BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Diabetes Melitus

Diabetes Melitus (DM) adalah kelompok gangguan metabolisme yang ditandai dengan tingginya kadar glukosa dalam darah (*hiperglikemia*). Kondisi ini disebabkan oleh gangguan dalam sekresi insulin, fungsi insulin, atau keduanya. Hiperglikemia kronis pada diabetes berhubungan dengan komplikasi jangka panjang yang dapat merusak atau menyebabkan kegagalan pada berbagai organ tubuh, terutama mata, ginjal, saraf, jantung, dan pembuluh darah (Scobie, 2019). Jika tidak ditangani dengan baik, diabetes melitus dapat menyebabkan komplikasi serius yang berdampak pada berbagai organ tubuh, antara lain:

1) *Retinopati Diabetik*: Kerusakan pembuluh darah di retina mata, yang dapat menyebabkan kebutaan.

2) *Nefropati Diabetik*: Kerusakan ginjal akibat kadar gula darah yang tinggi.

3) *Neuropati Diabetik*: Gangguan saraf yang menyebabkan mati rasa, terutama di kaki dan tangan.

4) Penyakit Kardiovaskular: Risiko penyakit jantung dan stroke meningkat signifikan.

5) Komplikasi Akut: Ketoasidosis diabetik dan hipoglikemia berat.

6) Insulin, hormon yang dihasilkan oleh pankreas, berperan penting dalam mengatur kadar gula darah dengan membantu glukosa masuk ke dalam sel untuk dijadikan energi (*American Diabetes Association,* 2021).

Berdasarkan penyebabnya, Diabetes Melitus diklasifikasikan menjadi:

1) Diabetes Tipe 1: Terjadi akibat kerusakan sel beta pankreas sehingga produksi insulin sangat rendah atau tidak ada sama sekali.

2) Diabetes Tipe 2: Disebabkan oleh resistensi insulin atau produksi insulin yang tidak mencukupi. Jenis ini paling umum terjadi, terutama akibat pola makan tidak sehat, kurang aktivitas fisik, dan obesitas.

3) Diabetes Gestasional: Diabetes yang muncul selama kehamilan akibat perubahan hormon yang memengaruhi kerja insulin.

Menurut Rathmann dkk., (2004), diabetes adalah salah satu masalah kesehatan terbesar di dunia, dengan prevalensi yang terus meningkat, termasuk di Indonesia. Menurut Kementerian Kesehatan Republik Indonesia (2020), prevalensi diabetes di Indonesia pada tahun 2020 mencapai 10,9% dari total populasi, dengan sebagian besar kasus terdiagnosis pada tahap lanjut. Hal ini menunjukkan pentingnya deteksi dini dan upaya pencegahan untuk mengurangi dampak negatif diabetes terhadap kesehatan masyarakat.

Faktor risiko yang dapat meningkatkan kemungkinan seseorang mengembangkan diabetes melitus meliputi obesitas, kurangnya aktivitas fisik, riwayat keluarga, serta faktor usia dan jenis kelamin (*American Diabetes Association*, 2021). Deteksi dini melalui pemeriksaan kesehatan yang rutin menjadi salah satu upaya penting untuk mencegah terjadinya komplikasi serius akibat diabetes, seperti penyakit jantung, stroke, dan gagal ginjal.

2.1.2 Machine Learning

Machine Learning adalah cabang dari kecerdasan buatan (*Artificial Intelligence*) yang berfokus pada kemampuan sistem untuk belajar dari data secara otomatis tanpa diprogram secara eksplisit. Algoritma *machine learning* mampu mengenali pola dari data yang diberikan, membuat prediksi, dan mengambil keputusan berdasarkan pola tersebut (Dinata & Hasdyna, 2020). Dalam konteks kesehatan, *machine learning* digunakan untuk menganalisis data medis dalam rangka mendeteksi, mengklasifikasikan, dan memprediksi risiko penyakit.

Pada prediksi risiko diabetes, *machine learning* memungkinkan penggunaan data medis seperti indeks massa tubuh (*BMI*), tekanan darah, pola makan, dan gaya hidup untuk memprediksi kemungkinan seseorang mengidap diabetes. Oleh karena itu, *machine learning* memberikan potensi yang besar dalam meningkatkan kesadaran akan risiko kesehatan, dan membantu masyarakat untuk melakukan tindakan preventif lebih awal. Hal ini didukung oleh penelitian yang dilakukan Verma & Verma (2022), bahwa *machine learning* dapat dimanfaatkan untuk meningkatkan efisiensi dan akurasi dalam pengambilan keputusan medis.

