PROFESSIONAL ELECTIVE COURSES: VERTICALS
VERTICAL 1: DATA SCIENCE
CCS346 EXPLORATORY DATA ANALYSIS
UNIT III - UNIVARIATE ANALYSIS

Introduction to Single variable: Distributions and Variables - Numerical Summaries of Level and Spread - Scaling and Standardizing – Inequality.

Introduction

In statistics, there are three kinds of techniques that are used in the data analysis. These are univariate analysis, bivariate analysis, and multivariate analysis. Univariate analysis refers to the analysis of one variable. You can remember this because the prefix “uni” means “one.

”Univariate analysis is a basic kind of analysis technique for statistical data. Here the data contains just one variable and does not have to deal with the relationship of a cause and effect.

The analysis of two variables and their relationship is termed bivariate analysis. If three or more variables are considered simultaneously, it is multivariate analysis.

Example 1:

- Counting the number of boys and girls in the classroom

Example 2:

Consider the following household dataset:

Household ID	Household Size	Annual Income	Number of Pets
1	2	$37,000	0
2	4	$49,000	0
3	4	$58,000	1
4	1	$68,000	3
5	3	$61,000	2
6	5	$64,000	2
7	6	$79,000	1
8	4	$89,000	1
9	7	$104,000	1
10	2	$95,000	0

Now perform univariate analysis on the variable Household Size:

Household ID	Household Size	Annual Income	Number of Pets
1	2	$37,000	0
2	4	$49,000	0
3	4	$58,000	1
4	1	$68,000	3
5	3	$61,000	2
6	5	$64,000	2
7	6	$79,000	1
8	4	$89,000	1
9	7	$104,000	1
10	2	$95,000	0

Three common ways to perform univariate analysis:

- Summary Statistics
- Frequency Distributions
- Charts

Summary Statistics

There are two popular types of summary statistics:

- Measures of central tendency: these numbers describe where the centre of a dataset is located.
    - Examples include the mean and the median.
- Measures of Dispersion: These numbers describe how evenly distributed the values are in a dataset.
    - Examples are range, standard deviation, interquartile range, and variance.
        - Range -the difference between the max value and min value in a dataset
        - Standard Deviation- an average measure of the spread
        - Interquartile Range- the spread of the middle 50% of values

Frequency Distributions

- Frequency means how often something takes place. The frequency observation tells the number of times for the occurrence of an event.
    - The frequency distribution table show qualitative and quantitative variables.

Charts

- Another way to perform univariate analysis is to create charts to visualize the distribution of values for a certain variable.
- Examples include Boxplots, Histograms, Density Curves, Pie Charts

Bar chart

- The bar chart is represented in the form of rectangular bars. The graph will compare various categories.
- The graph could be plotted vertically or horizontally. The horizontal or the x-axis will represent the category and the vertical y-axis represents the category's value. The bar graph looks at the data set and makes comparisons.

Histogram

- The histogram is the same as a bar chart which analysis the data counts. The bar graph will count against categories and the histogram displays the categories into bins. The bin is capable of showing the number of data positions, the range, or the interval.

Frequency Polygon

- The frequency polygon is similar to the histogram. However, these can be used to compare the data sets or in order to display the cumulative frequency distribution. The frequency polygon will be represented as a line graph.

Pie Chart

- The pie chart displays the data in a circular format. The graph is divided into pieces where each piece is proportional to the fraction of the complete category. So each slice of the pie in the pie chart is relative to categories size. The entire pie is 100 percent and when you add up each of the pie slices then it should also add up to 100.

Example 3:
Performing Univariate analysis using the Household Size variable from our dataset mentioned earlier:

Household ID	Household Size	Annual Income	Number of Pets
1	2	$37,000	0
2	4	$49,000	0
3	4	$58,000	1
4	1	$68,000	3
5	3	$61,000	2
6	5	$64,000	2
7	6	$79,000	1
8	4	$89,000	1
9	7	$104,000	1
10	2	$95,000	0

Summary Statistics

Measures of central tendency

- Mean (the average value): 3.8
- Median (the middle value): 4

Measures of Dispersion:

- Range (the difference between the max and min): 6
- Interquartile Range (the spread of the middle 50% of values): 2.5
- Standard Deviation (an average measure of spread): 1.87

Frequency Distributions

We can also create the following frequency distribution table to summarize how often different values occur:

Household Size	Frequency
1	1
2	2
3	1
4	3
5	1
6	1
7	1

- Most frequent household size = 4.

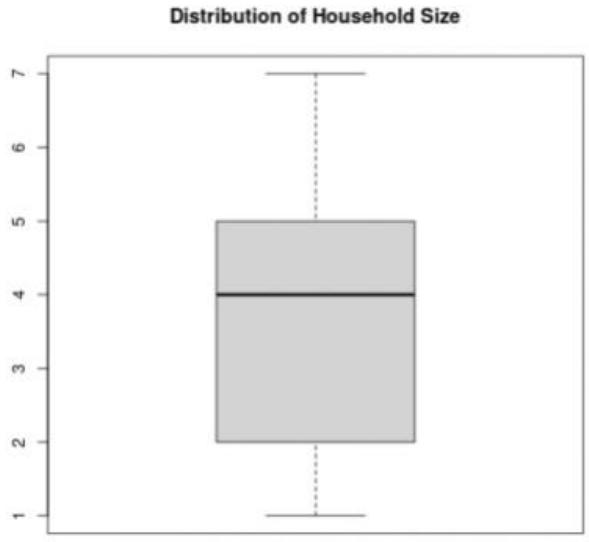
Charts

We can create the following charts to help us visualize the distribution of values for Household Size:

Boxplot

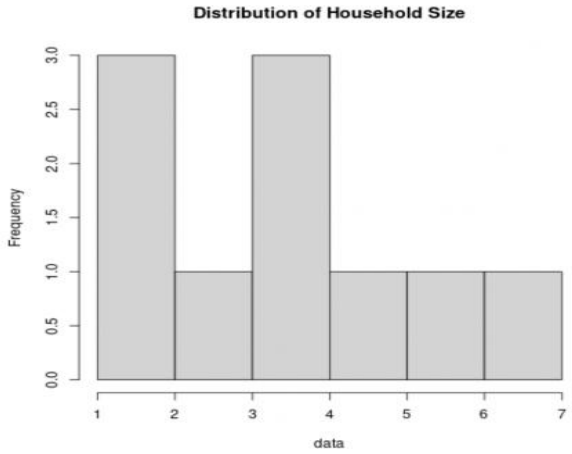
- A boxplot is a plot that shows the five-number summary of a dataset. The five-number summary includes:
  - Minimum value
  - First quartile
  - Median value
  - Third quartile
  - Maximum value

Here's what a boxplot would look like for the variable Household Size:



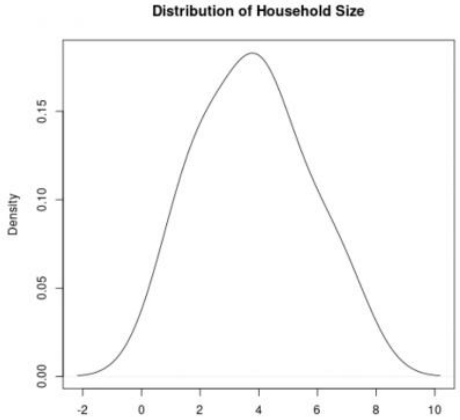
Histogram

- A histogram is a type of chart that uses vertical bars to display frequencies. This type of chart is a useful way to visualize the distribution of values in a dataset.



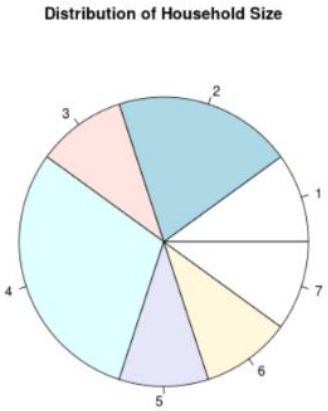
Density Curve

- A density curve is a curve on a graph that represents the distribution of values in a dataset. It's particularly useful for visualizing the "shape" of a distribution, including whether or not a distribution has one or more "peaks" of frequently occurring values and whether or not the distribution is skewed to the left or the right.



Pie Chart

- A pie chart is a type of chart that is shaped like a circle and uses slices to represent proportions of a whole.



Depending on the type of data, one of these charts may be more useful for visualizing the distribution of values than the others.

