BAB 2

TINJAUAN PUSTAKA

2.1 Analisis Sepak Bola

Analisis sepak bola merupakan proses yang kompleks dan melibatkan berbagai aspek dari permainan yang saling terkait. Secara mendasar, analisis ini mencakup pengukuran komunikasi antar pemain, kemampuan adaptasi, tempo permainan, serta evaluasi taktik penyerangan dan pertahanan (Mclean *et al*., 2017). Analisis ini memperhitungkan dimensi sosial dan teknis dalam sepak bola, di mana pemahaman akan sistem permainan sangat penting dalam mengoptimalkan kinerja tim secara keseluruhan.

Lebih jauh, analisis dalam sepak bola tidak hanya fokus pada aspek teknis dan taktis, tetapi juga memperhatikan variabel fisik yang relevan dalam konteks permainan sepak bola pria dewasa. Di samping itu, terdapat variabel situasional yang perlu diperhatikan seperti lokasi pertandingan, kualitas lawan, dan status pertandingan yang berpengaruh pada performa tim (Sarmento *et al*., 2014). Faktor-faktor ini menambah kompleksitas analisis dan menekankan pentingnya pendekatan menyeluruh yang mempertimbangkan kondisi dinamis permainan.

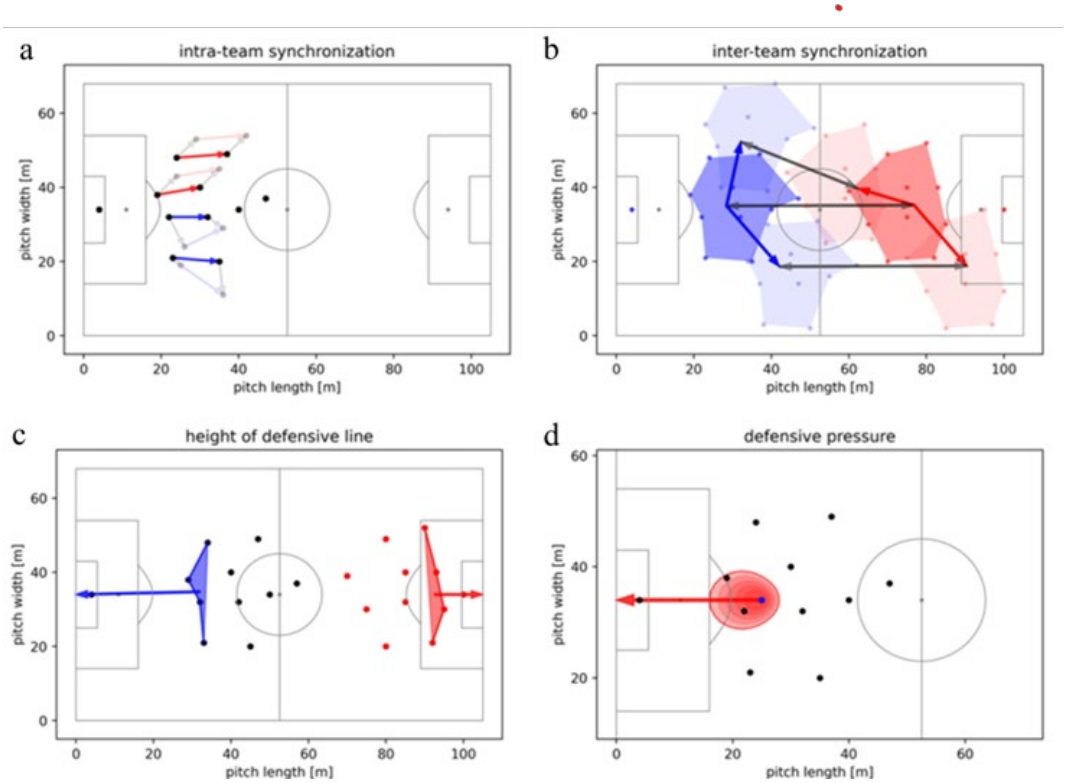
Dalam upaya meningkatkan performa pemain dan mengembangkan aktivitas pelatih, analisis sepak bola juga mengarah pada aspek-aspek mendetail seperti performa dalam situasi bola mati, perilaku sistem kolektif, komunikasi tim, dan profil aktivitas pemain. Fokus ini bertujuan untuk memberikan wawasan yang lebih dalam mengenai pola-pola permainan serta interaksi pemain di lapangan, yang

pada akhirnya membantu pelatih dalam menyesuaikan strategi berdasarkan analisis berbasis data yang komprehensif (Sarmento *et al*., 2018).

Salah satu contoh penerapan analisis sepak bola yang semakin populer adalah penggunaan *data tracking* pemain. Data ini memungkinkan analisis yang lebih mendalam terhadap struktur permainan dengan memberikan wawasan mengenai performa tim, terutama dalam strategi bertahan. Implementasi analisis ini mengidentifikasi karakteristik permainan defensif yang berhasil, ditandai dengan tekanan tinggi, sinkronisasi gerakan antar pemain, keseimbangan pertahanan, serta organisasi pertahanan yang kompak dan terkoordinasi. Melalui *data tracking*, pelatih dan analis dapat memahami pola pertahanan yang efektif dan mengoptimalkan strategi tim berdasarkan perilaku lapangan yang terukur (Forcher *et al*., 2022).

Gambar 2.1 menunjukkan contoh visualisasi hasil analisis menggunakan *data tracking*, yang menggambarkan indikator-indikator kinerja utama dalam permainan bertahan. Pada bagian (a), visualisasi menunjukkan tingkat sinkronisasi intra-tim di mana perilaku gerakan yang sinkron digambarkan pada warna merah dan perilaku yang asinkron pada warna biru. Bagian (b) menggambarkan sinkronisasi gerakan antar tim melalui pusat massa (*centroid*) dari tim-tim yang berlawanan. Pada bagian (c), visualisasi ini menunjukkan tinggi garis pertahanan (biru) yang digunakan sebagai pengukur posisi bertahan. Terakhir, bagian (d) menunjukkan tekanan bertahan yang dilakukan oleh dua pemain bertahan (hitam) terhadap pemain penyerang (biru), dengan arah ancaman ke gawang yang

ditunjukkan oleh panah merah. Visualisasi ini memberikan gambaran yang jelas tentang dinamika taktik bertahan dalam sepak bola (Forcher *et al*., 2022).



Gambar 2.1 Contoh Visualisasi pada Analisis Sepak Bola (Forcher *et al*., 2022)

2.2 *Expected Goals* (xG)

Expected Goals atau xG adalah salah satu metrik yang semakin digunakan dalam analisis sepak bola modern untuk menilai peluang terjadinya gol berdasarkan kualitas dan lokasi tembakan yang dilakukan (Mead, O'Hare, & McMenemy, 2023). Metrik ini memberikan prediksi probabilitas yang lebih akurat dibandingkan statistik konvensional dalam memperkirakan keberhasilan suatu tim di masa mendatang. Dalam hal ini, xG membantu memberikan pandangan yang lebih obyektif dan berbasis data mengenai kemungkinan pencapaian gol yang dihasilkan dari berbagai jenis tembakan selama pertandingan.

Metrik xG dirancang untuk memberikan skor probabilistik pada setiap tembakan, dengan nilai yang berkisar antara 0 dan 1, di mana 0 menunjukkan tidak ada peluang mencetak gol, dan 1 menunjukkan kepastian terjadinya gol. Penilaian ini memungkinkan xG untuk menangani unsur ketidakpastian dalam sepak bola dengan lebih baik dibandingkan metrik berbasis gol konvensional. Karena tembakan jauh lebih sering terjadi daripada gol, pendekatan ini memungkinkan analisis yang lebih stabil dan realistis dalam memahami efektivitas tim dan pemain di lapangan (Mead, O'Hare, & McMenemy, 2023).

Selain berguna untuk analisis taktis yang mendukung peningkatan performa di lapangan, xG juga memainkan peran penting dalam keputusan finansial klub. Metrik ini membantu dalam keputusan seperti perekrutan pemain dan negosiasi kontrak dengan memberikan wawasan yang lebih akurat mengenai kontribusi pemain. Dengan demikian, xG tidak hanya membantu klub dalam memaksimalkan performa di lapangan tetapi juga dalam mengelola sumber daya finansial secara lebih efisien (Mead, O'Hare, & McMenemy, 2023).

Penerapan xG memberikan keuntungan strategis bagi klub sepak bola dengan memperluas pemahaman terkait kualitas peluang yang dihasilkan. Hal ini memungkinkan klub untuk mengevaluasi kinerja pemain secara lebih mendalam dan membantu dalam pengembangan strategi permainan yang berbasis pada kualitas dan efektivitas peluang (Mead, O'Hare, & McMenemy, 2023). Dengan menerapkan xG, klub dapat membuat keputusan yang lebih baik dalam berbagai aspek, termasuk analisis performa, perekrutan, dan perencanaan jangka panjang,

yang menjadikan xG sebagai alat yang sangat berharga dalam manajemen modern sepak bola.

Di dalam konsepnya, perhitungan xG dapat dianggap sebagai permasalahan klasifikasi, karena melibatkan penentuan probabilitas tembakan menghasilkan gol berdasarkan berbagai faktor. Untuk menghitung probabilitas ini, metode *machine learning* dan statistika sering diterapkan, termasuk *logistic regression*, *gradient boosting*, *neural networks*, *support vector machines*, serta algoritma klasifikasi *tree-based*. Beragam pendekatan ini memungkinkan xG untuk memanfaatkan data historis dan pola dalam data tembakan untuk memodelkan kemungkinan gol secara lebih akurat, yang berguna dalam memberikan penilaian yang lebih detail tentang kualitas peluang tembakan (Herbinet, 2018).

Model xG dapat memiliki tingkat akurasi yang berbeda tergantung pada jumlah faktor yang dimasukkan ke dalam perhitungannya. Sebagai contoh, model xG standar biasanya memperhitungkan jarak tembakan ke gawang, sudut tembakan, bagian tubuh yang digunakan, dan jenis umpan yang mendahului tembakan.

