

LAPORAN
TUGAS PJBL KELOMPOK
MATA KULIAH DATA MINING A
“ANALISIS DATA PERSENTASE RUMAH TANGGA MENURUT
PROVINSI DAN SUMBER AIR MINUM LAYAK (PERSEN) 2010 - 2022”



DISUSUN OLEH:

1. Sandria Amelia Putri (21083010005)
2. Alya Setya Paramita (21083010046)

DOSEN PENGAMPU:

Trimono, S.Si., M.Si.
Tresna Maulana Fahrudin S.ST., M.T.

PROGRAM STUDI SAINS DATA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN” JAWA
TIMUR

2022

BAB 1: PENDAHULUAN

1.1. Permasalahan

Kami mengambil data pada website BPS tentang persentase rumah tangga menurut provinsi dan sumber air minum layak 2010-2022. Data tersebut berisi tentang persentase setiap rumah tangga dalam mendapatkan sumber air minum yang layak dan ditinjau dari setiap provinsi pada tahun 2010, 2021, dan 2022. Data tersebut masih termasuk data yang tidak terstruktur dengan baik karena masih terdapat data dengan nilai null di dalamnya dikarenakan terdapat salah satu provinsi yaitu provinsi Kalimantan Utara yang tidak memiliki data pada tahun 2010 karena merupakan yang baru saja diresmikan pada tahun 2013. Maka dari itu kami mengolah data ini agar menjadi data yang lebih baik. Dapat diakses melalui link berikut.

<https://www.bps.go.id/indicator/29/845/1/persentase-rumah-tangga-menurut-provinsi-dan-sumber-air-minum-layak.html>

1.2. Tinjauan Pustaka

1.2.1. Data Cleaning

Proses pembersihan data adalah proses kompleks dan terdiri dari beberapa tahap yang meliputi penentuan aturan kualitas data, mendeteksi eror/kesalahan data, dan memperbaiki kesalahan. Pembersihan data dibagi menjadi pembersihan data tradisional dan pembersihan untuk data skala besar [1]. Metode pembersihan data tradisional disebut tradisional karena tidak cocok untuk menangani sejumlah data dengan volume dan skala besar. Potter's Wheel and Intelliclean adalah beberapa contoh pembersihan data secara tradisional [1]. Implementasi proses data cleaning banyak digunakan pada Sentimen Analisis, text processing, Machine Learning, dan Natural Language Processing.

1.2.2. Outlier

Outlier berbeda dari data noise. Noise adalah random error atau varians dalam variabel yang diukur. Secara umum, noise ini tidak menarik dalam analisis data, termasuk deteksi outlier. Sebagai contoh, dalam deteksi penipuan kartu kredit, perilaku pembelian pelanggan dapat dimodelkan sebagai sebuah variabel acak. Sebagai contoh pelanggan dapat menghasilkan beberapa "noise transaksi" yang mungkin tampak seperti "kesalahan yang acak" atau "variens," seperti dengan membeli makan siang yang lebih besar satu hari, atau memiliki satu lebih cangkir kopi dari biasanya. Transaksi tersebut tidak akan diperlakukan sebagai outlier; Jika tidak, perusahaan kartu kredit akan dikenakan biaya berat dari memverifikasi bahwa banyak transaksi. Perusahaan juga akan kehilangan pelanggan oleh mengganggu mereka dengan beberapa alarm palsu. Seperti banyak lainnya analisis data dan pertambahan data tugas, noise harus dihapus sebelum deteksi outlier[2].

Outliers menarik karena mereka diduga tidak dihasilkan oleh mekanisme yang sama sebagai sisa data. Oleh karena itu, dalam deteksi outlier, penting untuk membenarkan mengapa outliers terdeteksi dihasilkan oleh beberapa mekanisme lain. Hal ini sering dicapai dengan membuat berbagai asumsi di sisa data dan menunjukkan

bahwa outliers terdeteksi melanggar anggapan-anggapan secara signifikan. Deteksi outlier juga berkaitan dengan novelty deteksi dalam berkembang data set. Sebagai contoh, dengan memantau situs web media sosial mana konten baru masuk, novelty deteksi dapat mengidentifikasi tren dan topik baru secara tepat waktu. Novelty topik mungkin pada awalnya muncul sebagai outliers. Sejauh ini, outlier deteksi dan novelty deteksi berbagi beberapa kesamaan dalam metode Deteksi dan pemodelan. Namun, kritis perbedaan antara keduanya adalah bahwa di novelty deteksi, setelah topik baru dikonfirmasi, mereka biasanya dimasukkan ke dalam model perilaku normal sehingga kasus tindak lanjut tidak diperlakukan sebagai outliers lagi [2].

1.2.3. Numpy

Numpy merupakan salah satu library dalam Python. Library ini dapat digunakan untuk banyak case dalam data science. Dengan adanya library ini, maka tidak diperlukan lagi baris kode yang panjang untuk menjalankan program machine learning. Numpy sendiri merupakan singkatan dari Numerical Python. Pada umumnya penggunaan library ini untuk menghitung operasi matematika pada array.

1.2.4. Pandas

Pandas dibangun di atas modul Numpy yang memiliki beberapa keunggulan, diantaranya menawarkan struktur data yang kaya dan memiliki banyak fungsi siap pakai untuk bekerja dengan data secara cepat, mudah dan gampang diikuti. Juga, penggunaan API yang memiliki konsistensi tinggi menjadikannya lebih mudah dipakai oleh para analis data.

