A Comparative Analysis of K-Means and K-Medoids Algorithms for Steam Game Clustering Based on Popularity, Price, and Playtime [Perbandingan Algoritma K-Means Dan K-Medoids Untuk Klasterisasi Game Pada Platform Steam Berdasarkan Popularitas, Harga, Dan Waktu Bermain]

Fathur Rohman Mufid [1)], Hindarto [2)]

[1)]Program Studi Informatika, Universitas Muhammadiyah Sidoarjo, Indonesia
[2)]Program Studi Informatika, Universitas Muhammadiyah Sidoarjo, Indonesia
*Email Penulis Korespondensi: hindarto@umsida.ac.id

Abstract. *Steam's large and heterogeneous game catalog cannot be adequately described by genre alone because games with similar genres may differ in market reach, pricing, and player engagement. This study clusters Steam games using estimated owners, price, and average lifetime playtime, while comparing K-Means with K-Medoids implemented through FasterPAM. A four-stage filtering process reduced 136,080 raw records to 22,659 eligible games. The clustering features were transformed using Yeo-Johnson and standardized before evaluating K values from 2 to 10. K=3 was selected because it produced the highest Silhouette Score. K-Medoids achieved a Silhouette Score of 0.2922, slightly higher than K-Means at 0.2913, although its Davies-Bouldin Index was higher. K-Medoids also produced a more balanced cluster distribution and was selected as the primary model. Kruskal-Wallis tests showed significant differences across all features, producing three segments: Popular-Premium-High Engagement, Budget/F2P, and Niche-Low Engagement. Overall, K-Medoids provided a suitable representation of the analyzed Steam game population.*

Keywords: K-Means; K-Medoids; game clustering; Steam; market segmentation

Abstrak. *Katalog game Steam yang besar dan beragam tidak dapat dijelaskan secara memadai hanya berdasarkan genre karena game dengan genre serupa dapat memiliki jangkauan pasar, harga, dan keterlibatan pemain yang berbeda. Penelitian ini mengelompokkan game Steam menggunakan estimasi jumlah pemilik, harga, dan rata-rata waktu bermain, serta membandingkan K-Means dengan K-Medoids menggunakan FasterPAM. Proses penyaringan empat tahap mengurangi 136.080 data mentah menjadi 22.659 game. Fitur klasterisasi ditransformasi menggunakan Yeo-Johnson dan distandarkan sebelum mengevaluasi nilai K dari 2 hingga 10. Nilai K=3 dipilih karena menghasilkan Silhouette Score tertinggi. K-Medoids memperoleh nilai 0,2922, sedikit lebih tinggi daripada K-Means sebesar 0,2913. K-Medoids juga menghasilkan distribusi klaster yang lebih seimbang sehingga dipilih sebagai model utama. Uji Kruskal-Wallis menunjukkan perbedaan signifikan pada seluruh fitur dan menghasilkan tiga segmen, yaitu Populer-Premium-Engagement Tinggi, Budget/F2P, dan Niche-Engagement Rendah. Secara keseluruhan, K-Medoids memberikan representasi yang sesuai terhadap populasi game Steam yang dianalisis.*

Kata Kunci: K-Means; K-Medoids; klasterisasi game; Steam; segmentasi pasar

I. PENDAHULUAN

Pertumbuhan katalog game digital menimbulkan kebutuhan terhadap metode pemetaan yang mampu menangkap variasi karakteristik produk secara terukur. Pada *Steam*, game tidak hanya berbeda menurut genre, tetapi juga menurut skala kepemilikan, harga, dan intensitas waktu bermain. *Genre* tetap relevan untuk menggambarkan jenis pengalaman permainan, namun atribut tersebut belum menjelaskan posisi sebuah *game* dari sisi jangkauan pasar maupun keterlibatan pemain. Dua *game* dalam *genre* yang sama, misalnya, dapat me miliki jumlah pemilik dan durasi bermain yang sangat berbeda. Atas dasar itu, klasterisasi digunakan untuk mengidentifikasi kelompok *game* berdasarkan kemiripan karakteristik numerik yang merepresentasikan popularitas, strategi harga, dan keterlibatan pemain.

K-Means dan *K-Medoids* merupakan metode klasterisasi partisi yang membentuk kelompok berdasarkan kedekatan objek terhadap pusat klaster, tetapi keduanya mendefinisikan pusat tersebut secara berbeda. *K-Means* menggunakan *centroid* yang dihitung dari rata-rata anggota klaster, sedangkan *K-Medoids* memilih observasi aktual sebagai *medoid* [1]–[3]. Pengembangan *FasterPAM* meningkatkan efisiensi optimasi *K-Medoids* sehingga metode berbasis *medoid* lebih layak diterapkan pada dataset yang lebih besar [4]. Studi komparatif pada beberapa domain memperlihatkan bahwa performa relatif *K-Means* dan *K-Medoids* bergantung pada struktur data dan metrik evaluasi yang digunakan [5]–[7]. Pada domain *Steam*, penelitian terdahulu telah membahas prediksi popularitas *game*, klasterisasi karakteristik *game*, pola pembelian, dan pengelompokan video *game* berdasarkan kesamaan karakteristik [16]–[19]. Temuan-temuan tersebut menunjukkan bahwa data *Steam* layak dianalisis menggunakan pendekatan

klasterisasi, sekaligus membuka ruang untuk membandingkan *K-Means* dan *K-Medoids* pada populasi dan fitur kuantitatif yang sama.

Kesenjangan penelitian dirumuskan dari tiga aspek. Pertama, penelitian berbasis data *Steam* telah menggunakan indikator popularitas, karakteristik *game*, dan pola pembelian [16]–[19], tetapi kombinasi estimasi kepemilikan, harga, dan waktu bermain sebagai tiga dimensi klasterisasi masih dapat dikaji lebih sistematis. Kedua, hasil perbandingan *K-Means* dan *K-Medoids* pada domain lain tidak menunjukkan satu metode yang selalu unggul [5]–[7]. Oleh sebab itu, penelitian ini mengendalikan fitur, indeks data, jumlah klaster, serta metrik evaluasi agar perbedaan hasil lebih mencerminkan karakteristik algoritma daripada perbedaan rancangan eksperimen. Ketiga, evaluasi internal dilengkapi dengan uji statistik terhadap fitur pada klaster yang terbentuk serta dokumentasi penyaringan data bertingkat. Pendekatan tersebut memungkinkan kualitas segmentasi dinilai tidak hanya dari skor evaluasi, tetapi juga dari keterlacakan pembentukan populasi dan perbedaan distribusi antarklaster.

Berdasarkan kesenjangan tersebut, penelitian ini bertujuan mengelompokkan *game* pada platform *Steam* menggunakan estimasi jumlah pemilik, harga, dan rata-rata waktu bermain, sekaligus membandingkan performa *K-Means* dan *K-Medoids* pada data yang memiliki distribusi sangat menceng. Tahapan penelitian meliputi akuisisi data, penyaringan bertingkat, rekayasa dan transformasi fitur, evaluasi jumlah klaster, pemodelan dengan kedua algoritma, pengukuran kualitas klaster, validasi statistik, serta interpretasi profil klaster. Rancangan ini diarahkan untuk menghasilkan segmentasi yang konsisten secara metodologis dan memberikan gambaran kuantitatif mengenai struktur populasi *game Steam* yang dianalisis.

II. METODE

A. Sumber dan Akuisisi Data

Penelitian ini menggunakan *Steam Games Dataset* yang disediakan oleh FronkonGames melalui *Kaggle*. *Dataset* tersebut juga telah dimanfaatkan dalam penelitian berbasis game digital, antara lain oleh Bartolomé et al. [20].
Data awal pada penelitian ini terdiri atas 136.080 entri dan 42 atribut yang mencakup harga, *genre*, kategori, ulasan, estimasi jumlah pemilik (*estimated_owners*), serta informasi waktu bermain.

Gambar 1. Mengunduh dataset

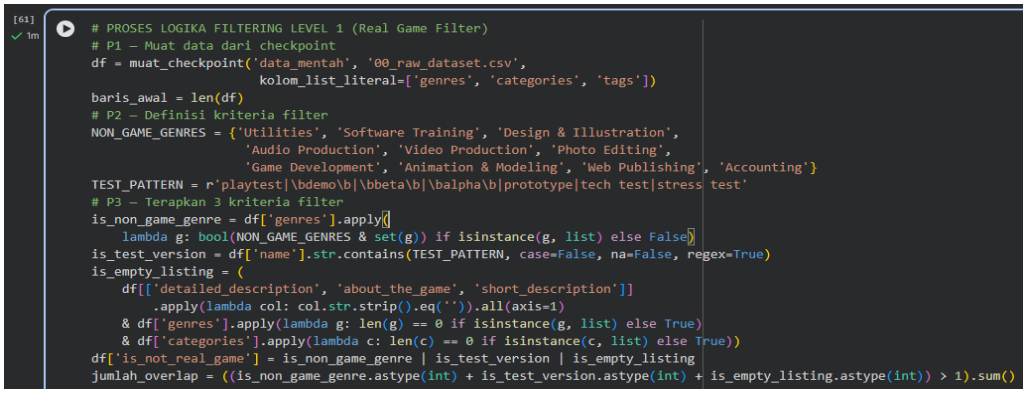
```
[7]
#B.1 DOWNLOAD DATASET KAGGLE

path_lock_raw = FOLDERS['data_mentah'] + '00_raw_dataset.csv'

if os.path.exists(path_lock_raw):
    print("Dataset mentah SUDAH TERKUNCI (checkpoint).")
    print(f"Path: {path_lock_raw}")
else:
    import kagglehub
    path_dataset = kagglehub.dataset_download("fronkongames/steam-games-dataset")
    print("Path dataset:", path_dataset)
    print("Isi folder  :", os.listdir(path_dataset))

    import shutil
    shutil.copy(f"{path_dataset}/games.json", "games.json")
    print("games.json berhasil diunduh")

...  Dataset mentah SUDAH TERKUNCI (checkpoint).
     Path: /content/drive/MyDrive/skripsi_steam/01_data_mentah/00_raw_dataset.csv
```

Dataset diunduh melalui *kagglehub* dan disimpan sebagai *checkpoint* mentah bernama 00_raw_dataset.csv sebelum penyaringan dilakukan. Seluruh tahap pemrosesan berikutnya selalu dimulai dari *checkpoint* yang sama. Prosedur ini digunakan untuk menjaga konsistensi sumber data dan mendukung reproduksibilitas proses penyaringan, rekayasa fitur, serta klasterisasi.

