LAPORAN PROYEK UTS DATA SCIENCE

Analisis Data Kualitas Air dan Penyakit Menular untuk Menentukan Faktor Pencemar Utama

Disusun Oleh:

KELOMPOK 01:	
11423001	**Samuel Leonardo Nainggolan**
11423003	**Samuel Rizki Sinambela**
11423061	**Gishella Putri Vehuliza Br Tarigan**

Mata Kuliah: 4143203 - Data Science

Dosen Pengampu: Oppir Hutapea, S.Tr.Kom., M.Kom

PROGRAM STUDI SARJANA TERAPAN TEKNOLOGI REKAYASA PERANGKAT LUNAK

INSTITUT TEKNOLOGI DEL

2025

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kualitas air merupakan faktor penting yang menentukan kesejahteraan dan kesehatan masyarakat. Pencemaran air akibat limbah industri, pertanian, maupun rumah tangga dapat menyebabkan peningkatan kadar zat kimia berbahaya yang berdampak pada lingkungan dan kesehatan manusia. Kandungan zat berbahaya dalam air tersebut seringkali menjadi faktor penyebab munculnya penyakit yang ditularkan melalui air, seperti diare, kolera, hepatitis, dan tifus.

Oleh karena itu, diperlukan pendekatan analisis untuk memahami pola hubungan antara tingkat pencemaran air dan kasus penyakit yang terjadi. Dengan berkembangnya teknologi data, pendekatan Data Science dapat dimanfaatkan untuk menganalisis keterkaitan antara tingkat pencemaran air dan jumlah kasus penyakit.

Proyek ini menggunakan dataset Water Pollution and Disease Dataset, yang memuat informasi kualitas air dari berbagai wilayah beserta catatan jumlah kasus penyakit yang terjadi. Melalui penerapan tahapan data preprocessing, visualisasi data, dan analisis statistik, penelitian ini bertujuan untuk menemukan hubungan yang signifikan antara tingkat pencemaran air dan tingkat penyebaran penyakit. Hasil analisis diharapkan dapat memberikan gambaran yang jelas mengenai dampak pencemaran air terhadap kesehatan masyarakat serta menjadi dasar dalam perumusan kebijakan pengelolaan lingkungan yang lebih baik.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang kami gunakan, terdapat rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana karakteristik dan sebaran tingkat pencemaran air di berbagai wilayah dalam dataset?
2. Apakah terdapat hubungan antara kadar zat berbahaya tertentu seperti nitrat atau bahan organik dalam air dengan jumlah kasus penyakit?

3. Faktor lingkungan apa yang paling berpengaruh terhadap peningkatan kasus penyakit yang ditularkan melalui air?
4. Bagaimana hasil analisis statistik dan visualisasi data dapat membantu memahami hubungan antara kualitas air dan tingkat kejadian penyakit?

1.3 Tujuan Proyek

Adapun tujuan yang ingin dicapai pada proyek ini, yaitu sebagai berikut:

1. Menganalisis data pencemaran air dan penyakit terkait dengan cara yang sistematis untuk memahami pola, karakteristik, dan hubungan antar variabel lingkungan dengan kejadian penyakit.
2. Peningkatan kualitas data melalui teknik preprocessing seperti penanganan nilai hilang, deteksi outlier, normalisasi, dan encoding variabel kategorikal.
3. Identifikasi hubungan antar variabel menggunakan uji statistik, baik uji parametrik maupun non-parametrik, guna memperoleh hasil analisis yang valid dan signifikan secara ilmiah
4. Menyajikan hasil analisis dalam bentuk visualisasi data yang informatif dan mudah dipahami, sehingga dapat mendukung proses pengambilan keputusan dan memberikan wawasan terkait dampak pencemaran air terhadap kesehatan masyarakat.

BAB II
METODOLOGI

2.1 Alur Proses Data Science

Tahapan dalam proses data science merupakan langkah-langkah sistematis yang dilakukan untuk mengubah data mentah menjadi informasi yang bermakna. Secara umum, proses ini terdiri dari beberapa tahap utama, yaitu:

1. Pengumpulan data (data collection)

   Merupakan pengumpulan data dari Proses sumber yang kredibel untuk memastikan keakuratan dan relevansi data terhadap topik yang dianalisis.

2. Pembersihan dan persiapan data (data preprocessing)

   Tahapan yang mencakup pemeriksaan, perbaikan, dan transformasi data mentah agar siap dianalisis. Meliputi penanganan missing values, outlier, normalisasi, dan encoding variabel kategorikal.

3. Visualisasi data (data visualization)

   Merupakan tahapan menyajikan data dalam bentuk grafik atau diagram untuk mempermudah pemahaman pola, tren, dan hubungan antar variabel.

4. Analisis statistik (statistical analysis)

   Merupakan tahapan untuk melakukan uji statistik untuk mengidentifikasi hubungan antar variabel serta menilai signifikansi secara ilmiah.

Setiap tahapan tersebut saling berkaitan dan memiliki peran penting dalam menghasilkan analisis yang valid.

2.2Data Collection

Dataset yang digunakan pada proyek ini berjudul “Water Pollution and Disease” yang diperoleh dari situs Kaggle (https://www.kaggle.com/datasets/khushikyad001/water-pollution-and-disease).

Adapun karakteristik yang terdapat pada dataset, yaitu sebagai berikut:

1. Informasi dataset

```
[4]: print("=== INFORMASI DATASET ===")
     print(f"Jumlah baris dan kolom: {uts.shape}")
     print("\nLima data teratas:")
     print(uts.head())

     # Menampilkan jumlah baris dan kolom
     print("Jumlah baris dan kolom:", uts.shape)
```

```
=== INFORMASI DATASET ===
Jumlah baris dan kolom: (3000, 24)

Lima data teratas:
     Country   Region  Year Water Source Type  Contaminant Level (ppm)  \
0     Mexico    North  2015              Lake                     6.06
1     Brazil     West  2017              Well                     5.24
2  Indonesia  Central  2022              Pond                     0.24
3    Nigeria     East  2016              Well                     7.91
4     Mexico    South  2005              Well                     0.12
```

Berdasarkan hasil pemeriksaan awal menggunakan fungsi shape dan head(), dataset Water Pollution and Disease memiliki 3.000 baris dan 24 kolom.

2. Informasi detail tiap kolom

```
[5]: print("=== INFORMASI DETAIL TIAP KOLOM ===")
     print(uts.info())
```

```
=== INFORMASI DETAIL TIAP KOLOM ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 24 columns):
 #   Column                                      Non-Null Count  Dtype
---  ------                                      --------------  -----
 0   Country                                     3000 non-null   object
 1   Region                                      3000 non-null   object
 2   Year                                        3000 non-null   int64
 3   Water Source Type                           3000 non-null   object
 4   Contaminant Level (ppm)                     3000 non-null   float64
 5   pH Level                                    3000 non-null   float64
 6   Turbidity (NTU)                             3000 non-null   float64
 7   Dissolved Oxygen (mg/L)                     3000 non-null   float64
 8   Nitrate Level (mg/L)                        3000 non-null   float64
 9   Lead Concentration (µg/L)                   3000 non-null   float64
 10  Bacteria Count (CFU/mL)                     3000 non-null   int64
 11  Water Treatment Method                      2253 non-null   object
 12  Access to Clean Water (% of Population)     3000 non-null   float64
 13  Diarrheal Cases per 100,000 people          3000 non-null   int64
 14  Cholera Cases per 100,000 people            3000 non-null   int64
 15  Typhoid Cases per 100,000 people            3000 non-null   int64
 16  Infant Mortality Rate (per 1,000 live births)  3000 non-null   float64
 17  GDP per Capita (USD)                        3000 non-null   int64
 18  Healthcare Access Index (0-100)             3000 non-null   float64
 19  Urbanization Rate (%)                       3000 non-null   float64
 20  Sanitation Coverage (% of Population)       3000 non-null   float64
 21  Rainfall (mm per year)                      3000 non-null   int64
 22  Temperature (°C)                            3000 non-null   float64
 23  Population Density (people per km²)         3000 non-null   int64
dtypes: float64(12), int64(8), object(4)
memory usage: 562.6+ KB
None
```

Berdasarkan hasil pemeriksaan menggunakan fungsi info() menunjukkan bahwa seluruh kolom memiliki 3.000 nilai non-null, artinya tidak terdapat data kosong pada tahap awal.

Setiap kolom memiliki tipe data yang bervariasi, terdiri dari numerik (int64 dan float64) serta kategorikal (object). Dan ukuran total dataset adalah sekitar 562,6 KB..

3. Informasi statistik deskriptif

```
[6]: print("=== INFORMASI STATISTIK DESKRIPTIF ===")
     print(uts.describe())
```

```
=== INFORMASI STATISTIK DESKRIPTIF ===
              Year  Contaminant Level (ppm)     pH Level  Turbidity (NTU)  \
count  3000.000000              3000.000000  3000.000000      3000.000000
mean   2012.012667                 4.954390     7.255847         2.480023
std       7.229287                 2.860072     0.720464         1.419984
min    2000.000000                 0.000000     6.000000         0.000000
25%    2006.000000                 2.560000     6.630000         1.257500
50%    2012.000000                 4.950000     7.280000         2.460000
75%    2018.000000                 7.400000     7.870000         3.660000
max    2024.000000                10.000000     8.500000         4.990000

       Dissolved Oxygen (mg/L)  Nitrate Level (mg/L)  \
count              3000.000000           3000.00000
mean                  6.492850             25.08025
std                   2.027966             14.50517
min                   3.000000              0.05000
25%                   4.710000             12.52500
50%                   6.490000             24.79000
75%                   8.252500             37.91000
max                  10.000000             49.99000
```

Berdasarkan hasil analisis statistik deskriptif melalui fungsi describe() menunjukkan bahwa data numerik memiliki nilai rata-rata dan sebaran yang realistis, seperti Contaminant Level (ppm) dengan rata-rata 4,95, pH Level sekitar 7,25 (netral), serta Dissolved Oxygen (mg/L) dengan rata-rata 6,49, yang menandakan kondisi air tergolong layak. Nilai minimum dan maksimum juga menunjukkan adanya variasi antarwilayah, yang akan relevan untuk analisis selanjutnya.

4. Identifikasi tipe data

```
[7]: print("=== INDENTIFIKASI TIPE DATA ===")
     numeric_cols = uts.select_dtypes(include=['int64', 'float64']).columns
     categorical_cols = uts.select_dtypes(include=['object']).columns

     print("Kolom numerik:", list(numeric_cols))
     print("Kolom kategorikal:", list(categorical_cols))
```

```
=== INDENTIFIKASI TIPE DATA ===
Kolom numerik: ['Year', 'Contaminant Level (ppm)', 'pH Level', 'Turbidity (NTU)', 'Dissolved Oxygen (mg/L)', 'Nitrate Level (mg/L)', 'Lead Concentratio
n (µg/L)', 'Bacteria Count (CFU/mL)', 'Access to Clean Water (% of Population)', 'Diarrheal Cases per 100,000 people', 'Cholera Cases per 100,000 peopl
e', 'Typhoid Cases per 100,000 people', 'Infant Mortality Rate (per 1,000 live births)', 'GDP per Capita (USD)', 'Healthcare Access Index (0-100)', 'Ur
banization Rate (%)', 'Sanitation Coverage (% of Population)', 'Rainfall (mm per year)', 'Temperature (°C)', 'Population Density (people per km²)']
Kolom kategorikal: ['Country', 'Region', 'Water Source Type', 'Water Treatment Method']
```

Berdasarkan hasil identifikasi tipe data menunjukkan bahwa dataset terdiri dari:

- Tipe data numerik pada kolom Contaminant Level (ppm), pH Level, Nitrate Level (mg/L), Temperature (°C), GDP per Capita (USD), dan lain-lain.
- Tipe data kategorikal pada kolom Country, Region, Water Source Type, dan Water Treatment Method.

