Exercise 1

1.1 Problem Statement:

Perform Exploratory Data Analysis on any online dataset from Kaggle or UCI Machine Learning Repository.

1.2 Description of the Dataset:

Title of the dataset: Iris Plants Database

The Iris Dataset contains information of three species of Iris flowers (Iris setosa, Iris virginica and Iris versicolor). The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

Data Set Characteristics: Multivariate

Area: Life Sciences

Number of samples (or instances) in the dataset: 150

Number of attributes (or features): 05 Attribute

Information:

1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm
5. class:
   -- Iris Setosa
   -- Iris Versicolour
   -- Iris Virginica
6. Number of samples of each species of iris flowers:
   Class Distribution: 33.3% for each of 3 classes.
   50 (Setosa), 50 (Versicolor), 50 (Virginica)
7. Predicted attribute: class of iris plant.
8. Missing Attribute Values: None

Feature Name	Units of measurement	Datatype	Description
sepal length	Centimeters	Real (Numerical)	Length of Iris flower's sepal
sepal width length	Centimeters	Real (Numerical)	Width of Iris flower's sepal
petal length	Centimeters	Real (Numerical)	Length of Iris flower's petal
petal width length	Centimeters	Real (Numerical)	Width of Iris flower's petal
variety	Variety of species [Setosa, Virginica, Versicolor]	Object (Categorical)	Variety of the species of Iris flower

1.3 Data Preprocessing and Exploratory Data Analysis (EDA):

Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining as we cannot work with raw data. The quality of the data should be checked before applying machine learning or data mining algorithms.

Major Tasks in Data Preprocessing:

1. Data cleaning
2. Data integration
3. Data reduction
4. Data transformation

Exploratory data analysis (EDA) is a technique that data professionals can use to understand a dataset before they start to model it. Some people refer to EDA as data exploration. The goal of conducting EDA is to determine the characteristics of the dataset. Conducting EDA can help data analysts make predictions and assumptions about data. Often, EDA involves data visualization, including creating graphs like histograms, scatter plots and box plots.

Major Tasks in EDA:

1. Observe your dataset
2. Find any missing values
3. Categorize your values
4. Find the shape of your dataset
5. Identify relationships in your dataset
6. Locate any outliers in your dataset

1.4 Machine Learning Package Used for Model building:

The scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It provides simple and efficient tools for predictive data analysis. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The sklearn.model_selection.train_test_split() method splits arrays or matrices into random train and test subsets. The parameters for the method are as follows:

sklearn.model_selection.train_test_split(*arrays, test_size=None, train_size=None, random_state=Non e, shuffle=True, stratify=None)

It returns lists containing train-test split of inputs.

1.5 Implementation:

```
    #uploading the Iris dataset
from google.colab import files

uploaded = files.upload()
#importing required libraries
```

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
sns.set()
#loading the dataset
iris_data = pd.read_csv('Iris.csv')
iris_data
```

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

```
iris_data.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
```

```python
#Statistical Insights
iris_data.describe()
```

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

```
iris_data[iris_data.duplicated()]
```

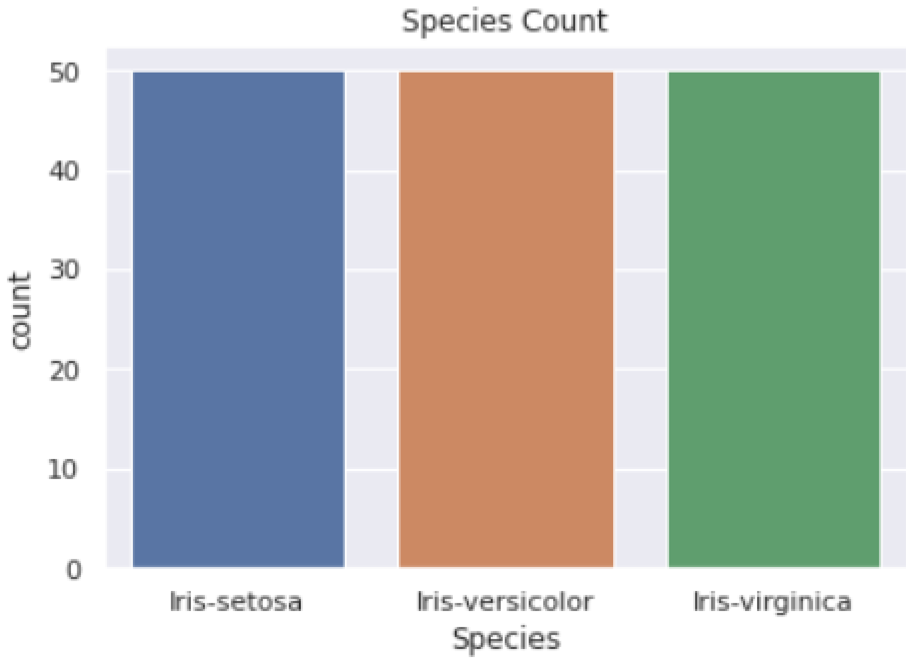
Id SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm Species