1.2.5. Seaborn

Seaborn adalah library yang bertujuan untuk membuat grafik dan statistik dengan menggunakan Python. Seaborn dibangun berdasarkan library matplotlib serta terintegrasi dengan struktur data pada Pandas.

1.2.6. Matplotlib

Matplotlib adalah library Python 2 Dimensi yang dapat menghasilkan plot dengan kualitas tinggi dalam berbagai format dan dapat digunakan di banyak platform. Matplotlib dapat digunakan sebagai pembuat grafik dalam berbagai platform, seperti Python dan Jupyter Notebook. Grafik yang dapat dibuat oleh matplotlib cukup beragam, seperti grafik garis, batang, lingkaran, histogram dan berbagai macam lainnya.

BAB 2: ANALISIS DAN PEMBAHASAN

2.1. Baca file excel pada python

Kami mengambil data pada website BPS tentang persentase rumah tangga menurut provinsi dan sumber air minum layak 2010-2022. Dapat diakses melalui link berikut.

<https://www.bps.go.id/indicator/29/845/1/persentase-rumah-tangga-menurut-provinsi-dan-sumber-air-minum-layak.html>

Untuk membaca file dimulai dengan import library yang diperlukan yaitu numpy dan pandas. Baca file excel data persentase rumah tangga menurut provinsi dan sumber air minum layak 2010-2022 dengan fungsi `pd.read_excel`.

```
[1]: # import library
import numpy as np
import pandas as pd
```

```
# data persentase rumah tangga menurut provinsi dan sumber air minum Layak 2010-2022
persentase_kelayakan = pd.read_excel(r"C:\Users\Sandria\Python\Semester 4\Data Mining\Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak.xlsx")
persentase_kelayakan
```

| | Provinsi | Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak (Persen) | Unnamed: 2 | Unnamed: 3 | |
|----|---|--|------------|------------|---------|
| 0 | NaN | | 2010 | 2021.00 | 2022.00 |
| 1 | ACEH | | 29.02 | 88.79 | 89.70 |
| 2 | SUMATERA UTARA | | 46.06 | 90.89 | 92.13 |
| 3 | SUMATERA BARAT | | 41.92 | 83.40 | 85.23 |
| 4 | RIAU | | 40.01 | 89.76 | 90.07 |
| 5 | JAMBI | | 48.28 | 79.70 | 79.19 |
| 6 | SUMATERA SELATAN | | 45.99 | 84.70 | 86.35 |
| 7 | BENGKULU | | 28.23 | 67.39 | 73.07 |
| 8 | LAMPUNG | | 38.07 | 80.20 | 81.60 |
| 9 | KEP. BANGKA BELITUNG | | 38.17 | 73.40 | 80.96 |
| 10 | KEP. RIAU | | 23.82 | 90.83 | 91.82 |
| 11 | DKI JAKARTA | | 28.41 | 99.86 | 97.93 |
| 12 | JAWA BARAT | | 35.32 | 93.24 | 93.04 |
| 13 | JAWA TENGAH | | 27.11 | 83.23 | 83.33 |
| | | | | | |
| 24 | KALIMANTAN TIMUR | | 43.27 | 85.80 | 87.14 |
| 25 | KALIMANTAN UTARA | - | | 86.80 | 89.96 |
| 26 | SULAWESI UTARA | | 44.41 | 91.65 | 94.15 |
| 27 | SULAWESI TENGAH | | 35.10 | 88.51 | 86.74 |
| 28 | SULAWESI SELATAN | | 45.12 | 91.18 | 91.96 |
| 29 | SULAWESI TENGGARA | | 50.74 | 91.94 | 94.64 |
| 30 | GORONTALO | | 40.09 | 94.57 | 96.16 |
| 31 | SULAWESI BARAT | | 37.44 | 78.35 | 78.98 |
| 32 | MALUKU | | 56.95 | 93.21 | 92.10 |
| 33 | MALUKU UTARA | | 54.18 | 88.66 | 88.10 |
| 34 | PAPUA BARAT | | 45.26 | 81.68 | 81.57 |
| 35 | PAPUA | | 32.42 | 64.92 | 65.39 |
| 36 | INDONESIA | | 44.19 | 90.78 | 91.05 |
| 37 | NaN | | NaN | NaN | NaN |
| 38 | \n\n\n\n\n\n\n\n Pada tahun 2000 pencacahan Su... | | NaN | NaN | NaN |
| 39 | Source Url: https://www.bps.go.id/indicator/29... | | NaN | NaN | NaN |
| 40 | Access Time: March 10, 2023, 10:49 am | | NaN | NaN | NaN |

