Aadarsh Goyal

MIS: 111915001

--

Import basic libraries

In [1]:

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

In [2]:

```
df = pd.read_csv("iris.csv")
```

In [3]:

```
df.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]:

```
df.shape
```

Out[4]:

```
(150, 6)
```

In [5]:

```
df.describe()
```

Out[5]:

	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

Dropping unwanted column

In [6]:

```
df.drop(columns="Id",inplace=True)
```

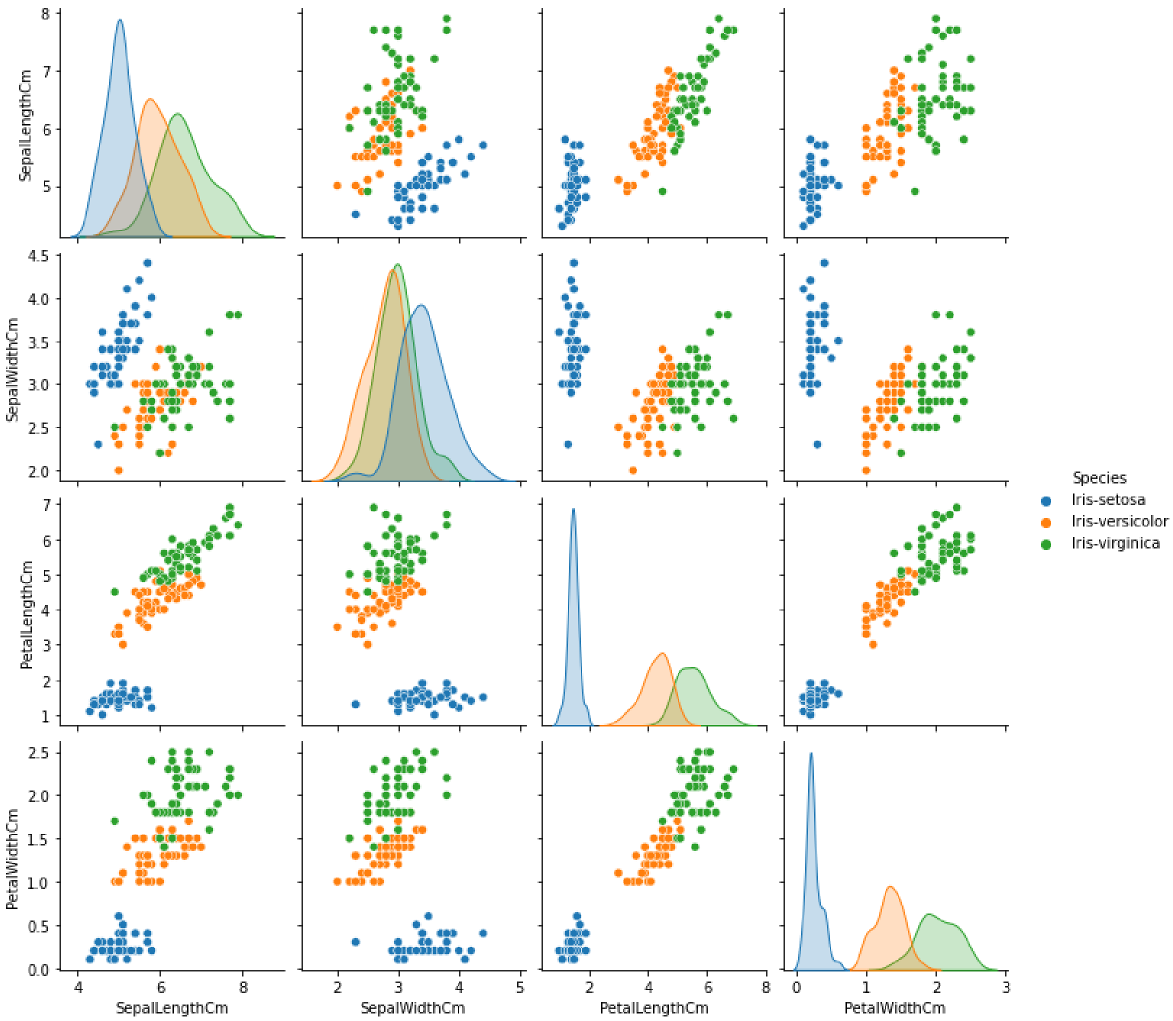
Visualization

In [7]:

```
sns.pairplot(df, hue='Species')
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x2326bdf4b08>
```



In [8]:

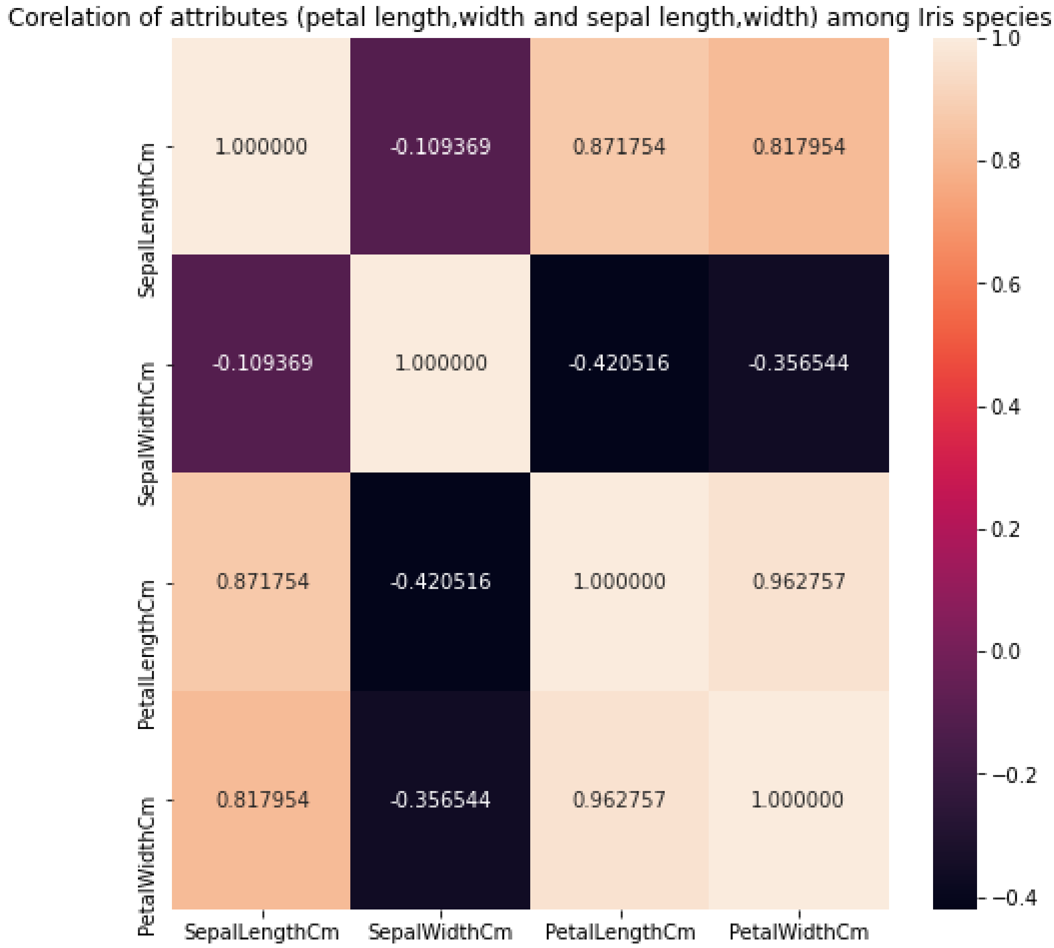
```
df.corr()
```

Out[8]:

	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 [9]:

```
plt.subplots(figsize = (8,8))
sns.heatmap(df.corr(),annot=True,fmt="f").set_title("Corelation of attributes (petal length
plt.show()
```



The Heatmap Shows that petal-Length and petal-Width are highly corelated

In [10]:

```
X=df.iloc[:,0:4].values
y=df.iloc[:,4].values
```

Label Encoding

In [11]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
```

Test, train Split

In [12]:

```
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=0)
```

Train Model

In [13]:

```
from sklearn.naive_bayes import GaussianNB

gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
```

Out[13]:

```
GaussianNB()
```

In [14]:

```
prediction = gaussian.predict(X_test)
```

See the results for predictions

In [15]:

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

g_score = gaussian.score(X_test, y_test)
conf_mat = confusion_matrix(y_test, prediction)
accuracy = accuracy_score(y_test, prediction)

print('Confusion matrix for Naive Bayes:\n',conf_mat)
print('\nAccuracy_Naive Bayes: ', accuracy*100 , "%")
print("Gaussian Naive Bayes score: ", g_score*100, "%")
```

```
Confusion matrix for Naive Bayes:
 [[16  0  0]
 [ 0 18  0]
 [ 0  0 11]]

Accuracy_Naive Bayes:  100.0 %
Gaussian Naive Bayes score:  100.0 %
```

We get a whopping 100% accuracy