```
#checking the balance
iris_data['Species'].value_counts()
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

Data Visualisation

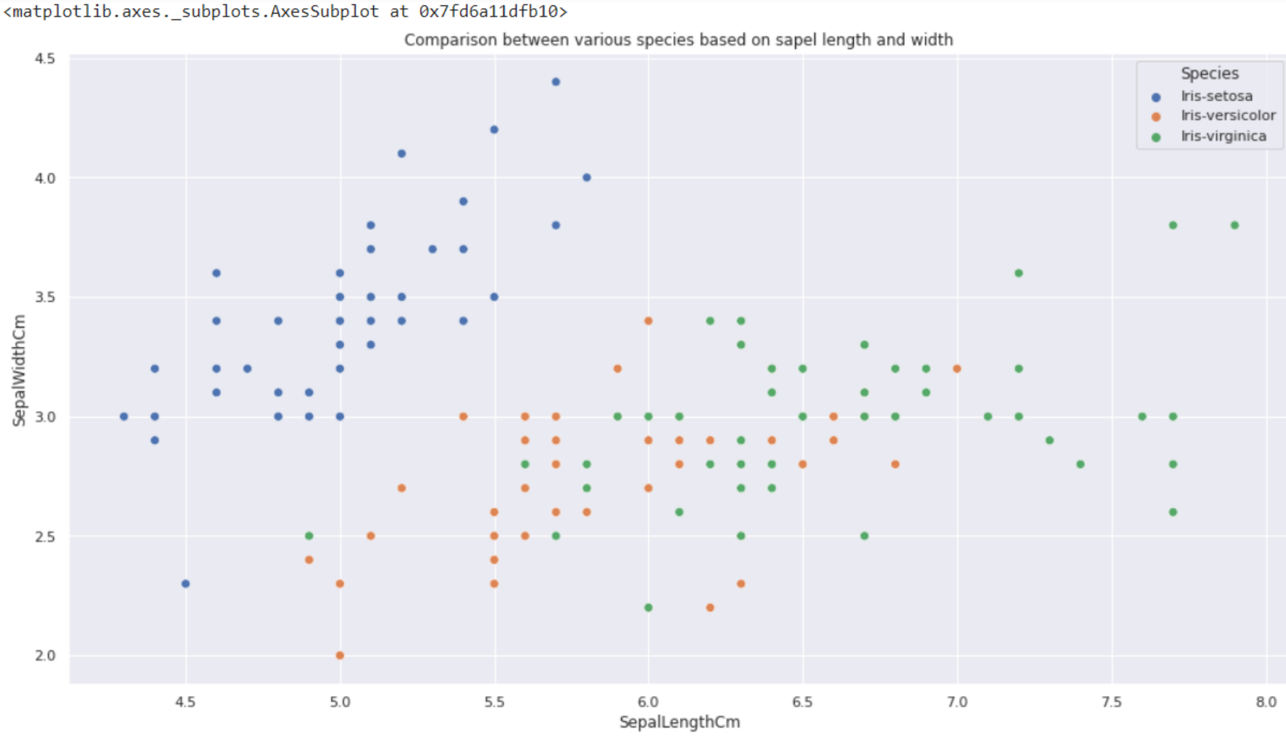
```
#species count

plt.title('Species Count')

sns.countplot(iris_data['Species'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.p
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fd6a1716b90>
```



