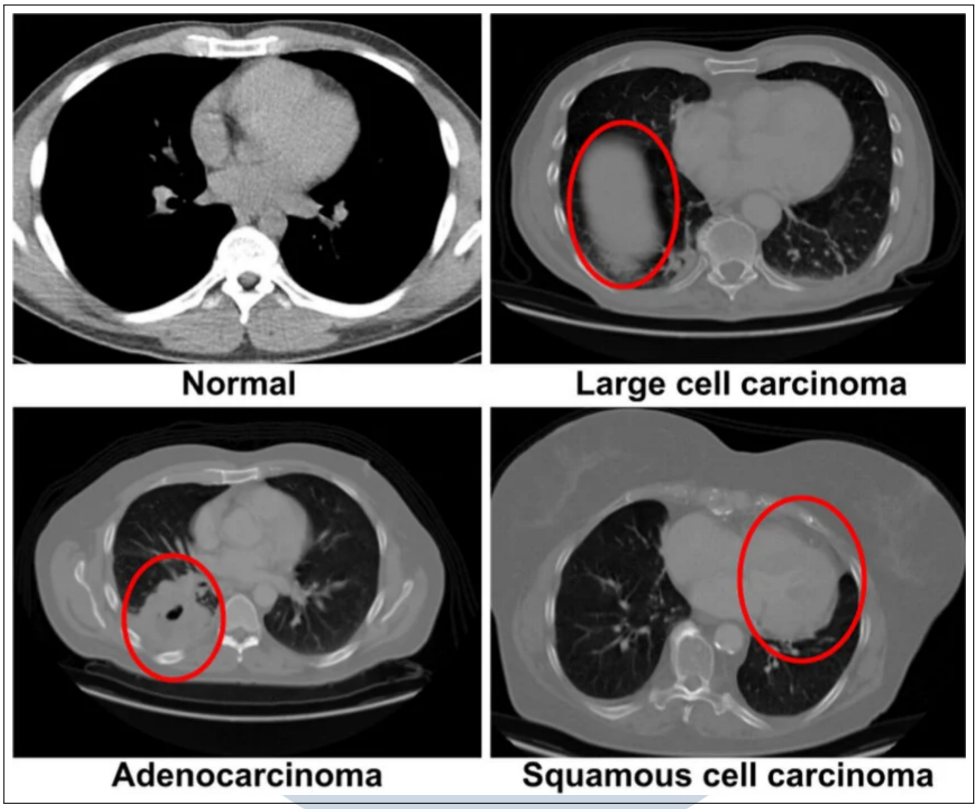
BAB 2
LANDASAN TEORI

2.1 Kanker Paru Sel Non-Kecil (*Non-Small Cell Lung Cancer*)

Etiologi NSCLC melibatkan interaksi kompleks antara paparan karsinogen dan kerentanan genetik. Meskipun merokok tetap menjadi faktor risiko utama melalui mekanisme kerusakan DNA oleh zat karsinogenik seperti PAH dan NNK, faktor lain seperti paparan asbes, gas radon, polusi udara, dan mutasi genetik juga berkontribusi signifikan [29]. Patofisiologi penyakit ini ditandai oleh disregulasi pertumbuhan sel dan penghindaran *apoptosis*, dengan profil molekuler yang sangat berbeda antara populasi Asia dan Barat. Sebuah studi terbaru mengungkapkan bahwa kanker paru pada populasi Asia didominasi oleh varian *Epidermal Growth Factor Receptor* (EGFR), terutama pada jenis *Non-Small Cell Lung Cancer* (NSCLC) [30].

Secara histopatologis, adenokarsinoma merupakan subtipe yang paling umum dan sering muncul pada perifer paru dengan berbagai pola pertumbuhan, termasuk varian lepidik. Karsinoma sel skuamosa (*Squamous Cell Carcinoma*/SCC) lebih sering ditemukan pada bagian sentral paru dan sangat berkaitan dengan riwayat merokok berat, yang ditandai oleh keratinisasi dan jembatan antarsel. Sementara itu, karsinoma sel besar merupakan diagnosis eksklusi untuk tumor yang tidak berdiferensiasi. Perbedaan karakteristik visual dan lokasi anatomi dari ketiga jenis kanker paru sel non-kecil (NSCLC) tersebut dibandingkan dengan kondisi paru normal dapat diamati pada citra CT yang disajikan pada Gambar 2.1.



Gambar 2.1. Visualisasi Citra CT Toraks pada Paru-Paru Normal dan Berbagai Jenis Kanker Paru Sel Non-Kecil

Sumber: [31]

Penegakan diagnosis NSCLC di Indonesia mengikuti alur yang terstruktur dengan melibatkan bronkoskopi sebagai prosedur utama untuk visualisasi dan biopsi, serta pencitraan radiologis standar emas berupa CT toraks dengan kontras untuk evaluasi anatomis dan *staging*. Biopsi jaringan, baik melalui bronkoskopi, TTNA, maupun torakoskopi, diperlukan tidak hanya untuk konfirmasi jenis sel, tetapi juga untuk analisis molekuler guna mendukung penerapan terapi presisi.

2.1.1 Sistem Klasifikasi TNM

Sistem klasifikasi Tumor, *Node*, dan Metastasis (TNM) merupakan standar internasional untuk mendeskripsikan perluasan anatomis penyakit kanker, khususnya pada *Non-Small Cell Lung Cancer* (NSCLC). Sistem ini dikelola secara kolaboratif oleh *American Joint Committee on Cancer* (AJCC) dan *Union for International Cancer Control* (UICC), serta berfungsi sebagai dasar dalam penentuan strategi pengobatan, estimasi prognosis, dan standarisasi riset onkologi global [32]. Evolusi sistem pementasan NSCLC dari edisi keenam

hingga kedelapan mencerminkan transisi dari model berbasis pengamatan geografis terbatas menuju pendekatan berbasis bukti statistik global yang lebih granular. Analisis basis data *International Association for the Study of Lung Cancer* (IASLC) menjadi pendorong utama revisi tersebut untuk memastikan bahwa setiap kategori memiliki nilai diskriminasi prognostik yang signifikan, sebagaimana dirangkum dalam Tabel 2.1.

Tabel 2.1. Evolusi Kategori TNM Kanker Paru Edisi ke-6, ke-7, dan ke-8

Sumber: [4]

Komponen	**Deskriptor / Ukuran**	**Edisi 6**	**Edisi 7**	**Edisi 8**
Kategori T (Ukuran)	≤ 1 cm	T1	T1a	T1a
	$> 1 - \leq 2$ cm	T1	T1a	T1b
	$> 2 - < 3$ cm	T1	T1b	T1c
	$> 3 - \leq 4$ cm	T2	T2a	T2a
	$> 4 - \leq 5$ cm	T2	T2a	T2b
	$> 5 - \leq 7$ cm	T2	T3	T3
	> 7 cm	T2	T4	T4
Kategori N (Nodal)	Hilar/peribronkhial ipsilateral	N1	N1	N1
	Mediastinal/subkarinal ipsilateral	N2	N2	N2
	Mediastinal kontralateral	N3	N3	N3
	Skalena/supraklavikular	N3	N3	N3
Kategori M (Metastasis)	Metastasis intratoraks	M1	M1a	M1a
	Ekstratoraks tunggal	M1	M1b	M1b
	Ekstratoraks multipel	M1	M1b	M1c
	Satu sistem organ	M1	M1b	M1b
	Banyak sistem organ	M1	M1b	M1c

Tabel tersebut merangkum perubahan metodologi pementasan dari basis data Amerika Utara (Edisi 6) menuju basis data internasional IASLC (Edisi 7 dan Edisi 8):

1. Kategori T (Tumor): Terjadi peningkatan granularitas yang signifikan. Ukuran tumor dipecah menjadi interval setiap 1 cm (dari T1a hingga T4) karena analisis statistik menunjukkan bahwa setiap peningkatan ukuran memberikan implikasi prognosis yang berbeda.

2. Kategori N (*Node*): Tidak terjadi perubahan pada deskriptor kategori. Meskipun analisis basis data IASLC telah mengevaluasi beberapa

subklasifikasi baru, seperti jumlah stasiun *node*, perubahan tersebut tidak diadopsi pada Edisi 7 maupun Edisi 8 guna menjaga stabilitas klasifikasi klinis. Lokasi anatomi kelenjar getah bening (N1, N2, dan N3) tetap menjadi standar emas yang konsisten.

3. Kategori M (Metastasis): Terjadi penyempurnaan klasifikasi berdasarkan lokasi dan jumlah metastasis. Edisi terbaru memisahkan metastasis intratoraks (M1a) dari metastasis ekstratoraks, serta membedakan metastasis tunggal (M1b) dan metastasis multipel (M1c) berdasarkan perbedaan hasil *survival* yang signifikan.

2.2 Segmentasi TotalSegmentator

TotalSegmentator tidak membangun arsitektur jaringan saraf dari awal, melainkan mengadopsi dan mengoptimalkan kerangka kerja nnU-Net (*no-new-Net*). Filosofi utama nnU-Net menekankan bahwa performa segmentasi citra medis tidak semata-mata ditentukan oleh inovasi arsitektur jaringan, melainkan lebih bergantung pada optimalisasi *pipeline* pembelajaran, seperti *preprocessing* data, konfigurasi pelatihan, serta strategi augmentasi. Pendekatan ini dibuktikan melalui kerangka *self-configuring* nnU-Net yang secara otomatis menyesuaikan seluruh *pipeline* terhadap karakteristik *dataset* sehingga mampu mencapai performa *state-of-the-art* tanpa memerlukan desain arsitektur yang kompleks maupun eksotis [33].

Keunggulan utama TotalSegmentator terletak pada penggunaan *dataset* pelatihan yang sangat beragam dan berskala besar. Versi pertama (V1) model ini dilatih menggunakan 1.204 pemeriksaan CT yang dipilih secara acak dari arsip klinis rutin di Universitas Hospital Basel, Swiss. *Dataset* tersebut merepresentasikan kondisi dunia nyata yang kompleks, mencakup pasien dengan rentang usia yang luas (16 hingga 96 tahun), berbagai jenis pemindai dari manufaktur yang berbeda, serta variasi penggunaan agen kontras dan parameter akuisisi citra, dimana struktur organ berdasarkan jumlah kelas dapat dilihat pada Tabel 2.2.

Tabel 2.2. Kategori Struktur Organ Berdasarkan Jumlah Kelas dan Contohnya
Sumber: [34]

Kategori Struktur (V1)	**Jumlah Kelas**	**Contoh Struktur**
Organ	27	Hati, Ginjal, Limpa, Paru-paru, Jantung, Pankreas
Tulang	59	Vertebra (C1-S1), Rusuk, Skapula, Klavikula, Femur
Otot	10	Iliopsoas, Gluteus, Otot Punggung Autochthon
Pembuluh Darah	8	Aorta, Vena Cava, Arteri Iliaka, Vena Portal

Evolusi menuju versi kedua (V2) menghasilkan peningkatan yang signifikan melalui penambahan jumlah kelas menjadi 117 struktur. Pembaruan ini tidak hanya meningkatkan jumlah organ yang dapat disegmentasi, tetapi juga memperkenalkan sub-tugas (*subtasks*) yang lebih spesifik untuk area yang secara klinis sangat menantang, seperti pembuluh darah paru, nodul paru, dan ruang jantung beresolusi tinggi. TotalSegmentator V2 juga meningkatkan ketangguhan model terhadap berbagai anomali anatomi, seperti organ yang hilang akibat tindakan pembedahan maupun organ yang mengalami pergeseran akibat massa patologis.

Salah satu pertimbangan teknis utama dalam segmentasi tiga dimensi adalah resolusi *voxel*. TotalSegmentator menyediakan dua varian utama yang dirancang untuk memenuhi kebutuhan yang berbeda antara akurasi dan kecepatan komputasi. Model standar beroperasi pada resolusi isotropik 1,5 mm yang memberikan keseimbangan optimal untuk sebagian besar struktur anatomi. Namun, untuk studi yang melibatkan ribuan subjek atau kondisi dengan keterbatasan sumber daya komputasi, tersedia mode *fast* yang beroperasi pada resolusi isotropik 3 mm.

Meskipun model 3 mm memiliki kebutuhan komputasi yang lebih rendah, model tersebut tetap menunjukkan kinerja yang sangat baik. Hasil evaluasi menunjukkan bahwa meskipun skor Dice mengalami sedikit penurunan, nilai *Normalized Surface Distance* (NSD) tetap tinggi karena sebagian besar deviasi segmentasi masih berada dalam ambang toleransi 3 mm. Tahap prapemrosesan yang dilakukan oleh TotalSegmentator mencakup berbagai langkah otomatis, seperti *resampling* citra ke resolusi target, normalisasi nilai *Hounsfield Unit* (HU), serta *cropping* area di luar tubuh untuk mengoptimalkan penggunaan memori *Graphics Processing Unit* (GPU), seperti yang terlihat dalam Tabel 2.3.

Tabel 2.3. Spesifikasi Teknis Perbandingan Model Totalsegmentator

Sumber: [34]

Spesifikasi Teknis	**Model 1.5 mm (Normal)**	**Model 3 mm (*Fast*)**
Kebutuhan Memori GPU	$\sim$12 GB	$\sim$6 GB
Waktu Inferensi (*Thorax*)	60 – 120 detik	10 – 20 detik
Fokus Penggunaan	Riset presisi, Klinis	Skrining massal, Pratinjau cepat

Dalam kondisi paru-paru yang sehat maupun pada kasus dengan patologi ringan, TotalSegmentator menunjukkan kinerja yang sangat stabil. Hasil pengujian pada berbagai *dataset* memperlihatkan skor *Dice* rata-rata yang sering kali melebihi 0,95. Nilai tersebut mencerminkan tingkat kesesuaian spasial yang sangat tinggi antara hasil segmentasi otomatis dan delineasi manual yang dilakukan oleh ahli. Namun, keberhasilan segmentasi paru-paru tidak hanya diukur menggunakan skor *Dice*. Metrik berbasis jarak, seperti *Hausdorff Distance* (HD) dan *Average Symmetric Surface Distance* (ASSD), memberikan informasi tambahan mengenai tingkat akurasi algoritma dalam menentukan batas fisura yang sangat tipis, dapat dilihat dalam Tabel 2.4.

Tabel 2.4. Metrik Evaluasi Segmentasi Struktur Paru-paru

Sumber: [35]

Struktur Paru-paru	**Skor *Dice* (DSC)**	***Hausdorff Distance* (HD95)**
Paru Kanan (Utuh)	0.96 – 0.98	$<$ 3.5 mm
Paru Kiri (Utuh)	0.96 – 0.98	$<$ 3.5 mm
Lobus Atas Kanan (RUL)	0.92 – 0.95	4.5 – 7.5 mm
Lobus Tengah Kanan (RML)	0.79 – 0.91	7.0 – 15.0 mm
Lobus Bawah Kanan (RLL)	0.93 – 0.96	3.5 – 6.5 mm

Analisis terhadap kesalahan segmentasi menunjukkan bahwa lobus tengah kanan (*Right Middle Lobe*/RML) secara konsisten menghasilkan performa yang lebih rendah dibandingkan lobus lainnya. Kondisi tersebut disebabkan oleh keberadaan fisura horizontal yang sering kali tidak lengkap secara anatomis atau sulit teridentifikasi pada citra CT dengan resolusi rendah maupun dosis radiasi rendah. Kegagalan dalam mendeteksi fisura tersebut dapat menyebabkan fenomena *segmentation leakage*, yaitu kondisi ketika model tidak mampu memisahkan batas antara lobus atas dan lobus tengah secara tepat sehingga menghasilkan *under-segmentation* pada satu bagian dan *over-segmentation* pada bagian lainnya [35].

2.3 Rekayasa Fitur

2.3.1 Ekstraksi Fitur Radiomik

Ekstraksi fitur radiomik merupakan tahapan krusial dalam mengubah citra medis dari sekadar representasi visual menjadi data digital yang dapat dikuantifikasi. Proses ini bertujuan untuk mengungkap fenotipe radiografis yang tidak dapat diamati secara langsung melalui pengamatan visual manusia dengan memanfaatkan algoritma komputasi yang terstandarisasi. Dalam penelitian ini, operasionalisasi perhitungan fitur radiomik didasarkan pada kerangka kerja PyRadiomics, yaitu platform *open-source* yang dirancang untuk memastikan reproduksibilitas dan transparansi dalam analisis radiomik [36].

Metode ekstraksi ini mencakup spektrum parameter yang luas, mulai dari distribusi intensitas sederhana hingga pola tekstur spasial yang kompleks. Secara sistematis, fitur-fitur tersebut dikelompokkan ke dalam beberapa kategori utama, yaitu statistik orde pertama (*first-order statistics*), fitur bentuk (*shape features*), serta matriks tekstur tingkat lanjut seperti *Gray Level Co-occurrence Matrix* (GLCM), *Gray Level Run Length Matrix* (GLRLM), *Gray Level Size Zone Matrix* (GLSZM), *Gray Level Dependence Matrix* (GLDM), dan *Neighboring Gray Tone Difference Matrix* (NGTDM). Selain diekstraksi dari citra asli, analisis juga diperluas menggunakan penerapan filter tingkat tinggi (*high-order filters*) dan transformasi citra untuk menyoroti karakteristik lesi yang tersembunyi, yang meliputi *Gradient Magnitude Filter*, *Laplacian of Gaussian* (LoG) *Filter*, serta *Wavelet Decomposition*. Rincian matematis dan logika komputasi dari masing-masing kelompok fitur dan transformasi tersebut dijabarkan sebagai berikut.

1. *First-Order Statistics*

   Fitur ini menggambarkan distribusi nilai intensitas *voxel* di dalam ROI tanpa mempertimbangkan hubungan spasial antar-*voxel*. Fitur-fitur tersebut dihitung berdasarkan histogram intensitas dan mencakup ukuran statistik, seperti *Mean*, *Variance*, *Skewness*, *Kurtosis*, *Entropy*, dan *Energy*. Parameter-parameter tersebut merepresentasikan karakteristik distribusi intensitas, seperti tingkat variasi, homogenitas, serta kompleksitas jaringan tumor. Rumus-rumus fitur orde pertama dijabarkan sebagai berikut.

(a) *Energy*

$$Energy = \sum_{i=1}^{N_p} (X(i) + c)^2 \tag{2.1}$$

Berdasarkan Rumus 2.1, fitur *Energy* digunakan untuk mengukur besarnya akumulasi kuadrat nilai intensitas pada seluruh *voxel* dalam ROI. Nilai *Energy* yang tinggi menunjukkan dominasi intensitas tertentu dengan magnitudo besar, sedangkan nilai yang rendah mengindikasikan distribusi intensitas yang lebih kecil atau tersebar. Pada rumus tersebut, $X(i)$ merepresentasikan nilai intensitas *voxel* ke-i, N_p adalah jumlah total *voxel* dalam ROI, dan c merupakan konstanta yang ditambahkan untuk menghindari permasalahan numerik pada data tertentu.

