PERBANDINGAN AKURASI ALGORITMA XGBOOST DAN SVR DALAM PREDIKSI HARGA CRYPTOCURRENCY

Nazwa Fadhil [1)]

[1)] Teknik Informatika, FTI Universitas Tarumanagara
Letjen S. Parman St No.1, Tomang, Grogol petamburan,
West Jakarta City, Jakarta 11440
email : najwagayo@gmail.com

ABSTRACT

The purpose of this study is to compare the effectiveness of two machine learning algorithms, XGBoost and Support Vector Regression (SVR), in predicting cryptocurrency prices to address the challenges posed by market volatility. This study evaluates the performance of both algorithms through various metrics including mean absolute error (MAE), root mean squared error (RMSE), mean squared error (MSE), and mean absolute percentage error (MAPE) using transaction data of 10 cryptocurrencies. The results show that XGBoost significantly outperforms SVR, achieving consistently low MAPE values across all cryptocurrencies, demonstrating its ability to effectively capture market price movements. In contrast, SVR showed mixed performance, succeeding with certain cryptocurrencies but struggling with others, highlighting their inconsistency in predicting market trends. This study concludes that XGBoost is a more effective algorithm in predicting cryptocurrency prices and demonstrates its potential to improve financial forecasting in the cryptocurrency sector.

Key words

cryptocurrency, machine learning, XGBoost, support vector regression, price prediction.ss

1. Pendahuluan

Perkembangan teknologi telah membawa perubahan yang signifikan bagi peradaban umat manusia terutama dalam bidang keuangan, industri keuangan telah berkembang pesat sejak munculnya teknologi – teknologi baru yang membantu kemajuan industri itu sendiri, Salah satunya *Cryptocurrency*. *Cryptocurrency* mengacu pada mata uang digital atau virtual yang menggunakan kriptografi untuk keamanan, yang beroperasi secara independen dari otoritas pusat atau pemerintah. Mata uang ini dirancang untuk bekerja sebagai alat tukar, memungkinkan transaksi yang aman dan terverifikasi pada jaringan terdesentralisasi [1], selain sebagai alat tukar, koin kripto atau *Cryptocurrency* juga memiliki inovasi dan project yang di jalankan, perdagangan koin ini merupakan salah satu cara untuk menghimpun dana untuk menjalankan projek tersebut, seperti contoh Bitcoin memperkenalkan *blockchain* sebagai sistem uang elektronik desentralisasi pertama, sementara Ethereum mengembangkan konsep *smart contracts* untuk aplikasi terdesentralisasi. *Stablecoin* seperti Tether (USDT) dan USD Coin (USDC) menawarkan stabilitas harga dengan mengikat nilai mereka pada dolar AS. Binance Coin dan Solana mendukung ekosistem bursa dan dApps dengan fokus pada kecepatan, skalabilitas, dan biaya rendah.

Gambar 1.1 Volume Perdagangan Crypto menurut CoinWire

Perdagangan mata uang kripto telah mengalami pertumbuhan yang luar biasa, dengan volume perdagangan global diproyeksikan melampaui $108 triliun pada tahun 2024, menandai peningkatan yang signifikan dari tahun-tahun sebelumnya [2]. Kutipan tersebut menjelaskan betapa masifnya volume perdagang dari *Cryptocurrency*, dikarenakan masifnya perkembangan transaksi di *Cryptocurrency* mendorong beberapa individu untuk mulai berinvestasi di mata uang digital ini, meningkatnya minat investasi di mata uang ini membuat harga *Cryptocurrency* bergerak sangat *volatile*, yang membuat sulitnya untuk menganalisis secara teknikal, dan sulit untuk memprediksi pergerakan harganya.

Hadirnya *Machine Learning* menjawab permasalahan tersebut dengan algortima *Machine Learning* memungkinkan investor untuk memprediksi harga dari koin di *Cryptocurrency*, Beberapa tahun ini sudah bermunculan banyak algoritma *Machine Learning* untuk memprediksi harga *Cryptocurrency*, salah satu contohnya adalah *Support Vector Regression* (SVR), dikarenakan banyaknya algoritma, membuat akurasi dari

setiap algoritma nya penting untuk menjadi pertimbangan memilih algoritma untuk digunakan memprediksi harga *Cryptocurrency*.

