DATA SCIENCE INDONESIA

bukalapak

Data Preprocessing and Cleansing

Vito Rihaldijiran
Business Intelligence Analyst at Bukalapak

INTRODUCTION

Name: Vito Rihaldijiran
Place and Date of Birth: Jakarta, 17 September 2001
Almamater: Institut Teknologi Sepuluh Nopember
Current Activities:

- Business Intelligence Analyst at Bukalapak
- Knowledge Management at Data Science Indonesia

Connect Me on:

- Linkedin: Vito Rihaldijiran
- Instagram: @vitorihaldijiran

WHAT ARE WE GONNA TALK

01

BASIC STATISTICS

Descriptive Statistics (Mean, Median, Mode, Quartile)

02

HANDLING MISSING & DUPLICATED VALUES

Drop, Imputation, Keep

03

HANDLING OUTLIERS

Box Plot, IQR, Z-score

04

DATA TRANSFORMATION

Normalization, Standardization, Feature Encoding

HANDS-ON REQUIRED

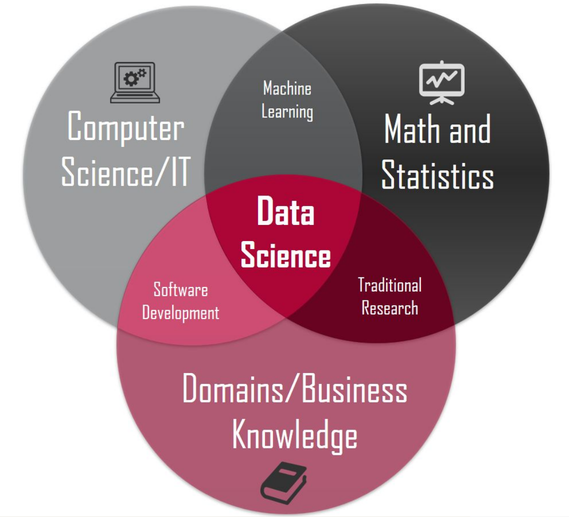
Dataset:
https://www.kaggle.com/datasets/carrie1/ecommerce-data

Tools: Google Colab/Jupyter Notebook

01

BASIC STATISTICS

Descriptive Statistics (Mean, Median, Mode, Quartile)


Computer Science/IT
Machine Learning
Math and Statistics
Data Science
Software Development
Traditional Research
Domains/Business Knowledge

THE QUESTIONS THAT CAN BE ANSWER WITH STATISTICS

What is the proportion of Indonesian citizens over 17 years of age?

Has a product experienced an increase or decrease in performance? What caused it?

DEFINITION

General Definition & Its Types

"Branch of Applied Mathematics that involves the collection, description, analysis, and inference of conclusions from quantitative data."

There are 2 types of statistics:

1. **Descriptive Statistics** (mean, median, mode, quartile) → **our discussion for today**
2. **Inferential Statistics**

DESCRIPTIVE STATISTICS

What & How

"A descriptive statistic is a summary statistic that quantitatively describes or summarizes features from a collection of information, while descriptive statistics is the process of using and analysing those statistics."

How we see the data through descriptive statistics?

- **Mean**
- **Median**
- **Mode**
- **Quartile**

MEAN, MEDIAN, MODE, and QUARTILE

Definition

Mean

- The average of the given numbers and is calculated by dividing the sum of given numbers by the total number of numbers
- Formula = ***Sum of All Data Points : Number of Data Points***

$$\text{Mean } (x) = \frac{\Sigma x}{n}$$

Median

- The middle number in a sorted, ascending or descending list of numbers
- Formula =

Median

n is odd,

$$\text{Median} = \left(\frac{n+1}{2}\right)^{th} \text{ observation}$$

n is even,

$$\text{Median} = \frac{\left(\frac{n}{2}\right)^{th} + \left(\frac{n}{2}+1\right)^{th} \text{ observation}}{2}$$

MEAN, MEDIAN, MODE, and QUARTILE

Definition

Mode

A number in a set of numbers that appears the most often

Quartile

- A type of quantile which divides the number of data points into four parts, or quarters, of more-or-less equal size. The data must be ordered from smallest to largest to compute quartiles; as such, quartiles are a form of order statistic.
- Type of Quartile
    - **Quartile 1 (Q1)** : falls in 25% of data
    - **Quartile 2 (Q2)** : median
    - **Quartile 3 (Q3)** : falls in 75% of data
    - **Quartile 4 (Q4)** : falls in 100% of data

