Jurnal Teknik Informatika (JUTIF)
Vol. 5, No. 2, April 2024, pp. 339-347

DOI: https://doi.org/10.52436/1.jutif.2024.5.2.1182
p-ISSN: 2723-3863
e-ISSN: 2723-3871

OPTIMIZATION OF DISEASE PREDICTION ACCURACY THROUGH ARTIFICIAL NEURAL NETWORK (ANN) ALGORITHMS IN DIAGNESE APPLICATION

Ruvita Faurina[1], M. Jumli Gazali[2], Icha Dwi Aprilia Herani[3]

[1,2,3]Informatics, Faculty of Engineering, Universitas Bengkulu, Indonesia
Email: [1]ruvita.faurina@unib.ac.id, [2]ahmadjumli77@gmail.com, [3]ichadwiii05@gmail.com

(Article received: July 03, 2023; Revision: August 26, 2023; published: April 04, 2024)

Abstract

This research aims to enhance the accuracy and speed of diagnoses in the Diagnese application by implementing the ANN algorithm for disease prediction. The dataset used for experimentation was featuring binary data types, containing 131 symptoms used to predict 41 types of diseases. The Diagnese application assists patients in identifying diseases and finding suitable specialist doctors based on reported symptoms. To achieve this goal, researchers explored various machine learning algorithms, such as decision trees, SVM, Random Forest, Logistic Regression, and ANN. Through comprehensive analysis, the ANN algorithm outperforms other algorithms and showcases the best performance. The research results demonstrate that integrating this application can significantly improve diagnostic accuracy and speed, thereby potentially reducing treatment delays and enhancing patient health outcomes. The neural network model displayed exceptional accuracy across training, validation, and testing datasets, scoring 97%, 99%, and 95%, respectively. Overall, this study showcases the potential of implementing the ANN algorithm within Diagnese applications to elevate the accuracy and efficiency of disease diagnosis. The application of this model is expected to augment the efficiency and precision of the medical diagnosis process, enabling doctors to make more accurate decisions and provide more effective patient care.

Keywords: *Artificial Neural Network, Disease diagnosis, Healthcare application, Machine Learning.*

OPTIMISASI AKURASI PREDIKSI PENYAKIT MELALUI ALGORITMA *ARTIFICIAL NEURAL NETWORK* (ANN) PADA APLIKASI DIAGNESE

Abstrak

Penelitian ini bertujuan untuk meningkatkan akurasi dan kecepatan diagnosis pada aplikasi Diagnese dengan menerapkan algoritma ANN untuk memprediksi penyakit. Dataset yang digunakan memiliki tipe data biner yang berisi 131 gejala untuk memprediksi 41 jenis penyakit. Aplikasi Diagnese bertujuan untuk memprediksi penyakit pasien dan menemukan dokter spesialis yang sesuai berdasarkan gejala yang diinputkan. Dalam upaya mencapai tujuan, peneliti telah melakukan eksperimen menggunakan berbagai algoritma *machine learning*, seperti *decision tree, SVM, random forest, logistic reggression*, dan *ANN*. Setelah analisis yang komprehensif, algoritma ANN terbukti unggul dan menampilkan kinerja yang paling baik dibandingkan algoritma lainnya. Hasil penelitian ini menunjukkan bahwa integrasi aplikasi ini memiliki potensi besar dalam meningkatkan akurasi dan kecepatan diagnosis, dengan potensi mengurangi keterlambatan dalam proses pengobatan dan menghasilkan perbaikan dalam hasil kesehatan pasien. Model ANN yang digunakan dalam penelitian ini mencapai akurasi sangat baik dalam dataset pelatihan, validasi, dan pengujian, dengan persentase akurasi masing-masing mencapai 97%, 99%, dan 95%. Secara keseluruhan, penelitian ini membahas potensi penerapan algoritma ANN dalam aplikasi Diagnese untuk meningkatkan akurasi dan efisiensi diagnosis penyakit. Dengan penerapan model ini, diharapkan proses diagnosis medis menjadi lebih efisien dan akurat, memberikan kesempatan kepada dokter untuk membuat keputusan yang lebih tepat, serta memberikan perawatan yang lebih efektif kepada pasien.

Kata kunci: *Aplikasi kesehatan, Artificial Neural Network, Diagnosis penyakit, Machine Learning.*

1. PENDAHULUAN

Dalam era digital yang sedang berlangsung saat ini, aplikasi kesehatan semakin mendapat popularitas dan menjadi solusi bagi pasien yang ingin mengenali penyakit serta menemukan dokter spesialis yang tepat. Meskipun begitu, presisi diagnosis penyakit masih menjadi hambatan yang memerlukan upaya penyelesaian, mengingat adanya beragam faktor yang bisa mempengaruhi hasil diagnosa. Pada tahun 2015, tercatat insiden dimana kesalahan diagnostik utama

ditemukan dalam rentang 10% hingga 20% dari keseluruhan autopsi yang dilakukan, yang mengindikasikan bahwa antara 40.000 hingga 80.000 pasien di Amerika Serikat meninggal setiap tahunnya akibat dari kesalahan diagnostik .

Perkembangan teknologi di era digital kini sangat dipengaruhi oleh konsep kecerdasan buatan, yang membuka potensi besar untuk meningkatkan akurasi dan efisiensi dalam berbagai aplikasi kesehatan, termasuk dalam proses diagnosis medis. Kecerdasan buatan dapat belajar cara agar komputer dapat melakukan tugas seperti yang manusia lakukan, dengan menggunakan pemikiran atau kecerdasan seperti manusia pada perangkat mekanik atau mesin untuk melakukan pekerjaan tertentu [1]. Seiring dengan perkembangan zaman, *machine learning* cabang dari kecerdasan buatan juga mengalami evolusi lebih lanjut dalam bentuk *deep learning*, yaitu cabang dari *machine learning* yang menggunakan metode yang lebih kompleks dan canggih. *Deep learning* memiliki kemampuan untuk mesin mempelajari komputasi dengan menggunakan "otak"nya sendiri [2]. Teknologi *deep learning* ini menjadi salah satu teknologi paling populer dalam pengenalan aktivitas atau objek yang memiliki tingkat akurasi lebih tinggi dibandingkan dengan metode mesin sebelumnya [3].