Berdasarkan faktor-faktor tersebut, sebuah tembakan mungkin diberi nilai 0,30 xG. namun model yang lebih presisi, seperti Statsbomb xG, mempertimbangkan informasi tambahan seperti posisi kiper, status kiper, posisi pemain bertahan dan penyerang, serta tinggi dampak tembakan. Dalam kondisi kiper yang tidak berada di posisinya, model ini mungkin memberikan nilai yang lebih tinggi, misalnya 0,65 xG, untuk menggambarkan kualitas peluang yang lebih tinggi (Statsbomb, 2024).

Visualisasi dari model ini pada Gambar 2.2, yang merupakan Visualisasi xG pada Pertandingan Langsung, memperlihatkan bagaimana setiap faktor dihitung untuk menghasilkan prediksi xG yang mendalam dan akurat.

Gambar 2.2 Visualisasi xG pada Pertandingan Langsung (Statsbomb, 2024)

2.3 *Machine Learning*

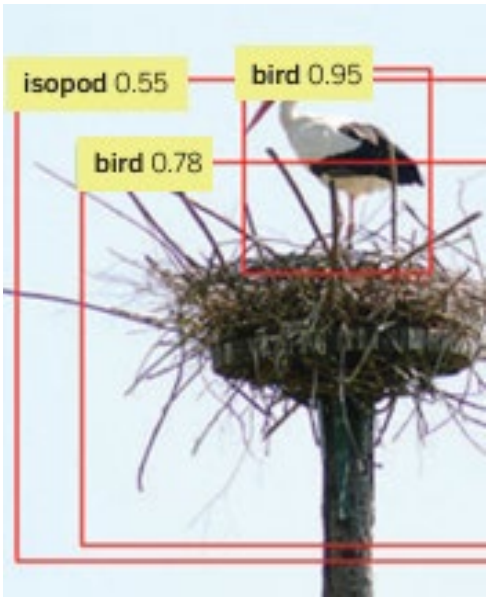
Machine Learning (ML) merupakan kemampuan suatu sistem untuk belajar dari data pelatihan yang spesifik terhadap masalah tertentu, dengan tujuan untuk mengotomatisasi proses pembangunan model analitik serta memecahkan tugas-tugas terkait. Dalam konteks ini, ML memungkinkan sistem komputer untuk mengidentifikasi pola dalam data tanpa campur tangan manual yang intensif, sehingga memungkinkan solusi otomatis terhadap berbagai masalah kompleks berbasis data (Janiesch *et al*., 2021).

Secara lebih mendalam, ML dapat dilihat sebagai bentuk kecerdasan buatan (AI) yang memanfaatkan data untuk melatih komputer dalam melakukan berbagai tugas tertentu, menggunakan algoritma untuk membangun serangkaian aturan

secara otomatis. Proses ini memungkinkan sistem untuk secara mandiri mengenali pola serta membuat keputusan berdasarkan data tanpa perlu diinstruksikan secara eksplisit, yang pada akhirnya meningkatkan ketepatan dan efisiensi sistem dalam memecahkan masalah kompleks (Schneider & Guo, 2018).

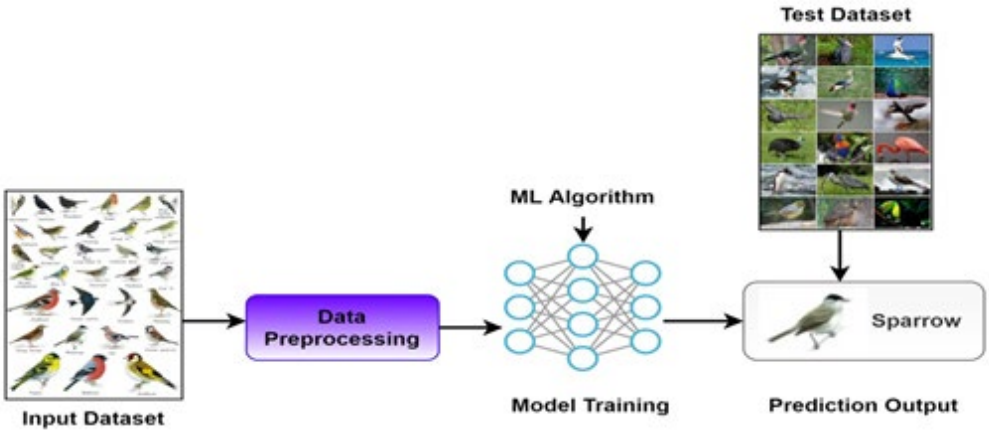
Machine Learning berbeda dari data *mining* dan statistik tradisional, baik dalam aspek filosofis maupun metodologis. Terdapat tiga pendekatan utama dalam ML yang membedakannya, yaitu statistika klasik, teori pembelajaran statistik Vapnik, serta teori pembelajaran komputasional (Kodama *et al*., 2023). Ketiga pendekatan ini menyediakan dasar yang berbeda untuk pengembangan algoritma, dimana ML fokus pada kemampuan untuk terus memperbaiki kinerja model berdasarkan data pelatihan, dibandingkan hanya melakukan analisis data retrospektif sebagaimana dalam statistik tradisional.

Machine Learning memiliki penerapan yang luas, salah satunya adalah *visual recognition*, yang memungkinkan pengenalan objek atau wajah dalam gambar secara otomatis. Dalam Gambar 2.3, ditampilkan implementasi ML pada aplikasi pengenalan visual, di mana teknologi ini mengenali dan mengklasifikasikan objek secara *real-time* berdasarkan data visual. Penerapan ini memanfaatkan kemampuan ML untuk belajar dari data visual guna membangun model yang mampu mendeteksi pola, bentuk, atau warna tertentu, serta mengenali objek-objek spesifik secara akurat dan efisien.



Gambar 2.3 Contoh Implementasi *Machine Learning* (Jordan & Mitchell, 2015)

Terdapat berbagai kategori dalam *Machine Learning*, meliputi *supervised learning*, *unsupervised learning*, dan *reinforcement learning*. Masing-masing pendekatan ini memiliki teknik-teknik unik, seperti *zero-shot learning, active learning, contrastive learning, self-supervised learning*, dan *semi-supervised learning* (Mahadevkar *et al*., 2022). Dalam Gambar 2.4, ditunjukkan contoh implementasi *supervised learning*, di mana model dilatih menggunakan data berlabel untuk dapat mengklasifikasikan atau memprediksi berdasarkan pola yang telah dikenali. Teknik-teknik ini memperkaya cara sistem mempelajari data visual, baik dengan data yang memiliki label atau tanpa label.



Gambar 2.4 Contoh Implementasi *Supervised Learning* (Mahadevkar *et al*., 2022)

Algoritma dasar dalam *Machine Learning* sangat beragam, mencakup *decision tree, random forest, artificial neural network, support vector machine* (SVM), serta algoritma *boosting* dan *bagging*, yang membantu dalam meningkatkan kinerja model dengan menggabungkan prediksi dari beberapa model. Selain itu, algoritma *backpropagation* (BP) berperan penting dalam *neural networks* untuk mengoptimalkan bobot model berdasarkan kesalahan yang dihasilkan pada prediksi awal, sehingga meningkatkan kemampuan sistem dalam memprediksi hasil dengan lebih akurat (Jin, 2020).

Dalam *Machine Learning*, metrik evaluasi adalah instrumen logis dan matematis yang digunakan untuk mengukur seberapa dekat hasil prediksi model terhadap nilai aktualnya. Metrik evaluasi memungkinkan analisis kinerja model secara mendalam, sehingga aspek seperti akurasi, kesalahan, dan ketepatan dalam memprediksi dapat diukur secara kuantitatif. Hal ini penting untuk memahami performa model dan menentukan langkah-langkah penyempurnaan lebih lanjut dalam pengembangan model (Plevris *et al*., 2022).

Beberapa metrik evaluasi yang paling sering digunakan dalam *Machine Learning* mencakup *Root Mean Squared Error* (RMSE), *Mean Absolute Error* (MAE), *Pearson Correlation Coefficient*, dan *Coefficient of Determination* (R^2) (Plevris *et al.*, 2022). Metrik-metrik ini membantu dalam mengukur seberapa akurat dan presisi prediksi model terhadap data yang diujikan, sehingga para praktisi dapat memilih metrik evaluasi yang paling relevan dengan konteks data dan tujuan analisis mereka.

2.4 Data *Preprocessing*

Data *preprocessing* adalah tahap penting yang bertujuan untuk menghasilkan kumpulan data akhir yang akurat, bersih, dan siap digunakan dalam algoritma penambangan data berikutnya (García, Luengo & Herrera, 2016). Proses ini memastikan bahwa data yang akan dianalisis telah melalui serangkaian langkah perbaikan dan penyesuaian, seperti pembersihan dari kesalahan, penghapusan data duplikat, serta transformasi data ke format yang lebih sesuai. Dengan demikian, data yang dihasilkan akan memiliki kualitas yang lebih tinggi dan dapat mendukung proses penambangan data secara lebih efektif serta menghasilkan informasi yang lebih andal.

Teknik *preprocessing* data mencakup serangkaian langkah penting yang mencakup:

a. Pembersihan data

b. Integrasi data

c. Reduksi data

d. Transformasi data.

Proses ini dirancang untuk memastikan bahwa data yang akan digunakan dalam penambangan memiliki kualitas yang tinggi dan struktur yang optimal (Sammut & Webb, 2017).

Data *preprocessing* merupakan tahap yang sangat krusial dalam *pipeline machine learning*, karena berperan langsung dalam menentukan kualitas data yang akan digunakan serta informasi yang dihasilkan dari proses tersebut (Bilal *et al*., 2022). Tahap ini memastikan bahwa data mentah yang tersedia diubah menjadi data yang lebih terstruktur, bersih, dan relevan untuk analisis lebih lanjut. Kualitas data yang diproses dengan baik akan sangat mempengaruhi kinerja model *machine learning* yang dibangun, sehingga menghasilkan prediksi atau keputusan yang lebih akurat dan andal.

