Handling Missing Data

DQLab LiveClass

Outline

- Ukuran pemusatan data: mean, median, modus
- Data imputation

Pemusatan data

- Ukuran pemusatan data (measures of central tendency) adalah nilai yang digunakan untuk menggambarkan sekumpulan data dengan mengidentifikasikan pusat dari kumpulan data tersebut.
- Ukuran pemusatan data yang paling sering digunakan adalah rata-rata (mean) , median, dan modus (mode).

Ukuran pemusatan data: Mean

- Rata-rata atau mean adalah hasil bagi antara jumlah (sum) nilai dibagi dengan banyaknya nilai
- Contoh, mean dari rangkaian bilangan berikut:

50,70,90,60,50,65,100,70,70,55,90

Adalah

(50+70+90+60+50+65+100+70+70+55+90)/11 = 70

- Untuk menghitung mean dari suatu kolom pada dataframe dapat menggunakan

df['nama_kolom'].mean()

Ukuran pemusatan data: Median

- Median adalah suatu nilai yang terletak di tengah kelompok data yang telah diurutkan dari nilai terkecil sampai terbesar atau sebaliknya.
- Contoh perhitungan median, perhatikan barisan bilangan berikut

50,70,90,60,50,65,100,70,70,55,90

Setelah diurutkan, bilangan tersebut menjadi

50,50,55,60,65,70,70,70,90,90,100

Diperoleh mediannya adalah 70 (data ke-6)

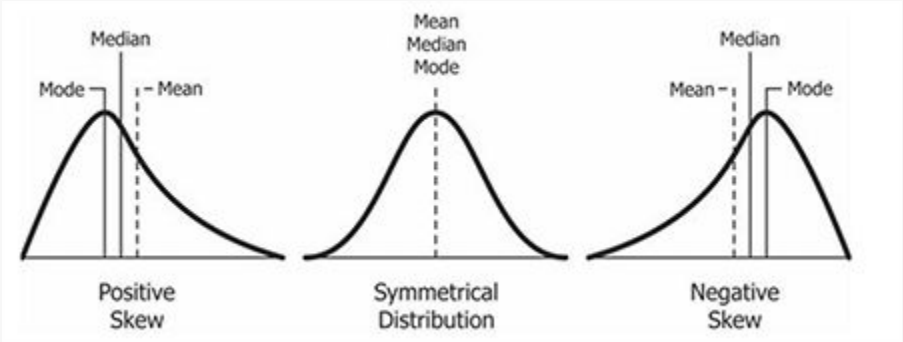
- Jika banyaknya data adalah genap maka nilai median adalah rata-rata dari dua bilangan yang berada di tengah
- Untuk menghitung median dari suatu kolom pada dataframe dapat menggunakan df['nama_kolom'].median()

Ukuran pemusatan data: Modus

- Modus adalah data atau nilai yang paling sering muncul
- Contoh, dari data berikut: 50,50,55,60,65,70,70,70,90,90,100 diperoleh nilai modus 70 (muncul tiga kali)
- Untuk menghitung modus dari suatu kolom pada dataframe dapat menggunakan df['nama_kolom'].mode()

Mean vs Median

- Mean sangat rentan terhadap pengaruh pencilan (outlier) sehingga nilai mean tidak cukup baik untuk menggambarkan pusat data untuk data yang asimetris
- Karena nilai median tidak ditentukan oleh perhitungan melainkan posisi data, pencilan tidak memiliki dampak cukup besar terhadap nilai median.



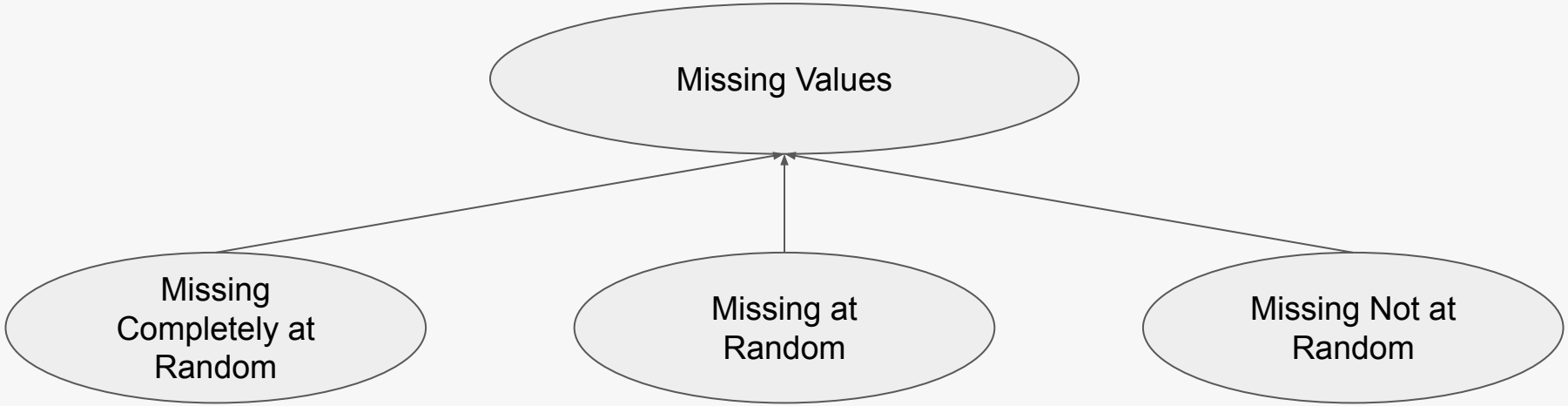
Missing values

- Missing values adalah kondisi di mana suatu variable tidak memiliki nilai
- Pada dataframe, missing values biasanya ditandai dengan NaN

	Part	Feature	Price
0	Monitor	LED	12,000
1	CPU	i7	35000
2	NaN	RGB	NaN
3	Mouse	Wireless	1200
4	NaN	Zebronics	NaN
5	Extentions	NaN	250
6	Table	Urban clap	7000
7	Chair	Apex Chairs	12000
8	Wifi Connection	Airtel	799

Missing values ditandai dengan NaN

Tipe-tipe missing values



Tipe-tipe missing values: MCAR

- **MCAR** (Missing Completely at random), adalah kondisi di mana missing values terjadi secara acak. Terjadinya missing values tidak memiliki hubungan dengan data lain.
  - Penyebab dari tipe ini antara lain: human/system error
  - Contoh: Seorang pustakawan secara acak lupa menginput tanggal pengembalian buku

Tipe-tipe missing values: MAR

- **MAR** (Missing At Random), adalah kondisi di mana terdapat hubungan antara kemunculan missing values dengan variable lain atau dengan kata lain pola kemunculan missing values dapat dijelaskan dari variabel lain
    - Contoh: Terdapat responden yang mengisi gender dengan 'Perempuan', saat diminta memasukkan angka usia, responden tersebut mengosongkan jawaban misalkan karena perempuan lebih sensitif jika ditanya mengenai usia sehingga lebih memilih tidak menjawab

Tipe-tipe missing values: MNAR

- **MNAR** (Missing Not At Random), adalah kondisi di mana pola kemunculan missing values yang tidak bisa dijelaskan oleh variabel lain. Tipe ini biasanya terjadi karena keengganan orang untuk memberikan jawaban atau samping bias
    - Contoh: Orang dengan gaji rendah cenderung untuk tidak mengisi kuesioner

Pentingnya menangani missing values

- Beberapa algoritma tidak dapat digunakan jika dataset mengandung missing values
- Menghasilkan hasil analisis yang bias
- Mengurangi akurasi analisis statistik

Mengetahui keberadaan missing values

```
[2] df = pd.read_csv('titanic.csv')
    df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Misalkan kita memiliki data seperti di atas

df.isnull().sum()

```
[3] df.isnull().sum()

PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

Mengetahui banyaknya missing values di setiap kolom

```
df.isnull().sum() / len(df) * 100.0

PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            19.865320
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.000000
Cabin          77.104377
Embarked        0.224467
dtype: float64
```

Banyaknya missing values dalam persentase

Menampilkan data dengan missing values

Syntax:

```
df[df[<nama_kolom>].isnull() == True]
```

Menampilkan baris dengan missing values pada Age

```
df[df['Age'].isnull() == True]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
17	18	1	2	Williams, Mr. Charles Eugene	male	NaN	0	0	244373	13.0000	NaN	S
19	20	1	3	Masselmani, Mrs. Fatima	female	NaN	0	0	2649	7.2250	NaN	C
26	27	0	3	Emir, Mr. Farred Chehab	male	NaN	0	0	2631	7.2250	NaN	C
28	29	1	3	O'Dwyer, Miss. Ellen "Nellie"	female	NaN	0	0	330959	7.8792	NaN	Q
...	...	...	...	...	...	...	...	...	...	...	...	...
859	860	0	3	Razi, Mr. Raihed	male	NaN	0	0	2629	7.2292	NaN	C
863	864	0	3	Sage, Miss. Dorothy Edith "Dolly"	female	NaN	8	2	CA. 2343	69.5500	NaN	S
868	869	0	3	van Melkebeke, Mr. Philemon	male	NaN	0	0	345777	9.5000	NaN	S
878	879	0	3	Laleff, Mr. Kristo	male	NaN	0	0	349217	7.8958	NaN	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S

Menangani missing values

- Strategi dalam menangani missing values harus didasari dengan memahami alasan di balik kemunculan missing values
- Terdapat 2 strategi menangani missing values:
  - Delesi (hapus)
  - Imputasi (diisi dengan suatu nilai)

Delesi

- Metode ini tidak disarankan terutama untuk tipe MNAR
- Kekurangan dari metode ini adalah adanya kemungkinan data yang penting ikut terhapus
- Delesi dapat dilakukan dengan dua cara
  - Delesi baris (`df.dropna(axis=0)`)
  - Delesi kolom (`df.dropna(axis=1)`)
- `dropna` akan menghilangkan baris/kolom yang mengandung missing values

Delesi baris

```
[7] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Sebelum

```
[6] delete_row = df.dropna(axis=0)
    delete_row.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 183 entries, 1 to 889
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  183 non-null    int64
 1   Survived     183 non-null    int64
 2   Pclass       183 non-null    int64
 3   Name         183 non-null    object
 4   Sex          183 non-null    object
 5   Age          183 non-null    float64
 6   SibSp        183 non-null    int64
 7   Parch        183 non-null    int64
 8   Ticket       183 non-null    object
 9   Fare         183 non-null    float64
 10  Cabin        183 non-null    object
 11  Embarked     183 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 18.6+ KB
```

Sesudah

Delesi kolom

```
[7] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Sebelum

```
delete_col = df.dropna(axis=1)
delete_col.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   SibSp        891 non-null    int64  
 6   Parch        891 non-null    int64  
 7   Ticket       891 non-null    object 
 8   Fare         891 non-null    float64
dtypes: float64(1), int64(5), object(3)
memory usage: 62.8+ KB
```

Sesudah

Imputasi

- Imputasi (imputation) adalah proses mengisi missing values dengan suatu nilai
- Pada pandas, imputasi dapat menggunakan fillna()
- Syntax:

df[<nama_kolom>].fillna(<nilai>)

- <nilai> dapat diganti dengan nilai skalar atau hasil perhitungan seperti mean, min, max, atau modus
- Selain itu, missing values dapat diisi dengan nilai valid sebelumnya atau sesudahnya (backfill vs frontfill)

Contoh

```
df['Age'].fillna(20)

0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888    20.0
889    26.0
890    32.0
```

Mengisi dengan nilai skalar

```
[10] df['Age'].fillna(df['Age'].mean())

0      22.000000
1      38.000000
2      26.000000
3      35.000000
4      35.000000
         ...
886    27.000000
887    19.000000
888    29.699118
889    26.000000
890    32.000000
```

Mengisi dengan nilai fungsi agregat, contoh: mean()

```
df['Embarked'].fillna(df['Embarked'].mode()[0])

0      S
1      C
2      S
3      S
4      S
      ..
886    S
887    S
888    S
889    C
890    Q
```

Mengisi dengan nilai modus

Backfill vs Frontfill

	name	type	AvgBill
0	Foreign Cinema	Restaurant	<NA>
1	Liho Liho	Restaurant	224.0
2	<NA>	bar	80.5
3	<NA>	bar	<NA>
4	<NA>	bar	65.23
5	Blue Barn	<NA>	361.98

Sebelum

```
df['name'].fillna(method='bfill')

0    Foreign Cinema
1         Liho Liho
2         Blue Barn
3         Blue Barn
4         Blue Barn
5         Blue Barn
Name: name, dtype: object
```

Backfill

```
df['name'].fillna(method='ffill')

0    Foreign Cinema
1         Liho Liho
2         Liho Liho
3         Liho Liho
4         Liho Liho
5         Blue Barn
Name: name, dtype: object
```

Frontfill

DQLab

live Class

Terimakasih!

dqlab www.dqlab.id