JURNAL FASILKOM
Volume 15 No. 2 | Agustus 2025: 229-237

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Peningkatan Kinerja Model *Random Forest* untuk Deteksi Kecurangan Kartu Kredit Menggunakan *RandomizedSearchCV*

Larisa Oriana[1], Anisa Dwi Sanggar Wati[2], Anggi Putri Ramahdani[3], Natasya Nurul Safira[4], Edi Ismanto[5]
[1, 2, 3, 4, 5] Teknik Informatika, Fakultas Ilmu Komputer, Universitas Muhammadiyah Riau
[1]230401164@umri.ac.id, [2]230401161@umri.ac.id, [3]230401166@umri.ac.id, [4]230401143@umri.ac.id,
[5]edi.ismanto@umri.ac.id

Abstract

The rapid growth of credit card transactions in Indonesia has been accompanied by a significant increase in fraudulent cases, necessitating more reliable detection systems. This study develops a fraud detection model by optimizing the Random Forest algorithm using RandomizedSearchCV technique. The dataset comprises 690 transactions with 16 features, including 14 predictor variables and class labels (0 for legitimate transactions, 1 for fraudulent). The research methodology includes exploratory data analysis (EDA) to understand dataset characteristics, feature standardization, data splitting (80% training, 20% testing), and implementation of Random Forest models in two versions: without tuning and with parameter optimization. The results demonstrate improved model performance after optimization, with testing accuracy reaching 90.58% (up from 89.86%). The optimized model successfully increased fraud detection precision from 0.94 to 0.98 while reducing false positives from 3 to 1 case. However, a slight decrease in fraud class recall from 0.82 to 0.80 was observed, indicating the need for balancing between accuracy and detection sensitivity. These findings prove that hyperparameter tuning with RandomizedSearchCV can enhance the performance of Random Forest models in credit card fraud detection, particularly in minimizing classification errors.

Keywords: fraud detection, credit card, random forest, hyperparameter optimization, randomizedsearchcv

Abstrak

Pertumbuhan transaksi kartu kredit di Indonesia yang pesat diiringi dengan peningkatan kasus penipuan membutuhkan sistem deteksi yang handal. Penelitian ini mengembangkan model deteksi penipuan dengan mengoptimalkan algoritma *Random Forest* melalui teknik *RandomizedSearchCV*. Dataset mencakup 690 transaksi dengan 16 fitur, termasuk 14 variabel prediktor dan label kelas (0 untuk transaksi normal, 1 untuk *fraud*). Tahapan penelitian meliputi eksplorasi data (EDA) untuk memahami karakteristik dataset, standarisasi fitur, pembagian data (80% pelatihan, 20% pengujian), serta implementasi model *Random Forest* dalam dua versi: tanpa tuning dan dengan optimasi parameter. Hasil penelitian menunjukkan peningkatan performa model setelah optimasi, dengan akurasi mencapai 90,58% pada data uji (naik dari 89,86%). Model yang dioptimasi juga berhasil meningkatkan *precision* deteksi *fraud* dari 0,94 menjadi 0,98 dan mengurangi kesalahan klasifikasi transaksi normal (*false positive*) dari 3 menjadi 1 kasus. Namun, terdapat sedikit penurunan *recall* kelas *fraud* dari 0,82 menjadi 0,80, yang mengindikasikan perlunya penyeimbangan antara akurasi dan sensitivitas deteksi. Temuan ini membuktikan bahwa *tuning hyperparameter* dengan *RandomizedSearchCV* dapat meningkatkan kinerja model Random Forest dalam mendeteksi penipuan kartu kredit, khususnya dalam meminimalkan kesalahan klasifikasi.

Kata kunci: deteksi penipuan, kartu kredit, *random forest*, optimasi *hyperparameter*, *randomizedsearchcv*

1. Pendahuluan

Transaksi menggunakan kartu kredit di Indoensia mengalami pertumbuhan signifikan, hal ini sejalan dengan digitalisasi dan pergeseran perilaku masyarakat ke pembayaran non-tunai. Kartu kredit adalah kartu yang bisa digunakan sebagai alat pembayaran, yang pelunasan tagihannya dapat dilakukan secara bartahap atau dicicil, kepada pemegang kartu kredit ditentukan jumlah batas kreditnya [1]. Bertambah banyaknya penggunaan kartu kredit, maka tindak kriminal seperti penipuan kartu kredit pun meningkat. Maka dari itu untuk mengatasi tindakan tersebut harus dilakukan deteksi penipuan untuk mencegah aktivitas penipuan agar tidak membuat kerugian besar [2].

Untuk mengatasi masalah tersebut, salah satu solusinya dengan penerapan *machine learning* yang akan menghasilkan pendeteksian yang lebih cepat dan akurat. *Machine learning* adalah bagian dari kecerdasan buatan atau *artificial intelligence* yang merupakan program komputer yang mempunyai algoritma untuk mempelajari data sehingga dapat berfikir dan brertindak seperti manusia [3].

Beberapa penelitian terdahulu telah menggunakan beberapa metode *machine learning*, seperti penggunaan *Logistic Regression, Random Forest*, *Decision Trees* dan *Deep Learning*. Namun diantara semua itu, *Random Forest* khususnya telah menunjukkan keunggulan dalam menangani kerumitan data yang besar dan bervariasi serta memberikan prediksi yang akurat [4]. *Random forest* memiliki kelebihan yaitu diantaranya dapat

P-ISSN : 2089-3353
E-ISSN : 2808-9162

memberikan hasil klasifikasi yang baik disertai dengan hasil residu/error yang lebih rendah, serta dapat mengatasi data training yang berukuran sangat besar secara efisien [2].

Salah satu penelitian terdahulu menunjukkan bahwa *Random Forest* mampu mengklasifikasikan data transaksi kartu kredit dengan sangat baik, yaitu menghasilkan akurasi 97,27%, sensitivitas 98,79%, presisi 97,97%, F-measure 98,38%, dan AUC 94,06% [2]. Meskipun hasilnya terbilang baik, penelitian ini belum menerapkan optimasi *hyperparameter*, sehingga ada potensi beberapa kinerja seharusnya bisa ditingkatkan, tetapi belum dimanfaatkan.

