Jurnal Komputer dan Informatika
Vol 19 No 2, Oktober 2024: hlm 111- 118

ISSN 1410-7228 (Versi Cetak)
ISSN 2716-4233 (Versi

EVALUASI EFEKTIVITAS ALGORITMA KLASIFIKASI BEBAN PENGGUNAAN LISTRIK PADA MESIN PABRIK BAJA

Michael Chan
Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Tarumanagara,
Jln. Letjen S. Parman No. 1, Jakarta, 11440, Indonesia
E-mail: michael.535220079@stu.untar.ac.id

ABSTRAK

Beban penggunaan listrik pada suatu pabrik sangatlah penting untuk merencanakan berjalannya proses produksi. Dengan mengetahui jenis beban pada pabrik, stabilitas penyediaan daya listrik pada mesin dan alat penting dapat dipastikan, penggunaan mesin secara berlebihan dapat dicegah, dan biaya dapat dihemat dengan menyesuaikan jadwal produksi tinggi dengan jam listrik di luar jam sibuk. Penelitian ini bertujuan untuk menggunakan algoritma *K-Nearest Neighbors* (KNN), *Decision Tree*, dan *Support Vector Machine* (SVM) untuk mengklasifikasi jenis beban pada suatu pabrik baja. Hasil penelitian merupakan persentase akurasi dari setiap algoritma yang masing-masing sebesar 90.2%, 82.8%, dan 92.8% pada pemisahan data 90% data latih dan 10% data uji. Dari hasil tersebut, dapat disimpulkan bahwa algoritma *Decision Tree* merupakan algoritma yang paling baik untuk dataset ini.

Kata kunci—Penggunaan Energi Pabrik, *Decision Tree, KNN, SVM*

ABSTRACT

A factory's load type on electrical usage is important in asserting how one has to manage its production. By identifying a factory's load type, one could better ensure the stable energy for core machines and appliances, prevent machine overload, as well as saving costs by aligning high production schedules with off-peak electricity hours. This research aimed to use K-Nearest Neighbors (KNN), Decision Tree, and Support Vector Machine (SVM) algorithms to classify load types in a steel industry factory. The research concluded that the Decision Tree algorithm is by far the most consistently accurate classifier at an accuracy of 92% in all ratios of training and testing dataset split, reaching 92.8% in a ratio of 90% training set and 10% testing set. While the KNN and SVM algorithms lag behind at an accuracy of 90.2% and 82.8% respectively.

Keywords—*Factory Energy Consumption, Decision Tree, KNN, SVM*

1. PENDAHULUAN

Jenis beban pada pabrik industri, atau biasa disebut dengan *Load Type* adalah besarnya tenaga listrik yang digunakan oleh pabrik tersebut pada suatu waktu [1]. Secara umum, jenis beban ini terbagi menjadi tiga yaitu beban ringan *(Light/Base Load)* yang mengacu pada penggunaan listrik dibawah 50% dari kapasitas total pabrik, beban sedang *(Medium/Intermediate Load)* yang mengacu pada penggunaan listrik di antara 50% hingga 80%, dan juga beban maksimal *(Maximum/Peak Load)* pada penggunaan tenaga listrik sebesar 80% hingga 100% dari kapasitas pabrik [2].

Jenis beban menggambarkan pola penggunaan listrik pada suatu pabrik seiring berjalannya waktu. Pola ini mengalami fluktuasi tergantung pada permintaan *(demand)* pada waktu tertentu. Beban ringan menunjukkan jumlah listrik minimal yang digunakan pabrik untuk tetap menjalankan alat-alat operasional yang penting untuk proses produksi [3]. Dengan mengidentifikasi jenis beban ini, daya listrik pabrik dapat dikelola untuk memastikan mesin inti pada pabrik dapat tetap berjalan tanpa gangguan[4].

