

AI Mastery Course



Kampus
Merdeka
INDONESIA JAYA

orbit
FUTURE ACADEMY
Skills
For
Future
Jobs

Artificial Intelligence
Mastery Program

Module 3

Data analytics with python-
applied analytics

Section


Data manipulation
with pandas





Learning Objectives

Di akhir modul ini, Anda akan dapat:

- Pengertian numpy dan penggunaannya.
 - Mempelajari dasar-dasar preprocessing data.
 - Mempelajari prosedur instalasi dan konsep dasar pada Pandas.
 - Memahami data import export, selection dan filtering dengan dataframes.
 - Data transformation menggunakan pandas.
- 

Agenda

01	NUMPY	Numpy introduction Numpy arrays
02	PANDAS	Pandas Introduction Working with dataframes Data import/export
03	PANDAS	Data Aggregation Statistical analysis Data Transformation
04	CHARTS & PLOTS	Matplotlib for data visualization Seaborn for data visualization
05	CONCLUSION	Summary Quiz



01

NUMPY

- Numpy introduction
- Numpy arrays

Numpy Introduction

A powerful N-dimensional array object.

Sophisticated
(broadcasting/universal)
functions.

Useful linear algebra,
Fourier transform, and
random number
capabilities.

Besides its obvious
scientific uses, NumPy can
also be used as an efficient
multi-dimensional
container of generic data.

Numpy data types

NumPy mendukung lebih luas berbagai type data jika dibandingkan dengan Python. Numpy didefinisikan oleh `numpy.dtype` class dan meliputi:

- `intc` (sama dengan data integer di bahasa C) and `intp` (used for indexing)
- `int8`, `int16`, `int32`, `int64`
- `uint8`, `uint16`, `uint32`, `uint64`
- `float16`, `float32`, `float64`
- `complex64`, `complex128`
- `bool_`, `int_`, `float_`, `complex_` are shorthand for defaults.

Ini dapat digunakan sebagai fungsi untuk cast literals atau sequence types, Serta argumen untuk numpy functions, yang menerima Argument kata kunci `dtype`.

Data type	Description
<code>bool_</code>	Boolean (True or False) stored as a byte
<code>int_</code>	Default integer type (same as C long; normally either <code>int64</code> or <code>int32</code>)
<code>intc</code>	Identical to C <code>int</code> (normally <code>int32</code> or <code>int64</code>)
<code>intp</code>	Integer used for indexing (same as C <code>ssize_t</code> ; normally either <code>int32</code> or <code>int64</code>)
<code>int8</code>	Byte (-128 to 127)
<code>int16</code>	Integer (-32768 to 32767)
<code>int32</code>	Integer (-2147483648 to 2147483647)
<code>int64</code>	Integer (-9223372036854775808 to 9223372036854775807)
<code>uint8</code>	Unsigned integer (0 to 255)
<code>uint16</code>	Unsigned integer (0 to 65535)
<code>uint32</code>	Unsigned integer (0 to 4294967295)
<code>uint64</code>	Unsigned integer (0 to 18446744073709551615)
<code>float_</code>	Shorthand for <code>float64</code>
<code>float16</code>	Half-precision float: sign bit, 5 bits exponent, 10 bits mantissa
<code>float32</code>	Single-precision float: sign bit, 8 bits exponent, 23 bits mantissa
<code>float64</code>	Double-precision float: sign bit, 11 bits exponent, 52 bits mantissa
<code>complex_</code>	Shorthand for <code>complex128</code>
<code>complex64</code>	Complex number, represented by two 32-bit floats
<code>complex128</code>	Complex number, represented by two 64-bit floats

Numpy Arrays

```
1 import numpy as np
2 # Create an array
3 civilian_birth = np.array([4352, 233, 3245, 256, 2394])
4 civilian_birth
```

```
array([4352, 233, 3245, 256, 2394])
```

```
1 np.zeros((3, 6)) # special arrays with numpy
```

```
array([[0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0.]])
```

```
1 np.linspace(0, 1, 5) # 0 to 1 (inclusive) with 5 points
```

```
array([0. , 0.25, 0.5 , 0.75, 1.  ])
```

```
1 a = np.array([0, 1])
2 b = np.array([2, 3])
3 ab = np.stack((a, b)).T # stacking arrays
4 print(ab)
```

```
[[0 2]
 [1 3]]
```

Linear Algebra with numpy

```
1 # (4x+5y=23)
2 # (6x-3y=3)
3
4 import numpy as np
5 A = np.array([[4,5],[6,-3]])
6 b = np.array([23, 3])
7 x,y = np.linalg.solve(A,b)
8 print("value of x = ",x,"\n value of y = ",y)
```

```
value of x =  2.0
value of y =  3.0
```

```
1 matrix = np.matrix([[1, 5, 3], [9, 5, 6], [2, 8, 9]])
2 matrix.flatten() # Flatten matrix
```

```
matrix([[1, 5, 3, 9, 5, 6, 2, 8, 9]])
```

```
1 np.linalg.det(matrix) # Return determinant of matrix
```

```
-162.00000000000009
```

```
1 matrix.diagonal().sum() # Calculate the trace of the matrix
```

```
15
```

```
1 np.linalg.inv(matrix) # Calculate inverse of matrix
```

```
matrix([[ 0.01851852,  0.12962963, -0.09259259],
        [ 0.42592593, -0.01851852, -0.12962963],
        [-0.38271605, -0.01234568,  0.24691358]])
```




