Laporan Case-Based 2
Mata Kuliah Pembelajaran Mesin
K-Means

Wandi Yusuf Kurniawan – 1301218601 – IFX-45-GAB – IKN

Program Studi Sarjana Informatika

Fakultas Informatika

Universitas Telkom

Bandung

2022

PERNYATAAN KODE ETIK AKADEMIK UNIVERSITAS TELKOM

Dengan ini saya bersaksi bahwa hasil kerja ini **saya kerjakan sendiri** dan tidak mencontek hasil kerja dari mahasiswa lain dan tidak melakukan kecurangan dalam pengerjaan tugas ini. Jika kesaksian saya ini tidak benar, maka **saya bersedia menerima sanksi diberi minimum nilai E untuk mata kuliah ini dan/atau maksimum untuk semua mata kuliah pada semester ini.**

Wandi Yusuf Kurniawan

Pertama-tama, penulis melakukan import library yang dibutuhkan untuk mengerjakan tugas ini pada Jupyter Notebook menggunakan bahasa Python, diantaranya adalah:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import random
```

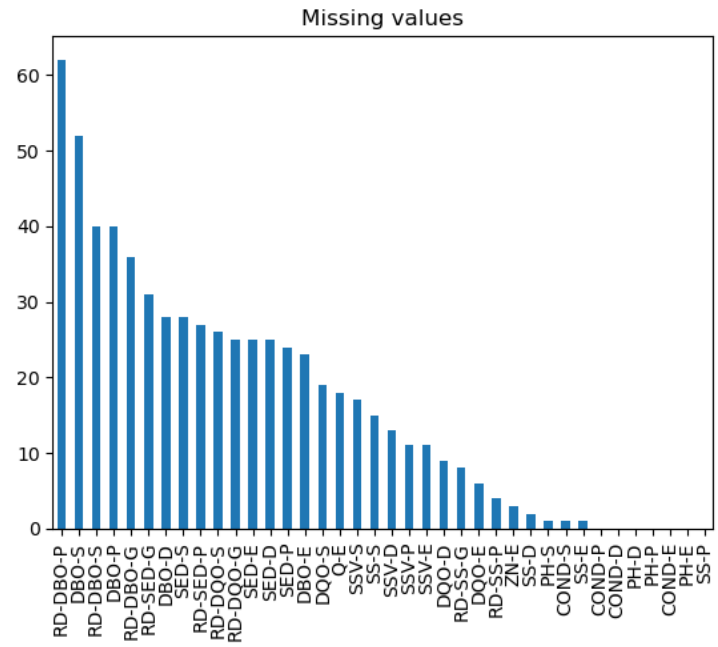
IKHTISAR KUMPULAN DATA YANG DIPILIH

Kemudian inisialisasi variable `data` untuk membaca file .data dan .names yang diambil dari https://archive.ics.uci.edu/ml/datasets/Water+Treatment+Plant dan lakukan konversi ke format .xlsx dan memanggil `.head()` untuk memastikan bahwa variable tersebut sudah berisi dataset yang nantinya akan dimasukkan ke dalam algoritma k-means yang dibuat penulis.

Date	Q-E	ZN-E	PH-E	DBO-E	DQO-E	SS-E	SSV-E	SED-E	COND-E	PH-P	...	COND-S	RD-DBO-P	RD-SS-P	RD-SED-P	RD-DBO-S	RD-DQO-S	RD-DBO-G	RD-DQO-G	RD-SS-G	RD-SED-G
1990-01-01	41230.0	0.35	7.6	120.0	344.0	136.0	54.4	4.5	993	7.5	...	903.0	NaN	62.8	93.3	NaN	62.5	86.7	71.8	87.5	99.4
1990-01-02	37386.0	1.40	7.9	165.0	470.0	170.0	76.5	4.0	1365	7.9	...	1481.0	NaN	50.0	94.4	85.9	73.6	86.7	79.4	89.4	100.0
1990-01-03	34535.0	1.00	7.8	232.0	518.0	220.0	65.5	5.5	1617	7.9	...	1492.0	32.6	62.4	95.0	81.3	59.9	87.5	71.8	85.9	99.8
1990-01-04	32527.0	3.00	7.8	187.0	460.0	180.0	67.8	5.2	1832	7.9	...	1590.0	13.2	57.6	95.5	85.3	70.4	85.0	77.2	83.3	100.0
1990-01-07	27760.0	1.20	7.6	199.0	466.0	186.0	74.2	4.5	1220	7.5	...	1411.0	38.2	46.6	95.0	84.9	61.1	89.4	73.8	86.6	99.6

Cek `.info()` pada data untuk mengetahui informasi kolom, berdasarkan table di bawah terdapat 527 baris dan 38 kolom, dan banyak baris yang bernilai null pada 31 kolom seperti pada barplot di bawah ini.

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 527 entries, 1990-01-01 to 1991-10-30
Data columns (total 38 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Q-E       509 non-null    float64
 1   ZN-E      524 non-null    float64
 2   PH-E      527 non-null    float64
 3   DBO-E     504 non-null    float64
 4   DQO-E     521 non-null    float64
 5   SS-E      526 non-null    float64
 6   SSV-E     516 non-null    float64
 7   SED-E     502 non-null    float64
 8   COND-E    527 non-null    int64
 9   PH-P      527 non-null    float64
 10  DBO-P     487 non-null    float64
 11  SS-P      527 non-null    int64
 12  SSV-P     516 non-null    float64
 13  SED-P     503 non-null    float64
 14  COND-P    527 non-null    int64
 15  PH-D      527 non-null    float64
 16  DBO-D     499 non-null    float64
 17  DQO-D     518 non-null    float64
 18  SS-D      525 non-null    float64
 19  SSV-D     514 non-null    float64
 20  SED-D     502 non-null    float64
 21  COND-D    527 non-null    int64
 22  PH-S      526 non-null    float64
 23  DBO-S     475 non-null    float64
 24  DQO-S     508 non-null    float64
 25  SS-S      512 non-null    float64
 26  SSV-S     510 non-null    float64
 27  SED-S     499 non-null    float64
 28  COND-S    526 non-null    float64
 29  RD-DBO-P  465 non-null    float64
 30  RD-SS-P   523 non-null    float64
 31  RD-SED-P  500 non-null    float64
 32  RD-DBO-S  487 non-null    float64
 33  RD-DQO-S  501 non-null    float64
 34  RD-DBO-G  491 non-null    float64
 35  RD-DQO-G  502 non-null    float64
 36  RD-SS-G   519 non-null    float64
 37  RD-SED-G  496 non-null    float64
dtypes: float64(34), int64(4)
memory usage: 160.6 KB
```