Beban sedang menunjukkan perubahan kebutuhan listrik sesuai dengan aktivitas produksi, tenaga kerja, ataupun penggunaan alat tertentu di pabrik [5]. Sedangkan beban maksimal

menunjukkan penggunaan listrik pada saat periode hasil produksi tinggi atau operasi intensif. Dengan mengetahui jenis beban ini, suatu pabrik dapat memastikan untuk menghindari kerusakan pada mesin akibat beban berlebihan dan bahkan menjadwalkan periode produksi tinggi saat tagihan listrik rendah untuk menghemat biaya [6]

Tujuan dari penelitian ini adalah untuk mengklasifikasi jenis beban penggunaan listrik pada suatu pabrik baja menggunakan tiga jenis algoritma yaitu *K-Nearest Neighbors (KNN)*, *Decision Tree*, dan *Support Vector Machine (SVM)* kemudian membandingkan hasil dari ketiga algoritma ini untuk menentukan algoritma yang paling efektif. [7]

2. METODE PENELITIAN

2.1 Dataset

Data yang digunakan merupakan data konsumsi energi pada suatu pabrik industri baja pada tahun 2018. Data ini dikumpulkan dari perusahaan *DAEWOO Steel Co. Ltd* yang berlokasi di Gwangyang, Korea Selatan [8]. Terdapat 35040 data dan 10 fitur seperti yang ditunjukkan pada Tabel 1.

Tabel 1 Dataset

Nama Atribut	Peran	Tipe Data	Deskripsi	Satuan
date	Fitur	*String*	Hari, tanggal, tahun, dan jam saat pengambilan data.	-
Usage_kWh	Fitur	*Floating point*	Total energi yang digunakan pada jangka waktu 15 menit.	kWh
Lagging_Current_Reactive.Power_kVarh	Fitur	*Floating point*	Energi yang terbuang dikarenakan gelombang arus yang tertinggal oleh gelombang tegangan	kVarh
Leading_Current_Reactive_Power_kVarh	Fitur	*Floating point*	Energi yang terbuang dikarenakan gelombang tegangan yang tertinggal oleh gelombang arus	kVarh
CO2 (tCO2)	Fitur	*Floating point*	Kadar CO_2 di udara	ppm
Lagging_Current_Power_Factor	Fitur	*Floating point*	Persentase efektivitas konversi energi terhadap besarnya perbedaan fasa antara gelombang arus yang tertinggal oleh gelombang tegangan	%
Leading_Current_Power_Factor	Fitur	*Floating point*	Persentase efektivitas konversi energi terhadap besarnya perbedaan fasa antara gelombang tegangan yang tertinggal oleh gelombang arus	%
NSM	Fitur	*Integer*	Berapa detik setelah tengah malam (*Number of seconds after midnight).*	sec
WeekStatus	Fitur	Kategori	Hari yang didata adalah hari kerja (0) atau akhir pekan (1).	-
Day_of_week	Fitur	Kategori	Senin, selasa, rabu, kamis, jumat, sabtu, minggu	-
Load_Type	Target	Kategori	Beban ringan *(Light Load)*, beban menengah *(Medium Load)*, beban maksimal *(Maximum Load)*	-

Tidak ada data kosong pada dataset, tetapi atribut *date*, *WeekStatus*, dan *Day_of_week* merupakan data berbentuk non-numerik. Oleh karena itu dataset harus diproses terlebih dahulu. Atribut *WeekStatus* dan *Day_of_Week* dapat dikonversikan menjadi angka dengan melakukan encoding. Atribut *date* memiliki 3 data yaitu tanggal, bulan, tahun, dan jam. Karena dataset sudah memiliki atribut *NSM* (berapa detik setelah tengah malam), maka data jam tidak diperlukan. Pada

ISSN 1410-7228 (Versi Cetak)
ISSN 2716-4233 (Versi

kasus ini, data tahun juga tidak diperlukan karena dataset hanya berisikan data pada tahun 2018. Dengan begini, atribut *date* dapat dipisah menjadi dua atribut baru yang berisikan tanggal dan bulan seperti pada Tabel 2. Setelah pemisahan, dataset sekarang memiliki 11 fitur.

