Journal of Applied Informatics and Computing (JAIC)
Vol.7, No.2, December 2023, pp. 231~239
e-ISSN: 2548-6861

K-Means Clustering with KNN and Mean Imputation on CPU Benchmark Compilation Data

Rofiq Muhammad Syauqi [1]*, Puspita Nurul Sabrina [2]*, Irma Santikarama [3]*
* Teknik Informatika, Universitas Jenderal Achmad Yani
rofiq.muhammad@student.unjani.ac.id [1], puspita.sabrina@lecture.unjani.ac.id [2], irma.santikarama@lecutre.unjani.ac.id [3]

Article Info

Article history:

Received 2023-09-05
Revised 2023-09-20
Accepted 2023-09-26

Keyword:

Clustering,
K-Means,
KNN Imputation,
Mean Imputation,
Silhouette coefficient.

ABSTRACT

In the rapidly evolving digital age, data is becoming a valuable source for decision-making and analysis. Clustering, as an important technique in data analysis, has a key role in organizing and understanding complex datasets. One of the effective clustering algorithms is k-means. However, this algorithm is prone to the problem of missing values, which can significantly affect the quality of the resulting clusters. To overcome this challenge, imputation methods are used, including mean imputation and K-Nearest Neighbor (KNN) imputation. This study aims to analyze the impact of imputation methods on CPU Benchmark Compilation clustering results. Evaluation of the clustering results using the silhouette coefficient showed that clustering with mean imputation achieved a score of 0.782, while with KNN imputation it achieved a score of 0.777. In addition, the cluster interpretation results show that the KNN method produces more information that is easier for users to understand. This research provides valuable insights into the effectiveness of imputation methods in improving the quality of data clustering results in assisting CPU selection decisions on CPU Benchmark Compilation data.

I. PENDAHULUAN

Dalam era digital saat ini, data telah menjadi sumber yang berharga dalam proses pengambilan keputusan dan analisis. Data dapat memberikan informasi penting bagi individu tertentu untuk mengoptimalkan kinerja dalam membuat keputusan yang lebih tepat. dalam menghadapi masalah tersebut, teknik analisis data memiliki peran yang sangat penting.

Salah satu teknik dalam analisis data adalah pengelompokkan (*Clustering).* Pengelompokkan tersebut dapat dibantu dengan algoritma *K-means clustering*, yang dimana adalah algoritma pengelompokkan pada metode *data mining* yang efektif untuk mengelompokkan suatu data. Algoritma *K-means clustering* berguna sebagai cara mengelompokkan data yang memiliki karakteristik sama ke dalam sebuah *cluster* atau kelompok yang sudah ditentukan sebelumnya[1]. Algoritma *K-means* merupakan salah satu algoritma *clustering* yang memiliki kinerja yang relatif cepat, mudah dan umum digunakan[2], namun rentan terhadap *noice*[3]. Tidak hanya *noice*, algoritma ini rentan terhadap *missing value* yang memungkinkan hasil kualitas *cluster* menjadi rendah[4].

Dalam upaya mengatasi problematika *missing value*, teknik imputasi adalah salah satu teknik yang dapat digunakan. Metode imputasi adalah sebuah metode untuk mengisi *missing value* dengan menggunakan metode estimasi atau prediksi [4]. Imputasi mengisi nilai yang hilang tersebut berdasarkan informasi yang tersedia pada data. Menurut penelitian terdahulu, teknik imputasi adalah teknik terbaik yang bisa digunakan untuk mengatasi problematika *missing value*[5]. Diantara berbagai metode imputasi yang ada, metode *mean imputation* dan *K-nearest neighbor imputation* adalah yang paling umum digunakan. Diantara kedua metode tersebut, imputasi *mean* adalah metode yang mudah digunakan, tetapi metode tersebut bisa menghasilkan error pada estimasi dan menghasilkan bias pada data[6]. Pada penelitian terdahulu menunjukkan bahwa metode imputasi *K-nearest neighbor* lebih unggul daripada metode imputasi *mean* [7].

Data yang terdapat pada kasus dunia nyata, tidak luput dari permasalahan *missing value*, yang dapat mempengaruhi

kualitas hasil pengelompokkan. Pada penelitian terdahulu, Sebelum dilakukan pengelompokkan data *CPU* berdasarkan spesifikasinya menggunakan metode *K-Means clustering*, perlu dilakukan data *preprocessing* untuk mengatasi permasalahan data yang hilang tersebut. Penelitian tersebut telah menginvestigasi hasil pengelompokkan data *CPU* berdasarkan atribut *lithography*, *cores*, *threads*, *processor base frequency*, *cache*, *cache type*, *TDP*, *max memory size*, dan *graphics base frequency* yang menghasilkan 2 *cluster CPU* spesifikasi rendah dan *CPU* spesifikasi tinggi. Namun, dalam rangka memperdalam pemahaman tentang pengelompokan data *CPU*, penelitian ini memfokuskan pada data *CPU benchmark Compilation*[8]. Data ini memiliki ciri khusus yang mewakili performa *CPU* dalam skenario tertentu. Data tersebut juga tidak luput dari missing value. Oleh karena itu, perbandingan hasil pengelompokan dengan metode imputasi tertentu menjadi penting untuk memahami dampaknya terhadap kualitas pengelompokan data *CPU Benchmark Compilation*.

Penelitian ini difokuskan untuk menganalisis pengaruh dari metode imputasi tertentu terhadap hasil pengelompokkan data menggunakan algoritma *K-means clustering* pada data *CPU Benchmark Compilation*. Dengan memahami perbedaan dalam kualitas hasil pengelompokkan, penelitian ini bertujuan memberikan interpretasi yang lebih tepat ketika data yang tersedia untuk pengelompokkan mengandung *missing value*.

II. METODE PENELITIAN

A. Pengumpulan Dataset

Dataset yang digunakan pada penelitian ini, dikumpulkan dari website bernama Kaggle.com yaitu website yang menyediakan data set untuk keperluan data scientist dan machine learning. Data yang digunakan bernama data set *CPU Benchmark Compilation* yang berisikan kumpulan hasil test benchmark suatu *CPU*.

B. Data Selection

Data selection adalah proses pemilihan data yang digunakan dalam proses analisis penelitian ini. Di tahap ini, data yang tidak relevan atau tidak digunakan dikeluarkan dalam analisis ini. Sedangkan untuk data yang relevan dipilih untuk digunakan pada proses selanjutnya.

C. Data Pre-processing Menggunakan Metode Imputation

Data *preprocessing* adalah tahapan yang harus dilalui sebelum data mining dilakukan. Data yang digunakan bisa saja terdapat beberapa *noise*, *error*, *exception*, data yang tidak pasti, atau bahkan tidak lengkapnya data (*Missing value*). *Error* dan *noise* biasanya bisa membuat kebingungan pada data mining process yang juga berujung ke error pada pencarian pattern yang dilakukan [4].

