Statistical Analysis Implementation in Python

Part 1: Descriptive Statistics

Created By : Felice Benita

DATA TABLES

During an experiment, data is typically collected and organized using data tables with the independent variable on the left side and the dependent variable on the right side -- with units included.

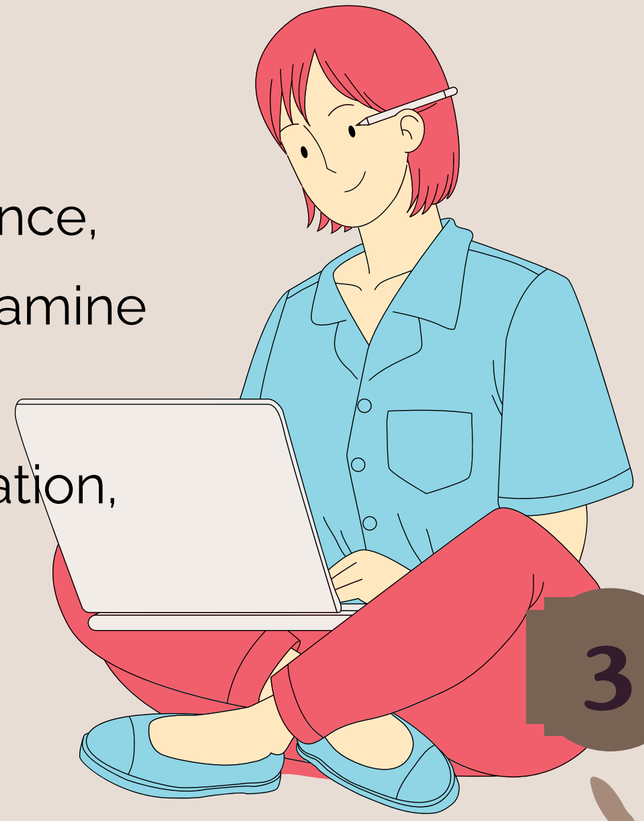
DESCRIPTIVE STATISTICS

While data tables are great for organizing data, they are not always easy to interpret, especially when it comes to large data sets. This is why we use statistics to help us understand data.
There are three basic statistics: mean, median, and mode. These values help to reveal what is 'typical' in a data set.
The other descriptive statistics are: variance, standard deviation, range, percentiles, and summary statistics.

EXAMPLE

Using dataset of **E-commerce Product Sales** with data columns:
Product_ID, Product_Category, Units_Sold, Revenue,
Discount_Percentage, Return_Rate, Customer_Rating,
Days_in_Inventory, Sales_Channel (online, in-store).
This dataset will allow you to explore trends in sales performance, analyze revenue and return rates by product category, and examine customer satisfaction metrics.

→ Find the mean, median, and mode, variance, standard deviation, range, percentiles, and summary statistics of this data set.



EXAMPLE

```
[2]: # Reading data
df = pd.read_csv('File Dirr/ecommerce_product_sales.csv')
df.head()
```

[2]:

	Product_ID	Product_Category	Units_Sold	Revenue	Discount_Percentage	Return_Rate	Customer_Rating	Days_in_Inventory	Sales_Channel
0	P0001	Beauty	39	207.04	50	0.12	2.7	91	In-Store
1	P0002	Books	44	12.62	20	0.04	2.9	141	In-Store
2	P0003	Home & Kitchen	43	293.55	25	0.20	4.7	289	Online
3	P0004	Books	38	130.29	0	0.22	3.6	218	Online
4	P0005	Books	40	227.88	15	0.17	2.2	77	Online

Here's the full code: Click Here

CALCULATING MEAN

A **mean** is another word for an average. To determine the average of a set of values, first, find the sum of all the numbers. Then, divide the sum by the total number of values in the set.

Example: (a + b + c) / 3

-- Code in Python

```
[6]: # Calculate the mean of each relevant numerical column
     # (this time, we only use Units_Sold, Revenue, and Customer_Rating column)
     mean_units_sold = df["Units_Sold"].mean()
     mean_revenue = df["Revenue"].mean()
     mean_customer_rating = df["Customer_Rating"].mean()

[7]: # Display the results
     print("Mean Units Sold:", round(mean_units_sold,2))
     print("Mean Revenue:", round(mean_revenue,2))
     print("Mean Customer Rating:", round(mean_customer_rating,2))

     Mean Units Sold: 49.89
     Mean Revenue: 249.15
     Mean Customer Rating: 2.99
```

CALCULATING MEDIAN

A **median** is the middle value in a set of numbers. To find the median, arrange all of the values in order from lowest to highest. Then, find the middle value. If there is no single middle number, you can find the average of the two middle numbers.

