Analisa Pengelompokan terhadap Individu Yang Mengalami Obesitas Dengan Menggunakan Metode K-Means

PROJECT UAS PEMBELAJARAN
MESIN KELAS C

Oleh

Aldiva Fadlie Fauzan

202131017

FAKULTAS TELEMATIKA ENERGI

INSTITUT TEKNOLOGI PERUSAHAAN LISTRIK NEGARA

JAKARTA

2024

Abstrak

Dalam Penelitian ini fokus pada analisis pengelompokan individu yang mengalami obesitas melalui penerapan metode K-Means. Dengan menggunakan data klinis dan demografis, penelitian ini bertujuan untuk mengidentifikasi kelompok homogen berdasarkan faktor-faktor seperti indeks massa tubuh (BMI), pola makan, aktivitas fisik, dan aspek kesehatan lainnya. Hasil analisis ini diharapkan memberikan wawasan yang lebih jelas terkait karakteristik dan pola perilaku dalam populasi obesitas, memungkinkan pengembangan strategi intervensi yang lebih spesifik dan personal. Penelitian ini menjadi kontribusi penting untuk mendukung pendekatan yang lebih terarah dalam menangani masalah obesitas.

Kata Kunci — Metode K-Means, faktor-faktor tertentu, hasil analisa.

Abstract

In this research, the focus is on analyzing the grouping of individuals who are obese through the application of the K-Means method. Using clinical and demographic data, this study aims to identify homogeneous groups based on factors such as body mass index (BMI), diet, physical activity and other health aspects. The results of this analysis are expected to provide clearer insight into the characteristics and behavioral patterns in the obese population, enabling the development of more specific and personalized intervention strategies. This research is an important contribution to supporting a more targeted approach in dealing with the problem of obesity.

Keywords — K-Means method, certain factors, analysis resul

Daftar Isi

Contents

BAB I

PENDAHULUAN

1.1 Latar Belakang

Obesitas telah menjadi isu kesehatan global yang semakin meresahkan, mempengaruhi berbagai aspek kehidupan masyarakat. Pertumbuhan prevalensi obesitas menandakan perlunya pendekatan yang lebih cermat dan terarah dalam upaya pencegahan serta penanganan. Dalam menghadapi kompleksitas masalah obesitas, diperlukan pemahaman yang lebih mendalam mengenai variasi karakteristik dan faktor yang mempengaruhi individu yang mengalami obesitas.

Pengelompokan individu berdasarkan karakteristik tertentu dapat menjadi kunci untuk menyusun strategi intervensi yang lebih efektif dan personalisasi dalam mengatasi masalah obesitas. Dalam konteks ini, metode analisis pengelompokan, seperti K-Means, menawarkan pendekatan yang potensial untuk mengidentifikasi pola-pola yang mungkin tersembunyi dalam data obesitas.

1.2 Rumusan Masalah

Meskipun telah ada berbagai penelitian terkait obesitas, penggunaan metode K-Means dalam konteks analisis pengelompokan untuk memahami variasi karakteristik individu yang mengalami obesitas masih terbatas. Oleh karena itu, penelitian ini bertujuan untuk merumuskan masalah sebagai berikut:

1. Bagaimana metode K-Means dapat digunakan untuk menganalisis dan mengelompokkan individu yang mengalami obesitas berdasarkan karakteristik tertentu?
2. Apa saja faktor-faktor yang dapat mempengaruhi pembentukan kelompok-kelompok dalam populasi obesitas?
3. Bagaimana hasil analisis pengelompokan dapat memberikan wawasan tambahan terkait dengan pola perilaku, demografi, dan faktor kesehatan pada populasi obesitas.

1.3 Tujuan

Penelitian ini bertujuan untuk menganalisis pengelompokan individu yang mengalami obesitas dengan menggunakan metode K-Means. Melalui analisis ini, diharapkan dapat diidentifikasi kelompok-kelompok dengan karakteristik serupa dalam populasi obesitas dan memberikan wawasan yang lebih mendalam terkait dengan faktor-faktor yang mempengaruhi pembentukan kelompok-kelompok tersebut.

1.4 Manfaat

1. Memberikan kontribusi pada pemahaman mendalam tentang variasi dalam populasi obesitas.
2. Memfasilitasi penyedia layanan kesehatan dalam merancang intervensi yang lebih terarah dan personalisasi.
3. Membuka peluang untuk penelitian lanjutan mengenai faktor-faktor risiko dan pola perilaku dalam konteks obesitas.

BAB II

KAJIAN PUSTAKA

2.1 Penelitian yang Relevan

Untuk memperkuat hasil penelitian, pada Bab ini berisikan tentang beberapa penelitian terdahulu yang akan dibahas sebagai pembanding serta pedoman dalam memahami dan merancang sebuah metode yang digunakan. Sebagai pembanding penelitian maka akan dirangkum penelitian terdahulu pada Tabel 2.1 sebagai berikut :

Tabel 2.1 Perbandingan Penelitian Dengan Penelitian yang Relevan

No.	1
Judul	K-means clustering of overweight and obese population using quantile-transformed metabolic data
Penulis	Li Li, Qifa Song, and Xi Yang
Tahun	2019
Hasil	Pengelompokan K-means dengan bantuan transformasi kuantil nilai atribut diterapkan untuk mengatasi dampak variasi nilai atribut obesitas yang cukup besar yang melibatkan outlier dan distribusi skewed.
Keterkaitan Penelitian	Keterkaitan antara judul saya dan jurnal tersebut adalah bahwa keduanya menggunakan data untuk memecahkan masalah. Penggunaan pengelompokan K-means untuk teknologi big data untuk mengelompokkan populasi yang kelebihan berat badan dan obesitas secara metabolik.

No.	2
Judul	***CASE BASED REASONING (CBR) FOR OBESITY LEVEL ESTIMATION USING K-MEANS INDEXING METHOD***
Penulis	Samantha Thomas, Sophie Lewis & John Petkov
Tahun	2013
Hasil	Metode pengindeksan seperti Algoritma K-Means diperlukan agar pencarian kasus serupa tidak melibatkan seluruh kasus pada satu basis kasus sehingga dapat mempersingkat waktu komputasi pada tahap pengambilan dan tetap menghasilkan solusi yang optimal. Kesamaan kosinus digunakan untuk menemukan kelompok kasus baru yang relevan dan kesamaan jarak Euclidean digunakan untuk menghitung kesamaan antar kasus. Metode subsampling acak digunakan untuk memvalidasi sistem CBR.
Keterkaitan Penelitian	Keterkaitan antara judul saya dan jurnal tersebut adalah bahwa keduanya menggunakan data untuk memecahkan masalah. Penggunaan pengelompokan K-means untuk teknologi big data untuk mengelompokkan populasi yang kelebihan berat badan dan obesitas secara metabolik.

