



Hotel Booking Cancellation Prediction: Random Forest and XGBoost with Leak-Free Pipeline and Ablation Study

Prediksi Pembatalan Pemesanan Hotel: Random Forest dan XGBoost dengan *Pipeline Leak-Free* dan *Ablation Study*

Fanes Arasadina^{1*}, Singgih Briandoko², Muhammad Akbar Setiawan³

^{1,2,3}Program Studi Teknik Informatika, STMIK Widya Utama, Purwokerto, Indonesia

E-Mail: ¹fanesarasadina11@gmail.com, ²briandokosinggih@swu.ac.id, ³akbar@swu.ac.id

Received Mar 13th 2026; Revised Apr 24th 2026; Accepted May 08th 2026; Available Online Jun 20th 2026
Corresponding Author: Fanes Arasadina

Copyright © 2026 by Authors, Published by Institut Riset dan Publikasi Indonesia (IRPI)

Abstract

Hotel booking cancellations cause operational inefficiencies and financial losses of up to 20% of potential daily revenue. This study aims to develop and compare cancellation prediction models using Random Forest and XGBoost on 119,390 hotel booking records, and to identify dominant cancellation factors as a basis for revenue management strategies. Four methodological improvements were applied: training-set-based IQR capping, an ablation study to evaluate deposit_type, post-split OrdinalEncoder, and stratified 70:15:15 data splitting. The ablation study showed that deposit_type did not contribute significantly (delta AUC = 0.0006) and was therefore excluded. Both models were optimized using GridSearchCV with 3-fold cross-validation. Tuned Random Forest achieved the best performance with ROC-AUC 0.9394, accuracy 0.8749, precision 0.8691, and F1-score 0.8219, while Tuned XGBoost showed better recall (0.8345) for early warning. Learning curve analysis confirmed that Tuned XGBoost has better generalization (gap 0.0345) compared to Tuned Random Forest (gap 0.0697). All performance differences were statistically significant based on McNemar's test ($p < 0.05$). Feature importance identified lead_time (12.42%) as the main predictor in Random Forest, while room_type_match (17.40%) dominated XGBoost, indicating room mismatch as a significant operational factor. These findings provide an empirical basis for designing proactive risk-based intervention strategies in hotel management.

Keywords: Ablation Study, Hotel Booking Cancellation Prediction, Leak-Free Pipeline, Random Forest, XGBoost

Abstrak

Pembatalan pemesanan hotel menyebabkan inefisiensi operasional dan kerugian finansial hingga 20% dari potensi pendapatan harian. Penelitian ini bertujuan mengembangkan dan membandingkan model prediksi pembatalan menggunakan Random Forest dan XGBoost pada 119.390 data pemesanan hotel, serta mengidentifikasi faktor dominan pembatalan sebagai dasar strategi manajemen pendapatan. Empat perbaikan metodologis diterapkan, yaitu *IQR capping* berbasis *train set*, *ablation study* untuk mengevaluasi *deposit_type*, *OrdinalEncoder post-split*, dan pembagian data *stratified 70:15:15*. *Ablation study* menunjukkan *deposit_type* tidak berkontribusi signifikan (delta AUC = 0,0006) sehingga dikeluarkan. Kedua model dioptimasi menggunakan *GridSearchCV* dengan *3-fold cross-validation*. *Tuned Random Forest* menghasilkan performa terbaik dengan ROC-AUC 0,9394, *accuracy* 0,8749, *precision* 0,8691, dan *F1-score* 0,8219, sementara *Tuned XGBoost* unggul pada *recall* (0,8345) untuk *early warning*. *Learning curve* mengonfirmasi bahwa *Tuned XGBoost* memiliki generalisasi lebih baik (gap 0,0345) dibandingkan dengan *Tuned Random Forest* (gap 0,0697). Seluruh perbedaan performa dikonfirmasi signifikan melalui uji *McNemar* ($p < 0,05$). *Feature importance* mengidentifikasi *lead_time* (12,42%) sebagai prediktor utama pada Random Forest (semakin panjang jarak pemesanan ke kedatangan, semakin tinggi risiko pembatalan), sementara *room_type_match* (17,40%) mendominasi XGBoost, mengindikasikan ketidaksesuaian kamar sebagai faktor pembatalan yang signifikan secara operasional. Temuan ini memberikan landasan empiris bagi manajemen hotel dalam merancang strategi intervensi berbasis risiko secara proaktif.

Kata Kunci: Ablation Study, Pipeline Leak-Free, Prediksi Pembatalan Pemesanan Hotel, Random Forest, XGBoost

1. PENDAHULUAN

Industri perhotelan merupakan sektor yang sangat dinamis dan kompetitif, di mana tingkat persaingan yang tinggi menuntut pengelolaan sumber daya serta strategi bisnis yang efisien dan adaptif. Salah satu permasalahan krusial yang dihadapi industri ini adalah tingginya tingkat pembatalan pemesanan hotel yang berdampak langsung terhadap kerugian finansial, gangguan perencanaan operasional, serta efektivitas manajemen sumber daya hotel [1], [2]. Perkembangan platform pemesanan daring (*Online Travel Agent/OTA*) meningkatkan aksesibilitas bagi pelanggan, namun sekaligus memperbesar risiko pembatalan mendadak yang berpotensi menyebabkan kehilangan pendapatan hingga 20% dari total transaksi harian [3], [4], [5].

Pembatalan pemesanan hotel dipengaruhi oleh berbagai faktor, meliputi perubahan rencana perjalanan, kondisi ekonomi, fleksibilitas platform pemesanan daring, serta kebijakan pembatalan yang diterapkan hotel [3], [6]. Oleh karena itu, kemampuan memprediksi probabilitas pembatalan secara akurat menjadi kebutuhan strategis guna mendukung pengambilan keputusan manajemen pendapatan hotel [5], [7]. Pemanfaatan *machine learning* untuk prediksi pembatalan pemesanan hotel telah banyak dikaji dengan hasil yang menjanjikan [5], [8], [9]. Random Forest dan XGBoost merupakan metode *ensemble learning* yang terbukti efektif dalam tugas klasifikasi ini. Random Forest membangun sejumlah *decision tree* secara independen menggunakan pendekatan *bagging*, sedangkan XGBoost menerapkan *gradient boosting* secara sekuensial untuk memperbaiki kesalahan prediksi sebelumnya [10], [11].

Namun demikian, kajian sistematis terhadap studi-studi terdahulu mengungkapkan pola kesenjangan metodologis yang konsisten dan belum terselesaikan. Dari perspektif *pipeline* data, Azhar *et al.* [9] menerapkan optimasi *hyperparameter* Random Forest, namun tidak melaporkan pemisahan tahap *preprocessing* dari proses pembagian data, sehingga estimasi performa berpotensi optimistik akibat *statistical leakage* [12]. Prasetya *et al.* [7] membangun model *stacking ensemble* dengan ROC-AUC kompetitif, namun tidak melaporkan penanganan eksplisit terhadap potensi *leakage* pada tahap *encoding* dan *scaling*. Yang dan Miao [5] menggunakan arsitektur *tree-based neural network* yang lebih kompleks, namun tidak menyertakan validasi statistik untuk mengonfirmasi signifikansi perbedaan performa antar model. Studi terbaru Ampountolas [13] dan Silvestre *et al.* [14] menggunakan pendekatan *ensemble* yang kompetitif, namun tetap tidak mengatasi isu *leakage* pada tahap *preprocessing* maupun validasi statistik formal pada konteks data hotel bintang empat.

