PENGARUH PENANGANAN KETIDAKSEIMBANGAN KELAS PADA *DATASET* PENYAKIT STROKE TERHADAP PERFORMA ALGORITMA *RANDOM FOREST*

SKRIPSI

Oleh :
FIRDA ARINDA EKA PUTRI
NIM. 210605110115

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2024

PENGARUH PENANGANAN KETIDAKSEIMBANGAN KELAS PADA *DATASET* PENYAKIT STROKE TERHADAP PERFORMA ALGORITMA *RANDOM FOREST*

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
FIRDA ARINDA EKA PUTRI
NIM. 210605110115

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

HALAMAN PERSETUJUAN

PENGARUH PENANGANAN KETIDAKSEIMBANGAN KELAS PADA *DATASET* PENYAKIT STROKE TERHADAP PERFORMA ALGORITMA *RANDOM FOREST*

SKRIPSI

Oleh :
FIRDA ARINDA EKA PUTRI
NIM. 210605110115

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 03 Desember 2024

Pembimbing I,	Pembimbing II,
Okta Qomaruddin Aziz, M.Kom	Nur Fitriyah Ayu Tunjung Sari, M.Cs
NIP. 19911019 201903 1 013	NIP. 19911226 202012 2 001

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT, IPU
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

PENGARUH PENANGANAN KETIDAKSEIMBANGAN KELAS PADA *DATASET* PENYAKIT STROKE TERHADAP PERFORMA ALGORITMA *RANDOM FOREST*

SKRIPSI

Oleh :
FIRDA ARINDA EKA PUTRI
NIM. 210605110115

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 12 Desember 2024

Susunan Dewan Penguji

Ketua Penguji	: Dr. Zainal Abidin, M.Kom NIP. 19760613 200501 1 004	( )
Anggota Penguji I	: Dr. Yunifa Miftachul Arif, M.T NIP. 19830616 201101 1 004	( )
Anggota Penguji II	: Okta Qomaruddin Aziz, M.Kom NIP. 19911019 201903 1 013	( )
Anggota Penguji III	: Nur Fitriyah Ayu Tunjung Sari, M.Cs NIP. 19911226 202012 2 001	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Irwan Fachrul Kurniawan, M.MT, IPU
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Firda Arinda Eka Putri
NIM	: 210605110115
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Pengaruh Penanganan Ketidakseimbangan Kelas Pada *Dataset* Penyakit Stroke Terhadap Performa Algoritma *Random Forest*

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 12 Desember 2024
Yang membuat pernyataan,

Firda Arinda Eka Putri
NIM.210605110115

MOTTO

"Bahaya Itu Nyata Takut Itu Pilihan
Jadi
Ambil Resikonya Atau Hidup Seperti Itu Selamanya"

"Jika Kamu Tak Sanggup Menahan Lelahnya Belajar

Maka Kamu Harus Sanggup Menahan Perihnya Kebodohan"

(Imam Syafi'i)

HALAMAN PERSEMBAHAN

Puji Syukur atas kehadirat Allah Subhanahu wa ta'ala, karena berkat rahmat dan petunjuk-Nya, sehingga penulis dapat menyelesaikan skripsi ini dengan baik dan lancar. Shalawat serta salam kepada Rasulullah Shallallahu 'alaihi wasallam, yang telah membawa kita dari zaman jahiliyah menuju addinul Islam. Skripsi ini tidak akan selesai tanpa adanya kontribusi dan dukungan dari berbagai pihak. Oleh karena itu, penulis mempersembahkan skripsi ini kepada seluruh pihak yang telah berjasa dalam pengerjaan penelitian ini. Karya ini penulis persembahkan kepada:

1. Keluarga tercinta Ayah Handik Tangkas Prasetya dan Ibu Yuliati selaku orang tua penulis, serta Nenek Siti dan Kakek Sahari yang sangat penulis cintai dan tidak pernah berhenti dalam memberikan semangat, dukungan maupun doa kepada penulis hingga penulis dapat menyelesaikan skripsi ini dengan baik dan lancar.
2. Bapak Okta Qomaruddin Aziz, M.Kom selaku dosen pembimbing I serta Ibu Nur Fitriyah Ayu Tunjung Sari, M.Cs selaku dosen pembimbing II atas segala pelajaran yang diberikan kepada penulis yang banyak kekurangannya ini dan senantiasa membimbing, memberikan semangat, memberi arahan dan masukan, serta membantu penulis dalam mengerjakan dan menyelesaikan skripsi ini.

KATA PENGANTAR

Assalamualaikum Wr.Wb

Alhamdulillah, puji dan syukur kehadirat Allah SWT yang telah melimpahkan segala rahmat dan hidayahNya, serta memberikan kelancaran kepada penulis sehingga dapat menyelesaikan skripsi ini dengan baik. Sholawat serta salam semoga senantiasa tercurah kepada junjungan kita, Nabi Muhammad SAW. Penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu dalam penyusunan skripsi ini. Oleh karena itu, dengan ini saya mengucapkan terima kasih kepada:

1. Prof. Dr. M. Zainuddin, M.A selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Sri Harini, M.Si selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Ir. Fachrul Kurniawan, M.MT, IPU selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Okta Qomaruddin Aziz, M.Kom selaku Dosen Pembimbing I yang telah memberi arahan, bimbingan, dan dorongan selama penulisan skripsi ini sehingga penulis dapat menyelesaikan skripsi ini dengan tepat waktu.
5. Nur Fitriyah Ayu Tunjung Sari, M.Cs selaku Dosen Pembimbing II yang telah membimbing, memberikan arahan serta masukan kepada penulis sehingga penulis dapat menyelesaikan skripsi ini.

6. Dr. Zaenal Abidin, M.Kom selaku Penguji I dan Dr Yunifa Miftachul Arif, M.T selaku penguji II yang telah memberikan saran dan kritik selama proses pengujian skripsi ini.
7. Nia Faricha S, Si., selaku admin Program Studi Teknik Informatika yang selalu sabar memberikan informasi, membantu, dan memberikan arahan selama perkuliahan dan proses penulisan skripsi ini
8. Segenap dosen, laboran, dan jajaran staff Program Studi Teknik Informatika yang telah memberikan ilmu, pengetahuan, dan dukungan selama penulis menjalani studi hingga selesainya skripsi ini
9. Keluarga penulis yaitu Ibu Yuliati, Ayah Handik Tangkas Prasetya, Nenek Siti dan Kakek Sahari yang selalu memberikan do'a dan usaha sehingga penulis bisa melaksanakan kuliah dengan baik hingga selesai.
10. Krisna Maulana yang telah menjadi *support system* terbaik mulai dari awal hingga akhir. Terima kasih atas ilmunya, bimbingannya, kesabarannya serta semangatnya yang senantiasa menasehati dan menguatkan penulis, terutama pada saat penulis merasa down.
11. Teman terdekat penulis, Nafiah Nur Muttaqin yang selalu bersama penulis mulai awal perkuliahan hingga penyusunan skripsi ini. Terimakasih sudah selalu ada, memberi semangat dan semua bantuan yang telah diberikan ketika penulis sedang tidak sehat atau sedang dalam keadaan sedih.
12. Teman dekat penulis "Mencoba Kuat Sampai Tamat", Nafiah, Wanda dan Arneiza, kehadiran kalian memberikan warna dalam perjalanan ini, dengan segala bantuan, perhatian, dan dukungan yang tak pernah kalian ragu untuk

berikan. Bersama kalian, penulis merasa tidak pernah sendirian, bahkan dalam situasi yang sulit. Terima kasih telah menjadi teman yang selalu ada dan penyemangat yang luar biasa

13. Seluruh warga Teknik Informatika khususnya angkatan 2021 “Aster” yang telah memberikan kehangatan, motivasi, semangat, dan dukungan kepada penulis.

Penulis menyadari dalam penulisan skripsi ini tidak luput dari kesalahan yang jauh dari kata sempurna. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun sehingga skripsi ini dapat lebih dikembangkan.

Malang, 03 Desember 2024

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Putri, Firda Arinda Eka. 2024. **Pengaruh Penanganan Ketidakseimbangan Kelas Pada *Dataset* Penyakit Stroke Terhadap Performa Algoritma *Random Forest*.** Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Okta Qomaruddin Aziz, M.Kom, (II) Nur Fitriyah Ayu Tunjung Sari, M.Cs.

.

Kata Kunci: ADASYN, Ketidakseimbangan kelas, Klasifikasi, *Random Forest*, *Random Over Sampling*, SMOTE, Stroke

Ketidakseimbangan kelas adalah kondisi di mana jumlah data pada kelas minoritas lebih kecil dibandingkan dengan kelas mayoritas. Dampak dari ketidakseimbangan kelas dalam dataset adalah terjadinya kesalahan klasifikasi pada kelas minoritas, sehingga dapat memengaruhi kinerja klasifikasi. Salah satu contoh data yang memiliki ketidakseimbangan yakni *dataset* penyakit stroke, yang memiliki perbedaan signifikan antara jumlah data yang mengindikasikan terjadinya stroke (kelas minoritas) dan data yang tidak mengindikasikan stroke (kelas mayoritas). Penelitian ini menggunakan teknik resampling yaitu SMOTE, ADASYN, dan *Random Over Sampling* (ROS) untuk mengatasi masalah ketidakseimbangan kelas yang dikombinasikan dengan algoritma klasifikasi *Random Forest*. Tujuan penelitian ini adalah untuk mengetahui pengaruh penanganan ketidakseimbangan kelas pada dataset terhadap kinerja klasifikasi. Pengujian dilakukan pada beberapa pengaturan parameter, dan berdasarkan hasil klasifikasi, metode SMOTE memberikan hasil terbaik dengan akurasi 87% dan F1-score 86%. Metode ROS menghasilkan akurasi yang sama (87%), namun performanya sedikit lebih rendah dibandingkan SMOTE dengan *F1-Score* 85%, sementara ADASYN menunjukkan akurasi terendah (84%) dan *F1-score* 82%. Hasil uji coba menunjukkan bahwa SMOTE adalah metode *balancing* data yang paling efektif dalam meningkatkan kinerja klasifikasi, khususnya dalam mendeteksi kelas minoritas.

ABSTRACT

Putri, Firda Arinda Eka. 2024. **The Impact of Class Imbalance Handling on Stroke Disease Dataset Performance Using the Random Forest Algorithm.** Thesis. Department of Informatics Engineering, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University Malang. Supervisors : (I) Okta Qomaruddin Aziz, M.Kom, (II) Nur Fitriyah Ayu Tunjung Sari, M.Cs.

Keywords: ADASYN, *Class Imbalance, Classification, , Random Forest, Random Over Sampling, SMOTE, Stroke*

.

Class imbalance is a condition where the amount of data in the minority class is smaller than the majority class. The impact of class imbalance in a dataset is the misclassification of the minority class, which can affect classification performance. One example of data that has imbalance is the stroke disease dataset, which has a significant difference between the amount of data that indicates a stroke (minority class) and data that does not indicate a stroke (majority class). This research uses resampling techniques namely SMOTE, ADASYN, and Random Over Sampling (ROS) to overcome the problem of class imbalance combined with the Random Forest classification algorithm. The purpose of this study is to determine the effect of handling class imbalance on the dataset on classification performance. Tests were conducted on several parameter settings, and based on the classification results, the SMOTE method gave the best results with 87% accuracy and 86% F1-score. The ROS method produced similar accuracy (87%), but performed slightly lower than SMOTE with an F1-score of 85%, while ADASYN showed the lowest accuracy (84%) and F1-score of 82%. The experimental results show that SMOTE is the most effective data balancing method in improving classification performance, especially in detecting minority classes.

مستخلص البحث

فوتري، فيردا أريندا إيكا. 2024. تأثير التعامل مع عدم التوازن الطبقي على مجموعة بيانات مرض السكتة الدماغية على أداء خوارزمية الغابة العشوائية. البحث الجامعي. قسم الهندسة المعلوماتية، كلية العلوم والتكنولوجيا بجامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف الأول: أوكتا قمر الدين عزيز، الماجستير. المشرف الثاني: نور فطرية أيو تونجونج ساري، الماجستيرة.

الكلمات الرئيسية: أخذ عينات اصطناعية تكيفية، عدم توازن طبقي، تصنيف، أخذ عينات أقليات اصطناعية، سكتة دماغية، غابة عشوائية، أخذ عينات زائدة عشوائية.

عدم التوازن الطبقي هو حالة تكون فيها كمية البيانات في فئة الأقلية أصغر مقارنة بفئة الأغلبية. يتمثل تأثير عدم التوازن الطبقي في مجموعة البيانات في حدوث أخطاء في التصنيف عند فئات الأقليات، بحيث يمكن أن يؤثر على أداء التصنيف. أحد الأمثلة على البيانات التي تحتوي على خلل في التوازن هو مجموعة بيانات مرض السكتة الدماغية، والتي لها فرق كبير بين كمية البيانات التي تشير إلى حدوث السكتة الدماغية (فئة الأقلية) والبيانات التي لا تشير إلى السكتة الدماغية (فئة الأغلبية). استخدم هذا البحث تقنيات إعادة العينات، وهي *SMOTE* و *ADASYN* و *ROS* للتغلب على مشكلة عدم التوازن الطبقي مع خوارزمية تصنيف الغابات العشوائية. الهدف من هذا البحث هو تحديد تأثير التعامل مع اختلالات الفئات في مجموعات البيانات على أداء التصنيف. تم إجراء الاختبار على عدة إعدادات للمعلمات، وبناء على نتائج التصنيف، أعطت طريقة أخذ عينات الأقليات الاصطناعية (*SMOTE*) أفضل النتائج بدقة 87٪ ودرجة ف1 بنسبة 86٪. أنتجت طريقة ، أخذ عينات زائدة عشوائية (*ROS*) نفس الدقة (87٪)، لكن أدائها كان أقل قليلا من أداء *SMOTE* مع درجة ف1 بنسبة 85%، بينما أظهرت طريقة أخذ عينات اصطناعية تكيفية (*ADASYN*) أدنى دقة (84٪) ودرجة ف1 بنسبة 82٪. أظهرت نتائج الاختبار أن *SMOTE* هي أكثر طرق موازنة البيانات فعالية في تحسين أداء التصنيف، خاصة في الكشف عن فئات الأقليات.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Seiring perkembangan teknologi, keberadaan data semakin meningkat dan akses terhadap data semakin mudah. Hal ini menjadikan data sebagai dasar untuk pemodelan dan analisis berbasis data. Namun, data yang diperoleh seringkali belum dalam bentuk yang optimal untuk digunakan, terutama pada kumpulan data besar. Salah satu tantangan utama adalah ketidakseimbangan kelas dalam *dataset*, di mana distribusi kelas tidak merata sehingga memengaruhi performa algoritma klasifikasi(Magnolia dkk., 2022).

Ketidakseimbangan data dapat dianggap sebagai masalah besar, khususnya dalam konteks data kesehatan. Dalam perspektif agama, Al-Quran mengejarkan keseimbangan sebagai prinsip penting dalam kehidupan. Allah SWT telah menjadikan umat Nabi Muhammad SAW sebagai umat pertengahan, sebagaimana firman-Nya dalam QS Al-Baqarah Ayat 143:

وَكَذٰلِكَ جَعَلْنَاكُمْ أُمَّةً وَسَطًا لِّتَكُونُوا شُهَدَاءَ عَلَى النَّاسِ وَيَكُونَ الرَّسُولُ عَلَيْكُمْ شَهِيدًا

"*Dan demikianlah (sebagaimana Kami telah memimpin kamu ke jalan yang lurus), Kami jadikan kamu (wahai umat Muhammad) satu umat yang pilihan lagi adil, supaya kamu layak menjadi orang yang memberi keterangan kepada umat manusia (tentang yang benar dan yang salah) dan Rasulullah (Muhammad) pula akan menjadi orang yang menerangkan kebenaran perbuatan kamu." (QS. Al-Baqarah/ 25:2)*

Imam al-Baghawi (510 H) menjelaskan di dalam tafsirnya, berdasarkan penjelasan al-Kalbi berkaitan maksud umat pertengahan adalah pengikut agama

yang adil, tidak berlebih-lebihan dalam beribadah dan tidak longgar dalam menjalankan syariat agama. Kedua sifat tersebut dicela dalam ajaran agama(Mushfique, 2019).

Dalam konteks ini, sejalan dengan konsep umat pertengahan yang menekankan pentingnya keadilan dan kesetaraan, termasuk dalam proses analisis data yang memerlukan penerapan metode *balancing* data yang tepat. Hal ini bertujuan untuk menciptakan keseimbangan antara kelas mayoritas dan minoritas, sehingga menghasilkan analisis yang lebih akurat dan meningkatkan performa klasifikasi penyakit.

Salah satu masalah utama dalam pengelolaan data kesehatan adalah ketidakseimbangan kelas pada *dataset*. Ketidakseimbangan ini dapat mengurangi performa model, terutama pada kasus penyakit kritis seperti stroke. Klasifikasi penyakit menggunakan pembelajaran mesin (*machine learning*) yang dapat membantu menemukan pola tersembunyi dalam data dan mendukung pengambilan keputusan yang lebih baik (Mahesh, 2020). Dalam melakukan klasifikasi terdapat beberapa metode seperti *Naïve Bayes, Random Forest, Neural Network, Classification* and *Regression Trees (CART), Support Vector Machine (SVM), Decision Tree* dan lain-lain. Penelitian yang dilakukan oleh Adnin Kamila dkk., (2023) mengenai algoritma *Random Forest* dengan judul Klasifikasi Penyakit Jantung Menggunakan *Decision Tree* dan *Random Forest* menghasilkan kesimpulan bahwa metode *Random Forest* lebih baik daripada metode *Decision Tree*. Metode *Random Forest* menghasilkan nilai akurasi 81.82%, nilai *precision* sebesar 87.04%, nilai *recall* sebesar 79.82%, dan *F1-score* sebesar 83.13%.

Random Forest merupakan salah satu algoritma yang sangat baik dalam melakukan identifikasi kesalahan dengan kecepatan dan ketepatan dalam kumpulan data yang besar. Namun, algoritma *Random Forest* ini memiliki kelemahan yang akan mengakibatkan terjadinya *overfitting* apabila menggunakan data yang tidak seimbang(Br. Sebayang dkk., 2023) .

Dalam konteks data kesehatan, data yang tidak seimbang dapat berdampak signifikan, terutama jika kelas minoritas merepresentasikan penyakit serius seperti stroke. Berdasarkan data dari *World Health Organization* (WHO) tahun 2022, risiko terkena stroke meningkat sebesar 50% dalam 17 tahun terakhir, dengan prevalensi global mencapai 1 dari 4 orang selama masa hidupnya. Di Indonesia, stroke menjadi penyebab utama kematian, dengan prevalensi meningkat dari 7 per 1.000 penduduk pada tahun 2013 menjadi 10,9 per 1.000 penduduk pada tahun 2018 (Direktorat Promosi Kesehatan dan Pemberdayaan Masyarakat, 2024). Selain itu, kecacatan yang disebabkan penyakit stroke sangat memengaruhi produktivitas penderitanya(Harmayetty dkk., 2020).

Data minoritas sering kali diberi label secara tidak akurat ketika diproses oleh sebuah model klasifikasi, sehingga untuk menangani masalah ketidakseimbangan kelas pada *dataset* diperlukan teknik *resampling*(Imama Sabilla & Bella Vista, 2021). Teknik ini menciptakan data baru pada kelas minoritas untuk menyeimbangkan distribusi kelas. Penelitian data *imbalance* yang dilakukan oleh Akbar & Hayaty, (2020) dengan judul Data *Balancing* untuk Mengatasi *imbalance Dataset* pada Prediksi Produksi Padi menghasilkan kesimpulan bahwa penggunaan algoritma *SMOTE* dalam *resampling* data untuk mengatasi *imbalance* data dapat

meningkatkan akurasi seperti pada pengujian menggunakan algoritma *CART* yang didapatkan hasil akurasi sebesar 47.67% sebelum *balancing* data dan mendapatkan hasil akurasi sebesar 55.73% setelah dilakukan *balancing* data.

Berdasarkan penjelasan tersebut, penanganan ketidakseimbangan data menjadi aspek yang sangat penting dalam meningkatkan performa klasifikasi, terutama pada kasus penyakit stroke yang memiliki dampak serius terhadap kesehatan dan produktivitas. Penerapan teknik *resampling*, seperti *oversampling* dapat menyeimbangkan jumlah kelas pada dataset. Dengan demikian performa model klasifikasi meningkat, hasil analisis menjadi lebih representative, dan model dapat diandalkan untuk mendukung pengambilan keputusan yang lebih baik.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dijelaskan sebelumnya, terdapat sebuah masalah yang diangkat dalam penelitian ini yakni bagaimana pengaruh penanganan ketidakseimbangan kelas menggunakan metode *balancing* data terhadap performa klasifikasi pada *dataset* penyakit stroke menggunakan algoritma *Random Forest*?

1.3 Batasan Masalah

Untuk memastikan pembahasan sesuai dengan rumusan dan tujuan masalah, diperlukan batasan-batasan masalah yang jelas. Batasan ini membantu mengarahkan fokus, menghindari penyimpangan, sehingga analisis dan penyelesaian masalah menjadi lebih terstruktur dan efisien. Berikut batasan masalah yang digunakan dalam penelitian ini:

1. *Dataset* yang digunakan dalam penelitian ini yaitu data kesehatan terkait pasien stroke dengan total 5.110 data. Fitur targetnya adalah stroke, yang terdiri dari 2 kelas yakni label 1 (terindikasi stroke) dan 0 (sehat atau tidak terindikasi stroke). Dataset pada penelitian ini tidak seimbang, dengan jumlah pasien terindikasi stroke lebih sedikit dibandingkan yang tidak terindikasi stroke.
2. Fitur yang digunakan pada dataset ini terdiri dari *gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg_glucose_level, bmi, smoking_status,* dan stroke.
3. Penelitian ini mengevaluasi performa model klasifikasi stroke dengan menggunakan akurasi dan *F1-Score*.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini sesuai dengan rumusan masalah yang telah dijelaskan, yakni untuk menganalisa pengaruh penanganan ketidakseimbangan kelas menggunakan metode *balancing* data terhadap performa klasifikasi pada *dataset* penyakit stroke menggunakan algoritma *Random Forest*?

1.5 Manfaat Penelitian

Manfaat penelitian ini adalah untuk menambah wawasan dalam ilmu data dan *machine learning*, serta menyediakan referensi bagi penelitian selanjutnya terkait metode *balancing* data dan algoritma klasifikasi.

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Penelitian yang dilakukan oleh Gusrialni Fitri dkk., mengimplementasikan metode klasifikasi *Naïve Bayes* dan *C4.5* untuk diagnosa penyakit stroke. *Dataset* pada penelitian ini berasal dari *Kaggle* yang berjumlah 5110 data dengan 12 atribut yang mana setelah dilakukan *preprocessing* terdapat 1767 data yang bisa tidak digunakan dan hanya 3343 data yang digunakan. Penelitian ini membandingkan 2 metode klasifikasi yaitu *Naïve Bayes* dan *C4.5* dengan menggunakan *Tools Rapid Miner*. Hasil dari penelitian ini menunjukkan bahwa metode *C4.5* menghasilkan akurasi yang lebih tinggi yaitu 92.22%, sedangkan metode *Naïve Bayes* menghasilkan akurasi sebesar 89.22% (Gusrialni Fitri dkk., 2023).

