

PENERAPAN METODE SUPPORT VECTOR MACHINE PADA DIAGNOSA HEPATITIS

Raudlatul Munawarah¹, Oni Soesanto², M. Reza Faisal³

^{1,3}Prodi Ilmu Komputer FMIPA UNLAM,

²Prodi Matematika FMIPA UNLAM

Jl. A. Yani Km 36 Banjarbaru, Kalimantan selatan

Email: raudlatulm9306@gmail.com

Abstract

Machine learning has been widely used in the medical to analyze medical datasets. One method in machine learning is Support Vector Machine (SVM). By looking at the concept of SVM method to finding the optimal separator function that can separate the two sets of data from two different classes, came the idea whether this method can be used to diagnose a person suffering from a particular disease or not, especially hepatitis. Therefore, this research aimed to determine and analyze the ability of Support Vector Machine method for diagnosing hepatitis. Analysis of the ability of method using data testing with both kernel with training data 100 positive data label and 100 negative data label. The result showed that trial using the linear kernel function get 68-83% for true prediction percentage and 70-96% for RBF kernel function. The conclusion from this study is SVM method can be used to diagnose hepatitis with high degree of accuracy and RBF kernel function has accuracy rate higher than linear kernel function.

Keywords: *classification, Machine learning, hepatitis, support vector machine*

Abstrak

Machine learning telah banyak digunakan dalam bidang medis untuk menganalisa dataset medis. Salah satu metode machine learning adalah Support Vector Machine (SVM). Dengan melihat konsep metode SVM yaitu menemukan fungsi pemisah optimal yang bisa memisahkan dua set data dari dua kelas yang berbeda, muncul pemikiran apakah metode ini dapat digunakan untuk mendiagnosa seseorang mengidap penyakit tertentu atau tidak, khususnya penyakit hepatitis. Untuk itu penelitian ini dilakukan untuk mengetahui dapat digunakannya metode SVM dan menganalisis kemampuan metode Support Vector Machine untuk mendiagnosa penyakit Hepatitis. Analisis kemampuan metode diketahui dengan uji coba menggunakan data testing dengan kedua kernel dengan data training 100 data positif dan 100 data negatif. Hasil uji coba dengan menggunakan fungsi kernel linier mendapatkan hasil persentase benar 68-83 % dan fungsi kernel RBF 70-96 %. Kesimpulan dari penelitian ini adalah metode SVM dapat digunakan untuk mendiagnosa penyakit hepatitis dengan tingkat akurasi cukup tinggi dan fungsi kernel RBF memiliki tingkat akurasi cenderung lebih tinggi dibandingkan fungsi kernel linier.

Kata kunci : *klasifikasi, machine learning, hepatitis, support vector machine*

1. PENDAHULUAN

Teknologi informasi turut berkembang sejalan dengan perkembangan peradaban manusia. Perkembangan teknologi informasi meliputi perkembangan infrastruktur, seperti hardware, software, teknologi penyimpanan data (*storage*), dan teknologi. Saat ini perkembangan teknologi informasi telah merambah ke berbagai sektor termasuk kesehatan.

Pada bidang kesehatan, perkembangan ilmu kedokteran mengalami kemajuan yang sangat pesat yang ditandai dengan ditemukannya penyakit-penyakit baru yang belum teridentifikasi sebelumnya. Salah satu penyakit yang berkembang saat ini yaitu penyakit pada organ hati. Salah satunya adalah penyakit hepatitis. Hepatitis adalah kelainan hati berupa peradangan (sel) hati. Peningkatan ini disebabkan adanya gangguan atau kerusakan membran hati. Ada dua faktor penyebabnya yaitu faktor infeksi dan faktor non infeksi. Faktor penyebab infeksi antara lain virus hepatitis dan bakteri. Peradangan ini ditandai dengan meningkatnya kadar enzim hati. Diagnosa awal penyakit ini setelah memperhatikan gejala adalah melakukan tes fungsi hati yang biasa disebut LFT (*Liver Function Test*)^[4].

Machine learning telah banyak digunakan dalam bidang medis untuk menganalisa dataset medis^[1]. Salah satu metode *machine learning* adalah *Support Vector Machine* (SVM). Ciri dari metode ini adalah menemukan fungsi pemisah (*klasifier*) yang optimal yang bisa memisahkan dua set data dari dua kelas yang berbeda.