Terdapat tiga tipe utama dalam *machine learning*, yaitu *supervised learning*, *unsupervised learnin*g, dan *reinforcement learning* (Dinata & Hasdyna, 2020), yaitu:

1) *Supervised Learning*

Pada tipe ini, model dilatih menggunakan data yang telah diberi label atau *output* yang diinginkan. Model belajar untuk memetakan *input* dengan *output* yang sesuai. *Supervised learning* sering digunakan untuk masalah klasifikasi dan regresi. Model yang sering digunakan dalam *supervised learning*, antara lain:

a) *Linear Regression*

Digunakan untuk memprediksi nilai kontinu berdasarkan input data.

b) *Decision Tree*

Model yang membagi data ke dalam subset yang lebih kecil berdasarkan keputusan yang dibuat pada setiap node.

c) *Support Vector Machine* (SVM)

Digunakan untuk klasifikasi dan regresi, dengan menemukan batasan optimal untuk memisahkan kelas-kelas data.

2) *Unsupervised Learning*

Berbeda dengan *supervised learning, unsupervised learning* tidak menggunakan data dengan label. Model ini berfokus pada pencarian pola atau struktur tersembunyi dalam data. Aplikasi dari *unsupervised learning* antara lain *clustering* dan asosiasi. Beberapa model yang digunakan adalah:

a) *K-Means Clustering*

Digunakan untuk membagi data ke dalam kelompok-kelompok berdasarkan kesamaan fitur.

b) *Hierarchical Clustering*

Teknik pengelompokan data yang membentuk hierarki dari cluster yang lebih kecil menjadi lebih besar.

c) *Principal Component Analysis* (PCA)

Digunakan untuk mengurangi dimensi data dengan mempertahankan informasi penting.

3) *Reinforcement Learning*

Pada tipe ini, model belajar melalui interaksi dengan lingkungan dan menerima *feedback* dalam bentuk *reward* atau *punishment*. Model berusaha

memaksimalkan *reward* dengan mengambil keputusan yang paling optimal. Contoh model dalam *reinforcement learning* adalah:

a) Q-Learning

Salah satu algoritma populer untuk pengambilan keputusan yang optimal dalam situasi dengan banyak kemungkinan.

b) *Deep Q-Network* (DQN)

Kombinasi antara *Q-learning* dengan neural *network* untuk menangani masalah yang lebih kompleks.

Sedangkan metodologi dalam *machine learning* mencakup beberapa tahap berikut (Kreuzberger dkk., 2023):

1) Pengumpulan Data

Proses ini mencakup pengumpulan data mentah dari berbagai sumber, baik melalui sistem otomatis, survei, atau dataset publik. Data harus representatif terhadap masalah yang ingin diselesaikan.

2) Pra-pemrosesan Data

Data yang dikumpulkan sering kali tidak sempurna dan memerlukan pembersihan, normalisasi, atau transformasi. Tahapan ini juga mencakup penanganan *missing values*, pengkodean variabel kategorikal, dan deteksi outlier.

3) *Feature Engineering*

Feature engineering adalah proses memilih, menciptakan, atau mengubah fitur dalam dataset untuk meningkatkan performa model. Teknik seperti seleksi fitur, ekstraksi fitur, dan *scaling* sering digunakan pada tahap ini.

4) Pemilihan Algoritma

Algoritma *machine learning* dipilih berdasarkan sifat masalah, seperti klasifikasi, regresi, clustering, atau rekomendasi. Pemilihan juga dipengaruhi oleh karakteristik data, seperti jumlah data, dimensi, dan tingkat ketidakseimbangan kelas.

5) Pelatihan Model

Pelatihan model dilakukan dengan menggunakan data latih, di mana model mempelajari pola dari data tersebut melalui iterasi. Hyperparameter tuning juga dilakukan pada tahap ini untuk mengoptimalkan kinerja algoritma.

6) Evaluasi Model

Model dievaluasi menggunakan data uji dan matriks tertentu seperti akurasi, *precision*, *recall*, F1 *score*, dan AUC-ROC. Evaluasi ini bertujuan untuk memastikan bahwa model dapat melakukan generalisasi pada data baru.

