BAB II
TINJAUAN PUSTAKA

2.1. Kajian Pustaka

Penelitian [10] membahas penggunaan metode SMOTE dengan *Random Forest* dan *Xgboost*. Fokus penelitian ini adalah penerapan metode *SMOTE* pada algoritma *Random Forest* dan *XGboost* dalam mengklasifikasikan tingkat penyakit Hepatitis C pada data dengan distribusi kelas yang tidak seimbang. Temuan dari penelitian ini menunjukkan bahwa kedua metode, yaitu *SMOTE Random Forest* dan *SMOTE XGboost* berhasil mencapai akurasi dan nilai *recall* yang melebihi 75%. Meskipun akurasi dari *Random Forest* dan *XGboost* mencapai sekitar 74%, nilai *recall* yang didapat tetap di bawah 2%.

Penelitian [11] meneliti mengenai pendekatan alternatif untuk memprediksi klasifikasi status vaksinasi Hepatitis-B menggunakan pendekatan *machine learning* seperti *Random Forest* dan *Naive Bayes.* Namun, untuk klasifikasi pada data yang tidak seimbang, algoritma cenderung bias terhadap kelas mayoritas. Untuk meningkatkan prediksi yang akurat, penelitian ini menggunakan *SMOTE*. Hasil penelitian menunjukkan bahwa penggunaan *SMOTE* pada *Random Forest* dan *Naive Bayes* meningkatkan akurasi identifikasi status non-vaksinasi Hepatitis-B sebesar 30,08% dan 26,09% secara berturut-turut, dibandingkan dengan *non-SMOTE*. *Random Forest* dengan *SMOTE* menjadi model terbaik untuk klasifikasi status vaksinasi Hepatitis-B.

Penelitian lain yang menerapkan *SMOTE* juga dilakukan pada penelitian [12] untuk mengatasi masalah ketidakseimbangan kelas. Penelitian ini menerapkan *Weighted Categorical Loss* (WCL) dan *Synthetic Minority Oversampling Technique* (*SMOTE*). Enam model *CNN state-of-the-art* dilatih dengan *Transfer Learning* (TL) pada tiga data COVID-19 yang berbeda, dengan fokus pada klasifikasi multi kelas antara COVID-19, normal, dan kasus pneumonia virus. Hasil penelitian ini menunjukkan *SMOTE* secara signifikan meningkatkan nilai *AUC* pada model *DenseNet201* dan *VGG19* dibandingkan dengan penggunaan data yang tidak seimbang.

Penelitian [13] berfokus pada meningkatkan prediksi terhadap pasien penyakit gagal jantung kronis dengan menggunakan teknik *SMOTE* dan pendekatan *machine learning*. Delapan model klasifikasi digunakan dalam penelitian ini, yaitu *Random Forest* (*RF*), *Extra Tree* (*ET*), *Naive Bayes* (*NB*), *k-Nearest Neighbor* (*KNN*), *Decision Tree J48*, *Decision Table/Naive Bayes hybrid classifier* (*DTNB*), *Optimized Forest*, dan *Alternating Decision Tree* (*ADTree*), untuk memprediksi kelangsungan hidup pasien. Hasil percobaan menunjukkan teknik *SMOTE* mengubah akurasi keluaran dari metode pengklasifikasi yang dipilih dan *DTNB* mencapai akurasi tertinggi dengan 87,08% menggunakan *SMOTE* untuk memperkirakan kelangsungan hidup pasien jantung. Semua percobaan dilakukan dalam lingkungan simulasi menggunakan alat WEKA. dengan menggunakan *SMOTE* untuk memprediksi kelangsungan hidup pasien jantung.

Pada penelitian [14] dilakukan penelitian mengenai perbandingan dua algoritma klasifikasi yaitu *C4.5* dan *k-Nearest Neighbor* (*KNN*), dan metode pra-pemrosesan *SMOTE* diterapkan dalam klasifikasi kinerja akademik mahasiswa. Hasil eksperimen menggunakan aplikasi Rapid Miner menunjukkan bahwa metode *C4.5 Decision Tree* menghasilkan performa prediksi yang lebih baik dalam hal akurasi, *recall*, dan presisi masing-masing sebesar 71,09%, 71,63%, dan 71,54% dibandingkan dengan algoritma *K-Nearest Neighbor*.

