Google Developer Groups
On Campus • Telkom University Bandung

Machine Learning 101
(Unsupervised Learning)

Ardavaa ardava-barus @rdavaa_

Muhammad Karov Ardava Barus
Machine Learning Mentor @GDGoC Tel-U,
Data Science Student | Telkom University

Presentation Link

ristek.link/ML-SLIDES-5

Google Developer Groups

Google Developer Groups
On Campus • Telkom University Bandung

Today's Topic

Core Concepts

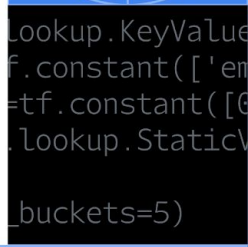
- Definition
- Main Algorithms
- Applications
- Challenges

Clustering

- Definition
- K-Means
- Agglomerative Clustering
- DBSCAN

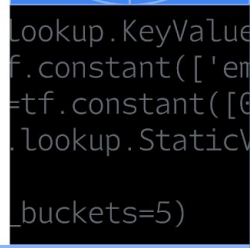
Dimensionality Reduction

- Definition
- PCA
- t-SNE

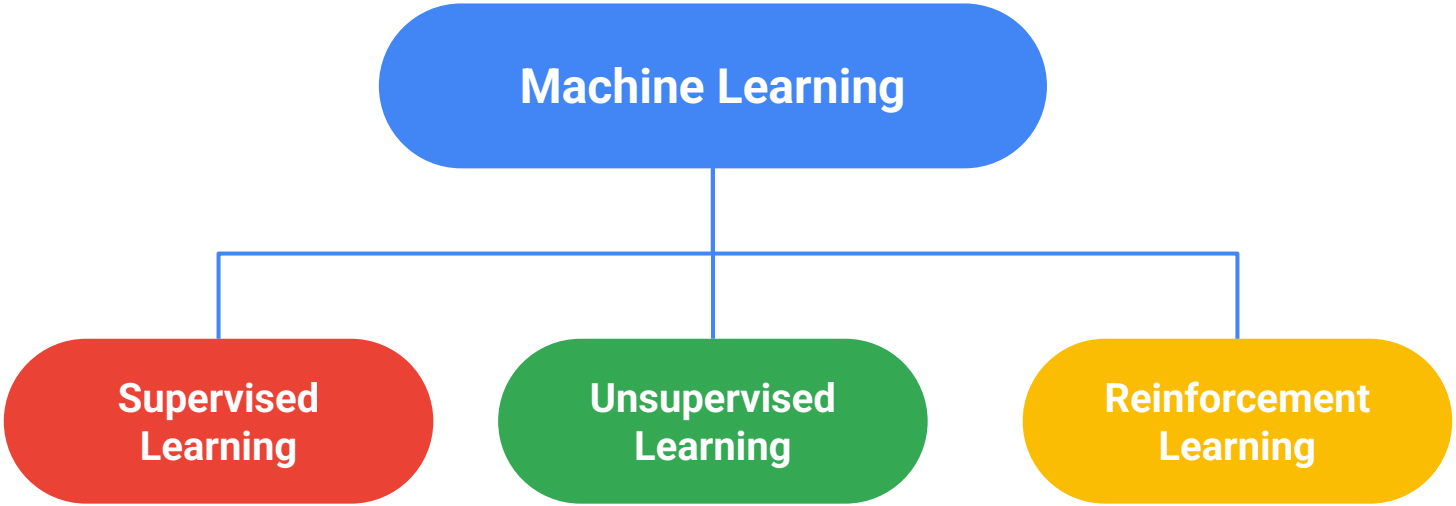


Google Developer Groups
On Campus • Telkom University Bandung

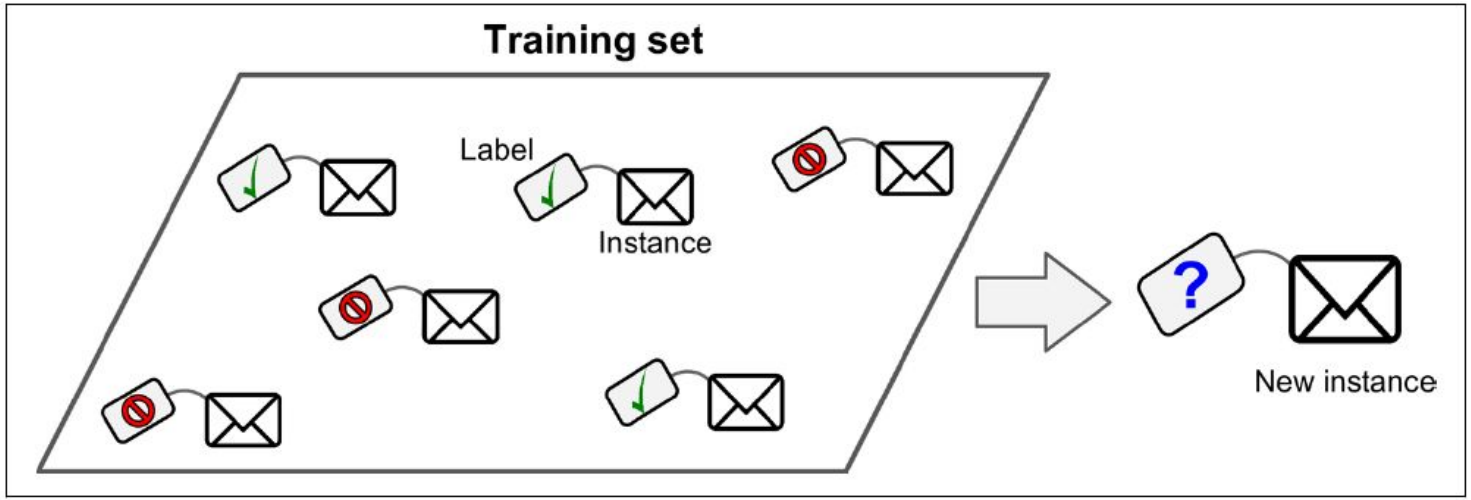
Let’s recap Supervised Learning!



Recap: Supervised Learning



Recap: Supervised Learning



Recap: Supervised Learning

Regression

Dalam Machine Learning, regresi adalah metode untuk memprediksi nilai kontinu dari variabel dependen berdasarkan hubungan dengan variabel independen.