2.1.3 Random Forest

Random Forest (RF) adalah algoritma *machine learning* berbasis *ensemble learning* yang menggabungkan *output* dari banyak pohon keputusan untuk menghasilkan prediksi yang lebih akurat. Menurut Breiman (2001), RF bekerja dengan membangun banyak pohon keputusan menggunakan *bootstrap sampling* dan memilih fitur secara acak pada setiap pemisahan pohon, yang membantu mengurangi *overfitting* dan membuat model lebih stabil. Berikut proses kerja dari *Random Forest* (Breiman, 2001) :

1) Pengambilan Sampel untuk Membentuk Pohon

Data latih dibagi menjadi beberapa subset menggunakan teknik *bootstrap sampling*. Setiap subset digunakan untuk membangun satu pohon keputusan (*decision tree*).

2) Pembuatan Pohon Keputusan

Setiap pohon keputusan dibuat dengan memilih sejumlah kecil fitur secara acak dari total fitur yang tersedia (*random feature selection*). Pada setiap simpul (node), pemisahan terbaik ditentukan dengan mengukur impurity, misalnya menggunakan Gini Impurity atau Entropy.

3) Prediksi oleh Setiap Pohon

Ketika data uji dimasukkan, setiap pohon memberikan prediksi terhadap kelas tertentu.

4) Penggabungan Prediksi (Pengumpulan Suara Mayoritas)

Prediksi akhir ditentukan berdasarkan voting mayoritas dari semua pohon. Proses voting ini memanfaatkan prinsip ensemble, di mana kombinasi dari banyak model sederhana dapat meningkatkan akurasi dibandingkan hanya menggunakan satu model saja.

Dalam konteks prediksi risiko diabetes, *Random Forest* telah terbukti efektif karena kemampuannya menangani data kompleks dengan banyak fitur. Penelitian oleh Liu dkk., (2022) menggunakan RF untuk memprediksi risiko perkembangan diabetes tipe 2 berdasarkan data prediabetes dan berhasil menunjukkan kinerja yang baik. Studi ini menunjukkan bahwa RF lebih unggul dibandingkan model lain seperti *Logistic Regression* dan *Artificial Neural Networks* dalam menganalisis variabel seperti *BMI*, tekanan darah, usia, dan gaya hidup

Selain itu, Sadeghi dkk., (2022), mengonfirmasi keunggulan RF dalam memprediksi risiko diabetes dengan akurasi yang lebih baik dibandingkan *Decision Tree* dan model klasik lainnya. Penelitian ini juga menegaskan bahwa *hyperparameter tuning* seperti jumlah pohon (*ntree*) dan pemilihan fitur (*mtry*) memiliki dampak signifikan terhadap kinerja RF dalam klasifikasi penyakit diabetes.

2.1.4 Aplikasi Berbasis Website Untuk Kesehatan

Aplikasi *website* untuk kesehatan telah populer karena kemudahan aksesnya sehingga menjangkau lebih banyak pengguna. Penelitian oleh König & Suhr, (2023) menunjukkan bahwa penggunaan aplikasi berbasis *website* memiliki potensi besar dalam meningkatkan literasi kesehatan digital dan kesadaran masyarakat terhadap pentingnya pencegahan penyakit. *Website* sebagai platform memberikan akses yang luas dan fleksibilitas bagi pengguna untuk memperoleh informasi kesehatan secara mandiri. Dalam konteks diabetes, aplikasi *website* dapat digunakan untuk memonitor parameter kesehatan seperti pola makan, aktivitas fisik, dan faktor risiko lainnya.

Penelitian lain yang mendukung relevansi aplikasi *website* dalam manajemen kesehatan adalah Sanhaji dkk., (2024), mengembangkan platform berbasis *website* untuk memprediksi risiko diabetes. Aplikasi ini dirancang untuk membantu pengguna dalam memantau dan mencegah perkembangan diabetes. Hal ini menunjukkan efektivitas teknologi digital dalam deteksi dini dan manajemen diabetes. Dengan memanfaatkan platform ini, pengguna dapat lebih proaktif dalam memantau kesehatan mereka, yang berkontribusi pada pencegahan dan pengelolaan diabetes secara lebih efektif.

Selain itu, aplikasi ini dapat memberikan rekomendasi atau langkah pencegahan bagi pengguna, mendorong mereka untuk mengambil langkah-langkah pencegahan atau segera berkonsultasi dengan tenaga medis.