Numerical Summaries

In data exploration, numerical summaries are essential for understanding the level (central tendency) and spread (variability) of a variable. These summaries provide a concise description of the distribution of data points and help in identifying patterns, outliers, and potential relationships with other variables. Here are some common numerical summaries used in data exploration:

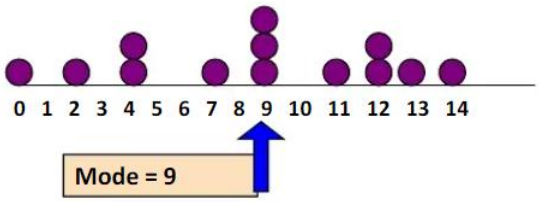
- *Measures of Central Tendency*
  - Mean: The arithmetic average of all the data points in a variable. It is calculated by summing all the values and dividing by the total number of observations.
  - Median: The middle value in an ordered list of data points. It divides the data into two equal halves, with 50% of the observations above and 50% below it.
  - Mode: The most frequently occurring value(s) in a variable. It represents the peak(s) of the distribution.
- *Measures of Variability*
  - Range: The difference between the maximum and minimum values in a variable. It provides an idea of the spread of the data but is sensitive to outliers.
  - Standard Deviation: A measure of how much the data points deviate from the mean. It quantifies the average distance between each data point and the mean.
  - Variance: The average squared deviation from the mean. It measures the variability of data points around the mean.
- *Additional Measures*
  - Quartiles: Values that divide the data into four equal parts. The first quartile (Q1) represents the 25th percentile, the median represents the 50th percentile, and the third quartile (Q3) represents the 75th percentile.
  - Interquartile Range (IQR): The range between the first and third quartiles. It provides a measure of the spread of the central half of the data and is less affected by extreme values.
  - Skewness: A measure of the asymmetry of the distribution. Positive skewness indicates a longer tail on the right side, while negative skewness indicates a longer tail on the left side.
  - Kurtosis: A measure of the peakedness or flatness of the distribution. It compares the tails and central peak to a normal distribution. Positive kurtosis indicates a more peaked distribution, while negative kurtosis indicates a flatter distribution.

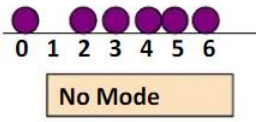
These numerical summaries provide insights into the characteristics of a variable, allowing for a better understanding of its distribution and variability. They serve as the foundation for further analysis and can guide decision-making processes in data exploration.

Measures of Central Tendency

Mode

- The mode is the most commonly occurring value in a distribution.





Example 1

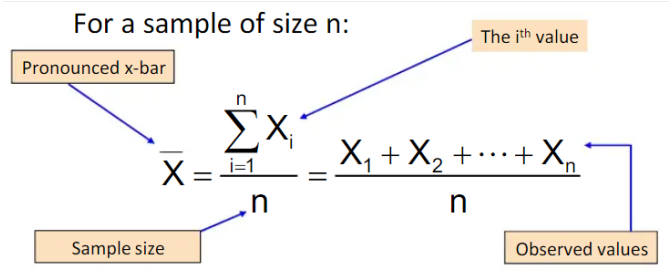
Consider this dataset showing the retirement age of 11 people, in whole years:
54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60
This table shows a simple frequency distribution of the retirement age data.

Age	Frequency
54	3
55	1
56	1
57	2
58	2
60	2

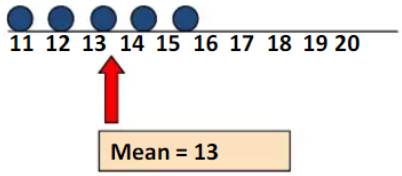
The most commonly occurring value is 54, therefore the mode of this distribution is 54 years.

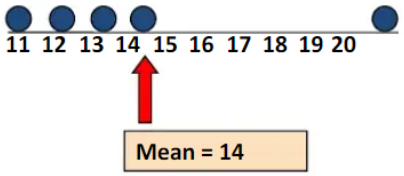
Mean

- The mean is the sum of the value of each observation in a dataset divided by the number of observations. This is also known as the arithmetic average.



$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n} = \frac{X_1 + X_2 + \cdots + X_n}{n}$$





$$\frac{11+12+13+14+15}{5} = \frac{65}{5} = 13$$

$$\frac{11+12+13+14+20}{5} = \frac{70}{5} = 14$$

Looking at the retirement age distribution again:
54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60

- The mean is calculated by adding together all the values (54+54+54+55+56+57+57+58+58+60+60 = 623) and dividing by the number of observations (11) which equals 56.6 years.

$$Mean = \frac{54 + 54 + 54 + 55 + 56 + 57 + 57 + 58 + 58 + 60 + 60}{11} = 56.6\ Years$$

Median

- The median is the middle value in distribution when the values are arranged in ascending or descending order.
- The median divides the distribution in half (there are 50% of observations on either side of the median value). In a distribution with an odd number of observations, the median value is the middle value.

$\tilde{x}$ = the middle value $[(\frac{n+1}{2})^{\text{th}}$ ordered value] if n is odd and
$\tilde{x}$ = the average of the 2 middle values $[(\frac{n}{2})^{\text{th}}$ and $(\frac{n}{2}+1)^{\text{th}}$ ordered values] if n is even.

For an odd number of observations:

26	18	27	12	14	27	19

7 **observations**

12	14	18	19	26	27	27

in ascending order

Median = 19

For an even number of observations:

26	18	27	12	14	27	30	19

8 **observations**

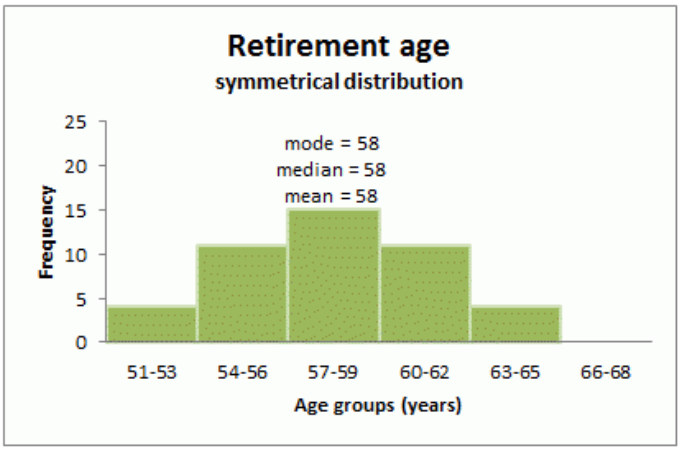
12	14	18	19	26	27	27	30

in ascending order

Median is the average of the middle two values

Median = $\frac{19+26}{2} = 22.5$

Looking at the retirement age distribution (which has 11 observations), the median is the middle value, which is 57 years:
54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60
When the distribution has an even number of observations, the median value is the mean of the two middle values. In the following distribution, the two middle values are 56 and 57, therefore the median equals 56.5 years.
52, 54, 54, 54, 55, 56, 57, 57, 58, 58, 60, 60



Important

- The mean is strongly affected by (not resistant to) outliers and skewness, whereas the median is not affected by (resistant to) outliers and skewness.
    - Outliers – mean is pulled towards the outliers
    - Skewness – mean is pulled towards the longer tail.
        - *Symmetric* : Mean = Median = Mode
        - *Left-skewed(Negatively skewed):* Mode > Median > Mean
        - *Right-skewed(positively skewed):* Mode < Median < Mean

Skewed distributions

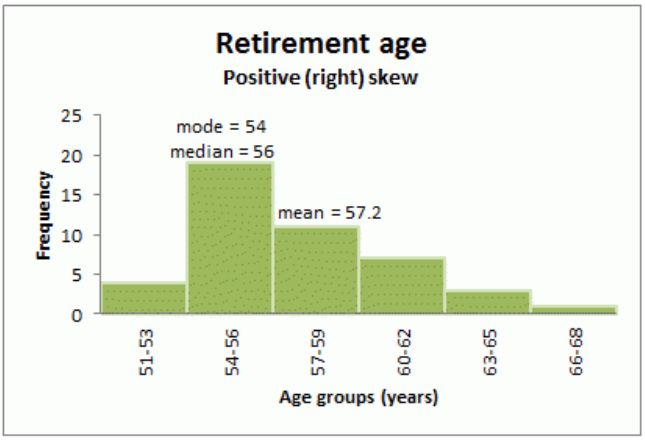
- When a distribution is skewed, the mode remains the most commonly occurring value, the median remains the middle value in the distribution, but the mean is generally 'pulled' in the direction of the tails. In a skewed distribution, the median is often a preferred measure of central tendency, as the mean is not usually in the middle of the distribution.

Positively or Right-skewed

- A distribution is said to be positively or right skewed when the tail on the right side of the distribution is longer than the left side. In a positively skewed distribution, the mean to be 'pulled' toward the right tail of the distribution.

- The following graph shows a larger retirement age data set with a distribution which is right skewed. The data has been grouped into classes, as the variable being measured (retirement age) is continuous. The mode is 54 years, the modal class is 54-56 years, the median is 56 years, and the mean is 57.2 years.

