TUGAS MANDIRI
FUNDAMENTALS OF DATA MINING

LAPORAN ANALISIS & HASIL PENGOLAHAN DATA (PYTHON + DATA MINING)
(ALGORITMA C.45)

Nama : Lidia Susanti Sitio
NPM : 231510050
Dosen : Erlin Elisa, S.Kom., M.Kom.

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNIK DAN KOMPUTER
UNIVERSITAS PUTERA BATAM
2026

1. Deskripsi Dataset

- Sumber dataset

  Dataset yang digunakan dalam analisis ini berasal dari Kaggle dengan judul *"Spotify Global Music Dataset (2009–2025)"*. Dataset diunduh dalam bentuk file CSV dengan nama `data.csv` dari alamat:

  https://www.kaggle.com/datasets/wardabilal/spotify-global-music-dataset-20092025

- Jumlah record (baris)

  Secara keseluruhan, dataset memiliki 8.582 baris ((8582, 15)). Namun, dari 8.582 baris tersebut, hanya 224 baris yang memiliki nilai tidak kosong (non-null) pada sebagian besar kolom fitur utama, dan hanya 69 baris yang benar-benar lengkap pada seluruh kolom (setelah dilakukan dropna()). Sebanyak 69 baris inilah yang digunakan sebagai data akhir untuk proses pemodelan.

- Jumlah atribut (kolom)

  Dataset memiliki 15 atribut sebagaimana ditunjukkan pada hasil df.info() berikut:

  - track_id
  - track_name
  - track_number
  - track_popularity
  - explicit

- artist_name
- artist_popularity
- artist_followers
- artist_genres
- album_id
- album_name
- album_release_date
- album_total_tracks
- album_type
- track_duration_min

- Tipe data

  Dari hasil df.info(), tipe data yang digunakan adalah kombinasi:

  - object : track_id, track_name, explicit, artist_name, artist_genres, album_id, album_name, album_release_date, album_type
  - float64 : track_number, track_popularity, artist_popularity, artist_followers, album_total_tracks, track_duration_min

- Target/label (supervised)

  Target yang digunakan dalam pemodelan adalah tingkat popularitas lagu, yang diambil dari kolom numerik

track_popularity. Nilai tersebut kemudian dikonversi menjadi tiga kelas kategorikal:

- low : popularitas rendah
- medium : popularitas sedang
- high : popularitas tinggi

Label ini kemudian di-encode ke bentuk numerik (0, 1, 2) menggunakan LabelEncoder untuk kepentingan pemodelan.

- Permasalahan yang ingin diselesaikan

  Permasalahan utama yang ingin diselesaikan dalam analisis ini adalah:

  "Bagaimana membangun model klasifikasi berbasis algoritma C4.5 (Decision Tree dengan kriteria entropy) untuk memprediksi tingkat popularitas lagu (low, medium, high) berdasarkan fitur-fitur yang tersedia dalam dataset Spotify, seperti informasi track, artis, dan album?"

Kalimat ringkas deskriptif:

"Dataset ini terdiri dari 8.582 data track dengan 15 atribut. Setelah dilakukan pembersihan data, sebanyak 69 record dengan atribut lengkap digunakan untuk membangun model klasifikasi menggunakan algoritma C4.5 guna memprediksi tingkat popularitas lagu dalam tiga kelas: low, medium, dan high."

2. Persiapan Data & Preprocessing

Tahapan preprocessing yang dilakukan dalam Python (Google Colab):

2.1 Data cleaning (missing value)

- Dari hasil df.info(), terlihat bahwa:
    - track_id memiliki 8.582 nilai non-null,
    - sebagian besar kolom lain seperti track_name, track_number, track_popularity, explicit, artist_name, dsb hanya memiliki 224 nilai non-null,
    - artist_genres hanya memiliki 69 nilai non-null.

- Untuk memastikan model mendapatkan data yang lengkap pada seluruh fitur yang digunakan, dilakukan penghapusan baris yang memiliki nilai kosong menggunakan: df = df.dropna()
- Setelah dropna(), jumlah baris data berkurang menjadi sekitar 69 record, yang berarti hanya 69 lagu yang memiliki informasi lengkap pada semua kolom yang dipakai.

