

**DESIGN OF A PREDICTIVE MODEL FOR TCM PULSE DIAGNOSIS
IN MALAYSIA USING MACHINE LEARNING**

ONG JIA YING

**A project report submitted in partial fulfilment of the
requirements for the award of Bachelor of Engineering
(Hons.) Electrical and Electronic Engineering**

**Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman**

April 2020

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : 

Name : Ong Jia Ying

ID No. : 1503356

Date : 25th April 2020

APPROVAL FOR SUBMISSION

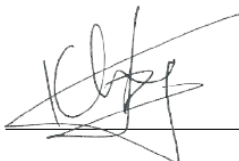
I certify that this project report entitled “**DESIGN OF A PREDICTIVE MODEL FOR TCM PULSE DIAGNOSIS IN MALAYSIA USING MACHINE LEARNING**” was prepared by **ONG JIA YING** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Hons.) Electrical and Electronic Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : 

Supervisor : Dr. Lai An Chow

Date : 25th April 2020

Signature : 

Co-Supervisor : Dr. Goh Yong Kheng

Date : 25th April 2020

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of Universiti Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2020, Ong Jia Ying. All right reserved.

ABSTRACT

Pulse diagnosis is one of the main diagnosis methods used on patients in Traditional Chinese medicine (TCM). TCM pulse is a time series signal which can be sensed using fingers by TCM practitioners in traditional way. In this project, the TCM pulse signal is collected using a pulse-taking system that consists of amplify spontaneous emissions (ASE), fibre Bragg grating analyser (FBGA) and fibre optic sensor (FBG).

In this project, Python is used to build the machine learning models to classify if a person is active in exercising or not through his/her TCM pulse. The machine learning algorithms applied in this project are k-nearest neighbors (KNN), naïve Bayes, random forest, gradient boosting and support vector machine (SVM).

People that active in exercising tends to have a slower pulse rate and higher pulse's height from left 'Cun' through the observation of the results in this project. SVM model has the best performance to classify the data set with 315 data samples.

TABLE OF CONTENTS

DECLARATION	ii
APPROVAL FOR SUBMISSION	iii
ABSTRACT	v
TABLE OF CONTENTS	vi
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF APPENDICES	xiv

CHAPTER

1	INTRODUCTION	1
	1.1 General Introduction	1
	1.2 Importance of the Study	4
	1.3 Problem Statement	4
	1.4 Aims and Objectives	5
	1.5 Limitation of the Study	5
2	LITERATURE REVIEW	6
	2.1 Introduction	6
	2.2 Artificial Neural Network (ANN)	7
	2.2.1 Activation Function	8
	2.2.2 Back propagation	9
	2.2.3 Related Work	10
	2.3 K-Nearest Neighbours (K-NN)	13
	2.3.1 Related Work	15
	2.4 Support Vector Machine (SVM)	16
	2.5 Decision Tree, Random Forest (RF), AdaBoost and Gradient Boosting	17

	2.5.1	Related Work	20
2.6		K-means	23
	2.6.1	Related Work	25
2.7		Fuzzy C-Means (FCM)	26
	2.7.1	Related Work	27
3		METHODOLOGY AND WORK PLAN	29
	3.1	Introduction	29
	3.2	Data Collection	30
	3.3	Data Pre-processing	35
		3.3.1 Filtering	36
		3.3.2 Remove Baseline Wander	38
		3.3.3 Cutting	39
		3.3.4 Labeling	41
		3.3.5 Normalizations	41
	3.4	Data Splitting	41
	3.5	Machine learning algorithms	42
	3.6	Evaluation	43
4		RESULTS AND DISCUSSION	46
	4.1	Introduction	46
	4.2	Data Analysis	47
	4.3	Machine Learning Models	50
		4.3.1 Results and Discussions based on 108 Data Samples	51
		4.3.2 Results and Discussions based on 315 Data Samples	53
	4.4	Compare both results between two data sets	54
	4.5	Possible Reasons for Low Accuracy of the Results	54
5		CONCLUSIONS AND RECOMMENDATIONS	56
	5.1	Conclusions	56
	5.2	Recommendations for future work	57

REFERENCES

58

APPENDICES

61

LIST OF TABLES

Table 1.1: The relationships between left inch/bar/cubit, right inch/bar/cubit and organs based on the Classic of Difficult Issues 《难经》 (朱文锋, 2013).	2
Table 2.1: Explanation of types of machine learning (Pant, n.d.).	6
Table 2.2: Types of distance functions and equations (K Nearest Neighbors - Classification, n.d.).	14
Table 4.1: Number of samples with and without deep layer pulse.	47
Table 4.2: Results of each machine learning models for 108 data samples.	51
Table 4.3: Results of each machine learning models for 315 data samples.	53

LIST OF FIGURES

Figure 1.1: 3 positions and 3 levels of pulse taking (Jennifer Dubowsky, 2012).	2
Figure 1.2: TCM Machine.	3
Figure 2.1: Differences between biological neurons and artificial neurons (Josh, 2015).	8
Figure 2.2: Activation functions (SHARMA, 2017).	9
Figure 2.3: Graph of error rate (MOAWAD, 2018).	10
Figure 2.4: TCM pulse model (Tang et al., 2012).	11
Figure 2.5: Selection and criteria of normotensive and hypertensive subjects (Tang et al., 2012).	11
Figure 2.6: Background information on the subjects (N=260) (Tang et al., 2012).	12
Figure 2.7: Comparison between different algorithms (Tang et al., 2012).	13
Figure 2.8: Example of KNN (Eremenko and Ponteves, n.d.).	15
Figure 2.9: Equation of ERP distance function (Zuo et al., 2010).	15
Figure 2.10: Five types of pulse patterns will be classified by classifiers (Zuo et al., 2010).	16
Figure 2.11: Average classification accuracy of EDKC and GEKC compared to other classifiers (Zuo et al., 2010).	16
Figure 2.12: Example of SVM with two classes (Eremenko and Ponteves, n.d.).	17
Figure 2.13: Example of a decision tree (Eremenko and Ponteves, n.d.).	17
Figure 2.14: Algorithm of gradient boosting (Zhang and Haghani, 2015).	19

Figure 2.15: Time-domain parameters of a pulse signal (Wang and Zhang, 2009).	20
Figure 2.16: An example of the decision trees (D-M4) (Wang and Zhang, 2009).	21
Figure 2.17: Predictive accuracy rate (PAR) (Wang and Zhang, 2009).	21
Figure 2.18: IMF ₃ - IMF ₆ from a patient with CHD (Guo et al., 2015).	22
Figure 2.19: IMF ₃ - IMF ₆ from a normal person (Guo et al., 2015).	22
Figure 2.20: Average recognition rate using random forest (%) (Guo et al., 2015).	22
Figure 2.21: Results of the classification models (Luo et al., 2018).	23
Figure 2.22: Example of three clusters (Eremenko and Ponteves, n.d.).	24
Figure 2.23: Problem of random initialization (Eremenko and Ponteves, n.d.).	25
Figure 2.24: Elbow method (Eremenko and Ponteves, n.d.).	25
Figure 2.25: K-means clustering analysis to perform noise reduction (Luo et al., 2018).	26
Figure 2.26: Example of two clusters of FCM (李政軒, 2015).	27
Figure 2.27: Result (FENG and Li, 2018).	28
Figure 3.1: Overview of project work plan.	29
Figure 3.2: FBGA.	31
Figure 3.3: PLA cover. (Hew, 2019)	31
Figure 3.4: System setting of Sense 2020.	32
Figure 3.5: Playback saved spectra of Sense 2020.	32

Figure 3.6: Graph of pulse data waveform plotted using Excel.	33
Figure 3.7: Run Python script in CMD.	33
Figure 3.8: Tk GUI.	34
Figure 3.9: Correct and incorrect sitting posture.	34
Figure 3.10: Read and retrieve data from excel files.	35
Figure 3.11: Flowchart of a Python program.	36
Figure 3.12: FFT low-pass filter codes.	37
Figure 3.13: Zoom-in power against frequency graph.	37
Figure 3.14: Example of an original signal.	38
Figure 3.15: Example of a filtered signal.	38
Figure 3.16: Python codes of baseline wander removal.	39
Figure 3.17: Power against frequency graph.	39
Figure 3.18: Example of a signal after removed baseline wander.	39
Figure 3.19: Example of minimum points of a signal.	40
Figure 3.20: Example of a signal after cutting process.	40
Figure 3.21: Data written in “Data.xlsx”.	41
Figure 3.22: Machine learning classifiers written in python codes.	43
Figure 3.23: Confusion matrix.	44
Figure 4.1: Number of pulses in 3 seconds of non-sport category.	48
Figure 4.2: Number of pulses in 3 seconds of sport category.	48
Figure 4.3: Graph of frequency versus heights of left Cun’s pulse from non-sport category.	49
Figure 4.4: Graph of frequency versus heights of left Cun’s pulse from sport category.	49

- Figure 4.5: Graph of frequency versus heights of right Cun's pulse from non-sport category. 50
- Figure 4.6: Graph of frequency versus heights of right Cun's pulse from sport category. 50
- Figure 4.7: Confusion matrix with labels of KNN classifier. 51
- Figure 4.8: ROC curves of each machine learning models. 52
- Figure 4.9: ROC curves of each machine learning models. 53

LIST OF APPENDICES

APPENDIX A: Python codes to plot and save pulse data from an excel file.	61
APPENDIX B: Data Collection Form	63

CHAPTER 1

INTRODUCTION

1.1 General Introduction

There are some main diagnostic methods in Traditional Chinese Medicine (TCM) which are inspection, auscultation and olfaction, interrogation and pulse feeling. The pulse feeling is also known as pulse examination. The knowledge of pulse examination is very similar and relate to the theory of resonant blood circulation.

In a traditional way, TCM practitioners take and feel the pulses of a patient through sensation in practitioner's fingers (Wang et al., 2012). The pulses can be taken from many parts of body which are head, hands and legs. However, TCM practitioners normally take the pulses from the patient's hand (inch opening/wrist pulse) due to both parties' convenience. The wrist pulse can be taken from three points which are 'inch', 'bar' and 'cubit'. Figure 1.1 shows the location of the points at the wrist. Sometimes, information from both left and right wrist pulses are important for the pulse examination. There are three pressing levels for the points which are float, middle and deep which the levels can be more understandable by referring to Figure 1.1. There are many different types of pulses and those pulses can be classified to 30 pulses including normal pulse (Ping Mai). The other 29 sick pulses are Fu Mai (Floating), San Mai (Scattered), Ge Mai (Hard), Kou Mai (Hollow), Chen Mai (Deep), Fu Mai (Hidden), Lao Mai (Firm), Chi Mai (Late / Slow), Huan Mai (Slowed down), Su Mai (Rapid), Ji Mai (Racing), Xu Mai (Forceless), Duan Mai (Short), Shi Mai (Forceful), Chang Mai (Long), Hong Mai (Surging), Da Mai (Large), Xi Mai (Thin), Ru Mai (Soft), Ruo Mai (Weak), Wei Mai (Minute), Hua Mai (Smooth), Dong Mai (Moving), Se Mai (Hesitant), Xian Mai (Wiry), Jin Mai (Tight), Jie Mai (Knotted), Dai Mai (Regularly intermittent) and Cu Mai (Rapid-irregular). The inch, bar, and cubit of both hands have a close relationship with the organs.

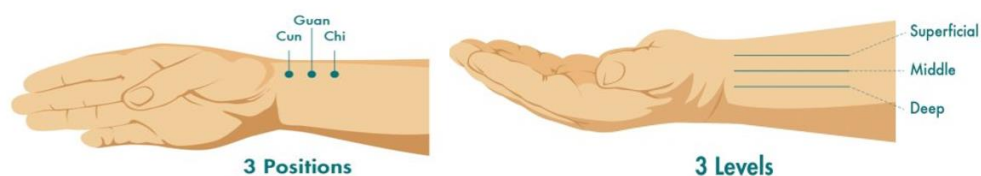


Figure 1.1: 3 positions and 3 levels of pulse taking (Jennifer Dubowsky, 2012).

Table 1.1: The relationships between left inch/bar/cubit, right inch/bar/cubit and organs based on the Classic of Difficult Issues 《难经》 (朱文锋, 2013).

Cun		Guan		Chi	
Left	Right	Left	Right	Left	Right
Heart	Lung	Liver	Spleen	Kidney	Kidney
Small Intestine	Large Intestine	gallbladder	Stomach	Bladder	Life Gate

For a modern way of pulse examination, there are some advanced technologies and devices were invented and developed as an aid for TCM practitioners to conduct pulse diagnosis on patients. One of the examples of the device is wrist pulse monitor. It can measure the blood pressure and heart rate of a person. Some of the devices can even provide images of the pulse waveforms. Some abnormal conditions of a person can be detected through these pulse waveforms. There are many different patterns of pulse waveforms which can match with the TCM pulse categories as mentioned above. TCM practitioners can use the device to help in pulse examination to have a more accurate and reliable diagnosis. Some of the devices have the ability to set different values of press force on the points of “Cun, Guan and Chi” and obtain three sets of waveforms which are superficial, middle and deep in order to suspect the correct level of the pulse. For example, a patient having “Chen Mai” will have obvious and clear pulse waveform at deep level which required the pressing force to be applied at a greater value. Moreover, the device will also give some other information such as the amplitude and time of the pulse waveform. Some of the devices able to provide diagnosis or classify the pulse type of the user. However, the diagnosis results are still not very accurate. This

might require to be improved by using machine learning or deep learning to create a program to classify the pulses and obtain a better result.

Fudan University partner with Shanghai University of Traditional Chinese Medicine which both located at China is inventing the Traditional medical machine named “TCM number one” shown in Figure 1.2. This machine provides TCM examination services which including pulse diagnosis, tongue diagnosis and etc. This machine can help to decrease the reliance on subjective judgements of TCM doctors and increase the precision of pulse examination (复旦研制出机器人“中医一号”, 2015). This is because the machine has the ability to learn the experiences from many different doctors through deep learning technique. This machine can be the key to create a “Baymax” in our real-life instead of just available in a fiction movie, “Big Hero 6”.



Figure 1.2: TCM Machine.

Since the coronavirus (COVID-19) pandemic occurred, doctors in some hospitals in China had combined TCM with Western medicine to treat COVID-19. Based on a CNN report on 16th March 2020, patients with mild symptoms in Wuhan that received combined treatments have a higher recovery rate and left the hospital sooner than patients that only received Western medicine. According to another report in Business Insider on 29th February, there are some young medical workers have died during this pandemic. if there are digitalized diagnosis devices are created to examine patients, then the medical workers can have less contact with the patients and able to decrease the risk of infection. Thus, it is urged and important to digitalize the TCM diagnosis to benefits as many humans as possible in the world.

1.2 Importance of the Study

This study plays an important role throughout the world as it can help to rescue people's lives in time because of the predictive model's accuracy and effectiveness. It can assist TCM practitioners to have some reliable information to be considered when diagnosing their patients. Some abnormal condition in a person can be detected as earlier as possible with the predictive model. It also can save many medical resources such as a patient with common and unserious disease can have their diagnosis or treatment quickly and easily. This may also help to save more times for doctors to focus on patients in a severe condition that require attention, surgery and treatment in time. This study might also help to improve the quality of life and as a step towards Society 5.0.

1.3 Problem Statement

Since there are so many types of pulse, it is hard for TCM practitioners to memorise all of the pulse conditions. There are many pulse diagnosis devices were invented and can generate the pulse waveform but the outputs still need to be differentiated, examined and determined by TCM practitioners. Is there any method and possibility to determine a patient's pulse automatically and more accurately? Machine learning comes in and being greatly applied into TCM and able to help TCM practitioners to predict the output of pulse examination of a patient.

TCM practitioners have their own ways and subjective judgements to examine their patients. Furthermore, they use fingers to sense the pulse conditions, examine the pulse rate by counting in heart and require a lot of experiences in order to have correct pulse-taking from a patient. This might lead to imprecise diagnosis if the TCM practitioners do not have enough experiences. TCM pulse diagnosis by machine learning can avoid these problems and can have deep learning based on different TCM doctor's experiences. The pulses will be digitized, provide clear pulse waveform images and create some parameters that can be used to analyse the pulse through machine learning.

Cun, Guan and Chi are not the only points for pulse taking. In fact, the pulse actually can also be taken from many other parts such as heaven, earth and man at the upper part (head), heaven, earth and man at the lower part (legs) and etc (Wang et al., 2012). Therefore, if the wrist pulse can be taken and diagnose

by machine learning, this can also be applied to parts that other than wrist. This can increase the efficiency of the examination and also able to obtain a more complete and reliable pulse diagnosis. This is because overall pulses can be taken into consideration when doing the diagnosis instead of just taking the wrist pulses only.

1.4 Aims and Objectives

The objectives of this project are listed below:

1. To understand different machine learning models for pulse data classification.
2. To automatically classify if a person is active in exercising or not through his/her TCM pulses using machine learning.
3. To evaluate the accuracy of the machine learning models in 2.

1.5 Limitation of the Study

There are some limitations will be faced throughout this project. One of the challenges is abundant of data is required in limited time. Besides, the data obtained initially might have many low accurate and low reliable data need to be processed or eliminated which makes data collections become tougher and time consumed. Moreover, the non-user friendly interface and inconvenient operation of pulse taking device lead to take a lot of time to do the data collection. Some inaccurate outputs and results obtained from the pulse-taking device also become one of the limitations. Professional TCM practitioners are crucial and very needed to assist in this project to have accurate, precise and reliable data as training set and test set for the machine learning model. It is also hard to obtain various types of pulses as data if the area of data collection is restricted unless hospital willing to offer and give permission to get pulse data from their patients. This is because hospital having huge sources of medical data which they have many patients with different types of disease and different severe levels of illness which can provide a lot of medical information that can be analyzed.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

What is machine learning? Machine learning is an application or subfield of artificial intelligence (AI) (Tagliaferri, 2017). The goal of machine learning is to develop the computer program or model and allow it to make decisions based on what it had learned through different patterns of data automatically. Data is the key requirement to develop machine learning. A massive quantity of data is needed because insufficient data will lead to inaccurate prediction and analysis. Generally, the minimum number of data need to be collected is 1000. The complexity of your problem increase, the number of data required will also increase in order to obtain an accurate and reliable output.

There is a difference between traditional programming and machine learning. Traditional programming is the program created manually and explicitly to solve a problem with algorithms (Parthasarathy, n.d.). Machine learning is augmented analytics that input and output data are fed to algorithms to create programs and automate decision-making process through learning from the data. This kind of programs created are also known as a model.

There are four categories of machine learning which are supervised learning, unsupervised learning, reinforcement learning and semi-supervised learning. The most two common types that frequently being applied are supervised learning and unsupervised learning.

Table 2.1: Explanation of types of machine learning (Pant, n.d.).