Selain itu, penelitian lain lebih fokus pada masalah ketidakseimbangan data pada deteksi *fraud* menggunakan SMOTE-Tomek. Hasil penelitiannya menunjukkan bahwa penggunaan SMOTE-Tomek pada model *Random Forest* secara signifikan meningkatkan kinerja model deteksi penipuan, terutama dalam mengenali transaksi penipuan yang jarang terjadi [5]. Namun masih sama seperti sebelumnya, penelitian ini belum menerapkan optimasi *hyperparameter*.

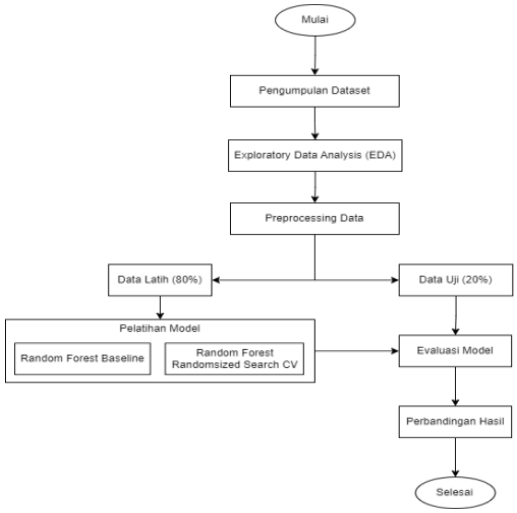
Meskipun sudah dianggap cukup baik, namun penggunaan *Random Forest* akan lebih baik lagi jika dilakukan *tuning hyperparameter* secara otomatis menggunakan *RandomizedSearchCV*. *Randomized Search* adalah teknik di mana kombinasi acak dari parameter penting dan parameter tidak penting digunakan untuk menemukan solusi terbaik untuk model yang sedang dibangun [6].

Berdasarkan penelitian-penelitian terdahulu terdapat celah penelitian, yaitu kurangnya penelitian yang mengombinasikan *Random Forest* dengan optimasi *hyperparameter*. Padahal hal ini penting karena dalam kasus *fraud detection*, salah mendeteksi transaksi *fraud* sebagai transaksi normal (*false negative*) dapat menimbulkan kerugian yang jauh lebih besar dibanding *false positive.*

Dengan demikian, penelitian ini bertujuan untuk menerapkan *RandomizedSearchCV* dalam proses tuning *hyperparameter* pada model *Random Forest* dan membandingkannya hasilnya jika model *Random Forest* tetap digunakan tanpa optimasi. Penelitian ini akan meninjau sejauh mana optimasi tersebut mampu meningkatkan performa model dari berbagai sisi dibandingkan dengan model baseline tanpa *tuning*, serta menganalisis dampaknya dalam mengurangi *false negative*. Adanya penelitian ini diharapkan tidak hanya memberikan kontribusi ilmiah dengan memperkuat literature mengenai deteksi *fraud* menggunakan *Random Forest,* tetapi juga memberikan manfaat praktis bagi sistem deteksi *fraud* di industri perbankan.

2. Metode Penelitian

Metodologi penelitian yang digunakan dalam penelitian ini mengikuti tahapan yang telah sesuai dengan standar *machine learning* seperti yang dapat dilihat pada gambar 1.



Gambar 1. Tahapan Penelitian

Penelitian ini menggunakan pendekatan dengan tahapan sistematis mulai dari pengumpulan data, *preprocessing*, pelatihan data dengan model *Random Forest* tanpa optimasi dan model *Random Forest* yang disertai optimasi *RandomizedSearchCV,* hingga evaluasi performa model dan perbandingan hasilnya.

2.1. Pengumpulan Data

Pengumpulan data didefinisikan sebagai proses mengumpulkan, mengukur, dan merekam data yang akurat dan relevan dari berbagai sumber untuk tujuan analisis dan pengambilan keputusan. Tujuan utama pengumpulan data adalah untuk mengumpulkan informasi dan data yang akurat dan relevan untuk membuat keputusan bisnis yang tepat, meningkatkan kualitas produk atau jasa, meningkatkan efisiensi operasional, dan meningkatkan kepuasan pelanggan [7]. Dalam penelitian ini, teknik pengumpulan data yang digunakan adalah studi literatur untuk mempelajari referensi dari berbagai sumber yang terpercaya. Selain itu, juga digunakan teknik pengumpulan data studi dokumentasi yang dilakukan dengan mengambil data riwayat transaksi kartu kredit yang terdapat pada sebuah dataset.

Dataset yang digunakan tersimpan dalam file berformat CSV dengan nama *CreditCard.csv.* File tersebut memuat 690 baris data, di mana setiap barisnya merepresentasikan satu transaksi kartu kredit yang tercatat melalui 16 variabel berbeda. Setiap transaksi akan diidentifikasi berdasarkan *CustomerID* yang kemudian dianalisis untuk menentukan apakah pelanggan tersebut menunjukkan indikasi keterlibatan dalam aktivitas kecurangan.

P-ISSN : 2089-3353
E-ISSN : 2808-9162

2.2. Exploratory Data Analysis (EDA)

Exploratory Data Analysis diperkenalkan oleh seorang matematikawan bernama John Tukey yang awalnya mengembangkan EDA pada tahun 1970 dan hingga saat ini masih terus digunakan. EDA adalah proses pemeriksaan atau pemahaman data dan penggalian wawasan atau karakteristik utama dari data [8]. EDA umumnya diklasifikasikan menjadi dua metode, yaitu analisis grafis dan analisis *non grafis*. Pada penelitian ini, digunakan dua metode EDA tersebut. Analisis grafis menampilkan data dalam visual, sedangkan analisis non grafis menampilkan statistic deskriptif seperti nilai mean dan median.

Ada beberapa upaya yang bisa dilakukan untuk mengeksplorasi data credit card yang dilakukan pada penelitian ini. Pertama, menampilkan data awal sebagai bentuk validasi untuk memastikan bahwa file diproses dalam tahap pengkodean adalah data yang sesuai dengan dataset awal, yaitu file *CreditCard.cs*v. Proses ini bertujuan untuk meminimalisir kesalahan pengguna dalam mengunggah file dataset.