Berdasarkan data dari *World Health Organization* (WHO) pada tahun 2020, sekitar 80% kematian disebabkan oleh penyakit. Salah satu faktor yang menyebabkan banyaknya kematian akibat penyakit adalah kesulitan individu dalam mendapatkan diagnosis yang akurat dan prognosis yang tepat. Hal ini dapat mengakibatkan keterlambatan dalam pengobatan dan dapat memperburuk kondisi kesehatan mereka. Proses diagnostik juga memakan waktu dan biaya yang tinggi, serta memerlukan banyak kunjungan ke penyedia layanan kesehatan untuk melakukan tes diagnostik [4].

Untuk membuat sistem kesehatan yang lebih skalabel dan efektif biaya, pada dasarnya ilmu komputer dan kedokteran bisa melampaui pencitraan medis tradisional dengan menggabungkan kedua bidang ini dengan analisis dan pengambilan data serta kecerdasan buatan [5]. Keakuratan prediksi penyakit dalam industri kesehatan dan medis sangat penting guna membuat keputusan yang efektif dalam menganalisis dan memprediksi penyakit yang diderita oleh pasien [6].

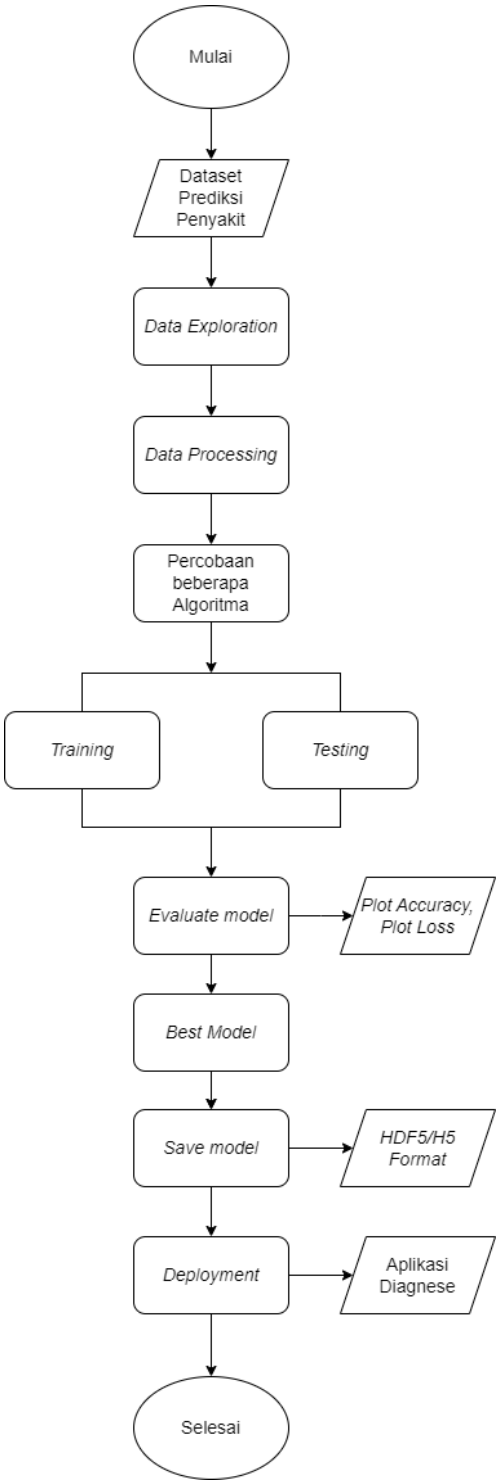
Penelitian sebelumnya memiliki peran yang penting dalam melakukan studi baru. Penelitian sebelumnya dapat membantu menemukan hubungan antara penelitian yang sedang dilakukan dengan penelitian-penelitian terdahulu yang telah ada. Contohnya, pada tahun 2019, dilakukan penelitian tentang "*Parkinson's Disease Prediction Using Artificial Neural Network*" oleh Ramzi M. Sadek, Salah A. Mohammed, Abdul Rahman K. Abunbehan, Abdul Karim H. Abdul Ghattas, Majed R. Badawi, Mohamed N. Mortaja, Bassem S. Abu-Nasser, dan Samy S. Abu-Naser. Penelitian ini menggunakan algoritma *neural network* dengan metode *backpropagation* untuk membantu dokter dalam prediksi penyakit Parkinson [7]. Dalam penelitian ini, data terdiri dari 195 rekaman yang diambil dari 31 orang yang 23 menderita penyakit Parkinson. Kemudian membaginya menjadi 170 sampel pelatihan dan 25 sampel validasi. Model yang dihasilkan dari penelitian ini diuji dan mencapai tingkat akurasi yang sangat tinggi yaitu 100%.. Penelitian lainnya pada tahun 2021 oleh Anas Faisal dan Agus Subekti berjudul "Deep *Neural Network* untuk Prediksi Stroke" juga memberikan kontribusi penting dalam bidang kesehatan. Penelitian ini menggunakan model arsitektur *Deep Neural Network* (DNN) dengan *pre-processing* SMOTE dan kombinasi dengan *Tomek link* [8]. Penelitian ini menghasilkan tingkat akurasi, *f1-score*, dan AUC yang tinggi. Pada tahun 2023, penelitian yang dilakukan oleh Simeon Yuda Prasetyo dengan judul "Prediksi Gagal Jantung Menggunakan *Artificial Neural Network*" juga memberikan kontribusi penting dalam prediksi penyakit. Penelitian ini mencapai tingkat akurasi dan AUC yang tinggi dalam memprediksi gagal jantung [9].

Berdasarkan penelitian-penelitian tersebut, peneliti dan tim mengembangkan aplikasi "Diagnese" yang dapat membantu pasien dalam mengidentifikasi penyakit dan menemukan dokter spesialis sesuai dengan gejala yang mereka alami. Dataset yang digunakan pada penelitian ini berasal dari platform *open source* Kaggle dengan dua file CSV yang berbeda untuk data uji dan data latih. Pada file data latih terdapat sebanyak 4920 data dan pada file data uji sebanyak 41 data. Sebelum menerapkan algoritma *machine learning* pada aplikasi ini, peneliti akan membandingkan beberapa algoritma seperti *Decision Tree, Artificial Neural Network, Super Vector Machine (SVM), Logistic Regression*, dan *Random Forest* untuk mencari performa terbaik. Penelitian ini berfokus pada memprediksi 41 jenis penyakit dengan menggunakan 131 gejala sebagai fitur. Penelitian ini bertujuan untuk membandingkan penerapan berbagai metode *machine learning* yang telah dijelaskan di atas dan menghasilkan akurasi yang tinggi. Diharapkan analisis metode yang dihasilkan memiliki nilai *accuracy, recall*, dan *precision* yang baik, sehingga dapat menjadi acuan untuk penelitian selanjutnya. Serta melalui aplikasi "Diagnese" ini juga, peneliti berharap dapat mengurangi biaya perawatan kesehatan, meningkatkan hasil bagi pasien, dan menjadi solusi berharga dalam mengatasi tantangan dalam industri kesehatan.

