**F081**

**LAPORAN EVALUASI TENGAH SEMESTER STATISTIK KOMPUTASI**



Oleh:

Syahrial Rizky [22081010020]

Dosen Pengampu:

Yisti Vita Via S., ST., M. Kom.

# FAKULTAS ILMU KOMPUTER

# UPN “VETERAN” JAWA TIMUR

# SURABAYA 2023

1. **PENGOLAHAN DATA**
2. **Dasar Teori**

Pengolahan data dalam statistik komputasi mencakup konsep – konsep inti seperti mean, modus, median, standar deviasi, dan varians . Mean atau rata – rata merupakan salah satu ukuran tendensi sentral yang digunakan dalam statistika. Modus merupakan nilai – nilai dalam suatu set data yang paling sering muncul atau memiliki frekuensi tertinggi. Modus memberikan indikasi tentang nilai yang paling umum atau dominan dalam kumpulan suatu set data. Median adalah nilai tengah dalam suatu set data ketika nilai – nilai diurutkan dalam urutan numerik. Standar deviasi merupakan ukuran seberapa tersebar atau seberapa jauh nilai – nilai dalam suatu data set dari nilai rata rata nya. Standar deviasi memberikan informasi tentang variabilitas atau dispersi data.

Varians adalah ukuran statistik yang mengukur seberapa jauh nilai – nilai dalam suatu set data tersebar dari nilai rata - rata nya. Dengan pemahaman yang mendalam tentang konsep – konsep diatas diharap pengolahan data statistik komputasi menjadi lebih efisien dan mudah untuk dipahami.

1. **Kode Program**

import pandas as pd

import numpy as np

from statistics import mode, StatisticsError

# Membaca data dari file Excel

path = r"C:\Users\Lenovo\Desktop\statkomMeet\statkom\data\_syahrial.xlsx"

df = pd.read\_excel(path, skiprows=0, usecols='A:E', nrows=2120)

# Menghitung mean, median, modus, standar deviasi, dan varians untuk setiap kolom

for kolom in df.columns:

    # print(df.columns)

    data = df[kolom].tolist()

    # Menghitung mean

    mean = np.mean(data)

    # Menghitung median

    median = np.median(data)

    # Menghitung modus

    try:

        modus = mode(data)

    except StatisticsError:

        modus = "Tidak ada modus"

    # Menghitung standar deviasi

    std\_deviation = round(np.std(data), 2)

    # Menghitung varians

    variance = round(np.var(data), 2)

    # Menampilkan hasil statistik

    print(f"Statistik untuk Kolom '{kolom}':")

    print(f'Jumlah data: {np.sum(data)}')

    print(f"Mean: {mean:.2f}")

    print(f"Median: {median:.2f}")

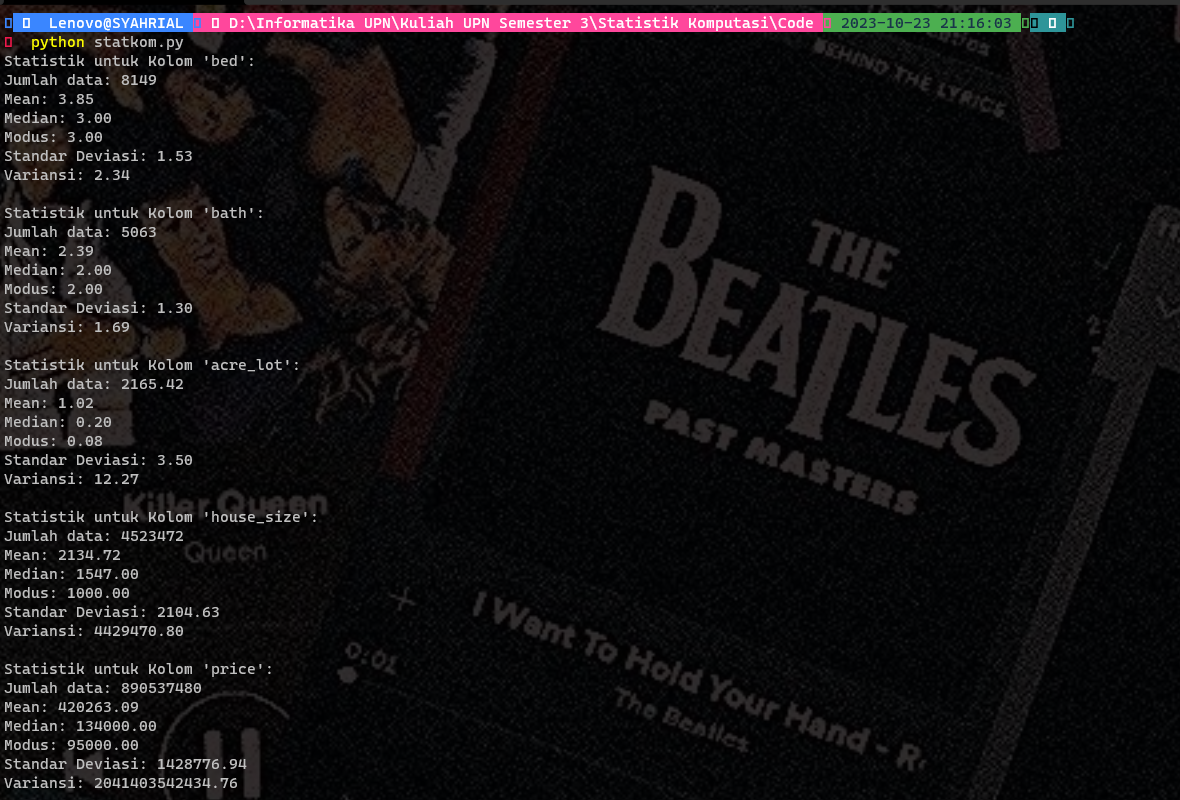
    print(f"Modus: {modus:.2f}")

