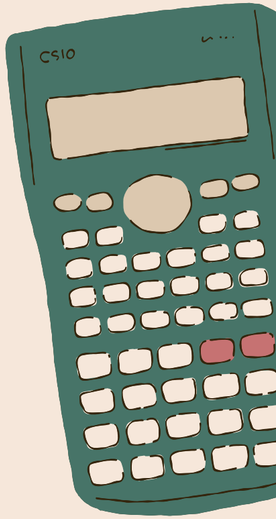
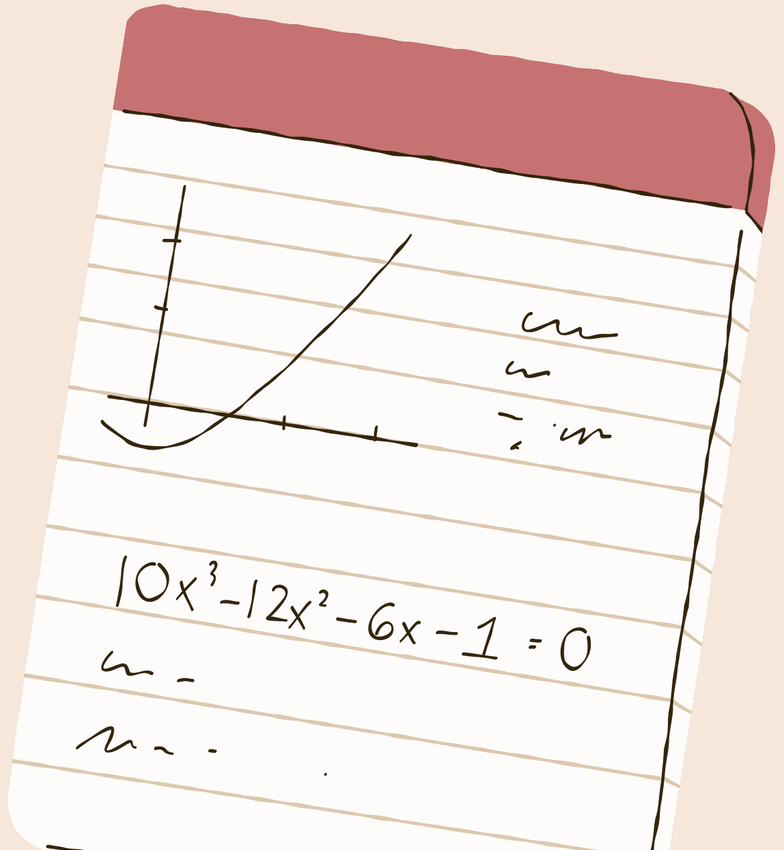
Linear Tranformation in Pandas

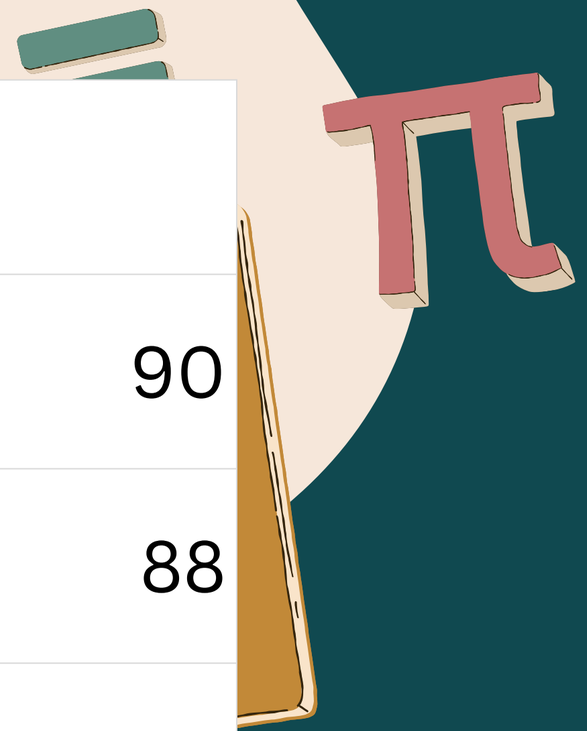
Angella Ananta Batubara
Marcella Aurelia Yatijan
Michaell Abelard Hendra





