Artificial Intelligence

Week 2 — Data & EDA

Tujuan sesi

- Memahami *apa itu data*, jenis-jenis data, dan kecocokan untuk tugas AI
- Melakukan **EDA** yang fokus pada kualitas dan kegunaan
- Menentukan **split** yang benar: train • validation • test
- Menyusun **pipeline praproses**: missing • encoding • scaling

Output Tugas M2

Notebook yang berisi **EDA lengkap seperlunya + pipeline praproses + split** dan berjalan.

1. Pengertian Data

Data = rekaman fakta tentang objek atau peristiwa yang bisa diproses komputer.

- **Record** baris yang mewakili satu contoh
- **Feature** kolom yang menjelaskan record
- **Label Target** jawaban yang ingin diprediksi pada supervised learning

> Inti: Data adalah **bahan mentah**, model adalah **alat olah**, metrik adalah **rasa akhir**.

Contoh skema klasifikasi tiket

id	text	channel	hour	has_refund_word	label
101	“minta pengembalian dana…”	email	09	yes	Payment
102	“paket belum sampai”	chat	02	no	Shipping

2. Jenis Data & Kecocokan Tugas

Jenis	Contoh TRPL	Cocok untuk	Target umum	Representasi minimal
Tabular	transaksi, profil user, log ringkas	Klasifikasi, Regresi	kelas atau angka	angka + kategori ter-encode
Teks	tiket, chat, review	Klasifikasi intent, topik, sentiment	kelas	**TF-IDF** atau embedding
Gambar	produk, daun, kerusakan	Klasifikasi objek, defect	kelas	**transfer learning** CNN
Time Series	traffic jam-an, metrik server	Forecasting, anomaly	angka masa depan	lag window features
Audio	rekaman call center	Emosi, ASR → teks	kelas atau teks	MFCC atau ASR lalu TF-IDF

> Klasifikasi menebak **kategori**, regresi menebak **angka**, forecasting menebak **angka di masa depan**.

3. Glosarium Inti

- **EDA (Exploratory Data Analysis)** eksplorasi untuk memahami kualitas dan pola
- **Imputation** mengisi **missing value** dengan strategi tertentu
- **One-hot encoding** ubah kategori menjadi vektor biner
- **Standardization** (x' = (x – mean) / std) menyamakan skala fitur numerik
- **Stratified split** pembagian data dengan proporsi kelas terjaga
- **Time-aware split** pembagian mengikuti urutan waktu
- **Data leakage** informasi dari val test bocor ke train sehingga evaluasi palsu
- **PII (Personally Identifiable Information)** = data yang dapat mengidentifikasi orang tertentu, langsung atau tidak langsung.

4. EDA — Fokus 5 Pertanyaan

1. **Bentuk & tipe**: berapa baris kolom; numerik kategori teks gambar
2. **Missing**: kolom mana yang sering kosong dan alasannya
3. **Distribusi**: numerik ringkas; kategori frekuensi; label balance
4. **Korelasi sederhana**: fitur kembar redundan
5. **Kelayakan fitur**: fitur masuk akal memisahkan kelas tidak mengandung masa depan

5. Data Teks — Tujuan & Cara

Cocok untuk intent, topik, sentiment, prioritisasi tiket.

Representasi minimal

- **TF-IDF** kuat untuk baseline klasifikasi teks
- Fitur bantu: panjang teks, presence kata kunci, jam kirim, kanal

Pipeline ringkas teks

1. Bersih ringan lowercase hapus URL angka berlebihan
2. Vectorize TF-IDF
3. Model **Logistic Regression** Linear SVM
4. Metrik Accuracy F1 per kelas