RINGKASAN PRA-PEMROSESAN DATA YANG DIUSULKAN

Penulis mengisi data null tersebut dengan metode ‘ffill’ dan ‘bfill’ karena dataset tersebut berupa time-series. Ffill atau forward fill berfungsi untuk mengisi nilai null dengan nilai di row sebelumnya, begitupun untuk bill atau backward fill yang berfungsi untuk mengisi nilai null dengan nilai di row setelahnya..

Date	Q-E	ZN-E	PH-E	DBO-E	DQO-E	SS-E	SSV-E	SED-E	COND-E	PH-P	...	COND-S	RD-DBO-P	RD-SS-P	RD-SED-P	RD-DBO-S	RD-DQO-S	RD-DBO-G	RD-DQO-G	RD-SS-G	RD-SED-G
1990-01-01	41230.0	0.35	7.6	120.0	344.0	136.0	54.4	4.5	993	7.5	...	903.0	32.6	62.8	93.3	85.9	62.5	86.7	71.8	87.5	99.4
1990-01-02	37386.0	1.40	7.9	165.0	470.0	170.0	76.5	4.0	1365	7.9	...	1481.0	32.6	50.0	94.4	85.9	73.6	86.7	79.4	89.4	100.0
1990-01-03	34535.0	1.00	7.8	232.0	518.0	220.0	65.5	5.5	1617	7.9	...	1492.0	32.6	62.4	95.0	81.3	59.9	87.5	71.8	85.9	99.8
1990-01-04	32527.0	3.00	7.8	187.0	460.0	180.0	67.8	5.2	1832	7.9	...	1590.0	13.2	57.6	95.5	85.3	70.4	85.0	77.2	83.3	100.0
1990-01-07	27760.0	1.20	7.6	199.0	466.0	186.0	74.2	4.5	1220	7.5	...	1411.0	38.2	46.6	95.0	84.9	61.1	89.4	73.8	86.6	99.6