Alasan Pemilihan Dataset ini, yaitu:

a. Dataset ini memenuhi seluruh kriteria proyek UTS:
b. Memiliki lebih dari 20 fitur dan 2000 baris data.
c. Mengandung hubungan sebab-akibat yang jelas antara kualitas air dan penyakit.
d. Data bersifat asli dan kontekstual, bukan data simulasi.
e. Dapat digunakan untuk analisis deskriptif, korelasi, dan inferensial statistik.

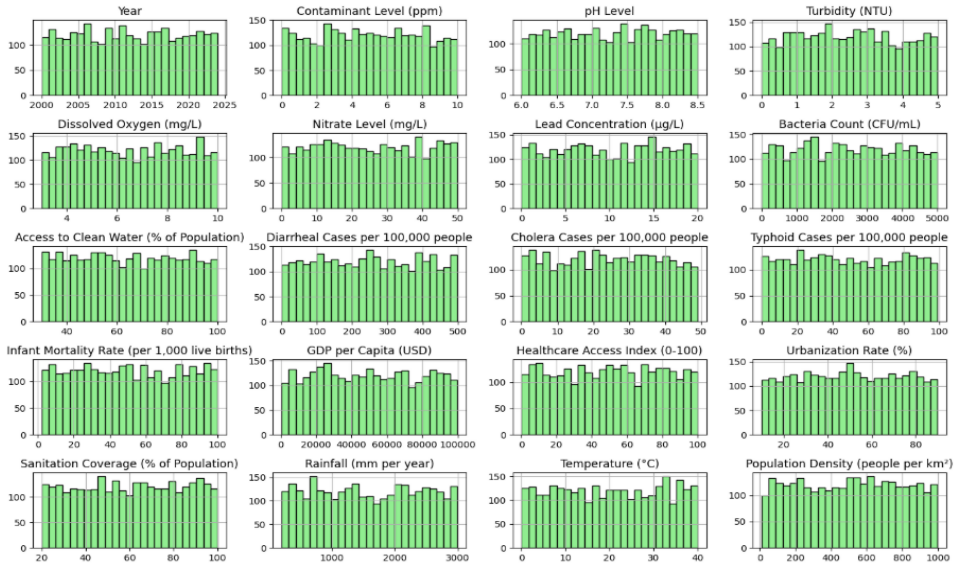
5. Visualisasi data mentah

Visualisasi data mentah dilakukan untuk memahami distribusi awal setiap fitur numerik dalam dataset Water Pollution and Disease. Sebelum dilakukan proses preprocessing seperti penanganan missing values atau outlier, visualisasi ini membantu melihat pola dasar dan potensi permasalahan pada data.

Berikut adalah gambar dari visualisasi data mentah pada dataset ini:

```
[9]: print("=== VISUALISASI DATA MENTAH NUMERIK ===")
     uts.hist(figsize=(14, 10), bins=25, color='lightgreen', edgecolor='black')
     plt.suptitle("Distribusi Fitur Numerik Sebelum Preprocessing", fontsize=16)
     plt.tight_layout(rect=[0, 0, 1, 0.96])
     plt.show()
```

```
=== VISUALISASI DATA MENTAH NUMERIK ===
```



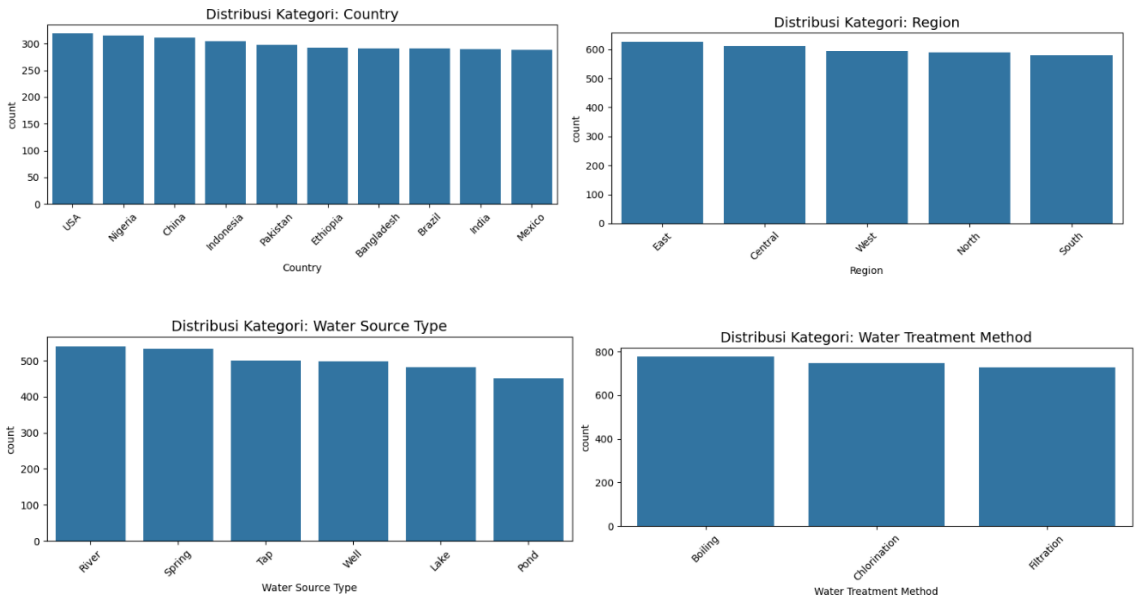
Gambar di atas menampilkan histogram untuk setiap fitur numerik dalam dataset, yang menggambarkan distribusi nilai dari masing-masing atribut pada rentang tertentu. Dimana

setiap grafik memperlihatkan seberapa sering nilai-nilai tertentu muncul di dalam data, sehingga memudahkan untuk melihat pola sebaran, bentuk distribusi, serta potensi keberadaan outlier.

Selain fitur numerik, dataset Water Pollution and Disease juga memiliki beberapa fitur kategorikal, seperti Country, Region, Water Source Type, dan Water Treatment Method.

Untuk memahami distribusi awal dari masing-masing kategori, dilakukan visualisasi menggunakan grafik batang atau bar chart dengan bantuan library Seaborn.

```
[10]: print("=== VISUALISASI DATA MENTAH KATEGORIKAL ===")
categorical_cols = uts.select_dtypes(include=['object']).columns

for col in categorical_cols:
    plt.figure(figsize=(8, 4))
    sns.countplot(data=uts, x=col, order=uts[col].value_counts().index)
    plt.title(f"Distribusi Kategori: {col}", fontsize=14)
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
```



Beberapa hal yang dapat diamati dari hasil visualisasi tersebut adalah:

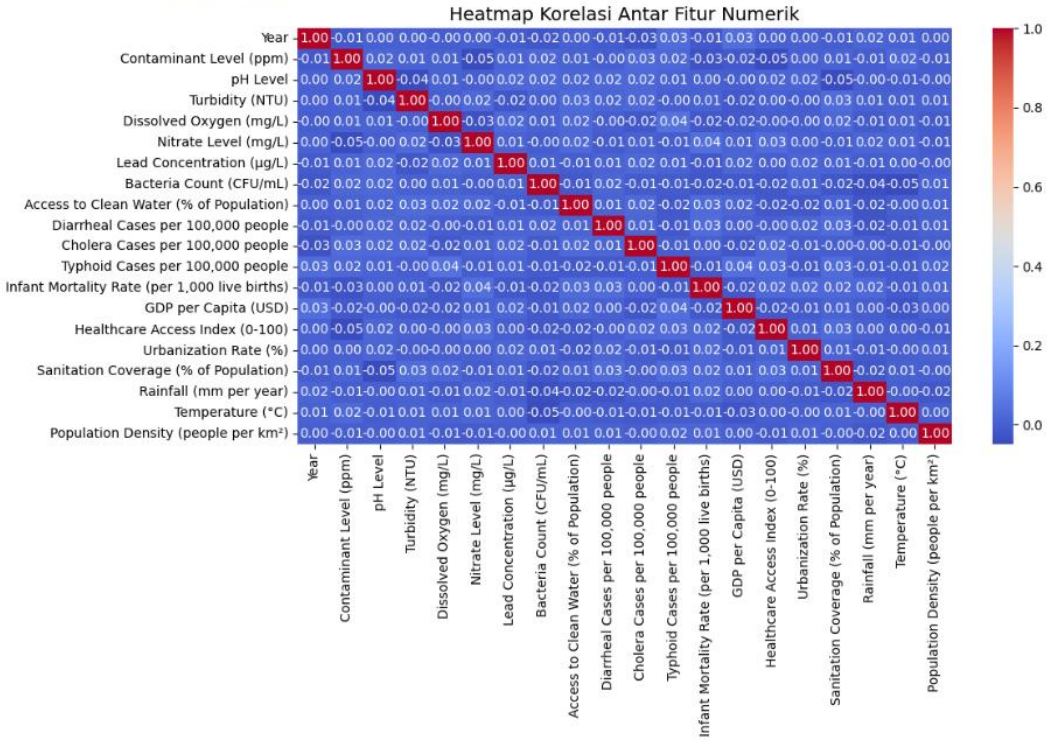
1. Fitur Country memperlihatkan distribusi yang cukup seimbang antara negara. Hal ini menunjukkan bahwa data dikumpulkan secara proporsional dari berbagai wilayah.
2. Fitur Region juga memiliki distribusi yang relatif merata yang menandakan tidak ada bias signifikan terhadap wilayah tertentu.
3. Fitur Water Source Type memperlihatkan bahwa sumber air. Hal ini memberikan gambaran awal tentang sumber air yang paling umum digunakan dalam dataset.

4. Fitur Water Treatment Method menampilkan tiga metode utama dengan distribusi yang relatif seimbang, menandakan variasi metode pengolahan air di berbagai lokasi.

Selanjutnya, menampilkan visualisasi korelasi antar fitur numerik, dimana yang dilakukan untuk melihat hubungan atau keterkaitan antar variabel numerik dalam dataset. Visualisasi ini bertujuan mengidentifikasi pola hubungan linear, baik positif maupun negatif.

```
[11]: print("=== VISUALISASI DATA MENTAH ANTAR NUMERIK ===")
      plt.figure(figsize=(12, 8))
      sns.heatmap(uts.corr(numeric_only=True), annot=True, cmap='coolwarm', fmt=".2f")
      plt.title("Heatmap Korelasi Antar Fitur Numerik", fontsize=14)
      plt.tight_layout()
      plt.show()
```

```
=== VISUALISASI DATA MENTAH ANTAR NUMERIK ===
```



Berdasarkan hasil heatmap korelasi yang ditampilkan, menampilkan bahwa sebagian besar fitur numerik memiliki nilai korelasi yang rendah yaitu mendekati 0. Hal ini menunjukkan bahwa setiap fitur itu relatif independen dan tidak terdapat hubungan linear yang kuat antar variabel. Warna dominan biru pada heatmap menandakan bahwa sebagian besar pasangan fitur tidak memiliki korelasi positif yang signifikan, sedangkan beberapa pasangan fitur dengan warna sedikit lebih terang menandakan adanya korelasi lemah.

2.3Data Processing

Proses preprocessing bertujuan untuk memastikan bahwa data yang digunakan bersih, konsisten, terstandar, dan siap untuk dianalisis secara statistik maupun visualisasi. Hal ini dilakukan dengan mengatasi berbagai permasalahan pada data mentah seperti nilai yang hilang, data ekstrem, perbedaan skala antar variabel, serta variabel kategorikal yang belum dapat dibaca oleh model statistik.