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline

pipe_text = Pipeline([
  ("tfidf", TfidfVectorizer(min_df=3, ngram_range=(1,2))),
  ("clf", LogisticRegression(max_iter=1000))
])
```

6. Data Gambar — Tujuan & Cara

Cocok untuk klasifikasi jenis objek defective quality.

Prinsip minimal

- **Resize** ke 224x224 normalisasi
- **Transfer learning** pakai backbone ringan MobileNetV2 ResNet-18
- **Augmentasi** tipis flip crop kecil brightness ringan
- **Metrik** Accuracy Confusion Matrix

Alasan: dataset kecil menengah lebih stabil dengan transfer learning dibanding melatih CNN dari nol.

7. Data Tabular — Tujuan & Cara

Cocok untuk churn, credit scoring, fraud, pricing, risiko.

Pipeline minimal

- Numerik → **Impute median → StandardScaler**
- Kategori → **Impute most_frequent → OneHotEncoder handle_unknown ignore**
- Model awal: **Logistic Regression** klasifikasi atau **RandomForest** umum

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

num_pipe = Pipeline([("imp", SimpleImputer(strategy="median")),
                     ("sc", StandardScaler())])
cat_pipe = Pipeline([("imp", SimpleImputer(strategy="most_frequent")),
                     ("oh", OneHotEncoder(handle_unknown="ignore"))])

pre = ColumnTransformer([("num", num_pipe, num_cols),
                         ("cat", cat_pipe, cat_cols)])
```

8. Data Time Series — Kapan & Cara

Cocok untuk forecasting traffic beban server penjualan.

Aturan

- **Split time-aware** train data lama → val → test terbaru
- Fitur sederhana: lag t-1 t-2 t-7 rolling mean hari dalam minggu jam

Model minimal Linear Regression RandomForest + lag features
Metrik MAE RMSE MAPE

9. Data Audio — Kapan & Cara

Dua jalur

1. **Langsung** klasifikasi emosi jenis call
   Ekstrak **MFCC** atau mel-spectrogram → classifier ringan
2. **Tidak langsung** ubah ke teks (ASR) → pakai pipeline **teks**

Metrik Accuracy F1 untuk klasifikasi • WER untuk ASR.

10. Lisensi, Privasi, Etika Data

- **Sumber dan lisensi** pastikan hak pakai ubah distribusi komersial
- **PII** gunakan minimisasi consent anonimisasi
- **Data card** ringkas: asal lisensi kolom risiko transformasi
- **Audit trail** catat tanggal pengambilan pembersihan perubahan

11. Split Data — Aturan Cepat

- **Train Validation Test** contoh 70 15 15
- **Stratified** untuk target kategori agar proporsi konsisten
- **Time-aware** untuk data berurutan *tanpa shuffle*
- **Anti-leakage** fit imputation encoding scaling **hanya pada train** lalu apply ke val test

```
from sklearn.model_selection import train_test_split
X_tr, X_te, y_tr, y_te = train_test_split(X, y, test_size=0.2,
                                          stratify=y, random_state=42)
```

12. Pipeline Praproses — Urutan Minimal

1. **Missing**
   Numerik median • Kategori most_frequent constant
2. **Encoding**
   OneHotEncoder handle_unknown ignore
3. **Scaling**
   StandardScaler untuk model linear

Bungkus dalam Pipeline agar rapi, anti-leakage, mudah *grid search*.

```
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression

clf = Pipeline([
  ("preprocess", pre),      # ColumnTransformer dari slide sebelumnya
  ("model", LogisticRegression(max_iter=500))
])
clf.fit(X_tr, y_tr)
```

13. Evaluasi — Membaca Hasil

Klasifikasi

- **Accuracy** mudah dipahami
- **Precision Recall F1** penting bila kelas tidak seimbang
- **Confusion Matrix** melihat salahnya di mana

Regresi

- **MAE** rata-rata error absolut
- **RMSE** penalti untuk error besar

> Simpan **seed** dan versi paket untuk reprodusibilitas.

