AN OPTIMIZED FRAMEWORK FOR THE PREDICTION OF BLOOD PRESSURE BASED ON MORPHOLOGICAL AND DYNAMIC FEATURES OF PPG AND ECG

TAI CHEN BOON

BACHELOR OF MECHATRONICS ENGINEERING WITH HONOURS
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

AN OPTIMIZED FRAMEWORK FOR THE PREDICTION OF BLOOD PRESSURE BASED ON MORPHOLOGICAL AND DYNAMIC FEATURES OF PPG AND ECG

TAI CHEN BOON

A report submitted
in partial fulfilment of the requirements for the degree of
Bachelor of Mechatronics Engineering with Honours

Faculty of Electrical Technology and Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

DECLARATION

I declare that this thesis entitled "AN OPTIMIZED FRAMEWORK FOR THE PREDICTION OF BLOOD PRESSURE BASED ON MORPHOLOGICAL AND DYNAMIC FEATURES OF PPG AND ECG is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in the candidature of any other degree.

Signature : 

Name : TAI CHEN BOON

Date : 22/6/2024

APPROVAL

I hereby declare that I have checked this report entitled " AN OPTIMIZED FRAMEWORK FOR THE PREDICTION OF BLOOD PRESSURE BASED ON MORPHOLOGICAL AND DYNAMIC FEATURES OF PPG AND ECG ", and in my opinion, this thesis fulfils the partial requirement to be awarded the degree of Bachelor of Mechatronics Engineering with Honours

Signature : .....

Supervisor Name : DR. LOH SER LEE

Date : 22/6/2024 .....

DEDICATIONS

To my beloved mother and father,

whose unwavering support and encouragement.

To my honoured supervisor,

Dr Loh Ser Lee

for her guidance, expertise, and invaluable insights

and to all those who give support to the completion of this project,

Thank you all.

ACKNOWLEDGEMENTS

First and foremost, I want to convey my genuine appreciation to Dr. Loh Ser Lee, my FYP supervisor. Her knowledge, tolerance, and dedication to academic success have always served as an inspiration. I truly appreciate her guidance and insightful advice throughout this project, which significantly shaped the direction and quality of my work, fostering both personal and academic growth.

Next, I would also like to extend my heartfelt thanks to my friends and classmates. Thank you for being a constant source of encouragement, providing not only valuable insights but also a listening ear during both the highs and lows of this journey.

Furthermore, I would like to thank Universiti Teknikal Malaysia Melaka (UTeM) for providing a conducive learning environment and valuable resources that have contributed to my academic progress. Studying at UTeM has enhanced my knowledge and abilities, thereby assisting me in preparing for my future career.

Last but not least, I want to express my deep gratitude to my family. Thank you for being my support system during tough times and for praying for my success. Your love means a lot to me. Your encouragement, especially during moments of self-doubt, provided the strength needed to persevere.

ABSTRACT

Continuous blood pressure (BP) monitoring is crucial for managing hypertension. However, current methods have drawbacks such as risks and discomfort for the patient. To address this, some studies have explored BP prediction through PPG and ECG signals. This project aims to develop two BP prediction models: one for systolic and one for diastolic pressure, by identifying and extracting BP-related features from PPG and ECG signals along with demographic features, using a machine learning model and Shapley Additive Explanations (SHAP). The model's performance is evaluated against AAMI and BHS standards. Two experiments were conducted, which includes identifying the best machine learning model and determining the best feature combination for BP prediction. Initially, features were extracted, and both Support Vector Regression (SVR) and Random Forest models were trained on the dataset. The results from model selection show that Random Forest performs better than SVR, hence, it is used to develop the BP prediction models. The results from feature analysis reveal that both signals and demographic features contribute to BP prediction. The inclusion of ECG signals and demographic features is found reduces the Mean Error (ME) of prediction by approximately 24.13% for SBP and 81.50% for DBP compared to using only PPG signal. In this project, SHAP feature selection is introduced, which involves ranking features according to their importance in machine learning model predictions, followed by an iterative process of removing the least important features to select the optimized feature combination based on the lowest root mean square error. The optimized feature combination is then used to develop the final BP prediction model. The final result indicates that SHAP feature selection managed to reduce the number of features used in SBP and DBP models by up to 48.72% and 50%, respectively, while still providing comparable results to the models with the full set of features. This output is expected to be beneficial for medical teams in clinical studies on blood pressure and cardiovascular diseases.

ABSTRAK

Pemantauan tekanan darah (BP) yang berterusan adalah penting untuk menguruskan hipertensi. Walau bagaimanapun, kaedah yang sedia ada mempunyai kelemahan seperti risiko dan ketidakselesaan bagi pesakit. Untuk menangani isu ini, beberapa kajian telah meneroka ramalan BP melalui isyarat PPG dan ECG. Projek ini bertujuan untuk membangunkan dua model ramalan BP: satu untuk tekanan sistolik dan satu untuk tekanan diastolik, dengan mengenal pasti dan mengekstrak ciri-ciri berkaitan BP daripada isyarat PPG dan ECG bersama dengan ciri-ciri demografi, menggunakan model pembelajaran mesin dan Shapley Additive Explanations (SHAP). Prestasi model dinilai berdasarkan piawaian AAMI dan BHS. Dua eksperimen telah dijalankan, termasuk mengenal pasti model pembelajaran mesin terbaik dan menentukan kombinasi ciri terbaik untuk ramalan BP. Pada awalnya, ciri-ciri telah diekstrak dan kedua-dua model Support Vector Regression (SVR) dan Random Forest telah dilatih pada set data. Hasil pemilihan model menunjukkan bahawa Random Forest berprestasi lebih baik daripada SVR, oleh itu ia digunakan untuk membangunkan model ramalan BP. Hasil analisis ciri menunjukkan bahawa kedua-dua isyarat dan ciri demografi menyumbang kepada ramalan BP. Kemasukan isyarat ECG dan ciri demografi didapati mengurangkan Ralat Purata (ME) ramalan sebanyak kira-kira 24.13% untuk SBP dan 81.50% untuk DBP berbanding hanya menggunakan isyarat PPG. Dalam projek ini, pemilihan ciri SHAP diperkenalkan, yang melibatkan peringkat ciri mengikut kepentingannya dalam ramalan model pembelajaran mesin, diikuti dengan proses iterasi mengeluarkan ciri yang paling tidak penting untuk memilih kombinasi ciri yang dioptimumkan berdasarkan ralat min kuasa dua terendah. Kombinasi ciri yang dioptimumkan kemudian digunakan untuk membangunkan model ramalan BP akhir. Hasil akhir menunjukkan bahawa pemilihan ciri SHAP berjaya mengurangkan bilangan ciri yang digunakan dalam model SBP dan DBP sehingga 48.72% dan 50% masing-masing, sambil masih memberikan hasil yang setanding dengan model dengan set ciri penuh. Hasil ini dijangka memberi manfaat kepada pasukan perubatan dalam kajian klinikal mengenai tekanan darah dan penyakit kardiovaskular.

.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

BP	-	Blood pressure
SBP	-	Systolic blood pressure
DBP	-	Diastolic blood pressure
mmHg	-	millimeters of mercury
PPG	-	Photoplethysmography
ECG	-	Electrocardiogram
ABP	-	Arterial blood pressure
SHAP	-	SHapley Additive exPlanations
AAMI	-	Association for the Advancement of Medical Instrumentation
BHS	-	British Hypertension Society
MAE	-	Mean absolute error
MSE	-	Mean square error
RMSE	-	Root mean square error
ME	-	Mean error
SD	-	Standard deviation
PTT	-	Pulse transmit time
PEP	-	Pre-ejection period
PAT	-	Pulse arrival time
PWV	-	Pulse wave velocity
LASI	-	Large artery stiffness index
IPA	-	Inflection point area
AI	-	Argument index
LSTM	-	Long-Short Term Memory
CNN	-	Convolutional Neural Network
SVR	-	Support Vector Regression
UCI	-	University of California, Irvine

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Blood pressure (BP) refers to the measurement of the pressure or force exerted by blood inside human arteries. These arteries deliver oxygen-rich blood pumped by the heart to the entire body. It is measured in millimetres of mercury (mmHg) and is usually given as systolic blood pressure (SBP) over diastolic blood pressure (DBP). SBP indicates the maximum BP in the arteries when the heart contracts, while DBP reflects the minimum BP when the heart muscle is resting between contractions [1]. Figure 1.1 illustrates the BP categories based on SBP and DBP values.

BLOOD PRESSURE CATEGORY	SYSTOLIC mm Hg (upper number)	and/or	DIASTOLIC mm Hg (lower number)
NORMAL	LESS THAN 120	and	LESS THAN 80
ELEVATED	120 – 129	and	LESS THAN 80
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 1	130 – 139	or	80 – 89
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 2	140 OR HIGHER	or	90 OR HIGHER
HYPERTENSIVE CRISIS (consult your doctor immediately)	HIGHER THAN 180	and/or	HIGHER THAN 120

Figure 1.1: BP category based on SBP and DBP [1].

High BP, or hypertension, is often called the silent killer because it typically has no symptoms. According to the World Health Organization (WHO), an estimated 1.28 billion adults between the ages of 30 to 79 worldwide have hypertension, with an estimated 46% being unaware of their illness, and less than half (42%) receive a diagnosis and appropriate treatment [2]. In the United States, hypertension was a primary cause of 691,095 deaths in 2021 [3]. Therefore, continuous BP monitoring is crucial, as it serves as the primary method for detecting hypertension.

Hypertension is a major risk factor that causes cardiovascular diseases, including heart attacks and strokes. Ischemic heart disease (IHD) is a type of cardiovascular disease that significantly increases with high SBP [4]. According to the Statista Research Department, ischemic heart disease was the leading cause of death in Malaysia in 2022, with more than 20 thousand deaths [5], as illustrated in Figure 1.2. This highlights the importance of continuous BP monitoring in the early detection and monitoring of cardiovascular disease in real-time.

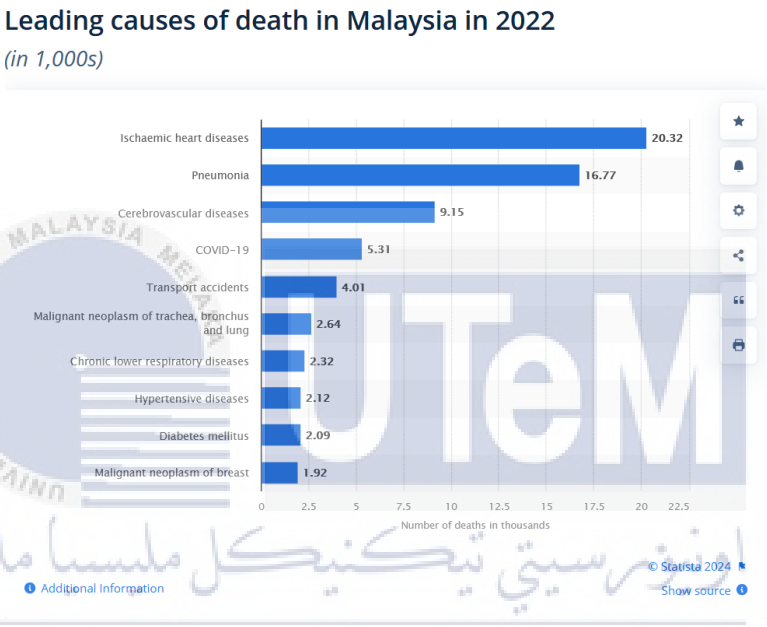


Figure 1.2: Leading causes of death in Malaysia in 2022[5].

Furthermore, for patients with a prior diagnosis of hypertension, continuous BP monitoring is crucial for assessing the effectiveness of medication and lifestyle modification. These BP readings can guide healthcare professionals in making long-term clinical decisions and assessing the effectiveness of management strategies [6]. Hypertension can be controlled by lifestyle change, as inadequate BP control was linked to weight gain, lack of physical activity, and high salt consumption [7]. Regularly monitoring BP offers feedback on the effectiveness of lifestyle modification.

Low BP, or hypotension, on the other hand occurs when BP reading is lower than 90/60 mmHg. Unlike hypertension, hypotension may cause symptoms such as

confusion, dizziness, and fainting. Patients often experience arterial hypotension during and after surgery, which could affect their recovery [8]. Hypotension can cause poor blood flow (hypoperfusion) to essential organs, which may result in organ failure [8]. BP monitoring is vital for promptly detecting and predicting hypotension, enabling the clinician to offer timely treatment.

These insights have inspired and motivated the initiation of a project aimed at developing a non-invasive, continuous BP prediction model, with a specific focus on predicting and managing hypertension or hypotension, particularly in the elderly population.

1.2 Problem Statement

There are two main methods to measure BP: invasive and non-invasive. The invasive method is used to obtain the arterial BP by inserting a catheter into a peripheral artery, offering continuous BP monitoring. However, invasive BP monitoring requires technical expertise and has associated risks, such as excessive bleeding from accidental disconnection and traumatic nerve damage [9].

On the other hand, the non-invasive BP monitoring method offers safer and more comfortable monitoring. The most frequently used method involves a cuff-based instrument that wraps around the patient's arm and inflates to obtain BP values, either manually using mercury sphygmomanometers and a stethoscope or automatically using digital sphygmomanometers. However, this method only provides intermittent values, which have limitations in continuous monitoring. Moreover, frequent cuff inflation can be uncomfortable, particularly during sleep [10].

In order to achieve the objective of continuous BP monitoring, various methods have been proposed such as the arterial tonometry method and volume-clamp method. Arterial tonometry involves using a tonometer that applies light pressure to slightly flatten the artery to monitor BP waveforms [11]. Despite offering continuous BP measurement, this method has a few disadvantages, such as the need

for precise device placement and the requirement for no movement during measurements. Figure 1.3 shows the working principle of arterial tonometry.

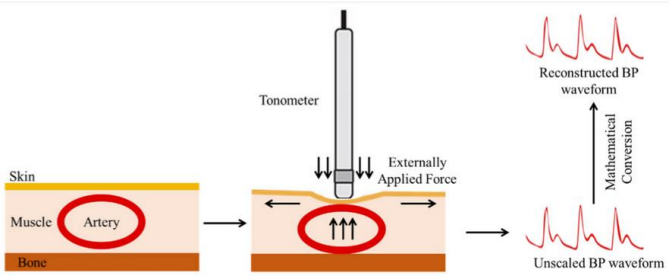


Figure 1.3: Arterial tonometry method [11].

For the volume-clamp method, BP is measured using the inflatable finger cuff with a photoplethysmography sensor and a pressure controller unit [12]. The pressure is applied to the finger cuff until the photoplethysmography signal becomes constant, indicating that the blood volume under the cuff is constant to measure the BP, as shown in Figure 1.4. While this method allows for continuous BP measurement, it comes with the drawbacks of being expensive and potentially uncomfortable for the patient.

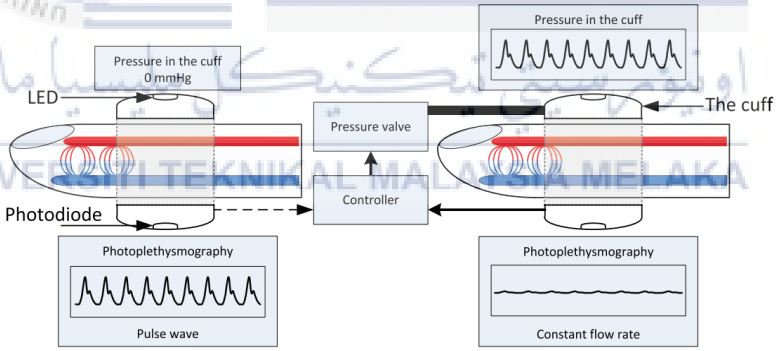


Figure 1.4: Volume-clamp method [12].

In order to achieve the objective of continuous BP monitoring and ensure stable comfort for patients, many researchers in the literature have suggested cuff-less BP estimation using Photoplethysmography (PPG) and Electrocardiogram (ECG) signals. The idea of using PPG and ECG signals to estimate BP is inspired by the relationship between pulse transmit time (PTT) or pulse arrival time (PAT), pulse wave velocity (PWV), and BP, which will be discussed further in the literature

review, under Section 2.2.5. However, due to individual variability, frequent calibration is needed [13].

More recently, with the advancements in machine learning and deep learning, an increasing number of studies are employing these methods to predict BP. One advantage of the deep learning approach is its capability to automatically learn relevant features from raw PPG and ECG signals without the need for manual feature engineering in estimating BP. However, using deep learning has its disadvantages, including being less interpretable due to its 'black box' and complex architecture, requiring a large dataset for training, and being computationally costly, which is unsuitable when battery life is limited [14].

In traditional machine learning approaches, feature selection is crucial for ensuring accurate predictions. Numerous studies have identified various features with strong predictive value for estimating BP. However, there is no consensus on the optimal combination of features for BP prediction.

Recent research has highlighted that data leakage is a common issue overlooked in many BP studies, leading to overly optimistic results [15, 16, 17]. When measures are taken to prevent data leakage, ensuring that records from the same subject do not appear in both the training and testing sets, many deep learning and traditional machine learning models fail to fulfill the AAMI and BHS criteria.

PPG signal has been primarily used in BP prediction model by both machine learning and deep learning models. Even though ECG has also been extensively used to provide diagnostic information about the blood pressure status, little studies have been carried out to document the relationship between combination of features from PPG and ECG for blood pressure prediction using traditional machine learning while addressing data leakage issues which was found in year 2023.

1.3 Objective

The objectives of this project are:

i. To extract features from PPG and ECG signals which associated with BP through their derivatives, as well as their demographic features.
ii. To develop a blood pressure prediction model based on machine learning model, utilizing an optimized feature combination identified through SHapley Additive exPlanations (SHAP) with machine learning model.
iii. To evaluate the performance of the model using mean error (ME), standard deviation (SD), and cumulative percentage error based on the criteria set by the Advancement of Medical Instrumentation (AAMI) standard and the British Hypertension Society (BHS) protocol.

1.4 Scope

The scopes of this project are as follows:

i. The PPG and ECG signals utilized in this project are obtained from Kaggle (https://www.kaggle.com/datasets/weinanwangrutgers/pulsedb-balanced-training-and-testing/code). This dataset consists of simultaneous measurements of Arterial Blood Pressure (ABP), finger-PPG, and channel II ECG signals at a 125 Hz sampling rate.
ii. This project involves only simulation which using MATLAB for feature extraction and Python to develop the BP prediction model.
iii. This project involves relatively clean signals from the dataset, excluding those signals that fail to detect the points.
iv. The SBP limit is set to between 80 to 180 mmHg, while DBP limit is set to between 50 to 120 mmHg, excluding very high or low BP values.
v. The machine learning models considered in this project are Random Forest and SVR.
vi. This project involves only 500 subjects, which is sufficient for the AAMI standard and BHS protocol that require at least 85 subjects.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

This chapter begins with the introduction of PPG and ECG signals in terms of the measurement method and the relationship between these signals and BP. Next, there are many features can be obtained from PPG and ECG signals, but not all of them significantly correlate to BP measurement. Hence, the features for BP prediction and feature selection methods will also be briefly discussed in this chapter. The last part of this chapter presents the discussion on the existing BP prediction models that had been proposed by researchers.

2.2 PPG and ECG signals

2.2.1 Definition of PPG signal

Photoplethysmography (PPG) is a non-invasive optical measurement that uses the light source and photodetector at the surface skin to detect the volumetric difference of the blood in peripheral circulation [18]. The signal obtained from this measurement is called PPG signal. This method is getting famous due to its convenience, non-invasiveness, and inexpensive diagnostic tools.

The PPG signal exhibits a correlation with BP. The PPG signal consists of two parts: the upper part represents the contraction of the heart (systolic), and the underside represents cardiac expansion (diastolic) [19]. Between the systolic and diastolic phases, there is a dicrotic notch, marking the transition between these

phases. Nevertheless, the diastolic peak and dicrotic notch may become less prominent as people become older [20].

The first derivative and second derivative of the PPG signal are called the velocity photoplethysmogram (VPG) and acceleration photoplethysmogram (APG), respectively. The fiducial points of the PPG signal and its derivatives signals are important in extracting features related to BP [21]. According to [20], there are four fiducial points for PPG (O, S, N, D), four fiducial points for VPG (w, x, y, z), and six fiducial points for APG (a, b, c, d, e, f). Figure 2.1 shows the fiducial points of the PPG signal and its derivatives signals.

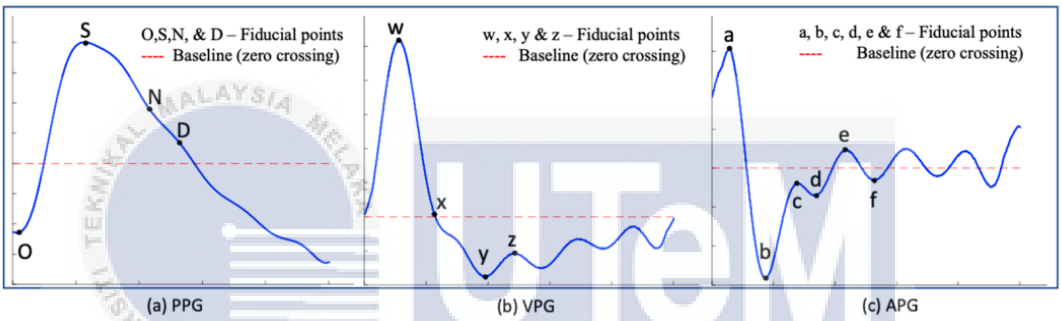


Figure 2.1: PPG signal and its derivatives signals [20].

2.2.2 Some applications of PPG signal in BP prediction

Some studies on continuous BP monitoring using only PPG signal had been proposed. One of the approaches involves using the PPG signal and a two-element Windkessel model to estimate both SBP and DBP [22, 23]. However, this method relies on a fixed assumption for some parameters, which reduces its robustness.

In [24], a novel smartwatch is presented, which obtains two PPG signals from the finger and wrist to estimate BP. This is achieved by measuring the Pulse Transmit Time (PTT) and using this parameter together with heart rate in a linear regression model to estimate BP. However, it should be noted that this device requires continuous calibration for hypertensive patients.

2.2.3 Definition of ECG signal

An Electrocardiogram (ECG) is a non-invasive medical test that records the electrical activity of the heart by placing electrodes on specific parts of the body [25]. The primary focus of ECG analysis is on diagnosing cardiovascular diseases. However, recent studies show an upward trend in an expanded application of biometric identification due to its unique features in an individual's ECG [26].

Typical ECG has five waves, which are P, Q, R, S, and T. Figure 2.2 shows the ECG signal together with the fiducial points. Muscle contraction involves electrical changes that known as 'depolarization', whereas the state of relaxation is known as 'repolarization.' The P wave represents the depolarization of the atria, the QRS complex represents the depolarization of the ventricles, and the T wave represents the repolarization of the ventricles [27].

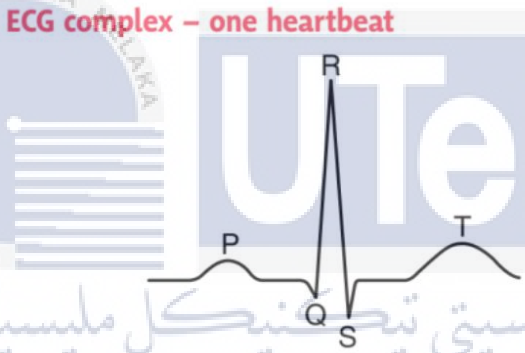


Figure 2.2: ECG signal in one beat [27].

A study on estimation BP using only ECG signal was conducted based on physiological principles governing the heart's electrical and mechanical activities, known as Mechano-Electric Coupling (MEC) [28]. The findings of the study revealed a non-linear relationship between the ECG signal and BP.

2.2.4 Some applications of ECG signal in BP prediction

Compared to the PPG signal, fewer studies in the literature focus on the use of only ECG signal for BP estimation. In [29], the study demonstrated that the use of PPG signal alone in SBP prediction outperforms the one that used ECG signal alone and suggest that PPG signal alone may be further explored as a potential single

source for BP prediciton. However, in [30], the study conducted BP estimation using machine learning based solely on ECG signals. Complexity analysis was performed for ECG feature extraction, and the results showed that applying a probability distribution-based calibration could achieve results close to those of a certified medical device for BP estimation, highlighting the potential of ECG signal in BP prediction.

Additionally, some studies also found that the ECG signal is crucial for enhancing BP prediction. In [31], certain ECG features were proposed for BP prediction. The results from the Genetic Algorithm feature selection indicate the preservation of specific ECG features, emphasizing the significant potential of the ECG signal in predicting BP. Besides that, in [32], the study suggested that the inclusion of the ECG signal with the PPG signal leads to improved performance of the deep learning network and allows for better generalization.

2.2.5 PTT/PAT: A PPG and ECG derived parameter

PAT is the sum of PTT and the pre-ejection period (PEP). PTT measures the time taken for a pressure pulse to travel between two arterial sites, typically assessed using two PPG signals at different locations [33, 34]. Meanwhile, PEP accounts for the time delay between the electrical depolarization of the left ventricle and the actual of mechanical ventricular ejection [33, 34]. Therefore, PAT includes the pressure wave propagation time with the duration between the initiation of electrical activity and the subsequent mechanical motion of the heart.

PAT can be measured using the time difference between *R* peak of the ECG signal to a point in PPG signal [33, 34]. In the literature, the terms PAT and PTT are used interchangeably. However, most of the studies use the time difference between *R* peak of ECG signal to systolic peak, foot, or maximum slope of PPG which refer to PAT. Figure 2.3 shows the difference between PTT and PAT along with the methodology for its determination.

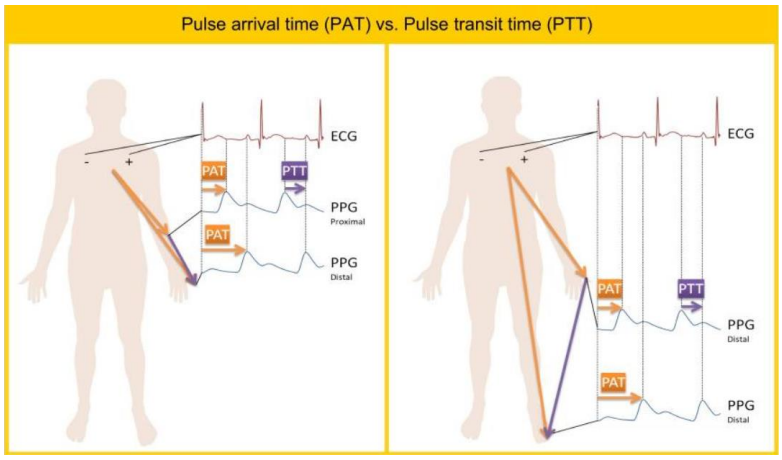


Figure 2.3: Difference between PAT and PTT [33].

The relationship between BP and PAT/PTT/pulse wave velocity (PWV) is based on the mechanism of arterial wall and the transmission of pressure waves within the arteries [35]. The elastic modulus E of arteries changes during the cardiac cycle due to the expansion and contraction caused by BP. The relationship between arterial elasticity and BP is given by Eqn. (2.1).

$$E = E_0 e^{\alpha P} \tag{2.1}$$

where E_0 denotes the Young's modulus for zero arterial pressure while α denotes vessel parameter.

PWV is the velocity of pressure pulse travel through the network of arteries. PWV is determined by the elasticity of arteries. By assuming the artery to be elastic tube, the Moens-Kortweg equation can be used to find relationship between PWV with E as shown in Eqn. (2.2).

$$PWV = \sqrt{\frac{hE}{\rho d}} \tag{2.2}$$

where h , ρ , and d represents thickness, blood density, and diameter, respectively.

PWV can be measured by dividing length between two measurement sites, L by time delay (PTT or PAT) as shown in Eqn. (2.3).

$$PWV = \frac{L}{time\ delay} \tag{2.3}$$

Finally, by combining Eqns. (2.1), (2.2), and (2.3), the relationship between BP and PAT/PTT/PWV can be expressed as in Eqn. (2.4).

$$PWV = \frac{L}{time\ delay} = \sqrt{\frac{hE_0 e^{\alpha P}}{\rho d}} \tag{2.4}$$

Based on this relationship, several mathematical models have been derived to predict BP using these time delays, including logarithmic, linear, inverse, and inverse square models [35]. Although PTT/PAT shows a strong correlation with BP, its accuracy is significantly influenced by individual physiological properties where frequent calibration is needed [13].

2.3 Feature for BP prediction

There are multiple features that can be extracted from the PPG and ECG signals, different researchers used different combinations of features to predict the BP. It is crucial to choose the proper features since incorporating irrelevant features during training may slow down the system, makes it more expensive to operate, and results in less accurate predictions.

2.3.1 A Review of Literature on Features for BP Prediction

PAT or PTT, as mentioned in earlier sections, has a strong correlation with BP. Hence, this feature is widely used in many research for predicting BP. It is the time difference between the *R* peak of the ECG signal and a specific point of the PPG signal. This point can be up to the peak of the PPG signal [13, 14, 31, 36, 37],

the maximum slope of the PPG signal [13, 31, 36, 38, 39], or the onset of the PPG signal [13, 31, 36, 38]. Figure 2.4 shows an example of extraction of PAT.

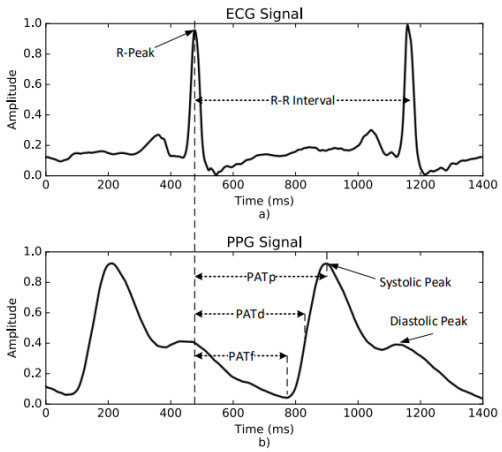


Figure 2.4: Extraction of PAT [36].

Heart rate is another feature commonly used in many studies to estimate BP [13, 21, 31, 36, 37, 38, 40]. If heart rate increased while the cardiac output and peripheral resistance remain constant, it may lead to an elevation in BP, and vice versa [37]. Besides that, the cardiac output can be correlated with PTT through heart rate indicating a correlation between heart rate and BP [38]. This feature can be obtained from R-R interval in ECG signal using Eqn. (2.5), where f_s is sampling frequency and RR is time difference between two R peaks of ECG.

$$Heart\ rate = \frac{60 \times f_s}{RR} \tag{2.5}$$

PPG_K value is used in some research as one of the features for BP prediction [37, 38, 39]. PPG_K value is a significant parameter in cardiovascular research and clinical practice [37]. It reflects factors such as peripheral resistance, arterial wall elasticity, and blood viscosity which play a role in affecting BP [38]. By referring to one complete cycle PPG signal in Figure 2.5, PPG_K value can be calculated using equation Eqn. (2.6) and Eqn. (2.7).

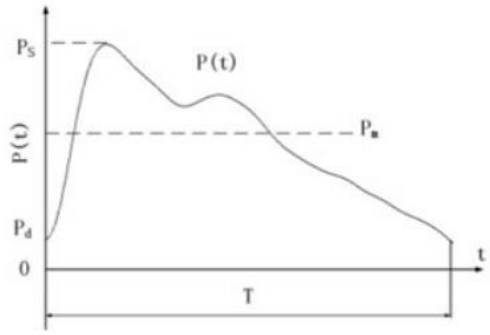


Figure 2.5: One complete cycle of PPG signal [39].

$$PPG\_Kvalue = \frac{p_m - p_d}{p_s - p_d} \tag{2.6}$$

$$p_m = \frac{1}{T}\int PPG(t)dt \tag{2.7}$$

where , p_d, p_s and T are shown in Figure 2.5.

Photoplethysmogram intensity ratio (PIR) is the ratio of PPG peak intensity, I_H to PPG bottom intensity, I_L and be used by other researchers as one of the features to estimate BP [31, 39, 41]. PIR is influenced by changes in arterial diameter and able to track low-frequency components in BP which is a crucial DBP indicator [41]. Figure 2.6 shows the extraction of PIR.