02

PANDAS

- Pandas Introduction
- Working with dataframes
- Data import/export



Pengenalan Pandas

Pandas adalah library Python bersifat open source yang banyak digunakan sebagai tools analisis data pada Python.

Pandas adalah tools yang sangat cocok untuk analisis dan pemodelan data.

Pandas adalah Library Python yang lebih baik dari NumPy dari sisi fungsionalitas, dan menyediakan implementasi DataFrame yang efisien.

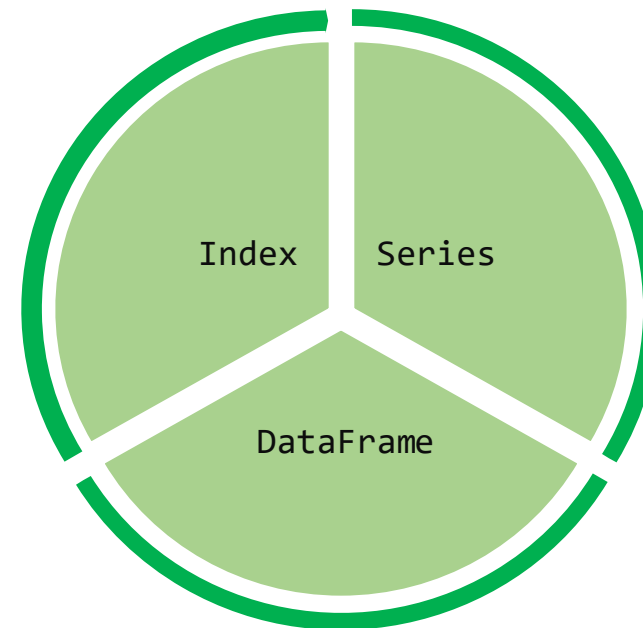
DataFrames pada dasarnya adalah array multidimensi dengan label baris dan kolom.

Pandas objects

Objek Pandas adalah versi array terstruktur NumPy yang disempurnakan, di mana baris dan kolom diidentifikasi menggunakan label daripada indeks integer sederhana.

Ada banyak alat, metode, dan fungsi berguna yang ditawarkan Panda, selain dari struktur data dasar.

Tiga struktur data dasar pada Pandas, seperti pada gambar disamping kanan:



Pandas series and dataframes

Dimensions	Name	Description
1	Series	1D labeled homogeneously-typed array
2	DataFrame	General 2D labeled, size-mutable tabular structure with potentially heterogeneously-typed column

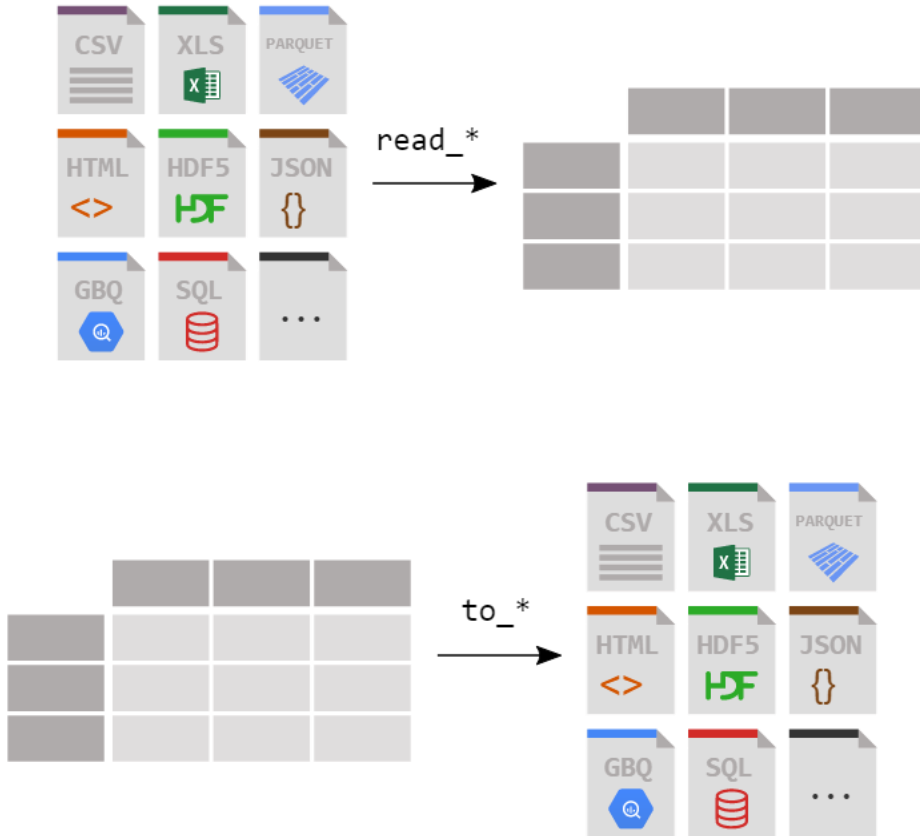
```
1 import pandas as pd
2 data = ['a','b','c','d']
3 s = pd.Series(data) # creating a pandas series
4 print(s)
```

```
0    a
1    b
2    c
3    d
dtype: object
```

```
1 data = {'name': ['Jason', 'Molly', 'Tina', 'Jake'],
2         'year': [2012, 2012, 2013, 2014], 'reports': [4, 24, 31, 2]}
3 index = ['Cochice', 'Pima', 'Santa Cruz', 'Maricopa']
4 df = pd.DataFrame(data, index = index) # Create an dataframe
5 df
```

