Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - KI141502

IMPLEMENTASI PENGUKURAN DAN KLASIFIKASI TEKANAN DARAH BERDASARKAN PULSE TRANSIT TIME MENGGUNAKAN METODE TRANSFORMASI WAVELET DAN SUPPORT VECTOR MACHINES

LELI MAHARANI
NRP 5112100011

Dosen Pembimbing I
Anny Yuniarti, S.Kom., M.Comp.Sc.

Dosen Pembimbing II
Dini Adni Navastara, S.Kom., M.Sc.

Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

TUGAS AKHIR – KI141502

IMPLEMENTASI PENGUKURAN DAN KLASIFIKASI TEKANAN DARAH BERDASARKAN PULSE TRANSIT TIME MENGGUNAKAN METODE TRANSFORMASI WAVELET DAN SUPPORT VECTOR MACHINES

LELI MAHARANI
NRP 5112100011

Dosen Pembimbing I
Anny Yuniarti, S.Kom., M.Comp.Sc.

Dosen Pembimbing II
Dini Adni Navastara, S.Kom., M.Sc.

Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

(Halaman ini sengaja dikosongkan)

UNDERGRADUATE THESES – KI141502

IMPLEMENTATION OF MEASUREMENT AND CLASSIFICATION BLOOD PRESSURE BASED ON PULSE TRANSIT TIME USING WAVELET TRANSFORMATION AND SUPPORT VECTOR MACHINES

LELI MAHARANI
NRP 5112100011

First Advisor
Anny Yuniarti, S.Kom., M.Comp.Sc.

Second Advisor
Dini Adni Navastara, S.Kom., M.Sc

Department of Informatics
Faculty of Information Technology
Sepuluh Nopember Institute of Technology
Surabaya 2016

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

IMPLEMENTASI PENGUKURAN DAN KLASIFIKASI TEKANAN DARAH BERDASARKAN PULSE TRANSIT TIME MENGGUNAKAN METODE TRANSFORMASI WAVELET DAN SUPPORT VECTOR MACHINE

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas dan Visi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
LELI MAHARANI
NRP: 5112100011

Disetujui oleh Pembimbing Tugas Akhir:

1. Anny Yuniarti, S.Kom., M.Comp.Sc. ..... (Pembimbing 1)
   (NIP. *198106222005120002*)
2. Dini Adni Navastara, S.Kom., M.Sc. ..... (Pembimbing 2)
   (NIP. *198510172015042001*)

SURABAYA
JULI, 2016

(Halaman ini sengaja dikosongkan)

IMPLEMENTASI PENGUKURAN DAN KLASIFIKASI TEKANAN DARAH BERDASARKAN PULSE TRANSIT TIME MENGGUNAKAN METODE TRANSFORMASI WAVELET DAN *SUPPORT VECTOR MACHINES*

Nama Mahasiswa : LELI MAHARANI
NRP : 5112100011
Jurusan : Teknik Informatika FTIF-ITS
Dosen Pembimbing 1 : Anny Yuniarti, S.Kom., M.Comp.Sc.
Dosen Pembimbing 2 : Dini Adni Navastara, S.Kom., M.Sc.

Abstrak

Pengukuran tekanan darah melalui kuf atau manset lengan umumnya masih digunakan di bidang medis, padahal pengukuran melalui manset merupakan pengukuran dengan cara tidak langsung. Hal ini tentunya akan mempengaruhi hasil dari pengukuran tekanan darah yang selanjutnya akan digunakan sebagai patokan tekanan darah seseorang apakah ia hipertensi atau tidak. Sinyal ECG dan PPG merupakan gelombang sinyal elektrik yang didapatkan langsung dari tubuh yang berkaitan dengan aktivitas jantung. Terdapat selang waktu saat jantung memompa darah ke seluruh tubuh, hal ini disebut dengan Pulse Transit Time (PTT). Dengan melakukan penghitungan PTT dapat diketahui tekanan sistol dan diastol sehingga dapat digunakan sebagai metode pengukur tekanan darah.

Pada rekaman sinyal ECG dan PPG, seringkali terganggu dengan derau sehingga harus dibersihkan terlebih dahulu. Oleh karena itu Tugas Akhir ini mengimplementasikan metode Transformasi Wavelet sebagai penghilang derau dan ekstraksi fitur dan diklasifikasi menjadi 4 kelas, yaitu normal, hipertensi, hipertensi stadium 1, dan hipertensi stadium 2 dengan menggunakan Support Vector Machines (SVM).

Pada tugas akhir ini hasil yang didapat untuk menghilangkan derau menggunakan dekomposisi sinyal

menggunakan DWT Daubechies 6 (db6) pada level 8 dan SVM one against all dengan kernel RBF dengan parameter c sebesar 20 dianggap efektif untuk mengambil fitur dan mengklasifikasi rekaman medis dari MIMIC II Database Physionet dan menghasilkan akurasi sebesar 91,67%

Kata kunci: Tekanan Darah, ECG, PPG, Pulse Transit Time, Wavelet, Klasifikasi, Support Vector Machines

IMPLEMENTATION OF MEASUREMENT AND CLASSIFICATION BLOOD PRESSURE BASED ON PULSE TRANSIT TIME USING WAVELET TRANSFORM AND SUPPORT VECTOR MACHINES

Student's Name : LELI MAHARANI
Student's IDE : 5112100011
Department : Teknik Informatika FTIF-ITS
First Advisor : Anny Yuniarti, S.Kom., M.Comp.Sc.
Second Advisor : Dini Adni Navastara, S.Kom., M.Sc.

Abstract

Measurement of blood pressure through the arm cuff commonly still used in the medical field, but through the cuff measurement is a measurement of an indirect way. This will certainly affect the results of blood pressure measurements will then be used as a benchmark a person's blood pressure or hypertension if he did not. ECG and PPG signal is an electrical signal waves obtained directly from the body that are associated with the activity of the heart. There is an interval of time when the heart pumps blood throughout the body, it is called the Pulse Transit Time (PTT). By calculating the PTT, diastolic and systolic pressure can be calculated so that it can be used as a method of measuring blood pressure.

In the recording ECG and PPG signals, often disturbed by noise and should be cleaned first. Therefore, this final project implements Wavelet Transform method as noise removal and features extraction and then classified into four classes, who are normal, hypertension, stage 1 hypertension and stage 2 hypertension using Support Vector Machines (SVM).

In this thesis the results obtained to remove noise using a decomposition signal using DWT Daubechies 6 at level 8 and SVM one against all using kernel RBF and parameter c of 20 is considered effective to take on the features and classifying medical

records of MIMIC II Database Physionet and produce accuracy of 91.67%

Keywords : Blood Pressure, ECG, PPG Pulse Transit Time, Wavelet, Classification, Support Vector Machines.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR KODE SUMBER

(Halaman sengaja dikosongkan)

BAB I
PENDAHULUAN

1.1 Latar Belakang

World Health Organization (WHO) pada tahun 2014 melaporkan bahwa hipertensi menyebabkan 9,4 juta kematian orang per tahun [1]. Hipertensi telah diakui sebagai faktor kedua penyakit kardiovaskular setelah diabetes. Hipertensi pada dasarnya merupakan penyakit yang bersifat *silent killer*, karena banyak orang yang tidak menyadari hipertensi dan cara mengendalikannya. Tekanan darah memiliki batas atas yang disebut tekanan sistolik dan batas bawah yang disebut tekanan diastolik.

Terdapat dua cara pengukuran tekanan darah yaitu secara langsung dan tidak langsung. Metode pengukuran yang sering digunakan di Indonesia menggunakan kuf di lengan yang menggunakan prinsip osilometrik atau prinsip auskultatori merupakan cara pengukuran tidak langsung. Penggunaan kuf ini menyebabkan alat pengukuran akan berkurang bagi segi keakuratan, penggunaan tenaga, dan kekerapan penggunaannya.

Permasalahannya adalah penggunaan kuf lengan memiliki pedoman-pedoman yang harus dipenuhi dan diperhatikan untuk mendapatkan hasil tekanan darah yang lebih akurat. Seperti ukuran manset harus disesuaikan dengan besarnya lengan pasien, karena ketidak sesuaian ukuran manset akan mengurangi validitas hasil pengukuran. Selain itu letak maset juga harus dipasang di lengan pasien secara benar. Sehingga faktor *human error* sangat mungkin terjadi pada metode pengukuran ini.

Tugas akhir ini bertujuan sebagai alternatif cara pengukuran tekanan darah tanpa kuf (manset) dengan kata menggunakan prinsip pengukuran darah langsung. Pengukuran tekanan darah berdasarkan sinyal *Elektrocardiogram* (ECG) yang berasal dari jantung dan *Photoplethysmography* (PPG) yang berasal dari denyut jantung pada jari tangan [2]. Salah satu metode dasar yang

dapat digunakan untuk mengukur tekanan darah langsung adalah dengan menggunakan *Pulse Transit Time* (PTT). Melalui tugas akhir ini, penulis ingin mengimplementasikan metode pengukuran darah berdasarkan PPT yang didapat dari jarak waktu sinyal ECG dan PPG yang sebelumnya melalui *prepocessing* dengan menggunakan transformasi wavelet [3] untuk mendapatkan data sinyal yang efektif dan efisien. Setelah mendapatkan nilai PTT, selanjutnya diproses menggunakan *Support Vector Machines* (SVM)[4] untuk mengetahui apakah seseorang memiliki tekanan darah normal, prehipertensi, stage 1 hipertensi, atau stage 2 hipertensi.

1.2 Rumusan Masalah

Tugas Akhir ini mengangkat beberapa rumusan masalah sebagai berikut:

1. Bagaimana mengekstraksi data sinyal ECG dan PPG?
2. Bagaimana mengimplementasikan *Pulse Transit Time* (PTT) untuk mengukur tekanan darah?
3. Bagaimana menerapkan metode *Support Vector Machine* (SVM) untuk mengklasifikasikan data sinyal ECG dan PPG?
4. Bagaimana tingkat akurasi dari metode yang dipakai?

1.3 Batasan Permasalahan

Permasalahan yang dibahas pada Tugas Akhir ini memiliki batasan sebagai berikut:

1. *Dataset* yang dijadikan data uji adalah data *cuff-less blood pressure* dari repositori UCI yang berisi atribut data sinyal ECG dari channel II, sinyal PPG, dan ABP (*Arterial Blood Pressure*) pada frekuensi 125Hz [5].
2. Jumlah data yang digunakan adalah 60 rekaman individu masing masing 15 untuk pasien normal, prehipertensi, stadium 1 hipertensi, dan stadium 2 hipertensi. Pelabelan

kelas (normal, prehipertensi, stadium 1 hipertensi, dan stadium 2 hipertensi) pada didapatkan dari dokter berdasarkan analisis tekanan darah arteri pada *dataset*.

3. Diasumsikan bahwa data pasien yang digunakan mempunyai usia dewasa (17-50 tahun), gender, dan tinggi badan yang sama.
4. Perangkat lunak yang digunakan adalah Matlab R2014a.

1.4 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah untuk mengimplementasikan pengukuran tekanan darah berdasarkan *Pulse Transit Time* dari hasil ekstraksi fitur sinyal ECG dan PPG dengan metode Transformasi Wavelet dan melakukan klasifikasi menggunakan metode *Support Vector Machines* (SVM).

1.5 Manfaat

Dengan dibuatnya tugas akhir ini diharapkan dapat memberikan manfaat pada dunia kedokteran untuk mendiagnosis apakah seseorang normal, prehipertensi, hipertensi stadium 1, atau hipertensi stadium 2 tanpa menggunakan kuf lengan yaitu dengan memanfaatkan sinyal ECG dan PPG untuk mendapatkan hasil pengukuran yang lebih tepat.

1.6 Metodologi

Pembuatan Tugas Akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

1.6.1 Penyusunan Proposal

Tahap awal Tugas Akhir ini adalah menyusun proposal Tugas Akhir. Pada proposal, diajukan gagasan untuk mengimplementasikan pengukuran tekanan darah kemudian mengklasifikasi apakah seorang normal, prehipertensi, hipertensi

stadium 1, atau hipertensi stadium 2. Transformasi wavelet untuk dekomposisi sinyal dan *Support Vector Machines* (SVM) digunakan sebagai metode klasifikasi.

1.6.2 Studi Literatur

Pada tahap ini dilakukan untuk mencari informasi dan studi literatur apa saja yang dapat dijadikan referensi untuk membantu pengerjaan Tugas Akhir ini. Informasi didapatkan dari buku dan literatur yang berhubungan dengan data dan metode yang digunakan. Informasi yang dicari adalah sinyal ECG, PPG, Tekanan ABP, *Pulse Transit Time* (PTT), Transformasi Wavelet, dan *Support Vector Machines* (SVM), dan informasi lain yang berkaitan.

1.6.3 Implementasi Perangkat Lunak

Implementasi merupakan tahap untuk membangun metode-metode yang sudah diajukan pada proposal Tugas Akhir. Untuk membangun algoritma yang telah dirancang sebelumnya, maka dilakukan implementasi dengan menggunakan suatu perangkat lunak yaitu Matlab R2014a.

1.6.4 Pengujian dan Evaluasi

Pada tahap ini algoritma yang telah disusun diuji coba dengan menggunakan data uji coba yang ada. Data uji coba tersebut diuji coba dengan menggunakan suatu perangkat lunak dengan tujuan mengetahui kemampuan metode yang dipakai. Dan mengevaluasi hasil tugas akhir dengan paper pendukung yang ada.

1.6.5 Penyusunan Buku

Pada tahap ini disusun buku sebagai dokumentasi dari pelaksanaan Tugas Akhir yang mencangkup seluruh konsep, teori, implementasi, serta hasil yang telah dikerjakan.

1.7 Sistematika Penulisan Laporan

Sistematika penulisan laporan Tugas Akhir adalah sebagai berikut:

1. Bab I. Pendahuluan
   Bab ini berisikan penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan dari pembuatan Tugas Akhir.
2. Bab II. Tinjauan Pustaka
   Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan Tugas Akhir ini. Secara garis besar, bab ini berisi tentang sinyal ECG, Sinyal PPG, *Pulse Transit Time*, Transformasi Wavelet dan *Support Vector Machines* (SVM).
3. Bab III. Perancangan Perangkat Lunak
   Bab ini berisi pembahasan mengenai perancangan dari metode *Pulse Transit Time*, Transformasi Wavelet dan *Support Vector Machines* (SVM) yang digunakan untuk klasifikasi tekanan darah
4. Bab IV. Implementasi
   Bab ini menjelaskan implementasi yang berbentuk kode sumber dari metode *Pulse Transit Time*, Transformasi Wavelet dan *Support Vector Machines* (SVM) untuk klasifikasi.
5. Bab V. Hasil Uji Coba dan Evaluasi
   Bab ini berisikan hasil uji coba dari *Pulse Transit Time*, Transformasi Wavelet dan *Support Vector Machines* (SVM)) yang sudah diimplementasikan pada kode sumber. Uji coba dilakukan dengan mengganti beberapa parameter antara lain nilai *k* pada *fold cross validation*, parameter *C*, dan metode *kernel* pada proses klasifikasi *Support Vector Machines* (SVM).
6. Bab VI. Kesimpulan dan Saran
   Bab ini merupakan bab yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami

pada proses pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.

7. Daftar Pustaka
   Bab ini berisi daftar pustaka yang dijadikan literatur dalam Tugas Akhir.
8. Lampiran
   Dalam lampiran terdapat tabel-tabel data hasil uji coba dan kode sumber program secara keseluruhan.

BAB II
TINJAUAN PUSTAKA

Bab ini berisi pembahasan mengenai teori-teori dasar yang digunakan dalam Tugas Akhir. Teori-teori tersebut diantaranya adalah tekanan darah, sinyal *Electrocardiogram* (ECG), sinyal *Photoplethysmograph* (PPG), *Pulse Transit Time (PTT)*, Transformasi Wavelet, *Support Vector Machines (SVM)* dan beberapa teori lain yang mendukung pembuatan Tugas Akhir.

2.1 Tekanan Darah

Tekanan darah adalah kekuatan yang diperlukan agar darah dapat mengalir di dalam pembuluh darah dan beredar mencapai seluruh jaringan. Tekanan darah merujuk kepada tekanan yang dialami darah pada pembuluh arteri darah ketika darah dipompa oleh jantung ke seluruh anggota tubuh manusia.

Tekanan darah dibuat dengan mengambil dua ukuran dan biasanya diukur seperti berikut 120/80 mmHg [6]. Nomor atas (120) menunjukkan tekanan ke atas pembuluh arteri akibat denyutan jantung yang disebut tekanan sistol. Nomor bawah (80) menunjukkan tekanan saat jantung beristirahat diantara pemompaan yang disebut dengan tekanan diastol. Saat yang paling baik untuk mengukur tekanan darah adalah saat istirahat dalam keadaan duduk atau berbaring. Sistol dan diastol merupakan dua periode yang menyusun satu siklus jantung. Diastol adalah kondisi relaksasi, yakni saat jantung terisi oleh darah yang kemudian diikuti oleh periode kontraksi atau sistol.

National Heart, Lung, And Blood Institute (NHLBI) bekerja sama dengan National High Blood Pressure (NHBPEP) dalam menyusun suatu *guideline* penanganan dan klasifikasi hipertensi secara global yang dijelaskan dalam Joint National Commitee (JNC) [7]. JNC berisi panduan mengenai pencegahan, pendeteksian, evaluasi, dan penanganan dari penyakit tekanan darah. Sejak tahun 2003, telah dipublikasikan JNC 7 yang merevisi

JNC 6 (1997) dengan konten yang lebih sempurna, ringkas dan jelas. Selain itu, juga didukung oleh data-data terbaru (1997-2003) yang diambil dari hasil percobaan klinik serta observasi. Klasifikasi tekanan darah menurut JNC-7 tahun 2003 dapat dilihat pada tabel 2.1.

