Universitas Nusantara PGRI Kediri
UPT. PERPUSTAKAAN, PUBLIKASI DAN INOVASI
Alamat: Kampus 1, Jl. KH. Ahmad Dahlan No.76 Kota Kediri 64112
Telp. (0354) 771576,(0354) 771503, (0354) 771495, Fax.(0354) 771576
Website: http://ppi.unpkediri.ac.id/ Email: perpustakaan@unpkediri.ac.id

SURAT KETERANGAN BEBAS SIMILARITY

Ketua UPT Perpustakaan, Publikasi dan Inovasi Universitas Nusantara PGRI Kediri menerangkan bahwa mahasiswa dengan identitas berikut:

Nama Mahasiswa	: Alief Cahyo Utomo
NPM	: 2113030101
Program Studi	: S1-Sistem Informasi

Judul Karya Ilmiah:
"ANALISIS HALVING RANDOM SEARCH CROSS VALIDATION PADA OPTIMASI MODEL MACHINE LEARNING"

Dinyatakan sudah memenuhi syarat batas maksimal 30% *similarity* sesuai dengan ketentuan yang berlaku pada setiap subbab naskah Laporan **Tugas Akhir/Skripsi/Tesis** yang disusun.
Demikian Surat Keterangan ini kami berikan untuk dapat dipergunakan seperlunya.

Kediri, 25 Desember 2025
Ka UPT PPI,

Dr. Abdul Aziz Hunaifi, M.A

Alief Cahyo Utomo

By Prodi SI

WORD COUNT	5690	TIME SUBMITTED	25-DEC-2025 09:57AM
		PAPER ID	119647777

BAB I

PENDAHULUAN

A. Latar Belakang

Di era digital saat ini, *machine learning* telah menjadi teknologi utama dalam berbagai bidang, seperti kesehatan, bisnis, manufaktur, dan keuangan, untuk menghasilkan prediksi dan wawasan yang bernilai dari data yang kompleks. Pengembangan model *machine learning* yang akurat dan dapat diandalkan semakin penting. Kemampuan model tersebut sangat tergantung pada pemilihan algoritma yang tepat serta penyesuaian *hyperparameter* secara optimal. *Hyperparameter* adalah parameter yang ditentukan sebelum proses pelatihan model dan memiliki dampak signifikan pada akurasi, dan komputasi yang dibutuhkan oleh model.

Namun, pemilihan *hyperparameter* yang optimal bukanlah tugas yang mudah, terutama ketika bekerja dengan dataset besar dan kompleks yang memiliki sejumlah besar variabel. Metode pencarian *hyperparameter* tradisional seperti *Grid Search Cross Validation* menjadi sangat mahal dari segi waktu dan sumber daya komputasi karena mencoba semua kombinasi dari parameter yang mungkin. Di sisi lain, metode *Random Search Cross Validation* lebih efisien dalam hal waktu, tetapi hasilnya terkadang kurang optimal karena proses pencarian *hyperparameter* dilakukan secara acak tanpa memperhatikan kebutuhan sumber daya dan waktu.

Sebagai solusi terhadap permasalahan ini, dikembangkan metode *Halving Random Search Cross Validation*. Metode ini adalah cara untuk mengoptimalkan parameter *hyperparameter* dengan tujuan meningkatkan efisiensi dalam proses pencarian. Metode ini menggunakan pendekatan bertahap, di mana pada setiap tahap iterasi, parameter *hyperparameter* yang kurang efektif secara perlahan dihilangkan. Dengan menggunakan *Halving Random Search Cross Validation*, proses pencarian *hyperparameter* dapat dipercepat karena model hanya berfokus pada kombinasi *hyperparameter* yang menjanjikan di setiap tahap seleksi, sehingga mengurangi waktu komputasi secara signifikan tanpa mengorbankan akurasi model.

Untuk menguji efektivitas metode tersebut, diperlukan dataset yang relevan dan representatif. Dalam penelitian ini, digunakan dataset *Internet Service Churn* yang tersedia secara publik melalui platform Kaggle. Dataset ini berisi informasi pelanggan dari sebuah

perusahaan telekomunikasi dan memuat berbagai fitur seperti jenis layanan, durasi berlangganan, hingga status langganan pelanggan. Dengan 72.274 data *entry*, dataset ini sangat sesuai untuk mengevaluasi kinerja metode optimasi hyperparameter, khususnya dalam konteks penggunaan data berskala besar. Melalui dataset ini, dapat dilakukan pembandingan antara metode *Halving Random Search Cross Validation* dengan metode optimasi lainnya dalam hal akurasi model serta efisiensi waktu komputasi.

B. Identifikasi Masalah

Identifikasi masalah yang terdapat dalam latar belakang tersebut adalah sebagai berikut:

1. Tingginya beban komputasi dan waktu yang diperlukan pada metode optimasi seperti *Grid Search Cross Validation.*
2. Kebutuhan akan metode yang lebih efisien dan efektif.

C. Rumusan Masalah

Berdasarkan permasalahan yang diuraikan latar belakang masalah, rumusan masalah dapat diambil adalah sebagai berikut :

1. Bagaimana perbandingan waktu yang diperlukan oleh *Halving Random Search Cross Validation* untuk menemukan *hyperparameter* optimal dibandingkan metode *Grid Search Cross Validation* dan *Random Search Cross Validation*?
2. Bagaimana efektivitas *Halving Random Search Cross Validation* dalam mengoptimalkan *hyperparameter* model machine learning dibandingkan dengan metode pencarian *hyperparameter Random Search Cross Validation* dan *Grid Search Cross Validation*?

D. Batasan Masalah

Berdasarkan latar belakang masalah diatas, terdapat beberapa batasan masalah agar penelitian lebih terfokus pada tujuan dan manfaat yang diharapkan, yaitu :

1. Penelitian ini hanya akan membandingkan *Halving Random Search Cross Validation* dengan *Grid Search Cross Validation* dan *Random Search Cross Validation* sebagai metode pencarian hyperparameter. Metode optimasi lainnya, seperti *Bayesian Optimization* atau *Particle Swarm Optimization*, tidak akan dibahas.
2. Penilaian terhadap kinerja model dilakukan dengan melihat tingkat akurasi, presisi, dan *recall* yang dicapai dan lamanya waktu yang dibutuhkan. Kriteria lain, seperti interpretabilitas model atau sensitivitas model terhadap perubahan data, tidak akan

menjadi fokus penelitian.

E. Tujuan Penelitian

Berikut merupakan tujuan yang diharapkan dari dilakukannya penelitian ini:

1. Mengetahui sejauh mana efisiensi waktu komputasi yang dapat dicapai dengan menggunakan metode *Halving Random Search Cross Validation*, jika dibandingkan dengan metode *Grid Search Cross Validation* dan *Random Search Cross Validation*, dalam proses optimasi *hyperparameter* pada model *machine learning*.

2. Menganalisa efektivitas metode *Halving Random Search Cross Validation* dalam mengoptimalkan hyperparameter model machine learning dibandingkan dengan metode optimasi *Grid Search Cross Validation* dan *Random Search Cross Validation*.

F. Manfaat Dan Kegunaan Penelitian

Manfaat dan tujuan dari melakukan penelitian tersebut adalah sebagai berikut:

1. Bagi Penulis

   Sebagai sarana untuk memperdalam pemahaman penulis mengenai metode optimasi *hyperparameter*, khususnya *Halving Random Search Cross Validation*, serta bagaimana penerapannya dapat meningkatkan kinerja model machine learning.

2. Bagi Industri

   Mendukung pengembangan model prediktif yang lebih akurat dan andal, yang pada akhirnya dapat membantu perusahaan dalam pengambilan keputusan yang lebih tepat dan berdampak pada peningkatan pelayanan serta kepuasan pelanggan.

3. Lingkungan Akademik

   Menyumbang informasi tambahan dalam penelitian mengenai cara mengoptimalkan parameter, khususnya metode Halving Random Search Cross Validation, yang hingga kini belum banyak dibahas secara mendalam.

   Menjadi referensi bagi mahasiswa, dosen, dan peneliti lain yang ingin mempelajari atau melakukan penelitian lebih lanjut tentang optimasi hyperparameter dan penerapannya dalam model machine learning.

BAB II

KAJIAN TEORI DAN HIPOTESIS

A. Landasan Teori

1. *Artificial Intelegent*

Kecerdasan Buatan atau *Artificial Intelligence* (AI) merupakan cabang ilmu komputer yang berfokus pada pengembangan sistem yang mampu melakukan tugas yang biasanya membutuhkan kecerdasan manusia, seperti mengenali pola, memahami bahasa, membuat keputusan, dan memecahkan masalah. Pembelajaran mesin atau *Machine Learning* (ML) adalah cabang dari kecerdasan buatan yang berfokus pada pengembangan algoritma dan model yang memungkinkan komputer untuk belajar dari data tanpa harus diprogram secara spesifik untuk setiap tugas.

Menurut Misnawati pada penelitian *ChatGPT*: Keuntungan, Risiko, Dan Penggunaan Bijak Dalam Era Kecerdasan Buatan Penggunaan tahun 2023, AI telah merambah hampir seluruh sektor, mulai dari kesehatan dan keuangan hingga transportasi dan pendidikan. Manfaat utama AI terletak pada kemampuannya meningkatkan efisiensi dan kualitas pekerjaan manusia di berbagai bidang. Secara fundamental, AI mampu mengolah dan menganalisis data dalam skala dan kecepatan yang jauh melampaui kemampuan manusia (Misnawati, 2023).

2. *Machine Learning*

Pembelajaran Mesin atau *Machine Learning* (ML) merupakan cabang kecerdasan buatan yang memungkinkan sistem komputer untuk belajar dan berkembang dari pengalaman tanpa perlu diprogram secara eksplisit.