2.5 *Feature Engineering*

Feature engineering adalah proses rekayasa data secara cerdas untuk meningkatkan kinerja model *machine learning* dengan cara meningkatkan akurasi dan kemampuan interpretasinya (Verdonck *et al*., 2024). Proses ini dilakukan melalui penyesuaian fitur yang telah ada atau dengan mengekstraksi fitur baru yang lebih bermakna dari berbagai sumber data. Teknik ini bertujuan untuk menciptakan representasi data yang lebih informatif, sehingga model dapat memahami hubungan yang lebih kompleks di dalam data. *Feature engineering* tidak hanya membantu dalam memperbaiki akurasi prediksi, tetapi juga memungkinkan pengguna untuk memahami bagaimana setiap fitur memengaruhi hasil akhir, menjadikannya

langkah penting dalam pengembangan model *machine learning* yang lebih efektif dan dapat diandalkan.

Feature engineering memungkinkan pengguna untuk membuat fitur-fitur baru secara mandiri yang lebih relevan dengan permasalahan yang sedang dianalisis (Das *et al*., 2022). Fitur-fitur ini kemudian dapat digunakan untuk meningkatkan proses penerapan algoritma *machine learning* dalam membuat prediksi yang lebih akurat. Dengan menciptakan fitur yang disesuaikan dengan kebutuhan analisis, pengguna dapat membantu model *machine learning* mengenali pola-pola penting yang sebelumnya tidak terdeteksi, sehingga hasil prediksi menjadi lebih optimal dan bermakna.

Teknik-teknik esensial dalam *feature engineering* berperan penting dalam meningkatkan kinerja model prediksi di berbagai bidang. Teknik-teknik ini mencakup (Katya, 2023):

a. *Feature Selection*

Feature Selection merupakan proses memilih fitur-fitur yang paling relevan dan informatif dari kumpulan data yang tersedia. Dengan menyaring fitur yang tidak signifikan atau *redundant*, proses ini membantu mengurangi *noise* dan kompleksitas data. Hal tersebut sangat penting untuk mencegah *overfitting* dan memastikan bahwa model hanya menggunakan informasi yang benar-benar berkontribusi terhadap variabel target. Dengan demikian, model prediksi dapat bekerja lebih efisien dan menghasilkan akurasi yang lebih tinggi.

b. *Dimensionality Reduction*

Dimensionality reduction adalah teknik yang bertujuan untuk mengurangi jumlah fitur dalam *dataset* tanpa mengorbankan informasi penting yang terkandung di dalamnya. Teknik ini menyederhanakan struktur data, sehingga memudahkan proses analisis dan meningkatkan performa model. Metode seperti *Principal Component Analysis* (PCA) mengubah fitur asli menjadi komponen baru yang lebih ringkas, tetapi tetap merepresentasikan variasi data secara keseluruhan. Pendekatan ini tidak hanya mempercepat proses pelatihan model, tetapi juga meningkatkan kemampuan interpretasi hasil.

c. Interaction Term Creation

Interaction term creation adalah proses menciptakan fitur baru dengan mengombinasikan dua atau lebih fitur yang ada. Teknik ini dirancang untuk menangkap interaksi atau hubungan sinergis antar fitur yang mungkin tidak terlihat saat dianalisis secara individual. Dengan menggabungkan fitur-fitur tersebut, model dapat lebih sensitif terhadap pola-pola kompleks yang berpengaruh terhadap hasil akhir, sehingga meningkatkan keakuratan prediksi.

Secara keseluruhan, penerapan teknik-teknik ini dalam *feature engineering* membantu mengoptimalkan data input sehingga algoritma *machine learning* dapat menghasilkan prediksi yang lebih akurat dan interpretasi yang lebih mendalam. Teknik-teknik tersebut berperan penting dalam menyederhanakan, menyoroti, dan memperkaya informasi yang terkandung dalam data, yang pada akhirnya berkontribusi terhadap peningkatan kinerja model di berbagai aplikasi.

2.6 *Gradient Boosting*

Gradient boosting merupakan teknik *machine learning* yang sangat efektif dan sering digunakan untuk menangani tugas dengan fitur heterogen serta data yang cenderung berisik. Teknik ini bekerja dengan menggabungkan prediksi dari sejumlah model sederhana atau *weak learners* untuk menghasilkan prediksi yang kuat. Dalam klasifikasi, *Gradient boosting* menghasilkan distribusi pada label kelas, sementara dalam regresi, model ini memberikan prediksi nilai tunggal atau *point prediction* untuk mendekati hasil yang diinginkan. Kemampuan *gradient boosting* dalam menghadapi variasi pada fitur dan ketidakpastian dalam data menjadikannya alat yang sangat kuat dalam berbagai aplikasi *machine learning* (Ustimenko, Prokhorenkova, & Malinin, 2021).

Proses *gradient boosting* dimulai dengan mengombinasikan *weak learners*, yaitu model yang performanya sedikit lebih baik dari prediksi acak, untuk membentuk *strong learner* secara iteratif. *gradient boosting* merupakan algoritma *boosting* yang dirancang khusus untuk masalah regresi.

Dalam algoritma ini, diberikan kumpulan data pelatihan $D = \{ x_i, y_i \}_1^N$, dengan tujuan utama mencari aproksimasi $\hat{F}(x)$ dari fungsi $F^*(x)$, yang memetakan instance $\boldsymbol{x}$ ke nilai output $\boldsymbol{y}$, melalui minimisasi nilai ekspektasi dari fungsi loss tertentu $L(y, F(x))$. *Gradient boosting* membangun aproksimasi tambahan dari $F^*(x)$ sebagai jumlah berbobot dari sejumlah fungsi, sehingga memungkinkan model meningkatkan akurasi prediksi melalui iterasi yang berfokus pada mengurangi kesalahan residu (Bentéjac, Csörgő, & Martínez-Muñoz, 2020).

Pada persamaan 2.1 menunjukkan bagaimana setiap model baru (x) ditambahkan secara bertahap dengan bobot pada iterasi ke-m, yang bertujuan untuk mengurangi kesalahan prediksi dari model sebelumnya.

$$F_m(\mathrm{x}) = F_{m-1}(\mathrm{x}) + \rho_m h_m(\mathrm{x}) \tag{2.1}$$

Dalam proses iteratif *gradient boosting*, ρ_m adalah bobot yang diberikan pada fungsi ke-m, yaitu $h_m(x)$**.** Fungsi-fungsi ini merupakan model-model dalam *ensemble*, seperti *decision tree*. Aproksimasi dari $F^*(x)$ dibangun secara bertahap, dimulai dengan mendapatkan aproksimasi konstan untuk $F^*(x)$ pada iterasi pertama. Hal ini dicapai dengan meminimalkan nilai *loss function* $L(y_i, \alpha)$ untuk setiap data pelatihan, dengan $\boldsymbol{\alpha}$ adalah parameter konstanta yang mengoptimalkan fungsi tersebut. Pada iterasi pertama, aproksimasi ini diberikan oleh persamaan 2.2.

$$F_0(\mathrm{x}) = \underset{\alpha}{\operatorname{argmin}} \sum_{i=1}^{N} L(y_i, \alpha) \tag{2.2}$$

Persamaan ini menunjukkan bahwa pada awalnya, model menghasilkan prediksi yang didasarkan pada nilai konstanta $\boldsymbol{\alpha}$ yang meminimalkan kesalahan prediksi keseluruhan, $L(y_i, \alpha)$, di seluruh *dataset* . Pendekatan ini digunakan untuk membangun dasar dari model *gradient boosting* sebelum melanjutkan ke iterasi selanjutnya, di mana model-model tambahan (seperti *decision tree*) akan berfungsi untuk memperbaiki prediksi dari model sebelumnya (Bentéjac, Csörgő, & Martínez-Muñoz, 2020).

Pada iterasi selanjutnya, model yang dibangun diharapkan dapat meminimalkan fungsi berikut:

$$\left(\rho_m, h_m(\mathrm{x})\right) = \underset{\rho,h}{\operatorname{argmin}} \sum_{i=1}^{N} L\left(y_i, F_m - 1(\mathrm{x}_i) + \rho h(\mathrm{x}_i)\right) \tag{2.3}$$

Namun, alih-alih menyelesaikan masalah optimisasi ini secara langsung, setiap model h_m dapat dipandang sebagai langkah *greedy* dalam optimisasi menggunakan metode *gradient descent* untuk F^*. Untuk itu, setiap model h_m dilatih menggunakan *dataset* baru $D = \{x_i, r_{mi}\}_{i=1}^{N}$, di mana *residual* palsu $\boldsymbol{r_{mi}}$ dihitung berdasarkan turunan dari fungsi *loss* $L(y, F(x))$ terhadap $F(x)$, yang dievaluasi pada $F(x) = F_{m-1(x)}$, dengan rumus:

$$\mathrm{r}_{mi} = \left[\frac{\partial L\left(y_i, F(\mathrm{x})\right)}{\partial L(\mathrm{x})}\right]_{F(\mathrm{x})=F_{m-1(\mathrm{x})}} \tag{2.4}$$

Nilai dari ρ_m kemudian dihitung dengan menyelesaikan masalah optimisasi pencarian garis. Proses ini, meskipun sangat efektif, dapat mengalami *overfitting* jika langkah-langkah iteratif tidak diatur dengan benar. Beberapa fungsi loss (misalnya loss kuadratik) dapat menyebabkan *residual* palsu menjadi nol pada iterasi berikutnya jika model h_m sangat cocok dengan *residual* palsu, yang akan menyebabkan proses tersebut berhenti terlalu cepat. Untuk mengatasi masalah ini dan mengontrol proses penambahan dalam *gradient boosting*, beberapa parameter

regularisasi dipertimbangkan. Salah satu cara alami untuk meredakan *overfitting* adalah dengan menerapkan *shrinkage*, yang berfungsi untuk mengurangi setiap langkah *gradient descent* (Bentéjac, Csörgő, & Martínez-Muñoz, 2020).