Tabel 2 Hasil Pemisahan Atribut *date* Menjadi *day* dan *month*

Nama Atribut	Peran	Tipe Data	Deskripsi
day	Fitur	*Integer*	Tanggal hari saat data diambil (1-31)
month	Fitur	*Integer*	Tanggal bulan saat data diambil (1-12)

2.2 Algoritma

2.2.1 *K-Nearest Neighbors* (KNN)

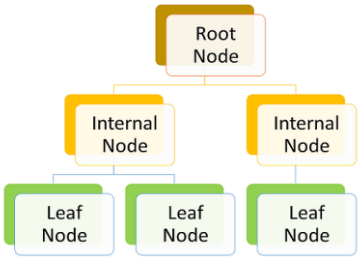
Algoritma ini merupakan sebuah algoritma sederhana yang dapat digunakan untuk klasifikasi dan regresi. Pada algoritma ini, setiap baris pada dataset akan dijadikan vektor yang berisikan semua angka nilai dari fitur [9]. Jarak antara setiap vektor atau titik data akan dihitung menggunakan berbagai jenis rumus, salah satu yang paling sering digunakan adalah *Euclidean Distance* [10][11]. Seluruh vektor kemudian akan dikelompokkan sesuai dengan jumlah tetangga (vektor terdekat dengan vektor tersebut) yang ditentukan oleh peneliti. Setelah model klasifikasi dibuat, ketika data baru diberikan ke algoritma model, maka model tersebut akan mengklasifikasi data tersebut berdasarkan jaraknya dengan jarak data lainnya yang telah dikelompokkan.

$$d(X_i, X_j) = \sqrt{\sum_{k=1}^{n} (X_{ik} - X_{jk})^2} \quad (1)$$

$$d(X_i, X_j) = \text{Jarak antara } X_i \text{ dan } X_j$$
$$n = \text{Jumlah fitur } (11)$$

2.2.2 Decision Tree

Decision Tree merupakan algoritma supervised yang dapat digunakan untuk melakukan klasifikasi dan regresi. Algoritma ini dapat digunakan pada data berbentuk numerik maupun kategorik . Pada dasarnya, algoritma ini memiliki struktur yang serupa dengan sebuah *flowchart* di mana setiap cabang terbentuk oleh sebuah kondisi *if-else*. Kategorisasi data akan dimulai pada *root node*, kemudian akan dikelompokkan lebih lanjut hingga akhirnya sampai di pembagian kelompoknya di *leaf node* [12].



Gambar 1 Struktur *Decision Tree*

Untuk menentukan isi kondisi *if-else* pada tiap *node* pada *Decision Tree*, digunakan algoritma *Information Gain* untuk mendapatkan kondisi yang paling efektif dalam mengkategorisasi data [13].

$$IG(T, a) = H(T) - H(T \mid a) \quad (2)$$

$$IG(T, a) = \text{Information Gain}$$
$$H(T) = \text{Entropy dari Data Training di Parent Node}$$
$$H(T \mid a) = \text{Jumlah dari entropy dari seluruh Child Node}$$
$$T = \text{Data Training}$$
$$a = \text{Atribut pada Data}$$

Algoritma *Information Gain* memerlukan nilai *entropy* pada hasil pemisahan kategori pada data dalam suatu *node*. Nilai *entropy* 0 artinya semua data yang masuk ke dalam *node* tersebut dikategorisasikan ke dalam satu kelompok. Sedangkan nilai 1 artinya data yang masuk merupakan campuran dari data dari kelompok yang berbeda [14].

$$H(T) = -\sum_{i=1}^{j} p_i \cdot \log_2 p_i \tag{3}$$

p_i = Pecahan rasio dari atribut pada suatu internal node

Dengan menggunakan rumus *entropy* di atas, rumus *Information Gain* dapat diubah menjadi bentuk berikut melalui substitusi *H(T)*:

$$IG(T,a) = -\sum_{i=1}^{j} p_i \cdot \log_2 p_i - \left(\sum_{v \in values(a)} \frac{||T_v||}{||T||} H(T_v) \right) \tag{4}$$

2.2.3 *Support Vector Machine* (SVM)

Support Vector Machine adalah suatu algoritma *machine learning* yang mengkategorikan data dengan mencari garis pemisah pada dimensi atau basis dalam ruang fitur *(hyperplane)* [15]. Pada data linear, algoritma ini memaksimalkan jarak antara garis pemisah dengan titik terdekat dari setiap jenis kelompok data. Algoritma ini dapat digunakan untuk klasifikasi dan regresi [16].