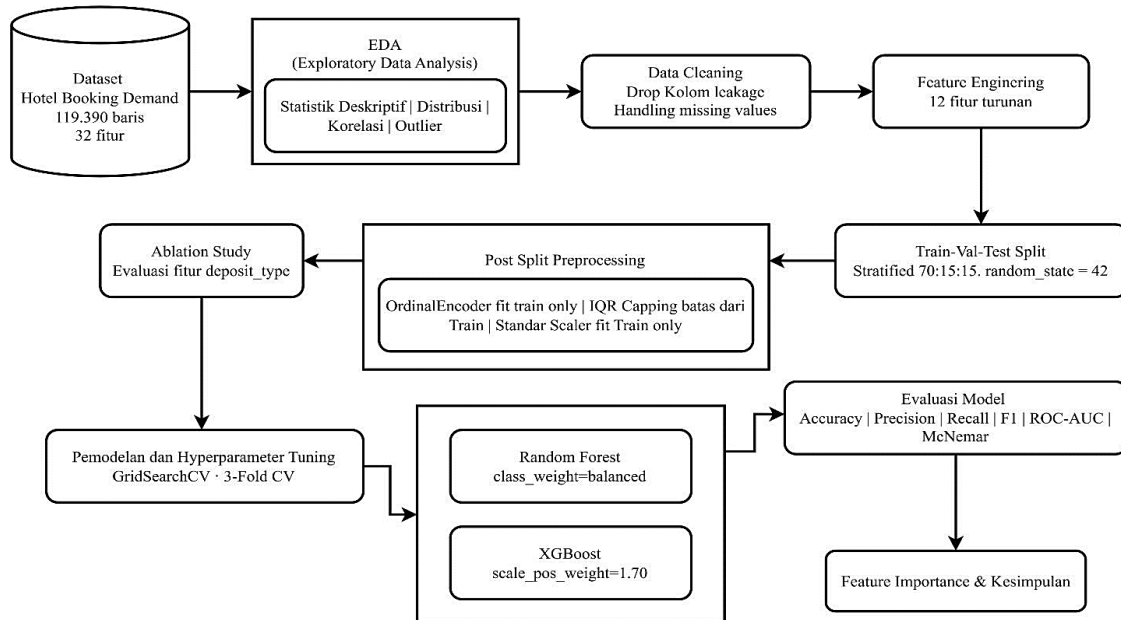
Pola kesenjangan ini bersifat sistematis: pertama, sebagian besar studi menerapkan *preprocessing* seperti *outlier handling* dan *feature encoding* sebelum pembagian data, yang berpotensi menyebabkan kebocoran informasi statistik (*statistical leakage*) dari *test set* ke *training set* [12]; kedua, tidak ada studi yang secara empiris mengevaluasi kontribusi fitur *deposit_type* yang memiliki ambiguitas temporal, padahal kategori *Non Refund* menunjukkan tingkat pembatalan 99,36% yang berpotensi mendominasi model secara artifisial; ketiga, tidak ada studi yang memvalidasi perbedaan performa antar model secara statistik, sehingga klaim keunggulan satu model atas model lain tidak dapat dikonfirmasi signifikansinya. Ketiga kesenjangan ini secara kolektif memotivasi diperlukannya pendekatan yang lebih ketat secara metodologis untuk menghasilkan estimasi performa yang lebih dapat dipercaya.

Selain kesenjangan metodologis tersebut, penelitian terbaru pada domain ini juga menghadapi kendala yang belum sepenuhnya diatasi. Bréchet *et al.* [8] menunjukkan bahwa model *machine learning* yang *interpretable* untuk prediksi pembatalan hotel masih sulit mencapai keseimbangan antara akurasi dan transparansi keputusan. Silvestre *et al.* [14] mengidentifikasi bahwa model yang dilatih pada data historis satu periode waktu menunjukkan degradasi performa ketika diterapkan pada periode berbeda, mengindikasikan tantangan *temporal generalizability* yang belum terpecahkan. Luo [15] menunjukkan bahwa XGBoost dengan rekayasa fitur yang tepat mampu mencapai performa tinggi pada *dataset* skala besar, namun tidak menyertakan evaluasi empiris terhadap kontribusi individual fitur. Kendala-kendala ini memperkuat urgensi pendekatan *pipeline leak-free* dan *ablation study* berbasis bukti empiris yang diusulkan dalam penelitian ini.

Penelitian ini hadir untuk mengatasi kendala-kendala tersebut. Tujuan penelitian ini adalah (1) mengembangkan model prediksi pembatalan pemesanan hotel menggunakan Random Forest dan XGBoost dengan *pipeline preprocessing leak-free* berbasis *post-split*; (2) mengevaluasi kontribusi fitur *deposit_type* yang berpotensi *quasi-leakage* melalui *ablation study* berbasis delta ROC-AUC; (3) melakukan evaluasi komparatif antar empat model menggunakan metrik ROC-AUC, *recall*, *precision*, dan F1-score; (4) menganalisis *feature importance* untuk mengidentifikasi faktor dominan pembatalan; dan (5) memvalidasi perbedaan performa antar model secara statistik menggunakan uji McNemar ($\alpha = 0,05$). Kebaruan penelitian ini terletak pada kombinasi *pipeline preprocessing leak-free*, *ablation study* berbasis bukti empiris, dan validasi statistik McNemar yang belum dilaporkan secara bersamaan dalam literatur prediksi pembatalan pemesanan hotel [16].

2. METODE PENELITIAN

Penelitian ini menggunakan pendekatan kuantitatif dengan desain eksperimen komparatif untuk membangun dan mengevaluasi model prediksi pembatalan pemesanan hotel. Gambaran umum alur penelitian disajikan pada Gambar 1. Tahapan penelitian meliputi pengumpulan data, analisis eksploratif data (*exploratory data analysis* (EDA)), *preprocessing* data, rekayasa fitur, pembagian data, *post-split preprocessing*, *ablation study*, pemodelan, dan evaluasi performa model.



Gambar 1. Alur Penelitian Prediksi Pembatalan Pemesanan Hotel

2.1. Dataset

Dataset yang digunakan adalah *Hotel Booking Demand* yang diperoleh dari platform Kaggle dan dipublikasikan oleh Antonio *et al.* [4]. *Dataset* ini berisi 119.390 data pemesanan hotel yang terdiri atas *City Hotel* dan *Resort Hotel*, dengan periode data dari Juli 2015 hingga Agustus 2017. *Dataset* ini mencakup 32 atribut karakteristik pemesanan dengan variabel target *is_canceled* (0 = tidak dibatalkan, 1 = dibatalkan).

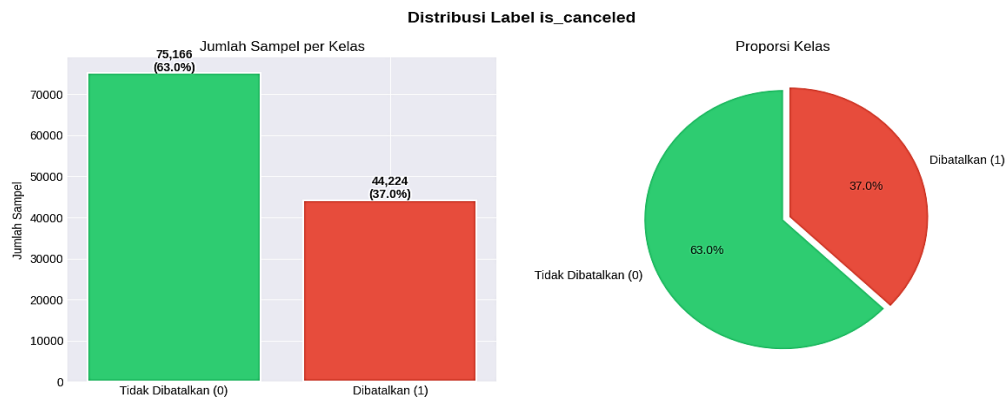
2.2. Preprocessing Data dan Exploratory Data Analysis (EDA)

Tahap *preprocessing* data dilakukan untuk memastikan kualitas dan kesiapan data. Proses ini mencakup penghapusan atribut yang berpotensi menyebabkan *data leakage*, yaitu *reservation_status* dan *reservation_status_date*, karena hanya diketahui setelah keputusan pembatalan terjadi [6]. Atribut *company* dihapus karena memiliki proporsi nilai kosong yang sangat tinggi (94,3%). Atribut *country* dihapus karena kardinalitas yang sangat tinggi (170+ nilai unik) akan menghasilkan ordinalitas artifisial jika menggunakan *ordinal encoding* atau menghasilkan *curse of dimensionality* jika menggunakan *one-hot encoding*.

Penanganan nilai hilang dilakukan sebagai berikut: (1) empat baris dengan nilai hilang pada kolom *children* dihapus karena proporsinya sangat kecil (0,003%), sehingga *dataset* bersih yang digunakan untuk seluruh tahapan selanjutnya berjumlah 119.386 baris; (2) 16.340 nilai hilang pada kolom *agent* diisi dengan konstanta 0 yang merepresentasikan pemesanan tanpa perantara agen, bukan nilai statistik yang dipelajari dari data, sehingga aman dilakukan sebelum *split*.

Analisis eksploratif menunjukkan ketidakseimbangan kelas dengan rasio 37:63 (*canceled: not canceled*), di mana 44.224 pemesanan dibatalkan (37,04%) dan 75.166 pemesanan tidak dibatalkan (62,96%). Distribusi kelas label *is_canceled* ini tertera pada Gambar 2. Rasio ketidakseimbangan sebesar 1,70:1 ini dapat ditangani dengan penyesuaian bobot kelas tanpa memerlukan teknik *resampling* [12], [17]. Analisis korelasi Pearson menunjukkan bahwa *lead_time* memiliki korelasi positif tertinggi terhadap *is_canceled* ($r = 0,29$), sedangkan *required_car_parking_spaces* menunjukkan korelasi negatif ($r = -0,20$).