    print(f"Standar Deviasi: {std\_deviation:.2f}")

    print(f"Variansi: {variance:.2f}")

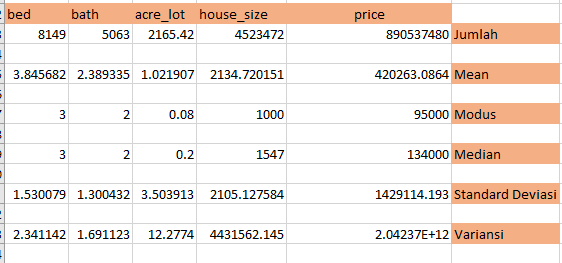
    print()

1. **Hasil Running Program**

****

1. **Analisis Hasil Running Program**

Dari hasil running program pengolahan data menunjukkan bahwa program menampilkan hasil yang hampir serupa dengan yang ada pada Excel. Hanya terdapat sedikit perbedaan angka, hal tersebut mungkin disebabkan oleh pembulatan digit desimal yang ada pada program dan Excel



1. **PENYAJIAN DATA**
2. **Dasar Teori**

Penyajian data statistik komputasi mencakup prinsip – prinsip seperti Distribusi frekuensi, Distribusi frekuensi kumulatif (lebih dari dan kurang dari), Distribusi Relatif, dan Distribusi Relatif kumulatif (lebih dari dan kurang dari). Pengertian yang komprehensif terhadap konsep-konsep ini memainkan peran krusial dalam merangkum dan memvisualisasikan data secara terstruktur, yang pada gilirannya memungkinkan analisis mendalam dalam statistik komputasi. Dengan pemahaman yang kuat mengenai distribusi frekuensi dan variasinya, presentasi data dapat dihasilkan dengan lebih informatif dan relevan, khususnya dalam konteks statistik komputasi.

1. **Kode Program**
2. **Normalisasi**

import pandas as pd

import matplotlib.pyplot as plt

path = r"D:\Informatika UPN\Kuliah UPN Semester 3\Statistik Komputasi\Code\excel\consume\_data.xlsx"

df = pd.read\_excel(path, skiprows=0, usecols='A:E', nrows=2120)

print(df.columns)

# Ganti dengan nama kolom yang sesuai

kolom\_normalisasi = ['bed', 'bath', 'acre\_lot', 'house\_size', 'price']

# Normalisasi data

for kolom in kolom\_normalisasi:

    # Menghitung nilai minimum dan maksimum dari setiap kolom

    min\_val = df[kolom].min()

    max\_val = df[kolom].max()

    # Min-Max Scaling

    df[kolom] = (df[kolom] - min\_val) / (max\_val - min\_val)

# Menampilkan hasil normalisasi

print("Data setelah dinormalisasi:")

print(df.head(30))

