ANALISIS 3 FAKTOR PENYEBAB JUMLAH TINDAK PIDANA DI INDONESIA

DEPARTEMEN S1 SAINS DATA • FMIPA UNESA

DISUSUN OLEH :

Ivan Cahya Aryasuta (24031554172)

Firda Nurkhairani (24031554209)

KELOMPOK 6
UAS MATA KULIAH DATA WRANGLING
Dosen Pengampu:
Dinda Galuh Guminta, S.Stat.,M.Stat. (NIDN : 0011129602)
Belgis Ainatul Iza, S.Si., M.Mat. (NIP : 202509237)

PENDAHULUAN

LATAR BELAKANG

Pemanfaatan data digital dalam analisis sosial ekonomi kini menjadi kebutuhan penting terutama karena banyak instansi pemerintah menyediakan data secara terbuka namun dalam kondisi yang tidak selalu rapi dan sulit langsung dianalisis (raw data). Oleh karena itu, dibutuhkan proses web scraping untuk mengambil data dari sumber daring serta data wrangling untuk membersihkan, menstandarkan, dan mengintegrasikan informasi dari berbagai indikator seperti tingkat pengangguran, IPM, kemiskinan, dan tindak kriminal. Melalui proses tersebut, data mentah yang semula berantakan dapat diubah menjadi dataset terstruktur yang dilanjut dengan dilakukannya analisis visual dan eksplorasi hubungan antarvariabel sehingga hasil akhirnya dapat memberikan gambaran yang lebih jelas mengenai kondisi sosial ekonomi dan kriminalitas di setiap provinsi di Indonesia.

RUMUSAN MASALAH

- Bagaimana cara memperoleh data dari sumber resmi seperti contoh BPS secara otomatis dan legal menggunakan teknik web scraping?
- Apa saja tahapan yang diperlukan untuk membersihkan, menggabungkan, dan mentransformasikan data agar siap digunakan dalam format CSV yang terstruktur dan dapat dianalisa?
- Bagaimana cara menyajikan data hasil scraping dalam bentuk visualisasi dan analisis yang dapat memberikan insight terhadap kondisi sosial ekonomi di Indonesia?
- Apa tantangan yang dialami dalam proses scraping dan bagaimana solusi yang dapat diterapkan untuk mengatasi masalah tersebut?

TUJUAN

- Mengotomatisasi pengambilan data dari situs resmi menggunakan teknik web scraping dengan bahasa python supaya proses akuisisi data menjadi efisien bisa digunakan kembali, dan bebas dari kesalahan yang terjadi secara manual (human error).
- Membersihkan dan menggabungkan data dalam format CSV
- Menyediakan dataset final dalam format CSV yang siap digunakan untuk analisis lanjutan seperti clustering, korelasi, atau machine learning dasar.
- Mensimulasikan skenario nyata di mana scraping menjadi satu-satunya cara untuk memperoleh data yang tidak tersedia dalam format unduhan langsung.

MANFAAT

- Meningkatkan keterampilan mahasiswa dalam mengakses dan mengolah data dari sumber resmi.
- Menyediakan dataset baru yang terintegrasi dan dapat digunakan sebagai analisis lanjutan.
- Menjadi referensi dan insight bagi mahasiswa lain yang ingin belajar scraping dari situs resmi melalui dokumentasi kode dan laporan yang sudah dilakukan dengan jelas.

LANDASAN TEORI

Data Wrangling

dikenal juga sebagai data minging adalah suatu proses mengubah dan mempersiapkan data mentah menjadi format yang lebih terstruktur dan siap dianalisis lebih lanjut

Scraping

Proses pengambilan (ekstraksi) data secara otomatis dari suatu sumber digital. Bertujuan utama untuk pengumpulan data dan menganalisis lebih lanjut data yang di dapat. Scraping banyak jenisnya yaitu web scraping, API scraping, screen scraping, image scraping, video/audio scraping, dan document scraping

Indikator sosial ekonomi

Indikator sosial ekonomi digunakan untuk mengukur kondisi dan kesejahteraan masyarakat. IPM (Indeks Pembangunan Manusia), tingkat pengangguran, tingkat kemiskinan yang buruk bisa mengakibatkan meningkatnya jumlah tindak pidana.

Sumber Data

Dalam proses data wrangling, sumber data dapat berasal dari berbagai tempat seperti database, file (CSV, Excel, JSON), API maupun hasil web scraping. Sumber-sumber ini menyediakan data mentah yang kemudian dibersihkan, diubah, dan disiapkan agar siap digunakan untuk analisis.

Raw data yang kami peroleh dari 4 sumber yaitu :

BPS (pdf)	= Indeks Pembangunan Manusia (IPM)
Kaggle (csv)	= Tingkat Pengangguran
BPS (csv)	= Jumlah Tindak Pidana
Open data Jabar (csv)	= Penduduk Miskin

METODOLOGI

Alat dan Lingkungan

Penelitian ini dilakukan menggunakan Python di Jupyter Notebook dengan berbagai pustaka pendukung, seperti Pandas dan NumPy untuk pengolahan data, Matplotlib dan Seaborn untuk visualisasi, Requests dan BeautifulSoup untuk web scraping, serta OpenPyXL untuk penyimpanan data Excel. Seluruh proses dijalankan pada laptop Windows dengan Python 3.x, dan hasil akhir disimpan dalam file Excel untuk analisis lebih lanjut.

