

Introduction to Pandas Dataframe and Exploratory Data Analysis

Artificial Intelligence dan Big Data | AAK2KAB3 | Kur. 2024 | 2024/2025

Overview:

- Introduction to Pandas DataFrame
- Exploratory Data Analysis

01

Introduction to Pandas Dataframe

- » Pandas dalam Python
- » Membuat DataFrame
- » Operasi pada DataFrame



- Pandas adalah pustaka Python yang digunakan untuk manipulasi dan analisis data. Salah satu struktur data utama dalam Pandas adalah **DataFrame**.
- Bersifat *open source* dan tersedia di <https://pandas.pydata.org/>
- Pandas sangat berkaitan dengan NumPy
- Jika library belum terpasang, tuliskan perintah instalasi:
 - pip install pandas
- Kemudian impor:
- import pandas as pd



Data Wrangling / Data Munging

Reshaping (mengubah bentuk data)
Joining (menggabungkan data)
Splitting (pemisahan data)
Time-series analysis (data berkala)

Data Cleansing

Membersihkan data tidak lengkap (*Error*)
Menangani data pencilan (*outliers*)
Menghapus data duplikat

DataFrame

Syntax – Creating DataFrames

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

```
df = pd.DataFrame(  
    {"a" : [4 ,5, 6],  
     "b" : [7, 8, 9],  
     "c" : [10, 11, 12]},  
    index = [1, 2, 3])
```

Specify values for each column.

```
df = pd.DataFrame(  
    [[4, 7, 10],  
     [5, 8, 11],  
     [6, 9, 12]],  
    index=[1, 2, 3],  
    columns=['a', 'b', 'c'])
```

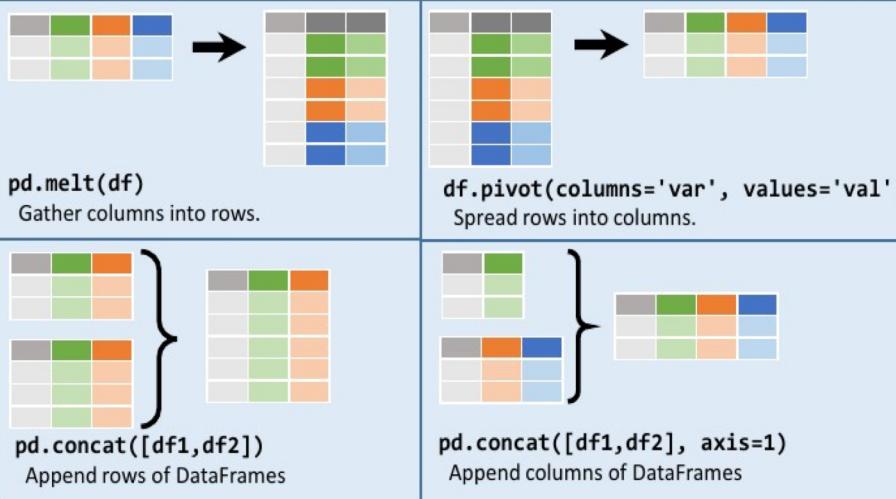
Specify values for each row.

		a	b	c
n	v			
d	1	4	7	10
e	2	5	8	11
		6	9	12

```
df = pd.DataFrame(  
    {"a" : [4 ,5, 6],  
     "b" : [7, 8, 9],  
     "c" : [10, 11, 12]},  
    index = pd.MultiIndex.from_tuples(  
        [('d',1),('d',2),('e',2)],  
        names=['n', 'v']))
```

Create DataFrame with a MultiIndex

Reshaping Data – Change the layout of a data set



df.sort_values('mpg')
Order rows by values of a column (low to high).

df.sort_values('mpg', ascending=False)
Order rows by values of a column (high to low).

df.rename(columns = {'y':'year'})
Rename the columns of a DataFrame

df.sort_index()
Sort the index of a DataFrame

df.reset_index()
Reset index of DataFrame to row numbers, moving index to columns.

df.drop(columns=['Length', 'Height'])
Drop columns from DataFrame

Subset Observations (Rows)



Subset Variables (Columns)



df[df.Length > 7]
Extract rows that meet logical criteria.

df.drop_duplicates()
Remove duplicate rows (only considers columns).

df.head(n)
Select first n rows.
df.tail(n)
Select last n rows.

df.sample(frac=0.5)
Randomly select fraction of rows.

df.sample(n=10)
Randomly select n rows.

df.iloc[10:20]
Select rows by position.
df.nlargest(n, 'value')
Select and order top n entries.
df.nsmallest(n, 'value')
Select and order bottom n entries.

df[['width', 'length', 'species']]
Select multiple columns with specific names.

df['width'] or df.width
Select single column with specific name.

df.filter(regex='regex')
Select columns whose name matches regular expression regex.

Operasi Dasar pada DataFrame

Melihat Data:

- **df.head()** untuk melihat 5 baris pertama
- **df.tail()** untuk melihat 5 baris terakhir
- **df.shape** untuk mendapatkan ukuran DataFrame (jumlah baris dan kolom)
- **df.info()** untuk informasi umum tentang DataFrame (tipe data, jumlah nilai non-null)

Pemilihan Kolom:

```
df['Name'] # Memilih kolom tertentu  
df[['Name', 'Age']] # Memilih beberapa kolom
```

Pemilihan Baris:

```
df.iloc[0] # Mengakses baris pertama  
df.loc[0] # Mengakses baris pertama berdasarkan index label
```

Memuat Data ke Pandas

```
df.head()
```

- Method head() dan tail() pada DataFrame membantu kita menampilkan 5 beberapa baris pertama/terakhir dari data yang kita muat.

```
df.head(3)
```

Unnamed: 0	id	player_name	games	time	goals	xG	assists
0	0	647	Harry Kane	35	3097	23	22.174859
1	1	1250	Mohamed Salah	37	3085	22	20.250847
2	2	1228	Bruno Fernandes	37	3117	18	16.019454