Kedua, menampilkan informasi dataset, seperti perhitungan terkait jumlah baris dan kolom yang akan dianalisa, disertai dengan identifikasi tipe data dari setiap kolom. Dari proses EDA yang telah dilakukan, diketahui bahwa dataset ini memiliki 690 baris data dan 16 kolom, di mana 13 kolom diantaranya bertipe *integer* dan 3 kolom lainnya bertipe *float*.

Ketiga, memeriksa *missing values* sebagai langkah awal untuk proses pembersihan data. Dari proses ini diketahui bahwa semua baris dan kolom pada dataset yang digunakan untuk penelitian ini memiliki nilai lengkap tanpa *missing value.*

Keempat, berdasarkan analisa yang telah dilakukan pada informasi dataset, langkah selanjutnya adalah memisahkan fitur dan target, serta menghapus fitur yang tidak relevan agar data lebih siap untuk pelatihan model.

Terakhir, visualisasi data dalam berbagai bentuk grafis. Pada penelitian ini divisualisasikan beberapa diagram untuk mempermudah memahami hubungan antar fitur pada dataset. Salah satu diantaranya diagram yang menampilkan distribusi kelas target.

2.3. Preprocessing Data

Preprocessing Data adalah serangkaian langkah yang dilakukan untuk mempersiapkan data mentah menjadi format yang lebih baik untuk dianalisis atau digunakan oleh model pembelajaran mesin. Tujuan utama dari data *preprocessing* adalah meningkatkan kualitas data, mengatasi masalah yang mungkin ada, dan memastikan bahwa data siap digunakan dalam proses analisis atau pelatihan model [7].

Ada dua tahapan yang dilakukan untuk memenuhi *preprocessing* data. Pertama standardisasi, suatu teknik yang digunakan dalam analisis data untuk mengubah nilai-nilai variabel agar memenuhi asumsi atau persyaratan tertentu [9]. Standarisasi data yang dilakukan pada penelitian ini menggunakan library berupa: *Import sklearn.preprocessing import StandardScaler.* Kedua, membagi dataset menjadi dua bagian, yaitu 80% untuk data latih dan 20% untuk data uji. Proses split data ini menggunakan *library* berupa: *from sklearn.model_selection import train_test_split.*

2.4. Pelatihan dan Pengujian Model

Pelatihan merupakan tahap di mana model pembelajaran mesin dibentuk dan dikembangkan melalui data latih yang ada sebagai sumber pembelajarannya. Tujuan dari pelatihan adalah untuk mengoptimalkan model sehingga dapat membuat prediksi yang akurat pada data baru [10]. Semakin banyak jumlah data latihnya, maka akan semakin baik pula hasil model yang terbentuk. Meskipun demikian, pelatihan juga tidak boleh berlebihan karena bisa menyebabkan *overfitting*, yaitu kondisi di mana model mampu mempelajari data pelatihan dengan sangat baik, tetapi tidak bisa memprediksi dengan akurat.

Penelitian ini menggunakan pelatihan model *Random Forest* yang disertai perbandingan antara *Random Forest Baseline* (tanpa disertai optimasi sama sekali) dan *Random Forest* yang menggunakan optimasi *RandomizedSearchcv.*

Random Forest (RF) merupakan salah satu algoritma *Machine Learning* yang bekerja dengan mengombinasikan sejumlah algoritma *Decision Tree* dalam pengambilan keputusannya [11]. *Random Forest* mempunyai dua parameter utama, yaitu parameter m yang merupakan presentasi dari jumlah pohon yang akan dipakai dan parameter k yang merupakan representasi dari banyaknya fitur maksimal yang dipertimbangkan ketika proses percabangan pada pohon [12].

Dalam *Random Forest*, pemilihan fitur terbaik menggunakan Indeks Gini dan Gini Split. Indeks Gini untuk memilih fitur di setiap simpul dari pohon keputusan. Rumus indeks Gini seperti rumus 1.

$$Gini(Si) = 1 - \sum_{i=0}^{c-1} pi^2 \quad (1)$$

dengan Indeks Gini untuk set data Si *(Gini (S_i))*, perhitungannya melibatkan 1−…. Ini berarti semakin tinggi jumlah kuadrat probabilitas kelas (pi^2), semakin rendah nilai Gini, menunjukkan kemurnian data yang lebih tinggi. Bagian $\sum_{i=0}^{c-1} pi^2$ adalah total jumlah kuadrat probabilitas (pi) setiap kelas dalam Si, di mana c adalah jumlah kelas unik dan pi adalah proporsi kelas ke-i.

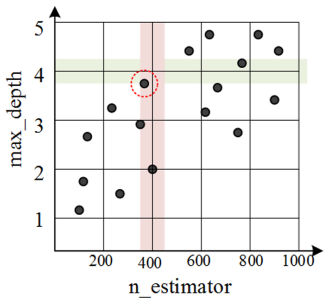
Kualitas split pada fitur k ke dalam subset Si merupakan jumlah sampel milik kelas C_i, kemudian dihitung sebagai jumlah pertimbangan indikasi Gini dari subset yang dihasilkan [13]. Rumus lengkapnya seperti pada rumus 2.

$$\text{Gini split} = \sum_{i=0}^{c-1} \left(\frac{ni}{n}\right) \text{Gini (Si)} \quad (2)$$

P-ISSN : 2089-3353
E-ISSN : 2808-9162

dengan variabel c menunjukkan jumlah simpul (*node*) anak yang dihasilkan dari proses pemisahan. Kemudian, ini merupakan jumlah sampel yang berada dalam simpul anak ke-i, sementara n adalah jumlah total sampel sebelum pemisahan. *Random Forest* memberikan prediksi yang lebih baik dibandingkan dengan model tunggal dan kurang cenderung *overfit* selama pelatihan [14].

