Networking Academy

Python DTS PRoA 2022

Library Session 3: Pandas

Muhammad Ogin Hasanuddin

KK Teknik Komputer
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Introduction

- Pandas is an open-source library for data analysis and manipulation. It is a go-to toolkit for data scientists.
- Pandas integrates seamlessly with other Python libraries such as NumPy and Matplotlib for numeric processing and visualizations.
- When using Pandas, we will primarily interact with DataFrames and Series.
- Pandas “panel data”

Importing Pandas

- In order to use Pandas, you must import it. This is as simple as:

```
import pandas
```

- However, you'll rarely see Pandas imported this way. By convention programmers rename Pandas to pd. This isn't a requirement, but it is a pattern that you'll see repeated often.
- To import Pandas in the conventional manner run the code block below.

```
import pandas as pd
```

- After importing Pandas as pd we can use pandas by calling methods provided by pd.
- For example we can print pandas version by run code below

```
pd.__version__
```

Pandas Series

A Series represents a sequential list of data. It is a foundational building block of the powerful DataFrame.

Creating a Series

- We create a new Series object as we would any Python object:

```
s = pd.Series()
```

- This creates a new, empty Series object, which isn't very interesting. You can create a series object with data by passing it a list or tuple:

```
temperatures = [55, 63, 72, 65, 63, 75, 67,
59, 82, 54]

series = pd.Series(temperatures)

print(type(series))
print(series)
```

- Here we created a new pandas.core.series.Series object with ten values presumably representing some temperature measurement.

Pandas DataFrame

If you picture Series as a list of data, you can think of DataFrame as a table of data. A DataFrame consists of one or more Series presented in a tabular format. Each Series in the DataFrame is a column.

Creating a DataFrame

- We can create an empty DataFrame using the DataFrame class in Pandas:

```
df = pd.DataFrame()
```

Terimakasih!