Gradient boosting membedakan dirinya dari metode *boosting* lainnya dengan menggabungkan konsep-konsep dari teori klasifikasi untuk estimasi dan seleksi efek prediktor dalam model regresi. Dalam hal ini, *gradient boosting* mempertimbangkan efek acak dan menawarkan pendekatan pemodelan yang lebih organik dan tidak bias. Berbeda dengan algoritma *boosting* lainnya yang mungkin mengasumsikan hubungan linier atau terlalu bergantung pada keputusan acak dalam tahap pemilihan model, *gradient boosting* memastikan bahwa estimasi prediktor disesuaikan secara cermat dengan data, meningkatkan akurasi model secara keseluruhan (Griesbach, Säfken, & Waldmann, 2020).

Selain itu, *gradient boosting* juga menawarkan kemampuan untuk menghasilkan perbaikan pada model non-konstan, dengan menggabungkan pengetahuan sebelumnya atau wawasan fisik terkait proses yang menghasilkan data (Wozniakowski, Thompson, Gu, & Binder, 2021). Ini menjadi keunggulan lain dari *gradient boosting*, karena ia tidak hanya mengandalkan data murni, tetapi juga dapat memanfaatkan pengetahuan domain atau pemahaman fisik tentang bagaimana data tersebut terbentuk. Dengan pendekatan ini, *gradient boosting* dapat meningkatkan prediksi dalam konteks yang lebih luas, termasuk dalam situasi di mana model yang lebih sederhana mungkin gagal.

Sebagai algoritma *Ensemble Learning* yang semakin berkembang, telah terbukti unggul dalam meningkatkan prediksi dibandingkan dengan model lain,

seperti *artificial neural network*, terutama dalam konteks pemodelan dinamis *bioprocess* (Mowbray *et al*., 2020). Dalam penerapan ini, *gradient boosting* menggabungkan beberapa model pembelajaran yang lemah untuk menghasilkan prediksi yang lebih akurat, menunjukkan keunggulannya dalam memodelkan dan memprediksi proses yang dinamis dan kompleks, serta mampu mengatasi variasi yang ada dalam data yang digunakan.

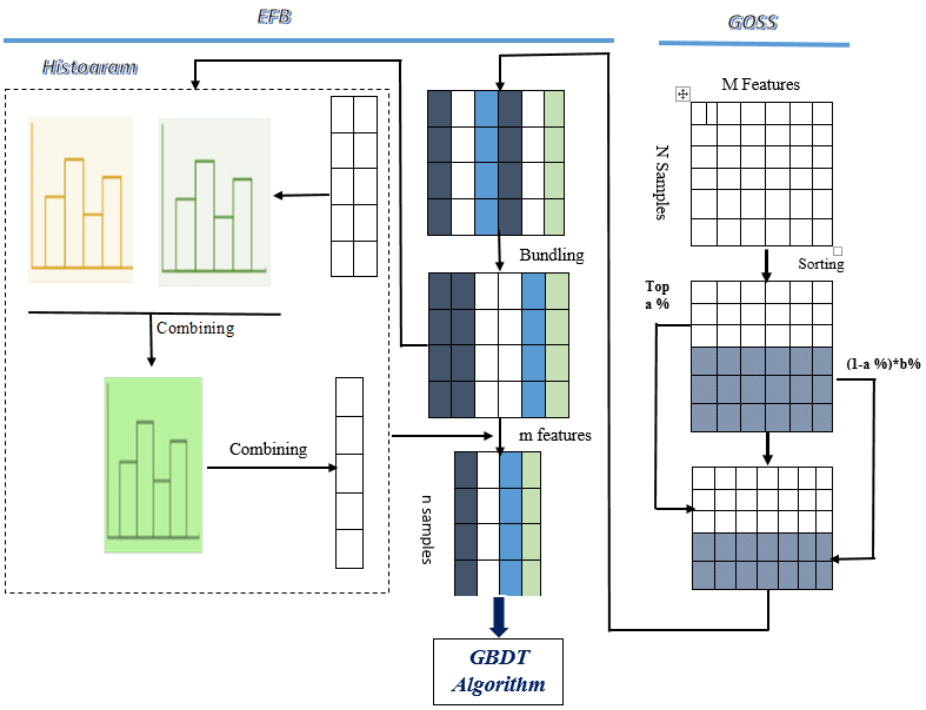
Beberapa parameter dalam *gradient boosting*, seperti jumlah *node*, kedalaman maksimum, dan tingkat pembelajaran, dapat disesuaikan berdasarkan kinerja model pada *testing* set (Hu *et al*., 2023). Pengaturan parameter ini penting untuk memastikan model tidak hanya memberikan prediksi yang akurat, tetapi juga menghindari *overfitting*. Menyesuaikan parameter-parameter tersebut memungkinkan pemodel untuk mengoptimalkan performa model sesuai dengan karakteristik data yang digunakan, menjadikannya lebih fleksibel dan dapat diandalkan dalam berbagai jenis aplikasi.

2.7 *Light Gradient Boosting Machine*

Light Gradient Boosting Machine adalah kerangka kerja yang dirancang untuk mengimplementasikan algoritma *Gradient Boosting Decision Tree* (GBDT). LIGHTGBM memiliki beberapa keunggulan, termasuk kecepatan pelatihan yang lebih tinggi, penggunaan memori yang lebih rendah, akurasi yang lebih baik, serta dukungan untuk distribusi data dalam jumlah besar. *Framework* ini dikembangkan untuk mengatasi keterbatasan dalam GBDT tradisional, khususnya dalam hal

kinerja dan efisiensi komputasi, sehingga memungkinkan pelatihan model pada *dataset* yang lebih besar dengan waktu yang lebih singkat (Huang & Chen, 2023).

LightGBM pertama kali dikembangkan pada tahun 2016 oleh tim peneliti di Microsoft sebagai peningkatan atas model GBDT yang populer, yaitu XGBoost. LightGBM diperkenalkan untuk meningkatkan efisiensi dan kecepatan yang lebih tinggi dari XGBoost, yang sering mengalami kendala kecepatan pada data berukuran besar. Dalam pengembangan LIGHTGBM, tim peneliti memperkenalkan dua teknik baru: *Gradient-based One-Side Sampling* (GOSS) dan *Exclusive Feature Bundling* (EFB). Teknik ini dirancang untuk mengurangi jumlah sampel data dan fitur yang perlu diproses dalam pelatihan GBDT, sehingga mengatasi tantangan komputasi yang terkait dengan pemrosesan *dataset* besar (Kriuchkova, Toloknova, & Drin, 2024). Gambar 2.5 adalah arsitektur peningkatan algoritma GBDT dengan EFB dan GOSS.



Gambar 2.5 Arsitektur GOSS dan EFB

Pada Gambar 2.5 disajikan mengilustrasikan arsitektur dan aliran data dalam kerangka kerja GBDT pada LightGBM yang ditingkatkan, mengintegrasikan teknik EFB dan GOSS. EFB bertujuan untuk mengurangi dimensi fitur dengan menggabungkan fitur-fitur yang jarang aktif bersamaan ke dalam bundel tunggal, sehingga menghasilkan matriks fitur yang lebih ringkas (m fitur dari M fitur awal). Proses ini melibatkan pembentukan histogram untuk setiap fitur dan kemudian menggabungkannya, yang secara efektif mengurangi kompleksitas komputasi tanpa mengorbankan informasi signifikan. Matriks fitur yang telah dibundel kemudian disatukan dengan sampel-sampel yang telah diseleksi oleh GOSS.

Sementara itu, GOSS mengatasi tantangan jumlah sampel yang besar dengan secara selektif mempertahankan instansi berdasarkan gradiennya. Sampel

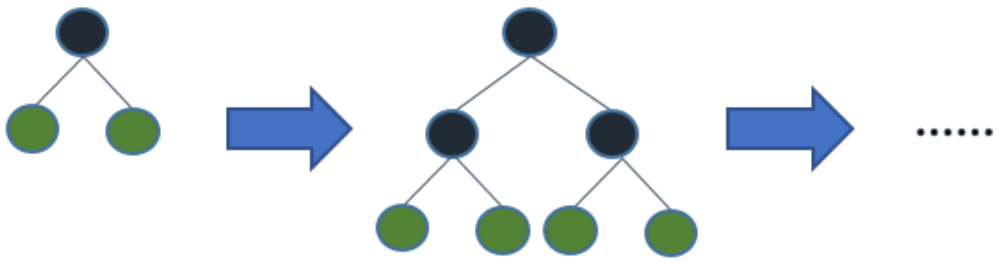
dengan gradien besar (*top* α%) dipertahankan secara utuh karena mereka berkontribusi paling signifikan terhadap *error* model, sedangkan sampel dengan gradien kecil diambil secara acak pada laju (1−α%)×*b*%. Pendekatan ini memungkinkan algoritma GBDT untuk fokus pada sampel yang paling informatif, mempercepat proses pelatihan sambil menjaga akurasi model. Kombinasi EFB dan GOSS secara sinergis mengurangi dimensi fitur dan jumlah sampel, secara substansial meningkatkan efisiensi komputasi dari algoritma GBDT tanpa mengorbankan kinerja, menjadikannya sangat efektif untuk *dataset* skala besar.

LightGBM menunjukkan kegunaan yang sangat luas dalam berbagai bidang dan masalah. Dalam masalah penugasan tugas *multi*-UAV (*Unmanned Aerial Vehicle*), model LightGBM memberikan solusi yang lebih baik dan cakupan solusi yang lebih luas dibandingkan algoritma lainnya. Hal ini menunjukkan kemampuannya dalam menangani masalah kompleks yang melibatkan banyak variabel dan pengambilan keputusan secara bersamaan (Wang & Zhang, 2023).

Dalam pemuliaan tanaman yang memanfaatkan data genom, LightGBM terbukti menghasilkan prediksi yang lebih akurat, model yang lebih stabil, dan proses komputasi yang lebih cepat, misalnya pada data sebanyak 50.000 sampel dan 10.000 SNP (*Single Nucleotide Polymorphism*) LightGBM hanya memerlukan delapan menit pelatihan dan 20 GB memori, sementara rrBLUP (*ridge regression Best Linear Unbiased Prediction*) memakan waktu lebih dari tujuh belas jam pelatihan dan memerlukan 116 GB memori, sehingga mempercepat proses seleksi sifat unggul berbasis genomik (Yan *et al*., 2021).

