JNATIA Volume 3, Nomor 2, Februari 2025 p-ISSN: 2986-3929
Jurnal Nasional Teknologi Informasi dan Aplikasinya e-ISSN: 3032-1948

Optimasi Hyperparameter Algoritma Support Vector Machine dalam Klasifikasi Penyakit β-Thalassemia

I Nyoman Adi Mahendra Putra[a1], Cokorda Pramartha[a2]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana
Jalan Raya Kampus Udayana, Bukit Jimbaran, Kuta Selatan, Badung, Bali, Indonesia
[1]henptra@gmail.com
[2]cokorda@unud.ac.id

Abstract

Beta-thalassemia, a type of thalassemia disease caused by genetic variations, forces sufferers to receive regular blood transfusions for survival. Therefore, classification of this disease is important to reduce the number of births of beta-thalassemia patients in the future. 5066 beta-thalassemia carrier patient data from the Punjab Thalassemia Prevention and Program (PTPP) case study was taken as the source in this study which was accessed through the github.com website. Preprocessing is done for class alignment to avoid data imbalance, feature selection to streamline model performance, and normalization of feature values to a certain scale. In this research, the main focus is on applying the performance of the Support Vector Machine (SVM) algorithm to obtain classification results. Before entering the final model, hyperparameter tuning is required to obtain suitable parameter values to be entered into the model. hyperparameter tuning that will be carried out include "C" (regularization parameter), "kernel" (kernel type), "gamma" (kernel parameter for non-linear kernels), and "degree" (polynomial degree for polynomial kernels), carried out before the model is evaluated. The accuracy results were evaluated using confusion matrix, resulting in precision of 99.81%, recall of 99.62%, f1-score of 99.71%, and accuracy of 99.71% after hyperparameter tuning where the best parameters are "'C': 1, 'gamma': 100, 'kernel': 'rbf'" with an average test score of 0.993494149.

Keywords: *Beta-thalassemia, Classification, Support Vector Machine, Hyperparameter Tuning, Confusion Matrix*

1. Pendahuluan

Thalassemia adalah sebuah kelainan genetik yang disebabkan oleh kurangnya bahkan tidak ada pembentukan satu atau lebih rantai globin yang membentuk hemoglobin (Hb) [1]. Data Kementerian Kesehatan RI menunjukkan bahwa sekitar 2500 bayi Indonesia terlahir dengan thalassemia mayor atau pada tingkat thalassemia yang berbahaya setiap tahunnya [2]. Beta-thalassemia merupakan salah satu jenis penyakit thalassemia yang disebabkan oleh pengaruh sejumlah variasi genetik pada berbagai tahap produksi globin beta, seperti proses transkripsi, translasi, atau stabilitas produk beta-globin [3]. Dampak dari beta-thalassemia adalah terbentuknya hemoglobin yang cacat, yang lebih mungkin untuk mengalami kerusakan dan terurai. Kerusakan pada hemoglobin menyebabkan individu yang menderita beta-thalassemia harus secara rutin menerima transfusi darah untuk keberlangsungan hidupnya [4]. Oleh karena itu, pentingnya mengklasifikasikan jenis penyakit beta-thalassemia untuk mengurangi jumlah kelahiran penderita beta-thalassemia kedepannya. Klasifikasi merupakan proses penting dalam analisis data yang bertujuan untuk mengelompokkan data ke dalam kategori atau kelas tertentu berdasarkan fitur-fitur yang dimilikinya. Dalam klasifikasi data terdapat banyak algoritma yang sering digunakan untuk mendapatkan hasil klasifikasi yang baik salah satunya algoritma Support Vector Machine (SVM). SVM memiliki keunggulan dalam kemampuannya untuk menentukan superclass yang berbeda sehingga dapat memaksimalkan margin antara dua kelas yang berbeda [5]. Terdapat banyak penelitian yang sudah menerapkan algoritma Support Vector Machine (SVM) ke dalam berbagai kasus terutama penyakit. Pada penelitian sebelumnya algoritma Support Vector Machine (SVM) diterapkan untuk mengklasifikasikan kategori obat dengan hasil