Selanjutnya dilakukan optimasi model dengan menggunakan *RandomizedSearchCV*. Optimasi *RandomizedSearchCV* merupakan metode alternatif yang digunakan untuk menemukan parameter terbaik dalam suatu model, sehingga model tersebut dapat memprediksi data secara akurat [15].



Gambar 2. Mekanisme Kerja RandomSearchCV

Pada *Random Search* dicari nilai yang optimal sepanjang nilai parameter yang telah ditentukan dengan jelas antara batas bawah dan batas atasnya [11]. Gambar 2 menggambarkan cara kerja *Randomized Search*, di mana parameter n_estimators dan *max_deepth* pada *Random Forest* telah ditentukan dalam rentang tertentu, lalu dipilih secara acak hingga ditemukan kombinasi yang terbaik.

2.5 Evaluasi Model

Evaluasi pada dasar pembelajaran mesin adalah proses mengukur kinerja model pembelajaran mesin pada data yang tidak digunakan dalam pelatihan model. Evaluasi ini penting untuk memastikan bahwa model dapat melakukan prediksi yang akurat pada data baru dan tidak hanya mengingat data pelatihan [16]. Evaluasi model ini dapat dilihat dari *Confusion Matrix* yang terdiri dari empat sel utama yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).

Dengan menggunakan empat nilai ini, kita dapat menghitung berbagai metrik evaluasi seperti *Accuracy, Precision, Recall,* dan *F1 Score* [17]. Akurasi digunakan ketika dataset seimbang, di mana tingkat akurasi yang tinggi menunjukkan model yang baik. Dengan demikian, rumus untuk menghitung akurasi seperti pada rumus 3:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

dengan perhitungan rasio antara jumlah prediksi benar (*True Positive* dan *True Negative*) dengan total keseluruhan prediksi (*True Positive, True Negative, False Positive,* dan *False Negative*). Rumus ini mengukur proporsi keseluruhan prediksi model yang tepat dari seluruh data yang ada.

Presisi dan *recall* digunakan untuk memahami bagaimana model menangani kesalahan positif dan negatif. Sementara itu, *F1 Score* digunakan ketika ada *trade-off* antara *precision* dan *recall* dan tidak ada yang lebih penting diantara keduanya, karena semua ini sama pentingnya [18]. Untuk rumus presisi seperti pada rumus 4, recall pada rumus 5, dan *F1Score* pada rumus 6:

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

dengan perhitungan rasio antara *True Positive* dan jumlah seluruh prediksi positif (*True Positive* ditambah *False Positive*), untuk mengukur proporsi prediksi positif yang benar di antara semua bagian yang diklaim sebagai positif.

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

dengan perhitungan rasio antara *True Positive* dan jumlah seluruh kasus positif aktual (*True Positive* ditambah *False Negative*), untuk mengukur kasus positif yang sebenarnya berhasil diidentifikasi dari semua kasus positif yang seharusnya terdeteksi.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (6)$$

dengan *F1-Score* yang dihitung melalui proses di mana nilai *Precision* dan *Recall* terlebih dahulu dikalikan, kemudian dijumlahkan, dan hasil perkalian tersebut dibagi dengan hasil penjumlahannya, lalu keseluruhan hasilnya dikalikan dua.

3. Hasil dan Pembahasan

3.1. Eksplorasi Dataset

Pada penelitian ini, dataset yang digunakan terdiri dari 690 sampel transaksi kartu kredit yang memiliki 16 kolom yang meliputi *CustomerID*, 14 fitur prediktor (A1-A14), dan satu variabel target *(class).* Berdasarkan hasil eksplorasi data awal, dataset tidak memiliki *missing values* pada seluruh kolom, yang menunjukkan kualitas dataset yang baik dan siap untuk dilakukan tahap selanjutnya yaitu tahap *preprocessing.* Struktur dataset mempunyai variasi tipe data yang terdiri dari 13 kolom bertipe *integer* (int64), dan 3 kolom bertipe *float* (64).

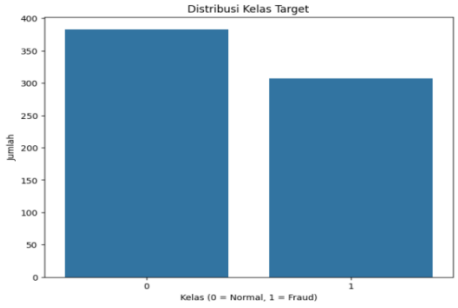
Setelah menghapus kolom *CustomerID* dan *Class* dari fitur prediktor, akan diperoleh matriks X dengan diensi 690x14 den *vector* target Y dengan 690 label kelas. Hasil menunjukkan fitur X berisi nilai numerik yang beragam dengan rentang nilai yang berbeda-beda untuk tiap antar kolom, dan target Y berisi label *biner* (0 dan 1).

3.2. Distribusi Kelas Target

Pada distribusi kelas target menunjukkan ketidakseimbangan data dengan kelas 0 (transaksi

P-ISSN : 2089-3353
E-ISSN : 2808-9162

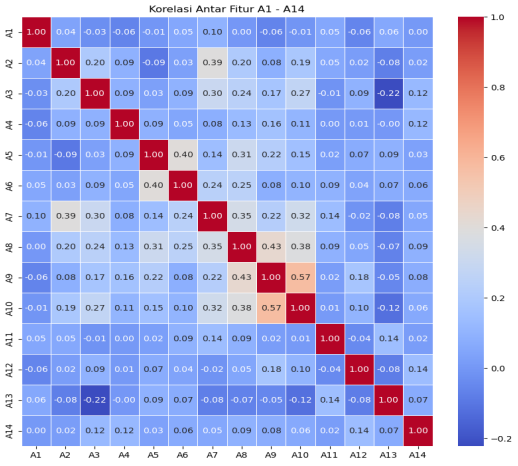
normal) sebanyak 383 sampel (55,5%) dan kelas 1 (penipuan) sebanyak 307 sampel (44,5%). Walaupun terdapat ketidaseimbangan pada dataset ini, rasio perbandingan antara kelas 0 dan kelas 1 masih termasuk kedalam batas yang dapat diterima untuk pemodelan data yang tak memerlukan teknik atau metode sampling tambahan seperti *SMOTE.*