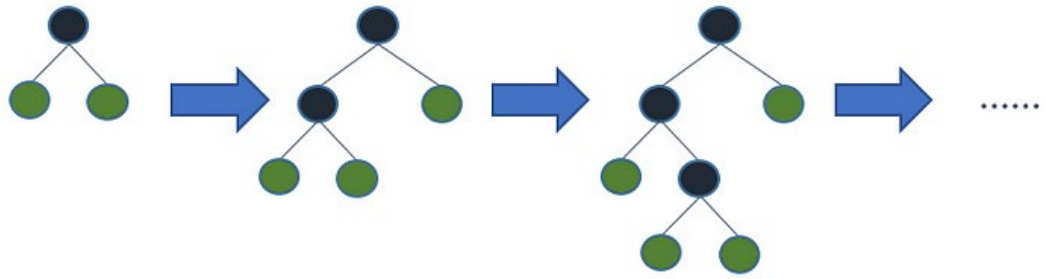
Dalam prediksi beban termal bangunan, LightGBM terbukti lebih unggul dibandingkan dengan algoritma *Random Forest* (RF) dan *Long Short-Term Memory* (LSTM) dalam hal akurasi prediksi dan efisiensi komputasi. Sebagai contoh, dalam studi oleh Chen *et al*. (2023), LightGBM mencapai nilai koefisien variasi dari *root mean squared error* (CVRMSE) sebesar 5,25 persen dan koefisien determinasi (R^2) sebesar 0,9959, dengan waktu komputasi hanya 7 detik. Sebaliknya, RF memiliki CVRMSE sebesar 18,54 persen, R^2 sebesar 0,9482, dan waktu komputasi 44,6 detik, sedangkan LSTM menunjukkan CVRMSE sebesar 22,06 persen, R^2 sebesar 0,9267, dan waktu komputasi 758,8 detik. Hasil ini menunjukkan bahwa LightGBM tidak hanya memberikan prediksi yang lebih akurat tetapi juga memerlukan waktu komputasi yang jauh lebih singkat, menjadikannya pilihan yang sangat efisien untuk aplikasi di bidang konstruksi dan manajemen energi bangunan.

LightGBM menggunakan pendekatan yang berbeda dalam *decision tree learning* dibandingkan algoritma *decision tree* tradisional yang biasanya tumbuh berdasarkan tingkat atau kedalaman pohon (*depth-wise*). Dalam metode tradisional ini, semua *node* pada tingkat yang sama dianggap sama pentingnya, dan pohon bertumbuh secara berjenjang untuk mencakup setiap *node* pada tingkat tertentu, seperti yang ditunjukkan pada gambar 2.5 (LightGBM, 2024).



Gambar 2.6 Ilustrasi *Level-wise Tree Growth* (LightGBM, 2024)

Namun, LightGBM mengadopsi strategi pertumbuhan pohon berbasis daun atau *leaf-wise*, yang hanya membagi daun yang diharapkan memberikan peningkatan terbesar terhadap akurasi model, seperti pada gambar 2.6. Dengan fokus pada daun yang paling berpotensi untuk meningkatkan performa model, LightGBM membangun pohon secara lebih selektif dan efisien. Strategi *leaf-wise* ini bertujuan untuk memaksimalkan akurasi model dengan sumber daya yang lebih minimal, dibandingkan dengan metode tradisional yang sering kali menghasilkan cabang-cabang pohon yang tidak diperlukan dan memperlambat proses pelatihan (LightGBM, 2024).



Gambar 2.7 Ilustrasi *Leaf-wise Tree Growth* (LightGBM, 2024)

Pendekatan *leaf-wise* dalam LightGBM sering disebut juga sebagai pertumbuhan "*greedy growth*," yang memungkinkan algoritma untuk menemukan dan membagi daun dengan dampak terbesar terhadap akurasi model tanpa harus mempertimbangkan semua cabang secara merata pada setiap tingkat (LightGBM, 2024). Hal ini dapat diibaratkan seperti memangkas cabang-cabang yang tidak perlu, dengan fokus pada jalur yang paling bermanfaat. Sebagai akibat dari

pendekatan yang selektif ini, struktur pohon dalam LightGBM menjadi asimetris, di mana beberapa cabang tumbuh lebih dalam daripada cabang lainnya, karena tujuan utamanya bukan simetri, melainkan peningkatan akurasi model.

Manfaat dari strategi pertumbuhan berbasis daun ini adalah dalam hal kecepatan dan akurasi (LightGBM, 2024). Dari segi kecepatan, LightGBM menjadi sangat efisien karena metode *leaf-wise*-nya hanya membagi daun yang memberikan dampak signifikan pada model, sehingga menghindari pengembangan sub-pohon yang tidak berkontribusi banyak terhadap peningkatan akurasi. Selain itu, pertumbuhan *leaf-wise* ini cenderung menghasilkan model dengan tingkat kesalahan (*loss*) yang lebih rendah dan akurasi yang lebih tinggi, karena algoritma dapat lebih terfokus pada bagian data yang paling informatif. Hal ini menjadikan LightGBM sebagai algoritma yang unggul dalam hal efisiensi dan ketepatan dalam menangani *dataset* yang besar dan kompleks.

Untuk mengimplementasikan LightGBM, *library* utama yang diperlukan adalah LightGBM itu sendiri, yang dapat diinstal melalui pengelola paket sesuai bahasa pemrograman yang digunakan, seperti *Python* atau R (LightGBM, 2024). Selain *library* utama tersebut, ada beberapa dependensi lain yang juga dibutuhkan, seperti CMake untuk membangun lingkungan pengembangan dan library CUDA jika ingin memanfaatkan akselerasi GPU untuk mempercepat proses komputasi. Dengan adanya dukungan GPU, LightGBM dapat menangani data dalam jumlah besar dengan lebih efisien, mempercepat pelatihan model secara signifikan.

Dalam pengembangan model LightGBM, kemampuan interpretasi dan keterbukaan model merupakan aspek penting, terutama untuk memahami alasan di

balik prediksi yang dihasilkan. Teknik interpretasi seperti *Permutation Feature Importance* (PFI) dan *Shapley additive explanations* (SHAP) menjadi metode yang sangat berguna untuk menjelaskan kontribusi setiap fitur dalam model terhadap prediksi akhir (Chaibi *et al*., 2021). PFI, misalnya, menilai pentingnya setiap fitur dengan mengevaluasi dampak perubahan nilai fitur terhadap akurasi model, sementara SHAP memberikan nilai yang menunjukkan pengaruh masing-masing fitur pada setiap prediksi. Dengan menggunakan teknik ini, pengguna dapat lebih memahami dan meningkatkan model yang mereka bangun.

Nilai SHAP, khususnya, dapat digunakan pada model LightGBM untuk memastikan kemampuan interpretasi prediksi dengan tingkat keterbukaan yang lebih tinggi. Dengan mengaplikasikan nilai SHAP, model dapat meningkatkan performa inferensi serta mempercepat waktu pelatihan, terutama pada *dataset* yang kompleks. Selain itu, penggunaan SHAP dapat mengurangi kecenderungan model untuk "*fit-to-noise*" atau penyesuaian yang terlalu sensitif terhadap data acak, yang sering kali menjadi masalah dalam analisis data berukuran besar (Bugaj *et al*., 2021). Hal ini membuat SHAP menjadi alat interpretasi yang sangat efektif dalam membangun model LightGBM yang andal dan terbuka terhadap evaluasi.

Selain PFI dan SHAP, kemampuan interpretasi dan keterbukaan dalam LightGBM dapat ditingkatkan melalui metode pembelajaran yang lebih adaptif seperti *personalized interpretability estimation* (ML-PIE). Dengan pendekatan ini, pengguna dapat mengarahkan proses sintesis model berdasarkan preferensi kemampuan interpretasi yang di personalisasi, melalui algoritma evolusi *bi-objektif* yang mempertimbangkan kemampuan interpretasi bersama dengan akurasi. Metode

ML-PIE ini memungkinkan pengguna untuk menentukan prioritas interpretasi dalam pengembangan model, sehingga menghasilkan model LightGBM yang tidak hanya efisien tetapi juga mudah diinterpretasi, sesuai dengan kebutuhan spesifik dari pengguna atau lingkungan aplikasinya (Virgolin *et al*., 2021).

2.8 Brier *Score*

Brier *Score* merupakan metrik evaluasi yang mengukur ketepatan dalam pemodelan prediksi, dengan cara membagi prediksi ke dalam beberapa kelompok atau "*bins*" berdasarkan kesamaan nilai prediksi (Foster & Hart, 2022). Metrik ini memadukan skor kalibrasi dan skor penyempurnaan (*refinement*) untuk mengukur keahlian dalam pemodelan prediktif. Dengan menggabungkan aspek kalibrasi, yang menunjukkan seberapa baik prediksi sejalan dengan hasil aktual, dan aspek penyempurnaan, yang melihat kemampuan model dalam memisahkan atau membedakan hasil yang berbeda, *Brier Score* memberikan gambaran komprehensif mengenai performa model dalam memberikan prediksi probabilistik.

Penggunaan *Brier Score* dalam evaluasi model probabilitas penting karena metrik ini dapat mengukur kemampuan diskriminasi dan performa prediktif secara keseluruhan. Dengan kata lain, *Brier Score* tidak hanya melihat akurasi dari prediksi probabilitas tetapi juga sejauh mana model dapat membedakan antara kejadian yang mungkin terjadi dengan yang tidak (Dimitriadis *et al*., 2023). Hal ini membuat *Brier Score* menjadi pilihan yang baik untuk mengevaluasi performa model probabilistik, khususnya ketika diperlukan pemahaman yang lebih dalam mengenai kualitas prediksi yang bersifat probabilistik.

$$Brier\ Score\ = (f_t - o_t)^2 \quad (2.5)$$

Brier Score digunakan untuk menghitung selisih kuadrat antara nilai prediksi dan nilai aktual, sebagaimana terlihat pada Persamaan 2.5. Dalam konteks ini, f_t merepresentasikan nilai probabilitas yang diprediksi untuk suatu peristiwa, sedangkan o_t adalah nilai aktual dari peristiwa tersebut (biasanya 1 jika terjadi dan 0 jika tidak terjadi). *Brier Score* memiliki rentang nilai antara 0 hingga 1, di mana nilai yang lebih rendah menunjukkan prediksi yang lebih akurat, mendekati hasil aktual (BMJ Open, 2018).

