# EXPLORING IRIS DATASET - IRIS_FLOWER_CLASSIFICATION

## INTRODUCTION

The Iris dataset is a classic and widely used dataset in the field of machine learning and data analysis. It was introduced by the British statistician and biologist Ronald Fisher in his 1936 paper "The use of multiple measurements in taxonomic problems" and has since become a standard dataset for practicing and benchmarking classification algorithms.

## ABOUT

The Iris dataset is popular in machine learning for several reasons:

- It is a small and well-understood dataset, making it ideal for beginners to practice classification algorithms.
- The dataset is balanced, meaning it contains an equal number of samples for each class, which helps prevent biases in model evaluation.
- The features are numeric and relatively simple to understand, making it easy to apply various machine learning techniques.
- Because of its simplicity and consistency, the Iris dataset is often used as a benchmark for comparing the performance of different machine learning algorithms.

Overall, the Iris dataset serves as an excellent starting point for learning and experimenting with machine learning algorithms, data visualization techniques, and exploratory data analysis.

## EXPLANATION COLUMNS

The Iris dataset consists of measurements of various characteristics of iris flowers belonging to three different species: Setosa, Versicolor, and Virginica. The four features measured for each flower are:

- Id
- Sepal length (in centimeters)
- Sepal width (in centimeters)
- Petal length (in centimeters)
- Petal width (in centimeters)
- Species

The dataset contains a total of 150 samples, with 50 samples for each of the three species. It is often used for tasks such as classification, clustering, and visualization.

## IMPORTING LIABRARIES

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


df = pd.read_csv('/content/Iris.csv')
df.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 |

Next steps: Generate code with df View recommended plots

## DIMENSION OF MY DATA

```
df.shape
```

```
(150, 6)
```

## DATATYPE OF MY COLUMNS

```
df.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
```

## CHECKING UNIQUE VALUES

```
df.nunique()
```

```
Id               150
SepalLengthCm     35
SepalWidthCm      23
PetalLengthCm     43
PetalWidthCm      22
Species            3
dtype: int64
```

## CHECKING NULL VALUES

```
df.isnull().sum()
```

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

## MATHEMATICAL OVERVIEW OF MY DATASET

```
df.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 |

```
df['Species'].unique()
```

```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

```
df_setosa = df.loc[df['Species']=='Iris-setosa']
df_versicolor = df.loc[df['Species']=='Iris-versicolor']
df_virginica = df.loc[df['Species']=='Iris-virginica']
df_setosa.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 |

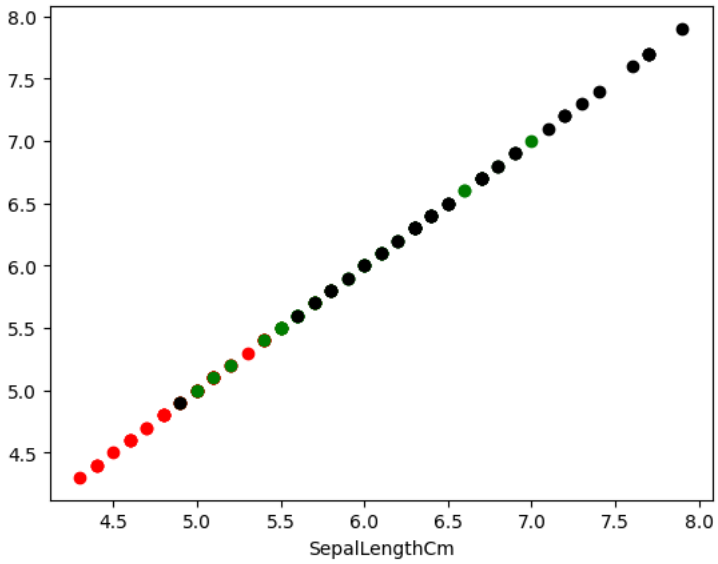
Next steps: Generate code with `df_setosa` View recommended plots

# QUISTIONS

## Line Plot:

- Plot a line graph showing the sepal length of the three species of Iris flowers.Use different colors to represent each species.

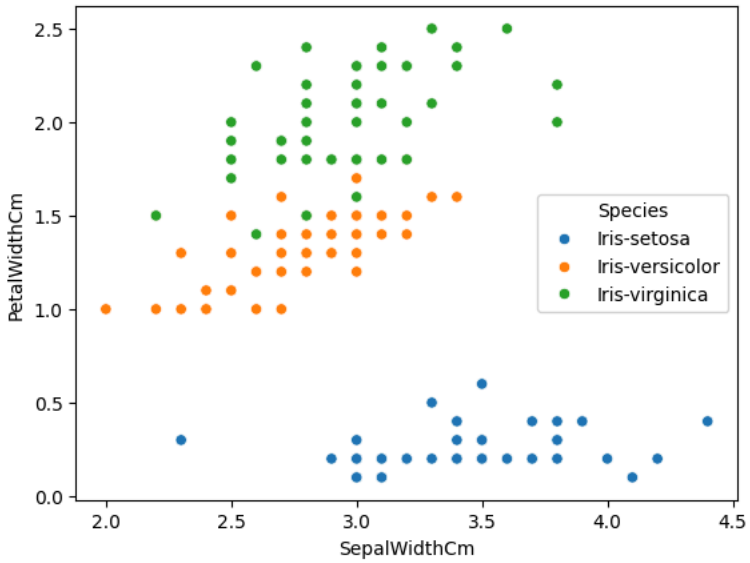
```
plt.plot(df_setosa['SepalLengthCm'],df_setosa['SepalLengthCm'],'o r')
plt.plot(df_versicolor['SepalLengthCm'],df_versicolor['SepalLengthCm'],'o g')
plt.plot(df_virginica['SepalLengthCm'],df_virginica['SepalLengthCm'],'o k')
plt.xlabel("SepalLengthCm")
plt.show()
```




## Scatter Plot -

- Create a scatter plot showing the relationship between the petal length and petal width.Use different colors to represent each species.