Dataset

Nama	Matematika	Bahasa	IPA
Abel	80	75	90
Angella	70	85	88
Cella	75	75	80

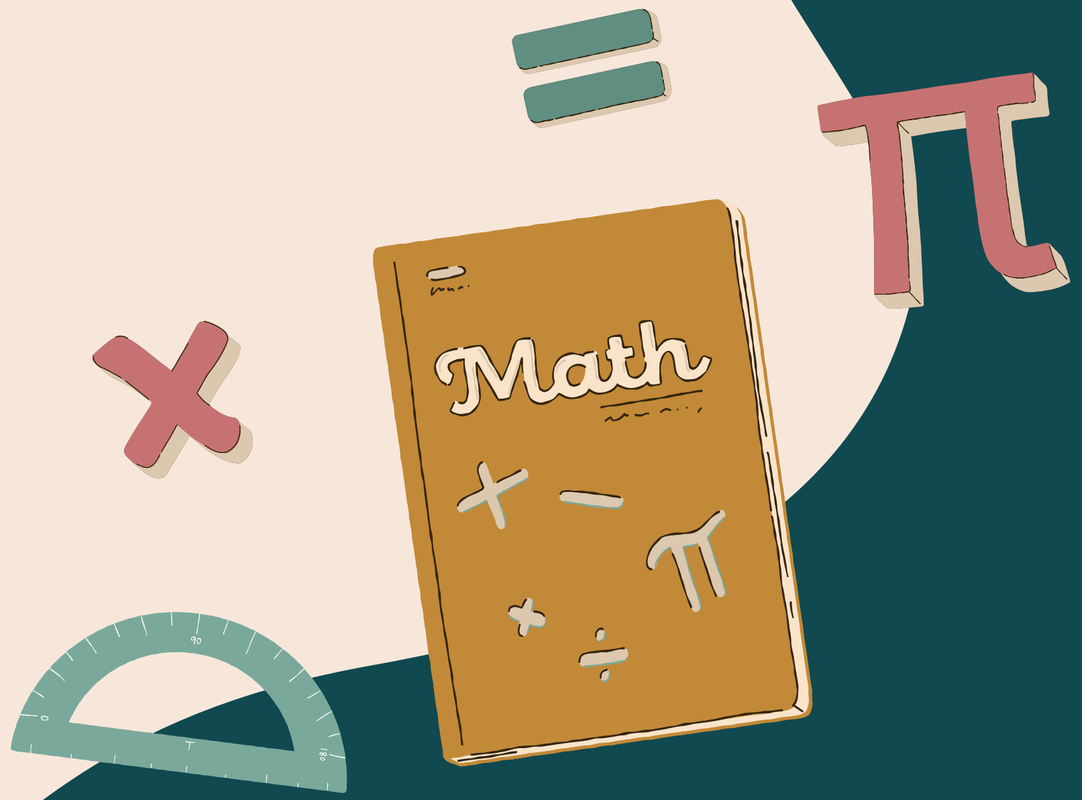


Kombinasi Linear

```
df['Skor_Akhir'] = (
    0.5 * df['Matematika'] +
    0.3 * df['Bahasa'] +
    0.2 * df['IPA'])

print(df[['Nama','Skor_Akhir']])
```

Skor Akhir=0.5 · Matematika+0.3 · Bahasa+0.2 · IPA

	Nama	Skor_Akhir
0	Abel	80.5
1	Angella	78.1
2	Cella	76.0

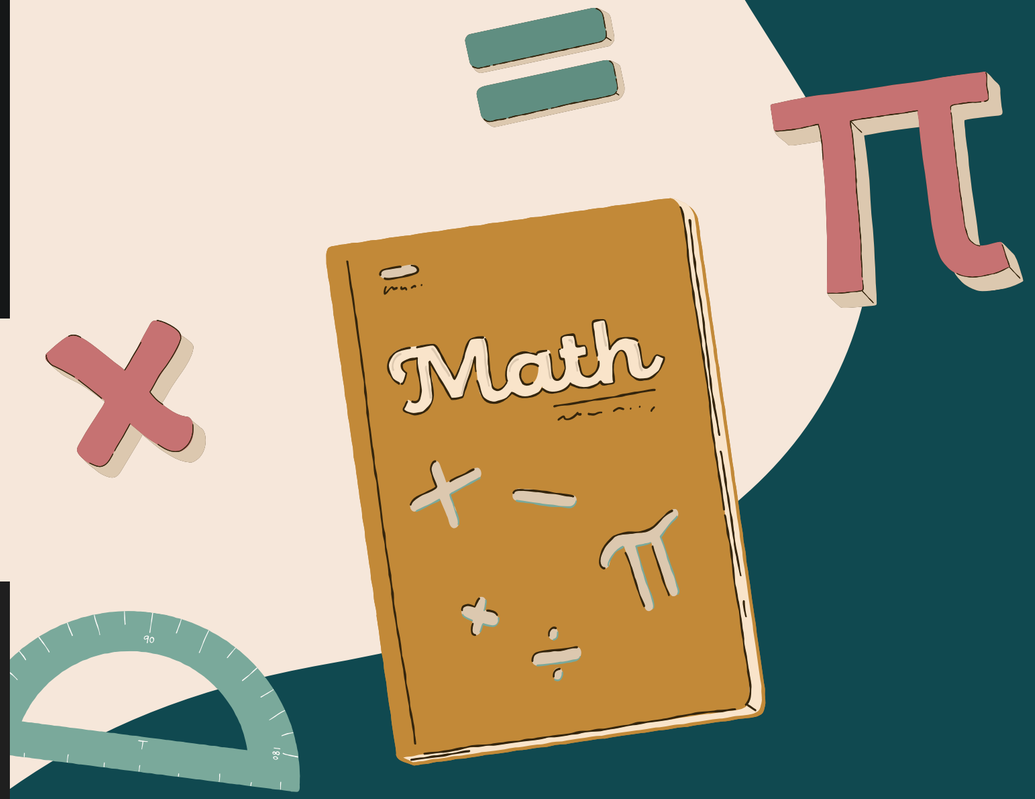


Transformasi Skalar

```python
for col in ['Matematika', 'Bahasa', 'IPA']:
    mean = df[col].mean()
    std = df[col].std()
    df[f'{col}_zscore'] = (df[col] - mean) / std

print(df[['Nama','Matematika_zscore','Bahasa_zscore','IPA_zscore']])
```

$$x' = \frac{x - \mu}{\sigma}$$

	Nama	Matematika_zscore	Bahasa_zscore	IPA_zscore
0	Abel	1.0	-0.577350	0.755929
1	Angella	-1.0	1.154701	0.377964
2	Cella	0.0	-0.577350	-1.133893