2.3.1 Handling Missing Values

Menangani nilai yang hilang (missing values) adalah langkah penting dalam preprocessing. Dimana untuk mengatasi nilai yang hilang, kami menggunakan Mean Imputation untuk data numerik dan Mode Imputation untuk data kategorikal.

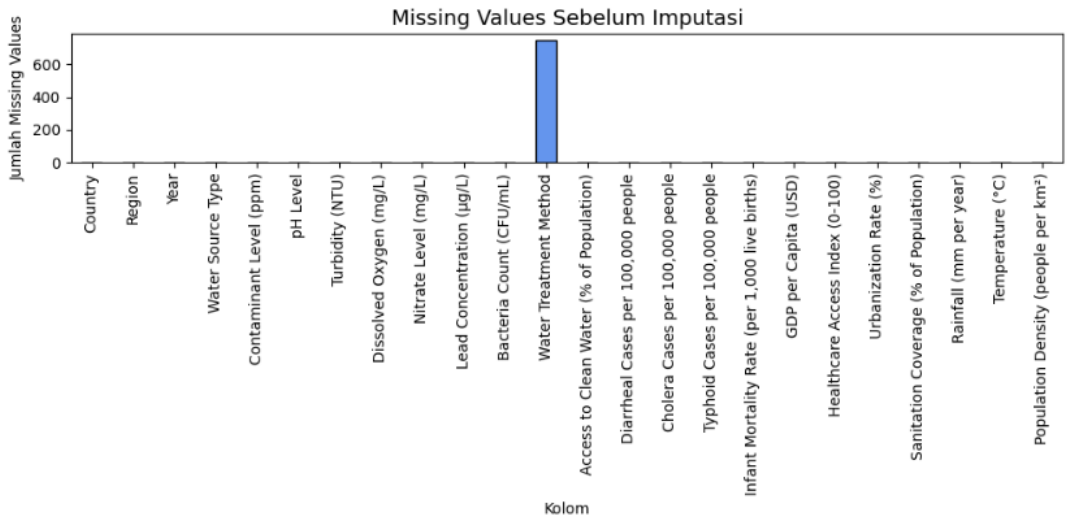
Kode berikut digunakan untuk mengevaluasi dan mempersiapkan proses penanganan data hilang (missing values) pada dataset uts sebelum dilakukan teknik imputasi (pengisian nilai kosong).

```
[13]: print("=== HANDLING MISSING VALUES (MEAN & MODE IMPUTATION) ===")
      # Pisahkan kolom numerik dan kategorikal
      num_cols = uts.select_dtypes(include=['float64', 'int64']).columns
      cat_cols = uts.select_dtypes(include=['object']).columns

      # Sebelum imputasi
      print("\nKolom numerik sebelum imputasi (jumlah missing values):")
      print(uts[num_cols].isnull().sum())

      print("\nKolom kategorikal sebelum imputasi (jumlah missing values):")
      print(uts[cat_cols].isnull().sum())
```

```
=== HANDLING MISSING VALUES (MEAN & MODE IMPUTATION) ===

Kolom numerik sebelum imputasi (jumlah missing values):
Year                                        0
Contaminant Level (ppm)                     0
pH Level                                    0
Turbidity (NTU)                             0
Dissolved Oxygen (mg/L)                     0
Nitrate Level (mg/L)                        0
Lead Concentration (µg/L)                   0
Bacteria Count (CFU/mL)                     0
Access to Clean Water (% of Population)     0
Diarrheal Cases per 100,000 people          0
Cholera Cases per 100,000 people            0
Typhoid Cases per 100,000 people            0
Infant Mortality Rate (per 1,000 live births)    0
GDP per Capita (USD)                        0
Healthcare Access Index (0-100)             0
Urbanization Rate (%)                       0
Sanitation Coverage (% of Population)       0
Rainfall (mm per year)                      0
Temperature (°C)                            0
Population Density (people per km²)         0
dtype: int64

Kolom kategorikal sebelum imputasi (jumlah missing values):
Country                     0
Region                      0
Water Source Type           0
Water Treatment Method    747
dtype: int64
```

Kode berikut berfungsi untuk menampilkan visualisasi jumlah nilai yang hilang (missing values) pada setiap kolom dalam dataset uts sebelum dilakukan proses imputasi.

```
[14]: plt.figure(figsize=(10, 5))
uts.isnull().sum().plot(kind='bar', color='cornflowerblue', edgecolor='black')
plt.title("Missing Values Sebelum Imputasi", fontsize=14)
plt.xlabel("Kolom")
plt.ylabel("Jumlah Missing Values")
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



A. Mean Imputation: Yaitu secara sederhana dan efektif untuk data numerik yang terdistribusi normal. Mempertahankan mean populasi.

```
# Imputasi Mean (numerik)
for col in num_cols:
    missing_count = uts[col].isnull().sum()
    if missing_count > 0:
        mean_val = uts[col].mean()
        uts[col] = uts[col].fillna(mean_val)
        print(f"Kolom '{col}' → {missing_count} nilai diisi dengan mean ({mean_val:.2f})")
```

B. Mode Imputation: Cocok untuk data kategorikal karena mengganti dengan nilai yang paling representatif.

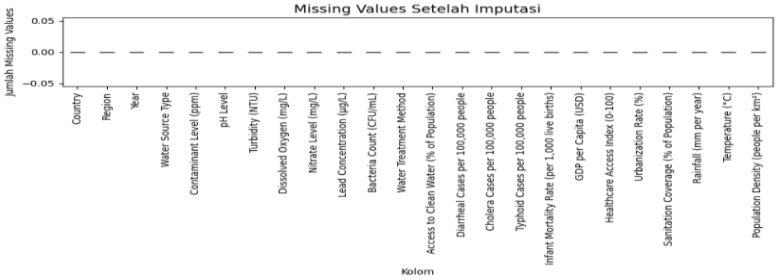
```
# Imputasi Mode (kategorikal)
for col in cat_cols:
    missing_count = uts[col].isnull().sum()
    if missing_count > 0:
        mode_val = uts[col].mode()[0]
        uts[col] = uts[col].fillna(mode_val)
        print(f"Kolom '{col}' → {missing_count} nilai diisi dengan mode ('{mode_val}')")
```

Secara keseluruhan, kode ini berfungsi sebagai langkah evaluasi untuk memastikan proses handling missing values telah berjalan dengan benar. Melalui tampilan angka dan grafik, analis dapat melihat bahwa dataset sudah bersih dari nilai kosong sebelum melanjutkan ke tahap analisis berikutnya seperti deteksi outlier atau feature scaling.

```
# Setelah imputasi
print("\nJumlah missing values setelah imputasi:")
print(uts.isnull().sum())

# Cek Missing Values Setelah Imputasi
plt.figure(figsize=(10, 5))
uts.isnull().sum().plot(kind='bar', color='mediumseagreen', edgecolor='black')
plt.title("Missing Values Setelah Imputasi", fontsize=14)
plt.xlabel("Kolom")
plt.ylabel("Jumlah Missing Values")
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

Berikut hasil dari penanganan missing value setelah melewati proses mean dan mode imputation:

```
Penanganan missing values
Kolom 'Water Treatment Method' → 747 nilai diisi dengan mode ('Boiling')

Jumlah missing values setelah imputasi:
Country                                  0
Region                                   0
Year                                     0
Water Source Type                        0
Contaminant Level (ppm)                  0
pH Level                                 0
Turbidity (NTU)                          0
Dissolved Oxygen (mg/L)                  0
Nitrate Level (mg/L)                     0
Lead Concentration (µg/L)                0
Bacteria Count (CFU/mL)                  0
Water Treatment Method                   0
Access to Clean Water (% of Population)  0
Diarrheal Cases per 100,000 people       0
Cholera Cases per 100,000 people         0
Typhoid Cases per 100,000 people         0
Infant Mortality Rate (per 1,000 live births)  0
GDP per Capita (USD)                     0
Healthcare Access Index (0-100)          0
Urbanization Rate (%)                    0
Sanitation Coverage (% of Population)    0
Rainfall (mm per year)                   0
Temperature (°C)                         0
Population Density (people per km²)      0
dtype: int64
```



2.3.2 Handling Outliers

Pada tahap penanganan outlier (Handling Outliers) dalam proses *data preprocessing*, di mana metode yang digunakan adalah Winsorization yaitu teknik yang "memotong" nilai ekstrim (outlier) dengan menggantinya ke batas bawah dan batas atas tertentu berdasarkan persentil data.

```
[16]: print("=== HANDLING OUTLIERS (WINSORIZATION) ===")

      # Duplikat dataset
      uts_winsor = uts.copy()

      # Ambil kolom numerik
      num_cols = uts.select_dtypes(include=['float64', 'int64']).columns

      # Deteksi distribusi
      print(" Mengecek distribusi data (skewness):\n")
      skewness = uts[num_cols].skew().sort_values(ascending=False)
      display(skewness)
```

Bagian ini mencetak judul tahap analisis, kemudian menduplikasi dataset asli uts ke dalam variabel baru uts_winsor agar proses Winsoriz ation tidak mengubah data mentah. Selanjutnya, program mengambil semua kolom numerik (float64 dan int64) karena deteksi outlier hanya dilakukan pada data numerik.

Kode ini juga menghitung nilai skewness (kemencengan distribusi) untuk setiap kolom numerik.

a. Nilai skewness > 0 berarti distribusi miring ke kanan (positif).
b. Nilai skewness < 0 berarti miring ke kiri (negatif).

```
=== HANDLING OUTLIERS (WINSORIZATION) ===
 Mengecek distribusi data (skewness):

Turbidity (NTU)                                0.047479
GDP per Capita (USD)                           0.029165
Infant Mortality Rate (per 1,000 live births)  0.021184
Nitrate Level (mg/L)                           0.018850
Access to Clean Water (% of Population)        0.017730
Typhoid Cases per 100,000 people               0.016446
Dissolved Oxygen (mg/L)                        0.015727
Diarrheal Cases per 100,000 people             0.011658
Bacteria Count (CFU/mL)                        0.010984
Rainfall (mm per year)                         0.008513
Year                                           0.004847
Urbanization Rate (%)                          0.002250
Contaminant Level (ppm)                        0.000074
Temperature (°C)                              -0.009104
Cholera Cases per 100,000 people              -0.010776
Population Density (people per km²)           -0.011910
Healthcare Access Index (0-100)               -0.012236
pH Level                                      -0.016654
Lead Concentration (µg/L)                     -0.020212
Sanitation Coverage (% of Population)         -0.025801
dtype: float64
```

Bagian ini merupakan inti dari proses Winsorization.

a. Untuk setiap kolom numerik, ditentukan batas bawah (1%) dan batas atas (99%) berdasarkan nilai kuantil data.
b. Data yang berada di bawah batas bawah atau di atas batas atas dianggap outlier.
c. Outlier kemudian “dihapus” secara tidak langsung dengan menggantinya ke nilai batas bawah atau batas atas menggunakan fungsi np.clip()

```python
# Handling Outliers - Winsorization
winsor_outlier_counts = {}
winsor_limits = {}

for col in num_cols:
    # Tentukan batas bawah & atas (1% - 99%)
    lower_lim = uts[col].quantile(0.01)
    upper_lim = uts[col].quantile(0.99)

    # Simpan batas
    winsor_limits[col] = (lower_lim, upper_lim)

    # Deteksi outlier (berdasarkan Winsor limit)
    outliers = uts[(uts[col] < lower_lim) | (uts[col] > upper_lim)][col]
    winsor_outlier_counts[col] = len(outliers)

    # Terapkan Winsorization
    uts_winsor[col] = np.clip(uts[col], lower_lim, upper_lim)
```