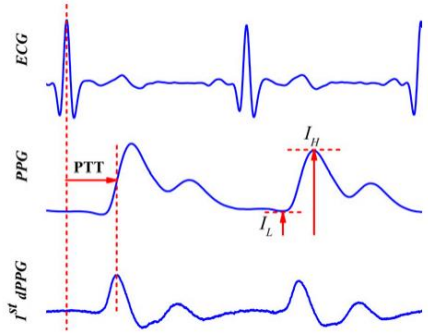


Figure 2.6: Extraction of PIR from PPG signal [41].

Features obtained from the APG have previously been shown to indicate age-related stiffness in the arteries. The ratio b/a increases with age, whereas c/a, d/a, and e/a decrease with age [29]. These relationships were implemented into a single feature using the ageing index (AGI), $A_{b-c-d-e/a}$ [29]. This feature also holds a high rank in the study [21]. Next, in [14], the study suggested that, from their self-collected signals, only points a and b are observed, while points c, d, and e are less prominent. Therefore, ratio b/a was used in their study. Besides that, this study also used the time difference between start and point a of the APG signal, T_{Oa} as a feature to predict BP, and in [21], T_{Oa} was ranked in the top 20 for SBP. The locations of these fiducial points are illustrated in Figure 2.7.

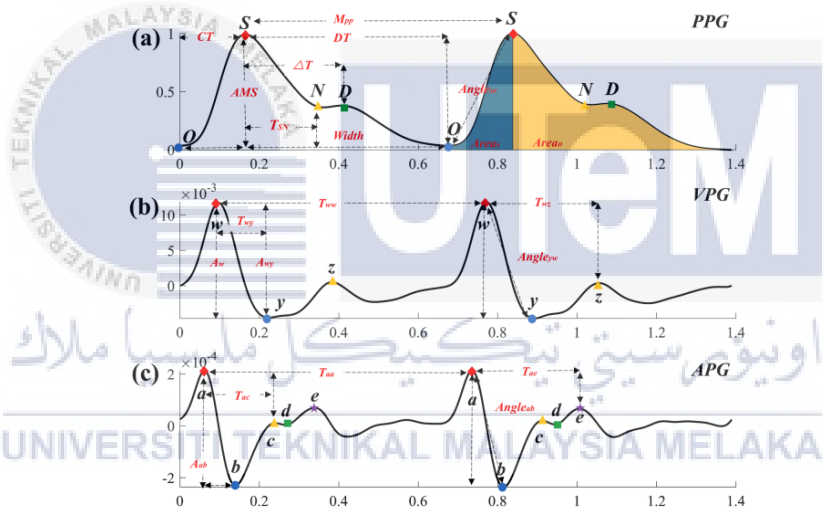


Figure 2.7: Example of features extracted from PPG, VPG, and APG signals [21].

PPG morphology and dynamic features have been used in many studies. For example, slope between the onset of PPG to the systolic of PPG, $Cslope$ [29, 37, 38], and the time difference between onset and systolic peak of PPG, T_{Os} [14, 21, 38]. Additionally, 50% width of PPG signal, Width_{50}, is another feature that has been used for predicting BP [29, 42, 43]. In [43] Width_{50} is located at the top feature for SBP after feature selection. Large artery stiffness index (LASI), the time difference between systolic peak and diastolic peak had also been used as one of the features to

predict BP [13, 21, 31, 36]. Argument index (AI) is the ratio between diastolic peak to the systolic peak of PPG [13, 31, 36, 37]. Inflection point area (IPA) is the ratio between two pulse area divided by dicrotic notch [29], while in [13, 31, 36] the study used the area $S1$, $S2$, $S3$, and $S4$ instead of IPA. Figure 2.8 represents the LASI, $S1$, $S2$, $S3$, $S4$, x and y in PPG signal, where x and y correspond to the diastolic and systolic peaks of PPG signal, respectively.

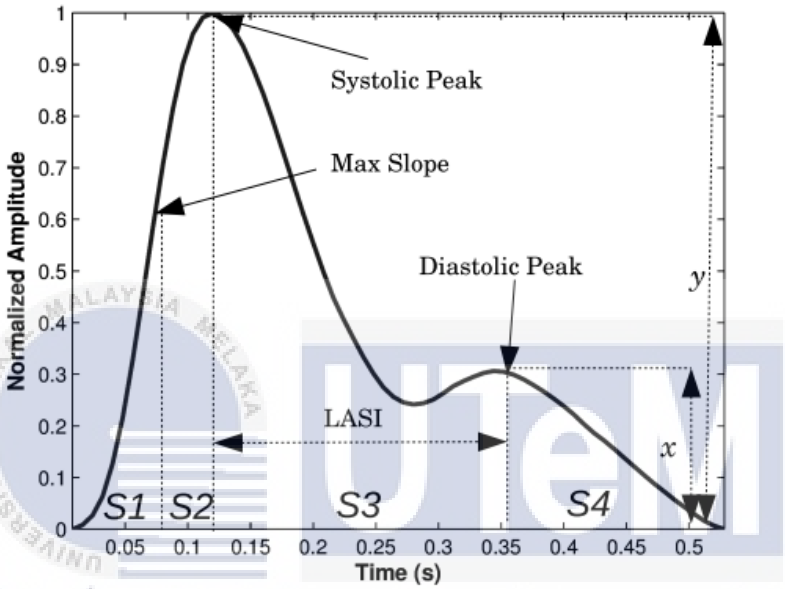


Figure 2.8: LASI, $S1$, $S2$, $S3$, $S4$, x and y in PPG signal [13].

Kurtosis, a statistical measure assessing data distribution, plays a crucial role in BP prediction as concluded in [29]. In this study, kurtosis of the PPG signal achieved the highest feature importance after assessment using SHAP. Furthermore, another study [21] employed SHAP to select important features for BP prediction. Among the top 6 selected features, which encompassed the morphology and dynamic features of PPG, VPG, and APG, were: the slope between the dicrotic notch and diastolic peak, $Angle_{ND}$, the slope between the systolic peak and diastolic peak, $Angle_{SD}$, the time between the dicrotic notch and diastolic peak, T_{ND}, the time difference between point w in VPG and point c in APG, T_{wc}, the slope between points z and y in VPG $Angle_{zy}$, and the slope of points e and d in APG, $Angle_{ed}$. The details of the extraction of some of these features are shown in Figure 2.7.

Hjorth parameters, including Hjorth mobility and Hjorth complexity, have been employed in [30] for predicting BP using only ECG signal. Hjorth mobility calculates the signal's mean frequency, while Hjorth complexity calculates the signal's bandwidth [29]. Additionally, these parameters are utilized in [40] for both PPG and ECG signals. Hjorth mobility demonstrated significant importance in BP prediction after being assessed using SHAP in [29]. The Eqn. (2.8) and Eqn. (2.9) show the equation for Hjorth mobility and Hjorth complexity, where $var()$ is variance and x is ECG segment.

$$Hjorth\ mobility = \sqrt{\frac{var(x')}{var(x)}} \tag{2.8}$$

$$Hjorth\ complexity = \frac{Mobility\ (x')}{Mobility(x)} \tag{2.9}$$

Futhermore, in [44], the prediction of BP using time domain-based features achieved better accuracy compared to other feature extraction techniques. The time domain-based features used in this study include the peak-to-peak interval of the PPG signal or cardiac period (CP), systolic upstroke time (SUT), diastolic upstroke time (DT), the width of the PPG pulse at 10%, 25%, 33%, 50%, 66%, and 75% of pulse height (DW10+SW10, DW25+SW25, DW33+SW33, DW50+SW50, DW66+SW66, and DW75+SW75), the width of the PPG pulse at the diastolic part at 10%, 25%, 33%, 50%, 66%, and 75% of pulse height (DW10, DW25, DW33, DW50, DW60, and DW75), and the ratio of the width of the PPG pulse at the diastolic part to the systolic part (DW10/SW10, DW25/SW25, DW33/SW33, DW50/SW50, DW60/SW60, and DW75/SW75). Moreover, in [45], all these time domain-based features were also used to predict BP. Additionally, the study proposed the ratio of the amplitude of APG as additional features, which included the ratios b/a, c/a, d/a, and e/a, as well as the aging index (AGI). Figure 2.9 shows the time domain-based features.

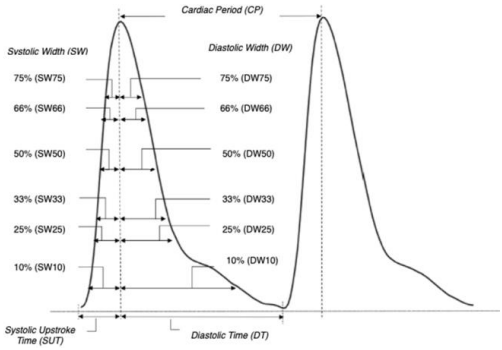


Figure 2.9: Time domain-based features of PPG signal [44].

Demographic features such as BMI, age, and weight are found to be important for BP prediction in [21]. The researchers also mentioned that although the importance ranking of height and gender is lower, these features should also be included to improve model generalization. Furthermore, in [46], feature selection using ReliefF retained the demographic features of age, weight, and BMI, highlighting their importance for BP prediction. Additionally, the researchers combined the time-domain features of the PPG signal with demographic features to create demographic time-domain features. From ReliefF feature selection, the retained features include the ratio of BMI to the systolic peak time of the PPG pulse (BMI/t_1), the ratio of weight to the PPG pulse interval (Weight/t_{pi}), the ratio of weight to the peak-to-peak interval of the PPG signal (Weight/t_{pp}), the ratio of weight to the systolic peak time of the PPG pulse (Weight/t_1), and the ratio of BMI to the peak-to-peak interval of the PPG signal (BMI/t_{pp}).

In [31], the researchers proposed using the Womersley number (α) and certain ECG features, together with features from the PPG signal, to predict BP. Genetic algorithms were used to explore the relevance of the proposed features. The results showed that the Womersley number that extracted using second method (α_n) and some of the ECG features remained after feature selection. These ECG features include the QRS complex, the QT interval (QT), the QT interval corrected for heart rate (QT_c), the systolic-diastolic time interval (SDI), the new systolic-diastolic interval (SDIn), and the amplitudes of the P, Q, R, S, and T waves of the ECG signal. Some ECG features are shown in Figure 2.10 and the equation of α_n are shown in Eqn. (2.10) and.

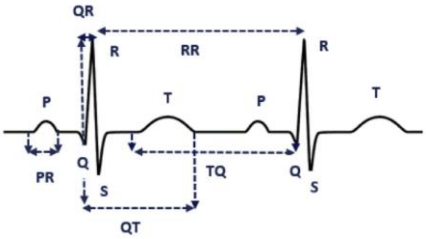


Figure 2.10: Some of the ECG features [31].

$$\alpha_n = R\sqrt{\frac{\omega\rho}{b}} \tag{2.10}$$

where R is the valley amplitude of the PPG signal, ω is the frequency of heart rate, ρ is the density of blood which assume to be 1060 kg/m^3 , and b is equal to $\frac{1}{GH}$, where GH is the magnitude of the first derivative of pulse signal at point C as shown in Figure 2.11.

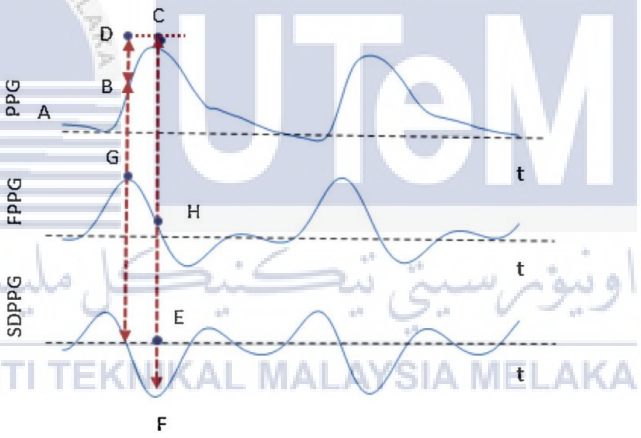


Figure 2.11: The extraction of GH [31].

In [47], the morphology features of ECG is used with the PAT to predict the BP, this features include time feature like time for a full QRS wave of each cardiac cycle (QRS), time from P peak to R peak of each cardiac cycle (RP), Time from R peak to T peak of each cardiac cycle (RT), Time from P peak to Q peak of each cardiac cycle (PQ), Time from S peak to T peak of each cardiac cycle (ST), time from P peak to T peak of each cardiac cycle (PT) and amplitude feature like P, R, T , ratio of peak of T to R (RT ratio), R peak amplitude difference from P peak amplitude (RP diff). The morphological features of ECG signal are shown in Figure 2.12.

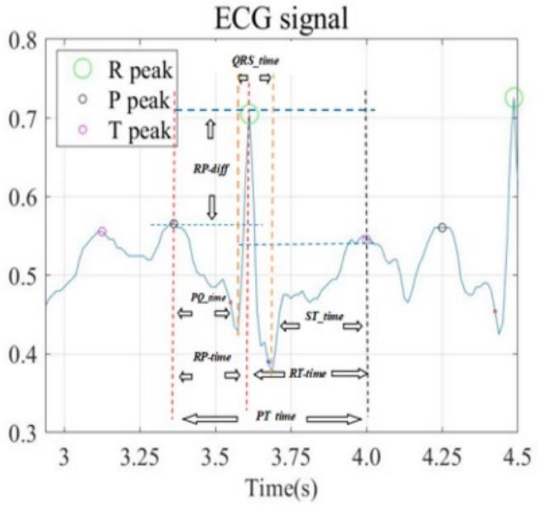


Figure 2.12: Morphology features of ECG signal [47].

2.3.2 Feature selection method

Different researchers in the literature had employed various feature selection methods in selecting the relevant features for BP prediction. As there is non-linearity between physiological features and BP, the feature selection method needs to account for both linear and non-linear relationships. Some relevant features will be missed if non-linear relationship approaches are not taken into consideration [48].

2.3.2.1 Mean Influence Value feature selection

Several studies utilize the Mean Impact Value (MIV) in feature selection. MIV is a quantitative measure of how each feature impacts the model's predictions. This is determined by making slight changes to the input values and observing how these changes impact the model's output [49]. Figure 2.13 shows the process of MIV algorithm.

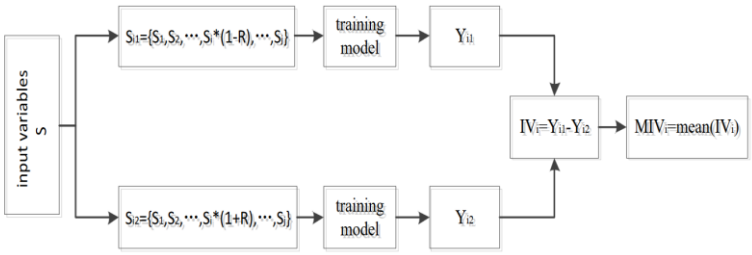


Figure 2.13: The process of MIV algorithm [49].

MIV was used in [49, 50], to select the optimized feature combination for the genetic-algorithm backpropagation neural network to improve efficiency and predictive accuracy of the model. These studies suggested that the used of MIV contributes to the construction of more accurate and simpler neural network models by focusing on the features that have the greatest influence on BP.

Moreover, in another study [37], MIV was used for feature selection in a hybrid machine learning GA-SVR model. The study suggested that MIV is able to provide strong evidence and describe the non-linear relationship between features and BP. Additionally, after applying MIV, the Mean Squared Error (MSE) of the model is reduced from 38.33 mmHg to 30.98 mmHg for SBP, and from 5.73 mmHg to 4.68 mmHg for DBP by choosing the features with 90% of cumulative contribution ratio to train the model.

2.3.2.2 Mutual Information feature selection

Mutual Information (MI) is a method used in certain studies to select optimized feature combination. MI is a measure of the mutual dependence between two random variables, with higher values indicating stronger dependence and vice versa [38]. In [43], the study aims to observe the relationship between features extracted from PPG and ECG with BP, using correlation coefficients, cross-sample entropy, and MI. The study suggested that MI is able to capture nonlinear statistical dependencies between the features and BP, where this is not achievable by correlation coefficients.

In another study [38], the Gaussian Copula Mutual Information (GCMI), a specific MI formulation tailored for Gaussian-distributed random variables, was used to select the optimized feature combination. The algorithm calculated GCMI values between each feature and BP, iteratively removing features with lower GCMI values until the threshold is reached. By employing the optimized feature combination, it manages to reduce the MAE±SD from 9.12±9.83 mmHg to 7.93±9.12 mmHg for SBP, and from 8.31±9.23 mmHg to 7.63±8.61 mmHg for DBP.

2.3.2.3 Genetic Algorithm feature selection

Genetic Algorithm (GA) is a well-known optimization technique inspired by the process of natural selection. This algorithm has been employed in various studies to find optimal feature combinations for BP prediction. In [39], data mining techniques, including GA, were utilized to identify indicators reflecting changes in BP before model construction. In this study, GA worked by iteratively evolves and selects feature subsets aiming to maximize the fitness function, particularly the correlation coefficient. The study's results demonstrated that the proposed method is not only more accurate but also slightly improved robustness compared to the traditional PAT-PIR-based model.

Different from aforementioned study, which implemented GA before constructing the model, [31, 51, 52] employed binary GA along with the model for feature selection, a technique known as the wrapper method. In these studies, GA worked by finding an optimal binary vector, where each bit corresponds to the inclusion or exclusion of any feature, to optimize the model prediction accuracy. A study suggested that applying GA to select the optimized feature combination for Random Forest led to a reduction in MAE values [31]. Specifically, the model achieved lower MAE values, decreasing from 13.20 to 9.54 mmHg for SBP and from 9.91 to 5.48 mmHg for DBP.

Moreover, in [51], GA was employed to select optimal feature combinations as an alternative method to overcome the limitations of the moving backward

algorithm, which arise from correlations between features. By using optimized feature combination through GA, the prediction accuracy is higher compared to feature combination obtained through the moving backward algorithm, with MAE ± SD values of 5.59±0.30 mmHg for SBP and 4.45±0.16 mmHg for DBP.

However, in [52], the study indicated that incorporating GA for feature selection does not significantly enhance the model's prediction accuracy in predicting BP compared to Principal Component Analysis (PCA), a mathematical dimension reduction technique.

2.3.2.4 SHapley Additive exPlanations feature selection

SHapley Additive exPlanations (SHAP) is a game theoretical approach that explains the output of machine learning models [53]. The fundamental concept behind SHAP is to compute Shapley values which represent the influence of a feature on a model's prediction. Shapley values are computed by determining the average marginal contribution of a feature value across all possible combinations of feature sets.

In [53], SHAP is employed to interpret the most relevant features from PPG signal for BP prediction in a machine learning model. The study utilizes visualizations, such as bar plots and beeswarm plots, to interpret SHAP values. The use of SHAP provides insights into the features' relevance for BP prediction. Furthermore, in another study [29], SHAP values and ranking coefficients were employed to evaluate the individual importance of PPG and ECG features in predicting BP. The use of SHAP allowed a robust assessment of feature importance, applicable to both linear and non-linear machine learning models including Random Forest.

The aforementioned studies use SHAP to identify the relevant features for BP prediction based on their importance in machine learning output**.** In [54], SHAP was used with Random Forest to select the optimized feature combination for BP prediction and personalized recommendations for managing BP. Shapley values were

used for feature selection, reducing MAE for SBP from 5.79 mmHg to 5.34 mmHg, and for DBP from 3.95 mmHg to 3.80 mmHg.This study also demonstrated SHAP's superiority over other feature selection methods employed in that study. Although this study does not use features from PPG and ECG signals to predict BP, it still provides valuable insight into how SHAP can be used for feature selection and enhance BP prediction using the machine learning model.

Furthermore, in [21], the study integrated SHAP with Light Gradient Boosting Machine, to select an optimized feature combination from PPG and its derivatives signals for BP prediction. This approach addressed the limitations of traditional feature selection methods, which often lack of sensitivity to non-linear data and exhibits low interpretability. The application of this combined method resulted in a notable reduction in the number of features from 121 to 20 and 16 for SBP and DBP, respectively. Additionally, by using the optimized feature combination, the MAE decreased from 4.23 mmHg to 3.41 mmHg for SBP and from 2.81 mmHg to 2.17 mmHg for DBP.

2.3.3 Summary of features for BP prediction

Table 2.1 shows the summary of features for BP prediction from literature.

Table 2.1: Summary of features for BP prediction from literature.

Feature	Description
PAT peak [13, 14, 31, 36, 37]	The time difference between the *R* peak of the ECG and a specific point of the PPG signal.
PAT maxslope [13, 31, 36, 38, 39]	
PAT onset [13, 31, 36, 38]	
Heart rate [13, 21, 31, 36, 38, 39,	Heart rate of the subject

40]	$Heart\ rate = \frac{60 \times fs}{RR}$
PPG_K value [37, 38, 39]	Can be obtained from PPG using following equation. $ppg\_Kvalue = \frac{p_m - p_d}{p_s - p_d}$ $p_m = \frac{1}{T}\int PPG(t)dt$
PIR [31, 37, 39]	Ratio between PPG peak intensity to PPG bottom intensity
$A_{b-c-d-e/a}$ [21, 29, 45]	Ageing index, can be obtained using amplitude of fiducial points in APG
T_{Oa} [14, 29]	The time difference between O in PPG and a in APG
$Cslope$ [29, 37, 38]	The slope between O and S in PPG
T_{Os} [14, 21, 38]	The time difference between O and S in PPG
LASI [13, 14, 21, 31, 36]	The time difference between S and D in PPG
AI [13, 14, 31, 36, 37]	Ratio between D to S in PPG
IPA [29]	Ratio between two pulse area divided by dicrotic notch in PPG
$S1$, $S2$, $S3$,$S4$ [13, 31, 36]	Area of PPG signal
Kurtosis [29]	Kurtosis of PPG
$Angle_{ND}$ [21]	The slope between N and D in PPG
$Angle_{SD}$ [21]	The slope between S and D in PPG

T_{ND} [21]	The time difference between N and D in PPG
T_{wc} [21]	The time difference between w in VPG and c in APG
$Angle_{zy}$ [21]	The slope between z and y in APG
$Angle_{ed}$ [21]	The slope between e and d in APG
Hjorth mobility [29, 30, 40]	Represent the signal's mean frequency. $$Hjorth\ mobility = \sqrt{\frac{var(x')}{var(x)}}$$
Hjorth complexity [29, 30, 40]	Represent the signal's bandwidth. $$Hjorth\ complexity = \frac{Mobility\ (x')}{Mobility(x)}$$
DW10+SW10 [44, 45]	Width of PPG pulse at 10% amplitude
DW25+SW25 [44, 45]	Width of PPG pulse at 25% amplitude
DW33+SW33 [44, 45]	Width of PPG pulse at 33% amplitude
DW50+SW50 [29, 42, 43, 44, 45]	Width of PPG pulse at 50% amplitude
DW66+SW66 [44, 45]	Width of PPG pulse at 66% amplitude
DW75+SW75 [44, 45]	Width of PPG pulse at 75% amplitude
DW10 [44, 45]	Width of PPG pulse at 10% amplitude in diastolic part
DW25 [44, 45]	Width of PPG pulse at 25% amplitude in diastolic part

DW33 [44, 45]	Width of PPG pulse at 33% amplitude in diastolic part
DW50 [44, 45]	Width of PPG pulse at 50% amplitude in diastolic part
DW60 [44, 45]	Width of PPG pulse at 66% amplitude in diastolic part
DW75 [44, 45]	Width of PPG pulse at 75% amplitude in diastolic part
DW10/SW10 [44, 45]	Ratio of the width in the diastolic part to the width in the systolic part of the PPG pulse at 10% amplitude
DW25/SW25 [44, 45]	Ratio of the width in the diastolic part to the width in the systolic part of the PPG pulse at 25% amplitude
DW33/SW33 [44, 45]	Ratio of the width in the diastolic part to the width in the systolic part of the PPG pulse at 33% amplitude
DW50/SW50 [44, 45]	Ratio of the width in the diastolic part to the width in the systolic part of the PPG pulse at 50% amplitude
DW60/SW60 [44, 45]	Ratio of the width in the diastolic part to the width in the systolic part of the PPG pulse at 60% amplitude
DW75/SW75 [44, 45]	Ratio of the width in the diastolic part to the width in the systolic part of the PPG pulse at 75% amplitude
CP [44, 45]	Cardiac period, the peak-to-peak interval of the PPG signal
DT [44, 45]	Diastolic time of PPG
b/a [14, 29, 45]	Ratio of b to a in APG
c/a [29, 45]	Ratio of c to a in APG
d/a [29, 45]	Ratio of d to a in APG
e/a [29, 45]	Ratio of e to a in APG

BMI [21, 46]	BMI of the subject
Age [21, 46]	Age of the subject
Weight [21, 46]	Weight of the subject
Height [21, 46]	Height of the subject
Gender [21, 46]	Gender of the subject
α_n [31]	Womersley number $\alpha_n = R\sqrt{\frac{\omega\rho}{b}}$
QRS complex [31, 47]	Time duration of the QRS wave in the ECG signal
QT [31]	Time difference between Q and T wave end in the ECG signal
QT_c [31]	QT interval corrected for heart rate $QT_c = \frac{QT}{\sqrt{RR\ interval}}$
SDI [31]	Systolic Diastolic time interval
SDIn [31]	New systolic diastolic interval
P [31, 47]	Amplitude of P wave in ECG signal
Q [31, 47]	Amplitude of Q wave in ECG signal
R [31, 47]	Amplitude of R wave in ECG signal
S [31, 47]	Amplitude of S wave in ECG signal
T	Amplitude of T wave in ECG signal

[31, 47]	
RP [47]	Time difference between R and P in the ECG signal
RT [47]	Time difference between R and T in the ECG signal
PQ [47]	Time difference between P and Q in the ECG signal
ST [47]	Time difference between S and T in the ECG signal
PT [47]	Time difference between P and T in the ECG signal
RT ratio [47]	The ratio of T peak to R peak in ECG signal
RP diff [47]	The difference between R peak amplitude from P peak amplitude

2.4 BP prediction model

This section discusses the existing models of BP prediction using PPG and ECG signals. These models are separated into deep learning approach and traditional machine learning approach. Due to differences in datasets, preprocessing techniques, feature selection methods, different hyperparameters setting, and different validation method, the results vary even when the same model is employed.

2.4.1 Deep learning model for BP prediction

2.4.1.1 Long short-term memory (LSTM)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) which widely been used in predicting BP. In [55], the study used LSTM with ABP signal processing technique called Two-stage Zero-order Holding algorithm to

predict BP from PPG and ECG signals, the result demonstrated a good prediction accuracy with RMSE of 2.751 mmHg for SBP and 1.604 mmHg for DBP. The choice of LSTM was motivated by its dynamic temporal behaviour, making it suitable for continuous BP prediction. However, the study only utilized 25 records from the UCI dataset. Since the UCI dataset lacks subject information, there is a possibility that the selected records are from the same individual. This could potentially lead to an over-optimistic result due to data leakage, a topic that will be further discussed in Section 2.4.3.

Some studies use features from PPG and ECG as input to the LSTM instead of using raw signals. In [56], the study used Bidirectional LSTM (Bi-LSTM) for BP prediction with additional ballistocardiogram signal features alongside PPG and ECG features. The study suggested that Bi-LSTM is capable of grasping patterns in both forward and backward directions which outperforms LSTM. The study assessed generalization performance using Leave-One-Subject-Out (LASO) analysis, achieving MAE and RMSE of 5.82 mmHg and 6.82 mmHg for SBP, while 5.24 mmHg and 6.06 mmHg for DBP, in multi-day tests.

However, the study included only 18 subjects with 30-minutes recording, with only about 8% and 2% of the BP data falling within the hypertension stage 1 and stage 2 ranges, respectively. Moreover, the MAE and RMSE values previously mentioned were obtained using a tuned model, where the fully connected layer was trained with 20% of the data from the excluded subject.Without applying the tuning, the MAE and RMSE increased to 10.01 mmHg and 11.26 mmHg for SBP, and 5.60 mmHg and 6.52 mmHg for DBP.

Furthermore, in [57], the study proposed a model with Bi-LSTM as the first layer, followed by residual connected LSTM layers to predict BP based on features from PPG and ECG signals. The use of Bi-LSTM enhances the learning of long-term dependencies, and the residual connected LSTM layers help in overcoming the vanishing gradient problem. The proposed model has the ability to understand the connection between input features and BP output by learning patterns over time. The

study reported an MAE, SD and RMSE of 6.726 mmHg, 14.505 mmHg, and 8.051mmHg for SBP, while 2.516 mmHg, 6.442 mmHg, and 3.998 mmHg for DBP. However, the authors using 3000 record with 678202 cycle of record from UCI dataset and randomly select the training and testing set. Hence, there might be data leakage issue.

In [58], the study employed an LSTM network integrated with an autoencoder aiming to convert raw PPG signals into a continuous ABP signal. LSTM was chosen in this study to replace feed-forward neural network as the base of autoencoder due to its ability to handle varying length series efficiently. The result demonstrated strong performance in BP prediction with MAE, SD, and RMSE of 4.05 mmHg, 4.60 mmHg, and 5.25 mmHg for SBP, while 2.41 mmHg, 3.11 mmHg, and 3.17 mmHg for DBP. However, this study uses 5289 records with 250000 segments of signals from the UCI dataset. Hence, there might be a possibility of data leakage.

2.4.1.2 Hybrid Convolution Neural Network with Long short-term memory (CNN+LSTM)

Some studies in literature combined Convolution Neural Network (CNN) with LSTM to form a hybrid deep learning model (CNN+LSTM) for BP prediction. CNN offers the advantage of being able to learn relevant features across different scales, and the weight-sharing mechanism helps minimize memory usage compared to fully connected networks [59].

In [59], two CNNs were used to extract morphological features from PPG signals to make initial SBP and DBP predictions. The next stage was followed by two-layer stacked LSTMs to capture temporal dependencies and enhance predictions by incorporating the dynamic relationship between SBP and DBP. Combining CNN with LSTM improved accuracy by considering both local and temporal variations compared to using only CNN. The study achieved ME±SD and MAE of +1.91±5.55 mmHg and 3.97 mmHg for SBP, while +0.67±2.84 mmHg and 2.10 mmHg for DBP. However, in this study, 200 records were utilized. Each record was divided into three segments comprising 70%, 10%, and 20% of the record's length for training,

validation, and testing purposes, respectively. This method raises concerns regarding potential data leakage.

Next, CNN+LSTM was used to design a model called ‘PP-Net’ in [60], for simultaneous prediction of SBP, DBP, and heart rate based on a single channel of PPG signal. This model eliminates the need for manual feature selection and extraction, making it less complex. The study reported ME±SD of -1.25±5.65 mmHg and 1.55±5.41 mmHg for SBP and DBP, respectively. LSTM layers are incorporated into the network to address the long chain problem that arises in CNN when dealing with time series data. However, the study also mentioned deep learning algorithms are power consuming and intensive memory which may restrict the deployment on mobile devices. In this study, only 1557 records are utilized out of a total of 12000 from the UCI dataset. Consequently, there is a potential risk of data leakage.

Aforementioned CNN+LSTM model uses only PPG signal as the input signal. In a different study [61], both PPG and ECG signals were used as the input signal to CNN+LSTM for BP prediction. The study suggested that combining CNN with LSTM contribute to strong predictive abilities for sequential waveform data like BP. Notably, this study conducted separate predictions for SBP and DBP, achieving MAE and SD values of 4.41 mmHg and 6.11 mmHg for SBP, while 2.91 mmHg and 4.23 mmHg for DBP.

In [62], the study employed a CNN+LSTM architecture to capture morphological and temporal features from the signal difference between PPG and ECG for the simultaneous prediction of SBP and DBP. The study achieved ME±SD and MAE values of -0.02±1.6 mmHg and 1.2 mmHg for SBP, while 0.2±1.3 mmHg and 1.0 mmHg for DBP, indicating high accuracy. However, the study pointed out that the accuracy of LSTM models may be influenced by the duration of data measurement and generalizability of the model was not verified. Additionally, this study only used 48 patients in their research. Consequently, the optimal results reported may be influenced by the small sample size, as acknowledged by the authors.