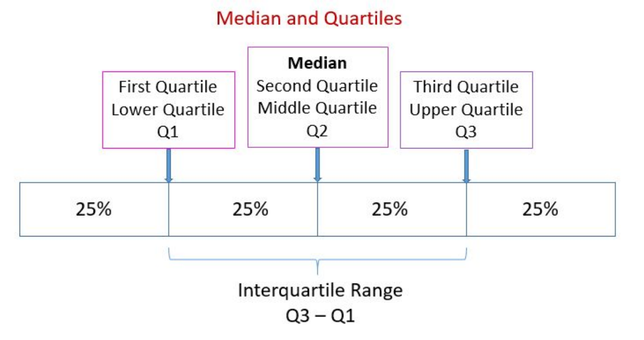
Quartile Formula

Lower Quartile (Q1) = $(N+1) \times \frac{1}{4}$

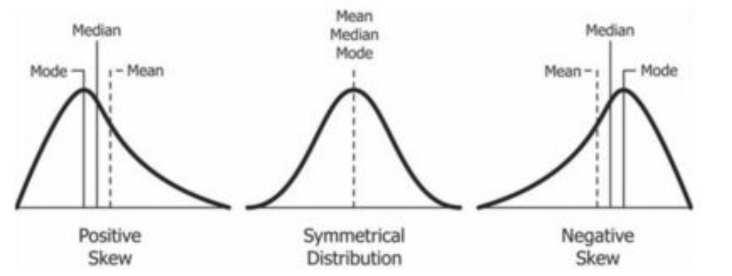
Middle Quartile (Q2) = $(N+1) \times \frac{2}{4}$

Upper Quartile (Q3) = $(N+1) \times \frac{3}{4}$

SNEAK PEAK



IQR



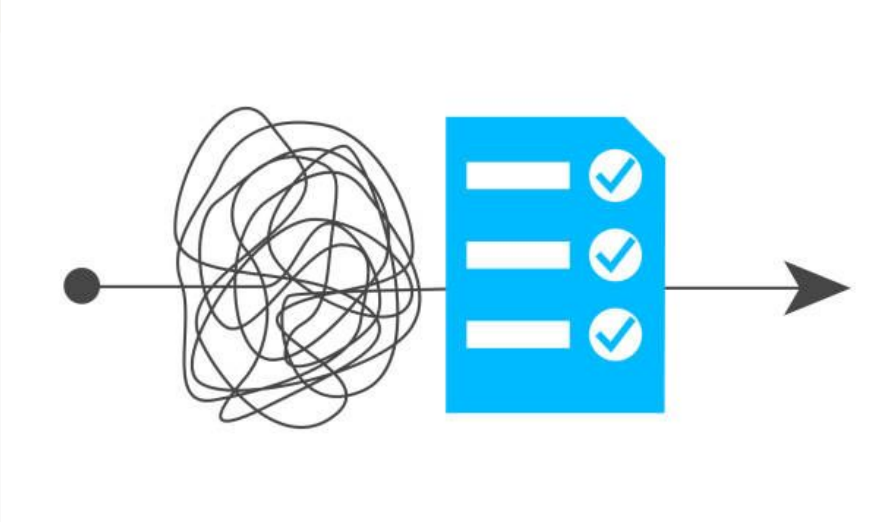
SKEWNESS

```python
# Import Library
import numpy as np
import scipy as sci

#Make a List
data = [99,86,87,88,111,86,103,87,94,78,77,85,86]

#state the operation
mean = np.mean(data)
median = np.median(data)
mode = sci.mode(data)
quartile = np.quantile(data, [0,0.25,0.5,0.75,1])
```