Penelitian yang dilakukan oleh Arie Nugroho (2022) berjudul Analisa *Splitting Criteria* Pada *Decision Tree* dan *Random Forest* untuk Klasifikasi Evaluasi Kendaraan yang mana penelitian ini bertujuan untuk membandingkan *splitting criteria* dalam metode klasifikasi menggunakan *Decision Tree* dan *Random Forest*. *Dataset* pada penelitian ini berasal dari dataset publik yaitu *UC Irvine Machine Learning repository* yang merupakan data evaluasi mobil dengan 1728 total data yang memiliki 7 atribut yang terdiri dari 6 fitur dan 1 label yang tidak memiliki *missing values.* Hasil akurasi pada penelitian ini dengan pengujian *cross validation* adalah *Random Forest* memperoleh akurasi tertinggi dengan *criterion gain ratio* sebesar 94.56%, sedangkan *Decision Tree* memperoleh akurasi

tertinggi dengan *criterion information gain* sebesar 93.06%. Pada pemilihan *criterion* didapatkan hasil akurasi tertinggi dari *Decision Tree* sebesar 93.60% dengan 90% data training dan 10% data testing, sedangkan hasil akurasi tertinggi dari *Random Forest* sebesar 96.51% dengan 90% data *training* dan 10% data *testing* (Nugroho, 2022) .

Penelitian yang dilakukan oleh Cherfly Kaope dan Yoga Pristyanto (2023) yang berjudul *The Effect of Class Imbalance Handling on Datasets Toward Classification Algorithm Performance* yang bertujuan untuk mengetahui pengaruh penanganan ketidakseimbangan kelas pada *dataset* terhadap kinerja klasifikasi. Penelitian ini menggunakan metode *ADASYN, SMOTE,* dan *SMOTE-ENN* untuk mengatasi ketidakseimbangan kelas dan dipadukan dengan algoritma klasifikasi *AdaBoost, K-Nearest Neighbor,* dan *Random Forest*. Pada penelitian ini menggunakan 5 *Dataset* yang berasal dari KEEL-Dataset yaitu Pima, Wisconsin, glass1, glass0, dan segment0. Hasil akurasi terbaik didapatkan pada pengujian kombinasi *ADASYN + Random Forest* dengan akurasi pada 5 *dataset* adalah 78.6%, 94.7%, 83.7%, 93%, and 99.8% (Kaope & Pristyanto, 2023).

Penelitian yang dilakukan oleh Imanuel Khrisna Ananda et., yang berjudul Penerapan *Random Oversampling* dan Algoritma *Boosting* untuk Memprediksi Kualitas Buah Jeruk. Penelitian ini menerapkan Teknik *random oversampling* dan empat algoritma *boosting* yaitu *Adaboost*, *Gradient Boosting*, *LightGBM* dan *Catboost* untuk memprediksi kualitas buah jeruk. Dataset yang digunakan pada penelitian ini bersumber dari dataset public Orange Quality Analysis Dataset. Hasil menunjukkan bahwa Random Oversampling dapat meningkatkan akurasi pada

semua algoritma yang digunakan, dan didapatkan hasil terbaik yaitu algoritma *boosting catboost* dengan nilai akurasi sebesar 91.42%(Ananda dkk., 2024).

Berdasarkan penelitian yang telah dipaparkan, penggunaan metode *Random Forest* dalam klasifikasi penyakit terbukti menghasilkan akurasi yang baik. Selain itu, penerapan metode *balancing* data mampu meningkatkan akurasi dalam proses klasifikasi. Oleh karena itu, penelitian ini ditulis untuk mengetahui performa metode *balancing* data dalam meningkatkan akurasi pada klasifikasi penyakit stroke. *Dataset* yang digunakan berasal dari *open dataset* Kaggle.

Tabel 2. 1 Penelitian Terkait

No	Referensi	Metode Penelitian	Hasil Penelitian	Perbedaan
1	Gusrialni Fitri dkk., (2023)	C4.5, *Naïve Bayes*	Metode C4.5 menghasilkan akurasi yang lebih tinggi yaitu 92.22%, sedangkan metode *Naïve Bayes* menghasilkan akurasi sebesar 89.22%	Metode yang digunakan berbeda, pada jurnal menggunakan C4.5 dan *Naïve Bayes* sedanagkan peneliti menggunakan *Random Forest.* Namun, *dataset* yang digunakan sama.
2	Arie Nugroho (2022)	*Decision Tree* dan *Random Forest*	Hasil akurasi tertinggi didapatkan oleh *Random Forest* yaitu sebesar 96.51%, sedangkan *Decision Tree* sebesar 95.30%	*Dataset* yang digunakan berbeda yakni pada jurnal menggunakan data evaluasi mobil sedangkan peneliti menggunakan data stroke. Salah satu metode yang digunakan sama yaitu peneliti dan jurnal sama sama menggunakan *Random Forest.*
3	Cherfly Kaope dan Yoga Pristyanto (2023)	Metode *balancing* yang digunakan yaitu *ADASYN, SMOTE*, dan *SMOTE-enn*. Algoritma Klasifikasi yang digunakan yaitu	Hasil akurasi terbaik didapatkan pada pengujian kombinasi *ADASYN+Random Forest* dengan akurasi pada 5 *dataset* adalah 78.6%, 94.7%, 83.7%, 93%, dan 99.8%.	*Dataset* yang digunakan berbeda, pada jurnal menggunakan 5 dataset yaitu Pima, Wisconsin, glass1, glass0, dan segment0, sedangkan peneliti menggunakan data

No	Referensi	Metode Penelitian	Hasil Penelitian	Perbedaan
		AdaBoost, *K-Nearest Neighbour*, dan *Random Forest*		stroke. Metode *balancing* data yang digunakan peneliti yaitu SMOTE, ADASYN, dan *Random Over Sampling*. Algoritma klasifikasi yang digunakan oleh peneliti yaitu *Random Forest*
4	Ananda dkk., (2024)	Metode Balancing yang digunakan yaitu random oversampling. Algoritma prediksi yang digunakan yaitu empat algoritma boosting, *Adaboost*, *Gradient Boosting*, *LightGBM*, dan *Catboost*	Hasil menunjukkan bahwa Random Oversampling dapat meningkatkan akurasi pada semua algoritma yang digunakan, dan didapatkan hasil terbaik yaitu algoritma *boosting catboost* dengan nilai akurasi sebesar 91.42%	*Dataset* yang digunakan berbeda, pada jurnal menggunakan dataset yaitu *public Orange Quality Analysis Dataset*, sedangkan peneliti menggunakan data stroke. Metode *balancing* data yang digunakan pada jurnal yaitu hanya *random over sampling*, sedangkan peneliti menggunakan tiga metode *balancing* data yaitu SMOTE, ADASYN, dan *random over sampling*. Algoritma klasifikasi yang digunakan oleh peneliti yaitu *Random Forest*

2.2 Stroke

Stroke adalah penyakit yang menyerang pembuluh darah di otak dan tidak menular yang dapat menyebabkan kecacatan hingga kematian bagi penderitanya(Adiyasa, 2024). Stroke disebabkan oleh berbagai faktor risiko seperti hipertensi, diabetes melitus, dan penyakit jantung(Amalia dkk., 2024). Secara lebih spesifik faktor risiko stroke yang umum dikenal yakni dibagi menjadi dua yaitu faktor yang dapat diubah meliputi hipertensi, merokok, diabetes mellitus, kelainan

jantung, alkohol, latihan fisik, lipoprotein, homoisistein, kontrasepsi oral, gangguan pola tidur, kegemukan, dan drug abuse. Faktor yang kedua adalah faktor risiko yang tidak dapat diubah meliputi jenis kelamin, umur faktor keturunan, dan ras/etnik (Handayani dkk., 2023).

Selain memahami faktor risiko yang sudah dijelaskan di atas, mengenali gejala-gejala stroke penting sekali untuk mendapat penanganan yang tepat. Secara umum, gejala-gejala stroke seperti kelumpuhan wajah atau anggota badan (biasanya satu sisi saja) yang timbul mendadak, gangguan kepekaan pada satu atau lebih anggota badan, perubahan mendadak status mental (bingung, mengigau, koma), afasia (bicara tidak lancar, ucapan kurang, atau sulit memahami ucapan), disartria (bicara pelo atau cadel), gangguan penglihatan atau diplopia (penglihatan dobel), ataksia (kesulitan gerakan), vertigo, mual, dan muntah, atau nyeri kepala (Yudha Chrisanto dkk., 2022).

2.3 Balancing Data

Kinerja suatu model *machine learning* sangat bergantung pada kualitas data yang digunakan. Jika kualitas data yang digunakan tidak bagus atau terjadi ketidakseimbangan di antara kelas-kelas yang berbeda maka kinerja suatu mode *machine learning* akan menurun secara signifikan (Islam dkk., 2022). Ketidakseimbangan *dataset* adalah kondisi dimana terjadinya jumlah kelas yang berbeda didalam *dataset*. Sebuah kelas dikatakan tidak seimbang apabila terdapat suatu kelas yang memiliki jumlah data lebih banyak daripada kelas yang lain. Kelas yang memiliki jumlah data lebih banyak disebut kelas mayoritas, sedangkan kelas yang memiliki jumlah data lebih sedikit disebut kelas minoritas (Sulistiyono dkk.,

2021). Terdapat dua metode yang sering digunakan dalam *balancing* data yakni metode *oversampling* dan *undersampling* (Chamidah dkk., 2021). Pada penelitian ini metode yang digunakan dalam *balancing* data adalah metode *oversampling*. Dalam penelitian ini teknik *balancing* data yang digunakan yakni SMOTE, ADASYN, dan *Random Over Sampling.*

2.3.1 *Synthetic Minority Oversampling Technique* (SMOTE)

SMOTE pertama kali diperkenalkan oleh Nithes V. Chawla sebagai salah satu metode untuk menangani ketidakseimbangan data(Sonjaya dkk., 2022). *SMOTE* merupakan salah satu teknik dalam *oversampling* untuk menyeimbangkan jumlah data kelas yakni dengan menambahkan jumlah data minoritas sejumlah data kelas mayoritas. Data baru dibuat berdasarkan *k-Nearest Neighbours*. Data dengan kelas numerik cara menambahkannya dengan menghitung jarak *Euclidian* sedangkan untuk data kelas kategorik menambahkannya dengan menghitung nilai modus (Ayuningtyas & Uswatun, 2021). Berikut merupakan prosedur perhitungan metode SMOTE, dimana untuk setiap pola data(X_0) dari kelas minoritas(Elreedy & Atiya, 2019):

1) Pilih salah satu data dari K tetangga terdekat X yang juga termasuk kelas minoritas.
2) Buat data baru (Z) pada titik dari segmen garis yang menghubungkan pola tetangga yang dipilih, sebagai berikut:

$$Z = X_0 + w(X - X_0) \tag{2.1}$$

Keterangan :
Z : Data baru
X_0 : Data kelas minoritas yang dijadikan acuan
X : Data kelas minoritas yang dijadikan pembanding
w : variabel acak dalam rentang [0,1]

2.3.2 *Adaptive Synthethic* (ADASYN)

ADASYN merupakan salah satu teknik *resampling* data yang diusulkan pertama kali oleh Haibo He dkk (Rahayu dkk., 2017). ADASYN merupakan salah satu teknik pembelajaran untuk menangani data yang tidak seimbang(Yulian Pamuji & Dwi Arma Putri, 2023). Konsep dari ADASYN adalah menggunakan distribusi densitas sebagai kriteria untuk membuat data baru dari kelas minoritas (Amien dkk., 2022). Sampel baru yang dibentuk oleh ADASYN berdasarkan distribusi data dimaksudkan agar mengurangi bias yang disebabkan oleh distribusi data yang tidak merata dari kelas mayoritas(Hidayat dkk., 2021). Berikut merupakan prosedur perhitungan metode ADASYN (Marlisa dkk., 2024):

1) Hitung rasio data minoritas terhadap data mayoritas (*d*)

$$d = \frac{m_s}{m_l} \tag{2.2}$$

d : rasio data minoritas terhadap data mayoritas
m_s : data minoritas
m_l : data mayoritas

2) Hitung total jumlah pengamatan data baru yang akan dihasilkan (*G*)

$$G = (m_l - m_s)\beta \tag{2.3}$$

G : jumlah data baru yang akan dihasilkan
m_s : jumlah data minoritas
m_l : jumlah data mayoritas
β : nilai *d* yang diinginkan

3) Temukan k-NN untuk setiap titik minoritas dan hitung nilai yang menunjukkan berapa banyak tetangga yang berasal dari kelas mayoritas (r_i)

$$r_i = \frac{\Delta i}{k} \tag{2.4}$$

r_i : perbandingan jumlah tetangga yang berasal dari kelas mayoritas
Δi : jumlah dari k tetangga terdekat yang berasal dari kelas mayoritas
K : jumlah K-NN yang diinginkan

4) Normalisasi r_i untuk membuatnya sama dengan 1

$$\hat{r}_i = \frac{r_i}{\sum_{i=1}^{m_s} r_i} \tag{2.5}$$

$\hat{r}_i$: nilai r_i yang dinormalisasi
r_i : jumlah tetangga yang berasal dari kelas mayoritas

5) Hitung jumlah pengamatan data baru yang dihasilkan untuk setiap kelompok tetangga terdekat dari data minoritas (g_i).

$$g_i = (\hat{r}_i)(G) \tag{2.6}$$

g_i : jumlah data baru yang dihasilkan untuk setiap kelompok tetangga terdekat
$\hat{r}_i$: nilai r_i yang dinormalisasi
G : total data baru yang dihasilkan pada persamaan 2.3

6) Hasilkan hasilkan data baru s_i dengan persamaan berikut

$$s_i = x_i + (x_{zi} - x_i)(\lambda) \tag{2.7}$$

s_i : data baru
x_i : data minoritas dari kelompok tetangga terdekat
x_{zi} : contoh data minoritas yang dipilih acak dari kelompok yang sama
λ : bilangan bulat yang dihasilkan secara acak antara 0-1

2.3.3 *Random Over Sampling* (ROS)

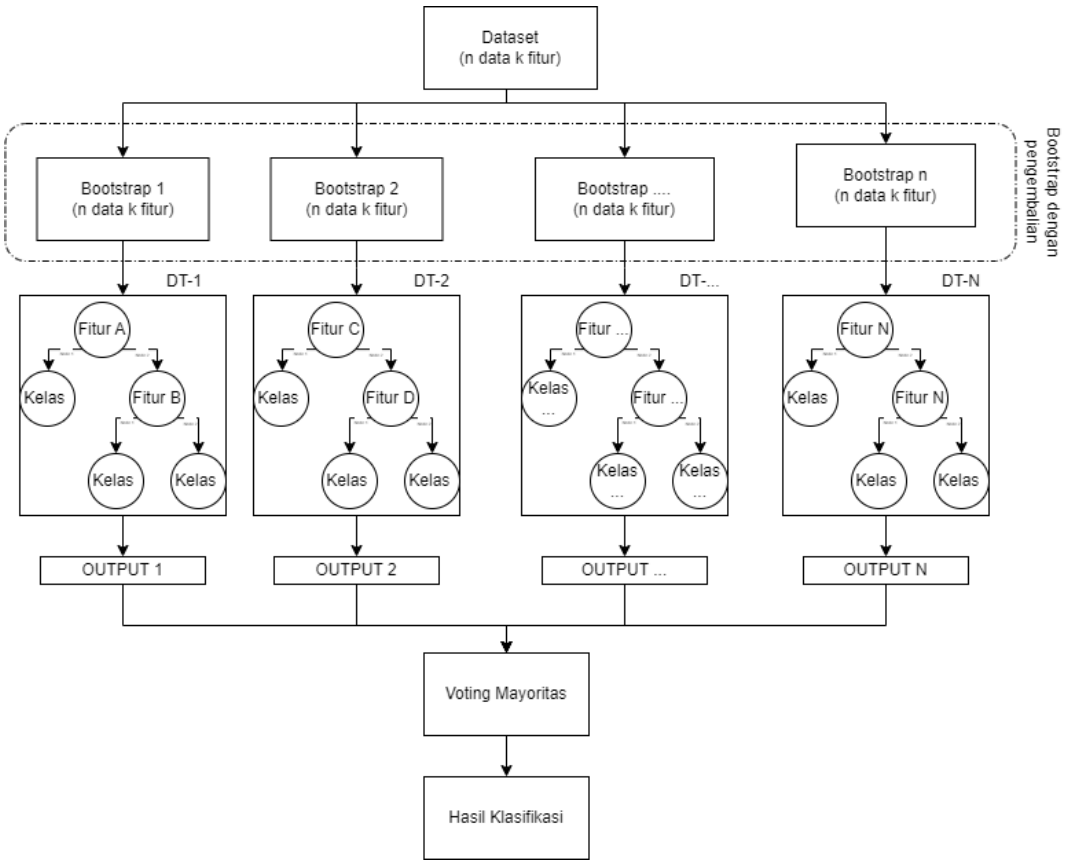
Random over sampling adalah salah satu teknik *balancing* data yang merupakan bagian *preprocessing* data untuk menyeimbangkan kelas dari suatu data yang tidak seimbang (Aryanti dkk., 2023). Selain itu, metode *random over sampling* merupakan metode *balancing* data yang paling sederhana dan mudah digunakan yakni hanya dengan menyalin data dari kelas minoritas dan tidak membuat data baru, sehingga dapat dengan mudah menyebabkan model mengalami *overfitting* (Dai dkk., 2022).

2.4 Random Forest

Random forest adalah salah satu algoritma klasifikasi yang diperkenalkan oleh Leo Breiman, dimana *random forest* ini terdiri dari kumpulan pohon-pohon keputusan yang berguna untuk klasifikasi data ke suatu kelas tertentu. Pohon keputusan dibuat untuk menentukan node dan diakhiri dengan beberapa *node* daun untuk mencapai hasil akhir(Suci Amaliah dkk., 2022).

Random forest merupakan pengembangan lebih lanjut dari metode CART. Salah satu perbedaan utamanya adalah jumlah pohon yang dibangun, dan algoritma *random forest* menggunakan banyak pohon. Hasil tiap pohon dalam klasifikasi atau regresi digabungkan melalui proses pemungutan suara untuk menentukan kelas atau label yang paling sering muncul di antara hasil individual pohon tersebut. Nama *random* dalam metode ini mengacu pada proses acak pembuatan pohon, di mana data diacak sebelum membuat setiap pohon(Dinova & Prasetiyo, 2024).

Salah satu keunggulan utama algoritma *random forest* adalah teknik *bootstrap aggregation* atau yang sering disebut *bagging*. Pada proses ini, *dataset* akan diambil secara acak dengan pengembalian. Setiap *subset* data, digunakan untuk membangun pohon keputusan. Pohon-pohon tersebut akan belajar dari *subset* data yang berbeda-beda, sehingga mampu mengurangi risiko *overfitting*. Setelah semua pohon keputusan selesai dibangun, hasil klasifikasi dari setiap pohon akan digabungkan dengan metode *voting mayoritas* untuk menentukan prediksi akhir seperti ditunjukkan pada Gambar 2. 1



Gambar 2. 1 Desain Sistem *Random Forest*

Berikut adalah langkah-langkah algoritma *random forest* (Jackins dkk., 2021) :

1. Mengambil sampel dari *dataset* beberapa kali dengan pengembalian atau bagging (*bootstrap aggregating*).
2. Membuat pohon keputusan dengan memilih fitur-fitur secara acak dari setiap *subset* data berdasarkan kriteria gini

$$Gini(s_i) = 1 - \sum_{i=0}^{c-1} p_i^2 \tag{2.8}$$

Keterangan:
Pi : Probabilitas *Node*
c : Jumlah kelas

3. Dari fitur-fitur yang dipilih secara acak tersebut carilah titik *split* terbaik berdasarkan kriteria gini

$$Gini_{split} = \sum_{i=0}^{k-1} (\frac{n_i}{n}) Gini(s_i) \quad (2.9)$$

Keterangan:
k : Jumlah kelas
n_i : Total data pada suatu kelas
n : Total seluruh data

4. Proses memilih fitur dan mencari titik split terbaik berlanjut untuk setiap *node* dalam pohon sampai pohon selesai dibangun.

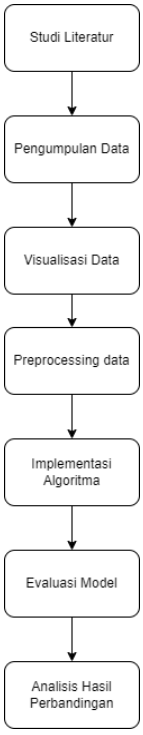
5 Melakukan klasifikasi melalui suara terbanyak (*voting*) dari gabungan pohon yang ada.

BAB III

DESAIN DAN IMPLEMENTASI

3.1 Alur Penelitian

Pada penelitian ini, alur penelitian ditunjukkan oleh Gambar 3. 1. Alur penelitian tersebut dimulai dengan studi literatur untuk memahami teori dan penelitian sebelumnya. Selanjutnya, pengumpulan data yakni berasal dari *open dataset kaggle*, lalu divisualisasikan untuk mengenali distribusi yang ada. Data tersebut kemudian diproses melalui tahap *preprocessing* agar siap digunakan. Setelah itu, algoritma diterapkan pada data yang telah diproses, dan model yang dihasilkan dievaluasi menggunakan *confusion matrix*. Hasil evaluasi dianalisis dan dibandingkan untuk menentukan metode *balancing* terbaik.



Gambar 3. 1 Alur Penelitian

3.2 Pengumpulan Data

Data yang digunakan pada penelitian ini berasal dari *open dataset* Kaggle, dengan nama *dataset* yaitu *healthcare-dataset-stroke-data* dengan link sebagai berikut (https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset). Total data adalah 5110 dengan 11 fitur yakni *gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg_glucose_level, bmi, smoking_status,* dan stroke. *Dataset* pada penelitian ini ditunjukkan oleh Tabel 3. 1.

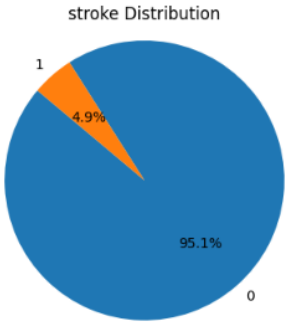
Tabel 3. 1 *Dataset*

gender	age	Hyper tension	Heart disease	Ever married	Work type	Residence type	Avg glucose level	bmi	Smoking status	stroke
Male	67	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
Female	61	0	0	Yes	Self-employed	Rural	202.21	N/A	never smoked	1
Male	80	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Female	35	0	0	Yes	Self-employed	Rural	82.99	30.6	never smoked	0
Male	51	0	0	Yes	Private	Rural	166.29	25.6	formerly smoked	0
Female	44	0	0	Yes	Govt_job	Urban	85.28	26.2	Unknown	0

3.3 Visualisasi Data

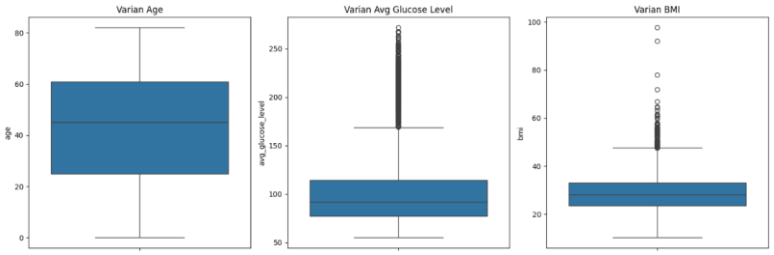
Visualisasi data adalah perpaduan antara sains dan seni untuk membantu orang memahami maksud suatu data dengan menampilkannya dalam bentuk visual (Ariandi & Rahma Puteri, 2022). Visualisasi data pada penelitian ini digunakan untuk melihat variasi jumlah data di fitur stroke, melihat fitur yang memiliki *outlier*, dan melihat data fitur yang memiliki nilai kosong.

1. Gambar 3.2 merupakan visualisasi presentase jumlah kelas pada *dataset* yang mana kelas 1(terindikasi stroke) memiliki presentase 4.9% dan yang kelas 0(tidak terindikasi stroke) memiliki presentase 95.1%.