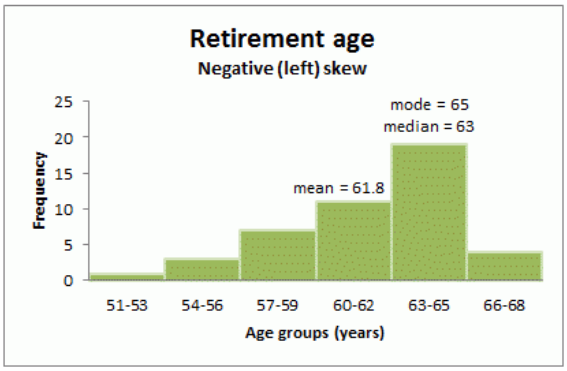
Retirement age: Positive (right) skew



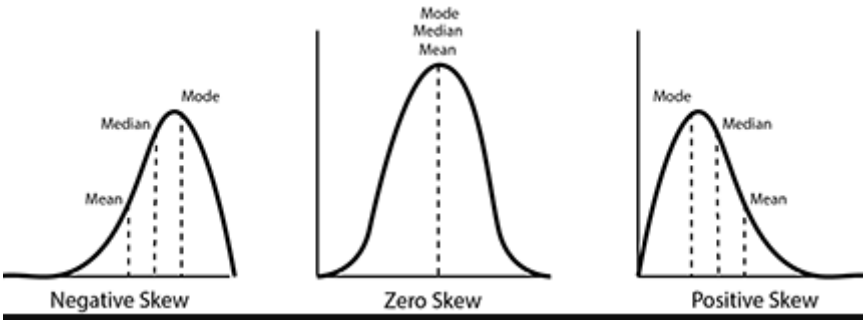
Negatively or left-skewed

- A distribution is said to be negatively or left skewed when the tail on the left side of the distribution is longer than the right side. In a negatively skewed distribution, the mean to be 'pulled' toward the left tail of the distribution.
- The following graph shows a larger retirement age dataset with a distribution which left skewed. The mode is 65 years, the modal class is 63-65 years, the median is 63 years and the mean is 61.8 years.

Retirement age: Negative (left) skew



Relation among mean, mode and median



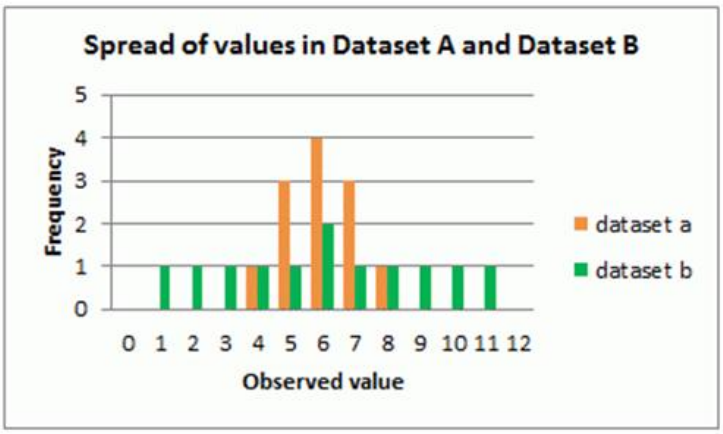
Example:
Consider the following example:

Dataset examples

Dataset A	Dataset B
4, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7, 8	1, 2, 3, 4, 5, 6, 6, 7, 8, 9, 10, 11

Analysis:

- Mode (most frequent value), Median (middle value*) and Mean (arithmetic average) of both datasets is 6.
- If we just look at the measures of central tendency, we assume that the datasets are the same. However, if we look at the spread of the values in the following graph, we can see that Dataset B is more dispersed than Dataset A.
    - The *measures of central tendency* and *measures of spread* help us to better understand the data.



- ***Range***
  Range = Difference between the smallest value and the largest value in a dataset.

4, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7, 8	Range =4 [High value (8) and Low value (4)]
1, 2, 3, 4, 5, 6, 6, 7, 8, 9, 10, 11	Range =10 [High value (8) and Low value (4)]

- ***Number line view of datasets***

Dataset A

4 5 6 7 8

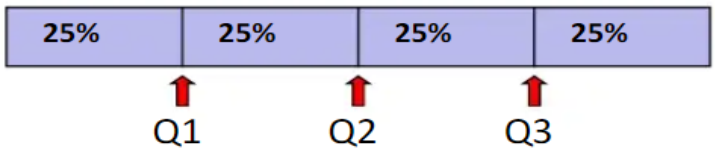
Dataset B

1 2 3 4 5 6 7 8 9 10 11

Range of values for Dataset B is larger than Dataset A.

- ***Quartiles***

Quartiles divide an ordered dataset into four equal parts, and refer to the values of the point between the quarters. A dataset may also be divided into quintiles (five equal parts) or deciles (ten equal parts)



Quartiles

25% of values	Quartile 1 (Q1)	25% of values	Quartile 2 (Q2)	25% of values	Quartile 3 (Q3)	25% of values

- *25th percentile*
  Lower quartile (Q1) is the point between the lowest 25% of values and the highest 75% of values.

- *50th percentile*
  Second quartile (Q2) is the middle of the data set. It is also called the median.

- *75th percentile*
  Upper quartile (Q3) is the point between the lowest 75% and highest 25% of values.

- *Calculating quartiles*

Dataset A

4	5	5	Q1	5	6	6	Q2	6	6	7	Q3	7	7	8

- As the quartile point falls between two values, the mean (average) of those values is the quartile value:
  - Q1 = (5+5) / 2 = 5
  - Q2 = (6+6) / 2 = 6
  - Q3 = (7+7) / 2 = 7

Dataset B

1	2	3	Q1	4	5	6	Q2	6	7	8	Q3	9	10	11

- As the quartile point falls between two values, the mean (average) of those values is the quartile value:
  - Q1 = (3+4) / 2 = 3.5
  - Q2 = (6+6) / 2 = 6
  - Q3 = (8+9) / 2 = 8.5

- ***Interquartile***

The interquartile range (IQR) is the difference between the upper (Q3) and lower (Q1) quartiles. IQR is often seen as a better measure of spread than the range as it is not affected by outliers.

The interquartile range for Dataset A is = 2
Interquartile range = Q3 - Q1 => 7 – 5 => 2
The interquartile range for Dataset B is = 5
Interquartile range = Q3 - Q1 => 8.5 - 3.5 => 5

Example 2:
Data is arranged in the ordered array as follows: 11, 12, 13, 16, 16, 17, 18, 21, 22.
Solution:

Number of items = 9
Q1 is in the (9+1)/4 = 2.5 position of the ranked data,
⇨ Q1 = (12+13)/2 = 12.5
Q2 is in the (9+1)/2 = 5th position of the ranked data,
⇨ Q2 = median = 16
Q3 is in the (9+1)/4 = 7.5 position of the ranked data,
⇨ Q3 = (18+21)/2 = 19.5

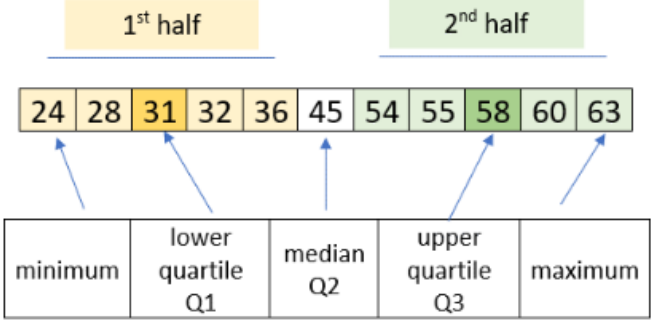
Five Number Summary

The five number summary consists of the minimum, lower quartile (Q1), median (Q2), upper quartile (Q3) and the maximum. The minimum is the smallest number, the maximum is the largest, the median is in the middle, Q1 is the median of the first half of the data and Q3 is the median of the second half.

$X_{smallest}$ -- Q_1 -- Median -- Q_3 -- $X_{largest}$

The five numbers that help describe the center, spread and shape of data are

- $X_{smallest}$
- First Quartile (Q1)
- Median (Q2)
- Third Quartile (Q3)
- $X_{largest}$



Variance and standard deviation

The variance and the standard deviation are measures of the spread of the data around the mean.

- Datasets with a small spread, all values are very close to the mean, resulting in a small variance and standard deviation.
- Datasets with a larger spread, values are spread further away from the mean, leading to a larger variance and standard deviation.

Population Variance Formula

$$\sigma^2 = \frac{\sum_{i=1}^{N}(X_i - \mu)^2}{N}$$

where:

- X_i→ Refers the i^{th} unit, starting from the first observation to the last
- M→Population mean
- N →Number of units in the population

Sample Variance Formula

$$s^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$$

where:

- x_i→Refers the i^{th} unit, starting from the first observation to the last
- $\bar{x}$ → Sample mean
- n→ Number of units in the sample

	Population	Sample
Variance	$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}$	$S^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$
Standard deviation	$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}}$	$S = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$

Example 1

Find the variance and standard deviation of the following scores on an exam:
92, 95, 85, 80, 75, 50

Solution

Step 1: Mean of the data

$$Mean = \frac{92 + 95 + 85 + 80 + 75 + 50}{6} = 79.5$$

Step 2: Find the difference between each score and the mean (deviation).

Score	Score - Mean	Difference from mean	Sum of squares
92	92 – 79.5	12.5	$(12.5)^2$
95	95 – 79.5	15.5	$(15.5)^2$
85	85 – 79.5	5.5	$(5.5)^2$
80	80 – 79.5	0.5	$(0.5)^2$
75	75 – 79.5	-4.5	$(-4.5)^2$
50	50 – 79.5	-29.5	$(-29.5)^2$
Mean = 79.5			1317.50

Next we square each of these differences and then sum them.