Proses Scraping

Tahapan web scraping dilakukan untuk mengumpulkan data sosial ekonomi dan kriminalitas dari situs resmi. Prosesnya meliputi pengambilan HTML menggunakan Requests, parsing struktur halaman dengan BeautifulSoup, mengekstraksi tabel, menyusunnya ke dalam DataFrame Pandas, lalu menyimpan hasilnya dalam CSV untuk tahap cleaning. Seluruh data dari berbagai sumber kemudian digabungkan menjadi satu dataset utama untuk proses wrangling.

Pembersihan dan Transformasi Data

Tahapan pembersihan dan transformasi data dilakukan untuk memastikan data siap dianalisis.
Langkah-langkah utamanya meliputi:
Menghapus missing value, menstandarisasi nama kolom, dan mengonversi tipe data ke format yang tepat. Data non-numerik seperti simbol khusus pada angka dibersihkan agar dapat dianalisis secara matematis. Proses juga mencakup normalisasi nilai dan validasi akhir untuk memastikan tidak ada duplikat, konsistensi jumlah data, dan kesesuaian dengan rentang nilai yang sahih.
Seluruh tahapan ini menghasilkan dataset yang bersih dan terstruktur yang kemudian disimpan sebagai file Excel untuk proses visualisasi dan analisis lebih lanjut.

Integrasi dan Penyimpanan

Proses integrasi dilakukan dengan menggabungkan berbagai dataset (TPT, IPM, kemiskinan, dan kriminal) ke dalam satu tabel utama menggunakan kolom 'provinsi' sebagai kunci. Tahapan ini mencakup pengecekan kesesuaian jumlah baris, penghapusan kolom redundan, serta pembersihan akhir untuk memastikan semua data dalam format numerik yang siap digunakan. Dataset terintegrasi yang dihasilkan kemudian disimpan dalam file Excel dan menjadi dasar untuk proses visualisasi serta analisis korelasi pada tahap selanjutnya.

IMPLEMENTASI CODE

Ekstrak PDF dari BPS untuk sumber data 1

```python
import pdfplumber
import pandas as pd

pdf_path = "/content/statistik-indonesia-2025.pdf"
output_csv = "ipm sudah ekstrak.csv"

target_page_pdf = 364

extracted_data = []

print(f"Mengekstrak Data Tabel dari halaman file PDF ke-{target_page_pdf}...")

try:
    with pdfplumber.open(pdf_path) as pdf:
        if target_page_pdf <= len(pdf.pages):
            page = pdf.pages[target_page_pdf - 1]
            table = page.extract_table()

            if table:
                for row in table:
                    if row and len(row) >= 4:
                        provinsi = row[0]
                        nilai_2023 = row[3]

                        if not provinsi:
                            continue

                        if "Provinsi" in provinsi or "Province" in provinsi:
                            continue

                        print(f"Mengambil: {provinsi} | {nilai_2023}")
                        extracted_data.append([provinsi, nilai_2023])
            else:
                print("Tabel tidak terdeteksi secara otomatis pada halaman ini.")
        else:
            print("Nomor halaman melebihi jumlah halaman PDF.")

    if extracted_data:
        df = pd.DataFrame(extracted_data, columns=["Provinsi", "2023"])
        df.to_csv(output_csv, index=False)
        print(f"\nSukses! {len(extracted_data)} baris data tersimpan di '{output_csv}'.")
    else:
        print("\nTidak ada data yang berhasil diekstrak. Cek kembali nomor halaman.")

except Exception as e:
    print(f"Terjadi Error: {e}")
```

Hasil Output

```
Mengekstrak Data Tabel dari halaman file PDF ke-364...
Mengambil: (1) | (4)
Mengambil: Aceh | 74,70
Mengambil: Sumatera Utara | 75,13
Mengambil: Sumatera Barat | 75,64
Mengambil: Riau | 74,95
Mengambil: Jambi | 73,73
Mengambil: Sumatera Selatan | 73,18
Mengambil: Bengkulu | 74,30
Mengambil: Lampung | 72,48
Mengambil: Kepulauan Bangka Belitung | 74,09
Mengambil: Kepulauan Riau | 79,08
Mengambil: DKI Jakarta | 83,55
Mengambil: Jawa Barat | 74,24
Mengambil: Jawa Tengah | 73,39
Mengambil: D.I. Yogyakarta | 81,09
Mengambil: Jawa Timur | 74,65
Mengambil: Banten | 75,77
Mengambil: Bali | 78,01
Mengambil: Nusa Tenggara Barat | 72,37
Mengambil: Nusa Tenggara Timur | 68,40
Mengambil: Kalimantan Barat | 70,47
Mengambil: w
/
Kalimantan Tengah | 73,73
...
Mengambil: Papua Pegunungan | 53,45
Mengambil: Indonesia | 74,39

Sukses! 40 baris data tersimpan di 'ipm sudah ekstrak.csv'.
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Scrap PDF