Imputation adalah salah satu cara untuk mengatasi *missing value* atau value yang hilang dari data. Menurut penelitian berjudul *"K-Nearest Neighbor (K-NN) based Missing Data Imputation"* imputasi adalah mengganti data yang hilang dengan memperkirakan nilai yang benar untuk mendapatkan data lengkap sehingga prosedur analisis data yang membutuhkan data lengkap dapat dilakukan[6]. Terdapat beberapa jenis metode imputasi yang dapat digunakan, diantaranya seperti berikut:

1) Mean Imputation

Mean imputation adalah teknik mengisi data yang hilang dengan menggunakan rata-rata dari data yang tidak memiliki value yang hilang [9] atau mengganti nilai yang hilang dengan nilai rata-rata dari suatu variable[10]. Meskipun metode ini tergolong mudah untuk mengatasi *missing value*, tetapi terkadang metode ini menghasilkan nilai yang tidak sesuai dengan data yang ada dan dapat menyebabkan bias pada perkiraan[11]. Rumus dari *mean imputation* tersebut dijabarkan sebagai berikut:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \quad (1)$$

x̄	=	Rata-rata data
xi	=	Data ke-i
n	=	Banyaknya data
i	=	Data ke-1 sampai dengan n

2) KNN Imputation

Teknik imputasi K-NN adalah teknik mengisi value yang hilang dengan menggunakan teknik *machine learning*. Metode ini memiliki keunggulan diantaranya dapat digunakan pada data bertipe diskrit dan kontinyu [6][12], *qualitative attribute* dan *quantitative attribute*[13]. Sederhananya, metode KNNI, menghitung jarak antara data tuple yang hilang dengan tuple yang memiliki data lengkap di setiap variabel yang dimilikinya dengan menggunakan perhitungan jarak seperti rumus Euclidean. Hasil dari perhitungan jarak dijadikan *"nearest neighbors"* atau tetangga terdekat, yang digunakan untuk pengisian value yang hilang pada atribut tuple tertentu[14]. Langkah-langkah algoritma KNN imputation dapat dijabarkan sebagai berikut[15]:

a. Tentukan parameter K, dimana parameter tersebut digunakan untuk memilih jumlah tetangga terdekat yang akan digunakan. Jika parameter K diisi dengan angka 5, algoritma KNN akan menentukan 5 tetangga terdekat berdasarkan nilai jarak terkecil antara 2 observasi
b. Menghitung jarak antara tuple yang memiliki *missing value* dan tuple yang memiliki data lengkap dengan menggunakan rumus *Euclidean*.

$$d_{(x,y)} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad (2)$$

$d_{(x,y)}$ = Merepresentasikan jarak Euclidean x,y

n = Dimensi data

i = Data variabel i = 1,2,3 … n

x_i = Nilai variabel x yang memiliki *missing value* ke-i

y_i = Nilai variabel yang memiliki data lengkap ke-i

c. Urutkan hasil perhitungan jarak yang memiliki nilai terbesar sampai nilai terkecil.

d. Pilih record k yang memiliki jarak *nearest neighbors* terkecil.

e. Melakukan imputasi missing data dengan menghitung nilai *weight mean estimation* pada K observasi terdekat yang tidak mengandung *missing value* dengan rumus berikut:

$$x_j = \frac{\sum_{k=1}^{K} w_k v_k}{\sum_{k=1}^{K} w_k} \quad (3)$$

x_j = Weight *Mean* Estimation

K = Jumlah parameter k yang digunakan dengan k=1,2,3,..k

w_k = Bobot observasi nearest neighbor ke-K dengan rumus berikut :

$$w_k = \frac{1}{d(x,y)^2} \quad (4)$$

Dimana $d(x,y)^2$ adalah jarak perhitungan *Euclidean distance* pada Langkah b yang dipangkat oleh angka 2.

v_k = Nilai pada data lengkap di variabel yang mengandung missing data berdasarkan observasi k.

D. Data Transformation

Tahapan selanjutnya yang akan dilakukan pada preprocessing data adalah tahapan data transformation. Proses data transformation akan dilakukan pada dataset untuk mengubah format data yang tersedia menjadi format yang lebih sesuai untuk analisis[16]. Dalam penggunaan data transformation, strategi yang bisa dilakukan adalah normalisasi. Normalisasi data adalah salah satu metode yang digunakan dalam data transformation. Teknik normalisasi data sendiri digunakan untuk menskalakan data numerik ke dalam bentuk rentang tertentu [17]. Teknik normalisasi yang digunakan pada penelitian ini adalah *Min-Max Normalization*. Rumus yang digunakan pada metode Min-Max tersebut adalah sebagai berikut:

$$x_{normalized} = \frac{(x - \min(x))}{(\max(x) - \min(x))} \quad (5)$$

x = Data original
x_normalized = Data yang telah di normalisasi
min(x) = Value minimum x
max(x) = Value maximum x

E. Clustering

Clustering adalah teknik untuk mengelompokkan sejumlah objek data yang memiliki kemiripan karakteristik yang sama dengan objek data yang lain dan jika semakin tinggi kemiripan data tersebut maka akan tergabung ke dalam satu kelompok (*cluster*) yang sama [18]. Teknik *Cluster*ing sendiri dibagi menjadi 2 metode yaitu *Hierarchical Clustering* dan *non-Hierarchical Clustering*. Metode yang paling sering digunakan pada data mining *cluster*ing sendiri adalah metode non-Hierarchical *Cluster*ing yang paling sering disebut dengan *partitioning*[19]. Metode partitioning dapat terdiri dari *K-means* dan *fuzzy K-means* [20]. Pada penelitian ini menggunakan metode *cluster partitioning* dibantu dengan algoritma *K-means*.

K-means Clustering merupakan metode pengelompokkan data yang bersifat *non-Hierarchical* yaitu mempartisi data ke dalam 2 kelompok atau bahkan lebih dari 2 kelompok. Metode ini mempartisi data yang memiliki karakteristik sama lalu data tersebut dimasukkan ke dalam *cluster* yang sama. Sedangkan untuk data yang memiliki karakteristik yang berbeda akan dimasukkan ke *cluster* yang berbeda [21]. Sederhananya algoritma ini mempunyai suatu prinsip, dimana pada awal penggunaannya memilih sebuah representative point K sebagai *initial centroids*. Setiap point di *assign* ke *centroid* terdekat berdasarkan pemilihan pengukuran proximity tertentu. Pada saat *cluster* terbentuk, centroid untuk setiap *cluster* di update. Kemudian algoritma ini mengulangi step tersebut sampai centroids tidak berubah. Tahapan algoritma pengelompokkan data menggunakan *K-means* adalah sebagai berikut:

a. Menginisialisasi K secara random untuk dijadikan centroid. Yang dimana K tersebut adalah jumlah *cluster* yang dimunculkan. Jumlah K tersebut ditentukan oleh user. centroid ini merepresentasikan titik tengah untuk setiap *cluster*.

b. Langkah selanjutnya adalah menempatkan setiap point ke *cluster* yang dimana memiliki *centroid* terdekat dengan point tersebut berdasarkan pengukuran jarak seperti *Euclidean Distance*.

$$d(i,k) = \sqrt{\sum_{j=1}^{m} \left(x_{ij} - c_{kj}\right)^2} \quad (6)$$

d(i,k) = Jarak antara data ke-i dengan titik pusat *cluster* (centroid) ke-k

m = Jumlah atribut

x_i = Data ke-i

c_k = Data pusat kluster ke-k

F. Penentuan Jumlah Cluster Menggunakan Elbow Method

Menentukan jumlah *cluster* yang terbaik dan tepat untuk digunakan dalam analisis klaster, dapat ditentukan dengan menggunakan metode Elbow. Metode Elbow menghasilkan grafik yang bersumber dari jumlah total K dan memberikan lekukan pada grafik seperti lekukan "siku"[22].