Beberapa peneliti terdahulu sudah menjelaskan akurasi dari algoritma *Support Vector Regression* (SVR) dalam memprediksi harga *Cryptocurrency*, Prediksi Harga Cryptocurrency Menggunakan Support Vector Regression untuk Website Crypto Oracle [3] merupakan salah satu penelitian yang membahas mengenai prediksi harga Cryptocurrency menggunakan algoritma *Support Vector Regression* (SVR), pada penelitian tersebut menerangkan hasil evalusasi SVR Polinomial menghasilkan rata - rata selisih prediksi 4.15% (MAPE) [3].

Terdapat 1 algortima prediksi *Machine Learning* yang bernama *Extreme Gradient Boosting* (XGBoost) adalah algoritma pembelajaran mesin yang canggih dan sangat efektif yang membangun dan menggabungkan beberapa pohon keputusan untuk meningkatkan kinerja prediksi, dengan memanfaatkan teknik optimasi gradien dan regulasi [4]. *Extreme Gradient Boosting* (XGBoost) unggul dalam kinerja prediksi dibandingkan algoritma *Machine Learning* lainnya karena metode boosting yang efisien dan teknik regulasi yang menyeluruh [5], menurut jurnal tersebut algoritma *XGBoost* unggul dari algoritma *Machine Learning* lainya.

Penelitian ini bertujuan untuk membandingkan akurasi dari algortima *Machine Learning Support Vector Regression* (SVR) dengan *Extreme Gradient Boosting* (XGBoost) untuk memprediksi harga *Cryptocurrency*, penelitian ini akan menggunakan 10 data transaksi koin *Cryptocurrency* untuk bahan pembanding dan akan menggunakan *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), *Mean Squared Error* (MSE) dan *Mean Absolute Percentage Error* (MAPE) untuk mengevaluasi akurasi dari kedua algoritma tersebut.

2. Metode Penelitian

2.1 Extreme Gradient Boosting (XGBoost)

XGBoost, singkatan dari *Extreme gradient boosting (XGBoost)* Yang diusulkan oleh Chen & Guestrin pada tahun 2016, adalah salah satu algoritma boosting yang telah terbukti unggul dalam berbagai kompetisi *data science*. XGBoost bekerja dengan menggabungkan beberapa model keputusan sederhana (*weak learners*) untuk membentuk model yang lebih kuat. Proses *boosting* secara iteratif meningkatkan akurasi dengan mengurangi kesalahan prediksi dari model sebelumnya. XGBoost digunakan untuk masalah *supervised learning* dimana menggunakan data latih dengan beberapa fitur $\mathcal{X}i$ untuk memprediksi variabel target $\mathcal{Y}i$. berikut penjelasan lebih detail mengenai perhitungan dari XGBoost, tahapan pertama pada XGBoost adalah inisialisasi dimana ditahap ini nilai rata-rata dari seluruh data dijadikan prediksi awal untuk seluruh data.

$$F0 = \frac{1}{N} i = \sum_{i=1}^{N} y \qquad (2.1)$$

Keterangan :

$F0$: Nilai rata-rata target $\bar{y}$

yi : nilai target dari data pelatihan

N : jumlah data

Pada tahap inisialisasi dalam algoritma XGBoost, model dimulai dengan menetapkan prediksi awal untuk semua data sebagai nilai rata-rata dari target pada data pelatihan. Ini dilakukan dengan menghitung rata-rata dari nilai target y di seluruh data pelatihan dan menggunakan nilai tersebut sebagai prediksi awal $F0$. Misalnya, jika data pelatihan adalah harga penutupan *cryptocurrency*, $F0$ adalah harga penutupan rata-rata dari data pelatihan tersebut. Inisialisasi ini memberikan titik awal yang sederhana untuk model sebelum memulai proses iteratif pembelajaran. Prediksi awal ini berfungsi sebagai dasar untuk perbaikan lebih lanjut yang akan dilakukan oleh pohon keputusan dalam iterasi berikutnya, dengan tujuan untuk mengurangi kesalahan prediksi secara bertahap melalui boosting [14]. Setelah menentukan inisialisasi dari data prediksi awal akan dilanjutkan dengan tahapan menghitung *gradient* dan *hessian*, *gradient* adalah turunan pertama dari fungsi loss terhadap prediksi. *Gradient* menunjukkan seberapa besar kesalahan pada prediksi saat ini dan dalam arah mana model perlu diperbaiki.

$$gi = \frac{\partial l(yi, \hat{y}i)}{\partial \hat{y}i} = 2(\hat{y}i - yi) \qquad (2.2)$$

Keterangan :

yi : Nilai aktual dari data.