Example: a b c d e

-- Code in Python

[8]:
```
# Calculate the median of Units_Sold, Revenue, and Customer_Rating
median_units_sold = df["Units_Sold"].median()
median_revenue = df["Revenue"].median()
median_customer_rating = df["Customer_Rating"].median()
```

[9]:
```
# Display the results
print("Median Units Sold:", round(median_units_sold,2))
print("Median Revenue:", round(median_revenue,2))
print("Median Customer Rating:", round(median_customer_rating,2))
```

```
Median Units Sold: 50.0
Median Revenue: 246.34
Median Customer Rating: 3.0
```

CALCULATING MODE

A **mode** is a value that occurs most often in a data set. To find the mode, simply look for the number that is repeated the most. Some data sets do not have a mode, whereas others have more than one mode.

Example: a a b c d

-- Code in Python

[10]:
```
# Calculate the mode of Units_Sold, Revenue, and Customer_Rating
mode_units_sold = df["Units_Sold"].mode()[0]  # [0] to get the first mode in case of multiple
mode_revenue = df["Revenue"].mode()[0]
mode_customer_rating = df["Customer_Rating"].mode()[0]
```

[11]:
```
# Display the results
print("Mode Units Sold:", mode_units_sold)
print("Mode Revenue:", mode_revenue)
print("Mode Customer Rating:", mode_customer_rating)
```

```
Mode Units Sold: 52
Mode Revenue: 113.26
Mode Customer Rating: 1.8
```

CALCULATING VARIANCE

A **variance** is a statistical measure that tells us how much the values in a data set differ from the mean (average) of that data set. It gives an idea of the spread or dispersion of the data points. When the variance is small, the data points are close to the mean, and when the variance is large, the data points are more spread out from the mean.

$$\sigma^2 = \frac{\sum(x - \mu)^2}{N}$$

- x represents each data point,
- μ is the mean of the data set, and
- N is the number of data points.

-- Code in Python

```
[12]: # Calculate the variance of Units_Sold, Revenue, and Customer_Rating
      variance_units_sold = df["Units_Sold"].var()
      variance_revenue = df["Revenue"].var()
      variance_customer_rating = df["Customer_Rating"].var()

[13]: # Display the results
      print("Variance of Units Sold:", round(variance_units_sold,2))
      print("Variance of Revenue:", round(variance_revenue,2))
      print("Variance of Customer Rating:", round(variance_customer_rating,2))

      Variance of Units Sold: 51.62
      Variance of Revenue: 20187.16
      Variance of Customer Rating: 1.36
```

CALCULATING STANDARD DEVIATION

A **standard deviation** is the square root of the variance that shows the extent of variation or dispersion of a set of values relative to its mean.

$$\sigma = \sqrt{\frac{\sum(x - \mu)^2}{N}}$$

- x represents each data point,
- μ is the mean of the data set, and
- N is the number of data points.

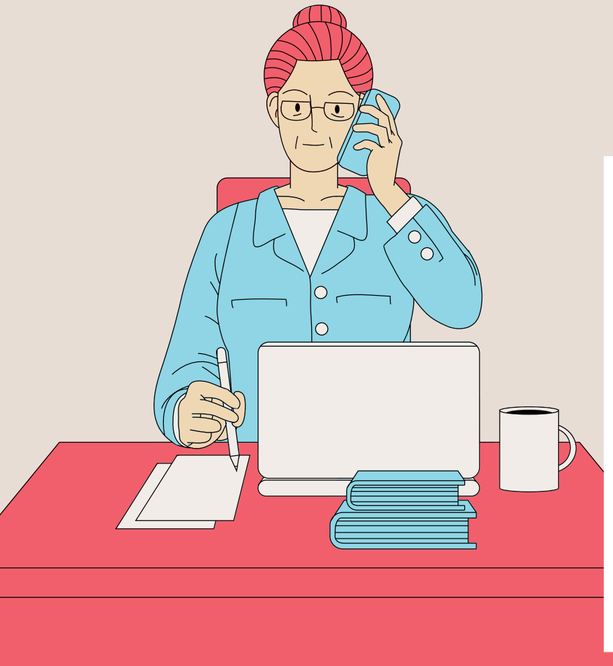
-- Code in Python

```
[14]: # Calculate the standard deviation of Units_Sold, Revenue, and Customer_Rating
std_units_sold = df["Units_Sold"].std()
std_revenue = df["Revenue"].std()
std_customer_rating = df["Customer_Rating"].std()

[15]: # Display the results
print("Standard Deviation of Units Sold:", round(std_units_sold,2))
print("Standard Deviation of Revenue:", round(std_revenue,2))
print("Standard Deviation of Customer Rating:", round(std_customer_rating,2))

Standard Deviation of Units Sold: 7.18
Standard Deviation of Revenue: 142.08
Standard Deviation of Customer Rating: 1.17
```