G. Evaluasi Hasil Cluster

Evaluasi hasil *cluster* adalah metode untuk mengukur keefektifan dan kualitas antar *cluster* yang dibentuk. Cara mengukur kualitas *cluster* dapat dilakukan dengan menggunakan metode *silhouette coefficient*. Metode *silhouette coefficient* merupakan gabungan antara 2 metode yaitu *cohession* untuk mengukur seberapa dekat relasi antar objek dalam sebuah *cluster*, dan separation untuk mengukur seberapa jauh *cluster* terpisah dengan *cluster* lain. Jika hasil *silhouette coefficient* mendekati 1, maka semakin baik hasil *cluster* dari pengelompokkan data tersebut[23].

III. HASIL DAN PEMBAHASAN

A. Pengumpulan Dataset

Dataset *CPU Benchmark Compilation* yang telah dikumpulkan memiliki total data berjumlah 3826 data. data set tersebut juga meliputi beberapa atribut. Atribut untuk setiap record *CPU* dimuat dalam tabel berikut:

TABEL I
DATA CPU BENCHMARK COMPILATION

No.	Variabel	Tipe Data	Deskripsi
1.	CPUName	Text	Variabel yang menjelaskan tentang nama dari sebuah *CPU*
2.	Price	Numerik	Variabel yang menjelaskan tentang harga sebuah *CPU*
3.	CPUMark	Numerik	Variabel yang menjelaskan tentang skor single thread *benchmark* sebuah *CPU*
4.	CPUValue	Numerik	Variabel yang menjelaskan tentang price to performance sebuah *CPU* dengan menggunakan single thread
5.	threadMark	Numerik	Variabel yang menjelaskan tentang sko *benchmark CPU* multi thread
6.	threadValue	Numerik	Variabel yang menjelaskan tentang price to performance sebuah *CPU* dengan menggunakan multi thread
7.	TDP	Numerik	Variabel yang menjelaskan tentang thermal design power dari sebuah *CPU*
8.	powerPerf	Numerik	Variabel yang menjelaskan tentang daya yang dihasilkan sebuah *CPU* pada saat dijalankan dengan performa maksimal
9.	Cores	Numerik	Variabel yang menjelaskan tentang jumlah *CPU* core pada sebuah *CPU*
10.	testDate	Date	Variabel yang menjelaskan tentang kapan *CPU* tersebut diuji
11.	socket	Text	Variabel yang menjelaskan tentang konektor yang digunakan sebuah *CPU*.
12.	category	Text	Variabel yang menjelaskan tentang kategori kegunaan sebuah *CPU*

B. Data Selection

Data selection adalah proses pemilihan data yang digunakan dalam proses analisis penelitian ini. Di tahap ini, data yang tidak relevan atau tidak digunakan dikeluarkan dalam analisis ini. Sedangkan untuk data yang relevan dipilih untuk digunakan pada proses selanjutnya. Populasi yang ditentukan adalah populasi yang mempunyai relevansi dengan analisis, yang memiliki karakteristik diantaranya, memiliki atribut price, CPUMark, CPUValue, threadMark, threadValue, TDP, powerPerf, cores.

C. Data Pre-processing Menggunakan Metode Imputation

Pada tahap data pre-processing, metode yang digunakan adalah metode imputasi KNN dan metode imputasi *mean* untuk mengatasi *missing value* pada data. Dalam upaya untuk mengatasi *missing value* dalam data dan meningkatkan kelengkapan serta akurasi data yang dihasilkan, pada bab berikut dijelaskan penggunaan metode imputasi K-Nearest Neighbors (KNN) dan *Mean*. Pada tahap ini terdapat beberapa langkah seperti:

1) Identifikasi Missing value pada Atribut

Data yang telah dikumpulkan terkandung *missing value* pada beberapa atribut. Jumlah *missing value* tersebut dapat disajikan dalam tabel berikut:

TABEL II
JUMLAH MISSING VALUE PADA ATRIBUT

No	Atribut	Jumlah Missing value
1	Price	1858
2	CPUValue	1858
3	threadValue	1858
4	TDP	685
5	powerPerf	685

2) Mean Imputation

Contoh pengerjaan *mean imputation* dibantu menggunakan 10 sampel yang berisikan 1 data kosong pada observasi 1 dan data yang tidak memiliki *missing value* pada setiap atribut nya. Berikut adalah gambar tabel data sampel untuk imputation:

observasi	price	cpuMark	cpuValue	threadMark	threadValue	TDP	powerPerf	cores
1		10882		3330		280	388,65	64
2	7299,99	88338	12,1	2635	0,36	280	315,49	64
6	7060	85861	12,16	2727	0,39	225	381,6	64
7	6807,98	83971	12,33	2626	0,39	280	299,9	64
8	8399,69	81568	9,71	2569	0,31	280	291,31	64
11	5424,99	76455	14,09	2695	0,5	225	339,8	48
12	4000	71646	17,91	2097	0,52	200	358,23	64
13	6817	71576	10,5	2054	0,3	225	318,12	64
14	5045,99	68749	13,62	2145	0,43	225	305,55	64
17	3684	67748	18,39	2155	0,59	225	301,1	48

Gambar 1 Tabel Data Sampel untuk Imputasi

Langkah-langkah perhitungan *Mean* imputation dirangkum pada poin berikut:

a. Identifikasi apakah variabel suatu data memiliki value yang hilang, seperti contoh pada sampel gambar tabel 1 memiliki *missing value* pada kolom price, cpuValue, dan threadValue.
b. Hitunglah nilai *mean* pada variabel price, CPUValue, dan threadValue lalu isi *missing value* tersebut dengan persamaan (1) pada masing-masing variabel.
   Hasil *Mean* Variabel price:

$$\bar{x} = \frac{7299{,}99+7060+6807{,}98+8399{,}69+5424{,}99+4000+6817+5045{,}99+3684}{9} = 6059{,}96$$

   Hasil *Mean* Variabel cpuValue:

$$\bar{x} = \frac{12{,}1+12{,}16+12{,}33+9{,}71+14{,}09+17{,}91+10{,}5+13{,}62+18{,}39}{9} = 13{,}42333333$$