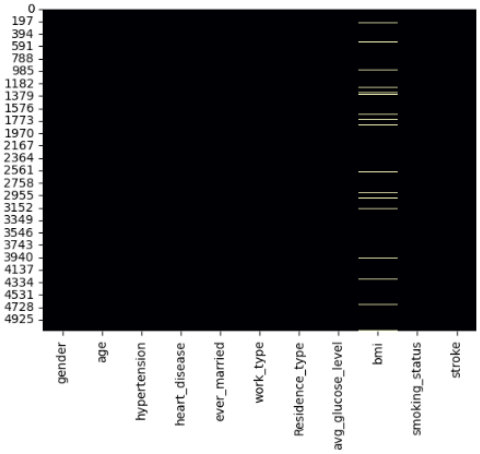
Gambar 3. 2 Visualisasi Presentase Kelas

2. Gambar 3.3 merupakan visualisasi untuk melihat fitur yang memiliki *outlier* yang mana pada fitur age tidak memiliki *outlier* namun pada fitur *avg_Glucose Level* dan BMI memiliki *outlier.*



Gambar 3. 3 Visualisasi *Outlier*

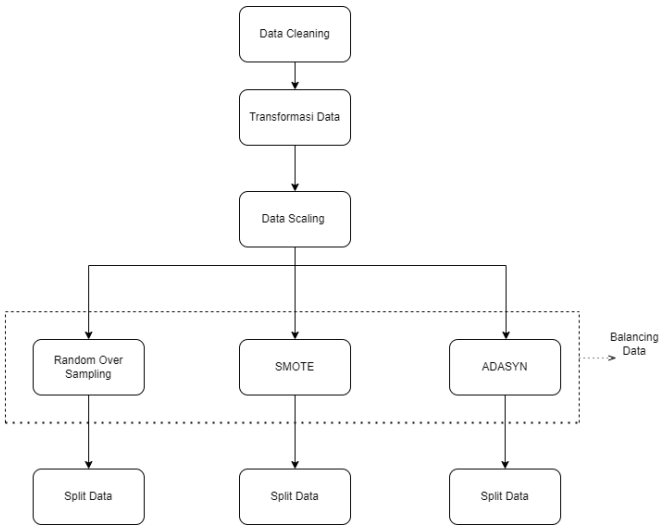
3. Gambar 3.4 merupakan visualisasi fitur pada *dataset* yang memiliki nilai kosong yang mana terdapat *missing values* pada fitur BMI.



Gambar 3. 4 Visualisasi *Missing Values*

3.4 Preprocessing Data

Preprocessing data adalah proses untuk mengubah data mentah menjadi data yang lebih terstruktur untuk siap digunakan dan dianalisis. Alur *preprocessing* pada penelitian ini ditunjukkan oleh Gambar 3.5, sedangkan Tabel 3.2 merupakan contoh data untuk melakukan perhitungan manual.



Gambar 3. 5 Alur *Preprocessing*

Tabel 3. 2 Data Perhitungan Manual

gender	age	Hyper tension	Heart disease	Ever married	Work type	Residence type	Avg glucose level	bmi	Smoking status	stroke
Male	67	0	1	Yes	Private	Urban	228.69	36.6	Formerly smoked	1
Female	61	0	0	Yes	Govt_Job	Rural	202.21	NaN	Never smoked	1
Male	80	0	1	Yes	Private	Rural	105.92	32.5	Never smoked	1
Female	49	0	0	Yes	Private	Urban	171.23	34.4	Smokes	1
Female	79	1	0	Yes	Self-Employed	Rural	174.12	24	Never smoked	1
Male	81	0	0	Yes	Private	Urban	186.21	29	Formerly smoked	1
Male	74	1	1	Yes	Private	Rural	70.09	27.4	Never smoked	1
Female	69	0	0	No	Private	Urban	94.39	22.8	Never smoked	1
Female	59	0	0	Yes	Private	Rural	76.15	NaN	Unkonown	1
Female	78	0	0	Yes	Private	Urban	58.57	24.2	Unknown	1
Female	56	0	0	Yes	Private	Urban	77.49	36	Formerly smoked	0
Female	60	0	0	Yes	Private	Urban	65.38	41.2	Formerly smoked	0
Female	57	0	0	Yes	Private	Urban	94.18	27.1	Never smoked	0

gender	age	Hyper tension	Heart disease	Ever married	Work type	Residence type	Avg glucose level	bmi	Smoking status	stroke
Male	29	0	0	Yes	Self-Employed	Urban	118.7	33.2	Unknown	0
Female	76	0	0	Yes	Govt_Job	Urban	96.29	25.4	Smokes	0

3.4.1 Data *Cleaning*

Data *cleaning* adalah langkah awal pada tahap *preprocessing* untuk memperbaiki, menghapus, mengurangi ketidakkonsistensi, dan memastikan keakuratan informasi suatu data agar kualitas data yang digunakan siap dianalisis. Pada penelitian ini dilakukan pengecekkan data duplikat, pengecekkan *missing values*, dan pengecekkan *outliers*.

1. Pengecekkan data duplikat: Tidak ditemukan data duplikat pada Tabel 3.2
2. Pengecekkan *missing values*: Ditemukan *missing values* pada *dataset* Tabel 3.2 yakni pada fitur BMI. Untuk memperbaiki data yang memiliki *missing values* peneliti mengisinya dengan menggunakan nilai rata-rata dari keseluruhan data BMI.

Tabel 3. 3 Data Setelah *Cleaning*

gender	age	Hyper tension	Heart disease	Ever married	Work type	Residence type	Avg glucose level	bmi	Smoking status	stroke
Male	67	0	1	Yes	Private	Urban	228.69	36.6	Formerly smoked	1
Female	61	0	0	Yes	Govt_Job	Rural	202.21	28.89	Never smoked	1
Male	80	0	1	Yes	Private	Rural	105.92	32.5	Never smoked	1
Female	49	0	0	Yes	Private	Urban	171.23	34.4	Smokes	1
Female	79	1	0	Yes	Self-Employed	Rural	174.12	24	Never smoked	1
Male	81	0	0	Yes	Private	Urban	186.21	29	Formerly smoked	1
Male	74	1	1	Yes	Private	Rural	70.09	27.4	Never smoked	1
Female	69	0	0	No	Private	Urban	94.39	22.8	Never smoked	1
Female	59	0	0	Yes	Private	Rural	76.15	28.89	Unkonown	1
Female	78	0	0	Yes	Private	Urban	58.57	24.2	Unknown	1
Female	56	0	0	Yes	Private	Urban	77.49	36	Formerly smoked	0

gender	age	Hyper tension	Heart disease	Ever married	Work type	Residence type	Avg glucose level	bmi	Smoking status	stroke
Female	60	0	0	Yes	Private	Urban	65.38	41.2	Formerly smoked	0
Female	57	0	0	Yes	Private	Urban	94.18	27.1	Never smoked	0
Male	29	0	0	Yes	Self-Employed	Urban	118.7	33.2	Unknown	0
Female	76	0	0	Yes	Govt_Job	Urban	96.29	25.4	Smokes	0

3. Pengecekkan *outliers*: Terdapat data *outlier* pada fitur *age* yang diketahui melalui IQR (*interquartile range*), yang selanjutnya data yang terindikasi *outlier* akan dihapus.

   Berikut langkah-langkahnya:

   1. Urutkan data *age*: 29, 49, 56, 57, 59, 60, 61, 67, 69, 74, 76, 78, 79, 80, 81
   2. Tentukan Q1 (kuartil pertama) dan Q3 (kuartil ketiga).
   3. Hitung IQR = Q3 - Q1.

      Posisi $Q1=0.25\times(n+1)$

      Posisi $Q3=0.75\times(n+1)$
   4. Tentukan batas bawah dan batas atas:

      Batas bawah = Q1 - 1.5 * IQR

      Batas atas = Q3 + 1.5 * IQR

   Sehingga, dalam data *age* tersebut, nilai **29** adalah satu-satunya *outlier*.

Tabel 3. 4 Data Setelah Penghapusan *Outlier*

gender	age	Hyper tension	Heart disease	Ever married	Work type	Residence type	Avg glucose level	bmi	Smoking status	stroke
Male	67	0	1	Yes	Private	Urban	228.69	36.6	Formerly smoked	1
Female	61	0	0	Yes	Govt_Job	Rural	202.21	28.89	Never smoked	1
Male	80	0	1	Yes	Private	Rural	105.92	32.5	Never smoked	1
Female	49	0	0	Yes	Private	Urban	171.23	34.4	Smokes	1
Female	79	1	0	Yes	Self-Employed	Rural	174.12	24	Never smoked	1

gender	age	Hyper tension	Heart disease	Ever married	Work type	Residence type	Avg glucose level	bmi	Smoking status	stroke
Male	81	0	0	Yes	Private	Urban	186.21	29	Formerly smoked	1
Male	74	1	1	Yes	Private	Rural	70.09	27.4	Never smoked	1
Female	69	0	0	No	Private	Urban	94.39	22.8	Never smoked	1
Female	59	0	0	Yes	Private	Rural	76.15	28.89	Unkonown	1
Female	78	0	0	Yes	Private	Urban	58.57	24.2	Unknown	1
Female	56	0	0	Yes	Private	Urban	77.49	36	Formerly smoked	0
Female	60	0	0	Yes	Private	Urban	65.38	41.2	Formerly smoked	0
Female	57	0	0	Yes	Private	Urban	94.18	27.1	Never smoked	0
Female	76	0	0	Yes	Govt_Job	Urban	96.29	25.4	Smokes	0

3.4.2 Transformasi Data

Transformasi data merupakan proses untuk mengubah data kategori menjadi data numerik(Aditya dkk., 2021). Data yang berbentuk kategori adalah data *gender*, *ever_married*, *work_type*, *residence type*, dan *smoking status*, dan stroke. Proses transformasi dilakukan dengan teknik *Label Encoder*, untuk variabel yang memiliki dua kategori dan teknik *One-Hot Encoding* untuk variabel dengan kategori yang lebih dari dua(Tahyudin dkk., 2024) . Penggunaan *One-Hot Encoding* pada variabel dengan lebih dari dua kategori bertujuan untuk menghindari asumsi bahwa ada urutan atau peringkat antara kategori yang sebenarnya tidak memiliki tingkatan. Sebagaimana dijelaskan dalam buku "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" oleh Geron, (2019) yang mana mengubah secara langsung data kategori ke dalam angka (seperti A=1, B=2, C=3) dapat membuat algoritma *machine learning* salah menganggap adanya hubungan linear di antara kategori tersebut. Dengan *One-Hot Encoding* setiap kategori dianggap sama penting oleh algoritma, sehingga tidak ada kategori yang dianggap lebih

tinggi atau lebih rendah dari kategori lainnya. Hal ini mencegah kesalahan algoritma dalam menganggap ada hubungan atau urutan tertentu di antara kategori tersebut. Hal ini sangat penting untuk memastikan data yang digunakan model sesuai dengan karakteristik algoritma, terutama pada model berbasis numerik seperti *random forest.* Perubahannya adalah sebagai berikut:

1. *Gender* yang memiliki nilai kategorik “Male”, “Female” diubah menjadi numerik 1, 0 menggunakan *Label Encoder.*
2. *Ever married* yang memiliki nilai kategorik “Yes” dan “No” diubah menjadi numerik 1, 0 menggunakan *Label Encoder.*
3. *Work type* yang memiliki nilai kategorik “Private”, “Govt_job”, “Self-employed”, “children”, “Never_worked” diubah menggunakan *One-Hot Encoding*
4. *Residence type* yang memiliki nilai kategorik “Rural”, “Urban” diubah menjadi numerik 0, 1 menggunakan *Label Encoder.*
5. *Smoking* status yang memiliki nilai kategorik “Never Smoked”, “Unknown”, “Smokes”, “Formerly Smoked” diubah menggunakan *One-Hot Encoding*.
6. Stroke yang memiliki nilai kategorik “Stroke”, “Tidak Stroke” diubah menjadi numerik 0, 1 menggunakan *Label Encoder.*

Hasil transformasi data pada perhitungan manual ditunjukkan oleh Tabel 3. 5

Tabel 3. 5 Data Setelah Transformasi

Gender	Age	Hyper Tension	Heart Disease	Ever Married	Private	Govt Job	Self Employed	Residence Type	Avg Glucose Level	Bmi	Unknown	Formerly Smoked	Never Smoked	Smokes	Stroke
1	67	0	1	1	1	0	0	1	228.69	36.6	0	1	0	0	1
0	61	0	0	1	0	1	0	0	202.21	28.89	0	0	1	0	1
1	80	0	1	1	1	0	0	0	105.92	32.5	0	0	1	0	1
0	49	0	0	1	1	0	0	1	171.23	34.4	0	0	0	1	1
0	79	1	0	1	0	0	1	0	174.12	24	0	0	1	0	1
1	81	0	0	1	1	0	0	1	186.21	29	0	1	0	1	1
1	74	1	1	1	1	0	0	0	70.09	27.4	0	0	1	0	1
0	69	0	0	0	1	0	0	1	94.39	22.8	0	0	1	0	1
0	59	0	0	1	1	0	0	0	76.15	28.89	1	0	0	0	1
0	78	0	0	1	1	0	0	1	58.57	24.2	1	0	0	0	1
0	56	0	0	1	1	0	0	1	77.49	36	0	1	0	0	0
0	60	0	0	1	1	0	0	1	65.38	41.2	0	1	0	0	0
0	57	0	0	1	1	0	0	1	94.18	27.1	0	0	1	0	0
0	76	0	0	1	0	1	0	1	96.29	25.4	0	0	0	1	0

3.4.3 Data Scaling

Data *scaling* adalah proses mengubah nilai fitur-fitur dalam *dataset* agar berada dalam skala yang sama, biasanya dalam rentang 0 hingga 1. Tujuan dari *scaling* adalah untuk memastikan bahwa semua fitur berkontribusi secara proporsional dalam model pembelajaran mesin. Hal ini penting karena fitur dengan skala yang sangat berbeda dapat mendominasi algoritma dan menyebabkan bias dalam model, yang dapat mengurangi akurasi prediksi dan performa keseluruhan model(Gde Agung Brahmana Suryanegara dkk., 2021).

Pada penelitian ini, data *scaling* dilakukan menggunakan metode *Min-Max Normalization* pada kolom-kolom dalam *dataset*. Setiap kolom dalam *dataset*, seperti *gender*, *age*, *hypertension*, dan lain-lain, dinormalisasi dengan Persamaan 3.1.

$$N_{new} = \frac{N_{old} - N_{min}}{N_{max} - N_{min}} \tag{3.1}$$

Transformasi data ini mengubah setiap nilai dalam kolom *dataset* menjadi rentang nilai 0 hingga 1. Dengan melakukan *scaling*, semua fitur dalam *dataset* menjadi sebanding, sehingga model pembelajaran mesin dapat memproses data dengan lebih efisien dan menghasilkan prediksi yang lebih akurat. Normalisasi ini juga membantu dalam mempercepat proses pelatihan dan meningkatkan stabilitas serta performa model. Hasil setelah *scaling* ditunjukkan oleh Tabel 3. 6.

Tabel 3. 6 Data Setelah *Scaling*

Gender	Age	Hyper Tension	Heart Disease	Ever Married	Private	Govt Job	Self Employed	Residence Type	Avg Glucose Level	Bmi	Unknown	Formerly Smoked	Never Smoked	Smokes	Stroke
1	0.5625	0	1	1	1	0	0	1	1	0.75	0	1	0	0	1
0	0.375	0	0	1	0	1	0	0	0.844345	0.331226	0	0	1	0	1
1	0.96875	0	1	1	1	0	0	0	0.278333	0.527174	0	0	1	0	1
0	0	0	0	1	1	0	0	1	0.662238	0.630435	0	0	0	1	1
0	0.9375	1	0	1	0	0	1	0	0.679226	0.065217	0	0	1	0	1
1	1	0	0	1	1	0	0	1	0.750294	0.336957	0	1	0	1	1
1	0.78125	1	1	1	1	0	0	0	0.067717	0.25	0	0	1	0	1
0	0.625	0	0	0	1	0	0	1	0.210557	0	0	0	1	0	1
0	0.3125	0	0	1	1	0	0	0	0.103339	0.331226	1	0	0	0	1
0	0.90625	0	0	1	1	0	0	1	0	0.076087	1	0	0	0	1
0	0.21875	0	0	1	1	0	0	1	0.111216	0.717391	0	1	0	0	0
0	0.34375	0	0	1	1	0	0	1	0.040031	1	0	1	0	0	0
0	0.25	0	0	1	1	0	0	1	0.209323	0.233696	0	0	1	0	0
0	0.84375	0	0	1	0	1	0	1	0.221726	0.141304	0	0	0	1	0

3.4.4 Balancing Data

Dataset yang digunakan pada penelitian ini memiliki label kelas yang tidak seimbang yakni label 1(terindikasi stroke) berjumlah 4.9% dari *dataset*, sedangkan sedangkan label 0 (tidak terindikasi stroke) berjumlah 95.1% dari *dataset*. Oleh karena itu, untuk mengatasi ketidaseimbangan data tersebut dilakukan *balancing data* menggunakan metode *oversampling.* Pada Tabel 3. 6 merupakan sebagian data yang diambil dari *dataset* stroke yang telah melalui beberapa tahap *preprocessing*, selanjutnya dilakukan perhitungan manual dengan data yang memiliki label 1 berjumlah 10 data, dan data yang memiliki label 0 berjumlah 4 data.

A. SMOTE

Metode SMOTE (*Synthetic Minority Oversampling Technique*) digunakan untuk menangani ketidakseimbangan kelas pada data dengan membuat data baru dari kelas minoritas. Berikut adalah langkah-langkah perhitungan manual metode SMOTE:

1. Data minoritas adalah kelas 0. Misalnya,

   X_0 = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0]

2. Hitung jarak *euclidian* antara X_0 dengan semua contoh lain dalam kelas minoritas. Persamaan jarak *euclidian* antara dua vektor A dan B adalah:

$$d(X_0, X_1) = \sqrt{\sum_{i=1}^{n} (X_0 - X_1)^2} \qquad (3.2)$$

   X_0 : data minoritas pertama
   X_1 : data minoritas kedua
   d : jarak *euclidian*

   Hitung jarak contoh kedua :

X_0 = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0]

X_1 = [0, 0.34375, 0, 0, 1, 1, 0, 0, 1, 0.040031, 1, 0, 1, 0, 0, 0]

$$d(X_0, X_1) =$$

$$\sqrt{(0-0)^2 + (0.21875 - 0.34375)^2 + (0-0)^2 + (0-0)^2 +}$$

$$\sqrt{(1-1)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 +}$$

$$\sqrt{(0.111216 - 0.040031)^2 + (0.717391 - 1)^2 + (0-0)^2 +}$$

$$\sqrt{(1-1)^2 + (0-0)^2 + (0-0)^2 + (0-0)^2}$$

$$= 0.317112206$$

Hitung jarak contoh ketiga:

X_0 = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0]

X_2 = [0, 0.25, 0, 0, 1, 1, 0, 0, 1, 0.209323, 0.233696, 0, 0, 1, 0, 0]

$$d(X_0, X_2) =$$

$$\sqrt{(0-0)^2 + (0.21875 - 0.25)^2 + (0-0)^2 + (0-0)^2 +}$$

$$\sqrt{(1-1)^2 + (1-1)^2 + (0-0)^2 + (0-0)^2 + (1-1)^2 +}$$

$$\sqrt{(\,0.111216 - 0.209323)^2 \; + (\,0.717391 - \; 0.233696)^2}$$

$$\sqrt{(0-0)^2 + (0-0)^2 + (1-0)^2 +}$$

$$\sqrt{(0-1)^2 + (0-0)^2 + (0-0)^2 +}$$

$$= \; 1.49818637$$

3. Pilih tertangga terdekat dari X_0 berdasarkan perhitungan sebelumnya

$$d\,(X_0, X_1) \; = 0.317112206$$

$$d\,(X_0, X_2) \; = \; 1.49818637$$

dari perhitungan tersebut tetangga terdekat adalah X_1

4. Menentukan jumlah KNN atau tetangga terdekat yang akan digunakan, pada perhitungan manual ini, yakni menggunakan KNN=2
5. Buat sampel baru (*Z*) dari X_0 dan X_1 dengan Persamaan 2.1

   Z = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0] + 0.4 * ([0, 0.34375, 0, 0, 1, 1, 0, 0, 1, 0.040031, 1, 0, 1, 0, 0, 0] - [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0])

   Z = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0] + 0.4 * [0, 0.125, 0, 0, 0, 0, 0, 0, 0, -0.071185, 0.282609, 0, 0, 0, 0, 0]

   Z = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0] + [0, 0.05, 0, 0, 0, 0, 0, 0, 0, -0.028474, 0.1130436, 0, 0, 0, 0, 0]

   Z = [0, 0.26875, 0, 0, 1, 1, 0, 0, 1, 0.082742, 0.8304346, 0, 1, 0, 0, 0]
6. Lakukan Langkah-langkah tersebut secara berulang dan konsisten dengan memilih data minoritas secara acak, menghitung jarak terhadap tetangga terdekat, serta menghasilkan sampel baru hingga jumlah data pada tiap label kelas mencapai kondisi seimbang atau memiliki jumlah yang sama. Langkah ini memastikan bahwa ketidakseimbangan kelas pada *dataset* berhasil diatasi dan distribusi data menjadi lebih proporsional antara kelas minoritas dan mayoritas..
7. Pada Tabel 3. 7 merupakan hasil data setelah dilakukan *balancing* menggunakan SMOTE dengan total kelas yang sudah seimbang, yakni 10 data berlabel 0 dan 10 data berlabel 1.

Tabel 3. 7 Data *Balancing* SMOTE

Gender	Age	Hyper Tension	Heart Disease	Ever Married	Private	Govt Job	Self Employed	Residence Type	Avg Glucose Level	Bmi	Unknown	Formerly Smoked	Never Smoked	Smokes	Stroke
1	0.5625	0	1	1	1	0	0	1	1	0.75	0	1	0	0	1
0	0.375	0	0	1	0	1	0	0	0.844345	0.331226	0	0	1	0	1
1	0.96875	0	1	1	1	0	0	0	0.278333	0.527174	0	0	1	0	1
0	0	0	0	1	1	0	0	1	0.662238	0.630435	0	0	0	1	1
0	0.9375	1	0	1	0	0	1	0	0.679226	0.065217	0	0	1	0	1
1	1	0	0	1	1	0	0	1	0.750294	0.336957	0	1	0	1	1
1	0.78125	1	1	1	1	0	0	0	0.067717	0.25	0	0	1	0	1
0	0.625	0	0	0	1	0	0	1	0.210557	0	0	0	1	0	1
0	0.3125	0	0	1	1	0	0	0	0.103339	0.331226	1	0	0	0	1
0	0.90625	0	0	1	1	0	0	1	0	0.076087	1	0	0	0	1
0	0.21875	0	0	1	1	0	0	1	0.111216	0.717391	0	1	0	0	0
0	0.34375	0	0	1	1	0	0	1	0.040031	1	0	1	0	0	0
0	0.25	0	0	1	1	0	0	1	0.209323	0.233696	0	0	1	0	0
0	0.84375	0	0	1	0	1	0	1	0.221726	0.141304	0	0	0	1	0
0	0.26875	0	0	1	1	0	0	1	0.082742	0.830435	0	1	0	0	0
0	0.29375	0	0	1	1	0	0	1	0.068505	0.8869564	0	1	0	0	0
0	0.2375	0	0	1	1	0	0	1	0.1700802	0.427174	0	0	1	0	0
0	0.60625	0	0	1	0	1	0	1	0.2167648	0.1782608	0	0	0	1	0
0	0.26875	0	0	1	1	0	0	1	0.082742	0.8304346	0	1	0	0	0
0	0.30625	0	0	1	1	0	0	1	0.1077478	0.6934784	0	1	0	0	0

B. ADASYN

Berikut contoh perhitungan manual metode ADASYN:

1. Menghitung rasio data minoritas terhadap data mayoritas pada data Tabel 3. 6 dengan Persamaan 2.2

$$d = \frac{4}{10} = 0.4$$

2. Menghitung data baru yang ingin dihasilkan dengan Persamaan 2.3

$$G = (10 - 4)1 = 6$$

3. Tentukan KNN untuk setiap titik minoritas dan hitung nilai yang menunjukkan berapa banyak tetangga yang berasal dari kelas mayoritas (r_i).