(b) *Total Energy*

$$Total\ Energy = V_{\text{voxel}} \sum_{i=1}^{N_p} (X(i) + c)^2 \tag{2.2}$$

Berdasarkan Rumus 2.2, fitur *Total Energy* merupakan pengembangan dari *Energy* dengan mempertimbangkan volume fisik setiap *voxel*. Fitur ini menggambarkan total energi intensitas dalam ROI secara keseluruhan. Semakin besar nilai *Total Energy*, semakin besar kontribusi intensitas dan volume jaringan yang dianalisis. Pada rumus tersebut, V_{voxel} menyatakan volume setiap *voxel*.

(c) *Entropy*

$$Entropy = -\sum_{i=1}^{N_g} p(i) \log_2(p(i) + \varepsilon) \tag{2.3}$$

Berdasarkan Rumus 2.3, fitur *Entropy* digunakan untuk mengukur tingkat ketidakteraturan atau kompleksitas distribusi intensitas pada ROI. Nilai *Entropy* yang tinggi menunjukkan distribusi intensitas yang lebih heterogen dan kompleks, sedangkan nilai yang rendah menunjukkan distribusi yang lebih homogen. Pada rumus tersebut, $p(i)$ adalah probabilitas kemunculan tingkat intensitas ke-i, N_g adalah jumlah tingkat intensitas yang tersedia, dan ε merupakan konstanta kecil untuk menghindari nilai logaritma nol.

(d) *Minimum*

$$Minimum = \min(X) \tag{2.4}$$

Berdasarkan Rumus 2.4, fitur *Minimum* merepresentasikan nilai

intensitas terkecil yang ditemukan pada ROI. Fitur ini digunakan untuk mengidentifikasi batas bawah distribusi intensitas dan dapat memberikan informasi mengenai area dengan densitas paling rendah dalam citra.

(e) *10th Percentile*

$$10th\ Percentile = P_{10} \tag{2.5}$$

Berdasarkan Rumus 2.5, fitur *10th Percentile* menunjukkan nilai intensitas yang membatasi 10% data terendah dalam ROI. Dengan kata lain, sebanyak 10% *voxel* memiliki nilai intensitas lebih kecil atau sama dengan nilai tersebut. Fitur ini digunakan untuk menggambarkan karakteristik distribusi pada bagian bawah histogram intensitas.

(f) *90th Percentile*

$$90th\ Percentile = P_{90} \tag{2.6}$$

Berdasarkan Rumus 2.6, fitur *90th Percentile* menunjukkan nilai intensitas yang membatasi 90% data dalam ROI. Sebanyak 90% *voxel* memiliki nilai intensitas lebih kecil atau sama dengan nilai tersebut. Fitur ini digunakan untuk merepresentasikan karakteristik distribusi pada bagian atas histogram intensitas.

(g) *Maximum*

$$Maximum = \max(X) \tag{2.7}$$

Berdasarkan Rumus 2.7, fitur *Maximum* merepresentasikan nilai intensitas terbesar yang ditemukan dalam ROI. Nilai ini menggambarkan titik intensitas tertinggi yang dapat berkaitan dengan area jaringan dengan densitas atau karakteristik pencitraan tertentu.

(h) *Mean*

$$Mean = \frac{1}{N_p} \sum_{i=1}^{N_p} X(i) \tag{2.8}$$

Berdasarkan Rumus 2.8, fitur *Mean* digunakan untuk menghitung rata-rata nilai intensitas seluruh *voxel* dalam ROI. Fitur ini memberikan gambaran umum mengenai tingkat intensitas citra secara keseluruhan. Pada rumus tersebut, $\bar{X}$ menyatakan nilai rata-rata intensitas dan N_p merupakan jumlah total *voxel*.

(i) *Median*

$$Median = P_{50} \tag{2.9}$$

Berdasarkan Rumus 2.9, fitur *Median* menunjukkan nilai intensitas tengah dari distribusi data yang telah diurutkan. Sebanyak 50% data berada di bawah nilai ini dan 50% lainnya berada di atasnya. Fitur ini relatif lebih *robust* terhadap keberadaan pencilan dibandingkan nilai rata-rata.

(j) *Interquartile Range*

$$Interquartile\ Range = P_{75} - P_{25} \tag{2.10}$$

Berdasarkan Rumus 2.10, fitur *Interquartile Range* (IQR) mengukur rentang sebaran data pada 50% bagian tengah distribusi intensitas. Nilai IQR yang besar menunjukkan variasi intensitas yang tinggi, sedangkan nilai yang kecil menunjukkan distribusi yang lebih terkonsentrasi.

(k) *Range*

$$Range = \max(X) - \min(X) \tag{2.11}$$

Berdasarkan Rumus 2.11, fitur *Range* dihitung sebagai selisih antara nilai intensitas maksimum dan minimum dalam ROI. Fitur ini digunakan untuk menggambarkan rentang keseluruhan variasi intensitas yang terdapat pada data.

(l) *Mean Absolute Deviation* (MAD)

$$MAD = \frac{1}{N_p} \sum_{i=1}^{N_p} |X(i) - \bar{X}| \tag{2.12}$$

Berdasarkan Rumus 2.12, fitur *Mean Absolute Deviation* (MAD) mengukur rata-rata deviasi absolut setiap nilai intensitas terhadap nilai rata-ratanya. Nilai MAD yang tinggi menunjukkan penyebaran intensitas yang lebih besar, sedangkan nilai yang rendah menunjukkan distribusi yang lebih homogen.

(m) *Robust Mean Absolute Deviation* (rMAD)

$$rMAD = \frac{1}{N_{10-90}} \sum_{i=1}^{N_{10-90}} |X_{10-90}(i) - \bar{X}_{10-90}| \tag{2.13}$$

Berdasarkan Rumus 2.13, fitur *Robust Mean Absolute Deviation* (rMAD) mengukur rata-rata deviasi absolut hanya pada data yang berada dalam rentang persentil ke-10 hingga ke-90. Pendekatan ini mengurangi pengaruh pencilan sehingga menghasilkan ukuran variasi yang lebih stabil dan *robust*.

(n) *Root Mean Squared* (RMS)

$$RMS = \sqrt{\frac{1}{N_p}\sum_{i=1}^{N_p}(X(i)+c)^2} \tag{2.14}$$

Berdasarkan Rumus 2.14, fitur *Root Mean Squared* (RMS) digunakan untuk mengukur besar rata-rata kuadrat intensitas pada ROI. Karena melibatkan operasi kuadrat, fitur ini memberikan bobot yang lebih besar terhadap nilai intensitas tinggi dibandingkan rata-rata biasa.

(o) *Standard Deviation*

$$Standard\ Deviation = \sqrt{\frac{1}{N_p}\sum_{i=1}^{N_p}(X(i)-\bar{X})^2} \tag{2.15}$$

Berdasarkan Rumus 2.15, fitur *Standard Deviation* mengukur tingkat penyebaran nilai intensitas terhadap nilai rata-ratanya. Nilai yang tinggi menunjukkan heterogenitas intensitas yang besar, sedangkan nilai yang rendah menunjukkan distribusi yang lebih seragam.

(p) *Skewness*

$$Skewness = \frac{\frac{1}{N_p}\sum_{i=1}^{N_p}(X(i)-\bar{X})^3}{\left(\sqrt{\frac{1}{N_p}\sum_{i=1}^{N_p}(X(i)-\bar{X})^2}\right)^3} \tag{2.16}$$

Berdasarkan Rumus 2.16, fitur *Skewness* digunakan untuk mengukur tingkat kemencengan distribusi intensitas. Nilai positif menunjukkan distribusi yang menceng ke kanan, sedangkan nilai negatif menunjukkan distribusi yang menceng ke kiri. Nilai yang mendekati nol mengindikasikan distribusi yang relatif simetris.

(q) *Kurtosis*

$$Kurtosis = \frac{\frac{1}{N_p}\sum_{i=1}^{N_p}(X(i)-\bar{X})^4}{\left(\frac{1}{N_p}\sum_{i=1}^{N_p}(X(i)-\bar{X})^2\right)^2} \tag{2.17}$$

Berdasarkan Rumus 2.17, fitur *Kurtosis* digunakan untuk mengukur tingkat keruncingan distribusi intensitas dibandingkan distribusi normal. Nilai *Kurtosis* yang tinggi menunjukkan distribusi dengan puncak yang lebih tajam dan ekor yang lebih berat, sedangkan nilai yang rendah menunjukkan distribusi yang lebih datar.

(r) *Variance*

$$Variance = \frac{1}{N_p}\sum_{i=1}^{N_p}(X(i)-\bar{X})^2 \tag{2.18}$$

Berdasarkan Rumus 2.18, fitur *Variance* mengukur rata-rata kuadrat deviasi nilai intensitas terhadap rata-ratanya. Nilai *Variance* yang besar menunjukkan variasi intensitas yang tinggi dalam ROI, sedangkan nilai yang kecil menunjukkan distribusi yang lebih homogen.

(s) *Uniformity*

$$Uniformity = \sum_{i=1}^{N_g} p(i)^2 \tag{2.19}$$

Berdasarkan Rumus 2.19, fitur *Uniformity* digunakan untuk mengukur tingkat keseragaman distribusi intensitas pada ROI. Nilai *Uniformity* yang tinggi menunjukkan bahwa sebagian besar *voxel* memiliki tingkat intensitas yang serupa, sedangkan nilai yang rendah menunjukkan distribusi yang lebih heterogen.

Pada keseluruhan fitur orde pertama, $X(i)$ menyatakan nilai intensitas *voxel* ke-i, N_p adalah jumlah total *voxel* dalam ROI, N_g merupakan jumlah tingkat intensitas hasil diskretisasi, dan $p(i)$ adalah probabilitas kemunculan tingkat intensitas ke-i. Fitur-fitur tersebut digunakan untuk merepresentasikan karakteristik statistik distribusi intensitas citra yang menjadi dasar dalam analisis radiomik dan proses klasifikasi.

2. *Shape Features*

Fitur bentuk merepresentasikan karakteristik geometris dari *Region of Interest* (ROI) tumor, baik dalam dua dimensi (2D) maupun tiga dimensi (3D). Fitur 3D menggambarkan properti volumetrik, seperti *Volume*, *Surface Area*, *Compactness*, dan *Sphericity*, sedangkan fitur 2D menangkap karakteristik bentuk pada irisan citra, seperti *Perimeter*, *Area*, dan *Circularity*. Selain itu, fitur seperti *Elongation* dan *Axis Length* digunakan untuk menggambarkan bentuk serta orientasi tumor. Informasi tersebut berkaitan dengan ukuran dan

morfologi tumor yang sering diasosiasikan dengan tingkat invasi dan stadium penyakit. Rumus-rumus fitur bentuk 3D dijabarkan sebagai berikut.

(a) *Mesh Volume*

$$V = \sum_{i=1}^{N_f} \frac{Oa_i \cdot (Ob_i \times Oc_i)}{6} \tag{2.20}$$

Berdasarkan Rumus 2.20, fitur *Mesh Volume* digunakan untuk menghitung volume tiga dimensi ROI berdasarkan representasi permukaan berbentuk *mesh* yang tersusun dari sejumlah bidang segitiga. Perhitungan dilakukan dengan menjumlahkan volume tetrahedron yang dibentuk oleh setiap segitiga permukaan terhadap titik asal koordinat. Nilai *Mesh Volume* merepresentasikan ukuran aktual massa tumor dalam ruang tiga dimensi. Pada rumus tersebut, N_f menyatakan jumlah total bidang segitiga pada permukaan *mesh*, sedangkan a_i, b_i, dan c_i merupakan titik sudut dari segitiga ke-i.

(b) *Voxel Volume*

$$V_{voxel} = \sum_{k=1}^{N_v} V_k \tag{2.21}$$

Berdasarkan Rumus 2.21, fitur *Voxel Volume* digunakan untuk menghitung volume ROI berdasarkan jumlah seluruh *voxel* yang termasuk dalam area segmentasi. Volume diperoleh dari penjumlahan volume masing-masing *voxel* penyusun ROI. Nilai yang lebih besar menunjukkan ukuran tumor yang lebih besar. Pada rumus tersebut, N_v merupakan jumlah total *voxel* dalam ROI dan V_k menyatakan volume *voxel* ke-k.

(c) *Surface Area*

$$A = \sum_{i=1}^{N_f} \frac{1}{2} |a_i b_i \times a_i c_i| \tag{2.22}$$

Berdasarkan Rumus 2.22, fitur *Surface Area* digunakan untuk menghitung luas permukaan tiga dimensi ROI. Nilai ini diperoleh dari penjumlahan luas seluruh segitiga yang membentuk permukaan objek. Semakin besar nilai *Surface Area*, semakin luas permukaan tumor yang diamati. Fitur ini dapat digunakan untuk menggambarkan kompleksitas bentuk dan batas permukaan tumor.

(d) *Surface Area to Volume Ratio*

$$Surface\ Area\ to\ Volume\ Ratio = \frac{A}{V} \tag{2.23}$$

Berdasarkan Rumus 2.23, fitur *Surface Area to Volume Ratio* digunakan untuk mengukur perbandingan antara luas permukaan dan volume ROI. Nilai yang tinggi menunjukkan bahwa objek memiliki permukaan yang relatif besar dibandingkan volumenya, yang umumnya mengindikasikan bentuk yang lebih tidak beraturan atau kompleks. Sebaliknya, nilai yang rendah menunjukkan bentuk yang lebih kompak.

(e) *Sphericity*

$$Sphericity = \frac{\sqrt[3]{36\pi V^2}}{A} \tag{2.24}$$

Berdasarkan Rumus 2.24, fitur *Sphericity* digunakan untuk mengukur tingkat kemiripan bentuk ROI terhadap bentuk bola sempurna. Nilai *Sphericity* berada pada rentang 0 hingga 1, di mana nilai yang semakin mendekati 1 menunjukkan bahwa bentuk tumor semakin menyerupai bola. Sebaliknya, nilai yang rendah mengindikasikan bentuk yang lebih tidak beraturan.

(f) *Compactness 1*

$$Compactness\ 1 = \frac{V}{\sqrt{\pi}A^3} \tag{2.25}$$

Berdasarkan Rumus 2.25, fitur *Compactness 1* digunakan untuk mengukur tingkat kekompakan bentuk ROI dengan mempertimbangkan hubungan antara volume dan luas permukaan. Nilai yang lebih tinggi menunjukkan objek yang lebih padat dan mendekati bentuk geometris yang efisien, sedangkan nilai yang lebih rendah mengindikasikan bentuk yang lebih kompleks atau tidak beraturan.

(g) *Compactness 2*

$$Compactness\ 2 = \frac{36\pi V^2}{A^3} \tag{2.26}$$

Berdasarkan Rumus 2.26, fitur *Compactness 2* digunakan untuk mengevaluasi tingkat kekompakan ROI berdasarkan hubungan antara volume kuadrat dan luas permukaan kubik. Nilai maksimum dicapai ketika objek berbentuk bola sempurna. Oleh karena itu, semakin tinggi nilai *Compactness 2*, semakin kompak dan semakin mendekati bentuk

bola ROI tersebut.

(h) *Spherical Disproportion*

$$Spherical\ Disproportion = \frac{A}{4\pi R^2} = \frac{A}{\sqrt[3]{36\pi V^2}} \tag{2.27}$$

Berdasarkan Rumus 2.27, fitur *Spherical Disproportion* digunakan untuk mengukur tingkat penyimpangan bentuk ROI terhadap bentuk bola sempurna. Fitur ini dihitung dengan membandingkan luas permukaan aktual ROI dengan luas permukaan bola yang memiliki volume yang sama. Nilai yang mendekati 1 menunjukkan bentuk yang menyerupai bola, sedangkan nilai yang lebih besar dari 1 menunjukkan bentuk yang semakin tidak beraturan.

(i) *Maximum 3D Diameter*

$$Maximum\ 3D\ Diameter = \max(||x_i - x_j||) \tag{2.28}$$

Berdasarkan Rumus 2.28, fitur *Maximum 3D Diameter* digunakan untuk mengukur jarak *Euclidean* maksimum antara dua titik pada permukaan ROI dalam ruang tiga dimensi. Fitur ini merepresentasikan dimensi terbesar tumor yang dapat diamati dari seluruh arah pengamatan dan sering digunakan sebagai indikator ukuran maksimum lesi.

(j) *Maximum 2D Diameter (Slice)*

$$Maximum\ 2D\ Diameter\ (Slice) = \max(||x_i - x_j||)_{axial} \tag{2.29}$$

Berdasarkan Rumus 2.29, fitur *Maximum 2D Diameter (Slice)* digunakan untuk menghitung jarak maksimum antar dua titik ROI pada bidang *axial*. Nilai ini menggambarkan dimensi terbesar tumor yang terlihat pada irisan melintang citra CT-Scan.

(k) *Maximum 2D Diameter (Column)*

$$Maximum\ 2D\ Diameter\ (Column) = \max(||x_i - x_j||)_{coronal} \tag{2.30}$$

Berdasarkan Rumus 2.30, fitur *Maximum 2D Diameter (Column)* digunakan untuk menghitung jarak maksimum antar dua titik ROI pada

bidang *coronal*. Fitur ini menggambarkan dimensi terbesar tumor yang diamati dari tampilan depan-belakang tubuh pasien.

(l) *Maximum 2D Diameter (Row)*

$$Maximum\ 2D\ Diameter\ (Row) = \max(||x_i - x_j||)_{sagittal} \tag{2.31}$$

Berdasarkan Rumus 2.31, fitur *Maximum 2D Diameter (Row)* digunakan untuk menghitung jarak maksimum antar dua titik ROI pada bidang *sagittal*. Nilai fitur ini merepresentasikan ukuran terbesar tumor yang terlihat dari tampilan samping tubuh pasien.

(m) *Major Axis Length*

$$Major\ Axis\ Length = 4\sqrt{\lambda_{major}} \tag{2.32}$$

Berdasarkan Rumus 2.32, fitur *Major Axis Length* digunakan untuk mengukur panjang sumbu utama ROI yang diperoleh dari analisis matriks kovarians . Pada ROI tiga dimensi, fitur ini merepresentasikan dimensi terbesar dari elipsoid ekuivalen yang merepresentasikan bentuk tumor. Sementara pada ROI dua dimensi, fitur ini merepresentasikan sumbu utama dari elips ekuivalen yang dibentuk oleh area segmentasi.

