LETS GROW MORE (VIRTUAL INTERNSHIP MAY 2023)

Name - Arushi

Data Science Intern

Task 3- Iris Flowers Classification ML Project

Importing Libraries

```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn.linear_model import LinearRegression
from sklearn import svm, datasets
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
import seaborn as sns
import matplotlib.pyplot as plt
```

Reading the data

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

Getting overall information about the dataset

```
iris.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

```
iris.shape
```

```
(150, 6)
```

✓ 0s completed at 12:17 PM

	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

```
#Checking for null values
iris.isnull().any()
```

```
Id               False
SepalLengthCm    False
SepalWidthCm     False
PetalLengthCm    False
PetalWidthCm     False
Species          False
dtype: bool
```

```
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
```

```
iris.nunique()
```

```
Id               150
SepalLengthCm     35
SepalWidthCm      23
PetalLengthCm     43
```

```
PetalWidthCm      22
Species            3
dtype: int64
```

```
iris.keys
```

```
<bound method NDFrame.keys of       Id  SepalLengthCm  SepalWidthCm  PetalLengthCm
PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   ...            ...           ...            ...           ...
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
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

[150 rows x 6 columns]>
```

```
iris.columns
```

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

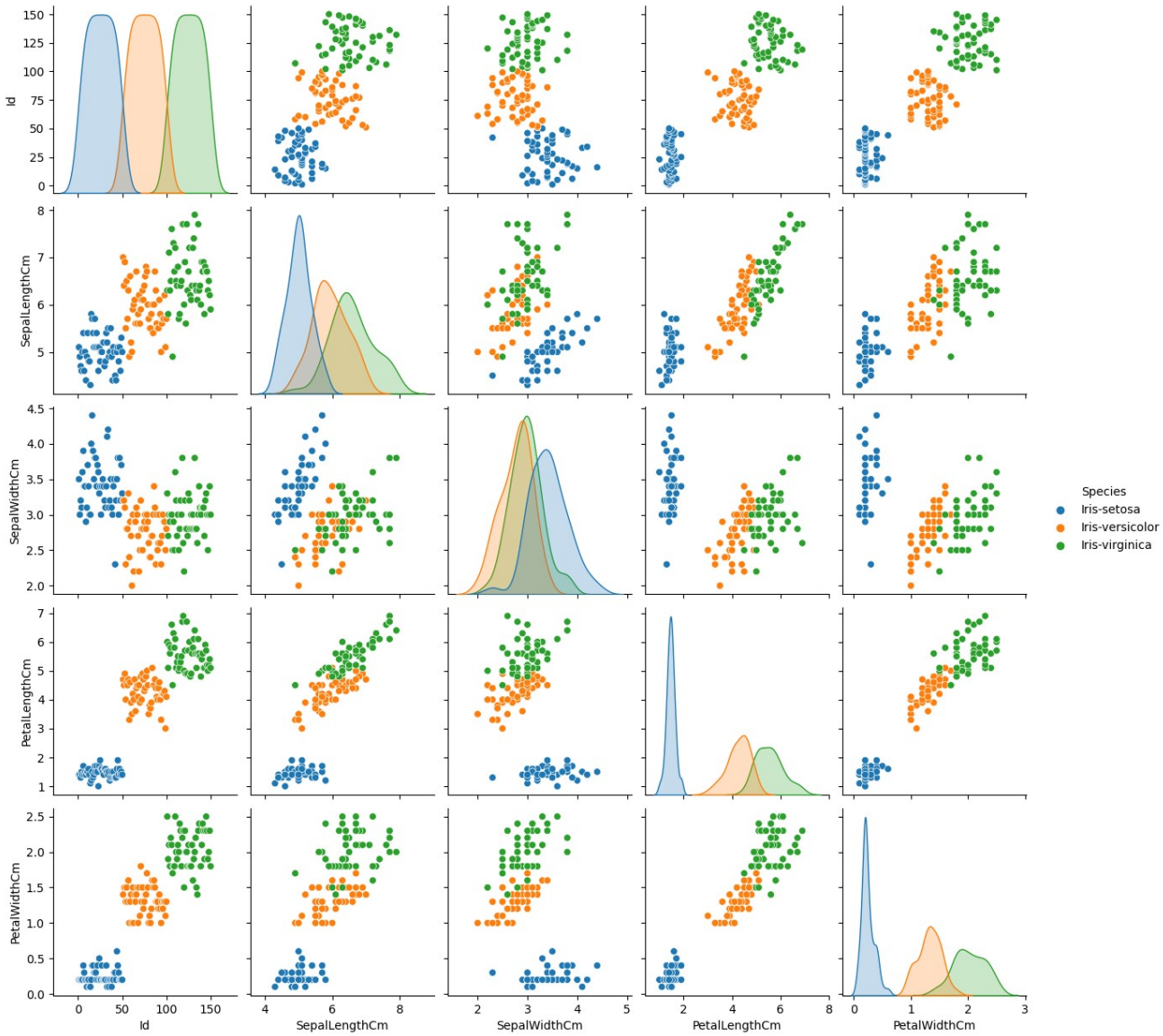
```
iris['Species'].value_counts()
```

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

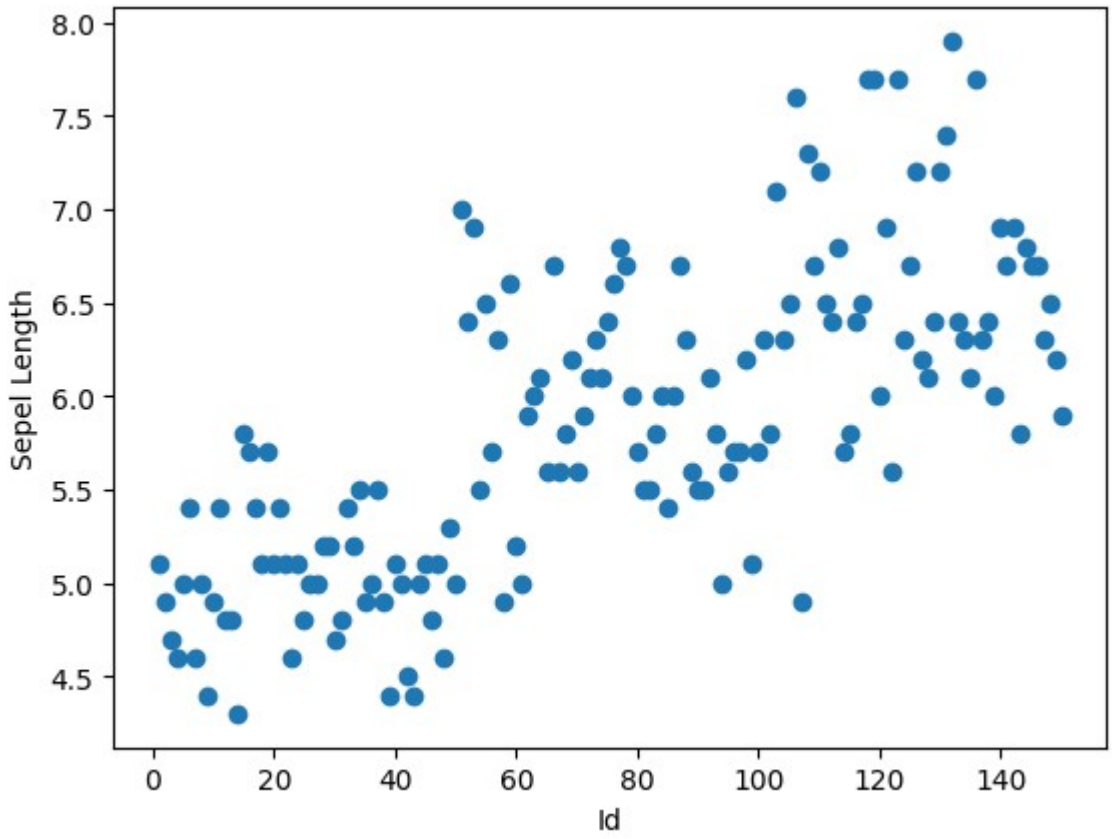
Plotting some graphs

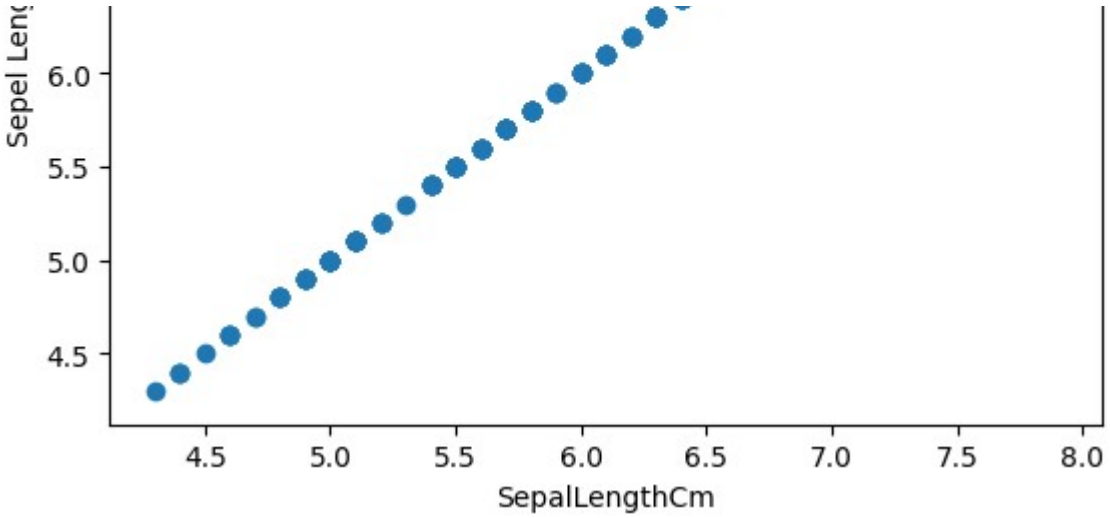
```
#Pair Plot
sns.pairplot(iris,hue='Species')
```

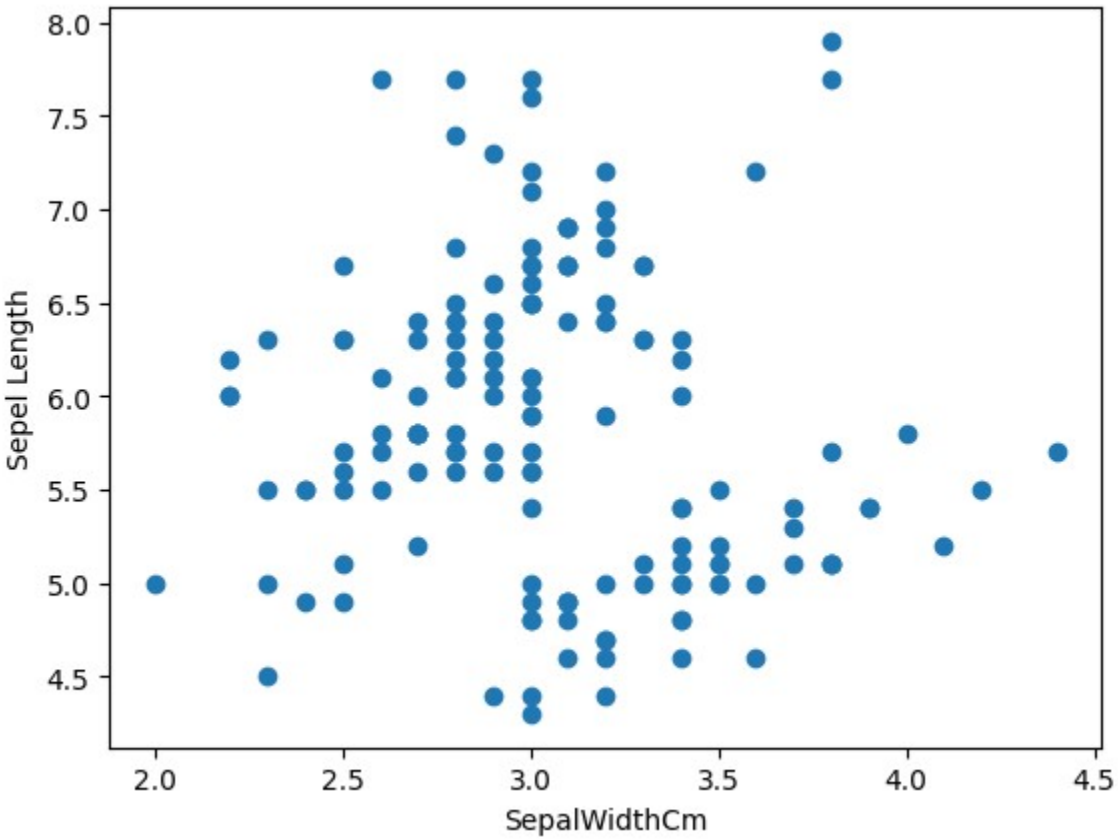
```
<seaborn.axisgrid.PairGrid at 0x7ffb1e8cb5e0>
