LGM VIRTUAL INTERNSHIP PROGRAM AUGUST 2021

BEGINNER LEVEL TASK

Iris Flowers Classification ML Project

BY: Prateek Kumar

In [1]:

```
# Import libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import scipy as sp
import warnings
import os
warnings.filterwarnings("ignore")
```

In [2]:

```
# Load Dataset
df = pd.read_csv("C:/Users/prate/Downloads/IRISS.csv")
```

In [3]:

```
#Head function
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]:

```
#Tail Function
df.tail()
```

Out[4]:

	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
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

In [5]:

```
# Info function
df.info()
```

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

In [6]:

```
# Describe function
df.describe()
```

Out[6]:

	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

In [7]:

```python
# Shape function
df.shape
```

Out[7]:

```
(150, 6)
```

In [8]:

```python
# Check missing values
df.isnull()
```

Out[8]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
145	False	False	False	False	False	False
146	False	False	False	False	False	False
147	False	False	False	False	False	False
148	False	False	False	False	False	False
149	False	False	False	False	False	False

150 rows × 6 columns

In [9]:

```python
# Check missing values in dataset
df.isnull().sum()
```

Out[9]:

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

In [10]:

```
# Count missing values
df.isnull().any()
```

Out[10]:

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

In [11]:

```
# Columns
df.columns
```

Out[11]:

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

In [12]:

```
df.Id.unique().shape
```

Out[12]:

```
(150,)
```

In [13]:

```
# Dtypes attributes
df.dtypes
```

Out[13]:

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

In [14]:

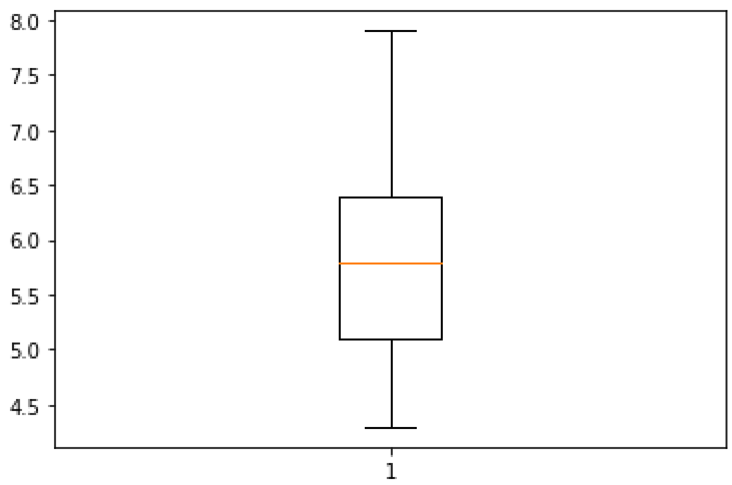
```
# Correlation
df.corr()
```

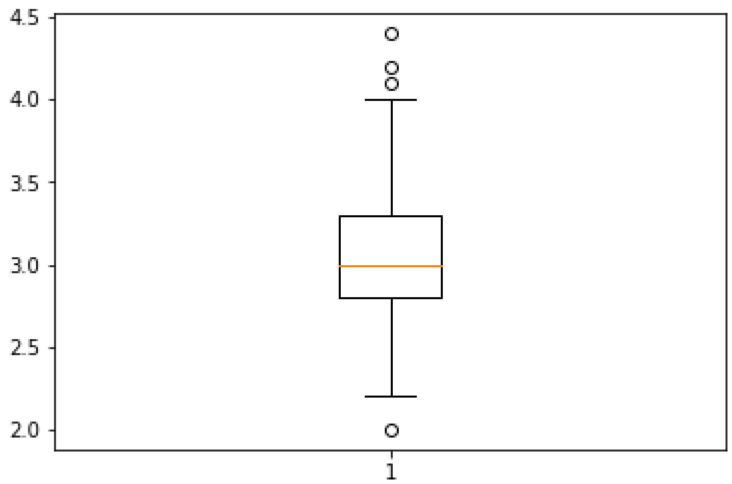
Out[14]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Id	1.000000	0.716676	-0.397729	0.882747	0.899759
SepalLengthCm	0.716676	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.397729	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.882747	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.899759	0.817954	-0.356544	0.962757	1.000000

In [55]:

```
# Boxplot figure
plt.figure(1)
plt.boxplot([df['SepalLengthCm']])
plt.figure(2)
plt.boxplot([df['SepalWidthCm']])
plt.show()
```



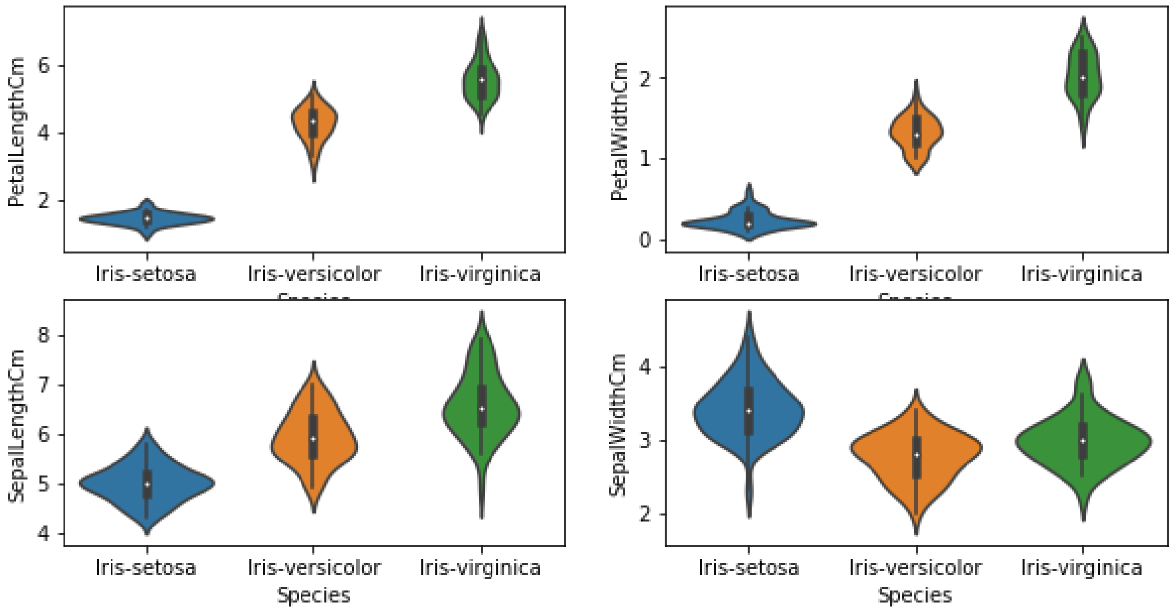


In [59]:

```
# Violin Plot
plt.figure(figsize=(10,5))
plt.subplot(2,2,1)
sns.violinplot(x='Species',y='PetalLengthCm',data=df)
plt.subplot(2,2,2)
sns.violinplot(x='Species',y='PetalWidthCm',data=df)
plt.subplot(2,2,3)
sns.violinplot(x='Species',y='SepalLengthCm',data=df)
plt.subplot(2,2,4)
sns.violinplot(x='Species',y='SepalWidthCm',data=df)
```

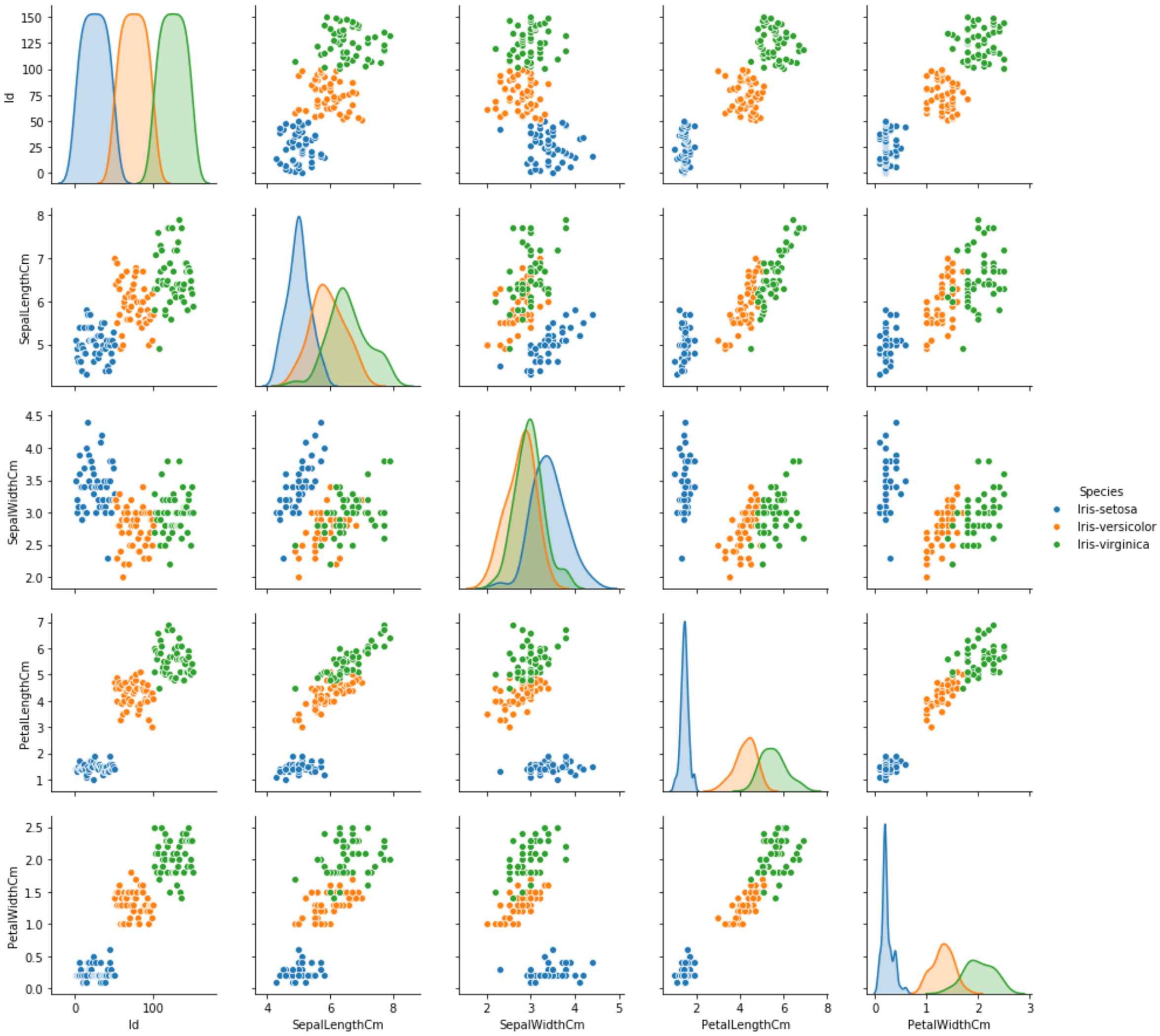
Out[59]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2398c49d688>
```