Gambar 3. Ketidakseimbangan antara 0 (normal) dan 1 (*fraud*)

Visualisasi pada Gambar 3 secara jelas memperlihatkan perbandingan jumlah sampel antara transaksi normal (kelas 0) dan transaksi *fraud* (kelas 1). Meskipun terdapat perbedaan jumlah, gap yang tidak terlalu signifikan (hanya sekitar 11%) menunjukkan bahwa dataset ini masih memenuhi syarat untuk pemodelan langsung tanpa perlu teknik penyeimbangan khusus. Dalam konteks deteksi penipuan, rasio seperti ini justru lebih merepresentasikan kondisi *real* di lapangan dimana transaksi normal memang cenderung lebih banyak namun tidak mendominasi secara ekstrim. Hal ini menjadi keunggulan dataset yang digunakan dalam penelitian ini, karena mampu mempertahankan karakteristik asli data sekaligus meminimalisir risiko bias pada model yang akan dibangun.

3.3. Analisis Korelasi Antar Fitur



Gambar 4. *Heatmap* Korelasi Antar Fitur A1-A14

Sebelum melakukan tahap pemodelan, penting untuk memahami hubungan antar variabel prediktor dalam dataset. Salah satu cara yang umum digunakan adalah dengan melihat korelasi antar fitur melalui visualisasi heatmap. Dengan heatmap, pola hubungan antar fitur dapat diamati secara lebih mudah, terutama untuk mengidentifikasi apakah terdapat fitur yang memiliki keterkaitan yang cukup kuat atau justru saling independen. Analisis ini membantu memastikan bahwa informasi yang dibawa setiap fitur tidak saling tumpang tindih secara berlebihan, sehingga dapat mendukung proses pemilihan fitur yang lebih efisien.

Pada gamba 4 menunjukkan heatmap korelasi dari 14 fitur (A1 hingga A14). Warna yang muncul menunjukkan tingkat hubungan antar fitur, di mana warna merah menandakan korelasi positif yang lebih kuat, sedangkan warna biru menunjukkan korelasi yang rendah atau negatif. Dengan melihat visualisasi ini, peneliti dapat memperoleh gambaran awal mengenai pola keterkaitan antar fitur yang nantinya berpengaruh terhadap kinerja model prediksi.

Heatmap korelasi ini menunjukkan hubungan linear antara 14 fitur *predictor* (A1 hingga A14) dalam data set. Nilai korelasi yang ditampilkanyangber berkisar antara -1 hingga 1, di mana nilai yang berkisamendekati 1 menunjukkan korelasi postif kuat, nilai yang mendekati korelasi nilai -1 menunjukkan korelasi negatif yang kuat, dan nilai yang mendekati 0 menunjukkan tidak ada korelasi linear. Diagonal utama pada gambar bernilai 1, menunjukkan korelasi setiap variabel dengan dirinya sendiri.

Dari *heatmap* korelasi anat fitur, terdapat beberapa fitur korelasi yang cukup tinggi antara fitur A9 dan A10 yang menunjukkan korelasi positif cukup tinggi dengan nilai 0.57. Hal itu menunjukkan bahwa kedua fitur tersebut cenderung berubah secara bersamaan, ketika nilai A9 meningkat, nilai A10 juga cenderung meningkat, begitu juga sebaliknya.

Selain itu, dari heatmap korelasi juga menunjukkan beberapa pasangan fitur lain yang perlu diperhatikan, walau tak seringgi hubungan antara A9 dn A10. Sebaliknya, banyak fitur yang menunjukkan korelasi yang lemah atau hampir tidak berkolerasi sama sekali, hal itu berarti fitur tersebut memberikan informasi yang relatif independen dalam dataset.

3.4. Preprocessing Data

Sebelum pelatihan model, perlu dilakukan proses standardisasi menggunakan z-score *normalization.* Transformasi ini mengubah mengubah seluruh fitur seingga memiliki rata-rata 0 dan diviasi standar 1. Hasilnya menunjukkan array dimensi 690×14. Tabel 1 berikut menampilkan ringkasan hasil standardisasi.

Tabel 1. Standardisasi Data

0.68873723	…	0.03738039
-1.45193254	…	-0.19541334
-1.45193254	…	-0.19541334
-1.45193254	…	-0.19541334
-1.45193254	…	-0.19330052
0.68873723	…	-0.19541334

Tabel di atas menunjukkan nilai hasil transformasi z-*score,* yaitu nilai-nilai standar (*z-score*) dari fitur-fitur asli. Meskipun algoritma *Random Forest* secara

P-ISSN : 2089-3353
E-ISSN : 2808-9162

umum tidak sensitive terhadap perbedaan skala, standardisasi tetap dilakukan sebagai langkah praproses agar tidak ada fitur yang secara tidak sengaja mendominasi karena skala nilai yang jauh lebih besar.

3.5. Pelatihan Model *Random Forest (Baseline)*

Model Random Forest pertama kali dibangun menggunakan parameter *default* terbatas, yaitu *max_depth=5, min_samples_leaf=10,* dan *n_estimators=100.* Ini dilakukan untuk memperoleh *baseline performa* sebagai pembaning sebelum dilakukan tuning *hyperparameter* menggunakan *RandomizedSearchCV*.