Tabel 2.1 Klasifikasi Tekanan Darah dari JNC-7 2003

Kategori	SBP (mmHg)	DBP (mmHg)
Normal	<120	dan <80
Prehypertension	120-139	atau 80-89
Stage 1 Hypertension	140-159	atau 90-99
Stage 2 Hypertension	>160	atau 100

Terdapat dua cara pengukuran tekanan darah [6], yaitu:

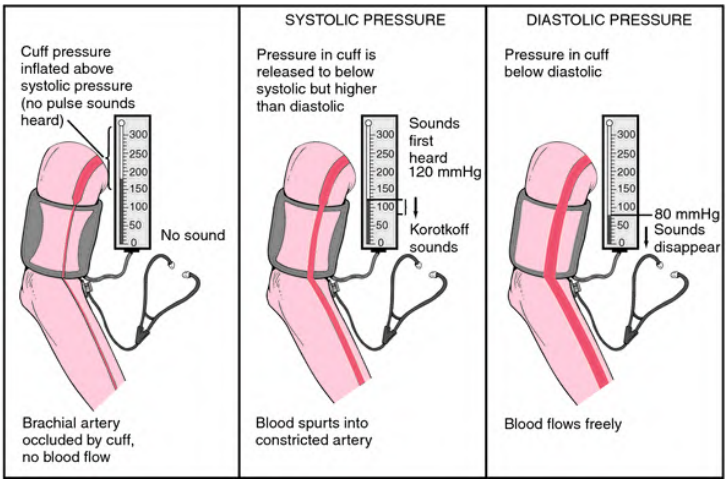
1. *Non-Invasive Blood Pressure* (Tidak langsung)

   Teknik pengukuran darah dengan menggunakan manset lengan atau *sphygmomanometer* dan stetoskop. Terdapat 2 macam *sphygmomanometer* , yaitu air raksa dan otomatis. Stetoskop digunakan untuk mendengar suara Korotkoff saat mengukur tekanan darah. Pada penggunaan manset sphygmomanometer otomatis, stetoskop tidak diperlukan. Pengukuran tekanan darah menggunakan *sphygmomanometer* air raksa dan stetoskop dapat dilihat pada Gambar 2.1. Tekanan sistol dengan cara melihat label tekanan pada air raksa saat mendengar suara pertama kali melalui stetoskop, dan saat suara menghilang menandakan bahwa itulah tekanan diastolnya.

   Meskipun *sphygmomanometer* air raksa yang dinilai lebih baik, tetapi ada faktor yang berbahaya yaitu efek racun dari tumpahan raksa. *Sphygmomanometer* otomatis lebih aman, akan tetapi perlu perawatan seara berkala dan diperlukan tenaga medis yang ahli untuk melakukan pengukuran. Karena peletakan manset yang tidak tepat akan dan pergerakan dari pasien akan mempengaruhi keakuratan dari kinerja alat.

   Selain itu posisi dan ukuran manset harus disesuaikan dengan besarnya lengan pasien, karena ketidak sesuaian

ukuran manset akan mengurangi validitas hasil pengukuran. Banyak hal yang harus diperhatikan seperti memposisikan pasien dengan benar, pemasangan alat, tenaga medis yang ahli, dan masih banyak lagi, dapat dilihat *British Hypertension Society* [8] yang merupakan salah satu *guideline* untuk pengukuran tekanan darah. Pada metode pengukuran ini, kemungkinan *human error* sangat tinggi.



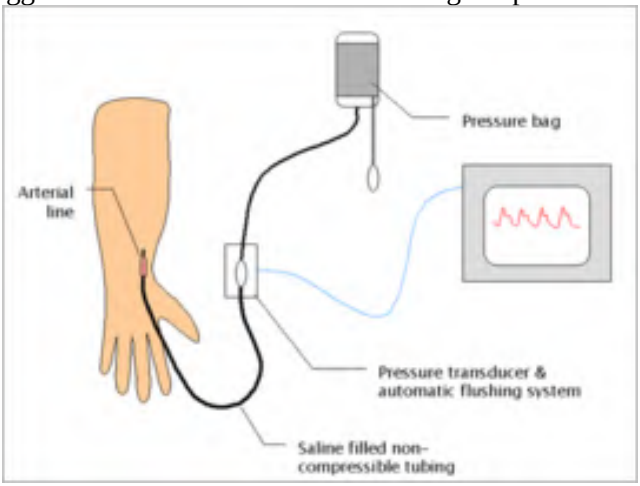
Gambar 2.1. Pengukuran tekanan darah menggunakan sphyromanometer [8]

2. *Invasive Blood Pressure* (Langsung)

Pengukuran tekanan darah secara *invasive* atau langsung dapat dilakukan dengan melakukan sadapan ke dalam arteri yang dihubungkan dengan *tranduser*. Sadapan dapat dilakukan pada bagian tubuh manusia, seperti pada jantung, ujung jari, ujung kaki, dan telinga yang dilalui oleh pembuluh arteri. *Tranduser* akan merubah tekanan hidrostatik menjadi sinyal elektrik dan menghasilkan tekanan sistolik, diastolik, maupun MAP pada layar monitor. Setiap perubahan dari ketiga parameter diatas, kapanpun,dan berapapun maka akan

selalu muncul dilayar monitor. Gambar 2.2 merupakan proses pengukuran tekanan darah secara langsung.

Ketika terjadi penyempitan pembuluh darah berat, dimana *stroke* volume sangat lemah, maka pengukuran dengan *cuff* tidak akurat lagi. Selain itu, karena langsung berhubungan dengan arteri, maka tekanan yang dihasilkan lebih akurat terutama pada pasien yang mengalami hipertensi. Kesalahan pada pengukuran tekanan darah akan berdampak pada kesalahan penanganan yang diberikan. Maka disinilah penggunaan *Invasive Blood Pressure* sangat diperlukan.

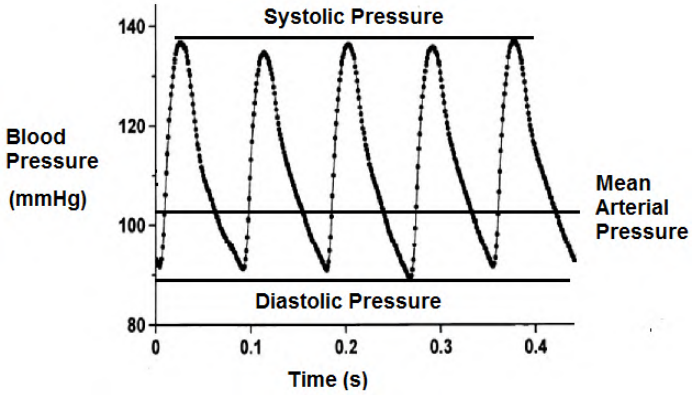


Gambar 2.2. Pengukuran tekanan darah secara langsung [8]

2.2 *Arterial Blood Pressure*

Pembuluh darah terbagi menjadi dua, yaitu arteri dan vena [6]. Tekanan darah juga dapat dibedakan menjadi dua macam, yaitu tekanan darah arteri dan tekanan darah vena. Tekanan darah yang kita kenal sehari-hari adalah tekanan darah arteri. Yang pengukurannya biasa menggunakan alat yang disebut sphygmomanometer atau tensimeter. Sedangkan tekanan darah

vena diukur dengan menggunakan CVP (Central Venous Pressure) atau tekanan vena sentral.



Gambar 2.3. Tekanan darah arteri[6]

Tekanan darah arteri adalah kekuatan tekanan darah ke dinding pembuluh darah yang memompanya. Tekanan ini berubah–ubah pada setiap tahap siklus jantung. Gambar 2.3 menunjukkan bahwa tekanan darah arteri dapat menggambarkan tekanan sistolik, diastolik, dan *Mean Arterial Pressure* (MAP). Selama sistol ventrikel kiri memaksa darah masuk aorta, tekanan naik sampai puncak disebut dengan tekanan sistolik. Selama diastol tekanan turun, nilai terendah yang dicapai disebut tekanan diastolik. Selisih tekanan sistolik dan tekanan diastolik disebut *Pulse Pressure* atau tekanan nadi. Sedangkan *Mean Arterial Pressure* (MAP) adalah tekanan rata – rata selama siklus jantung. MAP bisa didapatkan dengan persamaan 2.1.

$$MAP = \frac{S+2D}{3} \qquad (2.1)$$

Keterangan:

MAP = *Mean Arterial Pressure*
S = Tekanan darah sistolik
D = Tekanan darah diastolik

2.3 Sinyal ECG (*Electrocardiogram*)

Sinyal ECG (*Electrocardiogram*) adalah rekaman aktivitas elektrik yang dihasilkan oleh otot-otot jantung yang mencapai permukaan tubuh. Sinyal ini mampu memberi informasi mengenai kondisi jantung melalui alat elektrokardiograf yang dapat digunakan oleh dokter ahli untuk menentukan kondisi jantung pasien.

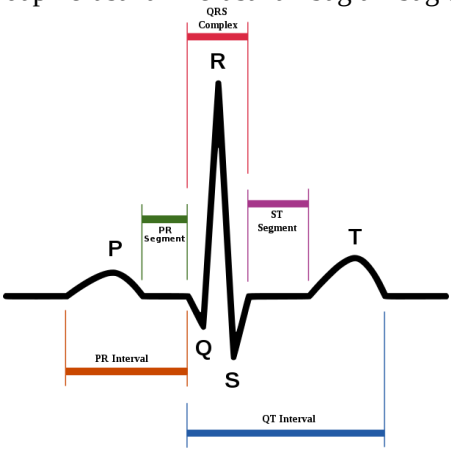
Sebuah sinyal yang didapat dari ECG normal [9] adalah seperti pada Gambar 2.4 gelombang ECG normal memiliki ciri-ciri sebagai berikut :

1. Gelombang P mempunyai amplitudo kurang dari 0,3 mVolt dan periode kurang dari 0,11 detik.
2. Gelombang Q mempunyai amplitudo sebesar minus 25% dari amplitudo gelombang R.
3. Gelombang R mempunyai amplitudo maksimum 3 mVolt
4. Gelombang S merupakan defleksi negatif sesudah gelombang R.
5. Kompleks QRS terdiri dari gelombang Q, R dan S yang memiliki periode 0,06-0,10 detik dengan periode rata-rata 0,08 detik.
6. Gelombang T mempunyai amplitudo minimum 0,1 mVolt.

Salah satu metode pengambilan sinyal ECG yang biasa digunakan untuk menganalisis kondisi kesehatan jantung pasien mengacu pada Clinical Guidlines by Consenses yang dikeluarkan oleh Bristish Cardiovascular Society [10], yaitu dengan menggunakan sepuluh buah elektroda dengan dua belas titik sadapan (12 leads). Sepuluh buah elektroda tersebut dihubungkan ke tubuh manusia yaitu, *Right Arm* (RA), *Left Arm* (LA), *Left Leg* (LL), AVF, *Right Leg* (RL), *Chest 1* (C1), C2, C3, C4, C5 dan C6.

Fungsi sadapan ECG adalah untuk menghasilkan sudut pandang yang jelas terhadap jantung. Sadapan ini dibaratkan dengan banyaknya mata yang mengamati jantungdari berbagai

arah. Semakin banyak sudut pandang, semakin sempurna pengamatan terhadap kerusakan-kerusakan bagian-bagian jantung.



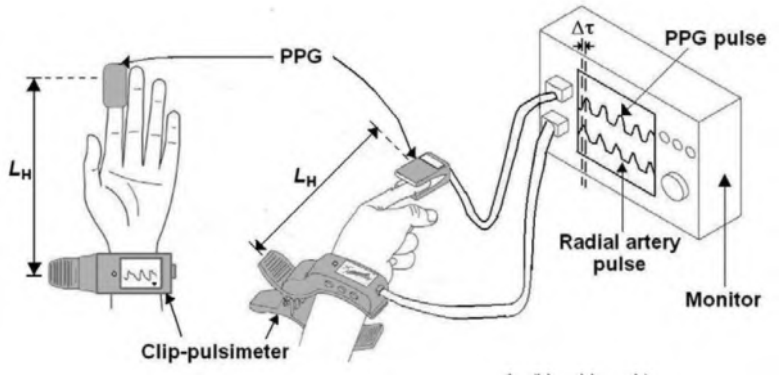
Gambar 2.4. Gelombang ECG normal [9]

2.4 *Photoplethysmograph* (PPG)

Plethysmografi merupakan suatu teknik untuk mendeteksi atau mengukur perubahan volume di dalam suatu organ. *Photoplethysmograph* (PPG) digunakan untuk mengukur kondisi peredaran darah yang dipompa oleh jantung pada organ tertentu dalam tubuh manusia. Pemanfaatan sinyal PPG [11] akan difokuskan pada perhitungan denyut jantung seseorang selama periode tertentu dan penampilan grafik pada LCD (*Liquid Crystal Display*) grafik, sehingga grafik tersebut dapat dimanfaatkan oleh ahli medis untuk mengetahui kondisi jantung seseorang. Biasanya merupakan hasil dari fluktuasi darah atau udara yang terkandung di dalamnya. PPG yang bekerja menggunakan sensoroptik, yaitu *Pulse oximeter*. *Pulse oximeter* merupakan perangkat medis yang dapat mengukur banyaknya kandungan oksigen dalam darah dan perubahan volume darah pada kulit. *Pulse oximeter* dapat dibuat menggunakan LED (*A Light-Emitting Diode*) dan LDR (*Light*

Dependan Resistor). Proses pengambilan sinyal PPG dapat dilihat pada Gambar 2.5.

PPG memiliki kekurangan yaitu hanya dapat mengukur atau mengamati perubahan volume tetapi besaran yang dihasilkan tidak dapat dikalibrasi amplitudonya. Sensor yang digunakan terdiri dari dua jenis, yaitu : *sensor transmitter* dan *sensor receiver*. *Sensor transmitter* yang digunakan adalah LED sedangkan sensor penerima yang digunakan yaitu LDR. Sebagai pemancar, LED memancarkan cahayanya merspon kulit dan peredaran darah yang dipompa oleh jantung yang ada pada jari. Kemudian sebagai penerima, LDR akan merespon adanya denyut jantung yang direspon oleh LED tadi.



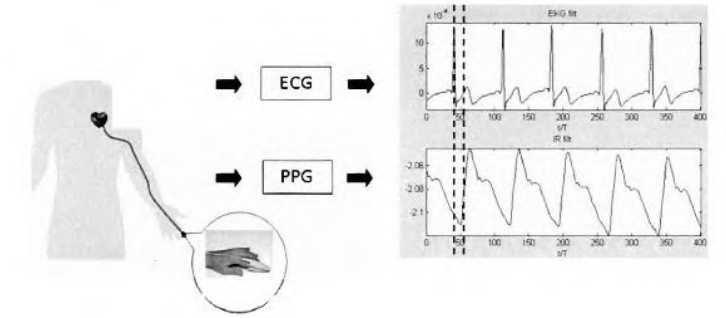
Gambar 2.5. Proses pengambilan sinyal PPG [11]

Sinyal PPG dapat dimanfaatkan dalam bidang kedokteran, seperti menghitung atau monitoring denyut jantung, mengamati kinerja dan kelainan jantung, memonitor pernafasan, dan mengatur kadar oksigen dalam darah.

2.5 *Pulse Transit Time* (PTT)

Pulse Transit Time (PTT) [3] adalah waktu interval antara dua pulsa terdeteksi pada arteri yang dibutuhkan dalam perjalanan dari katup aorta ke bagian tubuh *peripheral* (ujung jari). Gambar

2.6 menggambarkan aliran arteri dari jantung saat memompa darah ke jari tangan dimana jantung dan jari tangan telah dipasang alat untuk memonitor gelombangnya. Aliran dari arteri memberikan sinyal tegangan yang mengalami fluktuasi setiap detak jantungnya. Gelombang yang didapat dari jantung adalah sinyal ECG, sedangkan gelombang yang didapat dari jari tangan adalah sinyal PPG.



Gambar 2.6. Aliran arteri dari jantung ke jari [14]

Puncak gelombang sinyal menandakan denyut aliran yang dihasilkan oleh jantung. Puncak atas sinyal ECG merupakan waktu dimulainya jantung memompa darah yaitu saat darah meninggalkan aorta, sedangkan puncak sinyal PPG merupakan waktu ketika darah sampai pada ujung jari. Dari kedua puncak gelombang sinyal tersebut, dapat dihitung selisih waktunya. Selisih waktu dari puncak sinyal ECG dan PPG menandakan lamanya waktu yang dibutuhkan untuk mengedarkan darah dari jantung ke bagian tubuh tepi yang disebut dengan *Pulse Transit Time* (PTT) .