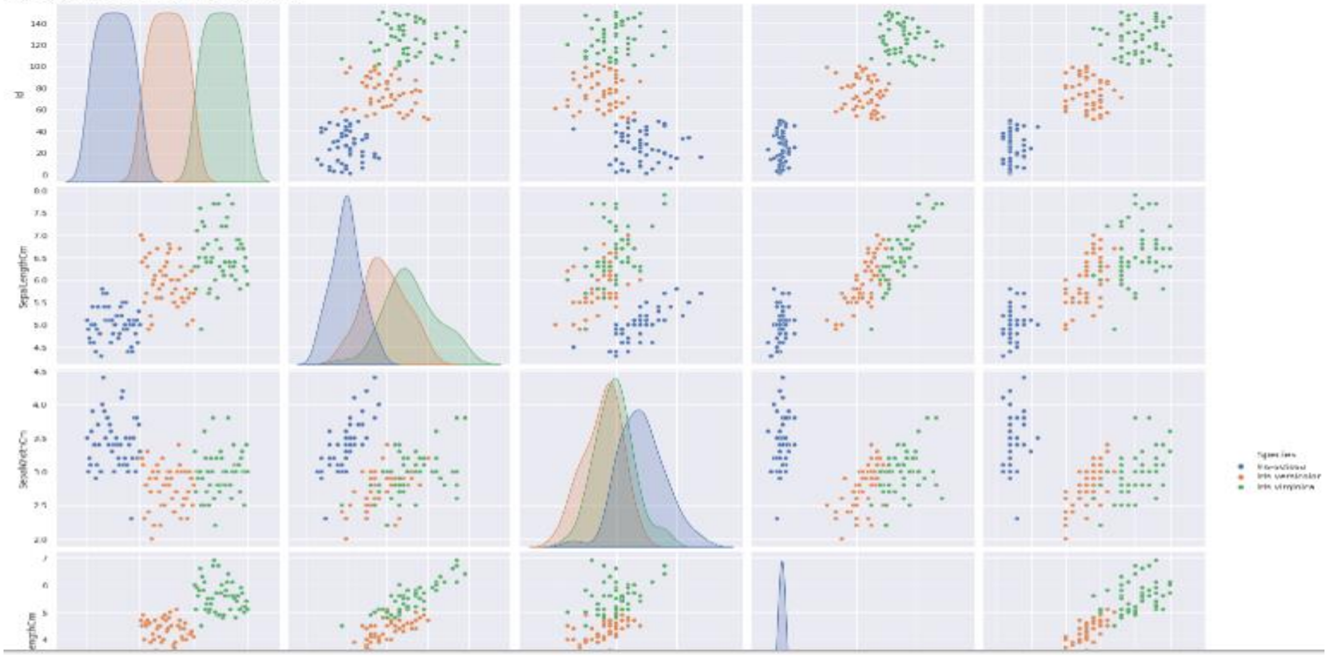

```
#Uni-variate Analysis
plt.figure(figsize=(17,9))
plt.title('Comparison between various species based on sapel length and width'
)
sns.scatterplot(iris_data['SepalLengthCm'],iris_data['SepalWidthCm'],hue =iris
_data['Species'],s=50)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd6a11dfb10>
```