Types	Explanation
supervised learning	<p>The training data are labelled with desired outputs.</p> <p>This system predicts label values on additional unlabelled data based on historical data that it had learnt.</p> <p>Classification and regression</p>

unsupervised learning	The training data are given without labelled. This system can discover hidden patterns or features of a dataset. Clustering and association
reinforcement learning	Rewards or penalties will be received from a sequence of actions. It is a trial and error system to determine ideal actions need to be taken to get maximum performance.
semi-supervised learning	Incomplete training data with some target outputs. This system can improve learning accuracy.

2.2 Artificial Neural Network (ANN)

Artificial neural network (ANN) is a computational model that intended to replicate the structure and function of a human brain. A biological neural network of a brain consists of billions of neurons and trillions of synapses. Based on Figure 2.1, the A and C show the actual human neurons or also known as nerve cells while B and D are the artificial neurons or also named as nodes in computer. The synapse is the connection between neurons that used to transmit signals from one neuron to another. The synapses in ANN will be assigned with weights. These weights are crucial and can be adjusted through the process of machine learning in order to decide whether the signal obtained is important or not. In an artificial neuron, it added a weighted sum of all the input values and then apply an activation function.

$$\text{Weighted Sum of All The Input Values} = \sum_{i=1}^m w_i x_i \quad (2.1)$$

$$\text{Activation Function} = \phi \left(\sum_{i=1}^m w_i x_i \right) \quad (2.2)$$

where

w_i = weight

x_i = input values

The B in Figure 2.1 shows an artificial neuron takes in several input signals and produce an output signal. Several outputs can be produced in some cases such as categorical outputs. The output signal can then become an input signal for other neurons. This creates layers which will look like something shown in D in Figure 2.1. The layers can be classified into three layers which are input layer, hidden layer and output layer. A deep neuron network consists of many hidden layers.

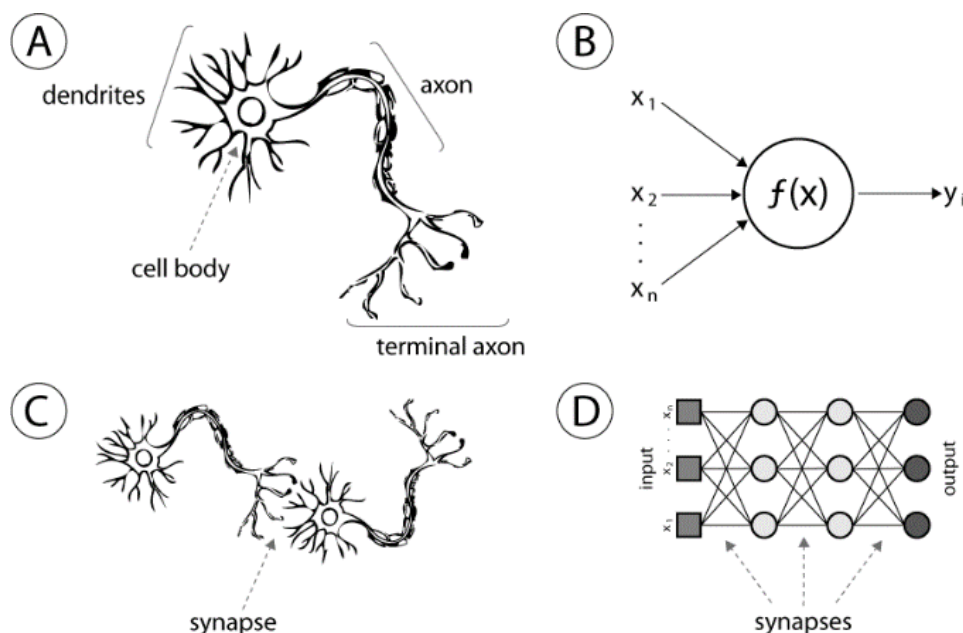


Figure 2.1: Differences between biological neurons and artificial neurons (Josh, 2015).

2.2.1 Activation Function

The activation function is used to convert the input signal of an artificial neuron to an output signal. There are many types of activation function such as binary step function (threshold function), linear activation function and non-linear activation function. Some examples of activation function are shown in Figure 2.2 below.










Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
Tanh		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parametric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Figure 2.2: Activation functions (SHARMA, 2017).

2.2.2 Back propagation

The predicted output obtained after the activation function of a neuron is applied will be compared to the actual output data. Then, an error rate will be received through equation (2.3).

$$\text{Error} = \sum \frac{1}{2} (\hat{y} - y)^2 \quad (2.3)$$

where

\hat{y} : predicted output

y: actual output

This error value will be back-propagated to the neuron and update or fine-tune the weights. Proper tuning of weight is needed to lower the error and have a more reliable model. Gradient descent or stochastic gradient descent will be applied in this back-propagation. From Figure 2.3, it is easy to understand how the gradient descent work. The minimum point is indicating the error is low or zero which also means the weight does not need to be adjusted. However, if the slope of the error is negative, it means that the weight needs to be increased

while if the slope of the error is positive, the weight needs to be decreased so that the weight can be adjusted until minimum error rate is obtained.

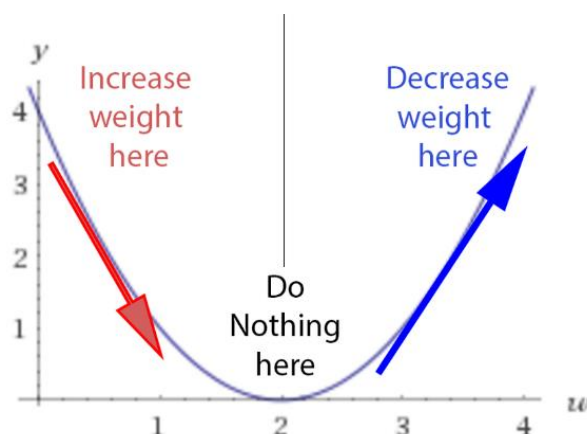


Figure 2.3: Graph of error rate (MOAWAD, 2018).

2.2.3 Related Work

ANN is the ideal modelling technique to build the TCM pulse diagnostic model based on the concept of Yin and Yang (Tang et al., 2012). TCM pulse quality needs to be described by different intensity of eight elements. The eight elements are analysed from depth to take the pulse, beats per breath (rate), the rhythm of a pulse (regularity), width and length that observed from the shape of the pulse and sensation of the pulse which is smoothness, forcefulness and stiffness. These eight elements are integrated with six locations or organs of a human which are lung, kidney, spleen, heart, liver and lifegate. A die model was created and shown in Figure 2.4. The Yin and Yang, eight elements and six locations can be illustrated in Figure 2.4. The health status or blood pressure of a person will be influenced by the interaction, interchange and balancing between the Yin and Yang. Both left and right of the pulse were taken to analyse the Yin and Yang of a person. The Yin is referred to as the assessment of the blood while the Yang is referred to as the assessment of lung or Qi. The shaded area at the lower right in Figure 2.4 is representing the Yin while the white area is representing the Yang. This is because Yang is always outside while Yin is always inside (Maciocia, 1989).

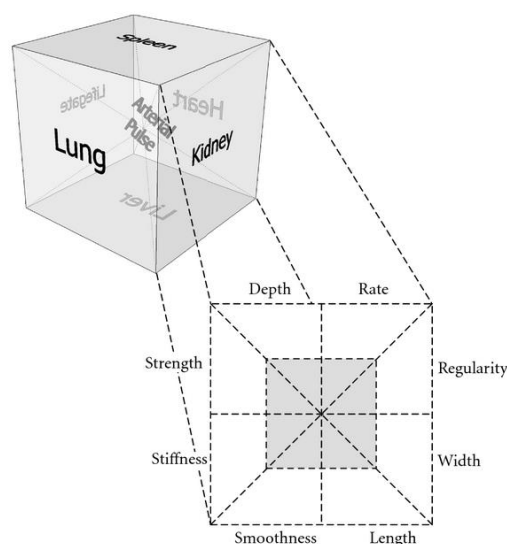


Figure 2.4: TCM pulse model (Tang et al., 2012).

In this paper, the selection of participants was processed. A standard criterion of a patient is shown in Figure 2.5. Normotensive and hypertensive are their target criteria to standardize and normalize the dataset for their ANN. Some standard pulse-taking procedures were also performed. Figure 2.6 shows that there is a significant difference in blood pressure and pulse rate between people with normotensive and hypertensive.

Selection criteria	Normotension	Hypertension
Inclusion criteria	(i) Aged 18 or above (ii) SBP <120 mmHg and DBP <80 mmHg at rest* [31]	(i) Aged 18 or above (ii) Diagnosis of essential hypertension (iii) SBP \geq 140 mmHg/DBP \geq 90 mmHg or both* [31]
Exclusion criteria	(i) Pregnant (ii) Loss of upper extremities (iii) Chronic diseases (iv) Infectious diseases (v) Current use of any medication including prescription from a medical doctor, herbal medicine and OTC drugs	(i) The same as those in normotension except the current use of antihypertensive drugs

* SBP: systolic blood pressure; DBP: diastolic blood pressure.

Figure 2.5: Selection and criteria of normotensive and hypertensive subjects (Tang et al., 2012).

		Normotensive group ^ψ (n = 139)	Hypertensive group ^ψ (n = 121)	Significance* (P < 0.05)
Gender	Male	55	53	—
	Female	84	68	
Age (yrs)	18–34	43	9	—
	35–64	86	93	
	≥65	10	19	
BMI (kg/m ²)	<18.50	12	5	—
	18.50– 22.99	103	96	
	≥23.00	24	20	
LBP (mmHg)	112 (13)/68 (7)	150 (16)/95 (11)	0.01	
RBP (mmHg)	113 (13)/68 (7)	150 (15)/95 (11)	0.01	
Pulse Rate (bpm)	65 (10)	70 (11)	0.01	

BMI: Body mass index; LBP: Left-side blood pressure; RBP: Right-side blood pressure.

^ψThe mean (SD) is reported for the continuous variables and the frequency for the categorical variables.

*P < 0.05 denotes statistical significance.

Figure 2.6: Background information on the subjects (N=260) (Tang et al., 2012).

Both three-layer ANN and four-layer ANN had been tried in this paper. Bayesian Regularization (BRANN), Levenberg-Marquardt (LM) and Resilient Backpropagation (RPROP) are different kinds of algorithms of the ANN system. LM and RPROP algorithms are faster than other algorithms to train a neural network. BRANN algorithm can decrease the need for cross-validation (Burden and Winkler, 2008). The activation function used to connect the layers are pure linear function and log-sigmoid function. A probabilistic neural network with radial basis function is also used in this paper. The probabilistic neural network is also known as a feedforward neural network. It has one hidden layer only. The comparison between different algorithms are made and the result is shown in Figure 2.7. From the result, the authors claimed that BRANN or RPROP is the best algorithm to train the model as both of the algorithms have higher specificity and sufficiently high accuracy and sensitivity.

Algorithm	Specificity (%)	Sensitivity (%)	Accuracy (%)	Remarks
Bayesian regularization	73.46	84.49	79.27	15 hidden neurons
Levenberg-Marquardt	68.29	86.75	78.06	10 hidden neurons
Resilient backpropagation	72.19	83.91	78.41	10 hidden neurons
Probabilistic neural network	68.49	78.32	73.76	—

Figure 2.7: Comparison between different algorithms (Tang et al., 2012).

Another ANN model was made by Yue to predict some common types of the pulse. The dataset for the model is obtained from 1456 selected patients through pulse examination. The training algorithm used in the model is the backpropagation algorithm and its accuracy is higher than 92% while the recognition ability is higher 82% (Zhang, 2010). Hu also tends to create ANN model that can classify combination types of the pulse (Zhang, 2010).

Wang and Xiang claimed that the accuracy of ANN to predict some types of the pulse such as normal and slippery is 12% higher than the fuzzy inference system (Yan Tang, 2012). All of this information proves that using ANN to diagnose pulses is workable.

2.3 K-Nearest Neighbours (K-NN)

K-NN is a simple and basic supervised learning, classification algorithm. It is widely used in the application of intrusion detection, pattern recognition and data mining. Classes of the dataset are predefined from labelled input data initially and the class of new data point can be predicted. The new data point is classified depends on majority votes of its neighbour. The distance function is used to determine the votes. There are some different distance functions can be applied to the model.

Table 2.2: Types of distance functions and equations (K Nearest Neighbors - Classification, n.d.).

Distance Functions	Equation
Euclidean	$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$ (2.4)
Manhattan	$\sum_{i=1}^k x_i - y_i $ (2.5)
Minkowski	$\left(\sum_{i=1}^k (x_i - y_i)^q \right)^{1/q}$ (2.6)
Hamming	$D_H = \sum_{i=1}^k x_i - y_i \ ; \ x = y \rightarrow D = 0$ $x \neq y \rightarrow D = 1$ (2.7)
Standardized distance	$X_s = \frac{X - Min}{Max - Min}$ (2.8)

There are some advantages and disadvantages of this algorithm. It can give a high accuracy of prediction and it is very versatile (Bronshtein, 2017). However, the drawback of this algorithm is computationally expensive which it requires to store all training data to build the model leads to high memory is needed, sensitive to irrelevant features and slow compute time.

An example is shown in Figure 2.8. The number of K neighbours need to be chosen and the default value of K is five. The distances between the new point and K neighbours will be computed. In the case shown in Figure 2.8, Euclidean distance is applied. The new point will then be grouped to the highest count of neighbours. The number of red neighbours is more than the number of green neighbours. So, the new point is assigned to the red category.



Figure 2.8: Example of KNN (Eremenko and Ponteves, n.d.).

2.3.1 Related Work

There is research used Edit Distance with Real Penalty (ERP) as the distance function in KNN classifier with dynamic feature weighting (DFW) to classify the pulse patterns. The equation of ERP distance function is shown in Figure 2.9. This kind of classifier is known as EDKC classifier. They also proposed another type of classifier which is Gaussian ERP kernel classifier (GEKC) that it extends from EDKC by defining kernel Gram matrix. The pulse patterns shown in Figure 2.10 are moderate, smooth, taut, hollow, and unsmooth pulse patterns that will be classified using different types of classifier (Zuo et al., 2010).

$$\begin{aligned}
 & d_{\text{erp}}(\mathbf{a}, \mathbf{b}) \\
 & = \begin{cases} \sum_{i=1}^m |a_i - g| & \text{if } n = 0, \\ \sum_{i=1}^n |b_i - g| & \text{if } m = 0, \\ \min \left\{ \begin{array}{l} d_{\text{erp}}(\text{Rest}(\mathbf{a}), \text{Rest}(\mathbf{b})) + |a_1 - b_1|, \\ d_{\text{erp}}(\text{Rest}(\mathbf{a}), \mathbf{b}) + |a_1 - g|, \\ d_{\text{erp}}(\mathbf{a}, \text{Rest}(\mathbf{b})) + |b_1 - g|, \end{array} \right\} & \text{otherwise,} \end{cases}
 \end{aligned}$$

Figure 2.9: Equation of ERP distance function (Zuo et al., 2010).

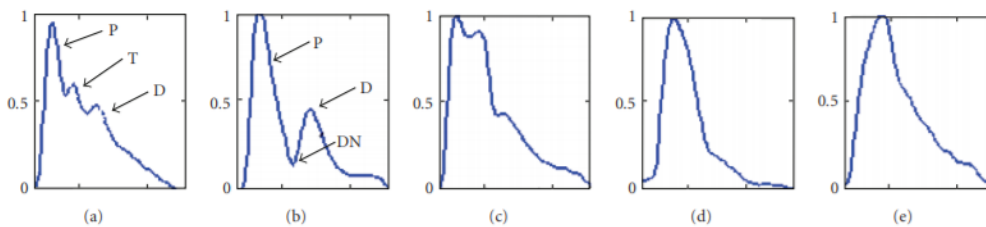


Figure 2.10: Five types of pulse patterns will be classified by classifiers (Zuo et al., 2010).

There is an interclass variation problem in the dataset. This means that there are some pulse waveforms look similar to another type of pulse waveform that can cause confusion to the classifier and classify the pulses wrongly. In Figure 2.11, it shows that the EDKC and GEKC can greatly solve this variation and time sifting problems and result in high accuracy to predict the types of the pulse.

Pulse waveform	1NN-Euclidean	1NN-DTW	1NN-ERP	Wavelet network [23]	IDTW [19]	EDKC	GEKC
<i>Moderate</i>	86.11	82.44	88.31	87.23	87.31	89.94	91.25
<i>Smooth</i>	85.02	81.16	86.31	85.36	80.38	86.00	87.09
<i>Taut</i>	95.76	87.95	95.10	89.63	93.15	95.50	96.88
<i>Hollow</i>	86.75	82.44	87.56	85.63	80.44	86.88	89.38
<i>Unsmooth</i>	84.06	70.81	84.75	80.63	89.50	85.00	86.88
<i>Average</i>	87.36	83.19	89.79	87.08	88.90	90.36	91.74

Figure 2.11: Average classification accuracy of EDKC and GEKC compared to other classifiers (Zuo et al., 2010).

2.4 Support Vector Machine (SVM)

The aim of SVM is to discover and set hyperplane or decision boundary in an N-dimensional space in order to classify the data points clearly (Gandhi, 2018). The best decision boundary is to have a maximum margin between support vectors. If one of the support vectors is removed, the decision boundary will change. The example of this SVM is shown in Figure 2.12. The advantages of this algorithm are it is very effective to be used in high dimensional spaces and gives a clear margin of separation of the classes (RAY, 2017).

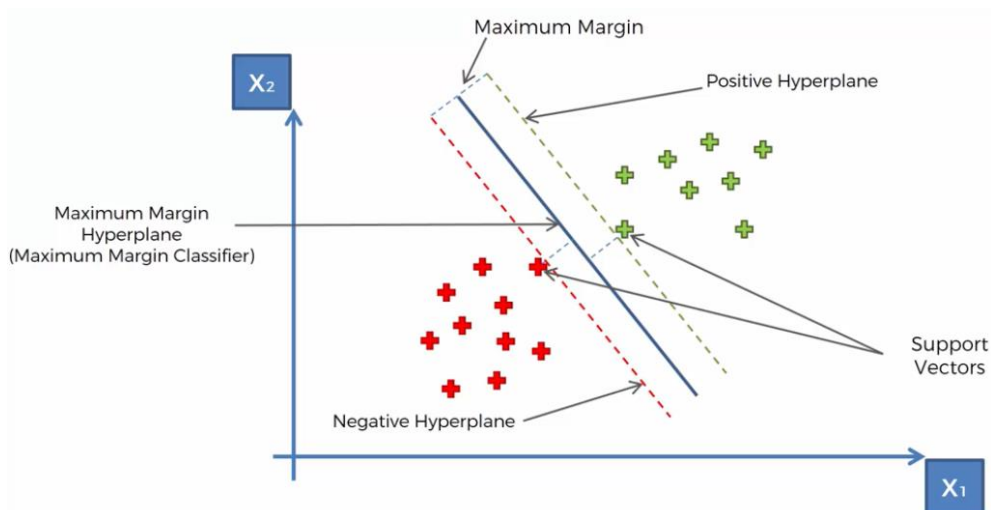


Figure 2.12: Example of SVM with two classes (Eremenko and Ponteves, n.d.).