Support Vector Machine (SVM) adalah sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang fitur (*feature space*) berdimensi tinggi, dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan *learning bias* yang berasal dari teori pembelajaran statistik. Teori yang mendasari SVM sendiri sudah berkembang sejak 1960-an, tetapi baru diperkenalkan oleh Vapnik, Boser dan Guyon pada tahun 1992 dan sejak itu SVM berkembang dengan pesat. SVM adalah salah satu teknik yang relatif baru dibandingkan dengan teknik lain, tetapi memiliki performansi yang lebih baik di berbagai bidang aplikasi seperti *bioinformatics*, pengenalan tulisan tangan, klasifikasi teks dan lain sebagainya^[5].

Dengan melihat dari konsep metode *Support Vector Machine* ini, muncul pemikiran apakah metode ini dapat digunakan untuk mendiagnosa seseorang mengidap penyakit tertentu atau tidak berdasarkan hasil tes fungsi hati atau rekam medis pasien, khususnya penyakit hepatitis. Hasil tes fungsi hati inilah yang menjadi bahan penelitian dengan menggunakan metode *machine learning*, yaitu metode *Support Vector Machine*.

2. METODOLOGI PENELITIAN

Pada penelitian ini metode yang digunakan adalah metode *Support Vector Machine* untuk klasifikasi atau yang bisa disebut SVC (*Support Vector Classification*). Klasifikasi adalah proses untuk menemukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data dengan tujuan untuk memperkirakan kelas yang tidak diketahui dari suatu objek. Secara umum dalam proses klasifikasi memiliki dua proses yaitu :

1. Proses training : pada proses training digunakan training set yang telah diketahui label-labelnya untuk membangun model atau fungsi.
2. Proses testing : untuk mengetahui keakuratan model atau fungsi yang akan dibangun pada proses training, maka digunakan data yang disebut dengan testing set untuk memprediksi label-labelnya^[5].

Dalam SVM, untuk memisahkan data terhadap kelasnya, SVM akan membangun sebuah *hyperplane* (bidang pemisah). Sebuah *hyperplane* (bidang pemisah) yang baik, bukan hanya *hyperplane* yang bisa digunakan untuk memisahkan data, akan tetapi *hyperplane* yang baik adalah *hyperplane* yang memiliki batasan (*margin*) yang paling besar. Pencarian bidang pemisah terbaik inilah yang menjadi inti dari support vector machine. Akan tetapi, dalam proses pencarian *hyperplane* tersebut, akan muncul permasalahan baru yaitu sebuah formula yang sangat sulit untuk dipecahkan, yang disebut dengan permasalahan *Quadratic Programming*^[1].

Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah kelas pada input space. Pattern yang merupakan anggota dari dua buah kelas : +1 dan -1 dan berbagi alternative garis pemisah (*discrimination boundaries*). *Margin* adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing kelas. *Pattern* yang paling dekat ini disebut sebagai *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM^[2].

Data yang tersedia dinotasikan sebagai $\vec{x}_i \in \mathbb{R}^d$ sedangkan label masing-masing dinotasikan $y_i \in \{-1, +1\}$ untuk $i = 1, 2, \dots, l$, yang mana l adalah banyaknya data. Diasumsikan kedua kelas -1 dan +1 dapat terpisah secara sempurna oleh *hyperplane* berdimensi d , yang didefinisikan :

$$\vec{w}_i \cdot \vec{x}_i + b = 0 \dots \dots \dots (1)$$

Pattern \vec{x}_i yang termasuk kelas -1 (sampel negatif) dapat dirumuskan sebagai pattern yang memenuhi pertidaksamaan :

$$\vec{w}_i \cdot \vec{x}_i + b \leq -1 \dots \dots \dots (2)$$

Sedangkan pattern \vec{x}_i yang termasuk kelas +1 (sampel positif) memenuhi pertidaksamaan :

$$\vec{w}_i \cdot \vec{x}_i + b \geq +1 \dots \dots \dots (3)$$

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara *hyperplane* dan titik terdekatnya, yaitu $1 / \|\vec{w}\|$. Hal ini dapat dirumuskan sebagai *Quadratic Programming* (QP) Problem, yaitu mencari titik minimal persamaan (4) dengan memperhatikan *constraint* persamaan (8).

