Laporan Praktikum Preprocessing dan Normalisasi

Nama : Surya Dwi Satria
Kelas : C7
NIM : 434231048

Tujuan Praktikum

Melakukan preprocessing (missing value, duplikat, outlier) dan normalisasi menggunakan tiga metode: Simple Feature Scaling, Min-Max, dan Z-Score. Dataset berisi profil pelanggan ritel: usia, pendapatan, skor belanja, masa bergabung, jumlah transaksi, dan nilai keranjang rata-rata, serta kategori (gender, kota).

Langkah preprocessing:
1. Menghapus duplikat baris.
2. Mengisi nilai hilang: median untuk numerik, modus untuk kategorikal.
3. Menangani outlier dengan teknik IQR capping (clipping berdasarkan Q1±1.5*IQR).
4. One-Hot Encoding untuk variabel kategorikal.

Normalisasi/Standarisasi:
- Simple Feature Scaling: x / max(x)
- Min-Max Scaling: (x - min) / (max - min)
- Z-Score: (x - mean) / std

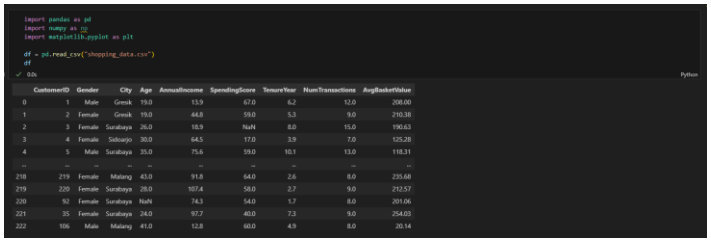
Visualisasi Data
Visualisasi histogram distribusi fitur dilakukan sebelum dan sesudah scaling untuk fitur utama: Age, AnnualIncome, SpendingScore, TenureYear.

Hasil:
- Setelah preprocessing, data bersih dari duplikasi dan nilai hilang.
- Outlier ekstrem berhasil ditahan (capping) sehingga skala fitur lebih stabil.
- Min-Max mengubah rentang ke 0..1, baik untuk model berbasis jarak.
- Z-Score menstandarkan mean≈0 dan std≈1, cocok untuk model linier/statistik.
- Simple Feature Scaling relatif sederhana dan cepat tetapi tidak mempertahankan distribusi.

Rekomendasi: pilih teknik sesuai algoritme. Untuk K-Means/KNN gunakan Min-Max; untuk regresi/logistic/PCA gunakan Z-Score.

Pengerjaan :

1. Inisialisasi library di notebook jupyter, dan memanggil data



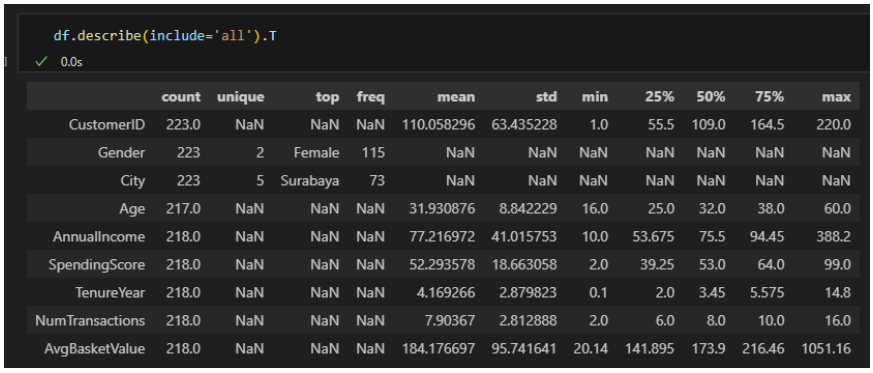
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_csv("shopping_data.csv")
df
```

0.0s

Python

	CustomerID	Gender	City	Age	AnnualIncome	SpendingScore	TenureYear	NumTransactions	AvgBasketValue
0	1	Male	Gresik	19.0	13.9	67.0	6.2	12.0	200.00
1	2	Female	Gresik	19.0	44.8	59.0	5.3	9.0	210.38
2	3	Female	Surabaya	26.0	18.9	NaN	8.0	15.0	190.63
3	4	Female	Sidoarjo	30.0	64.5	17.0	3.9	7.0	125.28
4	5	Male	Surabaya	35.0	75.6	59.0	10.1	13.0	118.31
...	...	...	...	...	...	...	...	...	...
218	219	Female	Malang	43.0	91.8	64.0	2.6	8.0	235.68
219	220	Female	Surabaya	28.0	107.4	58.0	2.7	9.0	212.57
220	92	Female	Surabaya	NaN	74.3	54.0	1.7	8.0	201.06
221	35	Female	Surabaya	24.0	97.7	40.0	7.3	9.0	254.03
222	106	Male	Malang	41.0	12.8	60.0	4.9	8.0	20.14

2. Statistik deskriftip data data shopping



