BAB 2
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penelitian oleh Nugraha et al. (2022) yang berjudul "Penerapan Metode *Stacking* dan *Random Forest* untuk Meningkatkan Kinerja Klasifikasi pada Proses Deteksi Web *Phishing*" berfokus pada pengaruh ketidakseimbangan kelas dalam data terhadap performa klasifikasi. Untuk mengatasi permasalahan ini, mereka mengusulkan pendekatan *ensemble learning* yang menggabungkan algoritma *Decision Tree* dan *Naïve Bayes* dalam metode *Stacking*, serta memanfaatkan *Random Forest* guna meningkatkan akurasi klasifikasi tanpa mengubah struktur *dataset*. Hasil penelitian menunjukkan bahwa *Random Forest* mencapai akurasi sebesar 96,4% pada *dataset* web *phishing binary class*, sementara metode *Stacking* memberikan hasil terbaik pada *dataset multiclass* dengan akurasi 88,8%. Meskipun demikian, penelitian ini memiliki beberapa kelemahan, antara lain belum melakukan uji signifikansi statistik untuk memastikan perbedaan performa antar model, belum membandingkan hasil dengan metode penanganan ketidakseimbangan data lain seperti *SMOTE* atau *ADASYN*, terbatasnya variasi kombinasi *base learner* pada *Stacking*, tidak adanya analisis fitur yang paling berpengaruh, serta belum diuji pada *dataset real-world* atau data dari sumber berbeda untuk mengukur kemampuan generalisasi.

Selanjutnya, penelitian yang dilakukan oleh (Tarigan et al., 2022) bermakna "Optimasi Algoritma *Iterative Dichotomiser 3* Kombinasi Metode MinMax - *Information Gain* untuk Deteksi *Website Phising*" bertujuan mengoptimalkan algoritma ID3 dalam mendeteksi situs web *phishing*. Penelitian ini menggabungkan teknik *Min-Max Scaling* untuk normalisasi data dan *Information Gain* untuk seleksi fitur. Hasil seleksi menunjukkan bahwa beberapa fitur, seperti *Favicon*, *Iframe*, dan *popUpWindow*, memiliki nilai *Information Gain* sebesar 0 sehingga dihapus karena tidak memberikan kontribusi signifikan terhadap prediksi. Pengujian menggunakan metode 8-*fold cross-validation* menunjukkan bahwa model ID3 dengan 27 fitur terbaik

memiliki akurasi sebesar 98,56%, presisi 98,63%, dan *recall* 98,78%, sedikit lebih baik dibandingkan model dengan seluruh 30 fitur. Meskipun demikian, penelitian ini memiliki beberapa keterbatasan, antara lain peningkatan kinerja yang dihasilkan relatif sangat kecil dan tidak diuji signifikansi statistiknya, hanya berfokus pada algoritma ID3 tanpa perbandingan dengan model yang lebih canggih seperti *Random Forest* atau *Gradient Boosting*, penggunaan *dataset* tunggal dari UCI tanpa pengujian pada data dunia nyata, tidak membahas potensi ketidakseimbangan kelas, serta evaluasi kinerja yang terbatas pada tiga metrik (akurasi, presisi, dan *recall*) tanpa menyertakan metrik lain seperti *F1-score* atau *AUC-ROC* yang relevan untuk deteksi *phishing*.

Penelitian lainnya dilakukan oleh (Rumini et al., 2023) dengan judul "Perbandingan Pengujian Deteksi Phising Menggunakan Metode SVM dengan Kernel RBF dan Linear". Mereka membandingkan performa dua jenis kernel dalam metode *Support Vector Machine* (SVM), yaitu kernel *Radial Basis Function* (RBF) dan kernel *Linear*, dalam mendeteksi *website phishing*. Tujuan utama penelitian ini adalah untuk menentukan metode yang paling efektif dalam mendeteksi *website phishing* guna meningkatkan keamanan pengguna dari ancaman pencurian data. Hasil penelitian menunjukkan bahwa SVM dengan kernel RBF memiliki akurasi tertinggi sebesar 96,426%, sedangkan kernel *Linear* mencapai akurasi sebesar 92,582%. Dengan demikian, kernel RBF dinilai lebih efektif dibandingkan kernel *Linear* dalam mendeteksi *website phishing*. Meskipun demikian, penelitian ini memiliki sejumlah kelemahan, antara lain penggunaan *dataset* yang terbatas pada satu sumber (UCI *Machine Learning Repository*) tanpa uji pada data dunia nyata, tidak adanya pembandingan dengan metode klasifikasi lain di luar SVM, tidak dilakukan pengujian signifikansi statistik atas perbedaan hasil akurasi, evaluasi performa yang terbatas pada akurasi dan *confusion matrix* tanpa menyertakan metrik lain seperti *F1-score* atau *AUC-ROC*, tidak dibahasnya potensi ketidakseimbangan kelas pada *dataset*, serta ketiadaan proses optimasi parameter (*hyperparameter tuning*) yang dapat memengaruhi kinerja SVM.