```python
import pandas as pd
import re

input_csv = "ipm sudah ekstrak.csv"
output_csv = "ipm sudah scrap.csv"

try:
    df = pd.read_csv(input_csv)

    def bersihkan_nama(text):
        if not isinstance(text, str):
            return text

        text = text.replace('\n', ' ')
        match = re.search(r'([A-Z].*)', text)

        if match:
            clean_text = match.group(1)
            clean_text = re.sub(r'[^\w\s.]', '', clean_text)

            return clean_text.strip()
        else:
            return text

    nama_kolom_provinsi = df.columns[0]
    print("Memproses pembersihan data...")
    df[nama_kolom_provinsi] = df[nama_kolom_provinsi].apply(bersihkan_nama)
    df = df[df[nama_kolom_provinsi].str.contains(r'[a-zA-Z]', na=False)]

    kolom_angka = df.columns[1]
    if df[kolom_angka].dtype == object:
        df[kolom_angka] = df[kolom_angka].astype(str).str.replace(',', '.', regex=False)

    df.to_csv(output_csv, index=False)
    print("-" * 30)
    print(f"Data berhasil dibersihkan! Disimpan ke: {output_csv}")
    print("-" * 30)

    print(df.head(50))

except Exception as e:
    print(f"Terjadi Error: {e}")
```

Hasil Output

```
Memproses pembersihan data...
------------------------------
Data berhasil dibersihkan! Disimpan ke: ipm sudah scrap.csv
------------------------------
                     Provinsi   2023
1                        Aceh  74.70
2              Sumatera Utara  75.13
3              Sumatera Barat  75.64
4                        Riau  74.95
5                       Jambi  73.73
6             Sumatera Selatan  73.18
7                     Bengkulu  74.30
8                     Lampung   72.48
9   Kepulauan Bangka Belitung  74.09
10             Kepulauan Riau  79.08
11                DKI Jakarta  83.55
12                Jawa Barat   74.24
13                Jawa Tengah  73.39
14            D.I. Yogyakarta  81.09
15                Jawa Timur   74.65
16                    Banten   75.77
17                      Bali   78.01
18        Nusa Tenggara Barat  72.37
19        Nusa Tenggara Timur  68.40
20           Kalimantan Barat  70.47
...
36              Papua Selatan  68.24
37              Papua Tengah   59.44
38            Papua Pegunungan  53.45
39                   Indonesia  74.39
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Cleaning Sumber Data 1

```python
import pandas as pd

# load data
df = pd.read_csv("ipm sudah scrap.csv")

# rename provinsi sesuai aturan
df["Provinsi"] = df["Provinsi"].str.upper()

df["Provinsi"] = df["Provinsi"].replace({
    "D.I. YOGYAKARTA": "DI YOGYAKARTA",
    "KEPULAUAN RIAU": "KEP. RIAU",
    "KEPULAUAN BANGKA BELITUNG": "KEP. BANGKA BELITUNG"
})

# daftar provinsi yang harus dihapus
hapus_prov = [
    "INDONESIA",
    "PAPUA TENGAH",
    "PAPUA SELATAN",
    "PAPUA PEGUNUNGAN",
    "PAPUA BARAT DAYA"
]

df = df[~df["Provinsi"].isin(hapus_prov)]

# ubah nama kolom 2023 menjadi IPM_2023
df = df.rename(columns={"2023": "IPM_2023"})

# simpan output
df.to_csv("IPM sudah cleaning.csv", index=False)

print("Selesai! File tersimpan sebagai 'IPM sudah cleaning.csv'")
```

Hasil Output

```
Selesai! File tersimpan sebagai 'IPM sudah cleaning.csv'
```

Sumber data ke 2

```python
import pandas as pd
from google.colab import files
uploaded = files.upload()

#Baca file melewati header non-data
df = pd.read_csv('data 2 tingkat pengangguran.csv', skiprows=3)

#Atur nama kolom biar rapi
df.columns = ['Provinsi', 'Februari_2023', 'Agustus_2023', 'Tahunan']

#Bersihkan nama provinsi dari spasi yg berlebih
df['Provinsi'] = df['Provinsi'].str.strip()

#Pastikan semua provinsi tetap ada, termasuk ACEH dan yang datanya kosong
df = df[df['Provinsi'].notna()]

#Hitung rata-rata tahunan dari Februari dan Agustus
def hitung_rata_rata(row):
    try:
        feb = float(row['Februari_2023'])
        agu = float(row['Agustus_2023'])
        return round((feb + agu) / 2, 2)
    except:
        return None

df['Rata_rata_Tahunan'] = df.apply(hitung_rata_rata, axis=1)

#Simpan ke CSV baru hanya dengan kolom Provinsi dan Rata-rata
output_df = df[['Provinsi', 'Rata_rata_Tahunan']]
output_filename = 'tingkat pengangguran sudah pre processing.csv'
output_df.to_csv(output_filename, index=False)

#Unduh file hasil dan auto download
files.download(output_filename)
```

Hasil Output

```
Saving data 2 tingkat pengangguran.csv to data 2 tingkat pengangguran.csv
```

Cleaning sumber data 2 dan rename nama

```python
import pandas as pd

# Load data
df = pd.read_csv("tingkat pengangguran sudah pre processing.csv")

# daftar provinsi yang ingin dihapus
hapus_prov = ["PAPUA BARAT DAYA", "PAPUA SELATAN", "PAPUA TENGAH"]

# cleaning
df_clean = df[
    (df["Provinsi"].str.upper() != "INDONESIA") &
    (~df["Provinsi"].str.upper().isin(hapus_prov))
].dropna()

# save output
df_clean.to_csv("tingkat pengangguran sudah pre processing part 2.csv", index=False)

print("Cleaning selesai, file tersimpan.")
```