   Titik 1 = 1 dari 3 tetangga terdekat adalah data mayoritas

   Titik 2 = 1 dari 3 tetangga terdekat adalah data mayoritas

   Titik 3 = 1 dari 3 tetangga terdekat adalah data mayoritas

   Titik 4 = 2 dari 3 tetangga terdekat adalah data mayoritas

   Untuk K=3, maka dengan Persamaan 2.4

$$r_1 = \frac{1}{3} = 0.33$$
$$r_2 = \frac{1}{3} = 0.33$$
$$r_3 = \frac{1}{3} = 0.33$$
$$r_4 = \frac{2}{3} = 0.66$$

4. Normalisasi r_i untuk membuatnya sama dengan 1 menggunakan Persamaan 2.5

$$\hat{r}_1 = \frac{0.33}{1.6} = 0.2$$

$$\hat{r}_2 = \frac{0.33}{1.6} = 0.2$$

$$\hat{r}_3 = \frac{0.33}{1.6} = 0.2$$

$$\hat{r}_4 = \frac{0.66}{1.6} = 0.4$$

5. Hitung jumlah pengematan data baru berdasarkan tetangga terdekat dari kelas mayoritas dengan Persamaan 2.6

$$g_1 = (0.2)(6) = 1.2 = 1\ data$$

$$g_2 = (0.2)(6) = 1.2 = 1\ data$$

$$g_3 = (0.2)(6) = 1.2 = 1\ data$$

$$g_4 = (0.4)(6) = 2.4 = 2\ data$$

Total data baru yang dihasilkan adalah 1+1+1+2=5

6. Hasilkan data baru dengan Persamaan 2.7

S_i = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0] + 0.4 * ([0, 0.34375, 0, 0, 1, 1, 0, 0, 1, 0.040031, 1, 0, 1, 0, 0, 0] - [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0])

S_i = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0] + 0.4 * [0, 0.125, 0, 0, 0, 0, 0, 0, 0, -0.071185, 0.282609, 0, 0, 0, 0, 0]

S_i = [0, 0.21875, 0, 0, 1, 1, 0, 0, 1, 0.111216, 0.717391, 0, 1, 0, 0, 0] + [0, 0.05, 0, 0, 0, 0, 0, 0, 0, -0.028474, 0.1130436, 0, 0, 0, 0, 0]

S_i = [0, 0.26875, 0, 0, 1, 1, 0, 0, 1, 0.082742, 0.8304346, 0, 1, 0, 0, 0]

7. Ulangi langkah sebelumnya berdasarkan jumlah data yang dihasilkan pada masing-masing titik data sampai data baru berjumlah *G* dan *dataset* menjadi seimbang. Tabel 3. 8 merupakan hasil data setelah dilakukan *balancing* menggunakan ADASYN dengan total kelas yang sudah seimbang.

Tabel 3. 8 Data *Balancing* ADASYN

Gender	Age	Hyper Tension	Heart Disease	Ever Married	Private	Govt Job	Self Employed	Residence Type	Avg Glucose Level	Bmi	Unknown	Formerly Smoked	Never Smoked	Smokes	Stroke
1	0.5625	0	1	1	1	0	0	1	1	0.75	0	1	0	0	1
0	0.375	0	0	1	0	1	0	0	0.844345	0.331226	0	0	1	0	1
1	0.96875	0	1	1	1	0	0	0	0.278333	0.527174	0	0	1	0	1
0	0	0	0	1	1	0	0	1	0.662238	0.630435	0	0	0	1	1
0	0.9375	1	0	1	0	0	1	0	0.679226	0.065217	0	0	1	0	1
1	1	0	0	1	1	0	0	1	0.750294	0.336957	0	1	0	1	1
1	0.78125	1	1	1	1	0	0	0	0.067717	0.25	0	0	1	0	1
0	0.625	0	0	0	1	0	0	1	0.210557	0	0	0	1	0	1
0	0.3125	0	0	1	1	0	0	0	0.103339	0.331226	1	0	0	0	1
0	0.90625	0	0	1	1	0	0	1	0	0.076087	1	0	0	0	1
0	0.21875	0	0	1	1	0	0	1	0.111216	0.717391	0	1	0	0	0
0	0.34375	0	0	1	1	0	0	1	0.040031	1	0	1	0	0	0
0	0.25	0	0	1	1	0	0	1	0.209323	0.233696	0	0	1	0	0
0	0.84375	0	0	1	0	1	0	1	0.221726	0.141304	0	0	0	1	0
0	0.59375	0	0	1	0	1	0	1	0.177522	0.3717388	0	0	0	1	0
0	0.50625	0	0	1	0	1	0	1	0.3979308	0.3369564	0	0	0	1	0
0	0.65625	0	0	1	0	1	0	1	0.4707736	0.2172728	0	0	0	1	0
0	0.26875	0	0	1	1	0	0	1	0.082742	0.8304346	0	1	0	0	0
0	0.29375	0	0	1	1	0	0	1	0.068505	0.8869564	0	1	0	0	0
0	0.2375	0	0	1	1	0	0	1	0.1700802	0.427174	0	0	0	0	0

C. Random Over Sampling

Random over sampling adalah metode sederhana yang tidak memerlukan perhitungan manual yang kompleks. Metode ini bekerja dengan cara menduplikasi data dari kelas minoritas sehingga jumlahnya menjadi seimbang dengan kelas mayoritas. Berikut adalah langkah-langkah perhitungan manual dalam *balancing* menggunakan *random over sampling*:

1. Menghitung selisih data: langkah pertama adalah menghitung selisih jumlah data antara kelas mayoritas dan kelas minoritas. Sebagai contoh, berdasarkan Tabel 3. *6*, terdapat 10 data dengan label stroke (kelas mayoritas) dan 4 data dengan label tidak stroke (kelas minoritas).
2. Menduplikasi data kelas minoritas: data dari kelas minoritas diduplikasi secara acak. Dalam contoh ini, data kelas minoritas (tidak stroke) diduplikasi sebanyak 6 kali, sehingga jumlahnya menjadi 10 data sehingga setara dengan jumlah kelas mayoritas.
3. Hasil setelah *balancing*: setelah proses *random over sampling* selesai, jumlah data dari kedua kelas menjadi seimbang. Tabel 3. 9 menunjukkan hasil data yang telah melalui proses *balancing* dengan jumlah data yang sama pada kedua kelas.

Tabel 3. 9 Data Balancing ROS

Gender	Age	Hyper Tension	Heart Disease	Ever Married	Private	Govt Job	Self Employed	Residence Type	Avg Glucose Level	Bmi	Unknown	Formerly Smoked	Never Smoked	Smokes	Stroke
1	0.5625	0	1	1	1	0	0	1	1	0.75	0	1	0	0	1
0	0.375	0	0	1	0	1	0	0	0.844345	0.331226	0	0	1	0	1
1	0.96875	0	1	1	1	0	0	0	0.278333	0.527174	0	0	1	0	1
0	0	0	0	1	1	0	0	1	0.662238	0.630435	0	0	0	1	1
0	0.9375	1	0	1	0	0	1	0	0.679226	0.065217	0	0	1	0	1
1	1	0	0	1	1	0	0	1	0.750294	0.336957	0	1	0	1	1
1	0.78125	1	1	1	1	0	0	0	0.067717	0.25	0	0	1	0	1
0	0.625	0	0	0	1	0	0	1	0.210557	0	0	0	1	0	1
0	0.3125	0	0	1	1	0	0	0	0.103339	0.331226	1	0	0	0	1
0	0.90625	0	0	1	1	0	0	1	0	0.076087	1	0	0	0	1
0	0.21875	0	0	1	1	0	0	1	0.111216	0.717391	0	1	0	0	0
0	0.34375	0	0	1	1	0	0	1	0.040031	1	0	1	0	0	0
0	0.25	0	0	1	1	0	0	1	0.209323	0.233696	0	0	1	0	0
0	0.84375	0	0	1	0	1	0	1	0.221726	0.141304	0	0	0	1	0
0	0.25	0	0	1	1	0	0	1	0.209323	0.233696	0	0	1	0	0
0	0.34375	0	0	1	1	0	0	1	0.040031	1	0	1	0	0	0
0	0.21875	0	0	1	1	0	0	1	0.111216	0.717391	0	1	0	0	0
0	0.34375	0	0	1	1	0	0	1	0.040031	1	0	1	0	0	0
0	0.25	0	0	1	1	0	0	1	0.209323	0.233696	0	0	1	0	0
0	0.84375	0	0	1	0	1	0	1	0.221726	0.141304	0	0	0	1	0

Berdasarkan hasil perhitungan manual, perbandingan hasil balancing dari metode SMOTE, ADASYN, dan ROS adalah sebagai berikut:

1. *Random over sampling*: Data baru yang dihasilkan memiliki pola pengulangan dari data asli, karena metode random over sampling dilakukan dengan melakukan duplikasi data dari kelas minoritas sehingga tidak ada variasi tambahan yang dihasilkan pada data.
2. SMOTE: Data baru yang dihasilkan memiliki variasi tambahan dari data asli, karena metode SMOTE dilakukan dengan melakukan perhitungan antara data minoritas dan tetangga terdekatnya dalam kelas minoritas. Dengan demikian, data sintetis yang dihasilkan bukanlah duplikasi langsung, melainkan sampel baru yang berada di antara dua titik data, sehingga menambahkan variasi
3. ADASYN: Data baru yang dihasilkan memiliki variasi tambahan dan lebih adaptif, karena metode ADASYN dilakukan dengan fokus pada sampel minoritas yang berada di area yang sulit diklasifikasikan. ADASYN menghasilkan lebih banyak data sintetis di sekitar sampel minoritas yang memiliki lebih banyak tetangga dari kelas mayoritas, sehingga menambahkan variasi yang sesuai dengan kebutuhan klasifikasi pada area yang sulit. Pembuatan data baru tetap dilakukan antara sampel minoritas dan tetangganya dari kelas minoritas, namun distribusi data baru lebih berfokus pada area yang memerlukan penyeimbangan lebih besar.

3.5 Split Data

Split data merupakan proses membagi *dataset* menjadi 2 bagian yakni data *training* dan data *testing*. Terdapat banyak sekali rasio untuk membagi *dataset*

seperti 6:4, 7:3, 8:2, dan 9:1. Pada penelitian yang pernah dilakukan oleh Dolly Indra dkk, mengenai pengklasifikasian menggunakan *random forest* dengan citra huruf BISINDO memberikan akurasi tertinggi yakni 94.2% dengan *scenario split* data 80:20(Indra dkk., 2024) . Oleh karena itu, pada penelitian ini menggunakan rasio *split* data 8:2 yang mana 80% *dataset* digunakan sebagai data *training*, dan 20% dataset digunakan sebagai data *testing*.

3.6 Algoritma *Random Forest*

Dataset yang digunakan unutuk perhitungan manual ditunjukkan oleh Tabel 3. 8 dengan fitur targetnya yaitu stroke yang memiliki 2 kelas, kelas 1 (terindikasi stroke) dan kelas 0 (sehat atau tidak terindikasi stroke). Pada perhitungan manual algoritma klasifikasi, peneliti menggunakan data *sample* setelah dilakukan *balancing* menggunakan ADASYN.

Random forest merupakan pengembangan algoritma CART yang memiliki perbedaan tahapan perhitungan pada langkah pertamanya, yakni pada tahap pertama algoritma *random forest* terdapat langkah untuk melakukan *bootstrap sampling* dari *dataset* yang tersedia. Langkah selanjutnya yaitu perhitungan *index* gini, fitur yang memiliki data bersifat numerik seperti *age, avg_glucose_level*, dan *bmi* diperlukan nilai *threshold* untuk menghitung probabilitas *index* gini yang akan dijadikan sebagai *splitting node*. Sedangkan, fitur kategorikal yang hanya memiliki dua nilai tidak diperlukan *threshold* untuk menghitung *index gini*. *Threshold* didapatkan dengan cara menghitung *mean.* Setelah itu dilakukan *sorting* dan hasil klasifikasi dihitung berdasarkan voting terbanyak dari semua pohon yang terbentuk. Berikut merupakan perhitungan manual algoritma *random forest*:

Dataset pada Tabel 3. 8 akan dilakukan *bootstrap sampling*, yakni melakukan pengambilan acak dengan pengembalian dan pemilihan fitur yang digunakan. Pemilihan jumlah fitur yang diambil secara acak dapat ditentukan dengan menghitung $\sqrt{m}$ dimana m adalah jumlah fitur selain fitur target(Mahmuda, 2024). Pada penelitian ini dilakukan bootstrap sebanyak 5 kali untuk contoh perhitungan manual dengan pemilihan jumlah fitur yakni $\sqrt{m}$ pada setiap bootstrap yang dilakukan. Dengan 5 kali *bootstrap* dan total fitur pada *dataset* adalah 15 fitur selain fitur target, maka pada penelitian ini seluruh fitur akan terwakili. Namun, jika jumlah *bootstrap* ditingkatkan, ada kemungkinan beberapa fitur terpilih lebih dari satu kali secara acak.

1. Perhitungan *Index Gini Bootstrap* Pertama

Tabel 3. 10 Data Hasil Bootstrap Pertama

Gender	Age	Hypertension	Stroke
1	0.5625	0	1
0	0.375	0	1
1	0.96875	0	1
0	0	0	1
0	0.9375	1	1
1	1	0	1
1	0.78125	1	1
0	0.625	0	1
0	0.3125	0	1
0	0.90625	0	1
0	0.21875	0	0
0	0.34375	0	0
0	0.25	0	0
0	0.84375	0	0
0	0.59375	0	0
0	0.50625	0	0
0	0.65625	0	0
0	0.26875	0	0
0	0.29375	0	0
0	0.2375	0	0

Setelah melakukan *bootstrap*, selanjutnya adalah menghitung nilai *index gini* dari semua fitur dengan Persamaan 2.8. Dari perhitungan Persaamaan 2.8

didapatkan hasil *index gini* untuk masing-masing fitur. Hasil tersebut diperoleh dengan mempertimbangkan jumlah data pada setiap kelas berdasarkan *node* yang telah ditentukan melalui *threshold* masing-masing fitur.

Tabel 3. 11 Rincian Data *Gender Bootstrap* Pertama

Gender Group	Stoke(1)	Tidak Stroke(0)	Total
0	6	10	16
1	4	0	4
total data			20

Tabel 3. 12 Rincian Data *Age Bootstrap* Pertama

Age Group	Stoke(1)	Tidak Stroke(0)	Total
≤ 0.53	3	7	10
> 0.53	7	3	10
total data			20

Tabel 3. 13 Rincian Data *Hypertension Bootstrap* Pertama

Hypertension Group	Stoke(1)	Tidak Stroke(0)	Total
0	8	10	18
1	2	0	2
total data			20

Setelah rincian data pada masing-masing fitur diketahui, langkah berikutnya adalah menghitung nilai probabilitas untuk setiap kelas pada masing-masing *node* berdasarkan *threshold* atau kategori fitur yang telah ditentukan. Probabilitas ini digunakan untuk menghitung *index gini* pada setiap fitur. Berikut adalah tabel probabilitas untuk fitur-fitur pada *bootstrap* pertama.

Tabel 3. 14 Probabilitas Fitur *Gender Bootstrap* Pertama

Node 1 = 0	
Probabilitas Stroke	0.375
Probabilitas Tidak Stroke	0.625
Node 2 = 1	
Probabilitas Stroke	1
Probabilitas Tidak Stroke	0

Tabel 3. 15 Probabilitas Fitur *Age Bootstrap* Pertama

Node 1 ≤ 0.53	
Probabilitas Stroke	0.3
Probabilitas Tidak Stroke	0.7
Node 2 > 0.53	
Probabilitas Stroke	0.7
Probabilitas Tidak Stroke	0.3

Tabel 3. 16 Probabilitas Fitur *Hypertension Bootstrap* Pertama

Node 1 = 0	
Probabilitas Stroke	0.375
Probabilitas Tidak Stroke	0.625
Node 2 = 1	
Probabilitas Stroke	1
Probabilitas Tidak Stroke	0

Langkah selanjutnya adalah perhitungan nilai *gini impurity* dari masing-masing fitur pada *bootstrap* pertama dengan Persamaan 2.9, sehingga menghasilkan nilai seperti tabel-tabel berikut.

Tabel 3. 17 *Index Gini Impurity* Fitur *Gender Bootstrap* Pertama

Node 1 = 0	0.46875
Node 2 = 1	0
Gini Total	0.375

Tabel 3. 18 *Index Gini Impurity* Fitur *Age Bootstrap* Pertama

Node 1 ≤ 0.53	0.42
Node 2 > 0.53	0.42
Gini Total	0.42

Tabel 3. 19 *Index Gini Impurity* Fitur *Hypertension Bootstrap* Pertama

Node 1 = 0	0.493827
Node 2 = 1	0
Gini Total	0.44444

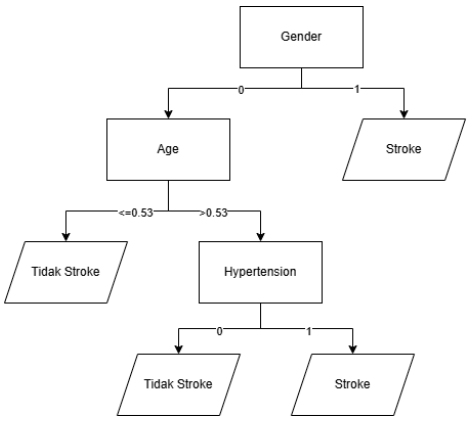
Setelah *indexs gini* dihitung untuk setiap fitur, langkah selanjutnya adalah melakukan pengurutan *index gini* dari nilai terkecil ke terbesar. Fitur dengan nilai

index gini terkecil akan dipilih sebagai *root node* pada pohon keputusan. Tabel 3. 20 adalah hasil pengurutan *index gini* pada *bootstrap* pertama

Tabel 3. 20 Hasil *Sorting Index* Gini *Bootstrap* Pertama

Fitur	Gini Total
Gender	0.30
Age	0.42
Hypertension	0.44

Setelah pengurutan *index gini* dari nilai terkecil ke terbesar. Fitur dengan nilai indeks Gini terkecil akan dipilih sebagai *root node* pada pohon keputusan. Gambar 3. 6 adalah pohon keputusan yang terbentuk dari hasil pengurutan *index gini*.



Gambar 3. 6 Pohon Keputusan Bootstrap Pertama

2. Perhitungan *Index Gini Bootstrap* Kedua

Tabel 3. 21 Data Hasil *Bootstrap* Kedua

Heart Disease	Ever Married	Private	Stroke
1	1	1	1
0	1	0	1
1	1	1	1
0	1	1	1
0	1	0	1
0	1	1	1
1	1	1	1
0	0	1	1
0	1	1	1
0	1	1	1
0	1	1	0

Heart Disease	Ever Married	Private	Stroke
0	1	1	0
0	1	1	0
0	1	0	0
0	1	0	0
0	1	0	0
0	1	0	0
0	1	1	0
0	1	1	0
0	1	1	0

Setelah melakukan *bootstrap*, selanjutnya adalah menghitung nilai *index gini* dari semua fitur dengan Persamaan 2.8. Dari perhitungan Persaamaan 2.8 didapatkan hasil *index gini* untuk masing-masing fitur. Hasil tersebut diperoleh dengan mempertimbangkan jumlah data pada setiap kelas berdasarkan *node* yang telah ditentukan melalui *threshold* masing-masing fitur.

Tabel 3. 22 Rincian Data *Heart Disease Bootstrap* Kedua

Heart Disease Group	Stoke(1)	Tidak Stroke(0)	Total
0	7	10	17
1	3	0	3
total data			20

Tabel 3. 23 Rincian Data *Ever Married Bootstrap* Kedua

Ever Married Group	Stoke(1)	Tidak Stroke(0)	Total
0	1	0	1
1	9	10	19
total data			20

Tabel 3. 24 Rincian Data *Private(Work Type) Bootstrap* Kedua

Private(Work Type) Group	Stoke(1)	Tidak Stroke(0)	Total
0	2	4	6
1	8	6	14
total data			20

Setelah rincian data pada masing-masing fitur diketahui, langkah berikutnya adalah menghitung nilai probabilitas untuk setiap kelas pada masing-masing *node*

berdasarkan *threshold* atau kategori fitur yang telah ditentukan. Probabilitas ini digunakan untuk menghitung *index gini* pada setiap fitur. Berikut adalah tabel probabilitas untuk fitur-fitur pada *bootstrap* kedua.

Tabel 3. 25 Probabilitas Fitur *Heart Disease Bootstrap* Kedua

Node 1 = 0	
Probabilitas Stroke	0.411765
Probabilitas Tidak Stroke	0.588235
Node 2 = 1	
Probabilitas Stroke	1
Probabilitas Tidak Stroke	0

Tabel 3. 26 Probabilitas Fitur *Ever Married Bootstrap* Kedua

Node 1 = 0	
Probabilitas Stroke	1
Probabilitas Tidak Stroke	0
Node 2 = 1	
Probabilitas Stroke	0.473684
Probabilitas Tidak Stroke	0.526316

Tabel 3. 27 Probabilitas Fitur *Private(Work Type) Bootstrap* Kedua

Node 1 = 0	
Probabilitas Stroke	0.333333
Probabilitas Tidak Stroke	0.666667
Node 2 = 1	
Probabilitas Stroke	0.571429
Probabilitas Tidak Stroke	0.428571

Langkah selanjutnya adalah perhitungan nilai *gini impurity* dari masing-masing fitur pada *bootstrap* kedua dengan Persamaan 2.9, sehingga menghasilkan nilai seperti tabel-tabel berikut.

Tabel 3. 28 *Index Gini Impurity* Fitur *Heart Disease Bootstrap* Kedua

Node 1 = 0	0.484429
Node 2 = 1	0
Gini Total	0.411765

Tabel 3. 29 *Index Gini Impurity* Fitur *Ever Married Bootstrap* Kedua

Node 1 = 0	0
Node 2 = 1	0.498615
Gini Total	0.473684

Tabel 3. 30 *Index Gini Impurity* Fitur *Private(Work Type) Bootstrap* Kedua

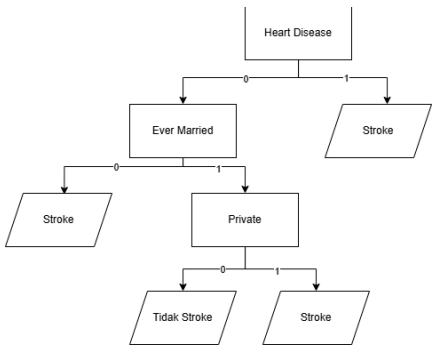
Node 1 = 0	0.444444
Node 2 = 1	0.489796
Gini Total	0.47619

Setelah *index gini* dihitung untuk setiap fitur, langkah selanjutnya adalah melakukan pengurutan *index gini* dari nilai terkecil ke terbesar. Fitur dengan nilai indeks gini terkecil akan dipilih sebagai *root node* pada pohon keputusan Tabel 3. 31 adalah hasil pengurutan *index gini* pada *bootstrap* kedua.