2. METODE PENELITIAN

Pada Gambar 1. menunjukkan alur perancangan model untuk prediksi penyakit pada aplikasi Diagnese. Pertama, data yang diperlukan untuk analisis diunduh dari Kaggle dengan judul "*Disease*

Prediction Using Machine Learning With GUI" [10]. Data ini masih berbentuk kategorikal yang nantinya akan di konversi dalam bentuk numerik melalui tahapan *preprocessing*.



Gambar 1. *Flowchart* Perancangan Model

2.1. Data *Exploration*

Pada tahap ini, dilakukan pengecekan kepada 5 data teratas, dimensi data, distribusi data, dan mengecek apakah terdapat kolom yang tidak berisi/*NULL*. Namun, tidak terjadi *imbalance* data pada dataset sehingga tidak perlu melakukan penanganan khusus terhadap ketidakseimbangan data seperti *oversampling* atau *undersampling*.

2.2. Data *Preprocessing*

Pada tahap data *preprocessing*, dilakukan pemisahan kolom target dan kolom fitur, lalu membagi porsi dari file training.csv menjadi data *training* dan *validation*. Selanjutnya, menggunakan *LabelEncoder* untuk menkonversi format kolom label menjadi numerik. Setelah itu, dilakukan pembagian dataset *training* menjadi 80% digunakan untuk data *training* dan 20% digunakan untuk data *validation* [11]. Setelah dilakukan pemisahan data, jumlah data *training* sebanyak 3968 dengan 131 kolom dan data *validation* sebanyak 993 data dengan 131 kolom yang digunakan untuk *validation* kinerja model saat melakukan *training*, sedangkan data *testing* tetap memiliki 41 data dengan 131 kolom yang digunakan untuk melakukan *testing* pada model yang telah dibuat.

2.3. *Modelling*

Pada awal proses pembuatan model peneliti melakukan eksperimen dengan menggunakan beberapa algoritma seperti: *decision tree* yaitu mengacu pada sistem bantuan pilihan yang menggunakan diagram pohon atau prototipe pilihan dan hasil yang memungkinkan [12], SVM adalah satu set model *machine learning* yang melibatkan penggunaan fungsi kernel terkait untuk melakukan regresi (SVR) dan menerapkan klasifikasi non linier (SVC) [13], *logistic regression* bekerja dengan baik untuk memprediksi hasil kategorikal dan hasil multinomial [14], *random forest* didefinisikan sebagai prinsip umum suatu ansambel acak dari suatu pohon keputusan [15]. dan yang terakhir yaitu ANN, untuk membangun model *ANN* peneliti menggunakan *library TensorFlow*. Model ini menggunakan `*Sequential*` sebagai kerangka model dan terdiri dari beberapa lapisan. Lapisan pertama adalah `*Dense*` dengan 64 unit dan fungsi aktivasi *ReLU*, yang mengambil bentuk input yang sesuai dengan dimensi `X_train`. Kemudian terdapat lapisan `*Dropout*` dengan tingkat *dropout* sebesar 0.5, yang berguna untuk mencegah *overfitting*. Terakhir, terdapat lapisan `*Dense*` terakhir dengan jumlah unit yang sesuai dengan jumlah kelas yang dikodekan menggunakan `*label_encoder*`, dan menggunakan fungsi aktivasi *softmax* untuk menghasilkan *output* probabilitas yang dapat diinterpretasikan sebagai prediksi kelas. Beberapa parameter yang digunakan meliputi, *learning rate* sebesar 0.01. Kemudian, model dikompilasi menggunakan *optimizer* SGD (*Stochastic Gradient Descent*) dengan tingkat pembelajaran yang telah ditentukan, *loss function* menggunakan *sparse categorical cross-entropy* karena cocok untuk masalah klasifikasi multikelas dengan label yang dikodekan secara numerik, dan

metrik akurasi untuk mengevaluasi kinerja model selama pelatihan dan validasi. Selanjutnya, model dilatih dengan 20 *epoch* (siklus pelatihan). Data pelatihan (X_train dan y_train_encoded) digunakan sebagai input, dengan *batch size* sebesar 13. Selama pelatihan, penulis juga menampilkan informasi *verbose* (detail) dengan tingkat kejelasan 2. Validasi data (X_val dan y_val_encoded) digunakan untuk memantau kinerja model selama pelatihan.

2.4. *Evaluation*

Untuk *evaluation* menggunakan *classification report* dengan *metric* evaluasi berupa *precision*, *recall*, *f1-score*, dan *support.* Tujuan dari *metric* evaluasi adalah untuk mengukur kualitas kedekatan dengan kebenaran atau nilai aktual yang dicapai oleh suatu sistem [16]. Kinerja model dalam mengklasifikasikan setiap kelas menunjukkan hasil yang sangat positif, tergambar melalui nilai presisi, *recall*, dan skor F1 yang bernilai 1. Dari hasil ini dapat disimpulkan bahwa model yang telah dibuat memiliki kemampuan yang sangat baik dalam mengklasifikasikan berbagai jenis penyakit. Model berhasil menangkap pola-pola yang relevan dari data pelatihan dan menerapkannya secara efektif dalam data pengujian.

3. HASIL DAN PEMBAHASAN

Penulis mendapatkan dataset dari situs *open source kaggle* dengan judul "*Disease Prediction using Machine Learning with GUI*" [10]. Di dalam dataset tersebut memiliki 2 *file* csv, yaitu sebanyak 4961 data dalam bentuk csv untuk *training*, dan 41 data untuk *testing*.

Dalam proses *modelling*, penulis melakukan 5 percobaan algoritma berbeda, yaitu *Neural Network, Decision Tree, Random Forest, Logistic Regression*, dan *Support Vector Machin*e. Selain itu, penulis juga melakukan pengaturan beberapa parameter untuk mencari model dengan performa terbaik. Tujuan dari melakukan banyak percobaan tersebut adalah untuk mencari model dengan performa terbaik, baik dalam segi keakuratan, efisiensi, dan kecepatan model dalam melakukan prediksi.