Analisis mendalam terhadap *deposit_type* menunjukkan bahwa kategori ‘Non Refund’ memiliki tingkat pembatalan sebesar 99,36% (14.494 dari 14.587 pemesanan), sementara ‘No Deposit’ sebesar 28,38% dan ‘Refundable’ sebesar 22,22%. Korelasi ekstrem ini memotivasi dilakukannya *ablation study* untuk mengevaluasi apakah fitur ini memberikan kontribusi diskriminatif yang sah atau berpotensi mendominasi model secara artifisial. Berdasarkan dokumentasi *dataset* (Antonio *et al.* [4]), nilai *deposit_type* dihitung dari tabel transaksi sebelum tanggal kedatangan atau pembatalan, yang mengimplikasikan potensi ambiguitas temporal pada observasi yang dibatalkan.



Gambar 2. Distribusi Kelas Label *is_canceled* pada *Dataset* Hotel Booking Demand (n = 119.390, Sebelum Penghapusan *Missing Values*)

2.3. Rekayasa Fitur (*Feature Engineering*)

Rekayasa fitur dilakukan untuk meningkatkan kualitas representasi data melalui transformasi dan kombinasi variabel mentah menjadi fitur turunan yang merepresentasikan pola perencanaan perjalanan, kompleksitas pemesanan, dan konsistensi perilaku pelanggan. Dua belas fitur turunan dihasilkan dari operasi *row-wise* sehingga tidak memperkenalkan informasi lintas baris yang berpotensi menyebabkan *leakage*.

Seluruh fitur turunan yang memerlukan statistik referensi dihitung secara eksklusif dari *X_train* setelah proses pembagian data, untuk memastikan tidak terjadi kebocoran informasi dari *validation* atau *test set*. Khususnya, fitur *has_discount* dihitung menggunakan rata-rata ADR yang dipelajari hanya dari *X_train*, kemudian diterapkan secara konsisten ke seluruh subset. Fitur *booking_stability* dan *repeated_with_cancellation* dibangun dari fitur historis yang telah tersedia dalam *dataset* asli, yaitu *previous_cancellations* dan *previous_bookings_not_canceled*, yang merepresentasikan riwayat sebelum tanggal *booking* saat ini sesuai dokumentasi Antonio *et al.* [4], bukan dari kolom target *is_canceled*. Rincian fitur disajikan pada Tabel 1.

Tabel 1. Daftar 12 Fitur Rekayasa untuk Representasi Pola Perencanaan Perjalanan dan Perilaku Pelanggan

Nama Fitur	Deskripsi
<i>total_stay</i>	Durasi menginap total (<i>weekend</i> + <i>weekday nights</i>).
<i>lead_time_bins</i>	Kategorisasi waktu pemesanan menjadi 5 tingkat: <i>very_short</i> (0-30 hari), <i>short</i> (31-90 hari), <i>medium</i> (91-180 hari), <i>long</i> (181-365 hari), <i>very_long</i> (>365 hari).
<i>season</i>	Kategori musim (<i>Winter</i> , <i>Spring</i> , <i>Summer</i> , <i>Fall</i>) berdasarkan bulan kedatangan tamu.
<i>is_weekend</i>	Indikator biner (1 jika pemesanan mencakup malam akhir pekan, 0 jika tidak).
<i>has_discount</i>	Indikator apakah ADR lebih rendah dari rata-rata ADR train set (proksi diskon atau harga promosi).
<i>total_guests</i>	Total jumlah tamu (dewasa + anak + bayi).
<i>is_long_stay</i>	Indikator pemesanan berdurasi lebih dari tujuh malam.
<i>has_children</i>	Indikator pemesanan yang melibatkan anak-anak atau bayi.
<i>repeated_with_cancellation</i>	Indikator tamu berulang dengan riwayat pembatalan sebelumnya.
<i>booking_stability</i>	Rasio pemesanan tidak dibatalan terhadap total riwayat pemesanan pelanggan.
<i>room_type_match</i>	Indikator kesesuaian antara tipe kamar yang dipesan dan yang diberikan.
<i>adr_per_night_per_guest</i>	Tarif rata-rata per malam per tamu sebagai normalisasi ADR terhadap durasi dan jumlah tamu.

2.4. Pembagian Data

Dataset dibagi menggunakan strategi tiga arah (70:15:15) dengan metode *stratified splitting* berdasarkan variabel target *is_canceled* untuk mempertahankan proporsi kelas pada setiap subset. *Train set* (70%, 83.570 sampel) digunakan untuk pelatihan model dan optimasi *hyperparameter* melalui GridSearchCV. *Validation set* (15%, 17.908 sampel) digunakan untuk monitoring performa dan konfirmasi model sebelum evaluasi final. *Test set* (15%, 17.908 sampel) digunakan secara eksklusif untuk evaluasi final model terpilih dan tidak pernah diakses selama proses pelatihan maupun seleksi model, sehingga estimasi performa yang dihasilkan bersifat *unbiased*.

Pemilihan rasio 70:15:15 didasarkan pada pertimbangan berikut: dengan *dataset* berukuran besar (119.386 baris), *test set* sebesar 15% menghasilkan 17.908 sampel yang representatif secara statistik. Pemisahan *validation set* dari *test set* mencegah *selection bias*, di mana keputusan pemilihan model tidak boleh didasarkan pada data yang sama yang digunakan untuk evaluasi final [12]. Verifikasi *stratified splitting* menunjukkan proporsi kelas konsisten pada ketiga subset (Train: 62,96%/37,04%; Val: 62,96%/37,04%; Test: 62,96%/37,04%).

2.5. Post-Split Preprocessing

Seluruh proses *preprocessing* yang memerlukan pembelajaran statistik dari data dilakukan setelah pembagian data (*post-split*) untuk menghindari *data leakage*. Tiga tahap *post-split preprocessing* diterapkan secara berurutan:

2.5.1 OrdinalEncoder Post-Split

OrdinalEncoder digunakan untuk mengonversi 11 fitur kategorikal. Pemilihan *OrdinalEncoder* didasarkan pada dua pertimbangan teknis: kemampuan parameter *handle_unknown='use_encoded_value'* yang memungkinkan encoder menangani kategori tidak dikenal saat *inference* tanpa menghasilkan *error*, serta konsistensi transformasi multi-kolom dalam satu *transformer*. Meskipun *OrdinalEncoder* menghasilkan ordinalitas artifisial pada fitur nominal, hal ini tidak menjadi masalah untuk *tree-based models* karena algoritma dapat menemukan *split* optimal pada setiap nilai *encoded* secara independen, sebagaimana dibuktikan secara teoretis oleh Wright dan König [18], [19]. Encoder di-*fit()* hanya pada *train set* dan di-*transform()* pada ketiga subset.

2.5.2 IQR Capping Post-Split

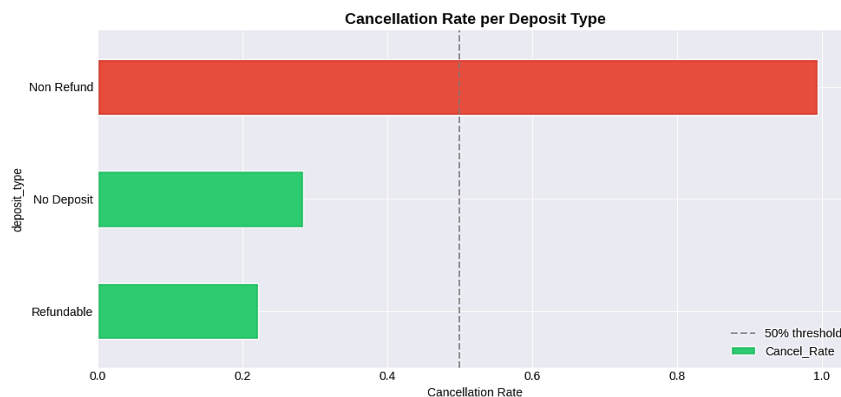
Outlier handling menggunakan metode *IQR capping* dipindahkan ke tahap *post-split*. Batas bawah ($Q1 - 1,5 \times IQR$) dan batas atas ($Q3 + 1,5 \times IQR$) dihitung secara eksklusif dari X_{train} , kemudian diterapkan ke *train*, *validation*, dan *test set* menggunakan batas yang sama untuk mencegah *statistical leakage* [12]. Lima fitur dikenai *IQR capping*: *lead_time* (2.130 nilai di-*cap* dari *train*), *adr* (2.577), *stays_in_weekend_nights* (179), *stays_in_week_nights* (2.387), dan *total_of_special_requests* (2.008).