Penelitian [15] melakukan penelitian tentang penanganan pada data yang tidak seimbang pada penelitian ini menggunakan metode *oversampling* dengan teknik *SMOTE*, *Borderline-SMOTE*, dan *ADASYN*, hasil dari penelitian ini menunjukkan bahwa penggunaan teknik *oversampling* meningkatkan akurasi dari 2% hingga 11% pada beberapa data.

Penelitian [16] berfokus pada memprediksi kegagalan mesin di industri melalui algoritma pembelajaran mesin. Namun, tantangan ketidakseimbangan data pada setiap kelas menghambat kinerja model. Solusi yang diterapkan yaitu menggunakan teknik *oversampling* menggunakan *SMOTE*. Teknik *oversampling* tersebut menghasilkan sampel sintetis pada kelas minoritas. Hasil penelitian menunjukkan bahwa penggunaan *SMOTE* meningkatkan kinerja *Random Forest*

dalam mengenali kondisi kegagalan dan *non*-kegagalan mesin dengan peningkatan skor *AUC* 7,83%.

Penelitian [17] meneliti mengenai penanganan masalah data yang tidak seimbang pada data penipuan, hasil dari penelitian ini adalah teknik *oversampling* terbukti dapat meningkatkan kinerja model, dan teknik *oversampling* terbaik adalah menggunakan *SMOTE*.

Penelitian [18] mengadopsi teknik sampling *SMOTE* dan *Random Oversampling* untuk mengatasi data tidak seimbang. Teknik sampling *SMOTE* melibatkan mensintesis data baru yang mirip dengan data minoritas, sedangkan teknik sampling *ROS* melibatkan duplikasi data minoritas. Hasil penelitian menunjukkan bahwa model klasifikasi *SMOTE* dengan *Naive Bayes* memiliki akurasi yang lebih tinggi 0,61% daripada model klasifikasi *ROS* dengan *Naive Bayes*.

Penelitian [19]meneliti mengenai penggunaan metode resampling *oversampling SMOTE* pada dataset yang tidak seimbang menunjukkan bahwa model yang dilatih pada data yang di*oversampling* dapat memprediksi lebih, daripada model yang dilatih pada data yang tidak di*oversampling*. Ini menunjukkan bahwa *oversampling* dengan *SMOTE* dapat mengubah kinerja *XAI* menjadi lebih baik.

Tabel 2.1. Penelitian Terdahulu

No	Penulis	Judul Penelitian	Metode Penelitian	Masalah Penelitian	Hasil Penelitian
1	Muhamad Syukron, Rukun Santoso, Tatik Widiharih (2020)	Perbandingan Metode *SMOTE Random Forest* dan *SMOTE XGBoost* untuk Klasifikasi Tingkat Penyakit Hepatitis C pada *Imbalanced Class Data*	Menggunakan metode *SMOTE Random Forest* dan *SMOTE XGboost* dalam klasifikasi tingkat penyakit Hepatitis C pada data tidak seimbang.	Data yang digunakan dalam penelitian ini memiliki ketidakseimbangan yang dapat mempengaruhi akurasi prediksi. Oleh karena itu, diperlukan pendekatan untuk meningkatkan akurasi prediksi, salah satunya adalah menggunakan algoritma *SMOTE*.	Nilai akurasi dan *recall* dari kedua metode lebih dari 75%. Namun, akurasi *Random Forest* dan *XGBoost* sekitar 74% dengan nilai *recall* kurang dari 2%. Variabel hasil tes laboratorium memiliki pengaruh besar dalam menentukan tingkat penyakit Hepatitis C.
2	V M Putri, M Masjkur, C Suhaeni (2020)	*Performance of SMOTE in a Random Forest and Naive Bayes Classifier for Imbalancedd Hepatitis-B Vaccination Status*	Menggunakan *SMOTE* dalam *Random Forest* dan *Naive Bayes Classifier* untuk klasifikasi status vaksinasi Hepatitis-B.	Masalah dalam penelitian ini adalah terkait dengan klasifikasi status vaksinasi Hepatitis-B yang tidak seimbang. Kondisi tidak seimbang ini mengacu pada perbedaan yang signifikan dalam jumlah sampel antara kelas mayoritas dan kelas minoritas. Dalam konteks penelitian ini, kelas mayoritas mungkin adalah status vaksinasi Hepatitis-B yang sudah divaksinasi, sedangkan kelas minoritas adalah status vaksinasi Hepatitis-B yang belum divaksinasi.	Penggunaan *SMOTE* meningkatkan akurasi identifikasi status *non*-vaksinasi Hepatitis-B sebesar 30,08% dan 26,09% pada *Random Forest* dan *Naive Bayes*. Model *Random Forest* dengan *SMOTE* menjadi yang terbaik.