   Hasil *Mean* Variabel threadValue:

$$\bar{x} = \frac{0{,}36+0{,}39+0{,}39+0{,}31+0{,}5+0{,}52+0{,}3+0{,}43+0{,}59}{9} = 0{,}421111111$$

c. Setelah rata-rata untuk tiap kolom yang memiliki *missing value* dihitung, isikan semua *missing value* dengan hasil *mean* yang telah dihitung sesuai kolom tertentu. Berikut adalah gambar tabel yang dimana *missing value* telah terisi dengan nilai *mean*:

observasi	price	cpuMark	cpuValue	threadMark	threadValue	TDP	powerPerf	cores
1	6059,96	10882	13,42333333	3330	0,421111111	280	388,65	64
2	7299,99	88338	12,1	2635	0,36	280	315,49	64
6	7060	85861	12,16	2727	0,39	225	381,6	64
7	6807,98	83971	12,33	2626	0,39	280	299,9	64
8	8399,69	81568	9,71	2569	0,31	280	291,31	64
11	5424,99	76455	14,09	2695	0,5	225	339,8	48
12	4000	71646	17,91	2097	0,52	200	358,23	64
13	6817	71576	10,5	2054	0,3	225	318,12	64
14	5045,99	68749	13,62	2145	0,43	225	305,55	64
17	3684	67748	18,39	2155	0,59	225	301,1	48

Gambar 2 Tabel Hasil Pengisian Missing Value Metode Imputasi Mean

Hasil dari pengisian *missing value* menggunakan *mean* akan tampak berbeda jika jumlah data yang digunakan adalah jumlah data asli. Berikut adalah gambar tabel hasil pengisian *missing value* menggunakan *mean* pada data asli.

observasi	price	cpuMark	cpuValue	hreadMark	hreadValu	TDP	powerPerf	cores
1		108822		3330		280	388,65	64
2	7299,99	88338	12,1	2635	0,36	280	315,49	64
3		86006		2387				64
4	7060	85861	12,16	2727	0,39	225	381,6	64
5	6807,98	83971	12,33	2626	0,39	280	299,9	64
6	8399,69	81568	9,71	2569	0,31	280	291,31	64
7		80842		3340		280	288,72	32
8		77460		2564				60
9	5424,99	76455	14,09	2695	0,5	225	339,8	48
10	4000	71646	17,91	2097	0,52	200	358,23	64

Gambar 3 Tabel Data Asli

observasi	price	cpuMark	cpuValue	hreadMark	hreadValu	TDP	powerPerf	cores
1	441,5009	108822	35,35491	3330	15,18901	280	388,65	64
2	7299,99	88338	12,1	2635	0,36	280	315,49	64
3	441,5009	86006	35,35491	2387	15,18901	62,37556	121,6232	64
4	7060	85861	12,16	2727	0,39	225	381,6	64
5	6807,98	83971	12,33	2626	0,39	280	299,9	64
6	8399,69	81568	9,71	2569	0,31	280	291,31	64
7	441,5009	80842	35,35491	3340	15,18901	280	288,72	32
8	441,5009	77460	35,35491	2564	15,18901	62,37556	121,6232	60
9	5424,99	76455	14,09	2695	0,5	225	339,8	48
10	4000	71646	17,91	2097	0,52	200	358,23	64

Gambar 4 Tabel Hasil Pengisian Missing Value Metode Imputasi Mean pada Data Asli

Metode Imputasi *Mean* adalah pendekatan yang sangat efisien dalam mengisi nilai yang hilang dalam data. Dalam metode ini, nilai yang hilang digantikan dengan rata-rata dari seluruh data dalam atribut tersebut. Hal ini memungkinkan untuk waktu eksekusi yang sangat cepat yaitu 0,01 detik.

3) KNN Imputation

Contoh pengerjaan *K-Nearest Neighbor Imputation* dibantu dengan data sampel pada gambar tabel 1. Langkah-langkah pengerjaan algoritma KNNI dirangkum dalam poin berikut:

a. Menginisialisasi k untuk digunakan sebagai jumlah tetangga terdekat yang akan digunakan. Misalkan k = 5.
b. Menghitung jarak tuple *missing value* dengan tuple yang memiliki data lengkap dengan menggunakan rumus *Euclidean Distance*. Misalkan mengisi value yang hilang pada observasi 1 variabel price. Hitung jarak observasi yang memiliki *missing value* dengan observasi yang memiliki data lengkap dengan menggunakan persamaan (2) untuk mendapatkan nearest neighbor.

TABEL III
PERHITUNGAN JARAK KNN

Observasi	Perhitungan Jarak	Hasil d(x,y)
1 dan 2	$d(x,y) = \sqrt{(Nan - 7299{,}99)^2 + (10882 - 88338)^2 + (NaN - 12{,}1)^2 + (3330 - 2635)^2 + (NaN - 0{,}36)^2 + (280 - 280)^2 + (388{,}65 - 315{,}49)^2 + (64 - 64)^2}$	77805,48399
1 dan 6	$d(x,y) = \sqrt{(Nan - 7060)^2 + (10882 - 85861)^2 + (NaN - 12{,}16)^2 + (3330 - 2727)^2 + (NaN - 0{,}39)^2 + (280 - 225)^2 +$	75315,49961

	$(388{,}65-381{,}6)^2+(64-64)^2$	
1 dan 7	$d(x,y)=\sqrt{(Nan-6807{,}98)^2+(10882-83971)^2+(NaN-12{,}33)^2+(3330-2626)^2+(NaN-0{,}39)^2+(280-280)^2+(388{,}65-299{,}9)^2+(64-64)^2}$	73412,19091
1 dan 8	$d(x,y)=\sqrt{(Nan-8399{,}69)^2+(10882-81568)^2+(NaN-9{,}71)^2+(3330-2569)^2+(NaN-0{,}31)^2+(280-280)^2+(388{,}65-291{,}31)^2+(64-64)^2}$	71191,52477
1 dan 11	$d(x,y)=\sqrt{(Nan-5424{,}99)^2+(10882-76455)^2+(NaN-14{,}09)^2+(3330-2695)^2+(NaN-0{,}5)^2+(280-225)^2+(388{,}65-339{,}8)^2+(64-48)^2}$	65803,20024
1 dan 12	$d(x,y)=\sqrt{(Nan-4000)^2+(10882-71646)^2+(NaN-17{,}91)^2+(3330-2097)^2+(NaN-0{,}52)^2+(280-200)^2+(388{,}65-358{,}23)^2+(64-64)^2}$	60920,53776
1 dan 13	$d(x,y)=\sqrt{(Nan-6817)^2+(10882-71576)^2+(NaN-10{,}5)^2+(3330-2054)^2+(NaN-0{,}3)^2+(280-225)^2+(388{,}65-318{,}12)^2+(64-64)^2}$	61102,35337
1 dan 14	$d(x,y)=\sqrt{(Nan-5045{,}99)^2+(10882-68749)^2+(NaN-13{,}62)^2+(3330-2145)^2+(NaN-0{,}43)^2+(280-225)^2+(388{,}65-305{,}55)^2+(64-64)^2}$	58110,84469
1 dan 17	$d(x,y)=\sqrt{(Nan-3684)^2+(10882-67748)^2+(NaN-18{,}39)^2+(3330-2155)^2+(NaN-0{,}59)^2+(280-225)^2+(388{,}65-301{,}1)^2+(64-48)^2}$	57009,52856

c. Mengurutkan hasil *nearest neighbor* yang telah didapatkan dari perhitungan jarak *Euclidean* dengan mengurutkannya dari nilai terbesar hingga nilai terkecil

