UNIT-V

Demonstrate Range

Python range() function generates the immutable sequence of numbers starting from the given start integer to the stop integer. The range() is a built-in function that returns a range object that consists series of integer numbers, which we can iterate using a for loop.

In Python, Using a for loop with range(), we can repeat an action a specific number of times.

# Example 1: Using range() in a for loop

for i in range(5):

print(i)

0

1

2

3

4

Demonstrate summary

In Python, the sum() function is used to calculate the sum of a sequence (such as a list or tuple) of numbers. The sum() function takes an iterable (e.g., a list) as its argument and returns the sum of all the elements in that iterable.

# Example: Using sum() to calculate the sum of a list

numbers = [1, 2, 3, 4, 5]

result = sum(numbers)

print("Sum:", result)

Sum: 15

Demonstrate Mean function in python

In Python, you can calculate the mean (average) of a collection of numbers using the mean() function from the statistics module. Here's an example:

from statistics import mean

# Example: Calculating the mean of a list of numbers

numbers = [2, 4, 6, 8, 10]

mean\_value = mean(numbers)

print("Mean:", mean\_value)

Mean: 6.0

n this example, the mean() function is applied to the list numbers, and it calculates the mean of all the elements in the list (2 + 4 + 6 + 8 + 10) / 5, which is 6.0.

Make sure to import the mean function from the statistics module before using it. If you're working with Python 3.4 or earlier, the statistics module may not be available, and you can use the numpy library instead:

import numpy as np

# Example using numpy for mean calculation

numbers = [2, 4, 6, 8, 10]

mean\_value = np.mean(numbers)

print("Mean:", mean\_value)

Mean: 6.0

Demonstrate variance in python

In Python, you can calculate the variance of a collection of numbers using the variance() function from the statistics module or the var() function from the numpy library. Here's an example using both approaches:

from statistics import variance

# Example: Calculating the variance of a list of numbers

numbers = [2, 4, 6, 8, 10]

variance\_value = variance(numbers)

print("Variance:", variance\_value)

Variance: 8.0

In this example, the variance() function from the statistics module is applied to the list numbers, and it calculates the variance of all the elements in the list.

Using the numpy library:

import numpy as np

# Example: Calculating the variance using numpy

numbers = [2, 4, 6, 8, 10]

variance\_value = np.var(numbers)

print("Variance:", variance\_value)

Variance: 8.0

In this example, the var() function from the numpy library is used to calculate the variance of the same list of numbers.

Both examples demonstrate how to use either the variance() function from the statistics module or the var() function from the numpy library to calculate the variance of a collection of numbers in Python. The variance is a measure of the spread or dispersion of a set of values.

Demonstrate Standard Deviation in Python

In Python, you can calculate the standard deviation of a collection of numbers using the stdev() function from the statistics module or the std() function from the numpy library. Here's an example using both approaches:

Using the statistics module:

from statistics import stdev

# Example: Calculating the standard deviation of a list of numbers

numbers = [2, 4, 6, 8, 10]

std\_deviation\_value = stdev(numbers)

print("Standard Deviation:", std\_deviation\_value)

Standard Deviation: 2.8284271247461903

In this example, the stdev() function from the statistics module is applied to the list numbers, and it calculates the standard deviation of all the elements in the list.

Using the numpy library:

import numpy as np

# Example: Calculating the standard deviation using numpy

numbers = [2, 4, 6, 8, 10]

std\_deviation\_value = np.std(numbers)

print("Standard Deviation:", std\_deviation\_value)

Standard Deviation: 2.8284271247461903

In this example, the std() function from the numpy library is used to calculate the standard deviation of the same list of numbers.

Both examples demonstrate how to use either the stdev() function from the statistics module or the std() function from the numpy library to calculate the standard deviation of a collection of numbers in Python. The standard deviation is a measure of the amount of variation or dispersion in a set of values.

histogram in python using population data set

To create a histogram in Python using a population dataset, you can use the matplotlib library. Here's an example:

import matplotlib.pyplot as plt

import numpy as np

# Example: Creating a histogram using a population dataset

population\_data = [65, 68, 70, 72, 73, 75, 77, 80, 82, 85, 88, 90, 92, 95, 98, 100]

# Plotting the histogram

plt.hist(population\_data, bins=10, edgecolor='black', alpha=0.7)

plt.title('Population Data Histogram')

plt.xlabel('Height')

plt.ylabel('Frequency')

plt.grid(True)

plt.show()

In this example, the plt.hist() function from matplotlib.pyplot is used to create a histogram. The population\_data list contains height values, and the bins parameter specifies the number of bins (intervals) for the histogram.