PTT didefinisikan [2] sebagai waktu yang dibutuhkan antara puncak sinyal ECG (Gelombang puncak R) menuju puncak sinyal PPG. Untuk lebih jelasnya dapat dilihat pada Gambar 2.7. Dari definisi tersebut, PTT dapat dituliskan seperti Persamaan 2.2.

$$PTT = t\ R\_PPG - t\ R\_ECG \quad (2.2)$$

Keterangan :

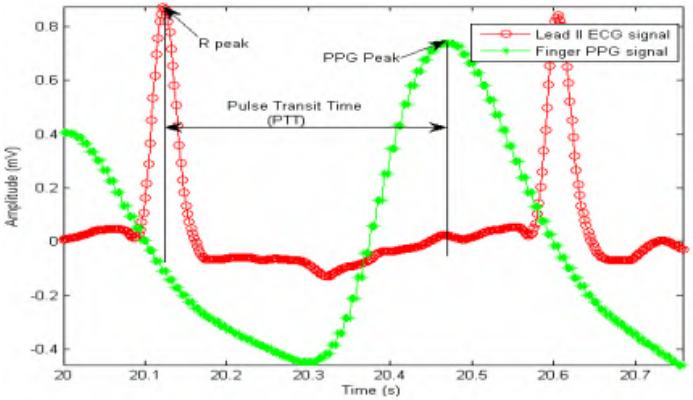
PTT = Pulse Transit Time

t_RPPG = Waktu saat terjadi puncak PPG (detik)
t_RECG = Waktu saat terjadi puncak ECG (detik)

Ketika jantung memompa darah, aliran darah yang diedarkan mempunyai kecepatan untuk sampai ke seluruh tubuh. Untuk beberapa bagian tubuh, seperti ujung jari tangan merupakan organ tubuh yang dapat dideteksi karena dapat menghasilkan gelombang yang kuat. Kecepatan tekanan darah tersebut disebut dengan *Pulse Wave Velocity* (PWV) dimana PWV dapat dihitung dari jarak antara jantung ke jari dibagi dengan nilai PTT[13], hal tersebut dapat dilihat pada Persamaan 2.3.

$$PWV = \frac{(BDC\ x\ Height)}{PTT} \quad (2.3)$$

Keterangan
PWV = Pulse Wave Velocity
BDC = *Body correlation factor*, BDC bernilai 0,5 untuk dewasa
Height = tinggi badan (cm)



Gambar 2.7. Definisi PTT

Apabila kecepatan tekanan darah (PWV) meningkat, menandakan bahwa darah mencapai situs perifer (jari tangan) dari katup aorta dalam waktu yang singkat sehingga waktu transit PTT yang kecil. Menurut penelitian sebelumnya[13], PWV dapat

berdampak pada tekanan darah seseorang. Apabila PWV berubah makan waktu untuk jantung berkontraksi dan berelaksasi akan berubah juga sehingga tekanan sistoal dan diastolnya juga akan berubah. digunakan untuk mencari nilai tekanan sistolik dan diastolik melalui persamaan linear. Semakin tinggi kecepatan tekanan darah maka tekanan darahnya juga semakin tinggi, hal itu dikarenakan jantung lebih cepat dalam melakukan pemompaan darah.

Untuk mengetahui besarnya tekanan darah dapat dicari dengan Persamaan 2.4

$$Tekanan\ darah = a * PWV + b \quad (2.4)$$

Dimana untuk tekanan sistolik, a bernilai 0.05089855 dan b bernilai 62.5590972 sedangkan untuk mengukur tekanan diastolik a bernilai 0.04940772 dan b bernilai 17.4800472

2.6 Transformasi Wavelet

Transformasi merupakan proses pengubahan data atau sinyal ke dalam bentuk lain agar lebih mudah dianalisis, seperti transformasi fourier yang mengubah sinyal ke dalam beberapa gelombang *sinus* atau *cosinus* dengan frekuensi yang berbeda, sedangkan transformasi wavelet (*wavelet transform*) mengubah sinyal ke dalam berbagai bentuk wavelet basis (*mother wavelet*) dengan berbagai pergeseran dan penyekalaan.

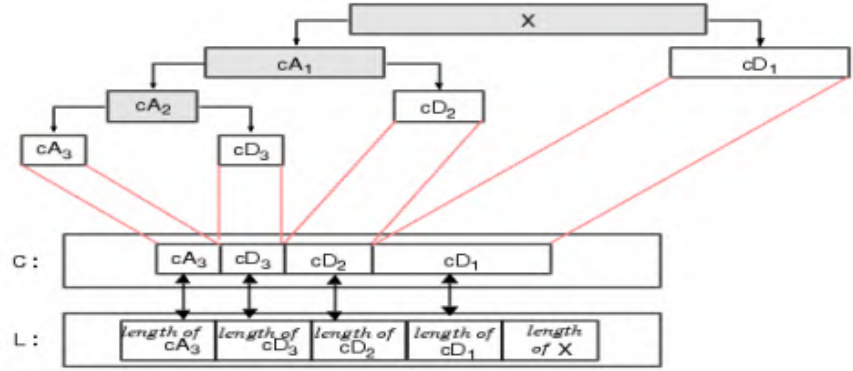
Proses transformasi wavelet [3] dapat dilakukan dengan konvolusi atau dengan proses pererataan dan pengurangan secara berulang. Proses ini banyak digunakan pada proses dekomposisi, deteksi, pengenalan *(recognition)*, pengambilan kembali citra *(image retrieval)*, dan lainnya yang masih dalam penelitian. Wavelet adalah fungsi matematika yang digunakan untuk merepresentasikan data atau suatu fungsi.

Ada berbagai jenis transformasi wavelet, akan tetapi pada bagian ini lebih menitikberatkan pada transformasi *Discrete Wavelet Transform* (DWT) 1-dimensi (1-D). Transformasi *wavelet* 1-D membagi sinyal menjadi dua bagian, frekuensi tinggi dan

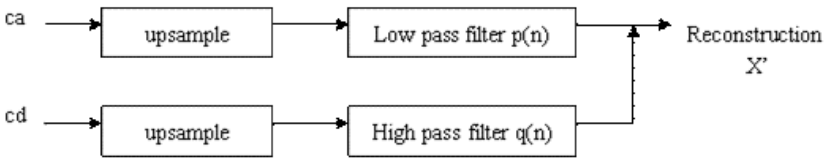
frekuensi rendah berturut-turut dengan tapis lolos-rendah (*low-pass filter*) dan tapis lolos tinggi (*high-pass filter*). Pada gambar 2.8. dijelaskan fungsi $g(n)$ adalah impuls setelah melalui *low-pass filter*. Dengan mengkonvolusikan $x(k)$ dengan $g(n)$ diperoleh koefisien aproksimasi. Variabel $h(n)$ adalah fungsi impulse setelah melalui *high-pass filter*. Dengan mengkonvolusikan $x(k)$ dengan $h(n)$ diperoleh koefisien detail.

Proses dekomposisi multilevel wavelet dapat dilihat pada Gambar 2.9, frekuensi rendah dibagi kembali menjadi frekuensi tinggi dan rendah dengan kata lain koefisien aproksimasi level 1 dipecah menjadi koefisien aproksimasi level 2 dan koefisien detail level 2 dan seterusnya. Proses diulang sampai sinyal tidak dapat didekomposisi lagi atau sampai pada level yang memungkinkan. Hasil analisis DWT adalah koefisien aproksimasi dan detail. Kelebihan dari analisis sinyal menggunakan wavelet dapat dipelajari karakteristik sinyal secara lokal dan detail, sesuai dengan skala-nya.

Gambar 2.8. Diagram analogi pembagian koefisien aproksimasi dan detail [3]



Gambar 2.9. Dekomposisi wavelet [3]



Gambar 2.10. Rekonstruksi wavelet [3]

Pada transformasi DWT terdapat proses pengembalian kembali komponen-komponen yang telah kita gunakan. *Invers Discrete Wavelet Transform* (IDWT) merupakan kebalikan dari *transformasi wavelet diskrit* (DWT). Pada transformasi ini dilakukan proses rekonstruksi sinyal, yaitu mengembalikan komponen frekuensi menjadi komponen sinyal semula.

Dalam satu sistem transformasi wavelet menggunakan empat macam filter, yaitu *low-pass filter* dan *high-pass filter* dekomposisi, dan *low-pass filter* dan *high-pass filter* rekonstruksi. *Low-pass filter* dan *high-pass filter* dekomposisi digunakan pada saat proses dekomposisi atau pemecahan sedangkan *low-pass filter* dan *high-pass filter* rekonstruksi digunakan untuk proses rekonstruksi wavelet menggunakan. Proses rekonstruksi wavelet dapat dilihat pada Gambar 2.10.

2.7 Normalisasi Data

Normalisasi data diperlukan ketika data yang ada bernilai terlalu besar maupun terlalu kecil sehingga pengguna kesulitan dalam memahami informasi yang dimaksud [15]. Tidak hanya untuk pengguna, terkadang nilai yang signifikan dapat mempersulit pemilik data dalam melakukan proses pengolahan untuk disajikan kepada pengguna. Berikut adalah salah satu cara menormalisasi data ke dalam suatu rentang nilai dengan tidak mengurangi bobot nilai sebenarnya yang dinamakan *Min-Max Scalling*. Normalisasi *Min-Max Scalling* akan menghasilkan rentang 0 hingga 1, dengan menggunakan Persamaan 2.4

$$normalisasi(x) = \frac{x - minValue}{maxValue - minValue} \quad (2.4)$$

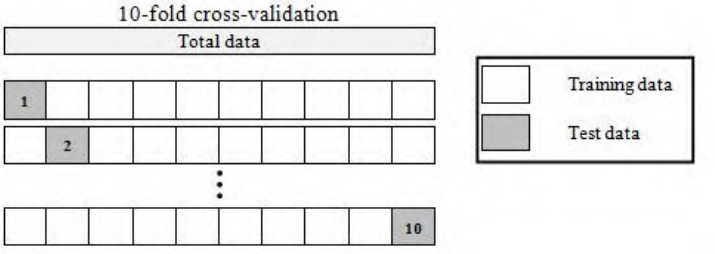
Keterangan :

x = Nilai yang akan dinormalisasi
minValue = Nilai terendah dari data set
maxValue = Nilai tertinggi dari data set

2.8 *K-Fold Cross Validation*

Cross Validation merupakan salah satu teknik untuk menilai atau memvalidasi keakuratan sebuah model yang dibangun berdasarkan *dataset* tertentu. Pembuatan model biasanya bertujuan untuk melakukan prediksi maupun klasifikasi terhadap suatu data baru yang boleh jadi belum pernah muncul di dalam *dataset*. Data yang digunakan dalam proses pembangunan model disebut data *training*, sedangkan data yang akan digunakan untuk memvalidasi model disebut sebagai data *testing*.

Salah satu metode *cross-validation* yang populer adalah *K-fold cross validation* [12]. Dalam teknik ini *dataset* dibagi menjadi sejumlah K-buah partisi secara acak. Kemudian dilakukan sejumlah K-kali eksperimen, dimana masing-masing eksperimen menggunakan data partisi ke-K sebagai data *testing* dan memanfaatkan sisa partisi lainnya sebagai data *training*. Untuk lebih jelasnya dapat dilihat pada Gambar 2.11, misalnya telah ditentukan *K* = 10 maka pembagian data *training* dan *testing* adalah seperti pada gambar tersebut.

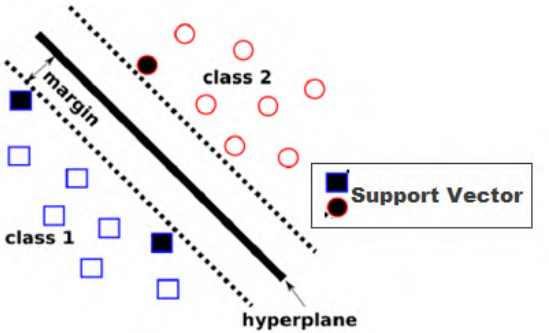


Gambar 2.11. Algoritma *K-fold Cross Validation[12]*

2.9 *Support Vector Machines(SVM)*

Support Vector Machine (SVM) adalah suatu teknik untuk melakukan prediksi, baik dalam kasus klasifikasi maupun regresi. SVM berada dalam satu kelas dengan Artificial Neural Network (ANN) dalam hal fungsi dan kondisi permasalahan yang bisa diselesaikan. Keduanya masuk dalam kelas *supervised learning.* 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 proses *learning* yang berasal dari teori pembelajaran statistik[12].

Konsep SVM dapat dijelaskan secara sederhana [16] sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah kelas pada *input space*. Gambar 2.12 menggambarkan konsep dasar dari SVM. Input data memiliki pola tersendiri (*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 SVM.



Gambar 2.12. Konsep dasar SVM[16]

Data yang tersedia dinotasikan sebagai $x_i \in \Re^d$ sedangkanlabel masing-masing dinotasikan $y_i \in \{-1,+1\}$ untuk i = 1,2,… ,l , dimana l adalah banyaknya data. Diasumsikan kedua kelas -1 dan +1 dapat terpisah secara sempurna oleh hyperplane berdimensi d, yang didefinisikan pada Persamaan 2.5

$$w \,.\, x + \mathrm{b} = 0 \qquad (2.5)$$

Pattern x_i yang ternasuk kelas -1 (sampel negatif) dapat dirumuskan sebagai *pattern* yang memenuhi Persamaan 2.6

$$w \,.\, x + \mathrm{b} \leq -1 \qquad (2.6)$$

Sedangkan *pattern* xi yang termasuk kelas +1 (sampel positif) memenuhi Persamaan 2.7

$$w \,.\, x + \mathrm{b} \geq +1 \qquad (2.7)$$

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara *hyperplane* dan titik terdekatnya, yaitu 1/ ‖w‖. Hal ini dapat dirumuskan sebagai *Quadratic Programming (QP) Problem*, yaitu mencari titik minimal Persamaan 2.8 dengan memperhatikan *constraint* persamaan 2.9.

$$min\tau(w) = \frac{1}{2}||w|_{\mathbf{2}} \qquad (2.8)$$

$$yi\ (x\ i.w + b) - 1 \geq\ 0, \forall i \qquad (2.9)$$

Problem ini dapat dipecahkan dengan berbagi teknik komputasi, diantaranya dengan *Lagrange Multiplier* dapat dilihat pada Persamaan 2.10.

$$min\tau(w) = \frac{1}{2}||w||^2 - \sum_{i=1}^{I} \alpha i\ (yi\ ((\ x\ i\ .wi\ + b) - 1)) \qquad (2.10)$$

α_i adalah *Lagrange multipliers*, yang bernilai nol atau positif ($\alpha_i \geq 0$). Nilai optimal dari Persamaan 2.10 dapat dihitung dengan meminimalkan L terhadap w dan b, dan memaksimalkan L terhadap α_i . Dengan memperhatikan sifat bahwa pada titik optimal gradient L = 0, Persamaan 2.10 dapat dimodifikasi sebagai maksimalisasi problem yang hanya mengandung α_i , sebagaimana Persamaan 2.11.

$$\sum_{i,j=1}^{I} \alpha i\ - \frac{1}{2}\sum_{i,j=1}^{I} \alpha i\ \alpha j\ yi\ yj\ xi\ xj \qquad (2.11)$$

Dengan *constraint* :

$$\alpha i \geq 0 (i = 1,2,3,\ldots,l) \sum_{i=1}^{I} \alpha i \, vi = 0$$

Dari hasil perhitungan ini diperoleh α_i yang kebanyakan bernilai positif. Data yang berkorelasi dengan α_i yang positif inilah yang disebut sebagai *support vector*. Penjelasan tersebut berdasarkan asumsi bahwa kedua kelas dapat terpisah secara sempurna oleh *hyperplane (linear separable)*. Akan tetapi, pada umumnya dua kelas pada *input space* tidak dapat terpisah secara sempurna (*non linear separable*). Hal ini menyebabkan *constraint* pada Persamaan 2.11 tidak dapat terpenuhi, sehingga optimisasi tidak dapat dilakukan. Untuk mengatasi masalah ini, SVM dirumuskan ulang dengan memperkenalkan *teknik softmargin*.

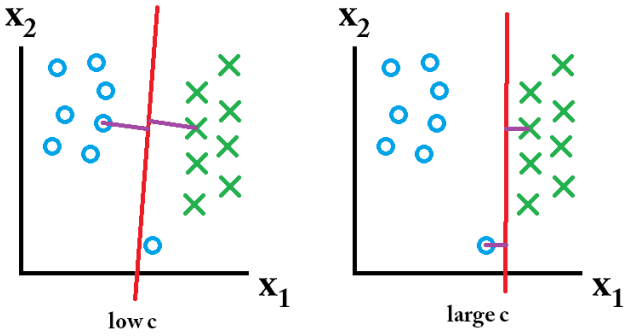
Dalam *softmargin*, Persamaan 2.9 dimodifikasi dengan memasukkan *slack variable* ξ_i ($\xi > 0$) menjadi Persamaan 2.12.

$$y_i (x_i . w + b) \geq 1 - \xi_i \, \forall_i \qquad (2.12)$$

Dengan demikian Persamaan 2.8 diubah menjadi Persamaan 2.13.

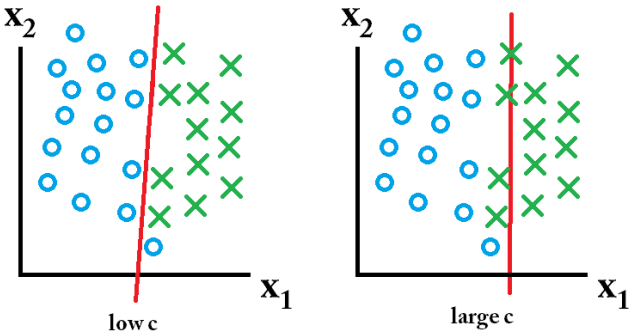
$$min\tau(w, \xi) = \frac{1}{2} ||w|_2 + C \sum_{i=1}^{I} \xi i \qquad (2.13)$$

C adalah parameter yang menentukan besar penalti akibat kesalahan dalam klasifikasi data dan nilainya ditentukan oleh pengguna [c2]. Parameter *C* dipilih untuk mengontrol *tradeoff* antara margin dan *error* klasifikasi ξ. Sebenarnya dalam SVM dicari 2 poin utama yaitu hyperlane dengan margin minimal terbesar, dan hyperlane yang benar-benar dapat memisahkan kelas dengan eror minimal. Permasalahannya adalah terkadang tidak bisa mendapatkan kedua poin diatas. Parameter *C* dapat disesuaikan untuk mendapatkan atau mendekati hasil yang maksimal. Gambar 2.13 menunjukkan bahwa pada parameter *C* yang rendah memberikan margin yang cukup besar (warna ungu) dan terdapat fitur yang tidak terklasifikasi dengan benar. Sedangkan pada parameter *C* tinggi, margin yang dihasilkan lebih kecil dan data terklasifikasi dengan benar.