B. Penyaringan Data Bertingkat

Dataset mentah tidak seluruhnya merepresentasikan game yang sesuai dengan tujuan analisis. Di dalamnya terdapat entri perangkat lunak non-game, versi uji coba, data dengan informasi penting yang tidak lengkap, serta entri tanpa indikator aktivitas pemain yang memadai. Oleh karena itu, populasi analisis dibentuk melalui empat tahap penyaringan berurutan. Setiap tahap menggunakan kriteria yang disesuaikan dengan tujuan seleksi, mulai dari mengeluarkan entri yang tidak relevan hingga memastikan kelengkapan fitur yang diperlukan untuk klasterisasi.

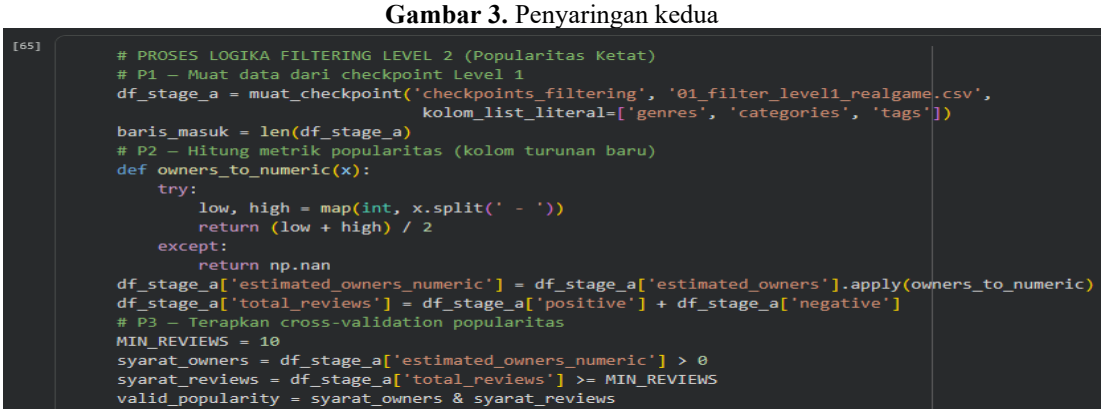
Gambar 2. Penyaringan data pertama



```
[61] ✓ 1m
# PROSES LOGIKA FILTERING LEVEL 1 (Real Game Filter)
# P1 — Muat data dari checkpoint
df = muat_checkpoint('data_mentah', '00_raw_dataset.csv',
                     kolom_list_literal=['genres', 'categories', 'tags'])
baris_awal = len(df)
# P2 — Definisi kriteria filter
NON_GAME_GENRES = {'Utilities', 'Software Training', 'Design & Illustration',
                   'Audio Production', 'Video Production', 'Photo Editing',
                   'Game Development', 'Animation & Modeling', 'Web Publishing', 'Accounting'}
TEST_PATTERN = r'playtest|\bdemo\b|\bbeta\b|\balpha\b|prototype|tech test|stress test'
# P3 — Terapkan 3 kriteria filter
is_non_game_genre = df['genres'].apply(
    lambda g: bool(NON_GAME_GENRES & set(g)) if isinstance(g, list) else False)
is_test_version = df['name'].str.contains(TEST_PATTERN, case=False, na=False, regex=True)
is_empty_listing = (
    df[['detailed_description', 'about_the_game', 'short_description']]
      .apply(lambda col: col.str.strip().eq('')).all(axis=1)
    & df['genres'].apply(lambda g: len(g) == 0 if isinstance(g, list) else True)
    & df['categories'].apply(lambda c: len(c) == 0 if isinstance(c, list) else True))
df['is_not_real_game'] = is_non_game_genre | is_test_version | is_empty_listing
jumlah_overlap = ((is_non_game_genre.astype(int) + is_test_version.astype(int) + is_empty_listing.astype(int)) > 1).sum()
```

Tahap pertama, *Real Game Filter*, menggunakan logika OR pada tiga indikator: *genre* yang mengarah pada perangkat lunak atau utilitas non-game, nama entri yang memuat penanda versi uji coba seperti *playtest*, *demo*, *beta*, atau *alpha*, serta entri dengan deskripsi dan kategori yang seluruhnya kosong. Sebuah entri dikeluarkan apabila memenuhi sedikitnya satu indikator tersebut. Penyaringan awal ini menurunkan jumlah data dari 136.080 menjadi 125.861 baris atau 92,5% dari populasi mentah. Secara teknis, satu indikasi saja sudah cukup mendiskualifikasi entri. Pilihan ini mencerminkan orientasi tahap awal: meminimalkan false negative (entri *non-game* yang lolos) dengan menerima risiko *false positive* yang lebih longgar.

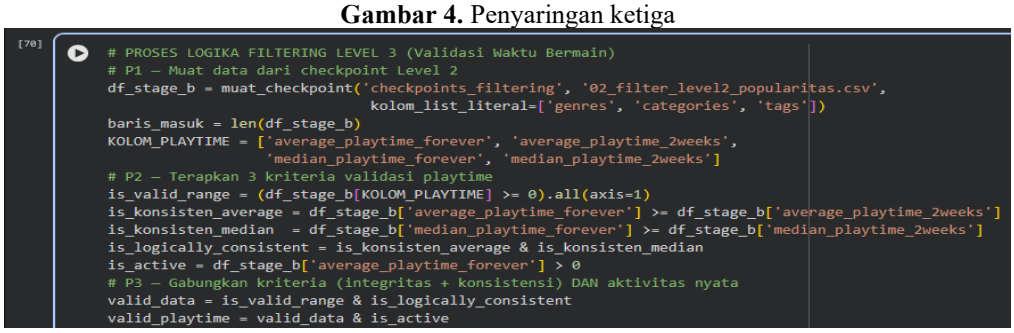
Gambar 3. Penyaringan kedua



```
[65]
# PROSES LOGIKA FILTERING LEVEL 2 (Popularitas Ketat)
# P1 — Muat data dari checkpoint Level 1
df_stage_a = muat_checkpoint('checkpoints_filtering', '01_filter_level1_realgame.csv',
                             kolom_list_literal=['genres', 'categories', 'tags'])
baris_masuk = len(df_stage_a)
# P2 — Hitung metrik popularitas (kolom turunan baru)
def owners_to_numeric(x):
    try:
        low, high = map(int, x.split(' - '))
        return (low + high) / 2
    except:
        return np.nan
df_stage_a['estimated_owners_numeric'] = df_stage_a['estimated_owners'].apply(owners_to_numeric)
df_stage_a['total_reviews'] = df_stage_a['positive'] + df_stage_a['negative']
# P3 — Terapkan cross-validation popularitas
MIN_REVIEWS = 10
syarat_owners = df_stage_a['estimated_owners_numeric'] > 0
syarat_reviews = df_stage_a['total_reviews'] >= MIN_REVIEWS
valid_popularity = syarat_owners & syarat_reviews
```

Proses *filtering* kedua, kolom *estimated_owners* berformat rentang teks dikonversi ke nilai numerik tunggal melalui titik tengah (*midpoint*), memungkinkannya diperlakukan sebagai variabel kontinu. Dua syarat independen yakni kolom *estimated_owners* >0 dan *total_reviews* ≥10 kemudian digabung dengan operator konjungsi (&). Tahap Popularitas Ketat mensyaratkan estimasi jumlah pemilik lebih dari nol dan sedikitnya 10 ulasan total, sehingga tersisa 55.708 baris.

Gambar 4. Penyaringan ketiga



```
[70]
# PROSES LOGIKA FILTERING LEVEL 3 (Validasi Waktu Bermain)
# P1 — Muat data dari checkpoint Level 2
df_stage_b = muat_checkpoint('checkpoints_filtering', '02_filter_level2_popularitas.csv',
                             kolom_list_literal=['genres', 'categories', 'tags'])
baris_masuk = len(df_stage_b)
KOLOM_PLAYTIME = ['average_playtime_forever', 'average_playtime_2weeks',
                  'median_playtime_forever', 'median_playtime_2weeks']
# P2 — Terapkan 3 kriteria validasi playtime
is_valid_range = (df_stage_b[KOLOM_PLAYTIME] >= 0).all(axis=1)
is_konsisten_average = df_stage_b['average_playtime_forever'] >= df_stage_b['average_playtime_2weeks']
is_konsisten_median  = df_stage_b['median_playtime_forever'] >= df_stage_b['median_playtime_2weeks']
is_logically_consistent = is_konsisten_average & is_konsisten_median
is_active = df_stage_b['average_playtime_forever'] > 0
# P3 — Gabungkan kriteria (integritas + konsistensi) DAN aktivitas nyata
valid_data = is_valid_range & is_logically_consistent
valid_playtime = valid_data & is_active
```

Tahap ketiga dimaksudkan untuk menyaring rentang nilai tidak negatif, konsistensi logis antara *playtime* sepanjang masa dan dua minggu terakhir, serta bukti aktivitas nyata (*playtime* >0) — digabung melalui AND bertingkat. Struktur ini memastikan hanya baris yang lolos ketiga syarat integritas data sekaligus yang dipertahankan. Tahap validasi waktu bermain memeriksa nilai waktu bermain yang tidak negatif, konsistensi antara waktu bermain

sepanjang masa dan dua minggu terakhir, serta rata-rata waktu bermain sepanjang masa yang lebih besar dari nol; tahap ini menghasilkan 22.739 baris

Gambar 5. Penyaringan keempat

```
BATAS_HARGA_MAX = 1000
is_price_valid_range = (df_kerja['price'] >= 0) & (df_kerja['price'] <= BATAS_HARGA_MAX)
is_price_missing = df_kerja['price'].isna()
```

Kolom *price* divalidasi berada dalam rentang wajar (0–1000 dolar AS), melengkapi dua kriteria lain pada tahap ini (kelengkapan kolom kunci dan deteksi duplikat) yang bersama-sama membentuk gerbang penyaringan terakhir sebelum populasi bersih ditetapkan. kelengkapan atribut yang dibutuhkan untuk analisis, serta duplikasi berdasarkan *AppID* dan kombinasi nama dengan tanggal rilis. Setelah seluruh kriteria diterapkan, populasi akhir terdiri atas 22.659 game.