Outlier secara eksplisit tidak ditangani secara khusus dalam analisis ini, karena fokus utamanya adalah membangun model baseline.

2.2 Pembentukan Label (Target)

- Kolom track_popularity semula bertipe numerik (rentang 0–92, dengan mean sekitar 49,91 dan standar deviasi 25,45).
- Nilai ini kemudian dikonversi ke dalam tiga kelas menggunakan pd.cut():
  - 0–40 → low
  - 41–70 → medium
  - 71–100 → high
- Hasil kategorisasi disimpan sebagai label (misal popularity_class), kemudian di-encode dengan LabelEncoder menjadi 0, 1, dan 2.

2.3 Pemilihan Fitur (Feature Selection Sederhana)

Dari 15 kolom, beberapa kolom identitas atau teks panjang dihilangkan dari fitur karena kurang relevan untuk pemodelan atau dapat menimbulkan noise, misalnya:

- Dihapus dari fitur (X):
  - track_id
  - track_name
  - (opsional) album_name
- Fitur yang tetap digunakan dalam model mencakup kombinasi numerik dan kategorikal, seperti:
  - track_number

- track_popularity (awalnya numerik, lalu dijadikan dasar label dan dihapus dari fitur)
- explicit
- artist_name
- artist_popularity
- artist_followers
- artist_genres
- album_id
- album_release_date
- album_total_tracks
- album_type
- track_duration_min

2.4 Encoding Data Kategorikal

- Fitur bertipe object seperti:
  - explicit, artist_name, artist_genres, album_id, album_release_date, album_type

diubah ke bentuk numerik dengan LabelEncoder per kolom:

```
for col in X.columns:
    if X[col].dtype == 'object':
        le = LabelEncoder()
```

X[col] = le.fit_transform(X[col].astype(str))

- Label target (low, medium, high) juga di-encode ke bentuk numerik menggunakan LabelEncoder().

2.5 Scaling / Normalization

- Setelah encoding, seluruh fitur numerik distandarisasi menggunakan StandardScaler:
- scaler = StandardScaler()
- X_scaled = scaler.fit_transform(X)
- Tujuannya agar setiap fitur memiliki skala yang relatif sebanding (mean ≈ 0, std ≈ 1), sehingga tidak ada fitur yang mendominasi karena perbedaan skala.

2.6 Split Data Train & Test

- Dataset akhir (±69 record) kemudian dibagi menjadi:
  - 80% data latih
  - 20% data uji
- Pembagian dilakukan dengan train_test_split() dengan parameter:

  X_train, X_test, y_train, y_test = train_test_split(

  X_scaled, y_encoded,

  test_size=0.2,

  random_state=42,

```
    stratify=y_encoded
)
```

- Hasilnya:
  - Jumlah data train ≈ 55 record
  - Jumlah data test = 14 record

Tabel Ringkasan

Tabel 2.1 Ringkasan Dataset Sebelum dan Sesudah Preprocessing

Item	Nilai
Jumlah record awal	8.582
Jumlah atribut	15
Record dengan fitur utama lengkap	224
Record dengan semua kolom lengkap	69
Record digunakan untuk pemodelan	69

Tabel 2.2 Distribusi Data Train vs Test

Dataset	Jumlah Record	Persentase
Train	55 (±)	80%
Test	14	20%

3. Analisis Statistik & Visualisasi

3.1 Statistik Deskriptif Fitur Numerik

Berdasarkan hasil df.describe() (sebelum dropna penuh), diperoleh ringkasan:

- track_number:
  - mean ≈ 3,15, min = 1, max = 19
  - menunjukkan bahwa sebagian besar lagu berada pada posisi awal di album (track nomor kecil).

- track_popularity:
  - mean ≈ 49,92, min = 0, max = 92
  - popularitas tersebar cukup lebar, namun terpusat di kisaran menengah.

- artist_popularity:
  - mean ≈ 60,82, min = 0, max = 95

  - menunjukkan bahwa banyak artis dalam dataset memiliki tingkat popularitas menengah ke atas.