(n) *Minor Axis Length*

$$Minor\ Axis\ Length = 4\sqrt{\lambda_{minor}} \tag{2.33}$$

Berdasarkan Rumus 2.33, fitur *Minor Axis Length* digunakan untuk mengukur panjang sumbu kedua dari elipsoid ekuivalen yang merepresentasikan ROI. Fitur ini menggambarkan dimensi menengah dari bentuk tumor. Pada rumus tersebut, λ_{minor} merupakan nilai *eigen* kedua terbesar dari matriks kovarians.

(o) *Least Axis Length*

$$Least\ Axis\ Length = 4\sqrt{\lambda_{least}} \tag{2.34}$$

Berdasarkan Rumus 2.34, pada ruang tiga dimensi, fitur ini menunjukkan sumbu menengah dari elipsoid ekuivalen. Pada ruang dua dimensi, fitur ini menunjukkan sumbu minor dari elips ekuivalen ROI.

(p) *Elongation*

$$Elongation = \sqrt{\frac{\lambda_{minor}}{\lambda_{major}}} \tag{2.35}$$

Berdasarkan Persamaan 2.35, fitur *Elongation* digunakan untuk mengukur tingkat pemanjangan *Region of Interest* (ROI). Pada analisis tiga dimensi, nilai *elongation* menggambarkan hubungan antara sumbu utama dan sumbu minor dari elipsoid ekuivalen. Sementara itu, pada analisis dua dimensi, nilai tersebut menggambarkan hubungan antara sumbu mayor dan sumbu minor dari elips ekuivalen. Nilai yang mendekati angka satu menunjukkan bentuk yang relatif simetris (membulat), sedangkan nilai yang semakin kecil menunjukkan bentuk yang semakin memanjang.

(q) *Flatness*

$$Flatness = \sqrt{\frac{\lambda_{least}}{\lambda_{major}}} \tag{2.36}$$

Berdasarkan Rumus 2.36, fitur *Flatness* digunakan untuk mengukur tingkat kepipihan bentuk ROI. Nilai yang mendekati 1 menunjukkan bentuk yang relatif seimbang pada seluruh sumbu, sedangkan nilai yang semakin kecil menunjukkan bahwa objek memiliki dimensi ketiga yang jauh lebih kecil dibandingkan dimensi utamanya sehingga cenderung berbentuk pipih. Pada rumus tersebut, perhitungan dilakukan menggunakan rasio antara nilai *eigen* terkecil dan nilai *eigen* terbesar.

Secara keseluruhan, fitur-fitur *shape* 3D digunakan untuk merepresentasikan karakteristik geometris dan morfologis tumor tanpa mempertimbangkan distribusi intensitas piksel atau *voxel*. Informasi yang diperoleh mencakup ukuran, volume, luas permukaan, tingkat kekompakan, kemiripan terhadap bentuk bola, serta orientasi dan dimensi objek dalam ruang tiga dimensi. Karakteristik tersebut berperan penting dalam analisis radiomik karena perubahan bentuk dan morfologi tumor sering kali berkaitan dengan tingkat agresivitas dan stadium penyakit. Untuk Rumus-rumus fitur bentuk 2D dijabarkan sebagai berikut.

(a) *Mesh Surface*

$$A = \sum_{i=1}^{N_f} \frac{1}{2}(Oa_i \times Ob_i) \tag{2.37}$$

Berdasarkan Rumus 2.37, fitur *Mesh Surface* digunakan untuk menghitung luas permukaan ROI yang direpresentasikan dalam bentuk *mesh* dua dimensi. Perhitungan dilakukan dengan menjumlahkan luas seluruh elemen penyusun permukaan yang dibentuk oleh titik-titik pada batas ROI. Nilai *Mesh Surface* merepresentasikan ukuran area geometris objek yang diamati. Pada rumus tersebut, N_f menyatakan jumlah elemen permukaan yang menyusun *mesh*, sedangkan a_i dan b_i merupakan titik-titik penyusun elemen ke-i.

(b) *Pixel Surface*

$$A_{pixel} = \sum_{k=1}^{N_v} A_k \tag{2.38}$$

Berdasarkan Rumus 2.38, fitur *Pixel Surface* digunakan untuk menghitung luas ROI berdasarkan jumlah seluruh piksel yang termasuk ke dalam area segmentasi. Luas total diperoleh dari penjumlahan luas masing-masing piksel yang menyusun ROI. Nilai yang lebih besar menunjukkan area lesi atau tumor yang lebih luas pada citra dua dimensi. Pada rumus tersebut, N_v merupakan jumlah piksel yang termasuk dalam ROI dan A_k menyatakan luas piksel ke-k.

(c) *Perimeter*

$$P = \sum_{i=1}^{N_f} \sqrt{(a_i - b_i)^2} \tag{2.39}$$

Berdasarkan Rumus 2.39, fitur *Perimeter* digunakan untuk mengukur panjang total batas atau kontur ROI. Nilai ini diperoleh dengan menjumlahkan jarak *Euclidean* antar titik-titik yang membentuk tepi objek. Semakin besar nilai *Perimeter*, semakin panjang batas ROI yang diamati. Fitur ini sering digunakan untuk menggambarkan kompleksitas bentuk objek pada citra dua dimensi.

(d) *Perimeter to Surface Ratio*

$$\textit{Perimeter to Surface Ratio} = \frac{P}{A} \tag{2.40}$$

Berdasarkan Rumus 2.40, fitur *Perimeter to Surface Ratio* digunakan untuk mengukur hubungan antara panjang batas objek dan luas area ROI. Nilai yang tinggi menunjukkan objek dengan batas yang relatif panjang dibandingkan luas areanya, yang umumnya mengindikasikan

bentuk yang lebih kompleks atau tidak beraturan. Sebaliknya, nilai yang rendah menunjukkan bentuk yang lebih kompak.

(e) *Sphericity* (*2D Circularity*)

$$Sphericity = \frac{2\pi R}{P} = \frac{2\sqrt{\pi A}}{P} \tag{2.41}$$

Berdasarkan Rumus 2.41, fitur *Sphericity* atau *Circularity* digunakan untuk mengukur tingkat kemiripan bentuk ROI terhadap lingkaran sempurna pada ruang dua dimensi. Nilai fitur ini berada pada rentang 0 hingga 1, di mana nilai yang semakin mendekati 1 menunjukkan bentuk yang semakin menyerupai lingkaran. Sebaliknya, nilai yang lebih rendah menunjukkan bentuk yang semakin tidak beraturan. Pada rumus tersebut, R merupakan jari-jari lingkaran yang memiliki luas area yang sama dengan ROI.

(f) *Spherical Disproportion* (2D)

$$Spherical\ Disproportion = \frac{P}{2\sqrt{\pi A}} \tag{2.42}$$

Berdasarkan Rumus 2.42, fitur *Spherical Disproportion* digunakan untuk mengukur tingkat penyimpangan bentuk ROI terhadap lingkaran sempurna. Fitur ini diperoleh dengan membandingkan perimeter aktual ROI terhadap perimeter lingkaran yang memiliki luas area yang sama. Nilai yang mendekati 1 menunjukkan bentuk yang menyerupai lingkaran, sedangkan nilai yang lebih besar menunjukkan bentuk yang semakin tidak beraturan.

(g) *Maximum 2D Diameter*

$$Maximum\ 2D\ Diameter = \max(||x_i - x_j||) \tag{2.43}$$

Berdasarkan Rumus 2.43, fitur *Maximum 2D Diameter* digunakan untuk mengukur jarak *Euclidean* maksimum antara dua titik pada batas ROI dalam ruang dua dimensi. Nilai ini merepresentasikan dimensi terbesar objek yang dapat diamati pada bidang citra. Fitur ini sering digunakan sebagai indikator ukuran maksimum lesi atau tumor pada suatu irisan citra.

Secara keseluruhan, fitur-fitur *shape* dua dimensi digunakan untuk merepresentasikan karakteristik geometris ROI pada suatu irisan citra tanpa mempertimbangkan informasi intensitas piksel. Fitur-fitur tersebut mencakup ukuran area, panjang batas, tingkat kekompakan, kemiripan terhadap bentuk lingkaran, serta orientasi dan dimensi objek. Karakteristik geometris tersebut dapat memberikan informasi penting mengenai morfologi lesi atau tumor yang berpotensi berkorelasi dengan tingkat agresivitas dan perkembangan penyakit.

3. *Gray Level Co-occurrence Matrix* (GLCM)

   GLCM mengukur hubungan spasial antar pasangan *voxel* dengan nilai intensitas tertentu pada jarak dan arah tertentu. Fitur seperti *Contrast*, *Correlation*, *Homogeneity*, dan *Dissimilarity* digunakan untuk menilai keteraturan pola tekstur serta kompleksitas struktur jaringan tumor. Rumus-rumus GLCM dijabarkan sebagai berikut.

   (a) *Autocorrelation*

$$Autocorrelation = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} p(i,j)ij \tag{2.44}$$

   Berdasarkan Rumus 2.44, fitur *Autocorrelation* digunakan untuk mengukur tingkat ketergantungan linear antar pasangan tingkat keabuan yang muncul pada matriks GLCM. Nilai yang tinggi menunjukkan adanya pola tekstur yang lebih teratur dan berulang pada ROI.

   (b) *Joint Average*

$$Joint\ Average = \mu_x = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} p(i,j)i \tag{2.45}$$

   Berdasarkan Rumus 2.45, fitur *Joint Average* digunakan untuk menghitung nilai rata-rata distribusi probabilitas tingkat keabuan pada matriks GLCM. Fitur ini merepresentasikan kecenderungan umum intensitas piksel yang berkontribusi terhadap pembentukan tekstur ROI.

   (c) *Cluster Prominence*

$$Cluster\ Prominence = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i+j-\mu_x-\mu_y)^4 p(i,j) \tag{2.46}$$

Berdasarkan Rumus 2.46, fitur *Cluster Prominence* digunakan untuk mengukur tingkat asimetri dan keruncingan distribusi kelompok tingkat keabuan pada matriks GLCM. Nilai yang tinggi menunjukkan adanya distribusi tekstur yang lebih kompleks dan tidak seragam.

(d) *Cluster Shade*

$$Cluster\ Shade = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g}(i+j-\mu_x-\mu_y)^3 p(i,j) \tag{2.47}$$

Berdasarkan Rumus 2.47, fitur *Cluster Shade* digunakan untuk mengukur tingkat kemencengan (*skewness*) distribusi kelompok tingkat keabuan. Nilai positif atau negatif menunjukkan arah dominasi distribusi tekstur terhadap rata-ratanya.

(e) *Cluster Tendency*

$$Cluster\ Tendency = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g}(i+j-\mu_x-\mu_y)^2 p(i,j) \tag{2.48}$$

Berdasarkan Rumus 2.48, fitur *Cluster Tendency* digunakan untuk mengukur kecenderungan pasangan tingkat keabuan membentuk kelompok di sekitar nilai rata-rata. Nilai yang tinggi menunjukkan adanya pengelompokan intensitas yang lebih kuat pada ROI.

(f) *Contrast*

$$Contrast = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g}(i-j)^2 p(i,j) \tag{2.49}$$

Berdasarkan Rumus 2.49, fitur *Contrast* digunakan untuk mengukur variasi lokal atau perbedaan intensitas antar piksel yang bertetangga. Nilai *Contrast* yang tinggi menunjukkan tekstur yang lebih heterogen, sedangkan nilai yang rendah menunjukkan tekstur yang lebih homogen.

(g) *Correlation*

$$Correlation = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j)ij - \mu_x\mu_y}{\sigma_x(i)\sigma_y(j)} \tag{2.50}$$

Berdasarkan Rumus 2.50, fitur *Correlation* digunakan untuk mengukur tingkat hubungan linear antara pasangan tingkat keabuan pada matriks

GLCM. Nilai yang tinggi menunjukkan adanya keterkaitan yang kuat antara intensitas piksel yang berdekatan.

(h) *Difference Average*

$$Difference\ Average = \sum_{k=0}^{N_g-1} k\, p_{x-y}(k) \tag{2.51}$$

Berdasarkan Rumus 2.51, fitur *Difference Average* digunakan untuk menghitung rata-rata selisih absolut tingkat keabuan antar pasangan piksel pada matriks GLCM. Nilai yang tinggi menunjukkan perbedaan intensitas yang lebih besar dalam ROI.

(i) *Difference Entropy*

$$Difference\ Entropy = \sum_{k=0}^{N_g-1} p_{x-y}(k) \log_2 \left(p_{x-y}(k) + \varepsilon\right) \tag{2.52}$$

Berdasarkan Rumus 2.52, fitur *Difference Entropy* digunakan untuk mengukur tingkat ketidakpastian distribusi selisih tingkat keabuan antar piksel. Nilai yang tinggi menunjukkan kompleksitas tekstur yang lebih besar.

(j) *Difference Variance*

$$Difference\ Variance = \sum_{k=0}^{N_g-1} (k - DA)^2 p_{x-y}(k) \tag{2.53}$$

Berdasarkan Rumus 2.53, fitur *Difference Variance* digunakan untuk mengukur variasi distribusi selisih tingkat keabuan pada ROI. Nilai yang besar menunjukkan heterogenitas tekstur yang lebih tinggi.

(k) *Joint Energy*

$$Joint\ Energy = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (p(i,j))^2 \tag{2.54}$$

Berdasarkan Rumus 2.54, fitur *Joint Energy* digunakan untuk mengukur tingkat keseragaman distribusi probabilitas pada matriks GLCM. Nilai yang tinggi menunjukkan tekstur yang lebih homogen dengan distribusi probabilitas yang terkonsentrasi.

(l) *Joint Entropy*

$$Joint\ Entropy = -\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j)\log_2\left(p(i,j)+\varepsilon\right) \tag{2.55}$$

Berdasarkan Rumus 2.55, fitur *Joint Entropy* digunakan untuk mengukur tingkat ketidakpastian distribusi probabilitas pada matriks GLCM. Nilai yang tinggi menunjukkan tekstur yang lebih kompleks dan tidak teratur.

(m) *Informational Measure of Correlation 1* (IMC1)

$$IMC1 = \frac{HXY - HXY1}{\max(HX, HY)} \tag{2.56}$$

Berdasarkan Rumus 2.56, fitur *Informational Measure of Correlation 1* (IMC1) digunakan untuk mengukur tingkat ketergantungan informasi antara distribusi marginal dan distribusi gabungan pada matriks GLCM. Nilai yang besar menunjukkan hubungan statistik yang lebih kuat antar tingkat keabuan.

(n) *Informational Measure of Correlation 2* (IMC2)

$$IMC2 = \sqrt{1 - e^{-2(HXY2 - HXY)}} \tag{2.57}$$

Berdasarkan Rumus 2.57, fitur *Informational Measure of Correlation 2* (IMC2) digunakan untuk mengukur derajat korelasi informasi antar pasangan tingkat keabuan berdasarkan konsep entropi. Nilai yang lebih tinggi menunjukkan keterkaitan tekstur yang lebih kuat.

(o) *Inverse Difference Moment* (IDM)

$$IDM = \sum_{k=0}^{N_g-1} \frac{p_{x-y}(k)}{1-k^2} \tag{2.58}$$

Berdasarkan Rumus 2.58, fitur *Inverse Difference Moment* (IDM) digunakan untuk mengukur homogenitas lokal tekstur. Nilai yang tinggi menunjukkan bahwa pasangan piksel cenderung memiliki tingkat keabuan yang serupa.

(p) *Maximal Correlation Coefficient* (MCC)

$$MCC = \sqrt{\lambda_2} \tag{2.59}$$

dengan λ_2 merupakan *eigenvalue* terbesar kedua dari matriks Q yang didefinisikan sebagai:

$$Q(i,j) = \sum_{k=0}^{N_g} \frac{p(i,k)p(j,k)}{p_x(i)p_y(k)} \tag{2.60}$$

Berdasarkan Rumus 2.59, fitur *Maximal Correlation Coefficient* (MCC) digunakan untuk mengukur kompleksitas hubungan antar tingkat keabuan pada matriks GLCM melalui analisis nilai *eigen*. Nilai yang tinggi menunjukkan adanya pola keterkaitan tekstur yang lebih kuat dan kompleks. Pada rumus tersebut, λ_2 merupakan nilai *eigen* terbesar kedua dari matriks Q yang didefinisikan pada Rumus 2.60.

(q) *Inverse Difference Moment Normalized* (IDMN)

$$IDMN = \sum_{k=0}^{N_g-1} \frac{p_{x-y}(k)}{1+\left(\frac{k}{N_g}\right)^2} \tag{2.61}$$

Berdasarkan Rumus 2.61, fitur *Inverse Difference Moment Normalized* (IDMN) digunakan untuk mengukur homogenitas tekstur dengan mempertimbangkan normalisasi terhadap jumlah tingkat keabuan. Nilai yang tinggi menunjukkan distribusi intensitas yang lebih seragam.

(r) *Inverse Difference* (ID)

$$ID = \sum_{k=0}^{N_g-1} \frac{p_{x-y}(k)}{1+k} \tag{2.62}$$

Berdasarkan Rumus 2.62, fitur *Inverse Difference* (ID) digunakan untuk mengukur tingkat kesamaan antar pasangan tingkat keabuan. Nilai yang tinggi menunjukkan bahwa piksel yang bertetangga memiliki intensitas yang relatif serupa.

(s) *Inverse Difference Normalized* (IDN)

$$IDN = \sum_{k=0}^{N_g-1} \frac{p_{x-y}(k)}{1+\left(\frac{k}{N_g}\right)} \tag{2.63}$$

Berdasarkan Rumus 2.63, fitur *Inverse Difference Normalized* (IDN) digunakan untuk mengukur homogenitas tekstur dengan pendekatan normalisasi terhadap rentang tingkat keabuan. Nilai yang lebih tinggi menunjukkan tekstur yang lebih homogen.

(t) *Inverse Variance*

$$Inverse\ Variance = \sum_{k=1}^{N_g-1} \frac{p_{x-y}(k)}{k^2} \tag{2.64}$$

Berdasarkan Rumus 2.64, fitur *Inverse Variance* digunakan untuk mengukur kontribusi pasangan piksel yang memiliki perbedaan tingkat keabuan kecil. Nilai yang tinggi menunjukkan dominasi pasangan piksel dengan intensitas yang mirip.

(u) *Maximum Probability*

$$Maximum\ Probability = \max(p(i,j)) \tag{2.65}$$

Berdasarkan Rumus 2.65, fitur *Maximum Probability* digunakan untuk menentukan probabilitas kemunculan pasangan tingkat keabuan yang paling dominan pada matriks GLCM. Nilai yang tinggi menunjukkan adanya pola tekstur tertentu yang muncul secara dominan.

