Pembelajaran Mesin

Case Based 2 - Unsupervised Learning

Clustering

Disusun oleh:

Rayhan Suryatama Raharyawhedi

1301204435

Universitas Telkom

Jl. Telekomunikasi. 1, Terusan Buahbatu - Bojongsoang, Telkom University, Sukapura, Kec. Dayeuhkolot, Kabupaten Bandung, Jawa Barat 40257

Tugas ini dikerjakan secara mandiri. Semua referensi terkait pembangunan tugas ini terlampir pada bab Referensi. Penulis sudah berusaha semaksimal mungkin untuk membuat laporan ini orisinil. Segala bentuk plagiarisme pada laporan ini mungkin dikarenakan kesamaan dari referensi tercantum atau kesalahan yang tidak disengaja. Tugas ini juga dikerjakan dengan tidak melanggar ketentuan perkuliahan dan kode etik akademisi.

Daftar Isi

1. Rumusan Masalah

Permasalahan yang disinggung pada laporan tugas Case Based 2 ini adalah pengelompokkan/pelabelan data kepada sebuah dataset water-treatment.data (https://archive.ics.uci.edu/ml/datasets/Water+Treatment+Plant). Dataset tersebut yang belum memiliki label akan diolah sedemikian rupa agar memiliki label yang nantinya dapat diolah lebih lanjut. Proses pelabelan data untuk Case Based 2 ini akan menggunakan metode K-Means, dimana nanti data akan dikelompokkan berdasarkan banyak kelompok K yang telah ditentukan. Data dikelompokkan berdasarkan nilai tengah yang nantinya akan dibuat dan dikalkulasi.

2. Analisis Data

Dataset water-treatment akan dianalisis kualitasnya yang nantinya dari kualitas tersebut akan dibenahi pada saat pra-pemrosesan data. Pengecekan kualitas tersebut meliputi:

- Dimensi dataset
- Kolom yang tidak berguna/tidak dapat diproses
- Nilai kosong
- Rentang nilai

2.1. Dimensi Dataset

Dataset water-treatment memiliki dimensi yang cukup besar. Dengan adanya 39 kolom, maka dapat dibilang dataset ini berdimensi 39.

	ID	Q-E	ZN-E	PH-E	DBO-E	DQO-E	SS-E	SSV-E	SED-E	COND-E	...	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
0	D-1/3/90	44101.0	1.50	7.8	?	407	166.0	66.3	4.5	2110	...	2000.0	?	58.8	95.5	?	70.0	?	79.4	87.3	99.6
1	D-2/3/90	39024.0	3.00	7.7	?	443	214.0	69.2	6.5	2660	...	2590.0	?	60.7	94.8	?	80.8	?	79.5	92.1	100
2	D-4/3/90	32229.0	5.00	7.6	?	528	186.0	69.9	3.4	1666	...	1888.0	?	58.2	95.6	?	52.9	?	75.8	88.7	98.5
3	D-5/3/90	35023.0	3.50	7.9	205	588	192.0	65.6	4.5	2430	...	1840.0	33.1	64.2	95.3	87.3	72.3	90.2	82.3	89.6	100
4	D-6/3/90	36924.0	1.50	8.0	242	496	176.0	64.8	4.0	2110	...	2120.0	?	62.7	95.6	?	71.0	92.1	78.2	87.5	99.5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
522	D-26/8/91	32723.0	0.16	7.7	93	252	176.0	56.8	2.3	894	...	942.0	?	62.3	93.3	69.8	75.9	79.6	78.6	96.6	99.6
523	D-27/8/91	33535.0	0.32	7.8	192	346	172.0	68.6	4.0	988	...	950.0	?	58.3	97.8	83.0	59.1	91.1	74.6	90.7	100
524	D-28/8/91	32922.0	0.30	7.4	139	367	180.0	64.4	3.0	1060	...	1136.0	?	65.0	97.1	76.2	66.4	82.0	77.1	88.9	99
525	D-29/8/91	32190.0	0.30	7.3	200	545	258.0	65.1	4.0	1260	...	1326.0	39.8	65.9	97.1	81.7	70.9	89.5	87.0	89.5	99.8
526	D-30/8/91	30488.0	0.21	7.5	152	300	132.0	69.7	?	1073	...	1224.0	?	69.5	?	81.7	76.4	?	81.7	86.4	?