No.	3
Judul	*Implementation of K-Means, K-Medoid and DBSCAN Algorithms In Obesity Data Clustering*
Penulis	**Elsa Setiawati, Ustara Dwi Fernanda, Suci Agesti**
Tahun	2024-01-10

Hasil	Penelitian ini fokus pada penentuan jumlah cluster yang optimal dalam konteks data obesitas, hal ini untuk memahami variasi kompleks pada populasi pasien obesitas dan merinci karakteristik klinis yang dapat digunakan untuk melakukan clustering lebih lanjut menggunakan K-Means, K-Medoid. dan algoritma DBSCAN. Dari penelitian ini dapat disimpulkan bahwa algoritma K-Means merupakan pilihan terbaik untuk clustering data obesitas.
Keterkaitan Penelitian	Keterkaitan antara judul saya dan jurnal tersebut adalah bahwa keduanya menggunakan data untuk memecahkan masalah. Penggunaan pengelompokan K-means untuk teknologi big data untuk mengelompokkan populasi yang kelebihan berat badan dan obesitas secara metabolik. -Medoid. dan algoritma DBSCAN. Dari penelitian ini dapat disimpulkan bahwa algoritma K-Means merupakan pilihan terbaik untuk clustering data obesitas.

No.	4
Judul	***Clustering as Data Mining Technique in Risk Factors Analysis of Diabetes, Hypertension and Obesity.***
Penulis	Mohammed Gulam Ahamad, Mohammed Faisal Ahmed, Mohammed Yousuf Uddin
Tahun	6: DECEMBER 2016
Hasil	Investigasi ini mengeksplorasi data mining menggunakan perangkat lunak open source WEKA dalam aplikasi layanan kesehatan. Teknik analisis cluster digunakan untuk mempelajari dampak diabetes, obesitas dan hipertensi dari database yang diperoleh dari Virginia School of Medicine. Teknik cluster k-means sederhana diadopsi untuk membentuk sepuluh cluster yang terlihat jelas untuk membedakan perbedaan antara faktor risiko seperti diabetes, obesitas dan hipertensi.
Keterkaitan Penelitian	Keterkaitan antara judul saya dan jurnal tersebut adalah bahwa keduanya menggunakan data untuk memecahkan masalah. Penggunaan pengelompokan K-means untuk teknologi big data untuk mengelompokkan populasi yang kelebihan berat badan dan obesitas secara metabolik.

No.	5
Judul	*A Comparative Analysis of Classification Techniques in Data Mining Algorithms*
Penulis	A.S.K.K. Sankar, V. Nirmala
Tahun	2023

Hasil	Studi ini memberikan gambaran yang komprehensif tentang keunggulan dan kelemahan masing-masing algoritma dalam konteks tertentu. Menyoroti keakuratan dan efisiensi relatif dari algoritma-algoritma tersebut. Jurnal ini membahas perbandingan berbagai teknik klasifikasi dalam algoritma data mining. Penelitian tersebut memberikan wawasan mendalam mengenai keakuratan, kecepatan, dan kekokohan berbagai algoritma dalam berbagai dataset.
Keterkaitan Penelitian	Keterkaitan antara kedua jurnal tersebut adalah bahwa keduanya menggunakan metode kalsifikasi untuk melakukan analisis data. Mengevaluasi dan membandingkan keakuratan, kecepatan (waktu CPU yang dikonsumsi), dan kekokohan berbagai algoritma klasifikasi dalam berbagai dataset.

2.2 Pembelajaran Mesin

Pembelajaran mesin adalah salah satu subdisiplin dalam bidang kecerdasan buatan yang fokus pada pengembangan algoritma dan model komputasi yang memungkinkan komputer untuk belajar dari data. Ini memungkinkan komputer untuk memahami pola, membuat prediksi, dan mengambil keputusan berdasarkan data yang diberikan, tanpa harus diprogram secara eksplisit. Kajian pustaka dalam pembelajaran mesin mencakup berbagai aspek yang mencerminkan perkembangan dan tren terbaru dalam bidang ini. Beberapa topik utama yang dibahas dalam kajian pustaka ini meliputi:

a. Algoritma Pembelajaran Mesin: Ini mencakup pengenalan berbagai algoritma pembelajaran mesin, termasuk regresi linear, pohon keputusan, k-means clustering, jaringan saraf tiruan, dan algoritma pembelajaran mendalam (deep learning). Kajian pustaka mengulas prinsip kerja, keuntungan, dan kelemaha.
b. Data dan Preprocessing: Bagian penting dalam pembelajaran mesin adalah data. Kajian pustaka akan membahas pentingnya data yang berkualitas, teknik pengumpulan data, serta langkah-langkah preprocessing data seperti normalisasi, pengisian data yang hilang, dan ekstraksi fitur.
c. Validasi Model: Penelitian mengenai metode validasi model yang digunakan untuk mengukur kinerja algoritma pembelajaran mesin. Ini mencakup konsep validasi silang (cross-validation), evaluasi model, serta bagaimana mencegah overfitting.
d. Penerapan Pembelajaran Mesin: Kajian pustaka juga mencakup aplikasi pembelajaran mesin dalam berbagai bidang, seperti pengenalan wajah, pengolahan bahasa alami, kendaraan otonom, perawatan kesehatan, dan sebagainya.

2.3 Clustering

Clustering adalah proses pengelompokan benda serupa ke dalam kelompok yang berbeda, atau lebih tepatnya partisi dari sebuah data set kedalam subset, sehingga data dalam setiap subset memiliki arti yang bermanfaat. Dimana sebuah cluster terdiri dari kumpulan benda-benda yang mirip antara satu denganyang lainnya dan berbeda dengan benda yang terdapat pada cluster lainnya. Algoritma clustering terdiri dari dua bagian yaitu secara hirarkis dan secara partitional. Algoritma hirarkis menemukan cluster secara berurutan dimana cluster ditetapkan sebelumnya, sedangkan algoritma partitional menentukan semua kelompok pada waktu tertentu (Madhulatha, 2012). Clustering juga bisa dikatakan suatu proses dimana mengelompokan dan membagi pola data menjadi beberapa jumlah data set sehingga akan membentuk pola yang serupa dan dikelompokan pada cluster yang sama dan memisahkan diri dengan membentuk pola yang berbeda ke cluster yang berbeda (HUNG et al., 2005).

Clustering dapat memainkan peran penting dalam kehidupan seharihari, karena tidak bisa lepas dengan sejumlah data yang menghasilkan informasi untuk memenuhi kebutuhan hidup. Salah satu sarana yang paling 13 penting dalam hubungan dengan data adalah untuk mengklasifikasikan atau mengelompokkan data tersebut ke dalam seperangkat kategori atau cluster. Clustering dapat ditemukan dibeberapa aplikasi yang ada di berbagai bidang. Sebagai contoh pengelompokan data yang digunakan untuk menganalisa data statistik seperti pengelompokan untuk pembelajaran mesin, data mining, pengenalan pola, analisis citra dan bioinformatika (Bataineh et al., 2011).