3.1. Percobaan pada *Neural Network*

Berikut adalah dokumentasi dari *modelling* menggunakan *Neural Network* yang penulis lakukan:

Tabel 1. Percobaan berbagai *hyperparameter neural network*

Algoritma	Hyperparameter	Acc	Val acc	Test
Neural Network	2 Layer, 1 Dropout, Lr = 0.01	97%	99%	95%
	3 Layer, 1 Dropout, Lr = 0.01	99%	100%	95%
	3 Layer, tanpa dropout, Lr = 0.01	99%	100%	95%

Dalam penelitian ini, dilakukan serangkaian percobaan dalam *modelling* menggunakan *Neural Network.* Percobaan ini bertujuan untuk mencari konfigurasi model terbaik dengan memvariasikan jumlah *layer* dan penggunaan *dropout*. Pada Tabel 1, terdapat tiga percobaan yang dilakukan dengan menggunakan *Neural Network*.

3.1.1. Hasil Percobaan pada *Neural Network*

Percobaan pertama menggunakan 2 *layer*, 1 *dropout*, dan *learning rate* (Lr) sebesar 0.01, menghasilkan akurasi sebesar 97% pada data pelatihan, 99% pada data validasi, dan 95% pada data uji. Percobaan kedua menggunakan 3 *layer*, 1 *dropout*, dan *Lr* = 0.01, menghasilkan akurasi sebesar 99% pada data pelatihan, 100% pada data validasi, dan 95% pada data uji. Percobaan terakhir menggunakan 3 *layer* tanpa *dropout* dan Lr = 0.01, dengan akurasi sebesar 99% pada data pelatihan, 100% pada data validasi, dan 95% pada data uji.

3.2. Percobaan pada *Decision Tree*

Berikut adalah dokumentasi dari *modelling* menggunakan *Decision Tree* yang penulis lakukan:

Tabel 2. Percobaan berbagai macam *hyperparameter decision tree*

Algoritma	Hyperparameter	Acc	Val acc	Test
Decision Tree	criterion='gini', min_samples_split = 10, max_depth = 40	97%	96%	80%
	criterion='gini', min_samples_split = 10, max_depth = 50,	99%	97%	87%
	criterion='entropy', min_samples_split = 5, max_depth = 50	99%	99%	86%
	criterion='entropy', min_samples_split = 10 max_depth = 40,	98%	96%	75%

Dalam penelitian ini, juga dilakukan serangkaian percobaan dalam modelling menggunakan *Decision Tree*. Percobaan ini bertujuan untuk mencari konfigurasi hyperparameter yang optimal dalam membangun model *Decision Tree*. Pada Tabel 2, terdapat empat percobaan yang dilakukan dengan menggunakan *Decision Tree*.

3.2.1. Hasil Percobaan pada *Decision Tree*

Percobaan pertama menggunakan *criterion 'gini'*, *min_samples_split* = 10, dan *max_depth* = 40, menghasilkan akurasi sebesar 97% pada data pelatihan, 96% pada data validasi, dan 80% pada data uji. Percobaan kedua menggunakan *criterion 'gini'*, *min_samples_split* = 10, dan *max_depth* = 50, dengan akurasi sebesar 99% pada data pelatihan, 97% pada data validasi, dan 87% pada data uji. Percobaan ketiga menggunakan *criterion 'entropy'*, *min_samples_split* = 5, dan *max_depth* = 50, menghasilkan akurasi sebesar 99% pada data pelatihan, 99% pada data validasi, dan 86% pada data uji. Percobaan terakhir menggunakan *criterion 'entropy'*, *min_samples_split = 10*, dan *max_depth* =

40, dengan akurasi sebesar 98% pada data pelatihan, 96% pada data validasi, dan 75% pada data uji.

3.3. Percobaan pada *Random Forest*

Berikut adalah dokumentasi dari *modelling* menggunakan *Random Forest* yang penulis lakukan:

Tabel 3. Percobaan berbagai macam *hyperparameter random forest*

Algoritma	Hyperparameter	Acc	Val acc	Test
Random Forest	n_estimators = 50, max_depth = 8, min_samples_split = 300	99%	98%	75%
	n_estimators = 50, max_depth = 16, min_samples_split = 300	99%	99%	75%
	n_estimators = 50, max_depth = 8, min_samples_split = 100	99%	99%	75%
	n_estimators = 50, max_depth = 16, min_samples_split = 100	99%	99%	75%

Selanjutnya dalam penelitian ini juga, dilakukan serangkaian percobaan dalam modelling menggunakan *Random Forest*. Percobaan ini bertujuan untuk mencari konfigurasi *hyperparameter* yang optimal dalam membangun model *Random Forest*. Pada Tabel 3, terdapat empat percobaan yang dilakukan dengan menggunakan *Random Forest*.

3.3.1. Hasil Percobaan pada *Random Forest*

Percobaan pertama menggunakan *n_estimators = 50, max_depth = 8,* dan *min_samples_split = 300,* menghasilkan akurasi sebesar 99% pada data pelatihan, 98% pada data validasi, dan 75% pada data uji. Percobaan kedua menggunakan *n_estimators = 50, max_depth = 16,* dan *min_samples_split = 300,* dengan akurasi sebesar 99% pada data pelatihan, 99% pada data validasi, dan 75% pada data uji. Percobaan ketiga menggunakan *n_estimators = 50, max_depth = 8,* dan *min_samples_split = 100,* menghasilkan akurasi sebesar 99% pada data pelatihan, 99% pada data validasi, dan 75% pada data uji. Percobaan terakhir menggunakan *n_estimators = 50, max_depth = 16,* dan *min_samples_split = 100,* dengan akurasi sebesar 99% pada data pelatihan, 99% pada data validasi, dan 75% pada data uji. Dari hasil percobaan ini, dapat disimpulkan bahwa semua percobaan memiliki akurasi yang sama pada data uji, tetapi percobaan pertama, yang menggunakan n_estimators = 50, max_depth = 8, dan min_samples_split = 300, memiliki akurasi yang paling rendah pada data validasi.

3.4. Percobaan pada *Logistik Regression*

Berikut adalah dokumentasi dari *modelling* menggunakan *Logistik Regression* yang penulis lakukan:

Tabel 4. Percobaan berbagai macam *hyperparameter logistik regression*

Algoritma	Hyperparameter	Acc	Val acc	Test
Logistik Regression	Tanpa Parameter	100%	100%	98%
	C = 0.01	100%	100%	93%
	C = [0.001, 0.1, 1, 10, 100]	100%	100%	98%

Dalam penelitian ini, dilakukan juga serangkaian percobaan dalam *modelling* menggunakan *Logistik Regression*. Percobaan ini bertujuan untuk mencari konfigurasi *hyperparameter* yang optimal dalam membangun model *Logistik Regression*. Pada Tabel 4, terdapat tiga percobaan yang dilakukan dengan menggunakan *Logistik Regression*.