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 527 entries, 1990-01-01 to 1991-10-30
Data columns (total 38 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Q-E       527 non-null    float64
 1   ZN-E      527 non-null    float64
 2   PH-E      527 non-null    float64
 3   DBO-E     527 non-null    float64
 4   DQO-E     527 non-null    float64
 5   SS-E      527 non-null    float64
 6   SSV-E     527 non-null    float64
 7   SED-E     527 non-null    float64
 8   COND-E    527 non-null    int64
 9   PH-P      527 non-null    float64
 10  DBO-P     527 non-null    float64
 11  SS-P      527 non-null    int64
 12  SSV-P     527 non-null    float64
 13  SED-P     527 non-null    float64
 14  COND-P    527 non-null    int64
 15  PH-D      527 non-null    float64
 16  DBO-D     527 non-null    float64
 17  DQO-D     527 non-null    float64
 18  SS-D      527 non-null    float64
 19  SSV-D     527 non-null    float64
 20  SED-D     527 non-null    float64
 21  COND-D    527 non-null    int64
 22  PH-S      527 non-null    float64
 23  DBO-S     527 non-null    float64
 24  DQO-S     527 non-null    float64
 25  SS-S      527 non-null    float64
 26  SSV-S     527 non-null    float64
 27  SED-S     527 non-null    float64
 28  COND-S    527 non-null    float64
 29  RD-DBO-P  527 non-null    float64
 30  RD-SS-P   527 non-null    float64
 31  RD-SED-P  527 non-null    float64
 32  RD-DBO-S  527 non-null    float64
 33  RD-DQO-S  527 non-null    float64
 34  RD-DBO-G  527 non-null    float64
 35  RD-DQO-G  527 non-null    float64
 36  RD-SS-G   527 non-null    float64
 37  RD-SED-G  527 non-null    float64
dtypes: float64(34), int64(4)
memory usage: 160.6 KB
```

Penulis melakukan pengecekan apakah setiap kolom memiliki korelasi yang kuat, Tetapi karena ada 38 kolom, maka sangat sulit untuk mengecek korelasi antar kolom, dan itu akan mempersulit visualisasi algoritma k-means.

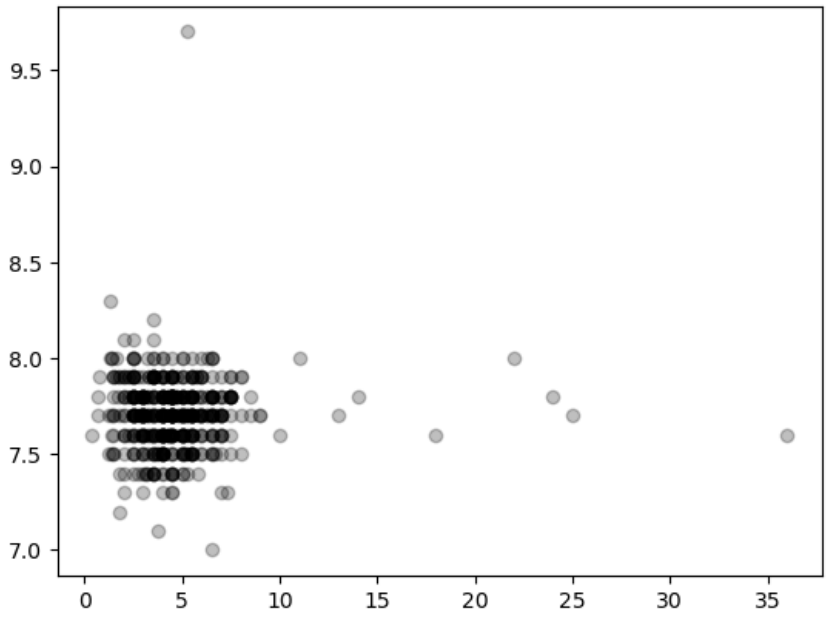
	Q-E	ZN-E	PH-E	DBO-E	DQO-E	SS-E	SSV-E	SED-E	COND-E	PH-P	DBO-P	SS-P	SSV-P	SED-P	COND-P	PH-D	DBO-D	DQO-D