2.2. Memberi nama variabel header

Inisialisasikan terlebih dahulu nama_variabel yang akan dijadikan header. Tambahkan nama_variabel tersebut pada header dataset.

```
# memberi nama variabel header
nama_variabel = ['Provinsi', '2010', '2021', '2022']
persentase_kelayaan = pd.read_excel(r"C:\Users\Sandria\Python\Semester 4\Data Mining\Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak.xlsx", header=None, names=nama_variabel)
persentase_kelayaan
```

| | Provinsi | 2010 | 2021 | 2022 |
|----|----------------------|-------|---------|---------|
| 0 | Provinsi | NaN | NaN | NaN |
| 1 | NaN | 2010 | 2021.00 | 2022.00 |
| 2 | ACEH | 29.02 | 88.79 | 89.70 |
| 3 | SUMATERA UTARA | 46.06 | 90.89 | 92.13 |
| 4 | SUMATERA BARAT | 41.92 | 83.40 | 85.23 |
| 5 | RIAU | 40.01 | 89.76 | 90.07 |
| 6 | JAMBI | 48.28 | 79.70 | 79.19 |
| 7 | SUMATERA SELATAN | 45.99 | 84.70 | 86.35 |
| 8 | BENGKULU | 28.23 | 67.39 | 73.07 |
| 9 | LAMPUNG | 38.07 | 80.20 | 81.60 |
| 10 | KEP. BANGKA BELITUNG | 38.17 | 73.40 | 80.96 |
| 11 | KEP. RIAU | 23.82 | 90.83 | 91.82 |
| 12 | DKI JAKARTA | 28.41 | 99.86 | 97.93 |
| 13 | JAWA BARAT | 35.32 | 93.24 | 93.04 |
| 14 | JAWA TENGAH | 57.44 | 93.62 | 93.32 |
| 15 | DI YOGYAKARTA | 60.41 | 95.69 | 96.50 |
| 16 | JAWA TIMUR | 52.94 | 95.02 | 95.05 |
| 17 | BANTEN | 22.32 | 93.51 | 92.71 |
| 18 | BALI | 48.44 | 97.56 | 98.42 |
| 19 | NUSA TENGGARA BARAT | 46.20 | 94.60 | 95.40 |

2.3. Menghapus beberapa baris data yang memiliki nilai NaN

Gunakan fungsi dropna().

```
[4]: # menghapus beberapa baris data yang memiliki nilai NaN
persentase_kelayakan = persentase_kelayakan.dropna()
persentase_kelayakan
```

```
[4]:
```

| | Provinsi | 2010 | 2021 | 2022 |
|----|----------------------|-------|-------|-------|
| 2 | ACEH | 29.02 | 88.79 | 89.70 |
| 3 | SUMATERA UTARA | 46.06 | 90.89 | 92.13 |
| 4 | SUMATERA BARAT | 41.92 | 83.40 | 85.23 |
| 5 | RIAU | 40.01 | 89.76 | 90.07 |
| 6 | JAMBI | 48.28 | 79.70 | 79.19 |
| 7 | SUMATERA SELATAN | 45.99 | 84.70 | 86.35 |
| 8 | BENGKULU | 28.23 | 67.39 | 73.07 |
| 9 | LAMPUNG | 38.07 | 80.20 | 81.60 |
| 10 | KEP. BANGKA BELITUNG | 38.17 | 73.40 | 80.96 |
| 11 | KEP. RIAU | 23.82 | 90.83 | 91.82 |
| 12 | DKI JAKARTA | 28.41 | 99.86 | 97.93 |
| 13 | JAWA BARAT | 35.32 | 93.24 | 93.04 |
| 14 | JAWA TENGAH | 57.44 | 93.62 | 93.32 |
| 15 | DI YOGYAKARTA | 60.41 | 95.69 | 96.50 |

2.4. Menghapus data duplikat pada kolom Provinsi dengan tetap mempertahankan salah satu baris (baris pertama)

Untuk menghilangkan data yang duplikat dapat menggunakan dprop_duplicates

```
[5]: # menghapus data duplikat pada kolom Provinsi
# menghapus data yang berulang dengan tetap mempertahankan salah satu baris (baris pertama)
persentase_kelayakan.drop_duplicates(subset="Provinsi", keep = 'first', inplace = True)
persentase_kelayakan

# subset: menetapkan nama kolom ke parameter subset untuk memeriksa nilai berulang
# keep: untuk menyimpan baris pertama
# inplace: boolean case (True or False), "True" akan dengan menghapus nilai selain baris pertama

C:\Users\Sandria\anaconda3\lib\site-packages\pandas\util\_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
return func(*args, **kwargs)
```

```
[5]:
```

| | Provinsi | 2010 | 2021 | 2022 |
|---|------------------|-------|-------|-------|
| 2 | ACEH | 29.02 | 88.79 | 89.70 |
| 3 | SUMATERA UTARA | 46.06 | 90.89 | 92.13 |
| 4 | SUMATERA BARAT | 41.92 | 83.40 | 85.23 |
| 5 | RIAU | 40.01 | 89.76 | 90.07 |
| 6 | JAMBI | 48.28 | 79.70 | 79.19 |
| 7 | SUMATERA SELATAN | 45.99 | 84.70 | 86.35 |
| 8 | BENGKULU | 28.23 | 67.39 | 73.07 |