2.5.3 StandardScaler Post-Split

StandardScaler dilatih hanya pada *train set* dan diterapkan pada ketiga subset. Sebanyak 21 fitur numerik kontinu dinormalisasi, sedangkan fitur biner (7 fitur) dan fitur hasil *encoding* tidak dinormalisasi. Meskipun Random Forest dan XGBoost secara inheren tidak terpengaruh oleh skala fitur numerik, *StandardScaler* tetap diikutsertakan dalam *pipeline* untuk dua pertimbangan: (1) menjaga konsistensi representasi data lintas seluruh subset (*train*, *validation*, *test*) sehingga tidak ada perbedaan distribusi skala yang tidak disengaja; dan (2) memastikan *pipeline* yang dibangun kompatibel apabila dikembangkan di masa depan dengan algoritma yang sensitif terhadap skala fitur, seperti *Support Vector Machine* atau *k-Nearest Neighbors*, tanpa memerlukan perubahan struktural pada arsitektur *preprocessing*.

2.6. Ablation Study deposit_type

Fitur *deposit_type* berpotensi mengandung *quasi-leakage* karena kategori ‘Non Refund’ memiliki tingkat pembatalan 99,36%. Meskipun berdasarkan dokumentasi Antonio *et al.* [4] nilai ini tersedia saat *booking* dibuat, frasa “before the booking’s arrival or cancellation date” dalam deskripsi variabel mengimplikasikan bahwa nilainya dapat berubah mendekati tanggal pembatalan sehingga mengandung ambiguitas temporal. Distribusi tingkat pembatalan per kategori *deposit_type* disajikan pada Gambar 3. Untuk mengukur dampak fitur ini secara empiris dan *data-driven*, dilakukan *ablation study*.



Gambar 3. Tingkat Pembatalan per Kategori *deposit_type*

Metodologi *ablation study*: XGBoost *baseline* dibandingkan pada dua konfigurasi, yaitu (A) *with deposit_type* dan (B) *without deposit_type*, yang dievaluasi pada *validation set*. Pendekatan ini mengikuti metodologi *ablation study* dalam *machine learning* yang secara sistematis menghapus komponen tertentu untuk memahami kontribusinya terhadap performa keseluruhan sistem [20]. *Threshold* delta ROC-AUC sebesar 0,05 ditetapkan sebagai batas praktis dalam penelitian ini, mengacu pada pertimbangan bahwa perbedaan AUC di bawah nilai tersebut dianggap tidak bermakna secara operasional dalam evaluasi model klasifikasi [20].

Hasil *ablation study* pada Tabel 2 menunjukkan delta ROC-AUC = 0,0006, jauh di bawah *threshold* 0,05. F1-score konfigurasi *without* (0,7951) bahkan sedikit lebih tinggi daripada *with* (0,7937). Berdasarkan hasil ini, *deposit_type* dikeluarkan dari fitur pemodelan. *Dataset* final yang digunakan untuk pemodelan memiliki 38 fitur.

Tabel 2. Hasil *Ablation study deposit_type* pada *Validation set*

Konfigurasi	ROC-AUC	F1-Score	Accuracy
XGBoost WITH <i>deposit_type</i>	0,9225	0,7937	0,8437
XGBoost WITHOUT <i>deposit_type</i>	0,9219	0,7951	0,8426
Delta (WITH - WITHOUT)	+0,0006	-0,0014	+0,0011

2.7. Pemodelan dan Optimasi *Hyperparameter*

Model prediksi dibangun menggunakan algoritma Random Forest dan XGBoost. Untuk mengatasi ketidakseimbangan kelas, diterapkan penyesuaian bobot kelas sebagai alternatif *resampling* yang efektif pada rasio ketidakseimbangan moderat [13], [17], [21]: Random Forest menggunakan *class_weight='balanced'* yang menghitung bobot secara otomatis berdasarkan frekuensi invers kelas, sedangkan XGBoost menggunakan *scale_pos_weight = 1,70* yang dihitung dari rasio *n_negatif/n_positif* pada *y_train* (bukan *full dataset*).

Optimasi *hyperparameter* dilakukan menggunakan GridSearchCV dengan 3-fold *StratifiedKfold cross-validation*. GridSearchCV melakukan pencarian *exhaustive* terhadap seluruh kombinasi parameter dalam ruang pencarian yang terdefinisi sehingga menjamin konfigurasi optimal tidak terlewatkan [22]. Pemilihan 3-fold didasarkan pada ukuran *train set* yang besar (83.570 sampel): setiap *fold* menghasilkan *validation set* internal ± 27.857 sampel, yang jauh melebihi minimum untuk estimasi stabil. Kohavi [23] menyatakan bahwa pemilihan nilai *k* dalam *cross-validation* perlu mempertimbangkan ukuran *dataset* untuk menyeimbangkan akurasi estimasi dan efisiensi komputasi; pada *dataset* berukuran besar, penggunaan *k* yang lebih tinggi tidak memberikan peningkatan stabilitas estimasi yang proporsional terhadap biaya komputasi yang ditimbulkan.

Metrik optimasi yang digunakan adalah ROC-AUC yang *robust* terhadap ketidakseimbangan kelas [17]. Pemilihan model final didasarkan pada ROC-AUC sebagai kriteria utama; keunggulan *Tuned* XGBoost pada metrik *recall* merupakan temuan observasional dari evaluasi *test set*, bukan target optimasi GridSearchCV. Ruang pencarian dan konfigurasi *hyperparameter* disajikan pada Tabel 3.

Tabel 3. Konfigurasi Ruang Pencarian *Hyperparameter* GridSearchCV

Model	<i>Hyperparameter</i>	<i>Search space</i>
Random Forest	<i>n_estimators</i>	[100, 200]
	<i>max_depth</i>	[10, 20, None]
	<i>min_samples_split</i>	[2, 5]
	<i>min_samples_leaf</i>	[1, 2]
	<i>max_features</i>	['sqrt', 'log2']
	<i>class_weight</i>	'balanced'
XGBoost	<i>n_estimators</i>	[100, 200]
	<i>max_depth</i>	[5, 7]
	<i>learning_rate</i>	[0.05, 0.1]
	<i>subsample</i>	[0.8, 1.0]
	<i>colsample_bytree</i>	[0.8, 1.0]
	<i>scale_pos_weight</i>	1,70 (dari <i>y_train</i>)

Parameter *max_depth=None* diikutsertakan dalam ruang pencarian mengikuti rekomendasi Breiman [10] untuk *dataset* berukuran besar, di mana pohon tanpa pembatasan kedalaman mampu menangkap pola kompleks tanpa risiko *underfitting*. Berbeda dengan XGBoost yang memiliki regularisasi *built-in* melalui *gamma* dan *lambda*, Random Forest tidak memiliki regularisasi eksplisit per pohon sehingga pembatasan *max_depth* yang ketat berpotensi mengorbankan kapasitas representasi model.

Ruang pencarian *hyperparameter* dirancang dengan mempertimbangkan efisiensi komputasi pada *dataset* berukuran besar. Dengan total 24 kombinasi konfigurasi untuk Random Forest dan 16 untuk

XGBoost, pendekatan ini menghasilkan waktu komputasi yang *feasible* sekaligus tetap mencakup parameter yang paling berpengaruh terhadap performa model [22], [24]. Perlu diakui bahwa ruang pencarian yang relatif terbatas berpotensi melewati konfigurasi optimal, khususnya pada parameter regularisasi XGBoost seperti *gamma* dan *min_child_weight* yang menggunakan nilai *default* dalam penelitian ini.

Namun demikian, stabilitas hasil dapat diverifikasi melalui dua indikator: pertama, *best CV AUC* yang diperoleh (RF: 0,9303; XGBoost: 0,9227) konsisten dengan *test AUC* final (RF: 0,9394; XGBoost: 0,9278), yang menunjukkan tidak adanya degradasi generalisasi; kedua, selisih performa antara *baseline* dan model *Tuned* yang relatif kecil (ΔAUC RF: +0,0007; ΔAUC XGBoost: +0,0020) mengindikasikan bahwa model telah mendekati titik konvergensi pada ruang pencarian yang digunakan. Hal ini konsisten dengan temuan Bergstra dan Bengio [25] bahwa pada banyak problem *machine learning*, sebagian kecil parameter memiliki pengaruh dominan terhadap performa sehingga pencarian terbatas pada parameter kunci dapat menghasilkan solusi yang mendekati optimal. Eksplorasi lebih lanjut menggunakan RandomizedSearchCV atau Bayesian Optimization dengan *search space* yang lebih luas tetap direkomendasikan sebagai arah penelitian selanjutnya [24].

