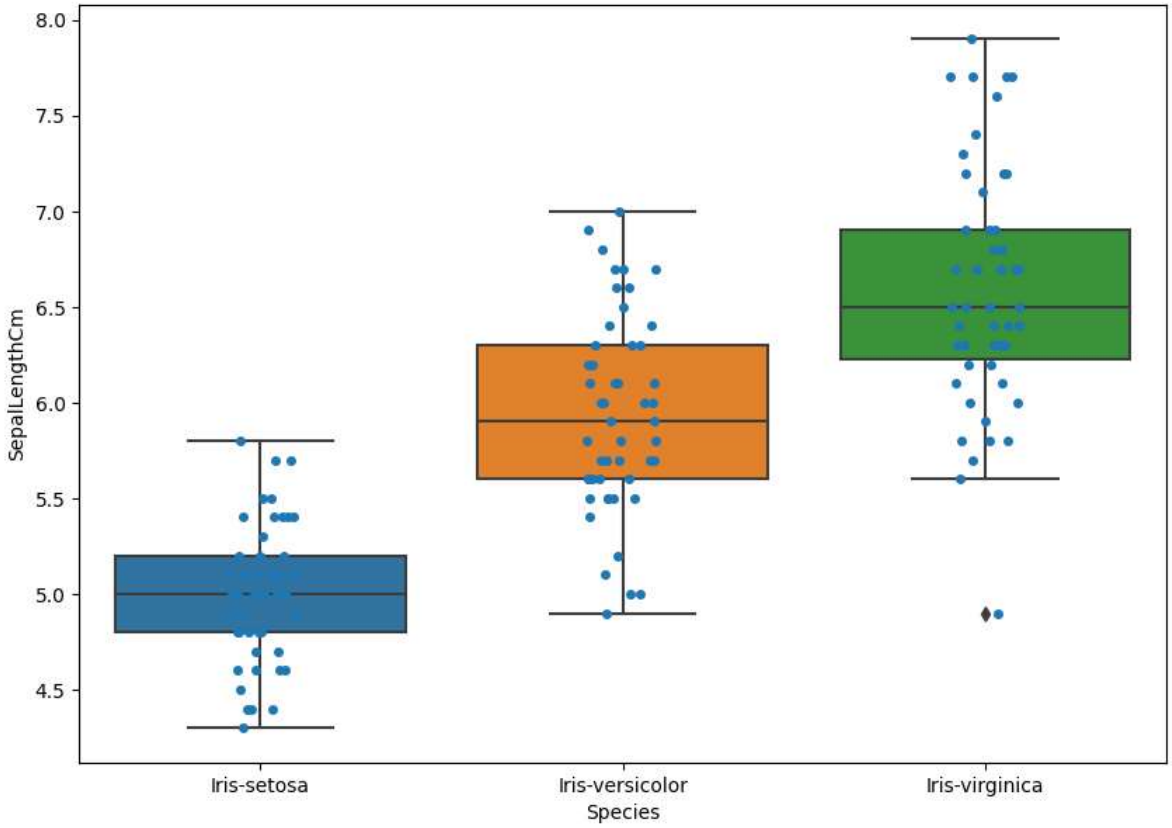
```
fig=sns.stripplot(data=iris,x='Species',y='PetalLengthCm',jitter=True,siz
```



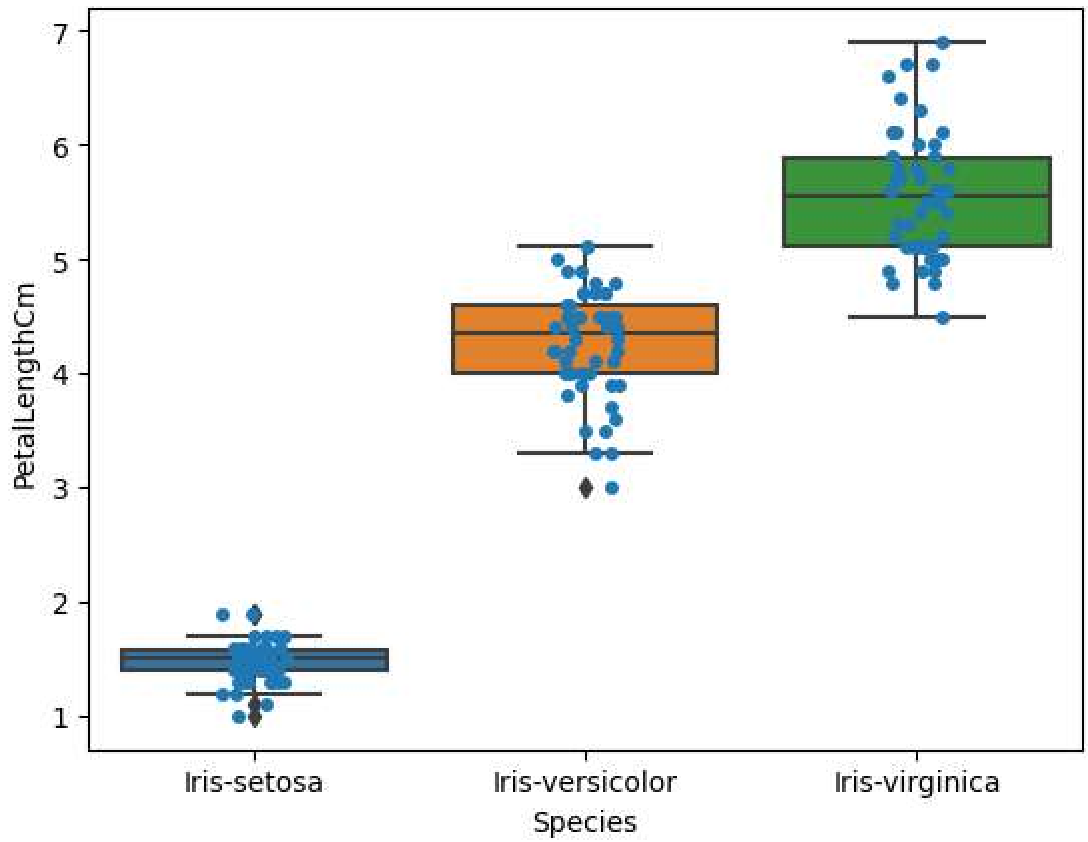
6) Combining Box Plot and Strip Plot

In [26]:

```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(data=iris,x='Species',y='SepalLengthCm')
fig=sns.stripplot(data=iris,x='Species' ,y='SepalLengthCm')
```



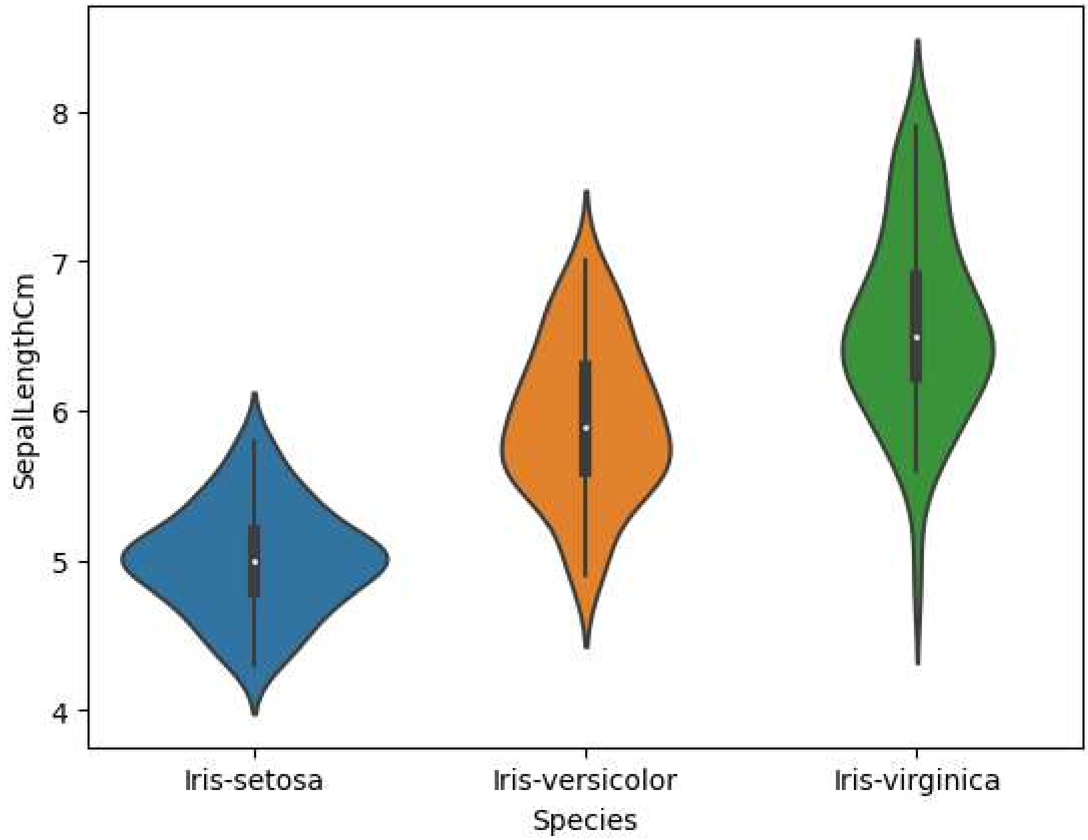
In [27]:
```
ax=sns.boxplot(x='Species',y='PetalLengthCm',data=iris)
ax=sns.stripplot(x='Species' ,y='PetalLengthCm',data=iris,jitter=True, ed
```



7) Violin Plot

- It is used to visualize the distribution of data and its probability distribution
- This plot is combination of Box plot & Density Plot.
- The thick black bar in the center represent the interquartile range
- Inter Quartile Range = Q3-Q1
- Where,Q3 is Third quartile(75%) & Q1 is First quartile(25%)
- The thin black line represent the 95% confidence interval.
- White dot in the middle represent the Median.