2.5 Decision Tree, Random Forest (RF), AdaBoost and Gradient Boosting

The **decision tree** is an old method and recently was replaced with many other new models such as random forest, gradient boosting and AdaBoost. For an example of a decision tree, if there is a new point comes in, $x = 22$, then it will fall into the green category based on the decision tree shown in Figure 2.13.

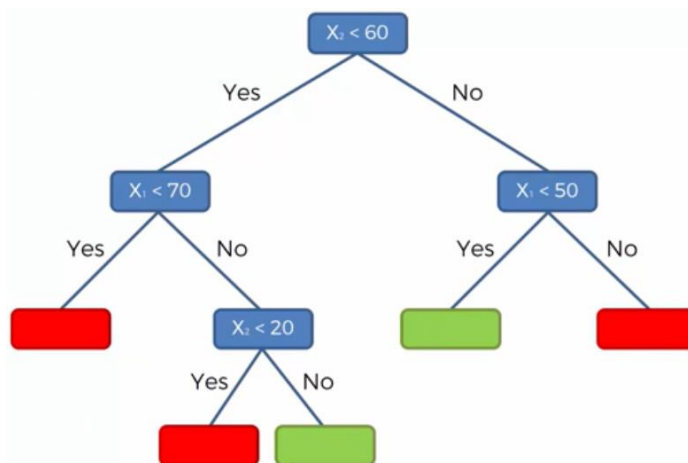


Figure 2.13: Example of a decision tree (Eremenko and Ponteves, n.d.).

Random forest is one of the ensemble learnings. In simple words, it is an upgraded version of the decision tree. The random forest as its name implies a lot of decision trees will be created to build up the RF. Firstly, K data points

from the training set will be chosen to build a decision tree. This step will be repeated based on number Ntree of trees is selected to build up the forest. When a new data point is added in, each Ntree trees predicts the suitable category for the new data point and the data point will be assigned to the majority votes from the Ntree trees.

AdaBoost is also based on the concept from the decision tree. Generally, the decision tree in AdaBoost has just one node and two leaves which also known as a stump. Many stumps together will form a forest of stumps. Stump is a “weak learner” because it makes the decision based on one variable only. The order of stump in AdaBoost is important as the errors made in the previous stump will influence the next stump is made. Some stumps have a higher level of influence (amount of say) in making the decision in classification than other stumps.

Firstly, sample weights were evenly contributed to each sample which is one divided by a total number of samples. Stumps were made based on each variable. Gini index will be calculated for each stump. The Gini index decides the sequence of the stump. The stump with the lowest Gini index will be the first. Then amount of say will be calculated based on equation (2.9). The total error or misclassification rate is equal to sum of the weights for incorrectly classified samples which can be calculated based on the equation (2.10) (Jung, 2018). If the total error is less, the amount of say will have a high positive value while if the total error is large, the amount of say will have a high negative value. A stump with negative value will inverse the vote. For example, a stump initially votes the predicted output as “have diabetes” will change to vote “no have diabetes” if it has negative amount of say. The sample that incorrectly classified in a stump will need to increase its sample weight while other samples need to decrease its sample weight. Equation (2.11) is used to adjust or update the sample weight. The calculated new sample weight will be normalised with normalization factor and replace the old sample weight. The new sample weight will be used by the next stump to calculate amount of say, total error and so on.

$$\text{Amount of say} = \frac{1}{2} \ln \left(\frac{1 - \text{Total Error}}{\text{Total Error}} \right) \quad (2.9)$$

$$\text{Total Error} = \frac{\sum_{i=1}^N w_i y_i}{\sum_{i=1}^N w_i} \quad (2.10)$$

$$w_{new,i} = w_{old,i} \times e^{-\text{Amount of say}(y_i)(h_i)} \quad (2.11)$$

Where

w_i : sample weight of the i^{th} sample.

y_i : the actual output of the i^{th} sample (the value will be 1 if wrongly classified and 0 if correctly classified)

h_i : predicted output of the i^{th} sample.

$w_{new,i}$: new sample weight of the i^{th} sample.

$w_{old,i}$: old sample weight of the i^{th} sample.

Gradient boosting is also made up of decision trees. The decision tree of gradient boosting is larger than the stump of AdaBoost. A single leaf that contains initially predicted output will be created first before build the decision trees. The fixed-sized decision tree in gradient boosting also depends on error or correct the error made by previous decision trees. The algorithm of gradient boosting is shown in Figure 2.14: Algorithm of gradient boosting (Zhang and Haghani, 2015)..

```

Initialize  $f_0(x)$  to be a constant,  $f_0(x) = \operatorname{argmin}_{\rho} \sum_{i=1}^N L(y_i, \rho)$ .
For  $m = 1$  to  $M$  do
  For  $i = 1$  to  $n$  do
    Compute the negative gradient
      
$$z_{im} = - \left[ \frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f=f_{m-1}}$$

  End;
  Fit a regression tree  $g_m(x)$  to predict the targets  $z_{im}$  from covariates  $x_i$  for all training data.
  Compute a gradient descent step size as
      
$$\rho_m = \operatorname{argmin}_{\rho} \sum_{i=1}^n L(y_i, f_{m-1}(x_i) + \rho g_m(x_i))$$

  Update the model as
      
$$f_m(x) = f_{m-1}(x) + \rho_m g_m(x)$$

End;
Output the final model  $f_M(x)$ 

```

Figure 2.14: Algorithm of gradient boosting (Zhang and Haghani, 2015).

2.5.1 Related Work

A classification of pulse strength using decision tree algorithm is done by Wang Huiyan and Zhang Peiyong. Some of the time-domain parameters of pulse signal used as the dataset to train the decision tree are shown in Figure 2.15. The parameters are extracted using wavelet transform (Wang and Zhang, 2009). Some ratios calculated from the division between two different types of heights are also collected as parts of the dataset. The ratios are important parameters to show some critical features to diagnose the pulses. For instances, the ratio of the height of the tidal wave divided by height of percussion wave (augmented index) can indicate the aortic pressure. An increase in this ratio will cause an increase in cardiovascular risk (Wilhelm et al., 2007). The decision tree in this paper is used to classify the pulse signals into three categories which are normal strength pulse (NS-pulse), replete pulse (R-pulse) and feeble pulse (F-pulse) (Wang and Zhang, 2009). Six decision tree models are made and each of them is different from each other in terms of parameter used or number of leaves. An example of the decision trees and predictive accuracy rate (PAR) of all the decision trees are shown in Figure 2.16 and Figure 2.17 respectively. D-M4 has the highest PAR among the decision trees.

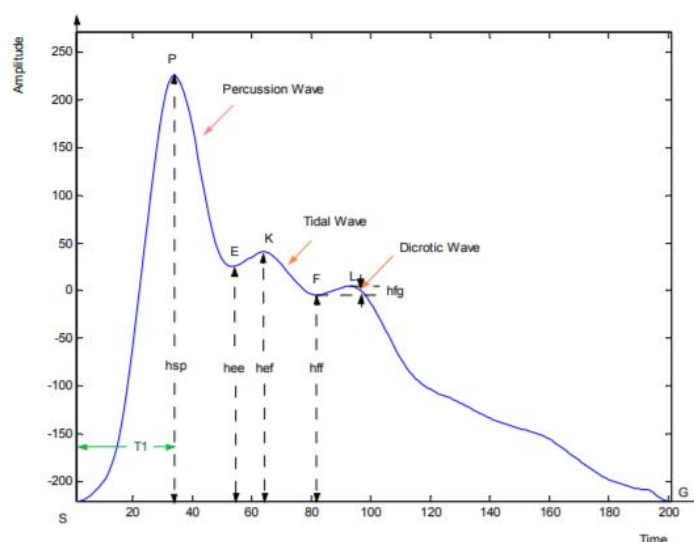


Figure 2.15: Time-domain parameters of a pulse signal (Wang and Zhang, 2009).

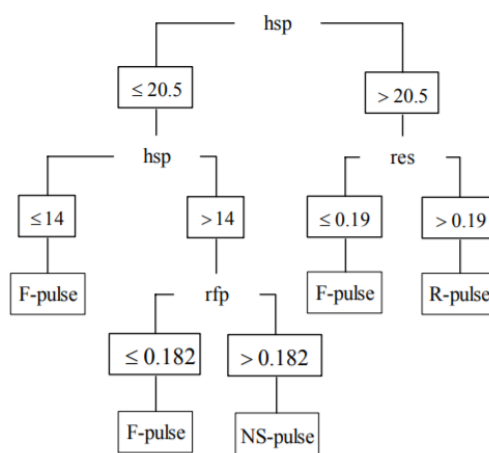


Figure 2.16: An example of the decision trees (D-M4) (Wang and Zhang, 2009).

D-M1	0.9103±0.0448
D-M2	0.9113±0.0281
D-M3	0.9167±0.1912
D-M4	0.9220±0.1882
D-M5	0.9017±0.0261
D-M6	0.9032±0.0298

Figure 2.17: Predictive accuracy rate (PAR) (Wang and Zhang, 2009).

There is another research uses Hilbert-Huang Transform (HHT) and random forest approach to analyse the TCM pulse of patients that have coronary heart disease (CHD). HHT is widely and typically used to analyse and decompose a non-stationary and nonlinear signal into a set of intrinsic mode functions (IMFs) through empirical mode decomposition (EMD) (Wang et al., 2010). HHT is implemented by identifying the local maxima and minima of a signal initially. Then, the upper envelope and lower envelope are deduced by interpolation and compute the mean envelope. The mean envelope will then be subtracted from the signal and these steps are iterated until the number of extrema is differed with the number of zeroes by one to obtain the IMF. The residual will be iterated again to obtain another IMF. Most of the pulse signals can have a number of 7 or higher of the IMF and a residual parameter (Guo et al., 2015). Two groups of pulse signals are collected from some healthy people and patients with CHD. The IMF can denote characteristics of pulse signal with some resolution. The IMF₃ to IMF₇ from CHD group have more high-frequency

parts compared to the normal group as shown in Figure 2.18 and Figure 2.19. The energy feature and sample entropy of each IMF of a pulse signal can be calculated and extracted to analyse and classify the pulse signal quantitatively using random forest with 500 decision trees. The result of the average recognition rate is shown in Figure 2.20.

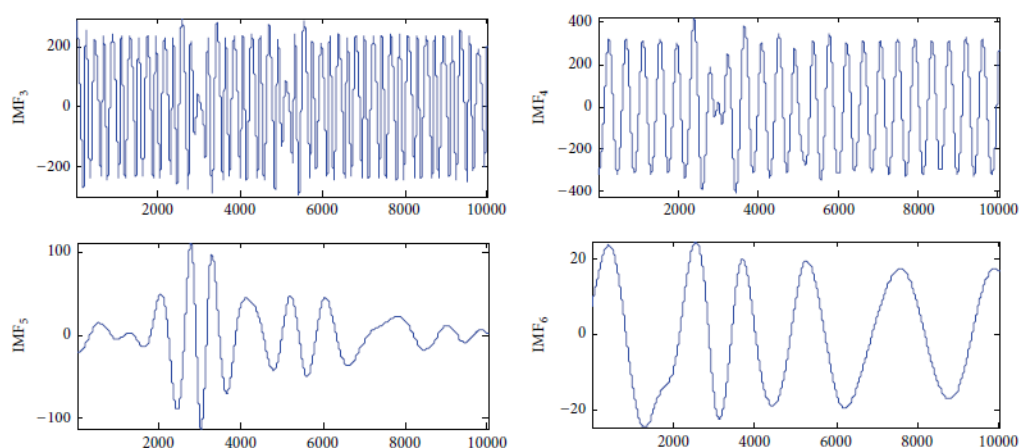


Figure 2.18: IMF₃ - IMF₆ from a patient with CHD (Guo et al., 2015).

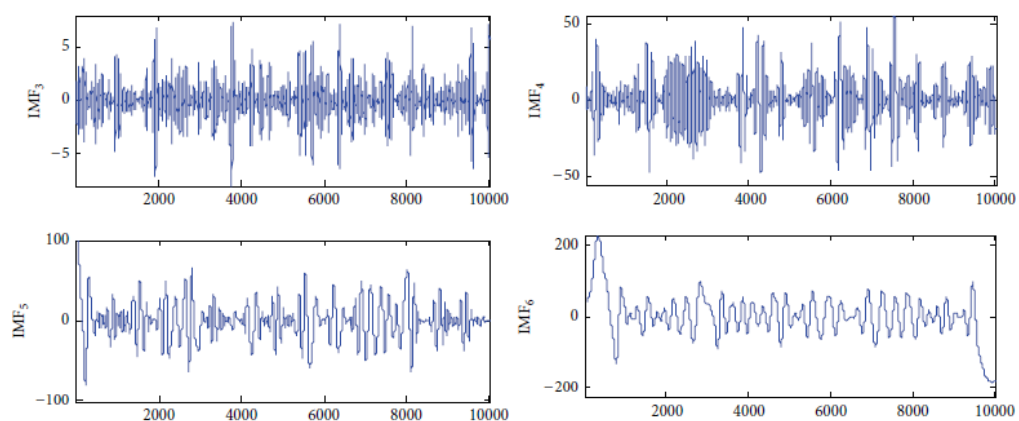


Figure 2.19: IMF₃ - IMF₆ from a normal person (Guo et al., 2015).

Group	Feature		
	IMF sample entropy	IMF entropy	Combination of IMF sample entropy and IMF sample entropy
The CHD group	88.32 ± 3.89	89.20 ± 6.76	95.49 ± 2.78
The normal group	53.48 ± 10.15	73.99 ± 15.47	80.07 ± 9.13
The two groups	76.35 ± 4.77	84.00 ± 5.04	90.21 ± 4.09

Figure 2.20: Average recognition rate using random forest (%) (Guo et al., 2015).

Furthermore, another study also used five data mining algorithms classifications which are RF, gradient boosting, SVM, AdaBoost and KNN to build machine learning model to identify pulse wave parameter between hypertension group and healthy group (Luo et al., 2018). The accuracy rate, area under ROC curve (AUC), sensitivity (ST) and specificity (SP) of the classification models used in the study are shown in Figure 2.21. The ROC stands for receiver operating characteristic.

Model	ACC	AUC	SP	ST
RF	0.841	0.832	0.936	0.728
RF*	0.853	0.848	0.905	0.792
Gradient Boosting	0.852	0.843	0.941	0.746
Gradient Boosting*	0.864	0.859	0.920	0.798
SVM	0.796	0.792	0.833	0.752
SVM*	0.832	0.828	0.865	0.792
AdaBoost	0.839	0.830	0.921	0.740
AdaBoost*	0.864	0.858	0.925	0.792
KNeighbors	0.729	0.716	0.852	0.580
KNeighbors*	0.736	0.728	0.830	0.625

Models with an asterisk * mean that K-means is applied before using these models.

Figure 2.21: Results of the classification models (Luo et al., 2018).

2.6 K-means

K-means is an unsupervised learning algorithm which can solve some clustering problem (Trevino, 2016). It is very simple, easy and popular to be applied. The purpose of this algorithm is to group the data points based on their features and similarities. Initially, a suitable number of K clusters need to be chosen. K centroids need to be selected based on K points. Then, each data point will be assigned to the closest centroids based on Euclidean distance function to form K clusters. After all the data points had been assigned, the positions of the K centroids need to be recalculated and the data points will be assigned again based on the new K centroids. This step will keep repeat until the centroids do not move anymore and the model is ready to be used.

Figure 2.22 shows an example of three clusters which the crosses are representing the data points, square boxes are the centroids and the circles are the clusters. This example shows a quite successful clustering model. However, there is a problem if the initialization of centroids is randomly selected. A bad random initialization of centroids will cause a possibility of bad or low reliable clusters will be produced. The example of the bad clustering model is shown in

Figure 2.23. There is a method to solve this problem which is K-means ++ algorithm and the equation (2.12) is used for this algorithm.

$$\text{WCSS or } J = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2 \quad (2.12)$$

Where

WCSS: within-cluster sum of squares

k: number of clusters

n: number of cases

$x_i^{(j)}$: case i

c_j : centroid for cluster j

Basically, it is used to calculate the square of distance between each data points of case i with its centroid. As the J is decreasing, the number of clusters is increasing, the better the data is fitted. However, the maximum number of cluster is equal to the number of data points and this is not a good select for the number of clusters. So, the best way to determine the optimal number of cluster is to use the elbow method. Based on Figure 2.24, there will have an obvious drop of J when the number of clusters is increased and then the J continue to drop insignificantly. The optimal number of cluster will be the point where the J drop not so substantially.

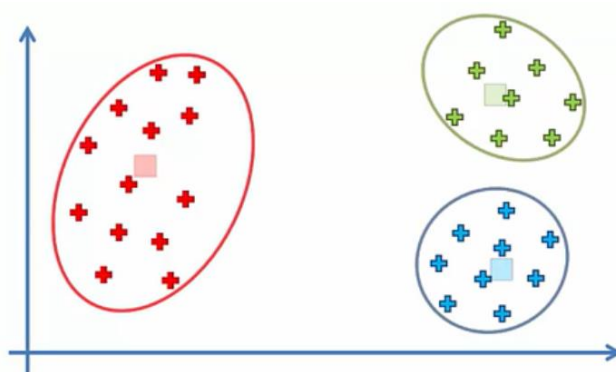


Figure 2.22: Example of three clusters (Eremenko and Ponteves, n.d.).

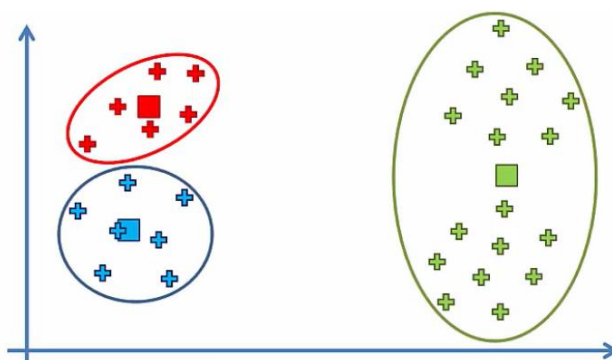


Figure 2.23: Problem of random initialization (Eremenko and Ponteves, n.d.).