Arie Nugroho dan kawan - kawan dalam penelitiannya tentang peningkatan performa *ensemble learning* pada segmentasi semantik gambar dengan teknik *oversampling* untuk class imbalance pada tahun 2019 menjelaskan bahwa machine learning terdiri dari dua jenis utama: *supervised learning* dan *unsupervised learning*. *Supervised learning* merupakan metode *machine learning* yang membutuhkan input manual berupa label klasifikasi pada data latih untuk membangun model, sedangkan *unsupervised learning* adalah metode yang tidak menggunakan label pada proses pelatihan data (Nugroho, Soeleman, Anggi Pramunendar, & Nurhindarto, 2023).

3. *Hyperparameter*

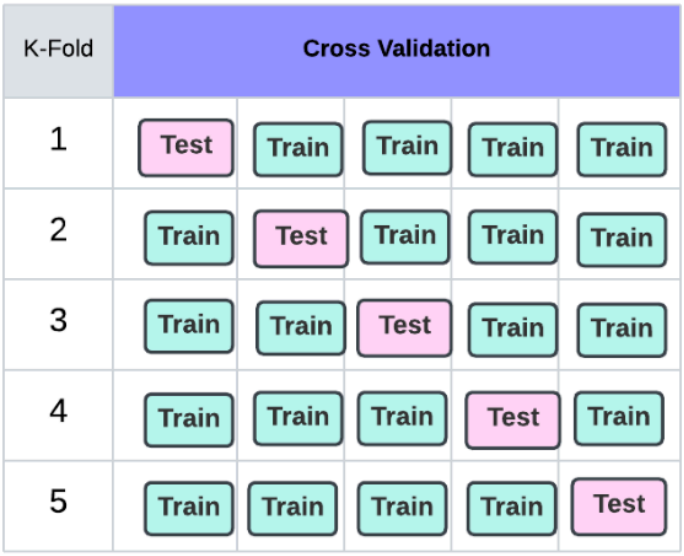
Hyperparameter adalah parameter pada algoritma machine learning yang nilainya diatur sebelum proses pelatihan model dimulai dan tidak berubah selama pelatihan. *Hyperparameter* berbeda dari parameter model seperti bobot (*weights*) dan bias yang diperbarui oleh algoritma pembelajaran selama proses pelatihan. Pada penelitian *hyperparameter tuning on classification algorithm with grid search* oleh wahyu nugraha tahun 2022 pemilihan *hyperparameter* berpengaruh dengan bagaimana cara model belajar dari data dan bagaimana kinerjanya pada data baru (Nugraha & Sasongko, 2022). Contoh *hyperparameter* meliputi jumlah pohon dalam algoritma *Random Forest*, nilai C dan gamma pada Support Vector Machine , atau jumlah hidden layer pada *Neural Network*.

Proses optimasi *hyperparameter* sangat penting untuk meningkatkan kinerja model. Pemilihan *hyperparameter* yang kurang tepat dapat menyebabkan model bekerja di bawah potensi maksimalnya, baik karena *overfitting* (model terlalu terpaku pada data latih) atau *underfitting* (model tidak menangkap pola data).

4. *K-Fold Cross Validation*

Cross-validation adalah teknik yang digunakan untuk menilai kinerja model *machine learning* dengan membagi dataset menjadi beberapa *subset* atau yang biasa disebut *fold*. Hal tersebut bertujuan untuk mengukur seberapa baik model mampu melakukan generalisasi terhadap data baru yang belum pernah dilihat sebelumnya, sehingga membantu mencegah masalah seperti *overfitting* dan *underfitting*. *Cross-validation* sangat penting karena performa model yang diukur hanya pada data pelatihan sering kali tidak mencerminkan kinerjanya pada data dunia nyata. Dengan *cross-validation*, model diuji pada beberapa subset data yang berbeda, memberikan estimasi kinerja yang lebih *robust* dan akurat.

Menurut Wijianto dalam penelitiannya yang berjudul Teknik *K-Fold Cross Validation* untuk Mengevaluasi Kinerja Mahasiswa (2024), kelebihan menggunakan *K-Fold Cross Validation* terletak pada kemampuannya untuk memberikan estimasi kinerja model yang lebih stabil dan akurat (Wijiyanto, Pradana, Sopingi, & Atina, 2024). Hal ini dicapai dengan menghitung rata-rata hasil dari K iterasi, sehingga risiko empiris menjadi lebih rendah dibandingkan jika hanya membagi dataset secara sederhana ke dalam data pelatihan dan data pengujian(Anggraini, Sucipto, & Indriati, 2018).



Gambar 2.1.1 Cross Validation 5 Fold

5. *Hyperparameter Tuning*

Hyperparameter tuning adalah proses mencari kombinasi *hyperparameter* optimal pada algoritma machine learning untuk meningkatkan performa model. Terdapat beberapa metode dalam *hyperparameter* tuning yaitu *Manual Search, Grid Search, Random Search, Halving Search,* dan *Bayesian Optimization*. Namun pada penelitian ini hanya akan berfokus pada metode *Halving Random Search Cross Validation.*

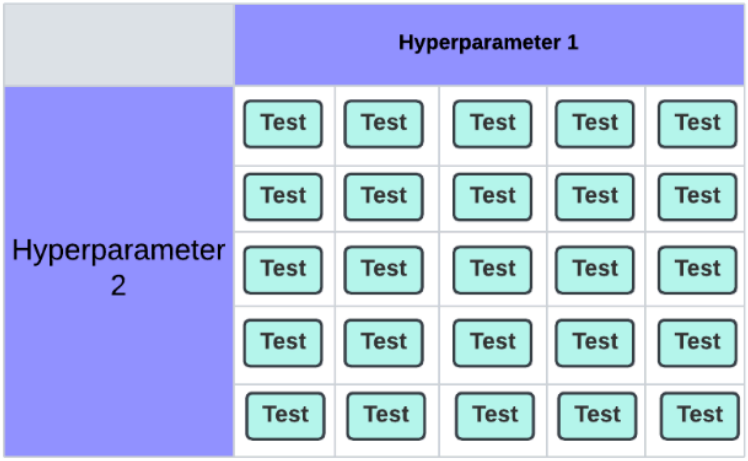
Menurut Iik Muhammad Malik Matin dalam penelitiannya yang berjudul *Hyperparameter Tuning* Menggunakan *GridSearchCV* pada *Random Forest* untuk Deteksi *Malware* tahun 2023, *hyperparameter tuning* memiliki peran yang sangat penting pada algoritma seperti *Random Forest,* karena performa algoritma tersebut sangat bergantung pada *hyperparameter* yang ditentukan (Muhamad & Matin, 2023).

Hyperparameter tuning sering dikombinasikan dengan *Cross Validation* untuk meningkatkan robustitas dan akurasi hasil prediksi model. Namun, metode ini memiliki kelemahan utama, yaitu biaya komputasi yang tinggi karena model harus dilatih berulang kali, terutama saat digunakan pada dataset berukuran besar.

6. *Grid Search Cross Validation*

Pada penelitian *Hyperparameter Tuning* Menggunakan *GridSearchCV* pada *Random Forest* untuk Deteksi *Malware* oleh Iik Muhammad Malik Matin (2023), dijelaskan bahwa

GridSearchCV adalah metode *hyperparameter tuning* yang memungkinkan pemindaian sejumlah kombinasi *hyperparameter* yang telah ditentukan. Metode ini menerapkan setiap kombinasi *hyperparameter* pada model, kemudian mengevaluasi performa model tersebut menggunakan teknik *cross-validation* untuk menentukan kombinasi terbaik (Muhamad & Matin, 2023).



Gambar 2.1.2 Ilustrasi Grid Search

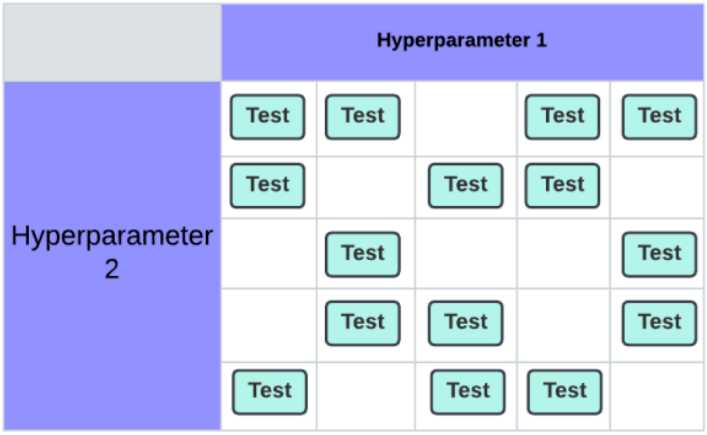
Pada *Grid Search,* setiap kombinasi *hyperparameter* diuji menggunakan teknik cross-validation, di mana dataset dibagi menjadi beberapa subset . Model dilatih pada sebagian data dan diuji pada sisa data secara bergantian. Rata-rata kinerja model pada setiap iterasi digunakan untuk menentukan kualitas kombinasi *hyperparameter* tersebut.

Kelebihan *Grid Search Cross Validation* adalah memberikan hasil yang akurat karena semua kombinasi *hyperparameter* dievaluasi. Namun, metode ini memiliki kelemahan, yaitu biaya komputasi yang tinggi, terutama ketika ruang pencarian *hyperparameter* besar atau dataset yang digunakan berukuran besar.

7. Random Search Cross Validation

Random Search Cross Validation adalah teknik optimasi *hyperparameter* yang dilakukan dengan memilih secara acak kombinasi *hyperparameter* dari ruang pencarian yang telah ditentukan. Menurut Tita Aulia Putri dalam penelitiannya yang berjudul Penerapan Tuning *Hyperparameter Random Search CV* pada *Adaptive Boosting* untuk Prediksi Kelangsungan Hidup Pasien Gagal Jantung tahun 2022, *Random Search Cross Validation* merupakan metode pencarian model dengan menentukan nilai *hyperparameter*

algoritma pembelajaran secara acak. Teknik ini memilih kombinasi *hyperparameter* secara acak untuk melatih model, memungkinkan eksplorasi ruang pencarian yang lebih luas tanpa memerlukan evaluasi semua kombinasi yang mungkin, seperti yang dilakukan pada *Grid Search Cross Validation* (Putri, Widiharih, & Santoso, 2023).