Brier Score diperkenalkan oleh Glenn W. Brier pada tahun 1950 sebagai alat untuk menilai akurasi prediksi probabilitas (Foster & Hart, 2022). Skor ini menghitung selisih antara nilai prediksi dan realisasi aktual, di mana hasil perhitungan *Brier Score* memperlihatkan seberapa dekat prediksi tersebut dengan hasil aktual menggunakan formula *mean squared error* standar.

Sejak pertama kali diperkenalkan yaitu pada evaluasi ramalan cuaca, *Brier Score* telah berkembang menjadi metode yang diakui untuk mengukur akurasi model probabilitas dalam berbagai bidang, termasuk bisnis dan aplikasi lainnya (Petropoulos *et al*., 2022). Penerapan awalnya pada meteorologi menunjukkan bagaimana metode ini dapat memberikan wawasan yang lebih mendalam terhadap ketepatan perkiraan, yang kemudian menjadikan *Brier Score* sebagai standar dalam penilaian akurasi probabilitas di berbagai disiplin ilmu.

2.9 *Receiver Operating Characteristic Area Under Curve* (ROC AUC)

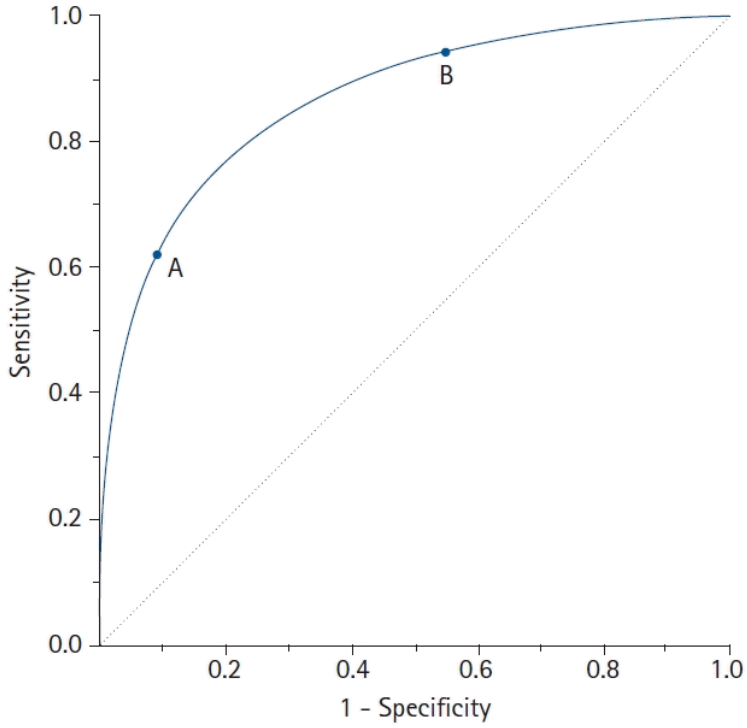
Receiver Operating Characteristic (ROC) adalah alat statistik yang digunakan untuk menilai kinerja model klasifikasi dengan menggambarkan hubungan antara dua parameter, yaitu *True Positive Rate* (TPR) dan *False Positive Rate* (FPR). Analisis ROC dapat dilakukan dengan memanfaatkan distribusi prior dan algoritma *elicitation* untuk memilih prior yang tepat, yang selanjutnya digunakan untuk menarik inferensi mengenai AUC (*Area Under the Curve*) dan karakteristik error model (Labadi *et al*., 2022).

ROC juga digunakan untuk mengevaluasi kinerja perangkat pengujian dan algoritma klasifikasi dalam menilai kepatuhan terhadap kriteria tertentu (Pendrill *et al*., 2023). Dengan demikian, ROC menjadi alat yang penting untuk perbandingan dan evaluasi relatif dari berbagai sistem klasifikasi dalam konteks yang berbeda.

Kurva ROC menggambarkan kinerja model klasifikasi pada berbagai ambang batas klasifikasi dengan memplot dua parameter utama, yaitu TPR dan FPR. Salah satu kelemahan dari kurva ROC adalah kesulitan dalam menginterpretasi kinerja model jika terdapat banyak titik keputusan, karena setiap titik mewakili *trade-off* antara TPR dan FPR, yang dapat membuat sulit untuk menentukan titik terbaik yang mencerminkan kinerja keseluruhan model (Chen *et al*., 2023). AUC mengukur luas dua dimensi di bawah kurva ROC, dimulai dari titik (0,0) hingga (1,1). Semakin tinggi nilai AUC, semakin baik model dalam membedakan antara kelas positif dan negatif.

Pada Gambar 2.7, menampilkan kurva ROC AUC, di mana sumbu x menunjukkan nilai 1 - spesifisitas (*False Positive Rate*) dan sumbu y menunjukkan sensitivitas pada semua nilai *cut-off* yang diukur dari hasil pengujian (Nahm, 2022).

Ketika nilai *cut-off* yang lebih ketat diterapkan, titik pada kurva akan bergerak ke bawah dan ke kiri (Titik A). Sebaliknya, saat *cut-off* lebih longgar diterapkan, titik pada kurva bergerak ke atas dan ke kanan (Titik B). Garis diagonal 45° pada grafik ini berfungsi sebagai garis referensi, yang merepresentasikan kurva ROC dari klasifikasi acak.



Gambar 2.8 Contoh ROC AUC (Nahm, 2022)

ROC AUC memiliki peran penting dalam evaluasi model karena mampu mengukur kinerja model dalam berbagai kelompok risiko yang diprediksi (Carrington *et al*., 2021). Ini memberikan informasi yang lebih mendalam yang dapat digunakan dalam pengambilan keputusan, memungkinkan pemahaman yang lebih komprehensif tentang bagaimana model berperforma di berbagai titik potong dan kelompok risiko.

Lebih lanjut, ROC AUC juga memungkinkan perbandingan yang wajar antar model dan membantu mengidentifikasi batas keputusan yang optimal serta potensi peningkatan AUC. Ini membuat AUC-ROC sangat bermanfaat dalam seleksi model yang lebih baik dan pemahaman tentang ruang yang dapat dioptimalkan untuk meningkatkan kinerja klasifikasi (Tafvizi *et al*., 2022).

2.10 *Knowledge Discovery in Databases* (KDD)

Knowledge Discovery in Databases atau KDD adalah proses yang bertujuan untuk mengekstraksi informasi yang dapat dipahami, menarik, dan bernilai dari data yang tidak terstruktur (Solanki & Sharma, 2021). Proses ini digunakan di berbagai bidang, seperti ilmu kehidupan, perdagangan, keuangan, dan kedokteran, untuk mengidentifikasi pola-pola yang tersembunyi dalam data yang besar dan kompleks (Solanki & Sharma, 2021). Proses ini mencakup berbagai teknik dan metode yang dapat digunakan untuk menggali wawasan dari data yang belum terorganisir.

KDD merupakan suatu bidang yang mengandalkan metode cerdas dalam data *mining* untuk menemukan pola-pola yang menjadi inti pengetahuan (Atloba, Balkir, & El-Mouadib, 2021). Pola-pola ini memungkinkan pengguna untuk memahami informasi yang terkandung dalam *dataset* besar, memberikan wawasan yang dapat diterapkan untuk pengambilan keputusan yang lebih baik dalam berbagai disiplin ilmu.

Proses KDD terdiri dari beberapa tahapan yang saling berinteraksi secara iteratif. Secara umum, tahapan tersebut mencakup (Chaudhary & Kishore, 2017):

a. Pembersihan data (penghapusan kebisingan dan data yang tidak konsisten),
b. Integrasi data (penggabungan beberapa sumber data),
c. Pemilihan data (pengambilan data yang relevan untuk tugas analisis),
d. Transformasi data (pengolahan atau konsolidasi data untuk memudahkan *mining*),
e. Data *mining* (aplikasi metode cerdas untuk mengekstraksi pola),
f. Evaluasi pola (penilaian pola yang menarik berdasarkan ukuran keterminatan), dan
g. Presentasi pengetahuan (penggunaan teknik visualisasi untuk menyajikan pengetahuan yang ditemukan kepada pengguna).

Data *mining* adalah bagian penting dalam proses KDD yang melibatkan berbagai alat dan teknik untuk menemukan pola yang berguna dalam basis data yang sangat besar (Chaudhary & Kishore, 2017). Salah satu bentuk terbaru dari data *mining* adalah ringkasan linguistik, yang bertujuan untuk memberikan deskripsi verbal yang dihasilkan oleh komputer mengenai pengetahuan yang tersembunyi dalam *database*, sering kali dalam bentuk aturan '*if-then*' yang menyerupai granula pengetahuan *fuzzy*. Selain itu, teknik *text mining* digunakan untuk mengekstraksi pola dari dokumen teks, yang melibatkan analisis teks untuk mengubah dokumen yang tidak terstruktur menjadi sekumpulan fitur yang sesuai, lalu menerapkan teknik data *mining* untuk ekstraksi pola (Chaudhary & Kishore, 2017).

Dalam KDD, *machine learning* berperan penting untuk menganalisis data, mengenali korelasi, dan memprediksi hasil yang akan terjadi (Kodati & Selvaraj, 2021). Teknik-teknik *machine learning* digunakan untuk melatih model dalam

mengidentifikasi pola-pola yang ada dalam data, yang kemudian dapat digunakan untuk membuat prediksi yang lebih akurat dalam berbagai aplikasi, seperti analisis kesehatan atau analisis perilaku konsumen.

Aplikasi KDD sangat luas, salah satunya adalah dalam bidang kesehatan, di mana KDD digunakan untuk mengembangkan sistem medis yang dapat mendeteksi dan memberikan saran pengobatan untuk penyakit dengan upaya minimal (Nwankwo, Ngene, & Onuora, 2023). Selain itu, KDD berbasis metode *gradient boosting machine* juga diterapkan dalam prediksi energi listrik, memberikan referensi praktis bagi aplikasi KDD pada sektor energi lainnya (Xie *et al*., 2022).