```
df.describe(include='all').T
```

0.0s

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
CustomerID	223.0	NaN	NaN	NaN	110.058296	63.435228	1.0	55.5	109.0	164.5	220.0
Gender	223	2	Female	115	NaN	NaN	NaN	NaN	NaN	NaN	NaN
City	223	5	Surabaya	73	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Age	217.0	NaN	NaN	NaN	31.930876	8.842229	16.0	25.0	32.0	38.0	60.0
AnnualIncome	218.0	NaN	NaN	NaN	77.216972	41.015753	10.0	53.675	75.5	94.45	388.2
SpendingScore	218.0	NaN	NaN	NaN	52.293578	18.663058	2.0	39.25	53.0	64.0	99.0
TenureYear	218.0	NaN	NaN	NaN	4.169266	2.879823	0.1	2.0	3.45	5.575	14.8
NumTransactions	218.0	NaN	NaN	NaN	7.90367	2.812888	2.0	6.0	8.0	10.0	16.0
AvgBasketValue	218.0	NaN	NaN	NaN	184.176697	95.741641	20.14	141.895	173.9	216.46	1051.16

Penjelasan :

count → jumlah data valid (tidak NaN).

mean → rata-rata.

std → standar deviasi (sebaran data).

min → nilai terkecil.

25% → kuartil bawah (Q1).

50% → median (Q2).

75% → kuartil atas (Q3).

max → nilai terbesar.

3. Cek Mising value dan duplikat

```
df.isna().sum()
```

0.0s

```
CustomerID          0
Gender              0
City                0
Age                 6
AnnualIncome        5
SpendingScore       5
TenureYear          5
NumTransactions     5
AvgBasketValue      5
dtype: int64
```

```
df.duplicated().sum()
```

0.0s

```
np.int64(0)
```

Penjelasa : terdapat missing value di kolom kolom yang di tujukan di output dan tidak ditemukan duplikat data

4. Imputasi missing value

```
num_cols = df.select_dtypes(include=['float64','int64']).columns.tolist()
cat_cols = df.select_dtypes(include=['object']).columns.tolist()

for c in num_cols:
    df[c] = df[c].fillna(df[c].median())

for c in cat_cols:
    df[c] = df[c].fillna(df[c].mode().iloc[0])

df.isna().sum()
```

✓ 0.0s

```
CustomerID         0
Gender             0
City               0
Age                0
AnnualIncome       0
SpendingScore      0
TenureYear         0
NumTransactions    0
AvgBasketValue     0
dtype: int64
```

Penjelasan : Mengambil semua kolom yang bertipe data numerik lalu dimasukan dalam kolom list, mengambil semua kolom yang bertipe string atau text dimasukan dalam kolom list.

- df[c] = df[c].fillna(df[c].median()) → mengisi data yang kosong dengan median dari semuaa data yang ada di kolom tersebut
- df[c] = df[c].fillna(df[c].mode().iloc[0]) → mengisi data yang kosong dengan modus / atau nilai yang paling sering muncul di kolom tersebut

5. Menangani Outlier (Capping IQR)

```
def iqr_cap(series, k=1.5):
    q1 = series.quantile(0.25)
    q3 = series.quantile(0.75)
    iqr = q3 - q1
    lower = q1 - k*iqr
    upper = q3 + k*iqr
    return series.clip(lower, upper)

num_cols_wo_id = [c for c in num_cols if c != "CustomerID"]
df[num_cols_wo_id] = df[num_cols_wo_id].apply(iqr_cap)