3.4.1. Hasil Percobaan pada *Logistik Regression*

Percobaan pertama menggunakan logistik regression tanpa parameter tambahan, menghasilkan akurasi sebesar 100% pada data pelatihan, 100% pada data validasi, dan 98% pada data uji. Percobaan kedua menggunakan C = 0.01, dengan akurasi sebesar 100% pada data pelatihan, 100% pada data validasi, dan 93% pada data uji. Percobaan terakhir menggunakan C = [0.001, 0.1, 1, 10, 100], dengan akurasi sebesar 100% pada data pelatihan, 100% pada data validasi, dan 98% pada data uji. Dari hasil percobaan ini, dapat disimpulkan bahwa semua percobaan memiliki akurasi yang sangat tinggi pada data pelatihan dan data validasi. Namun, percobaan kedua, yang menggunakan C = 0.01, memiliki akurasi yang sedikit lebih rendah pada data uji dibandingkan dengan percobaan lainnya.

3.5. Percobaan pada *Support Vector Machine*

Berikut adalah dokumentasi dari m*odelling* menggunakan *Support Vector Machine* yang penulis lakukan:

Tabel 5. Percobaan berbagai macam *hyperparameter Support Vector Machine*

Algoritma	Hyperparameter	Val acc	Test
Support Vector Machine	SVC(kernel='linear'), param_grid = 0.01, cv=5	99%	95%
	SVC(kernel='linear'), param_dist = {'C': reciprocal(0.001, 100)}, n_iter=10, cv=5	99%	97%

Terakhir, didalam penelitian ini, dilakukan serangkaian percobaan dalam *modelling* menggunakan *Support Vector Machine* (SVM). Percobaan ini bertujuan untuk mencari konfigurasi *hyperparameter* yang optimal dalam membangun model SVM. Pada Tabel 5, terdapat dua percobaan yang dilakukan dengan menggunakan SVM.

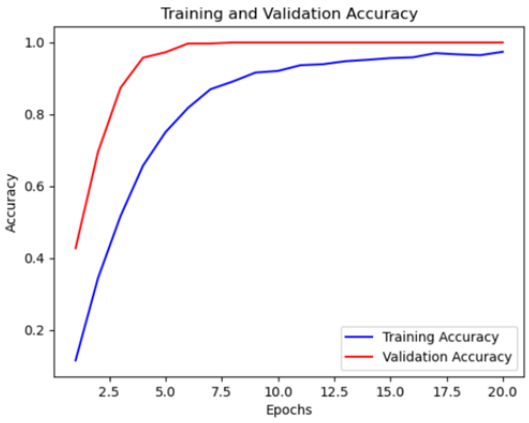
3.5.1. Hasil Percobaan pada *Support Vector Machine*

Percobaan pertama menggunakan SVM dengan *kernel='linear'*, *param_grid* = 0.01, dan cv=5. Hasilnya adalah akurasi sebesar 99% pada data pelatihan, 95% pada data validasi, dan belum disebutkan akurasi pada data uji. Percobaan kedua menggunakan SVM dengan *kernel='linear'*, *param_dist* = {'C': *reciprocal*(0.001, 100)}, n_iter=10, dan cv=5. Hasilnya adalah akurasi sebesar 99% pada data pelatihan, 97% pada data validasi, dan belum disebutkan akurasi pada data uji. Dari hasil percobaan ini, dapat disimpulkan bahwa kedua percobaan menghasilkan akurasi yang tinggi pada data pelatihan dan data validasi.

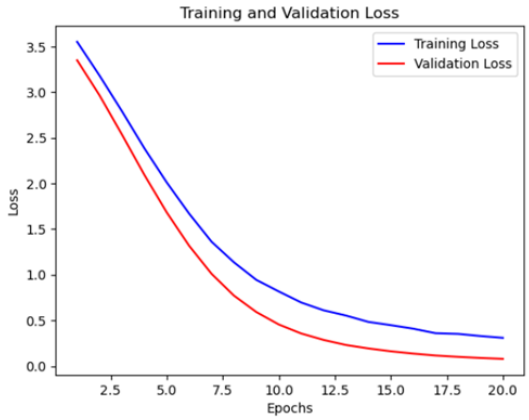
Berdasarkan hasil percobaan yang dilakukan dengan berbagai algoritma, dapat disimpulkan bahwa algoritma terbaik adalah menggunakan *Neural Network*. Pada percobaan menggunakan Neural Network, akurasi pada data pelatihan, data validasi, dan data uji memiliki nilai yang tinggi dan hampir berdekatan. Selanjutnya, untuk memilih parameter terbaik dalam *Neural Network*, evaluasi perlu dilakukan dengan menggunakan grafik akurasi dan *loss*.

3.6. Performa Tiap Percobaan Model *Neural Network*

3.6.1. Performa Percobaan Model 1



Gambar 2. Grafik *Accuracy* percobaan 1

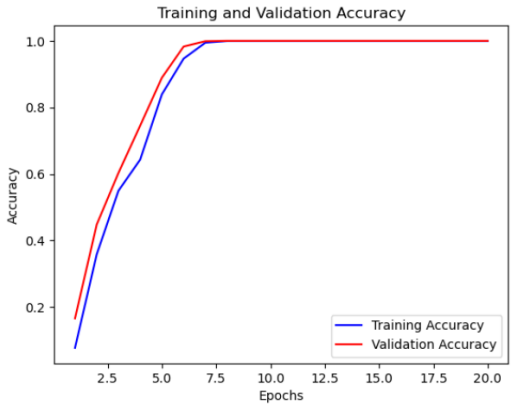


Gambar 3. Grafik *Loss* percobaan 1

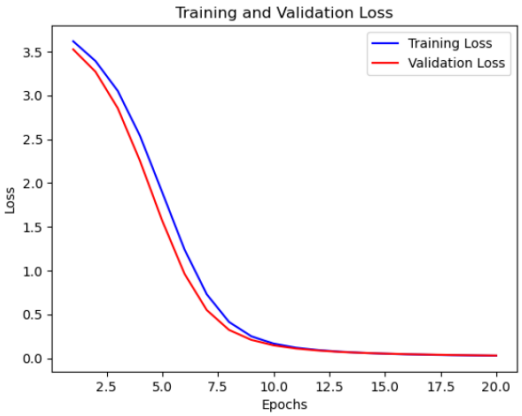
Dalam Gambar 2, terlihat plot akurasi model yang dipresentasikan. Pada grafik tersebut, akurasi model terhadap data pelatihan (*accuracy*) dan akurasi model terhadap data validasi (*val_accuracy*) dievaluasi pada setiap *epoch*. Dapat dilihat bahwa akurasi pada data pelatihan dan data validasi meningkat secara berdekatan seiring dengan berjalannya *epoch*, menunjukkan peningkatan kinerja model. Kedua kurva akurasi tersebut cenderung bergerak bersamaan, yang mengindikasikan bahwa model tidak mengalami *overfitting* atau *underfitting*.