No	Penulis	Judul Penelitian	Metode Penelitian	Masalah Penelitian	Hasil Penelitian
3	Ekram Chamseddine, Nesrine Mansouri, Makram Soui, Mourad Abed (2022)	*Handling class imbalance in COVID-19 chest X-ray images classification: Using SMOTE and weighted loss*	Penelitian ini menggunakan teknik *deep learning*, khususnya *Convolutional Neural Networks* (*CNNs*), untuk mengklasifikasikan Gambar *X-ray* dada pasien COVID-19, dan untuk mengatasi masalah ketidakseimbangan kelas, penelitian ini menerapkan *Weighted Categorical Loss* (*WCL*) dan *SMOTE*.	Prediksi *churn* pelanggan bank dalam konteks data yang tidak seimbang. Dalam era *big data*, dengan perkembangan pesat keuangan internet dan meningkatnya tekanan pada bank untuk bersaing, bank beralih fokus pada mempertahankan pelanggan lama, sehingga penting untuk memprediksi *churn* pelanggan. Dalam penelitian ini, penulis merancang model prediksi *Borderline-SMOTE-Random Forest* untuk mengatasi masalah data yang tidak seimbang seperti data pelanggan bank.	Penggunaan *SMOTE* secara signifikan meningkatkan nilai *AUC* pada model *DenseNet201* dan *VGG19* dibandingkan dengan menggunaan data yang tidak seimbang.
4	S. Priyadarshinee, M. Panda (2022)	*Improving Prediction of Chronic Heart Failure using SMOTE and Machine Learning*	Menggunakan teknik *SMOTE* dan delapan model klasifikasi untuk memprediksi kelangsungan hidup pasien penyakit gagal jantung kronis.	Masalah penelitian ini adalah prediksi kelangsungan hidup pasien penyakit jantung menggunakan model *machine learning*. Penelitian ini bertujuan untuk meningkatkan prediksi kelangsungan hidup pasien penyakit jantung dengan memanfaatkan model *machine learning*.	Teknik *SMOTE* meningkatkan akurasi keluaran klasifikasi dan model *DTNB* mencapai akurasi tertinggi yaitu 87,08% untuk memprediksi kelangsungan hidup pasien jantung.
5	U. Pujianto, W. Agung Prasetyo, A.	*Students Academic Performance Prediction with k-*	Membandingkan *C4.5* dan *K-Nearest Neighbor* (*KNN*)	Masalah penelitian ini adalah prediksi kinerja akademik siswa dari usia dini akan memudahkan guru	Metode *C4.5 Decision Tree* menghasilkan performa prediksi yang lebih baik

No	Penulis	Judul Penelitian	Metode Penelitian	Masalah Penelitian	Hasil Penelitian
	Rakhmat Taufani (2020)	*Nearest Neighbor and C4.5 on SMOTE-balanced data*	menggunakan metode *SMOTE* dalam klasifikasi kinerja akademik mahasiswa.	untuk memberikan bantuan kepada siswa yang memiliki kemampuan akademik di bawah rata-rata kelas atau kesulitan dalam mengikuti proses pembelajaran di kelas.	dengan nilai akurasi, *recall*, dan presisi masing-masing sebesar 71,09%, 71,63%, dan 71,54% dibandingkan dengan *K-Nearest Neighbor*.
6	Nurheri Cahyana, Siti Khomsah,Agus Sasmito Aribowo (2019)	*Improving Imbalanced Dataset Classification Using Oversampling and Gradient Boosting*	Penelitian ini menggunakan metode *oversampling* dengan teknik *SMOTE*, *Borderline-SMOTE*, dan *ADASYN* untuk mengatasi ketidakseimbangan dataset. *Gradient Boosting* digunakan sebagai model klasifikasi.	Penelitian ini bertujuan untuk mengatasi masalah klasifikasi data yang tidak seimbang dengan menggunakan metode *oversampling* dan *Gradient Boosting*. Dalam penelitian ini, dicoba tiga teknik *oversampling* (*SMOTE*, *Borderline-SMOTE*, dan *ADASYN*) untuk mengatasi ketidakseimbangan data dan melihat dampaknya terhadap akurasi klasifikasi. Penelitian ini juga mencoba mengukur performa klasifikasi menggunakan metrik akurasi, *recall*, *precision*, *F1-Score*, dan AUC.	Eksperimen menunjukkan bahwa teknik *oversampling* meningkatkan akurasi dari 2% hingga 11% untuk beberapa data seperti *Mammography*, *Liver Disorders*, *Diabetes (Pima Indian)*, *Indian Liver*, *Habberman*, dan *Immunotherapy*.

