

IDENTIFIKASI GAGAL GINJAL KRONIS DENGAN MENGIMPLEMENTASIKAN METODE SUPPORT VECTOR MACHINE BESERTA K-NEAREST NEIGHBOUR (SVM-KNN)

Alvin Tarisa Akbar^{*1}, Novanto Yudistira², Achmad Ridok³

^{1,2,3}Universitas Brawijaya, Malang

Email: ¹alvintarisaakba@student.ub.ac.id, ²yudistira@ub.ac.id, ³acridokb@ub.ac.id

^{*}Penulis Korespondensi

(Naskah masuk: 13 Januari 2022, diterima untuk diterbitkan: 11 April 2023)

Abstrak

Ginjal merupakan bagian vital bagi manusia karena berfungsi untuk menyaring atau membersihkan cairan yang kita minum agar dapat dikonsumsi oleh tubuh secara normal. Gagal ginjal adalah situasi dimana ginjal mengalami penurunan fungsionalnya secara terus-menerus yang mana dapat mengakibatkan ketidakmampuan ginjal untuk berfungsi untuk semestinya. Untuk membantu pasien yang terjangkit penyakit gagal ginjal kronis hal yang terlebih dahulu dilakukan adalah mengidentifikasi penyakit tersebut. Identifikasi gagal ginjal kronis dengan menggunakan *dataset* yang dibuat oleh L.Jerlin Rubini dkk. sudah dilakukan dengan berbagai metode klasifikasi, contohnya adalah implementasi metode klasifikasi Support Vector Machine (SVM) dan K-Nearest Neighbour (KNN). Salah satu kelemahan dari SVM adalah bila data terlalu dekat dengan *hyperplane* adanya potensi untuk salah mengklasifikasi. Lalu salah satu kelemahan dari KNN adalah berpotensi mengalami penurunan akurasi bila nilai k terlalu tinggi atau terlalu rendah yang mana masing-masing mengakibatkan banyaknya *noise* data atau terlalu kecil data yang digunakan sebagai pembandingan. Untuk penelitian ini, kami mengimplementasikan penggabungan metode SVM dengan KNN yang dikenal dengan SVM-KNN yang menggunakan optimasi *Simplified Sequential Minimal Optimization (Simplified SMO)*. Metode ini mencoba untuk menutupi kelemahan dari SVM dan KNN. Penelitian ini melakukan percobaan pada beberapa nilai parameter yang digunakan untuk mendapatkan akurasi pada metode klasifikasi SVM-KNN terbaik. Parameter yang diuji adalah *cost*, *tolerance*, *gamma*, dan *bias* pada metode SVM, parameter k pada metode KNN, serta parameter *miu* pada metode SVM-KNN. Nilai rata-rata akurasi terbaik didapatkan dengan menggunakan SVM-KNN dengan nilai 94,25% dan terbukti lebih baik dari pada SVM dengan 94,09% dan KNN dengan 91,73%.

Kata kunci: Metode Klasifikasi, Support Vector Machine, K-Nearest Neighbour, SVM-KNN

IDENTIFICATION OF CHRONIC KIDNEY DISEASE BY IMPLEMENTING THE SUPPORT VECTOR MACHINE METHOD AND K-NEAREST NEIGHBOR (SVM-KNN)

Abstract

Kidneys are a vital part for humans because they function to filter or clean the fluids we ingest so that they can be consumed safely. Kidney failure is a situation where the kidneys experience a continuous decline in function which can result in the inability of the kidneys to function properly. To help patients with chronic kidney failure, the first thing to do is to identify the disease. Identification of chronic kidney failure using the dataset created by L.Jerlin Rubini et. al. had been tested with various classification methods, for example the implementation of the Support Vector Machine (SVM) and K-Nearest Neighbor (KNN). One of the weaknesses of SVM is that if the data is too close to the hyperplane there is the potential for misclassification. Then one of the weaknesses of KNN is that it has the potential to experience a decrease in accuracy if the value of k is too high or too low which results in a lot of noise data or too little data used as a comparison respectively. For this research, we implemented a hybrid of SVM with KNN known as SVM-KNN which was optimized using Simplified Sequential Minimal Optimization (Simplified SMO). This study conducted experiments on several parameter values used to obtain the best accuracy in SVM-KNN. The parameters tested are *cost*, *tolerance*, *gamma*, *bias* on SVM, parameter k on KNN, and *miu* on SVM-KNN. The average value of accuracy was obtained using SVM-KNN with 94.25% and proved better than SVM with 94.09% and KNN with 91.73%.