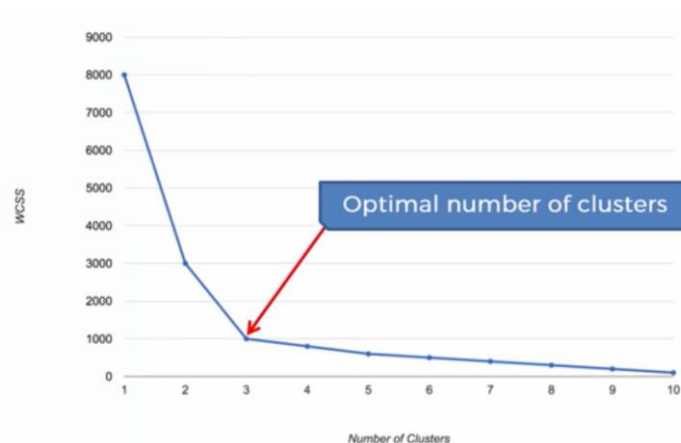


Figure 2.24: Elbow method (Eremenko and Ponteves, n.d.).

2.6.1 Related Work

K-means algorithm can be used to denoise the collected pulse signals before fit into other machine learning models. The data is divided into two groups through the K-means algorithm which one group of clusters have a certain degree of similarity in each cluster while another group does not (Luo et al., 2018). In Figure 2.25, the blue points are indicating the normal pulse wave while the red points are indicating the noise pulse wave that needs to be filtered. In Figure 2.21, the accuracy of the classification of pulse wave of machine learning models had an increased after filtered out the noise pulse wave.

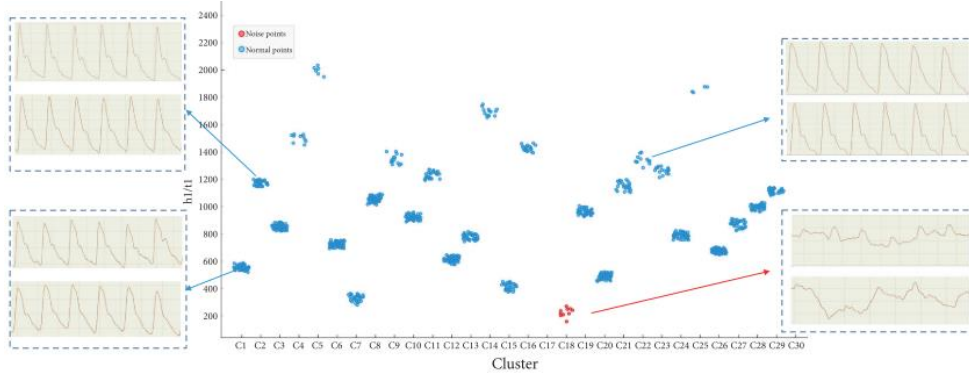


Figure 2.25: K-means clustering analysis to perform noise reduction (Luo et al., 2018).

2.7 Fuzzy C-Means (FCM)

FCM is a clustering method and it assigns “degree of truth” to the data instead of boolean logic to the data. This means that the data can belong to multiple clusters or classes instead of one cluster. It is useful in pattern recognition. Initially, every point is assigned with some membership values, u_{ij} to initialize an objective function, $J^{(0)}$. If there are two clusters, then each point will have two membership values and an example of this is shown in Figure 2.26. The centres will be calculated based on equation (2.13) (Fuzzy C-Means Clustering, n.d.). The membership values will need to be recalculated based on the new centres acquired in the previous step. Equation (2.14) is used to recalculate the membership value. The objective function needs to be computed after each time the centres and membership values are obtained. The objective function is calculated from the equation (2.15), where $m \geq 1$. The values of centres and memberships need to recalculate repeatedly until the difference between objective functions is lesser than the termination criterion based on equation (2.16).

$$c_i = \frac{\sum_{j=1}^N u_{ij}^m x_j}{\sum_{j=1}^N u_{ij}^m} \quad (2.13)$$

$$u_{ij} = \frac{1}{\sum_{k=1}^C \left(\frac{\|x_j - c_i\|}{\|x_j - c_k\|} \right)^{\frac{2}{m-1}}} \quad (2.14)$$

$$J^{(t)} = \sum_{i=1}^N \sum_{j=1}^C u_{ij}^m \|x_j - c_i\|^2 \quad (2.15)$$

$$\|J^{(t)} - J^{(t-1)}\| < \epsilon \quad (2.16)$$

Where

c_i : dimension centre of the cluster

u_{ij} : degree of membership of x_j in cluster i .

x_j : j^{th} of d -dimensional measured data

$J^{(t)}$: Objective function

ϵ : termination criterion with a value between 0 to 1

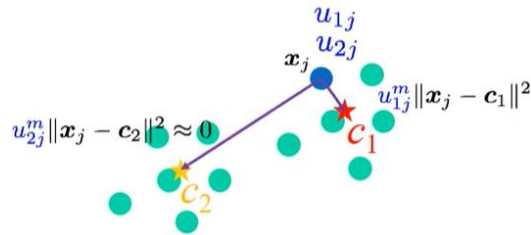


Figure 2.26: Example of two clusters of FCM (李政軒, 2015).

2.7.1 Related Work

In the previous discussion of the researches, most of them are using supervised learning algorithms to build the pulse classification machine learning model. When the data is increasing abundantly, these data will be required to labelled by different TCM doctors. As the price of the medical devices gradually decreased, the quantity of data is continually increasing but the number of professional doctors to analyse and determine the pulse data is limited. Moreover, different doctors will have their own subjective judgements on pulses. Therefore, this condition will cause some incorrect or inaccurate labelling of the pulse data which will later lead to wrong predictions. This will also be a hindrance for the objectification of pulse diagnosis (FENG and Li, 2018). Thus, the unsupervised learning algorithm can be an effective and ideal solution to train the data and classify the data without or less subjectively.

The collected pulse signals are usually inevitable from noise and drifting baseline. Dual-tree complex wavelet transform is implemented in the study to solve the drifting baseline problem in the pulse signals (FENG and Li, 2018). In the study, the pulse signal is normalised to 128 Hz for each cardiac cycle and perform zero padding to adjust and normalise the length of the pulse signal to 128 points. Then, the characteristic parameters or features of the pulse is extracted using Mel-frequency cepstral coefficient (MFCC). Some other approaches such as linear discrimination (LD), power spectral analysis (PSA) with Fast Fourier Transform (FFT), linear predictive coding (LPC) and linear prediction cepstral coefficient (LPCC) are also used to extract the characteristic parameters in the pulse signals. K-means, KNN and FCM classifiers are used to analyse and classify the pulse signals based on the characteristic parameters. Theoretically, the membership value should in between 0 to 1, but the membership value in this study is allowed to exceed one and this brings a better result. The overall result is shown in Figure 2.27. FCM classifier with membership value more than one, associated with parameters extraction from MFCC has the best result to classify the pulse signals.

聚类方案/特征提取	LD	FFT	LPC	LPCC	MFCC
k-MEANS	36.2	55.1	44.9	43.5	56.5
KNN	43.4	53.6	46.4	49.2	57.9
FCM(隶属度=1)	40.5	71.0	50.7	52.1	76.8
FCM(隶属度>1)	44.9	71.0	55.1	52.1	78.2

Figure 2.27: Result (FENG and Li, 2018).

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

Figure 3.1 shows the overview of project work plan. Data samples are collected from 56 people in this project. Machine learning classifiers are applied in this project to group the samples to two categories which are person who active in exercising or person who are not active in exercising. Type of exercising can be swimming, jogging, gymnastics and etc.

The two main software used in this project are Sense 2020 V1610 20141106 and Spyder. Sense 2020 is a software provided by Bayspec as an interface, operated with fibre Bragg grating analyser to collect raw data while Spyder is an integrated development environment (IDE) to code and run scripts written in Python programming language.

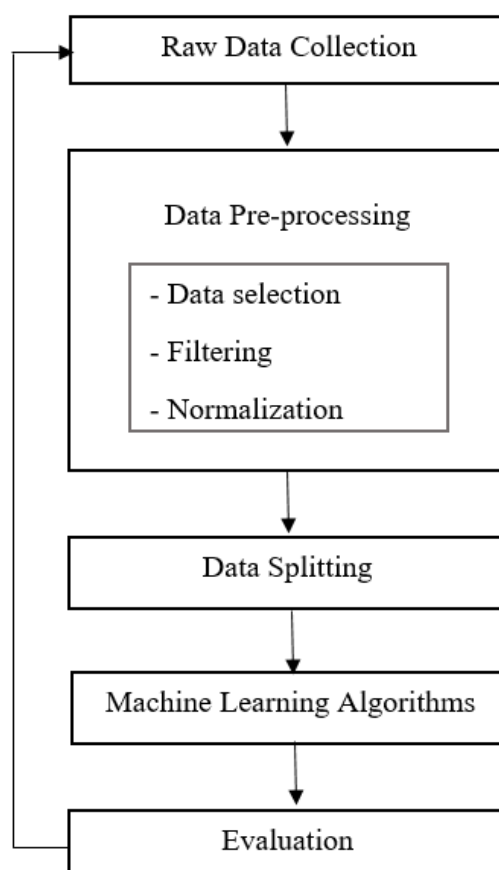


Figure 3.1: Overview of project work plan.

3.2 Data Collection

The required devices used to collect TCM pulse data in this project are amplify spontaneous emissions (ASE), fibre Bragg grating analyser (FBGA) that shown in Figure 3.2 and fibre optic sensor (FBG) made by previous FYP student (Hew, 2019). The fibre optic sensor was improved to allow a force to be exerted on it; i.e. the current design of the sensor has a polylactic acid (PLA) cover built using 3D printing to protect the fibre sensor from frequent ruptures. The design of the PLA cover is shown in Figure 3.3. The polydimethylsiloxane (PDMS) is the first layer of protection for the sensor and the PLA cover act as the second layer protection to protect the sensor from breaking when exerting a force on it. The PLA cover can also decrease or eliminate some noises such as surrounding voices that will affect the performance of the sensor. The melting point of PLA is 180 °C to 220 °C. The curing time of PDMS is 48 hours at room temperature. So, the sensor with PLA cover was put in a 70 °C drying oven without sharing with others in UTAR 7th floor chemistry laboratory to speed up the curing time of the PDMS instead of room temperature when building the sensor.

Next, to collect raw pulse data using Sense2020, the sample rate has to be set at 1000 Hz, high dynamic range mode, and peak search with uniform threshold at 2000. If it shows more than one peak, the output of ASE needs to be adjusted until only one peak. Then, back to the system setting and tick the “Fast Record Acquired Spectra into Binary Files”. The peak search setting need to be set as shown in Figure 3.4 before the “Confirm” button is clicked. Then, the exact locations of TCM pulses of one hand (Cun, Guan and Chi) need to be found before placing the sensor on the participant’s hand. Medical tape can be used to fix the position of the sensor on the participant’s hand. When the “start continuous acquisition” button is clicked, the system starts to record the spectrum data taken from the sensor in a binary file which is a .dat file. Different forces will exert on the sensor to take the pulses on the surface layer, middle layer and deep layer of the hand, 10 seconds are taken for each force during the recording. The time is calculated using an electronic watch. Then, “stop continuous acquisition” button is clicked when want to stop the process. Open and select the record file by clicking “Open Record File” at playback saved spectra shown in Figure 3.5. Once the “Search peaks in the Binary Spectrum File” button is clicked, the system will automatically find wavelengths of each

peak in the record file and the data can be saved into an excel file. The graph of pulse data waveform can be plotted in excel file by selecting the data in column “Peak_WL1” and insert “scatter with smooth lines” which shown in Figure 3.6. This plotting method is too slow, thus a Python program is built to fasten up the plotting process.



Figure 3.2: FBGA.

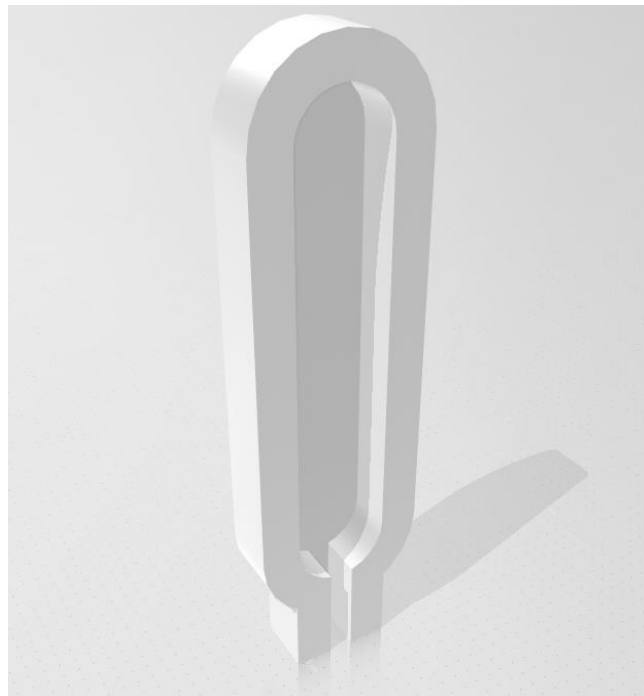


Figure 3.3: PLA cover. (Hew, 2019)

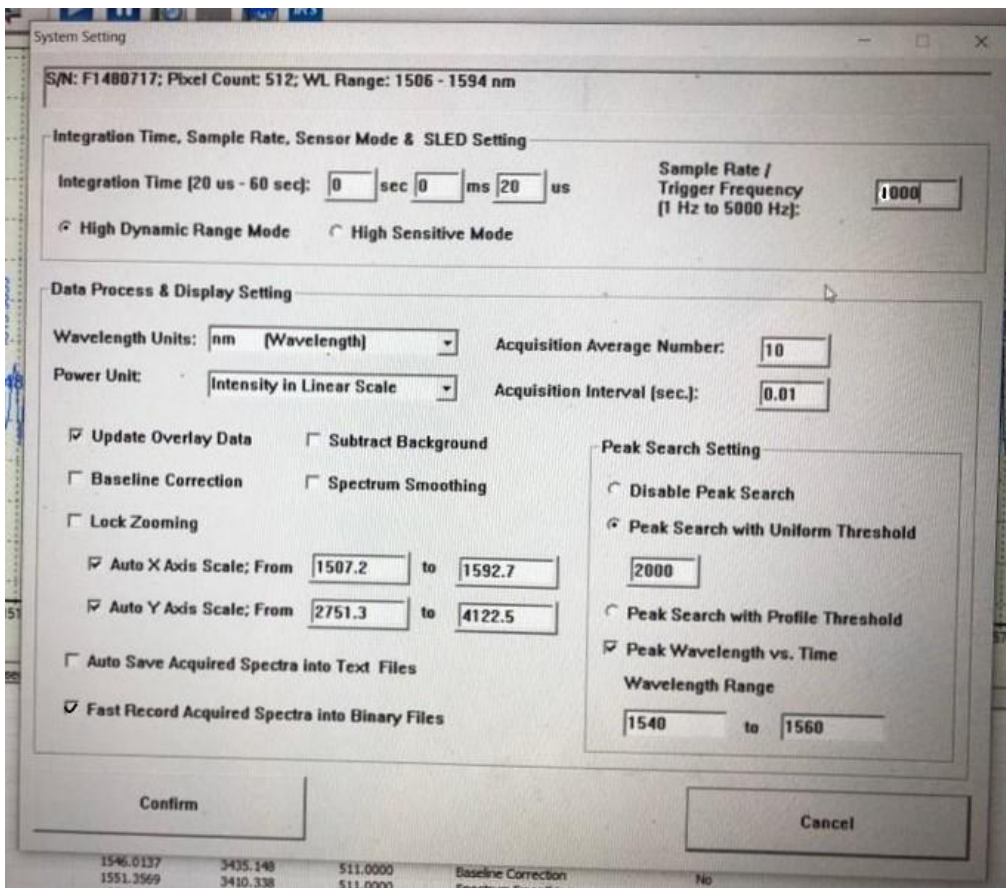


Figure 3.4: System setting of Sense 2020.

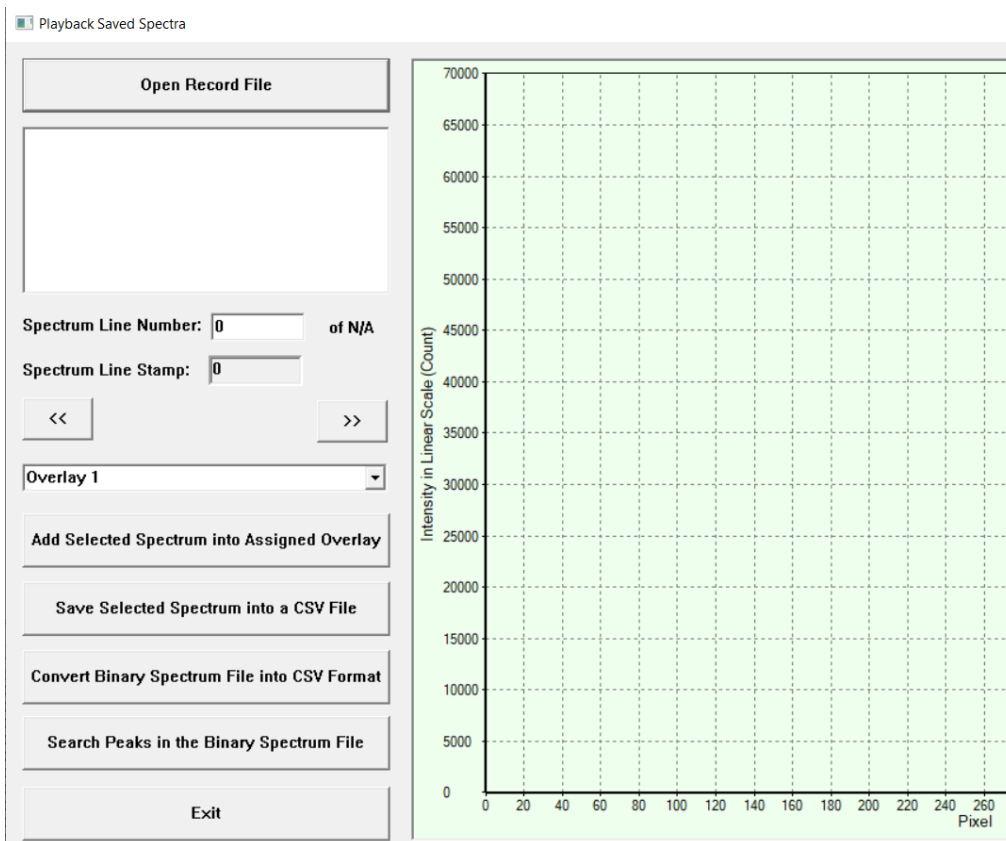


Figure 3.5: Playback saved spectra of Sense 2020.