2.1.5 SMOTE

SMOTE (*Synthetic Minority Over-sampling Technique*) merupakan teknik untuk mengatasi masalah ketidakseimbangan kelas dalam masalah klasifikasi. Ketidakseimbangan kelas terjadi ketika jumlah sampel dari satu kelas jauh lebih sedikit dibandingkan dengan kelas lainnya, yang dapat menyebabkan model klasifikasi lebih cenderung mengklasifikasikan ke kelas mayoritas. SMOTE bekerja dengan cara menghasilkan sampel sintetis untuk kelas minoritas, meningkatkan representasi kelas tersebut dan mengurangi bias terhadap kelas mayoritas (Matharaarachchi dkk., 2024).

SMOTE bekerja dengan mengidentifikasi sampel dalam kelas minoritas dan kemudian menghasilkan sampel sintetis dengan cara berikut:

1) Pemilihan Titik Data, dimana SMOTE memilih sebuah titik data x dalam kelas minoritas.
2) Pencarian Tetangga Terdekat, dimana untuk setiap titik data x, SMOTE memilih k tetangga terdekat yang ada dalam kelas minoritas berdasarkan jarak Euclidean.
3) Penciptaan Titik Sintetis, SMOTE kemudian menciptakan sampel sintetis dengan mengambil perbedaan antara titik data x dan tetangga terdekatnya, lalu menambahkan perbedaan tersebut ke titik data x. Proses ini dilakukan dengan cara berikut:

$$x_{new} = x + \delta \cdot (x_{neighbor} - x) \quad \text{2-7}$$

Dimana:

δ : faktor pengali yang menghasilkan titik data sintetis yang lebih realistis.

Tujuan dari SMOTE adalah untuk memperbaiki performa model klasifikasi yang dilatih pada dataset yang tidak seimbang. Dengan menghasilkan data sintetis untuk kelas minoritas, SMOTE membantu model untuk belajar lebih baik tentang karakteristik kelas minoritas, sehingga meningkatkan kemampuan prediksi terhadap kelas tersebut (Elreedy dkk., 2024).

2.1.6 Random Undersampling

Random Undersampling (RUS) merupakan salah satu metode penyeimbangan data (*resampling*) yang digunakan untuk mengatasi permasalahan ketidakseimbangan kelas (*imbalanced dataset*). Metode ini bekerja dengan cara mengurangi jumlah data pada kelas mayoritas secara acak hingga mencapai jumlah tertentu sehingga distribusi data menjadi lebih seimbang. Menurut Chen dkk., (2024), teknik *undersampling* bertujuan mengurangi dominasi kelas mayoritas yang sering menyebabkan model cenderung mempelajari pola dari kelas tersebut dan mengabaikan karakteristik kelas minoritas.

Pada dataset yang tidak seimbang, model pembelajaran mesin umumnya menghasilkan performa yang tinggi pada kelas mayoritas, tetapi memiliki kemampuan yang rendah dalam mengenali kelas minoritas. Kondisi ini dapat menyebabkan nilai *accuracy* terlihat baik meskipun model gagal mengidentifikasi data pada kelas minoritas secara optimal. Oleh karena itu, diperlukan teknik penyeimbangan data agar proses pelatihan model dapat mempelajari karakteristik setiap kelas secara lebih proporsional (Jiang dkk., 2025).

Proses *Random Undersampling* dilakukan dengan memilih sebagian data dari kelas mayoritas secara acak dan menghapus data lainnya hingga diperoleh rasio kelas yang diinginkan. Metode ini memiliki keunggulan berupa implementasi yang sederhana, mampu mengurangi ukuran dataset, serta mempercepat proses pelatihan model karena jumlah data yang digunakan menjadi lebih sedikit (Untoro & Yusuf, 2023). Selain itu, pengurangan dominasi kelas mayoritas dapat membantu model memberikan perhatian yang lebih besar terhadap pola yang dimiliki oleh kelas minoritas.

Meskipun demikian, *Random Undersampling* juga memiliki kelemahan karena sebagian data pada kelas mayoritas akan dihapus selama proses penyeimbangan. Apabila jumlah data yang dihilangkan terlalu banyak, terdapat kemungkinan informasi penting yang sebenarnya relevan bagi proses klasifikasi ikut terbuang. Oleh karena itu, pemilihan jumlah sampel yang dipertahankan pada kelas mayoritas perlu dilakukan secara hati-hati agar keseimbangan data dapat dicapai tanpa mengurangi representativitas dataset secara signifikan (Chen dkk., 2024).