Unnamed: 0	id	player_name	games	time	goals	xG	assists
0	0	647	Harry Kane	35	3097	23	22.174859
1	1	1250	Mohamed Salah	37	3085	22	20.250847
2	2	1228	Bruno Fernandes	37	3117	18	16.019454
3	3	453	Son Heung-Min	37	3139	17	11.023287
4	4	822	Patrick Bamford	38	3085	17	18.401863

Memuat Data ke Pandas

- Nyalakan Jupyter Notebook di folder kerja Anda.
- Buka atau buat baru suatu skrip ipynb (Python 3)
- Import pandas dan numpy. (Pastikan sudah terinstal sebelumnya).
- Load file CSV yang sudah diunduh sebelumnya (pada contoh "Mengambil Data secara Manual") ke dalam sebuah DataFrame
 - Gunakan perintah `read_csv(...)`

```
1 import pandas as pd
```

```
1 dataset = 'cloth_data.csv'
```

```
1 df = pd.read_csv(dataset)
```

02

Exploratory Data Analysis

» Langkah-Langkah dalam EDA

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) adalah langkah awal dalam analisis data yang bertujuan untuk memahami struktur dan pola dalam dataset. Proses ini melibatkan teknik statistik dan visualisasi data untuk menggali insight lebih dalam.

- **Tujuan EDA**
- Memahami distribusi data.
- Menemukan pola, hubungan, dan anomali dalam data.
- Mengidentifikasi outliers dan missing values.
- Menyusun hipotesis untuk analisis lebih lanjut.

Tipe-tipe data dari setiap kolom

- Atribut `dtypes` pada DataFrame berisi tipe data dari setiap kolom.
- Lihat Pandas User Guide untuk detil setiap tipe.
- `dtype:object` di akhir output `dtypes` mewakili Series yang merupakan objek Python yang dikembalikan oleh `dtypes` itu sendiri (bukan bagian dari tipe kolom manapun).
- Dalam contoh ini, terlihat bahwa 2 kolom pertama hanyalah ID numerik yang biasanya tidak memiliki makna riil. Maka, kita ambil bagian DataFrame mulai dari kolom "player_name" (untuk *zero-based index*, kita pakai kolom ke-2 dst).

```
print(df.dtypes)
```

```
Unnamed: 0          int64
id                int64
player_name       object
games             int64
time              int64
goals              int64
xG                float64
assists            int64
xA                float64
shots              int64
key_passes         int64
yellow_cards      int64
red_cards          int64
position           object
team_title         object
npg               int64
npxG              float64
xGChain            float64
xGBuildup          float64
dtype: object
```

```
df_noid = df.iloc[:,2:]
df_noid
```

	player_name	games	time	goals	xG	assists	xA
0	Harry Kane	35	3097	23	22.174859	14	7.577094
1	Mohamed Salah	37	3085	22	20.250847	5	6.528526
2	Bruno Fernandes	37	3117	18	16.019454	12	11.474996
3	Son Heung-Min	37	3139	17	11.023287	10	9.512992
4	Patrick Bamford	38	3085	17	18.401863	7	3.782247
...
517	Jaden Philogene-Bidace	1	1	0	0.000000	0	0.000000
518	Gaetano Berardi	2	113	0	0.074761	0	0.000000
519	Anthony Elanga	1	67	0	0.000000	0	0.000000
520	Femi Seriki	1	1	0	0.000000	0	0.000000

Langkah-langkah dalam EDA

• Memahami Statistik Deskriptif:

DataFrame method `describe()` menampilkan statistik dasar setiap kolom data yang bertipe numerik, mencakup banyaknya data (**count**), rerata aritmetik (**mean**), simpangan baku (**std**), nilai terkecil (**min**), kuartil pertama (**25%**), kuartil kedua/median (**50%**), kuartil ketiga (**75%**), dan nilai terbesar (**max**).

- **Mean:** Rata-rata
- **Median:** Nilai tengah
- **Mode:** Nilai yang paling sering muncul
- **Standard Deviation:** Penyebaran data
- **Min/Max:** Nilai minimum dan maksimum

```
df.describe() # Menampilkan statistik deskriptif
```

```
df_noid.describe()
```

	games	time	goals	xG	assists	xA	shots	key_passes	yellow_cards	red_cards	npg	
count	522.000000	522.000000	522.000000	522.000000	522.000000	522.000000	522.000000	522.000000	522.000000	522.000000	522.	
mean	19.643678	1420.068966	1.862069	2.000806	1.289272	1.376029	17.379310	12.963602	2.061303	0.091954	1.668582	1.
std	11.619836	1031.604819	3.338851	3.317946	2.083350	1.886510	21.572664	16.164361	2.203661	0.295800	2.909929	2.
min	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	10.000000	470.250000	0.000000	0.074668	0.000000	0.049245	2.000000	1.000000	0.000000	0.000000	0.000000	0.
50%	21.000000	1342.000000	1.000000	0.737295	0.000000	0.691122	10.000000	7.000000	2.000000	0.000000	0.500000	0.
75%	30.000000	2319.000000	2.000000	2.053378	2.000000	2.050509	23.750000	19.000000	3.000000	0.000000	2.000000	1.
max	38.000000	3420.000000	23.000000	22.174859	14.000000	11.474996	138.000000	95.000000	12.000000	2.000000	19.000000	19.