$$\min \tau(\vec{w}) = \frac{1}{2} \|\vec{w}\|^2 \dots \dots \dots (4)$$

$$y_i (\vec{x}_i \cdot \vec{w} + b) - 1 \geq 0, \forall i \dots \dots \dots (5)$$

Problem ini dapat dipecahkan dengan berbagai teknik komputasi, diantaranya dengan *Lagrange Multiplier*.

$$L(\vec{w}, b, \alpha) = \frac{1}{2} \|\vec{w}\|^2 - \sum_{i=1}^l \alpha_i (y_i (\vec{x}_i \cdot \vec{w} + b) - 1) \dots \dots \dots (6)$$

α_i adalah *Lagrange Multiplier*, yang bernilai nol atau positif ($\alpha_i \geq 0$). Nilai optimal dari persamaan (6) dapat dihitung dengan meminimalkan L terhadap \vec{w}

dan b, dan memaksimalkan L terhadap α_i . Dengan memperhatikan sifat bahwa pada titik optimal gradient $L = 0$, persamaan (6) dapat dimodifikasi sebagai maksimalisasi problem yang hanya mengandung α_i , sebagaimana persamaan (7) berikut.

Maksimasi :

$$\sum_{i=1}^l \alpha_i - \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j x_i x_j \dots\dots\dots(7)$$

Dengan Constraint :

$$\alpha_i \geq 0 \quad (i = 1, 2, 3, \dots, l) \quad \sum_{i=1}^l \alpha_i y_i \dots\dots\dots(8)$$

Dari perhitungan ini diperoleh α_i yang kebanyakan bernilai positif. Data yang berkorelasi dengan α_i yang positif inilah yang disebut *support vector*^[2].

Penjelasan diatas berdasarkan asumsi bahwa kedua belah kelas dapat terpisah secara sempurna oleh *hyperplane*. Akan tetapi, pada umumnya dua belah kelas pada *input space* tidak dapat terpisah secara sempurna (*non linear separable*). Hal ini menyebabkan *constraint* pada persamaan (8) tidak dapat terpenuhi, sehingga optimisasi tidak dapat dilakukan. Untuk mengatasi masalah ini, SVM dirumuskan dengan memperkenalkan teknik *softmargin*.

Dalam *softmargin*, persamaan (5) dimodifikasi dengan memasukkan slack variable ζ_i ($\zeta > 0$) sebagai berikut :

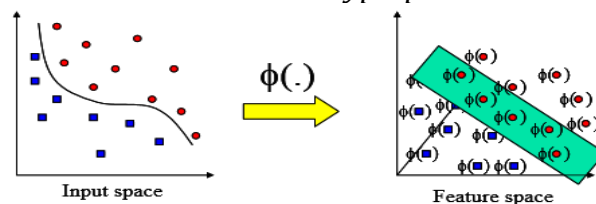
$$y_i (x_i \cdot w + b) \geq 1 - \zeta_i, \forall i \dots\dots\dots(9)$$

dengan demikian persamaan (4) diubah menjadi :

$$\min \tau(w) = \frac{1}{2} \|\vec{w}\|^2 + C \sum_{i=1}^l \zeta_i \dots\dots\dots(10)$$

Parameter C dipilih untuk mengontrol *tradeoff* antara margin dan error klasifikasi ζ . Nilai C yang besar berarti akan memberikan *penalty* yang lebih besar terhadap error klasifikasi tersebut^[1].

Untuk menyelesaikan problem *non-linear*, SVM dimodifikasi dengan memasukkan fungsi kernel. Dalam *non-linear SVM*, pertama-tama data \vec{x} dipetakan oleh fungsi $\Phi(\vec{x})$ ke ruang vektor yang berdimensi lebih tinggi. *Hyperplane* yang memisahkan kedua kelas tersebut dapat dikonstruksikan. Selanjutnya gambar (1) menunjukkan bahwa fungsi Φ memetakan tiap data pada input space tersebut ke ruang vektor baru yang berdimensi lebih tinggi (dimensi 3), sehingga kedua kelas dapat dipisahkan secara linear oleh sebuah *hyperplane*.



