
Group A
Assignment No: 10

Title of the Assignment: Data Visualization III

Download the Iris flower dataset or any other dataset into a DataFrame. (e.g., https://archive.ics.uci.edu/ml/datasets/Iris). Scan the dataset and give the inference as:

1. List down the features and their types (e.g., numeric, nominal) available in the dataset.
2. Create a histogram for each feature in the dataset to illustrate the feature distributions.
3. Create a box plot for each feature in the dataset.
4. Compare distributions and identify outliers.

Objective of the Assignment: Students should be able to perform the data Visualization
operation using Python on any open source dataset

Prerequisite:

1. Basic of Python Programming
2. Seaborn Library, Concept of Data Visualization.
3. Types of variables

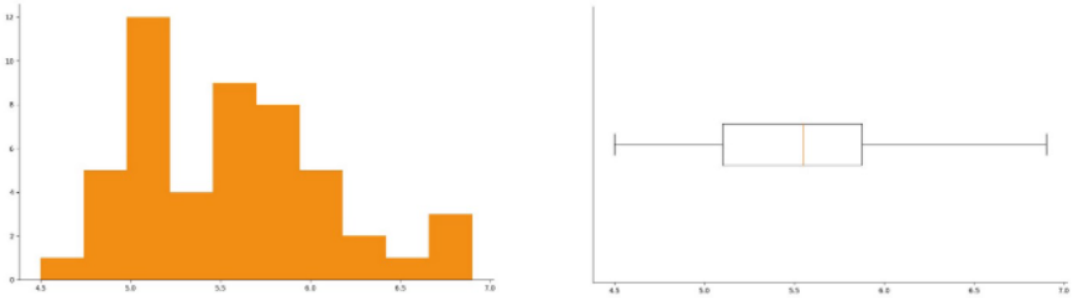
Theory:-

Boxplots are underrated. They are jam-packed with insights about the underlying distribution, because they condense lots of information about your data into a small visualization.

In this article you see how Boxplots are great tools to:

- Understand the spread of the data.
- Spot outliers.
- Compare distributions, and how small tweaks in the boxplot visualization make it easier spot differences between distributions.

- **Understanding the spread of the data**
- During exploratory data analysis, boxplots can be a great complement to histograms.
- With histograms it's easy to see the shape and trends in a distribution. Because histograms highlight how frequently each data point occurs in the distribution.
- Boxplots don't focus directly on frequency, but instead on the range of values in the distribution.

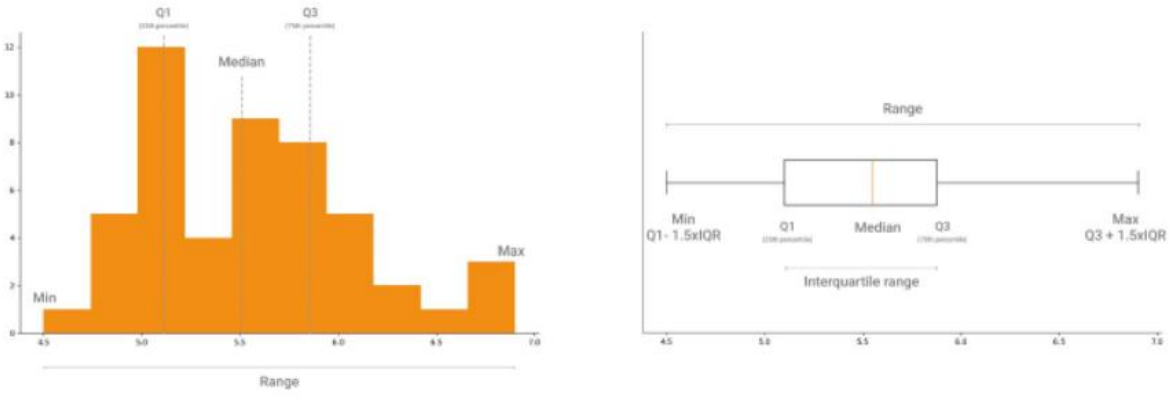


Histograms highlight frequency while boxplots highlight the range of the data.

We are used to think in terms of frequency and comparing proportions. That's why we're so comfortable interpreting the insights of an histogram, where we can spot the values where most data is concentrated around, and we can see the shape of the distribution.