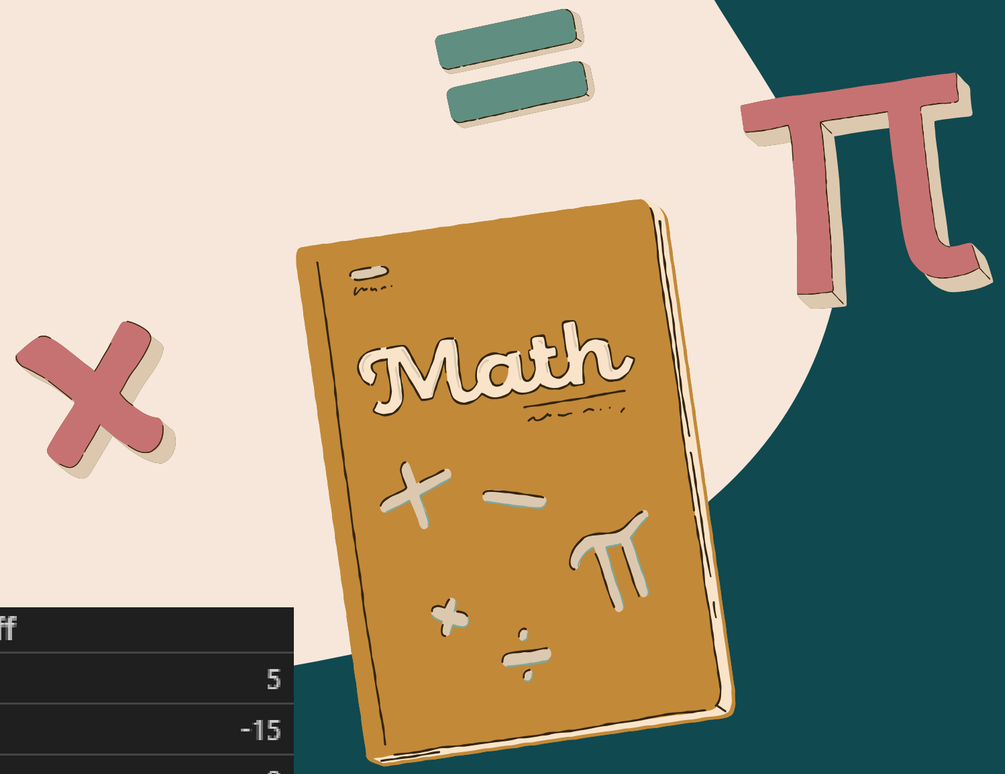


Proyeksi Linear

```
df['proj_sum'] = df['Matematika'] + df['Bahasa']
df['proj_diff'] = df['Matematika'] - df['Bahasa']

df[['Nama','proj_sum','proj_diff']]
```

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} \text{Matematika} \\ \text{Bahasa} \end{bmatrix}$$

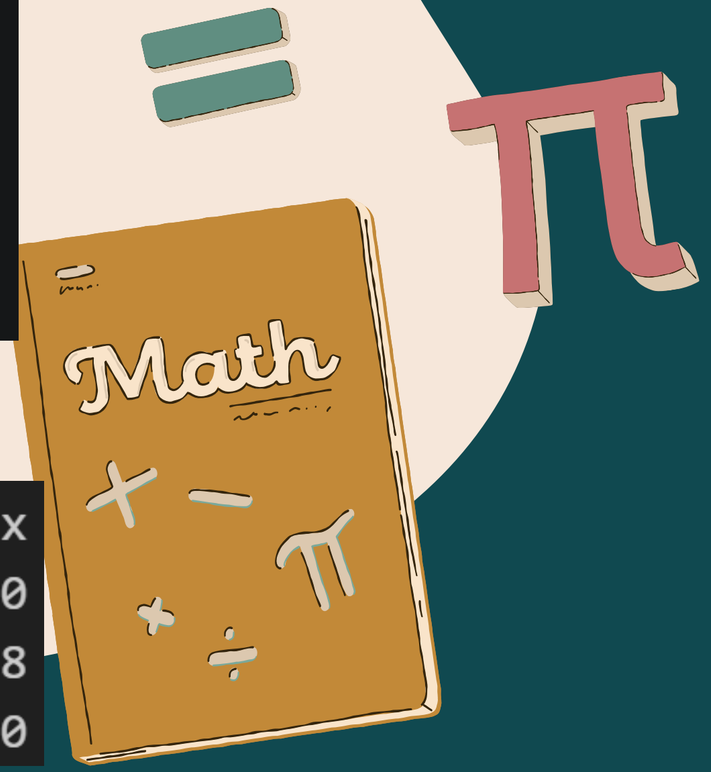
	Nama	proj_sum	proj_diff
0	Abel	155	5
1	Angella	155	-15
2	Cella	150	0