Model ini kemudian dilatih pada data *training* dan dievaluasi menggunakan data *testing.* Hasil evaluasi menunjukkan:

Tabel 2. Akurasi Model *Baseline*

Data *Training*	Data *Testing*
88.77%	89.86%

Nilai akurasi ini menunjukkan performa model yang cukup baik dan stabil antara data training dan testing, serta mengindikasikan tidak terjadi *overfitting.* Akan tetapi, dalam kasus klasifikasi *fraud* (yang mengandung ketidakseimbangan kelas), akurasi saja tidak cukup. Oleh karena itu, perlu diperhatikan juga nilai *precision, recall,* dan *f1-score*

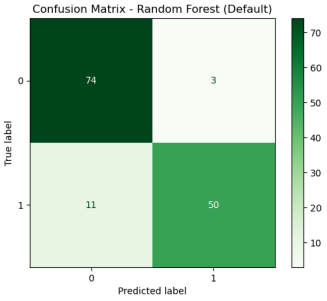
Tabel 3. Data *Training*

	precision	*recall*	*F1-score*	*support*
0	0.89	0.91	0.90	306
1	0.89	0.86	0.87	246
accuracy			0.89	552
Macro avg	0.89	0.88	0.89	552
Weighted avg	0.89	0.89	0.89	552

Tabel 4. Data *Testing*

	precision	*recall*	*F1-score*	*support*
0	0.87	0.96	0.91	77
1	0.94	0.82	0.88	61
accuracy			0.90	138
Macro avg	0.91	0.89	0.90	138
Weighted avg	0.90	0.90	0.90	138

3.5.1. Confusion Matrix-Random Forest (Default)



Gambar 5. *Confusion Matrix-Random Forest (Default)*

Visualisasi *confusion matrix* berdasarkan hasil prediksi pada data *testing.* Dari visualisasi, model berhasil mengklasifikasikan 74 transaksi *non-fraud* dengan benar, namun 3 transaksi *non-fraud* salah diklasifikasikan seagai *fraud (false positive).* Untuk transaksi *fraud,* model berhasil memprediksi 50 kasus dengan benar, namun 11 kasus fraud salah diklasifikasikan sebagai *non-fraud (false negative).*

Hal ini menunjukkan bahwa meskipun akurasi dan *f1-score* tergolong baik, masih ada peluang untuk meningkatkan *recall* pada kelas *fraud*, yang menjadi fokus utama sistem deteksi penipuan. *Recall* yang sedikit lebih rendah bisa terjadi karena model cenderung lebih berhati-hati menandai transaksi sebagai *fraud*, agar tidak terlalu banyak salah menandai transaksi normal (*false positive*). Hal ini sering muncul jika data tidak seimbang, di mana jumlah transaksi normal jauh lebih banyak daripada *fraud*. Akibatnya, sebagian transaksi *fraud* malah lolos dan diklasifikasikan sebagai transaksi normal (*false negative*)

Dalam penerapan nyata, kesalahan seperti ini cukup berisiko. Setiap kasus *fraud* yang terlewat bisa menyebabkan kerugian finansial, penyalahgunaan akun, dan menurunnya kepercayaan pengguna. Sementara itu, *false positive* biasanya hanya menimbulkan ketidaknyamanan yang masih bisa diatasi lewat pemeriksaan manual. Karena itu, recall pada kelas *fraud* sebaiknya dibuat setinggi mungkin, meskipun presisi sedikit turun. Tuning parameter dengan *RandomizedSearchCV* menjadi langkah penting untuk membuat model lebih sensitif mendeteksi *fraud* tanpa membuat terlalu banyak alarm palsu.

3.6. Optimasi Dengan *RandomizedSearchCV*

Setelah membangun model *Random Forest* dengan parameter default, dilakukan peningkatan performa dengan cara tuning hyperparameter menggunakan teknik *RandomizedSearchCV*. Proses *tuning* ini melibatkan pencarian kombinasi parameter terbaik seperti *n_estimators, max_depth, min_samples_split, min_samples_leaf,* dan *bootstrap.* Total 20 kombinasi kandidat diuji menggunakan validasi silang sebanyak 5 kali lipat *(5-fold CV)* dengan skor f1 sebagai metrik evaluasi utama.

Model terbaik yang diperoleh dari proses *tuning* menghasilkan:

Tabel 5. Akurasi Model Setelah *Tuning*

Data Training	Data Testing
90.76 %	90.58%

Nilai akurasi yang cukup tinggi dan seimbang antara data *training* dan *testing* mengindikasikan bahwa model tidak mengalami *overfitting.* Untuk mengevaluasi performa model secara lebih menyeluruh, dilakukan analisis menggunakan *classification report* yang mencakup nilai *precision*,

P-ISSN : 2089-3353
E-ISSN : 2808-9162

recall, dan *f1-score* untuk masing-masing kelas. Hasil evaluasi pada data *testing* menunjukkan bahwa:

Tabel 6. Data *Training*

	precision	*recall*	*F1-score*	*support*
0	0.90	0.93	0.92	306
1	0.91	0.87	0.89	246
accuracy			0.91	552
Macro avg	0.91	0.90	0.91	552
Weighted avg	0.91	0.91	0.91	552

Berdasarkan hasil pengujian pada data training (Tabel 6), model *Random Forest* yang telah dioptimasi menggunakan *RandomizedSearchCV* menunjukkan kinerja yang baik dengan nilai akurasi sebesar 0,91. Pada kelas *non-fraud* (label 0), model memperoleh nilai *precision* sebesar 0,90, *recall* sebesar 0,93, dan *F1-score* sebesar 0,92. Hal ini menandakan bahwa sebagian besar transaksi *non-fraud* dapat dikenali dengan benar oleh model.

Sementara itu, pada kelas *fraud* (label 1), diperoleh precision sebesar 0,91, *recall* sebesar 0,87, dan *F1-score* sebesar 0,89. Nilai *recall* yang sedikit lebih rendah dibandingkan dengan precision menunjukkan masih adanya beberapa transaksi fraud yang tidak teridentifikasi (*false negative*). Secara keseluruhan, nilai macro average dan weighted average sebesar 0,91 mengindikasikan bahwa model memiliki performa yang seimbang dalam membedakan kedua kelas.