Tabel 3. 31 Hasil *Sorting Index Gini Bootstrap* Kedua

Fitur	**Gini Total**
Heart Disease	0.411765
Ever Married	0.473684
Private(Work Type)	0.47619

Setelah pengurutan index gini, fitur dengan nilai indeks gini terkecil akan dipilih sebagai root node pada pohon keputusan. Gambar 3. 7 adalah pohon Keputusan yang terbentuk dari hasil pengurutan index gini



Gambar 3. 7 Pohon Keputusan *Bootstrap* Kedua

3. Perhitungan *Index Gini Bootstrap* Ketiga

Tabel 3. 32 Data Hasil *Bootstrap* Ketiga

Govt Job	Self Employed	Residence Type	Stroke
0	0	1	1
1	0	0	1
0	0	0	1
0	0	1	1
0	1	0	1
0	0	1	1
0	0	0	1
0	0	1	1
0	0	0	1
0	0	1	1
0	0	1	0
0	0	1	0
0	0	1	0
1	0	1	0
1	0	1	0
1	0	1	0
1	0	1	0
0	0	1	0
0	0	1	0
0	0	1	0

Setelah melakukan *bootstrap*, selanjutnya adalah menghitung nilai *index gini* dari semua fitur dengan Persamaan 2.8. Dari perhitungan Persaamaan 2.8 didapatkan hasil *index gini* untuk masing-masing fitur. Hasil tersebut diperoleh dengan mempertimbangkan jumlah data pada setiap kelas berdasarkan *node* yang telah ditentukan melalui *threshold* masing-masing fitur.

Tabel 3. 33 Rincian Data *Govt Job Bootstrap* Ketiga

Govt Job Group	Stoke(1)	Tidak Stroke(0)	Total
0	9	6	15
1	1	4	5
total data			20

Tabel 3. 34 Rincian Data *Self Employed Bootstrap* Ketiga

Self Employed Group	Stoke(1)	Tidak Stroke(0)	Total
0	9	10	19
1	1	0	1
total data			20

Tabel 3. 35 Rincian Data *Residence Type Bootstrap* Ketiga

Residence Type Group	Stoke(1)	Tidak Stroke(0)	Total
0	5	0	5
1	5	10	15
total data			20

Setelah rincian data pada masing-masing fitur diketahui, langkah berikutnya adalah menghitung nilai probabilitas untuk setiap kelas pada masing-masing *node* berdasarkan *threshold* atau kategori fitur yang telah ditentukan. Probabilitas ini digunakan untuk menghitung *index gini* pada setiap fitur. Berikut adalah tabel probabilitas untuk fitur-fitur pada *bootstrap* ketiga.

Tabel 3. 36 Probabilitas *Fitur Govt Job Bootstrap* Ketiga

Node 1 = 0	
Probabilitas Stroke	0.6
Probabilitas Tidak Stroke	0.4
Node 2 = 1	
Probabilitas Stroke	0.2
Probabilitas Tidak Stroke	0.8

Tabel 3. 37 Probabilitas Fitur *Self Employed Bootstrap* Ketiga

Node 1 = 0	
Probabilitas Stroke	0.473684
Probabilitas Tidak Stroke	0.526316
Node 2 = 1	
Probabilitas Stroke	1
Probabilitas Tidak Stroke	0

Tabel 3. 38 Probabilitas Fitur *Residence Type Bootstrap* Ketiga

Node 1 = 0	
Probabilitas Stroke	1
Probabilitas Tidak Stroke	0
Node 2 = 1	
Probabilitas Stroke	0.333333
Probabilitas Tidak Stroke	0.666667

Langkah selanjutnya adalah perhitungan nilai *gini impurity* dari masing-masing fitur pada *bootstrap* ketiga dengan Persamaan 2.9, sehingga menghasilkan nilai seperti tabel-tabel berikut.

Tabel 3. 39 *Index Gini Impurity* Fitur *Govt Job Bootstrap* Ketiga

Node 1 = 0	0.48
Node 2 = 1	0.32
Gini Total	0.44

Tabel 3. 40 *Index Gini Impurity* Fitur *Self Employed Bootstrap* Ketiga

Node 1 = 0	0.498615
Node 2 = 1	0
Gini Total	0.473684

Tabel 3. 41 *Index Gini Residence Type Bootstrap* Ketiga

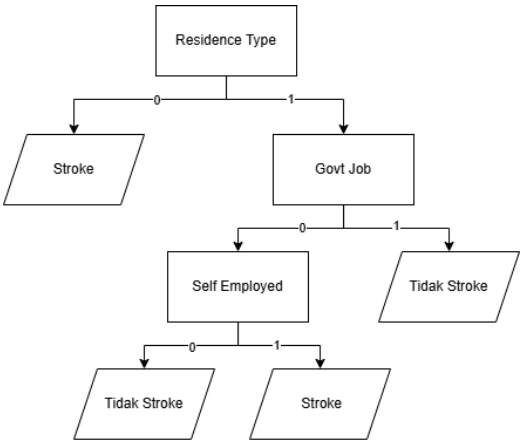
Node 1 = 0	0
Node 2 = 1	0.444444
Gini Total	0.333333

Setelah *index gini* dihitung untuk setiap fitur, langkah selanjutnya adalah melakukan pengurutan *index gini* dari nilai terkecil ke terbesar. Fitur dengan nilai *index gini* terkecil akan dipilih sebagai *root node* pada pohon keputusan. Tabel 3. 42 adalah hasil pengurutan *index gini* pada *bootstrap* ketiga.

Tabel 3. 42 Hasil *Sorting Index Gini Bootstrap* Ketiga

Fitur	**Gini Total**
Residence Type	0.333333
Govt Job	0.44
Self Employed	0.473684

Setelah pengurutan *index gini*. Fitur dengan nilai *index gini* terkecil akan dipilih sebagai *root node* pada pohon keputusan. Gambar 3. 8 adalah pohon keputusan yang terbentuk dari hasil pengurutan *index gini*



Gambar 3. 8 Pohon Keputusan *Bootstrap* Ketiga

4. Perhitungan *Index* Gini *Bootstrap* Keempat

Tabel 3. 43 Data Hasil *Bootstrap* Keempat

Bmi	Unknown	Formerly Smoked	Stroke
0.75	0	1	1
0.331226	0	0	1
0.527174	0	0	1
0.630435	0	0	1
0.065217	0	0	1
0.336957	0	1	1
0.25	0	0	1
0	0	0	1
0.331226	1	0	1
0.076087	1	0	1
0.717391	0	1	0
1	0	1	0
0.233696	0	0	0
0.141304	0	0	0
0.3717388	0	0	0
0.3369564	0	0	0
0.2172728	0	0	0
0.8304346	0	1	0
0.8869564	0	1	0
0.427174	0	0	0

Setelah melakukan *bootstrap*, selanjutnya adalah menghitung nilai *index gini* dari semua fitur dengan Persamaan 2.8. Dari perhitungan Persaamaan 2.8 didapatkan hasil *index gini* untuk masing-masing fitur. Hasil tersebut diperoleh

dengan mempertimbangkan jumlah data pada setiap kelas berdasarkan *node* yang telah ditentukan melalui *threshold* masing-masing fitur

Tabel 3. 44 Rincian Data Bmi *Bootstrap* Keempat

Bmi Group	Stoke(1)	Tidak Stroke(0)	Total
≤ 0.42	7	5	12
> 0.42	3	5	8
total data			20

Tabel 3. 45 Rincian Data *Unknown Bootstrap* Keempat

Unkown Group	Stoke(1)	Tidak Stroke(0)	Total
0	8	10	18
1	2	0	2
total data			20

Tabel 3. 46 Rincian Data *Formerly Smoked Bootstrap* Keempat

Formerly Smoked Group	Stoke(1)	Tidak Stroke(0)	Total
0	8	6	14
1	2	4	6
total data			20

Setelah rincian data pada masing-masing fitur diketahui, langkah berikutnya adalah menghitung nilai probabilitas untuk setiap kelas pada masing-masing *node* berdasarkan *threshold* atau kategori fitur yang telah ditentukan. Probabilitas ini digunakan untuk menghitung *index gini* pada setiap fitur. Berikut adalah tabel probabilitas untuk fitur-fitur pada bootstrap keempat.

Tabel 3. 47 Probabilitas Fitur Bmi *Bootstrap* Keempat

Node 1 ≤ 0.42	
Probabilitas Stroke	0.583333
Probabilitas Tidak Stroke	0.416667
Node 2 > 0.42	
Probabilitas Stroke	0.375
Probabilitas Tidak Stroke	0.625

Tabel 3. 48 Probabilitas Fitur *Unknown Bootstrap* Keempat

Node 1 = 0	
Probabilitas Stroke	0.444444
Probabilitas Tidak Stroke	0.555556
Node 2 = 1	
Probabilitas Stroke	1
Probabilitas Tidak Stroke	0

Tabel 3. 49 Probabilitas Fitur *Formerly Smoked Bootstrap* Keempat

Node 1 = 0	
Probabilitas Stroke	0.571429
Probabilitas Tidak Stroke	0.428571
Node 2 = 1	
Probabilitas Stroke	0.333333
Probabilitas Tidak Stroke	0.666667

Langkah selanjutnya adalah perhitungan nilai *gini impurity* dari masing-masing fitur pada *bootstrap* keempat dengan Persamaan 2.9, sehingga menghasilkan nilai seperti tabel-tabel berikut.

Tabel 3. 50 *Index Gini Impurity Bmi Bootstrap* Keempat

Node 1 $\leq$ 0.42	0.486111
Node 2 > 0.42	0.46875
Gini Total	0.479167

Tabel 3. 51 *Index Gini Impurity Unknown Bootstrap* Keempat

Node 1 = 0	0.493827
Node 2 = 1	0
Gini Total	0.444444

Tabel 3. 52 *Index Gini Formerly Smoked Bootstrap* Keempat

Node 1 = 0	0.489796
Node 2 = 1	0.444444
Gini Total	0.47619

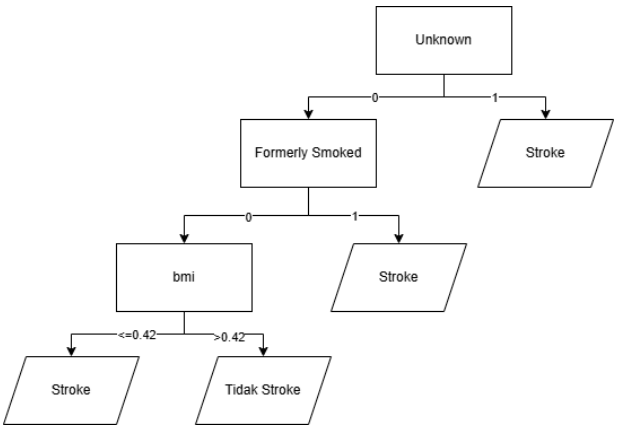
Setelah *index gini* dihitung untuk setiap fitur, langkah selanjutnya adalah melakukan pengurutan *index gini* dari nilai terkecil ke terbesar. Fitur dengan nilai

index gini terkecil akan dipilih sebagai *root node* pada pohon keputusan. Tabel 3. 53 adalah hasil pengurutan *index gini* pada *bootstrap* keempat.

Tabel 3. 53 Hasil *Sorting Index Gini Bootstrap* Keempat

Fitur	Gini Total
Unknown	0.44444
Formerly Smoked	0.47619
bmi	0.479167

Setelah pengurutan *index gini*. Fitur dengan nilai *index gini* terkecil akan dipilih sebagai *root node* pada pohon keputusan. Gambar 3. 9 adalah pohon keputusan yang terbentuk dari hasil pengurutan *index gini*.



Gambar 3. 9 Pohon Keputusan *Bootstrap* Keempat

5. Perhitungan *Index Gini Bootstrap* Kelima

Tabel 3. 54 Data Hasil *Bootstrap* Kelima

Average Glucose	Never Smokes	Smokes	stroke
1	0	0	1
0.844345	1	0	1
0.278333	1	0	1
0.662238	0	1	1
0.679226	1	0	1
0.750294	0	1	1
0.067717	1	0	1
0.210557	1	0	1
0.103339	0	0	1
0	0	0	1
0.111216	0	0	0

Average Glucose	Never Smokes	Smokes	stroke
0.040031	0	0	0
0.209323	1	0	0
0.221726	0	1	0
0.177522	0	1	0
0.3979308	0	1	0
0.4707736	0	1	0
0.082742	0	0	0
0.068505	0	0	0
0.1700802	0	0	0

Setelah melakukan *bootstrap*, selanjutnya adalah menghitung nilai *index gini* dari semua fitur dengan Persamaan 2.8. Dari perhitungan Persaamaan 2.8 didapatkan hasil *index gini* untuk masing-masing fitur. Hasil tersebut diperoleh dengan mempertimbangkan jumlah data pada setiap kelas berdasarkan *node* yang telah ditentukan melalui *threshold* masing-masing fitur.

Tabel 3. 55 Rincian Data *Average Glucose Bootstrap* Kelima

Avg Glucose Group	Stoke(1)	Tidak Stroke(0)	Total
≤ 0.32	5	8	13
> 0.32	5	2	7
total data			20

Tabel 3. 56 Rincian Data *Never Smoked Bootstrap* Kelima

Never Smoked Group	Stoke(1)	Tidak Stroke(0)	Total
0	5	9	14
1	5	1	6
total data			20

Tabel 3. 57 Rincian Data *Smokes Bootstrap* Kelima

Smokes Group	Stoke(1)	Tidak Stroke(0)	Total
0	8	6	14
1	2	4	6
total data			20

Setelah rincian data pada masing-masing fitur diketahui, langkah berikutnya adalah menghitung nilai probabilitas untuk setiap kelas pada masing-masing *node*

berdasarkan *threshold* atau kategori fitur yang telah ditentukan. Probabilitas ini digunakan untuk menghitung *index gini* pada setiap fitur. Berikut adalah tabel probabilitas untuk fitur-fitur pada *bootstrap* kelima

Tabel 3. 58 Probabilitas Fitur *Average Glucose Bootstrap* Kelima

Node 1 ≤ 0.32	
Probabilitas Stroke	0.384615
Probabilitas Tidak Stroke	0.615385
Node 2 > 0.32	
Probabilitas Stroke	0.714286
Probabilitas Tidak Stroke	0.285714

Tabel 3. 59 Probabilitas Fitur *Never Smoked Bootstrap* Kelima

Node 1 = 0	
Probabilitas Stroke	0.357143
Probabilitas Tidak Stroke	0.642857
Node 2 = 1	
Probabilitas Stroke	0.833333
Probabilitas Tidak Stroke	0.166667

Tabel 3. 60 Probabilitas Fitur *Smokes Bootstrap* Kelima

Node 1 = 0	
Probabilitas Stroke	0.571429
Probabilitas Tidak Stroke	0.428571
Node 2 = 1	
Probabilitas Stroke	0.333333
Probabilitas Tidak Stroke	0.666667

Langkah selanjutnya adalah perhitungan nilai *gini impurity* dari masing-masing fitur pada *bootstrap* pertama dengan Persamaan 2.9, sehingga menghasilkan nilai seperti tabel-tabel berikut:

Tabel 3. 61 *Index Gini Impurity Average Glucose Bootstrap* Kelima

Node 1 ≤ 0.32	0.473373
Node 2 > 0.32	0.408163
Gini Total	0.450549

Tabel 3. 62 *Index Gini Impurity Never Smoked Bootstrap* Kelima

Node 1 = 0	0.459184
Node 2 = 1	0.277778
Gini Total	0.404762

Tabel 3. 63 *Index Gini Smokes Bootstrap* Kelima

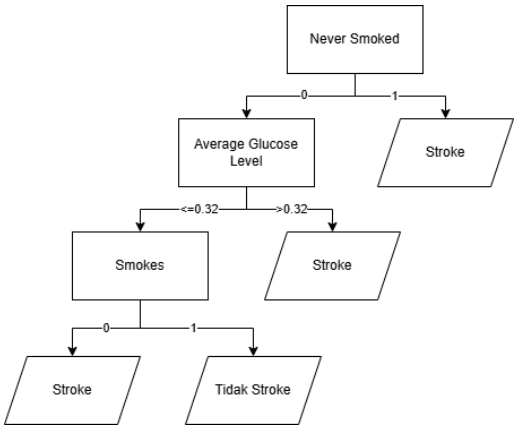
Node 1 = 0	0.489796
Node 2 = 1	0.444444
Gini Total	0.47619

Setelah indeks gini dihitung untuk setiap fitur, langkah selanjutnya adalah melakukan pengurutan indeks gini dari nilai terkecil ke terbesar. Fitur dengan nilai indeks gini terkecil akan dipilih sebagai root node pada pohon keputusan. Tabel 3. 64 adalah hasil pengurutan indeks gini pada bootstrap kelima

Tabel 3. 64 Hasil *Sorting Index Gini Bootstrap* Kelima

Fitur	**Gini Total**
Never Smoked	0.404762
Average Glucose	0.450549
Smokes	0.47619

Setelah pengurutan *index gini*. Fitur dengan nilai *index gini* terkecil akan dipilih sebagai *root node* pada pohon keputusan. Gambar 3. 10 adalah pohon keputusan yang terbentuk dari hasil pengurutan *index gini.*



Gambar 3. 10 Pohon Keputusan *Bootstrap* Kelima

Langkah terakhir adalah menentukan hasil klasifikasi berdasarkan suara terbanyak(*voting*) dari gabungan pohon Keputusan yang sudah ada.

3.7 Evaluasi Model

Evaluasi model pada penelitian ini menggunakan metode *confusion matrix.* Pada umumnya, *confusion matrix* digunakan untuk evaluasi model klasifikasi pada *dataset* (Hasnain dkk., 2020). *Confusion matrix* ditampilkan dalam matrik 2 x 2, yang mana didalamnya terdapat 4 ukuran, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).

Tabel 3. 65 *Confusion Matrix*

		Nilai Aktual	
		Positive	*Negative*
Nilai Prediksi	*Positive*	TP	FN
	Negative	FP	TN

True Positives (TP) merupakan jumlah data yang sebenarnya berlabel tidak terkena stroke (label 0) dan berhasil diprediksi dengan benar sebagai tidak terkena stroke oleh model. *False Positives* (FP) merupakan jumlah data yang sebenarnya terkena stroke (label 1) tetapi salah diprediksi sebagai tidak terkena stroke (label 0) oleh model. *False Negatives* (FN) merupakan jumlah data yang sebenarnya tidak terkena stroke (label 0) tetapi salah diprediksi sebagai terkena stroke (label 1) oleh model. *True Negatives* (TN) merupakan jumlah data yang sebenarnya terkena stroke (label 1) dan berhasil diprediksi dengan benar sebagai terkena stroke oleh model. Tabel 3. 65 digunakan untuk mengetahui nilai akurasi, presisi, *recall*, dan *F1-Score*.

1. Akurasi

Akurasi adalah presentase untuk mengukur seberapa sering model membuat prediksi yang benar. Persamaan 3.3 digunakan untuk mengetahui nilai akurasi.

$$Akurasi = \frac{TP + TN}{TP + FN + FP + TN} \quad (3.3)$$

2. Presisi

Presisi adalah presentase untuk mengukur seberapa rasio antara jumlah prediksi positif yang benar terhadap semua prediksi positif. Persamaan 3.4 digunakan untuk mengetahui nilai presisi

$$Presisi = \frac{TP}{TP + FP} \quad (3.4)$$

3. Recall

Recall adalah presentase untuk mengukur rasio antara jumlah prediksi positif yang benar terhadap seluruh data positif yang sebenarnya.

$$Recall = \frac{TP}{TP + FN} \quad (3.5)$$

4. *F1-Score*

F1-Score adalah rata-rata dari presisi dan *recall* dengan kata lain, *F1-Score* menunjukkan nilai keseimbangan antara seberapa tepat model memprediksi sesuatu yang benar dan seberapa baik model menemukan semua yang benar.

$$F1 - Score = \frac{2 \times Recall \times Presisi}{Recall + Presisi} \quad (3.6)$$

Berdasarkan Penelitian Fluorida Fibrianda & Bhawiyuga, (2018) , nilai performa dalam klasifikasi dapat dibagi menjadi beberapa kelompok, yaitu:

1. 0,90 – 1,00 = Klasifikasi Sangat Baik
2. 0,80 – 0,90 = Klasifikasi Baik
3. 0,70 – 0,80 = Klasifikasi Cukup
4. 0,60 – 0,70 = Klasifikasi Lemah
5. 0,50 – 0,60 = Klasifikasi Gagal

Tabel 3. 66 Data *Confusion Matrix*

		Nilai Aktual	
		Positive	*Negative*
Nilai Prediksi	*Positive*	3	2
	Negative	7	3

Berikut contoh perhitungan manual *confusion matrix* berdasarkan matrix pada Tabel 3. 66

$$\text{Akurasi} = \frac{3+3}{3+2+7+3} = 0.4$$

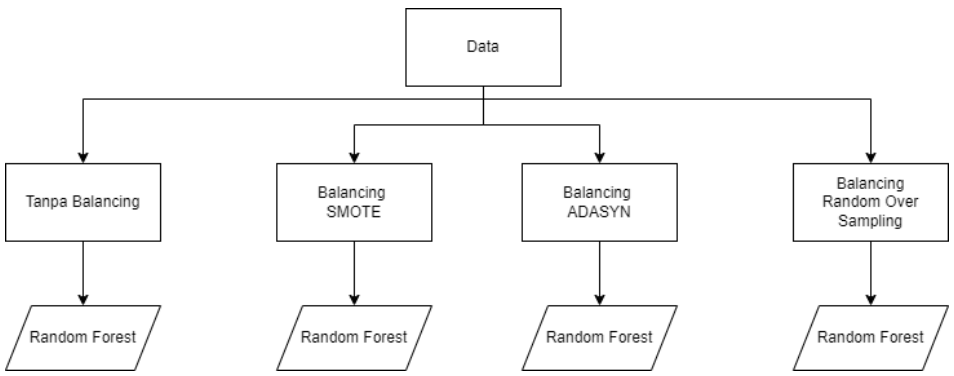
$$\text{Presisi} = \frac{3}{3+7} = 0.3$$

$$Recall = \frac{3}{3+2} = 0.6$$

$$F1 - Score = \frac{2 \times 0.6 \times 0.3}{0.6+0.3} = 0.4$$

3.8 Skenario Pengujian

Skenario yang dilakukan untuk pengujian pada penelitian ini ditunjukkan oleh Gambar 3. 11.



Gambar 3. 11 Skema Skenario Pengujian

Pada skenario pengujian seperti Gambar 3. 11, data akan melalui berbagai metode *balancing*, yaitu tanpa *balancing*, SMOTE, ADASYN, dan ROS. Setiap metode *balancing* kemudian diuji menggunakan algoritma, *Random Forest*. Selanjutnya,, pada algoritma *random forest* dilakukan *hyperparameter tuning* untuk mendapatkan performa terbaik. Parameter yang akan dioptimasi pada algoritma *random forest* meliputi (Awliya Muhammad Ashfania dkk., 2022):

1. n_estimators: Jumlah pohon atau jumlah *bootstrap*.
2. max_depth: Kedalaman maksimum pohon yang mana kedalaman pohon diukur dari *root node*(akar) hingga *leaf node*(daun)

Tabel 3. 67 Model Skenario Pengujian

Data	**Model**	***Random Forest***	
		N_Estimator	**Max_Depth**
Tanpa *Balancing*	1	10	3
	2	10	6
	3	50	3
	4	50	6
	5	100	3
	6	100	6
SMOTE (KNN=3)	7	10	3
	8	10	6
	9	50	3
	10	50	6
	11	100	3
	12	100	6
SMOTE (KNN=5)	13	10	3
	14	10	6
	15	50	3
	16	50	6
	17	100	3
	18	100	6
ADASYN (KNN=3)	19	10	3
	20	10	6
	21	50	3
	22	50	6
	23	100	3
	24	100	6
ADASYN (KNN=5)	25	10	3
	26	10	6

Data	Model	*Random Forest*	
		N_Estimator	**Max_Depth**
	27	50	3
	28	50	6
	29	100	3
	30	100	6
Random Over Sampling	31	10	3
	32	10	6
	33	50	3
	34	50	6
	35	100	3
	36	100	6

BAB IV

UJI COBA DAN PEMBAHASAN

4.1 Hasil Uji Coba Tanpa *Balancing Data*

Pada pengujian ini *dataset* yang digunakan tidak melalui proses *balancing*. Selain itu, pengujian terdiri dari 6 model yakni dengan menerapkan *tuning* parameter pada algoritma *random forest*.