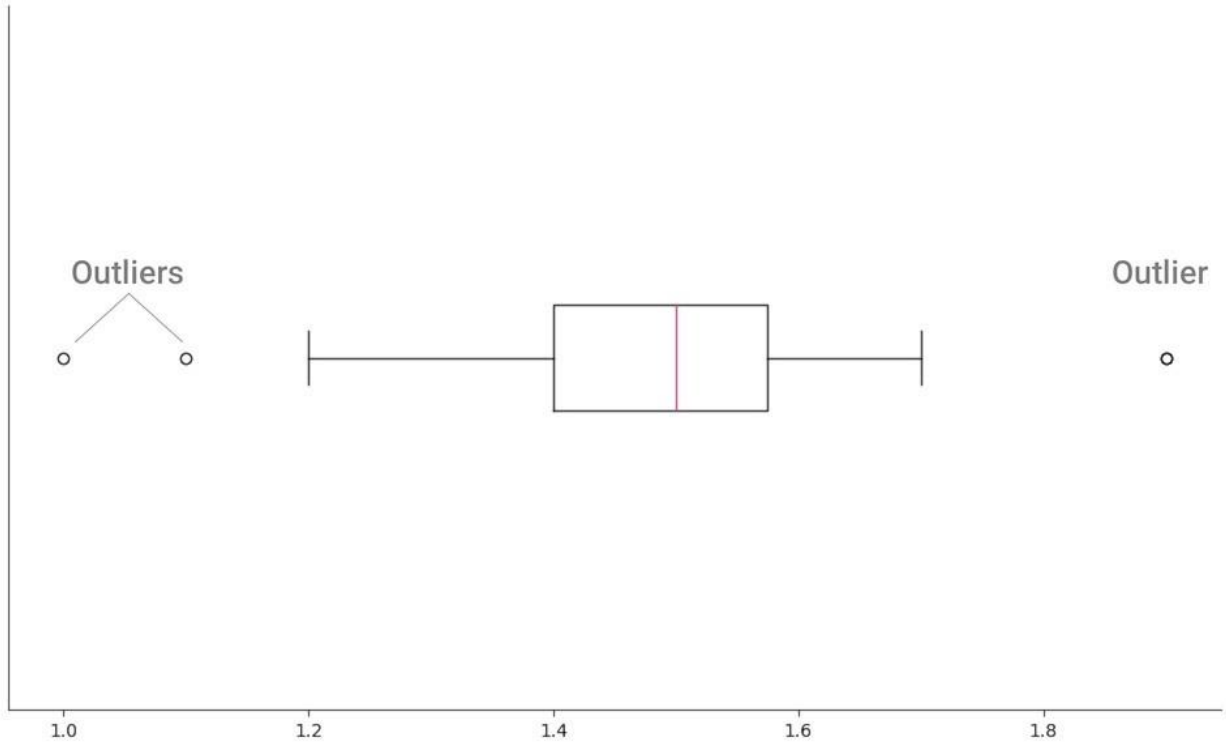
With a boxplot, we can extract the same insights as with an histogram. And while we can visualize the shape of the distribution with an histogram, a boxplot highlights the summary metrics that give the distribution its shape. The summary metrics we can extract from a boxplot are:

- *Quantiles*, specifically the first and third quantiles, which correspond to the 25th and 75th percentiles.
- *Median*, the mid-point in the distribution, which also corresponds to the 50th percentile.
- *Interquartile range (IQR)*, the width between the third and first quantiles. Expressed mathematically, we have IQR = Q3 — Q1.
- *Min,* minimum value in the dataset excluding outliers, which corresponds to Q1–1.5xIQR.
- *Max*, maximum value in the dataset, excluding outliers, which corresponds to Q3+1.5xIQR.



Summary metrics you can extract from an histogram and a boxplot.