(v) *Sum Average*

$$Sum\ Average = \sum_{k=2}^{2N_g} p_{x+y}(k)k \tag{2.66}$$

Berdasarkan Rumus 2.66, fitur *Sum Average* digunakan untuk menghitung rata-rata distribusi jumlah pasangan tingkat keabuan pada matriks GLCM. Fitur ini menggambarkan kecenderungan umum nilai intensitas gabungan dalam ROI.

(w) *Sum Entropy*

$$Sum\ Entropy = \sum_{k=2}^{2N_g} p_{x+y}(k) \log_2 \left(p_{x+y}(k) + \varepsilon\right) \tag{2.67}$$

Berdasarkan Rumus 2.67, fitur *Sum Entropy* digunakan untuk mengukur tingkat ketidakpastian distribusi jumlah pasangan tingkat keabuan. Nilai yang tinggi menunjukkan tekstur yang lebih kompleks dan beragam.

(x) *Sum of Squares* (*Variance*)

$$Sum\ of\ Squares = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i - \mu_x)^2 p(i,j) \tag{2.68}$$

Berdasarkan Rumus 2.68, fitur *Sum of Squares* atau *Variance* digunakan untuk mengukur tingkat penyebaran tingkat keabuan terhadap nilai rata-ratanya pada matriks GLCM. Nilai yang tinggi menunjukkan variasi intensitas yang lebih besar dan tekstur yang lebih heterogen.

Secara keseluruhan, fitur-fitur *Gray Level Co-occurrence Matrix* (GLCM) digunakan untuk merepresentasikan karakteristik tekstur orde kedua dengan mempertimbangkan hubungan spasial antar pasangan piksel atau *voxel* pada ROI. Fitur-fitur tersebut mampu menggambarkan tingkat homogenitas, heterogenitas, kompleksitas, keteraturan, serta korelasi distribusi intensitas pada jaringan tumor. Informasi tekstur yang diperoleh dari GLCM banyak digunakan dalam analisis radiomik karena mampu menangkap pola mikroskopis yang tidak dapat diamati secara visual oleh radiolog dan berpotensi berkaitan dengan karakteristik biologis serta stadium penyakit.

4. *Gray Level Run Length Matrix* (GLRLM)

GLRLM mengkuantifikasi panjang deretan (*run*) *voxel* yang memiliki tingkat keabuan yang sama pada arah tertentu. Setiap *run* merepresentasikan sekumpulan *voxel* berurutan dengan intensitas identik. Fitur yang diekstraksi, seperti *Short Run Emphasis* (SRE) dan *Long Run Emphasis* (LRE), digunakan untuk mengarakterisasi tekstur citra, khususnya dalam mengidentifikasi pola halus (*run* pendek) dan kasar (*run* panjang) pada jaringan tumor. Rumus-rumus GLRLM dijabarkan sebagai berikut.

(a) *Short Run Emphasis* (SRE)

$$SRE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \frac{\mathbf{P}(i,j|\theta)}{j^2}}{N_r(\theta)} \tag{2.69}$$

Berdasarkan Rumus 2.69, fitur *Short Run Emphasis* (SRE) digunakan untuk mengukur dominasi *run* dengan panjang pendek pada ROI. Nilai SRE yang tinggi menunjukkan bahwa tekstur didominasi oleh *run* pendek sehingga cenderung memiliki pola yang lebih halus (*fine texture*).

(b) *Long Run Emphasis* (LRE)

$$LRE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \mathbf{P}(i,j|\theta) j^2}{N_r(\theta)} \tag{2.70}$$

Berdasarkan Rumus 2.70, fitur *Long Run Emphasis* (LRE) digunakan untuk mengukur dominasi *run* dengan panjang yang besar pada ROI. Nilai LRE yang tinggi menunjukkan keberadaan area homogen yang luas dan tekstur yang lebih kasar (*coarse texture*).

(c) *Gray Level Non-Uniformity* (GLN)

$$GLN = \frac{\sum_{i=1}^{N_g} \left(\sum_{j=1}^{N_r} \mathbf{P}(i,j|\theta) \right)^2}{N_r(\theta)} \tag{2.71}$$

Berdasarkan Rumus 2.71, fitur *Gray Level Non-Uniformity* (GLN) digunakan untuk mengukur tingkat keseragaman distribusi tingkat keabuan pada matriks GLRLM. Nilai yang tinggi menunjukkan bahwa hanya beberapa tingkat keabuan yang mendominasi tekstur, sedangkan nilai yang rendah menunjukkan distribusi tingkat keabuan yang lebih merata.

(d) *Gray Level Non-Uniformity Normalized* (GLNN)

$$GLNN = \frac{\sum_{i=1}^{N_g} \left(\sum_{j=1}^{N_r} \mathbf{P}(i,j|\theta) \right)^2}{N_r(\theta)^2} \tag{2.72}$$

Berdasarkan Rumus 2.72, fitur *Gray Level Non-Uniformity Normalized* (GLNN) digunakan untuk mengukur ketidakseragaman tingkat keabuan

yang telah dinormalisasi terhadap jumlah total *run*. Nilai yang lebih rendah menunjukkan distribusi tingkat keabuan yang lebih homogen.

(e) *Run Length Non-Uniformity* (RLN)

$$RLN = \frac{\sum_{j=1}^{N_r} \left(\sum_{i=1}^{N_g} \mathbf{P}(i,j|\theta) \right)^2}{N_r(\theta)} \tag{2.73}$$

Berdasarkan Rumus 2.73, fitur *Run Length Non-Uniformity* (RLN) digunakan untuk mengukur variasi panjang *run* pada ROI. Nilai yang tinggi menunjukkan bahwa panjang *run* didominasi oleh ukuran tertentu, sedangkan nilai yang rendah menunjukkan distribusi panjang *run* yang lebih merata.

(f) *Run Length Non-Uniformity Normalized* (RLNN)

$$RLNN = \frac{\sum_{j=1}^{N_r} \left(\sum_{i=1}^{N_g} \mathbf{P}(i,j|\theta) \right)^2}{N_r(\theta)^2} \tag{2.74}$$

Berdasarkan Rumus 2.74, fitur *Run Length Non-Uniformity Normalized* (RLNN) digunakan untuk mengukur ketidakseragaman panjang *run* yang telah dinormalisasi. Nilai yang rendah menunjukkan distribusi panjang *run* yang lebih seragam dalam ROI.

(g) *Run Percentage* (RP)

$$RP = \frac{N_r(\theta)}{N_p} \tag{2.75}$$

Berdasarkan Rumus 2.75, fitur *Run Percentage* (RP) digunakan untuk mengukur kepadatan *run* relatif terhadap jumlah total piksel atau *voxel* pada ROI. Nilai yang tinggi menunjukkan tekstur yang lebih kompleks dengan jumlah *run* yang lebih banyak.

(h) *Gray Level Variance* (GLV)

$$GLV = \sum_{i=1}^{N_g} \sum_{j=1}^{N_r} p(i,j|\theta)(i-\mu)^2 \tag{2.76}$$

Berdasarkan Rumus 2.76, fitur *Gray Level Variance* (GLV) digunakan untuk mengukur variasi tingkat keabuan pada matriks GLRLM terhadap nilai rata-ratanya. Nilai GLV yang tinggi menunjukkan heterogenitas

intensitas yang lebih besar pada ROI.

(i) *Run Variance* (RV)

$$RV = \sum_{i=1}^{N_g} \sum_{j=1}^{N_r} p(i,j|\theta)(j-\mu)^2 \tag{2.77}$$

Berdasarkan Rumus 2.77, fitur *Run Variance* (RV) digunakan untuk mengukur variasi panjang *run* terhadap nilai rata-ratanya. Nilai yang tinggi menunjukkan keragaman panjang *run* yang lebih besar dalam ROI.

(j) *Run Entropy* (RE)

$$RE = -\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} p(i,j|\theta) \log_2 \left(p(i,j|\theta) + \varepsilon\right) \tag{2.78}$$

Berdasarkan Rumus 2.78, fitur *Run Entropy* (RE) digunakan untuk mengukur tingkat ketidakpastian atau kompleksitas distribusi *run* pada matriks GLRLM. Nilai yang tinggi menunjukkan tekstur yang lebih heterogen dan tidak teratur.

(k) *Low Gray Level Run Emphasis* (LGLRE)

$$LGLRE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \frac{\mathbf{P}(i,j|\theta)}{i^2}}{N_r(\theta)} \tag{2.79}$$

Berdasarkan Rumus 2.79, fitur *Low Gray Level Run Emphasis* (LGLRE) digunakan untuk mengukur dominasi *run* yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan bahwa ROI didominasi oleh area dengan intensitas rendah.

(l) *High Gray Level Run Emphasis* (HGLRE)

$$HGLRE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \mathbf{P}(i,j|\theta) i^2}{N_r(\theta)} \tag{2.80}$$

Berdasarkan Rumus 2.80, fitur *High Gray Level Run Emphasis* (HGLRE) digunakan untuk mengukur dominasi *run* yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan keberadaan area dengan intensitas tinggi yang dominan pada ROI.

(m) *Short Run Low Gray Level Emphasis* (SRLGLE)

$$SRLGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \frac{\mathbf{P}(i,j|\theta)}{i^2 j^2}}{N_r(\theta)} \tag{2.81}$$

Berdasarkan Rumus 2.81, fitur *Short Run Low Gray Level Emphasis* (SRLGLE) digunakan untuk mengukur dominasi *run* pendek yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan banyaknya area kecil dengan intensitas rendah dalam ROI.

(n) *Short Run High Gray Level Emphasis* (SRHGLE)

$$SRHGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \frac{\mathbf{P}(i,j|\theta) i^2}{j^2}}{N_r(\theta)} \tag{2.82}$$

Berdasarkan Rumus 2.82, fitur *Short Run High Gray Level Emphasis* (SRHGLE) digunakan untuk mengukur dominasi *run* pendek yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan banyaknya area kecil dengan intensitas tinggi pada ROI.

(o) *Long Run Low Gray Level Emphasis* (LRLGLE)

$$LRLGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \frac{\mathbf{P}(i,j|\theta) j^2}{i^2}}{N_r(\theta)} \tag{2.83}$$

Berdasarkan Rumus 2.83, fitur *Long Run Low Gray Level Emphasis* (LRLGLE) digunakan untuk mengukur dominasi *run* panjang yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan keberadaan area homogen yang luas dengan intensitas relatif rendah.

(p) *Long Run High Gray Level Emphasis* (LRHGLE)

$$LRHGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_r} \mathbf{P}(i,j|\theta) i^2 j^2}{N_r(\theta)} \tag{2.84}$$

Berdasarkan Rumus 2.84, fitur *Long Run High Gray Level Emphasis* (LRHGLE) digunakan untuk mengukur dominasi *run* panjang yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan keberadaan area homogen yang luas dengan intensitas tinggi pada ROI.

Secara keseluruhan, fitur-fitur *Gray Level Run Length Matrix* (GLRLM)

digunakan untuk merepresentasikan karakteristik tekstur berdasarkan panjang deret (*run*) piksel atau *voxel* yang memiliki tingkat keabuan yang sama pada arah tertentu. Fitur-fitur tersebut mampu menggambarkan tingkat homogenitas, heterogenitas, distribusi tingkat keabuan, serta variasi panjang *run* dalam ROI. Informasi yang dihasilkan oleh GLRLM memberikan deskripsi tekstur yang lebih mendalam dibandingkan statistik orde pertama karena mempertimbangkan keteraturan pola spasial jaringan tumor, sehingga berpotensi menjadi indikator penting dalam analisis radiomik dan klasifikasi stadium penyakit.

5. *Gray Level Size Zone Matrix* (GLSZM)

   GLSZM mengkuantifikasi ukuran zona *voxel* yang saling terhubung dan memiliki tingkat keabuan yang sama tanpa mempertimbangkan arah. Setiap zona merepresentasikan sekumpulan *voxel* dengan intensitas identik yang membentuk suatu area homogen. Fitur yang diekstraksi, seperti *Small Area Emphasis* (SAE) dan *Large Area Emphasis* (LAE), digunakan untuk mengarakterisasi distribusi ukuran zona yang berkaitan dengan tingkat heterogenitas tekstur pada area tumor. Rumus-rumus GLSZM dijabarkan sebagai berikut.

   (a) *Small Area Emphasis* (SAE)

   $$SAE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_s} \frac{\mathbf{P}(i,j)}{j^2}}{N_z} \tag{2.85}$$

   Berdasarkan Rumus 2.85, fitur *Small Area Emphasis* (SAE) digunakan untuk mengukur dominasi zona berukuran kecil pada ROI. Nilai SAE yang tinggi menunjukkan bahwa tekstur didominasi oleh banyak zona homogen berukuran kecil sehingga mengindikasikan tekstur yang lebih halus dan heterogen. Pada rumus tersebut, N_z menyatakan jumlah total zona pada matriks GLSZM.

   (b) *Large Area Emphasis* (LAE)

   $$LAE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_s} \mathbf{P}(i,j) j^2}{N_z} \tag{2.86}$$

   Berdasarkan Rumus 2.86, fitur *Large Area Emphasis* (LAE) digunakan untuk mengukur dominasi zona berukuran besar pada ROI. Nilai LAE

yang tinggi menunjukkan keberadaan area homogen yang luas sehingga mengindikasikan tekstur yang lebih kasar dan seragam.

(c) *Gray Level Non-Uniformity* (GLN)

$$GLN = \frac{\sum_{i=1}^{N_g} \left(\sum_{j=1}^{N_s} \mathbf{P}(i,j) \right)^2}{N_z} \tag{2.87}$$

Berdasarkan Rumus 2.87, fitur *Gray Level Non-Uniformity* (GLN) digunakan untuk mengukur tingkat ketidakseragaman distribusi tingkat keabuan pada seluruh zona. Nilai GLN yang tinggi menunjukkan bahwa hanya beberapa tingkat keabuan yang mendominasi ROI, sedangkan nilai yang rendah menunjukkan distribusi tingkat keabuan yang lebih merata.

(d) *Gray Level Non-Uniformity Normalized* (GLNN)

$$GLNN = \frac{\sum_{i=1}^{N_g} \left(\sum_{j=1}^{N_s} \mathbf{P}(i,j) \right)^2}{N_z^2} \tag{2.88}$$

Berdasarkan Rumus 2.88, fitur *Gray Level Non-Uniformity Normalized* (GLNN) digunakan untuk mengukur ketidakseragaman tingkat keabuan yang telah dinormalisasi terhadap jumlah total zona. Nilai yang lebih rendah menunjukkan distribusi tingkat keabuan yang lebih homogen pada ROI.

(e) *Size-Zone Non-Uniformity* (SZN)

$$SZN = \frac{\sum_{j=1}^{N_s} \left(\sum_{i=1}^{N_g} \mathbf{P}(i,j) \right)^2}{N_z} \tag{2.89}$$

Berdasarkan Rumus 2.89, fitur *Size-Zone Non-Uniformity* (SZN) digunakan untuk mengukur variasi ukuran zona pada ROI. Nilai yang tinggi menunjukkan bahwa ukuran zona didominasi oleh ukuran tertentu, sedangkan nilai yang rendah menunjukkan distribusi ukuran zona yang lebih seragam.

(f) *Size-Zone Non-Uniformity Normalized* (SZNN)

$$SZNN = \frac{\sum_{j=1}^{N_s} \left(\sum_{i=1}^{N_g} \mathbf{P}(i,j) \right)^2}{N_z^2} \tag{2.90}$$

Berdasarkan Rumus 2.90, fitur *Size-Zone Non-Uniformity Normalized* (SZNN) digunakan untuk mengukur ketidakseragaman ukuran zona yang telah dinormalisasi terhadap jumlah total zona. Nilai yang rendah menunjukkan distribusi ukuran zona yang lebih merata dalam ROI.

(g) *Zone Percentage* (ZP)

$$ZP = \frac{N_z}{N_p} \tag{2.91}$$

Berdasarkan Rumus 2.91, fitur *Zone Percentage* (ZP) digunakan untuk mengukur proporsi jumlah zona terhadap jumlah total piksel atau *voxel* pada ROI. Nilai yang tinggi menunjukkan banyaknya zona homogen berukuran kecil, sedangkan nilai yang rendah menunjukkan keberadaan zona-zona yang relatif lebih besar. Pada rumus tersebut, N_p menyatakan jumlah total piksel atau *voxel* dalam ROI.

(h) *Gray Level Variance* (GLV)

$$GLV = \sum_{i=1}^{N_g} \sum_{j=1}^{N_s} p(i,j)(i-\mu)^2 \tag{2.92}$$

Berdasarkan Rumus 2.92, fitur *Gray Level Variance* (GLV) digunakan untuk mengukur variasi tingkat keabuan antar zona terhadap nilai rata-ratanya. Nilai GLV yang tinggi menunjukkan heterogenitas intensitas yang lebih besar pada ROI. Pada rumus tersebut, μ merupakan rata-rata tingkat keabuan dari distribusi probabilitas GLSZM

(i) *Zone Variance* (ZV)

$$ZV = \sum_{i=1}^{N_g} \sum_{j=1}^{N_s} p(i,j)(j-\mu)^2 \tag{2.93}$$

Berdasarkan Rumus 2.93, fitur *Zone Variance* (ZV) digunakan untuk mengukur variasi ukuran zona terhadap nilai rata-ratanya. Nilai yang tinggi menunjukkan bahwa ukuran zona pada ROI sangat beragam, sedangkan nilai yang rendah menunjukkan ukuran zona yang relatif

seragam. Pada rumus tersebut, μ merupakan rata-rata ukuran zona.

(j) *Zone Entropy* (ZE)

$$ZE = -\sum_{i=1}^{N_g}\sum_{j=1}^{N_s} p(i,j)\log_2\left(p(i,j)+\varepsilon\right) \tag{2.94}$$

Berdasarkan Rumus 2.94, fitur *Zone Entropy* (ZE) digunakan untuk mengukur tingkat ketidakpastian atau kompleksitas distribusi zona pada ROI. Nilai yang tinggi menunjukkan tekstur yang lebih heterogen dan tidak teratur, sedangkan nilai yang rendah menunjukkan pola yang lebih homogen.

(k) *Low Gray Level Zone Emphasis* (LGLZE)

$$LGLZE = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_s}\frac{\mathbf{P}(i,j)}{i^2}}{N_z} \tag{2.95}$$

Berdasarkan Rumus 2.95, fitur *Low Gray Level Zone Emphasis* (LGLZE) digunakan untuk mengukur dominasi zona yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan bahwa ROI didominasi oleh area dengan intensitas rendah.