In [28]:
```
fig=sns.violinplot(data=iris,x='Species',y='SepalLengthCm')
```



Combination of 4 Plots

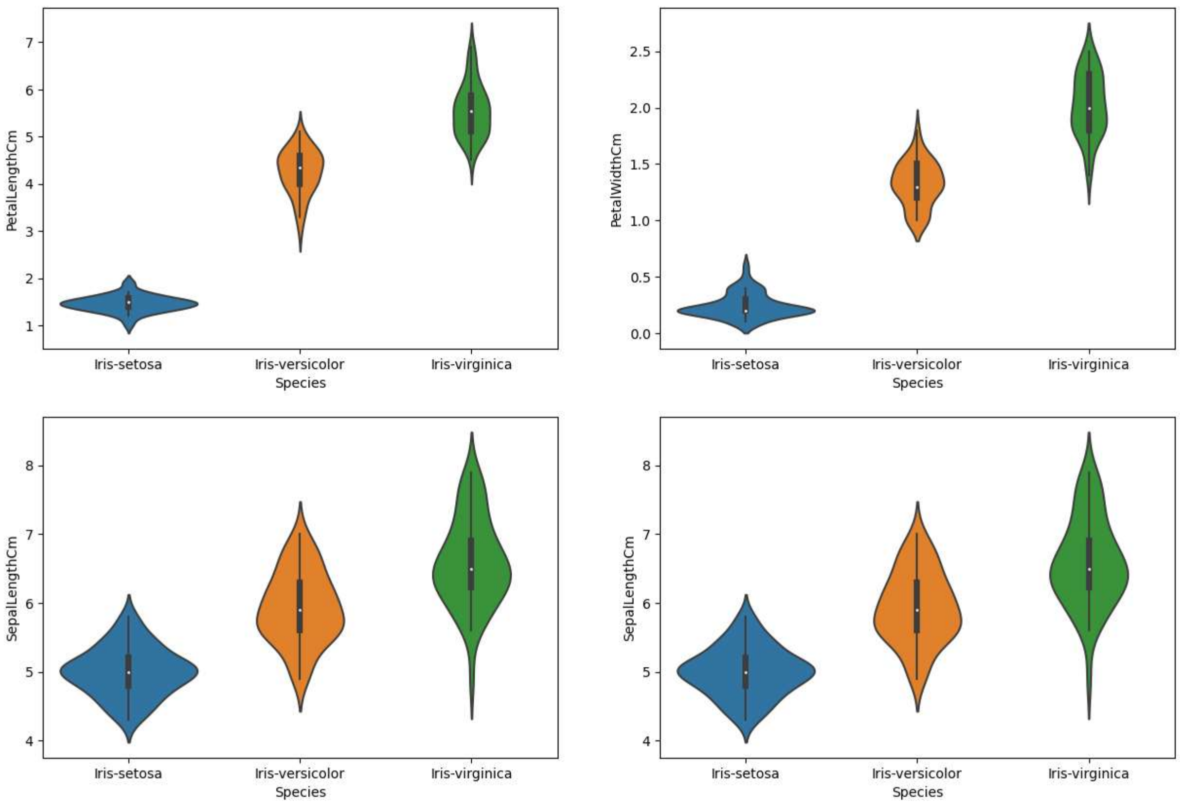
In [31]:
```python
plt.figure(figsize=(15,10))
plt.subplots
plt.subplot(2,2,1) # nrows=2, ncol=2, index=1
sns.violinplot(data=iris,x='Species',y='PetalLengthCm')

plt.subplot(2,2,2)
sns.violinplot(data=iris,x='Species',y='PetalWidthCm')

plt.subplot(2,2,3)
sns.violinplot(data=iris,x='Species',y='SepalLengthCm')

plt.subplot(2,2,4)
sns.violinplot(data=iris,x='Species',y='SepalLengthCm')
```