Q-E	1.0	0.036	0.15	-0.18	-0.33	-0.0084	-0.32	-0.12	-0.083	0.16	-0.21	-0.024	-0.3	-0.13	-0.073	0.14	-0.17	-0.21
ZN-E	0.036	1.0	-0.011	0.0057	0.085	0.091	-0.098	0.08	0.053	0.00046	0.036	0.032	-0.068	0.068	0.051	0.031	0.11	0.082
PH-E	0.15	-0.011	1.0	0.22	0.18	-0.046	0.18	0.05	0.27	0.9	0.21	-0.031	0.19	0.057	0.28	0.82	0.29	0.26
DBO-E	-0.18	0.0057	0.22	1.0	0.49	0.12	0.23	0.24	0.2	0.22	0.67	0.12	0.18	0.22	0.22	0.22	0.57	0.42
DQO-E	-0.33	0.085	0.18	0.49	1.0	0.29	0.25	0.43	0.31	0.17	0.53	0.24	0.21	0.38	0.31	0.19	0.52	0.64
SS-E	-0.0084	0.091	-0.046	0.12	0.29	1.0	-0.54	0.58	0.083	-0.1	0.13	0.88	-0.5	0.56	0.08	-0.071	0.0076	-0.024
SSV-E	-0.32	-0.098	0.18	0.23	0.25	-0.54	1.0	-0.14	0.17	0.22	0.23	-0.5	0.87	-0.16	0.17	0.19	0.37	0.43
SED-E	-0.12	0.08	0.05	0.24	0.43	0.58	-0.14	1.0	0.19	0.045	0.26	0.58	-0.14	0.84	0.19	0.074	0.23	0.22
COND-E	-0.083	0.053	0.27	0.2	0.31	0.083	0.17	0.19	1.0	0.24	0.2	0.091	0.16	0.18	0.97	0.23	0.3	0.38
PH-P	0.16	0.00046	0.9	0.22	0.17	-0.1	0.22	0.045	0.24	1.0	0.19	-0.093	0.19	0.028	0.26	0.89	0.27	0.28
DBO-P	-0.21	0.036	0.21	0.67	0.53	0.13	0.23	0.26	0.2	0.19	1.0	0.22	0.18	0.34	0.21	0.21	0.64	0.49
SS-P	-0.024	0.032	-0.031	0.12	0.24	0.88	-0.5	0.58	0.091	-0.093	0.22	1.0	-0.55	0.73	0.09	-0.054	0.055	-0.0035
SSV-P	-0.3	-0.068	0.19	0.18	0.21	-0.5	0.87	-0.14	0.16	0.19	0.18	-0.55	1.0	-0.22	0.16	0.14	0.35	0.42
SED-P	-0.13	0.068	0.057	0.22	0.38	0.56	-0.16	0.84	0.18	0.028	0.34	0.73	-0.22	1.0	0.18	0.051	0.26	0.25
COND-P	-0.073	0.051	0.28	0.22	0.31	0.08	0.17	0.19	0.97	0.26	0.21	0.09	0.16	0.18	1.0	0.24	0.31	0.41
PH-D	0.14	0.031	0.82	0.22	0.19	-0.071	0.19	0.074	0.23	0.89	0.21	-0.054	0.14	0.051	0.24	1.0	0.24	0.25
DBO-D	-0.17	0.11	0.29	0.57	0.52	0.0076	0.37	0.23	0.3	0.27	0.64	0.055	0.35	0.26	0.31	0.24	1.0	0.74
DQO-D	-0.21	0.082	0.26	0.42	0.64	-0.024	0.43	0.22	0.38	0.28	0.49	-0.0035	0.42	0.25	0.41	0.25	0.74	1.0

Sebagai contoh, penulis memilih dua kolom dengan korelasi terkecil, dalam hal ini adalah `'SED-E'` dan `'PH-S'`, untuk divisualisasikan dengan dot hitam karena belum ada titik kluster yang mengelompokkan label.