df.describe().T
```

✓ 0.0s

	count	mean	std	min	25%	50%	75%	max
CustomerID	220.0	110.500000	63.652704	1.00000	55.7500	110.50	165.250	220.00000
Age	220.0	31.907386	8.684278	16.00000	25.0000	32.00	37.250	55.62500
AnnualIncome	220.0	75.392045	30.669780	10.00000	53.9000	75.60	94.000	154.15000
SpendingScore	220.0	52.328977	18.534872	3.37500	39.7500	53.00	64.000	99.00000
TenureYear	220.0	4.074716	2.652479	0.10000	2.0000	3.40	5.425	10.56250
NumTransactions	220.0	7.900000	2.799054	2.00000	6.0000	8.00	10.000	16.00000
AvgBasketValue	220.0	178.205705	59.308471	32.55625	142.1875	173.63	215.275	324.90625

Penjelasan :
Fungsi iqr_cap = deteksi outlier pakai rumus IQR, lalu potong (clip).
Diterapkan ke semua kolom numerik (kecuali ID).
Hasil describe() menunjukkan nilai maksimum jadi masuk akal (tidak lagi ratusan/seribuan ekstrem).

6. Encoding Kategorial

Encoding Kategorikal (One-Hot)

```
df_encoded = pd.get_dummies(df, columns=cat_cols, drop_first=True)
df_encoded.head()
```

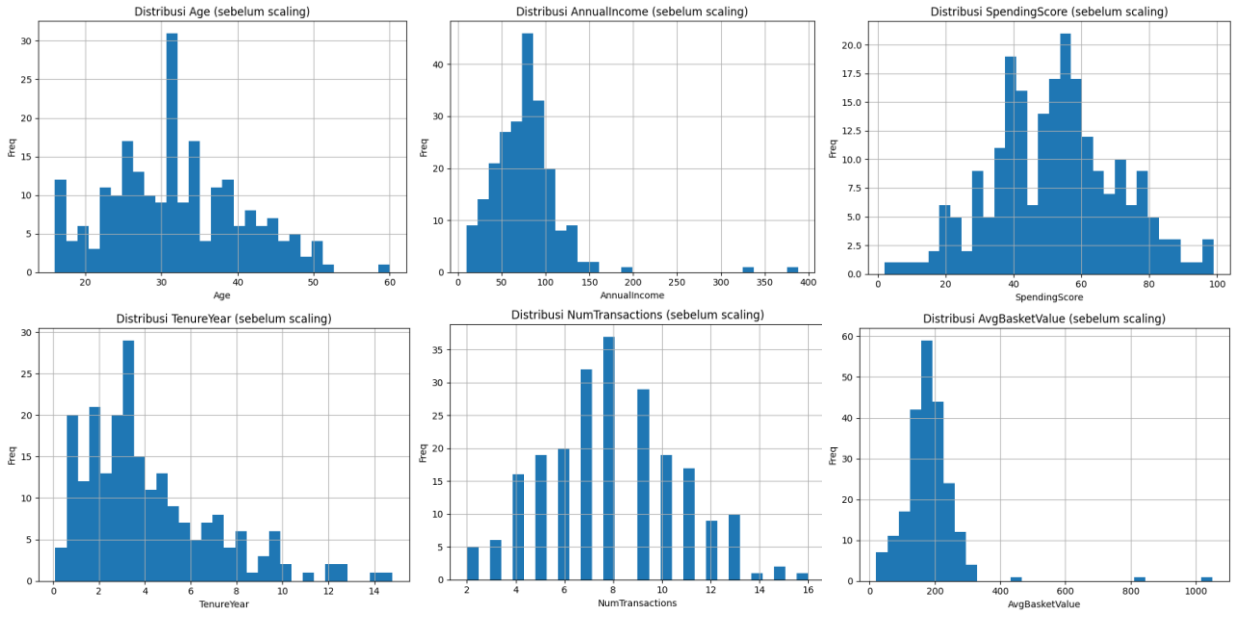
✓ 0.0s

	CustomerID	Age	AnnualIncome	SpendingScore	TenureYear	NumTransactions	AvgBasketValue	Gender_Male	City_Gresik	City_Malang	City_Sidoarjo	City_Surabaya
0	1	19.0	13.9	67.0	6.2	12.0	208.00	True	True	False	False	False
1	2	19.0	44.8	59.0	5.3	9.0	210.38	False	True	False	False	False
2	3	26.0	18.9	53.0	8.0	15.0	190.63	False	False	False	False	True
3	4	30.0	64.5	17.0	3.9	7.0	125.28	False	False	False	True	False
4	5	35.0	75.6	59.0	10.1	13.0	118.31	True	False	False	False	True

Penjelasan : mengubah kolom kategorial seperti gender dan city, menjadi binery(1/0)
Jadi missal, jika gender_male True maka gender_female false dan sebaliknya.
Lalu untuk city, misal dia tinggal di Surabaya, maka city_Surabaya True dan city_ yang lain false

7. Visualisasi Distribusi sebelum Scaling/Sebelum Normalisasi