Variance = $\frac{1317.50}{5} = 263.5$

Standard Deviation = $\sqrt{263.5} \approx 16.2$

Example 2

Find the standard deviation of the average temperatures recorded over a five-day period last winter:

18, 22, 19, 25, 12

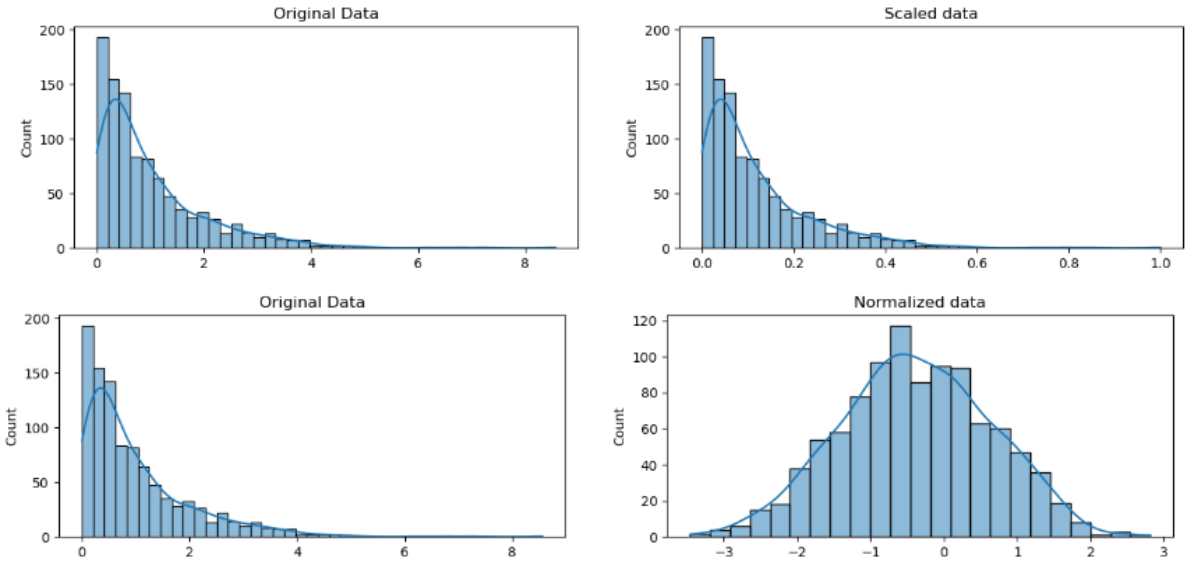
Temp	Temp - Mean	Difference from mean	Sum of squares
18	18 – 19.2	-1.2	1.44
22	22 – 19.2	2.8	7.84
19	19 – 19.2	-0.2	0.04
25	25 – 19.2	5.8	33.64
12	12 – 19.2	-7.2	51.84
Mean 96/5 = 19.2			94.80

Variance, we divide 5 – 1 = 4 ie. $\frac{94.8}{5} = 23.7$

Standard Deviation = $\sqrt{23.7} \approx 4.9$

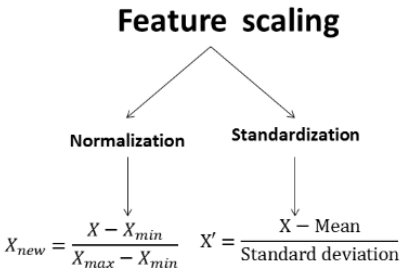
Scaling and Standardization

Scaling and standardization are techniques used in data exploration to preprocess and transform numerical variables to a common scale. These techniques are particularly useful when working with variables that have different units of measurement or different ranges.

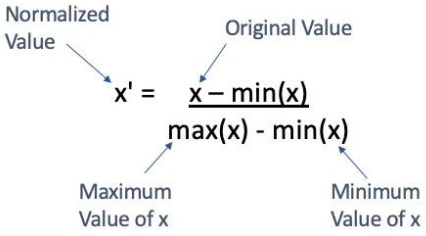


Scaling

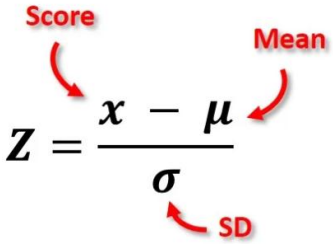
Scaling refers to the process of transforming variables to have a similar scale. It helps to ensure that all variables are on a comparable magnitude, preventing certain variables from dominating the analysis due to their larger values. Common scaling methods include:



- *Min-Max Scaling (Normalization):* This technique rescales the data to a fixed range, typically between 0 and 1. It is achieved by subtracting the minimum value of the variable and dividing it by the range (maximum minus minimum value).



- *Z-Score Standardization:* Z-score standardization transforms the data to have a mean of 0 and a standard deviation of 1. It is calculated by subtracting the mean of the variable and dividing it by the standard deviation.



Example 1

Let's calculate the Min-Max scaling for a dataset step by step. Consider the following dataset: [12, 15, 18, 20, 25].

Step 1: Identify the minimum and maximum values in the dataset.

min_val = 12
max_val = 25

Step 2: Calculate the range of the dataset.

Range = max_val - min_val
= 25 - 12
= 13

Step 3: Subtract the minimum value from each data point.

12 - min_val = 12 - 12 = 0
15 - min_val = 15 – 12 = 3
18 - min_val = 18 – 12 = 6
20 - min_val = 20 – 12 = 8
25 - min_val = 25 - 12 = 13

Step 4: Divide each result by the range

0 / range = 0 / 13 ≈ 0.000
3 / range = 3 / 13 ≈ 0.231
6 / range = 6 / 13 ≈ 0.462
8 / range = 8 / 13 ≈ 0.615
13 / range = 13 / 13 = 1.000

Resulting Min-Max scaled dataset is approximately [0.000, 0.231, 0.462, 0.615, 1.000]. Each value represents the scaled value for the corresponding data point in the original dataset, where 0 corresponds to the minimum value and 1 corresponds to the maximum value.

Example 2

Let's say we have a variable representing the income of individuals in a dataset. The original income values range from $20,000 to $100,000. We want to scale these values to a range between 0 and 1 using Min-Max Scaling.

Solution

Calculate the minimum and maximum values of the income variable:

min(x) = $20,000

max(x) = $100,000

Choose a data point, for example, $40,000, and apply the Min-Max Scaling formula:

$$x_{scaled} = \frac{40000 - 20000}{100000 - 20000} = 0.25$$

Therefore, $40,000 would be scaled to 0.25.

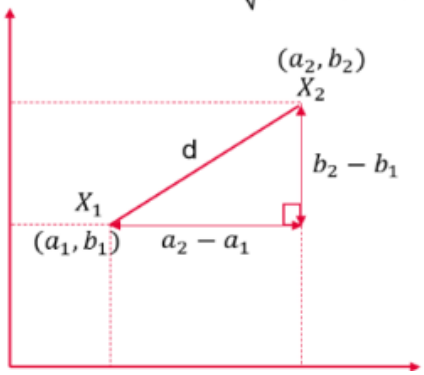
Repeat the scaling process for all other data points. The resulting scaled values will fall within the range of 0 to 1, with the minimum value transformed to 0 and the maximum value transformed to 1. When we are dealing with image processing, the pixels need normalized to be between 0 and 255.

Analysis

Employee Number	Age	Salary	
Emp1	44	73000	
Emp2	27	47000	
Emp3	30	53000	Age Range : 27 – 48
Emp4	38	62000	Salary Range : 47000 - 78000
Emp5	40	57000	
Emp6	35	53000	
Emp7	48	78000	

In machine learning everything is measured in terms of numbers and when we want to identify the nearest neighbours, similarity or dissimilarity of features. Euclidean distance is commonly used as a similarity or dissimilarity metric between data points.

$$\textbf{Euclidean Distance } d = \sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2}$$



- X_1 and X_2 are two data points means two different rows in data set. And Data set is two dimensional feature space
- For three dimensional feature space suppose two data points are $X_1(a_1, b_1, c_1)$ and $X_2(a_2, b_2, c_2)$ then distance can be calculated as below:

$$d = \sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2 + (c_2 - c_1)^2}$$

- In case of multi-dimensional vector space, More generic formula will be

$$\textbf{Euclidean Distance } d = \sum_{i=1}^{k} \sqrt{(X1_i - X2_i)^2}$$

Distance between Emp2 and Emp1 = $\sqrt{(27 - 44)^2 + (47000 - 73000)^2} = 31.06$

Distance between Emp2 and Emp3 = $\sqrt{(30-27)^2+(53000-47000)^2} = 6.70$

Normalization process:

#	Emp	Age	Salary
1	Emp1	44	73000
2	Emp2	27	47000
3	Emp3	30	53000
4	Emp4	38	62000
5	Emp5	40	57000
6	Emp6	35	53000
7	Emp7	48	78000

Normalization

Age	Normalized Age	Salary	Normalized Salary
44	0.80952381	73000	0.838709677
27	0	47000	0
30	0.142857143	53000	0.193548387
38	0.523809524	62000	0.483870968
40	0.619047619	57000	0.322580645
35	0.380952381	53000	0.193548387
48	1	78000	1
	Range 0-1		**Range 0-1**

How to calculate Normalized value?
X = 35, min = 27, max = 48 for column Age.
Xnorm(for 35) = $\frac{35-27}{48-27} = 0.3809$

Distance between Emp2 and Emp1 = $\sqrt{(0-.80)^2+(0-.83)^2} = 1.15$
Distance between Emp2 and Emp3 = $\sqrt{(.14-0)^2+(.19-0)^2} = 0.23$
} Comparison is more significant

Standardization process

Standardization is also known as z-score Normalization. In standardization, features are scaled to have zero-mean and one-standard-deviation. It means after standardization features will have mean = 0 and standard deviation = 1.