Perubahan jumlah data paling besar terjadi pada penyaringan popularitas dan waktu bermain. Pola tersebut menunjukkan bahwa banyak entri dalam *dataset* mentah tidak memenuhi ambang aktivitas yang ditetapkan penelitian. Temuan ini tidak dimaknai sebagai bukti bahwa game tersebut tidak memiliki pemain sama sekali, melainkan sebagai konsekuensi dari kriteria operasional yang digunakan untuk membentuk populasi analisis yang memiliki informasi popularitas dan keterlibatan yang memadai.

C. Seleksi dan Rekayasa Fitur

Klasterisasi menggunakan tiga fitur numerik, yaitu estimasi jumlah pemilik (*estimated_owners*), harga (*price*), dan rata-rata waktu bermain sepanjang masa (*average_playtime_forever*). Ketiganya dipilih untuk mewakili dimensi yang berbeda: jangkauan pasar, tingkat harga, dan keterlibatan pemain. Pembatasan pada tiga fitur dilakukan agar ruang klasterisasi tetap mudah diinterpretasikan sekaligus mempertahankan informasi utama yang diperlukan untuk membedakan karakteristik *game*. Hubungan antarfitur kemudian diperiksa secara statistik sebelum seluruh fitur digunakan bersama dalam pemodelan.

Atribut I pada dataset berbentuk rentang teks, misalnya "0 - 20000". Rentang tersebut dikonversi menjadi nilai numerik menggunakan titik tengah agar dapat diperlakukan sebagai variabel kontinu dalam perhitungan jarak. Hubungan antarketiga fitur diperiksa menggunakan koefisien korelasi *Spearman* (ρ), yaitu ukuran asosiasi berbasis peringkat yang sesuai untuk menilai hubungan monoton tanpa mensyaratkan linearitas pada skala asli [14]. Nilai korelasi absolut tertinggi adalah 0,398 antara estimasi kepemilikan dan waktu bermain, sedangkan pasangan fitur lainnya memiliki korelasi yang lebih rendah. Hasil tersebut menunjukkan bahwa tidak terdapat hubungan monoton yang kuat sehingga ketiga fitur tetap dipertahankan dalam ruang klasterisasi.

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)} \quad (1)$$

dengan d_i menyatakan selisih peringkat dua variabel pada observasi ke-i, sedangkan n menyatakan jumlah observasi. Persamaan ini digunakan untuk memperoleh koefisien korelasi *Spearman* antarpasangan fitur yang selanjutnya disajikan bersama hasil analisis karakteristik fitur.

D. Pra-Pemrosesan

Pada skala asli, ketiga fitur memiliki distribusi yang sangat menceng, dengan *skewness* 34,85 pada estimasi kepemilikan, 4,26 pada harga, dan 115,12 pada rata-rata waktu bermain. Distribusi seperti ini dapat menyebabkan sebagian kecil nilai ekstrem memberikan kontribusi yang terlalu besar terhadap perhitungan jarak. Penelitian ini menerapkan transformasi *Yeo-Johnson* untuk mengurangi kemencengan karena transformasi tersebut dapat digunakan pada data yang memuat nilai nol [13]. Karakteristik tersebut penting bagi fitur harga yang mencakup game gratis. Parameter transformasi diestimasi secara komputasional menggunakan *PowerTransformer*.

$$\psi(y, \lambda) = \begin{cases} \dfrac{(y+1)^{\lambda} - 1}{\lambda}, & \lambda \neq 0, y \geq 0 \\ \ln(y+1), & \lambda = 0, y \geq 0 \\ -\dfrac{(-y+1)^{2-\lambda} - 1}{2-\lambda}, & \lambda \neq 2, y < 0 \\ -\ln(-y+1), & \lambda = 2, y < 0 \end{cases} \quad (2)$$

Persamaan (2) memperlihatkan formulasi *Yeo-Johnson* untuk nilai non-negatif maupun negatif. Berbeda dari transformasi logaritmik standar, *Yeo-Johnson* tetap terdefinisi ketika y=0, sehingga dapat diterapkan tanpa

mengeluarkan observasi game dengan harga nol. Nilai parameter λ diestimasi oleh *PowerTransformer* untuk setiap fitur sebelum proses standardisasi dilakukan.

Gambar 6. Proses Pra-proses

```
[89]
# PROSES LOGIKA PREPROCESSING
# P1 — Transformasi Yeo-Johnson (menormalkan distribusi skewed)
from sklearn.preprocessing import PowerTransformer, StandardScaler
pt = PowerTransformer(method='yeo-johnson', standardize=False)
X_transformed = pt.fit_transform(df_features[FEATURE_COLS])
df_transformed = pd.DataFrame(X_transformed, columns=FEATURE_COLS, index=df_features.index)
# P2 — Scaling standar (menyamakan skala antar fitur)
scaler = StandardScaler()
X_scaled_arr = scaler.fit_transform(df_transformed)
df_scaled = pd.DataFrame(X_scaled_arr, columns=FEATURE_COLS, index=df_features.index)
```

Urutan baris kode ini secara eksplisit menunjukkan bahwa penyekalaan dilakukan pada data yang bentuk distribusinya sudah simetris, sehingga parameter *mean* dan deviasi standar yang dihasilkan tidak lagi didominasi nilai ekstrem.Transformasi *Yeo-Johnson* diterapkan sebelum *StandardScaler*. Tahap pertama ditujukan untuk mengurangi kemencengan distribusi, sedangkan tahap kedua menyetarakan skala fitur menjadi rata-rata nol dan simpangan baku satu agar perhitungan jarak tidak didominasi oleh perbedaan satuan. Setelah kedua tahap diterapkan, *skewness* turun menjadi 0,1256 pada estimasi kepemilikan, 0,0096 pada harga, dan 0,0315 pada rata-rata waktu bermain. Perubahan tersebut menunjukkan bahwa distribusi fitur menjadi jauh lebih simetris sebelum digunakan dalam klasterisasi.

E. Penentuan K Optimal

Jumlah klaster dievaluasi pada K=2 hingga K=10 menggunakan Inertia, *Silhouette Score*, dan *Davies-Bouldin Index* (DBI).

Gambar 7. Proses Menentukan K Optimal

```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score, davies_bouldin_score
K_range = range(2, 11)
hasil_k = []
for k in K_range:
    km = KMeans(n_clusters=k, init='k-means++', n_init=10, random_state=42)
    labels = km.fit_predict(X_scaled)
    hasil_k.append({
        'K': k, 'Inertia': km.inertia_,
        'Silhouette': silhouette_score(X_scaled, labels),
        'Davies-Bouldin': davies_bouldin_score(X_scaled, labels)
    })
```

` Desain ini menjamin kesepuluh kandidat K dievaluasi pada kondisi komputasi yang identik, sehingga perbandingan pada Tabel 1 dapat dipertanggungjawabkan secara metodologis.Inertia digunakan untuk mengamati penurunan variasi dalam klaster melalui pendekatan *Elbow*, tetapi penentuan titik siku bersifat heuristik dan tidak selalu menghasilkan batas yang tegas [11], [12]. *Silhouette Score* menilai kedekatan objek terhadap klasternya sekaligus pemisahan dari klaster terdekat [8], [9]. DBI melengkapi evaluasi dengan membandingkan dispersi internal dan pemisahan antarklaster [10]. Ketiga ukuran tersebut dianalisis secara bersama agar pemilihan jumlah klaster tidak bergantung pada satu indikator.

$$\text{Inertia} = \sum_{i=1}^{k} \sum_{x \in C_i} \| x - \mu_i \|^2 \quad (3)$$

Dengan k menyatakan jumlah klaster, C_i merupakan himpunan anggota klaster ke-i, dan μ_i merupakan *centroid* klaster tersebut. Nilai Inertia yang semakin kecil menunjukkan semakin kecilnya jumlah kuadrat jarak anggota terhadap pusat klasternya, tetapi penurunannya tetap perlu ditafsirkan bersama metrik lain.

$$s(i) = \frac{b(i) - a(i)}{\max\{a, (i)\ b(i)\}} \quad (4)$$

Dengan a(i) menyatakan rata-rata jarak objek i terhadap objek lain dalam klaster yang sama, sedangkan b(i) adalah rata-rata jarak terkecil objek i terhadap klaster lain yang terdekat. Nilai *Silhouette* berada pada rentang -1 hingga 1; nilai yang lebih mendekati 1 menunjukkan objek ditempatkan lebih sesuai dengan klasternya.

$$\text{DBI} = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right) \quad (5)$$

Dengan σ_i menyatakan dispersi rata-rata klaster ke-i terhadap centroid c_i, sedangkan d(c_i,c_j) menyatakan jarak antara pusat klaster i dan j. Pada *Davies-Bouldin Index*, nilai yang lebih rendah menunjukkan kombinasi dispersi internal dan pemisahan antarklaster yang lebih baik [10]. Dalam evaluasi K=2 hingga K=10, DBI berfluktuasi sehingga tidak digunakan sebagai kriteria tunggal.

Silhouette Score digunakan sebagai indikator utama, sedangkan pola Inertia dan DBI digunakan sebagai pertimbangan pendukung [8]–[12]. Nilai *Silhouette* tertinggi diperoleh pada K=3 sebesar 0,2913. Setelah K=3, skor menurun hingga K=9 dan hanya meningkat sedikit pada K=10. Kurva Inertia tidak menunjukkan titik siku yang tegas, sementara DBI berfluktuasi sepanjang rentang pengujian. Berdasarkan kombinasi hasil tersebut, K=3 ditetapkan sebagai jumlah klaster final dan digunakan secara konsisten pada *K-Means* maupun *K-Medoids*.

F. Algoritma Klasterisasi

K-Means dan *K-Medoids* dibandingkan pada kondisi eksperimen yang sama agar hasil evaluasi dapat ditelusuri secara konsisten. Kedua model menerima fitur terskala, urutan observasi, dan jumlah klaster yang identik. Dengan pengendalian tersebut, perbedaan keluaran dapat dianalisis terutama berdasarkan mekanisme pembentukan pusat klaster masing-masing algoritma.

