BAB II
KAJIAN LITERATUR

2.1 Dinamika Bersepeda dan Sensor Inersia

2.1.1 Risiko Jatuh dan Gerak Pesepeda

Sepeda merupakan sistem benda tegar *(rigid body)* dengan titik keseimbangan dinamis yang secara alami tidak stabil dalam kondisi diam. Dari perspektif biomekanika, seorang pengendara berisiko jatuh apabila gagal melakukan pemulihan kontrol akibat adanya gangguan pada stang *(handlebar disturbance)* [24]. Untuk mempertahankan keseimbangan, posisi *Center of Mass* (CoM) harus tetap didukung oleh area tumpuan roda *(Base of Support)*. Apabila pesepeda gagal melakukan pemulihan keseimbangan, maka risiko jatuh akan meningkat [25].

Dalam konteks pembacaan sensor inersia, hilangnya stabilitas ini dapat diidentifikasi melalui adanya deviasi pada variabilitas gerakan *(movement variability)*. Penelitian mengungkapkan bahwa pola osilasi kinematik tubuh dan sepeda cenderung konsisten saat dikayuh secara normal. Oleh karena itu, munculnya penyimpangan pola yang ekstrem dan mendadak dapat dikategorikan sebagai indikator bahaya atau risiko jatuh [26].

Saat kendali sepeda mulai hilang, sinyal sensor akan memperlihatkan karakteristik gelombang yang distingtif. Seperti penelitian pada sistem deteksi berbasis IoT, anomali gerakan akibat ketidakstabilan dapat dibedakan dari aktivitas berkendara biasa melalui perubahan drastis pada orientasi sudut dan percepatan [27]. Lebih spesifik lagi, pada skenario benturan keras *(impact)* seperti terjatuh dari ketinggian atau menabrak rintangan, beban dinamik yang diterima rangka sepeda akan melonjak tajam. Fenomena ini terekam oleh sensor sebagai nilai akselerasi vertikal (sumbu Z) yang sangat tinggi, di mana magnitudonya berbanding lurus dengan kecepatan dan ketinggian jatuh [28]. Pemahaman terhadap karakteristik fisika gerak inilah yang menjadi dasar penentuan fitur ekstraksi agar model dapat memprediksi risiko secara presisi.

2.1.2 Inertial Measurement Unit (IMU)

Inertial Measurement Unit (IMU) merupakan peran sensor yang bertugas merekam dinamika pergerakan objek di ruang tiga dimensi. Secara teknis, IMU merupakan integrasi beberapa sensor mikro seperti Akselerometer dan Giroskop [29].

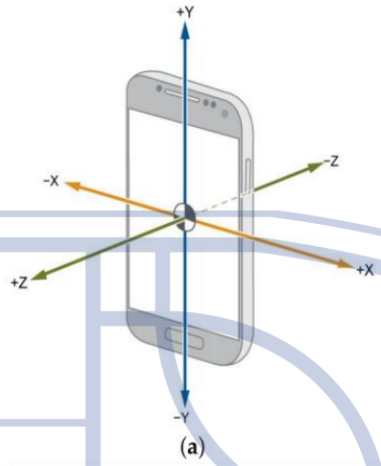
Komponen pertama, yaitu Akselerometer berbasis teknologi MEMS *(Micro-Electro-Mechanical Systems)* bekerja dengan mendeteksi percepatan objek yang dirasakan sebagai berat relatif terhadap kondisi jatuh bebas *(free fall)*, yang direpresentasikan dalam satuan m/s^2 atau unit *g-force*. Melalui komponen triaksialnya, data percepatan linier direkam secara simultan pada tiga sumbu ortogonal (X, Y, Z) untuk menghasilkan parameter representatif yang mencerminkan besarnya getaran, guncangan *(shock)*, maupun orientasi perangkat. [30] Dalam analisis pola gerak, akselerometer sangat berguna untuk menangkap fase benturan atau hentakan kaki. Namun, ketika digunakan sendirian, sensor ini rentan terhadap akumulasi kesalahan, terutama jika dimanfaatkan untuk memperkirakan posisi dalam jarak jauh [31].

Komponen kedua, Giroskop, berfungsi mengukur kecepatan sudut *(angular velocity)* atau laju rotasi objek terhadap sumbu-sumbunya. Berbeda dengan giroskop mekanis konvensional, giroskop pada *smartphone* menggunakan teknologi MEMS berbasis Prinsip Gaya Coriolis. Di dalamnya terdapat struktur massa mikro yang bergetar *(vibrating mass)* secara konstan. Ketika perangkat mengalami rotasi, gaya Coriolis akan menyebabkan massa tersebut menyimpang (berdefleksi) ke arah tegak lurus dari getarannya [32].

Selain dua sensor inersia utama tersebut, *dataset* ini sebenarnya melengkapi konfigurasi IMU dengan *Magnetometer* dan modul GPS. Secara teknis, *Magnetometer* berfungsi menyediakan referensi arah absolut terhadap utara magnetik bumi, yang dalam literatur sering dimanfaatkan untuk mengoreksi *drift* pada estimasi sudut orientasi tubuh atau navigasi [33]. Akan tetapi, dalam penelitian ini, data *Magnetometer* tidak diprioritaskan sebagai fitur utama. Keputusan ini didasari oleh fakta biomekanika bahwa mekanisme jatuh lebih didominasi oleh lonjakan gaya *(force)* dan kecepatan rotasi instan, bukan oleh perubahan arah mata angin (kardinal) yang statis. Sementara itu, data GPS tidak digunakan dalam penelitian ini karena tujuan penelitian adalah mendeteksi Risiko Jatuh berdasarkan perubahan gerakan yang direkam sensor inersia. Oleh karena itu, hanya data akselerometer dan giroskop yang digunakan pada proses analisis.

Dalam konteks penelitian ini, karakteristik sensor juga dipengaruhi oleh posisi perangkat dan sistem koordinat yang digunakan. Berdasarkan spesifikasi *dataset*, *smartphone* ditempatkan pada stang sepeda dengan orientasi layar *default* seperti yang diilustrasikan pada Gambar 1.1. Sistem koordinat didefinisikan secara relatif terhadap layar ponsel, sumbu X mewakili arah lebar ponsel (kanan positif), sumbu Y mewakili arah panjang ponsel (atas positif), dan sumbu Z tegak lurus keluar dari permukaan layar. Pemahaman terhadap orientasi lokal ini krusial karena ketika perangkat diposisikan mendatar pada stang

sepeda, pemetaan sumbu akan mewakili dimensi gerak yang spesifik seperti nilai Z negatif yang merepresentasikan arah gaya yang masuk ke dalam atau di belakang layar, yang menjadi indikator penting dalam analisis vektor jatuh [34].