Penelitian lain dilakukan oleh (Melvin & Soraya, 2023) yang berjudul "Analisis Perbandingan Algoritma XGBoost dan Algoritma Random Forest Ensemble Learning

pada Klasifikasi Keputusan Kredit". Penelitian ini bertujuan membantu bank atau lembaga keuangan dalam meningkatkan keakuratan penilaian kredit agar dapat mengurangi risiko kredit macet dan kesalahan dalam pengambilan keputusan pemberian kredit. Hasil penelitian menunjukkan bahwa *XGBoost* memiliki performa lebih unggul dibandingkan *Random Forest*, terutama pada *dataset* yang kecil dan tidak seimbang (*class imbalance*). Meskipun *Random Forest* memiliki akurasi tinggi, algoritma ini kurang optimal saat menangani *dataset* kecil. Namun, penelitian ini memiliki sejumlah kelemahan, antara lain penggunaan *dataset* yang terbatas pada satu sumber tanpa uji pada data dunia nyata, tidak adanya analisis signifikansi statistik untuk memastikan perbedaan performa kedua algoritma, minimnya variasi metode pembanding karena hanya fokus pada *XGBoost* dan *Random Forest*, ketiadaan penjelasan rinci mengenai proses *hyperparameter tuning* yang dapat memengaruhi hasil, tidak dibahasnya distribusi kelas secara mendalam untuk mengatasi masalah *class imbalance*, evaluasi kinerja yang terbatas pada metrik akurasi, presisi, dan *recall* tanpa menyertakan metrik lain seperti *F1-score* atau *AUC-ROC*, serta ketiadaan analisis kesalahan prediksi (*false positive* dan *false negative*) yang dapat memberikan wawasan lebih terhadap risiko kredit.

Sementara itu, (Windarni et al., 2023) melalui penelitian berjudul "Deteksi Website Phishing Menggunakan Teknik Filter pada Model Machine Learning" mencoba meningkatkan deteksi *website phishing* dengan menggunakan teknik filter *Pearson correlation*. Mereka menerapkan tiga metode klasifikasi, yaitu *Naïve Bayes*, *Decision Tree*, dan *Random Forest* untuk menentukan model yang paling efektif dalam mengidentifikasi *phishing*. Hasil penelitian menunjukkan bahwa metode *Naïve Bayes* memiliki akurasi sebesar 60,4%, *Decision Tree* mencapai 94,4%, dan *Random Forest* memiliki akurasi tertinggi yaitu 96,3%. Dengan demikian, *Random Forest* dinilai paling efektif dalam mendeteksi *website phishing* berdasarkan *dataset* yang digunakan. Namun, penelitian ini memiliki sejumlah kelemahan, antara lain penggunaan *dataset* yang terbatas pada satu sumber (UCI *Machine Learning Repository*) tanpa pengujian pada data dunia nyata, tidak adanya analisis mengenai potensi ketidakseimbangan kelas yang dapat memengaruhi akurasi, evaluasi performa yang hanya berfokus pada akurasi

tanpa menyertakan metrik penting lain seperti presisi, *recall*, *F1-score*, atau *AUC-ROC*, minimnya variasi metode pembanding karena hanya menguji tiga algoritma, penggunaan teknik seleksi fitur yang sederhana dengan *Pearson correlation* tanpa membandingkan dengan metode seleksi lain, tidak adanya penjelasan mengenai *hyperparameter tuning* yang dapat memengaruhi hasil, serta ketiadaan analisis kesalahan prediksi untuk memahami jenis kesalahan yang sering terjadi.

Terakhir, penelitian yang dilakukan oleh (ZUFAR, 2023) dengan judul "Aplikasi Pendeteksi Situs *Phising* Berbasis *Website* Menggunakan Metode Naïve Bayes" mengembangkan sebuah aplikasi berbasis web untuk mendeteksi situs *phishing*. Penelitian ini melibatkan pengumpulan data dari situs web yang diklasifikasikan sebagai aman dan berbahaya, kemudian melakukan *preprocessing*, pelatihan model dengan metode *Naïve Bayes*, dan evaluasi akurasi model. Hasilnya menunjukkan bahwa metode ini mampu mendeteksi situs *phishing* dengan tingkat akurasi yang tinggi, sehingga potensial digunakan sebagai alat bantu dalam meningkatkan keamanan siber. Meskipun demikian, penelitian ini memiliki beberapa kelemahan, antara lain penggunaan metode tunggal (*Naïve Bayes*) tanpa perbandingan dengan algoritma lain yang mungkin memberikan hasil lebih baik, keterbatasan *dataset* yang digunakan sehingga generalisasi model belum teruji pada data dunia nyata, tidak adanya pembahasan mengenai distribusi kelas atau potensi ketidakseimbangan data, evaluasi kinerja yang terbatas pada akurasi tanpa menyertakan metrik lain seperti presisi, *recall*, *F1-score*, atau *AUC-ROC*, ketiadaan analisis kesalahan prediksi untuk mengidentifikasi pola *false positive* dan *false negative*, serta tidak adanya penjelasan rinci terkait proses *hyperparameter tuning* yang dapat memengaruhi kinerja model.

Berdasarkan uraian penelitian terdahulu, dapat disimpulkan bahwa berbagai penelitian telah berhasil menerapkan algoritma *machine learning* untuk mendeteksi website *phishing* dengan tingkat akurasi yang tinggi. Namun, masih terdapat beberapa keterbatasan, seperti penggunaan dataset tunggal tanpa pengujian pada data dunia nyata, evaluasi model yang belum menggunakan metrik secara komprehensif, minimnya analisis terhadap kesalahan klasifikasi, serta belum adanya implementasi *Explainable Artificial Intelligence* (XAI) untuk menjelaskan alasan di balik hasil

prediksi model. Oleh karena itu, penelitian ini menggunakan algoritma XGBoost untuk membangun sistem deteksi URL *phishing* berbasis web yang mampu melakukan deteksi secara *real-time*, mengevaluasi performa model menggunakan metrik *accuracy*, *precision*, *recall*, *F1-score*, *confusion matrix*, dan *ROC-AUC*, serta mengimplementasikan metode SHAP sebagai bagian dari XAI untuk memberikan penjelasan mengenai kontribusi setiap fitur terhadap hasil prediksi sehingga proses pengambilan keputusan model menjadi lebih transparan dan mudah dipahami oleh pengguna.