Normalisasi Min-Max

```python
for col in ['Matematika', 'Bahasa', 'IPA']:
    min_val = df[col].min()
    max_val = df[col].max()
    df[f'{col}_minmax'] = (df[col] - min_val) / (max_val - min_val)

print(df[['Nama','Matematika_minmax','Bahasa_minmax','IPA_minmax']])
```

```
    Nama  Matematika_minmax  Bahasa_minmax  IPA_minmax
0   Abel                1.0            0.0         1.0
1  Angella              0.0            1.0         0.8
2   Cella               0.5            0.0         0.0
```

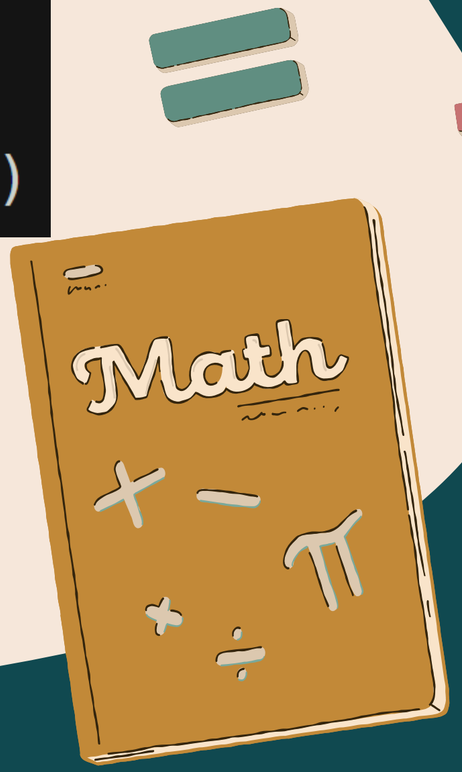


Shift

Membuat nilai lampau (lag), perbandingan antar waktu

```
df['Shifted Bahasa'] = df['Bahasa'].shift(1)
```

$$T(x)_t = x_{t-1}$$




```
Hasil transformasi linier:
      Nama  Matematika  Bahasa  IPA  Shifted Bahasa
0     Abel          80      75   90             NaN
1  Angella          70      85   88            75.0
2    Cella          75      75   80            85.0
```

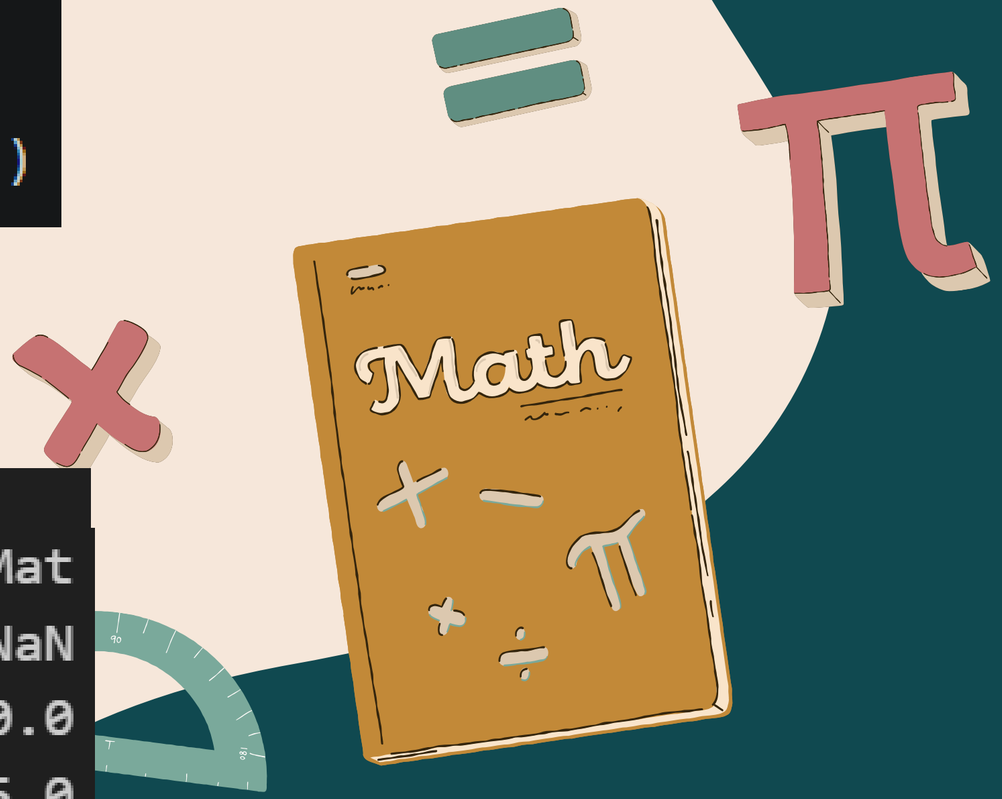
Difference

Tujuan: Menghitung perubahan antar waktu (growth, delta)

```
df['Diff Mat'] = df['Matematika'].diff()
```