```
plot_cols = ['Age','AnnualIncome','SpendingScore','TenureYear','NumTransactions','AvgBasketValue']
for c in plot_cols:
    plt.figure()
    df[c].hist(bins=30)
    plt.title(f"Distribusi {c} (sebelum scaling)")
    plt.xlabel(c), plt.ylabel("Freq")
    plt.tight_layout()
    plt.savefig("data.png")
    plt.show()
```



8. Simple Feature Scalling

(x/max(x))

1) Simple Feature Scaling (x / max(x))

```
def simple_feature_scaling(x: pd.Series):
    m = x.max()
    return x / m if m != 0 else x

sfs = df_encoded.copy()
for c in plot_cols:
    sfs[c] = simple_feature_scaling(sfs[c])

sfs.head()
```

✓ 0.0s

	CustomerID	Age	AnnualIncome	SpendingScore	TenureYear	NumTransactions	AvgBasketValue	Gender_Male	City_Gresik	City_Malang	City_Sidoarjo	City_Surabaya
0	1	0.341573	0.090172	0.676768	0.586982	0.7500	0.640185	True	True	False	False	False
1	2	0.341573	0.290626	0.595960	0.501775	0.5625	0.647510	False	True	False	False	False
2	3	0.467416	0.122608	0.535354	0.757396	0.9375	0.586723	False	False	False	False	True
3	4	0.539326	0.418424	0.171717	0.369231	0.4375	0.385588	False	False	False	True	False
4	5	0.629213	0.490431	0.595960	0.956213	0.8125	0.364136	True	False	False	False	True

Penjelasan :
Supaya semua fitur punya skala yang sama (0–1).
Tidak ada fitur yang mendominasi hanya karena angkanya lebih besar.
Contoh: tanpa scaling, "AnnualIncome" (max 154) bisa jauh lebih berpengaruh daripada "TenureYear" (max 14).

9. Min Max Scaling

2) Min-Max Scaling (0..1)

```
def minmax_scaling(x: pd.Series):
    mn, mx = x.min(), x.max()
    return (x - mn) / (mx - mn) if mx != mn else x

mm = df_encoded.copy()
for c in plot_cols:
    mm[c] = minmax_scaling(mm[c])

mm.head()
```

0.0s

	CustomerID	Age	AnnualIncome	SpendingScore	TenureYear	NumTransactions	AvgBasketValue	Gender_Male	City_Gresik	City_Malang	City_Sidoarjo	City_Surabaya
0	1	0.075710	0.027055	0.665359	0.583035	0.714286	0.600115	True	True	False	False	False
1	2	0.075710	0.241415	0.581699	0.497013	0.500000	0.608256	False	True	False	False	False
2	3	0.252366	0.061741	0.518954	0.755078	0.928571	0.540700	False	False	False	False	True
3	4	0.353312	0.378078	0.142484	0.363202	0.357143	0.317167	False	False	False	True	False
4	5	0.479495	0.455082	0.581699	0.955795	0.785714	0.293326	True	False	False	False	True

Penjelasan :
Ambil nilai minimum (mn) dan maksimum (mx) dari kolom.

Min-Max Scaling membuat semua kolom numerik punya rentang 0–1

10. Z Score

3) Z-Score Standardization

```
def zscore(x: pd.Series):
    mu, sd = x.mean(), x.std(ddof=0)
    return (x - mu) / sd if sd != 0 else x

zs = df_encoded.copy()
for c in plot_cols:
    zs[c] = zscore(zs[c])