akurasi terbaik didapatkan melalui penggunaan kernel “linear” dan “polinomial” sebesar 95,0% [6]. Adapun penelitian lain yang juga menggunakan algoritma Support Vector Machine (SVM) dalam medis yaitu memprediksi penyakit stroke. Permasalahan dataset yang digunakan tidak seimbang sehingga penulis menggunakan teknik oversampling untuk menyamakan distribusi kelas. Pada penelitian tersebut, dua hyperparameter dari Support Vector Machine (SVM) yakni “C” dan “Gamma” dilakukan tuning untuk mendapatkan hasil akurasi yang baik. Dalam uji parameter yang dilakukan, didapatkan hasil terbaiknya yaitu akurasi sebesar 96% dengan hyperparameter tuning dan 77% sebelum hyperparameter tuning [7]. Dalam upaya mengklasifikasikan penyakit beta-thalassemia, penting untuk menggunakan algoritma yang efektif guna memastikan akurasi yang optimal. Penelitian ini bertujuan untuk memanfaatkan performa algoritma Support Vector Machine (SVM) untuk mendapatkan model terbaik dalam mengklasifikasikan penyakit beta-thalassemia. Dengan melakukan hyperparameter tuning yang diterapkan pada penggunaan algoritma Support Vector Machine (SVM), diharapkan mampu mengoptimalkan hasil akurasi akhir yang baik. Oleh karena itu, diharapkan bahwa hasil dari penelitian ini akan memberikan manfaat bagi masyarakat luas serta menjadi referensi penelitian lebih lanjut kedepannya.

2. Metode Penelitian

2.1. Pengumpulan Data

Data yang diambil dalam penelitian ini berasal dari sumber sekunder, yakni pasien yang merupakan pembawa beta-thalassemia, yang diperoleh dari studi kasus Punjab Thalassemia Prevention and Programme (PTPP) melalui situs github.com. Data disimpan dalam bentuk file “.csv” dengan 5066 pasien yang telah melakukan screening untuk beta-thalassemia pada tahun 2019. Dari jumlah tersebut, 2594 pasien terbukti sebagai pembawa beta-thalassemia, sementara 2472 normal. Fitur yang terdapat dalam dataset dapat dilihat pada Tabel 1.

Tabel 1. Fitur Dataset

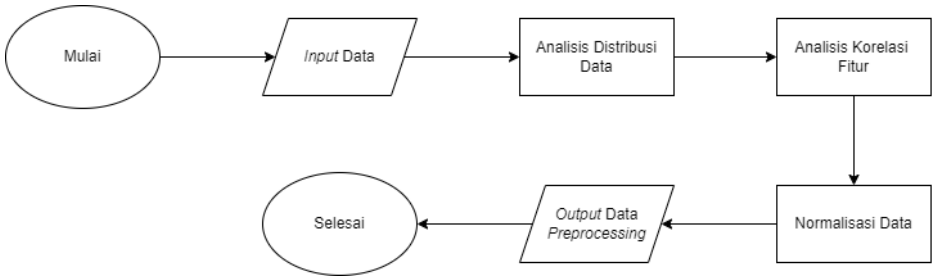
No	Fitur	Tipe	Keterangan
1	Age	Numerik	Umur pasien, yang terdiri dari angka 0 untuk pasien dewasa dan angka 1 untuk pasien anak-anak
2	Sex	Numerik	Jenis kelamin pasien, yang terdiri dari angka 0 untuk pasien laki-laki dan angka 1 untuk pasien perempuan
3	WBC	Numerik	White blood count (WBC) yang merupakan jumlah sel darah putih pasien
4	RBC	Numerik	Red blood cell (RBC) yang merupakan jumlah sel darah merah pasien yang normalnya sekitar
5	HGB	Numerik	Jumlah hemoglobin (HGB) pasien
6	HCT	Numerik	Jumlah hematokrit (HCT) pasien
7	MCH	Numerik	Mean corpuscular hemoglobin (MCH) yang merupakan rata-rata sel darah merah pasien
8	MCV	Numerik	Mean corpuscular volume (MCV) yang merupakan rata-rata volume sel darah pasien
9	MCHC	Numerik	Mean corpuscular hemoglobin concentration (MCHC) yang merupakan rata-rata konsentrasi hemoglobin sel darah pasien
10	RDW	Numerik	Red cell distribution width (RDW) yang merupakan lebarnya distribusi sel darah merah pasien
11	PLT	Numerik	Platelet (PLT) yang merupakan kadar trombosit dari pasien

p-ISSN: 2986-3929
e-ISSN: 3032-1948

No	Fitur	Tipe	Keterangan
12	Class	Numerik	Terdiri dari dua buah kelas yaitu 0 untuk pasien normal dan 1 untuk pasien dengan pembawa penyakit beta-thalassemia

2.2. Preprocessing Data

Tahapan preprocessing data merupakan langkah utama yang dilakukan untuk melakukan pengolahan data agar dapat memaksimalkan proses tahap selanjutnya [8]. Adapun tahapan dari preprocessing data yang dilakukan pada penelitian ini ditunjukkan oleh Gambar 1.