527 rows × 39 columns

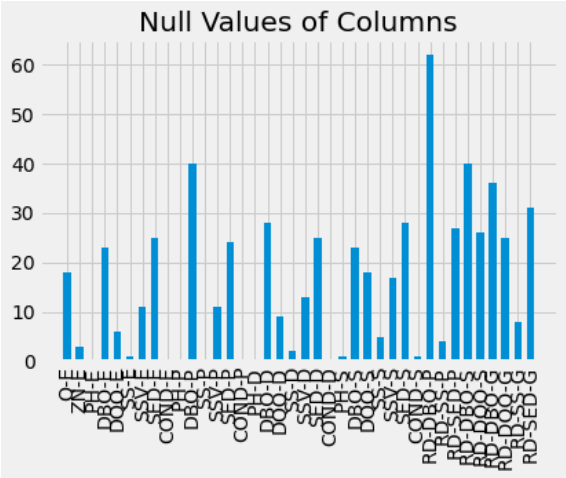
527 rows × 39 columns

2.2.Kolom Tidak Berguna/Tidak Dapat Diproses

	ID
0	D-1/3/90
1	D-2/3/90
2	D-4/3/90
3	D-5/3/90
4	D-6/3/90
...	...
522	D-26/8/91
523	D-27/8/91
524	D-28/8/91
525	D-29/8/91
526	D-30/8/91

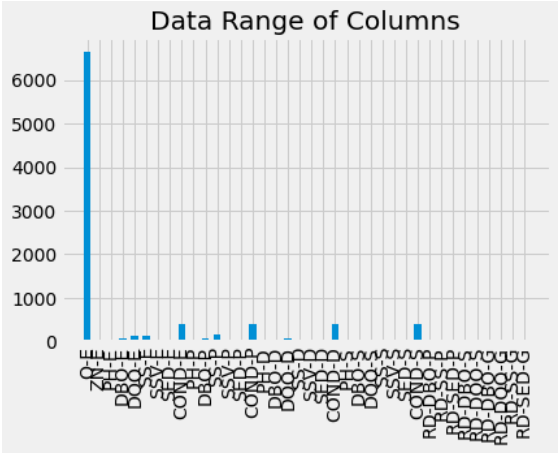
Dari analisis lebih lanjut, kolom ‘ID’ tidak dibutuhkan karena kolom tersebut, pertama, bernilai string/object, dan kedua, tidak berpengaruh signifikan ke data-data lain. ‘ID’ hanya dibutuhkan untuk pemberian identifikasi unik untuk setiap datanya. Index dari baris sudah cukup merepresentasikan hal tersebut. Maka kolom ‘ID’ tidak akan digunakan untuk proses Clustering ini.

2.3.Nilai Kosong



Nilai kosong di dataset water-treatment seperti yang ada di graf di atas. Hampir pada semua kolom yang ada, terdapat sebuah data kosong. Data kosong terbanyak berada di kolom ‘RD-DBO-P’. Nilai-nilai kosong ini nanti akan dihilangkan.

2.4.Rentang Nilai



Rentang nilai ini diambil dari perbandingan standar deviasi dari setiap kolom. Rentang nilai dari dataset water-treatment cukup kecil, namun ada satu kolom yang memiliki rentang nilai tertinggi, yaitu ‘O-E’ dikarenakan kolom tersebut adalah kolom utama pada dataset ini, yang berisikan data-data penting dari kolom lain. Perbandingan nilai yang jauh ini akan diselesaikan masalahnya dengan menggunakan metode scaling Min Max Scaling.