Gambar 2.13. Perbandingan *C* parameter data A [16]

Akan tetapi, parameter *C* yang tinggi belum tentu lebih baik. Hal tersebut bergantung pada *dataset* yang ada. Misalnya pada Gambar 2.14 saat menggunakan parameter *C* tinggi justru membuat akurasi dari klasifikasi berkurang, banyak kelas yang tidak terklasifikasi dengan benar. Parameter *C* harus disesuaikan dengan dataset dengan melalui uji coba terlebih dahulu.



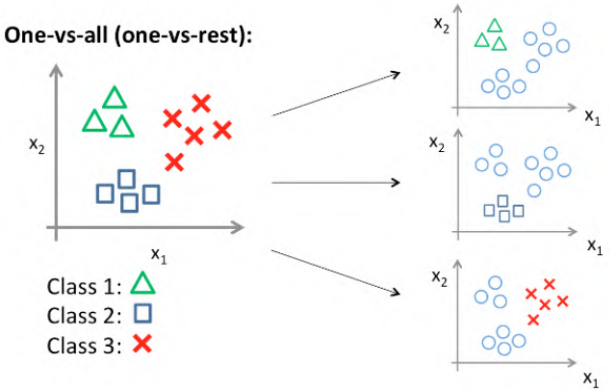
Gambar 2.14. Perbandingan *C* parameter data B [16]

Mengubah parameter *C* kemungkinan akan menghasilkan hyperplane yang berbeda atau bisa juga sama. Jika tidak menghasilkan hyperplane yang berbeda, tidak berarti bahwa *classifier* akan menampilkan kelas yang berbeda untuk data

tertentu, bisa saja marginnya berkurang atau justru malah bertambah. Anda telah menggunakannya untuk mengklasifikasikan.

Untuk mengimplementasikan multi kelas SVM terdapat dua pilihan yaitu dengan menggabungkan beberapa SVM biner atau menggabungkan semua data yang terdiri dari beberapa kelas ke dalam sebuah bentuk permasalahan optimal. Namun, pada pendekatan kedua permasalahan optimasi harus diselesaikan jauh lebih rumit. Berikut ini adalah metode yang umum digunakan untuk mengimplementasikan multiclass SVM dengan pendekatan yang pertama:

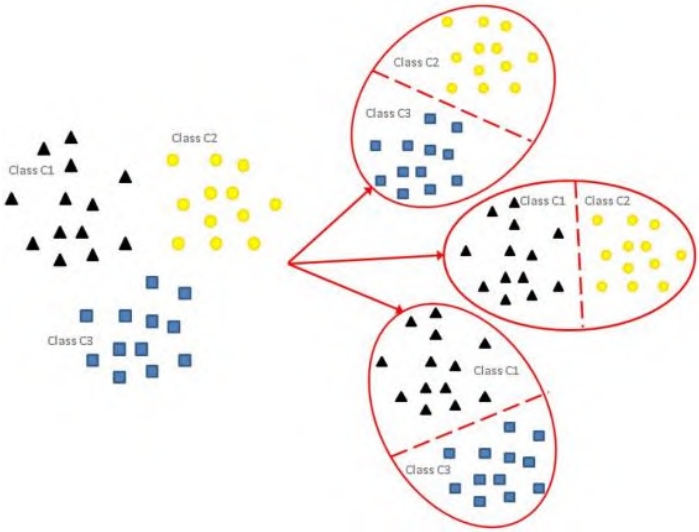
1. Metode *one against all* (satu lawan semua). Dengan menggunakan metode ini, dibangun *k* buah model SVM biner sesuai dengan jumlah kelasnya. Gambar 2.15 menunjukkan penggunaan metode *one against all* pada 3 kelas.



Gambar 2.15. SVM *one against all [16]*

2. Metode *one against one* (satu lawan satu). Dengan menggunakan metode ini, dibagun $k(k\text{-}1)/2$ buah model klasifikasi biner (*k* adalah jumlah kelas). Terdapat beberapa metode untuk melakukan pengujian setelah keseluruhan model klasifikasi $k(k\text{-}1)/2$ selesai dibangun. Gambar 2.15

menunjukkan penggunaan metode *one against one* dengan jumlah kelas sebanyak 3.



Gambar 2.16. SVM *one against one [16]*

Pada mulanya teknik *machine learning* dikembangkan dengan asumsi kelinearan. Sehingga algoritma yang dihasilkan terbatas untuk kasus-kasus yang linear saja. Akan tetapi untuk menghadapi kasus yang tidak linear maka dapat menggunakan bantuan berbagai macam fungsi *kernel*. *Kernel* memberikan berbagai kemudahan, karena dalam proses pembelajaran SVM, untuk menentukan *support vector*, maka cukup dengan mengetahui fungsi *kernel* yang dipakai, dan tidak perlu mengetahui wujud dari fungsi non-linear. Menurut Karatzouglou [12], ada beberapa fungsi *kernel* yang sering digunakan dalam literatur SVM antara lain sebagai berikut:

1. *Kernel* linear adalah *kernel* yang paling sederhana dari semua fungsi *kernel*. *Kernel* ini biasa digunakan dalam kasus klasifikasi teks.

2. *Kernel* Radial Basis Function (RBF) adalah *kernel gaussian* yang umum digunakan untuk data yang sudah valid dan merupakan *default* dalam *tools* SVM.
3. *Kernel Polynomial* adalah *kernel* yang sering digunakan untuk klasifikasi gambar.
4. *Kernel Tangent Hyperbolic* (Sigmoid) adalah *kernel* yang sering digunakan untuk *neural networks*.

2.10 *Confusion matrix*

Confusion matrix adalah suatu metode yang biasa digunakan untuk melakukan perhitungan akurasi pada konsep data mining [12]. *Confusion Matrix* memiliki informasi hasil prediksi dan aktual pada data yang telah di klasifikasi. Performa suatu sistem biasanya dievaluasi menggunakan *Confusion Matrix*. Tabel 2.2 menunjukkan *Confusion Matrix* pada dua kelas.

Tabel 2.2 Confusion Matrix dua kelas

		PREDIKSI	
		1	**0**
AKTUAL	**1**	TP	FN
	0	FP	TN

Nilai yang bisa dihitung menggunakan *Confusion Matrix* yaitu akurasi, sensitivitas dan sensitivitas. Akurasi merupakan hasil bagi dari jumlah prediksi yang terklasifikasi secara benar dibagi total data yang diklasifikasi seperti pada persamaan 2.14.

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.14}$$

Sensitivitas adalah perbandingan dari jumlah data TP dengan total data TP dan FN seperti pada persamaan 2.15.

$$Sensitivitas = \frac{TP}{TP+FN} \tag{2.15}$$

Spesifisitas adalah perbandingan dari jumlah data TN dengan total data TN dan FP seperti pada persamaan 2.16.

$$Spesifisitas = \frac{TN}{TN+FP} \quad (2.16)$$

BAB III
PERANCANGAN PERANGKAT LUNAK

Bab ini membahas mengenai perancangan dan pembuatan sistem perangkat lunak. Sistem perangkat lunak yang dibuat pada Tugas Akhir ini adalah mengolah data sinyal ECG dan PPG menggunakan *Discrete Wavelet Transform* (DWT) untuk *preprocessing* kemudian dilakukan ekstraksi fitur untuk mendapatkan nilai *Pulse Trasit Time* (PTT) dan puncak atas bawah sinyal ECG dan PPG untuk proses klasifikasi menggunakan *Support Vector Machines* (SVM) untuk mengetahui apakah pasien normal, prehipertensi, stadium 1 hipertensi, atau stadium 2 hipertensi. Pada data tekanan ABP dijadikan *ground truth* untuk pengklasifikasi tekanan darah dengan berdasar tekanan sistol dan diastol yang didapat dari pendapat ahli atau dokter yang mengacu pada Tabel 2.1.

3.1 Data

Pada sub bab ini akan dijelaskan mengenai data yang digunakan sebagai masukan perangkat lunak untuk selanjutnya diolah dan dilakukan pengujian sehingga menghasilkan data keluaran yang diharapkan.

3.1.1 Data Masukan *Preprocessing*

Data masukan adalah data yang digunakan sebagai masukan awal dari sistem. Data yang digunakan dalam perangkat lunak klasifikasi tekanan darah adalah data sinyal ECG, PPG, dan ABP yang diunduh dari *website* UCI *Machine Learning Repository* [5]. Dari data yang diunduh terdapat 67 individu, dimana masing-masing individu memiliki potongan rekaman sinyal yang bermacam-macam. Dari 67 individu tersebut, diambil 60 individu dengan masing-masing satu potongan rekaman sinyal. Sehingga didapat 60 potongan sinyal dari 60 individu.

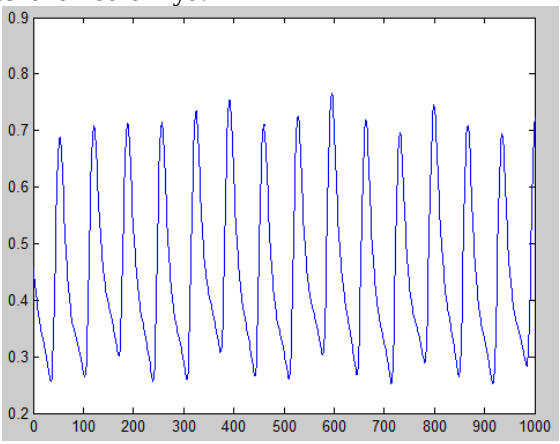
Satu potongan sinyal memiliki matriks 3x1000, dapat dilihat pada Gambar 3.1. Baris pertama data adalah sinyal ECG, Baris kedua adalah tekanan ABP, dan baris ketiga adalah sinyal PPG.

3x1000 double

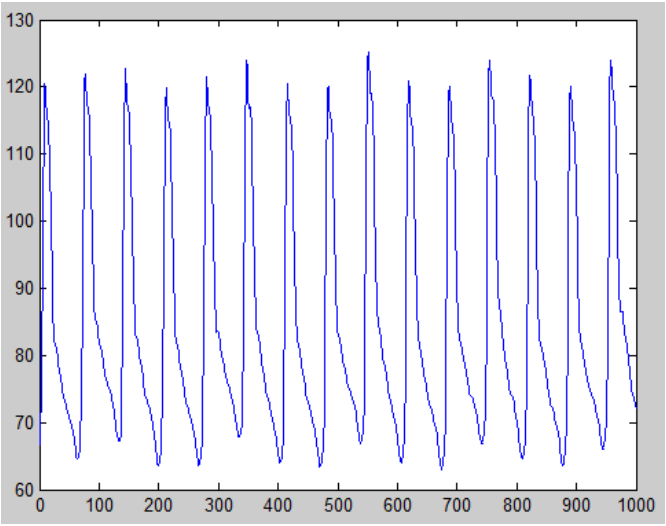
		1	2	3	4	5
ECG	1	0.5777	0.5552	0.5318	0.5103	0.4917
ABP	2	82.4511	82.1382	81.8253	81.8253	81.8253
PPG	3	0.5745	0.5647	0.5510	0.5353	0.5294

Gambar 3.1. Data rekaman sinyal pada satu pasien

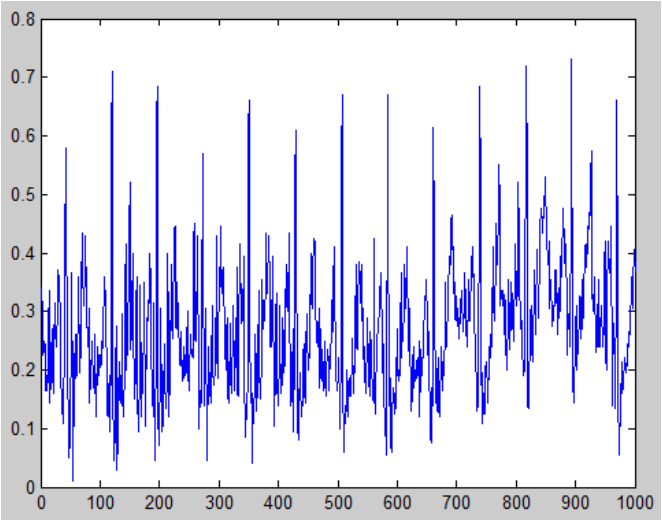
Agar data pada Gambar 3.1 lebih mudah dibaca, maka divisualisasikan menggunakan fungsi *plot* pada matlab sehingga tampak pergerakan naik turunnya gelombang. Visualisasi sinyal dapat ditunjukkan oleh Gambar 3.2 untuk sinyal ECG, Gambar 3.3 untuk tekanan ABP, dan Gambar 3.4 untuk sinyal PPG. Pada Gambar 3.2 dan Gambar 3.4, koordinat *x* menyatakan jumlah kolom, dan koordinat *y* adalah amplitudonya. Sedangkan pada Gambar 3.3 koordinat *x* menyatakan jumlah kolom, dan koordinat *y* adalah tekanan darahnya.



Gambar 3.2. Visualisasi sinyal ECG



Gambar 3.3. Visualisasi tekanan ABP



Gambar 3.4. Visualisasi sinyal PPG

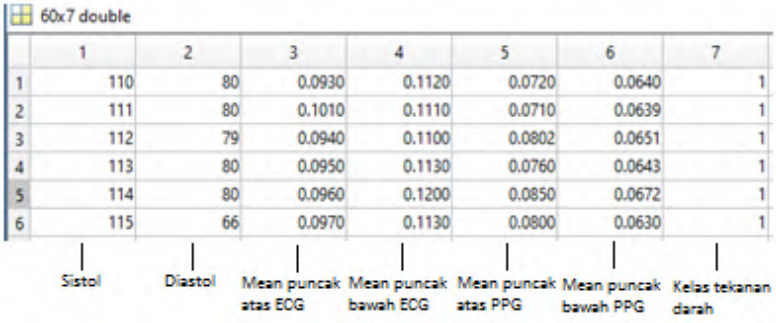
Gambar 3.2 dan Gambar 3.4 merupakan data masukan untuk preprocessing sinyal yang masih memiliki banyak *noise*, sehingga nilai puncak sinyal tidak bisa langsung dicari. Oleh sebab itu *noise* dari data ini perlu dihilangkan terlebih dahulu pada tahap *preprocessing*. Setelah *noise* dihilangkan, maka proses selanjutnya adalah melakukan pendeteksian puncak sinyal yang digunakan untuk penghitungan *Pulse Transit Time*. Indeks dari puncak-puncak ECG dan PPG akan disimpan, kemudian akan kurangkan untuk menghitung selisih jarak indeksnya. Lalu dikonversikan ke satuan detik dengan mengalikannya dengan 0,008 detik (satu kolom mewakili 0,008 detik). Sehingga didapatkan nilai PTT untuk proses selanjutnya

3.1.2 Data Masukan Klasifikasi

Data masukan proses klasifikasi adalah hasil dari proses ekstraksi fitur ditambah dengan pelabelan kelas tekanan darah. Pelabelan kelas didapat dilihat dari visualisasi tekanan ABP seperti pada gambar Gambar 3.3. Di sinilah diperlukan dokter untuk melakukan pembacaan tekanan darah. Puncak atas sinyal menunjukkan tekanan sistolik, dan puncak bawah sinyal menunjukkan tekanan diastoliknya. Dokter melakukan pengklasifikasian data dengan melihat kisaran tekanan sistol dan diastol berdasarkan visualisasi ABP seperti Gambar 3.4 dengan patokan data JNC 7 pada Tabel 2.1.

Dari pendapat dokter, diambil 60 rekaman data pasien dengan rincian 15 dari pasien normal, 15 pasien prehipertensi, 15 pasien stadium 1 hipertensi, dan 15 pasien stadium 2 hipertensi. Pelabelan kelas dari dokter tersebut dimasukkan ke dalam matriks bersama 6 fitur hasil ekstraksi data yaitu tekanan sistol, diastol, rata-rata puncak atas ECG, rata-rata puncak bawah ECG, rata-rata puncak atas PPG, dan rata-rata puncak bawah PPG. Sehingga data masukan untuk proses klasifikasi memiliki matriks 60x7.

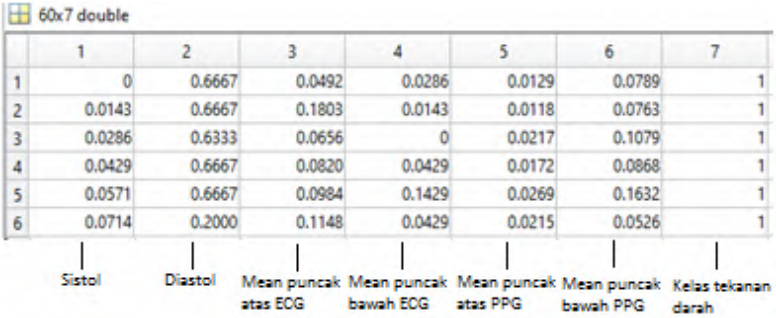
Pada Gambar 3.5, kolom 1-6 merupakan data hasil ekstraksi fitur. Kolom ke-1 adalah tekanan sistol, kolom ke-2 tekanan diastol, kolom ke-3 adalah rata-rata puncak atas ECG, kolom ke-4 adalah rata-rata puncak bawah ECG, kolom ke-5 adalah rata-rata puncak atas PPG, dan kolom ke-6 adalah rata-rata puncak PPG. Sedangkan kolom ke-7 adalah label kelas tekanan darah yang didapat dari dokter.