Gambar 1. Fungsi Φ memetakan data ke ruang vektor yang berdimensi lebih tinggi

Selanjutnya proses pembelajaran pada SVM dalam menemukan titik-titik *support vector*, hanya bergantung pada *dot product* dari data yang sudah ditransformasikan pada ruang baru yang berdimensi lebih tinggi, yaitu $\Phi(\vec{x}_i) \cdot \Phi(\vec{x}_j)$. Karena umumnya transformasi Φ ini tidak diketahui, dan sangat sulit untuk difahami secara mudah, maka perhitungan *dot product* dapat digantikan dengan

fungsi kernel $K(\vec{x}_i, \vec{x}_j)$ yang mendefinisikan secara implisit transformasi Φ . Hal ini disebut sebagai Kernel Trick, yang dirumuskan :

$$K(\vec{x}_i, \vec{x}_j) = \Phi(\vec{x}_i) \cdot \Phi(\vec{x}_j) \dots \dots \dots (12)$$

$$f(\Phi(\vec{x})) = \vec{w} \cdot \Phi(\vec{x}) + b \dots \dots \dots (13)$$

$$= \sum_{i=1, x_i \in SV}^n \alpha_i y_i \Phi(\vec{x}) \cdot \Phi(\vec{x}_i) + b \dots \dots \dots (14)$$

$$= \sum_{i=1, x_i \in SV}^n \alpha_i y_i K(x, x_i) + b \dots \dots \dots (15)$$

Syarat sebuah fungsi untuk menjadi fungsi kernel adalah memenuhi teorema Mercer yang menyatakan bahwa matriks kernel yang dihasilkan harus bersifat *positive semi-definite*. Fungsi kernel yang umum digunakan adalah sebagai berikut:

a. Kernel Linier
 $K(x, y) = x \cdot y \dots \dots \dots (16)$

b. Polynomial
 $K(x, y) = (x \cdot y + c)^d \dots \dots \dots (17)$

c. Radial Basis Function (RBF)
 $K(\vec{x}, \vec{y}) = \exp\left(-\frac{\|\vec{x} - \vec{y}\|^2}{2\sigma^2}\right) \dots \dots \dots (18)$

d. Sigmoid
 $K(\vec{x}, \vec{y}) = \tanh(\kappa \langle \vec{x} \cdot \vec{y} \rangle + \theta) \dots \dots \dots (19)$

Pada penelitian ini kernel yang digunakan adalah kernel linier dan kernel RBF. Fungsi kernel yang direkomendasikan untuk diuji pertama kali adalah fungsi kernel RBF karena memiliki performansi yang sama dengan kernel linier pada parameter tertentu, memiliki perilaku seperti fungsi kernel *sigmoid* dengan parameter tertentu dan rentang nilainya kecil [0,1]^[5].

Bentuk *primal form* yang tadinya sangat susah untuk dipecahkan, akan dirubah kedalam bentuk *dual form* yang hanya akan mengandung nilai α . Berbagai algoritma telah dikembangkan untuk mencari nilai α tersebut. Akan tetapi, algoritma-algoritma tersebut memerlukan waktu yang lama, apalagi jika dipakai untuk data yang berukuran besar, karena algoritma tersebut menggunakan *numerical quadratic programming* sebagai *inner loop*^[6].

Oleh karena itu, muncullah algoritma-algoritma yang dapat menangani masalah pemecahan nilai α dalam proses training tersebut. Salah satunya dengan metode sekuensial. Adapun langkah-langkah umum dari metode penyelesaian training ini adalah :

1. Menginisiasi $\alpha_i = 0 \dots \dots \dots (20)$

Hitung matriks $D_{ij} = y_i y_j (K(\vec{x}_i \cdot \vec{x}_j) + \lambda^2) \dots \dots \dots (21)$

2. Lakukan step (a), (b) dan (c) di bawah untuk $k = 1, 2, \dots, l$

(a) $E_i = \sum_{j=1}^l \alpha_j D_{ij} \dots \dots \dots (22)$

$$(b) \quad \delta\alpha_i = \min \{ \max [\gamma(1 - E_i), -\alpha_i], C - \alpha_i \} \dots \dots \dots (23)$$

$$(c) \quad \alpha_i = \alpha_i + \delta\alpha_i \dots \dots \dots (24)$$

3. Kembali ke step-2 sampai nilai α konvergen (tidak ada perubahan signifikan)^[6].

3. HASIL DAN PEMBAHASAN

3.1. HASIL

Hasil yang didapatkan dari perhitungan dengan menggunakan metode ini adalah sebuah persamaan bidang pemisah yang akan memisahkan dataset menjadi dua kelas berbeda, yaitu positif dan negatif. Adapun contoh hasil dari uji coba adalah sebagai berikut :

