Dataset & Pra-Pemrosesan Data

Metodologi Penelitian Kuantitatif

Hendri Karisma, M.T.
Dosen Teknik Informatika, STMIK Tazkia, Bogor, Indonesia
VP Engineering, Jejakin.com
hendri@stmik.tazkia.ac.id — hendri.karisma@jejakin.com

Program Studi Teknik Informatika
STMIK Tazkia, Bogor, Indonesia

2026

Tujuan Pembelajaran

Setelah menyelesaikan pertemuan ini, mahasiswa diharapkan mampu:

1. Mengetahui berbagai sumber dataset publik yang umum digunakan dalam riset Ilmu Komputer.
2. Memahami teknik pengumpulan data primer dan kapan menggunakan data sintetis.
3. Menerapkan berbagai teknik pra-pemrosesan data (*pre-processing*) secara tepat.
4. Mengimplementasikan *pipeline* pra-pemrosesan menggunakan Python (Pandas & Scikit-learn).
5. Memahami aspek etika dalam pengumpulan dan pengelolaan data penelitian.

Agenda Pertemuan

Bagian I – Pengumpulan Data

1. Sumber Dataset Publik
2. Data Primer (Scraping, API, IoT)
3. Etika Pengumpulan Data

Bagian II – Pra-Pemrosesan

4. *Data Cleaning*
5. Normalisasi & Standardisasi
6. Transformasi Data
7. *Feature Selection & Extraction*
8. *Data Augmentation*
9. Penanganan *Class Imbalance*

Sumber Dataset Publik untuk Riset

Repositori	**Domain Utama**	**Keunggulan**
Kaggle	Multi-domain	Komunitas aktif, *notebook* contoh
UCI ML Repository	ML klasik	Standar akademis, 600+ dataset
ImageNet	*Computer Vision*	14 jt citra, *benchmark* DL
COCO	*Object Detection*	330 rb citra, anotasi kaya
Hugging Face	NLP & *Gen AI*	Library `datasets` Python
Papers With Code	*Benchmark* ML	Terhubung langsung ke *paper*

Repositori Tambahan

- **Google Dataset Search** – Meta-search untuk dataset
- **Portal Satu Data Indonesia** (`data.go.id`) – Data pemerintah RI
- **AWS Open Data** – Dataset berskala besar di cloud

Data Primer: *Web Scraping* & API

Web Scraping

- Mengambil data dari web secara otomatis
- Library: `BeautifulSoup`, `Scrapy`, `Selenium`
- Hormati `robots.txt` & *rate limiting*

```
from bs4 import BeautifulSoup
import requests

url = "https://contoh.com/data"
resp = requests.get(url)
soup = BeautifulSoup(resp.text,
                     'html.parser')
judul = [h.text for h in
```

API (Application Programming Interface)

- Akses data terstruktur resmi
- Format: JSON / XML
- Lebih stabil & legal dari *scraping*

```
import requests

api_key = "YOUR_API_KEY"
url = ("https://api.weather.org"
       f"/data?appid={api_key}")
resp = requests.get(url)
data = resp.json()
print(data['main']['temp'])
```

Data Primer: Survei Digital & Sensor IoT

Survei Digital

- Platform: Google Forms, SurveyMonkey, Typeform
- Cocok untuk riset UX, *usability testing*
- Gunakan Skala Likert (1–5) untuk persepsi pengguna
- Uji validitas & reliabilitas instrumen (Cronbach's Alpha)

Sensor IoT (Internet of Things)

- Data *real-time* dari sensor fisik
- Contoh: suhu, akselerometer, GPS, *heart rate*
- Platform: Arduino, Raspberry Pi, ESP32
- Protokol: MQTT, HTTP
- Cocok: *smart city*, *health monitoring*

Data Sintetis

Digunakan saat dataset terlalu kecil, data sensitif, atau untuk menguji *edge case*.
Library: `sklearn.make_classification()`, `Faker`, GAN/CTGAN.