	name	year	reports
Cochice	Jason	2012	4
Pima	Molly	2012	24
Santa Cruz	Tina	2013	31
Maricopa	Jake	2014	2

Pandas: Data import, export



Format Type	Data Description	Reader	Writer
text	CSV	read_csv	to_csv
text	Fixed-Width Text File	read_fwf	
text	JSON	read_json	to_json
text	HTML	read_html	to_html
text	LaTeX		Styler.to_latex
text	XML	read_xml	to_xml
text	Local clipboard	read_clipboard	to_clipboard
binary	MS Excel	read_excel	to_excel
binary	OpenDocument	read_excel	
binary	HDF5 Format	read_hdf	to_hdf
binary	Feather Format	read_feather	to_feather
binary	Parquet Format	read_parquet	to_parquet
binary	ORC Format	read_orc	
binary	Stata	read_stata	to_stata
binary	SAS	read_sas	
binary	SPSS	read_spss	
binary	Python Pickle Format	read_pickle	to_pickle
SQL	SQL	read_sql	to_sql
SQL	Google BigQuery	read_gbq	to_gbq

Pandas: Data import, export

```
1 # Load a csv
2 url = 'https://raw.githubusercontent.com/anshupandey/WileyNXT/main/DataEngineer/01-Data-Import-Export/01-Data-Import-Export.csv'
3 df = pd.read_csv(url)
4 df.head(2)
```

	Dates	Temperature	Humidity	Pressure	Air Quality
0	30-04-2018	218	182	4	2
1	01-05-2018	2592	182	3	2

```
1 # Load a JSON file
2 url = 'https://raw.githubusercontent.com/anshupandey/WileyNXT/main/DataEngineer/01-Data-Import-Export/01-Data-Import-Export.json'
3 df = pd.read_json(url)
4 df.head(2)
```

	id	dates	Accepted	Rejected	App	Server
0	0	1525046400000	218	182	App_4	Server_02
1	1	1525132800000	2592	182	App_3	Server_02

```
1 # Save dataframe as csv in the working director
2 df.to_csv('server_data.csv')
```

Pandas: Selection & Filtering

```
1 import pandas as pd
2 data = {'name': ['Jason', 'Molly', 'Tina'],
3         'year': [2012, 2012, 2013],
4         'reports': [4, 5, 9],
5         'coverage': [25, 62, 70]}
6 df = pd.DataFrame(data)
7 df['name'] # View Column
```

```
0 Jason
1 Molly
2 Tina
Name: name, dtype: object
```

```
1 df[['name', 'reports']] # View Two Columns
```

	name	reports
0	Jason	4
1	Molly	24
2	Tina	31

```
1 df[:2] # View First Two Rows
```

	name	year	reports	coverage
0	Jason	2012	4	25
1	Molly	2012	24	62

```
1 df[df['coverage'] > 50] # Rows Where Coverage>50
```

	name	year	reports	coverage
1	Molly	2012	24	62
2	Tina	2013	31	70

```
1 # View Rows Where Coverage > 20 & Reports < 6
2 df[(df['coverage'] > 20) & (df['reports'] < 6)]
```

	name	year	reports	coverage
0	Jason	2012	4	25



03

PANDAS

- Data Aggregation
- Statistical analysis
- Data Transformation



Data Aggregation: groupby

```
1 import pandas as pd
2 data = {'name': ['Jen', 'Max', 'Tina', 'John', 'Kay', 'Ish', 'Ami'],
3         'year': [2012, 2012, 2013, 2012, 2013, 2014, 2014],
4         'score': [4, 5, 9, 8, 9, 5, 4]}
5 df = pd.DataFrame(data)
6
7 # calculating year wise average score
8 df.groupby(['year'])['score'].mean()
```

```
year
2012    5.666667
2013    9.000000
2014    4.500000
Name: score, dtype: float64
```