Gambar 1.1 Pemetaan Sumbu Ortogonal (X, Y, Z) pada Smartphone [33]

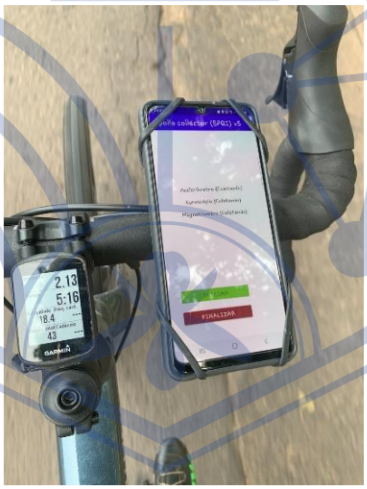
Dalam penerapannya, cara kerja sensor dipengaruhi oleh posisi dan orientasi perangkat pada sepeda. Penelitian ini menggunakan dataset sekunder yang memposisikan *smartphone* secara mendatar pada *handlebar* yang dapat dilihat pada Gambar 1.2, setiap perubahan gerakan yang terjadi akan direkam melalui tiga sumbu koordinat sensor. Berdasarkan orientasi tersebut, mekanisme pembacaan setiap sensor dalam merekam dinamika gerak pesepeda dapat dijelaskan sebagai berikut ini :

1. Akselerometer bekerja dengan cara mengukur perubahan percepatan linier pada tiga sumbu (X, Y, Z) selama sepeda bergerak. [30]
    a. Pada sumbu Y (pergerakan depan-belakang), sensor merekam perubahan percepatan sepeda dengan menghasilkan nilai positif ketika pesepeda menambah kecepatan (akselerasi) dan nilai negatif ketika pesepeda melakukan pengereman mendadak (deselerasi). [35]
    b. Pada sumbu X (pergerakan kanan-kiri), sensor menangkap pergerakan lateral akibat sepeda berbelok ke samping. Jika sepeda bergerak ke arah kanan, maka nilai yang dihasilkan adalah positif, dan sebaliknya menghasilkan nilai negatif jika sepeda bergerak ke arah kiri. [35]
    c. Pada sumbu Z (pergerakan atas bawah), sensor merekam percepatan vertikal yang searah gravitasi, dengan menghasilkan nilai positif ketika sepeda bergerak ke atas dan sebaliknya. Dengan karakteristik ini, sumbu Z akan menghasilkan fluktuasi nilai ketika sepeda melintasi permukaan jalan yang tidak rata seperti polisi tidur atau

lubang. Dalam kondisi yang lebih ekstrem, seperti benturan keras atau pesepeda terhempas ke aspal, sumbu Z dapat merekam hentakan ke bawah (negatif) sebagai lonjakan percepatan vertikal.

Perubahan percepatan dari ketiga sumbu kemudian dikombinasikan menjadi satu besaran magnitudo yang disebut sebagai *Resultant Acceleration* (RA) untuk menangkap intensitas guncangan yang dialami perangkat tanpa memperhatikan arah datangnya gaya. [36]

2. Giroskop bekerja secara kontinu untuk mengukur seberapa cepat sepeda berputar atau laju rotasi sepeda terhadap ketiga sumbunya. Pada perangkat yang dipasang di stang sepeda *(handlebar)*, pergerakan stang paling dominan terekam sebagai rotasi pada sumbu Z. Sensor ini membaca nilai positif ketika stang berputar berlawanan arah jarum jam *(counterclockwise)* dan membaca nilai negatif ketika stang berputar searah jarum jam *(clockwise)* [35]. Giroskop dapat digunakan untuk merekam momen ketidakstabilan karena ketika stang berputar secara tiba-tiba ke salah satu sisi, lonjakan kecepatan sudut yang tajam akan terekam. Kemudian, dalam penelitian ini, informasi lonjakan tersebut diekstraksi menjadi fitur *Resultant Angular Velocity* (RAV) untuk membedakan manuver berbelok yang masih normal dengan anomali gerakan yang mengindikasikan risiko kehilangan keseimbangan.



Gambar 1.2 Posisi Penempatan Sensor Inersia Berbasis Smartphone

2.2 Teknik Pengolahan Data Sensor

2.2.1 Feature Extraction untuk Data Sensor : Resultant Acceleration dan Resultant Angular Velocity

Data sensor inersia yang terekam memiliki format tiga sumbu (x, y, z) memiliki ketergantungan yang tinggi terhadap orientasi perangkat. Mengingat posisi perangkat dipasang pada stang sepeda dapat bervariasi tingkat kemiringannya atau bergeser saat terjadi guncangan, penggunaan langsung data mentah *(raw)* saja dapat menurunkan akurasi model. Oleh karena itu, tahap *extraction* dengan memfokuskan pada perhitungan nilai *resultant Signal Vector Magnitude* (SVM) untuk menghasilkan fitur yang bebas terhadap pengaruh orientasi *(Orientation Independent)* [34].

Fitur pertama yang diekstraksi adalah *Resultant Acceleration (Accres)*, yaitu fitur merepresentasikan nilai total magnitudo guncangan atau gaya inersia yang dialami perangkat tanpa memperhatikan arah datangnya gaya. Fitur ini merepresentasikan intensitas guncangan secara komprehensif dengan gabungan dari kontribusi akselerasi pada sumbu (x, y, z) [36]. Nilai ini dihitung dengan rumus :

$$Acc_{res} = \sqrt{Acc_x^2 + Acc_y^2 + Acc_z^2} \quad \text{..... ()}$$

Keterangan :

Acc_x , Acc_y , Acc_z : Nilai percepatan pada sumbu x, y dan z (m/s^2)

Fitur kedua adalah *Resultant Angular Velocity (Gyrores)* merupakan fitur yang mengukur nilai intensitas total rotasi perangkat dengan perhitungan dari kontribusi ketiga sumbu (x,y,z). Fitur ini merepresentasikan momen ketidakstabilan saat perangkat mulai kehilangan kendali sebelum jatuh, karena perubahan rotasi secara tiba-tiba sebelum kejadian jatuh. Fitur ini dihitung dengan rumus :

$$Gyro_{res} = \sqrt{Gyro_x^2 + Gyro_y^2 + Gyro_z^2} \quad \text{..... (1)}$$

Keterangan :

$Gyro_x$, $Gyro_y$, $Gyro_z$: Nilai kecepatan sudut pada sumbu x, y dan z (rad/s)

Penilaian ini terbukti efektif dalam studi keselamatan transportasi mikro seperti *e-scooter*, dimana ambang batas deteksi jatuh ditentukan berdasarkan total magnitudo akselerasi dan rotasi, bukan pada sumbu parsial semata [37].

2.2.2 Penanganan Ketidakseimbangan Data dengan SMOTE

Dalam ranah penelitian deteksi jatuh, kondisi ketidakseimbangan kelas *(class imbalance)* merupakan tantangan inheren yang umum dan sulit dihindari. Secara alami, frekuensi jatuh (kelas minoritas) memiliki jumlah yang rendah dibandingkan dengan aktivitas harian atau *Activities of Daily Living* (kelas mayoritas) [38]**.** Kondisi ini menyebabkan distribusi data yang tidak seimbang sehingga berpotensi terciptanya bias pada algoritma *Machine Learning*. Model klasifikasi standar cenderung memprioritaskan akurasi dalam mengenali kelas mayoritas namun gagal mengenali pola pada kelas minoritas yang dalam konteks keselamatan, kejadian jatuh merupakan target yang paling penting untuk dideteksi secara akurat [23]**.**

Untuk mengatasi permasalahan tersebut, penelitian ini menerapkan metode *Synthetic Minority Over-Sampling Technique* (SMOTE). Metode ini berbeda dengan teknik *random oversampling* konvensional yang hanya membuat duplikasi data mentah pada kelas minoritas dan berisiko menyebabkan *overfitting*. SMOTE bekerja dengan pendekatan pembangkitan data sintetis baru dengan memanfaatkan hubungan antar sampel pada ruang fitur. Proses ini dilakukan dengan memilih sampel dari kelas minoritas, menentukan sejumlah tetangga terdekat *(k-nearest neighbors)*, kemudian menciptakan sampel baru melalui interpolasi linier di antara sampel – sampel tersebut [39]**.**

Secara matematis, pembentukan data sintetis pada SMOTE dapat dirumuskan sebagai berikut :

$$x_{new} = x_i + \lambda\,(x_{nn} - x_i) \quad \text{..... (2)}$$

Keterangan:

x_{new} : Data sintetis baru yang dihasilkan

x_i : Sampel data dari kelas minoritas

x_{nn} : Salah satu tetangga terdekat *(nearest neighbor)* dari x_i

λ : Bilangan acak dengan rentang yang menentukan posisi relatif data sintetis di antara x_i dan x_{nn}

Berdasarkan persamaan tersebut, data sintetis x_{new} dihasilkan melalui interpolasi linier antara sampel minoritas dan salah satu tetangga terdekatnya [40]. Proses ini menyebabkan data baru terbentuk pada titik di sepanjang garis yang menghubungkan x_i dan x_{nn} dalam ruang fitur.