(l) *High Gray Level Zone Emphasis* (HGLZE)

$$HGLZE = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_s}\mathbf{P}(i,j)i^2}{N_z} \tag{2.96}$$

Berdasarkan Rumus 2.96, fitur *High Gray Level Zone Emphasis* (HGLZE) digunakan untuk mengukur dominasi zona yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan keberadaan area dengan intensitas tinggi yang dominan pada ROI.

(m) *Small Area Low Gray Level Emphasis* (SALGLE)

$$SALGLE = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_s}\frac{\mathbf{P}(i,j)}{i^2 j^2}}{N_z} \tag{2.97}$$

Berdasarkan Rumus 2.97, fitur *Small Area Low Gray Level Emphasis* (SALGLE) digunakan untuk mengukur dominasi zona berukuran kecil yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan banyaknya area kecil dengan intensitas rendah yang tersebar pada ROI.

(n) *Small Area High Gray Level Emphasis* (SAHGLE)

$$SAHGLE = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_s}\frac{\mathbf{P}(i,j)i^2}{j^2}}{N_z} \tag{2.98}$$

Berdasarkan Rumus 2.98, fitur *Small Area High Gray Level Emphasis* (SAHGLE) digunakan untuk mengukur dominasi zona berukuran kecil yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan banyaknya area kecil dengan intensitas tinggi pada ROI.

(o) *Large Area Low Gray Level Emphasis* (LALGLE)

$$LALGLE = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_s}\frac{\mathbf{P}(i,j)j^2}{i^2}}{N_z} \tag{2.99}$$

Berdasarkan Rumus 2.99, fitur *Large Area Low Gray Level Emphasis* (LALGLE) digunakan untuk mengukur dominasi zona berukuran besar yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan keberadaan area homogen yang luas dengan intensitas relatif rendah.

(p) *Large Area High Gray Level Emphasis* (LAHGLE)

$$LAHGLE = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_s}\mathbf{P}(i,j)i^2j^2}{N_z} \tag{2.100}$$

Berdasarkan Rumus 2.100, fitur *Large Area High Gray Level Emphasis* (LAHGLE) digunakan untuk mengukur dominasi zona berukuran besar yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan keberadaan area homogen yang luas dengan intensitas tinggi pada ROI.

Secara keseluruhan, fitur-fitur *Gray Level Size Zone Matrix* (GLSZM) digunakan untuk merepresentasikan karakteristik tekstur berdasarkan distribusi ukuran zona homogen yang terbentuk oleh sekumpulan piksel atau *voxel* dengan tingkat keabuan yang sama tanpa mempertimbangkan arah spasial tertentu. Berbeda dengan GLRLM yang berfokus pada panjang *run*, GLSZM mengukur ukuran area homogen secara langsung sehingga mampu menggambarkan tingkat heterogenitas, homogenitas, variasi ukuran zona, serta distribusi intensitas pada ROI. Karakteristik tersebut sangat bermanfaat dalam analisis radiomik karena perubahan pola dan ukuran zona homogen

sering kali berkaitan dengan kompleksitas struktur jaringan tumor serta perkembangan stadium penyakit.

6. *Gray Level Dependence Matrix* (GLDM)

   GLDM mengkuantifikasi tingkat ketergantungan *voxel* terhadap *voxel* pusat berdasarkan kesamaan intensitas dalam suatu lingkungan lokal. Sebuah *voxel* dianggap dependen jika perbedaan intensitasnya terhadap voxel pusat berada di bawah ambang batas tertentu. Fitur yang dihasilkan, seperti *Dependence Non-Uniformity* (DN) dan *Dependence Entropy* (DE), digunakan untuk mengkarakterisasi kompleksitas tekstur serta variasi struktur internal pada tumor. Rumus-rumus GLDM dijabarkan sebagai berikut.

   (a) *Small Dependence Emphasis* (SDE)

   $$SDE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \frac{\mathbf{P}(i,j)}{i^2}}{N_z} \tag{2.101}$$

   Berdasarkan Rumus 2.101, fitur *Small Dependence Emphasis* (SDE) digunakan untuk mengukur dominasi kelompok ketergantungan (*dependence*) berukuran kecil pada ROI. Nilai SDE yang tinggi menunjukkan bahwa tekstur didominasi oleh ketergantungan lokal yang kecil sehingga mengindikasikan struktur yang lebih halus dan kompleks. Pada rumus tersebut, N_z menyatakan jumlah total ketergantungan yang terdapat pada matriks GLDM.

   (b) *Large Dependence Emphasis* (LDE)

   $$LDE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \mathbf{P}(i,j)\, j^2}{N_z} \tag{2.102}$$

   Berdasarkan Rumus 2.102, fitur *Large Dependence Emphasis* (LDE) digunakan untuk mengukur dominasi kelompok ketergantungan berukuran besar pada ROI. Nilai LDE yang tinggi menunjukkan keberadaan area homogen yang luas sehingga tekstur cenderung lebih seragam dan kurang kompleks.

   (c) *Gray Level Non-Uniformity* (GLN)

   $$GLN = \frac{\sum_{i=1}^{N_g} \left(\sum_{j=1}^{N_d} \mathbf{P}(i,j) \right)^2}{N_z} \tag{2.103}$$

Berdasarkan Rumus 2.103, fitur *Gray Level Non-Uniformity* (GLN) digunakan untuk mengukur tingkat ketidakseragaman distribusi tingkat keabuan pada matriks GLDM. Nilai GLN yang tinggi menunjukkan bahwa hanya beberapa tingkat keabuan yang mendominasi ROI, sedangkan nilai yang rendah menunjukkan distribusi tingkat keabuan yang lebih merata.

(d) *Dependence Non-Uniformity* (DN)

$$DN = \frac{\sum_{j=1}^{N_d} \left(\sum_{i=1}^{N_g} \mathbf{P}(i,j) \right)^2}{N_z} \tag{2.104}$$

Berdasarkan Rumus 2.104, fitur *Dependence Non-Uniformity* (DN) digunakan untuk mengukur tingkat ketidakseragaman ukuran ketergantungan pada ROI. Nilai yang tinggi menunjukkan bahwa ukuran ketergantungan didominasi oleh ukuran tertentu, sedangkan nilai yang rendah menunjukkan distribusi ukuran ketergantungan yang lebih homogen.

(e) *Dependence Non-Uniformity Normalized* (DNN)

$$DNN = \frac{\sum_{j=1}^{N_d} \left(\sum_{i=1}^{N_g} \mathbf{P}(i,j) \right)^2}{N_z^2} \tag{2.105}$$

Berdasarkan Rumus 2.105, fitur *Dependence Non-Uniformity Normalized* (DNN) digunakan untuk mengukur ketidakseragaman ukuran ketergantungan yang telah dinormalisasi terhadap jumlah total ketergantungan. Nilai yang rendah menunjukkan distribusi ukuran ketergantungan yang lebih seragam pada ROI.

(f) *Gray Level Variance* (GLV)

$$GLV = \sum_{i=1}^{N_g} \sum_{j=1}^{N_d} p(i,j)(i-\mu_i)^2 \tag{2.106}$$

Berdasarkan Rumus 2.106, fitur *Gray Level Variance* (GLV) digunakan untuk mengukur variasi tingkat keabuan terhadap nilai rata-ratanya pada matriks GLDM. Nilai GLV yang tinggi menunjukkan heterogenitas intensitas yang lebih besar pada ROI. Pada rumus tersebut, μ_i

merupakan nilai rata-rata tingkat keabuan yang diperoleh dari distribusi probabilitas matriks GLDM.

(g) *Dependence Variance* (DV)

$$DV = \sum_{i=1}^{N_g} \sum_{j=1}^{N_d} p(i,j)(j-\mu_j)^2 \tag{2.107}$$

Berdasarkan Rumus 2.107, fitur *Dependence Variance* (DV) digunakan untuk mengukur variasi ukuran ketergantungan terhadap nilai rata-ratanya. Nilai DV yang tinggi menunjukkan keragaman ukuran ketergantungan yang lebih besar pada ROI. Pada rumus tersebut, μ_j merupakan nilai rata-rata ukuran ketergantungan yang diperoleh dari distribusi probabilitas matriks GLDM.

(h) *Dependence Entropy* (DE)

$$Dependence\ Entropy = -\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} p(i,j) \log_2(p(i,j)+\varepsilon) \tag{2.108}$$

Berdasarkan Rumus 2.108, fitur *Dependence Entropy* (DE) digunakan untuk mengukur tingkat ketidakpastian atau kompleksitas distribusi ketergantungan pada ROI. Nilai DE yang tinggi menunjukkan tekstur yang lebih heterogen dan tidak teratur, sedangkan nilai yang rendah menunjukkan pola tekstur yang lebih homogen.

(i) *Low Gray Level Emphasis* (LGLE)

$$LGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \frac{\mathbf{P}(i,j)}{i^2}}{N_z} \tag{2.109}$$

Berdasarkan Rumus 2.109, fitur *Low Gray Level Emphasis* (LGLE) digunakan untuk mengukur dominasi ketergantungan yang memiliki tingkat keabuan rendah. Nilai LGLE yang tinggi menunjukkan bahwa ROI didominasi oleh area dengan intensitas rendah.

(j) *High Gray Level Emphasis* (HGLE)

$$HGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \mathbf{P}(i,j)\, i^2}{N_z} \tag{2.110}$$

Berdasarkan Rumus 2.110, fitur *High Gray Level Emphasis* (HGLE) digunakan untuk mengukur dominasi ketergantungan yang memiliki tingkat keabuan tinggi. Nilai HGLE yang tinggi menunjukkan keberadaan area dengan intensitas tinggi yang dominan pada ROI.

(k) *Small Dependence Low Gray Level Emphasis* (SDLGLE)

$$SDLGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \frac{\mathbf{P}(i,j)}{i^2 j^2}}{N_z} \tag{2.111}$$

Berdasarkan Rumus 2.111, fitur *Small Dependence Low Gray Level Emphasis* (SDLGLE) digunakan untuk mengukur dominasi ketergantungan berukuran kecil yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan banyaknya struktur kecil dengan intensitas rendah pada ROI.

(l) *Small Dependence High Gray Level Emphasis* (SDHGLE)

$$SDHGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \frac{\mathbf{P}(i,j)\, i^2}{j^2}}{N_z} \tag{2.112}$$

Berdasarkan Rumus 2.112, fitur *Small Dependence High Gray Level Emphasis* (SDHGLE) digunakan untuk mengukur dominasi ketergantungan berukuran kecil yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan banyaknya struktur kecil dengan intensitas tinggi pada ROI.

(m) *Large Dependence Low Gray Level Emphasis* (LDLGLE)

$$LDLGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \frac{\mathbf{P}(i,j)\, j^2}{i^2}}{N_z} \tag{2.113}$$

Berdasarkan Rumus 2.113, fitur *Large Dependence Low Gray Level Emphasis* (LDLGLE) digunakan untuk mengukur dominasi ketergantungan berukuran besar yang memiliki tingkat keabuan rendah. Nilai yang tinggi menunjukkan keberadaan area homogen yang luas dengan intensitas relatif rendah.

(n) *Large Dependence High Gray Level Emphasis* (LDHGLE)

$$LDHGLE = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_d} \mathbf{P}(i,j)\, i^2 j^2}{N_z} \tag{2.114}$$

Berdasarkan Rumus 2.114, fitur *Large Dependence High Gray Level Emphasis* (LDHGLE) digunakan untuk mengukur dominasi ketergantungan berukuran besar yang memiliki tingkat keabuan tinggi. Nilai yang tinggi menunjukkan keberadaan area homogen yang luas dengan intensitas tinggi pada ROI.

7. *Neighboring Gray Tone Difference Matrix* (NGTDM)

NGTDM mengukur perbedaan antara intensitas suatu *voxel* dengan rata-rata intensitas *voxel* di sekitarnya (tetangga lokal). Fitur ini digunakan untuk menggambarkan karakteristik tekstur berdasarkan perubahan intensitas lokal. Beberapa fitur yang dihasilkan, seperti *Coarseness*, *Contrast*, *Busyness*, *Complexity*, dan *Strength*, memberikan informasi mengenai tingkat kekasaran, variasi, serta kompleksitas pola tekstur pada ROI. Rumus-rumus NGTDM dijabarkan sebagai berikut.

(a) *Coarseness*

$$Coarseness = \frac{1}{\sum_{i=1}^{N_g} p_i s_i} \tag{2.115}$$

Berdasarkan Rumus 2.115, fitur *Coarseness* digunakan untuk mengukur tingkat kekasaran tekstur berdasarkan perbedaan intensitas antara setiap *voxel* dengan rata-rata intensitas lingkungan di sekitarnya. Nilai *Coarseness* yang tinggi menunjukkan bahwa perubahan intensitas antarwilayah relatif kecil sehingga tekstur cenderung lebih homogen dan kasar. Sebaliknya, nilai yang rendah menunjukkan adanya perubahan intensitas yang lebih cepat sehingga tekstur menjadi lebih halus. Pada rumus tersebut, p_i menyatakan probabilitas kemunculan tingkat keabuan ke-i, sedangkan s_i merupakan jumlah selisih absolut antara intensitas *voxel* dengan rata-rata intensitas lingkungan tetangganya.

(b) *Contrast*

$$Contrast = \left(\frac{1}{N_{g,p}(N_{g,p}-1)}\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p_i p_j (i-j)^2\right)\left(\frac{1}{N_{v,p}}\sum_{i=1}^{N_g} s_i\right) \tag{2.116}$$

dengan $p_i \neq 0$ dan $p_j \neq 0$.

Berdasarkan Rumus 2.116, fitur *Contrast* digunakan untuk mengukur tingkat perbedaan intensitas antarwilayah pada ROI. Nilai *Contrast* yang tinggi menunjukkan adanya variasi intensitas yang besar sehingga tekstur cenderung lebih heterogen, sedangkan nilai yang rendah menunjukkan distribusi intensitas yang lebih seragam. Pada rumus tersebut, $N_{g,p}$ merupakan jumlah tingkat keabuan yang memiliki probabilitas lebih besar dari nol, sedangkan $N_{v,p}$ menyatakan jumlah *voxel* yang digunakan dalam perhitungan NGTDM.

(c) *Busyness*

$$Busyness = \frac{\sum_{i=1}^{N_g} p_i s_i}{\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} |ip_i - jp_j|} \tag{2.117}$$

dengan $p_i \neq 0$ dan $p_j \neq 0$.

Berdasarkan Rumus 2.117, fitur *Busyness* digunakan untuk mengukur tingkat perubahan intensitas antar *voxel* yang berdekatan. Nilai *Busyness* yang tinggi menunjukkan bahwa intensitas berubah secara cepat pada jarak yang pendek sehingga tekstur menjadi lebih kompleks dan tidak homogen. Sebaliknya, nilai yang rendah menunjukkan perubahan intensitas yang lebih lambat sehingga tekstur cenderung lebih homogen.

(d) *Complexity*

$$Complexity = \frac{1}{N_{v,p}}\sum_{i=1}^{N_g}\sum_{j=1}^{N_g} |i-j|\frac{p_i s_i + p_j s_j}{p_i + p_j} \tag{2.118}$$

dengan $p_i \neq 0$ dan $p_j \neq 0$.

Berdasarkan Rumus 2.118, fitur *Complexity* digunakan untuk mengukur tingkat kompleksitas tekstur berdasarkan variasi intensitas serta distribusi spasial antar tingkat keabuan. Nilai *Complexity* yang tinggi menunjukkan bahwa ROI memiliki struktur tekstur yang lebih kompleks dengan variasi intensitas yang besar antarwilayah. Sebaliknya, nilai

yang rendah menunjukkan pola tekstur yang lebih sederhana dan seragam.

(e) *Strength*

$$\text{Strength} = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_g}(p_i + p_j)(i-j)^2}{\sum_{i=1}^{N_g} s_i} \tag{2.119}$$

dengan $p_i \neq 0$ dan $p_j \neq 0$.

Berdasarkan Rumus 2.119, fitur *Strength* digunakan untuk mengukur kekuatan pola tekstur berdasarkan besarnya perbedaan intensitas antar tingkat keabuan yang masih membentuk struktur lokal yang relatif seragam. Nilai *Strength* yang tinggi menunjukkan adanya pola tekstur yang jelas dengan perbedaan intensitas yang besar antarwilayah, sedangkan nilai yang rendah menunjukkan tekstur yang lebih lemah atau kurang terstruktur.

8. *Gradient Magnitude Filter*

Filter ini menghitung norma *Euclidean* dari gradien intensitas lokal pada sumbu spasial x, y, dan z. Peta magnitudo yang dihasilkan merepresentasikan laju perubahan intensitas spasial yang secara efektif berfungsi sebagai detektor tepi 3D. Area dengan perubahan atenuasi yang tajam, seperti antarmuka antara tumor dan paru-paru, menghasilkan nilai gradien yang tinggi sehingga memungkinkan kuantifikasi presisi terhadap ketidakteraturan tepi tumor, yang dapat terlihat dalam Rumus 2.120 berikut.

$$I_{grad} = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2 + \left(\frac{\partial I}{\partial z}\right)^2} \tag{2.120}$$

9. *Laplacian of Gaussian (LoG) Filter*

Filter LoG beroperasi sebagai filter *band-pass* multiskala dengan mengonvolusi citra asli menggunakan kernel Gaussian G_σ yang diikuti oleh operator Laplacian ∇^2, yang dijabarkan dalam Rumus 2.121 berikut.

$$I_{LoG} = \nabla^2(G_\sigma * I) \tag{2.121}$$

Dimana operator Laplacian dijelaskan dalam Rumus 2.122 berikut.

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} + \frac{\partial^2 I}{\partial z^2} \tag{2.122}$$

dan konvolusi Gaussian awal untuk menekan *noise* pada skala spasial tertentu yang terlihat dalam Rumus 2.123 berikut.

$$G_\sigma(x,y,z) = \frac{1}{(2\pi\sigma^2)^{\frac{3}{2}}} \exp\left(-\frac{x^2+y^2+z^2}{2\sigma^2}\right) \tag{2.123}$$

10. *Wavelet Decomposition*

    Menggunakan transformasi wavelet diskrit 3D $W_{3D}\{\cdot\}$, volume CT didekomposisi menjadi delapan *sub-band* frekuensi dan arah (LLL, LLH, LHL, LHH, HLL, HLH, HHL, dan HHH) melalui penerapan filter *low-pass* (L) dan *high-pass* (H) pada tiga dimensi. Analisis frekuensi multiskala ini memisahkan detail tekstur halus dari representasi struktural yang lebih luas yang dijabarkan dalam Rumus 2.124 berikut.

$$I_{\text{wavelet}} = W_{3D}\{I(x,y,z)\} \tag{2.124}$$

Seluruh parameter matematis yang telah dijabarkan, membentuk profil radiomik yang komprehensif untuk mengkarakterisasi heterogenitas internal serta morfologi tumor. Integrasi antara fitur statistik, bentuk, dan tekstur memungkinkan identifikasi pola mikroskopis pada jaringan yang sering kali berkorelasi dengan perilaku biologis tumor, tingkat invasi, dan respons terhadap terapi. Penerapan rumus-rumus terstandarisasi menjadi aspek penting dalam mengatasi hambatan variabilitas algoritma yang selama ini membatasi komparabilitas antarstudi radiomik. Dengan demikian, fitur-fitur yang diekstraksi tidak hanya berfungsi sebagai deskriptor geometris dan intensitas, tetapi juga sebagai biomarker pencitraan yang memiliki landasan matematis yang kuat untuk mendukung pengambilan keputusan klinis berbasis kecerdasan buatan (*artificial intelligence*).