Pandas: Statistical analysis

```
1 import pandas as pd
2 data = {'name': ['Jason', 'Molly', 'Tina', 'Jake', 'Amy'],
3         'age': [42, 52, 36, 42, 73],
4         'score': [4, 24, 31, 2, 3]}
5 df = pd.DataFrame(data)
6 df.describe().T #Summary statistics
```

	count	mean	std	min	25%	50%	75%	max
age	5.0	49.0	14.594520	36.0	42.0	42.0	52.0	73.0
score	5.0	12.8	13.663821	2.0	3.0	4.0	24.0	31.0

```
1 print("mean of score ");print(df['score'].mean())
2 print("Median of score ");print(df['score'].median())
3 print("Mode of age ");print(df['age'].mode())
4 print("Variance of age ");print(df['age'].var())
5 print("Skewness of age ");print(df['age'].skew())
```

```
mean of score
12.8
Median of score
4.0
Mode of age
0    42
dtype: int64
Variance of age
213.0
Skewness of age
1.4700997201463943
```

Pandas : Transformasi Data menggunakan map



Teknik ini diterapkan jika kita perlu melakukan transformasi apa pun berdasarkan nilai dalam array, Seri, atau kolom dalam DataFrame.



Map methodology pada Seri mengambil fungsi atau objek seperti dikte yang berisi pemetaan.



Menggunakan map method adalah cara yang nyaman untuk melakukan transformasi elemen-bijaksana dan operasi pembersihan data terkait lainnya.



Hasil yang sama juga dapat dicapai dengan melewati fungsi.

Pandas : Data Transformation using map

```
1 import pandas as pd
2 data = {'name': ['Jen K Forst', 'Max M Brown', 'Tina W Musk',
3               'John M Gates', 'Kay W Carter', 'Ish T Sodhi'],
4         'score': [4, 5, 9, 8, 9, 5]}
5 df = pd.DataFrame(data)
6 df.T
```

	0	1	2	3	4	5
name	Jen K Forst	Max M Brown	Tina W Musk	John M Gates	Kay W Carter	Ish T Sodhi
score	4	5	9	8	9	5

```
1 def get_code(name):
2     return "".join([k[0].upper() for k in name.split(' ')])
3 # Creating a column with initials
4 df['code'] = df['name'].map(get_code)
5 df.T
```

	0	1	2	3	4	5
name	Jen K Forst	Max M Brown	Tina W Musk	John M Gates	Kay W Carter	Ish T Sodhi
score	4	5	9	8	9	5
code	JKF	MMB	TWM	JMG	KWC	ITS

Pandas : Transformasi Data – mengganti nilai



Metode penggantian dapat dengan mudah digunakan untuk mengganti nilai.



Kita dapat mengganti nilai sentinel dengan NA menggunakan metode ganti, yang akan menghasilkan seri baru.



Untuk mengganti beberapa nilai sekaligus, kita dapat melewati daftar dan kemudian nilai pengganti.



Untuk menggunakan pengganti yang berbeda untuk setiap nilai, berikan daftar pengganti. Sebuah dikte juga dapat dilewatkan sebagai argumen.



Metode `data.replace` berbeda dari `data.str.replace`, yang melakukan elemen substitusi string

Pandas : Data Transformation – replacing values

```
1 import pandas as pd
2 data = {'name': ['Jen K Forst', 'Max M Brown', 'Tina W Musk',
3                 'John M Gates', 'Kay W Carter', 'Ish T Sodhi'],
4         'score': [4, 5, 9, 8, 9, 5]}
5 df = pd.DataFrame(data)
6 df.T
```

	0	1	2	3	4	5
name	Jen K Forst	Max M Brown	Tina W Musk	John M Gates	Kay W Carter	Ish T Sodhi
score	4	5	9	8	9	5

```
1 df.replace("Jen K Forst", 'Jennifer K Forest', inplace=True)
2 df.T
```

	0	1	2	3	4	5
name	Jennifer K Forest	Max M Brown	Tina W Musk	John M Gates	Kay W Carter	Ish T Sodhi
score	4	5	9	8	9	5

Pandas : Data Transformation – discretization & binning

Data kontinu umumnya discretisasi atau dipisahkan menjadi “bins” Untuk analisis lebih lanjut.

Untuk membagi data menjadi bins, Kita bisa menggunakan cut function di pandas, yang akan mengembalikan objek Kategoris khusus.

Objek Kategoris mirip dengan array string, yang menunjukkan nama bin, yang secara internal berisi array, categories, menentukan nama kategori yang berbeda.

Fungsi lain yang disebut qcut, bins data berdasarkan kuanttil sampel. qcut menggunakan kuanttil sampel yang akan menghasilkan bins berukuran kira-kira sama.