Keywords: Classification Method, Support Vector Machine, K-Nearest Neighbour, SVM-KNN

1. PENDAHULUAN

Ginjal merupakan bagian vital bagi manusia karena berfungsi untuk menyaring atau membersihkan cairan yang kita minum agar dapat dikonsumsi oleh tubuh secara normal. Bila Ginjal kita mengalami gangguan maka ginjal tidak bisa untuk menyediakan asupan. Bila gangguan ginjal ini tidak bisa diidentifikasi secara cepat maka tubuh manusia tersebut akan rusak karena air yang digunakan untuk metabolisme masih belum dibersihkan dan dapat mengakibatkan berbagai macam penyakit. Salah satu gangguan dari ginjal adalah gagal ginjal.

Gagal ginjal adalah situasi dimana ginjal mengalami penurunan fungsionalnya secara terus-menerus yang mana dapat mengakibatkan ketidakmampuan ginjal untuk berfungsi untuk semestinya. Bahkan bila dibiarkan masuk kedalam kondisi kronis akan menghentikan kinerja ginjal selamanya. Gagal ginjal kronis (*Chronic Kidney Disease*) meningkat dengan cepat. Sebesar 10% dari penduduk dunia tercatat oleh penyakit ini pada tahun 2015 (Ahmad, dkk., 2017).

Untuk mengidentifikasi gagal ginjal kronis terdapat berbagai cara. Mengidentifikasi menggunakan metode klasifikasi adalah salah satunya. Contoh dari metode klasifikasi adalah metode *K-Nearest Neighbour* (K-NN) dan *Support Vector Machine* (SVM).

Penelitian yang dilakukan oleh Desai (2019) untuk mengidentifikasi apakah seseorang mengidap gagal ginjal kronis menggunakan algoritma *Random Forest* dengan menggunakan algoritma *Boruta*. Penelitian ini menggunakan data set dari *UCI Machine Learning Repository*. Pada penelitian ini metode tersebut mendapatkan akurasi sebesar 99,19% (Desai, 2019).

Pada penelitian yang dilakukan oleh Pujiyanto, dkk. (2018) untuk mengidentifikasi gagal ginjal kronis. Penelitian ini menggunakan metode penggabungan metode *K-Nearest Neighbour* (K-NN) dan *Support Vector Machine* (SVM). Perbedaan dengan metode yang digunakan pada penelitian ini adalah pada penelitian dari Pujiyanto, dkk. dilakukan pengelompokan menggunakan K-NN terlebih dahulu selanjutnya dimasukkan ke metode SVM. Sedangkan pada penelitian ini menggunakan SVM terdahulu selanjutnya dimasukkan ke metode K-NN. Hasil dari penelitian Pujiyanto dkk. menghasilkan akurasi tertinggi 100% (Pujiyanto, dkk., 2018).

Pada penelitian yang dilakukan oleh Charleonnann, dkk. (2016) untuk mengidentifikasi gagal ginjal kronis. Penelitian ini menggunakan metode *K-Nearest Neighbour* (KNN), *Support Vector Machine* (SVM), *Logistic Regression* (LR), dan *Decision Tree Classifiers*. Hasil dari penelitian ini adalah sistem-sistem yang mana menggunakan algoritma-algoritma KNN, SVM, LR, dan *Decision Tree Classifiers* dengan nilai akurasi masing-masing

bernilai 98,1%, 98,3%, 96,55%, dan 94,8% (Charleonnann, dkk., 2016).

Pada penelitian yang dilakukan oleh Ahmad, dkk. (2017) untuk menciptakan algoritma untuk mengidentifikasi gagal ginjal kronis menggunakan metode SVM. Pada penelitian ini beliau-beliau mendapatkan akurasi 98,34% (Ahmad, dkk., 2017).

Pada penelitian yang dilakukan oleh Devika, dkk. (2019) untuk mengembangkan program dengan tiga metode, yakni *Naïve Bayes*, KNN, *Random Forest*. Pada akhir penelitian ini didapatkan akurasi dari masing-masing metode dengan akurasi 99,63%, 87,78%, dan 99,84% yang mana *Random Forest* menghasilkan akurasi tertinggi (Devika, dkk., 2019).

Pada penelitian yang dilakukan oleh Ming, dkk. (2003) pernah melakukan percobaan penggabungan metode klasifikasi SVM dan KNN. Akurasi metode yang digunakan adalah 94,82% dibandingkan dengan hanya menggunakan metode *Support Vector Machine* dengan akurasi sebesar 92,17% (Ming, dkk., 2003).

Berdasarkan penelitian terdahulu, nilai dari akurasi SVM sudah cukup bagus. Kami ingin mencoba melakukan penelitian dengan menggunakan metode yang digunakan oleh Ming, dkk. yang mana metode SVM-KNN memiliki nilai akurasi lebih tinggi dibandingkan dengan hanya menggunakan metode SVM. Oleh karena itu kami melakukan penelitian mengidentifikasi gagal ginjal kronis adalah SVM dengan menggabungkan KNN.