Gambar 8. Fitting *K-Means*

```
[97]
✓ 15
d
    # P1 — Fit K-Means
    kmeans_model = KMeans(n_clusters=K_FINAL, init='k-means++', n_init=10, random_state=RANDOM_STATE)
    labels_kmeans = kmeans_model.fit_predict(X_scaled)
    sil_kmeans = silhouette_score(X_scaled, labels_kmeans)
    dbi_kmeans = davies_bouldin_score(X_scaled, labels_kmeans)
    vc_km = pd.Series(labels_kmeans).value_counts().sort_index()
```

K-Means dan *K-Medoids* sama-sama merupakan algoritma partisi berbasis prototipe, tetapi menggunakan representasi pusat faktual yang berbeda. *K-Means* menentukan pusat factual sebagai *centroid* dan meminimalkan jumlah kuadrat jarak *Euclidean* setiap objek terhadap *centroid* klasternya. Fungsi objektif tersebut dinyatakan pada Persamaan (6).

$$\text{Min} \sum k \sum_{x \in C_i} \| x - \mu_i \|^2 \quad (6)$$

Fungsi objektif K-Means pada Persamaan (6) secara matematis berkaitan dengan Inertia yang digunakan dalam evaluasi jumlah faktual. Setelah K ditetapkan, fungsi tersebut menjadi dasar optimasi *K-Means*. Sebaliknya, *K-Medoids*/*PAM* menggunakan *medoid*, yaitu observasi faktual yang dipilih untuk meminimalkan total ketidakmiripan anggota terhadap pusat faktual, sebagaimana dinyatakan pada Persamaan (7).

$$\text{Min} \sum k \sum_{x \in C_i} d\,(x, m_i) \quad (7)$$

Pada Persamaan (7), m_i ∈ C_i menunjukkan bahwa medoid harus merupakan salah satu observasi yang terdapat dalam faktual. Hal ini berbeda dari *centroid* μ_i yang diperoleh melalui rata-rata seluruh anggota. Dengan demikian, pusat *K-Medoids* selalu dapat dipetakan faktual ke objek faktual, sedangkan *centroid K-Means* tidak harus berimpit dengan observasi tertentu [1], [3]. Perbedaan tersebut menjadi dasar konseptual untuk membandingkan perilaku kedua algoritma pada data penelitian.

Gambar 9. Fitting *K-Medoids*

```
[99]
✓ 1m
    # P1 — Fit K-Medoids
    dist_matrix = pairwise_distances(X_scaled, metric='euclidean')
    km_result = kmedoids.fasterpam(dist_matrix, K_FINAL, random_state=RANDOM_STATE)
    labels_kmedoids = km_result.labels
    sil_kmedoids = silhouette_score(X_scaled, labels_kmedoids)
    dbi_kmedoids = davies_bouldin_score(X_scaled, labels_kmedoids)
    vc_kmd = pd.Series(labels_kmedoids).value_counts().sort_index()
```

Implementasi *K-Medoids* menggunakan *FasterPAM* untuk mengurangi biaya komputasi dibandingkan *PAM* klasik. Pemilihan ini mempertimbangkan populasi akhir sebanyak 22.659 observasi dan kebutuhan untuk menjalankan perbandingan pada data yang sama dengan *K-Means*. *FasterPAM* mempercepat proses evaluasi pertukaran medoid dalam keluarga PAM tanpa mengubah prinsip pusat klaster berbasis observasi aktual [4].

Sebelum pemodelan, indeks *AppID* pada data fitur terskala dicocokkan dengan data atribut hasil penyaringan untuk memastikan setiap baris merujuk pada game yang sama. Langkah ini diperlukan karena label hasil klasterisasi selanjutnya digabungkan kembali dengan atribut asli untuk penyusunan profil klaster. Kedua algoritma dijalankan pada K=3 dengan *random_state*=42 untuk menjaga reproduksibilitas proses yang melibatkan inisialisasi acak. Kualitas hasil dibandingkan menggunakan *Silhouette Score* dan DBI yang sama dengan metrik pada evaluasi jumlah klaster.

G. Validasi dan Pelabelan Klaster

Label klaster yang dihasilkan algoritma berupa kode numerik 0, 1, dan 2 sehingga belum memiliki makna substantif. Oleh karena itu, keluaran model melalui dua tahap lanjutan: pengujian perbedaan distribusi fitur antarklaster dan penyusunan label berdasarkan profil numerik masing-masing klaster. Tahap ini digunakan untuk menilai apakah kelompok yang terbentuk memiliki perbedaan yang dapat dipertanggungjawabkan secara statistik dan dapat diinterpretasikan dalam konteks karakteristik *game*.

Gambar 10. Implementasi Uji *Kruskal-Wallis* per Fitur

```
[105]
✓ 0 d
    # P1 — Uji Kruskal-Wallis tiap fitur antar cluster (skala asli, bukan z-score)
    from scipy.stats import kruskal

    hasil_uji = []
    for kol in FEATURE_COLS:
        grup = [df_result[df_result['cluster_kmedoids'] == cid][kol]
                for cid in sorted(df_result['cluster_kmedoids'].unique())]
        stat, p_value = kruskal(*grup)
        hasil_uji.append({'fitur': kol, 'H_statistic': round(stat, 4), 'p_value': p_value,
                          'signifikan': 'YA (p < 0.05)' if p_value < 0.05 else 'TIDAK'})

    df_uji_statistik = pd.DataFrame(hasil_uji)
```

Populasi dipecah menjadi tiga kelompok berdasarkan label klaster, lalu klaster H dan nilai-p dihitung untuk menguji perbedaan distribusi antar kelompok. Hasil iterasi ini langsung menghasilkan angka pada Tabel Hasil Uji *Kruskal-Wallis.* Perbedaan distribusi antarklaster diuji menggunakan *Kruskal-Wallis* pada setiap fitur dalam skala asli. *Uji Kruskal-Wallis* merupakan metode non-parametrik berbasis peringkat untuk membandingkan lebih dari dua kelompok [15]. Metode ini dipilih karena distribusi fitur asli sangat tidak simetris dan tidak digunakan sebagai dasar pemilihan algoritma. Hasil pengujian berfungsi sebagai validasi klaster bahwa fitur pembentuk klaster menunjukkan distribusi yang berbeda pada kelompok yang dihasilkan model terpilih.

$$H = \frac{12}{N(N+1)} \sum_{j=1}^{k} \frac{R_j^2}{n_j} - 3(N+1) \quad (8)$$

Dengan N menyatakan jumlah seluruh observasi, n_j jumlah anggota klister ke-j, dan R_j jumlah peringkat anggota klaster ke-j setelah seluruh observasi diberi peringkat klaster. Pendekatan berbasis peringkat dipilih karena tidak mensyaratkan normalitas pada skala data asli. Dengan demikian, pengujian dapat digunakan untuk menilai perbedaan distribusi fitur antarklaster tanpa bergantung pada asumsi parametrik yang tidak sesuai dengan karakteristik data penelitian. Setelah validasi , profil setiap klaster disusun menggunakan posisi rata-rata fitur terhadap rata-rata populasi yang dinyatakan dalam *z-score*.

$$z = \frac{x - \mu}{\sigma} \quad (9)$$

Pada Persamaan (9), μ dan σ masing-masing menyatakan rata-rata dan simpangan baku populasi untuk fitur yang dianalisis, sedangkan x merupakan rata-rata fitur pada suatu klaster. Ambang $|z| \geq 0{,}5$ digunakan sebagai kriteria operasional untuk mengidentifikasi karakteristik yang menonjol atau rendah. Ambang ini digunakan untuk membantu interpretasi profil, bukan sebagai ukuran *effect size* atau padanan *Cohen's* d. Label. ditentukan dari kombinasi pola ketiga fitur dan kemudian diperiksa terhadap contoh *game* pada setiap klister

Gambar 11. Fungsi Klasifikasi Dimensi Berbasis *Threshold*.

```
[107]
✓ 0d
# P1 — Fungsi klasifikasi 1 dimensi berdasarkan threshold z-score
def klasifikasi_dimensi(z_value, label_tinggi, label_rendah, threshold=THRESHOLD):
    if z_value >= threshold:
        return label_tinggi
    elif z_value <= -threshold:
        return label_rendah
    else:
        return None
```

Fungsi klasifikasi_dimensi mengimplementasikan aturan ambang $|z| \geq 0{,}5$ sesuai Persamaan (9): mengembalikan label "tinggi" jika *z-score* melampaui ambang positif, "rendah" jika melampaui ambang negatif, dan netral (None) di antara keduanya. Logika eksplisit ini menjadi dasar komputasional label deskriptif seperti "*Populer-Premium-Engagement* Tinggi" pada Bagian III.F guna memastikan pelabelan bersifat sistematis dan dapat direproduksi, bukan interpretasi visual subjektif.Pemeriksaan kualitatif dilakukan sebagai pelengkap hasil numerik dengan mengambil sampel game dari masing-masing klaster dan membandingkan karakteristiknya terhadap label yang diberikan. Pemeriksaan ini tidak menggantikan validasi statistik, tetapi digunakan untuk menilai keterbacaan interpretasi dalam konteks domain. Dengan menggabungkan profil *z-score*, hasil *Kruskal-Wallis*, dan contoh observasi aktual, pelabelan klaster disusun berdasarkan bukti kuantitatif sekaligus konteks substantif.

III. HASIL DAN PEMBAHASAN

A. Ringkasan Alur Penyaringan Data

Empat tahap penyaringan menghasilkan perubahan jumlah observasi sebagaimana dirangkum pada Tabel 1 Tabel tersebut memperlihatkan jumlah data yang tersisa pada setiap tahap relatif terhadap 136.080 entri mentah.

Tabel 1. Hasil penyaringan data bertingkat

Tahap	Jumlah Baris	Sisa dari Data Mentah
0. Akuisisi Data (mentah)	136.080	100,0%
1. Real Game Filter	125.861	92,5%
2. Popularitas Ketat	55.708	40,9%
3. Validasi Waktu Bermain	22.739	16,7%
4. Validasi Harga dan Kelengkapan	22.659	16,7%

Tabel 1 menunjukkan bahwa penyusutan data tidak terjadi secara merata. *Real Game Filter* mengeluarkan sekitar 7,5% entri awal, sedangkan penurunan terbesar terjadi pada tahap Popularitas Ketat dan Validasi Waktu Bermain. Setelah dua tahap tersebut, jumlah observasi turun menjadi 22.739 atau 16,7% dari data mentah. Tahap validasi harga, kelengkapan atribut, dan duplikasi hanya mengurangi 80 entri tambahan sehingga populasi akhir berjumlah 22.659 game. Pola ini menunjukkan bahwa kriteria aktivitas dan kelengkapan informasi memberikan pengaruh terbesar terhadap pembentukan populasi analisis.