Meskipun efektif dalam meningkatkan jumlah data minoritas, SMOTE menghasilkan sampel sintetis untuk seluruh data minoritas tanpa mempertimbangkan tingkat kesulitan

klasifikasinya. Akibatnya, sebagian data sintetis dapat terbentuk pada area yang relatif aman dan kurang memberikan kontribusi terhadap peningkatan kemampuan model dalam membedakan kedua kelas [41]. Oleh karena itu, penelitian ini menerapkan pengembangan varian SMOTE, yaitu *Borderline*-SMOTE (BLSMOTE). Proses pembentukan data sintetis dilakukan dengan mekanisme interpolasi linier yang sama seperti SMOTE. Namun, metode ini berfokus pada sampel kelas minoritas berada di sekitar batas pemisah antar kelas *(decision boundary).* Sampel – sampel tersebut dianggap lebih sulit diklasifikasikan karena dikelilingi oleh sejumlah besar tetangga dari kelas mayoritas. Dengan memprioritaskan area batas kelas, *Borderline*-SMOTE menghasilkan data sintetis yang lebih informatif dibandingkan SMOTE sehingga dapat membantu model mempelajari pola kelas minoritas secara lebih efektif [42].

2.2.3 Pelabelan Data dan Ambang Batas (Thresholding)

Pada penelitian ini, *dataset Bike&Safe* yang digunakan merupakan *dataset* sekunder yang tidak memiliki label *ground truth* mengenai kondisi Risiko Jatuh. Kondisi ini menyebabkan data tidak dapat langsung digunakan pada pendekatan *supervised learning*, sehingga diperlukan mekanisme untuk menghasilkan label secara sistematis. Sebagai solusi, literatur terbaru dalam analisis deret waktu mengusulkan mekanisme *Thresholding* sebagai alternatif dalam menetapkan label target secara otomatis. Dengan menggunakan cara ini, label akan digantikan oleh kriteria statistik, yaitu penetapan batas demarkasi antara data normal operasi dengan anomali [43].

Aturan Sigma adalah salah satu teknik *thresholding* berbasis statistik yang umum digunakan untuk mendeteksi data anomali. Secara teoritis, Aturan Tiga Sigma *(3-Sigma Rule)* mempostulasikan bahwa pada suatu μ distribusi data yang bekerja dalam 3σ kondisi wajar *(unimodal)*, probabilitas data untuk berada di sekitar nilai rata-ratanya, yaitu dalam rentang tiga kali standar deviasi , sebesar 99,7%. Artinya, hanya terdapat peluang yang sangat kecil (sekitar 0,3%) bagi sebuah data normal untuk muncul di luar interval [44,45] Sedangkan variasi lainnya, yaitu Aturan Dua Sigma *(2-Sigma Rule)* cenderung mengidentifikasi lebih banyak data *outlier* karena ambang batas ditetapkan pada dua standar deviasi di atas rata-rata. Dalam distribusi normal, sekitar 95% data berada dalam rentang $\pm 2\sigma$ sehingga sensitivitas terhadap data anomali meningkat [43].

$$Threshold = \mu + (2\ x\ \sigma) \quad (3)$$

Keterangan :

μ : Nilai rata-rata data

σ : Standar deviasi data

2σ : Dua kali standar deviasi

Threshold : Nilai ambang batas untuk mendeteksi anomali

Dalam hal implementasi, setiap fitur sensor yang memiliki nilai melebihi ambang batas hasil perhitungan 2-Sigma digolongkan sebagai *outlier*. Identifikasi jenis *outlier* ini merupakan salah satu metode krusial untuk menentukan apakah sinyal tersebut mengalami deviasi yang signifikan dari norma aktivitas yang biasa. Oleh karena itu, jenis *outlier* statistik ini diterjemahkan secara fisik sebagai "Risiko Jatuh" karena merepresentasikan lonjakan guncangan atau ketidakstabilan ekstrem yang secara probabilistik tidak memungkinkan terjadi pada aktivitas bersepeda yang aman.

2.3 Framework OSEMN

OSEMN merupakan suatu kerangka kerja dalam analisis data yang dirancang untuk mengarahkan proses pengolahan data dari tahap awal hingga tahap akhir. Pendekatan ini membantu memastikan bahwa data diproses dengan baik dan hasil analisis yang diperoleh dapat digunakan secara optimal. OSEMN terdiri atas lima tahapan utama, yaitu *Obtain*, *Scrub*, *Explore*, *Model*, dan *Interpret* [46]. Metode OSEMN diterapkan sebagai kerangka alur kerja *data science* yang menghubungkan proses pengumpulan data hingga pada tahap interpretasi hasil. Adapun rincian dari kelima tahapan OSEMN sebagai berikut:

1. *Obtain data*

   Tahap awal dalam kerangka kerja OSEMN adalah *Obtain*, yaitu tahap yang berfokus pada proses pengumpulan data dari sumber yang relevan dan dapat dipercaya. Pada tahap ini, data yang masih bersifat mentah *(raw data)* dikumpulkan melalui berbagai cara, seperti dari basis data internal, layanan *Application Programming Interface* (API), maupun melalui teknik pengambilan data berbasis web *(web scraping)* [46]. Tujuan dari tahap ini adalah memastikan data yang diperoleh telah mencukupi dan mampu merepresentasikan kondisi yang diteliti sebelum memasuki proses pengolahan dan analisis selanjutnya.

2. *Scrub data*

   Setelah data dikumpulkan, tahap *scrub data* dilakukan untuk menyiapkan data dan meningkatkan kualitas data sebelum dianalisis lebih lanjut. Tahapan ini mencakup proses pembersihan kesalahan dalam data, penanganan nilai hilang *(missing value)*, standarisasi format, serta penghapusan data yang tidak relevan dengan tujuan penelitian [46].

Pembersihan data diperlukan untuk mengurangi potensi bias dan menghasilkan *dataset* lebih konsisten sehingga layak untuk eksplorasi dan pemodelan.

3. *Explore data*

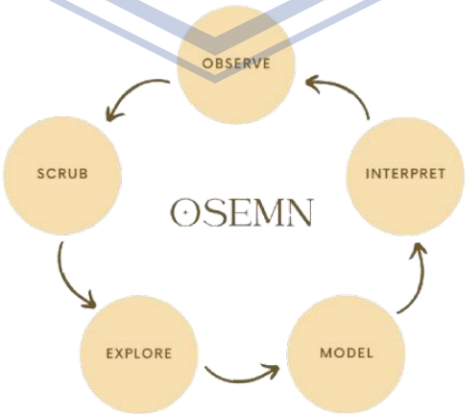
   Tahapan *explore* atau disebut juga dengan Analisis Data Eksploratif (EDA) merupakan tahap fase investigasi awal untuk memahami karakteristik intrinsik dari data yang sudah dibersihkan. Pada tahap ini, analisis data eksploratif (EDA) digunakan untuk mengidentifikasi pola, tren atau korelasi antar variabel sebelum melakukan pemodelan [46]. Visualisasi statistik sering digunakan dalam tahap ini untuk memetakan distribusi data, sehingga peneliti dapat menentukan strategi ekstraksi fitur yang paling optimal untuk tahap selanjutnya.

4. *Model data*

   Tahap Model merupakan bagian utama dalam proses analisis, dimana algoritma *Machine Learning* digunakan untuk mempelajari pola yang terdapat pada data yang telah disiapkan yang disebut dengan *data train*. Tahapan ini meliputi perancangan model, proses pemilihan algoritma, serta pengaturan parameter agar model bisa menangani karakteristik data baik bersifat linier maupun nonlinier [47]. Tujuannya adalah mengonstruksi model prediktif atau klasifikasi yang mampu melakukan generalisasi data baru dengan akurat.