CALCULATING RANGE

The **range** is a simple measure of dispersion in a data set, showing the difference between the largest and smallest values. It gives a quick sense of how spread out the values are, but it doesn't provide information about how the values are distributed between those two points.

Range = Maximum value – Minimum value



-- Code in Python

[16]:
```
# Calculate the range (max - min) for Units_Sold, Revenue, and Customer_Rating column, rounded to 2 decimal places
range_units_sold = df["Units_Sold"].max() - df["Units_Sold"].min()
range_revenue = df["Revenue"].max() - df["Revenue"].min()
range_customer_rating = df["Customer_Rating"].max() - df["Customer_Rating"].min()
```

[17]:
```
# Display the results
print("Range of Units Sold:", round(range_units_sold,2))
print("Range of Revenue:", round(range_revenue,2))
print("Range of Customer Rating:", round(range_customer_rating,2))
```

```
Range of Units Sold: 42
Range of Revenue: 494.46
Range of Customer Rating: 4.0
```

CALCULATING PERCENTILES

A **percentiles** are values that divide a data set into 100 equal parts, helping to understand how a particular data point compares to the rest of the data set. A percentile indicates the relative standing of a value within the data set. For example, the 50th percentile (also called the median) is the value at which 50% of the data points are less than or equal to that value.

-- Code in Python

[18]:
```python
# Define the percentiles we want to calculate
percentiles = [0.25, 0.5, 0.75]

# Calculate percentiles for Units_Sold, Revenue, and Customer_Rating colum
percentiles_units_sold = df["Units_Sold"].quantile(percentiles).round(2)
percentiles_revenue = df["Revenue"].quantile(percentiles).round(2)
percentiles_customer_rating = df["Customer_Rating"].quantile(percentiles).round(2)
```

[19]:
```python
# Display the results
print("Percentiles for Units Sold:\n", percentiles_units_sold)
print("Percentiles for Revenue:\n", percentiles_revenue)
print("Percentiles for Customer Rating:\n", percentiles_customer_rating)
```

```
Percentiles for Units Sold:
 0.25    45.0
0.50    50.0
0.75    55.0
Name: Units_Sold, dtype: float64
Percentiles for Revenue:
 0.25    124.42
0.50    246.34
0.75    374.94
Name: Revenue, dtype: float64
Percentiles for Customer Rating:
```

CALCULATING SUMMARY STATISTICS

Summary statistics are key numbers that describe and summarize the main characteristics of a data set. They provide a quick overview of data distribution, central tendency, and variability, helping analysts understand data patterns at a glance. Common summary statistics include measures of central tendency (such as mean, median, and mode), measures of variability (such as range, variance, and standard deviation), and measures of distribution (such as percentiles).

-- Code in Python

```
[20]: # Calculate summary statistics for each numerical column
summary_statistics = df.describe().round(2)

# Display the results
print("Summary Statistics:\n", summary_statistics)
```

```
Summary Statistics:
       Units_Sold  Revenue  Discount_Percentage  Return_Rate  Customer_Rating  \
count     1000.00  1000.00              1000.00      1000.00          1000.00
mean        49.89   249.15                19.24         0.15             2.99
std          7.18   142.08                14.46         0.09             1.17
min         30.00     5.32                 0.00         0.00             1.00
25%         45.00   124.42                10.00         0.08             2.00
50%         50.00   246.34                17.50         0.14             3.00
75%         55.00   374.94                25.00         0.22             4.00
max         72.00   499.78                50.00         0.30             5.00
```

CONCLUSION

In summary, descriptive statistics are essential for obtaining a quick overview of data sets, allowing researchers and analysts to draw initial insights about the data's central tendency, variability, and distribution. They serve as a foundation for further statistical analysis and hypothesis testing, enabling a more comprehensive understanding of the underlying data patterns and trends.

Created by Felice Benita

THANK YOU

Created By : Felice Benita