$$T(x)_t = x_t - x_{t-1}$$

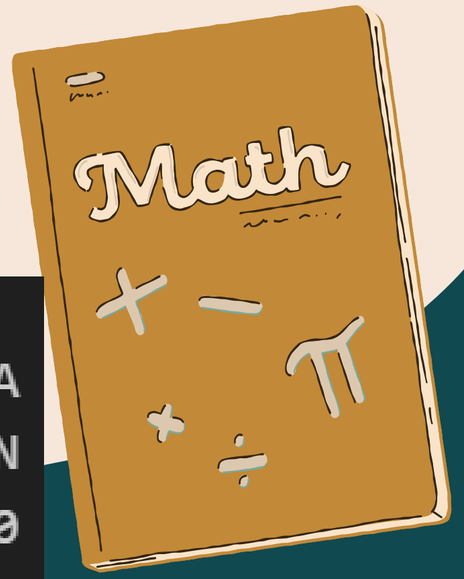
```
Hasil transformasi linier:
      Nama  Matematika  Bahasa  IPA  Diff Mat
0     Abel          80      75   90       NaN
1  Angella          70      85   88     -10.0
2    Cella          75      75   80       5.0
```



Rolling Mean

```
df['Rolling Mean IPA'] = df['IPA'].rolling(window=2).mean()
```

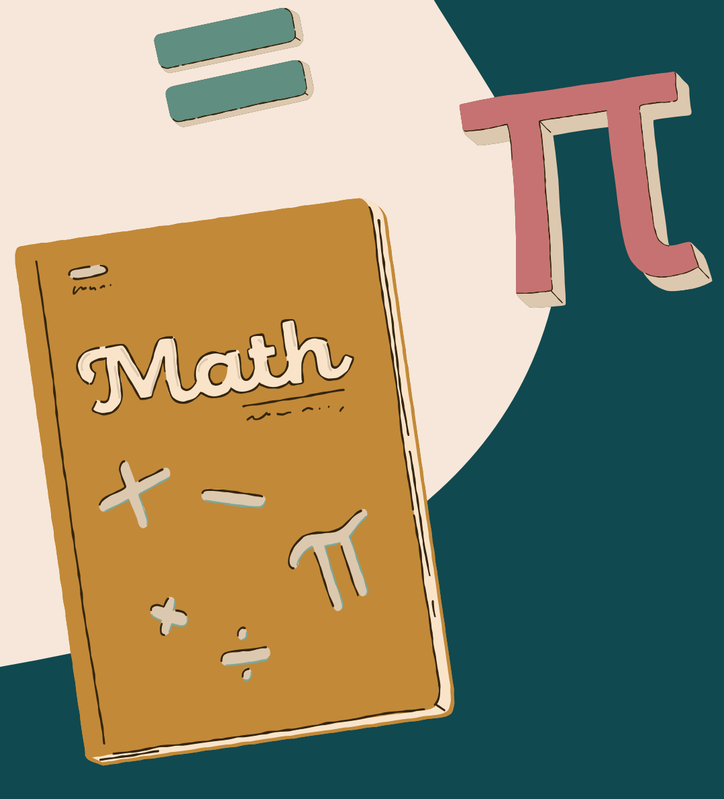
$$T(x)_t = \frac{x_t + x_{t-1}}{2}$$




```
Hasil transformasi linier:
      Nama  Matematika  Bahasa  IPA  Rolling Mean IPA
0     Abel          80      75   90              NaN
1  Angella          70      85   88             89.0
2    Cella          75      75   80             84.0
```

Model

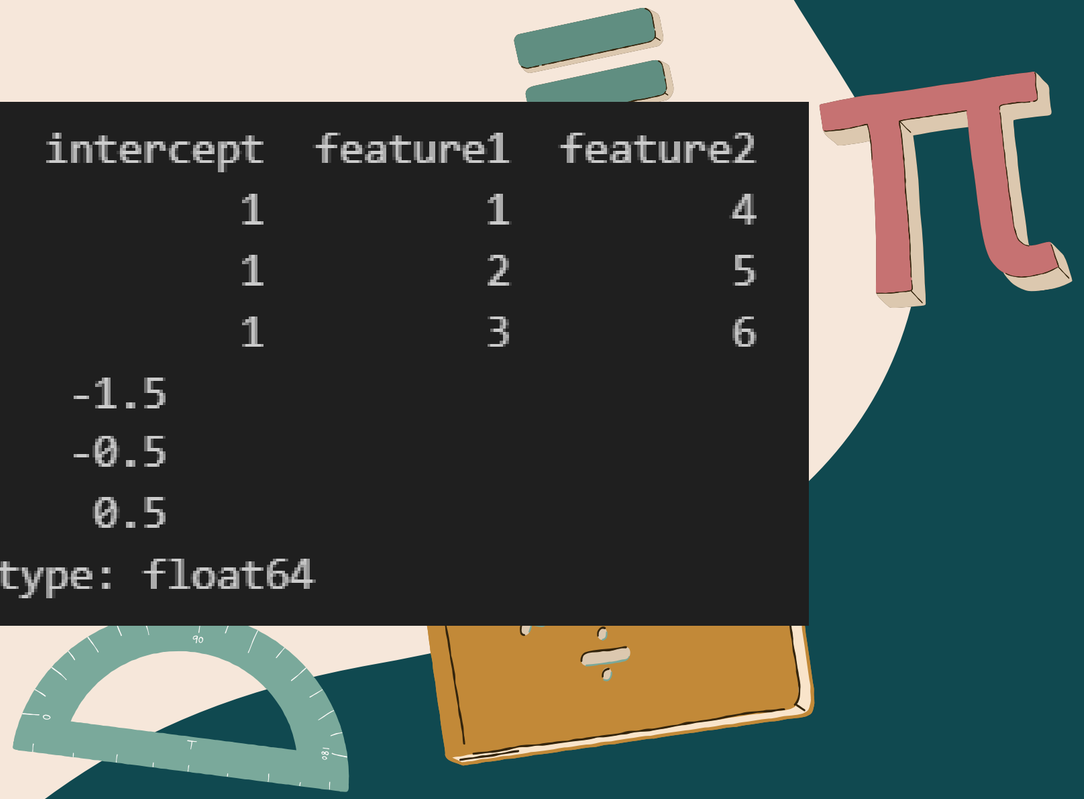
intercept	feature1	feature2
1	1	4
1	2	5
1	3	6