```

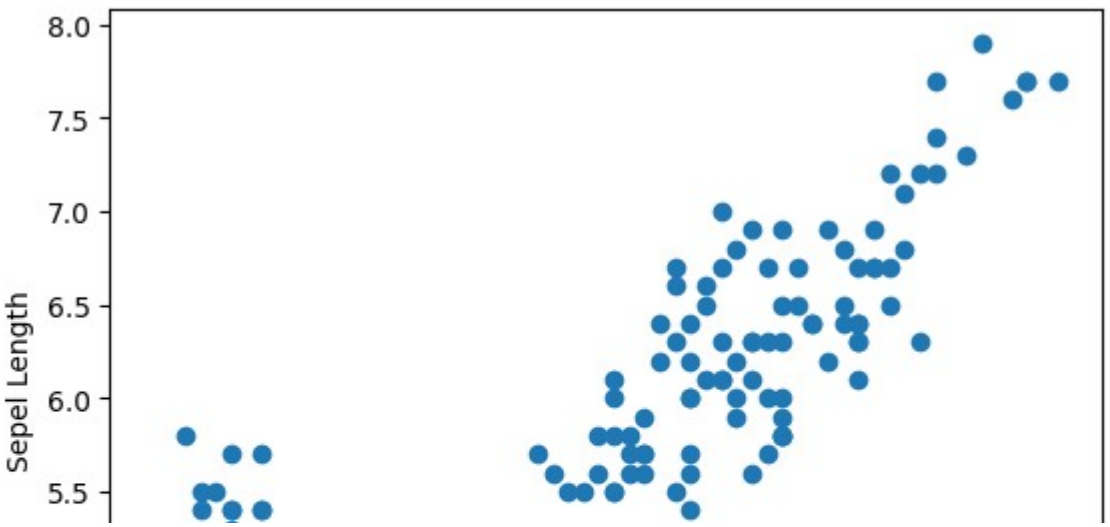


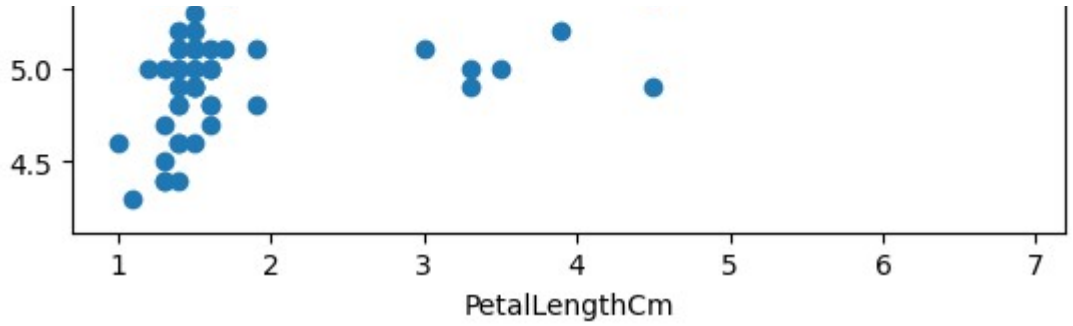
```
#Scatter Plot
for i in iris.columns:
    plt.scatter(iris[str(i)],iris['SepalLengthCm'])
    plt.xlabel(i)
    plt.ylabel("Sepel Length")
    plt.show()
```

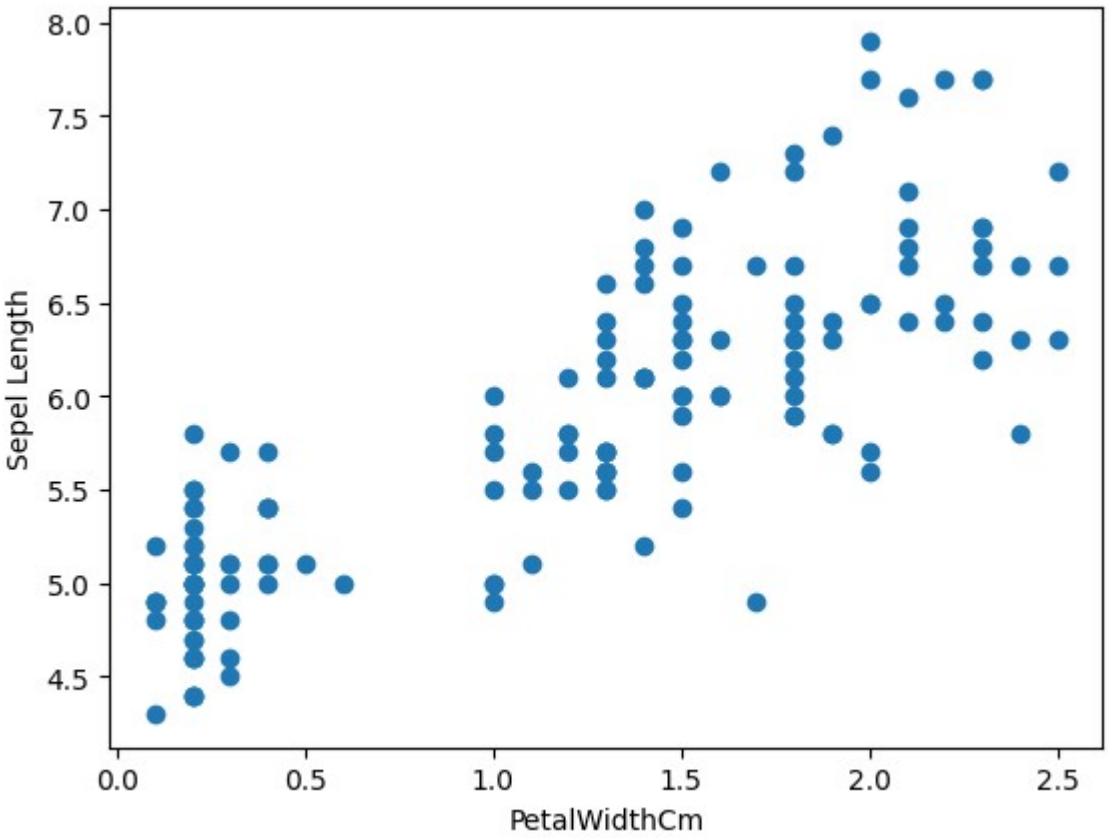


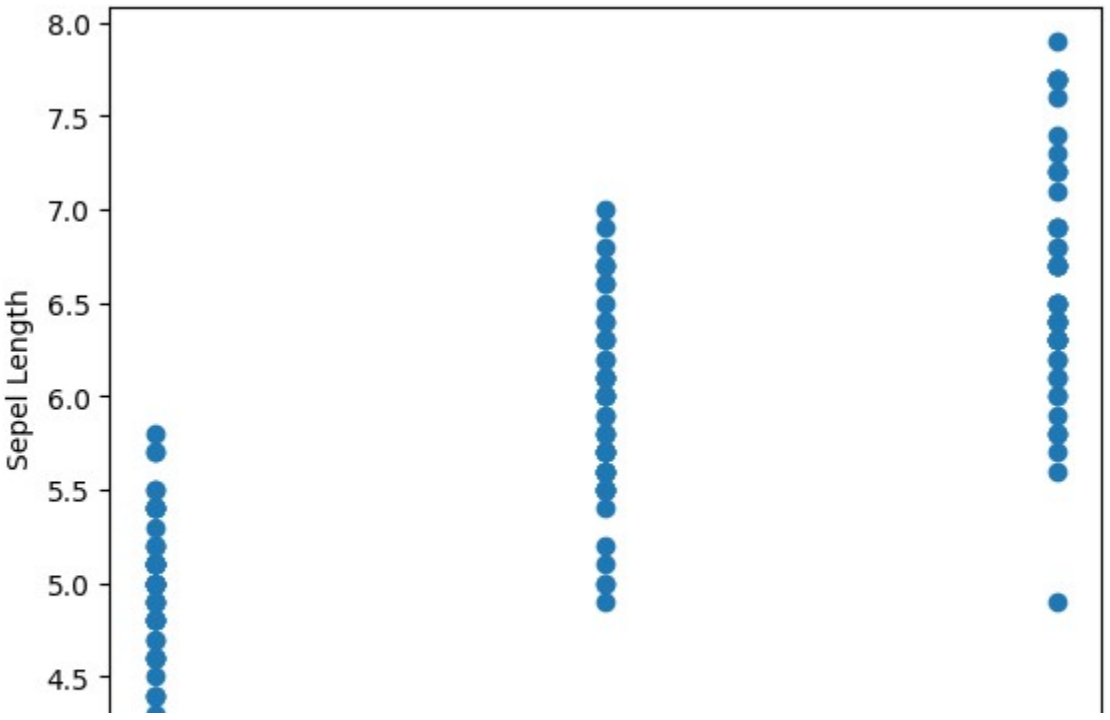

Sepel Leng
6.0
5.5
5.0
4.5
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
SepalLengthCm

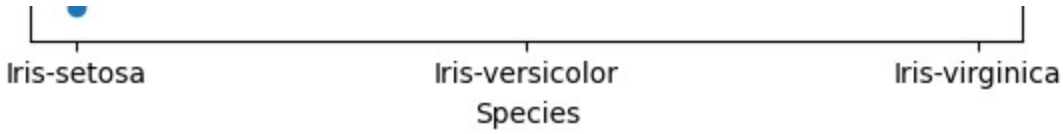