1. **Penyajian Data dan Grafik**

**a**import pandas as pd

import matplotlib.pyplot as plt

path = r"D:\Informatika UPN\Kuliah UPN Semester 3\Statistik Komputasi\Code\excel\consume\_data.xlsx"

df = pd.read\_excel(path, skiprows=0, usecols='A:E', nrows=2120)

print(df.columns)

# Ganti dengan nama kolom yang sesuai

kolom\_normalisasi = ['bed', 'bath', 'acre\_lot', 'house\_size', 'price']

# Normalisasi data

for kolom in kolom\_normalisasi:

    # Menghitung nilai minimum dan maksimum dari setiap kolom

    min\_val = df[kolom].min()

    max\_val = df[kolom].max()

    # Min-Max Scaling

    df[kolom] = (df[kolom] - min\_val) / (max\_val - min\_val)

# Menampilkan hasil normalisasi

print("Data setelah dinormalisasi:")

print(df.head(30))

# Mendefinisikan function

def kelompokDataBed():

    data = pd.read\_excel(path, skiprows=0, usecols='A', nrows=2120)

    interval = [1, 4, 7, 10, 14]

    # Membuat DataFrame dari data

    data\_kelompok = data.copy()

    # Mengelompokkan data ke dalam interval

    data\_kelompok['Kelompok data bed'] = pd.cut(data\_kelompok['bed'], bins=interval, labels=[

                                                '1-3', '4-7', '8-10', '11-13'], right=False)

    # Menghitung frekuensi untuk setiap kelompok

    frekuensi = data\_kelompok['Kelompok data bed'].value\_counts().sort\_index()

    # Mencari Distribusi Kumulatif Kurang dari

    distribusi\_kurang\_dari = [frekuensi.iloc[:i+1].sum()

                              for i in range(len(frekuensi))]

    # Mencari Distribusi Kumulatif Lebih dari

    distribusi\_lebih\_dari = [frekuensi.iloc[i:].sum()

                             for i in range(len(frekuensi))]

    # Mencari Distribusi Relatif

    distribusi\_relatif = frekuensi / frekuensi.sum() \* 100

    # Mencari Distribusi Relatif Kumulatif Kurang dari

    distribusi\_relatif\_kurang\_dari = [

        sum(distribusi\_relatif.iloc[:i+1]) for i in range(len(distribusi\_relatif))]

    # Mencari Distribusi Relatif Kumulatif Lebih dari

    distribusi\_relatif\_lebih\_dari = [

        sum(distribusi\_relatif.iloc[i:]) for i in range(len(distribusi\_relatif))]

    # Menampilkan hasil

    print("================ KELOMPOK DATA BED ================\n")

    print("Data dengan Kelompok dan Frekuensinya:")

    print(data\_kelompok.head(100))

    print("\nFrekuensi untuk Setiap Kelompok:")

    print(frekuensi)

    print(f"\nTotal : {frekuensi.sum()} ")

    print("\nDistribusi Kumulatif Kurang dari:")

    print(list(zip(frekuensi.index, distribusi\_kurang\_dari)))

    print("\nDistribusi Kumulatif Lebih dari:")

    print(list(zip(frekuensi.index, distribusi\_lebih\_dari)))

    print("\nDistribusi Relatif:")

    print(list(zip(frekuensi.index, distribusi\_relatif)))

    print("\nDistribusi Relatif Kumulatif Kurang dari:")

    print(list(zip(frekuensi.index, distribusi\_relatif\_kurang\_dari)))

    print("\nDistribusi Relatif Kumulatif Lebih dari:")

    print(list(zip(frekuensi.index, distribusi\_relatif\_lebih\_dari)))

    print("================================\n")

    # Membuat grafik Frekuensi kumulatif

    grafikFrekKumulatif = plt.figure(figsize=(10, 6))

    plt.bar(frekuensi.index, frekuensi.values, label='Frekuensi')

    plt.plot(frekuensi.index, distribusi\_kurang\_dari, marker='o',

             linestyle='-', color='skyblue', label='Freq. Kumulatif kurang dari')

    plt.plot(frekuensi.index, distribusi\_lebih\_dari, marker='o',

             linestyle='-', color='lime', label='Freq. Kumulatif lebih dari')

    # Menampilkan dan memberi nama

    plt.xlabel('Frekuensi')

    plt.ylabel('Nilai')

    plt.title('Grafik Data Kelompok Bed')

    plt.legend()

    plt.grid(True)

    grafikFrekKumulatif.show()