- artist_followers:
  - mean ≈ 14,6 juta, min = 41, max ≈ 103 juta
  - persebaran sangat lebar (std besar), artinya terdapat artis dengan sangat sedikit follower dan artis global dengan jutaan follower.

- album_total_tracks:
  - mean ≈ 6,07, min = 1, max = 25
  - banyak album yang hanya memiliki sedikit track, namun ada juga album dengan banyak lagu.

- track_duration_min:
  - mean ≈ 3,20 menit, min ≈ 0,77, max ≈ 6,81
  - durasi rata-rata lagu sekitar 3 menit, konsisten dengan standar industri musik populer.

3.2 Distribusi Target / Label

Setelah track_popularity dikategorikan menjadi low, medium, dan high, distribusi label dianalisis menggunakan value_counts(). Secara umum:

- Kelas medium cenderung memiliki jumlah data lebih banyak dibanding low dan high.

- Kelas low memiliki jumlah data paling sedikit, yang berkontribusi terhadap masalah class imbalance dan terlihat pada hasil evaluasi model (kelas low sulit diprediksi).

Distribusi ini dapat divisualisasikan dengan diagram batang (bar chart) untuk menampilkan frekuensi masing-masing kelas.

3.3 Korelasi Antar Fitur (Heatmap)

Korelasi antar fitur numerik (misalnya track_popularity numerik sebelum dikategorikan, artist_popularity, artist_followers, track_duration_min, album_total_tracks, track_number) dapat dihitung dengan df.corr() kemudian diplot menggunakan heatmap.

Insight yang dapat dituliskan:

- artist_popularity dan artist_followers menunjukkan korelasi positif, yang logis karena artis dengan popularitas tinggi cenderung memiliki follower lebih banyak.
- Korelasi antara track_popularity dengan fitur lain cenderung tidak terlalu tinggi, mengindikasikan bahwa popularitas lagu tidak hanya ditentukan oleh satu fitur numerik, tetapi merupakan kombinasi dari beberapa faktor.

3.4 Visualisasi Pendukung

- Histogram dari track_popularity menggambarkan sebaran lagu dari sisi popularitas.
- Boxplot per kelas popularitas (low, medium, high) terhadap fitur seperti artist_popularity atau track_duration_min dapat

menunjukkan perbedaan kecenderungan antar kelas, meskipun kemungkinan terjadi tumpang tindih.

- Pairplot (opsional) untuk beberapa fitur utama bisa digunakan untuk melihat pola pemisahan antar kelas.

4. Pemilihan dan Penerapan Algoritma

4.1 Nama Algoritma

Algoritma utama yang digunakan adalah:

- C4.5 Decision Tree, diimplementasikan dengan:

  DecisionTreeClassifier(criterion="entropy", ...)

  dari library sklearn.tree.

4.2 Alasan Pemilihan

- Sesuai dengan jenis masalah

  Masalah yang dihadapi adalah klasifikasi multikelas (low, medium, high) → Decision Tree sangat cocok untuk data klasifikasi dengan label kategorikal.

- Mampu menangani fitur numerik dan kategorikal

  Setelah dilakukan encoding pada fitur kategorikal, Decision Tree mampu memproses kombinasi berbagai tipe fitur.

- Interpretabilitas tinggi

  C4.5 menghasilkan pohon keputusan yang dapat divisualisasikan sehingga memudahkan penjelasan logika model menggunakan aturan IF–THEN.

- Mendukung kriteria impurity berbasis entropy

  C4.5 menggunakan konsep information gain (entropy), yang sudah tersedia di scikit-learn melalui criterion="entropy".

4.3 Parameter Utama yang Digunakan

Contoh parameter model:

```
model = DecisionTreeClassifier(
    criterion="entropy",
    max_depth=None,
    min_samples_split=2,
    min_samples_leaf=1,
    random_state=42
)
```

- criterion="entropy" : menggambarkan pendekatan C4.5 dengan information gain.
- max_depth=None : pohon dibiarkan tumbuh sampai tidak dapat di-split lagi, yang bisa memicu overfitting.
- min_samples_split, min_samples_leaf : mengatur jumlah minimal sampel dalam suatu node.