Spot outliers



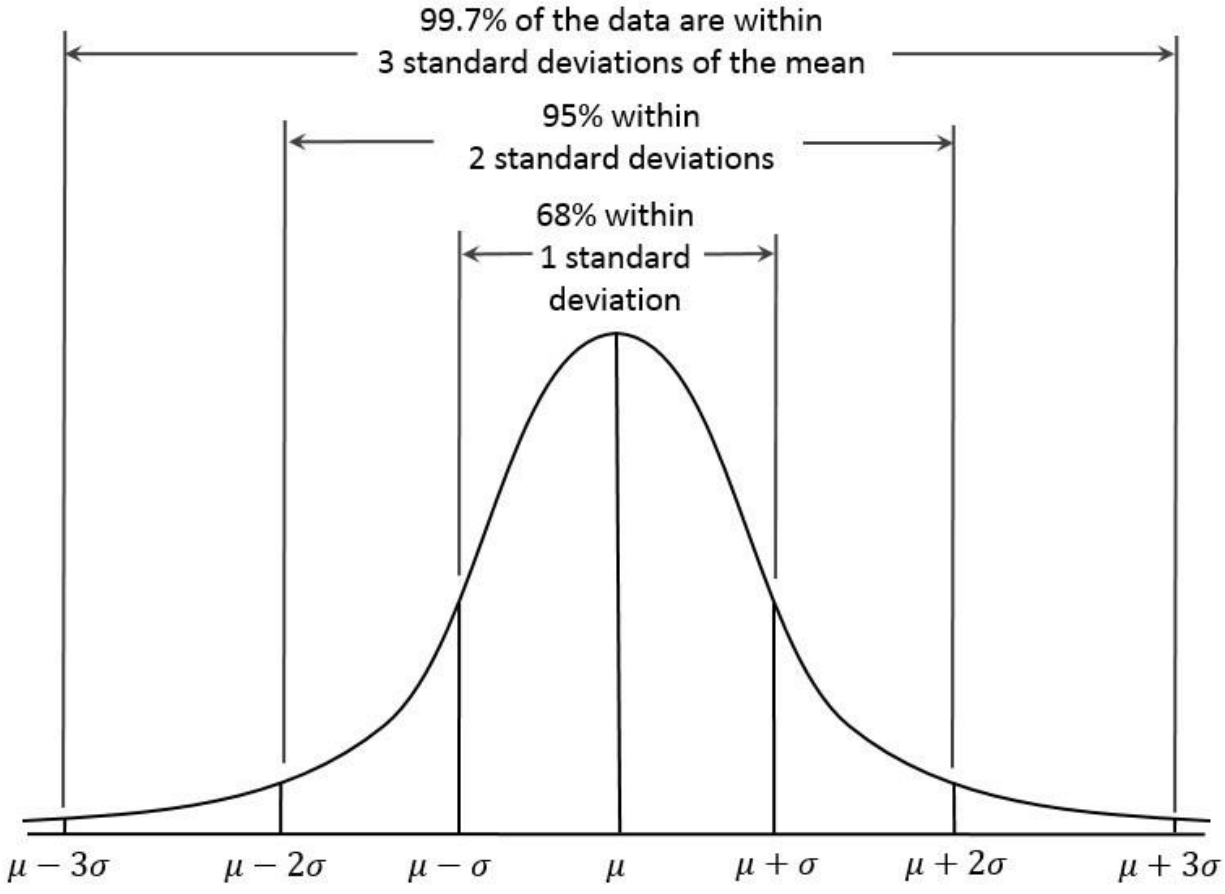
If your dataset has outliers, it will be easy to spot them with a boxplot. There are different methods to determine that a data point is an outlier. The most widely known is the *1.5xIQR rule*.

1.5xIQR rule

Outliers are extreme observations in the dataset. So a rule of thumb to determine if a data point is extreme is to compare it against the interquartile range.

It makes sense to use the interquartile range (IQR) to spot outliers. The IQR is the range of values between the first and third quartiles, i.e., 25th and 75th percentiles, so it will include the majority of the data points in the dataset.

But why 1.5 times the interquartile range? This is related to an important characteristic of the Normal Distribution known as the 68–95–99 rule.



With the 68–95–99 rule, we know that:

- 68% of the data is within one standard deviation above or below the mean,
- 95% of the data is within two standard deviations from the mean,
- 99.7% of the data is within three standard deviations from the mean.

Only very few data points will be beyond three standard deviations from the mean, more precisely, only 0.3% of the data points. So any data point that is seen farther than three standard deviations is considered extreme.