3. Pra-pemrosesan Data

Langkah setelah melakukan analisis data, hasil analisis tersebut akan dibenahi pada tahap Pra-Pemrosesan data ini. Data-data yang sekiranya tidak dibutuhkan atau masih kurang tepat jika digunakan akan diselesaikan masalahnya agar proses clustering nanti dapat berjalan dengan baik. Pra-pemrosesan data meliputi:

- Menghilangkan nilai kosong
- Scaling data
- Reduksi dimensi

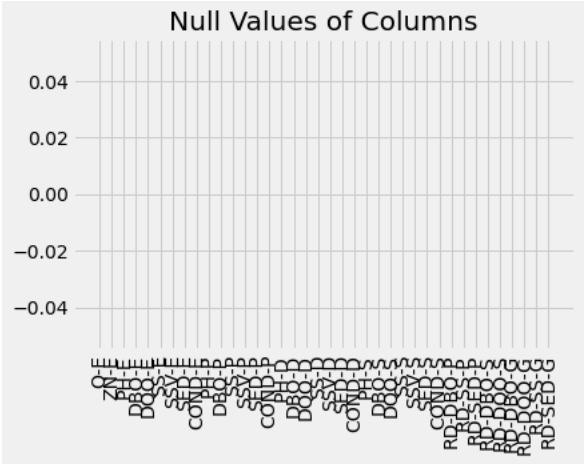
3.1.Menghilangkan Nilai Kosong

Nilai kosong yang telah dianalisis di bab sebelumnya akan ditangani dengan cara menghapus baris yang memiliki nilai kosong pada kolom tertentu.

Snippet Kode:

```
df_main = df_main.dropna(axis=0)
```

Hasil:



	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
3	35023.0	3.50	7.9	205.0	588.0	192.0	65.6	4.5	2430	7.8	...	1840.0	33.1	64.2	95.3	87.3	72.3	90.2	82.3	89.6	100.0
8	29156.0	2.50	7.7	206.0	451.0	194.0	69.1	4.5	1249	7.7	...	1338.0	46.1	43.6	92.5	85.6	58.2	92.2	73.8	90.2	99.4
9	39246.0	2.00	7.8	172.0	506.0	200.0	69.0	5.0	1865	7.8	...	1616.0	21.2	59.7	90.8	88.4	66.1	89.0	69.0	86.5	99.6
10	42393.0	0.70	7.9	189.0	478.0	230.0	67.0	5.5	1410	8.1	...	1575.0	0.6	45.8	92.0	11.6	25.7	19.6	36.0	43.0	36.4
14	40923.0	3.50	7.6	146.0	329.0	188.0	57.4	2.5	1300	7.6	...	1545.0	32.7	33.3	90.0	82.6	61.3	87.0	71.4	78.2	99.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
516	32363.0	0.10	7.6	159.0	310.0	146.0	68.5	1.6	1096	7.6	...	1083.0	25.2	61.4	91.2	78.6	65.1	86.8	81.0	89.0	99.4
517	31437.0	0.47	7.6	132.0	304.0	148.0	64.9	2.0	939	7.7	...	1012.0	45.6	60.3	94.4	82.5	72.9	89.4	86.2	91.2	99.5
519	28088.0	0.20	7.5	153.0	307.0	124.0	82.3	2.5	1044	7.6	...	1038.0	40.5	54.4	94.0	89.7	75.5	93.5	85.0	90.3	100.0
520	27838.0	0.13	7.6	179.0	265.0	128.0	71.9	1.8	992	7.6	...	1044.0	13.7	45.0	95.0	87.5	71.3	93.9	79.6	89.1	100.0
525	32190.0	0.30	7.3	200.0	545.0	258.0	65.1	4.0	1260	7.4	...	1326.0	39.8	65.9	97.1	81.7	70.9	89.5	87.0	89.5	99.8

380 rows × 38 columns

380 rows × 38 columns

Setelah dilakukan penghapusan baris yang memiliki nilai kosong, dapat dilihat sudah tidak ada nilai kosong di semua kolom, walaupun total jumlah baris dataset berkurang.

3.2.Scaling Data

Scaling dataset ini akan menggunakan cara Min Max Scaler yang memiliki rumus:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Pada proses scaling juga digunakan sebuah library dari scikit-learn yaitu dari kelas preprocessing-nya, MinMaxScaler().