Gambar 2.1.3 Ilustrasi *Random Search*

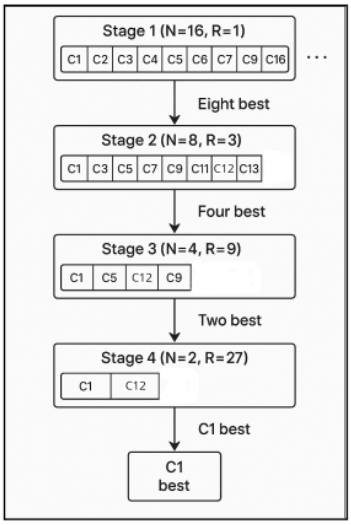
Random Search memiliki kelemahan utama, yaitu tidak menjamin pencapaian kombinasi *hyperparameter* yang benar-benar optimal karena pemilihannya bersifat acak. Selain itu, efisiensi metode ini sangat bergantung pada jumlah iterasi yang dilakukan; jika iterasi yang ditentukan terlalu sedikit, hasil yang diperoleh mungkin kurang representatif dan tidak mencakup kombinasi *hyperparameter* terbaik.

8. *Successive Halving*

Menurut Daniel S. Soper dalam penelitiannya yang berjudul *Hyperparameter Optimization Using Successive Halving with Greedy Cross Validation* tahun 2022, algoritma *successive halving* menjadi salah satu teknik yang menonjol dalam optimasi *hyperparameter*. Secara singkat, *successive halving* adalah metode optimasi *hyperparameter* iteratif di mana jumlah kandidat model berkurang secara eksponensial pada setiap iterasi, sementara jumlah data pelatihan yang digunakan meningkat secara eksponensial. Strategi utama algoritma ini mirip dengan turnamen eliminasi tunggal (Soper, 2022).

Proses dimulai dengan menggunakan sebagian kecil data pelatihan untuk dengan cepat mengidentifikasi dan mengeliminasi kandidat model yang kurang menjanjikan. Kandidat model yang lolos ke tahap berikutnya dievaluasi lebih mendalam dengan

proporsi data pelatihan yang lebih besar. Proses ini berlanjut secara bertahap hingga hanya tersisa beberapa kandidat model terbaik. Model-model ini kemudian dilatih dan dievaluasi menggunakan seluruh data yang tersedia. Seperti yang diilustrasikan pada Gambar 2.1.4, pada awal iterasi, model dilatih menggunakan sebagian kecil dataset, dan model dengan skor terbaik dipilih untuk melanjutkan ke tahap berikutnya. Pada iterasi terakhir, seluruh dataset digunakan untuk menghasilkan skor akhir.



Gambar 2.1.4 Ilustrasi *Successive Halving*

9. *Halving Random Search Cross Validation*

Halving Random Search Cross Validation merupakan metode optimasi *hyperparameter* yang menggabungkan strategi pencarian acak dari *Random Search* dengan pendekatan *successive halving*, yaitu teknik yang secara bertahap mengeliminasi kombinasi *hyperparameter* dengan performa rendah dan hanya melanjutkan kombinasi terbaik ke tahap evaluasi berikutnya. Tujuan dari metode ini adalah untuk meningkatkan efisiensi proses tuning dengan memfokuskan sumber daya komputasi pada kandidat yang paling menjanjikan sejak awal. Selain itu, untuk mengurangi risiko *overfitting* maupun *underfitting*, metode ini juga memanfaatkan *Cross Validation* dalam proses evaluasinya.

B. Kajian Hasil Penelitian Terdahulu

Penelitian terdahulu menunjukkan bahwa metode *successive halving*, baik standar maupun *greedy*, memang mampu mempercepat proses *tuning hyperparameter*, meskipun hasil model yang diperoleh tidak selalu optimal secara akurasi maupun error (Soper, 2023). Meskipun demikian, efisiensi ini menjadikannya alternatif menarik dibanding pencarian menyeluruh seperti grid search.

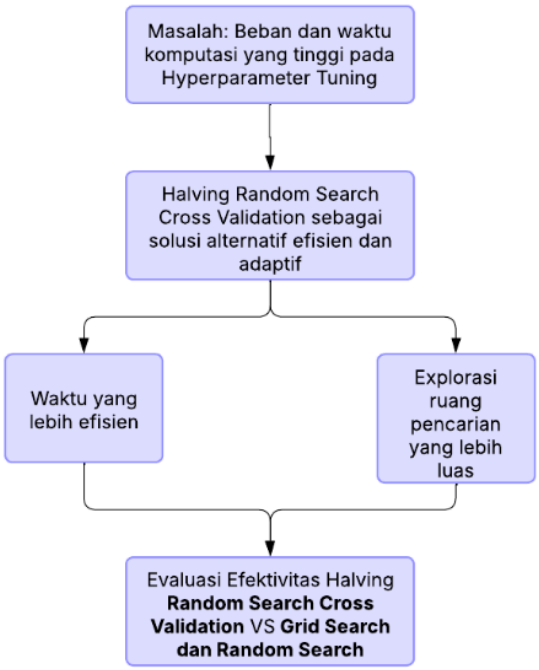
Schmucker et al. (2021) kemudian memperluas metode ini ke dalam bentuk *Asynchronous Successive Halving* untuk optimasi multi-tujuan, dengan hasil yang menjanjikan pada efisiensi dan paralelisasi. Namun, fokus utama penelitian-penelitian tersebut masih terbatas pada performa model akhir, tanpa membahas secara eksplisit berapa besar waktu komputasi yang dihemat atau sejauh mana ruang hyperparameter yang berhasil dijelajahi oleh metode seperti *Halving Random Search Cross Validation* .

Sebaliknya, metode seperti *Grid Search Cross Validation* memang memberikan hasil yang konsisten dan akurat, sebagaimana ditunjukkan dalam studi Wahyu Aprilliandhika & Abdulloh (2024) dan Jamiluddin (2024), namun memerlukan waktu eksekusi yang jauh lebih lama karena eksplorasi menyeluruh terhadap kombinasi hyperparameter.

Penelitian oleh Anggraini et al. (2023) menunjukkan bahwa *Random Search Cross Validation* efektif dalam optimasi hyperparameter AdaBoost pada data medis, dengan akurasi 0,85 dan AUC 0,897. Namun, studi ini belum membahas luas ruang pencarian maupun waktu komputasi yang diperlukan.

Berdasarkan celah tersebut, penelitian ini berfokus pada analisis waktu komputasi hyperparameter pada metode *Halving Random Search Cross Validation*, aspek yang belum banyak disorot dalam literatur sebelumnya. Dengan demikian, penelitian ini tidak hanya mengevaluasi performa model, tetapi juga menilai efisiensi dan kelayakan *Halving Random Search Cross Validation* Sebagai pendekatan hyperparameter optimization yang seimbang antara kecepatan dan cakupan eksplorasi.

C. Kerangka Berpikir



Gambar 2.3 kerangka berpikir

Proses pembangunan model machine learning yang akurat sangat bergantung pada pemilihan hyperparameter yang optimal. Metode *Grid Search Cross Validation* mampu memberikan akurasi tinggi karena mengevaluasi semua kombinasi, tetapi memiliki kelemahan utama yaitu waktu komputasi yang sangat besar. *Random Search Cross Validation* lebih efisien secara waktu karena hanya mengevaluasi sebagian kombinasi, tetapi akurasinya tidak selalu optimal dan ruang pencariannya tidak terstruktur.

Sebagai solusi, dikembangkanlah metode *Halving Random Search Cross Validation* yang menggabungkan efisiensi *random search* dengan eliminasi bertahap dari s*uccessive halving*. Metode ini melakukan evaluasi awal dengan resource minimal dan hanya melanjutkan kombinasi terbaik ke tahap berikutnya. Hal ini memungkinkan eksplorasi ruang pencarian yang luas tanpa beban komputasi sebesar *grid search*.

Namun, belum banyak penelitian yang secara eksplisit membandingkan *Halving Random Search Cross Validation*_dari sisi waktu komputasi yang dibutuhkan dan sejauh mana ruang kombinasi hyperparameter yang berhasil dijelajahi. Oleh karena itu, penelitian

ini fokus menganalisis efisiensi *Halving Random Search Cross Validation* dibanding *Grid Search Cross Validation* dan *Random Search Cross Validation*, baik dari sisi durasi pencarian maupun efektivitas model yang dihasilkan.

D. Hipotesis

Dengan menggunakan metode *Halving Random Search Cross Validation*, proses pencarian *hyperparameter* yang optimal dapat dilakukan secara lebih efisien, baik dari segi waktu dan secara tidak langsung biaya komputasi. Metode ini memungkinkan percepatan proses tanpa mengorbankan akurasi model, terutama saat diterapkan pada dataset berukuran besar.

BAB III
METODE PENELITIAN

Penelitian ini menggunakan pendekatan kuantitatif yang bertujuan untuk mengumpulkan dan menganalisis data secara sistematis guna mengevaluasi efektivitas berbagai metode optimasi *hyperparameter* dalam membangun model machine learning. Data yang dikumpulkan berupa waktu komputasi, akurasi, presisi, dan *recall* dari model akhir yang dihasilkan oleh masing-masing metode optimasi *Grid Search Cross Validation*, *Random Search Cross Validation* dan *Halving Random Search Cross Validation*.

Dataset yang digunakan dalam penelitian ini adalah *Internet Service Churn*, yang diunggah di platform Kaggle pada tahun 2021. Dataset ini memiliki lisensi CC0: Public Domain, sehingga dapat digunakan secara bebas untuk keperluan penelitian dan pengembangan. Data ini berasal dari perusahaan telekomunikasi dan berisi informasi pelanggan dengan total 11 kolom dan terdiri dari 72.274 *record* data, termasuk label target yang menunjukkan apakah seorang pelanggan masih berlangganan atau telah berhenti berlangganan.