Contoh dibawah adalah hasil tes fungsi hati seorang pasien :

1 : Total Bilirubin	:	2.7 mg/dl
2 : Direct Bilirubin	:	1.3 mg/dl
3 : Alkaline Phosphotase	:	275 U/L
4 : Sgpt Alamine	:	123 U/L
5 : Sgot Aspartate	:	73 U/L
6 : Total Proteins	:	6.2 g/L
7 : Albumin	:	3.3 g/L
8 : Ratio A/G	:	1.1

Setelah dilakukan proses maka didapatkan hasil bahwa pasien tersebut **positif** mengidap hepatitis.

No	Total Billir...	Direct billi...	Alkaline P...	Alamine A...	Aspartat...	Total prot...	Albumin	A/G Ratio	Prediksi
0	2.7	1.3	275.0	123.0	73.0	6.2	3.3	1.1	Positif

Gambar 2. Hasil klasifikasi

3.2. PEMBAHASAN

Pada penelitian ini, data yang berjumlah 579 data dibagi dua untuk data training dan data testing. Untuk data training digunakan 100 data kelas label positif dan 100 data negatif. Dan sisa data digunakan untuk testing. Untuk testing, sistem akan mengambil sejumlah data secara random dari dataset sesuai dengan jumlah input data testing yang diinputkan di form oleh user.

Data training dihitung dengan menggunakan salah metode penyelesaian training data SVM yaitu metode sekuantial. Hasil training adalah sebuah pembelajaran yang tersimpan didalam sistem yang akan menjadi acuan bagi sistem untuk menentukan sebuah inputan data tes fungsi hati baru mengidap penyakit hepatitis atau tidak. Penyelesaian dari langkah-langkah diatas dilakukan satu persatu seperti dibawah ini :

1. Menginisiasi awal untuk nilai α , C, epsilon, gamma, dan lamda
 $\alpha=0$, C = 1, epsilon = 0.001, gamma = 0.5, lamda=0.5
2. Memasukkan data uji

Tabel 1. Tabel contoh data uji

No.	X1	X2	X3	X4	X5	X6	X7	X8	Y
A1	0.9	0.3	202	22	19	7.4	4.1	1.2	-1
A2	0.7	0.1	187	16	18	6.8	3.3	0.9	1
A3	10.9	5.5	699	64	100	7.5	3.2	0.74	1
A4	7.3	4.1	490	60	68	7	3.3	0.89	1
A5	0.9	0.3	310	61	58	7	3.4	0.9	-1

Dimana 1 adalah label positif dan -1 adalah label negatif.

- Menentukan *dot product* setiap data dengan memasukkan fungsi kernel (K). Rumus fungsi kernel yang umum seperti pada persamaan (16) sampai dengan persamaan (19). Fungsi kernel digunakan adalah fungsi kernel linier. Sebelumnya data di transpose karena menggunakan perkalian matriks $A \times A^T$.

Tabel 2. Tabel transpose data

A1	A2	A3	A4	A5
0.9	0.7	10.9	7.3	0.9
0.3	0.1	5.5	4.1	0.3
202	187	699	490	310
22	16	64	60	61
19	18	100	68	58
7.4	6.8	7.5	7	7
4.1	3.3	3.2	3.3	3.4
1.2	0.9	0.74	0.89	0.9

Pada metode kernel, data tidak direpresentasikan secara individual, melainkan lewat perbandingan antara sepasang data. Setiap data akan dibandingkan dengan dirinya dan data lainnya. Kita misalkan untuk data uji berjumlah 5 data maka perbandingan datanya seperti terlihat pada tabel :

Tabel 3. Tabel Perbandingan Data

	A1	A2	A3	A4	A5
A1	K(A1,A1)	K(A1,A2)	K(A1,A3)	K(A1,A4)	K(A1,A5)
A2	K(A2,A1)	K(A2,A2)	K(A2,A3)	K(A2,A4)	K(A2,A5)
A3	K(A3,A1)	K(A3,A2)	K(A3,A3)	K(A3,A4)	K(A3,A5)
A4	K(A4,A1)	K(A4,A2)	K(A4,A3)	K(A4,A4)	K(A4,A5)
A5	K(A5,A1)	K(A5,A2)	K(A5,A3)	K(A5,A4)	K(A5,A5)

Berikut contoh perhitungan dengan data A1 dan A1 :

$$K(A1, A1) = ((0.9*0.9)+(0.3*0.3)+(202*202)+(22*22)+(19*19)+(7.4*7.4)+(4.1*4.1)+(1.2*1.2)) = 41722.91$$

Semua data dihitung dengan cara sama, baris x kolom sehingga menghasilkan nilai dot product seperti ditunjukkan oleh tabel dibawah :

Karena ada 5 data uji, maka didapatkan matriks 5x5.