TABEL IV
HASIL NEAREST NEIGHBOR SECARA TERURUT

Nearest Neighbor
77805,48
75315,5
73412,19
71191,52
65803,2
61102,35
60920,54
58110,84
57009,53

d. Memilih record yang memiliki jarak terkecil dengan data *missing value* berdasarkan parameter K yang telah ditentukan sebelumnya yaitu K=5

TABEL V
NEAREST NEIGHBOR K

K = 5
65803,2
61102,35
60920,54
58110,84
57009,53

e. Melakukan imputasi *missing value* dengan menghitung nilai bobot dengan menggunakan persamaan *weight mean estimation*.

TABEL VI
VARIABLE PERHITUNGAN WEIGHT MEAN ESTIMATION

w_k	v_k
$\frac{1}{65803{,}2^2}=$ 0,000000000230944	5424,99
$\frac{1}{61102{,}35^2}=$ 0,000000000267845	6817
$\frac{1}{60920{,}54^2}=$ 0,000000000269446	4000
$\frac{1}{58110{,}84^2}=$ 0,000000000296132	5045,99
$\frac{1}{57009{,}53^2}=$ 0,000000000307684	3684

Value Wk didapatkan dari persamaan (4). Value Vk adalah value asli yang dipilih berdasarkan target variabel mana yang ingin diisi. *Missing value* yang diisi pada contoh kasus gambar tabel 1 adalah atribut price observasi 1. Setelah variable telah didapatkan lakukan pengisian value dengan menggunakan persamaan (3) *weight mean estimation*.

$$x_j = \frac{0{,}000006784343064}{0{,}000000001372052} = 4944{,}66$$

Hasil dari value diatas akan diisikan pada atribut price pada data observasi pertama yang memiliki *missing value*. Berikut gambar tabel 1 yang telah diisi dengan metode KNN:

observasi	price	cpuMark	cpuValue	hreadMar	hreadValu	TDP	powerPerf	cores
1	4971,277	10882	14,94744	3330	0,468778	280	388,65	64
2	7299,99	88338	12,1	2635	0,36	280	315,49	64
6	7060	85861	12,16	2727	0,39	225	381,6	64
7	6807,98	83971	12,33	2626	0,39	280	299,9	64
8	8399,69	81568	9,71	2569	0,31	280	291,31	64
11	5424,99	76455	14,09	2695	0,5	225	339,8	48
12	4000	71646	17,91	2097	0,52	200	358,23	64
13	6817	71576	10,5	2054	0,3	225	318,12	64
14	5045,99	68749	13,62	2145	0,43	225	305,55	64
17	3684	67748	18,39	2155	0,59	225	301,1	48

Gambar 5 Tabel Hasil Pengisian Missing Value dengan Metode Imputasi KNN

Nilai yang telah didapatkan diatas tentu akan berbeda jika data yang digunakan adalah data asli. Berikut adalah gambar tabel 3 yang telah diisi dengan metode imputasi KNN:

observasi	price	cpuMark	cpuValue	hreadMar	hreadValu	TDP	powerPerf	cores
1	7061,017	108822	12,02842	3330	0,385003	280	388,65	64
2	7299,99	88338	12,1	2635	0,36	280	315,49	64
3	7087,813	86006	12,08601	2387	0,385115	243,1272	355,3134	64
4	7060	85861	12,16	2727	0,39	225	381,6	64
5	6807,98	83971	12,33	2626	0,39	280	299,9	64
6	8399,69	81568	9,71	2569	0,31	280	291,31	64
7	7528,762	80842	11,14853	3340	0,35981	280	288,72	32
8	5963,099	77460	13,34223	2564	0,445732	238,4226	325,257	60
9	5424,99	76455	14,09	2695	0,5	225	339,8	48
10	4000	71646	17,91	2097	0,52	200	358,23	64

Gambar 6 Tabel Hasil Pengisian Missing Value dengan Metode Imputasi KNN pada Data Asli

Hasil pengukuran waktu eksekusi sebesar 0,7 detik menunjukkan bahwa metode ini memerlukan waktu yang cukup signifikan untuk mengeksekusi tugasnya. KNN bekerja dengan mencari tetangga terdekat dari data yang hilang dan melakukan imputasi berdasarkan nilai-nilai tetangga ini. Ini adalah proses yang memerlukan waktu, terutama jika data memiliki dimensi yang tinggi seperti pada data *CPU Benchmark Compilation* yang memiliki dimensi tinggi.

D. Data Transformation

Data yang telah diatasi permasalahan *missing value* akan masuk kedalam proses data transformation untuk diubah nilai satuannya menjadi nilai satuan yang memiliki rentang yang sama. Strategi yang bisa dilakukan untuk mengubah nilai satuan tersebut yaitu, menggunakan metode normalisasi untuk menskalakan data numerik ke dalam bentuk rentang yang lebih adil untuk kedua nilai satuan pada masing-masing value atribut. Salah satu teknik normalisasi yang dilakukan pada penelitian ini adalah teknik *Min-Max normalization* karena karakteristik dataset memiliki jarak value yang diketahui. Berikut adalah sampel 10 data setelah normalisasi dilakukan:

observasi	price	cpuMark	cpuValue	hreadMar	hreadValu	TDP	powerPerf	cores
1	0,786385	1	0,034216	0,767327	0,000953	0,932886	0,285497	0,797468
2	0,813014	0,811633	0,034424	0,603489	0,000859	0,932886	0,231568	0,797468
3	0,789371	0,790188	0,034383	0,545026	0,000953	0,809152	0,260923	0,797468
4	0,786272	0,788855	0,034598	0,625177	0,000971	0,748322	0,2803	0,797468
5	0,758188	0,771475	0,03509	0,601367	0,000971	0,932886	0,220075	0,797468
6	0,935557	0,749377	0,027498	0,58793	0,000672	0,932886	0,213743	0,797468
7	0,838507	0,742701	0,031667	0,769684	0,000858	0,932886	0,211834	0,392405
8	0,664041	0,711601	0,038023	0,586752	0,001179	0,793365	0,238767	0,746835
9	0,604078	0,702359	0,04019	0,617633	0,001382	0,748322	0,249488	0,594937
10	0,445287	0,658136	0,051259	0,476662	0,001457	0,66443	0,263073	0,797468

