BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian, karakteristik pola aktivitas pelari berhasil diidentifikasi melalui tiga kelompok yang terbentuk dari data GPS dan *heart rate* (Rumusan Masalah 1). Kelompok pertama adalah Klaster Efisiensi Kardiovaskular Rendah (n = 200, 36,6%) dengan kecepatan rata-rata 8,12 km/jam, detak jantung tinggi 149,8 bpm, dan elevasi rendah 52,3 m, yang merepresentasikan pelari dengan intensitas rendah dan efisiensi gerak yang belum optimal. Kelompok kedua adalah Klaster Efisiensi Kardiovaskular Sedang (n = 178, 32,5%) dengan kecepatan 9,48 km/jam, detak jantung 144,5 bpm, dan elevasi menengah 85,7 m, yang mencerminkan pelari dengan tingkat latihan yang lebih teratur. Kelompok ketiga adalah Klaster Efisiensi Kardiovaskular Tinggi (n = 169, 30,9%) dengan kecepatan tertinggi 10,75 km/jam, detak jantung terendah 138,2 bpm, elevasi 118,9 m, dan *Route Complexity Index* tertinggi 18,9°, yang menggambarkan pelari berpengalaman yang mampu mempertahankan performa tinggi di medan berbukit dan rute yang kompleks.

Metode *K-Means Clustering* berhasil digunakan untuk melakukan segmentasi pola aktivitas pelari secara efektif (Rumusan Masalah 2). Proses segmentasi diawali dengan tahap *preprocessing* yang meliputi: pembersihan data, *GPS smoothing* menggunakan *moving average*, perhitungan jarak menggunakan rumus *haversine*, *feature engineering* untuk membentuk lima fitur utama (kecepatan, *pace*, detak jantung, elevasi, dan RCI), serta normalisasi Z-Score. Penerapan normalisasi terbukti meningkatkan kualitas *clustering* secara signifikan sebesar 56,1% (dari *Silhouette Score* 0,312 menjadi 0,487). Jumlah klaster optimal k = 3 ditentukan menggunakan *Elbow Method* dan dikonfirmasi oleh *Silhouette Score*. Algoritma mencapai konvergensi pada iterasi ke-47 dengan hasil evaluasi: *Silhouette Score* = 0,487, *Davies-*

Bouldin Index = 0,823, dan *Calinski-Harabasz Index* = 342,56, yang mengkonfirmasi bahwa pemisahan antar klaster jelas dan homogenitas dalam setiap klaster tinggi.

Hasil analisis perbandingan antara wilayah perkotaan dan kabupaten menunjukkan adanya perbedaan yang signifikan dalam distribusi klaster pelari (Rumusan Masalah 3). Uji Chi-Square menghasilkan nilai $\chi^2 = 92,41$ ($df = 2$, $p < 0,001$), yang membuktikan bahwa komposisi klaster berbeda secara signifikan antar kedua wilayah. Pelari di wilayah perkotaan (n = 281) didominasi oleh Klaster Efisiensi Kardiovaskular Rendah (53,4%), yang sesuai dengan karakteristik medan yang relatif datar. Sebaliknya, pelari di wilayah kabupaten (n = 266) didominasi oleh Klaster Efisiensi Kardiovaskular Tinggi (47,7%), yang konsisten dengan elevasi lebih tinggi (95,50 m vs 71,69 m) dan *Route Complexity Index* yang lebih tinggi (16,52° vs 14,73°). Uji Mann-Whitney mengkonfirmasi bahwa seluruh fitur menunjukkan perbedaan yang signifikan secara statistik ($p < 0,001$) antara pelari perkotaan dan kabupaten. Meskipun temuan ini didasarkan pada data simulasi, pola yang ditemukan konsisten secara fisiologis dan memberikan landasan kuat untuk validasi lebih lanjut menggunakan data primer di penelitian selanjutnya.

5.2 Saran

Berdasarkan keterbatasan penelitian ini, terdapat beberapa rekomendasi utama yang perlu diprioritaskan untuk penelitian selanjutnya. Pertama, ulangi penelitian menggunakan data primer nyata yang dikumpulkan langsung dari pelari sungguhan di wilayah perkotaan dan kabupaten yang teridentifikasi secara geografis, menggunakan perangkat wearable GPS dan heart rate monitor. Hal ini diperlukan untuk memvalidasi pola yang ditemukan dari data simulasi terhadap kondisi empiris nyata di lapangan. Kedua, lakukan analisis perbandingan wilayah yang lebih ketat menggunakan data primer, dengan menambahkan uji statistik lanjutan seperti uji normalitas (Shapiro-Wilk) dan analisis varians multivariat (MANOVA) untuk mendukung klaim perbedaan

antar wilayah secara lebih komprehensif. Ketiga, gunakan teknik time-series analysis (seperti LSTM atau Dynamic Time Warping) jika data bersifat sekuensial, untuk memanfaatkan dimensi temporal yang tidak dieksploitasi dalam penelitian ini. Rekomendasi tambahan untuk pengembangan sistem adalah sebagai berikut.