2.4 K – Means

Algoritma K-means merupakan salah satu algoritma dengan partitional, karena K-Means didasarkan pada penentuan jumlah awal kelompok dengan mendefinisikan nilai centroid awalnya (Madhulatha, 2012). Algoritma K -means menggunakan proses secara berulang-ulang 14 untuk mendapatkan basis data cluster. Dibutuhkan jumlah cluster awal yang diinginkan sebagai masukan dan menghasilkan titik centroid akhir sebagai output. Metode K-means akan memilih pola k sebagai titik awal centroid secara acak atau random. Jumlah iterasi untuk mencapai cluster centroid akan dipengaruhi oleh calon cluster centroid awal secara random. Sehingga didapat cara dalam pengembangan algoritma dengan menentukan centroid cluster yang dilihat dari kepadatan data awal yang tinggi agar mendapatkan kinerja yang lebih tinggi (HUNG et al., 2005, Saranya & Punithavalli, 2011, Eltibi & Ashour, 2011)

Dalam penyelesaiannya, algoritma K-Means akan menghasilkan titik centroid yang dijadikan tujuan dari algoritma K-Means. Setelah iterasi KMeans berhenti , setiap objek dalam dataset menjadi anggota dari suatu cluster. Nilai cluster ditentukan dengan mencari seluruh objek untuk menemukan cluster dengan jarak terdekat ke objek . Algoritma K -means akan mengelompokan item data dalam suatu dataset ke suatu cluster berdasarkan jarak terdekat (Bangoria et al., 2013).

BAB III

HASIL DAN PEMBAHASAN

3.1 K-Means

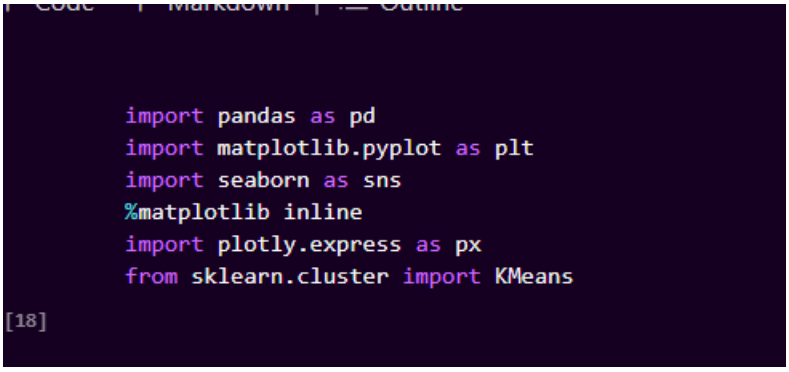
K-Means adalah salah satu algoritma pengelompokan (clustering) yang digunakan dalam analisis data dan machine learning.

3.1.1 Pengambilan Dataset

Kali ini saya mengambil dataset yang akan saya uji dari suatu website yaitu Kaggle.com.

Link Dataset :

3.1.2 Import Library



- import pandas as pd: Mengimpor pustaka pandas dengan memberikan alias 'pd'. Pandas digunakan untuk manipulasi dan analisis data.
- import matplotlib.pyplot as plt: Mengimpor modul pyplot dari matplotlib dengan memberikan alias 'plt'. Matplotlib digunakan untuk membuat visualisasi grafis.
- import seaborn as sns: Mengimpor pustaka seaborn dengan memberikan alias 'sns'. Seaborn merupakan pustaka visualisasi data yang dibangun di atas matplotlib, menyediakan tata letak yang lebih cantik dan mudah digunakan.
- rom sklearn.cluster import KMeans: Mengimpor kelas KMeans dari modul cluster dalam pustaka scikit-learn. Ini merupakan implementasi algoritma K-Means yang akan digunakan untuk melakukan klasterisasi.

3.1.3 Reading Dataset (Membaca Dataset)

```
df = pd.read_csv('Obesity Classification.csv')
```
[19]

```
df.head()
```
[20]

	ID	Age	Gender	Height	Weight	BMI	Label
0	1	25	Male	175	80	25.3	Normal Weight
1	2	30	Female	160	60	22.5	Normal Weight
2	3	35	Male	180	90	27.3	Overweight
3	4	40	Female	150	50	20.0	Underweight
4	5	45	Male	190	100	31.2	Obese

Dalam dua baris kode tersebut, pertama-tama, dataset dengan nama 'Mall_Customers.csv' dibaca menggunakan fungsi read_csv dari pustaka pandas dan disimpan dalam variabel read. Setelah itu, fungsi head() digunakan untuk menampilkan lima baris pertama dari dataset, memberikan gambaran awal tentang struktur dan konten dataset tersebut. Fungsi ini berguna untuk melihat secara cepat variabel-variabel dan nilai-nilai awal dalam dataset sehingga dapat dilakukan eksplorasi awal terhadap data yang akan digunakan dalam analisis atau pemrosesan lebih lanjut.

3.1.4 Proses Pengolahan Data Dan Mepenampilan Data

```
df.info()
```
[21]

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 108 entries, 0 to 107
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ID      108 non-null    int64
 1   Age     108 non-null    int64
 2   Gender  108 non-null    object
 3   Height  108 non-null    int64
 4   Weight  108 non-null    int64
 5   BMI     108 non-null    float64
 6   Label   108 non-null    object
dtypes: float64(1), int64(4), object(2)
memory usage: 6.0+ KB
```

```
df.nunique()
```
[22]

```
ID        108
Age        75
Gender      2
Height     10
Weight     23
BMI        25
Label       4
dtype: int64
```

- Metode **df.info()** digunakan untuk memberikan informasi rinci tentang DataFrame, termasuk informasi tentang jenis data, jumlah nilai non-null, dan penggunaan memori
- Metode **df.nunique()** digunakan untuk menghitung jumlah nilai unik (distinct) dalam setiap kolom DataFrame.

fig = px.histogram(...): Membuat objek plot histogram dengan menggunakan fungsi **px.histogram()** dari Plotly Express. Argumen-argumen yang diberikan adalah:

- **data_frame**: DataFrame yang berisi data.
- **x**: Kolom yang akan diplot pada sumbu x, dalam hal ini adalah 'Age' (usia).
- **color**: Kolom yang akan digunakan untuk memberi warna, dalam hal ini adalah 'Gender' (jenis kelamin).
- **facet_col**: Membuat subplot berdasarkan kolom 'Gender' (jenis kelamin).
- **marginal**: Menambahkan box plot pada bagian tepi histogram.
- **title**: Judul dari plot, yaitu 'Age distribution'.
- **color_discrete_sequence**: Sekuens warna diskrit untuk jenis kelamin.

fig.show(): Menampilkan plot yang telah dibuat.