To check if a data point is an outlier and check if it falls farther than three standard deviations, we calculate:

- Q1–1.5xIQR,
- Q3 + 1.5xIQR.

These represent the lower and upper bounds of the area in the distribution that is not considered extreme. Which ends up being approximately 3 standard deviations from the mean.

The multiplying factor is 1.5, because any number greater than that would result in a range bigger than 3 standard deviations. So, mathematicians settled in a number in the middle.

With the 68–95–99 rule, we know that:

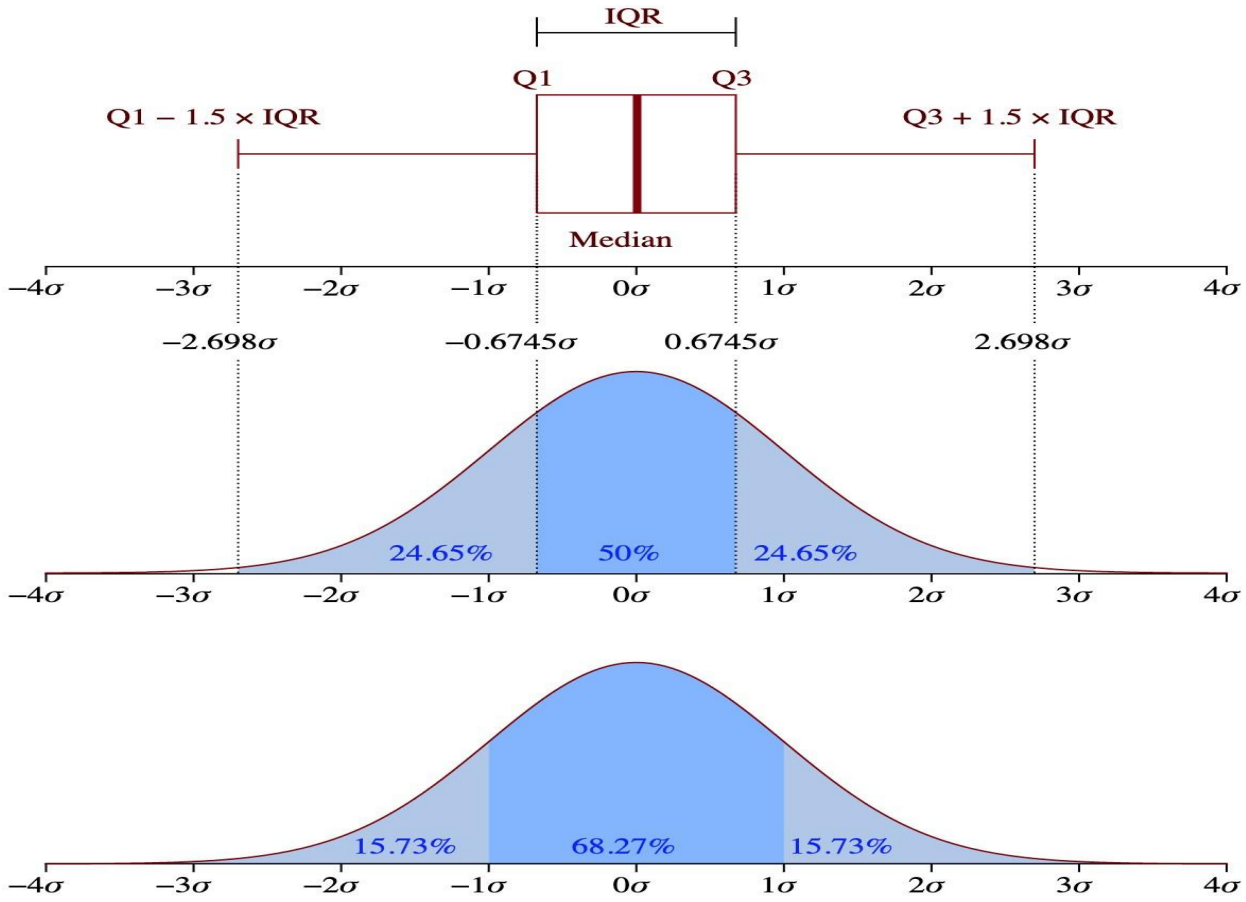
- 68% of the data is within one standard deviation above or below the mean,
- 95% of the data is within two standard deviations from the mean,
- 99.7% of the data is within three standard deviations from the mean.