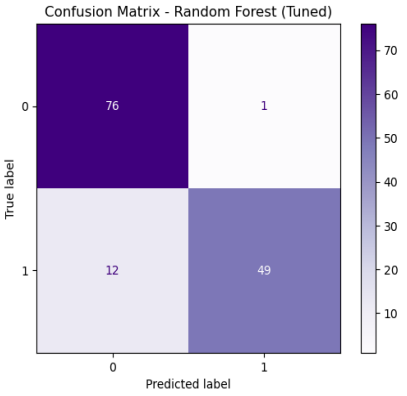
Tabel 7. Data *Testing*

	precision	*recall*	*F1-score*	*support*
0	0.86	0.99	0.92	77
1	0.98	0.80	0.88	61
accuracy			0.91	138
Macro avg	0.92	0.90	0.90	138
Weighted avg	0.92	0.91	0.90	138

Hasil pengujian pada data *testing* (Tabel 7) menunjukkan konsistensi kinerja model dengan akurasi yang meningkat menjadi 0,92. Pada kelas non-fraud (label 0), diperoleh *precision* sebesar 0,86, *recall* sebesar 0,99, dan F1-score sebesar 0,92. Hal ini menunjukkan bahwa model mampu mendeteksi hampir seluruh transaksi non-fraud dengan benar, meskipun masih terdapat beberapa kesalahan klasifikasi berupa false positive.

Sementara itu, pada kelas *fraud* (label 1), nilai *precision* sebesar 0,98, recall sebesar 0,85, dan *F1-score* sebesar 0,91 menegaskan bahwa model cukup andal dalam mengidentifikasi transaksi fraud dengan tingkat kesalahan yang relatif rendah. Namun, recall yang lebih rendah pada kelas *fraud* mengindikasikan masih adanya sebagian transaksi fraud yang tidak terdeteksi dengan baik. Secara keseluruhan, nilai macro average sebesar 0,92 dan *weighted average* sebesar 0,92 menunjukkan bahwa model yang dioptimasi melalui *RandomizedSearchCV* memiliki kinerja yang stabil serta seimbang pada kedua kelas.

3.6.1. Confusion Matrix-Random Forest (Tuned)



Gambar 3. *Confusion Matrix-Random Forest* (*Tuned*)

Confusion matrix pada model *Random Forest* yang telah di-*tuning* menunjukkan detail performa klasifikasi pada data *testing*. *Matrix* ini menampilkan perbandingan antara label aktual *(true label)* pada sumbu vertikal dan label prediksi *(predicted label)* pada sumbu horizontal.

Hasil klasifikasi menunjukkan bahwa model berhasil memprediksi dengan akurasi tinggi. Untuk kelas 0 (negatif), model berhasil mengklasifikasikan 76 sampel dengan benar *(true negative)* dan hanya salah mengklasifikasikan 1 sampel sebagai kelas 1 *(false positive)*. Sementara itu, untuk kelas 1 (positif), model berhasil memprediksi 49 sampel dengan tepat *(true positive)* namun terdapat 12 sampel yang salah diklasifikasikan sebagai kelas 0 *(false negative)*.

3.7. Perbandingan Performa Model Sebelum dan Sesudah *Tuning*

Setelah dilakukan *tuning* menggunakan *RandomizedSearchCV,* model *Random Forest* menunjukkan peningkatan performa. Pada data *training,* akurasi meningkat dari 88,77% menjadi 90,76% dan pada data *testing* naik dari 89,86% menjadi 90,58%. Selisih akurasi antara data *training* dan *testing* juga menjadi lebih kecil pada model yang telah di-*tuning* (-0,18%) dibandingkan model *default* (+1,09%), yang mengindikasikan peningkatan stabilitas dan generalisasi model.

Untuk metrik lainnya, *precision* pada kelas *fraud* meningkat dari 0,94 menjadi 0,98, meskipun *recall* sedikit menurun dari 0,82 menjadi 0,80. Namun, F1-score untuk kelas *fraud* tetap stabil di angka 0,88. Pada kelas normal, *recall* meningkat dari 0,96 menjadi 0,99, menunjukkan model menjadi lebih baik dalam mengenali transaksi yang benar. Secara keseluruhan, rata-rata F1-score tidak berubah signifikan, tetapi distribusi kesalahan menjadi lebih seimbang, terutama dengan berkurangnya *false positive* dari 3 menjadi 1 kasus.

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Dengan tuning hyperparameter menggunakan *RandomizedSearchCV*, Model tidak hanya mengalami peningkatan akurasi dan stabilitas, tetapi juga menunjukkan penurunan jumlah kesalahan klasifikasi pada transaksi normal *(false positive),* yang krusial dalam sistem deteksi penipuan agar tidak merugikan pelanggan

4. Kesimpulan

Penelitian ini menunjukkan bahwa dengan bantuan *RandomizedSearchCV*, algoritma *Random Forest* telah terbukti efektif dalam mengidentifikasi penipuan kartu kredit. Studi ini menunjukkan bahwa akurasi model meningkat dari 89,86% menjadi 90,58% pada data *testing* dan dari 88,77% menjadi 90,76% pada data *training.*

Setelah dioptimasi, model menunjukkan stabilitas yang lebih baik, selisih akurasi antara data *training* dan *testing* berkurang dari 1,09% menjadi -0,18%. Ini menunjukkan bahwa ada risiko *overfitting* yang lebih rendah. *Precision kelas fraud* meningkat signifikan dari 0,94 menjadi 0,98, menunjukkan bahwa model menjadi lebih akurat dalam mengidentifikasi transaksi penipuan. Recall kelas normal meningkat dari 0,96 menjadi 0,99, tetapi hal ini diimbangi dengan peningkatan *precision* kelas normal.

Dengan menggunakan *RandomizedSearchCV* dengan 20 kombinasi dan lima kali *cross-validation* menggunakan skor F1 sebagai evaluasi utama, telah terbukti bahwa teknik ini memiliki kemampuan untuk menemukan kombinasi parameter yang optimal yang menghasilkan kinerja model yang lebih baik dan stabil. Hasil penelitian menunjukkan bahwa perubahan *hyperparameter* adalah langkah penting menuju pembuatan sistem deteksi penipuan yang akurat dan dapat diandalkan.

Studi ini memberikan kontribusi praktis bagi industri perbankan dalam mengembangkan sistem deteksi penipuan yang lebih efisien dan juga memberikan landasan ilmiah untuk pengembangan penelitian selanjutnya dalam bidang *machine learning* untuk deteksi penipuan. Mengimplementasikan *Python* dengan *library scikit-learn* menunjukkan bahwa pengembangan model deteksi penipuan dapat dilakukan dengan perangkat komputasi biasa tanpa memerlukan infrastruktur mahal.