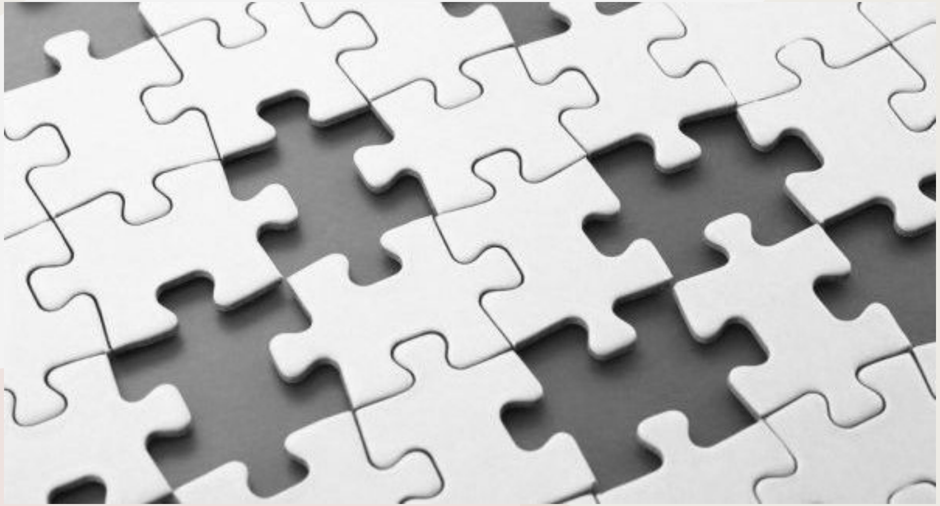
02

HANDLING MISSING & DUPLICATED VALUES

Drop, Imputation, Keep



DOES IT IMPORTANT?



HANDLING EMPTY VALUES (1st case)

"My Dataset has few Empty Values and it doesn't effect too much"

Suggestion: **Drop Empty Values**

```
#1st solution
df= df.dropna()

#2nd solution
df = df.dropna(subset= ['column_a', 'column_b'])

#3rd solution
df.dropna(inplace = True)

#4th solution
df.dropna(subset = ['column_a', 'column_b'], inplace = True)
```

- Subset: only remove empty values on selected columns
- Inplace: It can be True or False. It doesn't return the new dataset, but directly remove the old one empty values

HANDLING EMPTY VALUES (2nd case)

“My Dataset has quite many Numerical Empty Values”

Suggestion: **Numerical Imputation**

```
#1st solution
df= df.fillna(df.mean())

#2nd solution
df= df.fillna(df.min())

#3rd solution
df= df.fillna(df.max())

#4th solution
df = df.fillna(0)
```

Before you decide which solution to be implement in your imputation, please consider the following aspects:

1. **Which imputation method is the most suitable?**
2. **Which imputation method will be produce the most robust model?**

Common solutions for this case:

1. Impute with mean
2. Impute with maximal value in the dataset
3. Impute with minimal value in the dataset
4. Input constant value

HANDLING EMPTY VALUES (3rd case)

"My Dataset has quite many Categorical Empty Values"

Suggestion: **Categorical Imputation**

```
#1st solution
df['column_a']= df['column_a'].fillna('Not Empty Anymore')

#2nd solution
df['column_a']= df['column_a'].fillna(df['column_a'].mode()[0])
```

Common solutions for this case:
- Input constant value
- Input Mode Value in the column

HANDLING DUPLICATE VALUES

"I have many duplicate values"

Suggestion: **Depends, Just Drop or Keep**

```
#check duplicates
df.duplicates().sum()

#Drop Duplicates
df = df.drop_duplicates(subset = ['column_a', 'column_b'])
df.duplicates(inplace = True)

#Parameter Keep
df.drop_duplicates(keep = 'first')
df.drop_duplicates(keep = 'last')
df.drop_duplicates(keep = False)
```

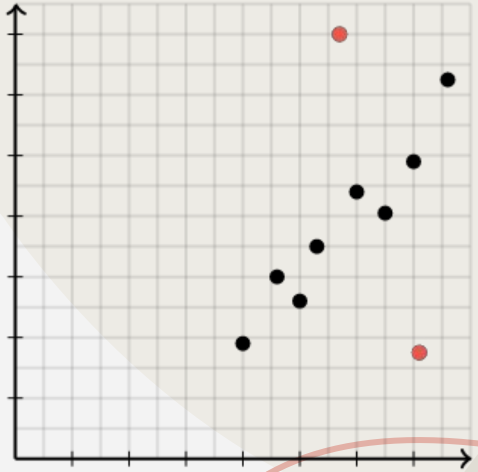
Drop if your duplicate values don't affect the future analysis

Keep if you want to keep certain values in your dataset (eg: keep = 'false' is the same like drop)