Pandas : Data Transformation – discretization & binning

```
1 import pandas as pd
2 data = {'name': ['Jen', 'Max', 'Tina', 'John', 'Kay', 'Ish', 'Ami'],
3         'year': [2012, 2012, 2013, 2012, 2013, 2014, 2014],
4         'score': [4, 5, 9, 8, 9, 5, 4]}
5 df = pd.DataFrame(data)
6
7 # creating a categorical attribute for score
8 df['score_cat'] = pd.cut(df['score'], bins=[0,3,6,10],
9                          labels=['Low', 'Medium', 'High'])
10 df.T
```

	0	1	2	3	4	5	6
name	Jen	Max	Tina	John	Kay	Ish	Ami
year	2012	2012	2013	2012	2013	2014	2014
score	4	5	9	8	9	5	4
score_cat	Medium	Medium	High	High	High	Medium	Medium



04

CHARTS & PLOTS

- Matplotlib for data visualization
- Seaborn for data visualization



Data Visualization

Visualisasi Data digunakan untuk mengkomunikasikan informasi dengan jelas dan efisien kepada pengguna dengan menggunakan grafik informasi seperti tabel dan bagan. Ini membantu pengguna dalam menganalisis sejumlah besar data dengan cara yang lebih sederhana. Itu membuat data kompleks lebih mudah diakses, dimengerti, dan dapat digunakan.



Matplotlib

Matplotlib adalah salah satu paket Python paling populer yang digunakan untuk visualisasi data.

Matplotlib adalah cross-platform library Untuk membuat 2D plots dari data dalam array.

Matplotlib ditulis dengan Python dan menggunakan NumPy, ekstensi matematika numerik Python.

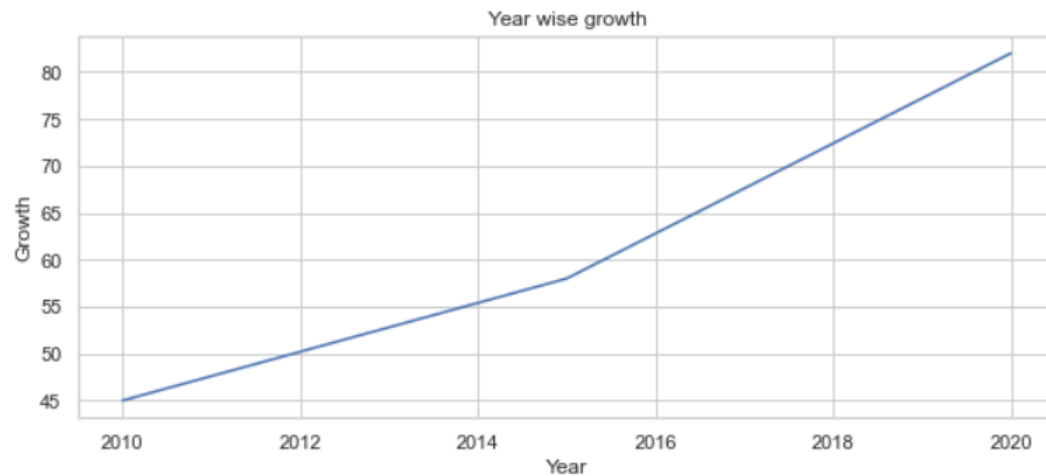
Matplotlib memberikan sebuah object-oriented API yang membantu dalam menanamkan plot dalam aplikasi menggunakan toolkit Python GUI seperti PyQt, WxPythonotTkinter.

Matplotlib Functions

Sr. No	Function	Description
1	Axes	Add axes to the figure.
2	plt.text	Add text to the axes.
3	plt.title	Set a title of the current axes.
4	plt.xlabel	Set the x axis label of the current axis.
5	plt.xlim	Get or set the x limits of the current axes.
6	plt.xscale	Set the scaling of x axis
7	plt.xticks	Get or set the x-limits of the current tick locations and labels.
8	plt.yscale	Set the y axis label of the current axis.
9	plt.ylim	Get or set the y-limits of the current axes.
10	plt.yscale	Set the scaling of the y-axis.
11	plt.yticks	Get or set the y-limits of the current tick locations and labels.

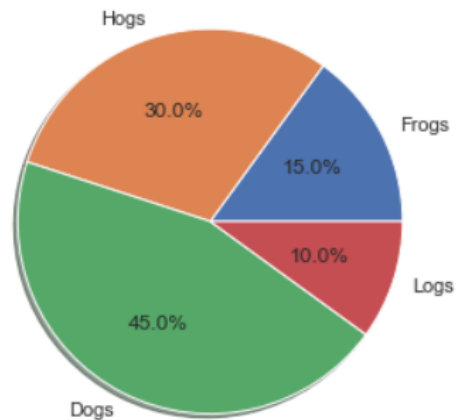
Line plot with matplotlib

```
1 import matplotlib.pyplot as plt
2 year = [2010,2015,2020]
3 growth = [45,58,82]
4
5 plt.figure(figsize=(10,4))
6 plt.plot(year,growth)
7 plt.xlabel("Year")
8 plt.ylabel("Growth")
9 plt.title("Year wise growth")
10 plt.show()
```



