Basic Data Mining with Python

pradeep

What is Data Mining?

> Data Mining is a process of extracting insights from data.

Technology/Tools

Programming Language

- Python, R

Software

- Weka, RapidMiner, Excel

Cloud

- R Studio Cloud, Power BI, Tableau, Google Collab

What problem to solve?

Classification - Supervised Learning

Clustering - Unsupervised

Methodology

Apa tu

What is Data?

Python

What

- High-Level Programming Language.
- Emphasizes on code readibilty.
- Rank = 1* for 2021 *(IEEE Spectrum)
- Consist of fantastic libraries!

Python Libraries

A Python library is a collection of related modules. It makes Python Programming simpler and convenient for the programmer.

```
# Pandas
import pandas as pd
```

Reading Data

Usually we can use pandas library. Pandas store the imported data as DataFrame.

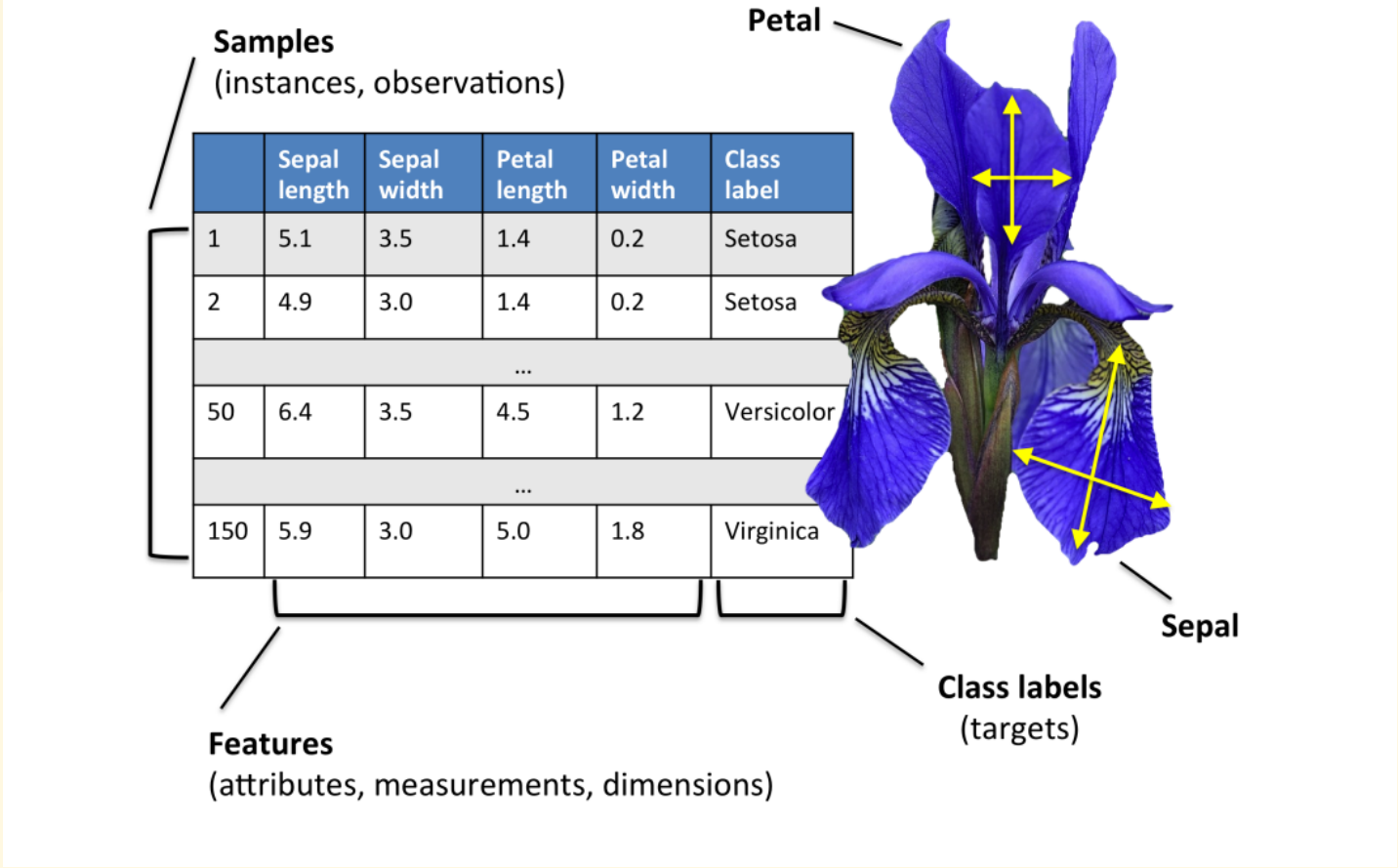
```
# Default sep = ','
df = pd.read_csv("iris_dirty.csv")
```