4.1.1 Model Tanpa *Balancing* dengan 10 Pohon dan Kedalaman 3

Pada model ini algoritma *random forest* menggunakan parameter *n_estimator* dengan nilai 10 dan *max_depth* dengan nilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.1

Tabel 4. 1 *Confusion* Matrix Model 1 Tanpa *Balancing*

TP	FP	TN	FN
841	36	0	0

Dari *confusion matrix* pada Tabel 4.1 didapatkan nilai akurasi sebesar 96%, nilai presisi sebesar 96%, nilai *recall* sebesar 100%, dan nilai *F1-Score* sebesar 98%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 48%, *recall macro average* sebesar 50%, dan *f1-score macro average* 49%.

4.1.2 Model Tanpa *Balancing* dengan 10 Pohon dan Kedalaman 6

Pada model ini algoritma *Random Forest* menggunakan parameter *n_estimator* dengan nilai 10 dan *max_depth* dengan nilai 6. Dari Model ini didapatkan hasil *confusion matrix* pada Tabel 4.2

Tabel 4. 2 *Confusion Matrix* Model 2 Tanpa *Balancing*

TP	FP	TN	FN
838	36	0	3

Dari *confusion matrix* pada Tabel 4.2 didapatkan nilai akurasi sebesar 96%, nilai presisi sebesar 96%, nilai *recall* sebesar 100%, dan nilai *F1-Score* sebesar 98%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 48%, *recall macro average* sebesar 50%, dan *f1-score macro average* 49%.

4.1.3 Model Tanpa *Balancing* dengan 50 Pohon dan Kedalaman 3

Pada model ini algoritma *random forest* menggunakan parameter *n_estimator* dengan nilai 50 dan *max_depth* dengan nilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.3

Tabel 4. 3 *Confusion Matrix* Model 3 Tanpa *Balancing*

TP	FP	TN	FN
841	36	0	0

Dari *confusion matrix* pada Tabel 4.3 didapatkan nilai akurasi sebesar 96%, nilai presisi sebesar 96%, nilai *recall* sebesar 100%, dan nilai *F1-Score* sebesar 98%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 48%, *recall macro average* sebesar 50%, dan *f1-score macro average* 49%.

4.1.4 Model Tanpa *Balancing* dengan 50 Pohon dan Kedalaman 6

Pada model ini algoritma *random forest* menggunakan parameter *n_estimator* dengan nilai 50 dan *max_depth* dengan nilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.4

Tabel 4. 4 *Confusion Matrix* Model 4 Tanpa *Balancing*

TP	FP	TN	FN
841	36	0	0

Dari *confusion matrix* pada Tabel 4.4 didapatkan nilai akurasi sebesar 96%, nilai presisi sebesar 96%, nilai *recall* sebesar 100%, dan nilai *F1-Score* sebesar 98%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 48%, *recall macro average* sebesar 50%, dan *f1-score macro average* 49%.

4.1.5 Model Tanpa *Balancing* dengan 100 Pohon dan Kedalaman 3

Pada model ini algoritma *random forest* menggunakan parameter *n_estimator* dengan nilai 100 dan *max_depth* dengan nilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.5

Tabel 4. 5 *Confusion Matrix* Model 5 Tanpa *Balancing*

TP	FP	TN	FN
841	36	0	0

Dari *confusion matrix* pada Tabel 4.5 didapatkan nilai akurasi sebesar 96%, nilai presisi sebesar 96%, nilai *recall* sebesar 100%, dan nilai *F1-Score* sebesar 98%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro averag*e sebesar 48%, *recall macro average* sebesar 50%, dan *f1-score macro average* 49%.

4.1.6 Model Tanpa *Balancing* dengan 100 Pohon dan Kedalaman 6

Pada model ini algoritma *random forest* menggunakan parameter *n_estimator* dengan nilai 100 dan *max_depth* dengan nilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.6

Tabel 4. 6 *Confusion Matrix* Model 6 Tanpa *Balancing*

TP	FP	TN	FN
840	36	0	1

Dari *confusion matrix* pada Tabel 4.6 didapatkan nilai akurasi sebesar 96%, nilai presisi sebesar 96%, nilai *recall* sebesar 100%, dan nilai *F1-Score* sebesar 98%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 48%, *recall macro average* sebesar 50%, dan *f1-score macro average* 49%.

4.2 Hasil Uji Menggunakan SMOTE Dengan Nilai KNN 3

Pada pengujian ini *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan parameter KNN bernilai 3. Selain itu, pengujian terdiri dari 6 model yakni dengan menerapkan *tuning* parameter pada algoritma *Random Forest*.

4.2.1 Model SMOTE KNN 3, 10 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.7

Tabel 4. 7 *Confusion Matrix* Model 7 SMOTE (KNN 3)

TP	**FP**	**TN**	**FN**
575	75	739	275

Dari *confusion matrix* pada Tabel 4.7 didapatkan nilai akurasi sebesar 79%, nilai presisi sebesar 88%, nilai *recall* sebesar 68%, dan nilai *F1-Score* sebesar 77%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 81%, *recall macro average* sebesar 79%, dan *f1-score macro average* 79%.

4.2.2 Model SMOTE KNN 3, 10 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.8

Tabel 4. 8 *Confusion Matrix* Model 8 SMOTE (KNN 3)

TP	FP	TN	FN
647	34	780	203

Dari *confusion matrix* pada Tabel 4.8 didapatkan nilai akurasi sebesar 86%, nilai presisi sebesar 95%, nilai *recall* sebesar 76%, dan nilai *F1-Score* sebesar 85%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 87%, *recall macro average* sebesar 86%, dan *f1-score macro average* 86%.

4.2.3 Model SMOTE KNN 3, 50 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *Random Forest* dengan parameter *n_estimator*

bernilai 50 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.9

Tabel 4. 9 *Confusion Matrix* Model 9 SMOTE (KNN 3)

TP	FP	TN	FN
573	69	745	277

Dari *confusion matrix* pada Tabel 4.9 didapatkan nilai akurasi sebesar 79%, nilai presisi sebesar 89%, nilai *recall* sebesar 67%, dan nilai *F1-Score* sebesar 77%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 81%, *recall macro average* sebesar 79%, dan *f1-score macro average* 79%.

4.2.4 Model SMOTE KNN 3, 50 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.10

Tabel 4. 10 *Confusion Matrix* Model 10 SMOTE (KNN 3)

TP	FP	TN	FN
642	32	782	208

Dari *confusion matrix* pada Tabel 4.10 didapatkan nilai akurasi sebesar 86%, nilai presisi sebesar 95%, nilai *recall* sebesar 76%, dan nilai *F1-Score* sebesar 84%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 87%, *recall macro average* sebesar 86%, dan *f1-score macro average* 85%.

4.2.5 Model SMOTE KNN 3, 100 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.11

Tabel 4. 11 *Confusion Matrix* Model 11 SMOTE (KNN 3)

TP	FP	TN	FN
578	62	752	272

Dari *confusion matrix* pada Tabel 4.11 didapatkan nilai akurasi sebesar 80%, nilai presisi sebesar 90%, nilai *recall* sebesar 68%, dan nilai *F1-Score* sebesar 78%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 82%, *recall macro average* sebesar 80%, dan *f1-score macro average* 80%.

4.2.6 Model SMOTE KNN 3, 100 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.12

Tabel 4. 12 *Confusion Matrix* Model 12 SMOTE (KNN 3)

TP	FP	TN	FN
665	34	780	185

Dari *confusion matrix* pada Tabel 4.12 didapatkan nilai akurasi sebesar 87%, nilai presisi sebesar 95%, nilai *recall* sebesar 78%, dan nilai *F1-Score* sebesar 86%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 88%, *recall macro average* sebesar 87%, dan *f1-score macro average* 87%.

4.3 Hasil Uji Coba Menggunakan SMOTE Dengan Nilai KNN 5

Pada pengujian ini *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan parameter KNN bernilai 5. Selain itu, pengujian terdiri dari 6 model yakni dengan menerapkan *tuning* parameter pada algoritma *random forest*.

4.3.1 Model SMOTE KNN 5, 10 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.13

Tabel 4. 13 *Confusion Matrix* Model 13 SMOTE (knn-5)

TP	FP	TN	FN
586	80	734	264

Dari *Confusion Matrix* pada Tabel 4.13 didapatkan nilai akurasi sebesar 79%, nilai presisi sebesar 88%, nilai *recall* sebesar 69%, dan nilai *F1-Score* sebesar 77%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro*

average sebesar 81%, *recall macro average* sebesar 80%, dan *f1-score macro average* 79%.

4.3.2 Model SMOTE KNN 5, 10 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.14

Tabel 4. 14 *Confusion Matrix* Model 14 SMOTE (knn-5)

TP	FP	TN	FN
632	26	788	218

Dari *confusion matrix* pada Tabel 4.14 didapatkan nilai akurasi sebesar 85%, nilai presisi sebesar 96%, nilai *recall* sebesar 74%, dan nilai *F1-Score* sebesar 84%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 87%, *recall macro average* sebesar 86%, dan *f1-score macro average* 85%.

4.3.3 Model SMOTE KNN 5, 50 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *Random Forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.15

Tabel 4. 15 *Confusion Matrix* Model 15 SMOTE (knn-5)

TP	FP	TN	FN
573	71	743	277

Dari *confusion matrix* pada Tabel 4.15 didapatkan nilai akurasi sebesar 79%, nilai presisi sebesar 89%, nilai *recall* sebesar 67%, dan nilai *F1-Score* sebesar 77%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 81%, *recall macro average* sebesar 79%, dan *f1-score macro average* 79%.

4.3.4 Model SMOTE KNN 5, 50 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.16

Tabel 4. 16 *Confusion Matrix* Model 16 SMOTE (knn-5)

TP	FP	TN	FN
658	38	776	192

Dari *confusion matrix* pada Tabel 4.16 didapatkan nilai akurasi sebesar 86%, nilai presisi sebesar 95%, nilai *recall* sebesar 77%, dan nilai *F1-Score* sebesar 85%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 87%, *recall macro average* sebesar 86%, dan *f1-score macro average* 86%.

4.3.5 Model SMOTE KNN 5, 100 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.17

Tabel 4. 17 *Confusion Matrix* Model 17 SMOTE (knn-5)

TP	FP	TN	FN
578	62	752	272

Dari *confusion matrix* pada Tabel 4.17 didapatkan nilai akurasi sebesar 80%, nilai presisi sebesar 90%, nilai *recall* sebesar 68%, dan nilai *F1-Score* sebesar 78%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 82%, *recall macro average* sebesar 80%, dan *f1-score macro average* 80%.

4.3.6 Model SMOTE KNN 5, 100 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode SMOTE dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.18

Tabel 4. 18 *Confusion Matrix* Model 18 SMOTE (knn-5)

TP	FP	TN	FN
633	29	785	217

Dari *confusion matrix* pada Tabel 4.18 didapatkan nilai akurasi sebesar 85%, nilai presisi sebesar 96%, nilai *recall* sebesar 74%, dan nilai *F1-Score* sebesar 84%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 87%, *recall macro average* sebesar 85%, dan *f1-score macro average* 85%.

4.4 Hasil Uji Coba Menggunakan ADASYN Dengan Nilai KNN 3

Pada pengujian ini *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan parameter KNN bernilai 3. Selain itu, pengujian terdiri dari 6 model yakni dengan menerapkan *tuning* parameter pada algoritma *random forest*.

4.4.1 Model ADASYN KNN 3, 10 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.19

Tabel 4. 19 *Confusion Matrix* Model 19 ADASYN (knn-3)

TP	FP	TN	FN
532	59	784	312

Dari *confusion matrix* pada Tabel 4.19 didapatkan nilai akurasi sebesar 78%, nilai presisi sebesar 90%, nilai *recall* sebesar 63%, dan nilai *F1-Score* sebesar 74%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 81%, *recall macro average* sebesar 78%, dan *f1-score macro average* 78%.

4.4.2 Model ADASYN KNN 3, 10 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.20

Tabel 4. 20 *Confusion Matrix* Model 20 ADASYN (knn-3)

TP	FP	TN	FN
616	57	786	228

Dari *confusion matrix* pada Tabel 4.20 didapatkan nilai akurasi sebesar 83%, nilai presisi sebesar 92%, nilai *recall* sebesar 73%, dan nilai *F1-Score* sebesar 81%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 85%, *recall macro average* sebesar 83%, dan *f1-score macro average* 83%.

4.4.3 Model ADASYN KNN 3, 50 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.21

Tabel 4. 21 *Confusion Matrix* Model 21 ADASYN (knn-3)

TP	FP	TN	FN
525	55	788	319

Dari *confusion matrix* pada Tabel 4.21 didapatkan nilai akurasi sebesar 78%, nilai presisi sebesar 91%, nilai *recall* sebesar 62%, dan nilai *F1-Score* sebesar 74%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 81%, *recall macro average* sebesar 78%, dan *f1-score macro average* 77%.

4.4.4 Model ADASYN KNN 3, 50 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.22

Tabel 4. 22 *Confusion Matrix* Model 22 ADASYN (knn-3)

TP	FP	TN	FN
658	38	776	192

Dari *confusion matrix* pada Tabel 4.22 didapatkan nilai akurasi sebesar 86%, nilai presisi sebesar 95%, nilai *recall* sebesar 77%, dan nilai *F1-Score* sebesar 85%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 87%, *recall macro average* sebesar 86%, dan *f1-score macro average* 86%.

4.4.5 Model ADASYN KNN 3, 100 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter

n_estimator bernilai 100 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.23

Tabel 4. 23 *Confusion Matrix* Model 23 ADASYN (knn-3)

TP	FP	TN	FN
578	62	752	272

Dari *confusion matrix* pada Tabel 4.23 didapatkan nilai akurasi sebesar 80%, nilai presisi sebesar 90%, nilai *recall* sebesar 68%, dan nilai *F1-Score* sebesar 78%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 82%, *recall macro average* sebesar 80%, dan *f1-score macro average* 80%.

4.4.6 Model ADASYN KNN 3, 100 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 3, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.24

Tabel 4. 24 *Confusion Matrix* Model 24 ADASYN (knn-3)

TP	FP	TN	FN
627	50	793	217

Dari *confusion matrix* pada Tabel 4.24 didapatkan nilai akurasi sebesar 84%, nilai presisi sebesar 93%, nilai *recall* sebesar 74%, dan nilai *F1-Score* sebesar 82%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 86%, *recall macro average* sebesar 84%, dan *f1-score macro average* 84%.

4.5 Hasil Uji Coba Menggunakan ADASYN Dengan Nilai KNN 5

Pada pengujian ini *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan parameter KNN bernilai 5. Selain itu, pengujian terdiri dari 6 model yakni dengan menerapkan *tuning* parameter pada algoritma *random forest*.

4.5.1 Model ADASYN KNN 5, 10 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.25

Tabel 4. 25 *Confusion Matrix* Model 25 ADASYN (knn-5)

TP	FP	TN	FN
570	56	787	276

Dari *confusion matrix* pada Tabel 4.25 didapatkan nilai akurasi sebesar 80%, nilai presisi sebesar 91%, nilai *recall* sebesar 67%, dan nilai *F1-Score* sebesar 77%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 83%, *recall macro average* sebesar 80%, dan *f1-score macro average* 80%.

4.5.2 Model ADASYN KNN 5, 10 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter

n_estimator bernilai 10 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.26

Tabel 4. 26 *Confusion Matrix* Model 26 ADASYN (knn-5)

TP	FP	TN	FN
621	40	803	225

Dari *confusion matrix* pada Tabel 4.26 didapatkan nilai akurasi sebesar 84%, nilai presisi sebesar 94%, nilai *recall* sebesar 73%, dan nilai *F1-Score* sebesar 82%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 86%, *recall macro average* sebesar 84%, dan *f1-score macro average* 84%.

4.5.3 Model ADASYN KNN 5, 50 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.27

Tabel 4. 27 *Confusion Matrix* Model 27 ADASYN (knn-5)

TP	FP	TN	FN
542	41	802	304

Dari *confusion matrix* pada Tabel 4.27 didapatkan nilai akurasi sebesar 80%, nilai presisi sebesar 93%, nilai *recall* sebesar 64%, dan nilai *F1-Score* sebesar 76%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 83%, *recall macro average* sebesar 80%, dan *f1-score macro average* 79%.

4.5.4 Model ADASYN KNN 5, 50 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.28

Tabel 4. 28 *Confusion Matrix* Model 28 ADASYN (knn-5)

TP	FP	TN	FN
611	33	810	235

Dari *confusion matrix* pada Tabel 4.28 didapatkan nilai akurasi sebesar 84%, nilai presisi sebesar 95%, nilai *recall* sebesar 72%, dan nilai *F1-Score* sebesar 82%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 86%, *recall macro average* sebesar 84%, dan *f1-score macro average* 84%.

4.5.5 Model ADASYN KNN 5, 100 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.29

Tabel 4. 29 *Confusion Matrix* Model 29 ADASYN (knn-5)

TP	FP	TN	FN
524	41	802	304

Dari *confusion matrix* pada Tabel 4.29 didapatkan nilai akurasi sebesar 79%, nilai presisi sebesar 93%, nilai *recall* sebesar 63%, dan nilai *F1-Score* sebesar 75%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 83%, *recall macro average* sebesar 80%, dan *f1-score macro average* 79%.

4.5.6 Model ADASYN KNN 5, 100 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ADASYN dengan KNN bernilai 5, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada tabel 4.30

Tabel 4. 30 *Confusion Matrix* Model 30 ADASYN (knn-5)

TP	FP	TN	FN
606	29	814	240

Dari *confusion matrix* pada tabel 4.30 didapatkan nilai akurasi sebesar 84%, nilai presisi sebesar 95%, nilai *recall* sebesar 72%, dan nilai *F1-Score* sebesar 82%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 86%, *recall macro average* sebesar 84%, dan *f1-score macro average* 84%.

4.6 Hasil Uji Coba Menggunakan ROS

Pada pengujian ini *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode *random over sampling*. Selain itu, pengujian terdiri dari 6 model yakni dengan menerapkan *tuning* parameter pada algoritma *random forest*.

4.6.1 Model ROS dengan 10 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ROS, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.31

Tabel 4. 31 *Confusion Matrix* Model 31 ROS

TP	FP	TN	FN
542	87	750	309

Dari *confusion matrix* pada Tabel 4.31 didapatkan nilai akurasi sebesar 77%, nilai presisi sebesar 86%, nilai *recall* sebesar 64%, dan nilai *F1-Score* sebesar 73%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 78%, *recall macro average* sebesar 77%, dan *f1-score macro average* 76%.

4.6.2 Model ROS dengan 10 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ROS, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 10 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.32

Tabel 4. 32 *Confusion Matrix* Model 32 ROS

TP	FP	TN	FN
652	27	810	199

Dari *Confusion Matrix* pada Tabel 4.32 didapatkan nilai akurasi sebesar 87%, nilai presisi sebesar 96%, nilai *recall* sebesar 77%, dan nilai *F1-Score* sebesar 85%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro*

average sebesar 88%, *recall macro average* sebesar 87%, dan *f1-score macro average* 86%.

4.6.3 Model ROS dengan 50 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ROS, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 3. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.33

Tabel 4. 33 *Confusion Matrix* Model 33 ROS

TP	FP	TN	FN
586	106	731	265

Dari *confusion matrix* pada Tabel 4.33 didapatkan nilai akurasi sebesar 78%, nilai presisi sebesar 85%, nilai *recall* sebesar 69%, dan nilai *F1-Score* sebesar 76%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 79%, *recall macro average* sebesar 78%, dan *f1-score macro average* 78%.

4.6.4 Model ROS dengan 50 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ROS, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 50 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.34

Tabel 4. 34 *Confusion Matrix* Model 34 ROS

TP	FP	TN	FN
637	10	827	214

Dari *confusion matrix* pada Tabel 4.34 didapatkan nilai akurasi sebesar 87%, nilai presisi sebesar 98%, nilai *recall* sebesar 75%, dan nilai *F1-Score* sebesar 85%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 89%, *recall macro average* sebesar 87%, dan *f1-score macro average* 87%.

4.6.5 Model ROS dengan 100 Pohon dan Kedalaman 3

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ROS, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 3. Dari model ini didapatkan *hasil confusion matrix* pada Tabel 4.35

Tabel 4. 35 *Confusion Matrix* Model 35 ROS

TP	FP	TN	FN
595	118	719	256

Dari *confusion matrix* pada Tabel 4.35 didapatkan nilai akurasi sebesar 78%, nilai presisi sebesar 83%, nilai *recall* sebesar 70%, dan nilai *F1-Score* sebesar 76%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 79%, *recall macro average* sebesar 78%, dan *f1-score macro average* 78%.

4.6.6 Model ROS dengan 100 Pohon dan Kedalaman 6

Pada model ini kelas pada *dataset* yang tidak seimbang dilakukan proses *balancing* menggunakan metode ROS, selanjutnya akan dilakukan pengujian pada algoritma *random forest* dengan parameter *n_estimator* bernilai 100 dan *max_depth* bernilai 6. Dari model ini didapatkan hasil *confusion matrix* pada Tabel 4.36

Tabel 4. 36 *Confusion Matrix* Model 36 ROS

TP	FP	TN	FN
647	18	819	204

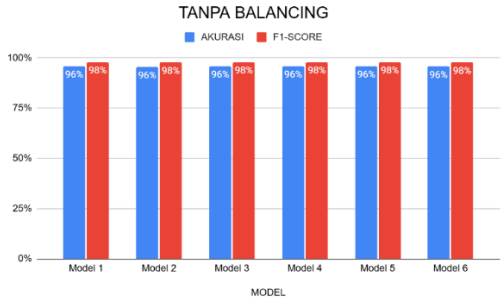
Dari *confusion matrix* pada Tabel 4.36 didapatkan nilai akurasi sebesar 87%, nilai presisi sebesar 97%, nilai *recall* sebesar 76%, dan nilai *F1-Score* sebesar 85%. Selain itu terdapat beberapa hasil lain yang didapatkan meliputi presisi *macro average* sebesar 89%, *recall macro average* sebesar 87%, dan *f1-score macro average* 87%.

4.7 Analisa Hasil Uji Coba

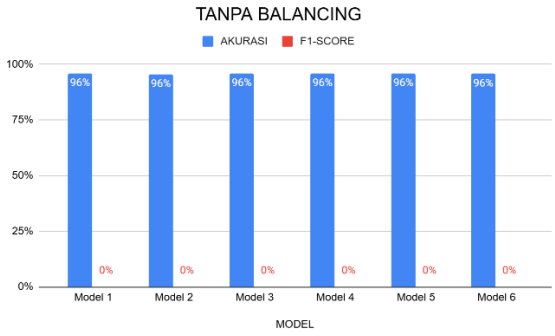
Pada penelitian ini, dilakukan pengujian untuk mengevaluasi performa tiga metode *balancing* data dalam mengatasi ketidakseimbangan kelas, dengan *random forest* sebagai algoritma klasifikasi. Setelah pengujian dilakukan, diperoleh hasil analisis sebagai berikut.

4.7.1 Analisa Pengujian Tanpa *Balancing*

Model yang tidak menggunakan metode *balancing* data memiliki akurasi tinggi, sebesar 96% untuk label positif (label 0). Namun, karena data tidak seimbang, performa *f1-score* untuk label negatif (stroke) sangat rendah, yaitu 0%, yang menunjukkan bahwa model tidak akurat dalam mendeteksi label minoritas., sehingga kasus stroke tidak terdeteksi dengan baik yang mana seperti pada grafik Gambar 4. 2. Dari hasil uji coba bisa disimpulkan bahwa tanpa *balancing* data, model kesulitan mengidentifikasi label minoritas, sehingga metode *balancing* diperlukan untuk meningkatkan performa pada label negatif.