Recap: Supervised Learning

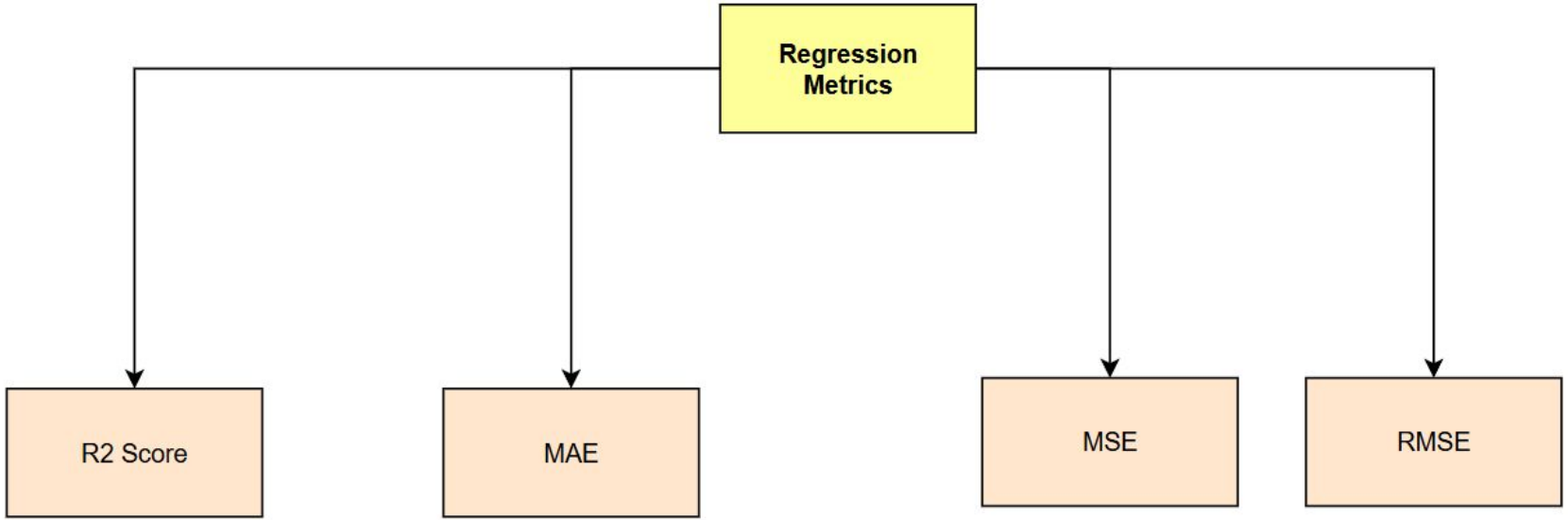
Classification

Dalam Machine Learning, klasifikasi adalah metode untuk memprediksi kategori atau kelas dari variabel dependen berdasarkan hubungan dengan variabel independen.

Classification: a Machine Learning technique to identify the category of new observations based on training data.

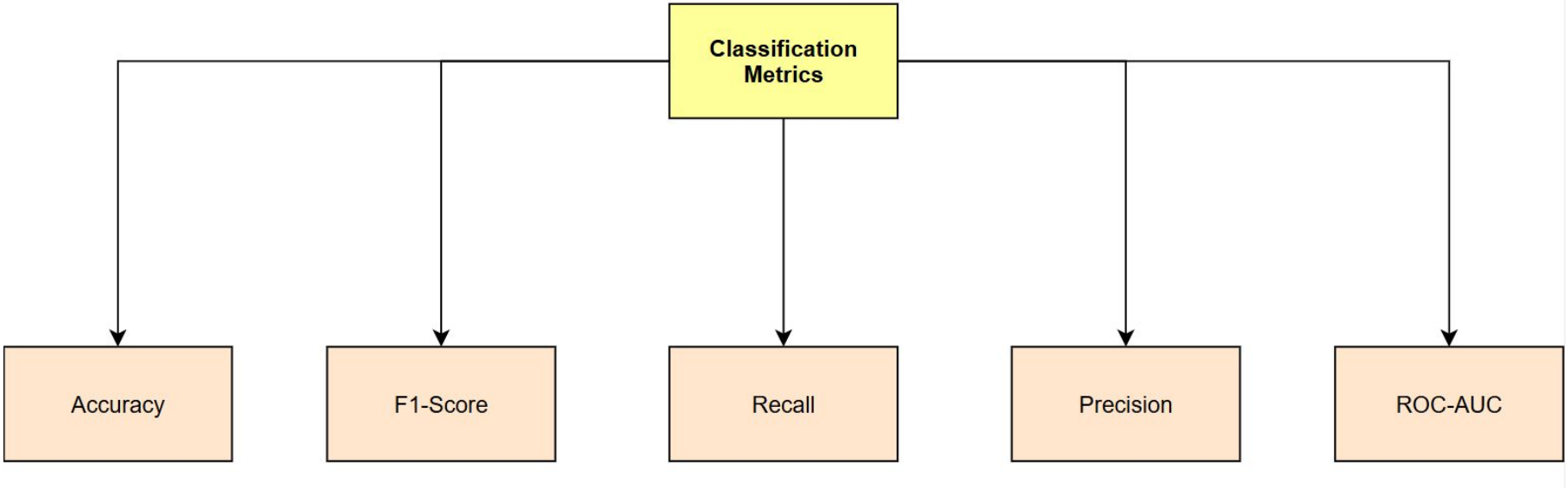
Recap: Supervised Learning

Evaluation Metrics - Regression



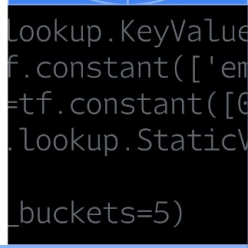
Recap: Supervised Learning

Evaluation Metrics - Classification



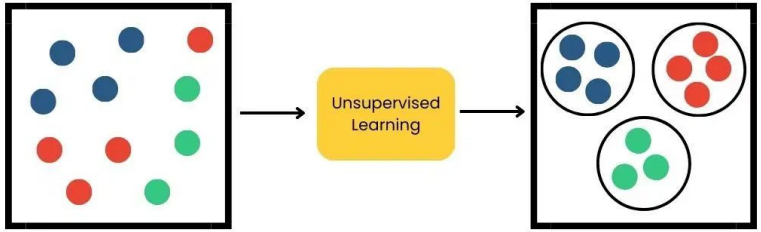
Google Developer Groups
On Campus • Telkom University Bandung

Core Concepts



Core Concepts: Definition

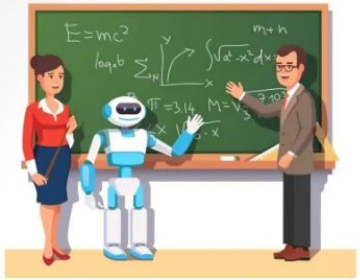
What is Unsupervised Learning?

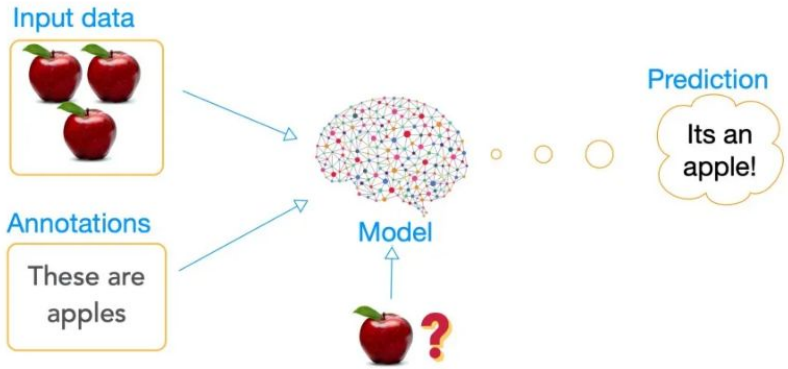


- **Unsupervised Learning** adalah teknik machine learning yang digunakan untuk menemukan pola atau struktur tersembunyi dalam data tanpa adanya label atau panduan sebelumnya.
    - Unsupervised: Data tidak memiliki atribut target
    - Supervised: Data memiliki atribut target

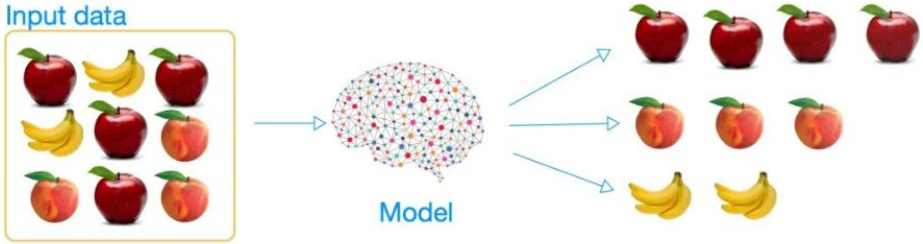
Core Concepts: Definition

Supervised Learning
(e.g. Regression, Classification)