Only very few data points will be beyond three standard deviations from the mean, more precisely, only 0.3% of the data points. So any data point that is seen farther than three standard deviations is considered extreme.

To check if a data point is an outlier and check if it falls farther than three standard deviations, we calculate:

- Q1–1.5xIQR,
- Q3 + 1.5xIQR.

These represent the lower and upper bounds of the area in the distribution that is not considered extreme. Which ends up being approximately 3 standard deviations from the mean.The multiplying factor is 1.5, because any number greater than that would result in a range bigger than 3 standard deviations. So, mathematicians settled in a number in the middle.



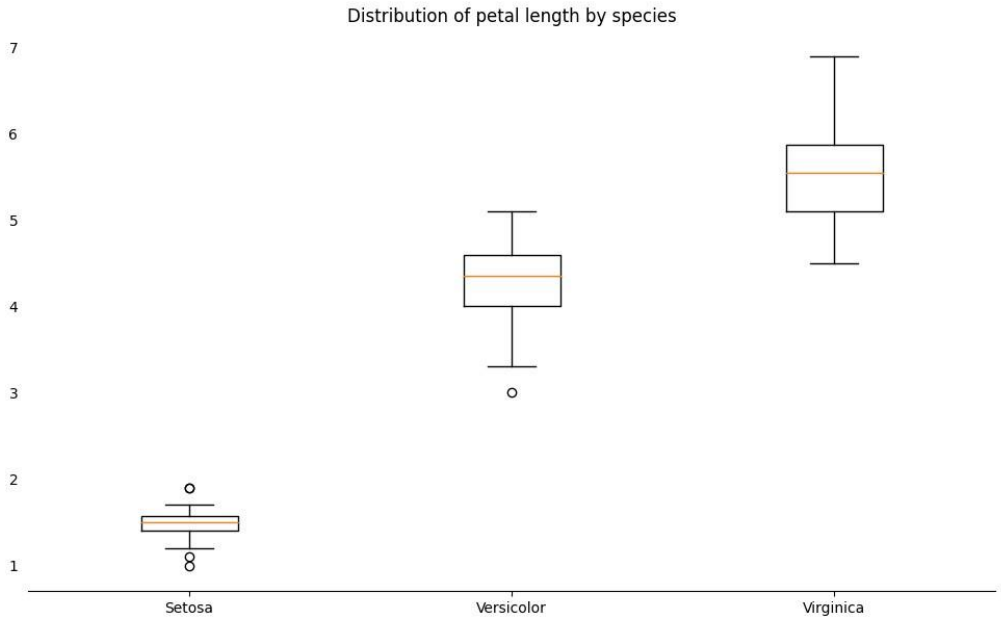
Any data point *lower than* the lower bound or *greater than* the upper bound is an outlier:

- (data point value) < Q1–1.5xIQR, then it's an outlier.
- (data point value) > Q3 + 1.5xIQR, then it's an outlier.

Customizing boxplots to compare distributions

Boxplots are also a great tool to compare different distributions.