Adjust the bins parameter to control the number of bins in the histogram. In this example, bins=10 means the data will be divided into 10 bins.

Make sure to have the matplotlib library installed. You can install it using:

pip install matplotlib

Histograms in Python

A histogram visualises the distribution of data across distinct groups with continuous classes. It is represented with set of rectangular bars with widths equal to the class intervals and areas proportional to frequencies in the respective classes. A histogram may hence be defined as a graphic of a frequency distribution that is grouped and has continuous classes. It provides an estimate of the distribution of values, their extremes, and the presence of any gaps or out-of-the-ordinary numbers. They are useful in providing a basic understanding of the probability distribution.

Constructing a Histogram: To construct a histogram, the data is grouped into specific class intervals, or “bins” and plotted along the x-axis. These represent the range of the data. Then, the rectangles are constructed with their bases along the intervals for each class. The height of these rectangles is measured along the y-axis representing the frequency for each class interval. It's important to remember that in these representations, every rectangle is next to another because the base spans the spaces between class boundaries.

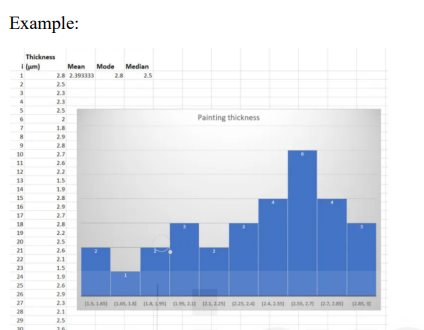
Best Practices

• Analyse various data groups: The best data groupings can be found by creating a variety of histograms.

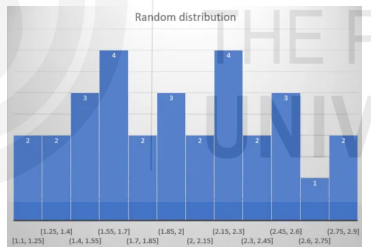
• Break down compartments using colour: The same chart can display a second set of categories by colouring the bars that represent each category.

Left-skewed distribution:

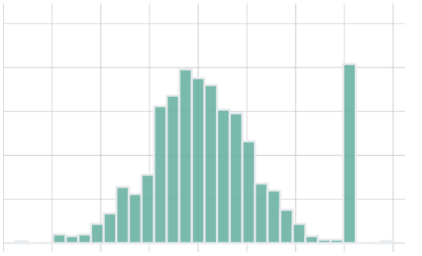
A distribution that is skewed to the left is sometimes referred to as a negatively skewed distribution. A distribution that is left-skewed will have a greater proportion of data values on the right side of the distribution and a lesser proportion of data values on the left. When the data have a range limit on the right side of the histogram, a right-skewed distribution commonly results. An alternative name for this is a right-tailed distribution.

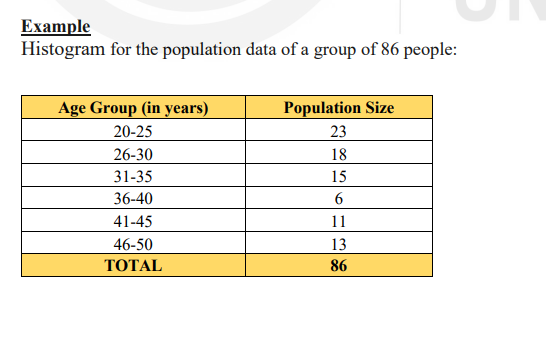


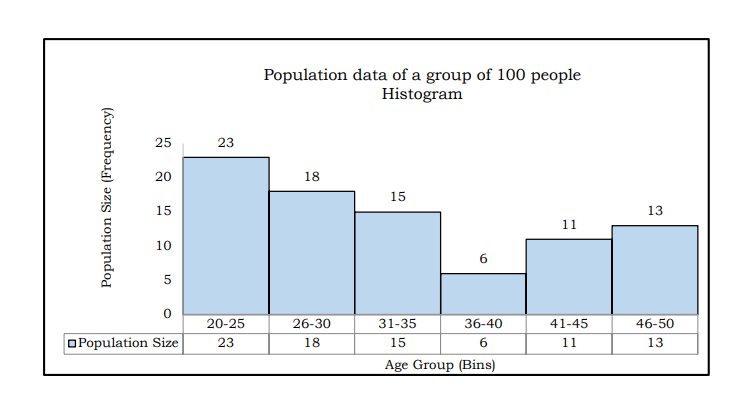
A random distribution: A random distribution is characterised by the absence of any clear pattern and the presence of several peaks. When constructing a histogram using a random distribution, it is possible that several distinct data attributes will be blended into one. As a result, the data ought to be partitioned and investigated independently



Edge Peak Distribution: When there is an additional peak at the edge of the distribution that does not belong there, this type of distribution is called an edge peak distribution. Unless you are very positive that your data set has the expected number of outliers, this almost always indicates that you have plotted (or collected) your data incorrectly (i.e. a few extreme views on a survey).