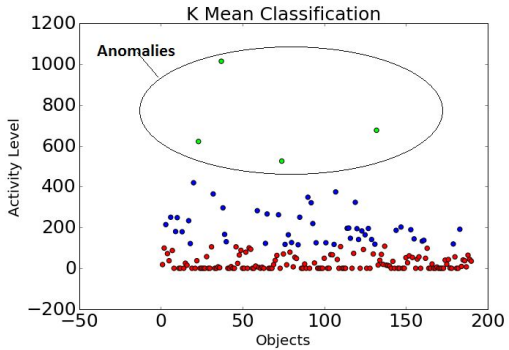


Unsupervised Learning
(e.g. Clustering)



Core Concepts: Definition

Real Life Applications



Anomaly Detection

Customer Segmentation

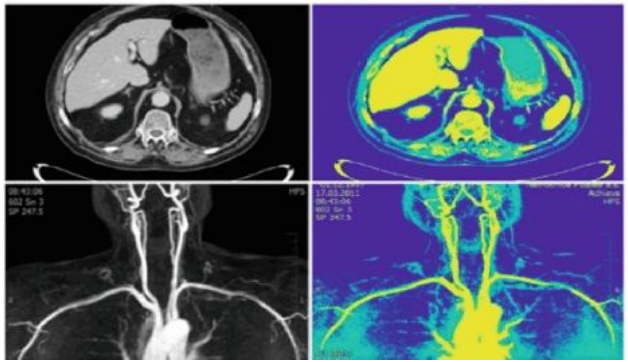
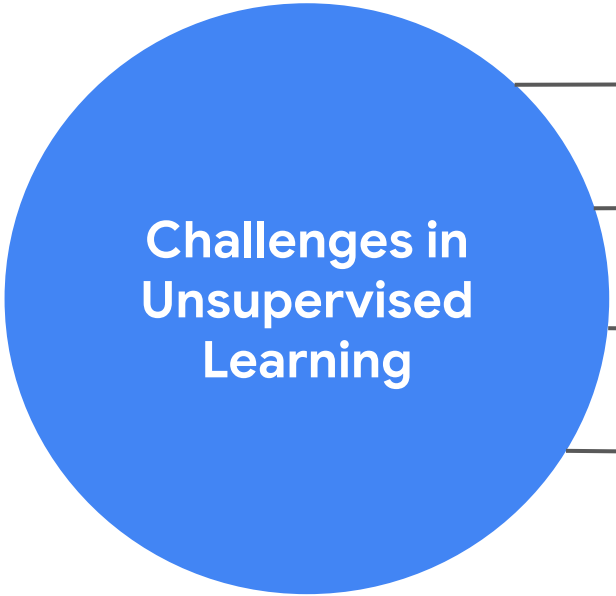


Image Segmentation

and more...!

Core Concepts: Challenges

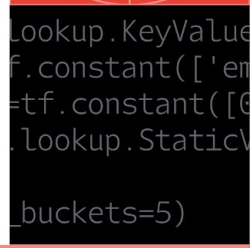


- Peningkatan kompleksitas komputasi yang timbul dari data pelatihan yang ekstensif
- Periode training yang berkepanjangan
- Peningkatan potensi hasil yang tidak akurat
- Perlunya keterlibatan manusia untuk memverifikasi variabel keluaran

Google Developer Groups
On Campus • Telkom University Bandung

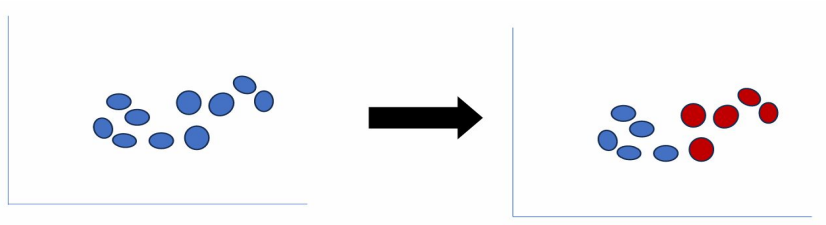
Clustering

Definition



What is Clustering?

- **Clustering** adalah teknik pengelompokan data yang memiliki karakteristik serupa ke dalam kelompok-kelompok atau klaster.
- **Tujuan** utama dari klasterisasi adalah untuk menemukan pola tersembunyi dalam data tanpa menggunakan label kelas sebelumnya.