5. *Interpret data*

   Tahapan *interpret* berfokus pada evaluasi dan pemaknaan terhadap kinerja model yang telah dibangun. Pada tahap ini, *Output* model divalidasi dengan menggunakan metrik evaluasi seperti *Accuracy*, *Precision*, *Recall*, dan *F1-score* untuk menentukan keandalan prediksi model [47]. Tahapan *Interpret* berfungsi mengubah hasil perhitungan statistik menjadi pemahaman yang bermakna, sehingga temuan penelitian dapat disimpulkan secara objektif dan digunakan sebagai dasar pengambilan keputusan berbasis data.



Gambar 1.3 Framework OSEMN

2.4 Machine Learning

Machine Learning (ML) merupakan bidang kecerdasan buatan yang berfokus pada pengembangan algoritma komputasi yang memiliki kemampuan belajar data empiris untuk menghasilkan prediksi atau keputusan [48]. Berbeda dengan pemrograman konvensional yang bergantung pada instruksi eksplisit yang statis, ML memungkinkan sistem untuk membangun model matematis melalui proses pelatihan *(training)*. Tujuan utama dari proses ini bukan sekadar menghafal data latih, melainkan untuk mencapai kemampuan generalisasi, yaitu kemampuan model untuk memproses dan memprediksi data baru yang belum pernah dilihat sebelumnya dengan akurat dan efisien [49]. Secara umum, *Machine Learning* terbagi menjadi dua, yaitu:

1. *Supervised Learning*

   Supervised Learning merupakan suatu pendekatan di mana algoritma pelatihan dilakukan dengan menggunakan data yang telah memiliki label target *(ground truth)* yang jelas. Dalam skema ini, suatu model memiliki tujuan utama untuk menemukan hubungan yang terdapat pada suatu variabel *output* (label) dengan suatu variabel *input* (fitur) dengan melakukan prediksi pada masa depan [48]. Dalam pendekatan ini, terdapat dua subkategori utama yang dapat dilakukan, yaitu klasifikasi yang bertujuan untuk melakukan prediksi pada suatu label diskrit, serta regresi yang bertujuan untuk melakukan prediksi pada suatu label kontinu. Pendekatan ini sangat bergantung pada kualitas pelabelan data untuk menghasilkan prediksi yang presisi [50].

2. *Unsupervised Learning*

   Sebaliknya, *Unsupervised Learning* beroperasi pada data yang tidak memiliki label atau instruksi target. Algoritma jenis ini dirancang untuk mengeksplorasi struktur intrinsik data guna menemukan pola tersembunyi, seperti pengelompokan *(clustering)* berdasarkan kemiripan karakteristik antar data. Metode ini sering digunakan untuk analisis eksploratif guna memahami distribusi data tanpa adanya panduan label dari *dataset* [50].

   Secara umum, alur kerja sistem *Machine Learning* melalui beberapa tahapan utama mencakup representasi data, evaluasi, dan optimasi. Proses dimulai dengan ekstraksi fitur dari data mentah, kemudian dilanjutkan dengan fase pelatihan *(training)* di mana algoritma menyesuaikan parameter internalnya untuk meminimalkan *error* prediksi. Indikator keberhasilan utama dari sebuah model ML bukanlah sekadar kemampuannya mengingat data latih, melainkan kemampuannya untuk melakukan generalisasi yaitu

mempertahankan performa akurasi yang tinggi saat diterapkan pada data pengujian *(testing data)* atau skenario dunia nyata yang dinamis [49].

2.5 Random Forest

Random Forest atau *Random Decision Forest* merupakan salah satu algoritma pembelajaran kelompok *(ensemble learning)* yang umum digunakan dalam bidang *fall detection.* Algoritma ini bekerja dengan cara membangun banyak pohon keputusan selama masa pelatihan dan menghasilkan luaran berupa mode kelas untuk klasifikasi atau rata-rata prediksi untuk regresi [9,51]. *Random Forest* dikembangkan oleh *Leo Breiman* dan berbasis pohon keputusan yang menggunakan teknik *bagging* dan pemilihan fitur acak dengan tujuan meningkatkan akurasi dan ketahanan hasil prediksi [9]. Secara teknis, *Random Forest* dirancang untuk mengurangi ketergantungan antar pohon keputusan dengan menerapkan proses pengambilan sampel dan pemilihan fitur secara acak [51]. Pendekatan ini memungkinkan model untuk mempelajari pola dan menghasilkan keputusan berdasarkan struktur pohon yang terbentuk saat proses pembelajaran data. *Random Forest* dapat digunakan untuk tujuan klasifikasi yang menghasilkan respon kategorikal maupun regresi yang menghasilkan respon kontinu [51]. Oleh karena itu, *Random Forest* banyak digunakan dalam pengkategorian otomatis, prediksi data dan berbagai aplikasi *machine learning* [6]

Random Forest dibangun dengan menghasilkan beberapa pohon keputusan, yang dilakukan melalui pemanfaatan sampel data hasil *Bootstrap* dan pemilihan fitur *input* secara acak. Setiap pohon yang ada dikonstruksi menggunakan porsi data yang dipilih secara acak, umumnya sebesar 0,632 kali dari ukuran sampel asli (n). Proses ini melibatkan dua bentuk acak : [51]

1. Pengambilan sampel dengan pengembalian *(sampling with replacement)*, yaitu kemungkinan kemunculan berulang dari beberapa poin data dalam satu set pelatihan baru.
2. Pemilihan fitur acak *(random feature selection)*, yaitu setiap pembelahan di sebuah pohon terjadi, variabel optimal dipilih dari subset prediktor acak.

Random Forest dapat dijabarkan sebagai suatu model *ensemble* yang terdiri atas *B* pohon keputusan, yaitu $\{T_1(X), \ldots., T_B(X)\}$. Dalam konteks penelitian ini, X adalah vektor deskriptor berdimensi p yang diasosiasikan dengan setiap jendela waktu pada sensor data IMU. Setiap pohon menghasilkan prediksi $\{\hat{Y}_1 = T_1(X), \ldots., \hat{Y}_B = T_B(X)\}$, dan $\hat{Y}_B$ adalah *output* dari pohon ke-b.[52] Setelah seluruh pohon terbentuk, hasil akhir $\hat{Y}$ ditentukan

melalui mekanisme *majority-voting* [51]. Dalam proses ini, *output* akhir sistem ditentukan oleh *output* yang dipilih oleh mayoritas pohon keputusan untuk tugas klasifikasi atau rata-rata prakiraan dari setiap pohon untuk tugas regresi [9].

Algoritma *Random Forest* dipilih karena kemampuannya dalam menangani data berdimensi tinggi dan ketahanannya terhadap *overfitting* [9]. Penelitian terdahulu yang disebutkan dalam penelitian [9] telah menunjukkan bahwa diantara algoritma klasifikasi tradisional, *Random Forest* efektif dalam mengurangi *noise* dan variabilitas pada data sensor. *Random Forest* memiliki sejumlah keunggulan utama, diantaranya tingkat kesalahan yang relatif rendah, performa yang konsisten, dan efisien dalam menangani *dataset* besar dan data yang tidak lengkap [6]. Selain itu, *Random Forest* dapat dieksekusi secara pararel sehingga dapat meningkatkan kecepatan pemrosesan data [51].

Dalam konteks data sensor yang sering kali tidak seimbang *(imbalanced data)*, *Random Forest* banyak digunakan karena kemampuan generalisasinya relatif baik dan tahan terhadap ketidakseimbangan data [53]. Selain itu, *Random Forest* mampu menghasilkan berbagai pengukuran tambahan seperti matriks proksimasi dan nilai penting fitur *(feature importance)*, yang efektif mengidentifikasi parameter sensor yang paling berpengaruh dalam deteksi risiko jatuh [51]. Kombinasi dari kumpulan pohon keputusan dalam *Random Forest* berkontribusi pada pengurangan bias dan variabilitas model [51], sehingga menghasilkan performa yang lebih stabil pada data baru.