```
fig=px.sunburst(df,path=['Gender','Label'],values=df.value_counts().values,color_discrete_sequence=['#2BCDC1','#F66095'])
fig.show()
```

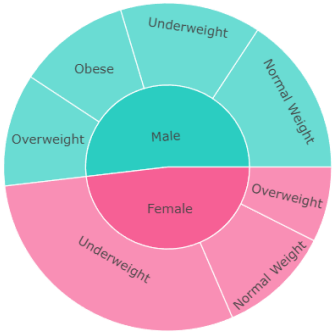


fig = px.sunburst(df, path=['Gender','Label'], values=df.value_counts().values, color_discrete_sequence=['#2BCDC1','#F66095']):

- **px.sunburst()**: Ini adalah fungsi dari Plotly Express untuk membuat diagram sunburst.
- **df**: Ini adalah DataFrame yang berisi data yang akan digunakan untuk membuat diagram sunburst.
- **path=['Gender','Label']**: Menentukan jalur atau hirarki data. Dalam hal ini, diagram akan memiliki tingkat hirarki berdasarkan kolom 'Gender' dan 'Label'.
- **values=df.value_counts().values**: Menentukan nilai atau bobot untuk setiap kategori di tingkat terakhir diagram. Dalam hal ini, digunakan nilai jumlah kemunculan setiap kategori dalam kolom 'Label'.
- **color_discrete_sequence=['#2BCDC1','#F66095']**: Menentukan urutan warna diskrit untuk setiap kategori. Dalam hal ini, dua warna yang ditetapkan adalah '#2BCDC1' dan '#F66095'.

```
import plotly.figure_factory as ff

fig=ff.create_distplot([df[df['Gender']=='Male']['BMI'],df[df['Gender']=='Female']['BMI']],['Male','Female'],colors=['#2BCDC1','#F66095'])
fig.update_layout(title_text='BMI distribution')
fig.show()
```

Pada baris ini, menggunakan fungsi **import plotly.figure_factory as ff**. Fungsi ini menerima beberapa argumen:

- **[df[df['Gender']=='Male']['BMI'], df[df['Gender']=='Female']['BMI']]**: Ini adalah data yang akan digunakan untuk plot. Data ini terdiri dari dua kelompok, yaitu BMI untuk pria dan BMI untuk wanita. Data ini diberikan dalam bentuk daftar.
- **['Male', 'Female']**: Ini adalah label untuk masing-masing kelompok. Dalam hal ini, label 'Male' untuk kelompok BMI pria dan 'Female' untuk kelompok BMI wanita.
- **colors=['#2BCDC1', '#F66095']:** Ini adalah parameter opsional yang menentukan warna untuk masing-masing kelompok. Dalam hal ini, warna hijau (#2BCDC1) digunakan untuk kelompok pria dan warna merah muda (#F66095) untuk kelompok wanita.

```
df2=df.drop('ID',axis=1)
df2=pd.get_dummies(df2)
df2.head()
```

	Age	Height	Weight	BMI	Gender_Female	Gender_Male	Label_Normal Weight	Label_Obese	Label_Overweight	Label_Underweight
0	25	175	80	25.3	0	1	1	0	0	0
1	30	160	60	22.5	1	0	1	0	0	0
2	35	180	90	27.3	0	1	0	0	1	0
3	40	150	50	20.0	1	0	0	0	0	1
4	45	190	100	31.2	0	1	0	1	0	0

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

df3=df.copy()
df3['Gender'] = le.fit_transform(df3['Gender'])
```

1. **idf2 = df.drop('ID', axis=1)**: Pada baris ini, DataFrame **df** disalin ke dalam DataFrame baru yang disebut **idf2**. Namun, kolom dengan nama 'ID' dihapus dari DataFrame baru ini. Operasi **drop** digunakan untuk menghapus kolom atau baris tertentu dari DataFrame. Parameter **axis=1** mengindikasikan bahwa kita ingin menghapus kolom.
2. **df2 = pd.get_dummies(idf2)**: Baris ini membuat DataFrame baru yang disebut **df2** dengan menggunakan fungsi **get_dummies** dari pandas. Fungsi ini digunakan untuk melakukan one-hot encoding pada kolom-kolom kategorikal dalam DataFrame. Dengan kata lain, setiap nilai kategorikal dalam kolom dipecah menjadi beberapa kolom baru (dummy variables) yang mewakili kehadiran atau ketiadaan nilai tersebut. Ini membantu dalam memproses data kategorikal saat membangun model machine learning.
3. **df2.head()**: Baris terakhir menampilkan lima baris pertama dari DataFrame **df2** menggunakan fungsi **head()**. Ini membantu untuk memberikan preview cepat tentang bagaimana DataFrame terlihat setelah operasi yang dilakukan di atas.

1. **from sklearn.preprocessing import LabelEncoder**: Mengimpor kelas **LabelEncoder** dari modul **preprocessing** di scikit-learn.
2. **le = LabelEncoder()**: Membuat objek **LabelEncoder** dengan nama **le**. Objek ini akan digunakan untuk mengubah nilai kategori menjadi bilangan bulat.
3. **df3 = df.copy()**: Membuat salinan dataframe **df** dan menetapkannya ke dataframe baru **df3**. Ini dilakukan agar perubahan tidak mempengaruhi dataframe asli.
4. **df3['Gender'] = le.fit_transform(df3['Gender'])**: Menggunakan **fit_transform** dari objek **LabelEncoder** (**le**) untuk mengubah nilai-nilai dalam kolom 'Gender' dari dataframe **df3**. Proses ini akan menggantikan nilai-nilai kategori dengan bilangan bulat yang sesuai. Metode **fit_transform** digunakan karena itu mencakup dua tahap: pelatihan model dengan menghitung parameter yang diperlukan untuk transformasi, dan transformasi sebenarnya. Dalam hal ini, metode ini dijalankan pada kolom 'Gender' untuk mengubah nilai kategori menjadi bilangan bulat.