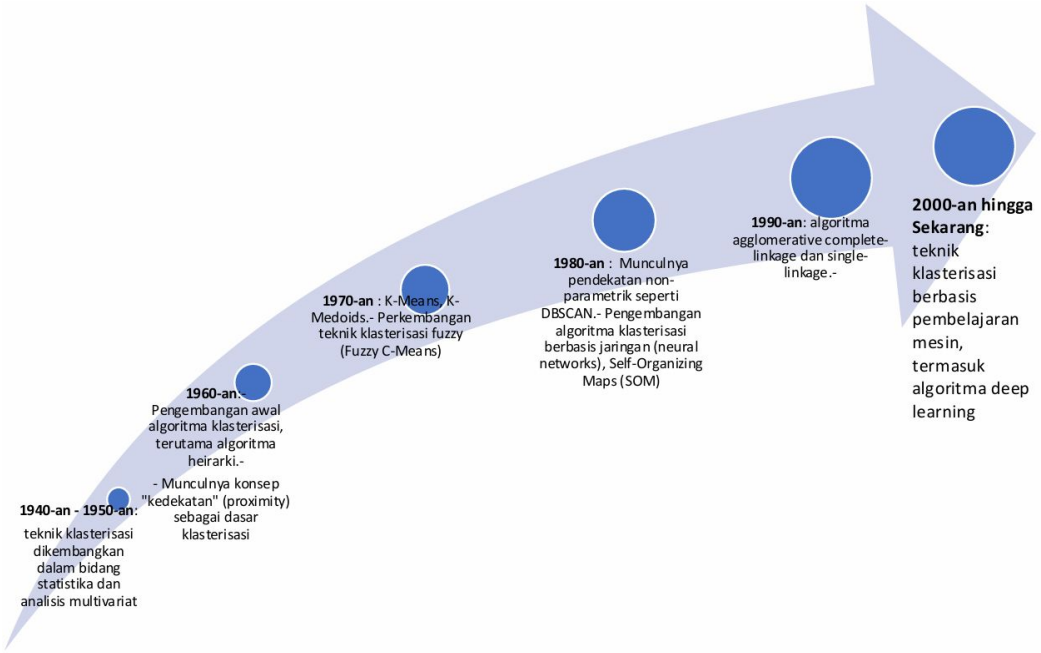


Illustration

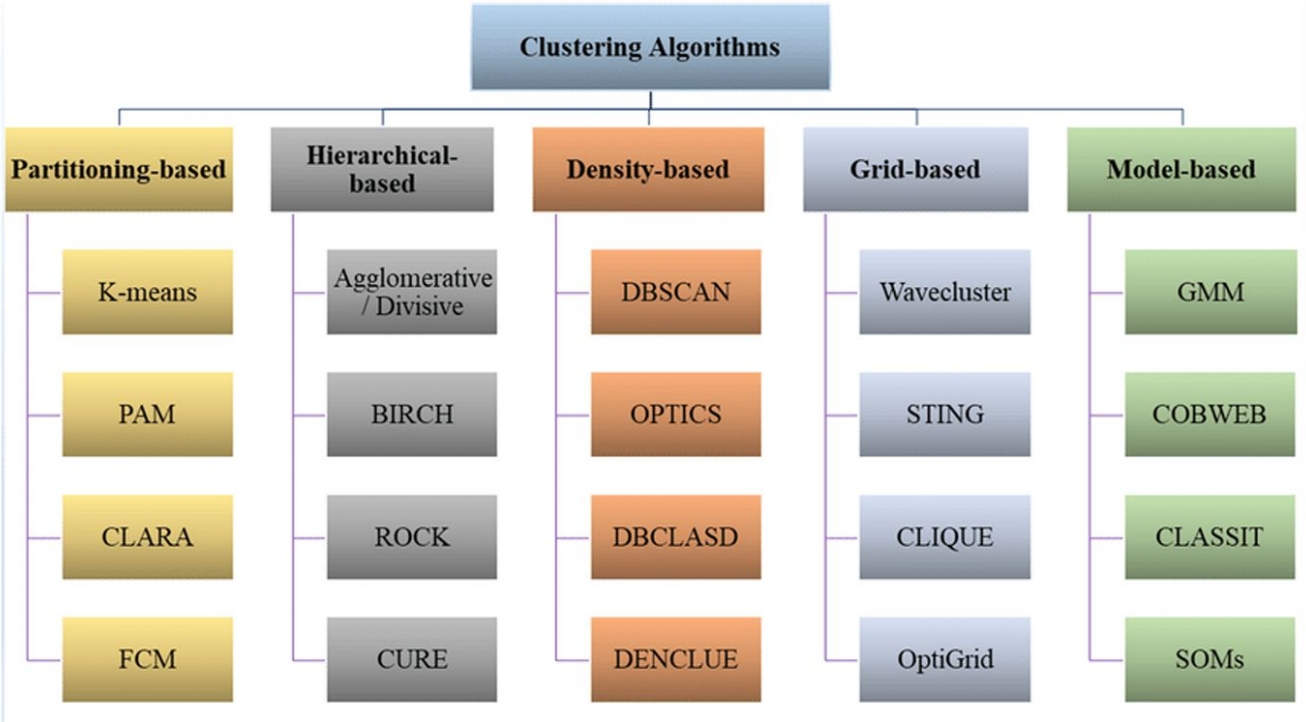
Clustering: Definition

Sejarah Clustering

- Clustering telah berkembang dari metode statistika sederhana menjadi serangkaian algoritma canggih dan dapat menangani berbagai jenis data.
- Penggunaannya yang luas dalam berbagai bidang ilmu pengetahuan dan industri, membuatnya menjadi salah satu teknik analisis data yang paling penting dan relevan hingga saat ini.



Clustering: Definition

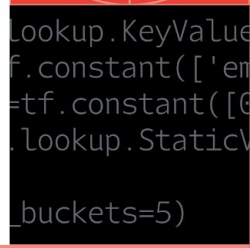


https://www.researchgate.net/figure/Broad-classification-of-clustering-algorithms_fig2_341111980

Google Developer Groups
On Campus • Telkom University Bandung

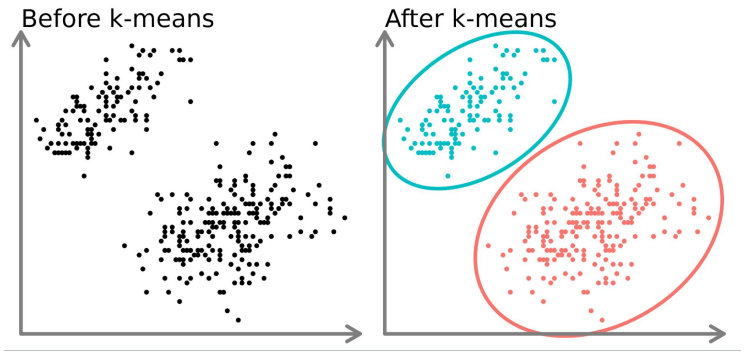
Clustering

K-Means



Clustering: K-Means

K-Means



- **K-Means** adalah algoritma clustering yang bertujuan membagi n titik data ke dalam k kelompok (klaster) berdasarkan jarak ke pusat klaster (centroid).

  Setiap klaster diwakili oleh rata-rata (mean) dari titik-titik di dalamnya, sehingga dinamakan "**K-Means**".

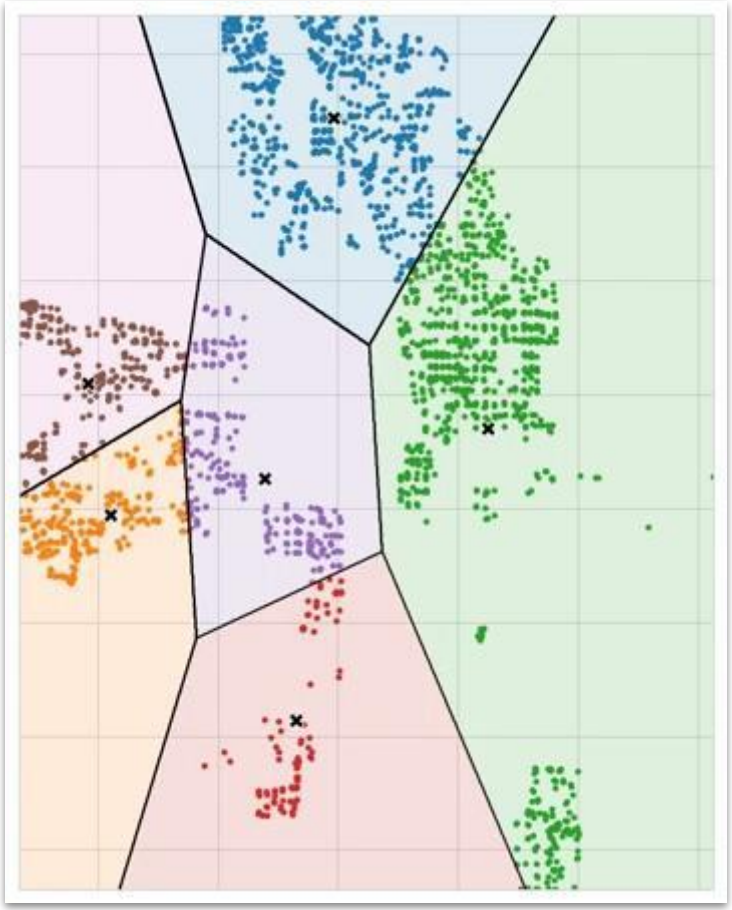
- **Kelebihan**
  - Sederhana dan cepat untuk dataset besar.
  - Efektif jika klaster berbentuk bulat dan terpisah dengan baik.
- **Kekurangan**
  - Harus menentukan k secara manual.
  - Rentan terhadap *outliers*.
  - Tidak cocok untuk klaster dengan bentuk tidak teratur

Notes: K-Means membutuhkan Feature Scaling di tahap Preprocessing

Clustering: K-Means

Algorithm 1 k-means clustering

1: Initialise Cluster Centers
2: **for** each iteration l **do**
3: Compute r_{nk}:
4: **for** each data point x_n **do**
5: Assign each data point to a cluster:
6: **for** each cluster k **do**
7: **if** k == argmin$\|x_n - \mu_k^{l-1}\|$ **then**
8: $r_{nk} = 1$
9: **else**
$r_{nk} = 0$
10: **end if**
11:
12: **end for**
13: **end for**
14: **for** each cluster k **do**
15: Update cluster centers as the mean of each cluster:
16: $\mu_k^l = \frac{\sum r_{nk} x_n}{\sum r_{nk}}$
17: **end for**
18: **end for**