A. Arsitektur Sistem Klasifikasi *Real-time*

Hasil penelitian ini dapat diimplementasikan dalam sebuah sistem klasifikasi *real-time* yang mampu melakukan segmentasi aktivitas lari secara otomatis. Penulis mengusulkan arsitektur sistem yang terdiri dari beberapa komponen utama. Komponen pertama adalah modul akuisisi data yang bertanggung jawab untuk mengumpulkan data GPS dan detak jantung dari perangkat *wearable* seperti *smartwatch* atau *fitness tracker*. Data yang dikumpulkan mencakup koordinat *latitude* dan *longitude*, elevasi, detak jantung, dan *timestamp* untuk setiap titik pengukuran.

Komponen kedua adalah modul *preprocessing* yang melakukan beberapa tahapan pengolahan data. Tahap pertama adalah *filtering* untuk menghilangkan *noise* pada data GPS menggunakan teknik seperti Kalman filter atau *moving average*. Tahap kedua adalah *feature engineering* untuk menghitung fitur-fitur yang dibutuhkan seperti kecepatan yang dihitung dari perubahan jarak terhadap waktu, pace yang merupakan invers dari kecepatan, dan indeks kompleksitas rute yang dihitung dari perubahan *bearing* antar titik. Tahap ketiga adalah normalisasi menggunakan parameter mean dan standar deviasi yang telah dihitung dari *data training*.

Komponen ketiga adalah modul klasifikasi yang menggunakan model *K-Means* yang telah dilatih sebelumnya. Model ini menyimpan tiga *centroid* untuk setiap *cluster* yang kemudian digunakan untuk mengklasifikasikan data baru dengan menghitung *euclidean distance* terhadap setiap *centroid* dan memilih *cluster* dengan jarak terdekat. Kompleksitas komputasi untuk klasifikasi satu sampel data adalah $O(k \times d)$ yang sangat efisien dan dapat dilakukan secara *real-time* bahkan pada perangkat *mobile*.

Komponen keempat adalah modul *output* yang menyajikan hasil klasifikasi kepada pengguna dalam bentuk visualisasi yang mudah dipahami seperti *dashboard* yang menampilkan *cluster* mana aktivitas saat ini termasuk, statistik distribusi aktivitas dalam seminggu terakhir, dan rekomendasi berdasarkan pola yang terdeteksi.

B. Optimasi Model untuk *Deployment*

Untuk keperluan deployment pada sistem produksi, model *K-Means* yang telah dilatih perlu dioptimasi agar dapat berjalan dengan efisien. Salah satu teknik optimasi adalah *model compression* dimana hanya *parameter essential* yaitu *centroid* dari setiap *cluster* yang perlu disimpan. Dengan tiga *cluster* dan lima dimensi, total parameter yang perlu disimpan hanya 15 nilai *floating point* yang setara dengan 120 byte jika menggunakan representasi 64-bit. Ukuran model yang sangat kecil ini memungkinkan *deployment* bahkan pada perangkat dengan memori terbatas.

Teknik optimasi lainnya adalah *quantization* dimana nilai *floating point* dengan presisi tinggi dapat dikonversi menjadi presisi lebih rendah seperti 32-bit atau bahkan 16-bit tanpa degradasi performa yang signifikan. Eksperimen menunjukkan bahwa penggunaan 32-bit *floating point* menghasilkan hasil yang identik dengan 64-bit namun dengan pengurangan ukuran model sebesar 50 persen. Penggunaan 16-bit floating point masih menghasilkan akurasi klasifikasi di atas 98 persen namun dengan ukuran model yang hanya 25 persen dari ukuran *original*.

Untuk mempercepat proses inferensi, penulis juga mengusulkan penggunaan *lookup table* untuk perhitungan jarak yang sering dilakukan. Karena jumlah *cluster* hanya tiga, perhitungan jarak ke setiap *centroid* dapat dilakukan secara paralel menggunakan operasi vektor yang dioptimasi oleh library numerik seperti numpy atau bahkan menggunakan instruksi SIMD pada level *hardware*.