#	Emp	Age	Salary
1	Emp1	44	73000
2	Emp2	27	47000
3	Emp3	30	53000
4	Emp4	38	62000
5	Emp5	40	57000
6	Emp6	35	53000
7	Emp7	48	78000
	Mean = 37.42857		Mean = 60428.5714
	Std. Dev. = 6.883876		Std. Dev = 10499.7570

Standardization

How to calculate Standardized value?
X = 35, mean = 37.42, Std. Dev. = 6.88
for column Age.
Xstd(for 35) = $\frac{35-37.42}{6.88} = -0.3527$

Age	Standardized Age	Salary	Standardized Salary
44	0.954611636	73000	1.197306616
27	-1.514927162	47000	-1.278941158
30	-1.079126198	53000	-0.707499364
38	0.083009708	62000	0.149663327
40	0.373543684	57000	-0.326538168
35	-0.352791257	53000	-0.707499364
48	1.535679589	78000	1.673508111
	Mean = 0 **Std. dev. = 1**		**Mean = 0** **Std. dev. = 1**

Distance between Emp2 and Emp1 = $\sqrt{(-1.51-0.95)^2+(-1.27-1.19)^2} = 3.47$
Distance between Emp2 and Emp3 = $\sqrt{(-1.07+1.51)^2+(-0.70+1.27)^2} = 0.71$
} Comparison is more significant

Example 1 - Scaling a Pandas DataFrame using min-max scaling

```
import pandas as pd
from mlxtend.preprocessing import minmax_scaling

s1 = pd.Series([1, 2, 3, 4, 5, 6], index=(range(6)))
s2 = pd.Series([10, 9, 8, 7, 6, 5], index=(range(6)))
df = pd.DataFrame(s1, columns=['s1'])
df['s2'] = s2
print(df)
print("Scaled DataFrame")
minmax_scaling(df, columns=['s1', 's2'])
```

```
   s1  s2
0  1  10
1  2  9
2  3  8
3  4  7
4  5  6
5  6  5
Scaled DataFrame
          s1    s2
0       0.0     1.0
1       0.2     0.8
2       0.4     0.6
3       0.6     0.4
4       0.8     0.2
5       1.0     0.0
```

Example 2 - Scaling arrays using min-max scaling

```
import numpy as np

X = np.array([[1, 10], [2, 9], [3, 8],
              [4, 7], [5, 6], [6, 5]])
print(X)
from mlxtend.preprocessing import minmax_scaling

minmax_scaling(X, columns=[0, 1])
```

```
[[ 1 10]
 [ 2  9]
 [ 3  8]
 [ 4  7]
 [ 5  6]
 [ 6  5]]

array([[0. , 1. ],
       [0.2, 0.8],
       [0.4, 0.6],
       [0.6, 0.4],
       [0.8, 0.2],
       [1. , 0. ]])
```

Example 3 - Scaling students marks using min-max scaling

```
import numpy as np

# Sample student dataset (marks in 3 subjects)
student_data = [
    [80, 70, 90],
    [60, 50, 70],
    [90, 85, 95],
    [75, 80, 70],
    [95, 60, 80]
]

# Find the minimum and maximum values for each subject
min_values = np.min(student_data, axis=0)
max_values = np.max(student_data, axis=0)

# Perform Min-Max Scaling on each subject's marks
scaled_data = (student_data - min_values) / (max_values -
min_values)

# Create x-axis labels for subjects
subjects = ['Subject 1',
```

```
[0.57142857
0.57142857 0.8 ]
[0. 0. 0.]
[0.85714286  1.
1. ]
[0.42857143
0.85714286 0. ]
[1.       0.28571429
0.4 ]
```

```
 'Subject 2', 'Subject 3']

# Print the scaled data
for scaled_marks in scaled_data:
    print(scaled_marks)
```

Example 4 - Scaling Sonar dataset using min-max scaling

Apart from supporting library functions other functions used are:

- fit(data) method -- compute the mean and std dev for a given feature so that it can be used further for scaling.
- transform(data) -- perform scaling using mean and std dev calculated using the .fit()
- fit_transform() -- does both fit and transform.

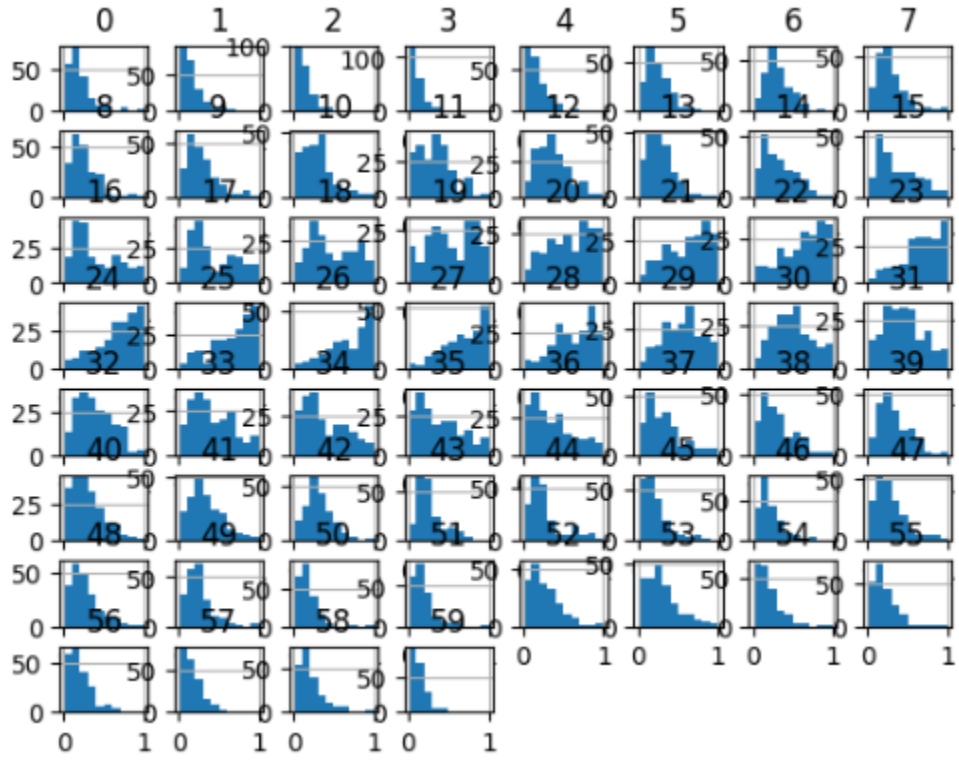
```
# visualize a minmax scaler transform of the sonar dataset
from pandas import read_csv
from pandas import DataFrame
from pandas.plotting import scatter_matrix
from sklearn.preprocessing import MinMaxScaler
from matplotlib import pyplot
# load dataset
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/sonar.csv"
dataset = read_csv(url, header=None)
# retrieve just the numeric input values
data = dataset.values[:, :-1]
# perform a robust scaler transform of the dataset
trans = MinMaxScaler()
data = trans.fit_transform(data)
# convert the array back to a dataframe
dataset = DataFrame(data)
# summarize
print(dataset.describe())
# histograms of the variables
dataset.hist()
pyplot.show()
```

Output

```
                0           1           2           3           4           5   \
count  208.000000  208.000000  208.000000  208.000000  208.000000  208.000000
mean     0.204011    0.162180    0.139068    0.114342    0.173732    0.253615
std      0.169550    0.141277    0.126242    0.110623    0.140888    0.158843
min      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
25%      0.087389    0.067938    0.057326    0.044163    0.079508    0.152714
50%      0.157080    0.129447    0.107753    0.090942    0.141517    0.220236
75%      0.251106    0.202958    0.185447    0.139563    0.237319    0.333042
max      1.000000    1.000000    1.000000    1.000000    1.000000    1.000000

                6           7           8           9   ...          50  \
count  208.000000  208.000000  208.000000  208.000000  ...  208.000000
mean     0.320472    0.285114    0.252485    0.281652  ...    0.160047
std      0.167175    0.187767    0.175311    0.192215  ...    0.119607
min      0.000000    0.000000    0.000000    0.000000  ...    0.000000
25%      0.209957    0.165215    0.132571    0.142964  ...    0.083914
50%      0.280438    0.235061    0.214349    0.244673  ...    0.138446
75%      0.407738    0.361852    0.334555    0.368082  ...    0.207420
max      1.000000    1.000000    1.000000    1.000000  ...    1.000000
```

	51	52	53	54	55	56	\
count	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	
mean	0.180031	0.265172	0.290669	0.197061	0.200555	0.213642	
std	0.137432	0.183385	0.213474	0.160717	0.147080	0.164361	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.092368	0.118831	0.127924	0.080499	0.102564	0.096591	
50%	0.151213	0.235065	0.242690	0.156463	0.165385	0.160511	
75%	0.227175	0.374026	0.394737	0.260771	0.260897	0.287642	
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	

	57	58	59
count	208.000000	208.000000	208.000000
mean	0.175035	0.216015	0.136425
std	0.148051	0.170286	0.116190
min	0.000000	0.000000	0.000000
25%	0.075515	0.098485	0.057737
50%	0.125858	0.173554	0.108545
75%	0.229977	0.281680	0.183025
max	1.000000	1.000000	1.000000



Example for Z-score Normalization

Example 1

Let's calculate the Z-scores for each data point in a dataset. Consider the following dataset: [75, 80, 85, 90, 95].