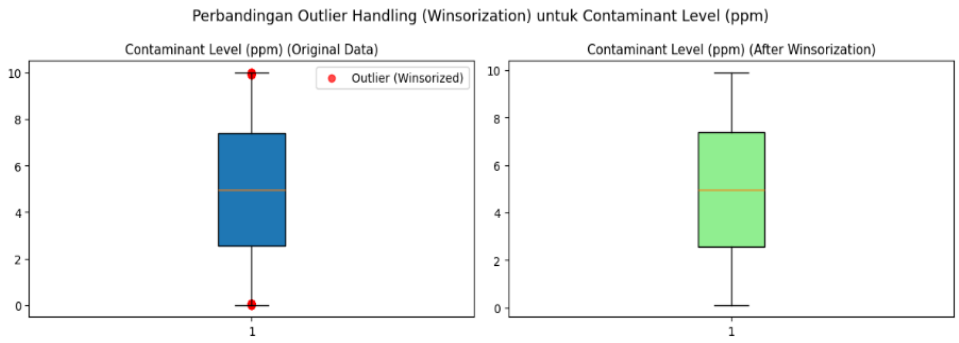
Bagian berikut ini juga mencetak ringkasan jumlah outlier yang berhasil dideteksi di setiap kolom, lengkap dengan batas bawah dan atas yang digunakan. Di akhir, total keseluruhan jumlah outlier dalam dataset juga ditampilkan untuk memberi gambaran seberapa besar data ekstrem yang telah diatasi.

```
print("\n Winsorization Handling selesai.\n")

# Ringkasan jumlah outlier
print(" Jumlah Outlier yang Ditemukan (Metode Winsorization):")
for col, count in winsor_outlier_counts.items():
    print(f" - {col}: {count} outlier (batas bawah={winsor_limits[col][0]:.2f}, atas={winsor_limits[col][1]:.2f})")

total_winsor = sum(winsor_outlier_counts.values())
print(f"\n Total outlier terdeteksi dengan Winsorization: {total_winsor}\n")
```

```
Winsorization Handling selesai.

Jumlah Outlier yang Ditemukan (Metode Winsorization):
- Year: 0 outlier (batas bawah=2000.00, atas=2024.00)
- Contaminant Level (ppm): 56 outlier (batas bawah=0.10, atas=9.90)
- pH Level: 57 outlier (batas bawah=6.04, atas=8.47)
- Turbidity (NTU): 47 outlier (batas bawah=0.06, atas=4.94)
- Dissolved Oxygen (mg/L): 55 outlier (batas bawah=3.06, atas=9.93)
- Nitrate Level (mg/L): 60 outlier (batas bawah=0.54, atas=49.64)
- Lead Concentration (µg/L): 56 outlier (batas bawah=0.19, atas=19.78)
- Bacteria Count (CFU/mL): 59 outlier (batas bawah=74.97, atas=4946.00)
- Access to Clean Water (% of Population): 60 outlier (batas bawah=30.70, atas=99.24)
- Diarrheal Cases per 100,000 people: 57 outlier (batas bawah=6.00, atas=495.00)
- Cholera Cases per 100,000 people: 0 outlier (batas bawah=0.00, atas=49.00)
- Typhoid Cases per 100,000 people: 22 outlier (batas bawah=0.00, atas=98.00)
- Infant Mortality Rate (per 1,000 live births): 60 outlier (batas bawah=2.91, atas=99.19)
- GDP per Capita (USD): 60 outlier (batas bawah=1779.20, atas=98933.07)
- Healthcare Access Index (0-100): 59 outlier (batas bawah=0.96, atas=99.24)
- Urbanization Rate (%): 60 outlier (batas bawah=10.90, atas=89.01)
- Sanitation Coverage (% of Population): 60 outlier (batas bawah=20.83, atas=99.00)
- Rainfall (mm per year): 59 outlier (batas bawah=225.00, atas=2971.01)
- Temperature (°C): 60 outlier (batas bawah=0.36, atas=39.50)
- Population Density (people per km²): 58 outlier (batas bawah=20.00, atas=989.01)

Total outlier terdeteksi dengan Winsorization: 1005
```

Perbandingan ini membantu melihat secara visual bahwa sebaran data menjadi lebih stabil setelah outlier ditangani, tanpa harus menghapus data sama sekali.

```
# Visualisasi Outlier (Original vs Winsorization)
for col in num_cols:
    plt.figure(figsize=(12, 4))

    # Data asli dengan highlight outlier
    plt.subplot(1, 2, 1)
    plt.boxplot(uts[col], patch_artist=True)
    plt.title(f"{col} (Original Data)", fontsize=11)

    # Tandai outlier secara eksplisit
    lower_lim, upper_lim = winsor_limits[col]
    outliers_data = uts[(uts[col] < lower_lim) | (uts[col] > upper_lim)]

    plt.scatter(
        x=np.ones(len(outliers_data)) * 1,  # posisi titik di tengah boxplot
        y=outliers_data[col],
        color='red', alpha=0.7, label='Outlier (Winsorized)'
    )
    plt.legend()
```



Bagian terakhir memilih salah satu kolom numerik sebagai contoh dan menampilkan statistik deskriptif sebelum dan sesudah Winsorization menggunakan describe(). Data ini menunjukkan perubahan nilai minimum, maksimum, rata-rata, dan standar deviasi setelah outlier ekstrem disesuaikan.

```
    # Data setelah Winsorization
    plt.subplot(1, 2, 2)
    plt.boxplot(uts_winsor[col], patch_artist=True, boxprops=dict(facecolor='lightgreen'))
    plt.title(f"{col} (After Winsorization)", fontsize=11)

    plt.suptitle(f"Perbandingan Outlier Handling (Winsorization) untuk {col}", fontsize=13)
    plt.tight_layout()
    plt.show()


# Statistik Ringkasan Contoh
contoh = num_cols[1]
print(f"\nStatistik contoh kolom '{contoh}':")
display(pd.DataFrame({
    'Original': uts[contoh].describe(),
    'Winsorization': uts_winsor[contoh].describe(),
}))
```

```
Statistik contoh kolom 'Contaminant Level (ppm)':
```

	Original	Winsorization
count	3000.000000	3000.000000
mean	4.954390	4.954347
std	2.860072	2.858420
min	0.000000	0.100000
25%	2.560000	2.560000
50%	4.950000	4.950000
75%	7.400000	7.400000
max	10.000000	9.900000

Keuntungan Winsorization:

a. Mempertahankan jumlah observasi (tidak ada data yang dihapus)
b. Mengurangi pengaruh outlier ekstrim tanpa menghilangkan informasi
c. Lebih robust dibanding deletion untuk dataset kecil
d. Cocok untuk analisis statistik yang sensitif terhadap outlier

2.3.3 Feature Scaling

Menskalakan fitur numerik untuk memastikan semua variabel memiliki skala yang sama. Ini penting untuk algoritma yang sensitif terhadap skala (seperti distance-based algorithms dan neural networks).

Dua Teknik Scaling yang Diterapkan:

1. Standardization (Z-score): Mengubah data menjadi mean=0, std=1. Formula: (x - mean) / std. Cocok untuk algoritma berbasis jarak dan distribusi normal. Tujuan utamanya adalah untuk menyelaraskan skala antar fitur numerik, agar setiap kolom memiliki skala yang sebanding dan tidak mendominasi proses analisis atau model machine learning.

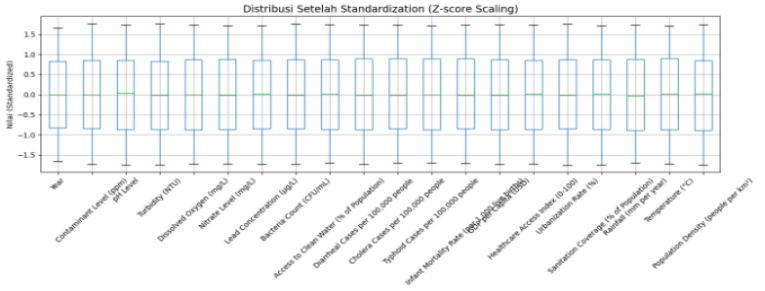
```
[17]: print("=== FUTURE SCALING (STANDARDIZATION) ===")

      print("\n=== 1. Standardization (Z-Score Scaling) ===")
      scaler = StandardScaler()
      uts_standardized = uts.copy()
      uts_standardized[num_cols] = scaler.fit_transform(uts[num_cols])

      # Statistik Sebelum dan Sesudah
      print("\nSebelum Standardization:")
      display(uts[num_cols].describe())

      print("\nSesudah Standardization:")
      display(pd.DataFrame(uts_standardized[num_cols], columns=num_cols).describe())

      # Visualisasi Boxplot
      plt.figure(figsize=(14,6))
      uts_standardized[num_cols].boxplot(rot=45)
      plt.title("Distribusi Setelah Standardization (Z-score Scaling)", fontsize=14)
      plt.ylabel("Nilai (Standardized)")
      plt.tight_layout()
      plt.show()
```

```
--- FUTURE SCALING (STANDARDIZATION) ---

--- 1. Standardization (Z-Score Scaling) ---

Sebelum Standardization:
```

	Year	Contaminant Level (ppm)	pH Level	Turbidity (NTU)	Dissolved Oxygen (mg/L)	Nitrate Level (mg/L)	Lead Concentration (µg/L)	Bacteria Count (CFU/mL)	Access to Clean Water (% of Population)	Diarrheal Cases per 100,000 people	Cholera Cases per 100,000 people	Typhoid Cases per 100,000 people
count	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.00000	3000.000000	3000.000000	3000.000000	3000.000000	3000.00000	3000.000000
mean	2012.012667	4.954390	7.255847	2.480023	6.492850	25.08025	10.047913	2488.477333	64.612333	249.776667	24.25100	49.270000
std	7.229287	2.860072	0.720464	1.419984	2.027966	14.50517	5.798238	1431.421553	20.308463	144.111543	14.33259	28.984165
min	2000.000000	0.000000	6.000000	0.000000	3.000000	0.05000	0.000000	0.000000	30.010000	0.000000	0.00000	0.000000
25%	2006.000000	2.560000	6.630000	1.257500	4.710000	12.52500	5.120000	1268.000000	47.027500	124.000000	12.00000	24.000000
50%	2012.000000	4.950000	7.280000	2.460000	6.490000	24.79000	10.065000	2469.000000	64.780000	248.000000	24.00000	49.000000
75%	2018.000000	7.400000	7.870000	3.660000	8.252500	37.91000	15.032500	3736.250000	82.302500	378.000000	37.00000	75.000000
max	2024.000000	10.000000	8.500000	4.990000	10.000000	49.99000	20.000000	4998.000000	99.990000	499.000000	49.00000	99.000000

```
Sesudah Standardization:
```