Metode analisis data dalam penelitian ini mengacu pada standar prosedur data mining yang dikenal dengan CRISP-DM (Cross-Industry Standard Process for Data Mining). CRISP-DM merupakan kerangka kerja yang umum digunakan dalam proses penelitian *data mining* karena bersifat sistematis dan fleksibel (Nila, Firliana, & Sucipto, 2023). Penelitian ini mengikuti tahapan-tahapan yang terdapat dalam fase CRISP-DM sebagai berikut:

1. *Business Understanding,* Seperti pada yang sudah dijabarkan pada latar belakang permasalahan yang ingin dipecahkan atau diteliti adalah efektifitas dari metode *halving random search cross validation* jika dibandingkan dengan metode optimasi lain.

2. *Data Understanding,* Penulis akan menggunakan *Internet Service Churn* yang tersedia pada *platform* kaggle.

3. *Data Preparation,* Data preprocessing yang penulis lakukan adalah penghapusan entri data yang kosong dan normalisasi data dengan metode *Min-Max Scaler*.

4. *Modeling,* Pada tahap ini dilakukan proses optimasi *hyperparameter* terhadap model KNN, *Decision Tree*, SVM, dan *Gaussian Naive Bayes* menggunakan *10 K-Fold Cross Validation*. Hasil dari tahap ini mencakup waktu yang dibutuhkan masing-masing metode optimasi untuk menemukan kombinasi hyperparameter terbaik, serta evaluasi kinerja model *final* berdasarkan metrik akurasi, presisi, dan *recall*.

5. *Evaluation,* Hasil dari tahap modeling akan dianalisis dan disajikan dalam bentuk visualisasi grafik, untuk memudahkan interpretasi dan perbandingan performa antar metode optimasi.

6. *Deployment,* Pada tahap ini penulis akan menyajikan hasil dari analisis efektifitas dari metode *halving random search cross validation* yang telah dilakukan.

Berikut adalah tabel yang berisi detail *hyperparameter* yang akan di tuning serta *list value* yang akan digunakan untuk metode optimasi *Grid Search Cross Validation* dan *Random Search Cross Validation*. Dengan total pelatihan yang harus dilakukan oleh kedua metode tersebut adalah 456 kali untuk metode *grid search* dan 46 kali untuk metode *random*. Metode random search akan menggunakan jumlah pencarian 10% dari total jumlah kombinasi.

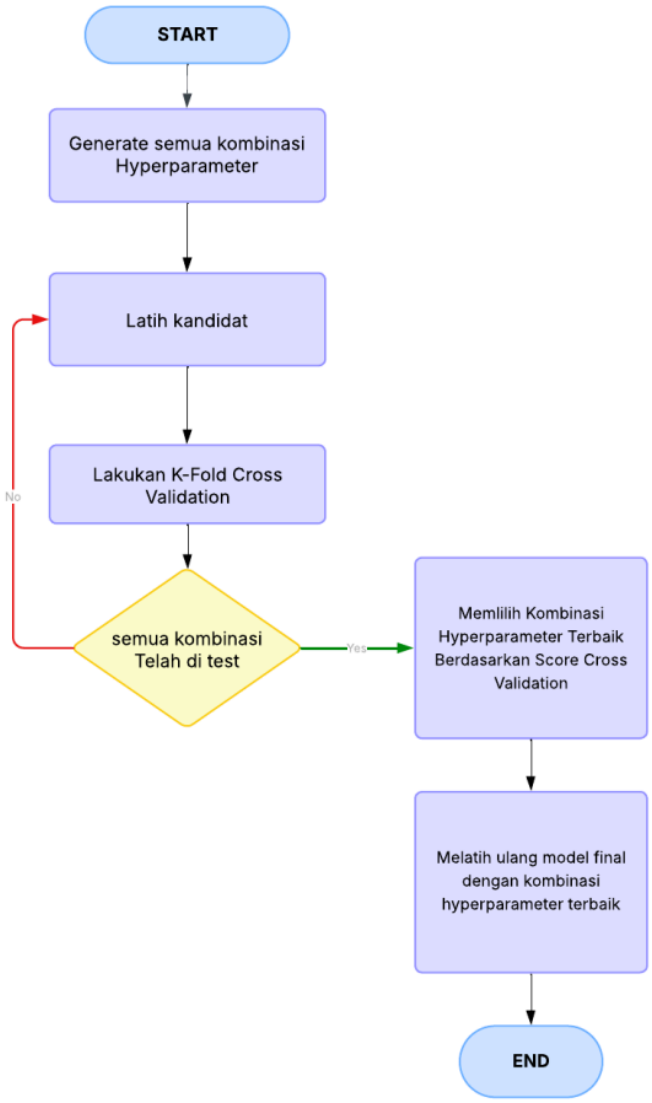
Berikut adalah alur dan proses tiap - tiap optimasi hyperparameter yang akan dibandingkan dengan *halving random search cross validation.*

Tabel 3.1 *List Hyperparameter* 1

Model	Hyperparameter	List value	Jumlah Semua Kombinasi unik
KNN	*n_neighbors*	3, 5, 7, 9, 11, 13, 15, 17, 19, dan 21	200 Kombinasi
	algorithm	'ball_tree' dan 'kd_tree'	
	weight	uniform, distance	
	metric	'cityblock', 'cosine', 'euclidean', 'manhattan', dan 'nan_euclidean'	
Decision Tree	*criterion*	'gini', 'entropy', dan 'log_loss'	240 Kombinasi
	max_depth	None, 3, 5, 7, 8, 9, 10, 15, 20, dan 25.	
	min_sample_leaf	10, 20, 30, 40, 50, 100, 200, dan 500	
Support Vector Machine	*C*	0.001, 1.0	6 Kombinasi
	kernel	'linear', 'rbf', dan 'poly'	
Gaussian Naive Bayes	*var_smoothing*	0.001, 0.00316, 0.01, 0.0316, 0.1, 0.316, 1.0, 3.16,	10 kombinasi

		10.0, 100.0	

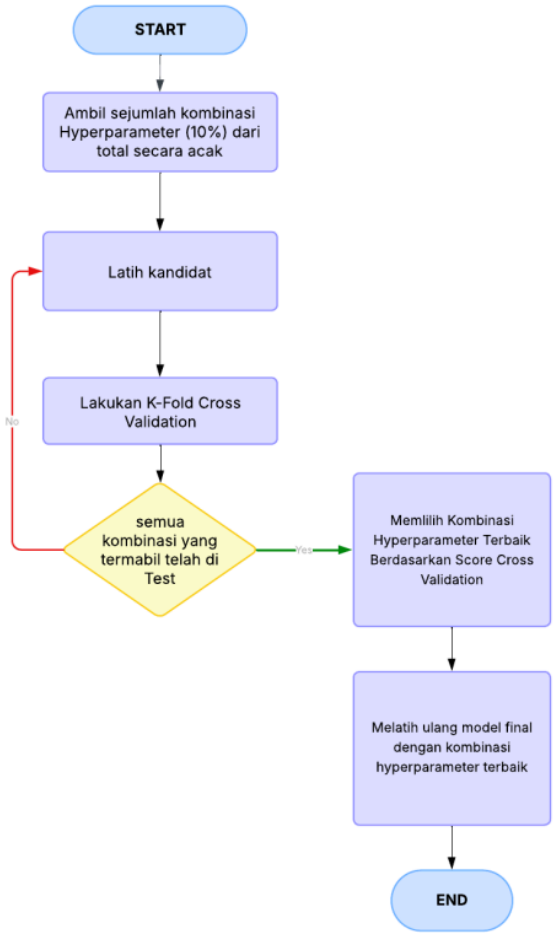
1. Proses *Grid Search Cross Validation*



Gambar 3.1 Proses *Grid Search Cross Validation*

Proses *Grid Search Cross Validation* dimulai dengan menghasilkan seluruh kombinasi *hyperparameter* yang mungkin. Setiap kombinasi kemudian diuji menggunakan 10 *K-Fold Cross Validation* untuk mengevaluasi performanya. Proses ini diulang untuk seluruh kombinasi. Setelah itu, kombinasi dengan skor *cross-validation* terbaik dipilih sebagai konfigurasi optimal. Kemudian, model akhir dilatih ulang menggunakan kombinasi *hyperparameter* menggunakan seluruh *dataset*.

2. Proses *Random Search Cross Validation*



Gambar 3.2 Proses *Random Search Cross Validation*

Proses dimulai dengan memilih sejumlah kecil kombinasi *hyperparameter* dalam penelitian ini 10% dari total kombinasi untuk tiap model yang sedang di tuning dipilih secara acak dari seluruh ruang kemungkinan. Setiap kombinasi yang terpilih kemudian dievaluasi menggunakan *10 K-Fold Cross Validation.* Proses ini berlanjut hingga semua kombinasi yang terambil selesai diuji. Setelah itu, dipilih kombinasi hyperparameter dengan skor *cross-validation* terbaik. Terakhir, model akhir dilatih ulang menggunakan kombinasi terbaik menggunakan seluruh *dataset.*

Sedangkan berikut adalah tabel yang berisi detail *hyperparameter* dan *list value* yang akan digunakan untuk metode optimasi *Halving Random Cross Validation*. Pembedaan ini dikarenakan jika menggunakan tabel yang sama dengan metode *grid search* dan *random search* tidak akan maksimal pada teknik *halving random search cross validation* dikarenakan ruang pencarian yang sangat terbatas, di mana keunggulan utama dari metode ini adalah efisiensi ketika mencari di ruang yang luas. Disini penelitian akan menggunakan fungsi *'loguniform'* dan *'randint'* dari *library scipy* yang akan mengembalikan bilangan acak dari rentang yang ditentukan atau jika perlu sesuai dengan distribusi tertentu.