In [32], the research examined different deep learning frameworks, incorporating Residual Network (ResNet) and WaveNet as CNNs, along with LSTM as an RNN for BP prediction. The combination of ResNet with LSTM produced the best outcomes, with a MAE and RMSE of 4.118 mmHg and 5.682 mmHg for SBP, while 2.228 mmHg and 2.986 mmHg for DBP. However, it is also the most computationally expensive. Additionally, the memory is still limited in capturing very long patterns observed in PPG signal. Furthermore, in the Leave-One-Out cross-validation, which excludes one subject from the training set and uses it as the test set, the MAE and RMSE increased to 16.128 mmHg and 17.875 mmHg for SBP, and to 6.743 mmHg and 7.902 mmHg for DBP.

2.4.2 Traditional machine learning model for BP prediction

2.4.2.1 Support Vector Regression (SVR)

One of the traditional machine learning algorithms used by many researchers for predicting BP is Support Vector Regression (SVR). SVR is a type of Support Vector Machine (SVM) used for regression tasks, works by finding a line hyperplane that best fits the data points in a higher dimension [42]. This algorithm maps input features to a higher-dimensional space through a kernel function [14].

In [42], the study used Linear Regression (LR), Artificial Neural Networks (ANN), and SVR based on only the PPG signal for BP prediction, and it suggested that SVR outperforms other machine learning algorithms with MAE±SD of 13.57±3.23 mmHg and 8.30±1.88 mmHg for SBP and DBP, respectively. Another study in [13] included some features from both PPG with ECG signals and suggested that SVR still outperforms Regularized Linear Regression (RLR) and ANN with MAE and SD of 12.38 mmHg and 16.17 mmHg for SBP, while 6.34 mmHg and 8.45 mmHg for DBP.

Besides that, some studies have employed SVR with real-world, self-collected data instead of using an online dataset. In [14], SVR showed a better performance than Random Forest, Adaboost, and ANN when real-world, self-

collected PPG and ECG signal is used, with MAE and RMSE of 6.97 mmHg and 8.15 mmHg for SBP.

After that, in [39], SVR and multivariate linear regression (MLR) were used to predict BP with GA as a feature selection method. Three experiments were conducted, including a static experiment, a dynamic experiment, and a follow-up experiment at 1 day, 3 days, and 6 months. SVR outperformed MLR, with ME±SD of −0.001±3.102 mmHg and −0.004±2.199 mmHg for SBP and DBP, respectively. The prediction accuracy remained relatively stable from one day to six months after the experiment's initiation. However, the study divided the 10-minute record of each subject is divided into 2 subsets, on for training and the other for testing, Hence, there are data leakage.

Moreover, in [63], the study proposed a hybrid model, where the SVR model was used as the last output layer of CNN model for BP prediction. The use of CNN removed the need of engineered feature extraction, and the result showed a good performance with MAE±SD and RMSE of 1.23±2.45 mmHg and 1.89 mmHg for SBP, while 3.08±5.67 mmHg and 3.91mmHg for DBP. In their study, the database was divided into training and test sets based on the number of subjects, ensuring there is no data leakage.

Furthermore, in [37], GA were implemented to optimize the hyperparameters of the SVR model, resulting in a reduction of the MSE from 337.37 mmHg to 38.33 mmHg for SBP and from 40.83 mmHg to 5.73 mmHg for DBP. This underscores the significance of fine-tuning SVR hyperparameters to improve the accuracy of BP predictions. The final model, when combined with MIV, achieved a MAE±SD of 3.27±5.52 mmHg and 1.16±1.97 mmHg for SBP and DBP, respectively. However, the authors only mentioned that 772 sets of waveform data were acquired without specifying the number of subjects involved, and they randomly selected the data to form the training and test sets. Hence, there is a possibility of data leakage.

2.4.2.2 Random Forest

Random Forest is another traditional machine learning algorithm used by many researchers in literature. Random forest utilizes ensemble learning techniques, combining multiple decision trees to make predictions through random sampling with replacement [38]. It is called 'Random' Forest as each tree in it is grown using randomized subset of predictors [64]. In regression tasks, the final value is obtained based on the average value predicted by each tree.

In [65], the study used Random Forest for BP prediction based on features from PPG and ECG signals. The result displayed in RMSE which were 13.01 mmHg and 12.89 mmHg for SBP and DBP, respectively. The study confirmed the potential of Random Forest in non-invasive BP estimation. After that, in [38], the Random Forest was used with GCMI to predict the BP by using online dataset to train the model and test the model using self-collected dataset. After calibration, the result showed MAE±SD of 5.21±5.98 mmHg and 4.15±5.66 mmHg for SBP and DBP, respectively, which demonstrate a good prediction performance. The calibration includes selecting a quarter of the testing set to fine-tune the model. Additionally, this study only focused on younger people.

Besides that, Random Forest outperformed other traditional machine learning including SVR in some studies. In [36], the study extracted parameter and whole based features from PPG and ECG and predicted BP using different traditional machine learning models. From the result, Random Forest achieved lower SD value compared to other traditional machine learning such as RLR, decision tree regression, SVR, and Adaboost, with MAE and SD of 11.80 mmHg and 9.87 mmHg for SBP, while 5.83 mmHg and 5.71 mmHg for DBP. Additionally, Random Forest also showed better performance in [31] when compared to other traditional machine learning models including LR, ridge regression, SVR, and Adaboost. After implementation of GA for feature optimization, the Random Forest model achieved MAE and RMSE of 9.54 mmHg and 13.83 mmHg for SBP, while 5.48 mmHg and 6.80 mmHg for DBP.

Furthermore, in [66], the study used the tree-based pipeline optimization tool (TPOT), an automated machine learning tool that uses genetic programming to find the optimized machine learning pipelines for BP estimation. According to the study, Random Forest was chosen as the best machine learning algorithm for SBP estimation with an MAE and MSE of 6.52 mmHg and 7.48 mmHg, suggesting that Random Forest would perform better than others. However, in this study, only 1000 records from the UCI dataset were included, posing a significant risk of data leakage.

Moreover, in [67], the study focused on non-invasive BP prediction based on Random Forest. In the study, SVR was used for comparison with the performance of Random Forest. GA was used to optimize the hyperparameters of SVR, and grid search was employed for Random Forest. The study suggested that Random Forest is significantly better than SVR under the same conditions, with an MAE of 4.45 mmHg and 3.95 mmHg for SBP and DBP, respectively.

In [47], MLR, SVR, and Random Forest were used to predict BP by combining morphological features of the ECG signal with PAT. From the results obtained, Random Forest achieved an MAE±SD of 6.12±9.52 mmHg for SBP and 4.02±6.58 mmHg for DBP. However, this study included approximately 3500 records from 227 patients, with data randomly selected to form the training and testing sets. This introduces a potential risk of data leakage, which could impact the robustness of the findings.

2.4.3 Data leakage in BP research

BP datasets usually contain multiple records for the same subject. If special care is not taken to prevent data leakage when splitting the dataset into training and testing sets, it can lead to misleading and overly optimistic results, as records from the same subject may appear in both sets. This issue is commonly overlooked in BP research, particularly for those using the UCI dataset. Recently, some researchers have addressed this problem in BP research and demonstrated the significant impact of data leakage on the results.

The University of California, Irvine (UCI) dataset, also known as the Cuff-Less Blood Pressure Estimation dataset [13, 36], consists of simultaneous ABP, PPG, and ECG signals, originally obtained from the Multi-parameter Intelligent Monitoring in Intensive Care (MIMIC) II database. Some preprocessing techniques have been applied to the original data, as detailed in [13]. This dataset includes information for 12000 records across various blood pressure categories. Due to its large size, this dataset has been widely used by researchers. However, it has the limitation of lacking a patient index number related to each record segment. Consequently, there is a risk that records from the same patient could appear in both the training and test sets, leading to overly optimistic results.

In [17], researchers conducted a benchmark study for machine learning-based non-invasive BP estimation using PPG signals. This study encompassed both traditional machine learning methods (LightGBM, SVR, Multi-Layer Perceptron, AdaBoost, and Random Forest) and deep learning approaches (ResNet, SpectroResNet, MLP-BP, U-Net, PPGIABP, and V-Net) to predict BP using four different publicly available datasets. The findings revealed that in scenarios where data leakage occurred, the results surpassed those of scenarios without leakage for both traditional machine learning and deep learning, irrespective of the evaluation metric employed.

Specifically, the mean absolute scaled error (MASE) of SBP and DBP dropped from approximately 98–92% to below 60%, while the SD metric (AAMI standard) also exhibited a significant decrease. Traditional machine learning methods outperformed deep learning for smaller datasets. For instance, one of their findings indicated that the SVR achieved MAE and ME±SD of 15.60 mmHg and -0.00±19.68 mmHg for SBP, and 7.50 mmHg and -1.45±9.81 mmHg for DBP.

Furthermore, in [16], the researchers addressed this issue by using the term "intra-subject" to denote the data leakage scenario and "inter-subject" to represent scenarios without data leakage. This study incorporated both feature-based machine learning (XGBoost, LightGBM, and CatBoost) and deep learning approaches (Residual U-Net, ResNet-18, and ResNet-LSTM) to predict BP from PPG signals using two distinct datasets.

The results obtained indicated that for the intra-subject scenario, almost all models met the AAMI standard, except for ResNet-LSTM. This suggests that both feature-based machine learning and deep learning approaches are equally capable of addressing the problem. However, in the inter-subject scenario, none of the methods met the AAMI standard, as the standard deviations of SBP and DBP averages of the dataset were 21.27 mmHg and 10.07 mmHg, respectively. Additionally, all methods failed to achieve the BHS standard, falling into Grade D.

Furthermore, in [15], the study also addressed the issue of data leakage in their results. XGBoost and CatBoost were utilized to predict BP based on features extracted from PPG signal. The findings indicated that CatBoost demonstrated superior performance, achieving a MAE and ME±SD of 5.368 mmHg and 0.050±7.837 mmHg for SBP, and 2.521 mmHg and 0.022±3.767 mmHg for DBP. However, these results were obtained when considering data leakage. When the researchers minimized the data leakage, the MAE and ME±SD increased to 18.209 mmHg and -1.230±22.368 mmHg for SBP, and 7.524 mmHg and -0.257±9.784 mmHg for DBP.

2.4.4 Summary of BP prediction model

In summary, the deep learning approach, particularly Long Short-Term Memory (LSTM) networks, outperforms traditional machine learning approaches in handling temporal dependencies, such as those exhibited by BP. However, this technique has the disadvantage of low interpretability, which is a significant concern in healthcare settings where model comprehension and transparency are essential. Additionally, deep learning requires a large amount of data to achieve good results, especially for predicting BP from PPG and ECG signals, as these signals vary among individuals. Consequently, substantial data is needed for tuning the model, which may not be feasible in real-world scenarios. Furthermore, recent research has suggested that many previous studies using deep learning approaches have overlooked data leakage issues, leading to overly optimistic results.

For traditional machine learning approaches, despite requiring a manual feature engineering process, offer benefits in terms of interpretability. This approach is less complex compared to deep learning, allowing for the identification of crucial aspects of PPG and ECG signals that contribute the most to predicting BP. Additionally, traditional machine learning requires less data and is less computationally expensive, making it more suitable for real-world scenarios. Random Forest and SVR are non-linear machine learning models that have demonstrated good accuracy in BP prediction in the literature. Hence, these two models were chosen to be compared in this project.

2.5 Summary

There are various features can be extracted from the PPG and ECG signals. Most of the researchers focused on the features from PPG. Even though some include the features from ECG in their prediction model, but involvement of ECG features is limited compared to PPG. Since ECG is proven to provide diagnostic information about the BP status, hence, there is a need to further explore into features from ECG which may be relevant to BP status in order to come out with a better set of features used in BP prediction. Besides, this project also aims to identify the best set of features among different combinations of PPG, ECG and their demographic features.

BP prediction models often involve deep learning or machine learning techniques. Deep learning extracted features automatically while machine learning involves the physiological feature extraction to improve accuracy. Deep learning is well-known with its powerful automated modelling but with large data size. Machine learning has the substantial practical benefit over deep learning where the features used are known, which benefit to medical teams in clinical study on BP and cardiovascular diseases, and it needs much smaller size of data. Hence, machine learning is chosen over deep learning in this project. The pro and cons of commonly used machine learning methods are discussed in Section 2.4.4.

In order to identify the best set of features for BP prediction model to be fed into machine learning, feature selection will be applied. There are a number of feature selection methods used by researchers to obtain the best combination of features. The most common used are MIV, MI, GA and SHAP. The MIV and MI have potential bias towards high-cardinality features and scalability issues, while GA has the risk of premature convergence. Even though SHAP has potential approximation errors, it has several advantages over the other three methods, particularly in terms of interpretability, consistency, and the ability to handle complex models. Hence, SHAP is chosen as the feature selection method in this project.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter provides a brief overview of the proposed method's implementation, utilizing SHAP with traditional machine learning model for optimized feature combination selection, and traditional machine learning model as the final BP prediction model. It also guides through the step-by-step process of predicting BP based on PPG and ECG signals, covering the preprocessing method, feature extraction approach, and model evaluation. Several experiments are also proposed. The BP prediction model includes both SBP model and DBP model.

3.2 Project overview

This project overview outlines a step-by-step process for completing the project. It begins by defining the problem statement and setting project objectives and scope. Next, a literature review is conducted to identify potential solutions, and a method for solving the problem is proposed. The project is then progressed to analyze the results and draw conclusions. Figure 3.1 shows the flow chart for the final year project development.

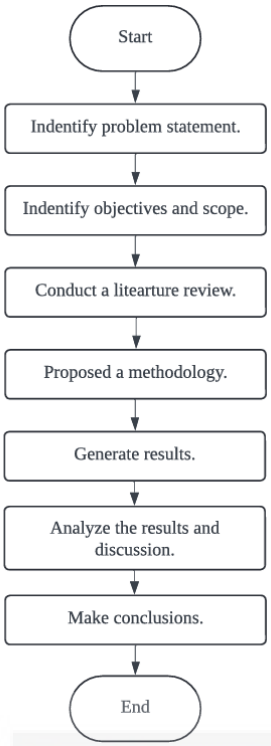


Figure 3.1: Flow chart for the final year project development.

3.3 Proposed methodology

In order to predict BP based on PPG and ECG signals, a series of steps must be taken. First, the relevant signals, including PPG, ECG, and ABP, are imported. Next, preprocessing is conducted on the PPG and ECG, this process involves excluding the corrupted signals to ensure the quality of the signal used. Once preprocessing is completed, the reference SBP and DBP values are extracted from the ABP signal, and relevant features are extracted from the PPG and ECG signals. Demographic features, which include subject characteristics, are also included in this study.

After extracting all the features, two experiments were conducted. The first experiment includes selection of the best machine learning model to form the model for BP prediction. The second experiment include testing different features combination towards prediction of BP to investigate the impact. The feature selection technique was also proposed to select the optimized feature combination. The final BP prediction models are developed for both SBP and DBP separately, based on their selected feature combination. The model's performance is then evaluated. The

flow chart of the process for developing the BP prediction model is shown in Figure 3.2.

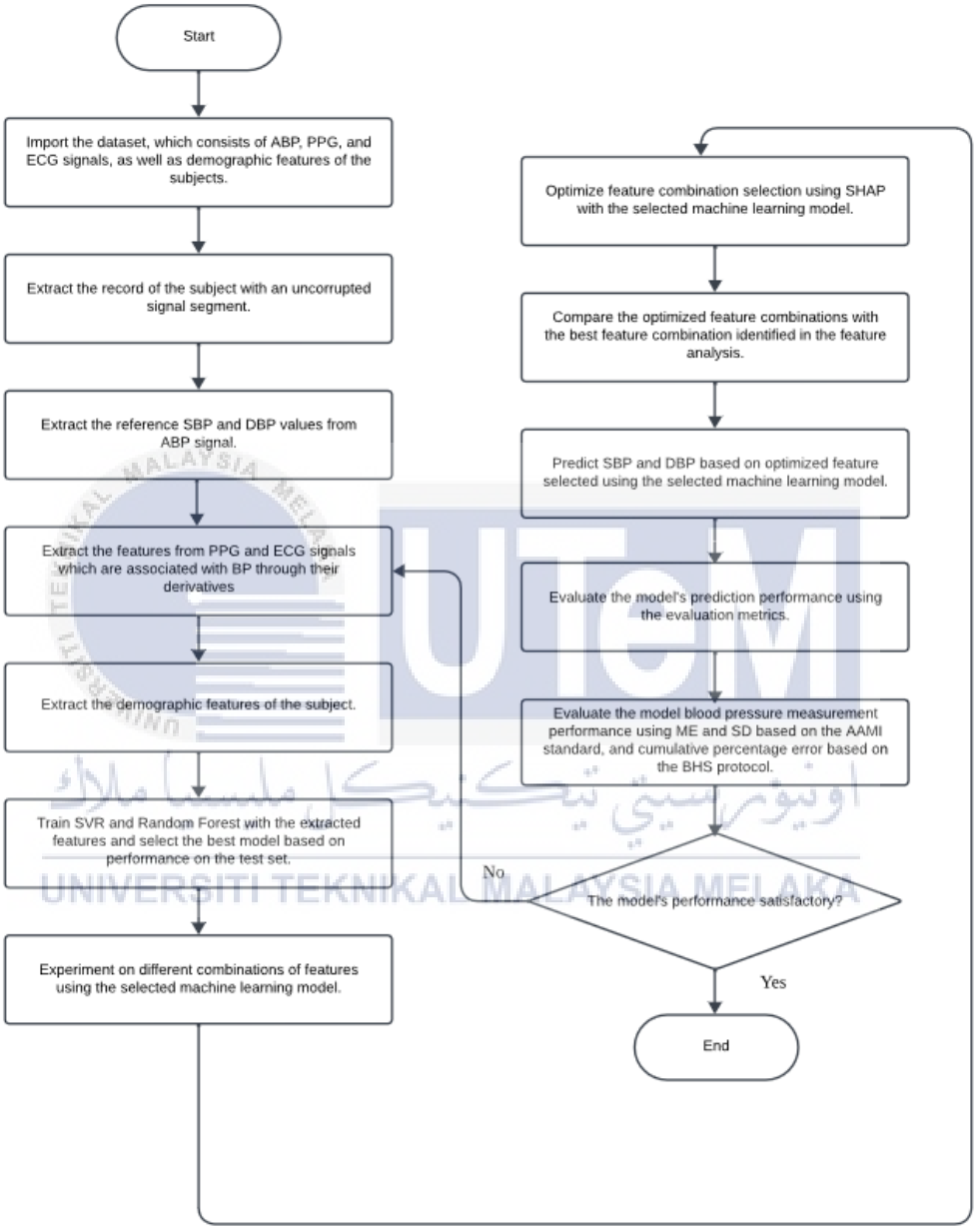


Figure 3.2: The flow chart of the process for developing the blood pressure prediction model.

3.4 Data Preprocessing

3.4.1 Dataset

Recently, a large cleaned dataset called “PulseDB” has been published [68]. This dataset consists of 5245454 high-quality 10-second segments of ECG, PPG, and ABP signal from 5361 subjects retrieved from the MIMIC-III waveform database matched subset and the VitalDB database. One of the advantages of this dataset is that it includes subjects’ identification and demographic information, allowing for the evaluation of the generalizability of models to unseen data and avoiding data leakage that could lead to overly optimistic results. The authors have shared a subset of PulseDB derived from the VitalDB dataset on Kaggle. Hence, this dataset has been used in this project.

From the dataset on Kaggle, the ‘VitalDB_Train_Subset’ is selected as it contains more subjects. Taking advantage of feature-based machine learning, which requires less data compared to deep learning, 10 signal segments and the subjects’ age, gender, weight, height, and body mass index (BMI) from 500 subjects are selected. Each signal segment consists of simultaneous, non-overlapping 10-second segments of ABP, PPG, and ECG signals.

3.4.2 Preprocessing of PPG and ECG signals

The dataset on Kaggle is the pre-processed version, and the process has been clearly explained in [68]. The filtering process involved resampling the ABP, PPG, and ECG signals from 500 Hz to 125 Hz, filtering the PPG signal with an 8^{th} order Chebyshev-II filter with cutoff frequencies of 0.5-8 Hz, and filtering the ECG signal with an 8^{th} order Butterworth filter with cutoff frequencies of 0.5-40 Hz. Additionally, the amplitudes of the PPG and ECG signals have been normalized to a range between 0 and 1. Figure 3.3 shows an example 10 second segment of the ABP, PPG, and ECG signals.

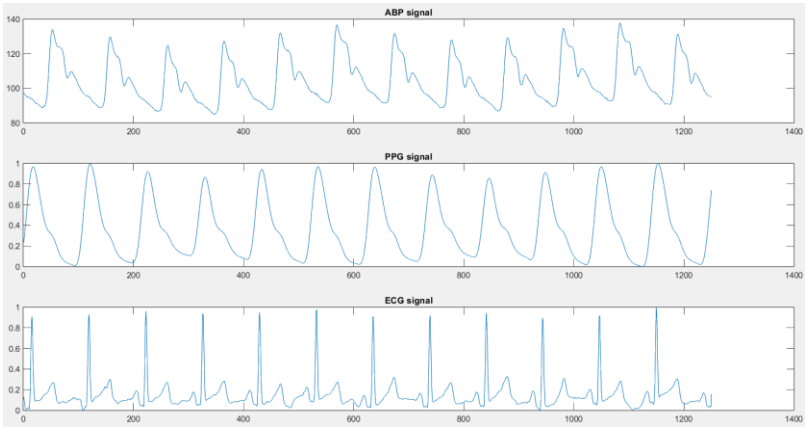


Figure 3.3: An example 10 second segment of the ABP, PPG, and ECG signals.

Despite some irrelevant signal segments have been discarded in [68], upon observation, it was noted that some signals remained corrupted. Figure 3.4 shows an example of a corrupted PPG signal, while Figure 3.5 shows an example of a corrupted ECG signal.

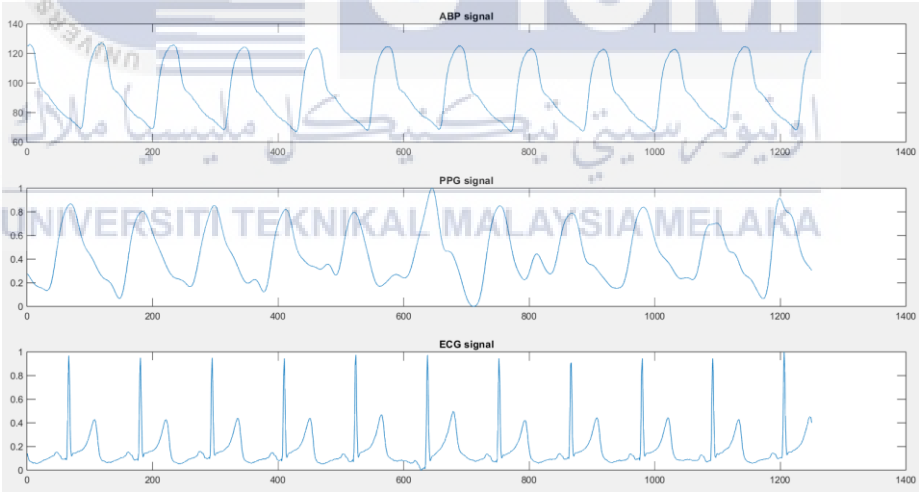


Figure 3.4: Corrupted PPG signal.

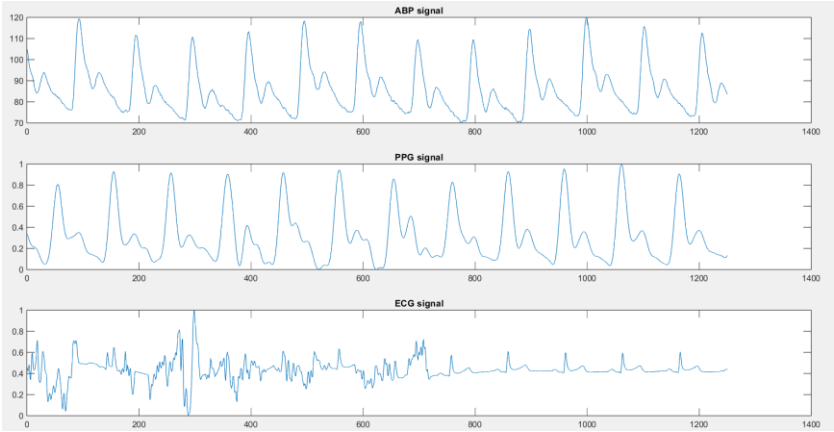


Figure 3.5: Corrupted ECG signal.

In order to address this issue, signal segments that were unable to extract features were excluded. Additionally, the SBP value is limited between 80 to 180 mmHg, while the DBP value is limited between 50 to 130 mmHg, as suggested in [31, 36]. The DBP is allowed to go as low as 50 mmHg to accommodate some hypotensive subjects, compared to [31, 36], which limits the DBP to above 60 mmHg.

In addition, certain conditions are set to ensure that the signals involved follow the correct sequence, which will be mentioned in next Section.

3.4.3 Segmentation of ABP, PPG and ECG Signals

All the *R* peak of the ECG signal is obtained using the 'findpeaks' function in MATLAB. The first and last *R* peak of the ECG signal is set as the beginning and the end of each of the segment as shown in Figure 3.6. This range will be used for feature extraction.

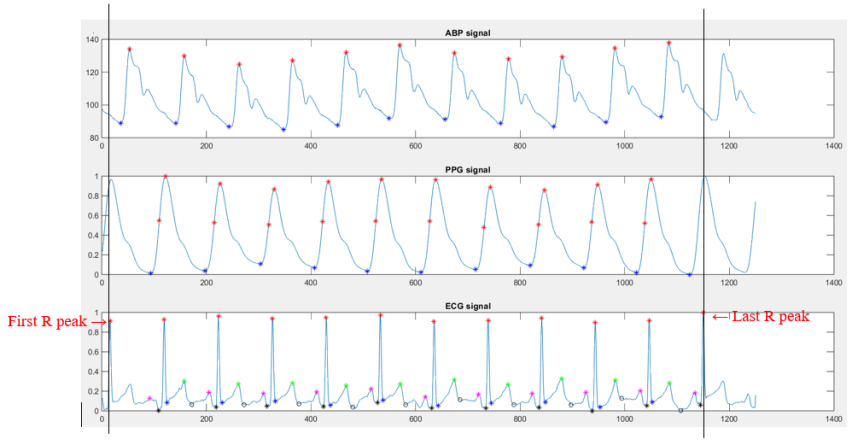


Figure 3.6: The range consideration for BP prediction.

Some predefined conditions are established to ensure that the extracted signals adhere to the correct order. Firstly, since one complete PPG cycle comprises one systolic peak and two onsets, if the total systolic peaks do not equal the total onsets plus one, the signal segments are excluded. Secondly, the number of systolic peaks in the PPG signal must be fewer than the number of R peaks in the ECG signal by two. Thirdly, any segment where the heart rate is more than 30 bpm higher than the average heart rate for that segment is excluded. The detection of the points will be discussed in the next section.

3.5 Feature extraction

3.5.1 Extraction of SBP and DBP values

The SBP and DBP values serve as the target values for BP prediction in this project. Initially, the *R* peaks of the ECG signal are detected using the 'findpeaks' function in Matlab. Subsequently, the SBP value is derived from the ABP signal by referencing the maximum value between the first *R* peak and the last *R* peak of the ECG signal, while DBP value is derived by referencing the minimum value between the first *R* peak and the last ABP systolic peak as illustrated in Figure 3.7. MATLAB's 'findpeaks' function is utilized to detect these values. Once all SBP and DBP values are detected, their averages are computed to form the final SBP and DBP values for that signal segment.

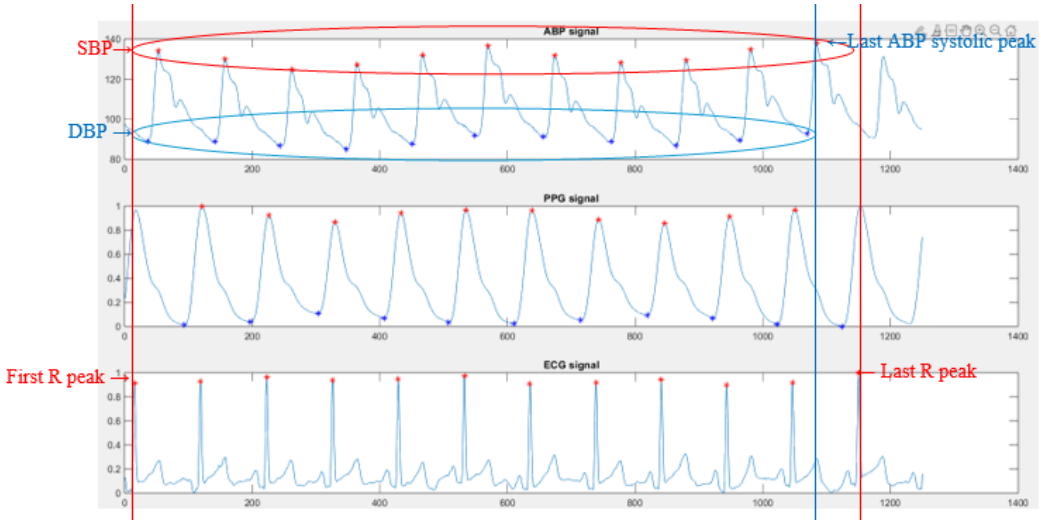


Figure 3.7: Reference points identified in the segmented signal.

3.5.2 Fiducial point detection for PPG signal

Some features require the identification of the signal's fiducial points before extraction becomes possible. For PPG and its derivatives signals, a total of 15 fiducial points need to be detected for the features used in this project.

A single PPG cycle starts and ends with onset points. Initially, the onset points of the PPG signal are detected using the 'findpeaks' function in MATLAB. This is done by finding the minimum point between the first ABP onset until the last *R* peak of the ECG signal, as shown in Figure 3.8. Next, the systolic peak of the PPG is detected by referring to the maximum point between the first systolic peak of ABP until the last onsets of PPG using the 'findpeaks' function, as shown in Figure 3.8.

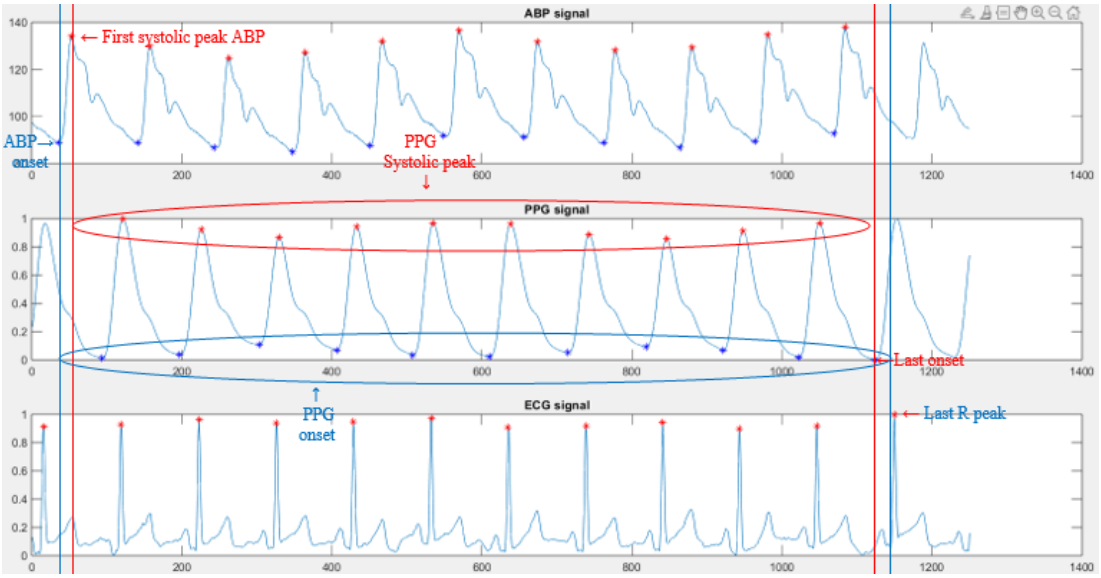


Figure 3.8: Detection of systolic peak and onsets of PPG signal.

