Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.6, December 2025, pp. 3892~3903
e-ISSN: 2548-6861

Match Outcome Prediction in Draft Pick and In-game Phases of MSC 2025 Mobile Legends using Random Forest and XGBoost

Dzaky Fadli Firmansyah [1]*, Adam Prayogo Kuncoro [2]*, Riyanto [3]*
* Informatika, Fakultas Ilmu Komputer, Universitas Amikom Purwokerto
zaaaafl654@gmail.com [1], adam@amikompurwokerto.ac.id [2], riyanto@amikompurwokerto.ac.id [3]

Article Info

Article history:

Received 2025-10-31
Revised 2025-11-27
Accepted 2025-12-10

Keywords:

Draft Pick,
In-game,
Mobile Legends,
Random Forest,
XGBoost.

ABSTRACT

Mobile Legends: Bang Bang is a widely played Multiplayer Online Battle Arena game in Southeast Asia, and its competitive ecosystem has driven the need for accurate match outcome prediction. Most existing studies analyze either the draft pick phase or the in game phase in isolation, limiting their ability to capture the full progression of a match. To address this limitation, this study evaluates the performance of Random Forest and Extreme Gradient Boosting (XGBoost) in predicting match outcomes across both phases using data from the MSC 2025 tournament. The dataset was collected from Liquipedia's official API and match replay recordings. Draft pick features represent team composition factors such as synergy, hero strength, and patch impact, while in game features consist of statistical indicators including gold, kills, turrets, and objectives extracted from multiple time based snapshots. Both models were trained using qualification stage matches and tested on the main event. A phase separated hybrid feature engineering approach was employed to represent strategic differences between the draft pick and in game phases. Evaluation metrics include accuracy, precision, recall, F1 score, and ROC AUC. Results show that the draft pick models achieved a maximum accuracy of 57%, whereas the in game models reached 88% for Random Forest and 84% for XGBoost, with both achieving a ROC AUC of 0.94. These findings indicate that snapshot based in game features provide stronger predictive signals than draft pick composition features, which reflect only the initial strategic potential rather than actual match conditions.

I. PENDAHULUAN

Mobile Legends: Bang Bang! (MLBB) merupakan salah satu *game* bergenre *Multiplayer Online Battle Arena* (MOBA) yang populer di Asia Tenggara [1] dengan pengguna aktif bulanan lebih dari 50 juta di Indonesia [2]. Genre *game* ini menempatkan dua tim untuk saling menghancurkan markas, di mana setiap pemain mengendalikan satu *hero* dengan peran serta kemampuan yang berbeda [3]. Popularitas MLBB mendorong pertumbuhan ekosistem *esports* kompetitif melalui liga profesional regional yang memerlukan strategi matang dalam setiap pertandingan.

Setiap pertandingan (*match*) profesional terdapat beberapa permainan (*game*), dengan tiap *game* diawali fase *draft pick* untuk menentukan komposisi serta pelarangan *hero* yang akan dipilih oleh tim [4]. Hingga kini terdapat 130 *hero*, termasuk Obsidia, yang memiliki keunggulan dan kelemahan masing-masing sehingga menambah kompleksitas strategi [5]. Fase ini menentukan strategi awal, tetapi hasil pertandingan tetap dipengaruhi faktor dinamis seperti performa pemain, durasi permainan, serta indikator statistik *in-game* (dalam permainan) seperti *gold* dan objektif utama.

Penelitian *machine learning* terkait prediksi hasil pertandingan MLBB masih terbatas. Putro dkk. [6] menggunakan *Gaussian Naïve Bayes* dan *Decision Tree* pada fase *draft pick*, dengan hasil bahwa model pada fase ini relatif lemah dengan nilai AUC sebesar 67% dari *Decision Tree* karena hanya mencerminkan strategi awal. Sena dan Emanuel [3] menunjukkan bahwa indikator statistik *in-game* lebih representatif terhadap peluang kemenangan, dengan akurasi

82% pada model *Artificial Neural Network* dan 80% pada model *Random Forest*. Penelitian Hamir dan Andono [7] yang menggabungkan kedua fase menggunakan *Neural Network* memperoleh akurasi 85%, namun proses pembentukan fitur tidak dijelaskan secara jelas. Sementara itu, pada *game* yang berbeda yaitu League of Legends yang dilakukan Chowdhury dkk. [8] menunjukkan bahwa model gabungan fase *pre-game* (statistik individu pemain publik) dan fase *in-game* mampu memberikan akurasi tertinggi sebesar 76,8%.

Pemilihan algoritma pada penelitian terdahulu juga menunjukkan pola yang konsisten. Artikel tinjauan oleh Kamal dkk. [9] menegaskan bahwa algoritma gabungan (*ensemble*) seperti *Random Forest* dan *Extreme Gradient Boosting* (*XGBoost*) unggul pada domain MOBA karena mampu memodelkan hubungan antarvariabel yang tidak bersifat linear, bekerja efektif pada data *tabular* dengan struktur fitur yang beragam, serta tetap stabil meskipun terdapat variasi dan ketidakpastian dalam strategi permainan. Temuan ini diperkuat oleh penelitian Stanlly dkk. [10] pada *game* DOTA2 yang merupakan MOBA dengan kompleksitas tinggi. Penelitian tersebut menghasilkan akurasi sekitar 91% untuk *Random Forest* dan mendekati 94% untuk *XGBoost*. Konsistensi performa lintas *game* MOBA tersebut menunjukkan bahwa kedua algoritma memiliki kemampuan generalisasi yang kuat sehingga relevan digunakan dalam prediksi hasil pertandingan MLBB.

Selain itu, algoritma lain seperti *LightGBM*, *CatBoost*, dan model *deep learning* tidak digunakan karena ukuran *dataset* yang cenderung terbatas serta seluruh fitur telah direkayasa menjadi bentuk numerik. Kondisi ini membuat keunggulan teknis model tersebut tidak memberikan manfaat signifikan dan justru meningkatkan risiko *overfitting*. Oleh karena itu, penelitian ini difokuskan pada algoritma *ensemble* berbasis *Decision Tree* yang lebih stabil dan sesuai untuk data *tabular* berskala menengah seperti pada studi ini.

Berdasarkan kajian tersebut, terdapat tiga celah penelitian yang belum terjawab, yaitu terbatasnya analisis yang membandingkan kemampuan prediksi antara fase *draft pick* dan fase *in-game*, belum adanya pembahasan mengenai pengaruh perubahan versi *game* khususnya rilis *patch* yang mengubah kekuatan *hero* sehingga pola permainan antara babak kualifikasi dan *main event* dapat bergeser, serta kurang jelasnya metode pembentukan fitur yang merepresentasikan kekuatan *draft pick*. Rumusan masalah dalam penelitian ini berangkat dari kebutuhan untuk memahami perbedaan kekuatan prediksi pada kedua fase tersebut, menilai ketahanan model terhadap perubahan *patch*, dan mengidentifikasi fitur yang paling berpengaruh terhadap peluang kemenangan.

Sejalan dengan itu, penelitian ini bertujuan menganalisis performa model pada kedua fase, mengevaluasi stabilitas algoritma *Random Forest* dan *XGBoost* terhadap *patch* yang berbeda, serta menelaah kontribusi setiap fitur melalui pendekatan *feature engineering* hibrida yang memisahkan fase *draft pick* dan *in-game*. Fitur *draft pick* dibangun dari komposisi *hero*, sedangkan fase *in-game* direpresentasikan melalui *snapshot* statistik menit tertentu untuk mencerminkan dinamika permainan pada pertengahan hingga akhir pertandingan. Pendekatan ini mengacu pada studi Tyran dan Chomatek [11] pada *game* League of Legend yang juga menggunakan pendekatan *snapshot*.

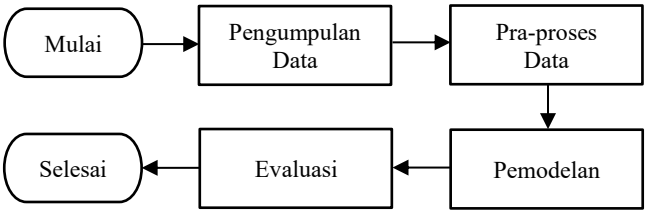
Ruang lingkup penelitian ini dibatasi pada turnamen Mobile Legends Mid Season Cup (MSC) 2025 demi menjaga konsistensi *patch* serta kualitas data. Model *draft pick* tidak diarahkan untuk generalisasi lintas *patch* karena faktor *Most Effective Tactic Available* (META), prioritas *hero*, dan sinergi tim dapat berubah mengikuti pembaruan *patch*. Sebaliknya, fitur *in-game* lebih stabil karena berbasis indikator statistik seperti *gold*, *kill*, *turret*, dan objektif utama. *Dataset* dipisahkan antara babak kualifikasi dan *main event* untuk menilai dampak perbedaan *patch* secara langsung.

MSC 2025 dipilih sebagai sumber data karena merupakan turnamen internasional berskala besar yang merepresentasikan META terbaru dengan perbedaan *patch* antara babak kualifikasi (*patch* 1.9.68) dan *main event* (*patch* 1.9.91). Kondisi ini memungkinkan penilaian lebih akurat terhadap perubahan keseimbangan *hero* antar *patch*, sementara sebagian besar studi sebelumnya fokus pada turnamen profesional regional seperti MPL [12] atau kejuaraan dunia M World Championship [4].