2.5. Memeriksa apakah terdapat missing value pada data Menggunakan isnul()

```
[6]: # missing Value Treatment
# fungsi .isnull() digunakan untuk memeriksa nilai yang hilang
missing = persentase_kelayakan.isnull()
print(missing)

# missing value dapat ditangani dengan dua cara: diisi atau di buang
```

| | Provinsi | 2010 | 2021 | 2022 |
|----|----------|-------|-------|-------|
| 2 | | False | False | False |
| 3 | | False | False | False |
| 4 | | False | False | False |
| 5 | | False | False | False |
| 6 | | False | False | False |
| 7 | | False | False | False |
| 8 | | False | False | False |
| 9 | | False | False | False |
| 10 | | False | False | False |
| 11 | | False | False | False |
| 12 | | False | False | False |
| 13 | | False | False | False |
| 14 | | False | False | False |
| 15 | | False | False | False |
| 16 | | False | False | False |
| 17 | | False | False | False |
| 18 | | False | False | False |
| 19 | | False | False | False |
| 20 | | False | False | False |
| 21 | | False | False | False |

2.6. Penyuntingan kesalahan kata atau mengganti sebuah kata

Untuk menangani kata yang salah atau typo dapat kita ubah lagi kata tersebut menggunakan str.replace

```
[7]: # penanganan kalimat typo
persentase_kelayakan['Provinsi'] = persentase_kelayakan['Provinsi'].str.replace('KEP. BANGKA BELITUNG', 'KEP BANGKA BELITUNG')
persentase_kelayakan
```

C:\Users\Sandria\AppData\Local\Temp\ipykernel_45600\2725608409.py:2: FutureWarning: The default value of regex will change from True to False in a future version.

C:\Users\Sandria\AppData\Local\Temp\ipykernel_45600\2725608409.py:2: SettingWithCopyWarning: A value is trying to be set on a copy of a slice from a DataFrame. Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
persentase_kelayakan['Provinsi'] = persentase_kelayakan['Provinsi'].str.replace('KEP. BANGKA BELITUNG', 'KEP BANGKA BELITUNG')
```

| | Provinsi | 2010 | 2021 | 2022 |
|---|------------------|-------|-------|-------|
| 2 | ACEH | 29.02 | 88.79 | 89.70 |
| 3 | SUMATERA UTARA | 46.06 | 90.89 | 92.13 |
| 4 | SUMATERA BARAT | 41.92 | 83.40 | 85.23 |
| 5 | RIAU | 40.01 | 89.76 | 90.07 |
| 6 | JAMBI | 48.28 | 79.70 | 79.19 |
| 7 | SUMATERA SELATAN | 45.99 | 84.70 | 86.35 |
| 8 | BENGKULU | 28.23 | 67.39 | 73.07 |
| 9 | LAMPUNG | 38.07 | 80.20 | 81.60 |

2.7. Mengganti data NaN dengan nilai rata-rata

Jika terdapat data yang bernilai null dan kita tidak ingin untuk menghapus datanya maka untuk menanganinya kita dapat mengganti nilai null tersebut dengan nilai baru yang didapatkan dari rata-rata nilainya. Data pada provinsi Kalimantan Utara tahun 2010 bersifat null maka, kita membuat list baru tanpa provinsi tersebut lalu menghitung nilai rata-ratanya, kemudian nilai rata-rata yang didapatkan kita masukkan pada data null tadi.

```
[8]: # cek tipe data, diketahui bahwa data pada kolom 2010 bertipe object
persentase_kelayakan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 35 entries, 2 to 36
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  -
 0   Provinsi    35 non-null    object
 1   2010        35 non-null    object
 2   2021        35 non-null    float64
 3   2022        35 non-null    float64
dtypes: float64(2), object(2)
memory usage: 1.4+ KB
```

```
[9]: # buat variabel baru df yang berisikan data persentase kelayakan tanpa baris ke-25
df = persentase_kelayakan.drop(25)
df
```

```
[9]:
```

| | Provinsi | 2010 | 2021 | 2022 |
|----|---------------------|-------|-------|-------|
| 2 | ACEH | 29.02 | 88.79 | 89.70 |
| 3 | SUMATERA UTARA | 46.06 | 90.89 | 92.13 |
| 4 | SUMATERA BARAT | 41.92 | 83.40 | 85.23 |
| 5 | RIAU | 40.01 | 89.76 | 90.07 |
| 6 | JAMBI | 48.28 | 79.70 | 79.19 |
| 13 | JAWA BARAT | 35.32 | 93.24 | 93.04 |
| 14 | JAWA TENGAH | 57.44 | 93.62 | 93.32 |
| 15 | DI YOGYAKARTA | 60.41 | 95.69 | 96.50 |
| 16 | JAWA TIMUR | 52.94 | 95.02 | 95.05 |
| 17 | BANTEN | 22.32 | 93.51 | 92.71 |
| 18 | BALI | 48.44 | 97.56 | 98.42 |
| 19 | NUSA TENGGARA BARAT | 46.20 | 94.60 | 95.40 |
| 20 | NUSA TENGGARA TIMUR | 49.29 | 85.40 | 86.76 |
| 21 | KALIMANTAN BARAT | 54.47 | 78.76 | 80.43 |
| 22 | KALIMANTAN TENGAH | 40.55 | 77.05 | 77.01 |
| 23 | KALIMANTAN SELATAN | 48.97 | 76.40 | 76.18 |
| 24 | KALIMANTAN TIMUR | 43.27 | 85.80 | 87.14 |
| 26 | SULAWESI UTARA | 44.41 | 91.65 | 94.15 |
| 27 | SULAWESI TENGAH | 35.10 | 88.51 | 86.74 |
| 28 | SULAWESI SELATAN | 45.12 | 91.18 | 91.96 |
| 29 | SULAWESI TENGGARA | 50.74 | 91.94 | 94.64 |
| 30 | GORONTALO | 40.09 | 94.57 | 96.16 |