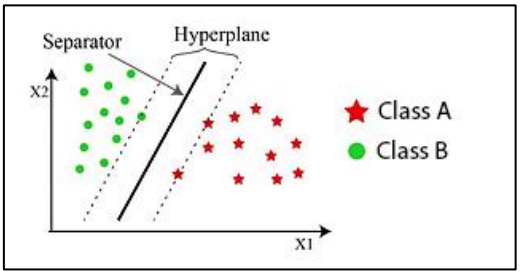


Gambar 1. Alur Tahapan Preprocessing Data

Dalam preprocessing data, tahapan pertama yang dilakukan adalah menganalisis distribusi data. Tahapan tersebut merupakan proses menganalisa seberapa seimbang kelas pada dataset yang digunakan untuk meminimalisir terjadinya kesetimpangan data antar kelas [9]. Tahapan selanjutnya adalah menganalisis korelasi fitur yang bisa dilakukan dengan mengamati korelasi setiap fitur melalui matrix correlation. Tahapan analisis korelasi fitur dapat menentukan fitur yang layak untuk dilakukan proses selanjutnya. Hal tersebut dapat meminimalisir kinerja dari model yang dibangun terhadap dataset yang cukup besar. Langkah terakhir pada preprocessing data yakni melakukan normalisasi data, dimana nilai data disamakan pada skala tertentu untuk dapat menghasilkan nilai akhir yang lebih optimal saat membangun pemodelan.

2.3. Membangun Model

Penelitian ini menggunakan algoritma Support Vector Machine (SVM) sebagai metode klasifikasi. SVM bertujuan untuk menemukan hyperplane atau sebuah bidang pemisah dengan memaksimalkan margin antara kelas yang berbeda dalam data [10]. SVM memiliki keunggulan dalam menciptakan hyperlane yang tidak linear, yang dapat dimanfaatkan untuk meningkatkan jarak margin antara kelas [11]. SVM bekerja dengan memetakan data ke dalam ruang fitur berdimensi tinggi untuk mengategorikan titik-titik data. Meskipun data tidak dapat dipisahkan secara linear, pembatas atau pemisah antara kategori dapat ditemukan, seperti yang ditunjukkan dalam Gambar 2 [7].



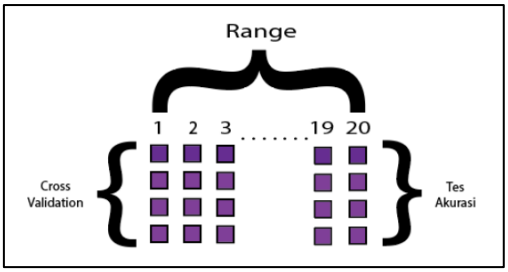
Gambar 2. Ilustrasi algoritma Support Vector Machine (SVM)

Pembangunan model dilakukan dengan menggunakan data yang sudah melalui tahap preprocessing. Data yang digunakan untuk melakukan pemodelan akan dibagi menjadi 80%

untuk data latih dan 20% untuk data uji. Pemodelan pada penelitian ini menggunakan algoritma Support Vector Machine (SVM) dengan mengimplementasikan bahasa pemrograman Python, serta untuk mendukung penerapan algoritma SVM tersebut, penelitian ini memanfaatkan library dari sklearn.

2.4. Hyperparameter Tuning

Hyperparameter tuning adalah tahapan penyesuaian parameter untuk mendapatkan nilai terbaik. Tahapan tersebut diperlukan dalam penelitian ini untuk memaksimalkan akurasi model yang sudah dibangun sebelumnya. Pada tahap hyperparameter tuning, penelitian ini memanfaatkan library dari GridSearchCV. GridSearchCV merupakan sebuah bagian dalam modul scikit-learn yang memungkinkan validasi untuk beberapa model secara efisien dengan memetakan berbagai nilai hyperparameter ke dalam suatu grid dan mengevaluasi kinerja model menggunakan teknik cross-validation [12]. Cross-validation adalah sebuah teknik pengembangan dari validasi split di mana validasi dilakukan dengan membagi data menjadi beberapa subset, kemudian dilakukan iterasi pengujian model dengan menggunakan setiap subset sebagai data uji secara bergantian seperti pada Gambar 3 [13].