KDD juga memiliki keterkaitan yang erat dengan analisis olahraga, khususnya sepak bola, di mana pendekatan KDD yang komprehensif memungkinkan persiapan data yang tepat untuk prediksi hasil pertandingan olahraga, termasuk hasil pertandingan sepak bola (Głowania, Kozak, & Juszczuk, 2023). Dengan menggunakan teknik KDD, analisis yang lebih mendalam dapat dilakukan terhadap data pertandingan untuk mengidentifikasi faktor-faktor yang mempengaruhi hasil akhir pertandingan.

2.11 Python

Python adalah bahasa pemrograman tingkat tinggi bersifat object-oriented, dikembangkan oleh Guido van Rossum, bahasa ini dirancang untuk menjadi mudah dipahami dan digunakan sehingga cocok baik untuk pemula yang sedang mempelajari dasar-dasar pemrograman maupun untuk para profesional yang mengerjakan proyek pemrograman di dunia nyata (Srinath, 2017). Python

menawarkan *syntax* yang sederhana dan intuitif, sehingga memungkinkan pengguna menulis kode dengan lebih cepat dan efisien. Selain itu, Python memiliki dukungan pustaka yang sangat luas serta komunitas yang aktif, menjadikannya pilihan populer untuk berbagai kebutuhan, mulai dari pengembangan web, analisis data, *machine learning*, hingga komputasi ilmiah dan otomatisasi sistem.

Python menawarkan keseimbangan antara kejelasan *syntax* dan fleksibilitas dalam pengembangan alat-alat penelitian komputasi, sehingga sangat mendukung dalam menciptakan solusi untuk berbagai jenis permasalahan yang kompleks. Bahasa ini dirancang untuk menangani beragam tantangan yang melibatkan pengolahan *dataset* berukuran besar, penerapan algoritma yang rumit, serta pengembangan sistem komputasi (Pérez, Granger & Hunter, 2011). Kemampuan Python untuk berintegrasi dengan berbagai pustaka dan *framework* membuatnya menjadi pilihan utama dalam penelitian berbasis data dan pengembangan teknologi inovatif. Dengan ekosistem yang luas, Python memungkinkan peneliti dan pengembang untuk membangun, menguji, serta mengimplementasikan solusi secara efisien dan *scalable*.

2.12 Pandas

Pandas adalah pustaka Python berperforma tinggi yang dirancang khusus untuk manipulasi, analisis, dan eksplorasi data. Pustaka ini banyak digunakan oleh peneliti data, analis, dan pengembang karena kemampuannya yang unggul dalam mengolah data secara efisien (Molin & Jee, 2021). Pandas menyediakan berbagai fungsi yang memudahkan proses pembersihan, transformasi, serta analisis data

dalam berbagai format, seperti tabel, *file* CSV, dan *database*. Selain itu, Pandas juga mendukung integrasi dengan pustaka visualisasi seperti Matplotlib dan Seaborn, sehingga memungkinkan pengguna untuk membuat visualisasi data yang informatif dan menarik. Kemudahan penggunaan serta fleksibilitas Pandas menjadikannya salah satu alat utama dalam analisis data modern dan pengembangan aplikasi berbasis data.

Salah satu kekuatan utama dari pustaka ini adalah penggunaan data *frame* dan *series*, yang menjadi inti dalam proses manipulasi, perhitungan, serta analisis data (Nelli, 2015). Data *frame* adalah struktur data berbentuk tabel dengan label pada baris dan kolom, mirip dengan tabel pada *database* atau *spreadsheet*, sehingga memudahkan pengolahan data dalam jumlah besar. Sementara itu, *series* merupakan struktur data satu dimensi yang berfungsi seperti *array*, tetapi dilengkapi dengan indeks yang memungkinkan akses data lebih fleksibel. Kombinasi dari dua struktur data ini memungkinkan pengguna untuk melakukan berbagai operasi analisis secara efisien, seperti pengolahan data numerik, transformasi data, serta agregasi hasil analisis dengan *syntax* yang sederhana namun *powerful*.

2.13 Scikit-learn

Scikit-learn merupakan pustaka Python yang menyediakan antarmuka standar untuk mengimplementasikan berbagai algoritma *machine learning*. Pustaka ini dirancang agar mudah digunakan, sehingga memudahkan pengguna dari berbagai latar belakang untuk mengembangkan model *machine learning* dengan

lebih efisien. Selain mendukung algoritma untuk klasifikasi, regresi, dan *clustering*, Scikit-learn juga dilengkapi dengan berbagai fungsi penting lainnya, seperti data *preprocessing*, *resampling*, evaluasi model, serta pencarian *hyperparameter*. Fungsi-fungsi tersebut membantu memastikan bahwa proses pengolahan data, pelatihan model, hingga evaluasi dapat dilakukan secara menyeluruh dan sistematis (Bisong, 2019).

2.14 Matplotlib

Matplotlib adalah pustaka Python yang digunakan untuk pembuatan grafik dan visualisasi data. Pustaka ini menyediakan berbagai fitur yang memungkinkan pengguna untuk membuat beragam jenis grafik dan diagram, mulai dari grafik garis (*line plot*), grafik sebar (*scatter plot*), peta panas (*heatmap*), diagram batang (*bar chart*), diagram lingkaran (*pie chart*), hingga visualisasi data dalam bentuk tiga dimensi (3D plot) (Hunt, 2019). Kemampuan Matplotlib dalam menghasilkan visualisasi yang informatif dan berkualitas tinggi menjadikannya salah satu alat utama bagi peneliti dan analis data. Selain itu, pustaka ini mendukung kustomisasi penuh pada setiap elemen grafik, seperti warna, label, dan sumbu, sehingga memudahkan pengguna untuk menyajikan data secara lebih menarik dan sesuai dengan kebutuhan analisis.

2.15 Seaborn

Seaborn adalah pustaka Python yang dirancang untuk membuat visualisasi grafik statistik dengan cara yang lebih mudah dan estetis. Pustaka ini menyediakan

antarmuka tingkat tinggi untuk Matplotlib, sehingga memungkinkan pengguna membuat grafik kompleks dengan sedikit kode (Waskom, 2021). Seaborn juga terintegrasi erat dengan Pandas, sehingga pengguna dapat langsung memvisualisasikan data dari struktur data *frame* tanpa perlu konversi tambahan. Dengan berbagai fitur bawaan, seperti pembuatan grafik hubungan antar variabel, distribusi data, serta anotasi statistik, Seaborn membantu dalam menyajikan visualisasi data yang informatif dan menarik. Kemudahan penggunaan serta desain visual yang lebih elegan membuat Seaborn menjadi pilihan utama bagi analis data dan ilmuwan data yang ingin meningkatkan kualitas visualisasi mereka.

2.16 Penelitian Sejenis

Penelitian sejenis yang digunakan pada penelitian ini ditunjukkan pada Tabel 2.1.

Tabel 2.1 Penelitian Sejenis

No	**Sumber**	**Judul dan Penulis**	**Metode dan *Tools***	**Data Set dan Hasil**	**Kelebihan dan Kekurangan**
1	*Asian Conference on Intelligent Information and Database Systems* (2023)	*Improving the Expected Goal Value in Football Using Multilayer Perceptron Networks* (Méndez, Montero, & Núñez, 2023)	*Multilayer Perceptron Neural Network* dan Python, Keras	Sumber: StatsBomb *event* data (> 12 000 tembakan dari berbagai liga top) Input: lokasi tembakan, situasi permainan, tipe *assist* Hasil: MLP ungguli regresi logistik dengan	Kelebihan: Menangkap pola non-linear kompleks Kekurangan: Membutuhkan *tuning* parameter teliti; rawan *overfitting* pada dataset

				peningkatan akurasi hingga 6%; AUC = 0,87	kecil; susah diinterpretasi langsung
2	SSAC 8th *Annual* MIT *Sloan Sports Analytics Conference* (2015)	*Quality vs Quantity: Improved Shot Prediction in Soccer Using Strategic Features from Spatiotemporal Data* (Lucey, Bialkowski, Monfort, Carr & Matthews, 2015)	*Conditional Random Fields, Custom code*	Sumber: Prozone/Stats Perform data (~ 9 732 tembakan + 10 s video pra-tembakan)Fitur: fase permainan, kedekatan & interaksi pemain, kecepatan Hasil: EGV lebih akurat daripada model lokasi-tunggal; peningkatan ROC-AUC vs *baseline*	Kelebihan: Menangkap konteks temporal & strategi tim Kekurangan: Asumsi probabilistik simplistik, kompleksitas inferensi tinggi; skalabilitas terbatas pada dataset besar
3	*Journal of Sports Analytics* 4(3) (2018)	*Spatial Analysis of Shots in MLS: A Model for Expected Goals and Fractal Dimensionality* (Fairchild, Pelechrinis & Kokkodis, 2018)	Regresi Logistik, Python, SciPy/Statsmodels	Sumber: 1 115 tembakan *non-penalty* dari 99 pertandingan MLS 2016 (koordinat x,y ditag manual) Hasil: kalibrasi kuat (*leave-one-out*); ROC-AUC ≈ 0,80	Kelebihan: Sederhana, mudah diinterpretasi, hemat komputasi, Kekurangan: Linieritas membatasi interaksi non-linear & konteks spasial-temporal
4	StatsBomb *Conference*	*"Estimated Player*	*Generalised Linear Mixed*	Sumber: StatsBomb *event* data (580	Kelebihan: Menangkap