Clustering: K-Means

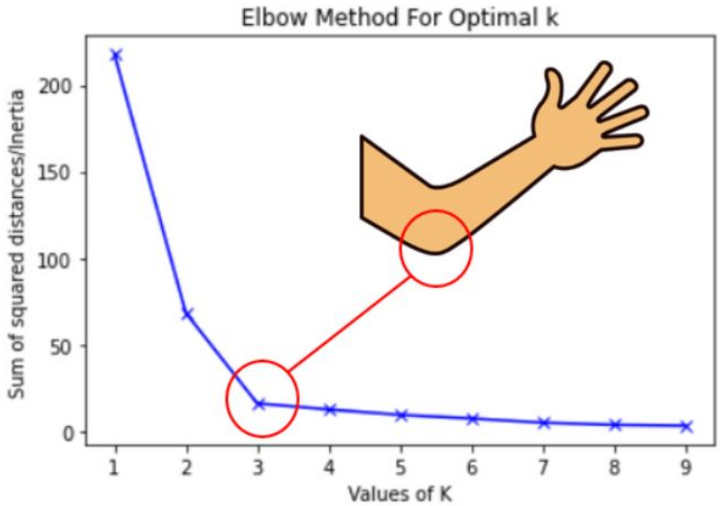
Model Initialization

```python
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

# feature scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# initialize and fit K-means model
kmeans = KMeans(
    n_clusters=3, # number of clusters
    init='k-means++', # initialization method
    n_init='auto', # number of initializations
    max_iter=300, # maximum number of iterations
    random_state=42 # random seed
)
kmeans.fit(X_scaled)

```

Elbow Method in K-Means

Elbow Method adalah teknik untuk **menentukan jumlah klaster optimal** (k) dalam algoritma K-Means

- **Kelebihan**
  - Sederhana dan intuitif untuk diinterpretasikan.
  - Memberikan panduan visual untuk memilih *k*.
  - Cocok untuk dataset dengan klaster yang jelas.
- **Kekurangan**
  - Tidak selalu ada "siku" yang jelas, terutama pada data kompleks atau klaster tidak terpisah baik

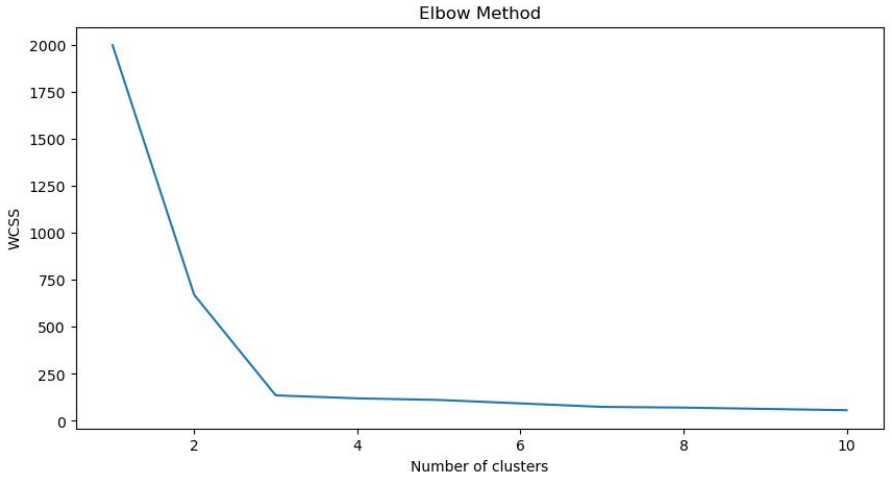


Line plot between K and inertia

Clustering: K-Means

Elbow Method in K-Means

```python
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(10, 5))
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

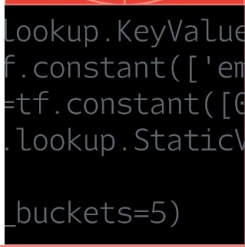


From the figure above, the optimal number of clusters based on the elbow method visualization is 3.

Google Developer Groups
On Campus • Telkom University Bandung

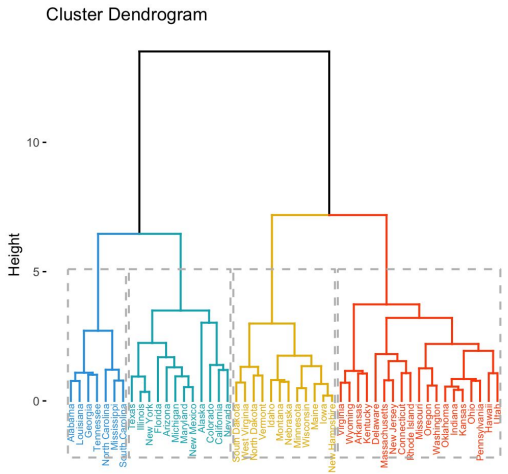
Clustering

Agglomerative Clustering



Clustering: Agglomerative Clustering

Agglomerative Clustering

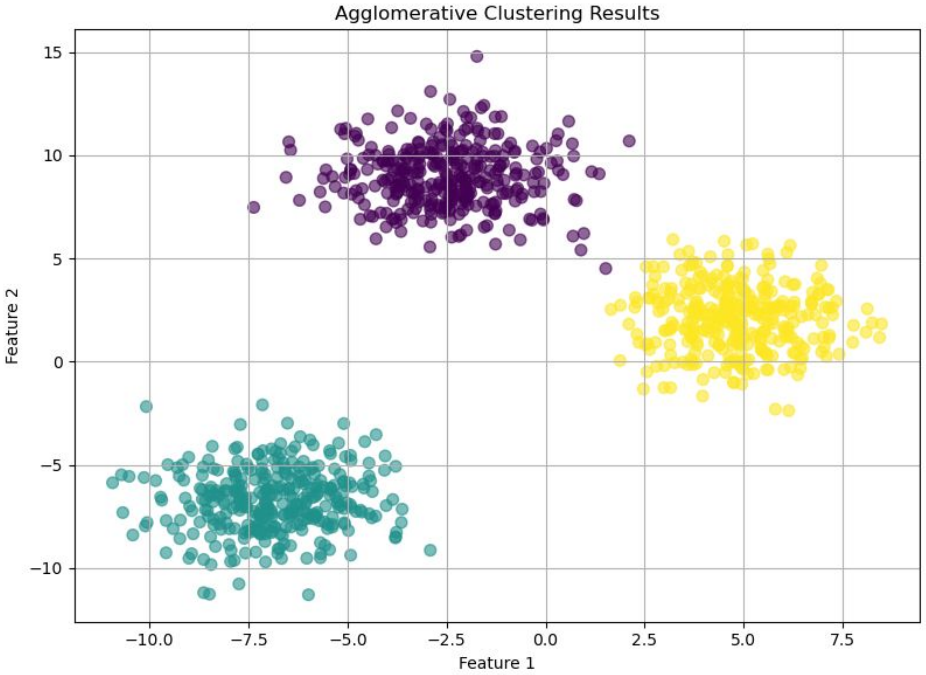


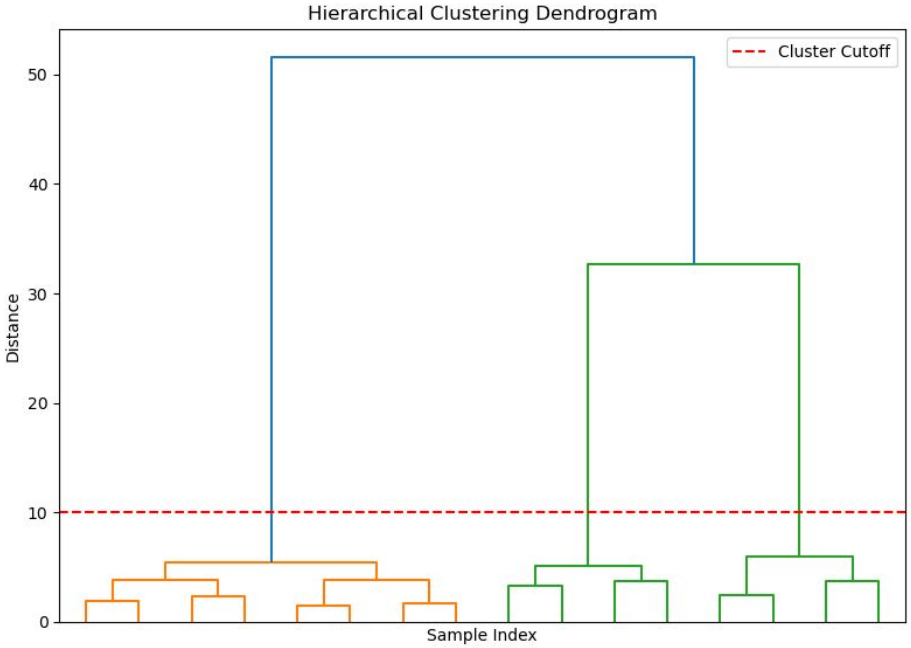
- **Agglomerative Clustering** adalah cara mengelompokkan data dengan pendekatan **dari bawah ke atas**.

  Setiap titik data mulai sebagai kelompok sendiri, lalu kelompok yang mirip digabung bertahap hingga jadi satu atau sesuai jumlah yang diinginkan.

- **Kelebihan**
  - Cocok untuk data kecil hingga sedang dan klaster dengan bentuk tidak teratur.
  - Tidak perlu tentukan k di awal (fleksibel dengan dendrogram).
- **Kekurangan**
  - Kompleksitas algoritma yang tinggi
  - Keputusan penggabungan tidak bisa dibatalkan (greedy algorithm).

Clustering: Agglomerative Clustering





Clustering: Agglomerative Clustering

Model Initialization