Box plot in Python

When displaying data distributions using the five essential summary statistics of minimum, first quartile, median, third quartile, and maximum, box-and-whisker plots, also known as boxplots, are widely employed. It is a visual depiction of data that aids in determining how widely distributed or how much the data values change. These boxplots make it simple to compare the distributions since it makes the centre, spread, and overall range understandable. They are utilised for data analysis wherein the graphical representations are used to determine the following

1. Shape of Distribution

2. Central Value

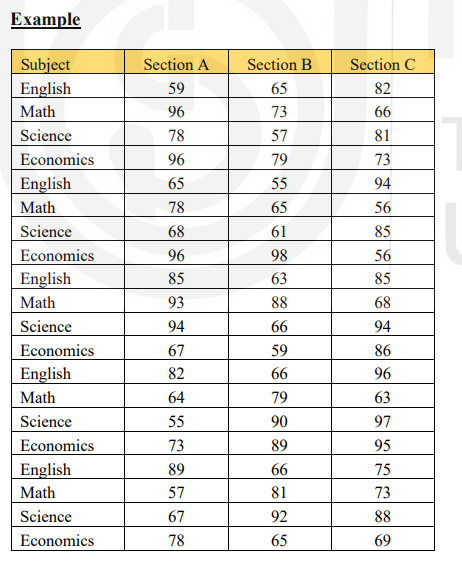
3. Variability of Data

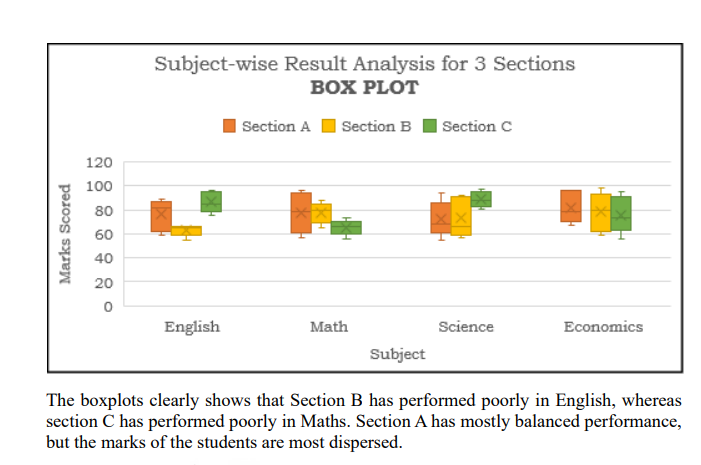
Constructing a Boxplot: The two components of the graphic are described by their names: the box, which shows the median value of data along with the first and third quartiles (25 percentile and 75 percentile), and the whiskers, which shows the remaining data. The 3rd quartile's difference from the first quartile of data is called the interquartile range. The highest and minimum points in the data can also be displayed using the whiskers. The points beyond 1.5 ´ interquartile range can be identified as suspected outliers

Best Practices

• Cover the points within the box: This aids the viewer in concentrating on the outliers.

• Box plot comparisons between categorical dimensions: Box plots are excellent for quickly comparing dataset distributions.





SCATTER PLOTS In Python

Scatter plot is the most commonly used chart when observing the relationship between two quantitative variables. It works particularly well for quickly identifying possible correlations between different data points. The relationship between multiple variables can be efficiently studied using scatter plots, which show whether one variable is a good predictor of another or whether they normally fluctuate independently. Multiple distinct data points are shown on a single graph in a scatter plot. Following that, the chart can be enhanced with analytics like trend lines or cluster analysis. It is especially useful for quickly identifying potential correlations between data points

Constructing a Scatter Plot: Scatter plots are mathematical diagrams or plots that rely on Cartesian coordinates. In this type of graph, the categories being compared are represented by the circles on the graph (shown by the colour of the circles) and the numerical volume of the data (indicated by the circle size). One colour on the graph allows you to represent two values for two variables related to a data set, but two colours can also be used to include a third variable.

Best Practices

• Analyze clusters to find segments: Based on your chosen variables, cluster analysis divides up the data points into discrete parts.

• Employ highlight actions: You can rapidly identify which points in your scatter plots share characteristics by adding a highlight action, all the while keeping an eye on the rest of the dataset.

• mark customization: individual markings Add a simple visual hint to your graph that makes it easy to distinguish between various point groups.