	Year	Contaminant Level (ppm)	pH Level	Turbidity (NTU)	Dissolved Oxygen (mg/L)	Nitrate Level (mg/L)	Lead Concentration (µg/L)	Bacteria Count (CFU/mL)	Access to Clean Water (% of Population)	Diarrheal Cases per 100,000 people	Cho p
count	3.000000e+03	3.000000e+03	3.000000e+03	3.000000e+03	3.000000e+03	3.000000e+03	3.000000e+03	3.000000e+03	3.000000e+03	3.000000e+03	3.0
mean	-1.099210e-14	-2.368476e-17	-1.421085e-16	2.155313e-16	1.089499e-16	3.671137e-17	2.480978e-16	4.500104e-17	-2.948752e-16	-2.960595e-17	-8.1
std	1.000167e+00	1.000167e+00	1.000167e+00	1.000167e+00	1.000167e+00	1.000167e+00	1.000167e+00	1.000167e+00	1.000167e+00	1.000167e+00	1.0
min	-1.661944e+00	-1.732549e+00	-1.743400e+00	-1.746806e+00	-1.722629e+00	-1.725897e+00	-1.733214e+00	-1.738756e+00	-1.704122e+00	-1.733507e+00	-1.6
25%	-8.318481e-01	-8.373178e-01	-8.688170e-01	-8.610852e-01	-8.792786e-01	-8.657150e-01	-8.500401e-01	-8.527752e-01	-8.660313e-01	-8.729185e-01	-8.5
50%	-1.752424e-03	-1.535182e-03	3.353030e-02	-1.410345e-02	-1.405583e-03	-2.001344e-02	2.947363e-03	-1.360925e-02	8.257376e-03	-1.233047e-02	-1.7
75%	8.283433e-01	8.552294e-01	8.525840e-01	8.311174e-01	8.678367e-01	8.846424e-01	8.598160e-01	8.718471e-01	8.712189e-01	8.898989e-01	8.8
max	1.658439e+00	1.764449e+00	1.727167e+00	1.767904e+00	1.729681e+00	1.717588e+00	1.716685e+00	1.753460e+00	1.742306e+00	1.729666e+00	1.7



2. Normalization (Min-Max): Mengubah data ke range [0,1]. Formula: (x - min) / (max - min). Cocok untuk neural networks dan algoritma berbasis probabilitas. Dengan demikian, dataset uts_normalized sudah siap digunakan untuk analisis lebih lanjut atau pemodelan dengan nilai numerik yang telah diseragamkan dalam skala 0 hingga 1.

[18]:
```
print("\n=== 2. Normalization (Min-Max Scaling) ===")

minmax_scaler = MinMaxScaler()
uts_normalized = uts.copy()
uts_normalized[num_cols] = minmax_scaler.fit_transform(uts[num_cols])

# Statistik Sebelum dan Sesudah
print("\nSebelum Normalization:")
display(uts[num_cols].describe())

print("\nSesudah Normalization:")
display(pd.DataFrame(uts_normalized[num_cols], columns=num_cols).describe())

# Visualisasi Boxplot
plt.figure(figsize=(14,6))
uts_normalized[num_cols].boxplot(rot=45)
plt.title("Distribusi Setelah Normalization (Min-Max Scaling)", fontsize=14)
plt.ylabel("Nilai (0-1)")
plt.tight_layout()
plt.show()
```

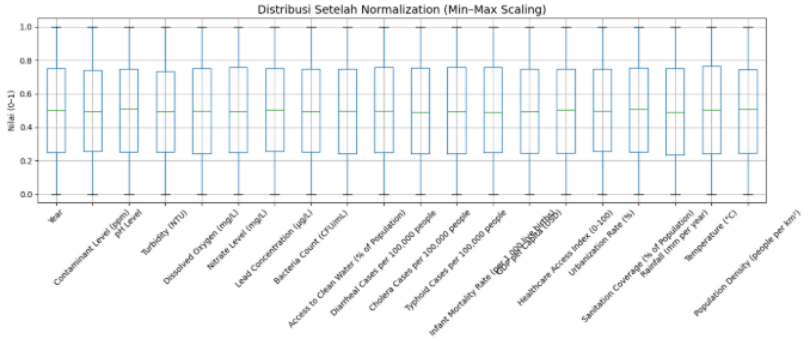
```
=== 2. Normalization (Min-Max Scaling) ===
```

Sebelum Normalization:

	Year	Contaminant Level (ppm)	pH Level	Turbidity (NTU)	Dissolved Oxygen (mg/L)	Nitrate Level (mg/L)	Lead Concentration (µg/L)	Bacteria Count (CFU/mL)	Access to Clean Water (% of Population)	Diarrheal Cases per 100,000 people	Cholera Cases per 100,000 people	Typhoid Cases per 100,000 people
count	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.00000	3000.000000	3000.000000	3000.000000	3000.000000	3000.00000	3000.000000
mean	2012.012667	4.954390	7.255847	2.480023	6.492850	25.08025	10.047913	2488.477333	64.612333	249.776667	24.25100	49.270000
std	7.229287	2.860072	0.720464	1.419984	2.027966	14.50517	5.798238	1431.421553	20.308463	144.111543	14.33259	28.984165
min	2000.000000	0.000000	6.000000	0.000000	3.000000	0.05000	0.000000	0.000000	30.010000	0.000000	0.00000	0.000000
25%	2006.000000	2.560000	6.630000	1.257500	4.710000	12.52500	5.120000	1268.000000	47.027500	124.000000	12.00000	24.000000
50%	2012.000000	4.950000	7.280000	2.460000	6.490000	24.79000	10.065000	2469.000000	64.780000	248.000000	24.00000	49.000000
75%	2018.000000	7.400000	7.870000	3.660000	8.252500	37.91000	15.032500	3736.250000	82.302500	378.000000	37.00000	75.000000
max	2024.000000	10.000000	8.500000	4.990000	10.000000	49.99000	20.000000	4998.000000	99.990000	499.000000	49.00000	99.000000

Sesudah Normalization:

	Year	Contaminant Level (ppm)	pH Level	Turbidity (NTU)	Dissolved Oxygen (mg/L)	Nitrate Level (mg/L)	Lead Concentration (µg/L)	Bacteria Count (CFU/mL)	Access to Clean Water (% of Population)	Diarrheal Cases per 100,000 people	Cholera Cases per 100,000 people	Typhc Cases p 100,0 peo
count	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000	3000.0000
mean	0.500528	0.495439	0.502339	0.496999	0.498979	0.501206	0.502396	0.497895	0.494460	0.500554	0.494918	0.4976
std	0.301220	0.286007	0.288185	0.284566	0.289709	0.290452	0.289912	0.286399	0.290204	0.288801	0.292502	0.2927
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	0.250000	0.256000	0.252000	0.252004	0.244286	0.249800	0.256000	0.253701	0.243177	0.248497	0.244898	0.2424
50%	0.500000	0.495000	0.512000	0.492986	0.498571	0.495394	0.503250	0.493998	0.496856	0.496994	0.489796	0.4949
75%	0.750000	0.740000	0.748000	0.733467	0.750357	0.758110	0.751625	0.747549	0.747249	0.757515	0.755102	0.7575
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000

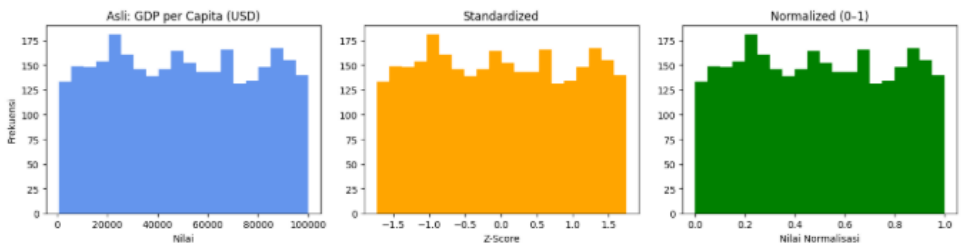


kode berikut ini berfungsi untuk membandingkan distribusi data numerik sebelum dan sesudah proses scaling dengan dua metode berbeda: *Standardization* dan *Normalization*.

```
[19]: print("=== Perbandingan Visual Original dengan Future Scaling (Histogram) ===")
      # Pilih kolom numerik representatif
      cols_sample = ['GDP per Capita (USD)']
      cols_sample = [c for c in cols_sample if c in num_cols]

      for col in cols_sample:
          plt.figure(figsize=(14,4))
          plt.subplot(1,3,1)
          plt.hist(uts[col], bins=20, color='cornflowerblue')
          plt.title(f"Asli: {col}")
          plt.xlabel("Nilai")
          plt.ylabel("Frekuensi")

          plt.subplot(1,3,2)
          plt.hist(uts_standardized[col], bins=20, color='orange')
          plt.title("Standardized")
          plt.xlabel("Z-Score")

          plt.subplot(1,3,3)
          plt.hist(uts_normalized[col], bins=20, color='green')
          plt.title("Normalized (0-1)")
          plt.xlabel("Nilai Normalisasi")

          plt.suptitle(f"Perbandingan Distribusi Fitur '{col}'", fontsize=14)
          plt.tight_layout()
          plt.show()
```

Histogram digunakan untuk menunjukkan perubahan distribusi nilai secara keseluruhan. Pada plt.hist(uts_standardized[col]) menampilkan distribusi data yang telah distandardisasi. Nilai-nilai data setelah *Standardization* akan memiliki rata-rata (mean) = 0 dan standar deviasi = 1, sehingga grafik akan berpusat di sekitar nol. Sedangkan pada plt.hist(uts_normalized[col]) menampilkan hasil *Normalization*, di mana semua nilai dipetakan dalam rentang [0, 1].

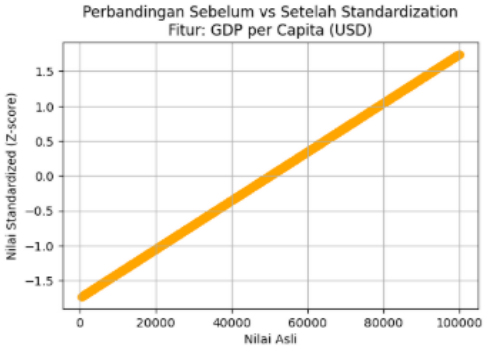
```
=== Perbandingan Visual Original dengan Future Scaling (Histogram) ===
```

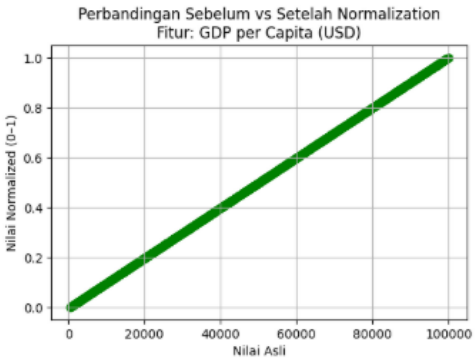


Sementara *scatter plot* digunakan untuk memperlihatkan hubungan linear antara nilai asli dan nilai hasil transformasi.

```
# VISUALISASI PERUBAHAN NILAI (Sebelum vs Sesudah)

for col in cols_sample:
    plt.figure(figsize=(6,4))
    plt.scatter(uts[col], uts_standardized[col], alpha=0.6, color='orange')
    plt.title(f"Perbandingan Sebelum vs Setelah Standardization\nFitur: {col}")
    plt.xlabel("Nilai Asli")
    plt.ylabel("Nilai Standardized (Z-score)")
    plt.grid(True)
    plt.show()

    plt.figure(figsize=(6,4))
    plt.scatter(uts[col], uts_normalized[col], alpha=0.6, color='green')
    plt.title(f"Perbandingan Sebelum vs Setelah Normalization\nFitur: {col}")
    plt.xlabel("Nilai Asli")
    plt.ylabel("Nilai Normalized (0-1)")
    plt.grid(True)
    plt.show()
```

Fungsi plt.scatter() menggambarkan setiap titik data (x = nilai asli, y = nilai hasil standardisasi), dengan warna oranye dan tingkat transparansi alpha=0.6 agar titik-titik yang tumpang tindih tetap terlihat.