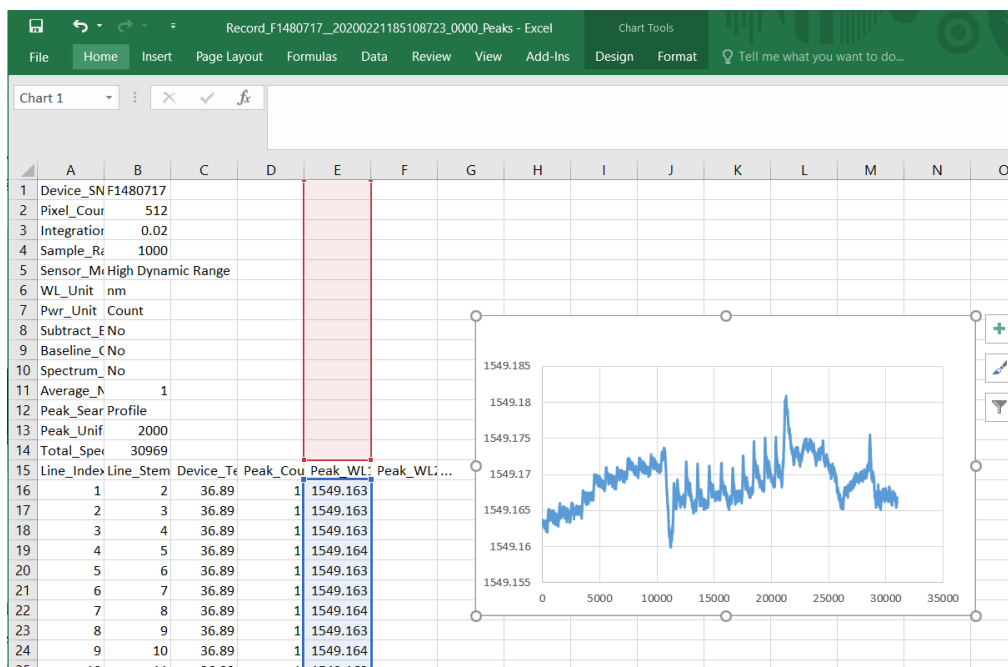


Figure 3.6: Graph of pulse data waveform plotted using Excel.

The original Python program to plot the graph was already built but there are some further modifications of the code was made in later. The modified Python script is attached in Appendix A. The Python script can be run in Windows Command Prompt (CMD) shown in Figure 3.7. The format of the command to run a Python script is “python filename.py” or “python3 filename.py” if more than one Python version was installed. Some errors encountered before are resolved by installing Python 3.8 in Microsoft Store. Moreover, a simple GUI toolkit that available in Python, Tkinter, is used in the coding. Examples of Tk GUI are shown in Figure 3.8. The Python program will automatically open the latest spectrum record excel file, the pulse waveform will be plotted in a pop-up window and ask user to choose a folder to save the excel file.

```

C:\> Command Prompt
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\ACER>python TCM1.py

```

Figure 3.7: Run Python script in CMD.

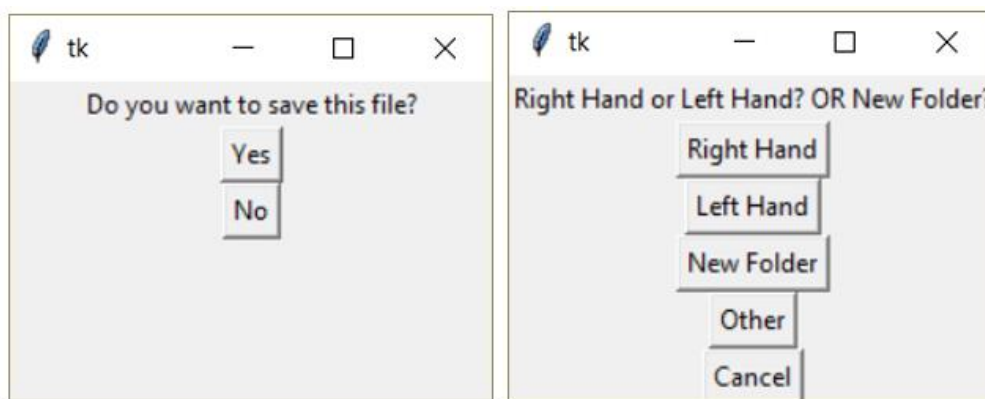


Figure 3.8: Tk GUI.

Before starting to collect pulse data, forms were prepared and printed. The form is used to record the health condition of a person when collecting the pulse data from the person. The template of the form is attached in Appendix B.

There are some rules needed to be followed by the participants to collect their pulse data; they need to put their hand on a pillow, stably fix their hand on the pillow, breath with a normal speed, and sit with a proper posture. The correct sitting posture is shown in Figure 3.9.

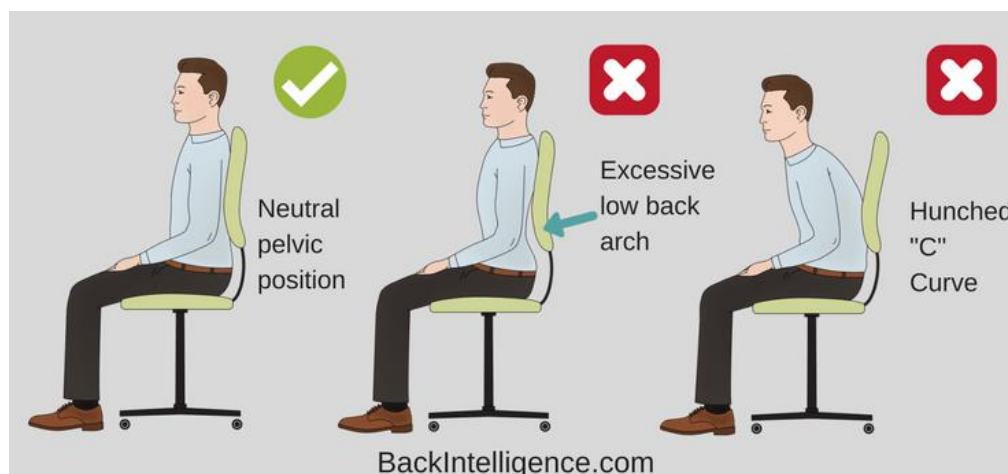


Figure 3.9: Correct and incorrect sitting posture.

3.3 Data Pre-processing

After the data are collected, some unsuitable or unacceptable data were excluded; this process is known as data selection. Examples of data need to be excluded are people who smoke a lot, people who had surgery on hands and people who are sick, such as fever, cough, and etc. This is because these kinds of data will affect its usability and preciseness to determine a person's sportability. For example, a sporty guy who has a slow pulse when he is in a healthy condition could have a quick pulse when he has a fever. In other words, the pulse of a sporty person who is sick will be affected and doped with a sick pulse's characteristic, hence, leading to a noisy characteristic in the data.

Since the collected data are in different folders, it is important to organise all the data into a single excel file. This is to reduce the overhead of opening each excel file multiple times to retrieve the data upon doing the machine learning later. Python can run more quickly and smoothly by just reading an excel file to retrieve all the pulse data at once. This also makes the data looks tidy. A Python script was coded to do this process semi-automatically. Before the data organised into an excel file, some other processes such as filtering and normalization will be performed. Figure 3.10 shows the initial part of Python codes to read and retrieve data from the excel files. A flowchart of the program is shown in Figure 3.11.

```

path = 'C:\\Users\\ACER\\Documents\\Usable Pulse Data\\1\\left hand\\Cun\\'

files = [f for f in glob.iglob(path + "*.csv", recursive=True)]

for f in files:
    print(f)
    with open(f) as f1:
        wavelength=list(csv.reader(f1,delimiter=","))
        N=np.array(wavelength[13:14])
        N=N[:,1]
        fs=np.array(wavelength[3:4])
        fs=fs[:,1]
        wavelength=wavelength[15:]
        wavelength=np.array(wavelength[0:],dtype=np.str)
        peak_wavelength=wavelength[:,4]

    f1.close()

sample_rate=1/int(fs)
End_time=len(peak_wavelength)*sample_rate
time=np.arange(0.,End_time,sample_rate)
if (len(time)!=len(peak_wavelength)):
    time=time[:-1]

```

Figure 3.10: Read and retrieve data from excel files.

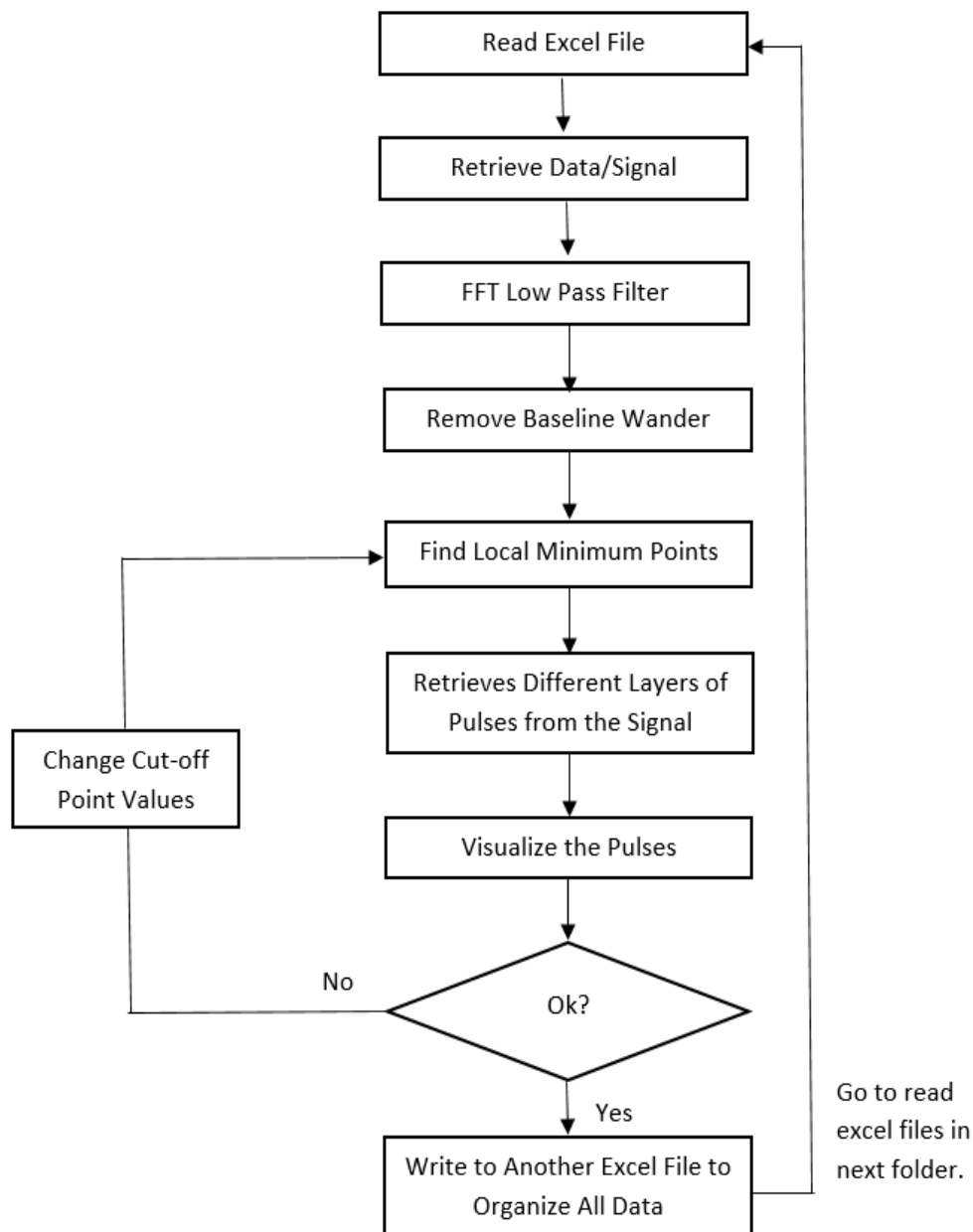


Figure 3.11: Flowchart of a Python program.

3.3.1 Filtering

It is important to filter the pulse data to eliminate the noise in the pulse. There are many methods can be used to filter the noise such as using MATLAB, Python, etc. The filtering process is done in this project through some Python codes shown in Figure 3.12. An open-source scientific library known as SciPy is used in the coding to perform complicated Fast Fourier Transform (FFT) mathematic calculation. FFT process a time-space signal to frequency-space signal to analyse frequency components of the signal. Based on the frequency

components obtained from FFT, the unwanted frequency such as noises of a signal can be observed and removed through a filter. A low-pass FFT filter cuts off high-frequency components above a limit but allows low-frequency components to pass through the filter. Based on the graph shown in Figure 3.13, the frequency of the signal is between -10 Hz to 10 Hz. Examples of the signal before and after filter are shown in Figure 3.14 and Figure 3.15 respectively.

```
#FFT LPF
peak_wavelength_1=peak_wavelength.astype(np.float)
sig_fft = scipy.fftpack.fft(peak_wavelength_1)
power = np.abs(sig_fft)
sample_freq = scipy.fftpack.fftfreq(peak_wavelength_1.size,sample_rate)
#Find the peak frequency
pos_mask = np.where(sample_freq > 10)
freqs = sample_freq[pos_mask]
peak_freq = freqs[power[pos_mask].argmax()]
#remove all the high frequencies
high_freq_fft = sig_fft.copy()
high_freq_fft[np.abs(sample_freq) > peak_freq] = 0
filtered_sig = scipy.fftpack.ifft(high_freq_fft)
```

Figure 3.12: FFT low-pass filter codes.

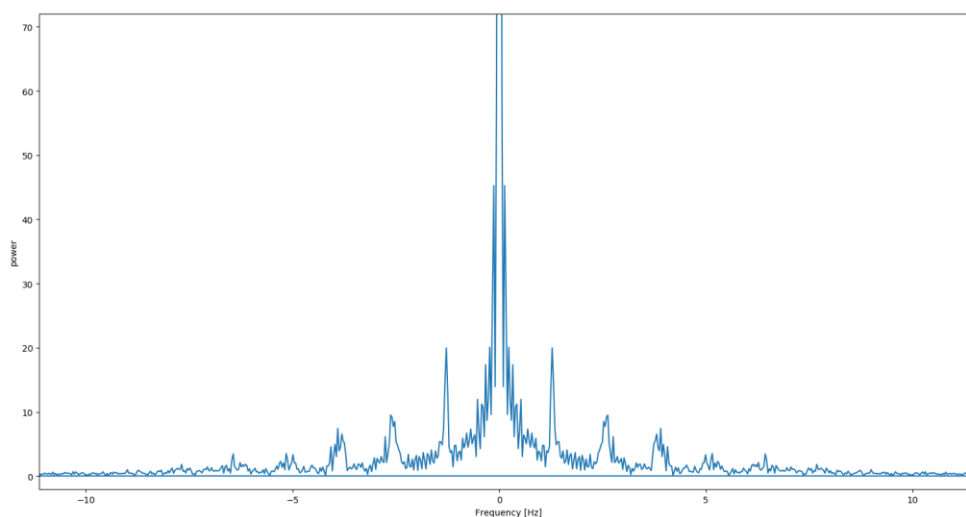


Figure 3.13: Zoom-in power against frequency graph.

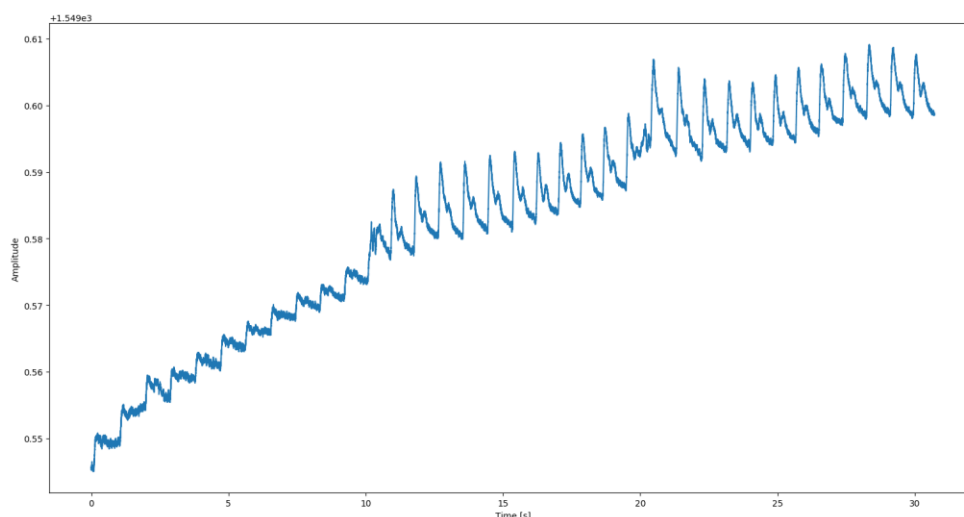


Figure 3.14: Example of an original signal.

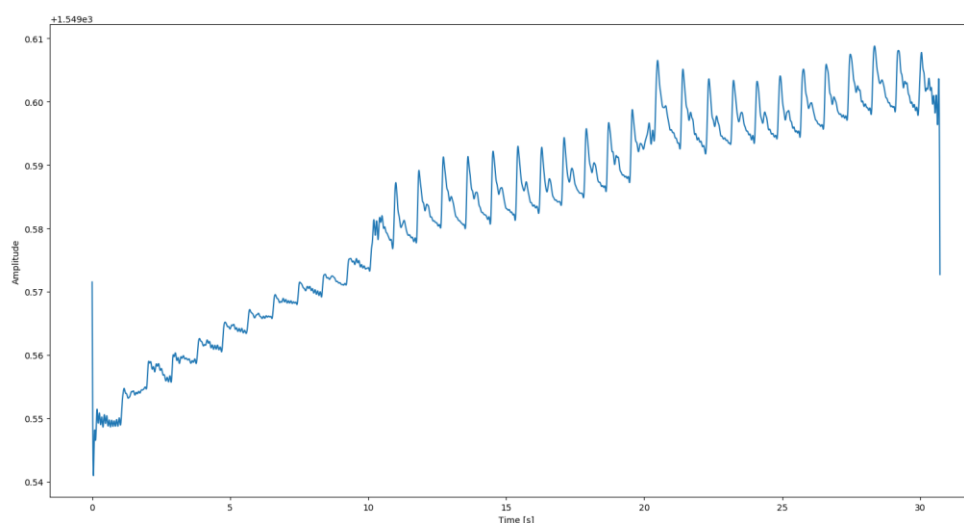


Figure 3.15: Example of a filtered signal.

3.3.2 Remove Baseline Wander

After done the filtering process, there is a need to remove baseline wander of the signal. Baseline wander is a low-frequency noise which can be eliminated through filters such as a high-pass filter (K. Brzostowski, 2016). A notch filter also able to be used for baseline drift removal of biomedical signals such as ECG (Luo et al., 2013). The python codes to remove baseline wander of a signal is shown in Figure 3.16. In this project, the “remove baseline wander” function from the *heartpy* library that implements a notch filter to filter the low-frequency noise. The low-frequency noise is near to zero which can be observed in Figure 3.17. Therefore, a value of 0.003 Hz is chosen to be the cutoff

frequency of the notch filter. Example of a signal after removed baseline wander is shown in Figure 3.18.

```
import heartpy as hp
x = hp.remove_baseline_wander(filtered_sig, 1000, cutoff=0.003)
```

Figure 3.16: Python codes of baseline wander removal.