$\hat{y}i$: adalah prediksi pada iterasi ke-i

$l(yi, \hat{y}i)$: Fungsi loss

gi : *Gradient* yang menunjukkan seberapa besar dan arah kesalahan yang perlu dikoreksi.

Tahapan selanjutnya adalah menghitung nilai *hessian*, *hessian* Merupakan turunan kedua dari fungsi loss terhadap prediksi. Hessian digunakan untuk mengukur kelengkungan fungsi loss, yang membantu dalam menentukan seberapa besar langkah perbaikan yang harus diambil.

$$hi = \frac{\partial^2 l(yi, \hat{y}i)}{\partial \hat{y}i^2} = 2 \qquad (2.3)$$

Keterangan :

yi : Nilai aktual dari data.

$\hat{y}i$: adalah prediksi pada iterasi ke-i

$l(yi, \hat{y}i)$: Fungsi loss

hi : *Hessian* yang menunjukkan perubahan gradient, memberi informasi tentang kecepatan perubahan kesalahan.

Langkah selanjutnya adalah memilih *split* terbaik berdasarkan nilai *hessian* dan *gradient*, langkah ini digunakan untuk menghitung pengurangan *loss* (atau peningkatan kualitas) setelah data dibagi menjadi dua bagian di sebuah *node* pohon (split).

$$L_{Split} = \frac{1}{2}\left(\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right) - \gamma \qquad (2.4)$$

Keterangan :

I_L dan I_R : Set data yang dibagi ke kiri (*left*) dan kanan (*right*) setelah *split*.

gi: *Gradient* dari data yang berada di node tersebut.

hi: *Hessian* dari data yang berada di node tersebut.

λ : Parameter regularisasi untuk mengontrol kompleksitas model.

γ : Parameter yang menentukan pengurangan loss minimum yang dibutuhkan untuk melakukan split.

Pada langkah ini Algoritma akan mencoba berbagai *split* (pemisahan) berdasarkan fitur yang tersedia pada data Untuk setiap split yang mungkin, tujuan dari langkah ini adalah mencari pembagian data yang menghasilkan set data yang lebih *homogen* (mendekati nilai target) sehingga meningkatkan akurasi prediksi. Selanjutnya algoritma XGBoost akan melakukan regularisasi, regularisasi digunakan untuk menghindari *overfitting* dengan menambahkan penalti terhadap kompleksitas model. Dalam konteks XGBoost, kompleksitas model diukur berdasarkan jumlah *leaf* dalam pohon dan besaran nilai di *leaf* tersebut.

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \qquad (2.5)$$

Keterangan :

T: Jumlah *leaf* (terminal nodes) dalam pohon.

w_j: Nilai atau bobot pada leaf ke-j.

hi: *Hessian* dari data yang berada di node tersebut.

λ : Parameter regularisasi untuk penalti besar kecilnya bobot di setiap leaf.

γ : Parameter regularisasi untuk penalti jumlah *leaf*.

Setelah membangun pohon baru, model XGBoost akan menghitung penalti atas kompleksitas pohon tersebut, tujuan dari langkah ini adalah untuk memastikan model tetap sederhana dan tidak overfit terhadap data latih. Dan tahap yang terakhir adalah setelah split terbaik ditemukan dan pohon dibangun, prediksi akhir diperbarui dengan menambahkan hasil prediksi dari pohon terbaru, yang dikalikan dengan learning rate η.

$$\hat{y}_i = \hat{y}_i + \eta\, f(x_i) \qquad (2.6)$$

Keterangan :

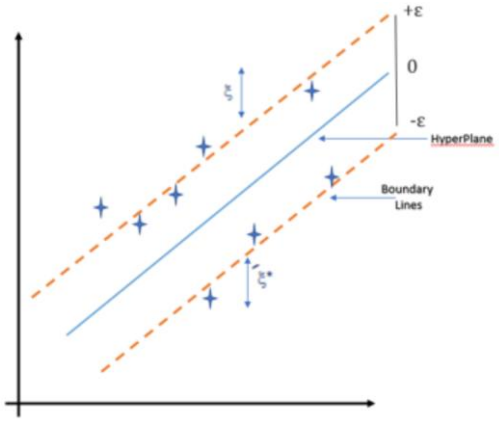
$f(x_i)$: Prediksi dari pohon baru berdasarkan fitur x_i

η: learning rate, yang mengontrol seberapa besar pengaruh pohon baru terhadap model keseluruhan.