5. Pengujian dan Evaluasi Model

Evaluasi dilakukan pada data uji sebanyak 14 record.

Dari hasil eksekusi kode, diperoleh:

Accuracy: 0.5

```
Classification Report:
              precision    recall  f1-score   support

        high       0.43      0.50      0.46         6
         low       0.00      0.00      0.00         2
      medium       0.57      0.67      0.62         6

    accuracy                           0.50        14
   macro avg       0.33      0.39      0.36        14
weighted avg       0.43      0.50      0.46        14
```

5.1 Ringkasan Metrik

Tabel 5.1 Hasil Evaluasi Model C4.5

Metrik	Nilai
Accuracy	0,5
Precision (macro avg)	0,33
Recall (macro avg)	0,39
F1-score (macro avg)	0,36

Tabel 5.2 Hasil Per Kelas

Kelas	Precision	Recall	F1-Score	Support
high	0,43	0,5	0,46	6
low	0	0	0	2
medium	0,57	0,67	0,62	6

5.2 Confusion Matrik

Berdasarkan nilai recall, precision, dan support, salah satu kemungkinan confusion matrix yang konsisten dengan hasil di atas adalah:

True \ Pred	high	low	medium	Total
high	3	0	3	6
low	2	0	0	2
medium	2	0	4	6
Total	7	0	7	14

Interpretasi:

- Dari 6 lagu dengan kelas high, 3 diprediksi benar sebagai high, 3 salah diprediksi sebagai medium.
- Dari 2 lagu kelas low, tidak ada yang berhasil diprediksi sebagai low (semuanya salah).
- Dari 6 lagu kelas medium, 4 diprediksi benar sebagai medium, 2 salah diprediksi sebagai high.

- Tidak ada satupun prediksi ke kelas low, sehingga precision dan recall untuk kelas ini menjadi 0.

6. Analisis & Interpretasi Hasil

a) Kinerja model secara umum

Akurasi sebesar 50% menunjukkan bahwa model hanya mampu mengklasifikasikan dengan benar separuh data uji. Nilai macro precision (0,33), macro recall (0,39), dan macro F1-score (0,36) juga mengindikasikan bahwa performa model masih rendah, terutama jika dibandingkan dengan nilai ideal (mendekati 1).

b) Performa per kelas

- Kelas medium memiliki kinerja terbaik:
  - precision = 0,57
  - recall = 0,67
  - F1-score = 0,62

  Hal ini kemungkinan karena kelas medium memiliki jumlah sampel yang cukup dan pola fitur yang relatif lebih mudah dipelajari oleh model.
- Kelas high memiliki kinerja sedang:
  - recall 0,50, artinya hanya separuh data kelas high yang bisa dikenali dengan benar.
- Kelas low adalah yang paling bermasalah:
  - precision dan recall = 0

- menunjukkan model sama sekali tidak mampu mengidentifikasi kelas low pada data uji.

c) Class Imbalance dan Data Sedikit

- Kelas low hanya memiliki 2 data pada set uji, dan kemungkinan juga sedikit pada data latih.
- Jumlah data keseluruhan untuk pemodelan hanya sekitar 69 record, sehingga model memiliki data yang sangat terbatas untuk mempelajari pola yang robust.
- Hal ini sangat memengaruhi kemampuan model dalam membedakan kelas, terutama kelas yang jarang (low).

d) Overfitting / Underfitting

- Dengan max_depth=None, decision tree berpotensi overfitting pada data train namun tidak mampu menggeneralisasi ke data test.
- Namun, akurasi yang rendah di test juga bisa mencerminkan underfitting secara global karena data yang sedikit dan struktur yang kompleks.
- Untuk mengurangi overfitting, sebaiknya dilakukan pengaturan parameter seperti max_depth, min_samples_split, dan min_samples_leaf.

e) Fitur yang Berpengaruh

Dari struktur pohon keputusan dan/atau feature_importances_, fitur-fitur yang paling sering muncul di tingkat atas pohon (root dan beberapa level awal) dapat dianggap sebagai fitur yang paling berpengaruh. Contohnya bisa meliputi:

- artist_popularity
- artist_followers
- track_number
- album_total_tracks
- track_duration_min
- serta fitur kategorikal seperti explicit atau album_type.