Gambar 3. Cross-validation GridSearchCV

Dalam algoritma Support Vector Machine SVM, parameter-parameter yang disesuaikan (tuned) meliputi “C” (parameter regularisasi), “kernel” (jenis kernel), “gamma” (parameter kernel untuk kernel non-linear), dan “degree” (derajat polinomial untuk kernel polinomial). Pada penelitian ini, peneliti menggunakan parameter “kernel” untuk menjadi acuan pencarian parameter terbaik. Hal tersebut dikarenakan didalam library sklearn, setiap “kernel” memiliki parameter yang mempengaruhi untuk pembuatan model yang mempengaruhi hasil klasifikasi model. Parameter default atau bawaan dari algoritma SVM yaitu “’C’:1, ‘kernel’:’rbf’, ‘gamma’:’scale’”. Penggunaan GridSearchCV dalam penelitian ini bertujuan untuk menemukan kombinasi nilai-nilai parameter yang menghasilkan model SVM dengan performa optimal untuk tugas klasifikasi yang diberikan [14].

2.5. Analisis Hasil Pemodelan

Evaluasi dilakukan dengan membandingkan hasil tuning hyperparameter dengan hasil tanpa melakukan hyperparameter tuning ke dalam model yang sudah dibuat. Pada tahap evaluasi, penelitian ini menggunakan confusion matrix untuk memperoleh beberapa hasil klasifikasi yakni berupa nilai akurasi, presisi, recall, dan f1-score. Confusion matrix akan menggunakan beberapa nilai untuk mendapatkan hasil klasifikasi yaitu true negative (TN) dan true positive (TP) untuk menyatakan bahwa model berhasil melakukan klasifikasi, sementara false negative (FN) dan false positive (FP) untuk menyatakan bahwa model salah dalam melakukan klasifikasi. Untuk menghitung hasil klasifikasi yang akan diperoleh, dapat menggunakan rumus berikut [15].

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \quad (1)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (2)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (3)$$

p-ISSN: 2986-3929
e-ISSN: 3032-1948

$$\text{F1} - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

3. Hasil dan Pembahasan

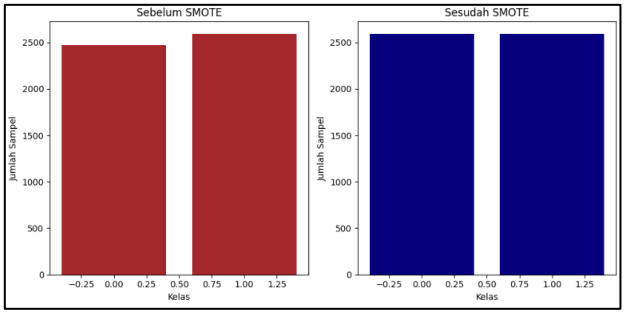
3.1. Preprocessing Data

a. Analisis Distribusi Data

Tahapan pertama dalam preprocessing data yakni melakukan analisis distribusi data untuk mengetahui seberapa seimbang kelas dalam dataset yang akan digunakan. Dalam dataset yang digunakan, dari total 5066 data, terdapat selisih 122 data antara dua kelas. Nilai tersebut memiliki perbedaan yang tidak terlalu signifikan, namun pada penelitian ini akan tetap menggunakan metode Synthetic Minority Oversampling Technique (SMOTE) untuk mendapatkan hasil akurasi model yang optimal. Penggunan metode SMOTE dapat dilihat pada Gambar 4 dengan memanfaatkan library dari imblearn. Sementara hasil perbandingan distribusi data sebelum dan sesudah oversampling dapat dilihat pada Gambar 5.

```
[13] from imblearn.over_sampling import SMOTE
     sm = SMOTE(random_state=0)
     #sampling smote
     x_sampling,y_sampling = sm.fit_resample(x,y)
```

Gambar 4. Implementasi Metode SMOTE pada Oversampling

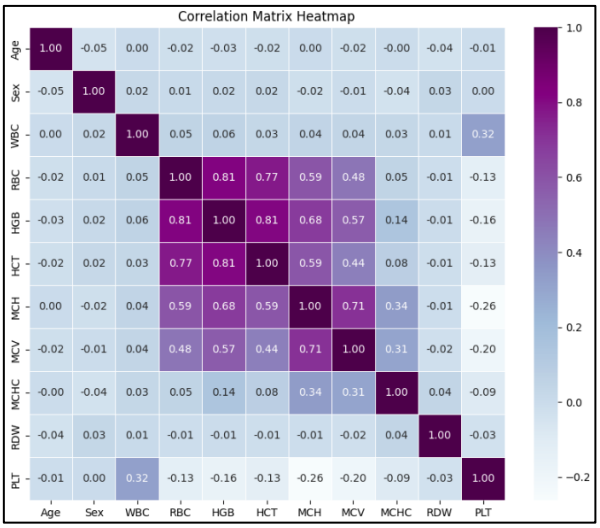


Gambar 5. Hasil Perbandingan Distribusi Data Sebelum dan Sesudah Oversampling

b. Analisis Korelasi Fitur

Menganalisis korelasi fitur pada dataset yang sudah melakukan oversampling digunakan untuk melihat seberapa erat hubungan antara setiap pasangan fitur. Analisis korelasi fitur yang dilakukan pada penelitian ini menggunakan correlation matrix heatmap dengan memanfaatkan library dari seaborn, pandas, dan pyplot. Correlation matrix heatmap sebagai analisis korelasi fitur dapat dilihat pada Gambar 6.