2.1.7 Hybrid Sampling

Hybrid sampling merupakan pendekatan penanganan *class imbalance* yang mengombinasikan teknik *oversampling* dan *undersampling* dalam proses *resampling*. Pendekatan tersebut digunakan untuk meningkatkan representasi kelas minoritas sekaligus mengurangi dominasi kelas mayoritas. Kombinasi kedua pendekatan tersebut telah digunakan dalam penelitian terkait klasifikasi data tidak seimbang, termasuk penggunaan SMOTE bersama *Random Undersampling* (RUS) (Jang, 2025).

Penggunaan kombinasi SMOTE dan *undersampling* juga telah diteliti sejak pengembangan SMOTE. Peranginangin dkk., (2020) mengevaluasi SMOTE yang dikombinasikan dengan *undersampling* kelas mayoritas sebagai salah satu pendekatan untuk menangani ketidakseimbangan kelas. Sementara itu, penelitian empiris terhadap berbagai metode *sampling* menunjukkan bahwa SMOTE dan *Random Undersampling* merupakan teknik yang dapat digunakan untuk mengubah distribusi kelas pada dataset yang tidak seimbang (Buda dkk., 2018).

Dalam penelitian ini, *hybrid sampling* diterapkan dengan mengombinasikan SMOTE dan *Random Undersampling* untuk menangani ketidakseimbangan kelas pada data risiko diabetes. Kombinasi tersebut diterapkan pada data sebelum proses pelatihan algoritma *Random Forest*. Dengan pendekatan tersebut, model dilatih menggunakan data yang telah diseimbangkan.

2.1.8 Confusion Matrix

Confusion Matrix adalah alat yang digunakan untuk mengukur kinerja model klasifikasi dalam *machine learning* dengan membandingkan prediksi model terhadap nilai aktual dari data (Erbani dkk., 2024). Matrix ini terdiri dari empat elemen utama:

1) *True Positive* (TP): Jumlah kasus yang benar diklasifikasikan sebagai positif.
2) *True Negative* (TN): Jumlah kasus yang benar diklasifikasikan sebagai negatif.
3) *False Positive* (FP): Kasus negatif yang salah diklasifikasikan sebagai positif.
4) *False Negative* (FN): Kasus positif yang salah diklasifikasikan sebagai negatif.

Tabel 2. 1 Bentuk Confusion Matrix

	Nilai Sebenarnya	
	Positif	**Negatif**

Nilai Prediksi	Positif	*True Positive* (TP)	*False Positive* (FP)
	Negatif	*False Negative* (FN)	*True Negative* (TN)

Confusion matrix menjadi alat penting dalam evaluasi model karena membantu menilai kesalahan klasifikasi yang terjadi, baik itu *false positives* maupun *false negatives* (Sathyanarayanan & Tantri, 2024). Matriks ini digunakan untuk menghitung matriks penting lainnya, seperti akurasi, presisi, *recall*, *F1 score*, *F1 Macro*, dan *Balanced Accuracy* yang dapat memberikan gambaran yang lebih lengkap tentang kinerja model dalam menangani data. Penjelasan mengenai matriks tersebut adalah sebagai berikut:

1) *Accuracy:* Mengukur seberapa banyak prediksi yang benar dibandingkan dengan seluruh data yang diprediksi. Jika dataset tidak seimbang, *reliance* hanya pada *accuracy* tidak cukup, karena matriks ini tidak mempertimbangkan distribusi kelas. Untuk mengatasi masalah ini, diperlukan pendekatan lain yang lebih informatif. Rumus dari *accuracy* adalah:

$$\frac{TP + TN}{TP + TN + FP + FN} \quad \text{2-1}$$

2) *Precision*: Mengukur seberapa banyak prediksi positif yang benar dibandingkan dengan semua prediksi positif yang diberikan oleh model. Pada deteksi dini diabetes, FP yang tinggi dapat menyebabkan individu yang sehat menjadi khawatir dan mungkin menjalani tes atau perawatan yang tidak perlu. *Precision* memastikan bahwa jika seseorang diprediksi memiliki risiko diabetes, prediksi tersebut memiliki keakuratan tinggi. Rumus *precision* adalah:

$$\frac{TP}{TP + FP} \quad \text{2-2}$$

3) *Recall (Sensitivity):* Mengukur seberapa banyak data positif yang berhasil terdeteksi oleh model dibandingkan dengan seluruh data positif yang ada. Dalam konteks deteksi dini risiko diabetes, *recall* tinggi sangat penting untuk meminimalkan *false negatives* (FN), yaitu kasus di mana seseorang yang benar-benar berisiko tidak terdeteksi oleh model. Rumus *recall* adalah:

$$\frac{TP}{TP + FN} \quad \text{2-3}$$

4) *F1 Score:* Merupakan rata-rata harmonic antara *precision* dan *recall*, yang memberikan gambaran lebih seimbang antara keduanya, terutama pada dataset yang tidak seimbang. Dalam aplikasi deteksi dini risiko diabetes, F1 Score membantu mengevaluasi model secara keseluruhan dengan mempertimbangkan seberapa baik model mendeteksi kasus positif tanpa mengabaikan akurasi prediksi, dengan rumus:

$$2 \times \frac{Precision \times Recall}{Precision + Recall} \quad \text{2-4}$$

5) *F1 Macro:* Pada klasifikasi multiclass, nilai F1 Score dihitung untuk setiap kelas terlebih dahulu, kemudian dirata-ratakan. Salah satu metode yang umum digunakan adalah Macro F1 Score, yaitu rata-rata aritmetika dari nilai F1 setiap kelas tanpa mempertimbangkan jumlah sampel pada masing-masing kelas. Dengan demikian, setiap kelas memiliki bobot yang sama terhadap nilai akhir evaluasi sehingga metrik ini sesuai digunakan pada dataset yang mengalami *class imbalance*.

$$\frac{\sum F1\ Score_i}{K} \quad \text{2-5}$$

Dimana:

$F1\ Score_i$: Nilai *f1 score* pada kelas ke-i

K : Jumlah kelas

6) *Balanced Accuracy:* Menghitung rata-rata nilai *recall* dari setiap kelas sehingga mampu memberikan penilaian yang lebih adil pada dataset yang tidak seimbang. Berbeda dengan *accuracy* yang dapat dipengaruhi oleh dominasi kelas mayoritas, *balanced accuracy* memperhatikan kemampuan model dalam mengenali seluruh kelas secara proporsional. Nilai *balanced accuracy* dihitung menggunakan rumus:

$$\frac{\sum Recall_i}{n} \quad \text{2-6}$$

Dimana:

$Recall_i$: Nilai *recall* pada kelas ke-i

n : Jumlah kelas

2.1.9 Functional Testing

Functional Testing adalah jenis pengujian perangkat lunak yang berfokus pada verifikasi fungsionalitas sistem berdasarkan spesifikasi yang telah ditetapkan. Dalam konteks *Black Box Testing*, pengujian ini dilakukan tanpa pengetahuan tentang struktur internal sistem, hanya menguji *input* dan *output*. Tujuan utamanya adalah untuk memastikan bahwa perangkat lunak berfungsi dengan benar dan sesuai dengan yang diharapkan oleh pengguna, serta mendeteksi kesalahan dalam fungsionalitas sistem. Pengujian ini sering kali digunakan untuk memvalidasi sistem informasi dengan menguji fungsi utama (Ayuningtyas dkk., 2023).

Proses ini dimulai dengan analisis kebutuhan dan pembuatan kasus uji berdasarkan skenario yang mungkin terjadi. Selanjutnya, pengujian dilakukan pada *boundary conditions* dan berbagai skenario *input* untuk memverifikasi bahwa sistem berfungsi sesuai harapan. Pemantauan hasil dan pelaporan dilakukan untuk mengidentifikasi kesalahan dan memastikan perbaikan dilakukan jika ditemukan masalah.

Dengan pendekatan ini, *Black Box Testing* berperan penting dalam memastikan kualitas dan keandalan perangkat lunak tanpa perlu memeriksa kode sumbernya.

2.1.10 Usability Testing

Usability testing merupakan metode evaluasi yang melibatkan pengguna secara langsung untuk mengetahui sejauh mana suatu sistem dapat digunakan dalam mencapai tujuan tertentu. Evaluasi *usability* umumnya mencakup tiga aspek, yaitu *effectiveness*, *efficiency*, dan *satisfaction*. *Effectiveness* berkaitan dengan keberhasilan pengguna dalam menyelesaikan tugas, *efficiency* berkaitan dengan sumber daya yang diperlukan untuk mencapai tujuan, sedangkan *satisfaction* berkaitan dengan persepsi dan pengalaman pengguna selama menggunakan sistem (Moumane dkk., 2016).

Pengujian *usability* dapat dilakukan dengan memberikan tugas atau skenario penggunaan kepada responden, kemudian memperoleh penilaian berdasarkan pengalaman mereka setelah menggunakan sistem. Metode ini dapat digunakan untuk mengidentifikasi kesulitan penggunaan dan mengetahui tingkat penerimaan

pengguna terhadap sistem (Georgsson & Staggers, 2016).