Langkah-langkah dalam EDA

- **Memahami Statistik Deskriptif:**

Gunakan `describe(include='all')` jika ingin menampilkan juga statistik kolom yang bertipe non-numerik, mencakup juga berapa banyak nilai unik dalam kolom (**unique**), nilai modus (**top**), serta frekuensi modus (**freq**).

df_noid.describe(include='all')									ards	position	team_title	npg	npxG	xGChain	xGBuildup	
	player_name	games	time	goals	xG	assists	xA	sI	Na	14	28	NaN	NaN	NaN	NaN	
count	522	522.000000	522.000000	522.000000	522.000000	522.000000	522.000000	522.000000	NaN	M S	Everton	NaN	NaN	NaN	NaN	
unique	522	Nan	Nan	Nan	Nan	Nan	Nan	Nan	NaN	106	28	NaN	NaN	NaN	NaN	
top	Joel Ward	Nan	Nan	Nan	Nan	Nan	Nan	Nan	NaN	954	NaN	NaN	1.668582	1.821450	5.663368	3.455060
freq	1	Nan	Nan	Nan	Nan	Nan	Nan	Nan	NaN	6800	NaN	NaN	2.909929	2.931176	5.600249	3.376584
mean	NaN	19.643678	1420.068966	1.862069	2.000806	1.289272	1.376029	17.379	NaN	NaN	NaN	0.000000	0.000000	0.000000	0.000000	
std	NaN	11.619836	1031.604819	3.338851	3.317946	2.083350	1.886510	21.572	NaN	NaN	NaN	0.000000	0.000000	0.000000	0.000000	
min	NaN	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000	NaN	NaN	NaN	0.000000	0.074668	1.191391	0.720353	
25%	NaN	10.000000	470.250000	0.000000	0.074668	0.000000	0.049245	2.000	NaN	NaN	NaN	0.500000	0.715585	4.252738	2.656397	
50%	NaN	21.000000	1342.000000	1.000000	0.737295	0.000000	0.691122	10.000	NaN	NaN	NaN	2.000000	1.945799	8.308002	5.254647	
75%	NaN	30.000000	2319.000000	2.000000	2.053378	2.000000	2.050509	23.750	NaN	NaN	NaN	19.000000	19.130183	28.968234	18.323006	
max	NaN	38.000000	3420.000000	23.000000	22.174859	14.000000	11.474996	138.000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

Langkah-langkah dalam EDA

- **Menangani Missing Data:**

- Mengidentifikasi Missing Values:

```
df.isnull().sum() # Menampilkan jumlah missing values pada setiap kolom
```

- Mengisi Missing Values:

```
df.fillna(value) # Mengisi missing values dengan nilai tertentu  
df.dropna()      # Menghapus baris dengan missing values
```

Langkah-langkah dalam EDA

- Konsep: Mean (Rerata Aritmetik)

- Nilai rerata yang lazim dipahami kebanyakan orang.
- Rerata aritmetik dari sekumpulan bilangan = jumlah semua bilangan tersebut dibagi dengan banyaknya bilangan dalam kumpulan.
- Diberikan sekumpulan N buah bilangan $S = \{x_1, \dots, x_N\}$, rerata aritmetik μ_S atau \bar{x} dari S didefinisikan sebagai:

$$\mu_S = \bar{x} = \frac{1}{N} \sum_{i=1}^N x_i = \frac{x_1 + \dots + x_N}{N}$$

- Merupakan salah satu ukuran pusat data (tendensi sentral) yang dapat dipakai untuk data bertipe interval dan rasio.
- **Sifat:** total jarak setiap bilangan x_i terhadap rerata aritmetik \bar{x} adalah 0.
- Dapat dipakai sebagai bilangan yang mewakili keseluruhan kumpulan, sepanjang distribusi datanya **tidak** bersifat *skew* (asimetris).

Langkah-langkah dalam EDA

- **Konsep: Standard Deviation (Simpangan Baku)**

- Simpangan baku (*standard deviation*) adalah salah satu ukuran sebaran data.
- Dipakai untuk data bertipe interval dan rasio.
- Untuk kumpulan bilangan $S = \{x_1, \dots, x_N\}$ dengan rerata aritmetik μ_S , simpangan baku σ_S dari S adalah

$$\sigma_S = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu_S)^2} = \sqrt{\frac{(x_1 - \mu_S)^2 + \dots + (x_N - \mu_S)^2}{N}}$$

- Kuadrat dari σ_S , yakni σ_S^2 disebut sebagai **varians**
- Nilai simpangan baku
 - besar = data secara umum tersebar jauh dari nilai rerata aritmetik
 - kecil = data secara umum terkumpul dekat dengan nilai rerata aritmetik
- Simpangan baku dapat pula dipandang sebagai derajat ketidakpastian pengukuran data
 - Contoh: pada pengukuran berulang dengan suatu instrument yang sama, jika simpangan baku data hasil pengukuran bernilai besar, berarti presisi pengukuran rendah.

Langkah-langkah dalam EDA

- **Konsep: Median dan Kuartil**

- Kuartil pertama (Q_1): nilai data sehingga 25% dari keseluruhan data bernilai lebih kecil darinya.
- Kuartil kedua (Q_2) atau median: nilai data sehingga separuh dari data yang ada bernilai lebih kecil darinya.
 - Dapat dipakai sebagai ukuran pusat data (tendensi sentral) sebagai alternatif dari rerata (khususnya jika distribusi data bersifat *skewed*).
- Kuartil ketiga (Q_3): nilai data sehingga 75% dari keseluruhan data bernilai lebih kecil darinya.
- Kuartil dapat dipakai untuk data bertipe ordinal, interval, dan rasio.

Langkah-langkah dalam EDA

- **Konsep: Modus**

- Modus (*mode*): nilai yang paling sering muncul pada sekumpulan data.
- Dipakai sebagai ukuran pusat data (tendensi sentral) untuk data bertipe nominal/kategoris.
 - Tidak dijamin unik dalam suatu distribusi data (bisa ada lebih dari satu modus dalam suatu distribusi).
 - Merupakan nilai yang berpeluang paling tinggi didapatkan ketika data di-*sample*.
- Contoh:
 - Himpunan data {1,2,2,3,4,4,7,8} memiliki dua modus: 2 dan 4.
- Jika data mengikuti distribusi kontinu, misal
 - {0.935, ..., 1.134, ..., 2.643, ..., 3.459, ..., 3.995,}
 - Definisi modus standar menjadi tidak bermakna.
 - Pendekatan 1: lakukan diskretisasi (dibahas di modul Data Preparation), sehingga didapat data bertipe nominal, lalu dicari modusnya.
 - Pendekatan 2: gunakan teknik *kernel density estimation* (tidak dibahas di sini).