```
[10]: # cek tipe data, diketahui bahwa data pada kolom 2010 masih bertipe object
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 34 entries, 2 to 36
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   Provinsi    34 non-null      object
1   2010        34 non-null      object
2   2021        34 non-null      float64
3   2022        34 non-null      float64
dtypes: float64(2), object(2)
memory usage: 1.3+ KB
```

```
[11]: # ubah tipe data pada kolom 2010 menjadi float agar dapat dihitung nilai mean-nya
df["2010"] = df["2010"].astype("float")
```

```
[12]: # cek tipe data, terlihat bahwa data pada kolom 2010 telah berubah menjadi tipe float
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 34 entries, 2 to 36
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   Provinsi    34 non-null      object
1   2010        34 non-null      float64
2   2021        34 non-null      float64
3   2022        34 non-null      float64
dtypes: float64(3), object(1)
memory usage: 1.3+ KB
```

```
[13]: # dengan fungsi describe() didapatkan nilai mean dari data kolom 2010 yaitu 42.75
df.describe()
```

```
[13]:
```

| | 2010 | 2021 | 2022 |
|-------|-----------|-----------|-----------|
| count | 34.000000 | 34.000000 | 34.000000 |
| mean | 42.750000 | 86.794706 | 87.672059 |
| std | 9.617082 | 8.490131 | 7.866825 |
| min | 22.320000 | 64.920000 | 65.390000 |
| 25% | 37.597500 | 80.570000 | 81.577500 |
| 50% | 44.300000 | 89.275000 | 89.885000 |
| 75% | 48.837500 | 93.232500 | 93.250000 |
| max | 60.410000 | 99.860000 | 98.420000 |

```
[14]: # selain itu, dapat menggunakan numpy untuk mencari nilai mean
a = np.mean(df['2010'])
print(a)

42.75
```

```
[15]: # mengisi data null atau NaN dengan nilai mean
persentase_kelayakan['2010'] = persentase_kelayakan['2010'].str.replace('-', '42.75')
persentase_kelayakan

C:\Users\Sandria\AppData\Local\Temp\ipykernel_45600\2516670003.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
persentase_kelayakan['2010'] = persentase_kelayakan['2010'].str.replace('-', '42.75')
```

| | Provinsi | 2010 | 2021 | 2022 |
|----|---------------------|-------|-------|-------|
| 2 | ACEH | 29.02 | 88.79 | 89.70 |
| 3 | SUMATERA UTARA | 46.06 | 90.89 | 92.13 |
| 4 | SUMATERA BARAT | 41.92 | 83.40 | 85.23 |
| 5 | RIAU | 40.01 | 89.76 | 90.07 |
| 6 | JAMBI | 48.28 | 79.70 | 79.19 |
| 7 | SUMATERA SELATAN | 45.99 | 84.70 | 86.35 |
| 8 | BENGKULU | 28.23 | 67.39 | 73.07 |
| 9 | LAMPUNG | 38.07 | 80.20 | 81.60 |
| 10 | KEP BANGKA BELITUNG | 38.17 | 73.40 | 80.96 |
| 11 | KEP. RIAU | 23.82 | 90.83 | 91.82 |
| 16 | JAWA TIMUR | 52.94 | 95.02 | 95.05 |
| 17 | BANTEN | 22.32 | 93.51 | 92.71 |
| 18 | BALI | 48.44 | 97.56 | 98.42 |
| 19 | NUSA TENGGARA BARAT | 46.20 | 94.60 | 95.40 |
| 20 | NUSA TENGGARA TIMUR | 49.29 | 85.40 | 86.76 |
| 21 | KALIMANTAN BARAT | 54.47 | 78.76 | 80.43 |
| 22 | KALIMANTAN TENGAH | 40.55 | 77.05 | 77.01 |
| 23 | KALIMANTAN SELATAN | 48.97 | 76.40 | 76.18 |
| 24 | KALIMANTAN TIMUR | 43.27 | 85.80 | 87.14 |
| 25 | KALIMANTAN UTARA | 42.75 | 86.80 | 89.96 |
| 26 | SULAWESI UTARA | 44.41 | 91.65 | 94.15 |
| 27 | SULAWESI TENGAH | 35.10 | 88.51 | 86.74 |
| 28 | SULAWESI SELATAN | 45.12 | 91.18 | 91.96 |
| 29 | SULAWESI TENGGARA | 50.74 | 91.94 | 94.64 |
| 30 | GORONTALO | 40.09 | 94.57 | 96.16 |
| 31 | SULAWESI BARAT | 37.44 | 78.35 | 78.98 |
| 32 | MALUKU | 56.95 | 93.21 | 92.10 |

```
[16]: # ubah tipe data pada kolom 2010 menjadi float agar dapat divisualisasikan
persentase_kelayakan["2010"] = persentase_kelayakan["2010"].astype("float")
# cek tipe data, terlihat bahwa data pada kolom 2010 telah berubah menjadi tipe float
persentase_kelayakan.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 35 entries, 2 to 36
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype  
---  --
 0   Provinsi    35 non-null     object  
 1   2010        35 non-null     float64 
 2   2021        35 non-null     float64 
 3   2022        35 non-null     float64 
dtypes: float64(3), object(1)
memory usage: 2.4+ KB