Dalam penelitian ini, *usability testing* digunakan untuk mengevaluasi kemudahan penggunaan aplikasi deteksi dini risiko diabetes berbasis website. Pengujian dilakukan dengan meminta responden menggunakan aplikasi secara langsung, kemudian memberikan penilaian melalui kuesioner dengan skala Likert lima tingkat untuk mengetahui persepsi pengguna terhadap usability aplikasi.

2.2 Penelitian Terdahulu

Penelitian terdahulu yang berkaitan dengan topik penelitian ini digunakan sebagai landasan dan pertimbangan dalam merancang studi yang sekarang. Studi – studi terdahulu ini memberikan wawasan tambahan dan gagasan yang memperkuat arah dan tujuan penelitian saat ini. Berikut adalah beberapa penelitian terdahulu yang dijadikan rujukan, yaitu:

Penelitian pertama oleh Abushahla & Pala, (2024) membahas tantangan dataset tidak seimbang dalam klasifikasi deteksi Diabetes Melitus. Dengan menggunakan teknik SMOTE untuk *oversampling* dan validasi silang K-Fold, model secara konsisten menunjukkan peningkatan akurasi, presisi, *recall*, dan F1 *score*. Hasil terbaik dicapai oleh *CatBoost*, yang mampu menangani dataset tidak seimbang secara efektif dengan kombinasi *preprocessing* yang tepat.

Penelitian kedua oleh (Chowdhury dkk., 2024) mengevaluasi algoritma *Random Forest* untuk mendiagnosis diabetes menggunakan dataset BRFSS yang menghadapi masalah ketidakseimbangan kelas. Teknik SMOTE digunakan untuk meningkatkan kualitas data, yang berkontribusi pada kestabilan performa *Random Forest*. Namun, akurasi *Random Forest* masih lebih rendah dibandingkan algoritma *Gradient Boosting* dan *XGBoost*.

Penelitian ketiga oleh Ghosh dkk., (2021) menunjukkan bahwa algoritma *Random Forest* (RF) menghasilkan performa terbaik untuk mengklasifikasikan diabetes. Dengan menggunakan metode *Minimal Redundancy Maximal Relevance* (MRMR) untuk pemilihan fitur, RF mencapai akurasi 99,35%**.** Selain itu, penelitian ini juga mengevaluasi algoritma lain seperti *Gradient Boosting* (GB)**,** *Support Vector Machine* (SVM), dan *AdaBoost* (AB)**.**

Penelitian keempat oleh Ashisha dkk., (2024) menunjukkan bahwa model yang menggunakan metode *random oversampling* untuk menyeimbangkan kelas,

deteksi *outlier* dengan teknik *interquartile range*, dan pemilihan fitur dengan algoritma *Boruta*, memberikan hasil yang sangat baik. Dari empat algoritma pembelajaran mesin yang diuji, yaitu *Random Forest, Gradient Boosting, Light Gradient Boosting,* dan *Decision Trees, Random Forest* menunjukkan performa terbaik. Akurasi tertinggi tercapai dengan 92% pada dataset BRFSS dan 94% pada dataset PIDD, yang lebih tinggi 3% dibandingkan dengan penelitian sebelumnya.

Peneliti kelima oleh Islam dkk., (2025) menunjukkan bahwa penerapan *random oversampling* dan *hyperparameter tuning* dapat meningkatkan performa klasifikasi diabetes pada dataset BRFSS. Empat algoritma yang diuji, yaitu *K-Nearest Neighbors, Random Forest, Decision Tree,* dan *Extra Trees*, menunjukkan peningkatan performa setelah dilakukan *oversampling* dan optimasi *hyperparameter*. Pada dataset BRFSS, *Random Forest* memperoleh *F1-score* sebesar 96,28%, sedangkan *Extra Trees* memberikan performa terbaik dengan *F1-score* sebesar 97,22%. Penelitian tersebut juga menggunakan SHAP, PDP, dan LIME untuk memberikan interpretasi terhadap hasil prediksi model.