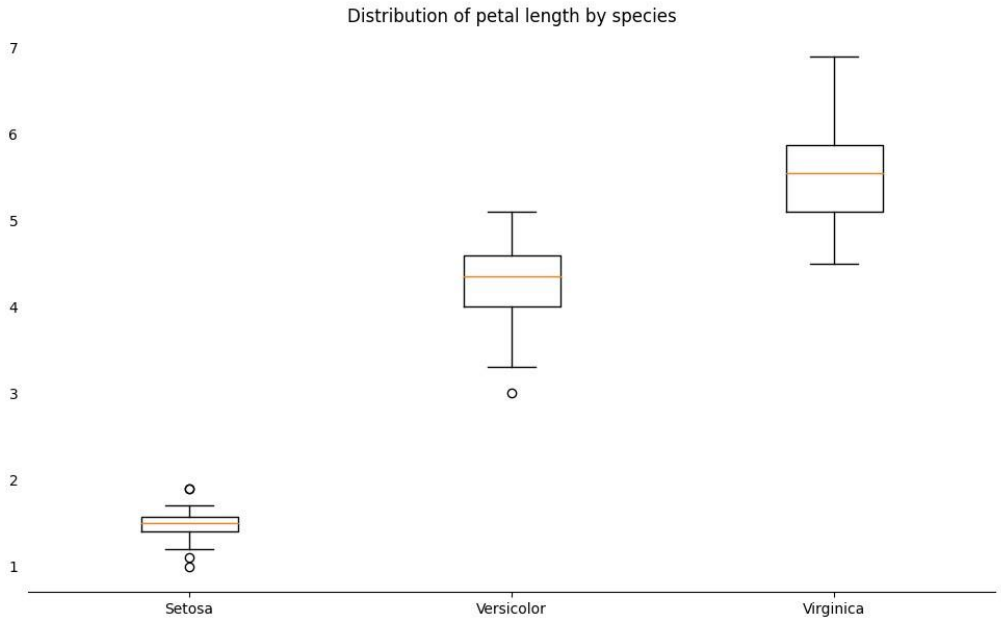
Let's compare the distributions of petal length for flowers in the Iris dataset.



Here's how you can create this plot.

```
import numpy as np
import pandas as pd
from sklearn import datasets
import matplotlib.pyplot as plt# Load Iris dataset
iris = datasets.load_iris()# Preparing Iris dataset
iris_data = pd.DataFrame(data=iris.data, columns=['sepal_length',
'sepal_width', 'petal_length', 'petal_width'])iris_target =
pd.DataFrame(data=iris.target, columns=['species'])
iris_df = pd.concat([iris_data, iris_target], axis=1)# Add species
name
iris_df['species_name'] = np.where(iris_df['species'] == 0,
'Setosa', None)iris_df['species_name'] =
np.where(iris_df['species'] == 1, 'Versicolor',
iris_df['species_name'])iris_df['species_name'] =
```

```
np.where(iris_df['species'] == 2, 'Virginica',
iris_df['species_name'])
# Prepare petal length by species datasets
setosa_petal_length = iris_df[iris_df['species_name'] ==
'Setosa']['petal_length']versicolor_petal_length =
iris_df[iris_df['species_name'] ==
'Versicolor']['petal_length']virginica_petal_length =
iris_df[iris_df['species_name'] == 'Virginica']['petal_length']
# Visualize petal length distribution for all speciesfig, ax =
plt.subplots(figsize=(12, 7))# Remove top and right border
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_visible(False)# Remove y-axis tick marks
ax.yaxis.set_ticks_position('none')# Add major gridlines in the y-
axis
ax.grid(color='grey', axis='y', linestyle='-', linewidth=0.25,
alpha=0.5)# Set plot title
ax.set_title('Distribution of petal length by species')# Set
species names as labels for the boxplot
dataset = [setosa_petal_length, versicolor_petal_length,
virginica_petal_length]labels = iris_df['species_name'].unique()
ax.boxplot(dataset, labels=labels)plt.show()
```



We can extract a few insights from this plot:

- Iris Setosa has a much smaller petal length than Iris Versicolor and Virginica. It ranges from approximately 1 to 2 centimeters.
- The range of petal length of Iris Virginica is bigger than both the ranges of values for Iris Setosa and Versicolor. We can see that from *how tall the box is* for Iris Virginica compared to the other two.
- Both Iris Setosa and Veriscolor have outliers.

We can also confirm these insights by looking at the summary metrics of each distribution.

	Min	Max	Mean	25th percentile	Median	75th percentile	IQR
Setosa	1.00	1.90	1.46	1.40	1.50	1.58	0.18
Versicolor	3.00	5.1	4.26	4.00	4.35	4.60	0.60
Virginica	4.50	6.90	5.55	5.10	5.55	5.88	0.78

Customizing your boxplot

At first glance, it's hard to distinguish between the boxplots of the different species. The labels at the bottom are the only visual clue that we're comparing distributions.