Gambar 4. 1 Grafik Hasil Uji Coba Tanpa *Balancing* (Tidak Stroke)



Gambar 4. 2 Grafik Hasil Uji Coba Tanpa *Balancing* (Stroke)

Tabel 4. 37 Pengaruh Parameter *n_estimator* Tanpa *Balancing*

n_estimator	Average akurasi	Average presisi	Average recall	Average F1-Score
10	96%	96%	100%	98%
50	96%	96%	100%	98%
100	96%	96%	100%	98%

Tabel 4. 38 Pengaruh Parameter *Max_depth* Tanpa *Balancing*

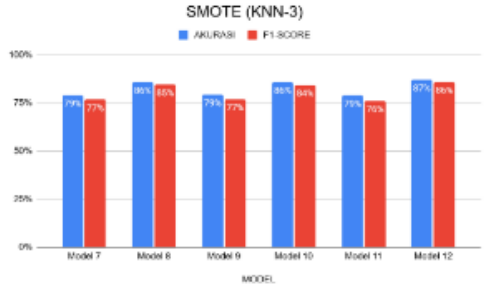
Max_depth	Average akurasi	Average presisi	Average recall	Average F1-Score
3	96%	96%	100%	98%
6	96%	96%	100%	98%

Berdasarkan Tabel 4. 37 menunjukkan bahwa parameter *n_estimators* tidak berpengaruh terhadap performa model yang ditunjukkan dengan tidak ada peningkatan performa dengan *dataset* tanpa dilakukannya *balancing* dalam mendeteksi kasus stroke. Sedangkan, pada Tabel 4. 38 menunjukkan bahwa jumlah

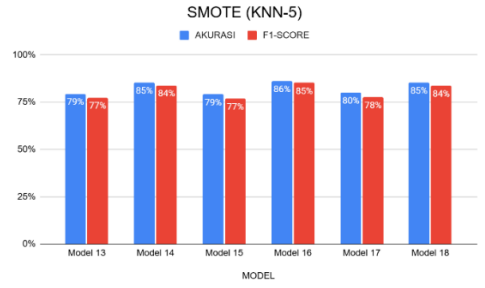
max_depth juga tidak berpengaruh terhadap performa model yang ditunjukkan dengan tidak adanya peningkatan performa.

4.7.2 Analisa Hasil Balancing Data Metode SMOTE

Penggunaan SMOTE dengan variasi nilai K pada KNN (K=3 dan K=5) serta pengaturan parameter *random forest* yakni *n_estimators* dan *max_depth* menunjukkan peningkatan performa terhadap kinerja model dalam mendeteksi label stroke. Meskipun akurasi keseluruhan sedikit menurun dibandingkan dengan model tanpa *balancing* data, namun nilai, *F1-score* untuk label stroke menunjukkan peningkatan yang berarti menunjukkan klasifikasi yang lebih baik pada kelas minoritas.



Gambar 4. 3 Grafik Hasil Uji Coba SMOTE (KNN 3)



Gambar 4. 4 Grafik Hasil Uji Coba SMOTE (KNN 5)

Tabel 4. 39 Pengaruh Parameter *n_estimator* (SMOTE k=3)

n_estimator	Average akurasi	Average presisi	Average recall	Average F1-Score
10	82%	92%	72%	81%
50	82%	92%	71%	81%
100	83%	93%	72%	81%

Tabel 4. 40 Pengaruh Parameter *n_estimator* (SMOTE k=5)

n_estimator	Average akurasi	Average presisi	Average recall	Average F1-Score
10	82%	92%	72%	81%
50	83%	92%	72%	81%
100	83%	93%	71%	81%

Tabel 4. 41 Pengaruh Parameter *Max_depth* (SMOTE k=3)

Max_depth	Average akurasi	Average presisi	Average recall	Average F1-Score
3	79%	89%	67%	76%
6	86%	95%	77%	85%

Tabel 4. 42 Pengaruh Parameter *Max_depth* (SMOTE k=5)

Max_depth	Average akurasi	Average presisi	Average recall	Average F1-Score
3	79%	89%	68%	77%
6	86%	95%	75%	84%

Berdasarkan Tabel 4. 39 dengan k=3, peningkatan nilai *n_estimator* dari 10, 50, hingga 100 menunjukkan sedikit perubahan performa. Rata-rata akurasi meningkat hanya 1%, sedangkan rata-rata F1-Score tetap di angka 81%. Hal ini menunjukkan bahwa parameter *n_estimator* terhadap performa model secara keseluruhan tidak terlalu berpengaruh signifikan.

Berdasarkan Tabel 4.40 dengan k=5, peningkatan nilai *n_estimator* dari 10, 50, hingga 100 memberikan sedikit perubahan pada rata-rata akurasi sebesar 1% dan rata-rata *F1-Score* tetap di angka 81%. Hal ini menunjukkan bahwa parameter *n_estimator* memiliki pengaruh kecil terhadap performa model dengan k=5.

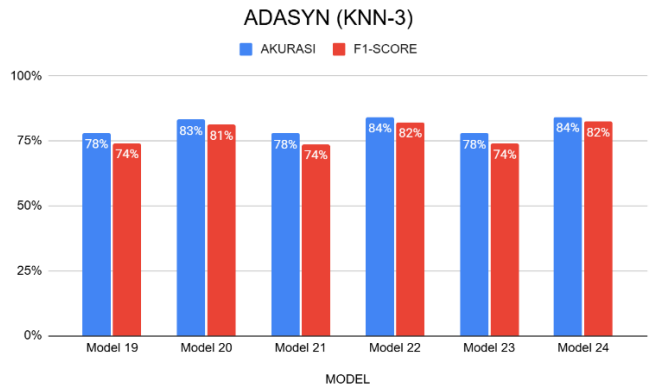
Berdasarkan Tabel 4. 41 peningkatan nilai *Max_depth* dari 3 ke 6 pada model k=3 menunjukkan peningkatan pada performa model yang ditunjukkan pada rata-rata akurasi meningkat sebesar 7% dan rata rata *F1-Score* meningkat sebesar

9%. Hal ini menunjukkan bahwa parameter *Max_depth* sangat memengaruhi performa model.

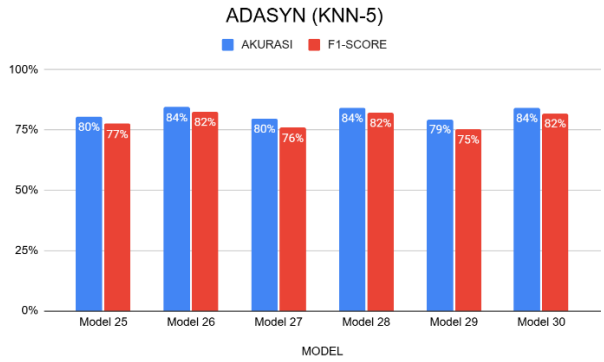
Berdasarkan Tabel 4. 42 peningkatan nilai *Max_depth* dari 3 ke 6 pada model dengan k=5 juga menunjukkan peningkatan, meskipun hasilnya sedikit lebih rendah dibandingkan k=3. Rata-rata akurasi meningkat sebesar 7%, dan rata-rata *F1-Score* naik sebesar 7%. Peningkatan, F1-Score pada k=5 sedikit lebih rendah dibandingkan k=3, sehingga menunjukkan bahwa titik optimal berada pada k=3 yang berarti ebih efektif dalam memanfaatkan peningkatan nilai *Max_depth*.

4.7.3 Analisa Hasil Balancing Data Metode ADASYN

Penggunaan ADASYN dengan variasi nilai K pada KNN (K=3 dan K=5) serta pengaturan parameter *random forest* yakni *n_estimators* dan *max_depth* menunjukkan hasil yang beragam. Peningkatan *n_estimator* dari 10 hingga ke 100 tidak memberikan perubahan signifikan. Peningkatan *max_depth* dari 3 ke 6 secara konsisten meningkatkan akurasi dan *F1-Score*.



Gambar 4. 5 Grafik Hasil Uji Coba ADASYN (KNN 3)



Gambar 4. 6 Grafik Hasil Uji Coba ADASYN (KNN 5)

Tabel 4. 43 Pengaruh Parameter *n_estimator* (ADASYN k=3)

n_estimator	Average akurasi	Average presisi	Average recall	Average F1-Score
10	81%	91%	68%	78%
50	81%	92%	68%	78%
100	81%	92%	68%	78%

Tabel 4. 44 Pengaruh Parameter *n_estimator* (ADASYN k=5)

n_estimator	Average akurasi	Average presisi	Average recall	Average F1-Score
10	82%	93%	70%	80%
50	82%	94%	68%	79%
100	82%	94%	67%	79%

Tabel 4. 45 Pengaruh Parameter *Max_depth* (ADASYN k=3)

Max_depth	Average akurasi	Average presisi	Average recall	Average F1-Score
3	78%	90%	63%	74%
6	84%	92%	74%	82%

Tabel 4. 46 Pengaruh Parameter *Max_depth* (ADASYN k=5)

Max_depth	Average akurasi	Average presisi	Average recall	Average F1-Score
3	80%	92%	65%	76%
6	84%	95%	72%	82%

Berdasarkan Tabel 4. 43 dengan k=3, menunjukkan bahwa peningkatan nilai *n_estimator* dari 10, 50, hingga 100 tidak menunjukkan perubahan performa. Rata-rata akurasi tetap sebesar 81% dan rata-rata *F1-score* tetap sebesar 78%. Hal ini menunjukkan bahwa parameter *n_estimator* tidak berpengaruh terhadap performa model secara keseluruhan.

Berdasarkan Tabel 4. 44 dengan k=5, menunjukkan bahwa peningkatan nilai parameter *n_estimator* dari 10, 50, hingga 100 tidak memberikan perubahan pada rata-rata akurasi, yang tetap sebesar 82% dan nilai *F1-Score* mengalami penurunan dari 80% pada *n_estimator* = 10 menjadi 79% pada *n_estimator* = 50 dan 100. Secara keseluruhan, nilai *n_estimator* = 10 menghasilkan *F1-Score* terbaik pada metode ADASYN dengan k=5.

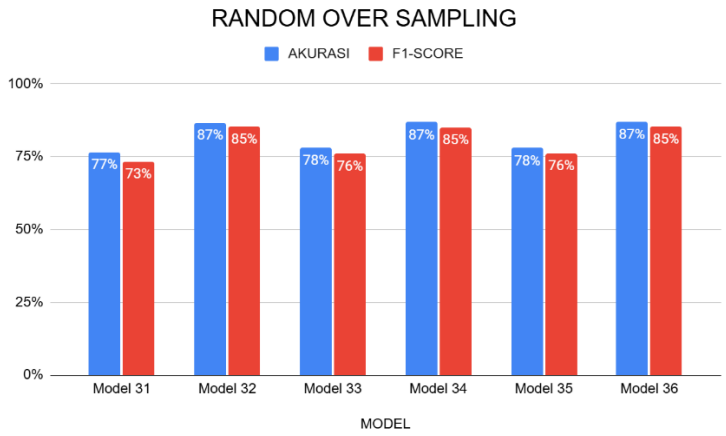
Berdasarkan Tabel 4. 45 peningkatan nilai *Max_depth* dari 3 ke 6 menunjukkan peningkatan terhadap performa model yakni nilai rata-rata akurasi meningkat sebesar 6% dan rata-rata *F1-Score* juga meningkat sebesar 8%. Hal ini menunjukkan kemampuan model menjadi jauh lebih baik dengan *max_depth* yang lebih besar. Secara keseluruhan, parameter *max_depth* = 6 memberikan performa terbaik

Berdasarkan Tabel 4. 46 peningkatan nilai *Max_depth* dari 3 ke 6 pada model dengan k=5 menujukkan peningkatan performa model. Nilai rata-rata akurasi meningkat sebesar 4% dan rata-rata *F1-Score* juga meningkat sebesar 6%. Hal ini menunjukkan bahwa peningkatan *max_depth* dapat meningkatkan performa model.

4.7.4 Analisa Hasil Balancing Data Metode ROS

Penggunaan ROS dengan pengaturan parameter *random forest* yakni *n_estimator* dan *max_depth* menunjukkan bahwa peningkatan *n_estimator* dari 10 hingga 100 tidak memengaruhi rata-rata akurasi yang tetap sebesar 82%, namun F1-score meningkat dari 79% pada *n*_estimator 10 menjadi 81% pada *n*_estimator 50 dan 100. Selain itu, peningkatan *max_depth* dari 3 ke 6 menghasilkan

peningkatan rata-rata akurasi dan F1-score masing-masing sebesar 10%, menunjukkan bahwa parameter ini secara konsisten meningkatkan performa model.



Gambar 4. 7 Grafik Hasil Uji Coba Balancing *Random Over Sampling*

Tabel 4. 47 Pengaruh Parameter *n_estimator* (ROS)

n_estimator	Average akurasi	Average presisi	Average recall	Average F1-Score
10	82%	91%	70%	79%
50	82%	92%	72%	81%
100	82%	90%	73%	81%

Tabel 4. 48 Pengaruh Parameter *Max_depth* (ROS)

Max_depth	Average akurasi	Average presisi	Average recall	Average F1-Score
3	77%	85%	67%	75%
6	87%	97%	76%	85%

Berdasarkan Tabel 4. 47, menunjukkan bahwa peningkatan nilai *n_estimator* dari 10, 50, hingga 10 tidak memberikan perubahan pada rata-rata akurasi, yang tetap sebesar 82%. Namun, rata-rata *F1-Score* mengalami peningkatan dari 79% pada *n_estimator* = 10 menjadi 81% pada *n_estimator* = 50 dan 100. Secara keseluruhan, nilai *n_estimator* 50 dan 100 menghasilkan *F1-Score* terbaik.

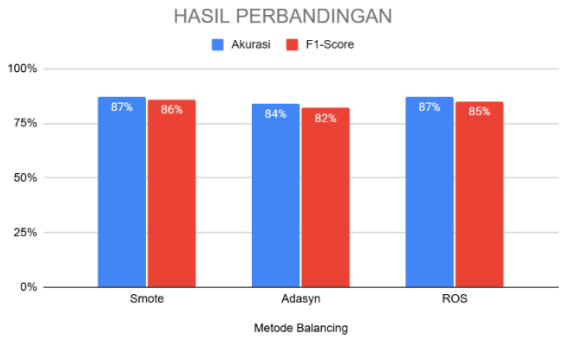
Berdasarkan Tabel 4. 48 parameter *max_depth* dari 3 ke 6 menunjukkan peningkatan terhadap performa model. Rata-rata akurasi meningkat sebesar 10% dan rata-rata *F1-score* juga meningkat sebesar 10%. Hal ini menunjukkan bahwa peningkatan *max_depth* bisa meningkatkan performa model.

4.7.5 Analisa Perbandingan Hasil Metode *Balancing*

Berdasarkan pengujian yang telah dilakukan terhadap metode *balancing data* yakni SMOTE, ADASYN, dan ROS, didapatkan hasil akurasi terbaik dari masing-masing metode seperti pada Tabel 4. 49 dan Gambar 4. 8 yang menunjukkan bahwa metode SMOTE menunjukkan performa terbaik dan menjadi metode paling seimbang dalam mendeteksi kelas minoritas (stroke) dengan akurasi 87% dan *F1-score* 86%. Metode ROS juga menghasilkan akurasi yang sama, yaitu 87%, namun *F1-Score* masih lebih rendah 1% dibandingkan SMOTE, sehingga menunjukkan bahwa SMOTE lebih unggul. Sedangkan ADASYN memiliki performa paling rendah dengan akurasi 84% dan *F1-score* 82%, menunjukkan bahwa metode ini kurang optimal dalam mendeteksi kelas minoritas dibandingkan dua metode lainnya. Secara keseluruhan, SMOTE adalah metode *balancing* terbaik dalam penelitian ini, karena mampu memberikan keseimbangan performa yang baik, sehingga lebih efektif untuk menangani ketidakseimbangan data.

Tabel 4. 49 Perbandingan Metode *Balancing*

Metode Balancing	Akurasi	Presisi	Recall	F1-Score
SMOTE	87%	95%	78%	86%
ADASYN	84%	95%	72%	82%
ROS	87%	97%	76%	85%



Gambar 4. 8 Grafik Perbandingan Metode *Balancing* Data

Presisi pada keseluruhan metode balancing data menunjukkan persentase yang tinggi, dengan nilai sebesar 95% hingga 97%. Hal ini menunjukkan bahwa model mampu memprediksi label sehat (positif) dengan akurasi yang baik terhadap data yang diprediksi sebagai sehat memang benar-benar sehat (*true positive*) dan hanya sedikit kesalahan dalam memprediksi data stroke (negatif) sebagai sehat (*false positive*).

Sedangkan, nilai *recall* menunjukkan persentase yang lebih rendah dibandingkan presisi, dengan kisaran 72% hingga 78%. *Recall* yang lebih rendah dibandingkan presisi menunjukkan bahwa meskipun model sangat yakin dengan prediksi data yang benar-benar sehat, model masih kehilangan sejumlah data yang sebenarnya sehat dan salah memprediksinya sebagai stroke (*false negative*). Dengan demikian, model ini cenderung lebih baik dalam memprediksi data yang benar-benar sehat (label positif) dengan akurasi tinggi, namun memiliki kelemahan dalam mendeteksi seluruh data sehat yang ada, sehingga masih terdapat beberapa kesalahan dalam memprediksi data sehat sebagai stroke.

4.8 Integrasi Penelitian

Dalam Al-Qur'an, Allah SWT menegaskan bahwa setiap ciptaan-Nya telah ditetapkan dengan ukuran dan keseimbangan yang sempurna. Konsep ini relevan dengan penelitian yang dilakukan yakni bertujuan mencapai keseimbangan dan ketelitian dalam pemilihan metode agar hasilnya akurat dan bermanfaat. Salah satu ayat yang sesuai dengan konsep tersebut adalah QS Al-Furqan ayat 2, yang berbunyi:

الَّذِي لَهُ مُلْكُ السَّمَاوَاتِ وَالْأَرْضِ وَلَمْ يَتَّخِذْ وَلَدًا وَلَمْ يَكُنْ لَهُ شَرِيكٌ فِي الْمُلْكِ وَخَلَقَ كُلَّ شَيْءٍ فَقَدَّرَهُ تَقْدِيرًا

"(Yaitu Zat) yang milik-Nyalah kerajaan langit dan bumi, (Dia) tidak mempunyai anak, dan tidak ada satu sekutu pun dalam kekuasaan(-Nya). Dia telah menciptakan segala sesuatu, lalu menetapkan ukuran-ukurannya dengan tepat." *(QS. Al-Furqan/ 25:2)*

Berdasarkan Tafsir Ibnu Katsir, ayat tersebut memiliki arti bahwa segala sesuatu selain Dia adalah makhluk (yang diciptakan) dan marbub (yang berada di bawah kekuasaan-Nya). Dia-lah pencipta segala sesuatu, Rabb, Raja dan Ilahnya. Sedangkan segala sesuatu berada di bawah kekuasaan, aturan, tatanan dan takdir-Nya(Abdullah, 2004). Tafsir tersebut menjelaskan bahwa Allah SWT menciptakan segala sesuatu dengan sangat cermat dan menetapkan ukuran yang tepat untuk masing-masing makhluk, sehingga tidak ada kekurangan atau ketidakseimbangan dalam alam semesta. Semua yang ada selain Allah SWT berada di bawah aturan dan takdir-Nya. Ini mengandung makna bahwa setiap hal memiliki ukuran yang sesuai, seperti halnya dalam penelitian ini, yang mana metode *balancing* dan parameter yang dipilih dengan tepat dapat mengurangi kesalahan dalam proses klasifikasi.

Selain itu, pemilihan algoritma klasifikasi yang tepat mencerminkan pentingnya kesesuaian antara metode yang digunakan dengan data yang dianalisis. Sebagaimana firman Allah SWT dalam Q.S. Al-Anbiya ayat 33:

وَهُوَ الَّذِي خَلَقَ اللَّيْلَ وَالنَّهَارَ وَالشَّمْسَ وَالْقَمَرَ ۖ كُلٌّ فِي فَلَكٍ يَسْبَحُونَ

"*Dan Dialah yang telah menciptakan malam dan siang, matahari dan bulan; semua itu beredar di dalam falaknya masing-masing" (QS. Al Anbiya: 33).*

Menurut Tafsir As-Sa'di, ayat ini menjelaskan bahwa Allah telah memelihara matahari dan bintang, masing masing beredar dalam garis edar yang ditentukan(Ash-Shiddieqy, 2011). Dalam konteks penelitian ini, pemilihan algoritma dan pengaturan parameter yang optimal dapat diibaratkan seperti ketelitian Allah dalam menciptakan setiap makhluk sesuai ukurannya, sehingga menghasilkan keseimbangan dan keharmonisan. Dengan demikian, pemilihan metode klasifikasi yang paling sesuai yakni *random forest*, diharapkan dapat memberikan performa yang baik dalam klasifikasi stroke, sehingga meminimalkan kesalahan prediksi.

Selain itu, Allah SWT juga menegaskan pentingnya memanfaatkan anugerah yang telah diberikan untuk kebaikan dunia dan akhirat. Salah satu ayat yang relevan dengan penelitian ini adalah QS Al-Qashash ayat 77, yang berbunyi:

وَابْتَغِ فِيمَا آتَاكَ اللَّهُ الدَّارَ الْآخِرَةَ وَلَا تَنسَ نَصِيبَكَ مِنَ الدُّنْيَا وَأَحْسِن كَمَا أَحْسَنَ اللَّهُ إِلَيْكَ وَلَا تَبْغِ الْفَسَادَ فِي الْأَرْضِ إِنَّ اللَّهَ لَا يُحِبُّ الْمُفْسِدِينَ

"*Dan carilah pada apa yang telah dianugerahkan Allah kepadamu (kebahagiaan) negeri akhirat, dan janganlah kamu melupakan bahagianmu dari (kenikmatan) duniawi dan berbuat baiklah (kepada orang lain) sebagaimana Allash telah berbuat baik kepadamu, dan janganlah kamu berbuat kerusakan di (muka) bumi.*

Sesungguhnya Allah tidak menyukai orang-orang yang berbuat kerusakan." *(QS. Al-Qashash/28: 77).*

Menurut Tafsir Ibn Katsir dalam *Tafsir al-Qur'an al-'Adzim*, ayat tersebut menekankan pentingnya menggunakan harta dan nikmat yang diberikan Allah sebagai sarana untuk meningkatkan ketaatan dan mendekatkan diri kepada-Nya melalui berbagai perbuatan baik. Allah memperbolehkan makan, minum, mengenakan pakaian, memiliki tempat tinggal, dan menikah, karena setiap manusia memiliki tanggung jawab kepada Tuhan, diri sendiri, dan keluarga. Oleh karena itu, kewajiban-kewajiban tersebut harus dipenuhi dengan baik, sembari berbuat kebaikan kepada sesama makhluk, sebagaimana Allah telah berbuat baik kepada manusia. Larangan untuk berbuat kerusakan di muka bumi dan tindakan jahat kepada makhluk-Nya(Salam, 2021). Dalam konteks penelitian ini, nilai-nilai yang terkandung dalam ayat tersebut dapat menjadi inspirasi dalam menerapkan prinsip kehati-hatian dan tanggung jawab, termasuk dalam proses pemilihan metode klasifikasi yang tepat untuk mencapai hasil yang akurat dan bermanfaat.