After that, the first and second derivatives of the PPG signal, known as the VPG and APG signals respectively, are obtained using the 'gradient' function in Matlab. Figure 3.9 shows an example of the 10 cycles of the PPG signal with its derivative signals. The peak of the VPG signal in each PPG cycle will be used to determine the maximum slope of the PPG cycle.

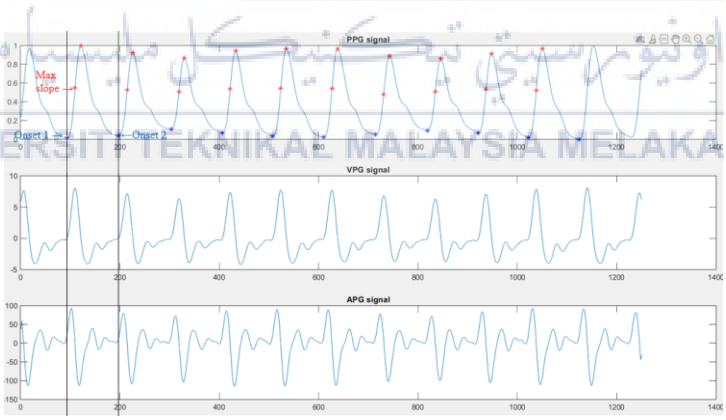


Figure 3.9: 10 cycles of the PPG signal with its derivative signals.

Once the derivative signals have been obtained, each cycle of the PPG signal will be used to extract features. The features of the VPG and APG signals will also be extracted based on the start and end of the PPG cycles. This process will repeat, starting from the PPG cycle after the first *R* peak until the PPG cycle before the last *R* peak. Figure 3.10 shows an example of the single PPG cycle with its derivatives.

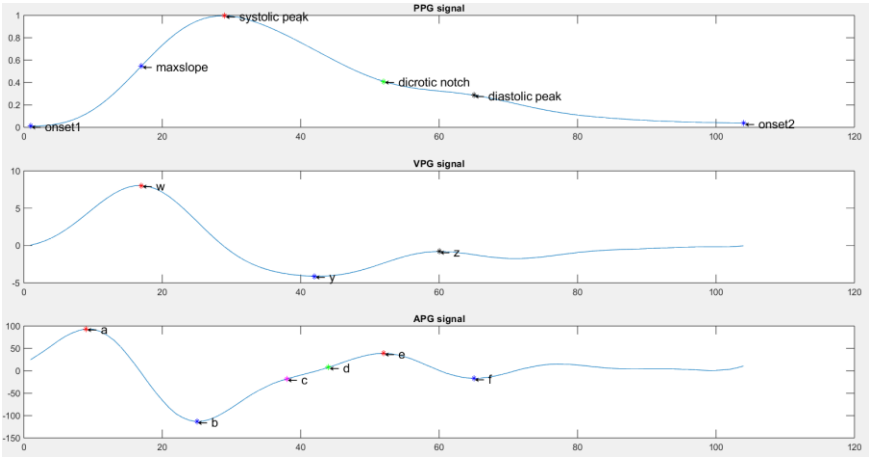


Figure 3.10: The first PPG cycle and its derivatives.

A normal PPG signal requires detecting a total of five points: two onset points, one systolic peak, one dicrotic notch, and one diastolic peak. Additionally, the maximum slope of the PPG cycle is essential for feature extraction in this project. The two onset points, one systolic peak, and the maximum slope of the PPG cycle can be detected by following the aforementioned procedure. However, the identification of the dicrotic notch and diastolic peak, which may be less pronounced, relies on the locations of points e and f in the APG, representing the second derivative of the PPG.

For the VPG, the first derivative of the PPG. There are three points in total that need to be detected which are w, y, and z. These points can be detected using the 'findpeak' function. The w is the referring to the maximum peak of VPG between the first onset and systolic peak of the PPG, y is the minimum point of the VPG, and z is the sub-peak of VPG after the y.

For the APG, the second derivative of the PPG. There are five points in total that need to be detected which are a, b, c, d, e, and f. For point a, b, e, and f, the 'findpeaks' function is used. For a, it is an early systolic positive peak, detected by locating the maximum peak, while for b, it is an early systolic negative peak,

detected by locating the trough of the APG after a. For e, it is an early diastolic positive wave, detected by locating the maximum peak after b, while for f, it is a diastolic negative wave, detected by locating the trough after e. The point e and f correspond to dicrotic notch and diastolic peak of PPG signal, respectively.

For points c and d, which are less prominent in most of the subjects in the dataset. Hence, the third derivative of PPG, called jerk photoplethysmography (JPG) is used, to detect these points [69]. There are three different cases for detecting points c and d. Case 1 for c and d when they are not prominent, case 2 for c and d when they are undetectable, and case 3 points c and d when they are prominent. All these cases are illustrated in Figure 3.11.

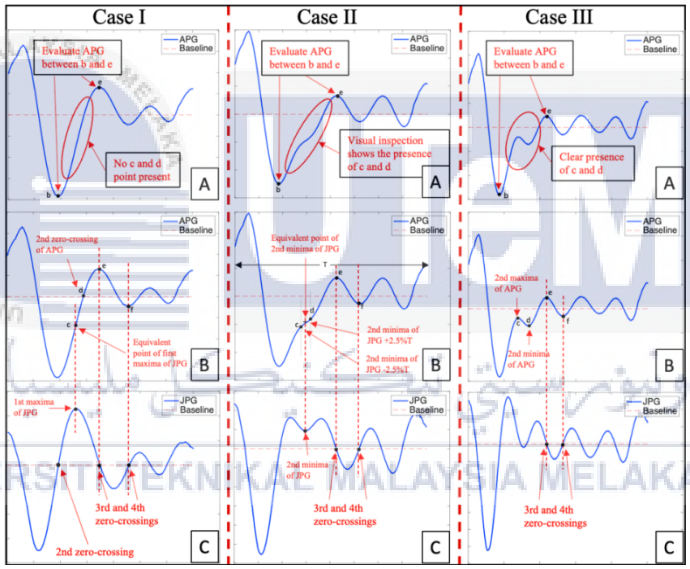


Figure 3.11: APG fiducial point detection for three different cases [69].

For case 1, point c is detected by referring to the maximum peak of JPG, and point d is detected by referring to the 2nd zero-crossing of APG. For case 2, the point c is detected by adding the location of 2nd minima of JPG with 2.5% of the total length of APG signal, while point d is detected by subtracting the location of the 2nd minima of JPG with 2.5% of the total length of APG signal. For case 3, the point c and d are detected by referring to the maxima and minima of APG between b and e, respectively.

3.5.3 Wave detection for ECG signal

There are a total of 5 wave that need to be detected on the ECG signal, namely P, Q, R, S, and T, as shown in Figure 3.12. Additionally, the end of the T wave also needs to be identified to extract some features.

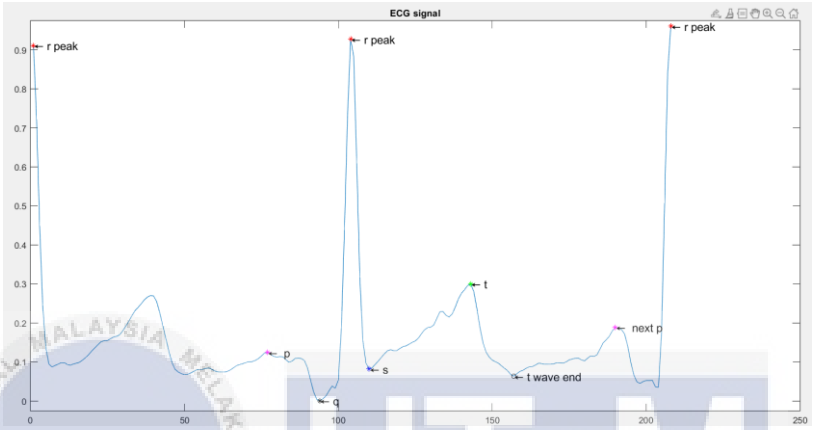


Figure 3.12: Wave for ECG signal.

As mentioned earlier, the R peak of the ECG signal is detected by referring to the maximum of the ECG signal. After detecting the R peak, Q is detected by referring to the last minimum point between the previous R peak and the current R peak, while S is detected based on the first minimum point between the current R peak and the next R peak. Next, P is detected by referring to the last peak between the previous R peak and the current Q. Following that, T is detected by referring to the first peak between the current S and the next P. Finally, the end of the T wave is detected by referring to the minimum between the current T and the next P.

All these points are detected using the 'findpeaks' function in MATLAB, starting from the first R peak until the last R peak of the ECG signal. Figure 3.13 displays a 10-second segment of the ECG signal with all the detected points.

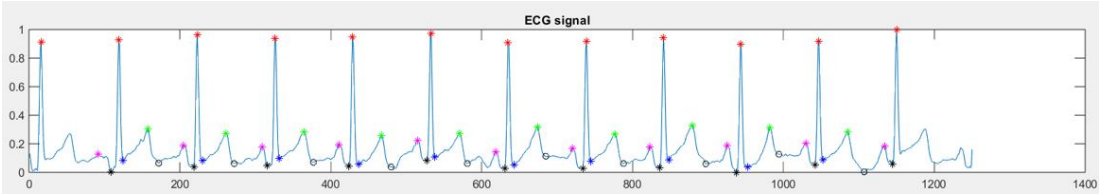


Figure 3.13: 10-second segment of the ECG signal.

3.5.4 Morphological and dynamic features of PPG and ECG signals

Based on previous studies, 78 features were selected and extracted from PPG and ECG signals. These features include morphological and dynamic features. Demographic features, consisting of subject information, were also extracted. All features were extracted using MATLAB R2020b. Features extracted individually from PPG and ECG are displayed in Tables 3.1 and 3.2, respectively, while features extracted from both PPG and ECG signals are displayed in Table 3.3, and demographic features are displayed in Table 3.4. For features from PPG and ECG signals, the features are extracted from each of the cycles within the first and last R peak of ECG. These features from all cycles are then averaged to obtain a single representative set of features

PPG features

Table 3.1: PPG features.

No	Feature name	Feature extraction method
1	PPG_K value	PPG_K value is computed using the following equation $$ppgKvalue = \frac{p_m - p_d}{p_s - p_d}$$ where p_s is the systolic peak in PPG, p_d is onset value in systolic phase, and $p_m = \frac{1}{T}\int PPG(t)dt$, respectively.
2	PIR	PIR is computed by taking the ratio of maximum value to the minimum value in the PPG signal.
3	$A_{b-c-d-e/a}$	$A_{b-c-d-e/a}$ is computed using the following formula, $$\frac{A_b - A_c - A_d - A_e}{A_a}$$ where A_a, A_b, A_c, A_d, and A_e represent the amplitude of

		points a, b, c, d, and e in the APG, respectively.
4	T_{Oa}	T_{Oa} is computed by measuring the time difference between the onset in PPG systolic phase and the point a in the APG.
5	C_{slope}	C_{slope} is computed by determining the slope from the onset in systolic phase to the systolic peak in the PPG. Slope is calculated by dividing the change in amplitude to the change in time.
6	T_{OS}	T_{OS} is computed by measuring the time difference between the onset in systolic phase and systolic peak in PPG.
7	LASI	LASI is computed by measuring the time difference between the systolic peak and diastolic peak in PPG.
8	AI	AI is computed by calculating the ratio of diastolic peak to systolic peak in PPG.
9	IPA	IPA is computed by calculating the ratio of the diastolic area to the systolic area. MATLAB's 'trapz' function is utilized for area calculation. The systolic area is determined by integrating the PPG from the onset in the systolic phase to the dicrotic notch, while the diastolic area is measured by integrating the PPG from the dicrotic notch to the onset in the diastolic phase.
10	$S1$	$S1$ is computed using the 'trapz' function in MATLAB, integrating the PPG from the onset in systolic phase to the location of the maximum slope (corresponding to the location of w in the VPG).
11	$S2$	$S2$ is computed using the 'trapz' function in MATLAB, integrating the PPG from the maximum slope to the systolic peak.
12	$S3$	$S3$ is computed using the 'trapz' function in MATLAB, integrating the PPG from the systolic peak to the diastolic peak.
13	$S4$	$S4$ is computed using the 'trapz' function in MATLAB, integrating the PPG from the diastolic peak to the onset in diastolic phase.
14	Kurtosis	Kurtosis is computed using 'kurtosis' function in MATLAB.
15	$Angle_{ND}$	$Angle_{ND}$ is computed by determining the slope from the

		location of the dicrotic notch to the location of the diastolic peak in the PPG. Slope is calculated by dividing the change in amplitude to the change in time.
16	$Angle_{SD}$	$Angle_{SD}$ is computed by determining the slope from the location of the systolic peak to the location of the diastolic peak in the PPG. Slope is calculated by dividing the change in amplitude to the change in time.
17	T_{ND}	T_{ND} is computed by measuring the time difference between the dicrotic notch and the diastolic peak in PPG.
18	T_{wc}	T_{wc} is computed by measuring the time difference between point w in VPG and point c in the APG.
19	$Angle_{zy}$	$Angle_{zy}$ is computed by determining the slope from point z to the point y in the VPG. Slope is calculated by dividing the change in amplitude to the change in time.
20	$Angle_{ed}$	$Angle_{ed}$ is computed by determining the slope from point e to the point d in the APG. Slope is calculated by dividing the change in amplitude to the change in time.
21	$Width_{10}$	$Width_{10}$ or DW10+SW10 is computed by measuring the time difference between two points in the PPG signal where the amplitude is at 10% of the peak, encompassing both the systolic and diastolic part.
22	$Width_{25}$	$Width_{25}$ or DW25+SW25 is computed by measuring the time difference between two points in the PPG signal where the amplitude is at 25% of the peak, encompassing both the systolic and diastolic part.
23	$Width_{33}$	$Width_{33}$ or DW33+SW33 is computed by measuring the time difference between two points in the PPG signal where the amplitude is at 33% of the peak, encompassing both the systolic and diastolic part.
24	$Width_{50}$	$Width_{50}$ or DW50+SW50 is computed by measuring the time difference between two points in the PPG signal where the amplitude is at 50% of the peak, encompassing both the

		systolic and diastolic part.
25	$Width_{66}$	$Width_{66}$ or DW66+SW66 is computed by measuring the time difference between two points in the PPG signal where the amplitude is at 66% of the peak, encompassing both the systolic and diastolic part.
26	$Width_{75}$	$Width_{75}$ or DW75+SW75 is computed by measuring the time difference between two points in the PPG signal where the amplitude is at 75% of the peak, encompassing both the systolic and diastolic part.
27	DW10	DW10 is computed by measuring the time interval between the systolic peak and the point in the diastolic phase where the PPG amplitude falls to 10% of the peak.
28	DW25	DW25 is computed by measuring the time interval between the systolic peak and the point in the diastolic phase where the PPG amplitude falls to 25% of the peak.
29	DW33	DW33 is computed by measuring the time interval between the systolic peak and the point in the diastolic phase where the PPG amplitude falls to 33% of the peak.
30	DW50	DW50 is computed by measuring the time interval between the systolic peak and the point in the diastolic phase where the PPG amplitude falls to 50% of the peak.
31	DW66	DW66 is computed by measuring the time interval between the systolic peak and the point in the diastolic phase where the PPG amplitude falls to 66% of the peak.
32	DW75	DW75 is computed by measuring the time interval between the systolic peak and the point in the diastolic phase where the PPG amplitude falls to 75% of the peak.
33	DW10/SW10	DW10/SW10 is computed by calculating the ratio between DW10 to the SW10, SW10 is similar with DW10 but referring to the point in systolic phase to the systolic peak.
34	DW25/SW25	DW25/SW25 is computed by calculating the ratio between DW25 to the SW25, SW25 is similar with DW25 but referring to the point in systolic phase to the systolic peak.

35	DW33/SW33	DW33/SW33 is computed by calculating the ratio between DW33 to the SW33, SW33 is similar with DW33 but referring to the point in systolic phase to the systolic peak.
36	DW50/SW50	DW50/SW50 is computed by calculating the ratio between DW50 to the SW50, SW50 is similar with DW50 but referring to the point in systolic phase to the systolic peak.
37	DW66/SW66	DW66/SW66 is computed by calculating the ratio between DW10 to the SW66, SW66 is similar with DW66 but referring to the point in systolic phase to the systolic peak.
38	DW75/SW75	DW75/SW75 is computed by calculating the ratio between DW75 to the SW75, SW75 is similar with DW75 but referring to the point in systolic phase to the systolic peak.
39	t_{pp}	t_{pp} or Cardiac period is computed by measuring the time difference between currect systolic peak to the next systolic peak in PPG signal.
40	DT	DT is computed by measuring the time difference between systolic peak to the onset at the diastolic phase in PPG.
41	b/a	b/a is computed by calculating the ratio of the amplitude of point b to the amplitude of point a in APG.
42	c/a	c/a is computed by calculating the ratio of the amplitude of point c to the amplitude of point a in APG.
43	d/a	d/a is computed by calculating the ratio of the amplitude of point d to the amplitude of point a in APG.
44	e/a	e/a is computed by calculating the ratio of the amplitude of point e to the amplitude of point a in APG.
45	α_n	α_n is computed by using the following equation, $$\alpha_n = R\sqrt{\frac{\omega\rho}{b}}$$ where R is the valley amplitude of the PPG signal, ω is the frequency of heart rate, ρ is the density of blood which assume to be 1060 kg/m^3 , and b is the inverse magnitude between the point in the VPG signal corresponding to the ω in VPG and the location that aligns with the systolic peak in the PPG signal

Table 3.2: ECG features.

No	Feature name	Description
1	Heart rate	Heart rate is computed by using the following equation, $$\text{Heart rate} = \frac{60 \times f_s}{RR}$$ where f_s represents the sampling frequency, and the RR interval is obtained by measuring time difference between the first and second R peaks in the ECG signal.
2	Hjorth mobility	Hjorth mobility is computed using the following equation, $$Hjorth\ mobility = \sqrt{\frac{var(x')}{var(x)}}$$ where x represents the ECG segment, x' is the derivative of the ECG, and var denotes variance, respectively.
3	Hjorth complexity	Hjorth complexity is computed by calculating the ratio of the mobility of the derivative of the ECG to the mobility of the ECG. $$Hjorth\ complexity = \frac{Mobility\ (x')}{Mobility\ (x)}$$
4	QRS complex	QRS complex is computed by measuring the time difference between Q and S in ECG signal
5	QT	QT is computed by measuring the time difference between Q and T wave end in the ECG signal
6	QT_c	QT_c is computed by dividing the QT interval by the square root of the RR interval in the ECG signal. $$QT_c = \frac{QT}{\sqrt{RR\ interval}}$$ where the RR interval is the time difference between current R peak to the next R peak.
7	SDI	SDI is computed by calculating the ratio of the QT interval to the TQ interval in the ECG signal.

		$$SDI = \frac{QT}{TQ}$$ where the TQ is time difference between currect T wave end to the next Q.
8	SDIn	SDIn is computed by calculating the ratio of QT to the RR interval in the ECG signal $$SDIn = \frac{QT}{RR\ interval}$$ where the RR interval is the time difference between current R peak to the next R peak.
9	P	P is computed by taking the amplitude of P wave in ECG signal.
10	Q	Q is computed by taking the amplitude of Q wave in ECG signal.
11	R	R is computed by taking the amplitude of R wave in ECG signal.
12	S	S is computed by taking the amplitude of S wave in ECG signal.
13	T	T is computed by taking the amplitude of T wave in ECG signal.
14	PR	PR is computed by measuring the time difference between P and R in the ECG signal.
15	RT	RT is computed by measuring the time difference between R and T in the ECG signal.
16	PQ	PQ is computed by measuring the time difference between P and Q in the ECG signal.
17	ST	ST is computed by measuring the time difference between S and T in the ECG signal.
18	PT	PT is computed by measuring the time difference between P and T in the ECG signal.
19	RT ratio	RT ratio is computed by calculating the ratio of T peak to R peak in ECG signal.

20	*RP* diff	*RP* diff is computed by calculating the difference between *R* peak amplitude from *P* peak amplitude.

Features from both PPG and ECG

Table 3.3: Features from both PPG and ECG.

No	Feature name	Description
1	PAT peak	PAT peak is computed by measuring the time difference between the first *R* peak in the ECG and the systolic peak in the PPG.
2	PAT maxslope	PAT maxslope is computed by measuring the time difference between the first *R* peak in the ECG and the maximum slope in the PPG. The maximum slope is obtained by referring to the maximum peak of VPG.
3	PAT onset	PAT onset is computed by measuring the time difference between the first *R* peak in the ECG and the onset in systolic phase in the PPG.

Demographic feature

Table 3.4: Demographic features.

No	Feature name	Description
1	BMI	BMI of the subject
2	Age	Age of the subject
3	Weight	Weight of the subject
4	Height	Height of the subject
5	Gender	Gender of the subject
6	BMI/t_1	BMI/t_1 is computed by calculating the ratio of the BMI of the subject to the time difference between the onset in systolic phase and systolic peak in PPG.
7	Weight/t_{pi}	Weight/t_{pi} is computed by calculating the ratio of the weight of the subject to the time difference between onset in systolic phase and onset in diastolic phase of PPG.
8	Weight/t_{pp}	Weight/ t_{pp} is computed by calculating the ratio of the

		weight of the subject to the peak-to-peak interval of the PPG signal
9	Weight/t_1	Weight/t_1 is computed by calculating the ratio of the weight of the subject to the time difference between the onset in systolic phase and systolic peak in PPG.
10	BMI/t_{pp}	BMI/t_{pp} is computed by calculating the ratio of the BMI of the subject to the peak-to-peak interval of the PPG signal

3.5.5 Optimized feature selection using SHAP with machine learning model

Having an optimized feature combination helps to improve the model performance since having too many features can lead to overfitting in machine learning and increase computational costs. In this project, SHAP is used with best perform machine learning model to select optimized feature combinations for prediction of BP. The details about the machine learning model will be explained in Section 3.6. Figure 3.14 shows the flow chart of feature selection using SHAP with machine learning model.

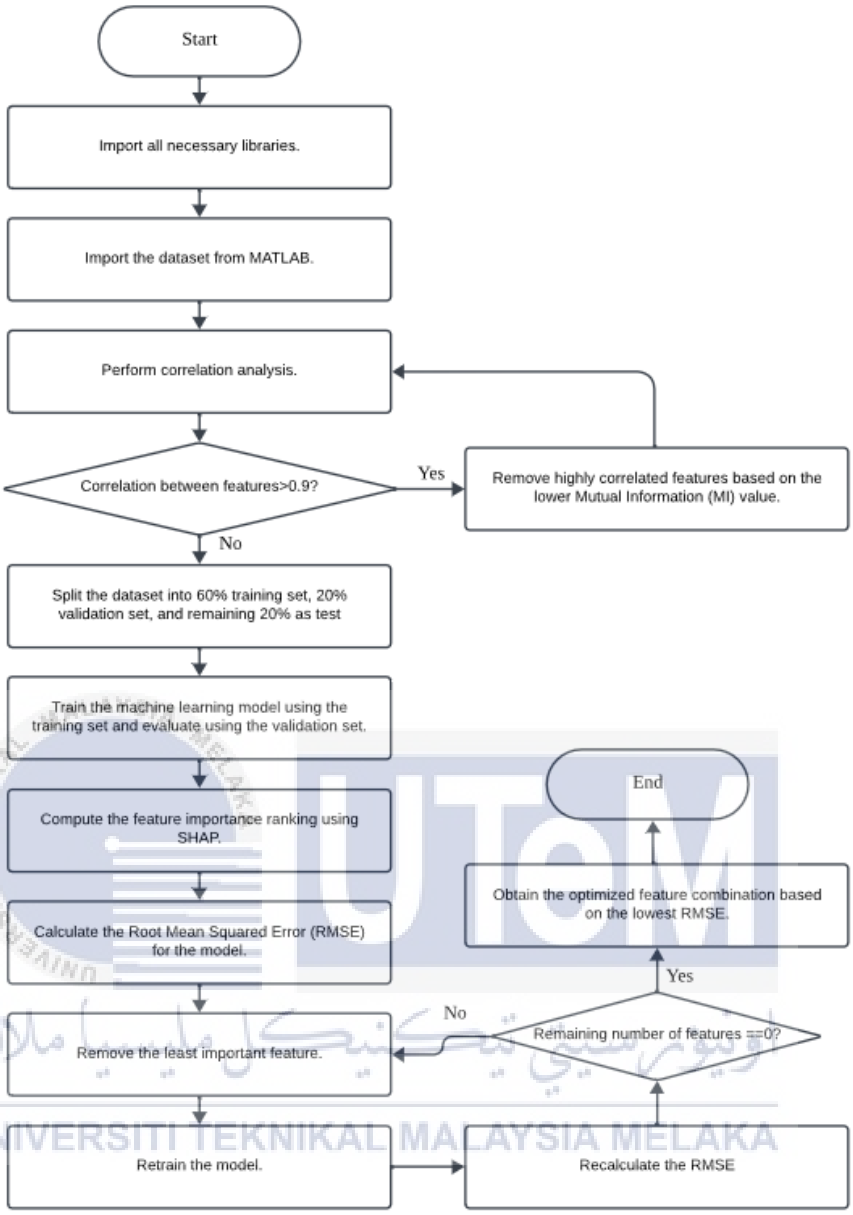


Figure 3.14: Flow chart of feature selection using SHAP with machine learning model.

SHAP works based on a fundamental concept in cooperative game theory, which aims to quantify the contribution of each player to a game or cooperative effort [21]. This method assigns an importance value which is called SHAP value to each of the features based on their contribution to the output of machine learning model. The core of this method is to calculate the sum of SHAP value of each using the formula in Eqn. (3.1):

$$g(x) = \emptyset_0 + \sum_{i=1}^{M} \emptyset_i x_i \tag{3.1}$$

where g is the explanatory model of the prediction model f, M is number of input features, $\emptyset_0$ is the average prediction of the model, $\emptyset_i$ is the SHAP values of the i-th eigenvalue $x_i \in \{0,1\}^M$ which indicates whether the corresponding feature is used where x_i=0 indicates not in used while x_i=1 indicates in used.

If the model g satisfies three desirable properties known as local accuracy, missingness, and consistency. SHAP values can be calculated using the formula in Eqn. (3.2):

$$\emptyset_i(f,x) = \sum_{S \subseteq N\{i\}} \frac{|S|!\,(M-|S|-1)!}{M!}[f_x(S \cup \{i\}) - f_x(S)] \tag{3.2}$$

where N is the set of all features, M is the number of features, S is a subset of N with 2^{M-1} potential combinations, $|S|$ is the count of elements in S, $f_x(S \cup \{i\})$ is the model's predicted value when only the features in $S \cup \{i\}$ are considered, and $f_x(S)$ is the predicted value when only the features in S are considered. The results from $[f_x(S \cup \{i\}) - f_x(S)]$ shows the boundary contribution of the i-th feature in the subset S.

From Eqns. (3.1) and (3.2), SHAP reflects both positive and negative influence of features in each sample. This aids in understanding how each feature contributes to predictions and enhances sensitivity to non-linear data.

In this project, the SHAP library is imported into the Python environment. The mean absolute SHAP value for each of the features is calculated across all subjects. The importance of each feature is determined by its mean absolute SHAP value, where a higher value indicates greater importance. The features are sorted in decreasing order of importance.

Before selecting the optimized feature combination, a feature correlation analysis is performed to eliminate redundant features. The correlation analysis is performed on the features, and features with correlation values above 0.9 are being

removed from the dataset. The correlation between features is calculated using the Pearson correlation coefficient (r) formula in Eqn.(3.3).

$$r = \frac{\sum_{i=1}^{N}(a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum_{i=1}^{N}(a_i - \bar{a})^2}\sqrt{\sum_{i=1}^{N}((b_i - \bar{b})^2}} \quad (3.3)$$

where a_i is the value of feature 1 for the i-th sample, b_i is the value of feature 2 for the i-th sample, $\bar{a}$ is the average value of feature 1 across all samples, $\bar{b}$ is the average value of feature 2 across all samples, and N is the total number of samples.

Mutual Information (MI) is then used to decide which features should be removed. MI is a measure of the mutual dependence between two random variables, with higher values indicating stronger dependence and vice versa. Among the highly correlated features, those with low MI values with respect to the SBP and DBP will be discarded. The MI values are calculated using the mutual_info_regression function from the sklearn library.

In order to select the optimized feature combination, the best perform machine learning model is first trained using the whole set features after correlation test, and feature importance scores are calculated for all features. The backward elimination technique is used to determine the appropriate number of features. This technique starts with all features included in the model and iteratively removes the least important features one at a time. The machine learning model is retrained with different numbers of features, and performance is assessed using the RMSE metric with validation set. The optimal number of features is determined based on the lowest RMSE, indicating the most informative subset for accurate prediction.

The selected features are subsequently used to train the final machine learning model, with two separate models developed, one for SBP and one for DBP. Both models use different optimized feature combinations.

3.6 Blood pressure prediction model

3.6.1 Random forest model

Random Forest is machine learning algorithm which combines multiple decision tree models in making a prediction [38, 64, 65]. It is highly effective for predictive analysis and capable of accurately estimating output values even when the features are not directly related to the targets in a linear manner.

A decision tree is a statistical model which consists of root nodes, decision nodes and leaf nodes. During the training phase, the algorithm divides the input data at each of the nodes based on specific conditions, aiming to optimize the parameters of split functions to best suit the dataset. The splitting process continues until reaching a terminal node or tree leaves, based on the hyperparameter setting. At the end of the training process, a prediction function $\hat{h}\,(X, S_n)$ is constructed over S_n, where X represents the input vector containing a number of features and S_n is the training set containing n observations.

The random forest model is the extension of decision tree, aims to improve prediction performance by constructing multiple uncorrelated decision trees, where each grown with a randomized subset of predictors. Random forest is built by randomly sampling features and training data subsets for each decision tree using a process called 'bootstrap'. A bootstrap sample is formed by randomly choosing n observations with replacement from S_n.

The bagging or bootstrap aggregation algorithm create a set of q prediction trees by choosing multiple bootstrap samples, $S_n^{\theta q}$, and apply these samples to the decision tree algorithm. The ensemble generates q outputs corresponding to each tree, denoted as $\hat{Y} = \hat{h}(X, S_n^{\theta q})$. Subsequently, the aggregation is carried out by averaging the outputs of all trees and obtain the estimation $\hat{Y}$ of the output. The estimation $\hat{Y}$ of the output can be calculated using the formula in Eqn. (3.4):

$$\hat{Y} = \frac{1}{q}\sum_{l=1}^{q} \hat{h}(X, S_n^{\theta_l}) \tag{3.4}$$

where l = 1,2,3, …,q. Figure 3.15 illustrates the construction of Random Forest.

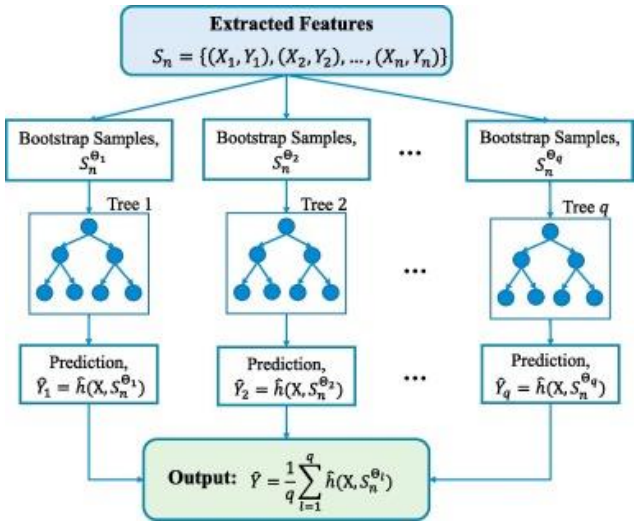


Figure 3.15: Illustration of Random Forest construction [64].