Dengan membandingkan dua fase secara terpisah, penelitian ini memberikan pemahaman yang lebih komprehensif mengenai dinamika prediksi hasil pertandingan MLBB. Penelitian ini berkontribusi pada aspek metodologis dengan menggabungkan fitur berbasis komposisi *hero* pada fase *draft pick* dan *snapshot* statistik permainan pada fase *in-game* sebagai pendekatan berbasis waktu. Pendekatan ini tidak ditunjukkan sebagai prediksi secara *real-time*, melainkan sebagai analisis titik waktu yang dapat menjadi dasar awal bagi penelitian lanjutan yang mengembangkan model prediksi berbasis aliran data saat permainan berlangsung.

II. METODE

Metode penelitian ini mengacu pada tahapan umum dalam *data mining* dengan beberapa penyesuaian, mencakup Pengumpulan Data, Pra-proses Data, Pembuatan Model, Evaluasi, dan Kesimpulan. Tahapan penelitian ini disusun pada Gambar 1.



Gambar 1. Alur tahapan penelitian

A. Pengumpulan Data

Data penelitian diperoleh dari tiga sumber utama, yaitu API Liquipedia, API pihak ketiga MLBB, dan rekaman pertandingan resmi. Seluruh data dibagi menjadi data latih yang berasal dari babak kualifikasi (*patch* 1.9.68) dan data uji dari *main event* MSC 2025 (*patch* 1.9.91). Pemisahan ini secara alami menimbulkan perbedaan karakteristik data karena perubahan *patch*, tingkat kompetitif tim yang lebih tinggi pada turnamen MSC 2025, serta komposisi *hero* yang muncul pada kedua fase turnamen.

1) *API Liquipedia*: Data dari API Liquipedia dikumpulkan melalui API *scraping* pada turnamen kualifikasi serta *main event* dari turnamen MSC 2025. Data yang dikumpulkan mencakup daftar *hero* yang dipilih, serta identitas pemenang pada sisi Blue atau Red. Beberapa entri mentah dari Liquipedia tidak selalu konsisten dengan data yang terlihat pada rekaman pertandingan, sehingga diperlukan proses verifikasi manual pada tahap pra-proses untuk memastikan bahwa data benar-benar sesuai dengan kondisi aktual di dalam *game*.

2) *API Pihak Ketiga MLBB*: Data *rating hero* diperoleh dari API pihak ketiga untuk menyediakan data yang dibutuhkan dalam pembentukan fitur *draft pick*. Data tersebut meliputi peran (*role*), tingkat ketahanan, daya serang, efek kontrol, tingkat kesulitan, tipe serangan (*damage type*), dan posisi *hero* pada arena. Data ini kemudian digunakan untuk membangun fitur turunan seperti *aggressiveness*, *damage efficiency*, *tankiness efficiency*, serta indikator lain yang diperlukan pada fase *draft pick*. Seluruh *rating hero* bersifat statis, yang artinya *rating hero* tidak berubah ketika terdapat perubahan pada *patch*.

3) *Rekaman Pertandingan*: Rekaman video pertandingan digunakan untuk mengekstraksi *snapshot* statistik permainan pada menit ke-9, ke-12, ke-15, dan ke-18. Pemilihan menit ke-9 mempertimbangkan kondisi permainan pada *mid game*, sedangkan menit ke-18 menggambarkan kondisi akhir pada *late game* yang ditandai dengan kemunculan *Lord* yang berevolusi. Proses ekstraksi dilakukan melalui dokumentasi visual menggunakan *screenshot* pada *timestamp* yang telah disinkronkan dengan waktu saat pertandingan berlangsung, untuk menjaga konsistensi tiap data *snapshot*. Setiap *screenshot* kemudian digunakan untuk mencatat nilai *gold*, *kill*, *turret*, *turtle*, *lord*, serta indikator objektif lainnya secara manual guna memastikan akurasi data.

4) *Patch Note MLBB*: *Patch note* versi pembaruan 1.9.68 dari data latih dan 1.9.91 dari data uji dikumpulkan melalui Discord server resmi MLBB. Informasi ini digunakan untuk menentukan perubahan kekuatan *hero*, dan dampak *patch* yang berpotensi memengaruhi pola kemenangan.

B. Pra-proses Data

Tahap pra-proses data dilakukan untuk memastikan data siap digunakan dalam proses pemodelan. Tahap ini mencakup verifikasi data *draft pick*, pembersihan dan sinkronisasi data *in-game*, serta pembuatan fitur pada kedua fase.

1) *Verifikasi dan Pembersihan Data Draft Pick*: Data *draft pick* dari API Liquipedia diverifikasi ulang menggunakan rekaman pertandingan karena beberapa entri tidak lengkap atau tidak sesuai dengan kondisi aktual di dalam *game*. Verifikasi dilakukan dengan mencocokkan susunan *hero* yang dipilih dengan data yang diperoleh dari API. Pertandingan yang tidak memiliki informasi *draft pick* yang valid atau tidak dapat diverifikasi melalui rekaman dihapus dari *dataset* agar tidak memengaruhi akurasi model.

2) *Pembuatan Fitur Draft Pick*: Fitur *draft pick* dibentuk menggunakan statistik *patch* 1.9.68 sebagai data latih. Statistik ini mencakup frekuensi pemilihan, tingkat kemenangan, dan larangan pemilihan (*banned*) *hero* untuk menggambarkan performa dan kecenderungan penggunaan *hero* pada *patch* tersebut. Data ini digunakan untuk memperoleh skor META, membentuk daftar kuat terhadap (*strong against)* dan lemah terhadap (*weak against*) berdasarkan tingkat kemenangan tiap *hero*, serta menghitung sinergi pasangan *hero* dalam satu tim melalui analisis frekuensi kemenangan pasangan.

Fitur yang dikembangkan meliputi *rating hero*, keseimbangan tipe *damage*, *power spike*, sinergi, *strong weak hero*, META, dan *patch impact*. *Rating hero* dihitung dari penjumlahan atribut *offense*, *durability*, *control effects*, dan *difficulty*, lalu diturunkan menjadi fitur seperti *aggressiveness*, *damage efficiency*, *tankiness efficiency*, dan *control efficiency*. Keseimbangan tipe *damage* diperoleh dari proporsi *physical*, *magic*, dan *true damage* tiap *hero*, kemudian diukur menggunakan entropi untuk menggambarkan variasi komposisi *damage* dalam tim.

Power spike dihitung dari total nilai dari waktu *early*, *mid*, dan *late*, yang kemudian digabungkan menjadi skor tempo permainan serta rasio *mid-late*. Sinergi diturunkan dari pasangan *hero* yang memiliki tingkat kemenangan di atas rata-rata ketika dimainkan bersama. *Strong weak hero* dihitung dari daftar *hero* yang secara konsisten unggul atau kurang efektif ketika saling berhadapan berdasarkan data *patch* 1.9.68. Fitur META *hero* dibentuk dari skor gabungan tingkat pemilihan, tingkat *banned*, dan tingkat kemenangan. *Patch impact* dihitung dengan memberikan bobot pada perubahan *buff*, *nerf*, dan penyesuaian pada *patch major* yaitu *patch* 1.9.68 dan minor yaitu patch 1.9.68.x.

Daftar *banned hero* tidak dimasukkan sebagai fitur karena prediksi pada fase *draft pick* dilakukan setelah seluruh *hero* selesai dipilih. Dengan demikian, model hanya menerima komposisi akhir yang digunakan kedua tim tanpa mempertimbangkan daftar *banned* sebelumnya. Seluruh fitur pada fase *draft pick* direpresentasikan sebagai nilai selisih antara tim Red dan tim Blue sehingga model dapat menangkap perbandingan kekuatan relatif kedua tim secara langsung.

3) *Pembuatan Fitur In-game*: Statistik in-game diperoleh melalui dokumentasi visual menggunakan

screenshot rekaman pertandingan. Setiap *snapshot* pada menit ke-9, 12, 15, dan 18 diambil dengan menghentikan video pada waktu saat permainan berlangsung untuk memastikan setiap *snapshot* berada pada fase permainan yang setara. Nilai yang dicatat meliputi total *gold*, jumlah *kill*, median *level*, *turret* yang dihancurkan, *turtle*, *lord*, dan indikator objektif lainnya. Seluruh nilai ini diubah menjadi fitur selisih yang merepresentasikan selisih antara tim Red dan Blue.

4) *Pembuatan Dataset Last Snapshot*: Untuk data uji, *last snapshot* ditentukan dengan mengonversi durasi pertandingan ke interval terdekat dari titik waktu *snapshot*. Proses ini menyediakan representasi kondisi permainan menjelang akhir pertandingan meskipun durasi permainan berbeda-beda.

5) *Penanganan Perbedaan Dataset*: Karena data latih dan uji berasal dari *patch* yang berbeda (1.9.68 dan 1.9.91), fitur *draft pick* seperti sinergi, *strong weak*, dan skor META disesuaikan hanya berdasarkan data latih untuk menghindari kebocoran data (*data leakage*). *Patch impact* tetap dihitung berdasarkan *patch* masing-masing agar tetap mencerminkan perubahan kekuatan *hero*, tetapi perhitungan sinergi dan META tidak menggunakan data dari *patch* 1.9.91 agar konsistensi model tetap terjaga.