Tabel 2. 1 Perbandingan variabel penelitian

Kategori	Peneliti 1	Peneliti 2	Peneliti 3	Peneliti 4	Peneliti 5
Penelitian	Nugraha, A. F., Aziza, R. F. A., & Pristyanto, Y. (2022)	Victor Tarigan, Jeremia Siregar, Adinda Franky Nelwan, Reinhard Komansilan, Ade Yusupa (2022)	Rumini, Norhikmah, Ali Mustofa, Sulistyo Pradana (2022)	Jan Melvin Ayu Soraya Dachi, Pardomuan Sitompul (2023)	Vikky A. Windarni, Anggit F. Nugraha, Surya T. A. Ramadhani, Dewi A. Istiqomah, Fiyas M. Puri, Adi Setiawan (2023)
Judul	Penerapan Metode Stacking dan Random Forest untuk Meningkatkan Kinerja Klasifikasi pada Proses Deteksi Web Phishing	Optimasi Algoritma Iterative Dichotomiser 3 Kombinasi Metode Minmax - Information Gain Untuk Deteksi Website Phising	Perbandingan Pengujian Deteksi Phising menggunakan Metode SVM dengan Kernel RBF dan Linear	Analisis Perbandingan Algoritma XGBoost dan Algoritma Random Forest Ensemble Learning pada Klasifikasi Keputusan Kredit	Deteksi Website Phishing Menggunakan Teknik Filter Pada Model Machine Learning
Metode	Random Forest ($n = 11$) & Stacking Ensemble (Base: Decision Tree + Naïve Bayes).	Iterative Dichotomiser 3 (ID3) dioptimasi dengan Min-Max Normalization & Seleksi Fitur Information Gain (IG).	Support Vector Machine (SVM) dengan melakukan perbandingan performa antara dua jenis kernel,	Perbandingan algoritma *Ensemble Learning* antara XGBoost (metode *boosting*) dan Random Forest (metode	Perbandingan algoritma klasifikasi machine learning antara Naïve Bayes, Decision Tree, dan Random

Kategori	Peneliti 1	Peneliti 2	Peneliti 3	Peneliti 4	Peneliti 5
			yaitu Kernel Linear dan Kernel Radial Basis Function (RBF).	*bagging*) dengan optimasi *parameter tuning* menggunakan Random Search.	Forest dengan menerapkan teknik reduksi fitur berbasis Pearson Correlation.
Pengujian	*Data splitting* 80:20, Confusion Matrix (Akurasi, Presisi, Recall, F1-Score, AUC).	8-Fold Cross-Validation, Perbandingan model 30 fitur vs 27 fitur (Akurasi, Presisi, Recall).	*Data splitting* dengan proporsi 80% untuk *train set* dan 20% untuk *test set*, dievaluasi menggunakan *Confusion Matrix* (kalkulasi nilai akurasi langsung).	*Pre-processing* (Label Encoding & Min-Max Normalization), *data splitting* dengan rasio 70% data *training* : 30% data *testing*, dan evaluasi menggunakan *Confusion Matrix* (Akurasi, Presisi, Recall, F1-Score).	*Data splitting* dengan rasio 70% data *training* : 30% data *testing*, diuji menggunakan 4 tahapan alur (persiapan, metode, analisis, evaluasi) untuk menilai tingkat akurasi model.
Data	Dataset UCI Machine Learning: *Binary Class*: 2.456 sampel, 31 atribut, IR 1,25%. *Multiclass*: 1.353 sampel, 10 atribut, IR 6,82%.	Dataset UCI Machine Learning, 22.110 sampel. Reduksi atribut dari 30 fitur menjadi 27 fitur terbaik.	Dataset publik dari UCI Machine Learning Repository dengan jumlah total sebanyak 11.056 sampel data.	Dataset sekunder *"Credit Card Approval - With Target"* dari Kaggle dengan 19 variabel. Pengujian menggunakan dua skala ukuran sampel data secara acak, yaitu 10.000 data dan 100.000 data.	Dataset *benchmark* dari UCI Machine Learning Repository dengan total 11.055 sampel data. Teknik Pearson Correlation memangkas jumlah atribut dari 30 fitur menjadi 12 fitur terbaik yang memiliki nilai korelasi > 0.1.

Kategori	Peneliti 1	Peneliti 2	Peneliti 3	Peneliti 4	Peneliti 5
Hasil Penelitian	*Binary*: Random Forest terbaik (Akurasi 96,4%, AUC 99,3%). *Multiclass*: Stacking terbaik (Akurasi 88,8%, AUC 96,1%).	Model 30 fitur: Akurasi 98,54%, Presisi 98,60%, Recall 98,78%. Model 27 fitur: Akurasi 98,56%, Presisi 98,63%, Recall 98,78%.	*SVM Kernel Linear*: Menghasilkan akurasi sebesar 92,582% *SVM Kernel RBF*: Menghasilkan akurasi tertinggi sebesar 96,426% (atau 96,06% berdasarkan matriks rincian).	XGBoost: Stabil dan akurat dengan nilai 1.0 (100%) pada semua metrik di kedua skala data (10k & 100k). Random Forest: Akurat pada data besar (100k: Akurasi 0.999, F1 0.995), namun anjlok pada data kecil (10k: F1 0.700, Recall 0.538).	*Naïve Bayes*: Menghasilkan nilai akurasi paling rendah sebesar 60,4%. *Decision Tree*: Menghasilkan nilai akurasi sebesar 94,4%. *Random Forest*: Menghasilkan nilai akurasi tertinggi sebesar 96,3%.
Temuan Penelitian	*Ensemble* efektif menangani *imbalanced class* tanpa mengubah komposisi data asli. Disarankan menguji *hybrid method* dengan teknik *resampling*.	Mengeliminasi 3 fitur *noise* bernilai IG = 0 (*Favicon, Iframe, popUpWindow*). Model menjadi lebih ramping, efisien, dan memangkas waktu komputasi.	Kernel RBF terbukti lebih unggul dan akurat dibandingkan kernel linear dalam menangani karakteristik data deteksi web phishing yang bersifat *non-linear*.	XGBoost unggul dan konsisten di berbagai ukuran data. Random Forest kurang akurat pada data kecil yang tidak seimbang (imbalanced data).	Random Forest terbukti paling efektif dan stabil dengan akurasi tertinggi sebesar 96,3% setelah melalui pemfilteran fitur.