Langkah-langkah dalam EDA

- **Grouping dan Agregasi:** Anda bisa menggunakan fungsi **groupby()** untuk mengelompokkan data dan melakukan agregasi.

```
df.groupby('City').mean() # Mengelompokkan berdasarkan 'City' dan menghitung rata-rata
```

```
In [30]: df.groupby('team_title')['goals'].std()  
Out[30]: team_title  
Arsenal                      3.352381  
Arsenal,Brighton              NaN  
Arsenal,Newcastle United       NaN  
Arsenal,West Bromwich Albion  NaN  
Aston Villa                   3.696489  
Aston Villa,Chelsea           NaN  
Brighton                      2.158703  
Burnley                       2.475210  
Chelsea                        2.350177  
Chelsea,Fulham                NaN  
Crystal Palace                 2.901461  
Everton                        3.467727  
Everton,Southampton            NaN  
Fulham                         1.439175  
Leeds                          4.153193  
Leicester                      4.020602  
Liverpool                      4.931439  
Liverpool,Southampton          NaN  
Manchester City                 3.867132  
Manchester United                4.317855  
Newcastle United                 2.483174  
Sheffield United                 1.467599  
Southampton                     3.141941  
Tottenham                       5.855135  
West Bromwich Albion             2.310260  
West Bromwich Albion,West Ham   NaN  
West Ham                        3.369240  
Wolverhampton Wanderers        1.648620  
Name: goals, dtype: float64
```

```
In [29]: df.groupby('team_title')['goals'].mean()  
Out[29]: team_title  
Arsenal                      1.961538  
Arsenal,Brighton              0.000000  
Arsenal,Newcastle United       8.000000  
Arsenal,West Bromwich Albion  0.000000  
Aston Villa                   2.130435  
Aston Villa,Chelsea           3.000000  
Brighton                      1.500000  
Burnley                       1.280000  
Chelsea                        2.240000  
Chelsea,Fulham                1.000000  
Crystal Palace                 1.625000  
Everton                        1.607143  
Everton,Southampton            3.000000  
Fulham                         0.925926  
Leeds                          2.608696  
Leicester                      2.370370  
Liverpool                      2.370370  
Liverpool,Southampton          3.000000  
Manchester City                 3.208333  
Manchester United                2.518519  
Newcastle United                 1.384615  
Sheffield United                 0.666667  
Southampton                     1.555556  
Tottenham                       2.750000  
West Bromwich Albion             1.178571  
West Bromwich Albion,West Ham   0.000000  
West Ham                        2.478261  
Wolverhampton Wanderers        1.222222  
Name: goals, dtype: float64
```

Langkah-langkah dalam EDA

- **Visualisasi Data**

Visualisasi adalah bagian penting dalam EDA untuk memahami distribusi data dan hubungan antar variabel.

- **Histogram:** Untuk melihat distribusi data.

```
# import library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

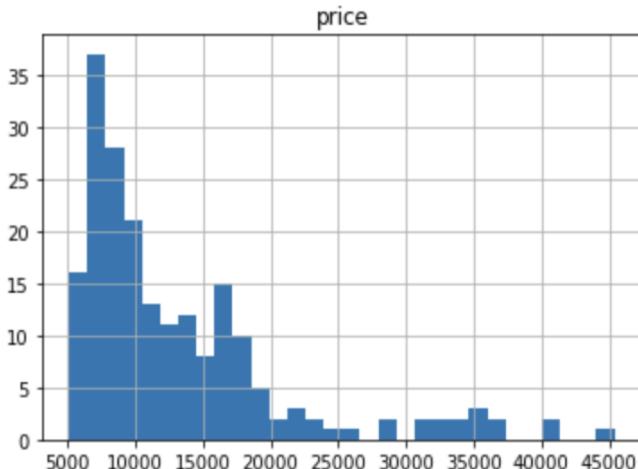


Import library

```
df.hist(column='price', bins=30);
```



Visualisasi histogram
dari kolom “Price”



Langkah-langkah dalam EDA

- Histogram

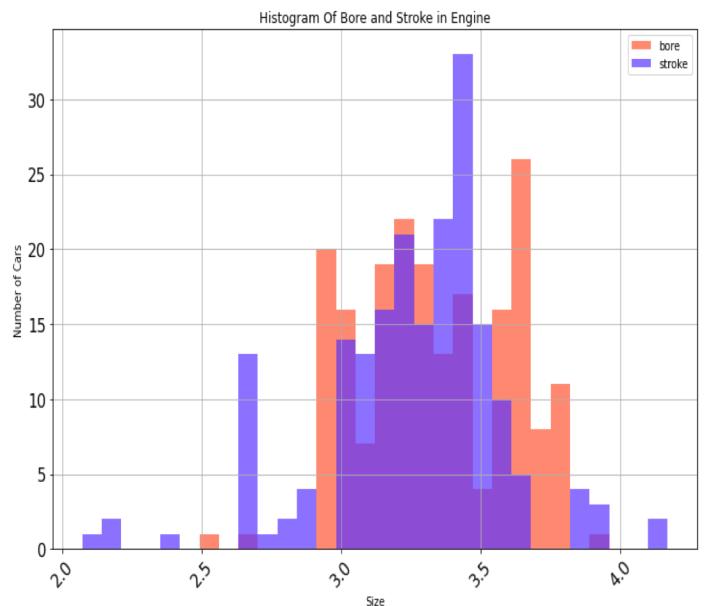
```
# import library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Import library

```
df[['bore','stroke']].plot(kind='hist',
                           alpha=0.7,
                           bins=30,
                           title='Histogram Of Bore and Stroke in Engine',
                           rot=45,
                           grid=True,
                           figsize=(12,8),
                           fontsize=15,
                           color=['#FF5733', '#5C33FF'])
plt.xlabel('Size')
plt.ylabel("Number of Cars");
```