p-ISSN: 2986-3929
e-ISSN: 3032-1948



Gambar 6. Correlation Matrix Heatmap Terhadap Fitur Dataset

Terlihat bahwa beberapa fitur tidak terlalu berkorelasi dengan fitur lainnya seperti “Age”, “Sex”, “WBC”, “RDW”, dan “PLT” maka daripada itu penelitian ini memutuskan untuk membuang fitur tersebut dikarenakan untuk meminimalisir kinerja mesin saat digunakan untuk dataset yang besar. Implementasi pembuangan fitur pada dataset dapat dilihat pada Gambar 7.

```
[26] X = df_scaled.drop(['Class', 'Age', 'Sex', 'WBC', 'RDW', 'PLT'],axis=1)
     y = df_scaled['Class']
```

Gambar 7. Implementasi Pembuangan Fitur pada Dataset

c. Normalisasi Data

Tahapan normalisasi data dilakukan untuk menyelaraskan nilai-nilai dari fitur, dengan menggunakan dataset yang sudah melalui tahap seleksi fitur. Dalam penelitian ini, tahapan normalisasi menggunakan fungsi “minmax” seperti yang ditunjukkan pada Gambar 8.

```
[305] def minmax(df_input):
        list_fitur = df_input.columns[:-1]
        for fitur in list_fitur:
          max = df_input[fitur].max()
          min = df_input[fitur].min()
          df_input[fitur] = (df_input[fitur]-min)/(max-min)
        return df_input
```

Gambar 8. Fungsi “minmax” pada Normalisasi Data

3.2. Model dan Hyperparameter Tuning

Pada penelitian ini, pembangunan model Support Vector Machine (SVM) dibangun dengan memanfaatkan library dari sklearn yang ditunjukkan pada Gambar 9. Tahapan pembuatan model, dilakukan dengan memanfaatkan dataset yang sudah melalui proses oversampling. Dataset yang digunakan dibagi menjadi 80% dataset uji dan 20% dataset latih.

p-ISSN: 2986-3929
e-ISSN: 3032-1948

```
[18] from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x_sampling,y_sampling, test_size=0.20, random_state=0)


     from sklearn.svm import SVC
     # Proses Algoritma
     svm= SVC()
     svm.fit(x_train,y_train)
```

Gambar 9. Implementasi Model Algoritma Support Vector Machine (SVM)

Tahapan hyperparameter tuning pada penelitian ini akan menyesuaikan hyperparameter dari model yang dibangun yaitu Support Vector Machine (SVM) berdasarkan parameter kernel. Penelitian ini memanfaatkan library sklearn yaitu GridSearchCV untuk menentukan hyperparameter terbaik, dimana nilai parameter berdasarkan kernel yang diuji ditunjukkan pada Tabel 2. Implementasi tahapan tuning ditunjukkan pada Gambar 10.

Tabel 2. Nilai Parameter untuk Hyperparameter Tuning Berdasarkan Kernel

No	Kernel	Parameter dan Nilai Uji
1	‘linear’	'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]
2	‘rbf'	'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000] 'gamma': [0.001, 0.01, 0.1, 1, 10, 100]
3	‘poly’	'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000] 'gamma': [0.001, 0.01, 0.1, 1, 10, 100] 'degree': [2, 3, 4, 5]

```
from sklearn.model_selection import GridSearchCV

# hyperparameter kernel yang akan diuji
parameters = {
    # masukan setiap nilai parameter kernel yang ingin diuji
}

svm_model = svm.SVC()
svm_grid = GridSearchCV(estimator=svm_model, param_grid=parameters, cv=3, verbose=2, n_jobs=-1)
svm_grid.fit(x_train, y_train)
```

Gambar 10. Implementasi GridSearchCV untuk Hyperparameter Tuning

Dalam proses hyperparameter tuning pada setiap nilai parameter uji yang ditunjukkan pada Tabel 2, didapatkan hasil lima kombinasi parameter terbaik setiap kernel ditunjukkan pada Tabel 3 untuk kernel “linear”, Tabel 4 untuk kernel “rbf”, dan Tabel 5 untuk kernel “poly”.