2.8. Evaluasi Model

Evaluasi performa final dilakukan pada *test set* yang tidak pernah diakses selama proses pelatihan dan seleksi model (*completely held-out*). Metrik evaluasi meliputi ROC-AUC sebagai kriteria utama (*robust* terhadap ketidakseimbangan kelas dan *threshold-independent* [7]), serta *accuracy*, *precision*, *recall*, dan F1-score sebagai metrik pelengkap.

Validasi statistik dilakukan menggunakan *McNemar's test* untuk menguji apakah perbedaan performa antarmodel signifikan secara statistik (H_0 : tidak ada perbedaan signifikan; $\alpha = 0,05$). *McNemar's test* dipilih mengikuti rekomendasi Dietterich [16] karena memiliki tingkat *Type I error* rendah dan dirancang khusus untuk membandingkan dua *classifier* pada *dataset* yang sama (mempertimbangkan pola *disagreement* antar model, bukan sekadar selisih akurasi agregat), sehingga menjadikannya lebih tepat dibandingkan uji-*t* konvensional yang mengasumsikan independensi pengukuran. Selain evaluasi berbasis metrik, analisis *feature importance* berbasis *Gini impurity* dilakukan pada model terbaik masing-masing algoritma. Meskipun demikian, metode ini diketahui dapat bias terhadap fitur berkardinalitas tinggi [26].

Sebagai analisis tambahan untuk mengevaluasi stabilitas dan kecukupan data pelatihan, *learning curve* diplot untuk *Tuned* Random Forest dan *Tuned* XGBoost menggunakan metrik ROC-AUC dengan 3-fold *cross-validation* yang konsisten dengan proses optimasi. *Learning curve* memvisualisasikan hubungan antara ukuran data latih dan performa model pada data latih maupun data validasi internal sehingga dapat mengidentifikasi apakah model mengalami *high variance* (gap besar antara *training* dan *validation score*) atau *high bias* (kedua skor rendah), serta mengonfirmasi apakah penambahan data berpotensi meningkatkan performa secara signifikan [17].

3. HASIL DAN PEMBAHASAN

3.1. Perbandingan Performa Model

Proses GridSearchCV menghasilkan *hyperparameter* terbaik untuk masing-masing model: Random Forest dengan *bootstrap=True*, *max_depth=None*, *max_features=sqrt*, *min_samples_leaf=1*, *min_samples_split=2*, *n_estimators=200* (best CV AUC = 0,9303); dan XGBoost dengan *colsample_bytree=1,0*, *learning_rate=0,1*, *max_depth=7*, *n_estimators=200*, *subsample=0,8* (best CV AUC = 0,9227). Konfigurasi *max_depth=None* pada Random Forest mengonfirmasi bahwa pohon tanpa batasan kedalaman optimal untuk *dataset* berukuran besar ini. Perbandingan performa lengkap keempat model pada *test set* disajikan pada Tabel 4.

Tabel 4. Perbandingan Performa Empat Model Prediksi Pembatalan pada *Test set* (N = 17.908)

Model	Accuracy	Precision	Recall	F1-Score	CV AUC (Train)	Test AUC	Cap (ΔAUC)
Baseline RF	0,8731	0,8677	0,7755	0,8190	-	0,9387	-
Baseline XGBoost	0,8479	0,7745	0,8314	0,8019	-	0,9258	-
Tuned RF	0,8749	0,8691	0,7796	0,8219	0,9303	0,9394	+0,0091
Tuned XGBoost	0,8520	0,7810	0,8345	0,8069	0,9227	0,9278	+0,0051

Catatan: CV AUC hanya tersedia untuk model yang melalui proses GridSearchCV dengan 3-fold StratifiedKFold cross-validation. Baseline Random Forest dan baseline XGBoost dilatih dengan parameter default tanpa optimasi cross-validation.

Tuned Random Forest menghasilkan performa terbaik pada empat dari lima metrik: ROC-AUC 0,9394, *accuracy* 0,8749, *precision* 0,8691, dan F1-score 0,8219. Peningkatan performa dibandingkan *baseline* Random Forest tercatat pada ROC-AUC (+0,0007), *accuracy* (+0,0018), *precision* (+0,0014), dan

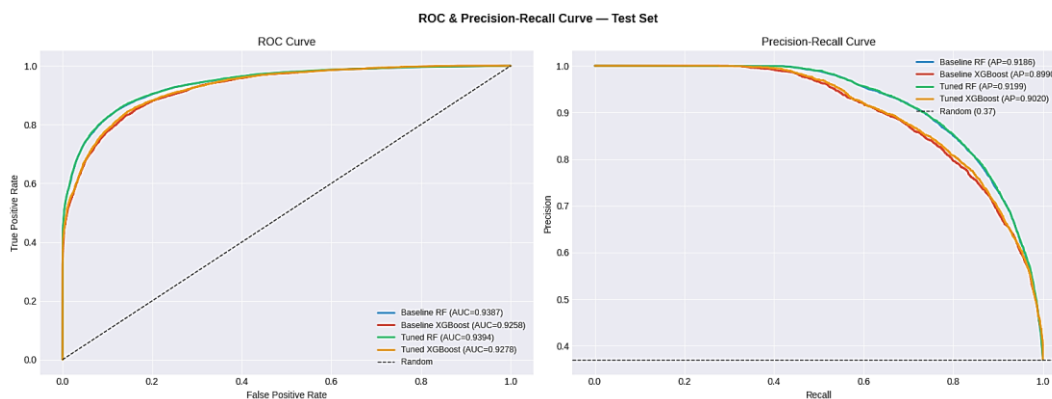
F1-score (+0,0029). *Tuned* XGBoost unggul pada *recall* (0,8345). Analisis *gap* antara CV AUC dan *test* AUC menunjukkan bahwa *Tuned* Random Forest menghasilkan *gap* sebesar +0,0091 dan *Tuned* XGBoost sebesar +0,0051, tanpa terdapat degradasi performa pada *test set*. Perbandingan visual *ROC curve* dan *precision-recall curve* keempat model disajikan pada Gambar 4.

Pada *ROC curve*, *Tuned* Random Forest secara konsisten menunjukkan kurva yang paling dekat dengan sudut kiri atas di seluruh rentang *threshold*. Kedekatan visual antar kurva konsisten dengan selisih AUC yang kecil pada Tabel 4. *Tuned* Random Forest mencapai *average precision* (AP) tertinggi sebesar 0,9199, diikuti *baseline* Random Forest (0,9186), *Tuned* XGBoost (0,9020), dan *baseline* XGBoost (0,8990). Dibandingkan dengan *baseline* acak sebesar 0,37 yang merepresentasikan proporsi kelas positif dalam *dataset*, seluruh model menunjukkan peningkatan *average precision* yang substansial.

Keunggulan konsisten *Tuned* Random Forest pada ROC-AUC dapat dikaitkan dengan mekanisme *bagging* yang membangun pohon secara independen dari subsampel *bootstrap*, sehingga lebih *robust* terhadap variabilitas distribusi fitur ordinal yang di-*encode* secara artifisial. Sebaliknya, XGBoost dengan pendekatan *boosting* sekuensial lebih sensitif terhadap pola residual pada kelas minoritas, yang berkontribusi pada *recall* yang lebih tinggi, namun *precision* yang lebih rendah dibandingkan Random Forest, sehingga menjadikannya lebih sesuai untuk implementasi sistem *early warning*, di mana *false negative* memiliki biaya tinggi berupa kehilangan peluang *remarketing* kamar.

Gap positif CV AUC vs *test* AUC konsisten dengan fakta bahwa model final dilatih pada keseluruhan 83.570 sampel *train*, sedangkan CV internal GridSearchCV hanya menggunakan sekitar 55.700 sampel per *fold*. Kapasitas representasi yang lebih besar ini secara konsisten menghasilkan *test* AUC yang sedikit lebih tinggi dari CV AUC. Konfigurasi *max_depth=None* pada Random Forest yang secara teoretis berpotensi *overfit* terhadap data *training* justru menunjukkan performa yang stabil, yang dapat dijelaskan oleh mekanisme *bagging* dan pemilihan fitur acak yang secara inheren berfungsi sebagai regularisasi implisit pada model *ensemble* ini [10].