Prediksi untuk setiap data diperbarui secara bertahap, setiap pohon menambahkan koreksi kecil pada prediksi sebelumnya, dengan tujuan untuk terus mengurangi error Learning rate (η) digunakan untuk mengontrol kecepatan pembelajaran, nilai yang lebih kecil membuat pembelajaran lebih lambat tapi lebih stabil, sedangkan nilai yang lebih besar membuat pembelajaran lebih cepat tetapi bisa membuat model lebih mudah overfitting.

2.2 Support Vector Regression (SVR)

Support Vector Regression (SVR) adalah algoritma *supervised learning* yang digunakan untuk memprediksi nilai variabel kontinu. SVR menggunakan prinsip yang sama dengan SVM, tujuan dasar dari algoritma SVR adalah menemukan garis keputusan yang paling sesuai. Dalam SVR, garis yang paling cocok adalah *hyperplane* yang memiliki jumlah poin maksimum, SVR bekerja untuk mencari nilai terbaik dalam margin tertentu yang disebut epsilon [15].



Gambar 2.1 Grafik SVR

Pada gambar 2.1 *Hyperlane* adalah garis pemisah antara dua kelas data dalam dimensi yang lebih tinggi dari dimensi sebenarnya. Dalam SVR, *hyperplane* didefinisikan sebagai garis yang membantu dalam memprediksi nilai target (kontinu). Titik-titik data di kedua sisi *hyperplane* yang paling dekat dengan *hyperplane* disebut *Support Vector*. Selanjutnya *kernel* adalah kumpulan fungsi matematika yang mengambil data sebagai input dan mengubahnya menjadi bentuk yang diperlukan, kernal umumnya digunakan untuk menemukan *hyperplane* di ruang dimensi yang lebih tinggi sedangkan *Boundary line* atau garis batas adalah dua garis yang ditarik di sekitar *hyperplane* pada jarak tertentu (epsilon). Tahapan mendetail mengenai algoritma SVR adalah sebagai berikut.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (2.7)$$

Keterangan :

x: nilai fitur dari data set.

$\min(x)$: Nilai minimun dari fitur.

$\max(x)$: Nilai Maksimum dari fitur.

x' : Nilai fitur setelah normalisasi.

Normalisasi data adalah tahap untuk memastikan bahwa semua fitur memiliki skala yang sama untuk menghindari bias dalam model dan mempercepat proses pelatihan, tahap selanjutnya adalah membangun Fungsi SVR.

$$f(x) = w^T . x + b \quad (2.8)$$

Keterangan :

w: Vektor bobot.

x: vektor fitur (input).

b: Nilai bias.

Selanjutnya kita harus menentukan jenis kernel yang digunakan, misalnya kernel RBF dan juga menentukan parameter yang dibutuhkan, seperti

C: Parameter Regularisasi.

ϵ: nilai epsilon.

γ: Parameter untuk kernel RBF.

Setiap parameter tersebut memiliki fungsi masing masing untuk parameter regularisasi mengontrol trade-off antara kesalahan pada training data dan kompleksitas model, sedangkan epsilon mengatur lebar margin bebas kesalahan di sekitar *hyperplane*, dan γ pada kernel RBF mengontrol bentuk dari fungsi kernel *Radial Basis Function* (RBF). Ini menentukan seberapa jauh pengaruh dari satu titik data meluas, dengan kata lain, seberapa "lonjong" atau "membulat" bentuk fungsi kernel tersebut.

$$K(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right) \quad (2.9)$$

Keterangan :

$K(xi, xj)$: nilai kernel antara dua titik data x_i dan x_j.

(xi, xj) : vektor fitur dari dua data yang berbeda.

$\left\| x_i - x_j \right\|^2$: jarak Euclidean kuadrat x_i dan x_j.

γ : parameter yang mengontrol jangkauan pengaruh dari fungsi kernel.