Nama label

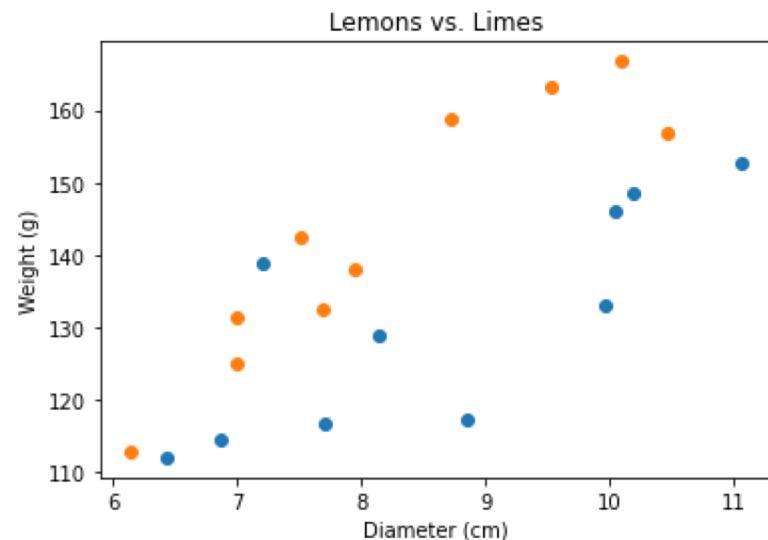
Visualisasi histogram
data "bore" dan
"stroke"



Langkah-langkah dalam EDA

- **Scatter Plot:** Untuk memeriksa hubungan antara dua variabel.

- **Scatter plot** berfungsi baik untuk data dengan dua komponen numerik.
- **Scatter plot** dapat memberikan informasi yang berguna terutama mengenai pola atau penciran.
- Pada contoh di bawah ini, kita memiliki data yang terkait dengan perbedaan lemon dan lime berdasarkan karakteristik fisiologis.
 - Berat (g)
 - Diameter (cm)



```
# import library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

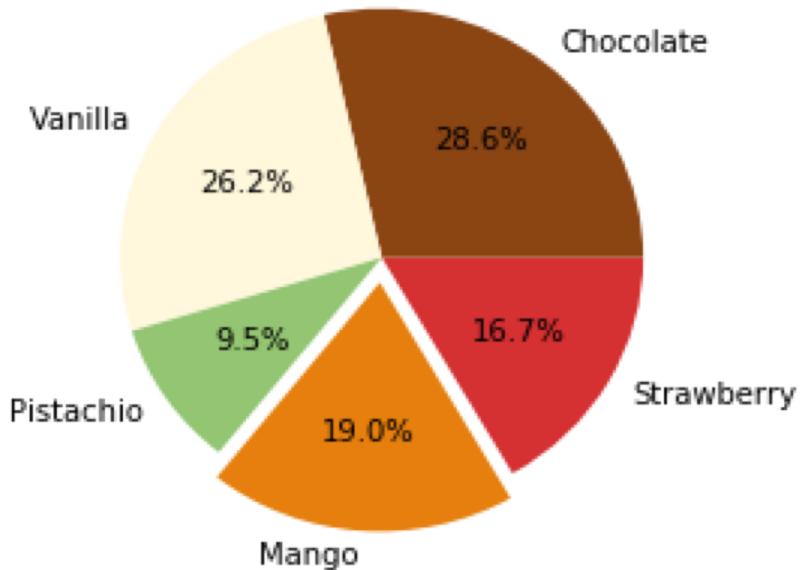
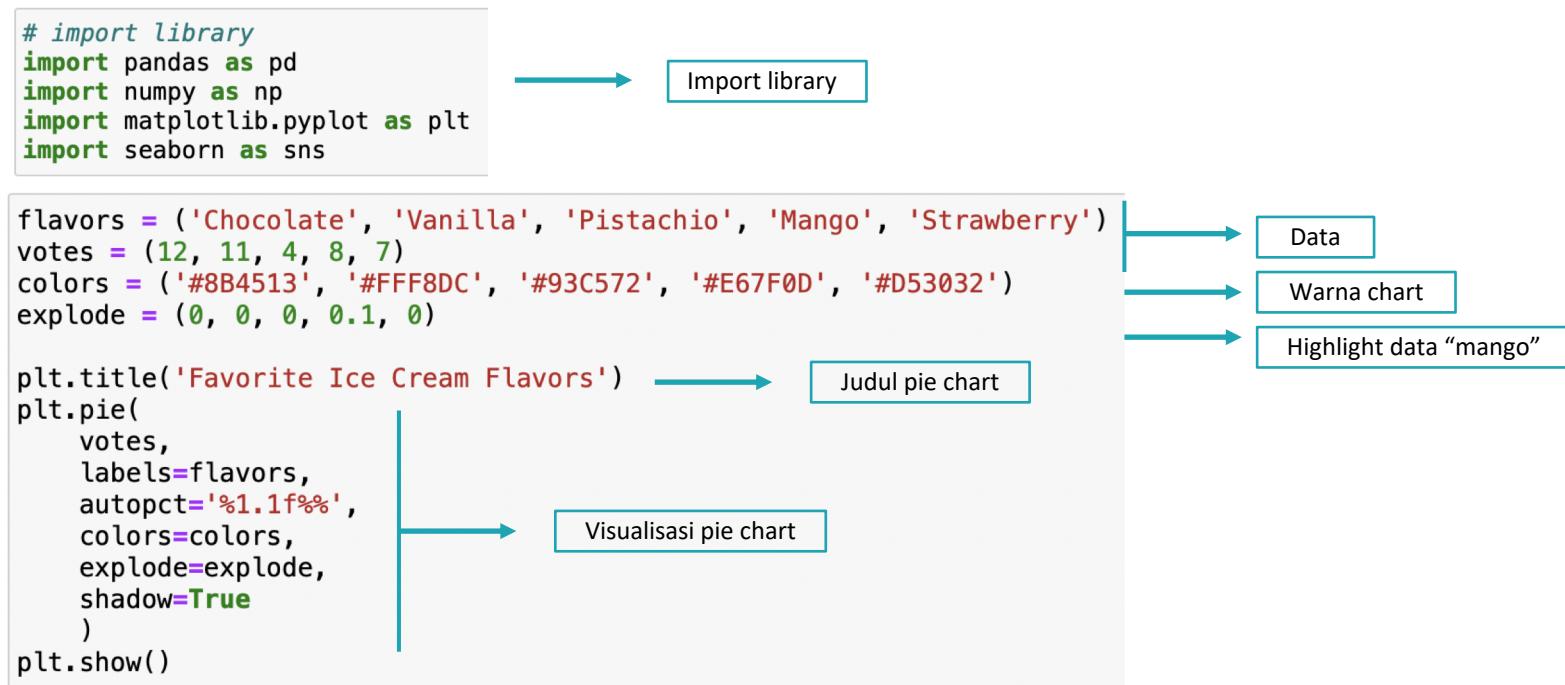
```
lemon_diameter = [6.44, 6.87, 7.7, 8.85, 8.15, 9.96, 7.21, 10.04, 10.2, 11.06]
lemon_weight = [112.05, 114.58, 116.71, 117.4, 128.93,
                132.93, 138.92, 145.98, 148.44, 152.81]
```