Pada *precision-recall curve*, kurva varian Random Forest yang berada di atas varian XGBoost sepanjang rentang *recall* mengindikasikan bahwa Random Forest mempertahankan *precision* lebih tinggi pada setiap level *recall*, keunggulan yang tidak sepenuhnya tertangkap oleh ROC-AUC saja. Pengeluaran *deposit_type* berdasarkan hasil *ablation study* (delta AUC = 0,0006) tidak mengorbankan performa secara signifikan, sekaligus menghasilkan model yang lebih *generalizable* karena prediksi terdistribusi pada fitur dengan validitas temporal yang lebih kuat.



Gambar 4. ROC Curve dan Precision-Recall Curve
Keempat Model Prediksi Pembatalan Hotel pada *Test set* (n = 17.908)

3.2. Uji Signifikansi Statistik (*McNemar's Test*)

Hasil *McNemar's test* untuk empat pasang perbandingan model disajikan pada Tabel 5.

Tabel 5. Hasil Uji *McNemar* Antar Model (A = 0,05)

Perbandingan	Statistik	p-value	Kesimpulan
Baseline RF vs Baseline XGBoost	109,4628	0,0000	Signifikan
Baseline RF vs <i>Tuned</i> RF	5,1117	0,0238	Signifikan
Baseline XGBoost vs <i>Tuned</i> XGBoost	6,7285	0,0095	Signifikan
<i>Tuned</i> RF vs <i>Tuned</i> XGBoost	97,4043	0,0000	Signifikan

Seluruh empat perbandingan model menunjukkan perbedaan yang signifikan secara statistik ($p < 0,05$), mengonfirmasi bahwa perbedaan performa antar model bukan disebabkan oleh kebetulan statistik. Perbandingan *Tuned* Random Forest vs *Tuned* XGBoost menghasilkan statistik *McNemar* tertinggi (97,4043;

$p = 0,0000$), yang menunjukkan perbedaan pola prediksi paling substansial antara kedua model yang telah dioptimasi.

3.3. Analisis *feature importance*

Analisis *feature importance* dilakukan pada model terbaik masing-masing algoritma berdasarkan ROC-AUC *test set*: *Tuned* Random Forest (AUC = 0,9394) dan *Tuned* XGBoost (AUC = 0,9278). *Top-10* fitur penting disajikan pada Tabel 6.

Tabel 6. Top-10 *Feature Importance* Random Forest vs XGBoost (* = Fitur Hasil Rekayasa)

Rank	Random Forest	Imp.	XGBoost	Imp.
1	<i>lead_time</i>	0,1242	<i>room_type_match*</i>	0,1740
2	<i>total_of_special_requests</i>	0,0757	<i>required_car_parking_spaces</i>	0,1481
3	<i>adr_per_night_per_guest*</i>	0,0689	<i>previous_cancellations</i>	0,1399
4	<i>adr</i>	0,0686	<i>total_of_special_requests</i>	0,0821
5	<i>arrival_date_day_of_month</i>	0,0548	<i>customer_type</i>	0,0638
6	<i>arrival_date_week_number</i>	0,0511	<i>market_segment</i>	0,0435
7	<i>agent</i>	0,0492	<i>booking_changes</i>	0,0428
8	<i>room_type_match*</i>	0,0472	<i>lead_time</i>	0,0292
9	<i>market_segment</i>	0,0419	<i>agent</i>	0,0198
10	<i>lead_time_bins*</i>	0,0397	<i>booking_stability*</i>	0,0196

Temuan utama dari analisis *feature importance* adalah fitur *lead_time* (12,42%) yang menjadi prediktor utama pada Random Forest, sedangkan *room_type_match* (17,40%) mendominasi XGBoost. Empat fitur rekayasa (*adr_per_night_per_guest*, *room_type_match*, *lead_time_bins*, dan *booking_stability*) masuk dalam *top-10* kedua model. Sebaliknya, fitur *deposit_type* tidak muncul dalam *top-10* fitur penting pada kedua model.

Dalam penelitian ini, Random Forest menunjukkan distribusi *importance* yang lebih merata, di mana fitur teratas (*lead_time*) hanya berkontribusi 12,42% dan sepuluh fitur teratas berkontribusi sekitar 58%. Sebaliknya, XGBoost menunjukkan konsentrasi lebih tinggi, dengan fitur teratas (*room_type_match*) berkontribusi 17,40% dan sepuluh fitur teratas berkontribusi sekitar 75%. Dominasi *lead_time* pada Random Forest konsisten dengan literatur [1], [5], yang menunjukkan bahwa semakin panjang jarak waktu antara pemesanan dan kedatangan, semakin tinggi probabilitas pembatalan. Dominasi *room_type_match* pada XGBoost mengindikasikan bahwa ketidaksesuaian kamar yang dipesan dengan kamar yang diberikan merupakan sinyal kuat pembatalan.

Distribusi *importance* yang lebih merata pada Random Forest mengindikasikan model yang lebih *robust* terhadap perubahan satu fitur tertentu, sedangkan konsentrasi tinggi pada XGBoost mengonfirmasi keputusan *ablation study* bahwa *deposit_type* tidak memberikan kontribusi diskriminatif yang signifikan ketika fitur-fitur dengan validitas temporal yang lebih kuat tersedia. Perlu dicatat bahwa interpretasi *feature importance* dalam penelitian ini didasarkan pada metode *Gini impurity*, yang bersifat deskriptif dan diketahui dapat menghasilkan estimasi yang bias terhadap fitur dengan kardinalitas tinggi [26]. Oleh karena itu, seluruh temuan pada bagian ini harus dipahami sebagai indikasi asosiasi antara fitur dan target, bukan sebagai hubungan kausal. Konfirmasi yang lebih *robust* memerlukan pendekatan seperti *permutation importance* atau nilai *SHapley Additive exPlanations* (SHAP), yang direkomendasikan sebagai arah penelitian lanjutan.

3.4. Implikasi Bisnis

Berdasarkan hasil analisis, beberapa implikasi bisnis dapat dirumuskan. Angka-angka berikut diperoleh dari analisis deskriptif pada *train set* setelah proses pembagian data. Pertama, *lead_time* yang panjang (>180 hari) berkorelasi dengan tingkat pembatalan 52% berbanding 18% untuk *lead_time* <30 hari, sehingga hotel disarankan menerapkan sistem pengingat konfirmasi bertahap dan skema penalti pembatalan progresif untuk pemesanan dengan *lead_time* panjang. Kedua, ketidaksesuaian *room_type* meningkatkan risiko pembatalan dari 35,2% (kamar sesuai pesanan) menjadi 44,8% (kamar tidak sesuai pesanan) berdasarkan analisis deskriptif fitur *room_type_match*, sehingga hotel perlu meminimalkan perubahan alokasi kamar dan mengimplementasikan sistem notifikasi otomatis kepada tamu. Ketiga, *required_car_parking_spaces* yang menunjukkan korelasi negatif dengan *is_canceled* ($r = -0,20$ pada analisis korelasi Pearson) mengindikasikan bahwa pemesanan tanpa permintaan parkir cenderung memiliki risiko pembatalan lebih tinggi, yang dapat menjadi sinyal awal bagi manajemen hotel.

Untuk mengkuantifikasi potensi dampak implementasi model, dilakukan estimasi sederhana berbasis performa *test set* ($n = 17.908$). Dari 17.908 sampel *test set*, terdapat 6.634 pemesanan yang benar-benar dibatalkan (37,04%). Dengan *recall* 0,8345, *Tuned* XGBoost mampu mendeteksi secara benar 5.536 dari 6.634 pembatalan tersebut (*true positive*), dengan 1.098 pembatalan tidak terdeteksi (*false negative*). Sebaliknya, *Tuned* Random Forest dengan *precision* 0,8691 menghasilkan lebih sedikit *false alarm*; dari

seluruh prediksi pembatalan yang dihasilkan model, 86,91% di antaranya benar-benar terjadi, sehingga lebih efisien dalam alokasi sumber daya intervensi.

Apabila diasumsikan bahwa setiap pembatalan yang berhasil dideteksi lebih awal memungkinkan hotel mengambil tindakan pencegahan, seperti *overbooking* terkontrol, program retensi pelanggan, atau penawaran ulang kamar, maka implementasi *early warning system* berbasis *Tuned XGBoost* berpotensi mengoptimalkan pengelolaan 5.536 pemesanan berisiko per setiap 17.908 transaksi yang diproses. Estimasi dampak finansial yang lebih spesifik memerlukan data operasional hotel yang sesungguhnya dan berada di luar cakupan penelitian ini. Dari perspektif implementasi model, *Tuned Random Forest* direkomendasikan untuk sistem penilaian risiko yang memprioritaskan *precision* dan stabilitas jangka panjang, sedangkan *Tuned XGBoost* direkomendasikan untuk sistem *early warning* yang memprioritaskan *recall* tinggi (deteksi maksimum pembatalan).