Tabel 2.2 Perbandingan variabel penelitian

Kategori	Peneliti 1	Peneliti 2	Peneliti 3	Peneliti 4	Peneliti 5
Peneliti	Abushahla & Pala (2024)	Chowdhury dkk. (2024)	Ghosh dkk. (2021)	Ashisha dkk. (2024)	(Islam dkk., 2025)
Metode	*SMOTE* dan *CatBoost*	*SMOTE* dan *Random Forest*	MRMR, *Random Forest, Gradient Boosting, SVM, AdaBoost*	*Random Oversampling, Boruta, Random Forest, Gradient Boosting, LightGBM, Decision Tree*	*Random Forest, KNN, Decision Tree, dan Extra Trees*
Pengujian	Validasi silang *K-Fold*	Evaluasi performa model klasifikasi	Perbandingan performa beberapa algoritma	Evaluasi performa model klasifikas	Perbandingan performa beberapa algoritma pada kondisi tanpa *oversampling*,

Kategori	Peneliti 1	Peneliti 2	Peneliti 3	Peneliti 4	Peneliti 5
					dengan *oversampling*, dan dengan *oversampling* serta *hyperparameter tuning*
Data	*Pima Indians Diabetes Database* (PIDD)	BRFSS 2015	*Pima Indians Diabetes Database* (PIDD)	BRFSS 2015 dan PIDD	BRFSS 2015
Jumlah Kelas	2 kelas (diabetes dan non-diabetes)	2 kelas (diabetes dan non-diabetes)	2 kelas (diabetes dan non-diabetes)	2 kelas (diabetes dan non-diabetes)	3 kelas (non-diabetes, pre-diabetes, diabetes)
Permasalahan	Ketidakseimbangan dataset diabetes	Ketidakseimbangan kelas pada BRFSS	Klasifikasi diabetes menggunakan Random Forest	Deteksi diabetes dengan data tidak seimbang	Ketidakseimbangan kelas pada dataset BRFSS
Teknik Penanganan Data	*Oversampling* menggunakan SMOTE	SMOTE untuk balancing dataset	Seleksi fitur menggunakan MRMR	*Random Oversampling* dan deteksi outlier	*Random Oversampling*, *Hyperparameter Tuning*,
Algoritma Utama	*CatBoost*	*Random Forest*	*Random Forest*	*Random Forest*	*Extraa Trees*
Hasil Penelitian	*CatBoost* memiliki performa terbaik dengan akurasi tinggi	*Random Forest* memberikan performa stabil namun lebih rendah dari Gradient Boosting dan XGBoost	*Random Forest* memperoleh akurasi 99,35% dan unggul dalam kompleksitas komputasi	*Random Forest* mencapai akurasi 92% pada BRFSS dan 94% pada PIDD	*Extra Trees* memberikan performa tertinggi dengan *Accuracy* 97,23% dan *F1-score* 97,22% pada dataset BRFSS.

Berdasarkan kajian terhadap beberapa penelitian terdahulu pada Tabel 2.1, diketahui bahwa penelitian mengenai deteksi dan klasifikasi diabetes telah menggunakan berbagai algoritma *machine learning*, seperti *Random Forest,*

CatBoost, dan *Gradient Boosting*, serta menerapkan teknik penanganan ketidakseimbangan data seperti SMOTE dan *Random Oversampling*. Beberapa penelitian menggunakan klasifikasi biner, sedangkan Islam dkk. (2025) telah menerapkan klasifikasi *multiclass* menggunakan dataset BRFSS 2015 dengan tiga kelas, yaitu Non-Diabetes, Pre-Diabetes, dan Diabetes. Penelitian tersebut juga menunjukkan bahwa penerapan *Random Oversampling* dan *hyperparameter tuning* dapat meningkatkan performa model Random Forest pada dataset BRFSS.

Penelitian ini memiliki perbedaan pada strategi penanganan *class imbalance* dan fokus pengembangannya. Berbeda dengan Islam dkk. yang menggunakan *Random Oversampling*, penelitian ini menerapkan kombinasi SMOTE dan *Random Undersampling* (RUS) untuk menangani ketidakseimbangan kelas. Selain itu, penelitian ini tidak hanya berfokus pada pengembangan model klasifikasi, tetapi juga mengimplementasikan *Random Forest* ke dalam aplikasi berbasis website untuk mendukung deteksi dini risiko diabetes. Dengan menggunakan dataset BRFSS 2015 dengan tiga kelas, penelitian ini juga memberikan perhatian terhadap kemampuan model dalam mengenali kelas Pre-Diabetes yang merupakan salah satu kelas yang sulit diklasifikasikan. Dengan demikian, penelitian ini berfokus pada pengembangan model *Random Forest* dengan penanganan *class imbalance* menggunakan *hybrid sampling* serta implementasinya sebagai aplikasi deteksi dini risiko diabetes berbasis website.