Out[31]: <Axes: xlabel='Species', ylabel='SepalLengthCm'>

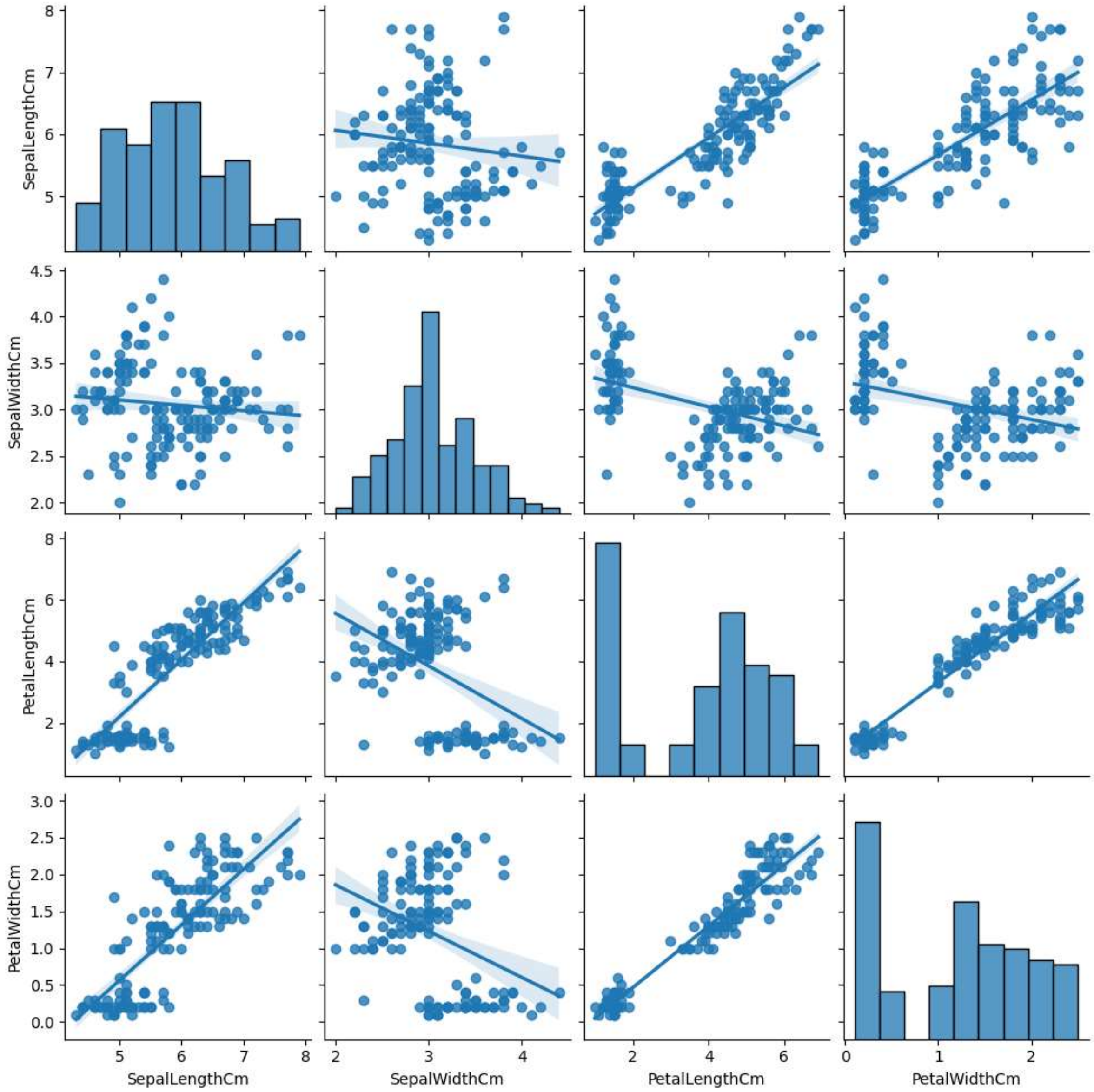


8) Pair Plot / Scatter Plot

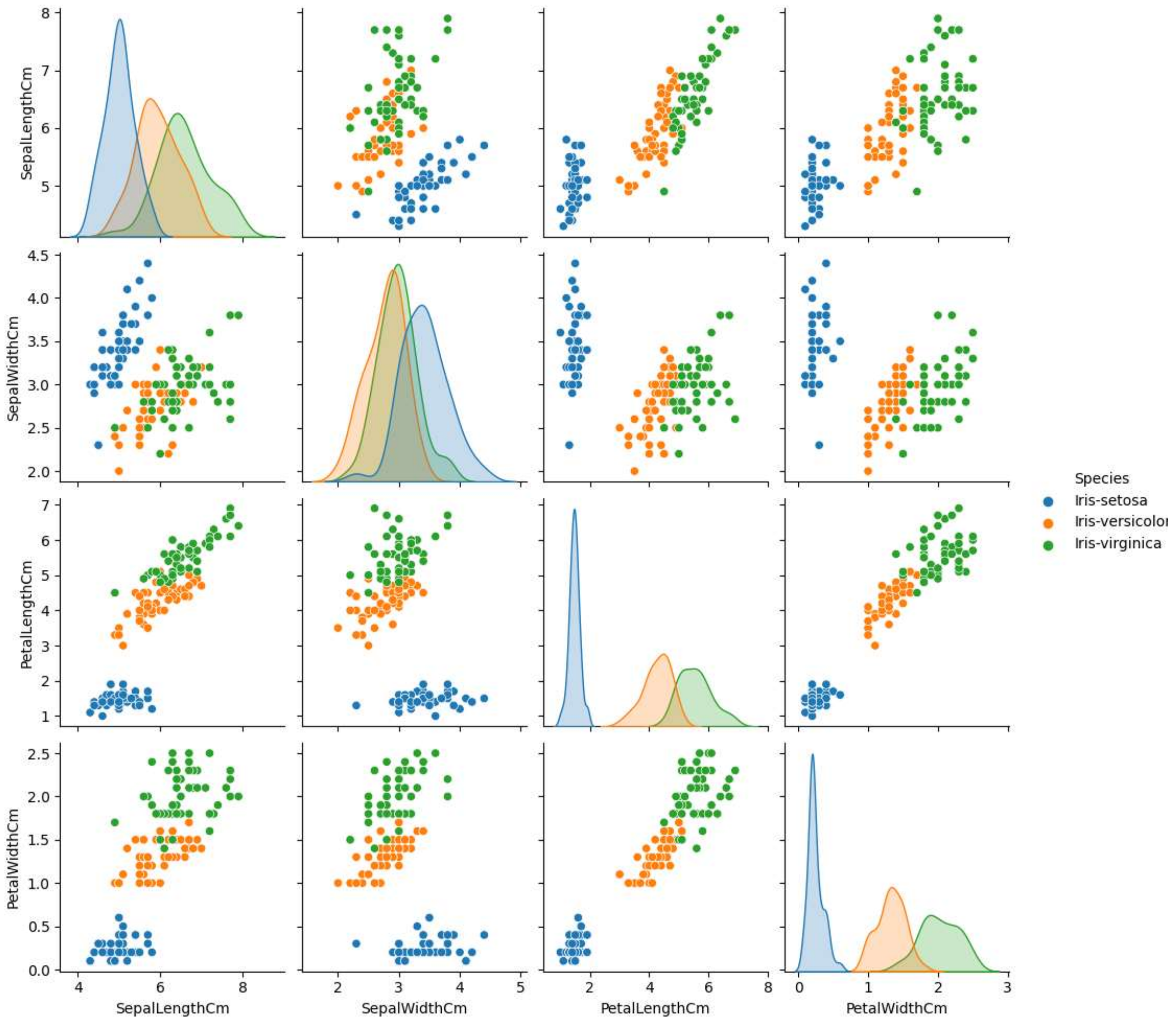
- It visualize pairwise relationship between multiple variables in a dataset.
- It displays scatter plots for each pair of variables,histogram for individual variables and correlation coefficients if required.

In [32]:

```
1 sns.pairplot(data=iris,kind='reg');
```



In [33]:
```
sns.pairplot(data=iris,hue='Species');
```



9) Heat Map

- It is used to find correlation between variables and show how variables are related to each other.

In [34]:

```
iris1=iris[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidth(
iris1
```

Out[34]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
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

In [35]:

```
iris1.corr()
```

Out[35]:

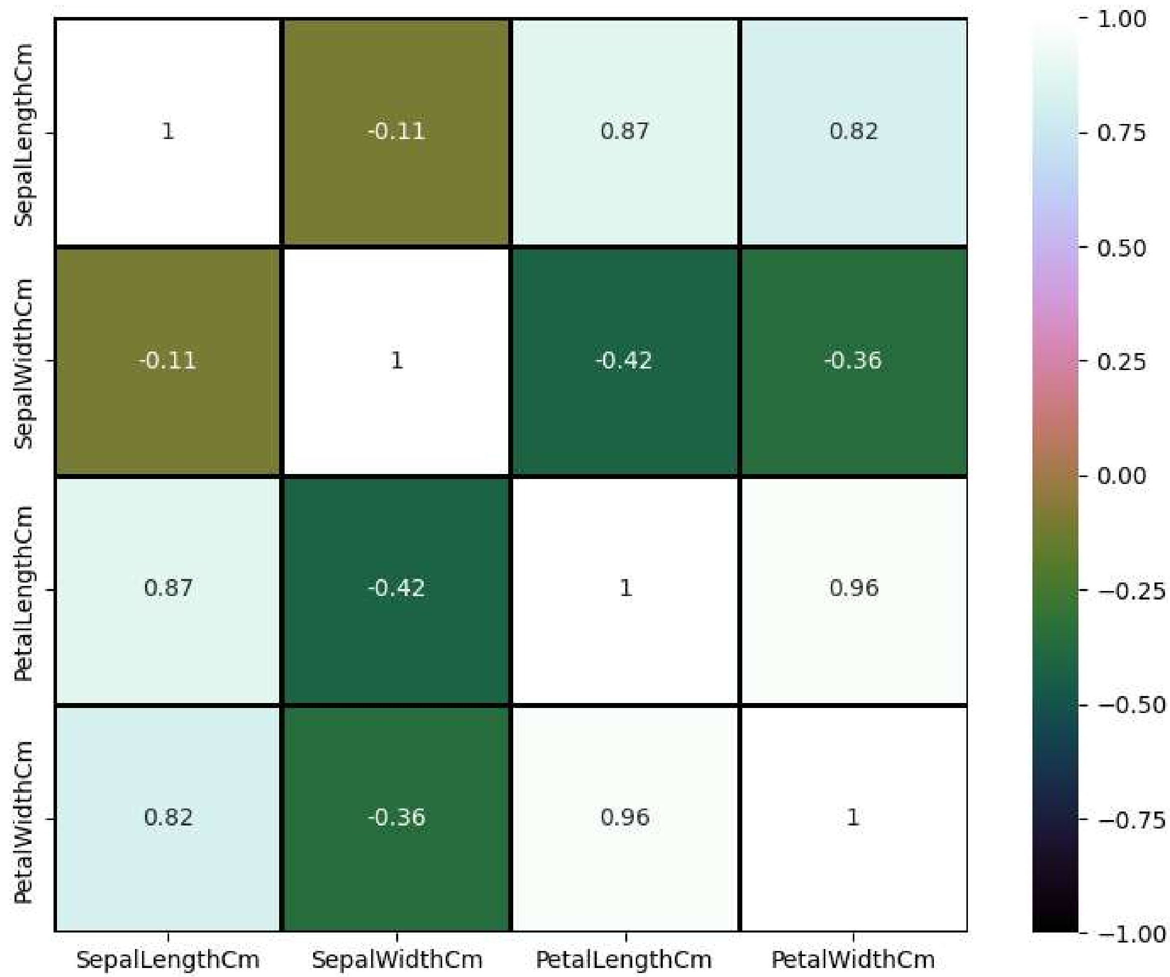
	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
SepalLengthCm	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.817954	-0.356544	0.962757	1.000000

In [36]:

```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.heatmap(data=iris1.corr(),annot=True,cmap='cubehelix',linewidth=1

# annot- Gives the correlation values on graph
# linewidth- Width of the lines that will divide each cell.
# linecolor- color of the lines that will divide each cell.
# square- Border of square

```



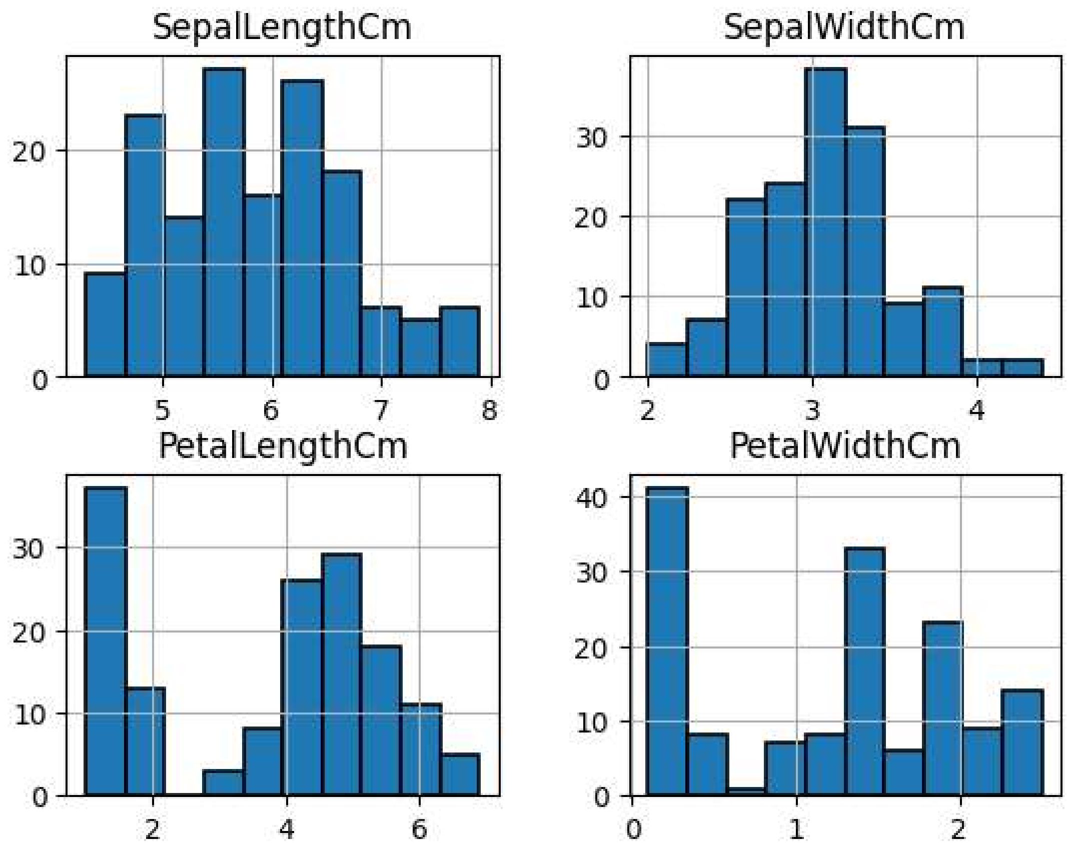
10) Distribution Plot

- A distribution plot also known as a probability density plot or density plot
- It provide an estimate of the probability density function of a continuous variable.