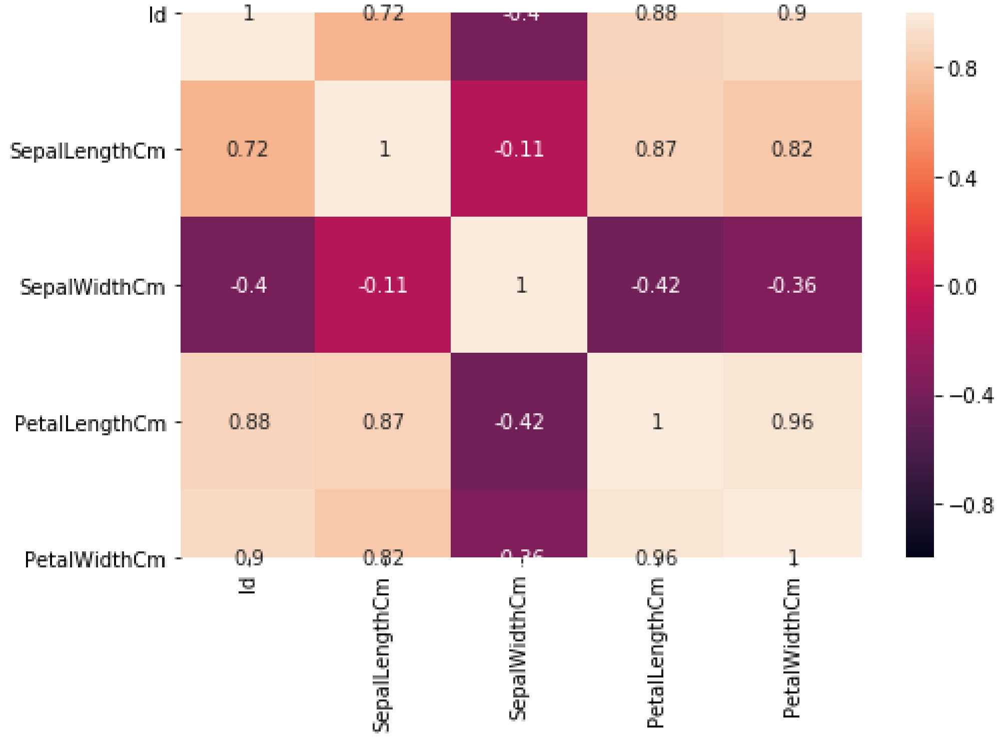
In [25]:

```
1  # Seaborn Plot
2  sns.pairplot(df,hue='Species');
```



In [75]:

```python
#Heat Maps
fig=plt.gcf()
fig.set_size_inches(8,5)
fig=sns.heatmap(df.corr(),annot=True,linewidths=0,linecolor='B',square=True, vmin=-1, 
```