Untuk mengkategorikan data non-linear, algoritma SVM menggunakan kernel khusus yang bertugas untuk melakukan transformasi matriks terhadap dataset input ke dimensi matriks yang lebih tinggi. Hal ini dilakukan karena garis pemisah lebih mudah untuk ditemukan pada suatu data yang bersifat non-linear pada dimensi yang lebih tinggi ini. Beberapa contoh kernel yang biasa digunakan adalah kernel linear, *polynomial*, dan *Radial Basis Function (RBF)* [17].

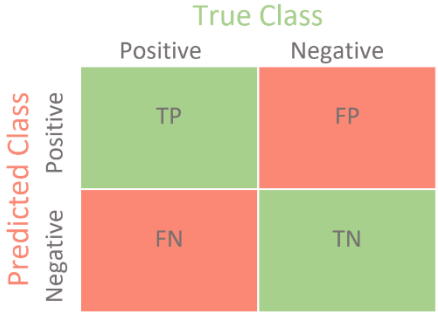
$$K(x,x') = \exp\left(-\frac{||x-x'||^2}{2\sigma^2}\right) \tag{5}$$

$K(x,x')$ = Nilai Kernel RBF dari x dan x'
$||x-x'||$ = Squared Euclidean Distance antara kedua vektor
σ = Parameter bebas

Penelitian ini menggunakan *kernel RBF* untuk melakukan klasifikasi dataset menjadi tiga kelas pada target atribut jenis beban *(Load Type)* yaitu *Light Load*, *Medium Load*, dan *Maximum Load*.

2.2.3 Metode Evaluasi

Penelitian ini dievaluasi menggunakan metrik kinerja *precision*, *recall*, *f1-score*, dan *accuracy* yang didapatkan melalui *confusion matrix* [18].



Gambar 2 *Confusion Matrix*

Pada *confusion matrix*, TP *(True Positive)* adalah prediksi algoritma yang berhasil memprediksi data yang benar. FP *(False Positive)* adalah prediksi data yang salah sebagai data yang benar. FN *(False Negative)* adalah prediksi data yang seharusnya benar sebagai data yang salah, sedangkan TN *(True Negative)* adalah prediksi suatu data salah terhadap data yang memang salah [19] [20].

ISSN 1410-7228 (Versi Cetak)
ISSN 2716-4233 (Versi

$$\text{Accuracy} = \frac{TN + TP}{TN + FP + TP + FN} \tag{6}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{8}$$

$$\text{F1-Score} = 2\left(\frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}\right) \tag{9}$$

3. HASIL DAN PEMBAHASAN

Proses pengujian dilakukan sebanyak 5 kali pada setiap algoritma untuk tiap pembagian dataset. Data dari kelima iterasi pengujian kemudian dirata-ratakan dan secara jelas menunjukkan bahwa pada setiap pembagian data, baik 90/10, 80/20, 70/30, maupun 60/40, KNN dengan hasil prediksi terbaik menggunakan 3 tetangga paling dekat, *Decision Tree* dengan hasil prediksi terbaik memiliki kedalaman maksimum 10, sedangkan SVM kernel RBF dengan hasil prediksi terbaik memiliki nilai C sebesar 500.

Tabel 3 Rata-Rata Seluruh Algoritma Pada Pembagian Data 60/40

Pembagian Data Latih 60% dan Data Uji 40%				
Algoritma	**Nilai Rata-Rata**			
	Precision	**Recall**	**F1-Score**	**Akurasi**
KNN (Nearest Neighbors = 3)	0.886	0.886	0.886	0.886
KNN (Nearest Neighbors = 5)	0.886	0.884	0.886	0.884
KNN (Nearest Neighbors = 7)	0.882	0.882	0.882	0.882
KNN (Nearest Neighbors = 10)	0.880	0.880	0.88	0.880
Decision Tree (Max Depth = 3)	0.780	0.760	0.742	0.760
Decision Tree (Max Depth = 5)	0.836	0.816	0.806	0.816
Decision Tree (Max Depth = 7)	0.900	0.900	0.900	0.900
Decision Tree (Max Depth = 10)	0.980	0.980	0.980	0.980
SVM kernel RBF (C = 100)	0.800	0.782	0.790	0.782
SVM kernel RBF (C = 250)	0.812	0.812	0.812	0.812
SVM kernel RBF (C = 500)	0.830	0.822	0.826	0.822