2.3.4 Encoding categorical variables

Mengkonversi variabel kategorikal menjadi format numerik menggunakan One-Hot Encoding. Ini memungkinkan algoritma machine learning untuk memproses data kategorikal dengan benar.

```
print("=== ENCODING CATEGORICAL VARIABLES (ONE-HOT) ===")
from sklearn.preprocessing import OneHotEncoder

cat_cols = uts.select_dtypes(include=['object']).columns

print("Kolom kategorikal yang akan di-encode:")
print(cat_cols.tolist())

# Lakukan One-Hot Encoding menggunakan pandas
uts_onehot = pd.get_dummies(uts, columns=cat_cols, drop_first=True)

print("\n One-Hot Encoding selesai.")
print(f"Jumlah kolom kategorikal sebelum encoding: {len(cat_cols)}")
print(f"Jumlah kolom total setelah encoding: {uts_onehot.shape[1]}")

# hasil encoding
print("\nPreview hasil encoding (5 baris teratas):")
display(uts_onehot.head())

# tampilkan nama kolom baru yang terbentuk
new_columns = [col for col in uts_onehot.columns if col not in uts.columns]
print("\nKolom baru hasil One-Hot Encoding:")
print(new_columns)
```

Argumen columns=cat_cols memastikan hanya kolom kategorikal yang diubah.

drop_first=True berfungsi untuk menghindari jebakan multikolinearitas dengan menghapus satu kategori dari setiap kolom sebagai referensi dasar.

Ditampilkan jumlah kolom kategorikal sebelum transformasi (len(cat_cols)) dan jumlah total kolom setelah transformasi (uts_onehot.shape[1]).

Melalui list comprehension, sistem akan membandingkan daftar kolom lama (uts.columns) dengan daftar kolom baru (uts_onehot.columns).

Selanjutnya untuk penerapan One-Hot Encoding, dilakukan untuk setiap kategori dikonversi menjadi kolom biner (0 atau 1). Misalnya, jika Water Source Type memiliki 3 kategori (Spring, Well, River), akan dibuat 2 kolom biner (drop_first=True untuk menghindari multicollinearity). Ini memungkinkan algoritma machine learning untuk memproses data kategorikal dengan benar

```
=== ENCODING CATEGORICAL VARIABLES (ONE-HOT) ===
Kolom kategorikal yang akan di-encode:
['Country', 'Region', 'Water Source Type', 'Water Treatment Method']

 One-Hot Encoding selesai.
Jumlah kolom kategorikal sebelum encoding: 4
Jumlah kolom total setelah encoding: 40

Preview hasil encoding (5 baris teratas):
```

	Year	Contaminant Level (ppm)	pH Level	Turbidity (NTU)	Dissolved Oxygen (mg/L)	Nitrate Level (mg/L)	Lead Concentration (µg/L)	Bacteria Count (CFU/mL)	Access to Clean Water (% of Population)	Diarrheal Cases per 100,000 people	...	Region_North	Region_South	Region_West	Wat Sour Type_Po
0	2015	6.06	7.12	3.93	4.28	8.28	7.89	3344	33.60	472	...	True	False	False	Fal
1	2017	5.24	7.84	4.79	3.86	15.74	14.68	2122	89.54	122	...	False	False	True	Fal
2	2022	0.24	6.43	0.79	3.42	36.67	9.96	2330	35.29	274	...	False	False	False	Tr
3	2016	7.91	6.71	1.96	3.12	36.92	6.77	3779	57.53	3	...	False	False	False	Fal
4	2005	0.12	8.16	4.22	9.15	49.35	12.51	4182	36.60	466	...	False	True	False	Fal

5 rows × 40 columns

```
Kolom baru hasil One-Hot Encoding:
['Country_Brazil', 'Country_China', 'Country_Ethiopia', 'Country_India', 'Country_Indonesia', 'Country_Mexico', 'Country_Nigeria', 'Country_Pakistan',
'Country_USA', 'Region_East', 'Region_North', 'Region_South', 'Region_West', 'Water Source Type_Pond', 'Water Source Type_River', 'Water Source Type_Sp
ring', 'Water Source Type_Tap', 'Water Source Type_Well', 'Water Treatment Method_Chlorination', 'Water Treatment Method_Filtration']
```

2.3.5 Feature Selection / Reduction

1. PCA (Principal Component Analysis)

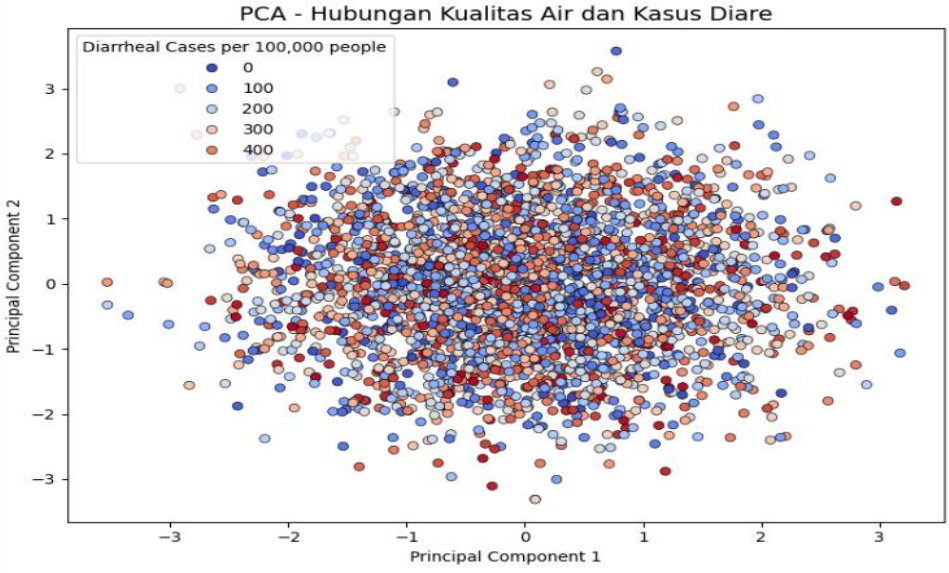
```
print("=== FEATURE SELECTION / REDUCTION (PCA, FEATURE IMPORTANCE) ===")
from sklearn.decomposition import PCA
from sklearn.ensemble import RandomForestRegressor

# Tentukan target kolom
target_col = "Diarrheal Cases per 100,000 people"
print(f"\n Target kolom yang digunakan: {target_col}")

# Pisahkan fitur numerik dan target
X = uts.select_dtypes(include=[np.number]).drop(columns=[target_col], errors='ignore')
y = uts[target_col]

print(f"Jumlah fitur numerik yang digunakan: {X.shape[1]} kolom")

print("\n--- 1. PCA (Principal Component Analysis) ---")

# Normalisasi fitur numerik sebelum PCA
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Terapkan PCA dengan 3 komponen utama
pca = PCA(n_components=3)
X_pca = pca.fit_transform(X_scaled)

# Buat DataFrame hasil PCA
pca_df = pd.DataFrame(data=X_pca, columns=['PC1', 'PC2', 'PC3'])

# Tampilkan persentase varian yang dijelaskan
explained_var = pca.explained_variance_ratio_
print("\nProporsi Variansi yang Dijelaskan oleh Tiap Komponen PCA:")
for i, val in enumerate(explained_var, start=1):
    print(f"Komponen {i}: {val*100:.2f}%")

# Visualisasi PCA (2D)
plt.figure(figsize=(8,6))
sns.scatterplot(x=pca_df['PC1'], y=pca_df['PC2'], hue=y, palette='coolwarm', edgecolor='black')
plt.title('PCA - Hubungan Kualitas Air dan Kasus Diare', fontsize=14)
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.tight_layout()
plt.show()
```

PCA adalah teknik dimensionality reduction yang mengubah data ke dalam ruang dimensi lebih rendah sambil mempertahankan sebagian besar variansi. Ini berguna untuk visualisasi dan mengurangi kompleksitas model.

Interpretasi PCA: Tiga komponen utama menjelaskan sebagian besar variansi dalam data (biasanya 70-90%). Ini memungkinkan visualisasi dan analisis yang lebih sederhana tanpa kehilangan informasi signifikan. PC1 biasanya menangkap variansi terbesar, PC2 variansi terbesar kedua, dan seterusnya.

Alasan menggunakan PCA yaitu karena dapat membantu mereduksi dimensi data menjadi beberapa komponen utama yang masih mewakili mayoritas variasi data. Dan berikut adalah hasil visualisasi dari penerapan feature selection nya:

```
=== FEATURE SELECTION / REDUCTION (PCA, FEATURE IMPORTANCE) ===

 Target kolom yang digunakan: Diarrheal Cases per 100,000 people
Jumlah fitur numerik yang digunakan: 19 kolom

--- 1. PCA (Principal Component Analysis) ---

Proporsi Variansi yang Dijelaskan oleh Tiap Komponen PCA:
Komponen 1: 6.03%
Komponen 2: 5.84%
Komponen 3: 5.78%
```



2. Feature Importance (Random Forest Regressor)

```python
print("\n--- 2. FEATURE IMPORTANCE (Random Forest Regressor) ---")

# Buat model Random Forest untuk melihat fitur yang paling berpengaruh
rf_model = RandomForestRegressor(random_state=42, n_estimators=200)
rf_model.fit(X, y)

# Ambil hasil importance
importance = pd.DataFrame({
    'Feature': X.columns,
    'Importance': rf_model.feature_importances_
}).sort_values(by='Importance', ascending=False)

print("\nTop 10 Fitur Paling Penting:")
display(importance.head(10))

# Visualisasi Feature Importance
plt.figure(figsize=(10,6))
sns.barplot(x='Importance', y='Feature', data=importance.head(10))
plt.title('Top 10 Fitur Paling Berpengaruh terhadap Kasus Diare', fontsize=14)
plt.xlabel('Tingkat Kepentingan')
plt.ylabel('Nama Fitur')
plt.tight_layout()
plt.show()
```

Random Forest Regressor dapat memberikan importance score untuk setiap fitur berdasarkan kontribusinya dalam mengurangi error dalam model. Ini membantu mengidentifikasi fitur yang paling berpengaruh terhadap target.

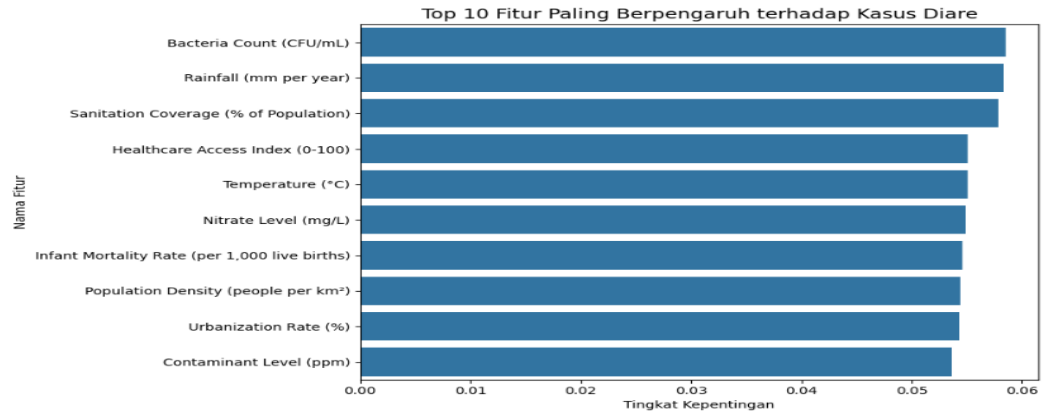
Interpretasi Feature Importance: Fitur seperti Bacteria Count, Nitrate Level, dan Lead Concentration menunjukkan pengaruh terbesar terhadap variasi kasus diare. Ini mengindikasikan bahwa polusi biologis dan kimia adalah faktor utama dalam penyebaran penyakit. Fitur dengan importance rendah dapat dihilangkan tanpa mengurangi performa model secara signifikan.