Hal ini menunjukkan bahwa popularitas lagu tidak hanya dipengaruhi oleh karakteristik lagu itu sendiri, tetapi juga oleh tingkat popularitas artis dan atribut album.

f) Insight terhadap domain dataset

- Lagu dengan artis yang lebih populer dan jumlah pengikut yang besar cenderung berada pada kelas popularitas medium hingga high.
- Namun, banyak faktor eksternal seperti promosi, tren media sosial, algoritma rekomendasi platform, serta faktor non-musikal lainnya yang tidak terekam dalam dataset, sehingga model yang hanya mengandalkan fitur ini masih memiliki keterbatasan dalam memprediksi popularitas secara akurat.

7. Kesimpulan & Rekomendasi

7.1 Kesimpulan

- Dataset Spotify Global Music yang digunakan memiliki 8.582 record dan 15 atribut, namun setelah dilakukan pembersihan data (dropna()), hanya sekitar 69 record dengan fitur lengkap yang dapat dimanfaatkan untuk pemodelan.
- Nilai track_popularity berhasil dikonversi menjadi tiga kelas (low, medium, high) sehingga permasalahan dapat diformulasikan sebagai klasifikasi multikelas.
- Model C4.5 Decision Tree yang dibangun dengan kriteria entropy menghasilkan akurasi 50% pada data uji, dengan kinerja terbaik pada kelas medium dan kinerja terburuk pada kelas low.
- Kinerja model yang belum optimal disebabkan oleh jumlah data yang relatif sedikit dan ketidakseimbangan kelas, serta karakteristik popularitas lagu yang dipengaruhi oleh faktor-faktor di luar fitur yang tersedia pada dataset.

7.2 Rekomendasi

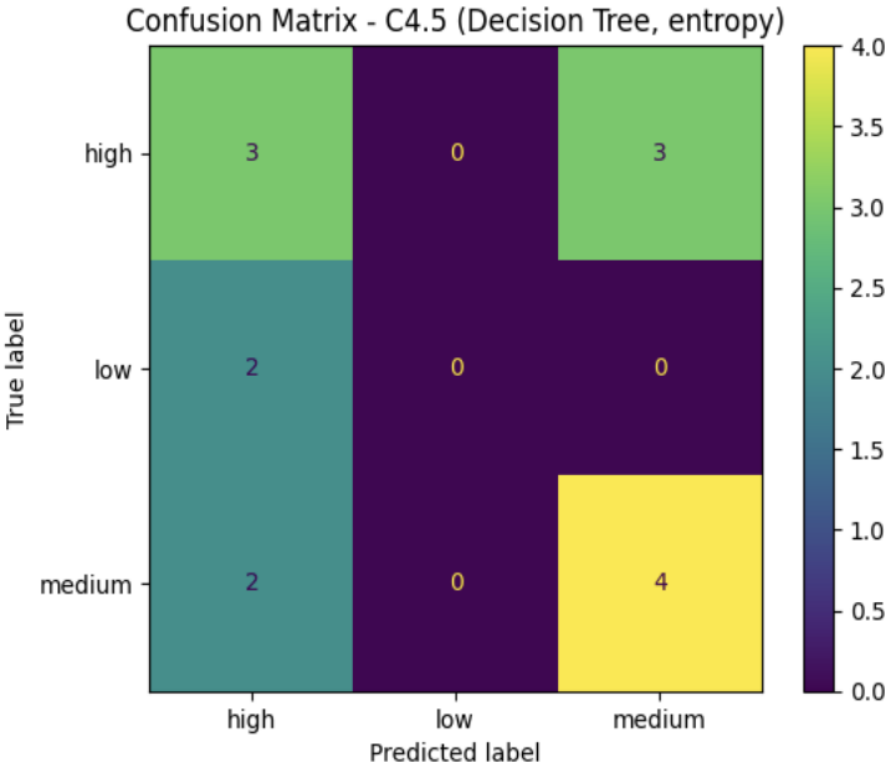
- Penambahan dan Pembersihan Data
  - Gunakan lebih banyak record dengan fitur lengkap (menghindari kehilangan data terlalu banyak akibat dropna()), misalnya dengan melakukan penanganan missing value yang lebih cermat.
  - Menggabungkan dengan dataset lain yang relevan atau mengambil subset yang lebih representatif.