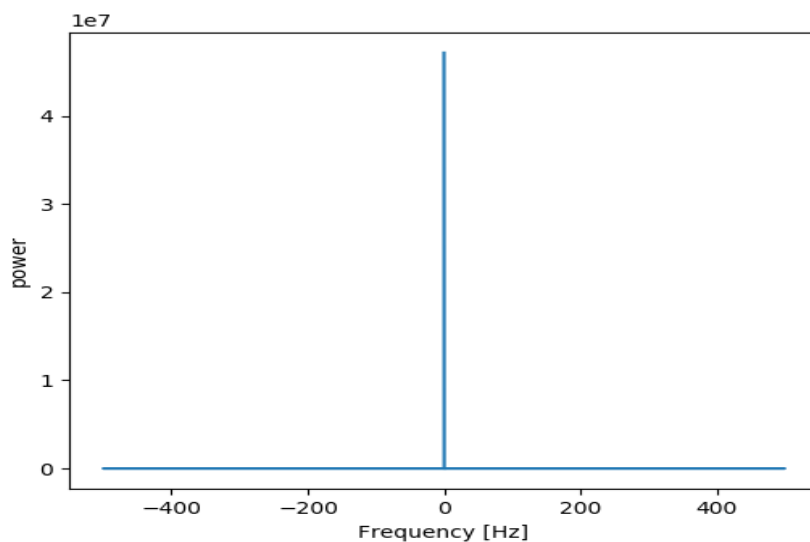


Figure 3.17: Power against frequency graph.

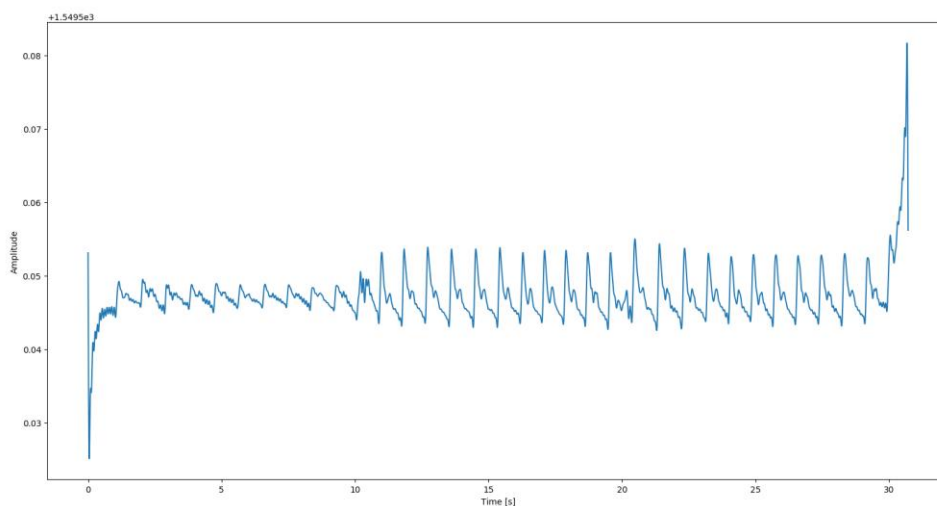


Figure 3.18: Example of a signal after removed baseline wander.

3.3.3 Cutting

Since the obtained signal containing pulses that are taken from the surface layer, middle layer and deep layer of the hand, the pulses can be further cut into three

pulse signals. The minimum points of a signal need to be found out to do the cutting process. So, the minimum points are found by inverting the signal and use a `find_peaks_cwt` function from Scipy library. A graph of a signal with minimum points is shown in Figure 3.19. Some minimum points are chosen to act as the starting points to cut the signal into three pulse signals data. Example of a signal after cutting process is shown in Figure 3.20. Then, the three pulse signals data will be written into an excel file named “Data.xlsx”.

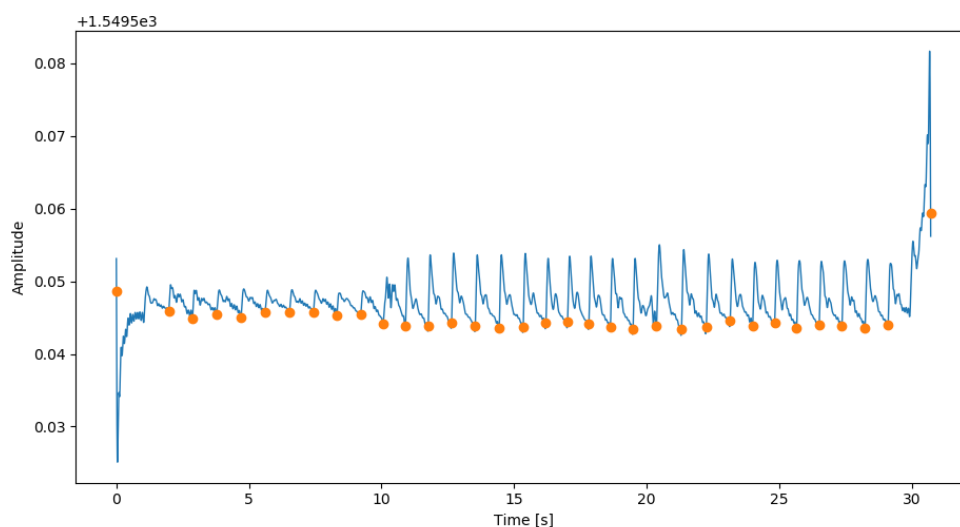


Figure 3.19: Example of minimum points of a signal.

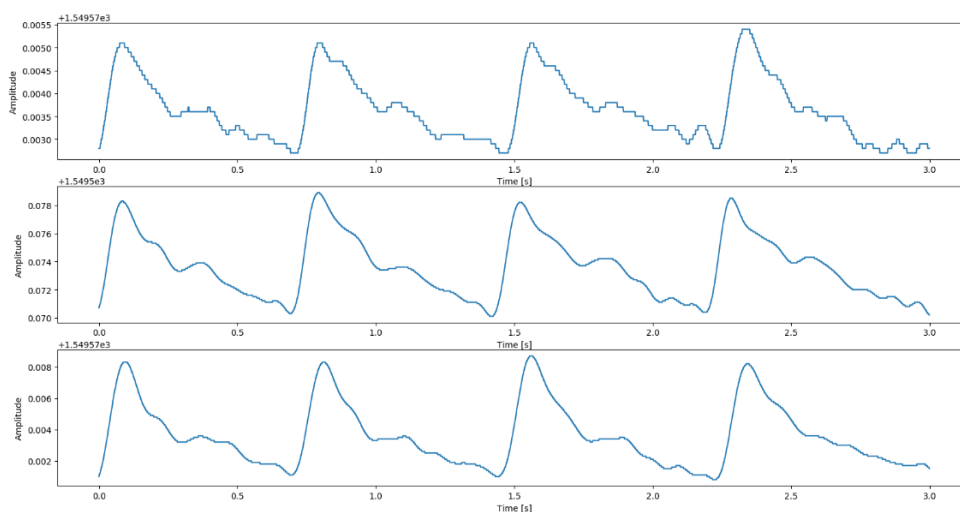


Figure 3.20: Example of a signal after the cutting process.

3.3.4 Labeling

Pulses that taken from left or right hand will be labelled when writing the pulse signals to “Data.xlsx”. ‘0’ represents right hand while ‘1’ represents left hand. The example of the data written in “Data.xlsx” is shown in Figure 3.21. After writing all the data in each excel file to Data.xlsx, the data are labelled into two categories which are sport and non-sport categories. ‘0’ will be represented as non-sport category while ‘1’ will be represented as sport category.

	A	B	C	D	E	F	G
1	Index	Waveform1	Waveform2	Waveform3	Hand	Sporty(1) / Not Sporty(0)	
2	1	1549.4074,1549.4	1549.4073,1549.4	1549.4069,1549.4	1	1	

Figure 3.21: Data written in “Data.xlsx”.

3.3.5 Normalizations

After that, another Python script is created to load the data at once from Data.xlsx and perform further normalizations such as offset and feature scaling. Since the original data have a value of 1549 and above, the data is offset by deducting the first point of the data with three decimal places from each data. This makes the data start from a value approximate to zero, not around 1549. Feature scaling on data is also done by using a standard scaler function from sklearn library. It transforms the data in a distribution with a mean value of zero and a standard deviation equal to one. Standardization can be used to improve the accuracy of machine learning classification models. The equation used to calculate the standard scores by the standard scaler function is shown in Eq (2.3).

$$\text{standard score, } Z = \frac{(\text{variable}, x - \text{mean}, u)}{\text{standard deviation}, s} \quad (3.1)$$

3.4 Data Splitting

Pulse data are taken three times Cun from each person to average the data by putting into the machine learning models and the total number of samples is 315 after processed data selection from 56 people. If the data is chosen based on Cun from each hand from a person, the number of samples is 108 samples after further data selection. A train_test_split function from sklearn library is used to

split the data into two data sets. The data samples are split into 75% training set to train the model and 25% test set used for validation or evaluation to test the performance of the machine learning models.

3.5 Machine learning algorithms

Sklearn which is also known as Scikit-learn, a machine learning library consists of many different algorithms used to build machine learning models. The machine learning algorithms used in this project are k-nearest neighbors (KNN), naïve Bayes, random forest, gradient boosting and support vector machine (SVM). The theory of KNN, random forest, gradient boosting and SVM were already discussed in Chapter 2.

naïve Bayes is implemented on the Bayes theorem to calculate posterior probability, $P(c|x)$. The equation of Bayes theorem is shown in (3.2). The c is represented as class and x is the value of a predictor. The probability of predictor given class is represented as $P(x|c)$ and it is calculated using the equation shown in Eq 3.3. The prediction will base on the calculated posterior probability of each class. For example, the calculated probability of female given a certain height, $P(\text{female}|\text{height})$ is 3.2 while the calculated probability of male given the height, $P(\text{male}|\text{height})$ is 5.5, then the predicted result will be male.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad (3.2)$$

$$P(x|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(x-\mu_c)^2}{2\sigma_c^2}} \quad (3.3)$$

The python codes to build each machine learning models are shown in Figure 3.22. The ‘p’ parameter is representing the distance function used in KNN classifier is Euclidean distance.


```

#KNN:-
print("KNN: ")
start = timeit.default_timer()
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 7, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)

#Naive Bayes:-
print("Naive Bayes: ")
start = timeit.default_timer()
from sklearn import naive_bayes
classifier = naive_bayes.GaussianNB()
classifier.fit(X_train, y_train)

#Random Forest:-
print("Random Forest: ")
start = timeit.default_timer()
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier()
classifier.fit(X_train, y_train)

#Gradient Boosting:-
print("Gradient Boosting: ")
start = timeit.default_timer()
from sklearn.ensemble import GradientBoostingClassifier as gbc
classifier = gbc()
classifier.fit(X_train, y_train)

#SVM:-
print("SVM: ")
start = timeit.default_timer()
from sklearn import svm
classifier = svm.SVC()
svc_model = classifier.fit(X_train, y_train)

```

Figure 3.22: Machine learning classifiers written in python codes.

3.6 Evaluation

There are various methods to evaluate the performance of a machine learning model. Common evaluation metrics used for classification are accuracy, confusion matrix, area under curve (AUC), F-measure and logarithmic loss. A confusion matrix is a matrix or table that presents and summarises the number of correct and incorrect predictions. The incorrect predictions are either false negatives (FN) or false positives (FP) which are the blue boxes shown in Figure 3.23 while the green boxes with true positives (TP) and true negatives (TN) are the correct predictions. Accuracy, precision, recall, specificity and F1-score can be calculated from the confusion matrix and the equations of each metrics are shown in Eq (3.4), Eq (3.5), Eq (3.6) and Eq (3.7) respectively. F1-score can represent both precision and recall by calculating harmonic mean between them

and the equation is shown in Eq (3.8). A good performance model should have a higher F1-score because a high precision and a high recall are required.

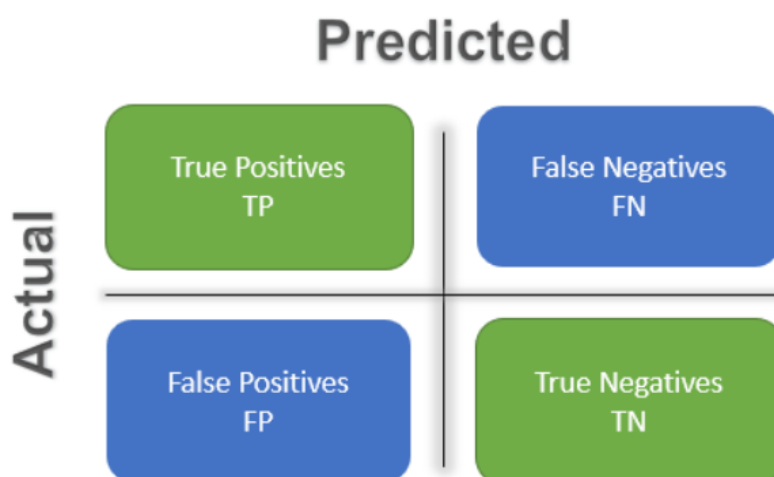


Figure 3.23: Confusion matrix.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \quad (3.4)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3.5)$$

$$\text{True Positive Rate or Recall} = \frac{TP}{TP + FN} \quad (3.6)$$

$$\text{False Positive Rate or Specificity} = \frac{TN}{TN + FP} \quad (3.7)$$

$$\text{F1 Score} = \frac{2(\text{Precision})(\text{Recall})}{\text{Precision} + \text{Recall}} \quad (3.8)$$

A Receiver Operating Characteristic (ROC) curve is a graph of true positive rate against false positive rate. Area under the curve (AUC) is the area under the ROC curve which has a value in a range of zero to one. An ideal classifier will have a value of AUC equal to one which means the ROC curve

will go along both y-axis and x-axis. This implies that the model able to differentiate the classes correctly at various thresholds settings. Logarithmic loss or log loss is the measure of the error of a machine learning model. If the log loss value decrease, the accuracy will increase.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

Students are the targeted group to collect their pulse data in this project. The number of participations who fulfilled the screening requirements illustrated in the previous chapter is 56. The shortest duration to collect pulse data of a person is 15 minutes. Sometimes, difficulties that occurred upon measuring certain people's pulse resulted in a longer collection time. Examples of difficulties mentioned here include people's bad sitting posture or sitting with a lot of movement, which eventually introduced unnecessary noise in the pulse signals.

Each pulse signal is measured at three different layers, i.e. shallow (浮), medium (中), and deep (沉), with each taking 3 seconds. Based on an article in "Medical News Today", a healthy adult has a normal breathing rate between 12 to 20 breaths in a minute. According to a book named "中医万问：常识篇", it stated that people who always do exercises such as an athlete or a healthy person might have "slow pulse" and a slow pulse means less than 4 pulses per breath. Assuming that a participant has 20 breaths per minute, a calculation can be conducted as shown below.

$$\frac{60 \text{ seconds}}{20 \text{ breaths}} = 3 \text{ seconds per breath}$$

Moreover, 10 columns are needed in an excel file to store the pulse data without the cutting process. This is because the maximum character can be stored in one cell of an excel file is limited to 32767 characters and the pulse data without cutting is 30 seconds, will leading to 30k data points. One data point has 9 characters which include 4 decimal places and the decimal point. A total of 10 characters is needed for one data point after including a comma ',' to separate each data points in the cell. Thus, the calculation is shown below to verify the numbers of columns needed to store 30k data points.

10 characters \times 30000 data points = 300000 characters

$$\text{columns needed} = \frac{300000 \text{ characters}}{32767 \text{ characters}} = 9.16 \approx 10$$

If the cutting process is done, the total number of data points is 9k which is equal to 90k characters. Thus, the number of columns needed to store the data in an excel file is approximate to 3 columns only. This can save a lot of memory space in an excel file, lesser noise contained in data and can process the data quickly upon training in a machine learning model.

4.2 Data Analysis

Data analysis was done based on a total of 108 pulse samples after data selection from 56 people both left and right “Cun” (寸). This is because left “Cun” is related to a human’s heart while right “Cun” is related to human’s lungs. This research can study the effect of exercising on human’s heart and lungs.

In the non-sport category, 37.70% of samples do not have deep layer pulse while only 2.13% of samples from sport category do not have deep layer pulse. This means that if a person who always exercise, there is a high chance will have pulses can be observed at the deep layer of their hand. A detailed number and percentage of samples with and without deep layer pulse is tabulated in Table 4.1.

Table 4.1: Number of samples with and without deep layer pulse.

		With deep layer pulse	Without deep layer pulse
Non-Sport	Left Cun	20	11
	Right Cun	18	12
	Total	38	23
	Percentage	62.3 %	37.7 %
Sport	Left Cun	24	0
	Right Cun	22	1
	Total	46	1
	Percentage	97.87 %	2.13 %

After excluding the data that do not have deep layer pulses, non-sport category left 38 samples and sport category left 46 samples. The number of pulses in 3 seconds of these samples are recorded and plotted in bar charts shown in Figure 4.1 and Figure 4.2. The number of pulses in 3 seconds are separated into a few divisions; i.e. 3, 3 to 4, 4 and 4 to 5. The frequency of 3 to 4 pulses in 3 seconds division of non-sport category is almost the same as the sport category while the frequency of 4 pulses in 3 seconds division of non-sport category is similar as the sport category. However, the non-sport category has a higher frequency than the sport category at 4 to 5 pulses in 3 seconds division. Sport category has a higher frequency than the non-sport category at 3 pulses in 3 seconds division. This validates the statement that people who always do exercises such as an athlete or a healthy person will have a “slow pulse”.

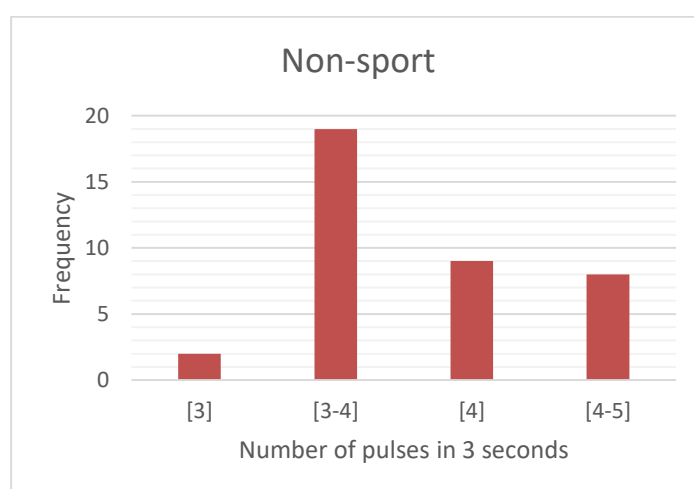


Figure 4.1: Number of pulses in 3 seconds of non-sport category.