Gambar 7 Tabel Data Hasil Normalisasi

E. Clustering

Data yang telah diolah pada fase *preprocessing* menggunakan metode imputasi dan telah diolah menggunakan metode normalisasi, dikelompokkan menggunakan algoritma *K-means Clustering*. Data hasil pengelompokkan tersebut dirangkum pada gambar tabel berikut:

Observasi	cpuName	price	cpuMark	cpuValue	threadMark	threadValue	TDP	powerPerf	cores	testDate	socket	category	cluster
1	AMD Ryze	7061,02	108822	12,03	3330	0,39	280	388,65	64	2022	sWRX8	Desktop	0
2	AMD EPYC	7299,99	88338	12,1	2635	0,36	280	315,49	64	2021	SP3	Server	0
3	AMD EPYC	7087,81	86006	12,09	2387	0,39	243,13	355,31	64	2021	unknown	Server	0
4	AMD EPYC	7060	85861	12,16	2727	0,39	225	381,6	64	2021	SP3	Server	0
5	AMD Ryze	6807,98	83971	12,33	2626	0,39	280	299,9	64	2020	sWRX8	Desktop	0
...													
2001	AMD Athlo	62,85	2183	34,73	1165	18,54	65	33,58	4	2015	FM1	Desktop	2
2002	MT6765H	588,11	2183	16,92	631	4,55	30,37	163,55	8	2022	unknown	Mobile/En	3
2003	MT6769V/	270,07	2177	37,33	1041	18,29	30,14	94,28	8	2021	unknown	Mobile/En	3
2004	Intel Atom	97,17	2174	46,18	291	8,71	20	108,7	8	2015	BGA 1283	Server	3
2005	AMD Phen	21	2173	103,46	1156	55,05	95	22,87	4	2009	AM3	Desktop	2
...													
3821	Intel Penti	172,38	84	4,47	225	6,68	38	2,22	1	2009	PGA478	Desktop	3
3822	Intel Penti	115,37	83	5,76	180	8,67	54,7	1,52	1	2009	PGA423	Desktop	3
3823	Intel Penti	70,25	81	5,11	223	7,68	57,8	1,41	1	2009	PGA423	Desktop	3
3824	VIA Eden S	155,39	80	4,95	83	7,29	5	16,08	1	2017	NanoBGA2	Laptop	3
3825	Intel Penti	133,24	77	5,36	203	7,99	51,6	1,5	1	2009	PGA423	Desktop	3

Gambar 8 Tabel Data Hasil Cluster dengan Metode Imputasi KNN

Observasi	cpuName	price	cpuMark	cpuValue	threadMark	threadValue	TDP	powerPerf	cores	testDate	socket	category	cluster
1	AMD Ryze	441,5	108822	35,35	3330	15,19	280	388,65	64	2022	sWRX8	Desktop	1
2	AMD EPYC	7299,99	88338	12,1	2635	0,36	280	315,49	64	2021	SP3	Server	1
3	AMD EPYC	441,5	86006	35,35	2387	15,19	62,38	121,62	64	2021	unknown	Server	1
4	AMD EPYC	7060	85861	12,16	2727	0,39	225	381,6	64	2021	SP3	Server	1
5	AMD Ryze	6807,98	83971	12,33	2626	0,39	280	299,9	64	2020	sWRX8	Desktop	1
...													
2000	Unisoc T6	441,5	2184	35,35	1146	15,19	62,38	121,62	8	2021	unknown	Mobile/En	2
2001	AMD Athlo	62,85	2183	34,73	1165	18,54	65	33,58	4	2015	FM1	Desktop	2
2002	MT6765H	441,5	2183	35,35	631	15,19	62,38	121,62	8	2022	unknown	Mobile/En	2
2003	MT6769V/	441,5	2177	35,35	1041	15,19	62,38	121,62	8	2021	unknown	Mobile/En	2
2004	Intel Atom	441,5	2174	35,35	291	15,19	20	108,7	8	2015	BGA 1283	Server	2
...													
3821	Intel Penti	441,5	84	35,35	225	15,19	38	2,22	1	2009	PGA478	Desktop	2
3822	Intel Penti	441,5	83	35,35	180	15,19	54,7	1,52	1	2009	PGA423	Desktop	2
3823	Intel Penti	441,5	81	35,35	223	15,19	57,8	1,41	1	2009	PGA423	Desktop	2
3824	VIA Eden S	441,5	80	35,35	83	15,19	5	16,08	1	2017	NanoBGA2	Laptop	2
3825	Intel Penti	441,5	77	35,35	203	15,19	51,6	1,5	1	2009	PGA423	Desktop	2

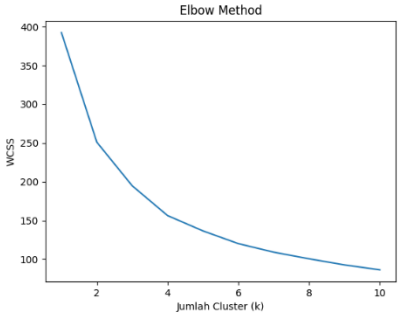
Gambar 9 Tabel Data Hasil Cluster dengan Metode Imputasi Mean

F. Penentuan Jumlah Cluster Menggunakan Elbow Method

Hasil dari penggunaan metode elbow pada dataset *CPU Benchmark Compilation* menunjukan bahwa penentuan kelompok yang paling optimal untuk digunakan pada dataset tersebut adalah sebagai berikut:

TABEL VII
HASIL METODE ELBOW PADA DATASET CPU BENCHMARK COMPILATION

Value K	Elbow
K = 2	251.094
K = 3	194.796
K = 4	156.328
K = 5	136.564



Gambar 10 Metode Elbow pada Dataset CPU Benchmark Compilation

G. Evaluasi Hasil Cluster

Pada evaluasi selanjutnya menggunakan metode *silhouette coefficient* melihat hasil dari data *cluster* yang menggunakan metode imputasi KNN dan data *cluster* yang menggunakan metode imputasi *Mean*. Hasil tersebut dimuat pada tabel berikut.

TABEL VIII
HASIL SILHOUETTE COEFFICIENT PADA DATA CLUSTER DENGAN IMPUTASI KNN

Value K	Silhouette coefficient
K = 2	0.737724
K = 3	0.754597
K = 4	0.777649

TABEL IX
HASIL SILHOUETTE COEFFICIENT PADA DATA CLUSTER DENGAN IMPUTASI MEAN

Value K	Silhouette coefficient
K = 2	0.745421
K = 3	0.762945
K = 4	0.782254

Hasil tabel diatas menunjukkan bahwa *silhouette coefficient* pada data kluster dengan imputasi *mean* memiliki skor yang lebih tinggi, mendekati angka 1. Jika angka tersebut mendekati 1, itu menandakan bahwa objek dalam *cluster* yang dibantu dengan metode imputasi *mean* lebih serupa satu sama lain, sementara objek antara *cluster* tersebut lebih berbeda dibandingkan dengan *cluster* yang dibantu dengan metode imputasi KNN.