Tabel 3. Hasil Parameter Terbaik menggunakan Kernel “linear”

mean_fit_time	mean_score_time	params	mean_test_score
0.172570388	0.032081842	{'C': 100, 'kernel': 'linear'}	0.946747214
0.783475955	0.046331565	{'C': 1000, 'kernel': 'linear'}	0.946747214
0.098690748	0.032089551	{'C': 10, 'kernel': 'linear'}	0.945783126
0.081038952	0.037542741	{'C': 1, 'kernel': 'linear'}	0.940963732
0.105776548	0.04589661	{'C': 0.1, 'kernel': 'linear'}	0.92722792

p-ISSN: 2986-3929
e-ISSN: 3032-1948

Tabel 4. Hasil Parameter Terbaik menggunakan Kernel “rbf”

mean_fit_time	mean_score_time	params	mean_test_score
0.343053341	0.209666332	{'C': 1, 'gamma': 100, 'kernel': 'rbf'}	0.993494149
0.050074657	0.021288713	{'C': 100, 'gamma': 10, 'kernel': 'rbf'}	0.993493453
0.052993377	0.02069219	{'C': 1000, 'gamma': 10, 'kernel': 'rbf'}	0.99180717
0.043960174	0.025547822	{'C': 10, 'gamma': 10, 'kernel': 'rbf'}	0.991806996
0.141822259	0.08481582	{'C': 1000, 'gamma': 100, 'kernel': 'rbf'}	0.991566496

Tabel 5. Hasil Parameter Terbaik menggunakan Kernel “poly”

mean_fit_time	mean_score_time	params	mean_test_score
108.3636426	0.011723359	{'C': 1000, 'degree': 5, 'gamma': 10, 'kernel': 'poly'}	0.99060206
108.7903892	0.016993364	{'C': 10, 'degree': 5, 'gamma': 10, 'kernel': 'poly'}	0.99060206
108.7040025	0.017054876	{'C': 100, 'degree': 5, 'gamma': 10, 'kernel': 'poly'}	0.99060206
84.76238942	0.011989037	{'C': 1, 'degree': 5, 'gamma': 10, 'kernel': 'poly'}	0.990361735
4.198003848	0.016676346	{'C': 1000, 'degree': 5, 'gamma': 1, 'kernel': 'poly'}	0.989878994

Pada hyperparameter tuning, didapatkan parameter terbaik yaitu “'C': 1, 'gamma': 100, 'kernel': 'rbf'” dengan skor uji rata-rata 0.993494149. Parameter terbaik tersebut akan dimasukan kedalam model untuk mencari hasil klasifikasi setelah mengalami proses hyperparameter tuning. Implementasi ke dalam pemodelan, ditunjukkan pada Gambar 11.

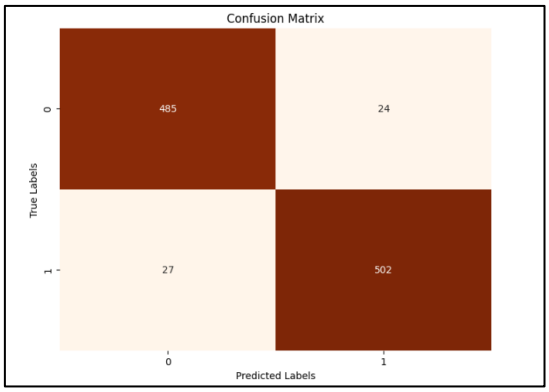
```
# Masukan hypermarameter terbaik
best_ModelSVM = SVC(C=1, gamma = 100, kernel='rbf')
best_ModelSVM.fit(x_train, y_train)

y_pred = best_ModelSVM.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='macro')
recall = recall_score(y_test, y_pred, average='macro')
f1_score = f1_score(y_test, y_pred, average='macro')
```

Gambar 11. Implementasi Pemodelan Support Vector Machine dan Hasil Klasifikasi

3.3. Evaluasi Model

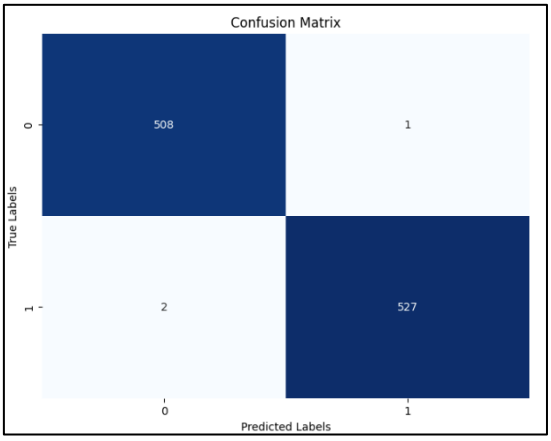
Hasil klasifikasi model sebelum hyperparameter tuning dan sesudah melalui tahap hyperparameter tuning, akan dievaluasi dengan menggunakan confusion matrix. Untuk hasil klasifikasi tanpa melakukan hyperparameter tuning ditunjukkan pada Gambar 12.