Berdasarkan kajian terhadap beberapa penelitian terdahulu pada Tabel 2.1, dapat diketahui bahwa sebagian besar penelitian membahas penerapan algoritma *machine learning* untuk menyelesaikan permasalahan klasifikasi, khususnya pada kasus deteksi *website phishing*. Berbagai metode yang digunakan antara lain *Random Forest*, *Stacking Ensemble*, *ID3*, *Support Vector Machine* (SVM), dan XGBoost yang menunjukkan tingkat akurasi yang cukup tinggi pada masing-masing penelitian. Namun, sebagian besar penelitian tersebut berfokus pada perbandingan algoritma,

optimasi parameter, atau seleksi fitur untuk meningkatkan performa model, serta masih memanfaatkan atribut yang tidak seluruhnya dapat diperoleh secara langsung dari URL.

Penelitian ini memiliki perbedaan dengan penelitian sebelumnya karena berfokus pada pengembangan sistem deteksi *website phishing* menggunakan algoritma XGBoost dengan memanfaatkan fitur-fitur yang sepenuhnya berasal dari karakteristik URL. Pendekatan tersebut dipilih agar proses deteksi dapat dilakukan secara *real-time* hanya berdasarkan URL yang dimasukkan oleh pengguna tanpa memerlukan pengambilan maupun analisis terhadap konten halaman *web*. Selain itu, penelitian ini juga mempertimbangkan kemampuan model dalam mengenali URL di luar data pelatihan sehingga lebih mendekati kondisi penggunaan pada lingkungan nyata. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan sistem deteksi *website phishing* yang lebih praktis, efisien, dan mudah diimplementasikan pada berbagai aplikasi berbasis *web* maupun layanan keamanan siber.

2.2 Landasan Teori

2.2.1 Cyber Security

Keamanan siber (*cyber security*) merupakan upaya untuk melindungi sistem, jaringan, dan data dari serangan digital yang dapat mengganggu kerahasiaan, integritas, dan ketersediaan informasi (CIA Triad). Menurut (Nurse, 2019), *cyber security* tidak hanya melindungi perangkat keras dan perangkat lunak, tetapi juga memperhatikan faktor manusia yang sering menjadi titik lemah dalam berbagai serangan. Dengan semakin meningkatnya ketergantungan masyarakat terhadap teknologi digital dan internet, maka penting untuk memahami berbagai bentuk ancaman siber yang terus berkembang.

Jenis-jenis ancaman dalam keamanan siber antara lain:

1. *Social Engineering* dan *Phishing*

   Teknik manipulasi psikologis yang bertujuan untuk menipu korban agar memberikan informasi sensitif. Contohnya seperti *phishing email*, penipuan melalui SMS (*smishing*), dan panggilan suara (*vishing*).

2. Penipuan Daring *(Online Scams)*

   Termasuk penipuan bantuan teknis, penipuan asmara (*romance scam*), dan *catfishing*, yang memanfaatkan kepercayaan atau emosi korban.

3. Pelecehan Daring *(Online Harassment)*

   Meliputi *cyberbullying, cyberstalking*, dan *sextortion*, yang dapat berdampak buruk secara psikologis.

4. Pencurian Identitas *(Identity Theft)*

   Pelaku mengakses dan menyalahgunakan data pribadi korban, seringkali melalui media sosial atau kebocoran data.

5. Peretasan dan *Malware*

   Termasuk penggunaan *virus, trojan, spyware*, dan *cryptojacking* untuk mencuri data atau merusak sistem.

6. Serangan DoS dan *Ransomware*

   Serangan yang mengganggu layanan digital atau mengenkripsi data dan meminta tebusan agar dapat diakses kembali.

2.2.2 Phishing

Phishing merupakan salah satu bentuk serangan siber yang tergolong dalam teknik rekayasa sosial *(social engineering*), yaitu upaya untuk memanipulasi psikologis korban agar memberikan informasi pribadi atau rahasia seperti kata sandi, nomor kartu kredit, maupun data sensitif lainnya. Menurut (Arshad et al., 2021), *phishing* dilakukan dengan cara menyamar sebagai pihak yang terpercaya melalui media seperti surat elektronik (email), pesan singkat (SMS), panggilan telepon, atau situs web palsu.

Jenis-jenis serangan *phishing* yang umum ditemukan antara lain:

1. *Spear Phising*

   Serangan phishing yang ditargetkan secara khusus kepada individu atau organisasi tertentu. Penyerang biasanya mempersonalisasi pesan agar tampak lebih meyakinkan.

2. *Whaling*

Merupakan bentuk spear phishing yang menargetkan pejabat tinggi atau eksekutif perusahaan, dengan tujuan memperoleh akses terhadap sistem penting atau informasi strategis.

3. *Smishing dan Vishing*

   Smishing adalah phishing yang dilakukan melalui pesan singkat (SMS), sedangkan *vishing* dilakukan melalui percakapan telepon. Keduanya bertujuan untuk menipu korban agar memberikan informasi pribadi.

4. *Business Email Compromise (BEC)*

   Serangan yang menyamar sebagai pihak internal perusahaan melalui email bisnis palsu, guna mengelabui korban agar melakukan transfer dana atau memberikan data sensitif.

5. *Website Spoofing (Web Phishing)*

   Teknik pemalsuan situs web yang dirancang menyerupai situs resmi. Korban diarahkan untuk mengunjungi tautan palsu dan memasukkan data pribadi tanpa menyadari bahwa situs tersebut tidak sah.

