

## Import Libraries we want to use

```
[2]: import pandas as pd # for data operations
import matplotlib.pyplot as plt # for plotting
import numpy as np

[3]: import seaborn as sns # for data visualization .. Seaborn is a powerful Python data visualization library
# built on top of Matplotlib. It provides high-level functions for creating attractive and informative statistical graphics.
```

## Database loading

```
[5]: # Load the iris dataset .. in same folder
dataset = pd.read_csv("iris.csv") # Load dataset into a pandas DataFrame.
dataset.head() # to ensure read it successfully
```

	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

```
[6]: # Renaming the columns
dataset.columns = ['id','sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species']

dataset.head() # to show data
```

	id	sepal_length	sepal_width	petal_length	petal_width	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

## Exploratory Data Analysis (EDA) of iris dataset

### Data Exploration - Information

```
[9]: dataset.describe() # describing columns in dataset .. provides a summary of each feature, including mean, standard deviation, and quartiles.
```

	id	sepal_length	sepal_width	petal_length	petal_width
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

```
[10]: dataset.shape
```

```
[10]: (150, 6)
```

```
[11]: dataset['species'].value_counts()
```

```
[11]: species
Iris-setosa      50
Iris-versicolor  50
Iris-virginica   50
Name: count, dtype: int64
```

### Checking for missing and Duplicate values

```
[13]: dataset.isnull().sum() #sum of null values
```

```
[13]: id          0
sepal_length  0
sepal_width   0
petal_length   0
petal_width   0
species       0
dtype: int64
```

```
[14]: dataset.duplicated().sum() # sum of duplicated values
```

```
[14]: 0
```

```
[15]: dataset = dataset.drop(columns=['id']) #not important to Explorations Data
```

### Data Visualizations

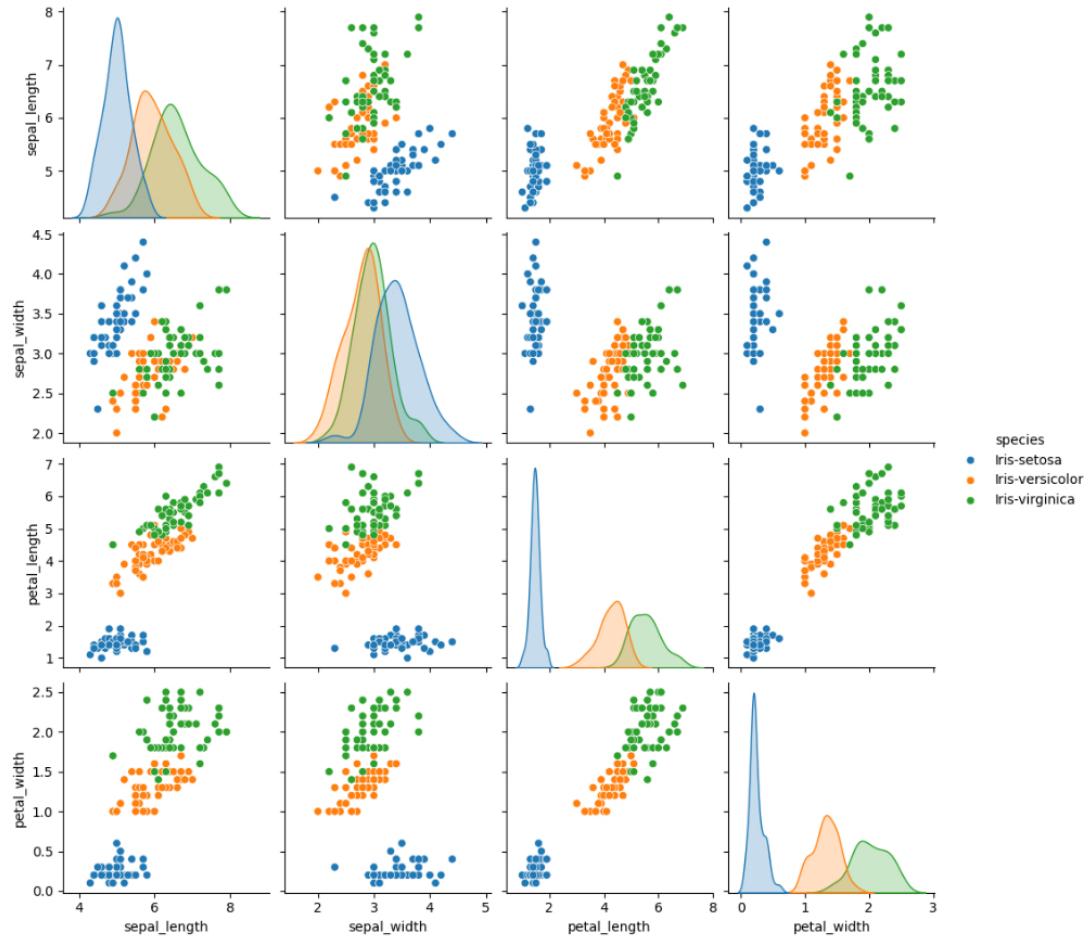
```
[17]: # Visual Analysis - Pair Plots
```

```
sns.pairplot(dataset, hue='species')
# The pairplot() function from Seaborn creates grid of scatter plots for pairwise relationships between variables.
```

```
# It takes dataframe as input.
```

```
# It plots each numeric variable against every other numeric variable in dataframe.
```

```
plt.show()
```