In [37]:

```
fig=plt.gcf()
fig.set_size_inches(12,6)
iris.hist(edgecolor='black',linewidth=1.2);
```

```
<Figure size 1200x600 with 0 Axes>
```

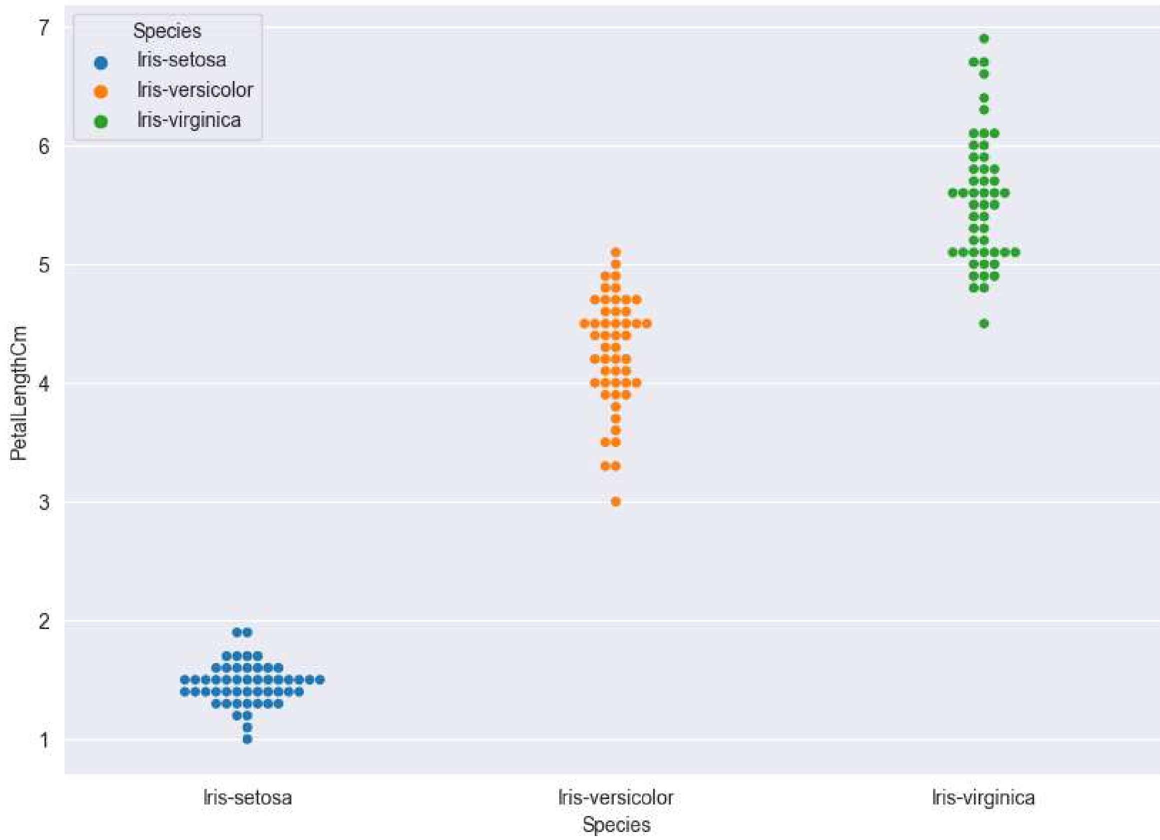


11) Swarm Plot

- A swarm plot is a categorical scatter plot that displays individual data points along a single axis based on categorical variable.
- Each data point is plotted with a slight displacement along the categorical axis to avoid overlap

In [38]:

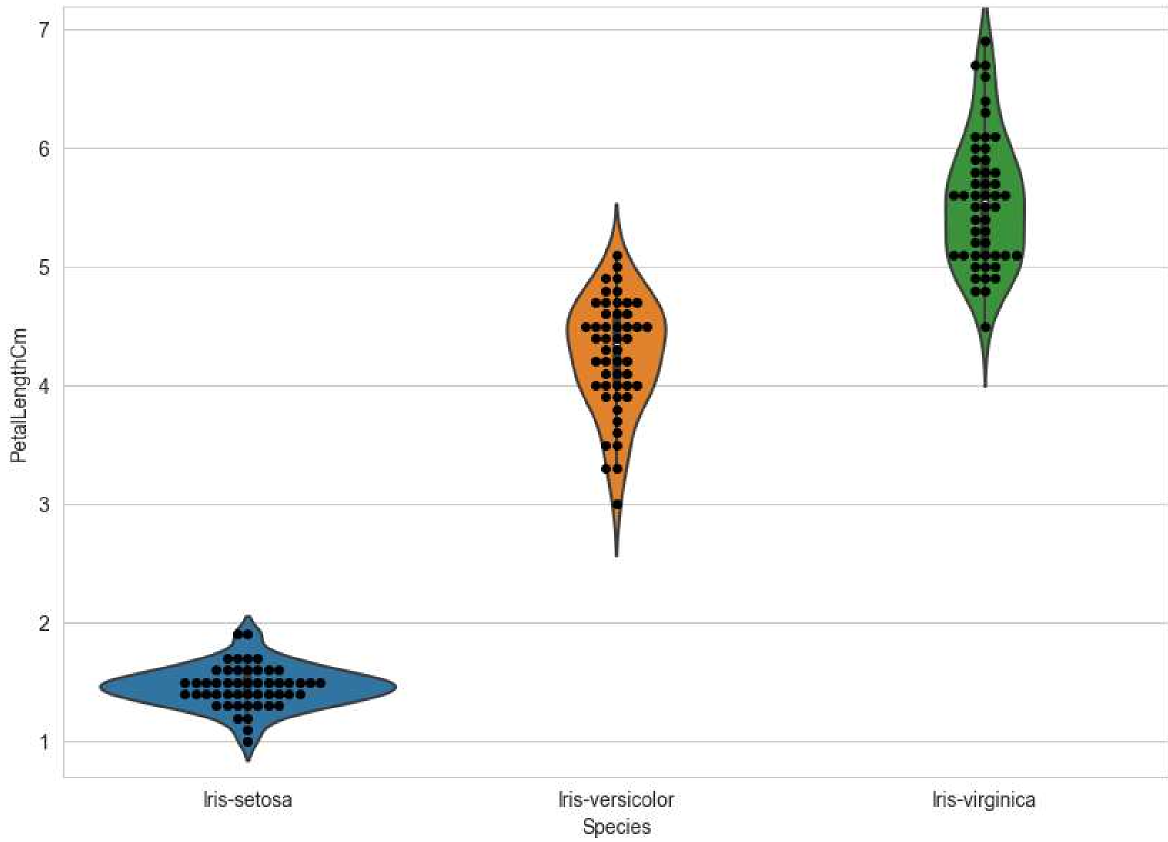
```
sns.set_style('darkgrid')
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.swarmplot(data=iris,x='Species',y='PetalLengthCm',hue='Species')
```



In [39]:

```
sns.set_style('whitegrid')
fig=plt.gcf()
fig.set_size_inches(10,7)

fig=sns.violinplot(data=iris,x='Species',y='PetalLengthCm')
fig=sns.swarmplot(data=iris,x='Species',y='PetalLengthCm',color='k',edgec
```

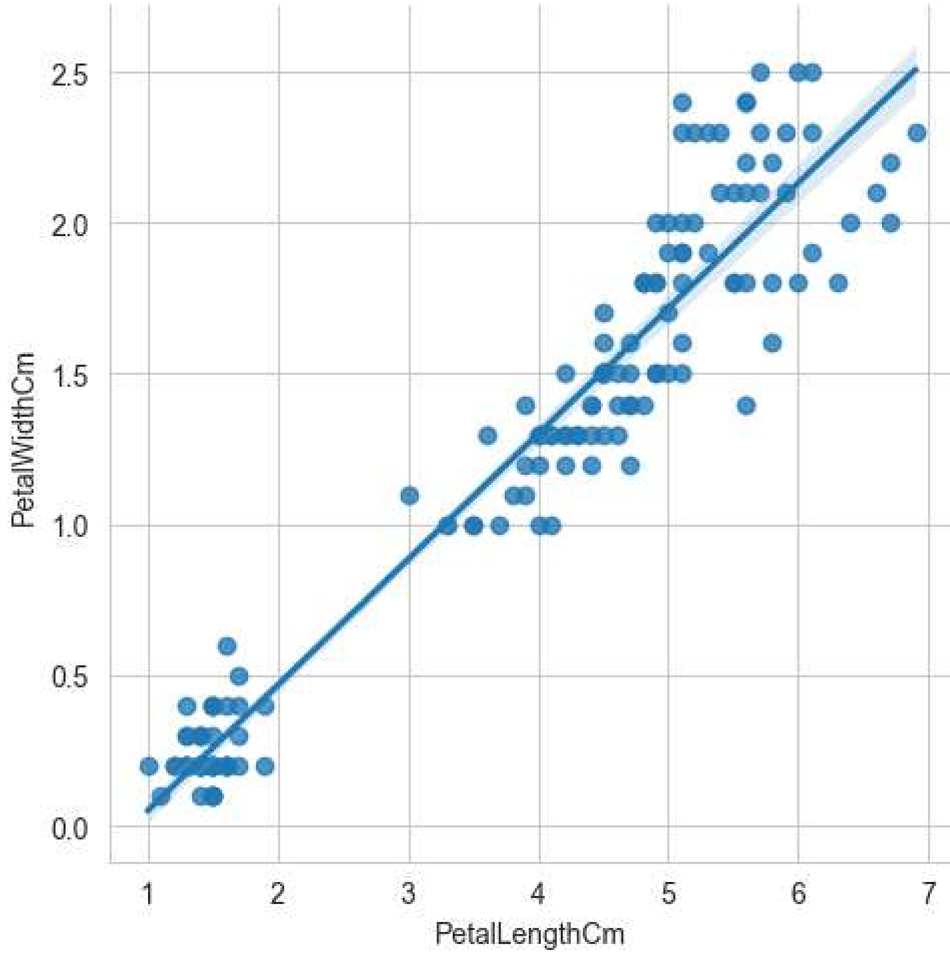


12) Linear Model (LM) Plot

- It is also called as regression plot.
- It gives relationship between two numerical variables using scatter plot and fitted regression line .
- It understanding how changes in one variable affect the other.