2.3.2 Normalisasi Data

Normalisasi data merupakan pilar fundamental dalam tahapan pra-pemrosesan data (*data preprocessing*) yang bertujuan untuk mentransformasikan nilai atribut numerik ke dalam skala yang seragam tanpa menghilangkan informasi esensial yang terkandung dalam distribusi asli data [37]. Dalam implementasi *machine learning*, sering ditemukan *dataset* yang memiliki fitur dengan satuan pengukuran dan rentang nilai yang sangat kontras. Sebagai contoh, suatu variabel pendapatan tahunan dapat memiliki rentang nilai dari puluhan juta hingga miliaran, sementara variabel usia hanya berkisar antara nol hingga seratus. Tanpa proses penyetaraan skala, algoritma pembelajaran cenderung memberikan bobot yang tidak proporsional terhadap variabel dengan magnitudo yang lebih besar, sehingga berpotensi mendistorsi proses pembelajaran dan menghasilkan model yang bias serta kurang akurat. Secara ilmiah, normalisasi didefinisikan sebagai metode penskalaan ulang fitur agar berada pada rentang yang lebih kecil dan konsisten, umumnya dalam interval $[0, 1]$ atau $[-1, 1]$. Proses ini tidak hanya memastikan bahwa setiap fitur berkontribusi secara seimbang terhadap fungsi objektif model, tetapi juga berperan penting dalam meningkatkan efisiensi dan stabilitas komputasi.

Urgensi stabilitas komputasi tersebut sangat terasa pada algoritma yang menggunakan teknik optimasi berbasis gradien (*gradient-based optimization*), seperti jaringan saraf tiruan dan regresi logistik. Pada algoritma tersebut, pelatihan model melibatkan pembaruan bobot secara iteratif. Jika fitur memiliki rentang nilai yang sangat berbeda, permukaan fungsi biaya (*cost function*) akan berbentuk elips yang sangat lonjong. Kondisi ini menyebabkan arah gradien berosilasi secara tidak efisien sehingga memperlambat proses konvergensi menuju titik minimum global. Melalui normalisasi, permukaan fungsi biaya menjadi lebih sferis atau mendekati lingkaran, sehingga langkah gradien menuju titik optimal dapat dilakukan secara lebih langsung dan cepat. Selain itu, normalisasi juga berperan penting dalam menghindari kondisi *Not a Number* (NaN), yaitu keadaan ketika nilai fitur yang terlalu besar menyebabkan hasil perhitungan melampaui batas presisi *floating-point* sistem komputer. Apabila kondisi tersebut terjadi, nilai NaN dapat menyebar selama proses propagasi dan mengakibatkan kegagalan total pada proses inferensi model. Dengan demikian, normalisasi memberikan manfaat berupa percepatan konvergensi, keseimbangan pembobotan antarfitur, penjagaan stabilitas numerik untuk mencegah *overflow*, serta optimalisasi perhitungan jarak pada algoritma seperti *K-Nearest Neighbor* (KNN) dan *K-Means* [38].

Untuk mengimplementasikan prinsip-prinsip tersebut, teknik yang paling sering digunakan adalah penskalaan linear atau *Min-Max Normalization*. Metode ini mentransformasikan data mentah secara proporsional ke dalam rentang tertentu (umumnya 0 hingga 1) menggunakan Rumus 2.125 sebagai berikut.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2.125}$$

Dalam persamaan tersebut, x merepresentasikan nilai asli, sedangkan x_{min} dan x_{max} merupakan nilai minimum dan maksimum dari suatu fitur pada keseluruhan *dataset*. Hasil transformasi x' akan bernilai 0 apabila x merupakan nilai minimum dan bernilai 1 apabila x merupakan nilai maksimum. Meskipun metode ini sederhana dan efektif, kelemahan utamanya terletak pada sensitivitas terhadap pencilan (*outlier*). Kehadiran satu nilai ekstrem dapat menekan sebagian besar data ke dalam rentang yang sangat sempit sehingga menyulitkan model dalam mempelajari variasi internal data. Oleh karena itu, *Min-Max Scaling* lebih sesuai diterapkan pada data yang memiliki batas atas dan bawah yang jelas serta tidak mengandung banyak nilai ekstrem yang jauh dari rata-rata.

Sebagai alternatif yang lebih tangguh terhadap keberadaan pencilan, metode *Z-Score Normalization* atau standardisasi sering digunakan. Pendekatan ini tidak membatasi data pada rentang absolut tertentu, melainkan mentransformasikannya berdasarkan ukuran pemusatan dan penyebaran data, yaitu rata-rata (μ) dan simpangan baku (σ). Persamaan yang digunakan seperti dalam Rumus 2.126 sebagai berikut.

$$x' = \frac{x - \mu}{\sigma} \tag{2.126}$$

Melalui transformasi tersebut, data hasil normalisasi akan memiliki rata-rata nol dan simpangan baku satu, di mana nilai x' merepresentasikan jarak suatu data terhadap rata-rata dalam satuan deviasi standar. Keunggulan standardisasi terletak pada kemampuannya memanfaatkan informasi dari keseluruhan distribusi data sehingga pengaruh nilai ekstrem dapat diminimalkan. Metode ini sangat sesuai digunakan pada algoritma yang memiliki asumsi distribusi normal, seperti regresi linear, *Support Vector Machine* (SVM), dan *Linear Discriminant Analysis* (LDA).

2.3.3 Penanganan Ketidakseimbangan Data

Dalam konteks pemodelan prediktif dan *machine learning*, *resampling* merupakan teknik pra-pemrosesan (*pre-processing*) data yang secara khusus diterapkan untuk memitigasi permasalahan ketidakseimbangan kelas (*class imbalance*). Kondisi ini terjadi ketika proporsi distribusi observasi pada satu kelas (kelas mayoritas) mendominasi secara signifikan dibandingkan kelas lainnya (kelas minoritas), misalnya pada rasio 99:1 dalam kasus deteksi penipuan atau klasifikasi medis. Ketidakseimbangan tersebut dapat menyebabkan model klasifikasi cenderung mengabaikan kelas minoritas sehingga menghasilkan prediksi yang bias. Untuk mengurangi bias distribusi dan meningkatkan metrik akurasi berimbang (*balanced accuracy*), pendekatan *resampling* memodifikasi himpunan data latih melalui dua mekanisme utama sebagai berikut:

1. *Oversampling*: Teknik penyeimbangan distribusi dengan cara meningkatkan jumlah data pada kelas minoritas hingga mendekati atau menyamai jumlah data pada kelas mayoritas. Pendekatan paling sederhana adalah *Random Oversampling*, yaitu mereplikasi sampel minoritas secara acak. Namun, metode ini berpotensi menyebabkan *overfitting* akibat duplikasi data yang identik. Untuk mengatasi kelemahan tersebut, dikembangkan metode yang lebih canggih, seperti *Synthetic Minority Oversampling Technique* (SMOTE), yang menghasilkan sampel sintetis baru di antara sampel minoritas yang berdekatan menggunakan pendekatan *k-Nearest Neighbors* (KNN). Selain itu, pendekatan berbasis model generatif, seperti *Generative Adversarial Network* (GAN), juga dapat digunakan untuk menghasilkan data sintetis yang lebih representatif.

2. *Undersampling*: Teknik penyeimbangan distribusi dengan cara mengurangi jumlah sampel pada kelas mayoritas agar proporsinya lebih seimbang dengan kelas minoritas. Metode dasar yang umum digunakan adalah *Random Undersampling*, yaitu menghapus sampel mayoritas secara acak. Meskipun efektif dalam menyeimbangkan distribusi kelas, metode ini berisiko menghilangkan informasi penting yang terkandung dalam data mayoritas. Oleh karena itu, dikembangkan berbagai metode yang lebih selektif, seperti *Tomek Links*, yang menghapus sampel mayoritas yang berada di sekitar batas keputusan klasifikasi, *Near Miss*, yang memilih sampel mayoritas berdasarkan kedekatan spasial terhadap kelas minoritas, serta

Cluster Centroids, yang membentuk representasi prototipe data mayoritas melalui pendekatan klasterisasi.

2.3.4 Regularisasi

Regularisasi merupakan strategi optimisasi dalam *machine learning* yang digunakan untuk mengendalikan kompleksitas model guna mencegah *overfitting* serta meningkatkan kemampuan generalisasi ketika model dihadapkan pada data baru yang tidak digunakan selama proses pelatihan. Mekanisme regularisasi bekerja dengan menambahkan fungsi penalti (*penalty function*) terhadap nilai parameter model ke dalam fungsi biaya (*cost function*) atau fungsi kerugian (*loss function*) yang dioptimalkan selama proses pelatihan.

Melalui penerapan penalti tersebut, model didorong untuk menghasilkan parameter dengan kompleksitas yang lebih rendah sehingga tidak terlalu menyesuaikan diri terhadap pola spesifik pada data latih. Akibatnya, kemampuan model dalam melakukan prediksi pada data yang belum pernah diamati menjadi lebih baik. Pendekatan regularisasi yang paling umum digunakan didasarkan pada norma matematis tertentu yang diterapkan pada parameter model.

A L1 (LASSO)

Regularisasi L1 memberikan penalti linear terhadap jumlah nilai absolut parameter (norma L_1). Berbeda dengan regularisasi Ridge, regularisasi L1 mampu menyusutkan banyak koefisien fitur yang kurang relevan hingga bernilai nol secara eksak, sehingga dapat melakukan seleksi fitur secara otomatis. Metode yang mengimplementasikan konsep ini dikenal sebagai *Least Absolute Shrinkage and Selection Operator* (LASSO), yaitu metode regresi linear yang secara simultan melakukan penyusutan koefisien (*shrinkage*) dan pemilihan variabel (*variable selection*). Optimalisasi pada metode LASSO dilakukan dengan menambahkan penalti norma L_1 ke dalam fungsi galat *least squares*. Dalam bentuk *Lagrangian* (*penalized form*), fungsi objektif LASSO terlihat dalam Rumus2.127 sebagai berikut.

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right\} \tag{2.127}$$

Dalam persamaan tersebut, parameter $\lambda \geq 0$ mengendalikan tingkat

penalti yang diberikan terhadap koefisien model, sedangkan $\|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$ merepresentasikan norma L_1. Kemampuan LASSO menghasilkan solusi yang *sparse* dipengaruhi oleh bentuk geometris daerah pembatas norma L_1 yang memiliki sudut-sudut tajam pada sumbu koordinat. Ketika kurva kontur fungsi galat bersinggungan dengan sudut daerah pembatas tersebut, sebagian koefisien dapat bernilai nol secara eksak. Kondisi ini menyebabkan variabel yang kurang relevan dieliminasi dari model. Akibatnya, model yang dihasilkan menjadi lebih sederhana, lebih mudah diinterpretasikan, dan memiliki kemampuan generalisasi yang lebih baik.

B L2 (Regresi *Ridge*)

Regularisasi L2 menambahkan penalti yang proporsional terhadap kuadrat magnitudo koefisien parameter (norma L_2). Fungsi objektif pada metode *Ridge* dirumuskan dengan menambahkan penalti norma L_2 ke dalam fungsi galat dalam Rumus 2.128 sebagai berikut.

$$\min_{\beta} \left\{ \frac{1}{N} \|y - X\beta\|_2^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{2.128}$$

Pendekatan ini menyusutkan nilai koefisien mendekati nol sehingga mampu mengurangi kompleksitas model dan menekan variansi prediksi. Namun, berbeda dengan LASSO, regularisasi L2 umumnya tidak menghasilkan koefisien yang bernilai nol secara eksak. Oleh karena itu, seluruh variabel tetap dipertahankan di dalam model meskipun kontribusinya relatif kecil.

2.4 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) merupakan salah satu algoritma pembelajaran terawasi (*supervised learning*) yang berlandaskan pada teori pembelajaran statistik dan prinsip minimalisasi risiko struktural. Secara fundamental, algoritma ini bertujuan untuk menemukan sebuah *hyperplane* optimal yang mampu memisahkan *dataset* ke dalam kelas-kelas yang berbeda dengan margin maksimum. Berbeda dengan algoritma klasifikasi konvensional yang hanya berfokus pada minimalisasi kesalahan klasifikasi pada data latih, SVM dirancang untuk meningkatkan kemampuan generalisasi terhadap data yang belum pernah

diamati sebelumnya. Dalam ruang fitur berdimensi p, *hyperplane* didefinisikan sebagai himpunan titik yang memenuhi Rumus 2.129 berikut.

$$w^T x + b = 0 \tag{2.129}$$

dengan w merupakan vektor normal terhadap *hyperplane* dan b merupakan parameter bias yang menentukan posisi bidang terhadap titik asal. Pada klasifikasi biner dengan label $y_i \in \{+1, -1\}$, posisi suatu titik data relatif terhadap *hyperplane* menentukan kelas prediksinya. Fungsi keputusan yang digunakan dinyatakan dalam Rumus 2.130 sebagai berikut.

$$h(x) = \text{sign}(w^T x + b) \tag{2.130}$$

Konsep utama dalam penentuan *hyperplane* adalah *margin*, yaitu jarak tegak lurus minimum antara batas keputusan dan titik data terdekat dari masing-masing kelas. Titik-titik data yang berada tepat pada batas *margin* disebut sebagai *support vectors*. Titik-titik tersebut memiliki peran penting karena posisi dan orientasi *hyperplane* sepenuhnya ditentukan oleh *support vectors*. Perubahan pada titik data lain yang berada jauh dari *margin* umumnya tidak memengaruhi bentuk *hyperplane*. Secara geometris, lebar *margin* berbanding terbalik dengan norma vektor bobot $\|w\|$. Oleh karena itu, proses memaksimalkan *margin* ekuivalen dengan meminimalkan norma vektor bobot. Lebar *margin* maksimum dinyatakan dalam Rumus 2.131 sebagai berikut.

$$\frac{2}{\|w\|} \tag{2.131}$$

Pada kondisi ketika data dapat dipisahkan secara sempurna, SVM menggunakan pendekatan *hard margin*. Pendekatan ini mengharuskan seluruh titik data berada pada sisi yang benar dari *margin* tanpa pelanggaran. Akan tetapi, data dunia nyata umumnya mengandung *noise* dan tumpang tindih antarkelas sehingga pendekatan tersebut menjadi terlalu restriktif dan berpotensi menyebabkan *overfitting*. Untuk mengatasi permasalahan tersebut, digunakan pendekatan *soft margin* yang memperkenalkan variabel *slack* (ξ_i). Variabel ini memungkinkan sejumlah titik data melanggar batas *margin* atau bahkan berada pada sisi yang

salah dari *hyperplane*, dengan konsekuensi berupa penalti pada fungsi objektif. Parameter regularisasi C digunakan untuk mengendalikan keseimbangan antara upaya memaksimalkan *margin* dan meminimalkan kesalahan klasifikasi. Nilai C yang besar akan menghasilkan *margin* yang lebih sempit dengan tingkat kesalahan yang lebih rendah pada data latih, sedangkan nilai C yang kecil memberikan toleransi kesalahan yang lebih tinggi sehingga menghasilkan *margin* yang lebih lebar dan kemampuan generalisasi yang lebih baik.

Meskipun *margin* lunak memberikan fleksibilitas, ada kalanya pola data benar-benar non-linear. Untuk menangani kompleksitas ini, SVM dibekali dengan sebuah mekanisme cerdas yang dikenal sebagai *Kernel Trick*. Prinsip dasar dari teknik ini adalah memetakan data *input* asli ke dalam ruang fitur berdimensi lebih tinggi—bahkan hingga dimensi tak terhingga—sehingga data yang awalnya rumit dapat dipisahkan secara linear oleh *hyperplane* di ruang yang baru. Alih-alih melakukan transformasi koordinat secara eksplisit yang akan membebani komputasi secara eksponensial, fungsi *kernel* bekerja dengan menghitung hasil kali titik (*dot product*) antar pasangan data langsung di ruang aslinya. Proses perhitungan *kernel* ini menjadi inti dari pembaruan nilai variabel dalam algoritma, yang direpresentasikan oleh variabel *eta* pada baris logika pembaruan di *pseudocode* yang terletak pada Kode 2.1.

```
ALGORITHM: Support_Vector_Machine(D, C, K)

INPUT:
    - D = {(x_i, y_i)} untuk i = 1,...,N (Dataset Pelatihan)
    - C: Parameter penalti (Regularisasi)
    - K(x, x'): Fungsi Kernel

OUTPUT:
    - f(x): Fungsi keputusan klasifikasi

BEGIN
    // 1. Penyelesaian Masalah Optimasi (Dual Form)
    Dapatkan vektor Lagrange multipliers α dengan memaksimalkan:
    max_α Σ_{i=1}^{N} α_i − (1/2) Σ_{i=1}^{N} Σ_{j=1}^{N} α_i α_j y_i y_j K(x_i, x_j)

    Tunduk pada kendala:
    0 ≤ α_i ≤ C untuk setiap i = 1,...,N
    Σ_{i=1}^{N} α_i y_i = 0

    // 2. Identifikasi Support Vectors (SV)
```

```
    SV ← {(x_i, y_i) ∈ D | α_i > 0}

    // 3. Perhitungan nilai Bias (b)
    Pilih sembarang data (x_k, y_k) ∈ SV di mana 0 < α_k < C
    b ← y_k − Σ_{i=1}^{N} α_i y_i K(x_i, x_k)

    // 4. Pembentukan Fungsi Keputusan
    f(x) ← sign(Σ_{i=1}^{N} α_i y_i K(x_i, x) + b)

    RETURN f(x)
END
```

Kode 2.1: *Pseudocode* Algoritma Pelatihan *Support Vector Machine*

Sumber: [39]

Pemilihan fungsi *kernel* sangat bergantung pada karakteristik data yang diolah. Sebagai contoh, *Kernel Linear* digunakan pada *dataset* yang dapat dipisahkan secara linear. Rumus *Kernel Linear* dinyatakan dalam Rumus 2.132 sebagai berikut.

$$K(x,x') = \langle x,x' \rangle \tag{2.132}$$

Untuk menangkap interaksi antarfitur hingga derajat tertentu (d), seperti pada pemrosesan citra, *Kernel Polynomial* sering digunakan. Rumus *Kernel Polynomial* dinyatakan dalam Rumus 2.133 sebagai berikut.

$$K(x,x') = (\gamma\langle x,x' \rangle + r)^d \tag{2.133}$$

Di sisi lain, *kernel* yang paling umum digunakan adalah *Radial Basis Function* (*RBF*), yang mampu memetakan data ke ruang berdimensi tinggi berdasarkan jarak *Euclidean*. Rumus *Kernel RBF* dinyatakan dalam Rumus 2.134 sebagai berikut.

$$K(x,x') = \exp(-\gamma|x-x'|^2) \tag{2.134}$$

Selain itu, terdapat *Kernel Sigmoid* yang diadaptasi dari teori jaringan saraf tiruan dan berperan menyerupai fungsi aktivasi perseptron. Rumus *Kernel Sigmoid* dinyatakan dalam Rumus 2.135 sebagai berikut.

$$K(x,x') = \tanh(k_1 \langle x,x' \rangle + k_2) \tag{2.135}$$

Efisiensi komputasi *Support Vector Machine* (*SVM*) dengan berbagai fungsi *kernel* tersebut didukung oleh penggunaan Matriks Gram yang menyimpan hasil perhitungan *kernel* antar pasangan titik data. Matriks ini mempermudah proses optimasi kuadratik selama pelatihan model. Secara umum, kompleksitas waktu algoritma *SVM* berada pada kisaran $O(n^2)$ hingga $O(n^3)$, bergantung pada implementasi dan fungsi *kernel* yang digunakan. Karakteristik tersebut menjadikan *SVM* sebagai metode yang efektif untuk menangani data berdimensi tinggi. Namun demikian, penerapannya pada *dataset* berskala besar memerlukan optimasi komputasi yang memadai.