Etika Pengumpulan Data

1. **Informed Consent** – Partisipan harus diberi informasi lengkap dan memberikan persetujuan sukarela.
2. **Anonimisasi (Anonymization)** – Data pribadi (nama, NIK, telepon) harus dihapus/disamarkan.
   Teknik: *k-anonymity*, *l-diversity*, *differential privacy*.
3. **Regulasi Privasi** – Patuhi GDPR (Uni Eropa) dan UU Perlindungan Data Pribadi / UU PDP (Indonesia).
4. **Lisensi Dataset** – Perhatikan lisensi (Creative Commons, MIT, GPL). Beberapa dataset hanya untuk non-komersial.
5. **Bias Data** – Waspadai bias gender, ras, atau geografis yang dapat menghasilkan model diskriminatif.

Prinsip Utama

"Kualitas riset dimulai dari integritas pengumpulan data."

Data Cleaning: Deteksi *Outlier*

Outlier = titik data yang nilainya sangat berbeda dari distribusi umum.

Metode IQR

- $IQR = Q3 - Q1$
- Batas bawah $= Q1 - 1{,}5 \times IQR$
- Batas atas $= Q3 + 1{,}5 \times IQR$

```
Q1 = df['nilai'].quantile(0.25)
Q3 = df['nilai'].quantile(0.75)
IQR = Q3 - Q1
bawah = Q1 - 1.5 * IQR
atas = Q3 + 1.5 * IQR
outlier = df[(df['nilai'] < bawah)
             | (df['nilai'] > atas)]
```

Metode Z-Score

- $Z = \dfrac{X - \mu}{\sigma}$
- Data dengan $|Z| > 3$ dianggap *outlier*
- Asumsi: distribusi mendekati normal

```
from scipy import stats
import numpy as np

z = np.abs(stats.zscore(
    df['nilai']))
outlier_z = df[z > 3]
```

Data Cleaning: Penanganan *Missing Values*

Strategi	Tipe Data	Kapan Digunakan
Listwise Deletion	Semua	Data hilang $< 5\%$, MCAR
Mean Imputation	Numerik	Distribusi simetris
Median Imputation	Numerik	Ada *outlier* / *skewed*
Mode Imputation	Kategorikal	Nilai paling sering
KNN Imputation	Numerik	Relasi antar fitur kuat
MICE	Kompleks	*Missing* besar & kompleks

```
from sklearn.impute import SimpleImputer, KNNImputer

# Imputasi mean untuk kolom numerik
imputer = SimpleImputer(strategy='mean')
df[['usia', 'gaji']] = imputer.fit_transform(df[['usia', 'gaji']])

# Imputasi mode untuk kolom kategorikal
imp_mode = SimpleImputer(strategy='most_frequent')
df[['kota']] = imp_mode.fit_transform(df[['kota']])
```

Normalisasi & Standardisasi

Min-Max Scaling (Normalisasi)
Mengubah nilai ke rentang $[0, 1]$:

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

- Mempertahankan distribusi asli
- Sensitif terhadap *outlier*
- Cocok: *Neural Network*, KNN

```
from sklearn.preprocessing \
    import MinMaxScaler
scaler = MinMaxScaler()
X_norm = scaler.fit_transform(X)
```

Z-Score Standardization
Mean $= 0$, Std $= 1$:

$$X_{\text{std}} = \frac{X - \mu}{\sigma}$$

- Lebih tahan terhadap *outlier*
- Tidak membatasi rentang nilai
- Cocok: SVM, PCA, *Logistic Reg.*

```
from sklearn.preprocessing \
    import StandardScaler
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
```

Transformasi: *One-Hot Encoding & Label Encoding*

One-Hot Encoding

- Variabel kategorikal nominal → representasi biner (0/1)
- Setiap kategori menjadi kolom baru
- Hati-hati *curse of dimensionality*

```
import pandas as pd

df = pd.DataFrame(
    {'warna': ['merah', 'biru',
               'hijau']})
df_enc = pd.get_dummies(
    df, columns=['warna'])
# warna_biru warna_hijau warna_merah
#      0          0          1
```