No	Penulis	Judul Penelitian	Metode Penelitian	Masalah Penelitian	Hasil Penelitian
7	Sashank Sridhar, Sowmya Sanagavarapu (2021)	*Handling Data Imbalance in Predictive Maintenance for Machines using SMOTE-based Oversampling*	Penelitian ini fokus pada pengembangan algoritma pembelajaran mesin untuk memprediksi kegagalan mesin dalam lingkungan industri. Data sintetis digunakan dalam model pemeliharaan prediktif yang mencerminkan kegagalan nyata yang terjadi di industri. Masalah ketidakseimbangan kelas data ditangani dengan menerapkan teknik *oversampling* berbasis *SMOTE*.	Penelitian ini bertujuan untuk mengatasi masalah ketidakseimbangan data dalam model prediksi pemeliharaan prediktif untuk mesin industri. Ketidakseimbangan kelas data dapat menghambat kinerja algoritma pembelajaran mesin, dan masalah ini ditangani dengan menerapkan teknik *oversampling* berbasis *SMOTE*.	Hasil eksperimen menunjukkan bahwa penggunaan teknik *SMOTE* meningkatkan skor *AUC* sebesar 7,83%, yang menghasilkan peningkatan kinerja *Random Forest* dalam membedakan antara contoh kegagalan dan contoh *non*-kegagalan pada mesin. Dengan menerapkan teknik *SMOTE*, kinerja *Random Forest* dalam mengidentifikasi instansi kegagalan dan *non*-kegagalan meningkat.
8	Raneem Qaddoura and Mariam M. Biltawi (2022)	*Improving Fraud Detection in An Imbalanced Class Distribution Using Different Oversampling Techniques*	Ketidakseimbangan data ini dapat menyebabkan model pembelajaran mesin yang dilatih pada data ini menjadi bias	Menggunakan lima teknik *oversampling* yaitu: *SMOTE*, *ADASYN*, *Borderline1 SMOTE*, B*orderline2 SMOTE*, dan *Support Vector Machine SMOTE* (*SVM SMOTE*) untuk mengatasi masalah ketidak seimbangan data.	Teknik *oversampling* tebukti dapat meningkatkan kinerja model , dan Teknik *oversampling* terbaik adalah menggunakan *SMOTE*

No	Penulis	Judul Penelitian	Metode Penelitian	Masalah Penelitian	Hasil Penelitian
			terhadap transaksi *non-fraud*.		
9	Nur Silviyah Rahmi, Ni Wayan Surya Wardhani, Maria Bernadetha Mitakda, Regina Syahla Fauztina, dan Imelda Salsabila (2022)	*SMOTE Classification and Random Oversampling Naive Bayes in Imbalanced Data: (Case Study of Early Detection of Cervical Cancer in Indonesia)*	Mengadopsi teknik sampling *SMOTE* dan *Random Oversampling* (*ROS*) untuk mengatasi data tidak seimbang	Keberadaan data tidak seimbang membuat kinerja metode klasifikasi dalam *machine learning* menurun	Model klasifikasi *SMOTE Naive Bayes* memiliki akurasi yang lebih tinggi 0,61% daripada model klasifikasi *ROS* dengan *Naive Bayes.*
10	Aum Patil, Aman Framewala, dan Faruk Kazi (2022)	*Explainability of SMOTE Based Oversampling for Imbalanced Dataset Problems*	Menggunakan metode resampling *oversampling SMOTE*.	Keberadaan data tidak seimbang dapat membuat kinerja model pembelajaran mesin menurun	Model yang dilatih pada data yang diberlakukan *oversampling* dapat memprediksi lebih daripada model yang dilatih pada data yang tidak seimbang. Hasil penelitian menunjukkan bahwa *oversampling* menggunakan *SMOTE* dapat meningkatkan kinerja *XAI*.

Berdasarkan penelitian terdahulu pada Tabel 2.1 diketahui bahwa metode *SMOTE* dapat digunakan dalam penanganan *imbalanced data* yang kemudian menghasilkan nilai akurasi yang cukup bagus. Penelitian ini menerapkan metode *Random Forest* dan *C4.5* pada analisis kinerja model pada data yang tidak seimbang dan sebagai perbedaan penelitian tugas akhir ini terletak pada studi kasus yang diterapkan.