60x7 double

	1	2	3	4	5	6	7
1	110	80	0.0930	0.1120	0.0720	0.0640	1
2	111	80	0.1010	0.1110	0.0710	0.0639	1
3	112	79	0.0940	0.1100	0.0802	0.0651	1
4	113	80	0.0950	0.1130	0.0760	0.0643	1
5	114	80	0.0960	0.1200	0.0850	0.0672	1
6	115	66	0.0970	0.1130	0.0800	0.0630	1

Gambar 3.5. Data masukan proses klasifikasi sebelum dinormalisasi



60x7 double

	1	2	3	4	5	6	7
1	0	0.6667	0.0492	0.0286	0.0129	0.0789	1
2	0.0143	0.6667	0.1803	0.0143	0.0118	0.0763	1
3	0.0286	0.6333	0.0656	0	0.0217	0.1079	1
4	0.0429	0.6667	0.0820	0.0429	0.0172	0.0868	1
5	0.0571	0.6667	0.0984	0.1429	0.0269	0.1632	1
6	0.0714	0.2000	0.1148	0.0429	0.0215	0.0526	1

Gambar 3.6. Data masukan proses klasifikasi setelah dinormalisasi

Karena data fitur (kolom 1-6) memiliki rentang nilai yang tinggi, maka dibutuhkan penyekalaan untuk memudahkan komputasi dan menghasilkan nilai yang optimal. Pada kolom 1-6 dilakukan normalisasi dengan metode *Min-Max Scalling* untuk

memetakan data dari rentang 0 hingga 1. Hasil proses normalisasi dapat dilihat pada Gambar 3.6 yang selanjutnya akan dipecah menjadi data *training* dan *testing* menggunakan *K-fold Cross Validation*.

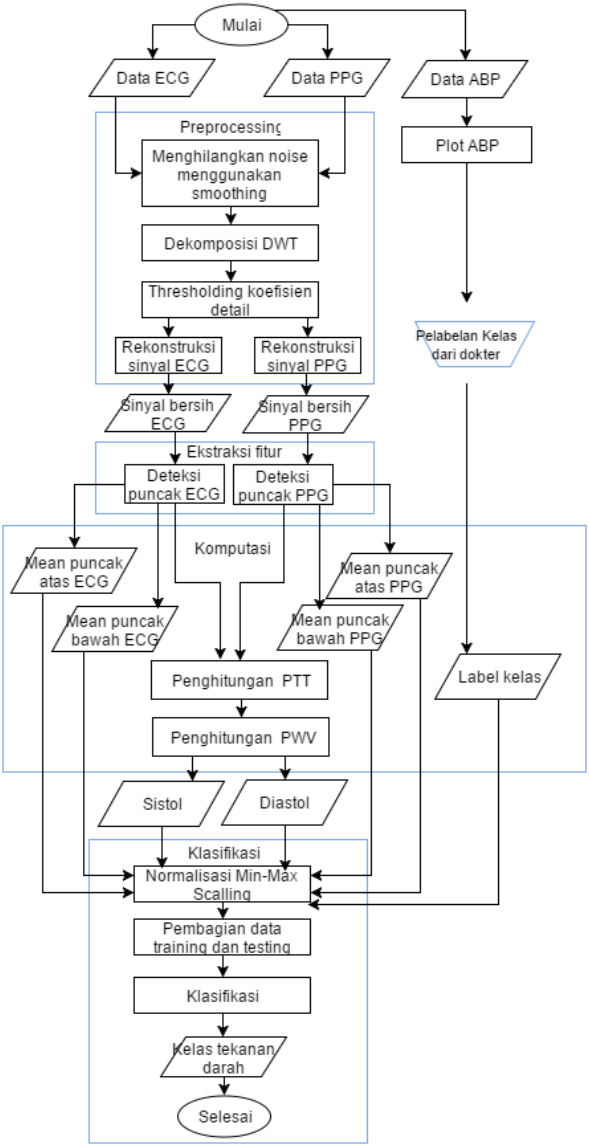
3.1.3 Data Keluaran

Data masukan akan diproses dengan menggunakan *Discrete Wavelet Transform* (DWT) dengan daubachies 6 dengan dekomposisi level 8 kemudian diklasifikasikan menggunakan *Support Vector Machines* (SVM). Hasil dari proses klasifikasi adalah nama kelas pada masing-masing data pengujian disertai dengan besar akurasinya.

3.2 Desain Umum Sistem

Rancangan perangkat lunak klasifikasi tekanan darah menggunakan *Discrete Wavelet Transform* (DWT) dan *Support Vector Machines* (SVM) mempunyai 4 proses utama yaitu *preproccessing*, ekstraksi fitur, komputasi fitur, dan klasifikasi. Gambar 3.7 menjelaskan mulai dari *preprocessing* data yaitu menghilangkan *noise* dari data sinyal ECG dan PPG dengan menggunakan *smoothing* sinyal. Selanjutnya sinyal diproses menggunakan *Discrete Wavelet Transform* (DWT). Kemudian direkonstruksi kembali untuk proses ekstraksi fitur. Hasil dari *preprocessing* adalah sinyal yang sudah bersih dari *noise.*

Setelah sinyal bersih didapat, tahapan selanjutnya adalah ekstraksi fitur. Dilakukan pendeteksian puncak atas dan puncak bawah pada sinyal ECG dan PPG. Puncak atas ECG dan PPG digunakan untuk menghitung nilai Pulse Transit Time (PTT). Setelah didapat PTT maka dilakukan komputasi untuk menghitung kecepatan tekanan darah (PWV). Setelah PWV diketahui, maka dapat menghitung nilai tekanan sistol dan diastolnya. Selain itu juga dilakukan komputasi penghitungan rata-rata puncak atas dan bawah sinyal ECG dan PPG.



Gambar 3.7. Desain gambaran umum sistem

Fitur yang dihasilkan dari proses komputasi berjumlah 6, yaitu tekanan sistolik, tekanan diastolik, puncak atas ECG, puncak bawah ECG, puncak atas PPG, dan puncak bawah PPG. Untuk meratakan skala atau bobot, maka diakukan normalisasi dengan rentang nilai 0 hingga 1. Setelah data dinormalisasi, data dibagi menjadi 2 bagian, yaitu data *training* dan *testing*.

Pembagian data *training* dan *testing* dilakukan dengan menggunakan *K-fold Cross Validation* yang merupakan metode yang paling sering digunakan. Dalam metode ini *dataset* dibagi menjadi sejumlah K-buah partisi secara acak. Kemudian dilakukan sejumlah K-kali eksperimen, dimana masing-masing eksperimen menggunakan data partisi ke-K sebagai data *testing* dan memanfaatkan sisa partisi lainnya sebagai data *training*.

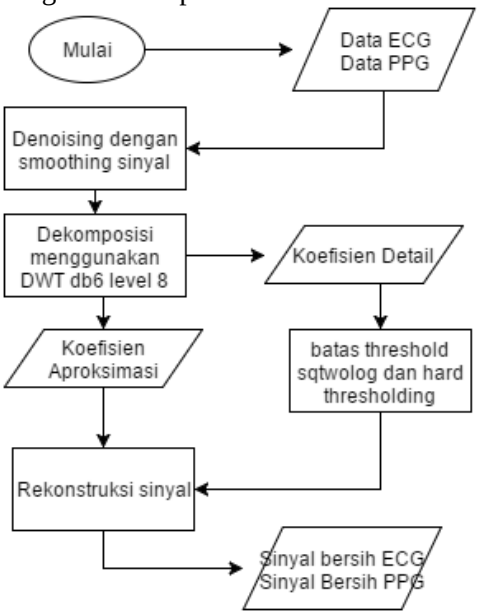
Pada data yang diujikan memiliki 4 kelas sehingga menggunakan multi kelas *Support Vector Machines* (SVM) *ones against all*. Proses klasifikasi dimulai dengan proses *learning* pada data *training*, yaitu melalui pemodelan biner pada masing masing kelas. Kemudian data *testing* diproses dengan membandingkan dengan model yang ada sehingga menghasilkan prediksi kelas dari data *testing*. Keluaran dari sistem ini adalah label kelas dari data *testing*.

3.3 Preprocessing

Preprocessing merupakan cara untuk mengubah data menjadi bentuk yang cocok untuk proses klasifikasi. Pada data sinyal ECG dan PPG *preprocessing* dimaksudkan untuk menghilangkan *noise* untuk mendapatkan sinyal yang lebih bersih. *Dataset* sinyal ECG dan PPG memiliki noise pada rentang tertentu sehingga harus dilakukan *filterisasi* yaitu pemberian ambang batas frekuensi yang harus dihilangkan.

Sebelum dilakukan *preprocessing*, data sinyal ECG dan PPG dipisah terlebih dahulu. Pada *preprocessing* terdapat beberapa 4 tahapan proses utama seperti *denoising*, dekomposisi *Discrete*

Wavelet Transform, *thresholding*, dan rekonstruksi sinyal untuk lebih jelasnya digambarkan pada Gambar 3.8.



Gambar 3.8. Langkah preprocessing

3.3.1 *Denoising*

Denoising dilakukan menggukan *smoothing* sinyal dengan tujuan untuk mengurangi derau pada sinyal sehingga sinyal dengan nilai abnormal dan di luar batas kewajaran akan dihilangkan. Selain itu sinyal yang mengalami diskontinuitas (sinyal yang semula sudah konstan berada pada amplitudo tertentu kemudian langsung turun pada drastis pada kurun waktu singkat, lalu kembali konstan ke amplitudo gelombang awal) juga akan dihilangkan.

Bentuk paling sederhana dari *smoothing* adalah *moving average* dengan meratakan filter dengan *point averaging*. *Moving average* bekerja dengan menggantikan setiap nilai data dengan rata-rata nilai tetangga sejulah *point averaging*-nya. Besar nilai

point averaging yang optimal pada percobaan ini adalah 40 untuk sinyal ECG dan 20 untuk sinyal PPG[3]. Respon frekuensi sinyal ECG menunjukkan konsentrasi utama pada 2-40Hz untuk ECG dan 0.05-10Hz.

3.3.2 Dekomposisi DWT

Setelah dilakukan *smoothing*, pada kedua sinyal tersebut dilakukan *Discrete Wavelet Transform* (DWT). Cara kerja *wavelet* adalah untuk mendekomposisi sinyal atau memecah sinyal ke *sub band* tertentu menggunakan *highpass filtering* dan *lowpass filtering*. *Lowpass filtering* yang merupakan set fungsi skala menghasilkan komponen aproksimasi yaitu komposnen sinyal berfrekuensi rendah dan berskala tinggi. Sedangkan *highpass filtering* yang merupakan set fungsi wavelet menghasilkan komponen detail yaitu komponen sinyal berfrekuensi tinggi dan berskala rendah. Komponen aproksimasi dan detail yang dihasilkan melalui proses pemfilteran ini kemudian melewati proses *down sampling*.

Tipe *mother wavelet* yang digunakan adalah *Daubechies* 6 (db6). Dilakukan proses dekomposisi sebanyak 8 kali, yaitu pada level 8 untuk mendapatkan 8 buah koefisien detail dan 8 buah koefisien aproksimasi.

3.3.3 *Thresholding*

Nilai *threshold* dicari dari tiap-tiap subband koefisien detail hasil dekomposisi wavelet. Koefisien detail dilakukan *thresholding* karena pada proses DWT dilewatkan pada batas atas filter sehingga menghasilkan sinyal yang ekstrim, dengan kenaikan dan penurunan amplitudo yang signifikan dan sangat rapat. *Thresholding* bertujuan untuk memberikan ambang batas pada sinyal, sinyal yang berada dibawah ambang batas akan diratakan (di-*set* 0), apabila lebih dari ambang batas akan dipertahankan.

Besar nilai *threshold* yang digunakan pada masing-masing koefisien detail menggunakan *sqtwolog* yaitu sebesar akar dari 2 kali log panjang sinyal [2]. Selanjutnya batas *threshold* tersebut digunakan untuk *hard thresholding* pada detail koefisien dengan tujuan untuk mempertahankan amplitudo sinyal dan meratakan sinyal supaya lebih rendah.

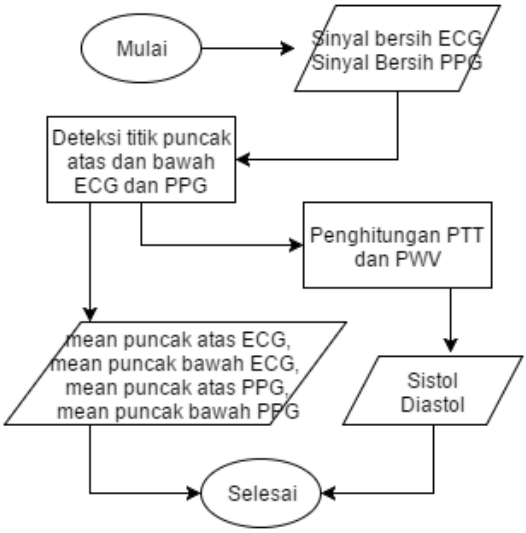
3.3.4 Rekonstruksi Sinyal

Proses rekonstruksi dari koefisien wavelet sinyal hasil transformasi berlangsung dengan proses *up-sampling* dan *filtering*. Proses *up-sampling* adalah memasukan nilai nol diantara setiap koefisien dan menjadikan panjangnya dua kali panjang semula. Selanjutnya data tersebut dikonvolusi dengan set fungsi balik skala dan wavelet untuk mendapatkan sinyal rekonstruksi.

Rekonstruksi bertujuan untuk membangun sinyal ke bentuk semula. Rekonstruksi sinyal dibangun dari sinyal aproksimasi dan detail serta *low-pass filter* dan *high-pass filter* rekonstruksi yang didapat dari proses DWT filter Daubachies 6. Untuk sinyal ECG dibangun dari koefisien aproksimasi A4-A6 dan detail D3, D4, D5, dan D6. Sedangkan sinyal PPG dibangun dari koefisien aproksimasi A6 dan detail D3, D4, D5, dan D6 [2]. Hasil keluaran dari rekonstruksi sinyal adalah sinyal bersih yang nantinya akan diproses pada tahapan selanjutnya. Sinyal Bersih yang dihasilkan berukuran matriks 1x1000 untuk masing-masing ECG dan PPG.

3.4 Ekstraksi dan Komputasi Fitur

Setelah diperoleh sinyal bersih, langkah selanjutnya adalah mengekstraksi fitur-fitur yang diperlukan untuk menunjang proses klasifikasi. Inti dari proses ekstraksi adalah melakukan pendeteksian titik puncak atas dan bawah pada sinyal ECG dan PPG serta melakukan pengitungan PTT dan PWV. Pada Gambar 3.9 digambarkan diagram alur proses eksraksi dan komputasi fitur



Gambar 3.9. Alur proses ekstraksi dan komputasi fitur

3.4.1 Deteksi puncak sinyal

Deteksi titik puncak atas dilakukan dengan mempertimbangkan indeks tetangganya. Jika mengalami kenaikan tajam dan diiringi penurunan yang tajam, maka akan dideteksi sebagai puncak. Pada matlab terdapat fungsi *findpeaks* yang dapat digunakan sebagai pendeteksi puncak sinyal. *Findpeaks* adalah metode untuk mendeteksi dan menganalisis ujung atau puncak dari gelombang sinyal. *Findpeaks* juga dikenal dengan *find local maxima*, dikarenakan metode ini menemukan ujung tertinggi pada gelombang local atau area tertentu.

Matlab telah menyediakan fungsi khusus yang digunakan untuk mengimplementasikan metode *findpeaks*, yaitu *pks* dan *locs*. Fungsi yang digunakan dengan *findpeaks* memiliki parameter data, parameter yang dimaksud adalah parameter berupa masukkan gelombang sinyal. *Pks* adalah jumlah puncak yang terdeteksi, *locs* adalah lokasi tiap puncak yang terdeteksi. Sedangkan untuk mengetahui puncak bawah hanya perlu menginvers matriks sinyal.

3.4.2 Komputasi fitur

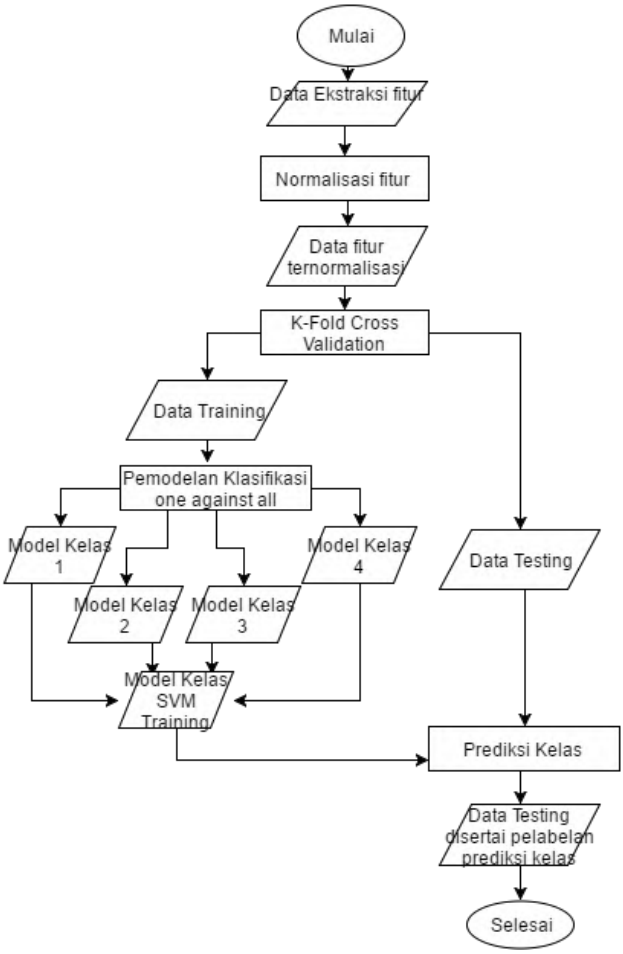
Setelah proses ekstraksi berhasil dilakukan, selanjutnya adalah menghitung nilai PTT dengan mengurangkan indeks lokasi dari puncak PPG ke puncak ECGG kemudian dikonversikan dengan ke dalam satuan detik dengan *sample rate* 125/detik. Terdapat 1000 sampel data, maka 1000 data tersebut mewakili 8 detik. Masing masing sampel mewakili 0,008 detik. Sehingga untuk mengetahui nilai PTT tinggal mengalikan selisih indeks lokasi dengan 0,008 detik. Nilai PTT tersebut kemudian digunakan untuk menghitung nilai PWV seperti pada persamaan 2.3.