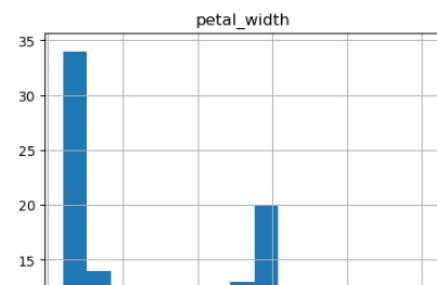
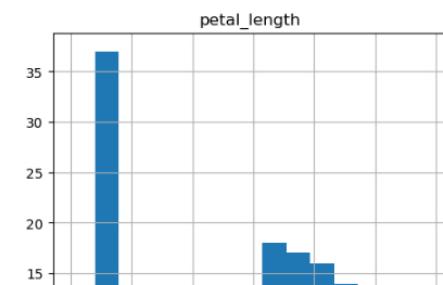
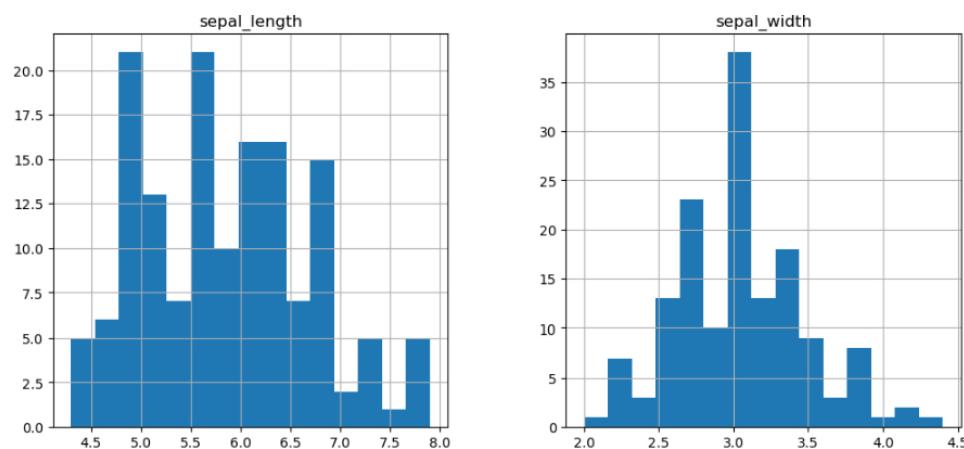


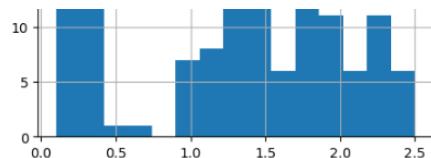
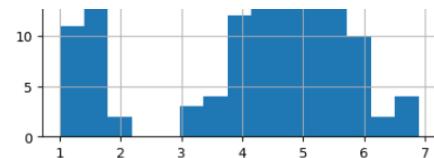
from pairplot can extract - Iris-species Setosa has the smallest of petals widths and lengths.

It also has the smallest sepal length but larger sepal widths. and some other info

```
[19]: # Visual Analysis - Histograms  
  
# Create histograms for the specified columns  
dataset[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']].hist(figsize=(12, 12), bins=15) # Creating histograms  
plt.suptitle('Histograms of Sepal and Petal Dimensions', size=12) #title to the histograms  
plt.show() # Display the histograms
```