Hasil Output

```
Cleaning selesai, file tersimpan.
```

```python
import pandas as pd

# load data
df = pd.read_csv("tingkat pengangguran sudah pre processing part 2.csv")

# rename kolom
df = df.rename(columns={"Rata_rata_Tahunan": "Tingkat_Pengangguran_2023"})

# simpan output
df.to_csv("tingkat pengangguran sudah cleaning.csv", index=False)

print("Selesai! File tersimpan sebagai 'tingkat pengangguran sudah cleaning.csv'")
```

```
Selesai! File tersimpan sebagai 'tingkat pengangguran sudah cleaning.csv'
```

Sumber data 3 dan cleaning

```python
# Unggah file CSV dari bps
from google.colab import files
uploaded = files.upload()

# Baca file tanpa menghilangkan baris ACEH
df = pd.read_csv('data 3 tindak pidana.csv', skiprows=3, header=None)

# Atur nama kolom secara manual
df.columns = ['Provinsi', 'Jumlah_Tindak_Pidana_2023']

# Bersihkan nama provinsi dari spasi ekstra
df['Provinsi'] = df['Provinsi'].astype(str).str.strip()

# Pastikan semua provinsi tetap ada, termasuk ACEH
df = df[df['Provinsi'].notna() & (df['Provinsi'] != '')]

# Simpan ke CSV baru
output_filename = 'tindak pidana sudah scrap.csv'
df.to_csv(output_filename, index=False)

# Unduh file hasil
files.download(output_filename)
```

```python
import pandas as pd

# load data
df = pd.read_csv("tindak pidana sudah scrap.csv")

# ubah METRO JAYA menjadi DKI JAKARTA
df["Provinsi"] = df["Provinsi"].replace("METRO JAYA", "DKI JAKARTA")

# hapus Indonesia
df = df[df["Provinsi"].str.upper() != "INDONESIA"]

# format ribuan pada kolom yang benar
df["Jumlah_Tindak_Pidana_2023"] = (
    df["Jumlah_Tindak_Pidana_2023"]
    .astype(str)
    .str.replace(r"[^0-9]", "", regex=True)
    .astype(int)
    .map(lambda x: f"{x:,}".replace(",", "."))
)

# simpan output
df.to_csv("tindak pidana sudah cleaning.csv", index=False)

print("Selesai! File tersimpan sebagai 'tindak pidana sudah cleaning.csv'")
```

Hasil Output

```
Saving data 3 tindak pidana.csv to data 3 tindak pidana.csv
```

```
Selesai! File tersimpan sebagai 'tindak pidana sudah cleaning.csv'
```

Sumber data 4 dan cleaning

```python
# uload CSV open data jabar
from google.colab import files
uploaded = files.upload()

#Baca file CSV
import pandas as pd

filename = list(uploaded.keys())[0]  # otomatis ambil nama file yang di upload
df = pd.read_csv('data 4 penduduk miskin.csv')

#Filter data untuk tahun 2023
df_2023 = df[df['tahun'] == 2023].reset_index(drop=True)

#Simpan hasil ke file baru
output_filename = "penduduk miskin sudah scrap.csv"
df_2023.to_csv(output_filename, index=False)

#Unduh file hasil
files.download(output_filename)
```

```python
import pandas as pd

# load data
df = pd.read_csv("penduduk miskin sudah scrap.csv")

# daftar provinsi yang ingin dihapus
hapus_prov = [
    "PAPUA TENGAH",
    "PAPUA BARAT DAYA",
    "PAPUA SELATAN",
    "PAPUA PEGUNUNGAN"
]

# hapus provinsi miss value
df = df[~df["nama_provinsi"].str.upper().isin(hapus_prov)]

# buat kolom penduduk_miskin dengan persen
df["penduduk_miskin"] = df["persentase_penduduk_miskin"].astype(str) + "%"

# hapus kolom yang diminta
df = df.drop(columns=["id", "tahun", "kode_provinsi", "persentase_penduduk_miskin", "satuan"])

# simpan output
df.to_csv("penduduk miskin pre processing.csv", index=False)

print("Cleaning selesai, file tersimpan.")
```

Hasil Output

```
Saving data 4 penduduk miskin.csv to data 4 penduduk miskin.csv
```

```
Cleaning selesai, file tersimpan.
```

Ubah kolom b1

```python
import pandas as pd

# load data
df = pd.read_csv("penduduk miskin pre processing.csv")

# rename kolom
df = df.rename(columns={"penduduk_miskin": "Penduduk_Miskin_2023"})

# simpan output
df.to_csv("penduduk miskin pre processing part 2.csv", index=False)

print("Selesai! File tersimpan sebagai 'penduduk miskin pre processing part 2.csv'")
```

```
Selesai! File tersimpan sebagai 'penduduk miskin pre processing part 2.csv'
```

```python
import pandas as pd

# load data
df = pd.read_csv("penduduk miskin pre processing part 2.csv")

# Rename 'nama_provinsi' to 'Provinsi' for consistency
df = df.rename(columns={'nama_provinsi': 'Provinsi'})

# ubah nama provinsi
df["Provinsi"] = df["Provinsi"].str.replace("KEPULAUAN RIAU", "KEP. RIAU", regex=False)
df["Provinsi"] = df["Provinsi"].str.replace("KEPULAUAN BANGKA BELITUNG", "KEP. BANGKA BELITUNG", regex=False)

# simpan output
df.to_csv("penduduk miskin sudah cleaning.csv", index=False)

print("Selesai! File tersimpan sebagai 'penduduk miskin sudah cleaning.csv'")
```

```
Selesai! File tersimpan sebagai 'penduduk miskin sudah cleaning.csv'
```