14. Contoh Pemetaan Masalah → Data → Cara

- **Intent tiket** → **Teks** → TF-IDF + Logistic Regression → **Klasifikasi**
- **Defect produk** → **Gambar** → Transfer learning → **Klasifikasi**
- **Churn user** → **Tabular** → One-hot + Scaling + LogReg Tree → **Klasifikasi**
- **Beban server** → **Time series** → Lag features + Linear Regression → **Forecasting**
- **Emosi call** → **Audio** → MFCC + classifier → **Klasifikasi**
- **ASR** → **Audio→Teks** → Pipeline teks

15. Kesalahan Umum

- **Leakage**: fit scaler encoder di seluruh data sebelum split **Salah**
- Memakai fitur **masa depan** saat melatih **Salah**
- Mengabaikan **imbalance** kelas minoritas lenyap **Waspada**
- Tidak ada **validation** hanya train test **Minim**
- Tidak menyimpan **seed** dan **versi paket** **Repro sulit**

16. Checklist Praktis M2

- [ ] Jenis data teridentifikasi teks gambar tabular time series audio
- [ ] EDA 5 poin terjawab dengan bullet
- [ ] Split benar stratified atau time-aware
- [ ] Pipeline praproses berjalan missing encoding scaling
- [ ] Evaluasi tampil accuracy F1 atau MAE RMSE
- [ ] Catatan 3–5 **perbaikan** untuk iterasi berikutnya

17. Tugas M2 — Output yang Dikumpulkan

Satu notebook berisi:

1. **EDA** 5 poin dan 3–5 keputusan praproses
2. **Split** sesuai karakter data + alasan singkat
3. **Pipeline praproses** di `ColumnTransformer + Pipeline`
4. **Model baseline** LogReg Tree CNN transfer learning sesuai data
5. **Evaluasi** metrik utama + 3 contoh error + 3 rencana perbaikan

Indikator penilaian

- EDA jelas dan relevan • Pipeline berjalan • Split benar • Laporan metrik ada

18. Template Kode Inti Tabular

```
# asumsikan df siap, num_cols dan cat_cols sudah ditentukan
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

X = df[num_cols + cat_cols]; y = df["label"]
X_tr, X_te, y_tr, y_te = train_test_split(X, y, test_size=0.2,
                                          stratify=y, random_state=42)

num_pipe = Pipeline([("imp", SimpleImputer(strategy="median")),
                     ("sc", StandardScaler())])
cat_pipe = Pipeline([("imp", SimpleImputer(strategy="most_frequent")),
                     ("oh", OneHotEncoder(handle_unknown="ignore"))])

pre = ColumnTransformer([("num", num_pipe, num_cols),
                         ("cat", cat_pipe, cat_cols)])

clf = Pipeline([("preprocess", pre),
                ("model", LogisticRegression(max_iter=500))])

clf.fit(X_tr, y_tr)
print(classification_report(y_te, clf.predict(X_te)))
```

19. Template Kode Inti Teks

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report

X_text = df["text"]; y = df["label"]
X_tr, X_te, y_tr, y_te = train_test_split(X_text, y, test_size=0.2,
                                          stratify=y, random_state=42)

pipe_text = Pipeline([
  ("tfidf", TfidfVectorizer(min_df=3, ngram_range=(1,2))),
  ("clf", LogisticRegression(max_iter=1000))
])

pipe_text.fit(X_tr, y_tr)
print(classification_report(y_te, pipe_text.predict(X_te)))
```

20. Template Alur Gambar Transfer Learning

```
# Pseudocode ringkas PyTorch Keras
# 1. Load backbone pretrained 2. Freeze sebagian 3. Tambah head klasifikasi
# 4. Augmentasi ringan 5. Fine tune beberapa epoch
# 6. Evaluasi accuracy dan confusion matrix
# Catatan: gunakan dataloader terstruktur dan split train val test yang konsisten
```

21. Ringkasan

- Definisikan **data** dengan jelas record feature label sesuai tugas
- Pilih jalur sesuai **jenis data** dan gunakan **representasi minimal** yang tepat
- Lakukan **EDA** fokus kualitas, **split** benar, **pipeline** rapi
- Ukur dengan **metrik yang sesuai** dan catat **rencana perbaikan**