2.2.3 Machine Learning

Machine Learning atau pembelajaran mesin merupakan salah satu cabang dari kecerdasan buatan yang fokus utamanya adalah pada pengembangan algoritma yang mampu mempelajari pola dari data dan membuat prediksi tanpa diprogram secara eksplisit untuk setiap tugas tertentu. Proses pembelajaran dilakukan dengan memanfaatkan data historis sebagai dasar untuk menemukan pola atau keterkaitan antar fitur dalam data. Setelah dilatih, model *machine learning* mampu melakukan generalisasi untuk menghasilkan prediksi yang akurat terhadap data baru yang belum pernah dilihat sebelumnya (Mahesh, 2020).

Jenis-jenis serangan *phishing* yang umum ditemukan antara lain:

1. *Supervised Learning* (Pembelajaran Terbimbing)

   Supervised Learning (Pembelajaran Terbimbing) adalah metode *machine learning* yang menggunakan data berlabel, di mana setiap data latih telah memiliki output atau kategori yang diketahui. Tujuan utama dari pendekatan ini adalah untuk membangun sebuah model yang mampu memprediksi atau mengklasifikasikan

data baru berdasarkan pola yang dipelajari dari data latih tersebut. Dalam penerapannya, terdapat beberapa algoritma yang umum digunakan, di antaranya adalah *Decision Tree,* yang memodelkan proses pengambilan keputusan dalam bentuk struktur pohon; *Naïve Bayes,* yang didasarkan pada *Teorema Bayes* dengan asumsi bahwa fitur-fitur yang digunakan bersifat saling independen; serta *Support Vector Machine (SVM)*, yang bekerja dengan menentukan margin optimal yang memisahkan kelas-kelas dalam data untuk menghasilkan klasifikasi yang akurat.

2. *Unsupervised Learning* (Pembelajaran Tak Terbimbing)

   Unsupervised Learning (Pembelajaran Tak Terbimbing) adalah metode *machine learning* yang digunakan pada data yang tidak memiliki label atau kategori yang diketahui sebelumnya. Tujuan utama dari pendekatan ini adalah untuk menemukan struktur, pola, atau hubungan tersembunyi di dalam data tanpa bantuan informasi output. Beberapa algoritma yang sering digunakan dalam *unsupervised learning* antara lain *K-Means Clustering*, yang mengelompokkan data ke dalam beberapa klaster berdasarkan kemiripan karakteristiknya, serta *Principal Component Analysis (PCA)*, yang berfungsi untuk mereduksi dimensi data dengan tetap mempertahankan informasi penting, sehingga analisis data menjadi lebih sederhana dan efisien.

3. Semi-Supervised Learning

   Semi-Supervised Learning merupakan pendekatan *machine learning* yang menggabungkan metode *supervised* dan *unsupervised* learning. Pendekatan ini sangat cocok digunakan ketika tersedia sejumlah kecil data berlabel dan sejumlah besar data tidak berlabel, yang umum terjadi dalam dunia nyata karena pelabelan data sering kali memerlukan waktu dan biaya. *Semi-supervised learning* memanfaatkan informasi dari data berlabel untuk membantu memahami struktur dari data tidak berlabel, sehingga dapat meningkatkan akurasi model. Contoh algoritma dalam pendekatan ini adalah *Transductive SVM*, yang memperluas penggunaan *Support Vector Machine* untuk data parsial, dan *Self-Training*, di mana model dilatih pada data berlabel, lalu digunakan untuk memprediksi label dari data tidak berlabel, dan hasil prediksi tersebut ditambahkan kembali ke dalam pelatihan.

4. Reinforcement Learning

   Reinforcement Learning adalah pendekatan pembelajaran di mana agen (model) belajar melalui interaksi langsung dengan lingkungan. Proses pembelajaran dilakukan dengan menerima hadiah *(reward)* atau hukuman *(penalty)* berdasarkan tindakan yang diambil, dan bertujuan untuk memaksimalkan total hadiah kumulatif dalam jangka panjang. Metode ini banyak digunakan dalam aplikasi seperti permainan *(game),* robotika, dan sistem rekomendasi, di mana pengambilan keputusan secara bertahap dan optimal sangat dibutuhkan.

Proses implementasi *machine learning* dalam bidang data science merupakan tahapan penting dalam pengembangan sistem cerdas berbasis data. Berdasarkan penelitian oleh (Nur Fajri et al., 2022), proses ini melibatkan serangkaian langkah sistematis yang dimulai dari pengumpulan data hingga evaluasi performa model. Tahapan-tahapan tersebut dapat dijelaskan sebagai berikut:

1. *Reinforcement Learning*

   Tahap awal dimulai dengan mengumpulkan data mentah yang relevan. Dalam konteks penelitian yang dilakukan, data dikumpulkan dari nilai mata kuliah mahasiswa dari semester satu hingga enam. Data ini berperan penting dalam membentuk dasar pembelajaran bagi model *machine learning* yang akan dibangun.

2. *Pra-pemrosesan Data (Preprocessing)*

   Setelah data terkumpul, dilakukan pra-pemrosesan untuk memastikan kualitas dan konsistensi data. Proses ini meliputi penghapusan missing values, pengkodean label, normalisasi nilai ke dalam rentang tertentu (misalnya 0 hingga 1), serta pemisahan antara data masukan (fitur) dan data target (label). Pra-pemrosesan bertujuan agar data siap digunakan dalam pelatihan model dan menghasilkan prediksi yang optimal.

3. *Implementasi Model Machine Learning*

   Data yang telah diproses selanjutnya digunakan dalam proses pelatihan model. Pada penelitian tersebut, metode *Neural Network* diimplementasikan menggunakan Google Colab dengan bahasa pemrograman *Python*. Model terdiri dari beberapa

lapisan neuron dengan fungsi aktivasi ReLU dan lapisan *output* dengan fungsi aktivasi sigmoid, karena klasifikasi yang dilakukan bersifat biner.