2. METODE PENELITIAN

Penelitian ini menggunakan penggabungan metode klasifikasi SVM dengan KNN.

2.1. Sumber Data

Data yang digunakan untuk pada penelitian ini didapatkan dari *UCI Machine Learning Repository* yang dikumpulkan oleh Universitas Alagappa di India (Rubini, dkk., 2015). Dataset tersebut memiliki 400 data dengan 24 fitur. Pada penelitian ini data yang digunakan sebanyak 302 dengan enam fitur. Pemilihan dari 302 data tersebut dikarenakan banyak nilai dari atribut-atribut yang lain memiliki nilai *missing value*.

Pada dataset tersebut terdapat dua jenis kategori data, yakni numerik dan kategorik. Data kategorik akan dirubah menjadi data numerik. Contohnya nilai pada "*Present*" akan diubah menjadi nilai 1 dan nilai "*Not Present*" akan diubah menjadi nilai 0.

Dari 302 data dibagi 2 bagian, 181 data digunakan sebagai data *training* dan 121 data digunakan sebagai data *testing*. Pengujian pengaruh hyperparameter dilakukan pada data *training* dan dibagi menjadi 10 bagian.

2.1.1. Missing Value

Missing value menurut Fernando, dkk. (2021) adalah data yang mana diakui oleh penerbit dataset dilakukan guna menyembunyikan data sensitif atau ketidaklengkapan data yang diambil pada saat pengambilan data.

2.2. Normalisasi

Proses normalisasi bertujuan untuk meningkatkan akurasi dan menurunkan waktu yang dibutuhkan untuk mengkomputasi (Raju, dkk., 2020). Pada penelitian ini jenis normalisasi yang digunakan adalah normalisasi Standar Scaler atau *z-score*. Pada persamaan (1) x adalah nilai dari data tersebut, \bar{x} adalah rata-rata dari fitur data, dan σ adalah nilai dari standar deviasi dari fitur data.

$$Z_{Scaled} = \frac{(x - \bar{x})}{\sigma} \quad (1)$$

2.3. Algoritma KNN

Nearest Neighbour (NN) adalah klasifikasi dengan menemukan solusi yang efisien dengan menggunakan perkiraan, atau deskriminasi non-parametrik pada statistik. *K-Nearest Neighbour* (KNN) adalah Klasifikasi dengan menggunakan nilai k sebagai objek yang ditemukan pada *dataset training* yang terdekat pada *dataset testing* (Zhang, 2021). Deskriminasi non-parametrik adalah setiap instansi disimpan pada tempat penyimpanan tersendiri, bukan sebuah nilai bobot pada suatu sistem (Wu, dkk., 2018). Pada penelitian ini, rumus yang digunakan adalah eucliden distance. Pada persamaan (2) x_i dan x_d adalah data, a_k adalah fitur pada data, k adalah indeks fitur.

$$d(x_i, x_d) = \sqrt{\sum_{k=1}^m (x_i(a_k) - x_d(a_k))^2} \quad (2)$$

2.4. Algoritma SVM

Support Vector Machine (SVM) adalah metode machine learning yang mengelompokkan data dengan kelasnya dengan menggunakan *hyperplane*. *Hyperplane* adalah bidang atau garis yang memisahkan data dan mengelompokkan data (Ghosh, dkk., 2019). Pada penelitian ini, jenis SVM yang digunakan adalah non-linier. Pada persamaan (3) α adalah *Lagrange Multiplier*, SV adalah data *train* yang nilai α tidak sama dengan 0, y adalah kelas data *train*, K adalah kernel, x_i adalah data *train*, x adalah data *test*, b adalah bias.

$$g(x) = \sum_{i \in SV} \alpha_i y_i K(x_i, x) + b \quad (3)$$

Lalu untuk mengelompokkan data tersebut menggunakan Persamaan (4) dimana $g(x)$ adalah hasil dari penghitungan SVM.

$$f(x_d) = \text{sign}(g(x)) \quad (4)$$

2.5. Algoritma Simplified SMO

Simplified Sequential Minimal Optimazation (*Simplified SMO*) adalah metode *kernel trick* pada SVM. *Kernel trick* adalah metode yang dapat meningkatkan akurasi dan mengurangi waktu komputasi yang dibutuhkan dari metode klasifikasi. *Simplified SMO* membantu proses training SVM dengan memberikan solusi *Quadric Problem* ke dalam bentuk yang lebih sederhana atau kecil. Pertama mencari nilai *error* pada setiap data. Pada persamaan (5) α adalah *Lagrange Multiplier*, x adalah data *train*, y adalah kelas data *train*, K adalah kernel, x_i adalah data *train*, x adalah data *test*, b adalah bias, serta i dan d adalah indeks dari data *train*.