Tabel 4 Rata-Rata Seluruh Algoritma Pada Pembagian Data 70/30

Pembagian Data Latih 70% dan Data Uji 30%				
Algoritma	**Nilai Rata-Rata**			
	Precision	**Recall**	**F1-Score**	**Akurasi**
KNN (Nearest Neighbors = 3)	0.890	0.890	0.890	0.890
KNN (Nearest Neighbors = 5)	0.888	0.886	0.886	0.886
KNN (Nearest Neighbors = 7)	0.884	0.882	0.882	0.882
KNN (Nearest Neighbors = 10)	0.880	0.880	0.880	0.880
Decision Tree (Max Depth = 3)	0.786	0.752	0.728	0.752
Decision Tree (Max Depth = 5)	0.832	0.814	0.804	0.814

Decision Tree (Max Depth = 7)	0.890	0.890	0.890	0.890
Decision Tree (Max Depth = 10)	0.980	0.980	0.980	0.980
SVM kernel RBF (C = 100)	0.832	0.794	0.800	0.794
SVM kernel RBF (C = 250)	0.806	0.810	0.806	0.810
SVM kernel RBF (C = 500)	0.830	0.822	0.828	0.822

Tabel 5 Rata-Rata Seluruh Algoritma Pada Pembagian Data 80/20

Pembagian Data Latih 80% dan Data Uji 20%				
Algoritma	**Nilai Rata-Rata**			
	Precision	**Recall**	**F1-Score**	**Akurasi**
KNN (Nearest Neighbors = 3)	0.892	0.892	0.892	0.892
KNN (Nearest Neighbors = 5)	0.890	0.890	0.890	0.890
KNN (Nearest Neighbors = 7)	0.886	0.886	0.886	0.886
KNN (Nearest Neighbors = 10)	0.884	0.884	0.884	0.884
Decision Tree (Max Depth = 3)	0.778	0.758	0.742	0.758
Decision Tree (Max Depth = 5)	0.836	0.814	0.808	0.814
Decision Tree (Max Depth = 7)	0.888	0.888	0.888	0.888
Decision Tree (Max Depth = 10)	0.980	0.989	0.980	0.980
SVM kernel RBF (C = 100)	0.814	0.810	0.810	0.810
SVM kernel RBF (C = 250)	0.806	0.806	0.806	0.806
SVM kernel RBF (C = 500)	0.836	0.826	0.830	0.826

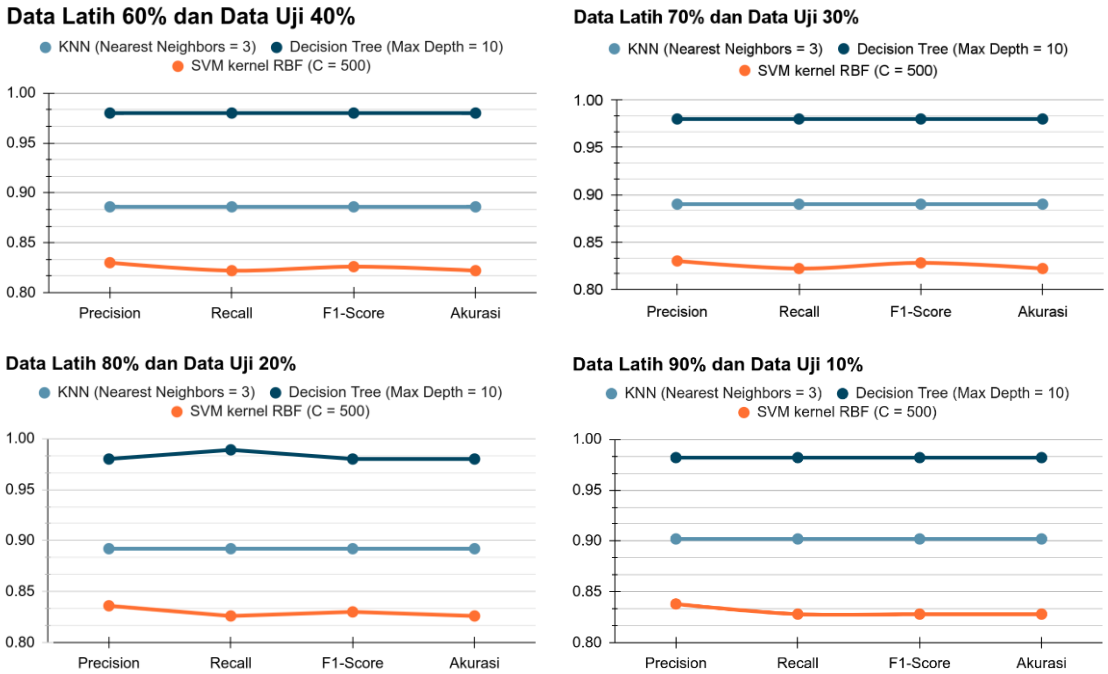
Tabel 6 Rata-Rata Seluruh Algoritma Pada Pembagian Data 90/10