Snippet Kode:

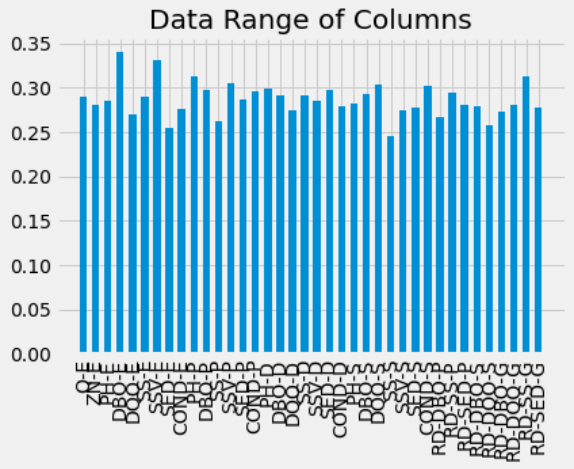
```
def dataScaling(df):
    scaler = MinMaxScaler()

    arr_scaled = scaler.fit_transform(df_main)

    return arr_scaled
```

```
data_scaled = dataScaling(df_main)
```

Hasil:



Hasil scaling data memperlihatkan bahwa data-data yang telah dilakukan proses scaling ini berkurang rentang data-nya. Sekarang data hanya berada pada rentang 0 hingga 1.

3.3. Reduksi Dimensi

Reduksi dimensi dilakukan agar data yang akan diproses nanti dapat divisualisasikan dengan benar. Karena dimensi data dari water-treatment yang memiliki 38 (39 sebelum dilakukan pembuangan kolom tidak berguna) maka perlu dilakukan reduksi dimensi menjadi 2 dimensi saja agar visualisasi dari data nanti akan lebih jelas.

Reduksi dimensi ini mengambil 2 kolom utama yang diambil dari kumpulan kolom-kolom lain.

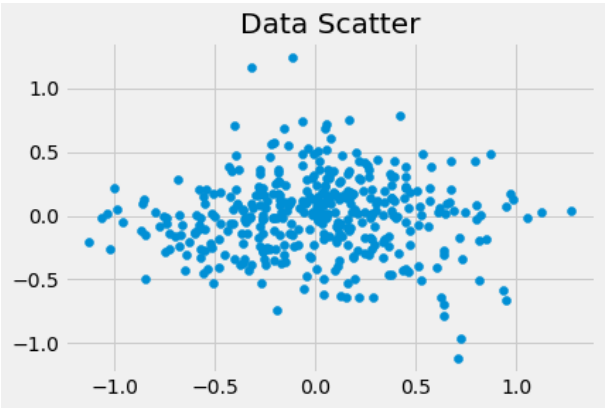
Proses reduksi dimensi menggunakan library scikit-learn, dari kelas decomposition, PCA().

Snippet Kode:

```
pca = PCA(n_components=2)

pc = pca.fit_transform(data_scaled)
```

Hasil:



4. Penerapan Algoritma Clustering

Algoritma Clustering yang akan digunakan pada tugas Case-Based 2 ini adalah K-Means. K-Means ini bekerja dengan cara menentukan berapa K cluster yang diinginkan, kemudian membuat sebuah centroid (data tengah) berjumlah K baru yang diambil dari titik random dari data, setelah itu mengelompokkan semua data dengan cluster centroid terdekat. Setelah pengelompokkan pertama selesai akan dilakukan penentuan centroid baru yang kemudian data-data akan dikelompokkan ulang, hingga nilai dari cluster tidak berubah lagi.