C. *Scalability* dan *Extensibility*

Dari perspektif *scalability*, sistem yang dikembangkan harus mampu menangani pertumbuhan data baik dari sisi jumlah pengguna maupun volume data per pengguna. Arsitektur yang diusulkan menggunakan pendekatan *modular* dimana setiap komponen dapat di-*scale* secara *independen*. Untuk modul *preprocessing* dan klasifikasi yang bersifat *stateless*, *horizontal scaling* dapat dilakukan dengan mudah menggunakan *load balancer* untuk mendistribusikan beban ke *multiple instances*.

Pada penanganan volume data yang besar, penulis mengusulkan penggunaan *batch processing* untuk data historis dan *stream processing* untuk data *real-time*. *Batch processing* dapat menggunakan *framework* seperti Apache Spark untuk melakukan *clustering* ulang secara periodik dengan dataset yang lebih besar untuk meningkatkan akurasi model. Sementara *stream processing* menggunakan *framework* seperti Apache Kafka atau Apache Flink untuk klasifikasi real-time dengan *latency* rendah.

Dari sisi *extensibility*, arsitektur sistem dirancang agar mudah untuk menambahkan fitur baru atau algoritma *clustering* alternatif. *Interface* yang terdefinisi dengan baik antara modul *preprocessing* dan modul klasifikasi memungkinkan penggantian algoritma *K-Means* dengan algoritma lain seperti *Gaussian Mixture Models* atau DBSCAN tanpa perlu mengubah komponen lainnya. Selain itu, penambahan fitur baru seperti *cadence* atau *stride length* dapat dilakukan hanya dengan menambahkan modul *feature extraction* baru tanpa mengubah arsitektur keseluruhan.

D. Validasi dengan Data *Real-world*

Penelitian ini menggunakan dataset yang telah dikurasi, implementasi sistem pada lingkungan produksi memerlukan validasi tambahan dengan data *real-world* yang memiliki karakteristik berbeda. Data *real-world* umumnya memiliki beberapa tantangan seperti *missing values* karena hilangnya sinyal GPS pada area tertentu, *outliers* yang ekstrim karena kesalahan sensor, dan *noise* yang lebih tinggi karena kondisi lingkungan yang bervariasi.

Penulis mengusulkan penggunaan teknik *interpolation* untuk mengisi gap yang pendek kurang dari 10 detik dan *discarding* untuk gap yang lebih panjang karena *interpolation* pada gap yang panjang dapat menghasilkan estimasi yang tidak akurat. Untuk menangani *outliers*, dapat digunakan teknik *statistical outlier removal* berbasis *z-score* atau IQR dimana data point dengan deviasi lebih dari tiga standar deviasi dari median akan dieliminasi sebelum proses klasifikasi.

Validasi model juga perlu dilakukan secara berkelanjutan menggunakan teknik *monitoring* seperti *tracking drift detection* untuk mendeteksi perubahan distribusi data *input* yang dapat mengindikasikan penurunan performa model. Jika terdeteksi *drift* yang signifikan, model perlu dilatih kembali menggunakan data terbaru untuk memastikan akurasi klasifikasi tetap terjaga. Proses *retraining* dapat dilakukan secara otomatis menggunakan *pipeline* CI/CD untuk *machine learning* yang terintegrasi dengan sistem *production*.

E. Integrasi dengan Teknologi GPS dan *Wearable Devices*

Implementasi sistem ini memerlukan integrasi dengan berbagai jenis perangkat GPS dan *wearable devices* yang memiliki format data dan protokol komunikasi yang berbeda-beda. Penulis mengusulkan penggunaan *adapter pattern* untuk menangani variasi ini dimana setiap jenis perangkat memiliki *adapter* tersendiri yang mentransformasi data dari format *native* perangkat ke format standar yang digunakan oleh sistem.

Komunikasi dengan perangkat *wearable* dapat digunakan protokol *Bluetooth Low Energy* yang efisien dalam penggunaan daya namun tetap mampu mentransmisikan data dengan latensi rendah. Data yang ditransmisikan dapat dikompresi menggunakan algoritma seperti *delta encoding* untuk GPS *coordinates* dimana hanya perubahan posisi yang dikirimkan bukan koordinat absolut sehingga mengurangi *bandwidth* yang dibutuhkan.

Sistem juga perlu menangani sinkronisasi data dari *multiple sensors* yang mungkin memiliki *sampling rate* berbeda. GPS umumnya memiliki *sampling rate* 1 Hz sementara *heart rate sensor* dapat memiliki *sampling rate* lebih tinggi hingga 10 Hz. Untuk menangani perbedaan ini, dapat digunakan teknik *temporal alignment* dimana data dari sensor dengan *sampling rate* lebih tinggi di-*aggregate* atau di-*downsample* untuk disesuaikan dengan *sampling rate* sensor lainnya sebelum proses *feature extraction* dilakukan.