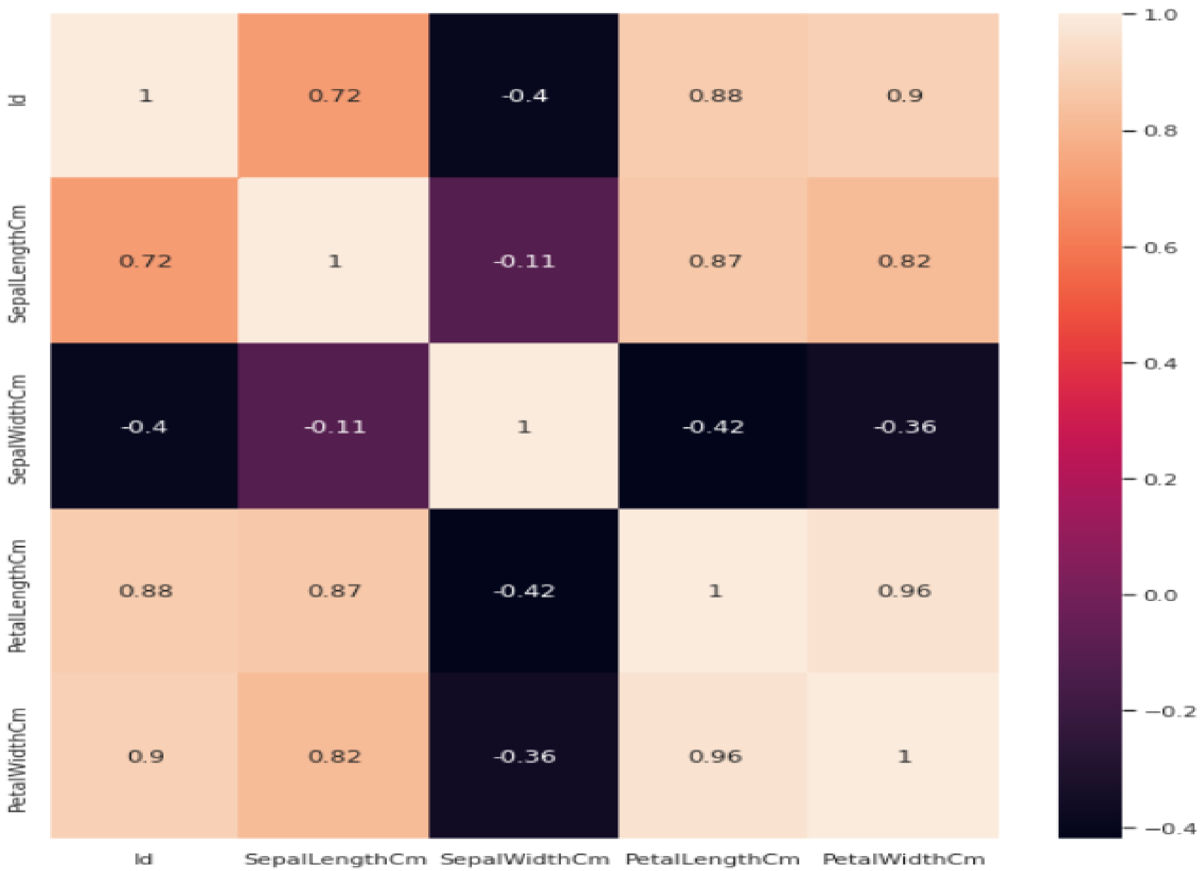
```
#Bi-variate Analysis
sns.pairplot(iris_data,hue='Species',height=4)
```