C:\Users\Sandria\AppData\Local\Temp\ipykernel_45600\2876142188.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
persentase_kelayakan["2010"] = persentase_kelayakan["2010"].astype("float")
```

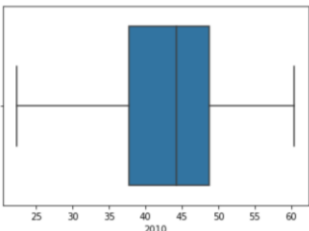
2.8. Mendeteksi outlier dengan boxplot

Untuk mendeteksi ada atau tidaknya outlier salah satu caranya dapat menggunakan boxplot dengan fungsi sns.boxplot.

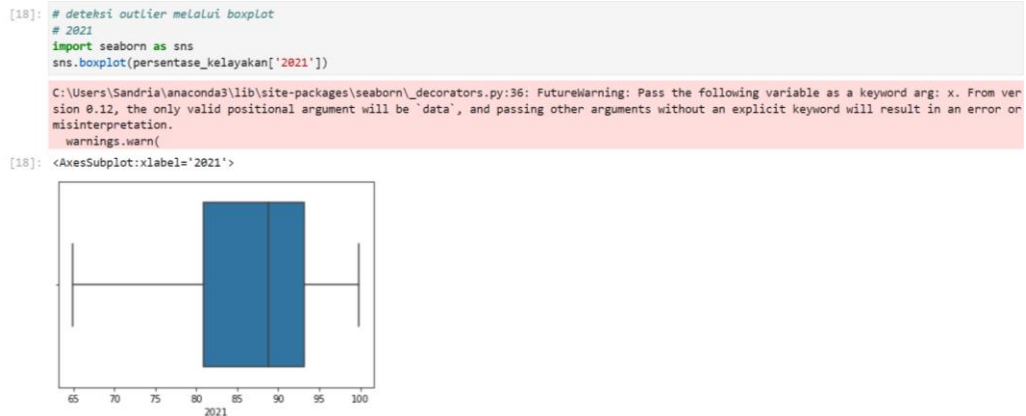
```
[17]: # deteksi outlier melalui boxplot
# 2010
import seaborn as sns
sns.boxplot(persentase_kelayakan['2010'])

C:\Users\Sandria\anaconda3\lib\site-packages\seaborn\decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From ver
sion 0.12, the only valid positional argument will be "data", and passing other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(

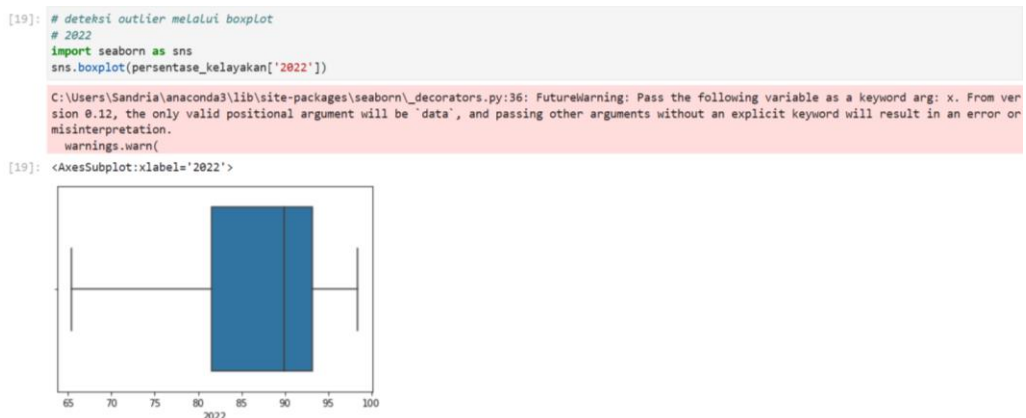
[17]: <AxesSubplot:xlabel='2010'>
```



Ketika memvisualisasikan data 2010 pada boxplot didapatkan bahwa nilai minimalnya adalah 20 dan nilai maksimalnya adalah 60. Nilai Q1 (kuartil bawah) yaitu 37.5, nilai Q2 (median) yaitu 44, dan nilai Q3 (kuartil atas) yaitu 49. Karena garis Q2 (median) hampir berada di tengah box dan panjang whisker hampir sama, maka bisa dikatakan bahwa data ini berdistribusi normal. Pada data 2010 juga tidak ditemukan outlier.



Ketika memvisualisasikan data 2021 pada boxplot didapatkan bahwa nilai minimalnya adalah 65 dan nilai maksimalnya adalah 100. Nilai Q1 (kuartil bawah) yaitu 81, nilai Q2 (median) yaitu 89, dan nilai Q3 (kuartil atas) yaitu 93. Karena garis Q2 (median) tidak berada di tengah box dan panjang whisker tidak sama, maka bisa dikatakan bahwa data ini tidak berdistribusi normal. Box lebih mengacu ke sisi kiri yang berarti lebih banyak data yang bernilai dibawah median. Pada data 2021 ini tidak ditemukan outlier.



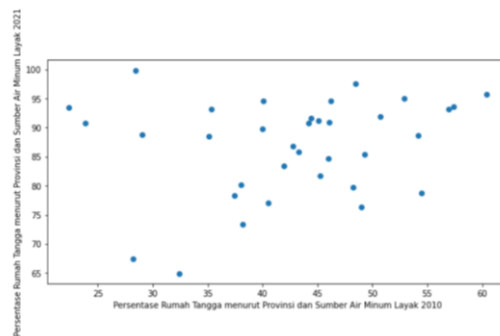
Ketika memvisualisasikan data 2022 pada boxplot didapatkan bahwa nilai minimalnya adalah 65 dan nilai maksimalnya adalah 100. Nilai Q1 (kuartil bawah) yaitu 81, nilai Q2 (median) yaitu 89, dan nilai Q3 (kuartil atas) yaitu 93. Karena garis Q2 (median) tidak berada di tengah box dan panjang whisker tidak sama, maka bisa dikatakan bahwa data ini tidak berdistribusi normal. Box lebih mengacu ke sisi kiri yang berarti lebih banyak data yang bernilai dibawah median. Pada data 2022 ini tidak ditemukan outlier.

2.9. Mendeteksi outlier dengan scatter plot

Untuk mendeteksi ada atau tidaknya outlier cara lainnya dapat menggunakan scatter plot. Dan didapatkan pada data bahwa data tahun 2010 dan 2021 serta data 2010 dan 2022 tidak terdapat outlier karena tersebar dengan baik, sedangkan data 2021 dan 2022 terdapat outlier di mana terdapat 3 titik yang sebarannya terlampaui jauh dari titik lainnya.

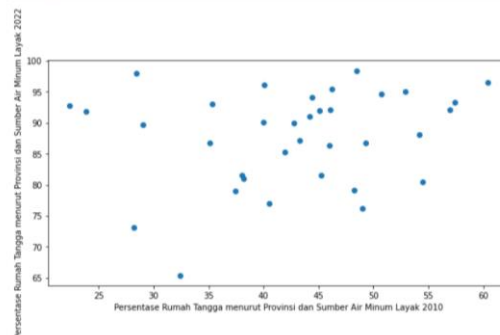
```
[20]: import matplotlib.pyplot as plt