Dot

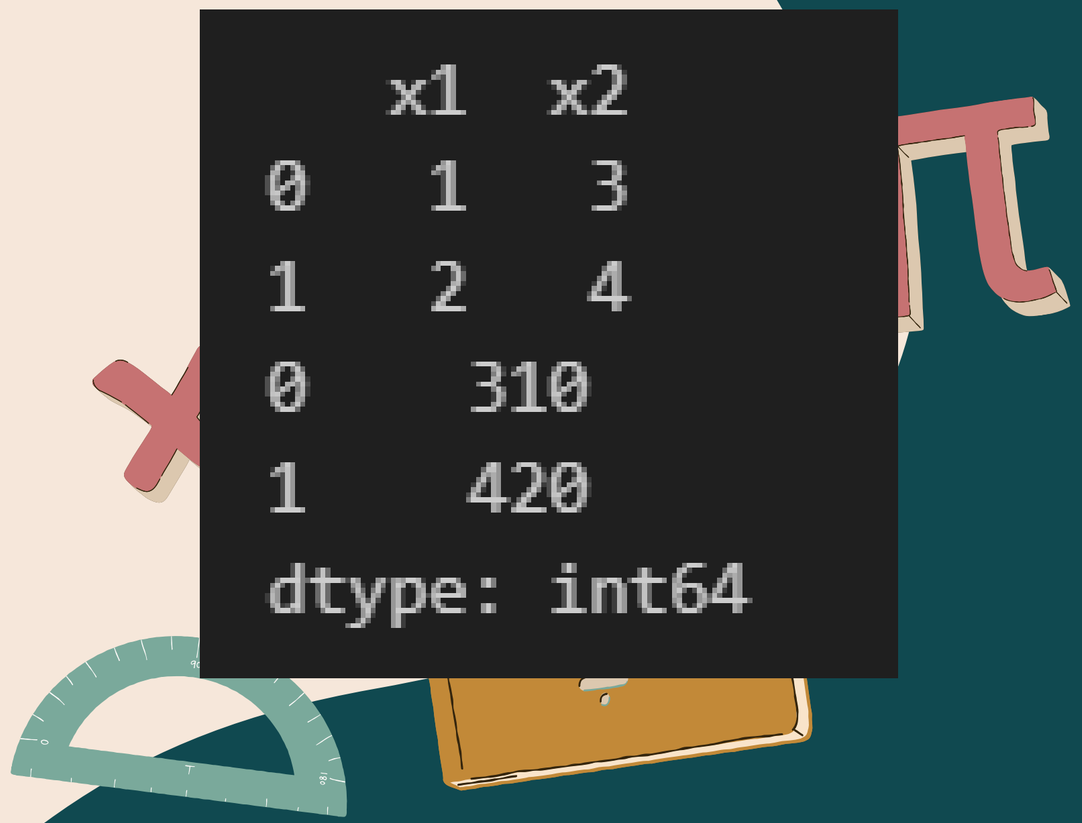
```python
X = pd.DataFrame({
    'intercept' : 1,
    'feature1' : [1,2,3],
    'feature2' : [4,5,6]
})
print(X)

beta = pd.Series({
    'intercept' : 0.5,
    'feature1' : 2.0,
    'feature2' : -1.0
})

y_pred = X.dot(beta)
print(y_pred)
```

```
   intercept  feature1  feature2
0          1         1         4
1          1         2         5
2          1         3         6
0   -1.5
1   -0.5
2    0.5
dtype: float64
```