Pembagian Data Latih 90% dan Data Uji 10%				
Algoritma	**Nilai Rata-Rata**			
	Precision	**Recall**	**F1-Score**	**Akurasi**
KNN (Nearest Neighbors = 3)	0.902	0.902	0.902	0.902
KNN (Nearest Neighbors = 5)	0.896	0.896	0.896	0.896
KNN (Nearest Neighbors = 7)	0.894	0.892	0.894	0.892
KNN (Nearest Neighbors = 10)	0.890	0.890	0.890	0.890
Decision Tree (Max Depth = 3)	0.790	0.756	0.732	0.756
Decision Tree (Max Depth = 5)	0.836	0.816	0.808	0.816
Decision Tree (Max Depth = 7)	0.892	0.890	0.890	0.890
Decision Tree (Max Depth = 10)	0.982	0.982	0.982	0.982
SVM kernel RBF (C = 100)	0.810	0.808	0.810	0.808
SVM kernel RBF (C = 250)	0.820	0.818	0.820	0.818
SVM kernel RBF (C = 500)	0.838	0.828	0.828	0.828

Pada rasio data latih 60% dan data uji 40%, algoritma *K-Nearest Neighbors (n = 3)* memiliki nilai rata-rata akurasi 88.6%, *Decision Tree (Max Depth = 10)* memiliki nilai akurasi 98%, sedangkan algoritma *SVM kernel RBF (C = 500)* memiliki nilai rata-rata akurasi 82.2%. Pada rasio data latih 70% dan data uji 30%, algoritma KNN memiliki akurasi 89%, *Decision Tree* dengan akurasi 98%, dan *SVM kernel RBF* dengan akurasi 82.2%. Pada rasio data latih 80% dan data uji 20%, algoritma KNN memiliki akurasi 89.2%, *Decision Tree* dengan akurasi 98%, dan *SVM kernel*

ISSN 1410-7228 (Versi Cetak)
ISSN 2716-4233 (Versi

RBF dengan akurasi 82.6%. Pada rasio data latih 90% dan data uji 10%, algoritma KNN memiliki akurasi 90.2%, *Decision Tree* dengan akurasi 98.2%, dan *SVM kernel RBF* dengan akurasi 82.8%.



Gambar 3 Grafik Evaluasi Algoritma Terbaik Pada Setiap Pembagian Dataset

4. KESIMPULAN

Dari antara algoritma *K-Nearest Neighbors*, *Decision Tree*, dan *SVM Kernel RBF*, algoritma *Decision Tree* dengan kedalaman maksimal 10 secara konsisten memiliki akurasi terbaik yaitu 98% persen pada pembagian data latih 60% dengan data uji 40%, pembagian data latih 70% dengan data uji 30%, dan pembagian data latih 80% dengan data uji 20%, sedangkan pada pembagian data latih 90% dengan data uji 10%, algoritma ini memiliki akurasi yang sedikit lebih baik lagi yaitu 98.2%.

Walaupun secara umum tingkat akurasi semua algoritma meningkat seiring naiknya persentase data latih, akan tetapi peningkatan tersebut tidak begitu signifikan. Meningkatnya jumlah data latih hanya meningkatkan akurasi sebanyak 1% hingga 2% pada algoritma dalam pengujian peneliti.