Integritas data atau Penggabungan 4 data

```python
import pandas as pd
from openpyxl import load_workbook
from openpyxl.styles import PatternFill, Font

# ====== LOAD DATA ======
df1 = pd.read_csv("penduduk miskin sudah cleaning.csv")
df2 = pd.read_csv("tindak pidana sudah cleaning.csv")
df3 = pd.read_csv("tingkat pengangguran sudah cleaning.csv")
df4 = pd.read_csv("IPM sudah cleaning.csv")

# Rename 'nama_provinsi' column in df1 to 'Provinsi' for consistent merging
df1 = df1.rename(columns={'nama_provinsi': 'Provinsi'})

# ====== GABUNG DATA ======
df = df1.merge(df2, on="Provinsi", how="outer") \
        .merge(df3, on="Provinsi", how="outer") \
        .merge(df4, on="Provinsi", how="outer")

# ====== SIMPAN KE EXCEL ======
output = "gabungan_belum_fix.xlsx"
df.to_excel(output, index=False)

# ====== FORMAT WARNA ======
wb = load_workbook(output)
ws = wb.active

# header styling
header_fill = PatternFill(start_color="4F81BD", end_color="4F81BD", fill_type="solid")
header_font = Font(color="FFFFFF", bold=True)

for cell in ws[1]:
    cell.fill = header_fill
    cell.font = header_font

# alternating row color
fill1 = PatternFill(start_color="DCE6F1", end_color="DCE6F1", fill_type="solid")

for row in ws.iter_rows(min_row=2):
    if row[0].row % 2 == 0:
        for cell in row:
            cell.fill = fill1

wb.save(output)

print("Integrasi selesai, file tersimpan:", output)
```

Hasil output

```
Integrasi selesai, file tersimpan: gabungan_belum_fix.xlsx
```

Integritas data atau Penggabungan 4 data

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re

# -------- path file (pakai path lokal yang ada) --------
file_path = "gabungan_belum_fix.xlsx" # Corrected file path

# -------- load data --------
df = pd.read_excel(file_path)

# tampilkan kolom awal untuk pengecekan cepat
print("Kolom dalam file:", list(df.columns))

# -------- helper: ubah series ke numeric dengan cleaning --------

def smart_to_numeric(s):
    """
    Bersihkan string-number menjadi numeric:
    - hilangkan '%' jika ada
    - deteksi apakah string menggunakan titik sebagai pemisah ribuan:
        pola seperti '11.916' atau '1.234.567' -> hilangkan semua titik
    - jika memakai koma sebagai desimal -> ubah ',' jadi '.'
    - fallback: coba to_numeric dengan errors='coerce'
    """
    s_orig = s.astype(str).str.strip()
    # kosongkan nilai yang benar-benar 'nan' str
    s_orig = s_orig.replace({'nan': np.nan, 'None': np.nan, 'none': np.nan})

    # hapus persen
    s = s_orig.str.replace('%', '', regex=False)

    # definisi fungsi pengecekan pola ribuan seperti 1.234 atau 12.345.678
    def is_thousands_pattern(x):
        if pd.isna(x):
            return False
        # hanya angka dan titik, tidak ada koma
        if ',' in x:
            return False
        return bool(re.match(r'^\d{1,3}(\.\d{3})+$', x))

    # cek apakah mayoritas non-null cocok pola ribuan
    non_null = s.dropna().head(500).astype(str)  # cek sample up to 500
    if len(non_null) > 0:
        matches = non_null.apply(is_thousands_pattern).sum()
        if matches / len(non_null) >= 0.6:  # ambang 60%
            # hapus semua titik (pemisah ribuan)
            s_clean = s.str.replace('.', '', regex=False)
            # setelah hapus titik, ganti koma jadi titik kalau ada
            s_clean = s_clean.str.replace(',', '.', regex=False)
            return pd.to_numeric(s_clean, errors='coerce')
```

```python
    # jika tidak dominantly thousands-pattern:
    # 1) ubah koma desimal -> titik
    s2 = s.str.replace(',', '.', regex=False)
    # 2) jika ada karakter lain non-digit/dot, buang (tapi hati-hati dengan minus dan dot)
    s2 = s2.str.replace(r'[^0-9.\-]', '', regex=True)
    return pd.to_numeric(s2, errors='coerce')


# -------- deteksi kolom yang mungkin numeric (kecuali kolom provinsi) -------
# asumsi: kolom non-numeric identitas: 'Provinsi' / 'Prov' / 'nama_provinsi', lainnya coba parse
id_cols = [c for c in df.columns if c.lower() in ("provinsi", "prov", "nama_provinsi")]
print("Kolom identitas terdeteksi (tidak akan dipaksa numeric):", id_cols)

candidate_cols = [c for c in df.columns if c not in id_cols]

# buat salinan untuk di-convert
df_clean = df.copy()

converted = {}
for col in candidate_cols:
    series_converted = smart_to_numeric(df_clean[col])
    # hitung berapa nilai non-null berhasil dikonversi
    nonnull_before = df_clean[col].notna().sum()
    nonnull_after = series_converted.notna().sum()
    converted[col] = {
        "converted_nonnull": nonnull_after,
        "original_nonnull": nonnull_before
    }
    # replace only if conversion berhasil untuk >0 nilai, simpan sebagai new col (suffix _num)
    if nonnull_after > 0:
        df_clean[col + "_num"] = series_converted
    else:
        # tidak bisa convert, biarkan saja
        df_clean[col + "_num"] = np.nan