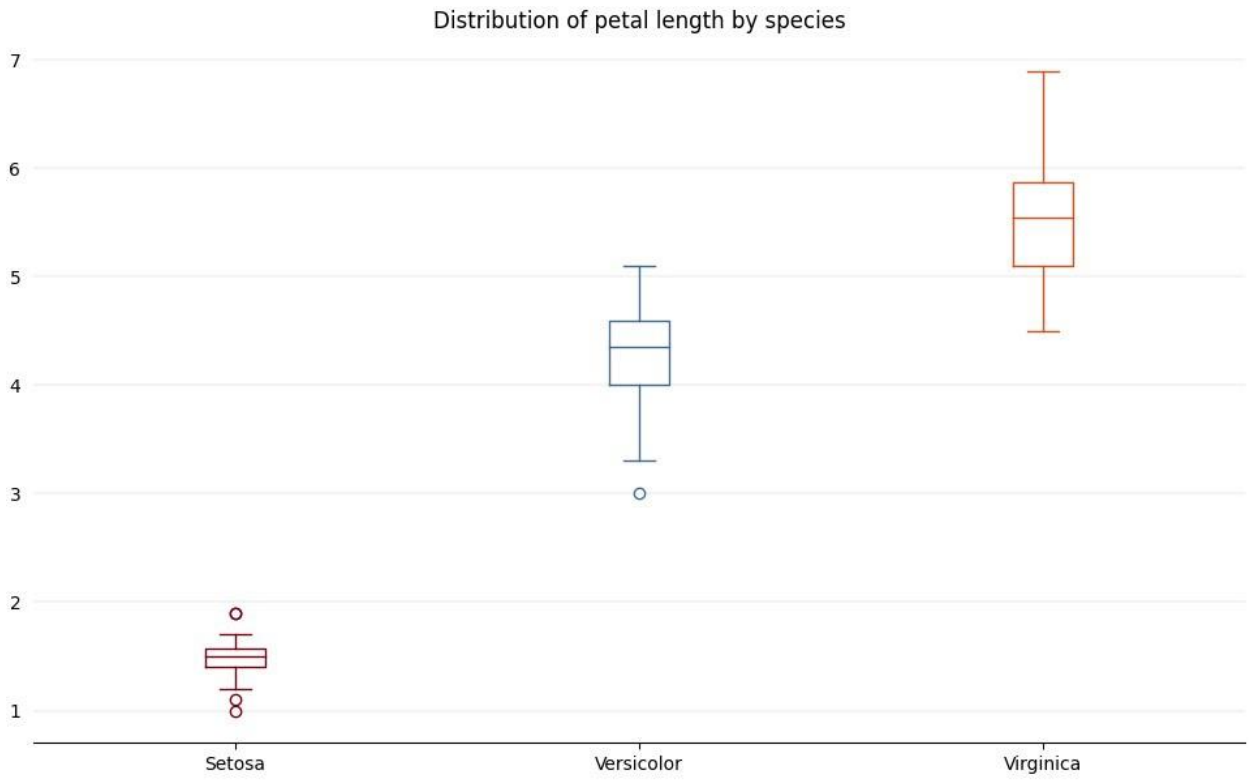
We can use the properties of the boxplot to customize each *box*. Since properties are applies to all the data that is *given* to the boxplot method, we can't take the approach of the last plot and use an array with the petal length for each species as an input.

We'll have to plot the petal length for each species and applies properties to each one of them.

We're going to use the following parameters:

- *positions*: position of the boxplot in the plot area. We don't want to plot each species' boxplot on top of each other, so we use this to set the position in the x-axis where each boxplot will be drawn.
- *medianprops*: dictionary of properties applied to median line inside the boxplot.
- *whiskerprops*: dictionary of properties applied to the whiskers.
- *capprops*: dictionary of properties applied to the caps on the whiskers.
- *flierprops*: dictionary of properties applied to outliers.

There are other several properties we can customize. In this example I'm going to just add a different color for each of the boxplots, so it's easier to see that we're visualizing different distributions.



Conclusion:-

As mentioned in the beginning, quality and efforts invested in data exploration differentiates a good model from a bad model.This ends our guide on data exploration and preparation. In this comprehensive guide, we looked at the seven steps of data exploration in detail. The aim of this series was to provide an in depth and step by step guide to an extremely important process in data science.