Tabel 3.2 *List Hyperparameter* 2

Model	Hyperparameter	List value	Jumlah Semua Kombinasi unik
KNN	*n_neighbors*	‘randint(3, 30)’ yang akan mengembalikan bilangan bulat acak mulai 3 sampai 29	810 Kombinasi
	algorithm	'ball_tree' dan 'kd_tree'	
	weight	uniform, distance	
	metric	'cityblock', 'cosine', 'euclidean', 'manhattan', dan 'nan_euclidean'	
Decision Tree	*criterion*	'gini', 'entropy', dan 'log_loss'	37,500 Kombinasi
	max_depth	‘[None, range(3, 26)]’ None dan bilangan bulat mulai 3 sampai 26	
	min_sample_leaf	‘randint(10, 501)’ yang akan mengembalikan bilangan bulat acak mulai 10 sampai 500	
Support Vector Machine	*C*	‘loguniform(1e-3, 1e2)’ akan mengembalikan bilangan acak antara 0.001 dan 100, tetapi dengan cara yang lebih adil untuk angka kecil dan besar.	Tidak terbatas
	kernel	'linear', 'rbf', dan 'poly'	
Gaussian Naive Bayes	*var_smoothing*	‘loguniform(1e-3, 1e2)’ akan mengembalikan bilangan acak antara 0.001 dan 100	Tidak terbatas

Proses *Halving Random Search Cross Validation* dimulai dengan penentuan jumlah iterasi dan kandidat awal yang sesuai dengan jumlah *entry data* dan

jumlah *folds*. Dalam *library scikit learn* pertama menghitung *min_resources* dengan rumus

$$min\_resources = 2 \times kelas\ klasifikasi \times fold \quad (3.1)$$

Dalam kasus ini maka nilai dari *min_resources* adalah 40, kemudian hitung iterasi yang memungkian dari jumlah entry data, setiap iterasi akan menggunakan *resource* dengan rumus

$$n\_resources\_i = factor^{i} * min\_resources \quad (3.2)$$

Dalam kasus ini maksimal iterasi yang dapat dilakukan dengan *min_resources* 40 dan *factor* 3 adalah 7 karena $3^7 \times 40$ = 87,480 yang berarti data entry telah habis pada iterasi ke 6.

Sebaliknya jumlah kandidat yang akan di test tiap iterasi dihitung dengan rumus

$$n\_candidates\_i = n\_candidates\ //\ (factor^{i}) \quad (3.3)$$

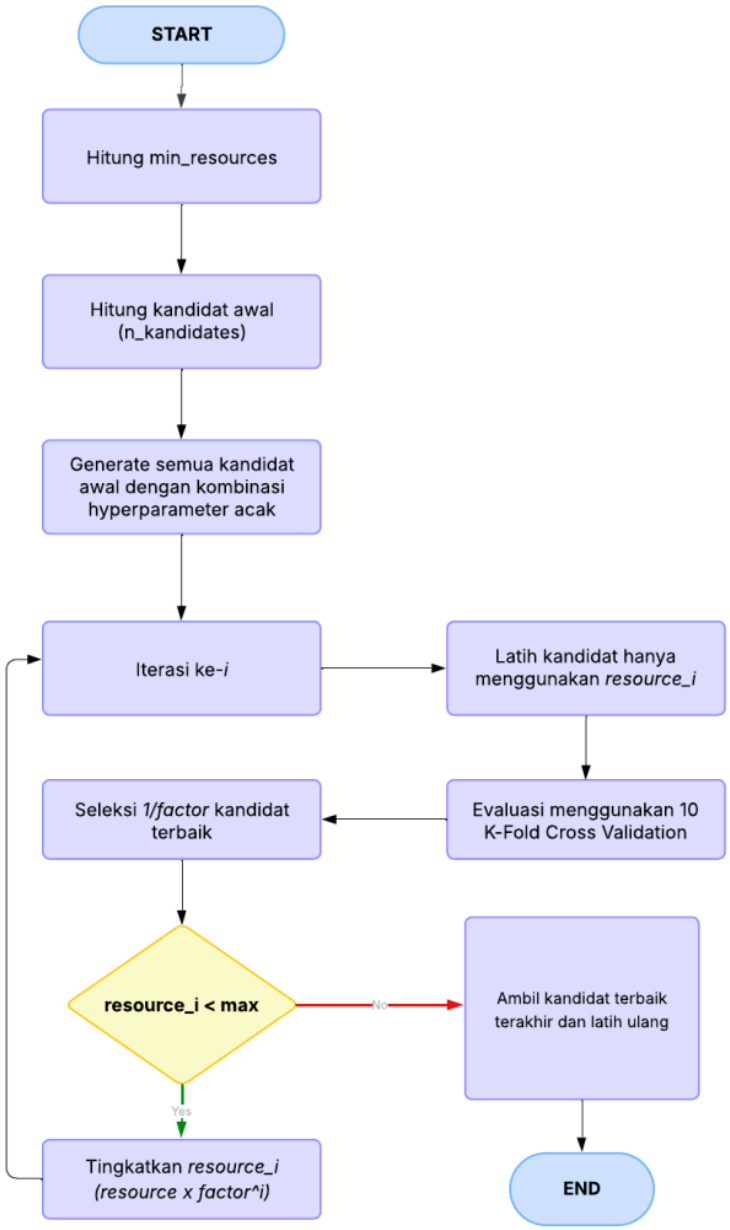
Sehingga untuk menemukan jumlah kandidat awal adalah $factor^{i-1}$ atau 3^{7-1} , jadi kandidat awal adalah 729 kombinasi hyperparameter. Dari sini kemudian bisa dibuat tabel iterasi *Successive Halving*.

Tabel 3.3 Tabel *Succesive Halving*

Iterasi	Kandidat	Resource
0	729	40
1	243	120
2	81	360
3	27	1,080
4	9	3.240
5	3	9.720
6	1	29.160

Baru kemudian proses Halving Random Search Cross Validation dapat dimulai dengan melatih semua kandidat awal dengan kombinasi hyperparameter acak dengan

dataset awal yang kecil, kemudian dilakukan seleksi 1/*factor* kandidat dengan skor *cross validation* terbaik, lakukan iterasi selanjutkan dan tambahkan dataset sesuai dengan Tabel 3.3 *Tabel Succesive Halving* hingga kandidat hanya tersisa 1 atau dataset tidak cukup.



Gambar 3.3 Proses *Halving Random Search Cross Validation*

Teknik yang digunakan untuk memperoleh waktu komputasi yang diperlukan oleh metode optimasi serta matriks evaluasi adalah sebagai berikut. Proses ini akan diulang sebanyak tiga kali guna memperoleh nilai rata-rata.

1. Waktu komputasi

   Waktu komputasi yang diperlukan diukur menggunakan fungsi time.*perf_counter* dari *standard library Python*. Hasil yang diperoleh merupakan durasi detik yang dibutuhkan metode untuk menyelesaikan proses optimasi *hyperparameter*.

2. Matriks penilaian.

   Evaluasi performa model dilakukan menggunakan fungsi *accuracy_score*, *precision_score*, dan *recall_score* dari *library Scikit-learn.* Akurasi digunakan untuk mengukur sejauh mana model dapat mengklasifikasikan data dengan benar. (Gusti Tammam, Indriati, & Sucipto, 2018). Recall mengukur kemampuan model dalam mendeteksi seluruh data positif yang benar(Sucipto, Prasetya, & Widiyaningtyas, 2025)

BAB IV
HASIL DAN PEMBAHASAN

A. Data Hasil Penelitian

1. Hasil Eksplorasi Data

a. Deskripsi Dataset

Tahap awal penelitian ini dilakukan dengan melakukan eksplorasi terhadap dataset *Internet Service Provider Customer Churn* yang digunakan sebagai objek penelitian. Dataset ini diperoleh dari platform *Kaggle* dan berisi 72.274 baris data dengan 10 fitur *independen* serta 1 fitur target yaitu *Churn* (status pelanggan berhenti atau tetap berlangganan).

Tabel 4.1 Deskripsi Atribut Dataset

Nama Fitur	Tipe Data	Keterangan
id	Numeric	Nomor identitas unik pelanggan
is_tv_subscriber	Numerik (biner)	Menunjukkan apakah pelanggan berlangganan layanan TV
is_movie_package_subscriber	Numerik (biner)	Menunjukkan apakah pelanggan berlangganan paket film
subscription_age	Numerik (float)	Lama waktu pelanggan berlangganan (bulan)
bill_avg	Numerik	Rata-rata tagihan pelanggan
reamining_contract	Numerik	Sisa kontrak pelanggan dalam bulan
service_failure_count	Numerik	Jumlah kegagalan layanan yang dialami pelanggan
download_avg	Numerik	Rata-rata kecepatan unduh pelanggan (Mbps)
upload_avg	Numerik	Rata-rata kecepatan unggah pelanggan (Mbps)
download_over_limit	Numerik	Frekuensi pelanggan

		melebihi batas kebijakan pemakaian wajar
churn	Numerik (biner)	Status berhenti berlangganan (1 = churn, 0 = aktif)

b. Statistik Deskriptif

Analisis statistik deskriptif dilakukan terhadap seluruh atribut numerik.

Tabel 4.2 Statistik Deskriptif Dataset

Fitur	*Mean*	*Median*	*Std. Dev*	*Min*	*Max*
subscription_age	11.72	11.86	6.57	0.00	24.00
bill_avg	24.62	24.00	14.47	0.00	48.00
reamining_contract	4.86	5.00	3.97	0.00	24.00
service_failure_count	0.35	0.00	0.77	0	5
download_avg	12.31	12.30	7.26	0.00	24.00
upload_avg	2.97	2.80	1.77	0.00	6.00
download_over_limit	0.38	0.00	0.78	0	5
churn	0.20	0.00	0.40	0	1

Berdasarkan hasil analisis deskriptif pada Tabel 4.2, dapat diketahui bahwa rata-rata pelanggan telah berlangganan selama sekitar 11,7 bulan dengan rata-rata tagihan bulanan sebesar 24,6. Selain itu, proporsi pelanggan yang berhenti berlangganan tercatat sekitar 20% dari total data.

c. Distribusi Data

Distribusi target *churn* menunjukkan bahwa dataset seimbang antara pelanggan yang masih berlangganan dan yang sudah putus.