Pseudocode K-Means Clustering:

```
KMeansClustering
Algoritma
      init K centroids
      repeat
            pairing closest point to K cluster
            recalculate centroids
      until Centroids don't change
```

Snippet Kode Model K-Means:
*terlampir di halaman selanjutnya

```python
class KMeansClustering:
    def __init__(self, k, max_iter=100, random_state=42):
        self.k = k
        self.max_iter = max_iter
        self.random_state = random_state

    def initCentroids(self, data):
        np.random.RandomState(self.random_state)
        rand_i = np.random.permutation(data.shape[0])
        centroids = data[rand_i[:self.k]]

        return centroids

    def computeCentroids(self, data, labels):
        centroids = np.zeros((self.k, data.shape[1]))
        for i in range(self.k):
            centroids[i, :] = np.mean(data[labels == i, :], axis=0)

        return centroids

    def computeDistance(self, data, centroids):
        dist = np.zeros((data.shape[0], self.k))
        for i in range(self.k):
            row_norm = np.linalg.norm(data - centroids[i, :], axis=1)
            dist[:, i] = np.square(row_norm)

        return dist

    def closestToCluster(self, dist):
        return np.argmin(dist, axis=1)

    def computeSSE(self, data, labels, centroids):
        dist = np.zeros(data.shape[0])
        for i in range(self.k):
            dist[labels == i] = np.linalg.norm(data[labels == i] - centroids[i], axis=1)

        return np.sum(np.square(dist))

    def fit(self, data):
        self.centroids = self.initCentroids(data)
        for i in range(self.max_iter):
            temp_centroids = self.centroids
            dist = self.computeDistance(data, temp_centroids)
            self.labels = self.closestToCluster(dist)
            self.centroids = self.computeCentroids(data, self.labels)
            if np.all(temp_centroids == self.centroids):
                break

        self.error = self.computeSSE(data, self.labels, self.centroids)

        # def predict(self, data):
        #     dist = self.computeDistance(data, temp_centroids)

        #     return self.closestToClust(dist)
```

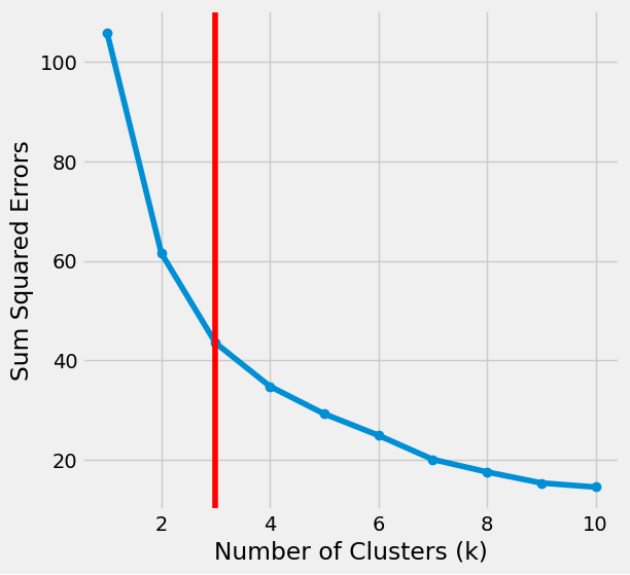
Kode mereferensi sebuah artikel dari @afrizalfir di medium.com (https://medium.com/@afrizalfir/kmeans-clustering-dan-implementasinya-5e967dc604cf)

Setelah perancangan model K-Means selesai, dilanjutkan dengan Elbow Method untuk mencari K cluster paling optimal.
Snippet Kode Elbow Method:

```
sse = []
for i in range(1, 11):
    kmeans = KMeansClustering(k=i, max_iter=100)
    kmeans.fit(pc)
    sse.append(kmeans.error)

plt.figure(figsize=(6,6))
plt.plot(range(1, 11), sse, '-o')
plt.xlabel(r'Number of Clusters (k)')
plt.ylabel('Sum Squared Errors')
plt.axvline(x=3, color='r')
plt.show
```

Hasil:



Dari analisis Elbow Method yang melihat Sum Square Error dari setiap K cluster, dapat dilihat bahwa K = 3 adalah jumlah cluster paling optimal karena K = 3 ini memiliki penurunan nilai Sum Square Error yang mulai landai daripada K = 4 dan seterusnya.

Dengan menggunakan K = 3 yang didapat dari Elbow Method, maka dapat dibuat sebuah cluster menggunakan K-Means dengan nilai K = 3.