2.2. Landasan Teori

2.2.1. *Imbalanced Dataset*

Setiap kumpulan data dengan distribusi kelas yang tidak merata secara teknis dianggap mengalami ketidakseimbangan kelas [1]. Ketidakseimbangan kelas terjadi ketika jumlah data yang mewakili suatu kelas jauh lebih rendah dibandingkan dengan kelas lainnya. Kelas yang memiliki jumlah lebih sedikit dikenal sebagai kelas minoritas, sedangkan kelas lainnya disebut sebagai kelas mayoritas. Umumnya, kelas mayoritas memiliki dominasi yang lebih signifikan. Akibatnya, *classifier* cenderung membuat prediksi yang lebih condong ke arah kelas mayoritas sambil mengabaikan kelas minoritas.

Berbagai teknik telah diusulkan dan dibahas untuk mengatasi masalah data yang tidak seimbang. Beberapa solusi yang umum mencakup pendekatan pada tingkat data, pendekatan pada algoritma, sensitivitas terhadap biaya, dan penggunaan ansambel pembelajaran [20]. Dalam praktiknya, pendekatan pada tingkat data melibatkan tindakan seperti *undersampling*, *oversampling*, atau kombinasi keduanya.

2.2.2. Pendekatan Level Data

Pada pendekatan level data, ada berbagai teknik *resampling* dan sintesis data yang digunakan untuk mengatasi ketidakseimbangan distribusi kelas dalam data latih. Namun, di tingkat algoritma, pendekatan utamanya adalah melakukan penyesuaian pada operasi algoritma yang sudah ada, sehingga pengklasifikasi (*classifier*) lebih responsif terhadap klasifikasi kelas minoritas [21]. Teknik *resampling* adalah salah satu teknik *preprocessing* di mana distribusi data diseimbangkan kembali untuk mengurangi efek distribusi kelas tidak seimbang dalam proses pembelajaran [22] teknik *resampling* menyamakan distribusi kelas secara algoritmik untuk meningkatkan *imbalance ratio* [23] dan mengurangi efek distribusi kelas tidak seimbang dalam proses pembelajaran *machine learning*. Teknik *resampling* dapat dilakukan dengan metode *undersampling*, *oversampling*, dan gabungan keduanya (*hybrid*).

2.2.3. *Oversampling*

Oversampling adalah salah satu metode untuk mengatasi ketidakseimbangan kelas dalam dataset dengan cara menambahkan lebih banyak sampel dari kelas minoritas. *Oversampling* dapat meningkatkan jumlah data dari kelas minoritas sehingga distribusi antara kelas mayoritas dan minoritas menjadi lebih seimbang [5]. Teknik *oversampling* yang akan digunakan dalam penelitian ini adalah *SMOTE*.

Metode *Synthetic Minority Oversampling Technique* (*SMOTE*) adalah salah satu bentuk dari teknik oversampling. Diciptakan pertama kali oleh Nithes V. Chawla, pendekatan ini berfokus pada pembuatan replika data pada kelas minoritas. Replika data ini juga dikenal sebagai data sintetis.

SMOTE bekerja dengan mencari k tetangga terdekat (*K-Nearest Neighbors*) untuk setiap data pada kelas minoritas. Kemudian, data sintetis dibuat dengan jumlah persentase tertentu dari duplikasi data minoritas yang diinginkan. Pemilihan *K-Nearest Neighbors* ini dilakukan secara acak. *SMOTE* melakukan sintesisasi atau meng-*generate* data sampel baru berupa data vektor yang diambil dari kelas minoritas. Data yang digunakan untuk *SMOTE* adalah data yang sudah dilakukan f*eature extraction*. Sintesis yang dibuat ditentukan dengan nilai k (tetangga terdekat). Berikut langkah-langkah atau algoritma yang dilakukan oleh *SMOTE* [5]:

Kode Program 2.1. Algoritma SMOTE(T, N, k)

```
Algoritma SMOTE(T, N, k)
Input: Jumlah sampel kelas minoritas T; Persentase SMOTE N%;
Jumlah tetangga terdekat k
Output: (N/100) * T sampel kelas minoritas sintetis
1. (* Jika N kurang dari 100%, acak sampel kelas minoritas karena hanya sebagian
   persentase acak dari mereka yang akan di-SMOTE. *)
2. If N < 100
3.    then acak sampel kelas minoritas sebanyak T
            T = (N/100) * T
            N = 100
4. endif
```

```
5.  N = (int)(N/100) (∗ Jumlah SMOTE diasumsikan dalam kelipatan integral dari 100.
    ∗)
6.  k = Jumlah tetangga terdekat
7.  numattrs = Jumlah atribut
8.  Sampel[ ][ ]: array untuk sampel kelas minoritas asli
9.  newindex: menyimpan hitungan jumlah sampel sintetis yang dihasilkan, diinisialisasi
    menjadi 0
10. Sintetis[ ][ ]: array untuk sampel sintetis
    (∗Hitung k tetangga terdekat hanya untuk setiap sampel kelas minoritas∗)
11. for i ← 1 hingga T
12.             Hitung k tetangga terdekat untuk i, dan simpan indeksnya dalam nnarray
13.             Populate(N, i, nnarray)
14. endfor
15. Populate(N, i, nnarray) (∗ Fungsi untuk menghasilkan sampel sintetis. ∗)
16. while N = 0
17.         (*Pilih angka acak antara 1 dan k, sebutlah nn. Langkah ini memilih salah
            satu dari k tetangga terdekat dari i*).
18.         for attr ← 1 hingga numattrs
19.                 Hitung: dif = Sample[nnarray[nn]][attr] − Sample[i][attr]
20.                 Hitung: gap = angka acak antara 0 dan
21.                 Sintetis[newindex][attr] = Sample[i][attr] + gap ∗ dif
22.         endfor
23.         newindex++
24.         N = N − 1
25. endwhile
26. return (∗ Akhir dari Populate. ∗)
```

Rumus untuk membuat data sintesis baru ditunjukkan pada formula 2.1 Di mana, x_{new} adalah data sintesis baru, xi adalah data sampel asli kelas minoritas, $\hat{x}$ adalah salah satu nilai dari *K-Nearest Neighbors*, δ merupakan nilai random dari 0 hingga 1. Lakukan sintesis ini berulang kali hingga jumlah kelas minoritas menyerupai kelas mayoritas.

$$x_{new} = x_i + (\hat{x} - x_i) \times \delta \tag{2.1}$$

2.2.4. *Imbalance Ratio*

Imbalance Ratio (*IR*) merujuk pada perbandingan antara jumlah sampel dalam kelas mayoritas dan jumlah sampel dalam kelas minoritas dalam sebuah dataset [1]. Beberapa peneliti menganggap bahwa kumpulan data mengalami masalah ketidakseimbangan jika IR lebih tinggi dari 3 [24]. Namun derajat ketidakseimbangannya dapat dibagi ke dalam kelompok-kelompok yang disajikan pada Tabel 2.2.

Klasifikasi optimal berhasil dicapai saat distribusi kelas seimbang dan jumlah sampel mencukupi untuk mewakili baik kelas mayoritas maupun minoritas. Hal ini penting karena sampel pelatihan yang lebih mewakili memberikan pemahaman yang lebih mendalam yang berkontribusi pada proses pembelajaran. Jika kondisi ini tidak terpenuhi, kinerja klasifikasi dapat menurun. Pada kasus dataset dengan dua kelas, perhitungan *imbalance ratio* bisa dijabarkan pada formula 2.2 [1].

$$IR = \frac{Jumlah\ Sampel\ Kelas\ Mayoritas}{Jumlah\ Sampel\ Kelas\ Minoritas} \tag{2.2}$$

Perhitungan proporsi kelas minortitas atas kelas mayoritas diberikan pada formula 2.3 [25].

$$minoritas\% = \frac{Jumlah\ Sampel\ Kelas\ Minoritas}{Jumlah\ Sampel\ Kelas\ Mayoritas}\ X\ 100 \tag{2.3}$$

Proporsi kelas minoritas dapat digolongkan berdasarkan derajat ketidakseimbangan datanya. Derajat ketidakseimbangan datanya mengacu pada seberapa besar perbedaan antara jumlah sampel pada kelas mayoritas (kelas yang lebih banyak) dan kelas minoritas (kelas yang lebih sedikit) dalam dataset klasifikasi. Proporsi kelas minoritas menggambarkan persentase kelas minoritas dari total dataset[24].