zs.head()
```

0.0s

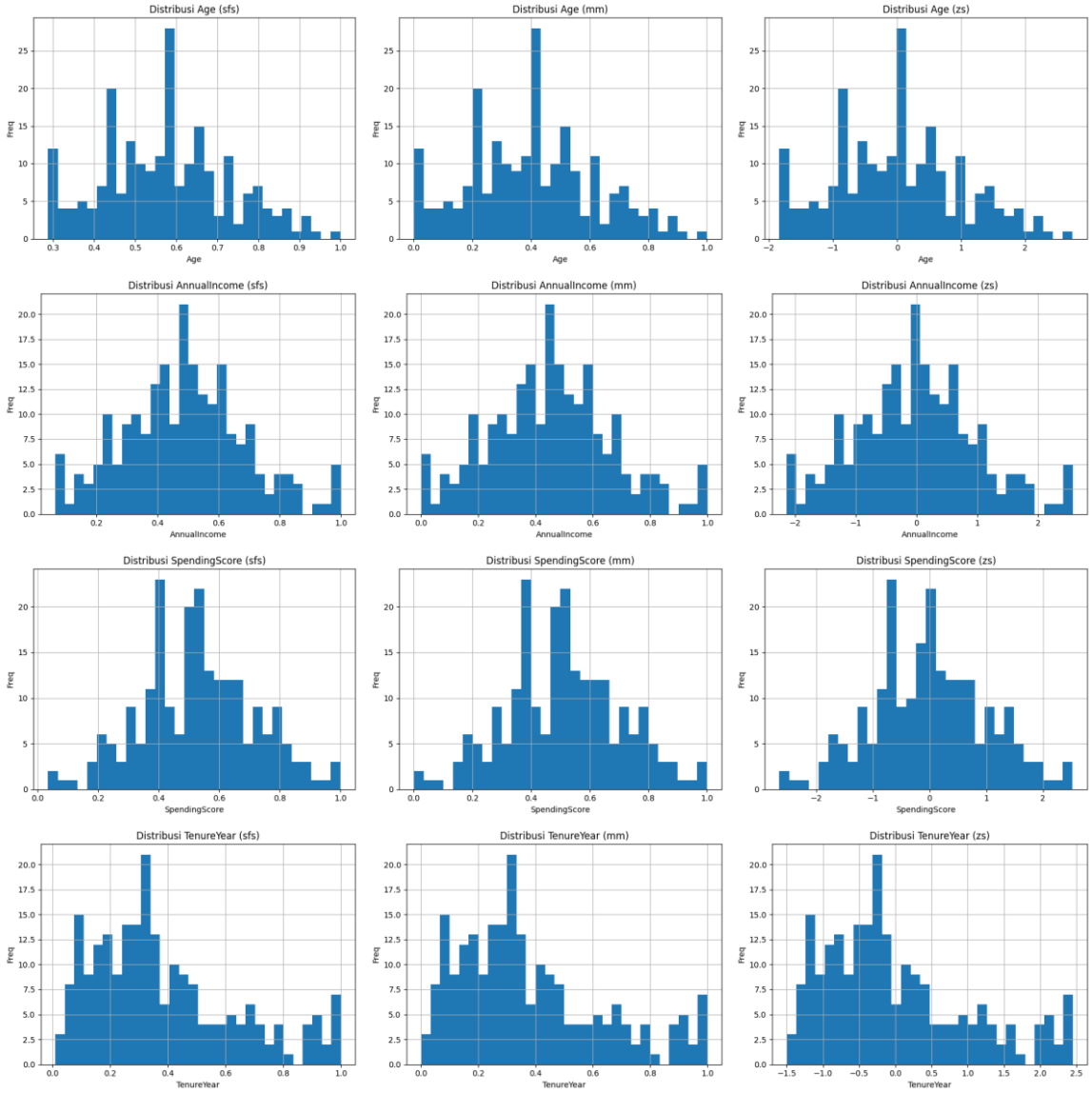
	CustomerID	Age	AnnualIncome	SpendingScore	TenureYear	NumTransactions	AvgBasketValue	Gender_Male	City_Gresik	City_Malang	City_Sidoarjo	City_Surabaya
0	1	-1.489683	-2.009544	0.793341	0.803072	1.468121	0.503507	True	True	False	False	False
1	2	-1.489683	-0.999740	0.360738	0.462993	0.393886	0.543728	False	True	False	False	False
2	3	-0.681790	-1.846145	0.036286	1.483230	2.542356	0.209964	False	False	False	False	True
3	4	-0.220138	-0.355949	-1.910428	-0.066019	-0.322270	-0.894415	False	False	False	True	False
4	5	0.356929	0.006796	0.360738	2.276747	1.826199	-1.012204	True	False	False	False	True

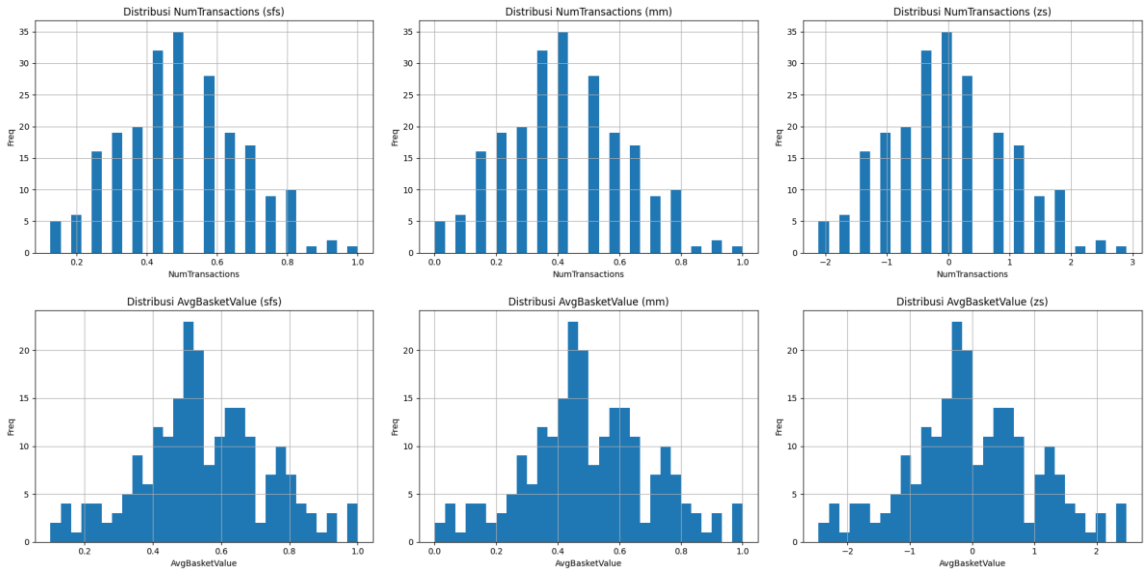
Penjelasan :
mu = x.mean() → ambil rata-rata dari kolom.
sd = x.std(ddof=0) → ambil standar deviasi dari kolom
zs = df_encoded.copy() → buat salinan dataset.
plot_cols = daftar kolom numerik (Age, AnnualIncome, SpendingScore, dsb).
Z-Score menormalkan data → rata-rata = 0, deviasi standar = 1.

11. Visualisasi Distribusi sesudah Scaling / Normalisasi