In [40]:

```
fig=sns.lmplot(data=iris,x='PetalLengthCm',y='PetalWidthCm')
```

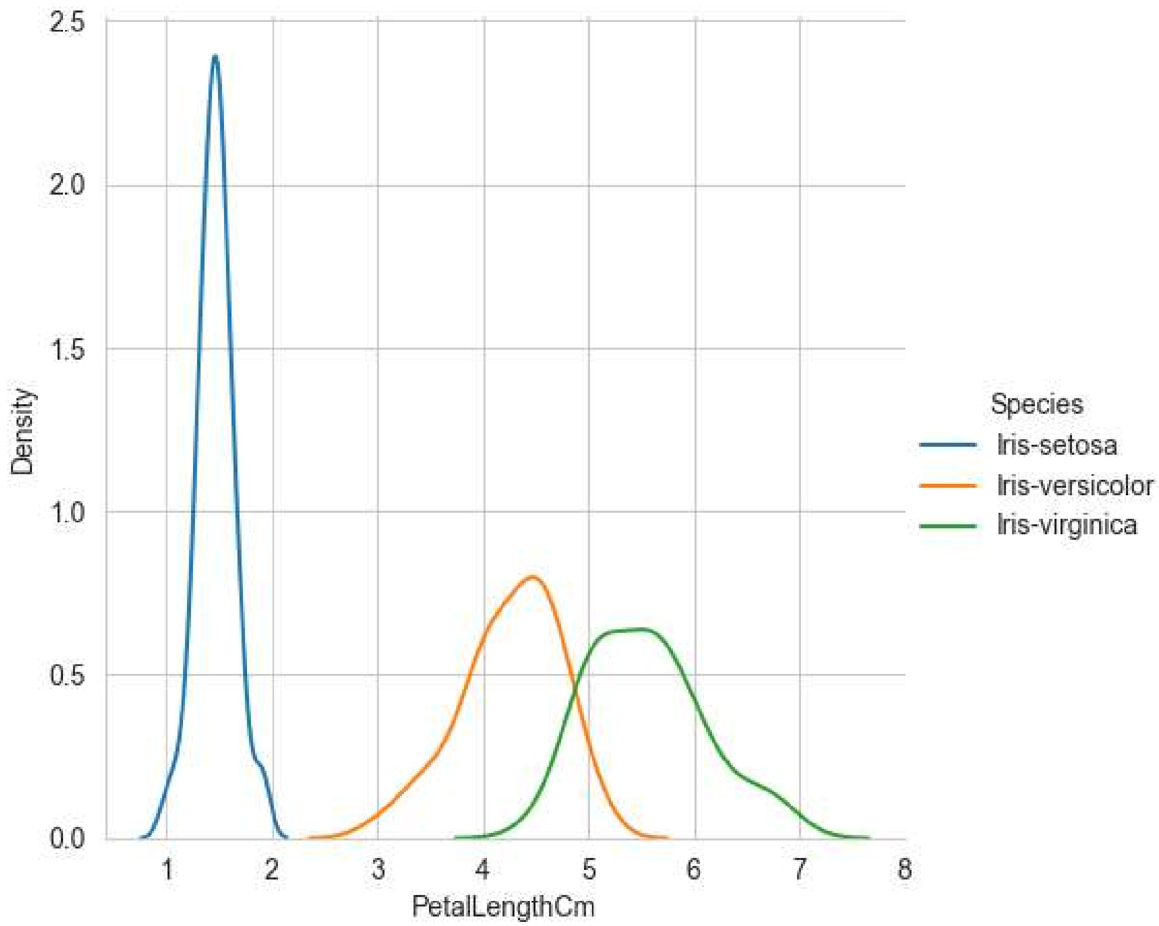


13) FacetGrid

- FacetGrid create a grid of subplots based on different levels of one or more categorical variables.

In [41]:
```
f=sns.FacetGrid(iris,hue="Species",height=5)
f.map(sns.kdeplot,'PetalLengthCm')
f.add_legend()
```

Out[41]: <seaborn.axisgrid.FacetGrid at 0x1ecf24e9f90>

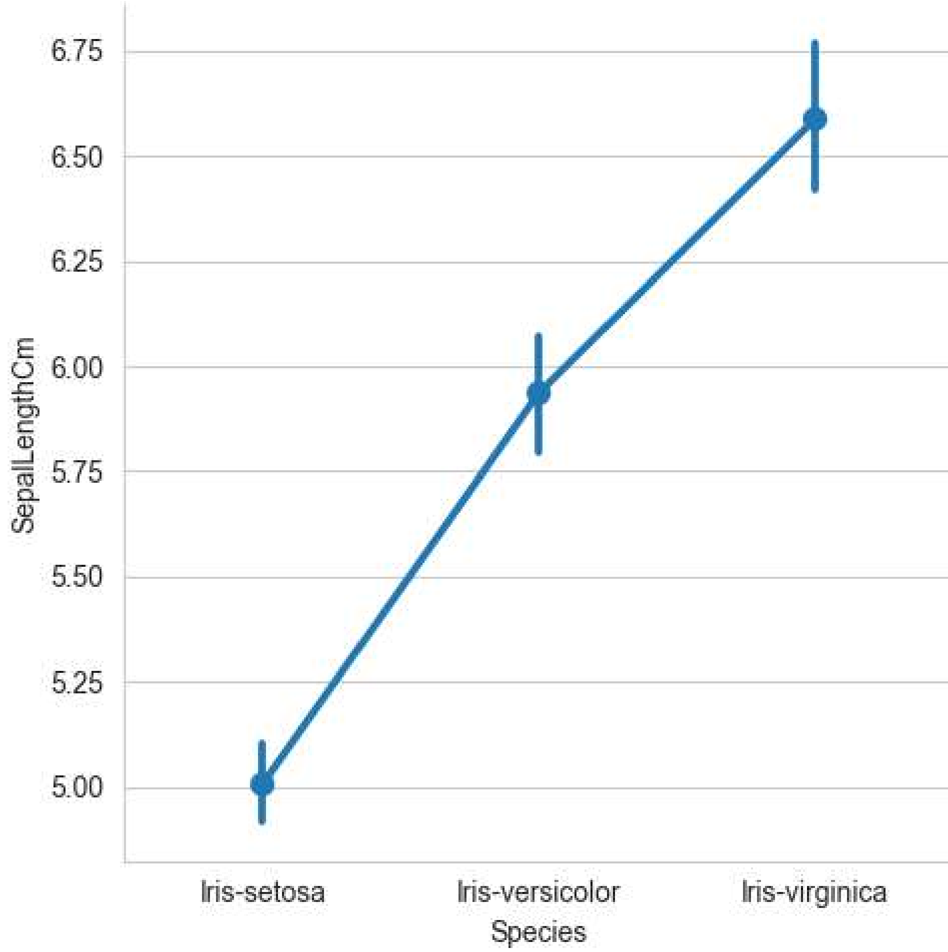


14) Catplot

- catplot is a new version of 'Factor' Plot
- Factor Plot was a function in old version(3.9.0)
- It create a variety of categorical plots including bar plot, count plot, points plot,and more.
- It create different types of plots based on categorical data.