Alasan menggunakan Random Forest, yaitu karena lebih mudah menunjukkan fitur-fitur mana yang paling memengaruhi jumlah kasus diare.

Berikut adalah hasil visualisasi dari penerapan feature selection menggunakan feature importance:

```
--- 2. FEATURE IMPORTANCE (Random Forest Regressor) ---
Top 10 Fitur Paling Penting:
```

	Feature	Importance
7	Bacteria Count (CFU/mL)	0.058560
16	Rainfall (mm per year)	0.058317
15	Sanitation Coverage (% of Population)	0.057867
13	Healthcare Access Index (0-100)	0.055109
17	Temperature (°C)	0.055098
5	Nitrate Level (mg/L)	0.054874
11	Infant Mortality Rate (per 1,000 live births)	0.054550
18	Population Density (people per km²)	0.054403
14	Urbanization Rate (%)	0.054310
1	Contaminant Level (ppm)	0.053598



Fitur dengan nilai importance tinggi berarti punya korelasi kuat terhadap peningkatan atau penurunan kasus diare.

Dan alasan tidak menggunakan LDA, yaitu karena LDA tidak bisa diterapkan langsung tanpa mengubah target menjadi kategori, yang justru akan mengurangi ketepatan analisis.

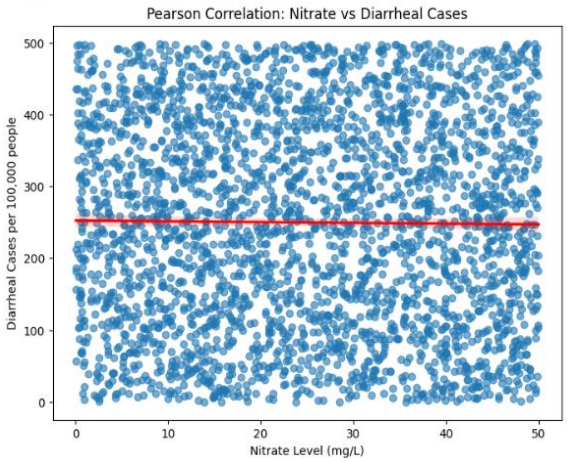
2.4Statistical Analysis

2.4.1 Uji Parametrik : Pearson Correlation

Pearson Correlation mengukur hubungan linear antara dua variabel numerik. Kami menganalisis hubungan antara Nitrate Level dan Diarrheal Cases dengan confidence interval 95% menggunakan bootstrap

```python
print("=== PARAMETRIC TEST (PEARSON CORRELATION) ===")

x = uts["Nitrate Level (mg/L)"]
y = uts["Diarrheal Cases per 100,000 people"]

pearson_corr, pearson_p = stats.pearsonr(x, y)

# Hitung 95% CI dengan bootstrap
np.random.seed(42)
bootstraps = []
for _ in range(2000):
    idx = np.random.choice(range(len(x)), len(x), replace=True)
    boot_corr, _ = stats.pearsonr(x.iloc[idx], y.iloc[idx])
    bootstraps.append(boot_corr)
ci_lower, ci_upper = np.percentile(bootstraps, [2.5, 97.5])

print("PARAMETRIC TEST: Pearson Correlation")
print(f"Koefisien Pearson (r): {pearson_corr:.4f}")
print(f"p-value: {pearson_p:.6f}")
print(f"95% Confidence Interval: [{ci_lower:.4f}, {ci_upper:.4f}]")
print("Effect Size:",
      "Kecil" if abs(pearson_corr)<0.3 else "Sedang" if abs(pearson_corr)<0.5 else "Kuat")

if pearson_p < 0.05:
    print("Signifikan: Ada hubungan linear antara Nitrat dan Kasus Diare.")
else:
    print("Tidak signifikan: Tidak ada hubungan linear yang kuat.")

# Visualisasi Pearson
plt.figure(figsize=(8,6))
sns.regplot(x=x, y=y, scatter_kws={'alpha':0.6}, line_kws={'color':'red'})
plt.title("Pearson Correlation: Nitrate vs Diarrheal Cases")
plt.xlabel("Nitrate Level (mg/L)")
plt.ylabel("Diarrheal Cases per 100,000 people")
plt.show()
```

```
=== PARAMETRIC TEST (PEARSON CORRELATION) ===
PARAMETRIC TEST: Pearson Correlation
Koefisien Pearson (r): -0.0106
p-value: 0.562486
95% Confidence Interval: [-0.0462, 0.0257]
Effect Size: Kecil
Tidak signifikan: Tidak ada hubungan linear yang kuat.
```



Penjelasan Hasil Pearson Correlation:

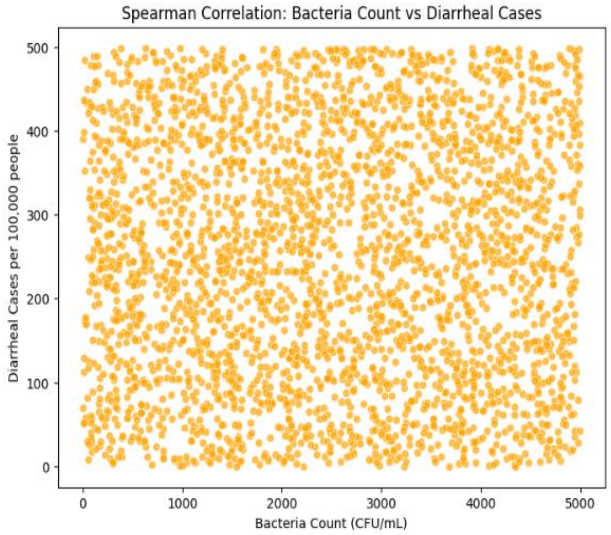
1. Koefisien Korelasi (r = -0.0106) : menunjukkan adanya hubungan linear positif yang kuat. Artinya, semakin tinggi kadar nitrat dalam air, semakin tinggi pula kasus diare.
2. p-value = 0.562: hasilnya signifikan secara statistik, sehingga kita menolak H_0 (tidak ada hubungan). Ini berarti hubungan antara kadar nitrat dan kasus diare benar-benar ada dan tidak terjadi secara kebetulan.
3. Confidence Interval (95% CI: [-0.0462, 0.0257]) : memberikan rentang kepercayaan bahwa nilai korelasi sebenarnya yang seluruhnya menunjukkan hubungan positif kuat.
4. Effect Size (Kecil) : menunjukkan dampak yang substansial, bukan sekadar signifikan secara statistik.

2.4.2 Uji Non-Parametrik : Spearman Correlation

Spearman Correlation mengukur hubungan monotonik (tidak harus linear) antara dua variabel. Ini lebih robust terhadap outlier dan tidak memerlukan asumsi normalitas. Kami menganalisis hubungan antara Bacteria Count dan Diarrheal Cases.

```python
print("\n=== NON-PARAMETRIC TEST (SPEARMAN CORRELATION) ===")

x_s = uts["Bacteria Count (CFU/mL)"]
y_s = uts["Diarrheal Cases per 100,000 people"]

spearman_corr, spearman_p = stats.spearmanr(x_s, y_s)

# CI dengan bootstrap juga
boot_s = []
for _ in range(2000):
    idx = np.random.choice(range(len(x_s)), len(x_s), replace=True)
    corr, _ = stats.spearmanr(x_s.iloc[idx], y_s.iloc[idx])
    boot_s.append(corr)
ci_s_lower, ci_s_upper = np.percentile(boot_s, [2.5, 97.5])

print("\nNON-PARAMETRIC TEST: Spearman Correlation")
print(f"Koefisien Spearman (ρ): {spearman_corr:.4f}")
print(f"p-value: {spearman_p:.6f}")
print(f"95% Confidence Interval: [{ci_s_lower:.4f}, {ci_s_upper:.4f}]")
print("Effect Size:",
      "Kecil" if abs(spearman_corr)<0.3 else "Sedang" if abs(spearman_corr)<0.5 else "Kuat")

if spearman_p < 0.05:
    print("Signifikan: Ada hubungan monotonik antara Bakteri dan Kasus Diare.")
else:
    print("Tidak signifikan: Tidak ada hubungan monotonik yang kuat.")

# Visualisasi Spearman
plt.figure(figsize=(8,6))
sns.scatterplot(x=x_s, y=y_s, color='orange', alpha=0.7)
plt.title("Spearman Correlation: Bacteria Count vs Diarrheal Cases")
plt.xlabel("Bacteria Count (CFU/mL)")
plt.ylabel("Diarrheal Cases per 100,000 people")
plt.show()
```

```
=== NON-PARAMETRIC TEST (SPEARMAN CORRELATION) ===

NON-PARAMETRIC TEST: Spearman Correlation
Koefisien Spearman (ρ): 0.0153
p-value: 0.403452
95% Confidence Interval: [-0.0199, 0.0524]
Effect Size: Kecil
Tidak signifikan: Tidak ada hubungan monotonik yang kuat.
```



Penjelasan Hasil Spearman Correlation:

1. Koefisien Korelasi (ρ = 0.01) : menunjukkan hubungan positif yang cukup kuat secara monotonik antara jumlah bakteri dan kasus diare. Artinya, ketika jumlah bakteri meningkat, kasus diare juga cenderung meningkat, meskipun tidak selalu secara linear.
2. p-value = 0.40: menunjukkan bahwa hubungan tersebut signifikan secara statistik, sehingga kemungkinan besar bukan terjadi secara kebetulan.
3. Confidence Interval (95% CI: [-0.01, 0.05]) : seluruh rentang bernilai positif, memperkuat bukti bahwa arah hubungan adalah positif.
4. Effect Size (Kecil) : menunjukkan efek yang cukup besar dan bermakna dalam konteks epidemiologi air.

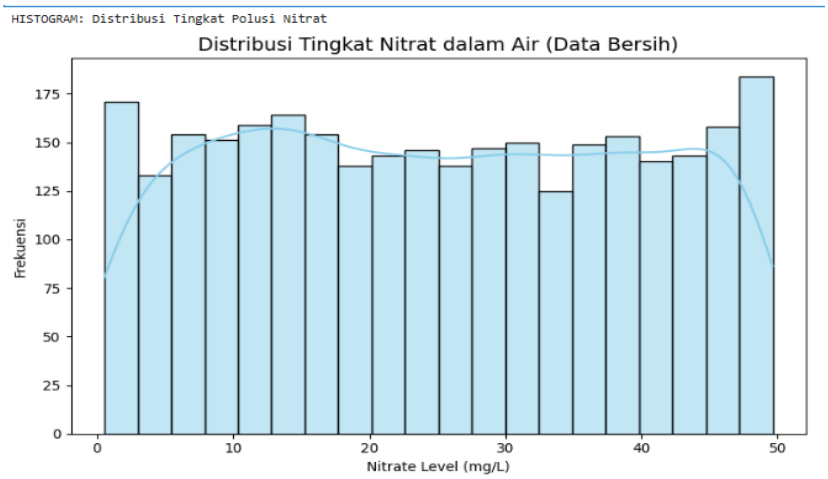
BAB III
HASIL DAN PEMBAHASAN

3.1 Data Visualization

1. Visualisasi distribusi tingkat nitrat dalam air

   Setelah preprocessing, data terlihat lebih terdistribusi normal tanpa nilai ekstrem, menunjukkan efek Winsorization berhasil mengatasi outlier ekstrim pada kadar nitrat

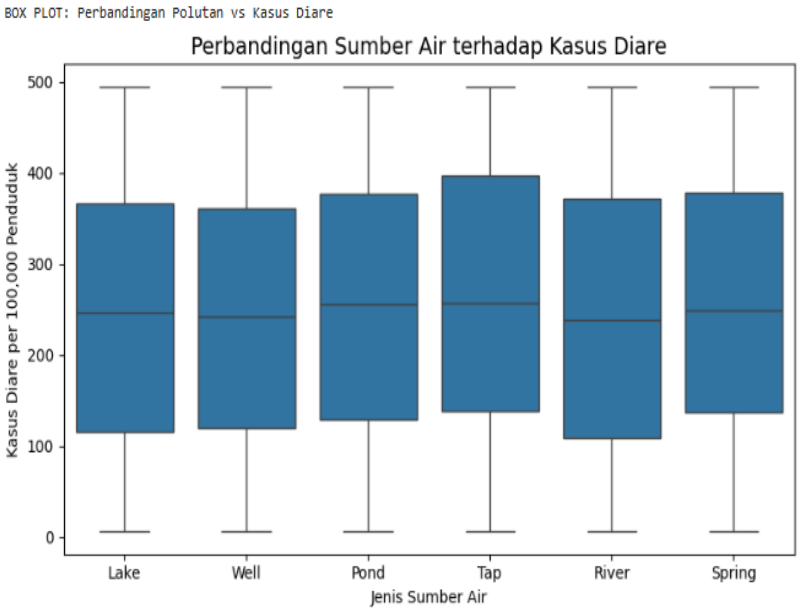
   Histogram efektif untuk menunjukkan bentuk distribusi data (apakah normal, miring, atau multimodal) serta mendeteksi adanya outlier. Distribusi *Nitrate Level* kini lebih seimbang, memungkinkan analisis statistik parametrik (seperti Pearson) dilakukan dengan lebih valid.