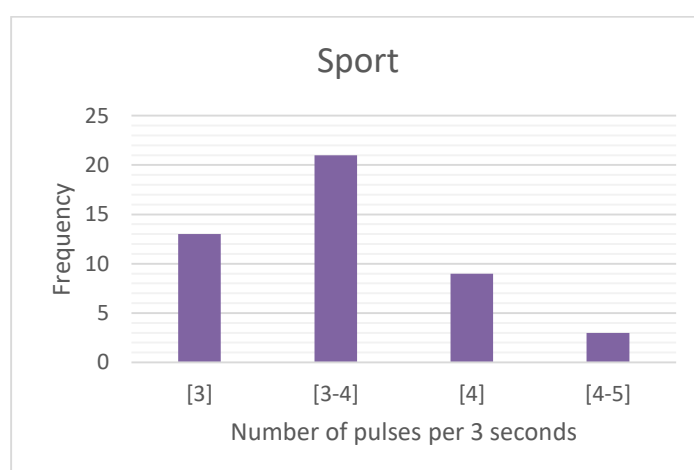


Figure 4.2: Number of pulses in 3 seconds of sport category.

Each category is further divided into 2 groups which are left Cun and right Cun. The bar charts of each group are shown in Figure 4.3, Figure 4.4, Figure 4.5 and Figure 4.6 respectively. The heights of pulse data from the left Cun of the non-sport category is compared with the heights of pulse data from the left Cun of the sport category. Comparison of heights of pulse data between right Cun from both categories is also analysed. The quantity of pulse data that have a height between 0.009 to 0.019 from left Cun of the sport category is significantly greater than the non-sport category. This is because people who always exercise have a stronger heart and left Cun is related to human's heart. However, there is no much distinct difference in heights of pulse data between right Cun of both categories.

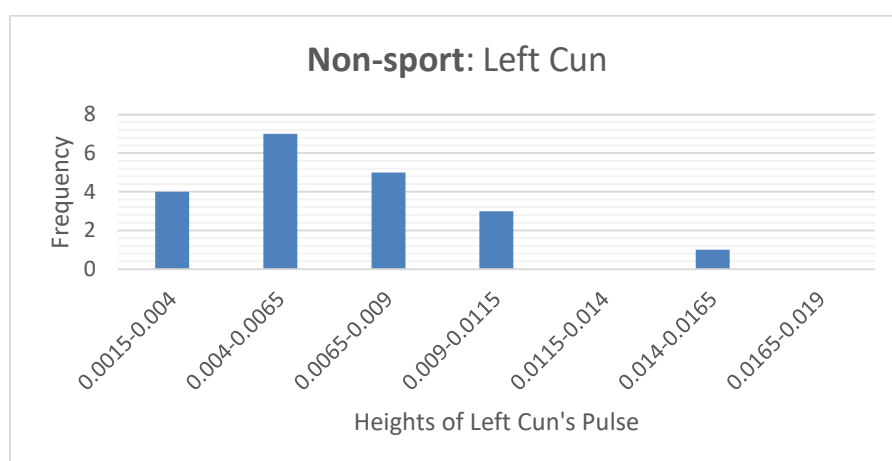


Figure 4.3: Graph of frequency versus heights of left Cun's pulse from non-sport category.

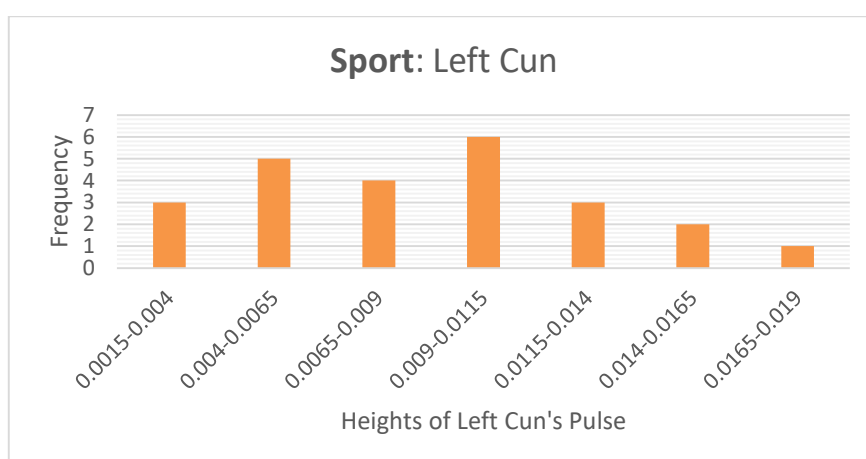


Figure 4.4: Graph of frequency versus heights of left Cun's pulse from sport category.

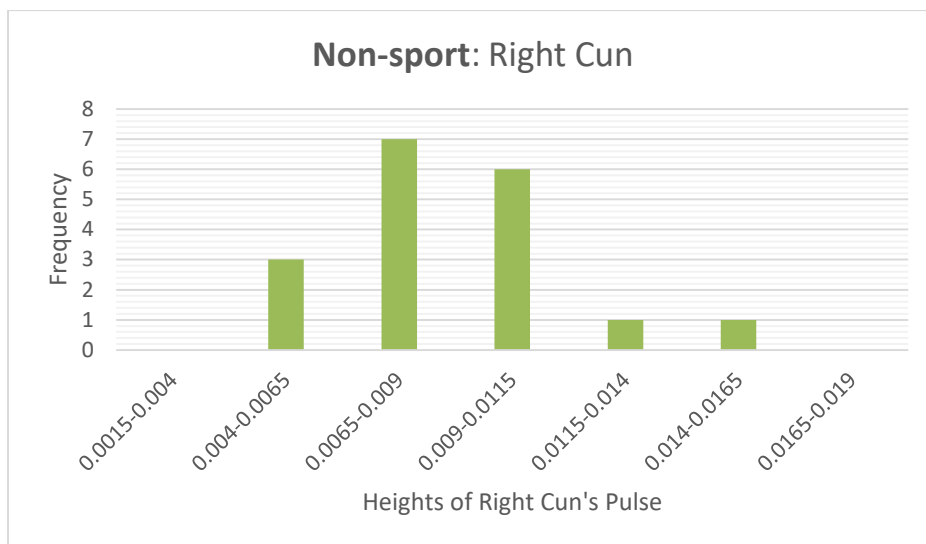


Figure 4.5: Graph of frequency versus heights of right Cun's pulse from non-sport category.

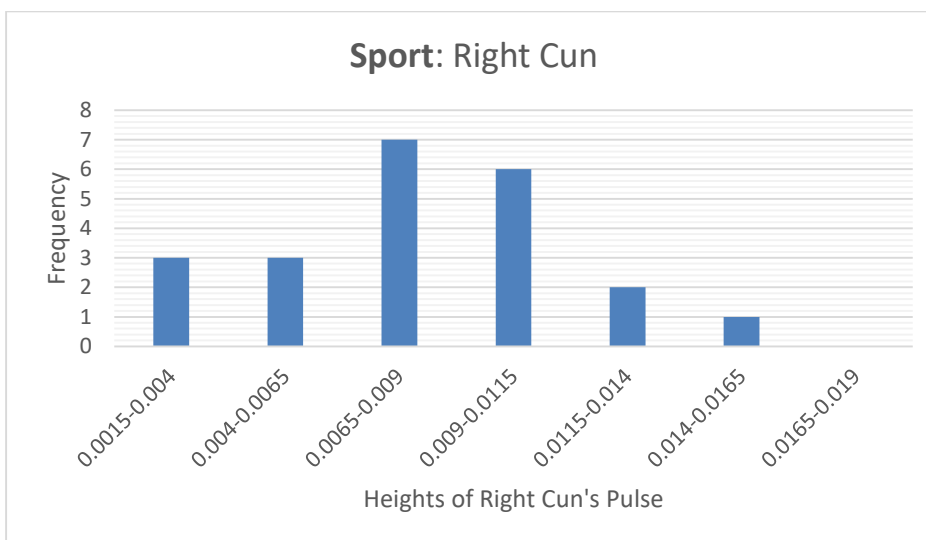


Figure 4.6: Graph of frequency versus heights of right Cun's pulse from sport category.

4.3 Machine Learning Models

The machine learning models used in this project are KNN, Naïve Bayes, Random Forest, Gradient Boosting and SVM. There are two sets of data samples which are 108 data samples and 315 data samples. Two data sets are trained in each machine learning models and make classifications.

4.3.1 Results and Discussions based on 108 Data Samples

These 108 data samples are fed into several machine learning models to train and make predictions. The results of each machine learning models are shown in Table 4.2. Example of a confusion matrix with labels is shown in Figure 4.7. All the ROC curves of each machine learning models are presented in the graph shown in Figure 4.8.

Table 4.2: Results of each machine learning models for 108 data samples.

Metrics Systems	Machine Learning Models				
	KNN	Naïve Bayes	Random Forest	Gradient Boosting	SVM
Accuracy, %	66.67	66.67	59.26	62.96	62.96
Confusion Matrix	$\begin{bmatrix} 12 & 4 \\ 5 & 6 \end{bmatrix}$	$\begin{bmatrix} 14 & 2 \\ 7 & 4 \end{bmatrix}$	$\begin{bmatrix} 12 & 4 \\ 7 & 4 \end{bmatrix}$	$\begin{bmatrix} 12 & 4 \\ 6 & 5 \end{bmatrix}$	$\begin{bmatrix} 15 & 1 \\ 9 & 2 \end{bmatrix}$
Precision, %	60.00	66.67	50.00	55.56	66.67
F1-score	0.57	0.47	0.42	0.50	0.29
AUC, %	71.59	62.78	63.35	67.05	72.16
Log Loss	11.51	11.51	14.07	12.79	12.79
Time, s	0.05	0.03	0.02	5.42	0.41

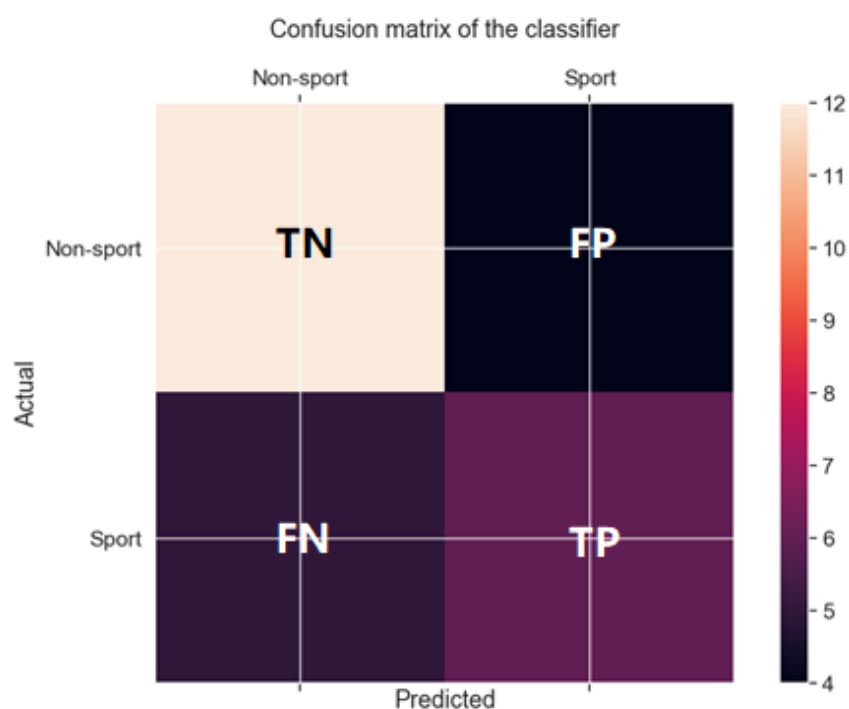


Figure 4.7: Confusion matrix with labels of KNN classifier.

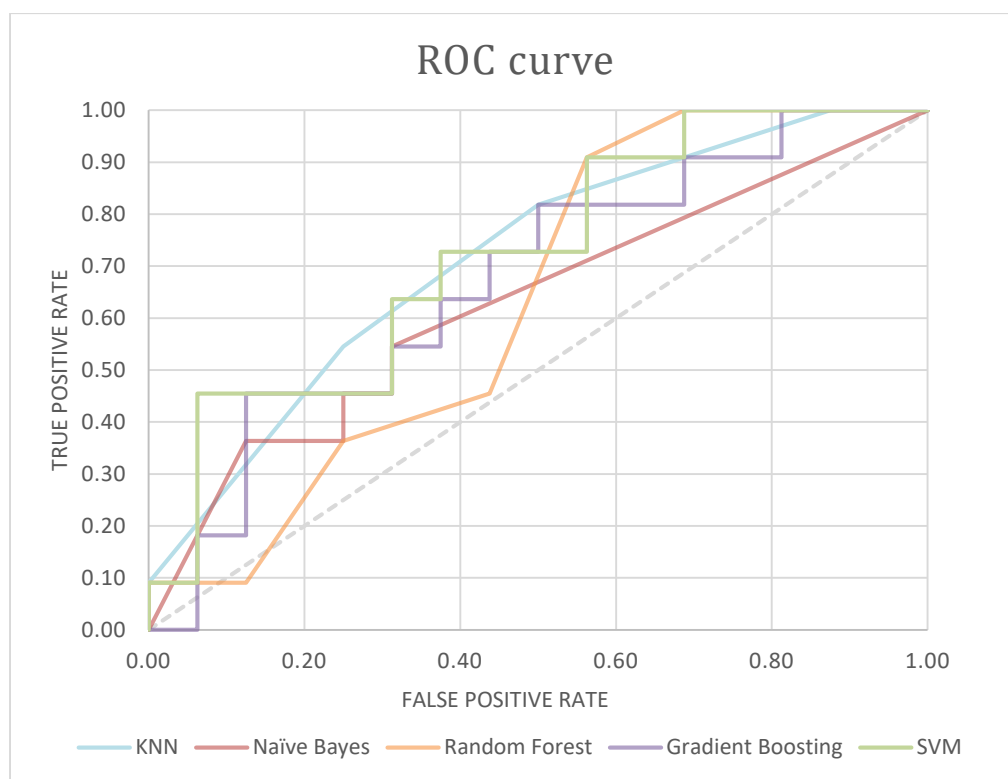


Figure 4.8: ROC curves of each machine learning models.

The machine learning models that have the highest accuracy with the lowest log loss are KNN and Naïve Bayes. The highest accuracy is 66.67% while the lowest log loss is 11.51. Naïve Bayes has a higher precision compared to KNN. However, Naïve Bayes has a lower F1-score and AUC percentage compared to KNN. This means that Naïve Bayes has a lower recall than KNN although it has higher precision. KNN has the highest F1-score which implies both precision and recall of KNN is better than the other models. SVM has the highest AUC percentage of 72.16% but it also has the lowest F1-score which is 0.29. Low recall of SVM can be aware as it has a high precision of 66.67% but low F1-score. Low recall of SVM indicates it has less correct predictions on sport category. KNN has the highest number of correct predictions on sport category which can be observed from its confusion matrix or F1-score. Furthermore, KNN has a smooth ROC curve compared to the others and has the second-highest AUC percentage which is 71.59%. In order words, it has a better capability to distinguish between categories. Therefore, KNN is the best machine learning model based on the results and comparison with other machine learning models for the data set.

4.3.2 Results and Discussions based on 315 Data Samples

These 315 data samples are fed into some machine learning models to train and classify the data. The results of each machine learning models are shown in Table 4.3: Results of each machine learning models for 315 data samples. Table 4.3. All the ROC curves of each machine learning models are presented in the graph shown in Figure 4.9.

Table 4.3: Results of each machine learning models for 315 data samples.

Metric Systems	Machine Learning Models				
	KNN	Naïve Bayes	Random Forest	Gradient Boosting	SVM
Accuracy, %	72.15	69.62	68.35	70.89	75.95
Confusion Matrix	$\begin{bmatrix} 37 & 7 \\ 15 & 20 \end{bmatrix}$	$\begin{bmatrix} 38 & 6 \\ 18 & 17 \end{bmatrix}$	$\begin{bmatrix} 35 & 9 \\ 16 & 19 \end{bmatrix}$	$\begin{bmatrix} 34 & 10 \\ 13 & 22 \end{bmatrix}$	$\begin{bmatrix} 40 & 4 \\ 15 & 20 \end{bmatrix}$
Precision, %	74.07	73.91	67.85	68.75	83.33
F1-score	0.65	0.59	0.60	0.66	0.68
AUC, %	72.34	69.25	69.97	75.71	78.12
Log Loss	9.62	10.49	10.93	10.06	8.31
Time, s	0.76	0.09	0.22	14.55	3.21

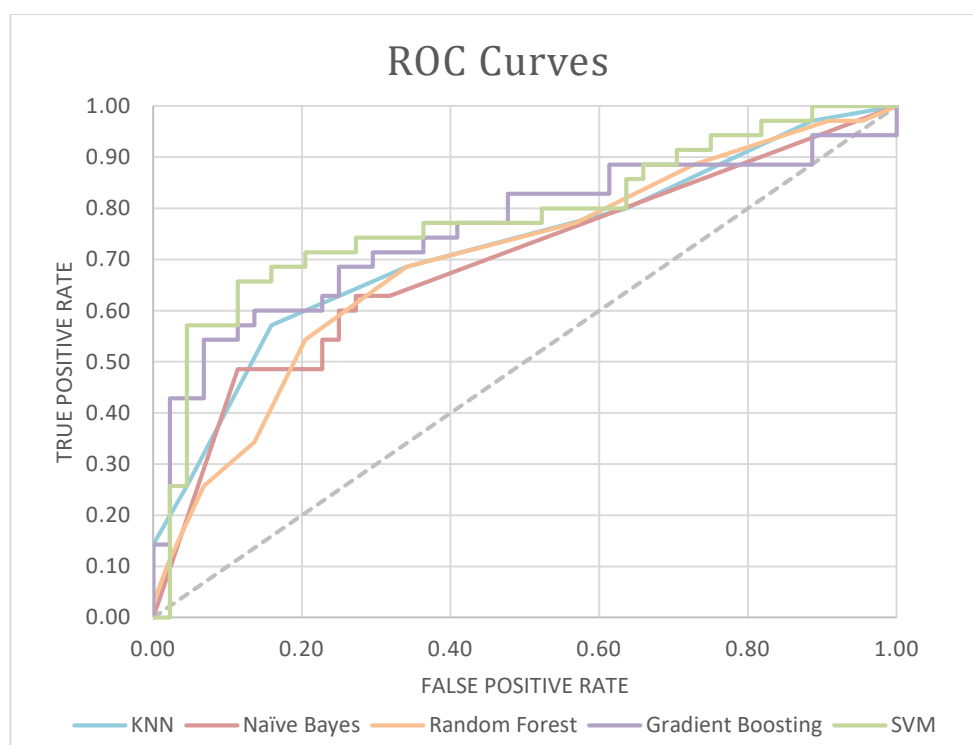


Figure 4.9: ROC curves of each machine learning models.