# tampilkan ringkasan konversi
print("\nRingkasan konversi (kolom: converted_nonnull / original_nonnull):")
for k, v in converted.items():
    print(f"{k}: {v['converted_nonnull']} / {v['original_nonnull']}")

# -------- pilih kolom numeric final untuk korelasi -------
# gunakan kolom *_num yang punya >0 non-null
numeric_cols = [c for c in df_clean.columns if c.endswith("_num") and df_clean[c].notna().sum() > 0]

# jika nama kolom aslinya ingin digunakan di heatmap, kita map ke nama asli
name_map = {c: c[:-4] for c in numeric_cols}  # hapus suffix _num dalam label

print("\nKolom numeric yang dipakai untuk korelasi:", [name_map[c] for c in numeric_cols])

if len(numeric_cols) < 2:
    raise ValueError("Tidak cukup kolom numeric untuk membuat heatmap korelasi (butuh minimal 2). "
                     "Periksa hasil konversi atau struktur file.")
```

Integritas data atau Penggabungan 4 data

```python
# buat dataframe numeric untuk korelasi
df_nums = df_clean[numeric_cols].rename(columns=name_map)

# -------- korelasi --------
corr = df_nums.corr()

print("\nMatriks korelasi:")
print(corr)

# -------- PLOT: heatmap korelasi -------
plt.figure(figsize=(8 + len(df_nums.columns), 6 + len(df_nums.columns)/2))
sns.set_theme(style="white")
sns.heatmap(corr, annot=True, fmt=".2f", cmap="vlag", square=True, cbar_kws={"shrink": .8})
plt.title("Heatmap Korelasi (otomatis dari kolom numeric terdeteksi)")
plt.tight_layout()
plt.savefig("heatmap_korelasi_gabungan.png", dpi=300)
plt.show()

# -------- Tambahan: pairplot (visual korelasi bersilang) --------
# pairplot bisa berat jika banyak provinsi; kita hanya plot numeric columns
try:
    sns.pairplot(df_nums.dropna(), diag_kind="kde", plot_kws={"alpha":0.6})
    plt.suptitle("Pairplot antar variabel numeric", y=1.02)
    plt.savefig("pairplot_gabungan.png", dpi=300)
    plt.show()
except Exception as e:
    print("Pairplot gagal dibuat (mungkin karena ukuran data). Error:", e)