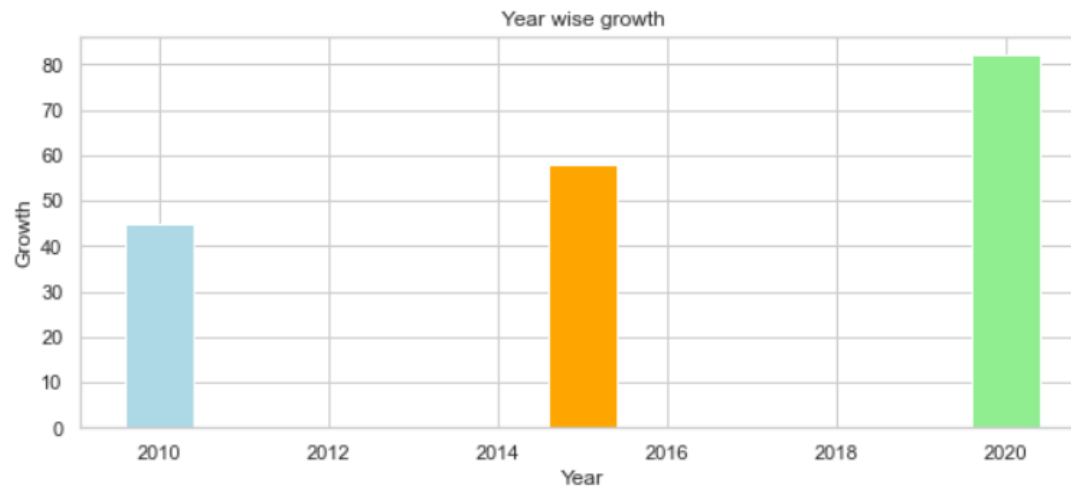
Pie chart with matplotlib

```
1 import matplotlib.pyplot as plt
2 from matplotlib.gridspec import GridSpec
3
4 labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
5 fracs = [15, 30, 45, 10]
6 explode = (0, 0.05, 0, 0)
7
8 plt.figure(figsize=(5,5))
9 plt.pie(fracs, labels=labels, autopct='%1.1f%%', shadow=True)
10 plt.show()
```



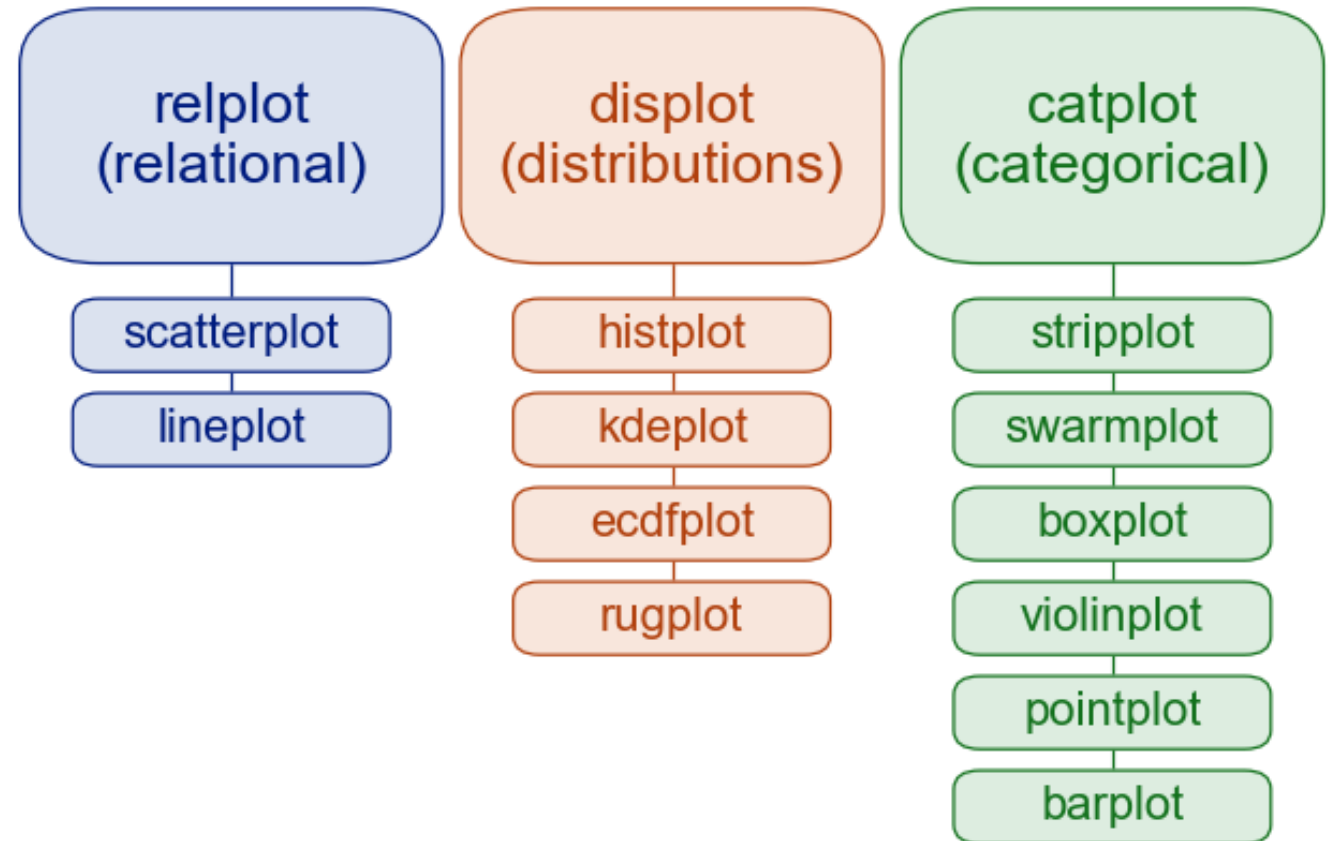
Bar plot with matplotlib

```
1 import matplotlib.pyplot as plt
2 year = [2010,2015,2020]
3 growth = [45,58,82]
4
5 plt.figure(figsize=(10,4))
6 plt.bar(year,growth,color=['lightblue','orange','lightgreen'])
7 plt.xlabel("Year")
8 plt.ylabel("Growth")
9 plt.title("Year wise growth")
10 plt.show()
```