4. Pelatihan dan Evaluasi Model

   Model kemudian dilatih menggunakan data latih dan diuji menggunakan data uji untuk mengukur performa. Parameter yang diamati meliputi akurasi dan loss. Hasil menunjukkan bahwa model memperoleh akurasi sebesar 83% pada data pelatihan dan 79% pada data pengujian. Evaluasi dilakukan menggunakan metrik seperti confusion matrix untuk mengukur ketepatan klasifikasi.

5. Penggunaan Model

   Setelah proses pelatihan dan evaluasi selesai, model dapat digunakan untuk melakukan prediksi terhadap data baru. Dalam kasus penelitian ini, model digunakan untuk merekomendasikan pemilihan mata kuliah pilihan kepada mahasiswa berdasarkan riwayat nilai mereka, sehingga membantu dalam pengambilan keputusan yang lebih tepat sesuai minat dan kemampuan. Dengan mengikuti langkah-langkah tersebut, implementasi *machine learning* dalam bidang data *science* dapat dilakukan secara sistematis dan menghasilkan sistem yang cerdas, adaptif, serta memberikan dampak nyata dalam pengambilan keputusan berbasis data.

2.2.4 Extreme Gradient Boosting (XGBoost)

Menurut (Chen & Guestrin, 2016), *Extreme Gradient Boosting* (XGBoost) merupakan salah satu algoritma *ensemble learning* berbasis pohon keputusan yang dirancang secara khusus untuk mencapai efisiensi, fleksibilitas, dan kinerja yang tinggi. XGBoost merupakan implementasi yang telah dioptimalkan dari *Gradient Boosting Decision Tree* (GBDT), di mana beberapa pohon keputusan dibangun secara bertahap dan setiap pohon baru ditujukan untuk mengoreksi kesalahan dari gabungan pohon sebelumnya dengan menggunakan pendekatan *gradient descent*.

XGBoost mendukung berbagai tipe data *input*, seperti atribut kategorikal, ordinal, maupun numerik, serta memiliki kemampuan untuk menangani data yang mengandung nilai hilang (*missing value*). Algoritma ini juga dilengkapi dengan fitur *regularization*, baik L1 maupun L2, yang berfungsi untuk mengurangi risiko

overfitting. Selain itu, XGBoost mendukung komputasi paralel (*parallel computing*), *pruning* otomatis untuk memangkas cabang yang tidak informatif, serta fitur *early stopping* untuk menghentikan proses pelatihan apabila tidak terdapat peningkatan performa pada data validasi.

1. Fungsi Tujuan (*Objective Function*)

   XGBoost mengoptimalkan fungsi tujuan yang terdiri atas dua komponen utama:

   a) Fungsi Kehilangan (*Loss Function*): Komponen ini mengukur perbedaan antara nilai prediksi dan nilai aktual dari data pelatihan. Contoh fungsi kehilangan yang umum digunakan adalah *mean squared error* (MSE) untuk regresi dan *log-loss* untuk klasifikasi.

   b) Regularisasi (*Regularization*): Fungsi ini berperan dalam mengontrol kompleksitas model agar tidak mengalami *overfitting*. Dengan menambahkan penalti terhadap model yang terlalu kompleks, regularisasi membantu menghasilkan model yang lebih generalisasi terhadap data baru.

   Rumus fungsi tujuan dapat ditulis sebagai berikut:

$$Obj(\theta) = \sum_{i=0}^{n} l(y_i, y_i) + \sum_{k=1}^{k} \Omega(f_k) \tag{2.1}$$

   Keterangan:

   - Obj(θ): Fungsi tujuan yang akan diminimalkan.
   - $l(y_i, \hat{y}_i)$: Fungsi kehilangan antara nilai aktual y_i dan prediksi model $\hat{y}_i$.
   - $\Omega(f_k)$: Fungsi regularisasi dari pohon ke-k.
   - n: Jumlah data.
   - K: Jumlah pohon yang dibentuk.
   - f_k: Fungsi prediktif dari pohon ke-k.

   Fungsi regularisasi dapat dirumuskan sebagai berikut:

$$\Omega\,(f_k) = \gamma T + \frac{1}{2}\lambda||\omega||^2 \tag{2.2}$$

   Keterangan:

- γ: Parameter untuk mengontrol jumlah daun pada pohon.
- T: Jumlah daun pada pohon keputusan.
- λ: Parameter regularisasi bobot.
- w: Vektor bobot pada setiap daun.

2. Proses Pembangunan Pohon

XGBoost membangun model prediktif secara iteratif. Pada setiap iterasi, algoritma akan menambahkan sebuah pohon keputusan yang ditujukan untuk memperbaiki kesalahan prediksi dari iterasi sebelumnya. Prediksi model pada iterasi ke-t dinyatakan sebagai:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i) \tag{2.3}$$

Untuk mengarahkan proses pembelajaran, XGBoost menghitung turunan pertama (*gradien*) dan turunan kedua (*hessian*) dari fungsi kehilangan:

$$g_i = \frac{\partial(y_i, \hat{y}i^{(t-1)}}{\partial \hat{y}i^{(t-1)}}, h_i = \frac{\partial^2(y_i, \hat{y}i^{(t-1)}}{\partial \hat{y}i^{(t-1)^2}} \tag{2.4}$$

Selanjutnya, fungsi tujuan pada iterasi ke-t dapat didekati sebagai:

$$Obj^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \right] + \Omega(f_t) \tag{2.5}$$

3. Pemilihan Split dan Nilai Gain

Setiap node pohon akan dibagi berdasarkan nilai gain, yaitu keuntungan yang diperoleh dari melakukan split:

$$Gain = \frac{1}{2}\left[\frac{\sum(i\in I_L\, g_i))^2}{\sum i\in I_L\, h_i+\lambda} + \frac{(\sum i\in I_R\, g_i)^2}{\sum i\in I_R\, h_{i+\lambda}} - \frac{(\sum i\in I\, g_i)^2}{\sum i\in I\ \ h^i+\lambda}\right] - \gamma \tag{2.6}$$

Keterangan:

- Gain: Nilai keuntungan dari split.
- I_L, I_R: Indeks data pada cabang kiri dan kanan setelah split.
- I: Indeks seluruh data pada node sebelum split.
- λ: Parameter regularisasi.