Sedangkan dalam Gambar 3, terlihat plot *loss* model yang dipresentasikan. Pada grafik tersebut, *loss* model terhadap data pelatihan (*loss*) dan *loss* model terhadap data validasi (*val_loss*) dievaluasi pada setiap *epoch*. Dalam contoh ini, dapat dilihat bahwa *loss* pada data pelatihan dan data validasi menurun secara berdekatan seiring dengan berjalannya *epoch*, menunjukkan peningkatan dalam kemampuan model untuk meminimalkan kesalahan. Kurva *loss* dan *val_loss* cenderung bergerak bersamaan, yang mengindikasikan bahwa model tidak mengalami *overfitting*.

3.6.2. Grafik Performa Percobaan Model 2



Gambar 4.Grafik *Accuracy* percobaan 2



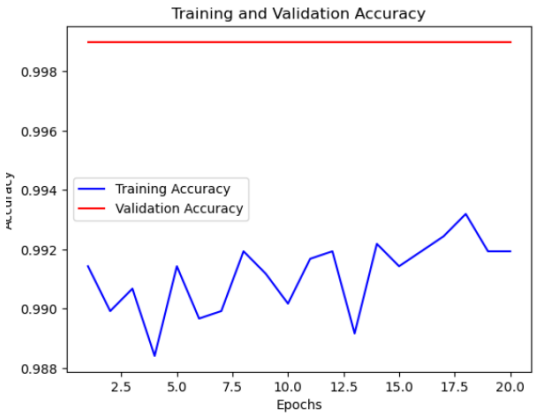
Gambar 5. Grafik *Loss* percobaan 2

Dalam Gambar 4, terlihat bahwa setelah mencapai *epoch* ke-5, terjadi perubahan pada akurasi model terhadap data pelatihan (*accuracy*) dan data validasi (*val_accuracy*). Pada awalnya, kedua kurva akurasi tersebut meningkat secara berdekatan,

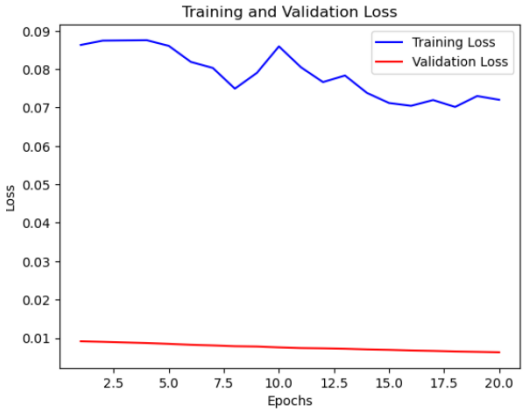
menunjukkan peningkatan kinerja model. Namun, setelah *epoch* ke-5, garis akurasi pada data t*raining* dan validasi menjadi datar dan mulai mendekati nilai 1. Hal ini mengindikasikan bahwa model mulai mengalami *overfitting*, di mana model terlalu "terbiasa" dengan data pelatihan dan tidak dapat secara efektif menggeneralisasi pada data baru meskipun gambar 5 menunjukan penurunan *loss* yang berdekatan.

Oleh karena itu, epoch ke-5 menjadi titik kritis di mana perlu dilakukan evaluasi lebih lanjut terhadap model untuk menghindari *overfitting* dan memastikan bahwa model memiliki kemampuan yang baik untuk melakukan prediksi pada data yang belum pernah dilihat sebelumnya.

3.6.3. Grafik Performa percobaan model 3



Gambar 6.Grafik *Accuracy* percobaan 3



Gambar 7. Grafik *Loss* percobaan 3

Dalam Gambar 6, terlihat bahwa data *training* accuracy mengalami fluktuasi (*spiking*) di rentang 0.988 sampai 0.992, sedangkan data validasi accuracy mendatar mendekati 1. Hal ini menunjukkan bahwa model memiliki kinerja yang sangat baik pada data validasi, dengan tingkat akurasi yang konsisten dan tinggi. Namun, fluktuasi yang terjadi pada data training accuracy menunjukkan bahwa model mengalami *overfitting* pada data pelatihan, di mana model secara tidak proporsional "mengingat" atau "menghafal" data pelatihan yang kompleks, tetapi tidak generalis pada data baru.

Dalam Gambar 7, terlihat bahwa data *training loss* mengalami fluktuasi (*spiking*), sedangkan data validasi loss mendatar mendekati 0. Hal ini menunjukkan bahwa model memiliki kemampuan yang baik dalam meminimalkan kesalahan pada data validasi. Namun, fluktuasi yang terjadi pada data *training loss* menunjukkan bahwa model mengalami *overfitting* pada data pelatihan, di mana model secara tidak proporsional mempelajari *noise* atau variasi yang terdapat dalam data pelatihan, yang pada akhirnya mengarah pada peningkatan *loss*.

Meskipun model memiliki performa yang baik pada data validasi, fluktuasi yang terjadi pada data training accuracy dan training loss menunjukkan adanya penyesuaian yang berlebihan terhadap data pelatihan yang kompleks, sehingga kemampuan generalisasi pada data baru mungkin terpengaruh. Oleh karena itu, perlu dilakukan upaya untuk mengurangi *overfitting*, seperti penggunaan teknik regularisasi, pengurangan kompleksitas model, atau pengumpulan data yang lebih banyak.