```
sns.scatterplot(x='SepalWidthCm',y='PetalWidthCm',data=df,hue='Species')
plt.show()
```

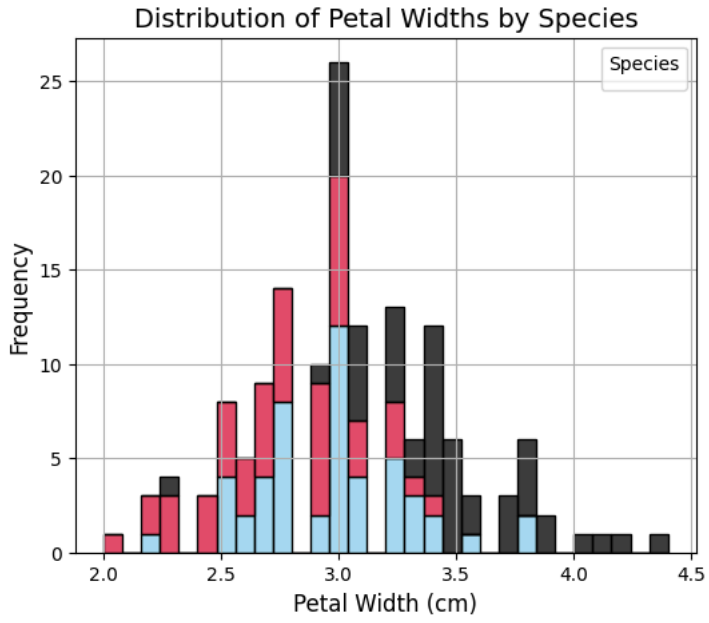



## Histogram -

- Plot a histogram showing the distribution of the petal widths. Use different colors to represent each species and display a legend.

```
color_map = {
    'Iris-setosa': 'k',
    'Iris-versicolor': 'crimson',
    'Iris-virginica': 'skyblue'
}

# Plot histogram using seaborn
plt.figure(figsize=(6, 5))
sns.histplot(data=df, x='SepalWidthCm', hue='Species', bins=30, multiple='stack', palette=color_map)

plt.xlabel('Petal Width (cm)', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.title('Distribution of Petal Widths by Species', fontsize=14)
plt.legend(title='Species')