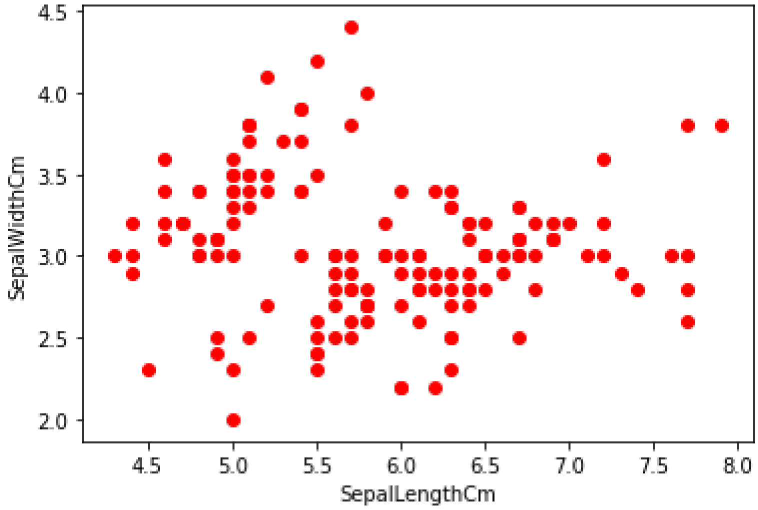
In [58]:

```python
X = df['SepalLengthCm'].values.reshape(-1,1)
Y = df['SepalWidthCm'].values.reshape(-1,1)
print(X)
print(Y)
```

```
[3.2]
[2.8]
[3. ]
[2.8]
[3. ]
[2.8]
[3.8]
[2.8]
[2.8]
[2.6]
[3. ]
[3.4]
[3.1]
[3. ]
[3.1]
[3.1]
[3.1]
[2.7]
[3.2]
```

In [30]:

```python
# Scatter Plot
plt.xlabel("SepalLengthCm")
plt.ylabel("SepalWidthCm")
plt.scatter(X,Y,color='R')
plt.show()
```



In [31]:

```python
#Correlation
cm = df.corr()
print(cm)
```

```
                     Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  \
Id             1.000000       0.716676     -0.397729       0.882747   
SepalLengthCm  0.716676       1.000000     -0.109369       0.871754   
SepalWidthCm  -0.397729      -0.109369      1.000000      -0.420516   
PetalLengthCm  0.882747       0.871754     -0.420516       1.000000   
PetalWidthCm   0.899759       0.817954     -0.356544       0.962757   

               PetalWidthCm  
Id                 0.899759  
SepalLengthCm      0.817954  
SepalWidthCm      -0.356544  
PetalLengthCm      0.962757  
PetalWidthCm       1.000000  
```

Model Building

In [32]:

```python
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn import metrics
from sklearn.tree import DecisionTreeClassifier
```

In [34]:

```
train, test = train_test_split(df, test_size = 0.25)
print(train.shape)
print(test.shape)
```

```
(112, 6)
(38, 6)
```

In [76]:

```
train_x = train[['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm']]
train_y = train.Species
test_x = test[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm','PetalWidthCm']]
test_y = test.Species
```

Logistic Regression

In [78]:

```
model1 = LogisticRegression()
model1.fit(train_x, train_y)
prediction = model.predict(test_x)
print('Accuracy:',metrics.accuracy_score(prediction,test_y))
```

```
Accuracy: 0.8947368421052632
```

In [79]:

```
#Confusion matrix
from sklearn.metrics import confusion_matrix,classification_report
confusion_matrix = confusion_matrix(test_y,prediction)
print("Confusion matrix\n",confusion_matrix)
```

```
Confusion matrix
 [[11  0  0]
 [ 0 11  3]
 [ 0  1 12]]
```

In [80]:

```
# Precision, recall, f1-score, accuracy
print(classification_report(test_y,prediction))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        11
Iris-versicolor       0.92      0.79      0.85        14
 Iris-virginica       0.80      0.92      0.86        13

       accuracy                           0.89        38
      macro avg       0.91      0.90      0.90        38
   weighted avg       0.90      0.89      0.89        38
```

KNeighborsClassifier

In [81]:

```
from sklearn.neighbors import KNeighborsClassifier
model2 = KNeighborsClassifier(n_neighbors=5)
model2.fit(train_x,train_y)
y_pred2 = model2.predict(test_x)
print("Accuracy:",accuracy_score(test_y,y_pred2))
```

```
Accuracy: 0.9473684210526315
```

Decision Tree

In [83]:

```
from sklearn.tree import DecisionTreeClassifier
model3 = DecisionTreeClassifier(criterion='entropy')
model3.fit(train_X,train_y)
y_pred3 = model3.predict(test_x)
print("Accuracy:",accuracy_score(test_y,y_pred3))
```

```
Accuracy: 0.8947368421052632
```

In [ ]:

```

```