Pengukuran kepentingan fitur *(feature importance)* pada *Random Forest* digunakan untuk menunjukkan fitur yang berpengaruh dalam risiko deteksi jatuh. Dalam implementasi penelitian ini, *feature importance* diukur menggunakan pendekatan *Mean Decrease Gini* (MDG). Pendekatan ini mengukur dengan mengevaluasi kontribusi setiap variabel berdasarkan akumulasi penurunan tingkat ketidakmurnian *(Gini Impurity)*. Proses ini dihitung pada setiap tahap pembelahan (*splitting*) node di seluruh pohon keputusan yang menyusun model tersebut. Semakin besar nilai MDG, maka fitur tersebut semakin berpengaruh [54]. Pendekatan *Mean Decrease Gini* dirumuskan sebagai berikut:

$$MDG = \frac{1}{n} \sum [d(h,t).I(h,t)] \quad (4)$$

Keterangan:

n : Total jumlah pohon keputusan

$d(h,t)$: Besaran nilai penurunan indeks gini pada variabel X saat proses pemisahan di simpul ke-t

$I(h,t)$: Fungsi indikator bernilai 1 saat terjadi di simpul *t*, dan bernilai 0 untuk lainnya.

2.6 Logistic Regression (LR)

Logistic Regression adalah salah satu model klasifikasi biner untuk membuat model probabilitas terjadinya suatu kejadian dengan dua kemungkinan nilai (0 dan 1) [55]. Model ini termasuk dalam keluarga *generalized linear models* dan digunakan ketika variabel dependen memiliki dua kategori [56]. Dalam konteks penelitian ini, nilai 0 mewakili kondisi Tidak Risiko Jatuh, sedangkan nilai 1 mewakili kondisi Risiko Jatuh.

Secara matematis, *Logistic Regression* memprediksi probabilitas suatu kejadian berdasarkan kombinasi linear variabel prediktor. Untuk memastikan bahwa nilai probabilitas berada pada rentang 0 hingga 1, digunakan transformasi *log-odds* melalui fungsi logistik (*sigmoid*) yang menghasilkan kurva berbentuk S [57] sehingga nilai kombinasi linear fitur sensor IMU dapat dikalkulasikan menjadi probabilitas terjadinya risiko jatuh.

$$ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k \quad (5)$$

$$p = \frac{1}{1+e^{-z}} \text{ dengan } z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k \quad (6)$$

Pada persamaan $ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$ (5), p menyatakan probabilitas terjadinya suatu kejadian, β_0 adalah konstanta (*intercept*), β_i adalah koefisien regresi, X_i adalah variabel prediktor. Bentuk pada persamaan $p = \frac{1}{1+e^{-z}}$ dengan $z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$ (6) merupakan fungsi *sigmoid* yang membatasi nilai probabilitas pada rentang 0 dan 1.

Model ini mengasumsikan bahwa *log-odds* dari *outcome* memiliki linearitas dengan variabel prediktor [57]. Selain itu, asumsi penting lainnya meliputi independensi observasi, tidak adanya multikolinearitas sempurna antara variabel prediktor, dan hubungan linear antara variabel prediktor dengan nilai *logit* dari variabel respon [56].

Dalam penelitian ini, *Logistic Regression* digunakan sebagai model *baseline* untuk dibandingkan dengan *Random Forest*. *Logistic Regression* dikenal dengan karakteristiknya yang sederhana, linier dan transparan sedangkan model nonlinier seperti *Random Forest* lebih kompleks [11] dalam menangkap hubungan dan interaksi antar variabel. Penelitian sebelumnya menunjukkan bahwa meskipun *Random Forest* menghasilkan akurasi yang sedikit lebih tinggi, namun *Logistic Regression* memiliki konsistensi dan interpretabilitas yang lebih baik [11]. Oleh karena itu, *Logistic Regression* ditetapkan sebagai titik acuan

dalam menilai apakah pola risiko jatuh dari fitur sensor IMU dapat dipisahkan secara *linear* sebelum menggunakan model yang lebih kompleks.

2.7 Leave-One-Lap-Out Cross-Validation (LOLO-CV)

Leave-One-Out Cross-Validation (LOOCV) adalah salah satu teknik untuk mengevaluasi model machine learning yang termasuk dalam keluarga *cross-validation*. Teknik ini digunakan untuk mengevaluasi kemampuan generalisasi model terhadap data yang dikecualikan dalam tahap pelatihan [58-59]. LOOCV adalah bentuk khusus dari *K-Fold Cross-Validation* yang merepresentasikan jumlah lipatan (*K*) sama dengan jumlah total observasi dalam *dataset (K=N)* [60]. Pada setiap iterasi, satu observasi akan dikecualikan untuk menjadi data validasi dan observasi yang tersisa akan digunakan sebagai data pelatihan. Proses ini berulang sebanyak jumlah total observasi sehingga setiap data (observasi) akan menjadi data uji masing-masing satu kali [58].

Namun, penggunaan *cross validation* pada data sensor yang berbentuk *time-series* dapat menyebabkan kebocoran data *(data leakage)* karena segmen data yang berdekatan memiliki tingkat korelasi yang tinggi. Sampel partisipan atau observasi dapat muncul pada kedua data pelatihan dan data pengujian akibat adanya paparan data sebelumnya. [61] Jika terdapat ketergantungan antar-residual karena pengelompokan data berdasarkan faktor waktu atau ruang, maka dibutuhkan metode pemisahan data yang dapat mempertahankan struktur tersebut, misalnya pendekatan *blocked* atau *leave-one-group-out*. [62]

Dataset yang digunakan dalam penelitian ini terdiri atas 9 unit observasi yang berasal dari 3 rute dengan rincian 3 putaran *(lap)* pada setiap rutenya. Dalam skema ini, 1 *lap* akan digunakan sebagai data uji sedangkan 8 *lap* lainnya digunakan sebagai data pelatihan. Proses ini diulang sebanyak 9 kali hingga seluruh *lap* sudah menjadi data uji tepat 1 kali. Dengan pendekatan ini, performa model mencerminkan kemampuan deteksi risiko jatuh pada lintasan yang belum digunakan dalam pelatihan. Metode LOGO-CV digunakan untuk mengevaluasi performa model berdasarkan kemampuannya dalam melakukan prediksi terhadap kelompok data yang belum pernah dilibatkan selama proses pelatihan [62]. Skema validasi ini dapat mengukur kemampuan generalisasi model terhadap kondisi baru atau pengguna baru dan memastikan tidak ada tumpang tindih antara data pelatihan dan pengujian [63].

Penelitian sebelumnya menyatakan bahwa metode LOOCV bermanfaat pada *dataset* berukuran kecil karena setiap iterasi menempatkan satu observasi sebagai data validasi sedangkan seluruh data sisanya *(N-1)* sebagai data pelatihan [59]. Selain itu, metode ini

digunakan untuk mengatasi ukuran *dataset* yang terbatas sekaligus mengurangi risiko *overfitting* pada model klasifikasi seperti *Random Forest* [58]. Akurasi dari skema LOSO *(Leave-One-Subject-Out)* sebagai salah satu bentuk LOGO-CV memberikan evaluasi yang lebih representatif karena menguji kemampuan model dalam menggeneralisasi subjek dengan karakteristik spesifik per-individu yang tidak terlibat dalam data pelatihan. Oleh karena itu, hasil LOSO lebih mencerminkan kondisi penerapan di dunia nyata. [61]

Evaluasi performa pada LOOCV diperoleh melalui proses iterasi sebanyak jumlah observasi dalam *dataset* kemudian dilanjutkan dengan perhitungan akurasi keseluruhan terhadap observasi yang terlah menjadi data uji [58]. Evaluasi model dapat diformalkan melalui kerangka matematika *K-Fold Cross-Validation* yang mencakup varian LOOCV, LOGO-CV dan lainnya. [62] Formulasi matematis ini berfungsi untuk memberikan estimasi akurasi prediktif model terhadap observasi yang tidak digunakan dalam fase pelatihan.