B. Karakteristik Fitur Sebelum dan Sesudah Transformasi

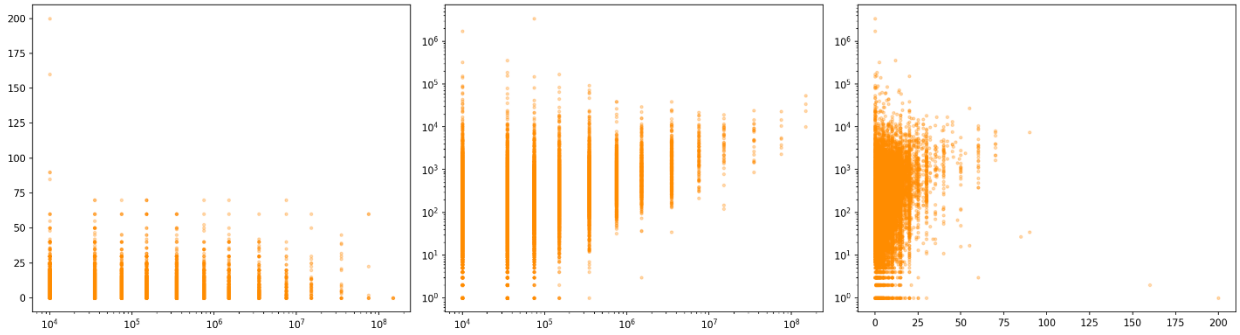
Sebelum klasterisasi dilakukan, hubungan antarketiga fitur diperiksa untuk mengetahui tingkat keterkaitan monotonnya, kemudian bentuk distribusinya dievaluasi melalui skewness. Kedua pemeriksaan tersebut diperlukan agar pemilihan fitur dan kebutuhan transformasi didasarkan pada karakteristik data yang terukur.

Tabel 2. Matriks korelasi *Spearman* antarfitur klasterisasi

	Estimasi Kepemilikan	Harga	Waktu Bermain
Estimasi Kepemilikan	1,000	-0,041	0,398
Harga	-0,041	1,000	0,247

Waktu Bermain	0,398	0,247	1,000

Gambar 12. Sebaran bivariat tiga fitur klasterisasi pada skala asli.



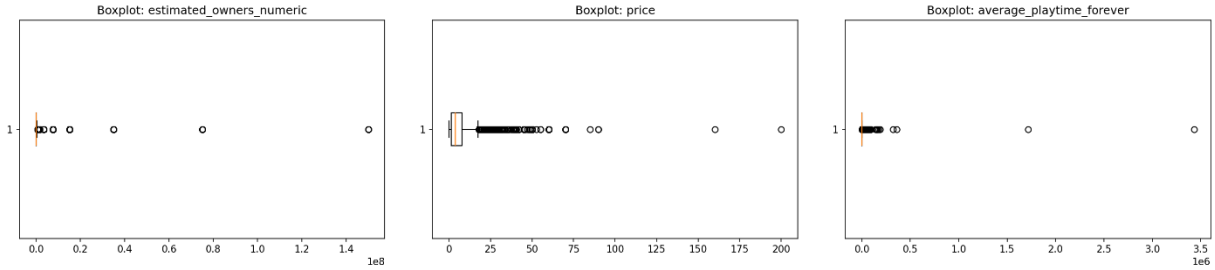
Matriks korelasi Spearman pada Tabel 2 menunjukkan nilai ρ sebesar -0,041 antara estimasi kepemilikan dan harga, 0,398 antara estimasi kepemilikan dan waktu bermain, serta 0,247 antara harga dan waktu bermain. Seluruh nilai absolut berada di bawah 0,7 sehingga tidak terdapat hubungan monoton yang kuat antarpasangan fitur. Gambar 3.1 memperlihatkan pola sebaran bivariat yang selaras dengan hasil tersebut: tidak tampak hubungan linear yang dominan pada ketiga pasangan fitur.

Tabel 3. Skewness fitur sebelum dan sesudah transformasi *Yeo-Johnson* dan standardisasi

Fitur	Skewness Sebelum	Skewness Sesudah Transformasi dan Standardisasi
Estimasi Kepemilikan	34,85	0,1256
Harga	4,26	0,0096
Waktu Bermain	115,12	0,0315

Tabel 3 memperlihatkan penurunan skewness yang besar pada seluruh fitur setelah transformasi dan standardisasi. Skewness estimasi kepemilikan turun dari 34,85 menjadi 0,1256, harga dari 4,26 menjadi 0,0096, dan waktu bermain dari 115,12 menjadi 0,0315. Gambar 12 menampilkan bentuk distribusi pada skala asli, yang memperlihatkan konsentrasi observasi pada rentang rendah dengan ekor kanan yang panjang. Hasil numerik dan visual tersebut mendukung penggunaan transformasi sebelum perhitungan jarak dilakukan.

Gambar 13. Distribusi fitur pada skala asli sebelum transformasi *Yeo-Johnson*.



Perubahan paling ekstrem terjadi pada rata-rata waktu bermain dan estimasi kepemilikan. Tanpa transformasi, nilai yang sangat besar pada sebagian kecil observasi berpotensi mendominasi jarak *Euclidean* dan mengurangi representasi struktur mayoritas data. Setelah transformasi, ketiga fitur memiliki distribusi yang lebih simetris dan skala yang setara untuk digunakan dalam pemodelan.

C. Hasil Evaluasi K Optimal

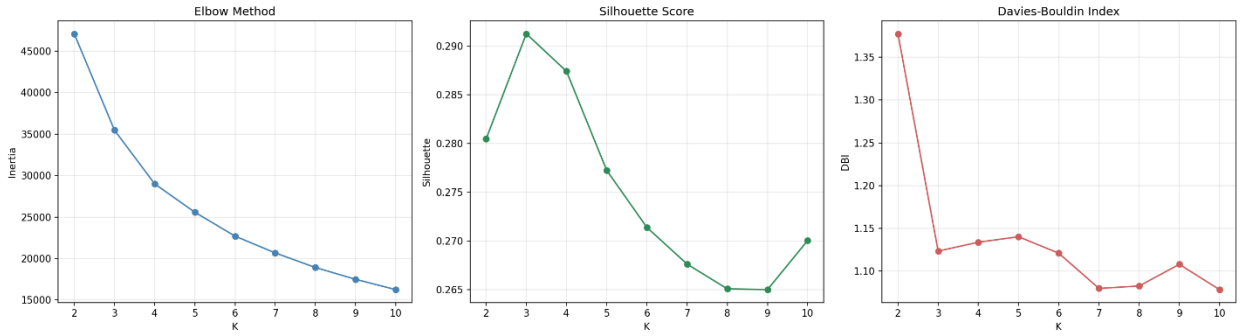
Evaluasi K=2 hingga K=10 menghasilkan *Silhouette Score* tertinggi pada K=3 sebesar 0,2913. Pada nilai K yang sama, DBI bernilai 1,1234. Inertia terus menurun ketika K bertambah, tetapi kurvanya tidak membentuk titik siku yang tegas. Karena *Silhouette Score* secara langsung menilai kohesi dan pemisahan objek [8], [9], sedangkan DBI dan *Elbow* digunakan sebagai informasi pendukung [10]–[12], K=3 dipilih sebagai jumlah klaster yang digunakan pada pemodelan berikutnya.

Tabel 4. Evaluasi jumlah klaster K=2 hingga K=10

K	Inertia	Silhouette Score	Davies-Bouldin Index
2	47.097,11	0,2805	1,3775
3	35.482,74	0,2913	1,1234
4	29.006,89	0,2874	1,1338
5	25.585,99	0,2773	1,1402
6	22.705,38	0,2714	1,1209
7	20.678,83	0,2676	1,0797
8	18.920,74	0,2651	1,0826
9	17.496,79	0,2650	1,1079
10	16.251,79	0,2701	1,0783

Nilai lengkap ketiga metrik pada Tabel 4 divisualisasikan pada Gambar 14 *Silhouette Score* meningkat dari 0,2805 pada K=2 menjadi 0,2913 pada K=3, kemudian turun menjadi 0,2874 pada K=4 dan terus berada di bawah nilai K=3 hingga K=10. Selisih *Silhouette* antara K=3 dan K=4 adalah 0,0039, sehingga keputusan juga dipertimbangkan bersama pola DBI dan Inertia.

Gambar 14. Perubahan Inertia, Silhouette Score, dan Davies-Bouldin Index pada K=2 hingga K=10.



DBI turun dari 1,3775 pada K=2 menjadi 1,1234 pada K=3, kemudian berfluktuasi pada K berikutnya dan mencapai nilai terendah 1,0783 pada K=10. Nilai DBI yang lebih kecil mengindikasikan rasio dispersi dan pemisahan yang lebih baik [10], tetapi metrik ini tidak menunjukkan titik keputusan yang konsisten dengan seluruh rentang. Inertia juga turun secara monoton seiring bertambahnya K, sehingga tidak memberikan siku yang jelas. Dengan mempertimbangkan puncak *Silhouette* dan tidak adanya keputusan tunggal yang tegas dari dua indikator lainnya, K=3 dipertahankan sebagai konfigurasi final.

D. Perbandingan K-Means dan K-Medoids

Setelah jumlah klaster ditetapkan pada K=3, *K-Means* dan *K-Medoids* dijalankan pada data terskala dan urutan observasi yang sama. Perbandingan difokuskan pada *Silhouette* Score, DBI, serta distribusi jumlah anggota untuk menilai struktur pengelompokan yang dihasilkan oleh kedua metode.