Tabel 2. 2 Kategori Tingkat Ketidakseimbangan Data

Derajat Ketidak Seimbangan data	Proporsi Kelas Minoritas
Mild	20-40% dari dataset
Moderate	1-20% dari dataset
Extreme	<1% dari dataset

2.2.5. Pendekatan Level Data

Pendekatan level data mencakup berbagai terknik resampling dan sintesis data. terknik resampling merupakan salah satu teknik preprocessing di mana distribusi data diseimbangkan kembali untuk mengurangi efek distribusi kelas tidak seimbang dalam proses pembelajaran[23] teknik resampling menyamakan distribusi kelas secara algoritmik untuk meningkatkan *imbalance ratio* [24] dan mengurangi efek distribusi kelas tidak seimbang dalam proses pembelajaran *machine learning*. Teknik resampling dapat dilakukan dengan metode undersampling, oversampling, dan gabungan keduanya (hybrid).

2.2.6. *Robust Scaller*

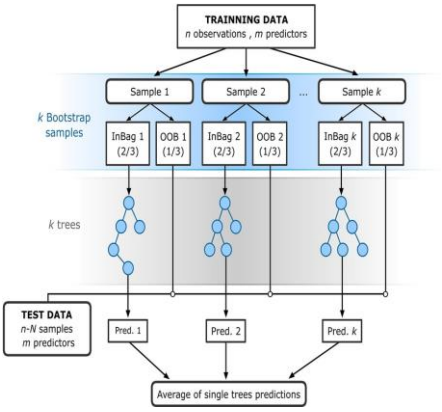
Robust Scaller adalah salah satu metode penskalaan fitur pada data yang digunakan dalam prapemrosesan data untuk analisis statistik atau pembelajaran mesin. Dengan menggunakan median dan kuartil rumus *robust scaller* dengan persamaan pada 2.7

$$X' = \frac{\mathrm{X} - \mathrm{X_{Q1}}}{\mathrm{X_{Q3}} - \mathrm{X_{Q1}}} \tag{2.7}$$

Keterangan	:	
X	:	Nilai awal
$\mathrm{X_{Q1}}$	:	Kuartil pertama dari data
$\mathrm{X_{Q3}}$	:	Kuartil ketiga dari data

2.2.7. *Random Forest*

Random Forest merupakan metode *ensemble* yang memanfaatkan kombinasi dari berbagai metode klasifikasi dari pemilah tunggal yang tidak stabil. Melalui proses pengambilan keputusan (*voting*), berbagai metode tersebut digabungkan untuk memberikan prediksi klasifikasi akhir yang lebih akurat [26]. Berikut merupakan alur algoritma *Random Forest* pada Gambar 2.1 [27].



Gambar 2. 1 Alur Algoritma Random Forest

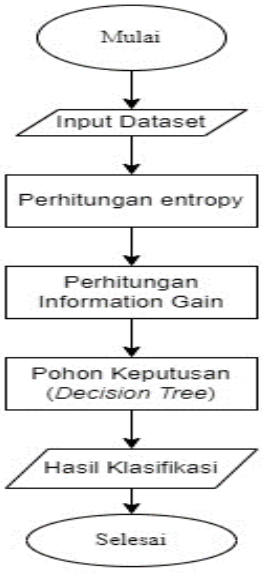
Random Forest terdiri atas kumpulan *Decision Tree* yang digunakan untuk mengklasifikasikan data ke dalam kelas-kelas tertentu[28]. *Random Forests* menggunakan pendekatan pemilihan atribut secara acak untuk membentuk beberapa pohon keputusan dengan atribut yang berbeda. Dalam *Decision Tree* konvensional, data uji diuji hanya pada satu pohon yang telah dibangun. Namun, pada *Random Forests*, data uji diuji pada semua pohon yang telah dibangun, dan klasifikasi dilakukan dengan memilih hasil mayoritas dari pemilihan yang dilakukan oleh setiap pohon [29].

2.2.8. Algoritma *C4.5*

Salah satu metode yang bisa dipakai untuk konstruksi pohon keputusan (*decision tree*) adalah mengunakan metode algoritma *C4.5* [30]. Model algoritma *C4.5* memiliki bentuk serupa pohon, dimana terdapat simpul internal yang merepresentasikan atribut-atribut, cabang-cabang menggambarkan hasil dari pengujian atribut, dan setiap ujung cabang (daun) mewakili kelas tertentu. Secara umum, data dalam struktur pohon keputusan direpresentasikan dalam format tabel yang memuat atribut-atribut. Atribut-atribut ini berperan sebagai parameter yang berfungsi sebagai kriteria dalam pembentukan pohon keputusan [31].