$$\hat{S}_k = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{y}_i^{-j[i]}) \quad (7)$$

Keterangan :

i : $1, \ldots, n$. n mengindeks setiap titik data

j : $1, \ldots, k$. k mengindeks lipatan *(folds)*

$-j[i]$: Menunjukkan bahwa model dapat dilatih tanpa menyertakan *fold* yang mengandung data ke-i

L : *Loss function* untuk mengukur kesalahan prediksi

y_i : Nilai observasi sebenarnya

$\hat{y}_i$: Nilai prediksi model

Dalam konteks penelitian ini yang menggunakan LOLO-CV sebagai bentuk implementasi khusus dari LOGO-CV, setiap *fold* j tidak lagi merepresentasikan subset acak data, melainkan satu kelompok lap. Oleh karena itu, *dataset* keseluruhan D dipartisi menjadi beberapa kelompok yang saling bebas yang dapat dinyatakan sebagai berikut :

$$D = \{D_1, D_2, \ldots . D_L\} \quad (8)$$

Pendekatan ini memastikan bahwa tidak terjadi kebocoran informasi antar kelompok lap serta meningkatkan akurasi dalam evaluasi kemampuan generalisasi model pada lap yang belum pernah dilihat sebelumnya.

2.8 Hyperparameter Tuning

Optimasi *Hyperparameter* adalah tahapan penting dalam alur kerja model *machine learning*. *Hyperparameter* atau *parameter tuning* adalah sejumlah konfigurasi tambahan

pada sebagian besar metode *supervised learning* yang mempengaruhi performa model dan berisiko *overfitting* [48]. Sebagai salah satu algoritma *machine learning*, *Random Forest* juga memiliki sejumlah *hyperparameter* yang harus dioptimalkan [51]. *Parameter tuning* harus melalui seleksi dan penyesuaian dengan mempertimbangkan risiko *overfitting* pada model [48]. Penerapan *hyperparameter* yang digabungkan dengan teknik *cross-validation* berbeda akan menghasilkan klasifikasi yang lebih optimal [51].

Terdapat penelitian terdahulu yang menerapkan metode pencarian terstruktur untuk mendapatkan kombinasi parameter yang optimal. Optimasi *hyperparameter* tersebut dilakukan menggunakan *Grid Search* dan *Randomized Search* dari pustaka *Scikit-learn* pada bahasa pemrograman *Python*. Dengan melakukan *tuning* seperti ini, model dapat meningkatkan efektivitasnya dalam menangkap pola data yang relevan sekaligus menyeimbangkan akurasi tinggi dengan efisiensi komputasi yang krusial untuk aplikasi *real-time* perangkat *wearable*. [9] Dengan mempertimbangkan efisiensi komputasi dari metode *Grid Search*, penelitian ini mengadopsi pendekatan serupa dengan mengimplementasikan metode *Grid Search* pada pustaka *caret* pada perangkat lunak R.

Karakteristik model yang digunakan dalam penelitian merupakan acuan dalam menentukan kebutuhan optimasi. Salah satu dari dua elemen stokastik dalam *Random Forest* berhubungan dengan pemilihan variabel untuk proses pemisahan (*splitting*), sehingga parameter *mtry* diperlakukan sebagai *parameter tuning* [51]. Sementara itu, model pembanding yang digunakan dalam penelitian ini dioptimasi melalui metode regularisasi atau penalti untuk meningkatkan performa. Metode regularisasi pada *Logistic Regression* adalah perkembangan dari regresi konvensional yang melibatkan proses penurunan koefisien menjadi nol *(LASSO)*, mendekati nol *(ridge)*, atau kombinasi dari keduanya *(elastic net)* untuk meningkatkan akurasi prediksi sekaligus mencegah *overfitting* pada model [48]. Menggabungkan kedua penalti menjadi satu kombinasi adalah tujuan pembentukan *elastic net* agar model yang dihasilkan lebih sederhana, stabil sekaligus akurat dibandingkan menggunakan salah satu metode secara terpisah [48].

Untuk mencapai hasil *tuning* yang objektif, proses optimasi harus disertakan dalam skema evaluasi yang ketat. Salah satu penelitian terdahulu menggunakan pendekatan *10-fold cross-validation* dalam pelatihan modelnya untuk mencegah *overfitting* dan memastikan ketahanannya [9]. Oleh karena itu, penelitian ini menerapkan skema *cross validation* berbasis lap (LOLO-CV) untuk memastikan objektivitas pengujian pada segmen data sensor. Dalam skema validasi ini, algoritma membutuhkan metrik acuan spesifik seperti sensitivitas atau *recall* yang merepresentasikan kemampuan model dalam mendeteksi kejadian jatuh

secara akurat. Metrik *recall* didefinisikan sebagai proporsi subjek yang diprediksi positif diantara semua subjek positif [48]. Dalam konteks penelitian ini, posisi metrik *recall* lebih diprioritaskan karena keberhasilan dalam identifikasi kasus jatuh yang benar-benar terjadi lebih penting daripada kasus yang tidak terjadi.

2.9 Metrik Evaluasi

Evaluasi kinerja model klasifikasi adalah tahapan penting untuk memastikan keandalan sistem deteksi jatuh sebelum diterapkan secara *real-time*. Penelitian dari [9] menggunakan serangkaian metrik yang komprehensif dalam evaluasi model deteksi jatuh dengan tujuan menyeimbangkan efisiensi komputasi dengan akurasi prediksi. Terlebih pada kasus data tidak seimbang, penggunaan metrik evaluasi yang tepat menjadi pertimbangan penting karena algoritma cenderung bias ke arah mayoritas yaitu aktivitas sehari-hari, yang menyebabkan performa deteksi pada kelas minoritas yaitu kejadian jatuh menjadi buruk [23]. Berdasarkan beberapa penelitian deteksi jatuh terdahulu [9,23,36,48,64], evaluasi kinerja model umumnya dilakukan menggunakan metrik seperti *Confusion Matrix, Accuracy, Precision, Recall (Sensitivity), F1-score* dan *Specifity* untuk mengukur kemampuan model dalam memprediksi kejadian jatuh di antara aktivitas normal.

1. *Confusion Matrix*

   Confusion Matrix adalah dasar dari perhitungan seluruh metrik evaluasi klasifikasi. Matriks ini merupakan tabel kontingensi sederhana 2 x 2 yang memetakan hasil prediksi model terhadap nilai aktual dari data uji [48,64]. *Confusion Matrix* membagi hasil klasifikasi menjadi empat kategori utama, yaitu [64]:

   a. *True Positive* (TP) : Sampel positif yang diklasifikasi dengan benar.
   b. *True Negative* (TN) : Sampel negatif yang diklasifikasikan dengan benar.
   c. *False Positive* (FP) : Sampel negatif yang salah diklasifikasikan sebagai positif.
   d. *False Negative* (FN) : Sampel positif yang salah diklasifikasikan sebagai negatif.

Tabel 1.1 Confusion Matrix

	Predict Positive	*Negative Positive*
Actual Positive	TP	FN
Actual Negative	FP	TN

2. *Accuracy*

   Akurasi merupakan metrik yang paling umum digunakan untuk mencerminkan tingkat kinerja model secara umum melalui pengukuran proporsi keberhasilkan prediksi

kejadian jatuh dibandingkan dengan keseluruhan prediksi yang dihasilkan oleh model [36]. Akurasi dirumuskan sebagai berikut :

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (9)$$

meskipun intuitif, akurasi seringkali *misleading* jika digunakan pada *dataset* yang tidak seimbang. Penelitian [23] menyebutkan bahwa pada skenario kelas tidak seimbang *(class-imbalanced)*, nilai akurasi yang tinggi mungkin hanya mencerminkan kemampuan model dalam memprediksi kelas mayoritas, sementara gagal mendeteksi kelas minoritas yang lebih krusial. Hal ini didukung oleh penelitian [64] yang menyatakan bahwa akurasi memberikan nilai bobot yang sama pada setiap kelas, sehingga performa yang rendah dalam mendeteksi kejadian jatuh dapat tertutupi oleh tingginya tingkat prediksi benar pada aktivitas normal.