3.5. Perbandingan dengan Penelitian Sebelumnya

Perbandingan langsung dengan studi sebelumnya bersifat indikatif karena perbedaan *dataset* dan strategi *preprocessing*. Tabel 7 menyajikan perbandingan kontekstual dengan studi-studi relevan.

Tabel 7. Perbandingan Kontekstual dengan Penelitian Terdahulu

Studi	Metode	Dataset	Metrik Terbaik	Penanganan Leakage	Validasi Statistik
Azhar et al. [9]	RF + GridSearchCV	Hotel Booking Demand (Kaggle)	<i>Accuracy</i> 86,72%	Tidak eksplisit	Tidak ada
Prasetya et al. [7]	<i>Stacking Ensemble</i>	Dataset lokal	<i>Accuracy</i> 0,87, <i>Recall</i> 0,86, AUCROC	Tidak dilaporkan	Tidak ada
Yang & Miao [5]	<i>Tree-based NN</i>	Hotel Booking Demand (Kaggle)	0,8554 (H1) / 0,8370 (H2)	Tidak eksplisit	Tidak ada
Ampountolas [13]	XGBoost + <i>Ridge Hybrid</i>	Dataset hotel 4-bintang	Tidak dilaporkan	Tidak dilaporkan	Tidak ada
Silvestre et al. [14]	ML Adaptif	Multi-hotel Eropa	Tidak dilaporkan	Tidak eksplisit	Tidak ada
Penelitian ini	RF + XGBoost	Hotel Booking Demand (Kaggle)	ROC-AUC 0,9394	Eksplisit (<i>post-split</i>)	<i>McNemar's test</i>

Satu-satunya studi dengan *dataset* identik adalah Azhar et al. [9] yang melaporkan *accuracy* 86,72%, yang lebih tinggi 1,57 poin persentase dibandingkan penelitian ini (87,49%). Perbedaan ini dapat dijelaskan oleh tiga faktor metodologis. Pertama, Azhar et al. [9] tidak memisahkan tahap *preprocessing* dari pembagian data sehingga statistik *outlier capping* dan *encoding* kemungkinan dihitung dari keseluruhan *dataset*, termasuk *test set*, yang berpotensi menghasilkan estimasi performa yang optimistis akibat *statistical leakage* [12]. Kedua, Azhar et al. [9] tidak menggunakan *validation set* terpisah untuk *model selection*, sehingga keputusan pemilihan model berpotensi dipengaruhi oleh data yang sama dengan evaluasi final (*selection bias*). Ketiga, penelitian ini mengeksklusi fitur *deposit_type* berdasarkan hasil *ablation study* (delta AUC = 0,0006), sementara Azhar et al. [9] kemungkinan mempertahankan fitur tersebut, yang dapat mendongkrak performa secara artifisial mengingat kategori Non Refund memiliki tingkat pembatalan 99,36%.

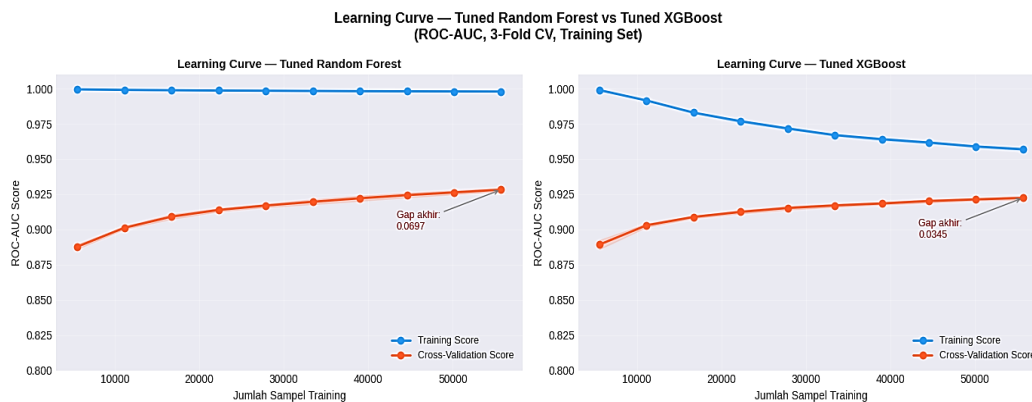
Dibandingkan dengan Prasetya et al. [7] yang menggunakan arsitektur *stacking ensemble* yang lebih kompleks, penelitian ini mencapai *accuracy* 0,87 dan *recall* 0,86 dengan pendekatan yang lebih sederhana dan metodologi yang lebih transparan. Yang dan Miao [5] menggunakan *tree-based neural network* dengan kompleksitas arsitektur lebih tinggi, namun tidak menyertakan validasi statistik sehingga klaim keunggulan performanya tidak dapat dikonfirmasi signifikansinya. Studi terbaru Ampountolas [13] dan Silvestre et al. [14] mengonfirmasi bahwa model *ensemble* berbasis *tree* tetap menjadi pendekatan dominan dalam prediksi pembatalan hotel, sementara Luo [15] menunjukkan bahwa XGBoost dengan rekayasa fitur yang tepat mampu mencapai performa tinggi pada *dataset* skala besar.

Secara keseluruhan, kontribusi utama penelitian ini bukan pada performa absolut tertinggi, melainkan pada validitas metodologis: kombinasi pipeline *leak-free*, *ablation study* berbasis bukti empiris, dan validasi statistik *McNemar's test* yang menghasilkan estimasi performa yang lebih dapat dipercaya dibandingkan studi-studi sebelumnya.

3.6. Analisis learning curve

Learning curve untuk *Tuned Random Forest* dan *Tuned XGBoost* disajikan pada Gambar 5. Hasil *learning curve* menunjukkan pola yang berbeda antara kedua model. *Tuned Random Forest* memperlihatkan *training score* yang sangat tinggi dan stabil mendekati 1,000 di seluruh rentang ukuran data, sementara *cross-validation score* meningkat secara bertahap dari 0,888 menuju 0,928, namun belum sepenuhnya konvergen

dengan *training score*. *Gap* akhir sebesar 0,0697 mengindikasikan kondisi *high variance*; model mampu mempelajari pola *training set* secara hampir sempurna, namun kapasitas generalisasinya masih terbatas. Kondisi ini konsisten dengan konfigurasi *max_depth=None* yang memungkinkan pohon tumbuh tanpa batas kedalaman, sehingga setiap pohon cenderung *overfit* pada subsampelnya masing-masing, meskipun mekanisme *bagging* mereduksi dampaknya pada level *ensemble* [10].



Gambar 5. Learning curve Tuned Random Forest dan Tuned XGBoost (ROC-AUC, 3-fold CV, training set $n = 83.570$)

Tuned XGBoost menunjukkan pola yang lebih sehat: *training score* menurun secara gradual dari 1,000 menuju 0,957 seiring bertambahnya data, sementara *cross-validation score* meningkat stabil dari 0,888 menuju 0,922. *Gap* akhir sebesar 0,0345 (jauh lebih kecil dibandingkan Random Forest) mengindikasikan generalisasi yang lebih baik, dengan *slight gap* yang masih dalam batas wajar untuk *dataset* berukuran besar [25]. Tren konvergensi yang terlihat pada XGBoost mengindikasikan bahwa penambahan data berpotensi mempersempit *gap* lebih lanjut, sedangkan pada Random Forest, penambahan data saja tidak akan cukup tanpa pembatasan kedalaman pohon.

Secara keseluruhan, temuan *learning curve* ini memberikan konteks penting terhadap hasil Tabel 4: keunggulan *test AUC* Tuned Random Forest (0,9394) dibandingkan Tuned XGBoost (0,9278) perlu dimaknai bersama fakta bahwa Random Forest memiliki *variance* lebih tinggi, sehingga Tuned XGBoost dapat menjadi pilihan yang lebih *robust* pada skenario distribusi data yang bergeser (*distribution shift*).