Like many other machine learning models, Random Forest has adjustable parameters that can be optimized. These include the number of trees (n_estimators), the maximum depth of tree (max_depth), the number of features sampled (max_features), the minimum number of samples in a leaf node (min_samples_leaf), and the minimum number of samples required to split a node (min_samples_split).

In summary, by combining the predictions from each tree, the Random Forest model effectively diminishes the variance of the overall model. Additionally, it exhibits insensitivity to outliers and demonstrates enhanced capabilities in preventing overfitting and ensuring stability.

3.6.2 SVR

SVR is another machine learning algorithm that having advantages in solving the small sample and non-linear regression problem. In SVR, the input sample x is transformed into a high-dimensional feature space using the non-linear mapping, $\emptyset(x)$, after which the regression function is estimated using a linear model constructed within this feature space [37, 63]. The equation used is shown in Eqn. (3.5)

$$f(x, \omega) = \omega \cdot \emptyset(x) + b \tag{3.5}$$

where ω is the weight vector, and b is the threshold. The ω an b can be obtained using the optimization equation in Eqn. (3.6)

$$\min \frac{1}{2} \|\omega\|^2 + c \sum_{i=1}^{l} (\xi_{\mathrm{i}} + \xi_i^*) \tag{3.6}$$

subject to:

$$y_i - (w^T x_i + b) < \varepsilon + \xi_i$$
$$(w^T x_i + b) - y_i < \varepsilon + \xi_i^*$$
$$\xi_i + \xi_i^* \geq 0$$

where c is a penalty factor, ε is the loss function, and ξ_i and ξ_i^* are different relaxation factors.

For simplifying the computation, Lagrange multipliers were utilized to convert the constrained optimization problems in (3.5) into a dual problem as shown in Eqn. (3.7).

$$f(x) = \sum_{i}^{l} (-\alpha_i + \alpha_i^*) K(x_i, x) + b \tag{3.7}$$

where α_i and α_i^* are Lagrange multipliers corresponding to support vectors (SVs), l is the number of SVs, and $K(x_i, x)$ is a kernel function.

There are different kernel functions, such as the RBF kernel, linear kernel, and polynomial kernel. The RBF kernel is selected because it can transform the database into a non-linear high-dimensional space compared to the linear kernel, thus allowing it to overcome the non-linear relationship between features and BP. Additionally, the RBF kernel has fewer tuning parameters compared to the polynomial kernel, making it less complex. The equation of the RBF kernel is shown in Eqn. (3.8).

$$K(x_i, x) = \exp\ (-\gamma\ \|x - x_i\|^2) \tag{3.8}$$

where γ is the kernel parameter.

From Eqns. (3.6) and (3.8), the hyperparameters of SVR that need to be tuned are the penalty factor c, the kernel parameter γ, and the loss function ε. For SVR, the input features undergo standard scaling before being fed into the model, as it relies on optimization and distance calculations.

In this project, SVR and Random Forest models were implemented using the scikit-learn library package in Python. The Python version used is 3.11.8, and the code was written in a Jupyter notebook with version 7.0.8.

3.6.3 Hyperparameter tunning with Optuna

Optuna is an open-source automatic hyperparameter optimization software framework [70]. It is design for automatically tune the hyperparameter of machine learning including both traditional machine learning and deep learning. In this project, Optuna is used to tune the hyperparameter of both SVR and Random Forest.

Optuna offers several advantages, including define-by-run programming, efficient sampling and pruning algorithms, and easy setup. Following are the step for hyperparameter using Optuna.

Firstly, the parameter search space for SVR and Random Forest is defined individually. The parameter search spaces are shown in Table 3.5. Next, the objective function, which serves as a guide in enhancing the models' performance, is defined. The prediction of BP values involves a regression task, where the MAE is set as the evaluation metric, and the objective function is defined for minimization.

Table 3.5: Parameter search spaces for SVR and Random Forest.

Algorithm	Parameter
SVR	Kernel: 'rbf'
	C: float (1, 100)
	Epsilon: float (0.01, 1)

	Gamma: ['scale', 'auto']
Random Forest	n_estimators: [50,100,200,300,400,500,600,700,800,900,1000]
	max_depth: int (10, 100)
	min_samples_split: int (2, 20)
	min_samples_leaf: int (1, 20)
	max_features:["sqrt","log2",None]

The Optuna study is run with n_trials iterations of 100 to find the best combination of hyperparameters. During each iteration, Optuna automatically tests various hyperparameter combinations and records the MAE of the objective function to monitor the model's performance. After 100 iterations, the parameters of the SVR and Random Forest models are reinitialized with the best hyperparameter combination, and the models are retrained using a combination of the training and validation datasets, resulting in optimized machine learning models. Subsequently, the models' performance is evaluated using the testing dataset.

3.6.4 Model performance evaluation

For a comprehensive evaluation of the regression model's performance, two common metrics for regression are employed, including mean absolute error (MAE), root mean squared error (RMSE). The formulas to calculate these metrics are shown in Eqns. (3.9) and (3.10). MAE will be used as the metric for Optuna optimization, while RMSE will be used in SHAP for feature selection with the machine learning model. The Pearson correlation coefficient (r) for actual values against predicted values is also calculated. The formula is similar to Equation (3.3), but with the variables changed to predicted and actual values, as shown in Equation (3.11).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \tag{3.9}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2} \tag{3.10}$$

$$r = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}} \tag{3.11}$$

where y_i is the actual value, x_i is predicted value, n is number of observations, $\bar{y}$ is the average of actual values, and $\bar{x}$ is the average of predicted values.

For the AAMI standard, the mean error (ME) and standard deviation (SD) need to be calculated. The formulas are shown in Eqns. (3.12) and (3.13).

$$ME = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i) \tag{3.12}$$

$$SD = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(z_i - \bar{z})^2} \tag{3.13}$$

where z_i represents the ME and $\bar{z}$ is the mean value.

Furthermore, to visualize the predictive performance of the model, a simple linear regression plot is generated, plotting actual values against predicted values. Additionally, Bland-Altman plots, which are analytical tools for evaluating the agreement between two measurements, are plotted to investigate the agreement between actual and predicted values.

MAE, RMSE, and ME determine the errors between predicted and actual values. Higher values for these metrics indicate lower prediction accuracy. SD measures the variation of a set of values relative to their mean value. Lower SD

indicates a better performance. The closer the Pearson correlation coefficient (r) is to one, the more accurately the predicted data matches the actual data.

For regression plot, the closer the data points to the regression line, the better is the prediction of model. For Bland-Altman plots, if the mean difference is close to zero and a larger proportion of predicted values fall within the 95% limits of agreement (± 1.96 SD), it indicates that the actual and predicted values are highly consistent.

The performance of the BP prediction model is assessed based on criteria set by AAMI standard [71] and BHS protocol [72], widely used by many researchers to validate BP measuring device. According to the AAMI standard, the overall mean error (ME) and standard deviation (SD) between the tested BP device and the reference should be less or equal to 5 mmHg and 8 mmHg, respectively. In the case of BHS protocol, grades are assigned to devices based on the cumulative percentage errors under three thresholds: 5, 10, and 15 mmHg, as shown in Table 3.6. To meet the BHS protocol, the tested device must achieve at least grade B. Both AAMI and BHS required a minimum of 85 subjects for the assessment.

Table 3.6: Grading criteria used by the BHS protocol [72].

Grade	≤5mmHg	≤10mmHg	≤15mmHg
A	60%	85%	95%
B	50%	75%	90%
C	40%	65%	85%
D	Worse than C		

3.7 Experiments

In order develop the final model for SBP and DBP prediction, two experiments were conducted.

3.7.1 Data partitioning

As mentioned earlier in Section 3.4.1, we selected 10 signal segments and gathered data on the subjects' age, gender, weight, height, and body mass index (BMI) from 500 participants, resulting in a total of 5000 data used in this project..

The data is split into 60% for the training set to train the model, 20% for the validation set for hyperparameter tuning, and the remaining 20% for the test set to verify the model's performance on unseen data. The data is split by subject, ensuring that the same subject does not appear in both the training and test sets, which prevents data leakage.

3.7.2 Experiment 1

The experiment 1 involved model selection, comparing Support Vector Regression (SVR) and Random Forest to determine the model that provides better results for BP prediction. The model with superior performance is chosen for the final SBP and DBP models.

Both models are first trained using the training set. Afterward, Optuna is used to tune the hyperparameters of SVR and Random Forest. The parameter search space for SVR and Random Forest is depicted in Table 3.5. Subsequently, the models are tested using the test set, and the optimal model is selected based on the evaluation results.

3.7.3 Experiment 2

Experiment 2 focused on feature testing, involving manipulation of features to evaluate their impact on BP prediction accuracy. The combinations of features used in this experiment are listed in Table 3.7:

Table 3.7: Features combinations for Experiment 2.

No	Features combination	Number of

		features
1	Using all features from PPG signals, ECG signals, and demographic features.	78
2	Using features from PPG and ECG signals without demographic features.	68
3	Using features from PPG signals and demographic features.	55
4	Using features from only PPG signals for prediction.	45

The results from the best combination of features are selected and compared with the results obtained when using the features selected by SHAP with the best machine learning model. This process is conducted for both SBP and DBP.

3.8 Ethics and safety of method

Even though this project involves simulation only, it is important to consider the ethics and safety of the method. The dataset utilized in this project is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) license, which permits the sharing and adaptation of the dataset for non-commercial purposes. Consequently, there are no issues concerning data privacy and confidentiality, as the dataset is legally obtained and used in accordance with its licensing terms.

One potential hazard of this project is the possibility of inaccuracies in the model's predictions, which could lead to misdiagnoses or incorrect treatment decisions if deployed in a clinical setting. In order to mitigate this risk, a comprehensive risk assessment approach is employed. The two most widely used protocols for the BP measurement device, AAMI and BHS, are used to evaluate the performance and reliability of the model.

In order to ensure the safety of practice, the model should undergo real-world experimentation before deployment in clinical settings to assess its efficacy and safety in practical scenarios. This includes trials in diverse patient populations to assess the model's adaptability and generalizability. Furthermore, continuous monitoring and validation of the model's performance over time are essential to ensure its safety and effectiveness over time.

3.9 Summary of methodology

PPG and ECG signals are utilized for BP prediction. The process involves of preprocessing these signals, extracting features from both signals as well as demographic features, identifying an optimized feature combination, and predicting BP through the developed model. Signal segments from a total of 500 subjects are utilized, and the reliability of the data is discussed in Section 3.4.1. A total of 78 features have been successfully extracted from each subject, chosen from related and validated research papers. Finally, the optimized feature combination is identified using SHAP with a machine learning model, and BP is predicted through the developed model using the selected machine learning model.

Two experiments have been conducted, including selecting the best machine learning model and identify the optimized features combination for BP prediction. The methodology is validated by using all the evaluation metrics, as presented in Section 4.2.4. The final blood pressure prediction models are validated by evaluating the ME and SD for AAMI standard and cumulative percentage error for BHS protocol, as presented in Section 4.3.

CHAPTER 4

RESULTS AND DISCUSSIONS

A total of 78 features are extracted from PPG and ECG signals, in addition to SBP and DBP reference values derived from the ABP signal, along with demographic features for 500 subjects. Appendix B provides detailed SBP and DBP values for these 500 subjects across 10 signal segments per subject.

4.1 Experiment 1 result

Based on the results from Optuna optimization using validation set, the best parameters for SVR and Random Forest for SBP are shown in Table 4.1, while those for DBP are shown in Table 4.2.

Table 4.1: Best hyperparameter for SVR and Random Forest for SBP prediction.

SVR	
C	17.834386610306435
Epsilon	0.030775864724018196
Gamma	scale
Random Forest	
n_estimators	200
max_depth	22
min_samples_split	11
min_samples_leaf	7
max_features	sqrt

Table 4.2: Best hyperparameter for SVR and Random Forest for DBP prediction.

SVR	
C	36.67809822911926
Epsilon	0.9950140307880824
Gamma	auto
Random Forest	
n_estimators	50
max_depth	61
min_samples_split	4
min_samples_leaf	3
max_features	log2

By using the best parameters, both models were retrained with the combined training and validation sets. The trained models are then tested with the test set. The results for the prediction of SBP are presented in Table 4.3, while the prediction results for DBP are shown in Table 4.4. The better results are highlighted in bold.

Table 4.3: Prediction of SBP using SVR and Random Forest with full feature set

SBP	SVR	**Random Forest**
ME± SD (mmHg)	1.8092 ± 14.6449	**0.9673 ± 14.0735**
r	0.5008	**0.5146**
Cumulative percentage error ≤ 5	27.3 %	**29.0 %**
Cumulative percentage error ≤ 10	52.5 %	**53.3 %**
Cumulative percentage error ≤ 15	70.8 %	**72.7 %**

Table 4.4: Prediction of DBP using SVR and Random Forest with full feature set.

DBP	SVR	**Random Forest**
ME± SD (mmHg)	1.1894± 9.7163	**0.041 ± 9.3297**
r	**0.4142**	0.3890
Cumulative percentage error ≤ 5	**41.2 %**	40.1 %
Cumulative percentage error ≤ 10	70.0 %	**74.6 %**
Cumulative percentage error ≤ 15	88.2 %	**90.0 %**

From Tables 4.3 and 4.4, the results obtained from Random Forest show better results compared to SVR when training and testing using the full set of features for both SBP and DBP. Even though the cumulative percentage error ≤ 5 and r of SVR is slightly better than that of Random Forest in DBP prediction, overall results show that Random Forest performs better. Hence, Random Forest was chosen to conduct the next experiment and develop the prediction model for both SBP and DBP.

4.2 Experiment 2 results

4.2.1 Features analysis

Tables 4.5 and 4.6 show the prediction results for different feature combinations for SBP and DBP, respectively, with the better results highlighted in bold.

Table 4.5: Feature analysis for SBP.

SBP	**PPG+ECG +Demographic**	PPG+ECG	PPG +Demographic	PPG
ME ± SD (mmHg)	**0.9673 ± 14.0735**	1.0798 ± 14.0859	1.0409 ± 14.7398	1.275 ± 14.7802
r	**0.5146**	0.5118	0.4382	0.4362
Cumulative percentage error ≤ 5	29.0 %	**29.4 %**	26.1 %	27.1 %
Cumulative percentage error ≤ 10	53.3 %	**53.7 %**	49.8 %	52.2 %
Cumulative percentage error ≤ 15	72.7 %	**73.4 %**	70.5 %	69.7 %

According to Table 4.5, for SBP model, the combination of all features resulted in better prediction accuracy. By comparing Mean Error (ME), by using only PPG signals for prediction produced the highest ME value of 1.275 mmHg. Meanwhile, by including demographic features, it slightly reduced the error to 1.0409 mmHg, whereas incorporating PPG with ECG signals slightly increased the error to 1.0798 mmHg. Utilizing all features from both PPG and ECG signals, along with demographic features, achieved the lowest ME value at 0.9673 mmHg.

For SBP, by using only PPG as a reference, incorporating all features resulted in a 24.13% reduction in error. The calculation is detailed in Eqn. (4.1).

$$Percentage\ reduction = \frac{0.9673 - 1.275}{1.275} \times 100\% \quad (4.1)$$

$$= -24.13\%$$

Table 4.6: Feature analysis for DBP.

DBP	**PPG+ECG +Demographic**	PPG+ECG	PPG +Demographic	PPG
ME ± SD (mmHg)	**0.041 ± 9.3297**	0.1227 ± 9.3486	0.1936 ± 9.7038	0.2216 ± 9.8246
r	**0.3890**	0.3843	0.3037	0.2771
Cumulative percentage error ≤ 5	**40.1 %**	**40.1 %**	38.3 %	39.7 %
Cumulative percentage error ≤ 10	**74.6 %**	74.3 %	71.3 %	71.1 %
Cumulative percentage error ≤ 15	90.0 %	90.0 %	**90.1 %**	88.3 %

Based on Table 4.6, consistent results were observed for DBP, where the ME achieves the lowest value when the prediction is done by using all features while the highest error achieved when using only PPG signal for prediction.

For DBP, by using only PPG as a reference, incorporating all features resulted in an 81.50% reduction in error. The calculation is detailed in Eqn. (4.2).

$$Percentage\ reduction = \frac{0.041 - 0.2216}{0.2216} \times 100\% \tag{4.2}$$

$$= -81.50\%$$

The result from feature analysis for both SBP and DBP indicate that the inclusion of ECG signal and demographic features enhance the prediction accuracy of the models.

4.2.2 Feature selection using SHAP with Random Forest for SBP

After conducting feature correlation analysis, the highly correlated features (correlation > 0.9) for both SBP and DBP are listed in Appendix C. The bold-highlighted features, which have the lower MI values, will be removed. Following this removal, 58 features remain for SBP, while 59 features remain for DBP.

The features are ranked based on the mean absolute SHAP values of each feature, computed using the Python SHAP library. A higher mean absolute SHAP value indicates the tendency of feature to contribute significantly to the output of the model.

Afterward, features with the least mean absolute SHAP values are iteratively removed, and the model is retrained to find the optimized feature combination for predicting SBP. Figure 4.1 displays the results of each iteration of the processes, where the y-axis indicates the RMSE of the prediction, and the x-axis indicates the number of features. The optimized feature combination is determined based on the lowest RMSE.

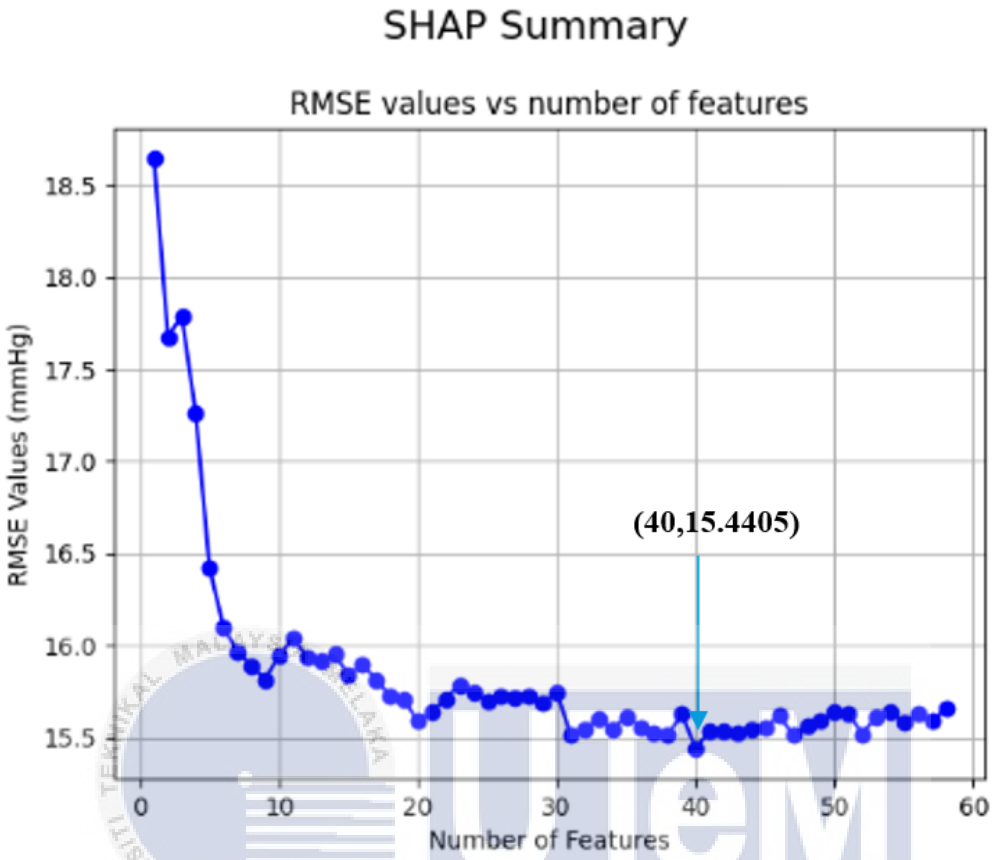


Figure 4.1: RMSE of model across different number of features used for SBP prediction.

The results show that the lowest RMSE is achieved when the first 40 highest-ranking features are used, with an RMSE value of 15.4405 mmHg on validation set. The optimized features combination for SBP are detailed in Appendix D. The feature that contribute the most to the SBP prediction is $Angle_{ed}$.

4.2.3 Optimized features combination for DBP

The same process is repeated for the Random Forest DBP model. Figure 4.2 displays the model's RMSE for different numbers of features used. Features are iteratively removed starting from the least significant value.

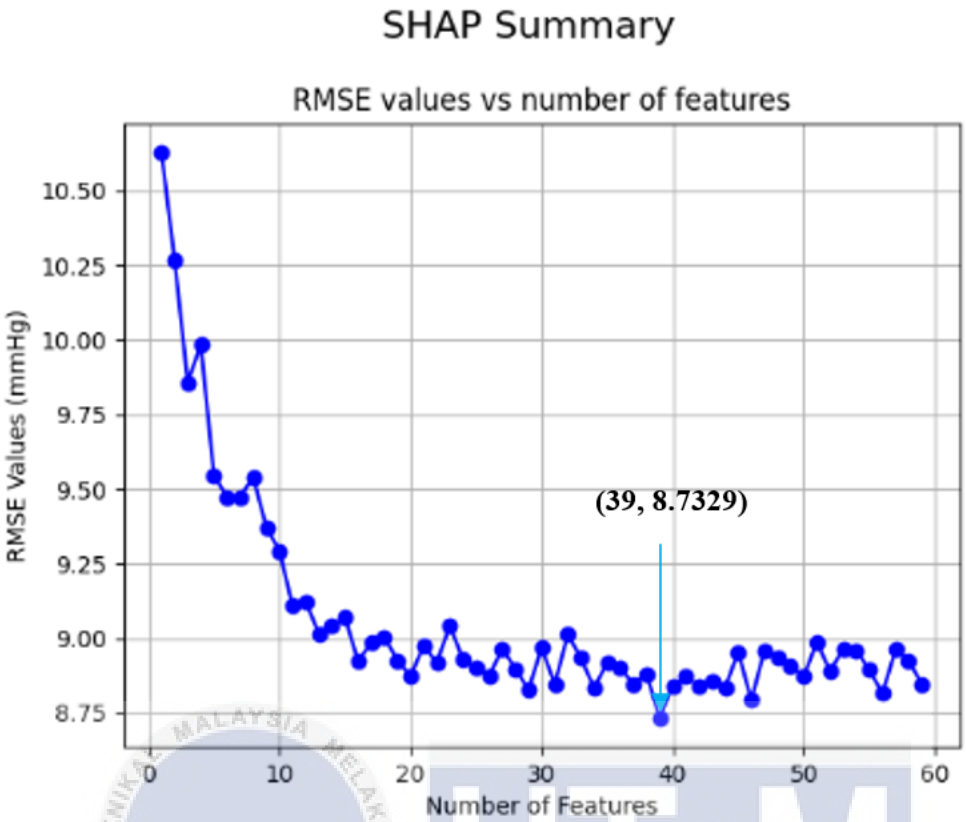


Figure 4.2: RMSE of model across different number of features used for DBP prediction.

The 39 highest-ranking features are selected for DBP based on the lowest RMSE of 8.7329 mmHg on the validation set. The features name are detailed in Appendix E. The feature with the highest mean absolute SHAP value for DBP prediction is *PT*, indicating that this feature contributes the most in predicting DBP.

4.2.4 Comparison of the best feature combinations with optimized combinations from SHAP with Random Forest

Based on the optimized features selected in Sections 4.2.2 and 4.2.3, both the SBP and DBP models were retrained using the training and validation sets, including only the selected features. The results obtained using the test set were compared with the best feature combinations from Section 4.2.1, and are presented in Tables 4.7 and 4.8 for SBP and DBP, respectively.

Table 4.7: Comparison between features for SBP.

SBP	PPG+ECG +Demographic (Number of features=78)	Feature selection using SHAP with Random Forest (Number of features=40)
ME± SD (mmHg)	0.9673 ± 14.0735	**0.9464 ± 14.0502**
r	0.5146	**0.5161**
Cumulative percentage error ≤ 5	**29.0 %**	28.6%
Cumulative percentage error ≤ 10	53.3 %	**53.6%**
Cumulative percentage error ≤ 15	72.7 %	**74.2%**

Based on Table 4.7, by using the optimized feature combination, it is found that the number of features for SBP is reduced from 78 to 40, a reduction of approximately 48.72%. Additionally, the performance shows slightly overall improvement. For instance, ME, the error decreased from 0.9673 mmHg to 0.9464 mmHg.

Reducing the number of features while maintaining or improving predictive accuracy helps streamline the model's complexity and enhances the efficiency of BP predictions, requiring fewer feature extractions. This not only speeds up the process but also decreases the risk of overfitting, where the model might otherwise learn to fit the noise in the training data.

Table 4.8: Comparison between features for DBP.

DBP	PPG+ECG +Demographic (Number of features=78)	Feature selection using SHAP with Random Forest (Number of features=39)
ME± SD (mmHg)	**0.041± 9.3297**	0.0574 ± 9.3436
r	**0.3890**	0.3868
Cumulative percentage error ≤ 5	**40.1 %**	39.6 %
Cumulative percentage error ≤ 10	74.6 %	**74.8 %**
Cumulative percentage error ≤ 15	90.0 %	**90.4 %**

Based on Table 4.8, the optimized feature combination reduces the number of features for DBP prediction from 78 to 39, representing a 50% of reduction. The ME increased slightly from 0.0574 mmHg to 0.041 mmHg. However, this difference is not significant compared to other feature combinations in Table 4.6.

These findings indicate that reducing the number of features successfully maintains prediction accuracy. This reduction not only simplifies the model but also enhances its potential applicability in real-world scenarios by reducing computational load.

Figures 4.3 and 4.4 display the linear regression plots for actual versus predicted SBP and DBP values, respectively, by using the optimized feature combinations. The closer the data points are to the regression line, the better the model's predictions.

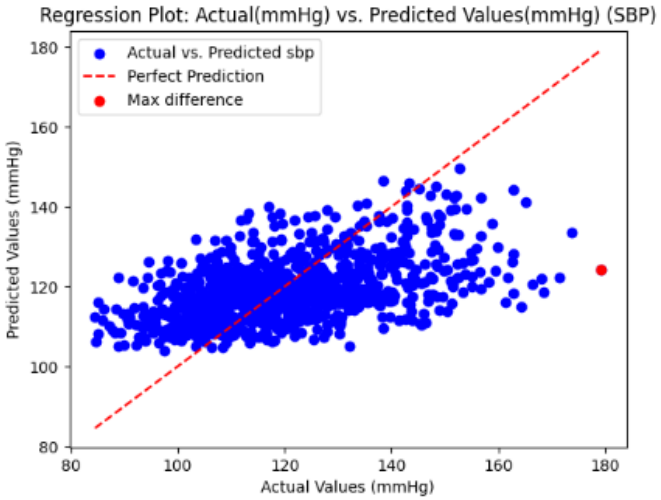


Figure 4.3: The regression plot of actual against predicted values for SBP.

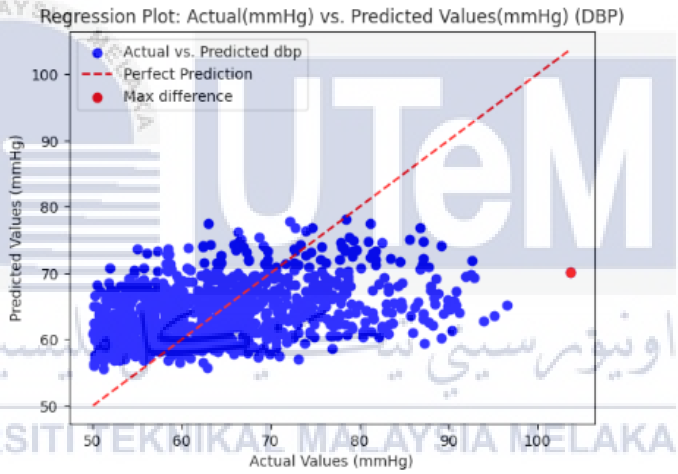


Figure 4.4: The regression plot of actual against predicted values for DBP.

From the graphs, both the SBP and DBP models are able to provide good predictions for data in the middle range. However, there are noticeable deviations, particularly at higher and lower actual values.

Figures 4.5 and 4.6 show the Bland-Altman plot for SBP and DBP respectively.

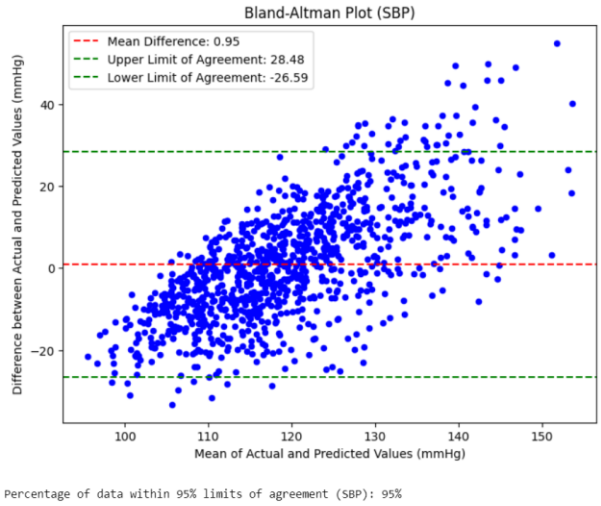


Figure 4.5: Bland-Altman plot for SBP.

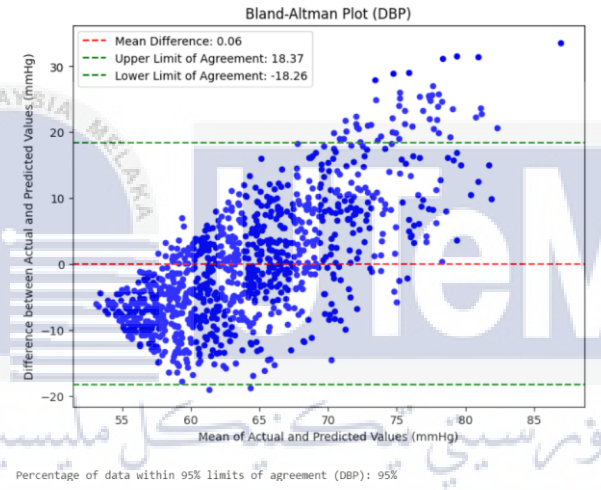


Figure 4.6: Bland-Altman plot for DBP.

Referring to Figures 4.5 and 4.6, the mean differences are 0.95 and 0.06 for SBP and DBP, respectively. These values differ insignificantly from 0 indicate that nonexistence of fixed bias for both models. The differences are more widely spread at both the lower and higher mean values, indicating variability in prediction accuracy across the range of values. There are several points that lie outside the 95% limits of agreement, indicating some outliers. However, for both models, 95% of the data are found fall within these limits, indicating overall good agreement between the actual and predicted values.

4.3 Evaluation of the SBP and DBP model based on AAMI and BHS

As previously discussed in Section 3.6.4, the AAMI standard requires that the overall ME and SD between the tested blood pressure device and the reference should be less than or equal to 5 mmHg and 8 mmHg, respectively. Additionally, for the BHS protocol, the tested device must achieve at least grade B.

Both the AAMI and BHS standards require testing on a total of 85 subjects. After data splitting, the number of subjects remaining is 100, fulfilling this requirement. Tables 4.9 and 4.10 show the evaluation of model based on AAMI standard and BHS protocol respectively.

Table 4.9: Evaluation of model based on AAMI standard.

BP	ME (≤5 mmHg)	SD (≤ 8mmHg)
SBP	0.9464	14.0502
DBP	0.0574	9.3436

Table 4.10: Evaluation of model based on BHS protocol.