<seaborn.axisgrid.PairGrid at 0x7fd5a10bcc10>

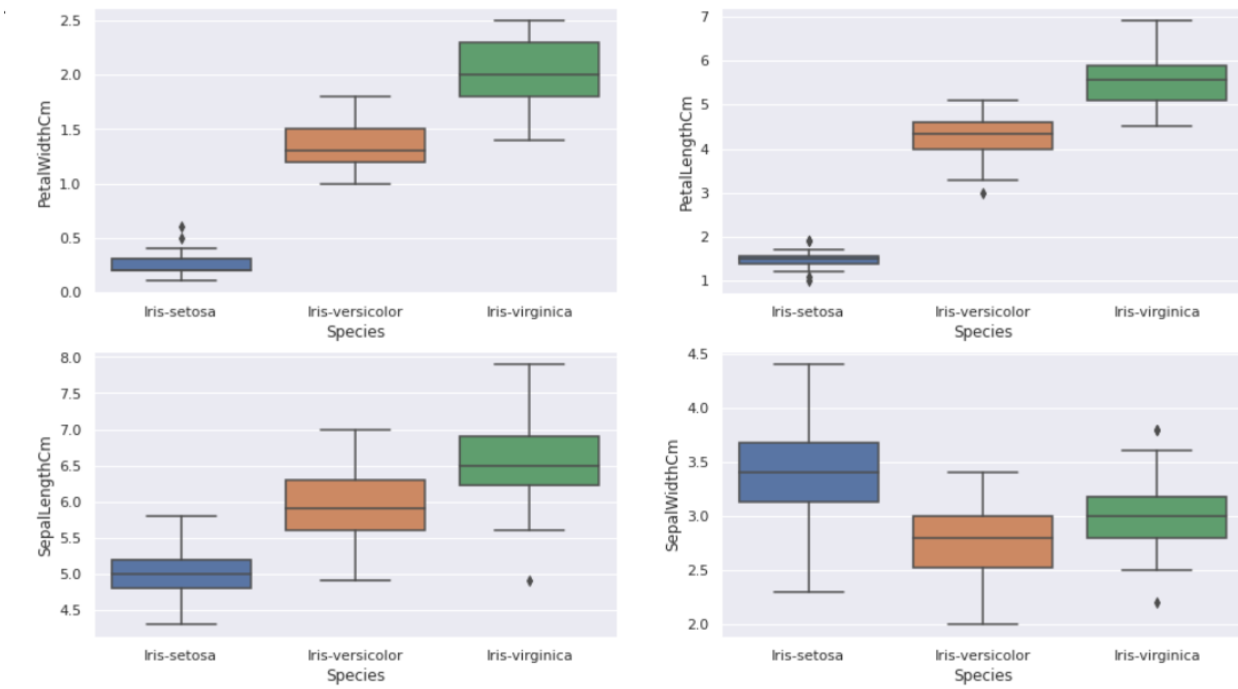


```
#Checking Correlation
plt.figure(figsize=(10,11))
sns.heatmap(iris_data.corr(),annot=True)
plt.plot()
```

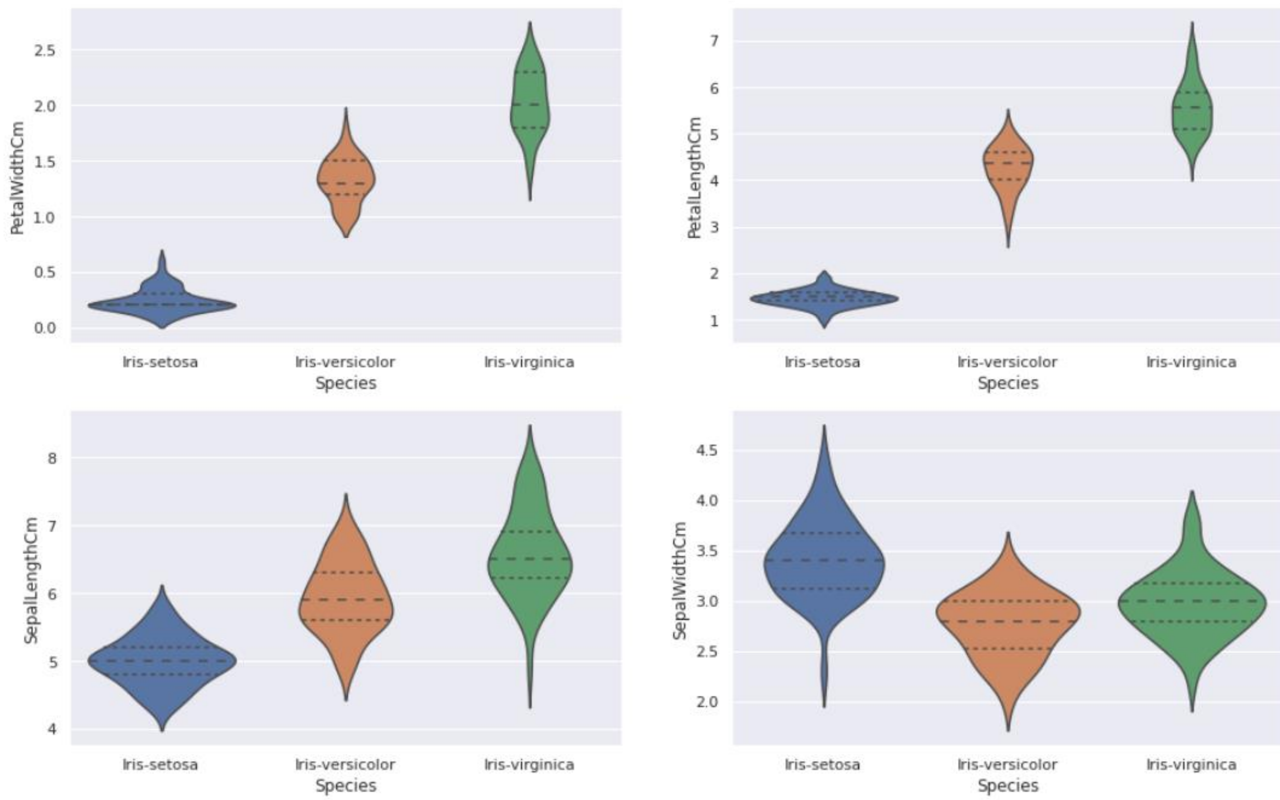
[]