8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
Sepel Length
2.0
2.5
3.0
3.5
4.0
4.5
SepalWidthCm


8.0
7.5
7.0
6.5
6.0
5.5
Sepel Length

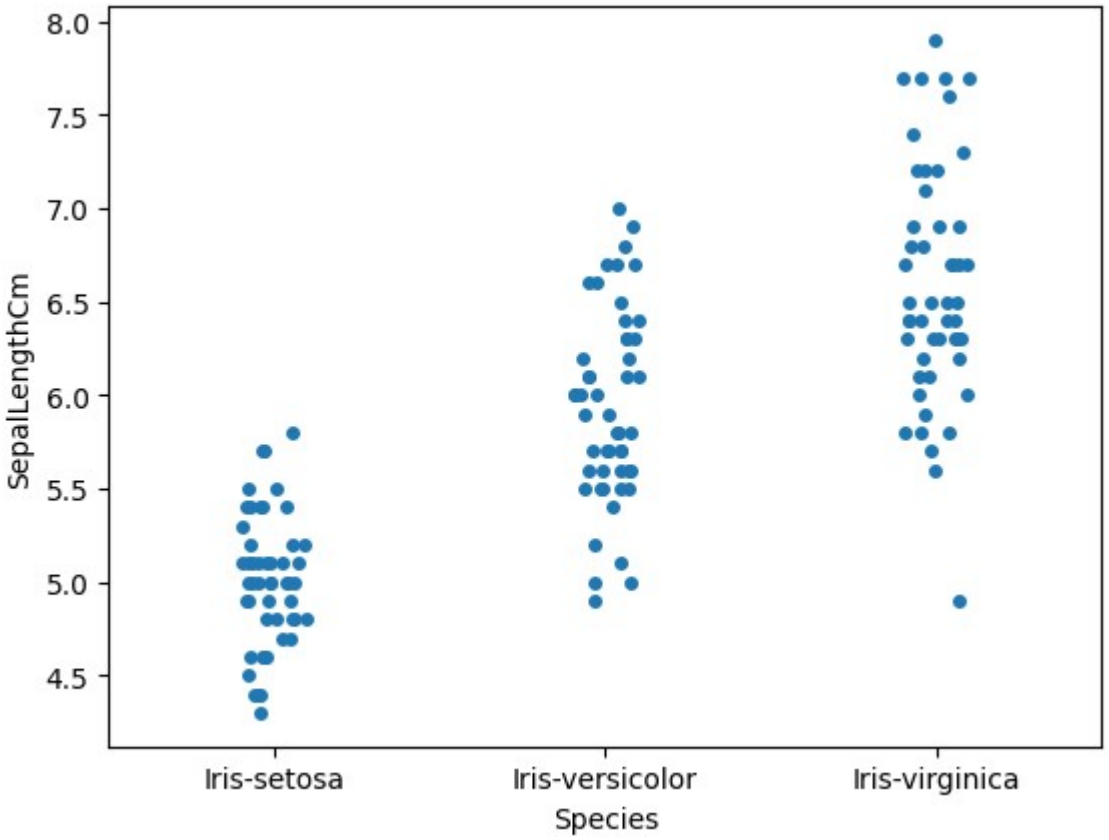

5.0
4.5
1
2
3
4
5
6
7
PetalLengthCm


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
Sepel Length
0.0
0.5
1.0
1.5
2.0
2.5
PetalWidthCm


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
Sepel Length

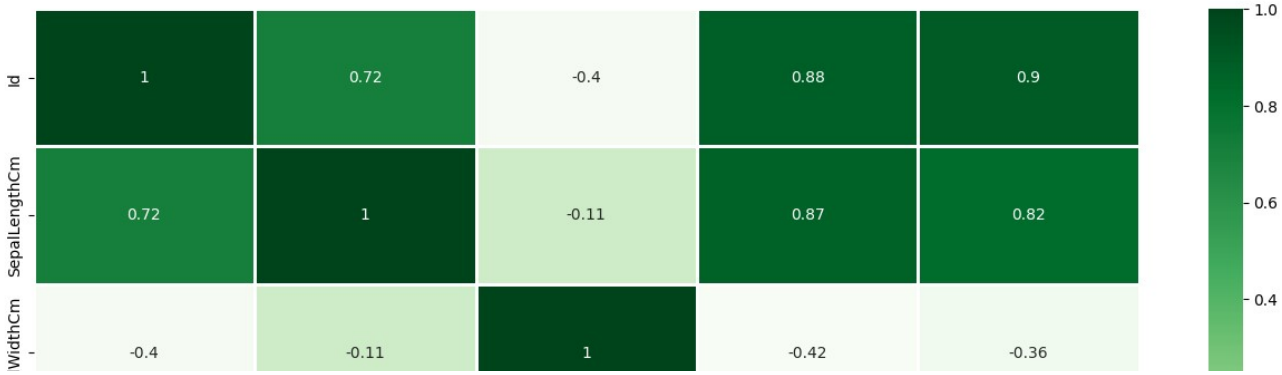


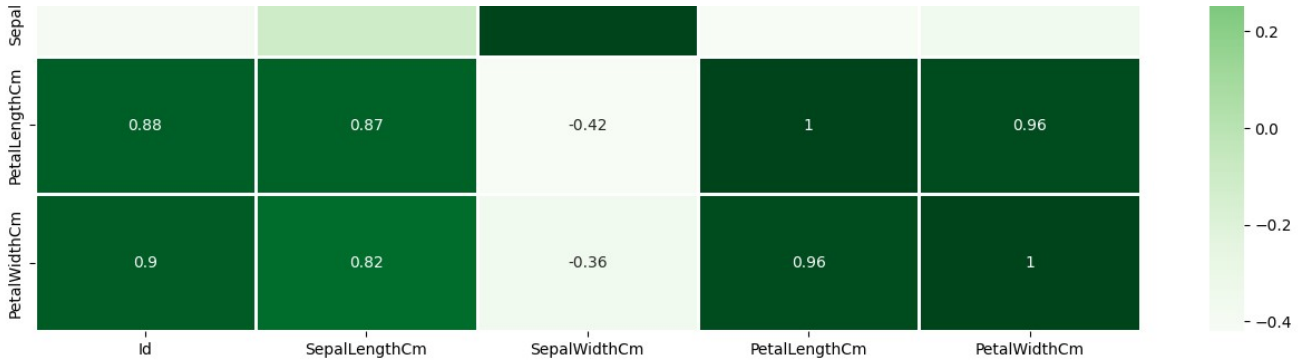