Solution

We will calculate the Z-score for each data point using the formula:

$Z = (x - \mu) / \sigma$,

where x is the data point, μ is the mean, and σ is the standard deviation

Step 1: Calculate the mean (μ) of the dataset.

mean = (75 + 80 + 85 + 90 + 95) / 5

= 85

Step 2: Calculate the standard deviation (σ) of the dataset.

std_dev = sqrt(((75-85)^2 + (80-85)^2 + (85-85)^2 + (90-85)^2 + (95-85)^2) / 5)

= sqrt((100 + 25 + 0 + 25 + 100) / 5)

= sqrt(250 / 5)

= sqrt(50)

= 7.071

Now, let's calculate the Z-score for each data point:

For data point 75:
Z = (75 - 85) / 7.071
= -1.414

For data point 80:
Z = (80 - 85) / 7.071
= -0.707

For data point 85:
Z = (85 - 85) / 7.071
= 0

For data point 90:
Z = (90 - 85) / 7.071
= 0.707

For data point 95:
Z = (95 - 85) / 7.071
= 1.414

Answer

The Z-scores for the dataset [75, 80, 85, 90, 95] are approximately [-1.414, -0.707, 0, 0.707, 1.414].

Example 2

John is a researcher for the infant clothes company BABYCLOTHES Inc. To assist in determining sizes for their baby outfits, they have hired him. The business wants to launch a new line of extra-small baby garments, but it's unsure of what size to produce. To estimate the potential size of the market for garments designed for infants weighing less than 6 pounds, scientists want to know how many premature babies are born weighing less than 6 pounds.

John can compute the Z-score, which aids in determining the deviation from the mean, if he discovers that the mean weight for a premature newborn infant is 5 pounds and the standard deviation is 1.25 pounds.

Answer

Z = (x - μ) / σ,

where x is the data point, μ is the mean, and σ is the standard deviation

⇨ Z –score = (6 - 5) / 1.25 = 0.80

Simple Python Program Z-score Normalization

```
import numpy as np

# Example dataset
```

```
data = np.array([10, 5, 7, 12, 8])

# Calculate mean and standard deviation
mean = np.mean(data)
std_dev = np.std(data)

# Z-score normalization
normalized_data = (data - mean) / std_dev

print(normalized_data)
```

Output

```
[ 0.66208471 -1.40693001 -0.57932412  1.4896906  -0.16552118]
```

```
# Calculate the z-score from with scipy
import scipy.stats as stats
values = [10, 5, 7, 12, 8]

zscores = stats.zscore(values)
print(zscores)
```

Output

```
[ 0.66208471 -1.40693001 -0.57932412  1.4896906  -0.16552118]
```

```
# stats.zscore() method
import numpy as np
from scipy import stats

arr1 = [[20, 2, 7, 1, 34],
        [50, 12, 12, 34, 4]]

print ("\narr1 : ", arr1)
print ("\nZ-score for arr1 : \n", stats.zscore(arr1))
print ("\nZ-score for arr1 : \n", stats.zscore(arr1, axis= 1))
```

Output

```
arr1 :  [[20, 2, 7, 1, 34], [50, 12, 12, 34, 4]]

Z-score for arr1 :
 [[-1. -1. -1. -1.  1.]
 [ 1.  1.  1.  1. -1.]]

Z-score for arr1 :
 [[ 0.57251144 -0.85876716 -0.46118977 -0.93828264  1.68572813]
 [ 1.62005758 -0.61045648 -0.61045648  0.68089376 -1.08003838]]
```

Z-scores calculation using Pandas Dataframe

```
# Loading a Sample Pandas Dataframe
import pandas as pd

df = pd.DataFrame.from_dict({
```

```
    'Name': ['Nik', 'Kate', 'Joe', 'Mitch', 'Alana'],
    'Age': [32, 30, 67, 34, 20],
    'Income': [80000, 90000, 45000, 23000, 12000],
    'Education' : [5, 7, 3, 4, 4]
})

print(df.head())

df['Income zscore'] = stats.zscore(df['Income'])
print(df.head())
```

Output

	Name	Age	Income	Education
0	Nik	32	80000	5
1	Kate	30	90000	7
2	Joe	67	45000	3
3	Mitch	34	23000	4
4	Alana	20	12000	4

	Name	Age	Income	Education	Income zscore
0	Nik	32	80000	5	0.978700
1	Kate	30	90000	7	1.304934
2	Joe	67	45000	3	-0.163117
3	Mitch	34	23000	4	-0.880830
4	Alana	20	12000	4	-1.239687

MinMax Scaler vs Standard Scaler

```
# import module
from sklearn.preprocessing import
StandardScaler

# create data
data = [[11,2],[3,7],[0,10],[11 8]]

# compute required values
scaler = MinMaxScaler()
model = scaler.fit(data)
scaled_data = model.transform(data)

# print scaled data
print(scaled_data)
```

Output

```
[[1.         0.        ]
 [0.27272727 0.625     ]
 [0.         1.        ]
 [1.         0.75      ]]
```

```
# import module
from sklearn.preprocessing import
StandardScaler

# create data
data = [[11,2],[3,7],[0,10],[11,8]]

# compute required values
scaler = StandardScaler()
model = scaler.fit(data)
scaled_data = model.transform(data)

# print scaled data
print(scaled_data)
```

Output

```
[[ 0.97596444 -1.61155897]
 [-0.66776515  0.08481889]
 [-1.28416374  1.10264561]
 [ 0.97596444  0.42409446]]
```

	import scipy.stats as stats values = data = [[11,2],[3,7],[0,10],[11,8]] zscores = stats.zscore(values) print(zscores) **Output** [[0.97596444 -1.61155897] [-0.66776515 0.08481889] [-1.28416374 1.10264561] [0.97596444 0.42409446]]

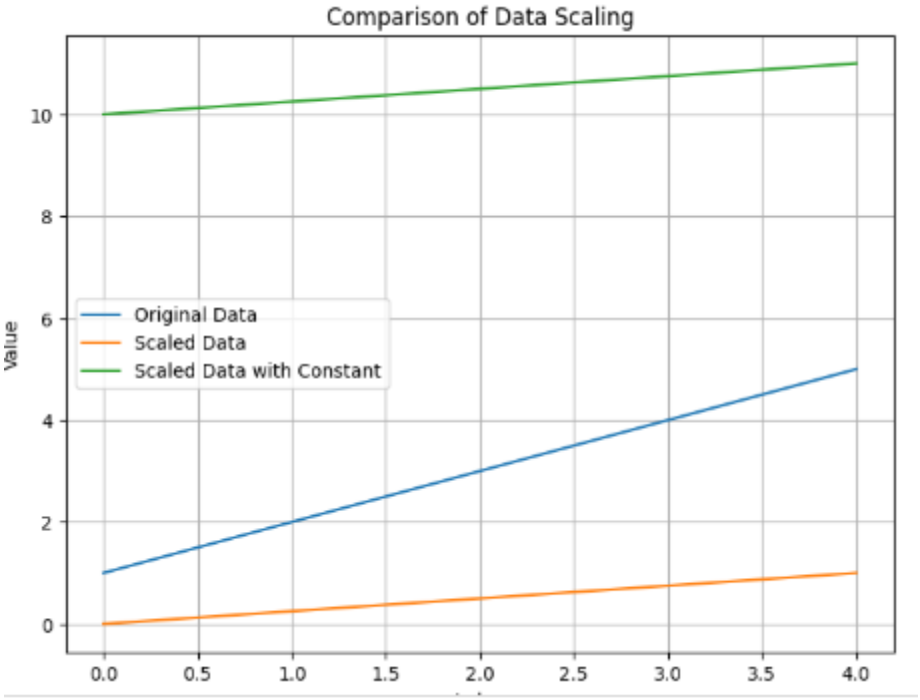
Adding Constant

Scaling refers to the process of transforming the values of a dataset to fit within a specific range. It is commonly used when the features of the dataset have different scales. Two common scaling techniques are adding or subtracting a constant and multiplying or dividing by a constant.