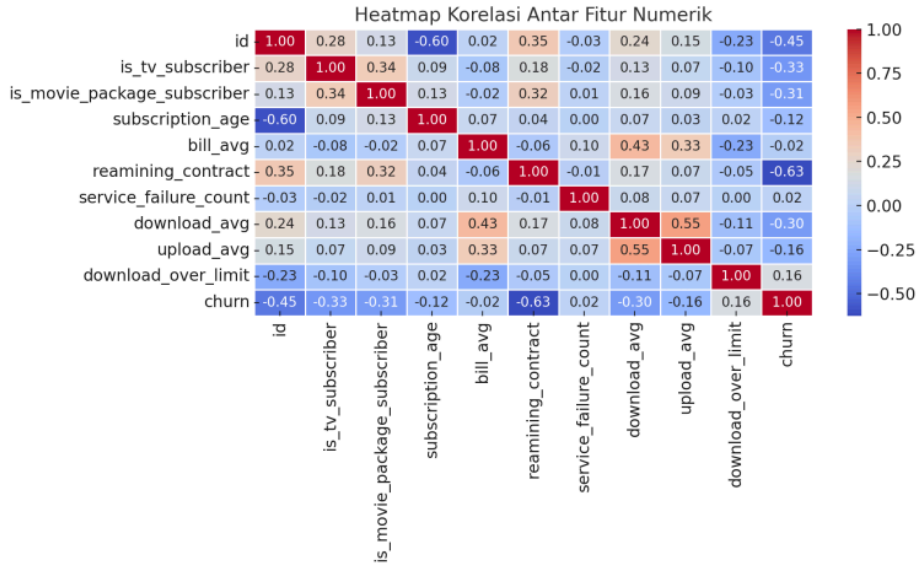
Tabel 4.3 Distribusi Pelanggan Berdasarkan Status Churn

Churn	Jumlah Pelanggan	Presentase
0	32224	44,6%
1	40050	55,4%

Berdasarkan Tabel 4.3 Distribusi Pelanggan Berdasarkan Status *Churn*, dapat diketahui bahwa pelanggan masih aktif berlangganan dengan persentase sebesar 44,6%, sedangkan pelanggan yang berhenti berlangganan hanya sebesar 55,4%.

d. Korelasi Antar Fitur

Analisis korelasi menggunakan metode Pearson menunjukkan hubungan antara fitur numerik terhadap variabel target churn.



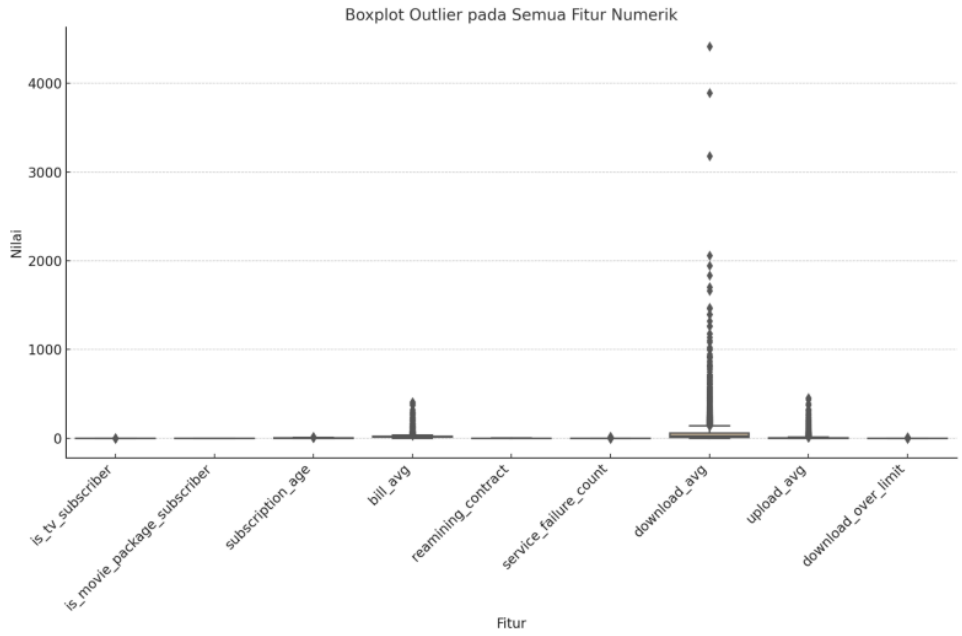
Gambar 4.1 Heatmap Korelasi Antar Fitur Numerik

Berdasarkan Gambar 4.1 Heatmap Korelasi Antar Fitur Numerik, Dari hasil korelasi terlihat bahwa semakin lama pelanggan berlangganan maka kemungkinan berhenti semakin rendah. Sebaliknya, pelanggan dengan tagihan tinggi cenderung memiliki risiko *churn* yang lebih besar.

e. Analisis *Outliner*

Deteksi outlier dilakukan menggunakan metode IQR (*Interquartile Range*). Untuk

mendeteksi nilai ekstrim pada fitur.



Gambar 4.2 Boxplot Outlier pada Seluruh Fitur Numerik

Berdasarkan Gambar 4.2 Fitur *bill_avg*, *download_avg* dan *upload_avg* memiliki nilai ekstrem yang cukup mencolok. Namun nilai tersebut tetap dipertahankan karena dianggap mewakili kondisi pelanggan tertentu, seperti pelanggan baru atau pelanggan dengan penggunaan dengan kecepatan rendah.

2. Pra-pemrosesan Data

a. Pemilihan Fitur

Berdasarkan hasil analisis korelasi, fitur yang memiliki hubungan paling signifikan terhadap *churn* adalah *subscription_age*, *bill_avg*, *reamining_contract*, *service_failure_count*, *download_avg*, dan *upload_avg*. Sementara itu, fitur *id* dihapus karena hanya berfungsi sebagai identitas pelanggan, sedangkan fitur *is_tv_subscriber*, *is_movie_package_subscriber*, dan *download_over_limit* tetap dipertahankan karena secara logis dapat memengaruhi perilaku pelanggan dalam berhenti berlangganan.

b. Penanganan Nilai Kosong

Berdasarkan hasil eksplorasi sebelumnya, ditemukan bahwa beberapa fitur memiliki nilai kosong, terutama pada kolom *reamining_contract*, *download_avg*, dan *upload_avg*.

Tabel 4.4 Jumlah Nilai Kosong pada Setiap Fitur

Fitur	Jumlah nilai kosong	Presentase (%)
reamining_contract	21.572	29,9%
download_avg	381	0,5%
upload_avg	381	0,5%

Untuk menjaga integritas data, nilai kosong pada *reamining_contract* diisi menggunakan nilai median karena distribusi datanya tidak simetris. Sedangkan Nilai kosong pada *download_avg* dan *upload_avg* diisi menggunakan rata-rata karena keduanya memiliki distribusi yang relatif normal.

c. Transformasi dan Konversi Data

Karena dataset ini tidak mengandung fitur kategorikal non-numerik, maka tidak diperlukan proses encoding tambahan.

d. Normalisasi Data

Semua fitur numerik *subscription_age, bill_avg, reamining_contract, sevice_failure_count, download_avg, upload_avg,* dan *download_over_linit* memiliki rentang nilai yang berbeda-beda. Untuk menghindari dominasi fitur dengan skala besar terhadap fitur dengan skala kecil, dilakukan proses normalisasi menggunakan metode *Min-Max Scaling* yang menghasilkan nilai pada rentang 0 hingga 1 untuk setiap fitur numerik. Dengan demikian, semua variabel memiliki bobot yang seimbang ketika digunakan dalam proses pelatihan model.

Tabel 4.5 Sample Dataset Setelah Pra-Pemrosesan Data

Fitur	value				
is_tv_subcriber	1	0	1	0	0
is_movie_package_subcriber	0	0	0	0	0
subscription_age	0.933697	0.642746	0.696568	0.537441	0.500000
bill_avg	0.061576	0.000000	0.039409	0.051724	0.000000
reamining_contract	0.047945	0.195205	0.000000	0.195205	0.195205
service_failure_count	0	0	0	1	0

download_avg	0.001903	0.000000	0.003103	0.000000	0.000000
upload_avg	0.005074	0.000000	0.001985	0.000000	0.000000
download_over_limit	0	0	0	0	0
churn	0	1	1	1	1

e. Pembagian Data

Dataset kemudian dibagi menjadi dua bagian, yaitu 90% data latih dan 10% data uji. Pembagian ini dilakukan secara acak dengan parameter *random_state = 42*.