```
for c in plot_cols:
    for name, frame in [("sfs", sfs), ("mm", mm), ("zs", zs)]:
        plt.figure()
        frame[c].hist(bins=30)
        plt.title(f"Distribusi {c} ({name})")
        plt.xlabel(c); plt.ylabel("Freq")
        plt.tight_layout()
        plt.savefig("distribusi.png")
        plt.show()
```

3.0s


Distribusi Age (sfs)
Distribusi Age (mm)
Distribusi Age (zs)
Distribusi AnnualIncome (sfs)
Distribusi AnnualIncome (mm)
Distribusi AnnualIncome (zs)
Distribusi SpendingScore (sfs)
Distribusi SpendingScore (mm)
Distribusi SpendingScore (zs)
Distribusi TenureYear (sfs)
Distribusi TenureYear (mm)
Distribusi TenureYear (zs)
Freq
Age
AnnualIncome
SpendingScore
TenureYear



12. Perbandingan nilai Head

Perbandingan Nilai (Head)

```
compare = pd.DataFrame({
    'Asli_Age': df['Age'].head(10).values,
    'SFS_Age': sfs['Age'].head(10).values,
    'MM_Age': mm['Age'].head(10).values,
    'ZS_Age': zs['Age'].head(10).values
})
compare
```

✓ 0.0s

	Asli_Age	SFS_Age	MM_Age	ZS_Age
0	19.0	0.341573	0.075710	-1.489683
1	19.0	0.341573	0.075710	-1.489683
2	26.0	0.467416	0.252366	-0.681790
3	30.0	0.539326	0.353312	-0.220138
4	35.0	0.629213	0.479495	0.356929
5	45.0	0.808989	0.731861	1.511061
6	40.0	0.719101	0.605678	0.933995
7	31.0	0.557303	0.378549	-0.104724
8	32.0	0.575281	0.403785	0.010689
9	23.0	0.413483	0.176656	-1.028030