4. KESIMPULAN

Empat temuan utama penelitian ini memiliki implikasi metodologis yang melampaui domain prediksi pembatalan hotel. Pertama, pipeline *leak-free* yang ketat terbukti menghasilkan estimasi performa yang lebih konservatif, namun lebih dapat dipercaya: Tuned Random Forest mencapai ROC-AUC 0,9394 yang kompetitif meskipun tanpa memanfaatkan kontribusi artifisial fitur *deposit_type* yang berambiguitas temporal. Kedua, *ablation study* berbasis delta ROC-AUC menunjukkan bahwa fitur dengan korelasi ekstrem terhadap target (99,36% untuk kategori ‘Non Refund’) tidak selalu meningkatkan performa model secara bermakna ketika pipeline diterapkan secara ketat; temuan ini mengimplikasikan bahwa sebagian performa tinggi yang dilaporkan studi sebelumnya kemungkinan berasal dari kontribusi artifisial, bukan kapasitas diskriminatif model yang sesungguhnya. Ketiga, validasi *McNemar’s test* mengonfirmasi bahwa seluruh perbedaan performa antar model signifikan secara statistik ($p < 0,05$), sehingga memberikan dasar yang lebih kuat untuk rekomendasi pemilihan model dibandingkan perbandingan metrik agregat semata. Keempat, analisis *learning curve* mengungkapkan bahwa Tuned XGBoost memiliki karakteristik generalisasi yang lebih baik (*gap train-CV AUC* = 0,0345) dibandingkan Tuned Random Forest (*gap* = 0,0697), yang mengindikasikan bahwa keunggulan *test AUC* Random Forest perlu dimaknai bersama fakta bahwa model tersebut memiliki *variance* lebih tinggi, sehingga XGBoost menjadi pilihan yang lebih *robust* pada skenario distribusi data yang bergeser (*distribution shift*).

Hasil penelitian ini berlaku dalam konteks berikut: *dataset* pemesanan hotel di Portugal (2015–2017), rasio ketidakseimbangan kelas moderat (37:63), dan fitur yang tersedia pada saat pemesanan dibuat. Generalisasi ke konteks hotel Indonesia, periode waktu berbeda, atau sistem reservasi dengan struktur fitur yang berbeda memerlukan validasi ulang. Khususnya, dominasi *lead_time* sebagai prediktor utama pada Random Forest dan *room_type_match* pada XGBoost perlu diverifikasi pada konteks hotel domestik Indonesia, di mana pola pemesanan dan kebijakan pembatalan dapat berbeda secara struktural dari *dataset* Portugal 2015–2017 yang digunakan. Keterbatasan pada ruang pencarian *hyperparameter* dan penggunaan *Gini impurity* sebagai metode *feature importance* juga perlu diperhatikan dalam menginterpretasi temuan. Untuk penelitian selanjutnya, tiga arah yang paling menjanjikan adalah: (1) replikasi pipeline metodologis ini

pada data hotel domestik Indonesia untuk menguji *transferability* model; (2) implementasi *permutation importance* atau nilai SHAP sebagai pengganti *Gini impurity* untuk interpretasi yang lebih *robust*; dan (3) eksplorasi Bayesian Optimization dengan *search space* yang lebih luas sebagai alternatif GridSearchCV yang lebih efisien secara komputasi.

REFERENSI

- [1] A. Herrera, Á. Arroyo, A. Jiménez, and Á. Herrero, "Forecasting hotel cancellations through *machine learning*," *Expert Syst.*, vol. 41, no. 9, pp. 1–19, 2024, doi: 10.1111/exsy.13608.
- [2] E. Rahmawati and G. S. Nurohim, "Optimization of Prediction for Cancellation of Hotel Room Reservation Using *Decision tree* with *Feature Selection* and *Resampling*," *J. Sist. Inf. Bisnis*, vol. 15, no. 2, pp. 211–215, Jun. 2025, doi: 10.14710/vol15iss2pp211-215.
- [3] A. J. Sánchez-Medina and E. C-Sánchez, "Using *machine learning* and big data for efficient forecasting of hotel booking cancellations," *Int. J. Hosp. Manag.*, vol. 89, Aug. 2020, doi: 10.1016/j.ijhm.2020.102546.
- [4] N. Antonio, A. de Almeida, and L. Nunes, "Hotel booking demand *datasets*," *Data Br.*, vol. 22, pp. 41–49, Feb. 2019, doi: 10.1016/j.dib.2018.11.126.
- [5] D. Yang and X. Miao, "Predicting hotel booking cancellations using tree-based neural network," *PeerJ Comput. Sci.*, vol. 10, pp. 1–16, 2024, doi: 10.7717/peerj-cs.2473.
- [6] E. C-Sánchez and A. J. Sánchez-Medina, "Detecting Short-Notice Cancellation in Hotels with *Machine learning*," *Eng. Proc.*, vol. 68, no. 1, 2024, doi: 10.3390/engproc2024068043.
- [7] J. Prasetya, S. I. Fallo, and M. A. Aprihartha, "Stacking *Machine learning* Model for Predict Hotel Booking Cancellations," *J. Mat. Stat. dan Komputasi*, vol. 20, no. 3, pp. 525–537, May 2024, doi: 10.20956/j.v20i3.32619.
- [8] C. Bréchet, R. Saura, and P. Salom, "*Machine learning* in Hospitality: Interpretable Forecasting of Booking Cancellations," *IEEE Access*, vol. 13, pp. 1–17, 2025, doi: 10.1109/ACCESS.2025.3536094.
- [9] Y. Azhar, G. A. Mahesa, and M. C. Mustaqim, "Prediction of hotel bookings cancellation using *hyperparameter optimization* on Random Forest algorithm," *J. Teknol. dan Sist. Komput.*, vol. 9, no. 1, pp. 15–21, 2021, doi: 10.14710/jtsiskom.2020.13790.
- [10] L. Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
- [11] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2016, pp. 785–794. doi: 10.1145/2939672.2939785.
- [12] S. Kaufman, S. Rosset, C. Perlich, and O. Stitelman, "*Leakage* in data mining: Formulation, detection, and avoidance," *ACM Trans. Knowl. Discov. Data*, vol. 6, no. 4, pp. 1–21, Dec. 2012, doi: 10.1145/2382577.2382579.
- [13] A. Ampountolas, "Predicting hotel booking cancellations: a comprehensive machine learning approach," *J. Revenue Pricing Manag.*, vol. 24, pp. 539–550, 2025, doi: 10.1057/s41272-025-00532-x.
- [14] P. Silvestre, N. Antonio, and P. Carrasco, "Navigating uncertainty: enhancing hotel cancellation predictions with adaptive machine learning," *Inf. Technol. Tour.*, vol. 28, no. 9, 2026, doi: 10.1007/s40558-025-00349-9.
- [15] Z. Luo, "Hotel cancellation rate prediction: a *machine learning* based prediction model," in *Proc. 3rd Int. Conf. Image, Algorithms, Artif. Intell. (ICIAAI 2025)*, Atlantis Press, 2025, pp. 318–327, doi: 10.2991/978-94-6463-823-3_31.
- [16] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, Oct. 1998, doi: 10.1162/089976698300017197.
- [17] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009, doi: 10.1109/TKDE.2008.239.
- [18] P. Cerda and G. Varoquaux, "Encoding high-cardinality string categorical variables," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 3, pp. 1164–1176, Mar. 2022, doi: 10.1109/TKDE.2020.2992529.
- [19] M. N. Wright and I. R. König, "Splitting on categorical predictors in random forests," *PeerJ*, vol. 7, p. e6339, Feb. 2019, doi: 10.7717/peerj.6339.
- [20] S. Sheikholeslami, M. Meister, T. Wang, A. H. Payberah, V. Vlassov, and J. Dowling, "AutoAblation: Automated parallel ablation studies for deep learning," in *Proc. 1st Workshop Mach. Learn. Syst. (EuroMLSys'21)*, ACM, 2021, doi: 10.1145/3437984.3458835.
- [21] S. Li et al., "Multi-Class Imbalance Classification Based on Data Distribution and Adaptive Weights," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 10, pp. 5265–5279, 2024, doi: 10.1109/TKDE.2024.3384961.
- [22] F. Pedregosa et al., "Scikit-learn: *Machine learning* in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

- [23] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in Proc. 14th Int. Joint Conf. Artif. Intell. (IJCAI), vol. 2, 1995, pp. 1137–1145.
- [24] H. Darmawan, M. Yuliana, and M. Z. S. Hadi, "GRU and XGBoost Performance with *Hyperparameter*Tuning Using *GridSearchCV* and Bayesian Optimization on an IoT-Based Weather Prediction System," Int. J. Adv. Sci. Eng. Inf. Technol., vol. 13, no. 3, pp. 851–862, 2023, doi: 10.18517/ijaseit.13.3.18377.
- [25] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," J. Mach. Learn. Res., vol. 13, pp. 281–305, 2012.
- [26] G. Louppe, L. Wehenkel, A. Suter, and P. Geurts, "Understanding variable importances in forests of randomized trees," in Adv. Neural Inf. Process. Syst. (NIPS), vol. 26, 2013, pp. 431–439. [Online]. Available: <https://proceedings.neurips.cc/paper/2013/hash/e3796ae838835da0b6f6ea37bcf8bcb7-Abstract.html>