Setelah fungsi kernel ditentukan tahapan selanjutnya adalah membangun model SVR dengan tahapan pertama yaitu menyusuh fungsi objektif sebagai berikut.

$$min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} [\xi_i + \xi_i^*] \quad (2.10)$$

Dengan syarat :

$$y_i - (\mathrm{w}^{\mathrm{T}} x_i + b) \leq \epsilon + \xi_i$$

$$(\mathrm{w}^{\mathrm{T}} x_i + b) - y_i \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

Selanjutnya perhitungan akan dilanjutkan untuk mengoptimalkan Fungsi objektif dan dilanjut dengan memprediksi nilai baru setelah model terbentuk.

$$f(x) = \sum_{i=1}^{n} (a_i - a_i^*) K(x_i, x) + b \quad (2.11)$$

Keterangan :

$f(x)$: Nilai prediksi SVR.

a_i : Pengali Lagrange untuk titik data i yang terkait dengan margin atas dari *hyperplane*.

a_i^* : Pengali Lagrange untuk titik data i yang terkait dengan margin bawah dari *hyperplane*.

$K(x_i, x)$: Fungsi *kernel* yang menghitung kesamaan antara titik data *training* x_i dan input baru x.

b : Nilai bias.

2.3 Metrik Evaluasi

Seluruh hasil prediksi data menggunakan setiap algoritma selesai, selanjutnya akan dilakukan evaluasi menggunakan beberapa metrik evaluasi, metrik evaluasi yang digunakan dalam penelitian ini adalah *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), *Mean Squared Error* (MSE) dan *Mean*

Absolute Percentage Error (MAPE) untuk menilai tingkat akurasi dari setiap algoritma.

3. Hasil Percobaan

3.1 Hasil XGBoost

Berdasarkan hasil percobaan menggunakan algoritma XGBoost dan dilakukan ke 10 macam mata uang *cryptocurrency* yang berbeda, dan setelah diambil data matrik evaluasinya di dapatkan hasil sebagai berikut.

Tabel 1 Hasil Evaluasi Algoritma XGBoost

Mata Uang	MAE	MSE	RMSE	MAPE
BNB	0.0908	0.0710	0.2665	0.3201
Bitcoin	0.5329	0.0120	0.1097	0.1619
Cardano	0.0153	0.0005	0.0240	0.0697
Dogecoin	0.0012	2.9284	0.0017	1.1600
Ethereum	0.0286	0.0026	0.5126	0.0967
Solana	0.0243	0.0019	0.0442	0.0903
Tether	0.0100	0.0005	0.2387	0.0469
Toncoin	0.2095	0.1073	0.3276	0.8525
USDC	0.0143	0.0010	0.0317	0.0657
XRP	0.0211	0.0012	0.0348	0.0983

Berdasarkan nilai MAPE diatas, kinerja dari algoritma *XGBoost* sangat baik, dengan tingkat akurasi prediksi yang sangat baik. Jumlah akurasi ini menunjukan bahwa *XGBoost* dapat membaca pola pergerakan harga pasar cryptocurrency dengan baik.

3.2 Hasil SVR

Berdasarkan hasil percobaan menggunakan algoritma SVR dan dilakukan ke 10 macam mata uang *cryptocurrency* yang berbeda, dan setelah diambil data matrik evaluasinya di dapatkan hasil sebagai berikut.

Tabel 2 Hasil Evaluasi Algoritma SVR

Mata Uang	MAE	MSE	RMSE	MAPE
BNB	0.0274	0.0013	0.0362	7.4198
Bitcoin	0.0283	0.0015	0.0397	3.6862
Cardano	0.0098	0.0001	0.0132	1.5114
Dogecoin	0.0344	0.0022	0.0475	38.70
Ethereum	0.0328	0.0019	0.0442	5.601
Solana	0.0222	0.0008	0.0292	3.764
Tether	0.1500	0.0394	0.1986	14.47
Toncoin	0.0258	0.0011	0.03332	6.898
USDC	0.1040	0.0200	0.1416	9.818
XRP	0.0079	9.448	0.0097	2.28

Berdasarkan nilai MAPE diatas, kinerja dari algoritma *SVR* cukup baik di beberapa mata uang, namun di beberapa lainya terlihat kinerjanya kurang baik, hal ini menunjukan bahwa SVR belum terlalu baik dalam membaca pergerakan pasar *cryptocurrency*.