3.1.5 Pengelompokan Usia

```
[29]
# Age binning
df4=df.copy()
df4['Age'].min(),df4['Age'].max()

...  (11, 112)
```

```
[30]
bins=[0,20,40,60,80,100,200]
df4['Age_bins']=pd.cut(df4['Age'],bins)
df4.head(3)
```

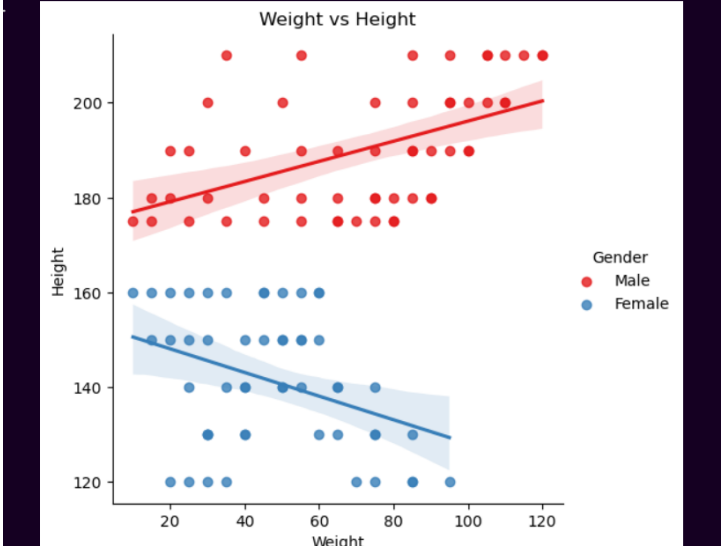
	ID	Age	Gender	Height	Weight	BMI	Label	Age_bins
0	1	25	Male	175	80	25.3	Normal Weight	(20, 40]
1	2	30	Female	160	60	22.5	Normal Weight	(20, 40]
2	3	35	Male	180	90	27.3	Overweight	(20, 40]

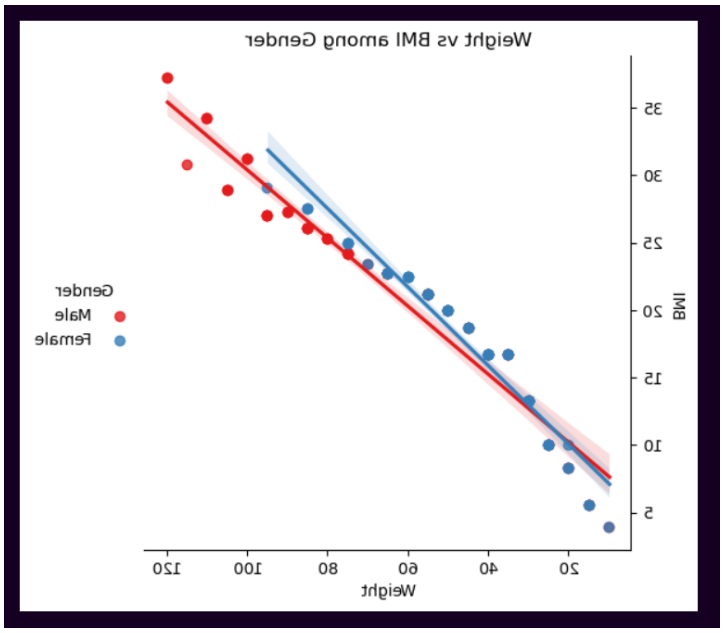
1. **df4=df.copy()**: Membuat salinan dari DataFrame yang ada (**df**) dan menyimpannya dalam variabel baru **df4**. Tujuannya adalah untuk memodifikasi DataFrame yang baru tanpa merusak yang asli.
2. **df4['Age'].min(), df4['Age'].max()**: Menggunakan metode **min()** dan **max()** untuk menemukan nilai minimum dan maksimum dari kolom 'Age' dalam DataFrame **df4**. Ini memberikan dua nilai, yaitu nilai minimum dan maksimum dari kolom 'Age'.

1. **bins=[0,20,40,60,80,100,200]**: Ini adalah daftar nilai batas untuk membuat kelompok (bins) berdasarkan rentang usia. Misalnya, kelompok pertama adalah [0,20), kelompok kedua adalah [20,40), dan seterusnya.
2. **df4['Age_bins']=pd.cut(df4['Age'], bins)**: Pada baris ini, kita menggunakan fungsi **pd.cut** dari pustaka Pandas untuk membuat kolom baru 'Age_bins' dalam DataFrame **df4**. Fungsi ini memotong nilai-nilai dalam kolom 'Age' ke dalam kelompok-kelompok yang ditentukan oleh **bins**.
    - **df4['Age']**: Ini adalah kolom 'Age' dalam DataFrame **df4**, yang berisi nilai usia.
    - **bins**: Parameter ini berisi daftar nilai batas yang telah kita tentukan sebelumnya.
3. **df4.head(3)**: Baris ini digunakan untuk menampilkan tiga baris pertama dari DataFrame **df4** setelah penambahan kolom baru 'Age_bins'.

```
sns.lmplot(x='Weight',y='Height',hue='Gender',data=df4,palette='Set1')
plt.title('Weight vs Height')
plt.show()

sns.lmplot(x='Weight',y='BMI',hue='Gender',data=df4,palette='Set1')
plt.title('Weight vs BMI among Gender')
plt.show()
```