Tabel 5. Perbandingan metrik evaluasi *K-Means* dan *K-Medoids* pada K=3

Model	Silhouette Score	Davies-Bouldin Index
K-Means	0,2913	1,1234
K-Medoids (FasterPAM)	0,2922	1,1493

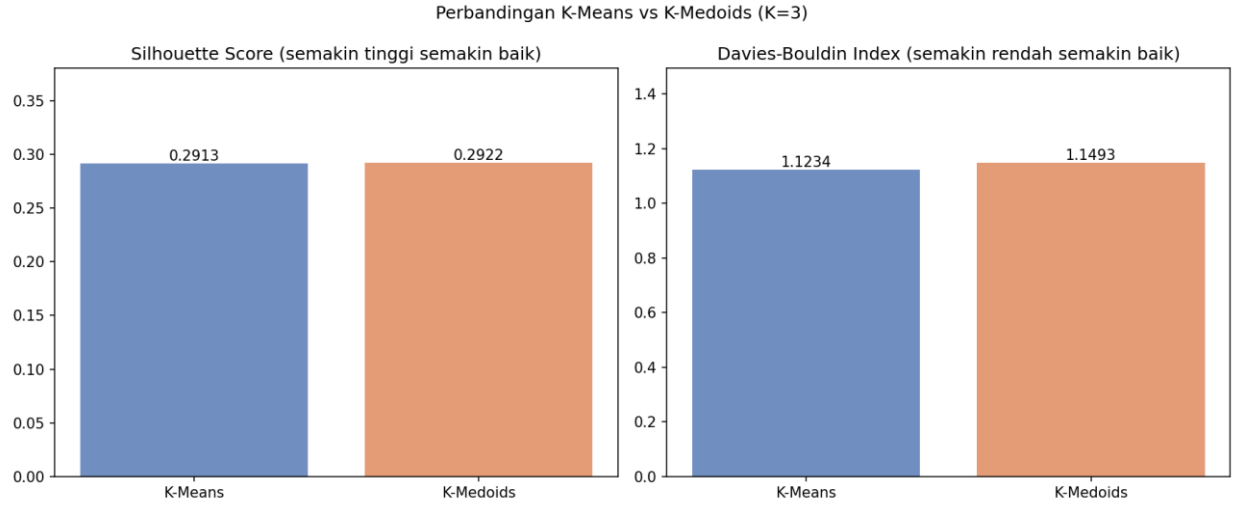
Tabel 6. Distribusi anggota klaster *K-Means* dan *K-Medoids*

Klaster	K-Means	K-Medoids
0	6.864	7.179
1	7.112	7.672
2	8.683	7.808

Tabel 5 menunjukkan bahwa perbedaan kedua model pada metrik internal relatif kecil. Untuk memperjelas selisih tersebut, *Silhouette Score* dan DBI divisualisasikan berdampingan pada Gambar 15, sedangkan komposisi jumlah anggota setiap klaster dirangkum pada Tabel 6.

K-Medoids memperoleh Silhouette Score 0,2922, lebih tinggi 0,0009 daripada *K-Means* yang memperoleh 0,2913. Sebaliknya, DBI *K-Means* sebesar 1,1234 lebih rendah daripada *K-Medoids* sebesar 1,1493; untuk metrik ini nilai yang lebih rendah menunjukkan hasil yang lebih baik. Dengan demikian, kedua metrik memberikan keunggulan pada model yang berbeda dan selisihnya relatif kecil, sehingga pemilihan model tidak didasarkan pada satu skor saja.

Gambar 15. Perbandingan *Silhouette Score* dan *Davies-Bouldin Index K-Means* dan *K-Medoids* pada K=3



Distribusi anggota pada Tabel 6 memberikan informasi tambahan mengenai struktur hasil pengelompokan. *K-Means* menghasilkan 6.864, 7.112, dan 8.683 anggota pada Klaster 0, 1, dan 2, sehingga selisih antara klaster terbesar dan terkecil mencapai 1.819 game. *K-Medoids* menghasilkan 7.179, 7.672, dan 7.808 anggota, dengan selisih terbesar-terkecil 629 game. Pada dataset ini, hasil *K-Medoids* karena itu memiliki ukuran klaster yang lebih seimbang.

Perbedaan distribusi anggota dapat ditinjau bersama mekanisme pusat klaster kedua algoritma. *K-Means* menggunakan *centroid* berbasis rata-rata, sedangkan *K-Medoids* menggunakan observasi aktual sebagai medoid [1], [3]. *Medoid* tidak ditentukan melalui rata-rata koordinat seluruh anggota, sehingga pusat klasternya memiliki karakteristik yang berbeda ketika data mengandung observasi ekstrem. *FasterPAM* mempertahankan prinsip *medoid* dengan optimasi yang lebih efisien [4]. Meskipun demikian, ukuran klaster yang lebih seimbang bukan kriteria universal untuk menyatakan satu metode selalu lebih unggul; performa relatif tetap bergantung pada karakteristik data dan ukuran evaluasi [5]–[7].

Dengan mempertimbangkan *Silhouette Score* yang sedikit lebih tinggi dan distribusi anggota yang lebih merata, *K-Medoids (FasterPAM)* dipilih sebagai model utama, meskipun DBI-nya sedikit lebih tinggi daripada *K-Means*. Keputusan ini berlaku untuk dataset dan konfigurasi penelitian yang digunakan, bukan sebagai klaim keunggulan *K-Medoids* secara umum. Analisis profil dan validasi statistik selanjutnya menggunakan label klaster dari model *K-Medoids*.

E. Validasi Statistik Klaster

Validasi statistik dilakukan untuk menilai apakah distribusi estimasi kepemilikan, harga, dan rata-rata waktu bermain berbeda di antara tiga klaster hasil *K-Medoids*. Pengujian menggunakan *Kruskal-Wallis* pada skala data asli karena distribusi fitur tidak memenuhi karakteristik yang mendukung penggunaan uji parametrik.

Tabel 7. Hasil uji *Kruskal-Wallis* pada tiga fitur klasterisasi

Fitur	H-statistic	p-value	Signifikan
Estimasi Kepemilikan	14.238,33	<0,001	Ya
Harga	12.330,41	<0,001	Ya
Rata-rata Waktu Bermain	6.468,94	<0,001	Ya

Tabel 7 menunjukkan p-*value* <0,001 untuk estimasi kepemilikan, harga, dan rata-rata waktu bermain. Dengan tingkat signifikansi 0,05, hipotesis kesamaan distribusi antarklaster ditolak untuk ketiga fitur. *H-statistic* terbesar terdapat pada estimasi kepemilikan sebesar 14.238,33, diikuti harga sebesar 12.330,41 dan rata-rata waktu bermain sebesar 6.468,94. Hasil ini menunjukkan bahwa klaster yang terbentuk memiliki perbedaan distribusi yang signifikan

pada seluruh fitur pembentuknya, sehingga interpretasi profil klaster dapat dilanjutkan dengan dasar statistik yang memadai.

F. Profil Klaster Terpilih

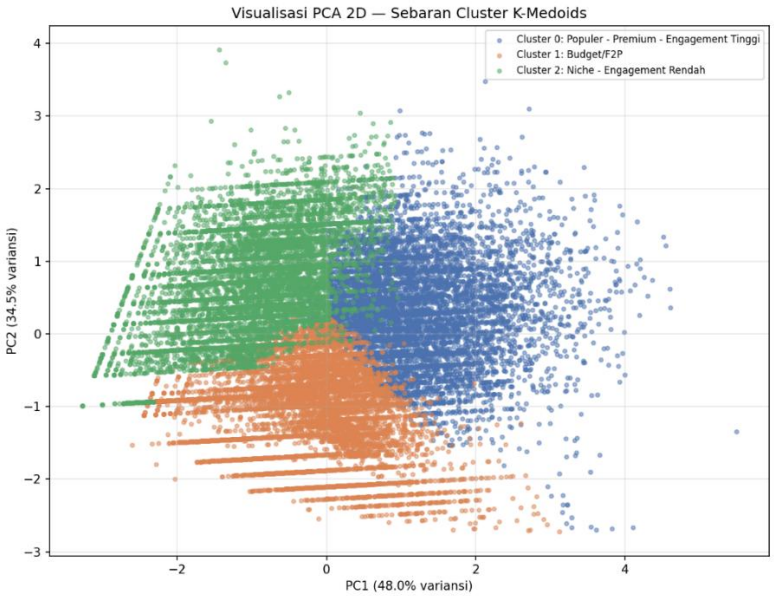
Profil klaster disusun menggunakan nilai pada skala asli agar perbedaan antar kelompok dapat ditafsirkan dalam satuan yang mudah dipahami. Rata-rata fitur, jumlah anggota, dan proporsi setiap klaster disajikan pada Tabel 8, kemudian dilengkapi visualisasi struktur klaster dan pemeriksaan sampel game.

Tabel 8. Profil klaster *K-Medoids* pada skala asli

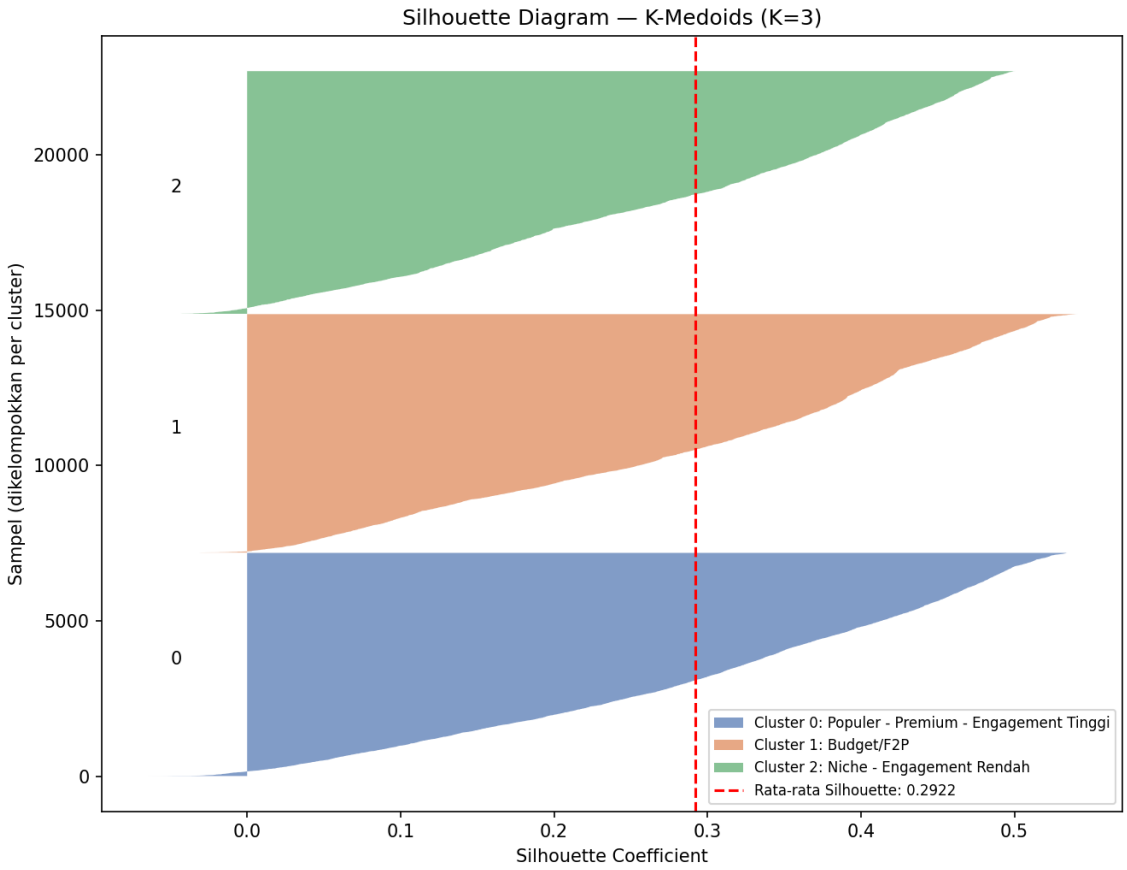
Klaster	Label	Estimasi Pemilik (rata-rata)	Harga (rata-rata)	Waktu Bermain (rata-rata)	Jumlah Game	Persentase
0	Populer-Premium-Engagement Tinggi	760.647	US$9,92	2.238 menit	7.179	31,68%
1	Budget/F2P	288.355	US$1,17	369 menit	7.672	33,86%
2	Niche-Engagement Rendah	13.734	US$6,15	230 menit	7.808	34,46%