4. Kesimpulan

Berdasarkan hasil percobaan yang dilakukan dengan 10 macam mata uang kripto menggunakan 2 metode yang berbeda yaitu XGBoost dan SVR berdasarkan hasil evaliuasi keduanya, maka dapat disimpulkan :

1. Algoritma *XGBoost* lebih memiliki nilai MAPE yang lebih rendah dari pada *SVR*, hal ini menunjukan bahwa akurasi dari algoritma *XGBoost* lebih baik dari *SVR* dengan rentang nilai MAPE diantara 0 – 1 %.
2. Kesimpulanya, algoritma yang paling tepat untuk data *cryptocurrency* adalah *XGBoost.*

REFERENSI

[1] S. J. H. S. a. D. R. E. Bouri, "Cryptocurrency: A digital payment system,," *J. Financ. Econom,* vol. 18, pp. 181-200, 2019.

[2] S. Fortis, "Crypto trading volume to exceed $108T in 2024, with Europe in the lead," 12 July 2024. [Online]. Available: https://cointelegraph.com/news/crypto-trading-volume-2024-europe-leads.

[3] T. Stephen, "Prediksi Harga Cryptocurrency Menggunakan Support Vector Regression untuk Website Crypto Oracle," Universitas Tarumanagara, Jakarta, 2024.

[4] T. L. Y. L. a. H. Z. J. Chen, "XGBoost: An Optimized Gradient Boosting Framework for Machine Learning Tasks," *Journal of Machine Learning Research, vol. 20,* vol. 20, pp. 1-31, 2019.

[5] L. Z. W. Zhang, "Comparative Analysis of XGBoost and Other Machine Learning Algorithms for Predictive Modeling," *Journal of Computational Science,* vol. 40, pp. 101 - 111, 2020.

[6] A. S. A. A Sholahuddin, "Extreme gradient boosting (XGBoost) method in making forecasting application and analysis of USD exchange rates against rupiah," *Journal of Physics: Conference Series,* p. 1, 2021.

[7] A. R. H. D. M. U. A. S. N. Y. Deny Haryadi, "Implementation of Support Vector Regression for Polkadot Cryptocurrency Price Prediction," *JOIV: International Journal on Informatics Visualization,* vol. 6, pp. 1-2, 2022.

[8] S. Nakamoto, "itcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf. [Accessed 29 08 2024].

[9] A. T. M. I. R. S. R. S. M. Sharif, "A Comprehensive Survey on Cryptocurrency: A New Paradigm of Digital Currency," in *2020 IEEE International Conference on Computing, Power and Communication Technologies (GUCON)*, Greater Noida, India, 2020.

[10] K. C. O'Mahony, "he Impact of Volatility on Cryptocurrency Prices: A Study on Bitcoin, Ethereum, and Ripple," *International Journal of Financial Studies,* vol. 8, no. 2, pp. 22-35, 2020.

[11] Y. S. M. Y. Siswo Adiguno, "Prediksi Peningkatan Omset Penjualan Menggunakan Metode Regresi," *JURNAL SISTEM INFORMASI TGD,* vol. 1, no. 4, p. 1, 2022.

[12] J. Rocca, "Ensemble methods: bagging, boosting and stacking," medium, 23 April 2019. [Online]. Available: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205. [Accessed 31 August 2024].

[13] E. A. Daoud, "Comparison between XGBoost, LightGBM and CatBoost Using a Home Credit Dataset," *World Academy of Science, Engineering and Technology,* vol. 13, p. 1, 2019.

[14] C. G. Tianqi Chen, "XGBoost: A Scalable Tree Boosting System," 2016.

[15] Trivusi, "Algoritma Support Vector Regression (SVR): Jenis SVM untuk Regresi," Trivusi, 17 September 2022. [Online]. Available: https://www.trivusi.web.id/2022/08/algoritma-svr.html.

[16] Trivusi, "Perbedaan MAE, MSE, RMSE, dan MAPE pada Data Science," Trivusi, 11 Maret 2023. [Online]. Available: https://www.trivusi.web.id/2023/03/perbedaan-mae-mse-rmse-dan-mape.html.

Nazwa Fadhil, saat ini sebagai mahasiswa program studi Teknik Informatika Universitas Tarumanagara angkatan 2021.

Bagus Mulyawan, memperoleh gelar S.Kom. dari Universitas Gunadarma. Kemudian memperoleh gelar MM. dari Universitas Budi Luhur. Saat ini aktif sebagai dosen tetap Program Studi Teknik Informatika Fakultas Teknologi Informasi Universitas Tarumanagara.

Manatap Dolok Lauro, Dosen Program Studi Teknologi Informatika Fakultas Teknologi Informasi Universitas Tarumanagara, Jakarta2021. Saat ini sebagai staf Pengajar Program Studi Sistem Informasi Universitas Tarumanagara.