Seaborn

Seaborn adalah library untuk membuat grafik statistik di Python. Seaborn dibangun di atas matplotlib dan terintegrasi dengan struktur data di Pandas.

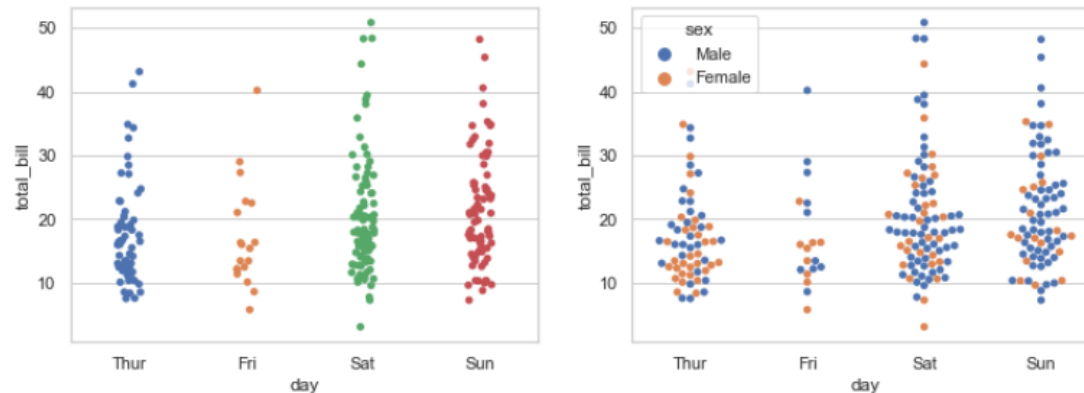


Stripplot and swarmplot with seaborn

Strip plot digunakan untuk menganalisis numerik v/s categorical data

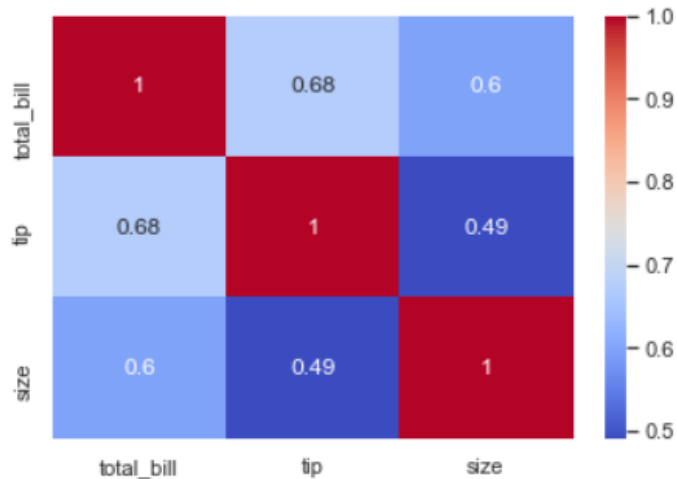
Swarm plot dapat digunakan untuk melakukan analisis multivariat antara numerik v/s categorical v/s categorical

```
1 import matplotlib.pyplot as plt
2 import seaborn as sns
3
4 sns.set(style="whitegrid", color_codes=True)
5 tips = sns.load_dataset("tips")
6
7 plt.figure(figsize=(12,4))
8 plt.subplot(121)
9 sns.stripplot(x="day", y="total_bill", data=tips, jitter=True)
10 plt.subplot(122)
11 sns.swarmplot(x="day", y="total_bill", hue="sex", data=tips)
12 plt.show()
```