	Proceedings (2022)	*Impact" (EPI): Quantifying the Effects of Individual Players on Football Actions Using Hierarchical Statistical Models* (Tureen & Olthof, 2022)	*Models* (GLMM), R (lme4)	*Premier League* + 326 *Women's Super League*) *Output*: skor EPI individual; *random player effects* jelaskan variabilitas *outcome*	struktur hierarkis antar pertandingan & pemain, Kekurangan: Asumsi linearitas & distribusi *residual* membatasi non-linear; komputasi skala besar berat; interpretasi *random effect* sulit
5	*Journal of the Operational Research Society* 76(1) (2022)	*Explainable Expected Goals* (Cavus & Biecek, 2022)	XGBoost, *Random Forest*, LightGBM, CatBoost via Forester AutoML, SHAP untuk interpretasi	Sumber: Understat *event* data (315 430 tembakan dari 7 musim Top 5 Eropa) Hasil: *Random Forest* terbaik pada ROC-AUC (*cross-validation*)	Kelebihan: AutoML memudahkan eksplorasi model; SHAP tingkatkan transparansi, Kekurangan: Risiko *overfitting* pada *forest* besar; waktu komputasi tinggi; SHAP masih terbatas dalam

					interpretasi interaksi fitur
6	IEEE MLDM *for Sports Analytics Workshop* (2016)	*Expected Goals in Soccer: Explaining Match Results Using Predictive Analytics* (Eggels, van Elk & Pechenizkiy, 2016)	*Logistic Regression*, *Decision Tree*, *Random Forest*, AdaBoost, Python (scikit-learn)	Sumber: ORTEC & Inmotio *event*/*tracking* + EA Sports *attributes* (~20 000 tembakan, 2 musim) Hasil: ROC-AUC: AdaBoost ~0,84, *Random Forest* ~0,82, *Logistic* ~0,78, Tree ~0,74	Kelebihan: *Ensemble methods* menangkap interaksi fitur kompleks, Kekurangan: Model kompleks rawan *overfitting*, mahal komputasi, dan sulit diinterpretasi
7	PLoS *ONE* 18(4): e0282295	*Expected Goals in Football: Improving Model Performance and Demonstrating Value* (Mead, O'Hare &	*Logistic Regression*, *Random Forest*, AdaBoost, XGBoost, Python (scikit-learn), *cross-validation*	Sumber: Wyscout (*Soccer_match_event_dataset,* ~250 000 tembakan), Fitur: lokasi, tipe tembakan, nilai pemain, ELO *rating*, penonton, Hasil: ROC-AUC tertinggi: *Random*	Kelebihan: *Ensemble* model tangkap interaksi kompleks dan tingkatkan akurasi, Kekurangan: *Overfitting*,

		McMenemy, 2023)		*Forest* ≈ 0,91; LogReg ≈ 0,85	*tuning* intensif, regresi logistik terlalu sederhana
8	Franklin *Open* 4 (2023)	*A Machine Learning Approach for Player and Position Adjusted Expected Goals in Football* (Hewitt & Karakuş, 2023)	*Logistic Regression*, *Gradient Boosted Trees*, scikit-learn, SHAP	Sumber: StatsBomb *event* data (~15 574 tembakan, 1 887 gol) Hasil: Korelasi: LogReg = 0.887 (xG = 1 866), GBT = 0.902 (xG = 1 870); Uji pada Messi: *uplift* +347 xG	Kelebihan: GBT menangkap dinamika non-linear, SHAP bantu interpretasi fitur, Kekurangan: *Overfitting* dan interpretasi sulit pada GBT, regresi logistik terlalu terbatas
9	Universitat Politècnica de Catalunya (2020)	*Creating a Model for Expected Goals in*	*Logistic Regression*, XGBoost, *Artificial*	Sumber: OPTA (~20 000 tembakan, 5 liga top Eropa) + atribut	Kelebihan: Model non-linear tangkap

		Football Using Qualitative Player Information (P. M. Pardo, 2020)	*Neural Network* (ANN), scikit-learn, Keras	pemain (740 pemain, FIFA) Hasil: AUC LogReg = 0,78; XGBoost = 0,85; ANN = 0,88. RMSE: 0,32; 0,27; 0,25	interaksi atribut kualitatif, Kekurangan: *Overfitting* pada XGBoost; ANN butuh *tuning* intensif & data besar; LogReg terlalu sederhana
10	PLoS *ONE* 19(10): e0312278	*Predicting Goal Probabilities with Improved* xG *Models Using Event Sequences* (Bandara *et al*., 2024)	*Random Forest* (100 *estimators*), scikit-learn	Sumber: StatsBomb Open Data; fitur: urutan 3 *event* (termasuk "*advancement factor*", posisi pemain) Hasil: AUC validasi = 0,833, AUC uji Euro 2020 = 0,826, lebih tinggi dari model *shot*-tunggal	Kelebihan: Fitur temporal meningkatkan akurasi prediksi xG, Kekurangan: Bergantung pada *random under-sampling*; RF rawan *overfitting* &

					interpretasi sulit
11	Unpublished (2024)	*The Power of Pixels: Exploring the Potential of CNNs for Expected Goals* (Matteotti & Sotudeh, 2024)	CNN (*TensorFlow*) Dibanding *LogReg* & *Gradient Boosting*	StatsBomb *event* & *tracking* (~18 000 citra lapangan/*shot*), AUC: CNN = 0,892; GBT = 0,858; LogReg = 0,815	Kelebihan: Ekstraksi spasial otomatis, akurasi tinggi, Kekurangan: Butuh komputasi tinggi, interpretasi rendah (*black box*)
12	JORS 76(1)	*A New xG Model for Football Analytics* (Cefis & Carpita, 2024)	*Logistic Regression*, R (glm)	Understat (~50 000 shots), SoFIFA, Math&Sport, AUC ≈ 0,81 dengan fitur: tekanan, *rating* pemain, kualitas lawan	Kelebihan: Sederhana, mudah diinterpretasi, Kekurangan: Sulit tangkap interaksi non-linear dan konteks kompleks

13	arXiv:2101.02104	*A Probabilistic Model for Predicting Shot Success in Football* (Wheatcroft & Sienkiewicz, 2021)	*Parametric probabilistic model,* Python (SciPy optimize)	>1 juta *shot* dari 22 liga (football-data.co.uk) Perbaikan *log-score* ~0.03 vs *equal-probability baseline*	Kelebihan: Sederhana, cocok untuk *pipeline* prediksi, Kekurangan: Abaikan lokasi/situasi *shot*, risiko propagasi *error*
14	*J. Human Sport and Exercise* 12(2)	*An Examination of Expected Goals and Shot Efficiency* (Rathke, 2017)	*Logistic Regression* per zona, Python (SciPy, statsmodels)	OPTA: EPL & Bundesliga (2012–13; 686 laga total), Fitur: zona lapangan, jarak & sudut *shot* AUC ≈ 0,80	Kelebihan: Sederhana, interpretatif, ringan komputasi, Kekurangan: Model linier tak tangkap interaksi kompleks
15	*Sloan Sports Conf.* (2012)	*An Expected Goals Model for Evaluating NHL Teams and Players*	OLS & *Ridge Regression*, Python (SciPy, pandas)	NHL (2007–2011), 4 musim, *odd*/*even split*, MSE: *Ridge* ≈ 0,06 vs OLS ≈ 0,08	Kelebihan: *Ridge* kurangi *overfitting*, OLS simpel, Kekurangan:

		(Macdonald, 2012)			Tak pakai fitur *shot quality*, terlalu linier

Berdasarkan tabel 2.1, terdapat lima belas penelitian yang membahas tentang perhitungan metrik xG dalam analisis sepak bola. Beberapa penelitian berfokus pada penggunaan metode regresi linier atau logistik. Penelitian yang dilakukan oleh (Fairchild, Pelechrinis & Kokkodis, 2018) menggunakan Regresi Logistik untuk perhitungan xG. Demikian pula, (Rathke, 2017) juga menerapkan Regresi Logistik per zona lapangan. Sementara itu, (Cefis & Carpita, 2024) menggunakan Regresi Logistik dengan fitur tekanan, *rating* pemain, dan kualitas lawan. Penelitian oleh (Wheatcroft & Sienkiewicz, 2021) menggunakan model probabilistik parametrik sederhana. (Macdonald, 2012) menggunakan OLS dan *Ridge Regression*.

Penelitian lainnya memanfaatkan metode *machine learning* dan *deep learning* yang lebih kompleks. (Méndez, Montero, & Núñez, 2023) menerapkan *Multilayer Perceptron Neural Network* dan menunjukkan peningkatan akurasi hingga 6% dibandingkan regresi logistik. (Lucey, Bialkowski, Monfort, Carr & Matthews, 2015) menggunakan *Conditional Random Fields* untuk menangkap konteks temporal dan strategi tim. (Tureen & Olthof, 2022) menggunakan *Generalised Linear Mixed Models* (GLMM) untuk mengukur dampak pemain individu. (Eggels, van Elk & Pechenizkiy, 2016) membandingkan *Logistic Regression*, *Decision Tree*, *Random Forest*, dan AdaBoost. (Cavus & Biecek, 2022)

mengeksplorasi berbagai model seperti XGBoost, *Random Forest*, LightGBM, dan CatBoost menggunakan Forester AutoML, serta SHAP untuk interpretasi. (Mead, O'Hare & McMenemy, 2023) juga membandingkan *Logistic Regression*, *Random Forest*, AdaBoost, dan XGBoost, dengan *Random Forest* menunjukkan ROC-AUC tertinggi. (Hewitt & Karakuş, 2023) menggunakan *Logistic Regression* dan *Gradient Boosted Trees* dengan SHAP. (P. M. Pardo, 2020) membandingkan *Logistic Regression,* XGBoost, dan *Artificial Neural Network* (ANN). (Bandara *et al*., 2024) menggunakan *Random Forest* dengan fitur urutan kejadian temporal. Terakhir, (Matteotti & Sotudeh, 2024) mengeksplorasi *Convolutional Neural Networks* (CNNs) untuk ekstraksi spasial otomatis.

Penelitian ini bertujuan untuk menerapkan LightGBM untuk perhitungan metrik xG dalam analisis sepak bola. Perbedaan penelitian ini dengan penelitian sebelumnya diantaranya yaitu:

1. Fokus pada penggunaan LightGBM sebagai metode utama untuk perhitungan xG, yang dikenal karena efisiensi dan kecepatan komputasinya.
2. Mengeksplorasi kemampuan LightGBM dalam menangani data sepak bola, yang sering kali memiliki interaksi fitur kompleks dan non-linear, dibandingkan dengan metode yang lebih sederhana seperti regresi logistik.