Secara fundamental, formulasi asli *Support Vector Machine* dirancang untuk menyelesaikan permasalahan klasifikasi biner. Namun, pada kasus yang melibatkan lebih dari dua kelas ($K > 2$), diperlukan suatu skema dekomposisi untuk mentransformasikan permasalahan multikelas menjadi serangkaian permasalahan klasifikasi biner. Salah satu pendekatan yang paling banyak digunakan karena efisiensi komputasinya adalah strategi *One-vs-Rest* (*OvR*) atau *One-vs-All* (*OvA*).

Pada strategi *OvR*, sebanyak K pengklasifikasi biner independen dilatih, di mana setiap pengklasifikasi ke-k ($k \in 1,\ldots,K$) bertugas membedakan kelas ke-k sebagai sampel positif terhadap gabungan seluruh kelas lainnya ($K-1$ kelas) sebagai sampel negatif. Secara matematis, untuk setiap pengklasifikasi f_k, proses optimasi dilakukan untuk memperoleh *hyperplane* yang memisahkan kelas C_k dari kelas lainnya sehingga menghasilkan fungsi keputusan dinyatakan dalam Rumus 2.136 sebagai berikut.

$$f_k(x) = w_k^T \phi(x) + b_k \tag{2.136}$$

Pada tahap inferensi, sampel data baru x dievaluasi menggunakan seluruh K model biner yang telah dilatih. Kelas prediksi ditentukan berdasarkan prinsip *winner-takes-all*, yaitu dengan memilih kelas yang menghasilkan nilai fungsi keputusan tertinggi. Pendekatan ini memastikan bahwa kelas yang dipilih memiliki tingkat keyakinan atau *margin* terbesar dibandingkan kelas lainnya. Secara matematis, proses tersebut dinyatakan dalam Rumus 2.137 sebagai berikut.

$$\hat{y} = \arg \max_{k \in 1,\ldots,K} (f_k(x)) \tag{2.137}$$

Pendekatan *OvR* memiliki keunggulan dalam mengurangi beban komputasi dibandingkan metode *One-vs-One* (*OvO*), terutama pada *dataset* dengan jumlah kelas yang besar. Namun demikian, strategi ini rentan terhadap permasalahan ketidakseimbangan kelas (*class imbalance*) karena jumlah sampel negatif yang berasal dari gabungan kelas lain cenderung lebih besar dibandingkan jumlah sampel positif. Oleh karena itu, pengaturan parameter regularisasi C serta pemilihan fungsi *kernel* yang tepat menjadi faktor penting untuk menjaga kualitas batas keputusan pada setiap pengklasifikasi biner.

2.5 *Ensemble Learning*

Ensemble learning merupakan paradigma statistika dan pembelajaran mesin yang mengintegrasikan sejumlah algoritma pembelajaran individual (*base learner*) untuk membangun model prediktif komposit dengan tingkat akurasi dan stabilitas yang lebih tinggi dibandingkan masing-masing algoritma penyusunnya. Landasan matematis utama pendekatan ini adalah mitigasi dilema *bias-variance trade-off*. Secara arsitektural, metode *ensemble* dapat dikelompokkan menjadi dua kategori utama, yaitu metode paralel yang dirancang untuk menurunkan variansi, seperti *Bagging*, dan metode sekuensial yang dirancang untuk mengurangi bias, seperti *Boosting*.

2.5.1 *Bagging*

Bagging (*Bootstrap Aggregating*) merupakan metode *ensemble* paralel homogen yang bertujuan mengurangi variansi model tanpa meningkatkan bias secara signifikan. Proses operasionalnya terdiri atas tiga tahapan, yaitu: (1) menghasilkan sejumlah subset data latih melalui proses *bootstrapping* (pengambilan sampel acak dengan pengembalian), (2) melatih setiap model dasar, umumnya berupa pohon keputusan berkedalaman penuh, secara paralel pada masing-masing subset data, dan (3) menggabungkan hasil prediksi seluruh model. Pada permasalahan klasifikasi, proses agregasi dilakukan menggunakan mekanisme suara mayoritas (*hard voting*), sedangkan pada permasalahan regresi dilakukan melalui rata-rata prediksi (*soft voting*). Fungsi agregasi pada metode *Bagging*

dirumuskan sebagai berikut:

$$F_{bag}(\mathbf{x}') = \frac{1}{M}\sum_{m=1}^{M} b_m(\mathbf{x}') \tag{2.138}$$

2.5.2 *Boosting*

Boosting merupakan metode *ensemble* sekuensial yang bertujuan mengurangi bias dengan menggabungkan sejumlah *weak learner* menjadi sebuah *strong learner*. Pada setiap iterasi, model baru dilatih untuk memperbaiki kesalahan prediksi yang dihasilkan oleh model sebelumnya. Dalam pendekatan *Gradient Boosting*, proses optimasi dilakukan menggunakan metode *Gradient Descent*. Pembaruan fungsi prediksi dilakukan secara aditif melalui penambahan model baru yang dikalikan dengan parameter laju pembelajaran (*learning rate*) ν. Proses tersebut dinyatakan dalam Rumus 2.139 sebagai berikut.

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \nu \cdot \gamma_m h_m(\mathbf{x}) \tag{2.139}$$

2.6 *Stacking Ensemble*

Stacked Generalization atau *Stacking Ensemble* adalah arsitektur ansambel heterogen yang beroperasi melalui skema algoritmik meta-pembelajaran hierarkis untuk mendeduksi dan mengoreksi bias dari kumpulan model prediktif dasar (*base learners*) terhadap suatu himpunan data. Berbeda secara fundamental dengan metode ansambel homogen konvensional seperti *bagging* atau *boosting*, *stacking* didesain secara intrinsik untuk mengakomodasi heterogenitas fungsional. Hal ini memungkinkan integrasi berbagai kelas algoritma prediktif yang sama sekali berbeda—seperti jaringan saraf tiruan, mesin vektor pendukung (SVM), dan ansambel pohon keputusan—ke dalam satu arsitektur terpadu. Secara konseptual, *meta-learner* (model tingkat tinggi) digunakan untuk menggabungkan prediksi dari *base learners* guna mencapai tingkat akurasi prediktif yang melampaui kapasitas masing-masing model konstituennya. Secara matematis, asalkan kerugian lintas-validasi diminimalkan, *meta-learner* akan secara asimtotik mempelajari kombinasi prediksi yang optimal dari model-model dasar, menjamin bahwa performa ansambel

berkonvergensi ke tingkat yang setara atau secara empiris lebih baik daripada anggota terbaik di dalam ansambel tersebut.

Dalam terminologi formal arsitektur *stacking*, ruang fungsional dibagi menjadi dua level hierarkis komputasi utama: Level-0 dan Level-1. Himpunan data observasi awal beserta model prediktif dasar yang dilatih untuk mengekstraksi pola diklasifikasikan sebagai data Level-0 dan model Level-0. Mengingat model *ensemble* heterogen ini tidak dapat dideferensiasi secara mulus (*non-differentiable*), optimalisasi bobot lintas arsitektur tidak dapat dilakukan menggunakan prosedur kalkulus standar seperti metode penurunan gradien (*gradient descent*). Sebagai gantinya, jaringan ini harus dibangun secara bertahap dan iteratif. Untuk memitigasi kebocoran informasi (*data leakage*)—di mana model menghafal data pelatihannya sendiri—proses pemindahan representasi dari Level-0 ke Level-1 mewajibkan partisi himpunan data menjadi subset-subset yang saling terpisah secara mutlak melalui mekanisme validasi silang (*cross-validation*).

Misalkan terdapat himpunan data latih awal (Level-0) $\mathscr{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$, di mana $\mathbf{x}_n$ adalah vektor fitur dan y_n adalah label target. Pada Level-0, sekumpulan algoritma dasar berjumlah K dilatih dan direpresentasikan sebagai $h_1, h_2, \ldots, h_K$. Setiap model Level-0 dilatih pada subset pelatihan dan dipaksa untuk memprediksi subset validasi (*holdout sets*). Prediksi yang dihasilkan oleh model ke-k untuk observasi ke-n didefinisikan sebagai $p_{nk} = h_k(\mathbf{x}_n)$. Tindakan merekonstruksi ruang dimensi baru dari hasil prediksi validasi silang ini secara teknis sering diistilahkan sebagai *blending*.

Setelah prediksi dari setiap model Level-0 terkumpul untuk seluruh instans dalam himpunan data, ruang matriks prediksi tersebut disatukan dengan vektor target asli untuk membentuk data Level-1. Matriks *meta*-fitur $\mathbf{Z}$ yang merepresentasikan data Level-1 memiliki dimensi $N \times K$, di mana baris ke-n adalah vektor prediksi $\mathbf{z}_n = (p_{n1}, p_{n2}, \ldots, p_{nK})$. Himpunan data Level-1 kemudian dapat diformulasikan sebagai $\mathscr{D}_{\text{meta}} = \{(\mathbf{z}_n, y_n)\}_{n=1}^{N}$.

Matriks *meta*-fitur ini dibentuk dari probabilitas prediksi yang dihasilkan oleh setiap *base learner*. Vektor *meta*-fitur untuk setiap observasi dapat dinyatakan pada Rumus 2.140.

$$\mathbf{z}_n = [p_{n1}, p_{n2}, p_{n3}] \tag{2.140}$$

dengan p_{n1}, p_{n2}, dan p_{n3} berturut-turut merupakan probabilitas prediksi

yang dihasilkan oleh model *base learner*.

Pada implementasinya, model perlu mempelajari kontribusi masing-masing probabilitas prediksi dari *base learner* melalui kombinasi linear berbobot. Untuk tiga *base learner*, skor linear yang dihasilkan dapat dinyatakan pada Rumus 2.141.

$$s = \beta_0 + \beta_1 p_{n1} + \beta_2 p_{n2} + \beta_3 p_{n3} \tag{2.141}$$

Skor linear yang dihasilkan oleh *meta-learner* kemudian ditransformasikan menjadi probabilitas kelas menggunakan fungsi *Softmax*. Fungsi ini mengubah skor untuk setiap kelas menjadi distribusi probabilitas yang bernilai antara 0 dan 1, dengan total probabilitas seluruh kelas sama dengan satu. Untuk masalah klasifikasi multikelas dengan C kelas, probabilitas suatu observasi termasuk ke kelas c dinyatakan pada Rumus 2.142.

$$P(y = c|\mathbf{z}_n) = \frac{\exp(s_c)}{\sum_{j=1}^{C} \exp(s_j)} \tag{2.142}$$

dengan $P(y = c|\mathbf{z}_n)$ menyatakan probabilitas observasi ke-n termasuk ke kelas c, s_c merupakan skor linear untuk kelas ke-c, dan C adalah jumlah kelas target. Kelas dengan probabilitas tertinggi kemudian dipilih sebagai prediksi akhir model *Stacking Ensemble*.

2.7 *Random Forest* (RF)

Random Forest (RF) merupakan pengembangan lanjutan dari pohon keputusan berbasis *Bagging* yang menerapkan mekanisme keacakan tambahan pada pemilihan fitur untuk meningkatkan keragaman model dan mengurangi variansi prediksi. Selain menggunakan sampel hasil *bootstrapping* untuk membangun sejumlah pohon keputusan sebanyak B, algoritma ini juga menerapkan *Random Subspace Method* atau *Feature Bagging*. Pada setiap proses pemisahan simpul (*node split*), hanya sebagian fitur yang dipilih secara acak sebanyak m_{try} dari total fitur sebanyak p, dengan ketentuan $m_{try} \ll p$. Pada kasus klasifikasi, nilai m_{try} umumnya ditentukan sebesar $\sqrt{p}$.

Pembatasan jumlah fitur yang dievaluasi pada setiap simpul bertujuan untuk mengurangi dominasi fitur tertentu dan menurunkan korelasi antar pohon keputusan. Dengan demikian, kemampuan generalisasi model dapat ditingkatkan.

Untuk kasus regresi, prediksi akhir diperoleh melalui rata-rata keluaran seluruh pohon keputusan yang dinyatakan dalam Rumus 2.143 sebagai berikut.

$$\hat{f}_{RF}(\mathbf{x}') = \frac{1}{B}\sum_{b=1}^{B} h_b(\mathbf{x}') \tag{2.143}$$

Dalam implementasi penelitian ini, arsitektur Random Forest tidak berdiri sendiri, melainkan diintegrasikan ke dalam sebuah *pipeline* komputasi yang ketat. *Pipeline* ini merangkai proses standarisasi (*Z-score*), penyeimbangan kelas (SMOTE-Tomek), seleksi fitur berpenalti L1 (LASSO), hingga klasifikasi akhir menggunakan Random Forest. Alur operasional dari proses pencarian parameter terbaik hingga tahap prediksi diekstraksi ke dalam abstraksi pada *pseudocode* yang terdapat di Kode 2.2.

```
ALGORITHM: Random_Forest(D, T, m)

INPUT:
    - D = {(x_i, y_i)} untuk i = 1,...,N (Dataset Pelatihan)
    - T: Jumlah pohon keputusan dalam ansambel
    - m: Jumlah sub-fitur yang dievaluasi pada setiap split node

OUTPUT:
    - f(x): Fungsi prediksi klasifikasi ansambel

BEGIN
    FOR t = 1 TO T DO
        // 1. Pengambilan sampel bootstrap
        D_t ← Ambil N sampel secara acak dari D dengan pengembalian
    (with replacement)

        // 2. Pembangunan pohon keputusan tunggal
        Inisialisasi pohon h_t dengan root node berisi D_t

        REPEAT (secara rekursif untuk setiap node yang belum
    dipisah):
            Pilih m fitur secara acak dari total ruang fitur
            Temukan fitur dan titik batas (split) terbaik di
    antara m fitur untuk meminimalkan ketidakmurnian (Gini/Entropy)
            Pisahkan node menjadi dua child node
        UNTIL kriteria henti terpenuhi (semua node murni)

        Simpan pohon h_t
```

```
    END FOR

    // 3. Pembentukan keputusan ansambel (Majority Vote)
    f(x) ← arg max_y Σ_{t=1}^{T} I(h_t(x) = y)

    // (I adalah fungsi indikator, bernilai 1 jika benar, 0 jika salah)

    RETURN f(x)
END
```

Kode 2.2: *Pseudocode* Algoritma Pelatihan *Random Forest*

Sumber: [40]

2.8 *Extreme Gradient Boosting* (XGBoost)

XGBoost adalah representasi kerangka keilmuan mutakhir yang mengoptimalkan fungsi *Gradient Boosted Decision Trees* (GBDT) tingkat lanjut untuk memproses penskalaan komputasi paralel dan menangani ketahanan matriks *overfitting* yang sangat superior. Algoritma ini secara inovatif mereformulasi determinasi fungsi objektif model di setiap iterasi (t) dengan menyisipkan penalti regularisasi fungsional kompleksitas analitik tingkat pohon menggunakan Rumus 2.144 sebagai berikut.

$$\text{obj}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \Omega(f_t) \tag{2.144}$$

Dalam rumusan tersebut, terdapat variabel pengekang $\Omega(f_t)$ yang secara imperatif mengontrol dan membatasi kuantitas perluasan titik daun (T) dan bobot penyusutan L_2-norm asimtotik prediksi daun (w_j). Variabel pengekang ini didefinisikan dengan Rumus 2.145 berikut.