```
corr_matrix = data.corr().abs().unstack().sort_values()
print(corr_matrix)
```

```
SED-E      PH-S        0.000056
PH-S       SED-E       0.000056
Q-E        RD-SED-G    0.000374
RD-SED-G   Q-E         0.000374
ZN-E       PH-P        0.000461
                         ...
PH-S       PH-S        1.000000
COND-D     COND-D      1.000000
SED-D      SED-D       1.000000
SS-D       SS-D        1.000000
RD-SED-G   RD-SED-G    1.000000
Length: 1444, dtype: float64
```

```
x = data['SED-E']
y = data['PH-S']
plt.scatter(x, y, c='black', alpha=0.25)
```



MENERAPKAN ALGORITMA YANG DI PILIH

Algoritma yang penulis rancang adalah k-means tanpa library sklearn karena dalam aturan pengerjaan tidak diperkenankan. Algoritma ini ditulis menggunakan referensi dari channel Emma Ding (https://www.youtube.com/watch?v=uLs-EYUpGAw) dengan sedikit modifikasi.

```
def k_means(data, k):
    centroids = initialize_centroids(data, k)

    while True:
        old_centroids = centroids
        labels = get_labels(data, centroids)
        centroids = update_centroids(data, labels, k)

        if should_stop(old_centroids, centroids):
            break

    return labels, centroids
```

Fungsi di atas membutuhkan dua parameter, yaitu data berisi array dua kolom dan jumlah cluster yang dilambangkan k. Fungsi ini mengembalikan label berupa array yang dihasilkan dari pasangan label, dan centroid berupa array yang berisi titik koordinat pada setiap centroid label.

```
def should_stop(old_centroids, centroids, threshold=1e-3):
    total_movement = 0
    for old_point, new_point in zip(old_centroids, centroids):
        total_movement += get_distance(old_point, new_point)
    return total_movement < threshold
```

Iterasi pada fungsi utama dibantu dengan fungsi should_stop yang akan terus berjalan sampai jumlah jarak pergerakan centroid lama ke centroid baru dibawah threshold, yaitu 0.001

```
def initialize_centroids(data, k):
    x_min = y_min = float('inf')
    x_max = y_max = float('-inf')
    for point in data:
        x_min = min(point[0], x_min)
        x_max = max(point[0], x_max)
        y_min = min(point[1], y_min)
        y_max = max(point[1], y_max)

    centroids = []
    for i in range(k):
        centroids.append([random_sample(x_min, x_max),
                          random_sample(y_min, y_max)])
    return centroids
```

```
def random_sample(low, high):
    return low + (high - low) * random.random()
```

Fungsi initialize_centroid menentukan posisi centroid secara random dengan rentang nilai minimum dan maksimum pada setiap kolomnya (fungsi random_sample).