Kemudian, dalam QS. Al-Mujadilah ayat 11, Allah SWT akan meninggikan derajat orang-orang beriman dan berilmu yang menggunakannya untuk kebaikan orang lain, sehingga ilmu yang dikembangkan membawa manfaat bagi lingkungan sekitarnya. Berikut merupakan QS Al-Mujadilah ayat 11, yang berbunyi:

يَا أَيُّهَا الَّذِينَ آمَنُوا إِذَا قِيلَ لَكُمْ تَفَسَّحُوا فِي الْمَجَالِسِ فَافْسَحُوا يَفْسَحِ اللَّهُ لَكُمْ ۖ وَإِذَا قِيلَ انْشُزُوا فَانْشُزُوا يَرْفَعِ اللَّهُ الَّذِينَ آمَنُوا مِنْكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ ۚ وَاللَّهُ بِمَا تَعْمَلُونَ خَبِيرٌ

"*Wahai orang-orang yang beriman, apabila dikatakan kepadamu "Berilah kelapangan di dalam majelis-majelis," lapangkanlah, niscaya Allah akan memberi kelapangan untukmu. Apabila dikatakan, "Berdirilah," (kamu) berdirilah. Allah*

niscaya akan mengangkat orang-orang yang beriman di antaramu dan orang-orang yang diberi ilmu beberapa derajat. Allah Mahateliti terhadap apa yang kamu kerjakan." *(QS. Al-Mujadilah/58: 11).*

Menurut tafsir Ibn Katsir, ayat tersebut mengajarkan orang-orang beriman untuk melapangkan tempat di majelis sebagai bentuk kebaikan yang akan dibalas Allah dengan kelapangan hidup. Rasulullah SAW juga menekankan pentingnya saling memberi ruang tanpa menyuruh orang lain berdiri. Allah menjanjikan derajat yang tinggi bagi orang-orang beriman dan berilmu yang taat pada perintah-Nya, serta membalas amal baik mereka di dunia dan akhirat(Suryati dkk., 2019).

Ayat tersebut menjelaskan bahwa Allah menjanjikan akan mengangkat derajat orang-orang yang berilmu, karena ilmu mereka dapat menjadi petunjuk dan manfaat bagi umat, melebihi derajat orang-orang yang tidak berilmu. Dan Allah Mahateliti terhadap niat, cara, dan tujuan dari apa yang kamu kerjakan, baik dalam urusan dunia maupun akhirat.(Sari & Retnaningsih, 2022).

Selanjutnya, QS. An-Nahl ayat 43 mengarahkan untuk terus belajar dan mengembangkan pengetahuan yang didasarkan pada penelitian-penelitian terdahulu. Salah satu ayat yang sesuai dengan pernyataan tersebut adalah QS An-Nahl ayat 43, yang berbunyi:

وَمَا أَرْسَلْنَا مِنْ قَبْلِكَ إِلَّا رِجَالًا نُوحِي إِلَيْهِمْ ۚ فَاسْأَلُوا أَهْلَ الذِّكْرِ إِنْ كُنْتُمْ لَا تَعْلَمُونَ

"*Kami tidak mengutus sebelum engkau (Nabi Muhammad), melainkan laki-laki yang Kami beri wahyu kepadanya. Maka, bertanyalah kepada orang-orang yang mempunyai pengetahuan jika kamu tidak mengetahui*." *(QS. An-Nahl/16: 43).*

Menurut Tafsir Ibn Katsir, ayat ini menegaskan bahwa Allah hanya mengutus manusia sebagai rasul, bukan malaikat atau makhluk lainnya, agar umat

manusia dapat lebih mudah menerima dan mengikuti ajaran yang dibawa para rasul tersebut. Selain itu, ayat ini juga mengarahkan umat untuk bertanya kepada *ahludz-dzikr* (orang yang memiliki ilmu) dalam hal-hal yang tidak mereka ketahui, agar mendapatkan pemahaman yang benar berdasarkan wahyu dan ilmu yang telah diajarkan(Abdullah, 2004).

Ayat tersebut terdapat tazkiyah (rekomendasi) terhadap ahli ilmu, karena Allah memerintahkan orang yang tidak tahu untuk bertanya kepada mereka, dan bahwa tugas orang awam adalah bertanya keada ahli ilmu. Sehingga, dengan dilandasi dari ayat tersebut peneliti merujuk pada pada penelitian-penelitian yang terdahulu untuk dikembangkan sebagai penelitian yang dilakukan saat ini.

Sebagai seorang Muslim, ayat-ayat tersebut menginspirasi kita untuk mencari manfaat tidak hanya untuk diri sendiri, tetapi juga bagi orang lain dan lingkungan sekitar. QS. Al-Furqan ayat 2 menggambarkan bagaimana Allah menciptakan segala sesuatu dengan ukuran yang sempurna dan tepat. Ayat ini menunjukkan bahwa tidak ada ketidakseimbangan dalam ciptaan-Nya, yang memberi teladan tentang pentingnya keharmonisan dan ketelitian dalam segala usaha manusia, termasuk penelitian ini. Prinsip ini mengarahkan peneliti untuk memilih metode dan parameter yang tepat dalam penelitian agar menghasilkan hasil yang seimbang dan akurat. Selain itu, ayat QS. Al-Anbiya ayat 33 memberikan pengingat tentang pentingnya keselarasan dalam setiap langkah, baik dalam kehidupan maupun penelitian. Keselarasan ini dicapai melalui pemilihan metode yang tepat, seperti Random Forest, yang diharapkan mampu memberikan performa terbaik dengan meminimalkan kesalahan prediksi. Selain itu, QS. Al-Mujadilah

ayat 11 menjanjikan peningkatan derajat bagi mereka yang berilmu dan membawa manfaat bagi sesama, sebagai pengingat bahwa ilmu yang dikembangkan sebaiknya digunakan demi kebaikan bersama. QS. An-Nahl ayat 43 juga mengarahkan untuk terus belajar dari penelitian terdahulu dan mengembangkan pengetahuan, sehingga ilmu yang dihasilkan dapat menjadi lebih baik. Berdasarkan landasan ini, penelitian yang berfokus pada perbandingan metode *balancing* data dalam klasifikasi stroke diharapkan dapat memberikan keseimbangan dalam distribusi data dan memberikan performa terbaik dalam klasifikasi.

Dengan demikian, penelitian ini menjadi wujud nyata dari perintah Allah untuk berbuat kebaikan yang tidak hanya bermanfaat secara ilmiah tetapi juga memenuhi aspek spiritual. Prinsip keseimbangan dalam penciptaan menjadi inspirasi bagi penelitian ini untuk memilih metode *balancing* yang tepat dalam klasifikasi data, sehingga didapatkan hasil terbaik.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan pengujian mengenai penanganan ketidakseimbangan data pada *dataset* penyakit stroke terhadap performa algoritma *random forest* didapatkan hasil bahwa sebelum di lakukan *balancing* akurasinya sebesar sebesar 96%. Namun, karena data tidak seimbang, performa *F1-score* untuk label negatif (stroke) sangat rendah, yakni 0%.

Pengujian menggunakan metode *balancing* diperoleh performa terbaik dari metode SMOTE dengan akurasi 87% dan *F1-score* 86%. Peningkatan nilai parameter *max_depth* dari 3 ke 6 menunjukkan peningkatan performa model, dengan rata-rata akurasi sebesar 7% dan rata-rata *F1-score* sebesar 7% pada SMOTE. Selain itu, peningkatan nilai *n_estimators* dari 10, 50, hingga 100 tidak menunjukkan pengaruh terhadap hasil akurasi maupun *F1-Score*.

Metode ROS menghasilkan performa yang setara dengan SMOTE dalam hal akurasi sebesar 87%, namun memiliki *F1-score* 1% lebih rendah dibandingkan SMOTE yakni sebesar 85%. Peningkatan nilai parameter *max_depth* dari 3 ke 6 menunjukkan peningkatan performa model, dengan rata-rata akurasi sebesar 10% dan rata-rata *F1-score* sebesar 10%. Selain itu, peningkatan nilai n_estimators dari 10, 50, hingga 100 tidak menunjukkan pengaruh terhadap hasil akurasi, sedangkan *F1-Score* hanya meningkat sebesar 2%.

Pengujian menggunakan metode ADASYN menunjukkan performa terendah dengan akurasi sebesar 84% dan *F1-score* sebesar 82%, yang menunjukkan keterbatasannya dalam mendeteksi kelas minoritas. Peningkatan nilai parameter *max_depth* dari 3 ke 6 menunjukkan peningkatan performa model, dengan rata-rata akurasi meningkat sebesar 5% dan rata-rata *F1-score* meningkat sebesar 7%. Selain itu, peningkatan nilai *n_estimators* dari 10, 50, hingga 100 tidak menunjukkan pengaruh terhadap hasil akurasi maupun *F1-Score*.

Secara keseluruhan, SMOTE memberikan hasil yang lebih baik dibandingkan metode ADASYN dan ROS, terutama dalam meningkatkan kemampuan deteksi kasus stroke pada data tidak seimbang. Dengan demikian, penelitian ini menunjukkan bahwa metode SMOTE adalah metode yang paling baik dalam menangani ketidakseimbangan data pada klasifikasi stroke, terutama ketika dipadukan dengan parameter max_depth yang lebih besar pada algoritma *random forest*.

5.2 Saran

Berdasarkan proses dan hasil penelitian ini, peneliti menyadari bahwa penelitian ini masih banyak kekurangan. Maka dari itu, diharapkan pada penelitian selanjutnya agar dapat melakukan peningkatan dan perbaikan agar hasil yang didapatkan menjadi lebih baik. Beberapa saran untuk penelitian selanjutnya yaitu:

1. Penggunaan *dataset* yang lain dari sumber yang berbeda agar dapat menilai kemampuan model dalam mengklasifikasikan pada populasi yang berbeda.
2. Penggunaan *balancing* yang berbeda seperti *Undersampling*.

3. Eksperimen dengan algoritma klasifikasi yang lain seperti SVM atau *Neural Network.*

DAFTAR PUSTAKA

Abdullah. (2004). *Tafsir Ibnu Katsir Jilid 6* (M. Y. Harun, F. Okbah, T. S. al-Katsiri, A. I. al-atsari, & F. G. Anuz, Eds.; 6th ed.). Pustaka Imam asy-Syafi'i.

Aditya, A., Nurina Sari, B., Nur Padilah, T., sitasi, C., Aditya, A., Sari, B. N., Padilah, T., & pengukuran jarak Euclidean dan Gower, P. (2021). 2021, 1-7 pada klaster k-medoids. *Jurnal Teknologi Dan Sistem Komputer*, *9*(1), 1–7. https://doi.org/10.14710/jtsiskom.2021.13747

Adiyasa, R. P. (2024). Pelatihan Pencegahan dan Penanganan Pasien Sroke dalam Rangka Membangun Masyarakat Sehat dan Produktif. *JAMAS Jurnal Abdi Masyarakat*.

Adnin Kamila, S., Sulistijowati, R. R. S., & Susanto, I. (2023). Klasifikasi Penyakit Jantung Menggunakan Decision Tree dan Random Forest. *Rosiding Seminar Nasional Teknologi Dan Sains*, 7–12.

Akbar, K., & Hayaty, M. (2020). Data Balancing untuk Mengatasi Imbalance Dataset pada Prediksi Produksi Padi Balancing Data to Overcome Imbalance Dataset on Rice Production Prediction. *Jurnal Ilmiah Intech : Information Technology Journal of UMUS*, *2*(02), 1–14.

Amalia, E., Maidaliza, Fradisa, L., Sesrianty, V., Arif, M., & Kartika, K. (2024). Edukasi Pencegahan Dan Penatalaksanaan Stroke Pada. *Journal of Human And Education*, *4*, 408–414.

Amien, J. Al, Yoze Rizki, & Mukhlis Ali Rahman Nasution. (2022). Implementasi Adasyn Untuk Imbalance Data Pada Dataset UNSW-NB15 Adasyn Implementation For Data Imbalance on UNSW-NB15 Dataset. *Jurnal CoSciTech (Computer Science and Information Technology)*, *3*(3), 242–248. https://doi.org/10.37859/coscitech.v3i3.4339

Ananda, I. K., Fanani, A. Z., Setiawan, D., & Wicaksono, D. F. (2024). Penerapan Random Oversampling dan Algoritma Boosting untuk Memprediksi Kualitas Buah Jeruk. *Edumatic: Jurnal Pendidikan Informatika*, *8*(1), 282–289. https://doi.org/10.29408/edumatic.v8i1.25836

Ariandi, M., & Rahma Puteri, S. (2022). Analisis Visualisasi Data Kecamatan Kertapati menggunakan Tableau Public. In *Jurnal Jupiter* (Vol. 14, Issue 2). Bulan Oktober.

Aryanti, R., Misriati, T., & Hidayat, R. (2023). Klasifikasi Risiko Kesehatan Ibu Hamil Menggunakan Random Oversampling Untuk Mengatasi Ketidakseimbangan Data. *Media Online*, *3*(5), 409–416. https://djournals.com/klik

Ash-Shiddieqy, T. M. H. (2011). *Tafsir Al-Qur''anul Majdid An-Nur Jilid 3*.

Awliya Muhammad Ashfania, G., Prahasto, T., Widodo, A., Warsokusumo, T., Soedarto, J., Tembalang, K., Semarang, K., Tengah, J., Gatot Subroto NoKav, J., Tim, K., Setiabudi, K., Jakarta Selatan, K., & Khusus Ibukota Jakarta, D. (2022). *Penggunaan Algoritma Random Forest untuk Klasifikasi berbasis Kinerja Efisiensi Energi pada Sistem Pembangkit Daya* (Vol. 24, Issue 3).

Ayuningtyas, I., & Uswatun, E. (2021). *Penerapan Synthetic Minority Oversampling Technique (Smote) pada Kasus Dampak Covid-19 Terhadap Penduduk Usia Kerja Di Kalimantan Timur (Synthetic Minority Oversampling Technique Approach In Case Of The Impact Of Covid-19 On Population Working Age In East Kalimantan)*.

Br. Sebayang, E. R., Herry Chrisnanto, Y., & Melina. (2023). Klasifikasi Data Kesehatan Mental di Industri Teknologi Menggunakan Algoritma Random Forest. *IJESPG Journal*, 237–253.

Chamidah, N., Mega Santoni, M., & Matondang, N. (2021). Pengaruh Oversampling pada Klasifikasi Hipertensi dengan Algoritma Naïve Bayes, Decision Tree, dan Artificial Neural Network (ANN). *JURNAL RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *4*(3), 635–641.

Dai, Q., Liu, J. wei, & Zhao, J. L. (2022). Distance-based arranging oversampling technique for imbalanced data. *Neural Computing and Applications*, *35*(2), 1323–1342. https://doi.org/10.1007/s00521-022-07828-8

Dinova, D. B., & Prasetiyo, B. (2024). Implementasi Random Forest dalam Klasifikasi Kanker Paru-Paru. *JOINTER-JOURNAL OF INFORMATICS ENGINEERING*, *05*(01).

Direktorat Promosi Kesehatan dan Pemberdayaan Masyarakat. (2024, August 6). *Tingkatan Kualitas dan Layanan Stroke Lewat Transformasi Kesehatan*. Https://Ayosehat.Kemkes.Go.Id/Kenali-Stroke-Dan-Penyebabnya#:~:Text=Berdasarkan%20hasil%20Riskesdas%20prevalensi%20stroke,1000%20penduduk%20pada%20tahun%202018.

Elreedy, D., & Atiya, A. F. (2019). A Comprehensive Analysis of Synthetic Minority Oversampling Technique (SMOTE) for handling class imbalance. *Information Sciences*, *505*, 32–64. https://doi.org/10.1016/j.ins.2019.07.070

Fluorida Fibrianda, M., & Bhawiyuga, A. (2018). *Analisis Perbandingan Akurasi Deteksi Serangan Pada Jaringan Komputer Dengan Metode Naïve Bayes Dan Support Vector Machine (SVM)* (Vol. 2, Issue 9). http://j-ptiik.ub.ac.id

Gde Agung Brahmana Suryanegara, Adiwijaya, & Mahendra Dwifebri Purbolaksono. (2021). Peningkatan Hasil Klasifikasi pada Algoritma Random

Forest untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *5*(1), 114–122. https://doi.org/10.29207/resti.v5i1.2880

Geron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : concepts, tools, and techniques to build intelligent systems (2nd Edition)*.

Gusrialni Fitri, N., Adilya, S., Apriliani, & Azizi, F. (2023). *Perbandingan Klasifikasi Naive Bayes dan C4.5 untuk Diagnosa Penyakit Stroke*. https://journal.irpi.or.id/index.php/sentimas

Handayani, I. Y., Aini Isnawati, I., Hamim, H. N., & Hasan Probolinggo, Z. (2023). *Faktor-Faktor Yang Mempengaruhi Tingkat Keparahan Stroke Di Ruang Melati Rsud Dr. Haryoto Lumajang*. https://journal-mandiracendikia.com/jikmc

Harmayetty, Ni'mah, L., & Shafly Nur Firdaus, A. (2020). *Hubungan Dukungan Keluarga Dan Kepatuhan Rehabilitasi Dengan Kemandirian Pasien Pasca Stroke* (Vol. 26).

Hasnain, M., Pasha, M. F., Ghani, I., Imran, M., Alzahrani, M. Y., & Budiarto, R. (2020). Evaluating Trust Prediction and Confusion Matrix Measures for Web Services Ranking. *IEEE Access*, *8*, 90847–90861. https://doi.org/10.1109/ACCESS.2020.2994222

Hidayat, W., Ardiansyah, M., & Setyanto, A. (2021). Pengaruh Algoritma ADASYN dan SMOTE terhadap Performa Support Vector Machine pada Ketidakseimbangan Dataset Airbnb. *Edumatic: Jurnal Pendidikan Informatika*, *5*(1), 11–20. https://doi.org/10.29408/edumatic.v5i1.3125

Imama Sabilla, W., & Bella Vista, C. (2021). Implementasi SMOTE dan Under Sampling pada Imbalanced Dataset untuk Prediksi Kebangkrutan Perusahaan. In *Jurnal Komputer Terapan* (Vol. 7, Issue 2). https://jurnal.pcr.ac.id/index.php/jkt/

Indra, D., Hayati, L. N., Daris, M. A., As'ad, I., & Mansyur, U. (2024). Penerapan Metode Random Forest dalam Klasifikasi Huruf BISINDO dengan Menggunakan Ekstraksi Fitur Warna dan Bentuk The Application of Random Forest Method in BISINDO Letter Classification Using Color and Shape Feature Extraction. *Jurnal Sistem Komputer*, *13*(1), 2020. https://doi.org/10.34010/komputika.v13i1.10363

Islam, A., Belhaouari, S. B., Rehman, A. U., & Bensmail, H. (2022). KNNOR: An oversampling technique for imbalanced datasets[Formula presented]. *Applied Soft Computing*, *115*. https://doi.org/10.1016/j.asoc.2021.108288

Jackins, V., Vimal, S., Kaliappan, M., & Lee, M. Y. (2021). AI-based smart prediction of clinical disease using random forest classifier and Naive Bayes. *Journal of Supercomputing*, *77*(5), 5198–5219. https://doi.org/10.1007/s11227-020-03481-x

Kaope, C., & Pristyanto, Y. (2023). The Effect of Class Imbalance Handling on Datasets Toward Classification Algorithm Performance. *MATRIK : Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer*, *22*(2), 227–238. https://doi.org/10.30812/matrik.v22i2.2515

Magnolia, C., Nurhopipah, A., Bagus, D., & Kusuma, A. (2022). Edu Komputika Journal Penanganan Imbalanced Dataset untuk Klasifikasi Komentar Program Kampus Merdeka Pada Aplikasi Twitter. In *Edu Komputika* (Vol. 9, Issue 2). http://journal.unnes.ac.id/sju/index.php/edukom

Mahesh, B. (2020). Machine Learning Algorithms-A Review. *International Journal of Science and Research*. https://doi.org/10.21275/ART20203995

Mahmuda, S. (2024). Implementasi Metode Random Forest pada Kategori Konten Kanal Youtube. *Jurnal Jendela Matematika*, *2*.

Marlisa, H., Satyahadewi, N., Imro'ah, N., & Debataraja, N. N. (2024). Application Of Adasyn Oversampling Technique On K-Nearest Neighbor Algorithm. *BAREKENG: Jurnal Ilmu Matematika Dan Terapan*, *18*(3), 1829–1838. https://doi.org/10.30598/barekengvol18iss3pp1829-1838

Mushfique, M. (2019, January 14). *Irsyad Al-Hadith Siro Ke-333: Sebaik-baik Perkara Adalah Yang Pertengahan*.

Nugroho, A. (2022). Analisa Splitting Criteria Pada Decision Tree dan Random Forest untuk Klasifikasi Evaluasi Kendaraan. *JSITIK: Jurnal Sistem Informasi Dan Teknologi Informasi Komputer*, *1*(1), 41–49. https://doi.org/10.53624/jsitik.v1i1.154

Rahayu, S., Bharata Adji, T., Akhmad Setiawan, N., & Teknik Elektro dan Teknologi Informasi, D. (2017). Penghitungan k-NN pada Adaptive Synthetic-Nominal (ADASYN-N) dan Adaptive Synthetic-kNN (ADASYN-kNN) untuk Data Nominal-Multi Kategori. *Ktrl.Inst (J.Auto.Ctrl.Inst)*, *9*(2).

Salam, A. (2021, October 9). *Tafsir Surah Al-Qasas Ayat 77: Ingat Akhirat Harus, Tapi Dunia Jangan Dilupakan*.

Sari, D. F. P. A., & Retnaningsih, D. A. (2022). Keutamaan Orang Berilmu Dalam Al-Qur'an Surat Al-Mujadalah Ayat 11. *Tarbiya Islamica*, *10*, 118–129.

Sonjaya, C. B., Fitri, A., Masruriyah, N., Kusumaningrum, D. S., & Pratama, A. R. (2022). The Performance Comparison of Classification Algorithm in Order to

Detecting Heart Disease. *INTERNAL (Information System Journal)*, *5*(2), 166–175. https://doi.org/10.32627

Suci Amaliah, Nusrang, M., & Aswi, A. (2022). Penerapan Metode Random Forest Untuk Klasifikasi Varian Minuman Kopi di Kedai Kopi Konijiwa Bantaeng. *VARIANSI: Journal of Statistics and Its Application on Teaching and Research*, *4*(3), 121–127. https://doi.org/10.35580/variansiunm31

Sulistiyono, M., Pristyanto, Y., Adi, S., & Gumelar, G. (2021). Implementasi Algoritma Synthetic Minority Over-Sampling Technique untuk Menangani Ketidakseimbangan Kelas pada Dataset Klasifikasi. *SISTEMASI*, *10*(2), 445. https://doi.org/10.32520/stmsi.v10i2.1303

Suryati, A., Nurmila, N., & Rahman, C. (2019). *Konsep Ilmu Dalam Al-Qur'an: Studi Tafsir Surat Al-Mujadilah Ayat 11 dan Surat Shaad Ayat 29*. *04*. https://doi.org/10.30868/at.v4i02.476

Tahyudin, I., Solikhatin, S. A., Tikaningsih, A., Lestari, P., Nambo, H., Winarto, E., & Hassa, N. (2024). Forecasting Hospital Length of Stay for Stroke Patients: A Machine Learning Approach. *International Journal of Advances in Soft Computing and Its Applications*, *16*(1), 99–117. https://doi.org/10.15849/IJASCA.240330.06

Yudha Chrisanto, E., Ernita, C., Erlianti, F., & Listiyo Putri, E. (2022). Penyuluhan kesehatan tentang stroke. In *JOURNAL OF Public Health Concerns* (Vol. 2, Issue 3).

Yulian Pamuji, F., & Dwi Arma Putri, S. (2023). Komparasi Metode SMOTE dan ADASYN Untuk Penanganan Data Tidak Seimbang MultiClass. *JIP (Jurnal Informatika Polinema)*.