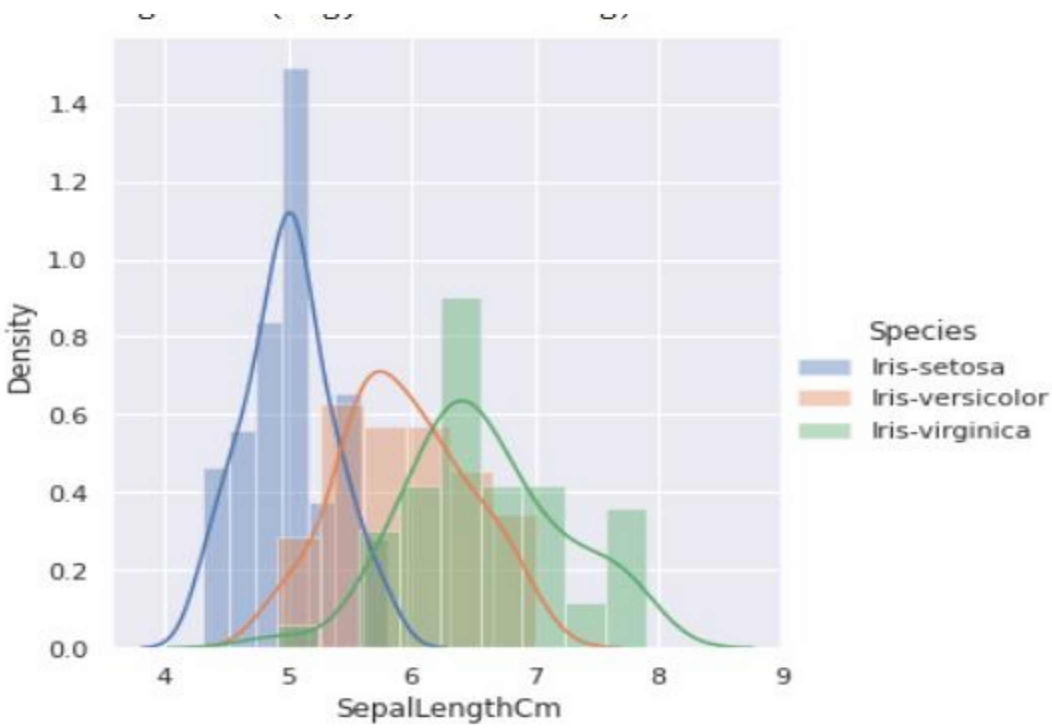
```
#Box plots
fig, axes = plt.subplots(2, 2, figsize=(16,9))
sns.boxplot( y='PetalWidthCm', x= 'Species', data=iris_data, orient='v' , ax=a
xes[0, 0])
sns.boxplot( y='PetalLengthCm', x= 'Species', data=iris_data, orient='v' , ax=
axes[0, 1])
sns.boxplot( y='SepalLengthCm', x= 'Species', data=iris_data, orient='v' , ax=
axes[1, 0])
sns.boxplot( y='SepalWidthCm', x= 'Species', data=iris_data, orient='v'  , ax=
axes[1, 1])
plt.show()
```

