Nama : Ghina Khoerunnisa Program : Data Scientist Intern

Pandas Introduction

Introduction

Untuk menggunakan pandas dapat dilakukan dengan *import pandas as pd*. Untuk membaca data dalam format csv dapat menggunakan *pd.read_csv(path)*. Sama seperti pada numpy untuk melihat dimensinya dapat menggunakan *df.shape*. Untuk melihat 5 data pertama menggunakan *df.head()*. Untuk melihat 5 data terakhir menggunakan *df.tail()*. Untuk mengkonfigurasi pandas agar menampilkan seluruh kolom menggunakan *pd.set_option("display.max.columns", None)* dan untuk membulatkan kolom yang desimalnya panjang menggunakan *pd.set_option("display.max.columns", 2)*.

• Getting to Know Your Data (Display data types, basic statistic)

Untuk melihat semua kolom dengan tipe datanya menggunakan *df.info()*. Untuk melihat statistik data dasar (mean, std, min, max, dll) dapat menggunakan *df.describe()*. Untuk mendapatkan seberapa sering nilai tertentu muncul dalam suatu kolom dapat menggunakan *df['nama kolom'].value counts()*.

• Data Structure

Series, series mempunyai dua komponen yaitu index dan value. Setiap komponen dapat diakses dengan *series_var.index* dan *series_var.values*. Index pada series dapat berbentuk label.

DataFrame, kumpulan series dengan index yang sama dalam satu tabel dapat disebut sebagai dataframe. Untuk mengakses index dan values sama seperti series hanya saja objek yang dipanggil dataframe. Di dataframe terdapat 2 axis. Axis 0 berarti baris dan axis 1 berarti kolom.

Append

```
1000
                                                                                       100.0
   df_city = df_city.append(pd.DataFrame(
       data=[{"revenue":1000, "employee_count":np.nan}],
                                                                        Sa
                                                                             2000
                                                                                       200.0
       index=["Toronto"]
                                                                       Jpn
                                                                             3000
                                                                                       300.0
4
   ))
                                                                                        NaN
                                                                     Toronto
                                                                             1000
```

• Accessing Element

Loc, mengacu pada label index (indexnya bisa angka ataupun objek). Penggunaannya : $df_city.loc['Jkt']$. Ini akan menampilkan data dari seluruh kolom dengan index 'Jkt'. Contoh lainnya adalah $df.loc[:,['gameorder','game_id']]$. Ini akan menampilkan seluruh baris dengan kolom gameorder dan game_id dari dataframe df. Loc di Series mirip seperti indexing & slicing di dictio-.

Hoc, mengacu pada positional index (walaupun index angkanya tidak teratur, tetap diakses sesuai dengan posisi). Penggunaannya : $df_city.iloc[0]$. Data yang tampil sama saja dengan contoh Loc. Contoh lainnya adalah df.iloc[:,0:2]. Ini akan menampilkan data yang sama dengan contoh lain di bagian loc yaitu semua baris dengan kolom index 0 s/d 1 (gameorder, game_id) dari dataframe df. Iloc di Series mirip seperti indexing & slicing di list.

• Querying, memilih baris berdasarkan nilai di kolom kumpulan data. Ini seperti membuat data frame baru dari indexing dan slicing dengan kondisi. Misalnya *current_decade* = $df[df['year_id'] > 2010]$. Ini akan membuat sebuah data frame baru yaitu current_decade yang berisi data df dengan year_id lebih dari 2010. Dan kondisi disini dapat lebih dari satu kondisi, antar kondisinya dipisah dengan bitwise operator & (and) atau | (or). Contohnya $df_city.loc[(df_city.revenue > 1000) & (df_city['employee_count'] > 200)]$.

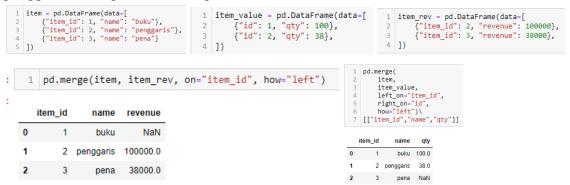
• Groupby and Aggregation

Groupby memungkinkan kita untuk mengelompokkan data dalam kumpulan item yang sama kemudian kita dapat melakukan **aggregation** seperti menjumlahkan, merata-ratakan, menghitung banyaknya, dll.

• Manipulating Columns

Untuk membuat salinan dataframe dapat menggunakan atribut .copy(). Kemudian untuk membuat data baru dari operasi kolom yang telah ada dapat dilakukan dengan df["namakolombaru"] = operasi. Misalnya nba["difference"] = df.pts - df.opp_pts. Ini merupakan kolom perbedaan point antara suatu team dengan team lawan. Kemudian untuk mengganti nama kolom dapat menggunakan .rename(columns={"kolom lama": "kolom baru"}). Untuk mendrop kolom yang tidak dibutuhkan dapat menggunakan atribut .drop(nama_kolom, inplace=True, axis=1). Inplace akan mengubah df nya langsung. Axis = 1 merupakan kolom sehingga drop berdasarkan kolomnya. Jika ada data type yang belum sesuai seperti tanggal bertipe object dapat diubah menjadi datetime dengan pd.to_datetime(). Untuk menghapus kolom/baris yang berisi nan dapat menggunakan dropna() dengan memasukkan axis yang sesuai. Untuk mengubah nan menjadi suatu data dapat menggunakan fillna.

• Concat & Merge, digunakan untuk menggabungkan multiple dataset. Biasanya menggunakan merge. Contoh penggunaan concat, pd.concat([df1, df2], sort=False). Defaultnya menggabungkan dengan axis = 0 (baris). Jika ingin concat kolom maka menggunakan axis = 1. Jika ingin melakukan inner join dapat menggunakan pd.concat([df1, df2], axis=1, join="inner") berarti ini join dengan irisan baris df1 dan df2. Contoh penggunaan merge adalah sebagai berikut,



Visualize

Untuk menampilkan plot langsung di notebook perlu memasukkan *%matplotlib inline* sebelumnya. Untuk membuat plot dapat menggunakan .plot(kind="tipe_plot"). Default kind adalah line, kind yang lainnya adalah bar, hist, box, scatter, dll.