$$\text{error}_i = \left(\left(\sum_{i=1}^N \alpha_i \cdot y_i \cdot K(x_i, x_d) \right) + b \right) - y_i \quad (5)$$

Berikutnya dilakukan pengecekan kondisi Karush-Kush-Tucker (KKT). Pengecekan Kondisi KKT dapat dilakukan dengan syarat nilai $\text{error}_i * \text{target}_i < -\text{tolerance}$ dan nilai $\alpha_i < \text{cost}$ atau nilai $\text{error}_i * \text{target}_i > \text{tolerance}$ dan nilai $\alpha_i > 0$. Bila kondisi KKT telah memenuhi maka akan melompati ke data selanjutnya, sedangkan jika kondisi KKT tidak dipenuhi maka menghitung nilai L dan H . Nilai L didapatkan menggunakan Persamaan (6) dan Nilai H didapatkan menggunakan Persamaan (7). Pada Persamaan (6) dan (7) α adalah *Lagrange Multiplier*, y adalah kelas data *train*, c adalah *cost*, i adalah indeks data *train*, dan j adalah indeks data *train* yang tidak sama dengan i .

$$L = \begin{cases} \max(0, \alpha_j - \alpha_i) & \text{jika } y_i \neq y_j \\ \max(0, \alpha_j + \alpha_i - c) & \text{jika } y_i = y_j \end{cases} \quad (6)$$

$$H = \begin{cases} \min(c, c + \alpha_j - \alpha_i) & \text{jika } y_i \neq y_j \\ \min(c, \alpha_j + \alpha_i) & \text{jika } y_i = y_j \end{cases} \quad (7)$$

Jika nilai L dan H adalah sama maka akan melompati ke data selanjutnya. Nilai η didapatkan menggunakan Persamaan (8). Pada Persamaan (8) K adalah *Kernel*, x adalah data *train*, i adalah indeks data *train*, dan j adalah indeks data *train* yang tidak sama dengan i .

$$\eta = 2K(x_i, x_j) - K(x_i, x_i) - K(x_j, x_j) \quad (8)$$

Jika nilai η lebih dari sama dengan 0 maka akan melompati ke data selanjutnya. Selanjutnya mencari nilai α_j^{baru} . Nilai α_j^{baru} didapatkan menggunakan Persamaan (9). Pada Persamaan (9) α_j^{baru} adalah *Lagrange Multiplier* baru pada indeks j , α_j^{lama} adalah *Lagrange Multiplier* lama pada

indeks j , y adalah kelas data *train*, *error* adalah nilai dari variabel *error* pada Peramaan (5), i adalah indeks data *train*, dan j adalah indeks data *train* yang tidak sama dengan i .

$$\alpha_j^{\text{baru}} = \alpha_j^{\text{lama}} - \frac{y_i(\text{error}_i - \text{error}_j)}{\eta} \quad (9)$$

Dengan menggunakan L dan H , didapatkan nilai $\alpha_j^{\text{baru, potongan}}$. Nilai $\alpha_j^{\text{baru, potongan}}$ didapatkan menggunakan Persamaan (10). Pada Persamaan (10) $\alpha_j^{\text{baru, potongan}}$ adalah *Lagrange Multiplier* baru yang berpotongan dengan L dan H pada indeks j , α_j^{baru} adalah *Lagrange Multiplier* baru pada indeks j , L dan H adalah nilai dari variabel L dan H pada Peramaan (6) dan (7), i adalah indeks data *train*, dan j adalah indeks data *train* yang tidak sama dengan i .

$$\alpha_j^{\text{baru, potongan}} = \begin{cases} H, & \text{jika } \alpha_j^{\text{baru}} > H \\ \alpha_j^{\text{baru}}, & \text{jika } L \leq \alpha_j^{\text{baru}} \leq H \\ L, & \text{jika } \alpha_j^{\text{baru}} < L \end{cases} \quad (10)$$

Selanjutnya mencari nilai α_i^{baru} . Nilai α_i^{baru} didapatkan menggunakan Persamaan (11). Pada Persamaan (11) α_i^{baru} adalah *Lagrange Multiplier* baru pada indeks i , α_i^{lama} adalah *Lagrange Multiplier* lama pada indeks i , y adalah kelas data *train*, $\alpha_j^{\text{baru, potongan}}$ adalah *Lagrange Multiplier* baru yang berpotongan dengan L dan H pada indeks j , i adalah indeks data *train*, dan j adalah indeks data *train* yang tidak sama dengan i .

$$\alpha_i^{\text{baru}} = \alpha_i^{\text{lama}} + y_i * y_j (\alpha_j^{\text{lama}} - \alpha_j^{\text{baru, potongan}}) \quad (11)$$