Label Encoding

- Kategori → integer (0, 1, 2, ...)
- Cocok untuk variabel **ordinal** (rendah < sedang < tinggi)
- Jangan untuk fitur nominal pada model linear!

```
from sklearn.preprocessing \
    import LabelEncoder

le = LabelEncoder()
df['kota_enc'] = \
    le.fit_transform(df['kota'])
# Bandung=0, Jakarta=1,
# Medan=2, Surabaya=3
```

Transformasi Teks: TF-IDF

TF-IDF (*Term Frequency – Inverse Document Frequency*) mengubah teks menjadi representasi numerik berdasarkan frekuensi dan kepentingan kata.

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \log \frac{N}{\text{DF}(t)}$$

- **TF**(t, d): Frekuensi term t dalam dokumen d
- **DF**(t): Jumlah dokumen yang mengandung term t
- N: Total jumlah dokumen

```
from sklearn.feature_extraction.text import TfidfVectorizer

dokumen = [
    "saya suka belajar machine learning",
    "machine learning adalah cabang ilmu komputer",
    "ilmu komputer sangat menarik untuk dipelajari"
]
vectorizer = TfidfVectorizer()
```

Feature Selection (Seleksi Fitur)

Memilih subset fitur yang paling relevan, membuang fitur redundan.

Correlation Analysis

- Korelasi Pearson antar fitur
- Fitur $r > 0{,}9$ bersifat redundan

```
corr = df.corr()
sns.heatmap(corr,
  annot=True,
  cmap='coolwarm')
```

Chi-Square Test

- Fitur kategorikal vs target kategorikal
- Nilai besar = hubungan kuat

```
from sklearn.\
  feature_selection \
  import chi2, SelectKBest
sel = SelectKBest(chi2, k=5)
X_sel = sel.fit_transform(
    X, y)
```

Information Gain

- Berbasis entropi Shannon
- Umum di *Decision Tree*

```
from sklearn.\
  feature_selection \
  import \
  mutual_info_classif
mi = mutual_info_classi
```

Feature Extraction: PCA & t-SNE

Membuat fitur **baru** dari fitur asli dengan mereduksi dimensi.

PCA (Principal Component Analysis)

- Transformasi linear → komponen utama
- Diurutkan berdasarkan variansi
- Cocok: data berdimensi tinggi, visualisasi, mengurangi multikolinearitas

```
from sklearn.decomposition \
    import PCA

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
print(pca.explained_variance_ratio_)
```

t-SNE

- Reduksi dimensi **non-linear**
- Sangat baik untuk **visualisasi** 2D/3D
- Tidak untuk *preprocessing* model (non-parametrik, tidak bisa transform data baru)

```
from sklearn.manifold import TSNE

tsne = TSNE(n_components=2,
    random_state=42,
    perplexity=30)
X_tsne = tsne.fit_transform(
    X_scaled)
```

Data Augmentation (untuk Citra)

Memperbanyak data *training* dengan transformasi pada data yang sudah ada.

Teknik Augmentasi Umum:

- Rotasi (15°, 30°, 90°)
- *Flipping* (horizontal/vertikal)
- *Random Crop*
- Perubahan *Brightness*
- *Zoom In/Out*
- *Gaussian Noise*
- *Color Jitter*

Library:

- `ImageDataGenerator` (Keras)
- `Albumentations` (lebih cepat)

```
# Keras ImageDataGenerator
from tensorflow.keras.preprocessing\
    .image import ImageDataGenerator

datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True,
    zoom_range=0.2,
    brightness_range=[0.8, 1.2]
)

train_gen = datagen.flow_from_directory
```

Penanganan *Class Imbalance*

Class imbalance: distribusi kelas sangat tidak seimbang (contoh: 99% normal, 1% *fraud*).