BP	Cumulative percentage error			Grade
	≤5 mmHg	≤10 mmHg	≤15 mmHg	
SBP	28.6 %	53.6 %	74.2 %	Worse than C
DBP	39.6 %	74.8 %	90.4 %	Worse than C

From Table 4.9, ME for both SBP and DBP models are 0.9464 mmHg and 0.0574 mmHg, respectively, which fall within the range of AAMI standard. However, SD of SBP and DBP, which are 14.0502 mmHg and 9.3436 mmHg, respectively, exhibit larger value than 8 mmHg. Table 4.10 shows that cumulative percentage errors of both models under 3 different categories based on BHS protocol. Both models fall below a C grade. The predictive accuracy for SBP is lower than that

for DBP due to the greater dynamism of SBP, which reflects higher pressures during the heartbeat. With its wider range and more variations, SBP poses a greater challenge for accurate prediction

From the results obtained, neither the SBP nor the DBP models meet the requirements set by AAMI and BHS. However, these findings are supported by recent researches that addressing data leakage [15, 16, 17].

4.4 Comparison with other research papers

In comparison with other research papers, the study referenced [15] is chosen for comparison due to high similarity in the use of feature-based machine learning, specifically CatBoost and XGBoost. CatBoost is specifically highlighted due to its superior performance. Table 4.11 presents the comparison results with the referenced paper, where data leakage issue is addressed.

Table 4.11: Comparison result with referenced paper [15].

Criteria	[15]	Proposed method
Dataset (subject/signal segment)	UCI dataset (- / 50182)	VitalDB (500 / 5000)
Method	CatBoost	Random Forest+ SHAP feature select
Number of features used (SBP/DBP)	133 / 133	39 / 40
ME±SD (SBP/DBP)	-1.23±22.368 mmHg / -0.257±9.784mmHg	**0.9464 ± 14.0502** mmHg / **0.0574 ± 9.3436** mmHg
***r* (SBP/DBP)**	0.218 / 0.306	**0.5146 / 0.3890**
CP<5mmHg (SBP/DBP)	16.2% / **40.7**%	**28.6 %** / 39.6 %
CP<10mmHg (SBP/DBP)	32.3% / 73.2%	**53.6 % / 74.8** %
CP<15mmHg	47.6% / 90.1%	**74.2 % / 90.4** %

(SBP/DBP)		

Noted that the reference paper considered data with SBP values greater than 70mmHg, DBP values less than 141mmHg, and a difference between SBP and DBP greater than 10mmHg, while in this project, SBP values range from 80 to 180 mmHg, and DBP values are limited between 50 to 130 mmHg. Based on the results, the SBP and DBP models in this project generally exhibit outstanding performance.

4.5 Summary of the result and discussion

In summary, the Random Forest model demonstrates superior performance compared to SVR. Additionally, the integration of SHAP with Random Forest optimizes feature selection, helping to maintain or even enhance predictive accuracy while number of features used is reduced significantly. This reduction in feature count contributes to reducing the model complexity, enhancing computational efficiency, and reducing the risk of overfitting, which is particularly crucial in real-time applications or scenarios with resource limitations.

Even though neither model meets the requirements of the AAMI standard and BHS protocol, the models in this project generally outperform those in the referenced paper where data leakage issue is addressed, disparities exist in terms of data utilization and thresholds for SBP and DBP.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In conclusion, the feature extraction code for PPG and ECG signals has been successfully implemented using MATLAB. Next, for the model performance testing, the Random Forest algorithm demonstrated superior performance compared to SVR in predicting SBP and DBP, thus it was selected to form the blood pressure prediction model.

After that, the results from feature analysis revealed that all features from PPG and ECG signals, as well as demographic features, contribute to the prediction of BP. The inclusion of ECG signals and demographic features reduces the ME for prediction compared to using only PPG signals by approximately 24.13% for SBP and 81.50% for DBP. The application of optimized feature selection through SHAP with Random Forest helped maintain or even enhance predictive accuracy while using significantly fewer features, specifically a reduction of 48.72% for SBP and 50% for DBP.

The evaluation of the model based on AAMI standard and BHS protocol indicated that the model was unable to fulfill the criteria when addressing the data leakage issue, consistent with findings from recent papers.

The results obtained demonstrate that the extracted features are associated with BP and are suitable for application in the BP prediction model. The inclusion of ECG signals shows potential for enhancing BP predictions. Nonetheless, creating a high-performance BP prediction model that fulfills the AAMI standard and BHS

protocol remains challenging due to variations in these signals among individuals and the complexity of physiological factors.

5.2 Future Works

Nevertheless, there are still limitations in this project. Firstly, not all the fiducial points of PPG and ECG signals were perfectly detected due to signal variability. Additionally, this project involved using relatively clean PPG and ECG signals, whereas in real-life applications, the signal may be corrupted due to noise and sensors used. Lastly, the results of the model in real-life applications have not been explored.

For future works, it is imperative to delve into advanced signal processing techniques aimed at refining the detection of fiducial points in PPG and ECG signals, thereby enhancing feature extraction. Furthermore, validation studies carried out in real-life clinical settings will be essential for evaluating the performance of the BP prediction model across diverse patient populations and conditions. Moreover, it is necessary to further evaluate the model's robustness to signal variability and noise encountered in real-world scenarios.

In terms of feature analysis, it is advisable to incorporate more features prior to feature selection to thoroughly explore high-potential features for predicting BP. Besides that, exploring features that do not rely on the detection of fiducial points should be emphasized to enhance the model's applicability to real-life conditions. Further exploration into integrating additional signals, such as ballistocardiogram signals, could further be explored to enhance prediction accuracy.

Considering the difficulty of creating a blood pressure (BP) prediction model that performs well for all individuals, one potential strategy is to create a personalized model that uses an individual's historical data.

REFERENCES

[1] American Heart Association, "Understanding Blood Pressure Readings," American Heart Association, May 30, 2023. [Online]. Available: https://www.heart.org/en/health-topics/high-blood-pressure/understanding-blood-pressure-readings [Accessed 2023].

[2] World Health Organization, "Hypertension," World Health Organization, Mar. 16, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/hypertension [Accessed 2023].

[3] Centers for Disease Control and Prevention, "Facts about Hypertension," Centers for Disease Control and Prevention, Sep. 27, 2021. [Online]. Available: https://www.cdc.gov/bloodpressure/facts.htm [Accessed 2023].

[4] C. Razo et al., "Effects of elevated systolic blood pressure on ischemic heart disease: a Burden of Proof study," *Nature Medicine*, vol. 28, no. 10, pp. 2056–2065, Oct. 2022. doi: 10.1038/s41591-022-01974-1.

[5] *Statista*, "Malaysia: most common causes of death 2021," [Online]. Available:https://www.statista.com/statistics/1343021/malaysia-most-common-causes-of-death/ [Accessed 2024].

[6] P. Muntner et al., "Blood Pressure Assessment in Adults in Clinical Practice and Clinic-Based Research: JACC Scientific Expert Panel," *Journal of the American College of Cardiology*, vol. 73, no. 3, pp. 317–335, Jan. 2019. doi: 10.1016/j.jacc.2018.10.069.

[7] M. H. Yang et al., "The Effect of Lifestyle Changes on Blood Pressure Control among Hypertensive Patients," *Korean Journal of Family Medicine*, vol. 38, no. 4, p. 173, 2017. doi: 10.4082/kjfm.2017.38.4.173.

[8] F. Guarracino and P. Bertini, "Perioperative Hypotension: causes and remedies," *Journal of Anesthesia, Analgesia and Critical Care*, vol. 2, no. 1, Apr. 2022. doi: 10.1186/s44158-022-00045-8.

[9] B. Alexander, M. Cannesson, and T. J. Quill, “Blood Pressure Monitoring,” *Anesthesia Equipment: Principles and Applications (Expert Consult: Online and Print)*, pp. 273–282, Jan. 2013, doi: 10.1016/B978-0-323-11237-6.00012-1.

[10] F. P. Cappuccio, "The Role of Nocturnal Blood Pressure and Sleep Quality in Hypertension Management," *European Cardiology Review*, vol. 15, Aug. 2020. doi: 10.15420/ecr.2020.13.

[11] T. Athaya and S. Choi, "A Review of Noninvasive Methodologies to Estimate the Blood Pressure Waveform," *Sensors*, vol. 22, no. 10, p. 3953, May 2022. doi: 10.3390/s22103953.

[12] L. Peter, N. Noury, and M. Cerny, "A review of methods for non-invasive and continuous blood pressure monitoring: Pulse transit time method is promising?," *IRBM*, vol. 35, no. 5, pp. 271–282, Oct. 2014. doi: 10.1016/j.irbm.2014.07.002.

[13] M. Kachuee, M. M. Kiani, H. Mohammadzade and M. Shabany, "Cuff-less high-accuracy calibration-free blood pressure estimation using pulse transit time," *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, Lisbon, Portugal, 2015, pp. 1006-1009, doi: 10.1109/ISCAS.2015.7168806.

[14] M. K. F. Wong, H. Hei, S. Z. Lim, and E. Y.-K. Ng, "Applied machine learning for blood pressure estimation using a small, real-world electrocardiogram and photoplethysmogram dataset," *Mathematical Biosciences and Engineering*, vol. 20, no. 1, pp. 975–997, 2022. doi: 10.3934/mbe.2023045.

[15] F. M. Dias, T. B. S Costa, D. A. C Cardenas, M. A. F Toledo, J. E. Krieger and M. A. Gutierrez, "A Machine Learning Approach to Predict Arterial Blood Pressure from Photoplethysmography Signal," 2022 Computing in

Cardiology (CinC), Tampere, Finland, 2022, pp. 1-4, doi: 10.22489/CinC.2022.238.

[16] T. B. D. S. Costa et al., "Blood Pressure Estimation From Photoplethysmography by Considering Intra- and Inter-Subject Variabilities: Guidelines for a Fair Assessment," in IEEE Access, vol. 11, pp. 57934-57950, 2023, doi: 10.1109/ACCESS.2023.3284458.

[17] S. González, W.-T. Hsieh, and T. P.-C. Chen, "A benchmark for machine-learning based non-invasive blood pressure estimation using photoplethysmogram," Scientific Data, vol. 10, no. 1, Mar. 2023, doi: 10.1038/s41597-023-02020-6.

[18] D. Castaneda, A. Esparza, M. Ghamari, C. Soltanpur, and H. Nazeran, "A Review on Wearable Photoplethysmography Sensors and Their Potential Future Applications in Health Care," *International Journal of Biosensors & Bioelectronics*, vol. 4, no. 4, 2018. doi: 10.15406/ijbsbe.2018.04.00125.

[19] S. S. Mousavi, M. Firouzmand, M. Charmi, M. Hemmati, M. Moghadam, and Y. Ghorbani, "Blood pressure estimation from appropriate and inappropriate PPG signals using A whole-based method," *Biomedical Signal Processing and Control*, vol. 47, pp. 196–206, Jan. 2019. doi: 10.1016/j.bspc.2018.08.022.

[20] S. Abdullah, A. Hafid, M. Folke, M. Lindén, and A. Kristoffersson, "PPGFeat: a novel MATLAB toolbox for extracting PPG fiducial points," *Frontiers in Bioengineering and Biotechnology*, vol. 11, p. 1199604, Jun. 2023. doi: 10.3389/fbioe.2023.1199604.

[21] J. Liu, S. Hu, Z. Xiao, Q. Hu, D. Wang, and C. Yang, "A novel interpretable feature set optimization method in blood pressure estimation using photoplethysmography signals," *Biomedical Signal Processing and Control*, vol. 86, p. 105184, Sep. 2023. doi: 10.1016/j.bspc.2023.105184.

[22] A. D. Choudhury, R. Banerjee, A. Sinha and S. Kundu, "Estimating blood pressure using Windkessel model on photoplethysmogram," *2014 36th*

Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Chicago, IL, USA, 2014, pp. 4567-4570, doi: 10.1109/EMBC.2014.6944640.

[23] V. Tan, A. Huong, and X. Ngu, "Continuous Monitoring of Blood Pressure Based on Heart Pulse Analysis," *Journal of Physics: Conference Series*, vol. 1049, p. 012062, Jul. 2018. doi: 10.1088/1742-6596/1049/1/012062.

[24] R. Lazazzera, Y. Belhaj, and G. Carrault, "A New Wearable Device for Blood Pressure Estimation Using Photoplethysmogram," *Sensors*, vol. 19, no. 11, p. 2557, Jun. 2019. doi: 10.3390/s19112557.

[25] A. Alberdi, A. Aztiria, and A. Basarab, "Towards an automatic early stress recognition system for office environments based on multimodal measurements: A review," *Journal of Biomedical Informatics*, vol. 59, pp. 49–75, Feb. 2016. doi: 10.1016/j.jbi.2015.11.007.

[26] A. Barros et al., "Data Improvement Model Based on ECG Biometric for User Authentication and Identification," *Sensors*, vol. 20, no. 10, p. 2920, May 2020. doi: 10.3390/s20102920.

[27] J. R. Hampton and J. Hampton, *The ECG Made Easy*, 9th ed. Edinburgh; New York (NY): Elsevier, 2019.

[28] S. S. Mousavi, M. Charmi, M. Firouzmand, M. Hemmati, M. Moghadam, and Y. Ghorbani, "ECG-Based Blood Pressure Estimation Using Mechano-Electric Coupling Concept," *arXiv (Cornell University)*, Aug. 2020.

[29] E. Finnegan, S. Davidson, M. Harford, P. Watkinson, L. Tarassenko, and M. Villarroel, "Features from the photoplethysmogram and the electrocardiogram for estimating changes in blood pressure," *Scientific Reports*, vol. 13, no. 1, Jan. 2023. doi: 10.1038/s41598-022-27170-2.

[30] M. Simjanoska, M. Gjoreski, M. Gams, and A. Madevska Bogdanova, "Non-Invasive Blood Pressure Estimation from ECG Using Machine Learning

Techniques," *Sensors*, vol. 18, no. 4, p. 1160, Apr. 2018. doi: 10.3390/s18041160.

[31] G. Thambiraj, U. Gandhi, U. Mangalanathan, V. J. M. Jose, and M. Anand, "Investigation on the effect of Womersley number, ECG and PPG features for cuffless blood pressure estimation using machine learning," *Biomedical Signal Processing and Control*, vol. 60, p. 101942, Jul. 2020. doi: 10.1016/j.bspc.2020.101942.

[32] A. Paviglianiti, V. Randazzo, S. Villata, G. Cirrincione, and E. Pasero, "A Comparison of Deep Learning Techniques for Arterial Blood Pressure Prediction," *Cognitive Computation*, vol. 14, no. 5, pp. 1689–1710, Aug. 2021. doi: 10.1007/s12559-021-09910-0.

[33] M. Elgendi et al., "The use of photoplethysmography for assessing hypertension," *npj Digital Medicine*, vol. 2, no. 1, pp. 1–11, Jun. 2019. doi: 10.1038/s41746-019-0136-7.

[34] T. Le *et al*., "Continuous Non-Invasive Blood Pressure Monitoring: A Methodological Review on Measurement Techniques," in *IEEE Access*, vol. 8, pp. 212478-212498, 2020, doi: 10.1109/ACCESS.2020.3040257.

[35] M. Sharma et al., "Cuff-Less and Continuous Blood Pressure Monitoring: A Methodological Review," *Technologies*, vol. 5, no. 2, p. 21, Jun. 2017. doi: 10.3390/technologies5020021.

[36] M. Kachuee, M. M. Kiani, H. Mohammadzade and M. Shabany, "Cuffless Blood Pressure Estimation Algorithms for Continuous Health-Care Monitoring," in *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 4, pp. 859-869, April 2017, doi: 10.1109/TBME.2016.2580904.

[37] S. Chen, Z. Ji, H. Wu, and Y. Xu, "A Non-Invasive Continuous Blood Pressure Estimation Approach Based on Machine Learning," *Sensors*, vol. 19, no. 11, p. 2585, Jun. 2019. doi: 10.3390/s19112585.

[38] G. Ma, J. Zhang, J. Liu, L. Wang, and Y. Yu, "A Multi-Parameter Fusion Method for Cuffless Continuous Blood Pressure Estimation Based on Electrocardiogram and Photoplethysmogram," *Micromachines*, vol. 14, no. 4, p. 804, Mar. 2023. doi: 10.3390/mi14040804.

[39] F. Miao *et al*., "A Novel Continuous Blood Pressure Estimation Approach Based on Data Mining Techniques," in *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 6, pp. 1730-1740, Nov. 2017, doi: 10.1109/JBHI.2017.2691715.

[40] S. Yang et al., "Blood pressure estimation from photoplethysmogram and electrocardiogram signals using machine learning," *Nottingham ePrints (University of Nottingham)*, Jan. 2018. doi: 10.1049/cp.2018.1721.

[41] X. -R. Ding, Y. -T. Zhang, J. Liu, W. -X. Dai and H. K. Tsang, "Continuous Cuffless Blood Pressure Estimation Using Pulse Transit Time and Photoplethysmogram Intensity Ratio," in *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 5, pp. 964-972, May 2016, doi: 10.1109/TBME.2015.2480679.

[42] S. Kılıçkaya, A. Güner and B. Dal, "Comparison of Different Machine Learning Techniques for the Cuffless Estimation of Blood Pressure using PPG Signals," *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, Ankara, Turkey, 2020, pp. 1-6, doi: 10.1109/HORA49412.2020.9152602.

[43] X.-R. Ding, B. P. Yan, Y.-T. Zhang, J. Liu, P. Su, and N. Zhao, "Feature Exploration for Knowledge-guided and Data-driven Approach Based Cuffless Blood Pressure Measurement," arXiv, pp. 1-4.

[44] S. Maqsood, S. Xu, M. Springer and R. Mohawesh, "A Benchmark Study of Machine Learning for Analysis of Signal Feature Extraction Techniques for Blood Pressure Estimation Using Photoplethysmography (PPG)," in *IEEE*

Access, vol. 9, pp. 138817-138833, 2021, doi: 10.1109/ACCESS.2021.3117969.

[45] M. Liu, L. M. Po, and H. Fu, "Cuffless blood pressure estimation based on photoplethysmography signal and its second derivative," *International Journal of Computer Theory and Engineering*, vol. 9, no. 3, p. 202, 2017.

[46] M. H. Chowdhury et al., "Estimating Blood Pressure from the Photoplethysmogram Signal and Demographic Features Using Machine Learning Techniques," *Sensors*, vol. 20, no. 11, p. 3127, Jun. 2020. doi: 10.3390/s20113127.

[47] H. Wang, "Random Forest Based Blood Pressure Prediction Model from ECG And PPG Signal," Jan. 2022. doi: 10.1145/3510427.3510428.

[48] M. M. R. Khan Mamun and A. T. Alouani, "Cuffless Blood Pressure Measurement Using Linear and Nonlinear Optimized Feature Selection," *Diagnostics*, vol. 12, no. 2, p. 408, Feb. 2022. doi: 10.3390/diagnostics12020408.

[49] Y. Zhang and Z. Wang, "A hybrid model for blood pressure prediction from a PPG signal based on MIV and GA-BP neural network," in *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pp. 1989–1993, Jul. 2017. doi: 10.1109/FSKD.2017.8393073.

[50] X. Tan, Z. Ji, and Y. Zhang, "Non-invasive continuous blood pressure measurement based on mean impact value method, BP neural network, and genetic algorithm," *Technology and Health Care*, vol. 26, pp. 87–101, May 2018. doi: 10.3233/thc-174568.

[51] A. Attarpour, A. Mahnam, A. Aminitabar, and H. Samani, "Cuff-less continuous measurement of blood pressure using wrist and fingertip photo-

plethysmograms: Evaluation and feature analysis," *Biomedical Signal Processing and Control*, vol. 49, pp. 212–220, Mar. 2019. doi: 10.1016/j.bspc.2018.12.006.

[52] A. Dagamseh, Q. Qananwah, H. Al Quran, and K. Shaker Ibrahim, "Towards a portable-noninvasive blood pressure monitoring system utilizing the photoplethysmogram signal," *Biomedical Optics Express*, vol. 12, no. 12, p. 7732, Nov. 2021. doi: 10.1364/boe.444535.

[53] V. Fleischhauer, A. Feldheiser, and S. Zaunseder, "Beat-to-Beat Blood Pressure Estimation by Photoplethysmography and Its Interpretation," *Sensors*, vol. 22, no. 18, p. 7037, Sep. 2022. doi: 10.3390/s22187037.

[54] P. -H. Chiang, M. Wong and S. Dey, "Using Wearables and Machine Learning to Enable Personalized Lifestyle Recommendations to Improve Blood Pressure," in *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 9, pp. 1-13, 2021, Art no. 2700513, doi: 10.1109/JTEHM.2021.3098173.

[55] F. P. . -W. Lo, C. X. . -T. Li, J. Wang, J. Cheng and M. Q. . -H. Meng, "Continuous systolic and diastolic blood pressure estimation utilizing long short-term memory network," *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Jeju, Korea (South), 2017, pp. 1853-1856, doi: 10.1109/EMBC.2017.8037207.

[56] D. Lee et al., "Beat-to-Beat Continuous Blood Pressure Estimation Using Bidirectional Long Short-Term Memory Network," *Sensors*, vol. 21, no. 1, p. 96, Dec. 2020. doi: 10.3390/s21010096.

[57] Y.-H. Li, L. N. Harfiya, K. Purwandari, and Y.-D. Lin, "Real-Time Cuffless Continuous Blood Pressure Estimation Using Deep Learning Model," *Sensors*, vol. 20, no. 19, p. 5606, Sep. 2020. doi: 10.3390/s20195606.

[58] L. N. Harfiya, C.-C. Chang, and Y.-H. Li, "Continuous Blood Pressure Estimation Using Exclusively Photopletysmography by LSTM-Based Signal-to-Signal Translation," *Sensors*, vol. 21, no. 9, p. 2952, Apr. 2021. doi: 10.3390/s21092952.

[59] J. Esmaelpoor, M. H. Moradi, and A. Kadkhodamohammadi, "A multistage deep neural network model for blood pressure estimation using photoplethysmogram signals," *Computers in Biology and Medicine*, vol. 120, p. 103719, May 2020. doi: 10.1016/j.compbiomed.2020.103719.

[60] M. Panwar, A. Gautam, D. Biswas and A. Acharyya, "PP-Net: A Deep Learning Framework for PPG-Based Blood Pressure and Heart Rate Estimation," in *IEEE Sensors Journal*, vol. 20, no. 17, pp. 10000-10011, 1 Sept.1, 2020, doi: 10.1109/JSEN.2020.2990864.

[61] S. Baker, W. Xiang, and I. Atkinson, "A hybrid neural network for continuous and non-invasive estimation of blood pressure from raw electrocardiogram and photoplethysmogram waveforms," *Computer Methods and Programs in Biomedicine*, vol. 207, p. 106191, Aug. 2021. doi: 10.1016/j.cmpb.2021.106191.

[62] D. U. Jeong and K. M. Lim, "Combined deep CNN–LSTM network-based multitasking learning architecture for noninvasive continuous blood pressure estimation using difference in ECG-PPG features," *Scientific Reports*, vol. 11, no. 1, Jun. 2021. doi: 10.1038/s41598-021-92997-0.

[63] S. Rastegar, H. Gholam Hosseini, and A. Lowe, "Hybrid CNN-SVR Blood Pressure Estimation Model Using ECG and PPG Signals," *Sensors*, vol. 23, no. 3, p. 1259, Jan. 2023. doi: 10.3390/s23031259.

[64] Y. Li et al., "Random forest regression for online capacity estimation of lithium-ion batteries," *Applied Energy*, vol. 232, pp. 197–210, Dec. 2018. doi: 10.1016/j.apenergy.2018.09.182.

[65] A. Tiloca, G. Pagana and D. Demarchi, "A Random Tree Based Algorithm for Blood Pressure Estimation," *2020 IEEE MTT-S International Microwave Biomedical Conference (IMBioC)*, Toulouse, France, 2020, pp. 1-4, doi: 10.1109/IMBIoC47321.2020.9385038.

[66] S. M. Fati, A. Muneer, N. A. Akbar, and S. M. Taib, "A Continuous Cuffless Blood Pressure Estimation Using Tree-Based Pipeline Optimization Tool," *Symmetry*, vol. 13, no. 4, p. 686, Apr. 2021. doi: 10.3390/sym13040686.

[67] X. Chen, S. Yu, Y. Zhang, F. Chu and B. Sun, "Machine Learning Method for Continuous Noninvasive Blood Pressure Detection Based on Random Forest," in *IEEE Access*, vol. 9, pp. 34112-34118, 2021, doi: 10.1109/ACCESS.2021.3062033.

[68] W. Wang, P. Mohseni, K. L. Kilgore, and L. Najafizadeh, "PulseDB: A large, cleaned dataset based on MIMIC-III and VitalDB for benchmarking cuff-less blood pressure estimation methods," Frontiers in Digital Health, vol. 4, Feb. 2023, doi: 10.3389/fdgth.2022.1090854.

[69] S. Abdullah, A. Hafid, M. Folke, M. Lindén, and A. Kristoffersson, "A Novel Fiducial Point Extraction Algorithm to Detect C and D Points from the Acceleration Photoplethysmogram (CnD)," *Electronics*, vol. 12, no. 5, p. 1174, Jan. 2023. doi: 10.3390/electronics12051174.

[70] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Jul. 2019. doi: 10.1145/3292500.3330701.

[71] Association for the Advancement Instrumentation, American National Standard for Electronic or Automated Sphygmomanometers, ANSI/AAMI SP 10 2002, Arlington, VA: AAMI, 2002.

[72] E. O'Brien et al., "The British hypertension society protocol for the evaluation of blood pressure measuring devices," J. Hypertension, vol. 11, no. 2, pp. S43–S63, 1993.

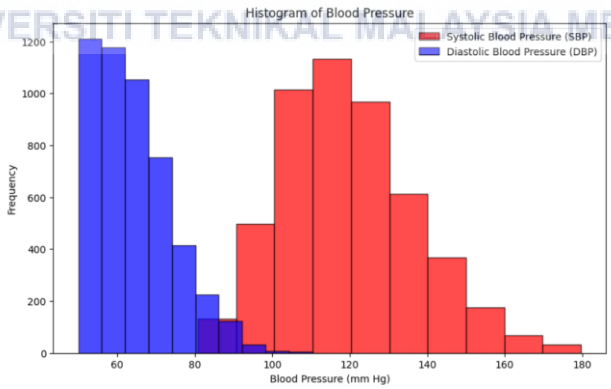
APPENDICES

APPENDIX A: GANTT CHART

Project Activities	Week														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
FYP 1 Briefing															
Chapter 1															
Chapter 2															
Chapter 3															
Early results															
FYP 1 Seminar															
Report writing															
Report Submission															

Project Activities	Week														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
FYP 2 Briefing															
Chapter 2															
Chapter 3															
Chapter 4															
Chapter 5															
Report writing															
Report Submission															
FYP 2 seminar															

APPENDIX B: BP DISTRIBUTION



BP Category	SBP and DBP range (mmHg)	Number of signal segments
Hypotension	SBP<90 and DBP<60	110
Normal	90≤SBP<120 and DBP<80	2621

Elevated	120≤SBP≤129 and DBP<80	855
Hypertension	SBP≥130 and/or DBP≥80	1414

APPENDIX C:HIGHLY CORRELATED FEATURES

SBP		DBP	
PAT maxslope	**PAT peak**	PAT maxslope	**PAT peak**
Heart rate	$\boldsymbol{t_{pp}}$	Heart rate	$\boldsymbol{t_{pp}}$
C_{slope}	$\boldsymbol{T_{OS}}$	C_{slope}	$\boldsymbol{T_{OS}}$
DW10	$Width_{10}$	**DW10**	$Width_{10}$
DW25	$\boldsymbol{Width_{25}}$	DW25	$\boldsymbol{Width_{25}}$
DW33	$Width_{33}$	**DW33**	$Width_{33}$
DW50	$Width_{50}$	**DW50**	$Width_{50}$
$\boldsymbol{Width_{50}}$	$Width_{66}$	$\boldsymbol{Width_{50}}$	$Width_{66}$
DW50/SW50	DW66/SW66	DW50/SW50	**DW66/SW66**
DW66	$Width_{66}$	**DW66**	$Width_{66}$
$\boldsymbol{Width_{66}}$	$Width_{75}$	$Width_{66}$	$\boldsymbol{Width_{75}}$
DW66/SW66	DW75/SW75	**DW75**	$Width_{66}$
DW75	$Width_{75}$	BMI/t_1	$\boldsymbol{Weight/t_1}$
$\boldsymbol{BMI/t_1}$	$Weight/t_1$	$\boldsymbol{Weight/t_{pi}}$	$Weight/t_{pp}$
$Weight/t_{pi}$	$\boldsymbol{Weight/t_{pp}}$	P	***RP* diff**
P	***RP* diff**	RT	***ST***
RT	***ST***	PT	***RT***
PT	RT	PT	***QT***
RT ratio	***T***	RT ratio	***T***
QT	RT		

APPENDIX D: OPTIMIZED FEATURES COMBINATION FOR SBP

Optimized features combination for SBP					
1	$Angle_{ed}$	15	LASI	29	BMI
2	RT	16	$Weight/t_{pi}$	30	PR
3	e/a	17	$S2$	31	QT_c
4	PAT maxslope	18	Heart rate	32	DW10/SW10
5	d/a	19	$Width_{10}$	33	c/a
6	Kurtosis	20	PAT onset	34	PQ

7	$Width_{75}$	21	RT ratio	35	$S3$
8	T_{ND}	22	T_{wc}	36	DW33/SW33
9	Age	23	$Width_{33}$	37	P
10	BMI/t_{pp}	24	Height	38	Weight/t_1
11	b/a	25	DW75/SW75	39	AI
12	DT	26	DW25/SW25	40	Hjorth mobility
13	$Angle_{zy}$	27	Weight		
14	DW25	28	$A_{b-c-d-e/a}$		

APPENDIX E: OPTIMIZED FEATURES COMBINATION FOR DBP

Optimized features combination for DBP					
1	PT	15	Weight	29	$Width_{33}$
2	LASI	16	Heart rate	30	Hjorth complexity
3	Kurtosis	17	Height	31	Hjorth mobility
4	Weight/t_{pp}	18	DT	32	PPG_K value
5	AI	19	$Angle_{zy}$	33	$Angle_{ND}$
6	b/a	20	DW25/SW25	34	$S1$
7	d/a	21	BMI	35	PQ
8	DW25	22	T_{Oa}	36	α_n
9	Age	23	BMI/t_1	37	PR
10	T_{ND}	24	$Width_{66}$	38	DW50/SW50
11	$Angle_{ed}$	25	T_{wc}	39	P
12	BMI/t_{pp}	26	$Width_{10}$		
13	QT_c	27	DW33/SW33		
14	e/a	28	PAT onset		