Histograms of Sepal and Petal Dimensions

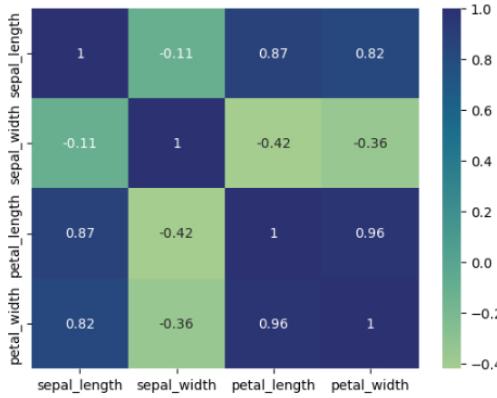




```
[20]: # Correlation Analysis
# Correlation between species - Heat Maps
corr = dataset[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']].corr()
corr
```

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.109369	0.871754	0.817954
sepal_width	-0.109369	1.000000	-0.420516	-0.356544
petal_length	0.871754	-0.420516	1.000000	0.962757
petal_width	0.817954	-0.356544	0.962757	1.000000

```
[21]: sns.heatmap(corr, annot=True, cmap='crest') #shows a correlation between all numerical variables in the dataset.
plt.show()
```



## So Result Of EDA

- The iris dataset contains 150 samples with 5 basic columns: 'sepal\_length', 'sepal\_width', 'petal\_length', 'petal\_width', and 'species'
- There are no missing values or duplicate values
- The dataset is balanced across the three species: iris-setosa, irisversicolor, and iris-virginica

## Visual Analysis

- Histograms : Show the distribution of each feature [sepal\_length and petal\_length] have a wider range compared to sepal\_width and petal\_width
- Pair Plot : Highlights the relationships between features. petal\_length and petal\_width show clear separation between species

## Correlation Analysis

- Heat Maps to visualize correlation between features

## Hypothesis Testing as an Exploratory Data Analysis Technique

```
[24]: from scipy import stats
# used in Hypothesis Testing to execute for example t-tests, chi-square tests, ANOVA, and more.
```

### One Sample T-Test

Null Hypothesis (H0)

$H_0: \mu = 5.1$

The sample mean sepal\_length is equal to the 5.1 =>  $\text{sepal\_length}=5.1$

Alternative Hypothesis (Ha or H1)

$H_1: \mu \neq 5.1$

The sample mean sepal\_length is not equal to the 5.1 "complement of H0" =>  $\text{sepal\_length} \neq 5.1$

```
[26]: # sample of sepal_lengths from the Iris dataset
sample = dataset[dataset['species'] == "Iris-setosa"]["sepal_length"] # Hypothes here that all Iris-setosa specie lenght = 5.2

hypothetical_value = 5.1 # Hypothetical mean
# to test whether the mean sepal_length of the Iris-setosa species is different from this hypothetical value

# Calculate sample statistics
sample_mean = np.mean(sample)
sample_std = np.std(sample, ddof=1)
sample_size = len(sample)
# Calculate the t_statistic using equation  $t_{\text{statistic}} = \bar{x} - \mu_0 / (s/\sqrt{n})$ 
t_statistic_manual = (sample_mean - hypothetical_value) / (sample_std / np.sqrt(sample_size))
# Calculate the p-value using equation  $p_{\text{value}} = 2 \times \text{CDF}(t, df)$ 
p_value_manual = 2 * stats.t.cdf(t_statistic_manual, sample_size - 1)

# Output results
print("Sample Mean:", sample_mean)
print("Sample Standard Deviation:", sample_std)
print("Sample Size:", sample_size)
print("t statistic:", t_statistic_manual)
print("p-value:", p_value_manual)

Sample Mean: 5.006
Sample Standard Deviation: 0.3524896872134512
Sample Size: 50
```

```

T statistic: -1.885673250669746
P-value: 0.06527445885090737

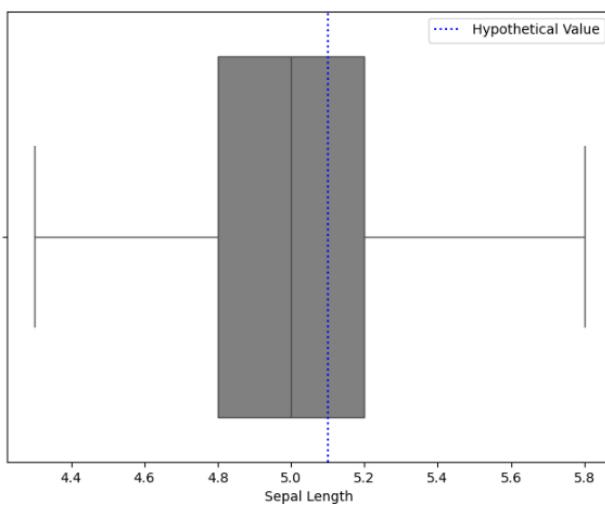
[27]: # t-test on sample directly using ttest_1samp on scipy library
t_statistic, p_value = stats.ttest_1samp(sample, hypothetical_value)

# ttest_1samp(sample, popmean) Calculate the T-test for the mean of sample from all data
# test for the null hypothesis that the expected value /mean of a sample of independent observations a is equal to the given population mean
# t_statistic: difference between the sample mean and the hypothetical mean relative to the variation in the sample
# p_value: probability of obtaining the observed data assuming the null hypothesis is true
print("T-statistic:", t_statistic)
print("P-value:", p_value)

T-statistic: -1.885673250669745
P-value: 0.06527445885090742

[28]: # Create a box plot
plt.figure(figsize=(8, 6)) # figure with a specified size
sns.boxplot(x=sample, color="gray") # box plot of the sample_data
plt.axvline(hypothetical_value, color="blue", linestyle="dotted", label="Hypothetical Value") # add vertical line at the specified_value
plt.legend()
plt.xlabel("Sepal Length") # Label for the x-axis
plt.show()