Tabel 4. Hasil perhitungan kernel

	A1	A2	A3	A4	A5
A1	41722.91	38533.59	144587	101666.2	65131.72
A2	38533.59	35607.44	133607.4	93878.81	60050.29
A3	144586.968	133607.4	502913.1	353315.8	226469.5
A4	101666.198	93878.81	353315.8	248454.8	159572.8
A5	65131.72	60050.29	226469.5	159572.8	103247.3

4. Menghitung matriks dengan rumus :

$$D_{ij} = y_i y_j (K(\vec{x}_i \cdot \vec{x}_j) + \lambda^2)$$

Dimana : D_{ij} = elemen matriks hessian ke-ij

y_i = kelas data ke-i

y_j = kelas data ke-j

λ = batas teoritis yang akan diturunkan

Contoh perhitungan untuk pasangan data A1 dan A1:

$$D_{ij} = (-1)(-1)(41722.91) + 0.5^2 = 41723.16$$

Maka didapatkan hasil untuk semua data:

Tabel 5. Matrik

	a1	a2	a3	a4	a5
a1	41723.16	-38533.3	-144587	-101666	65131.97
a2	-38533.3	35607.69	133607.7	93879.06	-60050
a3	-144587	133607.7	502913.3	353316.1	-226469
a4	-101666	93879.06	353316.1	248455	-159573
a5	65131.97	60050.54	-226469	-159573	103247.5

5. Mencari nilai error dengan rumus :

$$E_i = \sum_{j=1}^l \alpha_j D_{ij}$$

Dimana: E_i = nilai error data ke-i

Maka didapatkan nilai error setiap data adalah :

Tabel 6. Hasil Perhitungan Nilai Error

a1	-88965.4
a2	82255.51
a3	309390.6
a4	217205.8
a5	-78805.9

6. Menghitung nilai delta alpha :

$$\delta\alpha_i = \min \{ \max [\gamma(1 - E_i), -\alpha_i], C - \alpha_i \}$$

Untuk data pertama :

$$\text{Min} (\max (0.5(1 - (-88965.4)), -0.5), 1 - 0.5)$$

$$\text{Min}(\max(44483.2, -0.5), 0.5)$$

$$\text{Min}(44483.2, 0.5)$$

$$= 0.5$$

Maka didapatkan delta alpha sebagai berikut :

Tabel 7. Hasil Perhitungan delta alpha

a1	0.5
a2	-0.5
a3	0.5
a4	-0.5
a5	0.5

Karena nilai maksimum $\delta\alpha$ adalah 0.5 dan lebih dari epsilon (0.001) maka iterasi berlanjut.

7. Menghitung nilai α baru dengan menggunakan rumus :

$$\alpha_i = \alpha_i + \delta\alpha_i$$

Maka didapatkan α sebagai berikut :

Tabel 8. Hasil Perhitungan alpha baru

a1	1
a2	0
a3	0
a4	0
a5	1

8. Mencari nilai bias :

$$b = -\frac{1}{2}(<\vec{w} \cdot \vec{x}_{-1}> + <\vec{w} \cdot \vec{x}_{+1}>)$$

Terlebih dahulu dihitung nilai w :

W_i^+ adalah bobot *dot product* data dengan alpha terbesar di kelas positif

W_i^- adalah bobot *dot product* data dengan alpha terbesar di kelas negatif

$w x^+$ (kelas positif) = $(1 \times (-1) \times 38533.59) + (0 \times 1 \times 35607.44) + (0 \times 1 \times 133607.4) + (0 \times 1 \times 93878.81) + (1 \times (-1) \times 60050.29) = -98583.9$