H. Interpretasi Hasil Cluster

Interpretasi hasil *cluster* pada data *CPU Benchmark Compilation* ini menjelaskan interpretasi *cluster* yang dibantu dengan metode imputasi KNN dan metode imputasi *Mean*. Data hasil pengelompokkan *cluster* dengan menggunakan data imputasi KNN dan *mean* dimuat pada gambar berikut:

	price	cpuMark	cpuValue	threadMark	threadValue	TDP	powerPerf	cluster	core	category	jumlah data
MEAN	2484,478	33979,5	22,22157	2325,3687	1,8874654	181,3843	183,0509	0	16	Server	217
MAX	8978	108822	113,09	4317	11,55	300	388,65				
MIN	249,99	5985	2,03	1037	0,21	120	27,21				
MEAN	373,5281	12373,81	41,70175	2610,4101	9,7643137	48,81889	331,2725	1	4	Desktop/Laptop	612
MAX	1682,02	46195	134,18	4162	51,76	150	1357,93				
MIN	43	2169	5,18	1307	1,49	4,5	71,21				
MEAN	242,5433	4548,284	42,00706	1529,3842	19,687256	76,05053	70,76309	2	4	Desktop	1403
MAX	3838	26446	345,33	2511	267,82	160	390,38				
MIN	3,99	469	0,98	390	0,13	10	3,61				
MEAN	147,216	1000,003	15,6218	673,62461	12,931067	30,73607	61,68073	3	2	Laptop	1593
MAX	3382,07	7353	121,8	1583	75,04	115	496,55				
MIN	9,1	77	0,22	75	0,23	2	1,35				

Gambar 11 Tabel Interpretasi Data Hasil Cluster dengan Metode Imputasi KNN

	price	cpuMark	cpuValue	threadMark	threadValue	TDP	powerPerf	cluster	core	category	jumlah data
MEAN	2094,872	34024,32	24,28024	2330,75943	4,102971698	177,3003	178,4985	1	16	Server	212
MAX	8978	108822	113,09	4317	15,19	300	388,65				
MIN	249,99	5985	2,03	1037	0,21	62,38	27,21				
MEAN	390,0448	14207,29	43,73509	2741,45633	11,89460699	48,2731	374,8014	0	4	Desktop/Laptop	458
MAX	1808,99	46195	134,18	4162	46,03	150	1357,93				
MIN	43	2169	6,06	1307	1,31	4,5	84,55				
MEAN	326,1257	5296,445	40,45135	1699,03026	17,39242066	71,76844	90,65653	3	4	Desktop	1355
MAX	3838	26659	345,33	2805	267,82	160	415,82				
MIN	3,99	943	1,52	925	0,36	6,5	7,54				
MEAN	346,7142	1124,329	30,68542	705,051111	15,67528333	45,35921	73,81463	2	2	Laptop	1800
MAX	3780	9236	300,6	1433	180,84	140	496,55				
MIN	3,99	77	0,22	75	0,13	2	1,35				

Gambar 12 Tabel Interpretasi Data Hasil Cluster dengan Metode Imputasi Mean

Interpretasi kedua data tersebut bisa difokuskan pada atribut yang sebelumnya memiliki *missing value* seperti atribut price. Pada data *cluster* dengan metode imputasi KNN memiliki rata-rata harga yang terbilang berbeda-beda pada setiap *cluster*. Sedangkan untuk *cluster* dengan metode imputasi *mean*, memiliki rata-rata harga yang mirip pada atribut price berkategori kategori desktop/laptop, desktop, laptop. Hal tersebut bisa terjadi dikarenakan pengisian *missing value* dengan menggunakan metode imputasi *mean* menyebabkan semua *missing value* yang diisi pada atribut tertentu memiliki value yang sama. Rata-rata nilai yang serupa pada *cluster* 0, 2, dan 3, yang dibantu dengan metode imputasi *mean*, memengaruhi interpretasi *cluster*, terutama jika atribut price dianggap penting untuk pemisahan suatu kelompok. Oleh karena itu, metode imputasi KNN lebih disarankan untuk mengatasi masalah nilai yang hilang.

Berdasarkan hasil pengelompokkan data pada kedua gambar 11 dan gambar 12, hasil *clustering* yang didukung oleh metode imputasi KNN telah terbukti memberikan hasil yang lebih baik dan informatif untuk interpretasi *cluster*. Seperti contoh pada hasil pengelompokkan *cluster* ke 3 memiliki rata-rata harga yang terbilang rendah daripada *cluster* lain. Hal ini dapat menjadi indikasi penting dalam pemilihan suatu CPU.

Harga tersebut juga dapat mempengaruhi dalam pemilihan suatu *CPU*. jika harga yang ditawarkan pada *CPU* suatu *cluster* relatif rendah, maka performa *CPU* yang didapat user nantinya tergolong rendah. Hal ini juga dapat memberikan pemahaman yang lebih baik tentang pemilihan suatu *CPU* berdasarkan kebutuhan user nantinya.

Label yang dapat diberikan berdasarkan *cluster* yang dihasilkan dibantu dengan metode imputasi KNN tersebut dapat diartikan seperti berikut: *cluster* 0 untuk pengelompokkan data CPU "Ultimate" dikarenakan data yang terdapat pada *cluster* tersebut memiliki karakteristik CPU yang terbilang paling tinggi dibandingkan dengan *cluster* lain, *cluster* 1 untuk pengelompokkan data CPU "High", *cluster* 2 untuk pengelompokkan data CPU "Mid" dan terakhir *cluster* 3 untuk pengelompokkan data CPU

"Low" dikarenakan *cluster* tersebut menghimpun pengelompokkan data *CPU* yang memiliki performa rendah.

IV. KESIMPULAN

Dalam penelitian ini, ditemukan bahwa hasil interpretasi *cluster K-means* yang dibantu dengan metode imputasi KNN lebih memberikan hasil pengelompokkan yang informatif dan mudah dimengerti. Namun, perlu dicatat bahwa evaluasi dengan menggunakan metrik seperti *silhouette coefficient* menunjukkan bahwa hasil *cluster K-means* dibantu dengan metode imputasi *mean* menghasilkan *cluster* yang lebih baik daripada metode imputasi KNN. Hasil tersebut juga menyimpulkan bahwa pentingnya memahami keunggulan masing-masing metode imputasi dan mempertimbangkan tujuan analisis yang diinginkan. Peneliti juga perlu mempertimbangkan hasil yang dapat muncul dari beberapa teknik interpretasi *cluster* untuk mencapai pemahaman yang lebih baik tentang hasil pengelompokkan *cluster*.

DAFTAR PUSTAKA

[1] M. R. Nahjan, N. Heryana, and A. Voutama, "Implementasi Rapidminer Dengan Metode Clustering K-Means Untuk Analisa Penjualan Pada Toko Oj Cell," *J. Mhs. Tek. Inform.*, vol. 7, no. 1, pp. 1–4, 2023.