# -------- Statistik deskriptif -------
print("\nDeskriptif statistik (kolom numeric):")
print(df_nums.describe())

# -------- simpan versi numeric yang dipakai (opsional) -------
df_nums.to_csv("gabungan_numeric_used_for_corr.csv", index=False)
print("\nSelesai. Heatmap disimpan sebagai 'heatmap_korelasi_gabungan.png'.\n"
      "File numeric yang dipakai disimpan sebagai 'gabungan_numeric_used_for_corr.csv'.")
```

ANALISIS DAN HASIL DATA

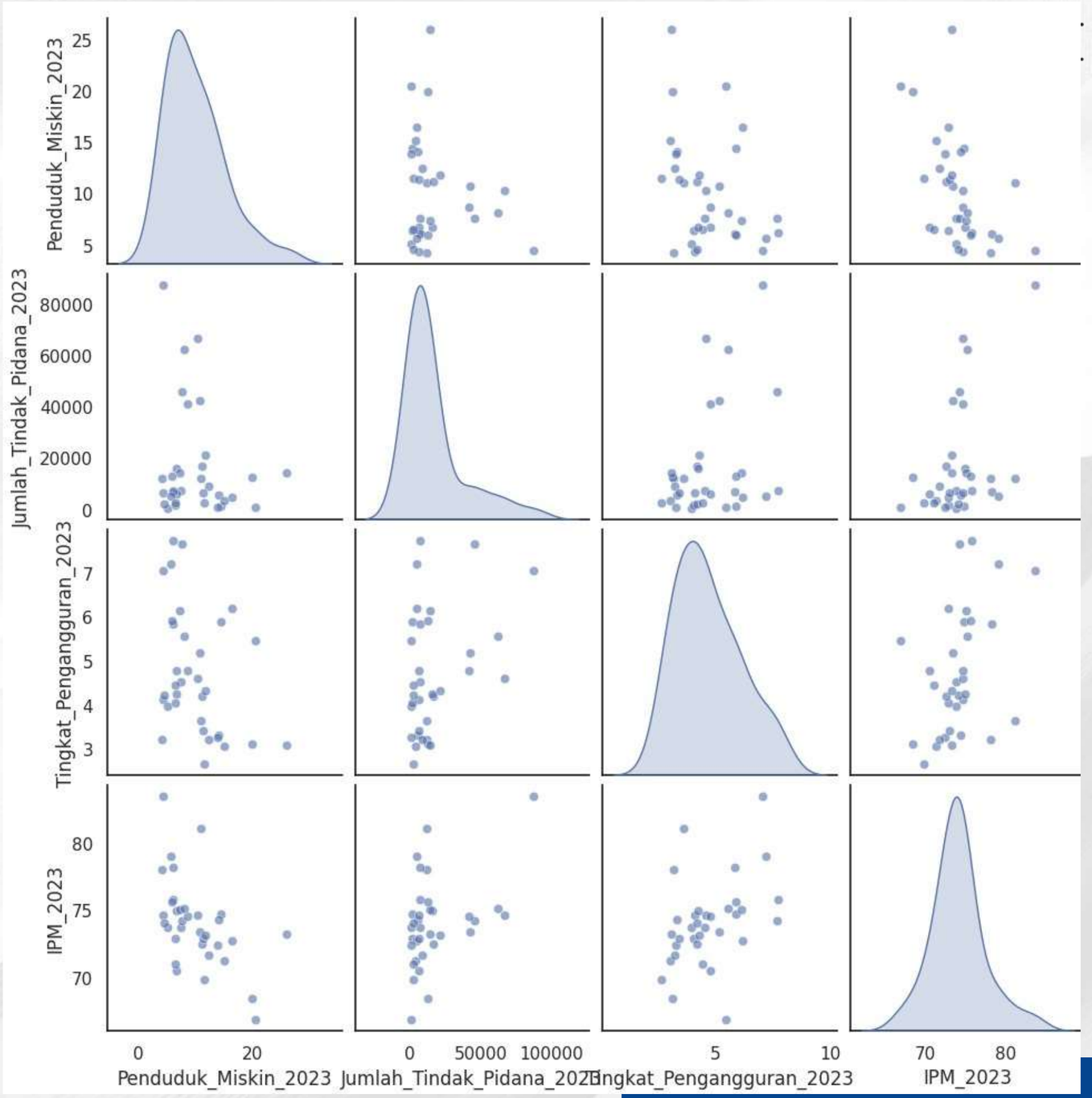
A	B	C	D	E
Provinsi	Penduduk_Miskin_2023	Jumlah_Tindak_Pidana_2023	Tingkat_Pengangguran_2023	IPM_2023
ACEH	14.45%	12.42	5.89	74.7
BALI	4.25%	11.916	3.21	78.01
BANTEN	6.17%	7.392	7.74	75.77
BENGKULU	14.04%	5.579	3.31	74.3
DI YOGYAKARTA	11.04%	12.061	3.63	81.09
DKI JAKARTA	4.44%	87.426	7.05	83.55
GORONTALO	15.15%	3.574	3.06	71.25
JAMBI	7.58%	7.432	4.52	73.73
JAWA BARAT	7.62%	45.694	7.67	74.24
JAWA TENGAH	10.77%	42.304	5.19	73.39
JAWA TIMUR	10.35%	66.741	4.61	74.65
KALIMANTAN BARAT	6.71%	6.028	4.79	70.47
KALIMANTAN SELATAN	4.29%	6.375	4.13	74.66
KALIMANTAN TENGAH	5.11%	4.42	3.97	73.73
KALIMANTAN TIMUR	6.11%	6.762	5.84	78.2
KALIMANTAN UTARA	6.45%	1.701	4.05	72.88
KEP. BANGKA BELITUNG	4.52%	2.211	4.22	74.09
KEP. RIAU	5.69%	5.074	7.21	79.08
LAMPUNG	11.11%	16.608	4.21	72.48
MALUKU	16.42%	4.741	6.2	72.75
MALUKU UTARA	6.46%	2.334	4.46	70.98
NUSA TENGGARA BARAT	13.85%	7.55	3.26	72.37
NUSA TENGGARA TIMUR	19.96%	12.692	3.12	68.4
PAPUA	26.03%	14.074	3.08	73.23
PAPUA BARAT	20.49%	6.41	5.46	66.84
RIAU	6.68%	15.777	4.24	74.95
SULAWESI BARAT	11.49%	2.679	2.66	69.8
SULAWESI SELATAN	8.7%	41.196	4.79	74.6
SULAWESI TENGAH	12.41%	8.944	3.22	71.66
SULAWESI TENGGARA	11.43%	6.276	3.41	72.94
SULAWESI UTARA	7.38%	14.265	6.14	75.04
SUMATERA BARAT	5.95%	12.722	5.92	75.64
SUMATERA SELATAN	11.78%	21.335	4.32	73.18
SUMATERA UTARA	8.15%	62.278	5.56	75.13

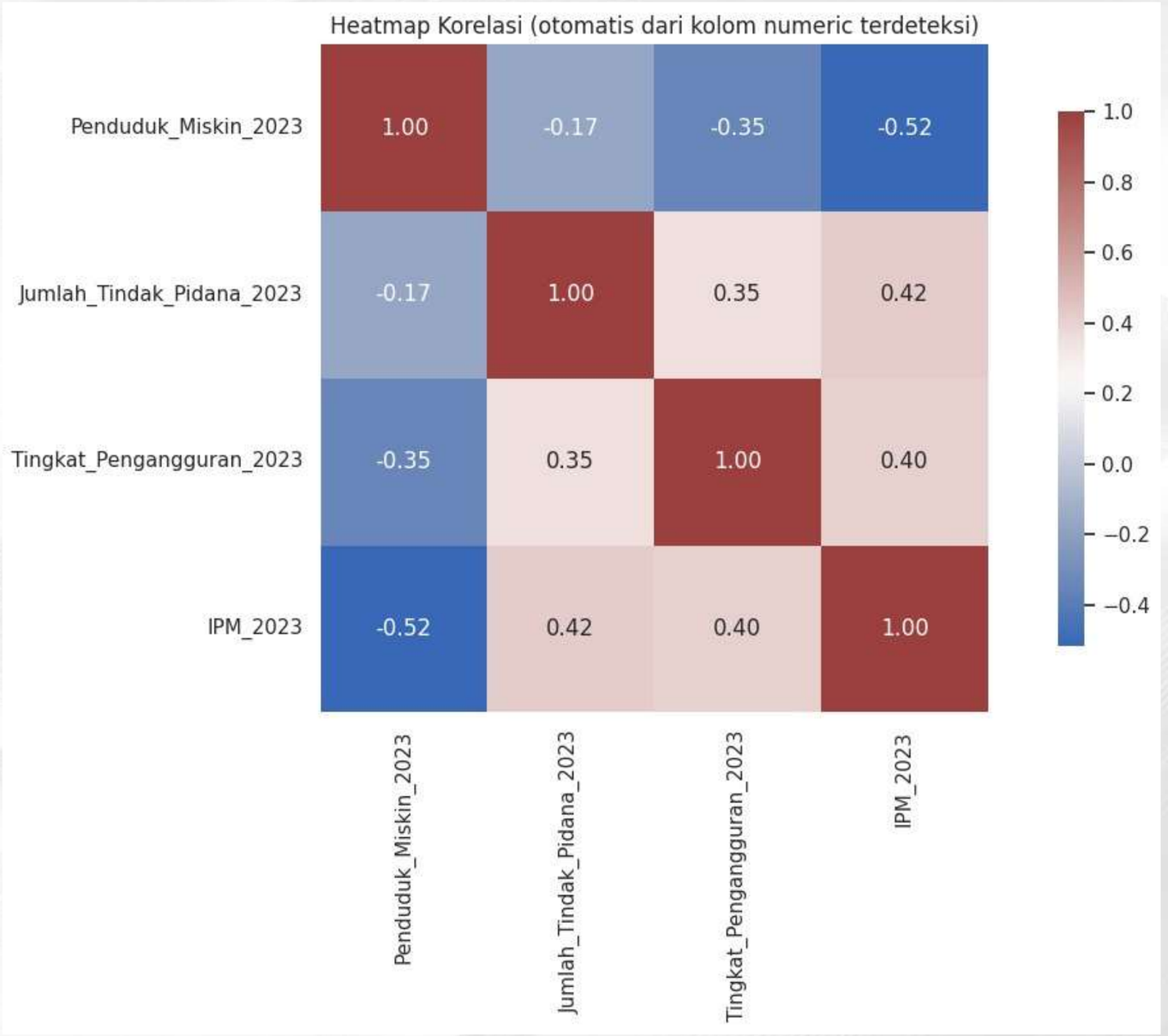
```
Ringkasan konversi (kolom: converted_nonnull / original_nonnull):
Penduduk_Miskin_2023: 34 / 34
Jumlah_Tindak_Pidana_2023: 34 / 34
Tingkat_Pengangguran_2023: 34 / 34
IPM_2023: 34 / 34
```