```python
from sklearn.cluster import AgglomerativeClustering
from sklearn.preprocessing import StandardScaler

# feature scaling (consistent with previous example)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# initialize and fit Agglomerative Clustering
agg_cluster = AgglomerativeClustering(
    n_clusters=3,          # number of clusters
    linkage='ward',        # linkage criteria
    compute_full_tree=True # for dendrogram analysis
)

cluster_labels = agg_cluster.fit_predict(X_scaled)
```

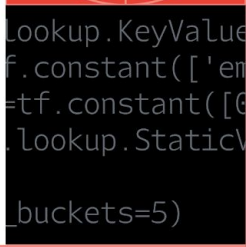
Dendogram Visualization

```python
from scipy.cluster.hierarchy import dendrogram, linkage
import matplotlib.pyplot as plt

# create linkage matrix
linkage_matrix = linkage(
    X_scaled,
    method='ward',  # Consistent with clustering linkage
    metric='euclidean'
)

# configure dendrogram plot
plt.figure(figsize=(12, 6))
dendrogram(
    linkage_matrix,
    truncate_mode='level',  # Simplify visualization
    p=3,                    # Truncation depth
    show_leaf_counts=True,  # Show cluster sizes
    orientation='top',      # Orientation of tree
    labels=agg_cluster.labels_
)

plt.axhline(y=10, color='r', linestyle='--', label='Cluster Cutoff')
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.legend()
plt.show()
```

Google Developer Groups
On Campus • Telkom University Bandung

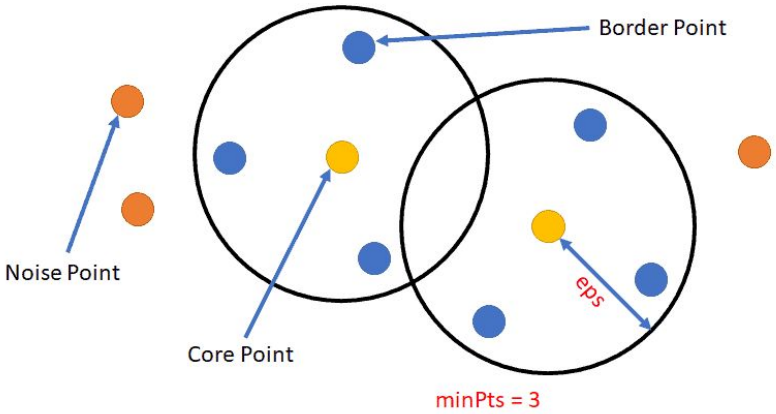
Clustering

DBSCAN



Clustering: DBSCAN

DBSCAN



- **DBSCAN** adalah algoritma pengelompokan (clustering) berbasis kepadatan yang dirancang untuk mengidentifikasi kelompok data dalam ruang fitur berdasarkan **distribusi kepadatan titik-titik**.

  Setiap titik data mulai sebagai kelompok sendiri, lalu kelompok yang mirip digabung bertahap hingga jadi satu atau sesuai jumlah yang diinginkan.

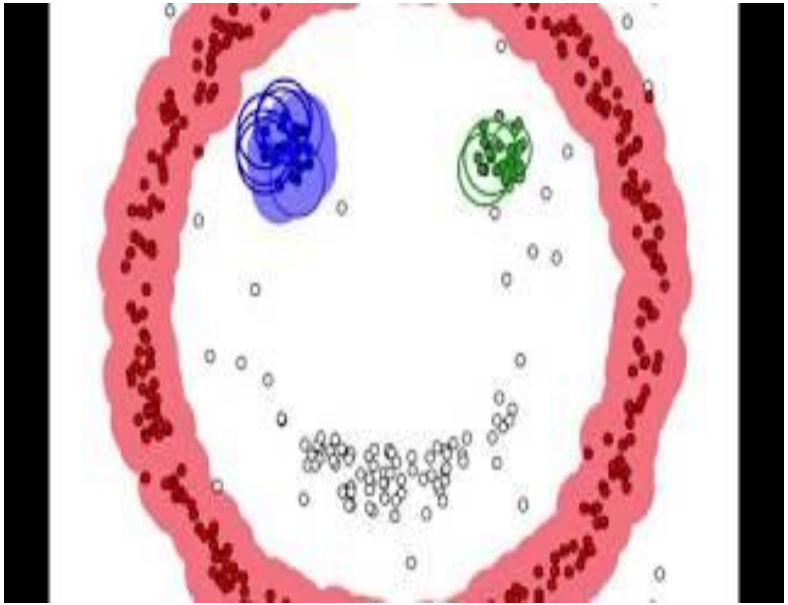
- **Kelebihan**
  - Tidak perlu tentukan jumlah klaster (k).
  - Mampu menangani klaster dengan bentuk tidak beraturan (misalnya spiral atau lingkaran).
  - Secara eksplisit mendeteksi *outlier*, cocok untuk data noisy.
- **Kekurangan**
  - Kompleksitas algoritma yang tinggi
  - Performa menurun pada data berdimensi sangat tinggi karena "curse of dimensionality" (jarak jadi kurang bermakna).

Clustering: DBSCAN

ALGORITHM 1: Pseudocode of Original Sequential DBSCAN Algorithm

Input: *DB*: Database
Input: ε: Radius
Input: *minPts*: Density threshold
Input: *dist*: Distance function
Data: *label*: Point labels, initially *undefined*