Pada algoritma *K-Nearest Neighbors*, akurasi tertinggi berada pada algoritma KNN dengan jumlah tetangga 3, kemudian akurasi algoritma menurun semakin ditambahnya jumlah tetangga. Pada algoritma *Decision Tree*, akurasi algoritma semakin tinggi semakin tingginya nilai kedalaman maksimal. Algoritma SVM dengan kernel RBF juga memiliki tren yang serupa dengan algoritma *Decision Tree* karena akurasi algoritma meningkat seiring naiknya nilai *hyperparameter C*.

DAFTAR PUSTAKA

[1] Kaiser, V. 1993. *Industrial Energy Management: Refining Petrochemicals and Gas Processing Techniques*. Editions TECHNIP.

[2] Doty, S., & Turner, W. C. 2012. *Energy Management Handbook, Eighth Edition*. Fairmont Press.

[3] Ahlbrandt, R. S., Fruehan, R. J., & Giarratani, F. 1996. *The renaissance of American steel: Lessons for Managers in Competitive Industries*. Oxford University Press, USA.

[4] Song, B., & Wang, D. 2013. *Reducing Energy Consumption in Manufacturing: Opportunities and Impacts*. In Technology and the Future of Energy (pp. 149–184).

[5] Franke, J., Kreitlein, S., Reinhart, G., Gebbe, C., Steinhilper, R., & Böhner, J. 2017. *Energy Efficiency in Strategy of Sustainable Production III*. In Trans Tech Publications Ltd. eBooks. https://doi.org/10.4028/b-g6znkd

[6] Hannan, M. A., Ker, P. J., Mansor, M., Lipu, H., Al-Shetwi, A. Q., Alghamdi, S., Begum, R. A., & Tiong, S. K. 2023. *Recent advancement of energy internet for emerging energy management technologies: Key features, potential applications, methods and open issues*. *Energy Reports*, *10*, 3970–3992. https://doi.org/10.1016/j.egyr.2023.10.051.

[7] Monaco, R., Liu, X., Murino, T., Cheng, X., & Nielsen, P. S. 2023. *A non-functional requirements-based ontology for supporting the development of industrial energy management systems*. *Journal of Cleaner Production*, *414*, 137614. https://doi.org/10.1016/j.jclepro.2023.137614.

[8] V E, Sathishkumar, Shin, Changsun, and Cho, Yongyun. 2023. *Steel Industry Energy Consumption*. UCI Machine Learning Repository. https://doi.org/10.24432/C52G8C.

[9] Mucherino, A., Papajorgji, P.J., Pardalos, P.M. 2009. *k-Nearest Neighbor Classification. In: Data Mining in Agriculture. Springer Optimization and Its Applications, vol 34*. Springer, New York, NY. https://doi.org/10.1007/978-0-387-88615-2_4.

[10] Stewart, J., Redlin, L., & Watson, S. 2005. *Precalculus: Mathematics for calculus.* Cengage Learning.

[11] Liberti, L., & Lavor, C. (2018). *Euclidean Distance Geometry: An Introduction*. Springer, New York, NY.

[12] Koning, M., & Smith, C. 2017. *Decision trees and random forests: A Visual Introduction for Beginners*. Diterbitkan Secara Independen.

[13] Gray, M., R. 2023. *Entropy and Information Theory First Edition, Corrected*. Springer-Verlag, New York, NY.

[14] Palm, G. 2012. *Conditioning, Mutual Information, and Information Gain. In: Novelty, Information and Surprise*. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-29075-6_11

[15] Cristianini, N., & Shawe-Taylor, J. 2000. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press. Springer, New York, NY.

[16] Wang, L. 2005. *Support Vector Machines: Theory and applications. In Studies in fuzziness and soft computing*. https://doi.org/10.1007/b95439

[17] Aggarwal, C. 2023. *Radial Basis Function Networks. In: Neural Networks and Deep Learning*. Springer, Cham. https://doi.org/10.1007/978-3-031-29642-0_6.

[18] Duvenaud, K., D. 2014. *Automatic Model Construction with Gaussian Processes*. https://www.cs.toronto.edu/~duvenaud/thesis.pdf. Diakses tanggal 20 April 2024.

[19] Ting, K.M. 2011. *Confusion Matrix. In: Sammut, C., Webb, G.I.* (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_157.

[20] Japkowicz, N., & Boukouvalas, Z. 2024. *Machine Learning Evaluation: Towards Reliable and Responsible AI*. Cambridge: Cambridge University Press.