APPENDIX F: FEATURE EXTRATION CODE

```
% load('VitalDB_Train_Subset.mat', 'Subset')
clc,close all
data=struct();num=1;old_subject='p000001_1';no_subject=1;z=0;
featureNames = {'sbp', 'dbp', 'patpeak', 'patmaxslope', 'patonset', 'k_value', 'hr','tos', 'lasi', 'pir', 'ai','s1','s2','s3','s4','ipa', 'cslope', 'toa',...
  'ratioba','kurtosis_ppg','mobility','complexity', 'angleND', 'angleSD','angleZY', 'angleED','a_bcde','tnd','twc','widthDT_10','width_10','widthratio_10',...
  'widthDT_25','width_25','widthratio_25','widthDT_33','width_33','widthratio_33','widthDT_50','width_50','widthratio_50','widthDT_66','width_66',...
  'widthratio_66','widthDT_75','width_75','widthratio_75','ratioca','ratioda','ratioea','dt','tpp','age', 'height', 'weight','bmi','gender','bmit1',...
  'weighttpi','weighttpp','weigtt1','bmitpp','p_value','q_value','r_value','s_value','t_value','pr','rt','pq','st','pt','rt_ratio','rp_diff',...
  'qt','qrs','qtc','sdi','sdiN','alphaN','subject_ID','no_subject'};

featureTypes = {'double', 'double', 'double', 'double', 'double', 'double', 'double', 'double','double', 'double', 'double',
'double','double','double','double','double','double',...
  'double','double','double','double', 'double', 'double','double', 'double','double','double','double','double','double','double',...
  'double','double','double','double','double','double','double','double','double','double','double','double', 'double',...
  'double','double','double','double','double','double','double','double', 'double', 'double','double','double',...
  'double','double','double','double','double','double','double','double','double','double','double','double','double','double','double',...
  'double','double','double','double','double','double','string','double'};

dataTable = table('Size', [0, numel(featureNames)], 'VariableNames', featureNames, 'VariableTypes', featureTypes);
for auto=1:600 %600 to ensure that the data can reach 500 after rejected some signal segments
    patient = [];
    patient_ID = [];newRow_append = {};
    index_1=(z*360)+1;
    index_2=((z+1)*360);
    z=z+1;

for index=index_1:index_2
index
try
fs=125;
abp =squeeze(Subset.Signals(index, 3, :)).';
normalized_ppg =squeeze(Subset.Signals(index, 2, :)).';
normalized_ecg =squeeze(Subset.Signals(index, 1, :)).';
age=(Subset.Age(index, 1));
gender=(Subset.Gender{index, 1});
if gender=='M'
   gender=1;
else
   gender=0;
end
height=(Subset.Height(index, 1));
weight=(Subset.Weight(index, 1));
bmi=(Subset.BMI(index, 1));
subject_ID=(Subset.Subject{index, 1});

%detect r
[rpeakVal,rpeakLoc]=findpeaks(normalized_ecg(1,:),'MinPeakHeight',0.8*max(normalized_ecg),'MinPeakDistance',55);

inverted_ecg = max(normalized_ecg)-normalized_ecg;

%detect q
qLoc=[];qVal=[];q_Loc=[];q_Val=[];

for j = 1:length(rpeakLoc)
      if j>1%after the first R peak
      [qVal, qLoc] = findpeaks(inverted_ecg(rpeakLoc(j-1):rpeakLoc(j)));%find peak of inverse ecg within the first and second
      qLoc=qLoc+rpeakLoc(j-1)-1;%correct back the position
      qVal_threshold=qVal(end);
      q_Loc(j-1)=qLoc(end);
      q_Val(j-1)=qVal(end);
      for y=1:length(qLoc)
         if qLoc(y)> qLoc(end)-10
            if qVal(y)>qVal_threshold
               q_Loc(j-1)=qLoc(y);
               q_Val(j-1)=qVal(y);
               qVal_threshold=qVal(y);
            end
         end
      end
      end
end
```

```
newqLoc=q_Loc;newqVal=normalized_ecg(q_Loc);

%detect s
sLoc=[];sVal=[];s_Loc=[];s_Val=[];

for j = 1:length(rpeakLoc)
    if j>1 && j<length(rpeakLoc) %make sure within the first and last r peak
    [sVal, sLoc] = findpeaks(inverted_ecg(rpeakLoc(j):rpeakLoc(j+1))); %find the peak between current and next rpeak
    sLoc=sLoc+rpeakLoc(j)-1;
    sVal_threshold=sVal(1);
    s_Loc(j-1)=sLoc(1);
    s_Val(j-1)=sVal(1);
    for y=1:length(sLoc)
        if sLoc(y)< sLoc(1)+20
            if sVal(y)>sVal_threshold
                s_Loc(j-1)=sLoc(y);
                s_Val(j-1)=sVal(y);
                sVal_threshold=sVal(y);
            end
        end
    end
    end
end

newsLoc=s_Loc;newsVal=normalized_ecg(s_Loc);

%detect p
pLoc=[];pVal=[];p_Loc=[];p_Val=[];

for j = 1:length(rpeakLoc)
    if j>1
    [pVal, pLoc] = findpeaks(normalized_ecg(rpeakLoc(j-1):q_Loc(j-1)-3));
    pLoc=pLoc+rpeakLoc(j-1)-1;
    pVal_threshold=pVal(end);
    p_Loc(j-1)=pLoc(end);
    p_Val(j-1)=pVal(end);
    for y=1:length(pLoc)
        if pLoc(y)> pLoc(end)-20
            if pVal(y)>pVal_threshold
                p_Loc(j-1)=pLoc(y);
                p_Val(j-1)=pVal(y);
                pVal_threshold=pVal(y);
            end
        end
    end
    end
end

newpLoc=p_Loc;newpVal=normalized_ecg(p_Loc);

%detect t
tLoc=[];tVal=[];t_Loc=[];t_Val=[];

for j = 1:length(rpeakLoc)
    if j>1 && j<length(rpeakLoc)
    [tVal, tLoc] = findpeaks(normalized_ecg(newsLoc(j-1):newpLoc(j)));
    tLoc=tLoc+newsLoc(j-1)-1;
    tVal_threshold=tVal(1);
    t_Loc(j-1)=tLoc(1);
    t_Val(j-1)=tVal(1);
    for y=1:length(tLoc)
            if tVal(y)>tVal_threshold
                t_Loc(j-1)=tLoc(y);
                t_Val(j-1)=tVal(y);
                tVal_threshold=tVal(y);
            end
    end
    end
end

newtLoc=t_Loc;newtVal=normalized_ecg(t_Loc);

%detect t wave end
tLoc_end=[];tVal_end=[];t_Loc_end=[];t_Val_end=[];

for j=1:length(newtLoc)
    [tVal_end, tLoc_end] = findpeaks(inverted_ecg(newtLoc(j)+5:newpLoc(j+1)));%find peak of inverse ecg within the first and second
    tLoc_end=tLoc_end+newtLoc(j)+5-1;%correct back the position
    t_Loc_end(j)=tLoc_end(1);
    t_Val_end(j)=tVal_end(1);
end

newtLoc_end=t_Loc_end;newtVal_end=normalized_ecg(t_Loc_end);
```

```
%detect t wave end
tLoc_end=[];tVal_end=[];t_Loc_end=[];t_Val_end=[];

for j=1:length(newtLoc)
      [tVal_end, tLoc_end] = findpeaks(inverted_ecg(newtLoc(j)+5:newpLoc(j+1)));%find peak of inverse ecg within the first and second
      tLoc_end=tLoc_end+newtLoc(j)+5-1;%correct back the position
      t_Loc_end(j)=tLoc_end(1);
      t_Val_end(j)=tVal_end(1);
end

newtLoc_end=t_Loc_end;newtVal_end=normalized_ecg(t_Loc_end);

%detect the abp points
abppeaksVal=[];abppeaksLoc=[];abponsetsVal=[];abponsetsLoc=[];
abp_threshold_height=0.6*(max(abp(rpeakLoc(1):rpeakLoc(end)))-min(abp(rpeakLoc(1):rpeakLoc(end))))+min(abp(rpeakLoc(1):rpeakLoc(end)));
[abppeaksVal,abppeaksLoc]=findpeaks(abp(rpeakLoc(1):rpeakLoc(end)),'MinPeakHeight',abp_threshold_height,'MinPeakDistance',55);
abppeaksLoc=abppeaksLoc+rpeakLoc(1)-1;
inverse_abp=(max(abp)-abp);
inverse_abp_threshold_height=0.6*(max(inverse_abp(rpeakLoc(1):abppeaksLoc(end)))-
min(inverse_abp(rpeakLoc(1):abppeaksLoc(end))))+min(inverse_abp(rpeakLoc(1):abppeaksLoc(end)));
[abponsetsVal,abponsetsLoc]=findpeaks(inverse_abp(rpeakLoc(1):abppeaksLoc(end)),'MinPeakHeight',inverse_abp_threshold_height,'MinPeakDistance',5
5);
abponsetsLoc=abponsetsLoc+rpeakLoc(1)-1;
abponsetsVal=abp(abponsetsLoc);
%ensuring detect the correct onsets
for i=1:length(abppeaksLoc)-1
   onsets_Loc = (abponsetsLoc(abponsetsLoc > abppeaksLoc(i) & abponsetsLoc < abppeaksLoc(i+1)));
   if length(onsets_Loc)>1
      onsets_temp=find(ismember(abponsetsLoc,onsets_Loc));
      [~, idxOfMax] = max(abponsetsVal(onsets_temp));
      abponsetsLoc(onsets_temp(idxOfMax)) = [];
      abponsetsVal(onsets_temp(idxOfMax)) = [];
   end
end

ppgpeaksVal=[];ppgpeaksLoc=[];ppgonsetsVal=[];ppgonsetsLoc=[];
inverse_ppg=(max(normalized_ppg)-normalized_ppg);

[ppgonsetsVal,ppgonsetsLoc]=findpeaks(inverse_ppg(abponsetsLoc(1):rpeakLoc(end)),'MinPeakHeight',0.8*max(inverse_ppg),'MinPeakDistance',55);
ppgonsetsLoc=ppgonsetsLoc+abponsetsLoc(1)-1;
ppgonsetsVal=normalized_ppg(ppgonsetsLoc);

[ppgpeaksVal,ppgpeaksLoc]=findpeaks(normalized_ppg(abppeaksLoc(1):ppgonsetsLoc(end)),'MinPeakHeight',0.5*max(normalized_ppg(abponsetsLoc(1):
ppgonsetsLoc(end))),'MinPeakDistance',55);
ppgpeaksLoc=ppgpeaksLoc+abppeaksLoc(1)-1;
%%%
%plot graph
%%%
% figure
% subplot(3,1,1)
% plot(abp)
% hold on
% plot(abppeaksLoc,abppeaksVal,'r*', 'MarkerSize', 5)
% hold on
% plot(abponsetsLoc,abponsetsVal,'b*', 'MarkerSize', 5)
```

```
% title(['Index ' num2str(index) ' ID ' subject_ID ' sbp ' num2str(sbp) ' dbp ' num2str(dbp)]);
%
% subplot(3,1,2)
% plot(normalized_ppg)
% hold on
% plot(ppgpeaksLoc,ppgpeaksVal,'r*', 'MarkerSize', 5)
% plot(maxslopeLoc,maxslopeVal,'r*', 'MarkerSize', 5)
% plot(ppgonsetsLoc,ppgonsetsVal,'b*', 'MarkerSize', 5)
% hold off
%
% subplot(3,1,3)
% plot(normalized_ecg)
% hold on
% plot(rpeakLoc,rpeakVal, 'r*', 'MarkerSize', 5)
% plot(newsLoc,newsVal,'b*', 'MarkerSize', 5)
% plot(newqLoc,newqVal,'k*', 'MarkerSize', 5);
% hold off

sbp=mean(abppeaksVal);dbp=mean(abponsetsVal);

if sbp >= 80 && sbp <= 180 && dbp >=50 && dbp <= 120
%all the features
ppg_t=(0:length(normalized_ppg)-1) * (1/fs);
vpg=gradient(normalized_ppg,ppg_t);
apg=gradient(vpg,ppg_t);
jpg=gradient(apg,ppg_t);

[wVal,wLoc]=findpeaks(vpg(ppgonsetsLoc(1):ppgpeaksLoc(end)),'MinPeakHeight',0.5*max(vpg(ppgonsetsLoc(1):ppgpeaksLoc(end))),'MinPeakDistance',
55);
wLoc=wLoc+ppgonsetsLoc(1)-1;

maxslopeLoc=wLoc;
maxslopeVal=normalized_ppg(maxslopeLoc);

hr=[];tpp=[];k_value_array=[];tos_array=[]; lasi_array=[]; pir_array=[]; ai_array=[];s1_array=[];s2_array=[];s3_array=[];s4_array=[];...
ipa_array=[];
cslope_array=[];toa_array=[];kurtosis_ppg_array=[];angleND_array=[];angleSD_array=[];angleZY_array=[];angleED_array=[];a_bcde_array=[];...
tnd_array=[];twc_array=[];patpeak_array=[];patmaxslope_array=[];patonset_array=[];widthDT_10_array=[];width_10_array=[];widthratio_10_array=[];...
widthDT_25_array=[];width_25_array=[];widthratio_25_array=[];widthDT_33_array=[];width_33_array=[];widthratio_33_array=[];widthDT_50_array=[];
...
width_50_array=[];widthratio_50_array=[];widthDT_66_array=[];width_66_array=[];widthratio_66_array=[];widthDT_75_array=[];width_75_array=[];...
widthratio_75_array=[];ratioba_array=[];ratioca_array=[];ratioda_array=[];ratioea_array=[];dt_array=[];tpi_array=[];p_value_array =[];...
q_value_array=[];r_value_array=[];s_value_array =[];t_value_array=[];pr_array=[];rt_array=[];pq_array=[];st_array=[];pt_array=[];rt_ratio_array=[];...
rp_diff_array=[];qt_array=[];qrs_array=[];qtc_array=[];sdi_array=[];sdiN_array=[];il_array = [];gh_array=[];

%feature for ppg
for skr=1:(length(rpeakLoc)-2)%to ensure the ppg is within the first and last of ecg

    single_ppg_peak=ppgpeaksLoc(skr);
    ppg_onset_index_1= find(ppgonsetsLoc < single_ppg_peak);%find the onset before the current single ppg peak
    ppg_onset_index_2= find(ppgonsetsLoc > single_ppg_peak);%find the onset after the current single ppg peak
    single_ppg_onset_1=ppgonsetsLoc(ppg_onset_index_1(end));%the first onset is the nearest before r peak
    single_ppg_onset_2=ppgonsetsLoc(ppg_onset_index_2(1));%the second onset is the nearest after r peak
    single_ppg=normalized_ppg(single_ppg_onset_1:single_ppg_onset_2);%extract the single ppg

    single_vpg=vpg(single_ppg_onset_1:single_ppg_onset_2);%extract the single vpg
    single_apg=apg(single_ppg_onset_1:single_ppg_onset_2);%extract the single apg
    single_jpg=jpg(single_ppg_onset_1:single_ppg_onset_2);%extract the single jpg for c and d detection
    %move it to 1
    single_ppg_onset_1_newLoc=single_ppg_onset_1-(single_ppg_onset_1-1);
    single_ppg_onset_1_newVal=single_ppg(single_ppg_onset_1_newLoc);

    single_ppg_peak_newLoc=single_ppg_peak-(single_ppg_onset_1-1);
    single_ppg_peak_newVal=single_ppg(single_ppg_peak_newLoc);

    ppg_maxslope_index= find(maxslopeLoc < single_ppg_peak);
    single_ppg_max_newLoc=maxslopeLoc(ppg_maxslope_index(end))-(single_ppg_onset_1-1);%find the nearest maxslope to ppg peak and move it
    single_ppg_max_newVal=single_ppg(single_ppg_max_newLoc);

    single_ppg_onset_2_newLoc=single_ppg_onset_2-(single_ppg_onset_1-1);
    single_ppg_onset_2_newVal=single_ppg(single_ppg_onset_2_newLoc);

%vpg
vpgwLoc=0;vpgyLoc=0;vpgzLoc=0;
vpgwVal=0;vpgyVal=0;vpgzVal=0;
%initialize
[vpgwVal,vpgwLoc]=max(single_vpg(1:single_ppg_peak_newLoc));

[vpgyVal,vpgyLoc] = max(-single_vpg(1:round(0.75*(length(single_vpg)))));
vpgyVal=-vpgyVal;

[vpgzVal,vpgzLoc]= findpeaks(single_vpg(vpgyLoc:round(0.85*(length(single_vpg)))));
vpgzLoc = vpgzLoc+vpgyLoc-1;

if numel(vpgzLoc) >= 2 || isempty(vpgzLoc)
```

```
[vpgzVal,vpgzLoc]= max(single_vpg(vpgyLoc:round(0.85*(length(single_vpg)))));
vpgzLoc = vpgzLoc+vpgyLoc-1;
end


apgaLoc=0;apgbLoc=0;apgcLoc=0;apgdLoc=0;apgeLoc=0;apgfLoc=0;apgaVal=0;apgbVal=0;apgcVal=0;apgdVal=0;apgeVal=0;apgfVal=0;jpgminLoc=0;
%finding a and b
[apgaVal,apgaLoc] = max(single_apg(1:single_ppg_peak_newLoc));
[apgbVal,apgbLoc]= max(-single_apg(apgaLoc:single_ppg_peak_newLoc+10));
apgbVal=-apgbVal;
apgbLoc = apgbLoc + apgaLoc - 1;

%finding e and f
[apgeVal,apgeLoc]=findpeaks(single_apg(apgbLoc:vpgzLoc));
apgeLoc = apgeLoc + apgbLoc - 1;

if numel(apgeLoc) >= 2 || isempty(apgeLoc)
[emaxVal,emaxLoc]= max(apgeVal);
check = apgeLoc(emaxLoc);

if length(check:round(0.85*(length(single_apg))))<3 %To ensure that the detected e not at the end of apg
      apgeVal(emaxLoc) = []; % Exclude the current maximum
      [emaxVal, emaxLoc] = max(apgeVal);
end
      apgeLoc = apgeLoc(emaxLoc);
      apgeVal = emaxVal;
end

[apgfVal,apgfLoc]=findpeaks(-single_apg(apgeLoc:round(0.85*(length(single_apg)))));
apgfLoc = apgfLoc + apgeLoc -1;
apgfVal=-apgfVal;
if numel(apgfLoc) >= 2 || isempty(apgfLoc)

[apgfVal,apgfLoc]= min((single_apg(apgeLoc:round(0.85*(length(single_apg))))));
apgfLoc = apgfLoc + apgeLoc - 1;
end

%finding c and d
[apgdVal,apgdLoc]=findpeaks(-single_apg(apgbLoc:apgeLoc));
apgdLoc = apgdLoc + apgbLoc - 1;
apgdVal = -apgdVal;

if numel(apgdVal) >= 2 %if more than 2 d detected, take the lowest
   [apgdVal,dloc] = min(apgdVal);
   apgdLoc=apgdLoc(dloc);
end

if ~isempty(apgdLoc) %if d is detected in previous
[apgcVal,apgcLoc]=findpeaks(single_apg(apgbLoc:apgdLoc+1)); %c between b and d
apgcLoc = apgcLoc + apgbLoc - 1;
if numel(apgcVal) >= 2 %if more than 2 d detected, take the highest
   [apgcVal,cLoc] = max(apgcVal);
   apgcLoc=apgcLoc(cLoc);
end
end

%using cnd algorithm
[jpgmaxVal,jpgmaxLoc]=max(single_jpg(apgbLoc:apgeLoc));
jpgmaxLoc=jpgmaxLoc+apgbLoc-1;
jpgthreshold=jpgmaxLoc*0.4;

if (apgeLoc-jpgmaxLoc)<10 %set the min peak distance only if distance between e and jpgpeak>10
[jpgpeakVal,jpgpeakLoc]=findpeaks(single_jpg(apgbLoc:apgeLoc),'MinPeakHeight',jpgthreshold);
else
[jpgpeakVal,jpgpeakLoc]=findpeaks(single_jpg(apgbLoc:apgeLoc),'MinPeakDistance',10,'MinPeakHeight',jpgthreshold);
end
jpgpeakLoc = jpgpeakLoc + apgbLoc - 1;
apgzero_crossing = find(single_apg(apgbLoc:apgeLoc) >= 0, 1, 'first');
apgzero_crossing=apgzero_crossing+apgbLoc-1;

if isempty(apgcLoc) || isempty(apgdLoc)
   if numel(jpgpeakLoc)>1
      [jpgminVal,jpgminLoc]=min(single_jpg(jpgpeakLoc(1):jpgpeakLoc(2)));
      jpgminLoc = jpgminLoc + jpgpeakLoc(1) - 1;
      apgcLoc=round(jpgminLoc-0.025*(length(single_apg)));
      apgdLoc=round(jpgminLoc+0.025*(length(single_apg)));
   elseif numel(jpgpeakLoc)==1
      apgcLoc=jpgpeakLoc;
      apgdLoc=apgzero_crossing;
   end
   apgcVal=single_apg(round(apgcLoc));
   apgdVal=single_apg(round(apgdLoc));
end

%using e and f to find diastolic and diacrotic of PPG
```

```
single_ppgnLoc=apgeLoc;
single_ppgnVal=single_ppg(single_ppgnLoc);
single_ppgdLoc=apgfLoc;
single_ppgdVal=single_ppg(single_ppgdLoc);

%%%%
%plot graph
%%%%
%    figure
%    subplot(3,1,1)
%    plot(single_ppg)
%    hold on
%    plot(single_ppg_peak_newLoc,single_ppg_peak_newVal,'r*', 'MarkerSize', 5)
%    plot(single_ppg_onset_1_newLoc,single_ppg_onset_1_newVal,'b*', 'MarkerSize', 5)
%    plot(single_ppg_onset_2_newLoc,single_ppg_onset_2_newVal,'b*', 'MarkerSize', 5)
%    plot(single_ppg_max_newLoc,single_ppg_max_newVal,'b*', 'MarkerSize', 5)
%    plot(single_ppgnLoc,single_ppgnVal,'r*', 'MarkerSize', 5)
%    plot(single_ppgdLoc,single_ppgdVal,'r*', 'MarkerSize', 5)
%    hold off
%
%    subplot(3,1,2)
%
%    plot(single_vpg)
%    hold on
%    plot(vpgwLoc,vpgwVal,'r*', 'MarkerSize', 5)
%    plot(vpgyLoc,vpgyVal,'b*', 'MarkerSize', 5)
%    plot(vpgzLoc,vpgzVal,'k*', 'MarkerSize', 5)
%    hold off
%    subplot(3,1,3)
%    plot(single_apg)
%    hold on
% plot(apgaLoc,apgaVal,'r*', 'MarkerSize', 5)
% text(apgaLoc,apgaVal,'? a', 'FontSize', 13);
% plot(apgbLoc, apgbVal,'b*', 'MarkerSize', 5)
% text(apgbLoc, apgbVal,'? b', 'FontSize', 13);
% plot(apgcLoc,apgcVal,'m*', 'MarkerSize', 5)
% text(apgcLoc,apgcVal,'? c', 'FontSize', 13);
% plot(apgdLoc,apgdVal,'g*', 'MarkerSize', 5)
% text(apgdLoc,apgdVal,'? d', 'FontSize', 13);
% plot(apgeLoc, apgeVal,'r*', 'MarkerSize', 5)
% text(apgeLoc, apgeVal,'? e', 'FontSize', 13);
% plot(apgfLoc,apgfVal,'b*', 'MarkerSize', 5)
% text(apgfLoc,apgfVal,'? f', 'FontSize', 13);
% hold off

%K value
k_value_t=(0:length(single_ppg)-1) * (1/fs);
pm=trapz(k_value_t,single_ppg)/(1/fs);
k_value=(pm-single_ppg_onset_1_newVal)/(single_ppg_peak_newVal-single_ppg_onset_1_newVal);
k_value_array=[k_value_array,k_value];
%Tos
tos=(single_ppg_peak_newLoc-1)./fs;
tos_array=[tos_array,tos];
%LASI
lasi=(single_ppgdLoc-single_ppg_peak_newLoc)./fs;
lasi_array=[lasi_array,lasi];
%PIR
if single_ppg_onset_1_newVal<single_ppg_onset_2_newVal
   pir=single_ppg_peak_newVal/single_ppg_onset_1_newVal;
else
   pir=single_ppg_peak_newVal/single_ppg_onset_2_newVal;
end
if pir==Inf
   pir=0;
end
pir_array=[pir_array,pir];
%AI
ai=single_ppgdVal/single_ppg_peak_newVal;
ai_array=[ai_array,ai];
%area under the curve
s1_t=(0:length(single_ppg(1:vpgwLoc))-1) * (1/fs);
s2_t=(0:length(single_ppg(vpgwLoc:single_ppg_peak_newLoc))-1) * (1/fs);
s3_t=(0:length(single_ppg(single_ppg_peak_newLoc:single_ppgdLoc))-1) * (1/fs);
s4_t=(0:length(single_ppg(single_ppgdLoc:end))-1) * (1/fs);
ipa_1=(0:length(single_ppg(single_ppgnLoc:end))-1) * (1/fs);
ipa_2=(0:length(single_ppg(1:single_ppgnLoc))-1) * (1/fs);

s1= trapz(s1_t,single_ppg(1:vpgwLoc));
s2= trapz(s2_t,single_ppg(vpgwLoc:single_ppg_peak_newLoc));
s3= trapz(s3_t,single_ppg(single_ppg_peak_newLoc:single_ppgdLoc));
s4= trapz(s4_t,single_ppg(single_ppgdLoc:end));
ipa=trapz(ipa_1,single_ppg(single_ppgnLoc:end))/trapz(ipa_2,single_ppg(1:single_ppgnLoc));

s1_array=[s1_array,s1];s2_array=[s2_array,s2];s3_array=[s3_array,s3];s4_array=[s4_array,s4];ipa_array=[ipa_array,ipa];
```

```
%Cslope
cslope=(single_ppg_peak_newVal-single_ppg_onset_1_newVal)/((single_ppg_peak_newLoc-1)/fs);
cslope_array=[cslope_array,cslope];
%TOa
toa=(apgaLoc-1)./fs;
toa_array=[toa_array,toa];
%Kurtosis
kurtosis_ppg=kurtosis(single_ppg);
kurtosis_ppg_array=[kurtosis_ppg_array,kurtosis_ppg];
%anglend
angleND=(single_ppgdVal-single_ppgnVal)/((single_ppgdLoc-single_ppgnLoc)/fs);
angleND_array=[angleND_array,angleND];
%anglesd
angleSD=(single_ppgdVal-single_ppg_peak_newVal)/((single_ppgdLoc-single_ppg_peak_newLoc)/fs);
angleSD_array=[angleSD_array,angleSD];
%anglezy
angleZY=(vpgyVal-vpgzVal)/((vpgyLoc-vpgzLoc)/fs);
angleZY_array=[angleZY_array,angleZY];
%angleed
angleED=(apgdVal-apgeVal)/((apgdLoc-apgeLoc)/fs);
angleED_array=[angleED_array,angleED];
%a_bcde
a_bcde=(apgbVal-apgcVal-apgdVal-apgeVal)./apgaVal;
a_bcde_array=[a_bcde_array,a_bcde];
%tnd
tnd=(single_ppgdLoc-single_ppgnLoc)/fs;
tnd_array=[tnd_array,tnd];
%twc
twc=(apgcLoc-vpgwLoc)/fs;
twc_array=[twc_array,twc];
%pat
%patpeak
patpeak=(single_ppg_peak-rpeakLoc(skr))/fs;
patpeak_array=[patpeak_array,patpeak];
%patmaxslope
patmaxslope=(wLoc(skr)-rpeakLoc(skr))/fs;
patmaxslope_array=[patmaxslope_array,patmaxslope];
%patonset
patonset=(ppgonsetsLoc(ppg_onset_index_1(end))-rpeakLoc(skr))/fs;
patonset_array=[patonset_array,patonset];
% width feature
thresholds_width_percent = [0.1, 0.25, 0.33, 0.5, 0.66, 0.75];
for i = 1:length(thresholds_width_percent)
    threshold_width = min(single_ppg) + thresholds_width_percent(i) * (max(single_ppg) - min(single_ppg));
    crossings = find(single_ppg >= threshold_width);
    widthST = (single_ppg_peak_newLoc - crossings(1)) / fs;
    widthDT = (crossings(end) - single_ppg_peak_newLoc) / fs;
    width = (crossings(end) - crossings(1)) / fs;
    widthratio = widthDT / widthST;

    % width 10,25,33,50,66,75
     switch thresholds_width_percent(i)
        case 0.10
            widthDT_10_array = [widthDT_10_array, widthDT];
            width_10_array = [width_10_array, width];
            widthratio_10_array = [widthratio_10_array, widthratio];
        case 0.25
            widthDT_25_array = [widthDT_25_array, widthDT];
            width_25_array = [width_25_array, width];
            widthratio_25_array = [widthratio_25_array, widthratio];
        case 0.33
            widthDT_33_array = [widthDT_33_array, widthDT];
            width_33_array = [width_33_array, width];
            widthratio_33_array = [widthratio_33_array, widthratio];
        case 0.5
            widthDT_50_array = [widthDT_50_array, widthDT];
            width_50_array = [width_50_array, width];
            widthratio_50_array = [widthratio_50_array, widthratio];
        case 0.66
            widthDT_66_array = [widthDT_66_array, widthDT];
            width_66_array = [width_66_array, width];
            widthratio_66_array = [widthratio_66_array, widthratio];
        case 0.75
            widthDT_75_array = [widthDT_75_array, widthDT];
            width_75_array = [width_75_array, width];
            widthratio_75_array = [widthratio_75_array, widthratio];
    end
end

%b/a
ratioba= apgbVal/apgaVal;
ratioba_array=[ratioba_array,ratioba];
%c/a
ratioca= apgcVal/apgaVal;
ratioca_array=[ratioca_array,ratioca];
```

```
%d/a
ratioda= apgdVal/apgaVal;
ratioda_array = [ratioda_array, ratioda];
%e/a
ratioea= apgeVal/apgaVal;
ratioea_array=[ratioea_array,ratioea];

%dt
dt=(single_ppg_onset_2_newLoc-single_ppg_peak_newLoc)./fs;
dt_array=[dt_array,dt];

%tpi for demographic features
tpi=(single_ppg_onset_2_newLoc-single_ppg_onset_1_newLoc)/fs;
tpi_array=[tpi_array,tpi];

% ecg feature
% Amplitude
%p value
p_value=newpVal(skr);
p_value_array=[p_value_array,p_value];
%q value
q_value=newqVal(skr);
q_value_array=[q_value_array,q_value];
%r value
r_value=rpeakVal(skr+1);
r_value_array=[r_value_array,r_value];
%s value
s_value=newsVal(skr);
s_value_array=[s_value_array,s_value];
%t value
t_value=newtVal(skr);
t_value_array=[t_value_array,t_value];

% % temporal feature
%pr
pr=(rpeakLoc(skr+1)-newpLoc(skr))/fs;
pr_array=[pr_array,pr];
%rt
rt=(newtLoc(skr)-rpeakLoc(skr+1))/fs;
rt_array=[rt_array,rt];
%pq
pq=(newqLoc(skr)-newpLoc(skr))/fs;
pq_array=[pq_array,pq];
%st
st=(newtLoc(skr)-newsLoc(skr))/fs;
st_array=[st_array,st];
%pt
pt=(newtLoc(skr)-newpLoc(skr))/fs;
pt_array=[pt_array,pt];
%rt_ratio
rt_ratio=newtVal(skr)/rpeakVal(skr+1);
rt_ratio_array=[rt_ratio_array,rt_ratio];

%rp_diff
rp_diff=rpeakVal(skr+1)-newpVal(skr);
rp_diff_array=[rp_diff_array,rp_diff];

% %qt
qt=(newtLoc_end(skr)-newqLoc(skr))/fs;
qt_array=[qt_array,qt];

% %tq
tq=(newqLoc(skr+1)-newtLoc_end(skr))/fs;
% %qrs
qrs= (newsLoc(skr)-newqLoc(skr))/fs;
qrs_array=[qrs_array,qrs];

%QTC
qtc= (qt)/sqrt((rpeakLoc(skr+2)-rpeakLoc(skr+1))/fs);
qtc_array=[qtc_array,qtc];

%SDI
sdi= qt/tq;
sdi_array=[sdi_array,sdi];

%SDIn
sdiN = qt/((rpeakLoc(skr+2)-rpeakLoc(skr+1))/fs);
sdiN_array=[sdiN_array,sdiN];

%for getting alpha N
%il
if single_ppg_onset_1_newVal<single_ppg_onset_2_newVal
   il=single_ppg_onset_1_newVal;
else
```

```
    il=single_ppg_onset_2_newVal;
end
il_array=[il_array,il];

%GH
gh=sqrt(((vpgwLoc-single_ppg_peak_newLoc)/fs)^2+(vpgwVal-single_vpg(single_ppg_peak_newLoc))^2);
gh_array=[gh_array,gh];

end

%heart rate
hr=60*fs./diff(rpeakLoc);
hr_new=nanmean(hr);
%Hjoth parameter
diffecg=0;mobility=0;dmobility=0;complexity=0;
ecg_new=normalized_ecg(rpeakLoc(1):rpeakLoc(end));
ecg_t=(0:length(ecg_new)-1) * (1/fs);
diffecg=gradient(ecg_new,ecg_t);
%signal mobility
mobility=sqrt(var(diffecg)/var(ecg_new));
%signal complexity
dmobility=sqrt(var(gradient(diffecg,ecg_t))/var(diffecg));
complexity=dmobility/mobility;
%alphaN
gh_new=nanmean(gh_array);
il_new=nanmean(il_array);
alphaN=il_new*sqrt((hr_new*1060)/(1/gh_new));

patpeak_new=nanmean(patpeak_array);patmaxslope_new=nanmean(patmaxslope_array); patonset_new=nanmean(patonset_array);
k_value_new=nanmean(k_value_array);tos_new=nanmean(tos_array); lasi_new=nanmean(lasi_array);
pir_new=nanmean(pir_array); ai_new=nanmean(ai_array);s1_new=nanmean(s1_array);s2_new=nanmean(s2_array);s3_new=nanmean(s3_array);
s4_new=nanmean(s4_array);ipa_new=nanmean(ipa_array); cslope_new=nanmean(cslope_array);
toa_new=nanmean(toa_array); ratioba_new=nanmean(ratioba_array);kurtosis_ppg_new=nanmean(kurtosis_ppg_array);
angleND_new=nanmean(angleND_array);angleSD_new=nanmean(angleSD_array);angleZY_new=nanmean(angleZY_array);
angleED_new=nanmean(angleED_array);a_bcde_new=nanmean(a_bcde_array);tnd_new=nanmean(tnd_array);twc_new=nanmean(twc_array);
widthDT_10_new=nanmean(widthDT_10_array);width_10_new=nanmean(width_10_array);widthratio_10_new=nanmean(widthratio_10_array);
widthDT_25_new=nanmean(widthDT_25_array);width_25_new=nanmean(width_25_array);widthratio_25_new=nanmean(widthratio_25_array);
widthDT_33_new=nanmean(widthDT_33_array);width_33_new=nanmean(width_33_array);widthratio_33_new=nanmean(widthratio_33_array);
widthDT_50_new=nanmean(widthDT_50_array);width_50_new=nanmean(width_50_array);widthratio_50_new=nanmean(widthratio_50_array);
widthDT_66_new=nanmean(widthDT_66_array);width_66_new=nanmean(width_66_array);widthratio_66_new=nanmean(widthratio_66_array);
widthDT_75_new=nanmean(widthDT_75_array);width_75_new=nanmean(width_75_array);widthratio_75_new=nanmean(widthratio_75_array);
ratioca_new=nanmean(ratioca_array);ratioda_new=nanmean(ratioda_array);ratioea_new=nanmean(ratioea_array);dt_new=nanmean(dt_array);
tpi_new=nanmean(tpi_array);p_value_new=nanmean(p_value_array);q_value_new=nanmean(q_value_array);
r_value_new=nanmean(r_value_array);s_value_new=nanmean(s_value_array);t_value_new=nanmean(t_value_array);pr_new=nanmean(pr_array);
rt_new=nanmean(rt_array);pq_new=nanmean(pq_array);st_new=nanmean(st_array);pt_new=nanmean(pt_array);rt_ratio_new=nanmean(rt_ratio_array);
rp_diff_new=nanmean(rp_diff_array);qt_new=nanmean(qt_array);qrs_new=nanmean(qrs_array);qtc_new=nanmean(qtc_array);
sdi_new=nanmean(sdi_array);sdiN_new=nanmean(sdiN_array);

%some demographic features
bmit1=bmi/tos_new;
tpp=nanmean(diff(ppgpeaksLoc)/fs);
weighttpi=weight/tpi_new;
weighttpp=weight/tpp;
weigtt1=weight/tos_new;
bmitpp=bmi/tpp;

deviations = abs(hr - hr_new);
outliers=0;
outliers = hr(deviations > 30);
%set predefined rules
if isempty(outliers)&& abs(length(rpeakLoc) - length(ppgpeaksLoc)) == 2 && length(ppgonsetsLoc)==length(ppgpeaksLoc)+1

newRow= {sbp, dbp, patpeak_new, patmaxslope_new, patonset_new, k_value_new, hr_new,tos_new, lasi_new, pir_new, ai_new,...
    s1_new,s2_new,s3_new,s4_new,ipa_new,cslope_new,toa_new,ratioba_new,kurtosis_ppg_new,mobility,complexity,angleND_new, angleSD_new,...
    angleZY_new, angleED_new,a_bcde_new,tnd_new,twc_new,widthDT_10_new,width_10_new,widthratio_10_new,widthDT_25_new,width_25_new,...
    widthratio_25_new,widthDT_33_new,width_33_new,widthratio_33_new,widthDT_50_new,width_50_new,widthratio_50_new,...
    widthDT_66_new,width_66_new,widthratio_66_new,widthDT_75_new,width_75_new,widthratio_75_new,ratioca_new,ratioda_new,ratioea_new,...
    dt_new,tpp,age, height, weight,bmi,gender,bmit1,weighttpi,weighttpp,weigtt1,bmitpp,...
    p_value_new,q_value_new,r_value_new,s_value_new,t_value_new,pr_new,rt_new,pq_new,st_new,pt_new,rt_ratio_new,rp_diff_new,...
    qt_new,qrs_new,qtc_new,sdi_new,sdiN_new,alphaN,subject_ID,no_subject};

if ~any(cellfun(@(x) any(isnan(x)), newRow))%exclude if NAN
newRow_append = [newRow_append; newRow];
patient = [patient, index];
patient_ID = [patient_ID, {subject_ID}];
if length(patient)==30
    subjectname=['p_', num2str(num)];
    subjectIDname=['p_', num2str(num),'_ID'];
    save([subjectname '.mat'], 'patient');
    save([subjectIDname '.mat'], 'patient_ID');
    num=num+1;
    dataTable = [dataTable; newRow_append];
    no_subject=no_subject+1;
    newRow_append = {};
    break;
```

```
end
end
end
end
   catch Me
   continue;
end
end
if no_subject>500 %when reach 500subject
   break
end
end

unique_subjects = unique(dataTable.no_subject); % Find unique subjects
selectedIndices_10 = [];selectedIndices_20 = [];selectedIndices_30 = [];

for i = 1:length(unique_subjects)
   subject = unique_subjects(i);
   subjectIndices = find(dataTable.no_subject == subject);
   selectedIndices_10 = [selectedIndices_10; subjectIndices(1:min(10, length(subjectIndices)))];
end

% Write the selected rows to CSV files
writetable(dataTable(selectedIndices_10, :), 'my_fyp_dataset_1.csv');
```