```
1  foreach point p in database DB do                    // Iterate over every point
2      if label(p) ≠ undefined then continue            // Skip processed points
3      Neighbors N ← RangeQuery(DB, dist, p, ε)         // Find initial neighbors
4      if |N| < minPts then                             // Non-core points are noise
5          label(p) ← Noise
6          continue
7      c ← next cluster label                           // Start a new cluster
8      label(p) ← c
9      Seed set S ← N \ {p}                             // Expand neighborhood
10     foreach q in S do
11         if label(q) = Noise then label(q) ← c
12         if label(q) ≠ undefined then continue
13         Neighbors N ← RangeQuery(DB, dist, q, ε)
14         label(q) ← c
15         if |N| < minPts then continue                // Core-point check
16         S ← S ∪ N
```

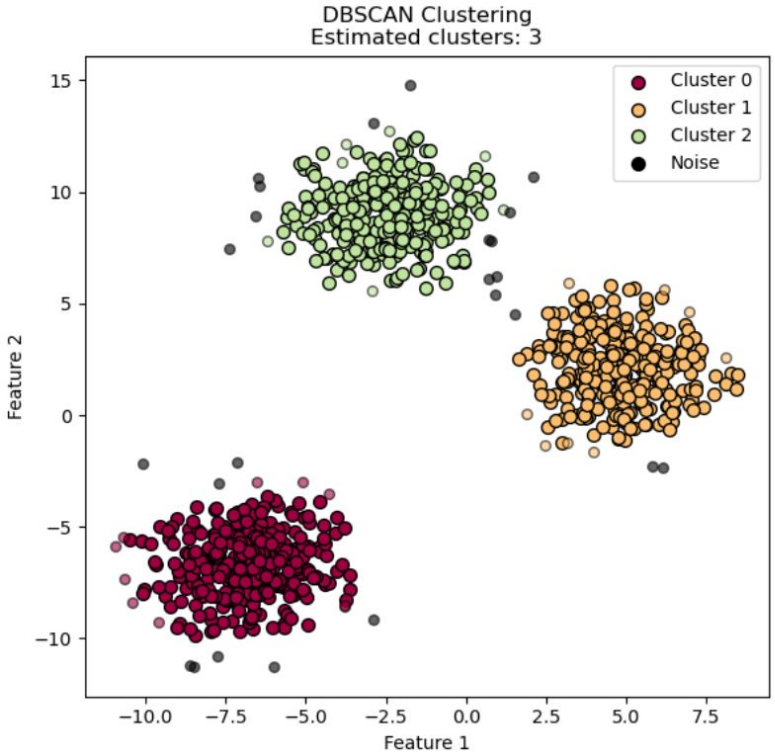


Clustering: DBSCAN

Model Initialization

```python
from sklearn.cluster import DBSCAN

# initialize and fit DBSCAN
dbscan = DBSCAN(
    eps=epsilon,          # neighborhood radius
    min_samples=5,        # minimum points in neighborhood
    metric='euclidean',   # distance metric
)
db_labels = dbscan.fit_predict(X_scaled)
```

Clustering: DBSCAN



Try to visualize it yourself!

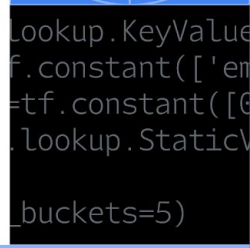
Source: https://clustering-visualizer.web.app/

Google Developer Groups

Google Developer Groups
On Campus • Telkom University Bandung

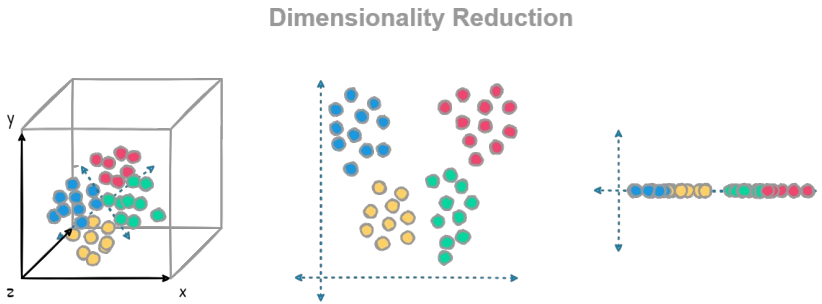
Dimensionality Reduction

Definition



Dimensionality Reduction:
Definition

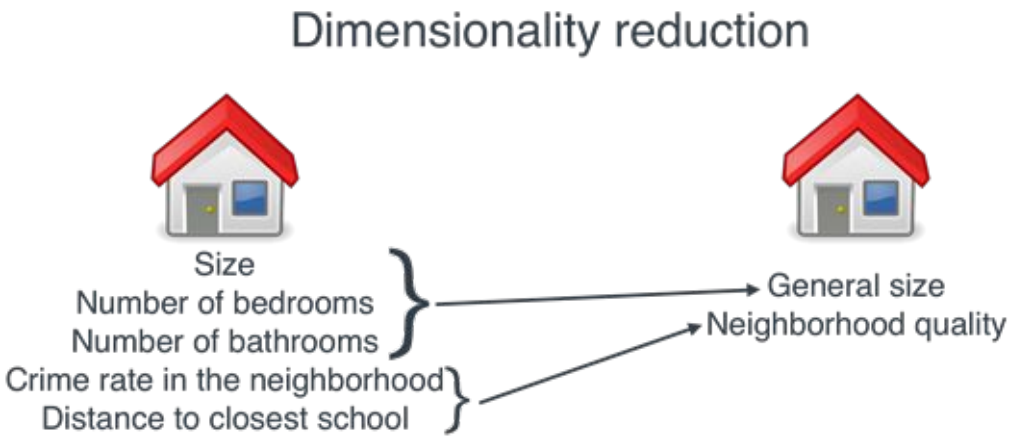
What is Dimensionality Reduction?



- **Dimensionality Reduction** adalah **proses mengurangi jumlah dimensi** (fitur atau variabel) dalam dataset sambil mempertahankan sebanyak mungkin informasi penting.
- **Tujuannya** adalah **menyederhanakan data** agar lebih mudah dianalisis, divisualisasikan, atau diproses oleh model machine learning tanpa kehilangan makna utama.
- **Mengapa Penting?**
  - **Efisiensi**
  - **Noise Reduction**
  - **Curse of Dimensionality**: Data berdimensi tinggi seringkali sulit diproses, membutuhkan lebih banyak komputasi, dan rentan terhadap overfitting.

Dimensionality Reduction:
Definition

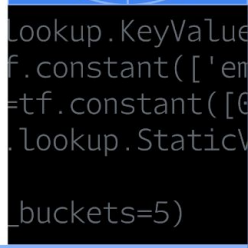
A great example to describe Dimensionality Reduction



Google Developer Groups
On Campus • Telkom University Bandung

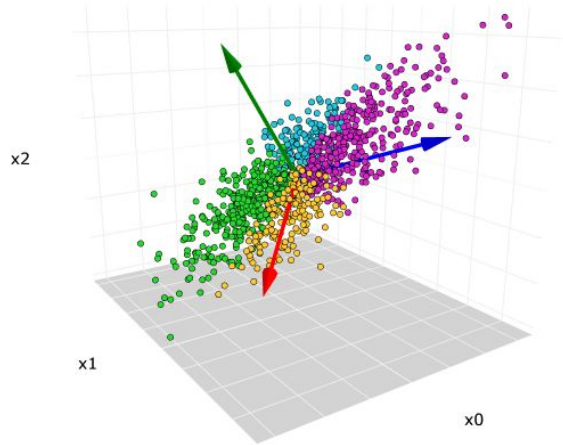
Dimensionality Reduction

Principal Component Analysis (PCA)



Dimensionality Reduction: PCA

PCA



- PCA adalah teknik untuk reduksi dimensi linier yang digunakan untuk **mengubah data berdimensi tinggi menjadi representasi dengan dimensi lebih rendah** sambil mempertahankan sebanyak mungkin variasi (informasi) dalam data
- **Kelebihan**
  - Tidak perlu tentukan jumlah klaster (k).
  - Mampu menangani klaster dengan bentuk tidak beraturan (misalnya spiral atau lingkaran).
  - Secara eksplisit mendeteksi *outlier*, cocok untuk data noisy.
- **Kekurangan**
  - **Linearitas**: PCA adalah teknik linier dan mungkin tidak dapat menangkap hubungan nonlinier yang kompleks dalam data.
  - **Sensitif terhadap skala data**: harus dinormalisasi terlebih dahulu.
  - **Hilangnya Informasi**: Meskipun PCA mempertahankan sebagian besar varians, ada beberapa informasi yang hilang ketika mengurangi dimensi.

Dimensionality Reduction: PCA

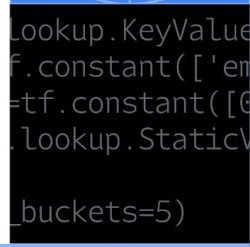
Initialization