3. *Precision dan Recall (Sensitivity)*

   Untuk mengatasi bias akurasi pada data sensor yang tidak seimbang, evaluasi juga mempertimbangkan metrik *Precision* dan *Recall*.

   a. *Precision* adalah metrik yang digunakan untuk mengukur ketepatan model dalam mengidentifikasi sampel positif [36]. Metrik ini penting untuk meminimalkan *false alarm* yang dapat mengganggu kenyamanan pengguna. *Precision* dirumuskan sebagai berikut :

   $$Precision = \frac{TP}{TP+FP} \quad (10)$$

   b. *Recall (Sensitivity)* adalah metrik yang digunakan untuk mengukur kemampuan model dalam mengidentifikasi sampel positif [36]. Dalam aplikasi medis dan keselamatan, *Sensitivity* atau *Recall* menjadi acuan penting karena kegagalan mendeteksi risiko jatuh *(false negative)* jauh lebih berbahaya dibandingkan peringatan palsu *(false positive)* [23,64]. *Recall* dapat dirumuskan sebagai berikut :

   $$Recall = \frac{TP}{TP+FN} \quad (11)$$

4. *F1-score*

   F1-score adalah rata-rata harmonik antara *Precision* dan *Recall* yang merepresentasikan keseimbangan kinerja model dalam menangani kesalahan *false positive* dan *false negative* dalam satu ukuran [9]. *F1-score* dirumuskan sebagai berikut :

   $$F1 - score = \frac{2*Precision*Recall}{Precision+Recall} \quad (12)$$

Penelitian [23] menyatakan bahwa *F1-score* adalah metode yang *robust* untuk mengukur diskriminasi model, terutama ketika sensitivitas dan spesifitas saling membatasi satu sama lain pada *dataset* berskala besar.

5. *Specificity*

Dalam penelitian *fall detection*, selain memiliki kemampuan dalam mendeteksi kejadian jatuh, sistem juga dituntut untuk andal dalam mengenali aktivitas normal agar tidak menghasilkan peringatan palsu (*false alarm*). *Specificity* didefinisikan sebagai proporsi subjek yang diprediksi negatif (tidak jatuh) di antara seluruh subjek yang benar-benar negatif [48]. *Specificity* dirumuskan sebagai berikut : [23]

$$Specificity = \frac{TN}{TN+FP} \quad (13)$$

Nilai *Specificity* mencerminkan proporsi sampel negatif yang berhasil diidentifikasi dengan benar oleh model. Sebagai tambahan, meskipun kesalahan *false positive* dalam deteksi jatuh tidak sefatal kegagalan deteksi *(false negative)*, kesalahan ini tetap berpotensi memicu respons berlebihan yang tidak perlu dan kelelahan terhadap peringatan *(alert fatigue)*. [9]

2.10 Tinjauan Pustaka (Literature Review)

No	Referensi	Judul	Konteks / Subjek	Sensor dan Posisi	Metode	Temuan Utama	Gap (Terhadap skripsi)	Relevansi Penelitian
1	[9]	*Enhanced Fall Detection using Optimized Random Forest Classifier on Wearable Sensor Data*	Deteksi jatuh berbasis *wearable sensor* untuk keselamatan lansia. Menggunakan *dataset* publik	Akselerometer dan giroskop yang ditanam pada *wearable devices* seperti *smartwatch* dan *fitness tracker*	*Optimized* RF, *feature engineering*, PCA, *hyperparameter tuning (Grid & Randomized Search), 10-fold cross validation*	RF teroptimasi menghasilkan akurasi ± 90%, F1 ±90%, *inference time* hanya ~0.045 detik. *Outperform baseline* RF, SVM, KNN	Penelitian menggunakan data terkontrol berlabel dari *wearable devices* sehingga tidak mengeksplorasi strategi pelabelan otomatis berbasis statistik pada data naturalistik dengan sensor *non-body mounted*.	Menjadi referensi penggunaan RF, melakukan *hyperparameter tuning*, dan memprioritaskan metrik *recall* dalam deteksi jatuh. Selain itu, mendukung penggunaan fitur ekstraksi RA dan RAV dalam penelitian ini
2	[19]	*How smooth is your ride? Comparison of sensors and methods for surface quality assessment using IMUs*	Kualitas permukaan jalan raya	*Smartphone* IMU dan *industrial-grade* IMU dipasang pada *handlebar*	Perbandingan lima metode kalkulasi indeks kekasaran permukaan. Analisis statistik komparatif untuk membandingkan stabilitas dan konsistensi hasil.	*Smartphone* yang diposisikan di *handlebar* efektif dalam menangkap getaran jalan raya, namun terpapar *noise* yang tinggi dari permukaan jalanan.	Hanya fokus pada kondisi dan kualitas jalan, tidak mencakup deteksi atau klasifikasi risiko jatuh.	Menegaskan pentingnya model yang *robust* untuk meminimalisir *false positives* sebab getaran akibat kondisi jalan dapat mempengaruhi sinyal IMU

3	[21]	*Evaluation of Algorithm for the Fall and Fall Direction Detection during Bike Riding*	Keamanan pesepeda	Akselerometer pada helm untuk mengukur percepatan dan arah jatuh	*Physical threshold* (ambang percepatan)	Nilai percepatan jatuh pesepeda secara empiris berada pada rentang 2.84g hingga 3.61 g	Menggunakan *threshold* fisik tanpa membahas pelabelan otomatis dan menerapkan *machine learning*	Konteks penelitian relevan karena Membahas deteksi jatuh pada pesepeda menggunakan fitur berbasis SVM yang setara dengan RA. Keterbatasan metode *threshold* statis dalam membedakan kejadian jatuh dan getaran ekstrem memperkuat urgensi penerapan metode berbasis ML dan pendekatan statistik untuk melabeli data
4	[22]	*Wearable Fall Detection System with Real-Time and Notification Capabilities*	Deteksi jatuh lansia di *outdoor*	Akselerometer + giroskop di pinggang yang terintegrasi dengan modul GPS dan modul komunikasi	FSM berbasis *statistical thresholding* untuk membedakan fase *pre-fall, impact dan post-fall*	Penggunaan ambang batas statis menghasilkan akurasi sebesar 99.7% dalam mendeteksi fase *impact*.	Dinamika tubuh lansia berbeda dengan sepeda sehingga ambang batas yang digunakan mungkin kurang sensitif untuk pesepeda.	Mendukung pendekatan heuristik / *statistical labeling* melalui analsisi karakteristik fisik sinyal IMU. Pola perubahan nilai ekstrem pada fase