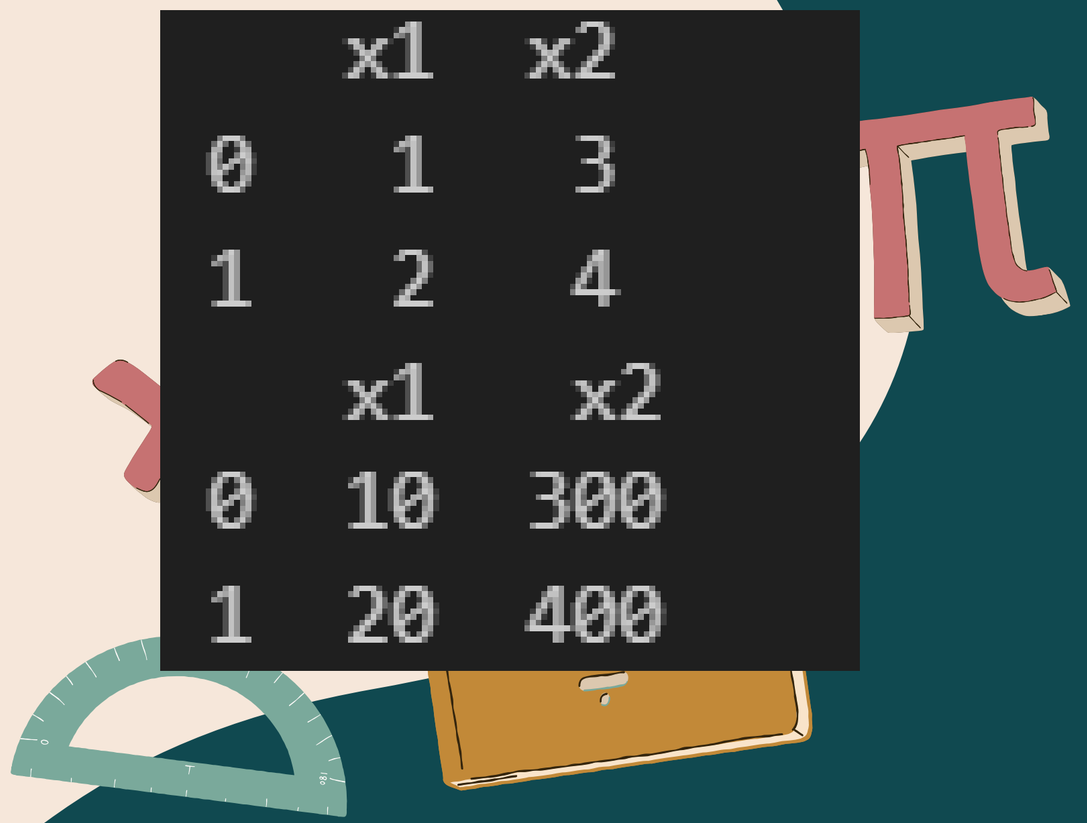
Dot Product

```python
# DataFrame (matriks X)
df = pd.DataFrame({
    'x1': [1, 2],
    'x2': [3, 4]
})
print(df)

# Series (vektor w)
w = pd.Series([10, 100], index=['x1', 'x2'])

# Dot product (X @ w)
result = df.dot(w)
print(result)
```



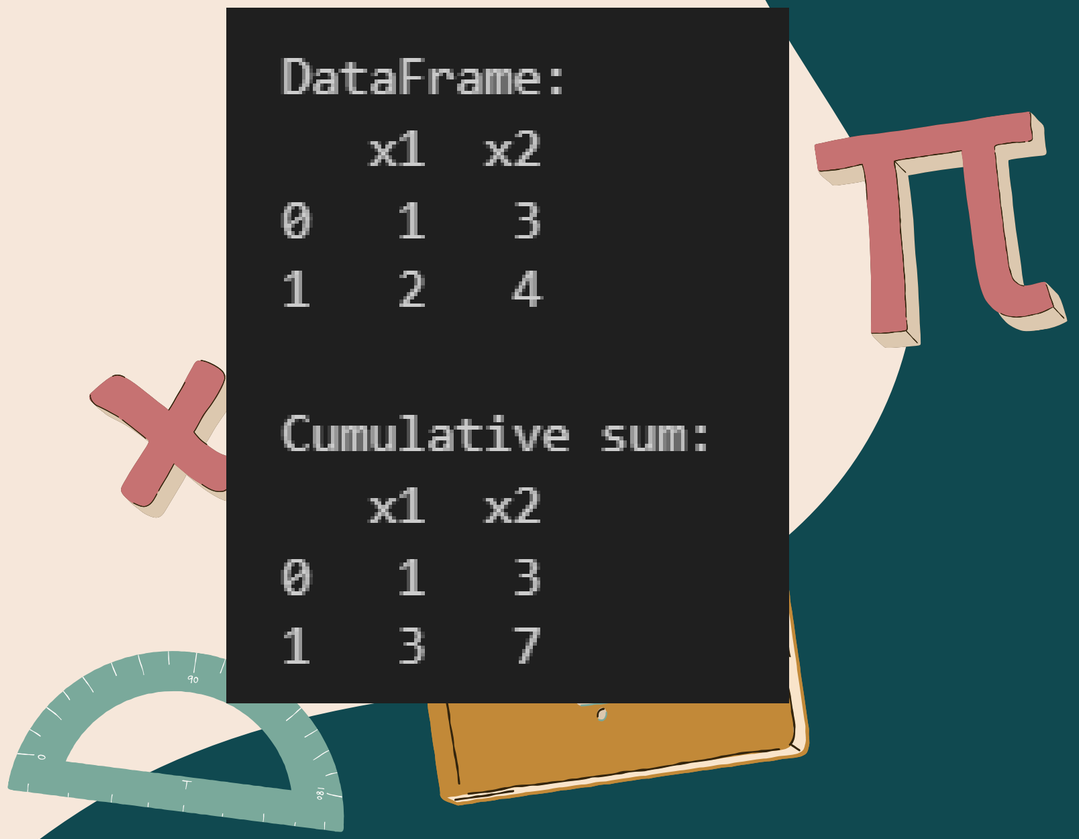
Multiply

```python
# DataFrame (matriks X)
df = pd.DataFrame({
    'x1': [1, 2],
    'x2': [3, 4]
})
print(df)

# Series (vektor w)
w = pd.Series([10, 100], index=['x1', 'x2'])

# Dot product (X @ w)
result = df.mul(w,axis=1)
print(result)
```