```python
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import numpy as np

# data standardization (critical for PCA)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# initialize and fit PCA
pca = PCA(n_components=2)  # keep top 2 components
principal_components = pca.fit_transform(X_scaled)
```

Google Developer Groups
On Campus • Telkom University Bandung

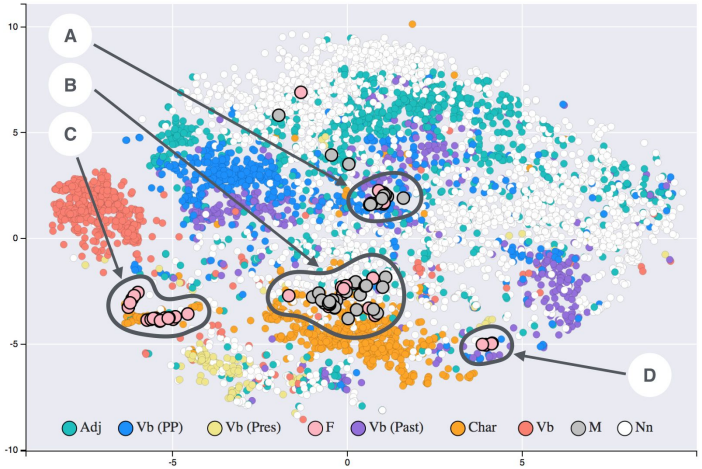
Dimensionality Reduction

t-distributed Stochastic Neighbor Embedding (t-SNE)



Dimensionality Reduction: t-SNE

t-SNE



- **t-SNE** adalah teknik reduksi dimensi non-linier yang digunakan terutama **untuk visualisasi data berdimensi tinggi** (seperti ribuan fitur) ke ruang dimensi rendah, biasanya 2D atau 3D.
- **Kelebihan**
  - Unggul dalam mempertahankan struktur lokal dan menangani data non-linier.
  - Menghasilkan visualisasi yang intuitif untuk data kompleks (misalnya gambar, teks, embedding).
  - Mampu menangkap klaster dengan bentuk tidak beraturan.
- **Kekurangan**
  - Sangat lambat untuk dataset besar
  - Tidak cocok untuk feature extraction atau prediksi (hanya visualisasi).

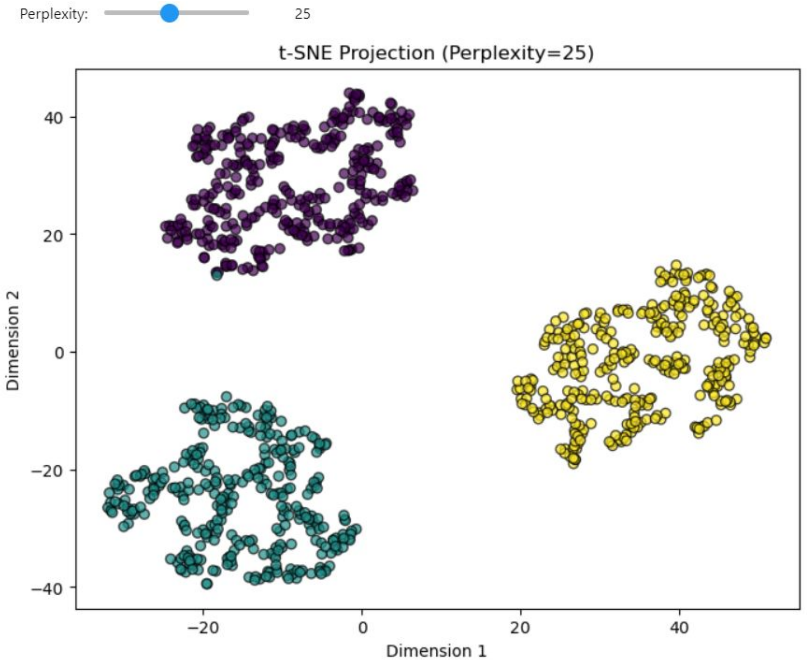
Dimensionality Reduction: PCA

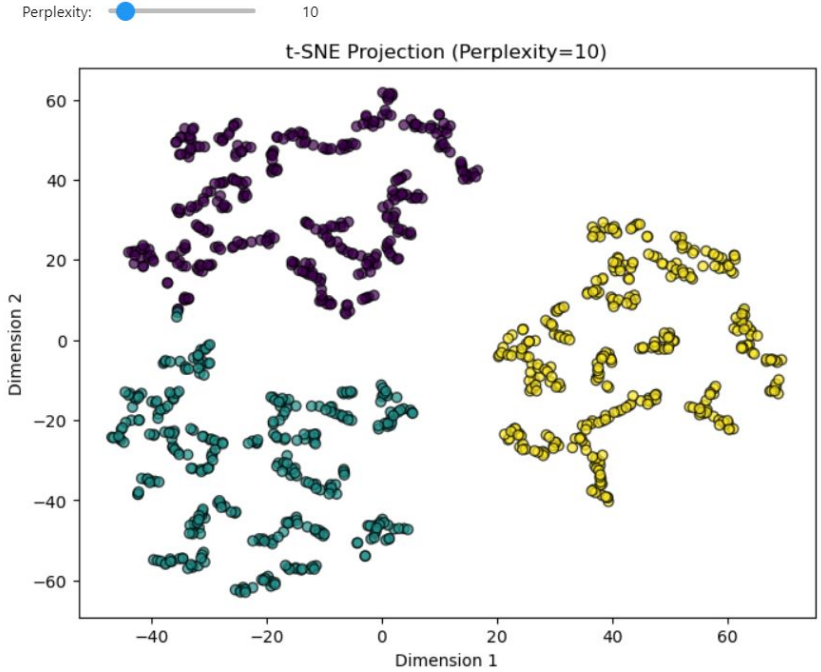
Initialize t-SNE

```python
from sklearn.manifold import TSNE

# data standardization (recommended for t-SNE)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# initialize and fit t-SNE
tsne = TSNE(
    n_components=2,          # Output dimensionality
    perplexity=30,           # Default: 30, try values between 5-50
    early_exaggeration=12.0, # Controls cluster spacing
    learning_rate='auto',    # Auto-adjust based on data size
    n_iter=1000,             # Minimum iterations
    random_state=SEED,
    n_jobs=-1
)
tsne_embeddings = tsne.fit_transform(X_scaled)
```

Dimensionality Reduction: t-SNE

Initialize t-SNE





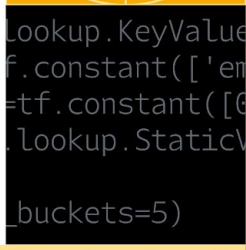
Google Developer Groups
On Campus • Telkom University Bandung

Hands On

ristek.link/marketing-data22

Google Developer Groups

Demo Source Code



Tugas

Lakukan metode-metode Clustering atau Dimensionality Reduction menggunakan metode unsupervised learning yang sudah dibahas, seperti K-Means, DBSCAN, Agglomerative, PCA, atau t-SNE.

Analisis hasilnya untuk mendapatkan insight yang bermakna, seperti pola kelompok, hubungan antar data, atau karakteristik utama dataset.

Minimal pengerjaan adalah 3 dataset!

Source: ristek.link/ML-penugasan-5

Google Developer Groups

Any Question ?

Google Developer Groups

Let’s Connect!

- **Instagram: @rdavaa_**
- **LinkedIn:** www.linkedin.com/in/ardava-barus