```
Deskriptif statistik (kolom numeric):
       Penduduk_Miskin_2023  Jumlah_Tindak_Pidana_2023  \
count             34.000000                  34.000000
mean              10.089118               16390.323529
std                5.183509               21231.863454
min                4.250000                 442.000000
25%                6.240000                3865.750000
50%                8.425000                7412.000000
75%               12.252500               15399.000000
max               26.030000               87426.000000

       Tingkat_Pengangguran_2023    IPM_2023
count                  34.000000  34.000000
mean                    4.710000  74.052353
std                     1.407609   3.291068
min                     2.660000  66.840000
25%                     3.465000  72.547500
50%                     4.390000  73.910000
75%                     5.770000  75.017500
max                     7.740000  83.550000

Selesai. Heatmap disimpan sebagai 'heatmap_korelasi_gabungan.png'.
File numeric yang dipakai disimpan sebagai 'gabungan_numeric_used_for_corr.csv'.
```


Penduduk_Miskin_2023
Jumlah_Tindak_Pidana_2023
Tingkat_Pengangguran_2023
IPM_2023
25
20
15
10
5
80000
60000
40000
20000
0
7
6
5
4
3
80
75
70
0
20
0
50000
100000
5
10
70
80
Penduduk_Miskin_2023
Jumlah_Tindak_Pidana_2023
Tingkat_Pengangguran_2023
IPM_2023




```
Deskriptif statistik (kolom numeric):
       Penduduk_Miskin_2023  Jumlah_Tindak_Pidana_2023  \
count             34.000000                  34.000000
mean              10.089118               16390.323529
std                5.183509               21231.863454
min                4.250000                 442.000000
25%                6.240000                3865.750000
50%                8.425000                7412.000000
75%               12.252500               15399.000000
max               26.030000               87426.000000

       Tingkat_Pengangguran_2023   IPM_2023
count                  34.000000  34.000000
mean                    4.710000  74.052353
std                     1.407609   3.291068
min                     2.660000  66.840000
25%                     3.465000  72.547500
50%                     4.390000  73.910000
75%                     5.770000  75.017500
max                     7.740000  83.550000

Selesai. Heatmap disimpan sebagai 'heatmap_korelasi_gabungan.png'.
File numeric yang dipakai disimpan sebagai 'gabungan_numeric_used_for_corr.csv'.
```

TERIMA KASIH