Perbedaan *patch* ini juga secara alami menimbulkan pergeseran karakteristik data (*domain shift*) antara data latih dan uji. Perubahan nilai *win rate*, popularitas hero, dan pola sinergi *antarhero* pada *patch* 1.9.91 menyebabkan distribusi fitur pada data uji tidak sepenuhnya selaras dengan data latih. Kondisi ini dapat memengaruhi performa model, tetapi sekaligus memberikan kondisi evaluasi yang lebih realistis karena model dinilai dalam situasi transisi *patch* yang memang terjadi pada MSC 2025.

C. Pemodelan

Tahap pemodelan dilakukan dengan melatih algoritma *Random Forest* dan *Extreme Gradient Boosting* menggunakan data latih yang telah melalui pra-proses data. Model dibangun untuk dua fase, yaitu *draft pick* dan *in-game* sehingga total model pada kedua fase berjumlah 4. Selanjutnya, model dilatih menggunakan data latih untuk menghasilkan prediksi pada data uji.

1) *Random Forest*: *Random Forest* Merupakan algoritma klasifikasi berbasis *Supervised Learning* yang dikembangkan oleh Leo Breiman. Algoritma ini membangun beberapa *Decision Tree* dari sampel data acak yang identik dan independen [13], kemudian hasil prediksi ditentukan melalui *majority voting* dari seluruh pohon. Metode ini dikenal mampu menangani jumlah variabel masukan yang besar tanpa menyebabkan *overfitting*, serta efektif dalam mengurangi korelasi antara setiap pohon keputusan karena prinsip kerja menggunakan metode *ensemble* [14].

Menurut Suryanegara dkk [14], alur kerja *Random Forest* dimulai dengan memilih sejumlah fitur secara acak (misalnya 'R' fitur) dari total fitur yang tersedia (m) dengan 'R' jauh lebih kecil dari 'm'. Fitur tersebut ditentukan titik pemisah terbaik untuk membagi data pada tiap simpul, kemudian *node* dibagi menjadi simpul anak. Proses ini diulang hingga tercapai jumlah *node* atau kriteria tertentu, membentuk satu pohon keputusan. Langkah tersebut diulang untuk membentuk sejumlah 'n' pohon, lalu prediksi diperoleh melalui voting mayoritas.

2) *Extreme Gradient Boosting (XGBoost)*: *XGBoost* merupakan algoritma *machine learning* berbasis *gradient boosting* dengan *Decision Tree* sebagai model dasar. Berbeda dengan *Random Forest* yang independen antara setiap pohon, *XGBoost* membangun pohon secara berurutan, di mana setiap pohon baru berfungsi memperbaiki kekurangan pohon sebelumnya [15]. Rumus prediksi model atau $\hat{y}_i$ (*ensemble trees*) *XGBoost* ditunjukkan sebagai berikut [16]:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i),\ f_k \in \mathcal{F} \quad (1)$$

dengan:
$\mathcal{F}$ = himpunan fungsi (ruang) *decision tree*
f_k = fungsi yang dipelajari pada iterasi *boosting* ke-k.

Fungsi objektif *XGBoost* atau $\mathcal{L}$ dirumuskan sebagai:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k),\ \Omega(f) = \gamma T + \frac{1}{2}\lambda |w|^2 \quad (2)$$

dengan:
$l(y_i, \hat{y}_i)$ = fungsi *loss* untuk meminimalkan selisih antarnilai aktual dan prediksi
$\Omega(f_k)$ = suku regularisasi
T = jumlah *leaf* pada sebuah *tree*
w = vektor bobot *leaf*
γ = penalti untuk penambahan sebuah *leaf node*
λ = parameter yang mengontrol kekuatan regularisasi L2

Secara keseluruhan, persamaan prediksi pada Rumus (1) menunjukkan bahwa *XGBoost* membangun model dengan cara menjumlahkan hasil dari sejumlah *Decision Tree* secara bertahap melalui proses *boosting*. Sementara itu, pada Rumus (2) menunjukkan bahwa *XGBoost* berusaha menyeimbangkan antara ketepatan prediksi dan kompleksitas model. *Loss function* bertugas untuk meminimalkan perbedaan antara nilai aktual dan nilai prediksi, sedangkan suku regularisasi berperan mengontrol jumlah *leaf* serta bobotnya untuk mencegah *overfitting*. Dengan demikian, *XGBoost* mampu menghasilkan model prediksi yang akurat tetapi juga terkontrol kompleksitasnya.

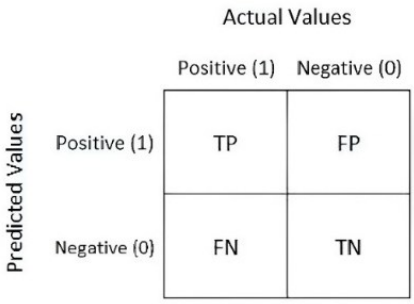
3) *Pembagian Model*: model akan dibagi menjadi empat berdasarkan fase dan algoritma yang ada. Model *in-game* akan dilatih menggunakan data gabungan seluruh *snapshot*, dan fitur snapshot_stage digunakan untuk memberikan konteks waktu kepada model. Sehingga model dapat menangkap perbedaan data antara *snapshot* menit ke-9 dan menit ke-18.

D. Evaluasi

Tahap ini bertujuan untuk menguji performa kedua model menggunakan data uji turnamen MSC 2025 melalui prediksi fase *draft pick* dan *in-game*. Evaluasi dilakukan

menggunakan *confusion matrix* (akurasi, presisi, *recall*, dan F1-score) serta ROC-AUC untuk menilai kemampuan prediksi secara menyeluruh. Pada fase *draft pick*, metrik presisi, *recall*, dan F1-score dihitung dengan *macro average* agar penilaian seimbang pada kedua kelas. Sementara pada fase *in-game*, perbandingan dilakukan di setiap *snapshot*, termasuk *last snapshot*. Hasil evaluasi kedua model kemudian dibandingkan pada tiap fase.

1) *Confusion Matrix*: *Confusion matrix* merupakan tabel yang digunakan untuk menilai performa model klasifikasi dengan membandingkan hasil prediksi model terhadap label aktual pada data uji. Dari *confusion matrix* ini, berbagai metrik evaluasi seperti akurasi, presisi, *recall*, dan F1-score dapat dihitung untuk memahami performa model secara menyeluruh [17].



Gambar 2. Ilustrasi *confusion matrix*[17]

Pada Gambar 2, komponen *confusion matrix* terdiri dari: *True Positive* (TP) yaitu prediksi kelas positif yang benar, *True Negative* (TN) yaitu prediksi kelas negatif yang benar, *False Positive* (FP) yaitu prediksi kelas positif yang salah, dan *False Negative* (FN) yaitu prediksi kelas negatif yang salah. Berdasarkan komponen pada Gambar 2, perhitungan metrik evaluasi dilakukan menggunakan rumus berikut.

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

Pada rumus (3), akurasi menunjukkan rasio jumlah prediksi yang benar terhadap keseluruhan data uji. Nilai akurasi yang tinggi menandakan model memiliki kemampuan umum yang baik dalam melakukan klasifikasi, meskipun belum tentu seimbang antarkelas.

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

Pada rumus (4), presisi mengukur ketepatan model dalam mengidentifikasi kelas positif. Nilai presisi yang tinggi menunjukkan bahwa sebagian besar prediksi positif model memang benar, sehingga *False Positive* relatif rendah.

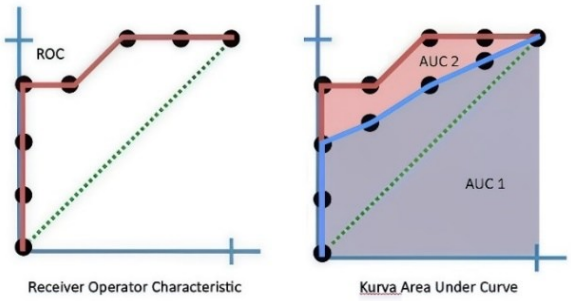
$$\text{Recall} = \frac{TP}{TP+FN} \quad (5)$$

Pada rumus (5), *recall* menunjukkan kemampuan model dalam mengenali seluruh data positif yang sebenarnya. Semakin tinggi *recall*, semakin kecil kemungkinan model gagal mendeteksi data positif atau *False Negative*.

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

Pada rumus (6), F1-score merupakan rata-rata harmonis antara presisi dan *recall* yang digunakan untuk menilai keseimbangan performa model. Nilai F1-score yang tinggi menandakan model tidak hanya tepat dalam prediksi positif, tetapi juga konsisten dalam mendeteksi seluruh data positif.

2) Kurva Receiver Operating Characteristic: Kurva *Receiver Operating Characteristic* (ROC) merupakan grafik yang digunakan untuk mengevaluasi kinerja model klasifikasi biner melalui hubungan antara *True Positive Rate* (TPR) dan *False Positive Rate* (FPR) yang diturunkan dari *confusion matrix*. Luas di bawah kurva ROC disebut *Area Under the Curve* (AUC), yang digunakan untuk membandingkan kinerja algoritma [18] dalam membedakan kelas positif dan negatif.