Tabel 4.6 Pembagian Dataset

	jumlah baris	presentase
Data Latih	65,046	90,00%
Data uji	7,228	10,00%

3. Proses Pelatihan

a. Hardware

Sebagai referensi hardware yang digunakan untuk pelatihan adalah *Google Colab* dengan runtime *CPU* default. Dengan spesifikasi lengkap sebagai berikut:

Tabel 4.7 Spesifikasi Hardware

Spesifikasi	
OS	Ubuntu 22.04.4 LTS x86_64
Host	Google Compute Engine
Kernel	6.6.105+
CPU	Intel Xeon (2) @ 2.199GHz
RAM	12975 MiB

b. Kode pelatihan

```python
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, precision_score, recall_score
from sklearn.neighbors import KNeighborsClassifier
from time import perf_counter
import joblib


model = KNeighborsClassifier()
params = params_dict_knn_1
cv = GridSearchCV(estimator=model, param_grid=params, cv=10, verbose=2)

start = perf_counter()
cv.fit(X_train, y_train)
elapsed = perf_counter() - start
print(f"Elapsed time: {elapsed:.2f} seconds")

print("Best Parameters:", cv.best_params_)
print("Best Accuracy:", cv.best_score_)

acuracy = accuracy_score(y_test, cv.predict(X_test))
precicion = precision_score(y_test, cv.predict(X_test))
recall = recall_score(y_test, cv.predict(X_test))

print("Akurrasi pada data test:", acuracy)
print("Presisi pada data test:", precicion)
print("Recall pada data test:", recall)
joblib.dump(cv.best_estimator_, 'knn_grid_1.pkl')
```

Gambar 4.3 Kode *Grid Search CV*

Proses pelatihan model dengan *Grid Search Cross Validation* (Gambar 4.3 Kode *Grid Search CV*) diawali dengan mendefinisikan *estimator (model)* dan *params* yang berisi seluruh kombinasi hyperparameter. Selanjutnya dibuat objek *GridSearchCV* dengan parameter *estimator=model, param_grid=params,* cv=10, dan verbose=2. Pengukuran waktu eksekusi dilakukan menggunakan *perf_counter()* sebelum dan sesudah pemanggilan *cv.fit(X_train, y_train)* sehingga diperoleh durasi pelatihan dalam detik. Setelah pencarian selesai, performa model terbaik dievaluasi pada data uji menggunakan *accuracy_score, precision_score, recall_score, dan f1_score*, kemudian model disimpan ke dalam file menggunakan *joblib.dump*. Langkah serupa dengan penyesuaian kelas pencarian juga diterapkan pada *Random Search Cross Validation (*Gambar *4.4* Kode *Random Search CV)* dan *Halving Random Search Cross Validation (Gambar 4.5 Kode Halving Random Search CV)*. Perbedaan implementasi terletak pada penggunaan *param_distributions* untuk mendefinisikan distribusi nilai *hyperparameter*, serta penambahan parameter *random_state=42* guna memastikan reproduksibilitas hasil pencarian acak pada *Random Search Cross Validation* dan *Halving Random Search Cross Validation*.

```python
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import accuracy_score, precision_score, recall_score
from sklearn.neighbors import KNeighborsClassifier
from time import perf_counter
import joblib


model = KNeighborsClassifier()
params = params_dict_knn_1
n_iter = int(params_dict_knn_1_total*0.1)
cv = RandomizedSearchCV(n_iter=n_iter,estimator=model, param_distributions=params, cv=10, verbose=

start = perf_counter()
cv.fit(X_train, y_train)
elapsed = perf_counter() - start
print(f"Elapsed time: {elapsed:.2f} seconds")

print("Best Parameters:", cv.best_params_)
print("Best Accuracy:", cv.best_score_)

acuracy = accuracy_score(y_test, cv.predict(X_test))
precicion = precision_score(y_test, cv.predict(X_test))
recall = recall_score(y_test, cv.predict(X_test))

print("Akurrasi pada data test:", acuracy)
print("Presisi pada data test:", precicion)
print("Recall pada data test:", recall)
joblib.dump(cv, 'knn_random_1.pkl')
```

Gambar 4.4 Kode *Random Search CV*

```python
from sklearn.experimental import enable_halving_search_cv  # noqa
from sklearn.model_selection import HalvingRandomSearchCV
from sklearn.metrics import accuracy_score, precision_score, recall_score
from sklearn.neighbors import KNeighborsClassifier
from time import perf_counter
import joblib


model = KNeighborsClassifier()
params = params_dict_knn_2
cv = HalvingRandomSearchCV(estimator=model, param_distributions=params, cv=10, verbose=2, random_s

start = perf_counter()
cv.fit(X_train, y_train)
elapsed = perf_counter() - start
print(f"Elapsed time: {elapsed:.2f} seconds")

print("Best Parameters:", cv.best_params_)
print("Best Accuracy:", cv.best_score_)

acuracy = accuracy_score(y_test, cv.predict(X_test))
precicion = precision_score(y_test, cv.predict(X_test))
recall = recall_score(y_test, cv.predict(X_test))

print("Akurrasi pada data test:", acuracy)
print("Presisi pada data test:", precicion)
print("Recall pada data test:", recall)
joblib.dump(cv, 'knn_halving_1.pkl')
```

Gambar 4.5 Kode Halving Random Search CV

c. Hasil Pelatihan

Berikut adalah tabel hasil data yang dikumpulkan setelah melakukan pelatihan:

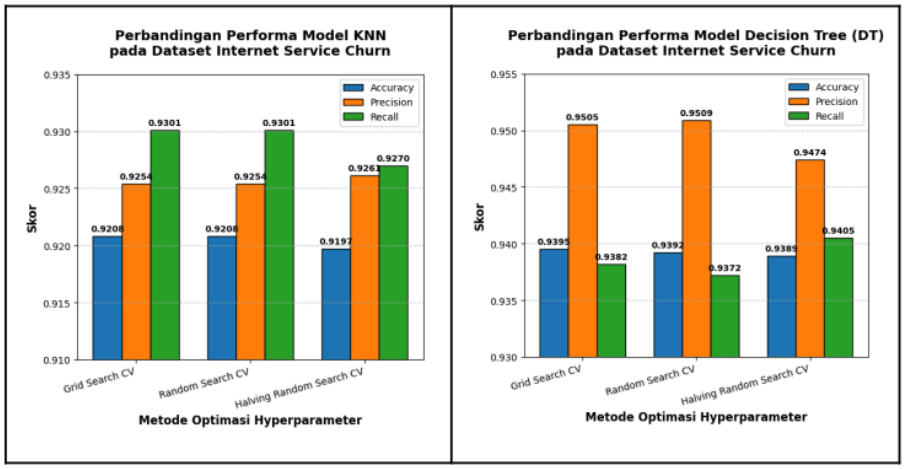
Tabel 4.8 Hasil Pelatihan

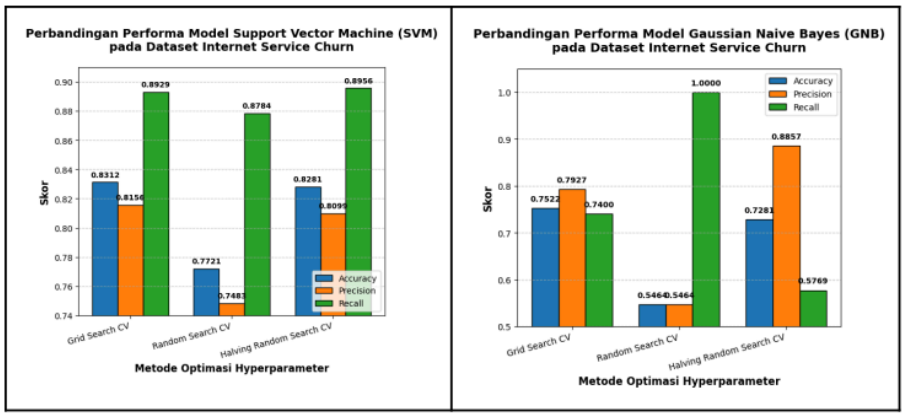
Metode Optimasi	Model	Run ke-	Waktu (detik)	Akurasi	Presisi	f-recall
Grid Search Cross Validation	KNN	1	1540.61	0.9208	0.9254	0.9301
		2	1556.37			
		3	1577.01			
	DT	1	487.07	0.9395	0.9505	0.9382
		2	484.42			
		3	483.17			
	SVM	1	7319.16	0.8312	0.8156	0.8929
		2	6892.50			
		3	6501.26			
	GNB	1	2.22	0.7522	0.7927	0.74
		2	2.91			
		3	2.13			
Random Search Cross Validation	KNN	1	301.37	0.9208	0.9254	0.9301
		2	299.22			
		3	309.18			
	DT	1	44.67	0.9392	0.9509	0.9372
		2	43.86			
		3	44.56			
	SVM	1	1121.51	0.7721	0.74832	0.8784
		2	1261.76			
		3	1014.56			
	GNB	1	0.87	0.5464	0.5464	1.0
		2	0.35			

		3	0.34			
Halving Random Search Cross Validation	KNN	1	419.95	0.9197	0.9261	0.9270
		2	412.06			
		3	411.08			
	DT	1	182.32	0.9389	0.9474	0.9405
		2	187.44			
		3	186.31			
	SVM	1	1804.39	0.8281	0.8099	0.8956
		2	1800.82			
		3	1794.14			
	GNB	1	204.32	0.7281	0.8857	0.5769
		2	200.98			
		3	206.03			