Selanjutnya mencari nilai b_1 dan b_2 . Nilai b_1 dan b_2 didapatkan menggunakan Persamaan (12) dan (13). Pada Persamaan (12) b_1^{lama} adalah *bias* lama, *error* adalah nilai *error* pada Persamaan (5), $\alpha_j^{\text{baru, potongan}}$ adalah *Lagrange Multiplier* baru yang berpotongan dengan L dan H pada indeks j , α_j^{lama} adalah *Lagrange Multiplier* lama pada indeks j , K adalah *kernel*, i adalah indeks data *train*, dan j adalah indeks data *train* yang tidak sama dengan i .

$$b_1 = b_1^{\text{lama}} - \text{error}_i - y_i (\alpha_i^{\text{baru}} - \alpha_i^{\text{lama}}) K(x_i, x_i) - y_j (\alpha_j^{\text{baru, potongan}} - \alpha_j^{\text{lama}}) K(x_i, x_j) \quad (12)$$

$$b_2 = b_1^{\text{lama}} - \text{error}_j - y_i (\alpha_i^{\text{baru}} - \alpha_i^{\text{lama}}) K(x_i, x_j) - y_j (\alpha_j^{\text{baru, potongan}} - \alpha_j^{\text{lama}}) K(x_j, x_j) \quad (13)$$

Dari nilai b_1 dan b_2 didapatkan nilai b dengan menggunakan Persamaan (11).

$$b = \begin{cases} b_1, & \text{jika } 0 < \alpha_i < c \\ b_2, & \text{jika } 0 < \alpha_j < c \\ \frac{(b_1 + b_2)}{2}, & \text{lainnya} \end{cases} \quad (14)$$

Hasil Simplified SMO adalah SV dan *bias* yang digunakan pada SVM. SV adalah α yang bernilai tidak sama dengan 0.

2.6. Kernel Radial Basis Fuction

Kernel trick yang digunakan adalah kernel *Radial Basis Fuction* (RBF). Kernel ini digunakan bila solusi menggunakan kernel *linear* tidak dapat dihasilkan. Selain itu akurasi dari kernel *non-linear* lebih baik daripada *linier* (Que dan Belkin, 2017). Pada persamaan (15) n adalah banyak fitur, l adalah fitur pada *dataset*, K adalah *kernel*, x_i adalah data *train*, x adalah data *test*. γ adalah nilai dari *gamma*.

$$K(x_i, x) = \prod_{l=1}^n \exp \left(-\frac{\|x_i^l - x^l\|}{\gamma} \right) \quad (15)$$

2.6. Algoritma SVM-KNN

Pada penelitian ini metode klasifikasi SVM-KNN menggunakan *kernel trick* dan metode optimasi *Simplified Sequential Minimal Optimization* (Simplified SMO). *Kernel trick* yang digunakan adalah kernel *Radial Basis Fuction* (RBF). Setelah mendapatkan nilai α dan *bias*, menghitung nilai SVM atau $g(x)$ dengan menggunakan Persamaan (3).

Bila nilai $g(x)$ lebih dari 1 maka data akan diklasifikasikan dengan SVM. Sedangkan bila nilai $g(x)$ kurang dari 1 maka akan menghitung jarak data *test* dengan setiap data *train*. Jarak tersebut dihitung dengan menggunakan Persamaan (2).

Lalu dari nilai-nilai jarak tersebut akan diurutkan dari terkecil sebanyak $\mu * k$. Pada tahap selanjutnya menghitung jarak tanpa diabsolutkan. Pada persamaan (16) x_i dan x_d adalah data, a_k adalah fitur pada data, k adalah indeks fitur.

$$(x - x^L) = \sum_{k=1}^m (x_i(a_k) - x_d(a_k)) \quad (16)$$

Selanjutnya adalah menghitung nilai M_+ dan M . Pada persamaan (17) dan (18) k^+ adalah banyak data dengan kelas positif, k adalah banyak data.

$$M_+(x) = \frac{1}{k^+} \sum_{L^+=1}^{k^+} (x^{L^+} - x) \quad (17)$$

$$M(x) = \frac{1}{k} \sum_{L=1}^{\mu k} (x^L - x) \quad (18)$$

Selanjutnya adalah menghitung nilai *delta* (∇) pada persamaan (19).

$$\nabla_l(x - x^L) = |M_+(x) - M(x)| \quad (19)$$

Dari hasil nilai-nilai ∇ tersebut diurutkan dari terkecil dan dilakukan proses voting.

2.7. Hyperparameter

Hyperparameter adalah parameter yang digunakan untuk *machine learning* pada proses *training* dan *testing*. Parameter ini digunakan karena memiliki efek yang besar pada kinerja dari *machine learning* yang mana krusial (Hoque dan Aljamaan, 2021). Hyperparameter pada SVM pada umumnya adalah *cost*, *tolerance*, *gamma*, dan *bias*. Lalu untuk KNN hyperparameter adalah *k*. Dan untuk SVM-KNN pada penelitian Ming, dkk. adalah *miu*.