Gambar 3. Ilustrasi kurva ROC dan AUC [18]

Pada Gambar 3, TPR identik dengan *recall*, yaitu rasio prediksi positif yang benar terhadap total kasus positif. Sedangkan FPR menunjukkan rasio prediksi positif yang salah terhadap total kasus negatif. Nilai AUC berada pada rentang 0-1, semakin tinggi nilainya maka semakin baik kemampuan model dalam membedakan antarkelas [19]. Berbeda dengan akurasi yang menggambarkan proporsi prediksi benar dari keseluruhan data dan sensitif terhadap ketidakseimbangan kelas, ROC-AUC menilai performa model secara menyeluruh tanpa bergantung pada ambang batas tertentu. Oleh karena itu, ROC-AUC memberikan ukuran yang lebih stabil dan representatif untuk mengevaluasi model pada *dataset* yang tidak seimbang.

3) *Feature Importance*: *Feature importance* digunakan untuk mengukur kontribusi setiap fitur terhadap kinerja model dalam memprediksi hasil pertandingan. Metode ini memberikan skor pada masing-masing fitur untuk menunjukkan fitur mana yang paling berpengaruh dan mana yang memiliki dampak rendah terhadap keputusan model. Penilaian dilakukan dengan melihat perubahan performa model ketika suatu fitur dimodifikasi, misalnya melalui teknik permutasi atau evaluasi bobot internal model. Hasil *feature importance* membantu meningkatkan interpretabilitas model serta menjadi dasar untuk memahami faktor-faktor utama yang memengaruhi prediksi pada fase *draft pick* maupun *in-game* [20].

4) *Evaluasi Per-snapshot Fase In-game*: Pada fase *in-game*, evaluasi dilakukan untuk setiap *snapshot* menit ke-9, ke-12, ke-15, ke-18, dan *last snapshot* untuk menilai stabilitas model terhadap perubahan kondisi permainan dari waktu ke waktu.

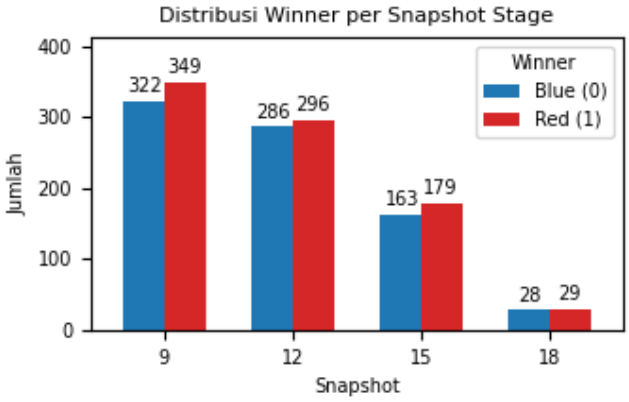
III. HASIL DAN PEMBAHASAN

Penelitian ini menggunakan Google Colabolatory sebagai platform untuk pembuatan model *Random Forest* dan *XGBoost* dalam memprediksi peluang kemenangan pada pertandingan turnamen MSC 2025 dari fase *draft pick* dan *in-game*. Hasil dari kedua model kemudian dibandingkan berdasarkan metrik evaluasi untuk menilai kinerja terbaik dari fase *draft pick* maupun *in-game*.

A. Dataset

Dataset pada fase *draft pick* merupakan kumpulan data dari 6 turnamen kualifikasi dan 1 *main event*, di mana setiap baris merepresentasikan satu permainan individu dalam sebuah pertandingan. Total data mencakup 671 permainan yang berasal dari 553 permainan pada data latih dan 118 permainan pada data uji. Secara keseluruhan terdapat 54 tim yang berpartisipasi pada babak kualifikasi dan 23 tim pada *main event*. Jumlah kelas adalah 322 permainan dimenangkan sisi Blue dan 349 oleh sisi Red, dengan rasio 48:52. Distribusi ini menunjukkan bahwa tim Red lebih sering menang, kemungkinan karena memiliki keunggulan strategis dalam pemilihan *hero* terakhir (*last pick*), yang sering dimanfaatkan untuk memilih *hero* yang kuat terhadap komposisi lawan.

Dataset pada fase *in-game* terdiri atas *snapshot* pada menit ke-9, ke-12, ke-15, dan ke-18 dengan total 1.652 data (1.348 data latih dan 304 data uji) serta rasio data 82:18, sama seperti fase *draft pick*. Jumlah *snapshot* menurun seiring bertambahnya durasi permainan, dengan perincian: menit ke-9 sebanyak 671, menit ke-12 sebanyak 582, menit ke-15 sebanyak 342, dan menit ke-18 sebanyak 57. Distribusi rasio pemenang tiap *snapshot* pada gabungan data latih dan data uji ditunjukkan pada Gambar 4.



Gambar 4. Distribusi tiap *snapshot* pada data latih dan data uji

Gambar 4 memperlihatkan bahwa jumlah *snapshot* menurun seiring berjalannya permainan. Sebelum menit ke-18, jumlah pemenang dari sisi Red lebih banyak dibandingkan Blue, sedangkan pada menit ke-18 hasilnya hampir seimbang. Kondisi ini menunjukkan bahwa setelah menit ke-9, tim yang berhasil menghancurkan *turret* atau mendapatkan *lord* memiliki peluang besar untuk menang. Namun, ketika permainan berlangsung lebih lama, peluang membalikkan keadaan meningkat karena kesalahan kecil dapat berpengaruh besar terhadap hasil pertandingan, terutama pada menit ke-18 dan seterusnya.

B. Pra-proses Data

Tahap pra-proses data dilakukan untuk menyiapkan *dataset* yang siap digunakan pada pembangunan model prediksi. Proses *feature engineering* dilakukan secara terpisah pada dua fase, yaitu *draft pick* dan *in-game* untuk menghasilkan fitur-fitur yang mampu merepresentasikan karakteristik strategi dan dinamika selama permainan.

1) *Feature Engineering Fase Draft Pick*: *Feature engineering* pada fase ini bertujuan membentuk fitur yang merepresentasikan kekuatan, sinergi, serta skor tingkat META dari masing-masing tim berdasarkan susunan *hero* yang dipilih. Data *rating hero*, statistik *patch* 1.9.68, *power spike*, dan *patch impact* digunakan sebagai dasar pembentukan fitur *draft pick* agar model mampu menangkap perubahan meta dan kekuatan *hero* pada kedua tim. Fitur yang dikonstruksi menghasilkan 12 fitur untuk model pada fase *draft pick*, sebagaimana ditunjukkan pada Tabel 1. Seluruh fitur merupakan hasil perhitungan selisih (*difference*) antara tim Red dan Blue, dengan penamaan diawali awalan diff_ untuk menunjukkan arah perbandingan kedua sisi.

TABEL I
DAFTAR FITUR YANG DIGUNAKAN PADA FASE DRAFT PICK

Nama Fitur	Keterangan
aggressiveness	Rasio daya serang terhadap daya tahan untuk menunjukkan gaya bermain agresif
damage_efficiency	Perbandingan antara daya serang dan tingkat kesulitan *hero*
tankiness_efficiency	Perbandingan antara daya tahan dan tingkat kesulitan *hero*
control_efficiency	Perbandingan antara efek kontrol dan tingkat kesulitan *hero*
damage_balance	Keseimbangan antara *physical*, *magic*, dan *true damage*
power_spike	Waktu peningkatan kekuatan *hero* selama permainan
mid_late_ratio	Rasio kekuatan tim pada waktu *mid* hingga *late game*
tier_score	Skor tingkat META *hero* berdasarkan data latih
synergy_score	Skor sinergi antar-*hero* dalam satu tim
strong_score	Skor kekuatan tim terhadap komposisi lawan
weak_score	Skor kelemahan tim terhadap komposisi lawan
patch_impact_score	Skor pengaruh *patch* terhadap performa *hero*

2) *Feature Engineering Fase In-game*: *Feature engineering* pada fase *in-game* dilakukan dengan memanfaatkan data *snapshot* permainan pada menit ke-9, ke-12, ke-15, ke-18 dan *last snapshot* untuk merepresentasikan kondisi kedua tim. Nama dan makna fitur ini diadaptasi dari artikel oleh Rebeca Chinicz [21], yang menggunakan indikator statistik seperti *gold*, *kills*, *experience*, dan

objectives untuk memprediksi hasil pertandingan pada *game* League of Legends. Setiap indikator ditransformasikan menjadi nilai selisih antara tim Red dan Blue agar model dapat menangkap perbandingan performa antartim secara objektif. Proses ini menghasilkan 14 fitur yang mencerminkan kondisi permainan pada setiap *snapshot*, sebagaimana ditunjukkan pada Tabel 2. Seluruh fitur yang bersifat statistik diberi awalan diff_ untuk menandakan nilai selisih antara tim Red dan Blue.

