DigitalSkola
Uncover The World of Digital Skills with Us

Advanced DataFrame

Introduction to API

Basic Statistics

Learning Progress Review Week 9

By:
MARVEL TEAM
Fikrie | Natalia | Satria

1. *Advanced DataFrame*

Join DataFrame

Join DataFrame di maksudkan jika kita ingin mengambil *value* dari *DataFrame* lain dengan mengandalkan sebuah ID dan *Primary Key* yang ada di masing-masing *DataFrame*.

```
# merge() for combining data on common columns or indices
df = pd.merge(df1, df2, how="left", on=["col1", "col2"])

# .join() for combining data on a key column or an index
df = df1.join(df2, how="left", on=["col1", "col2"])
```

Concatenate DataFrame

Concat adalah salah satu metode untuk melakukan penggabungan data berdasarkan sumbu tertentu.

Fungsi **concat()** mampu menyatukan DataFrame dengan kolom dalam urutan yang berbeda.

```
# axis = 1 is horizontally (columns)
# axis = 0 is vertically (rows)
df = pd.concat([df1, df2], axis=1)
```

Append DataFrame

Fungsi utama **append()** adalah pada DataFrame adalah untuk menambahkan baris baru pada sebuah list atau DataFrame itu sendiri.

Jika object atau kolom di dalam append tersebut tidak sama dengan tabel DataFrame yang pertama maka secara otomatis akan menambahkan kolom baru dan akan menghasilkan missing value, karena kolom tidak sama antara Tabel 1 dan Tabel 2.

```
df = df1.append(df2)
```

Indexing DataFrame

Akses DataFrame sering juga di kenal **indexing** atau subset selection. Secara sederhana kita memilih suatu data dari baris tertentu dan column tertentu.

Fungsi **reset_index**() pada DataFrame Pandas digunakan untuk mengembalikan index DataFrame ke – 0.

```
df.reset_index()
```

Pivoting Table

Pivoting adalah suatu pengubahan bentuk data dengan memutar data yang terletak di baris menjadi di column.

Kita juga dapat mengatakan **pivoting** tabel adalah melakukan perubahan pada bentuk data dari yang tadinya Panjang menjadi lebar.

```
table = pd.pivot_table(df, values=['D', 'E'], index=['A', 'C'],
                       aggfunc={'D': np.mean, 'E': np.mean}, fill_value=0)
```

Melting Table

Melting adalah Unpivoting, melting mengubah suatu data dengan memutar dari yang semula berada di posisi column menjadi di posisi row.

Melting mengubah data yang tadinya lebar menjadi Panjang.

```
d1 = {"Name": ["Pankaj", "Lisa", "David"],
      "ID": [1, 2, 3],
      "Role": ["CEO", "Editor", "Author"]}
df = pd.DataFrame(d1)
df_melted = pd.melt(df, id_vars=["ID"], value_vars=["Name", "Role"])
```

Lambda Function

Ekspresi **Lambda** adalah fungsi anonym yang berbentuk 1 baris.

Contoh : (lambda arguments : expression).

Misalnya kita ingin membuat fungsi untuk melakukan penambahan terhadap suatu variable.

Contoh : (lambda x : x + x)

```
x = lambda a : a + 10
print(x(5))
```

Lambda Function

Ekspresi **Lambda** sangat cocok digunakan untuk manipulasi List, Series, dan DataFrame. Fungsi Map() dan Apply() akan sangat memudahkan dalam melakukan operasi Lambda.

- Fungsi **Map()**, hanya bekerja pada Pandas Series. Jadi kita hanya bisa mentransformasikan data untuk satu kolom saja.
- Fungsi **Apply()**, dapat bekerja pada pandas Series dan DataFrame. Jadi kita bisa melakukan transformasi data lebih dari satu kolom.

Lambda Function

```
# creating and initializing a list
values= [['Rohan',455],['Elvish',250],['Deepak',495],
         ['Soni',400],['Radhika',350],['Vansh',450]]

# creating a pandas dataframe
df = pd.DataFrame(values,columns=['Name','Total_Marks'])

# Applying lambda function to find
# percentage of 'Total_Marks' column
# using df.assign()
df = df.assign(Percentage = lambda x: (x['Total_Marks'] /500 * 100))
```

2. *Introduction to API*

Apa itu API?

API adalah akronim dari *Application Programming Interface*.
Pada dasarnya adalah syntax codes, yang memiliki kegunaan khusus, berbentuk *software intermediary* yang mempunyai fungsi untuk menjadi perantara dua aplikasi lainnya untuk saling berbicara.

Kinerja API

API adalah pemberi pesan yang mengambil permintaan dari *user* dan memberitahukan ke sistem mengenai apa yang diminta *user*. Sistem kemudian memberikan respon yang kemudian dibawa oleh API kepada *user*.

Contoh API salah satunya adalah ketika kita menggunakan aplikasi cuaca di *handphone*.

Manfaat API

1. **Otomatisasi** : kerja menjadi lebih terstruktur, lebih cepat dan produktif.

2. **Mendapatkan lebih banyak data** : bagi *Data Scientist* dapat *scrapping* data pada website-website penyedia data.

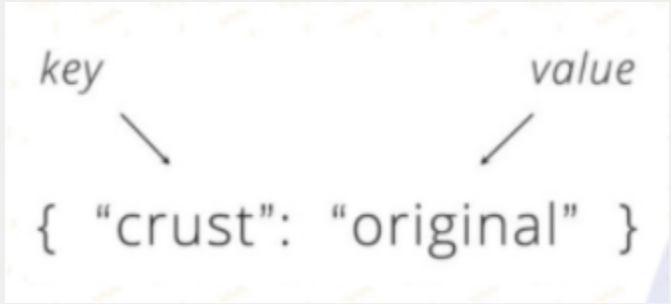
3. **Integrasi** : memungkinkan aplikasi dapat dibuka pada berbagai macam platform.

4. **Personalisasi** : *user*/perusahaan dapat melakukan *customisasi* konten dan servis pada aplikasi.

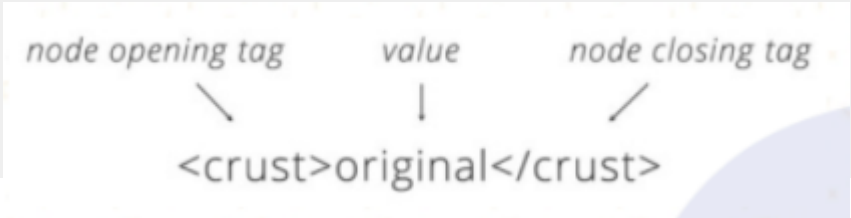
Data & API di Python

- Terdiri dari 2 jenis data yaitu :

1. **JSON**



2. **XML**



- API di Python dapat diunakan denan berbaai libraries, sepeti requests, flask, fastapi, djano

Peran HTTP, APIs & REST

- API adalah *web service* yang memungkinkan untuk mengakses data & metode spesifik melalui protokol HTTP yang standar seperti website biasanya. Kesederhanaan ini memudahkan interasi APIs ke dalam berbagai jenis aplikasi.

- Salah satu pendekatan APIs adalah REST (*Representational State Transfer*). Salah satu arsitektural APIs yan paling popular di *web service*. REST terdiri dari *guidelines* untuk memudahkan komunikasi antara klien & *server communication*. REST APIs membuat akses data lebih logis.

Komponen API

- **Endpoint :** kode baris panjang, gabungan *simbol, numeric* dan *alphabet* setelah tanda *slice* untuk memberikan karakteristik data.
- **Data :** jika menggunakan metode yang memungkinkan perubahan data, kita membutuhkan *payload data* yang memungkinkan data untuk diubah atau dibuat.
- **Headers** : metadata yang harus dimasukkan ke dalam *request*, terletak di bagian depan url.
- **Method :** Interaksi spesifik sesuai denan tujuan awal, seperti *Create, Read, Update & Delete*. Metode umum di Rest APIs : GET, PUT, POST, DELETE.

Status API

Jenis – jenis status ketika kita mengajukan API *requests* :

- *Informational responses* (‘100’ - ’199’)
- *Successful responses* (‘200’ - ’299’)
- *Redirects* (‘300’ - ’399’)
- *Client errors* (‘400’ - ’499’)
- *Servers errors* (‘500’ – ‘599’)

Processing Data API

Setelah mendapatkan data dari API, kita perlu untuk memproses data tersebut.

Umumnya *processing* data menggunakan dua pendekatan yakni,

- *cross join*
- *for loop*

Contoh API

```
[ ] import pandas as pd, requests
```

```
[ ] response = requests.get('http://api.open-notify.org/astros.json')
    print(response.status_code)
```

```
<Response [200]>
```

```
[ ] data
```

```
{'message': 'success',
 'number': 7,
 'people': [{'craft': 'ISS', 'name': 'Sergey Ryzhikov'},
  {'craft': 'ISS', 'name': 'Kate Rubins'},
  {'craft': 'ISS', 'name': 'Sergey Kud-Sverchkov'},
  {'craft': 'ISS', 'name': 'Mike Hopkins'},
  {'craft': 'ISS', 'name': 'Victor Glover'},
  {'craft': 'ISS', 'name': 'Shannon Walker'},
  {'craft': 'ISS', 'name': 'Soichi Noguchi'}]}
```

Processing Data

DigitalSkola
Uncover The World of Digital Skills with Us

```
message = data['message']
number = data['number']
people = data['people']

processed_data = []
for person in people:
    craft = person['craft']
    name = person['name']
    processed_data.append([message, number, craft, name])

print(processed_data)
```

```
[['success', 7, 'ISS', 'Sergey Ryzhikov'], ['success', 7, 'ISS', 'Kate Rubins'], ['success', 7, 'ISS', 'Sergey Kud-Sverchkov'], ['success', 7, 'I
```

```
[ ] df1 = pd.DataFrame(processed_data, columns=['message', 'number', 'craft', 'name'])
    df1
```

	message	number	craft	name
0	success	7	ISS	Sergey Ryzhikov
1	success	7	ISS	Kate Rubins
2	success	7	ISS	Sergey Kud-Sverchkov
3	success	7	ISS	Mike Hopkins
4	success	7	ISS	Victor Glover
5	success	7	ISS	Shannon Walker
6	success	7	ISS	Soichi Noguchi

3. *Basic Statistics*

Apa itu statistika?

Statistika adalah ilmu yang mempelajari tentang perencanaan, pengumpulan, pengolahan, analisis, interpretasi, dan presentasi data.

Bagaimana cara menarik kesimpulan?

Terdiri dari 2 metode yaitu :

1. **Statistika Deskriptif** : metode atau cara mendeskripsikan, menggambarkan, menjabarkan atau menguraikan data. *Data scientist* menggunakan metode ini.

2. **Statistika Inferensia** : metode atau cara penarikan kesimpulan berdasarkan data yang diperoleh dari sampel untuk menggambarkan karakteristik dari suatu populasi.

Populasi dan Sampel

Populasi adalah keseluruhan pengamatan atau objek yang menjadi perhatian.

Karakterisik untuk populasi disebut parameter.

Sampel adalah bagian dari populasi yang menjadi perhatian.

Karakteristik untuk sampel disebut **statistik.**

Tipe –tipe Data

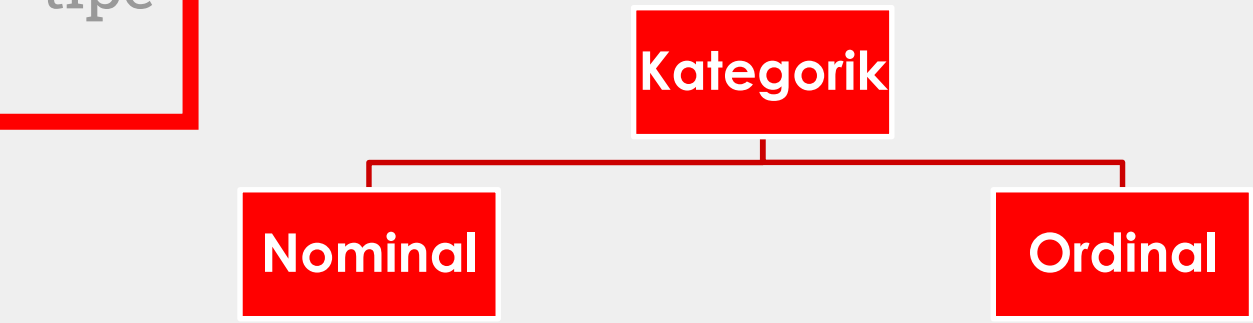
Data adalah satuan unit informasi. Data dapat berupa angka dapat pula bukan berupa angka.

Tipe data dibagi menjadi 2 yaitu :

1. **Kategorik**
2. **Numerik**

Tipe –tipe Data

Tipe –tipe Data



Sifat data **nominal**:
- Hanya sebagai kategori
- Tidak memiliki satuan
- Tidak bisa diurutkan
- Tidak dapat dilakukan operasi matematika

Contoh : Jenis kelamin, alamat pelanggan, menu makanan

Sifat data **ordinal**:
- Hanya sebagai kategori
- Tidak memiliki satuan
- Bisa diurutkan
- Tidak dapat dilakukan operasi matematika

Contoh : Tingkat Pendidikan, rating kredit, ukuran baju

Tipe –tipe Data

Sifat data **diskret**:
- Berupa angka
- Memiliki satuan
- Dapat dihitung dengan dicacah

Contoh : Jumlah murid, jumlah kendaraan

Sifat data **kontinu**:
- Berupa angka
- Memiliki satuan
- Tidak dapat dihitung dengan dicacah
- Bisa digambarkan secara interval

Contoh : Temperatur, Tinggi badan, kecepatan

Statistika Deskriptif

Terbagi menjadi dua pengukuran, yaitu :

***Measure of Central Tendency* (Ukuran Pemusatan)**

***Measure of Spread* (Ukuran Penyebaran)**

Statistika Deskriptif

Measure of Central Tendency (Ukuran Pemusatan)

1. ***Mean* (Rata-rata)** : Penjumlahan dari setiap nilai dibagi dengan banyaknya data. Rata-rata sangat sensitif terhadap *outliers* yang adalah nilai sangat besar sekali atau kecil sekali dalam data. Tidak disarankan menggunakan outliers untuk menghitung rata-rata. Sifat ini dinamakan not robust.

2. **Median** : Titik tengan data yang terletak 50% dari keseluruhan data.

3. ***Mode* (Modus)** : data yang memiliki jumlah frekuensi paling tinggi

Measure of Spread (Ukuran Penyebaran)

1. ***Range* (Jangkauan)** : Jarak antara nilai maksimum dan nilai minimum. Jangkauan sangat sensitif terhadap outliers (sama seperti rata-rata).

2. ***Variance* (Ragam)** : Rata-rata kuadrat selisih dari mean. Ragam juga sangat sensitif terhadap outliers (sama seperti rata-rata).

Sample Variance	Sample Standard Deviation
$s^2 = \frac{\sum(x - \bar{x})^2}{n - 1}$	$s = \sqrt{\frac{\sum(x - \bar{x})^2}{n - 1}}$

3. ***Interquartile* (Interkuartil)** : jarak antara kuartil 3 (Q3) dan kuartil 1 (Q1)

Special Thanks to :

Slide template by SlideCarnival