- γ: Penalti kompleksitas untuk setiap split.
- g_i: Gradient data ke-i.
- h_i: Hessian data ke-i.

4. Tahap Pelatihan dan Pengujian

   Secara umum, proses kerja XGBoost terdiri atas dua tahap:

   a) Pelatihan (*Training Phase*): Model dibangun melalui iterasi dengan menambahkan pohon-pohon keputusan untuk mengurangi kesalahan prediksi.
   b) Pengujian (*Testing Phase*): Model yang telah terbentuk digunakan untuk memprediksi data baru, dan hasil prediksi dibandingkan dengan nilai aktual.

   Evaluasi terhadap performa model dilakukan menggunakan metrik seperti *accuracy, precision*, *recall, F1-score*, atau *area under curve* (AUC), tergantung pada jenis masalah yang dihadapi (klasifikasi atau regresi). Untuk menghindari bias evaluasi, teknik validasi silang seperti *k-fold cross-validation* sering diterapkan.

2.2.5 Explainable Artificial Intelligence (XAI)

Explainable Artificial Intelligence (XAI) merupakan suatu pendekatan dalam bidang kecerdasan buatan yang bertujuan untuk meningkatkan transparansi dan interpretabilitas model *machine learning*, sehingga proses pengambilan keputusan yang dilakukan oleh model dapat dipahami oleh manusia. Berbeda dengan model kecerdasan buatan konvensional yang umumnya hanya memberikan hasil prediksi tanpa menjelaskan alasan di balik keputusan tersebut, XAI mampu memberikan informasi mengenai faktor-faktor yang memengaruhi hasil prediksi. Dengan demikian, pengguna tidak hanya mengetahui hasil akhir yang diberikan oleh sistem, tetapi juga dapat memahami dasar pengambilan keputusan yang dilakukan oleh model. Menurut (Minh et al., 2022), *Explainable Artificial Intelligence* dikembangkan untuk mengatasi keterbatasan model *machine learning* modern yang memiliki tingkat akurasi tinggi, tetapi sulit dijelaskan cara kerjanya kepada pengguna.

Perkembangan metode *machine learning*, khususnya algoritma yang memiliki struktur kompleks seperti *ensemble learning* dan *deep learning*, telah meningkatkan kemampuan sistem dalam melakukan prediksi maupun klasifikasi. Namun, peningkatan performa tersebut sering kali diikuti dengan menurunnya tingkat

transparansi model. Kondisi ini dikenal sebagai konsep *black box*, yaitu keadaan ketika model hanya menampilkan hasil akhir tanpa memberikan informasi mengenai proses atau alasan yang mendasari keputusan tersebut. Akibatnya, pengguna mengalami kesulitan dalam memahami, mengevaluasi, maupun memverifikasi hasil yang dihasilkan oleh sistem. Oleh karena itu, *Explainable Artificial Intelligence* dikembangkan sebagai solusi untuk memberikan penjelasan yang lebih mudah dipahami sehingga mampu meningkatkan kepercayaan pengguna terhadap sistem berbasis kecerdasan buatan (Minh et al., 2022).

Secara umum, *Explainable Artificial Intelligence* memiliki beberapa tujuan utama, yaitu meningkatkan transparansi model, mempermudah interpretasi hasil prediksi, mendukung proses pengambilan keputusan, serta meningkatkan kepercayaan pengguna terhadap sistem. Selain itu, XAI juga berperan dalam membantu pengembang mengidentifikasi kesalahan model, mendeteksi potensi *bias* pada data maupun algoritma, serta mempermudah proses evaluasi dan pengembangan model *machine learning*. Dengan adanya penjelasan mengenai alasan suatu prediksi dihasilkan, pengguna dapat memahami faktor-faktor yang memengaruhi keputusan model sehingga hasil prediksi menjadi lebih mudah dipahami dan dipercaya (Minh et al., 2022).

Menurut (Minh et al., 2022), metode *Explainable Artificial Intelligence* dapat dikelompokkan ke dalam beberapa kategori berdasarkan karakteristik penjelasannya. Salah satu pengelompokan yang umum digunakan adalah berdasarkan cakupan penjelasan, yaitu *global explanation* dan *local explanation*. *Global explanation* bertujuan menjelaskan perilaku model secara keseluruhan sehingga pengguna dapat memahami cara kerja model terhadap seluruh data. Sementara itu, *local explanation* berfokus pada penjelasan terhadap satu hasil prediksi tertentu dengan menunjukkan kontribusi masing-masing fitur yang memengaruhi keputusan model pada data yang sedang dianalisis. Selain itu, metode XAI juga dapat dibedakan menjadi *model-specific*, yaitu metode yang dirancang khusus untuk algoritma tertentu, serta *model-agnostic*, yaitu metode yang dapat diterapkan pada berbagai jenis model *machine learning* tanpa bergantung pada struktur algoritma yang digunakan.

Meskipun memberikan berbagai manfaat, penerapan *Explainable Artificial Intelligence* masih menghadapi beberapa tantangan. Salah satu tantangan utama adalah adanya kompromi antara tingkat akurasi model dan tingkat interpretabilitasnya. Model yang memiliki performa prediksi tinggi umumnya memiliki struktur yang lebih kompleks sehingga lebih sulit dijelaskan dibandingkan model yang lebih sederhana. Selain itu, hingga saat ini belum terdapat standar yang baku untuk mengevaluasi kualitas penjelasan yang dihasilkan oleh metode XAI. Oleh karena itu, pemilihan metode *Explainable Artificial Intelligence* harus disesuaikan dengan karakteristik model, kebutuhan pengguna, serta tujuan penggunaan sistem agar penjelasan yang diberikan tetap akurat, mudah dipahami, dan memberikan manfaat bagi pengguna (Minh et al., 2022).