plt.grid(True)
plt.show()
```

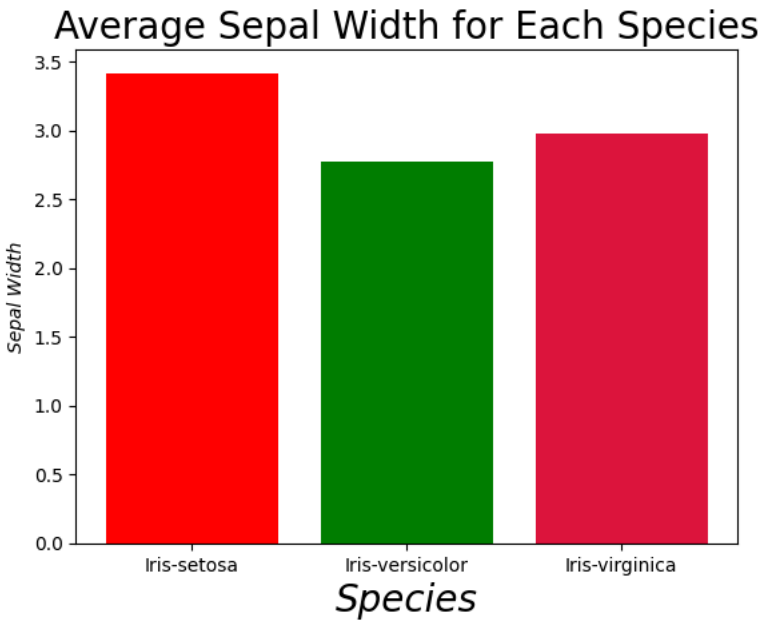
```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that a
```




## Bar Plot -

- Create a bar plot showing the average sepal width for each species

```
color = ['red', 'green', 'crimson']
species = df['Species'].unique()
sepal_avg = df.groupby('Species')['SepalWidthCm'].mean()

plt.bar(species, sepal_avg, color=color)
plt.xlabel('Species', fontsize=20, style='oblique')
plt.ylabel('Sepal Width', fontsize=10, style='oblique')
plt.title('Average Sepal Width for Each Species', fontsize=20)
plt.show()
```

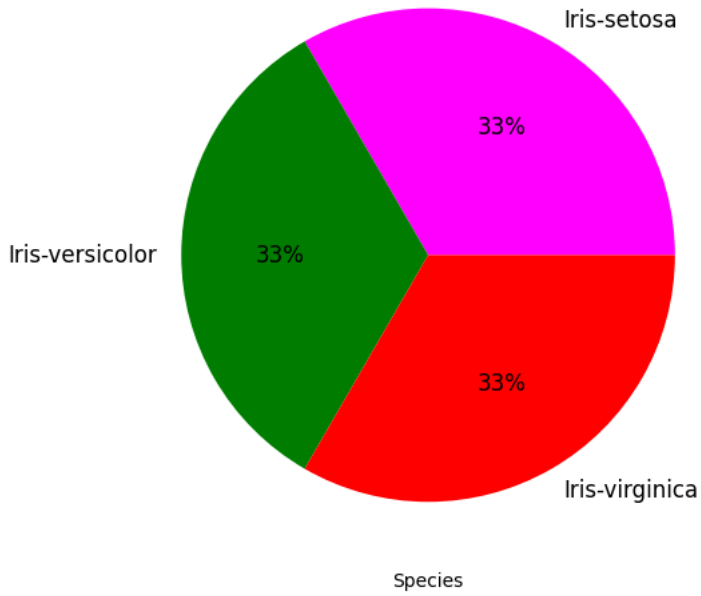



## Pie Chart -

- Assume that the dataset represents the entire population of Iris flowers. Create a pie chart showing the proportion of each species in the population.

```
spe_co = df['Species'].value_counts()
colors = ['magenta', 'green', 'red']
spe_co.plot(kind ='pie',figsize = (10,6),colors = colors,autopct='%1.f%%',fontsize=12,)
plt.ylabel(' ')
plt.xlabel('Species')
```