Selanjutnya dilakukan penghitungan seperti persamaan 2.4 untuk mencari nilai sistol dan diastolnya. Selain sistol dan diastol, dilakukan komputasi fitur lain seperti rata-rata puncak atas dan bawah ECG dan PPG. Sehingga jumlah atribut yang digunakan untuk proses klasifikasi berjumlah 6, yaitu sistol, diastol, puncak atas ECG, puncak bawah ECG, puncak atas ECG, dan puncak bawah PPG. Untuk lebih jelasnya dapat dilihat pada Gambar 3.9.

3.5 Klasifikasi *Support Vector Machines* (SVM)

Data yang didapat dari proses ekstraksi tidak langsung diolah, terlebih dahulu dinormalisasi dengan rentang skala 0 hingga 1. Proses normalisasi bertujuan supaya data memiliki bobot yang baik karena telah dibandingkan dengan nilai minimal dan maksimalnya sehingga tidak terlalu signifikan. Pada tahap ini juga dilakukan penambahan label kelas secara manual, berdasarkan pada pendapat dokter.

Keenam fitur yang didapat dari proses ekstraksi dilakukan normalisasi dengan memasukkannya ke dalam Persamaan 2.4. Proses klasifikasi dapat dilihat pada Gambar 3.10. Proses klasifikasi terbagi menjadi 2, yaitu pemodelan dan penerapan model. Pemodelan dilakukan menggunakan data *training* sedangkan penerapan model dilakukan pada data *testing* untuk mengetahui prediksi kelas dari data yang diujikan.



Gambar 3.10. Alur Klasifikasi SVM

3.5.1 Normalisasi Fitur

Data yang digunakan untuk SVM terdapat 7 fitur yaitu sistol, diastol, rata-rata puncak atas ECG, rata-rata puncak bawah

ECG, rata-rata puncak atas PPG, rata-rata puncak bawah PPG dan pelabelan kelas dari dokter. Dilakukan normalisasi pada teknan sistol, diastol, rata-rata puncak atas ECG, rata-rata puncak bawah ECG, rata-rata puncak atas PPG, dan rata-rata puncak bawah PPG dengan menggunakan *Min-Max Scalling* untuk medapatkan rentang data 0-1. Label kelas tidak perlu dilakukan normalisasi karena label kelas adalah merupakan patokan yang sudah bersifat unik.

3.5.2 *K-Fold Cross Validation*

Setelah data telah dinormalisasi, dilakukan pembagian data *training* dan *testing* dengan menggunakan *K-fold Cross Validation.* Data *training* dan *testing* akan dibagi sejumlah *K* bagian. Salah satu bagian akan dijadikan sebagai data *testing* dan sisanya dijadikan data *training*. Begitu seterusnya, hingga semua bagian telah diujikan sebagai data testing.

3.5.3 Pemodelan *one against all*

Pemodelan klasifikasi *one against all* dilakukan pada data training sejumlah data training hasil pembagian dari *K-fold Cross Validation*. Dengan menggunakan algoritma *one against all,* karena tugas akhir ini memiliki 4 kelas, maka pada proses ini akan dihasilkan 4 macam pemodelan juga. Pemodelan ini bersifat biner, seperti penjelasan berikut :

- Model 1, Kelas 1 bernilai 1 dan selain kelas 1 bernilai 0.
- Model 2, Kelas 2 bernilai 1 dan selain kelas 2 bernilai 0.
- Model 3, Kelas 3 bernilai 1 dan selain kelas 3 bernilai 0.
- Model 4, Kelas 4 bernilai 1 dan selain kelas 4 bernilai 0.

3.5.4 Prediksi kelas

Data yang digunakan untuk prediksi kelas adalah data *testing*. Data testing akan diprediksi kelasnya berdasarkan

pemodelan atau proses learning yang telah dilakukan. Proses prediksi bertujuan untuk memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. Hasil keluaran dari klasifikasi adalah label kelas dari data *testing*, yaitu normal, prehipertensi, hipertensi stadium 1, atau hipertensi stadium 2

BAB IV
IMPLEMENTASI

Bab ini berisi penjelasan mengenai implementasi dari perancangan yang sudah dilakukan pada bab sebelumnya. Implementasi berupa kode sumber untuk membangun program.

4.1 Lingkungan Implementasi

Implementasi Pengukuran dan Klasifikasi Tekanan Darah berdasarkan *Pulse Transit Time* menggunakan metode Transformasi Wavelet dan *Support Vector Machines* (SVM) menggunakan spesifikasi perangkat keras dan perangkat lunak seperti yang ditunjukkan pada Tabel 4.1.

Tabel 4.1 Lingkungan Implementasi Perangkat Lunak

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras	Prosesor	AMD A4-1250 APU with Radeon (TM) HD Graphics (2CPUS)~1.0 GHz
	Memori	4 GB
Perangkat Lunak	Sistem Operasi	Wondows 10 Pro 64-bit versi 10.0
	Perangkat Pengembang	MATLAB R2014a
	Perangkat Pembantu	Microsoft Office Excel 2016 Microsoft Office Word 2016

4.2 Implementasi

Pada sub bab implementasi ini menjelaskan mengenai pembangunan perangkat lunak secara detail dan menampilkan kode sumber yang digunakan mulai tahap *preprocessing* hingga klasifikasi. Pada tugas akhir ini data yang digunakan seperti yang

telah dijelaskan di bab sebelumnya yaitu 60 data acak dengan pembangian data training dan testing menggunakan metode *k-fold cross validation*.

4.2.1 Implementasi Pemprosesan Data Masukan

Data masukan adalah sinyal ECG dan PPG yang mempunyai frekuensi sebesar 125 Hz. Pada Kode Sumber 4.1 baris 1-2 merupakan proses pencarian file yang berformat .mat yang akan digunakan. Kemudian pada baris 4-5 data dipecah menjadi 2, yaitu sinyal ECG dan PPG. Data sinyal ECG terletak pada baris 1 dari *dataset*, sedangkan sinyal PPG terletak pada baris 3. Pada baris ke 6-7 dilakukan smoothing dengan menggunakan panjang *point averaging* atau *window* sebesar 40 untuk sinyal ECG dan 20 untuk sinyal PPG.

```
1  [FileName,PathName] = uigetfile('*.mat','Select
   the MATLAB data file')
2  load(FileName,'-mat')
3  fs = 125
4  raw_ecg = FileName(1,:)
5  raw_ppg = FileName(3,:)
6  ecgsmooth = smooth(raw_ecg,40)
7  ppgsmooth = smooth(raw_ppg,20)
```

Kode Sumber 4.1 Kode masukan

4.2.2 Implementasi DWT

Hasil keluaran dari preprosesing digunakan sebagai masukan pada proses DWT. Berdasarkan Kode Sumber 4.2, baris 1-3 merupakan dekomposisi DWT. Sinyal diproses menggunakan 4 filter yang ditunjukkan oleh baris pertama yaitu Lo_D untuk *low-pass* dekomposisi, Hi_D untuk *high-pass* dekomposisi, Lo_R untuk *low-pass* rekonstruksi, dan Hi_R untuk *high-pass* rekonstruksi. Pada baris kedua sinyal didekomposisi pada level 8 dengan menggunakan filter *low-pass* dan *high-pass* dekomposisi. Setelah didekomposisi, proses selanjutnya adalah mencari koefisien aproksimasi dan detail seperti tampak pada bari 5-21.

```
[Lo_D,Hi_D,Lo_R,Hi_R] = wfilters('db6');
[C,L] = wavedec(ecgsmooth,8,Lo_D,Hi_D);
[d1,d2,d3,d4,d5,d6,d7,d8]=detcoef(C,L,[1,2,3,4
5,7,8]);

A8 = wrcoef('a',C,L,Lo_R,Hi_R,8);
A6 = wrcoef('a',C,L,Lo_R,Hi_R,6);
A7 = wrcoef('a',C,L,Lo_R,Hi_R,7);
A5 = wrcoef('a',C,L,Lo_R,Hi_R,5);
A4 = wrcoef('a',C,L,Lo_R,Hi_R,4);
A3 = wrcoef('a',C,L,Lo_R,Hi_R,3);
A2 = wrcoef('a',C,L,Lo_R,Hi_R,2);
A1 = wrcoef('a',C,L,Lo_R,Hi_R,1);

D1 = wrcoef('d',C,L,Lo_R,Hi_R,1);
D2 = wrcoef('d',C,L,Lo_R,Hi_R,2);
D3 = wrcoef('d',C,L,Lo_R,Hi_R,3);
D4 = wrcoef('d',C,L,Lo_R,Hi_R,4);
D5 = wrcoef('d',C,L,Lo_R,Hi_R,5);
D6 = wrcoef('d',C,L,Lo_R,Hi_R,6);
D7 = wrcoef('d',C,L,Lo_R,Hi_R,7);
D8 = wrcoef('d',C,L,Lo_R,Hi_R,8);
```

Kode Sumber 4.2 Kode program DWT pada sinyal ECG

Sedangkan untuk melakukan DWT pada sinyal PPG dilakukan proses yang sama seperti pada sinyal ECG. Implementasi DWT pada sinyal PPG dapat ditunjukkan pada Kode Sumber 4.3

```
[Loo_D,Hii_D,Loo_R,Hii_R] = wfilters('db6');
[C,L] = wavedec(ppgsmooth,8,Loo_D,Hii_D);
[d1,d2,d3,d4,d5,d6,d7,d8]=detcoef(C,L,[1,2,3,4
5,7,8]);

A_8 = wrcoef('a',C,L,Loo_R,Hii_R,8);
A_6 = wrcoef('a',C,L,Loo_R,Hii_R,6);
A_7 = wrcoef('a',C,L,Loo_R,Hii_R,7);
A_5 = wrcoef('a',C,L,Loo_R,Hii_R,5);
A_4 = wrcoef('a',C,L,Loo_R,Hii_R,4);
A_3 = wrcoef('a',C,L,Loo_R,Hii_R,3);
A_2 = wrcoef('a',C,L,Loo_R,Hii_R,2);
A_1 = wrcoef('a',C,L,Loo_R,Hii_R,1);
```

```
13
14  D_1 = wrcoef('d',C,L,Loo_R,Hii_R,1);
15  D_2 = wrcoef('d',C,L,Loo_R,Hii_R,2);
16  D_3 = wrcoef('d',C,L,Loo_R,Hii_R,3);
17  D_4 = wrcoef('d',C,L,Loo_R,Hii_R,4);
18  D_5 = wrcoef('d',C,L,Loo_R,Hii_R,5);
19  D_6 = wrcoef('d',C,L,Loo_R,Hii_R,6);
20  D_7 = wrcoef('d',C,L,Loo_R,Hii_R,7);
21  D_8 = wrcoef('d',C,L,Loo_R,Hii_R,8);
```

Kode Sumber 4.3 Kode program DWT pada sinyal PPG

4.2.3 Implementasi Thresholding

Setelah didekomposisi, koefisien detail dithreshold menggunakan sqtwolog dan hard thresholding untuk mempertahankan sinyal yang memiliki amplitudo besar. Proses thresholding untuk sinyal ECG dan PPG adalah sama. Besarnya nilai threshold dari sqtwolog adalah akar dari 2 kali log panjang sinyal. Thresholding koefisien detail dapat dilihat pada Kode Sumber 4.4 baris 1-9. *thselect* merupakan fungsi dari matlab yang digunakan untrus proses *denoising* 1 dimensi. Fungsi ini akan mengembalikan nilai *threshold* sesuai aturan yang dipakai, pada kasus ini memakai aturan sqtwolog. Pada baris 10-17 merupakan fungsi dari *hard thresholding*. *wthresh* merupakan fungsi matlab yang dipakai untuk melakukan *hard* dan *soft thresholding*. Pada tugas akhir ini memakai hard thresholding untuk mempertahankan amplitudo sinyal yang tinggi.

```
1   tr = 'sqtwolog';
2   thr_D1 = thselect(D1,tr)
3   thr_D2 = thselect(D2,tr)
4   thr_D3 = thselect(D3,tr)
5   thr_D4 = thselect(D4,tr)
6   thr_D5 = thselect(D5,tr)
7   thr_D6 = thselect(D6,tr)
8   thr_D7 = thselect(D7,tr)
9   thr_D8 = thselect(D8,tr)
10  tD1 = wthresh(D1,'h',thr_D1)
11  tD2 = wthresh(D2,'h',thr_D2)
```

```
12  tD3 = wthresh(D3,'h',thr_D3)
13  tD4 = wthresh(D4,'h',thr_D4)
14  tD5 = wthresh(D5,'h',thr_D5)
15  tD6 = wthresh(D6,'h',thr_D6)
16  tD7 = wthresh(D7,'h',thr_D7)
17  tD8 = wthresh(D8,'h',thr_D8)
```

Kode Sumber 4.4 Kode thresholding menggunakan sqtwolog

4.2.4 Implementasi Rekonstruksi sinyal

Selanjutnya mencari koefisien aproksimasi yang paling optimal dari sinyal ECG dan PPG. Berdasarkan percobaan yang dilakukan[3] koefisien aproksimasi A4-A6 pada sinyal ECG dan A6 pada sinyal PPG adalah yang paling optimal. Selanjutnya sinyal tersebut direkontruksi bersama dengan koefisien detail 3 hingga 6 bersama dengan *low-pass filter* dan *high-pass filter* pada level 8. yang dapat dilihat pada Kode Sumber 4.5 untuk mendapatkan sinyal bersih. Pada baris 1-2 merupakan rekonstruksi sinyal ECG, sedangkan baris 3-4 merupakan rekonstruksi sinyal PPG.

```
1  ApECG = A4-A6;
2  ECSignal = waverec(ApECG + tD3 + tD4 + tD5 +
3  tD6) ,8,Lo_R,Hi_R
4  ApPPG = A_6;
   PPSignal = waverec (ApPPG + Td_3 + Td_4 + Td_5
   + Td_6),8,Loo_R,Hii_R
```

Kode Sumber 4.5 Kode sinyal bersih

4.2.5 Implemestasi Ekstraksi Fitur

Pada proses ekstraksi fitur dilakukan proses deteksi R puncak ECG dan PPG. Untuk mendapatkan titik puncak atas, menggukan fungsi *findpeaks*. *Findpeaks* dengan membandingkan sinyal yang telah dilalui dengan sinyal selanjutnya. Apabila ia mengalami kenaikan lalu mengalami penurunan dalam waktu singkat maka akan dideteksi sebagai puncak. Apabila puncak yang

berikutnya tidak lebih besar atau setara dengan empat puncak yang telah disimpan sebagai puncak maksimal, maka puncak tersebut bukan merupakan puncak R. Pada Kode Sumber 4.6 pada baris 1-2 merupakan inisialisasi dari besarnya *window* dan nilai meannya. *Window* yang diset bernilai 5 (Didapat dari penghitungan Fs/25). Fs bernilai 125. Pada baris 4-6 terdapat *buffer_long* untuk menyimpan sampel sinyal berikutnya. Dan *buffer_base* digunakan untuk menyimpan sampel yang sudah dilewati. Pada baris 7-10 sampel sinyal diteruskan ke indeks berikutnya, apabila panjanh dari buffer yang dilewati lebih dari 250, maka *buffer_base*nya diperbarui.

Pada baris 11-13 dilakukan pembaharuan mean dan posisinya apabila data berikutnya masih lebih besar dari sebelumnya. Puncak dapat dideteksi dengan memberi *flag* 1 apabila posisi yang sekarang lebih besar daripada posisi pada *buffer_plot* sebelumnya. Pada baris 16-18 jumlah puncak di*counting* dan disimpan nilai amplitudonya serta indeks puncaknya.

```
1   window = round(Fs/25)%panjang sinyal
2   buffer_mean=mean(abs(ecg(1:2*fs)-
3   mean(ecg(1:2*fs))));

4   for i = 1 : length(ecg)
5   buffer_long = [buffer_long ecg(i)] ; %save
    upcoming new samples
6   buffer_base = [buffer_base ecg(i)] ; % save
    samples

    %memperbarui sampel
7   if length(buffer_base) >= 2*fs
8   buffer_mean = mean(abs(buffer_base(1:2*fs)-
    mean(buffer_base(1:2*fs))));
9   buffer_base =[];
10  end

11  if length(buffer_long)>= window %memperbarui
    mean
12  mean_online = mean(buffer_long); % menyimpan
    mean
```

```
13  bufferc_plot =[bufferc_plot mean_online]; %
    menyimpan posisi sinyal

14  if state == 1 % Puncak terting
15  if  currentmax > buffer_plot(i-window)
16     dum = dum + 1; %deteksi puncak R
17     R_i = [R_i ind];%indeks disimpan
18     Rwave = [Rwave buffer_plot(ind)];%amplitudo
    disimpan
19  end
20  end
21  end
```

Kode Sumber 4.6 Implementasi Algoritma deteksi puncak

Untuk mendapatkan titik puncak bawah, sinyal ECG normal diinvers kemudian dicari titik maksimumnya seperti Kode Sumber 4.7. pada baris ke-3.

```
1  [~,ecg_Rwave] = findpeaks(ECGSignal)
2  ECGSignal_inverted = -ECGSignal
3  [~,ecg_Swave] = findpeaks(ECGSignal_inverted)
4  ampR_ecg = findpeaks(ECGSignal)
5  ampS_ecg = findpeaks(ECGSignal_inverted)
```

Kode Sumber 4.7 Implementasi deteksi puncak atas dan bawah ECG

Untuk sinyal PPG diberi perlakuan yang sama dengan sinyal ECG, hal tersebut dapat dilihat pada Kode Sumber 4.8.

```
1  [~,ppg_Rwave] = findpeaks(PPGSignal)
2  PPGSignal_inverted = -PPGSignal
3  [~,ecg_Swave] = findpeaks(PPGSignal_inverted)
4  ampR_ppg = findpeaks(PPGignal)
5  ampS_ppg = findpeaks(PPGignal_inverted)
```