3. HASIL DAN PEMBAHASAN

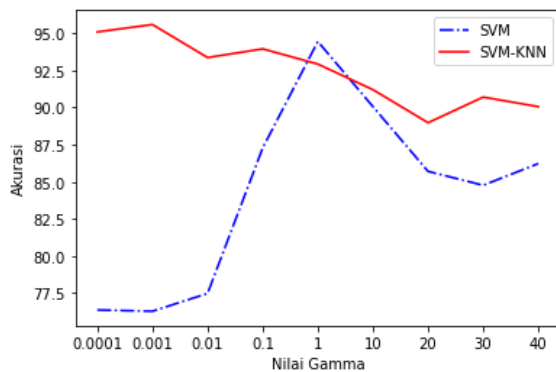
Pada awal percobaan nilai dari masing-masing hyperparameter ditentukan secara acak. Untuk nilai dari masing-masing hyperparameter dapat dilihat pada Tabel 1.

Tabel 1. Pengujian gamma

Hyperparameter	Nilai
cost (c)	1
Tolerance (tol)	0,0001
K	7
Miu (μ)	1
Gamma (γ)	20

3.1. Pengujian Gamma dengan Akurasi

Pengujian hyperparameter *gamma* (γ) dengan akurasi pada metode klasifikasi SVM dan SVM-KNN. Nilai γ digunakan pada pada pengujian tahap ini antara lain : 0,0001, 0,001, 0,01, 0,1, 1, 10, 20, 30, 40. Hasil dari pengujian dapat dilihat pada Gambar 1.

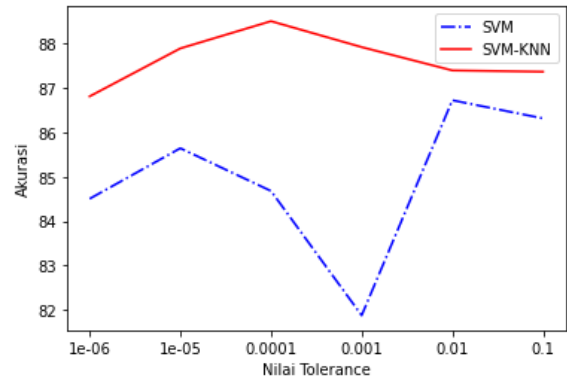


Gambar 1. Pengujian gamma

Pada proses pengujian pengaruh hyperparameter γ , hasil dari nilai akurasi pada pengujian dapat dilihat pada Gambar 1. Dari hasil pengujian tersebut γ memiliki pengaruh pada kinerja dari SVM dan SVM-KNN. Pengaruh γ adalah tebal atau tipisnya garis *hyperplane*. Bila nilai γ semakin kecil maka semakin tipisnya garis *hyperplane* dan sebaliknya (Al-Mejibli, Alwan, Abd, dan Hamed, 2020).

3.2. Pengujian Tolerance dengan Akurasi

Pengujian hyperparameter *tolerance* (tol) dengan akurasi pada metode klasifikasi SVM dan SVM-KNN. Nilai tol digunakan pada pada pengujian tahap ini antara lain : 0,1, 0,01, 0,001, 0,0001, 0,00001, 0,000001. Hasil dari pengujian dapat dilihat pada Gambar 2.

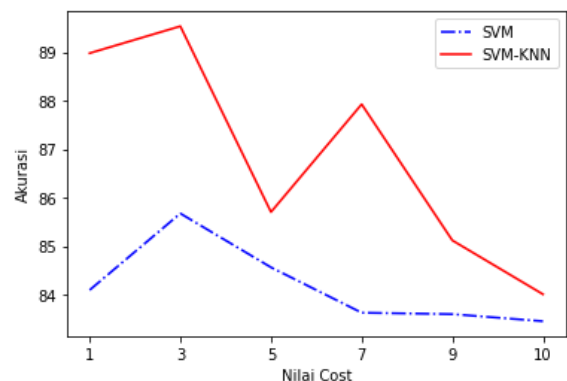


Gambar 2. Pengujian tolerance

Pada proses pengujian pengaruh hyperparameter tol, hasil dari nilai akurasi pada pengujian dapat dilihat pada Gambar 2. Dari hasil pengujian tersebut tol memiliki pengaruh pada kinerja dari SVM dan SVM-KNN. Pengaruh tol adalah salah satu syarat dari konvergensi. Bila nilai tol semakin kecil maka kecil perbedaan nilai saat proses penghitungan pada fungsi *simplified SMO*.