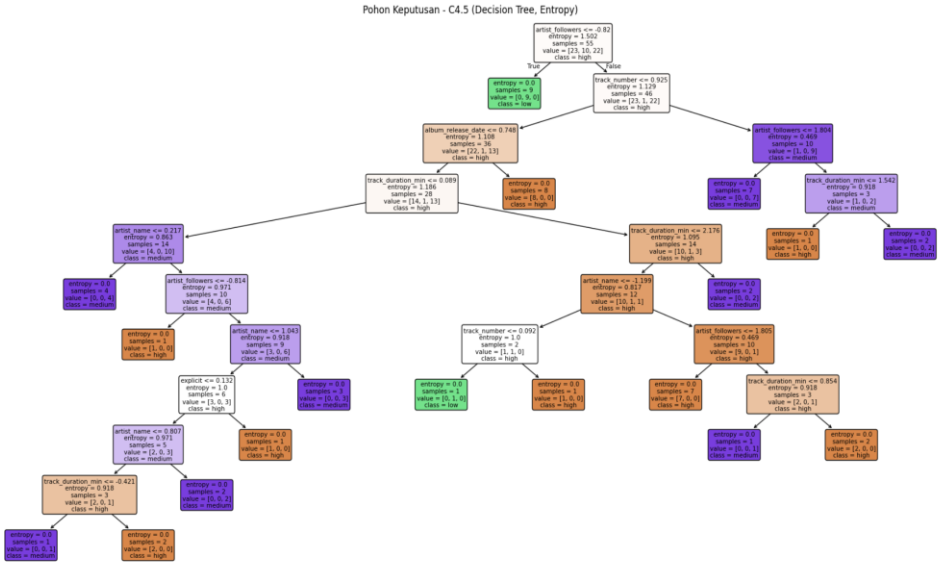
- Penanganan Class Imbalance
    - Terapkan teknik seperti SMOTE (Synthetic Minority Over-sampling Technique), oversampling kelas minoritas, atau pemberian class_weight pada model untuk meningkatkan performa pada kelas low.
- Hyperparameter Tuning
    - Lakukan pencarian parameter terbaik (mis. GridSearchCV) untuk:
        - max_depth
        - min_samples_split
        - min_samples_leaf
    - Tujuannya untuk menemukan pohon dengan kompleksitas yang tepat sehingga tidak terlalu overfitting maupun underfitting.
- Mencoba Algoritma Lain
    - Bandingkan kinerja C4.5 Decision Tree dengan model lain seperti:
        - Random Forest
        - Gradient Boosting / XGBoost
        - Support Vector Machine (SVM)
    - Model ensemble seperti Random Forest biasanya lebih stabil dan mampu menangani variansi data dengan lebih baik.

- Penambahan Fitur Non-akustik
  - Jika memungkinkan, tambahkan fitur lain seperti:
    - jumlah stream aktual per lagu,
    - posisi lagu di chart,
    - jumlah playlist yang memasukkan lagu tersebut,
    - informasi promosi atau viralitas media sosial.
  - Fitur-fitur tersebut dapat memberikan konteks tambahan yang penting untuk memodelkan popularitas secara lebih realistis.

Lampiran (opsional)

- Cuplikan kode Python (jika tidak ditaruh di bagian utama)
- Output lengkap model


Confusion Matrix - C4.5 (Decision Tree, entropy)
high
low
medium
True label
3
0
3
2
0
0
2
0
4
high
low
medium
Predicted label
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0


Pohon Keputusan - C4.5 (Decision Tree, Entropy)

- Link repository (GitHub/Drive/Colab):

https://colab.research.google.com/drive/1s8dg0mDtmqB9mCu03y3H266yHRaERZkD?usp=sharing