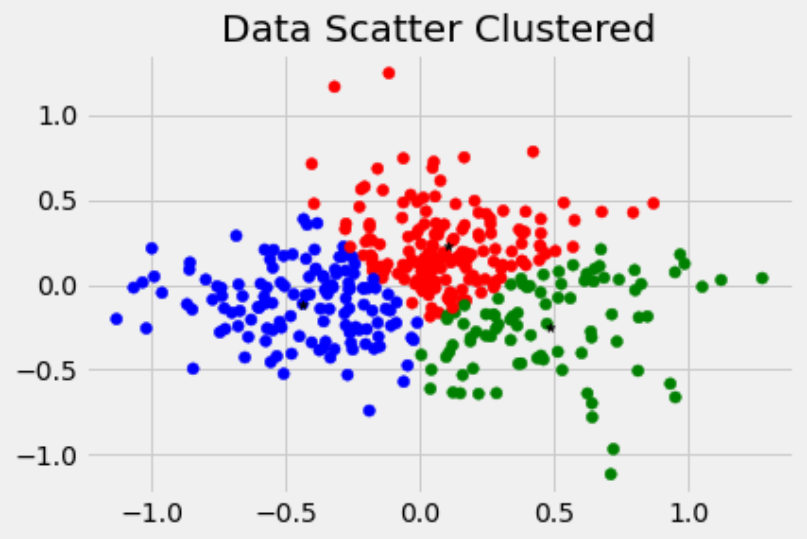
Snippet Kode Implementasi K-Means:

```
kmeans = KMeansClustering(k=3)
kmeans.fit(pc)

colors = {
    0:'r',
    1:'b',
    2:'g'
}
kmeans_labels = [colors[label] for label in kmeans.labels]

plt.scatter(pc[:, 0], pc[:, 1], c=kmeans_labels)
plt.scatter(kmeans.centroids[:, 0], kmeans.centroids[:, 1], marker='*', s=24, c='k')
plt.title('Data Scatter Clustered')
plt.show()
```

Hasil:



Setelah dilakukan proses implementasi model K-Means yang telah dibuat, dapat dilihat bahwa data-data yang dulunya tidak memiliki label kelompok sudah memiliki kelompok. Juga terlihat centroid mana yang menjadi patokan kelompok tersebut (centroid ditandai dengan simbol bintang (★) hitam pada graf).

5. Evaluasi

Proses K-Means ini mudah digunakan, namun kurang cocok untuk beberapa dataset yang tidak sesuai kriteria K-Means. Kriteria yang dimaksud seperti dimensi data, dimensi paling cocok untuk pengelompokkan K-Means agar visualisasi data-nya mudah dipahami adalah data 2 dimensi saja. Kriteria lain adalah bentuk data. Karena bentuk data dari water-treatment ini berkumpul di satu titik, K-Means cenderung cocok diterapkan, namun jika persebaran data-nya berbeda mungkin K-Means tidak akan cocok untuk data tersebut.

Cluster K dari K-Means juga tidak semena-mena ditentukan. Digunakan elbow method untuk menentukan nilai K tersebut memiliki alasan agar cluster yang dibangun adalah cluster paling optimal. Walaupun K-Means juga bisa membuat K tersendiri sesuai yang diinginkan pengolah data.

6. Presentasi Video

https://drive.google.com/file/d/14Q5xDFpa1wjX0fTm0lvAXwnm4PJVJ2NB/view?usp=share_link

7. Link Google Colab

https://colab.research.google.com/drive/10rU14PS-7JUWTbK0AWmtBDxtjvVenswu?usp=sharing

8. Referensi

https://www.bradleysawler.com/engineering/ml-clustering-of-a-waste-water-treatment-plant/

https://medium.com/@afrizalfir/kmeans-clustering-dan-implementasinya-5e967dc604cf

https://predictivehacks.com/k-means-elbow-method-code-for-python/

https://realpython.com/k-means-clustering-python/

https://lms.telkomuniversity.ac.id/mod/page/view.php?id=2589254