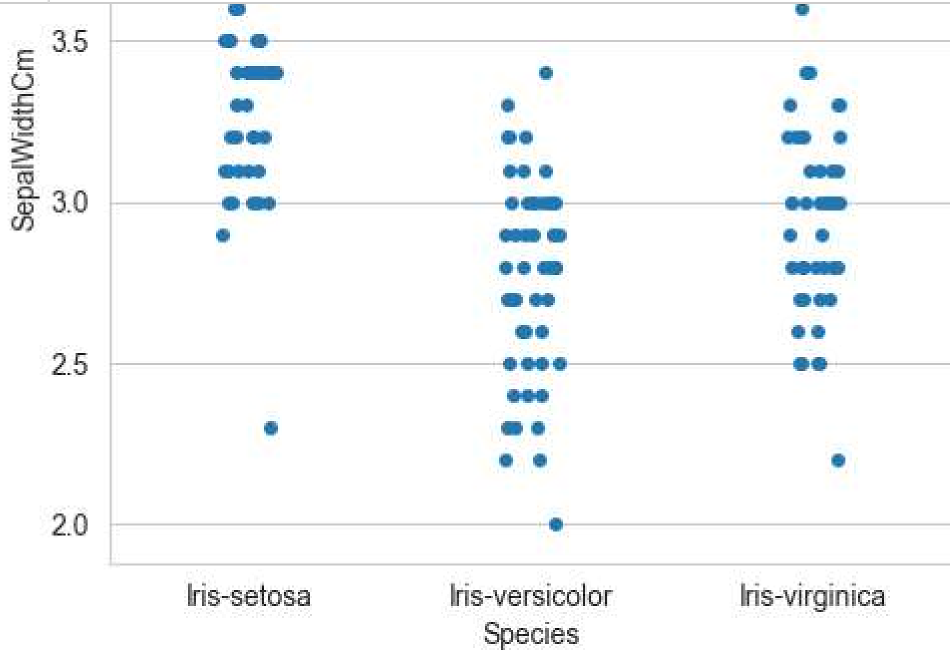
In [42]:

```
fig=sns.catplot(data=iris,x='Species',y='SepalLengthCm',kind='point')
```



In [43]:

```
k1=sns.catplot(x='Species',y='SepalLengthCm',data=iris)
k2=sns.catplot(x='Species',y='SepalWidthCm',data=iris)
k3=sns.catplot(x='Species',y='PetalLengthCm',data=iris)
k4=sns.catplot(x='Species',y='PetalWidthCm',data=iris)
```



15) Boxen Plot

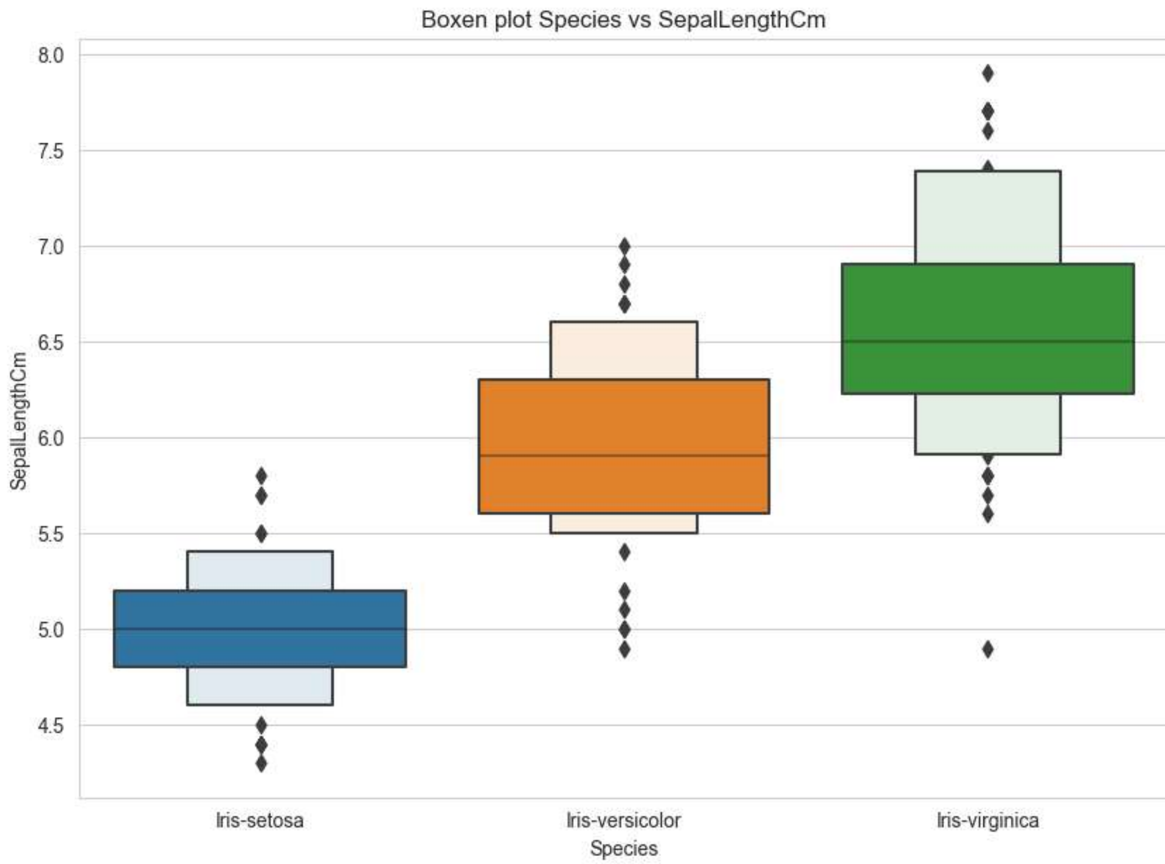
- It is also known as 'letter value plot'.
- A boxen plot is a variation of box plot
- It provide more detailed view of the distribution of data,especially for large datasets with many ouliers.

In [44]:

```
fig=plt.gcf()
fig.set_size_inches(10,7)

fig=sns.boxenplot(data=iris,x='Species',y='SepalLengthCm')
fig.set_title('Boxen plot Species vs SepalLengthCm')
```

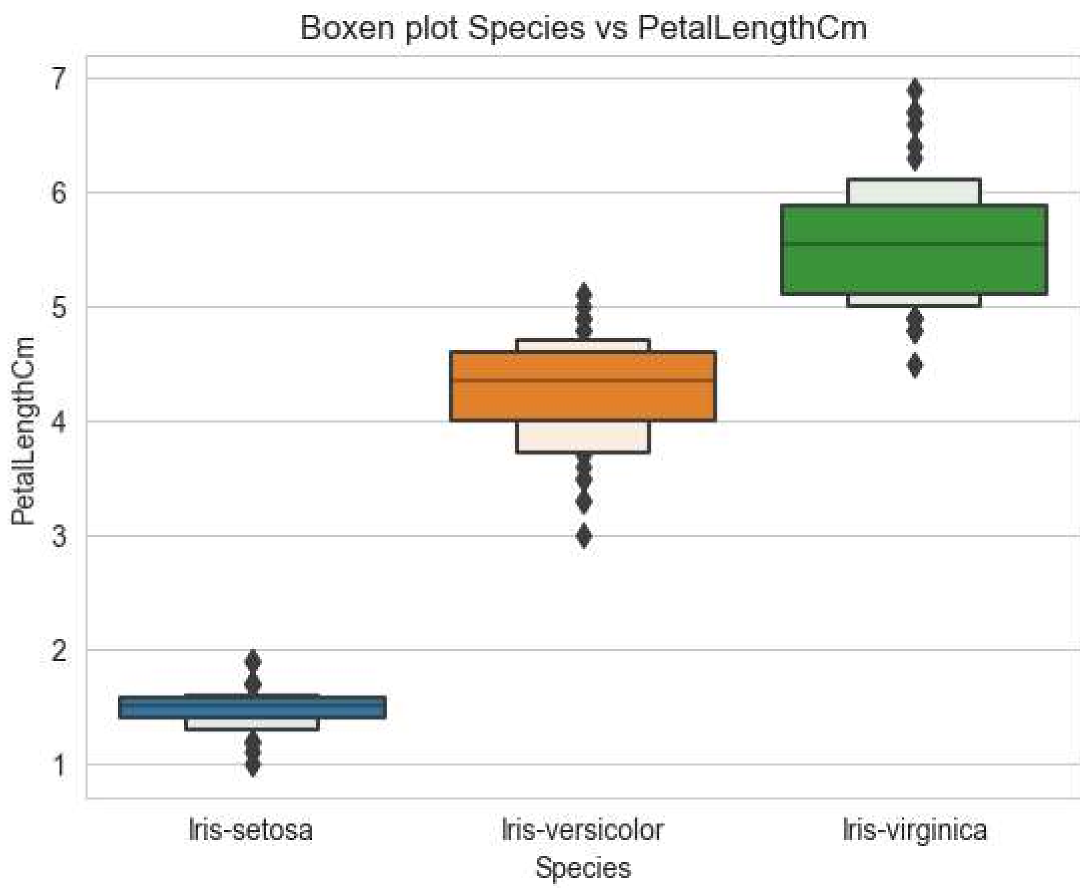
Out[44]: Text(0.5, 1.0, 'Boxen plot Species vs SepalLengthCm')



In [45]:

```
fig=sns.boxenplot(data=iris,x='Species',y='PetalLengthCm')
fig.set_title('Boxen plot Species vs PetalLengthCm')
```

Out[45]: Text(0.5, 1.0, 'Boxen plot Species vs PetalLengthCm')