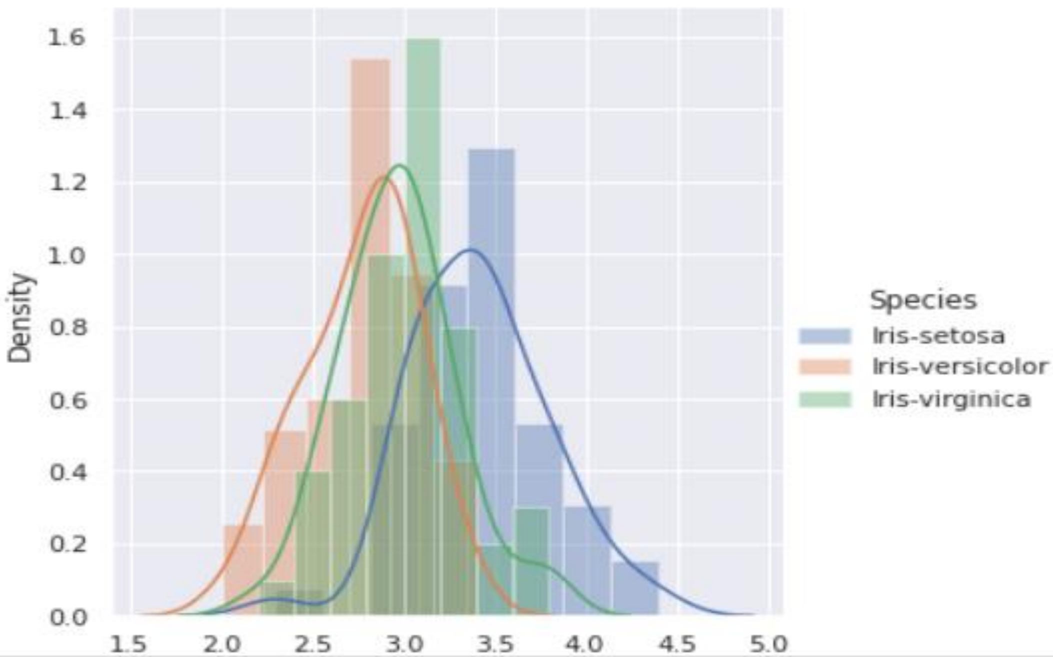



```
#Violin Plot
fig, axes = plt.subplots(2, 2, figsize=(16,10))
sns.violinplot(y='PetalWidthCm', x= 'Species', data=iris_data, orient='v' , ax
=axes[0, 0],inner='quartile')
sns.violinplot(y='PetalLengthCm', x= 'Species', data=iris_data, orient='v' , a
x=axes[0, 1],inner='quartile')
sns.violinplot( y='SepalLengthCm', x= 'Species', data=iris_data, orient='v' ,
ax=axes[1, 0],inner='quartile')
sns.violinplot( y='SepalWidthCm', x= 'Species', data=iris_data, orient='v'  ,
ax=axes[1, 1],inner='quartile')
plt.show()
```




```
#Plotting the Histogram & Probability Density Function (PDF)
sns.FacetGrid(iris_data, hue="Species", height=5).map(sns.distplot, "SepalLeng
thCm").add_legend()
sns.FacetGrid(iris_data, hue="Species", height=5).map(sns.distplot, "SepalWidt
hCm").add_legend()
```





1.6 Results and Discussion

The Perform of Exploratory Data Analysis on any online dataset from Kaggle or UCI Machine Learning Repository is done successfully by doing Uni variant Analysis, Bi variant Analysis, checking the Corelation and plotting different graphs of the data in the dataset.