TABEL II
DAFTAR FITUR YANG DIGUNAKAN PADA FASE IN-GAME

Nama Fitur	Keterangan
snapshot_stage	Menunjukkan menit pengambilan *snapshot*.
gold	total *gold* tim
kills	total *kill* tim
med_kills	median *kill* per *hero* dalam tim
med_level	median *level* per *hero* dalam tim
first_blood	Menandai tim yang mendapatkan *kill* pertama
first_turret	Menandai tim yang menghancurkan *turret* pertama
turret	Total *turret* yang dihancurkan tim
first_inhib	Menandai tim yang menghancurkan *turret inhibitor* pertama
inhibs	Total *turret inhibitor* yang dihancurkan tim
first_turtle	Menandai tim yang mendapat *turtle* pertama
turtles	Total *turtle* yang diperoleh tim
first_lord	Menandai tim yang mendapat *lord* pertama
lords	Total *lord* yang diperoleh tim

C. Implementasi Model

Tahap ini merupakan pembuatan dan pelatihan model *Random Forest* dan *XGBoost* menggunakan data latih dari kualifikasi turnamen MSC 2025.

1) *Model Fase Draft Pick*: Model fase *draft pick* dilatih menggunakan 553 data latih. Pada model *Random Forest*, digunakan parameter n_estimators=100, max_depth=4, dan random_state=28. Nilai max_depth=4 diterapkan untuk membatasi kedalaman pohon agar model tidak mengalami *overfitting*, sedangkan random_state=28 digunakan untuk memastikan hasil pelatihan yang konsisten.

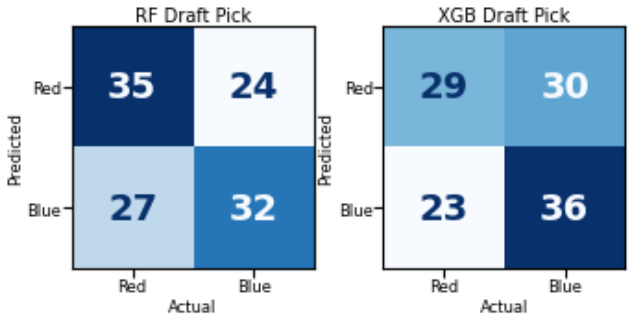
Model *XGBoost* menggunakan parameter n_estimators=100, learning_rate=0.05, max_depth=4, colsample_bytree=0.8, subsample=0.85, random_state=28, dan eval_metric="logloss". Parameter random_state, n_estimators, dan max_depth disamakan dengan *Random Forest* agar hasil kedua model dapat dibandingkan secara seimbang. Nilai learning_rate=0.05 dipilih agar proses pembelajaran berlangsung stabil, sementara colsample_bytree=0.8 dan subsample=0.85 digunakan untuk meningkatkan keragaman pohon keputusan dan mengurangi risiko *overfitting*. Parameter eval_metric="logloss" digunakan untuk menilai kualitas prediksi probabilitas pada klasifikasi biner.

2) *Model Fase In-game*: Model fase *in-game* dilatih menggunakan 1.348 data latih yang mencakup seluruh *snapshot* pada menit ke-9, 12, 15, dan 18. Penggabungan seluruh *snapshot* dilakukan untuk mengurangi jumlah model sekaligus memungkinkan model mengenali konteks waktu melalui fitur snapshot_stage. Model *Random Forest* dan *XGBoost* menggunakan konfigurasi parameter yang serupa dengan model pada fase *draft pick*, tetapi dengan random_state = 9 agar proses pelatihan tidak menghasilkan pola acak yang identik dengan model sebelumnya.

D. Evaluasi dan Analisis Kinerja Model

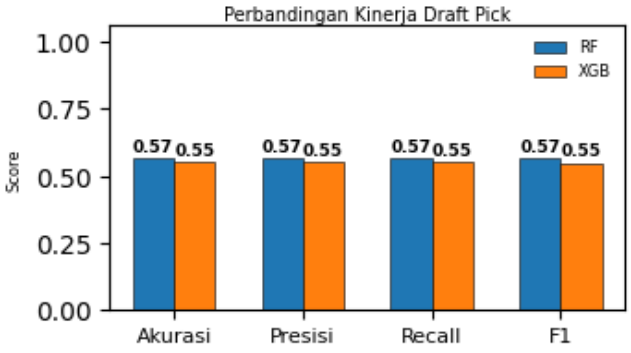
Evaluasi model *Random Forest* dan *XGBoost* pada fase *draft pick* dan *in-game* dilakukan untuk menilai kemampuan prediksi kemenangan menggunakan data uji dari turnamen MSC 2025. Proses evaluasi menggunakan metrik *confusion matrix* yang mencakup metrik akurasi, presisi, *recall*, dan F1-score, serta metrik ROC-AUC untuk menilai kemampuan model dalam membedakan kelas Blue dan Red.

1) *Model Fase Draft Pick*: Model fase *draft pick* dievaluasi menggunakan 118 data uji MSC 2025 dengan rasio kelas yang seimbang (50% Blue dan 50% Red) dan mendapatkan *confusion matrix* yang dapat dilihat pada Gambar 5.



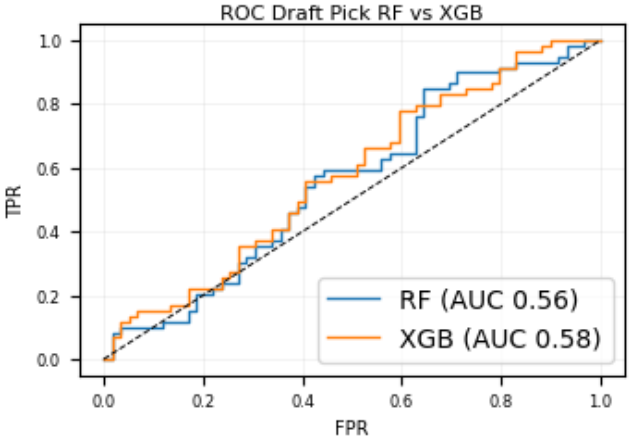
Gambar 5. *Confusion matrix* model *Random Forest* dan *XGBoost* pada fase *draft pick*

Berdasarkan Gambar 5, model *Random Forest* dan *XGBoost* menunjukkan kemampuan yang relatif seimbang dalam memprediksi hasil pertandingan pada fase *draft pick*. *Random Forest* menghasilkan 35 prediksi benar untuk tim Red dan 32 untuk tim Blue, sedangkan *XGBoost* mencatat 29 prediksi benar untuk tim Red dan 36 untuk tim Blue. Meskipun keduanya memiliki tingkat kesalahan yang sebanding, *Random Forest* tampak lebih konsisten dalam mengenali kemenangan tim Red, sementara *XGBoost* sedikit lebih baik dalam mendeteksi kemenangan tim Blue. Perbedaan ini kemungkinan dipengaruhi oleh parameter dan jumlah data masing-masing model atau mekanisme pembelajaran, di mana *Random Forest* lebih stabil terhadap variasi data, sedangkan *XGBoost* cenderung lebih adaptif terhadap pola yang dominan.



Gambar 6. Perbandingan kinerja model *Random Forest* dan *XGBoost* fase *draft pick*

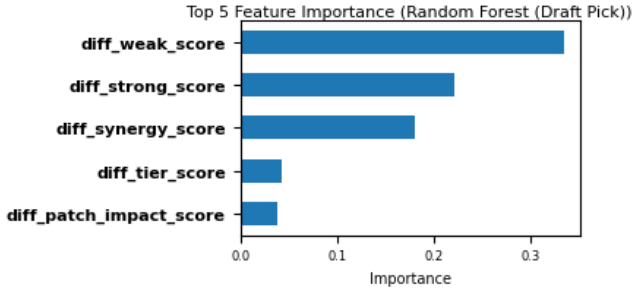
Terlihat pada Gambar 6 bahwa *Random Forest* unggul pada seluruh metrik utama, dengan akurasi dan *nilai macro average* untuk presisi, *recall*, dan F1-score berada di nilai 57%, sedangkan *XGBoost* memperoleh nilai 55% pada semua metrik. Hasil ini menunjukkan bahwa meskipun *XGBoost* memiliki mekanisme *boosting* yang kuat, *Random Forest* lebih stabil dalam memodelkan pola kemenangan pada data fase *draft pick*.



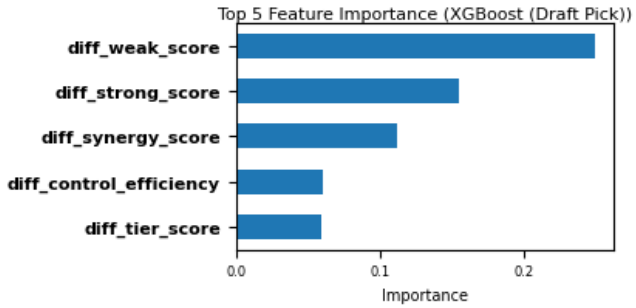
Gambar 7. Kurva ROC-AUC antara model *Random Forest* dan *XGBoost* pada fase *draft pick*

Sebagaimana terlihat pada Gambar 7, nilai AUC *XGBoost* sebesar 0,58, sedikit lebih tinggi dibandingkan *Random Forest* sebesar 0,56. Meskipun akurasi *XGBoost* lebih rendah 3%, nilai AUC yang lebih tinggi menunjukkan kemampuannya membedakan kelas positif dan negatif secara lebih konsisten. Hal ini dapat disebabkan oleh sifat *XGBoost* yang lebih adaptif terhadap distribusi probabilitas kelas dan parameter pembelajaran, sementara *Random Forest* cenderung menghasilkan prediksi seimbang namun kurang optimal dalam kalibrasi probabilitas.