```



```

[29]: # Define the significance Level -- the probability of rejecting the null hypothesis when it the null hypothesis is true
significance_level = 0.05

# Compare the p-value to the significance level
if p_value < significance_level:
    print("Reject the null hypothesis  $\mu = 5.1$  , Significant difference in sepal length")
else:
    print("Accept the null hypothesis  $\mu = 5.1$  , No significant difference in sepal length")

Accept the null hypothesis  $\mu = 5.1$  , No significant difference in sepal length

```

### One - Sample T-Test "left tail test"

Null Hypothesis (H0)

$$H0: \mu \geq 2.5$$

The sample mean sepal\_width is greater than or equal to the 2.5 => sepal\_width  $\geq 2.5$

Alternative Hypothesis (Ha or H1)

$$H1: \mu < 2.5$$

The sample mean sepal\_width is less than 2.5 "complement of H0" => sepal\_width  $< 2.5$

```

[31]: # sepal width data from Iris-virginica species
sample = dataset[dataset["species"] == "Iris-virginica"]["sepal_width"]

hypothetical_value = 2.5 # Hypothetical mean

```

```

[32]: # t-test sample
t_statistic, p_value = stats.ttest_1samp(sample, hypothetical_value)

# p-value for a one-sided test "left tail test"
if t_statistic < 0:
    p_value /= 2 # Divide the p-value by 2 for a one-sided test

p_value, t_statistic

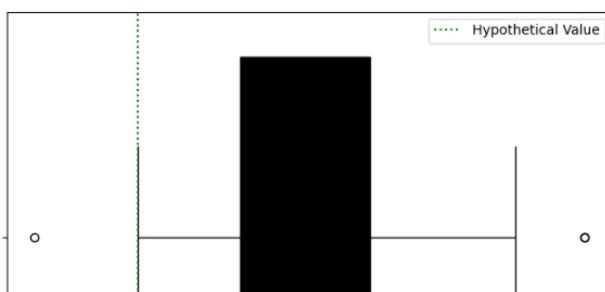
```

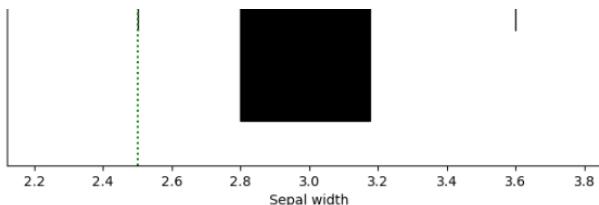
```
[32]: (5.5137715014737266e-14, 10.392933587822478)
```

```

[33]: # Create a box plot
plt.figure(figsize=(8, 6))
sns.boxplot(x=sample, color="black")
plt.axvline(hypothetical_value, color="green", linestyle="dotted", label="Hypothetical Value")
plt.legend()
plt.xlabel("Sepal width")
plt.show()

```





```
[34]: # Define the significance Level -- the probability of rejecting the null hypothesis when it the null hypothesis is true
significance_level = 0.05

# Compare the p-value to the significance level
if p_value < significance_level:
    print("Reject the null hypothesis  $\mu \geq 2.5$  , Significant difference in sepal width.")
else:
    print("Accept the null hypothesis  $\mu \geq 2.5$  , No significant difference in sepal width.")

Reject the null hypothesis  $\mu \geq 2.5$  , Significant difference in sepal width.
```