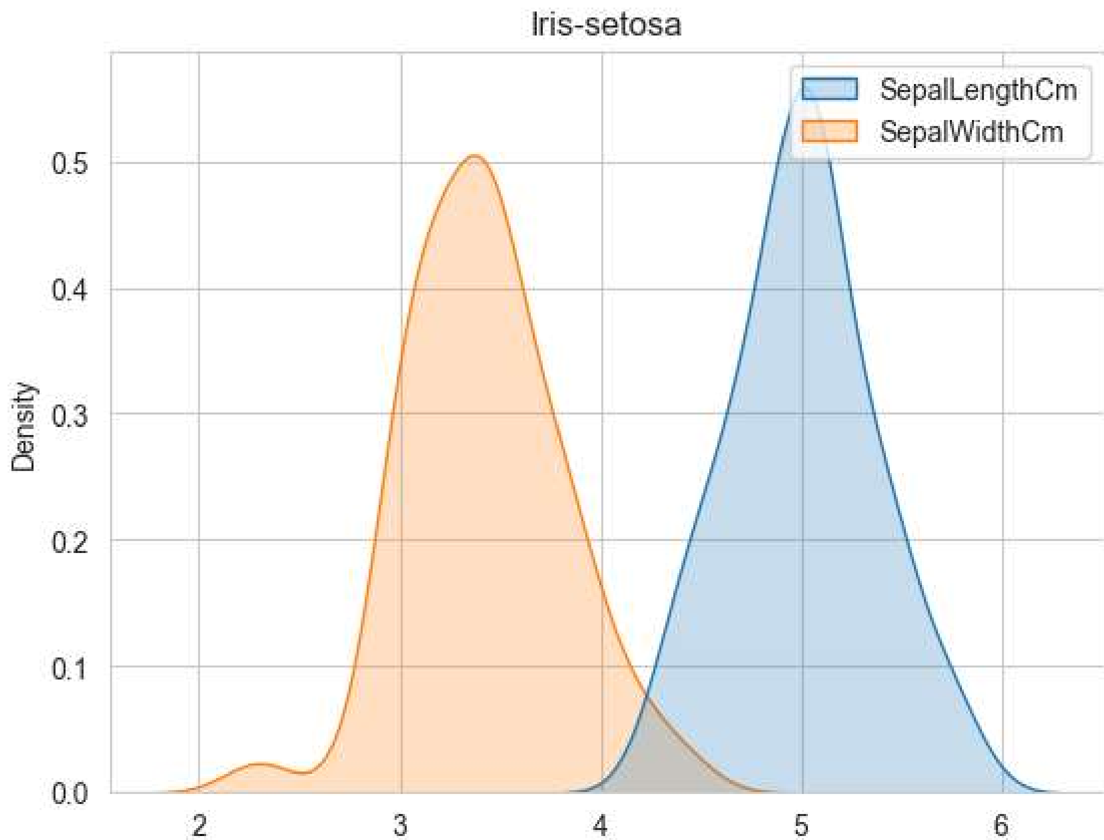
16) Kernel Density Estimation (kde) Plot

- It estimates the probality density function of a continuous variable.
- It showing data points are more likely to occur.
- kde plots are useful for understanding the shape and spread of a dataset's distribution.

In [46]:

```
#Create a kde plot of SepalLengthCm vs SepalWidthCm for setosa of flower
fig=sns.kdeplot(data=iris[iris.Species=='Iris-setosa'][['SepalLengthCm','S
fig.set_title('Iris-setosa')
```

Out[46]: Text(0.5, 1.0, 'Iris-setosa')

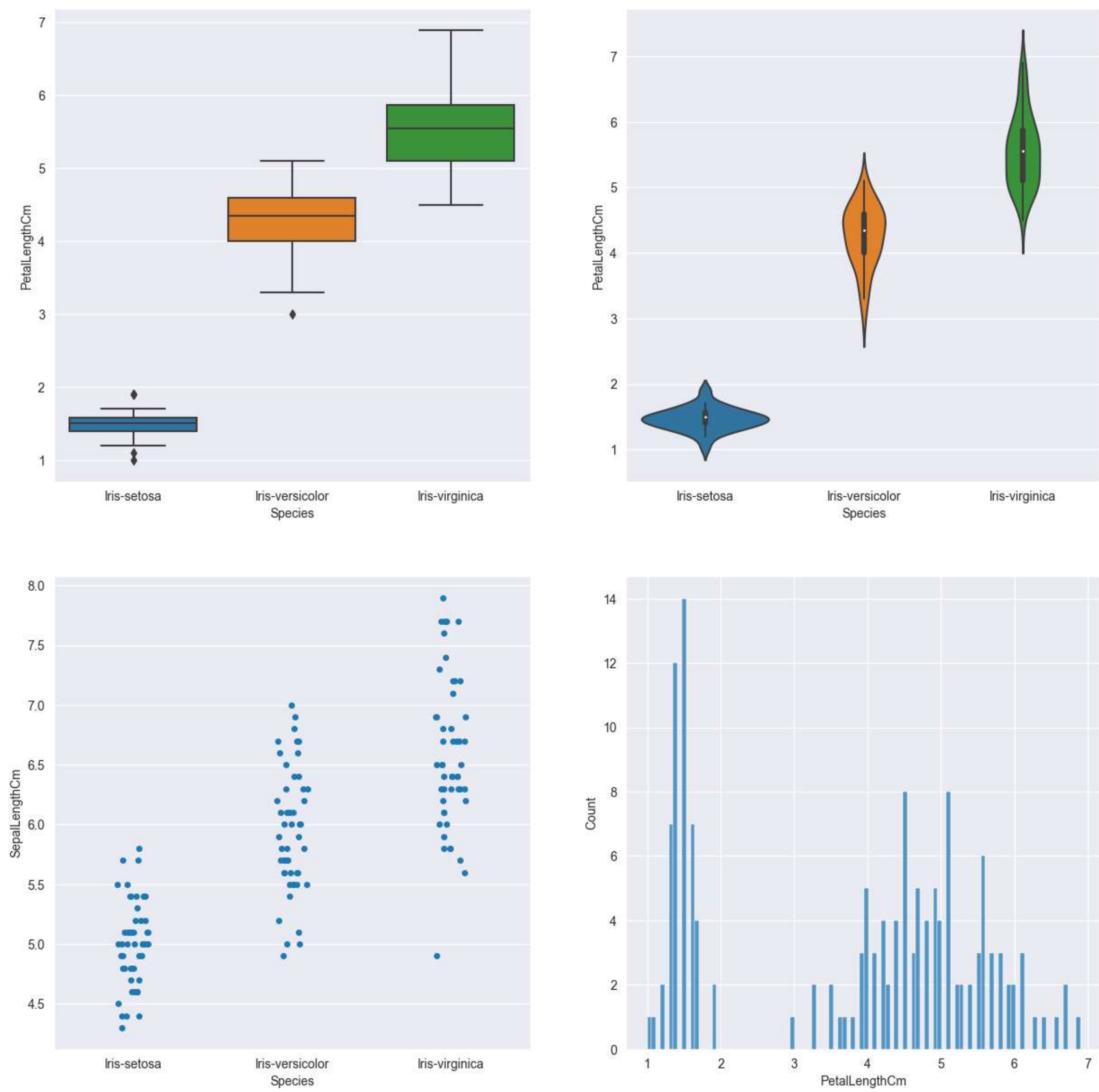


17) Dashboard

In [48]:

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

k1=sns.boxplot(data=iris,x='Species',y='PetalLengthCm',ax=axes[0,0])
k2=sns.violinplot(data=iris,x='Species',y='PetalLengthCm',ax=axes[0,1])
k3=sns.stripplot(data=iris,x='Species',y='SepalLengthCm',ax=axes[1,0])
k4=sns.histplot(iris.PetalLengthCm,bins=100,ax=axes[1,1])
```



18) Stacked Histogram

- It displays the distribution of multiple variables or categories stacked on top of each other.
- It's useful for comparing the distribution of different variables within the same dataset.

In [49]:
```
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 [50]:
```
iris['Species']=iris.Species.astype('category')
```

In [51]:
```
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    category
dtypes: category(1), float64(4)
memory usage: 5.1 KB
```

In [52]:

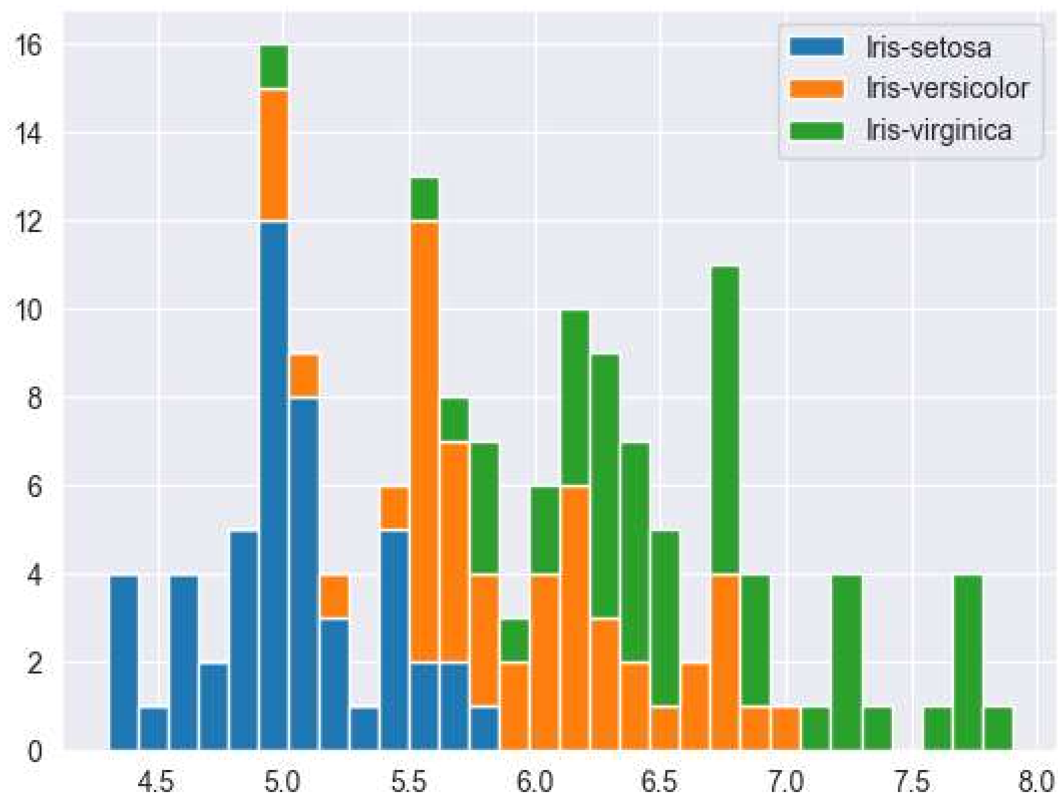
```python
#iris[iris.Species==iris.Species.cat.categories].SepalLengthCm

setosa=iris[iris.Species=='Iris-setosa'].SepalLengthCm
versicolor=iris[iris.Species=='Iris-versicolor'].SepalLengthCm
virginica=iris[iris.Species=='Iris-virginica'].SepalLengthCm

species=[setosa,versicolor,virginica]
labels=('Iris-setosa','Iris-versicolor','Iris-virginica')

fig=plt.hist(species,stacked=True,bins=30,label=labels)
plt.legend()
```