```
#Strip Plot
sns.stripplot( y= 'SepalLengthCm', x= 'Species',data=iris)
plt.show()
```




```
#Heat map
plt.figure(figsize=(16,8))
sns.heatmap(iris.corr(), annot= True, linewidth=2,linecolor='white',cmap='Greens')
plt.show()
```

```
<ipython-input-37-65c467b6f150>:3: FutureWarning: The default value of numeric_only i
  sns.heatmap(iris.corr(), annot= True, linewidth=2,linecolor='white',cmap='Greens')
```





Splitting the dataset into train and test

```
train,test= train_test_split(iris,test_size=0.25)
```

```
train
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
110	111	6.5	3.2	5.1	2.0	Iris-virginica
88	89	5.6	3.0	4.1	1.3	Iris-versicolor
122	123	7.7	2.8	6.7	2.0	Iris-virginica
31	32	5.4	3.4	1.5	0.4	Iris-setosa
112	113	6.8	3.0	5.5	2.1	Iris-virginica
...	...	...	...	...	...	...
63	64	6.1	2.9	4.7	1.4	Iris-versicolor
140	141	6.7	3.1	5.6	2.4	Iris-virginica

69	70	5.6	2.5	3.9	1.1	Iris-versicolor
9	10	4.9	3.1	1.5	0.1	Iris-setosa
33	34	5.5	4.2	1.4	0.2	Iris-setosa