```
lime_diameter = [6.15, 7.0, 7.0, 7.69, 7.95, 7.51, 10.46, 8.72, 9.53, 10.09]
lime_weight = [112.76, 125.16, 131.36, 132.41, 138.08,
               142.55, 156.86, 158.67, 163.28, 166.74]
```

```
plt.title('Lemons vs. Limes')
plt.xlabel('Diameter (cm)')
plt.ylabel('Weight (g)')
plt.scatter(lemon_diameter, lemon_weight)
plt.scatter(lime_diameter, lime_weight)
plt.show()
```

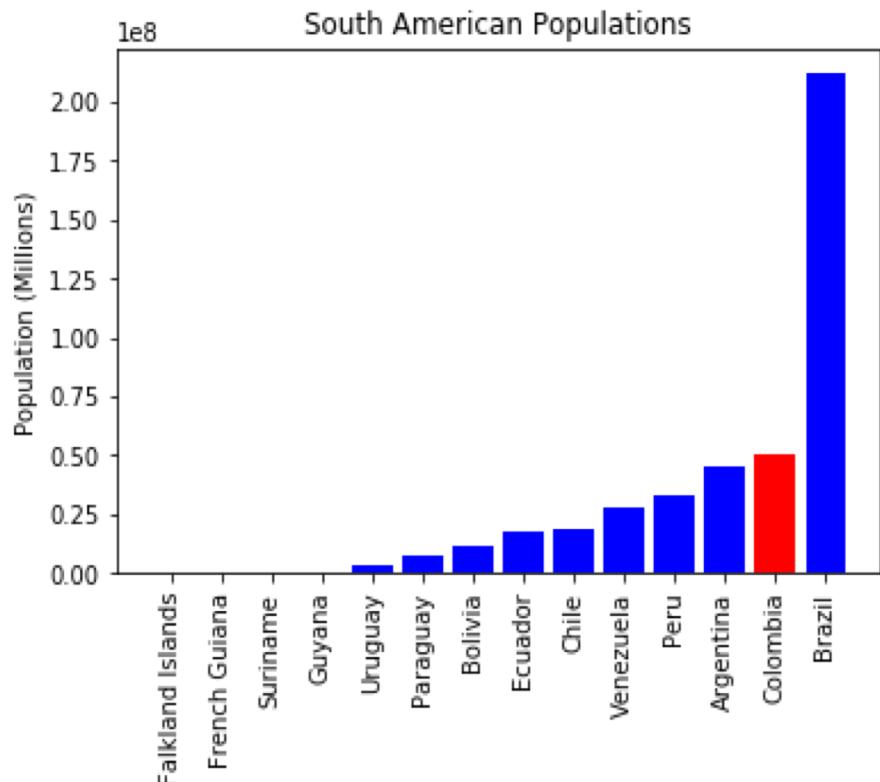
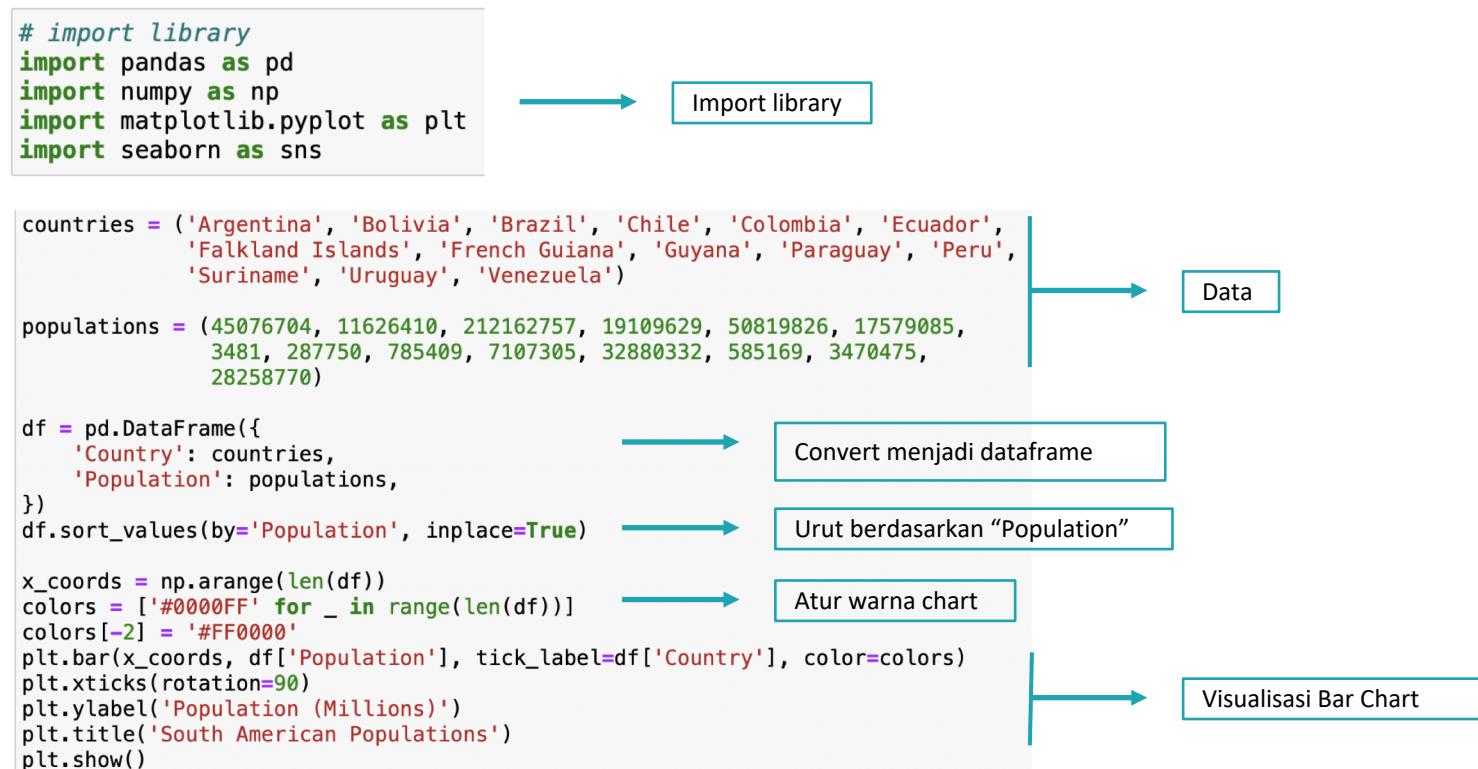
Langkah-langkah dalam EDA

- **Pie Chart:** untuk menunjukkan seberapa banyak dari setiap jenis kategori dalam dataset berbanding dengan keseluruhan.
 - Variabel label berisi tupel rasa es krim
 - Variabel voting berisi tupel voting
 - Data tersebut mewakili jumlah voting rase es krim favorit



Langkah-langkah dalam EDA

- **Bar Chart:** untuk membandingkan data kategorikal.
- Mirip dengan diagram lingkaran, diagram ini dapat digunakan untuk membandingkan kategori data satu sama lain.
- Diagram batang dapat menampilkan lebih banyak kategori data daripada diagram lingkaran.



Langkah-langkah dalam EDA

- **Line Graph:** bentuk visualisasi lainnya selain diagram lingkaran dan diagram batang.
- Diagram garis lebih berguna untuk menunjukkan bagaimana kemajuan data selama beberapa periode.
- Misalnya, grafik garis dapat berguna dalam membuat grafik temperatur dari waktu ke waktu, harga saham dari waktu ke waktu, berat menurut hari, atau metrik berkelanjutan lainnya.