```
def get_labels(data, centroids):
    labels = []
    for point in data:
        min_dist = float('inf')
        label = None
        for i, centroid in enumerate(centroids):
            new_dist = get_distance(point, centroid)
            if min_dist > new_dist:
                min_dist = new_dist
                label = i
        labels.append(label)
    return labels
def get_distance(point_1, point_2):
    return((point_1[0] - point_2[0]) ** 2 +
           (point_1[1] - point_2[1]) ** 2) ** 0.5
```

Fungsi get_label menghitung jarak semua titik (kolom) ke centoid dengan rumus euclidan seperti pada fungsi get_distance. Jika jarak titik ke centroid baru lebih dekat daripada jarak ke centroid lama, maka titik tersebut termasuk ke label baru.

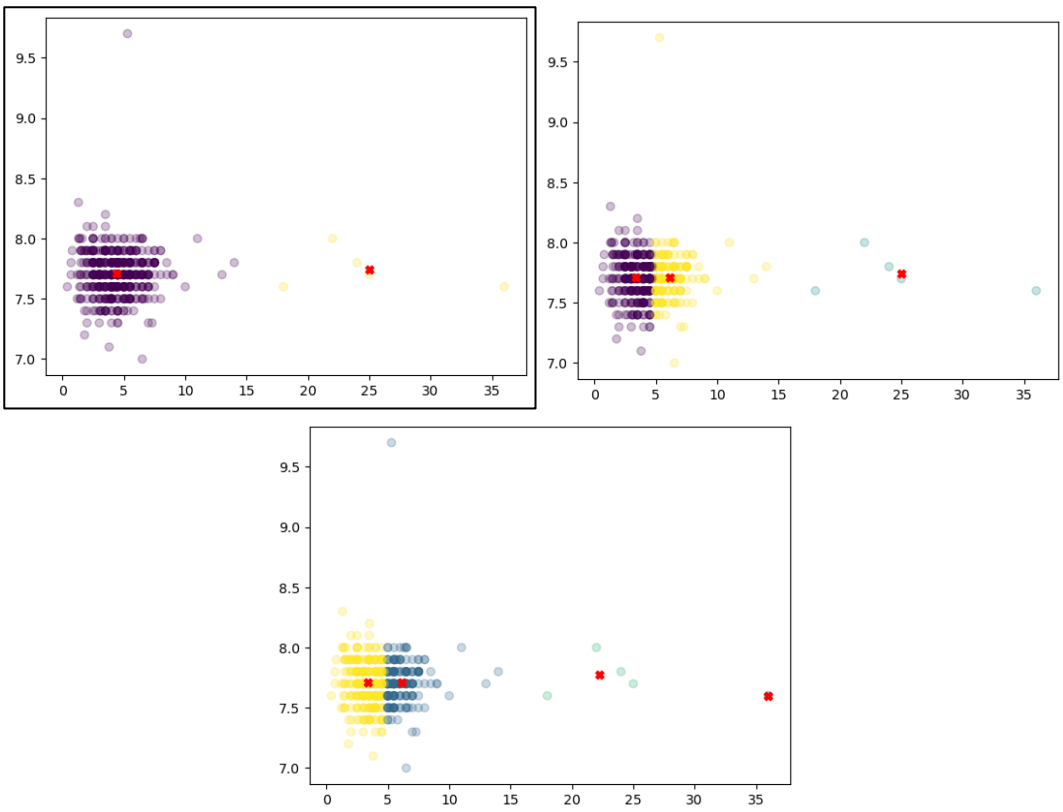
```
def update_centroids(points, labels, k):
    new_centroids = [[0, 0] for i in range(k)]
    counts = [0] * k

    for point, label in zip(points, labels):
        new_centroids[label][0] += point[0]
        new_centroids[label][1] += point[1]
        counts[label] += 1