or if you want to use another separator, simply add sep='\t'

```
df = pd.read_csv("file_name.csv", sep = '\t')
```

Iris Dataset

Fisher's Iris data set introduced by Ronald Fisher in his 1936 paper.



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

View Data

You can have a look at the first five rows with .head():

```
# by default is 5 rows
df.head()
# you can also customize the #-rows
df.head(10)
```

or the last five rows with .tail():

```
df.tail()
```

Data Info

The shape property returns a tuple representing the dimensionality of the DataFrame.

```
df.shape
```

The info() method prints information about the DataFrame.

```
df.info
```

Statistical Description

All standard statistical operations are present in Pandas:

```
# Show the statistical summary on the numerical columns
df.describe()
# or individually
df.mean()
```

```
# Show the statistical summary on the categorical columns
df.describe(include = 'object')
```

Data Cleaning

Finding Missing Values

It is common to have not-a-number (NaN) values in your data set.

```
# Will give the total number of NaN in each column
df.isna().sum()
```

Data Cleaning

Handling Missing Values

```
# Remove the rows with NaN, not recommended
df.dropna()
```

```
# fill NaN with 0, also not recommended
df.fillna(0)

# fill NaN with mean, better
df1 = df.fillna(df.mean(numeric_only=True))
```

Data Cleaning

Problematic Values

Typo can be considered problematic.

```
# Count unique categorical values
df1.Species.unique()
df1['Species'].value_counts()

# View problematic values
df1.iloc[[7]]
```

Data Cleaning

Handling Problematic Values

We can replace with the correct value using replace()

```
df2 = df1.replace(['SETSA'],'setosa')
```

Cleaning done!

Check out my Kaggle post for more data cleaning example.