Cumulative Sum

```python
import pandas as pd

# DataFrame (matriks X)
df = pd.DataFrame({
    'x1': [1, 2],
    'x2': [3, 4]
})
print("DataFrame:")
print(df)

# Series (vektor bobot w)
w = pd.Series([10, 100], index=['x1', 'x2'])

# Step 3: Cumulative sum dari hasil dot product baris
cumulative = df.cumsum()
print("\nCumulative sum:")
print(cumulative)
```

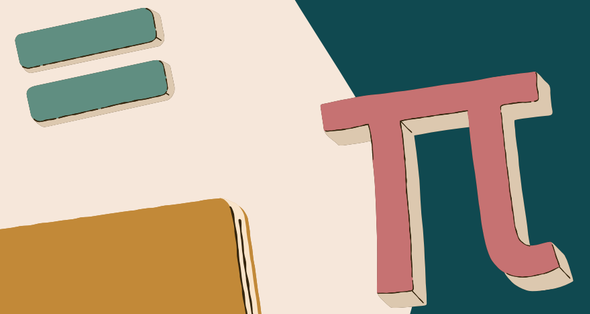


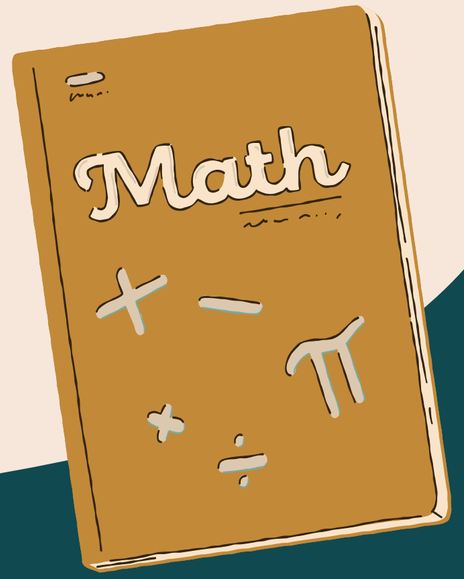
Sifat Transformasi Linear

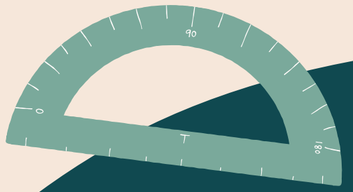
Homogenitas

```python
import numpy as np

# Matriks transformasi A
A = np.array([[2, 0],
              [0, 3]])

# Vektor x
x = np.array([1, 2])

# Skalar a
a = 5

# T(a·x)
T_ax = A @ (a * x)

# a·T(x)
a_Tx = a * (A @ x)

print("T(a·x):", T_ax)
print("a·T(x):", a_Tx)
print("Sama? :", np.allclose(T_ax, a_Tx))
```

```
...   T(a·x): [10 30]
      a·T(x): [10 30]
      Sama? : True
```



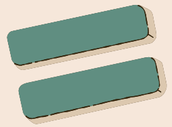


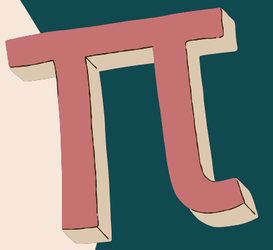


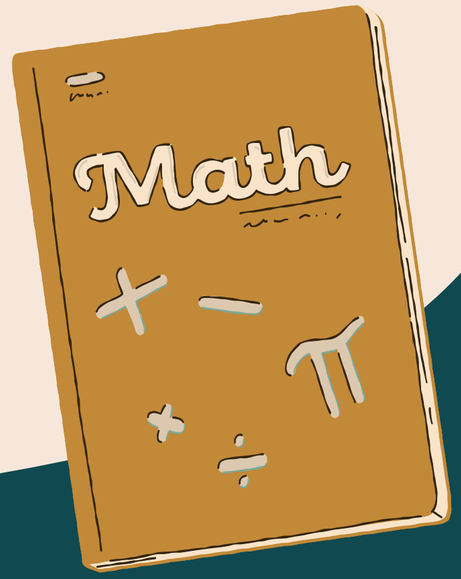
Sifat Transformasi Linear
Additivitas

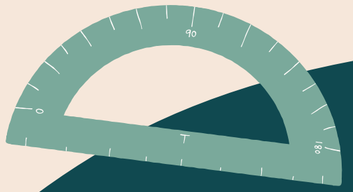
```python
# Dua vektor
x = np.array([1, 2])
y = np.array([3, 4])

# T(x + y)
T_x_plus_y = A @ (x + y)

# T(x) + T(y)
T_x = A @ x
T_y = A @ y
T_x_plus_T_y = T_x + T_y

print("T(x + y):", T_x_plus_y)
print("T(x) + T(y):", T_x_plus_T_y)
print("Sama?     :", np.allclose(T_x_plus_y, T_x_plus_T_y))
```

```
...   T(x + y): [ 8 18]
      T(x) + T(y): [ 8 18]
      Sama?     : True
```









Thank you for listening

Saranghae <3