```
# import library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

temperature_c = [2, 1, 0, 0, 1, 5, 8, 9, 8, 5, 3, 2, 2]
hour = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]

plt.plot(
    hour,
    temperature_c,
    marker='x',
)
plt.title('Temperatures in Kirkland, WA, USA on 2 Feb 2020')
plt.ylabel('Temperature Celsius')
plt.xlabel('Hour')
plt.show()
```



Import library



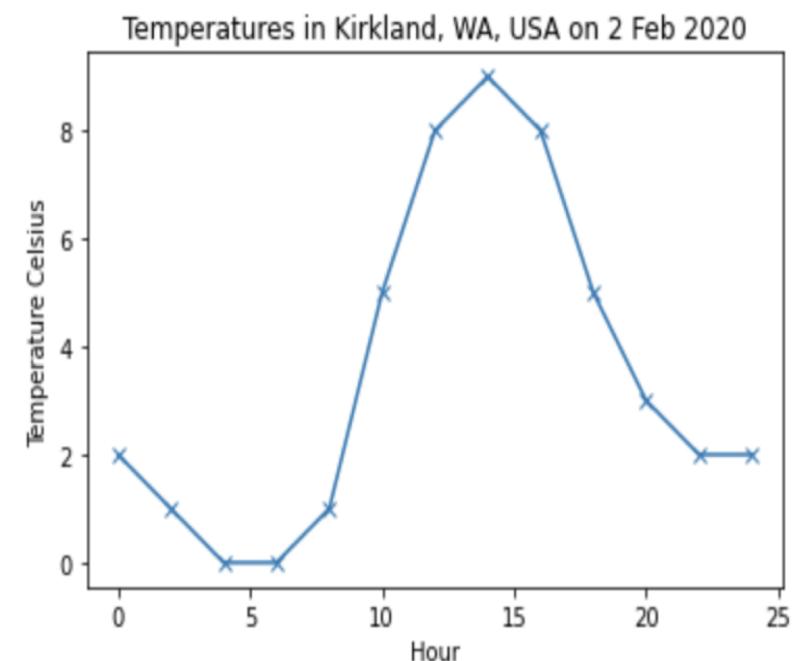
Data



Menambah tanda "x"



Visualisasi linegraph



Fungsi Statistik dalam Pandas

count	Number of non-NA observations
sum	Sum of values
mean	Mean of values
mad	Mean absolute deviation
median	Arithmetic median of values
min	Minimum
max	Maximum
mode	Mode
abs	Absolute Value
prod	Product of values
quantile	Sample quantile (value at %), 1st quartile = quantile(0.25)

std	Bessel-corrected sample standard deviation
var	Unbiased variance
sem	Standard error of the mean
skew	Sample skewness (3rd moment)
kurt	Sample kurtosis (4th moment)
cumsum	Cumulative sum
cumprod	Cumulative product
cummax	Cumulative maximum
cummin	Cumulative minimum

Contoh Fungsi Statistik dalam Pandas