Struktur klaster ditinjau dari dua sudut visual. Gambar 16 memproyeksikan data tiga dimensi ke dua komponen utama menggunakan *Principal Component Analysis (PCA)*, sedangkan Gambar 17 menampilkan distribusi nilai *silhouette* setiap klaster. PCA digunakan untuk membantu pembacaan posisi relatif antarklaster dalam ruang dua dimensi, sedangkan diagram *silhouette* menunjukkan kualitas penempatan objek terhadap klasternya.

Gambar 16. Proyeksi dua dimensi klaster *K-Medoids* menggunakan *Principal Component Analysis (PCA)*.



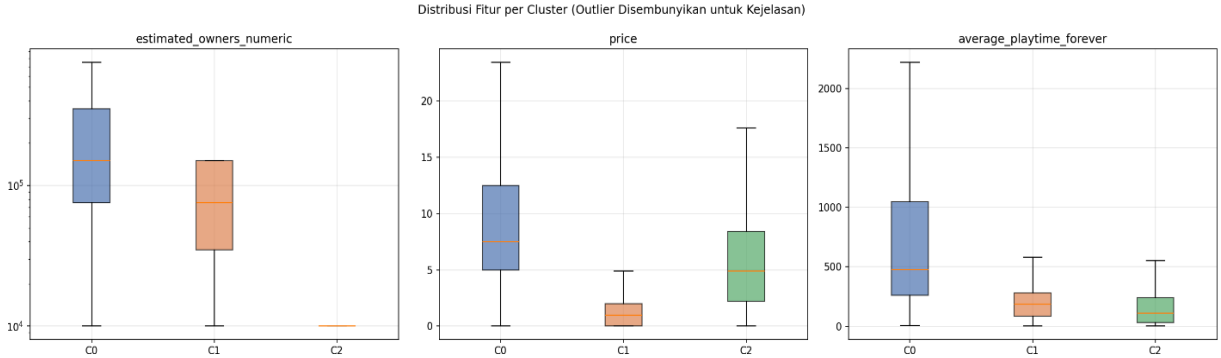
Gambar 17. Diagram *silhouette* untuk setiap klaster *K-Medoids*



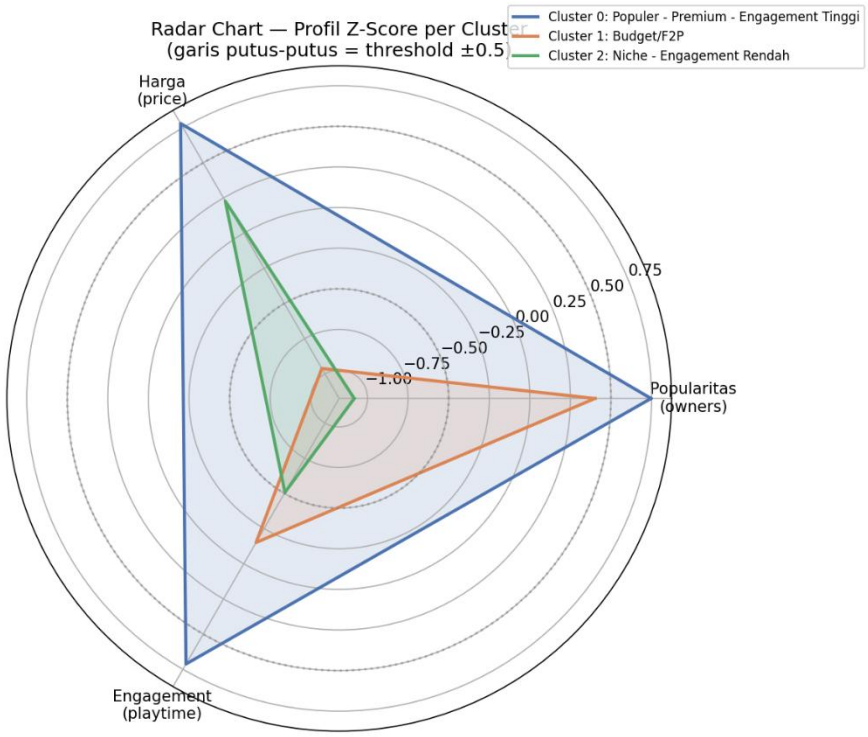
Dua komponen utama pada Gambar 16 menjelaskan 82,45% variansi data. Ketiga klaster menempati wilayah yang relatif berbeda, tetapi masih terdapat tumpang tindih pada area batas; kondisi ini wajar karena visualisasi hanya mempertahankan dua dari tiga dimensi fitur. Gambar 17 menunjukkan bahwa sebagian besar observasi memiliki nilai *silhouette* positif, sementara sejumlah titik pada Klaster 1 dan 2 berada dekat atau di bawah nol. Pola tersebut mengindikasikan bahwa pemisahan klaster tersedia secara umum, tetapi batas antarkelompok tidak sepenuhnya terpisah, konsisten dengan *Silhouette Score* keseluruhan sebesar 0,2922.

Perbedaan karakteristik fitur pada skala asli ditampilkan melalui Gambar 18, sedangkan profil relatif berbasis z-score disajikan pada Gambar 19. Proporsi jumlah anggota setiap klaster ditunjukkan pada Gambar 20. Ketiga visualisasi digunakan secara komplementer untuk membaca sebaran nilai, posisi relatif terhadap populasi, dan keseimbangan ukuran klaster.

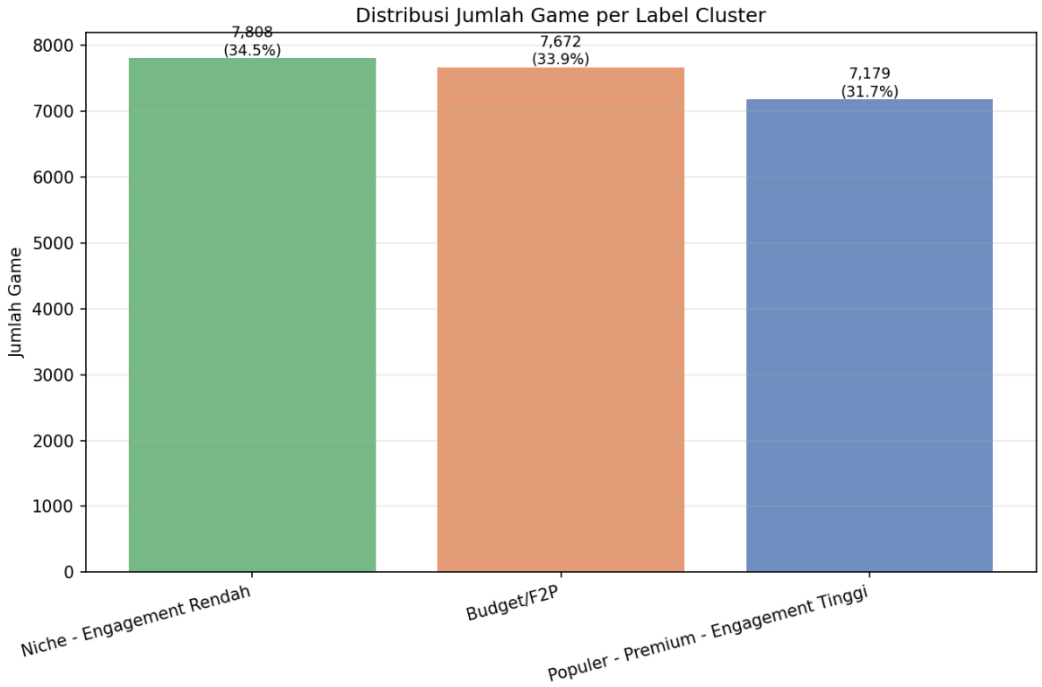
Gambar 18. Distribusi nilai estimasi kepemilikan, harga, dan waktu bermain pada setiap klaster.



Gambar 19. Profil relatif setiap klaster berdasarkan *z-score* rata-rata fitur



Gambar 20. Proporsi anggota tiga klaster K-Medoids



Dengan ambang operasional $|z| \geq 0{,}5$, Klaster 0 memiliki tingkat estimasi kepemilikan, harga, dan waktu bermain yang relatif tinggi dibandingkan populasi sehingga diberi label "*Populer-Premium-Engagement* Tinggi".