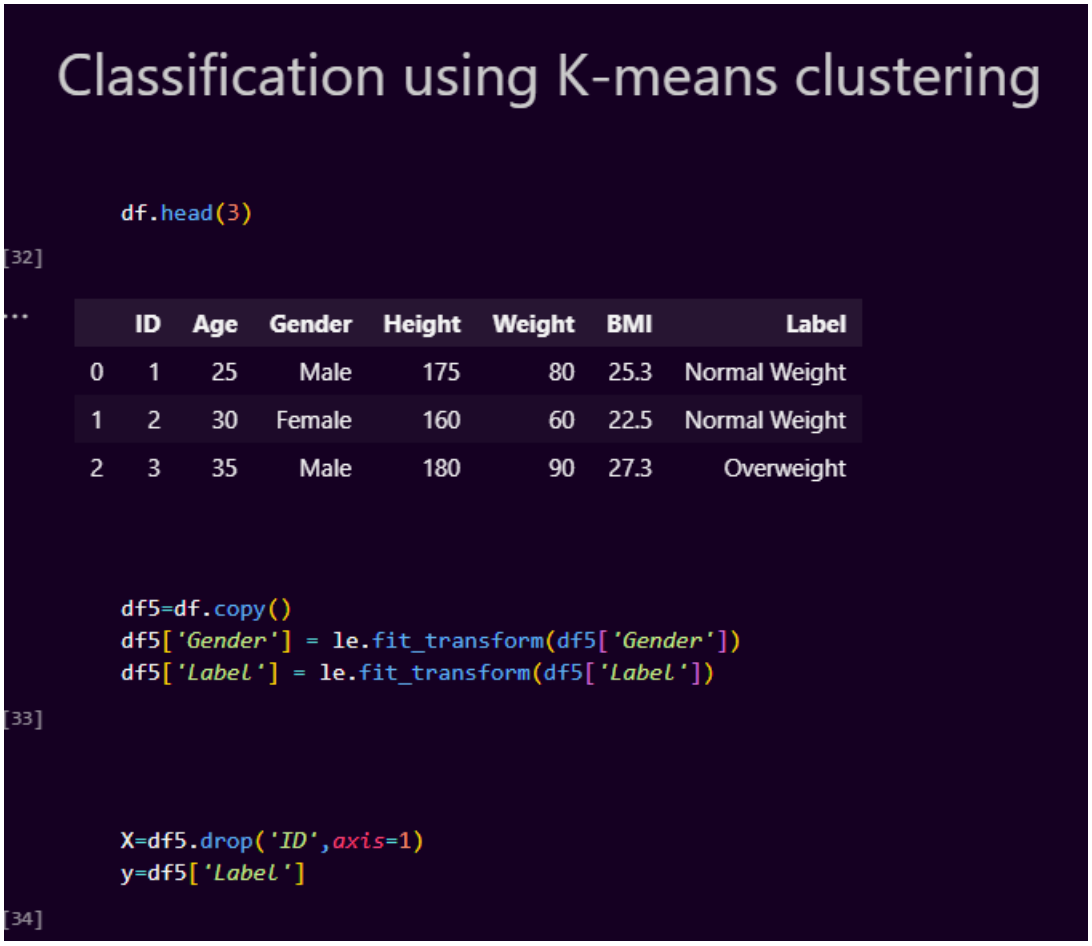




1. **sns.lmplot**: Fungsi ini digunakan untuk membuat scatter plot dengan garis regresi linear di atasnya. Di sini, sumbu x diisi dengan data dari kolom 'Weight', sumbu y diisi dengan data dari kolom 'Height', dan plot dibagi berdasarkan kategori 'Gender'. Palet warna yang digunakan adalah 'Set1'.
2. **plt.title('Weight vs Height')**: Menambahkan judul ke plot, yang dalam hal ini adalah 'Weight vs Height'.
3. **plt.show()**: Menampilkan plot ke layar.

3.1.6 Classification using K-means clustering

1. **sns.lmplot**: Kembali digunakan untuk membuat scatter plot dengan garis regresi linear, namun kali ini sumbu y diisi dengan data dari kolom 'BMI'. Seperti sebelumnya, plot dibagi berdasarkan kategori 'Gender' dan menggunakan palet warna 'Set1'.
2. **plt.title('Weight vs BMI among Gender')**: Menambahkan judul ke plot, yang dalam hal ini adalah 'Weight vs BMI among Gender'.
3. **plt.show()**: Menampilkan plot ke layar.



df.head(3) : menampilkan 10 data teratas

1. **df5=df.copy()**: Membuat salinan DataFrame **df** dan menyimpannya dalam variabel **df5**. Dengan langkah ini, kita dapat melakukan perubahan pada DataFrame **df5** tanpa mempengaruhi DataFrame asli **df**.
2. **df5['Gender'] = le.fit_transform(df5['Gender'])**: Menggunakan LabelEncoder (**le**) untuk melakukan encoding pada kolom 'Gender' dalam DataFrame **df5**. LabelEncoder akan mengubah nilai-nilai dalam kolom 'Gender' menjadi bilangan bulat. Metode **fit_transform** digunakan untuk menghitung parameter transformasi dari data dan kemudian menerapkan transformasi tersebut.
3. **df5['Label'] = le.fit_transform(df5['Label'])**: Sama seperti langkah sebelumnya, namun kali ini encoding dilakukan pada kolom 'Label' dalam DataFrame **df5**.

1. **X = df5.drop('ID', axis=1)**

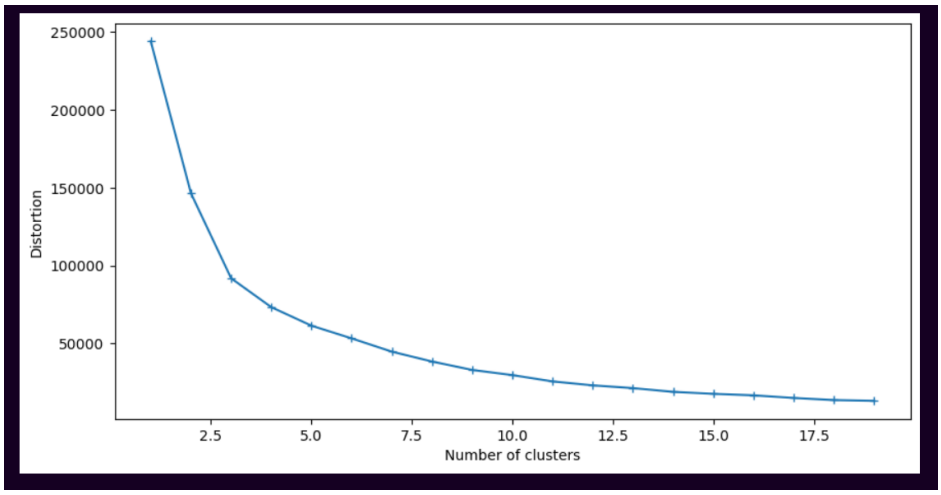
Baris ini membuat variabel **X** yang akan digunakan sebagai features (fitur) dalam suatu model. Secara khusus, 'ID' dihapus dari DataFrame **df5** dengan menggunakan metode **drop**. Parameter **axis=1** digunakan untuk menunjukkan bahwa yang dihapus adalah kolom, bukan baris. Sebagai contoh, jika DataFrame **df5** awalnya memiliki kolom 'ID', baris ini akan menghasilkan DataFrame baru **X** tanpa kolom 'ID'.

2. **y = df5['Label']**

Baris ini membuat variabel **y** yang akan digunakan sebagai target atau label dalam suatu model. 'Label' diambil dari kolom 'Label' dalam DataFrame **df5**. Dengan kata lain, **y** akan berisi nilai-nilai dari kolom 'Label' dalam DataFrame **df5**.

```
dist_list=[]
for i in range(1,20):
  kmeans=KMeans(n_clusters=i,init='random',random_state=101)
  kmeans.fit(X)
  dist_list.append(kmeans.inertia_)

plt.figure(figsize=(10,5))
plt.plot(range(1,20),dist_list,marker='+')
plt.xlabel('Number of clusters')
plt.ylabel('Distortion')
plt.show()
```



1. dist_list=[]: Membuat sebuah list kosong bernama dist_list yang akan digunakan untuk menyimpan nilai inertia dari model KMeans untuk setiap jumlah cluster.
2. for i in range(1, 20):: Melakukan iterasi dari 1 hingga 19 (20 tidak termasuk) untuk mencoba berbagai jumlah cluster dalam rentang tersebut.
3. kmeans=KMeans(n_clusters=i, init='random', random_state=101): Membuat objek KMeans dengan parameter n_clusters diatur sesuai dengan iterasi saat ini, init diatur sebagai 'random' (menggunakan inisialisasi titik pusat acak), dan random_state diatur

sebagai 101 untuk memastikan hasil yang dapat direproduksi.