Berdasarkan hasil penelitian ini, terdapat beberapa masukan untuk pengembangan kasus yang sama lebih lanjut. Pertama, membandingkan model machine learning lainnya untuk memperoleh performa yang lebih optimal. Kedua, disarankan untuk menerapkan teknik hyperparameter *tuning* lain seperti *GridSearchCV* untuk meningkatkan akurasi tanpa *overfitting*. Terakhir, gunakan dataset yang lebih besar dan beragam untuk memperkuat kemampuan model. Dengan pengembangan ini, model diharapkan dapat memberikan kontribusi nyata dalam pencegahan penipuan kartu kredit di sektor perbankan.

Daftar Rujukan

[1] D. J. Ardha, "Analisis Kasus Pemalsuan Kartu Kredit Sebagai Bentuk Tindak Pidana Perbankan," *J. Huk. Doctrin.*, vol. 5, no. 2, hal. 245–263, 2020.

[2] T. S. Lestari dan D. A. N. Sirodj, "Klasifikasi Penipuan Transaksi Kartu Kredit Menggunakan Metode Random Forest," *J. Ris. Stat.*, vol. 1, no. 2, hal. 160–167, 2022, doi: 10.29313/jrs.v1i2.525.

[3] R. Armiani dan E. P. Agustini, "Analisa Fraud Pada Transaksi Kartu Kredit Menggunakan Algoritma Random Forest," *J. Teknol. Inf. dan Terap.*, vol. 9, no. 2, hal. 118–126, 2022, doi: 10.25047/jtit.v9i2.297.

[4] U. N. W. Gerry William Mathew Kurniawan, "Pendeteksian Penipuan Menggunakan Pendekatan Metode Klasifikasi Random Forest," in *e-Proceeding of Engineering*, 2025, vol. 12, no. 1, hal. 2216.

[5] M. Ilham, A. Winarno, M. Lutfi, dan A. Indrasetianingsih, "Handling Imbalanced Fraudulent Transaction Data Using SMOTE-Tomek and Random Forest: A Classification Approach," *BEST J. Appl. Electr. Sci. Technol.*, vol. 7, no. 1, hal. 35–38, 2025, doi: 10.36456/best.vol7.no1.10335.

[6] Y. Yennimar, A. Rasid, dan S. Kenedy, "Implementation of Support Vector Machine Algorithm With Hyper-Tuning Randomized Search in Stroke Prediction," *J. Sist. Inf. dan Ilmu Komput. Prima(JUSIKOM PRIMA)*, vol. 6, no. 2, hal. 61–65, 2023, doi: 10.34012/jurnalsisteminformasidanilmukomputer.v6i2.3479.

[7] M. D. Salman *et al.*, "Perbandingan Kinerja Algoritma Clustering K-Means dan K-Medoids dalam Pengelompokan Sekolah di Provinsi Riau Berdasarkan Ketersediaan Sarana dan Prasarana," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 5, no. 3, hal. 797–806, 2025, doi: 10.57152/malcom.v5i3.1950.

[8] J. Kurniawan *et al.*, *Analisis Dan Visualisasi Data*, 1 ed. Bandung: Penerbit Widina Bhakti Persada Bandung, 2023.

[9] P. Rahayu *et al.*, *Buku Ajar Data Mining*, 1 ed., vol. 1, no. January 2024. Jambi: PT. Sonpedia Publishing Indonesia, 2024.

[10] Muttaqin *et al.*, *Data Science dan Pembelajaran Mesin*, 1 ed. Medan: Yayasan Kita Menulis, 2023.

[11] U. Sunarya dan T. Haryanti, "Perbandingan Kinerja Algoritma Optimasi pada Metode Random Forest untuk Deteksi Kegagalan Jantung," *J. Rekayasa Elektr.*, vol. 18, no. 4, hal. 241–247, 2022, doi: 10.17529/jre.v18i4.26981.

[12] M. A. Abubakar, M. Muliadi, A. Farmadi, R. Herteno, dan R. Ramadhani, "Random Forest Dengan Random Search Terhadap Ketidakseimbangan Kelas Pada Prediksi Gagal Jantung," *J. Inform.*, vol. 10, no. 1, hal. 13–18, 2023, doi: 10.31294/inf.v10i1.14531.

[13] Suci Amaliah, M. Nusrang, dan A. Aswi, "Penerapan Metode Random Forest Untuk Klasifikasi Varian Minuman Kopi di Kedai Kopi Konijiwa Bantaeng," *VARIANSI J. Stat. Its Appl. Teach. Res.*, vol. 4, no. 3, hal. 121–127, 2022, doi: 10.35580/variansiunm31.

[14] M. Isangediok dan K. Gajamannage, "Fraud Detection Using Optimized Machine Learning Tools Under Imbalance Classes," in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, hal. 4275–4284. doi: 10.1109/BigData55660.2022.10020723.

[15] T. A. E. Putri, T. Widiharih, dan R. Santoso, "Penerapan Tuning Hyperparameter Randomsearchcv Pada Adaptive Boosting Untuk Prediksi Kelangsungan Hidup Pasien Gagal Jantung," *J. Gaussian*, vol. 11, no. 3, hal. 397–406, 2023, doi: 10.14710/j.gauss.11.3.397-406.

[16] A. Tholib, *Buku Refrensi Implementasi Algoritma Machine Learning Berbasis Web dengan Framework Streamlit*, 1 ed. Probolinggo: Pustaka Nurja, 2023. doi: 10.1128/AAC.03728-14.

[17] R. F. Putra *et al.*, *ALGORITMA PEMBELAJARAN MESIN (Dasar, Teknik, dan Aplikasi)*, 1 ed. Jambi: PT. Sonpedia Publishing Indonesia, 2024.

[18] Hartono, *Modul Digital Machine Learning*. [Daring]. Tersedia di: https://lmsspada.kemdiktisaintek.go.id/pluginfile.php/795078/mod_resource/content/2/Modul Digital - Machine Learning -

P-ISSN : 2089-3353
E-ISSN : 2808-9162

Hartono-compressed.pdf. [Accessed 23 Juni 2025]