# deteksi outlier melalui Scatter plot
# 2010 dan 2021
fig, ax = plt.subplots(figsize = (10,5))
ax.scatter(persentase_kelayakan['2010'], persentase_kelayakan['2021'])
# x-axis label
ax.set_xlabel('Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak 2010')
# y-axis label
ax.set_ylabel('Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak 2021')
plt.show()
```



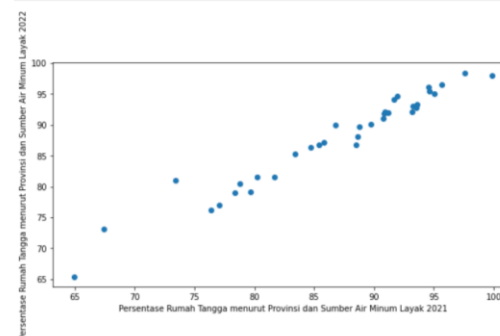
```
[21]: import matplotlib.pyplot as plt

# deteksi outlier melalui Scatter plot
# 2010 dan 2022
fig, ax = plt.subplots(figsize = (10,5))
ax.scatter(persentase_kelayakan['2010'], persentase_kelayakan['2022'])
# x-axis label
ax.set_xlabel('Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak 2010')
# y-axis label
ax.set_ylabel('Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak 2022')
plt.show()
```



```
[22]: import matplotlib.pyplot as plt

# deteksi outlier melalui Scatter plot
# 2021 dan 2022
fig, ax = plt.subplots(figsize = (10,5))
ax.scatter(persentase_kelayakan['2021'], persentase_kelayakan['2022'])
# x-axis label
ax.set_xlabel('Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak 2021')
# y-axis label
ax.set_ylabel('Persentase Rumah Tangga menurut Provinsi dan Sumber Air Minum Layak 2022')
plt.show()
```



2.10. Mendeteksi outlier dalam dua data

Di sini kita mendeteksi nilai outlier pada data tahun 2021 dan 2022 dengan menggunakan batas pada angka 75

```
[23]: # deteksi nilai outlier
# 2021 dan 2022
print(np.where((persentase_kelayakan['2021']<75) & (persentase_kelayakan['2022']<75)))

(array([ 6, 33], dtype=int64),)
```

2.11. Mendeteksi outlier dengan zscore

Untuk mendeteksi ada atau tidaknya outlier cara lainnya dapat menggunakan z score dengan fungsi np.abs(stats.zscore()).