4. kmeans.fit(X): Melatih model KMeans dengan data X.
5. dist_list.append(kmeans.inertia_): Menambahkan nilai inertia dari model KMeans yang telah dilatih ke dalam list dist_list. Inertia adalah ukuran seberapa dekat titik data dalam suatu cluster dengan pusat clusternya.
6. plt.figure(figsize=(10,5)): Membuat sebuah gambar (figure) dengan ukuran 10x5 inci.
7. plt.plot(range(1,20), dist_list, marker='+'): Membuat plot dengan sumbu x adalah jumlah cluster (dari 1 hingga 19) dan sumbu y adalah nilai inertia yang disimpan dalam dist_list. Marker '+' digunakan untuk menandai setiap titik pada plot.
8. plt.xlabel('Number of clusters'): Menyertakan label pada sumbu x dengan teks 'Number of clusters'.
9. plt.ylabel('Distortion'): Menyertakan label pada sumbu y dengan teks 'Distortion'.
10. plt.show(): Menampilkan plot yang telah dibuat.

```
kmeans4=KMeans(n_clusters=4,random_state=101)

kmeans4.fit(X)

labels=kmeans4.labels_

correct_labels=sum(y==labels)
print('n_clusters=4: %d out of %d samples were correctly labeled.' % (correct_labels,y.size))
print('Accuracy score: {0:0.2f}'.format(correct_labels/float(y.size)))
```

1. **kmeans4 = KMeans(n_clusters=4, random_state=101)**: Membuat objek KMeans dengan menyertakan parameter **n_clusters=4**, yang menentukan jumlah kluster yang ingin dihasilkan oleh algoritma K-Means. **random_state=101** digunakan untuk memastikan hasil yang reprodusibel, artinya, setiap kali kode ini dijalankan, hasilnya akan tetap sama jika nilai **random_state** tidak berubah.
2. **kmeans4.fit(X)**: Melatih model K-Means dengan data **X**. Model ini akan mencoba mengelompokkan data menjadi 4 kluster sesuai dengan jumlah kluster yang ditentukan sebelumnya.
3. **labels = kmeans4.labels_**: Menyimpan hasil label kluster untuk setiap sampel dalam variabel **labels**.
4. **correct_labels = sum(y == labels)**: Menghitung jumlah label yang benar sesuai dengan label sebenarnya (**y**). Ini dilakukan dengan membandingkan setiap elemen **y** dengan label yang diberikan oleh model (**labels**). **sum(y == labels)** akan menghasilkan jumlah elemen yang benar.
5. **print('n_clusters=4: %d out of %d samples were correctly labeled.' % (correct_labels, y.size))**: Mencetak jumlah sampel yang telah diberi label dengan benar dari total sampel. Ini memberikan informasi tentang seberapa baik model K-Means melakukan tugas pengelompokkan pada data.
6. **print('Accuracy score: {0:0.2f}'.format(correct_labels/float(y.size)))**: Mencetak skor akurasi, yang dihitung sebagai rasio jumlah sampel yang benar diberi label dengan total

sampel. Format **{0:0.2f}** digunakan untuk mencetak nilai floating point dengan dua digit desimal.

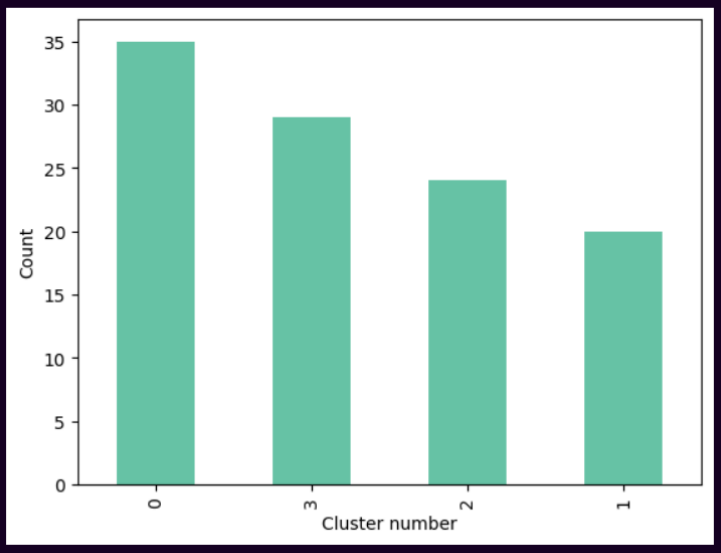
```
[37]
labels=pd.Series(kmeans4.labels_,name='cluster_number')
```

```
[38]
df_with_cluster=pd.concat([df,labels],axis=1)
df_with_cluster.head(3)
```

	ID	Age	Gender	Height	Weight	BMI	Label	cluster_number
0	1	25	Male	175	80	25.3	Normal Weight	2
1	2	30	Female	160	60	22.5	Normal Weight	2
2	3	35	Male	180	90	27.3	Overweight	2

1. **labels=pd.Series(kmeans4.labels_, name='cluster_number')**:
    - **kmeans4.labels_** mengandung informasi tentang klaster (cluster) yang telah ditentukan oleh algoritma K-Means pada data yang ada. Setiap data poin akan diberikan label klaster tertentu.
    - **pd.Series()** digunakan untuk membuat objek Series, yang merupakan struktur data satu dimensi di dalam pustaka pandas.
    - **name='cluster_number'** memberikan nama pada Series yang dibuat, yaitu 'cluster_number'.
    - Hasilnya adalah sebuah objek Series yang berisi label klaster untuk setiap data poin dalam data frame.
2. **df_with_cluster=pd.concat([df, labels], axis=1)**:
    - **pd.concat()** digunakan untuk menggabungkan (concatenate) dua objek pandas, dalam hal ini, data frame **df** dan objek Series **labels**.
    - **[df, labels]** adalah daftar (list) objek yang akan digabungkan. Dalam hal ini, data frame **df** dan objek Series **labels** akan digabungkan berdampingan.
    - **axis=1** menunjukkan bahwa penggabungan dilakukan secara horizontal (sepanjang sumbu kolom). Dengan kata lain, kolom-kolom akan ditambahkan samping-samping.
    - Hasilnya adalah **df_with_cluster**, yaitu data frame baru yang memiliki kolom-kolom dari **df** dan kolom 'cluster_number' yang berisi label klaster.
3. **df_with_cluster.head(3)**:
    - Baris ini menampilkan tiga baris pertama dari data frame **df_with_cluster** untuk memberikan gambaran singkat tentang hasil penggabungan.
    - **head(3)** digunakan untuk menampilkan tiga baris pertama dari data frame.

```
df_with_cluster['cluster_number'].value_counts().plot(kind='bar',cmap='Set2')
plt.xlabel('Cluster number')
plt.ylabel('Count')
plt.show()
```