03

HANDLING OUTLIERS

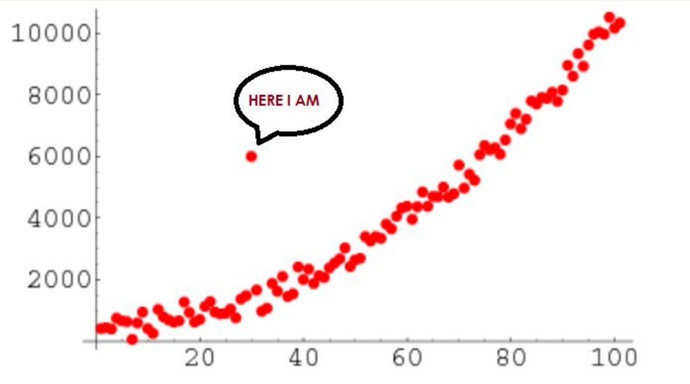
Box Plot, IQR, Z-score



OUTLIERS

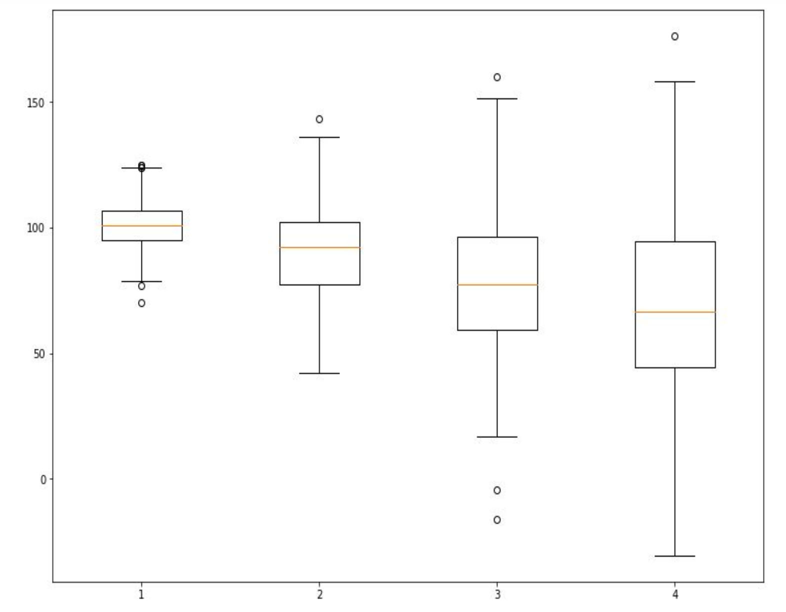
What is that? Is it Important?

In statistics, an outlier is a data point that differs significantly from other observations. An outlier may be due to a variability in the measurement, an indication of novel data, or it may be the result of experimental error; the latter are sometimes excluded from the data set.



DETECTING OUTLIERS

"Boxplot"

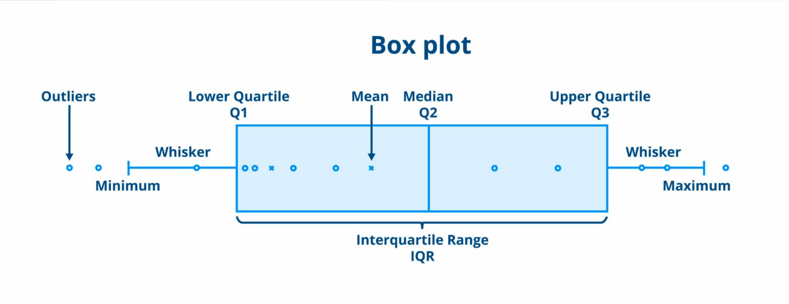


In descriptive statistics, a box plot or box plot is a method for graphically demonstrating the locality, spread and skewness groups of numerical data through their quartiles.

```python
# Import libraries
import matplotlib.pyplot as plt
import numpy as np


# Creating dataset
np.random.seed(10)
data = np.random.normal(100, 20, 200)

fig = plt.figure(figsize =(10, 7))

# Creating plot
plt.boxplot(data)

# show plot
plt.show()
```

HANDLING OUTLIERS: IQR

"Interquartile Range"




```
Q1 = df['column_a'].quantile(0.25)
Q3 = df['column_a'].quantile(0.75)
IQR = Q3 - Q1

low_limit = Q1 - (1.5 * IQR)
high_limit = Q3 + (1.5 * IQR)

filtered = ((df['column_a'] >= low_limit) & (df['column_a'] <= high_limit))

df = df[filtered]
```