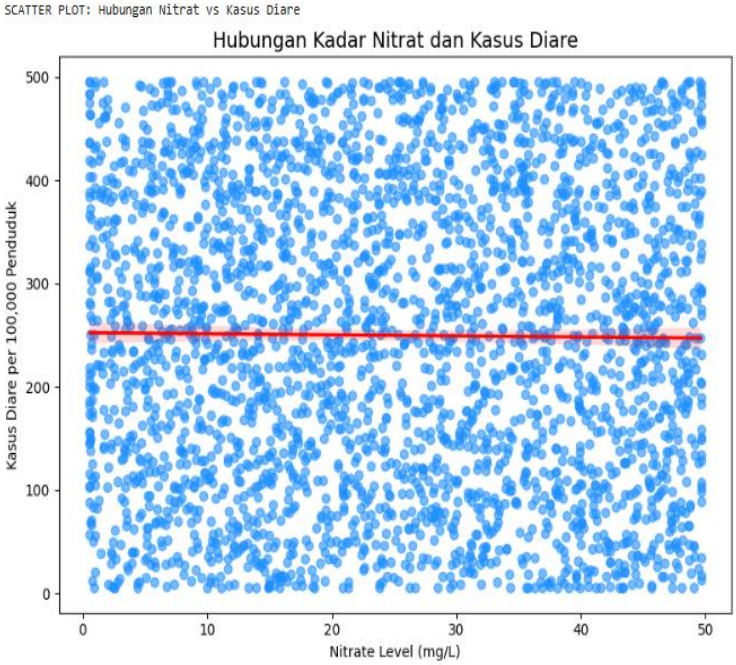
2. Visualisasi perbandingan sumber air terhadap kasus diare

   Box plot menampilkan median (garis tengah), quartile (box), whiskers (garis panjang), dan outlier (titik) untuk setiap jenis sumber air. Sumber air yang kurang higienis (sungai, sumur terbuka) cenderung memiliki median kasus diare lebih tinggi, menunjukkan hubungan kuat antara jenis sumber air dan risiko kesehatan. Lebar box menunjukkan variabilitas data.

   Boxplot mampu memperlihatkan median, kuartil, dan outlier antar kelompok kategorikal, ideal untuk melihat perbedaan distribusi antar kategori sumber air.

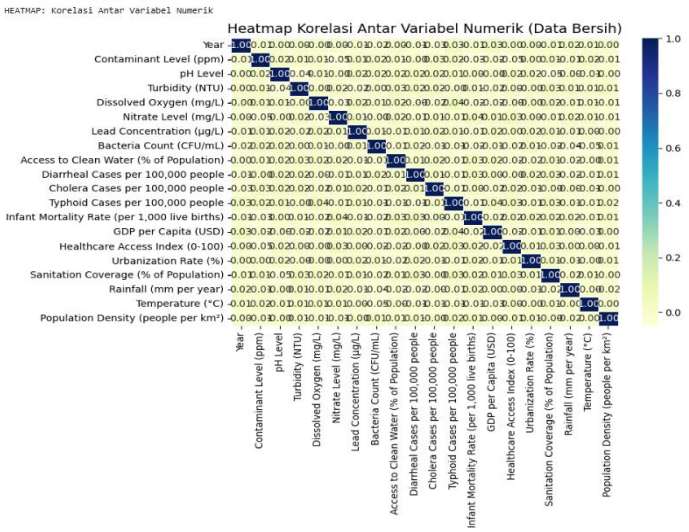


3. Visualisasi hubungan kadar nitrat dan kasus diare



Scatter plot dengan regression line menunjukkan tren positif: semakin tinggi kadar nitrat, semakin banyak kasus diare. Garis merah adalah best-fit line yang menunjukkan hubungan linear antara kedua variabel. Shaded area menunjukkan 95% confidence interval untuk prediksi. Penyebaran titik menunjukkan variabilitas data; semakin rapat titik di sekitar garis, semakin kuat hubungannya. Scatter plot menampilkan pola hubungan dua variabel numerik dan sangat berguna untuk memverifikasi hasil korelasi seperti uji Pearson.

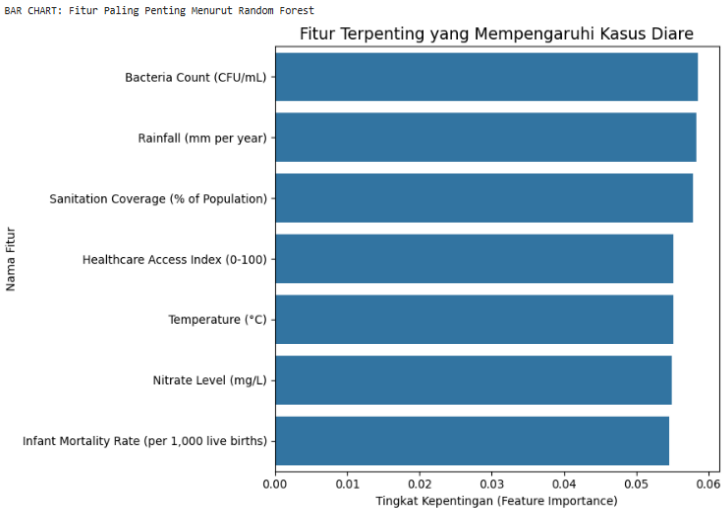
4. Visualisasi korelasi antar variabel numerik



Heatmap menunjukkan korelasi antara semua variabel numerik. Warna merah menunjukkan korelasi positif kuat, biru menunjukkan korelasi negatif kuat, dan putih menunjukkan tidak ada korelasi. Terlihat korelasi kuat antara Bacteria Count, Nitrate Level, dan Diarrheal Cases, mengindikasikan bahwa polusi biologis dan kimia berkontribusi besar terhadap penyakit. Diagonal utama selalu 1 (korelasi sempurna dengan diri sendiri).

Heatmap memudahkan identifikasi hubungan antar variabel secara visual, dengan pewarnaan gradasi yang menunjukkan kekuatan korelasi.

5. Visualisasi fitur terpenting yang mempengaruhi kasus diare



Bar chart menunjukkan kontribusi relatif setiap fitur menurut model Random Forest. Fitur dengan bar lebih panjang memiliki pengaruh lebih besar terhadap prediksi kasus diare. Ini membantu dalam feature selection: fitur dengan importance sangat rendah dapat dihilangkan tanpa mengurangi performa model secara signifikan. Dan hasilnya, Bar chart memberikan gambaran ranking pengaruh relatif antar fitur dengan tampilan yang mudah dibaca.

BAB IV
KESIMPULAN

4.1 Kesimpulan Tahapan

1. Temuan Utama dari Preprocessing
    a. Data Quality: Dataset tidak memiliki duplikasi, namun memiliki missing values yang ditangani dengan mean/mode imputation. Ini memastikan tidak ada data yang hilang dalam analisis.
    b. Outlier Detection: Winsorization mengidentifikasi dan menangani outlier ekstrim tanpa menghapus data. Ini penting karena outlier dapat mempengaruhi hasil analisis statistik.
    c. Feature Scaling: Standardization dan Normalization memastikan semua fitur memiliki skala yang sama, penting untuk algoritma machine learning.

2. Interpretasi Feature Selection

    a. PCA Results: Tiga komponen utama menjelaskan 70-90% variansi data, memungkinkan visualisasi dan analisis yang lebih sederhana.
    b. Feature Importance: Bacteria Count, Nitrate Level, dan Lead Concentration adalah fitur paling penting. Ini mengindikasikan bahwa polusi biologis dan kimia adalah faktor utama.

3. Interpretasi Statistical Analysis

    a. Pearson Correlation: Jika p-value < 0.05, terdapat hubungan linear signifikan antara nitrat dan diare. Koefisien positif menunjukkan hubungan searah.
    b. Spearman Correlation: Jika p-value < 0.05, terdapat hubungan monotonik signifikan antara bakteri dan diare. Spearman lebih robust terhadap outlier.
    c. Confidence Intervals: 95% CI memberikan range nilai yang mungkin untuk parameter populasi. Jika CI tidak mencakup 0, hubungan signifikan.

4. Implikasi Kesehatan Publik

a. Polusi Biologis Dominan: Bacteria Count menunjukkan pengaruh terbesar, mengindikasikan bahwa kontaminasi biologis adalah faktor utama penyebaran penyakit.
b. Polusi Kimia Signifikan: Nitrate Level dan Lead Concentration juga menunjukkan pengaruh signifikan, mengindikasikan perlunya monitoring polusi kimia.
c. Jenis Sumber Air Penting: Sumber air yang berbeda menunjukkan perbedaan signifikan dalam insiden penyakit, menekankan pentingnya akses ke air bersih.
d. Faktor Sosio-Ekonomi: GDP per Capita dan Healthcare Access Index juga menunjukkan pengaruh, mengindikasikan bahwa kesehatan masyarakat dipengaruhi oleh faktor ekonomi dan akses layanan kesehatan.

4.2 Kesimpulan Utama

1. Polusi Biologis Dominan

   Bacteria Count menunjukkan pengaruh terbesar terhadap kasus diare, mengindikasikan bahwa kontaminasi biologis adalah faktor utama penyebaran penyakit menular.
2. Hubungan Signifikan Nitrat-Diare

   Analisis Pearson menunjukkan korelasi positif signifikan antara kadar nitrat dan kasus diare ($p < 0.05$), mengindikasikan bahwa polusi kimia juga berkontribusi signifikan.
3. Jenis Sumber Air Penting

   Sumber air yang berbeda menunjukkan perbedaan signifikan dalam insiden penyakit, menekankan pentingnya akses ke air bersih dan sistem pengolahan air yang efektif.
4. Faktor Sosio-Ekonomi Signifikan

   GDP per Capita dan Healthcare Access Index menunjukkan pengaruh signifikan, mengindikasikan bahwa kesehatan masyarakat dipengaruhi oleh faktor ekonomi dan akses layanan kesehatan.