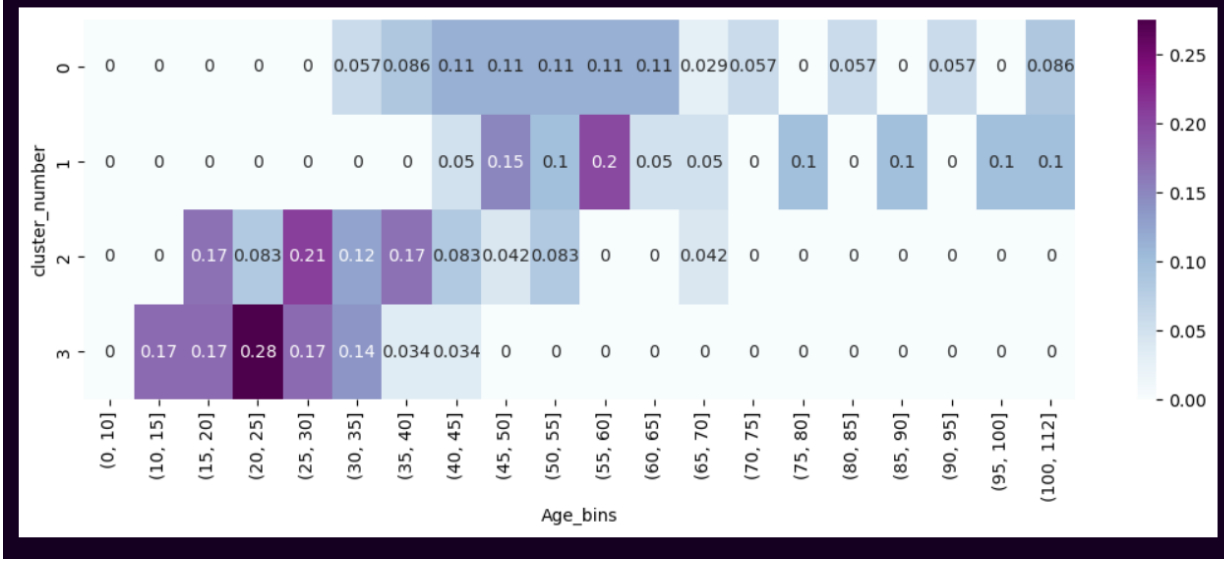
1. df_with_cluster['cluster_number'].value_counts(): Ini adalah bagian pertama dari kode yang menghitung frekuensi kemunculan setiap nilai unik dalam kolom 'cluster_number' dari DataFrame df_with_cluster. Fungsi value_counts() mengembalikan Series yang berisi jumlah kemunculan setiap nilai.
2. .plot(kind='bar', cmap='Set2'): Setelah mendapatkan frekuensi kemunculan, kita menggunakan metode plot() untuk membuat plot. Parameter kind='bar' menunjukkan bahwa kita ingin membuat plot berjenis batang (bar plot). Parameter cmap='Set2' menentukan skema warna yang akan digunakan untuk plot batang. Dalam hal ini, skema warna 'Set2' dipilih.
3. plt.xlabel('Cluster number'): Ini menambahkan label sumbu x pada plot dengan teks 'Cluster number'.
4. plt.ylabel('Count'): Ini menambahkan label sumbu y pada plot dengan teks 'Count'.
5. plt.show(): Ini menampilkan plot yang telah dibuat.

3.1.7 clustering pengelompokan usia

```
# Age binning
bins=[0,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100,112]
df_with_cluster['Age_bins']=pd.cut(df_with_cluster['Age'],bins)



df_cluster_age=df_with_cluster.groupby(['cluster_number','Age_bins']).size().unstack().fillna(0)
sns.heatmap(df_cluster_age.apply(lambda x:x/x.sum(),axis=1),annot=True,cmap='BuPu')
plt.gcf().set_size_inches(13,4)
plt.show()
```



1. **bins=[0,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100,112]**: Ini adalah daftar batas interval yang digunakan untuk membagi nilai dalam kolom '**Age'.** Misalnya, nilai 0-10 akan berada dalam bin pertama, 10-15 dalam bin kedua, dan seterusnya. Interval ini dapat disesuaikan.
2. **pd.cut(df_with_cluster['Age'], bins):** Fungsi cut dari pandas digunakan untuk memotong atau membagi data ke dalam interval yang sudah ditentukan. Pada contoh ini, kolom '**Age**' dari DataFrame **df_with_cluster** dipotong menjadi bin sesuai dengan batas yang didefinisikan oleh variabel bins. Hasilnya disimpan dalam kolom baru yang disebut '**Age_bins'.**

BAB IV

PENUTUP

4.1 Kesimpulan

Penelitian ini memiliki tujuan utama untuk menganalisis pengelompokan individu yang mengalami obesitas menggunakan metode K-Means, dengan harapan dapat mengidentifikasi kelompok-kelompok dengan karakteristik serupa dalam populasi obesitas dan memberikan wawasan lebih mendalam mengenai faktor-faktor yang mempengaruhi pembentukan kelompok-kelompok tersebut.

Dalam proses analisis pengelompokan, penelitian ini fokus pada faktor-faktor seperti indeks massa tubuh (BMI), pola makan, aktivitas fisik, dan aspek kesehatan lainnya. Metode K-Means dipilih sebagai alat analisis yang potensial untuk mengungkap pola-pola tersembunyi dalam data obesitas, sehingga dapat membantu merumuskan strategi intervensi yang lebih efektif dan personalisasi.

4.2 Saran

Pentingnya Integrasi Data. Memastikan data yang digunakan untuk analisa obesitas yang relevan, dan siap untuk diproses. Menggali lebih dalam ke dalam penelitian terdahulu untuk mendapatkan wawasan tambahan dan memperkaya kontekspenelitian.

DAFTAR PUSTAKA

Apa Itu Machine Learning? Beserta Pengertian Dan Cara Kerjanya (2022) *Dicoding Blog*. Available at: https://www.dicoding.com/blog/machine-learning-adalah/ (Accessed: 17 January 2024).

Jurnal Case Based Reasoning (Cbr) For Obesity Level Estimation Using K-Means Indexing Method http://www.kursorjournal.org/index.php/kursor/article/view/268

NEURAL NETWORK Nur Hadianto1; Hafifah Bella Novitasari2; Ami Rahmawat

Cluster Analysis of Obesity Disease Based on Comorbidities Extracted from Clinical Notes Published: 28 January 2019 https://link.springer.com/article/10.1007/s10916-019-1172-1

K-means clustering of overweight and obese population using quantile-transformed metabolic data Published online: 23 Aug 2019 https://www.tandfonline.com/doi/full/10.2147/DMSO.S206640

Predicting the Students Performance using Regularization-based LinearRegressionOctober 2021

LAMPIRAN

- Lampiran 1

  Link github : https://github.com/aldivafzn/K-Means-UAS-PM-C-202131017-ALDIVAFADLIEFAUZAN

- Lampiran 2

Link dataset dari Kaggle : https://www.kaggle.com/code/earije/k-means-clustering-with-obesity-dataset/input