3.3. Pengujian Cost dengan Akurasi

Pengujian hyperparameter *cost* (c) dengan akurasi pada metode klasifikasi SVM dan SVM-KNN. Nilai c digunakan pada pada pengujian tahap ini antara lain : 1, 3, 5, 7, 9, 10. Hasil dari pengujian dapat dilihat pada Gambar 3.



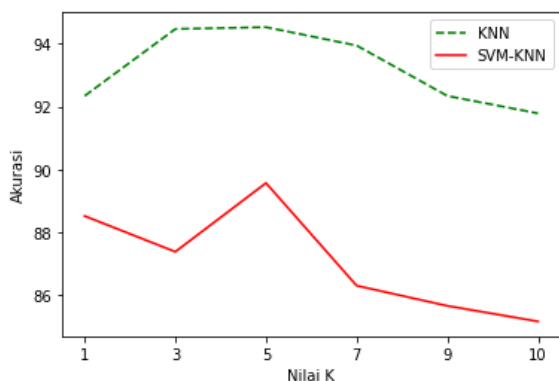
Gambar 3. Pengujian cost

Pada proses pengujian pengaruh hyperparameter c, hasil dari nilai akurasi pada pengujian dapat dilihat pada Gambar 3. Dari hasil pengujian tersebut c memiliki pengaruh pada kinerja dari SVM dan SVM-KNN. Pengaruh c adalah perubahan nilai *Lagrange Multiplier*. Bila nilai c

semakin kecil maka perubahan *Lagrange Multiplier* akan kecil juga yang dilakukan pada proses penghitungan pada fungsi *simplified SMO*.

3.4. Pengujian K dengan Akurasi

Pengujian hyperparameter k dengan akurasi pada metode klasifikasi KNN dan SVM-KNN. Nilai k digunakan pada pengujian tahap ini antara lain : 1, 3, 5, 7, 9, 10. Hasil dari pengujian dapat dilihat pada Gambar 4.

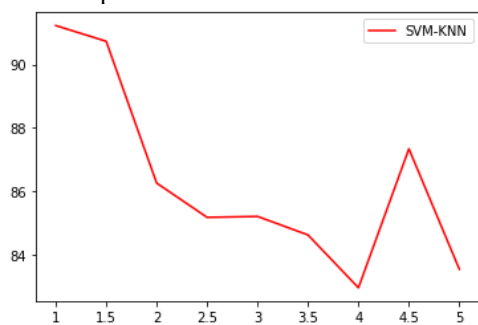


Gambar 4. Pengujian k

Pada proses pengujian pengaruh hyperparameter k , hasil dari nilai akurasi pada pengujian dapat dilihat pada Gambar 4. Dari hasil pengujian tersebut k memiliki pengaruh pada kinerja dari KNN dan SVM-KNN. Pengaruh k adalah pemberat pada proses klasifikasi. Bila nilai k semakin kecil maka hanya sedikit yang dimasukkan ke dalam perhitungan yang mana dapat mengurangi nilai akurasi. Akan tetapi jika nilai k terlalu tinggi maka akan mengurani nilai akurasi juga.

3.5. Pengujian Miu dengan Akurasi

Pengujian hyperparameter μ dengan akurasi pada metode klasifikasi SVM-KNN. Nilai μ digunakan pada pengujian tahap ini antara lain : 1, 1,5, 2, 2,5, 3, 3,5, 4, 4,5, 5. Hasil dari pengujian dapat dilihat pada Gambar 5.



Gambar 5. Pengujian μ

Pada proses pengujian pengaruh hyperparameter μ , hasil dari nilai akurasi pada pengujian dapat dilihat pada Gambar 5. Dari hasil pengujian tersebut μ memiliki pengaruh pada kinerja

dari SVM-KNN. Pengaruh μ adalah pengalihan dari variable k . Bila nilai μ semakin kecil maka hanya sedikit yang dimasukkan ke dalam perhitungan yang mana dapat mengurangi nilai akurasi. Akan tetapi jika nilai μ terlalu tinggi maka akan mengurani nilai akurasi juga.

3.6. Hasil

Dengan menggunakan hyperparameter yang optimal setiap metode klasifikasi akan dilakukan pengujian. Hasil pengujian tersebut dapat dilihat pada Tabel 2.

Tabel 2. Hasil Klasifikasi

Metode Klasifikasi	Akurasi	
	Rata-rata	Standar Deviasi
SVM ($\gamma=1$, $\text{tol}=0,01$, $c=1$)	94,0909	1,5301
KNN ($k=5$)	91,7355	0,0
SVM-KNN ($\gamma=0,001$, $\text{tol}=0$, 0001 , $c=3$, $k=5$, $\mu=1$)	92,562	0,826
SVM-KNN ($\gamma=1$, $\text{tol}=0,0001$, $c=1$, $k=5$, $\mu=1$)	94,2562	1,5345

Dari hasil pengujian tersebut, algoritma klasifikasi SVM-KNN memiliki nilai akurasi tertinggi akan tetapi memiliki nilai standar deviasi yang tinggi. Dapat disimpulkan metode SVM-KNN dapat mengidentifikasi lebih baik daripada hanya menggunakan SVM atau KNN namun presisi dari metode SVM-KNN kurang baik dibandingkan dengan SVM atau KNN.