Nilai AUC model *XGBoost* pada fase *draft pick* sebesar 0,58 masih lebih rendah dibandingkan hasil penelitian oleh Putro dkk. [6] yang mencapai nilai AUC 0,67 menggunakan *Decision Tree* dengan fitur berbasis daftar *hero*. Perbedaan ini kemungkinan disebabkan oleh jumlah data latih dan data uji yang berbeda pada penelitian terdahulu serta variasi kompleksitas fitur pada penelitian. Selain itu, strategi *draft pick* pada periode penelitian Putro dkk. kemungkinan lebih mudah diprediksi karena META permainan saat itu relatif stabil karena adanya *hero-hero* kuat yang sering dipilih dan secara langsung memengaruhi peluang kemenangan.



Gambar 8. Feature Importance model *Random Forest* pada fase *draft pick*



Gambar 9. Feature Importance model *XGBoost* pada fase *draft pick*

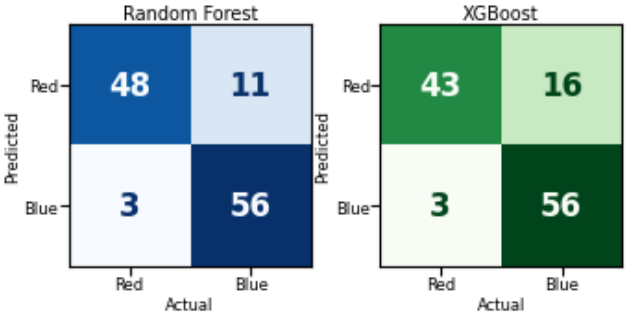
Gambar 8 dan Gambar 9 menunjukkan bahwa diff_weak_score, diff_strong_score, dan diff_synergy_score secara konsisten menempati tiga kontribusi tertinggi pada kedua model. Ketiga fitur tersebut merepresentasikan kualitas komposisi tim terhadap komposisi lawan. Fitur diff_weak_score menangkap komposisi *hero* yang lemah dari tim terhadap *hero* yang digunakan lawan. Nilai *importance* yang tinggi menunjukkan bahwa kelemahan *draft pick* lebih mudah dieksploitasi dan berpengaruh langsung terhadap peluang kemenangan. Fitur diff_strong_score menggambarkan komposisi *hero* yang kuat dari tim ke *hero* lawan. Sementara itu, fitur dif_synergy_score menunjukkan bahwa kecocokan interaksi kemampuan antara tiap *hero* dalam satu tim tetap menjadi faktor penentu dalam meningkatnya peluang kemenangan.

Fitur diff_tier_score memiliki kontribusi lebih rendah, tetapi tetap relevan. Hal ini menunjukkan bahwa META dari *patch* data latih masih ditangkap oleh model, meskipun pengaruhnya tidak sebesar kualitas komposisi tim secara langsung. Sementara itu diff_patch_impact_score yang berada di urutan terbawah mengindikasikan bahwa perubahan *patch* tidak memberikan dampak prediktif yang kuat pada *draft pick*, sehingga komposisi *hero* lebih berperan dibanding perubahan kekuatan individual *hero* dalam *patch* tersebut.

Akurasi fase *draft pick* yang hanya mencapai 57% berasal dari sifat fase ini yang memang jauh lebih tidak stabil dibandingkan fase *in-game*. Komposisi *hero* sangat

dipengaruhi oleh daftar *hero* yang terkena *banned*, sehingga kombinasi yang muncul pada data uji sering kali tidak identik dengan pola sinergi serta *strong weak hero* yang dipelajari pada data latih. *Banned hero* sengaja tidak dimasukkan sebagai fitur karena prediksi dilakukan setelah komposisi final terbentuk, namun absennya fitur ini tetap menyebabkan ketidakselarasan pola *draft pick* dan menurunkan konsistensi model. Selain itu, *draft pick* hanya memuat strategi awal tanpa menangkap performa pemain, penguasaan objektif, serta alur perolehan *gold* yang justru menentukan hasil akhir pertandingan. Perbedaan antara *patch* 1.9.68 dan 1.9.91 juga memperbesar pergeseran karakteristik data, membuat strategi META pada data latih dan data uji tidak sepenuhnya selaras. Faktor-faktor tersebut menjadikan *draft pick* secara mendasar lebih sulit diprediksi dibandingkan fase *in-game* yang menggunakan indikator statistik langsung dari kondisi permainan.

2) *Model Fase In-game*: Model pada fase *in-game* dievaluasi menggunakan *Random Forest* dan *XGBoost* berdasarkan *snapshots* pada menit ke-9, ke-12, ke-15, ke-18, dan *last snapshot* untuk mengamati perubahan performa model seiring berjalannya permainan. Model *Random Forest* memperoleh akurasi sebesar 76%, 83%, 75%, 72%, dan 88%, sedangkan *XGBoost* memperoleh 75%, 83%, 76%, 66%, dan 84% pada *snapshot* yang sama. Tren metrik presisi, *recall*, dan F1-score dari *macro average* mengikuti pola akurasi untuk kedua model, sebagaimana ditunjukkan pada Tabel III. Berdasarkan Tabel III, performa kedua model berfluktuasi seiring lamanya permainan. Akurasi tertinggi pada menit ke-12 menunjukkan bahwa waktu *mid game* merupakan titik paling stabil untuk memprediksi hasil pertandingan karena keuntungan sejak *early game* masih berpengaruh kuat terhadap arah permainan. Setelah melewati menit ke-12, tingkat ketidakpastian meningkat karena kesalahan kecil dapat mengubah kondisi dengan cepat, sebagaimana terlihat pada penurunan performa di menit ke-18. Untuk menilai performa akhir model secara lebih jelas tanpa menampilkan *confusion matrix* pada setiap *snapshot*, analisis difokuskan pada *last snapshot* yang merepresentasikan kondisi permainan menjelang akhir permainan.

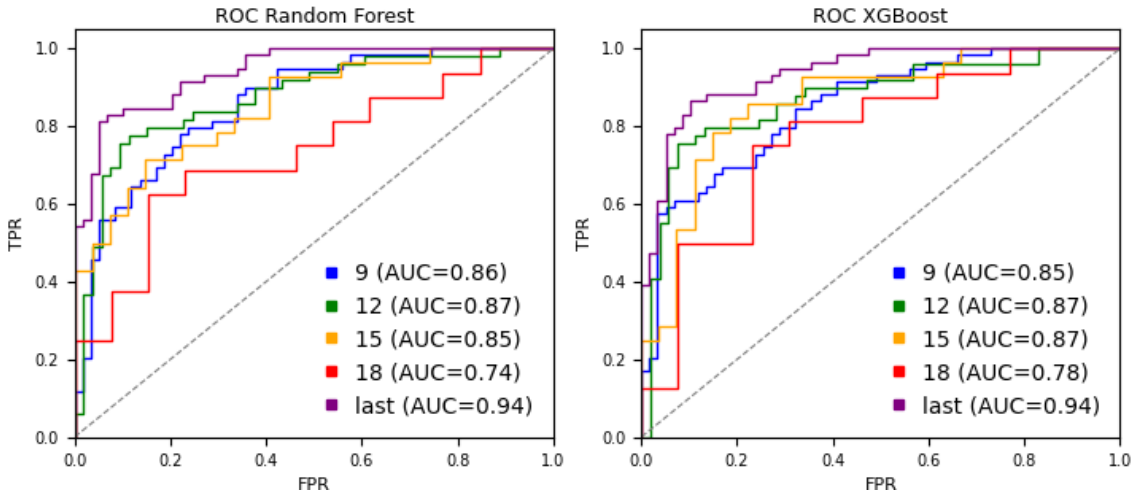


Gambar 10. *Confusion matrix* model *Random Forest* dan *XGBoost* pada fase *in-game* dari evaluasi *last snapshot*

Pada Gambar 10 menunjukkan bahwa c*onfusion matrix* pada *last snapshot*, bahwa akurasi kedua model relatif serupa meskipun *Random Forest* sedikit lebih tinggi (88%) dibandingkan *XGBoost* (84%). Keunggulan *Random Forest* disebabkan oleh kemampuannya menangkap pola non-linear dan interaksi *antarfitur* secara stabil, sehingga lebih tahan terhadap variasi data. Nilai akurasi tinggi pada *last snapshot* wajar terjadi karena data pada tahap ini merepresentasikan kondisi permainan menjelang akhir, di mana hasil pertandingan lebih mudah diprediksi berdasarkan selisih statistik antar tim.