Kode Sumber 4.8 Implementasi deteksi puncak PPG

4.2.6 Implemestasi Komputasi Fitur

Kemudian dilakukan proses penghitungan PTT, PWV, sistol, dan diastol seperti pada Kode Sumber 4.9. Pada baris 1-4 dilakukan penghitungan rata-rata puncak atas sinyal ECG dan

PPG. Proses penghitungan PPT dapat dilihat pada baris 6-20 berdasarkan pada Persamaan 2.2. Pada baris 18-24 dilakukan penghitungan nilai sistol, dan diastol berdasarkan pada Persamaan 2.4.

```
1  meanR_ECG = mean(ampR_ecg)
2  meanS_ECG = mean(ampS_ecg)
3  meanR_EPPG = mean(ampR_ppg)
4  meanS_EPPG= mean(ampS_ppg)
5  Sum = 0
6  for i=1:length(R_ecg)
7  for j=1:length(R_ppg)
8      if R_ecg (i)>R_ppg (j)
9          Selisih = (ampR_ecg (i) - ampR_ppg (j)
10         Sum = Sum + Selisih
11         Meansum = mean(sum)
12         PTT = meansum *0.008
13     Else
14         PTT = 0;
15     End
16   End
17 End
18 disp('PTT=');disp(PTT');
19 PWV = 0.5*160*1000/PTT;
20 disp('PWV=');disp(PWV');
21 BPsis = (0.05089855*PWV)+62.5590972;
22 disp('BPsis=');disp(BPsis');
23 BPdis = (0.04940772*PWV)+17.4800472;
24 disp('BPdis=');disp(BPdis');
```

Kode Sumber 4.9 Implementasi komputasi fitur

4.2.7 Implementasi Normalisasi

Sebelum proses klasifikasi, dilakukan normalisasi untuk melakukan penyekalaan data dapat dilihat pada Kode Sumber 4.10.

```
1  for i = 1:length(coltr)
2  Normcoltr(:i) = (coltr(:i)-
   min(coltr))/((max(coltr)-(min(coltr))
3  End
```

Kode Sumber 4.10 Implementasi normalisasi

4.2.8 Implementasi *K-fold Cross Validation*

Selanjutnya data dari sinyal bersih dibagi menjadi 2 yaitu data testing dan data training. Data terdiri dari 60 data yang dijadikan satu dan terbagi masing masing 15 untuk pasien normal, prehipertensi, hipertensi stadium 1, dan hipertensi stadium 2. Pembagian data training dan testing menggunakan *k-fold cross validation* dengan *k* = 3 dapat dilihat pada kode sumber 4.11.

```
1   K = 3;
2   data = meas_2;
3   C = crossvalind('Kfold',size(data,1), K);
4
5   for b = 1:K
6       Test = (C == b);
7       Train = ~Test;
8       trainData = data(Train,1:2);
9       trainLabel = data(Train,3);
10      testData = data(Test,1:2);
11      testLabel = data(Test,3);
12      [~,~,labels] = unique(trainLabel);
13      numInst = size(data,1);
14      numLabels = max(labels);
15      numTrain = length(trainData);
16      numTest = numInst - numTrain;
```

Kode Sumber 4.11 Pembagian data training dan testing

4.2.9 Implementasi Support Vector Machines (SVM)

Menggunakan multikelas SVM *ones agains all*, dimana setiap kelas dibuat model binernya. Terdapat 4 kelas, sehingga dibuat 4 model, yaitu model kelas 1, model kelas 2, model kelas 3, dan model kelas 4. Proses pemodelan ditunjukkan pada Kode Sumber 4.12 pada baris 1-4. Pemodelan dilakukan pada data *training* menggunakan fungsi svmtrain. Terdapat beberapa parameter yang dapat dimasukkan seperti *-c* untuk *cost margin*, *-t* untuk metode *kernel*, *-b* untuk kemungkinan probabilitas, dan *-g* untuk *gamma*

```
1   model = cell(numLabels,1)
2       for i=1:numLabels
3       model{i} = svmtrain(double(trainLabel==i),
    trainData, '-c 20 -t 2 -b 1');
4       end
```

Kode Sumber 4.12 Pemodelan masing-masing kelas

Pada baris ke-3, `-c 20 -t 2 -b 1` menjelaskan nilai parameter *c* yang digunakan adalah 20, *kernel* yang dipakai adalah RBF, dan nilai estimasi probabilitasnya adalah 1. Sedangkan nilai *gamma* secara *default*nya sebesar 1 dibagi dengan jumlah fiturnya, yang pada percobaan ini bernilai 0,167. Nilai *gamma* ini boleh ditulis boleh tidak. Namun, apabila ingin mengganti nilai *gamma*, maka harus dituliskan dengan cara -g diikuti dengan nilai *gamma* yang diinginkan.

Nilai *c* dapat disesuaikan sesuai keinginan pengguna. Nilai defaultnya adalah 1, jika ingin mengganti maka tinggal menuliskan *-c* diikuti nilai yang diinginkan. Sedangkan pada parameter *kernel* terdapat beberapa panduan, nilai 0 digunakan umtuk metode kernel linear, 1 untuk polynomial, 2 untuk RBF, dan 3 untuk sigmoid, dan 4 untuk precomputed kernel.

Setelah dibuat model, maka akan dilakukan prediksi dan penghitungan akurasi. Pada Kode Sumber 4.13 baris 1-3 dilakukan proses prediksi menggunakan *svmpredict* menggunakan data *testing*. Model yang sudah dibuat pada tahap sebelumnya, diterapkan pada data *testing* untuk memperkirakan label kelasnya. Pada baris 4-7 dilakukan penghitungan akurasi antara kelas prediksi dengan kelas sebenarnya.

```
1   prob = zeros(numTest,numLabels);
2       for i=1:numLabels
3   [~,~,p] = svmpredict(double(testLabel==i),
    testData, model{i}, '-b 1');
    [~,pred] = max(prob,[],2);
4   acc = sum(pred == testLabel) ./
    numel(testLabel)
5   Confus = confusionmat(testLabel, pred)
```

```
6   ac(b)=acc;
7   meanAcc=mean(ac)*100;
```

Kode Sumber 4.13 Prediksi kelas dan akurasi

(Halaman ini sengaja dikosongkan)

BAB V
HASIL UJI COBA DAN EVALUASI

Bab ini berisi penjelasan mengenai skenario uji coba dan evaluasi pada klasifikasi tekanan darah dari data ECG dan PPG berdasarkan *Pulse Transit Time* menggunakan metode *Discrete Wavelet Transform* dan *Support Vector Machines*. Hasil uji coba didapatkan dari implementasi pada BAB IV dengan skenario yang berbeda. Bab ini berisikan pembahasan mengenai lingkungan pengujian, data pengujian, dan uji kinerja.

5.1 Lingkungan Pengujian

Lingkungan pengujian pada uji coba permasalahan klasifikasi tekanan darah dari data ECG dan PPG berdasarkan *Pulse Transit Time* menggunakan metode *Discrete Wavelet Transform* dan *Support Vector Machines* menggunakan spesifikasi keras dan perangkat lunak seperti yang ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Lingkungan Pengujian

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras	Prosesor	AMD A4-1250 APU with Radeon (TM) HD Graphics (2CPUS)~1.0 GHz
	Memori	4 GB
Perangkat Lunak	Sistem Operasi	Wondows 10 Pro 64-bit versi 10.0
	Perangkat Pengembang	MATLAB R2014a
	Perangkat Pembantu	Microsoft Office Excel 2016 Microsoft Office Word 2016

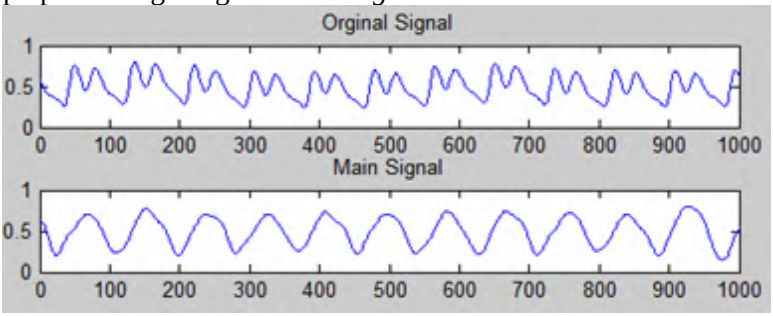
5.2 Data Pengujian

Sub bab ini menjelaskan mengenai data yang digunakan pada uji coba. Seperti yang telah dijelaskan sebelumnya, data berasal dari *www.physionet.org*. Data mentah kemudian diolah menggunakan *smoothing* dan DWT sehingga menghasilkan 2 matriks yaitu *training* dan *testing*. Data hasil DWT dan ekstraksi fitur akan menjadi masukan pada proses klasifikasi.

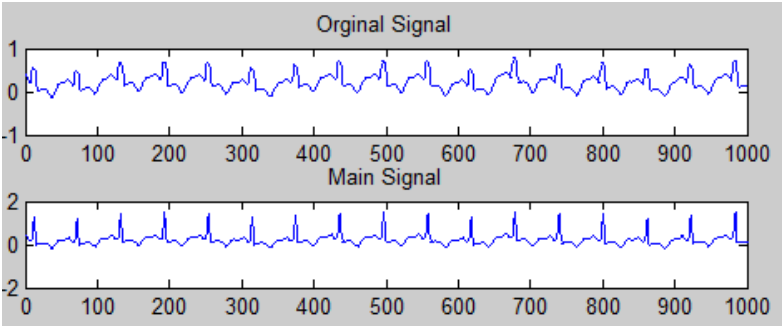
Data yang akan diolah adalah 60 buah data yang dipilih secara acak dengan pembagian masing-masing 15 untuk kelas normal, prehipertensi, hipertensi, hipertensi stadium 1, dan hiperteensi stadium 2. Data mentah tersebut dibagi menjadi sejumlah *k* dalam pembagian data *training* dan *testing* dengan menggunakan metode *k-fold cross validation*.

5.3 Preprocessing Sinyal

Pada tahap *preprocessing* data dengan *smoothing* dan DWT, terlebih dahulu data mentah dibagi menjadi sinyal ECG dan PPG masing-masing matriks berukuran 1x1000 yang selanjutkan dihilangakan deraunya. Gambar 5.1 dan 5.2 adalah contoh sinyal data ECG dan PPG pada kelas hipertensi sebelum dan sesudah di-*preprocessing*. *Original Signal* adalah sinyal rekaman asli yang masih banyak deraunya, sedangkan *main signal* adalah sinyal hasil preprocessing dengan *smoothing* dan DWT.



Gambar 5.1. Sinyal ECG sebelum dan sesudah *preprocessing*

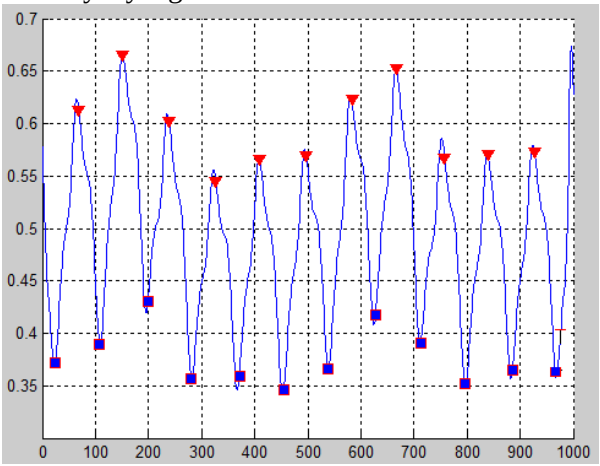


Gambar 5.2. Sinyal PPG sebelum dan sesudah *preprocessing*

Gambar 5.1 dan 5.2 adalah contoh sinyal data ECG dan PPG pada kelas hipertensi sebelum dan sesudah di-*preprocessing*. *Original Signal* adalah sinyal rekaman asli yang masih banyak deraunya, sedangkan *main signal* adalah sinyal hasil preprocessing dengan *smoothing* dan DWT.

5.4 Deteksi Puncak

Gambar 5.3 menunjukkan hasil deteksi puncak atas dan puncak bawah sinyal yang sudah dilakukan.



Gambar 5.3. Deteksi puncak atas bawah sinyal PPG

5.5 Skenario Uji Coba

Sebelum melakukan uji coba, perlu ditentukan skenario yang akan digunakan dalam uji coba. Melalui skenario ini, perangkat akan diuji apakah sudah berjalan dengan benar dan bagaimana performa pada masing-masing skenario. Dan membandingkan skenario manakah yang memiliki hasil lebih baik. Terdapat 3 macam skenario uji coba, yaitu :

1. Perhitungan performa dengan mengubah nilai *k* pada *fold cross validation* untuk membagi data *training* dan *testing*. Nilai *k* yang diujikan adalah 3, 4, 5 dan 10.
2. Perhitungan performa dengan mengubah besaran parameter *C* pada proses klasifikasi SVM. Nilai *c* yang diujikan adalah 1, 5, 10, 15, 20, 25, dan 30.
3. Perhitungan performa dengan mengubah metode *kernel* pada proses klasifikasi menggunakan SVM. Metode *kernel* yang diujikan antara lain linear, *polynomial*, RBF (*Radial Basis Function*), dan *sigmoid.*

5.5.1 Skenario Uji Coba 1

Skenario uji coba 1 adalah perhitungan akurasi dengan mencoba beberapa nilai *k* pada *k-fold cross validation* untuk pembagian data *training* dan *testing*. Nilai *k* yang diuji coba yaitu 3, 4, 5 dan 10. Setiap nilai *k* akan membagi jumlah data *training* dan testing yang *berbeda*. Pada nilai *k* = 3 menghasilkan 3 bagian data masing-masing berisi 20 data. Satu bagian data tersebut akan dijadikan data *training* dan sisanya adalah data *testing* begitu seterusnya sesuai dengan *k* bagiannya. Pada nilai *k* = 3 menghasilkan 3 bagian data masing-masing berisi 20 data. Satu bagian data tersebut akan dijadikan data *testing* dan sisanya adalah data *training*. Pada nilai *k* = 4 menghasilkan 4 bagian data masing-masing berisi 15 data. Satu bagian data tersebut akan dijadikan data *testing* dan sisanya adalah data *training*. Pada nilai *k* = 5 menghasilkan 5 bagian data masing-masing berisi 12 data. Satu

bagian data tersebut akan dijadikan data *testing* dan sisanya adalah data *training*. Sedangkan Pada nilai *k* = 10 menghasilkan 10 bagian data masing-masing berisi 6 data. Satu bagian data tersebut akan dijadikan data *testing* dan sisanya adalah data *training*. Hasil performa dan rata-rata akurasi dapat dilihat pada Tabel 5.2.

Tabel 5.2 Performa masing-masing nilai k

Nilai *k*	Akurasi
3	82,91%
4	84,16%
5	**87,25%**
10	81,25%

Berdasarkan hasil yang ditunjukkan pada perhitungan performa diatas, nilai *k* = 5 memiliki akurasi tertinggi yaitu 87,08% dibandingkan nilai *k* sebesar 3, 4, dan 10 yang paling tinggi hanya mencapai 84,16%.

5.5.2 Skenario Uji Coba 2

Pada skenario uji coba yang kedua akan diuji dengan mengganti nilai parameter *C* pada proses klasifikasi SVM. Skenario ini menggunakan performa terbaik dari skenario uji coba 1, yaitu menggunakan parameter *k* = 5. Tujuan dari mengganti parameter *C* adalah untuk mendapatkan *hyperlane* dan *margin* yang paling optimal pada proses klasifikasi. *Hyperlane* diharapkan dapat membagi kelas secara tepat atau memiliki *error* akurasi seminimal mungkin. Sedangkan margin diharapkan mempunyai nilai sekecil-kecilnya, karena jika margin antara *hyperlane* dan *support vector* bernilai kecil maka toleransi Kelas Nilai *k* yang akan diuji coba adalah 1, 5, 10, 15, 20, dan 25. Nilai *C* yang dipilih adalah nilai yang menghasilkan akurasi paling baik pada yaitu 20.

Hasil performa dan rata-rata pada masing-masing nilai *C* dapat dilihat pada Tabel 5.3.

Tabel 5.3 Performa masing-masing nilai *C*

C	**Akurasi**
1	82,50%
5	84,00%
10	88,33%
15	90,00%
20	**91,67%**
25	90,98%

Berdasarkan hasil yang ditunjukkan pada Tabel 5.3 akurasi dapat ditingkatkan dengan memperbesar nilai *C*. Nilai akurasi meningkat seiring dengan naiknya nilai parameter C, akan tetapi pada nilai *C* = 20 menghasilkan akurasi paling besar yaitu 91,67%.

5.5.3 Skenario Uji Coba 3

Pada uji skenario yang ketiga akan dilakukan percobaan pada parameter *kernel*. Pada skenario 1 dan 2 telah didapatkan nilai *k* adalah 5 dan nilai *c* sebesar 20 menghasilkan akurasi paling baik. Metode *kernel* yang akan diujicobakan adalah linear, polinomial, RBF, dan sigmoid. Hasil dari uji icoba parameter *kernel* dapat dilihat pada Tabel 5.4. Parameter *gamma* yang digunakan untuk *kernel* RBF adalah *default*, yaitu bernilai 1 per jumlah fitur. Terdapat 6 fitur dalam percobaan ini, sehingga *gamma* bernilai 0,167.

Berdasarkan hasil yang ditunjukkan oleh Tabel 5.4 *kernel RBF* memiliki hasil akurasi paling tinggi yaitu 91,67%. Meskipun *kernel RBF* merupakan *default* yang digunakan pada SVM multi

kelas, percobaan ini menguatkan bahwa *kernel RBF* merupakan yang terbaik.