B. Pembahasan

1. Perbandingan Akurasi, Presisi, F-recall

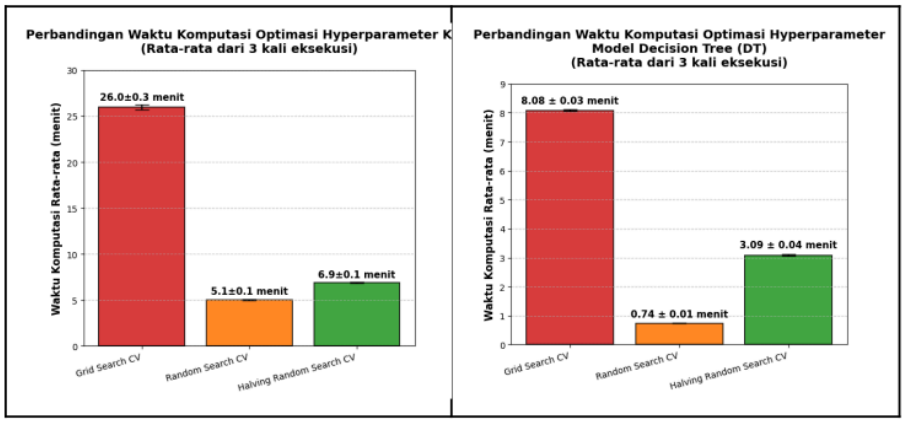


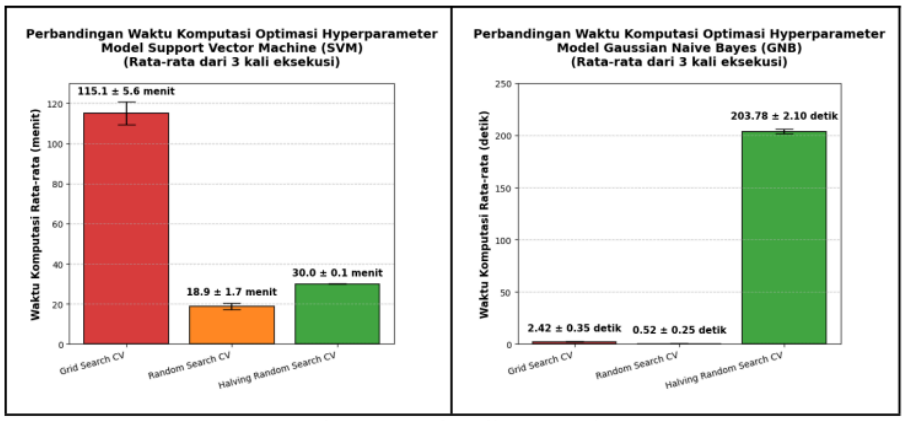


Gambar 4.6 Perbandingan Akurasi, Presisi, F-Recall

Berdasarkan Gambar 4.6 Perbandingan Akurasi, Presisi, F-Recall *Halving Random Search Cross Validation* secara konsisten menghasilkan performa yang sangat kompetitif terhadap *Grid Search CV,* dengan selisih akurasi rata-rata di bawah 0,5 % pada model KNN (0,9197 vs 0,9208), *Decision Tree* (0,9389 vs 0,9395), dan SVM (0,8281 vs 0,8312). Sementara itu, Random Search Cross Validation mampu mengimbangi performa Grid Search CV pada model KNN dan Decision Tree , namun kurang stabil pada model SVM dan Gaussian Naive Bayes, di mana akurasinya turun signifikan SVM 0,7721 dan GNB 0,5464 akibat terjebak pada konfigurasi hyperparameter yang buruk. Sebaliknya, Halving Random Search CV berhasil menghindari jebakan tersebut dan tetap memberikan hasil yang mendekati atau bahkan lebih baik daripada Grid Search pada semua model, terutama pada GNB (akurasi 0,7281 vs 0,7522).

2. Perbandingan Waktu





Gambar 4.7 Perbandingan Waktu

Hasil pengukuran waktu komputasi rata-rata dari tiga kali pengulangan menunjukkan bahwa *Halving Random Search Cross Validation* secara signifikan lebih efisien daripada *Grid Search CV* pada hampir semua model. Penghematan waktu paling dramatis terjadi pada model SVM dari rata-rata 115 menit menjadi 30 menit, Decision Tree dari 8,1 menit menjadi 3,1 menit, dan KNN dari 26 menit menjadi 6,9 menit. Namun pada model *Gaussian Naive Bayes* justru *Halving Random Search* menjadi paling lambat karena *overhead successive halving* pada ruang *hyperparameter* yang kecil. Sedangkan memang *Random Search CV* menjadi yang tercepat pada semua model, namun hasil akurasi model yang dihasilkan tidak optimal.

3. Analisis Hasil dan Evaluasi terhadap Tujuan Penelitian.

Hasil penelitian yang dilakukan pada dataset *Internet Service Churn* menunjukkan bahwa *Halving Random Search Cross Validation* berhasil memenuhi ekspektasi yang ditetapkan pada *Business Understanding*, yaitu tercapainya model prediksi yang akurat sekaligus efisien dari segi waktu komputasi. Metode *Halving* mampu menghasilkan performa metrik (akurasi, presisi, dan recall) yang sangat kompetitif terhadap *Grid Search CV* dengan selisih akurasi kurang dari 0,5 %, serta menghemat waktu komputasi secara drastis 62–74 %.

BAB V
PENUTUP

Kesimpulan

Berdasarkan seluruh hasil eksperimen yang telah dilakukan pada dataset *Internet Service Churn*, dapat disimpulkan bahwa *Halving Random Search Cross Validation* merupakan metode optimasi hyperparameter yang paling seimbang untuk kasus ini. Metode ini berhasil mencapai performa akurasi, presisi, dan recall yang sangat kompetitif dibandingkan *Grid Search*, sekaligus memangkas waktu komputasi secara drastis hingga 62–74 %. Pada model dengan ruang *hyperparameter* kecil seperti *Gaussian Naive Bayes*, *Halving* memang menjadi lebih lambat karena *overhead successive halving*, namun tetap memberikan hasil yang jauh lebih stabil daripada *Random Search*.

Selain itu, hasil dari penelitian ini diharapkan dapat menjadi landasan bagi studi lanjutan di bidang optimasi hyperparameter, serta mendorong pengembangan dan perbandingan metode *Halving Random Search Cross Validation* dengan pendekatan lain seperti *Particle Swarm Optimization*, *Bayesian Optimization*, atau algoritma pencarian lainnya.

Alief Cahyo Utomo

ORIGINALITY REPORT

13%
SIMILARITY INDEX

PRIMARY SOURCES

1	prosiding-senada.upnjatim.ac.id Internet	64 words — 1%
2	repository.unpkediri.ac.id Internet	46 words — 1%
3	docplayer.info Internet	41 words — 1%
4	Nabil M. AbdelAziz, Mostafa Bekheet, Ahmad Salah, Nissreen El-Saber, Wafaa T. AbdelMoneim. "A Comprehensive Evaluation of Machine Learning and Deep Learning Models for Churn Prediction", Information, 2025 Crossref	37 words — 1%
5	jurnal.itg.ac.id Internet	34 words — 1%
6	kc.umn.ac.id Internet	33 words — 1%
7	360digitmg.com Internet	24 words — < 1%
8	jurnal.pnj.ac.id Internet	24 words — < 1%
9	www.liputan6.com Internet	24 words — < 1%

10	123dok.com Internet	23 words — < 1%
11	Nur Pratama, Aswan Supriyadi Sunge, Eko Budiarto. "Penerapan Model MobileNetV2 Untuk Prediksi Tingkat Roasting Biji Kopi Berbasis Gambar Pada Bot Telegram", RIGGS: Journal of Artificial Intelligence and Digital Business, 2025 Crossref	23 words — < 1%
12	jtiik.ub.ac.id Internet	23 words — < 1%
13	Taufik Hidayat, Irwan Sembiring, Hindriyanto Dwi Purnomo, Ade Iriani. "Prediction of Stunting Prevalence in Toddlers Using Support Vector Machine Algorithm and Synthetic Minority Oversampling Technique (SMOTE)", Jurnal Pekommas, 2025 Crossref	21 words — < 1%
14	repo-dosen.ulm.ac.id Internet	21 words — < 1%
15	proceeding.unpkediri.ac.id Internet	20 words — < 1%
16	www.coursehero.com Internet	20 words — < 1%
17	jurnal.stkippgritulungagung.ac.id Internet	18 words — < 1%
18	Larisa Oriana, Anisa Dwi Sanggar Wati, Anggi Putri Ramahdani, Natasya Nurul Safira, Edi Ismanto. "Peningkatan Kinerja Model Random Forest untuk	17 words — < 1%

Deteksi Kecurangan Kartu Kredit Menggunakan RandomizedSearchCV", JURNAL FASILKOM, 2025
Crossref

19	journal.aisyahuniversity.ac.id Internet	17 words — < 1%
20	essay.utwente.nl Internet	16 words — < 1%
21	eprintslib.ummgl.ac.id Internet	13 words — < 1%
22	journal.universitasmulia.ac.id Internet	13 words — < 1%
23	repository.usd.ac.id Internet	12 words — < 1%
24	Wedad Alawad, Mohamed Zohdy, Debatosh Debnath. "Tuning Hyperparameters of Decision Tree Classifiers Using Computationally Efficient Schemes", 2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), 2018 Crossref	11 words — < 1%
25	www.e3s-conferences.org Internet	11 words — < 1%
26	vufind.katalog.k.utb.cz Internet	10 words — < 1%
27	Neny Rosmawarni, Thoyyibah. T, Imam Ahmad, Eka Ardhianto, Dede Handayani, Willis Puspita Sari. "Hyperparameter Tuning On Machine Learning Transformers For Mood Classification In Indonesian Music", 2023 International Conference on Informatics, Multimedia, Cyber and Informations System (ICIMCIS), 2023	9 words — < 1%

Crossref

28	adoc.pub Internet	9 words — < 1%
29	eprints.radenfatah.ac.id Internet	9 words — < 1%
30	juandomingofarnos.wordpress.com Internet	9 words — < 1%
31	www.linuxquestions.org Internet	9 words — < 1%
32	www.preprints.org Internet	9 words — < 1%
33	Alfataniah Nur Fajrina, Zein Hanni Pradana, Sevia Indah Purnama, Shinta Romadhona. "Penerapan Arsitektur EfficientNet-B0 Pada Klasifikasi Leukimia Tipe Acute Lymphoblastik Leukimia", Jurnal Riset Rekayasa Elektro, 2024 Crossref	8 words — < 1%
34	eprints.walisongo.ac.id Internet	8 words — < 1%
35	jim.unisma.ac.id Internet	8 words — < 1%
36	journal.aptii.or.id Internet	8 words — < 1%
37	library.universitaspertamina.ac.id Internet	8 words — < 1%
38	repository.unja.ac.id Internet	8 words — < 1%

39	text-id.123dok.com Internet	8 words — < 1%
40	Luly Hermawan, Sovian Aritonang, Novky Asmoro. "Pemanfaatan Kecerdasan Buatan (AI) Untuk Pemeliharaan Alutsista Pesawat Tempur Dalam Meningkatkan Kesiapan Operasional TNI AU", MARAS: Jurnal Penelitian Multidisiplin, 2024 Crossref	6 words — < 1%
41	Nur Zelina, Afiyati Afiyati. "Analisis Sentimen Ulasan Pengguna Aplikasi M-Banking Menggunakan Algoritma Support Vector Machine dan Decision Tree", Jurnal Linguistik Komputasional (JLK), 2024 Crossref	6 words — < 1%
42	repositori.uin-alauddin.ac.id Internet	6 words — < 1%

EXCLUDE QUOTES	ON	EXCLUDE SOURCES	OFF
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	OFF