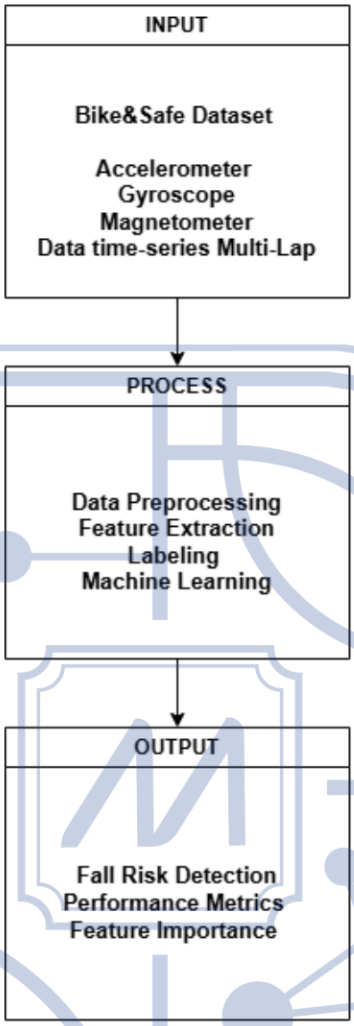
								impact memvalidasi penggunaan aturan statistik.
5	[65]	*A model based on cyclist fall experiments which predicts the maximum allowable handlebar disturbance from which a cyclist can recover balance*	Eksperimen jatuh yang berfokus pada dinamika sepeda dan respon tubuh pesepeda	12 kamera *motion capture*, *force sensors, EMG sensors*, dan IMU.	*Bayesian multilevel logistic regression model* dengan pendekatan *Bayesian Model Averaging* untuk prediksi kemungkinan jatuh. Estimasi *Maximum Allowable Handlebar Disturbance* (MAHD). Analisis faktor prediktor (kecepatan dan keahlian menjaga keseimbangan)	Kecepatan melaju dan keahlian menjaga keseimbangan adalah faktor utama yang menentukan besarnya gangguan yang dapat ditahan pesepeda sebelum jatuh	Model MAHD divalidasi melalui eksperimen laboratorium terkontrol, tetapi belum menyesuaikannya dengan data IMU naturalistik dan belum menggunakan pendekatan *Machine Learning* untuk labeling otomatis	Memperkuat relevansi LR sebagai model pembanding algoritma RF yang lebih kompleks dalam konteks analisis risiko jatuh
6	[37]	*Pre-Impact Fall*	Deteksi jatuh sebelum	Akselerometer + giroskop	Perbandingan antara	Fitur RA dan RAV lebih	Belum ada pengujian spesifik	Memvalidasi penggunaan fitur

		Detection for E-scooter Riding Using an IMU : Threshold-Based, Supervised, and Unsupervised Approaches	benturan pada pengguna *e-scooter* berbasis IMU	dipasang di *e-scooter*	*threshold-based*, *supervised learning* dan *unsupervised learning*. Ektraksi fitur *Resultant Acceleration* dan *Resultant Angular Velocity*.	*robust* terhadap perubahan orientasi dibandingan komponen aksis individual	pada pesepeda dengan gangguan getaran tinggi pada *handlebar* sepeda	berbasis magnitudo vektor (ASVM dan GSVM) yang setara dengan RA dan RAV. Selain itu, perbandingan antara pendekatan *threshold-based* dan ML menjustifikasi penggunaan model klasifikasi RF dan LR dalam menangani kompleksitas dinamika kendaraan roda dua
7	[38]	*Automatic Fall Risk Detection based on Imbalanced Data*	Deteksi risiko jatuh pada lansia	Akselerometer + Giroskop pada tubuh *(wearable)*	Menguji algoritma KKN, SVM, *AdaBoost, XGBoost*. Data imbalance menggunakan SMOTE berbasis video dan *pose estimation*.	Teknik SMOTE berkontribusi pada peningkatan performa model pada data jatuh yang tidak seimbang. SMOTE meningkatkan *recall fall* dari	Konteks penelitian berfokus pada masalah ketidakseimbangan kelas data pada aktivitas harian yang sudah berlabel.	Menjadi referensi dalam menangani *imbalanced data* menggunakan SMOTE dan menegaskan pentingnya metrik *recall* dalam konteks deteksi jatuh

					Evaluasi dengan metrik akurasi, *precision*, *recall* dan *F1-score*.	0.73 ke 0.88 dan menghasilkan *F-score* tertinggi 0.85.		
8	[66]	*Cadence Detection in Road Cycling Using Saddle Tube Motion and Machine Learning*	Deteksi irama kayuhan pada jalanan pesepeda menggunakan pergerakan rangka sepeda	Akselerometer + Giroskop pada rangka sepeda. Perolehan *ground truth* berasal dari pemasangan *Hall-effect sensor* pada *crank arm* dengan *magnet*	*Supervised deep learning* menggunakan arsitektur CNN-LSTM. Melakukan *resampling* ke 50Hz, normalisasi, *windowing time-series*, *split training-validation-test* (70-15-15) dalam tahap *preprocessing*.	Sensor IMU yang dipasang pada komponen sepeda dapat mendeteksi irama kayuhan dengan akurasi 95% dibandingkan sensor *Hall-effect* sebagai *ground truth*.	Penelitian fokus pada pengukuran performa kayuhan dari 4 pesepeda, belum mengeksplorasi deteksi risiko jatuh berbasis sinyal IMU pada pesepeda.	Mendukung penggunakan *sliding window* untuk menangkap pola sinyal serta memberikan referensi penanganan *imbalanced data* melalui penyesuaian bobot *(weighting)* yang memperkuat penggunaan teknik SMOTE pada penelitian ini
9	[67]	*Predicting Turn Maneuvers of Cyclists Using Bicycle-Mounted IMU with CNN-LSTM*	Prediksi manuver belok pesepeda	*Smartphone-based* IMU pada stang sepeda + sensor *Nordic Thingy:52* pada *frame* untuk	*Deep learning* menggunakan 1D-CNN + LSTM + attention layer. Ektraksi fitur spasial dan	IMU pada stang sepeda dengan model CNN-LSTM dapat memperkirakan manuver belok menggunakan indikator pra-	Penelitian memprediksi manuver belok sebelum kejadian, tetapi belum diaplikasikan dalam lingkungan lalu lintas nyata. Selain itu, belum	Menunjukkan peletakan sensor IMU pada stang sepeda mampu menangkap dinamika rotasi dan kemiringan sepeda. Hal ini relevan dengan

				eksperimen komparatif	temporal dari data IMU. Evaluasi dengan *F1-score*.	manuver dengan performa tinggi dalam skenario terkontrol	mengkaji deteksi kejadian benturan atau risiko jatuh pesepeda.	penggunaan fitur RAV untuk membedakan antara manuver normal dan indikasi jatuh
10	[27]	*Cyclist Fall Detection System via the Internet of Things (IoT)*	Deteksi jatuh untuk pesepeda dilengkapi notifikasi darurat berbasis IoT	Sensor IMU MPU6050 dipasang pada sepeda dan terintregasi dengan modul WiFi dan GPS untuk notifikasi lokasi melalui aplikasi Blynk	Pendekatan *threshold-based* dengan acuan sudut kemiringan tubuh (melebihi ±60°)	Sistem mampu mendeteksi dan mengirimkan notifikasi lokasi jatuh secara *real-time* melalui platform IoT	Penelitian menggunakan *threshold* berbasis sudut yang sederhana tanpa pendekatan *Machine Learning*	Menunjukkan implementasi deteksi jatuh dengan *threshold-based* sederhana pada sensor IMU. Keterbatasan metode *threshold* sederhana memvalidasi kebutuhan pendekatan ML yang lebih adaptif.

2.11 Flow Pipeline Input-Process-Output Sistem Deteksi Risiko Jatuh Pesepeda



Gambar 1.4 Alur Pipeline IPO Sistem Deteksi Risiko Jatuh Pesepeda

Gambar 1.4 menunjukkan alur umum dari penelitian ini yang terdiri atas tiga tahapan utama, yaitu *input*, *process* dan *output*. Tahap *input* menggunakan *dataset Bike&Safe* yang memuat data dari tiga jenis sensor inersia, *timestamp* dan data hasil pengukuran *multi-lap* yang diperoleh selama bersepeda. Selanjutnya, data diproses melalui tahapan *preprocessing*, *feature extraction* dan pemodelan menggunakan algoritma *machine learning*. Tahap *output* menghasilkan klasifikasi risiko jatuh, nilai metrik evaluasi kinerja model, dan informasi *feature importance* untuk mengetahui fitur apa saja yang berperan besar dalam proses klasifikasi.