```
df_noid.median()
```

```
games           21.000000
time          1342.000000
goals          1.000000
xG            0.737295
assists        0.000000
xA             0.691122
shots          10.000000
key_passes     7.000000
yellow_cards   2.000000
red_cards      0.000000
npg            0.500000
npxG           0.715585
xGChain        4.252738
xGBuildup     2.656397
dtype: float64
```

```
df_noid.std()
```

```
games          11.619836
time          1031.604819
goals          3.338851
xG            3.317946
assists        2.083350
xA             1.886510
shots          21.572664
key_passes     16.164361
yellow_cards   2.203661
red_cards      0.295800
npg            2.909929
npxG           2.931176
xGChain        5.600249
xGBuildup     3.376584
dtype: float64
```

```
df_noid.quantile(0.75) # 3rd quartile
```

```
games          30.000000
time          2319.000000
goals          2.000000
xG            2.053378
assists        2.000000
xA             2.050509
shots          23.750000
key_passes     19.000000
yellow_cards   3.000000
red_cards      0.000000
npg            2.000000
npxG           1.945799
xGChain        8.308002
xGBuildup     5.254647
Name: 0.75, dtype: float64
```

Menentukan pencilan (secara kasar) berdasarkan statistik

- **3-sigma rule:** Jika data kira-kira terdistribusi normal:
 - x_i adalah pencilan jika $x_i < \mu_s - 2\sigma_s$ atau $x_i > \mu_s + 2\sigma_s$
→ peluang bahwa data berjarak ke rerata lebih jauh dari 2 kali simpangan baku adalah 4.55%.
 - x_i adalah pencilan jika $x_i < \mu_s - 3\sigma_s$ atau $x_i > \mu_s + 3\sigma_s$
→ peluang bahwa data berjarak ke rerata lebih jauh dari 3 kali simpangan baku adalah 0.27%.
 - Kekurangan: (i) asumsi distribusi normal (belum tentu!), (ii) rerata dan simpangan baku dipengaruhi nilai pencilan itu sendiri, dan (iii) tidak dapat mendeteksi pencilan jika jumlah data sedikit (*small sample size*).
- **Tukey's fences:** memakai **rentang antarkuartil (interquartile range)** $IQR = Q_3 - Q_1$.
 - x_i adalah pencilan jika $x_i < Q_1 - 1.5(IQR)$ atau $x_i > Q_3 + 1.5(IQR)$.
 - x_i adalah pencilan ekstrim jika $x_i < Q_1 - 3(IQR)$ atau $x_i > Q_3 + 3(IQR)$.
- Metode-metode lain (mungkin lebih baik): Visualisasi, Grubb's test, Dixon's Q test, Algoritma Expectation Maximization, Jarak k-Nearest Neighbor, *local outlier factor* berbasis *density* (variasi *density-based clustering*), dll.

Mencari pencilan dengan Tukey's fences (1)

- Hitung IQR tiap kolom
- Variabel `q1` dan `q3` adalah Pandas Series berisi nilai-nilai kuartil pertama dan kuartil ketiga dari kolom-kolom numerik data.
- Variabel `iqr` adalah Pandas Series berisi nilai rentang antar kuartil untuk kolom-kolom numerik data.

```
q1 = df_noid.quantile(0.25)
q3 = df_noid.quantile(0.75)
iqr = q3 - q1
iqr
```

```
games           20.000000
time          1848.750000
goals           2.000000
xG            1.978711
assists        2.000000
xA             2.001264
shots          21.750000
key_passes     18.000000
yellow_cards   3.000000
red_cards      0.000000
npg            2.000000
npxG          1.871131
xGChain        7.116612
xGBuildup     4.534294
dtype: float64
```

Mencari pencilan dengan Tukey's fences

- Filter nilai-nilai di df_noid yang termasuk pencilan sesuai kriteria Tukey.
- Sebelum filter dihitung, harus dilakukan *join* dulu antara DataFrame df_noid dan Series iqr

```
df_noid_align, iqr_new = df_noid.align(iqr, axis=1, copy=False, join='outer')
outlier_filter = (df_noid1 < q1 - 1.5 * iqr_new) | (df_noid1 > q3 + 1.5 * iqr_new)
outlier_filter
```

	assists	games	goals	key_passes	npg	npxG	player_name	position	red_cards	shots	team_title	time	xA	xG	xGBuildup	xGChain	yellow_c
0	True	False	True	True	True	True		False	False	True	False	False	True	True	False	True	F
1	False	False	True	True	True	True		False	False	True	False	False	True	True	False	True	F
2	True	False	True	True	True	True		False	False	True	False	False	True	True	False	True	F
3	True	False	True	True	True	True		False	False	True	False	False	True	True	False	True	F
4	True	False	True	False	True	True		False	False	True	False	False	True	True	False	True	F
...	
517	False	False	False	False	False	False		False	False	False	False	False	False	False	False	False	F
518	False	False	False	False	False	False		False	False	False	False	False	False	False	False	False	F
519	False	False	False	False	False	False		False	False	False	False	False	False	False	False	False	F
520	False	False	False	False	False	False		False	False	False	False	False	False	False	False	False	F
521	False	False	False	False	False	False		False	False	False	False	False	False	False	False	False	F

522 rows × 17 columns