Library Session Pandas

April 11, 2022

0.0.1 Fungsi dan argumen

[2]:
```
print("Muhammad Ogin Hasanuddin")
```

```
Muhammad Ogin Hasanuddin
```

[3]:
```
print("11 April 2022")
```

```
11 April 2022
```

[4]:
```
print(100/10)
```

```
10.0
```

[5]:
```
type(100/10)
```

[5]:
```
float
```

[6]:
```
print(type(100/10))
```

```
<class 'float'>
```

0.0.2 variable dan tipe data

[7]:
```
nama = "Muhammad Ogin Hasanudin"
```

[8]:
```
jarak_km = 5
```

[9]:
```
waktu_tempuh_jam = 2
```

[10]:
```
kecepatan = jarak_km/waktu_tempuh_jam
```

[11]:
```
print(kecepatan)
```

```
2.5
```

[12]:
```
print(type(jarak_km))
```

```
<class 'int'>
```

[13]:
```
print(type(kecepatan))
```

```
<class 'float'>
```

0.0.3 List

[14]:
```
c = [True, 100, 2.5, 'Muhammad Ogin Hasanuddin']
```

[15]:
```
print(type(c))
```

```
<class 'list'>
```

[16]:
```
c[0]
```

[16]:
```
True
```

[17]:
```
# slice dua data pertama
c[:2]
```

[17]:
```
[True, 100]
```

0.0.4 Array

[18]:
```
a = [1, 2, 5, 7, 9]
b = [4, 2, 7, 1, 8]
```

[19]:
```
c = [a, b]
```

[20]:
```
c[0][1]
```

[20]:
```
2
```

[21]:
```
c[0][4]
```

[21]:
```
9
```

[22]:
```
c
```

[22]:
```
[[1, 2, 5, 7, 9], [4, 2, 7, 1, 8]]
```

[23]:
```
print(type(c))
```

```
<class 'list'>
```

[24]:
```
import numpy as np
```

[25]:
```
np.mean(a)
```

[25]:
```
4.8
```

[26]:
```
jum_a = 1 + 2 + 5 + 7 + 9
```

[27]:
```
rata_rata_a = jum_a/5
```

[28]:
```
rata_rata_a
```

[28]: 4.8

[29]:
```
### plotting
```

[30]:
```
import matplotlib.pyplot as plt
```

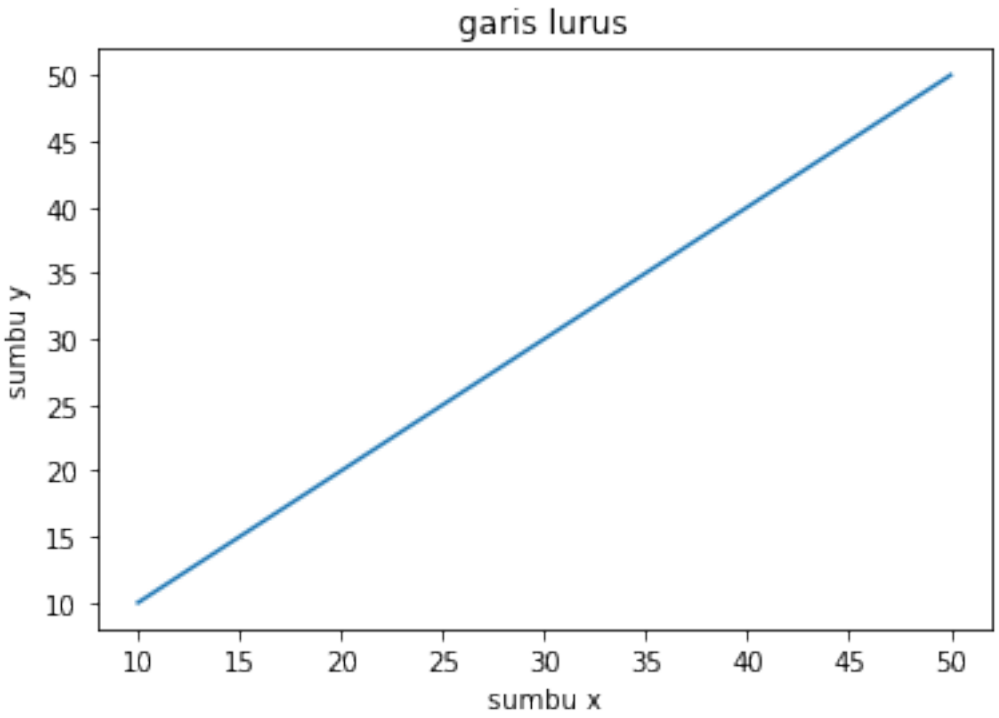
[32]:
```
x = [10 , 20, 30, 40, 50]
y = [10 , 20, 30, 40, 50]

plt.plot(x,y)

# judul
plt.title("garis lurus")

plt.xlabel('sumbu x')
plt.ylabel('sumbu y')

plt.show()
```



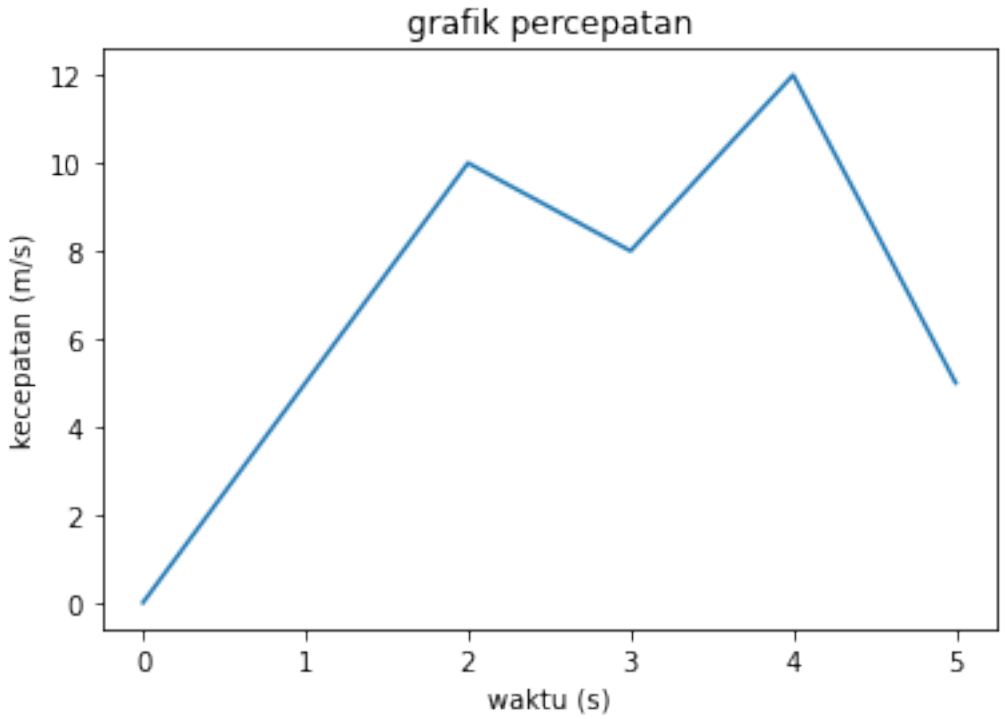
[33]:
```
kecepatan = [0, 5, 10, 8, 12, 5] # sumbu y
waktu = [0, 1, 2, 3, 4, 5] # sumbu x

plt.plot(waktu, kecepatan)

# judul
plt.title("grafik percepatan")

plt.xlabel('waktu (s)')
plt.ylabel('kecepatan (m/s)')

plt.show()
```