p-ISSN: 2986-3929
e-ISSN: 3032-1948



Gambar 12. Hasil Pemodelan Tanpa Menggunakan Hyperparameter Tuning

Gambar 12 menunjukan bahwa masih terdapat prediksi yang salah terhadap kelas pada dataset yang digunakan. Terdiri dari 24 data yang masih salah dalam prediksi kelas yang seharusnya bukan merupakan pembawa beta-thalassemia namun diprediksi pembawa beta-thalassemia, serta 27 data masih salah dalam prediksi kelas yang seharusnya merupakan pembawa beta-thalassemia namun diprediksi bukan pembawa thalassemia. Untuk hasil klasifikasi disertai penggunaan hyperparameter tuning ditunjukkan pada Gambar 13.



Gambar 13. Hasil Pemodelan Disertai Penggunaan Hyperparameter Tuning

Gambar 13 menunjukan bahwa masih terdapat prediksi yang salah terhadap kelas pada dataset yang digunakan, namun sudah berkurang jauh dari prediksi yang ditunjukkan pada Gambar 12. Terdiri dari 1 data yang masih salah dalam prediksi kelas yang seharusnya bukan merupakan pembawa beta-thalassemia namun diprediksi pembawa beta-thalassemia, serta 2 data masih salah dalam prediksi kelas yang seharusnya merupakan pembawa beta-thalassemia namun diprediksi bukan pembawa thalassemia. Peningkatan prediksi tersebut membuat perbedaan hasil klasifikasi yang ditunjukkan pada Tabel 6.

Tabel 6. Perbandingan Hasil Akhir Pemodelan

	Akurasi	**Presisi**	**Recall**	**F1-score**
Sebelum Hyperparameter Tuning	95,08%	95,43%	94,89%	95,16%

Setelah Hyperparameter Tuning	99,71%	99,81%	99,62%	99,71%

Berdasarkan Tabel 6, hasil akurasi, presisi, recall, f1-score yang diperoleh sebelum melakukan hyperparameter tuning secara berturut-turut yaitu 95,08%, 95,43%, 94,89%, dan 95,16%. Sementara hasil akurasi, presisi, recall, f1-score yang diperoleh setelah melakukan hyperparameter tuning secara berturut-turut yaitu 99,71%, 99,81%, 99,62%, dan 99,71%. Hal tersebut secara keseluruhan membuktikan bahwa hyperparameter tuning berpengaruh dalam pemodelan menggunakan algoritma Support Vector Machine (SVM) yang mengalami peningkatan hasil akurasi, presisi, recall, f1-score yang diperoleh.

4. Kesimpulan

Penerapan algoritma Support Vector Machine (SVM) pada klasifikasi penyakit beta-thalassemia dapat disimpulkan bahwa optimasi hyperparameter dapat meningkatkan hasil klasifikasi yang diperoleh. Hyperparameter tuning yang dilakukan memanfaatkan GridSearchCV untuk menentukan parameter terbaik sesuai dengan nilai yang ingin diujikan. Hasil terbaik yang didapatkan setelah melakukan hyperparameter tuning yaitu nilai akurasi sebesar 99,71%, nilai presisi sebesar 99,81%, nilai recall sebesar 99,62%, dan nilai f1-score sebesar 99,71% dengan parameter terbaiknya yaitu “'C': 1, 'gamma': 100, 'kernel': 'rbf'” dengan skor uji rata-rata 0.993494149. Untuk hasil klasifikasi yang lebih baik, perlu diadakannya tahap hyperparameter tuning lebih lanjut untuk memaksimalkan kinerja dari model yang dibangun serta pengujian dengan algoritma lain untuk membandingkan performa berdasarkan hasil akhir klasifikasi.

Daftar Pustaka

[1] D. Setiawan, H. Setiawan, and A. Nurmalasari, “Edukasi Penyakit Thalasemia Pada Mahasiswa Stikes Muhammadiyah Ciamis,” SELAPARANG: Jurnal Pengabdian Masyarakat Berkemajuan, vol. 6, no. 3, pp. 1098-1102, 2022.