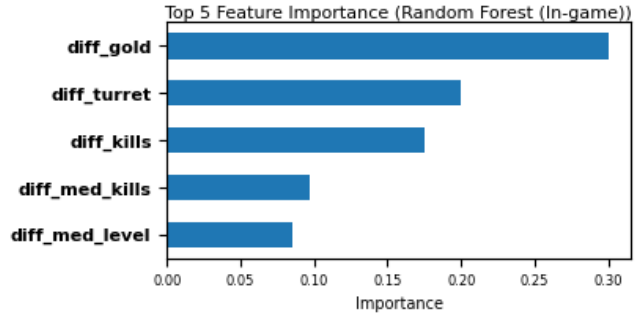
TABEL III
KINERJA MODEL *RANDOM FOREST* DAN *XGBOOST* PADA FASE *IN-GAME*

Snapshot	***Random Forest***				***XGBoost***			
	Akurasi	**Presisi**	***Recall***	**F1-score**	**Akurasi**	**Presisi**	***Recall***	**F1-score**
9	76%	76%	76%	76%	75%	76%	75%	74%
12	**83%**	84%	83%	83%	**83%**	84%	83%	83%
15	75%	75%	75%	75%	76%	78%	77%	76%
18	72%	74%	74%	72%	66%	72%	68%	64%
Last Snapshot	**88%**	89%	88%	88%	**84%**	86%	84%	84%

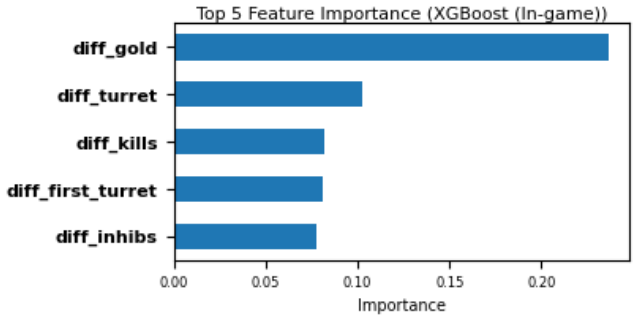


Gambar 11. Kurva ROC model *Random Forest* dan *XGBoost* pada fase *in-game* untuk tiap *snapshot*

Berdasarkan Gambar 11, nilai AUC kedua model naik dari menit ke-9 ke menit ke-12 lalu turun kembali pada menit ke-15 dan ke-18. *Random Forest* mencatat AUC 0,86 pada menit ke-9, meningkat menjadi 0,87 pada menit ke-12, kemudian menurun ke 0,85 dan 0,74 pada menit ke-15 dan ke-18. *XGBoost* memiliki pola serupa dengan AUC 0,85 dan 0,87 pada menit ke-9 dan ke-12, namun tetap stabil pada menit ke-15 dengan 0,87 dan menurun menjadi 0,78 pada menit ke-18. *Random Forest* hanya unggul pada menit ke-12 sementara *XGBoost* lebih stabil pada waktu *late game* berkat mekanisme *boosting* yang lebih adaptif terhadap kondisi permainan yang semakin kompleks. Pada *last snapshot*, kedua mencapai nilai AUC tertinggi yang sama, yaitu 0,94. Hasil ini menegaskan bahwa data statistik menjelang akhir permainan lebih informatif sehingga model lebih mudah membedakan tim yang menang dan kalah.



Gambar 12. *Feature Importance* model *Random Forest* pada fase *in-game*



Gambar 13. *Feature Importance* model *XGBoost* pada fase *in-game*

Gambar 12 dan Gambar 13 menunjukkan bahwa diff_gold menjadi fitur paling dominan pada kedua model. Selisih *gold* mencerminkan kontrol tempo permainan karena *gold* menentukan kecepatan perolehan item dan kekuatan *hero*, sehingga menjadi indikator paling langsung terhadap peluang kemenangan. Fitur diff_turret menempati urutan berikutnya, sejalan dengan sifat *game* dengan genre MOBA sebagai *game* berbasis objektif. Keunggulan *turret* membuka ruang map, mempercepat perolehan *gold*, dan mempermudah pengambilan objektif utama sehingga dampaknya lebih stabil dibanding perolehan *kill*.

Fitur diff_kills berada di posisi selanjutnya dan berkontribusi terutama melalui pergeseran selisih *gold* serta momentum *team fight*, meskipun tidak selalu menghasilkan keuntungan objektif. Pada model *Random Forest*, median *kill* dan median *level* menunjukkan keunggulan konsistensi kontribusi tim dan keunggulan *level*. Pada *XGBoost*, fitur seperti diff_first_turret dan diff_inhibs masuk dalam lima besar, menandakan pentingnya *turret* awal dan pengambilan *turret* terakhir sebelum *base turret*.

Secara keseluruhan, hasil ini menegaskan bahwa pada fase *in-game*, variabel seperti *gold* dan *turret* memberikan kontribusi prediktif tertinggi, sementara variabel seperti *kill* lebih bersifat pendukung. Pola ini konsisten dengan dinamika *game* bergenre MOBA kompetitif yang menempatkan kontrol map dan objektif sebagai penentu utama hasil pertandingan.

Jika dibandingkan dengan penelitian oleh Sena dan Emanuel [3] yang menggunakan fitur statistik seperti *average gold*, *average level*, *total kill*, *first blood*, *first turtle*, dan *first lord*, akurasi tertinggi diraih oleh model *Artificial Neural Network* dan *Random Forest* yang masing-masing adalah 82% dan 80%. Nilai tersebut lebih rendah dibandingkan hasil pada penelitian ini, di mana model *Random Forest* dan *XGBoost* memperoleh akurasi 88% dan 84% pada *last snapshot*. Perbandingan dilakukan pada *last snapshot* karena tahap ini paling mendekati data historis pasca pertandingan yang digunakan dalam penelitian tersebut. Perbedaan hasil kemungkinan disebabkan oleh penggunaan fitur yang lebih kompleks dan data *snapshot* yang merepresentasikan kondisi permainan secara bertahap, sehingga memberikan konteks yang lebih akurat terhadap dinamika pertandingan dan meningkatkan ketepatan prediksi.

E. Perbandingan Kinerja Model Random Forest dan XGBoost pada Fase Draft Pick dan In-game

Pada fase *draft pick*, *Random Forest* sedikit unggul di seluruh metrik utama dengan selisih 2%, menunjukkan kemampuannya menangkap hubungan non-linear antarfitur dengan lebih stabil. Sementara itu, *XGBoost* mencatat nilai ROC-AUC lebih tinggi (0,58 dibanding 0,56), menandakan keunggulan dalam membedakan kelas kemenangan secara probabilistik. Dengan demikian, *Random Forest* lebih kuat dalam kestabilan prediksi, sedangkan *XGBoost* unggul dalam kalibrasi probabilitas. Hasil evaluasi model *Random Forest* dan *XGBoost* pada fase *draft pick* ditunjukkan pada Tabel IV.

TABEL IV
PERBANDINGAN KINERJA MODEL *RANDOM FOREST* DAN *XGBOOST* PADA FASE *DRAFT PICK*

Metrik Evaluasi	Model	
	Random Forest	***XGBoost***
Akurasi	**57%**	55%
Presisi	57%	55%
Recall	57%	55%
F1-Score	57%	55%
ROC-AUC	0,56	**0,58**

Pada fase *in-game*, performa kedua model berfluktuasi mengikuti dinamika permainan. Akurasi dan ROC-AUC meningkat pada waktu *mid game* (menit ke-9 hingga ke-12), lalu menurun pada waktu *late game* (menit ke-15 hingga ke-18 dan seterusnya) sebelum mencapai nilai tertinggi pada *last snapshot* (88% untuk *Random Forest* dan 84% untuk *XGBoost*, dengan ROC-AUC 0,94 pada keduanya). Pola ini menunjukkan bahwa data menjelang akhir permainan sudah sangat menentukan hasil, sehingga perbedaan performa lebih disebabkan oleh dominasi statistik tim pemenang dibanding peningkatan generalisasi model. Rangkuman hasil evaluasi kedua model pada fase in-*game* ditampilkan pada Tabel V.

TABEL V
PERBANDINGAN KINERJA MODEL *RANDOM FOREST* DAN *XGBOOST* PADA FASE *IN-GAME*

Snapshot	**RF Akurasi**	**XGB Akurasi**	**RF ROC-AUC**	**XGB ROC-AUC**
9	76%	75%	0.86	0.85
12	83%	83%	0.87	0.87
15	75%	76%	0.85	0.87
18	72%	66%	0.74	0.78
Last Snapshot	**88%**	**84%**	**0.94**	**0.94**

Secara keseluruhan, hasil perbandingan menunjukkan bahwa kedua model memiliki karakteristik yang saling melengkapi. *Random Forest* menunjukkan performa yang lebih stabil baik pada fase *draft pick* maupun *in-game*, sementara *XGBoost* unggul dalam konsistensi probabilitas dan efektivitas adaptasi terhadap perubahan kondisi permainan.

Meskipun MSC 2025 merupakan turnamen internasional, model dalam penelitian ini tetap belum diuji untuk generalisasi lintas *event* maupun lintas *patch*. Pola permainan dan preferensi *hero* pada turnamen MSC 2025 sangat dipengaruhi META serta *patch* secara spesifik, sehingga distribusi fiturnya tidak mewakili variasi gaya bermain turnamen lain.

Selain itu, performa model pada penelitian ini sangat dipengaruhi distribusi fitur pada babak kualifikasi karena fitur pada model *draft pick* dihitung berdasarkan statistik *patch* 1.9.68. Ketika *patch* berubah, struktur sinergi, popularitas hero, serta daftar *strong* dan *weak antarhero* ikut berubah sehingga pola hubungan yang dipelajari model menjadi kurang valid. Hal ini berpotensi menghasilkan penurunan performa jika model diterapkan pada *patch* atau turnamen yang berbeda. Pada konteks lintas turnamen, variasi gaya bermain, perbedaan META antarwilayah, serta perbedaan tingkat kompetitif tim dapat menggeser distribusi statistik *in-game*, yang berisiko menimbulkan pergeseran karakteristik data dan membuat model kurang relevan dalam menangkap kemenangan yang umum. Dengan demikian, keterbatasan generalisasi model merupakan konsekuensi langsung dari ruang lingkup *dataset* yang hanya mencakup satu turnamen dan satu rentang *patch* tertentu.