4. KESIMPULAN

Dari pengujian yang telah dilakukan, rata-rata nilai akurasi dari 20 instansi yang dilakukan pada setiap *hyperparameter* optimal. Rata-rata dari nilai akurasi dari metode SVM, KNN, dan SVM-KNN adalah sebagai berikut 94,04%, 91,73%, 92,56%. Penggabungan metode SVM-KNN yang digunakan pada penelitian Ming, dkk. memiliki akurasi yang lebih baik daripada hanya menggunakan SVM atau KNN namun presisi dari metode SVM-KNN kurang baik dibandingkan dengan SVM atau KNN.

KNN dan SVM-KNN mengalami penurunan akurasi dikarenakan rasio data pembelajaran dan pengujian berbeda. Pada pengujian hyperparameter dilakukan Kfold validation dengan 10 fold. Rasio data pembelajaran dan data pengujian pada pengujian hyperparameter adalah 9:1 yang mana metode KNN dan SVM-KNN berperforma lebih baik dari pada SVM.

Dari hasil pengujian SVM-KNN, karena nilai γ terlalu kecil maka SVM-KNN tidak menggunakan metode klasifikasi SVM dan hanya menggunakan metode klasifikasi KNN.

DAFTAR PUSTAKA

AHMAD, M., TUNDJUNGSARI, V., WIDIANTI, D., AMALIA, P., & RACHMAWATI, U.

2017. Diagnostic Decision Support System of Chronic. Second International Conference on Informatics and Computing (ICIC).
- AL-MEJIBLI, I. S., ALWAN, J. K., ABD, & HAMED, D. 2020. The effect of gamma value on support vector machine performance with different kernels. *International Journal of Electrical and Computer Engineering (IJECE)*, 10, pp.5497-5506.
- CHARLEONNAN, A., FUFAUNG, T., NIYOMWONG, T., & CHOKCHUEYPATTANAKIT, W. 2016. Predictive Analytics for Chronic Kidney Disease. *Management and Innovation Technology International Conference (MITicon)*.
- DESAI, M. 2019. Early Detection and Prevention of Chronic Kidney . 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA).
- DEVIKA, R., AVILALA, S. V., & SUBRAMANIASWAMY, V. 2019. Comparative Study of Classifier for Chronic. 3rd International Conference on Computing Methodologies and Communication (ICCMC).
- FERNANDO, M. P., CESAR, F., DAVID, N., JOSE, H. O. 2021. Missing the missing values: The ugly duckling of fairness in machine learning. *International Journal of Intelligent Systems*.
- GHOSH, S., DASGUPTA, A., SWETAPADMA, A. 2019. A Study on Support Vector Machine based Linear and Non-Linear Pattern Classification. *International Conference on Intelligent Sustainable Systems (ICISS)*.
- HOQUE dan ALJAMAAN. 2021. Impact of Hyperparameter Tuning on Machine Learning Models in Stock Price Forecasting. *IEEE Access*.
- MING, T., YI, Z., & SONGCAN, C. 2003. Improving Support Vector Machine Classifier by Improving Support Vector Machine Classifier by Based on the Best Distance Measurement. *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*.
- PUJANTO, U., RAMADHANI, N. A., WIBAWA, A. P. 2018. Support Vector Machine with Purified K-Means Clusters for Chronic Kidney Disease Detection. *The 2nd East Indonesia Conference on Computer and Information Technology (EIconCIT)*, pp.56-60.
- QUE, Q., dan BELKIN, M. 2017. Back To The Future: Radial Basis Function Network Revisited. *IEEE Transactions On Pattern Analysis and Machine Intelligence*.
- RAJU, V. N. G., LAKSHMI, K. P., JAIN, V. M., KALIDINDI, A., & PADMA, V. 2020. Study the Influence of Normalization/Transformation process on the Accuracy of Supervised Classification. *Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*.
- RUBINI, L. J., PANDIAN, S., & ESWARAN. 2015. *Chronic Kidney Disease Data Set*. [online] Tersedia di: <https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease> [Diakses 10 Januari 2021]
- WU, Z., XIONG, Y., YU, S. X., LIN, D. 2018. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.
- ZHANG, S. 2021. Challenges in KNN Classification. *IEEE Transactions on Knowledge and Data Engineering*.

Halaman ini sengaja dikosongkan