Python program for adding a constant in Scaling

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler

# Original data
original_data = np.array([1, 2, 3, 4, 5])

# Scaling using Min-Max scaling
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(original_data.reshape(-1,
1)).flatten()

# Scaling by adding a constant
constant = 10
scaled_data_with_constant = scaled_data + constant

# Plotting the data
plt.figure(figsize=(8, 6))
plt.plot(original_data, label='Original Data')
plt.plot(scaled_data, label='Scaled Data')
plt.plot(scaled_data_with_constant, label='Scaled Data with
Constant')
plt.legend()
```

```
plt.xlabel('Index')
plt.ylabel('Value')
plt.title('Comparison of Data Scaling')
plt.grid(True)
plt.show()
```



Gaussian distribution

In real life, many datasets can be modeled by Gaussian Distribution (Univariate or Multivariate). So it is quite natural and intuitive to assume that the clusters come from different Gaussian Distributions. Or in other words, it tried to model the dataset as a mixture of several Gaussian Distributions.

A Gaussian distribution, also known as a normal distribution or bell curve, is a probability distribution that is symmetric and characterized by its mean and standard deviation. It is named after the German mathematician, Carl Friedrich Gauss. It is one of the most commonly encountered distributions in statistics and probability theory. Some common example datasets that follow Gaussian distribution are:

- Body temperature
- People's Heights
- Car mileage
- IQ scores

The Gaussian distribution has the following properties:

- *Symmetry*: The distribution is symmetric around its mean, which means that the probability density function (PDF) is symmetric.
- *Mean*: The mean (μ) represents the central value or the average of the distribution.

- *Standard Deviation*: The standard deviation (σ) represents the spread or dispersion of the distribution. It determines how much the values deviate from the mean.
- *Shape:* The Gaussian distribution has a characteristic bell-shaped curve, with the highest point at the mean and the tails extending towards infinity.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

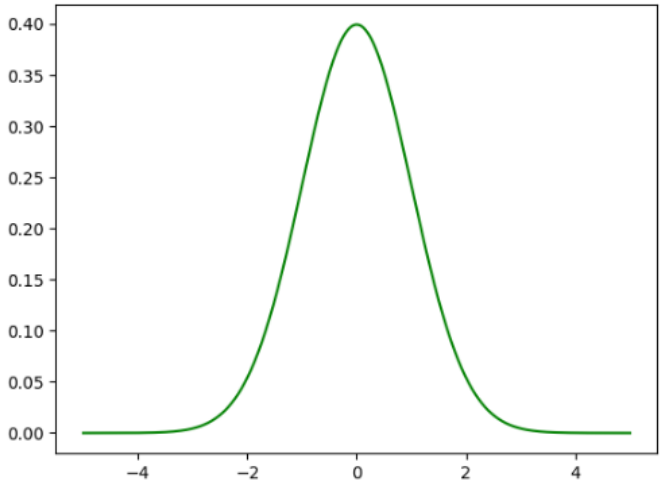
where:

- f(x) is the probability density function at a specific value x.
- μ is the mean of the distribution.
- σ is the standard deviation of the distribution.
- π is a mathematical constant (approximately 3.14159).
- exp() is the exponential function.

Simple python program to explain Gaussian distribution

```
import numpy as np
import scipy as sp
from scipy import stats
import matplotlib.pyplot as plt

## generate the data and plot it for an ideal normal curve

## x-axis for the plot
x_data = np.arange(-5, 5, 0.001)

## y-axis as the gaussian
y_data = stats.norm.pdf(x_data, 0, 1)

## plot data
plt.plot(x_data, y_data,'green')
plt.show()
```

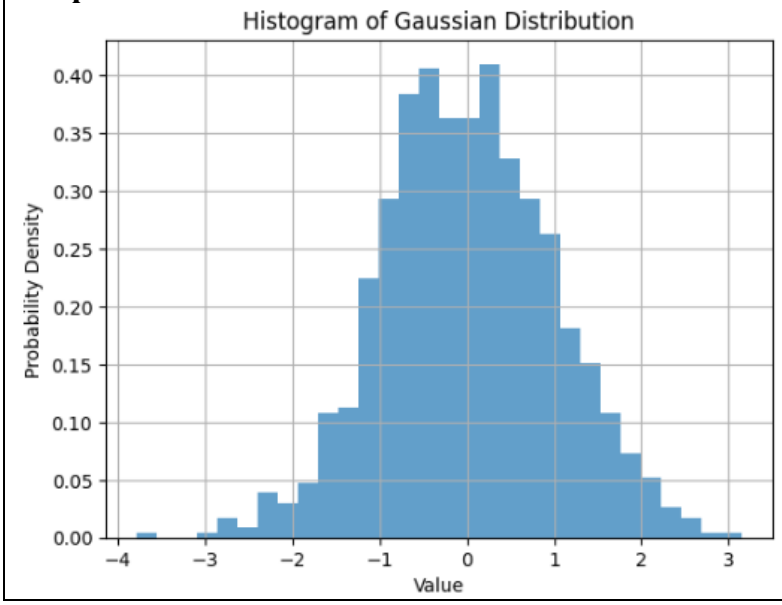
Output



Simple python program to explain Histogram of Gaussian distribution

```
import numpy as np
import matplotlib.pyplot as plt

# Generate random numbers from a Gaussian distribution
mean = 0  # Mean of the distribution
std_dev = 1  # Standard deviation of the distribution
sample_size = 1000  # Number of samples to generate

samples = np.random.normal(mean, std_dev, sample_size)

# Plot a histogram of the generated samples
plt.hist(samples, bins=30, density=True, alpha=0.7)
plt.xlabel('Value')
plt.ylabel('Probability Density')
plt.title('Histogram of Gaussian Distribution')
plt.grid(True)
plt.show()
```

Output



Inequality

Income inequality using quantiles and quantile-shares

Inequality analysis involves examining the distribution of a variable and exploring the disparities or differences in values across different groups or segments of the data. Quantiles and quantile shares are useful measures for analyzing inequality. Here's how they can be used:

- *Quantiles:*
  Quantiles divide a dataset into equal-sized groups, representing specific percentiles of the data. For example, the median represents the 50th percentile, dividing the data into two equal halves. Other quantiles, such as quartiles (25th, 50th, and 75th percentiles), quintiles (20th, 40th, 60th, and 80th percentiles), or deciles (10th, 20th, ..., 90th percentiles), divide the data into smaller segments.

- *Quantile Shares*:
  Quantile shares represent the cumulative proportion of a variable's distribution held by each quantile group. They help us in understanding the concentration or dispersion of values across different parts of the distribution. For example, if the top 10% of earners in a population hold 50% of the total income, it indicates a higher level of income concentration.

To illustrate these concepts, let's consider an example of analyzing income inequality using quantiles and quantile shares:

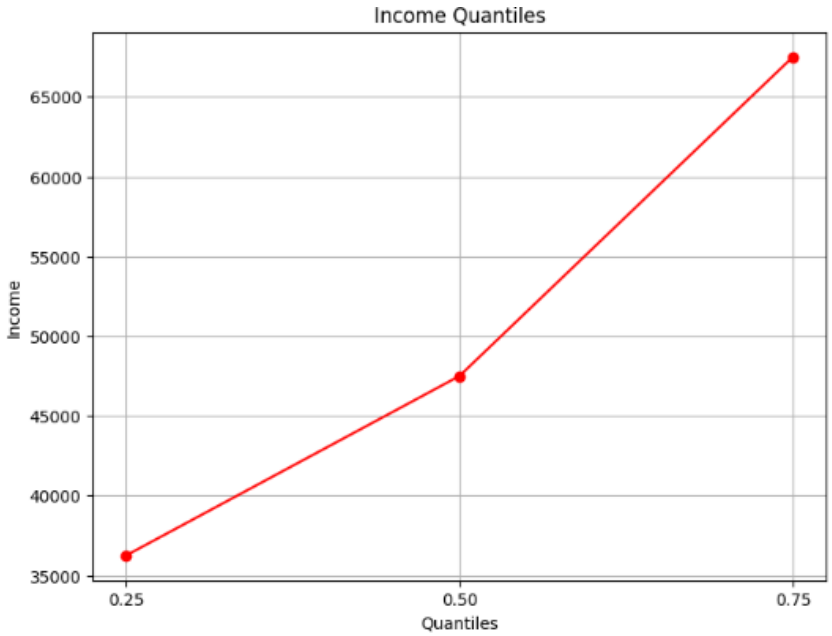
Simple python program to explain Income inequality using quantiles and quantile-shares

```
import pandas as pd
import matplotlib.pyplot as plt

# Example dataset with income values
data = pd.DataFrame({'Income': [25000, 30000, 35000, 40000,
45000, 50000, 60000, 70000, 80000, 100000]})

# Calculate quantiles and quantile shares
quantiles = [0.25, 0.5, 0.75]
quantile_values = data['Income'].quantile(quantiles)
quantile_shares = data['Income'].quantile(quantiles) /
data['Income'].sum()

# Print quantiles and quantile shares
print("Quantiles:")
print(quantile_values)
print("\nQuantile Shares:")
print(quantile_shares)
```

```
# Plotting quantile values
plt.figure(figsize=(8, 6))
plt.plot(quantiles, quantile_values, marker='o', linestyle='-
', color='red')
plt.xlabel('Quantiles')
plt.ylabel('Income')
plt.title('Income Quantiles')
plt.xticks(quantiles)
plt.grid(True)
plt.show()
```

Output