$w x^-$ (kelas negatif) = $(1 \times (-1) \times 41722.91) + (0 \times 1 \times 38533.59) + (0 \times 1 \times 144587) + (0 \times 1 \times 101666.2) + (0 \times 1 \times 65131.72) = -106855$

maka nilai $b = -1/2 (w x^+ + w x^-) = 102719.3$

Sebenarnya perhitungan diatas belum bisa digunakan untuk fungsi keputusan karena iterasi masih harus diteruskan. Perhitungan dibawah ini hanya untuk contoh perhitungan dengan fungsi keputusan saja

9. Setelah mendapatkan nilai α , w dan b maka dapat dilakukan pengujian dengan contoh data uji berikut :

Tabel 9. Tabel data uji

X1	X2	X3	X4	X5	X6	X7	X8
2.7	1.3	275	123	73	6.2	3.3	1.1

Langkah pertama untuk menguji adalah menghitung dot product antara data uji dengan semua data latih dengan fungsi kernel.

$$K(x,y) = x.y$$

Dimana x adalah data uji dan y adalah semua data latih.

Data ke 1 : $K(x_i, x) = (2.7 \times 0.7) + (1.3 \times 0.1) + (275 \times 187) + (123 \times 16) +$

$$(73 \times 18) + (6.2 \times 6.8) + (3.3 \times 3.3) + (1.1 \times 0.9) = 59706.55$$

$$\text{Data ke 2 : } (0.9 \times 0.7) + (0.2 \times 0.1) + (194 \times 187) + (52 \times 16) + (45 \times 18) + (6 \times 6.8) + (3.9 \times 3.3) + (1.85 \times 0.9) = 59706.55$$

Begitu seterusnya untuk semua data latih maka didapatkan dot product data uji sebagai berikut :

Tabel 10. Hasil perhitungan dot product data uji dengan data latih

A1	A2	A3	A4	A5
59706.55	54763.06	207491.5	147174.3	97045.43

Selanjutnya dilakukan perhitungan fungsi keputusan :

$$f(x) = w \cdot x + b \quad \text{atau} \quad f(x) = \sum_{i=1}^m \alpha_i y_i K(x, x_i) + b$$

Data ke 1 :

$$\begin{aligned} f(x) &= \text{sign}((1 \times (-1) \times 59706.55) - 102719.3 \\ &+ (0 \times 1 \times 54763.06) - 102719.3 \\ &+ (0 \times 1 \times 207491.5) - 102719.3 \\ &+ (0 \times 1 \times 147174.3) - 102719.3 \\ &+ (1 \times 1 \times 97045.43) - 102719.3 \\ &= 796687 \end{aligned}$$

$$= \text{sign}(-796687) = 1$$

Jadi, data uji diatas termasuk kelas **positif**.

4. SIMPULAN

Simpulan yang didapatkan dari penelitian ini adalah :

- Metode *Support Vector Machine* dapat digunakan untuk mendiagnosa penyakit hepatitis dengan belajar dari pembelajaran yang dihasilkan dari pelatihan data training.
- Pembelajaran yang dihasilkan berasal dari pelatihan data tes fungsi hati.
- Pelatihan data training dilakukan dengan metode sekuensial untuk memecahkan masalah *Quadratic Programming (QP) Problem* yaitu pencarian nilai α yang cukup sulit diselesaikan.

DAFTAR PUSTAKA

- [1] Munawarah, Raudlatul. 2016. **Implementasi Metode Support Vector Machine untuk Mendiagnosa Penyakit Hepatitis**. Program S-1 Ilmu Komputer, Universitas Lambung Mangkurat: Banjarbaru.
- [2] Christianini, Nello dan John S. Taylor. 2000. **An Introduction to Support Vector Machines and Other Kernel-based Learning Methods**. Cambridge University Press
- [3] Feldman, David and Gross, Shulamith. 2004. **Mortgage Default: Classification Trees Analysis**. University of New South Wales - Banking & Finance, Australian School of Business.
- [4] Green, Chris W. 2005. **Hepatitis Virus dan HIV**. Yayasan Spiritia: Jakarta
- [5] Ramana, B. V., Babu, S. P., & Venkateswarlu, N. B. 2011. **A Critical Study Of Selected Classification Algorithms For Liver Disease Diagnosis**. International Journal Of Database Management Systems, Vol. 3 (2), Hal. 101-114.

- [6] Platt, John. 1998. ***Sequential Minimal Optimization: A fast Algorithm for Training Support Vector Machine***. Microsoft Research.
<http://www.research.microsoft.com/jplatt>