Dalam algoritma *C4.5*, dilakukan pemilihan solusi terbaik dengan menghitung dan membandingkan rasio keuntungan (*gain ratio*) hingga mencapai simpul-simpul pada level berikutnya. Algoritma *C4.5* menerima masukan berupa

contoh latihan (*training samples*). Tahapan dalam pembuatan pohon keputusan menggunakan algoritma *C4.5* pada Gambar 2.2 [32].



Gambar 2. 2 Alur Algoritma *C4.5*

Berikut merupakan alur langkah-langkah dari cara kerja algoritma *C4.5*:

1. Persiapan data latih dapat dilakukan dengan memperoleh data historis sebelumnya yang telah terkelompokkan ke dalam kelas-kelas yang ditentukan.
2. Menentukan akar pohon, penentuan akar pohon dapat dilakukan dengan menghitung nilai *gain* yang tertinggi dari setiap atribut atau berdasarkan nilai *entropi* terendah. Dalam konteks *data mining*, *entropy* mengacu pada ukuran heterogenitas atau ketidakpastian dalam kumpulan data [33]. *Entropy* digunakan sebagai parameter untuk mengukur tingkat ketidakpastian dalam pengambilan keputusan. Semakin heterogen atau tidak teratur kumpulan data, semakin tinggi nilai *entropy*-nya. Sebelumnya, perlu dihitung nilai *entropi* dengan menggunakan rumus pada formula 2.8

$$Entropy\ (S)\ = \sum_{j=i}^{m} -pi\ *\ log_2 pi \tag{2.8}$$

Keterangan	:	
i	:	Himpunan kasus
m	:	Jumlah partisi i
pi	:	Atribut

3. Menghitung nilai *gain* pada formula 2.9

$$gain\ (S.A)\ =\ Entropy(s) - \sum_{I-1}^{N} \frac{|si|}{|s|} * Entropy(si) \quad (2.9)$$

Keterangan	:	
S	:	Himpunan kasus
A	:	Fitur
N	:	Jumlah atribut partisi A
|si|	:	Proporsi terhadap s
s	:	Jumlah kasus dalam s

4. Ulangi langkah ke-2 hingga semua *record* terpartisi. Proses partisi pohon keputusan akan berhenti disaat:
   a) Semua tupel dalam *record* dalam simpul *m* mendapat kelas yang sama
   b) Tidak ada atribut dalam *record* yang dipartisi lagi
   c) Tidak ada *record* didalam cabang yang kosong

2.2.9. Metrik Evaluasi

Metrik Evaluasi digunakan untuk mengevaluasi seberapa baik model bekerja pada data uji [36]. Hasil metrik evaluasi model klasifikasi berisikan berbagai metrik seperti *precision* (presisi), *recall* (sensitivitas), *F1-score*, dan lainnya. Berikut beberapa persamaan yang ada pada metrik evaluasi:

Akurasi berisi nilai yang menentukan jumlah prediksi yang benar atas jumlah prediksi oleh model. Rumus dari akurasi ditunjukkan pada formula 2.10

$$\text{Akurasi}\ =\ \frac{TP + TN}{TP + FP + TN + FN} \quad (2.10)$$

Presisi berisi nilai prediksi benar positif terhadap keseluruhan data yang diprediksi positif. Rumus dari Presisi ditunjukkan pada formula 2.10

$$Presisi = \frac{TP}{(TP + FP)} \quad (2.11)$$

Recall menggambarkan kapabilitas dalam mengidentifikasi seluruh contoh yang relevan dalam kumpulan data, sementara presisi mengindikasikan persentase titik data yang oleh model dianggap relevan dan sesuai dengan apa yang memang benar-benar relevan. Rumus *recall* ditunjukkan pada formula 2.12

$$Recall = \frac{TP}{(TP + FN)} \tag{2.12}$$

F1 Score merupakan nilai rata-rata dari presisi dan *recall* yang memasukkan perhitungan kedua metrik tersebut kedalam persamaan 2.13

$$F1 = 2 * \frac{precision * recall}{Precision + recall} \tag{2.13}$$

Metrik evalusi, dalam bentuk tabel matriks seperti yang ditunjukkan pada Tabel 2.3, menunjukkan kinerja model klasifikasi pada sekumpulan data uji dengan nilai sebenarnya yang diketahui.

Tabel 2. 3 *Confusion Matriks*

Confusion Matriks		Actual	
		Positive	Negative
Predicted	Positive	TP	FP
	Negative	TN	FN

Keterangan :

TP : True Positive

TN : True Negative

FP : False Positive

FN : False Negative