```
Text(0.5, 0, 'Species')
```

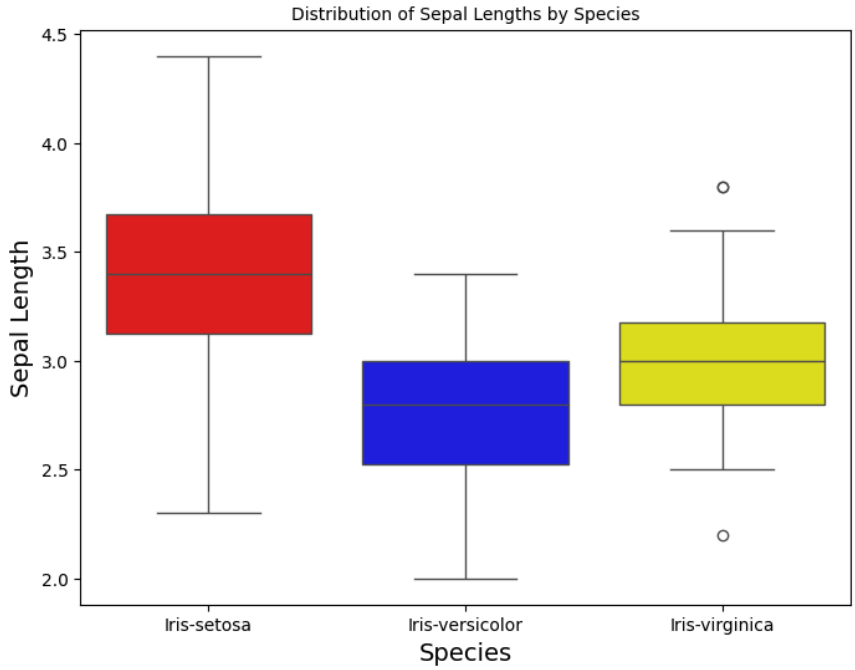


## Box Plot -

- Create a box plot showing the distribution of sepal length for each species.

```
color = ['red', 'blue', 'yellow']
spe_l = df['Species'].unique()
avg_sepal_length = df.groupby('Species')['SepalWidthCm'].sum()
plt.figure(figsize=(8, 6))
sns.boxplot(data=df, x='Species', y='SepalWidthCm', palette=color)

plt.xlabel('Species', fontsize=14)
plt.ylabel('Sepal Length', fontsize=14)
plt.title('Distribution of Sepal Lengths by Species', fontsize=10)
plt.show()
```

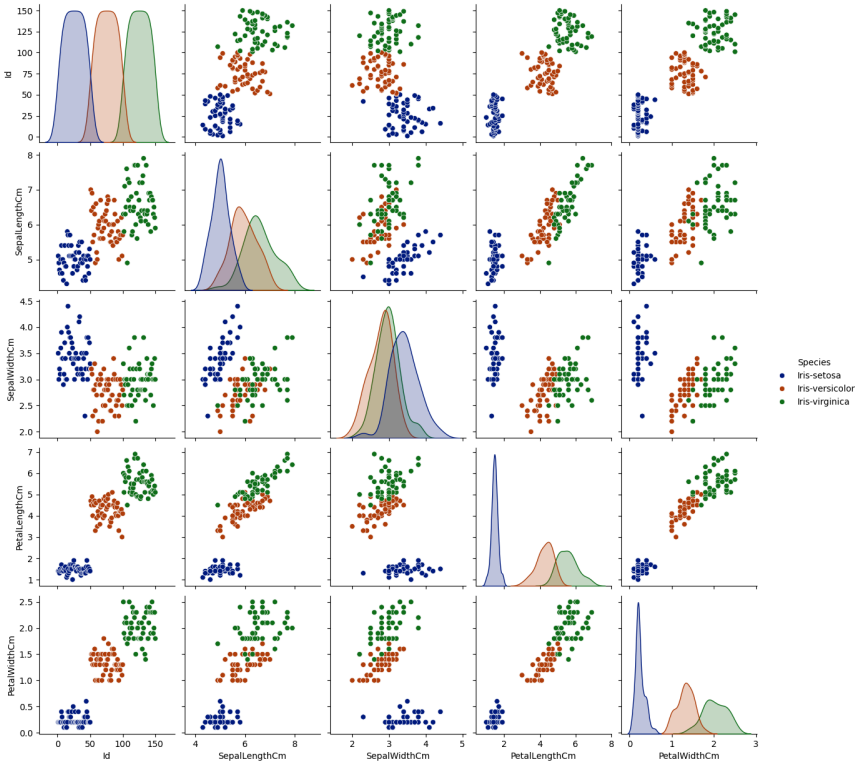
```
<ipython-input-73-b200acaa6e89>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14

  sns.boxplot(data=df, x='Species', y='SepalWidthCm', palette=color)
```




## Pairplots -

- Combine the scatter plot and the histogram into one figure by using pairplots

```
sns.pairplot(df, hue='Species', palette='dark')
plt.show()
```




Start coding or generate with AI.