Out[52]: <matplotlib.legend.Legend at 0x1ecf24e59d0>



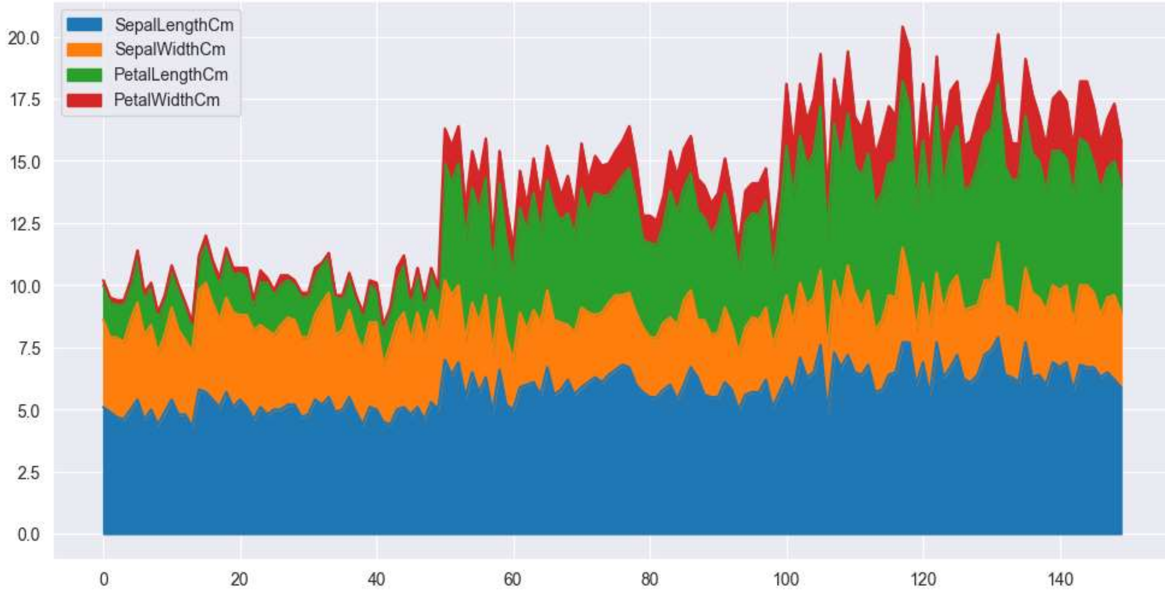
19) Area Plot

- An area plot also known as filled area plot or a stacked area plot.
- It displays the evolution of quantitative data over time or across categories.

In [53]:

```
1 iris.plot.area(y=['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWi
```

Out[53]: <Axes: >



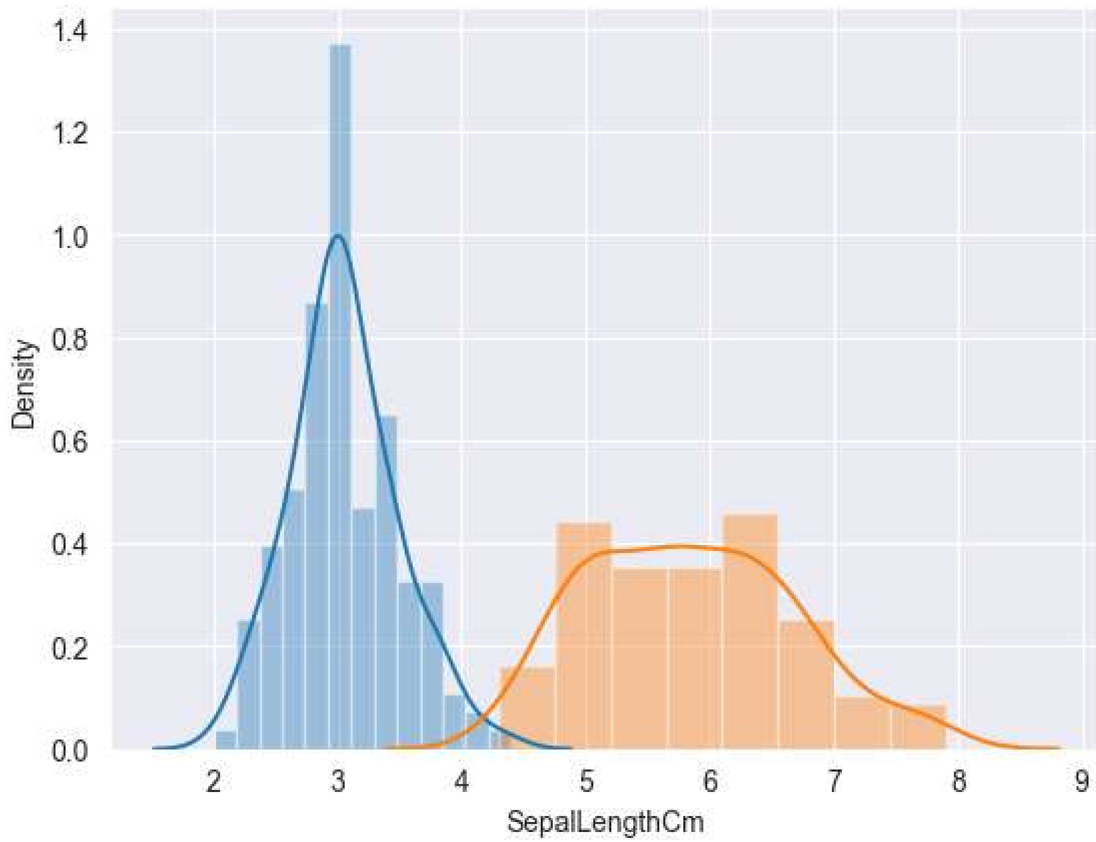
20) Distplot

- It create combination of histogram and kernel density estimate(KDE) plot.

In [54]:

```
sns.distplot(iris.SepalWidthCm)
sns.distplot(iris.SepalLengthCm)
```

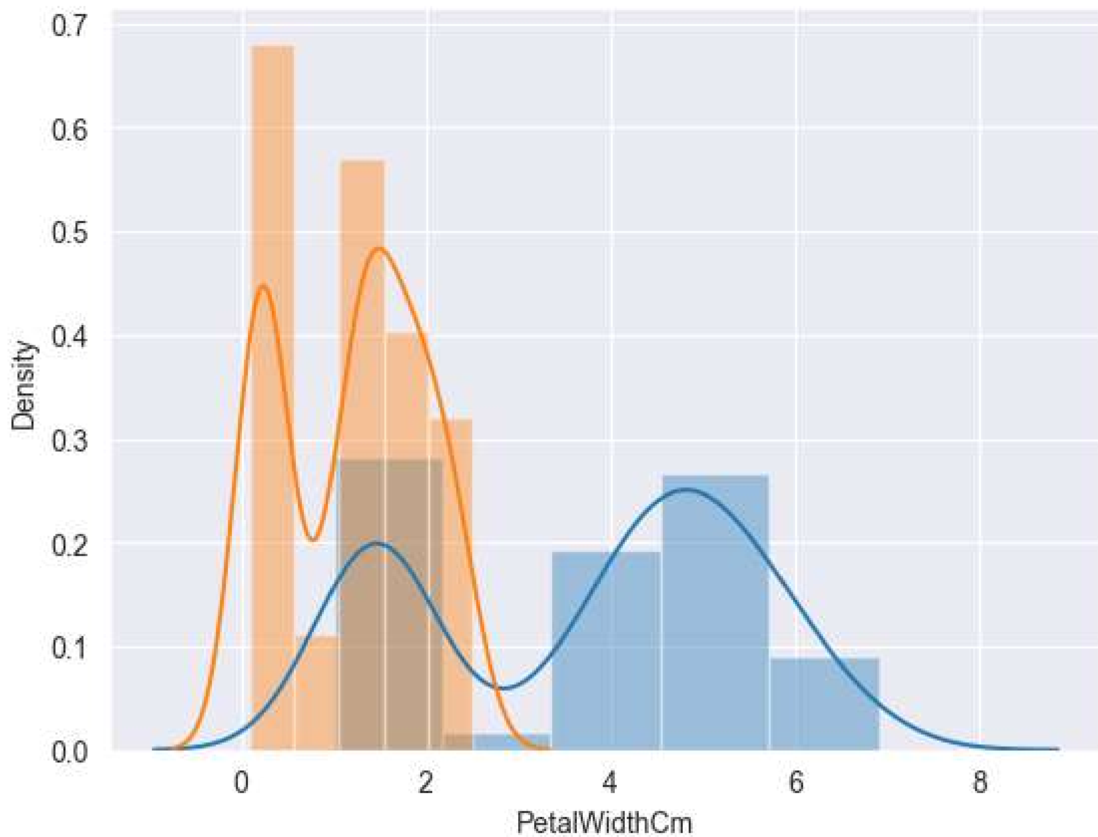
Out[54]: <Axes: xlabel='SepalLengthCm', ylabel='Density'>



In [55]:

```
sns.distplot(iris.PetalLengthCm)
sns.distplot(iris.PetalWidthCm)
```

Out[55]: <Axes: xlabel='PetalWidthCm', ylabel='Density'>



In [ ]:

```

```