[2] R. Hasibuan Budiansyah, H. Hafizah, and R. Mahyuni, "Penerapan Data Mining Clustering Dengan Menggunakan Algoritma K-Means Pada Data Nasabah Kredit Bermasalah PT. BPR Milala," *J-SISKO TECH (Jurnal Teknol. Sist. Inf. dan Sist. Komput. TGD)*, vol. 5, no. 1, p. 7, 2022, doi: 10.53513/jsk.v5i1.4767.

[3] J. Hutagalung, "Pemetaan Siswa Kelas Unggulan Menggunakan Algoritma K-Means Clustering," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 9, no. 1, pp. 606–620, 2022, doi: 10.35957/jatisi.v9i1.1516.

[4] J. Han, Jiawei; Kamber, Micheline; Pei, *Data Mining: Concepts and Techniques*, 3rd ed. Waltham: Morgan Kaufmann, 2011.

[5] T. Raudhatunnisa and N. Wilantika, "Performance Comparison of Hot-Deck Imputation, K-Nearest Neighbor Imputation, and Predictive Mean Matching in Missing Value Handling, Case Study: March 2019 SUSENAS Kor Dataset," *Proc. Int. Conf. Data Sci. Off. Stat.*, vol. 2021, no. 1, pp. 753–770, 2022, doi: 10.34123/icdsos.v2021i1.93.

[6] D. M. P. Murti, U. Pujianto, A. P. Wibawa, and M. I. Akbar, "K-Nearest Neighbor (K-NN) based Missing Data Imputation," *Proceeding - 2019 5th Int. Conf. Sci. Inf. Technol. Embrac. Ind. 4.0 Towar. Innov. Cyber Phys. Syst. ICSITech 2019*, pp. 83–88, 2019, doi: 10.1109/ICSITech46713.2019.8987530.

[7] A. Fadlil, Herman, and D. Praseptian M, "K Nearest Neighbor Imputation Performance on Missing Value Data Graduate User Satisfaction," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 6, no. 4, pp. 570–576, 2022, doi: 10.29207/resti.v6i4.4173.

[8] T. Andriyanto, M. Cs, R. Indriati, and M. Kom, "Rekomendasi Spesifikasi Processor Menggunakan Analisa K-Means Cluster," *Simki-Techsain*, vol. 2, 2018, [Online]. Available: http://simki.unpkediri.ac.id/detail/14.1.03.03.0143

[9] E. Sartika, "Analisis Metode K Nearest Neighbor Imputation (KNNI) untuk Mengatasi Data Hilang Pada Estimasi Data Survey," *J. TEDC*, vol. 12, no. 3, pp. 219–227, 2018.

[10] T. Hendrawati, "Kajian Metode Imputasi dalam Menangani Missing Data," *Pros. Semin. Nas. Mat. dan Pendidik. Mat. UMS*, pp. 637–642, 2015, [Online]. Available: http://hdl.handle.net/11617/5804

[11] C. Curley, R. M. Krause, R. Feiock, and C. V. Hawkins, "Dealing with Missing Data: A Comparative Exploration of Approaches Using the Integrated City Sustainability Database," *Urban Aff. Rev.*, vol. 55, no. 2, pp. 591–615, 2019, doi: 10.1177/1078087417726394.

[12] A. Ilham, "Hybrid Metode Bootstrap Dan Teknik Imputasi Pada Metode C4-5 Untuk Prediksi Penyakit Ginjal Kronis," *Statistika*, vol. 8, no. 1, pp. 43–51, 2020.

[13] R. G. Minakshi Vohra, "Missing Value Imputation in Multi Attribute Data Set," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 4, pp. 5315–5321, 2014.

[14] X. Liu, X. Lai, and L. Zhang, "A hierarchical missing value imputation method by correlation-based K-nearest neighbors," *Adv. Intell. Syst. Comput.*, vol. 1037, pp. 486–496, 2020, doi: 10.1007/978-3-030-29516-5_38.

[15] S. Martha and E. Sulistianingsih INTISARI, "K Nearest Neighbor Dalam Imputasi Missing Data," *Bul. Ilm. Math. Stat. dan Ter.*, vol. 07, no. 1, pp. 9–14, 2018, [Online]. Available: http://archive.ics.uci.edu/ml/datas/Iris.

[16] A. Supriyadi, A. Triayudi, and I. D. Sholihati, "Perbandingan Algoritma K-Means Dengan K-Medoids Pada Pengelompokan Armada Kendaraan Truk Berdasarkan Produktivitas," *JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform.*, vol. 6, no. 2, pp. 229–240, 2021, doi: 10.29100/jipi.v6i2.2008.

[17] D. A. Nasution, H. H. Khotimah, and N. Chamidah, "Perbandingan Normalisasi Data untuk Klasifikasi Wine Menggunakan Algoritma K-NN," *Comput. Eng. Sci. Syst. J.*, vol. 4, no. 1, p. 78, 2019, doi: 10.24114/cess.v4i1.11458.

[18] L. Suriani, "Pengelompokan Data Kriminal Pada Poldasu Menentukan Pola Daerah Rawan Tindak Kriminal Menggunakan Data Mining Algoritma K-Means Clustering," *J. Sist. Komput. dan Inform.*, vol. 1, no. 2, p. 151, 2020, doi: 10.30865/json.v1i2.1955.

[19] A. M. Sikana and A. W. Wijayanto, "Analisis Perbandingan Pengelompokan Indeks Pembangunan Manusia Indonesia Tahun 2019 dengan Metode Partitioning dan Hierarchical Clustering," *J. Ilmu Komput.*, vol. 14, no. 2, p. 66, 2021, doi: 10.24843/jik.2021.v14.i02.p01.

[20] F. Yunita, "Penerapan Data Mining Menggunkan Algoritma K-Means Clustring Pada Penerimaan Mahasiswa Baru," *Sistemasi*, vol. 7, no. 3, p. 238, 2018, doi: 10.32520/stmsi.v7i3.388.

[21] S. Aulia, "Klasterisasi Pola Penjualan Pestisida Menggunakan Metode K-Means Clustering (Studi Kasus Di Toko Juanda Tani Kecamatan Hutabayu Raja)," *Djtechno J. Teknol. Inf.*, vol. 1, no. 1, pp. 1–5, 2021, doi: 10.46576/djtechno.v1i1.964.

[22] I. Wahyudi, M. B. Sulthan, and L. Suhartini, "Analisa Penentuan Cluster Terbaik Pada Metode K-Means Menggunakan Elbow Terhadap Sentra Industri Produksi Di Pamekasan," *J. Apl. Teknol. Inf. dan Manaj.*, vol. 2, no. 2, pp. 72–81, 2021, doi: 10.31102/jatim.v2i2.1274.

[23] S. Asmiatun, "Penerapan Metode K-Medoids Untuk Pengelompokkan Kondisi Jalan Di Kota Semarang," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 6, no. 2, pp. 171–180, 2019, doi: 10.35957/jatisi.v6i2.193.