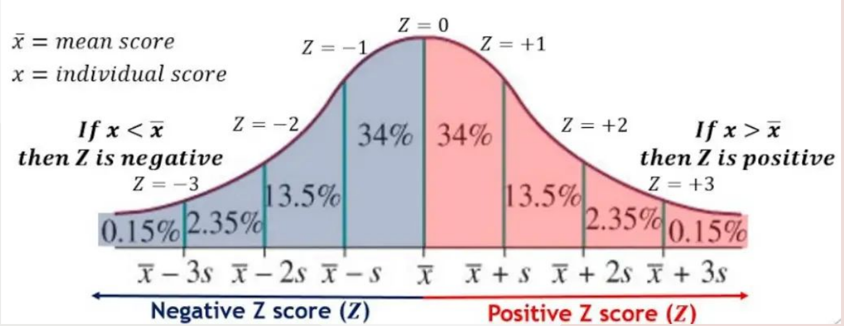
HANDLING OUTLIERS: Z-SCORE

"Z Score"

Definition: Number of standard deviations from the mean of a normal distribution.

How to Measure Outlier: *abs(z_score) > 3*

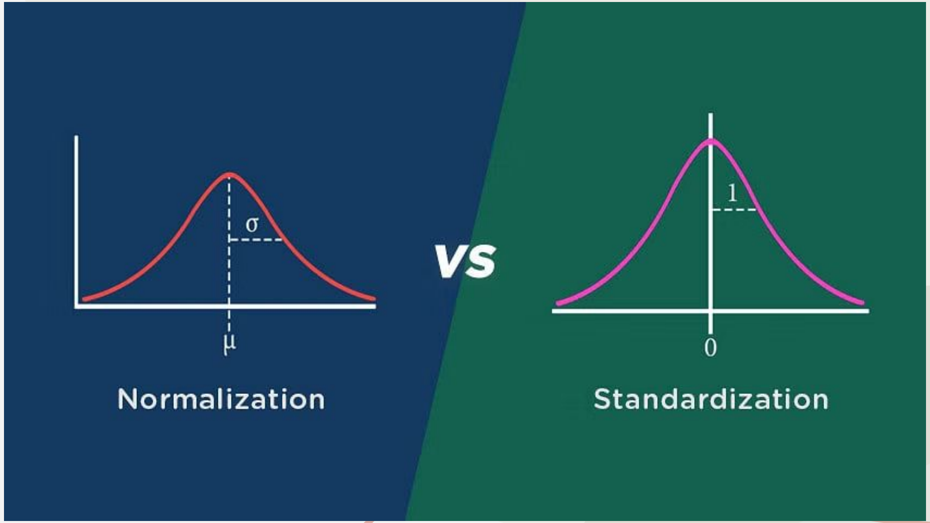
```python
from scipy import stats

z_score = np.abs(stats.zscore(df['column_a']))
filtered = z_score > 3
df = df[filtered]
```



04

DATA TRANSFORMATION

Normalization, Standardization, Feature Encoding



NORMALIZATION VS STANDARDIZATION

Definition

Normalization: "The process of changing the values of a feature to a certain scale, and does not change the data distribution."

Standardization: "the process of changing feature values so that the mean = 0 and standard deviation = 1 & change the data distribution closely to normal distribution"

WHY DO WE HAVE TO DO THAT?

Because…

1. Data with the same scale will ensure that the learning algorithm treats all features fairly
2. Data with the same scale and centered will speed up the learning algorithm (training model)
3. Data at the same scale makes it easier to interpret multiple ML models

MinMaxScaler & StandardScaler

How-to

We will do the normalization with MinMaxScaler and standardization with StandardScaler from scikit-learn library.

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler

df['column_norm'] = MinMaxScaler().fit_transform(df['column_a'].values.reshape(len(df), 1))
df['column_std'] = StandardScaler().fit_transform(df['column_a'].values.reshape(len(df), 1))
```

One Hot Encoding

Definition

One hot encoding is a technique used to **represent categorical variables as numerical values in a machine learning model**. The advantages of using one hot encoding include:

1. It allows the use of categorical variables in models that require numerical input.
2. It can improve model performance by providing more information to the model about the categorical variable.
3. It can help to avoid the problem of ordinality, which can occur when a categorical variable has a natural ordering (e.g. “small”, “medium”, “large”).

```
pd.get_dummies(df['column_name'], prefix = 'column_name')
```

HANDS-ON

Thank You