APPENDIX G: EXPERIMENT 1 CODE (SVR)

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
from sklearn.svm import SVR
from sklearn.preprocessing import StandardScaler
import optuna
from scipy.stats import pearsonr

file_path = 'D:/dataset/my_fyp_dataset_1.csv'
df = pd.read_csv(file_path)
df.dropna(inplace=True)
sbp = df['sbp'].values
dbp = df['dbp'].values
columns_to_drop = ['sbp', 'dbp', 'subject_ID', 'no_subject']
X = df.drop(columns_to_drop, axis=1).values
print(X.shape)


def optimize_model(X_train, X_val, y_train, y_val, objective, target):
    study = optuna.create_study(direction='minimize', sampler=optuna.samplers.TPESampler(seed=42))
    study.optimize(objective, n_trials=100)
    print(f'Best hyperparameters_{target}:', study.best_params)
    print(f'Best MAE_{target}:', study.best_value)

    best_params = study.best_params
    X_train_val = np.concatenate((X_train, X_val), axis=0)
    y_train_val = np.concatenate((y_train, y_val), axis=0)
    model = SVR(**best_params)
    model.fit(X_train_val, y_train_val)
    return model


def evaluate_model(model, X_test, y_test, target):
    y_test_pred = model.predict(X_test)
    mae_test = mean_absolute_error(y_test, y_test_pred)
    sd_test_mae = np.std(abs(y_test - y_test_pred), ddof=1)
    mse_test = mean_squared_error(y_test, y_test_pred)
    rmse_test = np.sqrt(mse_test)
    me_test = np.mean(y_test - y_test_pred)
    sd_test_me = np.std(y_test - y_test_pred, ddof=1)
    cumulative_percentage_error_five = np.sum(np.abs(y_test - y_test_pred) <= 5) / len(y_test) * 100
    cumulative_percentage_error_ten = np.sum(np.abs(y_test - y_test_pred) <= 10) / len(y_test) * 100
    cumulative_percentage_error_fifteen = np.sum(np.abs(y_test - y_test_pred) <= 15) / len(y_test) * 100

    print(f"For {target}")
    print("Mean Absolute Error and standard deviation (MAE +/- SD) on Test Set:", round(mae_test, 4), "+/-",
          round(sd_test_mae, 4))
    print("Mean Error and standard deviation (ME+/- SD) on Test Set:", round(me_test, 4), "+/-", round(sd_test_me, 4))
    print("Mean Squared Error (MSE) on Test Set:", round(mse_test, 4))
    print("Root mean squared error (RMSE) on Test Set:", round(rmse_test, 4))
    print("cumulative percentage error:","<=5:",round(cumulative_percentage_error_five,4),"%")
    print("cumulative percentage error:","<=10:",round(cumulative_percentage_error_ten,4),"%")
    print("cumulative percentage error:","<=15:",round(cumulative_percentage_error_fifteen,4),"%")
```

```
    difference = y_test - y_test_pred
    index = np.argmax(np.abs(difference))
    plt.scatter(y_test, y_test_pred, color='blue', label=f'Actual vs. Predicted {target}')
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--', color='red',
             label='Perfect Prediction')
    plt.scatter(y_test[index], y_test_pred[index], color='red', label='Max difference')
    plt.xlabel('Actual Values')
    plt.ylabel('Predicted Values')
    plt.title(f'Regression Plot: Actual vs. Predicted Values ({target})')
    plt.legend()
    plt.show()

    pearson_corr_coefficient, _ = pearsonr(y_test, y_test_pred)
    print("Pearson correlation coefficient:", pearson_corr_coefficient)


def objective(trial):
    params = {
        "kernel": 'rbf',
        "C": trial.suggest_float('C', 1, 100),
        "epsilon": trial.suggest_float('epsilon', 1e-2, 1),
        "gamma": trial.suggest_categorical('gamma', ['scale', 'auto'])
    }

    model = SVR(**params)
    model.fit(X_train, y_train)
    predictions = model.predict(X_val)
    mae = mean_absolute_error(y_val, predictions)
    return mae
X_train, X_temp, y_train, y_temp = train_test_split(X, sbp, test_size=0.40, shuffle=False)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, shuffle=False)

# Standardize the data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_val = scaler.transform(X_val)
X_test = scaler.transform(X_test)
print(X_train.shape)
print(X_val.shape)
print(X_test.shape)

# Train and evaluate model for SBP
model_sbp = optimize_model(X_train, X_val, y_train, y_val, objective, "SBP")
# Evaluate model for SBP
evaluate_model(model_sbp, X_test, y_test, "SBP")
X_train, X_temp, y_train, y_temp = train_test_split(X, dbp, test_size=0.40, shuffle=False)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, shuffle=False)

# Standardize the data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_val = scaler.transform(X_val)
X_test = scaler.transform(X_test)
print(X_train.shape)
print(X_val.shape)
print(X_test.shape)
# Train and evaluate model for DBP
model_dbp = optimize_model(X_train, X_val, y_train, y_val, objective, "DBP")
# Evaluate model for DBP
evaluate_model(model_dbp, X_test, y_test, "DBP")
```

APPENDIX H: EXPERIMENT 1 CODE (RANDOM FOREST)

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestRegressor
import optuna
from scipy.stats import pearsonr

file_path = 'D:/dataset/my_fyp_dataset_1.csv'
df = pd.read_csv(file_path)
df.dropna(inplace=True)
sbp = df['sbp'].values
dbp = df['dbp'].values
columns_to_drop = ['sbp', 'dbp', 'subject_ID', 'no_subject']
X = df.drop(columns_to_drop, axis=1).values
print(X.shape)



def optimize_model(X_train, X_val, y_train, y_val, objective, target):
    study = optuna.create_study(direction='minimize', sampler=optuna.samplers.TPESampler(seed=42))
    study.optimize(objective, n_trials=100)
    print('Best hyperparameters_randomforest:', study.best_params)
    print('Best MAE_randomforest:', study.best_value)

    best_params = study.best_params
    X_train_val = np.concatenate((X_train, X_val), axis=0)
    y_train_val = np.concatenate((y_train, y_val), axis=0)
    model = RandomForestRegressor(**best_params, random_state=42, n_jobs=-1)
    model.fit(X_train_val, y_train_val)
    return model


def evaluate_model(model, X_test, y_test, target):
    y_test_pred = model.predict(X_test)
    mae_test = mean_absolute_error(y_test, y_test_pred)
    sd_test_mae = np.std(abs(y_test - y_test_pred), ddof=1)
    mse_test = mean_squared_error(y_test, y_test_pred)
    rmse_test = np.sqrt(mse_test)
    me_test = np.mean(y_test - y_test_pred)
    sd_test_me = np.std(y_test - y_test_pred, ddof=1)
    cumulative_percentage_error_five = np.sum(np.abs(y_test - y_test_pred) <= 5) / len(y_test) * 100
    cumulative_percentage_error_ten = np.sum(np.abs(y_test - y_test_pred) <= 10) / len(y_test) * 100
    cumulative_percentage_error_fifteen = np.sum(np.abs(y_test - y_test_pred) <= 15) / len(y_test) * 100

    print(f"For {target}")
    print("Mean Absolute Error and standard deviation (MAE +/- SD) on Test Set:", round(mae_test, 4), "+/-",
          round(sd_test_mae, 4))
    print("Mean Error and standard deviation (ME+/- SD) on Test Set:", round(me_test, 4), "+/-", round(sd_test_me, 4))
    print("Mean Squared Error (MSE) on Test Set:", round(mse_test, 4))
    print("Root mean squared error (RMSE) on Test Set:", round(rmse_test, 4))
    print("cumulative percentage error:","<=5:",round(cumulative_percentage_error_five,4),"%")
    print("cumulative percentage error:","<=10:",round(cumulative_percentage_error_ten,4),"%")
    print("cumulative percentage error:","<=15:",round(cumulative_percentage_error_fifteen,4),"%")
    difference = y_test - y_test_pred
    index = np.argmax(np.abs(difference))
    plt.scatter(y_test, y_test_pred, color='blue', label=f'Actual vs. Predicted {target}')
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--', color='red',
             label='Perfect Prediction')
    plt.scatter(y_test[index], y_test_pred[index], color='red', label='Max difference')
    plt.xlabel('Actual Values')
    plt.ylabel('Predicted Values')
    plt.title(f'Regression Plot: Actual vs. Predicted Values ({target})')
    plt.legend()
    plt.show()

    pearson_corr_coefficient, _ = pearsonr(y_test, y_test_pred)
    print("Pearson correlation coefficient:", pearson_corr_coefficient)


def objective(trial):
    params = {
        "n_estimators": trial.suggest_categorical("n_estimators", [50, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000]),
        "max_depth": trial.suggest_int("max_depth", 10, 100),
        "min_samples_split": trial.suggest_int("min_samples_split", 2, 20),
        "min_samples_leaf": trial.suggest_int("min_samples_leaf", 1, 20),
        "max_features": trial.suggest_categorical("max_features", ["sqrt", "log2", None]),
    }

    model = RandomForestRegressor(**params, random_state=42)
    model.fit(X_train, y_train)
    predictions = model.predict(X_val)
    mae = mean_absolute_error(y_val, predictions)
    return mae
```

```
X_train, X_temp, y_train, y_temp = train_test_split(X, sbp, test_size=0.40, shuffle=False)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, shuffle=False)
print(X_train.shape)
print(X_val.shape)
print(X_test.shape)
# Train and evaluate model for SBP
model_sbp = optimize_model(X_train, X_val, y_train, y_val, objective, "SBP")
# Evaluate model for SBP
evaluate_model(model_sbp, X_test, y_test, "SBP")
```

```
X_train, X_temp, y_train, y_temp = train_test_split(X, dbp, test_size=0.40, shuffle=False)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, shuffle=False)
print(X_train.shape)
print(X_val.shape)
print(X_test.shape)
# Train and evaluate model for DBP
model_dbp = optimize_model(X_train, X_val, y_train, y_val, objective, "DBP")
# Evaluate model for DBP
evaluate_model(model_dbp, X_test, y_test, "DBP")
```

APPENDIX I: EXPERIMENT 2 CODE (FEATURE ANALYSIS)

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestRegressor
from scipy.stats import pearsonr

# Load and preprocess the dataset
file_path = 'D:/dataset/my_fyp_dataset_1.csv'
df = pd.read_csv(file_path)
df.dropna(inplace=True)

# Function to train and evaluate the model
def train_and_evaluate_model(target, columns_to_drop, model_params):
    X = df.drop(columns_to_drop, axis=1).values
    y = df[target].values

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, shuffle=False)

    model = RandomForestRegressor(**model_params)
    model.fit(X_train, y_train)

    y_test_pred = model.predict(X_test)

    mae_test = mean_absolute_error(y_test, y_test_pred)
    sd_test_mae = np.std(abs(y_test - y_test_pred), ddof=1)
    mse_test = mean_squared_error(y_test, y_test_pred)
    rmse_test = np.sqrt(mse_test)
    me_test = np.mean(y_test - y_test_pred)
    sd_test_me = np.std(y_test - y_test_pred, ddof=1)
    cumulative_percentage_error_five = np.sum(np.abs(y_test - y_test_pred) <= 5) / len(y_test) * 100
    cumulative_percentage_error_ten = np.sum(np.abs(y_test - y_test_pred) <= 10) / len(y_test) * 100
    cumulative_percentage_error_fifteen = np.sum(np.abs(y_test - y_test_pred) <= 15) / len(y_test) * 100

    print(f"For {target.upper()}")
    print("Mean Absolute Error and standard deviation (MAE +/- SD) on Test Set:", round(mae_test, 4), "+/-", round(sd_test_mae, 4))
    print("Mean Error and standard deviation (ME +/- SD) on Test Set:", round(me_test, 4), "+/-", round(sd_test_me, 4))
    print("Mean Squared Error (MSE) on Test Set:", round(mse_test, 4))
    print("Root mean squared error (RMSE) on Test Set:", round(rmse_test, 4))
    print("Cumulative percentage error:", "<=5:", round(cumulative_percentage_error_five, 4), "%")
    print("Cumulative percentage error:", "<=10:", round(cumulative_percentage_error_ten, 4), "%")
    print("Cumulative percentage error:", "<=15:", round(cumulative_percentage_error_fifteen, 4), "%")

    difference = y_test - y_test_pred
    index = np.argmax(np.abs(difference))
    plt.scatter(y_test, y_test_pred, color='blue', label=f'Actual vs. Predicted {target.upper()}')
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--', color='red', label='Perfect Prediction')
    plt.scatter(y_test[index], y_test_pred[index], color='red', label='Max difference')
    plt.xlabel('Actual Values')
    plt.ylabel('Predicted Values')
    plt.title(f'Regression Plot: Actual vs. Predicted Values ({target.upper()})')
    plt.legend()
    plt.show()
```

```
    pearson_corr_coefficient, _ = pearsonr(y_test, y_test_pred)
    print("Pearson correlation coefficient:", pearson_corr_coefficient)

# Define columns to drop for different cases
columns_to_drop_ppg_ecg = ['sbp', 'dbp', 'subject_ID', 'no_subject','age',
                           'height','weight','bmi','gender','bmit1','weighttpi','weighttpp','weigtt1','bmitpp']
columns_to_drop_ppg_demo = ['sbp', 'dbp', 'subject_ID', 'no_subject','hr','patpeak','patmaxslope','patonset','qrs','complexity','mobility',
                            'p_value','q_value',    'r_value','s_value','t_value','pr','rt','pq','st','pt',
                            'rt_ratio','rp_diff','qt','qrs','qtc','sdi','sdiN']
columns_to_drop_single_ppg = ['sbp', 'dbp', 'subject_ID', 'no_subject','age',
                              'height','weight','bmi','gender','bmit1','weighttpi','weighttpp','weigtt1','bmitpp',
                             'hr','patpeak','patmaxslope','patonset','qrs','complexity','mobility',
                             'p_value','q_value',  'r_value','s_value','t_value','pr','rt','pq','st','pt',
                             'rt_ratio','rp_diff','qt','qrs','qtc','sdi','sdiN']
# Define model parameters for sbp and dbp
model_params_sbp = {
    'n_estimators': 200,
    'max_depth': 22,
    'min_samples_split': 11,
    'min_samples_leaf': 7,
    'max_features': 'sqrt',
    'random_state': 42,
    'n_jobs': -1
}

model_params_dbp = {
    'n_estimators': 50,
    'max_depth': 61,
    'min_samples_split': 4,
    'min_samples_leaf': 3,
    'max_features': 'log2',
    'random_state': 42,
    'n_jobs': -1
}

# For SBP with ppg+ecg
train_and_evaluate_model('sbp', columns_to_drop_ppg_ecg, model_params_sbp)
# For DBP with ppg+ecg
train_and_evaluate_model('dbp', columns_to_drop_ppg_ecg, model_params_dbp)
# For SBP with ppg+demographic
train_and_evaluate_model('sbp', columns_to_drop_ppg_demo, model_params_sbp)
# For DBP with ppg+demographic
train_and_evaluate_model('dbp', columns_to_drop_ppg_demo, model_params_dbp)
# For SBP with single ppg cases
train_and_evaluate_model('sbp', columns_to_drop_single_ppg, model_params_sbp)
# For DBP with single ppg cases
train_and_evaluate_model('dbp', columns_to_drop_single_ppg, model_params_dbp)
```

APPENDIX J: EXPERIMENT 2 CODE (CORRELATION ANALYSIS)

```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.feature_selection import mutual_info_regression

# Function to calculate mutual information and find highly correlated features
def analyze_features(file_path, target_column):
    # Load the dataset into a pandas DataFrame
    df = pd.read_csv(file_path)
    df.dropna(inplace=True)

    # Separate the target variable from the features
    features = df.drop(columns=['sbp', 'dbp', 'subject_ID', 'no_subject'])
    target = df[target_column]

    # Calculate mutual information
    mi = mutual_info_regression(features, target, random_state=42)
    mi_series = pd.Series(mi, index=features.columns).sort_values(ascending=False)

    # Calculate correlation matrix
    corr_matrix = features.corr()

    # Check if a feature is already removed
    def already_removed(feature, features_to_remove):
        return feature in features_to_remove

    # Find pairs with high correlation and determine which to remove based on MI
    features_to_remove = []
    high_corr_pairs = {}

    for i in corr_matrix.columns:
        for j in corr_matrix.index[corr_matrix.index > i]:  # Ensure j>i to handle only upper triangle
            if abs(corr_matrix.at[i, j]) > 0.9 and not already_removed(i, features_to_remove) and not already_removed(j, features_to_remove):
                # Determine the feature with the lower mutual information score
                lower_mi_feature = i if mi_series[i] < mi_series[j] else j
                high_corr_pairs[f"{i} & {j}"] = (corr_matrix.at[i, j], lower_mi_feature)
                if lower_mi_feature not in features_to_remove:
                    features_to_remove.append(lower_mi_feature)

    # Print out the results
    print(f"High Correlation Pairs and Features with Lower MI for target '{target_column}':")
    for pair, (corr_value, lower_mi_feature) in high_corr_pairs.items():
        print(f"Highly correlated features: {pair} (Correlation: {corr_value:.2f})")
        print(f"Feature with lower mutual information score: {lower_mi_feature}\n")

    print("Features to remove:", features_to_remove)
    return features_to_remove
file_path = 'D:/dataset/my_fyp_dataset_1.csv'
# Analyze for SBP
features_to_remove_sbp = analyze_features(file_path, 'sbp')
# Analyze for DBP
features_to_remove_dbp = analyze_features(file_path, 'dbp')
```

APPENDIX K: EXPERIMENT 2 CODE (RANDOM FOREST WITH SHAP)

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestRegressor
import shap
from scipy.stats import pearsonr

def optimize_feature_select(file_path, target, columns_to_drop, model_parameters):
    # Load the dataset into a pandas DataFrame
    df = pd.read_csv(file_path)
    df.dropna(inplace=True)

    # Separate the target variable
    target_values = df[target].values

    # Prepare features
    X = df.drop(columns=columns_to_drop, axis=1)
    feature_names = X.columns
    X = X.values

    # Train-test split
    X_train, X_temp, y_train, y_temp = train_test_split(X, target_values, test_size=0.40, shuffle=False)
    X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, shuffle=False)

    # Model training
    model = RandomForestRegressor(**model_parameters)
    model.fit(X_train, y_train)

    # Validation predictions
    y_val_pred = model.predict(X_val)
    rmse_val = np.sqrt(mean_squared_error(y_val, y_val_pred))

    # SHAP summary plot
    explainer = shap.TreeExplainer(model)
    shap_value = explainer.shap_values(X_val)
    shap.summary_plot(shap_value, X_val, feature_names=feature_names, plot_type="bar")
    plt.suptitle('SHAP Summary', y=1.02, fontsize=16)

    # Feature importance
    important_feature = np.mean(np.abs(shap_value), axis=0)
    sorted_index = np.argsort(important_feature)[::-1]

    # Feature selection
    best_rmse = rmse_val
    best_features = feature_names
    rmse_total = [rmse_val]
    optimal_features = best_features[sorted_index]
    i = 0
    while len(sorted_index) > 1:
        i += 1
        sorted_index = sorted_index[:-1]
        model.fit(X_train[:, sorted_index], y_train)
        y_val_pred = model.predict(X_val[:, sorted_index])
        rmse = np.sqrt(mean_squared_error(y_val, y_val_pred))
        if rmse < best_rmse:
            best_rmse = rmse
            optimal_features = best_features[sorted_index]
        rmse_total.append(rmse)

    print(f"For {target.upper()}:")
    print("The number of features:", len(optimal_features))
    print("Best Features:", optimal_features)
    print("Best RMSE:", best_rmse)

    # Plot RMSE
    x_values = np.arange(1, len(rmse_total) + 1)
    rmse_reversed = rmse_total[::-1]
    plt.plot(x_values, rmse_reversed, marker='o', linestyle='-', color='b')
    plt.xlabel('Number of Features')
    plt.ylabel('RMSE Values (mmHg)')
    plt.title('RMSE values vs number of features')
    plt.grid(True)
    plt.show()

    # Test set evaluation
    X_test_selected = df[optimal_features].values
    X_train_selected, X_test_selected, y_train_selected, y_test_selected = train_test_split(X_test_selected, target_values, test_size=0.20, shuffle=Fals
    model.fit(X_train_selected, y_train_selected)
    y_test_pred = model.predict(X_test_selected)
    mae_test = mean_absolute_error(y_test, y_test_pred)
    sd_test_mae = np.std(abs(y_test - y_test_pred),ddof=1)
    mse_test = mean_squared_error(y_test, y_test_pred)
    rmse_test = np.sqrt(mse_test)
    #AAMI
    me_test = np.mean(y_test - y_test_pred)
    sd_test_me = np.std(y_test - y_test_pred,ddof=1)
```

```
    #BHS
    cumulative_percentage_error_five = np.sum(np.abs(y_test - y_test_pred) <= 5) / len(y_test) * 100
    cumulative_percentage_error_ten = np.sum(np.abs(y_test - y_test_pred) <= 10) / len(y_test) * 100
    cumulative_percentage_error_fifteen = np.sum(np.abs(y_test - y_test_pred) <= 15) / len(y_test) * 100
    print(f"For {target.upper()}:")
    print("Mean Absolute Error and standard deviation (MAE +/- SD) on Test Set:", round(mae_test,4) ,"+/-",round(sd_test_mae,4) )
    print("Mean Error and standard deviation (ME+/- SD) on Test Set:", round(me_test,4),"+/-",round(sd_test_me,4))
    print("Mean Squared Error (MSE) on Test Set:", round(mse_test,4))
    print("Root mean squared error (RMSE) on Test Set:", round(rmse_test,4))
    print("cumulative percentage error:","<=5:",round(cumulative_percentage_error_five,4),"%")
    print("cumulative percentage error:","<=10:",round(cumulative_percentage_error_ten,4),"%")
    print("cumulative percentage error:","<=15:",round(cumulative_percentage_error_fifteen,4),"%")

    difference = y_test - y_test_pred
    index = np.argmax(np.abs(difference))
    plt.scatter(y_test, y_test_pred, color='blue', label=f'Actual vs. Predicted {target}')
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--', color='red',
             label='Perfect Prediction')
    plt.scatter(y_test[index], y_test_pred[index], color='red', label='Max difference')
    plt.xlabel('Actual Values')
    plt.ylabel('Predicted Values')
    plt.title(f'Regression Plot: Actual(mmHg) vs. Predicted Values(mmHg) ({target.upper()})')
    plt.legend()
    plt.show()
    pearson_corr_coefficient, _ = pearsonr(y_test, y_test_pred)
    print("Pearson correlation coefficient:", pearson_corr_coefficient)

    # Bland-Altman plot
    means = np.mean([y_test_selected, y_test_pred], axis=0)
    differences = y_test_selected - y_test_pred
    mean_differences = np.mean(differences)
    std_differences = np.std(differences, ddof=1)
    # Calculate 95% limits of agreement
    upper_limit = mean_differences + 1.96 * std_differences
    lower_limit = mean_differences - 1.96 * std_differences

    # Calculate percentage within 95% limits
    within_limits = np.sum((differences >= lower_limit) & (differences <= upper_limit))
    total_data_points = len(y_test)
    percentage_within_95 = round((within_limits / total_data_points) * 100)
    plt.figure(figsize=(8, 6))
    plt.scatter(means, differences, color='blue', s=20)
    plt.axhline(mean_differences, color='red', linestyle='--', label=f'Mean Difference: {mean_differences:.2f}')
    plt.axhline(upper_limit, color='green', linestyle='--', label=f'Upper Limit of Agreement: {upper_limit:.2f}')
    plt.axhline(lower_limit, color='green', linestyle='--', label=f'Lower Limit of Agreement: {lower_limit:.2f}')
    plt.xlabel('Mean of Actual and Predicted Values (mmHg)')
    plt.ylabel('Difference between Actual and Predicted Values (mmHg)')
    plt.title(f'Bland-Altman Plot ({target.upper()})')
    plt.legend()
    plt.show()
    print(f"Percentage of data within 95% limits of agreement ({target.upper()}): {percentage_within_95}%")
    return optimal_features, rmse_test, mae_test

file_path = 'D:/dataset/my_fyp_dataset_1.csv'
# Parameters for SBP model
columns_to_drop_sbp = ['sbp', 'dbp','subject_ID','no_subject',
                       'patpeak', 'tpp', 'tos', 'widthDT_10',
                       'width_25', 'widthDT_33', 'widthDT_50', 'width_50',
                       'widthratio_50', 'widthDT_66', 'width_66', 'widthratio_66',
                       'widthDT_75', 'bmit1', 'weighttpp', 'rp_diff', 'st', 'pt', 't_value', 'qt']
model_parameters_sbp = {'n_estimators': 200, 'max_depth': 22, 'min_samples_split': 11, 'min_samples_leaf': 7,
                        'max_features': 'sqrt', 'random_state': 42, 'n_jobs': -1}
optimal_features_sbp, rmse_test_sbp, mae_test_sbp = optimize_feature_select(file_path, 'sbp', columns_to_drop_sbp, model_parameters_sbp)

# Parameters for DBP model
columns_to_drop_dbp = ['sbp', 'dbp','subject_ID','no_subject',
                       'patpeak', 'tpp', 'tos', 'widthDT_10',
                       'width_25', 'widthDT_33', 'widthDT_50', 'width_50',
                       'widthratio_66', 'widthDT_66', 'width_75', 'widthDT_75',
                       'weigtt1', 'weighttpi', 'rp_diff', 'st', 'rt', 'qt', 't_value']
model_parameters_dbp = {'n_estimators': 50, 'max_depth': 61, 'min_samples_split': 4, 'min_samples_leaf': 3,
                        'max_features': 'log2', 'random_state': 42, 'n_jobs': -1}
optimal_features_dbp, rmse_test_dbp, mae_test_dbp = optimize_feature_select(file_path, 'dbp', columns_to_drop_dbp, model_parameters_dbp)
```