    for i, (x, y) in enumerate(new_centroids):
        new_centroids[i] = (x / counts[i], y / counts[i])
    return new_centroids
```

Fungsi update_centroid menyesuaikan perpindahan centroid berdasarkan rata-rata kelompok titik yang termasuk pada suatu label.

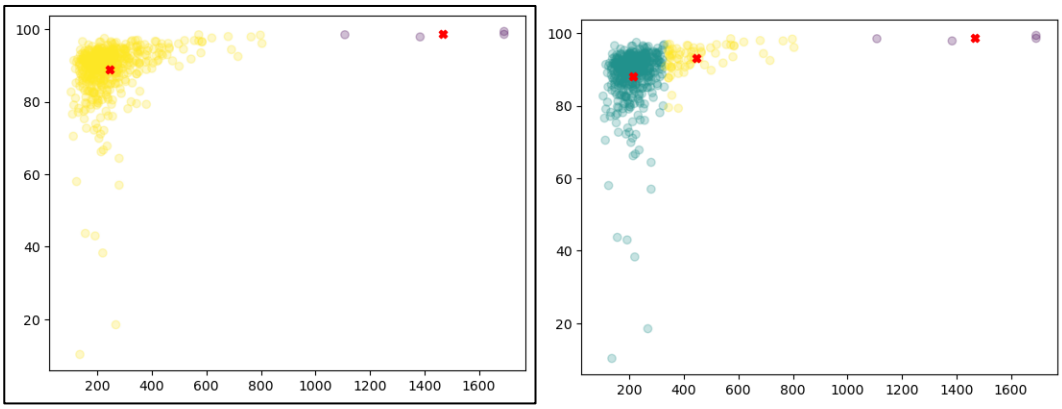
EVALUASI HASIL

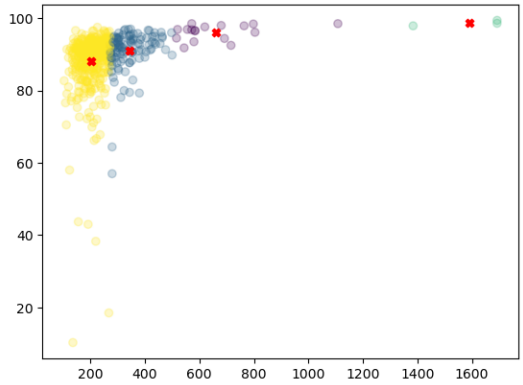
Penulis mencoba tiga pasang kolom dengan korelasi (yang sudah bernilai mutlak) terkecil, mendekati 0.25, dan terbesar mendekati 1 untuk mengecek berapa kluster yang ideal dalam pengelompokkan data.

Untuk pasangan kolom yang memiliki korelasi mendekati 0, yaitu kolom `'SED-E'` dan `'PH-S'`, scatter yang dihasilkan adalah sebagai berikut (untuk k = **2**, 3, 4. **Huruf tebal** menyatakan jumlah kluster terbaik*):

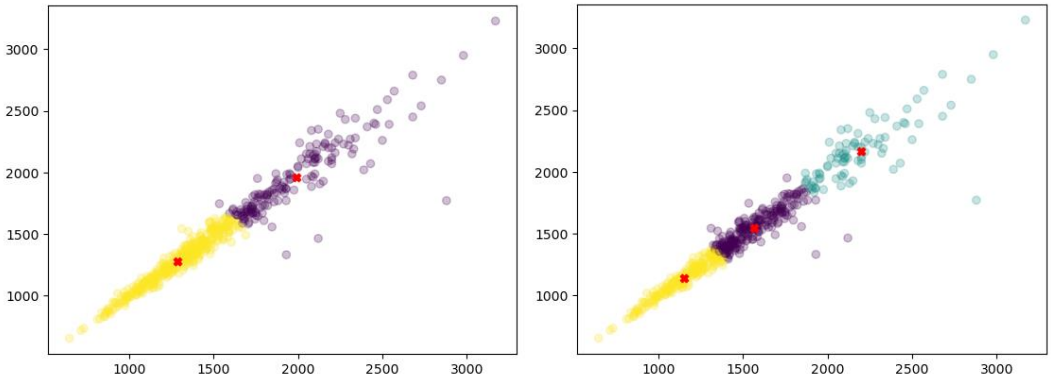


Untuk pasangan kolom yang memiliki korelasi mendekati 0.25, yaitu kolom `'SS-P'` dan `'RD-SS-G'`, scatter yang dihasilkan adalah sebagai berikut (untuk k = **2**, 3, 4. **Huruf tebal** menyatakan jumlah kluster terbaik*):





Untuk pasangan kolom yang memiliki korelasi mendekati 1, yaitu kolom `'COND-P'` dan `'COND-E'`, scatter yang dihasilkan adalah sebagai berikut (untuk k = 2, **3**. **Huruf tebal** menyatakan jumlah kluster terbaik*):



*Catatan: Penulis tidak melakukan elbow method karena hanya bisa dilakukan menggunakan library sklearn, yang mana berlawanan dengan aturan pengerjaan yang ditentukan, sehingga penulis hanya bisa mengira-ngira jumlah kluster terbaik berdasarkan distribusi titik. Selain itu, algoritma yang penulis rancang mempunyai kekurangan dalam penentuan jumlah kluster yang lebih banyak, seperti pada keterangan error di bawah yang menyatakan tidak ada label yang bernilai 4).

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Input In [70], in <cell line: 2>()
      1 points = np.transpose((x, y))
----> 2 labels, centroids = k_means(points, 5)
      4 labels = np.array(labels)
      5 centroids = np.array(centroids)

Input In [31], in k_means(data, k)
      5 old_centroids = centroids
      6 labels = get_labels(data, centroids)
----> 7 centroids = update_centroids(data, labels, k)
      9 if should_stop(old_centroids, centroids):
     10     break

Input In [31], in update_centroids(points, labels, k)
     56     counts[label] += 1
     58 for i, (x, y) in enumerate(new_centroids):
---> 59     new_centroids[i] = (x / counts[i], y / counts[i])
     60 return new_centroids

ZeroDivisionError: division by zero
```

Link Source Code, Laporan, dan Slide Presentasi : https://github.com/onedeetelyu/tugas_machine_learning
Link Video Presentasi : https://youtu.be/RbKI4LJXrFQ