112 rows × 6 columns

```
test
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
93	94	5.0	2.3	3.3	1.0	Iris-versicolor
128	129	6.4	2.8	5.6	2.1	Iris-virginica
85	86	6.0	3.4	4.5	1.6	Iris-versicolor
149	150	5.9	3.0	5.1	1.8	Iris-virginica
139	140	6.9	3.1	5.4	2.1	Iris-virginica
100	101	6.3	3.3	6.0	2.5	Iris-virginica
80	81	5.5	2.4	3.8	1.1	Iris-versicolor
145	146	6.7	3.0	5.2	2.3	Iris-virginica
74	75	6.4	2.9	4.3	1.3	Iris-versicolor
19	20	5.1	3.8	1.5	0.3	Iris-setosa
13	14	4.3	3.0	1.1	0.1	Iris-setosa
35	36	5.0	3.2	1.2	0.2	Iris-setosa
32	33	5.2	4.1	1.5	0.1	Iris-setosa
66	67	5.6	3.0	4.5	1.5	Iris-versicolor
104	105	6.5	3.0	5.8	2.2	Iris-virginica
92	93	5.8	2.6	4.0	1.2	Iris-versicolor
64	65	5.6	2.9	3.6	1.3	Iris-versicolor
68	69	6.2	2.2	4.5	1.5	Iris-versicolor
2	3	4.7	3.2	1.3	0.2	Iris-setosa
53	54	5.5	2.3	4.0	1.3	Iris-versicolor
36	37	5.5	3.5	1.3	0.2	Iris-setosa
125	126	7.2	3.2	6.0	1.8	Iris-virginica
84	85	5.4	3.0	4.5	1.5	Iris-versicolor
143	144	6.8	3.2	5.9	2.3	Iris-virginica

137	138	6.4	3.1	5.5	1.8	Iris-virginica
109	110	7.2	3.6	6.1	2.5	Iris-virginica
81	82	5.5	2.4	3.7	1.0	Iris-versicolor
144	145	6.7	3.3	5.7	2.5	Iris-virginica
24	25	4.8	3.4	1.9	0.2	Iris-setosa
5	6	5.4	3.9	1.7	0.4	Iris-setosa
41	42	4.5	2.3	1.3	0.3	Iris-setosa
113	114	5.7	2.5	5.0	2.0	Iris-virginica
120	121	6.9	3.2	5.7	2.3	Iris-virginica
47	48	4.6	3.2	1.4	0.2	Iris-setosa
38	39	4.4	3.0	1.3	0.2	Iris-setosa
147	148	6.5	3.0	5.2	2.0	Iris-virginica
37	38	4.9	3.1	1.5	0.1	Iris-setosa
29	30	4.7	3.2	1.6	0.2	Iris-setosa

```
x_train=train[['SepalLengthCm' ,    'SepalWidthCm' ,    'PetalLengthCm' ,   'PetalWidthCm']]
```

```
y_train=train.Species
```

```
x_test=test[['SepalLengthCm' ,  'SepalWidthCm' ,    'PetalLengthCm' ,   'PetalWidthCm']]
```

```
y_test=test.Species
```

```
x_train
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
110	6.5	3.2	5.1	2.0
88	5.6	3.0	4.1	1.3
122	7.7	2.8	6.7	2.0
31	5.4	3.4	1.5	0.4
112	6.8	3.0	5.5	2.1
...	...	...	...	...
63	6.1	2.9	4.7	1.4

63	6.1	2.9	4.7	1.4
140	6.7	3.1	5.6	2.4
69	5.6	2.5	3.9	1.1
9	4.9	3.1	1.5	0.1
33	5.5	4.2	1.4	0.2

112 rows × 4 columns

```
y_train
```

```
110     Iris-virginica
88     Iris-versicolor
122     Iris-virginica
31         Iris-setosa
112     Iris-virginica
            ...
63     Iris-versicolor
140     Iris-virginica
69     Iris-versicolor
9          Iris-setosa
33         Iris-setosa
Name: Species, Length: 112, dtype: object
```

Support Vector Machine

```
from sklearn.svm import SVC
svn=SVC()
svn.fit(x_train,y_train)
```

▾ SVC
SVC()

```
#Prediction
predictions=svn.predict(x_test)
```

Checking accuracy of data

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,predictions)
```

```
1.0
```

Classification report

```
from sklearn.metrics import classification_report
classification_report(y_test,predictions)
```

```
'                 precision    recall  f1-score   support\n\n    Iris-setosa
1.00      1.00      1.00        13\nIris-versicolor       1.00      1.00      1.00
11\n Iris-virginica       1.00      1.00      1.00        14\n\n       accuracy
1.00        38\n      macro avg       1.00      1.00      1.00        38\n   weighte
```

Test Model

```
x_new=np.array([[2,3,5,5],[1,2,3.8,0],[5.3,2.5,4.6,1.9]])
predictions=svn.predict(x_new)
print("Predictions of Species: {}".format(predictions))
```

```
Predictions of Species: ['Iris-virginica' 'Iris-versicolor' 'Iris-virginica']
/usr/local/lib/python3.9/dist-packages/sklearn/base.py:439: UserWarning: X does not h
  warnings.warn(
```

THANK YOU