0.0.5 importing pandas

[34]:
```
import pandas as pd
```

[35]:
```
pd.__version__
```

[35]:
```
'1.2.4'
```

[36]:
```
temperatures = [55, 63, 72, 65, 63, 75, 67, 59, 82, 54]
```

```
series = pd.Series(temperatures)
```

[37]:
```
print(type(temperatures))
```

```
<class 'list'>
```

[38]:
```
print(type(series))
```

```
<class 'pandas.core.series.Series'>
```

[40]:
```
print(temperatures)
```

```
[55, 63, 72, 65, 63, 75, 67, 59, 82, 54]
```

[41]:
```
print(series)
```

```
0    55
1    63
2    72
3    65
4    63
5    75
6    67
7    59
8    82
9    54
dtype: int64
```

0.0.6 analisis terhadap series

[42]:
```
series.describe()
```

[42]:
```
count    10.000000
mean     65.500000
std       8.847473
min      54.000000
25%      60.000000
50%      64.000000
75%      70.750000
max      82.000000
dtype: float64
```

[43]:
```
series_a = pd.Series(a)
```

[44]:
```
series_a.describe()
```

[44]:
```
count    5.00000
mean     4.80000
std      3.34664
```

```
min      1.00000
25%      2.00000
50%      5.00000
75%      7.00000
max      9.00000
dtype: float64
```

[46]:
```
series.is_unique
```

[46]:
```
False
```

[47]:
```
series_a.is_unique
```

[47]:
```
True
```

[48]:
```
series_a.is_monotonic
```

[48]:
```
True
```

[49]:
```
series_b = pd.Series(b)
```

[50]:
```
series_b.is_monotonic
```

[50]:
```
False
```

[55]:
```
berat = (120, 143, 98, 280, 175, 205, 210, 115, 122, 175, 201)
```

[54]:
```
print(type(berat))
```

```
<class 'tuple'>
```

[56]:
```
series_berat = pd.Series(berat)
```

[57]:
```
series_berat
```

[57]:
```
0     120
1     143
2      98
3     280
4     175
5     205
6     210
7     115
8     122
9     175
10    201
dtype: int64
```

[58]:
```
series_berat.describe()
```

[58]:
```
count     11.000000
mean     167.636364
std       54.421085
min       98.000000
25%      121.000000
50%      175.000000
75%      203.000000
max      280.000000
dtype: float64
```

[59]:
```
series_berat[9]
```

[59]:
```
175
```

[60]:
```
for temp in series_berat:
    print(temp)
```

```
120
143
98
280
175
205
210
115
122
175
201
```

0.0.7 modifikasi nilai

[61]:
```
berat = [70, 75, 72, 68, 71, 69, 65, 67, 73, 74, 64]
```

[62]:
```
series = pd.Series(berat)
```

[63]:
```
series
```

[63]:
```
0     70
1     75
2     72
3     68
4     71
5     69
6     65
7     67
8     73
9     74
10    64
```

```
dtype: int64
```

[64]:
```
series[1]
```

[64]:
```
75
```

[65]:
```
series[1] = 78
```

[66]:
```
print(series[1])
```

```
78
```

[67]:
```
series
```

[67]:
```
0     70
1     78
2     72
3     68
4     71
5     69
6     65
7     67
8     73
9     74
10    64
dtype: int64
```

[68]:
```
series = series + 1
```

[69]:
```
series
```

[69]:
```
0     71
1     79
2     73
3     69
4     72
5     70
6     66
7     68
8     74
9     75
10    65
dtype: int64
```

0.0.8 meremove elemen

[70]:
```
series.pop(5)
```

[70]:
```
70
```

[71]:
```
series
```

[71]:
```
0     71
1     79
2     73
3     69
4     72
6     66
7     68
8     74
9     75
10    65
dtype: int64
```

[73]:
```
try:
    print(series[5])
except:
    print('nilai pada index 5 tidak ada')
```

```
nilai pada index 5 tidak ada
```

[74]:
```
series.reset_index()
```

[74]:

	index	0
0	0	71
1	1	79
2	2	73
3	3	69
4	4	72
5	6	66
6	7	68
7	8	74
8	9	75
9	10	65

[75]:
```
series
```

[75]:
```
0     71
1     79
2     73
3     69
4     72
6     66
7     68
8     74
9     75
```

```
10    65
dtype: int64
```

[76]:
```
series.reset_index(drop=True, inplace=True)
```

[77]:
```
series
```

[77]:
```
0    71
1    79
2    73
3    69
4    72
5    66
6    68
7    74
8    75
9    65
dtype: int64
```

0.0.9 menambah elemen

[78]:
```
elemen_tambahan = [76, 64]
```

[79]:
```
series_tambahan = pd.Series(elemen_tambahan)
```

[80]:
```
series_tambahan
```

[80]:
```
0    76
1    64
dtype: int64
```

[81]:
```
series.append(series_tambahan, ignore_index=True)
```

[81]:
```
0     71
1     79
2     73
3     69
4     72
5     66
6     68
7     74
8     75
9     65
10    76
11    64
dtype: int64
```

[82]:
```
series
```

[82]:
```
0    71
1    79
2    73
3    69
4    72
5    66
6    68
7    74
8    75
9    65
dtype: int64
```

[83]:
```
series = series.append(series_tambahan, ignore_index=True)
```

[84]:
```
series
```

[84]:
```
0     71
1     79
2     73
3     69
4     72
5     66
6     68
7     74
8     75
9     65
10    76
11    64
dtype: int64
```

0.0.10 sorting

[85]:
```
series.sort_values()
```

[85]:
```
11    64
9     65
5     66
6     68
3     69
0     71
4     72
2     73
7     74
8     75
10    76
1     79
dtype: int64
```

0.0.11 creating dataframes

[86]:
```
df = pd.DataFrame()
```

[87]:
```
df
```

[87]:
```
Empty DataFrame
Columns: []
Index: []
```

[88]:
```
type(df)
```

[88]:
```
pandas.core.frame.DataFrame
```

[89]:
```
nama_kota = pd.Series([
    'Bandung',
    'Jakarta',
    'Batam',
    'Depok',
    'Bekasi',
    'Cimahi',
    'Bogor'
])
```

[90]:
```
nama_kota
```

[90]:
```
0    Bandung
1    Jakarta
2      Batam
3      Depok
4     Bekasi
5     Cimahi
6      Bogor
dtype: object
```

[91]:
```
type(nama_kota)
```

[91]:
```
pandas.core.series.Series
```

[92]:
```
populasi = pd.Series([
    448094,
    954201,
    32391,
    401923,
    894201,
    34572,
    50932
])
```

[93]:
```
jumlah_kampus = pd.Series([
    20,
    40,
    15,
    10,
    18,
    8,
    12
])
```

[94]:
```
print(nama_kota, populasi, jumlah_kampus)
```

```
0    Bandung
1    Jakarta
2      Batam
3      Depok
4     Bekasi
5     Cimahi
6      Bogor
dtype: object 0    448094
1    954201
2     32391
3    401923
4    894201
5     34572
6     50932
dtype: int64 0    20
1    40
2    15
3    10
4    18
5     8
6    12
dtype: int64
```

[95]:
```
df = pd.DataFrame({
    'Nama Kota': nama_kota,
    'Populasi': populasi,
    'Jumlah Kampus': jumlah_kampus
})
```

[96]:
```
df
```

[96]:

	Nama Kota	Populasi	Jumlah Kampus
0	Bandung	448094	20
1	Jakarta	954201	40
2	Batam	32391	15
3	Depok	401923	10

```
4    Bekasi    894201             18
5    Cimahi     34572              8
6     Bogor     50932             12
```

0.0.12 analisis terhadap dataframe

[98]:
```
df.describe()
```

[98]:
```
            Populasi  Jumlah Kampus
count       7.000000       7.000000
mean   402330.571429      17.571429
std    396689.331701      10.768119
min     32391.000000       8.000000
25%     42752.000000      11.000000
50%    401923.000000      15.000000
75%    671147.500000      19.000000
max    954201.000000      40.000000
```

[100]:
```
df.describe(include='all')
```

[100]:
```
       Nama Kota       Populasi  Jumlah Kampus
count          7       7.000000       7.000000
unique         7            NaN            NaN
top        Bogor            NaN            NaN
freq           1            NaN            NaN
mean         NaN  402330.571429      17.571429
std          NaN  396689.331701      10.768119
min          NaN   32391.000000       8.000000
25%          NaN   42752.000000      11.000000
50%          NaN  401923.000000      15.000000
75%          NaN  671147.500000      19.000000
max          NaN  954201.000000      40.000000
```

[101]:
```
df.head()
```

[101]:
```
  Nama Kota  Populasi  Jumlah Kampus
0   Bandung    448094             20
1   Jakarta    954201             40
2     Batam     32391             15
3     Depok    401923             10
4    Bekasi    894201             18
```

[102]:
```
df.tail()
```

[102]:
```
  Nama Kota  Populasi  Jumlah Kampus
2     Batam     32391             15
3     Depok    401923             10
4    Bekasi    894201             18
```

```
5    Cimahi     34572              8
6     Bogor     50932             12
```

[103]:
```
df.head(6)
```

[103]:
```
  Nama Kota  Populasi  Jumlah Kampus
0   Bandung    448094             20
1   Jakarta    954201             40
2     Batam     32391             15
3     Depok    401923             10
4    Bekasi    894201             18
5    Cimahi     34572              8
```

data california housing data: https://dl.google.com/mlcc/mledu-datasets/california_housing_train.csv

[105]:
```
df = pd.read_csv("california_housing_train.csv")
```

[106]:
```
df
```

[106]:
```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
0         -114.31     34.19                15.0       5612.0          1283.0
1         -114.47     34.40                19.0       7650.0          1901.0
2         -114.56     33.69                17.0        720.0           174.0
3         -114.57     33.64                14.0       1501.0           337.0
4         -114.57     33.57                20.0       1454.0           326.0
…             …         …                   …            …               …
16995     -124.26     40.58                52.0       2217.0           394.0
16996     -124.27     40.69                36.0       2349.0           528.0
16997     -124.30     41.84                17.0       2677.0           531.0
16998     -124.30     41.80                19.0       2672.0           552.0
16999     -124.35     40.54                52.0       1820.0           300.0

       population  households  median_income  median_house_value
0          1015.0       472.0         1.4936             66900.0
1          1129.0       463.0         1.8200             80100.0
2           333.0       117.0         1.6509             85700.0
3           515.0       226.0         3.1917             73400.0
4           624.0       262.0         1.9250             65500.0
…             …           …              …                  …
16995       907.0       369.0         2.3571            111400.0
16996      1194.0       465.0         2.5179             79000.0
16997      1244.0       456.0         3.0313            103600.0
16998      1298.0       478.0         1.9797             85800.0
16999       806.0       270.0         3.0147             94600.0

[17000 rows x 9 columns]
```

[107]:
```
df.head()
```

[107]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
0	-114.31	34.19	15.0	5612.0	1283.0	
1	-114.47	34.40	19.0	7650.0	1901.0	
2	-114.56	33.69	17.0	720.0	174.0	
3	-114.57	33.64	14.0	1501.0	337.0	
4	-114.57	33.57	20.0	1454.0	326.0	

	population	households	median_income	median_house_value
0	1015.0	472.0	1.4936	66900.0
1	1129.0	463.0	1.8200	80100.0
2	333.0	117.0	1.6509	85700.0
3	515.0	226.0	3.1917	73400.0
4	624.0	262.0	1.9250	65500.0

[ ]:
```

```