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{2.145}$$

Lebih lanjut, XGBoost mengadopsi operasi perluasan fungsi deret kalkulus Taylor orde kedua untuk mengoptimalkan presisi penemuan langkah konvergensi ruang fungsi kerugian terpendek. Dalam implementasi penelitian ini, algoritma XGBoost dikonfigurasi sebagai salah satu model dasar (*base learner*) yang melalui proses optimasi sistematis dalam sebuah *pipeline* terintegrasi. Prosedur

komputasi yang mencakup tahap pra-pemrosesan, seleksi fitur LASSO, hingga pencarian konfigurasi *hyperparameter* optimal divisualisasikan melalui abstraksi pada *pseudocode* yang terletak di Kode 2.3.

```
ALGORITHM: XGBoost(D, T, η, λ, γ)

INPUT:
    - D = {(x_i, y_i)} untuk i = 1,...,N (Dataset Pelatihan)
    - T: Jumlah maksimum iterasi (jumlah pohon)
    - η: Learning rate (Tingkat pembelajaran)
    - λ: Parameter regularisasi L2 (Penalti bobot daun)
    - γ: Penalti kompleksitas pohon (Syarat minimum gain untuk split)
    - L(y, ŷ): Fungsi kerugian (Loss function) yang dapat diturunkan

OUTPUT:
    - F_T(x): Fungsi prediksi ansambel final

BEGIN
    // 1. Inisialisasi model dengan nilai konstan (misal: probabilitas awal)
    F_0(x) ← argmin_c Σ_{i=1}^{N} L(y_i, c)

    FOR t = 1 TO T DO
        // 2. Hitung Gradient dan Hessian untuk setiap sampel
        FOR i = 1 TO N DO
            g_i ← ∂L(y_i, F_{t-1}(x_i)) / ∂F_{t-1}(x_i)    // Turunan pertama
            h_i ← ∂²L(y_i, F_{t-1}(x_i)) / ∂F_{t-1}(x_i)²  // Turunan kedua
        END FOR

        // 3. Bangun pohon regresi f_t(x) untuk meminimalkan fungsi objektif
        Inisialisasi pohon dengan node akar yang memuat seluruh dataset

        REPEAT (untuk memecah node):
            Evaluasi setiap kemungkinan split fitur dengan menghitung Gain:
            Gain = 1/2 [ (Σ_L g_i)²/(Σ_L h_i+λ) + (Σ_R g_i)²/(Σ_R h_i+λ) - (Σ g_i)²/(Σ h_i+λ) ] - γ

            Pilih split dengan Gain terbesar.
            JIKA Gain < 0, hentikan percabangan (Pruning).
```

```
        UNTIL kedalaman maksimum tercapai atau kondisi henti terpenuhi

        // 4. Hitung bobot optimal w_j untuk setiap daun j pada pohon f_t(x)
        w_j ← −(Σ_{i∈I_j} g_i) / (Σ_{i∈I_j} h_i + λ)

        // 5. Perbarui model ansambel dengan penambahan learning rate
        F_t(x) ← F_{t−1}(x) + η f_t(x)
    END FOR

    RETURN F_T(x)
END
```

Kode 2.3: *Pseudocode* Algoritma Pelatihan *eXtreme Gradient Boosting* (XGBoost)
Sumber: [41]

2.9 Validasi Silang Bersarang (*Nested Cross-Validation*)

Validasi silang (*Cross-Validation* / CV) konvensional, seperti *k-fold cross-validation*, merupakan prosedur standar yang lazim digunakan untuk mengevaluasi kapabilitas generalisasi suatu algoritma pembelajaran mesin. Namun, literatur analitik prediktif modern secara tegas menyoroti kelemahan fundamental dari CV standar apabila prosedur tersebut digunakan secara bersamaan untuk dua tujuan sekaligus: melatih model dan melakukan optimasi parameter (*hyperparameter tuning*). Ketika sebuah model berulang kali dievaluasi pada himpunan data validasi yang sama untuk mencari konfigurasi parameter terbaik, informasi dari data validasi tersebut secara perlahan akan ”bocor” ke dalam model. Fenomena kebocoran data (*data leakage*) ini memicu terjadinya *overfitting* pada ruang parameter, yang berakibat pada estimasi performa akhir yang sangat bias dan terlalu optimis (*optimistic bias*).

Untuk menanggulangi bias estimasi tersebut, arsitektur Validasi Silang Bersarang (*Nested Cross-Validation*) diimplementasikan sebagai standar emas (*gold standard*) dalam evaluasi model, terutama pada *dataset* klinis berdimensi tinggi namun dengan ukuran sampel yang terbatas. Secara konseptual, *Nested Cross-Validation* memisahkan fase seleksi model (optimasi parameter) dan fase evaluasi performa generalisasi ke dalam dua lingkar iterasi yang saling independen: Lingkar Dalam (*Inner Loop*) dan Lingkar Luar (*Outer Loop*).

1. Lingkar Luar (*Outer Loop*): Fase ini bertugas secara eksklusif untuk mengevaluasi seberapa baik performa model pada data yang benar-benar

belum pernah dilihat. *Dataset* awal dipartisi menjadi himpunan latih luar (*outer training set*) dan himpunan uji luar (*outer testing set*). Himpunan uji luar ini diisolasi secara mutlak dan dilarang keras untuk digunakan dalam proses penyesuaian parameter apa pun.

2. Lingkar Dalam (*Inner Loop*): Fase ini beroperasi murni sebagai arena optimasi internal. Menggunakan himpunan latih luar dari *Outer Loop*, data dipartisi kembali menjadi himpunan latih dalam (*inner training set*) dan validasi dalam (*inner validation set*). Proses komputasi iteratif, seperti pencarian acak (*GridSearchCV*), dieksekusi sepenuhnya di dalam ruang lingkup ini untuk memetakan kombinasi *hyperparameter* terbaik berdasarkan metrik objektif tertentu (misalnya *Macro Recall*).

Setelah algoritma mencapai konvergensi dan menemukan parameter paling optimal di *Inner Loop*, model dengan konfigurasi tersebut akan dilatih ulang menggunakan keseluruhan himpunan latih luar. Model final ini kemudian dievaluasi menggunakan himpunan uji luar dari *Outer Loop* yang sedari awal telah diisolasi.

Pemisahan struktural ini secara matematis menjamin bahwa data uji yang digunakan untuk melaporkan metrik performa akhir tidak memiliki kontribusi sedikit pun terhadap keputusan pemilihan arsitektur model maupun parameternya. Dalam konteks pemodelan radiomik dan prediksi *N-Stage*, penggunaan *Nested Cross-Validation* sangat krusial untuk menghasilkan taksiran akurasi diagnostik yang objektif, kokoh (*robust*), dan dapat dipertanggungjawabkan saat diimplementasikan pada skenario klinis dunia nyata.

2.10 *Tuning* Hiperparameter (*Hyperparameter Tuning*)

Hyperparameter tuning atau *tuning* hiperparameter adalah proses pencarian kombinasi nilai optimal untuk parameter-parameter yang tidak dipelajari secara otomatis oleh model selama proses pelatihan (pelatihan parameter internal melalui *weight* atau *bias*). Berbeda dengan parameter model, hiperparameter ditentukan sebelum proses pelatihan dimulai dan sangat menentukan arsitektur serta kemampuan generalisasi model terhadap data baru. Tujuan utama dari proses ini adalah untuk meminimalkan fungsi kerugian (*loss function*) dan mengendalikan kompleksitas model guna menghindari fenomena *overfitting* maupun *underfitting*.

Dalam penelitian ini, optimasi dilakukan menggunakan metode *Grid Search* yang dikombinasikan dengan prosedur *Nested Cross-Validation*. Berikut adalah

beberapa pendekatan utama dalam optimasi hiperparameter:

1. *Grid Search*

   Grid Search adalah metode pencarian sistematis yang mengevaluasi seluruh kombinasi yang mungkin dari sekumpulan nilai hiperparameter yang telah ditentukan sebelumnya (ruang pencarian diskrit). Meskipun memerlukan biaya komputasi yang tinggi seiring bertambahnya jumlah hiperparameter, metode ini menjamin ditemukannya kombinasi terbaik di dalam ruang pencarian yang diberikan. Keunggulan utamanya adalah kemampuannya untuk dijalankan secara paralel.

2. *Random Search*

   Berbeda dengan *Grid Search*, *Random Search* memilih kombinasi hiperparameter secara acak dari distribusi ruang pencarian tertentu. Metode ini sering kali lebih efisien daripada *Grid Search* dalam menemukan kombinasi yang memadai dengan waktu komputasi yang lebih singkat, terutama ketika hanya beberapa hiperparameter yang memiliki pengaruh signifikan terhadap performa model.

2.11 Evaluasi Metrik

Untuk mengevaluasi kinerja model *Stacking Ensemble* dalam memprediksi stadium kanker (*N-Stage*) yang terdiri atas tiga kelas, digunakan beberapa metrik evaluasi. Penggunaan metrik makro (*macro-averaged*) bertujuan untuk memastikan bahwa setiap kelas memperoleh bobot yang setara, mengingat adanya potensi ketidakseimbangan distribusi data klinis. Berikut merupakan penjelasan mengenai metrik evaluasi yang digunakan:

1. Akurasi (*Accuracy*)

   Akurasi merepresentasikan proporsi keseluruhan sampel yang berhasil diklasifikasikan dengan benar terhadap jumlah total sampel dalam *dataset*. Meskipun memberikan gambaran performa model secara global, akurasi perlu diinterpretasikan secara hati-hati pada data klinis yang tidak seimbang karena cenderung dipengaruhi oleh kelas mayoritas. Akurasi dihitung menggunakan Rumus 2.146 berikut.

$$\text{Akurasi} = \frac{\sum_{c=1}^{C} TP_c}{N} \tag{2.146}$$

dengan $\sum_{c=1}^{C} TP_c$ menyatakan jumlah prediksi yang benar pada seluruh kelas dan N menyatakan jumlah total sampel.

2. Presisi (*Precision*) Makro

   Presisi atau nilai prediktif positif mengukur proporsi sampel yang diprediksi positif secara benar dibandingkan dengan seluruh prediksi positif pada kelas tertentu. Nilai presisi yang tinggi menunjukkan rendahnya tingkat *false positive*. Untuk kelas ke-c, presisi dijabarkan dalam Rumus 2.147 berikut.

$$\text{Presisi}_c = \frac{TP_c}{TP_c + FP_c} \tag{2.147}$$

   dengan TP_c menyatakan *true positive* dan FP_c menyatakan *false positive* pada kelas ke-c. Nilai *Macro-Precision* diperoleh melalui rata-rata tidak tertimbang dari seluruh C kelas yang dijabarkan dalam Rumus 2.148 berikut.

$$\text{Presisi}_{macro} = \frac{1}{C} \sum_{c=1}^{C} \text{Presisi}_c \tag{2.148}$$

3. Recall / Sensitivitas (*Macro Recall*)

   Recall mengukur proporsi sampel positif aktual yang berhasil diidentifikasi secara benar oleh model. Dalam konteks klasifikasi stadium tumor, metrik ini mencerminkan kemampuan model dalam mengidentifikasi seluruh sampel yang benar-benar termasuk ke dalam suatu kategori *N-Stage* tertentu serta meminimalkan kesalahan *false negative*. Nilai *Recall* pada kelas ke-c dijabarkan dalam Rumus 2.149 berikut.

$$Recall_c = \frac{TP_c}{TP_c + FN_c} \tag{2.149}$$

   dengan FN_c menyatakan *false negative*. Nilai *Macro-Recall* diperoleh melalui rata-rata tidak tertimbang dari seluruh kelas yang dijabarkan dalam Rumus 2.150 berikut.

$$Recall_{macro} = \frac{1}{C}\sum_{c=1}^{C} Recall_c \tag{2.150}$$

4. *F1-Score* Makro

   F1-Score merupakan rata-rata harmonik antara presisi dan *recall*. Metrik ini sangat berguna untuk menilai keseimbangan antara kesalahan *false positive* dan *false negative*, terutama pada *dataset* yang tidak seimbang, di mana akurasi saja dapat memberikan interpretasi yang menyesatkan. Nilai *F1-Score* untuk kelas ke-c dijabarkan dalam Rumus 2.151 berikut.

$$F1_c = 2 \times \frac{\text{Presisi}_c \times Recall_c}{\text{Presisi}_c + Recall_c} \tag{2.151}$$

   Selanjutnya, nilai *Macro F1-Score* dihitung sebagai rata-rata dari seluruh nilai *F1-Score* pada masing-masing kelas, seperti yang dijabarkan dalam Rumus 2.152 berikut.

$$F1_{macro} = \frac{1}{C}\sum_{c=1}^{C} F1_c \tag{2.152}$$

5. *Receiver Operating Characteristic – Area Under the Curve* (ROC AUC)

   Kurva ROC menggambarkan kemampuan diskriminatif model klasifikasi dengan memetakan hubungan antara *True Positive Rate* (TPR) dan *False Positive Rate* (FPR) pada berbagai nilai ambang (*threshold*). Untuk mengadaptasi metrik yang pada dasarnya bersifat biner ke dalam klasifikasi *N-Stage* tiga kelas, digunakan pendekatan *One-vs-Rest* (OvR). Pada pendekatan ini, satu kelas diperlakukan sebagai kelas positif, sedangkan seluruh kelas lainnya digabungkan menjadi kelas negatif.

   Nilai AUC merepresentasikan luas area di bawah kurva ROC dari titik $(0,0)$ hingga $(1,1)$. Nilai *Macro ROC AUC* dihitung sebagai rata-rata tidak tertimbang dari nilai AUC seluruh kelas seperti yang dijabarkan dalam Rumus 2.153 berikut.

$$\text{ROC AUC}_{macro} = \frac{1}{C}\sum_{c=1}^{C} \text{AUC}(c, rest) \tag{2.153}$$

Nilai AUC sebesar 1,0 menunjukkan kemampuan diskriminasi yang sempurna, sedangkan nilai 0,5 menunjukkan performa yang setara dengan klasifikasi acak.

6. *Precision-Recall Area Under the Curve* (PR AUC)

   Berbeda dengan ROC AUC yang memasukkan *true negative* ke dalam perhitungan *false positive rate*, kurva *Precision-Recall* berfokus pada hubungan antara presisi dan *recall*. Oleh karena itu, metrik ini sangat informatif untuk *dataset* medis yang tidak seimbang karena lebih sensitif terhadap performa pada kelas minoritas.

   Nilai PR AUC dihitung sebagai luas area di bawah kurva *Precision-Recall*, yang dapat didekati melalui integral dalam Rumus 2.154 berikut.

$$\text{PR AUC}_c = \int_0^1 \text{Presisi}_c(R)\, dR \tag{2.154}$$

   Nilai *Macro PR AUC* diperoleh melalui rata-rata aritmetika tidak tertimbang dari seluruh nilai PR AUC pada masing-masing kelas.

7. Estimasi Ketidakpastian dan Interval Kepercayaan 95% (*95% Confidence Interval*)

   Dalam pemodelan prediksi klinis, penggunaan estimasi titik (*point estimate*) saja tidak cukup untuk menggambarkan keandalan model secara menyeluruh. Oleh karena itu, untuk memperhitungkan variasi yang muncul akibat ukuran sampel yang terbatas dan variasi pembagian data selama proses *Cross-Validation*, dihitung interval kepercayaan 95% (*95% Confidence Interval* atau 95% CI) untuk setiap metrik evaluasi.

   Nilai 95% CI merepresentasikan rentang nilai yang secara statistik memiliki tingkat kepercayaan sebesar 95% untuk mencakup performa sebenarnya dari model apabila diterapkan pada populasi yang belum pernah diamati sebelumnya (*unseen population*). Interval tersebut dihitung menggunakan Rumus 2.155 berikut.

$$\mathrm{CI}_{95\%} = \bar{x} \pm 1.96 \times \left(\frac{s}{\sqrt{n}} \right) \tag{2.155}$$

dengan:

- $\bar{x}$ merupakan nilai rata-rata (*mean*) metrik performa yang diperoleh dari seluruh iterasi evaluasi.
- s merupakan simpangan baku (*standard deviation*) dari metrik performa.
- n merupakan jumlah total pengujian atau *fold*.
- 1.96 merupakan nilai kritis distribusi normal standar untuk tingkat kepercayaan sebesar 95

Pelaporan metrik beserta interval kepercayaan 95% sangat penting dalam penelitian klinis karena mampu mengkuantifikasi stabilitas dan kemampuan generalisasi model. Interval kepercayaan yang sempit menunjukkan bahwa performa model relatif konsisten terhadap variasi sampel data, sedangkan interval yang lebar mengindikasikan adanya variabilitas yang tinggi dan potensi *overfitting*. Dengan demikian, estimasi ketidakpastian memberikan dasar yang lebih kuat dalam mengevaluasi keandalan model *Stacking Ensemble* untuk mendukung pengambilan keputusan klinis.

2.12 *Explainable Artificial Intelligence* (XAI)

Implementasi algoritma pembelajaran mesin yang kompleks, khususnya model *Stacking Ensemble*, sering kali menghadapi tantangan *black-box*, yaitu kondisi ketika mekanisme internal model dalam menghasilkan prediksi sulit dipahami secara langsung. Dalam domain klinis, karakteristik *black-box* tersebut menjadi hambatan penting karena pengambilan keputusan medis memerlukan tingkat transparansi dan akuntabilitas yang tinggi. *Explainable Artificial Intelligence* (XAI) dikembangkan untuk menjembatani kesenjangan antara performa prediktif yang tinggi dan kebutuhan akan interpretabilitas, sehingga faktor-faktor yang mendasari prediksi stadium tumor (*N-Stage*) dapat dipahami secara lebih komprehensif.

Salah satu metode XAI yang paling banyak digunakan dalam penelitian radiomik adalah SHAP (*SHapley Additive exPlanations*). Metode ini berlandaskan

pada konsep *cooperative game theory* yang diperkenalkan oleh Lloyd Shapley. Dalam kerangka tersebut, setiap fitur radiomik diperlakukan sebagai pemain dalam suatu permainan kooperatif, sedangkan hasil prediksi model dipandang sebagai imbalan yang harus didistribusikan secara adil kepada setiap fitur berdasarkan kontribusi marginalnya.

Keunggulan utama SHAP terletak pada kemampuannya dalam menghasilkan interpretasi lokal maupun global yang konsisten. Secara matematis, kontribusi setiap fitur dihitung dengan membandingkan prediksi model yang menggunakan fitur tersebut dengan prediksi model tanpa fitur tersebut pada berbagai kombinasi fitur lainnya. Proses ini menghasilkan nilai SHAP (*SHAP values*) yang bersifat aditif dan dapat direpresentasikan melalui model penjelasan linear dalam Rumus 2.156 berikut.

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j \tag{2.156}$$

dengan g menyatakan model penjelasan, ϕ_0 menyatakan nilai dasar (*base rate*) atau rata-rata prediksi model, dan ϕ_j menyatakan kontribusi fitur ke-j yang direpresentasikan sebagai *Shapley value*.

Dalam analisis radiomik untuk klasifikasi *N-Stage*, SHAP memungkinkan identifikasi fitur-fitur tekstur maupun bentuk yang memiliki pengaruh paling besar terhadap hasil prediksi. Melalui visualisasi seperti *summary bar plot* dan *beeswarm plot*, SHAP tidak hanya mengidentifikasi fitur yang paling dominan, tetapi juga menjelaskan arah pengaruh masing-masing fitur terhadap prediksi model. Dengan demikian, peningkatan atau penurunan nilai suatu fitur dapat dihubungkan dengan perubahan probabilitas suatu stadium tumor secara lebih objektif. Oleh karena itu, penggunaan SHAP memberikan lapisan interpretasi tambahan yang membantu menyelaraskan temuan komputasional dengan pemahaman patologis dalam bidang onkologi.