    # Membuat grafik Frekuensi Relatif

    plt.figure(figsize=(10, 6))

    plt.plot(frekuensi.index, distribusi\_relatif, marker='o',

             linestyle='-', color='red', label='Freq. Relatif')

    plt.plot(frekuensi.index, distribusi\_relatif\_kurang\_dari, marker='o',

             linestyle='-', color='orange', label='Freq. Relatif kurang dari')

    plt.plot(frekuensi.index, distribusi\_relatif\_lebih\_dari, marker='o',

             linestyle='-', color='yellow', label='Freq. Relatif lebih dari')

    # Menampilkan dan memberi nama

    plt.xlabel('Kelompok Data Bed (Relatif)')

    plt.ylabel('Nilai')

    plt.title('Grafik Data Kelompok Bed')

    plt.legend()

    plt.grid(True)

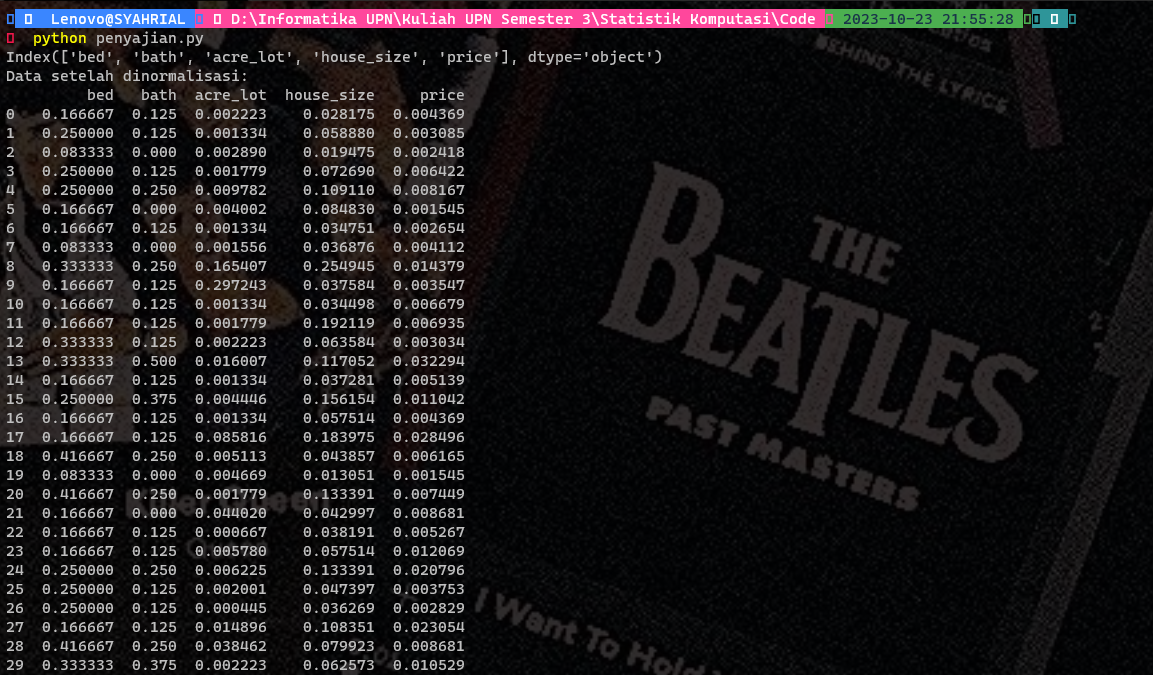
    plt.show()

# Panggil function

kelompokDataBed()