Tabel 5.4 Performa masing-masing *kernel*

Kernel	**Akurasi**
Lineal	80,00%
Polinomial	68,67%
RBF	**91,67%**
Sigmoid	63,33%

5.6 Analisis Hasil Uji Coba

Berdasarkan 3 skenario uji coba yang telah dilakukan, penulis memiliki analisa sebagai berikut:

1. Pada uji coba 1, pada penentuan nilai *k* sebesar 3 dan 4 memiliki data training 40 dan 45 masih kurang untuk proses *learning*. Sehingga pada *k* = 5 hasilnya lebih baik dengan data *testing* yang lebih sedikit dimana menghasilkan nilai akurasi sebesar 87,25% dengan data *training* sebanyak 48 dan *testing* sebanyak 12. Namun, semakin banyak data *training* dan sedikit data *testing* tidak menjamin akurasi yang dihasilkan semakin baik. Pada *k* sebesar 10, nilai akurasi turun 6%. Berdasarkan uji coba tersebut nilai *k* = 5 adalah yang paling optimal.
2. Pada uji coba 2 menghasilkan akurasi paling baik pada yaitu nilai *C* = 20 dengan tingkat akurasi 91,67%. Pada Tabel 5.3 menggambarkan bahwa semakin besar parameter *C* maka akurasinya akan semakin meningkat. Hal itu terjadi karena parameter *C* memberikan margin yang lebih kecil antara *hyperlane* dan *support vector* serta memberikan *hyperlane* yang mampu memisahkan

kelas secara tepat. Akan tetapi tidak selamanya semakin tinggi parameter *C* semakin tinggi pula akurasinya. Seperti pada parameter *C* bernilai 1, 5, 10, 15, dan 20 mengalami kenaikan akurasi, tetapi pada parameter *C* = 25 akurasinya menurun. Hal itu dapat terjadi tergantung pola data yang diujikan.

3. *Kernel* RBF degan *gamma* = 0,167 menghasilkan hasil terbaik dengan akurasi sebesar 91,67%, dikarenakan *kernel* RBF menggunakan fungsi *Gaussian* dimana mengimplementasikan fungsi eksponensial dan quadratik.
4. Akurasi terbaik dari ketiga skenario menghasilkan akurasi sebesar 91,67%, menggunakan parameter *k* = 5, *C* = 20, dan *kernel* RBF degan *gamma* = 0,167.

LAMPIRAN

Tabel A. 1 Daftar file yang digunakan sebagai masukan sistem

Data			
Normal	**Prehipertensi**	**Hipertensi Stadium 1**	**Hipertensi stadium 2**
A1	B1	C1	D1
A2	B2	C2	D2
A3	B3	C3	D3
A4	B4	C4	D4
A5	B5	C5	D5
A6	B6	C6	D6
A7	B7	C7	D7
A8	B8	C8	D8
A9	B9	C9	D9
A10	B10	C10	D10
A11	B11	C11	D11
A12	B12	C12	D12
A13	B13	C13	D13
A14	B14	C14	D14
A15	B15	C15	D15

Tabel A. 2 Confusion matrix uji coba 1, *K*=3

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AK	**Normal**	3	0	0	0

TUAL	**Pre-hipertensi**	4	2	1	0
	Hipertensi stadium 1	0	0	7	1
	Hipertensi stadium 2	0	0	0	2

Tabel A. 3 Confusion matrix uji coba 1, *K*=4

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	3	0	0	0
	Pre-hipertensi	1	1	0	0
	Hipertensi stadium 1	0	1	2	0
	Hipertensi stadium 2	0	0	0	7

Tabel A. 4 Confusion matrix uji coba 1, *K*=5

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	3	0	0	0
	Pre-hipertensi	1	1	0	0
	Hipertensi stadium 1	0	1	2	0
	Hipertensi stadium 2	0	0	0	7

Tabel A. 5 Confusion matrix uji coba 1, *K*=10

	PREDIKSI			
	Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**

AKTUAL	**Normal**	2	1	0	0
	Pre-hipertensi	0	0	0	0
	Hipertensi stadium 1	0	0	1	0
	Hipertensi stadium 2	0	0	0	2

Tabel A. 6 Confusion matrix uji coba 2, *C* = 1

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	4	0	0	0
	Pre-hipertensi	1	3	1	0
	Hipertensi stadium 1	0	0	0	0
	Hipertensi stadium 2	0	0	0	3

Tabel A. 7 Confusion matrix uji coba 2, *C* = 5

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	4	0	0	0
	Pre-hipertensi	0	2	1	0
	Hipertensi stadium 1	0	0	2	0
	Hipertensi stadium 2	0	0	0	3

Tabel A. 8 Confusion matrix uji coba 2, *C* = 10

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	4	0	0	0
	Pre-hipertensi	0	3	0	0
	Hipertensi stadium 1	0	0	3	1
	Hipertensi stadium 2	0	0	0	1

Tabel A. 9 Confusion matrix uji coba 2, *C* = 15

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	1	1	0	0
	Pre-hipertensi	0	2	0	0
	Hipertensi stadium 1	0	1	1	1
	Hipertensi stadium 2	0	0	0	5

Tabel A. 10 Confusion matrix uji coba 2, *C* = 20

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	2	0	0	0
	Pre-hipertensi	0	2	0	0
	Hipertensi stadium 1	0	1	1	0

	Hipertensi stadium 2	0	0	0	6

Tabel A. 11 Confusion matrix uji coba 2, *C* = 25

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	1	0	0	0
	Pre-hipertensi	0	4	0	0
	Hipertensi stadium 1	0	0	1	0
	Hipertensi stadium 2	0	0	1	5

Tabel A. 12 Confusion matrix uji coba 3, *kernel* linear *gamma* 0,167

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
AKTUAL	**Normal**	4	0	0	0
	Pre-hipertensi	0	1	0	0
	Hipertensi stadium 1	0	0	2	0
	Hipertensi stadium 2	0	0	0	5

Tabel A. 13 Confusion matrix uji coba 3, *kernel* polinomial *gamma* 0,167

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
A	**Normal**	2	0	0	0

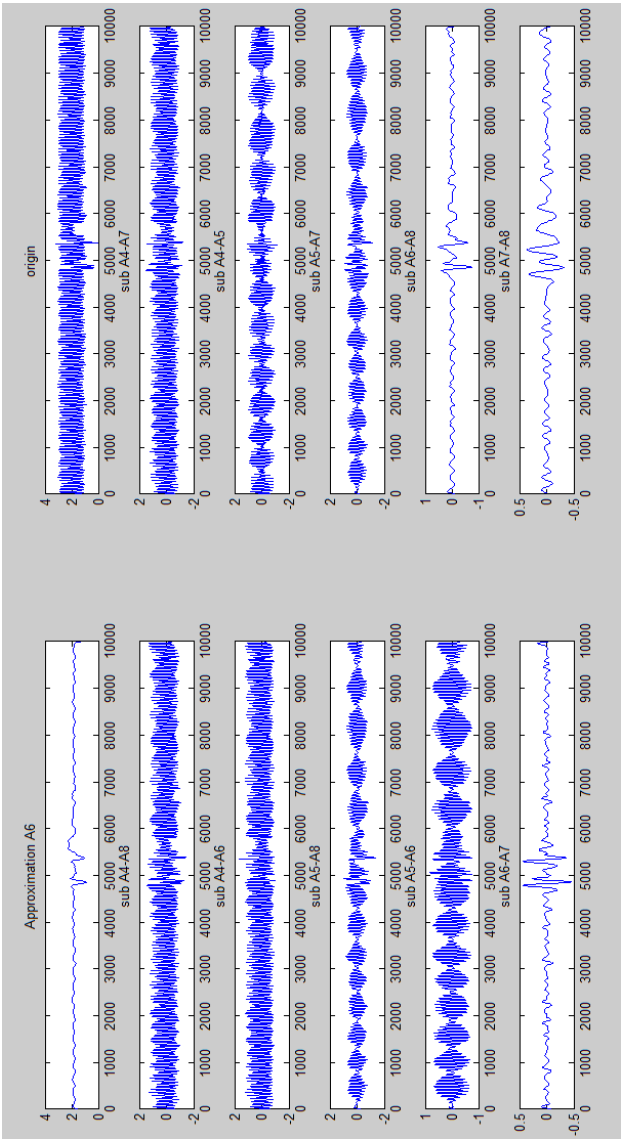
K T U A L	**Pre-hipertensi**	4	0	0	0
	Hipertensi stadium 1	1	0	3	0
	Hipertensi stadium 2	0	0	0	2

Tabel A. 14 Confusion matrix uji coba 3, *kernel* RBF *gamma* 0,167

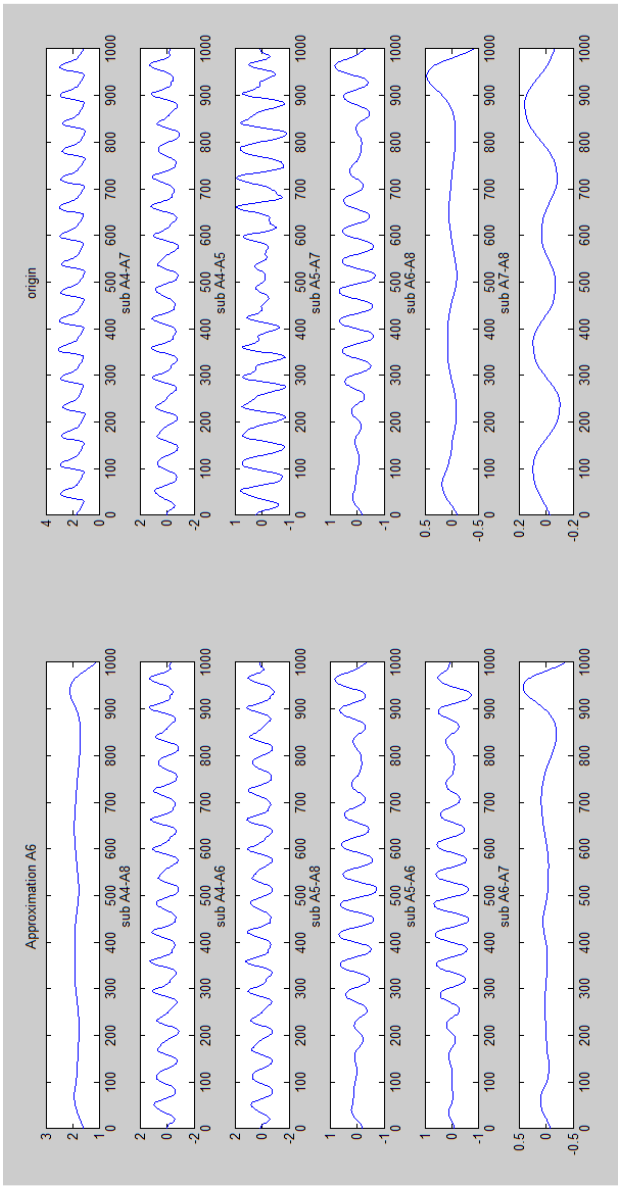
		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
A K T U A L	**Normal**	2	0	0	0
	Pre-hipertensi	0	2	0	0
	Hipertensi stadium 1	0	1	1	0
	Hipertensi stadium 2	0	0	0	6

Tabel A. 15 Confusion matrix uji coba 3, *kernel* sigmoid *gamma* 0.167

		PREDIKSI			
		Normal	**Pre-hipertensi**	**Hipertensi stadium 1**	**Hipertensi stadium 2**
A K T U A L	**Normal**	2	0	0	0
	Pre-hipertensi	0	2	2	0
	Hipertensi stadium 1	0	0	2	0
	Hipertensi stadium 2	0	0	0	4



Gambar A. 1 Proses pemilihan koefien aproksimasi sinyal PPG



Gambar A. 2 Proses pemilihan koefien aproksimasi Sinyal ECG

BAB VI
KESIMPULAN DAN SARAN

Bab ini berisikan kesimpulan yang dapat diambil dari hasil uji coba yang telah dilakukan. Selain kesimpulan, terdapat juga saran yang ditujukan untuk pengembangan perangkat lunak nantinya.

6.1 Kesimpulan

Kesimpulan yang didapatkan berdasarkan hasil uji coba klasifikasi tekanan darah dari data ECG dan PPG menggunakan *Discrete Wavelet Transform* (DWT) dan *Support Vector Machines*(SVM) adalah sebagai berikut:

1. Implementasi metode *Discrete Wavelet Transform* (DWT) *Daubachies* 6 level 8 dapat digunakan untuk menghilangkan derau dan mengekstraksi fitur data ECG dan PPG.
2. *Pulse Transit Time* (PTT) dapat digunakan untuk mengukur tekanan darah, yaitu dengan mengurangkan selisih waktu puncak sinyal ECG dengan waktu puncak sinyal PPG.
3. Implementasi dari metode *Support Vector Machines*(SVM) multi kelas *one against all* mampu mengklasifikasikan apakah seseorang normal, prehipertensi, hipertensi stadium 1, atau hipertensi stadium 2 dengan akurasi terbaik yaitu sebesar 91,67% dengan menggunakan parameter K = 5, C = 20, dan *kernel* RBF *gamma* 0,167.

6.2 Saran

Saran yang diberikan terkait pengembangan pada Tugas Akhir ini adalah:

1. Sistem sebaiknya dilengkapi dengan GUI untuk memudahkan penggunaan.
2. Dari segi preprocessing data menggunakan DWT pada rekonstruksi sinyal, sebaiknya dilakukan penelitian lebih

lanjut untuk mengetahui koefisien aproksimasi yang lebih optimal.

3. Pada data sinyal ECG terdapat pola sinyal yang berbeda-beda, hal itu dikarenakan ECG merupakan gelombang elektrik yang dihasilkan oleh otot jantung. Sehingga apabila terdapat otot jantung yang rusak akan sangat berpengaruh pada sinyal yang dihasilkan, penulis menyarankan agar berkonsultasi dengan dokter terkait sinyal ECG yang baik untuk digunakan dalam percobaan.

DAFTAR PUSTAKA

[1] World Health Organization, “World Health Statistics 2014”, 2014.

[2] M. Kachuee, M. M. Kiani, H. Mohammadzade, M. Shabany, Cuff-Less High-Accuracy Calibration-Free Blood Pressure Estimation Using Pulse Transit Time, IEEE International Symposium on Circuits and Systems (ISCAS'15), 2015

[3] S.Ashima, P.Manimegalai, Dr.K.Thanushkodi, Wavelet Based Pulse Rate and Blood Pressure Estimation System from ECG and PPG Signals” International Conference on Computer, Communication and Electrical Technology – ICCCET2011, 18th & 19th March, 2011: 285-289

[4] V. D. Sanchez, Advanced support vector machines and kernel methods, Neurocomputing 2003, 55, 5-20.

[5] Machine Learning Repository Cuff-Less Blood Pressure Estimation Data Set (https://archive.ics.uci.edu) diakses pada 15 November 2015.

[6] Smeltzer, Suzanne C dan Brenda G. Bare. 2001. Keperawatan Medikal Bedah 2, Edisi 8. Jakarta

[7] JNC 7 diakses pada http://www.nhlbi.nih.gov/files/docs/guidelines/jnc7full.pdf (online) tanggal 5 Mei 2016

[8] Bristish Hipertension Society diakses pada http://bhsoc.org/ tanggal 1 Januari 2016

[9] S, Wijaya. 1990. EKG Praktis. Jakarta : Binarupa Aksara.

[10] Bristish Cardiovascular Society diakses pada https://www.bcs.com/documents/consensus_guidelines.pdf (online) tanggal 4 Maret 2016

[11] Ali, Musyaffa'. Aplikasi Photoplethysmograph sebagai Alat Ukur Detak Jantung Menggunakan Transmisi Light Emitting Diode Inframerah dan Fotodioda. 2013. diakses (online) http://repository.unej.ac.id/bitstream/handle/123456789/3880/Musyaffa%20Ali%20-%20081910201018_1.pdf?sequence=1 pada 10 Mei 2016.

[12] Osuna, Edgar E. et. al.1997. Support Vector machines: Training and Applications. MIT, 1997.

[13] Wang R, Jia W, Mao ZH, Sclabassi RJ, Sun M , Cuff-Free Blood Pressure Estimation Using Pulse Transit Time and Heart Rate, Int Conf Signal Process Proc. 2014 Oct; 2014: 115–118

[14] Khandelwal S, Sahni S, Kumar S Pressure Sensor Based Estimation of Pulse Transit TimeInt Conf Signal Process International Journal of Information & Computation Technology, 2014, 1321-1328

[15] Min-Max Scalling diakses pada http://www.ni.com/white-paper/11320/en/ (online) 20 Juli 2016

[16] Christianini, Nello dan John S. Taylor. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, 2000

BIODATA PENULIS

Leli Maharani, lahir di Sukoharjo (Jawa Tengah) pada tanggal 16 Januari 1993 merupakan anak pertama dari Bapak Rakidi dan Ibu Eny Suswati. Penulis menempuh pendidikan formal dimulai dari TK Teladan (1998-2000), SD Negeri Gayam 01 (2000-2006), SMP Negeri 1 Sukoharjo (2006-2009), SMA Negeri 1 Sukoharjo (2009-2012) dan S1 Teknik Informatika ITS (2012-2016). Bidang studi yang diambil oleh penulis pada saat berkuliah di Teknik Informatika ITS adalah Komputasi Cerdas dan Visi (KCV). Penulis aktif dalam organisasi seperti Ikatan Keluarga Mahasiswa Sukoharjo se-Surabaya (2012-2015), Himpunan Mahasiswa Teknik Computer-Informatika (2013-2015), Keluarga Muslim Informatika (2013-2015), dan Jamaah Masjid Manarul Ilmi (2013-2014). Penulis juga aktif dalam berbagai kegiatan kepanitiaan yaitu pada SCHEMATICS 2013 dan SCHEMATICS 2014 pada divisi Kestari. Penulis juga menyukai kegiatan sosial dan bergabung pada perkumpulan Sunday Sharing Heroes Mahasistwa Teknik Informatika. Penulis memiliki hobi bersepeda dan menyukai hal baru. Penulis dapat dihubungi melalui email: maharanileli@gmail.com