IV. KESIMPULAN

Penelitian ini memprediksi hasil pertandingan Mobile Legends: Bang Bang pada fase *draft pick* dan *in-game* menggunakan algoritma *Random Forest* dan *XGBoost* dengan data turnamen MSC 2025. Pada fase *draft pick*, akurasi tertinggi hanya mencapai 57% sehingga komposisi *hero* belum cukup mampu untuk menjelaskan hasil akhir pertandingan secara akurat. Pada fase *in-game*, performa model meningkat signifikan dengan akurasi pada menit ke-12 sebesar 83% dan akurasi tertinggi pada *last snapshot* sebesar 88% untuk *Random Forest* dan 84% untuk *XGBoost*, disertai nilai ROC-AUC sebesar 0,94 pada kedua model. Hasil ini

indikator statistik *in-game* memberikan informasi yang lebih kuat dibandingkan fitur pada *draft pick* yang hanya mencerminkan potensi peluang kemenangan dari strategi awal. Temuan ini tetap terbatas pada konteks turnamen MSC 2025 sehingga belum merepresentasikan variasi META pada turnamen lain dengan *patch* atau kondisi kompetitif yang berbeda.

Pendekatan pemisahan dua fase melalui *feature engineering* yang disesuaikan karakteristik masing-masing fase terbukti efektif untuk mengevaluasi kontribusi informasi pada setiap tahap permainan. Pendekatan ini tidak dirancang sebagai sistem prediksi *real-time*, melainkan sebagai evaluasi berbasis *snapshot* untuk menilai seberapa informatif kondisi permainan pada titik waktu tertentu.

Pada penelitian selanjutnya, pendekatan gabungan model seperti *ensemble stacking* dapat dieksplorasi untuk meningkatkan performa prediksi. Selain itu, penggunaan data *snapshot* sejak menit awal permainan berpotensi memberikan gambaran yang lebih komprehensif mengenai perubahan dinamika pertandingan. Pengembangan lebih lanjut juga dapat dilakukan melalui model pembelajaran mendalam (*deep learning*) yang lebih sensitif terhadap pola perubahan kondisi permainan secara lebih detail.

DAFTAR PUSTAKA

[1] B. R. Irwanto dan N. W. Wuryandari, "Character Transformation In The Game Mobile Legends: Bang Bang! (MLBB) And Its Impact," *Bambuti : Bahasa Mandarin dan Kebudayaan Tiongkok*, vol. 5, no. 2, hlm. 1–28, Nov 2023, doi: 10.53744/bambuti.v5i2.84.

[2] M. E. Yulianto dan Y. Kristian, "Utilization of MLP and LSTM Methods in Hero Selection Recommendations for the Game of Mobile Legends: Bang Bang," *Teknika*, vol. 14, no. 1, hlm. 142–149, Mar 2025, doi: https://doi.org/10.34148/teknika.v14i1.1201.

[3] I. G. W. Sena dan A. W. R. Emanuel, "Mobile Legend Game Prediction Using Machine Learning Regression Method," *Jurnal Teknologi dan Sistem Informasi*, vol. 9, no. 2, hlm. 221–230, Mar 2023, doi: 10.33330/jurteksi.v9i2.1866.

[4] M. A. Tamaza, S. Defit, dan S. Sumijan, "Implementasi Naïve Bayes dalam M-Series 4 Mobile Legends untuk Prediksi Kemenangan," *Computer Science and Information Technology*, vol. 5, no. 1, hlm. 205–214, Mei 2024, doi: 10.37859/coscitech.v5i1.6707.

[5] D. Sulaiman dan W. S. Utami, "Rancang Bangun Sistem Rekomendasi Pemilihan Hero Pada Game Mobile Legends Menggunakan Algoritma Greedy," *Journal of Information Technology and Computer Science*, vol. 6, no. 2, hlm. 998–1007, Nov 2023, doi: 10.31539/intecoms.v6i2.8128.

[6] Y. N. R. Putro, A. Afriansyah, dan R. Bagaskara, "Penggunaan Algoritma Gaussian Naïve Bayes & Decision Tree Untuk Klasifikasi Tingkat Kemenangan Pada Game Mobile Legends," *Jurnal Komputer dan Informatika*, vol. 6, no. 1, hlm. 10–26, Mei 2024, doi: 10.53842/juki.v6i1.472.

[7] M. Hamir dan P. N. Andono, "Model Neural Network untuk Memprediksi Tingkat Kemenangan Berdasarkan Draft Pick Mobile Legends," *Jurnal Pendidikan dan Teknologi Indonesia*, vol. 5, no. 4, hlm. 899–908, Apr 2025, doi: 10.52436/1.jpti.723.

[8] S. Chowdhury, M. Ahsan, dan P. Barraclough, "Applications of Linear and Ensemble-Based Machine Learning for Predicting Winning Teams in League of Legends," *Applied Sciences*, vol. 15, no. 10, hlm. 5241, Mei 2025, doi: 10.3390/app15105241.

[9] A. A. Kamal, Mohd. A. Mansor, L. Truna, N. Mohd. Shaipullah, dan N. H. H. Sultan, "Machine Learning Applications in Multiplayer Online Battle Arena Esports—A Systematic Review," *JST*, vol. 33, no. 2, Mar 2025, doi: https://doi.org/10.47836/pjst.33.2.11.

[10] Stanlly, F. A. Putra, dan N. N. Qomariyah, "DOTA 2 Win Loss Prediction from Item and Hero Data with Machine Learning," dalam *2022 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, BALI, Indonesia: IEEE, Jul 2022, hlm. 204–209. doi: 10.1109/IAICT55358.2022.9887525.

[11] J. Tyran dan L. Chomatek, "Influence of outliers in MOBA games winner prediction," dalam *Procedia Computer Science*, Okt 2021, hlm. 1973–1981. doi: 10.1016/j.procs.2021.08.203.

[12] A. P. Putra dan P. N. Andono, "Win Probability of Heroes in Mobile Legends MPL ID S12 Competitions Using Naïve Bayes," *Jurnal Media Informatika Budidarma*, vol. 8, no. 1, hlm. 203–210, Jan 2024, doi: 10.30865/mib.v8i1.7185.

[13] IBM, "Apa Itu Random Forest? | IBM," IBM. Diakses: 7 Agustus 2025. [Daring]. Tersedia pada: https://www.ibm.com/id-id/think/topics/random-forest

[14] G. A. B. Suryanegara, A. Adiwijaya, dan M. D. Purbolaksono, "Peningkatan Hasil Klasifikasi pada Algoritma Random Forest untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 1, hlm. 114–122, Feb 2021, doi: 10.29207/resti.v5i1.2880.

[15] D. Kurnia, M. I. Mazdadi, D. Kartini, R. A. Nugroho, dan F. Abadi, "Seleksi Fitur dengan Particle Swarm Optimization pada Klasifikasi Penyakit Parkinson Menggunakan XGBoost," *JTIIK*, vol. 10, no. 5, hlm. 1083–1094, Okt 2023, doi: 10.25126/jtiik.20231057252.

[16] J. Lee dan N. Kim, "Development of Machine Learning-Based Indicators for Predicting Comeback Victories Using the Bounty Mechanism in MOBA Games," *Electronics*, vol. 14, no. 7, hlm. 1445, Apr 2025, doi: 10.3390/electronics14071445.

[17] L. Hakim, A. Sobri, L. Sunardi, dan D. Nurdiansyah, "Prediksi penyakit jantung berbasis mesin learning dengan menggunakan metode k-nn," vol. 07, no. 02, hlm. 14–20, Sep 2024, doi: 10.32502/digital.v7i2.9429.

[18] K. Kristiawan dan A. Widjaja, "Perbandingan Algoritma Machine Learning dalam Menilai Sebuah Lokasi Toko Ritel," *JuTISI*, vol. 7, no. 1, Apr 2021, doi: 10.28932/jutisi.v7i1.3182.

[19] R. Harahap, M. Irpan, M. A. Dinata, dan L. Efrizoni, "Perbandingan Algoritma Random Forest Dan Xgboost Untuk Klasifikasi Penyakit Paru-Paru Berdasarkan Data Demografi Pasien," *Jurnal Ilmiah Betrik*, vol. 15, no. 02, hlm. 130–141, Agu 2024.

[20] A. M. Musolf, E. R. Holzinger, J. D. Malley, dan J. E. Bailey-Wilson, "What makes a good prediction? Feature importance and beginning to open the black box of machine learning in genetics," *Hum Genet*, vol. 141, no. 9, hlm. 1515–1528, Sep 2022, doi: 10.1007/s00439-021-02402-z.

[21] R. Chinicz, "Practical Machine Learning with LoL: a Simple Predictive Use-Case with Data Collection, Learning…," Medium. Diakses: 5 Oktober 2025. [Daring]. Tersedia pada: https://levelup.gitconnected.com/practical-machine-learning-with-lol-a-simple-predictive-use-case-with-data-collection-learning-c2b6e621df66