2.2.6 SHapley Additive exPlanations (SHAP)

SHapley Additive exPlanations (SHAP) merupakan salah satu metode *Explainable Artificial Intelligence* (XAI) yang digunakan untuk menjelaskan hasil prediksi model *machine learning*. SHAP dikembangkan sebagai suatu kerangka kerja (*unified framework*) yang mampu memberikan penjelasan terhadap hasil prediksi berbagai jenis model secara konsisten. Menurut Lundberg dan Lee (2017), SHAP memberikan nilai kontribusi pada setiap fitur yang digunakan oleh model sehingga dapat diketahui seberapa besar pengaruh masing-masing fitur terhadap hasil prediksi. Dengan demikian, pengguna tidak hanya memperoleh hasil klasifikasi, tetapi juga dapat memahami alasan yang mendasari keputusan model.

Konsep SHAP didasarkan pada *Shapley Value* yang berasal dari *cooperative game theory*. Dalam teori tersebut, setiap pemain (*player*) dianggap memberikan kontribusi terhadap hasil akhir yang diperoleh secara bersama-sama. Konsep ini kemudian diadaptasi ke dalam *machine learning*, di mana setiap fitur diperlakukan sebagai pemain yang berkontribusi dalam menghasilkan suatu prediksi. Nilai *Shapley* menunjukkan besarnya kontribusi rata-rata setiap fitur terhadap hasil prediksi dengan mempertimbangkan seluruh kemungkinan kombinasi fitur lainnya. Oleh karena itu, setiap fitur memperoleh nilai kontribusi yang mencerminkan pengaruhnya terhadap keputusan model secara adil dan konsisten (Lundberg & Lee, 2017).

Dalam proses interpretasi, SHAP menghasilkan nilai yang dikenal sebagai *SHAP value*. Nilai tersebut menunjukkan arah dan besarnya kontribusi suatu fitur terhadap hasil prediksi model. Nilai SHAP yang bernilai positif menunjukkan bahwa suatu fitur memberikan kontribusi yang mendorong hasil prediksi menuju kelas tertentu, sedangkan nilai negatif menunjukkan bahwa fitur tersebut mengurangi kecenderungan model terhadap kelas tersebut. Semakin besar nilai absolut dari *SHAP value*, maka semakin besar pula pengaruh fitur tersebut terhadap keputusan yang dihasilkan oleh model (Lundberg & Lee, 2017).

Lundberg dan Lee (2017) menjelaskan bahwa SHAP memiliki beberapa sifat penting, yaitu *local accuracy*, *consistency*, dan *missingness*. Sifat *local accuracy* memastikan bahwa penjelasan yang diberikan sesuai dengan keluaran model untuk suatu prediksi tertentu. Sifat *consistency* menjamin bahwa peningkatan kontribusi suatu fitur terhadap model akan diikuti dengan peningkatan nilai penjelasan fitur tersebut. Sementara itu, sifat *missingness* menyatakan bahwa fitur yang tidak memberikan kontribusi terhadap hasil prediksi akan memiliki nilai SHAP sebesar nol. Ketiga sifat tersebut menjadikan SHAP sebagai metode interpretasi yang konsisten dan dapat dipercaya dalam menjelaskan keputusan model *machine learning*.

Meskipun SHAP dapat diterapkan pada berbagai jenis model *machine learning*, proses perhitungan nilai *Shapley* secara langsung memerlukan evaluasi terhadap seluruh kemungkinan kombinasi fitur sehingga membutuhkan waktu komputasi yang sangat besar, terutama ketika jumlah fitur semakin banyak. Untuk mengatasi permasalahan tersebut, (Nair et al., 2020) mengembangkan *TreeSHAP*, yaitu algoritma khusus yang dirancang untuk menghitung nilai SHAP secara efisien pada model berbasis pohon keputusan (*tree-based model*) seperti *Decision Tree*, *Random Forest*, *Gradient Boosting*, dan XGBoost. *TreeSHAP* mampu menghasilkan nilai SHAP yang sama dengan perhitungan *Shapley Value* secara teoritis, namun dengan waktu komputasi yang jauh lebih cepat sehingga lebih sesuai digunakan pada model yang memiliki struktur pohon yang kompleks.

Berdasarkan karakteristik tersebut, penelitian ini memilih menggunakan *TreeSHAP* karena model klasifikasi yang dibangun menggunakan algoritma XGBoost,

yang termasuk ke dalam kelompok *tree-based model*. Penggunaan *TreeSHAP* memungkinkan proses perhitungan nilai SHAP dilakukan secara lebih efisien dibandingkan perhitungan *Shapley Value* secara langsung, tanpa mengurangi ketepatan hasil interpretasi. Dengan demikian, sistem mampu menghasilkan penjelasan terhadap setiap hasil prediksi secara cepat sehingga sesuai untuk diterapkan pada sistem deteksi URL *phishing* yang bekerja secara *real-time*.

Pada penelitian ini, metode SHAP diimplementasikan menggunakan algoritma *TreeSHAP* untuk menjelaskan hasil prediksi model XGBoost dalam mendeteksi URL *phishing*. Melalui nilai SHAP yang dihasilkan, sistem tidak hanya memberikan hasil klasifikasi berupa *phishing* atau *legitimate*, tetapi juga menampilkan kontribusi masing-masing fitur URL terhadap keputusan model. Dengan demikian, pengguna dapat memahami faktor-faktor yang memengaruhi hasil prediksi sehingga sistem menjadi lebih transparan, mudah diinterpretasikan, dan mampu meningkatkan kepercayaan pengguna terhadap hasil yang diberikan.