```
Quantiles:
0.25    36250.0
0.50    47500.0
0.75    67500.0
Name: Income, dtype: float64

Quantile Shares:
0.25    0.067757
0.50    0.088785
0.75    0.126168
Name: Income, dtype: float64
```



Income inequality using Cumulative income shares and Lorenz curves

Cumulative income shares and Lorenz curves are concepts used to analyze income inequality. Here's an explanation of each:

- *Cumulative Income Shares:*

Cumulative income shares represent the cumulative proportion of total income held by different segments or groups of the population. It helps us understand the distribution of income across various portions of the population.

  - To calculate cumulative income shares, we sort the income values in ascending order and calculate the cumulative sum of sorted incomes divided by the sum of all incomes. This gives us the proportion of total income accumulated by each portion of the population as we move from the lowest to the highest incomes.

- Lorenz Curve:
  The Lorenz curve is a graphical representation of income inequality. It plots the cumulative income shares on the y-axis against the cumulative population shares on the x-axis. The Lorenz curve helps visualize how income is distributed in relation to the population.
  - The equality line is also plotted on the same graph, which represents perfect income equality. It is a diagonal line starting from the origin (0,0) and ending at (1,1). The Lorenz curve shows how the actual distribution of income deviates from the equality line.

Analysis

- The further the Lorenz curve is from the equality line, the greater the income inequality. If the Lorenz curve lies below the equality line, it indicates income concentration among a smaller portion of the population. If it lies above the equality line, it indicates income dispersion among a larger portion of the population.
- The area between the Lorenz curve and the equality line represents income inequality. A larger area indicates higher income inequality, while a smaller area indicates lower inequality.

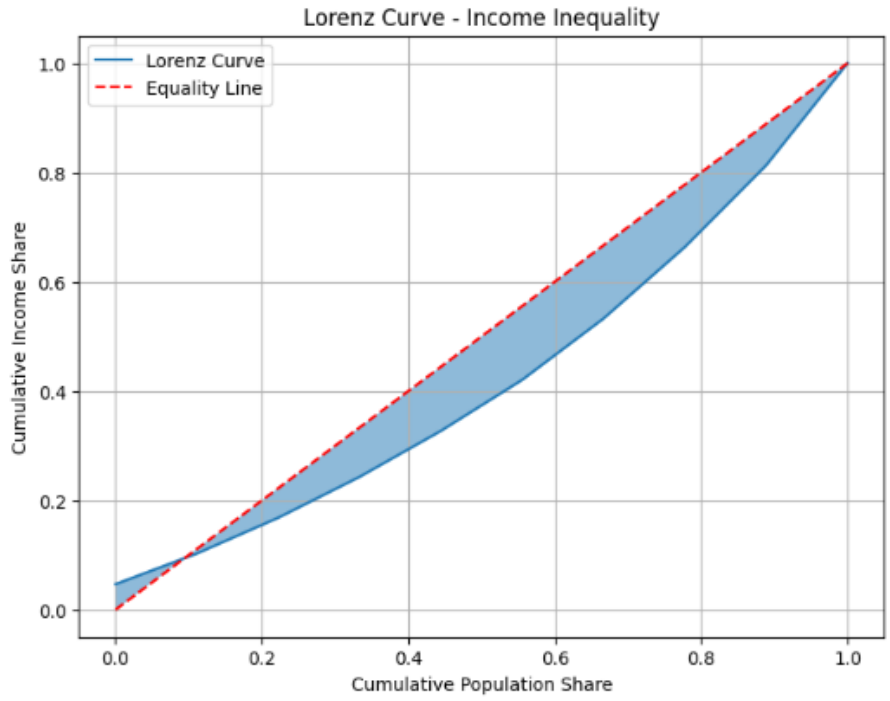
By plotting the Lorenz curve and comparing it to the equality line, we can visually assess the level of income inequality in a population. The Lorenz curve provides a comprehensive overview of income distribution, allowing us to analyze disparities and evaluate the fairness and equity of income allocation.

Simple python program to explain Income inequality using Cumulative income shares and Lorenz curves

```
# Example dataset with income values
income = np.array([25000, 30000, 35000, 40000, 45000, 50000, 60000, 70000,
80000, 100000])

# Sort the income values in ascending order
sorted_income = np.sort(income)

# Calculate cumulative income shares
```

```
cumulative_income_shares = np.cumsum(sorted_income) / np.sum(sorted_income)

# Calculate cumulative population shares
cumulative_population_shares = np.linspace(0,1, len(sorted_income))

# Calculate diagonal line (equality line)
equality_line = np.linspace(0, 1, len(sorted_income))

# Plotting Lorenz curve
plt.figure(figsize=(8, 6))
plt.plot(cumulative_population_shares,cumulative_income_shares,
label='Lorenz Curve')
plt.plot(cumulative_population_shares,    equality_line,    label='Equality
Line', linestyle='--', color='red')
plt.fill_between(cumulative_population_shares,    cumulative_income_shares,
equality_line, alpha=0.5)
plt.xlabel('Cumulative Population Share')
plt.ylabel('Cumulative Income Share')
plt.title('Lorenz Curve - Income Inequality')
plt.legend()
plt.grid(True)
plt.show()
```

Output



Gini coefficient

The Gini coefficient is a widely used measure of income or wealth inequality. It quantifies the extent of income inequality in a population, ranging from 0 to 1, where 0 represents perfect equality (all individuals have the same income) and 1 represents maximum inequality (one individual has all the income, while others have none).

The Gini coefficient is derived from the Lorenz curve, which plots the cumulative income shares against the cumulative population shares. To calculate the Gini coefficient, you can follow these steps:

- Obtain the cumulative population shares and cumulative income shares from the Lorenz curve. These represent the x-axis and y-axis values, respectively.
- Calculate the area between the Lorenz curve and the equality line (the diagonal line). This area is known as the "area between the Lorenz curve and the line of perfect equality."
- Calculate the area under the equality line (the area of the triangle formed by the equality line and the two axes).
- Divide the "area between the Lorenz curve and the line of perfect equality" by the "area under the equality line" to get the Gini coefficient.
- Mathematically, the formula for the Gini coefficient can be expressed as:

$$G = (A) / (A + B)$$

where:

G: Gini coefficient

A: Area between the Lorenz curve and the line of perfect equality

B: Area under the equality line

- The Gini coefficient ranges from 0 to 1, with higher values indicating higher income inequality.

Simple python program to explain Income inequality using Gini coefficient

```
import numpy as np

def calculate_gini_coefficient(income):
    # Sort the income values in ascending order
    sorted_income = np.sort(income)

    # Get the cumulative population shares
    cumulative_population_shares = np.arange(1, len(sorted_income) + 1) /
len(sorted_income)

    # Calculate the cumulative income shares
    cumulative_income_shares = np.cumsum(sorted_income)/np.sum(sorted_income)

    # Calculate the area between Lorenz curve and line of income equality
    area_between_curve_and_equality = np.sum(cumulative_income_shares[:-1] *
(cumulative_population_shares[1:] - cumulative_population_shares[:-1]))

    # Calculate the area under the line of perfect income equality
    area_under_equality = cumulative_population_shares[-1]

    # Calculate the Gini coefficient
    gini_coefficient= 1-(2*area_between_curve_and_equality/area_under_equality)

    return gini_coefficient

# Example income distribution
income = np.array([1000, 2000, 3000, 4000, 5000])
```

```
# Calculate the Gini coefficient
gini = calculate_gini_coefficient(income)

# Print the result
print("Gini coefficient:", gini)
```

Output

```
Gini coefficient: 0.4666666666666667
```

Summary Measure of Inequality

When evaluating a summary measure of inequality, there are several desirable properties to consider. Here are some of the key properties that are often desired:

- Scale invariance: The measure should not change when the units of measurement are transformed or scaled. It should provide consistent results regardless of the scale of the variable being measured.
- Anonymity: The measure should not be affected by changes in the identity or characteristics of individuals within the population. It should only depend on the income or wealth distribution itself.
- Population size independence: The measure should not be sensitive to changes in the total population size. It should provide meaningful comparisons even when the population size varies.
- Transfer principle: The measure should reflect changes in inequality when income or wealth is redistributed among individuals. If there is a transfer of resources from richer to poorer individuals, the measure should indicate a decrease in inequality.
- Decomposability: The measure should allow for the decomposition of inequality into different components, such as within-group and between-group inequality. This property helps in understanding the sources and drivers of inequality.
- Mathematical properties: The measure should possess well-defined mathematical properties, such as continuity, differentiability, and boundedness. These properties ensure that the measure is mathematically tractable and interpretable.
- Intuitiveness and interpretability: The measure should be easy to understand and interpret intuitively. It should provide meaningful insights into the distribution of income or wealth and the level of inequality.

It's important to note that no single measure can fully satisfy all of these properties simultaneously. Different measures of inequality emphasize different aspects and trade-offs. Therefore, it is often recommended to consider multiple measures and interpret the results collectively to gain a comprehensive understanding of inequality.