Heatmap with Seaborn

```
1 import matplotlib.pyplot as plt
2 import seaborn as sns
3
4 sns.set(style="whitegrid", color_codes=True)
5 tips = sns.load_dataset("tips")
6
7 sns.heatmap(tips.corr(), annot=True, cmap='coolwarm')
8 plt.show()
9
```





05

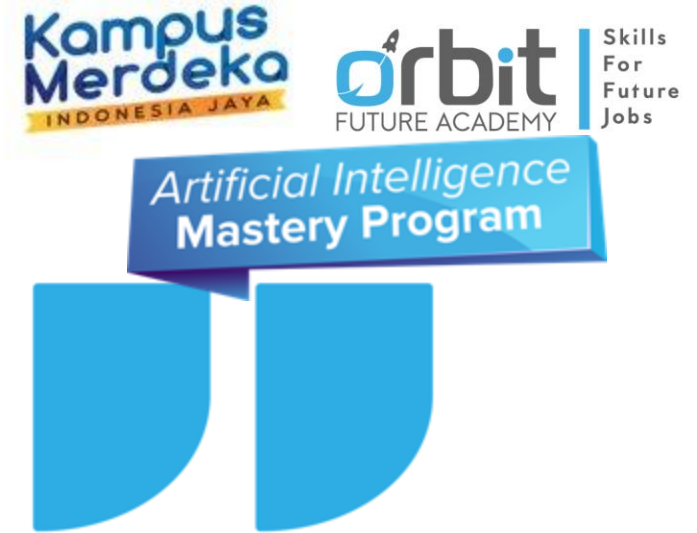
CONCLUSION

- Summary



Summary

1. Numpy adalah paket python yang banyak digunakan untuk perhitungan matematis dengan python.
2. Pandas dapat digunakan untuk melakukan data import, eksport, data cleaning, data wrangling, data aggregation.
3. Series dan dataframes adalah primary data types dalam python.
4. Matplotlib dan seaborn sebagai paket yang digunakan untuk visualisasi data dengan python.



Quiz

Question

manakah dari paket berikut yang tidak digunakan untuk visualisasi data?

- A. Matplotlib
- B. Seaborn
- C. Numpy
- D. plotly

Quiz

Question

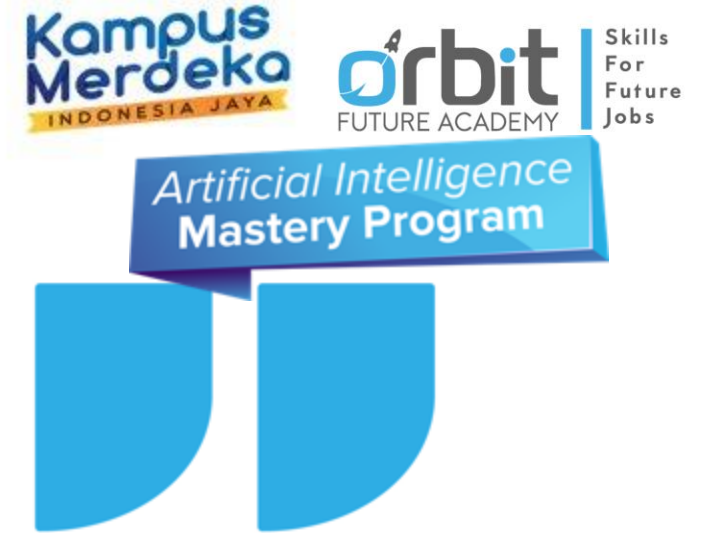
manakah dari paket berikut yang tidak digunakan untuk visualisasi data?

- A. Matplotlib
- B. Seaborn
- C. Numpy
- D. plotly

Answer: C

Refference

- Data Analysis with Pandas, Stefani Molin
- Effective Pandas, Matt Harrison





**Kampus
Merdeka**
INDONESIA JAYA

orbit
FUTURE ACADEMY | Skills
For
Future
Jobs

**Artificial Intelligence
Mastery Program**

TERIMA KASIH

Orbit Future Academy

PT Orbit Ventura Indonesia
Center of Excellence (Jakarta Selatan)
Gedung Veteran RI, Lt.15
Unit Z15-002, Plaza Semanggi
Jl. Jenderal Sudirman Kav.50, Jakarta
12930, Indonesia

- 📖 Jakarta Selatan/Pusat
- 📖 Jakarta Barat/BSD
- 📖 Kota Bandung
- 📖 Kab. Bandung
- 📖 Jawa Barat

Hubungi Kami

Director of Sales & Partnership
ira@orbitventura.com
+62 858-9187-7388

Social Media

 Orbit Future Academy

 OrbitFutureAcademy

 @OrbitFutureAcademyIn1

 Orbit Future Academy