## Two-Sample T-Test

Null Hypothesis (H0)

$$H0: \mu_1 = \mu_2$$

The means of the two groups['Iris-setosa', 'Iris-versicolor'] are equal

Alternative Hypothesis (Ha or H1)

$$H1: \mu_1 \neq \mu_2$$

The means of the two groups['Iris-setosa', 'Iris-versicolor'] are not equal

```
[36]: # Samples for Setosa and Versicolor species
setosa_sample = dataset[dataset['species'] == 'Iris-setosa']['petal_length']
versicolor_sample = dataset[dataset['species'] == 'Iris-versicolor']['petal_length']

#using Equation t= ((x1-x2)/(sp √(1/n1 +1/n2 ))
# Step 1: Calculate the means
setosa_mean = np.mean(setosa_sample)
versicolor_mean = np.mean(versicolor_sample)

# Step 2: Compute the standard deviations (use ddof=1 for sample standard deviation)
std_setosa = np.std(setosa_sample, ddof=1)
std_versicolor = np.std(versicolor_sample, ddof=1)

# Step 3: Calculate the pooled standard deviation
setosa_len = len(setosa_sample)
versicolor_len = len(versicolor_sample)

# sp=√((n1-1)s1^2+(n2-1)s2^2)/(n1+n2-2)
std = np.sqrt(((setosa_len - 1) * std_setosa**2 + (versicolor_len - 1) * std_versicolor**2) / (setosa_len + versicolor_len - 2))

# Step 4: Compute the t-statistic
t_statistic = (setosa_mean - versicolor_mean) / (std * np.sqrt(1/setosa_len + 1/versicolor_len))

# Step 5: calculate the p-value
df = setosa_len + versicolor_len - 2
p_value = 2 * stats.t.sf(np.abs(t_statistic), df)

# results
print("T-Statistic:", t_statistic)
print("P-Value:", p_value)

# Step 6: Compare the p-value to the significance level
significance_level = 0.05

if p_value < significance_level:
    print("Reject the null hypothesis  $\mu_1 = \mu_2$  , Significant difference in petal lengths.")
else:
    print("Accept the null hypothesis  $\mu_1 = \mu_2$  , No significant difference in petal lengths.)
```

T-Statistic: -39.4686625939727  
P-Value: 5.717463758170784e-62  
Reject the null hypothesis  $\mu_1 = \mu_2$  , Significant difference in petal lengths.

```
[37]: # samples for setosa and versicolor species
setosa_sample = dataset[dataset['species'] == 'Iris-setosa']['petal_length']
versicolor_sample = dataset[dataset['species'] == 'Iris-versicolor']['petal_length']

# Perform two-sample t-test
t_statistic, p_value = stats.ttest_ind(setosa_sample, versicolor_sample)
print("T-statistic:", t_statistic)
print("P-value:", p_value)

# Define the significance Level -- the probability of rejecting the null hypothesis when it the null hypothesis is true
significance_level = 0.05

# Compare the p-value to the significance level
if p_value < significance_level:
    print("Reject the null hypothesis  $\mu_1 = \mu_2$  , Significant difference in petal lengths.")
else:
    print("Accept the null hypothesis  $\mu_1 = \mu_2$  , No difference in petal lengths.)
```

## ANOVA TEST

Null Hypothesis (H0)

$$H0: \mu_{\text{Setosa}} = \mu_{\text{Versicolor}} = \mu_{\text{Virginica}}$$

There is no significant difference in mean petal lengths among the three species.

Alternative Hypothesis (Ha or H1)

$$H1: \mu_{\text{Setosa}} \neq \mu_{\text{Versicolor}} \neq \mu_{\text{Virginica}}$$

There is a significant difference in mean petal lengths among the three species.

```
[2]: anova_result = stats.t_oneway(
    dataset[dataset['species'] == 'Iris-setosa']['petal_length'],
    dataset[dataset['species'] == 'Iris-versicolor']['petal_length'],
    dataset[dataset['species'] == 'Iris-virginica']['petal_length']
)

print("T-statistic:", anova_result.statistic)
print("P-value:", anova_result.pvalue)

# Define the significance level -- the probability of rejecting the null hypothesis when it the null hypothesis is true
significance_level = 0.05

# Compare the p-value to the significance Level
if p_value < significance_level:
    print("Reject null hypothesis  $\mu$  Setosa =  $\mu$  Versicolor =  $\mu$  Virginica : at least one group mean is different.")
else:
    print("Accept the null hypothesis  $\mu$  Setosa =  $\mu$  Versicolor =  $\mu$  Virginica: all group means are the same")

T-statistic: 1179.0343277002194
P-value: 3.0519758018278374e-91
Reject null hypothesis  $\mu$  Setosa =  $\mu$  Versicolor =  $\mu$  Virginica : at least one group mean is different.
```

[ ]: