Nama : Ghina Khoerunnisa
Program : Data Scientist Intern

Data Cleaning with Pandas

Proses data cleaning sangat penting untuk dilakukan karena apabila data yang kita gunakan jelek, maka mau modelnya sebagus apapun hasilnya tetap tidak akan memuaskan. Selain itu proses cleaning data menghabiskan waktu 80% dari seluruh proses yang ada. Yang perlu diperhatikan sebelum membersihkan data adalah memahami datanya seperti apa saja data fiturnya, tipe data, standard missing data, non standard missing data, unexpected missing value, dan bagaimana replace missing value tersebut.

- **EDA yang dilakukan pada dataset property**

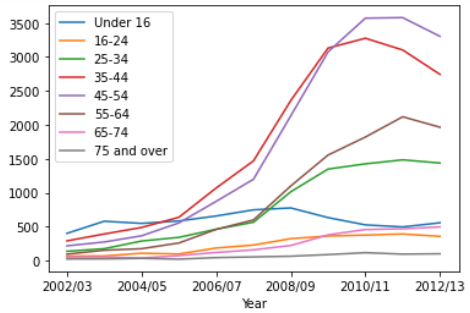
  Untuk melihat fitur-fiturnya dapat memanggil datanya atau 5 data teratas dengan .head(). Dari dataset tersebut diketahui bahwa terdapat fitur PID -> property id (int atau float), ST_NUM -> Street number (int atau float), ST_NAME -> Street Name (string), OWN_OCUPIED -> Ditempati (Y) atau tidak (N) (string), NUM_BEDROOMS -> number of bedrooms (int atau float), NUM_BATH -> number of bathroom (int atau float), SQ_FT -> square foot or area (int atau float). Kemudian untuk melihat dan mengecek tipe datanya dapat menggunakan .info(). Dari data tersebut diketahui bahwa fitur NUM_BEDROOMS, NUM_BATH, dan SQ_FT memiliki tipe data objek yang seharusnya bertipe numerik.

  Selanjutnya adalah adalah mengecek ada atau tidaknya missing values. Pertama, missing values yang dapat dideteksi oleh pandas dapat menggunakan .isnull(). Ini akan mengembalikan boolean, jika True maka value tersebut adalah missing value. Untuk menghitung semua missing value di dataset tersebut dapat menggunakan df.isnull().sum(). Kemudian untuk missing value yang memiliki format yang berbeda dapat dicek seperti gambar di bawah,

```
1 invalid_num_bedrooms = ["na", "--"]
2 df[df['NUM_BEDROOMS'].isin(invalid_num_bedrooms)]
3 # df.drop(df[df['NUM_BEDROOMS'].isin(invalid_num_bedrooms)]
4 # .index)
```

PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
)00.0	213.0	TREMONT	Y	--	1	NaN
)00.0	215.0	TREMONT	Y	na	2	1800

Lalu kita perlu mengecek unexpected missing value, unexpected ini merupakan value yang seharusnya tidak ada di suatu fitur. Misal pada fitur own_occupied dengan menggunakan .value_counts() terdapat value Y, N, dan 12. Value 12 itu yang disebut dengan unexpected missing value. Tetapi ini jika row nya hanya sedikit, jika datanya banyak dan beragam akan sulit untuk menentukannya. Oleh karena itu, kita dapat membuat *block code* dengan perulangan untuk mengecek apabila fitur tersebut harusnya bertipe data integer maka kita lakukan konversi dari string ke integer. Jika dapat dikonversi maka datanya valid tapi jika tidak maka akan diubah menjadi NaN. Contoh lainnya adalah jika suatu fitur memiliki value Y atau N maka dilakukan pengecekan apabila bukan dari kedua nilai tersebut akan diganti menjadi NaN. Untuk menangani missing value dapat menggunakan .fillna() yaitu dengan mengganti NaN menjadi suatu nilai tertentu (value, median, mean, dll). Setelah menangani missing value kita dapat



menganalisis trend yang ada di dataset tersebut. Misalnya kita akan menganalisis dataset obesitas yang mana data tersebut terdiri dari kolom tahun, total, kategori umur. Kemudian agar dapat divisualisasikan set kolom tahun menjadi index. Dari gambar dapat dilihat bahwa yang perlu memperhatikan tingkat obesitasnya adalah orang-orang dengan rentang umur 35-54.

- **Time Series**

Pada data time series kita dapat menjadikan fitur date menjadi index sehingga kita dapat melakukan manipulasi data dengan mudah. Misalnya pada gambar di bawah kita ingin menampilkan data pada tanggal atau bulan atau tahun sekian, menjumlahkan data pada tanggal sekian, atau resample sesuai dengan Day (D) (W -> Week, M -> month) kemudian melakukan aggregation, dan dapat melakukan rolling window.

```
timeseries_df.loc['2020-01-02']
```

```
timeseries_df.loc['2020-01-01']['data'].sum()
```

```
timeseries_df.resample('D').sum()
```

```
### timeseries erat hubungannya dengan rolling window(?)
timeseries_df['rolling_sum'] = timeseries_df.rolling(3).sum()
```

```
opsd_daily.loc['2017-12-01']
```

```
opsd_daily.loc['2017-12']
```

```
opsd_daily.loc['2017']
```

Kita juga bisa memisahkan tanggal, bulan, dan tahun dari indexnya seperti pada gambar di bawah,

```
opsd_daily['Year'] = opsd_daily.index.year
opsd_daily['Month'] = opsd_daily.index.month
opsd_daily['Weekday'] = opsd_daily.index.weekday
```

Selanjutnya kita dapat menganalisis data dengan rentang tertentu atau kondisi tertentu sehingga kita mendapatkan hal yang insightful. Contohnya adalah pada dataset opsd_daily, kita ingin melihat daily production solar energy dan rata-rata tiap minggunya pada tahun 2017 paruh pertama. Dari hasil grafiknya dapat diketahui bahwa dari awal tahun hingga pertengahan tahun, produksi solar terus meningkat hal ini dapat disebabkan oleh faktor cuaca di Jerman yaitu musim panas di pertengahan tahun sehingga produksi solar pada bulan-bulan tersebut tinggi.

```
start, end = '2017-01', '2017-06'
opsd_daily.loc[start:end, 'Solar'].plot(marker='.', linestyle='
opsd_weekly_mean.loc[start:end, 'Solar'].plot(marker='o', marke
plt.title('Solar Energy Production Trend')
plt.show()
```