3.7. Analisa Hasil Penelitian

Berdasarkan hasil analisis pada 3 percobaan pada *Neural Network*, dapat disimpulkan bahwa model *Neural Network* dalam percobaan 1 memiliki performa yang terbaik. Dalam Gambar 1, terlihat bahwa akurasi model terhadap data pelatihan (*accuracy*) dan data validasi (*val_accuracy*) meningkat secara berdekatan seiring dengan berjalannya *epoch*, menunjukkan peningkatan kinerja model yang stabil. Hal ini mengindikasikan bahwa model tidak mengalami *overfitting* atau *underfitting*, dan mampu melakukan prediksi dengan akurat pada data yang belum pernah dilihat sebelumnya.

Gambar 3 juga memperlihatkan bahwa *loss* model terhadap data pelatihan (*loss*) dan data validasi (*val_loss*) menurun secara berdekatan seiring dengan berjalannya *epoch*. Hal ini menunjukkan bahwa model mampu meminimalkan kesalahan dengan baik, dan kemampuan model dalam mempelajari pola-pola yang ada dalam data semakin meningkat.

Dengan demikian, performa percobaan 1 dapat dianggap sebagai yang terbaik karena model mampu menghasilkan akurasi yang tinggi pada data validasi dan meminimalkan kesalahan dengan baik pada data pelatihan. Penggunaan model ini dapat diandalkan untuk melakukan prediksi dengan tingkat keakuratan yang tinggi.

3.8. Evaluasi

	precision	recall	f1-score	support
AIDS	1.00	1.00	1.00	20
Alergi	1.00	1.00	1.00	29
Artritis	1.00	1.00	1.00	24
Asma Bronkial	1.00	1.00	1.00	30
Cacar air	1.00	1.00	1.00	27
Demam berdarah	1.00	1.00	1.00	28
Diabetes	1.00	1.00	1.00	30
GERD	1.00	1.00	1.00	25
Gastroenteritis	1.00	1.00	1.00	18
Hemoroid dimorfik (ambeien)	1.00	1.00	1.00	31
Hepatitis A	1.00	1.00	1.00	21
Hepatitis Alkoholik	1.00	1.00	1.00	26
Hepatitis B	1.00	1.00	1.00	29
Hepatitis C	1.00	1.00	1.00	26
Hepatitis D	1.00	1.00	1.00	29

Hepatitis E	1.00	1.00	1.00	19
Hipertensi	1.00	0.96	0.98	26
Hipertiroidisme	1.00	1.00	1.00	15
Hipoglikemia	1.00	1.00	1.00	27
Hipotiroidisme	1.00	1.00	1.00	17
Impetigo	1.00	1.00	1.00	16
Infeksi jamur	1.00	1.00	1.00	23
Infeksi saluran kemih	1.00	1.00	1.00	20
Jerawat	1.00	1.00	1.00	33
Kolestasis kronis	1.00	1.00	1.00	22
Kuning (penyakit kuning)	1.00	1.00	1.00	27
Malaria	1.00	1.00	1.00	23
Migraine	1.00	1.00	1.00	27
Osteoartritis	1.00	1.00	1.00	21
Paralisis (pendarahan otak)	1.00	1.00	1.00	23
Penyakit ulkus peptikum	1.00	1.00	1.00	25
Pilek biasa	1.00	1.00	1.00	25
Pneumonia	1.00	1.00	1.00	23
Psoriasis	1.00	1.00	1.00	19
Reaksi obat	1.00	1.00	1.00	27
Serangan jantung	1.00	1.00	1.00	21
Spondilosis Serviks	1.00	1.00	1.00	21
Tuberculosis	1.00	1.00	1.00	22
Typus	1.00	1.00	1.00	24
Varises	1.00	1.00	1.00	27
Vertigo Posisional Paroksimal	0.96	1.00	0.98	27
accuracy			1.00	993
macro avg	1.00	1.00	1.00	993
weighted avg	1.00	1.00	1.00	993

Terlihat hasil metrik evaluasi yang lebih rinci seperti *precision*, *recall*, dan *f1-score* untuk setiap kelas pada data pengujian. Hasil evaluasi menunjukkan bahwa model memiliki performa yang sangat baik dengan nilai akurasi 1.00 serta *precision*, *recall*, dan *f1-score* yang baik (1.00) untuk setiap kelas. Dengan demikian, model mampu melakukan klasifikasi dengan sangat baik pada data pengujian

Selanjutnya adalah menyimpan model dengan performa bagus ke format H5. Dengan menyimpan model ke dalam file .h5, model dapat dipanggil dan digunakan kembali di waktu yang akan datang tanpa perlu melakukan proses pelatihan ulang. Ini memungkinkan untuk menghindari waktu yang diperlukan untuk melatih ulang model dan mempertahankan kemampuan prediksi yang telah diperoleh dari model yang dilatih sebelumnya.

Selanjutnya adalah menyimpan model dengan performa bagus ke format H5. Dengan menyimpan model ke dalam file .h5, model dapat dipanggil dan digunakan kembali di waktu yang akan datang tanpa perlu melakukan proses pelatihan ulang. Ini memungkinkan untuk menghindari waktu yang diperlukan untuk melatih ulang model dan mempertahankan kemampuan prediksi yang telah diperoleh dari model yang dilatih sebelumnya.

4. DISKUSI

4.1. Penelitan Sebelumnya

Dalam penelitian-penelitian sebelumnya mengenai prediksi penyakit menggunakan *Neural Network* dan *Deep Neural Network*, telah terbukti bahwa penggunaan model ini dapat memberikan prediksi yang akurat. Pada tahun 2013, penelitian oleh Bakhtiar Rifai menunjukkan bahwa model *Neural Network* memiliki tingkat akurasi 91,45% dan nilai AUC 0,937 dalam prediksi penyakit jantung. Pada tahun 2021, Anas Faisal dan Agus Subekti menggunakan *Deep Neural Network* untuk prediksi stroke dengan tingkat akurasi 0,96, *f1-score* 0,9611, dan AUC 0,981. Pada tahun 2023, Simeon Yuda Prasetyo menggunakan *Artificial Neural Network* untuk prediksi gagal jantung dengan tingkat akurasi 92,032% dan AUC 93%.

4.2. Interpretasi Hasil

Dalam keseluruhan penelitian-penelitian tersebut, penggunaan *Neural Network*, termasuk *Deep Neural Network* dan *Artificial Neural Network*, telah terbukti efektif dalam meningkatkan akurasi prediksi penyakit. Dengan demikian, dalam konteks penelitian ini, *Neural Network* dengan 2 *layer*, 1 *Dropout*, dan *learning rate* 0,01 dapat dipilih sebagai model terbaik untuk prediksi penyakit yang diteliti. Model ini memiliki tingkat akurasi yang tinggi dan dapat memberikan prediksi yang akurat dalam mengidentifikasi hingga 41 jenis penyakit.