Teknik Utama:

- **SMOTE** – Membuat sampel sintetis kelas minoritas melalui interpolasi
- **Random Oversampling** – Duplikasi sampel minoritas (risiko *overfitting*)
- **Random Undersampling** – Hapus sampel mayoritas (kehilangan informasi)
- **ADASYN** – Fokus pada area *decision boundary*

```
from imblearn.over_sampling \
    import SMOTE

smote = SMOTE(random_state=42)
X_res, y_res = smote.fit_resample(
    X_train, y_train)

print("Sebelum:", pd.Series(
    y_train).value_counts().to_dic
print("Sesudah:", pd.Series(
    y_res).value_counts().to_dict(
# Sebelum: {0: 900, 1: 100}
# Sesudah: {0: 900, 1: 900}
```

Panduan Pemilihan Teknik *Imbalance*

Kondisi	**Teknik yang Disarankan**
Dataset kecil, perlu variasi baru	SMOTE atau ADASYN
Dataset kecil, sederhana	*Random Oversampling*
Dataset besar, kelas mayoritas berlebih	*Random Undersampling*
Imbalance sangat ekstrem (>1:100)	Kombinasi SMOTE + *Undersampling*
Model mendukung `class_weight`	Gunakan parameter `class_weight='balanced'`

Penting

Selalu evaluasi dengan metrik yang tepat untuk data *imbalanced*: gunakan **F1-Score**, **Precision-Recall**, atau **AUC-ROC**, **bukan** hanya akurasi!

Contoh *Pipeline Preprocessing* (Scikit-learn)

```
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.impute import SimpleImputer

fitur_num = ['Age', 'Fare', 'SibSp', 'Parch']
fitur_cat = ['Sex', 'Embarked', 'Pclass']

pipeline_num = Pipeline([
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])
pipeline_cat = Pipeline([
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('encoder', OneHotEncoder(handle_unknown='ignore'))
])
```

Ringkasan Pertemuan 12

1. **Dataset** dapat diperoleh dari repositori publik (Kaggle, UCI, ImageNet, COCO, Hugging Face), dikumpulkan sendiri (*scraping*, API, survei, IoT), atau dibuat sintetis.
2. **Etika data** mencakup *informed consent*, anonimisasi, kepatuhan regulasi, dan kesadaran terhadap bias.
3. **Data cleaning** meliputi penanganan *outlier*, *missing values*, dan duplikasi.
4. **Normalisasi** (Min-Max) dan **standardisasi** (Z-Score) menyeragamkan skala fitur.
5. **Transformasi data** mengubah fitur kategorikal dan teks menjadi representasi numerik (*One-Hot*, *Label Encoding*, TF-IDF).
6. **Feature selection** memilih fitur terbaik; **feature extraction** membuat fitur baru (PCA, t-SNE).
7. **Data augmentation** memperbanyak data citra; **penanganan class imbalance** dengan SMOTE, *oversampling*, *undersampling*.
8. Gunakan **pipeline Scikit-learn** untuk otomatisasi dan reproduktifitas.

Tugas Pertemuan 12

Tugas Individu

Pilih **satu dataset publik** dari Kaggle atau UCI ML Repository yang relevan dengan topik penelitian Anda, kemudian:

1. Lakukan *Exploratory Data Analysis* (EDA) dasar: tampilkan `shape`, `info()`, `describe()`, dan distribusi target.
2. Identifikasi dan tangani *missing values* dengan strategi yang tepat (jelaskan alasan pemilihan strategi).
3. Lakukan deteksi *outlier* menggunakan metode IQR atau Z-Score.
4. Terapkan normalisasi/standardisasi pada fitur numerik.
5. Lakukan *encoding* pada fitur kategorikal.
6. Jika data *imbalanced*, terapkan teknik SMOTE atau *oversampling/undersampling*.

Format: Jupyter Notebook (`.ipynb`), unggah ke LMS.
Deadline: Pertemuan berikutnya.