Data Visualization

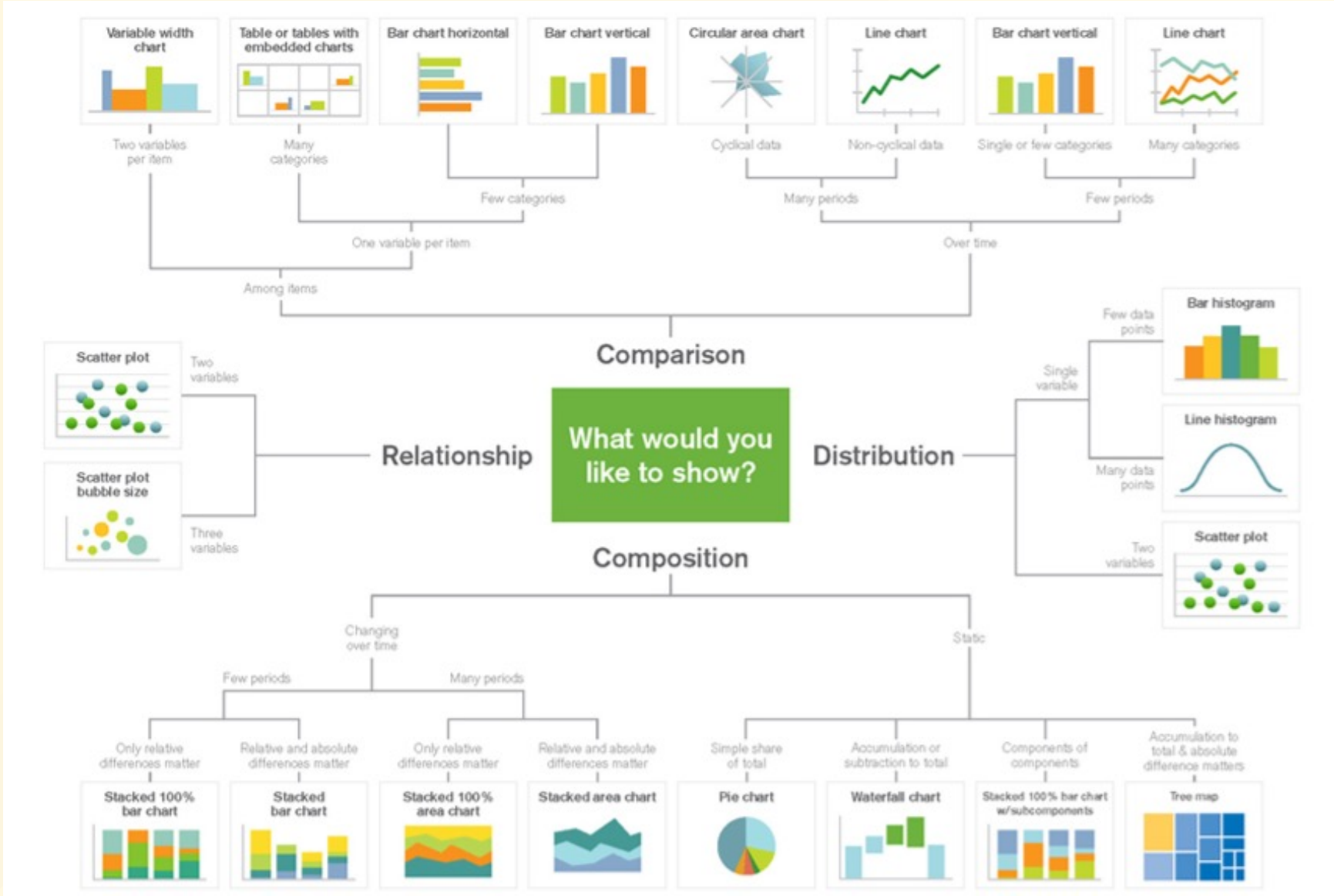
Why?

- Visualizing data prior to analysis is a good practice.
- Statistical description do not fully depict the data set in its entirety.

Check out my video *HERE explaining the importance of visualizing data when analyzing it.

*promo syok sendiri

Cheat Sheet


What would you like to show?
Comparison
Relationship
Distribution
Composition
Variable width chart
Table or tables with embedded charts
Bar chart horizontal
Bar chart vertical
Circular area chart
Line chart
Bar chart vertical
Line chart
Two variables per item
Many categories
Few categories
Cyclical data
Non-cyclical data
Single or few categories
Many categories
One variable per item
Many periods
Few periods
Among items
Over time
Scatter plot
Two variables
Scatter plot bubble size
Three variables
Few data points
Single variable
Many data points
Bar histogram
Line histogram
Two variables
Scatter plot
Changing over time
Static
Few periods
Many periods
Only relative differences matter
Relative and absolute differences matter
Only relative differences matter
Relative and absolute differences matter
Simple share of total
Accumulation or subtraction to total
Components of components
Accumulation to total & absolute difference matters
Stacked 100% bar chart
Stacked bar chart
Stacked 100% area chart
Stacked area chart
Pie chart
Waterfall chart
Stacked 100% bar chart w/subcomponents
Tree map

Data Visualization

Scatter plot

```
df2.plot.scatter(x = 'Petal.Length', y = 'Petal.Width')
```

Using colour as third variable

```
# Dictionary mapping colour with categorical values
colors = {'setosa':'red','virginica':'blue','versicolor':'green'}

df2.plot.scatter(x = 'Petal.Length', y = 'Petal.Width', c = df2['Species'].map(colors))
```

Machine Learning

scikit-learn

```
df2.plot.scatter(x = 'Petal.Length', y = 'Petal.Width')
```