Klaster ini memiliki rata-rata 760.647 pemilik, harga US$9,92, dan waktu bermain 2.238 menit. Klaster 1 memiliki harga rata-rata paling rendah, yaitu US$1,17, dengan estimasi kepemilikan 288.355 dan waktu bermain 369 menit; pola tersebut mendasari label "*Budget*/F2P". Klaster 2 memiliki estimasi kepemilikan paling rendah, yakni 13.734, serta waktu bermain rata-rata 230 menit, sementara harga rata-ratanya US$6,15. Kombinasi tersebut mendasari label "*Niche-Engagement* Rendah". Label yang digunakan bersifat deskriptif terhadap profil numerik pada dataset penelitian dan tidak dimaksudkan sebagai klasifikasi resmi pada *platform Steam*.

Pemeriksaan sampel game digunakan untuk menilai kesesuaian label dengan observasi aktual. Pada Klaster 0 terdapat, antara lain, *Transistor*, *Starpoint Gemini Warlords*, dan *Spear Song*. Klaster 1 mencakup contoh berharga rendah seperti *Zombie Wars*: *Invasion* (US$0,80) dan *Spellbind*: *Luppe's Tale* (US$0,49). Klaster 2 memuat *Kingdom Elemental* dan *Art of Murder*: *The Secret Files*, yang pada sampel penelitian memiliki waktu bermain rendah. Contoh-contoh tersebut tidak digunakan sebagai dasar statistik pembentukan klaster, melainkan sebagai pemeriksaan interpretatif setelah model terbentuk. Dengan demikian, validasi kuantitatif tetap bertumpu pada metrik klasterisasi dan uji *Kruskal-Wallis*, sedangkan sampel aktual membantu menjelaskan makna profil kepada pembaca.

IV. SIMPULAN

Penelitian ini mengelompokkan 22.659 *game Steam* menggunakan estimasi jumlah pemilik, harga, dan rata-rata waktu bermain setelah melalui empat tahap penyaringan dari 136.080 entri mentah. Ketiga fitur ditransformasi menggunakan *Yeo-Johnson* dan distandarkan untuk mengurangi kemencengan serta menyetarakan skala. Evaluasi K=2 hingga K=10 menghasilkan *Silhouette Score* tertinggi pada K=3 sebesar 0,2913, sehingga tiga klaster digunakan pada perbandingan *K-Means* dan *K-Medoids*.

Pada K=3, *K-Means* menghasilkan *Silhouette Score* 0,2913 dan DBI 1,1234, sedangkan *K-Medoids (FasterPAM)* menghasilkan *Silhouette Score* 0,2922 dan DBI 1,1493. Perbedaan kedua metrik relatif kecil dan memberikan keunggulan pada model yang berbeda. *K-Medoids* dipilih sebagai model utama karena memperoleh Silhouette Score sedikit lebih tinggi serta distribusi anggota yang lebih merata pada dataset penelitian. Tiga klaster hasil model tersebut memiliki proporsi 31,68%, 33,86%, dan 34,46%. Uji *Kruskal-Wallis* menunjukkan perbedaan distribusi yang signifikan pada seluruh fitur ($p<0,001$), sehingga profil klaster dapat diinterpretasikan sebagai "Populer-*Premium*-Engagement Tinggi", "*Budget*/F2P", dan "*Niche-Engagement* Rendah".

Segmentasi yang dihasilkan memberikan gambaran kuantitatif mengenai variasi jangkauan pasar, harga, dan keterlibatan pemain pada populasi yang dianalisis. Klaster Populer-*Premium-Engagement* Tinggi merepresentasikan kelompok dengan rata-rata kepemilikan dan waktu bermain yang lebih tinggi, Klaster *Budget*/F2P dicirikan oleh harga rata-rata yang sangat rendah, sedangkan Klaster *Niche-Engagement* Rendah memiliki jangkauan dan waktu bermain yang lebih rendah. Hasil tersebut dapat digunakan sebagai dasar eksploratif untuk analisis pemosisian produk, strategi harga, atau pemetaan portofolio *game*. Pemanfaatannya perlu mempertimbangkan bahwa penelitian menggunakan tiga fitur, satu sumber dataset, dan kriteria penyaringan tertentu, sehingga generalisasi terhadap seluruh pasar *Steam* memerlukan pengujian lanjutan pada data dan periode yang berbeda.

UCAPAN TERIMA KASIH

Penulis menyampaikan terima kasih kepada dosen pembimbing dan Program Studi Informatika, Universitas Muhammadiyah Sidoarjo, atas arahan serta dukungan akademik selama pelaksanaan penelitian. Penulis juga mengapresiasi FronkonGames sebagai penyedia Steam Games Dataset yang digunakan dalam proses analisis. Secara pribadi juga penulis memberikan kredit khusus yaitu terima kasih dan apresiasi setinggi tingginya kepada keluarga serta sahabat yang tercinta. Karena telah mendukung perjalanan dan petualangan mengarungi luasnya samudera kehidupan, hingga akhirnya dapat menyelesaikan studi di perguruan tinggi. Salam hormat untuk semua makhluk hidup yang bergembira.

REFERENSI

[1] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data," *Information Sciences*, vol. 622, pp. 178–210, 2023, doi: 10.1016/j.ins.2022.11.139.

[2] A. E. Ezugwu, A. M. Ikotun, O. O. Oyelade, L. Abualigah, J. O. Agushaka, C. I. Eke, and A. A. Akinyelu, "A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy,

challenges, and future research prospects," *Engineering Applications of Artificial Intelligence*, vol. 110, Art. no. 104743, 2022, doi: 10.1016/j.engappai.2022.104743.

[3] J. Sreevalsan-Nair, "K-Medoids," in *Encyclopedia of Mathematical Geosciences*. Cham, Switzerland: Springer, 2023, pp. 697–700, doi: 10.1007/978-3-030-85040-1_172.

[4] E. Schubert and P. J. Rousseeuw, "Fast and eager k-medoids clustering: O(k) runtime improvement of the PAM, CLARA, and CLARANS algorithms," *Information Systems*, vol. 101, Art. no. 101804, 2021, doi: 10.1016/j.is.2021.101804.

[5] T. Hardiani and E. P. Silmina, "Comparative Analysis of K-Means and K-Medoids Algorithms in New Student Admission," *International Journal of Informatics and Computation*, vol. 6, no. 2, pp. 113–123, 2024, doi: 10.35842/ijicom.v6i2.91.

[6] H. A. Ulvi and M. Ikhsan, "Comparison of K-Means and K-Medoids Clustering Algorithms for Export and Import Grouping of Goods in Indonesia," *Sinkron*, vol. 8, no. 3, pp. 1671–1685, 2024, doi: 10.33395/sinkron.v8i3.13815.

[7] S. Ariska, D. Puspita, and I. Anggraini, "Comparison of K-Means and K-Medoids Algorithm for Clustering Data UMKM in Pagar Alam City," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 13, no. 2, pp. 193–199, 2024, doi: 10.32736/sisfokom.v13i2.2090.

[8] F. Batool and C. Hennig, "Clustering with the average silhouette width," *Computational Statistics & Data Analysis*, vol. 158, Art. no. 107190, 2021, doi: 10.1016/j.csda.2021.107190.

[9] L. Lenssen and E. Schubert, "Medoid silhouette clustering with automatic cluster number selection," *Information Systems*, vol. 120, Art. no. 102290, 2024, doi: 10.1016/j.is.2023.102290.

[10] F. Ros, R. Riad, and S. Guillaume, "PDBI: A partitioning Davies-Bouldin index for clustering evaluation," *Neurocomputing*, vol. 528, pp. 178–199, 2023, doi: 10.1016/j.neucom.2023.01.043.

[11] E. Schubert, "Stop using the elbow criterion for k-means and how to choose the number of clusters instead," *ACM SIGKDD Explorations Newsletter*, vol. 25, no. 1, pp. 36–42, 2023, doi: 10.1145/3606274.3606278.

[12] N. A. Maori and E. Evanita, "Metode Elbow dalam Optimasi Jumlah Cluster pada K-Means Clustering," *Simetris: Jurnal Teknik Mesin, Elektro dan Ilmu Komputer*, vol. 14, no. 2, pp. 277–288, 2023, doi: 10.24176/simet.v14i2.9630.

[13] M. Riani, A. C. Atkinson, and A. Corbellini, "Automatic robust Box–Cox and extended Yeo–Johnson transformations in regression," *Statistical Methods & Applications*, vol. 32, pp. 75–102, 2023, doi: 10.1007/s10260-022-00640-7.

[14] S. Tu, C. Li, and B. E. Shepherd, "Between- and within-cluster Spearman rank correlations," *Statistics in Medicine*, vol. 44, no. 3–4, Art. no. e10326, 2025, doi: 10.1002/sim.10326.

[15] J. S. C. Clark, P. Kulig, K. Podsiadło, K. Rydzewska, K. Arabski, M. Białecka, K. Safranow, and A. Ciechanowicz, "Empirical investigations into Kruskal-Wallis power studies utilizing Bernstein fits, simulations and medical study datasets," *Scientific Reports*, vol. 13, Art. no. 2352, 2023, doi: 10.1038/s41598-023-29308-2.

[16] A. De Luisa, J. Hartman, D. Nabergoj, S. Pahor, M. Rus, B. Stevanoski, J. Demšar, and E. Štrumbelj, "Predicting the popularity of games on Steam," *Elektrotehniški vestnik*, vol. 88, no. 4, pp. 151–162, 2021.

[17] A. Hidayat, B. Priyatna, F. Nurapriani, and T. Tukino, "Klasterisasi karakteristik game Steam menggunakan metode K-Means (studi kasus: rilisan game tahun 2024)," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 9, no. 5, pp. 8890–8894, 2025, doi: 10.36040/jati.v9i5.15244.

[18] S. F. F. Aulia, Y. A. Gerhana, and E. Nurlatifah, "Game and application purchasing patterns on Steam using K-Means algorithm," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 13, no. 3, pp. 303–310, 2024, doi: 10.32736/sisfokom.v13i3.2214.

[19] N. Grelier and S. Kaufmann, "Automated clustering of video games into groups with distinctive names," in *Entertainment Computing – ICEC 2024*, Lecture Notes in Computer Science, vol. 15192, pp. 223–231, Springer, 2024, doi: 10.1007/978-3-031-74353-5_16.

[20] J. Bartolomé, I. del Río, A. Martínez, A. Aranguren, I. Laña, and S. Alloza, "Game On: Exploring the Potential for Soft Skill Development Through Video Games," *Information*, vol. 16, no. 10, Art. no. 918, 2025, doi: 10.3390/info16100918.

Conflict of Interest Statement:
The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.