4.3. Saran

Berdasarkan hasil penelitian ini, terdapat beberapa saran untuk pengembangan lebih lanjut dalam prediksi penyakit menggunakan model *Machine Learning*, antara lain perluasan *dataset* dengan lebih banyak data gejala penyakit terkait untuk meningkatkan generalisasi model dan kemampuan prediksi yang lebih beragam, evaluasi model dengan data *real-time* dari lingkungan klinis melibatkan ahli medis dan uji coba pada data pasien nyata, pengembangan antarmuka pengguna yang intuitif dan mudah digunakan, serta penelitian lanjutan untuk menjelajahi penggunaan arsitektur yang lebih kompleks atau algoritma *Machine Learning* yang berbeda, dan melibatkan analisis interpretabilitas model untuk memahami kontribusi fitur dan keterkaitannya dengan diagnosis penyakit.

5. KESIMPULAN

Penelitian ini berhasil menghasilkan model *Machine Learning* yang dapat melakukan prediksi penyakit dengan tingkat akurasi yang tinggi berdasarkan gejala-gejala yang terdapat dalam dataset. Model ini mencapai tingkat akurasi yang tinggi pada data *training*, *validation*, dan *testing*, dengan akurasi masing-masing sebesar 97%, 99%, dan 95%. Grafik akurasi dan *loss* model juga menunjukkan peningkatan yang baik seiring dengan berjalannya *epoch*. Dengan adanya model ini, diharapkan dapat meningkatkan efisiensi dan keakuratan dalam proses diagnosis medis, sehingga membantu dokter dalam pengambilan keputusan yang lebih tepat dan memberikan perawatan yang lebih efektif kepada pasien.

DAFTAR PUSTAKA

[1] D. Manongga, U. Rahardja, I. Sembiring, N. Lutfiani, and A. B. Yadila, "Dampak Kecerdasan Buatan Bagi Pendidikan," *ADI Bisnis Digital Interdisiplin Jurnal*, vol. 3, no. 2, pp. 41–55, 2022, doi: 10.34306/abdi.v3i2.792.

[2] Y. A. Hasma and W. Silfianti, "Implementasi Deep Learning Menggunakan Framework Tensorflow Dengan Metode Faster Regional

Convolutional Neural Network Untuk Pendeteksian Jerawat," *Jurnal Ilmiah Teknologi dan Rekayasa*, vol. 23, no. 2, pp. 89–102, 2018, doi: 10.35760/tr.2018.v23i2.2459.

[3] S. Wahyuni and M. Sulaeman, "Penerapan Algoritma Deep Learning Untuk Sistem Absensi Kehadiran Deteksi Wajah Di PT Karya Komponen Presisi," *Jurnal Informatika SIMANTIK*, vol. 7, no. 1, pp. 5–6, 2022.

[4] World Health Organization (WHO), "The top 10 causes of death." 2020.

[5] R. Rachman, "Sistem Pakar Deteksi Penyakit Refraksi Mata Dengan Metode Teorema Bayes Berbasis Web," *Jurnal Informatika*, vol. 7, no. 1, pp. 68–76, 2020, doi: 10.31311/ji.v7i1.7267.

[6] A. NurJumala, N. A. Prasetyo, and H. W. Utomo, "Sistem Pakar Diagnosis Penyakit Rhinitis Menggunakan Metode Forward Chaining Berbasis Web," *JURIKOM (Jurnal Riset Komputer)*, vol. 9, no. 1, p. 69, 2022, doi: 10.30865/jurikom.v9i1.3815.

[7] R. Deshmukh, P. Gourkhede, and S. Rangari, "Heart Disease Prediction Using Artificial Neural Network," *Ijarcce*, vol. 8, no. 1, pp. 85–89, 2019, doi: 10.17148/ijarcce.2019.8119.

[8] A. Faisal and A. Subekti, "Deep Neural Network untuk Prediksi Stroke," *JEPIN (Jurnal Edukasi dan Penelitian Informatika)*, vol. 7, no. 3, pp. 443–449, 2021.

[9] S. Y. Prasetyo, "Prediksi Gagal Jantung Menggunakan Artificial Neural Network," *Jurnal SAINTEKOM*, vol. 13, no. 1, pp. 79–88, Mar. 2023, doi: 10.33020/saintekom.v13i1.379.

[10] Neelima, "DISEASE PREDICTION USING MACHINE LEARNING WITH GUI | Kaggle." 2019.

[11] F. Dharma Adhinata, D. Putra Rakhmadani, M. Wibowo, and A. Jayadi, "A Deep Learning Using DenseNet201," vol. 9, no. 1, pp. 115–121, 2021.

[12] R. O. Ogundokun, P. O. Sadiku, S. Misra, O. E. Ogundokun, J. B. Awotunde, and V. Jaglan, "Diagnosis of Long Sightedness Using Neural Network and Decision Tree Algorithms," *Journal of Physics: Conference Series*, vol. 1767, no. 1, 2021, doi: 10.1088/1742-6596/1767/1/012021.

[13] D. A. Otchere, T. O. Arbi Ganat, R. Gholami, and S. Ridha, "Application of supervised machine learning paradigms in the prediction of petroleum reservoir properties: Comparative analysis of ANN and SVM models," *Journal of Petroleum Science and Engineering*, vol. 200, no. August 2020, p. 108182, 2021, doi: 10.1016/j.petrol.2020.108182.

[14] T. Ciu and R. S. Oetama, "Logistic Regression Prediction Model for Cardiovascular Disease," *IJNMT*, vol. VII, no. 1, p. 33, Jun. 2020.

[15] N. Wuryani, S. Agustiani, I. Komputer, and N. Mandiri, "Random Forest Classifier untuk Deteksi Penderita COVID-19 berbasis Citra CT Scan," *Jurnal Teknik Komputer AMIK BSI*, vol. 7, no. 2, 2021, doi: 10.31294/jtk.v4i2.

[16] F. Putra Utama, T. Mardiansyah, R. Faurina, and A. Vatresia, "Scientific Articles Recommendation System Based On User's Relatedness Using Item-Based Collaborative Filtering Method," *Jurnal Teknik Informatika (Jutif)*, vol. 4, no. 3, pp. 467–475, Jun. 2023, doi: 10.52436/1.jutif.2023.4.3.702..