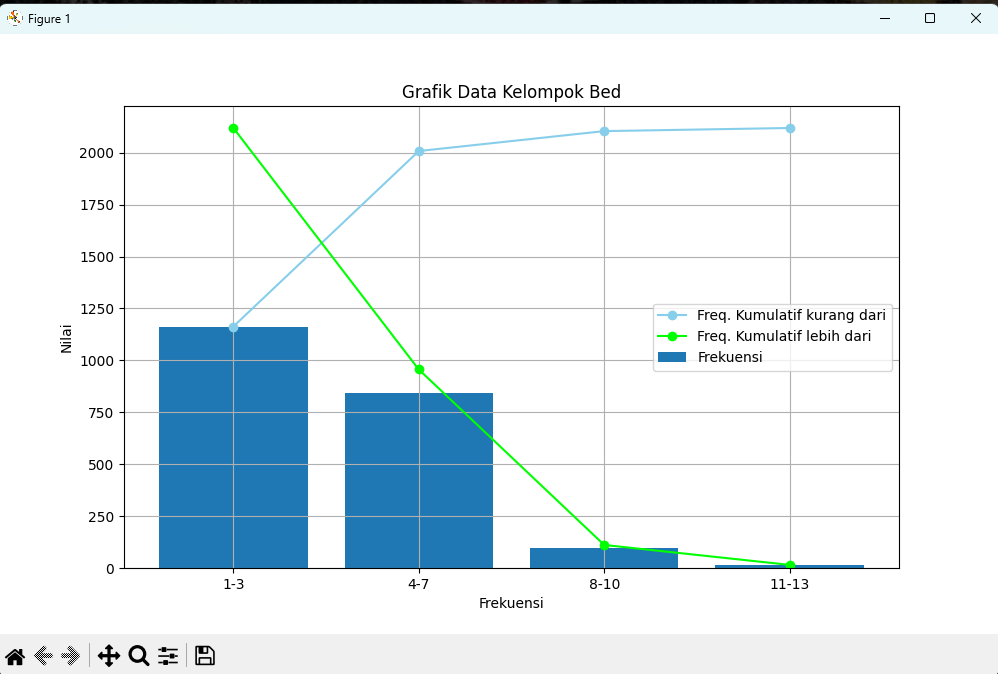
print("\n")

1. **Hasil Running Program**
2. **Hasil Normalisasi**

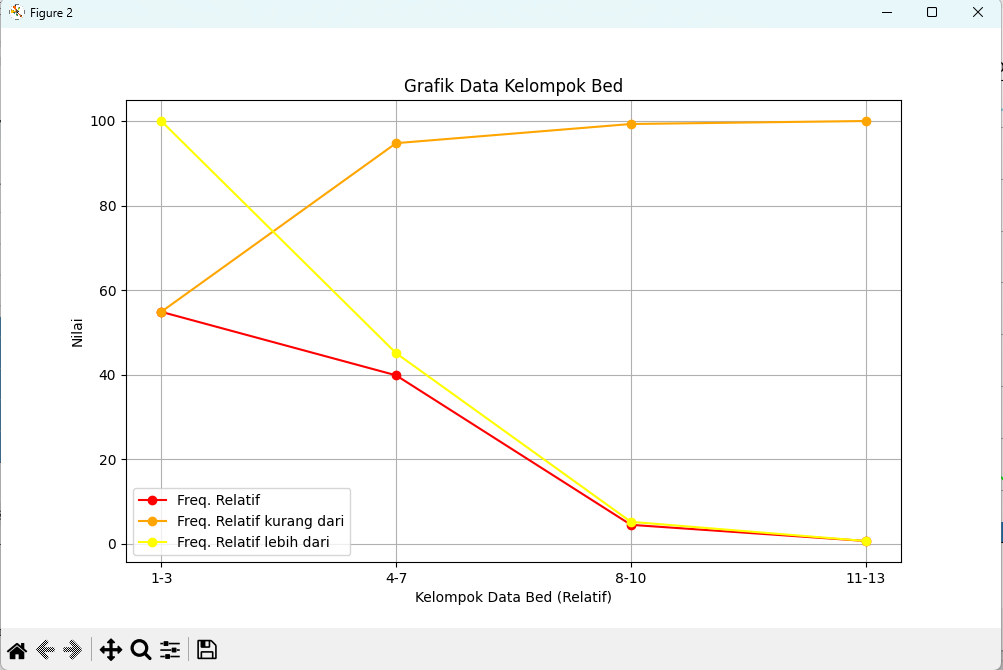
****

1. **Hasil Penyajian Data dan Grafik**

****

****

(gambar hasil grafik Frekuensi, Frek kumulatif kurang dari dan lebih dari)



(gambar hasil grafik Frekuensi Relatif, Frek Relatif kurang dari dan lebih dari)

1. **Analisis Hasil Running Program**

Analisis hasil running program penyajian data menunjukkan bahwa output yang dihasilkan dari program ini cukup konsisten dengan hasil yang ada pada excel. Dengan memisah”kan atau mengelompokkan grafik mungkin akan lebih memudahkan pembaca untuk memahami informasi yang disajikan, dengan begitu akan meningkatkan visual data dan mepermudah keterbacaan data yang disajikan, serta dapat dilakukan analisis lebih mendalam dengan lebih mudah.