[2] R. Roslaeni, S. Ratunanda, and A. L. Susanti, “Skrining Talasemia Bagi Mahasiswa Fakultas Kedokteran Universitas Jenderal Achmad Yani,” Jurnal Abdimas Kartika Wijayakusuma, vol. 5, no. 1, pp.168-173, 2024. doi: 10.26874/jakw. v5i1.369.

[3] M. N. Praramdana, M. A. Rusydi, and M. Rizky, “Sebuah Tinjauan Pustaka: Penatalaksanaan Beta Thalasemia,” Jurnal Medika Hutama, vol. 4, no. 2, pp. 3257-3264, 2023.

[4] W. Chauhan, S. Shoaib, R. Fatma, Z. Zaka-ur-Rab, and M. Afzal, “Beta-Thalassemia and the Advent of New Interventions Beyond Transfusion and Iron Chelation,” British Journal of Clinical Pharmacology, vol. 88, no. 8, pp. 3610–3626, 2022. doi: 10.1111/bcp.15343.

[5] D. S. Rahayu, Nursafika, J. Afifah, and S. Intan, “Classification of Diabetes Mellitus Using C4.5 Algorithm, Support Vector Machine (SVM) and Linear Regression,” SENTIMAS: Seminar Nasional Penelitian dan Pengabdian Masyarakat, vol. 1, no. 1, pp. 56-63, 2023.

[6] Mardewi, N. Yarkuran, Sofyan, and F. Aziz, “Klasifikasi Kategori Obat Menggunakan Algoritma Support Vector Machine,” Journal Pharmacy and Application of Computer Sciences, vol. 1, no. 1, pp. 27-32, 2023.

[7] Yennimar, A. Rasid, and S. Kenedy, “Implementation of Support Vector Machine Algorithm with Hyper-Tuning Randomized Search In Stroke Prediction,” Journal of Information Systems and Computer Science Prima, vol. 6, no. 2, pp. 61-65, 2023.

[8] Yuyun, N. Hidayah, and S. Sahibu, “Algoritma Multinomial Naïve Bayes Untuk Klasifikasi Sentimen Pemerintah Terhadap Penanganan Covid-19 Menggunakan Data Twitter,” Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 5, no. 4, pp. 820–826, 2021, doi: 10.29207/resti. v5i4.3146.

[9] H. Hairani, K. E. Saputro, and S. Fadli, “K-means-SMOTE for handling class imbalance in the classification of diabetes with C4.5, SVM, and naive Bayes,” Jurnal Teknologi dan Sistem Komputer, vol. 8, no. 2, pp. 89–93, 2020, doi: 10.14710/jtsiskom.8.2.2020.89-93.

[10] H. S. Wafa, A. Id Hadiana, and F. R. Umbara, “Prediksi Penyakit Diabetes Menggunakan Algoritma Support Vector Machine (SVM),” Informatics And Digital Expert (INDEX), vol. 4, no. 1, pp. 40-45, 2022.
[11] P. W. A. Wibawa and C. Pramartha, “Systematic Literature Review: Machine Learning Methods in Emotion Classification in Textual Data,” Jurnal Sisfokom (Sistem Informasi dan Komputer), vol. 12, no. 3, pp. 425–433, 2023, doi: 10.32736/sisfokom.v12i3.1787.
[12] Z. M. E. Darmawan and A. F. Dianta, “Implementasi Optimasi Hyperparameter GridSearchCV Pada Sistem Prediksi Serangan Jantung Menggunakan SVM,” Teknologi: Jurnal Ilmiah Sistem Informasi, vol. 13, no. 1, pp. 8–15, 2023, doi: 10.26594/teknologi.v13i1.3098.
[13] S. T. Kusuma and T. B. Sasongko, “Optimasi K-Nearest Neighbor dengan Grid Search CV pada Prediksi Kanker Paru-Paru,” Indonesian Journal of Computer Science Attribution, vol. 12, no. 4, pp. 2162-2171, 2023.
[14] M. Fajri and A. Primajaya, “Komparasi Teknik Hyperparameter Optimization pada SVM untuk Permasalahan Klasifikasi dengan Menggunakan Grid Search dan Random Search,” Journal of Applied Informatics and Computing (JAIC), vol. 7, no. 1, pp. 10-15, 2023.
[15] R. Yunita and M. Kamayani, “Perbandingan Algoritma SVM Dan Naïve Bayes Pada Analisis Sentimen Kebijakan Penghapusan Kewajiban Skripsi,” Indonesian Journal of Computer Science, vol. 12, no.5, pp. 2879-2890, 2023.

p-ISSN: 2986-3929
e-ISSN: 3032-1948

Halaman ini sengaja dibiarkan kosong