SVM has the highest accuracy which is 75.95% while random forest has the lowest accuracy which is 68.35% among other machine learning models. Although Naïve Bayes has higher accuracy and precision than the random forest, Naïve Bayes has a slightly lower F1-score and AUC percentage than the random forest which implies that Naïve Bayes has a lesser correct quantity in classifying the data as sport category. This condition can also be observed from the confusion matrix, ROC curves of Naïve Bayes, and random forest. Although gradient boosting has the most correct number in distinguishing the data as sport class, it also has the least correct amount in categorising the data as non-sport class and this can be discerned by comparing the confusion matrixes of each machine learning models. KNN has the same amount of true positives with SVM but the other metric system's results of KNN are lower than SVM. The precision, F1-score and AUC percentage of SVM are the highest and it also has the lowest log loss. SVM also has a better ROC curve compared to others. Thus, SVM has a good performance in classifying between categories for the data set.

4.4 Compare both results between two data sets

When the amount of data raises, the accuracy of machine learning models also increases. This is because adding more data can increase the diversity of the samples and decrease the chance of overfitting which leads to a better model is created. Moreover, the time to train the data using gradient boosting for both data sets is the highest. This is due to gradient boosting will need to perform gradient descent algorithms which a new tree will be added to the model each iteration to reduce the loss while the existing trees in the model have remained unchanged so it takes longer duration to train the data. According to the results, the best machine learning model to be used for 108 data samples is KNN but the best machine learning model to be used for 315 data samples is SVM.

4.5 Possible Reasons for Low Accuracy of the Results

Firstly, the participants that classified as sport category claimed themselves do exercises regularly but there is no evidence to validate the truth of their claims which will influence the accurateness of the labelled data. In addition, the participants are chosen randomly which they have a different constitution or body mass index (BMI). This may increase the diversity of the data but it also

has a probability to become an outlier. Besides, the best time to do pulse diagnosis is early morning before breakfast (吕文曾, 2008). According to midnight-noon and ebb-flow doctrine (子午流注), health physical condition of a human will have different at different hours. Since the physical condition will change at different hours, the pulse will also have changes. However, there is no fixed time to take the pulse from participants in this project and the pulses taken from each hand of a participant are different too. This may become a hidden influence on the correctness of the obtained data. Lastly, the coronavirus pandemic impacts the process of data collection. Precaution is applied by providing hand sanitizer to the participants and some candies are prepared to attract volunteers but the number of volunteers is less than expected and results in fewer data are collected.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

In this project, different machine learning models for pulse data classification had been understood by studying their theories and implementing various machine learning algorithms. The implemented machine learning algorithms to train and classify the pulse data using Python are k-nearest neighbors (KNN), naïve Bayes, random forest, gradient boosting and support vector machine (SVM). Theories about each machine learning model had been discussed in this report. Steps such as data pre-processing, labelling and data splitting were applied in order to build the machine learning models. Some pulse data had been analysed and found out that the pulse data of people who always do exercises will have 3 or 3 to 4 number of pulses occurred in 3 seconds. They also usually have a higher pulse's height of their left hand.

The collected TCM pulse data had been trained with the machine learning models and automatically classify the test data set into two classes which are non-sport category and sport category. The non-sport category means people who not active in exercising and vice versa for sport category. The generated results of each model are presented and discussed in the report. The objective of evaluating the accuracy of each machine learning model was done by comparing various metric systems between the models. The metric systems are accuracy, confusion matrix, precision, F1-score, percentage of AUC, ROC curves, log loss and the costed time to train the data. As the data sample increase, the accuracy of each model also increases. Random forest has the lowest accuracy among the machine learning models although it also shows an increase in accuracy when the data sample increased. The highest accuracy to automatically classify the TCM pulse into non-sport or sport categories is 75.95% using SVM model that trained based on 315 data samples. This satisfies the other aim which is to classify if a person is active in exercising or not through his or her TCM pulses using machine learning.

5.2 Recommendations for future work

Some suggestions are listed below to improve future work that wants to classify different types of TCM pulse data.

1. A group of targeted people needed to be fixed so that the pulse can be taken from the same group of people and able to monitor the changes of their pulse each day. For example, if want to classify or predict whether a female is menstruating, before her menstrual period or after her period, it is important to obtain her TCM pulses data each day as different people will have a different menstrual cycle length.
2. BMI of the targeted people needed to be in the same range or the BMI of each targeted person needed to be recorded and becomes an important variable to train and classify the data. This is because BMI can be a big factor to affect the position of the person's pulse (surface, middle or deep).
3. The time to collect people's pulse data needed to be fixed due to midnight-noon and ebb-flow doctrine. As a suggestion, the time can be separated into the morning session, afternoon session, evening session and night session. One of the timeframes is selected and fixed to collect pulse data.
4. The pulse-taking system is required to be further enhanced to obtain a more stable pulse data and able to speed up the pulse-taking process from each hand of a person.
5. TCM practitioners are needed to help and cooperate in this kind of project to verify the pulse type of each collected pulse data. This also can attract volunteers as they can know their health conditions from TCM practitioners and their pulse data can be obtained for the project which will result in a win-win situation. Students who study TCM can be considered to cooperate with them which they can get more experience in pulse diagnosis while at the same time the pulse data can be verified.

REFERENCES

- Anon, *Fuzzy C-Means Clustering* [Online]. Available at: https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/cmeans.html.
- Anon, *K Nearest Neighbors - Classification* [Online]. Available at: https://www.saedsayad.com/k_nearest_neighbors.htm.
- Anon, 2015. 复旦研制出机器人“中医一号.” 医药前沿, 5(10), p.3.
- Bronshtein, A., 2017, *A Quick Introduction to K-Nearest Neighbors Algorithm* [Online]. Available at: <https://blog.usejournal.com/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7>.
- Burden, F. and Winkler, D., 2008, *Bayesian regularization of neural networks*. [Online]. Available at: <https://www.ncbi.nlm.nih.gov/pubmed/19065804>.
- Eremenko, K. and Ponteves, H. de, *Machine Learning A-Z™: Hands-On Python & R In Data Science* [Online]. Available at: <https://www.udemy.com/machinelearning/>.
- FENG, B. and Li, S., 2018. Unsupervised clustering analysis of human-pulse signal in traditional Chinese medicine. *CAAI Transactions on Intelligent System*, 13, pp.564–570.
- Gandhi, R., 2018, *Support Vector Machine — Introduction to Machine Learning Algorithms* [Online]. Available at: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>.
- Guo, R. et al., 2015. Analysis and recognition of traditional chinese medicine pulse based on the hilbert-huang transform and random forest in patients with coronary heart disease. *Evidence-based Complementary and Alternative Medicine*, 2015.
- Hew, P.C., 2019. *Development of Fiber Optic Sensor based TCM Pulse Monitoring System*. Universiti Tunku Abdul Rahman.
- Jennifer Dubowsky, La., 2012, *Pulse Power: Understanding Chinese Pulse Diagnosis* [Online]. Available at: <https://qiblog.emperors.edu/2014/09/pulse-power-understanding-tcm-pulse-diagnosis/> [Accessed: 14 August 2019].
- Josh, 2015, *Everything You Need to Know About Artificial Neural Networks* [Online]. Available at: <https://medium.com/technology-invention-and-more/everything-you-need-to-know-about-artificial-neural-networks-57fac18245a1>.

Jung, H., 2018, *Adaboost for Dummies: Breaking Down the Math (and its Equations) into Simple Terms* [Online]. Available at: <https://towardsdatascience.com/adaboost-for-dummies-breaking-down-the-math-and-its-equations-into-simple-terms-87f439757dcf>.

Luo, Y. et al., 2013. A Hierarchical Method for Removal of Baseline Drift from Biomedical Signals : Application in ECG Analysis. , 2013.

Luo, Z.Y. et al., 2018. A Study of Machine-Learning Classifiers for Hypertension Based on Radial Pulse Wave. *BioMed Research International*, 2018.

Maciocia, G., 1989. *The Foundations of Chinese Medicine: A Comprehensive Text for Acupuncturists and Herbalists.*,

MOAWAD, A., 2018, *Neural networks and back-propagation explained in a simple way* [Online]. Available at: <https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e>.

Pant, A., *Introduction to Machine Learning for Beginners* [Online]. Available at: <https://towardsdatascience.com/introduction-to-machine-learning-for-beginners-eed6024fdb08>.

Parthasarathy, S., *Difference between Traditional programming versus Machine Learning from a PM perspective* [Online]. Available at: <https://productcoalition.com/difference-between-traditional-programming-versus-machine-learning-from-a-pm-perspective-3802b02bc7f6>.

RAY, S., 2017, *Understanding Support Vector Machine algorithm from examples (along with code)* [Online]. Available at: <https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/>.

SHARMA, S., 2017, *Activation Functions in Neural Networks* [Online]. Available at: <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>.

Tagliaferri, L., 2017, *An Introduction to Machine Learning* [Online]. Available at: <https://www.digitalocean.com/community/tutorials/an-introduction-to-machine-learning#machine-learning-methods>.

Tang, A.C.Y., Chung, J.W.Y. and Wong, T.K.S., 2012. Validation of a novel traditional Chinese medicine pulse diagnostic model using an artificial neural network. *Evidence-based Complementary and Alternative Medicine*, 2012.

Trevino, A., 2016, *Introduction to K-means Clustering* [Online]. Available at:

<https://www.datascience.com/blog/k-means-clustering>.

Wang, G. et al., 2010. On intrinsic mode function. *Advances in Adaptive Data Analysis*, 2(3), pp.277–293.

Wang, H. and Zhang, P., 2009. A quantitative method for pulse strength classification based on decision tree. *Journal of Software*, 4(4), pp.323–330.

Wang, Y.Y., Wang, S.H., Jan, M.Y. and Wang, W.K., 2012. Past, present, and future of the pulse examination (mài zhen). *Journal of Traditional and Complementary Medicine*, 2(3), pp.164–177.

Wilhelm, B. et al., 2007. Increased arterial augmentation and augmentation index as surrogate parameters for arteriosclerosis in subjects with diabetes mellitus and nondiabetic subjects with cardiovascular disease. *Journal of Diabetes Science and Technology*, 1(2), pp.260–263.

Yan Tang, A.C., 2012. Review of Traditional Chinese Medicine Pulse Diagnosis Quantification. *Complementary Therapies for the Contemporary Healthcare*.

Zhang, Y. and Haghani, A., 2015. A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies*, 58, pp.308–324. Available at: <https://www.sciencedirect.com/science/article/pii/S0968090X15000741>.

Zhang, Z., 2010. 脉象信息分析和识别方法评析. *中国中药信息杂志*, pp.1–4.

Zuo, W. et al., 2010. Classification of pulse waveforms using edit distance with real penalty. *Eurasip Journal on Advances in Signal Processing*, 2010.

朱文锋, 2013. 中医诊断学 (新世纪第二版), 中国中医药出版社.

APPENDICES

APPENDIX A: Python codes to plot and save pulse data from an excel file.

```

import csv
import numpy as np
import matplotlib.pyplot as plt
import scipy.fftpack
import os
import glob
import shutil
import tkinter as tk
from tkinter import simpledialog

list_of_files=glob.glob('C:/Users/ACER/Documents/1.FYP - bayspec/Bayspec/Spectrum_Data/vibration/*.csv')
latest_file=max(list_of_files,key=os.path.getctime)

with open(latest_file) as f1:
    wavelength=list(csv.reader(f1,delimiter=","))
    fs=np.array(wavelength[3:4])
    fs=fs[:,1]
    wavelength=wavelength[15:]
    wavelength=np.array(wavelength[0:],dtype=np.str)
    peak_wavelength=wavelength[:,4]

f1.close()

sample_rate=1/int(fs)
End_time=len(peak_wavelength)*sample_rate
time=np.arange(0.,End_time,sample_rate)
if (len(time)!=len(peak_wavelength)):
    time=time[:-1]

peak_wavelength_1=peak_wavelength.astype(np.float)

plt.figure()
plt.xlabel('time,s')
plt.ylabel('wavelength,nm')
plt.title('Subject TCM Pulse waveform')
plt.plot(time,peak_wavelength_1, '#E0301',label='raw data')
plt.show()

def raise_frame(frame):
    frame.tkraise()

root = tk.Tk()

frame1 = tk.Frame(root)
frame1.pack(fill=tk.BOTH, expand=True)
frame2 = tk.Frame(root)
frame3 = tk.Frame(root)
frame4 = tk.Frame(root)

def CheckFolder(newpath):
    if not os.path.exists(newpath):
        os.makedirs(newpath)
    else:
        print("file ady exist.")

def clickButtonOther():
    new_path="C:/Users/ACER/Documents/Usable Pulse Data"
    shutil.move(latest_file,new_path)
    root.destroy()

def clickButton2():
    root.destroy()

```

APPENDIX A (continued)

```

def clickButtonRCun():
    directory = "C:\\Users\\ACER\\Documents\\Usable Pulse Data"
    right_path = max([os.path.join(directory,d) for d in os.listdir(directory)], key=os.path.getmtime) + "\\right hand\\Cun"
    print(right_path)
    shutil.move(latest_file,right_path)
    root.destroy()

def clickButtonRGuan():
    directory = "C:\\Users\\ACER\\Documents\\Usable Pulse Data"
    right_path = max([os.path.join(directory,d) for d in os.listdir(directory)], key=os.path.getmtime) + "\\right hand\\Guan"
    print(right_path)
    shutil.move(latest_file,right_path)
    root.destroy()

def clickButtonRChi():
    directory = "C:\\Users\\ACER\\Documents\\Usable Pulse Data"
    right_path = max([os.path.join(directory,d) for d in os.listdir(directory)], key=os.path.getmtime) + "\\right hand\\Chi"
    print(right_path)
    shutil.move(latest_file,right_path)
    root.destroy()

def clickButtonLCun():
    directory = "C:\\Users\\ACER\\Documents\\Usable Pulse Data"
    left_path = max([os.path.join(directory,d) for d in os.listdir(directory)], key=os.path.getmtime) + "\\left hand\\Cun"
    print(left_path)
    shutil.move(latest_file,left_path)
    root.destroy()

def clickButtonLGuan():
    directory = "C:\\Users\\ACER\\Documents\\Usable Pulse Data"
    left_path = max([os.path.join(directory,d) for d in os.listdir(directory)], key=os.path.getmtime) + "\\left hand\\Guan"
    print(left_path)
    shutil.move(latest_file,left_path)
    root.destroy()

def clickButtonLChi():
    directory = "C:\\Users\\ACER\\Documents\\Usable Pulse Data"
    left_path = max([os.path.join(directory,d) for d in os.listdir(directory)], key=os.path.getmtime) + "\\left hand\\Chi"
    print(left_path)
    shutil.move(latest_file,left_path)
    root.destroy()

def clickButtonNew():
    x = simpledialog.askstring(title = "Create New Folder", prompt="Key in new folder name:")
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\right hand"
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\left hand"
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\right hand\\Cun"
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\right hand\\Guan"
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\right hand\\Chi"
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\left hand\\Cun"
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\left hand\\Guan"
    CheckFolder(newpath)
    newpath = "C:\\Users\\ACER\\Documents\\Usable Pulse Data\\" + x + "\\left hand\\Chi"
    CheckFolder(newpath)

for frame in (frame1, frame2, frame3, frame4):
    frame.grid(row=0, column=0, sticky='news')

tk.Label(frame1, text = "Do you want to save this file?").pack()
tk.Button(frame1, text="Yes", command=lambda: raise_frame(frame2)).pack() #raise frame 2
tk.Button(frame1, text="No", command=clickButton2).pack()

tk.Label(frame2, text = "Right Hand or Left Hand? OR New Folder?").pack()
tk.Button(frame2, text = "Right Hand", command =lambda: raise_frame(frame3)).pack() #raise frame 3
tk.Button(frame2, text = "Left Hand", command =lambda: raise_frame(frame4)).pack() #raise frame 4
tk.Button(frame2, text = "New Folder", command = clickButtonNew).pack()
tk.Button(frame2, text = "Other", command = clickButtonOther).pack()
tk.Button(frame2, text = "Cancel", command = clickButton2).pack()

tk.Label(frame3, text = "Cun, Guan or Chi?(Right Hand)").pack()
tk.Button(frame3, text = "Cun", command = clickButtonRCun).pack()
tk.Button(frame3, text = "Guan", command = clickButtonRGuan).pack()
tk.Button(frame3, text = "Chi", command = clickButtonRChi).pack()
tk.Button(frame3, text = "Cancel", command = clickButton2).pack()

tk.Label(frame4, text = "Cun, Guan or Chi?(Left Hand)").pack()
tk.Button(frame4, text = "Cun", command = clickButtonLCun).pack()
tk.Button(frame4, text = "Guan", command = clickButtonLGuan).pack()
tk.Button(frame4, text = "Chi", command = clickButtonLChi).pack()
tk.Button(frame4, text = "Cancel", command = clickButton2).pack()

raise_frame(frame1)
root.mainloop()

```

APPENDIX B: Data Collection Form

No: 

UNIVERSITI TUNKU ABDUL RAHMAN

中医脉象数据采集表格
TCM Pulse Diagnosis Data Collection Form

尊敬的先生/女士,

我们是来自拉曼大学 (UTAR) 的学生。我们目前正在做一个名为“基于机器学习的中医脉诊”的研究项目。因此，我们需要收集大量脉象数据样本以进行研究。我们致力于保护您所提供的数据以及个人资料，并承诺所有资料仅作为研究用途。非常感谢您的帮助。

Dear Sir/Madam,

We are students from Universiti Tunku Abdul Rahman (UTAR). We are currently doing a project entitled “DESIGN OF PREDICTIVE MODEL FOR TCM PULSE DIAGNOSIS IN MALAYSIA USING MACHINE LEARNING”. Therefore, we need to collect an abundance of pulse data for research purpose. We are committed to protecting the data samples and personal information you provide, and we promise that all data will be used for research purposes only. Your help is greatly appreciated.

年龄 Age: _____

性别 Gender: 男 Male 女 Female

身体健康状况 Health Condition:

 健康 Healthy 发烧 Fever 咳嗽 Cough 伤风 Cold 熬夜/迟睡 Stay up late – Time: _____ 失眠 Insomnia 经期 On period – Day: _____ 痛经 Period pain 月经不调 Irregular menstruation 月经前 Before period 月经后 After period 贫血 Anaemia 运动量 Duration of Exercise: -Days: _____ 没有运动 No Exercise

-Hrs/mins: _____

其他 Other: _____

签名 Signature: _____