```
[24]: # deteksi outlier melalui zscore
# Z score 2010
from scipy import stats
import numpy as np
z = np.abs(stats.zscore(persentase_kelayakan['2010']))
print(z)
# Position of the outlier
print(np.where(z > 3))

2      1.470294
3      0.354455
4      0.088882
5      0.293416
6      0.592187
7      0.346959
8      1.554892
9      0.501164
10     0.490455
11     2.027143
12     1.535617
13     0.795651
14     1.573097
15     1.891143
16     1.091209
17     2.187772
18     0.609321
19     0.369448
20     0.700344

11     2.027143
12     1.535617
13     0.795651
14     1.573097
15     1.891143
16     1.091209
17     2.187772
18     0.609321
19     0.369448
20     0.700344
Name: 2010, dtype: float64
(array([], dtype=int64),)
```

```
[25]: # deteksi outlier melalui zscore
# Z score 2021
from scipy import stats
import numpy as np
z = np.abs(stats.zscore(persentase_kelayakan['2021']))
print(z)
# Position of the outlier
print(np.where(z > 3))

2      0.242012
3      0.496743
4      0.411798
5      0.359673
6      0.860610
7      0.254107
8      2.353819
9      0.799960
10     1.624803
11     0.489465
12     1.584809
13     0.781799
14     0.827893
15     1.078985
16     0.997714
17     0.814550
18     1.305817
19     0.946768
20     0.169197

11     0.489465
12     1.584809
13     0.781799
14     0.827893
15     1.078985
16     0.997714
17     0.814550
18     1.305817
19     0.946768
20     0.169197
Name: 2021, dtype: float64
(array([], dtype=int64),)
```

```
[26]: # deteksi outlier melalui zscore
# Z score 2022
from scipy import stats
import numpy as np
z = np.abs(stats.zscore(persentase_kelayakan['2022']))
print(z)
# Position of the outlier
print(np.where(z > 3))

2      0.256604
3      0.574323
4      0.327843
5      0.304981
6      1.117566
7      0.181405
8      1.917748
9      0.802461
10     0.886140
11     0.533791
12     1.332666
13     0.693305
14     0.729914
15     1.145695
16     0.956110
17     0.650157
18     1.396733
19     1.001872
20     0.127798

11     0.533791
12     1.332666
13     0.693305
14     0.729914
15     1.145695
16     0.956110
17     0.650157
18     1.396733
19     1.001872
20     0.127798
Name: 2022, dtype: float64
(array([], dtype=int64),)
```

2.12. Mendeteksi outlier dengan IQR value

Untuk mendeteksi ada atau tidaknya outlier cara lainnya dapat menggunakan IQR value.

```
[27]: # deteksi outlier melalui IQR value 2010
# IQR
Q1 = np.percentile(persentase_kelayakan['2010'], 25,
                    interpolation = 'midpoint')
Q3 = np.percentile(persentase_kelayakan['2010'], 75,
                    interpolation = 'midpoint')
IQR = Q3 - Q1
# Above Upper bound
upper=Q3+1.5*IQR
upper_array=np.array(persentase_kelayakan['2010']>=upper)
print("Upper Bound: ",upper," \n")
print(upper_array)
# Below Lower bound
lower=Q1-1.5*IQR
lower_array=np.array(persentase_kelayakan['2010']<=lower)
print("Lower Bound: ",lower," \n")
print(lower_array)

Upper Bound: 65.13

[False False False False False False False False False False False False
 False False False False False False False False False False False False
 False False False False False False False False False False False False]
Lower Bound: 21.329999999999999

[False False False False False False False False False False False False
 False False False False False False False False False False False False
 False False False False False False False False False False False False]

[28]: # deteksi outlier melalui IQR value 2021
# IQR
Q1 = np.percentile(persentase_kelayakan['2021'], 25,
                    interpolation = 'midpoint')
Q3 = np.percentile(persentase_kelayakan['2021'], 75,
                    interpolation = 'midpoint')
IQR = Q3 - Q1
# Above Upper bound
upper=Q3+1.5*IQR
upper_array=np.array(persentase_kelayakan['2021']>=upper)
print("Upper Bound: ",upper," \n")
print(upper_array)
# Below Lower bound
lower=Q1-1.5*IQR
lower_array=np.array(persentase_kelayakan['2021']<=lower)
print("Lower Bound: ",lower," \n")
print(lower_array)

Upper Bound: 111.65249999999999

[False False False False False False False False False False False False
 False False False False False False False False False False False False
 False False False False False False False False False False False False]
Lower Bound: 62.5125

[False False False False False False False False False False False False
 False False False False False False False False False False False False
 False False False False False False False False False False False False]

[29]: # deteksi outlier melalui IQR value 2022
# IQR
Q1 = np.percentile(persentase_kelayakan['2022'], 25,
                    interpolation = 'midpoint')
Q3 = np.percentile(persentase_kelayakan['2022'], 75,
                    interpolation = 'midpoint')
IQR = Q3 - Q1
# Above Upper bound
upper=Q3+1.5*IQR
upper_array=np.array(persentase_kelayakan['2022']>=upper)
print("Upper Bound: ",upper," \n")
print(upper_array)
# Below Lower bound
lower=Q1-1.5*IQR
lower_array=np.array(persentase_kelayakan['2022']<=lower)
print("Lower Bound: ",lower," \n")
print(lower_array)

Upper Bound: 110.57250000000002

[False False False False False False False False False False False False
 False False False False False False False False False False False False
 False False False False False False False False False False False False]
Lower Bound: 64.192499999999997

[False False False False False False False False False False False False
 False False False False False False False False False False False False
 False False False False False False False False False False False False]
```

DAFTAR PUSTAKA

- [1] F. Ridzuan and W. M. N. Wan Zainon, “A review on data cleansing methods for big data,” *Procedia Comput. Sci.*, vol. 161, pp. 731–738, 2019, doi:10.1016/j.procs.2019.11.177.
- [2] F. A. Hermawati, *Data Mining*, Surabaya: Andi, 2009